# Decentralizovaná Crowdfundingová Aplikace na Solana Blockchainu

Bc. David Bilnica

Diplomová práce
2024

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2023/2024

# ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

| | |
|---|---|
| Jméno a příjmení: | **Bc. David Bilnica** |
| Osobní číslo: | **A22290** |
| Studijní program: | **N0613A140022 Informační technologie** |
| Specializace: | **Softwarové inženýrství** |
| Forma studia: | **Prezenční** |
| Téma práce: | **Decentralizovaná Crowdfundingová Aplikace na Solana Blockchainu** |
| Téma práce anglicky: | **Decentralized Crowdfunding Application on the Solana Blockchain** |

## Zásady pro vypracování

1. Specifikujte požadované funkce vyvíjené crowdfundingové aplikace.
2. Analyzujte rozdíly mezi Solana a Ethereum Blockchainem.
3. Navrhněte aplikační řešení crowdfundingové aplikace.
4. Navržený systém implementujte v testovacím prostředí.
5. Ověřte funkčnost a zabezpečení vašeho řešení.

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:


1. TAPSCOTT, Don a TAPSCOTT, Alex. *Blockchain revolution*. [London]: Portfolio/Penguin, 2018. ISBN 978-0-241-23786-1.
2. BASHIR, Imran. Mastering blockchain. Birmingham: Packt Publishing, Limited, 2020. ISBN 9781839211379.
3. BAMBARA, Joseph J. a ALLEN, Paul R. Blockchain: A Practical Guide to Developing Business, Law, and Technology Solutions. New York : McGraw-Hill Education, 2018. ISBN 9781260115864.
4. MATZINGER, Claus. *Learn Rust Programming: Safe Code, Supports Low Level and Embedded Systems Programming with a Strong Ecosystem*. BPB Publications, 2022. ISBN 9789355511546.
5. PIERRO, Giuseppe A. a TONELLI, Roberto. Can Solana be the Solution to the Blockchain Scalability Problem? Online. 2022. Dostupné z: https://doi.org/10.1109/SANER53432.2022.00144.
6. PIERRO, Giuseppe A. a AMOORDON, Andy. A Tool to check the Ownership of Solana's Smart Contracts. Online. 2022. Dostupné z: https://doi.org/10.1109/SANER53432.2022.00140.
7. ANTONOPOULOS, Andreas M. a WOOD, Gavin. Mastering Ethereum: Building Smart Contracts and DApps. Sebastopol, CA : O'Reilly Media, 2018. ISBN 9781491971918.


Vedoucí diplomové práce: **Ing. David Malaník, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **5. listopadu 2023**
Termín odevzdání diplomové práce: **13. května 2024**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN
L.S.

**doc. Ing. Jiří Vojtěšek, Ph.D.** v.r.
děkan

**prof. Mgr. Roman Jašek, Ph.D., DBA** v.r.
ředitel ústavu

Ve Zlíně dne 5. ledna 2024

**Prohlašuji, že**

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

**Prohlašuji,**

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 5. 5. 2024

Bc. David Bilnica, v. r.
podpis studenta

## ABSTRAKT

Diplomová práce se zaměřuje na vývoj decentralizované crowdfundingové aplikace s využitím technologie Solana blockchain. V teoretické části práce jsou vysvětleny základní koncepty blockchainu a crowdfundingových platforem, včetně jejich funkcí a typů, s důrazem na modely založené na dárcovství. Práce podrobně vysvětluje důvody výběru Solana blockchainu, včetně popisu jeho architektury a inovativního konsensuálního mechanismu Proof of History, který zajišťuje rychlé a efektivní zpracování transakcí. Praktická část práce se zaměřuje na návrh a vývoj aplikace, zahrnující definování rolí uživatelů, zabezpečení, transparentnost systému a testování implementovaného programu. Závěrem práce je prezentace decentralizované crowdfundingové platformy, která se vyznačuje efektivitou a bezpečností. Platforma je navržena tak, aby byla přístupná širokému spektru uživatelů a umožnila financování inovativních a sociálně prospěšných projektů.



Klíčová slova: blockchain, Solana, crowdfunding, decentralizované aplikace, Rust, Proof of History



## ABSTRACT

This thesis focuses on developing a decentralized crowdfunding application utilizing Solana blockchain technology. The theoretical part explains the fundamental concepts of blockchain and crowdfunding platforms, including their functions and types, emphasizing donation-based models. The work elaborately discusses the reasons for selecting the Solana blockchain, including a description of its architecture and the innovative consensus mechanism, Proof of History, which ensures fast and efficient transaction processing. The practical part of the thesis focuses on the design and development of the application, encompassing the definition of user roles, system security, transparency, and testing of the implemented program. In conclusion, the work presents a decentralized crowdfunding platform characterized by efficiency and security. The platform is designed to be accessible to a wide range of users and enables innovative and socially beneficial projects to be funded.



Keywords: blockchain, Solana, crowdfunding, decentralized applications, Rust, Proof of History

# PODĚKOVÁNÍ

Rád bych poděkoval panu Ing. Davidu Malaníkovi, Ph.D., za jeho vynikající vedení a odbornou podporu během vypracování mé diplomové práce. Jeho rady a připomínky mi v posledním roce poskytly především směr, což bylo zásadní pro můj akademický růst a úspěšné dokončení této práce.

Dále bych chtěl vyjádřit hlubokou vděčnost své rodině a přátelům za jejich nekonečnou podporu, lásku a povzbuzení, které mi poskytovali po celou dobu mého studia. Vážím si jejich trpělivosti, porozumění a důvěru, která mi dodávala sílu překonávat překážky a pokračovat v mém akademickém snažení. Bez jejich morální a emoční podpory by byla moje cesta vzděláním mnohem náročnější.

Toto poděkování je také věnováno každému, kdo měl vliv na můj osobní a profesní rozvoj. Zvláštní poděkování náleží Nikole Hlinské a Martinu Hlinskému za jejich cenné rady a gramatickou korekci mé práce. Děkuji všem, kdo přispěli k mému úspěchu. Vaše podpora a inspirace jsou pro mě neocenitelné.

Obrázek 1. DALL·E – Visionary Portrait [36]

*„Your time is limited, so don't waste it living someone else's life."*

*Steven Paul Jobs*

Prohlašuji, že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Prohlašuji, že při tvorbě této práce jsem použil nástroj generativního modelu AI ChatGPT; chat.openai.com za účelem generování vizuálního obsahu pomocí DALL·E, asistence při formálních úpravách textu včetně oprav gramatiky a zlepšení stylistiky, a také pro refaktorování kódu. Po použití tohoto nástroje jsem provedl kontrolu obsahu a přebírám za něj plnou zodpovědnost.

# OBSAH

# ÚVOD

V současné době, kdy technologie formují nové přístupy k financování projektů a podnikání, se objevují stále inovativnější metody a nástroje. Crowdfunding, který prostřednictvím internetových platforem umožňuje shromažďování finančních prostředků od široké veřejnosti, je jedním z těchto přístupů. Tento trend vedl k požadavkům na větší decentralizaci a transparentnost, což následně podnítilo rozvoj decentralizovaných aplikací tohoto typu na blockchainových platformách. Tato diplomová práce se zaměřuje na návrh a vývoj decentralizované crowdfundingové aplikace s využitím technologie blockchainu Solana, který se vyznačuje rychlostí, efektivitou a nízkými transakčními náklady.

Hlavním cílem této práce je prozkoumat možnosti a výzvy spojené s vývojem a nasazením takové aplikace a identifikovat klíčové rozdíly oproti tradičním centralizovaným řešením. Práce dále zkoumá praktické využití blockchainové platformy Solana, známé pro svůj inovativní konsensuální mechanismus.

V teoretické části je zkoumána architektura a konsensuální mechanismus Proof of History blockchainu Solana, který zajišťuje vyšší efektivitu zpracování transakcí a škálovatelnost oproti tradičním platformám jako je Ethereum. Tato část práce objasňuje klíčové principy decentralizovaných aplikací a rovněž zdůrazňuje specifika Solany, která ji činí vhodnou pro aplikace vyžadující rychlé transakce s nízkými náklady. Praktická část se následně zaměřuje na implementaci konkrétní crowdfundingové aplikace na této platformě. Detailně jsou popsány fáze od návrhu až po realizaci aplikace, včetně identifikace uživatelů, zabezpečení kampaní a testování funkcionalit aplikace, čímž je demonstrována schopnost Solana blockchainu transformovat teoretické koncepty do efektivní praktické podoby.

Tato práce poskytuje důkladný přehled o využití blockchainové technologie ve sféře crowdfundingových platforem, s důrazem na zlepšení transparentnosti, bezpečnosti a dostupnosti. Detailní analýza a vývoj decentralizované aplikace na platformě Solana ukazuje, jak lze moderní technologické inovace efektivně uplatnit v praxi. Výsledky práce otevírají nové perspektivy pro financování projektů a naznačují směry dalšího vývoje v této dynamicky se rozvíjející oblasti.

# I. TEORETICKÁ ČÁST

# 1 DEFINICE POJMŮ Z OBLASTI MODERNÍCH TECHNOLOGIÍ

Ve světě moderních technologií se běžně setkáváme s názvy, výrazy či zkratkami které jsou převzaté z jiných jazyků (nejčastěji z angličtiny). Často nedokážeme najít vhodný překlad, a proto používáme jejich původní název nebo zkratku.

## 1.1 Crowdfunding

Crowdfunding je metoda a také způsob, jak získat peníze na financování projektů a podniků. Umožňuje jednotlivým organizátorům kampaní vybírat peněžní prostředky od jednotlivců prostřednictvím online platforem. Tato metoda tedy spojuje zakladatele jednotlivých kampaní a podporovatele, kteří jsou ochotni tyto kampaně finančně podpořit. Nejedná se tedy o běžné metody financování jako jsou půjčky, bankovní úvěry, či peníze od investorů, a nabízí tedy širší možnosti získání kapitálu [1]; [2].


Obrázek 2. Rozdíl mezi tradičním financováním a crowdfundingem [3]

Crowdfunding je často využíván začínajícími i rostoucími podniky jako jeden z možných způsobů financování, což představuje inovativní metodu získávání finančních prostředků pro nové projekty a nápady [1]; [2].

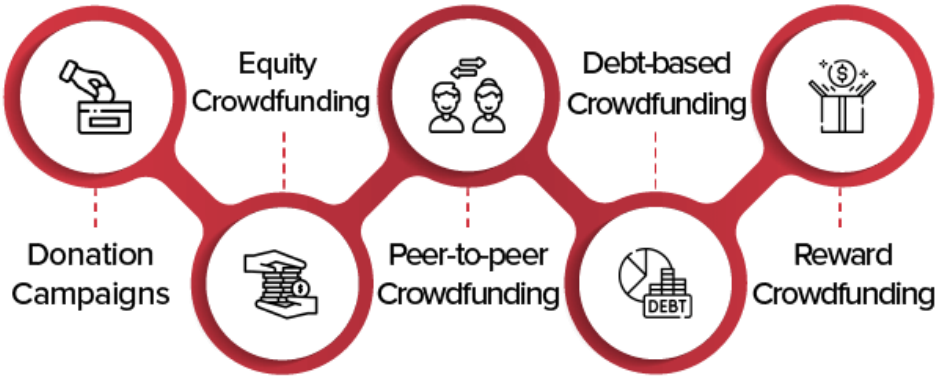
## 1.2 Funkcionalita crowdfundingu

Crowdfundingové platformy představují v dnešní době klíčový nástroj pro možnosti financování se zapojením široké veřejnosti, která je tato úsilí ochotna podporovat. Jednotlivé platformy lze popsat jako webové stránky, které umožňují interakci mezi organizátory a jednotlivými podporovateli [2]; [4].

Mnoho crowdfundingových platforem účtuje organizátorům kampaní poplatky pouze v případě, že jejich kampaň dosáhne požadovaného finančního cíle. Jako protihodnota se od poskytovatelů těchto platforem očekává nabídka služeb, které jsou nejen bezpečné, ale také i intuitivní a snadné k používání. Tyto vlastnosti mohou být rovněž zachovány i v případě, že se jedná o primárně dárcovské platformy bez jakýchkoliv poplatků [2]; [4].

## 1.3 Přehled hlavních typů crowdfundingu

V digitálním světě existuje několik hlavních typů crowdfundingu a každý z nich má unikátní zaměření a přístup k financování jednotlivých kampaní. Některé z platforem mohou být více vhodné například pro startupy[1] a neziskové organizace, kdežto jiné mohou být více přívětivé pro inovátory a kreativce. Je důležité podotknout, že všechny crowdfundingové platformy mají společný cíl, a tím je odstranění tradičních bariér, poskytování podpory a sdílení jednotlivých projektů. Crowdfundingové platformy jsou tedy děleny na různé modely, což vizuálně znázorňuje obrázek *3. Rozdělení hlavních typů crowdfundingu* [2]; [4].




Obrázek 3. Rozdělení hlavních typů crowdfundingu [5]

[1] **Startup** – jedná se o mladou společnost nebo projekt, který je založen s cílem rychle najít udržitelný a škálovatelný obchodní model, často v technologickém odvětví [6]

### 1.3.1 Akciový crowdfunding

Tento typ crowdfundingu poskytuje jednotlivým podporovatelům možnost koupit podíl v podniku (kampani), a to výměnou za jejich investici do dané kampaně. Tato forma financování je podobná nákupu akcií na burze cenných papírů nebo investování do rizikových kapitálů. Hlavním rozdílem je také to, že v případě klasických akcií je tato investice dostupná pouze pro určitou oblast investorů, a často se také jedná o soukromé akciové fondy, zatímco v případě crowdfundingu jsou akcie nabízeny široké veřejnosti [1]; [2].

### 1.3.2 Crowdfunding založený na odměnách

V případě tohoto typu crowdfundingu přispívají podporovatelé na jednotlivé projekty s očekáváním, že jako protihodnotu získají nefinanční odměnu, což může být produkt či služba. Tento model je velmi oblíben mezi startupovými projekty [1]; [2]; [4].

### 1.3.3 Crowdfunding založený na dárcovství

Crowdfunding založený na dárcovství je formou digitální podpory, kde přispěvatelé finančně podporují různé projekty z altruistických[2] důvodů, bez očekávání osobního zisku. Tento typ financování je často využíván pro podporu charitativních a sociálních iniciativ, umožňujících dosáhnout konkrétních cílů a napomáhajících lepšímu světu. Klíčovým prvkem je kolektivní úsilí, kde i malé příspěvky mohou ve velkém přispět k úspěchu vybrané kampaně a přinést změnu tam, kde je to nejvíce potřeba [2]; [4].

### 1.3.4 Crowdfunding založený na sdílení zisku / sdílení příjmů

Některé projekty nabízejí podporovatelům podíl na budoucích ziscích nebo příjmech, a to výměnou za kapitál získaný prostřednictvím crowdfundingu. Tento model je atraktivní především pro podporovatele hledající dlouhodobý výnos z provedené investice [2]; [4].

### 1.3.5 Crowdfunding založený na dluhových cenných papírech

Tento způsob crowdfundingu představuje investici do dluhových cenných papírů. Typickým příkladem je dluhopis, kde si emitent (vláda, obec, či korporace) půjčuje peníze od jednotlivých investorů a následně se zavazuje vrácením původní částky s úroky na konci

[2] **Altruismus** – popisuje jednání ve prospěch jiných bez očekávání osobního zisku [66]

předem dohodnutého období. Podporovatelé tedy poskytují svůj kapitál s očekáváním, že jejich investice bude splacena s úrokovým výnosem [2]; [4].

### 1.3.6 Crowdfunding založený na Peer-to-Peer půjčkách

Peer-to-Peer[3] (P2P) půjčky umožňují jednotlivcům půjčovat peníze přímo jiným osobám, nebo podnikům bez zprostředkování skrze tradiční finanční instituce. Tento model vychází ze stejného principu jako u běžných bankovních půjček, tedy že peníze budou splaceny s úroky [2]; [4].

### 1.3.7 Crowdfunding založený na hybridních modelech

Hybridní modely jsou v případě crowdfundingu často využívány v případech, kdy je zapotřebí kombinovat prvky více než jednoho typu financování, což ve výsledku poskytuje větší flexibilitu v přístupech získávání kapitálu. Tato možnost je vhodná především pro projekty, které nezapadají přesně do jedné z kategorií crowdfundingu [2]; [4].

## 1.4 Výběr crowdfundingového modelu pro vytvářenou aplikaci

Pro vývoj crowdfundingové aplikace byl zvolen dárcovský model crowdfundingu, který je blíže popsán v podkapitole *1.3.3 Crowdfunding založený na dárcovství*. Tento model byl vybrán na základě přesvědčení, že jeho přístup nejlépe podporuje projekty s významnými společenskými, či kulturními dopady. Jedná se o dárcovský model, kde přispěvatelé finančně podporují projekty bez jakéhokoliv očekávání hmotné odměny. Model tak umožňuje potenciálním podporovatelům přímou podporu iniciativ, které mají pozitivní dopad na komunitu, a zároveň zvyšuje transparentnost mezi jednotlivými organizátory kampaní a podporovateli. Volba dárcovského modelu tak představuje klíčový krok k dosažení cíle pro vytvoření platformy, která nejen usnadňuje financování hodnotově orientovaných projektů, ale také podporuje vytváření smysluplných konexí mezi lidmi, kteří chtějí cíleně přispět k danému projektu.

[3] **Peer-to-peer** – označuje systém, kde jednotlivci interagují přímo s jinými jednotlivci, obvykle přes internet, na základě rovnocenného postavení, bez účasti centrální autority nebo zprostředkovatele [7]

## 1.5 Specifikace požadovaných funkcí vyvíjené crowdfundingové aplikace

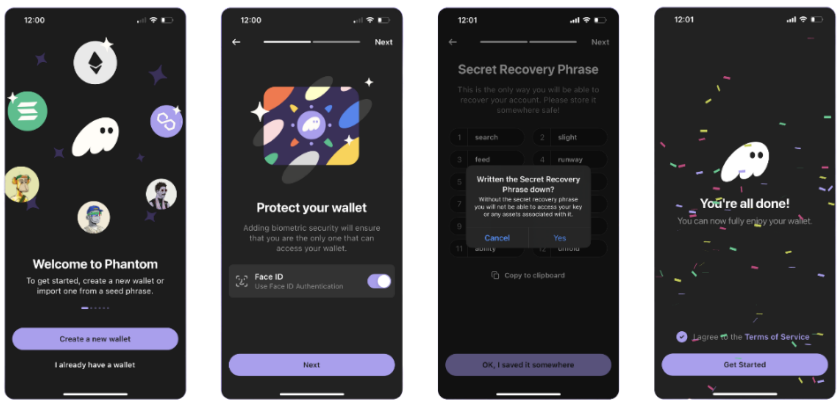
V éře digitálních inovací se crowdfunding ukázal jako zcela revoluční způsob, jak přeměnit různé kreativní nápady v realitu. Vývoj decentralizované crowdfundingové aplikace na blockchainu je reakcí na tuto potřebu, nabízející příslib zvýšené efektivity, transparentnosti a bezpečnosti. V kontextu vyvíjené aplikace jsou klíčové prvky a funkce zaměřeny na vytvoření uživatelsky přívětivého prostředí, které působí jako most mezi tvůrci kampaní a jejich potenciálními podporovateli. Na základě podkapitoly *1.4 Výběr crowdfundingového modelu pro vytvářenou aplikaci* bude model aplikace vytvořen typem crowdfundingu založeném na dárcovství včetně vybudování vhodného uživatelského prostředí.

Následující podkapitoly se zaměřují na hlavní požadované funkce, které jsou stěžejní pro úspěch vytvářené crowdfundingové aplikace.

### 1.5.1 Identifikace uživatelů

V decentralizované aplikaci bude zajištěna identifikace uživatelů prostřednictvím Solana Wallet Adapteru [8], což uživatelům umožní pohodlné a bezpečné přihlašování pomocí peněženek, jako jsou Phantom [9] nebo Soflare [10]. Tento přístup k autentizaci bude klíčem k ochraně digitální identity uživatele a zajistí tak, že celkový proces přihlášení do aplikace je jak bezpečný, tak uživatelsky přívětivý.

Autentizace prostřednictvím blockchainových peněženek nepožaduje od uživatelů, aby sdíleli jakékoliv své osobní údaje nebo hesla, což výrazně zvyšuje ochranu soukromí. Uživatelé tedy jednoduše prokážou vlastnictví své peněženky a dostupných peněžních prostředků pouze svým „*podpisem*“ transakce požadavku na přihlášení, což umožní aplikaci ověřit jejich identitu bez použití tradičních přihlašovacích metod.




Obrázek 4. Ukázka postupu vytvoření peněženky Phantom [11]

### 1.5.2 Role uživatelů

V rámci vyvíjené aplikace je kladen důraz na strukturovaný a transparentní systém rolí pro všechny uživatele. Aplikace disponuje třemi základními rolemi: **Administrátor**, **Organizátor kampaně** a **Podporovatel**. Každá z těchto rolí má specifické funkce a oprávnění, které společně tvoří kostru celého systému. Tento přístup poskytuje základ pro budování důvěry mezi všemi zúčastněnými stranami.

**Administrátor**: Je zásadním pilířem vytvářené platformy a jeho role zahrnuje schvalování, revizi a případné zamítání kampaní vytvořených Organizátory. Tato úloha je klíčová pro udržení vysoké kvality a relevanci projektů prezentovaných na platformě. Administrátor má také možnost delegovat své pravomoci jiným uživatelům, což umožňuje flexibilní a dynamickou správu platformy.

**Organizátor kampaně**: Organizátoři neboli zakladatelé kampaní, představují klíčové role vyvíjené platformy. Jejich úkolem je detailně specifikovat své projekty, včetně názvu, popisu, cílové částky a doby trvání. Platforma jim bude poskytovat potřebné nástroje a podporu pro úspěšné dosažení financování, zároveň stanovuje jasné podmínky pro výběr prostředků z úspěšných kampaní. Jedna z podmínek, aby organizátor mohl svou kampaň předčasně ukončit a vybrat prostředky, je že dosáhne alespoň 80 % vybraných prostředků z požadované částky. Toto opatření umožňuje organizátorům flexibilitu v řízení kampaně a zajišťuje, že mohou efektivně využít nasbírané finanční prostředky pro realizaci svých projektů. Po předčasném ukončení kampaně a výběru prostředků již nebude možné od podporovatelů přijímat další příspěvky, což organizátorům umožňuje plánovat a přecházet k dalším fázím svých projektů s jistotou získaného financování.

**Podporovatel**: Podporovatelé jsou klíčoví aktéři, kteří svými příspěvky podporují projekty na platformě. Mohou volně vybírat, které kampaně chtějí podpořit a finančně přispívat jakýmkoli obnosem. Systém navíc umožňuje podporovatelům přehodnotit své rozhodnutí a příspěvek zrušit, což dodává procesu větší kontrolu nad investicemi.

### 1.5.3 Vytváření a správa kampaní

Aplikace by měla umožňovat snadné a efektivní založení kampaní, to stejné platí i pro správu kampaní. Organizátoři při vytváření mohou nastavit základní parametry kampaně, jako jsou název, popis, cílová částka a doba trvání. Platforma jim umožňuje v reálném čase sledovat průběh a plnění cílů kampaně.

### 1.5.4 Správa finančních cílů

Organizátor sbírky získá přístup k vlastnímu navigačnímu prvku v navigační liště, který bude obsahovat prezentaci jeho sbírky s možností v reálném čase sledovat aktuální stav financování díky vývojovému ukazateli a dalším vizuálním indikátorům. Tyto nástroje nabídnou nejen přehled o dosažení finančních cílů, ale také umožní identifikovat, zda byla kampaň již úspěšně dokončena nebo jaký jiný stav právě zaujímá. Tato vizuální zpětná vazba je klíčová pro organizátory i podporovatele, jelikož poskytuje jasnou a okamžitou informaci o pokroku kampaně.

### 1.5.5 Bezpečnost a transparentnost aplikace

Klíčové aspekty bezpečnosti a transparentnosti vytvářené aplikace jsou zásadně podpořeny využitím technologie Solana blockchainu. Bezpečnost je zajištěna díky decentralizované povaze blockchainu, která eliminuje jediný bod selhání a ztěžuje neoprávněné manipulace [12]; [19].

Transparentnost je zajištěna pečlivým zaznamenáváním každé transakce a poskytováním přístupu k detailním informacím o kampaních, včetně adresy kampaně a aktuální adresy jejího vlastníka. Integrace se Solana Explorerem[4] [13] umožňuje všem uživatelům snadno ověřit validitu každé transakce vykonané v aplikaci.

Tyto prvky aplikace tak vytváří důvěru mezi všemi účastníky platformy. Umožňuje úplnou transparentnost a kontrolu nad průběhem kampaní, což každému uživateli poskytuje jasné a ověřitelné informace.

[4] **Solana Explorer** – nástroj pro prohlížení transakcí a aktivit na blockchainu Solana, umožňující uživatelům získat přehled o síti a ověřovat transakce [13]

## 1.6 Blockchain

Technologie blockchain, která je považována za jednu z nejvýznamnějších inovací této doby, nabízí mimořádný potenciál pro radikální transformaci způsobu, jakým realizujeme transakce, ukládáme data a chráníme naši digitální identitu [14]; [15].

Tato technologie lze pojmout jako speciální typ robustní databáze. Na rozdíl od tradičních databází, které jsou typicky řízeny jednou centrální entitou, blockchain funguje na decentralizovaném principu, kde jsou data distribuována napříč celou sítí [12]; [15].

V této distribuované knize neboli *ledgeru*, jsou jednotlivé transakce zaznamenávány do strukturovaných bloků, které jsou vzájemně kryptograficky propojeny a díky tomu je vytvářen nepřetržitý a kontinuálně se rozvíjející řetězec spojených bloků. Každý blok obsahuje unikátní kód, známý jako hash[5], a tento hash se následně používá jako digitální otisk daného bloku. Začátek takového řetězce neboli také první blok je označován za *Genesis blok*, neobsahuje odkaz na předchozí blok, protože před ním žádný blok neexistuje [14]; [15].


Obrázek 5. Vizuální schéma bloků v blockchainové síti [17]

Blockchainová struktura zajišťuje, že jakmile jsou data do bloků zapsána, tak se stávají neměnnými a zároveň se zvyšuje transparentnost pro uživatele dané sítě díky decentralizaci. Tímto způsobem blockchain přispívá k větší důvěře mezi jednotlivými stranami bez nutnosti zásahu třetí strany, jako mohou být banky nebo vládní instituce. Díky vlastnostem, které nabízí blockchain, je každý blok verifikovatelný jakoukoliv stranou v síti, což má

[5] **Hash** – jedná se o výpočetní proces, při kterém se vstupní data libovolné délky transformují na výstup pevné délky, často sloužící k ověření integrity dat nebo jako digitální otisk [16]

značný význam v oblastech jako jsou crowdfunding, finanční služby, dodavatelské řetězce, hlasovací systémy a mnoho dalších [12]; [15].

Blockchainová síť je pomocí decentralizace konstruována tak, aby odolala selhání jednotlivých částí (uzlů), což zabraňuje zkolabování celého systému, a to v případě technické chyby nebo útoku. Toho je docíleno tím, že data nejsou soustředěna pouze na jedno místo, ale jsou distribuována po celé síti, díky tomu je blockchain přirozeně bezpečnější a zároveň vhodný pro manipulaci s citlivými daty [14]; [15]; [18].

V kapitole *1.5 Specifikace požadovaných funkcí vyvíjené crowdfundingové aplikace* jsou specifikovány požadované funkce aplikace, které přímo čerpají z výhod, jež blockchainové technologie přináší.

Následující kapitola se již plně věnuje konkrétní blockchainové platformě včetně porovnání s jinou platformu.

# 2 SOLANA

Solana představuje přelom v blockchainových technologiích, zaměřený na vytvoření systému schopného efektivní tolerance vůči byzantským chybám[6]. Rovněž disponuje rychlými finálními časy, což je důležité a nezbytné pro moderní aplikace, které vyžadují real-time zpracování a okamžitý transakční výsledek [15]; [19].

Solana přináší značné inovace v decentralizovaném výpočetním výkonu a zaznamenávání důvěryhodného času bez nutnosti vzájemné důvěry mezi uzly, což je docílené skrze konsensus mechanismu *Proof of History*. Tento mechanismus je stěžejní pro jedinečnou architekturu Solany, poněvadž umožňuje síťovým uzlům efektivně a bezpečně seřadit transakce a události s přesně stanoveným časovým razítkem [15]; [19].

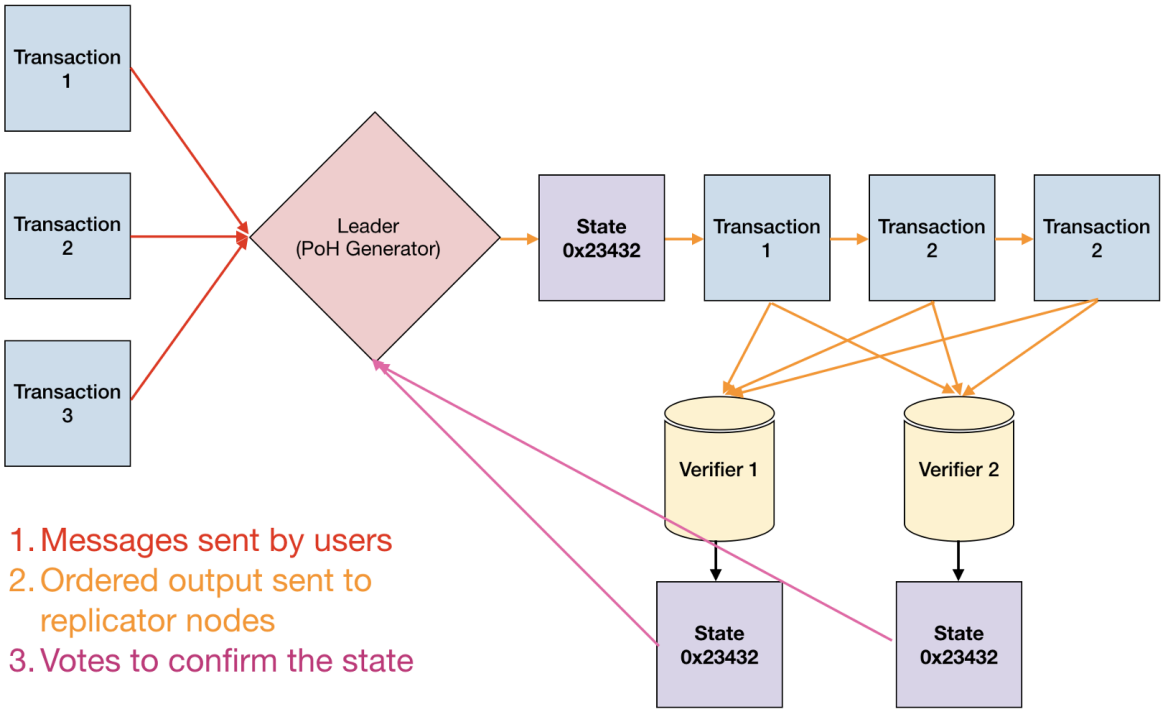
## 2.1 Solana Network Design

Návrh sítě obvykle představuje klíčový prvek při vytváření systému s unikátními mechanismy jako je Proof of History (PoH) spolu s Proof of Stake (PoS). Tento návrh umožňuje výběr specifického uzlu, často nazývaného „*Leader*“, který je odpovědný za generování sekvencí PoH, čímž zajišťuje verifikovatelný a konzistentní průchod v síti. Leader tedy uspořádává jednotlivé transakce uživatelů tak, aby byly ostatními uzly efektivně zpracovatelné, čímž maximalizuje propustnost sítě [19].

Po zpracování transakcí Leader publikuje výsledky a podpisy koncového stavu replikačním uzlům, a proto se těmto uzlům často říká „*Verifiers*“. Tyto uzly pak následně ověřují transakce a potvrzují své tvrzení podpisy, které se používají pro hlasování o změně stavu [19].

Tento proces je znázorněn na obrázku *6. Tok transakce skrze síť Solana*, kde je zobrazen tok transakcí v síti, což poskytuje názornější přehled na problematiku návrhu sítě a funkčnosti tohoto mechanismu.

[6] **Byzantská chyba** – jedná se o chybovou situaci v distribuovaném systému, kdy jednotlivé komponenty selhávají a nemůžou spolehlivě komunikovat, což má za následek nedosažení shody mezi nimi [67]




Obrázek 6. Tok transakce skrze síť Solana [19]

## 2.2 Proof of History

Základní podstata PoH představuje klíčovou inovaci v oblasti blockchainové technologie, sloužící k efektivnímu zaznamenávání a ověřování pořadí a časového sledu událostí, bez jakékoliv potřeby vzájemné důvěry nebo přímé komunikace mezi uzly sítě. Tato kryptografická metoda umožňuje jednotlivým systémovým komponentům nezávisle generovat a ověřovat důkazy o specifických událostech a časových okamžicích, což v konečném důsledku zvyšuje efektivitu a škálovatelnost blockchainů [19].

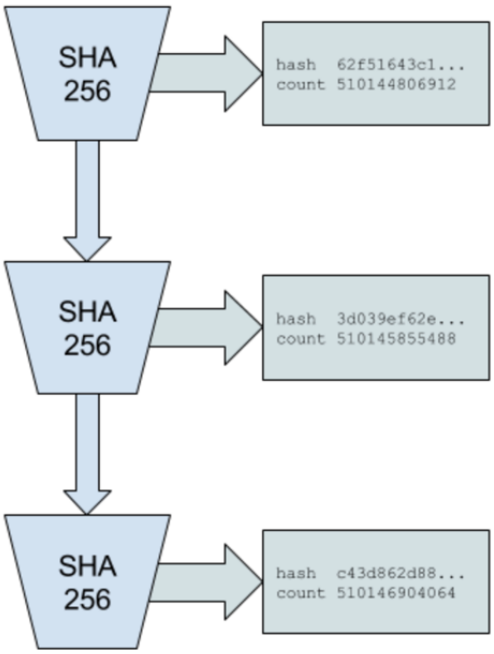
### 2.2.1 Sekvence Proof of History

Sekvence PoH, která je ilustrována na obrázku *7. Sekvence Proof of History*, představuje posloupnost hashů generovanou hashovacím algoritmem SHA-256. Každý hash v této posloupnosti je vytvořen aplikací algoritmu na kombinaci jeho předchozí hodnoty a specifického číselného vstupu, tzv. *count*. Tímto způsobem je zajištěno, že nelze odvodit konkrétní hash bez provedení výpočtů pro všechny předchozí kroky v sekvenci.

V tomto popisovaném obrázku je hash '*62f51643c1...*' spojen s počtem *510144806912* a hash '*c43d862d88...*' s počtem *510146904064*. Proces vytváření jednotlivých hashů tedy

poskytuje nezpochybnitelný důkaz, že mezi těmito dvěma počty, a tedy i mezi generováním příslušných hashů, musel uplynout reálný čas.

Využití PoH je efektivní alternativou k PoW, které vyžadují významné množství výpočetního výkonu a energie. PoH je proto v blockchainovém prostředí oceňován pro svou energetickou účinnost a schopnost přinést rychlejší a ekologičtější řešení.




Obrázek 7. Sekvence Proof of History [19]

### 2.2.2 Funkce a mechanismy Proof of History

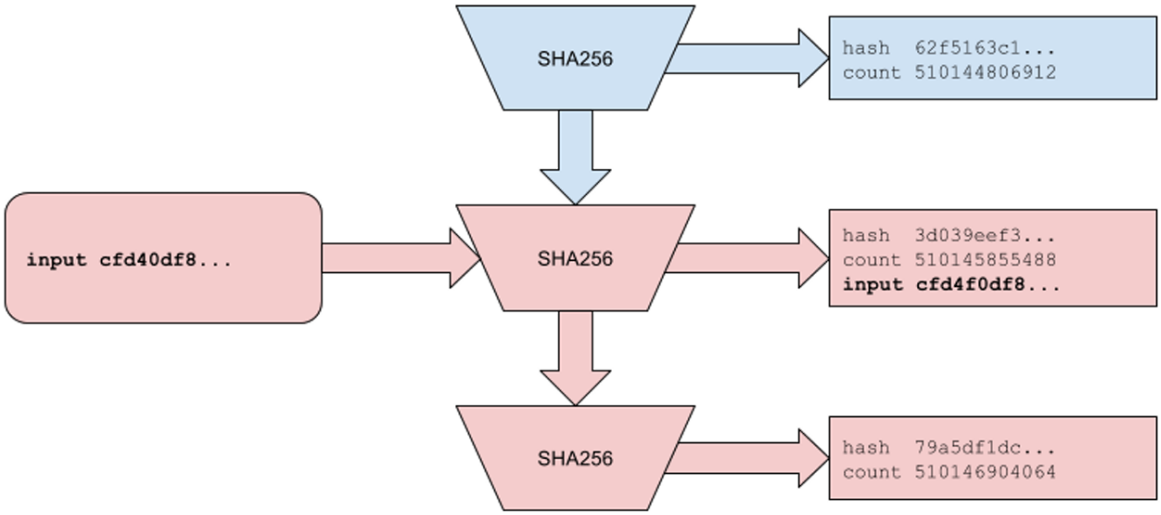
V rámci konsensu PoH, hrají sekvenční hashovací funkce zásadní roli při vytváření jedinečných otisků pro každou událost vytvořenou v síti. Tento proces je aplikován na vstupních datech, nejčastěji v podobě transakcí, na které se aplikuje hashovací funkce. Výstup z této funkce se stává vstupem pro další hashování, čímž vzniká řetězec závislých hashů, reprezentující sekvenční průběh událostí [19].

