České vysoké učení technické v Praze

Fakulta elektrotechnická

Bakalářská práce

Vasil Merkul

Návrh a vývoj web aplikaci "flash cards"

Katedra počítačů

Vedoucí: Ing. Božena Mannová, Ph.D.

Studijní program: Softwarové inženýrství a technologie

Praha 2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Merkul** Jméno: **Vasil** Osobní číslo: **452896**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Návrh a vývoj web aplikaci "flash cards"

Název bakalářské práce anglicky:

Design and development of web application "flash cards"

Pokyny pro vypracování:

Cílem práce je návrh a vývoj web aplikace pro práci s "flash cards". Aplikace bude sloužíí jako pomůcka pro výuku a přípravu ke zkouškám. Kartička (flash card) na jedné straně má pojem a na obrácené straně definici. Aplikace umožní vytvářet sety kartiček a systematizovat výuku.
Pro realizaci použijte pro front-end část single page aplikaci, která bude napsaná v Reactu.js. Pro serverovou část použijte Spring Boot. Data ukládejte do SQL databáze, jako ORM použijte Hibernate. Aplikaci navrhněte a implementujte s použitím vhodných prostředků SE. Aplikaci otestujte.

Seznam doporučené literatury:

[1] Pressmann R. S.: Software Enegineering,
[2] https://www.ankiapp.com/
[3] https://collegeinfogeek.com/flashcard-apps/

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Božena Mannová, Ph.D., kabinet výuky informatiky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **12.02.2021** Termín odevzdání bakalářské práce: **21.05.2021**

Platnost zadání bakalářské práce: **30.09.2022**

Ing. Božena Mannová, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

.	
Datum převzetí zadání	Podpis studenta

Poděkování

Rád bych tímto poděkoval vedoucí mého projektu Ing. Boženě Mannové Ph.D. za profesionální přístup, trpělivost, užitečné rady a cenné připomínky. Chtěl bych také poděkovat svým kolegům studentům za podporu a spolupráci.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje.

V ..... datum ..... podpis autora

Abstrakt:

Tato bakalářská práce je zaměřena na návrh a vývoj webové aplikace typu "flash cards", která bude sloužít jako pomůcka pro výuku a přípravu ke zkouškám. Kartička (flash čard) na jedné straně má pojem a na obrácené straně definice. V rámci dané práce byla provedena analýza existujících řešení a následné návrh a vývoj vlastního řešení. Výstupem této práce je webová aplikace, která umožňuje pracovat s kartičkami (flash cards) pro výuku.

Klíčová slova: webová aplikace, flashcards, samostudium.

Vedoucí: Ing. Božena Mannová, Ph.D

Abstract:

The Bachelor's thesis deals with design and development of the "Flash cards" type web application. The application is defined as a learning aid for studying and preparing for examinations. The flash card bears a question on the one side and an answer on the other. The thesis carries out an analysis of existing solutions and the subsequent design and development of the solution itself. The work output represents a web application allowing to work with flash cards for studying.

Keywords: web application, flash cards, self-studying

Supervisor: Ing. Božena Mannová, Ph.D

Obsah

1. Úvod

Tato bakalářská práce se zabývá analýzou metody výuky flashcards a následným návrhem a vývojem webové aplikace. Aplikace uživateli umožní systematizovat výuku a bude sloužit jako pomůcka pro přípravu ke zkouškám. Během rozhodování, které téma vybrat, se jich nabízelo několik. Pro konečné téma se autor práce rozhodl z několika důvodů:

- v minulosti používal podobné aplikace, které mu pomohly při učení
- znalosti, které získal během studia jsou dostatečné pro vývoj vlastního řešení
- výsledná aplikace se může v budoucnosti rozšířit o další funkční prvky

Cílem této bakalářské práce je vysvětlit způsob výuky s pomocí kartiček, popsat způsob použití a jeho výhody. Dalším cílem je porovnat na trhu existující aplikace jak pro konkrétní platformy (Android, IOS atd.), tak i multiplatformní, které jsou dostupné v prohlížeči na jakémkoliv zařízení. Finálním cílem je provedení návrhu a následná implementace vlastního řešení. V teoretické části je nejdříve popsána metoda výuky flashcards a následně provedení rešerše existujících řešení. Praktická část je věnována návrhu uživatelského rozhraní navržené aplikace, výběru technologií a implementaci. Výsledná aplikace byla uživatelsky otestována. Na základě provedeného testování autor popsal aktuální stav vyvinuté aplikace a možnosti jejího dalšího vývoje.

2. Flashcards

Flashcards je jednou z velice populárních a moderních metod pro výuku, která pomáhá zapamatovat si velké množství pojmů a definic za poměrně krátký čas. Na jedné straně kartičky je heslo a na straně druhé jeho překlad, vysvětlení nebo další doplňující informace. Kartičky se často používají pro výuku cizích jazyků, historických dat, formulí, ale budou vhodné pro učení širokého spektra látek.

Flashcards je cvičení pro aktivní opakování látky: student přečte pojem na jedné straně a musí ho definovat. Špatně zodpovězené kartičky se odkládají stranou k dalšímu procvičování.

Memorovací papírové karty mají hned několik výhod najednou. Jsou levné na výrobu. Vystačíte si s klasickým papírem, perem a případně pastelkami. Jsou univerzální. Lze se s nimi zkrátka učit cokoliv a kdekoliv. Díky své velikosti a skladnosti je můžete mít stále u sebe, takže jsou doopravdy účinné a vy navíc využijete efektivně svůj čas. Zároveň vám dělají učení jednodušší, protože nejste zahlceni informacemi litého textu, ale máte jej příjemně rozložen do karet s vysvětlivkami. Můžete si vyhrát s jejich použitím. Jednou si budete opakovat podle metody vodopád, jindy si můžete nejdříve přečíst odpověď.[1]

Mezi nevýhody papírových kartiček patří například nutnost vyrobit si je ručně, nebezpečí ztráty nebo poškození, dále pak složité úpravy již hotových kartiček nebo doplňování dalších údajů, některá technická omezení jako nemožnost použití audio nebo webového odkazu.

Aplikace typu flashcards mají stejné výhody jako papírové kartičky, navíc ale umožňují následující využití:

- Vytváření statistik
- Notifikace a jednodušší plánování výuky
- Jsou dostupné na vícero zařízeních
- Možnost sdílení sady karet se spolužáky nebo kolegy
- Zpestřují výuku formou hry
- Snadno se editují a sady karet lze rozšiřovat

3. Rešerše existujících řešení

Tato kapitola se věnuje již existujícím aplikacím. Trh v současnosti nabízí mnoho řešení pro různá zařízení (desktop, apple, android, prohlížeč), která disponují různými druhy funkcionalit a způsoby použití. Cílem této kapitoly je podat představu o nejpopulárnějších produktech a popsat jejich výhody a nevýhody.

