F3

Fakulta elektrotechnická
Katedra počítačů

Bakalářská práce

Návrh a vývoj web aplikace „TestQuiz“

Dmytro Lylo
Softwarové inženýrství a technologie

Leden 2024
Vedoucí práce: Ing. Božena Mannová, Ph.D

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Lylo Dmytro** Personal ID number: **485389**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Computer Science**

Study program: **Software Engineering and Technology**

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Design and development of the "TestQuiz" web application

Bachelor’s thesis title in Czech:

Návrh a vývoj web aplikace „TestQuiz“

Guidelines:

Bibliography / sources:

[1] Roger S. Pressmann Bruce Maxim: Software Engineering: A Practitioner's Approach ,
ISBN-10: 9780078022128
[2] Help. HTTPS://WWW.FLEXIQUIZ.COM/HELP/. FlexiQuiz [online]. 2023 [cit. 2023- 01-17]. Dostupné z:
https://www.flexiquiz.com/Help/
[3] Kahoot! [online]. 2023 [cit. 2023-01-17]. Dostupné z: https://kahoot.com/what-iskahoot/ 3. How it works. Kahoot! [online].
[cit. 2023-01-17]. Dostupné z: https://kahoot.com/schools/how-it-works/
[4] Testmoz. Council of Europe (ECML/CELV) [online]. [cit. 2023-01-17]. Dostupné z:
https://www.ecml.at/Resources/InventoryofICTtools/tabid/1906/InventoryID/251/langu age/en-GB/Default.aspx

Name and workplace of bachelor’s thesis supervisor:

Ing. Božena Mannová, Ph.D. Center for Software Training FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **13.02.2023** Deadline for bachelor thesis submission: **09.01.2024**

Assignment valid until: **22.09.2024**

Ing. Božena Mannová, Ph.D. Supervisor’s signature	Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student’s signature

Poděkování / Prohlášení

Chtěl bych poděkovat vedoucí práce Ing. Boženě Mannové, Ph.D. za její rady a pomoc při psaní závěrečné práce. Také bych chtěl poděkovat své rodině a kamarádům za podporu.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 08. 01. 2024

.....

Abstrakt / Abstract

Tato bakalářská práce je zaměřena na návrh a vývoj webové aplikace TestQuiz, která umožní uživatelům vytvářet a procházet testy a bude sloužit jako pomůcka během studia.

V rámci bakalářské práce byla provedena analýza, návrh, implementace a testovaní webové aplikace TestQuiz.

Klíčová slova: webová aplikace, aplikace, výuka, testovaní, Java, Spring Boot, React

This bachelor's thesis is focused on the design and development of the web application TestQuiz, which allows users to create and take tests and serves as a study aid.

Within the scope of the bachelor's thesis, an analysis, design, implementation, and testing of the TestQuiz web application were carried out.

Keywords: web application, application, learning, testing, Java, Spring Boot, React

Obsah /

Tabulky / Obrázky

Kapitola 1
Úvod

1.1 Cíl bakalářské práce

Cílem bakalářské práce je návrh a následná implementace webové aplikace TestQuiz. Její hlavní funkce spočívá v usnadnění procesu zapamatování školní látky, protože aplikace umožňuje uživatelům samostatně vytvářet testy a procházet jejich obsah.

Hlavní funkce webové aplikace TestQuiz:

- Tvorba vlastních otázek s odpověďmi
- Možnost přidání obrázku k otázkám a odpovědím
- Tvorba témat a podtémat a přidávání otázek k nim
- Tvorba a spuštění testu
- Opakované spuštění testu s pouze špatně zodpovězenými otázkami
- Sdílení testu s ostatními uživateli

1.2 Osobní motivace

Před tím, než student úspěšně ukončí studium, musí se naučit a zapamatovat si rozsáhlé množství informací. Vzhledem k tomu, že je čas na výuku omezený, snaží se studenti používat různé metody, techniky a způsoby, aby si zapamatovali co nejvíce informací v co nejkratším čase.

Během studia jsem vyzkoušel spoustu různých způsobů, jak si lépe zapamatovat informace a připravit se ke zkouškám. Jednou z mých oblíbených technik bylo vytváření vlastních otázek a odpovědí na základě výukových materiálů a následné samotestování. Existující řešení webových aplikací, které jsem používal pro samotestování, nebyla úplně vyhovující pro mé požadavky a obsahovala velký počet nevýhod, mezi kterými se objevují například zastaralé nebo neintuitivní grafické zpracování, cena a jiné. Z tohoto důvodu bylo rozhodnuto, že v rámci bakalářské práce bude navržena a implementována webová aplikace s širokým spektrem funkcí, která umožní vytvářet a procházet studentům vlastní otázky ve formě interaktivního testu. Aplikace pomůže ušetřit čas a celkově optimalizovat výukový proces studentů.

1.3 Struktura

Bakalářská práce je rozdělena do čtyř částí: analýza, návrh, implementace a testovaní. Hlavním cílem první části je získat přehled o požadovaných funkcích, které aplikace TestQuiz musí obsahovat, a následně vybrat vhodné technologie pro jejich implementaci.

Na základě této analýzy je druhá část zaměřena na výběr vhodné architektury aplikace a také navržen doménový model tříd, diagram nasazení a vzhled uživatelského rozhraní.

Ve třetí části bakalářské práce je popsáno, jak proběhl proces implementace webové aplikace TestQuiz. Kapitola bude zaměřená na podrobný popis nástrojů, které byly použity při vývoji aplikace a jejích jednotlivých částí. Dále je v této části popsán způsob zabezpečení aplikace. Na konci kapitoly je popsán proces nasazení aplikace na hostingovou platformu Heroku.

V poslední části bakalářské práce budou popsány postupy a metody, kterými byla testována webová aplikace TestQuiz. Na základě výsledků uživatelského testování byly identifikované prvky, které bylo zapotřebí opravit s cílem zlepšení kvality výsledného produktu.

Kapitola 2
Analýza

V rámci kapitoly Analýza jsou představeny již existující webové aplikace, které umožňují vytvářet a provádět testy. Následně je provedena analýza stávajících řešení, která umožnila nadefinovat funkční i nefunkční požadavky a případy užití. Na konci této fáze jsou znázorněny a analyzovány zvolené technologie, které budou použity při implementaci webové aplikace TestQuiz.

2.1 Existující řešení

Na trhu se vyskytuje spousta různých řešení, která umožňují vytvářet a spouštět testy. Cílem této kapitoly je poskytnout přehled o aktuálních webových aplikacích a také představit jejich hlavní výhody a nevýhody. Při analýze je kladen důraz na to, jaké existují možnosti při tvorbě a spravovaní testů a otázek. Dále je prozkoumáno uživatelské rozhraní s ohledem na vizuální atraktivitu, grafický vzhled a intuitivnost při používání.

2.1.1 TestMoz

Testmoz je online nástroj pro vytváření testů. Uživatelé si mohou vybrat mezi různými typy otázek a importovat otázky ze souborů nebo jiných testů. Po vyplnění testu je možné zobrazit všechny výsledky společně nebo zvýraznit výsledky studentů. Nástroj také umožňuje vytvářet a poskytovat zpětnou vazbu. [1]

Výhody:
- Jednoduché používaní
- Různé typy otázek
- Lze vkládat multimediální obsah k otázkám
- Lze importovat vlastní otázky
- Možnost poskytovat zpětnou vazbu

Nevýhody:
- Zastaralé grafické rozhraní
- Některé funkce jsou dostupné pouze v placené verzi
- Využití spíše ve školách nebo firmách než pro samostudium

2.1.2 FlexiQuiz

FlexiQuiz je online platforma, která poskytuje možnost vytváření testů pro firmy, školy nebo pro zábavu. Platforma umožňuje uživatelům vytvářet rozmanité typy otázek doplněné o obrázky, video a audio nahrávky, procházet testem a sdílet test. V rámci FlexiQuiz lze také vytvořit privátní banku otázek, která podporuje různé funkce: ukládat otázky a odpovědi, rozdělovat otázky podle kategorií, přidávat otázky k testům. [2]

Výhody:

- Různé typy otázek
- Vkládání multimediálního obsahu k otázkám
- Možnost rozdělovat otázky podle kategorií
- Možnost sdílení testů mezi uživateli

Nevýhody:

- Neintuitivní grafické rozhraní
- Velké množství funkcí je dostupné pouze v placené verzi
- Využití spíše ve školách nebo firmách než pro samostudium

2.1.3 Kahoot!

Kahoot! je herní výuková platforma, která usnadňuje vytváření a sdílení výukových her nebo kvízů. [3] Platforma nabízí možnost vytvářet různé typy otázek a přidávat k nim obrázky, videa a diagramy. Také platforma umožňuje spustit vlastní kvíz, do kterého se mohou připojit jiní uživatelé pomocí unikátního PIN a sdílet kvízy s ostatními uživateli. [4]

Výhody:

- Různé typy otázek
- Vkládání multimediálního obsahu k otázkám
- Interaktivní a atraktivní přístup k výuce
- Možnost spouštění vlastních kvízů a připojení ostatních uživatelů

Nevýhody:

- Velké množství funkcí je dostupné pouze v placené verzi
- Zaměřeno více na zábavu než pro studium

2.1.4 Shrnutí analýzy existujících řešení

Po detailní analýze existujících řešení pro tvorbu a procházení testů bylo zjištěno, že představené aplikace obsahují pro studenty některé nadbytečné funkce, které spíše než pro samotné studenty slouží ve skutečnosti více jako přehled pro učitele. Nepřispívají tedy tolik samostudijnímu procesu. Mezi příklady nadbytečné funkcionality lze uvézt například různorodost statistik (výsledků jednotlivců, všech studentů ve skupině, porovnání výsledků), plánování doby spuštění testů a jiné. Představená existující řešení jsou konceptuálně odlišná od cíle této práce. Ten se zaměřuje spíše na potřeby studenta, podle kterých vyvíjí také funkcionalitu.

