# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Ňorba** Jméno: **Arťom** Osobní číslo: **499189**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

## II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

**Digitalizace výuky funkcí v matematice pro SŠ**

Název bakalářské práce anglicky:

**The digitization of teaching of functions in math for secondary schools**

Pokyny pro vypracování:

Cílem práce je přispět k digitalizaci výuky matematiky na střední škole. Východiskem práce je analýza stávajících aplikací vhodných pro digitální vzdělávání na střední škole. Systém navržený na základě analýzy by měl být vhodnou digitální pomůckou pro podporu výuky funkcí v matematice.
Požadavky projektu:
1. Seznamte se s požadavky, které jsou kladeny na výuku matematiky na středních školách, a to jak ze strany vládních orgánů, tak ze strany didaktiky matematiky.
2. Vyhledejte a analyzujte aplikace vhodné pro podporu digitální výuky na střední škole. Specifikujte jejich výhody a nevýhody při využití ve výuce matematiky, se zaměřením na téma funkcí.
3. Na základě analýzy navrhněte aplikaci pro podporu výuky funkcí a připravte materiály pro reálnou praktickou výuku. Aplikaci implementujte a otestujte.
4. Zúčastněte se praktické výuky s využitím aplikace a sledujte přínos aplikace pro výuku se zaměřením na přínos pro studenty SŠ.

Seznam doporučené literatury:

Fryč, Jindřich & kol. 2020. „Strategie vzdělávací politiky ČR do roku 2030+.“ Praha: MŠMT. Dostupné 31. led. 2023 https://www.msmt.cz/uploads/Brozura_S2030_online_CZ.pdf
Jeřábek, Jaroslav & kol. 2007. „RVP pro gymnázia.“ Praha: VÚP v Praze. Dostupné 31. led. 2023 https://www.edu.cz/rvp-ramcove-vzdelavaci-programy/ramcove-vzdelavaci-programy-pro-gymnazia-rvp-g/
Kalchman, Mindy and Koedinger , Kenneth R. (2005) „Teaching and Learning Functions.“ How Students Learn: History, Mathematics, and Science in the Classroom. Washington, DC: The National Academies Press.
Hejný, Milan, Novotná, Jarmila, Stehlíková, Naďa. (2004) „Dvacet pět kapitol z didaktiky matematiky.“ Praha: UK v Praze. Dostupné 31. led. 2023 http://mdisk.pedf.cuni.cz/SUMA/MaterialyKeStazeni/PublikaceKnihy/25KapitolZDM.pdf

Jméno a pracoviště vedoucí(ho) bakalářské práce:

**RNDr. Ingrid Nagyová, Ph.D. kabinet výuky informatiky FEL**

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

Datum zadání bakalářské práce: **09.02.2023** Termín odevzdání bakalářské práce: **26.05.2023**

Platnost zadání bakalářské práce: **22.09.2024**

| RNDr. Ingrid Nagyová, Ph.D.<br>podpis vedoucí(ho) práce | podpis vedoucí(ho) ústavu/katedry | prof. Mgr. Petr Páta, Ph.D.<br>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

České vysoké učení technické v Praze
Fakulta elektrotechnická

# Digitalizace výuky funkcí v matematice pro SŠ

Bakalářská práce

*Arťom Ňorba*

Studijní program: Softwarové inženýrství a technologie
Vedoucí práce: RNDr. Ingrid Nagyová, Ph.D.

Praha, 2023

**Vedoucí práce:**
RNDr. Ingrid Nagyová, Ph.D.
Kabinet výuky informatiky
Fakulta elektrotechnická
České vysoké učení technické v Praze
Karlovo náměstí 13
121 35 Praha 2
Česká republika

# Deklarace

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 26.5.2023

.....
Arťom Ňorba

# Poděkování

Rád bych poděkoval vedoucí této bakalářské práce, paní doktorce Ingrid Nagyové, za pomoc při zpracování této bakalářské práce, za její rady a pravidelné konzultace, které byly pro psaní této práce velmi přínosné.

# Anotace

Bakalářská práce začíná obecným popisem matematiky a matematických funkcí. Poté se zaměřuje na výuku matematiky na středních školách a poukazuje na snahy a možnosti integrace informačních a komunikačních technologií do výuky. Poté práce analyzuje současný stav výukových aplikací, které mají za cíl pomoci studentům s probíranou látkou. Na základě této analýzy je předložen návrh nové aplikace. Na návrh aplikace navazuje výběr vhodných technologií pro implementaci. Nakonec je popsána samotná implementace a testování aplikace.

**Klíčová slova:** matematika, informační a komunikační technologie, podpora výuky

**Vedoucí práce:** RNDr. Ingrid Nagyová, Ph.D.

# Annotation

The bachelor thesis starts with a general description of mathematics and mathematical functions. It then focuses on the teaching of mathematics in secondary schools and highlights the efforts and possibilities of integrating information and communication technologies into teaching. Then, the thesis analyses the current state of educational applications that aim to help students with the material discussed. Based on this analysis, a proposal for a new application is made. The application design is followed by the selection of appropriate technologies for implementation. Finally, the implementation and testing of the application itself is described.

**Keywords:** mathematics, information and communication technologies, teaching support

**Title translation:** The digitization of teaching of functions in math for secondary schools

# Seznam obrázků

# Seznam ukázek kódu

# Seznam zkratek

| | |
|---|---|
| CSS | Cascading Style Sheets |
| DBMS | Database Management System |
| HTML | Hypertext Markup Language |
| ICT | Informační a komunikační technologie |
| JS | JavaScript |
| JVM | Java Virtual Machine |
| MVC | Model-View-Controller |
| MŠMT | Ministerstvo školství, mládeže a tělovýchovy |
| RVP | Rámcový vzdělávací program |
| SPA | Single page application |
| UI | User interface |
| ČR | Česká republika |

# Obsah

# Kapitola 1

# Úvod

V současné době zájem o matematiku pomalu, ale jistě klesá. Potvrzuje to i fakt, že maturanti mají o maturitu z matematiky menší zájem, nebo skutečnost, že na problémy ve výuce matematiky v roce 2020 reagoval ministr školství Robert Plaga, když označil systém výuky matematiky za neodpovídající požadavkům na maturanty z matematiky.[1] Této situaci by mohly pomoci moderní informační a telekomunikační technologie, které se rychle rozvíjejí. Bohužel způsob výuky se nevyvíjí spolu s technologiemi, a to může být hlavní problém pro modernizaci způsobu výuky matematiky a vzdělávání jako celku.

Cílem této práce je získat přehled o výuce matematiky, zejména o výuce matematických funkcí na středních školách v České republice. Dále je třeba analyzovat v současnosti dostupné online aplikace vytvořené na podporu výuky matematiky a s pomocí této analýzy navrhnout novou aplikaci, která by sloužila žákům středních škol k lepšímu pochopení matematiky a matematických funkcí a také učitelům k tvorbě digitálních materiálů.

# Kapitola 2

# Úvod do matematického vzdělání

V této kapitole se definuje matematika jako samostatná disciplína. Dále se matematika rozděluje na menší části a blíže se zaměřuje na matematické funkce. Dále tato kapitola popisuje, jak MŠMT vnímá výuku matematiky na středních školách a jaký je zájem studentů o studium matematiky.

## 2.1 Úvod do matematiky

První dochované písemné záznamy o matematice pocházejí ze starověké Mezopotámie z doby kolem roku 2000 př. n. l. V těchto písemných záznamech se nacházejí základy algebry a geometrie. [2] Matematika se dále rozvíjela po celém světě a k jejímu rozvoji přispělo mnoho kultur. Například indičtí matematici zavedli pojmy jako šestnáctkové číslo a zápis nuly, zatímco čínská matematika zahrnovala pokročilé pojmy jako sčítání, násobení a dělení. [3]

Matematika je v současnosti vysoce abstraktní a přesná věda, která nepřipouští chyby ve výpočtech. Jejím charakteristickým znakem je důraz na absolutní přesnost metod a nezpochybnitelnost výsledků. Je základním nástrojem vědy a techniky a její principy se využívají v mnoha oborech, včetně fyziky, astronomie, ekonomie a informatiky. Formálně se matematika zabývá množstvím, strukturou, prostorem a změnou. Často je také popisována jako disciplína, která vytváří abstraktní entity a hledá mezi nimi zákonitosti.[4]

## 2.2 Rozdělení matematiky

Hlavní klasické matematické disciplíny se vyvinuly ze čtyř praktických lidských potřeb - potřeby počítat v obchodě, pochopit vztahy mezi číselnými veličinami, měřit pozemky a budovy a předpovídat astronomické jevy. Tyto potřeby vedly ke vzniku čtyř klasických matematických disciplín: aritmetiky, algebry, geometrie a matematické analýzy [5], které se zabývají čtyřmi základními oblastmi zájmu matematiky - množstvím, strukturou, prostorem a změnou.

## 2.3 Matematické funkce

Matematické funkce jsou součástí aritmetiky, která se zabývá čísly a operacemi s nimi. Funkce je předpis, který každému $x$ z definičního oboru přiřazuje právě jednu funkční hodnotu $y$. [6] Funkce je obvykle zapisována ve tvaru $y = f(x)$ , nebo může být vyjádřena explicitně $f : y = x$, kde proměnná $x$ je argument funkce.

V matematice se funkce často používají k popisu závislostí mezi veličinami a jsou velmi užitečné při modelování různých jevů v přírodě i společnosti. Funkce lze také použít ke zjednodušení složitých výpočtů a k lepšímu pochopení vztahů mezi jednotlivými veličinami. Je důležité zapamatovat si základní typy funkcí a jejich vlastnosti, což je základem při řešení matematických problémů a při pochopení složitějších matematických konceptů.[7]

## 2.4 Matematika jako školní předmět

Matematika je předmět, který se vyučuje již od první třídy, kde se děti seznámí se základními principy čísel a operacemi mezi nimi, a postupně na tomto základu budují další poznatky matematiky. Výuka matematiky na gymnáziu rozvíjí a prohlubuje pochopení kvantitativních a prostorových vztahů reálného světa, utváří kvantitativní gramotnost žáků a schopnost geometrického vhledu. Ovládnutí požadovaného matematického aparátu, elementy matematického myšlení, vytváření hypotéz a deduktivní úvahy jsou prostředkem pro nové hlubší poznání a předpokladem dalšího studia. Osvojené matematické pojmy, vztahy a procesy pěstují myšlenkovou ukázněnost, napomáhají žákům k prožitku celistvosti.[8] RVP dále definuje přesné okruhy, které se ve školách vyučují. Konkrétně u matematických funkcí se RVP zaměřuje na znalost grafů základních funkcí, vlastnosti funkcí a modelování závislosti reálných dějů pomocí známých funkcí.

## 2.5 Zájem o matematiku

Zájem o matematiku v posledních letech klesá, což je patrné zejména na účasti na státní maturitní zkoušce. V roce 2022 si ji zvolilo pouze 17% maturujících studentů, zatímco v roce 2013 to bylo 39%. [9] I přesto se ale úspěšnost studentů na této zkoušce mírně zvyšuje, což naznačuje, že si ji zapisují především ti, kteří ji lépe ovládají. V současnosti se zdá, že změna způsobu výuky matematiky by mohla přispět k navýšení zájmu o studium tohoto předmětu.

# Kapitola 3

# Zapojení ICT do výuky

Tato kapitola se zaměřuje na to, kam se Ministerstvo školství, mládeže a tělovýchovy hodlá v blízké budoucnosti ubírat v oblasti matematického vzdělávání. Popisuje také metodiky, které k tomu lze využít, a způsoby, jakými lze Informační a komunikační technologie využít ve výuce.

## 3.1 Informační a komunikační technologie

Informační a komunikační technologie (ICT z anglického Informatic and Communication Technologies) je široce používaný pojem, který zahrnuje veškeré technologie používané pro práci s informacemi a komunikaci. Nejběžnějšími zástupci moderních informačních a komunikačních technologií jsou stále osobní počítač, internet a mobilní telefon. Mezi současně nastupující zástupce patří tablety, chytré telefony a další.

## 3.2 Využití ICT ve výuce

Současné pojetí výuky, které využívá ICT, kombinuje instruktivní a konstruktivní přístupy k výuce. Používání ICT po celou dobu vyučování ale není úplně vhodné. Učitel by měl kombinovat klasickou výuku s výukou, která využívá prvky ICT. Žáci by se měli postupně učit využívat a pracovat s ICT ve výuce. Je to ale dlouhodobý proces. Nesprávné používání ICT vede k neefektivní výuce. Cílem ICT ve vzdělávání je zajištění kvalitního ICT vzdělávání učitelů, rozvoj schopností ICT zakomponovat do výuky jednotlivých předmětů a zajištění dostupnosti technologií (dodávka a správa hardwaru, připojení k internetu). Tuto metodiku lze využívat při výuce v libovolném předmětu, není vázáno pouze na jeden konkrétní předmět.[10]

## 3.3 Strategie vzdělávací politiky ČR do roku 2030+

Cílem Strategie 2030+ je modernizovat vzdělávání tak, aby děti i dospělí obstáli v dynamickém a neustále se měnícím světě 21. století.

