

Tvorba progresivní webové aplikace na podporu laktace

Bc. Filip Tomeš



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

Akademický rok: 2024/2025

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: Bc. Filip Tomeš
Osobní číslo: A24592
Studijní program: N0613A140022 Informační technologie
Specializace: Softwarové inženýrství
Forma studia: Prezenční
Téma práce: Tvorba progresivní webové aplikace na podporu laktace
Téma práce anglicky: Development of a Progressive Web Application for Lactation Support

Zásady pro vypracování

- Provedte průzkum stávajících řešení s obdobnou funkcionalitou, stručně je popište a porovnejte.
- Vyberte technologie a nástroje vhodné pro vývoj aplikace s ohledem na cílové prostředí a zaměření.
- Definujte klíčové funkcionální a nefunkcionální požadavky a vytvořte návrh aplikace.
- Na základě návrhu implementujte jednotlivé funkční části aplikace a proveďte důkladné testování zaměřené na funkčnost, stabilitu a uživatelskou přívětivost.
- Výsledky práce vhodně prezentujte, vyhodnoťte, zda aplikace splnila stanovené cíle, a navrhněte možná vylepšení pro budoucí verze.

Seznam doporučené literatury:

1. DOMES, Scott, 2017. Progressive Web Apps with React. Packt Publishing. ISBN 978-1788297554.
2. WARGO, John, 2020. Learning Progressive Web Apps. Addison-Wesley Professional. ISBN 978-0136484226.
3. WAMBACH, Karen a SPENCER, Becky, 2019. Breastfeeding and Human Lactation. 6. Jones & Bartlett Learning. ISBN 978-1284151565.
4. WALKER, Marsha, 2021. Breastfeeding management for the clinician: using the evidence. Fifth edition. Burlington, MA: Jones & Bartlett Learning. ISBN 978-1-284-22548-8.
5. KRUG, Steve, 2013. Don't make me think, revisited: a common sense approach to web usability. 3. Voices that matter. [Berkeley]: New Riders. ISBN 978-0-321-96551-6.
6. LOVE, Chris, 2018. Progressive Web Application Development by Example. Packt Publishing. ISBN 978-1787125421.
7. WALTER, Aaron, 2011. Designing for emotion. Brief books for people who make websites. New York: A Book Apart. ISBN 978-1-937557-00-3.
8. NIELSEN, Jakob a BUDI, Raluca, 2013. Mobile usability. Voices that matter. Berkeley: New Riders. ISBN 978-0-321-88448-0.

Vedoucí diplomové práce: **Ing. Radek Vala, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **27. října 2024**

Termín odevzdání diplomové práce: **2. června 2025**



L.S.

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

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

PROHLÁŠENÍ AUTORA DIPLOMOVÉ PRÁCE

Beru na vědomí, že

- odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení diplomové práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- 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;
- podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- podle § 60 odst. 2 a 3 mohu užít své dílo – diplomovou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- 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 řádně citoval; v případě publikace výsledků budu uveden jako spoluautor;
- odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.
- Při zpracování této práce jsem využil nástroj generativní umělé inteligence ChatGPT výhradně za účelem jazykové a stylistické úpravy textu. Veškerý obsah jsem po použití nástroje pečlivě zkontroloval a upravil dle vlastního uvážení. Za výslednou podobu práce nesu plnou odpovědnost.

Ve Zlíně, dne

.....
podpis autora

ABSTRAKT

Táto diplomová práca sa zaoberá návrhom progresívnej webovej aplikácie určenej pre matky s nedonosenými deťmi. Hlavným cieľom je, aby si matky mohli zaznamenávať a sledovať množstvo odsatého materského mlieka. Práca sa zameriava na analýzu stávajúcich riešení na trhu, identifikáciu špecifických potrieb matiek v tejto oblasti a na vývoj aplikácie, ktorá tieto kritéria splňuje.

Kľúčová slova: nedonosené deti, materské mlieko, laktácia, aplikácia, PWA, KNTB

ABSTRACT

This thesis deals with designing a progressive web application for mothers with premature babies. The main goal is for mothers to be able to record and track the amount of breast milk expressed. The thesis focuses on analyzing the existing solutions on the market, identifying the specific needs of mothers in this area and developing an application that meets these criteria.

Keywords: premature babies, breast milk, lactation, application, PWA, KNTB

Na tomto mieste by som rád pod'akoval všetkým, ktorí ma počas tvorby tejto práce podpo-
rovali.

Ďakujem predovšetkým svojim rodičom za ich mentálnu podporu na diaľku a priateľke za
trpezlivosť a pozitívny prístup, ktorý mi pomáhal zvládnuť náročné chvíle počas písania
práce.

Veľká vďaka patrí vedúcemu mojej záverečnej práce, Ing. Radkovi Valovi, Ph.D., za od-
borné vedenie, konštruktívne rady a ochotu venovať mi svoj čas aj pri nastavovaní univerzit-
ného servera. Bez jeho pomoci by aplikácia nebola v testovateľnom stave.

Špeciálne pod'akovanie patrí personálu Krajské nemocnice T. Bati (KNTB), ktorý prišiel s
ústrednou myšlienkou aplikácie a poskytol mi cenné informácie potrebné k jej realizácii.

Tiež by som rád pod'akoval dekanovi Fakulty aplikovanej informatiky, doc. Ing. Jiřímu Voj-
těškovi, Ph.D., za jeho optimizmus a cenné postrehy počas našich diskusií v rámci stretnutí
na KNTB.

OBSAH

ÚVOD	13
I TEORETICKÁ ČÁST	14
1 LAKTÁCIA A JEJ VÝZNAM	15
1.1 ZÁKLADY LAKTÁCIE A KOJENIA	15
1.1.1 Fyziológia laktácie	15
1.1.2 Polohy pri dojčení a techniky odsávania mlieka	15
1.1.3 Výhody materského mlieka pre imunitný systém dieťaťa	16
1.1.4 Význam kojenia	17
1.1.5 Špecifiká laktácie u predčasne narodených detí	17
1.2 SÚČASNÝ STAV ZAZNAMENÁVANIA ÚDAJOV O LAKTÁCIÍ	17
1.3 POTREBA DIGITALIZÁCIE	18
2 PROGRESÍVNE WEBOVÉ APLIKÁCIE (PWA)	19
2.1 ČO JE TO PWA?	19
2.1.1 Service Worker	20
2.1.2 Web App Manifest	21
2.1.3 HTTPS	22
2.2 PWA V ROKU 2025	22
2.2.1 Limitácie	23
2.2.2 Možnosti	24
2.3 TECHNOLÓGIE VYUŽÍVANÉ PRE VÝVOJ PWA	25
2.3.1 Ionic	26
2.3.2 React	26
2.3.3 Angular	27
2.3.4 Vue	27
2.3.5 Magento PWA Studio	27
2.3.6 Google LightHouse	28
2.4 ÚSPEŠNÉ PWA V RÔZNYCH OBLASTIACH	28
2.4.1 The washington post	28
2.4.2 Uber	29

2.4.3	AliExpress	29
2.5	POROVNANIE PWA, NATÍVNÝCH A HYBRIDNÝCH APLIKÁCIÍ	30
2.5.1	Architektúra a vývoj	32
2.5.2	Výkon a používateľský zážitok	32
2.5.3	Prístup k zariadeniu a funkciám	32
2.5.4	Distribúcia a aktualizácie	33
2.5.5	Náklady na vývoj a údržbu	33
2.5.6	Bezpečnosť	33
2.6	TRENDY A BUDÚCNOSŤ PWA	34
3	POUŽÍVATEĽSKÉ ROZHRAŇIE A PRÍVETIVOSŤ (UI/UX)	35
3.1	ZÁKLADY DESIGNU POUŽÍVATEĽSKÉHO ROZHRAŇIA	35
3.2	DÔLEŽITOSŤ POUŽÍVATEĽSKEJ PRÍVETIVOSTI	37
3.3	PRÍKLADY DOBREJ PRAXE V DESIGNE APLIKÁCIÍ PRE ZDRAVOTNÉ ÚČELY	38
4	BEZPEČNOSŤ A OCHRANA OSOBNÝCH ÚDAJOV	41
4.1	ZÁKLADY OCHRANY OSOBNÝCH ÚDAJOV (GDPR)	41
4.2	AKO ZAISTIŤ BEZPEČNOSŤ DÁT V PWA	42
II	PRAKTICKÁ ČASŤ	44
5	ANALÝZA POŽIADAVKOV	45
5.1	CIEĽOVÁ SKUPINA	45
5.1.1	Potrebné digitálne zručnosti	45
5.1.2	Scenáre použitia	45
5.1.3	Špecifické potreby	46
5.2	KONKUREČNÝ PRIESKUM	46
5.2.1	Medela Family	46
5.2.2	Pump Log	46
5.2.3	Milk Stash	47
5.2.4	My Baby	47
5.2.5	Breastfeeding	48
5.2.6	Silné stránky konkurencie	48
5.2.7	Nedostatky zistené počas testovania	49

5.3	FUNKČNÉ POŽIADAVKY	49
5.4	NEFUNKČNÉ POŽIADAVKY	51
5.5	VÝSLEDOK ANALÝZY	52
6	NÁVRH APLIKÁCIE	53
6.1	ARCHITEKTÚRA APLIKÁCIE	53
6.1.1	Frontend	53
6.1.2	Backend	55
6.2	NÁVRH DATABÁZY	55
6.3	BEZPEČNOSTNÉ OPATRENIA	55
6.4	NÁVRH POUŽÍVATEĽSKÉHO ROZHRANIA	57
7	IMPLEMENTÁCIA	62
7.1	POČIATOČNÉ TECHNICKÉ VÝZVY	62
7.1.1	Notifikácie	62
7.2	BACKEND	65
7.2.1	Lokálne prostredie a inštalácia	65
7.2.2	Modely a databázová schéma	66
7.2.3	API a routovanie	66
7.3	FRONTEND	66
7.3.1	Nastavenie projektu	67
7.3.2	Prvotná pomoc pri inštalácii PWA	68
7.3.3	Štruktúra komponentov a stránok	70
7.3.4	Komunikácia s API a správa stavu	73
7.3.5	Funkcionalita a používateľská interakcia	75
7.4	TESTOVANIE	76
7.4.1	Testovanie backendu	77
7.4.2	Testovanie frontendu	77
7.4.3	Používateľské testovanie	77
7.4.4	Testovanie PWA funkcionality	78
8	VYHODNOTENIE A DISKUSIA	79
8.1	POUŽÍVATEĽSKÉ TESTOVANIE	79

ZÁVĚR.....	80
8.2 MOŽNOSTI DALŠIEHO ROZVOJA.....	80
SEZNAM POUŽITÉ LITERATURY	83
SEZNAM OBRÁZKŮ	85
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	86

ÚVOD

Kojenie je jedným z najdôležitejších aspektov starania sa o novorodenca. Hlavne u predčasne narodených detí, u ktorých má laktácia zásadný vplyv na ich zdravie a vývoj. Matky týchto detí však často čelia mnohým výzvam, ako je potreba častejšieho kojenia, sledovanie príjmu mlieka a udržanie laktácie v psychicky náročných podmienkach. V súčasnej dobe sú údaje o laktácií často zaznamenávané ručne na papier. Tento spôsob zberu dát je pomerne nepraktický, neefektívny a môže viesť k chybám, prípadne strátam dôležitých informácií.

Motiváciou pre vytvorenie tejto práce je potreba digitalizácie procesu zaznamenávania údajov o laktácií. Deje sa tak na podnet od Krajské nemocnice T. Bati ve Zlíně (KNTB) v Zlíne, ktorej personál vidí budúcnosť v transformácií zaužívaných procesov na ich modernizáciu.

Progresívna webová aplikácia (PWA) sprostredkúva moderné a užívateľsky prívetivé riešenie, ktoré umožní matkám rýchlo a prehľadne zaznamenávať informácie o kojení a mať prístup k dátam kedykoľvek a kdekoľvek. Táto aplikácia má potenciál výrazne uľahčiť matkám ich skúsenosť s kojením a prispieť tak k lepšiemu zdraviu detí.

Cieľom tejto práce je navrhnuť a implementovať PWA, ktorá bude podporovať matky v procese laktácie. Aplikácia umožní registráciu a prihlásenie používateľov, personalizáciu profilu, zaznamenávanie údajov o kojení, poskytne možnosti k tvorbe upozornení na ďalšie kojenie, a poskytne prehľadné štatistiky. Dôraz bude prevážne na používateľskú prívetivosť, integritu dát a dostupnosť na rôznych zariadeniach.

Práca je rozdelená na časť teoretickú a praktickú. V teoretickej časti sú popísané základy laktácie, súčasný stav zaznamenávania údajov a technológie použité pre vývoj aplikácie. Praktická časť sa zameriava na analýzu požiadavkov, návrh, implementáciu a testovanie aplikácie. V závere práce sú zhrnuté výsledky a navrhnuté možnosti ďalších vylepšení pre budúce verzie.

I TEORETICKÁ ČÁST

1 LAKTÁCIA A JEJ VÝZNAM

V úvode si treba osvetliť samotný základ problematiky, aby bol nasledujúci prieskum a vývoj o niečo jednoduchší.

1.1 Základy laktácie a kojenia

Laktácia a kojenie predstavujú kľúčové aspekty starostlivosti o novorodenca, a to nielen z hľadiska výživy, ale aj posilňovania imunity a vytvárania vzťahu medzi matkou a dieťaťom. V nasledujúcich častiach budú stručne predstavené základné fyziologické a praktické aspekty týchto procesov.

1.1.1 Fyziológia laktácie

Laktácia je popísaná ako komplexný fyziologický proces, ktorý zahŕňa tvorbu, sekréciu a uvoľňovanie materského mlieka. Tento proces je riadený najmä dvoma hormónmi – prolaktínom a oxytocínom. Prolaktín je zodpovedný za samotnú produkciu mlieka v mliečnych žľazách, zatiaľ čo oxytocín spúšťa tzv. „let-down reflex“ (reflex uvoľnenia mlieka), ktorý umožňuje vypudenie mlieka z prsníka pri saní dieťaťa. [1]

Produkcia materského mlieka prechádza viacerými štádiami:

- Kolostrum: Prvé mlieko produkované po pôrode, je bohaté na imunoglobulíny, bielkoviny a minerály, a poskytuje novorodencovi dôležitú ochranu pred infekciami.
- Prechodné mlieko: Vzniká niekoľko dní po pôrode, má vyšší obsah tukov a laktózy, čím podporuje rast a vývoj dieťaťa.
- Zrelé mlieko: Tvorí sa približne od druhého týždňa po pôrode a je prispôsobené potrebám rastúceho dieťaťa.

[1]

1.1.2 Polohy pri dojčení a techniky odsávania mlieka

Úspešné dojčenie závisí od správnej polohy a prisatia dieťaťa. Existuje viacero polôh, ktoré môžu matky využívať, a je dôležité, aby si každá našla tú, ktorá je pre ňu a jej dieťa najpohodlnejšia. Medzi bežné polohy patria:

- Kolíska: Dieťa je položené bruškom k matke, hlavička je v ohybe lakt'a.
- Krížová kolíska: Podobná ako kolíska, ale matka podopiera hlavičku dieťaťa opačnou rukou.
- Futbalová poloha: Dieťa je položené pod pazuchu matky, ako keby držala futbalovú loptu.
- Ležanie na boku: Matka aj dieťa ležia na boku, tvárou k sebe.

[1]

Správna technika prisatia je kľúčová pre efektívne dojčenie a prevenciu bolesti bradaviek. Dieťa by malo mať v ústach celý dvorec, nielen bradavku. [1]

Odsávanie mlieka je dôležité v prípadoch, keď dieťa nemôže byť dojčené priamo (napr. predčasne narodené deti), alebo keď si matka potrebuje vytvoriť zásobu mlieka. Používajú sa manuálne alebo elektrické odsávačky. Dôležité je dodržiavať hygienu a správnu techniku odsávania, aby sa predišlo komplikáciám [2].

1.1.3 Výhody materského mlieka pre imunitný systém dieťaťa

Materské mlieko je pre dieťa ideálnou potravou nielen z hľadiska výživy, ale aj z hľadiska posilňovania imunitného systému. Obsahuje množstvo biologicky aktívnych látok, ktoré chránia dieťa pred infekciami a alergiami [3].

Medzi tieto látky patria:

- Imunoglobulíny (najmä IgA): Protilátky, ktoré pokrývajú sliznice čreva a dýchacích ciest, čím bránia prenikaniu patogénov.
- Laktoferín: Bielkovina, ktorá viaže železo a tým obmedzuje rast baktérií.
- Lyzozým: Enzým, ktorý ničí bunkové steny baktérií.
- Oligosacharidy: Prebiotiká, ktoré podporujú rast prospešných baktérií v čreve.
- Živé bunky (leukocyty): Biele krvinky, ktoré priamo bojujú proti infekciám.

[4]

Tieto látky pomáhajú chrániť dieťa pred bežnými infekciami, ako sú hnačky, zápaly uší, dýchacie ťažkosti, a znižujú riziko vzniku alergií a autoimunitných ochorení. [4]

1.1.4 Význam kojenja

Kojenie je považované za najlepší spôsob výživy pre novorodenca a dojča. Materské mlieko obsahuje všetky potrebné živiny v optimálnom pomere a je ľahko stráviteľné. Okrem nutričných benefitov poskytuje aj ochranné látky, ktoré posilňujú imunitný systém dieťaťa a znižujú riziko infekcií, alergií, syndrómu náhleho úmrtia dojčiat (SIDS) či niektorých chronických ochorení (napr. obezita, diabetes typu 1 a 2). [3]

Kojenie má významné benefity aj pre matku:

- Rýchlejšie zavinovanie maternice po pôrode vďaka oxytocínu,
- Znížené riziko popôrodného krvácania,
- Dlhodobá ochrana pred rakovinou prsníka a vaječníkov,
- Podpora psychickej pohody a upevnenie vzťahu medzi matkou a dieťaťom.

[3]

1.1.5 Špecifiká laktácie u predčasne narodených detí

Predčasne narodené deti (narodené pred 37. týždňom gravidity) majú špecifické potreby a čelia viacerým zdravotným rizikám. Materské mlieko je pre tieto deti obzvlášť dôležité, pretože obsahuje vyššie koncentrácie ochranných látok a rastových faktorov, ktoré podporujú dozrievanie črevného traktu, imunitného systému a celkový rast. [5]

Kojenie alebo podávanie materského mlieka predčasne narodeným deťom je často náročné – matky môžu čeliť problémom s rozbehom laktácie, potrebujú časté odsávanie mlieka a intenzívnu podporu zo strany zdravotníckeho personálu. V týchto prípadoch je kľúčová edukácia, motivácia a pravidelné monitorovanie laktácie, aby sa zabezpečil dostatočný prísun materského mlieka pre novorodenca. [5]

1.2 Súčasný stav zaznamenávania údajov o laktácií

Na základe diskusie s nemocničným personálom bolo zistené, že súčasný spôsob zaznamenávania údajov o laktácii prebieha tradičnou formou pomocou „pera a papiera“. Tento proces začína už v popôrodnej miestnosti, kde sú matky po pôrode umiestnené spoločne na

oddelení. Každéj matke je pri príchode poskytnutý papier s predtlačenou tabuľkou a pero, prostredníctvom ktorých si sama vedie záznamy o dojčení alebo odsávaní mlieka.

Formát tabuľky je zvyčajne jednoduchý a prehľadný. Obsahuje základné polia, do ktorých matka zaznamenáva:

- čas dojčenia alebo odsávania,
- množstvo odsatého mlieka (v prípade potreby),
- poznámky o priebehu dojčenia (napríklad problémy s prisatím, bolesť, únava),
- prípadné podanie dokrmovania alebo iných doplnkov.