V tabulce 2.1 jsou uvedené hlavní výhody a nevýhody analyzovaných existujících řešení.

	TestMoz	FlexiQuiz	Kahoot!
Výhody	Jednoduchost použití	Privátní banka otázek	Interaktivní a atraktivní přístup
Nevýhody	Zastaralé grafické rozhraní	Neintuitivní uživatelské rozhraní	Zaměření na zábavu

Tabulka 2.1. Výhody a nevýhody analyzovaných existujících řešení.

2.2 Specifikace požadavků

V rámci této kapitoly jsou dle zadání práce a analýzy existujících řešení definovány klíčové požadavky pro webovou aplikaci TestQuiz.

2.2.1 Funkční požadavky

Tato podkapitola představuje přehled nadefinovaných funkčních požadavků, tedy specifikaci funkcionalit, které systém poskytuje uživatelům.

- **FRQ1 – Autorizace**
  - **SRQ101 – Registrace**
    Systém umožňuje zaregistrovat nového uživatele. Registrace bude probíhat vyplněním registračního formuláře.
  - **SRQ102 – Přihlášení**
    Systém se umožňuje přihlásit registrovanému uživateli. Přihlášení bude probíhat vyplněním formuláře, kde uživatel zadá přihlašovací jméno a heslo.
  - **SRQ103 – Odhlášení**
    Systém se přihlášenému uživateli umožňuje odhlásit.
  - **SRQ104 – Změna hesla**
    Systém umožnuje registrovanému a přihlášenému uživateli změnit heslo.
- **FRQ2 – TÉMA/PODTÉMA**
  - **SRQ201 – Vytvoření tématu**
    Systém umožňuje přihlášenému uživateli vytvořit nové téma.
  - **SRQ202 – Spravování témat**
    Systém umožňuje přihlášenému uživateli spravovat vlastní téma.
  - **SRQ203 – Vytvoření podtémat**
    Systém umožňuje přihlášenému uživateli vytvořit nové podtéma.
  - **SRQ204 – Spravování podtémat**
    Systém umožňuje přihlášenému uživateli spravovat vlastní téma.
  - **SRQ205 – Zobrazení témat/podtémat**
    Systém umožňuje přihlášenému uživateli zobrazit vlastní téma a podtéma.
- **FRQ3 – OTÁZKY**
  - **SRQ301 – Vytvoření otázek**
    Systém umožnuje přihlášenému uživateli přidávat do témat a podtémat nové otázky.
  - **SRQ302 – Spravování otázek**
    Systém umožňuje přihlášenému uživateli spravovat vlastní otázky
  - **SRQ303 – Zobrazení otázek**
    Systém umožňuje přihlášenému uživateli zobrazit vlastní otázky v rámci tématu nebo podtématu.
  - **SRQ304 – Přidání odpovědí**
    Systém umožňuje přihlášenému uživateli přidávat odpovědi k vlastním otázkám.
  - **SRQ305 – Spravování odpovědí**
    Systém umožňuje přihlášenému uživateli spravovat odpovědi.
  - **SRQ306 – Zobrazení odpovědí**
    Systém umožňuje přihlášenému uživateli zobrazit vlastní odpovědi v rámci otázky.
- **FRQ4 – TESTY**
  - **SRQ401 – Vytvoření testu**
    Systém umožňuje přihlášenému uživateli vytvořit nový test.

  - **SRQ402 – Spravování testu**
    Systém umožňuje přihlášenému uživateli spravovat vlastní testy.
  - **SRQ403 – Zveřejnění testu**
    Systém umožňuje přihlášenému uživateli zveřejnit test ostatním přihlášeným uživatelům.
  - **SRQ404 – Spuštění testu**
    Systém umožňuje přihlášenému uživateli spustit vlastní testy nebo testy zveřejněné ostatními uživateli.
  - **SRQ405 – Spuštění testu pouze špatně zodpovězených otázek**
    Systém umožňuje přihlášenému uživateli po vyplnění testu spustit test z pouze špatně zodpovězených otázek.
- **FRQ5 REPORTY**
  - **SRQ501 – Vytvoření reportu**
    Systém umožňuje přihlášenému uživateli vytvořit report vyplněním formuláře.
  - **SRQ502 – Zobrazení reportu**
    Systém umožňuje uživateli z role administrátor zobrazit všechny reporty.

2.2.2 Nefunkční požadavky

Nefunkční požadavky definují vlastnosti týkající se použitelnosti a funkčnosti aplikace.

- **NFR1 – Podpora prohlížečů**
  Aplikace bude podporovaná v aktuálních verzích různých prohlížečů.
- **NFR2 – Bezpečnost**
  Aplikace neumožní přístup k jednotlivým stránkám a funkcionalitám bez přihlášení. Aplikace musí být ochráněna proti XSS útokům.
- **NFR3 – Intuitivní rozhraní**
  Aplikace bude mít intuitivně pochopitelné uživatelské rozhraní.
- **NFR4 – Obrázky**
  Aplikace umožní přidávání obrázků k otázkám a odpovědím.

2.3 Případy užití

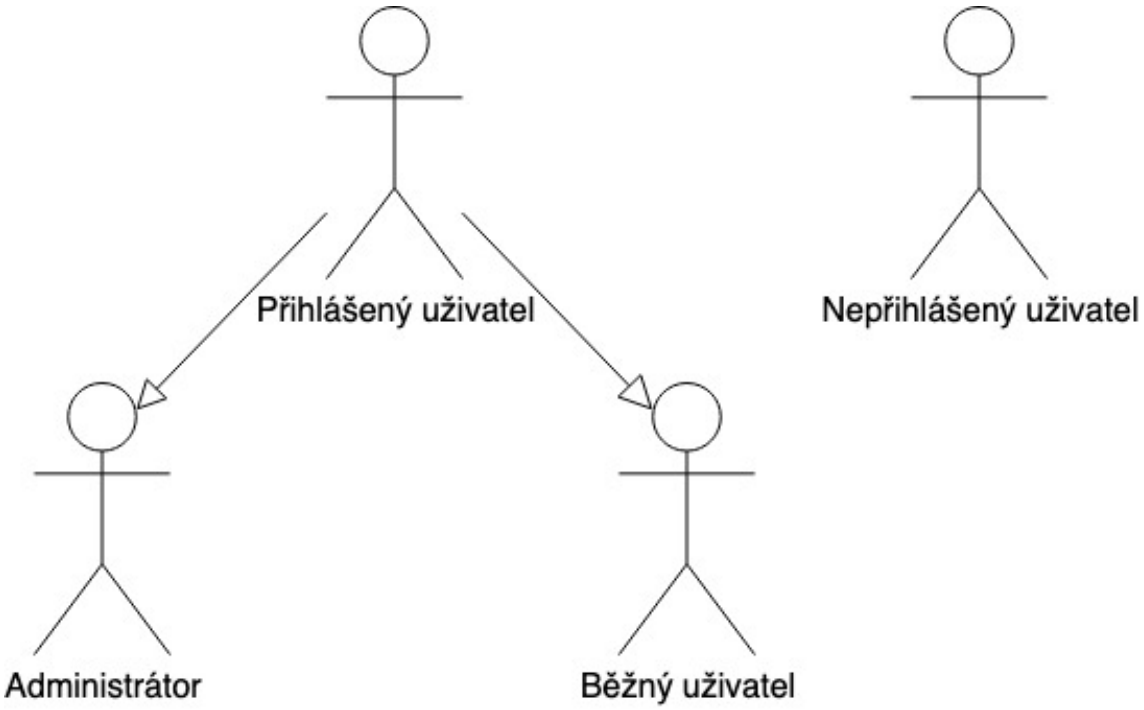
V této podkapitole jsou definovány uživatelské role a také je navržen use case diagram, který reprezentuje interakce mezi uživateli a systémem.

