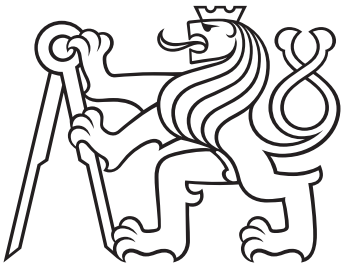
České vysoké učení technické v Praze
Fakulta elektrotechnická

**Katedra počítačů**
**Obor: Softwarové inženýrství a technologie**



# Příprava na přijímací zkoušky z matematiky

# Preparation for the mathematics entrance exams

BAKALÁŘSKÁ PRÁCE

Vypracoval: František Severin
Vedoucí práce: RNDr. Ingrid Nagyová, Ph.D.
Rok: 2024

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Severin** Jméno: **František** Osobní číslo: **503231**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

## II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

**Příprava na přijímací zkoušky z matematiky**

Název bakalářské práce anglicky:

**Preparation for the mathematics entrance exams**

Pokyny pro vypracování:

Cílem projektu je rozšířit možnosti využití digitálních technologií v přípravě na přijímací zkoušky na střední školy z matematiky. Práce se zaměří na téma rovnic.
1. Seznamte se s vládní strategií v oblasti základního vzdělávání. Prostudujte si cíle a očekávané výstupy RVP pro základní vzdělávání v oblasti matematiky a analyzujte, s jakými typy příkladů na téma rovnice se žáci setkávají.
2. Vytvořte sbírku typických příkladů na rovnice, které bývají součástí přijímacích testů na střední školy. Analyzujte postup řešení příkladů a příklady kategorizujte.
3. Vyhledejte portály, mobilní aplikace a videa, které složí žákům k procvičování rovnic a vyhodnoťte jejich přínos pro přípravu žáků na přijímací zkoušky. Inspirujte se i v zahraničí.
4. Navrhněte a implementujte systém, který usnadní žákům základní školy přípravu na přijímací zkoušky z matematiky.
5. Vytvořený systém pilotně otestujte, ideálně na vybrané základní škole.

Seznam doporučené literatury:

Jednotná přijímací zkouška. Praha: Centrum pro zjišťování výsledků vzdělávání, 2019. Dostupné na: https://prijimacky.cermat.cz/
Jeřábek, J. a kol. RVP pro základní vzdělávání. Praha: Národní ústav pro vzdělávání, 2021. Dostupné na: https://www.edu.cz/rvp-ramcove-vzdelavaci-programy/
Státní přijímačky z matematiky na SŠ. Nový Amos, 2023. Dostupné na: https://www.statniprijimacky.cz/matematika
Fryč J. a kol. Strategie vzdělávací politiky ČR do roku 2030+. MŠMT České republiky, 2020. Dostupné na: https://www.msmt.cz/vzdelavani/skolstvi-v-cr/strategie-2030

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: **14.08.2023** Termín odevzdání bakalářské práce: **09.01.2024**

Platnost zadání bakalářské práce: **16.02.2025**

| 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

**Prohlášení**

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 4.1.2024 .....

František Severin

**Poděkování**

Děkuji vedoucí práce RNDr. Ingrid Nagyové, Ph.D. za pečlivé vedení práce, odborný dohled a rady. Dále také děkuji Mgr. Ivě Vackové, Mgr. Monice Polanské a Mgr. Renatě Drážné za umožnění testování aplikace během vyučovacích hodin matematiky na základní škole v Červené Vodě. Za zprostředkování také děkuji řediteli školy, Mgr. Luďku Bílému.

František Severin

*Název práce:*
**Příprava na přijímací zkoušky z matematiky**
*Autor:* František Severin

*Studijní program:*
*Obor:* Softwarové inženýrství a technologie
*Druh práce:* Bakalářská práce

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

*Abstrakt:* Tato práce se zabývá problematikou přípravy žáků základních škol na jednotné přijímací zkoušky z matematiky na střední školy, především tématem rovnic. Čerpá z dostupných dokumentů Ministerstva školství, mládeže a tělovýchovy České republiky, specificky dokumentu Strategie 2030+, kterým MŠMT určilo cíle školství pro následujících deset let a mezi něž patří také digitalizace vzdělávání. Výsledkem analýzy již existujících alternativ pro přípravu na přijímací zkoušky z matematiky, jako jsou různé webové portály s videotutoriály, cvičnými testy apod., je poté návrh a vytvoření řešení. Tím je podpůrný systém, který kombinuje výuku témat přítomných v testech, jejich procvičování a následnou aplikace nabytých znalostí ve formě cvičného testu vytvořeného na základě dostupných testů z minulých let. Systém je nakonec podroben testování na určené základní škole během vyučovací hodiny matematiky žáků 9. třídy.
*Klíčová slova:* digitalizace, přijímací zkoušky, základní škola, matematika, generování matematických příkladů, mobilní aplikace, webová aplikace, framework Flutter