Tieto papierové záznamy sú pravidelne monitorované zdravotníckym personálom, najmä sestrami na oddelení. Sestry na základe údajov vyhodnocujú úspešnosť laktácie, identifikujú možné problémy a v prípade potreby poskytujú matke dodatočnú podporu, motiváciu alebo odporúčania na zlepšenie techniky dojčenia či odsávania.

1.3 Potreba digitalizácie

Súčasný papierový spôsob zaznamenávania údajov má viacero nedostatkov, ktoré viedli k potrebe uvažovať o digitalizácii tohto procesu. Medzi hlavné problémy patrí riziko straty papierových záznamov, obmedzená čitateľnosť údajov, náročnosť archivácie a sťažená možnosť rýchleho sprístupnenia informácií zdravotníckemu personálu. Digitalizácia by umožnila efektívnejšie zhromažďovanie údajov, jednoduchší prístup k historickým záznamom a lepšiu analytiku výsledkov. Navyše by uľahčila spätnú väzbu matkám a podporila ich v zlepšovaní laktácie prostredníctvom prehľadných štatistík a odporúčaní.

2 PROGRESÍVNE WEBOVÉ APLIKÁCIE (PWA)

V posledných rokoch sa progresívne webové aplikácie (PWA) stali populárnym riešením pri vývoji multiplatformových aplikácií. Ide o technologický prístup, ktorý umožňuje vytvárať webové aplikácie s vlastnosťami bežnými pre natívne mobilné aplikácie, ako je možnosť inštalácie, offline režim či push notifikácie. V nasledujúcej časti budú predstavené základné charakteristiky, výhody a technológie, ktoré stoja za týmto konceptom.

2.1 Čo je to PWA?

Progresívna webová aplikácia (PWA) je moderný typ webovej aplikácie, ktorá kombinuje výhody klasických webových stránok a natívnych mobilných aplikácií. PWA sú navrhnuté tak, aby poskytovali používateľom zážitok podobný mobilnej aplikácii priamo vo webovom prehliadači, pričom využívajú štandardné webové technológie ako HTML, CSS a JavaScript. [6]

Základné vlastnosti PWA zahŕňajú:

- Rýchle načítanie a vysokú odozvu: Vďaka technológiám ako je cachovanie obsahu a service workers sa PWA načítavajú rýchlejšie než bežné webové stránky a ponúkajú svižnú odozvu podobnú natívnym aplikáciám.
- Práca offline: PWA umožňujú používateľom pracovať s aplikáciou aj bez pripojenia k internetu, pretože obsah a funkcionality môžu byť uložené priamo v zariadení.
- Push notifikácie: Podobne ako mobilné aplikácie, aj PWA dokážu posielat' používateľom upozornenia, čo zvyšuje ich zapojenie a návratnosť do aplikácie.
- Možnosť inštalácie: Používateľ si môže PWA „pridať na plochu“ svojho zariadenia, čím získava rýchly prístup k aplikácii bez nutnosti sťahovania z App Store alebo Google Play.
- Prístup k hardvéru: PWA majú čiastočný prístup k vybraným funkciám zariadenia, ako sú GPS alebo fotoaparát, čo rozširuje možnosti použitia aplikácie.
- Vzhľad podobný natívnej aplikácii: Prostredie PWA je navrhnuté podľa štandardov mobilných aplikácií, takže používateľský zážitok je veľmi podobný natívnym aplikáciám.

[6]

Technologický základ PWA tvorí:

- **Service Worker:** Skript bežiaci na pozadí, ktorý umožňuje offline režim, cachovanie a spracovanie push notifikácií.
- **Web App Manifest:** Súbor s informáciami o aplikácii (názov, ikona, farby, spôsob spustenia), ktorý umožňuje inštaláciu na domovskú obrazovku zariadenia.
- **Bezpečné spojenie (HTTPS):** PWA musia byť prevádzkované cez zabezpečené spojenie, aby bola zaistená bezpečnosť používateľských dát.

[7]

2.1.1 Service Worker

Service Worker je skript, ktorý beží na pozadí nezávisle od webovej stránky alebo aplikácie a funguje ako sprostredkovateľ medzi aplikáciou, prehliadačom a sieťou. Umožňuje zachytávať a spracovávať sieťové požiadavky, čo umožňuje implementovať offline režim, cachovanie zdrojov, push notifikácie a synchronizáciu na pozadí. [8, 9]

Životný cyklus Service Workera pozostáva z niekoľkých fáz:

- **Registrácia** – aplikácia zaregistruje Service Worker, ktorý sa následne stiahne a pripraví na inštaláciu.
- **Inštalácia** – Service Worker sa inštaluje a môže prednačítavať (cache) základné zdroje aplikácie.
- **Aktivácia** – po inštalácii sa aktivuje a preberá kontrolu nad stránkami v rámci svojho rozsahu (scope).
- **Prevádzka** – zachytáva sieťové požiadavky, spravuje cache a vykonáva ďalšie úlohy na pozadí.
- **Aktualizácia** – prebieha, keď je dostupná nová verzia Service Workera, pričom prechod na novú verziu je riadený tak, aby neovplyvnil používateľský zážitok.

[8]

Klíčové funkcie Service Workera zahŕňajú:

- **Offline podpora** – umožňuje aplikácii fungovať bez pripojenia k internetu tým, že poskytuje obsah z cache.
- **Cachovanie** – efektívne ukladanie a sprístupňovanie zdrojov, čím sa zlepšuje rýchlosť načítania a spoľahlivosť aplikácie.
- **Push notifikácie** – umožňujú aplikácii posielat' používateľom upozornenia aj vtedy, keď aplikácia nie je aktívna.
- **Synchronizácia na pozadí** – umožňuje vykonávať úlohy, ako je synchronizácia dát, keď je zariadenie pripojené k sieti.

[8, 10]

Výhody pre používateľov zahŕňajú rýchly prístup k obsahu aj offline, plynulý zážitok a možnosť dostávať upozornenia. Pre vývojárov znamená Service Worker možnosť vytvárať robustné a spoľahlivé aplikácie, ktoré fungujú aj v nestabilných sieťových podmienkach. [8]

2.1.2 Web App Manifest

Web App Manifest je JSON súbor, ktorý poskytuje prehliadaču informácie o webovej aplikácii potrebné na jej inštaláciu a správne zobrazenie na zariadení používateľa. Tento súbor definuje základné atribúty aplikácie, ktoré ovplyvňujú jej vzhľad a správanie po inštalácii.

[11]

Medzi povinné a najčastejšie používané atribúty patria:

- **name** a **short_name** – názov aplikácie, ktorý sa zobrazuje používateľovi.
- **icons** – zoznam ikon rôznych veľkostí, ktoré sa používajú na domovskej obrazovke alebo v spúšťáči aplikácií.
- **start_url** – URL, ktorá sa načíta po spustení aplikácie.
- **display** – určuje režim zobrazenia (napr. fullscreen, standalone, minimal-ui).

- **background_color** a **theme_color** – definujú farby pozadia a témy aplikácie, ktoré ovplyvňujú vizuálne prvky, ako je splash screen alebo panel s adresou.

[11]

Funkcie Web App Manifestu umožňujú používateľom pridať aplikáciu na domovskú obrazovku s natívnym vzhľadom a správaním, čím sa zlepšuje používateľský zážitok a integrácia s operačným systémom. Aplikácia sa môže spúšťať bez prehliadačového rámu a s vlastným ikonickým označením, čo zvyšuje jej vnímanie ako plnohodnotnej aplikácie. [11]

2.1.3 HTTPS

HTTPS je základným predpokladom bezpečnosti webových aplikácií, vrátane PWA. Tento protokol zabezpečuje šifrovaný prenos dát medzi klientom a serverom, čím chráni komunikáciu pred odpočúvaním, manipuláciou a útokmi typu man-in-the-middle.[12]

HTTPS funguje na princípe SSL/TLS certifikátov, ktoré overujú identitu servera a zabezpečujú dôvernosť prenášaných dát. Certifikáty môžu byť získané zdarma napríklad cez Let's Encrypt, čo umožňuje jednoduché a cenovo dostupné zabezpečenie webových služieb. [12]

Pre PWA je HTTPS povinné, pretože mnohé kľúčové funkcie, ako sú service workery, push notifikácie a prístup k niektorým API, fungujú iba na zabezpečených stránkach. Okrem bezpečnosti HTTPS zlepšuje dôveru používateľov a pozitívne vplýva na SEO a hodnotenie stránok vyhľadávačmi. [12]

PWA sú multiplatformné, čo znamená, že ich stačí vyvinúť len raz a fungujú na všetkých moderných prehliadačoch a zariadeniach bez ohľadu na operačný systém. To výrazne znižuje náklady na vývoj a údržbu v porovnaní s natívnymi aplikáciami, ktoré je potrebné vyvíjať samostatne pre každý operačný systém.[7]

Vďaka týmto vlastnostiam sa PWA stali populárnou voľbou pre firmy a organizácie, ktoré chcú používateľom ponúknuť moderný, rýchly a dostupný digitálny produkt bez nutnosti zložitej inštalácie a správy aplikácií. [6]

2.2 PWA v roku 2025

Naprieč rokmi prešiel vývoj a podpora progresívnych webových aplikácií (PWA) výraznými zmenami od svojho vzniku v roku 2015. V roku 2025 predstavujú PWA technológiu, ktorá sa stabilne rozvíja, ale zároveň čelí viacerým technickým a platformovým výzvam. Moderné

PWA ponúkajú robustnejšie možnosti ako kedykoľvek predtým, najmä vďaka pokroku v oblasti webových štandardov, no stále existujú isté limitácie, ktoré vývojári musia zohľadniť.

Vzhľadom na rozdielnu mieru podpory jednotlivých webových API medzi prehliadačmi a platformami (napr. iOS vs. Android) je pri vývoji PWA dôležité overiť, aké funkcie sú v konkrétnom prostredí dostupné. Na tento účel existujú špecializované online nástroje:

- <https://whatwebcando.today/> – overenie podpory senzorov, notifikácií, prístupu ku kamere, mikrofону a iných PWA funkcií.
- <https://whatpwacando.today/> – interaktívna demonštrácia možností PWA vrátane inštalácie, offline režimu, natívneho vzhľadu a notifikácií.

Tieto nástroje pomohli aj pri vývoji samotnej aplikácie a počas testovania boli použité na identifikáciu technických limitácií, napríklad pri podpore notifikácií na zariadeniach so systémom iOS.

2.2.1 Limitácie

Aj v roku 2025 PWA čelí niekoľkým zásadným obmedzeniam, ktoré spomaľujú jej širšie prijatie, najmä na platforme iOS. [13]

Hlavné problémy zahŕňajú:

- **Manuálny proces inštalácie** - Na rozdiel od Androidu, Safari na iOS neponúka automatickú výzvu na inštaláciu PWA. Používateľ musí aplikáciu manuálne pridať cez funkciu „Pridať na plochu“.
- **Obmedzený prístup k zariadeniam a senzorom** - PWA majú obmedzený prístup k funkciám ako Bluetooth, USB, NFC a iným natívnym API.
- **Závislosť od Safari** - Všetky PWAs sú na iOS viazané na WebKit (Safari engine), aj keď sú otvorené cez iný prehliadač (napr. Chrome alebo Firefox).
- **Neúplná podpora pokročilých webových API** - Safari tradične zavádza nové webové API pomalšie, čo obmedzuje PWA v oblasti moderných funkcionalít (napr. Web Bluetooth, rozšírené platobné brány).

- **Limitácie push notifikácií** - Push notifikácie pre PWAs sú od iOS 16.4 podporované, ale vyžadujú špeciálne podmienky – aplikácia musí byť pridaná na plochu a používateľ musí manuálne udeliť explicitný súhlas.

[13]

2.2.2 Možnosti

Napriek existujúcim limitáciám sa v roku 2025 objavuje viacero pozitívnych trendov, ktoré podporujú adopciu PWA:

- **Hlbšia integrácia s operačnými systémami:** Moderné operačné systémy ako Android, iOS a Windows umožňujú PWA prístup k väčšiemu množstvu natívnych funkcií zariadenia. Medzi nové možnosti patrí napríklad priamy prístup k súborovému systému (upload, download, editácia súborov priamo v aplikácii), pokročilé notifikácie s multimediálnym obsahom a akčnými tlačidlami, či bezproblémové využitie kamery, mikrofónu a Bluetooth rozhraní pre video hovory alebo IoT aplikácie.
- **Rýchlejšie načítanie vďaka HTTP/3:** Zavedenie protokolu HTTP/3 (postaveného na QUIC) prináša dramatické zrýchlenie načítania aplikácií a stabilnejšie pripojenie. To je zásadné najmä pre používateľov s nestabilným internetom, keďže nový protokol znižuje latenciu a zvyšuje bezpečnosť dátových prenosov.
- **Podpora komplexných aplikácií:** Architektúra PWA sa v roku 2025 posunula tak, že umožňuje vývoj veľkých a komplexných aplikácií, ktoré sa svojou funkcionalitou vyrovnajú desktopovým softvérom. Nové možnosti zahŕňajú:
 - *Real-time kolaborácia:* Vďaka vylepšeniam WebRTC a WebSockets môžu PWA podporovať kolaboratívne editovanie a komunikáciu v reálnom čase.
 - *Rozšírené offline úložisko:* Využitie IndexedDB a ďalších technológií umožňuje ukladať väčšie objemy dát offline, čo je vhodné pre CRM systémy či multimediálne platformy.
 - *Natívny výkon:* Vylepšené JavaScriptové enginy a renderovacie pipeline zabezpečujú plynulejší chod a rýchlejšie reakcie aplikácií.

- **Jednoduchšia distribúcia aplikácií:** PWA je možné distribuovať nielen cez web, ale aj prostredníctvom oficiálnych obchodov (Google Play, Microsoft Store), kde sa objavujú vedľa natívnych aplikácií. Vývojári tak môžu udržiavať jednotný kód pre všetky platformy a zároveň osloviť širšie publikum.
- **Zvýšená bezpečnosť s WebAuthn:** Štandard WebAuthn umožňuje PWA využívať biometrické overenie (odtlačok prsta, rozpoznávanie tváre) priamo v prehliadači, čo zvyšuje bezpečnosť a komfort pri prihlasovaní do citlivých aplikácií, ako sú bankovníctvo či e-commerce.
- **Pokročilé offline možnosti:** Najnovšie service workery a caching stratégie umožňujú PWA fungovať ešte efektívnejšie v offline režime, čo je kľúčové pre oblasti s nestabilným internetom alebo pre aplikácie vyžadujúce zber dát v teréne.
- **Vyššia angažovanosť používateľov:** S hlbšou integráciou do operačného systému môžu PWA lepšie využívať push notifikácie, geolokačné služby a synchronizáciu na pozadí, čím udržiavajú používateľov v kontakte s aplikáciou aj mimo jej aktívneho používania.

[14]

Medzi najnovšie vývojové nástroje patrí Workbox 6.x, ktorý zjednodušuje správu service workerov a cachovania, a Lighthouse, ktorý poskytuje detailné audity kvality, výkonu a prístupnosti PWA. [14]

Vďaka týmto inováciám sú progresívne webové aplikácie v roku 2025 ešte výkonnejšie, bezpečnejšie a univerzálnejšie, pričom ich adopcia naprieč odvetvami neustále rastie. [14]

2.3 Technológie využívané pre vývoj PWA

Vývoj progresívnych webových aplikácií (PWA) je dnes postavený na kombinácii štandardných webových technológií (HTML5, CSS3, JavaScript) a špecializovaných frameworkov a nástrojov, ktoré zjednodušujú tvorbu, správu a testovanie moderných aplikácií s natívnym používateľským zážitkom. [15, 16]

Každý z JavaScript frameworkov a knižníc umožňuje jednoducho vytvoriť PWA aplikáciu len pomocou integrácie web app manifestu. [17]

Service worker je voliteľný, nakoľko umožňuje "len" offline podporu. Avšak, nie každý framework má predpripravené komponenty pre čo najlepší užívateľský zážitok. [17]

2.3.1 Ionic

Ionic je open-source framework, ktorý umožňuje vývoj multiplatformových aplikácií s využitím HTML, CSS a JavaScriptu. Jeho silnou stránkou je rozsiahla knižnica UI komponentov optimalizovaných pre mobilné zariadenia a možnosť využívať natívne funkcie zariadenia vďaka pluginom. Vývojári môžu s Ionicom tvoriť PWA, ktoré vyzerajú a správajú sa ako skutočné mobilné aplikácie, pričom framework podporuje aj integráciu s ďalšími populárnymi knižnicami ako Angular, React či Vue. [15, 16]

Výhody:

- Jednoduchá tvorba responzívneho dizajnu a rýchly vývoj vďaka hotovým komponentom.
- Prístup k natívnym API cez pluginy.
- Podpora multiplatformového vývoja (web, iOS, Android) z jedného kódu.

[15, 16]

2.3.2 React

React, vyvinutý spoločnosťou Facebook, je jedným z najflexibilnejších a najvýkonnejších frameworkov na tvorbu progresívnych webových aplikácií. Jeho komponentová architektúra a využitie virtuálneho DOM-u umožňujú efektívne spravovať zložité používateľské rozhrania a optimalizovať výkon aplikácie. React je vhodný pre dynamické a rozsiahle projekty, pričom ponúka bohatý ekosystém knižníc a nástrojov pre PWA (napr. Create React App s PWA šablónou). [15, 16]

Výhody:

- Veľká komunita a množstvo dostupných zdrojov.
- Rýchly vývoj a jednoduchá údržba vďaka opakovane použiteľným komponentom.
- Efektívne renderovanie pomocou virtuálneho DOM-u.

[15, 16]

2.3.3 Angular

Angular, framework od Google, je robustné riešenie pre vývoj rozsiahlych PWA. Disponuje vstavanou podporou pre service workers, generovanie manifestu a optimalizáciu výkonu. Angular CLI umožňuje rýchle generovanie PWA projektov a integráciu s Angular Material pre moderné UI komponenty. Angular je vhodný pre komplexné aplikácie s požiadavkou na škálovateľnosť a dlhodobú údržbu. [15, 16]

Výhody:

- Vstavaná podpora PWA funkcionalít (service worker, manifest).
- Silná typová kontrola vďaka TypeScriptu.
- Rozsiahla dokumentácia a komunita.

[15, 16]

2.3.4 Vue

Vue.js je progresívny JavaScript framework, ktorý vyniká jednoduchosťou použitia a rýchlou implementáciou PWA funkcionalít. Vďaka Vue CLI môžu vývojári jednoducho generovať PWA projekty s predpripravenou štruktúrou a základným service workerom. Vue je vhodný pre menšie aj stredne veľké projekty, kde je dôležitá rýchlosť vývoja a nízka vstupná bariéra. [15, 16]

Výhody:

- Jednoduché učenie a implementácia.
- Rýchle vykresľovanie vďaka virtuálnemu DOM-u.
- Flexibilita pri integrácii ďalších knižníc.

[15, 16]

2.3.5 Magento PWA Studio

Magento PWA Studio je špeciálne navrhnutý balík nástrojov pre vývoj progresívnych webových aplikácií na platforme Magento. Umožňuje vytvárať rýchle, responzívne a škálovateľné e-commerce aplikácie s natívnym používateľským zážitkom. Výhodou je možnosť

využívať existujúcu infraštruktúru Magento a komplexné nástroje na správu, testovanie a nasadenie aplikácie. [16]

Výhody:

- Úplná integrácia s Magento ekosystémom.
- Jednoduchá správa a aktualizácia PWA.
- Optimalizované pre e-commerce riešenia.