Tento unikátní mechanismus konsensu zajišťuje, že každý hash unikátně reprezentuje specifický moment v čase, protože je přímo závislý na předchozím hashi v řetězci. Pomocí této závislosti je možné jednoznačně ověřit pořadí a časovou následnost událostí bez jakékoliv potřeby externího časovače nebo koordinace mezi uzly [19].

Tato sekvenční závislost rovněž znamená, že jakékoliv pokusy o manipulaci s pořadím událostí nebo retroaktivní změny by byly snadno detekovatelné, díky nezměnitelnosti a kryptografickým vazbám hashů. Díky tomu je zajištěna integrita a celková transparentnost

celého procesu, což je základem pro důvěru a bezpečnost v blockchainových sítích využívajících konsensus algoritmu PoH [19].

Obrázek níže *8. Proces sekvenčního hashování a vkládání dat do PoH*, demonstruje proces PoH pomocí hashovací funkce SHA-256. Vstupní data (označená *cf4d0df8...)* procházejí opakovaným procesem vytváření hashů, vytvářející sekvenční řadu vzájemně propojených hashů. První hash (*3d039eef3...*) je určen jako vstup pro další operaci, výsledkem je následující hash (*79a5df1dc...*). Tento proces kontinuálně pokračuje, přičemž každý nový hash je odvozen od předchozího, a zahrnuje i hodnotu „*count*“, která indikuje počet provedených operací. Tím je zajištěna nezpochybnitelná časová posloupnost v PoH [19].

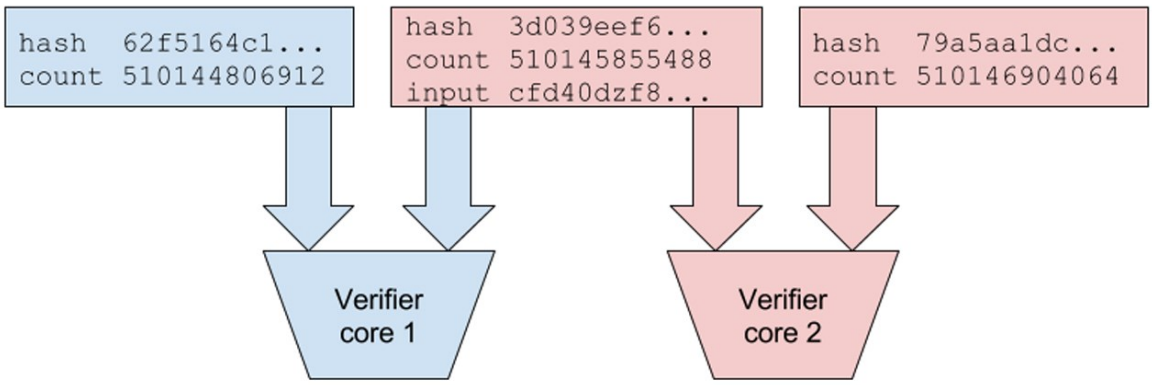



Obrázek 8. Proces sekvenčního hashování a vkládání dat do PoH [19]

### 2.2.3 Význam pro verifikaci a synchronizaci konsensu

Konsensus algoritmu PoH přináší rovněž revoluční změnu v oblasti verifikace a synchronizace v blockchainových sítích. Svou schopností zajišťuje, že každá transakce a událost může být nezávisle ověřena s přesným časovým razítkem, výrazně snižuje potřebu časové synchronizace mezi různými uzly v síti [19].

Obrázek *9. Verifikace s vícejádrovým ověřováním*, demonstruje proces nezávislé verifikace za využití mnohačetných ověřovacích jader. Každé z těchto jader je odpovědné za kontrolu určitých hashových hodnot a číselných sekvenčních údajů, což vede k lepší efektivitě a spolehlivosti celého systému. Jak je viditelné na ilustraci, vstupy na levé i pravé straně, obsahující hash a sekvenční čísla, jsou distribuovány mezi různá jádra pro samostatné ověření. To umožňuje systému udržet vysokou míru integrity a snižuje riziko chyb způsobených centralizovaným zpracováním dat [19].




Obrázek 9. Verifikace s vícejádrovým ověřováním [19]

Díky tomuto snížení závislosti mezi vzájemnou komunikací uzlů dochází nejen k rapidnímu snížení latence, ale také výraznému zvýšení celkové propustnosti a efektivity sítě. V důsledku těchto vlastností PoH umožňuje blockchainům dosáhnout mnohem vyšší míry škálovatelnosti a zpřístupňuje cestu k dosažení horizontálního škálování, což je kritické pro podporu rostoucího objemu transakcí a adaptaci v různých aplikačních doménách [19].

### 2.2.4 Implementace a využití Proof of History

Implementace PoH v blockchainové síti Solana představuje zcela významný pokrok ve zvyšování propustnosti a snižování latence. PoH tak umožňuje blockchainu Solana efektivně zaznamenávat a ověřovat pořadí transakcí pomocí kryptografických otisků, což jak bylo zmíněno, eliminuje potřebu pro synchronizaci času mezi uzly, a zároveň dochází ke zjednodušení celkového provozu sítě [19].

S pomocí PoH Solana dokáže zpracovat stovky tisíc transakcí každou sekundu, což je mnohonásobně vyšší v porovnání s tradičními známými blockchainy. Tato vlastnost tak otevírá dveře široké škále aplikací až pod decentralizované aplikace, které vyžadují vysokou rychlost a spolehlivost [19]; [20].

Solana díky PoH nabízí robustní a škálovatelnou platformu pro vývoj blockchainových aplikací, přičemž dochází k zachování klíčových principů decentralizace a bezpečnosti. Implementace PoH se nejen vypořádá s některými základními omezeními současných blockchainů, ale také rozšiřuje možnosti jejich následného praktické využití [19].

# 3 ANALÝZA ROZDÍLŮ MEZI SOLANA A ETHEREUM BLOCKCHAINEM

Rozdíly mezi Solana a Ethereum [79] blockchainem představují klíčové téma pro pochopení současného a budoucího vývoje decentralizovaných aplikací. Ethereum, jako zakladatel éry smart kontraktů, si vybudovalo robustní ekosystém podporující širokou škálu aplikací. Nicméně s rostoucím využitím se objevily výzvy spojené s poplatky, škálovatelností a bezpečností. Oproti tomu Solana přichází se zcela novým přístupem, nabízejícím významné zlepšení v rychlosti a efektivitě transakcí díky svému unikátnímu konsenzu mechanismu. Tato kapitola se zaměřuje na porovnání těchto platforem s cílem identifikovat, jak jejich vlastnosti ovlivňují vývoj a nasazení decentralizovaných crowdfundingových aplikací [15]; [19]; [20].

## 3.1 Porovnání složitosti a čitelnosti kódu

V úvodní části podkapitoly o složitosti a čitelnosti kódu je klíčové podtrhnout hlavní programovací jazyky obou blockchainů. Ethereum, využívající Solidity [80], a Solana, jež se spoléhá na robustní jazyk Rust [64]. Tyto jazyky přinášejí odlišné přístupy, které mají významný dopad na vývoj aplikací, zejména v rámci specifik vytvářené decentralizované crowdfundingové platformy. Solidity je především známé pro svou přístupnost a jednoduchost, což usnadňuje práci začínajícím vývojářům, zatímco Rust nabízí pokročilé možnosti řízení a optimalizace, které si žádají hlubší technické znalosti a jsou tak náročnější na vývoj a údržbu. Rozdíly v složitosti a čitelnosti mezi Solidity a Rust jsou významné a musí být zohledněny při rozhodování o platformě pro konkrétní aplikaci. Tabulka níže lépe popisuje jednotlivé odlišnosti těchto dvou blockchainů [22]; [23].

Tabulka 1. Porovnání složitosti a čitelnosti kódu [22]; [23]

| | **Ethereum** | **Solana** |
|---|---|---|
| **Programovací jazyk** | Solidity | Rust |
| **Syntaxe kódu** | Vysoká úroveň abstrakce, připomínající JavaScript | Nízká úroveň abstrakce, připomínající C++ |
| **Průměrná délka programu** | Kratší, kvůli abstrakcím a šablonám | Delší, detailnější kód kvůli manuální správě paměti |

| | | |
|---|---|---|
| **Škálovatelnost kódu** | Omezena rychlostí sítě Ethereum | Vysoká, využívá paralelní zpracování transakcí |
| **Údržba kódu** | Snadná díky jasnější struktuře a modularitě | Komplexnější kvůli detailnější správě systémových zdrojů |
| **Bezpečnostní prvky** | Rozsáhlé bezpečnostní šablony a vysoký důraz na bezpečnost | Vyžaduje pokročilé znalosti pro zajištění bezpečnosti |
| **Přístup k dokumentaci** | Dobře dokumentované, s bohatými zdroji a komunitní podporou | Dobře dokumentované, ale s větším důrazem na technické detaily |
| **Komunitní podpora** | Velká a aktivní komunita | Rychle rostoucí komunita s technickým zaměřením |

### 3.1.1 Ukázka kódu podpora kampaně v programovacím jazyku Rust

Kód napsaný v jazyku Rust pro blockchain Solana ukazuje, jak přistupovat a manipulovat s účty pomocí typově bezpečného kontextu, což vyžaduje přesné definice a explicitní zpracování stavů a transakcí. Příklad demonstruje důkladné bezpečnostní ověření pomocí *require!* makra, které zajistí, že transakce splňuje všechny potřebné podmínky, tedy aktivitu kampaně, časové omezení a neprovedení výběru. Převod finančních prostředků je realizován přímým voláním systémové instrukce, což odhaluje nižší úroveň abstrakce a zvyšuje kontrolu vývojáře nad transakcemi. Rust na Solaně tedy poskytuje vývojářům mocný nástroj pro vytváření efektivních a bezpečných decentralizovaných aplikací, avšak za cenu vyšší složitosti a potřeby pokročilejších technických znalostí [22]; [23].

Funkce *support_campaign()*, v kódu *1. Funkce pro podpoření kampaně v jazyku Rust,* slouží uživatelům k financování crowdfundingových kampaní. Funkce nejprve ověří, že kampaň je aktivní, neukončená a také zda-li prostředky již nebyly vybrány jejím zakladatelem. Finanční příspěvek následně pomáhá dosáhnout cílové částky kampaně, podporuje transparentnost a efektivní distribuci prostředků v digitálním ekosystému.

Kód 1. Funkce pro podpoření kampaně v jazyku Rust

```
pub fn support_campaign(ctx: Context<Support>, amount: u64) -> Result<()>
{
    let campaign = &mut ctx.accounts.campaign;
    require!(campaign.is_active, CrowdFundingError::CampaignNotActive);
    require!(campaign.end_campaign >= Clock::get()?.unix_timestamp,
        CrowdFundingError::CampaignEnded);
    require!(campaign.is_withdrawn == false,
        CrowdFundingError::WithdrawnCampaign);
    if amount + campaign.pledged >= campaign.goal {
        campaign.is_pledged = true;}
    let txn = an-chor_lang::solana_program::system_instruction::transfer(
        &ctx.accounts.user.key(),&ctx.accounts.campaign.key(),amount);
    anchor_lang::solana_program::program::invoke(
        &txn,&[ctx.accounts.user.to_account_info(),
            ctx.accounts.campaign.to_account_info(),],)?;
    (&mut ctx.accounts.campaign).pledged += amount;
    Ok(())
}
```

### 3.1.2 Ukázka kódu podpora kampaně v programovacím jazyku Solidity

Solidity kód pro Ethereum v sobě nese vysokou úroveň abstrakce, zjednodušující vývoj smart kontraktů pomocí strukturované syntaxe. Tato část programu ukazuje, jakým způsobem lze implementovat bezpečnostní kontroly prostřednictvím *require* funkcí, které validují stavy kampaně. Manipulace s finančními prostředky a emitování událostí jsou příklady funkcí, které Solidity zjednodušuje, umožňující tak snadnější a rychlejší vývoj. Ethereum platforma a jazyk Solidity tak nabízejí silné nástroje pro vytváření bezpečných a uživatelsky přívětivých aplikací s menším důrazem na nízkoúrovňovou manipulaci s blockchainem, což je ideální pro projekty, které potřebují rychlý vývoj a nasazení [21].

Funkce *supportCampaign()*, v kódu *2. Funkce pro podpoření kampaně v jazyku Solidity* umožňuje uživatelům přispívat na konkrétní *campaignId* s určenou částkou *amount*. Nejprve ověří, zda-li je kampaň aktivní, neukončená a přijímaná částka odpovídá poslané hodnotě. Finanční příspěvek se pak přičte k celkové částce kampaně a v případě dosažení cíle se kampaň označí, jako úspěšně financovaná. Nakonec se částka převede majiteli

kampaně a vyvolá se událost o podpoření kampaně, což zajišťuje transparentnost finančních prostředků.

Kód 2. Funkce pro podpoření kampaně v jazyku Solidity

```
function supportCampaign(uint campaignId, uint amount) public payable {
    Campaign storage campaign = campaigns[campaignId];
    require(campaign.isActive, "Campaign not active");
    require(block.timestamp <= campaign.endTime, "Campaign ended");
    require(!campaign.isWithdrawn, "Campaign withdrawn");
    require(msg.value == amount, "Amount does not match value sent");

    Pledger storage pledger = pledgers[campaignId][msg.sender];
    pledger.amount += amount;
    campaign.pledged += amount;
    if (campaign.pledged >= campaign.goal) {
        campaign.isPledged = true;}
    payable(campaign.owner).transfer(amount);

    emit CampaignSupported(campaignId, msg.sender, amount);
}
```

## 3.2 Porovnání konsensních mechanismů a bezpečnosti

Porovnání konsensních mechanismů a bezpečnostních strategií mezi Ethereum a Solana odkrývá zásadní rozdíly ve způsobu, jakým tyto dvě blockchainové platformy přistupují k zajištění soudržnosti a ochrany svých sítí. Ethereum, které se v nedávné době odklonilo k Proof of Stake[7], si klade za cíl zvýšit energetickou účinnost a posílit bezpečnostní protokoly. Naproti tomu Solana nabízí inovativní spojení PoH s PoS, což má za výsledek větší rychlost transakcí a efektivitu bez oslabení bezpečnostních aspektů. Tato diverzita v metodách pro dosahování konsenzu a zabezpečení sítě má výrazný dopad na jejich technické a operační charakteristiky, což je zásadní při vývoji a nasazení decentralizovaných crowdfundingových aplikací [19]; [22].

[7] **Proof of Stake (PoS)** – jedná se o konsensuální algoritmus, který vyžaduje, aby uživatelé drželi a zamykali určité množství kryptoměny, aby mohli validovat transakce a tvořit nové bloky [15]; [23]

### 3.2.1 Konsensní mechanismus Ethereum

Ethereum je považováno za průkopníka smart kontraktů, dlouho se tento blockchain spoléhal na Proof of Work[8] k dosažení konsensu. Tento mechanismus, i přes svou prokázanou bezpečnost a decentralizaci, byl velmi často kritizován za jeho vysokou energetickou náročnost. S nedávným přechodem na PoS v rámci aktualizace na Ethereum 2.0, platforma očekává významný pokles v energetické spotřebě a zvýšení transakční kapacity. PoS rovněž přináší i další bezpečnostní vylepšení, jako je lepší ochrana proti útokům typu 51%[9], kde místo výpočetní síly rozhoduje validační metoda na základně množství stakingu[10] Etherea. Tento krok by měl Ethereum učinit udržitelnějším a přístupnějším pro širší spektrum aplikací včetně těch, které vyžadují vyšší transakční rychlost, což může být požadavkem pro crowdfundingovou platformu [19]; [22].

### 3.2.2 Konsensní mechanismus Solana

Na druhé straně stojí Solana, která přichází s jedinečnou kombinací PoH (PoH blíže popsáno v kapitole *2.2 Proof of History*) spolu s PoS, aby poskytla rychlý a bezpečný konsensní mechanismus. Když je PoH kombinován s PoS, výsledkem je systém, který může zpracovávat tisíce transakcí za sekundu při zachování vysoké úrovně bezpečnosti. Tato vysoká propustnost a nízké poplatky dělají ze Solany ideální platformu pro aplikace vyžadující rychlé transakce, jako jsou crowdfundingové platformy, které mohou těžit z její schopnosti rychle zpracovávat malé příspěvky od velkého počtu uživatelů. Nízké poplatky jsou obzvláště přitažlivé pro podporovatele přispívající menšími částkami, což z transakčních poplatků činí významný faktor při rozhodování o podpoře projektu [19]; [27].

---

[8] **Proof of Work (PoW)** – jedná se o konsensuální algoritmus v blockchainu, který vyžaduje od účastníků řešení složitých matematických úloh pro ověření transakcí a vytváření nových bloků [15]; [21]

[9] **Útok typu 51%** - tento útok nastane když útočník získá nadpoloviční kontrolu nad výpočetní silou blockchainové sítě, což mu umožňuje manipulovat s transakcemi [24]

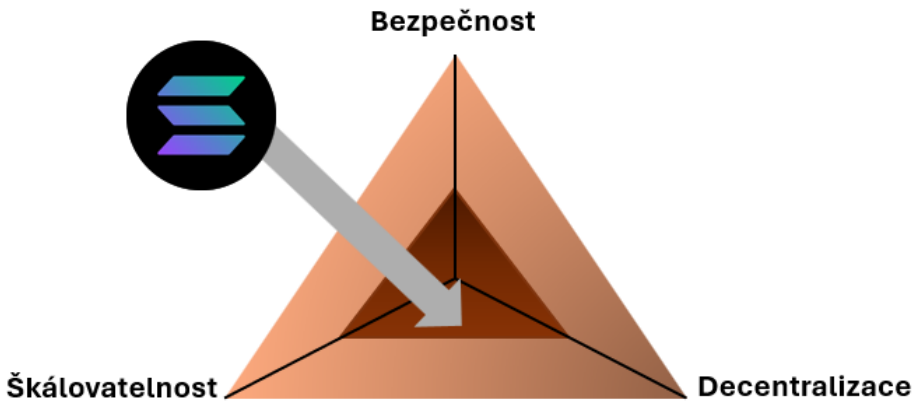
[10] **Staking** – je to proces, při kterém uživatelé zablokují určité množství kryptoměny jako vklad v síti, aby podpořili operace blockchainu a výměnou za to získávají odměny [25]

## 3.3 Porovnání škálovatelnosti a decentralizace

Klíčové aspekty jako jsou škálovatelnost a decentralizace jsou považovány za zásadní pro fungování a výkon blockchainových platforem. Škálovatelnost a decentralizace často představují dva koncové body spektra, kde zvýšení jednoho může vést k potenciálnímu snížení druhého. Ethereum a Solana k těmto aspektům přistupují odlišně, což má také významný dopad na výběr platformy pro vývoj decentralizovaných aplikací [15].

### 3.3.1 Škálovatelnost a decentralizace na Solana blockchainu

Solana představuje značný průlomový přístup v oblasti škálovatelnosti blockchainových technologií a platforem. Díky svému unikátnímu konsensnímu algoritmu, který spojuje PoH s PoS tak Solana nabízí nevídanou rychlost a kapacitu zpracování transakcí, které dosahují až 65 000 TPS [26]. Tato vysoká škálovatelnost umožňuje Solaně podporovat širokou škálu aplikací, počínaje od finančních služeb až po decentralizované aplikace vyžadující rychlou odezvu a nízké transakční poplatky. Značně tak otevírá dveře pro vývoj aplikací, které jsou typu crowdfundingové platformy, kde rychlé a efektivní transakce zvyšují uživatelskou přívětivost a snižují bariéry pro nasazení a použití [20]; [22]; [27].



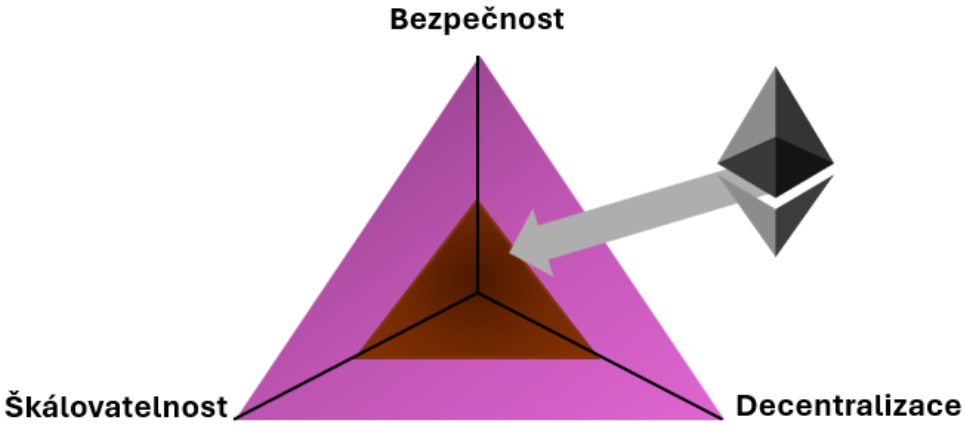

Obrázek 10. Trilemma Solana blockchainu [37]

Jak je znázorněno na obrázku *10. Trilemma Solana blockchainu*, Solana úspěšně skloubí vysokou škálovatelnost s postupným zlepšováním své decentralizace. Díky svému inovativnímu přístupu, který zahrnuje PoH a PoS, se Solana stává atraktivní volbou pro širokou škálu decentralizovaných aplikací. Zároveň platforma pracuje na rozšiřování svého validačního ekosystému, aby zlepšila decentralizaci a s tím související zabezpečení sítě. Tyto kroky primárně směřují k udržení škálovatelnosti a decentralizací, což Solanu posouvá

do čela inovací v oblasti blockchainu a představuje tak základ pro její další rozšíření a rozvoj [20]; [27].

### 3.3.2 Škálovatelnost a decentralizace na Ethereum blockchainu

Ethereum, jakožto v roli dlouholetého lídra v oblasti decentralizovaných aplikací a smart kontraktů postavilo svůj ekosystém na základech decentralizace a bezpečnosti. Přechod z PoW na PoS v rámci proběhlého upgradu na Ethereum 2.0 bylo klíčovým krokem ke zlepšení škálovatelnosti, přičemž si zachovává vysokou úroveň decentralizace. Tento přechod zajistil výrazně lepší energetickou efektivitu a transakční kapacitu, zatímco stále udržuje síť odolnou vůči útokům a manipulaci [21]; [27].




Obrázek 11. Trilemma Ethereum blockchainu [38]

Na obrázku *11. Trilemma Ethereum blockchainu* je znázorněno, že i přes významné pokroky v oblasti bezpečnosti a decentralizace se Ethereum stále potýká s výzvami, jako jsou transakční poplatky a latence, které mohou omezovat jeho použitelnost pro aplikace vyžadující vysokou propustnost, jako jsou některé formy crowdfundingových platforem. Uživatelé, a především vývojáři aplikaci k tomu problému přistoupili způsobem, že řešení jsou druhé vrstvy, jako jsou rollups[11] a sharding[12], které mají primárně zlepšit škálovatelnost, dostupnost dat a validitu dat, aniž by byla ohrožena decentralizace a bezpečnost [28]; [29].

[11] **Ethereum rollups** – technologie zlepšují škálovatelnost Ethereum tím, že provádějí výpočty mimo hlavní blockchain a následně zaznamenávají souhrnné výsledky zpět, čímž snižují poplatky a zvyšují rychlost transakcí [30]

[12] **Ethereum sharding** – jedná se o metodu pro zvýšení kapacity sítě rozdělením databáze na menší segmenty (shardy), které umožňují paralelní zpracování transakcí a údajů, což vede ke zvýšené škálovatelnosti a rychlosti transakcí [31]

## 3.4 Ekosystém a komunita platforem

Ve zcela dynamickém světě blockchainových technologií jsou ekosystém a komunita významnými faktory, které značně ovlivňují růst, inovace a adaptace jednotlivých platforem.

Jak Ethereum, tak i Solana, si svým jedinečným přístupem a filozofií, vybudovaly rozmanité ekosystémy. Ty jsou podpořeny aktivními uživateli a vývojáři, kteří formují směr, jímž se tyto platformy ubírají [27].

### 3.4.1 Ekosystém Solana

Solana se velmi rychle etablovala jako jedna z hlavních platforem pro vývoj decentralizovaných aplikací, zejména díky své výjimečné škálovatelnosti a rychlosti transakcí. Její ekosystém zahrnuje širokou škálu projektů, od DeFi[13] a NFT[14] až přes decentralizované sociální sítě. Inovativní technologie a vysoký výkon aplikací přilákaly mnoho vývojářů a startupů, kteří vyhledávali platformu schopnou podporovat aplikace s vysokými požadavky na transakční rychlost a kapacitu. Komunita této platformy je známá především svou technologickou sofistikovaností a otevřeností k novým nápadům, což významně podporuje rychlý vývoj, implementaci a nasazení nových inovativních řešení [19]; [27].

### 3.4.2 Ekosystém Etherea

Platforma Ethereum si od svých počátků vybudovala pozici hlavní platformy pro smart kontrakty a decentralizované aplikace. Jeho systém je i kvůli jeho vrstvám rozmanitější a nejbohatší na trhu, a ve svém jádru zahrnuje tisíce projektů zaměřených na DeFi, NFT, DAO[15] a mnoho dalších. Ethereum disponuje vysoce aktivní a rozsáhlou komunitou vývojářů, od uživatelů a fanoušků, kteří nestále přispívají k jeho rozvoji a inovačním procesům. Díky silné komunitě a rozsáhlé podpoře vývojářských nástrojů, včetně bohaté dokumentace, se tak Ethereum stává zajímavou možností pro nové projekty a začínající vývojáře, kteří se chtějí ponořit do světa blockchainových technologií [21]; [27].

---

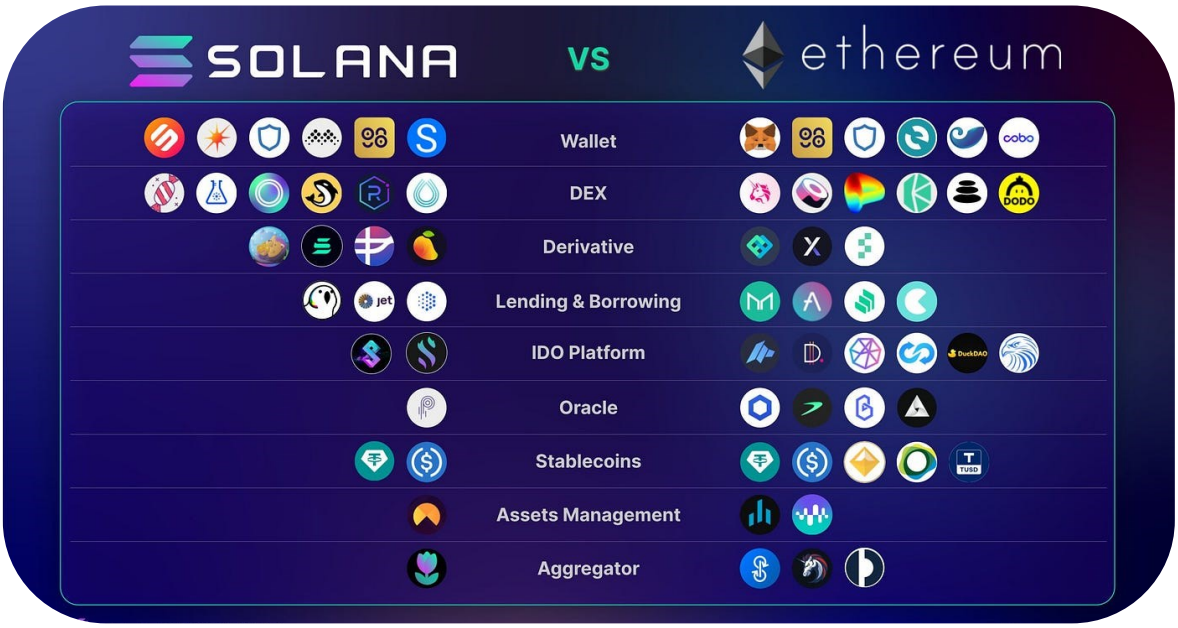
[13] **DeFi** – jedná se o ekosystém finančních aplikací na blockchainu, který umožňuje uživatelům provádět transakce a finanční služby bez centrálních autorit [32]

[14] **NFT** – unikátní digitální token na blockchainu, který představuje vlastnictví určitého aktiva, jako je umění nebo sběratelský předmět, a není vzájemně zaměnitelný [33]

[15] **DAO** – decentralizovaná organizace na blockchainu, kde členové hlasují o rozhodnutích pomocí tokenů, což umožňuje transparentní a demokratické řízení bez centrální autority [34]

### 3.4.3 Zhodnocení ekosystému Solany a Etherea

Obě platformy se vyprofilovaly jako klíčové v oblasti blockchainových technologií, každá s vlastními silnými stránkami odpovídajícími různým potřebám trhu. Zatímco Ethereum je uznávané převážně pro svou robustní decentralizaci a široký ekosystém aplikací, Solana se naopak prosazuje svou vysokou škálovatelností a rychlostí, což přitahuje projekty hledající efektivitu pro vlastní aplikace. Obě platformy nabízejí bohaté prostředí pro vývojáře a na obrázku *12. Populární aplikace na platformě Solana a Ethereum*, je možné vidět typy aplikací, které v jejich ekosystému prosperují. Při výběru mezi nimi je třeba zvážit specifické cíle a požadavky daného projektu v případě této práce se jedná především o crowdfundingovou aplikaci, přičemž obě platformy nabízí perspektivu pokračování v inovačním vedení a poskytování hodnoty v sektoru blockchainových technologií [22]; [27].



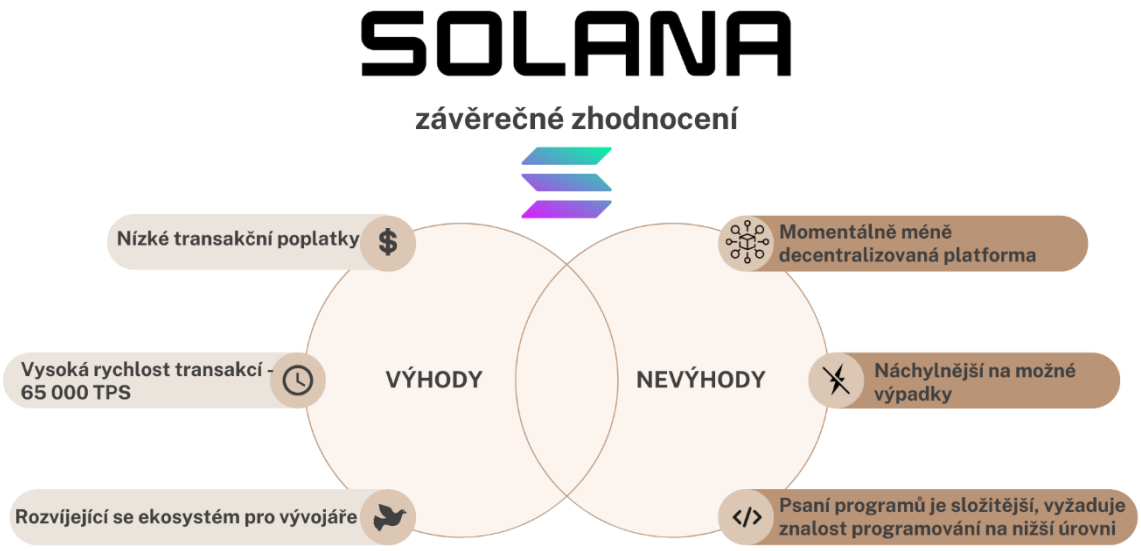

Obrázek 12. Populární aplikace na platformě Solana a Ethereum [35]

## 3.5 Závěrečné zhodnocení platforem

V kontextu vývoje decentralizovaných crowdfundingových aplikací se Solana rychle profiluje jako více preferovaná platforma, která nabízí unikátní kombinaci vysoké propustnosti, nízkých transakčních nákladů a pokročilých bezpečnostních funkcí. Tyto klíčové vlastnosti, společně s efektivním vývojovým prostředím založeném na Rustu, tak Solanu výrazně odlišují jako ideální volbu pro projekty hledající inovace a uživatelskou přívětivost ve světě crowdfundingových aplikací. V porovnání s Ethereem, které sice přináší široký ekosystém a silnou komunitu, se Solana vymezuje svou výjimečnou výkonností v klíčových aspektech, které jsou zásadní pro dynamické a účinné shromažďování kapitálu, čímž se stává přednější volbou pro projekty v této oblasti [22]; [27].

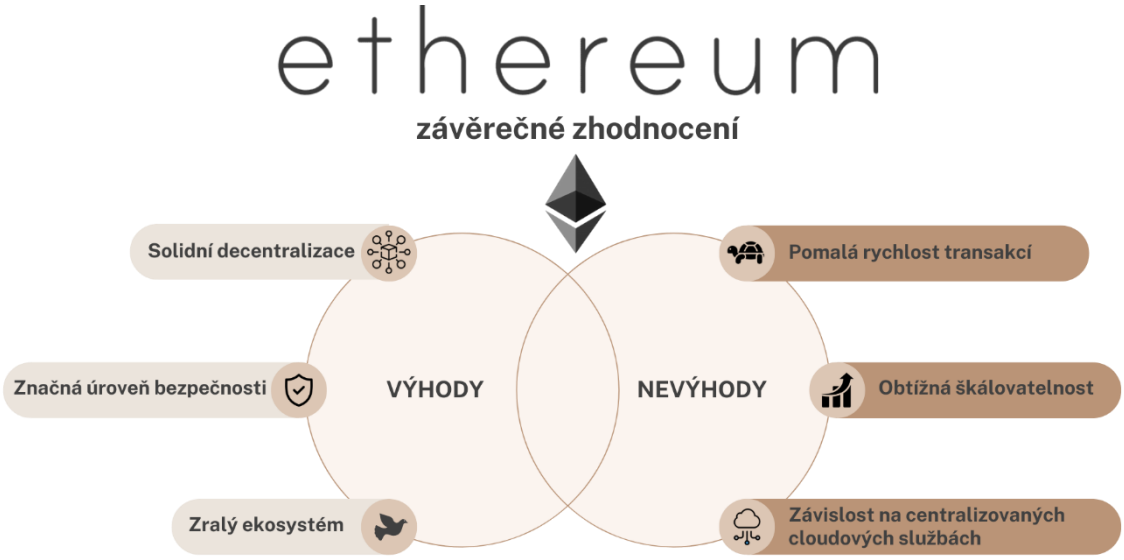
Solana se stává platformou volby pro vývojáře decentralizovaných crowdfundingových aplikací, jak dokládá obrázek *13. Závěrečné zhodnocení platformy Solana*. Tato platforma, s transakční rychlostí až 65 000 TPS [26] a konkurenceschopnými poplatky, nabízí efektivní prostředí pro fundraisingové kampaně. Díky rozvoji svého ekosystému a uživatelského rozhraní Solana podporuje inovace a ulehčuje uživatelům shromažďování kapitálu.

I přes výzvy jako jsou sporadické výpadky a požadavky na pokročilé programovací dovednosti z důvodu nízkoúrovňového programování, tak se Solana neustále zlepšuje a směřuje k poskytování rychlých a bezpečných řešení pro vytvoření, spuštění a řízení projektů [19]; [22]; [27].




Obrázek 13. Závěrečné zhodnocení platformy Solana [37]

Ethereum, považované za základní kámen v oblasti decentralizovaných aplikací, je respektované pro svou decentralizaci a značnou úroveň bezpečnosti, což činí z jeho ekosystému robustní prostředí pro vývoj decentralizovaných aplikací, jak je vidět na obrázku *14. Závěrečné zhodnocení platformy Ethereum*. Přestože Ethereum vyniká ve stabilitě a bezpečnosti, narazilo na bariéry v podobě pomalé rychlosti transakcí a obtížné škálovatelnosti, které se stávají překážkami pro některé uživatele a vývojáře. Tyto výzvy, společně se závislostí na centralizovaných cloudových službách, ukazují, že i přes jeho silné stránky je potřeba neustálého vývoje a inovací, aby bylo možné udržet krok s neustále se zrychlujícím tempem technologického pokroku v oblasti blockchainového prostoru [21]; [27].




Obrázek 14. Závěrečné zhodnocení platformy Ethereum [38]

# II. PRAKTICKÁ ČÁST

# 4 DECENTRALIZOVANÁ CROWDFUNDINGOVÁ APLIKACE

Tato kapitola diplomové práce je věnována vývoji decentralizované crowdfundingové aplikace. Následující podkapitoly obsahují postup, od návrhu přes výběr technologií po samotnou realizaci. Důraz bude kladen především na klíčové aspekty vývoje, včetně architektury, technologického stacku, programovacích paradigmat a bezpečnostních opatření. Z pohledu těchto prvků bude nabídnut komplexní přehled procesu tvorby aplikace na blockchainové platformě Solana.

## 4.1 Představení řešeného problému

Tato podkapitola je věnována problematice vývoje decentralizované aplikace určené pro crowdfunding, což jak již bylo zmiňováno v kapitole *1.1 Crowdfunding*, jedná se o moderní sbírání prostředků od veřejnosti pro podporu různých kampaní a iniciativ. Díky své schopnosti sjednocovat komunitu a získávat kapitál pro nové nápady a podniky, je crowdfunding stále populárnějším způsobem financování. V digitálním světě 21. století však čelíme několika výzvám a problémům, které mohou podkopat důvěru ve veřejné financování, a tak ztěžovat transparentní distribuci získaných prostředků.