Snahou projektu je mimo jiné uzpůsobit vzdělávací systém tak, aby se byl schopen adekvátně adaptovat na dynamické prostředí a pokrok spojený s rozvojem nových technologií, digitalizace a internacionalizace. Strategie bude usilovat o zvýšení úrovně digitálních dovedností a informatického myšlení, respektive digitálních kompetencí. Důležité je kritické a odpovědné používání digitálních technologií při výuce i mimo ni. Vzdělávání bude zahrnovat informační a datovou gramotnost, komunikaci a spolupráci, mediální gramotnost, tvorbu digitálního obsahu, bezpečnost v on-line prostředí, ale i řešení problémů a kritické myšlení.[11]

## 3.4 Konstruktivismus

Konstruktivismus zahrnuje široký proud teorií ve vědách o chování a sociálních vědách, zdůrazňující jak aktivní úlohu subjektu a význam jeho vnitřních předpokladů v pedagogických a psychologických procesech, tak důležitost jeho interakce s prostředím a společností. V didaktice je jedním z dominantních současných paradigmat. [12]

## 3.5 Konstruktivismus ve výuce

Konstruktivismus považuje učení za hluboce individuální proces a zdůrazňuje, že poznání a realita nemají objektivní nebo absolutní hodnotu (nebo způsob poznání této reality neznáme). Člověk konstruuje a interpretuje realitu na základě své vlastní individuální zkušenosti. Při aplikaci konstruktivismu ve školní praxi student na základě seznámení s několika různými teoriemi (a to v rovině přístupu vědeckého i senzitivního), v aktivní diskusi s učitelem, spolužáky a v kritickém přehodnocení svých původních názorů dospívá k vybudování vlastní a neopakovatelné struktury vědomostí a postojů. Základními metodickými východisky je činnostní pojetí výuky, zkušenostní učení v reálném kontextu a sebereflexe. Důraz je kladen na sociální rozměr vzdělávání (utváření vlastních názorů v konfrontaci s názory ostatních). Student potřebuje aktivně něco dělat, diskutovat a konfrontovat své názory s názory ostatních, tvořit a získat reálné zkušenosti.[12]

## 3.6 Gamifikace

Gamifikace je proces, kterým se přidávají prvky hry do neherních aktivit, aby se zlepšila motivace, zapojení a zábava.[13] Může to být aplikováno v mnoha různých oblastech, například v obchodě, vzdělávání, zdraví a wellness, v oblasti zákaznického servisu a mnoho dalších. Cílem gamifikace je přimět lidi, aby se více zapojili do dané aktivity, tím že jim poskytne hříčky, odměny a cíle, které jsou podobné těm, které se nacházejí ve hrách.

## 3.7 Gamifikace ve výuce

Gamifikace může být účinným nástrojem pro zlepšení motivace a zapojení studentů ve školství. Například může být použita k poskytnutí odměn studentům za splnění určitých cílů nebo úkolů, nebo k vytvoření soutěží mezi studenty, které jsou zaměřené na vzdělávání. Gamifikace také může být použita k vytvoření interaktivních her nebo aktivit, které jsou zaměřené na vzdělávání, a které mohou být zábavné pro studenty a zároveň účinné pro pochopení učiva.[14]

Gamifikace ve školství by měla být používána s rozvahou a měla by být začleněna do vzdělávacího plánu tak, aby byla začleněna do výuky a aby byla účinná pro studenty. Také je důležité vzít v úvahu, že gamifikace nemusí fungovat pro všechny studenty a může mít nežádoucí vedlejší účinky, pokud není používána správně.

# Kapitola 4

# Analýza současného stavu

V této kapitole se podíváme na existující aplikace, které jsou vyvinuty pro podporu výuky. Také se u každé aplikace zaměříme na technologii nebo zajímavost, kterou se odlišuje od ostatních. Tyto odlišnosti pak budou sloužit jako inspirace při návrhu výsledné aplikace této práce.

## 4.1 Aplikace pro podporu výuky

V první části analýzy se zaměříme na aplikace, které se využívají studenty i učiteli k výuce, či jako podpůrný materiál. Tyto aplikace se mohou využívat v hodině i doma při samostudiu.

### 4.1.1 Duolingo

Duolingo[1] je vzdělávací aplikace, která umožňuje uživatelům se učit nové jazyky prostřednictvím interaktivních lekcí a cvičení. Je dostupná na internetu nebo jako mobilní aplikace pro chytré telefony a tablety. Duolingo se používá pro výuku jazyků, jako je angličtina, španělština, francouzština, němčina, italština, portugalština, holandština a mnoho dalších. Aplikace je zdarma a je určena pro všechny věkové kategorie. Duolingo se používá k učení jazyků pro různé účely, jako je příprava na cestování, zlepšení schopností komunikace pro pracovní účely nebo pro osobní rozvoj. Aplikace je známá pro svoje zábavné uživatelské rozhraní a efektivní metody učení, jako například rozdělení učení na malé části a použití okamžité zpětné vazby. Duolingo také umožňuje uživatelům nastavit si vlastní cíle a odměňuje je za splnění těchto cílů.

[1] https://www.duolingo.com

### 4.1.2 Kahoot

Kahoot[2] je webová aplikace nebo mobilní aplikace, která umožňuje vytvářet a hrát interaktivní hry a kvízy pro vzdělávací účely. Aplikace se používá ve školách, na vysokých školách a v dalších vzdělávacích institucích po celém světě. Kahoot je určena pro všechny věkové kategorie a je k dispozici zdarma.

Kahoot se používá k učení různých předmětů, jako je matematika, přírodní vědy, historie, literatura a mnoho dalších. Je také často používána pro zpestření výuky a udržení pozornosti studentů. Může být použita jako nástroj pro procvičení látky nebo jako motivační prvek při učení. Kahoot také umožňuje učitelům vytvářet vlastní hry nebo vybírat z již existujících her, které jsou k dispozici v databázi aplikace. Hry se hrají ve skupině a studenti odpovídají na otázky pomocí mobilního zařízení nebo počítače. Aplikace umožňuje učitelům sledovat pokrok studentů a poskytovat jim zpětnou vazbu.

### 4.1.3 Brainscape Flashcards

Brainscape je společnost, která poskytuje platformu[3] pro učení, která zahrnuje funkci nazvanou "flashcards".Flashcards jsou nástroj, který lze použít k pomoci při učení nových informací. Skládají se z otázky nebo podnětu na jedné straně a odpovědi na straně druhé. Myšlenka spočívá v tom, že můžete použít flashcards k samostatnému zkoušení sebe s materiálem, který se snažíte naučit, s cílem zlepšit si paměť a porozumění danému předmětu. Funkce flashcards společnosti Brainscape umožňuje uživatelům vytvářet vlastní flashcards nebo používat ty, které již byly vytvořeny jinými lidmi. Také obsahuje nástroje pro efektivnější učení, jako například algoritmy pro opakované procvičování, které pomáhají naplánovat revizní sezení pro maximální účinnost.

### 4.1.4 Isibalo

Isibalo[4] je vzdělávací webová aplikace, která nabízí výukové materiály v oblasti matematiky, chemie, fyziky a biologie. Stránka obsahuje zdarma videa, placená videa, články a interaktivní prvky ve formě testů nebo kvízů. Isibalo také pěkně graficky poskytuje celkové statistiky stránky, jako například průměrnou délku jednoho kurzu nebo počet videí vzhledem k počtu interaktivních prvků, aby uživatel ihned věděl, z čeho se kurz skládá.

[2]https://www.kahoot.com
[3]https://www.brainscape.com
[4]https://www.isibalo.com

### 4.1.5 Khan academy

Khan Academy[5] je nezisková organizace, která poskytuje online vzdělávací materiály a řešení pro studenty všech věkových kategorií. Cílem Khan Academy je poskytnout kvalitní a dostupné vzdělání pro všechny lidi na celém světě. K dispozici je široká škála předmětů, včetně matematiky, přírodních věd, programování, ekonomie a mnoha dalších. Khan Academy obsahuje interaktivní lekce a úkoly, které jsou navrženy tak, aby pomohly studentům lépe pochopit dané téma. Učitelé také mohou využít Khan Academy jako nástroj pro sledování pokroku svých studentů a poskytovat jim individuální pomoc při učení. Khan Academy je zdarma pro všechny a je dostupná online přes webové stránky nebo prostřednictvím aplikace pro chytré telefony a tablety.

### 4.1.6 Hot potatoes

Hot Potatoes[6] je software, který umožňuje vytvářet interaktivní cvičení a testy pro studenty. Je určen pro využití ve vzdělávacích institucích a na vysokých školách, ale může být také použit pro soukromé studium nebo sebevzdělávání. Hot Potatoes je k dispozici pro operační systémy Windows a Mac a je k dispozici zdarma ke stažení. Software umožňuje vytvořit různé typy cvičení, jako například křížovky, slova s významem, klasifikační testy a další. Cvičení mohou být publikována na webu nebo rozdána studentům prostřednictvím souboru, který lze otevřít v počítači nebo na mobilním zařízení. Hot Potatoes také umožňuje učitelům sledovat pokrok studentů a poskytovat jim zpětnou vazbu.

[5] https://www.khanacademy.org
[6] https://www.hotpot.uvic.ca

## 4.2 Aplikace pro podporu výuky matematiky

V druhé části analýzy si přiblížíme aplikace, kterou jsou určené přímo pro výuku matematiky. Tyto aplikace často slouží jako kalkulačky nebo grafické vyobrazení probírané látky.

### 4.2.1 PhotoMath

Photomath[7] je mobilní aplikace pro chytré telefony a tablety, která umožňuje uživatelům získat nápovědu při řešení matematických úloh. Aplikace funguje tak, že uživatel nafotí matematický problém pomocí fotoaparátu svého zařízení a aplikace pak automaticky vygeneruje řešení úlohy. Photomath je k dispozici zdarma a je určena pro všechny věkové kategorie.

Photomath se používá k pomoci studentům při řešení matematických úloh, které mohou být obtížné nebo které studenti nechápou. Aplikace je užitečná pro studenty na základních školách, středních školách i na vysokých školách. Kromě toho, že poskytuje řešení úlohy, také obsahuje vysvětlení postupu, kterým bylo řešení dosaženo, což pomáhá studentům lépe pochopit problém a případně se naučit řešit podobné úlohy samostatně. Photomath je také užitečná pro dospělé, kteří potřebují rychle získat nápovědu při řešení matematických úloh pro pracovní nebo osobní účely.

### 4.2.2 Geogebra

Geogebra[8] je bezplatný software pro výuku matematiky a geometrie. Je určen pro všechny věkové kategorie. Geogebra umožňuje uživatelům vytvářet a pracovat s matematickými objekty, jako jsou body, čáry, křivky, mnohoúhelníky a kružnice, a pomáhá jim vysvětlit matematické principy pomocí animací a interaktivních prvků.

Geogebra se používá pro výuku matematiky a geometrie na všech úrovních vzdělávání, od základních škol až po vysoké školy. Je také často používána pro sebevzdělávání a pro výzkumné účely. Software obsahuje širokou škálu nástrojů a funkcí, které umožňují učitelům vytvářet a upravovat vlastní lekce a cvičení, a studentům se učit interaktivním způsobem. Geogebra také obsahuje nástroje pro výpočet složitějších matematických výrazů a pro tvorbu grafů funkcí. Umožňuje uživatelům pracovat s 3D objekty a zobrazovat matematické objekty v prostoru. Software je flexibilní a umožňuje uživatelům pracovat s různými typy matematických objektů a přizpůsobovat jejich vlastnosti podle potřeby. Geogebra je také vybavena funkcí pro vytváření a spolupráci na pracovních listech s ostatními uživateli.

[7] https://www.photomath.com
[8] https://www.geogebra.org

### 4.2.3 Portál středoškolské matematiky

Matematicko-fyzikální fakulta Univerzity Karlovy vytvořila webové stránky[9] pro výuku matematiky, kde je k dispozici sekce věnovaná pouze funkcím. Stránka obsahuje stručné vysvětlení různých druhů funkcí a také grafické zobrazení, se kterým může student manipulovat a snadno zjistit, jaké jsou účinky jednotlivých parametrů na podobu grafu funkce.

### 4.2.4 Mathematicator

Mathematicator[10] je webová stránka zaměřená na výuku matematiky. Byla iniciálně spuštěna učitelem Markem Valáškem, který se stal známým zejména díky svým videím na YouTube, ve kterých vysvětluje matematiku a zároveň propaguje svoji stránku. Kromě výukových kurzů pro základní, střední a vysoké školy nabízí Mathematicator také doučování a matematické fórum, které je zaměřeno na soutěže a motivuje uživatele k přispívání svými otázkami a odpověďmi.

[9] https://www2.karlin.mff.cuni.cz/ portal/funkce/?page=title
[10] https://www.mathematicator.com

## 4.3 Přínos pro navrhované řešení

V této sekci shrneme, čím mohou jednotlivé aplikace přispět bakalářské práci. Neboli čím se jednotlivé aplikace odlišují od ostatních nebo díky čemu jsou tyto aplikace populární. V návrhu aplikace mohou tyto části sloužit například jako samostatné moduly nebo jako pomůcky, či jiné části softwaru.

### Duolingo

Zobrazení aktuálního progresu a vizuální oddělení témat, která jsou již probraná, které student ještě nezačal a která jsou zablokovaná, jelikož navazují na látku, která není ještě student splněna, což dá studentovi okamžitě jasně znát jak si stojí ve svém pokroku a motivuje ho pokračovat dál. Student má pocit, že hraje hru a každé téma je nová úroveň. Také možnost sbírání bodů za každou splněnou lekci a každodenní návštěvu aplikace motivuje studenty absolvovat kurzy. Nasbírané body se pak zobrazí v žebříčkách, kde student soupeří s ostatními studenty z celého světa.

### Kahoot

Interaktivní soutěžení se spolužáky, kamarády či náhodnými lidmi, kdy se při odpovídání na otázku vybírá ze 2-4 možností a délka soutěže je v rámci minut. Vytváření si vlastních kvízů nebo použití již vytvořených od jiných uživatelů slouží jako jednoduchý a efektivní nástroj pro zábavu a výuku zároveň. Pro hravost slouží i použité zvukové efekty a znělky, které dokáží uživatele vtáhnout do soutěže.