[16]

2.3.6 Google LightHouse

Google Lighthouse je open-source nástroj na automatizované testovanie a auditovanie progresívnych webových aplikácií. Pomáha vývojárom identifikovať slabé miesta v oblasti výkonu, prístupnosti, SEO a splnenia PWA štandardov. Lighthouse poskytuje detailné reporty a odporúčania na zlepšenie kvality aplikácie a je neoddeliteľnou súčasťou vývojového procesu moderných PWA. [15, 16]

2.4 Úspešné PWA v rôznych oblastiach

Máme tu pár veľmi zdárnych prípadov, kedy sa veľké korporáty snažia integrovať PWA aj do ich produktov. [18]

2.4.1 The washington post

The Washington Post jedným z najvýznamnejších a najuznávaných denníkov v USA, ktorý sa špecializuje na presné spravodajstvo o politických, ekonomických a vládnych otázkach. V snahe zvýšiť efektivitu svojej mobilnej platformy a osloviť širšie publikum, spoločnosť v roku 2016 implementovala progresívnu webovú aplikáciu (PWA). Vývojári kombinovali technológiu Google AMP (Accelerated Mobile Pages) s PWA, aby dosiahli rýchlejšie načítanie stránok a umožnili offline prístup k obsahu. [19]

Výsledkom bolo:

- 88 % zlepšenie času načítania v porovnaní s tradičnou mobilnou webovou verziou,
- 23 % nárast mobilného vyhľadávania a zvýšenie počtu vracajúcich sa používateľov.

[19]

Implementácia PWA výrazne zlepšila používateľský zážitok, zvýšila počet mesačných návštevníkov a optimalizovala mobilný prístup k správam.[19]

2.4.2 Uber

Spoločnosť Uber, globálny líder v oblasti online objednávanie prepravy, čelila potrebe zrýchliť proces objednávky jazd v prostredí s nízkou kvalitou siete. V reakcii na tieto výzvy Uber vytvoril vlastnú progresívnu webovú aplikáciu (PWA), ktorá priniesla zásadné vylepšenia:

- Aplikácia sa načítava aj v podmienkach nízkej kvality siete (2G),
- Veľkosť aplikácie bola zredukovaná na približne 50 KB po gzip kompresii.

[19]

Vďaka týmto opatreniam sa Uberu podarilo vytvoriť extrémne ľahkú a rýchlu platformu, ktorá umožňuje používateľom rýchlo a efektívne objednávať jazdy bez ohľadu na kvalitu internetového pripojenia. [19]

2.4.3 AliExpress

AliExpress, patriaci pod skupinu Alibaba, predstavuje jednu z najvýznamnejších platforiem v oblasti elektronického obchodu, najmä v segmente mobilného nakupovania. V snahe optimalizovať používateľský zážitok na webovej platforme, ktorý nebol dostatočne plynulý, spoločnosť prešla na využitie PWA technológie. [19]

Vďaka tejto zmene dosiahla:

- Dvojnásobný nárast počtu zobrazených stránok na používateľa počas jednej relácie
- 74 % predĺženie trvania relácie,
- 104 % zvýšenie miery konverzie u nových používateľov a
- 82 % nárast miery konverzie na zariadeniach iOS.

[19]

Nasadenie progresívnej webovej aplikácie výrazne prispelo k zlepšeniu zákazníckej skúsenosti, zvýšeniu angažovanosti používateľov a upevnilo pozíciu AliExpressu na globálnom trhu. [19]

Tieto príklady ukazujú, že progresívne webové aplikácie nachádzajú uplatnenie v rôznych oblastiach — od spravodajstva cez dopravu až po elektronický obchod. Výsledky jednotlivých spoločností jasne naznačujú, že implementácia PWA technológie vedie k rýchlejšiemu načítavaniu stránok, vyššej miere zapojenia používateľov a v konečnom dôsledku k lepším obchodným výsledkom. Preto možno PWA považovať za kľúčový nástroj modernizácie webových platforiem v digitálnej ére.

2.5 Porovnanie PWA, natívnych a hybridných aplikácií

Pri vývoji mobilných aplikácií existujú tri hlavné prístupy: progresívne webové aplikácie (PWA), natívne aplikácie a hybridné aplikácie. Každý z týchto prístupov má svoje výhody, nevýhody a špecifické využitie v závislosti od požiadaviek projektu, rozpočtu či cieľovej skupiny používateľov. [20, 21]

MOBILE APP CLASSIFICATION COMPARISON TABLE				
FEATURES	NATIVE APPS	WEB APPS	HYBRID APPS	PROGRESSIVE WEB APPS
 DEVELOPMENT TECHNOLOGY	Specific to platform (e.g., Swift for iOS, Java/Kotlin for Android)	HTML, CSS, JavaScript	HTML, CSS, JavaScript + native wrappers (e.g., Cordova, Ionic)	HTML, CSS, JavaScript
 INSTALLATION	Installed from app stores (e.g., Apple App Store, Google Play Store)	Accessed via web browser, no installation required	Installed from app stores	Accessed via web browser, can be installed to the home screen
 ACCESS TO DEVICE FEATURES	Full access to device features (camera, GPS, contacts, etc.)	Limited access to device features	Access to device features through plugins	Limited access to device features
 PERFORMANCE	Generally faster and more responsive	May be slower and less responsive than native apps	Slower than native apps, but performance can vary	Performance can be comparable to native apps
 USER EXPERIENCE	Native look and feel, consistent with platform	May lack native look and feel	Can mimic native look and feel, but may not be perfect	Can provide native-like experience
 OFFLINE FUNCTIONALITY	Can work offline with proper implementation	Limited offline functionality	Can work offline with proper implementation	Can work offline with service workers
 UPDATES	Updates require app store approval and user download	Instant updates on the web	Updates require app store approval and user download	Updates instantly available on the web, like a website
 DISTRIBUTION	Distributed through app stores	Accessed via web URLs	Distributed through app stores	Accessed via web URLs, can be installed to the home screen
 CROSS-PLATFORM	Requires separate development for each platform	Accessible on any device with a web browser	Can be deployed across multiple platforms	Accessible on any device with a web browser
 COST	Higher development cost due to platform-specific development	Lower development cost	Moderate development cost due to hybrid development	Lower development cost

Obrázek 2.1 Komparativní tabulka zobrazující hlavní rozdíly mezi natívnými aplikacemi, hybridními aplikacemi, webovými aplikacemi a progresívnými webovými aplikacemi (PWA).

(zdroj: 21)

2.5.1 Architektúra a vývoj

- **PWA** sú vyvíjané pomocou štandardných webových technológií (HTML, CSS, JavaScript) a bežia v prehliadači, no vďaka service workerom a manifestu poskytujú zážitok podobný natívnym aplikáciám. Výhodou je jeden kód pre všetky platformy a jednoduchá distribúcia cez web.
- **Natívne aplikácie** sú vyvíjané špecificky pre konkrétny operačný systém (napr. Swift pre iOS, Java/Kotlin pre Android) a využívajú všetky natívne možnosti zariadenia. Pre každú platformu je potrebné vyvíjať a udržiavať samostatnú aplikáciu.
- **Hybridné aplikácie** kombinujú webové technológie s natívnym obalom (napr. Cordova, Ionic), čo umožňuje nasadenie do obchodov s aplikáciami a prístup k niektorým natívnym funkciám cez JavaScript bridge. Vývoj je rýchlejší ako pri natívných aplikáciách, no výkon a používateľský zážitok môžu byť obmedzené.

[20, 21]

2.5.2 Výkon a používateľský zážitok

- **Natívne aplikácie** poskytujú najvyšší výkon, najplynulejšie animácie a najlepšiu integráciu s operačným systémom.
- **PWA** dosahujú veľmi dobrý výkon, najmä vďaka cachovaniu a offline podpore, no v niektorých prípadoch môžu zaostávať za natívnymi aplikáciami (napr. pri náročných animáciách alebo práci s hardvérom).
- **Hybridné aplikácie** ponúkajú stredný výkon, ktorý závisí od použitého frameworku a komplexnosti aplikácie. Pri zložitejších úlohách môžu byť pomalšie ako natívne aplikácie.

[20, 21]

2.5.3 Prístup k zariadeniu a funkciám

- **Natívne aplikácie** majú plný prístup ku všetkým funkciám zariadenia (kamera, GPS, Bluetooth, senzory, notifikácie atď.).

- **Hybridné aplikácie** majú prístup k väčšine základných funkcií cez pluginy, no nie vždy ku všetkým pokročilým API.
- **PWA** majú prístup len k tým funkciám, ktoré podporuje prehliadač a štandardy webových API. V posledných rokoch sa však možnosti PWA rozširujú a umožňujú napríklad prácu offline, push notifikácie, prístup ku kamere či geolokácii, no stále nemusia pokryť všetky natívne možnosti.

[20, 21]

2.5.4 Distribúcia a aktualizácie

- **PWA** sú dostupné priamo cez web, môžu byť inštalované z prehliadača alebo (v niektorých prípadoch) z obchodov s aplikáciami. Aktualizácie sú okamžite dostupné všetkým používateľom.
- **Natívne a hybridné aplikácie** sa distribuujú prostredníctvom obchodov (Google Play, App Store), kde podliehajú schvaľovaciemu procesu. Aktualizácie musia byť najskôr schválené a používateľ ich musí stiahnuť.

[20, 21]

2.5.5 Náklady na vývoj a údržbu

- **PWA** majú najnižšie náklady, keďže stačí jeden kód pre všetky platformy a aktualizácie sú jednoduché.
- **Hybridné aplikácie** sú stredne nákladné, keďže väčšina kódu je zdieľaná, no je potrebné riešiť špecifiká jednotlivých platforiem.
- **Natívne aplikácie** sú najdrahšie na vývoj aj údržbu, keďže pre každú platformu je potrebné vyvíjať samostatne.

[20, 21]

2.5.6 Bezpečnosť

- **Natívne aplikácie** využívajú bezpečnostné mechanizmy operačného systému a môžu implementovať pokročilé zabezpečenie (napr. biometria, šifrovanie).

- **Hybridné aplikácie** majú prístup k natívnym bezpečnostným prvkom cez obal, ale ich bezpečnosť závisí aj od použitých pluginov.
- **PWA** sa spoliehajú na bezpečnostný model prehliadača a HTTPS, nemajú však prístup k niektorým natívnym bezpečnostným API.

[20, 21]

2.6 Trendy a budúcnosť PWA

- **Viac natívnych funkcionalít vďaka Project Fugu** – Vďaka iniciatíve Project Fugu (spoločný projekt Google, Microsoft, Intel a ďalších) získavajú PWA prístup k čoraz väčšiemu množstvu natívnych API, ako je systémová schránka, Bluetooth, USB či prístup k súborovému systému. Tým sa ešte viac stiera rozdiel medzi webovými a natívnymi aplikáciami.
- **Integrácia umelej inteligencie a strojového učenia** – Vďaka knižniciam ako TensorFlow.js môžu byť do PWA implementované AI funkcie priamo v prehliadači (napr. rozpoznávanie obrázkov, spracovanie prirodzeného jazyka), čím sa zvyšuje interaktivita a personalizácia aplikácií.
- **Zvýšená bezpečnosť a dôvera používateľov** – Nové štandardy ako WebAuthn umožňujú bezpečné prihlasovanie bez hesla (biometria, FIDO2), čím sa zvyšuje ochrana údajov a dôvera používateľov.
- **Optimalizácia výkonu pomocou nových technológií** – Techniky ako lazy loading, code splitting či využitie WebAssembly umožňujú rýchlejšie načítanie a efektívnejšie spracovanie aj náročných úloh (napríklad editácia médií alebo hry priamo v prehliadači).
- **Jednoduchší vývoj a robustnejšie nástroje** – Neustále sa rozširuje ponuka frameworkov, knižníc a nástrojov pre vývoj PWA, čo zjednodušuje tvorbu robustných a funkčne bohatých aplikácií aj pre menšie tímy.

[20]

3 POUŽÍVATEL'SKÉ ROZHRAŇIE A PRÍVETIVOSŤ (UI/UX)

UI a UX predstavujú rozhodujúce aspekty úspešnosti každej aplikácie. V kontexte zdravotníckych aplikácií určených pre čerstvé mamičky, je intuitívne a vizuálne zrozumiteľné rozhranie nevyhnutnosťou. Táto kapitola sa zameriava na zásady moderného mobilného UI dizajnu, špecifiká dotykových rozhraní a faktory, ktoré ovplyvňujú pozitívnu užívateľskú skúsenosť.

3.1 Základy designu používateľského rozhrania

Design používateľského rozhrania v mobilných aplikáciách predstavuje kľúčový faktor úspechu, najmä v zdravotníckom prostredí, kde intuitívnosť a jednoduchosť používania hrajú kritickú úlohu. Podľa najnovších výskumov viac ako 52% webovej návštevnosti pochádza z mobilných zariadení, čo zdôrazňuje dôležitosť optimalizácie mobilného rozhrania. [22]

Medzi základné princípy efektívneho mobilného UI designu patria:

- **Jednoduchosť a prehľadnosť:** Preplnené a neprehľadné rozhranie môže používateľov frustrovať a prinútiť ich aplikáciu opustiť. Pre vytvorenie čistého a intuitívneho UI je potrebný minimalistický prístup, ktorý odstraňuje zbytočné prvky a zameriava sa na základné funkcie a obsah. Konzistentnosť v používaní farebných schém, fontov a ikon je rovnako dôležitá pre ľahšie používanie aplikácie.
- **Vizuálna hierarchia:** Efektívna vizuálna hierarchia usmerňuje pozornosť používateľov k najdôležitejším častiam aplikácie. Využitie veľkosti a blízkosti prvkov, farebných kontrastov a typografie pomáha organizovať obsah a zvyšuje jeho zrozumiteľnosť. Väčšie a výraznejšie prvky priťahujú viac pozornosti, zatiaľ čo správne použitie typografie uľahčuje čítanie a orientáciu v obsahu.

[22]



Obrázek 3.1 Porovnanie odporúčaných veľkostí dotykových prvkov v mobilných aplikáciách.

(zdroj: 21)

- **Responzívny design:** Aplikácia musí vyzerat' a fungovat' dobre na všetkých zariadeniach a veľkostiach obrazoviek. To možno dosiahnuť pomocou adaptívnych layoutov, ktoré sa prispôbujú rôznym veľkostiam obrazoviek, a škálovateľných prvkov, ktoré si zachovávajú svoju jasnosť na rôznych hustotách displejov.
- **Intuitívna navigácia:** Dobrá navigácia je základom plynulého používateľského zážitku. Zahrňuje jasné označenie prvkov pomocou rozpoznateľných ikon a textových popisov, spodnú navigačnú lištu pre rýchly prístup k primárnym funkciám, a intuitívne gestá ako swipovanie alebo pinch-to-zoom pre zlepšenie interakcií používateľ'a.



Obrázek 3.2 Heatmapy znázorňujúce zóny jednoduchého dosahu prstu pri rôznych spôsoboch držania mobilného telefónu

(zdroj: 23)

V kontexte zdravotníckych aplikácií je obzvlášť dôležité dodržiavať pravidlo troch kliknutí, ktoré zabezpečuje, že používatelia môžu dosiahnuť akúkoľvek časť aplikácie maximálne tromi kliknutiami. Toto pravidlo je kritické najmä v nemocničnom prostredí, kde rýchly prístup k informáciám môže byť rozhodujúci. [22]

3.2 Dôležitosť používateľskej prívetivosti

Používateľská prívetivosť nie je len o estetickom vzhľade aplikácie, ale predovšetkým o funkčnosti, ktorá umožňuje používateľom efektívne dosiahnuť svoje ciele. V zdravotníckom prostredí, kde aplikácie často využívajú pod tlakom alebo v stresových situáciách, je prívetivosť rozhrania ešte dôležitejšia. [22]

Kľúčové aspekty používateľskej prívetivosti zahŕňajú:

- **Prístupnosť a inkluzívnosť:**

- **Farebný kontrast:** Podľa odporúčaní WCAG 2.1 by mal byť kontrastný pomer medzi textom a pozadím aspoň 4.5:1 pre normálny text a 3:1 pre veľký text (nad 18 bodov). Pre ikony a grafické prvky sa odporúča kontrast aspoň 3:1. [24]
- **Alternatívny text pre obrázky:** Všetky významové obrázky a ikony by mali mať popisný alt text, ktorý stručne vysvetľuje ich účel. Decoračné obrázky by mali mať prázdny alt atribút (alt=""), aby čítačky obrazovky ich ignorovali. [25]

- **Štruktúra obsahu:** Použitie sémantického HTML (napr. nadpisy <h1> až <h6>, značky <nav>, <main>) a logická hierarchia obsahu zvyšuje prístupnosť pre používateľov so zrakovým postihnutím. Správne štruktúrovaný obsah umožňuje čítačkám obrazovky efektívne navigovať. [26]
- **Ovládanie klávesnicou:** Aplikácia musí byť plne funkčná pomocou klávesnice (tab navigácia, enter na výber). Focus stavy musia byť vizuálne rozlíšiteľné, aby používatelia vedeli, ktorý prvok je aktívny. [24]
- **Spätná väzba a odozva:** Poskytovanie včasnej spätnej väzby používateľom zlepšuje ich skúsenosť s aplikáciou. Používanie animácií a prechodov na potvrdenie akcií používateľa, zobrazenie indikátorov načítavania počas dlhších procesov a jasné zobrazenie chybových hlášok s návodmi na riešenie problémov sú kľúčové prvky prívetivého designu. [22]
- **Optimalizácia pre dotykové interakcie:** Vzhľadom na to, že väčšina mobilných aplikácií je ovládaná dotykom, je dôležité navrhnuť dostatočne veľké dotykové ciele, ktoré zohľadňujú rôzne veľkosti prstov a minimalizujú náhodné ťuknutia. Medzi prvkami by mal byť dostatočný priestor, aby používatelia mohli presne a pohodlne klikáť. [22]
- **Plynulý onboarding:** Proces onboardingu nastavuje tón pre používateľskú skúsenosť s aplikáciou. Efektívny onboarding by mal jasne komunikovať hodnotu a základné funkcie aplikácie, postupne predstavovať nové funkcie, aby sa predišlo zahlteniu používateľov, a personalizovať skúsenosť na základe preferencií používateľa. [22]

Pre zdravotnícku aplikáciu je kľúčové zaistiť, že používateľské rozhranie nevyžaduje rozsiahle školenie alebo technické znalosti. Jednoduchosť použitia priamo prispieva k efektívnosti personálu a spokojnosti pacientov, čo je obzvlášť dôležité v oblasti starostlivosti o predčasne narodené deti a podpory laktácie. [22]

3.3 Príklady dobrej praxe v designe aplikácií pre zdravotné účely

Pri designe zdravotníckych aplikácií možno čerpať inšpiráciu z úspešných komerčných aplikácií, pričom je potrebné zohľadniť špecifické potreby zdravotníckeho prostredia. [22]

Na základe analýzy najlepších mobilných aplikácií môžeme identifikovať niekoľko princípov, ktoré sú aplikovateľné aj na zdravotnícke riešenia:

- **Personalizácia obsahu:** Podobne ako Netflix personalizuje odporúčania na základe preferencií používateľa, zdravotnícka aplikácia by mala prispôbiť obsah špecifickým potrebám každého používateľa. V kontexte podpory laktácie by to mohlo znamenať personalizované odporúčania a informácie založené na štádiu laktácie, zdravotnom stave dieťaťa a predchádzajúcich záznamoch.
- **Jasné označenie produktov a funkcií:** Aplikácie by mali používať jasné a popisné označenia pre všetky funkcie. Vyhybanie sa nejasným termínom a využívanie techník ako Card Sorting na pochopenie mentálnych modelov používateľov môže výrazne zlepšiť používateľskú skúsenosť.
- **Zohľadnenie vzorov skenovania:** Používatelia často skenujú text v F-pattern alebo Z-pattern. Umiestnenie dôležitých informácií a vizuálov tam, kde je najväčšia pravdepodobnosť, že budú videné, môže zlepšiť efektivitu aplikácie. V zdravotníckej aplikácii by kritické údaje ako množstvo mlieka, čas kŕmenia a základné zdravotné ukazovatele mali byť umiestnené v súlade s týmito vzormi.
- **Používateľsky prívetivé formuláre:** Čistý a jednoduchý design formulárov s minimálnym počtom požadovaných polí, jasnou validáciou a kontextuálnymi klávesnicami pre rôzne typy vstupov (numerické, e-mail, dátum) môže výrazne zjednodušiť zber údajov v nemocničnom prostredí.
- **Offline funkcionálnosť:** Podobne ako aplikácie Uber a Spotify optimalizujú svoje služby pre prostredie s nízkou kvalitou siete, zdravotnícke aplikácie by mali fungovať spoľahlivo aj v nemocničných priestoroch s potenciálne obmedzeným pripojením. Možnosť pracovať offline a synchronizovať údaje, keď je pripojenie dostupné, je kľúčová pre kontinuitu starostlivosti.

