

Rozšíření webové aplikace pro evidenci aktivit a činností zaměstnanců UTB

Martin Koňářík

Bakalářská práce
2025



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

Ústav informatiky a umělé inteligence

Akademický rok: 2024/2025

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: Martin Koňářík
Osobní číslo: A22016
Studijní program: B0613A140020 Softwarové inženýrství
Forma studia: Prezenční
Téma práce: Rozšíření webové aplikace pro evidenci aktivit a činností zaměstnanců UTB
Téma práce anglicky: Extension of the Web Application for Recording Activities and Tasks of TBU Employees

Zásady pro vypracování

- Zpracujte literární rešerši na dané téma.
- Vysvětlete pojmy čistého a lehce udržitelného kódu frontendových aplikací.
- Proveďte analýzu stávajícího systému evidence aktivit zaměstnanců UTB a navrhnete jeho rozšíření o požadované funkce.
- Navrhnete a popíšete základní prvky nového uživatelského prostředí webové aplikace. Popište princip napojení nového uživatelského rozhraní na databáze stávající aplikace.
- Zvolte vhodný programovací jazyk a webovou aplikaci sestavte. Zdůvodněte volbu programovacího jazyka, ve kterém je výsledná front-endová část aplikace napsána.
- Komentujte rizika a zabezpečení vytvořené aplikace.

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

Seznam doporučené literatury:

1. PETŘÍK, Jakub. *Porovnání JS frameworks podle specifik projektu*. 2022. Dostupné také z: <http://hdl.handle.net/10563/50562>.
2. MARTIN, Robert C.; OTTINGER, Timothy R.; LANGR, Jeff; SCHUCHERT, Brett L.; GRENNING, James W. et al. *Clean code: a handbook of agile software craftsmanship*. Robert C. Martin series. Upper Saddle River: Prentice Hall, [2009]. ISBN 9780132350884.
3. BLOCH, Joshua. *Effective Java*. 2nd ed. The Java series. Upper Saddle River: Addison-Wesley, 2008. ISBN 9780321356680.
4. *Next.js [online] [cit. 2023-11-10]* Dostupné z: <https://nextjs.org/>
5. REED, Mark. *Python programming and SQL: 5 book in 1 – The #1 coding course from beginners to advanced. Learn it well & fast*, [2022]. ISBN 9798373278850.

Vedoucí bakalářské práce: **doc. Ing. Bc. Bronislav Chramcov, Ph.D.**
Ústav informatiky a umělé inteligence

Konzultant bakalářské práce: **Ing. Ivan Masár**
Ústav informatiky a umělé inteligence

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

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

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



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

Ve Zlíně dne 17. listopadu 2024

PROHLÁŠENÍ AUTORA BAKALÁŘSKÉ PRÁCE

Beru na vědomí, že

- odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení bakalářské práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- na moji bakalářskou 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 – bakalářskou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mě 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 bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- pokud je výstupem bakalářské 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 bakalářské práci pracoval samostatně a použitou literaturu jsem řádně citoval; v případě publikace výsledků budu uveden jako spoluautor;
- odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.
- při tvorbě této práce jsem využil nástroj generativního modelu AI ChatGPT, dostupné z <https://chatgpt.com/> za účelem pomoci parafrázování v teoretické části. Po použití tohoto nástroje jsem provedl kontrolu obsahu a přebírám za něj plnou zodpovědnost.

Ve Zlíně, dne

.....

podpis autora

ABSTRAKT

Cílem této bakalářské práce je rozšíření webové aplikace určené k zobrazení hodnocení akademických pracovníků Univerzity Tomáše Bati ve Zlíně. Výsledná aplikace je postavena nad databází původního proprietárního systému, jehož uživatelské rozhraní není možné dále rozšiřovat. Nově vyvinuté řešení přináší nové funkcionality, jako jsou šablony otázek pro předvyplnění hodnotících formulářů nebo bodovací šablony, které umožňují výpočet výsledného hodnocení pracovníků na základě různých předpisů či univerzitních směrnic.

Klíčová slova: webová aplikace, hodnocení akademických pracovníků, Flask, Python

ABSTRACT

The aim of this bachelor theses is to extend the web application designed to display the evaluation of academic staff at Tomas Bata University in Zlín. The resulting application is built on top of the database of the original proprietary system, whose user interface cannot be further extended. The newly developed solution introduces additional functionalities such as question templates for pre-filling evaluation forms and scoring templates that allow calculation of final evaluations results based on various regulations or university guidelines.

Keywords: web application, academic staff evaluation, Flask, Python

Tímto bych rád poděkoval vedoucímu práce panu doc. Ing. Bronislavu Chramcovovi, Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích.

Dále bych chtěl vyjádřit vděčnost panu Ing. Ivanu Masárovi za jeho rady a čas, který věnoval konzultacím praktické části práce.

V neposlední řadě děkuji své rodině a přítelkyni za podporu a trpělivost po celou dobu mého studia.

OBSAH

ÚVOD.....	11
I TEORETICKÁ ČÁST	12
1 SYSTÉMY PRO HODNOCENÍ AKADEMICKÝCH PRACOVNÍKŮ	13
1.1 IS HAP.....	13
1.2 DALŠÍ SYSTÉMY PRO HODNOCENÍ.....	14
2 TECHNOLOGIE PRO VÝVOJ WEBOVÝCH APLIKACÍ	16
2.1 ZÁKLADNÍ WEBOVÉ TECHNOLOGIE.....	16
2.1.1 HTML	16
2.1.2 CSS.....	17
2.1.3 JavaScript	18
2.2 FRAMEWORKY.....	19
2.2.1 Backend Frameworky	19
2.2.2 Frontend Frameworky	21
3 FLASK.....	24
3.1 ROUTOVÁNÍ.....	24
3.2 JINJA2 ŠABLONY	26
3.3 SESSION.....	27
3.4 KEŠOVÁNÍ	27
4 PRINCIPY ČISTÉHO A LEHCE UDRŽITELNÉHO KÓDU.....	28
4.1 SROZUMITELNÉ NÁZVY	28
4.2 FUNKCE.....	29
4.3 KOMENTÁŘE	30
4.4 FORMÁTOVÁNÍ.....	30
4.5 TESTOVÁNÍ.....	30
II PRAKTICKÁ ČÁST	32
5 POPIS EXISTUJÍCÍHO SYSTÉMU.....	33
6 FUNKČNÍ POŽADAVKY.....	37
7 IMPLEMENTACE WEBOVÉ APLIKACE	38
7.1 STRUKTURA PROJEKTU	38
7.2 PRÁCE S DATABÁZEMI.....	40
7.3 IMPLEMENTACE JEDNOTLIVÝCH FUNKCIONALIT	40
7.3.1 Osobní hodnocení.....	40
7.3.2 Otázky z formulářů	42
7.3.3 Hodnocení pracoviště.....	43
7.3.4 Bodovací šablony	44
7.3.5 Šablony otázek	47

7.3.6	Kopírování produkční databáze.....	49
7.3.7	Párování otázek z nového formuláře	50
7.3.8	Import publikací	50
8	ZABEZPEČENÍ APLIKACE	51
8.1	ROLE A ŘÍZENÍ PŘÍSTUPU	51
8.2	VALIDACE ŠABLON OTÁZEK	52
	ZÁVĚR.....	54
	SEZNAM POUŽITÉ LITERATURY	55
	SEZNAM OBRÁZKŮ	60
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	61
	SEZNAM PŘÍLOH.....	62

ÚVOD

Vysoké školy v České republice jsou ze zákona povinny provádět hodnocení pedagogických, tvůrčích a s nimi souvisejících činností. S tímto hodnocením je často spojeno také hodnocení akademických pracovníků. K tomuto účelu vysoké školy využívají různé informační systémy. Jedním z nich je IS HAP, který je využíván i na Univerzitě Tomáše Bati ve Zlíně.

Akademičtí pracovníci v rámci tohoto systému vyplňují každoročně hodnotící formulář, do něhož zaznamenávají informace o svých vykázaných činnostech za uplynulý rok. Pro zobrazení výsledků hodnocení byla na univerzitě vyvinuta webová aplikace, která je však proprietární a jejíž uživatelské rozhraní nelze dále rozšiřovat.

Cílem této bakalářské práce je vytvořit nové uživatelské rozhraní nad databází stávajícího systému, a to podle požadavků zadavatele.

Teoretická část práce se věnuje přehledu existujících systémů pro hodnocení akademických pracovníků a základním technologiím používaným při vývoji webových aplikací. Zároveň jsou zde popsány konkrétní technologie využité při vývoji výsledné aplikace. Součástí teoretické části je i kapitola zaměřená na principy psaní čistého a lehce udržitelného kódu.

Praktická část se zaměřuje na popis stávajícího systému pro zobrazení výsledků hodnocení, shrnutí aplikačních požadavků na nové uživatelské rozhraní a následnou implementaci jednotlivých funkcionalit. Součástí této části jsou také ukázky z uživatelského prostředí výsledné aplikace a popis způsobu zabezpečení přístupu k aplikaci.

I TEORETICKÁ ČÁST

1 SYSTÉMY PRO HODNOCENÍ AKADEMICKÝCH PRACOVNÍKŮ

Vysoké školy v České republice musí dle § 77a zákona č. 111/1998 Sb. o vysokých školách pravidelně provádět hodnocení vzdělávacích, tvůrčích a s nimi souvisejících činností. Pojem „akademický pracovník“ je definován v § 70 zákona o vysokých školách: *„Akademickými pracovníky jsou ti profesori, docenti, mimořádní profesori, odborní asistenti, asistenti, lektori a vědečtí, výzkumní a vývojoví pracovníci, kteří jsou zaměstnanci veřejné nebo soukromé vysoké školy nebo zaměstnanci České republiky zařazenými k výkonu práce na státní vysoké škole a vykonávají v pracovním poměru podle sjednaného druhu práce jak pedagogickou, tak tvůrčí činnost.“* [1]

K hodnocení činností akademických pracovníků se využívají různé nástroje a systémy. Mezi příklady patří IS HAP používaný na Univerzitě Tomáše Bati ve Zlíně nebo systém EVAK na Masarykově Univerzitě v Brně. V některých případech, jako například na Univerzitě Hradec Králové, probíhá hodnocení prostřednictvím jednoduchých formulářů bez specializovaného softwaru.

1.1 IS HAP

Informační systém pro hodnocení akademických pracovníků (dále jen IS HAP) funguje již od roku 2012, kdy byl vyvinut na Přírodovědecké fakultě Univerzity Palackého v Olomouci. Vývoji tohoto systému předcházelo několik let výzkumu, kdy byl hledán vhodný matematický model pro hodnocení a proběhla analýza mnoha systémů pro hodnocení akademických pracovníků v České republice a ve světě. Aktuálně je IS HAP využíván jako nástroj na celkem šesti českých vysokých školách. [2][3]

Jednotliví akademičtí pracovníci jsou hodnoceni ve vzdělávací a tvůrčí činnosti. Hodnocení probíhá formou formuláře, který je předvyplněn daty z dalších informačních systému univerzity (např. STAG¹, OBD²). Kromě toho se do formuláře nahrají data i z jiných zdrojů (např. RUV³) a zbytek formuláře je doplněn daty, které poskytnou sami pracovníci. Otázky ve formuláři jsou bodově ohodnoceny na základě významu či časové náročnosti aktivity, kterou daná otázka popisuje. Na obrázku 1 je zobrazena část hodnotícího formuláře za rok 2024. [2]

¹ STAG – Informační systém pro administraci studijní agendy vysokých škol

² OBD – Osobní bibliografická databáze; systém pro evidenci publikační činnosti

³ RUV – Registr uměleckých výstupů; systém pro evidenci a hodnocení výsledků tvůrčí činnosti

Obrázek 1 Část formuláře IS HAP

Výsledná hodnocení za jednotlivé činnosti jsou znázorněna graficky a popsána slovně. Celkové hodnocení je taktéž dostupné v grafické a verbální podobě a je vypočítáno sjednocením hodnocení za vzdělávací a tvůrčí činnost pomocí jazykově orientovaného fuzzy expertního systému. Agregace tímto způsobem je srozumitelná i pro laiky, neboť je popsána pomocí jednoduchých pravidel typu „Jestliže..., pak...“ Kromě hodnocení v jednotlivých oblastech a agregovaného celkového hodnocení IS HAP poskytuje i grafický a verbální popis celkového vytížení akademického pracovníka, který při výpočtu zohledňuje manažerské činnosti a zatížení pracovníka jeho akademickými funkcemi. [2]

Přestože si vysoké školy mohou některé části IS HAP upravit podle svých potřeb, například přidat vlastní oblasti a otázky do formuláře, výpočet výsledného hodnocení nabízený tímto systémem jim nemusí vyhovovat. Z tohoto důvodu si některé české vysoké školy vytvářejí vlastní řešení, případně využívají systémy od externích dodavatelů.

1.2 Další systémy pro hodnocení

Masarykova Univerzita v Brně je příkladem vysoké školy, která využívá vlastní řešení – aplikaci EVAK, jejíž název vznikl zkrácením slov „evaluace akademiků“. Tento nástroj je součástí informačního systému univerzity a umožňuje vedoucím pracovníkům získat přehled o pracovních činnostech jednotlivých zaměstnanců, který následně slouží jako podklad pro jejich hodnocení. Výstupem ze systému je souhrn vykonávaných činností, zhodnocení výkonu a plán aktivit na další období. [4]

Samotné hodnocení probíhá v EVAKu v několika krocích:

1. Hodnotitel založí novou etapu hodnocení.
2. Hodnotitel doplní a edituje kritéria pro hodnocení.
3. Hodnocení zaměstnanci vyplní v aplikaci požadovaná data.
4. Hodnotitel doplní ke kritériím své komentáře.
5. Na základě sesbíraných dat je proveden hodnotící pohovor mezi hodnotitelem a hodnoceným zaměstnancem.
6. Závěry z hodnotícího pohovoru jsou uloženy do písemného záznamu o hodnocení.
7. Hodnocený zaměstnanec odsouhlasí závěrečný záznam o hodnocení. [5]

Na Univerzitě Hradec Králové vychází hodnocení akademických pracovníků z „*Kariérního řádu a pravidelného hodnocení akademických pracovníků*“. Pracovní výsledky ve vzdělávací a tvůrčí oblasti jsou posuzovány jednou ročně, a to s ohledem na další funkce, které pracovník v rámci univerzity zastává. [6]

Hodnocení probíhá prostřednictvím „*Formuláře osobního hodnocení a plánu kariérního rozvoje akademických pracovníků*“, který zaměstnanec vyplňuje na základě výsledků uplynulého období a zároveň navrhuje plán pro další rok. Vedoucí pracoviště následně provádí výsledné hodnocení na základě formuláře a informací ze systémů univerzity. Toto hodnocení slouží jako podklad pro návrh osobního ohodnocení pracovníka na následujících dvanáct měsíců. Výši odměny navrhuje děkanovi vedoucí pracoviště s ohledem na ekonomické možnosti daného pracoviště. [6]

2 TECHNOLOGIE PRO VÝVOJ WEBOVÝCH APLIKACÍ

Vývoj webových aplikací umožňuje zpřístupnit uživatelům konkrétní funkcionalitu přímo ve webovém prohlížeči. Oproti desktopovým aplikacím, které jsou často vytvářeny pouze pro konkrétní platformy a před spuštěním je musíme nainstalovat, lze k webovým aplikacím přistoupit z jakéhokoliv zařízení, které má přístup k internetu a obsahuje webový prohlížeč.

Právě kvůli multiplatformnosti a přístupnosti jsou webové aplikace v dnešní době velice populární. Kromě toho jsou webové aplikace vytvářeny pomocí stejných technologií jako webové stránky, a tak na první pohled uživatel mezi těmito produkty ani nemusí poznat rozdíl. Odlišnost webových aplikací se ale skrývá v pokročilejší funkcionalitě oproti klasickým webům. Webové aplikace se tak, spíše než k webovým stránkám, blíží k počítačovým programům. [7]

2.1 Základní webové technologie

Když se bavíme o základních webových technologiích, máme nejčastěji na mysli trojici HTML, CSS a JavaScript. Tyto technologie řadíme do tzv. frontendu. Frontend označuje část aplikace, kterou uživatel přímo vidí a s níž interaguje. Frontend také nazýváme jako „client-side“ nebo prezentační vrstvu aplikace.