2.3.1 Aktéři

Aktér reprezentuje roli uživatele, který komunikuje se systémem. [5] V systému jsou definované následující uživatelské role:

- Nepřihlášený uživatel – Může se přihlásit nebo se registrovat do systému.
- Přihlášený uživatel – Uživatel, který je zaregistrován v systému. Tato role se dělí na dvě další:
  - Běžný uživatel – Může spravovat vlastní téma/podtéma, otázky a testy.
  - Administrátor – Může provádět stejné operace jako běžný uživatel, ale navíc může odstranit zveřejněné testy a také má přístup k reportům.

Uživatelské role jsou znázorněné na obrázku 2.1.

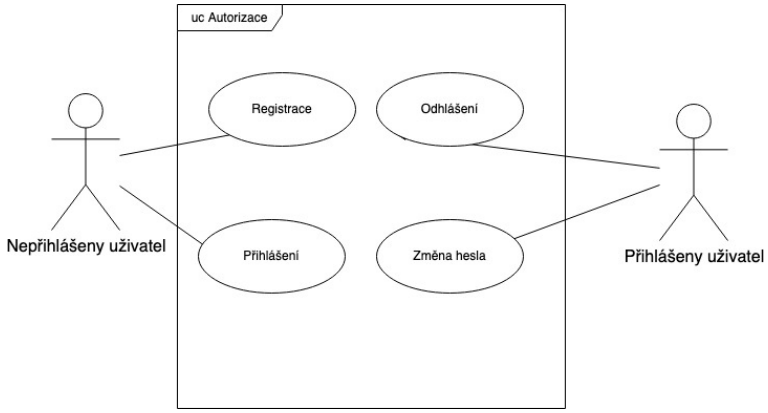


Obrázek 2.1. Uživatelské role.

2.4 Diagram případů užití

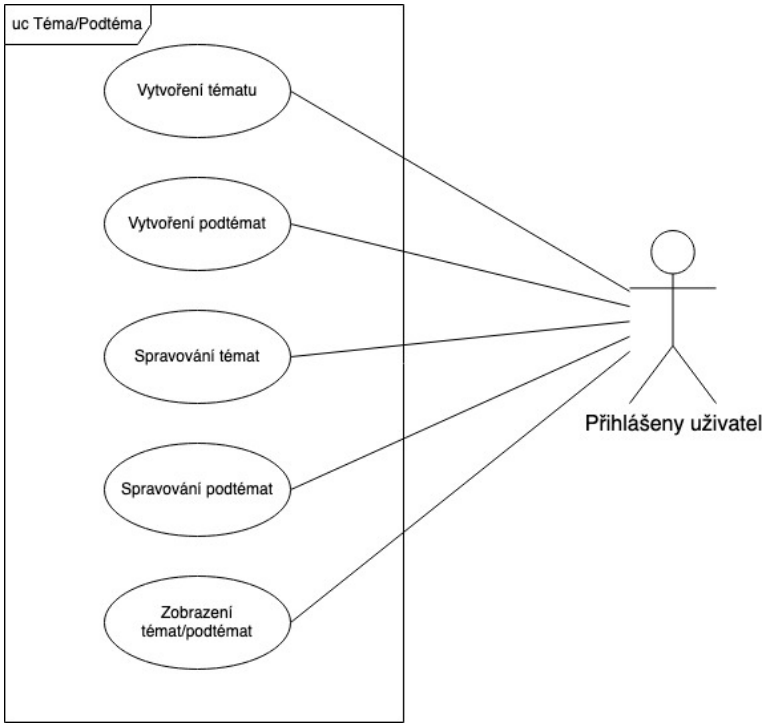
Diagramy případů užití ilustrují kontext a požadavky buď celého systému, nebo jeho důležitých částí. Kromě toho popisují, co systém dělá a jak jej aktéři používají, ovšem neznázorňují, jak systém interně funguje. [5] V této kapitole případy užiti ilustrují výše popsané funkční požadavky.

Na obrázku 2.2 je znázorněn diagram případů užití pro funkční požadavek autorizace.



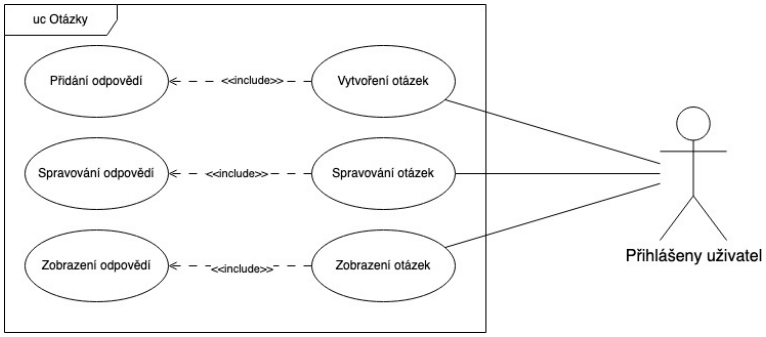
Obrázek 2.2. Diagram případů užití – Autorizace.

Na obrázku 2.3 je znázorněn diagram případů užití pro funkční požadavek téma/podtéma.



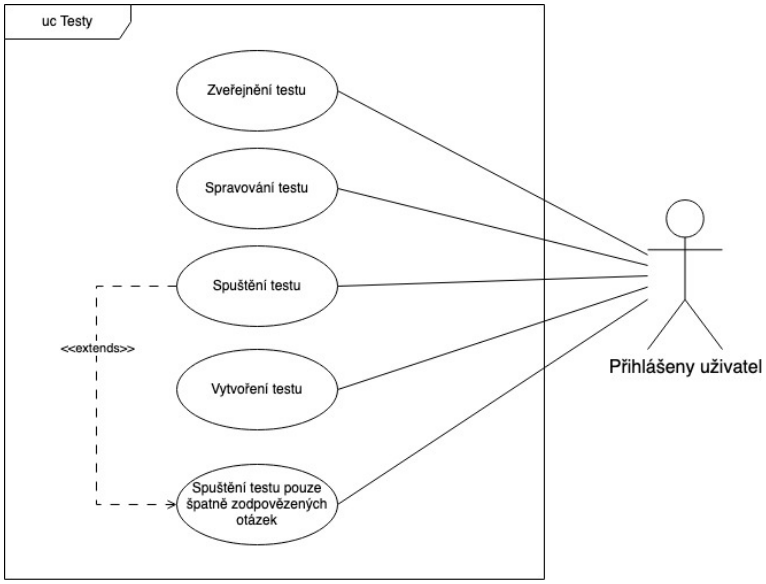
Obrázek 2.3. Diagram případů užití – Téma/Podtéma.

Na obrázku 2.4 je znázorněn diagram případů užití pro funkční požadavek otázky.



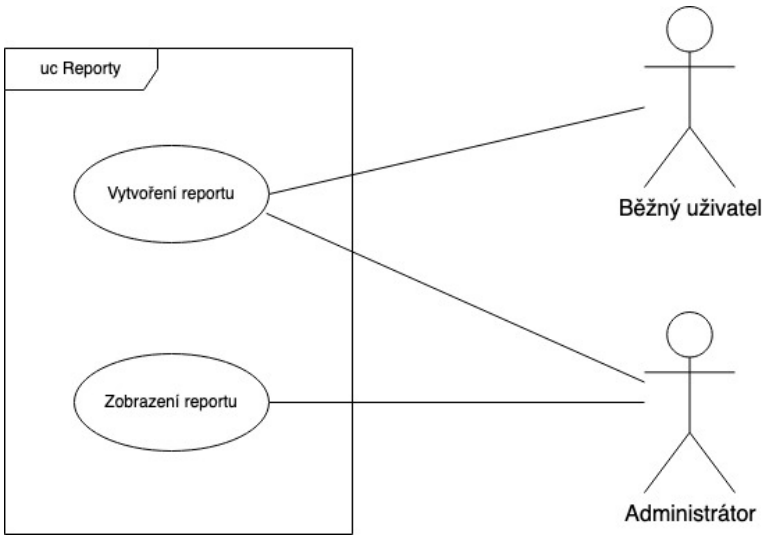
Obrázek 2.4. Diagram případů užití – Otázky.

Na obrázku 2.5 je znázorněn diagram případů užití pro funkční požadavek testy.



Obrázek 2.5. Diagram případů užití – Testy.

Na obrázku 2.6 je znázorněn diagram případů užití pro funkční požadavek reporty.



Obrázek 2.6. Diagram případů užití – Reporty.

2.5 Výběr technologie

V této části je provedena analýza technologií používaná pro vývoj webových aplikací. Cílem této kapitoly je vybrat nejvhodnější technologie pro jednotlivé části aplikace TestQuiz. Výběr vycházel z definovaných požadavků, jednoduchosti implementace a z vlastní zkušenosti.