[22]

Pri implementácii týchto princípov do zdravotníckych aplikácií je dôležité pamätať na špecifické požiadavky a obmedzenia zdravotníckeho prostredia, ako je prísna ochrana osobných údajov, potreba presných medicínskych informácií a často stresové podmienky použí-

vania. Testovanie aplikácie priamo s koncovými používateľmi – zdravotníckym personálom a pacientmi – je nevyhnutné pre zabezpečenie, že design skutočne spĺňa ich potreby a zlepšuje pracovné postupy. [22]

4 BEZPEČNOST A OCHRANA OSOBNÝCH ÚDAJŮV

Vzhľadom na charakter zhromažďovaných údajov a cieľovú skupinu aplikácie (matky, často v citlivom životnom období), je zabezpečenie dát a ochrana súkromia jednou z najdôležitejších oblastí vývoja. V tejto kapitole sú preto popísané základné princípy ochrany osobných údajov v súlade s nariadením GDPR, ako aj konkrétne odporúčania a nástroje na zabezpečenie PWA.

4.1 Základy ochrany osobných údajov (GDPR)

Progresívne webové aplikácie (PWA) podliehajú rovnakým požiadavkám ochrany osobných údajov ako tradičné webové stránky. V kontexte GDPR je nevyhnutné, aby každá PWA jasne definovala účel a právny základ zberu osobných údajov, poskytla používateľom transparentné informácie o spracovaní dát a umožnila im spravovať svoje súhlasy. [27]

Medzi hlavné zásady GDPR, ktoré sa týkajú vývoja a prevádzky PWA, patria:

- **Transparentnosť a informovanie:** Používateľ musí byť jasne informovaný o tom, aké údaje sú zhromažďované, na aký účel a ako budú použité.
- **Súhlas a správa súhlasov:** Je potrebné implementovať mechanizmus na získanie a správu súhlasu používateľa so spracovaním údajov, najmä pri sledovaní alebo ukladaní preferencií.
- **Minimalizácia a bezpečné uchovávanie údajov:** Zbierať len nevyhnutné údaje a uchovávať ich bezpečne, vrátane šifrovania pri ukladaní aj prenose.
- **Práva používateľov:** Umožniť používateľovi prístup k jeho údajom, možnosť ich opravy, vymazania a obmedzenia spracovania (tzv. právo byť zabudnutý).
- **Správa tretích strán:** Pri integrácii služieb tretích strán (napr. analytika) je potrebné zabezpečiť, že aj tieto služby sú v súlade s GDPR.
- **Aktualizácie a audit:** Pravidelne aktualizovať zásady ochrany osobných údajov a vykonávať audity zberu a spracovania dát.

Dôraz sa kladie na *privacy by design*, teda začlenenie ochrany súkromia už do návrhu a vývoja aplikácie. Vývojári by mali pravidelne kontrolovať súlad s platnou legislatívou a prispôbovať aplikáciu novým požiadavkám. [27]

4.2 Ako zaistiť bezpečnosť dát v PWA

Bezpečnosť progresívnych webových aplikácií je postavená na implementácii viacerých vrstiev ochrany, ktoré chránia používateľské údaje pred zneužitím, únikom alebo útokmi:

- **Používanie HTTPS:** Všetky PWA musia byť prevádzkované výhradne cez zabezpečený protokol HTTPS, ktorý šifruje komunikáciu medzi klientom a serverom a umožňuje fungovanie kľúčových vlastností PWA, ako sú service workery a push notifikácie.
- **Zabezpečenie service workerov:** Service workery by mali mať obmedzený rozsah (scope) len na potrebné časti aplikácie. Je potrebné validovať a sanitizovať všetky vstupy, pravidelne aktualizovať kód a neukladať citlivé dáta do cache.
- **Content Security Policy (CSP):** CSP je účinný nástroj na prevenciu útokov typu XSS a vkladania škodlivého kódu. Definuje, z akých zdrojov môže aplikácia načítavať skripty, štýly či obrázky. CSP je potrebné pravidelne monitorovať a aktualizovať podľa vývoja aplikácie.
- **Šifrovanie dát a bezpečné ukladanie:** Citlivé údaje by mali byť šifrované nielen počas prenosu (HTTPS), ale aj pri ukladaní na strane klienta (napr. v IndexedDB alebo localStorage) pomocou Web Crypto API. Heslá a osobné údaje by nikdy nemali byť uložené v nešifrovanej podobe.
- **Silná autentifikácia a autorizácia:** Odporúča sa implementovať viacfaktorovú autentifikáciu, používať bezpečné heslové politiky a integrovať štandardy ako OAuth 2.0 alebo OpenID Connect. Prístupové práva je potrebné kontrolovať na klientovi aj serveri.
- **Pravidelné bezpečnostné audity a aktualizácie:** Pravidelné audity, penetračné testy a aktualizácie závislostí pomáhajú identifikovať a odstrániť zraniteľnosti. Automatizované nástroje (napr. OWASP ZAP, Snyk) a bezpečnostné kontroly v CI/CD pipeline sú odporúčané.

- **Edukácia používateľov a tímu:** Používateľom je vhodné poskytovať odporúčania na tvorbu silných hesiel a bezpečné správanie. Vývojársky tím by mal byť pravidelne školený v oblasti bezpečnosti.
- **Incident response plán:** Mať pripravený plán reakcie na bezpečnostné incidenty, ktorý definuje postupy pri úniku dát alebo útoku na aplikáciu.

[12, 27]

Tieto opatrenia tvoria základ bezpečnosti PWA a prispievajú k ochrane osobných údajov, dôvery používateľov a splneniu legislatívnych požiadaviek [12, 27].

II PRAKTICKÁ ČÁST

5 ANALÝZA POŽIADAVKOV

Pred vývojom aplikácie bola vykonaná dôkladná analýza potrieb cieľovej skupiny a preskúmanie existujúcich riešení na trhu. Táto časť práce sa zameriava na identifikáciu špecifických požiadaviek používateľov, určenie typických scenárov použitia a porovnanie s konkurenčnými aplikáciami.

5.1 Cieľová skupina

Primárnou cieľovou skupinou sú matky predčasne narodených detí. Jedná sa teda o veľmi úzko špecifikovanú skupinu používateľov. Matky čelia pomerne vysokému stresu a sú denne vystavované vysokým nárokom.

Tieto matky často potrebujú sledovať každé kojenie alebo odsávanie mlieka, aby zabezpečili dostatočný príjem výživy pre svoje dieťa/deti.

Z diskusií s nemocničným personálom vyplýva, že aktuálne nevyžaduje priamy prístup k jednotlivým údajom každej z matiek, ale do budúcnosti určite zvažujú zdieľanie laktálnych údajov aspoň z hľadiska tvorby štatistík.

5.1.1 Potrebné digitálne zručnosti

Aplikácia je určená pre bežného používateľa so základnými digitálnymi zručnosťami. Očakáva sa, že cieľová skupina dokáže bežne používať smartfón alebo tablet, inštalovať aplikácie a navigovať sa v jednoduchom používateľskom rozhraní. Zohľadňuje sa však aj to, že nie všetky používateľky môžu byť technicky zdatné, preto musí byť aplikácia intuitívna, s minimom krokov potrebných na vykonanie základných akcií.

5.1.2 Scenáre použitia

Aplikácia bude používaná v rôznych situáciách počas celého dňa, často v časovej tiesni alebo pri únave:

- V nemocnici, keď matka zaznamenáva množstvo odsávaného mlieka počas hospitalizácie dieťaťa.
- Doma, po prepustení, kde pokračuje v sledovaní dojčenia, aby mohla konzultovať stav s lekárom alebo poradkyňou.

- Počas noci, pri pravidelnom dojčení alebo odsávaní, kedy je dôležité rýchlo a jednoducho zaznamenať údaje bez potreby rozsiahlej interakcie.

5.1.3 Špecifické potreby

Cieľová skupina má niekoľko špecifických potrieb, ktoré musia byť zohľadnené pri návrhu aplikácie:

- Jednoduché a rýchle ovládanie, ideálne jednou rukou.
- Veľké a prehľadné tlačidlá, prispôsobené na použitie počas nosenia dieťaťa alebo v znížených svetelných podmienkach.
- Offline režim, ktorý umožní zadávať údaje aj bez pripojenia na internet, s neskoršou synchronizáciou.
- Notifikácie, ktoré pripomenú ďalšie kojenie alebo odsávanie podľa nastaveného plánu.
- Možnosť viacnásobných profilov, napríklad pre matky s viacerými deťmi (napr. dvojčatá).
- Zabezpečenie súkromia, keďže ide o citlivé osobné a zdravotné údaje.

5.2 Konkurečný prieskum

V rámci prieskumu trhu bolo prakticky otestovaných niekoľko mobilných aplikácií, ktoré ponúkajú funkcionality pre sledovanie laktácie a súvisiacich aktivít. Nasleduje stručný prehľad najrelevantnejších aplikácií.

5.2.1 Medela Family

Aplikácia od známeho výrobcu dojčiacich pomôcok vyžaduje registráciu a ponúka komplexné funkcie vrátane Knowledge Center – sekcie s blogmi a článkami pre rodičov. Má moderné a vizuálne atraktívne rozhranie, ktoré však miestami pôsobí menej intuitívne. Je určená skôr na celkovú správu starostlivosti o dieťa, nie výhradne na laktáciu.

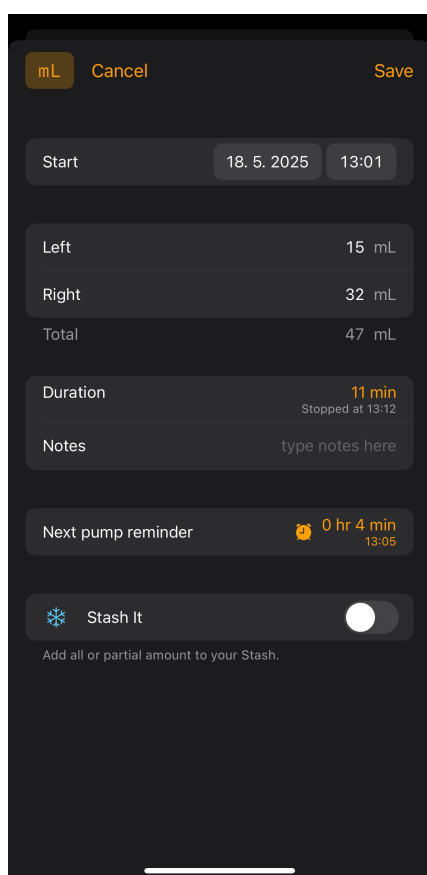
5.2.2 Pump Log

Táto aplikácia sa zameriava na zaznamenávanie pumpovania mlieka a vyniká jednoduchým, rýchlym rozhraním. Umožňuje zmenu jednotiek, prehľadné grafy, export/import údajov a

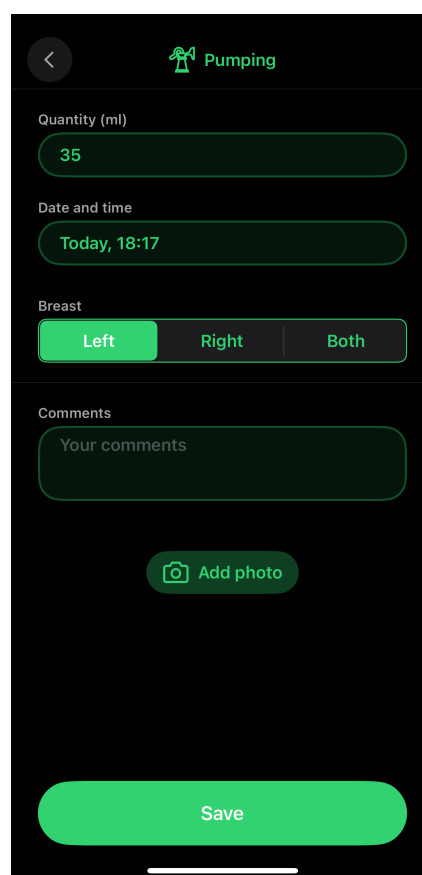
má aj podporu pre Apple Watch. Používateľské prostredie je však vizuálne menej atraktívne a zamerané výlučne na technickú funkciu.

5.2.3 Milk Stash

Aplikácia pre pokročilé sledovanie ukladania materského mlieka. Umožňuje evidenciu mlieka v rôznych nádobách (chladnička, mrazák), disponuje základnými štatistikami a nastavitelnými jednotkami. Registrácia je povinná. Úvodný onboarding je prívetivý, no recenzie používateľov chýbajú.

The screenshot shows a dark-themed mobile app interface for logging a pump session. At the top, there are buttons for 'mL', 'Cancel', and 'Save'. Below this is a 'Start' section with a date and time picker set to '18. 5. 2025' and '13:01'. A table follows with columns for 'Left', 'Right', and 'Total' amounts in mL, showing values of 15, 32, and 47 respectively. Below the table is a 'Duration' section showing '11 min' and 'Stopped at 13:12'. There is a 'Notes' field with the placeholder 'type notes here'. At the bottom, there is a 'Next pump reminder' section showing '0 hr 4 min' and '13:05', and a 'Stash It' toggle switch.

Obrázek 5.1 Ukážka aplikácie Pump Log
- pridanie záznamu

The screenshot shows a dark-themed mobile app interface for logging a pump session. At the top, there is a back arrow and a 'Pumping' header with a pump icon. Below this is a 'Quantity (ml)' section with a value of '35'. A 'Date and time' section shows 'Today, 18:17'. Below this is a 'Breast' section with three buttons: 'Left', 'Right', and 'Both'. There is a 'Comments' section with a text input field and the placeholder 'Your comments'. At the bottom, there is an 'Add photo' button with a camera icon and a large 'Save' button.

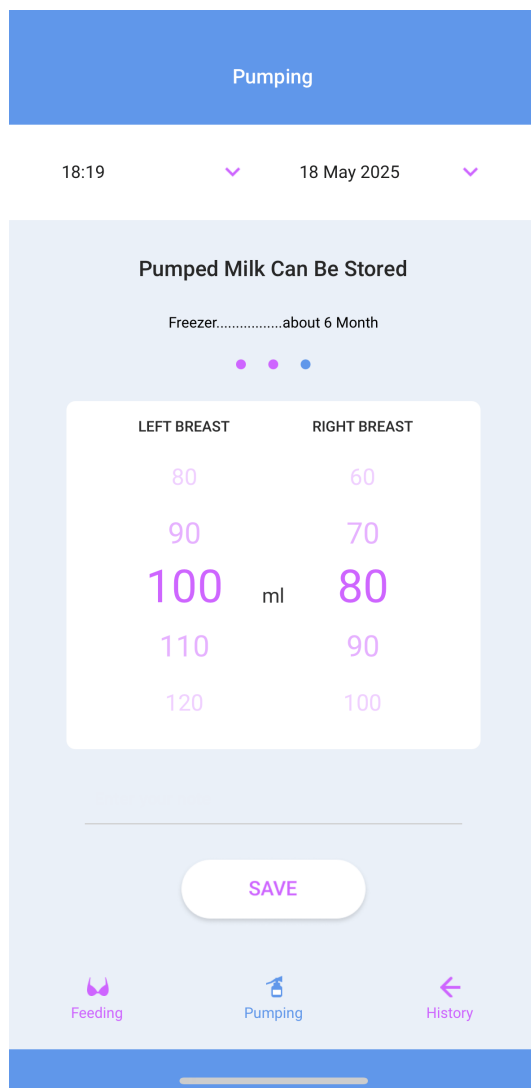
Obrázek 5.2 Ukážka aplikácie My Baby
- pridanie záznamu

5.2.4 My Baby

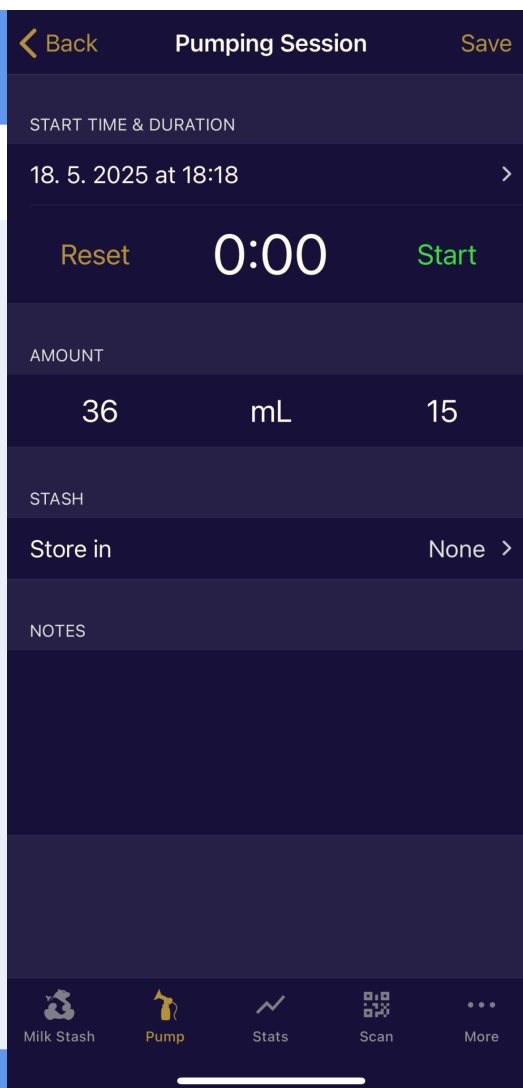
Komplexná aplikácia na správu starostlivosti o dieťa. Záznam pumpovania je jednoduchý, no chýba možnosť zadať konkrétne prsník a množstvo mlieka, čo je pre niektoré matky dôležité. Používateľské hodnotenia sú veľmi pozitívne (4.9/5 z 1300 hodnotení) a aplikácia neobsahuje reklamy ani skryté poplatky.

5.2.5 Breastfeeding

Jedna z najjednoduchších aplikácií, nevyžaduje registráciu a umožňuje rýchly záznam dojčenia aj pumpovania. Obsahuje však veľké množstvo reklamy a má funkčné nedostatky – napríklad stopky sa po minimalizovaní aplikácie zastavia, čo môže viesť k nesprávnym záznamom. Používateľské rozhranie je veľmi základné.