Mezi klíčové obavy patří zabezpečení transakcí a jejich ověřitelnost, včetně autentičnosti identit podporovatelů a příjemců, jakož i úplná transparentnost a sledovatelnost transakcí, které efektivně brání jakékoli možné manipulaci s daty. V tradičních modelech crowdfundingových platforem může být zajištění úplné průhlednosti při současném zachování ochrany osobních údajů uživatelů značně náročné.

Řešením těchto je problémů je využití technologie blockchain k vytvoření crowdfundingové aplikace. Jak je popsáno v kapitole *1.6 Blockchain*, technologie blockchain přináší do procesu crowdfundingového financování novou úroveň bezpečnosti, transparentnosti a důvěry. Tyto přínosy jsou umožněny jedinečnými vlastnostmi blockchainu.

Decentralizovaná crowdfundingová aplikace na Solana blockchainu zajistí:

- **Transparentnost a ověřitelnost transakcí**: Každý finanční příspěvek je zaznamenán v blockchainové síti, což umožňuje jeho nezpochybnitelné ověření a také sledování toku finančních prostředků.
- **Decentralizaci**: Bez centrální autority zůstává průběh financování projektu chráněn před možným ovlivněním nebo manipulací jakoukoli stranou.

- **Snížení nákladů**: Blockchain umožňuje snížení transakčních nákladů a odstraňuje tak potřebu třetích stran a tím umožňuje širší spektrum crowdfundingových projektů.
- **Imunitu proti cenzuře**: Distribuovaný charakter blockchainu brání jednotlivým subjektům či autoritám v omezení přístupu k projektům nebo manipulaci s výsledky o financování kampaně.
- **Efektivitu a rychlost transakcí**: Solana nabízí vysokou rychlost a nízké transakční náklady, což zjednodušuje a zrychluje proces podpory jednotlivých kampaní.
- **Globální dostupnost**: Projekty mohou přijímat příspěvky z celého světa, což eliminuje bariéry spojené s tradičními finančními systémy.

Další podkapitoly detailně rozebírají vývojový proces decentralizované crowdfundingové aplikace, od úvodních kroků přes implementaci po testování.

## 4.2 Navrhované použití

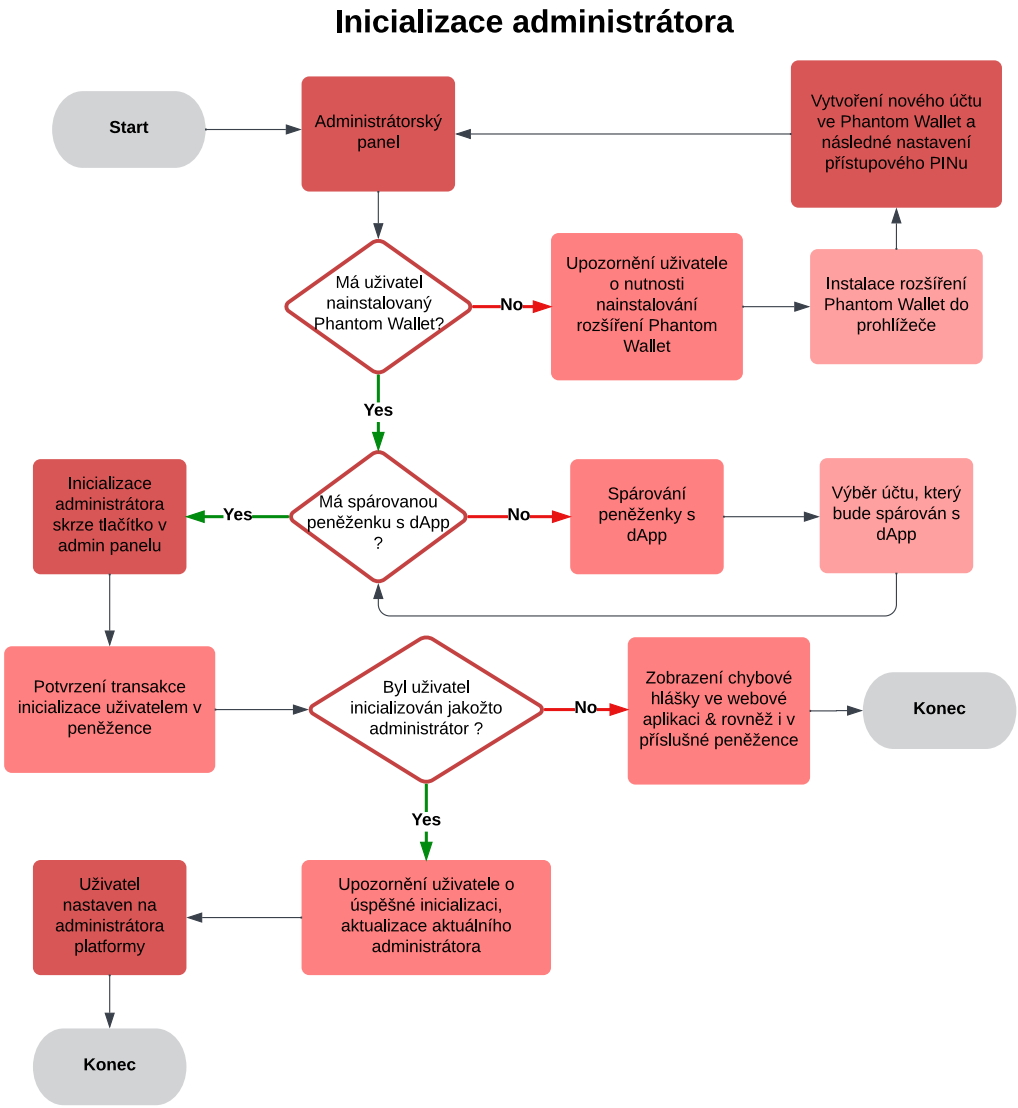
Ve vyvíjené crowdfundingové aplikaci se vyskytují tři klíčové role: **administrátor, organizátor kampaně a podporovatel**. Tyto role jsou podrobněji popsány v kapitole *1.5.2 Role uživatelů*, kde každá z nich je přiřazena specifickým funkcím a odpovědnostem v ekosystému platformy.

- **Organizátor kampaně** je odpovědný za založení a správu své kampaně. Tato role zahrnuje nejen iniciaci kampaně, ale i správu nasbíraných finančních prostředků po dosažení minimálně 80% z požadovaného finančního cíle. Po úspěšném vybrání financí je kampaň považována za dokončenou a další sběr prostředků není možný.
- **Administrátor** plní roli klíčového pilíře platformy, jehož úkolem je schvalovat nebo zamítat kampaně podané organizátory. Zajišťuje, že na platformě jsou prezentovány pouze projekty, které splňují nastavené standardy kvality a relevanci. Administrátor má navíc možnost delegovat své pravomoci jiným uživatelům, což přispívá k udržení flexibilního a efektivního provozu platformy.
- **Podporovatel**, tvořící základní kámen celého systému, přispívá finančními prostředky na podporu vybraných kampaní. Mají svobodu ve výběru projektů, které chtějí podpořit, a mohou kdykoliv přehodnotit svá rozhodnutí a stáhnout své příspěvky, pokud kampaň ještě nebyla oficiálně uzavřena. Tato možnost dodává systému větší dynamiku a umožňuje podporovatelům efektivněji spravovat jejich investice.

Zabezpečení a transparentnost jednotlivých transakcí na platformě je zajištěno využitím technologie Solana blockchain. Všechny transakce jsou zpracovány a potvrzovány skrze peněženku, přičemž primárně je používána Phantom Wallet, která je blíže popsána v kapitole *4.7 Phantom Wallet*.

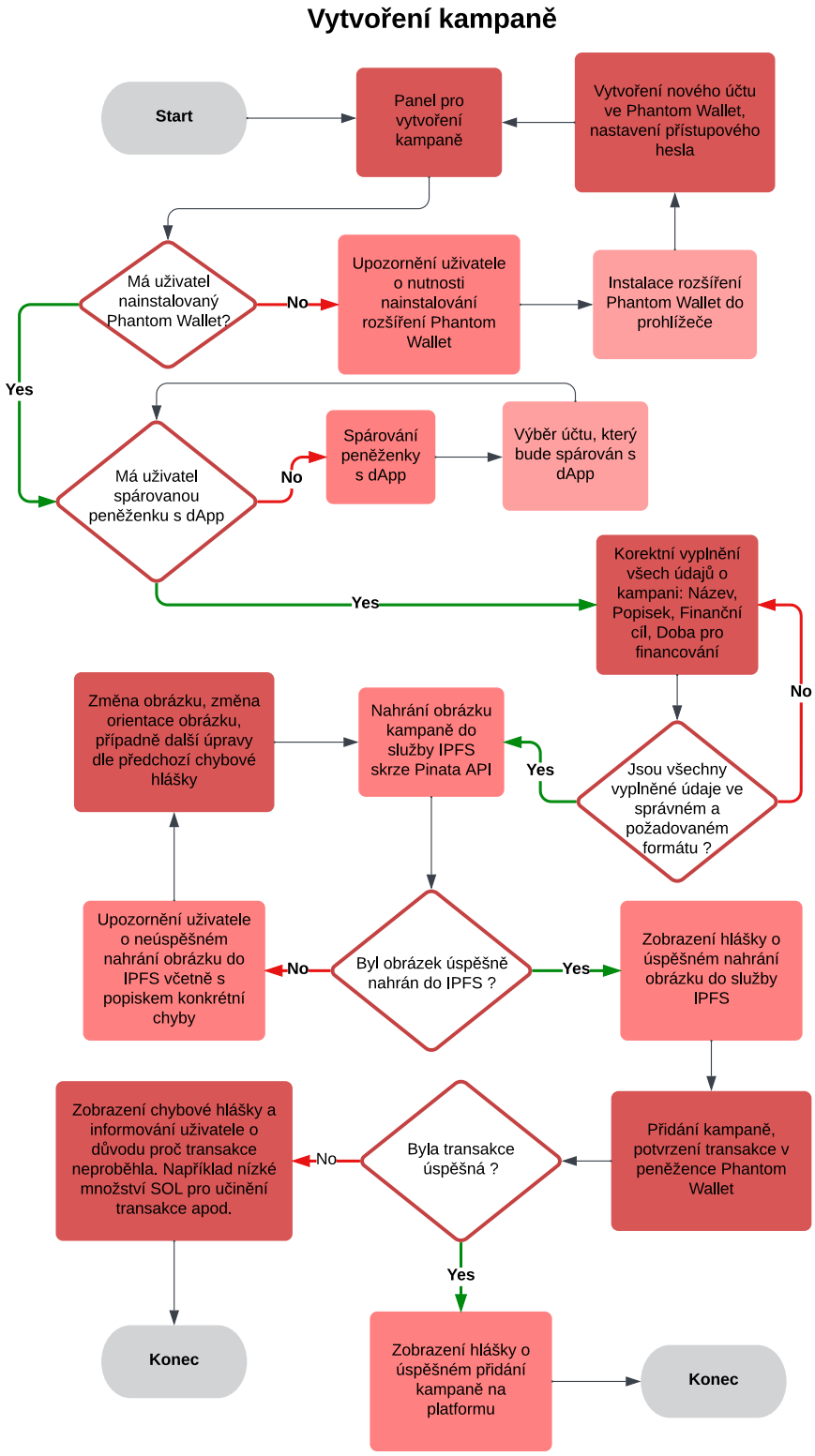
Pro zvýšení srozumitelnosti a jednoduchosti návrhu aplikace bylo vypracováno sedm aktivitních diagramů, které ilustrují kroky jednotlivých uživatelských interakcí. Aktivitní diagramy tak popisují průběh jednotlivých procesů pomocí detailních kroků.

Diagram zobrazený níže popisuje proces pro inicializaci administrátora, bez této inicializace nelze na platformě provádět žádné jiné operace. Administrátor platformy má na starosti řízení celé platformy, konkrétně schvalování nebo zamítání jednotlivých kampaní. Tyto funkce jsou pak popsány v následujících diagramech.



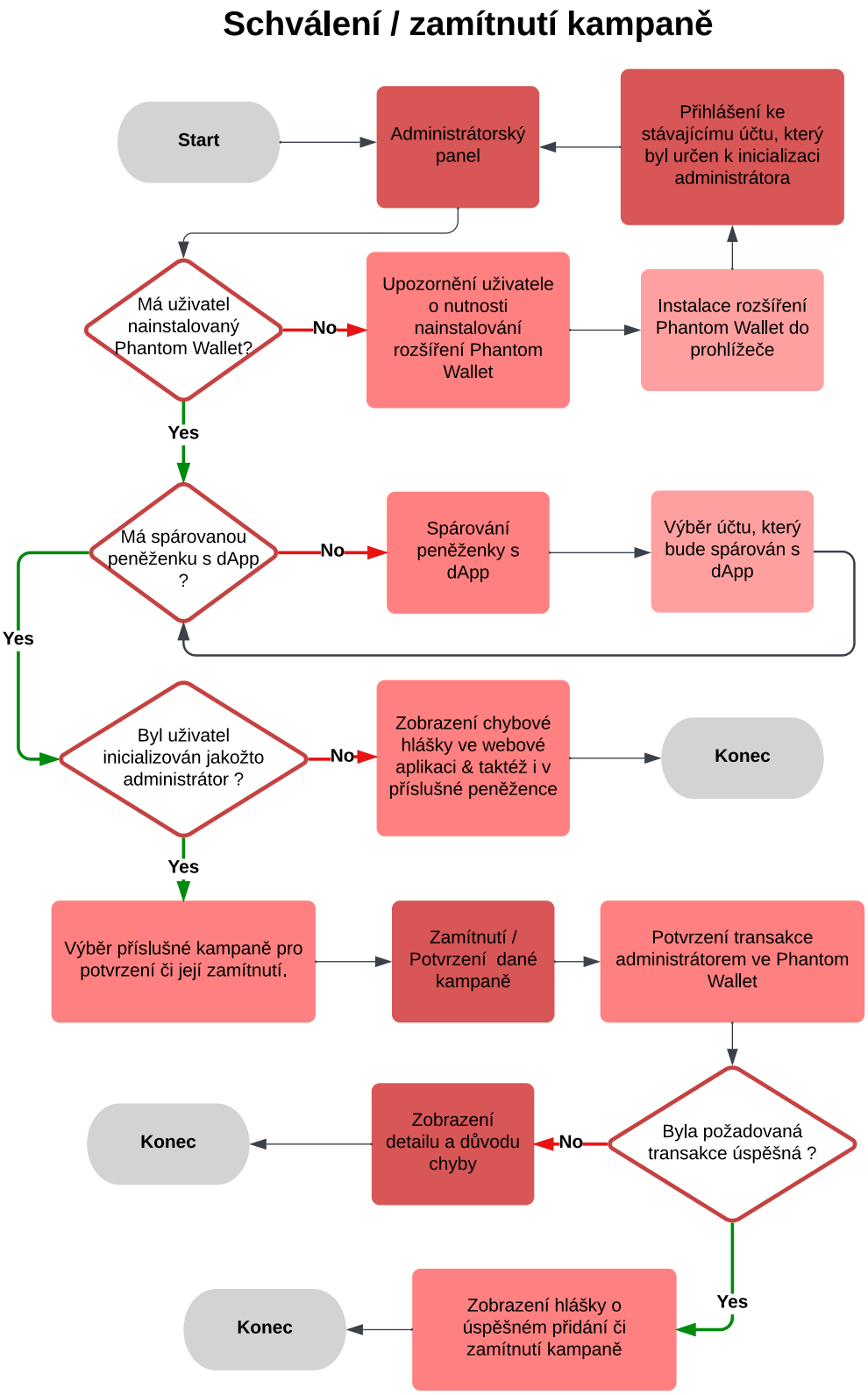

Obrázek 15. Diagram aktivit procesu inicializace administrátora

V tomto aktivitním diagramu organizátor vytvoří novou kampaň, při které vyplní základní údaje jako jsou název, popisek, doba trvání a požadovaný finanční cíl. Poté nahraje obrázek, který je uložen pomocí služby IPFS, a celý proces zakončí úspěšným potvrzením transakce v peněžence Phantom Wallet.



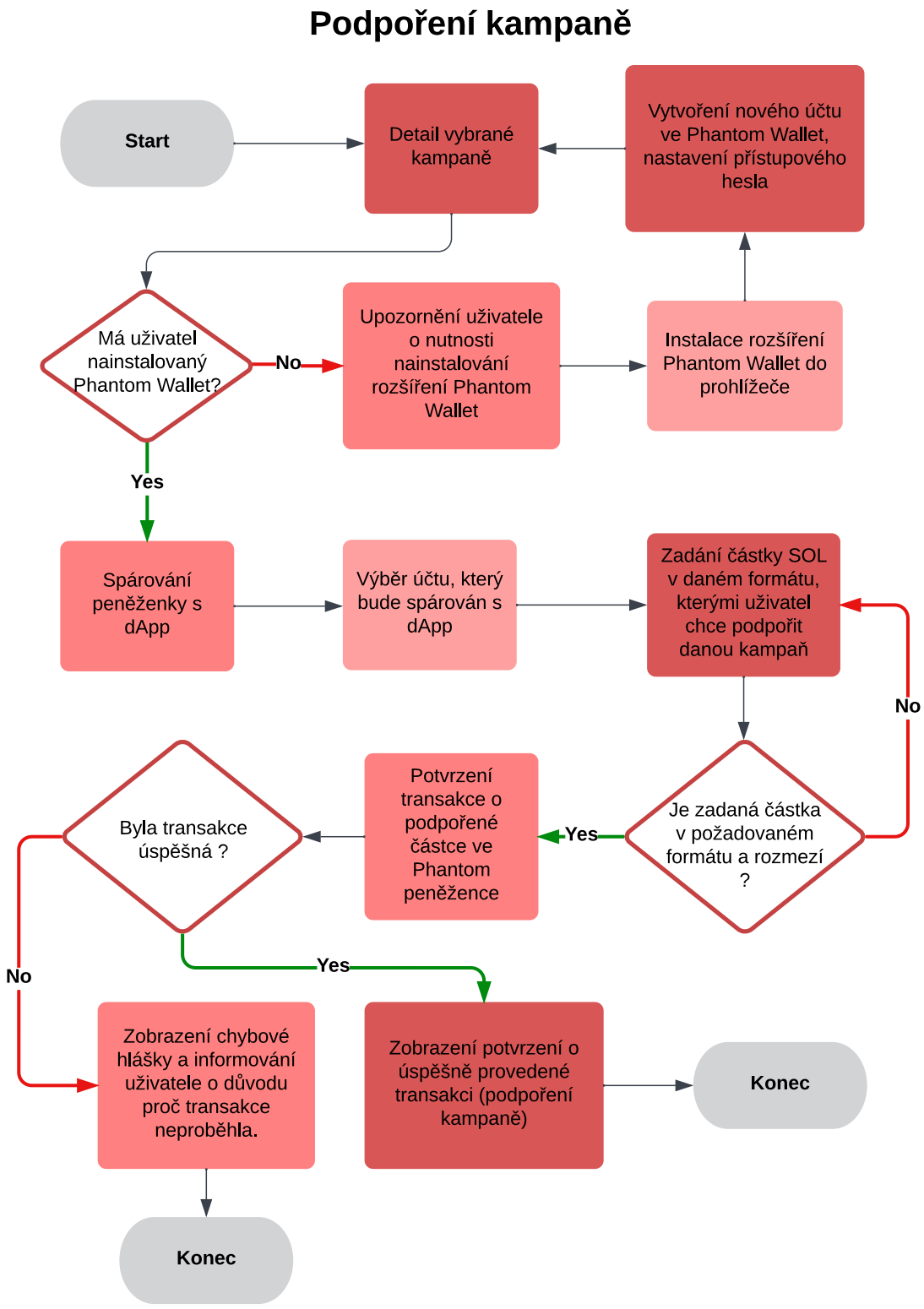

Obrázek 16. Diagram aktivit procesu pro vytvoření kampaně

Vytvořená kampaň nejprve musí projít procesem schvalování, kde uživatel s oprávněním administrátor kampaň buďto schválí či v opačném případě zamítne. Schválené kampaně jsou dynamicky propisovány a následně zobrazovány na úvodní obrazovce platformy.



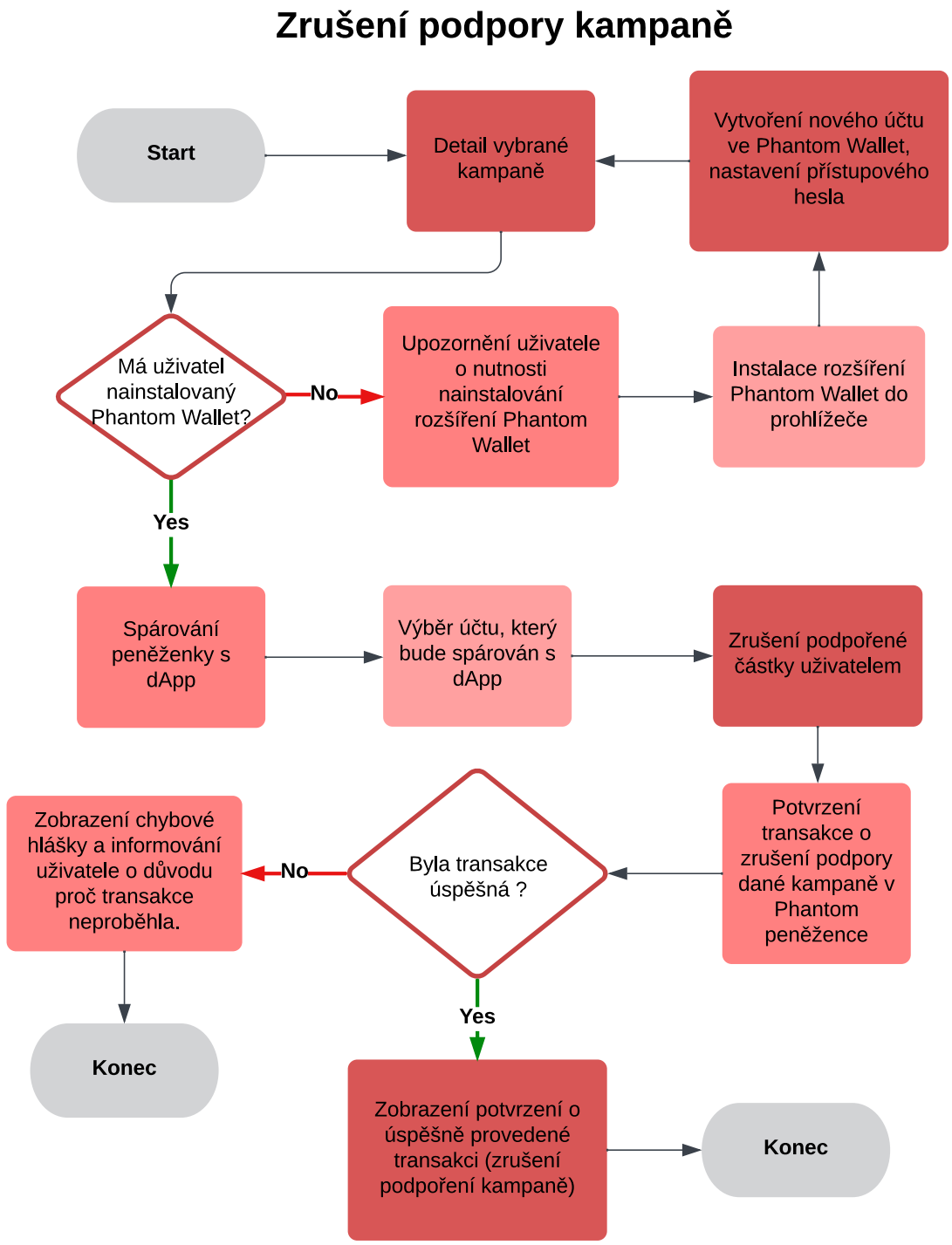

Obrázek 17. Diagram aktivit procesu pro schválení či zamítnutí kampaně

Schválené kampaně jsou dostupné pro možné udělení podpory od podporovatele. Ten pro její udělení přejde do detailu zvolené kampaně, následně zvolí částku, kterou chce vybranou kampaň podpořit, a to na základě svého osobního rozhodnutí.



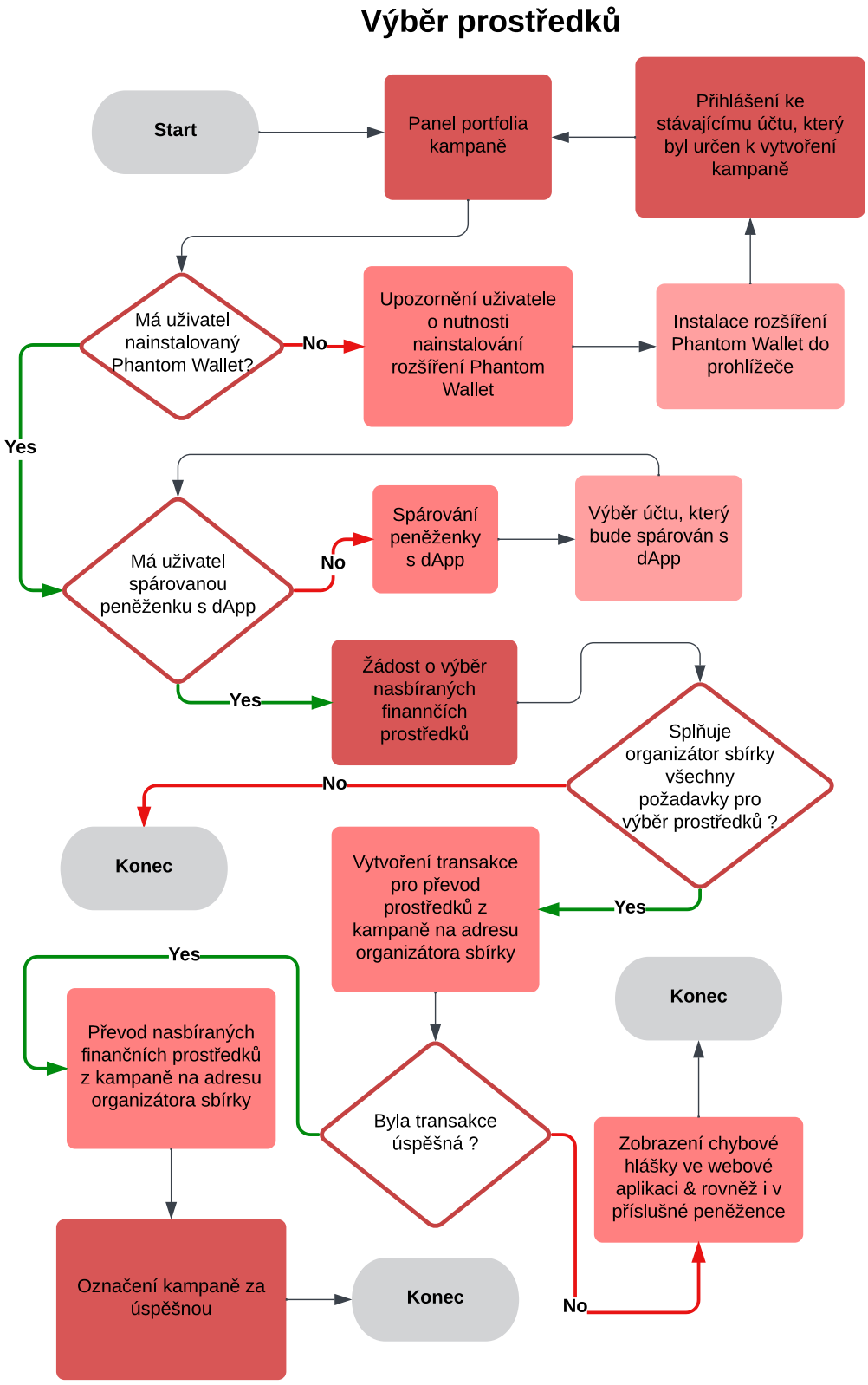

Obrázek 18. Diagram aktivit procesu pro podpoření kampaně

Diagram níže zobrazuje proces, kdy podporovatel změní své rozhodnutí o podpoře vybrané kampaně a rozhodne se podporu zrušit. V detailu kampaně, kde již poskytl svou finanční podporu, stačí pouze potvrdit zrušení podpory, a prostředky mu budou následně vráceny skrze peněženku Phantom Wallet.



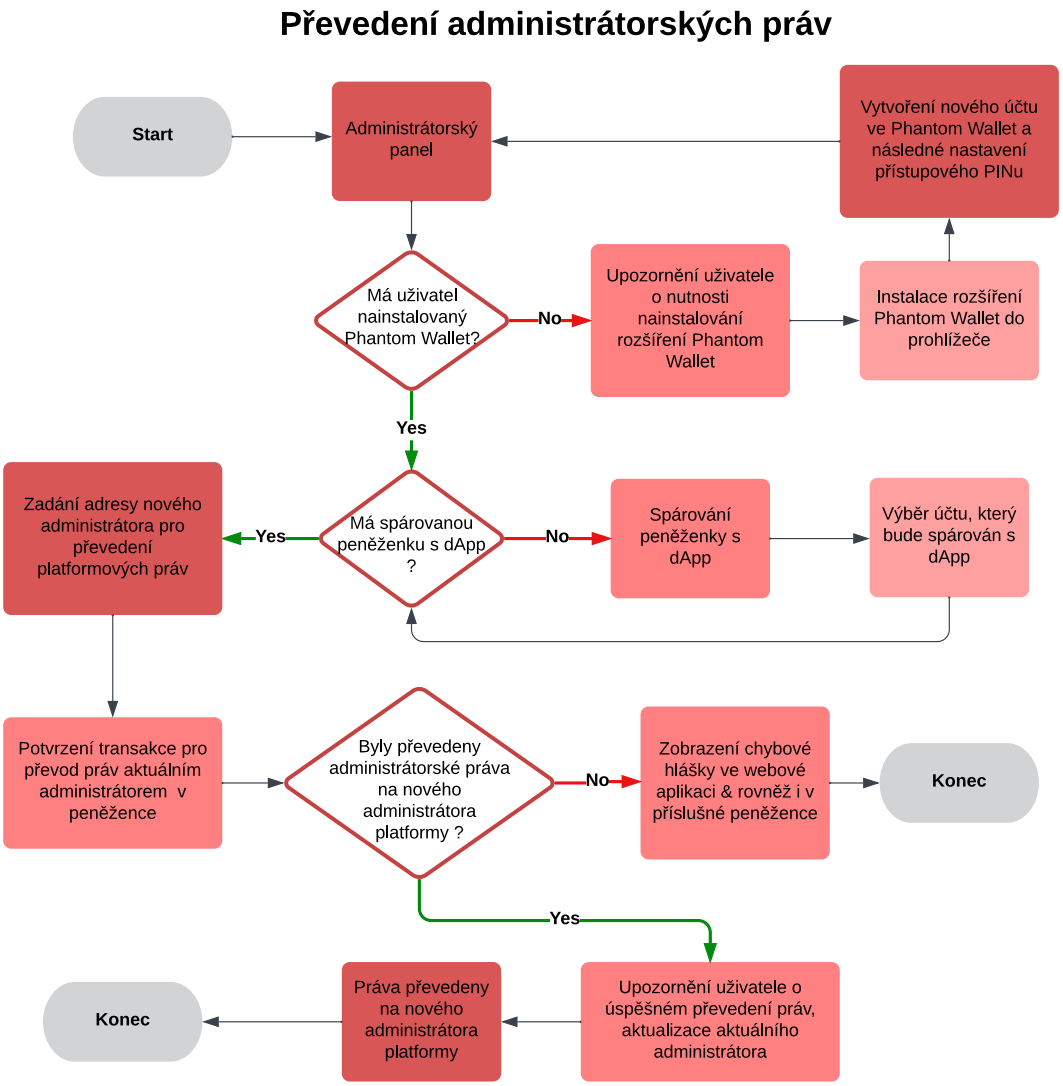

Obrázek 19. Diagram aktivit procesu pro zrušení podpory kampaně

Ukončení kampaně nastane až v momentě jejího výběru, kdy organizátor kampaně po splněný všech požadavků pro výběr prostředků požádá o převedení vybraných finančních prostředků na svoji adresu ve Phantom Wallet. Tento proces je zobrazen na diagramu níže.



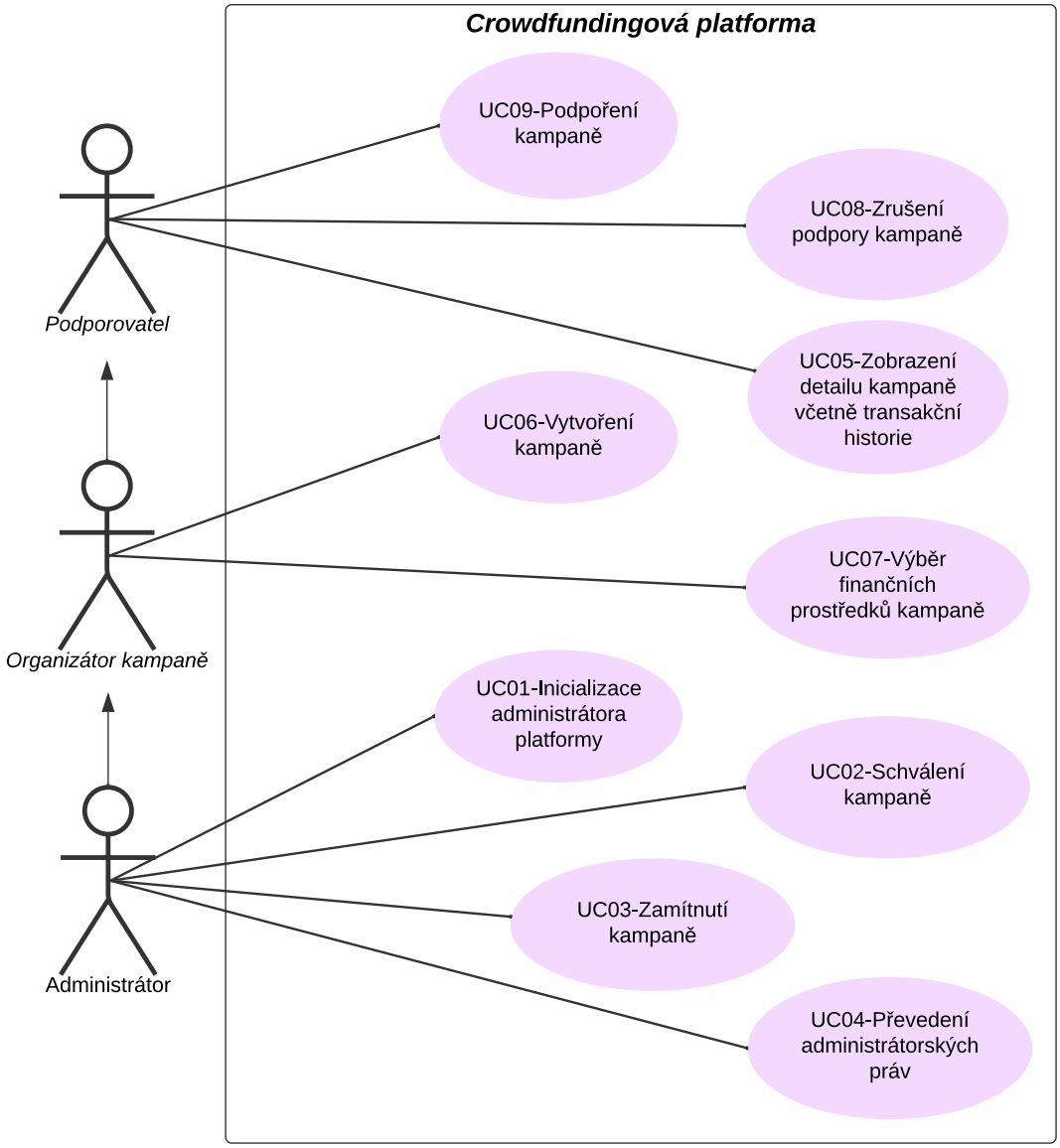

Obrázek 20. Diagram aktivit procesu pro výběr prostředků

Diagram zobrazený níže popisuje proces, kdy současný administrátor se snaží převést svá oprávnění na jiného uživatele platformy. Po úspěšném dokončení tohoto procesu jsou současnému administrátorovi odebrána veškerá práva a stává se z něj obyčejný uživatel, zatímco administrátorská oprávnění byla převedena na adresu uživatele, kterou definoval při převodu.




Obrázek 21. Diagram aktivit procesu pro převedení administrátorských práv

Pro lepší znázornění funkcí vyžadovaných pro realizovanou aplikaci a její správnou funkčnost byl vytvořen diagram případů užití, zobrazený na obrázku *22. Diagram případů užití vytvářené platformy*. V tomto diagramu jsou hlavními aktéry organizátor kampaně, administrátor a podporovatel. Z hlediska financí je nejdůležitější aktér podporovatel kampaně, který buďto může danou kampaň podpořit a na základě svého rozhodnutí následně podpořenou částku zrušit. Mezi aktivitu organizátora kampaně patří vytvoření a následný výběr finančních prostředků z kampaně. Administrátor se následně stará o svou inicializaci na platformě a taktéž o schvalování či zamítání jednotlivých kampaní. Závěrnou funkcionalitou je převedení administrátorských práv na jiného uživatele.