### Brainscape Flashcards

Spaced repetition, což je metodika, která nenabízí slovíčka (učivo), které student již umí, ale nabízí pouze látku, která studentovi momentálně dělá problém. Kartičky nejprve zobrazí otázku, slovíčko v cizím jazyce nebo obrázek a student má čas si rozmyslet odpověď. Pak se kartička otočí a zobrazí správné řešení. Student si pak může označit kartičku jako hotovou (pokud věděl odpověď) nebo ji znovu zařadí do balíčku kartiček, aby se mu kartička v budoucnu znovu zobrazila.

### Isibalo

Výuková videa doprovázená jednoduchým, ale zároveň hravým uživatelským rozhraním a seznamy, které jsou pěkně přehledné.

### Khan academy

Hravé tutoriály, které nejprve zobrazí probíranou látku v učebnicové podobě a následně přichází kvíz nebo test pro potvrzení znalosti látky.

### Hot potatoes

Možnost vytváření vlastních testů či materiálů pomocí úloh, které se skládají z interaktivních aktivit jako jsou drag&drop, přiřazování atd.

### Photomath

Interaktivní kalkulačka, které dokáže vypočítat libovolný druh příklady a dodává i popis vloženého zadání a podrobně vyřešený příklad krok po kroku. Pokud je to u daného příkladu možné, tak přiloží i graf s popisky nebo dodá více možných postupů, které vedou ke správnému řešení.

### Geogebra

Možnost zobrazení více grafů v jednu chvíli a jejich barevné oddělení pro lepší přehlednost, ale také komplexní grafický nástroj pro jakoukoliv úlohu z geometrie.

### Portál středoškolské matematiky

Interaktivní grafy funkcí, které se dají měnit pomocí tahu myši (či prstem na mobilních zařízeních) doprovázené teoretickou částí pro lepší pochopení vykreslování grafů funkcí.

### Mathematicator

Fórum, kde si uživatelé mohou navzájem odpovídat na úlohy, které uživatel nemohl sám vypočítat nebo mu si není jistý správností svého postupu.

# Kapitola 5

# Návrh aplikace

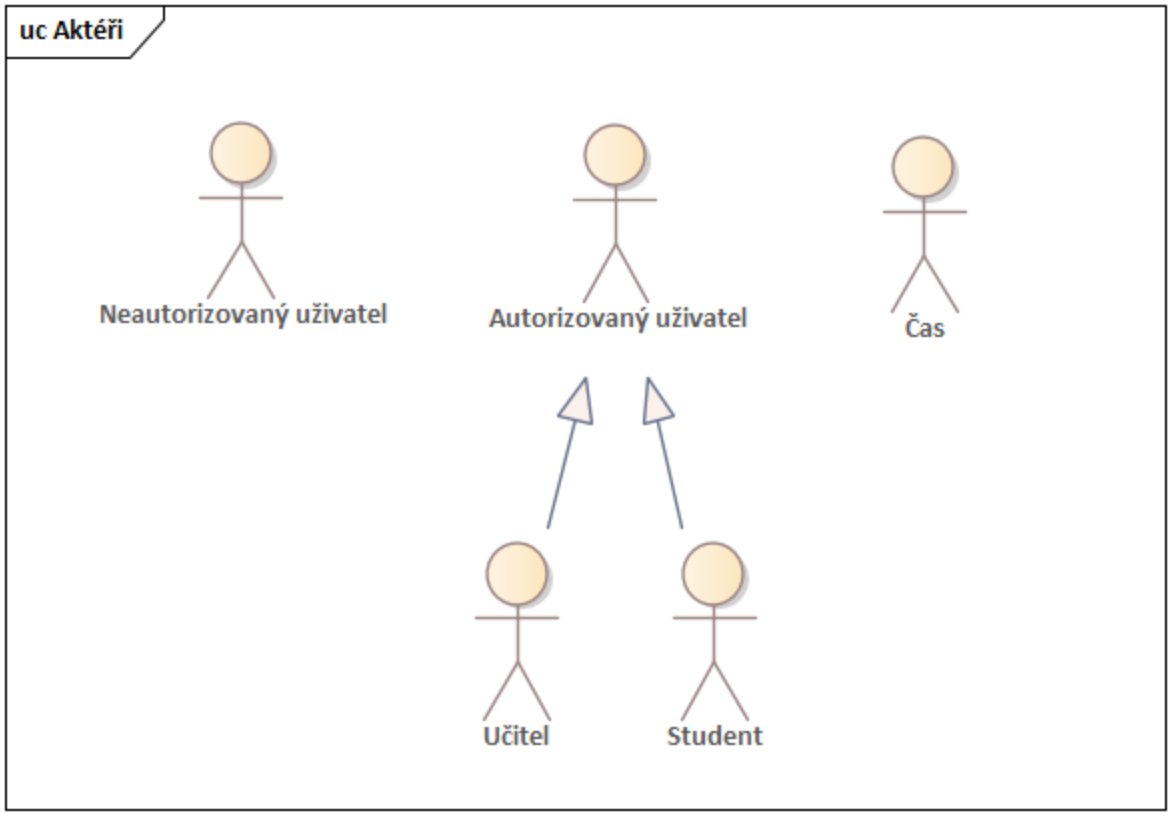
Tato kapitola obsahuje návrh aplikace a všechna potřebná schémata k tomuto účelu. Nejprve jsou zobrazeni aktéři systému, poté jsou popsány požadavky na systém, které jsou rozděleny na funkční a nefunkční. Poté jsou vytvořeny diagramy tříd a diagramy použití. Nakonec je na konci oddílu uveden diagram nasazení a grafický návrh aplikace.

## 5.1 Úvod

Aplikace ze sekce analýzy současného stavu [4] ukazují, že výsledná aplikace této práce by měla mít hravé a moderní uživatelské rozhraní. Kromě toho je třeba, aby učitelé mohli vytvářet výukové materiály, lekce a testy, které budou kromě základních prvků výuky obsahovat i interaktivní prvky, v nichž se žák aktivně zapojí do výuky. Výuku matematiky lze podpořit i soutěžními prvky, kdy se žáci snaží co nejrychleji vybrat správnou odpověď na otázku zadanou učitelem nebo být za svůj pokrok odměněni body do celkové soutěže mezi všemi žáky. Zejména při výuce matematických funkcí by měly být pro lepší demonstraci látky k dispozici interaktivní grafy funkcí, které se mění podle studentem zadaných parametrů. Aplikace musí být přehledná, aby uživatel věděl, kde se v daném okamžiku nachází a jaké kroky musí učinit, aby dosáhl svého cíle.

## 5.2 Aktéři

Systém se skládá z několika účastníků, mezi něž patří neautorizovaný uživatel a autorizovaný uživatel. Oprávněný uživatel je dále rozdělen do dvou kategorií: učitel a student. Oba tito aktéři mají v systému různá práva a přístupy. Neautorizovaný uživatel nemá přístup ke všem funkcím systému, zatímco autorizovaný uživatel, ať už se jedná o učitele, nebo studenta, má v systému přidělena konkrétní práva, která mu umožňují provádět určité akce a manipulovat s daty. Posledním aktérem je čas, který také může zasahovat do chodu aplikace.



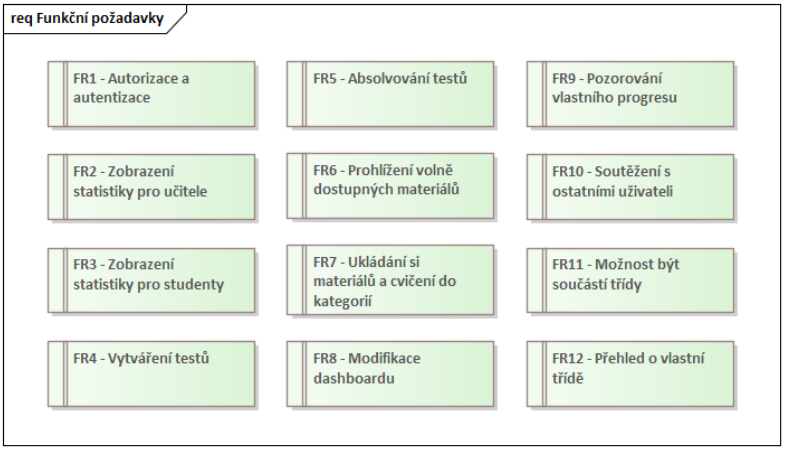

Obrázek 5.1: Aktéři systému

## 5.3 Systémové požadavky

Systémové požadavky jsou specifikací požadovaného chování vyvíjeného systému. Tyto požadavky slouží jako podrobný popis funkcí, vlastností a omezení, které by měl systém splňovat. Jejich účelem je definovat očekávané výstupy, vstupy, interakce s uživateli a další aspekty systému. Požadavky na systém tak představují základ pro návrh, vývoj a testování systému a umožňují zajistit, aby výsledný systém splňoval požadavky a potřeby uživatelů.

### 5.3.1 Funkční požadavky

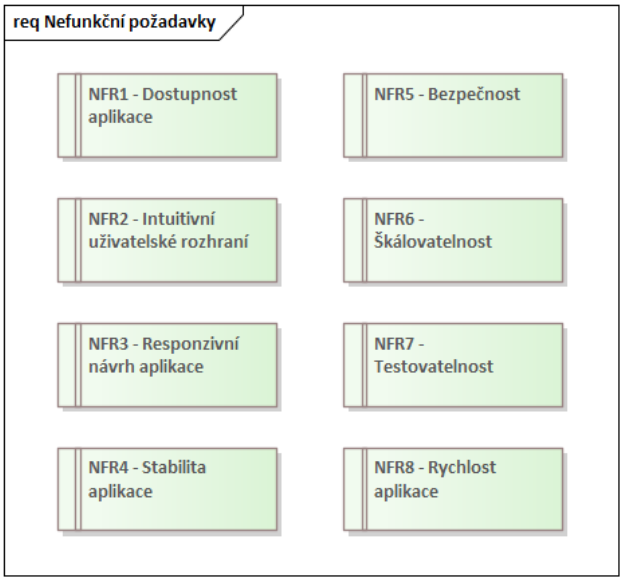
Funkční požadavky se zabývají specifikací konkrétních funkcí, které musí systém poskytovat. Tyto požadavky specifikují, jak má systém reagovat na určité vstupy a provádět určité akce. Funkční požadavky se zaměřují na to, co systém dělá, jaké operace umožňuje uživatelům provádět a jaké výstupy generuje.




Obrázek 5.2: Funkční požadavky

### 5.3.2 Nefunkční požadavky

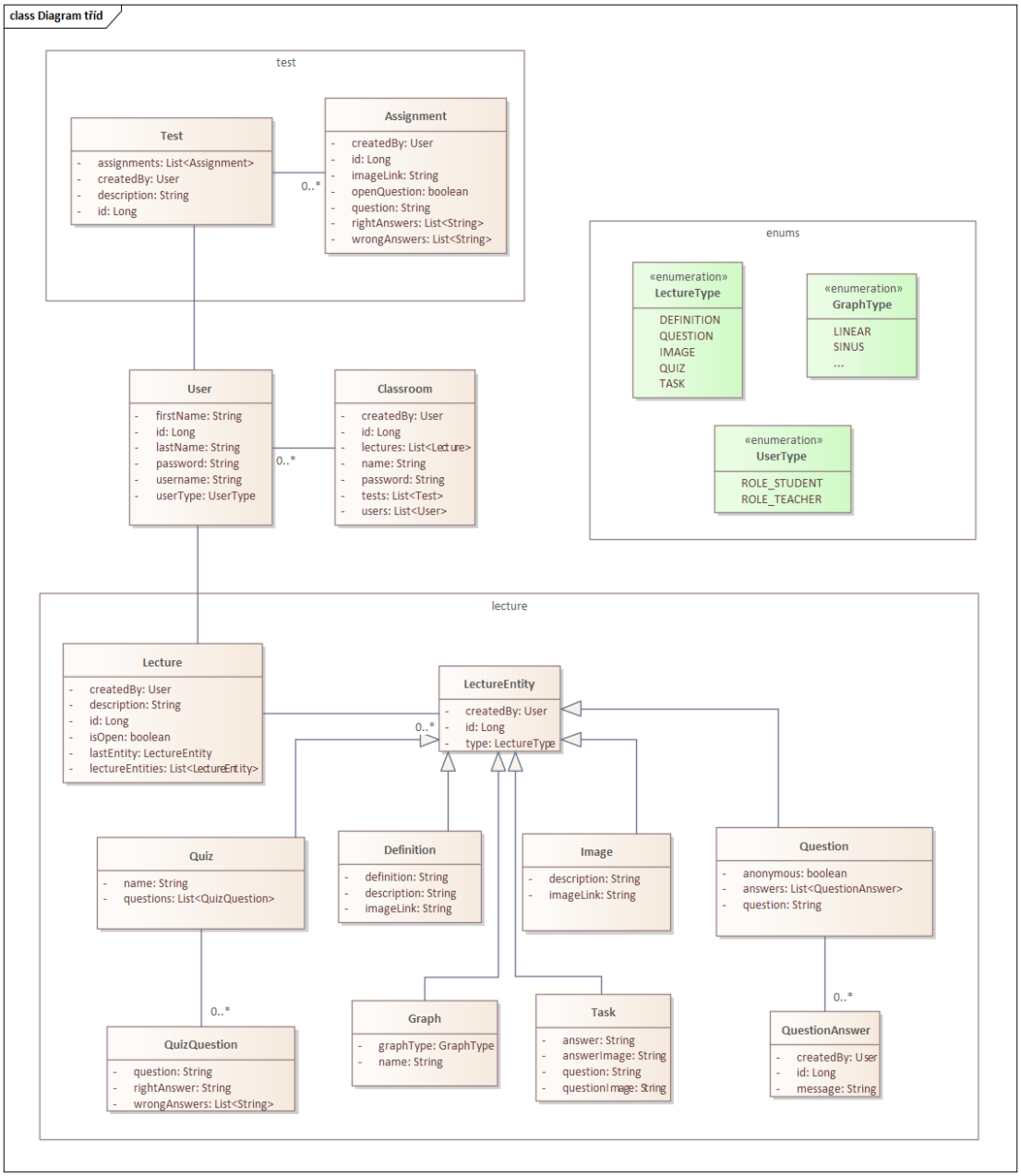
Nefunkční požadavky se zaměřují na vlastnosti a omezení systému, které přímo nesouvisejí s jeho funkčností, ale ovlivňují jeho celkový výkon, spolehlivost, uživatelskou přívětivost a další aspekty.