3.1 Některá Existující řešení

3.1.1 Anki

Anki je jednou z nejpopulárnějších aplikací pro práci s flashcards. Nabízí možnost sdílení různých balíčků karet s pomocí databází, což může ušetřit čas při vytváření vlastních setů. Nabízí různé statistiky a umožňuje definovat vlastní denní plán výuky. Aplikace nabízí také hodně možností pro editování: Samotný editor databázových záznamů nabízí opravdu bohaté možnosti – kromě textu můžete vložit také vzorec, LaTeX kód, nahrát zvuk nebo editovat HTML, což dává využití kartiček zcela jiný rozměr – mohou být delší, strukturované a multimediální. To může být zajímavé jak pro samostudium, tak také při dalším šíření kartiček, jež jsou určeny například pro studenty.[2]

Anki nabízí práci s větším množstvím databází, synchronizaci jednotlivých databází přes připojenou webovou službu, práci se štítky, pluginy a řadu dalších užitečných funkcí, které jistě užije téměř každý, kdo se chce práci s flashcard věnovat intenzivně a soustavně. Nástroj je k dispozici jak pro všechny obvyklé platformy, tak také jako aplikace pro iPhone, Android či maemo.[2]

Výhody: skoro zadarmo (kromě IOS), více možností pro editaci a rozšiřování studijních materiálů, podpora práce s větším množstvím databází.

Nevýhody: designově nepříliš pěkné a jednoduché uživatelské rozhraní oproti konkurenci, aplikace není bezplatná pro IOS.

3.1.2 Quizlet

Quizlet je jednoduchá a uživatelsky přívětivá aplikace simulující zkušenosti s papírovými kartičkami. Bude vhodná pro uživatele, kteří hledají snadné a logické uživatelské rozhraní. Obsahuje také funkci přidávání obrázků, diagramů a vlastních nahrávek. V quizlet je možné sdílet studijní balíčky a také za doplatek využívat sady karet již připravených pro online sebevzdělání (khan academy, skill share atd.).

Pro uživatele, kteří studují cizí jazyky, jsou výhody oproti konkurenci následující: aplikace vyslovuje slova pomocí softwaru, ale hlas nezní příliš “roboticky”, odpadá nutnost přepínat klávesnici mezi různými jazyky, při psaní jsou k dispozici různé symboly.[3]

Další zajímavou vlastností aplikace je chytré hodnocení odpovědí, v případě překlepu si Quizlet všimne, že odpověď se podobá správné možnosti a uživateli nabídne možnost se ohodnotit sám.[3]

Většina těchto možností je bohužel dostupná pouze v placené verzi. Pro studium offline, bez reklamy, sledování zlepšování výkonu nebo použití obrázků při vytváření vlastních setů se musí zaplatit předplatné.[3]

Výhody: jednoduché a logické uživatelské rozhraní, extra možnosti pro jazykovou výuku.

Nevýhody: jen částečně zdarma.

3.1.3 Brainscape

Brainscape používá metodu výuky "spaced repetition" (systém intervalových opakování s použitím kartiček). Aplikace je podobná aplikaci Anky, navíc ale nabízí několik způsobů, které pomáhají sledovat vlastní zlepšování se. Vytváření kartových sad je velmi jednoduché a funguje na bázi jednoho sloupce s otázkami a druhého s odpověďmi. Pro vytváření karet s obrázkami nebo zvukem je potřeba pro verze.[4]

Své odpovědi uživatel hodnotí sám na stupnici jedné až pěti hvězd). Na základě odpovědí aplikace definuje úroveň znalostí a vyhodnotí časový interval potřebný pro další cvičení.[4]

Výhody: osobní výpočet intervalů potřebných k zopakování

Nevýhody: ve verzi zadarmo je možnost vytvářet pouze textové sady bez obrázků a audio nahrávek.

3.1.4 Cram

Cram je standardní flashcards aplikace, která nabízí různé způsoby výuky včetně her. Zajímavostí je možnost ke každé kartičce vytvořit nápovědu, což představuje výhodu ve chvíli, kdy se uživatel potřebuje sám otestovat. Kartičky, které uživatel odpoví správně se v testu již nezobrazují. Aplikace umožňuje výuku formou dvou her: "Jewels of Wisdom" a "Stellar Speller." Ačkoliv tato aplikace není tak pokročilá a nenabízí tolik možností jako Anki nebo Brainscape, určitě je vhodná pro studenty oceňující jednoduchost.[4]

Výhody: jednoduchost, která ale zároveň může být i nevýhodou.

Nevýhody: ve verzi zadarmo jsou reklama a omezené možnosti formátování.

3.1.5 IDoRecall

IDoRecall je zajímavá aplikace, které integruje se studijními materiály. Používá systém výuky "spaced repetition" podobný aplikaci Brainscape, ale s několika důležitými rozdíly.[4]

Hlavní vlastností aplikace je to, že IDoRecall byl vyvinut především k výuce akademických materiálů. Namísto toho, aby student musel vytvořit balíček karet sám, aplikace mu umožňuje nahrát potřebné učební materiály a vytvořit si z nich vlastní balíček. Kartičky následně odkazují na látku, z níž byly vytvořeny a pomůžou studentovi opakovat to, co zapomněl. Velkou výhodu představuje podpora mnoha formátů médií, jako například PDF, PowerPoint, Word a YouTube.[4]

Zajímavostí je, že aplikace umožňuje vytvářet studijní skupiny, ve kterých lze sdílet materiály a připravovat se ke zkouskam spolu s ostatními studenty.[4]

Výhody: podpora mnoha formátů médií.

Nevýhody: dostupnost pouze ve webové verzi.

3.1.6 Shrnutí

V této kapitole autor práce srovnal již existující aplikace, které jsou na trhu populární. Výsledky srovnání jsou zobrazeny v tabulce 3.1.

Aplikace	Anki	Quizlet	Brainscape	Cram	IDoRecall
Zdarma	Ano. Kromě IOS.	Jen částečně.	Jen částečně.	Jen částečně a s reklamou.	Jenom pro studenty. Jinak částečně.
Dostupnost	Android. IOS. Prohlížeč.	Android. IOS. Prohlížeč.	Android. IOS. Prohlížeč.	Android. IOS. Prohlížeč.	Prohlížeč.
Podpora média formátů	Ano	Ano	Ano	Ano	Ano
Podpora samotestování	Ano	Ano	Ano	Ano. Formou hry.	Ano.
Sledování progresu	Ano	Ano	Ano	Ne	Ano.

Tabulka 3.1: Srovnání aplikací

4. Návrh aplikace

Tato kapitola se věnuje návrhu uživatelského rozhraní a základní funkcionality aplikace Flashcards. Flashcards budou implementovány formou moderní webové "single page" aplikace dostupné v moderních prohlížečích jako jsou Google Chrome nebo Mozilla Firefox.

4.1 Návrh uživatelského rozhraní a funkcionalita

Aplikace bude mít následující základní vlastnosti:

1. Možnost přihlášení se a registrace nových uživatelů.
2. Možnost vytvářet sady kartiček.
3. Možnost editace již vytvořených setů kartiček včetně jejich mazání.
4. Režim výuky.

4.2 Uživatelské rozhraní

Uživatelské rozhraní by mělo mít moderní a logický uživatelsky přátelský design, aby umožňovalo orientaci bez návodu a uživatelské dokumentace. Verze pro mobilní prohlížeč bude jednodušší a bude určena pouze pro výuku.