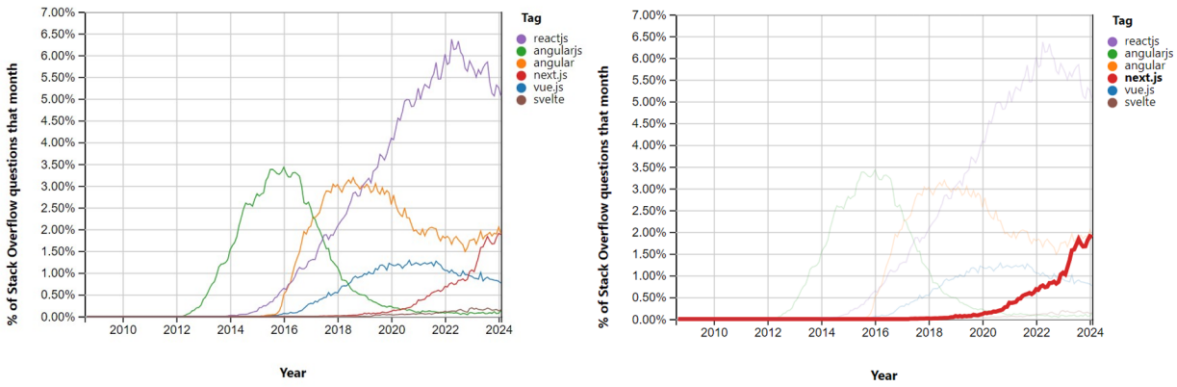

Obrázek 22. Diagram případů užití vytvářené platformy

## 4.3 Zdůvodnění výběru daného front-end frameworku

Pro vývoj decentralizované crowdfundingové aplikace byla klíčová volba vhodného front-end frameworku. Důležitým kritériem byla rovněž kompatibilita s knihovnami pro práci s blockchainem, jako je @solana/web3.js [41], ale i s dalšími specifickými nástroji, jako je @project-serum/anchor [42] pro Solana ekosystém.

Při výběru byl také brán v úvahu současný trend a popularita různých frameworků. Tyto faktory jsou důležitým ukazatelem aktuálního postavení na trhu a nabízejí bohaté zdroje díky aktivním vývojářským komunitám. To v konečném důsledku přispívá k vytvoření pevného základu pro budoucí rozvoj a podporu aplikace.

Na základě analýzy aktuálních trendů, které jsou zobrazeny na přiloženém obrázku *23. Analýza trendů vývoje popularity front-end frameworků*, bylo rozhodnuto zvolit framework Next.js pro jeho rostoucí popularitu a přijetí ve vývojářské komunitě. Tento nárůst zájmu je doložen výrazným vzestupem procentuálního zastoupení otázek týkajících se frameworku Next.js na platformě Stack Overflow. Zvýšený zájem o framework může být interpretován jako náznak jeho kvality a spolehlivosti, a také jako příslib kontinuální podpory a rozvoje ze strany vývojářské komunity [43]; [44].




Obrázek 23. Analýza trendů vývoje popularity front-end frameworků [39]

Zároveň byl Next.js [43] zvolen pro crowdfundingovou aplikaci na Solana blockchainu, protože nabízí významné výhody v rychlosti načítání a optimalizaci uživatelského prostředí díky server-side renderingu. Tato technologie nejen zlepšuje celkovou reaktivitu a dostupnost aplikace, ale také posiluje její bezpečnostní aspekty, což je v kontextu blockchainových aplikací zcela nezbytné. Kromě toho Next.js usnadňuje integraci s klíčovými blockchainovými knihovnami a nástroji specifickými pro ekosystém Solana, což urychluje vývoj a zvyšuje efektivitu celého projektu.

## 4.4 Příprava vývojového prostředí

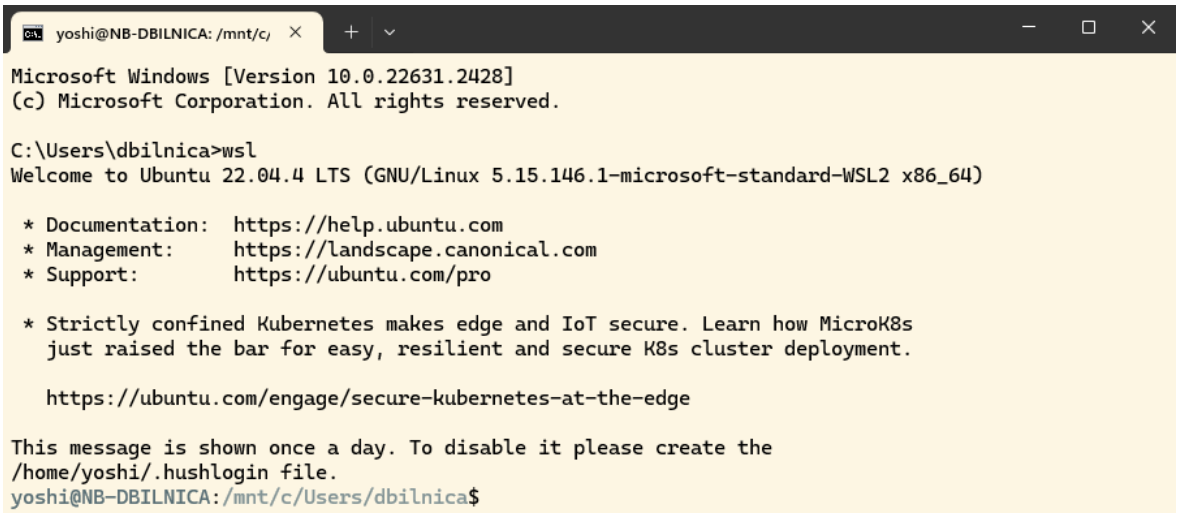
Pro úspěšný vývoj a správu decentralizovaných aplikací na Solana blockchainu je prvním a nezbytným krokem instalace Solana CLI. Solana CLI je mocný nástroj, který vývojářům umožňuje efektivně komunikovat s blockchainem, spravovat klíče, odesílat transakce a provádět mnoho dalších akcí potřebných během vývojového cyklu. Následující podkapitoly poskytnou postup, jak Solana CLI nainstalovat a připravit tak pracovní prostředí pro vývoj [45].

### 4.4.1 Instalace WSL pro použití se Solana CLI

Windows Subsystem for Linux umožňuje vývojářům na Windows pohodlně spouštět a využívat Linuxové nástroje, což zjednodušuje používání Solana CLI pro vývoj decentralizovaných aplikací. Pro instalaci WSL stačí spustit příkaz níže v terminálu Windows [46]:

```
$ wsl --install
```

Po restartování počítače lze snadno dokončit nastavení vybrané distribuce a přejít k vývoji na platformě Solana bez toho, aniž by bylo nutné opouštět prostředí Windows.




Obrázek 24. Terminál se spuštěnou distribucí Ubuntu za pomocí WSL

### 4.4.2 Instalace jazyka Rust a Cargo

Rust [64] je programovací jazyk, vyznačující se vysokou mírou bezpečnosti a efektivity, což jej činí zcela ideálním pro vývoj programů na Solana platformě. Jeho instalace je jednoduchá s využitím Rustup, což je nástroj nejen pro instalaci Rustu, ale i pro jeho správu. Při instalaci Rustu je automaticky nainstalován i Cargo, balíčkovací systém Rustu, který usnadňuje správu a sestavování projektů [47]; [48].

Pro instalaci Rustu a Cargo v prostředí WSL stačí spustit následující příkaz v terminálu se spuštěnou distribucí WSL:

```
$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

Tento příkaz tak stáhne a následně spustí instalační skript, který provede veškeré potřebné kroky. Po dokončení instalace je možné instalaci ověřit pomocí příkazu:

```
$ rustc --version
```

### 4.4.3 Instalace Node.js

Nodej.js je prostředí postavené na JavaScriptovém enginu, které umožňuje spouštění JavaScriptu určeného jako Server-Side[16]. Pro vývoj aplikací na Solana platformě, zejména při využívání Solana Web3.js knihoven, je Node.js nezbytný nástroj. Jeho instalace, tak zajišťuje, že vývoj a následné testování bude moci probíhat lokálně [41]; [49].

Pro instalaci Node.js v prostředí Windows Subsystem for Linux je doporučeno použít Node Version Manager, který umožňuje snadnou správu různých verzí Node.js. Instalace NVM se provádí prostřednictvím následujícího příkazu v terminálu [46]; [49]:

```
$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/master/install.sh
| bash
```

Následně lze ověřit instalace NVM pomocí příkazu:

```
$ nvm --version
```

Po provedení těchto operací je následní možné nainstalovat Node.js v poslední stabilní verzi pomocí následujícího příkazu:

```
$ nvm install --lts
```

Jako poslední krok je doporučeno ověření instalace Node.js včetně balíčkovací součásti a tím je Node Package Manager.

```
$ node --version

$ npm --version
```

[16] **Server-Side** – tato metoda se týká operací a procesů, které probíhají na serveru, jako je například zpracování dat a generování obsahu pro klienty [70]

### 4.4.4 Instalace Solana CLI

Solana Command Line Interface je základním nástrojem pro vývojáře pracující na Solana platformě. Toto Interface umožňuje správu účtů, nasazování programů, interakci s blockchainem a mnoho dalších důležitých operací. Pro efektivní práci se Solana aplikacemi je instalace Solana CLI nezbytnou součástí workflow [45].

Solana poskytuje jednoduchý skript, pro rychlou a snadnou instalaci CLI. Spuštění následujícího skriptu stáhne a nainstaluje aktuálně nejnovější verzi Solana CLI [45]:

```
$ sh -c "$(curl -sSfL https://release.solana.com/v1.18.4/install)"
```

Po dokončení instalace je možné ověřit, zda-li bylo Solana CLI úspěšně nainstalováno a to příkazem:

```
$ solana --version
```

#### *4.4.4.1 Konfigurace Solana CLI*

Po instalaci je důležitá správná konfigurace Solana CLI pro správnou práci s blockchainem a následujícím nasazením aplikací. Toto nastavení zahrnuje především nastavení výchozího clusteru, který bude s CLI komunikovat. Pro následný efektivní vývoj a testování programu je často využívána síť *devnet* [50].

Pro následné vývojové a testovací účely je *devnet* nastaven pomocí příkazu:

```
$ solana config set --url https://api.devnet.solana.com
```

V případě potřeby připojení k lokálnímu clusteru je použit následující příkaz:

```
$ solana config set --url localhost
```

Následné ověření, na jaký cluster je aktuální konfigurace nastavena je provedeno pomocí příkazu:

```
$ solana config get
```

#### *4.4.4.2 Vytvoření lokální systémové peněženky*

Pro nasazení programu s využitím Solana CLI je zapotřebí mít lokální Solana peněženku se Sol tokeny na pokrytí nákladů transakcí a taky za nájem pro ukládání dat na blockchainu. Vytvoření lokální systémové peněženky je umožněno pomocí příkazu [51]:

```
$ solana-keygen new
```

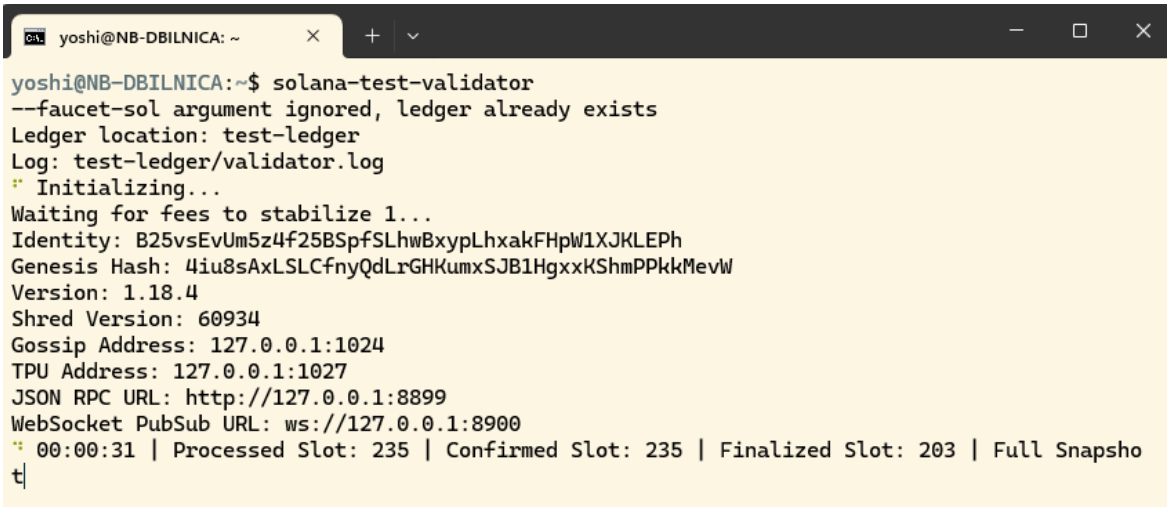
Po provedení bude vytvořena peněženka umístěná na této výchozí adrese *~/.config/solana/id.json*. Následně je zapotřebí si na peněženku alokovat určité množství tokenů, které slouží k pokrytí transakčních poplatků a nákladů spojených s ukládáním dat do blockchainu během vývoje a testování. Alokace tokenů je provedena pomocí tohoto příkazu, kde hodnota v příkazu znamená množství alokovaných tokenů [52]:

```
$ solana airdrop 3
```

#### *4.4.4.3 Spuštění Solana validátoru*

Solana Test Validator je nástroj, který vývojářům umožňuje spustit lokální verzi Solana validátora, speciálně určenou pro testování aplikací v izolovaném prostředí. Nabízí vývojářům možnost detailně otestovat své programy v bezpečném, virtuálně vytvořeném prostředí. Jeho používání je nezbytné pro rychlý a efektivní vývoj na Solana platformě. Pro spuštění konfigurovaného validátora je použit následující příkaz [53]:

```
$ solana-test-validator
```




Obrázek 25. Terminál se spuštěným lokálním validátorem

### 4.4.5 Instalace Anchor frameworku

Anchor framework poskytuje sadu nástroji a předdefinovaných abstrakcí, které zjednodušují vývoj programů na Solana platformě. Jeho instalace je tak klíčová pro vývojáře, kteří se zaměřují na robustní a bezpečné decentralizované aplikace na Solana platformě [54].

Pro jeho instalaci je důležitým předpokladem mít nainstalovaný Node.js, Rust a Solana CL. Anchor CLI se do systému instaluje pomocí balíčkového systému Cargo, který je součástí instalace jazyka Rust. Příkaz pro instalaci pomocí Cargo je následující [55]; [56]:

```
$ cargo install --git https://github.com/coral-xyz/anchor avm --locked --force
```

Po stažení a instalaci je možné ověřit úspěšné instalaci do systému, a to za pomocí příkazu:

```
$ anchor --version
```

#### *4.4.5.1 Sestavení aplikace pomocí Anchor frameworku*

Po instalaci Anchor frameworku, je dalším důležitým krokem sestavení vytvářeného projektu. Anchor tak poskytuje efektivní prostředí pro vývoj programů, které jsou přizpůsobeny požadavkům vývojářů pracujících na vytváření bezpečných decentralizovaných aplikací [55].

Proces nastavení projektu, včetně vytvoření základní strukturu a konfigurace je prováděno pomocí příkazu [55]:

```
$ anchor init <project_name>
```

Jakmile je projekt připraven a jednotlivé části programu jsou implementovány, tak je možné vytvořit sestavení. Zejména je důležité, aby zkompilovali všechny součásti projektu, včetně programů. Všechny tyto operace jsou provedeny tímto příkazem [55]:

```
$ anchor build
```

#### *4.4.5.2 Nasazení aplikace pomocích Anchor frameworku*

Po dokončení sestavení aplikace, je dalším krokem nasazení programu na Solana blockchain. Anchor z velké části zjednodušuje tento proces a umožňuje vývojářům efektivně nasazovat jednotlivé programy do blockchainové sítě [57].

Pro nasazení aplikace je nutné mít již úspěšně sestavený projekt, dále je zapotřebí nastavit konfigurační soubor *Anchor.toml*, který je vygenerován s projektem a nachází se v hlavním adresáři projektu. Důležitými body konfigurace pro nasazení jsou [57]:

- **[program.localnet]**: Tato konfigurace jednoznačně říká, že nasazovaný cluster bude localnet a tudíž se bude jednat o lokální vývoj včetně testování.
- **[provider]**: Určuje, který cluster bude aplikace při nasazování využívat a rovněž umožňuje definovat zdroj peněženky, která bude nasazení financovat.

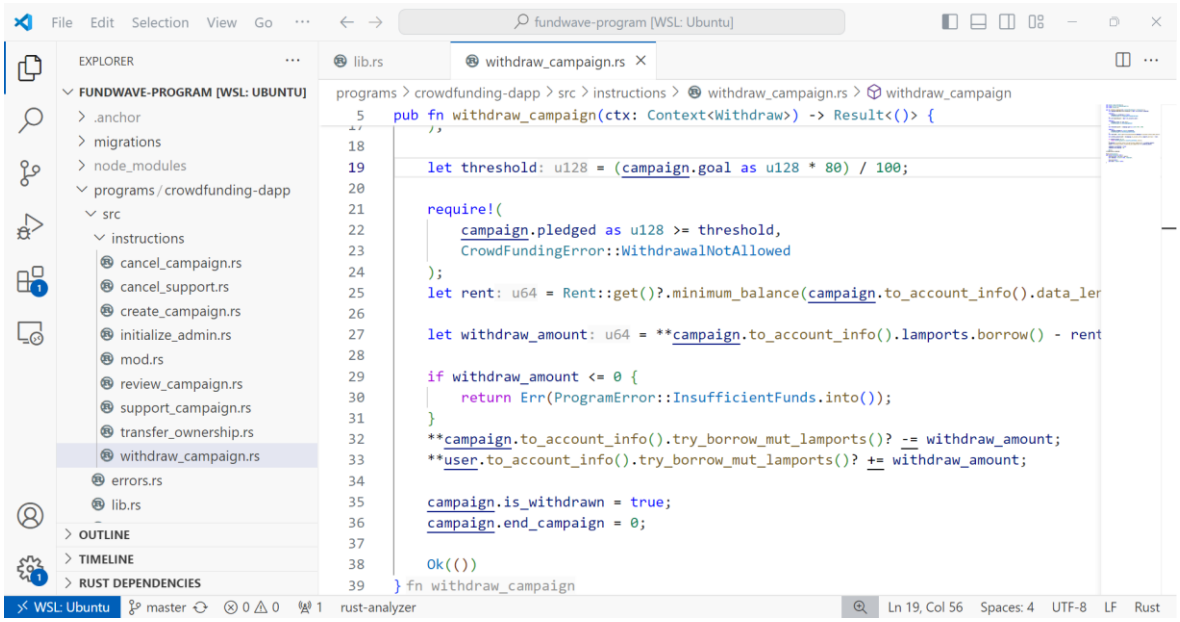
Po splnění těchto kroků je možné tedy aplikaci nasadit do prostřední Solana blockchain na definovaném clusteru, a to provedením příkazu:

```
$ anchor deploy
```

Po úspěšném nasazení programu do blockchainové sítě je důležité, aby každý uživatel měl možnost ověřit, že nasazený program skutečně odpovídá zdrojovému kódu, který byl použit. Ověření je důležité z hlediska integrity a transparentnosti nasazeného programu [65].

## 4.5 Vytváření programu ve Visual Studio Code

Pro efektivní vývoj na platformě Solana je klíčové mít správně nastavené vývojové prostředí. Volba padla na Visual Studio Code, oblíbené mezi integrovanými vývojovými prostředími, díky široké škále funkcí a rozšíření speciálně určených pro blockchainové aplikace. Jeho instalace a konfigurace představují zásadní kroky pro vývoj crowdfundingové aplikace [58].




Obrázek 26. Prostředí Visual Studio Code

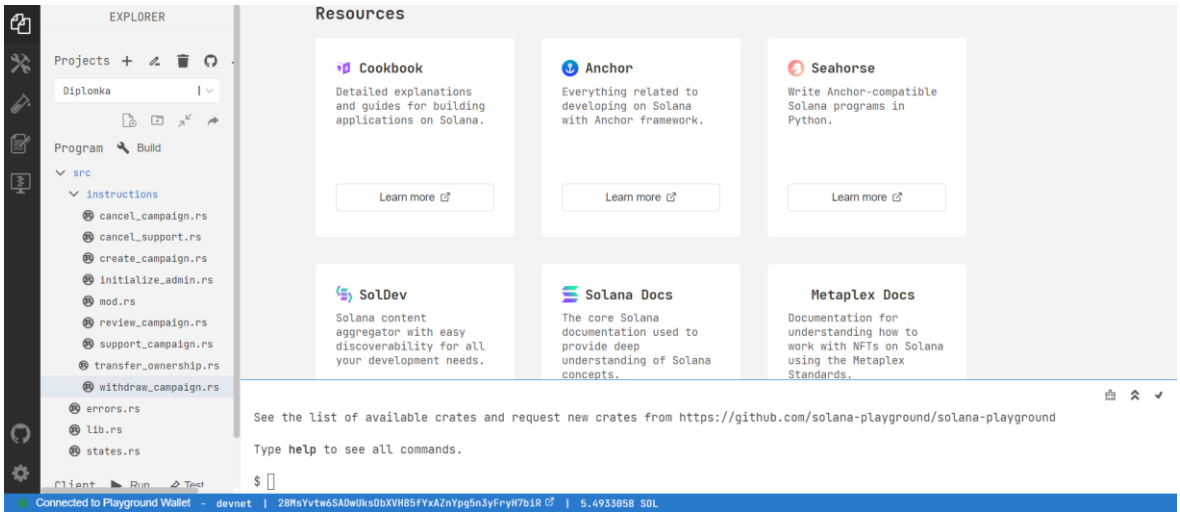
### *4.5.1.1 Rust Analyzer pro Visual Studio Code*

Rust Analyzer [40] je klíčové rozšíření pro Visual Studio Code, které je navržené pro vývoj v jazyce Rust, které vylepšuje efektivitu a pohodlí práce díky funkcím jako chytré doplnění kódu, zvýraznění syntaxe apod. Jeho instalace je nezbytným krokem pro vytváření kvalitních aplikací v Rustu a zároveň poskytuje pokročilé nástroje pro analýzu a refaktorování kódu.

## 4.6 Solana Playground

Rozhraní Solana Playground[17] je zcela moderní a plně funkční vývojové prostředí vyvíjené pouze pro platformu Solana, která je dostupná pouze jako webová aplikace. Toto IDE je speciálně navrženo pro programování v jazyku Rust [64] a Anchor [54], které jsou primárně využívány pro vývoj programů na Solana blockchainu. Jednou z hlavních předností Solana Playground je jeho zcela intuitivní grafické rozhraní a taktéž snadná instalace potřebných nástrojů a knihoven [23]; [42].

Solana Playground integruje Solana runtime, což umožňuje především rychlé lokální nasazení a testování programů. Mimo lokální uzel nabízí toto rozhraní připojení k dalším typům sítě, jako je například mainnet, či devnet [59]; [60].




Obrázek 27. Prostředí Solana Playground

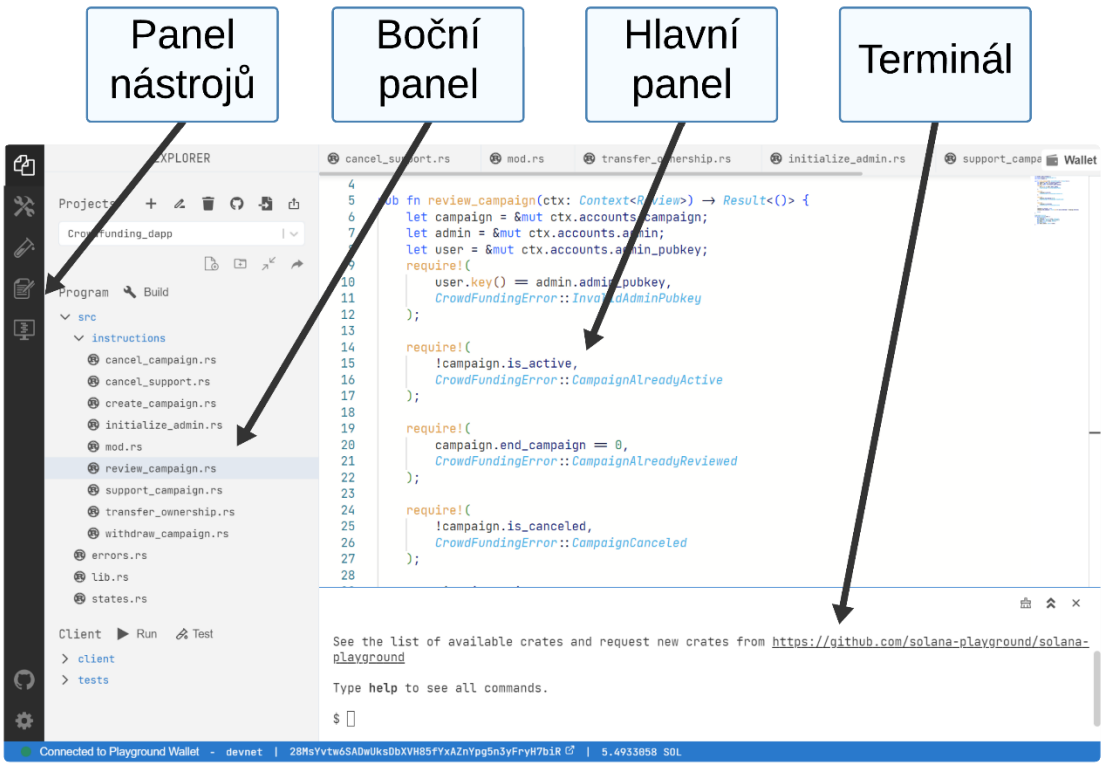
### 4.6.1 Rozhraní pro vytváření programů v Solana Playground

Struktura Solana Playground je přizpůsobena pro maximální přehlednost a zároveň uživatelskou přívětivost je navržena rozdělením jednotlivých částí do několika klíčových oblastí, z nichž každá poskytuje důležité funkce potřebné pro vývojáře. Tyto části tvoří [59]; [60]:

- **Panel nástrojů**: V této sekci uživatelé naleznou ikony pro procházení souborů, kompilaci, či nasazení programů do vybrané blockchainové sítě. Lze zde také následně vyvolávat funkce, které obsahuje nasazený program.

[17] Webové vývojové prostředí je dostupné na adrese *www.beta.solpg.io*

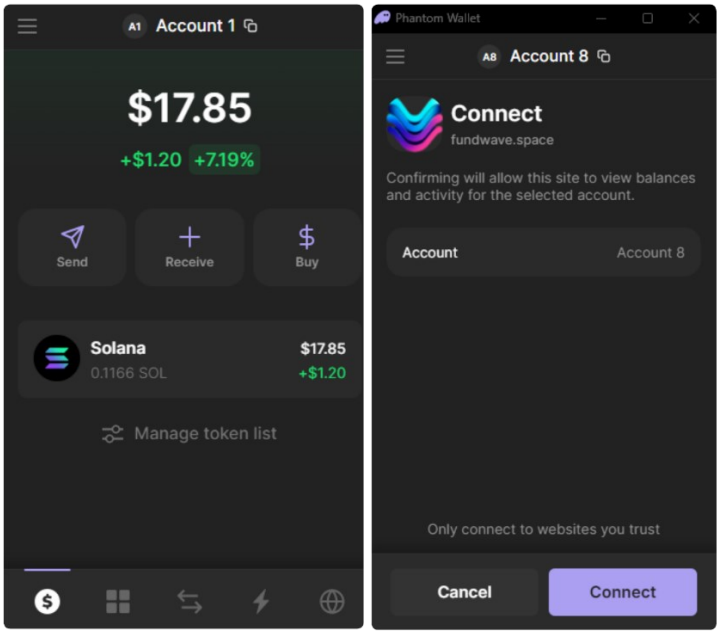
- **Boční panel**: Poskytuje především adresářový strom projektu, což usnadňuje navigaci a také správu souborů projektu.
- **Hlavní panel**: Tento panel je centrum pro oblast vytváření či editaci kódové stránky programu. Obsahuje rovněž i karty, které umožňují přepínání mezi jednotlivými otevřenými soubory či dokumenty.
- **Terminál**: Vypisuje výsledky jednotlivých prováděných operací. Může zobrazovat chybové hlášky při sestavování aplikace, nebo pokud dojdu v transakci nastane chyba, tak zde zobrazí detailní informace. Rovněž po nasazení programu zobrazí podrobné informace o stavu nasazení a funkčnosti programu.




Obrázek 28. Části prostředí Solana Playground

## 4.7 Peněženka Phantom Wallet

Phantom Wallet je populární prohlížečová kryptoměnová peněženka určená primárně pro ekosystém blockchainu Solana, která umožňuje uživatelům snadno interagovat s dApp. Mezi hlavní pozitiva této aplikace patří intuitivní a uživatelsky přívětivé prostředí, dále je třeba zohlednit silnou bezpečnostní architekturu této aplikace. Kromě základních operací jako je odesílání, či přijímání kryptoměny Solana umožňuje například i staking mincí a tokenů založených na Solana blockchainu. Díky svému přívětivému spojení s decentralizovanými aplikacemi, či burzami většina uživatelů zvolí právě tuto peněženku [9]; [11].




Obrázek 29. Připojení Phantom Wallet k dApp

## 4.8 IPFS

IPFS neboli InterPlanetary File System je distribuovaný systém pro ukládání a rovněž přístup k datům, který funguje na principu decentralizované sítě jednotlivých uzlů, které se nacházejí v dané síti. Každý vložený soubor má svůj unikátní hash, který slouží jako adresa pro lokalizaci souboru v síti. Hlavní výhodou oproti klasickým centralizovaným službám je odolnost vůči možným výpadkům a zároveň jsou vkládaná data odolná vůči cenzuře, což je zejména vhodné při použití této služby s dApp [62].

## 4.9 Next.js

Jedná se o moderní webový framework založený na platformě React, který umožňuje snadný a efektivní vývoj webových aplikací. Nabízí podporu pro server-side rendering a statické generování stránek, což zlepšuje především rychlost při načítání stránek a celkovou uživatelskou přívětivost s aplikacemi vytvořenými pomocí tohoto frameworku. Především díky své modulární architektuře poskytuje sady nástrojů a pluginů, které vývojářům usnadňují integraci s různými back-endovými technologiemi a podpoře pro API. V neposlední řadě je zapotřebí zmínit i rychlé obnovení stránek a možnosti ukládat operace do stavů, se kterými lze dále pracovat [43].

# 5 IMPLEMENTACE A POSTUP VÝVOJE

Cílem této diplomové práce je nejen teoreticky zkoumat možnosti využití blockchainu pro crowdfunding, ale především prakticky realizovat decentralizovanou aplikaci, která tyto možnosti využívá. Klíčovým krokem je implementace aplikace na robustní open-source platformě Solana blockchain, jejíž vlastnosti – jako jsou rychlost, nízké transakční poplatky a škálovatelnost ji přednostně předurčují pro podporu efektivních a transparentních crowdfundingových projektů.

Následující části budou zaměřeny na technické aspekty aplikace, popis její architektury, volbu dalších nezbytných souborů technologií a detailní průběh vývoje. V další části budou představeny jednotlivé fáze vývoje od inicializace projektu, přes definování funkcí a komponent aplikace, vývoj uživatelského rozhraní, až po back-endovou integraci a v neposlední řadě celkové testování aplikace.

## 5.1 Struktura a základní funkce pro Solana program

V rámci vývoje decentralizovaných aplikací na Solana blockchainu je nezbytné porozumění základním stavebním blokům a organizační struktuře Solana programů. Tyto základní aspekty jsou klíčové pro konstrukci jakéhokoliv efektivního programu, umožňují vývojářům plně využít unikátní vlastnosti a výkonnost této platformy. Porozumění modulární architektuře, implementaci logiky a správné manipulaci s datovými strukturami jsou zásadní pro vytváření robustních, bezpečných a škálovatelných aplikací, které mohou vyhovět požadavkům moderních blockchainových řešení.

### 5.1.1 Modulová organizace programu

Soubor *mod.rs* je ve frameworku Anchor pro ekosystém Solana aplikace zásadní pro definování modulové struktury programu. Jedná se o místo, kde jsou seskupeny všechny hlavní komponenty programu, čímž se stanoví jasná organizace kódu a usnadňuje se jeho správa.

#### *5.1.1.1 Struktura souboru mod.rs*

Ve vytvářeném programu, *mod.rs* slouží především jako rejstřík pro všechny klíčové moduly a poskytuje k nim přímý přístup. Níže je uvedena implementace *mod.rs*, která ilustruje tento systém organizace.

Kód 3. Import a modulární struktura funkcí

```
pub use cancel_support::*;
pub mod cancel_support;

pub use create_campaign::*;
pub mod create_campaign;

pub use initialize_admin::*;
pub mod initialize_admin;
// …
```

Každý *pub mod* příkaz deklaruje existenci modulu, zatímco pub use umožňuje ostatním částem aplikace importovat funkce, struktury nebo konstanty definované v daném modulu. Takovéto uspořádání podporuje modularitu a znovupoužitelnost kódu, což jsou klíčové postupy pro udržitelný a efektivní vývoj aplikace.

#### *5.1.1.2 Význam jednotlivých modulů v souboru mod.rs:*

- **cancel_support**: Modul obsahující logiku pro zrušení podpory kampaně uživatelem.
- **create_campaign**: Umožňuje organizátorům tvorbu nových kampaní na platformě.
- **initialize_admin**: Zodpovídá za inicializaci a nastavení administrátorských práv.
- **cancel_campaign**: Poskytuje funkcionality pro zrušení již založené kampaně.
- **review_campaign**: Obsahuje procesy pro přezkoumání a schválení kampaní.
- **support_campaign**: Umožňuje uživatelům finančně přispět jednotlivým kampaním.
- **transfer_ownership**: Zajišťuje přenos vlastnictví administrátorských práv.
- **withdraw_campaign**: Umožňuje organizátorům kampaní vybrat nasbírané finanční prostředky.

### 5.1.2 Implementace logiky programu

V rámci Anchor frameworku slouží *lib.rs* jako kritický soubor definující externí rozhraní Solana programu. Tento soubor rovněž slouží k propojení modulů programu a reakci na volání funkcí a transakce odeslané uživateli ze sítě Solana blockchain. Struktura a obsah tohoto souboru jsou proto klíčové pro funkčnost celé aplikace.

#### *5.1.2.1 Struktura souboru lib.rs*

Soubor *lib.rs* je klíčový pro definici struktury a funkčnosti vytvářeného Anchor programu pro crowdfundingovou aplikaci. Importy z *crate::instructions::** poskytují základ pro

akce programu, zatímco makro *#[program]* vytváří modul obsahující veřejné API pro interakci s blockchainovou sítí. Strukturu souboru *lib.rs* tak tvoří:

- **Moduly (pub mod)**: errors, instructions, a states jsou základní pro organizaci logiky a stavu programu.
- **Globální identifikátor (declare_id!)**: Unikátní ID programu zajišťuje jeho rozpoznání v síti.
- **API funkce (#[program])**: Obsahuje definici všech funkcí pro interakci účastníků platformy, včetně administrátorů, organizátorů kampaní a podporovatelů. Mezi klíčové funkce patří například *campaign_support* pro podporu kampaní, *ownership_transfer* pro převod správních práv, a další operace důležité pro správu a účast na platformě.

Tato struktura umožňuje efektivní správu a interakci programu crowdfundingové aplikace se Solana blockchainem.

Kód 4. Hlavní program a definice funkcí crowdfundingové aplikace

```
use crate::instructions::*;
use anchor_lang::prelude::*;
pub mod errors;
pub mod instructions;
pub mod states;

declare_id!("AKy9vrL3dUJjnkAzskjJSrcUketnfUAe2JAtmwRLc1i1");
#[program]
pub mod crowdfunding_dapp {
    use super::*;

    pub fn campaign_support(ctx: Context<Support>, amount: u64) ->
      Result<()> {support_campaign(ctx, amount)
    }

    pub fn ownership_transfer(ctx: Context<TransferOwnerShip>,
      new_admin: Pubkey) -> Result<()> {
        transfer_ownership(ctx, new_admin)
    }
      // …
}
```

#### *5.1.2.2 Přehled funkcí v souboru lib.rs*

Každá funkce definovaná pod makrem *#[program]* představuje specifickou operaci ve vytvářeném Solana programu, hraje klíčovou roli ve funkcionalitě a správě celé aplikace.

Tyto funkce jsou základními stavebními kameny, umožňující všem uživatelům interagovat s programem a realizovat jeho hlavní účely. Níže jsou detailně popsány klíčové funkce:

- **campaign_cancel**: Tato funkce umožňuje zrušení kampaně. Je zásadní pro správu kampaní a umožňuje tak administrátorovi zrušit kampaň v případě, že kampaň není relevantní pro schválení.
- **support_cancel**: Umožňuje podporovatelům stáhnout již poskytnutou finanční podporu kampani. To zvyšuje důvěru v platformu tím, že poskytuje možnost revize financování ze strany podporovatelů.
- **campaign_create**: Funkce určená k založení nové kampaně s definovanými parametry jako jsou název, popis, finanční cíl, trvání kampaně a odkaz na obrázek uložený pomocí IPFS [62]. Tato funkce je základním stavebním kamenem pro jakoukoliv crowdfundingovou platformu.
- **admin_initialize**: Zajišťuje prvotní inicializaci a konfiguraci administrátorských práv, což je kritické pro řízení a udržení integrity celé platformy.
- **campaign_review**: Umožňuje administrátorovi přezkoumání a schválení nových kampaní, což je zásadní pro zajištění kvality a důvěryhodnosti kampaní prezentovaných na platformě.
- **campaign_support**: Poskytuje mechanismus, přes který mohou uživatelé přispívat financemi na vybrané kampaně, přičemž podpora od komunity je zásadní pro jejich úspěch.
- **ownership_transfer**: Zajišťuje přenos vlastnictví administrátorských práv na nového správce, klíčový proces umožňující adaptabilní řízení a udržení bezpečného prostředí na platformě.
- **campaign_withdraw**: Umožňuje organizátorům kampaní vybrat shromážděné finanční prostředky, což je závěrečný a nezbytný krok pro realizaci projektů financovaných skrze platformu.

### 5.1.3 Implementace datových struktur

Ve vývoji Solana programu, především těch vytvořených pomocí frameworku Anchor, hraje klíčovou roli správná definice a implementace datových struktur. Soubor s názvem *states.rs* je tak zásadní pro definování stavových struktur, které uchovávají data a stavy specifické pro aplikaci. Tento soubor je nezbytný pro správnou funkčnost a logiku celého programu, jelikož umožňuje sledování a správu stavů kampaní, uživatelů a transakcí.

Kód 5. Definice datových struktur crowdfundingové aplikace