Obrázek 5.3: Nefunkční požadavky

## 5.4 Diagram tříd

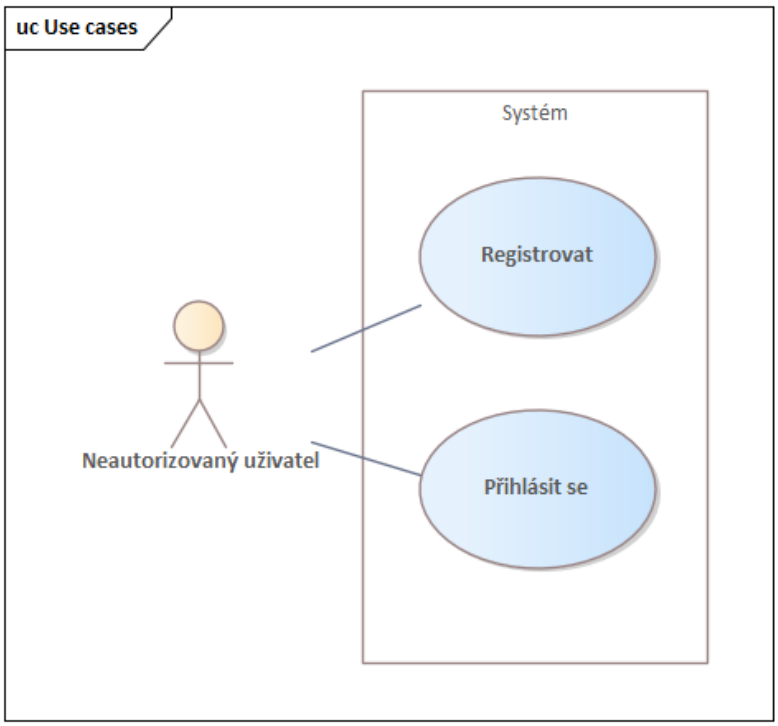
Diagram tříd graficky znázorňuje strukturu dat v aplikaci. Diagram popisuje použité třídy v aplikaci a jejich atributy. Diagram je nezávislý na výsledné implementaci a vytváří se ještě před začátkem vývoje. Dále se na základě tohoto diagramu vytváří tabulky v databázi, aby odpovídaly použitým třídám a atributům.



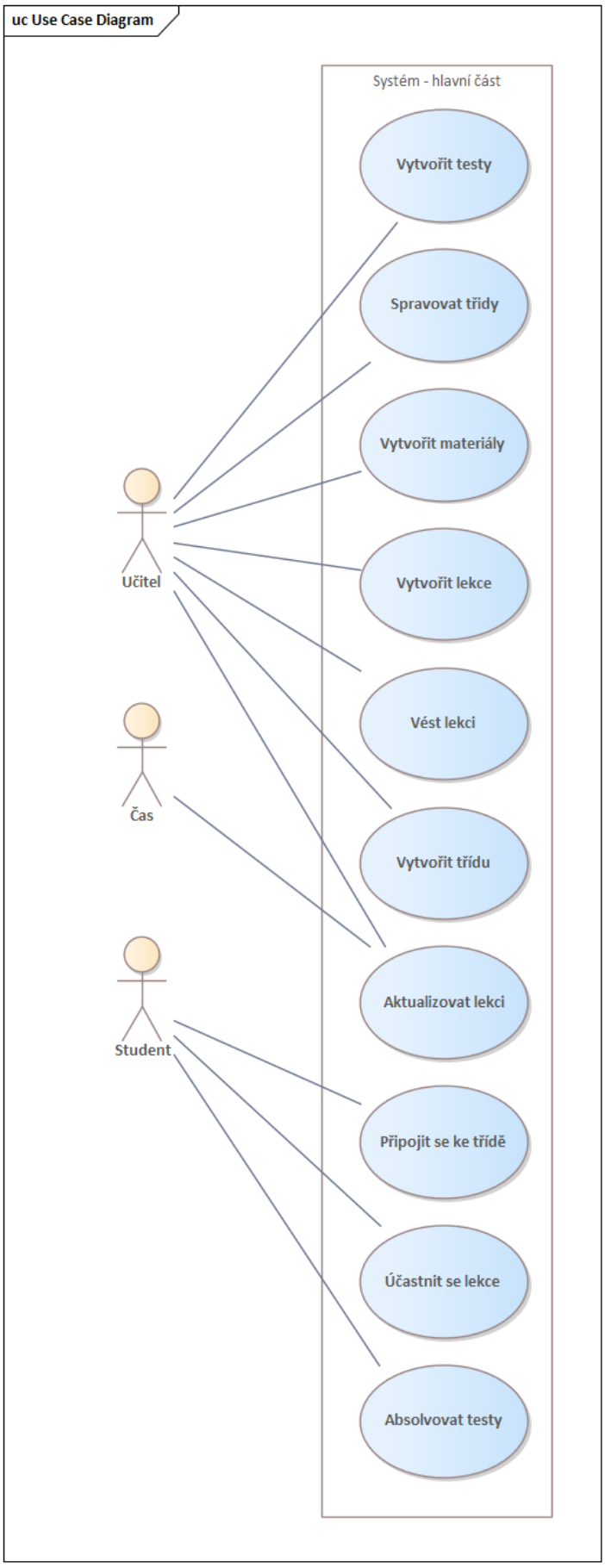

Obrázek 5.4: Diagram tříd

## 5.5 Diagramy užití

Diagramy užití mají za úkol namapovat na každého aktéra z 5.2 aktivitu spojenou se správou nebo využitím systému. Na obrázku 5.5 jsou znázorněny základní aktivity systému, které se týkají autorizace nebo registrace uživatele. Na dalším obrázku 5.6 jsou znázorněny nejdůležitější aktivity, které definují chování výsledné aplikace. Aktivity se týkají autorizovaného uživatele a jsou rozděleny na aktivity studenta, aktivity učitele nebo aktivity, které může provádět čas.



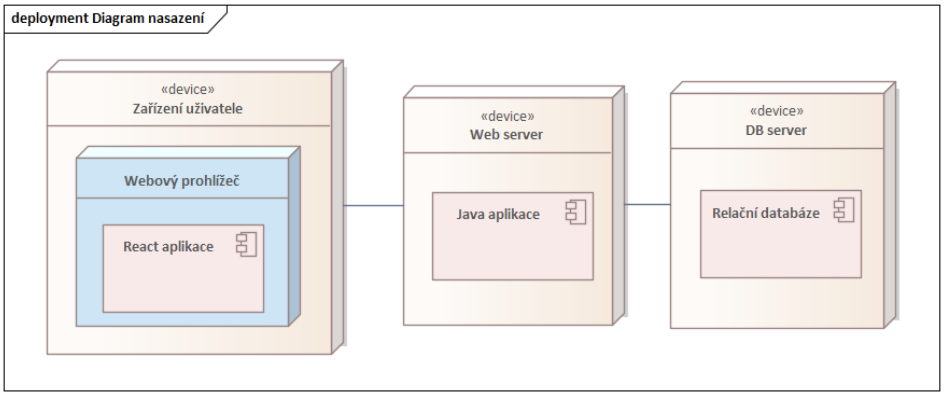

Obrázek 5.5: Základní aktivity se systémem




Obrázek 5.6: Hlavní část systému

## 5.6 Diagram nasazení

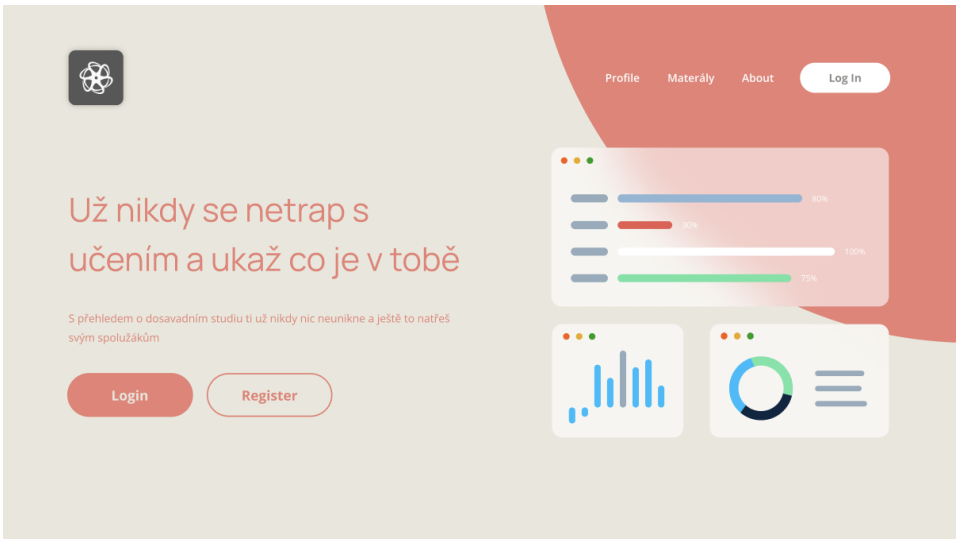
Na obrázku 5.7 je znázorněná struktura nasazení aplikace do produkce.




Obrázek 5.7: Diagram nasazení

## 5.7 Grafický návrh

Obrázek 5.8 zobrazuje grafický návrh, který slouží jako vzor a inspirace pro implementaci. Návrh byl vytvořen v programu Figma, což je velmi oblíbená aplikace pro návrh aplikací. Návrh nemusí odpovídat konečné podobě aplikace a neobsahuje přesná data, která budou vykreslena. Cílem předběžného návrhu je zvýraznit rozložení komponent a barevnou paletu aplikace.




Obrázek 5.8: Vytváření testu učitelem - návrh obrazovky

# Kapitola 6

# Analýza technologií

Tato kapitola obsahuje přehled dostupných technologií, které lze k realizaci řešení použít. Kapitola nejprve zkoumá technologie pro vývoj frontendu a backendu. Nakonec je vybrána technologie pro databázi.

**Model-View-Controller**

MVC je softwarový architektonický vzor pro implementaci uživatelských rozhraní. Rozděluje danou aplikaci na tři vzájemně propojené části, aby se oddělilo vnitřní zobrazení informací od způsobů, jakými jsou informace prezentovány a schvalovány uživatelem. Tento návrhový vzor odděluje tyto hlavní součásti a umožňuje efektivní opakované použití kódu a paralelní vývoj. [15]

## 6.1 Frontend

Frontend je část webu, se kterou uživatel přímo komunikuje. Říká se jí také "klientská strana aplikace". Je to písmeno "V"v MVC. Zahrnuje vše, s čím se uživatelé přímo setkávají: barvy a styly textu, obrázky, grafy a tabulky, tlačítka, barvy a navigační nabídky. Hlavními cíli front-endu jsou odezva a výkon. Vývojář musí zajistit, aby web byl responzivní, tj. aby se správně zobrazoval na zařízeních všech velikostí a aby se žádná část webu při jakékoli velikosti obrazovky nechovala nestandardně. [16] Základem frontendu jsou Hypertext Markup Language (HTML), Cascading Style Sheets (CSS) a JavaScript (JS).

### 6.1.1 React

React je deklarativní, efektivní a flexibilní knihovna jazyka JavaScript pro vytváření uživatelských rozhraní. ReactJS je open-source knihovna pro front-end založená na komponentách, která se stará pouze o vrstvu zobrazení aplikace. Spravuje ji společnost Facebook. React používá deklarativní paradigma, které usnadňuje tvorbu aplikací a zaměřuje se na efektivitu a flexibilitu. Navrhuje jednoduché zobrazení pro každý stav vaší aplikace a React při změně dat efektivně aktualizuje a vykresluje pouze správnou komponentu. Díky deklarativním pohledům je kód předvídatelnější a snadněji se ladí. [17]

#### Přínos pro aplikaci

Vývoj aplikací pomocí knihovny React je mezi vývojáři nejoblíbenějším řešením, což přináší obrovskou výhodu, protože v současné době existuje velká podpora ze strany komunity. Další velkou výhodou je snadnost a rychlost vývoje webových aplikací. [18]

### 6.1.2 Angular

Angular je webový framework pro vývoj frontendu založený na jazyce TypeScript společnosti Google. Angular je skvělá, použitelná knihovna uživatelského rozhraní (UI) pro vývojáře, která pomáhá vytvářet atraktivní, stabilní a užitečné webové stránky a webové aplikace. Angular je revoluční framework JavaScriptu, který umožňuje vytvářet atraktivní jednostránkové aplikace (SPA). Jeho první verze byla vydána v roce 2012 a nazývá se AngularJS. [19]

#### Přínos pro aplikaci

Angular zavádí *Dependency Injection*, velmi populární návrhový vzor, který rozděluje danou úlohu mezi více nezávislých komponent. Vývojář se nemusí o nic starat, protože o tento patern se stará přímo Angular. [20]

### 6.1.3 Vue.js

VueJS je jedním z nejpopulárnějších frameworků pro JavaScript. VueJS se používá pro navržení uživatelského rozhraní a je snadno pochopitelný pro všechny vývojáře. Je také kompatibilní s ostatními knihovnami a rozšířeními. Pokud vývojář chce vytvořit jednostránkovou aplikaci, pak je VueJS skvělou volbou. V oblasti vývoje může být mnoho problémů, které nelze vyřešit použitím jedné knihovny, proto je VueJS kompatibilní s ostatními knihovnami. VueJS podporují všechny populární prohlížeče, jako jsou Chrome, Safari atd. [21]

**Přínos pro aplikaci**

Hlavní výhodou vývoje ve Vue je rychlá křivka učení. Díky tomu může začínající vývojář bez jakýchkoli zkušeností rychle začít vyvíjet. [22]

### 6.1.4 Výběr technologie

Vzhledem k podobnosti zmíněných technologií a rozsahu práce byla zvolena knihovna React, protože React má velkou komunitní podporu, což velmi usnadňuje rychlý vývoj aplikace, a v současné době existují nástroje, které vygenerují plně funkční aplikaci během několika minut. Zároveň React nabízí snadné přesměrování, které přiřadí příslušnou komponentu ke konkrétní adrese URL, která je pak viditelná pro uživatele. Ostatní řešení jsou náročnější na zahájení vývoje, což je v kontextu této práce velkou nevýhodou.