Opakem frontendu je poté backend, tedy část aplikace, pomocí které je celá aplikace řízena. Jedná se o „server-side“ neboli vrstvu, která operuje s daty na straně serveru. V backendu pracujeme s programovací jazyky jako jsou Python, Java či C#, se systémy řízení báze dat (např. MySQL, PostgreSQL) či zajišťujeme autentizaci a autorizaci jednotlivých uživatelů. [8][9]

2.1.1 HTML

„HTML neboli HyperText Markup Language je značkovací jazyk, který je určen k vytváření dokumentů, obsahujících hypertextové odkazy a pokročilejší formátování.“ [10]

Díky hypertextům může HTML jednoduše navzájem propojit jednotlivé dokumenty pomocí odkazů. Tímto propojením vzniká síť informací, základ webu, jak ho v dnešní době známe. [11]

Cílem HTML není programování v klasickém slova smyslu, ale spíše organizování obsahu prostřednictvím tzv. elementů. Každý element se zapisuje pomocí značek (tagů), které určují, jaká část dokumentu představuje nadpis, formulářové pole, odkaz nebo třeba

obrázek. Každý HTML dokument je uložen jako textový soubor s příponou .html a může být interpretován webovým prohlížečem.

```
<!DOCTYPE html>
<html>
<head>
  <title>Titul stránky</title>
</head>
<body>
  <h1> Nadpis 1. úrovně </h1>
  <p style="color: red;"> Paragraf obsahující červený text </p>
  <a href="https://fai.utb.cz"> Odkaz na stránky FAI UTB </a>
</body>
</html>
```

Nadpis 1. úrovně

Paragraf obsahující červený text

[Odkaz na stránky FAI UTB](https://fai.utb.cz)

Obrázek 2 Základní ukázka HTML kódu (vlevo) a výstup (vpravo)

Na obrázku 2 lze vidět základní ukázkou souboru obsahující kód jazyka HTML. Každý HTML dokument musí začínat deklarací `<!DOCTYPE html>`. Tato deklarace udává verzi jazyka HTML, která byla použita pro tvorbu daného dokumentu. V ukázce je použita deklarace pro aktuálně nejnovější verzi jazyka – HTML5. Značkou v této ukázce je například `<h1>` a její uzavírací značka `</h1>`. Elementem se potom rozumí celý zápis včetně obsahu mezi těmito značkami, tedy `<h1> Nadpis 1. úrovně </h1>`. [12]

Celý zdrojový kód HTML dokumentu se nachází uvnitř elementu ohraničeného značkami `<html>` a `</html>`. Dokument je rozdělen na dvě hlavní části: hlavičku a tělo. Hlavička vymezená značkami `<head>` a `</head>` obsahuje důležité informace o stránce. Její obsah se na stránce přímo nezobrazuje, ale najdeme zde například název stránky, definici stylů a další metadata. Vlastní obsah stránky poté zapisujeme do těla stránky, které je ohraničeno značkami `<body>` a `</body>`. Vše, co je umístěno mezi tyto značky, se zobrazí na výsledné stránce. [10]

Samotný HTML kód umožňuje vytvoření základní webové stránky se stylováním jako je tučné písmo nebo kurzíva. Pro její pokročilou vizuální podobu, včetně rozvržení, barev nebo responzivního designu, je však nutné využít CSS.

2.1.2 CSS

Cascading Style Sheets (česky kaskádové styly, dále jen CSS) popisují, jakým způsobem se zobrazí HTML elementy na obrazovce, papíře či jiném médiu. Pomocí CSS můžeme ovlivnit více značek najednou, čímž si ušetříme práci a zamezíme zbytečnému opakování kódu. [13]

CSS můžeme do HTML dokumentu připojit třemi různými způsoby: pomocí inline, internal nebo external CSS. Při použití inline CSS nastavujeme stylování přímo uvnitř konkrétní značky, pomocí atributu `style`. Tento způsob však problém s opakováním kódu

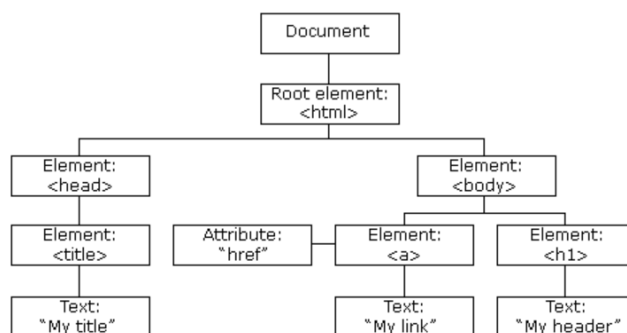
neřeší. Opakování kódu pomáhá eliminovat internal CSS, kdy definici stylů přidáváme přímo do hlavičky dokumentu mezi značky `<style>` a `</style>`. Třetím a zároveň nejvíce univerzálním přístupem je využití external CSS. V tomto případě vytvoříme samostatný soubor s příponou `.css`, který obsahuje všechna stylovací pravidla. Tento soubor se pak do HTML dokumentu připojí pomocí značky `<link>`. Největší výhodou externího CSS je možnost připojit jeden soubor k více HTML stránkám, a tím sjednotit vzhled různých částí webu. Zároveň dokážeme úpravou jednoho externího CSS souboru změnit vzhled celé stránky. [10][13][14]

2.1.3 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk. Jedná se o nejčastěji používaný jazyk při tvorbě webových stránek. Čistý HTML kód nedokáže stránku po jejím načtení nijak měnit. Pomocí JavaScriptu můžeme statickým webovým stránkám přidat dynamičnost a interaktivitu. [15][16]

V souvislosti s využitím JavaScriptu při tvorbě webových stránek o něm hovoříme jako o „client-side“ skriptu, což znamená, že se kód vykonává až po stažení webové stránky přímo na straně klienta. Tento způsob je rychlejší než u „server-side“ programovacích jazyků, jejichž kód se spouští na serveru ještě před odesláním stránky uživateli. [16][17]

K jednotlivým prvkům HTML dokumentu je možné přistupovat pomocí DOM (Document Object Model), který vytváří webový prohlížeč po načtení stránky. HTML DOM představuje programové rozhraní v jazyce JavaScript pro práci s HTML dokumenty. Každý HTML prvek je v DOM reprezentován jako JavaScriptový objekt, který má své vlastnosti a metody. DOM zároveň definuje způsob, jak lze k těmto prvkům přistupovat, měnit je a reagovat na události, které na nich nastávají. Celý model je strukturován jako strom objektů, jak je znázorněno na obrázku 3. [18]



Obrázek 3 HTML DOM jako strom objektů [18]

Podobně jako u CSS lze JavaScript k HTML dokumentu připojit více způsoby: interně, kdy kód vložíme přímo mezi značky `<script>` a `</script>` a externě, kdy vytvoříme samostatný soubor s příponou `.js` a připojíme jej k HTML dokumentu pomocí značky `<script>` s atributem `src`, který odkazuje na umístění souboru. [19]

2.2 Frameworky

Ať už pracujeme na jakémkoliv projektu, práci si můžeme výrazně usnadnit pomocí frameworků. Frameworky nabízejí sadu předem definovaných nástrojů a struktur, díky kterým nemusíme psát celý kód od úplného začátku. Tím lze mimo jiné výrazně urychlit vývoj. Frameworky slouží jako šablony, které můžeme použít a upravit podle specifikací konkrétního projektu. Jejich využití je obzvlášť výhodné zejména při týmové práci, jelikož vzniká jednotný, přehledný, snadno udržitelný a lépe testovatelný kód. Při vývoji webových aplikací používáme webové aplikační frameworky, které rozdělujeme na frontend a backend frameworky. [20]

2.2.1 Backend Frameworky

Backend frameworky jsou základem pro vývoj robustních a škálovatelných webových aplikací, správu serverových operací a databází. Správná volba backend frameworku může výrazně ovlivnit proces vývoje i celkovou výkonnost finální aplikace. [21]

S rostoucí oblibou programovacího jazyka Python vzrostla v posledních letech také popularita frameworků založených právě na Pythonu. Přispěla k tomu především poměrně krátká učicí křivka a univerzální povaha jazyka. Python patří mezi vysokoúrovňové programovací jazyky, což znamená, že se jeho zápis více podobá lidskému jazyku než strojovému kódu. Kód v Pythonu připomíná klasickou angličtinu, a proto je poměrně snadno pochopitelný i zapamatovatelný. Mezi dva hlavní webové backend frameworky vycházející z Pythonu patří Flask a Django. [22][23][24]

Django je vysokoúrovňový webový framework, který nabízí širokou škálu vestavěných funkcí. Díky tomu je ideální volbou pro tvorbu velkých a komplexních aplikací. Projekty v Django se vyznačují přehlednou strukturou souborů, což přispívá k větší konzistenci a kvalitě kódu. Django dále nabízí například předem zabudovanou správu uživatelské autentizace, administrátorské rozhraní a ORM (Object-Relational Mapping) pro snadnější práci s daty z relačních databází. [23]

Flask je minimalistický webový framework, ideální pro menší aplikace s prostorem pro budoucí rozšíření. Nabízí pouze základní funkce, což z něj dělá flexibilní nástroj, který si uživatel může upravit podle potřeb konkrétního projektu. Při použití Flasku tedy nejsme svázáni konkrétními funkcemi jako u Django a máme mnohem větší volnost v návrhu a přizpůsobení aplikace. To je jeden z hlavních důvodů, proč byl v této práci pro vývoj výsledné webové aplikace zvolen právě tento framework. Flask bude podrobněji popsán v kapitole 3. [23]

Joshua Bloch ve své knize říká: „*Žádný programovací jazyk není dokonalý, ale některé jsou opravdu výborné. Dle mého názoru Java a její knihovny nabádají k produktivní, kvalitní práci a je radost s nimi pracovat.*“ [25]

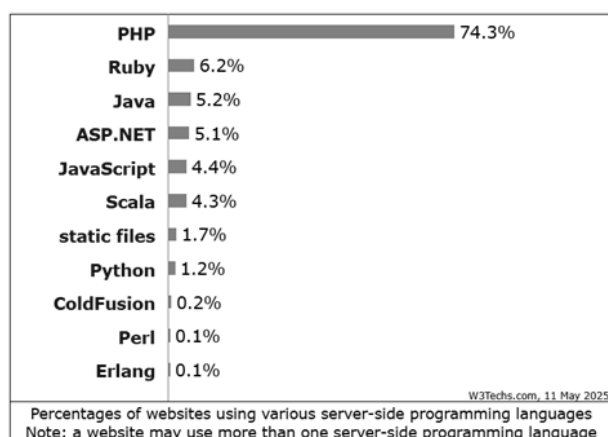
Jedním z nejznámějších backend frameworků pro vývoj webových aplikací v jazyce Java je Spring Framework. Jedná se o výkonný nástroj pro tvorbu podnikových aplikací, který nabízí komplexní programovací a konfigurační model. Díky tomu je vývoj aplikací rychlý, škálovatelný a snadno udržitelný. Mezi hlavní koncepty Springu patří vkládání závislostí (Dependency Injection, zkráceně DI), vytváření DI kontejnerů a aspektově orientované programování (AOP). Spring Framework tvoří základ pro všechny ostatní projekty ekosystému Spring. Spring Boot, Spring Data, Spring Batch – všechny tyto moduly jsou postaveny právě na něm. [26][27]

Dependency Injection je návrhový vzor, který spočívá v tom, že třída nezískává své závislosti (například jiné objekty nebo služby) sama, ale jsou jí předány z vnějšího prostředí – obvykle při jejím vytvoření. Tento přístup zvyšuje flexibilitu, testovatelnost a znovupoužitelnost kódu, protože závislosti lze snadno zaměnit, například za testovací implementace. DI je konkrétní implementací obecnějšího principu Inversion of Control, který říká, že tok řízení aplikace (například vytváření a správa závislostí) by měl být přenesen z objektu samotného na externí entitu, jako je kontejner nebo framework. Místo toho, aby si objekt své závislosti vytvářel sám, dostává je zvenčí. [28]

Aspektově orientované programování je programovací paradigma, jehož cílem je pomoci vývojářům psát čistý a organizovaný kód tím, že odděluje opakující se úkony, například logování či zpracování chyb, od hlavní logiky aplikace. V AOP se podobné funkcionality zapouzdřují do samostatných modulů zvaných aspekty. Tyto aspekty usnadňují správu a úpravy dané části kódu. Představme si, že na každé stránce webu potřebujeme provést bezpečnostní kontrolu. Místo toho, abychom tento kód psali

opakovaně, vytvoříme jeden bezpečnostní aspekt, který se automaticky aplikuje všude, kde je potřeba. Výsledkem je přehlednější a snadněji udržitelný kód. [29]

Podle průzkumu W3Techs je stále nepoužívanějším jazykem při vytváření backend části webových stránek programovací jazyk PHP. Na obrázku 4 lze vidět, že k 11. květnu 2025 je využíván na více než 74 % všech webových stránek. [30]



Obrázek 4 Nepoužívanější server-side programovací jazyky na webových stránkách [30]

Jedním z nepoužívanějších frameworků pro jazyk PHP je Laravel. Jde o open-source nástroj, který využívá architektonický styl Model-View-Controller (MVC). Ten rozděluje aplikaci na tři hlavní části a klade důraz na oddělení obchodní logiky od uživatelského rozhraní. Model se stará o práci s daty, View zajišťuje zobrazení stránky uživateli a Controller řídí komunikaci mezi těmito dvěma částmi – na základě vstupu od uživatele pracuje s Modelem a získaná data předává do View pro zobrazení. [31]

Mezi výhody Laravelu patří například efektivní práce s databází pomocí ORM nebo vlastní šablonovací engine Blade. Ten umožňuje vytvářet dynamické, znovupoužitelné šablony s jednoduchou syntaxí. Blade podporuje také dědičnost, což usnadňuje úpravy vzhledu celé aplikace. Laravel také obsahuje in-line skript Artisan, který slouží k rychlému generování šablonového kódu. [32][33]

2.2.2 Frontend Frameworky

Zatímco backendové frameworky slouží k efektivnějšímu vývoji části webové aplikace, která pro uživatele není viditelná, frontendové frameworky usnadňují tvorbu její klientské části. Tyto frameworky poskytují nástroje a struktury postavené na frontend technologiích – nejčastěji CSS a JavaScriptu. Zejména JavaScriptové frameworky jsou dnes velkým

tématem ve vývoji webových aplikací, protože vhodný výběr frontendového frameworku může výrazně urychlit celý vývojový proces.

Bootstrap je jedním z nejpoužívanějších a velmi populárních CSS frameworků. Byl vyvinut ve společnosti Twitter (dnes známé jako X) s cílem sjednotit vzhled interních aplikací a zjednodušit práci při vizuálních úpravách. Společnost následně framework uvolnila jako open-source, což umožňuje jeho volné používání, úpravy a sdílení mezi vývojáři. [34][35]

Bootstrap je známý především svým důrazem na responzivní design webových stránek a aplikací. K tomu využívá dvanáctisloupcový grid systém, který rozdělí obrazovku na dvanáct neviditelných sloupců. Vývojář pak pouze určuje, kolik sloupců má konkrétní prvek zabírat. Použití tohoto grid systému je velmi jednoduché, stačí označit prvek odpovídající třídou. Bootstrap byl do jisté míry využit pro tvorbu výsledné webové aplikace i v této bakalářské práci. [35]

V posledních letech se stále více diskutuje o JavaScriptových frameworkcích. Názvy jako Angular, React nebo Vue.js dnes zná většina webových vývojářů. K popularitě Angularu a Reactu výrazně přispívá i to, že za nimi stojí velké technologické společnosti – Angular byl vyvinut společností Google, zatímco za Reactem stojí společnost Meta (dříve Facebook). [36]

Angular je open-source framework postavený na jazyce TypeScript – nadstavbě JavaScriptu, který přináší typování a principy objektového programování. Základem Angularu jsou komponenty, které se skládají ze tří částí: HTML šablony, TypeScript třídy a CSS selektoru. Šablona udává, co se má na stránce zobrazit, TypeScript třída definuje chování komponenty a pomocí CSS selektoru specifikujeme, jak je komponenta v rámci šablony použita. [36][37]

Předchůdcem Angularu je AngularJS, který, jak už název napovídá, vychází z klasického JavaScriptu. Ve srovnání s moderním Angularem je pomalejší a méně výkonný. Obě verze jsou navzájem nekompatibilní, a to především proto, že vycházejí z odlišných jazyků a staví na rozdílných principech. [36]

React není framework v pravém slova smyslu, jedná se o knihovnu jazyka JavaScript určenou pro tvorbu uživatelského rozhraní. Pojmy knihovna a framework bychom neměli zaměňovat, protože označují rozdílné věci. Knihovna nabízí sadu pomocných funkcí, objektů či modulů, které náš kód využívá k řešení konkrétních úloh. Oproti tomu framework

tvoří základní kostru aplikace, do které následně doplňujeme vlastní logiku. Knihovny bývají zpravidla menšího rozsahu a máme nad nimi plnou kontrolu – my voláme je. Frameworky bývají rozsáhlejší a definují koncept a strukturu aplikace, přičemž dochází k tzv. inverzi kontroly – framework volá nás. [36][38]

Stejně jako v případě Angularu je i v Reactu hlavní myšlenkou skládání komponent, ze kterých se následně tvoří celá aplikace. Komponenty přijímají data, mohou mít vlastní stav (state) a při změně stavu se tato změna automaticky projeví v uživatelském rozhraní. React využívá tzv. virtuální DOM – interní reprezentaci skutečného DOMu. Při změně komponenty nedochází k překreslení celé stránky, ale React porovná nový virtuální DOM s předchozím a provede pouze minimální nezbytné změny ve skutečném DOMu. Díky tomu je vykreslování velmi efektivní a rychlé. [36][39]

Knihovna React tvoří základ pro několik frameworků. Jedním z nejznámějších je Next.js, který je dokonce doporučován v dokumentaci Reactu jako vhodný nástroj při vývoji full-stack aplikací. V rámci Next.js používáme komponenty Reactu spolu s dalšími funkcemi a optimalizacemi, které tento framework nabízí. Next.js umožňuje vytvářet interaktivní, dynamické a zároveň výkonné aplikace postavené na Reactu. [40][41]

Ve výchozím nastavení generuje Next.js HTML pro každou stránku dopředu, namísto toho, aby bylo vytvářeno až na straně klienta. Tento přístup přispívá k lepšímu výkonu a lepší optimalizaci pro vyhledávače. Každé generované HTML obsahuje pouze nezbytné množství JavaScriptu potřebné pro zajištění interaktivity. Po načtení stránky v prohlížeči se JavaScript spustí, čímž se stránka stává plně interaktivní. [42]