```
use anchor_lang::prelude::*;

pub const MIN_NAME_LEN: usize = 3;
pub const MAX_NAME_LEN: usize = 50;
// …

#[account]
pub struct Campaign {
    pub name: String,
    pub description: String,
    pub image_ipfs_hash: String,
    pub goal: u64,
    pub pledged: u64,
    pub pledgers: Vec<Pledgers>,
    // …
}

#[account]
pub struct Admin {
    pub admin_pubkey: Pubkey,
    pub is_initialized: bool,
}
// …
```

#### *5.1.3.1 Základní konstanty a omezení v souboru states.rs*

V rámci správy dat a operací programu jsou kritické pevně stanovené hodnoty, které definují parametry pro různé aspekty kampaní. Tyto konstanty jsou nezbytné pro zajištění integrity dat a pro prevenci potenciálních problémů spojených s nevalidními vstupy. V souboru *states.rs* jsou specifikovány následující klíčové konstanty:

- **MIN_NAME_LEN a MAX_NAME_LEN**: Tyto konstanty určují rozmezí povolené délky názvu kampaně, kde *MIN_NAME_LEN* stanovuje minimální počet znaků a *MAX_NAME_LEN* maximální počet znaků, které může název kampaně obsahovat. Tato omezení jsou zavedena k zajištění, že názvy kampaní budou dostatečně informativní, avšak zároveň nebudou příliš dlouhé pro zpracování a celkovou přehlednost.
- **MIN_DESC_LEN a MAX_DESC_LEN**: Tyto konstanty definují minimální a maximální povolenou délku popisu kampaně. Minimální limit zajistí, že popis bude

dostatečně podrobný, aby poskytl uživatelům klíčové informace o kampani, zatímco maximální limit pomáhá omezit spotřebu paměti a zjednodušuje správu dat.

- **MAX_GOAL**: Tato konstanta určuje maximální povolenou výši finančního cíle kampaně. Stanovením této horní hranice se zabraňuje vytvářením nerealisticky vysokých cílů, které by nemusely být dosažitelné a mohly by vést k frustraci uživatelů crowdfundingové platformy.
- **MAX_DURATION**: Definuje maximální dobu trvání kampaně v sekundách, což umožňuje kampaním běžet po omezenou dobu. Toto omezení pomáhá zajistit, že kampaně budou mít jasně definovaný konec, což jednoznačně podporuje efektivní plánování a správu očekávání jak pro organizátory, tak pro podporovatele.

Výše uvedené konstanty a omezení jsou klíčové pro udržení strukturovaného a efektivního prostředí v rámci crowdfundingové platformy, pomáhají předcházet problémům spojeným s datovými vstupy a zajišťují tak, že všechny kampaně jsou vytvořeny a spravovány v souladu s předem stanovenými pravidly.

#### *5.1.3.2 Stavové struktury v souboru states.rs*

Stavové struktury jsou klíčové pro uchování a správu dat v Solana programu. Tyto struktury definují formát a typy informací, které jsou nezbytné pro fungování a interakci všech aspektů crowdfundingové platformy. Od údajů o kampaních, až po administrátorské přístupy, tyto struktury zajišťují, že program může efektivně sledovat stav a změny, což umožňuje transparentní a spravedlivou správu celého systému. V souboru *states.rs* jsou specifikovány následující struktury:

**Campaign**

Struktura *Campaign* představuje základ každé kampaně v rámci crowdfundingové platformy. Obsahuje komplexní sadu informací, které umožňují její plno charakterizaci a správu:

- **name**: Identifikuje kampaň unikátním názvem, který musí splňovat definované délkové omezení.
- **description**: Poskytuje detailní popis kampaně, včetně jejího účelu a cílů, s dodržením stanovené maximální délky.
- **image_ipfs_hash**: Uchovává hash obrázku kampaně uloženého na IPFS, což zvyšuje decentralizaci a odolnost dat.

- **goal**: Definuje finanční cíl kampaně, tedy částku, kterou se snaží shromáždit od podporovatelů.
- **pledged**: Zaznamenává celkovou shromážděnou částku, což poukazuje na dosavadní úspěch kampaně v získávání finančních prostředků.
- **pledgers**: Obsahuje seznam všech podporovatelů, včetně informací o jejich příspěvcích, což přispívá k transparentnosti financování.
- **duration**: Specifikuje časovou délku kampaně od jejího začátku do plánovaného konce.
- **end_campaign**: Časové razítko ukončení kampaně, po kterém již nelze přispívat.
- **owner**: Veřejný klíč vlastníka kampaně, který má právo na její správu a možnost vybrání shromážděných finančních prostředků.
- **is_active, is_canceled, is_pledged, is_withdrawn**: Logické stavy reflektující aktuální stav kampaně v různých fázích jejího životního cyklu.

**Admin**

Struktura Admin zastupuje administrátorskou roli s oprávněními potřebnými pro řízení a dohled nad celým programem. Tato struktura tak zahrnuje:

- **admin_pubkey**: Veřejný klíč, který slouží jako unikátní identifikátor pro administrátorský účet, umožňující autentizaci a oprávnění k provádění administrativních akcí v rámci programu.
- **is_initialized**: Tento logický stav zajišťuje, jestli byly administrátorské práva a účet správně nastaveny. Pokud není hodnota nastavena na platnou (*true*), vytváření kampaní a další klíčové operace na platformě jsou zablokovány, aby se zajistila správná autorizace a kontrola.

**Pledgers**

Struktura Pledgers představuje transparentní záznam o jednotlivých přispěvatelích v rámci kampaně, což je klíčové pro správu a evidenci finanční podpory. Tato struktura tak obsahuje:

- **pledger_pubkey**: Veřejný klíč přispěvatele, který slouží jako unikátní identifikátor pro každého, kdo přispěje do kampaně. Umožňuje sledovat a ověřovat příspěvky od jednotlivých uživatelů a zajišťuje transparentnost finanční podpory kampaní.

- **pledged_amount**: Částka, kterou daný přispěvatel věnoval kampani. Tento údaj je klíčový pro vyhodnocení úspěšnosti kampaně v dosahování jejího finančního cíle.

Tyto tři základní stavové struktury tvoří jádro Solana programu pro správu crowdfundingové platformy, poskytují základní mechanismy pro uchování, správu a sledování všech relevantních informací o kampaních a administraci. Díky nim je možné zajistit plynulý průběh kampaní, efektivní správu finančních prostředků a bezpečnou administraci platformy.

### 5.1.4 Definice chyb v programu

Soubor *errors.rs* v Anchor frameworku pro vytvářený Solana program slouží k definování výčtu chybových stavů (enumerates), které mohou nastat při provádění různých operací (transakcí). Tento mechanismus umožňuje programu jednoznačně identifikovat a reportovat specifické problémy, které se vyskytnou během jeho běhu.

Kód 6. Definice chybových kódů crowdfundingové aplikace

```
use anchor_lang::prelude::*;

#[error_code]
pub enum CrowdFundingError {
    #[msg("Campaign name must be between a specified range.")]
    InvalidCampaignNameLength,
    #[msg("Campaign description must be between a specified range.")]
    InvalidDescriptionLength,
    #[msg("Invalid campaign duration or goal.")]
    InvalidDurationOrGoal,
    #[msg("Campaign or description cannot be empty.")]
    InvalidNameOrDescription,
    #[msg("Invalid admin pubkey provided.")]
    InvalidAdminPubkey,
    #[msg("Campaign is not active.")]
    CampaignNotActive,
    #[msg("Campaign has ended.")]
    CampaignEnded
    // …
}
```

Enum *CrowdFundingError* zahrnuje chyby specifické pro aplikaci crowdfundingové platformy a pomáhá zajistit správnou interakci uživatelů s programem. Některé z definovaných chyb zahrnují:

- **InvalidCampaignNameLength**: Chyba indikující, že délka názvu kampaně je mimo povolený rozsah.

- **InvalidDescriptionLength**: Signalizuje, že délka popisu kampaně nesplňuje definované délkové omezení.
- **InvalidAdminPubkey**: Chyba nastane, pokud je poskytnut neplatný veřejný klíč administrátora.
- **CampaignNotActive**: Signalizuje, že kampaň není aktivní.
- **CampaignEnded**: Oznamuje, že kampaň již skončila.
- **ExcessSupportAmount**: Upozornění, pokud množství podpory přesahuje zbývající finanční cíl.
- **WithdrawalNotAllowed**: Indikuje, že vybrání finančních prostředků není povoleno, protože cíl kampaně nebyl dosažen.
- **AdminNotInitialized**: Upozorňuje, že administrátorský účet musí být inicializován před vytvořením kampaně.
- **UnauthorizedUser**: Signalizuje, že uživatel nemá potřebná oprávnění pro provedení akce.
- **InvalidIpfsHash**: Upozornění na neplatný IPFS hash poskytnutý pro obrázek dané kampaně.

Tyto a další definované chyby jsou klíčové pro řízení toku programu a umožňují uživatelům pochopit příčiny jednotlivých problémů, které mohou během interakce s programem nastat. Efektivní správa těchto chyb zvyšuje uživatelskou přívětivost a bezpečnost celé platformy.

## 5.2 Modifikátory funkcí Solana programu

V rámci Solana programů, zvláště těch vyvinutých prostřednictvím Anchor frameworku, hrají modifikátory funkcí klíčovou roli v zajištění správného chování a zabezpečení operací. Modifikátory, jako jsou podmíněné kontroly implementované pomocí makra *require!*, umožňují programům ověřit platnost určitých podmínek před provedením nebo pokračováním v jakékoli funkci. Tato metodika zásadně přispívá k bezpečnosti a odolnosti aplikace prostřednictvím prevence nevalidních či neoprávněných operací. Výše uvedený kód ilustruje několik příkladů použití modifikátorů funkcí:

- **Aktivní kampaň**: Tento modifikátor ověřuje aktivní stav kampaně. Pokud není kampaň aktivní, tak zastaví operaci a následně vyvolá chybu *CampaignNotActive*.

Kód 7. Ověření aktivního stavu kampaně

```
require!(campaign.is_active, CrowdFundingError::CampaignNotActive);
```

- **Kontrola konce kampaně**: Modifikátor ověřuje, že aktuální čas ještě nepřekročil časové razítko ukončení kampaně. Jestliže kampaň již skončila, vyvolá se chyba *CampaignEnded*, což brání v provádění akcí, kromě výběru vysbírané částky organizátorem kampaně.

Kód 8. Kontrola platnosti časového rámce

```
require!(
    campaign.end_campaign >= Clock::get()?.unix_timestamp,
    CrowdFundingError::CampaignEnded
);
```

- **Stav výběru finančních prostředků kampaně**: Tento modifikátor zajišťuje, že finanční prostředky z kampaně ještě nebyly vybrány. Pokud už byly prostředky vybrány, dojde k zablokování dalších pokusů o výběr a je vyvolána chyba *WithdrawnCampaign*.

Kód 9. Ověření nevybraných finančních prostředků kampaně

```
require!(
    campaign.is_withdrawn == false,
    CrowdFundingError::WithdrawnCampaign
);
```

Modifikátory funkcí, jako jsou podmíněné kontroly skrze *require!*, klíčově ověřují, zda jsou splněny podmínky pro provádění operací v Solana programech. Tato praxe pomáhá zajistit, že akce, jako je zahájení, ukončení, nebo finanční manipulace s kampaněmi, respektují aktuální pravidla a stavy, čímž podporuje spravedlivé a bezpečné prostředí pro všechny účastníky.

## 5.3 Implementace klíčových funkcí programu

Následující část je zaměřena na jednotlivé funkce, které tvoří jádro vytvářené decentralizované crowdfundingové platformy na Solana blockchainu. Tato kapitola rovněž slouží jako praktický průvodce klíčovými operacemi, které platforma nabízí, počínaje od založení kampaně přes její správu až po finální výběr shromážděných prostředků. Každá z funkcí, uložená v samostatných souborech jako je *cancel_campaign.rs*, *create_campaign.rs* a další,

je v následujících podkapitolách detailně vysvětlena a popsána s cílem osvětlit její klíčové vlastnosti, účel a dopad na uživatelské prostředí. Tato analýza jednotlivých funkcí tak poskytuje ucelený pohled na to, jak se každá z nich podílí na technické struktuře vytvářené platformy a jak jsou tyto funkce zásadní pro vedení a správu crowdfundingových projektů.

### 5.3.1 Inicializace administrátora

Inicializace administrátora je kritickým krokem pro správné nastavení a bezpečnostní základy vytvářené crowdfundingové platformy. Tato operace je realizována skrze soubor *initialize_admin.rs*, jehož hlavním účelem je založení administrátorského účtu, který má oprávnění spravovat celou platformu. Proces inicializace zahrnuje několik klíčových kroků, které zajišťují, že administrátorský účet je správně nastaven a chráněn. Klíčovými kroky inicializace se detailně zabývají následující podkapitoly.

#### *5.3.1.1 Ověření platnosti uživatelského veřejného klíče*

Funkce začíná kontrolou, zda klíč uživatele, který se pokouší inicializovat administrátorský účet, není výchozí klíč (tj. není neplatný nebo nulový). Tím se zajišťuje, že inicializaci provádí skutečný a identifikovatelný uživatel

Kód 10. Ověření platnosti veřejného klíče uživatele při inicializaci administrátorského účtu

```
require!(
    *ctx.accounts.user.key != Pubkey::default(),
    CrowdFundingError::InvalidAdminPubkey
);
```

#### *5.3.1.2 Ověření neinicializovaného administrátora*

V této fázi se provádí verifikace, zda administrátorský účet dosud nebyl aktivován. Tento krok brání možnosti, že by byl administrátorský účet nastaven vícekrát, což by mohlo vést k bezpečnostním komplikacím spojeným s přítomností několika účtů s administračními právy.

Kód 11. Kontrola již inicializovaného účtu

```
require!(
    !admin.is_initialized,
    CrowdFundingError::AdminAlreadyInitialized
);
```

#### *5.3.1.3 Nastavení a inicializace administrátorského účtu*

V rámci tohoto kroku se veřejný klíč uživatele volající operaci uloží do administrátorského účtu pod označením *admin_pubkey*, a zároveň se přepne logický stav *is_initialized* na hodnotu *true*. Tento proces efektivně převádí účet do stavu, kdy je plně operativní a schopen zastávat roli správce celé platformy.

Kód 12. Aktivace a přiřazení klíče administrátorského účtu

```
admin.admin_pubkey = *ctx.accounts.user.key;
admin.is_initialized = true;
```

#### *5.3.1.4 Konfigurace kontextu pro inicializaci administrátora*

Konfigurace kontextu InitializeAdmin určuje sadu účtů, které jsou nezbytné k úspěšnému nastavení administrátorské role. Jednotlivé účty kontextu tak tvoří:

- **admin**: Tato položka určuje vytvoření nového účtu pro administrátora. Proces zahrnuje specifikaci požadovaného prostoru pro účet a využití unikátních seedů k jeho identifikaci.
- **user**: Identifikuje uživatele, jenž inicializaci vyvolává na základě veřejného klíče.
- **system_program**: Odkazuje na základní systémový program Solana, který je nezbytný pro vytváření a správu účtů na blockchainu. Tato součást umožňuje, aby byly v rámci inicializace správně alokovány zdroje pro nově vznikající administrátorský účet.

Kód 13. Definice kontextové struktury pro inicializaci administrátorského účtu

```
#[derive(Accounts)]
pub struct InitializeAdmin<'info> {
    #[account(init, payer=user, space=Admin::max_size(),
      seeds=[ADMIN_SEED.as_bytes()], bump)]
    pub admin: Account<'info, Admin>,
    #[account(mut)]
    pub user: Signer<'info>,
    pub system_program: Program<'info, System>,
}
```

### 5.3.2 Převod vlastnictví

Převod správních práv na nového administrátora je zásadní funkcionalita zajišťující flexibilitu a dlouhodobou udržitelnost platformy. Realizace této operace, kterou zastřešuje soubor *transfer_ownership.rs*, vyžaduje pečlivé provedení několika bezpečnostních kroků

pro zajištění autorizovaného a bezpečného převodu. Tyto kroky jsou blíže specifikovány v následujících podkapitolách.

#### *5.3.2.1 Ověření aktivního stavu administrátorského účtu*

Proces začíná kontrolou aktivního stavu účtu současného administrátora. Toto ověření funguje jako první obranná linie, zajišťující, že převod práv proběhne pouze v případě, že administrátorský účet je v plně funkčním a ověřeném stavu.

Kód 14. Kontrola inicializace administrátorského účtu

```
require!(
    current_admin.is_initialized,
    CrowdFundingError::AdminNotInitialized
);
```

#### *5.3.2.2 Autorizace aktuálního administrátora*

Klíčovým momentem procesu je ověření, zda je uživatel požadující převod oprávněným administrátorem, což je prováděno porovnáním veřejných klíčů. Tato verifikace slouží jako záruka proti neautorizovaným změnám ve vedení platformy.

Kód 15. Ověření oprávnění převodu vlastnictví

```
require!(
    user.key() == current_admin.admin_pubkey,
    CrowdFundingError::UnauthorizedUser
);
```

#### *5.3.2.3 Kontrola platnosti nového administrátorského veřejného klíče*

Před provedením převodu se musí zkontrolovat, že nový administrátor disponuje platným a jedinečným veřejným klíčem. Tato opatření předcházejí možnosti nastavení administrátorské role na účty s neplatnými či již existujícími klíči, což je klíčové pro celkovou bezpečnost a integritu platformy.

Kód 16. Ověření platnosti a jedinečnosti nového administrátorského klíče

```
require!(
    new_admin != Pubkey::default() && new_admin !=
      current_admin.admin_pubkey,
    CrowdFundingError::InvalidAdminPubkey
);
```

#### *5.3.2.4 Aktualizace a výsledná inicializace*

Po ověření všech nutných podmínek se veřejný klíč nového administrátora přiřadí k administrátorskému účtu, čímž se úspěšně finalizuje proces převodu správních pravomocí. Tímto krokem se nový administrátor stává plnohodnotným správcem platformy, přičemž na sebe přebírá veškeré povinnosti spojené s jejím vedením a bezpečností.

#### *5.3.2.5 Konfigurace kontextu pro převod vlastnictví*

Konfigurace kontextu *TransferOwnerShip* klade základ pro bezpečný a autorizovaný převod správních pravomocí mezi administrátory. Struktura kontextu popsána níže, detailně specifikuje účty, které jsou zapojeny do procesu převodu, zajišťující správný běh a autorizaci této operace. Jednotlivé účty kontextu tak tvoří:

- **current_admin**: Tento účet představuje aktuálního administrátora, jehož práva mají být převedena. Proces zahrnuje ověření, že tento účet je opravdu inicializován a jeho veřejný klíč odpovídá aktuálně přihlášenému uživateli, což zajišťuje, že práva může převést pouze oprávněný administrátor.
- **admin_pubkey**: Uživatel provádějící převod vlastnictví, typicky aktuální administrátor, musí být podepisující stranou operace. Tento účet zajišťuje, že převod je iniciován a autorizován osobou s dostatečnými oprávněními.
- **system_program**: Odkaz na základní systémový program Solana je nezbytný pro technické provedení převodu, včetně aktualizace administrátorských klíčů a zajištění kompatibility operace s infrastrukturou Solana blockchainu.

Kód 17. Definice kontextové struktury pro převod vlastnictví

```
#[derive(Accounts)]
pub struct TransferOwnerShip<'info> {
    #[account(mut, has_one = admin_pubkey)]
    pub current_admin: Account<'info, Admin>,
    #[account(mut)]
    pub admin_pubkey: Signer<'info>,
}
```

### 5.3.3 Vytvoření kampaně

Funkce pro zakládání kampaní poskytuje tvůrcům možnost vytvářet nové crowdfundingové projekty přímo na platformě. Umístěná v souboru *create_campaign.rs*, tato funkce je zásadní pro začátky kampaní, umožňující jim začít shromažďovat finanční prostředky od podporovatelů. Tímto způsobem se kampaně nejen efektivně prezentují potenciálním

přispěvatelům, ale jsou také vybaveny nezbytnými parametry pro správné nastavení cílů, trvání a dalších klíčových aspektů, zajišťujících jejich úspěch a dosažení stanovených cílů. Detailní pohled na tuto funkci je popsán v následujících podkapitolách.

#### *5.3.3.1 Validace názvu a popisu*

Funkce nejprve provádí kontrolu, aby se ujistila, že název a popis kampaně vyhovují předem stanoveným limitům na délku a nejsou ponechány prázdné. Tímto přístupem se zaručuje, že každá kampaň na platformě disponuje jasným a stručným názvem a popisem, které společně poskytují efektivní a informativní představení projektu potenciálním přispěvatelům.

Kód 18. Ověření délkových limitů názvu a popisu kampaně

```
require!(
    name.len() >= MIN_NAME_LEN && name.len() <= MAX_NAME_LEN,
    CrowdFundingError::InvalidCampaignNameLength
);
require!(
    description.len() >= MIN_DESC_LEN && description.len() <=
      MAX_DESC_LEN,
    CrowdFundingError::InvalidDescriptionLength
);
```

#### *5.3.3.2 Ověření cíle a doby trvání*

Následně se prověřuje, že jak finanční cíl, tak i doba trvání kampaně odpovídají limitům nastaveným platformou. Tento postup zaručuje, že stanovené cíle jsou realistické a doba trvání kampaně je spravedlivá, čímž se podporuje vyvážené prostředí pro všechny kampaně a zvyšuje se šance na jejich úspěšné financování.

Kód 19. Ověření platnosti doby trvání a finančního cíle kampaně

```
require!(
    duration > 0 && duration <= MAX_DURATION && goal > 0 && goal <=
      MAX_GOAL,
    CrowdFundingError::InvalidDurationOrGoal
);
```

#### *5.3.3.3 Kontrola IPFS hashe obrázku*

Z důvodu podpoření decentralizace a zvýšení odolnosti informací, je vyžadován platný IPFS hash odkazující na obrázek kampaně. Tento krok přináší do procesu větší

transparentnost a posiluje důvěru v prezentované kampaně tím, že zajišťuje přístup ke zcela ověřitelným a neměnným vizuálním materiálům.

Kód 20. Kontrola platnosti IPFS hashe obrázku kampaně

```
require!(
    !image_ipfs_hash.is_empty(),
    CrowdFundingError::InvalidIpfsHash
);
```

#### *5.3.3.4 Inicializace kampaně*

Po úspěšném ověření všech parametrů se vytvoří nový účet kampaně, do kterého se přenesou všechny poskytnuté informace. Kampaně jsou nastaveny jako neaktivní až do jejich oficiálního spuštění administrátorem platformy.

#### *5.3.3.5 Konfigurace kontextu pro vytvoření kampaně*

Konfigurace kontextu klíčově předchází vytvoření nové kampaně, určující role a požadavky pro účty zúčastněné v procesu. Tato struktura zahrnuje účty kampaně, uživatele, administrátora a systémový program, zajišťujíc správné nastavení a autorizaci zakládajícího uživatele, přičemž každý účet hraje specifickou roli v procesu založení kampaně. Jednotlivé účty kontextu tak tvoří:

- **campaign**: Účet, který je inicializován jako nová kampaň, obsahující všechny klíčové informace a nastavení.
- **user**: Uživatel, který kampaň zakládá, zde vystupuje jako podepisující osoba transakce, což potvrzuje jeho roli organizátora.
- **admin**: Administrátorský účet, který musí být předem inicializován a ověřen, aby bylo možné založit kampaň.
- **system_program**: Systémový program Solana, nezbytný pro správnou alokaci prostoru a zdrojů pro nově vytvořený účet kampaně.

Kód 21. Definice kontextové struktury pro vytvoření kampaně

```
#[derive(Accounts)]
pub struct Create<'info> {
    #[account(init, payer=user, space=Campaign::max_size(), seeds=
      [CROWDFUNDING_SEED.as_bytes(), user.key().as_ref()], bump)]
    pub campaign: Account<'info, Campaign>,
    #[account(mut)]
    pub user: Signer<'info>,
    #[account(mut)]
    pub admin: Account<'info, Admin>,
    pub system_program: Program<'info, System>,
}
```

### 5.3.4 Schválení kampaně administrátorem

Funkce *review_campaign* poskytuje administrátorovi platformy možnost schvalovat kampaně, čímž jim umožňuje začít přijímat finanční příspěvky od veřejnosti. Realizována v souboru *review_campaign.rs*, tato funkce je klíčová pro proces schvalování a aktivaci kampaní, umožňuje přechod od jejich koncipování k možnosti získávání podpory. Zajištění všech nezbytných kritérií pro schválení popisují následující podkapitoly.

#### *5.3.4.1 Verifikace oprávnění administrátora*

Úvodní fáze procesu schvalování klade důraz na ověření, že akci schvalování kampaně provádí výhradně autorizovaný administrátor. Toto ověření probíhá porovnáním veřejných klíčů, aby se potvrdila shoda s klíčem zaznamenaným u administrátorského účtu. Tím se efektivně brání neautorizovaným zásahům do procesu a zajišťuje se tak integrita schvalovacího mechanismu.

Kód 22. Ověření administrátorského oprávnění

```
require!(
    user.key() == admin.admin_pubkey,
    CrowdFundingError::InvalidAdminPubkey
);
```

#### *5.3.4.2 Kontrola aktivního stavu kampaně*

Následná kontrola zajišťuje, že kampaň ještě nevstoupila do fáze aktivního přijímání příspěvků, což brání redundantnímu schvalování kampaní, které jsou již aktivní. Tím se zachovává přehlednost a správný postup při aktivaci nových kampaní.

Kód 23. Ověření neaktivního stavu kampaně

```
require!(
    !campaign.is_active,
    CrowdFundingError::CampaignAlreadyActive
);
```

#### *5.3.4.3 Kontrola možného stavu zrušení kampaně*

Tato fáze ověřuje, že daná kampaň nebyla před svým potenciálním schválením již jednou zrušena. Je důležité zajistit, že kampaně, které byly z jakéhokoliv důvodu ukončeny, nebudou následně reaktivovány, čímž se udržuje integrita procesu schvalování na platformě.

Kód 24. Ověření že kampaň nebyla zrušena

```
require!(
    !campaign.is_canceled,
    CrowdFundingError::CampaignCanceled
);
```

#### *5.3.4.4 Aktivace kampaně*

Po úspěšném průchodu kontrolními mechanismy dochází k finálnímu nastavení kampaně, kdy je její stav přepnut na aktivní a stanovena její doba trvání. Tento proces je zásadní pro oficiální zahájení kampaně a umožnění získávání finanční podpory od veřejnosti.

Kód 25. Aktivace kampaně a nastavení doby ukončení

```
campaign.is_active = true;
campaign.end_campaign = Clock::get()?.unix_timestamp
      + campaign.duration;
```

#### *5.3.4.5 Konfigurace kontextu pro schválení kampaně*

Konfigurace kontextu *Review* přesně určuje účty zapojené do procesu schvalování kampaně. Struktura kontextu zajistí, že všechny nezbytné účty jsou řádně nastaveny a mají přístupová práva potřebná k provedení schválení. Jednotlivé účty kontextu tak tvoří:

- **campaign**: Tento účet reprezentuje kampaň, která má být schválena. Je nutné, aby tento účet byl označen jako měnný pomocí *mut*, což umožňuje jeho aktualizaci během procesu schvalování.
- **admin**: Účet administrátora, který provádí recenzi kampaně. Tento účet musí být spojen s veřejným klíčem administrátora, aby byla zajištěna jeho oprávnění pro schvalování kampaně.

- **admin_pubkey**: Veřejný klíč administrátora jako podepisující entity procesu. To zajišťuje, že pouze oprávněný administrátor může kampaně schvalovat.

Kód 26. Struktura kontextu pro schválení kampaně

```
#[derive(Accounts)]
pub struct Review<'info> {
    #[account(mut)]
    pub campaign: Account<'info, Campaign>,
    #[account(mut, has_one = admin_pubkey)]
    pub admin: Account<'info, Admin>,
    #[account(mut)]
    pub admin_pubkey: Signer<'info>,
}
```

### 5.3.5 Zrušení kampaně administrátorem

V rámci správy a dohledu nad kampaněmi na crowdfundingové platformě představuje funkce *cancel_campaign* nezbytný nástroj pro administrátory. Implementovaná v souboru *cancel_campaign.rs*, tato funkce umožňuje administrátorům efektivně reagovat na situace, kdy kampaň neodpovídá pravidlům nebo cílům platformy, nebo když je z jakéhokoliv důvodu nutné kampaň zrušit. Následující podkapitoly detailně popisují klíčové kontroly a kroky prováděné během procesu zrušení kampaně.

#### *5.3.5.1 Kontrola vybraných finančních prostředků*

Počátečním krokem je zajištění, že z kampaně nebyly vybrány žádné finanční prostředky. Tato opatření chrání integritu finančních transakcí na platformě a předchází komplikacím spojeným se zrušením již financované kampaně.

Kód 27. Ověření nevybraných finančních prostředků

```
require!(
    !campaign.is_withdrawn,
    CrowdFundingError::CampaignAlreadyWithdrawed
);
```

#### *5.3.5.2 Ověření oprávnění administrátora*

Pokračujícím důležitým aspektem je ověření, že akce zrušení kampaně je prováděna oprávněným administrátorem. To zajistí, že ke zrušení dojde pouze na základě rozhodnutí kompetentních osob s příslušnými oprávněními.

Kód 28. Ověření oprávnění administrátora pro zrušení kampaně

```
require!(
    user.key() == admin.admin_pubkey,
    CrowdFundingError::InvalidAdminPubkey
);
```

#### *5.3.5.3 Kontrola již zrušené kampaně*

Ověření, že kampaň dosud nebyla zrušena, je důležitým krokem pro zachování integrity a řádného postupu v procesu schvalování a následného zrušení kampaní. Tento krok pomáhá předejít redundanci v rozhodovacích procesech a zajišťuje, že každá akce zrušení je řádně odůvodněna a v souladu s aktuálním stavem kampaně.

Kód 29. Kontrola že kampaň nebyla předtím zrušena

```
require!(
    !campaign.is_canceled,
    CrowdFundingError::CampaignAlreadyCanceled
);
```

#### *5.3.5.4 Aplikace zrušení kampaně*

Po úspěšném ověření, že kampaň splňuje podmínky pro zrušení, se její status aktualizuje na neaktivní a je formálně prohlášena za zrušenou. Tento krok zastavuje veškeré další příspěvky a signalizuje konec kampaně.

Kód 30. Změna stavu kampaně na zrušenou

```
campaign.is_active = false;
campaign.is_canceled = true;
```

#### *5.3.5.5 Konfigurace kontextu pro zrušení kampaně*

V této fázi je definována struktura *Cancel*, který určuje účty zapojené do procesu zrušení kampaně. Zahrnuje účty kampaně a administrátora, spolu s veřejným klíčem administrátora, což umožňuje správné provedení zrušení s přihlédnutím k oprávněním a rolím účastníků. Jednotlivé účty kontextu tak tvoří:

- **campaign**: Tento účet představuje konkrétní kampaň, která má být zrušena. Jako mutabilní účet umožňuje aktualizaci stavu kampaně, což je nezbytné pro změnu jejího aktivního stavu na neaktivní a označení jako zrušené.
- **admin**: Administrátorský účet, který má oprávnění k zrušení kampaně. Atribut *has_one = admin_pubkey* zajišťuje, že pouze administrátor, který kampaň vytvořil

nebo byl později přiřazen, může kampaň zrušit. To je klíčové pro udržení kontroly nad tím, kdo může provádět zásadní změny.

- **admin_pubkey**: Veřejný klíč administrátora jako podepisující osoba transakce. To zajišťuje, že akci zrušení kampaně může iniciálně provést pouze uživatel s příslušným oprávněním.

Kód 31. Struktura kontextu pro zrušení kampaně

```
#[derive(Accounts)]
pub struct Cancel<'info> {
    #[account(mut)]
    pub campaign: Account<'info, Campaign>,
    #[account(mut, has_one = admin_pubkey)]
    pub admin: Account<'info, Admin>,
    #[account(mut)]
    pub admin_pubkey: Signer<'info>,
}
```

### 5.3.6 Podpoření kampaně

Funkce *support_campaign* zajišťuje, že podpora kampaní je provedena správně a v souladu s pravidly platformy. Implementovaná v souboru *support_campaign.rs*, tato funkce detailně popisuje, jak podporovatelé mohou finančně přispět k aktivním kampaním a jak jsou tyto příspěvky spravovány. Následující podkapitoly detailně popisují klíčové kroky prováděné během podpoření kampaně.

#### *5.3.6.1 Ověření stavu kampaně pomocí modifikátorů*

Před podpořením jakékoli kampaně je klíčové ověřit, že splňuje specifické požadavky: musí být aktivní, nesmí překročit svůj stanovený termín ukončení a zároveň nesmí mít již vybrány všechny prostředky. Tyto předpoklady zajišťují, že každý finanční příspěvek je přijat v souladu s transparentními zásadami platformy, což přispívá k důvěryhodnosti a efektivitě crowdfundingových záležitostí.

Kód 32. Ověřovací kroky před podporou kampaně

```
// ověření, že kampaň je aktivní a otevřená pro podporu
require!(campaign.is_active, CrowdFundingError::CampaignNotActive);

// zajištění, že aktuální čas není po datu ukončení kampaně
require!(campaign.end_campaign >= Clock::get()?.unix_timestamp,
      CrowdFundingError::CampaignEnded);

// kontrola, že z kampaně ještě nebyly vybrány žádné finanční prostředky
require!(campaign.is_withdrawn == false, CrowdFundingError::
      WithdrawnCampaign);
```

#### *5.3.6.2 Přidávání a aktualizace finanční podpory*

Proces podpory kampaně umožňuje účastníkům platformy přispět finančními prostředky k vybraným projektům. Klíčovou součástí této funkce je efektivní správa příspěvků od podporovatelů. Pokud podporovatel již přispěl do kampaně, jeho dřívější příspěvek je aktualizován o novou poskytnutou částku. V případě, že jde o první příspěvek daného podporovatele, je jeho záznam nově vytvořen v seznamu *pledgers* dané kampaně. Tento postup zajišťuje, že každý finanční příspěvek je řádně zaznamenán a přičten k celkové shromážděné částce kampaně.

Kód 33. Aktualizace a přidání příspěvků podporovatelů

```
match pledger_index {
    Some(index) => {
        let pledger = &mut campaign.pledgers[index];
        pledger.pledged_amount = pledger
            .pledged_amount
            .checked_add(amount)
            .ok_or(ProgramError::InvalidArgument)?;
    }
    None => {
        campaign.pledgers.push(Pledgers {
            pledger_pubkey: *ctx.accounts.user.key,
            pledged_amount: amount,
        });
    }
}
```

#### *5.3.6.3 Konfigurace kontextu pro podporu kampaně*

V této fázi procesu podpory kampaně je zásadní definovat strukturu *Support*, která specifikuje účty zapojené do tohoto procesu. To zahrnuje účty samotné kampaně, podporovatele

a systémový program, nezbytný pro realizaci finančních převodů. Přesné nastavení těchto účtů umožňuje efektivní a bezpečnou podporu kampaně s přihlédnutím k příslušným oprávněním a rolím účastníků. Klíčové účty kontextu zahrnují:

- **campaign**: Tento účet reprezentuje specifickou kampaň, která je předmětem podpory. Označení tohoto účtu jako měnného pomocí *mut* umožňuje aktualizaci informací o kampani v reakci na příchozí finanční příspěvky, což je zásadní pro správné řízení aktivit.
- **user**: Účet uživatele, který přispívá na kampaň, zde slouží jako podepisující strana transakce. Toto označení potvrzuje jeho závazek k podpoře a umožňuje přímou interakci s kampaní.
- **system_program**: Základní systémový program Solana, který je klíčový pro správnou realizaci finančních převodů mezi podporovatelem a kampaní. Tento program zajišťuje, že převod prostředků proběhne podle Solana protokolů a bezpečnostních standardů.

Kód 34. Struktura kontextu pro podporu kampaně

```
#[derive(Accounts)]
pub struct Support<'info> {
    #[account(mut)]
    pub campaign: Account<'info, Campaign>,