*Title:*
**Preparation for the mathematics entrance exams**
*Author:* František Severin

*Abstract:* The thesis deals with the issue of preparation for common secondary school entrance examinations in mathematics with the focus on equations. Main sources for the analysis are documents published by Ministry of Education, Youth and Sport of the Czech Republic, e.g. Strategie 2030+. With this document the Ministry defined its goals for upcoming decade, one of those goals being digitalization of education. After analysis of already existing options for preparation for entrance exams in mathematics, such as various videotutorial or practice test websites, a proposal of a solution is created. From the solution a educational system is developed, one that offers its users opportunities to study specific topics that examinations contain, practice them and then attempt a mock test based on available older tests. Finally, the system is tested during a real 9th grade math lesson in a chosen primary school.
*Key words:* digitalization, entrance examinations, primary school, mathematics, math problem generation, mobile application, web application, framework Flutter

# Obsah

# Seznam použitých zkratek

**JPZ** Jednotná přijímací zkouška

**MŠMT** Ministerstvo školství, mládeže a tělovýchovy

**MŠMT ČR** Ministerstvo školství, mládeže a tělovýchovy České republiky

**RVP** Rámcové vzdělávací programy

**RVP ZV** Rámcový vzdělávací program pro základní vzdělávání

**RVP PV** Rámcový vzdělávací program pro předškolní vzdělávání

**RVP G** Rámcový vzdělávací program pro gymnázia

**RVP SOV** Rámcový vzdělávací program středního odborného vzdělávání

**ŠVP** Školní vzdělávací program

**EU** Evropská unie

**ČR** Česká republika

**ZŠ** základní škola

**SŠ** střední škola

**VŠ** vysoká škola

**UI** user interface (překl. uživatelské rozhraní)

**UML** Unified Modeling Language

**BaaS** Backend as a Service

# Seznam obrázků



# Seznam tabulek

# Úvod

Tato práce se zabývá analýzou možností digitalizace přípravy žáků základních škol na přijímací zkoušky z matematiky, konkrétně tématem rovnic. Dále se zabývá návrhem a realizací pomocného systému, jehož hlavním účelem je tuto činnost podporovat jak z hlediska výuky učiva, tak i jeho procvičování. Součástí je také rozbor stávajících výukových dokumentů a obecně učebních praktik v rámci tohoto tématu a sběr typických matematických příkladů z přijímacích testů.

Hlavním cílem práce je podpořit přípravu žáků základních škol na přijímací zkoušky z matematiky. Pro dosažení tohoto cíle jsme definovali následující dílčí cíle:

- analýza tématu rovnic, a to jak z pohledu rovnic jako takových, tak i jakým způsobem jsou vyučovány v současném školství a jak je tato výuka vymezena;
- rozbor existujících portálů a mobilních aplikací na to téma, jejich porovnání a možný přínos pro vlastní platformu;
- návrh a implementace podpůrného systému, který je zaměřen specificky na přípravu studentů na část přijímací zkoušky věnované rovnicím a s nimi souvisejícím příkladům a který nabízí výuku učiva, sbírku příkladů pro volné řešení a cvičné testy;
- otestování systému na vybrané základní škole.

Struktura teoretické části odpovídá zadaným cílům práce a je rozdělena celkem do 6 kapitol. První kapitola obsahuje rozbor současného stavu výuky matematiky na 2. stupni základních škol. Obsahuje analýzu dokumentů ministerstva školství ČR, jako jsou například Rámcové vzdělávací programy, a jejím výstupem je obecný přehled o tom, co by systém pro přípravu na přijímací zkoušky měl obsahovat. Druhá kapitola se věnuje již existujícím platformám pro výuku a procvičování nejen v rámci přípravy na přijímací zkoušky, ale matematiky obecně. Hlavním výstupem jsou poznatky z analyzovaných webových stránek či mobilních aplikací pro účely navrhnutí vlastního systému. Dále také kapitola obsahuje porovnání těchto platforem a analýzu obsahu přijímacích testů z minulých let, s touto částí souvisí také externí krátká sbírka příkladů. Následující kapitola je zaměřena na návrh požadavků na obsah a funkcionalitu systému spolu s jejich zobrazením ve formě diagramu případů užití. Čtvrtá kapitola obsahuje samotný technický návrh aplikace, jehož součástí je i výběr formy aplikace, spolu s rešerší dostupných technologií a výběrem z nich. Kapitola dále obsahuje návrh struktury objektů v aplikaci, návrh její architektury a vizuální formy právě na základě zvolených nástrojů pro vývoj systému. Následuje kapitola o průběhu implementace, kde jsou především podrobněji popsány nástroje použité k realizaci systému a jejich použití při vývoji. Součástí této kapitoly je i porovnání současného stavu systému oproti návrhu. Poslední, šestá kapitola se zabývá