## 6.2 Backend

Jako backend se označuje část webové aplikace, která slouží k administraci webu a ke zpracování dat. V případě redakčního systému tedy backend umožňuje vkládání a úpravy článků, zakládání účtů dalších administrátorů či třeba manipulace se strukturou celého webu.[23] V případě e-shopového řešení slouží Backend ke kompletní správě webu. Primárně se přes administraci přidávají nové produkty a upravují jejich vlastnosti, nahrávají fotky ke zboží, vytváří a editují se články, nastavuje vzhled celého e-shopu, spravují veškeré objednávky i zákazníci, vystavují se doklady (faktury, výdajové listy apod.), evidují se skladové zásoby produktů apod.[24]

### 6.2.1 Java

Java je programovací jazyk a výpočetní platforma, která byla poprvé vydána společností Sun Microsystems v roce 1995. Používá se k vývoji a spouštění široké škály aplikací, od mobilních zařízení po webové aplikace a hry. Java je navržena tak, aby byla snadná na pochopení a používání, s konzistentním sadou funkcí a jasnou syntaxí. Je to objektově orientovaný jazyk, což znamená, že je založen na konceptu "objektů", které mohou představovat data a funkčnost. Java je také nezávislá na platformě, což znamená, že může běžet na jakémkoli zařízení, které podporuje Java, bez potřeby platformových úprav.[25]

#### Přínos pro aplikaci

Vývoj aplikace v jazyce Java přináší mnoho výhod pro rychlý a snadný vývoj. Kromě nezávislosti na platformě, která usnadňuje výslednou integraci, je hlavní výhodou vestavěné zabezpečení, které lze rozšířit, což usnadňuje rozdělení uživatelů podle rolí a zabezpečení dat. [26]

### 6.2.2 Kotlin

Kotlin je staticky typovaný, objektově orientovaný programovací jazyk, který je kompatibilní s Java Virtual Machine (JVM), Java Class Libraries a Androidem. Programovací jazyk Kotlin byl původně navržen tak, aby vylepšil programovací jazyk Java a často se používá společně s Javou. Navzdory tomu, že je Kotlin výchozím jazykem pro vývoj aplikací pro Android, kompatibilita Kotlinu s Javou ho přivedla k používání s mnoha typy aplikací. [27]

#### Přínos pro aplikaci

Kotlin je čitelnější než Java a vyžaduje méně kódu. Kotlin také řeší obrovský problém Javy, a to *NullPointerException*, navíc vývojář nemusí přímo určovat typ proměnných. [28]

### 6.2.3 PHP

PHP je jedním z nejvíce rozšířených programovacích jazyků používaných k vytváření webových aplikací. PHP se používá na straně serveru a slouží tedy ke generování HTML kódu stránky, jenž pak server odesílá do prohlížeče (na rozdíl od klientského JavaScriptu, který funguje až při zobrazení stránky v prohlížeči).

Hlavním kladem PHP je jeho nezávislost na platformě (Windows, Linux, Unix...), mezi výhody PHP patří i široké možnosti použití. PHP například umí pracovat se soubory a s mnoha různými databázemi, s PHP lze generovat a upravovat grafiku, umí odesílat a přijímat emaily, vytvářet PDF, podporuje všechny důležité internetové protokoly. [29]

#### Přínos pro aplikaci

Kromě velké komunity vývojářů a snadného vývoje přináší PHP také snadnou komunikaci s většinou známých databází, což umožňuje široký výběr pro vývoj. [30]

### 6.2.4 Výběr technologie

Pro vývoj této práce byl zvolen jazyk Java. Jazyk Java je velmi oblíbeným jazykem pro vývoj backendové části webových aplikací a existuje v něm mnoho knihoven a frameworků, které tuto práci ještě více usnadňují. Nejoblíbenějším frameworkem je Spring Boot, který umožňuje vytvořit funkční aplikaci v několika krocích a po přidání několika závislostí a konfigurací je aplikace zcela připravena k použití v libovolném prostředí. Vývoj v jazyce Java zároveň umožňuje rozdělit aplikaci pro lokální vývoj a být nezávislý na produkčním prostředí.

## 6.3 Databáze

Databáze je sbírka dat, která je uspořádaná, což se též nazývá strukturovaná data. Může být přístupná nebo uložená v počítačovém systému. Může být spravována prostřednictvím systému pro správu databází (DBMS), což je software používaný pro správu dat. Databáze se vztahuje na související data v strukturované formě.

V databázi jsou data uspořádána do tabulek, které se skládají z řádků a sloupců a jsou indexována, takže data mohou být snadno aktualizována, rozšířena a odstraněna. Počítačové databáze obvykle obsahují záznamy souborů s daty, jako jsou transakce peněz z jednoho bankovního účtu na druhý, prodeje a podrobnosti o zákaznících, podrobnosti o poplatcích studentů a podrobnosti o produktech. [31]

### 6.3.1 Relační databáze

Relační databáze využívají jako základu tabulek, které jsou propojeny předem nastavenými vztahy. Tabulka obsahuje jednotlivé sloupce – atributy a řádky – záznamy. Atributy tabulky určují vlastnosti objektů, které se do tabulky budou vkládat – do tabulky se vkládají objekty stejného druhu, nicméně každý vždy jen jednou. Atributy při návrhu tabulky však rozhodně neobsahují samotné hodnoty, pouze určují jaké vlastnosti budou vkládané objekty mít v databázi uložené. [32]

#### Přínos pro aplikaci

Relační databáze mají velmi jednoduchý model a snadný přístup k uloženým datům, což ve spojení s backendem napsaným v jazyce Java přináší rychlý a jednoduchý vývoj.

### 6.3.2 Nerelační databáze

Nerelační databáze je databáze, která nepoužívá tabulkové schéma řádků a sloupců nacházející se ve většině tradičních databázových systémů. Nerelační databáze místo toho používají model úložiště, který je optimalizovaný pro konkrétní požadavky typu ukládaných dat. Data mohou být například uložena jako jednoduché páry klíč/hodnota, jako dokumenty JSON nebo jako graf skládající se z hran a vrcholů. [33]

#### Přínos pro aplikaci

Nerelační databáze poskytují flexibilní, ale komplexní model a umožňují flexibilní a dynamické změny objektů, které nemají vliv na ostatní záznamy.

### 6.3.3 Výběr technologie

Pro vývoj této práce byla zvolena relační databáze, protože má jednoduchý model. Java zároveň obsahuje nástroje, které práci s databází ještě více usnadňují a řeší všechny problémy, které mohou při integraci databáze nastat. Konkrétně byla zvolena databáze PostgreSQL.

# Kapitola 7

# Implementace aplikace

Tato kapitola popisuje hlavní postup implementace aplikace. Začíná popisem backendové části, následuje frontendová část a končí popisem databáze a nasazením aplikace na server.

## 7.1 Backend

V této části je popsáno vytvoření projektu pro backendovou část a dále jsou popsány základní principy pro vytvoření funkční backendové části připravené přijímat požadavky od klienta (frontendové části). Kapitola je doplněna konkrétními příklady implementace jednotlivých částí a na závěr je popsán princip autorizace uživatelů.

### 7.1.1 Vývojového prostředí

Pro vývoj backendové části aplikace bylo zvoleno vývojové prostředí od české společnosti JetBrains, konkrétně IntelliJ IDEA. Toto vývojové prostředí plně podporuje vývoj v jazyce Java a obsahuje také podpůrné pluginy a rozšíření pro framework Spring Boot, hibernate a další.

### 7.1.2 Založení projektu

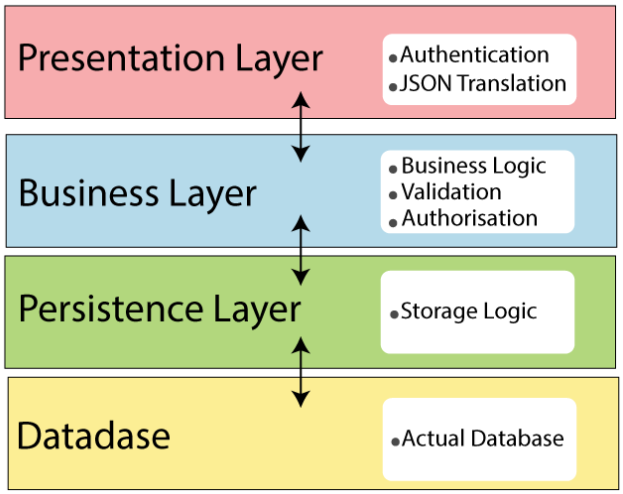
Pro implementaci backendové části byl zvolen framework Spring Boot. Pro použití tohoto frameworku lze použít Spring Initializr, který vygeneruje základní projekt, do kterého může programátor přidávat své změny. Tato metoda je k dispozici online[1], nebo lze použít vestavěnou metodu přímo v Intellij Idea, která byla použita v této práci. Při volbě této metody se projekt vytvoří přímo ve vývojovém prostředí a všechny potřebné

[1] https://start.spring.io

závislosti se vloží do konfiguračního souboru. V tuto chvíli již lze aplikaci spustit a aplikace běží na lokálním počítači na portu 8080.

### 7.1.3 Struktura projektu

Při implementaci serverové části pomocí Spring Boot je potřebné rozdělit strukturu aplikace do několika vrstev. Klientská strana dále s prezentační vrstvou komunikuje pouze pomocí DTO, které odesílá pomocí HTTP(S) dotazů.




Obrázek 7.1: Rozdělení vrstev ve Spring Boot

#### DTO

Data Transfer Object je objekt, který přenáší data mezi procesy. Je to pouze datová obálka a je nezávislá na implementaci. Strana serveru tyto objekty přijímá a dále s nimi pracuje podle vlastní logiky. Objekt DTO by se měl skládat především z primitivních datových typů a neměl by v sobě obsahovat žádnou logiku.

```
@Getter
@Setter
@NoArgsConstructor
public class CredentialsDto {

    private String username;

    private String password;
}
```

Ukázka kódu 7.1: Příklad DTO v jazyce Java

### Prezentační vrstva

Prezentační vrstva přijímá dotazy od uživatele (klienta) a dále je zpracovává. Jednotlivé třídy jsou označeny anotací *@RestController* a pro každou metodu je nutné specifikovat, jaký typ požadavku HTTP přijímá, jaká data přijímá a jaká data vrací uživateli.

```
@RestController
@RequestMapping
@Slf4j
@CrossOrigin
public class UserController {

    private final UserMapper userMapper;

    private final UserService userService;

    @Autowired
    public UserController(UserMapper userMapper, UserService userService
    ) {
        this.userMapper = userMapper;
        this.userService = userService;
    }

    @PostMapping(value = "/users", consumes = MediaType.
    APPLICATION_JSON_VALUE)
    public ResponseEntity<?> createUser(@RequestBody UserDto userDTO) {
        User user = userMapper.toUser(userDTO);
        user.setPassword(encoder.encode(userDTO.getPassword()));
        userService.createUser(user);
        log.info("New user created {}", user.getUsername());
        final HttpHeaders headers = RestUtil.createLocationHeaderNewUri(
    "/users/{username}", user.getUsername());
        return new ResponseEntity<>(headers, HttpStatus.CREATED);
    }
```

Ukázka kódu 7.2: Příklad Rest kontroleru v jazyce Java

### Mappery

Mappery se používají k přemapovávání objektů DTO na business objekty, se kterými se dále pracuje v dalších aplikačních vrstvách. Mappery se také používají v případě, že chceme DTO odeslat zpět uživateli, ale nechceme zobrazit všechny informace, například hesla nebo vzájemné závislosti s jinými objekty.

```java
@Component
public class TestMapper {

    public TestDto toDto(Test test) {
        TestDto testDto = new TestDto();
        testDto.setId(test.getId());
        testDto.setDescription(test.getDescription());
        testDto.setAssignments(test.getAssignments().stream().map(
    Assignment::getId).collect(Collectors.toList()));
        return testDto;
    }

    public Test toTest(TestDto testDto, User user, List<Assignment>
    assignmentList) {
        Test test = new Test();
        test.setDescription(testDto.getDescription());
        test.setCreateBy(user);
        test.setAssignments(assignmentList);
        return test;
    }
}
```