Uživatel se bude muset nejdříve zaregistrovat, a to co nejjednodušším způsobem. Po registraci a přihlášení mu už budou dostupné všechny možnosti pro vytváření karet a také výuka s již vytvořenými kartami.

Po přihlášení bude mít uživatel možnost vytvářet, editovat nebo studovat již existující sety. Pro přidání nového setu bude stačit vybrat možnost "Add set" a vyplnit název nového setu. Na základě toho se vytvoří prázdný set a uživatel bude přesměrován na stránku pro jeho doplnění a editaci. Pro editaci již existujícího setu bude nutné vybrat požadovaný set ze seznamu a zvolit z menu "Edit set".

V režimu editace uživatel může kartičky doplnit, editovat nebo smazat. Pro smazání karty stačí stisknout "Delete" a karta bude následně ze sady vymazána. Pro přidání kartičky uživatel zvolí "Add card". Poté se zobrazí modální okno, ve kterém uživatel vyplní pojem a definici, na což bude karta přidána do setu. Pro editaci je potřeba zvolit "Edit" a pojem nebo definici upravit. Karta se posléze obnoví.

4.3 Shrnutí

Výhody řešení:

- Hlavní výhodou oproti ostatním řešením je to, že aplikace bude úplně zdarma a uživatele nebude při studiu obtěžovat reklama.
- Jednoduchost - uživatelské rozhraní bude logické a uživatelsky přátelské, nebude přetížené rušivými elementy a jeho design bude minimalistický.
- Projekt bude fungovat na bázi Open Source. Je proto důležité ho vytvořit tak, aby aplikace mohla být v budoucnosti rozšiřována o nové funkcionality, jako jsou například možnost využívání formátů médií nebo sdílení materiálů mezi spolužáky.

5. Vyber technologií

Tato kapitola porovnává populární jazyky a jejich framework pro serverovou a klientskou část. Kvůli možnosti ukládání dat bude provedeno také srovnání databázových systémů. Na základě provedené analýzy bude udělán výběr technologického stacku.

5.1 Výběr technologií

5.1.1 Frontend

Frontend je klientská část aplikace, která komunikuje se serverem a klientem s pomocí uživatelského rozhraní. Hlavním jazykem vytváření programu pro frontend je Javascript, a proto si shrneme několik nejpopulárnějších JS frameworků.

React

React vyvinula společnost Facebook s cílem vyřešit problém stálého přidávání nových vlastností do aplikace. Je jedním z nejjednodušších frameworků pro výuku. Jedná se o "open-source" framework. Od ostatních se odlišuje svým DOM (Document Object Model), který nabízí vynikající funkčnost. Je ideální volbou v případě vysokého objemu přenášených dat a potřebě stabilní a pro ovládání jednoduché platformy.[5]

Virtuální DOM odborně posouvá jak uživatele, tak také programátora. Virtuální DOM pomáhá obnovit libovolné uživatelské změny bez součinnosti ostatních částí uživatelského rozhraní, a to za podpory aplikace izolovaných komponentů. To představuje pro uživatale představuje výhodu v tom, že virtuální DOM pomáhá zlepšovat jejich zkušenosti v reálném čase.[6]

Velkou výhodou Reactu je opakovaná možnost použití jeho komponentů - React pracuje s izolovanými komponenty, což znamená, že vývojář je může kdykoli použít znovu.[6]

Podmínkou pro práci s komponenty je stabilní kód na bázi jednosměrného proudu dat. Pro globální stav aplikace se používá knihovna Redux, přičemž každý komponent je nastavitelný tak, aby reagoval na změny globálního stavu.

Vlastnosti:

- DOM model zajišťuje výkon webové aplikace.[5]
- Nabízí opakovanou možnost použití jeho komponentů.[5]
- JSX - možnost kombinovat HTML a JavaScript kód.[5]
- K dispozici má knihovny s velkým množstvím nástrojů a hotových řešení.
- Pracuje s ním široká komunita vývojářů.

Angular

Angular - je framework používající TypeScript (vyvinutý na základě JavaScript a poskytující přísnou typizaci). Byl představen v roce 2016 Googlem. Jeho účelem bylo setření rozdílu mezi neustále rostoucí poptávkou nových technologií a tradičním přístupem, což přineslo své výsledky.

V porovnání s Reactem je Angular jedinečný tím, že nabízí dvě možnosti přenášení dat. To znamená, že umožňuje synchronizaci mezi Modelem a View v reálném čase, přičemž libovolná změna v Modelu se okamžitě promítne ve View a naopak.

V porovnání s Reactem není Angular tak jednoduchý pro výuku. Existuje však velké množství dokumentace, která ale může být příliš komplikovaná.

Vlastnosti:

- Funkcionalita pro promítání změn provedených v Modelu do View a naopak.
- Oddělení component od závislosti pomoci definovani jich jako externich elementu.
- Možnost opakovaného použití komponentů.
- Široká komunita vývojářů.
- TypeScript.

View JS

Vue Js je v současnosti jedním z nejpopulárnějších frontend frameworků. Jedná se o jednoduchý a přímočarý framework. Výhodou tohoto frameworku je to, že vývojáři ho ve srovnání s Angularem dokázali zjednodušit. Je menší a nabízí dvě hlavní výhody: vizuální DOM a komponentní přístup.[5]

Vue Js je univerzální a využitelný pro různé projekty - od stavby webových a mobilních aplikací až po realizaci progresivních webů. Dokáže si snadno poradit s snadnými a dynamickými procesy.[5]

Vue Js byl navržený jako framewor, který spojí výhody Reactu a Angularu.

Vlastnosti:

- Rozsáhlá a detailní dokumentace.[5]
- Jednoduchá syntax - programátoři, kteří mají znalosti JavaScriptu, ho mohou snadno začít programovat.[5]
- Flexibilní pro design aplikační struktury.[5]
- Podpora TypeScript.[5]
- Poměrně malá komunita vývojářů.[5]

5.1.2 Backend

V této podkapitole je uveden výběr serverových jazyků potřebných pro komunikaci serveru s databází a aplikací. Často používané jazyky v této oblasti jsou: C#, PHP, Python a Java. V poslední době je stále populárnější JavaScript a Node.Js.

Pro realizaci serverové části aplikace se autor práce rozhodoval mezi následujícími jazyky, frameworky a platformami, se kterými měl alespoň nějaké zkušenosti: C# a .NET, Java a Spring, JavaScript a Node.Js. Pro výběr jazyka byl kladen důraz hlavně na jednoduchost vývoje, možnost následného rozšíření a podporu, velikost komunity vývojářů.

JavaScript, Node.Js a Express

JavaScript je programovací jazyk pro frontendovou část aplikace, ale v současnosti se používá nejen pro frontend, ale také pro backend. S pomocí Node.Js lze psát v JavaScript také serverovou část.