Next.js podporuje dvě formy přednačítání obsahu: statické generování (Static Site Generation, SSG) a serverové renderování (Server Side Rendering, SSR). Rozdíl mezi nimi spočívá v tom, kdy je HTML vytvářeno. U statického generování je HTML vygenerováno při sestavení a následně znovu používáno při každém požadavku. Naopak při serverovém renderování se HTML generuje při každém požadavku na serveru znovu. [42]

Vývojáři si v Next.js mohou zvolit způsob renderování pro každou stránku zvlášť. Díky tomu lze vytvořit hybridní aplikaci, která na některých stránkách využívá statické generování a na jiných serverové renderování. Z hlediska výkonu je obecně doporučeno využívat statické generování, protože výsledné stránky mohou být snadno kešovány bez nutnosti další konfigurace. V některých případech (například u stránek s velkým obsahem dat) je však serverové renderování nezbytné. [42]

3 FLASK

Flask je webový aplikační framework postavený na rozhraní Web Server Gateway Interface (dále jen WSGI). WSGI je specifikace rozhraní mezi webovým serverem a webovou aplikací napsaná v Pythonu. Umožňuje webovým serverům spouštět Python kód a přenášet požadavky a odpovědi mezi serverem a aplikací jednotným způsobem. Flask bývá označován jako microframework, protože jeho jádro je navrženo tak, aby zůstalo co nejjednodušší a zároveň snadno rozšířitelné. Ačkoliv Flask poskytuje určitá doporučení, nevnučuje vývojářům konkrétní strukturu projektu ani specifické knihovny pro řešení běžných problémů. Namísto toho umožňuje volný výběr nástrojů a rozšíření podle potřeb konkrétní aplikace. Díky rozsáhlé komunitě je navíc k dispozici široká škála knihoven, které lze snadno integrovat. [43][44]

Dvě základní knihovny, které jsou součástí Flask projektů, jsou Werkzeug a Jinja2. Werkzeug slouží jako pomocná knihovna pro WSGI a poskytuje například routovací systém pro mapování URL (Uniform Resource Locator, běžně označované jako webové adresy) na konkrétní endpointy. Jinja2 je šablonovací systém pro jazyk Python, který umožňuje vkládání dynamického obsahu do HTML pomocí speciálních zástupných výrazů se syntaxí podobnou Pythonu. [45]

3.1 Routování

Routování je jedním ze základních prvků každé webové aplikace. Spočívá v přiřazení URL ke konkrétní funkci, která zajišťuje logiku dané cesty. Názvy URL by měly být smysluplné, aby byly snadno zapamatovatelné a dobře odrážely účel stránky. Ve Flasku se k mapování funkcí na URL používá dekorátor `@route()`. Na obrázku 5 je ukázka zdrojového kódu jednoduché Flask aplikace. Po spuštění se na úvodní stránce zobrazí jednoduchý text. [46]

```
1  # Jednoduchá Flask aplikace
2  from flask import Flask
3
4  app = Flask(__name__)
5
6  @app.route("/")
7  def index():
8      return "Hlavní stránka"
9
10 if __name__ == "__main__":
11     app.run()
12
```

Obrázek 5 Ukázka jednoduché Flask aplikace

Pomocí tzv. `<variable_name>` pravidla lze ve Flasku snadno vytvářet dynamické URL adresy. Přidáním proměnné přímo do cesty zajistíme, že funkce, na kterou je daná adresa namapována, obdrží její hodnotu jako argument. S touto hodnotou pak můžeme dále pracovat uvnitř funkce nebo ji předat jako parametr do šablony. Volitelně lze využít tzv. konvertor ve tvaru `<converter:variable_name>`, kterým je možné specifikovat datový typ požadované proměnné. Přehled dostupných datových typů konvertorů je uveden na obrázku 6. [46]

<code>string</code>	(default) accepts any text without a slash
<code>int</code>	accepts positive integers
<code>float</code>	accepts positive floating point values
<code>path</code>	like <code>string</code> but also accepts slashes
<code>uuid</code>	accepts UUID strings

Obrázek 6 Typy konvertorů [46]

Ukázka zdrojového kódu s využitím dynamických URL adres je zobrazena na obrázku 7. První z cest očekává proměnnou `name`, kterou následně předává jako parametr do šablony. Druhá cesta očekává proměnnou typu `integer`, tedy celé číslo, se kterou se dále pracuje přímo uvnitř funkce, na kterou je na daná adresa namapována.

```
6 @app.route("/hello/<name>")
7 def say_hello(name):
8     return render_template("hello.html", name=name)
9
10 @app.route("/item/<int:id>")
11 def item_detail(id):
12     # např. získání produktu z databáze podle id
13     ...
14     # poté předáme produkt do šablony
15     return render_template("item_detail.html", item=item)
16
```

Obrázek 7 Ukázka dynamických URL adres

Další velmi užitečnou funkcí je `url_for()`, která slouží k dynamickému sestavení URL na základě názvu funkce. Jako první parametr přijímá název funkce, ke které chceme vygenerovat URL, a dále libovolný počet klíčových argumentů odpovídajících proměnným částem URL. Nepřiřazené klíčové argumenty se automaticky předají jako query parametry. [46]

3.2 Jinja2 šablony

Jak již bylo zmíněno, Flask používá jako šablonovací systém knihovnu Jinja2. Šablony v Jinja2 mohou obsahovat proměnné a výrazy, které jsou při vykreslení nahrazeny konkrétními hodnotami, a také řídicí konstrukce pro logiku šablon. Syntaxe Jinja2 vychází z Django a Pythonu. V šablonách se používají následující oddělovače:

- `{% ... %}` pro příkazy (např. podmínky, cykly)
- `{{ ... }}` pro výrazy, které se zobrazí ve výstupu
- `{# ... #}` pro komentáře, které se ve výstupu neobjeví [47]