Obrázek 5.3 Ukážka aplikácie Breastfeeding - pridanie záznamu



Obrázek 5.4 Ukážka aplikácie Milk Stash - pridanie záznamu

5.2.6 Silné stránky konkurencie

- Viaceré aplikácie ponúkajú prehľadné štatistiky a grafy, možnosť exportu údajov a niektoré aj napojenie na nositeľnú techniku (napr. Apple Watch).
- Niektoré aplikácie sa vyznačujú moderným a čistým dizajnom (napr. Medela Family).

- Aplikácie ako My Baby alebo Pump Log umožňujú jednoduché pridávanie záznamov s možnosťou výberu jednotiek.

5.2.7 Nedostatky zistené počas testovania

- Povinná registrácia vo viacerých prípadoch môže pôsobiť ako bariéra pre rýchle použitie.
- Reklamy a zbytočné funkcie znižujú používateľský komfort, najmä v jednoduchších aplikáciách.
- Nedostatočná presnosť záznamov (napr. absencia informácie o prsníku) je častým problémom aj v inak prepracovaných aplikáciách.
- Chýbajúci offline režim znižuje využiteľnosť v nemocničnom prostredí bez internetu.
- Niektoré aplikácie nie sú lokalizované do češtiny alebo slovenčiny, čo sťažuje použitie technicky menej zdatným matkám.

Analýza konkurencie ukazuje, že na trhu existuje viacero riešení pre zaznamenávanie laktácie, no žiadne z nich nie je navrhnuté s dôrazom na potreby matiek predčasne narodených detí v kontexte českého zdravotníctva. Navrhovaná aplikácia sa zameriava práve na túto úzku skupinu používateľov, ponúkne zjednodušené a cielené prostredie pre zaznamenávanie kojenia a odsávania, offline režim. Jej cieľom je poskytnúť používateľsky prívetivé riešenie, ktoré zjednoduší starostlivosť o dieťa bez zbytočného zahltenia funkciami.

5.3 Funkčné požiadavky

Funkčné požiadavky popisujú konkrétne funkcie a správanie systému, ktoré musí aplikácia poskytovať:

- **Registrácia a prihlásenie používateľa**

Umožniť používateľovi vytvoriť si účet pomocou emailovej adresy a hesla, vrátane možnosti obnovenia zabudnutého hesla.

- **Overenie emailovej adresy**

Po registrácii je používateľ vyzvaný na potvrdenie emailovej adresy. K dispozícii je aj možnosť opätovného zaslania overovacieho emailu.

- **Personalizácia používateľského profilu**

Používateľ môže nastaviť:

- meno dieťaťa a jeho fotografiu,
- gestačný týždeň pri narodení,
- informáciu o viacerčatách a ich počet,
- informácie o predchádzajúcich pôrodoch, predchádzajúcom dojčení a jeho trvaní (v mesiacoch).

- **Validácia vstupov**

Všetky vstupné formuláre budú validované na strane klienta aj servera. Validácia zahŕňa kontrolu požadovaných polí, formátov emailových adries, číselných hodnôt (napr. množstvo mlieka) a logických obmedzení (napr. kladné hodnoty, rozsahy časov).

- **Domovská obrazovka (autentifikovaný používateľ)**

Obsahuje hlavné tlačidlo na pridanie nového záznamu, vrchné tlačidlo Pomoc, spodnú navigáciu (Domov, Prehľad, Upozornenia, Profil), meno a fotografiu dieťaťa a zoznam posledných piatich záznamov. Používateľka si môže pomocou gesta potiahnutia zoznamu zdola nahor zobrazit' celú históriu záznamov.

- **Pridanie nového záznamu**

Formulár na zaznamenanie údajov o odsávaní:

- čas začiatku odsávania,
- trvanie odsávania,
- množstvo mlieka z ľavého a pravého prsníka.

- **Úprava a odstránenie záznamov**

Každý vytvorený záznam možno upraviť alebo odstrániť.

- **Upozornenia a pripomienky**

Používateľ si môže nastaviť upozornenia v intervaloch po 10 minútach. Tieto slúžia ako pripomienky na odsávanie a sú realizované ako lokálne notifikácie priamo v zariadení.

- **Profilová stránka**

Obsahuje výpis informácií zadaných používateľom, možnosť ich úpravy, odhlásenie a možnosť odstrániť účet.

- **Prehľadové grafy**

Zobrazenie pokroku formou cirkulárnych grafov, ktoré ukazujú, ako sa používateľ približuje k dennému cieľu (ten sa prispôsobuje podľa počtu detí).

- **Obrazovka Pomoc**

Obsahuje užitočné odkazy, kontakty a čísla pomoci pre matky.

- **Motivačné hlášky**

Po dosiahnutí určitých milníkov v dennom objeme odsatého mlieka sa zobrazujú motivačné hlášky v modálnom okne.

5.4 Nefunkčné požiadavky

Nefunkčné požiadavky popisujú kvalitatívne vlastnosti systému a obmedzenia, ktoré musí aplikácia spĺňať:

- **PWA (Progressive Web Application)**

Aplikácia musí byť implementovaná ako progresívna webová aplikácia, teda musí podporovať offline režim, inštaláciu na mobilné zariadenia a rýchlu odozvu.

- **Mobile-first dizajn**

Používateľské rozhranie musí byť navrhnuté „mobile-first“, pretože cieľovou skupinou sú matky používajúce predovšetkým mobilné telefóny. Dizajn bude optimalizovaný pre menšie obrazovky a dotykové ovládanie.

- **Dostupnosť a jednoduchosť použitia**

Rozhranie musí byť intuitívne, prístupné a navrhnuté s ohľadom na mamičky v popôrodnom období, vrátane možnosti ovládania jednou rukou. Minimalizácia zložitých interakcií a pohybov v aplikácii (swipy, scrollovanie) bude prioritou.

- **Bezpečnosť a ochrana údajov**

Aplikácia musí zabezpečiť ochranu osobných údajov podľa nariadení GDPR, vrátane bezpečnej autentifikácie a šifrovania údajov.

- **Výkonnosť**

Aplikácia musí rýchlo reagovať na akcie používateľ a a minimalizovať dobu načítania stránok.

- **Rozšíriteľnosť systému**

Architektúra systému má byť navrhnutá modulárne, aby bolo jednoduché pridávať nové funkcie, ako je sledovanie dojčenia, váha dieťaťa alebo nálady matky, bez nutnosti zásahu do základného jadra aplikácie.

- **Prístupnosť (Accessibility)**

Aplikácia má dodržiavať princípy prístupnosti podľa WCAG (Web Content Accessibility Guidelines), ako sú dostatočný kontrast textu, čitateľné písmo, navigácia pomocou klávesnice a alternatívne texty k obrázkom.

- **Škálovateľnosť**

Backend a databáza musia byť pripravené zvládnuť vyšší počet používateľov bez straty výkonu.

- **Dostupnosť 24/7**

Aplikácia musí byť dostupná pre používateľov nepretržite.

- **Kompatibilita s modernými prehliadačmi**

Aplikácia musí bezproblémovo fungovať na najbežnejších prehliadačoch ako Chrome, Safari, Firefox a Edge.

5.5 Výsledok analýzy

Na základe vykonanej analýzy možno konštatovať, že dostupné aplikácie na trhu majú často špecifické zameranie (napr. evidencia pumpovania, správa zásob mlieka alebo komplexná starostlivosť o dieťa). Väčšina z nich však trpí nedostatkami v oblasti používateľskej prívetivosti, absenciou lokalizácie, reklám či slabšej podpore pre viacnásobných používateľov. Tieto zistenia boli zohľadnené pri návrhu vlastnej aplikácie, ktorá sa snaží spojiť funkčnosť s jednoduchým a prístupným používateľským rozhraním.

6 NÁVRH APLIKÁCIE

Za samotným návrhom stojí viac než len samotný vývoj. Je potreba premyslieť všetko od toho kde bude aplikácia nasadená, až po jednotlivé vizuálne prvky. Vzhľadom na zameranie sa na matky po pôrode, je treba klásť dôraz na jednoduchosť, dostupnosť a spoľahlivosť riešenia.

6.1 Architektúra aplikácie

Aplikácia je navrhnutá ako progresívna webová aplikácia (PWA) práve z dôvodu jednoduchosti. Či už je to z pohľadu nasadenia aplikácie do produkčného prostredia s takmer nulovými finančnými nákladmi, alebo z pohľadu jednoznačnosti pre používateľku. Na rozdiel od natívnych aplikácií, ktoré si vyžadujú publikáciu v oficiálnych obchodoch (napr. Google Play alebo App Store), PWA môže byť okamžite dostupná prostredníctvom bežného webového prehliadača. Používateľka si aplikáciu nemusí sťahovať, čím sa znižuje bariéra prvého použitia. Zároveň si ju však môže "pripnúť" na domovskú obrazovku ako bežnú aplikáciu a používať ju aj v offline režime – čo môže byť výhodné na miestach s nestabilným internetovým pripojením, napríklad v nemocnici. Technologická voľba progresívnej webovej aplikácie pred natívnym riešením teda prirodzene podporuje cieľ aplikácie – pomáhať čo najširšiemu spektru používateľiek bez technických bariér a s minimálnymi nákladmi.

Celý projekt je rozdelený do dvoch samostatných častí - frontend a backend, každá z nich má samostatný repozitár a využíva iný súbor technológií.

6.1.1 Frontend

Z počiatku sa rozhodovalo medzi vývojom typu "from-scratch" použitím len knižnice React a písanie si vlastných komponentov. Avšak po diskusiách s odborníkmi v tejto oblasti a zváženia časovej náročnosti bolo toto rozhodnutie prehodnotené. Použitie Ionic frameworku spolu s Capacitorom prinieslo viaceré výhody. Ionic umožňuje vývoj multiplatformových aplikácií na báze webových technológií (HTML, CSS, JavaScript), pričom zachováva natívny vzhľad a správanie sa aplikácie. To je ideálne pre progresívnu webovú aplikáciu, ktorú možno v prípade potreby rozšíriť aj na natívne platformy (Android/iOS), bez nutnosti zásadných úprav kódu. Capacitor zohráva dôležitú úlohu pri premostení medzi webovým a natívnym svetom – umožňuje prístup k natívnym API zariadenia, ako sú notifikácie, filesystem alebo senzory,

a zároveň poskytuje jednoduchý spôsob nasadenia aplikácie aj do obchodov s aplikáciami, ak by sa neskôr vývoj týmto smerom rozšíril.

Medzi hlavné použité technológie patria:

- **React a Ionic React** - základná knižnica pre tvorbu užívateľského rozhrania. Ionic framework poskytuje množstvo preddefinovaných komponentov, ktoré sa prispôbujú rôznym platformám (iOS, Android, web).
- **Capacitor** – nástroj vyvíjaný tímom Ionic, ktorý umožňuje nasadiť webovú aplikáciu ako natívnu aplikáciu pre iOS a Android. Umožňuje prístup k natívnym API zariadenia (napr. kamera, upozornenia, vibrácie) prostredníctvom JavaScriptu. V projekte bol použitý napr. na integráciu lokálnych notifikácií
- **TailwindCSS** - utility-first CSS framework, ktorý umožňuje rýchle a konzistentné štylovanie komponentov pomocou priamo vkladateľných CSS tried. Bol využívaný predovšetkým pri návrhu vizuálne prispôbených komponentov.
- **Typescript** - nadmnožina jazyka JavaScript, ktorá prináša statické typovanie. Vďaka nemu je možné odhaliť chyby už počas vývoja a zároveň zlepšiť čitateľnosť, robustnosť a udržiavateľnosť kódu.
- **React Hook Form a zod** - nástroje na správu formulárov a validáciu vstupov. React Hook Form zabezpečuje efektívnu manipuláciu s formulárovými dátami, zatiaľ čo Zod umožňuje deklaratívne definovať validačné schémy pre jednotlivé polia (napr. pri zadávaní množstva mlieka).
- **Zustand** - stavový manažér pre React, umožňuje jednoduché zdieľanie a správu aplikácieho stavu naprieč komponentmi. V projekte bol použitý na ukladanie používateľských nastavení a iných zdieľaných údajov.
- **Axios a React Query** - Axios zjednodušuje tvorbu HTTP požiadaviek na server. React Query slúži na efektívnu správu asynchrónnych dát (napr. volania na backend), ich cacheovanie a refetching. Vhodné na synchronizáciu údajov so serverom.

6.1.2 Backend

Celý backend je postavený okolo PHP frameworku Laravel. Voľba tohto frameworku nebola prvotným plánom, avšak vychádzala z praktických obmedzení a požiadaviek. Nakoľko univerzitný server, na ktorom aplikácia pobeží je primárne nakonfigurovaný pre jazyk PHP a databázový systém MySQL.

Z tohto dôvodu boli JavaScriptové backendové technológie ako Node.js či Express.js vylúčené už v počiatočnej fáze návrhu. Laravel však predstavuje moderný a robustný framework s bohatým ekosystémom, vynikajúcou dokumentáciou a množstvom vstavaných funkcií (napr. správa databázových migrácií, autentifikácia, validácia, REST API), ktoré vývoj značne urýchlili a zjednodušili.

Použitie Laravela zároveň umožnilo vytvoriť čistú a dobre štruktúrovanú aplikačnú architektúru, ktorá je jednoducho rozšriteľná do budúcnosti.

6.2 Návrh databázy

Databázová štruktúra bola navrhnutá s dôrazom na jednoduchosť, rozšriteľnosť a bezpečnosť. Primárnym cieľom návrhu bolo zabezpečiť efektívne ukladanie informácií o laktálnych záznamoch a používateľských nastaveniach.

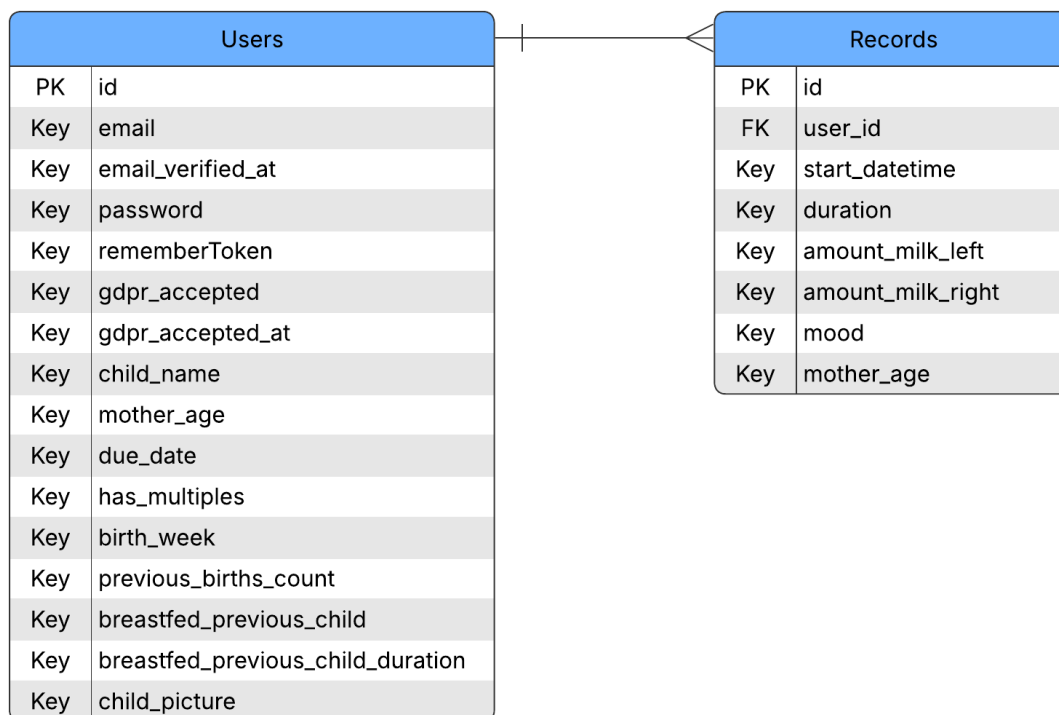
Ako databázový systém bola použitá MySQL, ktorá je prednastavená na univerzitnom serveri a poskytuje dostatočný výkon aj pre prípadné budúce rozšírenie systému.

V databáze sa nachádzajú dve základné entity (tabuľky):

- **users** – uchováva informácie o používateľkách aplikácie, ako je meno, e-mail (služiaci ako prihlasovacie meno), zašifrované heslo a dátum registrácie.
- **records** - tabuľka pre zaznamenávanie údajov o jednotlivých dávkach odsávaného mlieka. Obsahuje množstvo (v mililitroch), dátum a čas záznamu a ID používateľky.

6.3 Bezpečnostné opatrenia

Hoci aplikácia nepracuje so zdravotnými záznamami v pravom zmysle slova, uchováva citlivé údaje týkajúce sa matky a jej laktácie. Z tohto dôvodu bolo potrebné implementovať aspoň základné bezpečnostné mechanizmy, ktoré zabránia neoprávnenému prístupu k týmto údajom.



Obrázek 6.1 Základná databázová štruktúra aplikácie s entitami *users* a *records*.

(zdroj: vlastný)

Základným bezpečnostným opatrením je autentifikácia používateľky. Pri registrácii si používateľka vytvára konto s emailovou adresou a heslom, pričom jej účet je následne overený prostredníctvom potvrdzovacieho emailu. Tým sa minimalizuje riziko vytvárania falošných alebo anonymných účtov.

Na backendovej strane sú heslá bezpečne hašované pomocou moderného kryptografického algoritmu (bcrypt), čo zaručuje, že ani v prípade úniku databázy nie je možné spätne získať pôvodné heslá. Komunikácia medzi frontendovou aplikáciou a serverom prebieha výhradne cez šifrované HTTPS spojenie.

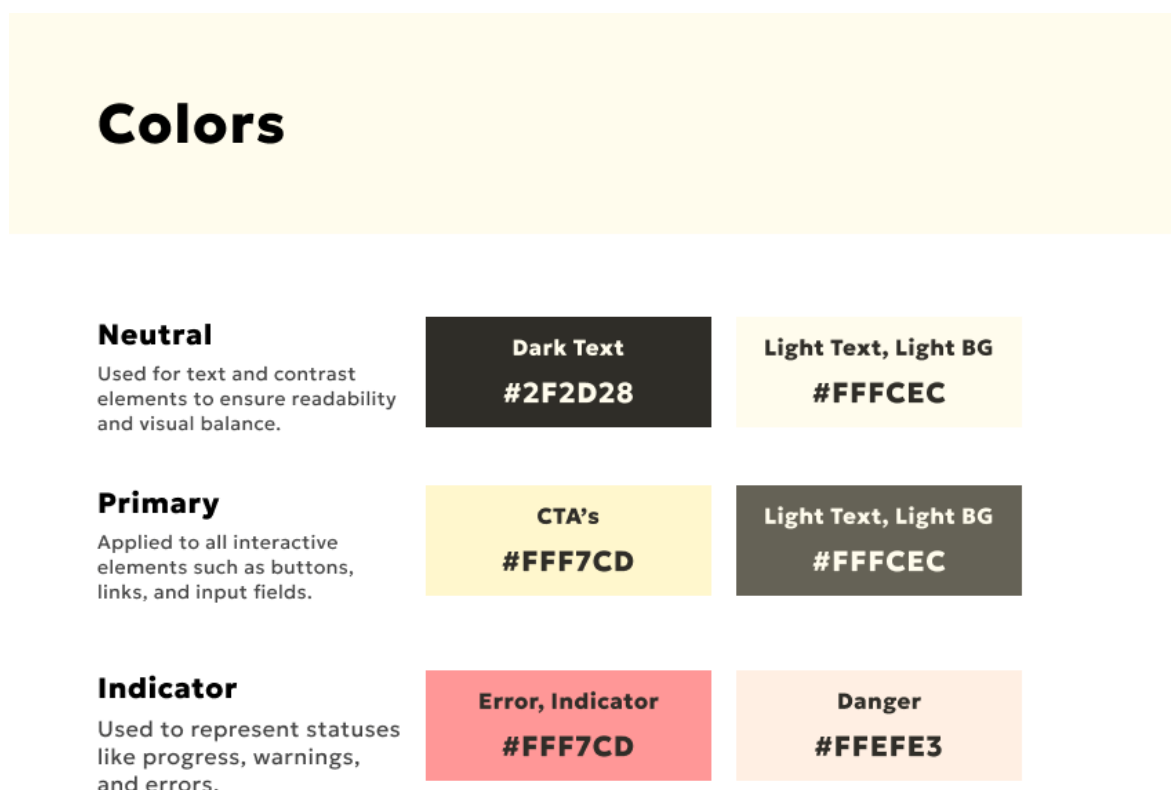
Vzhľadom na charakter aplikácie zatiaľ nie je implementovaná podpora pre viacúrovňové oprávnenia, keďže každý používateľ pracuje výhradne so svojimi dátami. V prípade budúceho rozšírenia aplikácie (napr. možnosť prístupu zdravotníckeho personálu) bude potrebné doplniť autorizáciu a správu prístupových práv.