Node.Js je JavaScript běhové prostředí postavené na V8 JavaScript engine pro Chrome. Node.Js poskytuje "event driven", asynchronně vstupní a výstupní prostředí pro stavbu vysoce škálovatelné serverové části aplikace, díky čemuž je velmi efektivní a jednoduchý. NPM je správce balíčku pro Node.Js a je největším ekosystémem otevřených knihoven na světě.[7]

Express je rychlý a jednoduchý framework pro Node.Js. Poskytuje širokou škálu základních prvků pro webové aplikace, aniž by zastíral výhody Node.Js. Kromě toho umožňuje snadno postavit plné API, a to s pomocí nejrůznějších HTTP metod a dostupného middleware.[8]

K dispozici jsou také nadstavby pro Express, například Ts.ED, usnadňující vývoj aplikací.

Příklady webů používajících Express: Uber, Accenture, IBM.[8]

Vlastnosti:

- Vynikající routovací API.[8]
- Minimalistický a nepřetížený framework.[8]
- Jednoduchý k učení a nenáročný pro začátečníky.[8]
- Disponuje velkým množstvím pluginů, které lze využít během vývoje.[8]

Java a Spring Boot

JavaScript je programovací jazyk pro frontendovou část aplikace, ale v současnosti se používá nejen pro frontend, ale také pro backend. S pomocí Node.Js lze psát v JavaScript také serverovou část.[9]

Spring Boot je framework vytvořený na základě Java. Byl vyvinut společností Spring s cílem snadnějšího použití a umožnění rychlého vytvoření funkcí v aplikaci. Spring Boot ma svuj pohled na platformu spring a dalsi knihovny a diky tomu se da se Spring Boot zacit velice snadno, spoustu veci udela za vyvojare.[8]

Mezi výhody Spring Boot patří: není nutná šablonová konfigurace, snadný management závislosti, jednoduché nastavení vlastností aplikace, možnost vytvořit funkční zaklad (ve stylu "Hello World") a propojit nutné závislosti přímo na webu, vestavěná podpora Tomcat.[10]

- Umožňuje vytvářet "stand-alone" Spring aplikace.[8]
- Je vysoce škálovatelný.[8]
- Rozsáhlá dokumentace.[8]
- Postavený pro velké aplikace, které používají "cloudový" přístup.[8]
- Rozsáhlý ekosystém.[8]
- Je multiplatformní.

C# a .NET

C# je vysokoúrovňový jazyk, programování v něm se syntakticky velmi podobá Jave. Díky tomu je přechod z Javy do C pro programátora rychlý a snadný. Programátoři si syntax v C# oblíbili. Podle některých je lepší než Java

ASP.NET je pro Oracle s Javou konkurenčním jazykem společnosti Microsoft. Jedná se o webový framework pro programování v jazycích jako jsou Visual Basic, C#, F# a další.[9]

V roce 2016 se .NET stal "open source", díky čemuž je možné ho integrovat s iOS, Linux a Android, a to s pomocí .NET Core. Kód je stabilní a spolehlivý, díky čemuž je .NET a C# populární volbou pro byznys. Protože se jedná o produkt společnosti Microsoft, je programátorům v případě nejasností k dispozici silná podpora.[9]

Jednou z klíčových vlastností .NET Core frameworku je jeho velká výkonnost, což z něj dělá jedno z nejflexibilnějších řešení současnosti[10]

- Velký výkon.[10]
- Asynchronnost. Široká podpora a možnost využití patternu asynchronního.[10]
- Multiplatformní.[10]
- Na rozdíl od Javy progresivnější syntax

5.1.3 Databáze

Databázové systémy umožňují více uživatelům zároveň updatovat, editovat a aktualizovat uložené informace rychle, bezpečně a efektivně. Proto jsou tyto systémy užitečné pro různé typy požadavků, jako jsou správa uživatelů, ukládání velkého množství dat nebo využití pro vývoj webových aplikací. Jsou různé typy moderních databázových systémů.[11]Každý z nich má svá pozitiva a negativa.

SQL a NoSQL databáze

SQL

SQL databáze neboli relační databáze představují kolekci tabulek propojených mezi sebou, přičemž každý z řádků v tabulce je záznamem. Tomuto typu databází se říká relační, protože tabulky jsou mezi sebou propojené pomocí klíčů. Každý záznam v tabulce obsahuje klíč, který odkazuje na další tabulku.[12]

Pro dotazování se v relačních databázích používá jazyk typu SQL, který umožňuje indexaci sloupců v tabulkách pro rychlé vyhledávání v často užívaných datech.[12]

Vzhledem k tomu, že základem pro relační typy databází je struktura, schéma databáze se řeší již před samotným vložením dat.[12]

NoSQL

Na rozdíl od relačních databází, jejichž obsah je strukturovaný do řádků a sloupců, v NoSQL databázích neexistuje obecné schéma pro všechny záznamy. Většina NoSQL systémů obsahuje záznamy ve formátu JSON a různé záznamy mají ruzná políčka.[12]

Přestože se tento typ databází nazýva NoSQL, podporuje také SQL dotazování.[12]

Nejpopulárnějším typem NoSQL databáze je na dokumenty orientovaný typ. Každý jednotlivý prvek tvoří v databázi dokument, každý dokument je ve formátu JSON. Schéma mezi dokumenty se může lišit a obsahovat různé typy dat.[12] V současnosti jsou populární tato řešení: MongoDB, CouchDB, DocumentDB.[12]

5.1.4 Jednotkové a integrační testy

Jednotkové testy

Jednotkový (unit) test je speciální druh testu, ve kterém se testuje jedna jediná komponenta a všechny její vnější závislosti jsou simulovány. Takový test má za úkol zadanou komponentu důkladně otestovat v běžných i náročných podmínkách. Pokud jsou takto otestovány všechny komponenty, je celkem vysoká pravděpodobnost, že program jako celek bude fungovat dobře.[13]

Pro psaní integračních testů v jazyce Java je nejpopulárnější technologií JUnit.

Knihovna JUnit slouží k implementaci a provádění jednotkových a integračních testů. Kromě podpory pro implementaci a provádění testů obsahuje také mnohé funkce pro kontrolu a porovnávání hodnot. Celá knihovna je navržena tak, aby byl kód přehledný a obsahoval minimální množství infrastrukturního kódu.[13]

V JUnit je každý test reprezentován třídou, jejíž název je odvozen z názvu testované třídy a příponou Test (například CalculatorTest, UserManagementServiceTest). JUnit od verze 4 může být zcela řízen anotacemi, takže testovací třída nemusí od žádné jiné třídy přejímat a ani není nutné ji přidávat do seznamů, jako tomu bylo dříve. Testovací třída obsahuje pouze samotný test, případně přípravu testovacích dat simulovaných závislostí.[13]

Integrační testy

Pro testování interakce komponentů jako celku se používají integrační testy.