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4  |   <title>Jinja2</title>
5  </head>
6  <body>
7  |   <h1>Ukázka Jinja2 šablon</h1>
8
9  |   {{ name }}
10
11 |   {# komentář #}
12
13 |   <ul>
14 |   |   {% for x in range(10) %}
15 |   |   |   <li>{{ x }}</li>
16 |   |   {% endfor %}
17 |   </ul>
18
19 </body>
20 </html>

```

Obrázek 8 Ukázka dokumentu s Jinja2 syntaxí

Na obrázku 8 je zobrazena ukázka HTML dokumentu, který využívá šablonovací systém Jinja2. Pokud šablona obdrží jako jeden z parametrů proměnnou `name` (jak bylo ukázáno i na obrázku 7), její hodnota se vloží na výstup na řádku 9, kde nahradí oddělovač. Na řádcích 14 až 16 je pak demonstrován jednoduchý cyklus `for` pomocí Jinja2 syntaxe.

Nejdůležitější vlastností šablonovacího systému Jinja2 je dědičnost šablon. Ta nám umožňuje vytvořit základní šablonu, která obsahuje společné prvky všech stránek – například hlavičku, navigaci nebo patičku. Tato šablona definuje také speciální bloky, které mohou být v podřízených šablonách přepsány. Díky tomu lze jednoduše vytvořit jednotné rozhraní webu, přičemž každá stránka definuje pouze svůj specifický obsah. Výsledkem je přehledný a znovupoužitelný kód. [47]

3.3 Session

Session ve Flasku je objekt, který funguje podobně jako klasický slovník v Pythonu. Využívá se k uchování dat mezi jednotlivými HTTP požadavky. Flask toho dosahuje pomocí podepsaných cookies, což znamená, že se data ukládají na straně klienta, ale jsou kryptograficky podepsaná, aby se zabránilo jejich neoprávněné úpravě. K použití session je nutné nastavit proměnnou `SECRET_KEY`, která slouží k vytvoření a ověření podpisu. Pomocí konfiguračních atributů lze také nastavit například dobu platnosti session. Objekt session je jednou z globálních proměnných, které jsou automaticky dostupné i v Jinja2 šablonách. [48]

3.4 Kešování

Ve webové aplikaci občas dochází k náročným výpočtům nebo načítání velkého množství dat, což může trvat delší dobu. Pokud víme, že se tyto data často nemění, můžeme zlepšit uživatelský zážitek tím, že data začneme kešovat, tedy dočasně ukládat do paměti pro opakované použití v blízké budoucnosti. V opačném případě by se při každém načtení stránky musely výpočty nebo dotazy provádět znovu, což by vedlo ke zbytečnému zatížení serveru a delším odezvám aplikace. [49]

Samotný Flask sice neobsahuje vestavěnou podporu pro kešování, ale lze jej jednoduše rozšířit pomocí knihovny `Flask-Caching`. Ukládání dat do keše zajišťuje instance třídy `Cache`. Základním typem je `SimpleCache`, která využívá běžný python slovník v paměti aplikace. Kromě toho lze využít i jiné techniky, například kešování do souborového systému, databáze nebo jiných externích nástrojů jako Redis či Memcached. U každého záznamu lze v cache nastavit dobu jeho platnosti – výchozí hodnota je 300 sekund (tedy 5 minut), ale lze ji upravit dle potřeby. [49][50]

4 PRINCIPY ČISTÉHO A LEHCE UDRŽITELNÉHO KÓDU

Žijeme v době, kdy je kladen důraz na rychlost – ať už při vývoji softwaru či uvedení finálního produktu na trh. Moderní vývoj často využívá agilní přístupy, jako je například SCRUM, které upřednostňují rychlé iterace před dokonalostí. S tímto přístupem však snadno zapomínáme na důležitost psaní čistého a snadno udržitelného kódu, který lze v případě potřeby rychle upravit či rozšířit. [51]

Každý vývojář se určitě někdy setkal se situací, kdy si při pohledu na vlastní kód uvědomil jeho nepřehlednost. Přesto ji často ignorujeme, protože „kód funguje“. V takových chvílích často odložíme refaktoring s tím, že se k němu vrátíme později. Jenže jak říká LeBlancův zákon: „*Později znamená nikdy*.“ [51][52]

S rostoucí nečitelností a složitostí kódu výrazně klesá produktivita vývoje – zejména při práci v týmu. Zatímco v počátečních fázích projektu může vývoj probíhat velmi rychle, později se i malé úpravy mohou stát rizikovými. V nepřehledně napsaném kódu totiž každá změna může neúmyslně narušit jinou, dosud funkční část systému. Žádná úprava pak už náhle není triviální. [51]

Naučit se psát čistý kód není snadné. Nestačí znát základní principy, psaní čistého kódu vyžaduje pravidelný trénink, zkušenosti a ochotu učit se z vlastních i cizích chyb. Důležitou roli hraje také pozorování, jak ostatní vývojáři své chyby řeší. Je však důležité si uvědomit, že dokonalý kód neexistuje. Naším cílem by ale mělo být se dokonalému kódu co nejvíce přiblížit. [51]

4.1 Srozumitelné názvy

S názvy se setkáváme v celém softwarovém vývoji. Pojmenováváme proměnné, funkce, parametry, třídy, balíčky, soubory s kódem, ale také složky, které tyto soubory obsahují. Právě proto, že je pojmenovávání tak časté, bychom mu měli věnovat dostatečnou pozornost a snažit se ho dělat co nejlépe. [51]

Názvy by měly jasně odhalit záměr. Správně zvolený název dokáže ušetřit více času, než kolik zabere jeho vymyšlení. Proměnné, funkce nebo třídy by měly být pojmenované tak, aby samy odpovídaly na základní otázky: proč existují, co dělají a jak se používají. Pokud je nutné ke jménu přidat komentář, název pravděpodobně dostatečně nevystihuje svůj účel. Obecně platí, že kratší názvy jsou vhodnější než příliš dlouhé názvy, neměli bychom tedy do pojmenování zahrnovat více informací, než je skutečně potřeba. [51]

Pro výběr vhodných názvů je důležitá schopnost dobře a výstižně popisovat. Mnoho vývojářů se bojí měnit nevhodné názvy ze strachu možných námitek ostatních. Přitom pokud zvolíme lepší pojmenování, kolegové to zpravidla ocení. V praxi si totiž názvy metod nebo tříd často nepamätujeme. Spoléháme na moderní nástroje, jako jsou našeptávače, a vybíráme z nabízených možností podle toho, co odpovídá našemu záměru. Přesnější a výstižnější pojmenování tedy může ostatním nejen pomoci, ale také je příjemně překvapit. [51]

4.2 Funkce

Funkce jsou jedním ze základních způsobů, jak organizovat kód v programu. Hlavní pravidlo při jejich vytváření zní: funkce by měly být malé. A druhé pravidlo říká, že by měly být ještě menší. Ideálně by každá funkce měla provádět pouze jednu konkrétní činnost. Její název by měl jasně vystihovat, co přesně daná funkce dělá. Není třeba se obávat delších názvů. Výstižný a srozumitelný název je vždy lepší než dlouhý komentář popisující její účel. Při pojmenovávání funkcí bychom měli dbát na konzistenci – používání stejných frází, podstatných jmen a sloves zlepšuje čitelnost a usnadňuje porozumění kódu ostatním vývojářům. [51]

Ideálním počtem argumentů funkce je nula. Pokud to není možné, stále akceptovatelná je i funkce s jedním či dvěma argumenty. Funkce se třemi a více vstupními argumenty by měly být dobře zdůvodněny. V ideálním případě bychom se měli snažit jejich počet snížit. Každý argument zvyšuje složitost funkce, neboť zavádí novou úroveň abstrakce, kterou je třeba při jejím použití chápat. Navíc komplikují i testování – čím více argumentů funkce přijímá, tím obtížnější je ověřit všechny možné kombinace těchto argumentů. Naopak funkce bez parametrů tento problém zcela eliminují. [51]

Zvláštní pozornost si zaslouží také výstupní argumenty. Běžně předáváme data do funkcí pomocí vstupních parametrů a očekáváme návratovou hodnotu jako výsledek. Pokud ale funkce vrací hodnoty skrze výstupní argumenty, může to být matoucí a proti očekávání vývojáře. Tento způsob by se proto měl používat jen ve výjimečných případech. [51]

Psaní funkcí je jako psaní jakéhokoliv jiného textu. Než začneme psát text, obvykle si nejprve ujasníme myšlenky a následně je upravujeme do čitelné podoby tak dlouho, dokud s výsledkem nejsme spokojeni. První návrh často bývá neorganizovaný, obsahuje neobratná spojení nebo opakování, a proto jej přepisujeme. Stejně tak i funkce může být na začátku zbytečně komplikovaná, příliš dlouhá, plná podmínek, cyklů nebo duplicitního kódu. Naším cílem by mělo být postupným refaktoringem – rozdělením na menší části, přejmenováním

proměnných nebo odstraněním duplicit – dosáhnout funkce, která bude přehledná, dobře pojmenovaná a bude odpovídat zásadám čistého kódu. Dokonalou funkci obvykle nikdo nenapíše hned napoprvé, a proto bychom se neměli obávat, pokud se to nepodaří ani nám. [51]

4.3 Komentáře

Jedním z hlavních důvodů, proč vývojáři využívají komentáře, je nečitelný nebo nejednoznačný kód. Místo toho, abychom komentářem vysvětlili, co se nám nepovedlo napsat jasně, bychom se měli pokusit kód přeformulovat tak, aby mluvil sám za sebe. Komentáře často zastupují naši neschopnost vyjádřit záměr přímo v kódu. Čím déle kód existuje, tím větší je šance, že je komentář k němu neaktuální. A není nic horšího než komentář, který neodpovídá realitě a vytváří tak falešná očekávání. Kód je jediným spolehlivým zdrojem pravdy. Pokud lze něco vyjádřit přímo v kódu, komentář by neměl být potřeba. Samozřejmě existují i dobré komentáře – například ty, které upozorňují na důležitý kontext, popisují důvody nestandardního řešení nebo zmiňují, co ještě zbývá dodělat (tzv. *TODO* komentáře). I přes to však platí, že nejlepší komentáře jsou často ty, které jsme díky dobře napsanému kódu nemuseli vůbec napsat. [51]

4.4 Formátování

Formátování kódu je důležité. Dokonce natolik, že si ho nemůžeme dovolit ignorovat. Je to způsob, jakým komunikujeme s ostatními vývojáři (a sami se sebou v budoucnosti), a právě proto by mělo patřit mezi naše nejvyšší priority. Někdo by mohl namítnout, že nejdůležitějším cílem vývojáře je, aby kód fungoval. Funkcionalita, kterou dnes vytvoříme, se však v budoucích verzích pravděpodobně změní. Zato čitelnost kódu bude mít zásadní vliv na všechny úpravy, které v systému během jeho životního cyklu provedeme. Styl psaní a čitelnost nastavují standard, který ovlivňuje udržitelnost a rozšiřitelnost kódu – i dlouho poté, co původní implementace zmizí. Náš styl a disciplína při psaní kódu přežívají. Náš kód často ne. [51]

4.5 Testování

Testy jsou klíčovou součástí při tvorbě čistého a lehce udržitelného kódu. Zejména jednotkové testy přináší do vývoje potřebnou flexibilitu. Pomáhají nám udržet kód přehledný, stabilní a znovupoužitelný. Důvod je jednoduchý – pokud máme testy, nebojíme se dělat změny. Bez testů se však každá úprava může stát potenciálním zdrojem chyb.

Nezáleží na tom, jak kvalitní a dobře strukturovaný kód je – bez testů nikdy nemáme jistotu, že úprava nezpůsobí nečekaný problém. S dobře napsanými testy tyto obavy výrazně mizí. Čím větší je pokrytí kódu testy, tím větší máme důvěru v prováděné změny. Klíčem k udržení čistého kódu je tak vlastnictví automatizované sady testů, která spolehlivě ověřuje chování produkčního kódu. [51]

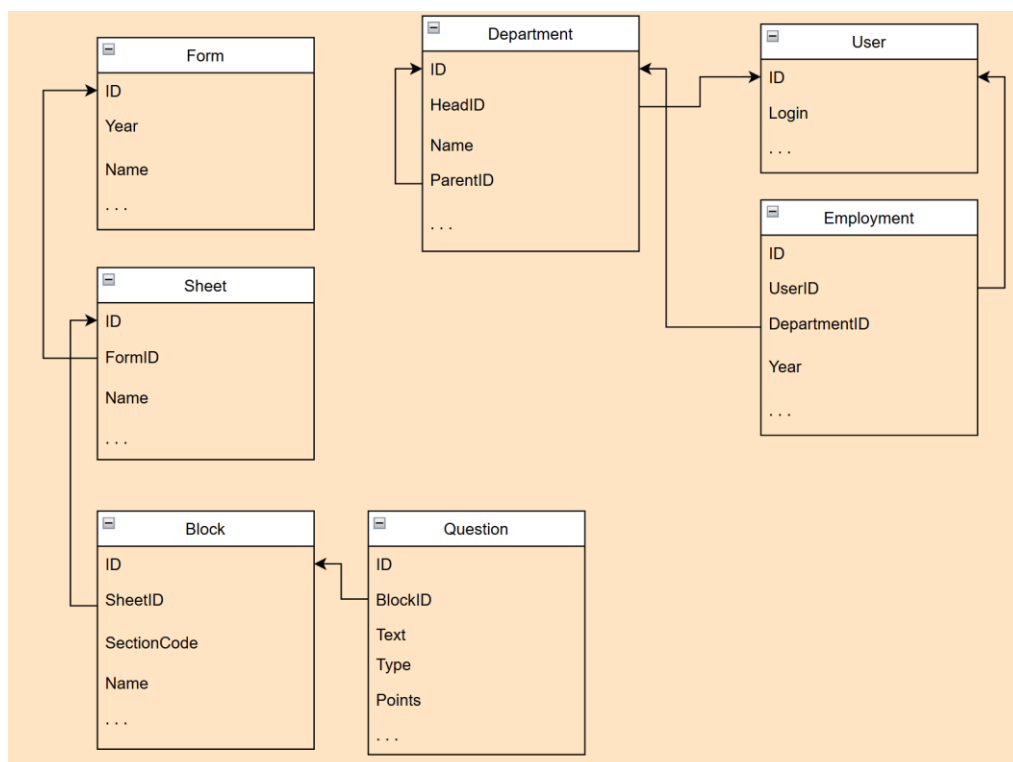
Testy by měly být stejně čisté jako produkční kód. Hlavním ukazatelem čistoty testu je jeho čitelnost. Čitelnost testu může být dokonce důležitější než čitelnost samotného kódu, který testujeme. Čitelný test poznáme stejně jako jakýkoliv jiný dobře napsaný kód: je jasný, jednoduchý a výstižně popisuje, co ověřuje. V ideálním případě by měl jednotkový test sdělit co nejvíce pomocí co nejméně slov. Testy by také měly dodržovat princip F.I.R.S.T., což je zkratka pro pět klíčových vlastností: Fast (rychlé), Independent (nezávislé), Repeatable (opakovatelně spustitelné), Self-Validating (testy by měly dokázat ověřit sami sebe) a Thorough (důkladné). [51][53]

II PRAKTICKÁ ČÁST

5 POPIS EXISTUJÍCÍHO SYSTÉMU

Akademičtí pracovníci na Univerzitě Tomáše Bati ve Zlíně vyplňují jednou ročně hodnotící formulář v systému IS HAP. Do tohoto formuláře zaznamenávají informace o svých pedagogických, tvůrčích, řídicích a dalších činnostech, které vykonali v uplynulém roce.

Formulář představuje jednu ze základních entit systému IS HAP. Každý formulář se skládá z listů, které dále obsahují bloky sdružující jednotlivé otázky. Klíčovými atributy každé otázky jsou její bodová hodnota a typ otázky, který určuje formát a očekávaný typ odpovědi. Přehled základních entit využívaných v systému IS HAP a jejich nejdůležitější atributy je znázorněn na obrázku 9.



Obrázek 9 Základní využívané entity v IS HAP

Další významnou entitou systému je pracoviště, které je reprezentováno jako stromová hierarchie. Kořen této struktury tvoří pracoviště Univerzity Tomáše Bati ve Zlíně, z něhož se větví jednotlivé fakulty a další organizační jednotky, až po nejnižší úroveň v podobě ústavů. Vizualizace této stromové struktury je uvedena na obrázku 10.

Neméně důležitou entitou jsou uživatelé a jejich pracovní úvazky. Každému uživateli je pro daný akademický rok přiřazen úvazek, který zahrnuje informace jako pracoviště,

pracovní pozici či velikost úvazku. Na základě těchto úvazků následně akademičtí pracovníci vyplňují hodnotící formuláře.

- Univerzita Tomáše Bati ve Zlíně
 - + Fakulta technologická
 - + Fakulta multimediálních komunikací
 - + Fakulta managementu a ekonomiky
 - + Rektorát
 - + Univerzitní institut
- Fakulta aplikované informatiky
 - Ústav matematiky
 - Ústav řízení procesů
 - Ústav automatizace a řídicí techniky
 - Ústav počítač. a komunikačních systémů
 - Ústav informatiky a umělé inteligence
 - Ústav bezpečnostního inženýrství
 - Ústav elektroniky a měření
- + Fakulta humanitních studií
- + Knihovna UTB
- + Fakulta logistiky a krizového řízení
- + Centrum polymerních systémů
- + Centrum bezp.,inf. a pokroč. technologií

Obrázek 10 Stromová struktura pracovišť

Univerzita si přizpůsobila systém bodového hodnocení jednotlivých aktivit s ohledem na jejich časovou náročnost. Od výstupní formy hodnocení, kterou IS HAP nabízí, tedy barevné škály doplněné slovním popisem, se upustilo. Místo toho se důraz klade zejména na bodové hodnocení v jednotlivých oblastech formuláře, případně na celkový počet získaných bodů. Nově je také kladen důraz na hodnocení vedoucího – některé činnosti totiž nelze bodově ohodnotit, a proto je nutné, aby jejich zhodnocení provedl nadřízený individuálně.

Za účelem efektivnějšího přístupu k výsledkům hodnocení si univerzita vytvořila vlastní systém, který umožňuje vedoucím jednotlivých pracovišť zobrazit bodové hodnocení jejich akademických pracovníků. Otázky jsou v hodnotícím formuláři rozděleny do sekcí dle toho, jaké činnosti se týkají a tohle rozdělení se zachovává i ve výsledných přehledech. Ukázkový výstup hodnocení v existujícím systému lze vidět na obrázku 11.

Body z IS HAP za fakultu: **Fakulta kouzel**
 Formulář pro hodnocení kal. rok: **2023-2021**

Number format: 1234,56 – decimal comma

Body z IS HAP

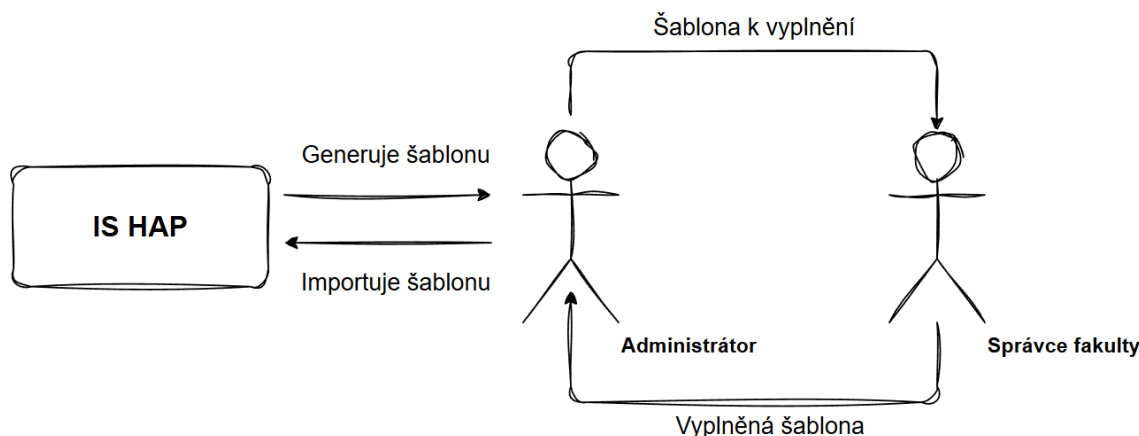
EmploymentID	Year	department	FirstName	LastName	name	fte	utb_section	score_fte	score
1	2023	Ústav černé magie	Petr	Brzlik	Ing. Petr Brzlik Ph.D.	1,00	2023 1a	630,19	630,19