V rámci této kapitoly je provedena obecná analýza technologií použitých pro vývoj webových aplikací. Cílem této kapitoly je získat přehled o technologiích a vybrat ty nejvhodnější pro implementaci jednotlivých částí aplikace TestQuiz. Výběr technologie vycházel z definovaných požadavků, jednoduchosti implementace a z vlastní zkušenosti.

2.5.1 Serverová část

V této části je provedena analýza technologie, které se používají pro vývoj serverové části webových aplikací. V analýze jsou vymezené technologie, se kterými jsem měl zkušenosti v minulých projektech z vývoje webových aplikací.

- **Java**

  Java je objektově orientovaný programovací jazyk vynalezený v roce 1991 Jamesem Goslingem ze Sun Microsystems, který je podporován na velkém počtu zařízení, včetně notebooků, mobilních zařízení, herních konzolí a mnoha dalších. [6]

  Výhody:
  - Java je podporovaná na velkém počtu zařízení
  - Rozsáhlá podpora a velký počet knihoven
  - Větší osobní zkušenost

  Nevýhody:
  - Oproti jazyku Python může být proces psaní kódu pomalejší

- **Python**

  Python je interpretovaný a vysokoúrovňový objektově orientovaný programovací jazyk. Python je široce používán ve vědeckých výpočtech, vývoji webových aplikací a automatizaci. [7]

  Výhody:
  - Díky lehce čitelné syntaxi psaní kódu může být rychlejší
  - Rozsáhlá podpora a velký počet knihoven

  Nevýhody:
  - Oproti Javě může mít menší výkonnost
  - Menší osobní zkušenost

- **Spring a Spring Boot**

  Spring je framework pro vývoj aplikací v programovacím jazyce Java a Spring Boot je modulem Spring Framework, který umožňuje vytvořit samostatnou aplikaci s minimálními nebo nulovými konfiguracemi. Její optimální použití je pro vytvoření Spring-based aplikace nebo RESTful služby. [8]

  Výhody:
  - Poskytuje bezpečnostní mechanismy pro autorizace a autentizace
  - Poskytuje snadnou práci s datovou vrstvou
  - Poskytuje snadnou práci s koncovými body
  - Větší osobní zkušenost

  Nevýhody:
  - Spring Boot poskytuje rozsáhlé možnosti konfigurace, občas může být problém se v něm zorientovat

- **Django**

  Django je open source webový framework pro vývoj webové aplikace v jazyce Python. Django byl navržen tak, aby umožnil vývojáři rychle a prakticky vyvíjet webové stránky. Jednou z výhod Django je velká komunitní podpora. [9]

  Výhody:
  - Snadný vývoj webových aplikací
  - Velká komunita vývojářů
  - Automatická administrace

  Nevýhody:
  - Velké množství kódu
  - Menší osobní zkušenost

2.5.2 Klientská část

V této části práce je provedena analýza technologií, které se používají pro vývoj klientské části webových aplikací. V analýze se používaly technologie, se kterými jsem měl zkušenosti při vývoji webových aplikací.

- **JavaScript**

  JavaScript je programovací jazyk vyvinutý v roce 1995, jehož autorem je Brendan Eich. JavaScript byl navržen tak, aby umožnil programátorem vytvářet interaktivní webové aplikace. [10]

- **React**

  React je open source JavaScript knihovna od firmy Meta pro vývoj uživatelské rozhraní. React umožňuje vytvářet uživatelské rozhraní pomocí malých a izolovaných částí kódu, tzv. komponentů. [11] Hlavní výhodou Reactu je používání virtuálního DOMu, který přispěje ke zlepšení výkonu aplikace, protože virtuální DOM je rychlejší něž běžný. [12] Další výhodou Reactu je velká podpora komunity.

  Výhody:
  - Virtuální DOM
  - Rozsáhlá podpora a velká komunita vývojářů
  - Větší osobní zkušenost

  Nevýhody:
  - Oproti Vue může mít menší výkonnost

- **Vue**

  Vue je JavaScript framework pro vývoj uživatelského rozhraní. Stejně jako React, Vue poskytuje deklarativní programovací model založený na komponentách.

  Hlavní výhodou Vue je jeho rychlost díky malé velikosti. [13]

  Výhody:
  - Virtuální DOM
  - Rychlost

  Nevýhody:
  - Menší komunita

2.5.3 Databáze

Pro ukládání dat je zapotřebí vybrat vhodný databázový systém, který bude splňovat cíle programu TestQuiz. V rámci této podkapitoly jsou analyzované dva hlavní typy databází: SQL a NoSQL.

SQL databáze neboli relační databáze jsou založeny na jednoduchém způsobu prezentace dat ve strukturovaných tabulkách. V relační databázi je každý řádek v tabulce záznamem s jedinečným identifikátorem, který se nazývá klíč. Sloupce tabulky obsahují datové atributy a každý záznam má obvykle hodnotu pro každý atribut, což usnadňuje vytváření vztahů mezi datovými body. [14] Relační databáze představuje optimální volbu v případě, když máme přesně definovanou strukturu dat a nemusíme zpracovávat velké množství dat. Příklady relačních databázových systémů: PostgreSQL, MySQL, OracleSQL.

NoSQL databáze neboli nerelační databáze jsou databáze, které nepoužívají tabulky a místo toho využívají flexibilnější strukturu dat. Můžou ukládat data například jako dvojice klíč-hodnota, grafy nebo v JSON formátu. [15] NoSQL databáze jsou vhodné pro zpracování velkého množství dat a v situacích, kdy nemáme přesně definováno, jak budou data uložena. Příklady nerelačních databázových systémů: MongoDB, Redis, Amazon DynamoDB.

2.5.4 API Protokol

API (Application Programing Interface) je soubor definovaných pravidel, který slouží pro komunikaci mezi různými aplikacemi.

Pro komunikaci s webovými stránkami se používá protokol HTTP, proto v této části byla provedena analýza webových API, které využívá HTTP a se kterými mám zkušenosti.

- **SOAP**

  SOAP (The Simple Object Access Protocol) – je protokol založený na XML pro výměnu informací decentralizovaném, distribuovaném prostředí. [16] SOAP podporuje komunikaci mezi systémy pomocí zpráv ve formátu XML. Protokol se často používá v případech, kde je potřeba bezpečnosti přenosu dat.

  Výhody:
  - Bezpečnost
  - Nezávislost na platformě

  Nevýhody:
  - Velikost souboru XML
  - Čitelnost

- **REST (RESTful API)**

  RESTful API představuje architektonický styl, který používá HTTP dotazy pro provádění standardních funkcí pro manipulaci s daty jako je vytváření, zobrazení, aktualizace a mazaní. Pro tyto účely se používají standardní metody HTTP: GET pro zobrazení, POST pro vytvoření, PUT pro aktualizaci, DELETE pro mazání. Také REST je schopen přenášet stav dat v různých formátech jako například JSON, XML, HTML nebo prostý text. [17]

  Výhody:
  - Jednoduchost
  - Nezávislost na platformě
  - Různé formáty pro přenos dat

  Nevýhody:
  - REST v sobě nemá žádné bezpečnostní mechanizmy, například oproti SOAP

2.5.5 Zvolené řešení

Na základě provedené analýzy a definovaných požadavků byly vybrány optimální technologie pro implementaci aplikace TestQuiz.

Pro serverovou část byly zvoleny programovací jazyky Java a framework Spring Boot. Hlavním důvodem tohoto řešení byla snadná práce s datovou vrstvou a osobní zkušenost.

Pro klientskou část byly zvoleny jazyk JavaScript a knihovna React. Hlavním důvodem tohoto řešení byla větší komunita a ekosystém knihovny React, což může pomoct v případě problémů při implementaci.

Pro manipulaci s daty byl vybrán databázový systém PostgreSQL pro relační databáze. PostgreSQL je jeden z nejpokročilejších open source databázových systémů a má velké množství funkcí, které široce využívají vývojáři a správci systému. [18] Hlavním důvodem pro výběr relační databáze jsou přesně definovaná struktura dat a předpoklad, že systém nebude potřebovat zpracovávat velké množství dat.