Ukázka kódu 7.3: Příklad mapperu v jazyce Java

### Business vrstva

Business vrstva obsahuje hlavní business logiku aplikace. Tato vrstva se stará o autorizaci a ověřování požadavků od klienta. Jednotlivé třídy jsou označeny anotací *@Service*. Business vrstva zajišťuje spojení mezi perzistentní a prezentační vrstvou.

```java
@Service
public class LectureService {

    private final LectureRepository lectureRepository;

    @Autowired
    public LectureService(LectureRepository lectureRepository) {
        this.lectureRepository = lectureRepository;
    }

    public void createOrUpdateLecture(Lecture lecture){
        lectureRepository.save(lecture);
    }

    public List<Lecture> getAllLecturesByUser(User user){
```

```
16          return lectureRepository.getAllByCreatedBy(user);
17      }
18
19      public Lecture getById(Long id){
20          return lectureRepository.getLectureById(id);
21      }
22
23
24      public Lecture getLast(User user) {
25          List<Lecture> entities = lectureRepository.getAllByCreatedBy(
    user);
26          if (entities.isEmpty()){
27              return null;
28          }
29          Lecture last = entities.get(0);
30          for (Lecture entity : entities) {
31              if (entity.getId() > last.getId()){
32                  last = entity;
33              }
34          }
35          return last;
36      }
37  }
```

Ukázka kódu 7.4: Příklad servisy v jazyce Java

### Perzistentní vrstva

Perzistentní vrstva je ve Spring Boot implementována pomocí repozitářů. Tyto repozitáře se označí pomocí anotace *@Repository* a kompletně řeší komunikaci aplikace s databází. Vygenerované repozitáře umožňují základní práci s business objekty, například uložení do databáze. V případě, že je potřeba implementovat složitější dotaz do databáze, stačí vytvořit metodu, která názvem odpovídá potřebnému dotazu. Velikou výhodou tohoto přístupu je, že není potřeba znát dotazovací jazyk pro používanou databázi, jelikož repozitář automaticky převede Java metodu na potřebný databázový dotaz.

```
1  @Repository
2  public interface QuizRepository extends JpaRepository<Quiz, Integer> {
3
4      List<Quiz> getAllByCreatedBy(User user);
5
6      Quiz getQuizById(Long id);
7  }
```

Ukázka kódu 7.5: Příklad repozitáře ve Spring Boot

#### Business objekty

Business objekty jsou základním objektem. Ukládají v sobě všechny potřebné informace sloužící k chodu aplikace, jsou navzájem provázané a jsou dále uložená v databázi.

```java
@Entity
@Table(name = "Users")
@Getter @Setter @NoArgsConstructor
public class User{
    @Id
    @GeneratedValue
    private Long id;

    @Column(unique = true)
    private String username;

    @Column(nullable = false)
    private String password;

    @Column(nullable = false)
    private String firstName;

    @Column(nullable = false)
    private String lastName;

    @Enumerated(EnumType.STRING)
    @Column(nullable = false)
    private UserType userType;
}
```

Ukázka kódu 7.6: Příklad business objektu v jazyce Java

### 7.1.4 Autorizace a autentifikace

Vzhledem k tomu, že každý uživatel aplikace vidí různá data a různé obrazovky v závislosti na své roli, je nutné, aby byl uživatel při každém požadavku na backend ověřen a autorizován. To je realizováno pomocí JSON webového tokenu. Použití JWT je doporučený webový standard pro vytváření přístupových tokenů, které se dále používají k identifikaci uživatele. Při prvním přihlášení uživatele je na backendové straně vygenerován jedinečný token, který je přiřazen aktuálnímu uživateli. Tento token je uložen na straně frontendu a odesílán při každém požadavku (více viz 7.2.4). Při každém dotazu na server je tento token ověřen, aby se ověřilo, že skutečně patří uživateli, který požadavek odeslal. Pokud zaslaný token nepatří uživateli, je požadavek zamítnut. Pokud je ověření úspěšné, je uživatel identifikován a jeho část je zobrazena podle přidělených rolí.

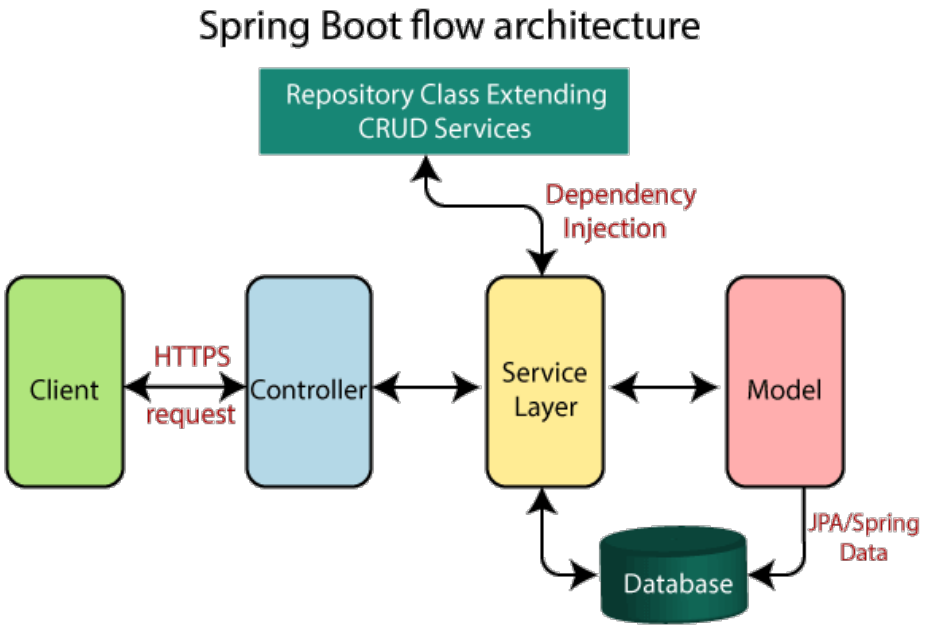
V případě této práce existují pouze dvě role - ROLE_TEACHER a ROLE_STUDENT. Autorizace se dále provádí pomocí anotace *@PreAuthorize("hasRole()")*, která zkontroluje přiřazené uživatelské role, a pokud má uživatel příslušnou roli, jsou údaje poskytnuty, jinak je požadavek opět zamítnut.

### 7.1.5 Problémy při implementaci

V listopadu 2022 byla vydána nová verze Spring Boot 3.0.0, což je pro vývoj velký problém, protože většina dostupné dokumentace a rad od ostatních programátorů je určena pro starší Spring Boot 2.x.x. Nová verze přinesla nový způsob zabezpečení, autorizace, zpracování požadavků a práce s entitami. Kvůli této změně je většina dostupné literatury nepoužitelná, což je při vývoji velký problém.

### 7.1.6 Závěr

Backendová část aplikace slouží ke zpracování požadavků od uživatele, manipulaci s daty a jejich ukládání do databáze. Při zadávání dotazu je nejprve identifikován uživatel a poté jsou data zaslaná z DTO zpracována mapperem. Zpracovaná data se dále používají v kontrolerech, které provolávají servisy. Tyto servisy provádějí veškerou potřebnou business logiku a volají příslušné repozitáře, když potřebují pracovat s databází. Tyto repozitáře pak pracují přímo s databází. Celý proces je znázorněn na následujícím obrázku 7.2.




Obrázek 7.2: Výsledná komunikace klienta se serverem

## 7.2 Frontend

V této části je popsáno, jak nastavit projekt pro vývoj frontendu, jak pracovat s Reactem a jak aplikaci implementovat a zabezpečit. Nakonec jsou v této části popsány problémy spojené s vývojem frontendové části aplikace.

### 7.2.1 Výběr vývojového prostředí

Pro vývoj frontendové části aplikace bylo rovněž zvoleno vývojové prostředí od společnosti JetBrains, konkrétně WebStorm. Toto vývojové prostředí poskytuje široký výběr nástrojů pro implementaci webové aplikace v jazyce Javascript a nabízí také vlastní server, pokud vývojář ještě nemá hotovou backendovou část.

### 7.2.2 Založení projektu

K vytvoření frontendové části projektu byl použit příkaz *npx create-react-app*, který vytvoří plně nakonfigurovanou aplikaci React, kterou stačí pouze přizpůsobovat. Pro spuštění tohoto příkazu musíte mít na svém počítači nainstalovaný Node.js a pro spuštění aplikace stačí v terminálu zadat *npm start*, čímž dojde ke kompilaci a spuštění aplikace. Aplikace je pak přístupná na lokálním počítači na portu 3000.

### 7.2.3 Základní práce s Reactem

Při vývoji v Reactu je nutné mít základní znalosti HTML, CSS a Javascriptu, protože na rozdíl od běžného přístupu Javascriptu, tj. přímého zásahu do *DOM*, React vytváří vlastní virtuální *DOM*, kde provádí veškeré změny a následně je realizuje v prohlížeči.[34] Kromě základních přístupů Javascriptu má React vlastní nadstavby, které jsou popsány dále.

#### Komponenty

Komponenty jsou nezávislé a opakovaně použitelné části kódu, které fungují podobně jako funkce, ale jsou izolované a vracejí HTML.

```
export const IndexPage = () => {
    return ( <div className={styles.mainContainer}>
                <HeaderSection/>
                <TextIndexSection/>
                <GraphSection/>
            </div> )}
```

Ukázka kódu 7.7: Základní struktura komponenty

Ukázka kódu 7.7 ukazuje komponentu *IndexPage*, která obsahuje další komponenty, což vede k přehlednějšímu kódu, protože jednotlivé funkce aplikace můžeme rozdělit do menších bloků, které lze kdekoli přepoužít.

### Props

Vzhledem k tomu, že se komponenty chovají jako funkce, můžeme těmto komponentám posílat data jako argumenty funkcí. Na rozdíl od funkcí nemusíme specifikovat, kolik argumentů komponenta očekává, ale pouze zavedeme *props*, které obsahují všechny předávané argumenty. Také nemusíme posílat všechny argumenty, které komponenta používá, nebo můžeme poslat více argumentů, které se jen nepoužijí.

```
const myComponent = <AllSquare text="Vytiskni toto" />;
```

```
export const AllSquare = (props) => {
    return (
        <div className={styles.square}>
            <div>{props.text}</div>
        </div>
    )
}
```

Ukázka kódu 7.8: Využití React props

### Hooks

React Hooks umožňují funkcionálnímu přístupu komponent použitých v této práci pracovat se stavem. Pokud se tento stav během běhu aplikace změní, část HTML závislá na tomto stavu se překreslí podle nového stavu.

```
    const [lectureDescription, setLectureDescription] = useState("")
    const [active, setActive] = useState(false)
    const [lectureEntities, setLectureEntities] = useState([])
    const [currentEntity, setCurrentEntity] = useState({})
    useEffect(()=> {
        LectureApi.getLectureById(params.get("id")).then(res => {
            console.log(res.data)
            LectureApi.isActive(params.get("id")).then(res => {
                setActive(res.data)
            })
        })
    },[params])
```

Ukázka kódu 7.9: Příklad využití Hooks

V ukázce 7.9 jsou znázorněny stavy aplikace pomocí *useState()* a také se využívá *useEffect*, což je metoda, která se volá při vytváření komponenty. Nejčastěji tato metoda získává data ze serveru nebo provádí další logiku. Zároveň je tato metoda volána při každé změně její závislosti, v tomto případě *params*.

#### React Router

React Router umožňuje vývojáři přiřadit každé adrese URL komponentu, která se při návštěvě adresy URL vykreslí.

```
1        <Routes>
2            <Route exact path="/" element={<IndexPage/>}/>
3            <Route exact path="/login" element={<LoginPage/>}/>
4            <Route exact path="/register" element={<RegistrationPage
   />}/>
5            ...
6        </Routes>
```

Ukázka kódu 7.10: Nastavení URL adres a komponent

### 7.2.4 Autorizace a autentifikace

Jak již bylo napsáno v 7.1.4, při přihlášení backendová strana vygeneruje JWT, který frontendová strana uloží do *Local Storage*[2] a při každém dalším požadavku na server je tento token odeslán v hlavičce požadavku. Současně je na základě této informace přiřazena role uživatele a zobrazují se pouze obrazovky, které této roli náleží. Pokud se uživatel odhlásí, tento token se smaže a pro další práci s aplikací je nutné se znovu přihlásit. Tento přístup zároveň řeší problém opakovaných návštěv, protože pokud uživatel stránku zavře a v budoucnu ji znovu navštíví, je automaticky přihlášen pod stejným profilem a nemusí se již znovu přihlašovat.

### 7.2.5 Komunikace se serverem

Pro komunikaci se serverem je možné použít základní metodu fetch(), ale pro tuto práci byla zvolena knihovna *react-axios*[35], protože axios sám převádí data, která chceme odeslat, do formátu JSON bez zásahu vývojáře. Tímto způsobem lze snadno odesílat listy a větší objekty. Při dotazování serveru pomocí systému axios určíme typ metody, což jsou základní metody HTTP - GET, POST, PATCH a DELETE. Dále zadáme adresu cílového místa, které chceme zavolat, a vložíme data, která chceme odeslat (pokud se nejedná o metodu GET nebo DELETE). Nakonec určíme, co se má stát po vrácení dat. Můžeme

[2]https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage

specifikovat, co se má stát, pokud je dotaz úspěšný, ale můžeme také specifikovat chování v případě, že server vrátí chybu.

```
const createDefinition = (description, definition, imageUrl) => {
    console.log("trying to create definition: " + description + " " +
   definition)
    return (axios.post(`${baseUrl}/definition`,
        {
            "description": description,
            "definition": definition,
            "imageUrl" : imageUrl
        },{headers}).then(response => {
        console.log(response)
        return response
    }));
}
```

Ukázka kódu 7.11: Komunikace se serverem

Jak je vidět v ukázce kódu 7.11, při dotazování serveru není třeba specifikovat uživatele, který dotaz odesílá, protože JWT je odeslán v *headers* a backend uživatele sám rozpozná.