2	2023	Ústav bylinkářství	Jan	Černý	doc. Ing. Jan Černý Ph.D.	1,00	2023 1b	58	58
2	2023	Ústav bylinkářství	Jan	Černý	doc. Ing. Jan Černý Ph.D.	1,00	2023 1a	580	580
3	2023	Ústav černé magie	Karel	Lakomý	Ing. Karel Lakomý	1,00	2023 2b	45	45
3	2023	Ústav černé magie	Karel	Lakomý	Ing. Karel Lakomý	1,00	2023 2c	300	300
4	2023	Ústav dějin čar a kouzel	Prokop	Hrozný	Ing. Prokop Hrozný Ph.D.	1,00	2023 1a	498	498
4	2023	Ústav dějin čar a kouzel	Prokop	Hrozný	Ing. Prokop Hrozný Ph.D.	1,00	2023 2a	110	110
4	2023	Ústav dějin čar a kouzel	Prokop	Hrozný	Ing. Prokop Hrozný Ph.D.	1,00	2023 2b	58	58
5	2023	Ústav astronomie	Jiří	Zelenka	prof. Mgr. Jiří Zelenka Ph.D.	1,00	2023 1b	28	28
5	2023	Ústav astronomie	Jiří	Zelenka	prof. Mgr. Jiří Zelenka Ph.D.	1,00	2023 2a	20	20
5	2023	Ústav astronomie	Jiří	Zelenka	prof. Mgr. Jiří Zelenka Ph.D.	1,00	2023 2b	186	186

Obrázek 11 Ukázka hodnocení v existujícím systému (zdroj: existující systém univerzity)

Další funkcionalitou systému je možnost zobrazení jednotlivých otázek v konkrétním hodnotícím formuláři. Systém IS HAP umožňuje hromadné předvyplnění libovolné otázky ve formuláři pomocí importu dat z Excelové šablony.

Pro správné spárování údajů využívá systém jednoznačné identifikátory otázek (Id), které jsou v šabloně uvedeny. Z tohoto důvodu je nutné, aby správce fakulty znal přesné Id otázky, kterou chce předvyplnit.

V aktuálně používaném systému však správce fakulty nemá možnost generovat šablonu s předvyplněnými údaji všech akademických pracovníků. Průběh práce s šablonami otázek je zobrazena na obrázku 12 níže.



Obrázek 12 Průběh práce s šablonami otázek v IS HAP

Šablony pro jednotlivé otázky generuje administrátor přímo v systému IS HAP a následně je rozesílá správcům fakult k doplnění. Po vyplnění šablony správcem fakulty je dokument zaslán zpět administrátorovi, který provádí import dat do systému.

Zobrazit otázky za UTB / fakultu: UTB ☐ Skrytí sloupce: ☐ list ☐ blok ☐ TextID ☐ zdroj

Formulář pro hodnocení kal. rok: UTB 2023

Zjištění ID otázky v IS HAP

počet otázek ve formuláři UTB 2023: 506 otázek

List	Blok	Fakulta / UTB	TextID	zdroj	Typ otázky (kís šablona)	%	id otázky	otázka
Pedagogická činnost...	Výuka	UTB	EDU.LectureHours.CzechSlovak+1Y		Seznam a popis	1	5827	Přednáška v prezenční / kombinované formě studia BSP, MSP
Pedagogická činnost...	Výuka	UTB			Seznam a popis	1	5828	Řízené konzultace v kombinované formě BSP, MSP
Pedagogická činnost...	Výuka	UTB	EDU.LectureHours.ForeignLang+1Y		Seznam a popis	1	5829	Přednáška ve světovém jazyce v BSP, MSP prezenční a kombinované formě
Pedagogická činnost...	Výuka	UTB			Seznam a popis	1	5830	Přednáška v DSP
Pedagogická činnost...	Výuka	UTB			Seznam a popis	1	5831	Přednáška v DSP ve světovém jazyce
Pedagogická činnost...	Výuka	UTB	EDU.LabAndSeminarHours.ForeignLang+1Y		Seznam a popis	1	5832	Seminář nebo cvičení ve světovém jazyce (včetně rozvrhovaných online hodin konzultací)
Pedagogická činnost...	Výuka	UTB	EDU.LabAndSeminarHours.CzechSlovak+1Y		Seznam a popis	1	5833	Semináře, cvičení, laboratoře, ateliéry, v prezenční a kombinované formě (včetně rozvrhovaných online hodin konzultací)
Pedagogická činnost...	Hodnocení kvality pedagogických činností	UTB			Hodnota ano/ne	1	6338	Prohlašuji, že jsem pečlivě zkontroloval/a veškerou svou výuku a ujistil/a se, že všechny informace odpovídají skutečnosti.
Pedagogická činnost...	Hodnocení kvality pedagogických činností	UTB			Počet a popis	1	5834	Hodnocení kvality pedagogických činností
Pedagogická činnost...	Hodnocení kvality pedagogických činností	UTB			Počet a popis	1	5835	Vytvoření 1 vzdělávacího videa o rozsahu nejméně 5 minut
Pedagogická činnost...	Hodnocení kvality pedagogických činností	UTB			Počet a popis	1	5836	Vytvoření 1 vzdělávacího podcastu o rozsahu nejméně 20 minut
Pedagogická činnost...	Hodnocení kvality pedagogických činností	UTB			Počet a popis	1	5837	Vytvoření 1 vzdělávacího animovaného spotu o délce nejméně 2 minuty
Pedagogická činnost...	Vedení studentů	UTB	EDU.DoctoralStudents.All.NonBonifiedYears.NonWorldLang+1Y		Seznam a popis	1	5838	Školitel doktoranda v 1. roce studia
Pedagogická činnost...	Vedení studentů	UTB	EDU.DoctoralStudents.All.BonifiedYears.NonWorldLang+1Y	STAG	Seznam a popis	1	5839	Školitel doktoranda ve 2. - 4. roce studia
Pedagogická činnost...	Vedení studentů	UTB	EDU.DoctoralStudents.All.NonBonifiedYears.WorldLang+1Y		Seznam a popis	1	5840	Školitel doktoranda v 1. roce studia ve světovém jazyce
Pedagogická činnost...	Vedení studentů	UTB	EDU.DoctoralStudents.All.BonifiedYears.WorldLang+1Y	STAG	Seznam a popis	1	5841	Školitel doktoranda ve 2. - 4. roce studia ve světovém jazyce
Pedagogická činnost...	Vedení studentů	UTB			Seznam a popis	1	5842	Vedení doktoranda ke zkoušce z komunikačních dovedností v cizím jazyce
Pedagogická činnost...	Vedení studentů	UTB	EDU.SupervisedBcThesisDefended+Y		Seznam a popis	1	5843	Obhájené bakalářské práce [NEBODOVÁNO]
Pedagogická činnost...	Vedení studentů	UTB	EDU.OtherSupervisedBcStudents+Y		Seznam a popis	1	5844	Ostatní vedené bakalářské práce [NEBODOVÁNO]

Obrázek 13 Otázky ve formuláři UTB 2023 (zdroj: existující systém univerzity)

Na obrázku 13 je vidět tabulka s otázkami z hodnotícího formuláře za rok 2023. Různé typy otázek ve formuláři vyžadují odlišné datové struktury, a proto mají i odlišné formáty šablon pro import. Každý typ otázky obsahuje specifické sloupce, které jsou pro správné zpracování dat relevantní.

Na obrázku 14 je zobrazena prázdná šablona pro otázky typu Počet a popis. Pro úspěšný import údajů do systému je nezbytné vyplnit hodnoty ve žlutě podbarvených sloupcích. Tyto sloupce představují povinné položky, bez jejichž vyplnění import selže. Ostatní sloupce jsou nepovinné a slouží pouze k doplnění kontextu nebo pro informativní účely. Data z těchto polí nejsou při importu dále zpracovávána.

	A	B	C	D	E	F	G	H	I	J	K	L
	id otázky	id otázky	Tituly před jménem	Jméno	Přijmení	Tituly za jménem	Login	Pracoviště	Stav formuláře	Počet	Studentské hodnocení	Popis
2												
3												
4												
5												
6												
7												
8												
9												
10												
11												
12												
13												
14												
15												
16												
17												
18												
19												
20												
21												
22												
23												
24												
25												
26												
27												
28												
29												
30												
31												
32												
33												
34												
35												
36												
37												
38												
39												
40												
41												
42												
43												
44												
45												
46												

Obrázek 14 Šablona v Excelu pro předvyplnění konkrétní otázky daty

6 FUNKČNÍ POŽADAVKY

Hlavním důvodem pro vytvoření nového uživatelského rozhraní je dostupnost potřebných dat ze systému IS HAP, avšak zároveň omezené možnosti přizpůsobení tohoto systému specifickým požadavkům univerzity.

Univerzita má specifické potřeby na práci s výsledky hodnocení – ať už se jedná o způsob jejich prezentace, možnosti exportu, nebo řízení přístupu k těmto datům na základě uživatelských rolí. Vzhledem k uzavřenosti systému IS HAP však není možné tyto požadavky realizovat přímo v jeho prostředí. Z tohoto důvodu bylo rozhodnuto o vytvoření samostatné nadstavby, která bude s existujícím systémem datově propojena, ale nabídne flexibilnější a uživatelsky přívětivější řešení.

Výsledná aplikace bude rozlišovat tři hlavní role uživatelů: akademické pracovníky, vedoucí pracovníky (případně správce pracoviště) a administrátory. Každá z těchto rolí bude mít přístup k sadě funkcionalit, které odpovídají jejich oprávněním a odpovědnostem v rámci systému.

Akademický pracovník bude mít možnost:

- zobrazit si body za své uzavřené hodnocení
- stáhnout výsledné hodnocení ve formátu XLS
- přistupovat k otázkám z hodnotících formulářů

Vedoucí pracovník (správce pracoviště) bude mít navíc k dispozici následující funkcionality:

- zobrazení hodnocení svého pracoviště
- možnost vytvářet a upravovat bodovací šablony
- možnost exportu předvyplněné šablony otázek

Administrátor bude mít rozšířené oprávnění na správu aplikace a dat, včetně:

- kopírování produkční databáze do testovací instance
- párování otázek z nových hodnotících formulářů
- přístupu k systému v roli jiného uživatele (např. pro testování či podporu)
- importu publikací pro konkrétní fakultu

7 IMPLEMENTACE WEBOVÉ APLIKACE

Nové uživatelské rozhraní bylo vytvořeno jako nadstavba nad databází stávajícího systému, která již obsahuje většinu dat potřebných pro implementaci požadovaných funkcionalit.

Pro vývoj aplikace byl zvolen webový framework Flask, který umožňuje rychlé a flexibilní vytváření backendové logiky. Na straně klienta byly použity základní webové technologie – HTML, CSS a JavaScript. Pro vizuální úpravu byl využit frontendový framework Bootstrap.

Výsledná aplikace byla vytvořena jako open-source a je dostupná pod licencí BSD 2–Clause. Kompletní text licence je uveden v příloze 1.

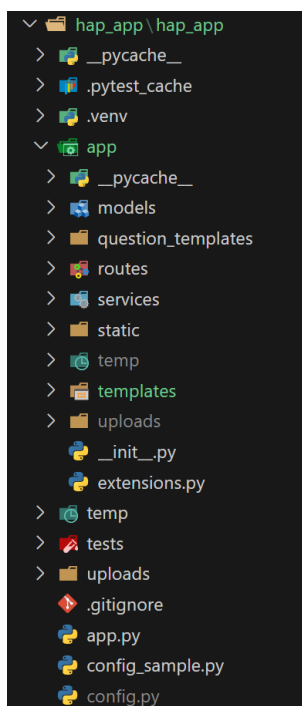
7.1 Struktura projektu

Webová aplikace je postavena na frameworku Flask a její adresářová struktura odpovídá běžné organizaci menších až středně velkých Flask projektů. Projekt je rozdělen do několika hlavních logických částí, které oddělují různé vrstvy aplikace:

- `app/routes/`
Složka obsahující soubor s definicemi rout (tras) aplikace. V rámci jednotlivých tras se řeší směrování požadavků, komunikace se šablonami nebo volání logiky z pomocných služeb.
- `app/templates/`
Složka obsahující HTML šablony využívané při renderování stránek.
- `app/models/`
Tato složka obsahuje definice datových modelů, které představují jednotlivé tabulky v databázi. V aplikaci jsou tyto modely implementovány pomocí knihovny SQLAlchemy, která umožňuje pracovat s databází objektově-orientovaným způsobem. Modely slouží především k načítání dat z MySQL databáze. Každý model reprezentuje jednu databázovou entitu – například uživatele, pracoviště, formulář, nebo otázku.
- `app/static/`
Obsahuje statické soubory jako jsou obrázky, kaskádové styly, nebo JavaScriptové soubory. Tato složka slouží k obsluze prostředků, které se za běhu aplikace nemění.

- `app/services/`
V této složce jsou umístěny soubory s aplikační logikou oddělenou od tras. Tím je zajištěna větší přehlednost a testovatelnost kódu.
- `config.py`
Konfigurační soubor obsahující nastavení aplikace, jako jsou přístupové údaje k databázi, nastavení session, kešování a další.
- `tests/`
Složka obsahující jednotkové testy, jako jsou testy rolí pro přístup do systému.
- `app/__init__.py`
Inicializační soubor aplikace. Obsahuje funkci `create_app`, ve které dochází k vytvoření instance Flask aplikace, načtení potřebné konfigurace ze souboru `config.py`, připojení k databázi pomocí SQLAlchemy a inicializaci dalších rozšíření. V rámci této funkce jsou rovněž registrovány jednotlivé trasy, čímž se aplikace připraví ke spuštění.
- `app.py`
Hlavní spouštěcí skript aplikace.

Celá adresářová struktura projektu je vidět na obrázku 15.



Obrázek 15 Adresářová struktura projektu

7.2 Práce s databázemi

Pro práci s daty jsou v aplikaci využívány dva systémy řízení báze dat – MySQL a PostgreSQL. MySQL slouží primárně pro práci s existujícími daty ze systému IS HAP. Pro komunikaci s touto databází je využita knihovna SQLAlchemy, která pomocí objektově-relačního mapování umožňuje definovat datové modely a provádět nad nimi běžné databázové operace bez nutnosti psát přímé SQL dotazy.

Hlavním důvodem k nasazení PostgreSQL v původním systému byl jednoduchý přístup k datům z různých zdrojů, zejména z externích systémů řízení báze dat. V rámci nově vyvíjené aplikace je PostgreSQL dále využíván pro realizaci komplexních databázových dotazů, u nichž objektově-relační mapování naráží na limity z hlediska výkonu či flexibility. Mezi typické příklady patří náročné výpočty bodového hodnocení, agregace, práce s databázovými pohledy nebo složitější vazby mezi entitami. V databázovém prostředí PostgreSQL bylo rovněž vytvořeno nové schéma, které obsahuje entity nezbytné pro implementaci funkcionality bodovacích šablon. Jejich popis a význam je podrobněji rozebrán v jedné z následujících kapitol.

7.3 Implementace jednotlivých funkcionalit

Do aplikace se uživatelé přihlašují pomocí uživatelského jména, které je shodné s přihlašovacími údaji do ostatních systémů Univerzity Tomáše Bati ve Zlíně. Po úspěšném přihlášení je uživateli zpřístupněno navigační menu s jednotlivými funkcionalitami aplikace. Tyto funkcionality jsou implementovány ve formě samostatných „karet“ v rámci uživatelského rozhraní.

Dostupnost jednotlivých částí systému je řízena na základě uživatelských rolí, které jsou definovány v systému IS HAP. Pro definici a přiřazování těchto rolí je i nadále využíváno původní rozhraní tohoto systému. Každý uživatel má na základě své role přístup pouze k těm funkcím, které odpovídají jeho oprávněním. Přehled dostupných funkcionalit pro jednotlivé role je uveden v kapitole 6, zatímco jejich konkrétní implementace je popsána v následujících podkapitolách této kapitoly.

7.3.1 Osobní hodnocení

Jednou z klíčových funkcionalit nového systému je možnost zobrazení osobního hodnocení pro jednotlivé akademické pracovníky. V existujícím systému, který si univerzita vytvořila

a je popsán v kapitole 5, je přístup k hodnocení umožněn primárně vedoucím pracovníkům, kteří vidí výsledky všech akademiků na svém pracovišti.

Samotní akademičtí pracovníci mají sice možnost zobrazit si své celkové hodnocení přímo v systému IS HAP, nicméně toto zobrazení neumožňuje detailní rozlišení podle jednotlivých oblastí (listů nebo bloků) formuláře, případně zobrazení bodového ohodnocení za konkrétní otázky.

Z tohoto důvodu byla do nové aplikace doplněna funkcionality, která umožňuje akademickým pracovníkům přehledně zobrazit bodové hodnocení za jednotlivé části formuláře i za jednotlivé otázky. Tím získávají lepší přehled o svém výkonu a mohou přesněji identifikovat, ve které oblasti dosáhli nejvyššího nebo naopak nejnižšího bodového zisku.

List	Body
Pedagogická činnost - Univerzitní	990.41
Pedagogická činnost - Fakultní	226.8
Tvůrčí činnosti - Univerzitní	340.0
Tvůrčí činnosti - Fakultní	225.0
Další činnosti - Univerzitní	126.0
Další činnosti - Fakultní	15.0

Obrázek 16 Osobní hodnocení