6.4 Návrh používateľského rozhrania

Pred samotnou implementáciou frontendovej časti bola vytvorená podrobná návrhová dokumentácia v nástroji Figma. Cieľom bolo nielen zefektívniť vývoj, ale aj zabezpečiť konzistentnosť a používateľskú prívetivosť celého rozhrania – čo je obzvlášť dôležité vzhľadom na cieľovú skupinu, ktorou sú matky po pôrode, často v stresovej alebo vyčerpanej situácii.

Na začiatku návrhu bol vytvorený jednoduchý design systém, ktorý definoval základné stavebné prvky používateľského rozhrania:

- **Farebná schéma** – boli zvolené jemné béžové odtiene pre pozadia, ktoré vytvárajú teplú a prívetivú atmosféru. Texty a ikony sú zobrazené v tmavej šedej farbe kvôli vysokému kontrastu a dobrej čitateľnosti. Na vyznačenie chybových hlášok a upozornení bola použitá sýta červeno-rúžová farba, ktorá okamžite pritiahne pozornosť bez pôsobenia príliš rušivo.



Obrázek 6.2 Farebná schéma používateľského rozhrania.

(zdroj: vlastný)

- **Typografia** – bol zvolený moderný a prehľadný bezpätkový font **Geologica**, ktorý kombinuje čitateľnosť s jemným vizuálnym štýlom. Vďaka rôznym hrúbkam písma umožňuje dobre odlíšiť nadpisy, podnadpisy a bežný text pri zachovaní jednotného vizuálneho dojmu.

Typography

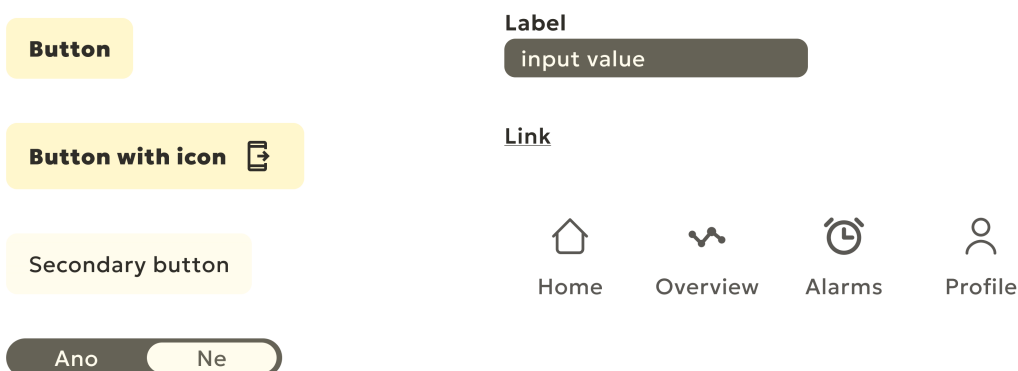
H1 Headline	32px 900	body	16px 400
H2 Headline	28px 900	body semi	16px 500
H3 Headline	24px 800	body bold	16px 600
H4 Headline	20px 800		
H5 Headline	18px 900		

Obrázek 6.3 Typografická hierarchia aplikácie definovaná podľa veľkosti písma a hrúbky rezu pre nadpisy (H1–H5) a bežné texty.

(zdroj: vlastný)

- **Komponenty** – boli navrhnuté opakovane použiteľné komponenty ako tlačidlá, textové polia, karty, notifikácie či navigačný panel. Tieto komponenty boli navrhnuté s ohľadom na použiteľnosť na malých obrazovkách a ovládanie jednou rukou.

UI Components

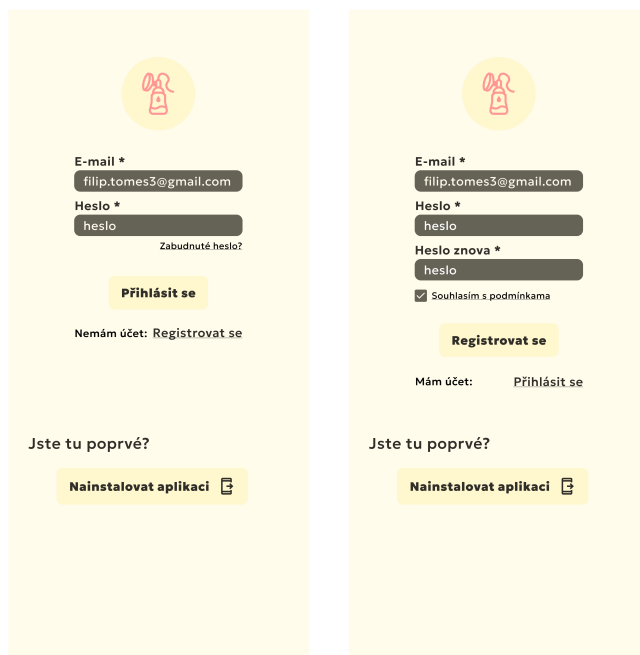


Obrázek 6.4 Základné UI komponenty aplikácie navrhnuté pre konzistentnú interakciu používateľ a s rozhraním.

(zdroj: vlastný)

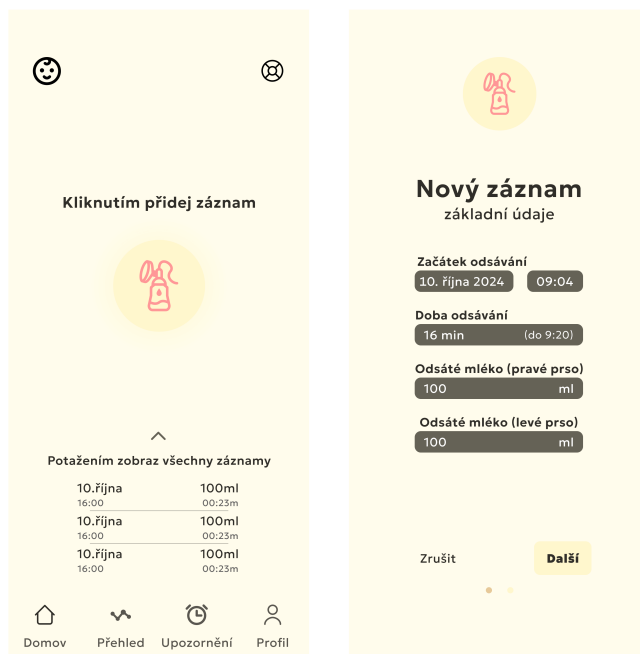
Po definovaní vizuálneho štýlu bol v rámci Figma vytvorený kompletný interakčný tok aplikácie (user flow). Ten pokrýva všetky hlavné scenáre použitia aplikácie – od registrácie a prihlasovania, cez zaznamenávanie dávok mlieka, nastavovanie upozornení, až po prezeraťie histórie alebo úprav nastavení.

Ešte pred začiatkom samotného vývoja bol celý návrh prezentovaný vo forme klikateľného prototypu v nástroji Figma zástupcom neonatologického oddelenia KNTB. Konzultácia prebehla formou krátkeho testovania, diskusie a spätnej väzby, ktorá bola následne zapracovaná do návrhu. Tento postup pomohol overiť vhodnosť rozhrania pre cieľovú skupinu a upraviť niektoré aspekty ovládania, aby boli intuitívne aj pre menej technicky zdatné používatel'ky. Design systém tak slúžil ako referencia aj pri úpravách a iteráciách počas samotného vývoja.



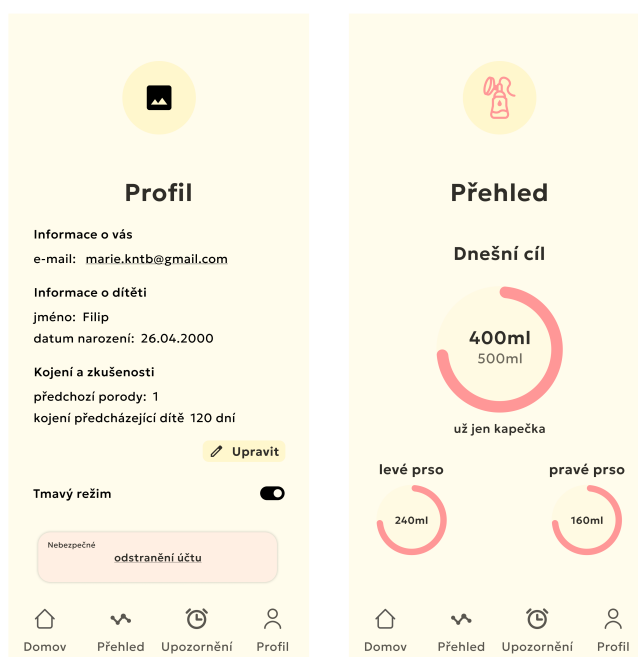
Obrázek 6.5 Obrazovky přihlášení a registrace – prvý kontakt používateľky s aplikáciou. Dôraz na jednoduchosť a čitateľnosť.

(zdroj: vlastný)



Obrázek 6.6 Hlavná obrazovka a obrazovka pre zadanie nového záznamu o dávke mlieka – navrhnuté pre rýchlu a intuitívnu interakciu.

(zdroj: vlastný)



Obrázek 6.7 Obrazovka profilu a sekcia s prehľadom – umožňujú správu osobných údajov a kontrolu nad laktáciou v čase.

(zdroj: vlastný)

7 IMPLEMENTÁCIA

Obsahom tejto kapitoly je priblížiť technickú realizáciu aplikácie – od počiatočných výziev pri vývoji, cez návrh backendovej architektúry, až po vývoj a testovanie frontendovej časti. Opísané sú aj použité nástroje, knižnice a princípy, ktoré boli zvolené s dôrazom na jednoduchosť, udržateľnosť a multiplatformovosť riešenia.

Zároveň je popísaný spôsob nasadenia počas vývoja. Frontendová časť aplikácie bola dočasne nasadená na platforme **Vercel**, ktorá poskytuje bezplatný hosting, automatické nasadzovanie z GitHub repozitára, SSL certifikát a podporu pre PWA. Toto riešenie umožnilo jednoduché zdieľanie aplikácie s testujúcimi používateľkami a nemocničným personálom. Do budúcnosti sa uvažuje o migrácii na vlastnú doménu a stabilnú produkčnú infraštruktúru.

7.1 Počiatočné technické výzvy

Po definovaní požiadaviek na aplikáciu bolo potrebné otestovať niektoré kľúčové funkcionality ešte pred samotným vývojom. Špeciálnu pozornosť si vyžadovali notifikácie, ktoré predstavujú jednu z hlavných funkcií aplikácie.

7.1.1 Notifikácie

Notifikácie predstavujú zásadný prvok aplikácie – umožňujú používateľke nastaviť si pripomienky na odsávanie materského mlieka v časových intervaloch po desiatich minútach. Z pohľadu implementácie sa rozlišujú dva hlavné typy notifikácií:

- **Push notifikácie** – sú odosielané z externého servera a zobrazované na zariadení používateľky. Vyžadujú internetové pripojenie, registráciu zariadenia a často aj využitie externých služieb ako Firebase Cloud Messaging (FCM) alebo Apple Push Notification Service (APNs). V rámci tejto aplikácie sa push notifikácie nepoužívajú.
- **Lokálne notifikácie** – sú plánované a zobrazované priamo v zariadení, bez nutnosti trvalého internetového pripojenia. Používajú sa na jednoduché časovače alebo pripomienky, čo ich robí ideálnymi pre účel tejto aplikácie.

K realizácii lokálnych notifikácií bol použitý plugin `@capacitor/local-notifications`, ktorý poskytuje natívne API pre Android aj iOS prostredníctvom Capacitoru. Počas testovania sa však prejavili viaceré technické obmedzenia:

- **PWA podpora v prehliadačoch** - výrazne obmedzená. Lokálne notifikácie fungujú spoľahlivo len pri inštalovanej PWA (t.j. pripnutej na domovskú obrazovku) a nie vo všetkých prehliadačoch.
- **iOS obmedzenia** – iOS platforma neposkytuje podporu pre lokálne notifikácie v rámci PWA v prehliadači Safari. Funkčnosť je výrazne limitovaná a pre plnú podporu by bola potrebná natívna aplikácia alebo publikovaná hybridná aplikácia cez App Store.
- **Android podpora** – Android umožňuje zobrazovať lokálne notifikácie aj v PWA, ale vyžaduje súhlas používateľky so zobrazením notifikácií a povolenie cez systémové nastavenia.

Na základe týchto zistení bolo rozhodnuté, že aplikácia bude využívať **lokálne notifikácie** ako doplnkovú funkcionality, a nie ako kritickú súčasť. Upozornenia budú fungovať spoľahlivo predovšetkým na zariadeniach s Androidom a pri inštalácii aplikácie ako PWA.

```
const ensureNotificationPermission = async () => {
  if ('Notification' in window && Notification.permission !== 'granted') {
    const permission = await Notification.requestPermission();
    if (permission !== 'granted') {
      present({
        message: ERROR_MESSAGES.NOTIFICATIONS_PERMISSION_DENIED,
        duration: 4000,
        position: 'bottom',
      });
      return false;
    }
  }
  return true;
};

const scheduleNotification = async () => {
  const hasPermission = await ensureNotificationPermission();
  if (!hasPermission) return;

  const notificationTime = new Date(
    Date.now() + notificationDelaySeconds * 1000,
  );
  setScheduledTime(notificationTime);

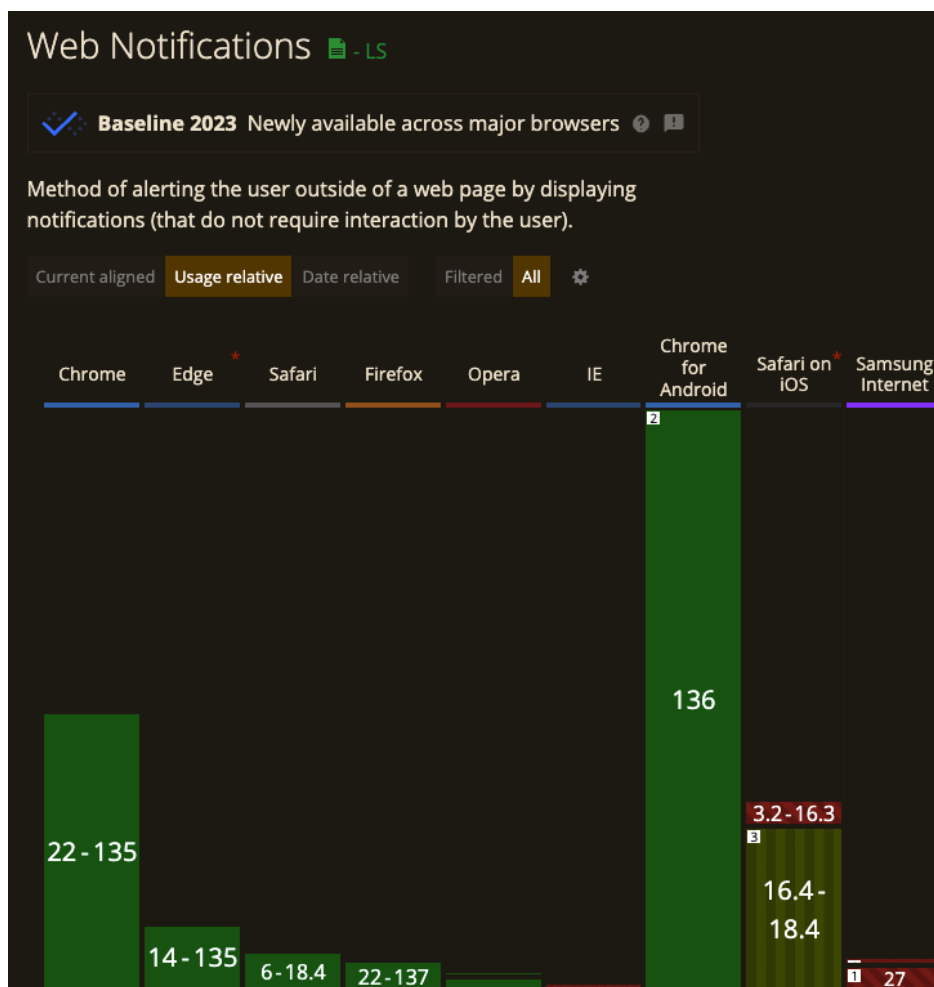
  if ('serviceWorker' in navigator) {
    showBrowserNotification(notificationDelaySeconds, setTimeoutId);
  } else if (LocalNotifications) {
    scheduleCapacitorNotification(
      notificationDelaySeconds,
      notificationTime,
      setTimeoutId,
    );
  }
};
```

Obrázek 7.1 Úryvok JavaScript kódu zobrazuje logiku pre zabezpečenie oprávnenia a naplánovanie lokálnej notifikácie.

(zdroj: vlastný)

Implementácia rozlišuje medzi webovým a mobilným prostredím a prispôsobuje spôsob doručovania notifikácií dostupným API. Zabezpečuje tiež používateľské oprávnenie a spätne informuje používateľku o probléme, v prípade, že nastane.

Z nasledujúceho grafu je zrejmé, že podpora je veľmi silná najmä na platformách Chrome, Firefox a Android. Obmedzená podpora je na Safari pre iOS, Samsung Internet a starších verziách Opera a IE.



Obrázek 7.2 Prehľad podpory webových notifikácií (Web Notifications API) v jednotlivých webových prehliadačoch a ich verziách. (zdroj: *caniuse.com*)

7.2 Backend

Na tvorbu backendovej časti aplikácie bol použitý framework **Laravel**, ktorý je známy svojou prehľadnou štruktúrou a rozsiahlymi možnosťami prispôsobenia. Vďaka kvalitnej dokumentácii bolo možné vytvoriť plnohodnotné API v relatívne krátkom čase.

7.2.1 Lokálne prostredie a inštalácia

Ako lokálny vývojový server bola použitá aplikácia XAMPP, ktorá poskytuje jednoduchý spôsob nasadenia MySQL databázy a PHP servera. Laravel bol nainštalovaný pomocou správcu balíčkov Composer:

1. Inštalácia Laravel frameworku pomocou `composer create-project laravel/laravel`.
2. Úprava konfiguračného súboru `.env` pre prepojenie s MySQL databázou cez XAMPP.