Integrační testy obvykle postupují od jednotlivých modulů směrem k věštím celkům. Nejdříve se testují rozhraní jednotlivých modulů. Mnohdy jsou v této fázi využívány tzv. fake moduly. Jde o simulátory, jejichž úkolem je napodobovat komunikaci ostatních modulů, ale bez jejich funkčnosti. Díky nim se ověří, že modul umí korektně odesílat i přijímat komunikaci s dalšími moduly.[14]

Mockito je testovací framework pro tvorbu mocků, tedy zástupných objektů, které se používají při tvorbě komponentových testů. Hlavním rysem každého mocku je to, že se navenek tváří jako obyčejný objekt a přitom můžeme libovolně definovat jeho chování. Tak lze jednoduše nasimulovat mnohem více testovacích případů než bez nich a nároky na modularitu zdrojového kódu nejsou tak veliké.[15]

Ve výchozím stavu je mock prázdný a nemá definované žádné chování. To znamená, že žádná metoda nic nedělá a pokud má návratovou hodnotu, vrací null. Chování mocku lze specifikovat později.[15]

V testech se mocky znovu vytváří zpravidla pro každé provedení testu, aby se pro každý testovací případ mohlo definovat jiné chování mocku a testovací případy se vzájemně neovlivňovaly.[15]

Podobnou funkcionalitu jako Mockito poskytuje i framework Spring MVC Test.

Spring MVC Test framework také známý jako MockMvc, poskytuje podporu pro testování Spring MVC (Model Predstaveni Kontroler) aplikaci. Framework poskytuje možnosti handlovani dotazu pomoci mockování dotazu a odpovědi, bez toho aniž by server běžel.[16]

5.1.5 Shrnutí

Na základě provedené analýzy se autor práce rozhodnul vybrat následující technologický stack: Java + Spring pro backend, React.JS a Redux pro frontend, a PostgreSQL pro ukládání dat.

Pro serverovou část jsem rozhodoval mezi: C a .NET, Java a Spring, JavaScript a Node.Js. Node.JS je dobrá volba pro jednoduché řešení. Node.JS není tolik složitý jako například Java se Spring, ale jeho nevýhoda je, že funkční kód píše velmi nepraktickým způsobem, což by představovalo překážku pro další podporu a případné rozšíření aplikace. C a .NET je jako jazyk lepší a modernější než Java, ale díky stáří Javy, tato disponuje mnohem širší komunitou programátorů. Proto se autor práce rozhodl pro Java a Spring. Krome toho nejpopularnejsi vyvojove prostredi Intellij IDEA pro Java je mnohem pohodlnejsi nez VisualStudio pro C.

Klientská část bude napsaná v React.JS. React je v porovnání s Angular poměrně jednoduchá technologie, která současně disponuje velkou vývojářskou komunitou. To bude velkou výhodou při implementaci. Vue.JS by ve srovnání s Reactem mohl být také jednoduchým a vhodným řešením, ale prozatím mezi vývojáři není tak populární. Kvůli tomu nemá tak rozšířenou podporu a jeho budoucnost není zcela jasná. Velkou výhodou ReactJs je objektový přístup.

Pro ukládání dat autor práce zvolil PostgreSQL, protože s ním má největší zkušenost a nevidí důvody pro preferenci NoSQL DB. Pro mapování tabulek do objektu se použije Hibernate. Hibernate umožňuje snadno reprezentovat databázovou tabulku jako třídu s pomocí různých anotací, díky čemuž se už dlouho a standardně používá pro ORM (object relational mapping) a využití nenajde jen v Javě.

Vzhledem k tomu, že pro vývoj backendové části aplikace byl vybrán jazyk Java a Spring Boot framework, pro integrační a unit testy jsem zvolil nejoblíbenější knihovnu JUnit. Spring MVC Test framework byl vybrán pro mockování chování testované funkcionality.

6. Implementace

V této kapitole budou popsány: architektura aplikace, implementace jednotlivých součástí a použité technologie. Výsledná aplikace se skládá z následujících částí: frontendová část, backendová část, databázová vrstva.

6.1 Frontend

Frontendová část byla implementovaná jako jednoduchá jednostránková aplikace. Pro její vývoj byla použita JavaScript knihovna React.Js a Redux pro práci s globálním stavem aplikace.

Hlavní prvky: záhlaví s menu stránky a komponenty pro vytváření a editaci setu karet, registrační a přihlašovací formuláře. Každý komponent se skládá z menších atomických komponentů.

Pro ukládání globálního stavu byla použita knihovna Redux. Pro různé komponenty byly napsány funkce mapStateToProps a mapDispatchToProps. MapStateToProps je funkce, která předává komponentu potřebná data z globálního úložiště. MapDispatchToProps je funkce, kterou komponent používá ve chvíli, kdy vznikne potřeba provést obnovení globálního stavu.

Komunikace se serverovou částí zajišťuje technologie Axios. Axios je jednoduchý HTTP client, který se používá pro frontendové aplikace. Komponenty komunikující se serverem používají "hooks". Byl implementován Hook component WithCardService, což je funkce, kterou se dá "obalit" jakýkoliv klasický class komponent. "Obalený" komponent pak má možnost volat server.

6.2 Backend

Backendová část byla implementována v Javě s pomocí frameworku Spring Boot. Pro dotazování ze strany frontendu byla vystavena sada endpointů, které byly implementovany v třídách kontrolorů.

Pro zasílání dat na frontend bylo implementováno několik tříd, které obsahují metody-endpointy. Pro vrácení uživatelských dat, jako jsou jméno nebo e-mail, slouží třída UserController. Pro vrácení setu karet a jednotlivých karet slouží třídy CardSetController a CardController.

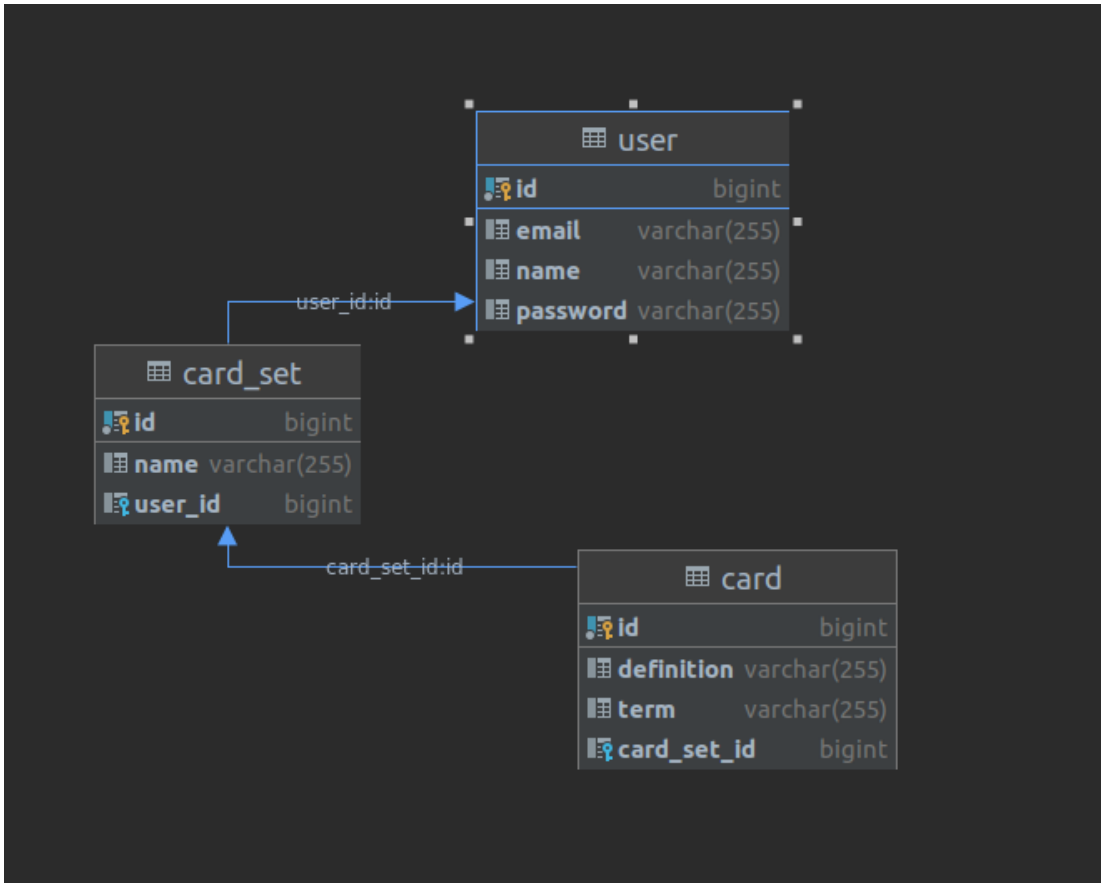
Každý z kontrolorů využívá třídu typu Repository. Třídy typu Repository jsou zděděny od třídy CrudRepository a poskytují interface pro aktualizaci, přidávání a mazání dat z databázových tabulek popsaných dříve. Příklad třídy typu repository je na obrázku 6.1.

```
public interface CardSetRepository extends CrudRepository<CardSet, Long> {
}
```