Na obrázku 16 je zobrazena ukázka osobního hodnocení akademického pracovníka. Uživatel si nejdříve vybírá své pracoviště z nabídky těch, na kterých měl alespoň jednou v minulosti evidován pracovní úvazek. Následně si volí bodovací šablonu, podle které má být výpočet proveden. Různé šablony umožňují přiřazení odlišných bodových hodnot jednotlivým otázkám, čímž se pracovník může nechat ohodnotit například podle konkrétní hodnotící směrnice univerzity. Dále si uživatel volí způsob agregace výsledného hodnocení. Nejnižší úrovní jsou otázky formuláře, další možností je seskupení do bloků, které sdružují související otázky (např. blok *Výuka*). Nejvyšší úrovní jsou listy formuláře, které zahrnují

více bloků do větších celků, typicky podle charakteru činnosti (např. list *Pedagogická činnost – Fakultní*).

Po stisknutí tlačítka Zobrazit hodnocení dojde k provedení dotazu na databázi a následnému zobrazení výsledných dat v tabulce. Pokud jsou data k dispozici, je zároveň zpřístupněna možnost exportu výsledků do Excelového souboru.

Data potřebná pro výpočet hodnocení jsou získávána pomocí komplexního SQL dotazu nad databází PostgreSQL. Tento dotaz využívá vícestupňovou strukturu prostřednictvím tzv. Common Table Expressions (CTE), které umožňují modulární sestavování dotazů. CTE propojují tabulky formulářů, otázek, odpovědí a údajů o konkrétním akademickém pracovníkovi. Součástí dotazu jsou i výpočty výsledného bodového ohodnocení, přičemž je zohledněn přepočít podle velikosti pracovního úvazku. Tento přístup umožňuje efektivní agregaci a přehlednou prezentaci detailního hodnocení pracovníka dle zvolené šablony hodnocení.

7.3.2 Otázky z formulářů

Funkcionalita prohlížení otázek z formulářů je implementována podobně jako v existujícím systému a je dostupná i bez přihlášení. Umožňuje uživatelům vyhledávat otázky v rámci konkrétního formuláře, což je zejména přínosné pro fakultní správce při vyplňování šablon otázek. Tyto šablony (blíže popsány v kapitole 7.3.5) využívají jednoznačné identifikátory otázek, které lze prostřednictvím této funkcionality snadno dohledat.



Univerzita Tomáše Bati ve Zlíně

Hodnocení akademických pracovníků UTB

Přihlášený uživatel: Nepřihlášen

HAP Otázky

Zobrazit otázky za UTB / fakultu:

FAI

Formulář pro hodnocení kal. rok:

UTB 2023

Skrýt sloupce:

List

☐

Blok

☒

Typ Otázky

☐

Počet otázek celkem: 52

List	Blok	Typ Otázky	ID Otázky	Text Otázky
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5873	FAI: Vedení bakalářské práce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5874	FAI: Vedení diplomové práce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5875	FAI: Vedení bakalářské práce v anglickém jazyce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5876	FAI: Vedení diplomové práce v anglickém jazyce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5877	FAI: Oponování bakalářské práce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5878	FAI: Oponování diplomové práce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5879	FAI: Oponování bakalářské práce v anglickém jazyce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5880	FAI: Oponování diplomové práce v anglickém jazyce
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5881	FAI: Školitel / Konzultant doktoranda studujícího na UTB, který úspěšně ukončil studium ve standardní době studia + 1 rok
Pedagogická činnost - Fakultní	FAI: Vedení studentů	Seznam a popis	5882	FAI: Školitel / Konzultant doktoranda studujícího na UTB, který úspěšně ukončil studium po delší době než standardní doba studia + 1 rok
Pedagogická činnost - Fakultní	FAI: Další pedagogické činnosti	Seznam a popis	5883	FAI: Člen komise při kontrolních dnech BP/DP
Pedagogická činnost - Fakultní	FAI: Další pedagogické činnosti	Seznam a popis	5884	FAI: Cvičení, které je realizováno na počítačích najednou pro více než 16 studentů
Pedagogická činnost - Fakultní	FAI: Další pedagogické činnosti	Seznam a popis	5885	FAI: Vedení práce v soutěžích STOČ apod.

Obrázek 17 Otázky z formulářů

Na obrázku 17 jsou zobrazeny otázky z hodnotícího formuláře pro rok 2023, konkrétně otázky určené primárně pro Fakultu aplikované informatiky. Uživatel má možnost zobrazit buď celý univerzitní formulář, nebo pouze otázky relevantní pro konkrétní fakultu, jak je tomu v tomto případě. Po výběru požadovaného formuláře lze s tabulkou otázek dále pracovat, například skrývat sloupce, které nejsou pro aktuální hledání důležité. Každá otázka má navíc svou vlastní kotvu, díky níž je možné na konkrétní otázku vygenerovat přímý odkaz a tento odkaz následně sdílet.

7.3.3 Hodnocení pracoviště

Hodnocení pracoviště je funkcionality určená pro vedoucí pracovníky. Tato funkcionality prošla v novém systému výrazným vylepšením oproti původní verzi. Zatímco ve stávajícím systému je bodové hodnocení agregováno podle kódových označení jednotlivých bloků (viz sloupec *utb_section* na obrázku 11), která jsou pro běžného uživatele bez širší znalosti systému prakticky nesrozumitelná, v nové aplikaci je sjednocení založeno na názvech celků, které otázky logicky spojují. Podobně jako v případě osobního hodnocení si tak uživatel může zobrazit výsledky hodnocení všech akademiků na pracovišti buď dle bloků, nebo dle listů formuláře. Detailní zobrazení na úrovni jednotlivých otázek zde záměrně chybí, protože by kvůli množství dat vedlo k nepřehledné tabulce a snížení informační hodnoty výstupu. Ukázka bodového hodnocení na pracovišti je zobrazena na obrázku 18.

Jméno zaměstnance	Pracoviště	Úvazek	Ped...	Ped...	Tvů...	Tvů...	Rid...	Rid...	Dal...	Dal...	Body celkem
Ing. Karolína Bílá	Ústav přeměňování	1.00	800.0	105.45	0	0	0	0	0	0	905.45
Ing. Robert Brodávka Ph.D.	Ústav přeměňování	1.00	930.0	318.4	0	0	0	0	0	0	1248.4
Ing. Roman Koutný	Ústav přeměňování	1.00	1000.12	0	0	5.0	0	0	0	0	1005.12
Ing. Jaromír Kramný Ph.D.	Ústav přeměňování	1.00	710.28	85.0	0	5.0	0	0	0	0	800.28
Ing. Jan Martinek Ph.D.	Ústav přeměňování	1.00	673.2	30.0	0	0	0	0	50.5	0	750.8
Ing. Antonín Pravý Ph.D.	Ústav přeměňování	1.00	660.0	408.0	0	226.0	0	0	0	0	1294.0
doc. Ing. Radomír Prokop	Ústav přeměňování	1.00	500.0	0	5	0	500.0	95.0	0	0	1100.0
Ing. Tomáš Svatý Ph.D.	Ústav přeměňování	1.00	1010.0	90.0	0	24.0	500.0	0	30.0	5.0	1659.0
doc. Ing. Martin Šmutný Ph.D.	Ústav přeměňování	1.00	905.0	0	1405.0	95.0	0	0	95.0	0	2500.0
Ing. Radek Trubec Ph.D.	Ústav přeměňování	1.00	900.46	200.0	1412.0	486.0	0	0	60.0	0	3058.46
Ing. Pavel Volný.	Ústav přeměňování	1.00	954.54	581.9	0.0	65.0	45.0	20.0	0	18.0	1684.44

Obrázek 18 Hodnocení pracoviště

Vedoucí pracovníci si v rámci této funkcionality volí pracoviště, pro které chtějí zobrazit hodnocení. Na výběr mají však pouze z těch pracovišť, ke kterým mají přístup na základě přidělených oprávnění. Následně, podobně jako v osobním hodnocení, vybírají bodovací šablonu a způsob sjednocení hodnocení (dle bloků nebo listů). Výsledky hodnocení lze stejně jako u osobního hodnocení exportovat do Excelového souboru. Oproti osobnímu hodnocení však mají vedoucí pracovníci navíc možnost vyhledávat a filtrovat v tabulce zaměstnance dle jména nebo dle pracoviště, na kterém měli v daném roce úvazek.

7.3.4 Bodovací šablony

Bodovací šablony představují jednu z klíčových novinek, která v původním systému zcela chyběla. V existujícím systému je možné vypočítat hodnocení pouze pro každý konkrétní rok zvlášť, přičemž každá otázka ve formuláři má pevně stanovenou bodovou hodnotu bez možnosti úprav.

Nový systém díky zavedení bodovacích šablon umožňuje:

- propojení bodování napříč více formuláři
- nastavení bodového ohodnocení pro jednotlivé otázky podle potřeby
- flexibilní výpočet hodnocení dle různých pravidel a směrnic

Díky této funkcionalitě si mohou akademičtí pracovníci spočítat své osobní hodnocení různými způsoby – například podle konkrétní univerzitní metodiky, fakultní směrnice nebo jiného hodnotícího přístupu. Stejnou flexibilitu získávají také vedoucí pracovníci při hodnocení svých pracovišť.

Aby bylo možné bodovací šablony vůbec vytvářet, bylo nutné nejprve provést spárování otázek napříč jednotlivými formuláři a roky. Každý hodnotící formulář totiž přiděluje svým otázkám unikátní identifikátor, a proto nelze meziročně rozpoznat, zda se jedná o tutéž otázku.

I v případě, že je konkrétní otázka obsažena ve více formulářích z různých let, má v každém z nich jiný identifikátor. Pro potřeby jednotného bodového ohodnocení tedy bylo nezbytné provést propojení otázek, které mají stejný význam, ale pocházejí z rozdílných formulářů.

Z tohoto důvodu byla v databázi vytvořena nová entita, která slouží k evidenci spárovaných otázek napříč hodnotícími formuláři z odlišných let. Párování otázek bylo

provedeno na základě jejich textového znění. Tato entita je realizována tabulkou s názvem `question_years`, která je navržena podle principu EAV (Entity-Attribute-Value) modelu. Díky tomuto flexibilnímu přístupu je možné snadno přidávat nové záznamy vždy, když vznikne nový formulář a je potřeba jeho otázky napárovat na odpovídající otázky z předchozích let.

Table: `question_years`

[Select data](#) [Show structure](#)

Column	Type	Comment
ID	integer	
FormID	integer	
QuestionID	integer	

Indexes

PRIMARY ID, FormID

Obrázek 19 Struktura tabulky `question_years`

Jak lze vidět na obrázku 19, entitou v rámci EAV modelu je v tomto případě samotný identifikátor propojení (*ID*). Atribut představuje identifikátor formuláře, ze kterého daná otázka pochází a hodnotou je následně identifikátor konkrétní otázky.

Kombinace identifikátoru propojení a identifikátoru formuláře tvoří primární klíč této tabulky. Tím je zajištěno, že v rámci jednoho propojení (tj. jednoho záznamu typu entita) je vždy obsažena nejvýše jedna otázka z každého formuláře. Tento přístup zároveň umožňuje jednoduše rozšiřovat stávající propojení o nové formuláře bez nutnosti měnit databázovou strukturu.

Druhá entita, která byla vytvořena, popisuje samotnou šablonu hodnocení. Na obrázku 20 jsou vidět jednotlivé atributy této entity. Atribut *ID* představuje identifikátor bodovací šablony, *QuestionYearsID* je cizí klíč odkazující na entitu `question_years`, která zajišťuje propojení otázek napříč formuláři. Atribut *Year* označuje konkrétní rok formuláře, ke kterému se bodová hodnota vztahuje a *Points* určuje samotnou hodnotu bodového ohodnocení, která bude dané otázce přidělena.

Table: evaluation_template

[Select data](#) [Show structure](#) [Alter table](#)

Column	Type	Comment
ID	integer	
QuestionYearID	integer	
Year	integer	
Points	numeric	

Indexes

PRIMARY ID, QuestionYearID, Year

Obrázek 20 Struktura tabulky evaluation_template

Poslední novou entitou, která byla vytvořena v souvislosti s bodovacími šablonami, je pomocná entita `evaluation_template_meta`. Tato entita slouží k uchování základních informací o jednotlivých šablonách, jako je jejich název, určení pro konkrétní pracoviště a informace o viditelnosti šablony – tedy zda je určena pouze pro výpočet hodnocení na úrovni pracoviště, nebo ji mohou využívat i akademičtí pracovníci pro výpočet osobního hodnocení. Strukturu této tabulky je možné vidět na obrázku 21.

Table: evaluation_template_meta

[Select data](#) [Show structure](#) [Alter table](#) [New item](#)

Column	Type	Comment
ID	integer	
Name	character varying(255)	
DepartmentID	integer	
VisibleForAcademic	boolean	

Indexes

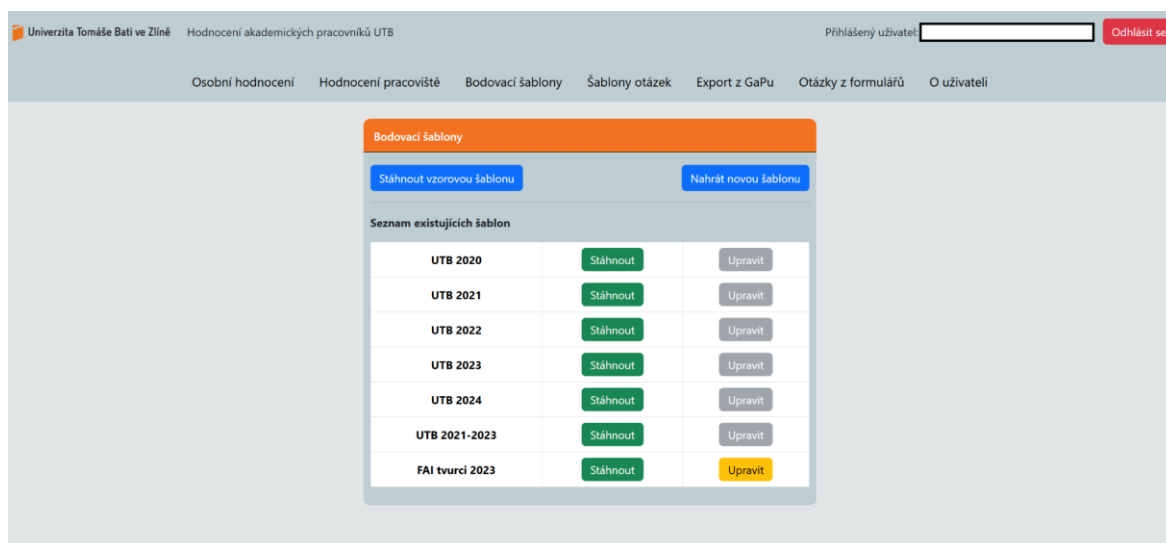
PRIMARY ID
UNIQUE Name

Obrázek 21 Struktura tabulky evaluation_template_meta

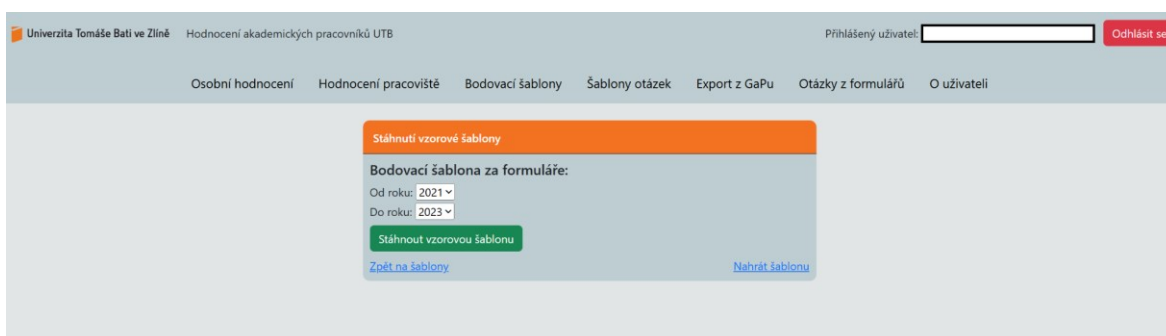
Vedoucí pracovníci mají oprávnění vytvářet nové bodovací šablony a upravovat ty, které jsou určeny pro pracoviště, ke kterým mají přístup. Mají rovněž možnost stáhnout si již existující šablony, což zjednodušuje tvorbu nové šablony v případě, kdy nová šablona vychází z šablony již dříve vytvořené. Uživatelé s rolí administrátora mají přístup k úpravám všech šablon bez omezení, a navíc mají možnost existující šablony mazat.

Na obrázku 22 je zobrazena stránka s přehledem bodovacích šablon. Vedoucí pracovníci zde mají možnost stáhnout si vzorovou šablonu pro konkrétní roky (viz obrázek 23 a příloha 2). Tato šablona obsahuje předvyplněné bodové hodnoty jednotlivých otázek tak, jak jsou standartně nastaveny v databázi. Vedoucí pracovník může tyto hodnoty upravit

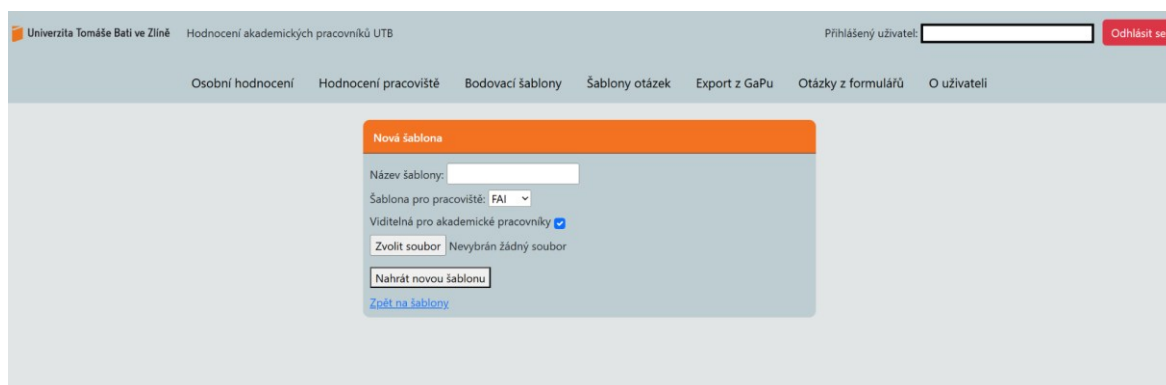
podle vlastních potřeb a vytvořit tak novou bodovací šablonu, kterou následně nahraje zpět do systému (viz obrázek 24).



Obrázek 22 Přehled existujících bodovacích šablon



Obrázek 23 Stáhnutí vzorové šablony



Obrázek 24 Nahrání nové šablony do systému

7.3.5 Šablony otázek

Díky šablonám otázek mohou vedoucí pracovníci předvyplnit otázky hodnotícího formuláře za všechny akademické pracovníky na vybraném pracovišti. Typicky se jedná o otázky

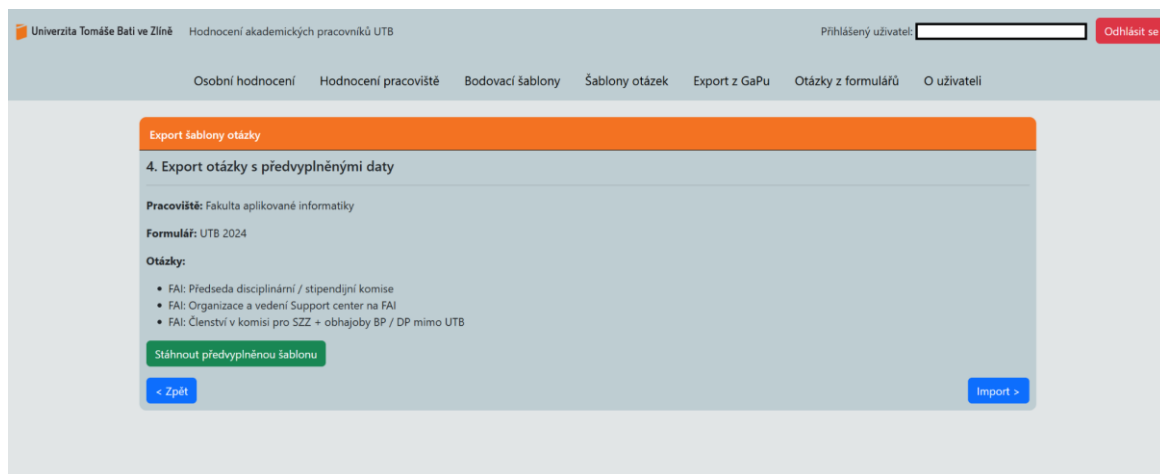
formuláře, ke kterým mají relevantní informace právě vedoucí pracovišť. Každý nový formulář je nejprve automaticky předvyplněn daty z dalších univerzitních systémů, jako jsou STAG nebo OBD. Následně dochází k předvyplnění vybraných otázek vedoucími pracovníky. Teprve poté je formulář zpřístupněn samotným akademickým pracovníkům, kteří doplňují zbývající informace.

V původním systému vytvořeném univerzitou tato funkcionality dostupná nebyla. Předvyplněnou šablonu pro konkrétní otázky musel generovat v systému IS HAP administrátor, který ji následně je rozesílal správcům fakult k doplnění. Ti šablonu po vyplnění zasílali zpět administrátorovi, který ji následně importoval opět přímo v IS HAP.

Funkcionality šablon otázek v nové aplikaci umožňuje vedoucím pracovníkům vygenerovat šablonu otázek se základními údaji všech akademiků na vybraném pracovišti. Vedoucí pracovník následně doplní pouze chybějící data, šablonu uloží a nahraje zpět do webové aplikace. Po úspěšné kontrole, že šablona neobsahuje žádné chyby, je umožněno její odeslání administrátorovi spolu s krátkou zprávou a žádostí o import.

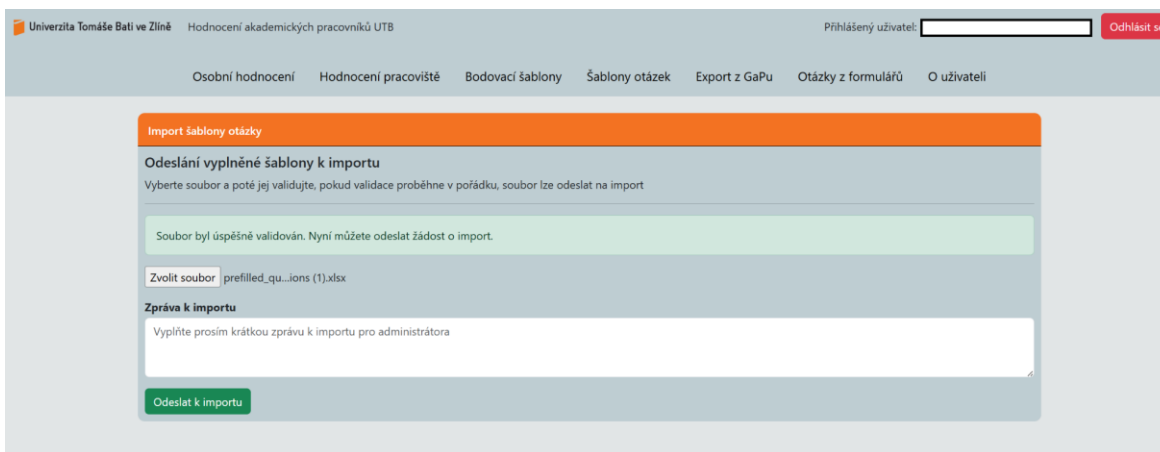
Samotný export šablony otázek je rozdělen do čtyř kroků. V prvním kroku vedoucí pracovník vybírá rok a pracoviště, pro které chce šablonu vytvořit. Následně volí konkrétní formulář a typ otázky, přičemž různé typy otázek vyžadují odlišnou strukturu šablony – liší se požadovanými sloupci s daty. Výběr typu otázky zároveň umožňuje zahrnout více otázek do jedné exportované šablony, čímž se výrazně zjednodušuje proces, který dříve vyžadoval vyplňování každé otázky zvlášť a práci s několika šablonami.

Ve třetím kroku dochází k výběru konkrétních otázek, u nichž má být šablona vyplněna. V posledním kroku je uživateli zobrazen souhrn zvoleného pracoviště, formuláře a vybraných otázek, aby si mohl zkontrolovat správnost výběru. Na stejné stránce (viz obrázek 25) se nachází také tlačítko pro export Excelové šablony s předvyplněnými základními údaji všech pracovníků vybraného pracoviště.