testováním aplikace a to jak interním testováním ve formě jednotkových a integračních testů, tak i testováním uživatelským.

Praktickými výstupy práce jsou výuková aplikace, která je prozatím nasazená ve formě webové aplikace a jako instalační soubor pro operační systémy Android, a externí knihovna pro generování matematických příkladů ve vybraných tématech.

# Kapitola 1

# Přehled vzdělávání

Cílem této kapitoly je především představit hlavní téma, rovnice, a nastínit způsob, jakým je matematika prezentována na základních školách. První a druhá podkapitola se zaměří na téma přijímacích zkoušek a téma rovnic. Důležitou částí je poté přehled vymezení výuky a očekávaných výstupů studentů po ukončení základního vzdělávání. Veškeré informace pocházejí z dokumentů MŠMT[1], přičemž každému individuálnímu dokumentu je věnována samostatná podkapitola. Poslední podkapitola slouží jako souhrn předchozích podkapitol a obsahuje závěry dosažené analýzou učiva rovnic.

## 1.1 Motivace

Organizace Cermat pod hlavičkou MŠMT zajišťuje od roku 2017 jednotné přijímací zkoušky pro střední školy. [29] Ty jsou rozděleny do dvou částí, český jazyk a matematika. Nedílnou součástí testů z matematiky jsou příklady spojené s tématem rovnic. Těm předchází operace se zápornými čísly, zlomky, mocninami a dalšími, a navazují na ně různé slovní úlohy. Samotné rovnice lze také rozdělit do několika kategorií podle obtížnosti, ať už se jedná o rovnice se zlomky, rovnice s násobením závorek s neznámými podle vzorců, soustavy rovnic a další. Všechny tyto typy příkladů jsou zahrnuty ve standardech RVP ZV[2], ale vzhledem k volnosti, kterou školy při rozdělování učiva do jednotlivých ročníků mají, nemusí nutně být úroveň znalostí všech žáků účastnících se přijímacích zkoušek vyrovnaná. Pro účely přijímacích zkoušek proto některé ze základních škol nabízí doučování nad rámec standardní výuky.

V návaznosti na dokument Strategie 2030+ (více v podkapitole Strategie 2030+), který vytyčil jako jednu ze strategických linií také digitalizaci výuky, a zároveň, jak je později znázorněno v kapitole dostupných online zdrojů (viz obrázkový přehled 2.3), neexistuje online pomůcka zaměřená přímo na přípravu na přijímací zkoušky z matematiky, jež by dokázala opravdu zahrnout všechny součásti výuky určitého tématu - od ukázek, procvičování až po reálnou aplikaci v rámci přijímacích testů. Nutné je dodat, že příklady v přijímacích testech lze rozdělit do dvou kategorií, které lze spojit s tématickými okruhy ze standardů RVP ZV (více v kapitole 1.5): Číslo a početní operace (početní operace, příklady na úpravu výrazů, rovnice nebo slovní úlohy) a Geometrie v rovině a v prostoru (úlohy na výpočet obvodu či obsahu, úlohy

[1] Ministerstvo školství, mládeže a tělovýchovy
[2] Rámcový vzdělávací program pro základní vzdělávání

na konstrukci nebo početní operace s úhly). Tato práce se věnuje pouze kategorií první.

## 1.2 Jednotné přijímací zkoušky v ČR

Jak již bylo zmíněno v předchozí podkapitole Motivace, tak od roku 2017 jsou uchazeči o maturitní obory povinni se zúčastnit jednotné přijímací zkoušky. Výsledek přijímací zkoušky se na celkovém vyhodnocování přijetí do oboru podílí nejméně 60%, existují ale také výjimky.[1] Organizace Cermat pro účely přijímacích zkoušek stanovila podle RVP ZV minimální požadavky pro přijetí do maturitních oborů.[26] Definice těchto požadavků je řízena určitými dokumenty MŠMT, které vyměřují obsah učiva, požadavky na získané znalosti a indikátory postupu. Tyto dokumenty jsou představeny v následujících podkapitolách.