3. Vytvorenie novej databázy v phpMyAdmin.

7.2.2 Modely a databázové schéma

Databázová vrstva bola implementovaná pomocou Eloquent ORM:

- Modely boli generované príkazom `php artisan make:model NázovModelu -all`, ktorý vytvoril model, migráciu, kontrolér a ďalšie komponenty potrebné pre základné databázové operácie CRUD (Create, Read, Update, Delete).
- Bol použitý preddefinovaný model `User`, ku ktorému boli pridané nové vzťahy.
- Migrácie definovali štruktúru tabuliek pre záznamy o odsávaní, používateľské nastavenia a ostatné dáta.
- Na testovanie boli vytvorené factory triedy a seed skripty, ktoré umožnili vygenerovať fiktívne údaje pomocou príkazu `php artisan migrate:fresh --seed`.

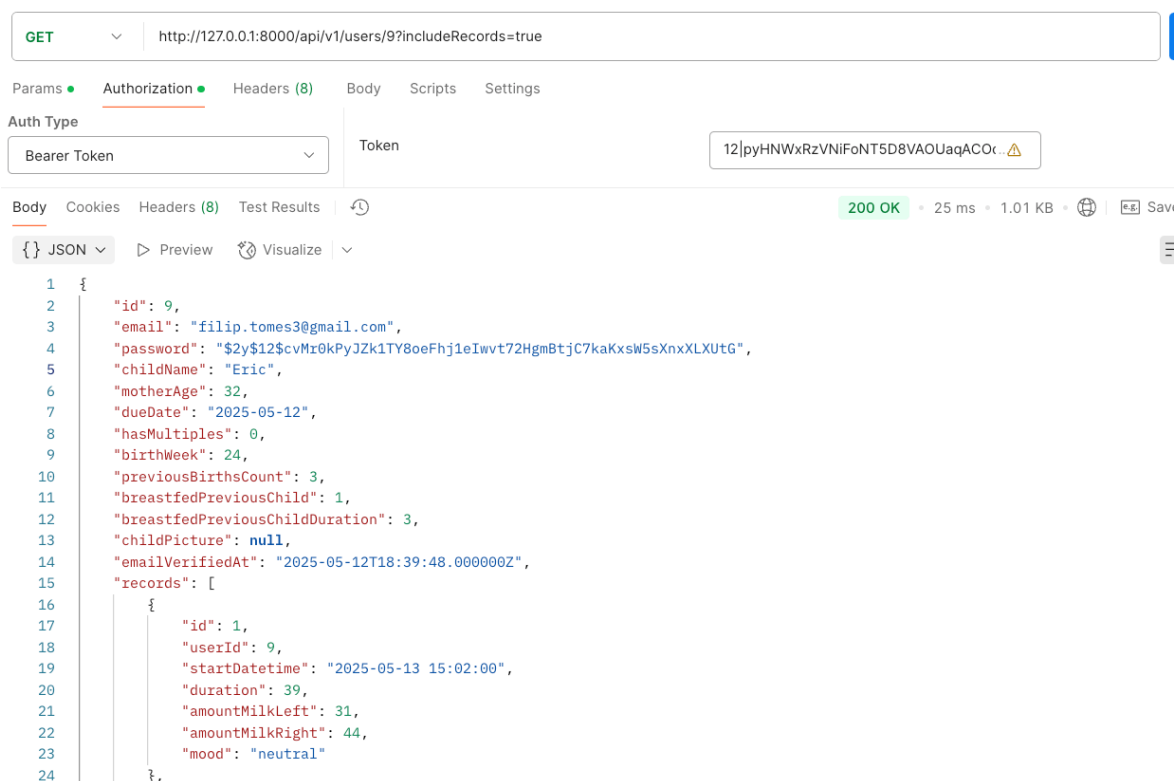
7.2.3 API a routovanie

Aplikácia poskytuje RESTful API pre komunikáciu s frontendovou časťou:

- API routy boli definované v súbore `routes/api.php`, vrátane GET, POST, PUT a DELETE požiadaviek.
- Na zabezpečenie konzistentného formátu odpovedí bol použitý príkaz `php artisan make:resource`, ktorý generuje výstup v JSON formáte. Pomocou `resource` tried je zároveň možné prispôbiť štruktúru výstupu, napríklad konvertovať názvy atribútov do `pascalCase`, čo je bežná konvencia pri JSON výstupe.
- Na zabezpečenie autentifikácie API bol použitý balík `Laravel Sanctum`, ktorý poskytuje jednoduché a bezpečné tokenové overenie používateľiek.

7.3 Frontend

Frontendová časť aplikácie bola postavená s využitím frameworku **Ionic React**, ktorý je vhodný pre vývoj multiplatformových PWA aplikácií s responzívnym dizajnom. Vývoj prebiehal podľa oficiálnej dokumentácie dostupnej na <https://ionicframework.com/docs/react/pwa>.



Obrázek 7.3 Ukážka JSON odpovede, znázorňujúca formátovanie dát.

(zdroj: vlastný)

Aplikácia bola v testovacej fáze nasadená prostredníctvom platformy Vercel, ktorá poskytuje bezplatný hosting pre frontendové aplikácie, automatické nasadzovanie z GitHub repozitára a integrovaný SSL certifikát. Vďaka jednoduchému nasadeniu a podpore PWA bola Vercel ideálnym riešením pre vývoj a testovanie. Do budúcnosti sa však uvažuje o migrácii na vlastný server alebo iné produkčné riešenie.

7.3.1 Nastavenie projektu

- Projekt bol inicializovaný cez webovú službu <https://ionic.io/pwa> Ionic PWA, ktorá vytvorila github repozitár so základným nastavením pre React a Ionic.
- Podľa dokumentácie bol pridaný VitePWA plugin na jednoduché nastavenie service workeru a celkového procesu konfigurácie PWA.
- Boli nakonfigurované nástroje ako eslint, prettier a tsconfig.json, ktoré zabezpečujú konzistentný kódový štýl a typovú bezpečnosť.
- Pri pokuse o integráciu najnovšej verzie tailwindcss v4 sa zistilo, že nie je kompa-

tabilná s React 19, ktorý ešte nie je podporovaný zo strany Ionic. Preto bola použitá stabilná verzia tailwindcss v3 a React 18.

- Do projektu boli pridané vlastné fonty a grafické assety, ktoré zodpovedajú dizajnu aplikácie.

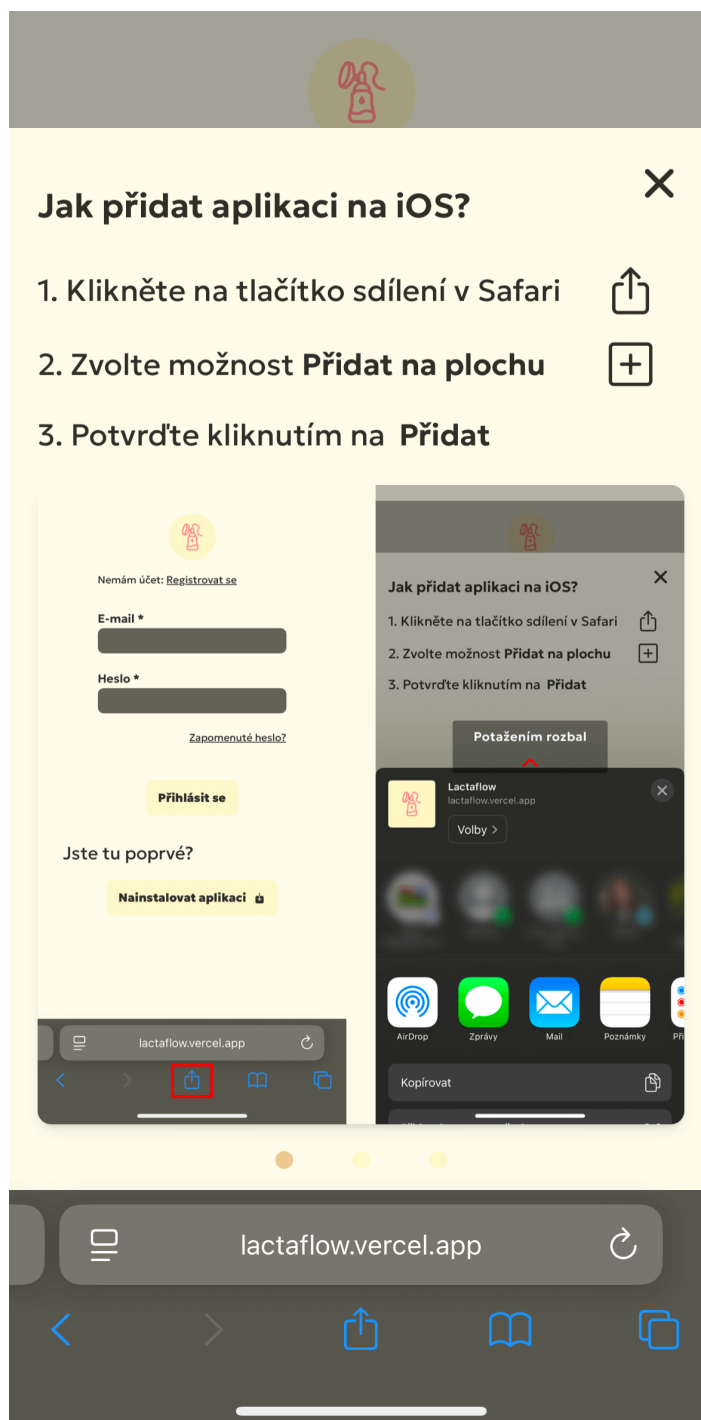
7.3.2 Prvotná pomoc pri inštalácii PWA

Vzhľadom na to, že aplikácia je distribuovaná ako progresívna webová aplikácia (PWA), je cieľom naviesť používateľku k tomu, aby si ju nainštalovala na domovskú obrazovku. Vďaka tomu získava aplikácia vlastnú ikonu, plnohodnotný fullscreen režim a môže byť spustená aj offline (v rozsahu dostupnej cache).

Táto funkcionálna bola vytvorená pomocou komponentu `HowToInstallModal`, ktorý je súčasťou layoutu prihlasovacej a registračnej obrazovky. Po kliknutí na tlačidlo „Nainštalovať aplikáciu“ sa na základe platformy vyvolá príslušná akcia:

- **Android (Chrome a iné)** – ak prehliadač podporuje `beforeinstallprompt`, zobrazí sa natívna výzva na inštaláciu aplikácie.
- **iOS (Safari)** – keďže iOS platforma túto výzvu nepodporuje, používateľke sa zobrazí manuálny návod formou modálneho okna s textom a ilustráciou (viď Obr. 7.4).

Cieľom je zvýšiť mieru inštalácií aplikácie a zároveň zabezpečiť, že funkcie ako notifikácie alebo offline režim budú fungovať korektne.



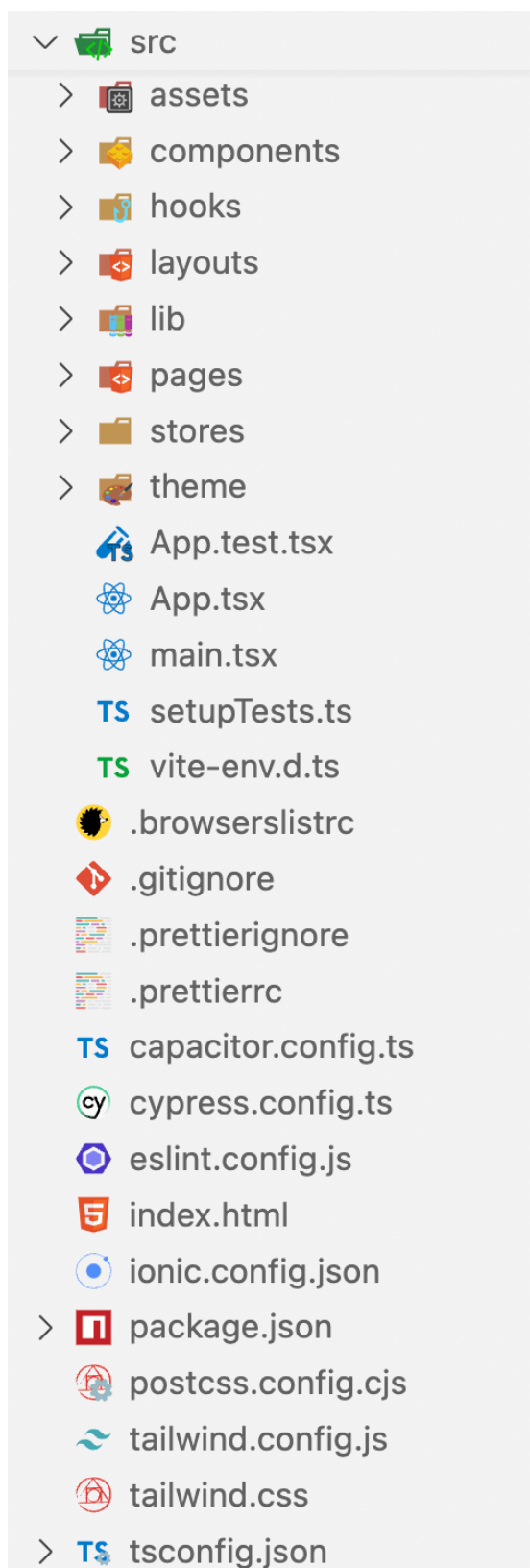
Obrázek 7.4 Ukážka modálneho okna so sprievodcom inštalácie aplikácie na domovskú obrazovku na zariadeniach so systémom iOS. Používateľka je vedená krok po kroku pomocou obrázka a textu.

(zdroj: vlastný)

7.3.3 Štruktúra komponentov a stránok

Na nasledujúcom obrázku je zobrazená priečinková štruktúra projektu, konkrétne zložky `src`. Pre vývojárov so skúsenosťami s vývojom webových aplikácií, najmä pri práci s menej striktné štruktúrovanými (non-opinionated) frameworkami, by mala byť celková organizácia kódu intuitívna a prehľadná.

Aplikácia využíva štruktúrovaný prístup k rozdeleniu kódu – boli vytvorené znovupoužiteľné UI komponenty (napr. tlačidlá, formulárové prvky) a zdieľané layouty. Každá hlavná obrazovka bola vytvorená ako samostatná Page komponenta. Navigácia medzi stránkami bola riešená pomocou `ionic-router` a komponenty `ion-tab-bar` pre spodné menu. `TabBar` komponenta je prispôbena tak, aby vyhovovala dizajnovým predlohám.



Obrázek 7.5 Priečinková štruktúra frontendovej časti aplikácie – komponenty, stránky, zdieľané rozhrania a konfigurácia.

(zdroj: vlastný)

```

App.tsx > ...
const App = () => {
  return (
    <QueryClientProvider client={queryClient}>
      <IonApp className="mx-auto max-w-[400px]">
        <IonReactRouter>
          <IonTabs>
            <IonRouterOutlet>
              <Route
                exact
                path={ROUTES.HOME}
                render={() => {
                  if (!isUserLogged) return <Login />;
                  if (!user?.emailVerifiedAt) return <VerifyMail />;
                  return <Home />;
                }}
              />
              <Route exact path={ROUTES.OVERVIEW} render={() => <Overview /> } />
              <Route
                exact
                path={ROUTES.NOTIFICATIONS}
                render={() => <Notifications /> }
              />
              <Route exact path={ROUTES.PROFILE} render={() => <Profile /> } />
              <Route exact path={ROUTES.AUTH.LOGIN} render={() => <Login /> } />
              <Route
                exact
                path={ROUTES.AUTH.REGISTER}
                render={() => <Register /> }
              />
            </IonRouterOutlet>
          </IonTabs>
        </IonReactRouter>
      </IonApp>
    </QueryClientProvider>
  );
}

```

Obrázek 7.6 Ukážka definície hlavných stránok a routovacích pravidiel pomocou komponentov IonRouterOutlet a Route.

(zdroj: vlastný)

```

export const TabBar = () => {
  const location = useLocation();

  return (
    <IonTabBar
      slot="bottom"
      style={{
        '--color': 'rgba(var(--ion-text-color-rgb), 0.5)',
        '--color-selected': 'var(--ion-text-color)',
      }}
    >
      {TAB_NAV_ITEMS.map((item) => (
        <IonTabButton
          key={item.key}
          tab={item.key}
          href={item.path}
          selected={location.pathname === item.path}
        >
          <IonIcon aria-hidden="true" icon={item.icon} />
          <IonLabel>{item.label}</IonLabel>
        </IonTabButton>
      ))}
    </IonTabBar>
  );
}

```

Obrázek 7.7 Komponenta TabBar, ktorá definuje spodnú navigáciu aplikácie. Na základe aktuálnej cesty (pathname) zvýrazňuje aktívnu položku a načítava príslušnú ikonu a názov z konštanty TAB_NAV_ITEMS.

(zdroj: vlastný)

7.3.4 Komunikácia s API a správa stavu

Pre komunikáciu s backendovým API bol použitý knižničný nástroj `axios`, pričom všetky requesty boli centralizované v súbore `api.ts`.

```
const api = axios.create({
  baseURL: API_BASE_URL,
  withCredentials: true,
  headers: {
    'Content-Type': 'application/json;charset=utf-8',
  },
});

api.interceptors.request.use(
  (config) => {
    const token = useUserAuthStore.getState().token;
    if (token) {
      config.headers['Authorization'] = `Bearer ${token}`;
    }
    return config;
  },
  (error) => {
    return Promise.reject(error);
  },
);
```

Obrázek 7.8 Inicializácia Axios inštalácie s nastavením základnej URL adresy a hlavičiek. Interceptor zabezpečuje automatické pridanie autorizačného tokenu ku každému requestu.

(zdroj: vlastný)

```
lib > TS api.ts > ...
export const getUser = async ({
  id,
  includeRecords = true,
}: getUserRequestType): Promise<UserTypeWithRecords> => {
  const response = await api.get(
    `/users/${id}?includeRecords=${includeRecords}`,
  );
  return response.data;
};

export const registerUser = async (
  userData: Pick<UserType, 'email' | 'password' | 'password_confirmation'>,
) => {
  const response = await api.post('/users/register', userData);
  return response.data;
};
```

Obrázek 7.9 Ukážka použitia Axios inštalácie v konkrétnych API funkciách: získanie používateľky podľa ID a registrácia novej používateľky pomocou POST požiadavky.

(zdroj: vlastný)

Pri práci s `api` v komponentoch sa využil `TanStack Query` na volanie API funkcií a spracovanie ich stavu (loading, error, success).

```
const createRecordMutation = useMutation({
  mutationFn: createRecord,
  onSuccess: () => {
    queryClient.invalidateQueries({ queryKey: ['records'] });
    present({
      message: 'Záznam byl úspěšně vytvořen',
      duration: 2000,
      position: 'bottom',
    });
  },
  onError: () => {
    present({
      message: 'Chyba při vytváření záznamu',
      duration: 2000,
      position: 'bottom',
    });
  },
});
```

Obrázek 7.10 Ukážka použitia useMutation z knižnice TanStack Query pre vytvorenie nového záznamu. V prípade úspechu sa invaliduje cache pre kľúč records a používateľka je informovaná o výsledku operácie.