Obrázek 25 Šablony otázek (poslední krok exportu)

Po vyplnění údajů v šabloně mají vedoucí pracovníci možnost ověřit správnost vyplněných dat pomocí nástroje pro kontrolu souboru. V případě, že validace proběhne úspěšně, je možné šablonu odeslat administrátorovi k importu do systému. Součástí odeslání je také krátká zpráva, která vysvětluje účel nebo kontext daného souboru (viz obrázek 26). Po odeslání šablony je administrátor informován prostřednictvím e-mailu, který obsahuje danou zprávu, soubor k importu jako přílohu a odkaz na stránku, kde může samotný import provést.



Obrázek 26 Import šablony otázek po úspěšné validaci

7.3.6 Kopírování produkční databáze

Funkcionalita kopírování produkční databáze *hap* do její testovací instance je dostupná pro administrátory již v existujícím systému. Testovací instance slouží k bezpečnému ověřování nových funkcionalit, testování importů dat, nebo k ladění chyb, aniž by bylo ohroženo produkční prostředí s reálnými daty. Z hlediska implementace se jedná o sadu příkazů, které

jsou prostřednictvím připraveného skriptu automaticky spuštěny v příkazovém řádku a zajišťují přenos aktuálního stavu databáze do testovacího prostředí.

7.3.7 Párování otázek z nového formuláře

Pro zajištění správné funkce bodovacích šablon je nezbytné udržovat aktuální párování otázek napříč jednotlivými roky. Při každém vytvoření nového hodnotícího formuláře je nutné aktualizovat existující párování otázek tak, aby byly nové otázky správně propojeny s těmi, které již byly spárovány v minulosti. Tento proces zajišťuje konzistenci dat napříč lety a umožňuje bezproblémové použití bodovacích šablon. Přístup k této funkcionalitě mají výhradně uživatelé s rolí administrátora.

Při provádění nového párování otázek se nejprve z databáze získá identifikátor nejnovějšího formuláře. Následně se ověří, zda pro tento formulář již neexistuje záznam v tabulce `question_years`, která uchovává informace o spárovaných otázkách. Poté dochází k samotnému porovnání otázek z nového formuláře s již existujícím párováním, na jehož základě jsou nové otázky doplněny do databáze. Jak již bylo zmíněno v kapitole 7.3.4, díky využití EAV modelu v tabulce `question_years` je tento proces velmi flexibilní – nové záznamy je možné přidávat bez potřeby zásahu do struktury databáze. Tím je zajištěna dlouhodobá udržitelnost a rozšiřitelnost celého systému.

7.3.8 Import publikací

Poslední z funkcionalit určených výhradně pro administrátory je import publikací pro vybranou fakultu. Tento import je realizován prostřednictvím jednoduchého shell příkazu, který přijímá několik parametrů potřebných pro jednoznačnou identifikaci dat určených k importu. Prozatím je tato funkcionalita dostupná ve formě základního formuláře, jehož prostřednictvím je možné požadované parametry pohodlně zvolit. Finální implementaci této funkcionality provede dle potřeb správce HAPu.

8 ZABEZPEČENÍ APLIKACE

Bezpečnost a integrita dat hrají v rámci navržené aplikace klíčovou roli. Aplikace pracuje s informacemi, jejichž neoprávněný přístup či nechtěná modifikace by mohly negativně ovlivnit výsledné hodnocení akademických pracovníků a narušit důvěryhodnost celého procesu. Z tohoto důvodu byly do systému implementovány základní prvky aplikační bezpečnosti – řízení přístupu na základě uživatelských rolí a validace dat při práci se šablonami otázek.

Aplikace pracuje primárně s existujícími daty z IS HAP, přičemž většina funkcionalit slouží pouze k jejich zobrazení, výpočtu a exportu. Tím se výrazně snižuje riziko nechtěného narušení konzistence databáze.

Pro bezpečný přístup k relační databázi spravované systémem MySQL, která obsahuje data pro většinu těchto funkcionalit, je využívána knihovna SQLAlchemy. Ta díky využití ORM přístupu a parametrizace dotazů zajišťuje ochranu proti útokům typu SQL injection, neboť vstupní data nejsou do SQL dotazů vkládána přímo.

Z hlediska bezpečnosti je systém dále chráněn proti útokům typu XSS (Cross-site scripting), a to automatickým escapováním dynamicky vkládaných dat prostřednictvím šablonovacího systému Jinja2. Tím se zamezuje spuštění škodlivého skriptu v prohlížeči uživatele.

8.1 Role a řízení přístupu

Aplikace využívá roli uživatele jako základní prvek pro řízení přístupu. Role uživatele je určena na základě záznamu v tabulce `userroles` v databázi *hap*. Uživatelské role určují, k jakým částem systému má konkrétní uživatel přístup. Přehled jednotlivých rolí a jejich oprávnění vychází z funkčních požadavků, které jsou popsány v kapitole 6.

Na obrázku 27 je zobrazena tabulka `userroles`, která slouží k propojení uživatelů a jejich přiřazených rolí. Každý záznam v této tabulce obsahuje také informaci o období platnosti dané role – konkrétně datum, od kdy a do kdy je role pro uživatele aktivní. Hodnota *NULL* v některém z těchto polí značí, že dané datum není omezeno. Tabulka obsahuje pouze záznamy pro uživatele s vyššími rolemi, jako je administrátor, vedoucí pracovník nebo správce pracoviště. Roli akademického pracovníka mají všichni uživatelé systému automaticky přiřazenou a není ji tedy nutné evidovat zvlášť.

☐	Modify	ID	UserID	RoleID	DepartmentID	Comment	ModificationDate	ModifiedBy	ValidFrom	ValidUntil
☐	edit	5		1	NULL	NULL	NULL	NULL	NULL	NULL
☐	edit	6		1	NULL		2018-10-12 02:50:07	993	NULL	NULL
☐	edit	12		7	1251	//kvůli vyplnění kariérních plánů	2024-09-17 11:03:22	24453	NULL	NULL
☐	edit	13		7	1251		2020-03-26 14:08:37	993	NULL	NULL
☐	edit	14		1	NULL		2019-06-04 13:22:31	993	NULL	NULL
☐	edit	15		7	1276		2022-04-06 16:34:59	23800	2021-09-01 00:00:00	NULL
☐	edit	16		7	401		2022-04-06 16:35:08	23800	2021-09-01 00:00:00	NULL
☐	edit	17		7	651		2022-04-06 16:36:16	23800	2021-09-03 00:00:00	NULL
☐	edit	25		7	1076		2022-04-06 16:35:38	23800	2021-10-08 00:00:00	NULL
☐	edit	27		7	4	- vkládání projektů do HAPu	2024-01-03 14:10:50	24453	2022-05-03 00:00:00	NULL
☐	edit	39		7	98		2021-05-11 14:06:46	23800	NULL	NULL
☐	edit	40		7	98		2021-05-11 14:07:28	23800	NULL	NULL
☐	edit	44		7	3		2023-10-05 15:45:26	24453	2021-08-11 00:00:00	NULL
☐	edit	71		7	2		2022-04-07 09:12:51	23800	2022-04-01 00:00:00	NULL
☐	edit	78		1	NULL		2022-07-25 09:08:27	993	NULL	NULL
☐	edit	109		7	1		2025-02-12 06:47:29	24453	2025-02-01 00:00:00	2025-12-31 00:00:00
☐	edit	111		5	1451		2023-09-06 11:08:38	24453	NULL	2023-09-30 00:00:00
☐	edit	112		5	1451		2024-09-05 12:10:26	24453	NULL	2024-10-31 00:00:00
☐	edit	113		5	46		2023-09-06 11:12:38	24453	NULL	2023-09-30 00:00:00
☐	edit	114		5	46		2023-09-06 11:13:07	24453	NULL	2023-09-30 00:00:00
☐	edit	115		5	1351		2023-09-06 11:13:47	24453	NULL	2023-09-30 00:00:00
☐	edit	116		5	1351		2023-09-06 11:14:20	24453	NULL	2023-09-30 00:00:00
☐	edit	117		5	877		2023-09-06 11:14:45	24453	NULL	2023-09-30 00:00:00
☐	edit	118		5	877		2024-09-05 12:12:09	24453	NULL	2024-10-31 00:00:00
☐	edit	119		5	826		2023-09-06 11:15:42	24453	NULL	2023-09-30 00:00:00
☐	edit	120		5	826		2024-09-05 12:12:36	24453	NULL	2024-10-31 00:00:00
☐	edit	121		5	84		2023-09-06 11:16:45	24453	NULL	2023-09-30 00:00:00
☐	edit	122		5	84		2024-09-05 12:12:57	24453	NULL	2024-10-31 00:00:00
☐	edit	123		7	651		2023-09-06 11:22:14	24453	NULL	NULL

Obrázek 27 Tabulka userroles v databázi hap

Pro ověření správnosti řízení přístupu byly vytvořeny jednotkové testy, které kontrolují oprávnění uživatelů na základě dat z tabulky userroles. K implementaci testů byla použita knihovna `pytest`. Testy lze spustit po přechodu do kořenového adresáře projektu příkazem `pytest tests/test_roles.py`.

8.2 Validace šablon otázek

Hlavním cílem validace vyplněných šablon otázek je snížení počtu chybně vyplněných souborů, které by musely být vráceny vedoucím pracovníkům k opravě. Ještě před zpřístupněním samotné žádosti o import dat probíhá automatická validace obsahu souboru. Ke snížení chybovosti významně přispívá již samotný proces exportu předvyplněné šablony otázek. Tato šablona obsahuje identifikátory úvazků a otázek, které jsou automaticky předvyplněny na základě výběru vedoucího pracovníka. Tyto informace jsou nezbytné pro následný úspěšný import do systému. Exportovaný soubor dále obsahuje doplňující údaje, například jména pracovníků, které slouží pouze pro informaci a lepší orientaci. Vedoucím pracovníkům stačí pouze doplnit údaje do žlutě označených sloupců, které jsou, stejně jako identifikátory úvazků a otázek, nezbytné pro následný úspěšný import dat do systému (viz obrázek 28).

	A	B	C	D	E	F	G	H	I	J	K	L
	Id úvazku	Id otázky	Tituly před jménem	Jméno	Příjmení	Tituly za jménem	Login	Pracoviště	Stav formuláře	Počet	Studentské hodnocení	Popis
1												
2	13832	7314 Ing.						Ústav	Předvyplněný			
3	13465	7314 Mgr.						Ústav	Předvyplněný			
4	13637	7314 RNDr.						Ústav	Předvyplněný			
5	13464	7314 Mgr.						Ústav	Předvyplněný			
6	13467	7314 Ing.						Ústav	Předvyplněný			
7	13458	7314 Mgr.						Ústav	Předvyplněný			
8	13466	7314 Ing.						Ústav	Předvyplněný			
9	13457	7314 doc. Mgr.						Ústav	Předvyplněný			
10	13462	7314 Mgr.						Ústav	Předvyplněný			
11	14119	7314 prof. Ing.						Ústav	Předvyplněný			
12	13672	7314 Mgr.						Ústav	Předvyplněný			
13	13460	7314 Mgr.						Ústav	Předvyplněný			
14	13463	7314 Ing.						Ústav	Předvyplněný			
15	13459	7314 Ing.						Ústav	Předvyplněný			
16	13817	7314 Ing.						Ústav	Předvyplněný			
17	13461	7314 RNDr.						Ústav	Předvyplněný			
18	13832	7317 Ing.						Ústav	Předvyplněný			
19	13465	7317 Mgr.						Ústav	Předvyplněný			
20	13637	7317 RNDr.						Ústav	Předvyplněný			
21	13464	7317 Mgr.						Ústav	Předvyplněný			
22	13467	7317 Ing.						Ústav	Předvyplněný			
23	13458	7317 Mgr.						Ústav	Předvyplněný			
24	13466	7317 Ing.						Ústav	Předvyplněný			
25	13457	7317 doc. Mgr.						Ústav	Předvyplněný			
26	13462	7317 Mgr.						Ústav	Předvyplněný			
27	14119	7317 prof. Ing.						Ústav	Předvyplněný			
28	13672	7317 Mgr.						Ústav	Předvyplněný			
29	13460	7317 Mgr.						Ústav	Předvyplněný			
30	13463	7317 Ing.						Ústav	Předvyplněný			
31	13459	7317 Ing.						Ústav	Předvyplněný			
32	13817	7317 Ing.						Ústav	Předvyplněný			
33	13461	7317 RNDr.						Ústav	Předvyplněný			

Obrázek 28 Předvyplněná šablona otázek

Při samotné validaci se ověřuje, zda v databázi existují úvazky a otázky uvedené v prvních dvou sloupcích šablony. Následně se podle konkrétního typu otázky kontrolují i další sloupce, které obvykle obsahují číselnou hodnotu, případně doplňující textový popis. Ostatní sloupce, jako jsou například jména zaměstnanců, označení pracoviště nebo aktuální stav formuláře, se nevalidují, protože jejich obsah není při importu dat do systému využíván.

Pokud validace souboru neproběhne úspěšně, je uživatel o této skutečnosti informován a současně se mu automaticky stáhne upravená verze šablony, ve které jsou chybné buňky přehledně označeny. Kromě toho šablona obsahuje také popis konkrétní chyby u každé problematické buňky – například chybějící hodnota, neexistující odkaz na záznam v databázi, nebo nesprávný datový typ. Ukázka takto označeného souboru je zobrazena na obrázku 29. V případě, že validace proběhne bezchybně, je vedoucímu pracovníkovi zpřístupněna možnost odeslat soubor administrátorovi k importu do systému. K souboru je nutné připojit krátkou zprávu, která popisuje účel nebo kontext odesílaných dat.

	A	B	C	D	E	F	G	H	I	J	K	L
	Id úvazku	Id otázky	Tituly před jménem	Jméno	Příjmení	Tituly za jménem	Login	Pracoviště	Stav formuláře	Počet	Studentské hodnocení	Popis
1												
2	13832	7314 Ing.						Ústav	Předvyplněný			
3	13465	7314 Mgr.						Ústav	Předvyplněný			
4	13637	7314 RNDr.						Ústav	Předvyplněný			
5	13464	7314 Mgr.						Ústav	Předvyplněný			
6	13467	7314 Ing.						Ústav	Předvyplněný			
7	13458	7314 Mgr.						Ústav	Předvyplněný			
8	13466	7314 Ing.						Ústav	Předvyplněný			
9	13457	7314 doc. Mgr.						Ústav	Předvyplněný			
10	13462	7314 Mgr.						Ústav	Předvyplněný			
11	14119	7314 prof. Ing.						Ústav	Předvyplněný			
12	13672	7314 Mgr.						Ústav	Předvyplněný			
13	13460	7314 Mgr.						Ústav	Předvyplněný			
14	13463	7314 Ing.						Ústav	Předvyplněný			
15	13459	7314 Ing.						Ústav	Předvyplněný			
16	13817	7314 Ing.						Ústav	Předvyplněný			
17	13461	7317 RNDr.						Ústav	Předvyplněný			
18	13832	7317 Ing.						Ústav	Předvyplněný			
19	13465	7317 Mgr.						Ústav	Předvyplněný			
20	13637	7317 RNDr.						Ústav	Předvyplněný			
21	13464	7317 Mgr.						Ústav	Předvyplněný			
22	13467	7317 Ing.						Ústav	Předvyplněný			
23	13458	7317 Mgr.						Ústav	Předvyplněný			
24	13466	7317 Ing.						Ústav	Předvyplněný			
25	13457	7317 doc. Mgr.						Ústav	Předvyplněný			
26	13462	7317 Mgr.						Ústav	Předvyplněný			
27	14119	7317 prof. Ing.						Ústav	Předvyplněný			
28	13672	7317 Mgr.						Ústav	Předvyplněný			
29	13460	7317 Mgr.						Ústav	Předvyplněný			
30	13463	7317 Ing.						Ústav	Předvyplněný			
31	13459	7317 Ing.						Ústav	Předvyplněný			
32	13817	7317 Ing.						Ústav	Předvyplněný			
33	13461	7317 RNDr.						Ústav	Předvyplněný			

Obrázek 29 Šablona otázek s chybnými buňkami

ZÁVĚR

V rámci této bakalářské práce bylo navrženo a implementováno nové uživatelské rozhraní pro zobrazení hodnocení akademických pracovníků Univerzity Tomáše Bati ve Zlíně. Aplikace je navržena minimalisticky a přehledně s ohledem na cílovou skupinu uživatelů, kterou tvoří především právě akademičtí pracovníci.

Vývoj probíhal s využitím webového frameworku Flask, který je vhodný pro menší až středně velké projekty a umožňuje snadné budoucí rozšiřování. Přístup k aplikaci je řízen na základě rolí. Akademičtí pracovníci mají možnost zobrazit a exportovat svá hodnocení za uzavřené formuláře, zatímco vedoucí pracovníci mají k dispozici hodnocení zaměstnanců svého pracoviště. Vedoucím pracovišť je navíc umožněno vytvářet bodovací šablony pro přizpůsobení výpočtu hodnocení dle specifických univerzitních předpisů a exportovat předvyplněné šablony otázek pro zjednodušení následného sběru dat.

Uživatelé s administrátorskou rolí mají k dispozici pokročilé funkce, jako je možnost kopírování produkční databáze do testovací instance za účelem ověřování nových funkcionalit či importů dat. Zároveň mohou spravovat párování otázek z nových formulářů, což je klíčové pro správné fungování bodovacích šablon.

Aplikace je plně funkční, implementuje všechny zásadní funkcionality a je připravena k nasazení v prostředí univerzity.

Do budoucna lze aplikaci rozšířit například o možnost importu struktury pracovišť, uživatelů a jejich úvazků pro nadcházející rok, což je v současnosti řešeno prostřednictvím jiného systému. Dále by bylo vhodné zvážit implementaci průběžného vyplňování hodnotícího formuláře – zejména v oblastech jako je tvůrčí činnost – nezávisle na době, kdy je otevřený k doplnění akademickým pracovníkům. Tento přístup by mohl akademickým pracovníkům výrazně snížit množství práce při finálním sběru dat.