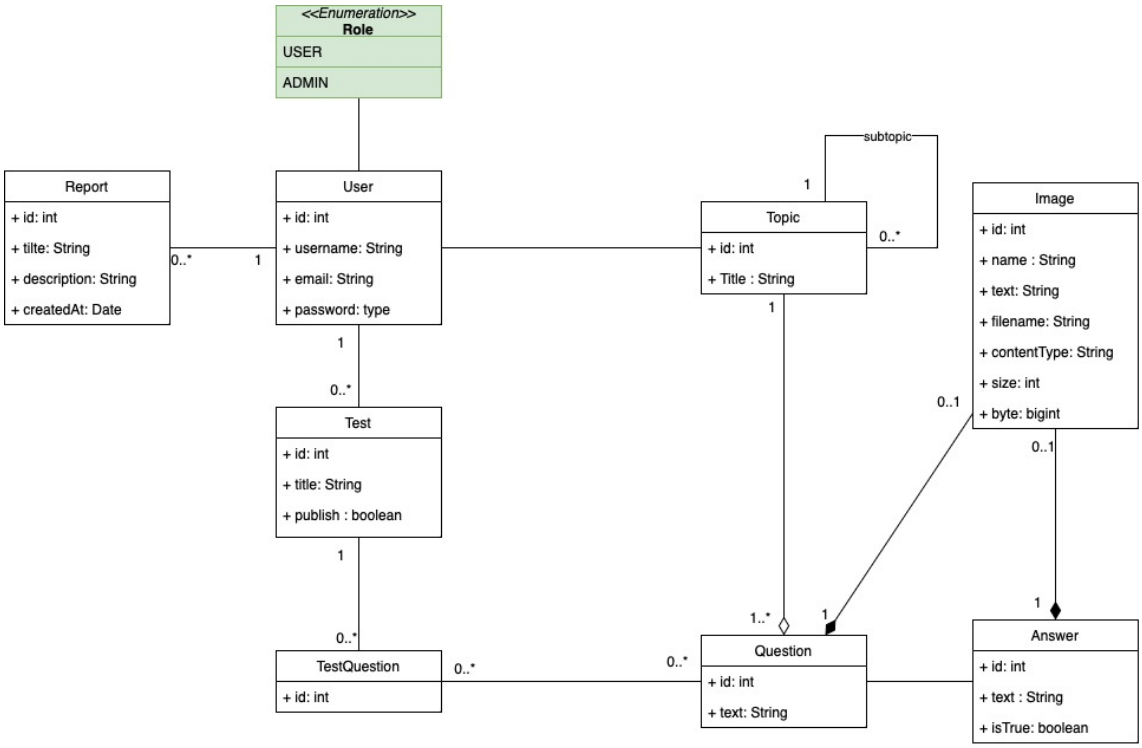
Pro komunikaci mezi serverovou a klientskou částí aplikace byl zvolen RESTful API. Zdůvodněním tohoto řešení je osobní zkušenost a schopnost efektivně přenášet různé typy dat.

Kapitola 3
Návrh

3.1 Doménový model tříd

Na základě analýzy funkčních požadavků a případů užití byl navržen doménový model tříd, který popisuje klíčové entity aplikace, jejich atributy a vztahy mezi nimi.

Doménový model tříd je znázorněn na obrázku 3.1.



Obrázek 3.1. Doménový model tříd.

Seznam entit:

- User – reprezentuje uživatele systému, může mít roli administrátora nebo běžného uživatele
- Topic – reprezentuje téma a podtéma
- Question – reprezentuje otázku
- Answer – reprezentuje odpověď na otázku
- Image – reprezentuje obrázek, který obsahuje otázku nebo odpověď
- TestQuestion – asociační třída reprezentuje otázku, která se nachází v testu
- Report – reprezentuje reporty
- Role – reprezentuje uživatelskou roli

3.2 Architektura

Třívrstvá architektura (three-tier architecture) je dobře zavedená architektura softwarových aplikací, která organizuje aplikaci do tří logických a fyzických výpočetních vrstev: prezentační vrstva, aplikační vrstva a datová vrstva.

Prezentační vrstva je uživatelské rozhraní a komunikační vrstva aplikace, kde koncový uživatel komunikuje s aplikací. Jejím hlavním účelem je zobrazovat informace a shromažďovat informace od uživatele. Aplikační vrstva používá business logiku, která podporuje základní funkce aplikace. Aplikační vrstva může také přidávat, odstraňovat nebo upravovat data v datové vrstvě.

Datová vrstva je vrstva, ve které se zpracovávají a ukládají data.

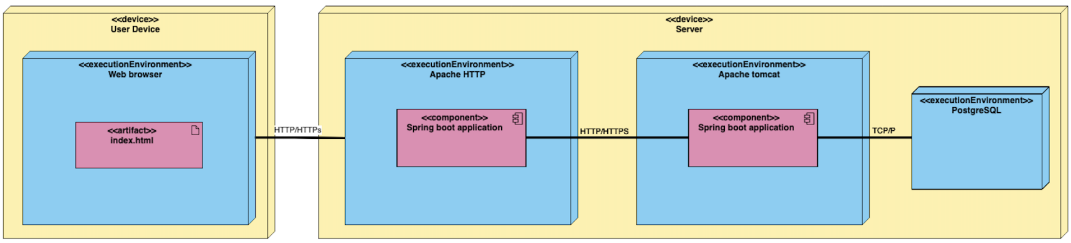
V třívrstvé aplikaci prochází veškerá komunikace přes aplikační vrstvu. Prezentační vrstva a datová vrstva spolu nemohou přímo komunikovat. [19]

Pro webovou aplikaci TestQuiz byla zvolena tato architektura, protože každá část aplikace je samostatně funkční a může být implementována separátně a změny provedené v jedné vrstvě nebudou ovlivňovat ostatní.

3.3 Diagram nasazení

Diagram nasazení podrobně popisuje fyzickou architekturu systému. Také diagram nasazení zobrazuje vztahy mezi hardwarovými a softwarovými komponentami systému a interakce mezi nimi. [20]

Fyzická architektura aplikace je znázorněna na obrázku 3.2.



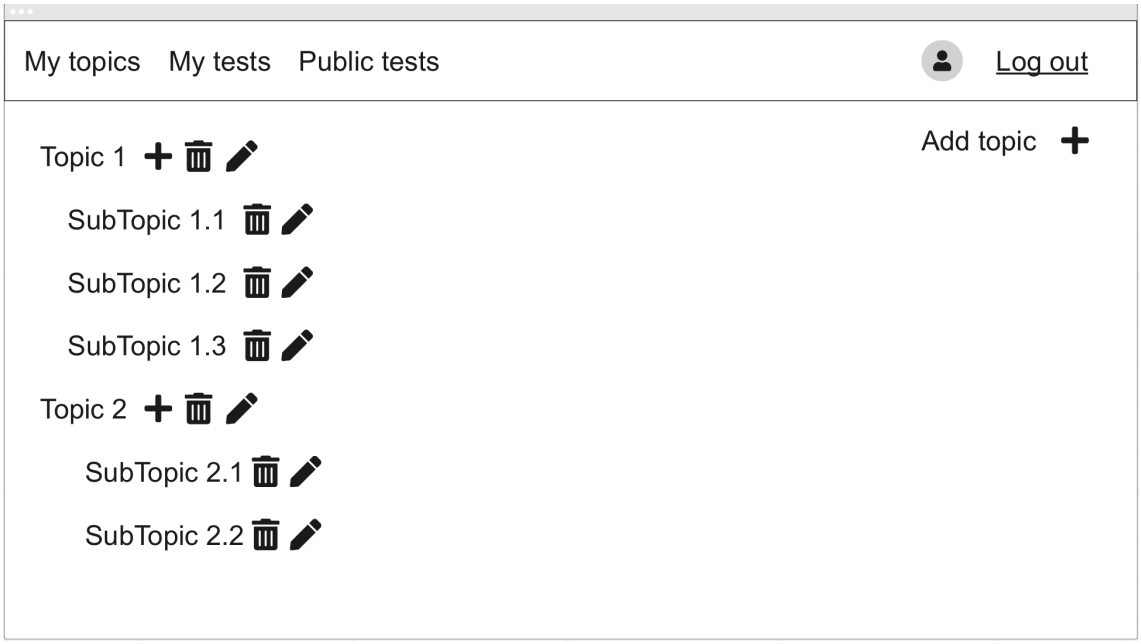
Obrázek 3.2. Diagram nasazení.

3.4 Návrh uživatelského rozhraní

Pro návrh uživatelského rozhraní byl použit nástroj Miro. Miro je online platforma pro digitální řízení projektu. Platforma umožňuje přidávat různý obsah od textů po obrázky, vytvářet diagramy a vizuální šablony na digitálním plátně a sdílet je jiným uživatelům. [21]

Navržené uživatelské rozhraní je dostupné na adrese uvedené v příloze B.

Na obrázku 3.3 je znázorněn příklad návrhu stránky s téma.



Obrázek 3.3. Návrh uživatelského rozhraní – *My topics*.

Kapitola 4
Implementace

V této kapitole je popsaný postup při implementaci: nástroje, které byly použity během implementace, a také to, jak byly implementovány jednotlivé části aplikace.

4.1 Postup při implementaci

Po provedení analýzy a návrhu jsem přistoupil k implementaci webové aplikace TestQuiz. Na začátku byla implementována serverová část. Po vytvoření serverové části aplikace jsem přistoupil k implementaci klientské části. Během implementace se provádělo testování jednotlivých funkcí aplikace a nalezené chyby se následně opravovaly. Po dokončení implementace byla webová aplikace TestQuiz nasazena na hosting.

4.2 Použité nástroje

V této části jsou popsané nástroje, které byly použity během implementace.