(zdroj: vlastný)

Na správu globálneho stavu bol použitý minimalistický nástroj Zustand, ktorý umožňuje uložiť informácie o používateľke a jej bearer tokenu do localStorage. Okrem toho sa využíva napríklad pri zdieľaní stavu vytváraných záznamov, uchovávaní dát medzi krokmi formulára alebo pri zobrazovaní motivačných prvkov. Zustand bol uprednostnený oproti Redux knižnici primárne kvôli jeho jednoduchosti a veľkosti balíka.

```

import { RecordSchemaType } from '@lib/formSchemas';
import { create } from 'zustand'; 822 (gzipped: 466)
import { createJSONStorage, persist } from 'zustand/middleware'; 2.3k (gzipped: 1.1k)

type RecordState = Partial<RecordSchemaType>;

type RecordActions = {
  setData: (data: Partial<RecordSchemaType>) => void;
  clearData: () => void;
};

const initialState: RecordState = {
  startDatetime: undefined,
  duration: undefined,
  amountMilkLeft: undefined,
  amountMilkRight: undefined,
  mood: undefined,
};

export const useCreateRecordStore = create<RecordState & RecordActions>(()(
  persist(
    (set) => ({
      ...initialState,
      setData: (data) => set(data),
      clearData: () => {
        set(initialState);
      },
    }),
    {
      name: 'create-record-storage',
      storage: createJSONStorage(() => localStorage),
    },
  ),
);

```

Obrázek 7.11 Ukážka definície úložiska useCreateRecordStore vytvoreného pomocou knižnice Zustand. Stav záznamu je perzistentný v localStorage a poskytuje funkcie pre aktualizáciu a vymazanie dát pri viac-krokovom formulári.

(zdroj: vlastný)

7.3.5 Funkcionalita a používateľská interakcia

Po správe používateľov boli implementované hlavné funkcionality: pridávanie, úprava a mazanie záznamov o odsávaní.

Súčasťou aplikácie je aj zobrazovanie tzv. **milestones** – motivačných bublín, ktoré povzbudzujú používateľku pri dosiahnutí určitých cieľov.

Nový záznam
základní údaje

Začátek odsávání
13.05.2025, 22:01

Doba odsávání
23 min (do 22:24)

Odsáté mléko (pravý prs)
22 ml

Odsáté mléko (levý prs)
21 ml

Zrušit **Pokračovat**

Domov Přehled Upozornění Profil

Obrázek 7.12 Obrazovka s formulářem pre vytvorenie nového záznamu o odsávaní. Používateľka zadáva základné údaje ako čas začiatku, trvanie a množstvo odsátého mlieka z oboch prsníkov.



Obrázek 7.13 Příklad zobrazení motivačnej bubliny (milestone) na domovskej obrazovke po splnení cieľa. Tieto bubliny podporujú používateľku v pravidelnom odsávaní.

Počas celého vývoja bol vykonávaný priebežný refactoring, aby bola zachovaná čistota kódu a jeho rozšíriteľnosť do budúcnosti.

7.4 Testovanie

Testovanie bolo neoddeliteľnou súčasťou vývoja aplikácie, pričom sa kládol dôraz na manuálne testovanie používateľskej interakcie, API komunikácie a funkčnosti na rôznych zariadeniach a prehliadačoch.

7.4.1 Testovanie backendu

Backendové API bolo testované primárne pomocou nástroja Postman, kde boli vytvorené kolekcie požiadaviek pre CRUD operácie (vytvorenie, čítanie, aktualizácia, mazanie záznamov).

Overovala sa správnosť odpovedí, HTTP kódy, formát JSON výstupov aj spracovanie autentifikácie cez Laravel Sanctum (napr. overenie platnosti tokenu).

Testovali sa aj okrajové prípady ako napr. nesprávne zadané údaje, pokus o prístup bez autorizácie alebo neexistujúce ID záznamu.

Počas vývoja boli priebežne využívané aj Laravel validácie a chybové hlášky, ktoré poskytovali spätnú väzbu o nesprávnych vstupoch.

7.4.2 Testovanie frontendu

Frontend bol testovaný priamo v prehliadačoch Chrome, Firefox a Safari, ako aj v mobilných zariadeniach so systémami **Android** a **iOS**. Zameranie testovania bolo najmä na:

- správne zobrazovanie komponentov a responzívny dizajn,
- funkčnosť formulárov (validácia, zobrazenie chýb, resetovanie po odoslaní),
- spracovanie API odpovedí (napr. úspešné a neúspešné requesty),
- navigáciu medzi obrazovkami a ukladanie údajov do lokálneho úložiska.

Notifikácie boli testované na reálnych zariadeniach, aby sa overila funkcionálnosť lokálnych notifikácií v inštalovanej PWA. Zároveň sa testovalo, či aplikácia správne detekuje obmedzenia podľa operačného systému (Android vs. iOS).

7.4.3 Používateľské testovanie

Aplikácia bola otestovaná aj koncovými používateľkami v pilotnej fáze, pričom cieľom bolo získať spätnú väzbu ohľadom intuitívnosti používateľského rozhrania, zrozumiteľnosti textov a celkovej použiteľnosti.

Na základe spätnej väzby boli vykonané drobné úpravy v textoch, umiestnení tlačidiel a odstránení nepotrebných častí.

7.4.4 Testovanie PWA funkcionality

Testovalo sa správanie aplikácie ako progresívnej webovej aplikácie (PWA): offline režim, inštalácia na domovskú obrazovku, rýchlosť načítania, cacheovanie zdrojov a fungovanie bez internetu. Bol použitý nástroj Lighthouse (v DevTools), ktorý overil PWA parametre ako napríklad dostupnosť, výkonnosť a dodržiavanie best practices. Overovalo sa aj, či aplikácia správne funguje po aktualizácii verzie (napr. vymazanie starej cache).

8 VYHODNOTENIE A DISKUSIA

Výsledná aplikácia spĺňa väčšinu definovaných funkcionálnych a nefunkcionálnych požiadaviek. Poskytuje používateľkám základ k zaznamenávaniu a sledovaniu odsávania materského mlieka. Projekt sa vyvíja správnym smerom a je pripravený na prvé kolá testovania v reálnom prostredí.

Medzi implementované funkcionality patria:

- pridávanie, úprava a mazanie záznamov odsávania,
- vytvorenie a úprava používateľského profilu,
- možnosť pridať fotografiu dieťaťa,
- možnosť resetovať zabudnuté heslo
- zobrazenie denného prehľadu odsávania,
- zobrazovanie motivačných správ na základe denného cieľa
- nastavenie upozornenia na odsávanie za vybraný čas
- zobrazenie kontaktov podpory v rámci KNTB

8.1 Požívateľské testovanie

Necelé dva týždne pred termínom odovzdania tejto práce bola aplikácia sprístupnená na testovanie aj sestrám z nemocničného personálu KNTB. Aplikácia im bola distribuovaná formou URL adresy a QR kódu, ktorý umožnil jednoduchý prístup prostredníctvom mobilných zariadení a zároveň simuloval reálne použitie v nemocničnom prostredí.

V čase písania tejto práce však ešte nebola k dispozícii relevantná spätná väzba zo strany testujúcich osôb. Predpokladá sa, že poznatky z pilotného testovania budú prezentované počas obhajoby diplomovej práce.

ZÁVĚR

Stále sa ukazuje, že odbor softvérového inžinierstva sa dá úspešne kombinovať s inými oblasťami a výsledkom môže byť niečo praktické a prospešné pre spoločnosť. V tomto prípade sa podarilo vytvoriť aplikáciu, ktorá môže pomôcť matkám, najmä tým s nedonosenými deťmi, pri každodennej a často náročnej starostlivosti o dieťa. Obe časti systému – frontend aj backend – boli úspešne nasadené na server a aplikácia je pripravená na ďalšie testovanie a zlepšovanie.

Vývoj aplikácie nebol len technickým cvičením, ale reálnym krokom smerom k zlepšeniu starostlivosti o novorodencov. Spojenie technológií s konkrétnym zdravotníckym kontextom ukazuje potenciál podobných projektov v širšej praxi.

8.2 Možnosti ďalšieho rozvoja

Spolupráca s KNTB bude pokračovať aj po ukončení tejto etapy vývoja. Aplikácia bude predstavená čerstvým matkám, ktoré môžu poskytnúť dôležitú spätnú väzbu ohľadom funkčnosti, použiteľnosti a dizajnu. Ich názory budú kľúčové pre ďalšie smerovanie projektu.

Ako každý produkt, aj tento má svoje nedostatky, ktoré je možné odstrániť len iteratívnym vývojom. Spätná väzba od používateľov, ale aj od nemocničného personálu, bude dôležitým vstupom pri návrhu a implementácii ďalších verzií.

Medzi konkrétnymi možnosťami ďalšieho rozvoja patria:

- **Internacionalizácia (i18n)** – preklad aplikácie do ďalších jazykov, minimálne češtiny a angličtiny, aby bola vhodná aj pre cudzojazyčné používateľky alebo iné nemocnice.
- **Vlastná doména a stabilná hostingová infraštruktúra** – presun z testovacieho prostredia (Vercel) na plnohodnotnú produkčnú platformu s vlastným backend serverom.
- **Export a zdieľanie dát** – možnosť exportovať údaje o odsávaní vo formáte PDF alebo CSV pre potreby zdravotného personálu alebo vlastné sledovanie.
- **Podpora synchronizácie medzi zariadeniami** – najmä v prípade, že matka používa aplikáciu na viacerých zariadeniach alebo v spolupráci s nemocnicou.
- **Zlepšenie používateľského rozhrania** – napríklad zjednodušenie zadávania údajov, vizuálne štatistiky, denné notifikácie alebo pripomienky.

- **Zabezpečenie a ochrana osobných údajov** – v prípade širšieho nasadenia bude nutné riešiť šifrovanie dát a dodržiavanie legislatívnych noriem (napr. GDPR).

Cieľom do budúcnosti je vytvoriť stabilný, používateľsky prívetivý a zdravotnícky overený nástroj, ktorý môže pomáhať matkám nielen v jednej nemocnici, ale potenciálne aj v celonárodnom alebo medzinárodnom rozsahu.

SEZNAM POUŽITÉ LITERATURY

- [1] WORLD HEALTH ORGANIZATION, 2017. *Protecting, promoting and supporting breastfeeding in facilities providing maternity and newborn services*. World Health Organization. ISBN 978-92-4-155008-6.
- [2] ACADEMY OF BREASTFEEDING MEDICINE, 2022. Academy of Breastfeeding Medicine Clinical Protocol #36: The Mastitis Spectrum, Revised 2022. *Breastfeeding Medicine*. 2022. 17, č. 5, s. 360–376. Dostupné z: 10.1089/bfm.2022.29202. [cit. 2025-05-18].
- [3] VICTORA, Cesar G.; BAHL, Rajiv; BARROS, Aluísio J. D.; FRANÇA, Giovanny V. A.; HORTON, Susan et al., 2016. Breastfeeding in the 21st century: epidemiology, mechanisms, and lifelong effect. *The Lancet*. 2016. 387, č. 10017, s. 475–490. Dostupné z: 10.1016/S0140-6736(15)01024-7. [cit. 2025-27-04].
- [4] LAWRENCE, Ruth A. a LAWRENCE, Robert M., 2016. *Breastfeeding: A Guide for the Medical Profession*. Elsevier. Eighth vyd. ISBN 978-0323358011.
- [5] TOMLINSON, et al. S. K., 2023. Human milk and breastfeeding in the preterm infant. *UpToDate*. 2023. [cit. 2025-27-04].
- [6] RASCASONE, 2021. *Co jsou progresivní webové aplikace (PWA) a jaké mají výhody*. Online. Dostupné z: <https://www.rascasone.com/cs/blog/co-jsou-progresivni-webove-aplikace-pwa-a-jake-maji-vyhody/>. [cit. 2025-27-04].
- [7] MDN WEB DOCS, 2025. *What is a progressive web app?* Online. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Guides/What_is_a_progressive_web_app. [cit. 2025-28-04].
- [8] GOOGLE DEVELOPERS, 2021. *Service workers*. Online. Dostupné z: <https://web.dev/learn/pwa/service-workers>. [cit. 2025-19-05].
- [9] MDN WEB DOCS, 2025. *Service Worker API*. Online. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API. [cit. 2025-19-05].
- [10] SIMICART, 2025. *PWA Service Worker for Dummies*. Online. Dostupné z: <https://simicart.com/blog/pwa-service-worker/>. [cit. 2025-19-05].
- [11] MDN WEB DOCS, 2025. *Web application manifest - Progressive web apps*. Online. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Manifest. [cit. 2025-19-05].
- [12] PIXELFREESTUDIO, 2024. *Best Practices for PWA Security*. Online. Dostupné z: <https://blog.pixelfreestudio.com/best-practices-for-pwa-security/>. [cit. 2025-30-04].
- [13] BRAINHUB, 2025. *PWA on iOS - Current Status, Limitations and Best Practices*. Online. Dostupné z: <https://brainhub.eu/library/pwa-on-ios>. [cit. 2025-28-04].
- [14] ATAK INTERACTIVE, 2025. *What's New in PWAs for 2025?* Online. Dostupné z: <https://www.atakinteractive.com/blog/whats-new-in-pwas-for-2025>. [cit. 2025-28-04].
- [15] CHAUHAN, Ayush, 2023. *Top Frameworks and Tools to Build Progressive Web Apps*. Online. Dostupné z: <https://www.techaheadcorp.com/blog/top-frameworks-tools-to-build-progressive-web-apps/>. [cit. 2025-30-04].
- [16] APPINVENTIV, 2024. *Top 10 Frameworks and Tools To Build Progressive Web Apps*. Online. Dostupné z: <https://appinventiv.com/blog/top-pwa-development-frameworks/>. [cit. 2025-30-04].

- [17] MDN WEB DOCS, 2025. *Making PWAs installable*. Online. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Guides/Making_PWAs_installable. [cit. 2025-28-04].
- [18] FOURRAGE, Ludo, 2025. *Progressive Web Apps (PWAs) in 2025: The Future of Web and Mobile Integration*. Online. Dostupné z: <https://www.nucamp.co/blog/coding-bootcamp-full-stack-web-and-mobile-development-2025-progressive-web-apps-pwas-in-2025-the-future-of-web-and-mobile-integration>. [cit. 2025-05-20].
- [19] PANCHAL, Rahul, 2024. *12+ Top Examples of PWA (Progressive Web Applications) From Different Industry*. Online. Dostupné z: <https://www.rlogical.com/blog/examples-of-pwa/>. [cit. 2025-05-20].
- [20] ZORBIS INC., 2024. *PWA vs Native vs Hybrid App: A Comprehensive Comparison*. Online. Dostupné z: <https://medium.com/@zorbisinc/pwa-vs-native-vs-hybrid-app-a-comprehensive-comparison-9b7fbe870d13>. [cit. 2025-30-04].
- [21] PRAKASH TIWARI, 2024. *Native vs. Hybrid vs. Web vs. PWA: Understanding Mobile App Development Options*. Online. Dostupné z: <https://medium.com/@prakashtet49/native-vs-hybrid-vs-web-vs-pwa-understanding-mobile-app-development-options-6fba844cb73a>. [cit. 2025-30-04].
- [22] TECHAHEAD, 2024. *Mobile App UI Design: Examples and Design Principles*. Online. Dostupné z: <https://www.techaheadcorp.com/blog/mobile-app-ui-design-examples-design-principles/>. [cit. 2025-30-04].
- [23] VERTICALMOTIONTHUMBHEATMAP, 2023. *Mobile App UX Design: 3 Key Principles for Crafting a Meaningful User Experience*. Online. Dostupné z: <https://www.verticalmotion.ca/mobile-app-ux-design-3-key-principles-for-crafting-a-meaningful-user-experience/>. [cit. 2025-30-04].
- [24] W3C, 2018. *Web Content Accessibility Guidelines (WCAG) 2.1*. Online. Dostupné z: <https://www.w3.org/TR/WCAG21/>. [cit. 2025-19-05].
- [25] W3C WEB ACCESSIBILITY INITIATIVE, 2023. *Images Tutorial*. Online. Dostupné z: <https://www.w3.org/WAI/tutorials/images/>. [cit. 2025-19-05].
- [26] W3C WEB ACCESSIBILITY INITIATIVE, 2023. *Semantic Structure*. Online. Dostupné z: <https://www.w3.org/WAI/standards-guidelines/act/rules/html-semantic-roles/>. [cit. 2025-19-05].
- [27] GUPTA, Aashi, 2025. *Best Practices and Strategies for PWA Security for Developers*. Online. Dostupné z: <https://www.searchmyexpert.com/resources/progressive-web-app/security-aspects-progressive-web-app>. [cit. 2025-30-04].

SEZNAM OBRÁZKŮ

Obrázek 2.1	Porovnanie rôznych typov mobilných aplikácií.....	31
Obrázek 3.1	Porovnanie odporúčaných veľkostí dotykových prvkov v mobilných aplikáciách.....	36
Obrázek 3.2	Heatmapy dosahu prstu na mobilnom telefóne.....	37
Obrázek 5.1	Ukážka aplikácie Pump Log	47
Obrázek 5.2	Ukážka aplikácie My Baby	47
Obrázek 5.3	Ukážka aplikácie Beastfeeding	48
Obrázek 5.4	Ukážka aplikácie Milk Stash	48
Obrázek 6.1	Základná databázová štruktúra aplikácie.	56
Obrázek 6.2	Farebná schéma používateľského rozhrania.	57
Obrázek 6.3	Typografická hierarchia aplikácie.	58
Obrázek 6.4	Základné UI komponenty aplikácie.....	59
Obrázek 6.5	Obrazovky prihlásenia a registrácie.....	60
Obrázek 6.6	Hlavná obrazovka a pridanie nového záznamu.....	60
Obrázek 6.7	Obrazovka profilu a prehľad zaznamenaných údajov.....	61
Obrázek 7.1	Logika pre lokálne notifikácie.	64
Obrázek 7.2	Podpora webových notifikácií v prehliadačoch	65
Obrázek 7.3	Ukážka JSON odpovede	67
Obrázek 7.4	Návod na inštaláciu aplikácie na iOS zariadení.....	69
Obrázek 7.5	Štruktúra projektu vo frontend časti	71
Obrázek 7.6	Definícia routovania aplikácie v rámci komponenty App	72
Obrázek 7.7	Implementácia komponenty spodnej navigácie TabBar.....	72
Obrázek 7.8	Nastavenie Axios inštancie s interceptormi	73
Obrázek 7.9	Príklady funkcií pre prácu s používateľskými dátami	73
Obrázek 7.10	Použitie TanStack Query pri vytváraní záznamu	74
Obrázek 7.11	Správa stavu pri vytváraní záznamu pomocou Zustand	75
Obrázek 7.12	Obrazovka pre pridanie nového záznamu	76
Obrázek 7.13	Motivačná bublina pri dosiahnutí cieľa.....	76

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AMP	Accelerated Mobile Pages
API	Application Programming Interface
APNs	Apple Push Notification Service
CLI	Command Line Interface
CRM	Customer Relationship Management
CRUD	Create, Read, Update, Delete
CSV	Comma-Separated Values
CSS	Cascading Style Sheets
DOM	Document Object Model
FCM	Firebase Cloud Messaging
GDPR	General Data Protection Regulation
GPS	Global Positioning System
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IoT	Internet of Things
JSON	JavaScript Object Notation
KNTB	Krajská nemocnice Tomáše Bati
NFC	Near Field Communication
ORM	Object-Relational Mapper
PDF	Portable Document Format
PWA	Progressive Web Application
QR	Quick Response
REST	REpresentational State Transfer
SEO	Search Engine Optimization
SIDS	Sudden Infant Death Syndrome
SSL	Secure Sockets Layer
UI	User Interface
USB	Universal Serial Bus
UX	User Experience
WCAG	Web Content Accessibility Guidelines