    #[account(mut)]
    pub user: Signer<'info>,
    pub system_program: Program<'info, System>,
}
```

### 5.3.7 Zrušení podpory kampaně

Funkce *cancel_support* zahrnutá v souboru *cancel_support.rs* poskytuje uživatelům možnost odvolat svůj finanční příspěvek k podpořeným kampaním. Tento proces je důležitý pro zajištění flexibility a uživatelské samosprávy na platformě, umožňuje uživatelům změnit své rozhodnutí o podpoře a zároveň chrání integritu finančních transakcí. Následující podkapitoly detailně popisují klíčové kroky prováděné během zrušení podpory kampaně.

#### *5.3.7.1 Ověření stavu kampaně pomocí modifikátorů*

Pro zrušení podpory kampaně je důležité ověřit, že je kampaň stále aktivní a není již ukončena, zároveň nesmí mít již vybrané finanční prostředky. Tyto kontroly zajišťují, že proces zrušení podpory je v souladu s pravidly crowdfundingové platformy a podporuje tak

její transparentnost a spravedlnost. Umožňuje uživatelům reagovat na změny nebo přehodnocení své finanční podpory na kampani, zatímco zachovává integritu a spravedlivé podmínky pro všechny zúčastněné strany.

Kód 35. Ověřovací kroky před zrušením podpory kampaně

```
// ověření, že kampaň je aktivní a otevřená pro podpoření
require!(campaign.is_active, CrowdFundingError::CampaignNotActive);

// zajištění, že aktuální čas není po datumu ukončení kampaně
require!(campaign.end_campaign >= Clock::get()?.unix_timestamp,
      CrowdFundingError::CampaignEnded);

// kontrola, že z kampaně ještě nebyly vybrány žádné finanční prostředky
require!(campaign.is_withdrawn == false, CrowdFundingError::
      WithdrawnCampaign);
```

#### *5.3.7.2 Vrácení a aktualizace finančních prostředků*

Po ověření, že kampaň splňuje všechny nezbytné podmínky pro zrušení podpory, dochází k odstranění příspěvku daného podporovatele ze seznamu *pledgers* dané kampaně. Následně je příslušná finanční částka převedena zpět na účet podporovatele, čímž se upravuje celková suma shromážděná kampaní. Tento proces nejen zajišťuje přesné a transparentní sledování finančních transakcí spojených s kampaní, ale také umožňuje uživatelům flexibilně reagovat na proměny v rámci kampaně nebo přehodnotit své rozhodnutí v podpoření kampaně.

Kód 36. Aktualizace a vrácení příspěvků podporovatelů

```
if let Some(index) = pledger_index {
    let pledger = campaign.pledgers.remove(index);
    let withdrawable_amount =
      **campaign.to_account_info().lamports.borrow() - rent;
    require!(
        withdrawable_amount >= pledger.pledged_amount,
        CrowdFundingError::InsufficientFundsForRent
    );
    **campaign.to_account_info().try_borrow_mut_lamports()? -=
            pledger.pledged_amount;
    **user.to_account_info().try_borrow_mut_lamports()? +=
            pledger.pledged_amount;

    campaign.pledged = campaign
        .pledged
        .checked_sub(pledger.pledged_amount)
        .ok_or(ProgramError::InvalidArgument)?;
}
```

#### *5.3.7.3 Konfigurace kontextu pro zrušení podpory kampaně*

Struktura *CancelSupport* určuje účty zapojené do procesu zrušení podpory. Zahrnuje účet kampaně, který je předmětem odvolání podpory, a účet uživatele, který svůj příspěvek odvolává. Tato konfigurace zajišťuje, že finanční transakce proběhnou správně a jsou v souladu s pravidly a požadavky platformy. Klíčové účty kontextu zahrnují:

- **campaign**: Tento účet reprezentuje konkrétní kampaň, jejíž podpora je uživatelem zrušena. Jako mutabilní účet umožňuje úpravu stavových proměnných kampaně, včetně aktualizace seznamu podporovatelů a celkové shromážděné částky.
- **user**: Účet uživatele, který se rozhodl zrušit svou podporu, musí být autentizovaný skrze svůj privátní klíč, čímž se zabezpečuje, že akce pochází opravdu od něj. Toto opatření chrání před neoprávněnými pokusy o manipulaci s příspěvky.

Kód 37. Struktura kontextu pro zrušení podpory kampaně

```
#[derive(Accounts)]
pub struct CancelSupport<'info> {
    #[account(mut)]
    pub campaign: Account<'info, Campaign>,

    #[account(mut)]
    pub user: Signer<'info>,
}
```

### 5.3.8 Vybrání prostředků z kampaně

Funkce *withdraw_campaign* poskytuje organizátorům kampaní možnost vybrat shromážděné finanční prostředky po dosažení nebo překročení stanoveného finančního cíle. Realizována v souboru *withdraw_campaign.rs*, tato funkce je zásadní pro úspěšné dokončení kampaně a umožňuje efektivní redistribuci finančních prostředků.

#### *5.3.8.1 Ověření před vybráním pomocí modifikátorů*

Před vybráním finančních prostředků z kampaně je nezbytné provést několik klíčových ověření. Na počátku se zkontroluje, že prostředky nebyly již dříve vybrány, aby se zabránilo jejich opakovanému výběru. Dále je ověřeno vlastnictví kampaně s cílem zajistit, že žádost o výběr podává skutečný vlastník kampaně. Nakonec je potřeba zkontrolovat, zda byl dosažen finanční cíl kampaně, přesněji alespoň 80 % z celkové cílové částky, což značí, že kampaň byla alespoň částečně úspěšná. Tento proces zajišťuje spravedlivé a transparentní řízení finančních prostředků shromážděných pro kampaně a podporuje důvěru mezi organizátory kampaní a podporovateli.

Kód 38. Ověření podmínek pro výběr finančních prostředků z kampaně

```
// určení hodnoty prahu pro úspěšnost kampaně
let threshold = (campaign.goal as u128 * 80) / 100;

// zkontrolujte, že prostředky nebyly již vybrány
require!(
    campaign.is_withdrawn == false,
    CrowdFundingError::CampaignAlreadyWithdrawed
);

// ověření, že akci provádí vlastník kampaně
require!(
    campaign.owner == user.key(),
    CrowdFundingError::NotCampaignOwner
);

// zajištění, že bylo dosaženo alespoň 80% cílové částky
require!(
    campaign.pledged as u128 >= threshold,
    CrowdFundingError::WithdrawalNotAllowed
);
```

#### *5.3.8.2 Proces vybírání prostředků z kampaně*

Po ověření všech potřebných kritérií, včetně dosažení stanoveného prahu úspěšnosti a potvrzení, že žádost o výběr podává skutečný vlastník, dochází k finálnímu převodu shromážděných finančních prostředků z účtu kampaně na účet vlastníka. Zároveň se stav kampaně mění na „vybráno“, což signalizuje konec možnosti dalšího financování. Tento proces zajišťuje, že zdroje jsou převedeny správně a transparentně.

Kód 39. Výběr finančních prostředků z kampaně

```
// výpočet částky k výběru, odečtení pronájmu
let withdraw_amount = **campaign.to_account_info().lamports.borrow()
      - rent;

// odečtení finančních prostředků z účtu kampaně
**campaign.to_account_info().try_borrow_mut_lamports()?
      -= withdraw_amount;

// přidání finančních prostředků na účet vlastníka
**user.to_account_info().try_borrow_mut_lamports()? += withdraw_amount;

// označení kampaně jako vybrané a resetace ukončení kampaně
campaign.is_withdrawn = true;
campaign.end_campaign = 0;
```

#### *5.3.8.3 Konfigurace kontextu pro výběr finančních prostředků z kampaně*

Struktura kontextu *Withdraw* je klíčová pro správný průběh výběru finančních prostředků z kampaně. Specifikuje účty nezbytné pro transakci, zahrnující účet kampaně a vlastníka. Tato konfigurace zajišťuje, že pouze oprávnění uživatelé mohou provádět výběr, což přispívá k transparentnosti a spravedlnosti na crowdfundingové platformě. Klíčové účty kontextu zahrnují:

- **campaign**: Účet představující specifickou kampaň, z níž se mají prostředky vybrat. Atribut *mut* umožňuje aktualizaci informací o kampani během procesu výběru, včetně změny stavu na "vybráno" a úpravy shromážděné částky.
- **owner**: Reprezentuje vlastníka kampaně, který je zodpovědný za vyvolání procesu výběru prostředků. Jako podepisující osoba transakce tato role zaručuje, že výběr může provést pouze uživatel s oprávněním spravovat daný účet kampaně, což přispívá k integritě celého procesu.

Kód 40. Struktura kontextu pro výběr finančních prostředků z kampaně

```
#[derive(Accounts)]
pub struct Withdraw<'info> {
    #[account(mut, has_one = owner)]
    pub campaign: Account<'info, Campaign>,

    #[account(mut)]
    pub owner: Signer<'info>,
}
```

## 5.4 Nasazení programu do blockchainové sítě

Pro nasazení programu do blockchainové sítě, byl zvolen cluster *devnet*, který je vhodný pro vývojové a testovací účely a poskytuje ideální prostředí, kde je možné bezpečně experimentovat a ověřovat funkčnost vytvářeného kódu bez potřeby použití skutečných finančních prostředků. Projekt je nutné nasadit skrze Anchor framework, který byl blíže popsán v předchozích kapitolách.

Pomocí následující příkazu je možné vytvořit sestavení programu, což je zejména důležité, aby se zkompilovalo všechno, co je součásti projektu, včetně programů. Všechny tyto operace jsou provedeny tímto příkazem [55]:

```
$ anchor build
```

Následně je nutné provést příkaz, pomocí kterého framework Anchor nasadí program do sítě a bude tedy veřejný. To je provedeno pomocí [55]:

```
$ anchor deploy
```

Po nasazení programu je vytvořen soubor IDL soubor, který je pojmenován po inicializaci projektu. Tento soubor následně slouží tvorbě tvorbě front-endu, protože obsahuje veškeré potřebné informace o rozhraních a funkcích programu.

## 5.5 Front-endová implementace crowdfundingové aplikace

Front-endová část decentralizované crowdfundingové aplikace tvoří především klíčové uživatelské rozhraní, kde daní uživatelé platformy interagují přímo se Solana blockchainem. Hlavním cílem je zajisti intuitivní a efektivní design s interakcemi s aplikací, která umožňuje uživatelům snadno přispívat k vybraným kampaním, sledovat vývoj, spravovat kampaň a provádět operace na úrovni administrátorského oprávnění.

### 5.5.1 Šablona Solana dApp Scaffold Next

Pro zjednodušení vývoje a integrace se Solana blockchainem byl v projektu použit Solana Scaffold [71], což je šablona poskytována společností Solana Labs [72]. Tato šablona je navržena především pro okamžitý vývoj či testování na platformě, protože v základu obsahuje všechny potřebné knihovny pro okamžitou interakci s blockchainem. Pro instalaci a použití této šablony je nutné vykonat několik kroků. Nejprve je nutné naklonovat repozitář pomocí tohoto příkazu:

```
$ git clone https://github.com/solana-labs/dapp-scaffold.git
```

Po naklonování je možné přejít do adresáře projektu a následně nainstalovat všechny potřebné dependence, které jsou definovány v souboru *package.json*. Tato operace je provedena pomocí příkazu:

```
$ npm install
```

Po instalaci všech potřebných dependencí lze webový projekt spustit pomocí příkazu:

```
$ npm run dev
```

### 5.5.2 Web3 API

V rámci vytvářeného projektu s použitím Solana Scaffold je zcela klíčové pochopení a rovněž implementace Web3 API, které umožňuje komunikace front-endové části s tou backendovou. Web3 API je tedy sada knihoven, které poskytují funkcionalitu potřebnou k provádění transakcí, správě účtu a komunikaci se Solana programy [41].

Mezi základní funkce, které toto Web3 API poskytuje patří především získávání informací o aktuálním stavu blockchainu, dále se jedná o propojení peněženek s decentralizovanou aplikací, a především toto API poskytuje celkovou interakci s programy [41].

#### *5.5.2.1 Konfigurace Web3 API*

Základem je konfigurace sítě pomocí hooku *useNetworkConfiguration*, který poskytuje především informace o aktuální síti, což může být devnet, mainnet, nebo localnet. Následně je použit *useMemo* k dynamickému vytvoření endpointu pro připojení aplikace k Solana clusteru, což je závislé především na zvolené síti. Následně pomocí *clusterApiUrl* je transformován typ sítě na příslušnou URL [41].

Kód 41. Konfigurace endopointu pro Solana blockchain

```
const { networkConfiguration } = useNetworkConfiguration();
const network = networkConfiguration as WalletAdapterNetwork;
const endpoint = useMemo(() => clusterApiUrl(network), [network]);
```

Dále je potřeba zapouzdřit komponenty, které zajišťují připojení k Solana blockchainu. Uvnitř *WalletProvider*, který spravuje peněženky uživatelů, zpracovává chyby či podporuje automatické připojení. Následně je zde definovaná konstanta *ReactUIWalletModalProviderDynamic* dynamicky načítá a zobrazuje modální okno pro správu peněženek, což umožňuje uživatelům interagovat pomocí svých peněženek s decentralizovanou aplikací [41].

Kód 42. Konfigurace endopointu pro Solana blockchain

```
<ConnectionProvider endpoint={endpoint}>
    <WalletProvider wallets={wallets} onError={onError} autoConnect={au-
toConnect}>
        <ReactUIWalletModalProviderDynamic>
            {children}
        </ReactUIWalletModalProviderDynamic>
    </WalletProvider>
</ConnectionProvider>
```

## 5.6 Aplikační vrstva

Aplikační vrstva představuje klíčový prvek vytvářené decentralizované crowdfundingové aplikace, který je zodpovědný za zpracování všech uživatelských interakcí a komunikacemi s blockchainem Solana. Aplikační vrstva zahrnuje rovněž veškerou logiku pro načítání a následné zobrazení a správu crowdfundingových kampaní.

### 5.6.1 Aplikační vrstva dashboard-feature.tsx

V souboru *dashboard-feature.tsx* je implementované aplikační rozhraní pro správu kampaní vytvářené crowdfundingové aplikaci na Solana blockchainu. Hlavní jsou především funkce umožňující načítání a zobrazení včetně manipulace s daty kampaní, což zahrnuje dynamické zobrazení jednotlivých kampaní, jejich následné filtrování a správu UI.

#### *5.6.1.1 Funkce getAllCampaigns*

Tato asynchronní funkce pomáhá k načítání všech kampaní prostřednictvím Solana blockchainu, následně kampaně vyfiltruje a tím je zjištěno, zda-li jsou jednotlivé kampaně stále aktivní (probíhající), nebo již byly ukončeny. Kampaně jsou dále tříděny, následně

seřazeny podle data ukončení a poté uloženy do stavu *campaigns*. Funkce je řízena za pomocí indikačního stavu *isLoading*.

Kód 43. Aplikační funkce getAllCampaigns

```
const getAllCampaigns = useCallback(async () => {
    if (program && program.account && program.account.campaign) {
      try {
        const fetchedCampaigns = await program.account.campaign.all();
        const currentTime = new Date().getTime();
        const validAndFilteredCampaigns = fetchedCampaigns.filter(
          (campaign) =>
            campaign.account !== null &&
                  // …
        );

        validAndFilteredCampaigns.sort((a, b) => {
      // …
        });
        setCampaigns(validAndFilteredCampaigns);
        setIsLoading(false);
      } catch (error) {
        console.error("Error fetching campaigns:", error);
        setIsLoading(true);
      }
      // …
  }, [program, showEndedCampaigns]);
```

#### *5.6.1.2 Funkce toggleCampaignsView*

Tato funkce slouží k přepínání zobrazení mezi aktivními a již ukončenými kampaněmi. Manipuluje s boolean stavem *showEndedCampaigns*, který následně určuje, které kampaně mají být zobrazeny.

Kód 44. Aplikační funkce toggleCampaignsView

```
  const toggleCampaignsView = () => {
    setShowEndedCampaigns(!showEndedCampaigns);
  };
```

#### *5.6.1.3 Funkce Countdown*

Komponenta Countdown je určena pro zobrazení zbývajícího času do ukončení kampaně. Využívá *useEffect*, který je určen pro pravidelné aktualizace času, ten se vypočítává v intervalu jedné sekundy pomocí funkce *calculateTimeRemaining*.

Kód 45. Aplikační funkce calculateTimeRemaining

```
const calculateTimeRemaining = (endTimestamp) => {
    const now = new Date().getTime();
    const distance = endTimestamp - now;
    const days = Math.floor(distance / (1000 * 60 * 60 * 24));
    const hours = Math.floor(
      (distance % (1000 * 60 * 60 * 24)) / (1000 * 60 * 60));
    const minutes = Math.floor((distance % (1000 * 60 * 60)) /
      (1000 * 60));
    const seconds = Math.floor((distance % (1000 * 60)) / 1000);
    return {
      days,
      hours,
      minutes,
      seconds,
      distance,};
};
```

#### *5.6.1.4 Funkce ProgressBar*

Tato komponenta zobrazuje vizuální reprezentaci pokroku kampaně vůči cílové částce. Vypočítává aktuální procentuální úspěšnost shromážděných prostředků ve vztahu k cíli a zobrazuje ji v ukazateli průběhu.

Kód 46. Aplikační funkce ProgressBar

```
const ProgressBar: FC<{ goal: number; pledged: number }> = ({
    goal,
    pledged,
  }) => {
    const progressPercent = Math.min(100, (pledged / goal) * 100);

    return (
      // …
    );
  };
```

#### *5.6.1.5 Funkce CampaignCard*

Komponenta *CampaignCard* slouží k zobrazení detailu jednotlivých kampaní včetně jejich názvu, popisu, IPFS obrázku, délky kampaně, včetně finanční částky, kterou se kampaň snaží vybrat. Tato komponenta kromě detailních informací poskytuje logiku pro načítání obrázků skrze IPFS a zobrazení aktuálního stavu financování.

Kód 47. Aplikační funkce CampaignCard

```
const CampaignCard: FC<{ campaign: ProgramAccount }> = ({ campaign }) =>
{
    const campaignId = campaign.publicKey.toBase58();
    const endTime = new Date(
      campaign.account.endCampaign.toNumber() * 1000
    const [imageLoading, setImageLoading] = useState(true);
    const [showImage, setShowImage] = useState(false);
    const goalInSol = lamportsToSol(Number(campaign.account.goal)).toFi-
xed(2);
    const progressPercent = (
      (parseFloat(pledgedInSol) / parseFloat(goalInSol)) *
      100
    ).toFixed(1);
    const ipfsProviders = [
      "https://dweb.link/ipfs/",
      "https://gateway.pinata.cloud/ipfs/",
    ];
    const explorerBaseUrl = "https://explorer.solana.com";
    const networkParam = "?cluster=devnet";
    const ownerPubkey = campaign.account.owner.toString();
    const shortenedOwnerPubkey = shortenAddress(ownerPubkey);
}
```

#### *5.6.1.6 Funkce ShowCampaigns*

Komponenta *ShowCampaigns* je na nejvyšší úrovni a zaručuje inicializaci programu Solana pomocí knihovny Anchor. Následně využívá hooks *useWallet* a *useConnection* pro přístup prohlížečové peněženky a následné připojení k Solana blockchainu.

Kód 48. Aplikační funkce ShowCampaigns

```
export const ShowCampaigns: FC = () => {
  const ourWallet = useWallet();
  const { connection } = useConnection();
  const [program, setProgram] = useState<Program | null>(null);

  const initializeProgram = useCallback(async () => {
    const getProvider = async (): Promise<AnchorProvider> => {
      const provider = new AnchorProvider(
        connection,
        ourWallet,
        AnchorProvider.defaultOptions()
      );
      return provider;
    };
}
```

### 5.6.2 Aplikační vrstva create-feature.tsx

V souboru *create-feature.tsx* je implementována aplikační vrstva pro vytváření nových kampaní organizátory. Tento soubor obsahuje komplexní funkcionalitu pro zpracování dat, které organizátor kampaně vyplňuje, jedná se tedy o název, popisek, délka kampaně, požadovaná částka pro darování a posledním údajem je IPFS [62] obrázek, který slouží jako hlavní obrázek v kampani. Dále tento soubor jednotlivé zpracovaná data validuje a skrze transakce umožňuje jejich zapsání do blockchainové sítě.

#### *5.6.2.1 Funkce createCampaign*

Jedná se o asynchronní funkci, která po validaci vstupů od organizátora kampaně vytváří zápis nové kampaně na Solana blockchainu. Před samotným zapsáním se kontrolují všechny vstupní data, konkrétně délky názvu a popisu kampaně a zda cílová částka včetně doby trvání kampaně splňuje nastavené limity. Dále je kontrolována délka IPFS hashe, který slouží jako odkaz na obrázek kampaně. Po úspěšném vytvoření kampaně se formulář resetuje a informuje uživatele prostřednictvím toast notifikace.

Kód 49. Aplikační funkce createCampaign

```
const createCampaign = async (ipfsCid) => {
    try {
      if (name.length < MIN_NAME_LEN || name.length > MAX_NAME_LEN) {
        toast.error(
          `Name must be between ${MIN_NAME_LEN} and ${MAX_NAME_LEN}
            characters.`
        );
        return;
      }
      if (
        description.length < MIN_DESC_LEN ||
        description.length > MAX_DESC_LEN
      // …
      )
      // …
      clearForm();
      toast.success("Campaign successfully created!");
    } catch (error) {
      toast.error("Failed to create campaign.");}
  };
```

#### *5.6.2.2 Funkce uploadImageToIPFS*

Tato komponenta se stará o nahrávání obrázků organizátorem kampaně skrze službu IPFS, což je velmi důležitá služba pro decentralizované uchovávání a nahrávání souborů kampaně. Funkce rovněž provádí kontrolu typu nahrávaného souboru, jeho rozlišení a na základě toho dokáže doporučit uživateli nahrát jiný obrázek, či ten současný upravit, tak aby splňoval podmínky pro nahrání skrze platformu decentralizované crowdfundingové aplikace. Po vykonání informuje uživatele o úspěšném nahrání do sítě IPFS.

Kód 50. Aplikační funkce uploadImageToIPFS

```
const uploadImageToIPFS = async (fileToUpload) => {
  setUploading(true);
  try {
    const formData = new FormData();
    formData.append("file", fileToUpload, fileToUpload.name);
    const res = await fetch("/api/files", {
      method: "POST",
      body: formData,
    });
    const ipfsHash = await res.text();
    console.log(ipfsHash);
    return ipfsHash;
  } catch (e) {
    console.error(e);
    toast.error("Trouble uploading file to IPFS.");
    return null;
  } finally {
    setUploading(false);
  }
};
```

#### *5.6.2.3 Pinata SDK*

Pinata SDK je nástroj který se používá především ve spojení služby která umožňuje snadné interagování s IPFS skrze SDK. Pro použití je nutné nainstalování daného SDK jako *npm* balíček, a to pomocí příkazu:

```
$ npm i @pinata/sdk
```

Následně je možné vytvořit počáteční inicializaci pinataSDK spolu s autentizačním klíčem, tedy JWT tokenem, což umožňuje bezpečný přístup k funkcím Pinata pro manipulaci s IPFS. Dalším klíčovým bodem je provádění již zmíněné validace souborů a je tedy možné nahrát pouze soubory ve formátech JPG, JPEG, PNG, WEBP.

Kód 51. IPFS handler pro nahrávání obrázků

```
import formidable from "formidable";
import fs from "fs";
import { fileTypeFromBuffer } from "file-type";

const pinataSDK = require("@pinata/sdk");
const pinata = new pinataSDK({ pinataJWTKey: process.env.PINATA_JWT });

interface ParsedFormData {
  fields: formidable.Fields;
  files: formidable.Files;
}
```

#### *5.6.2.4 Funkce handleFileButtonClick a handleChange*

Tyto funkce spravují přímou interakci uživatele s nahráváním daného souboru. Funkce *handleFileButtonClick* aktivuje daný výběr souboru načež, funkce *handleChange* zpracovává vybraný soubor a validuje jeho typ a rozměry vloženého obrázku.

Kód 52. Aplikační funkce handleChange

```
  const handleChange = async (e) => {
    e.preventDefault();
    const file = e.target.files[0];
    if (file) {
      const validTypes = ["image/jpeg", "image/png", "image/webp",
      "image/jpg"];
      if (!validTypes.includes(file.type)) {
        toast.error(
          "Invalid file type. Only JPG, JPEG, PNG, and WEBP images are
             allowed."
        );
        return;
      }
```

### 5.6.3 Aplikační vrstva admin-feature.tsx

Soubor *admin-feature.tsx* je část aplikační vrstvy, který poskytuje administrátorské funkce (oprávnění) pro správu kampaní v decentralizované crowdfundingové aplikaci. Lze rovněž říci, že tento soubor obsahuje klíčové funkce pro celkovou správu, přehled a aktualizaci stavu kampaní, což z pohledu administrátorských operací zahrnuje, schvalování, či zamítání jednotlivých kampaní. Další důležitou funkcionalitou, která je zde poskytnuta je převedení administrátorských práv na jiného uživatele (adresu peněženky).

#### *5.6.3.1 Funkce getAllCampaigns*

Tato asynchronní funkce pomáhá k načítání všech kampaní prostřednictvím Solana blockchainu, následně kampaně vyfiltruje a tím je zjištěno, zda-li jsou jednotlivé kampaně stále aktivní (probíhající), nebo již byly ukončeny. Kampaně jsou dále tříděny, následně seřazeny podle data ukončení a poté uloženy do stavu campaigns. Proces načítání je moderován pomocí boolean stavu isLoading.

Kód 53. Aplikační funkce getAllCampaigns

```
const getAllCampaigns = useCallback(async () => {
  if (program && program.account && program.account.campaign) {
    try {
      const fetchedCampaigns = await program.account.campaign.all();
      const validCampaigns = fetchedCampaigns.filter(
        (c) => c.account !== null
      );
      validCampaigns.sort(
        (a, b) =>
          a.account.duration.toNumber() - b.account.duration.toNumber()
      );
      setCampaigns(validCampaigns);
      setIsLoading(false);
    } catch (error) {
      console.error("Error fetching campaigns:", error);
      setIsLoading(true);
    }
  } else {
    console.log("Program not initialized");
  }
}, [program]);
```

#### *5.6.3.2 Funkce reviewCampaign a cancelCampaign*

Tyto komponenty umožňují administrátorovi platformy schvalovat či zamítat konkrétní kampaně. Díky tomu, že veškeré operace jsou zaznamenávány do blockchainové sítě Solana, jsou obě operace naprosto transparentní a veřejně dohledatelné. Po potvrzení transakce administrátorem, je kampaň dynamicky přesunuta to podsekce, schválení či zamítnutí. V případě schválení je kampaň dostupná k financování a je zobrazována na hlavní stránce platformy.

Kód 54. Aplikační funkce reviewCampaign a cancelCampaign

```
const reviewCampaign = async (publicKey) => {
  try { //..
    const [admin] = await PublicKey.findProgramAddressSync(
      [utils.bytes.utf8.encode("admin_account")],
      program.programId );
    // …
    await program.methods.campaignReview().accounts({ // …
      }) // … .rpc();
    );
    await getAllCampaigns();
  } catch (error) {
    toast.error("Error while reviewing " + publicKey);
  } // … };
const cancelCampaign = async (publicKey) => {
  try {
    // …
    await program.methods.campaignCancel().accounts({campaign:
    publicKey, admin, user: anchProvider.wallet.publicKey,
      }).rpc();
    // …);  };
```

#### *5.6.3.3 Funkce initAdmin a transferOwnership*

Komponenta *initAdmin* pro inicializaci administrátora je prvotní funkce, která musí být na platformě, vykonána. Bez této inicializace nelze na platformě pokračovat, všechny jiné funkce jsou zablokovány do doby, než proběhne nastavení práv prvotnímu administrátorovi. Současný administrátor může kdykoliv své práva převést na jiného uživatele pomocí funkce *transferOwnership*. Toto převedení je nevratné, a tudíž po zavolání této funkce současný administrátor ztratí aktuální oprávnění.

Kód 55. Aplikační funkce initAdmin a transferOwnership

```
const initAdmin = async () => {
  try { // …
    const signerPubkey = anchProvider.wallet.publicKey;
    const [admin] = await PublicKey.findProgramAddressSync(
      [utils.bytes.utf8.encode("admin_account")], program.programId);
    await program.methods
      .adminInitialize().accounts({
        admin, user: signerPubkey,
        systemProgram: web3.SystemProgram.programId,
      }).rpc();
    toast.success("Admin initialized successfully!");
    await getAdminPubkey();
  } // …
};
const transferOwnership = async () => {
  try { // …
    const newAdminPubkey = new PublicKey(pubkeyNewAdminInput);
    const [currentAdmin] = await PublicKey.findProgramAddressSync(
      [utils.bytes.utf8.encode("admin_account")],
      program.programId
    );
    await program.methods
      .ownershipTransfer(newAdminPubkey).accounts({currentAdmin,
        user: signerPubkey,}).rpc();
    toast.success("Ownership has been transfered successfully!");
    // …
};
```

#### *5.6.3.4 Funkce CampaignCard*

Komponenta CampaignCard slouží k zobrazení detailu jednotlivých kampaní včetně jejich názvu, popisu, IPFS obrázku, délky kampaně, včetně finanční částky, kterou se kampaň snaží vybrat. Tato komponenta kromě detailních informací poskytuje logiku pro načítání obrázků skrze IPFS a zobrazení aktuálního stavu financování.

Kód 56. Aplikační funkce CampaignCard

```
const CampaignCard: FC<{ campaign: ProgramAccount }> = ({ campaign }) =>
{
  const [showImage, setShowImage] = useState(false);
  const ipfsProviders = [
    "https://dweb.link/ipfs/",
    "https://gateway.pinata.cloud/ipfs/",
  ];
  const goalInSol = lamportsToSol(campaign.account.goal).toFixed(2);
  const explorerBaseUrl = "https://explorer.solana.com";
  const [currentGatewayIndex, setCurrentGatewayIndex] = useState(0);
  const durationInDays = Math.floor(
    campaign.account.duration.toNumber() / (60 * 60 * 24)
  );
  const computeImageUrl = () => {
    const baseUri = ipfsProviders[currentGatewayIndex %
      ipfsProviders.length];
    return `${baseUri}${campaign.account.imageIpfsHash}`;
  }; // …
  }, [campaign.account.imageIpfsHash]);
```

### 5.6.4 Aplikační vrstva portfolio-feature.tsx

V souboru *portfolio-feature.tsx* je aplikována aplikační vrstva určená pro správu a zobrazení detailů kampaně určené pro organizátory decentralizované crowdfundingové aplikace. Tento soubor rovněž obsahuje funkcionalitu pro zobrazení jednotlivých detailů o kampani. Klíčovou funkcionalitou je správa včetně možnosti vybrání finančních prostředků kampaně, a v závěru pro transparentnost jsou veškeré finanční příspěvky viditelné za pomocí komponenty, která zobrazuje veškeré podporovatele dané kampaně.

#### *5.6.4.1 Funkce getCampaign*

Tato asynchronní funkcionalita slouží pro načítání kampaně, která je přiřazena a vlastněna k aktuálně přihlášenému uživateli, připojeného na základě své peněženky. Tato funkce tedy vyhledává kampaň zapsanou v blockchainové síti za pomoci programu a klíče uživatele vycházejícího z jeho Phantom peněženky. Pokud uživatel nemá vytvořenou žádnou kampaň tak aplikace záložku Portfolio nenabízí a tím dynamicky reaguje na případné změny.

Kód 57. Aplikační funkce getCampaign

```
const getCampaign = useCallback(async () => {
  setIsLoading(true);
  try {
    const campaigns = await program.account.campaign.all();
    const userOwnedCampaign = campaigns.find(
      ({ account }) =>
        account.owner.toBase58() === ourWallet.publicKey?.toBase58());
    if (userOwnedCampaign) {
      setCampaign(userOwnedCampaign.account);
      setCampaignPublicKey(userOwnedCampaign.publicKey);
      setIsWithdrawn(userOwnedCampaign.account.isWithdrawn);
    } else {
      setCampaign(null);}
      // …
  }
}, [program, ourWallet.publicKey]);
```

#### *5.6.4.2 Funkce withdrawCampaign*

Tato funkce umožňuje vlastníkovi kampaně vybrat shromážděné finanční prostředky po dosažení finančního cíle. Funkce obsahuje logiku zahrnující volání programu, který realizuje celý proces převodu prostředků. Organizátor je informován o průběhu a následném výsledku transakce prostřednictvím jednotlivých uživatelských notifikací.

Kód 58. Aplikační funkce withdrawCampaign

```
  const withdrawCampaign = async (publicKey) => {
    setIsWithdrawing(true);
    try { // …
      const program = new Program(idl_object, programID, anchProvider);
      await program.methods
        .campaignWithdraw().accounts({campaign: publicKey,
            user: anchProvider.wallet.publicKey,
        }).rpc();
      toast.success("Campaign has been successfully withdrawn: "
      + publicKey);
    }  // …
    setIsWithdrawing(false);
  };
```

#### *5.6.4.3 Funkce toggleDonorsVisibility*

Tato funkce umožňuje zobrazení seznamu jednotlivých podporovatelů dané kampaně, což přispívá k transparentnosti vytvářené crowdfundingové platformy. U každého

podporovatele je zobrazena jeho veřejná adresa a rovněž i celková částka kterou danou kampaň podpořil.

Kód 59. Aplikační funkce toggleDonorsVisibility

```
const toggleDonorsVisibility = () => {
  setShowDonors(!showDonors);
};
```

### 5.6.5 Aplikační vrstva [campaignId].tsx

Soubor *[campaignId].tsx* slouží jako aplikační vrstva pro zobrazení detailů svázaných s crowdfundingovou kampaní. Tento soubor rovněž obsahuje funkce pro zobrazení a interakci konkrétní kampaně, což zahrnuje funkce pro podpoření kampaně, případně funkce pro zrušení podpory kampaně.

#### *5.6.5.1 Funkce supportCampaign*

Umožňuje uživatelům finančně podpořit vybranou kampaň, uživatel pouze zvolí podporovanou částku a následně ji skrze peněženku potvrdí což vede k převedení prostředků a zápisu dat do blockchainové sítě. Zda-li transakce proběhla, či neproběhla správně je uživatel o konečném stavu notifikován.

Kód 60. Aplikační funkce supportCampaign

```
const supportCampaign = async (amount) => {
  const amountLamports = solToLamports(amount);
  if (
    isNaN(amount) ||
    amount < MIN_SUPPORT_AMOUNT_SOL ||
    amount > MAX_SUPPORT_AMOUNT_SOL
  ) // …
  setIsSupporting(true);
  try {
    await program.methods
      .campaignSupport(new BN(amountLamports))
      .accounts({ campaign: campaignPublicKey,
        user: anchProvider.wallet.publicKey,
        systemProgram: web3.SystemProgram.programId,}).rpc();
    toast.success(
      `Campaign "${campaign.name}" has been successfully supported!`
    ); // … }
  setIsSupporting(false);
};
```

#### *5.6.5.2 Funkce cancelSupport*

Funkce umožňuje uživatelům zrušit již poskytnutou podporu, tedy za předpokladu, že jsou splněny podmínky pro zrušení této podpory. Po potvrzení zrušení v peněžence je uživatel informován prostřednictvím notifikací o provedených operací.

Kód 61. Aplikační funkce cancelSupport

```
const cancelSupport = async (publicKey) => {
  try {
    const anchProvider = await getProvider();
    const program = new Program(idl_object, programID, anchProvider);

    await program.methods.supportCancel()
      .accounts({campaign: publicKey,
        user: anchProvider.wallet.publicKey,}).rpc();
    // …
};
```

#### *5.6.5.3 Funkce toggleDonorsVisibility*

Tato funkce umožňuje zobrazení seznamu jednotlivých podporovatelů dané kampaně, což přispívá k transparentnosti vytvářené crowdfundingové platformy. U každého podporovatele je zobrazena jeho veřejná adresa a rovněž i celková částka kterou danou kampaň podpořil.

Kód 62. Aplikační funkce toggleDonorsVisibility

```
const toggleDonorsVisibility = () => {
  setShowDonors(!showDonors);
};
```

## 5.7 Prezentační vrstva

Pomocí prezentační vrstvy je možné vytvořit celkový a rovněž jednotný design aplikace, včetně jednotlivých interaktivních prvků, jako jsou tlačítka, které jsou propojeny s různými funkcemi, seznam aktivních kampaní, formulář pro vytvoření kampaně, výběrové okno pro nahrání obrázku skrze IPFS [62] a další prvky, které zjednodušují používání aplikace uživateli. Protože je aplikace včetně prezentační vrstvy vytvořena skrze Next.js (*kapitola 4.8 Next.js*), lze pro prezentační vrstvu využívat TSX, který umožňuje rozšíření syntaxe a dovoluje používat značkovací jazyk do TypeScriptu [73].

### 5.7.1 Prezentační vrstva dashboard-feature.tsx

Prezentační vrstva souboru *dashboard-feature.tsx* je vytvořena s využitím modulárního přístupu Next.js. Jednotlivé kampaně jsou tedy dynamicky zobrazovány pomocí funkce mapování a syntaxe map(), kde každá z kampaní je reprezentována kartou, obsahující základní informace o kampani, jako je název, popisek, doba trvání kampaně a finanční cíl kampaně.

Kód níže vytváří seznam kampaní, který je vykreslován v uživatelském rozhraní a to pomocí komponenty *CampaignCard*.

Kód 63. Vykreslení kampaní pomocí komponenty CampaignCard

```
<div className="flex flex-wrap justify-start">
  {campaigns.map(campaign => (
    <CampaignCard key={campaign.publicKey.toBase58()}
            campaign={campaign} />
  ))}
</div>
```

Komponenta CampaignCard vykresluje pro každou kampaň kartu s podrobnými informacemi a interaktivními prvky, jako je například detail kampaně, či vizuální animace progresu financování kampaně.

Pro grafické zobrazení průběhu financování byla vytvořena komponenta ProgressBar, která vytváří vizualizaci procentuální úspěšnosti financování dané kampaně. Kód komponenty níže vykresluje pruh v miniatuře kampaně, který barevně značí úspěšnost financování.

Kód 64. Komponenta ProgressBar určená k vizualizaci průběhu financování.