- **Vývojové prostředí**

  IDE (integrated development environment) je software pro vývoj aplikace, který spojuje vývojářské nástroje do jednoho grafického rozhraní. Vývojové prostředí se obvykle skládá z textového editoru pro psaní kódů a různých nástrojů, které automatizují některé opakovatelné úlohy. Jedná se, například o kompilaci zdrojového kódu, debuggeru neboli ladicího programu, který pomáhá najít umístnění chyby v kódu. [22]

  Pro psaní kódu bylo použito komerční vývojové prostředí IntelliJ IDEA od společnosti JetBrains. Hlavními důvody pro používání této IDE jsou podpora různých programovacích jazyků a podpora množství pluginu, které usnadňují psaní kódu. S tímto vývojářským prostředím mám vlastní zkušenosti.

- **GitHub**

  GitHub je platforma pro verzování kódu založená na technologii Git. Git je distribuovaný systém pro správu verzí, který umožnuje vytvářet lokální kopie vzdáleného repozitáře a zaznamenávat do nich provedené změny. Poté lze tyto změny poslat zpět a uložit jako novou verzi. Výhodou Gitu je také možnost snadného přepínání mezi různými verzemi repozitáře.

  Pro aplikaci TestQuiz byly vytvořeny dva repozitáře: jeden pro klientskou část a druhý pro serverovou část.

  Odkaz na repozitáře se nachází v příloze B.

- **Postman**

  Postman je platforma pro vytváření a testovaní API. [23] Tato platforma byla použita pro testování koncových bodů serverové části ještě před tím, než byla implementovaná klientská část.

- **Firefox**

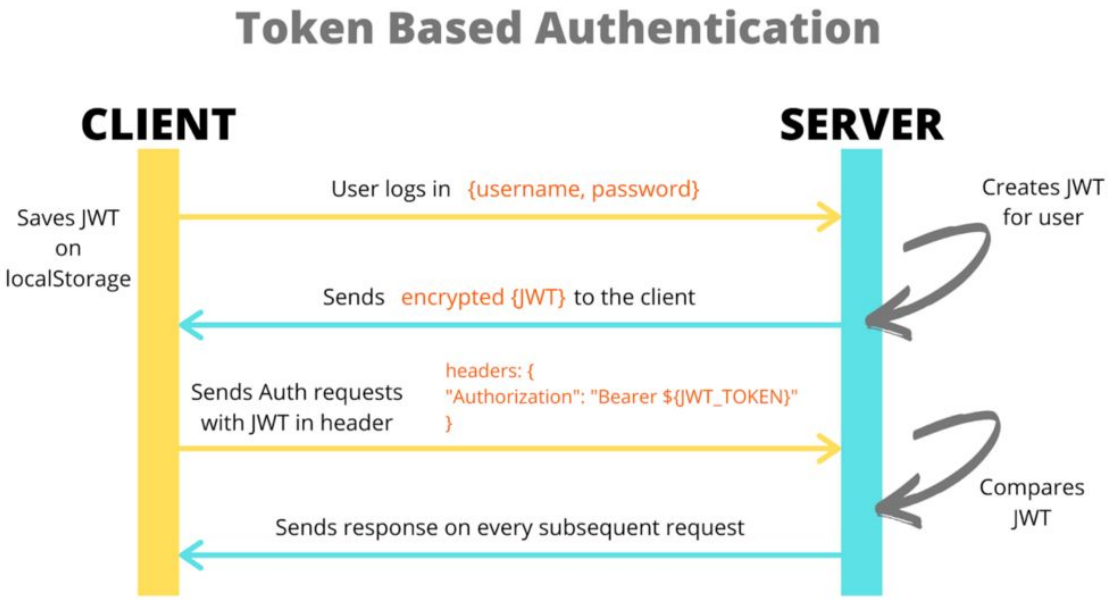
  Firefox je prohlížeč od společnosti Mozilla. Tento prohlížeč byl použit pro ladění klientské části aplikace.

4.3 Zabezpečení

Pro autentizaci a autorizaci se používá způsob zabezpečení pomocí JWT tokenů.

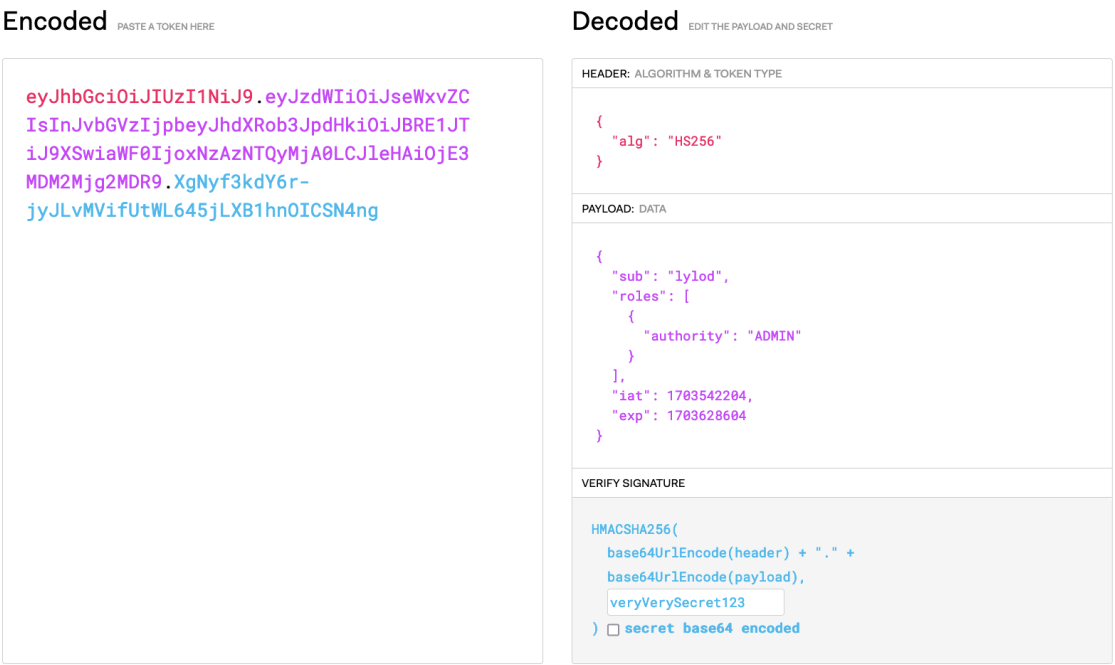
JWT neboli JSON web token je otevřený standard, který definuje kompaktní a autonomní způsob bezpečného přenášení informací ve formátu JSON, které jsou podepsané pomocí tajného klíče nebo páru soukromého a veřejného klíče. [24]

Na obrázku 4.1 je znázorněn proces autentizace pomocí JWT.



Obrázek 4.1. Proces autentizace pomoci JWT. [25]

Na obrázku 4.2 je znázorněna struktura JWT tokenu.



Obrázek 4.2. Struktura JWT tokenu.

4.4 Serverová část

V této části je popsána struktura projektu serverové části.

4.4.1 BackendAplication.java

Tato třída je vstupním bodem aplikace. Anotace *@SpringBootApplication* určuje, že aplikace bude běžet jako spring bootová aplikace a zajišťuje automatickou konfiguraci.

4.4.2 Config

Tento balík obsahuje třídu *WebSecurityConfig*, která slouží ke konfiguraci zabezpečení aplikace.

4.4.3 Entities

Tento balíček obsahuje třídy, které reprezentují tabulku v databázi a odpovídají navrženému diagramu tříd. Třídy obsahují anotace *@Entity*, *@Table*, *@Id*, *@Column*, které určují, jak mapovat entitu na databáze.

Na obrázku 4.3 je znázorněn příklad entity *Image*.

```
6   @Entity
7   @Table(name = "TO_IMAGE")
8   public class Image {
9
10      @Id
11      @GeneratedValue(strategy = GenerationType.IDENTITY)
12      private Long id;
        2 usages
13      @Column(name = "name")
14      private String name;
        2 usages
15      @Column(name = "filename")
16      private String filename;
        2 usages
17      @Column(name = "contentType")
18      private String contentType;
        2 usages
19      @Column(name = "size")
20      private Long size;
        2 usages
21      @Lob
22      @Column(name = "bytes", columnDefinition = "bigint")
23      private byte[] bytes;
```

Obrázek 4.3. Realizace entity Image.

4.4.4 Dto

Tento balíček obsahuje speciální třídy DTO sloužící k přenosu dat a reprezentující data z entit. DTO je datová struktura, která v sobě nemá žádnou logiku. [26]

4.4.5 Repository

Tento balíček obsahuje rozhraní, které slouží pro práci s daty v databázi. Rozhraní je označení anotace *@Repository*, která indikuje, že rozhraní slouží jako repozitář pro práci s daty.

4.4.6 Services

Tento balíček obsahuje třídy, které slouží pro implementaci business logiky aplikace a jsou označené anotací *@Service*.