Obrázek 6.1: Příklad třídy typu repository

6.3 Databáze

Pro ukládání dat byla použita relační databáze PostgreSQL. Její schéma se skládá z následujících tabulek: user, card_set, card. Uživatele představuje tabulka user, která má atributy: id, name, email, password. Sadu karet uživatele představuje tabulka card_set s atributy: id, name a cizí klíč user_id. Klíč user_id odkazuje na tabulku user a řeší, která sada patří konkrétnímu uživateli. Tabulka card reprezentuje flash kartičku a má následující atributy: id, term, definition a cizí klíč card_set_id, který odkazuje na card_set a určuje, do jaké sady kartička patří. Schéma databáze je vyobrazeno na obrázku 6.2.



Obrázek 6.2: Schéma databáze

6.4 Jednotkové a integrační testy

Pro implementaci integračních testů byla použita knihovna JUunit. Pro mockování testovacích dat autor práce použil framework MockMvc od Spring.

V testech kontrolerů byly s pomocí anotace Autowired vytvořeny instance třídy typu Repository, s jejichž pomocí byla následně testovací data uložena. S pomocí anotace Autowired byly vygenerovány také instance třídy MockMvc, které umožňují simulovat dotazy pro ukládání nebo aktualizaci a mazání již uložených dat.

7. Testování

Testování bylo provedeno ve dvou fázích.

První fáze probíhala v průběhu vývoje a jejím cílem bylo odhalit zásadní problémy v používání aplikace. Tato fáze pomohla odstranit největší chyby, a to ještě před samotným představením aplikace koncovým uživatelům.

Druhá fáze testování byla uživatelská. Autor práce požádal několik lidí o vyzkoušení hlavních prvků a funkcionalit aplikace a jejich zpětnou vazbu.

Hlavní problém, který byl během testování uživateli zjištěn, je okamžité smazání dat po stisknutí tlačítka "Delete". Většina uživatelů by preferovala, aby smazání dat proběhlo až po zobrazení dialogového okna s otázkou "Opravdu chcete smazat?" a výběrem možností mezi "Ano, smazat" nebo "Storno".

Další nedostatek a nápad pro zlepšení představuje podle uživatelů to, že aplikace podporuje pouze výchozí jazyk.

Celkem se testování zúčastnilo osm osob. K získání zpětné vazby je autor požádal o vyplnění krátkého dotazníku. Výsledky dotazníku zobrazil v tabulce. V tabulce josu uvedené odpovědi pouze tří respondentů, protoze odpovědi ostatních byly vesměs stejné.

Otázka	Respondent č.1	Respondent č.2	Respondent č.3
Měl(a) jste problém orientovat se v aplikaci?	Žádný problém jsem nemel.	Neměl jsem.	Neměla jsem.
Jak se vám líbil vzhled uživatelského rozhraní?	Dal by se vyĺepšit.	V pořádku. Minimalistický design.	Spokojenost.
Chybí podle vás v aplikaci nějaké funkce?	Dialogové okno před smazáním dat.	Pokročilejší režim pro výuku. Testy pro samokontrolu.	Podpora více jazyků. Testy pro samokontrolu.
Jak jste celkově spokojen(a) s funkcemi pro editaci a vytváření sad karet?	V pořádku.	Spokojenost.	V pořádku.
Jak byste celkově ohodnotil(a) webovou aplikaci Flashcards?	5/5	4/5	4/5

Tabulka 7.1: Odpovědi z dotazníku

8. Aktuální stav a možnosti rozvoje vyvinuté aplikace

V rámci této bakalářské práce se povedlo vytvořit jednoduchou aplikaci se serverovou a klientskou částí. Aplikace je snadno škálovatelná, a proto má potenciál pro další rozvoj. Aplikace by se mohla stát součástí výukového procesu, mohli by ji používat nejen studenti, ale také učitelé. Učitelé by s její pomocí mohli vytvářet například soubory důležitých definic pro své studenty.

Navigace

Navigace je řešena pomocí horního panelu (záhlaví). Záhlaví obsahuje dvě menu, která lze rozkliknout:

- Mange sets
- Learn sets

Manage Sets po rozkliknutí zobrazí seznam vytvořených setů dostupných pro editaci. V menu bude také možnost výběru "Create Set".

Learn Sets je seznam uživatelem vytvořených sad karet. Po rozkliknutí konkrétního setu v menu se otevře stránka s příslušnou sadou karet pro výuku.