## 1.3 Bílá kniha

Obecně Bílá kniha představuje doporučující dokument v rámci členských států EU[3]. Představuje návrhy na řešení určitých společenských problémů. Jednou z knih, které vstoupily v platnost v ČR[4] je Národní program rozvoje vzdělávání v České republice (dále Bílá Kniha MŠMT ČR[5]).

Samotná Bílá kniha MŠMT ČR pak představuje závazný základ pro budoucí realizační plány tohoto resortu. Je otevřeným dokumentem a může tak být v předmětem revizí a jiných úprav. Byla představena v roce 2001 na základě výročních zpráv MŠMT předchozích let (např. "Školství na křižovatce"z r.1999 nebo "Na prahu změn"z r.2000).

Schválením dokumentu "Strategie do roku 2020"v roce 2014 pozbyla Bílá kniha platnosti. Na tento dokument poté pro léta 2020 - 2030 navazuje dokument Strategie 2030+. [18]

## 1.4 Strategie 2030+

V roce 2020 skončila Strategie vzdělávací politiky do roku 2020 a proto MŠMT pro rozvoj vzdělávání v roce 2018 zahájilo přípravu dokumentu pro následující období (2020 - 2030+). Osmičlenný tým poté podle vytyčených cílů vypracoval dokument Strategie 2030+ [7] a ten byl 19.10.2020 schválen vládou ČR. [28] Tyto cíle byly dva:

- Zaměřit vzdělávání více na získávání kompetencí potřebných pro aktivní občanský, profesní i osobní život.
- Snížit nerovnosti v přístupu ke kvalitnímu vzdělávání a umožnit maximální rozvoj potenciálu dětí, žáků a studentů.

K dosažení těchto cílů bylo následně vytyčeno 5 strategických linií. Jednou z nich je *Proměna obsahu, způsobu a hodnocení vzdělání*, což ve výsledku znamená především pokusit se do výuky implementovat moderní technologie skrze digitalizaci obsahu.

[3] Evropská unie
[4] Česká republika
[5] Ministerstvo školství, mládeže a tělovýchovy České republiky

Strategie byla rozdělena do tří implementačních období obsahujících harmonogram s indikátory pro následné zhodnocení plnění stanovených cílů. Ty jsou rozděleny do karet opatření a jednou z nich je revize rámcových vzdělávacích programů. [17]

## 1.5 Rámcové vzdělávací programy - RVP

### 1.5.1 Popis

Definice: "RVP[6] tvoří obecně závazný rámec pro tvorbu školních vzdělávacích programů škol všech oborů vzdělání v předškolním, základním, základním uměleckém, jazykovém a středním vzdělávání."[6]

RVP pro základní vzdělávání je navržen tak, aby navazoval na RVP PV[7] (předškolní) a zároveň poskytoval dostatečnou úroveň výuky pro přechod na RVP G[8] či RVP SOV[9] (gymnázia a střední odborné vzdělávání respektive). Ačkoliv je především určen pro ZŠ[10], vzdělávací obsah 2. stupně odpovídá vymezení obsahu pro šesti- či osmiletá gymnázia v odpovídajících ročnících. Je také podkladem pro tvorbu přijímacích zkoušek pro střední školy. [6]

### 1.5.2 Vzdělávací oblast v rámci RVP

RVP ZV je rozdělen do 9 vzdělávacích oblastí mezi něž patří např. Jazyková komunikace (Český jazyk a cizí jazyky), Matematika a její aplikace či Člověk a příroda (Fyzika, Přírodopis, Chemie a Zeměpis). Každý vzdělávací obor (předmět) je poté definován očekávanými výstupy a učivem. Pro přehlednost je rozdělen do 3 částí (1. období 1. stupně = 1. až 3. třída, 2. období 1. stupně = 4. a 5. třída, 2. stupeň = 6. až 9. třída). [6]

**Očekávané výstupy** jsou založeny na osvojení si prakticky využitelných znalostí vymezených daným výstupem na konci 5. a 9. ročníku. Jsou identifikovatelné pomocí kódu, který obsahuje:

- zkratku oboru (např. M = Matematika a její aplikace),
- označení ročníku (např. 09 = 9. ročník),
- označení tématického okruhu (např. 1 = Číslo a proměnná v rámci oboru Matematika),
- pořadí výstupu v rámci daného okruhu (např. 08 = "Žák formuluje a řeší reálnou situaci pomocí rovnic a jejich soustav").