4.4.7 Security

Tento balíček obsahuje soubory pro zajištění a nastavení autorizace a také autentizace pomocí JWT tokenu.

4.4.8 Controller

Tento balíček obsahuje třídy, které představují REST rozhraní aplikace a jsou označené anotací *@RestController*, která označuje, že bude třída schopna obsluhovat jednotlivé koncové body. [27] Každá entita má svůj kontrolér, který zpracovává HTTP dotazy.

4.5 Klientská část

Klientská část byla implementována jako jednostránková aplikace, která byla vyvíjena pomocí Javascriptu a knihovny React. Pro každou část uživatelského rozhraní byl vytvořen samostatný komponent reprezentující tuto část.

Hlavními komponentami jsou: Login, NavigationBar, Profile, Public, Questions, Quiz, Register, Reports, TestQuestions, Tests, Topics. Tyto komponenty reprezentují hlavní stránky aplikace. Také hlavní stránky obsahují jiné komponenty. Pro design uživatelského rozhraní byla použita open-source framework Bootstrap, která usnadňuje vývoj responsivních webových stránek pomocí kolekce syntaxe pro návrhy šablon. [28]

Pro komunikaci se serverovou částí byl použit HTTP klient Axios, který usnadňuje provádění HTTP dotazů a je dostupný jako knihovna pro JavaScript.

4.6 Nasazení na hosting

Nasazení webové aplikace na hosting proběhlo pomocí platformy Heroku, se kterou jsem se seznámil během výuky.

Heroku je cloudová platforma, která umožňuje nasazovat, poskytovat a škálovat aplikace. Platforma nabízí možnost pracovat s aplikacemi napsanými pomocí programovacích jazyků Ruby, Node.js, Java, Python, Clojure, Scala, Go a PHP. Také Heroku poskytuje databázové služby založené na PostgreSQL nebo Redis. [29–30]

Na platformu byla nasazena klientská a serverová část jako samostatná aplikace. Proces nasazení aplikace je popsaný níže:

1. Založení učtu Heroku
2. Vytvoření projektu na Heroku
3. Nastavení databáze
4. Připravení aplikace
5. Propojení s Git repozitářem
6. Odesílání kódu s Git repozitáře na Herkou
7. Spuštění aplikace
8. Otevření aplikace v prohlížeči

Po nasazení na hosting jsou klientská a serverová část aplikace dostupné na adresách uvedených v příloze B.

Kapitola 5
Testování

Testování představuje důležitou fázi ve vývoji softwaru, pomocí kterého můžeme odhalit chyby a defekty, které mohly vzniknout během implementace. Tato kapitola detailně popisuje, jak byla aplikace TestQuiz testovaná v průběhu i po ukončení implementace.

5.1 Jednotkové testování

Jednotkové testování je proces testování softwaru pomocí jednotkových testů neboli unit testů. Cílem jednotkových testů je odhalit chyby na úrovni kódu pomocí srovnání výsledku volání metody nebo procedury s očekávaným výsledkem pomocí vytvoření další části kódu, který kontroluje jednotlivé metody. [31]

5.2 Integrační testy

V integračních testech se testují interakce mezi různými částmi systému. Integrační testy můžou zahrnovat komunikaci s databází, webovými službami atd. [32] Cílem integračních testů je zjistit, jestli různé části systémů správně fungují jako celek.

Integrační testy byly prováděny manuálně pomocí nástrojů Postman pro kontrolu koncových bodů aplikace. Kromě toho, pomocí vývojářského nástroje v prohlížeči Firefox byla otestována interakce mezi klientskou a serverovou částí aplikace. Pomocí integračních testů bylo detekováno několik chyb, které byly následně opraveny.

5.3 Uživatelské testování

Cílem uživatelského testování je získat přehled o tom, jak budou reální uživatelé používat vytvořené řešení, a získat konstruktivní zpětnou vazbu ohledně jejich interakcí s aplikací. Uživatelské testování pomáhá určit možnosti dalšího zlepšení systému pro uživatele a detekovat chyby, které mohly uniknout při předchozích testech prováděných vývojáři či testery. Pro získání relevantní a přínosné zpětné vazby během uživatelského testování je důležitý správný výběr respondentů. Aplikace TestQuiz je především určena jako pomůcka během výuky na univerzitě, proto byli pro testování vybráni studenti z různých univerzit a studijních oborů.

Pro testování byl připraven testovací scénář zahrnující běžné úkoly, které bude provádět uživatel, a dotazník, který respondenti vyplní po splnění všech úkolů v testovacím scénáři.

Testovací scénář:

- Zaregistrovat se do aplikace
- Přihlásit se do aplikace
- Změnit heslo

- Odhlásit se z aplikace
- Přihlásit se do aplikace pomocí nového hesla
- Vytvořit nové téma a podtéma
- Přidat tématu a podtématu otázku s odpověďmi, alespoň jedna otázka a odpověď musí obsahovat obrázek
- Vytvořit test a přidat k němu vytvořené otázky
- Spustit vytvořený test, projít ho tak, aby alespoň jedna otázka byla špatně zodpovězená, a následně projít test znovu s pouze špatně zodpovězenými otázkami
- Zveřejnit vytvořený test
- Spustit libovolný veřejný test vytvořený jiným uživatelem
- Odhlásit se z aplikace

Dotazník:

- Použili byste webovou aplikaci TestQuiz jako pomůcku během studia?
- Narazili jste během testování na nějaké chyby? Pokud ano, popište chybu a vysvětlete, jak jste na ni narazili.
- Jakým způsobem se by mohla zlepšit vaše interakce s webovou aplikací TestQuiz?

5.3.1 Výsledky uživatelského testování

Uživatelského testování se zúčastnili 3 studenti. Výsledky testování jsou znázorněné pomocí zpětné vazby, kterou poskytli respondenti prostřednictvím dotazníku. Dotazník byl vyplněn každým respondentem po splnění testovacího scénáře. V tabulce 5.1 jsou uvedeny vyplněné dotazníky.

Respondent	Použili byste webovou aplikace TestQuiz jako pomůcku během studia?	Narazili jste během testovaní na nějaké chyby? Pokud ano, popište chybu a vysvětlete, jak jste na ni narazili.	Jakým způsobem se by mohla zlepšit vaše interakce s webovou aplikací TestQuiz?
Respondent č.1	Ano	Nezobrazoval se mi obrázek během testu	Předělat uživatelské rozhraní, pro telefony, je vice zaměřena pro používaní pomocí počítače
Respondent č.2	Ano	Po vytvoření nového testu se neaktualizovali testy a musel znovu natáčet stránku aby test bylo vidět	Bylo by fajn kdyby aplikace byla lokalizovaná nejenom anglicky
Respondent č.3	Ano, a i budu používat	Problémy ne, stejně nebyly i chyby. Chci ocenit minimalistický design a také možnost vkládaní obrázku. Jediná připomínka – během editace otázky nelze měnit obrázky, ale při běžném použití mi to vůbec nevadilo.	Chtěla bych možná mít aplikaci dostupnou i ve svém rodném jazyce, ale to, že je to v angličtině mi nevadí. Také bylo by pro mě skvělé, kdyby existovala možnost měnit velikost obrázku.

Tabulka 5.1. Vyplněné dotazníky.

5.4 Shrnutí testování

Tato kapitola se zabývala testováním vyvíjené webové aplikace TestQuiz. V této fázi byly použity různé způsoby testování, jako jsou jednotkové testy, integrační testy a uživatelské testování. Během této fáze byly opraveny některé chyby, například aktualizace seznamu testů při vytváření nového. Některé další problémy, které nebyly řešeny kvůli časové tísni, jsou popsány v kapitole 6.2.

Kapitola 6
Závěr

6.1 Shrnutí

Cílem této bakalářské práce byl návrh a vývoj webové aplikace TestQuiz pro tvorbu a vyplňování testů, která by během studia sloužila jako pomůcka studentům.

V práci byla provedena analýza již existujících řešení, která umožňuji vytvářet a procházet testy. Po provedené analýze hotových řešení byly definovány funkční a nefunkční požadavky, podle nichž byl vytvořen diagram případů užití, a byly zvoleny technologie pro implementaci budoucího řešení.

Na základě provedené analýzy proběhl návrh webové aplikace TestQuiz. Během návrhu byla zvolena architektura aplikace, navržena fyzická architektura, doménový model tříd a uživatelské rozhraní.

Dále bylo popsáno, jak byla aplikace TestQuiz implementována. V implementační části byly popsány použité nástroje a způsob zabezpečení aplikace a také struktura a způsob implementace serverové a klientské části aplikace. Na konci této části bylo popsáno nasazení hotového řešení na hosting. V poslední části byla popsaná testovací fáze. Byly zde popsány způsoby, jakými byla webová aplikace TestQuiz testovaná.