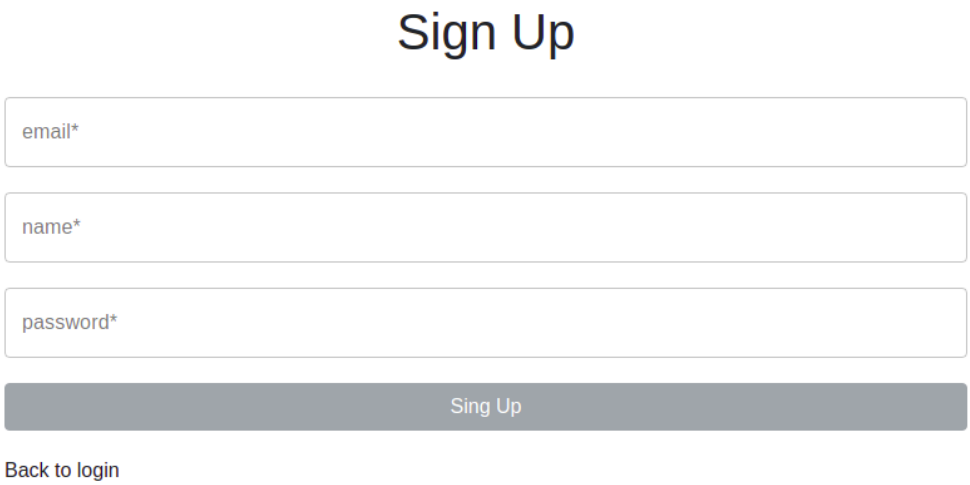
Funkčnost

Aplikace má základní funkce pro:

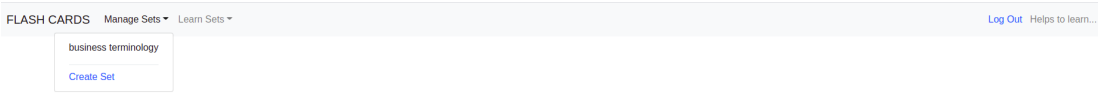
- Přihlášení a registraci
- Tvorbu setu karet
- Editaci a smazání karet
- Výuku pomocí kartiček

Uživatelské rozhraní

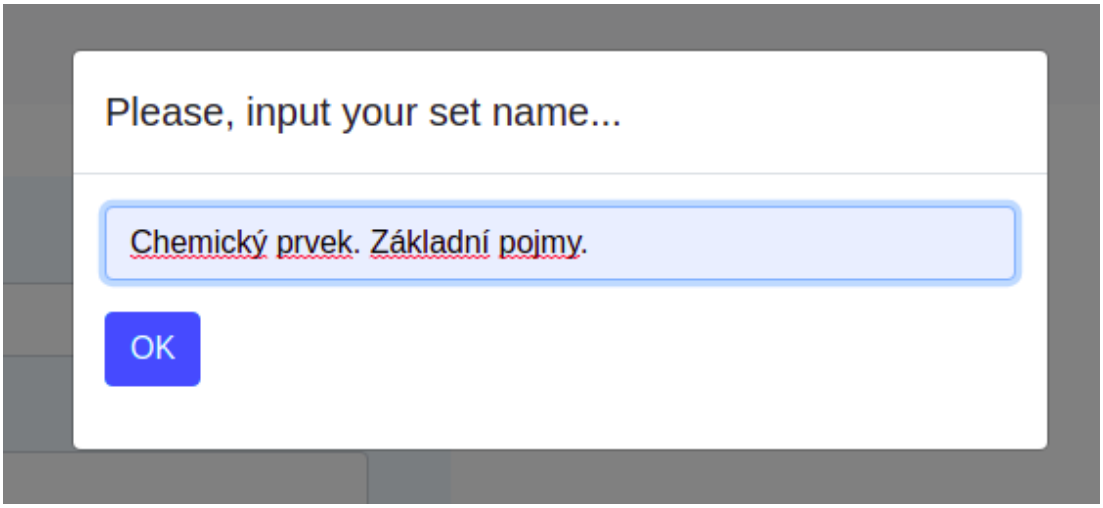
Příklady uživatelského rozhraní jsou vidět na obrázkách:



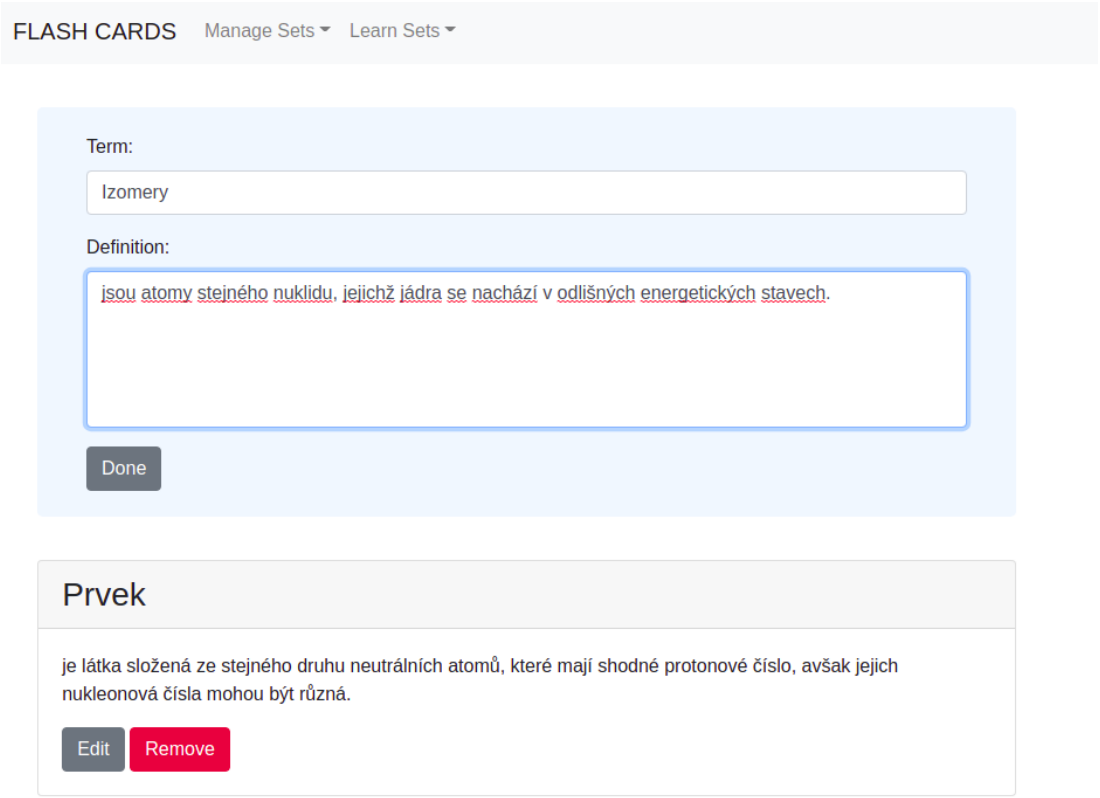
Obrázek 8.1: Registrace



Obrázek 8.2: Hlavička



Obrázek 8.3: Tvorba nového setu



Obrázek 8.4: Vytvoření a úprava kartiček

8.1 Možnosti dalšího rozvoje

První možností pro další rozvoj aplikace je odstranění aktuálních problémů, které se objevily během uživatelského testování. Uživatelé nebyli spokojení především s tím, že, aplikace má pouze výchozí jazyk - angličtinu. Pro podporu překladu v React jsou už hotová řešení, například react-i18next. Další možnost pro vylepšení aplikace, která vyplynula z testu uživatelů, je přidání dialogového okna s potvrzením, zda-li uživatel požadovaná data chce opravu smazat.