```
<ProgressBar goal={Number(campaign.goal)}
      pledged={Number(campaign.pledged)} />
```

### 5.7.2 Prezentační vrstva create-feature.tsx

Prezentační vrstva v souboru *create-feature.tsx* umožňuje organizátorům vytvořit novou kampaň, která po schválení administrátorem by byla zobrazována na platformě blockchainu Solana. Tento modul zároveň poskytuje formulář, kde organizátoři zadávají informace o vytvářené kampani, tedy název, popisek, dobu trvání a cílovou částku kampaně. Zároveň zde organizátor nahrává skrze vytvořenou implementaci obrázek do služby Pinata IPFS.

Níže je zobrazen kód formuláře, do kterého organizátor vyplní název s popisem, dále dobu, kdy bude kampaň ukončena a finanční cíl kampaně spolu s IPFS obrázkem. Celý formulář je stylizován tak aby byla zajištěna přehlednost a uživatelská přívětivost včetně vhodné estetické vizualizace.

Kód 65. Formulář pro vytvoření kampaně

```
<form id="create" onSubmit={handleSubmit} className="bg-base-100 p-6">
  <div className={styles.inputContainer}>
    <input
      id="name"
      type="text"
      placeholder="Name of the campaign"
      className="input input-bordered input-primary w-full"
      value={name}
      onChange={onNameChange}
      required
    />
    <textarea
      id="description"
      placeholder="Description of the campaign"
      className="input-bordered input-primary w-full"
      value={description}
      onChange={onDescriptionChange}
      required
    />
      // …
  </div>
  <button type="submit" className="btn btn-primary">
      Create Campaign</button>
```

Pro vizualizaci a notifikace uživatele o stavu prováděných akcí je využita komponenta z balíčku react-toastify [63], která zpřehledňuje vykonané stavy. Uživatel tak okamžitě má přehled, zda-li byla daná akce vykonána s úspěchem či ne.

Kód 66. Integrace toast notifikací pro vizuální zpětnou vazbu

```
<ToastContainer position="top-center" />
<button onClick={uploadImageToIPFS}
      className="btn btn-secondary">Upload Image</button>
```

### 5.7.3 Prezentační vrstva admin-feature.tsx

V rámci souboru *admin-feature.tsx* je implementována prezentační vrstva, která je určena pro uživatele s administrátorskými oprávněním. Administrátor tak za pomocí této vrstvy může spravovat jednotlivé kampaně v rámci dané crowdfundingové platformy.

Administrátor po inicializaci zde může jednotlivé kampaně schválit či zamítnout a v rámci svých práv rovněž převést své oprávnění na jiného uživatele. Tato vrstva rovněž poskytuje filtrování na základě, zda-li byla kampaň schválena či zamítnuta.

Kód níže slouží poskytuje základní rozhraní administrátora v rámci kampaně, kde má na výběr pouze ze dvou možností a to schválení, či zamítnutí kampaně.

Kód 67. Rozhraní pro administrátorskou kontrolu kampaní

```
<div className="admin-interface">
  {campaigns.map((campaign) => (
    <div key={campaign.publicKey.toBase58()} className="campaign">
      <h3>{campaign.account.name}</h3>
      <button onClick={() => reviewCampaign(campaign.publicKey)}>
            Review</button>
      <button onClick={() => cancelCampaign(campaign.publicKey)}>
            Cancel</button>
    </div>
  ))}
</div>
```

Pro informování uživatele a zobrazení stavu akcí, které uživatel provede, je v tomto případě využívána komponenta z balíčku react-toastify [63]. Tato komponenta umožňuje administrátorovi okamžitě vidět, zda-li byla akce úspěšně dokončena nebo ne.

Kód 68. Integrace toast notifikací pro informování administrátora

```
<ToastContainer position="top-center" />
<button onClick={initAdmin} className="btn btn-primary">
      Initialize Admin</button>
```

### 5.7.4 Prezentační vrstva portfolio-feature.tsx

Soubor *portfolio-feature.tsx* obsahuje prezentační vrstvu určenou pro správu a zobrazení kampaně. Platforma organizátorovi zobrazuje pouze kampaň jejíž je vlastníkem v rámci crowdfundingové aplikace. Rovněž tato vrstva zobrazuje detaily o kampani, včetně možnosti pro vybrání finančních prostředků, v neposlední řadě rovněž zobrazuje *ProgressBar*, díky kterému je vizuálně znázorněna aktuální finanční podpora dané kampaně.

Kód níže zahrnuje komponentu *Countdown*, která slouží pro dynamický odpočet do ukončení kampaně, který se aktualizuje každou sekundu. Tato komponenta tedy zaručuje informaci o aktuálně zbývajícím čase kampaně.

Kód 69. Implementace časového odpočtu pro zbývající čas kampaně

```
const Countdown: FC<{ endTime: number }> = ({ endTime }) => {
  const [timeLeft, setTimeLeft] = useState(() =>
    calculateTimeRemaining(endTime));
  useEffect(() => {
    const timer = setInterval(() => {
      const newTimeLeft = calculateTimeRemaining(endTime);
      setTimeLeft(newTimeLeft);}, 1000);
    return () => clearInterval(timer);
  }, [endTime]);
  return (
    <div>
      {timeLeft.distance <= 0 ? "ended" : `${timeLeft.days} days
      ${timeLeft.hours}h ${timeLeft.minutes}m`}
    </div>);};
<Countdown endTime={campaign.endCampaign.toNumber() * 1000} />
```

### 5.7.5 Prezentační vrstva [campaignId.tsx]

Soubor *[campaignId].tsx*, je zaměřen na zobrazování detailů jednotlivých crowdfundingových kampaní na platformě. Poskytuje podporovatelům možnost podpořit vybrané kampaně, případně zrušit svoji podporu či zobrazit seznam podporovatelů. Rovněž tento soubor integruje komponenty jako je dynamický časový odpočet do ukončení kampaně, či vizuální zobrazení pokroku financování.

Kód níže zobrazuje implementaci podpoření kampaně, což umožňuje podporovatelům finančně přispět na vybranou kampaň pomocí dynamického formuláře.

Kód 70. Podpora kampaně s validací a odesláním dat k zapsání do blockchainu

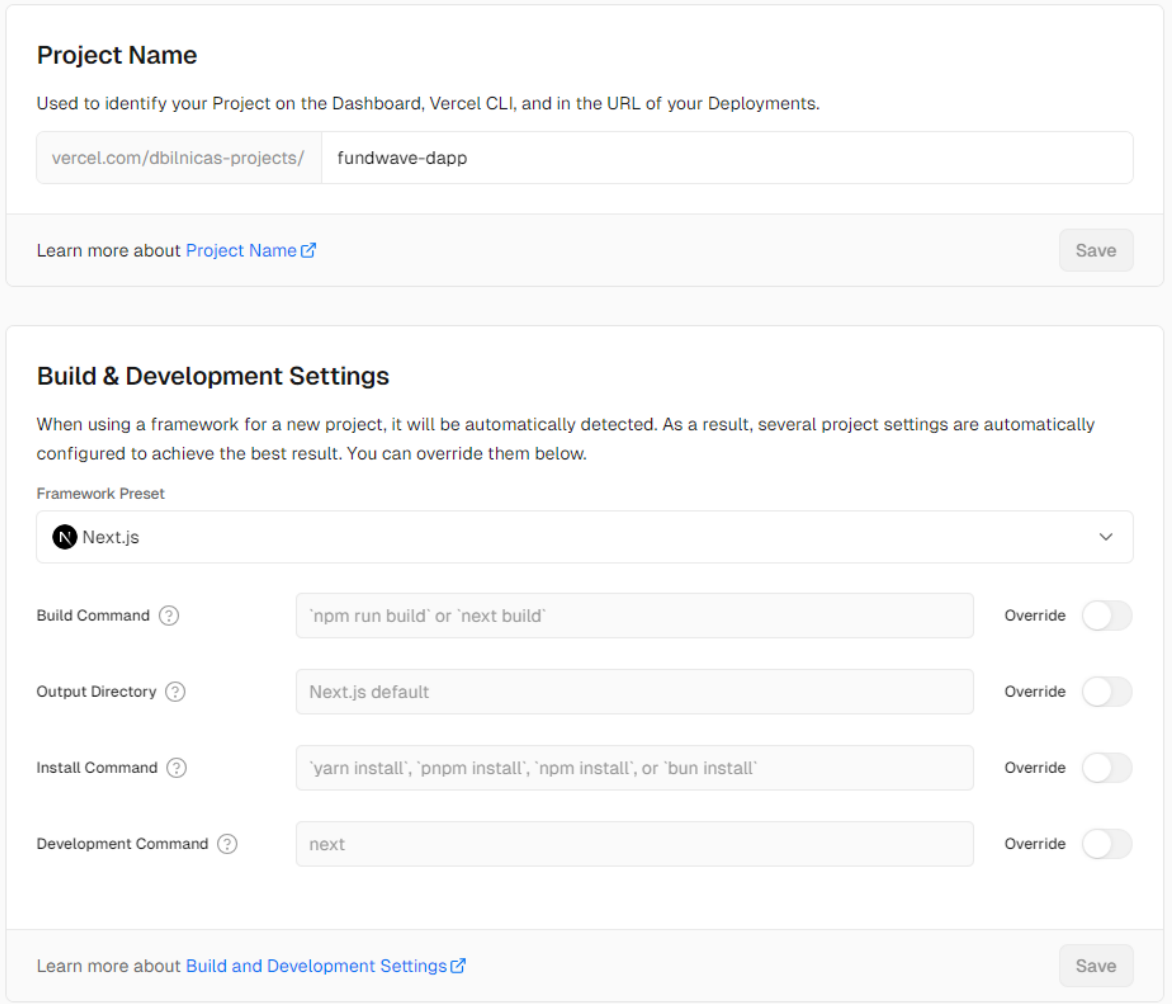
```
const supportCampaign = async (amount) => {
  const amountLamports = solToLamports(amount);
  if (isNaN(amount) || amount < MIN_SUPPORT_AMOUNT_SOL || amount >
      MAX_SUPPORT_AMOUNT_SOL) {
    toast.error(`Support amount must be between
${MIN_SUPPORT_AMOUNT_SOL} and ${MAX_SUPPORT_AMOUNT_SOL} SOL.`);
    return;} setIsSupporting(true);
  try {
    const anchProvider = await getProvider();
    const program = new Program(idl_object, programID, anchProvider);
    await program.methods
      .campaignSupport(new BN(amountLamports)).accounts({
        campaign: campaignPublicKey,
        user: anchProvider.wallet.publicKey,
        systemProgram: web3.SystemProgram.programId,}).rpc(); //…
};
```

## 5.8 Publikování projektu pomocí služby Vercel

Pro publikování vytvořené crowdfundingové aplikace byla použita služba Vercel, která je hojně využívána projekty vytvořenými pomocí Next.js. Platforma Vercel je využívána především pro hostování a následnou automatizaci nasazení projektu s integrací vývojového a produkčního prostředí s přednastavenými funkcemi pro optimalizaci výkonu a bezpečnosti [74].

### 5.8.1 Konfigurace a publikování pomocí Vercel

Aplikace lze velmi jednoduše přenést do služby Vercel skrze GitHub repozitář, který se propojí s touto platformou. Vercel zároveň automaticky detekuje konfiguraci projektu Next.js a doporučí přednastavené workflow pro build a deploy aplikace. Ve výsledku je proces nasazení zcela automatizovaný a v případě potřeby lze nastavení dále upravovat podle specifických potřeb vytvářeného projektu [75].




Obrázek 30. Nastavení buildu a vývojového prostředí na platformě Vercel

### 5.8.2 Přidání vlastní domény

Služba Vercel umožňuje k jednotlivým projektům přiřadit vlastní adresu k nasazenému projektu. Celý proces přidání domény zahrnuje konfiguraci DNS záznamů u poskytovatele domény. V rámci této konfigurace je nutné nastavit požadované A a CNAME záznamy u požadované domény.

Kód 71. Nastavení CNAME a A záznamů pro Vercel

```
// A záznamy pro Vercel
76.76.21.21

// CNAME záznam pro Vercel
cname.vercel-dns.com
```

Po konfiguraci DNS záznamů a ověření vlastnictví dané domény v prostředí Vercel je doména aktivní a lze ji použít v přiřazeném projektu.


Obrázek 31. Domény přiřazené k projektu na platformě Vercel

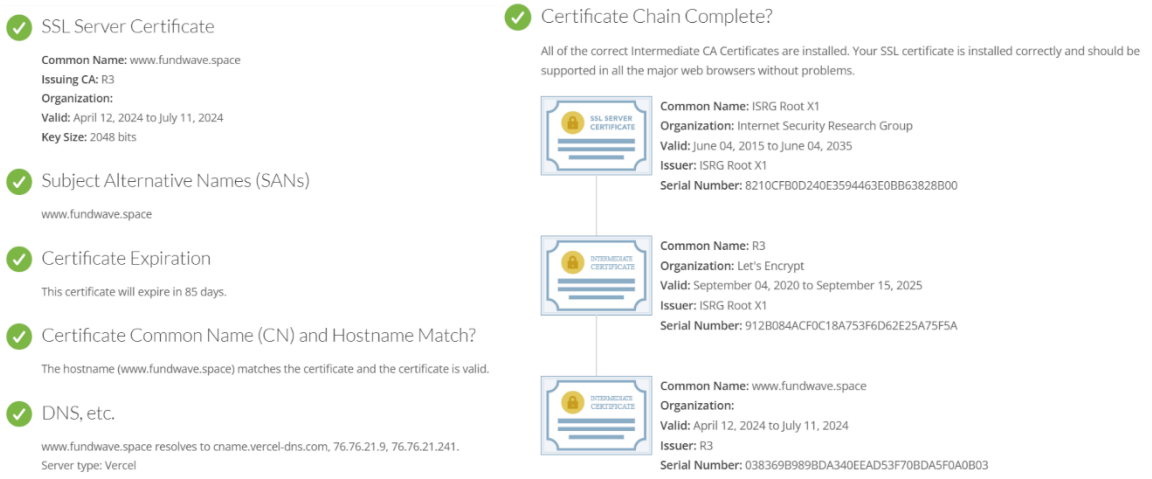
### 5.8.3 Zabezpečení nasazené aplikace

Vercel při nasazení aplikaci automaticky zajišťuje HTTPS spojení pro všechny projekty pomocí SSL/TLS certifikáty. Tato funkcionalita primárně zvyšuje bezpečnost a důvěryhodnost aplikace, protože k tomu využívá certifikáty.

V rámci front-endové části je tedy dostupný SSL/TLS certifikát, který zajišťuje přímý přístup skrze šifrovaný protokol HTTPS. Výsledný report prováděných testů SSL certifikátu pomocí služby www.ssllabs.com a www.geocerts.com neobjevily žádné skutečné, či potencionální rizika a vyhodnotily tedy webovou stránku jako bezpečnou.

| | Server | Test time | Grade |
|---|---|---|---|
| 1 | 76.76.21.142<br>Ready | Tue, 16 Apr 2024 21:49:44 UTC<br>**Duration:** 44.315 sec | A+ |
| 2 | 76.76.21.22<br>Ready | Tue, 16 Apr 2024 21:50:28 UTC<br>**Duration:** 43.863 sec | A+ |

Obrázek 32. Report o provedeném SSL/TLS testu pomocí služby www.ssllabs.com [76]




Obrázek 33. Report o provedeném SSL/TLS testu pomocí služby www.geocerts.com [77]

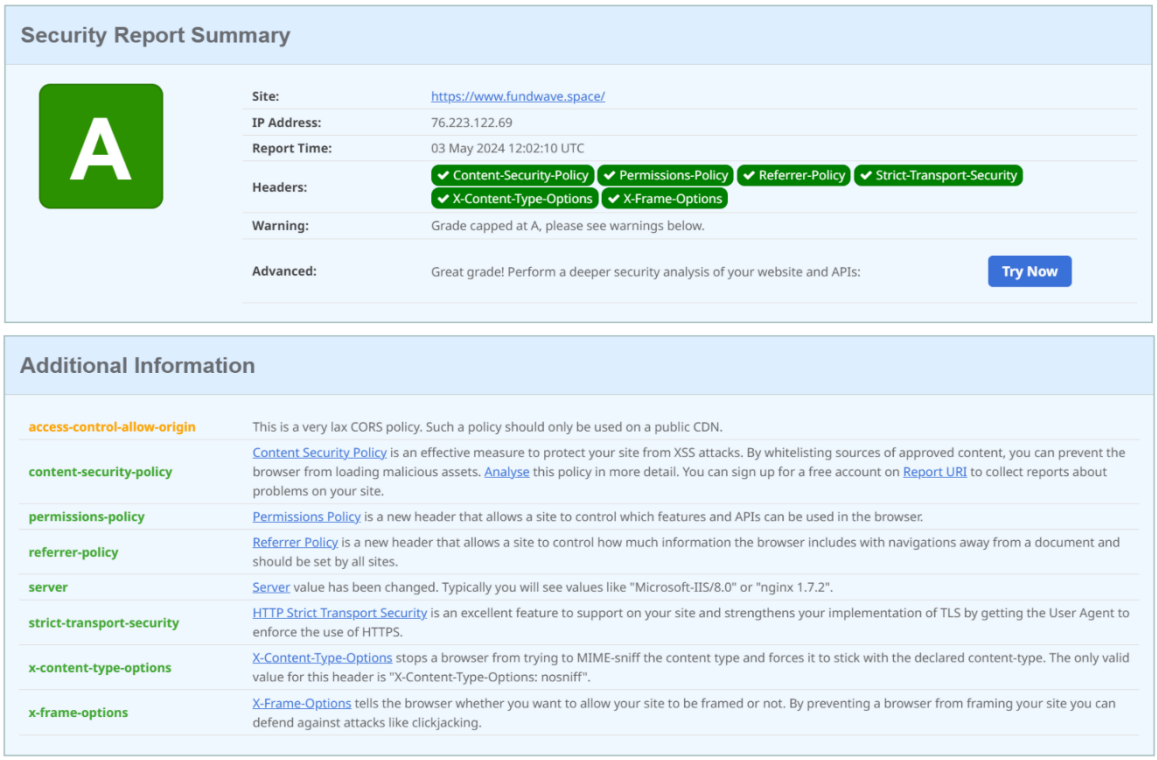
### 5.8.4 Security Headers

Nastavení bezpečnostních hlaviček je klíčovou součástí zabezpečení webové aplikace, neboť umožňuje omezit určité funkce a tím pádem zvýšit ochranu soukromí uživatelů. Tyto hlavičky rovněž slouží jako určitý druh nástroje pro prevenci vůči řadě webovým útokům. Pro implementaci Security Headers bylo zapotřebí vytvoření konfiguračního souboru *vercel.json* v kořenovém adresáři aplikace, ve kterém byly specifikovány požadované bezpečnostní direktivy.

Kód 72. Nastavení Security Headers pro Vercel aplikaci

```
{"headers": [
        {"source": "/(.*)",
         "headers": [ // … ,
          {"key": "X-Frame-Options", "value": "DENY"},
          {"key": "X-Content-Type-Options", "value": "nosniff"},
          {"key": "Permissions-Policy",
              "value": "geolocation=(), microphone=()"}, //… ,]}
    ]
}
```

Pro ověření efektivity nastavených Security Headers bylo využito online vyhodnocovacího nástroje www.securityheaders.com. Tento nástroj poskytuje analýzu aktuálního nastavení bezpečnostních hlaviček a identifikuje potencionální rizika s nimi spojená. Přestože byla aplikace nasazena a testována pouze v testovacím prostředí, výsledky analýzy byly povzbuzující. Aplikace obdržela hodnocení A, což představuje vysokou úroveň zabezpečení, a zároveň zde byly uvedeny i případně doporučení pro zlepšení. Výsledný report je prezentován na obrázku níže.



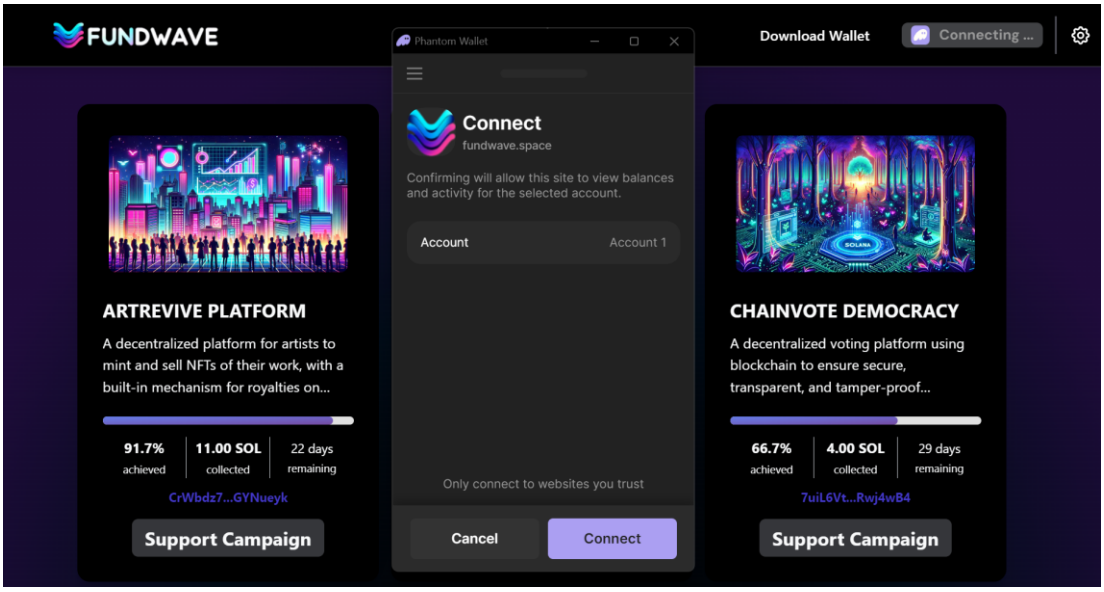

Obrázek 34. Testování Security Headers vytvořené aplikace pomocí služby www.securityheaders.com [78]

## 5.9 Webová prezentace

V rámci webové prezentace pro potřeby vytvoření loga a rovněž při vytváření obrázků pro jednotlivé kampaně bylo využito generativního AI modelu DALL·E [68]; [69].

Pro interakci s aplikací je zapotřebí, aby uživatelé připojili svou webovou peněženku Phantom Wallet, který byla popsána v kapitole *4.7 Phantom Wallet*. Proces připojení začíná výzvou, ve které je uživatel požádán o potvrzení své peněženky prostřednictvím transakce. Po ověření adresy může uživatel provádět jednotlivé operace na platformě.

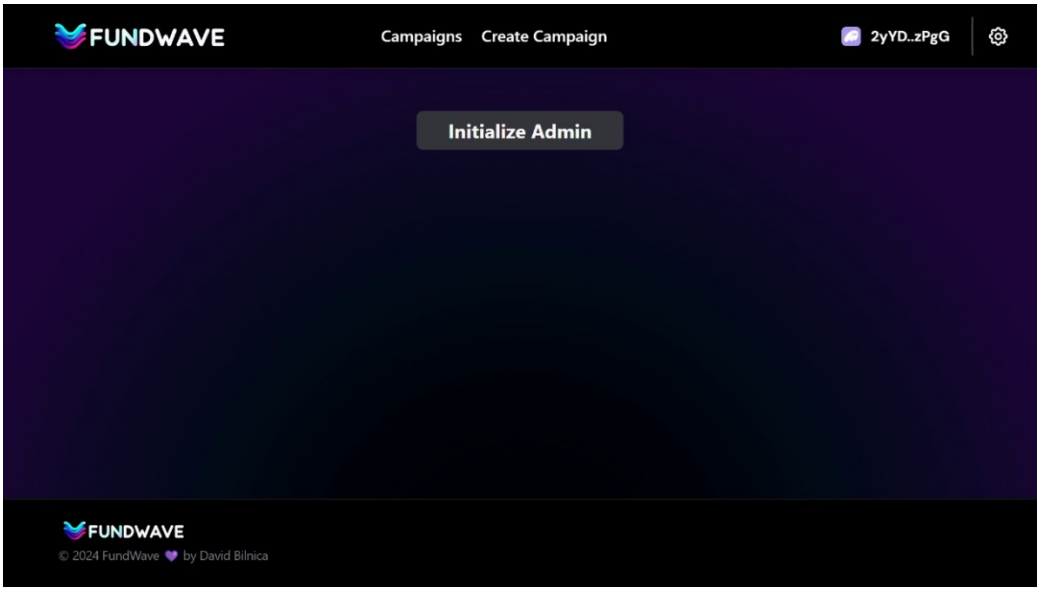
Na obrázku níže je zobrazena operace pro připojení peněženky do platformy. Dialogové okno zde žádá uživatele o potvrzení připojení k platformě FundWave.




Obrázek 35. Připojení peněženky Phantom Wallet do aplikace

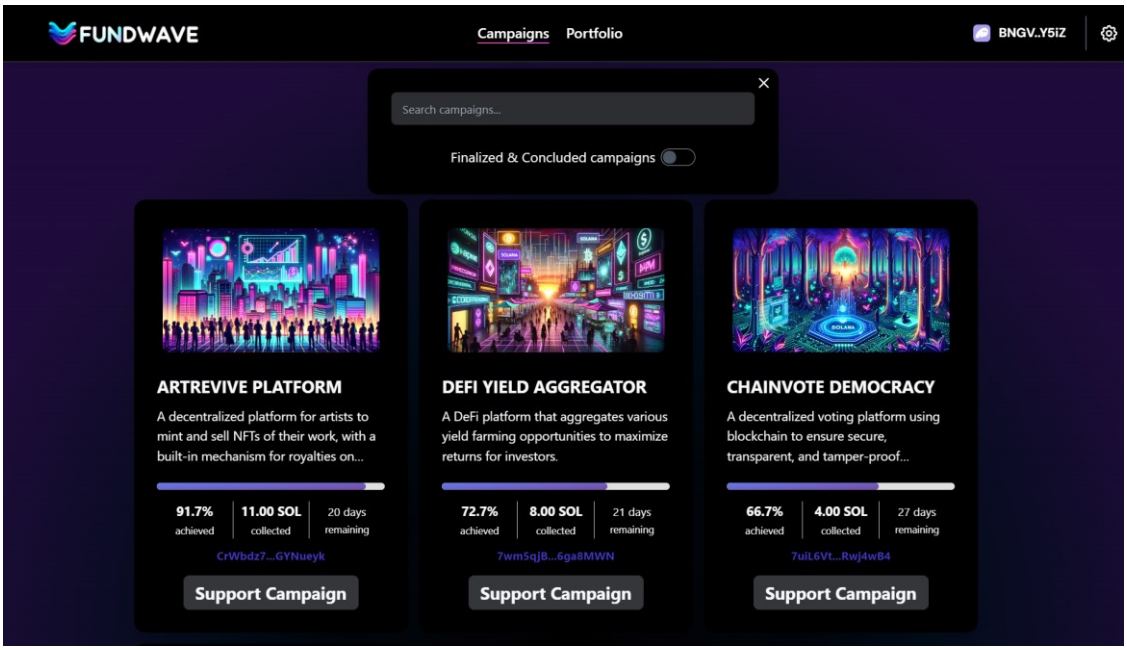
Inicializace administrátora – Pro zajištění řízení a správu crowdfundingové platformy je zapotřebí provést inicializaci administrátora. Tento proces se odehrává na administrátorském dashboardu, kde je z prostředí peněženky potvrzena transakce, pomocí které je nastavena adresa administrátora crowdfundingu. Teprve po úspěšné inicializaci administrátora mohou být prováděny jednotlivé operace na platformě.

Na obrázku níže je možné vidět rozhraní s tlačítkem „Initialize Admin“, které slouží k zahájení procesu inicializace administrátora a následně potvrzen skrze peněženku Phantom Wallet.




Obrázek 36. Inicializace administrátora platformy

Hlavní stránka s aktivními kampaněmi na platformě – Hlavní stránka platformy zahrnuje přehled všech aktivních kampaní. Každá kampaň zde má vlastní vizuál, včetně ukazatele finančního pokroku a zbývající doby kampaně. Funkce vyhledávání pak umožňuje uživatelům objevovat specifické projekty, zatímco filtr nabízí možnost pro zobrazení již ukončených kampaní.




Obrázek 37. Hlavní stránka s aktivními kampaněmi

Vytvoření kampaně – Organizátor kampaně ve formuláři vyplní základní informace, které zahrnují název kampaně a její detailní popis. Pro zajištěné přesného vymezení cílů

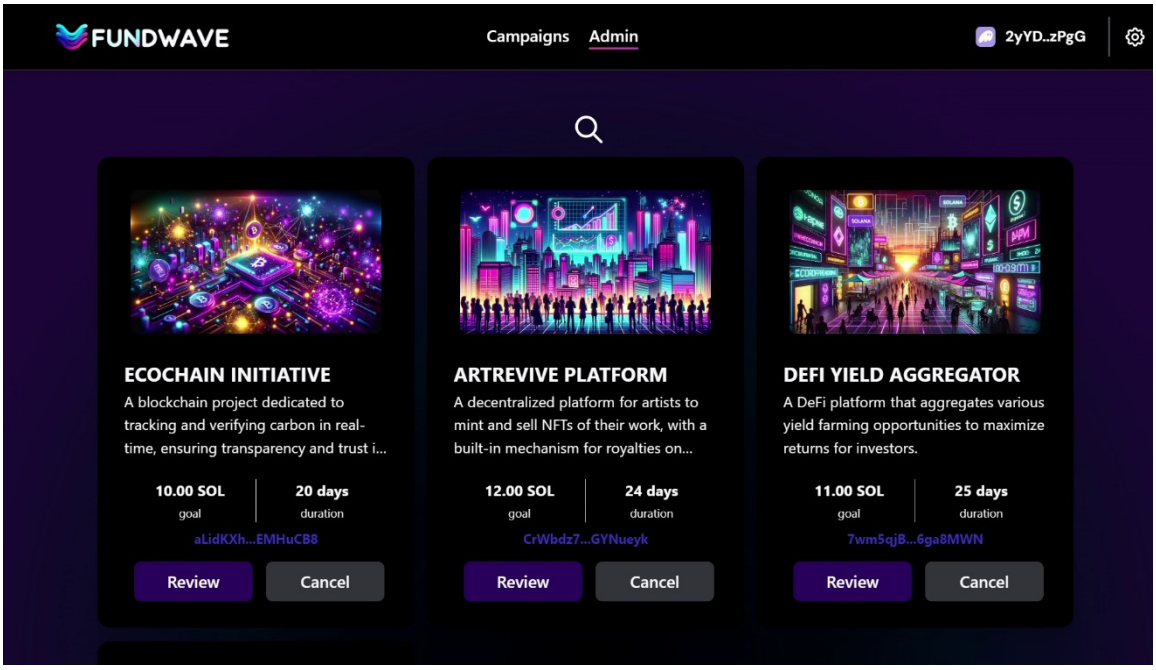
kampaně organizátor kampaně dále nastaví finanční cíl a dobu trvání kampaně. Finální informací je nahrání obrázku, které bude dále prezentován v hlavičce detailu kampaně a v kampani samotné. Po dokončení těchto kroků organizátor kampaně potvrdí transakci pomocí peněženky Phantom Wallet.

Na obrázku níže je zobrazeno popisované rozhraní pro vytvoření kampaně, která obsahuje vstupní pole a tlačítko pro nahrání obrázku.


Obrázek 38. Rozhraní pro vytvoření nové kampaně na platformě

Proces schvalování a zamítnutí kampaní – Po odeslaném návrhu kampaně organizátorem je administrátor pověřen přezkoumáním jednotlivých kampaní. Jeho úkolem je tedy posouzení kampaně a v případě, že kampaň odpovídá kritériím, tak administrátor kampaň schválí, čímž se stane viditelná a může tedy získávat finanční podporu od podporovatelů. V opačném případě je kampaň zamítnuta a tím pádem ztrácí nárok na možnost podpory. Veškeré akce jsou potvrzovány skrze Phantom Wallet a stav kampaně je v uživatelském prostředí dynamicky aktualizován pomocí stavů a vykonaných eventů.

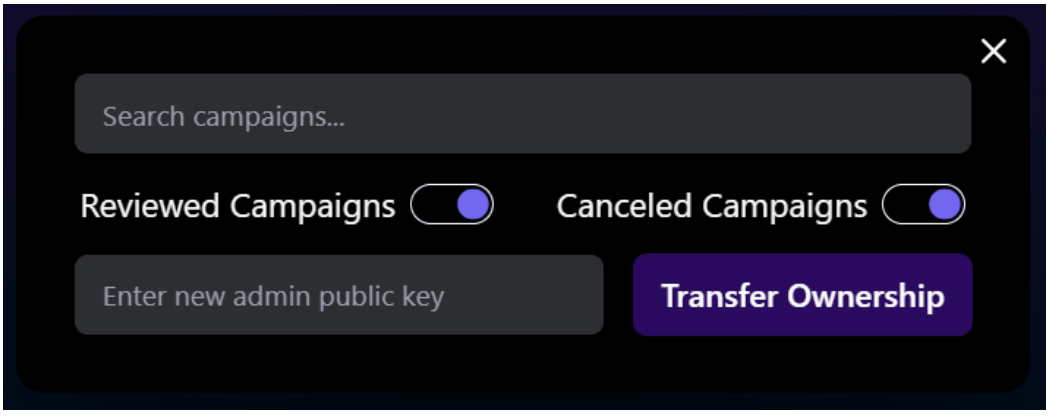
Obrázek níže zachycuje administrátorský dashboard s rozhraním operací pro schvalování, nebo zamítnutí kampaní.




Obrázek 39. Rozhraní administrátora pro schválení nebo zamítnutí kampaně

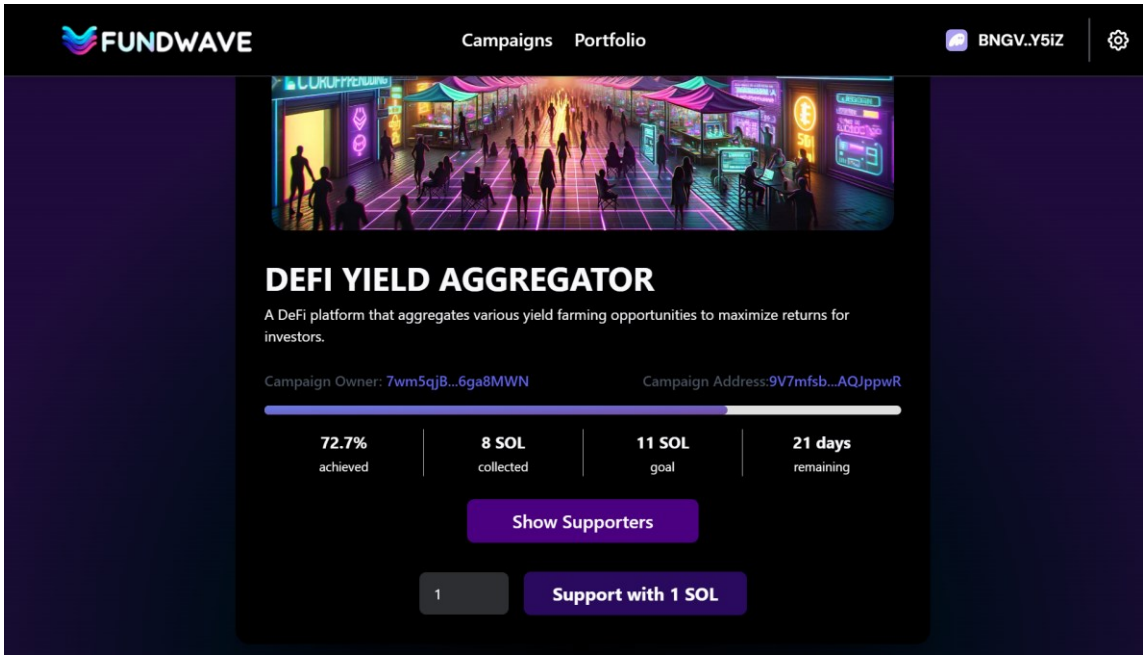
Funkce vyhledávání, filtrování a správy v administračním panelu – V horní části administračního panelu se nachází vyhledávací pole, které umožňuje administrátorovi vyhledat kampaně podle názvu nebo dle popisku. Pro snadnější navigaci v historii projektů jsou k dispozici filtry pro zobrazení již schválených či zrušených kampaní.

Důležitou administrativní funkcí je také možnost převedení vlastnictví administrátorských práv. Administrátor může v poli pro zadání nového veřejného klíče administrátora vložit odpovídající klíč a následně akci potvrdit tlačítkem „Transfer Ownership“. Graficky tyto operace ilustruje obrázek níže.




Obrázek 40. Nástroje pro vyhledávání a správu v administračním panelu

Detaily kampaně a její podpora – Detailní stránka kampaně na platformě poskytuje jednotlivým uživatelům všechny důležité informace: název, popis a reprezentační obrázek v úvodu. Finanční cíl je v detailu znázorněn pomocí progress baru, který barevně zobrazuje procentuální dosažení cíle. Adresa kampaně včetně organizátora jsou součástí detailu a po následném prokliku slouží k přesměrování na Solana Explorer pro zvýšení transparentnosti vytvořené platformy. V dolní části je možné přispět zvolenou částkou a následně transakci potvrdit skrze tlačítko „Support …“, nebo je možné si zobrazit seznam všech podporovatelů. Všechny tyto popisované části jsou zobrazeny na obrázku níže.

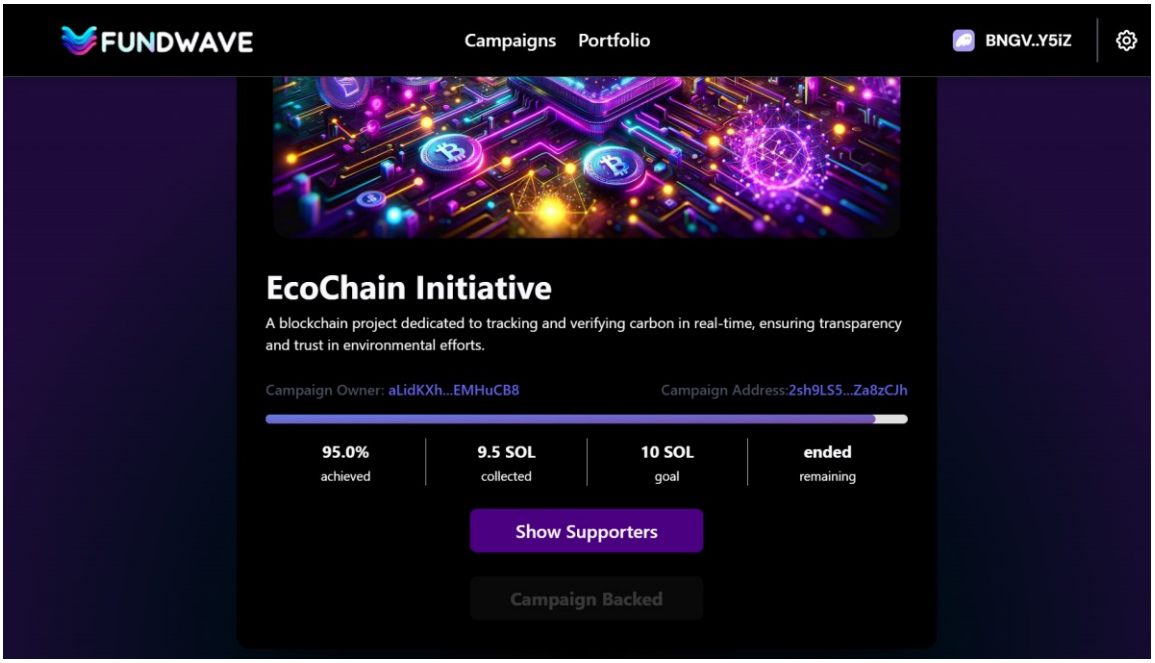



Obrázek 41. Detail kampaně s progress barem a možností podpory na platformě

Zobrazení seznamu podporovatelů – Detail kampaně rovněž zahrnuje komponentovou sekci pro zobrazení seznamu podporovatelů. V tomto seznamu jsou vidět jednotlivé transakce s adresami přispěvatelů a odpovídajícími částkami, kterými podporovatelé přispěli. Každá adresa je tedy přímým odkazem, umožňující ověžení transakce na Solana Exploreru. Obrázek níže demonstruje tuto komponentu.

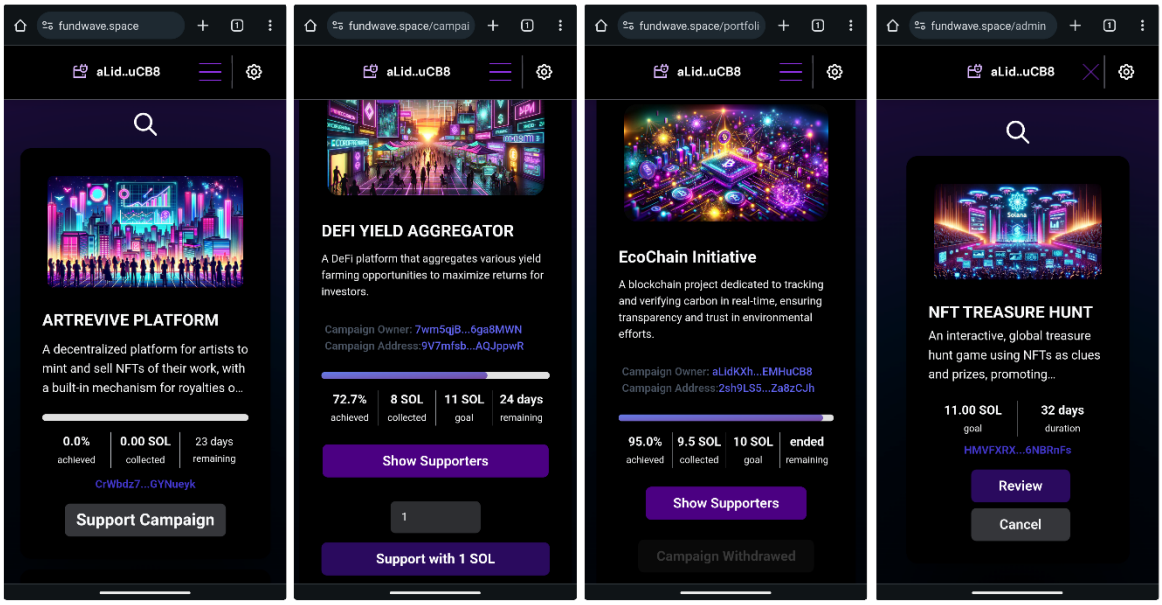

Obrázek 42. List podporovatelů kampaně

Panel Portfolio – Tento panel je specificky navržen pro organizátory kampaně v rámci platformy, díky čemuž poskytuje detailní přehled o stavu jejich kampaně. Zde organizátoři vidí aktuální finanční úspěch kampaně za pomocí progress baru, a po splnění cíle zde mají možnost prostředky vyzvednout. Jakmile jsou finanční prostředky vybrány, kampaň je označena jako úspěšně financovaná a následně ukončena. Obrázek níže demonstruje již ukončenou kampaň.




Obrázek 43. Detailní přehled kampaně v panelu Portfolio

Responzivita na mobilních zařízeních – Crowdfundingová platforma byla navržena tak, aby zajistila responzivitu a přizpůsobení pro mobilní zařízení. Uživatelé tak mají možnost snadného ovládání všech funkcí a panelů přímo ze svých mobilních zařízení. Kromě úpravy responzivity byla přizpůsobena i navigační část platformy, tak aby mohla být ovládána z různých mobilních zařízení. Obrázek níže demonstruje responzivitu vytvořené platformy na mobilním zařízení.




Obrázek 44. Ukázka responzivity platformy na mobilním zařízení

## 5.10 Video prezentace vytvořené platformy

Pro důkladnou demonstraci funkcí a uživatelského prostředí vytvořené decentralizované crowdfundingové platformy byla vytvořena série instruktážních videí. Tato videa ilustrují postup při používání aplikace, tedy od inicializace administrátora, přes vytvoření kampaně, až po samotné finanční podpoření kampaně. Video materiály jsou dostupné v příloze, uložené ve složce *Videos* na přiloženém datovém nosiči. Tímto způsobem může čtenář získat vizuální přehled a pochopení o celkovém průběhu a interakcí s aplikací v rámci všech rolí uživatelů, které platforma podporuje.

# 6 TESTOVÁNÍ SOLANA PROGRAMU

Solana program představuje veškerou logiku crowdfundingových operací, proto proces testování je nezbytný pro ověření vytvořených funkcí. Jakákoliv chyba v logice programu by mohla vést k nevalidním transakcím a v horším případě by mohlo dojít ke zneužití programu útočníky, což by ve výsledku znamenalo ztrátu prostředků daného uživatele, kampaně, či celé platformy. Proto bylo zapotřebí vytvořit několik detailních testovacích scénářů, které simulují různé situace, včetně testování jednotlivých vstupních parametrů a oprávnění uživatele, který transakci tohoto typu vyvolá. Následující podkapitoly detailně popisují, jak probíhalo testování Solana programu.

## 6.1 Testování pomocí frameworku Anchor

Pro zajištění celkové odolnosti a bezpečnosti vytvářené crowdfundingové platformy na Solana blockchainu byl použit již popisovaný framework Anchor (viz. kapitola *4.4.5 Instalace Anchor frameworku*), který zajišťuje nástroje pro snadné vytváření a spouštění jednotkových testů. Testy byly především zaměřeny na ověření bezpečnosti a správnosti všech implementovaných funkcí, včetně bezpečnostních aspektů programu, tak aby bylo možné minimalizovat riziko potencionálních chyb. Pokud by k tomuto ověření a otestování nedošlo, je velká pravděpodobnost, že by se veřejně publikovaná aplikace ocitla velmi rychle pod manipulací útočníků, kteří by se snažili nalézt jakoukoliv chybu v tomto Solana programu.

### 6.1.1 Ukázka jednotkového testu pro inicializaci administrátora

Funkce, která inicializuje administrátora platformy je klíčová pro celkovou správu celé platformy. Test začíná vytvořením běžného účtu, následně pokračuje airdropem Solana tokenů na účet volající transakci a následně je ověřeno, zda-li airdrop byl úspěšně proveden. Následně je přímo zavolá funkce pro inicializaci administrátora a přímo po jejím vykonání je assertováno, zda-li inicializovaná veřejná adresa administrátora shodná s uživatelem, který transakci zavolal a potvrdil.

Kód 73. Ověření inicializace administrátorského účtu po airdropu