Největším přínosem bakalářské práce je samotná vyvíjená webová aplikace TestQuiz, kterou lze používat pro samostudium. Velmi přínosná pro mě byla také možnost vyzkoušet si díky znalostem získaným v rámci studia vytvořit webovou aplikaci.

6.2 Možnost dalšího vývoje

Webová aplikace TestQuiz funguje a splňuje všechny funkční a nefunkční požadavky, které byly specifikovány v rámci této bakalářské práce. Na základě výsledků uživatelského testování a vlastních nápadů jsou určeny možnosti pro další zlepšení a rozšíření funkcionality webové aplikace TestQuiz. První možnost pro další vývoj je opravit chyby a problémy, které se objevily během testování. Další možnost je rozšířit funkcionalitu pro vytváření a zpracování otázek a testů. Příklady rozšíření jsou znázorněny níže:

- Možnost zpracovat obrázky pro již vytvořené otázky
- Přidat možnost vytvářet různé druhy otázek
- Přidat možnost nastavit časovač pro testy
- Možnost sdílet test s určitým uživatelem
- Možnost ohodnotit testy
- Přidat možnost vyhledávání a řazení otázek a testů

Další možností je zlepšit interakci uživatele s webovou aplikací. Zaprvé nebyli někteří uživatelé spokojeni s tím, že aplikace obsahuje pouze jeden jazyk. Kvůli tomu bylo pro uživatele složité pochopit, jak se aplikace používá. Někteří uživatelé

také nebyli spokojeni s používáním TestQuizu na mobilním telefonu, proto je nutné opravit uživatelské rozhraní. Pro zvýšení spokojenosti uživatelů je nutné rozšířit možnosti pro správu uživatelského profilu, jako například změna hesla pomocí mailu v případě, že uživatel zapomene heslo, změna uživatelského jména a mailu a nakonec možnost smazání vlastního profilu.

Literatura

[1] Council of Europe(ECML/CELV). *Testmoz*. Online. https://www.ecml.at/Resources/InventoryofICTtools/tabid/1906/InventoryID/251/language/en-GB/Default.aspx. (cit. 2023-01-17).

[2] FlexiQuiz. *Help*. Online. 2023. https://www.flexiquiz.com/Help/. (cit. 2023-01-17).

[3] *Kahoot!* Online. 2023. https://kahoot.com/what-is-kahoot/. (cit. 2023-01-17).

[4] Kahoot!. *How it works*. Online. https://kahoot.com/schools/how-it-works/. (cit. 2023-01-17).

[5] IBM CORPORATION. *Use-case diagrams*. Online. 2016. https://www.ibm.com/docs/en/rational-soft-arch/9.7.0?topic=diagrams-use-case. (cit. 2023-01-17).

[6] IBM CORPORATION. *What is Java?* Online. https://www.ibm.com/topics/java. (cit. 2023-01-17).

[7] Oracle. *What is Python?* Online. 2022-05-31. https://developer.oracle.com/learn/technical-articles/what-is-python. (cit. 2023-01-17).

[8] Javatpoint. *Spring vs. Spring Boot vs. Spring MVC*. Online. https://www.javatpoint.com/spring-vs-spring-boot-vs-spring-mvc. (cit. 2023-01-17).

[9] Tutorials Point. *Django Framework*. Online. https://www.tutorialspoint.com/python_web_development_libraries/python_web_development_libraries_django_framework.htm. (cit. 2023-01-17).

[10] Zachary SHUTE. *Advanced JavaScript: Speed up Web Development with the Powerful Features and Benefits of JavaScript*. Online. 2019. (cit. 2023-01-17).

[11] Inc. Meta Platforms. *Tutorial: Intro to React*. Online. https://17.reactjs.org/tutorial/tutorial.html##what-is-react. (cit. 2023-01-17).

[12] Tutorials Point. *ReactJS - Overview*. Online. https://www.tutorialspoint.com/reactjs/reactjs_overview.htm. (cit. 2023-01-17).

[13] Cordenne BREWSTER. *What Is Vue.js? The Pros and Cons of Vue.js in 2022*. Online. https://www.trio.dev/blog/why-use-vue-js. (cit. 2023-01-17).

[14] Oracle. *What is a Relational Database (RDBMS)?* Online. https://www.oracle.com/database/what-is-a-relational-database/. (cit. 2023-01-17).

[15] Microsoft. *Nerelační data a NoSQL*. Online. https://learn.microsoft.com/cs-cz/azure/architecture/data-guide/big-data/non-relational-data. (cit. 2024-01-07).

[16] Oracle. *Oracle9i Application Server Oracle9iAS SOAP Developer's Guide: Simple Object Access Protocol Overview*. Online. https://docs.oracle.com/cd/A97335_02/integrate.102/a90297/overview.htm. (cit. 2024-01-07).

[17] IBM. *What is a REST API?* Online. https://www.ibm.com/topics/rest-apis. (cit. 2024-01-07).

[18] Hans-Jürgen SCHÖNIG. *Mastering PostgreSQL 10: Expert Techniques on PostgreSQL 10 Development and Administration*. Online. 2018. (cit. 2023-01--17).

[19] IBM CORPORATION. *What is three-tier architecture?* Online. https://www.ibm.com/topics/three-tier-architecture. (cit. 2023-01--17).

[20] IBM. *Deployment diagrams*. Online. 2023. https://www.ibm.com/docs/en/rational-soft-arch/9.7.0?topic=diagrams-deployment. (cit. 2024-01-07).

[21] Christina SRIVASTAVA. *How to get your team started in Miro*. Online. https://miro.com/blog/starting-remote-team-collaboration/. (cit. 2024-01-07).

[22] Red Hat. *What is an IDE?* Online. 2019. https://www.redhat.com/en/topics/middleware/what-is-ide. (cit. 2024-01-07).

[23] Postman. *Postman API Platform*. Online. https://www.postman.com/home. (cit. 2024-01-07).

[24] Okta. *Introduction to JSON Web Tokens*. Online. https://jwt.io/introduction. (cit. 2024-01-07).

[25] Shreya GHATE. *Using Session Cookies Vs. JWT for Authentication*. Online. https://hackernoon.com/using-session-cookies-vs-jwt-for-authentication-sd2v3vci. (cit. 2024-01-07).

[26] Tarnum Java SRL. *The DTO Pattern (Data Transfer Object)*. Online. 2022. https://www.baeldung.com/java-dto-pattern. (cit. 2024-01-07).

[27] ČVUT FEL. *REST API pomocí Spring Boot*. Online. 2022. https://cw.fel.cvut.cz/b222/courses/b0b36pjv/tutorials/11/java_rest. (cit. 2024-01-07).

[28] Jordana ALEXANDREA. *What Is Bootstrap?* Online. 2023. https://www.hostinger.com/tutorials/what-is-bootstrap/. (cit. 2024-01-07).

[29] HEROKU. *How Heroku Works*. Online. 2023. https://devcenter.heroku.com/articles/how-heroku-works##defining-an-application. (cit. 2024-01-07).

[30] HEROKU. *Databases Data Management*. Online. https://devcenter.heroku.com/categories/data-management. (cit. 2024-01-07).

[31] ČVUT FEL. *Testování softwaru: Jendotkové testování: JUnit, TestNG a základy efektivního návrhu*. Online. 2016. https://moodle.fel.cvut.cz/pluginfile.php/384123/mod_resource/content/1/TS1_prednaska_6_7_8.pdf. (cit. 2024-01-07).

[32] Miroslav BUREŠ. *TESTOVÁNÍ SOFTWARE: Unit testy - mockování, integrační testy*. Online. https://moodle.fel.cvut.cz/pluginfile.php/384174/mod_resource/content/1/TS1%20CV05%202022.pdf. (cit. 2024-01-07).

[33] Roger S. Pressman a Bruce Maxim. *Software Engineering: A Practitioner's Approach*. Publisher Name, 2014. ISBN 9780078022128. (cit. 2024-01-07).

Příloha A
Seznam zkratek

CSS	Cascading Style Sheets
DOM	Document Object Model
FRQ	Functional Requirements
HTML	Hypertext Markup Language
NFR	Non-functional requirements
PIN	Personal Identification Number
REST	Representational State Transfer
SQR	System Requirements
UML	Unified Modeling Language
XSS	Cross-Site Scripting
JSON	JavaScript Object Notation
JWT	JSON Web Token
API	Application Programming Interface
XML	Extensible Markup Language
DTO	Data Transfer Object
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment

Příloha B
Seznam odkazů

- Návrh uživatelského rozhraní: `https://miro.com/app/board/uXjVNEwMVJw=/?share_link_id=82376931729`
- Git repozitář:
  - Serverová část: `https://github.com/lylodmyt/backend`
  - Klientská část: `https://github.com/lylodmyt/frontend`
- Adresy nasazené aplikace:
  - Serverová část: `https://tqbackend-853fda4d5e9c.herokuapp.com`
  - Klientská část: `https://testquiz-f0f732432681.herokuapp.com`