### 7.2.6 Problémy při implementaci

Jediným větším problémem při implementaci bylo, že změna stavu v *useState([])*, který držel list, nepřekreslila část závislou na tomto stavu. Problém souvisí s tím, že JavaScript nevnímá změnu uvnitř listu jako změnu stavu, takže při načtení dat do listu se list uživateli nepřekreslí.

### 7.2.7 Využití ikonek

Při vývoji frontendu byly použity ikony od icons8[3], což značně usnadnilo vývoj a přehlednost aplikace.

### 7.2.8 Závěr

React využívá základní javascriptové technologie a přidává k nim vlastní technologie, díky nimž je vývoj webové aplikace komunikující s backendem rychlejší, jednodušší a pohodlnější.

[3]https://icons8.com/icons

## 7.3 Databáze

Chcete-li ukládat data, musíte mít připravenou databázi pro jejich ukládání. Pro lokální vývoj lze vytvořit lokální databázi, ale jakmile je aplikace nasazena na web, je třeba mít databázi, která běží i na serveru.

### 7.3.1 Připojení databáze

Připojení k databázi je velmi jednoduché, pokud backend používá Spring Boot, protože díky tomuto frameworku stačí zapsat několik řádků konfigurace do *application.properties*, které obsahují adresu databáze, uživatelské jméno, heslo a konkrétní driver[4]. Žádná další konfigurace a nastavení nejsou potřeba a připojení se vytvoří automaticky po spuštění aplikace.

```
spring.datasource.url=jdbc:postgresql://localhost:5432/tamburinektor
spring.datasource.username=tamburinek
spring.datasource.password=tamburinek
spring.datasource.jdbc-url=${spring.datasource.url}
spring.datasource.driver-class-name=org.postgresql.Driver
spring.sql.init.platform=postgres
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
```

Ukázka kódu 7.12: Připojení se k databázi

### 7.3.2 Vytváření tabulek

Vytváření tabulek je při použití Spring Boot také velmi snadné. Chcete-li vytvořit tabulku, stačí přidat anotaci *@Entity* nad objekt v jazyce Java a volitelně přidat název tabulky.

```
import javax.persistence.Table;
import javax.persistence.Entity;

@Entity
@Table(name = "Users")
public class User{}
```

Ukázka kódu 7.13: Definice tabulky

Po tomto kroku již není třeba do kódu Javy nic přidávat a stačí do souboru *application.properties* přidat další konfigurační řádky, které vytvoří příslušné SQL dotazy pro vytváření tabulek na základě objektů Javy.

[4] https://www.jdatalab.com/information_system/2017/02/16/database–driver.html

```
spring.jpa.generate-ddl=true
spring.jpa.hibernate.ddl-auto=update
spring.sql.init.mode=always
spring.jpa.defer-datasource-initialization=true
```

Ukázka kódu 7.14: Vygenerování tabulek

Po tomto kroku jsou vytvořeny všechny potřebné tabulky a backend a databáze jsou připraveny ke komunikaci.

## 7.4 Nasazení aplikace na web

Po dokončení implementace aplikace je třeba celý projekt nasadit na web, aby k němu uživatelé mohli přistupovat ze svých prohlížečů. Tato část popisuje nasazení backendu a frontendu. Databázi není třeba nasazovat, protože byla vybrána stávající školní databáze, která je již nasazena.

### 7.4.1 Nasazení backendu

Pro nasazení backendu byla zvolena platforma Heroku[5]. Pro nasazení aplikace stačí vytvořit repozitář Github a poskytnout jej platformě Heroku. Heroku si navíc může sama stáhnout nejnovější verzi kódu, sestavit aplikaci a nasadit ji na web. Jediné, co se po vývojáři vyžaduje, je přidání další konfigurace do souboru *build.gradle*. Jedinou nevýhodou Heroku je, že v současné době nemůžete jejich služby využívat zdarma. Výsledná aplikace také běží na adrese https://{název repozitáře}.herokuapp.com/.

```
task stage(type: Copy, dependsOn: [clean, build]) {
    from jar.archivePath
    into project.rootDir
    rename { 'app.jar'}
}
stage.mustRunAfter(clean)

jar {
    enabled = false
    manifest {attributes('Main-Class': 'fel.cvut.cz.tamburinektor.
    TamburinektorApplication')
    }
}
```

Ukázka kódu 7.15: Konfigurace Heroku

[5]https://www.heroku.com/home

### 7.4.2 Nasazení frontendu

Pro nasazení frontendové části byla zvolena platforma Vercel, která na rozdíl od Heroku umožňuje bezplatné nasazení aplikace. Nasazení probíhá podobně jako u backendové části, protože Vercel po připojení repozitáře Github také stáhne nejnovější verzi kódu a nasadí ji na web. Jediným problémem při nasazení aplikace React na web jsou adresy URL definované v souboru 7.2.3 a kvůli tomu je třeba vytvořit nový soubor *vercel.json*, který tento problém vyřeší. Výsledná aplikace se také spustí na adrese URL https://{název repozitáře}.vercel.app/.

```
{
  "rewrites": [
    {
      "source": "/((?!api/.*).*)",
      "destination": "/index.html"
    }
  ]
}
```

Ukázka kódu 7.16: Konfigurace Vercel

# Kapitola 8

# Testování aplikace

Tato kapitola popisuje testování vyvinuté aplikace pro podporu výuky matematiky. Testování je rozděleno na dvě části, přičemž první část probíhá v součinnosti s implementací, kdy se o testování stará vývojář, a druhá část je předána potenciálnímu koncovému zákazníkovi, který bude výslednou aplikaci používat.

## 8.1 Vývojářské testování

Během vývoje aplikace se testy provádějí současně s implementací funkcí. Testování backendu a frontendu se liší a je popsáno níže.

### 8.1.1 Backendová část

Testy backendu testují, zda je serverová část připravena na provoz aplikace v produkčním prostředí. Nejprve se provádějí unit testy, které testují malé bloky kódu, poté lze provádět integrační testy, které využívají externí závislosti, a nakonec se testuje, zda je aplikace připravena přijímat požadavky od více uživatelů naráz.

#### Unit testy

Unit testy testují nejmenší testovatelné bloky aplikace, kterými jsou metody. Pro správné otestování metody musí vývojář vytvořit všechny možné vstupy do této metody a otestovat, zda metoda vrací správná data nebo provádí správné operace. Pro jednotkové testování aplikací napsaných v jazyce Java se často používá framework JUnit[1], což platí i v kontextu této práce.

[1] https://junit.org/junit5/

```
    @Test
    void testGetLastMaterial_returnsLastMaterialIndex() {
        // Setup
        ...
        // Run the test
        ...
        // Verify the results
        ...
    }
```

Ukázka kódu 8.1: Struktura unit testu

Název unit testu by měl obsahovat testovanou metodu a očekávaný výstup. Struktura testu se skládá z nastavení všech potřebných proměnných a entit, poté ze spuštění testované metody a nakonec z kontroly, zda metoda vrátila očekávaný výstup.

#### Výkonové testy

Testy výkonu slouží k ověření stability, rychlosti a odezvy celé aplikace. Pro tuto práci bylo obzvláště důležité otestovat, zda aplikaci může používat více uživatelů najednou, protože hlavním využitím aplikace je výuka matematiky na středních školách, kde je v současné době ve třídě přibližně 30 studentů. Z tohoto důvodu byly napsány skripty, které po spuštění vytvořily více než 30 uživatelů a testovaly, zda aplikace zvládne tento počet najednou.

### 8.1.2 Frontendová část

Frontendové testy prováděné vývojářem testují, zda se všechny komponenty vykreslují správně a ve správné pozici.

#### Testování prohlížeče

Vzhledem k tomu, že každý prohlížeč vykresluje jednotlivé komponenty trochu jinak, je nutné otestovat chování jednotlivých komponent, aby bylo zajištěno, že se ve všech prohlížečích vykreslí vše podle zamýšleného návrhu a že nedochází k nesrovnalostem. Proto bylo chování aplikace testováno v nejpoužívanějších prohlížečích, tedy v Safari, Chrome a Firefoxu, kde byly testovány zejména posuvníky, protože při implementaci jsem narazil na problém s jejich vykreslováním.

#### Test přístupnosti

Při testech přístupnosti bylo ověřeno, že se uživatel nikdy nemůže zaseknout na jedné obrazovce a že všechna tlačítka a odkazy fungují přesně tak, jak mají. Tímto testem může být například správné přesměrování uživatele po přihlášení nebo odhlášení.

## 8.2 Uživatelské testování

Tato část popisuje uživatelské testování aplikace, které provedli potenciální koncoví zákazníci a nezávislí uživatelé. Pro účely testování byly vytvořeny testovací scénáře a proces testování je popsán níže.

### 8.2.1 Testovcí scénáře

Testovací scénáře obsahují jednotlivé kroky, které má uživatel provést, aby otestoval funkčnost aplikace. Vzhledem k tomu, že se aplikace chová odlišně podle přihlášeného uživatele, jsou scénáře rozděleny na scénáře pro učitele a scénáře pro studenty.

#### Učitel

Testovací scénáře pro učitele obsahují základní funkce popsané v 5.6, tedy tvorbu materiálů a další manipulaci s nimi, tvorbu a správu lekcí a totéž platí pro testy. Nakonec byla testována správa tříd.

#### Student

Pro testování studentské části aplikace byly vytvořeny scénáře, které kontrolovaly možnost připojit se ke třídě, zobrazit lekce a spustit je, a totéž pro testy.

### 8.2.2 Průběh testování

Test byl proveden na dvou skupinách uživatelů, přičemž první skupinu tvořili budoucí potenciální zákazníci a druhou skupinu nezávislí uživatelé, kteří aplikaci nebudou používat, ale mají zkušenosti s vývojem webových aplikací. Oběma skupinám byly zadány úkoly z 8.2.1 a během procházení úkolů bylo sledováno chování uživatelů, zda vědí, kam kliknout a co dělat dál, a také proběhla volná diskuse o použitelnosti aplikace.

## 8.3 Závěr

Vytvořená aplikace byla testována pomocí unit testů, které testují malé bloky kódu. Dále byly provedeny výkonové testy pro kontrolu funkčnosti backendu. Frontendová strana byla testována pomocí testů přístupnosti a bylo ověřeno, zda aplikace funguje správně ve všech moderních prohlížečích. Poté byla aplikace předána uživatelům k uživatelskému testování. Výsledky testování a případné úpravy jsou sepsány v sekci 9.2.

# Kapitola 9

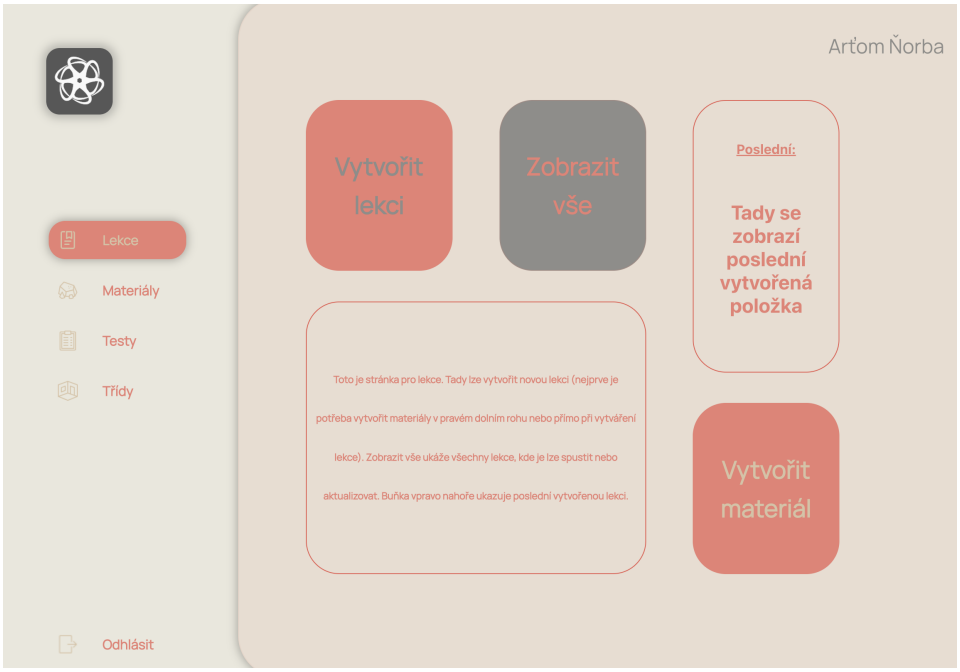
# Vyhodnocení výstupů práce

## 9.1 Vytvořená aplikace

Byla vytvořena webová aplikace, která slouží učitelům matematiky i jejím studentům. V této části jsou popsány základní vytvořené obrazovky a funkcionality aplikace.

### 9.1.1 Dashboard

V závislosti na přihlášeném uživateli se mění obsah ovládacího panelu, který slouží jako hlavní rozcestník aplikace. Tento ovládací panel obsahuje lekce a testy, navíc má učitel k dispozici správu materiálů.