SEZNAM POUŽITÉ LITERATURY

- [1] *Zákon č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách) - znění od 1. 3. 2025.* Online. In: *Zákony pro lidi*. ©2010–2025. Dostupné z: <https://www.zakonyprolidi.cz/cs/1998-111>. [cit. 2025-04-22].
- [2] *Informační systém pro hodnocení akademických pracovníků (IS HAP).* Online. Ishap.cz. Dostupné z: http://ishap.cz/index_cz.html. [cit. 2025-04-22].
- [3] *Aula.* Online. 2018, roč. XXVI, č. 1-2 / 2018. Praha: Centrum pro studium vysokého školství, 2018. ISSN 2533-4433. Dostupné z: <https://www.csvs.cz/wp-content/uploads/2020/01/Aula-01-2-2018.pdf>. [cit. 2025-04-22].
- [4] *Redesign agendy EVAK - Novinky ze světa IS MU.* Online. Informační systém Masarykovy univerzity. 2022. Dostupné z: https://is.muni.cz/blog/is_info/nov_20220613_evak?lang=cs. [cit. 2025-05-06].
- [5] *EVAK (Evaluate akademické činnosti) – pro hodnocené.* Online. Informační systém Masarykovy univerzity. Dostupné z: <https://is.muni.cz/napoveda/komunikace/evak?cop=2690054>. [cit. 2025-05-06].
- [6] *Rektorské řády a další akty: Kariérní řád a pravidelné hodnocení akademických pracovníků Univerzity Hradec Králové.* Online. In: . 2023. Dostupné z: <https://www.uhk.cz/cs/univerzita-hradec-kralove/uhk/uredni-deska/vnitri-predpisy-a-ridici-akty-2/ridici-akty/rektorske-rady-a-dalsi-akty/rady/karierni-rad-a-pravidelne-hodnoceni-akademickych-pracovniku-univerzity-hradec-kralove>. [cit. 2025-05-06].
- [7] KOŘDOUSKOVÁ, Barbora. *Web, webová stránka a webová aplikace, v čem je rozdíl?* Online. Rascasone. 2025. Dostupné z: <https://www.rascasone.com/cs/blog/web-webova-aplikace-rozdil>. [cit. 2025-04-25].
- [8] *Co dělá backend developer?* Online. Coders Lab. 2023. Dostupné z: <https://coderslab.cz/cz/blog/co-dela-backend-developer>. [cit. 2025-04-25].
- [9] *Front End vs Back End of Your Website: Everything You Need to Know.* Online. DOJO Creative. 2020. Dostupné z: <https://letsgododojo.com/front-end-vs-back-end/>. [cit. 2025-04-25].
- [10] PÍSEK, Slavoj. *HTML začínáme programovat.* 4. aktualizované vydání. Praha: Grada, 2014. ISBN 978-80-247-5059-0.

- [11] *HTML: HyperText Markup Language*. Online. MDN Web Docs. 2025. Dostupné z: <https://developer.mozilla.org/en-US/docs/Web/HTML>. [cit. 2025-04-26].
- [12] *HTML Basic Examples*. Online. W3Schools. ©1999-2025. Dostupné z: https://www.w3schools.com/html/html_basic.asp. [cit. 2025-04-26].
- [13] *CSS Introduction*. Online. W3Schools. ©1999-2025. Dostupné z: https://www.w3schools.com/css/css_intro.asp. [cit. 2025-04-26].
- [14] *What is CSS and how does it work?* Online. Hostinger. 2025. Dostupné z: <https://www.hostinger.com/tutorials/what-is-css>. [cit. 2025-04-26].
- [15] *What is Javascript (JS)?* Online. Amazon Web Services. ©2025. Dostupné z: <https://aws.amazon.com/what-is/javascript/>. [cit. 2025-04-27].
- [16] JANOVSÝ, Dušan. *Úvod do JavaScriptu*. Online. Jak psát web. ©1999–2022. Dostupné z: <https://www.jakpsatweb.cz/javascript/javascript-uvod.html>. [cit. 2025-04-27].
- [17] WILLIAMS, Margi. *What Is JavaScript Used For?* Online. ComputerScience.org. 2024. Dostupné z: <https://www.computerscience.org/bootcamps/guides/javascript-uses/>. [cit. 2025-04-27].
- [18] *JavaScript HTML DOM*. Online. W3Schools. ©1999-2025. Dostupné z: https://www.w3schools.com/js/js_htmlDOM.asp. [cit. 2025-04-27].
- [19] ALEXANDREA, Jordana. *What Is JavaScript: A Beginner's Guide to the Basics of JS*. Online. Hostinger. 2025. Dostupné z: <https://www.hostinger.com/tutorials/what-is-javascript>. [cit. 2025-04-27].
- [20] *What is a Framework?* Online. GeeksforGeeks. 2024. Dostupné z: <https://www.geeksforgeeks.org/what-is-a-framework/>. [cit. 2025-04-25].
- [21] GADHAVI, Maitray. *Most Popular Backend Frameworks for Web Development in 2025*. Online. Radixweb. 2024. Dostupné z: <https://radixweb.com/blog/best-backend-frameworks>. [cit. 2025-04-27].
- [22] SIDDHAPARA, Vishal. *Frontend vs Backend Development: All You Need to Know in 2025*. Online. Radixweb. 2024. Dostupné z: <https://radixweb.com/blog/frontend-vs-backend>. [cit. 2025-04-27].

- [23] MASHUTIN, Denis. *Django vs. Flask: Which Is the Best Python Web Framework?* Online. JetBrains Blog. 2024. Dostupné z: <https://blog.jetbrains.com/pycharm/2023/11/django-vs-flask-which-is-the-best-python-web-framework/>. [cit. 2025-04-27].
- [24] REED, Mark. *Python programming and SQL*. Great Britain: Amazon, 2022. ISBN 979-8-3732-7885-0.
- [25] BLOCH, Joshua. *Effective Java*. 2nd ed. The Java series. Upper Saddle River: Addison-Wesley, 2008. ISBN 978-0-321-35668-0.
- [26] *Spring Framework*. Online. Spring. ©2005-2025. Dostupné z: <https://spring.io/projects/spring-framework>. [cit. 2025-05-11].
- [27] BEHLER, Marco. *What is Spring Framework? An Unorthodox Guide*. Online. Marco Behler. 2022. Dostupné z: <https://www.marcobehler.com/guides/spring-framework>. [cit. 2025-05-11].
- [28] ABIOLA, Ridwan. *Software Design Principle: Inversion of Control(IOC) and Dependency Injection*. Online. DEV Community. 2024. Dostupné z: <https://dev.to/abuhisib/software-design-principle-inversion-of-controlioc-and-dependency-injection-5e5l>. [cit. 2025-05-11].
- [29] KANADE, Vijay. *What Is Aspect-Oriented Programming (AOP)? Meaning, Working, and Frameworks*. Online. Spiceworks. 2023. Dostupné z: <https://www.spiceworks.com/tech/devops/articles/what-is-aop/>. [cit. 2025-05-11].
- [30] *Usage statistics of server-side programming languages for websites*. Online. W3Techs. ©2009-2025. Dostupné z: https://w3techs.com/technologies/overview/programming_language. [cit. 2025-05-11].
- [31] *MVC*. Online. MDN Web Docs. 2023. Dostupné z: <https://developer.mozilla.org/en-US/docs/Glossary/MVC>. [cit. 2025-05-11].
- [32] *Laravel*. Online. MEMOS Software. ©2003-2025. Dostupné z: <https://www.memos.cz/technologie/laravel/>. [cit. 2025-05-11].
- [33] *Artisan Console*. Online. Laravel. ©2025. Dostupné z: <https://laravel.com/docs/12.x/artisan>. [cit. 2025-05-11].

- [34] *Lekce 1 - Úvod do CSS frameworku Bootstrap*. Online. Itnetwork.cz. ©2025. Dostupné z: <https://www.itnetwork.cz/html-css/bootstrap/kurz/uvod-do-css-frameworku-bootstrap>. [cit. 2025-05-11].
- [35] *Top 7 CSS Frameworks for Developers in 2025*. Online. BrowserStack. 2024. Dostupné z: <https://www.browserstack.com/guide/top-css-frameworks>. [cit. 2025-05-11].
- [36] PETŘÍK, Jakub. *Porovnání JS frameworks podle specifik projektu*. Bakalářská práce, vedoucí Žáček, Petr. Zlín: Univerzita Tomáše Bati ve Zlíně. Fakulta aplikované informatiky, Ústav informatiky a umělé inteligence, 2022. Dostupné také z: <http://hdl.handle.net/10563/50562>.
- [37] *Anatomy of a component*. Online. Angular. ©2010-2025. Dostupné z: <https://angular.dev/guide/components>. [cit. 2025-05-11].
- [38] *Software Framework vs Library*. Online. GeeksforGeeks. 2023. Dostupné z: <https://www.geeksforgeeks.org/software-framework-vs-library/>. [cit. 2025-05-12].
- [39] *ReactJS Virtual DOM*. Online. GeeksforGeeks. 2025. Dostupné z: <https://www.geeksforgeeks.org/reactjs-virtual-dom/>. [cit. 2025-05-12].
- [40] *Next.js*. Online. ©2025. Dostupné z: <https://nextjs.org/>. [cit. 2025-05-12].
- [41] *Creating a React App*. Online. React. Dostupné z: <https://react.dev/learn/creating-a-react-app#recommended-toolchains>. [cit. 2025-05-12].
- [42] *Rendering*. Online. Next.js. ©2025. Dostupné z: <https://nextjs.org/docs/pages/building-your-application/rendering>. [cit. 2025-05-12].
- [43] SPEIGHT, April. *Visual Studio® Code for Python® Programmers*. Online. Wiley, 2021. ISBN 9781119773368. Dostupné z: <https://app-knovel-com.proxy.k.utb.cz/s.v?8BKa1Xmj>. [cit. 2025-05-06].
- [44] *Werkzeug*. Online. Pallets. Dostupné z: <https://www.palletsprojects.com/projects/werkzeug/>. [cit. 2025-05-06].
- [45] *Flask*. Online. Pallets. Dostupné z: <https://www.palletsprojects.com/projects/flask>. [cit. 2025-05-06].
- [46] *Quickstart*. Online. Welcome to Flask — Flask Documentation (3.1.x). ©2010. Dostupné z: <https://flask.palletsprojects.com/en/stable/quickstart/#routing>. [cit. 2025-05-06].

- [47] *Template Designer Documentation*. Online. Jinja. ©2007. Dostupné z: <https://jinja.palletsprojects.com/en/stable/templates/>. [cit. 2025-05-12].
- [48] *Sessions*. Online. Flask. ©2010. Dostupné z: <https://flask.palletsprojects.com/en/stable/api/#flask.session>. [cit. 2025-05-12].
- [49] *Caching*. Online. Flask. ©2010. Dostupné z: <https://flask.palletsprojects.com/en/stable/patterns/caching/>. [cit. 2025-05-13].
- [50] *Flask-Caching*. Online. Flask-Caching. ©2016. Dostupné z: <https://flask-caching.readthedocs.io/en/latest/>. [cit. 2025-05-13].
- [51] MARTIN, Robert C.; OTTINGER, Timothy R.; LANGR, Jeff; SCHUCHERT, Brett L.; GRENNING, James W. et al. *Clean code: a handbook of agile software craftsmanship*. Robert C. Martin series. Upper Saddle River: Prentice Hall, 2009. ISBN 978-0-13-235088-4.
- [52] *LeBlanc's Law (a.k.a Later Equals Never)*. Online. Yiming Chen. ©2012-2025. Dostupné z: <https://yiming.dev/clipping/2019/03/21/le-blanc's-law-a-k-a-later-equals-never/>. [cit. 2025-05-07].
- [53] ŻYLIŃSKI, Bartłomiej. *FIRST Principles as Solid Rules for Tests*. Online. DZone. 2021. Dostupné z: <https://dzone.com/articles/first-principles-solid-rules-for-tests>. [cit. 2025-05-07].

SEZNAM OBRÁZKŮ

Obrázek 1 Část formuláře IS HAP	14
Obrázek 2 Základní ukázka HTML kódu (vlevo) a výstup (vpravo)	17
Obrázek 3 HTML DOM jako strom objektů	18
Obrázek 4 Nejpopulárnější server-side programovací jazyky na webových stránkách	21
Obrázek 5 Ukázka jednoduché Flask aplikace	24
Obrázek 6 Typy konvertorů	25
Obrázek 7 Ukázka dynamických URL adres	25
Obrázek 8 Ukázka dokumentu s Jinja2 syntaxí	26
Obrázek 9 Základní využívané entity v IS HAP	33
Obrázek 10 Stromová struktura pracovišť	34
Obrázek 11 Ukázka hodnocení v existujícím systému	35
Obrázek 12 Průběh práce s šablonami otázek v IS HAP	35
Obrázek 13 Otázky ve formuláři UTB 2023	36
Obrázek 14 Šablona v Excelu pro předvyplnění konkrétní otázky daty	36
Obrázek 15 Adresářová struktura projektu	39
Obrázek 16 Osobní hodnocení	41
Obrázek 17 Otázky z formulářů	42
Obrázek 18 Hodnocení pracoviště	43
Obrázek 19 Struktura tabulky question_years	45
Obrázek 20 Struktura tabulky evaluation_template	46
Obrázek 21 Struktura tabulky evaluation_template_meta	46
Obrázek 22 Přehled existujících bodovacích šablon	47
Obrázek 23 Stáhnutí vzorové šablony	47
Obrázek 24 Nahrání nové šablony do systému	47
Obrázek 25 Šablony otázek (poslední krok exportu)	49
Obrázek 26 Import šablony otázek po úspěšné validaci	49
Obrázek 27 Tabulka userroles v databázi hap	52
Obrázek 28 Předvyplněná šablona otázek	53
Obrázek 29 Šablona otázek s chybnými buňkami	53

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AOP	Aspect-oriented programming
CSS	Cascading Style Sheets
CTE	Common Table Expression
DI	Dependency injection
DOM	Document Object Model
EAV	Entity–attribute–value
EVAK	Evaluace akademiků
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identifikátor
IS HAP	Informační systém pro hodnocení akademických pracovníků
MVC	Model-view-controller
OBD	Osobní bibliografická databáze
ORM	Object–relational mapping
RUV	Registr uměleckých výstupů
SQL	Structured Query Language
SSG	Static Site Generation
SSR	Server Side Rendering
STAG	Studijní agenda
URL	Uniform Resource Locator
WSGI	Web Server Gateway Interface
XSS	Cross-site scripting

SEZNAM PŘÍLOH

Příloha P I: Text BSD 2–Clause licence

Příloha P II: Část vzorové bodovací šablony 2021–2023

Příloha P III: Zdrojové kódy vytvořené aplikace

PŘÍLOHA P I: TEXT BSD 2-CLAUSE LICENCE

Copyright 2024 Martin Koňářík

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

PŘÍLOHA P II: ČÁST VZOROVÉ BODOVACÍ ŠABLONY 2021–2023

Text Otázky	2021	2022	2023
Hodiny přednášek (počty rozvrhovaných hodin v ZS a LS akade	0,00	-	-
Přednáška v prezenční / kombinované formě studia BSP, MSP	1,80	2,25	2,25
Řízené konzultace v kombinované formě BSP, MSP	1,80	2,25	2,25
Přednáška ve světovém* jazyce prezenční/kombinované form	2,40	-	-
Přednáška v DSP	2,40	3,00	3,00
Přednáška v DSP ve světovém jazyce	3,60	4,50	4,50
Hodiny seminářů, cvičení, laboratoří, ateliérů (počty rozvrhova	0,00	-	-
Seminář nebo cvičení ve světovém jazyce (včetně rozvrhovaný	1,80	2,25	2,25
Semináře, cvičení, laboratoře, ateliéry, v prezenční a kombino	1,20	1,50	1,50
Vedení povinné odborné stáže [0.2 PB/položku]	0,20	-	-
Vedené bakalářské práce obhájené v akademickém roce {Acac	0,00	-	-
Ostatní bakalářské práce vedené v akademickém roce {Acader	0,00	-	-
Oponentní posudky na bakalářské práce (akademický rok {Aca	0,00	-	-
Vedené diplomové práce obhájené v akademickém roce {Acad	0,00	-	-
Ostatní vedení diplomanti v akademickém roce {AcademicYea	0,00	-	-
Oponentní posudky na diplomové práce (akademický rok {Aca	0,00	-	-
Vedení doktorandi, kteří obhájili dizertační práci v akademické	0,00	-	-
Ostatní vedení doktorandi v akademickém roce {AcademicYea	0,00	-	-
Oponentní posudky na disertační práce (akademický rok {Acac	0,00	-	-
Školitel doktoranda v 1. roce studia	30,00	30,00	30,00
Školitel doktoranda ve 2. – 4. roce studia	55,00	55,00	55,00
Školitel doktoranda v 1. roce studia ve světovém jazyce	35,00	35,00	35,00
Školitel doktoranda ve 2. – 4. roce studia ve světovém jazyce	60,00	60,00	60,00
Vedení doktoranda ke zkoušce z komunikačních dovedností v	10,00	10,00	10,00
FAI: Vedení bakalářské práce	30,00	30,00	30,00
FAI: Vedení diplomové práce	30,00	30,00	30,00
FAI: Oponování bakalářské práce	4,00	4,00	4,00
FAI: Oponování diplomové práce	4,00	4,00	4,00
FAI: Vedení bakalářské práce v anglickém jazyce	45,00	45,00	45,00
FAI: Vedení diplomové práce v anglickém jazyce	45,00	45,00	45,00
FAI: Oponování bakalářské práce v anglickém jazyce	6,00	6,00	6,00
FAI: Oponování diplomové práce v anglickém jazyce	6,00	6,00	6,00
FAME (2): zápočet před zkouškou (seminární práce, průběžné	0,20	-	-
FAME (2): zápočet před zkouškou v AJ [0.3 PB/položku]	0,30	-	-
FAME (2): semestrální, nebo vícesemestrální zkouška v BSP, NI	0,50	-	-
FAME (2): klasifikovaný zápočet v BSP, NMSP v AJ [0.4 PB/polo	0,40	-	-
FAME (2): dílčí zkouška v DSP v AJ [0.8 PB/položku]	0,80	-	-
FAME (2): člen komise pro SZZ v BSP v AJ, včetně obhajoby BP	1,00	-	-
FAME (2): člen komise pro SZZ v NMSP v AJ, včetně obhajoby I	1,30	-	-
FAME (2): člen komise pro státní doktorskou zkoušku v DSP v /	2,00	-	-

PŘÍLOHA P III: ZDROJOVÉ KÓDY VYTVOŘENÉ APLIKACE

- Zdrojové kódy aplikace nahrané v systému IS/STAG