Také možnost vytvářet statistiky by výrazně zlepšila uživatelské zkušenosti a spokojenost s aplikací. Možnost pozorovat svůj postup by byl zároveň motivací pro pokračování. Osobní týdenní nebo měsíční zprávy a grafy, které by ukázaly uživatelův progres a upozornily by ho na nedostatky, by mohly být dostupné přímo v aplikaci, a nebo formou pdf zprávy, kterou by uživatel obdržel e-mailem. Tvorba grafu je snadná k implementaci, a to s pomocí JavaScript knihoven, například Chart.Js. Pro vytváření pdf zprávy jsou JavaScript knihovny také vhodné, a to jak pro Node.Js, tak i na straně frontendu. Nicméně pro tyto typy zpráv by bylo potřeba vytvořit jednoduchou službu navíc, ve které bude implementován matematický model pro počítání statistik. Pro ukládání statistik je nutné rozšířit databazovy model.

Za účelem zvýšení uživatelské spokojenosti by se aplikaci dalo rozšířit o možnost zasílání upozornění a připomínek. Uživatel může dostávat e-mailem upozornění na to, že mu zbývá naučit se zbytek látky nebo s vysvětlením pojmu z flash karty, který pro něj byl obtížný. Uživatel musí mít možnost notifikace sám nastavovat.

Další možností pro rozšíření funkcionality aplikace je přidání obrázku pro konkrétní kartičku. Tato funkcionalita pomůže výrazně rozšířit oblast použití flash karet. Obrázky umožní zpřehlednit definici pomocí schémat, grafů atd. Nejdůležitějším úkolem pro přidání takové funkcionality bude správná volba technologie pro ukládání obrázků a následná implementace, a to z důvodu, že každý obrázek více paměťového místa než text.

Lze také uvažovat o vytvoření algoritmu pro generování testu na základě vytvořených sad karet pro samokontrolu. Test bude vygenerován následující formou: otázka - pojem a poté možnost výběru z několika definic. Pro složitější test by bylo možné napsat definici ručně, případně by se uživatel mohl ohodnotit sám. To v případě, že napíše odpověď správně, ale s malým překlepem.

Z hlediska architektury aplikace by se dalo uvažovat o přidání orchestrační vrstvy. V případě, že bude vyvinuta služba pro generování pdf zprávy, mohl by se na ni dotazovat frontend. Tato praxe ale není nejlepším řešením. Mnohem vhodnější by bylo implementovat orchestrator, který by poskytoval interface pro frontend. Jakýkoliv dotaz by prošel tímto orchestratorem, který by poté dotaz validoval a zavolal příslušnou službu: bud' aktuální flashcards backend, nebo pdf službu, která vygeneruje a vrátí report. Orchestrator by bylo vhodné zároveň použít i pro autentifikaci.

9. Závěr

Cílem této bakalářské práce byl návrh a vývoj webové aplikace pro práci s flash kartičkami. Aplikace by měla sloužit jako pomůcka pro výuku a přípravu ke zkouškám. V této bakalářské práci autor nejprve provedl analýzu metody výuky flashcards a a dnes na trhu existujících řešení. Následně byl proveden návrh vlastního řešení. Výsledkem této práce je implementovaná jednoduchá aplikace, která se skládá z klientské a serverové části. Aplikace byla otestována a a hlavní cíle projektu byly splněny.

V teoretické části této práce byla vysvětlena metoda výuky flashcards, k čemu je dobrá a její výhody. Následně byla provedena rešerše současných elektronických řešení. U existujících řešení byly popsány jejich klady a zápory.

V praktické části této práce její autor provedl návrh aplikace. Pro backend část byla provedena analýza populárních jazyků a frameworků pro backend. Pro klientskou část autor srovnal nejpopulárnější JavaScript frameworky. Na základě provedené analýzy byl proveden výběr technologického stacku pro vývoj aplikace.

Pro implementaci byl použit JavaScript framework React JS pro klientskou část a Java se Spring Boot frameworkem pro serverovou část. Pro ukládání dat autor zvolil PostgreSQL databázi a Hibernate pro mapování databázových entit. V kapitole 5 byl popsán aktuální stav vyvinuté aplikace. V kapitole 7 autor popsal možnosti a nápady pro další možný rozvoj aplikace.

Největší přínos této bakalářské práce byla pro autora možnost vyzkoušet celý proces vytváření funkční aplikace: od analýzy a návrhu uživatelského rozhraní, po volbu technologií a následnou implementaci. Autor této práce si velmi cení možnosti vyzkoušet pozici full stack vývojáře.

Literatura

[1] *Metoda učení pomocí memorovacích kartiček*. Available online: https://bobo.cz/blog/metoda-uceni-pomoci-memorovacich-karticek. BOBO BLOK spol. s r.o.

[2] Michal Černý. *Učte se s flashcards*. Available online: https://www.root.cz/clanky/ucte-se-s-flashcards/. 22.2.2011.

[3] Jill Duffy. *Quizlet Review*. Available online: https://www.pcmag.com/reviews/quizlet. 6.10.2020.

[4] Ransom Patterson. *These Flashcard Apps Will Help You Study Better in 2021*. Available online: https://collegeinfogeek.com/flashcard-apps/. 9.04.2021.

[5] Hiren Dhaduk. *TBest Frontend Frameworks of 2021 for Web Development*. Available online: https://www.simform.com/best-frontend-frameworks/. 2021.

[6] *Top Front-End Frameworks in 2021*. Available online: https://www.simform.com/best-frontend-frameworks/. 14.01.2021.

[7] *What is Node.js?* Available online: https://www.tutorialsteacher.com/nodejs/what-is-nodejs.

[8] Martin Williams. *Top 8 Best Backend Frameworks*. Available online: https://www.keycdn.com/blog/best-backend-frameworks. 14.09.2020.

[9] Krystal Tolani. *The Beginner's Guide to Backend Development (2021 Guide)*. Available online: https://learntocodewith.me/posts/backend-development/#java. 3.05.2020.

[10] Jessica Clark. *Top 10 backend frameworks*. Available online: https://blog.back4app.com/backend-frameworks/.

[11] Donal Tobin. *Which Modern Database Is Right for Your Use Case?* Available online: https://www.xplenty.com/blog/which-database/#overviewofmostpopular. 10.06.2020.

[12] Tzoof Avny Brosh. *How to Choose the Right Database*. Available online: https://towardsdatascience.com/how-to-choose-the-right-database-afcf95541741. 1.09.2019.

[13] Vojtěch Hordějčuk. *JUnit 4*. Available online: http://voho.eu/wiki/java-junit/.

[14] *Druhy testování v procesu vývoje SW*. Available online: http://test.swtestovani.cz/index.php?option=com_content&view=article&id=18:druhy-testovani-v-procesu-vyvoje-sw&catid=3:zaklady&Itemid=11.

[15] Vojtěch Hordějčuk. *Mockito*. Available online: http://voho.eu/wiki/mockito/.

[16] *3.7. MockMvc*. Available online: https://docs.spring.io/spring-framework/docs/current/reference/html/testing.html#spring-mvc-test-framework.

[17] *AnkiApp*. Available online: `https://www.ankiapp.com`.

[18] Roger Pressman a Bruce Maxim. *Software Engineering: A Practitioner's Approach*. McGraw-Hill Education, 2014.

Seznam obrázků

Seznam tabulek