Obrázek 9.1: Dashboard pro učitele


Obrázek 9.2: Dashboard pro studenta

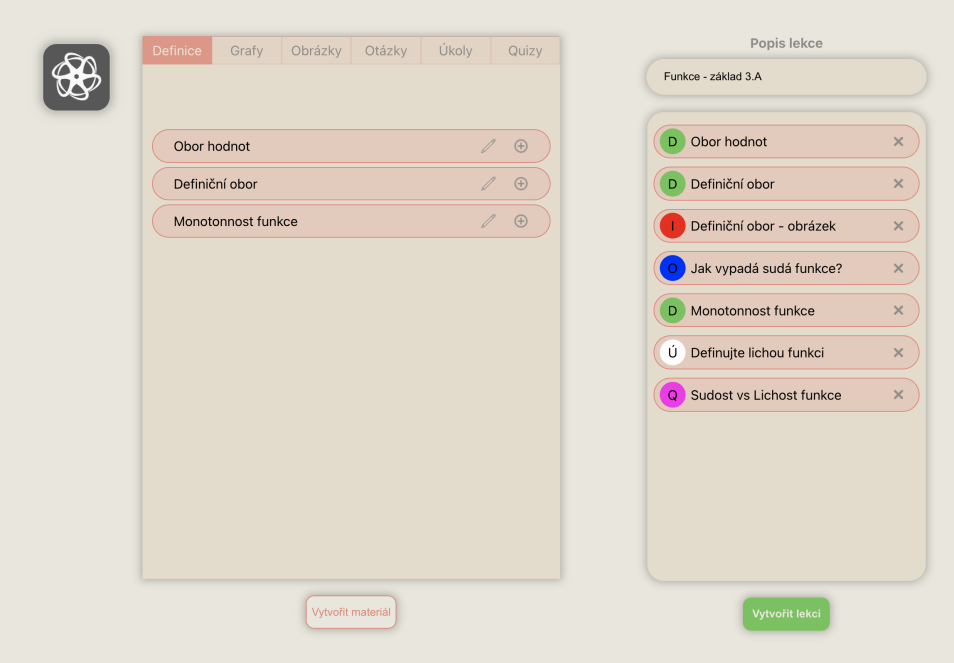
### 9.1.2 Vytváření materiálů

Uživatel přihlášený jako učitel může vytvářet a upravovat materiály, které jsou základem tvorby lekcí.


Obrázek 9.3: Vytváření definice

### 9.1.3 Vytváření lekce

Učitel má možnost vytvořit a upravit lekci, která se skládá z jednotlivých materiálů, které učitel připravil, a může tuto lekci prezentovat studentům ve třídě nebo ji připravit k domácímu studiu.




Obrázek 9.4: Vytváření lekce

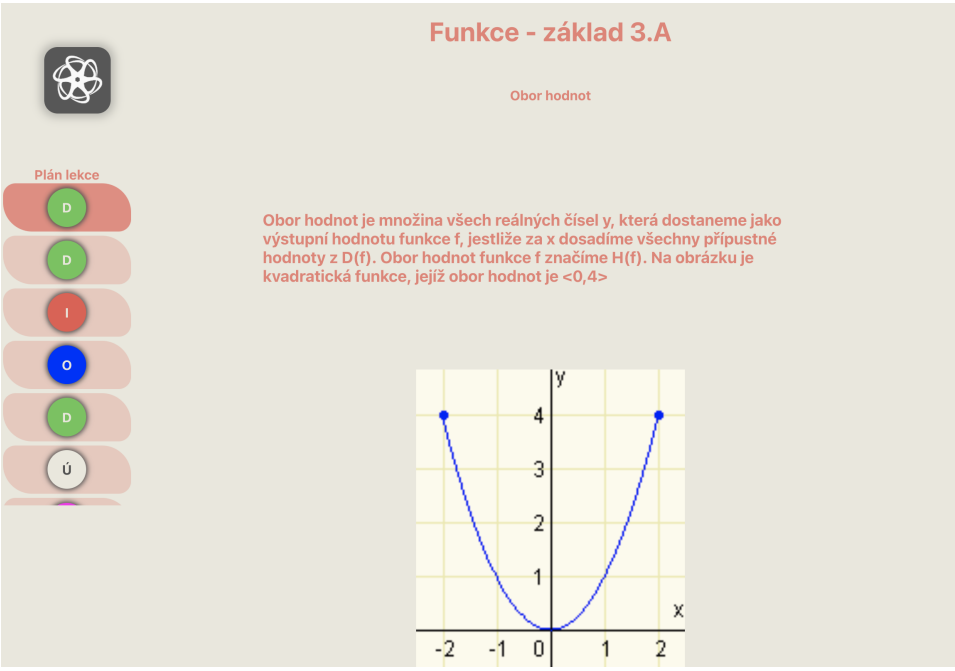
### 9.1.4 Vytváření testů

Učitel má možnost vytvářet a upravovat testy, které se skládají z otevřených nebo uzavřených otázek, které může učitel vytvořit.


Obrázek 9.5: Vytváření testů

### 9.1.5 Prezentace lekce

Učitel má možnost prezentovat učivo ve třídě. Pokud to učitel ve vedení třídy dovolí, může se s učivem seznámit i žák.




Obrázek 9.6: Zobrazení lekce

## 9.2 Možnosti dalšího rozvoje aplikace

Tato část zpracovává výsledky testování a navrhuje další vývoj aplikace, který může být proveden jako diplomová práce.

1. **Větší rozmanitost materiálů** - aplikace by mohly být rozšířeny na další typy materiálů.

2. **Podpora latexu v materiálech** - vytvořené materiály by mohly podporovat syntaxi Latexu a zobrazovat definice, rovnice a další matematické vzorce pomocí Latexu.

3. **Responsivita pro mobilní a jiná zařízení** - v současné době je aplikace vyvinuta pro počítače a větší zařízení, takže by mohla být upravena tak, aby design podporoval menší zařízení.

4. **Banka úloh dostupná všem** - vytváření nových materiálů je pro nového uživatele aplikace časově náročné, proto by bylo dobré mít definice, úkoly a další materiály připravené tak, aby k nim měl každý přístup ihned po registraci. Zároveň by učitelé mohli mít možnost materiály mezi sebou sdílet.

5. **Přidání statistiky** - aplikace by mohla zobrazovat dosavadní pokroky studentů.

# Kapitola 10

# Závěr

Předmětem této práce je analýza současného přístupu k výuce matematiky na středních školách, zejména k výuce matematických funkcí, a také návrh a realizace možného zlepšení výuky, které bude sloužit především učitelům středních škol, ale také jejich studentům.

V analýze současného stavu je nejprve popsán úvod do matematického vzdělávání v kapitole 2, kde se hovoří o matematice jako takové, a postupně je v této kapitole popsána matematika ve školství.

Dále jsou v kapitole 3 vysvětleny možnosti integrace informačních a komunikačních technologií do výuky matematiky. Tato kapitola popisuje, jak by technologie mohly naplnit cíle ministerstva školství v oblasti rozvoje učebních osnov ve Strategii 2030+. Tato kapitola také upozorňuje na různé metodiky, které slouží ke zvýšení zapojení žáků do výuky, a tím ke zvýšení zájmu o učení.

Poslední kapitola analytické části této práce 4 se zabývá zkoumáním v současnosti dostupných aplikací, které slouží žákům a učitelům k hravému a efektivnímu učení látky. Kapitola je rozdělena na aplikace, které jsou určeny k podpoře výuky jako takové, a aplikace, které slouží přímo k výuce matematiky.

Další kapitola 5 uvádí druhou část této práce, která se zabývá návrhem, implementací a testováním aplikace. Konkrétně se tato kapitola zabývá návrhem aplikace, což znamená modelování aktérů systému, vypsání systémových požadavků, které se dále dělí na funkční a nefunkční požadavky. V neposlední řadě tato kapitola obsahuje diagram tříd, na jehož podkladě byl vytvořen základ backendové části aplikace. V závěru kapitoly jsou uvedeny případy užití jednotlivých aktérů, diagram tříd a přibližný návrh aplikace, na jehož bázi byl vytvořen základ frontendové části aplikace.

V kapitole 6 je provedena analýza a následný výběr jednotlivých technologií pro vývoj, které byly dále využity při implementaci aplikace. Technologie byly vybrány pro frontendovou část, backendovou část a nakonec pro databázi.

Poté následuje kapitola 7, která se zabývá samotnou implementací aplikace. Kapitola

je rozdělena na backendovou část, frontendovou část, připojení k databázi a nasazení aplikace na web. V částech backend a frontend jsou popsány základní principy práce se zvolenou technologií, konkrétní kroky implementace a způsob zabezpečení aplikace.

Další kapitola technologické části práce 8 popisuje testování aplikace vývojářem a následně koncovými a dalšími nezávislými uživateli.

Poslední kapitola 9 shrnuje výstupy celé práce, kterými jsou vytvořená webová aplikace, jednotlivé funkcionality a možnosti dalšího rozvoje aplikace.

Vyvinutá aplikace je v současné době určena pro výuku matematiky, ale v budoucnu by mohla být rozšířena pro jakýkoli předmět a obor.

# Bibliografie


[1] ČTK a A. Brzybohatá, *Povinná maturita z matematiky nebude, prezident Zeman podepsal její zrušení*, čvn. 2020. URL: https://www.idnes.cz/zpravy/domaci/maturita-matika-matematika-zeman-podepsal-zruseni-povinne-maturity-z-matematiky.A200617_120519_domaci_brzy.

[2] P. Spáčil, *Historie matematiky*, 2010. URL: http://www.geo-info-mat.cz/mat_historie.php.

[3] Jihočeská univerzita v Českých Budějovicích, *Dějiny matematiky*. URL: https://old.pf.jcu.cz/stru/katedry/m/knihy/DejinyM.pdf.

[4] Tennessee Tech University, *What is Mathematics?* URL: https://www.tntech.edu/cas/math/what-is-mathematics.php.

[5] Cambridge Business School, *Matematika*. URL: http://www.cambschool.eu/Matematika.

[6] R. Zeman, *Úvod do funkce a její definice*. URL: https://onlineschool.cz/matematika/uvod-do-funkce-a-jeji-definice/.

[7] L. Havrlant, *Co je to funkce*. URL: https://www.matweb.cz/co-je-to-funkce/.

[8] Výzkumný ústav pedagogický v Praze, *Rámcový vzdělávací program pro gymnázia*, 2007. URL: https://www.msmt.cz/file/7150_1_2/.

[9] Centrum pro zjišťování výsledků vzdělávání, *Agregované výsledky maturitní zkoušky*. URL: https://vysledky.cermat.cz/data/Default.aspx.

[10] P. Mgr. Miroslav Půža, *Využití ICT ve výuce*. URL: https://digifolio.rvp.cz/artefact/file/download.php?file=71667&view=11067.

[11] MŠMT ČR, *Strategie vzdělávací politiky ČR do roku 2030+*. URL: https://www.msmt.cz/vzdelavani/skolstvi-v-cr/strategie-2030.

[12] ZČU Plzeň, *Konstruktivismus v praxi VŠ*, 2010. URL: https://www.konstruktiv.zcu.cz/menu.php?akce=construct.

[13] IT Slovník, *Co je to Gamifikace?* URL: https://it-slovnik.cz/pojem/gamifikace/?utm_source=cp.

[14] Z-AGENCY, *Gamifikace a její využití ve vzdělávání*. URL: https://www.terrahunt.cz/blog/gamifikace-a-jeji-vyuziti-ve-vzdelavani.

[15] IT Slovník, *Co je to MVC?* URL: https://it-slovnik.cz/pojem/mvc.

[16] GeeksforGeeks, *Frontend vs Backend*, led. 2023. URL: https://www.geeksforgeeks.org/frontend-vs-backend/.

[17] GeeksforGeeks, *ReactJS Tutorials*. URL: https://www.geeksforgeeks.org/reactjs-tutorials/.

[18] B. Skuza, *Pros and Cons of React Native Development in 2022 — Business Perspective*, led. 2022. URL: https://www.thedroidsonroids.com/blog/react-native-pros-and-cons?utm_source=google.

[19] GeeksforGeeks, *Angular 8 — Introduction*, zář. 2020. URL: https://www.geeksforgeeks.org/angular-8-introduction/.

[20] AltexSoft, *The Good and the Bad of Angular Development*, ún. 2020. URL: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/.

[21] GeeksforGeeks, *Vue.js — Introduction & Installation*, čvn. 2022. URL: https://www.geeksforgeeks.org/vue-js-introduction-installation/.

[22] AltexSoft, *The Good and the Bad of Vue.js Framework Programming*, ún. 2020. URL: https://www.altexsoft.com/blog/engineering/pros-and-cons-of-vue-js/.

[23] J. Štráfelda, *Co je backend*, břez. 2022. URL: https://www.strafelda.cz/backend.

[24] Shoptet, *Backend*. URL: https://www.shoptet.cz/slovnik-pojmu/backend/.

[25] Oracle, *What is Java technology and why do I need it?* URL: https://www.java.com/en/download/help/whatis_java.html.

[26] DataFlair, *Pros and Cons of Java*. URL: https://data-flair.training/blogs/pros-and-cons-of-java/.

[27] B. Lutkevich, *Kotlin*, pros. 2022. URL: https://www.techtarget.com/whatis/definition/Kotlin.

[28] G. Samojło a Netguru, *Pros and Cons of Kotlin for Android App Development*, 2022. URL: https://www.netguru.com/blog/kotlin-pros-and-cons.

[29] J. Štráfelda, *Co je PHP*, břez. 2021. URL: https://www.strafelda.cz/php.

[30] Editorial Team, *Advantages and Disadvantages of PHP Programming Language — EPAM Anywhere Business*. URL: https://anywhere.epam.com/business/pros-and-cons-of-php.

[31] GeeksforGeeks, *What is Database?*, pros. 2022. URL: https://www.geeksforgeeks.org/what-is-database/.

[32] Gymnázium Čakovice, *Relační databáze*. URL: http://vyuka.greendot.cz/materialy/material-4.pdf.

[33] Martinekuan, *Nerelační data a NoSQL - Azure Architecture Center*. URL: https://learn.microsoft.com/cs-cz/azure/architecture/data-guide/big-data/non-relational-data.

[34] *Introduction to React*. URL: https://www.w3schools.com/REACT/react_intro.asp.

[35] F. Kelhini, *Axios vs. fetch(): Which is best for making HTTP requests? - LogRocket Blog*, ún. 2022. URL: https://blog.logrocket.com/axios-vs-fetch-best-http-requests/.

# Struktura elektronické verze práce