```
it("TS01TC01 - Should validate admin account initialization with
      airdrop ", async () => {
  const [admin_pkey] = getAdminInitAddress(program.programId);
  await airdrop(provider.connection, admin.publicKey);
  const expectedBalance = 1_000_000_000;
  const actualBalance = await provider.connection.getBalance(
      admin.publicKey);
  assert.strictEqual(actualBalance, expectedBalance, "Post-airdrop
      balance does not match expected.");

  await program.methods.adminInitialize().accounts({
    admin: admin_pkey,
    user: admin.publicKey,
    systemProgram: anchor.web3.SystemProgram.programId
  }).signers([admin]).rpc({ commitment: "confirmed" });
  const adminData = await program.account.admin.fetch(admin_pkey);
  assert.strictEqual(adminData.adminPubkey.toString(), admin.public-
Key.toString(), "Mismatch in admin public key after initialization.");
});
```

### 6.1.2 Validace vstupních parametrů funkcí

Pro zachování potřebné funkcionality jednotlivých funkcí vytvářené crowdfundingové kampaně, je nutné, aby vytvářené testy kontrolovaly a validovaly vstupní parametry. Je tedy zapotřebí vytvářet ověření správných i nesprávných vstupních parametrů, což může být při vytváření kampaně, například délka názvu, či popisku kampaně. Potvrzení transakce se správnými parametry by mělo vést k jejímu úspěšnému vytvoření a v opačném případě by měla transakce s nevalidními parametry končit chybou.

Kód 74. Vytvoření kampaně s validací hraničních hodnot vstupních parametrů

```
it("TS02TC04 - Should create a campaign with boundary values for name
and description ", async () => {
  const [admin_pkey] = getAdminInitAddress(program.programId);
  const [campaign_pkey] =
getCampaignInitAddress(campaignOwner2.publicKey, program.programId);
  await airdrop(provider.connection, campaignOwner2.publicKey);
  const name = 'A'.repeat(MIN_NAME_LEN);
  const description = 'B'.repeat(MIN_DESC_LEN);
      // …
  await program.methods.campaignCreate(name, description, goal,
      duration, imageIpfsHash).accounts({campaign: campaign_pkey,
    user: campaignOwner2.publicKey, admin: admin_pkey,
    systemProgram: anchor.web3.SystemProgram.programId
  }).signers([campaignOwner2]).rpc({ commitment: "confirmed" });

  const campaign = await program.account.campaign.fetch(campaign_pkey);
  assert.strictEqual(campaign.name, name, "Campaign name mismatch at
boundary value");
      // …
  assert.strictEqual(campaign.imageIpfsHash, imageIpfsHash, "Campaign
IPFS hash mismatch at maximum boundary value");
});
```

Vytváření testů je tedy nezbytné a je zapotřebí jednotlivé testovací případy aktualizovat na základě aktualizace kódu Solana programu, nebo změny požadované funkcionality. Zároveň je zapotřebí testovat extrémní, či hraniční možnosti například u vstupních parametrů, a zároveň vytvářet i testovací kombinace spolu s uživatelskými rolemi.

Pro vytvoření skriptu, který bude spouštět Anchor framework je zapotřebí vytvořit soubor *crowdfunding-dapp.ts*, a ten následně definovat v konfiguraci *Anchor.toml* pomocí:

Kód 75. Skript definovaný v Anchor.toml určený pro spuštění testů

```
[scripts]
test = "yarn run ts-mocha -p ./tsconfig.json -t 1000000 tests/**/*.ts"
```

Vytvořené testy, či testovací sady jsou spuštěny za pomocí příkazu:

```
$ anchor test
```

Po spuštění jsou volány jednotlivé testovací případy, které byly definovány a výsledek testů je vypsán do terminálu.


Obrázek 45. Report o provedených testech pomocí Anchor frameworku

## 6.2 Testovací audit

V rámci vytvářeného testovacího procesu byl vytvořen testovací audit, který je součástí této práce v digitální podobě. Audit je součástí přílohy na datovém médiu a v rámci práce se nachází ve složce *FundWaveTesting* jako soubor *Audit FundWave-dApp*. Audit obsahuje kompletní dokumentaci použitých testovacích metodologií a vyhodnocení výsledků vytvořených testů.

# ZÁVĚR

Diplomová práce se zaměřila na vývoj a následnou implementaci decentralizované crowdfundingové aplikace na blockchainu Solana, která demonstruje silný potenciál moderních technologií v oblasti aplikací zaměřených na financování projektů. V teoretické části byly objasněny základní principy a mechanismy této blockchainové technologie, zejména unikátní konsensus mechanismu Proof of History, který je klíčový pro efektivitu a rychlost transakcí na platformě Solana.

Následně praktická část práce detailně popisuje proces návrhu, vývoje a testování aplikace, včetně specifikace funkcí pro zajištění transparentnosti a bezpečnosti transakcí. Již od počátku byl vývoj aplikace realizován s důrazem na uživatelskou přívětivost a možnost širokého využití v různých sociálních a kulturních projektech, čímž je potvrzena adaptabilita Solana blockchainu pro široké spektrum možných aplikací.

Decentralizované aplikace by měly být vytvářeny pomocí bezpečného kódu, a z těchto důvodů bylo tomuto tématu věnována celá kapitola, která rozebírá základní principy a nástroje pro otestování buď jednotlivých funkcí programu, nebo aplikace jako takové. Na základě provedených jednotkových testů byl vytvořen audit, který slouží jako detailní testovací report s přehledem o nalezených problémech.

Vytvořené závěry z této práce ukazují, že blockchain Solana nabízí významný potenciál s mnoha výhodami pro decentralizované aplikace, které vyžadují rychlé a efektivní zpracování transakcí, a to za cenu nízkých nákladů. Tato práce tak přispívá k pochopení možností decentralizovaného crowdfundingu a jeho implementace v praxi, což může do budoucna sloužit jako potencionální vodítko pro další výzkum či vývoj v této aktivně se rozvíjející oblasti.

Výsledky vytvořené práce nejenže potvrzují zmíněné teoretické předpoklady o výhodách decentralizovaných blockchainových platforem, ale také prakticky demonstrují, jak lze tyto technologie aplikovat do reálných sociálně prospěšných projektů. Případný budoucí vývoj by měl pokračovat ve zdokonalování bezpečnostních aspektů a rozšiřování funkcionalit, aby lépe reagoval na měnící se potřeby uživatelů s ohledem na dynamicky se měnící požadavky trhu.

# SEZNAM POUŽITÉ LITERATURY

[1] Internal Market, Industry, Entrepreneurship and SMEs. Online. European Commission. C 2024. Dostupné z: https://single-market-economy.ec.europa.eu/access-finance/guide-crowdfunding/what-crowdfunding/crowdfunding-explained_en. [cit. 2024-03-17].

[2] FSCS Team. What is crowdfunding? Online. FSCS. C 2023. Dostupné z: https://www.fscs.org.uk/news/protection/crowdfunding/. [cit. 2024-03-17].

[3] Nový investiční fenomén – crowdfunding. Online. Investown. C 2024. Dostupné z: https://www.investown.cz/post/novy-investicni-fenomen-crowdfunding. [cit. 2024-03-17].

[4] SMITH, Tim, JAMES, Margaret (ed.). Crowdfunding: What It Is, How It Works, and Popular Websites. Online. Investopedia. C 2023. Dostupné z: https://www.investopedia.com/terms/c/crowdfunding.asp. [cit. 2024-03-17].

[5] VOUTIK, Lana. Crowdfunding App Development: A Technical Guide. Online. In: QUYTECH. C 2022. Dostupné z: https://www.quytech.com/blog/crowdfunding-app-development-guide/. [cit. 2024-03-17].

[6] Startup. Online. Wikipedie. 2023. Dostupné z: https://cs.wikipedia.org/wiki/Startup. [cit. 2024-03-18].

[7] Co je to peer-to-peer. Online. Správa sítě - slovník pojmů. C 2022. Dostupné z: https://www.sprava-site.eu/peer-to-peer/. [cit. 2024-03-18].

[8] Wallet Adapter. Online. GitHub. C 2024. Dostupné z: https://github.com/anza-xyz/wallet-adapter. [cit. 2024-03-18].

[9] Phantom. Online. C 2024. Dostupné z: https://phantom.app/. [cit. 2024-03-18].

[10] Solflare. Online. C 2024. Dostupné z: https://solflare.com/. [cit. 2024-03-18].

[11] How to create a new wallet. Online. Phantom. C 2024. Dostupné z: https://help.phantom.app/hc/en-us/articles/8071074929043-How-to-create-a-new-wallet. [cit. 2024-03-18].

[12] BAMBARA, Joseph J. a ALLEN, Paul R. Blockchain: A Practical Guide to Developing Business, Law, and Technology Solutions. New York : McGraw-Hill Education, 2018. ISBN 9781260115864.

[13] Solana Explorer. Online. C 2024. Dostupné z: https://explorer.solana.com/. [cit. 2024-03-18].

[14] TAPSCOTT, Don a TAPSCOTT, Alex. Blockchain revolution. [London]: Portfolio/Penguin, 2018. ISBN 978-0-241-23786-1.

[15] BASHIR, Imran. Mastering blockchain. Birmingham: Packt Publishing, Limited, 2020. ISBN 9781839211379.

[16] ŠTRÁFELDA, Jan. Co je hash či hashování. Online. Jan Štráfelda: průvodce online projektem. C 2024. Dostupné z: https://www.strafelda.cz/hash. [cit. 2024-03-19].

[17] Blockchain data structure and how Blockchain works. Online. Bytesoft. 2024. Dostupné z: https://bytesoft.vn/en/blockchain-data-structure-and-how-blockchain-works. [cit. 2024-03-19].

[18] How does a blockchain work? Online. Bitpanda. C 2024. Dostupné z: https://www.bitpanda.com/academy/en/lessons/how-does-a-blockchain-work/. [cit. 2024-03-19].

[19] YAKOVENKO, Anatoly. Solana: A new architecture for a high performance blockchain v0.8.13. Online. Web3 Infrastructure for Everyone | Solana. C 2024. Dostupné z: https://solana.com/solana-whitepaper.pdf. [cit. 2024-03-19].

[20] PIERRO, Giuseppe Antonio a TONELLI, Roberto. Can Solana be the Solution to the Blockchain Scalability Problem? Online. 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). 2022, s. 1219-1226. ISBN 978-1-6654-3786-8. Dostupné z: https://doi.org/10.1109/SANER53432.2022.00144. [cit. 2024-04-21].

[21] ANTONOPOULOS, Andreas M. a WOOD, Gavin. Mastering Ethereum: Building Smart Contracts and DApps. Sebastopol, CA : O'Reilly Media, 2018. ISBN 9781491971918.

[22] ROSIC, Ameer. Solana vs Ethereum: A Comprehensive Comparison for 2024. Online. Blockgeeks. C 2024. Dostupné z: https://blockgeeks.com/guides/solana-vs-ethereum/. [cit. 2024-03-19].

[23] EMMANUEL EBUBE, Aguchukwu. Solidity vs. Rust: Everything You Need to Know. Online. Alchemy. 2022. Dostupné z: https://www.alchemy.com/overviews/solidity-vs-rust. [cit. 2024-03-19].

[24] 51% Attack: Definition, Who Is At Risk, Example, and Cost. Online. Investopedia. 2023. Dostupné z: https://www.investopedia.com/terms/1/51-attack.asp. [cit. 2024-03-20].

[25] What is staking? Online. Coinbase. C 2024. Dostupné z: https://www.coinbase.com/learn/crypto-basics/what-is-staking. [cit. 2024-03-20].

[26] CRYPTO- SYSCOIN, Corey. Solana: The 65,000 TPS Blockchain: Warp Speed. Online. Medium. 2020. Dostupné z: https://coreycrypto.medium.com/solana-the-65-000-tps-blockchain-warp-speed-ab34d3ebb85c. [cit. 2024-03-20].

[27] Ethereum vs. Solana: Everything You Need to Know. Online. Drift | Perpetual Swaps on Solana. C 2022. Dostupné z: https://www.drift.trade/blog/ethereum-vs-solana. [cit. 2024-03-20].

[28] MORE, Vishal. Ethereum's Future: Sharding, L2s, and the Rise of Rollups. Online. The Payments Association. 2024. Dostupné z: https://thepaymentsassociation.org/article/ethereums-future-sharding-l2s-and-the-rise-of-rollups/. [cit. 2024-03-20].

[29] AKHMETOV, Vyacheslav. Layer 2 Solutions For Ethereum. Online. Mercuryo Explore. C 2024. Dostupné z: https://mercuryo.io/explore/article/layer-2. [cit. 2024-03-21].

[30] SEN, Sahil. Introduction to Ethereum Rollups. Online. QuikNode. 2023. Dostupné z: https://www.quicknode.com/guides/web3-fundamentals-security/cryptography/introduction-to-ethereum-rollups. [cit. 2024-03-21].

[31] D, Unchaine. What Is Ethereum Sharding? A Beginner's Guide. Online. CoinDesk. 2023. Dostupné z: https://www.coindesk.com/learn/what-is-ethereum-sharding-a-beginners-guide/. [cit. 2024-03-21].

[32] SHARMA, Rakesh. What Is Decentralized Finance (DeFi) and How Does It Work? Online. Investopedia. 2023. Dostupné z: https://www.investopedia.com/decentralized-finance-defi-5113835. [cit. 2024-03-21].

[33] SHARMA, Rakesh. Non-Fungible Token (NFT): What It Means and How It Works. Online. Investopedia. 2024. Dostupné z: https://www.investopedia.com/non-fungible-tokens-nft-5115211. [cit. 2024-03-21].

[34] REIFF, NATHAN. Decentralized Autonomous Organization (DAO): Definition, Purpose, and Example. Online. Investopedia. 2023. Dostupné z: https://www.investopedia.com/tech/what-dao/. [cit. 2024-03-21].

[35] W, Oscar. Solana and its Ecosystem. Online. Medium. 2021. Dostupné z: https://medium.com/block-insight/solana-and-its-ecosystem-ddffa41dbd5a. [cit. 2024-03-22].

[36] BILNICA, David. Visionary Portrait. Online. In: ChatGPT. 2024. Dostupné z: https://chat.openai.com/g/g-2fkFE8rbu-dall-e/c/1738687b-8e79-4c50-ac5d-1ad7e9ac214e. [cit. 2024-04-06].

[37] Solana (blockchain platform). Online. In: Wikipedia. 2024. Dostupné z: https://upload.wikimedia.org/wikipedia/en/b/b9/Solana_logo.png. [cit. 2024-04-06].

[38] Ethereum Logo. Online. In: Commons Wikipedia. 2024. Dostupné z: https://commons.wikimedia.org/wiki/File:Ethereum_logo_2014.svg. [cit. 2024-04-06].

[39] Stack Overflow Trends. Online. In: Stack Overflow - Where Developers Learn, Share, & Build Careers. 2024. Dostupné z: https://insights.stackoverflow.com/trends?tags=vue.js%2Cangular%2Cangularjs%2Cnext.js%2Creactjs%2Csvelte. [cit. 2024-04-06].

[40] The Rust Programming Language. Rust-analyzer - Visual Studio Marketplace. Online. Extension, Visual Studio Marketplace. C 2024. Dostupné z: https://marketplace.visualstudio.com/items?itemName=rust-lang.rust-analyzer. [cit. 2024-04-08].

[41] @solana/web3.js. Online. @solana/web3.js. 2024. Dostupné z: https://solana-labs.github.io/solana-web3.js/. [cit. 2024-04-08].

[42] @project-serum/anchor. Online. Npm. 2024. Dostupné z: https://www.npmjs.com/package/@project-serum/anchor. [cit. 2024-04-08].

[43] Next.js by Vercel is the full-stack React framework for the web. Online. C 2024. Dostupné z: https://nextjs.org/. [cit. 2024-04-09].

[44] Stack Overflow - Where Developers Learn, Share, & Build Careers. Online. C 2024. Dostupné z: https://stackoverflow.com/. [cit. 2024-04-09].

[45] Install the Solana CLI. Online. Home | Solana Validator. C 2024. Dostupné z: https://docs.solanalabs.com/cli/install. [cit. 2024-04-09].

[46] How to install Linux on Windows with WSL. Online. Microsoft Learn: Build skills that open doors in your career. C 2024. Dostupné z: https://learn.microsoft.com/en-us/windows/wsl/install. [cit. 2024-04-09].

[47] Install Rust. Online. Rust Programming Language. 2024. Dostupné z: https://www.rust-lang.org/tools/install. [cit. 2024-04-09].

[48] Crates.io: Rust Package Registry. Online. 2024. Dostupné z: https://crates.io/. [cit. 2024-04-09].

[49] Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Online. 2024. Dostupné z: https://nodejs.org/en. [cit. 2024-04-09].

[50] Connecting to a Cluster with the Solana CLI. Online. Home | Solana Validator. C 2024. Dostupné z: https://docs.solanalabs.com/cli/examples/choose-a-cluster. [cit. 2024-04-09].

[51] Solana Wallet Guide. Online. Web3 Infrastructure for Everyone | Solana. C 2024. Dostupné z: https://solana.com/docs/intro/wallets. [cit. 2024-04-09].

[52] How to get Solana devnet SOL (including airdrops and faucets). Online. Web3 Infrastructure for Everyone | Solana. 2024. Dostupné z: https://solana.com/developers/guides/getstarted/solana-token-airdrop-and-faucets#using-web3-js. [cit. 2024-04-09].

[53] Solana Test Validator. Online. Home | Solana Validator. C 2024. Dostupné z: https://docs.solanalabs.com/cli/examples/test-validator. [cit. 2024-04-09].

[54] Anchor - Introduction. Online. 2024. Dostupné z: https://www.anchor-lang.com/. [cit. 2024-04-09].

[55] CLI. Online. Anchor. 2024. Dostupné z: https://www.anchor-lang.com/docs/cli. [cit. 2024-04-09].

[56] Installation. Online. Anchor - Introduction. 2024. Dostupné z: https://www.anchor-lang.com/docs/installation. [cit. 2024-04-09].

[57] Anchor.toml Reference. Online. Anchor - Introduction. 2024. Dostupné z: https://www.anchor-lang.com/docs/manifest. [cit. 2024-04-09].

[58] Visual Studio Code - Code Editing. Redefined. Online. 2024. Dostupné z: https://code.visualstudio.com/. [cit. 2024-04-09].

[59] Solana-playground. Online. GitHub. Dostupné z: https://github.com/solana-playground/solana-playground. [cit. 2024-04-13].

[60] Solana Playground | Solana IDE. Online. 2024. Dostupné z: https://beta.solpg.io/. [cit. 2024-04-13].

[61] @pinata/sdk. Online. Npm | Home. 2024. Dostupné z: https://www.npmjs.com/package/@pinata/sdk. [cit. 2024-04-14].

[62] An open system to manage data without a central server | IPFS. Online. 2024. Dostupné z: https://ipfs.tech/. [cit. 2024-04-15].

[63] React-Toastify. Online. Npm | Home. 2024. Dostupné z: https://www.npmjs.com/package/react-toastify. [cit. 2024-04-15].

[64] MATZINGER, Claus. Learn Rust Programming: Safe Code, Supports Low Level and Embedded Systems Programming with a Strong Ecosystem. BPB Publications, 2022. ISBN 9789355511546.

[65] PIERRO, Giuseppe Antonio a AMOORDON, Andy. A Tool to check the Ownership of Solana's Smart Contracts. Online. 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). 2022, s. 1197-1202. ISBN 978-1-6654-3786-8. Dostupné z: https://doi.org/10.1109/SANER53432.2022.00140. [cit. 2024-04-21].

[66] Altruismus. Online. Wikipedie, otevřená encyklopedie. 2023. Dostupné z: https://cs.wikipedia.org/wiki/Altruismus. [cit. 2024-04-21].

[67] Vysvětlení byzantské odolnosti proti chybám. Online. Zjistěte vše o blockchainu a kryptoměnách | Binance Academy. 2018. Dostupné z: https://academy.binance.com/cs/articles/byzantine-fault-tolerance-explained. [cit. 2024-04-21].

[68] BILNICA, David. Solana Crowdfunding App Icon. Online. ChatGPT. 2024. Dostupné z: https://chat.openai.com/c/aa8119bf-5e88-4b6d-a237-2ee05a807129. [cit. 2024-04-23].

[69] BILNICA, David. Solana Crowdfunding App Campaign. Online. ChatGPT. 2024. Dostupné z: https://chat.openai.com/g/g-2fkFE8rbu-dall-e/c/8c957000-abf6-426a-8ed0-8f02e6212c56. [cit. 2024-04-23].

[70] CLINTON, David. What is Node.js? Server-Side JavaScript Development Basics. Online. FreeCodeCamp Programming Tutorials: Python, JavaScript, Git & More. 2023. Dostupné z: https://www.freecodecamp.org/news/node-js-basics/. [cit. 2024-04-30].

[71] Solana-labs/dapp-scaffold: Scaffolding for a dapp built on Solana. Online. GitHub. C 2024. Dostupné z: https://github.com/solana-labs/dapp-scaffold. [cit. 2024-05-02].

[72] Bringing Blockchain to the World | Solana Labs. Online. C 2024. Dostupné z: https://solanalabs.com/. [cit. 2024-05-02].

[73] TypeScript: JavaScript With Syntax For Types. Online. C 2024. Dostupné z: https://www.typescriptlang.org/. [cit. 2024-05-02].

[74] Vercel: Build and deploy the best Web experiences with The Frontend Cloud. Online. C 2024. Dostupné z: https://vercel.com/. [cit. 2024-05-03].

[75] Vercel Documentation. Online. Vercel: Build and deploy the best Web experiences with The Frontend Cloud. C 2024. Dostupné z: https://vercel.com/docs. [cit. 2024-05-03].

[76] Qualys SSL Labs. Online. C 2024. Dostupné z: https://www.ssllabs.com/. [cit. 2024-05-03].

[77] SSL Certificates from GeoTrust, Comodo, DigiCert at Wholesale Prices - GeoCerts. Online. C 2024. Dostupné z: https://www.geocerts.com/. [cit. 2024-05-03].

[78] Analyse your HTTP response headers. Online. C 2024. Dostupné z: https://securityheaders.com/. [cit. 2024-05-03].

[79] Home | ethereum.org. Online. C 2024. Dostupné z: https://ethereum.org/en/. [cit. 2024-05-03].

[80] Home | Solidity Programming Language. Online. 2023. Dostupné z: https://solidity-lang.org/. [cit. 2024-05-03].

# SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

| | |
|---|---|
| A | Address record |
| API | Application Programming Interface |
| CLI | Command Line Interface |
| CNAME | Canonical Name record |
| DAO | Decentralized Autonomous Organization |
| dApp | Decentralized Application |
| DeFi | Decentralized Finance |
| DNS | Domain Name System |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol Secure |
| IDE | Integrated Development Environment |
| IDL | Interface Definition Language |
| IPFS | InterPlanetary File Systém |
| JPEG | Joint Photographic Experts Group |
| JPG | Joint Photographic Experts Group |
| JWT | JSON Web Token |
| NFT | Non Fungible Token |
| NPM | Node Package Manager |
| NVM | Node Version Manager |
| P2P | Peer to peer |
| PNG | Portable Network Graphics |
| PoH | Proof of History |
| PoS | Proof of Stake |
| PoW | Proof of Work |

| | |
|---|---|
| SDK | Software Development Kit |
| SSL | Secure Sockets Layer |
| TLS | Transport Layer Security |
| TPS | Transactions per second |
| UI | User Interface |
| URL | Uniform Resource Locator |
| WSL | Windows Subsystem for Linux |
| WWW | World Wide Web |

# SEZNAM OBRÁZKŮ

# SEZNAM TABULEK

# SEZNAM UKÁZEK KÓDU

# SEZNAM PŘÍLOH

# PŘÍLOHA P I: OBSAH CD

**fulltext.pdf** – textová část diplomové práce

Složka **FundWave** obsahuje:

- složku **fundwave-dapp** obsahující:
  - front-endovou část aplikace
  - konfigurační soubory
  - komponenty pro front-endové funkce
- složku **fundwave-program** obsahující:
  - back-endovou část aplikace
  - soubory pro běh a správu programů
  - soubory pro migrace a testy back-endových komponent

Složka **FundWaveTesting** obsahuje:

- soubor **Audit FundWave-dApp**
  - audit dokumentace testování dApp

Složka **Videos** obsahuje:

- video **1_FundWave-add_wallet**
- video **2_FundWave-admin_initialize**
- video **3_FundWave-create_campaign**
- video **4_FundWave-review_and_cancel_campaign**
- video **5_FundWave-support_campaign**
- video **6_FundWave-cancel_campaign_support**
- video **7_FundWave-withdraw_campaign_and_show_supporters**
- video **8_FundWave-phone_responsivity**
- video **9_FundWave-transfer_ownership**
- video **FundWave_dApp_by_David_Bilnica**

## PŘÍLOHA P II: DALL·E PROMPTS

[36] Obrázek vygenerovaný pomocí DALL·E – Visionary Portrait:

A stylized portrait of an individual symbolizing innovation, vision, and leadership, set against a white background. The individual is depicted in a minimalist style, wearing a black turtleneck, blue jeans, and round glasses, embodying the essence of simplicity and focus. They are holding a modern digital device, representing the intersection of technology and human creativity. The overall composition conveys a message of looking forward with determination and insight, inspired by the ethos of visionary leaders without depicting any specific person.

[68] Obrázek vygenerovaný pomocí DALL·E – FundWave Logo:

Create a logo using only the text 'FUNDWAVE' in a contemporary font style. The colors should be a gradient of vibrant blues, purples, and pinks, similar to the color scheme of the previously liked wave icon, but without any additional design elements or icons. The text logo should be sleek, modern, and reflective of a high-tech, innovative crowdfunding platform.

[69] Obrázky vygenerované pomocí DALL·E – Blockchain Crowdfunding Project:

Create an image that represents a crowdfunding app for the cryptocurrency platform Solana. The image should feature a dark, mountainous landscape under a starry night sky. The mountains should have a digital, wireframe look, symbolizing blockchain technology. Above the mountains, incorporate neon lines and geometric shapes that evoke the feeling of digital networks and connections. In the center, include an abstract, glowing neon symbol or icon that represents crowdfunding, such as a neon hand holding a coin or a neon wallet with coins flowing into it. The overall color scheme should include vibrant neon blues, pinks, and purples, giving the image a futuristic and technological ambiance. The style should be sleek and modern, with a touch of cyberpunk aesthetic.

Design an illustration for a crowdfunding platform associated with Solana, emphasizing a neon aesthetic. The scene should be set in a futuristic cityscape at night with tall skyscrapers. The buildings are outlined with bright neon lights in blues, pinks, and purples. In the foreground, include a diverse crowd of people looking at a holographic

display that features the Solana logo and dynamic charts indicating crowdfunding progress. The people are casting colorful shadows that blend with the neon glow of the environment. The sky above is clear with a digital grid pattern, suggesting a connection to the digital realm. The atmosphere should convey a sense of vibrant community engagement within a high-tech financial world.

Craft a visual for a Solana-based crowdfunding application, incorporating a neon and cyberpunk theme. The image should depict a bustling digital marketplace at dusk, with virtual stalls and avatars of people trading cryptocurrencies and funding projects. The atmosphere should be alive with energy, with neon signs flashing Solana's logo and various cryptocurrencies. The sky should be a gradient of sunset colors transitioning into a digital grid, symbolizing the blend of the physical and virtual worlds. The market should be filled with holographic displays showing fundraising campaigns and digital contributions in progress, with streams of light representing the flow of digital currency. The color palette should be rich with neon magentas, cyans, and yellows, creating a vibrant and immersive cybernetic environment.

Illustrate a vibrant scene for a Solana crowdfunding platform, with a focus on a neon futuristic theme. The image should show a cybernetic forest at twilight, with trees made of glowing circuitry and leaves represented by shimmering light particles. In the midst of this digital forest, there should be a clear glade with a holographic interface displaying Solana's symbol and real-time crowdfunding data. Around this interface, avatars of various investors and entrepreneurs are interacting, symbolizing a global, connected community. The colors of the forest should be a mixture of neon green, electric blue, and deep purple, creating a magical, tech-inspired environment. This scene should evoke a sense of wonder and the infinite possibilities of technology and community-driven finance.