Příklad výsledného kódu: **M-9-1-08 "Žák formuluje a řeší reálnou situaci pomocí rovnic a jejich soustav"**

[6] Rámcové vzdělávací programy
[7] Rámcový vzdělávací program pro předškolní vzdělávání
[8] Rámcový vzdělávací program pro gymnázia
[9] Rámcový vzdělávací program středního odborného vzdělávání
[10] základní škola

Očekávané výstupy vymezují na konci 5. a 9. ročníku **závaznou úroveň** pro osnovy ŠVP[11], kterou je nutné dodržet. Výstupy na konci 3. ročníku vymezují jen **orientační úroveň**, která má dopomoci stanovit vhodnou vzdělávací cestu pro dosažení požadavků na konci 5. ročníku. [6]

**Učivo** určuje jednotlivá témata v rámci okruhu pro dané období. V rámci RVP ZV je vymezené učivo pro školy uvedeno pro vlastní rozpracování do jednotlivých ročníků, což je poté uvedeno v ŠVP školy. [6]

### 1.5.3 Náplň předmětu Matematika pro 2. stupeň

Další částí RVP jsou **Standardy**, které podrobně určují očekávané výstupy a indikátory postupu, které jsou určeny jako nápomocné k dosažení stanovených cílů. Zároveň jsou rozděleny do 4 tématických okruhů (např. Číslo a početní operace). [16] Hlavním smyslem této práce je digitalizace výuky tématu rovnic a s ním spojených témat jako je např. práce se zlomky, práce s procenty či obecná znalost oborů celých a racionálních čísel. Z pohledu standardů pro vybrané téma rovnic je třeba se zaměřit na následující standardy:

- **M-9-1-01** "Žák provádí početní operace v oboru celých a racionálních čísel; užívá ve výpočtech druhou mocninu a odmocninu."
- **M-9-1-02** "Žák zaokrouhluje a provádí odhady s danou přesností, účelně využívá kalkulátor."
- **M-9-1-03** "Žák modeluje a řeší situace s využitím dělitelnosti v oboru přirozených čísel."
- **M-9-1-04** "Žák užívá různé způsoby kvantitativního vyjádření vztahu celek – část (přirozeným číslem, poměrem, zlomkem, desetinným číslem, procentem)."
- **M-9-1-06** "Žák řeší aplikační úlohy na procenta (i pro případ, že procentová část je větší než celek)."
- **M-9-1-07** "Žák matematizuje jednoduché reálné situace s využitím proměnných; určí hodnotu výrazu, sčítá a násobí mnohočleny, provádí rozklad mnohočlenu na součin pomocí vzorců a vytýkáním"
- **M-9-1-08** "Žák formuluje a řeší reálnou situaci pomocí rovnic a jejich soustav."
- **M-9-1-09** "Žák analyzuje a řeší jednoduché problémy, modeluje konkrétní situace, v nichž využívá matematický aparát v oboru celých a racionálních čísel."
- **M-9-2-03** "Žák určuje vztah přímé anebo nepřímé úměrnosti."
- **M-9-2-05** "Žák matematizuje jednoduché reálné situace s využitím funkčních vztahů."[16]

[11] Školní vzdělávací program

Souhrnem potřebných standardů lze dojít k závěru, že absolvent 9. třídy ZŠ by podle určených výstupů měl být schopen řešit rovnice či soustavy rovnic obsahující celá a racionální čísla (zlomky), druhé mocniny a odmocniny a rozpoznat zda-li se jedná o příklad na přímou či nepřímou úměru. Zároveň by měl být schopen z textového zadání slovní úlohy najít řešení pomocí rovnice či soustavy rovnic. Navržené řešení by mělo všechny tyto typy úloh pokrýt.

# Kapitola 2

# Dostupná řešení

Tato kapitola se věnuje přehledu již existujících možností pro přípravu na přijímací zkoušky jako jsou různé online sbírky příkladů, testy z minulých let, portály s videotutoriály a další. Pro přehled pak kapitola obsahuje také tabulkový rozbor vlastností jednotlivých alternativ. V závěru jsou popsány poznatky získané z průzkumu a jejich využití pro návrh vlastního řešení. Na ten také navazuje rozbor relevantních typů příkladů, které by navrhované řešení mělo obsahovat.

## 2.1 Webové stránky

Již existující výukové prostředky online lze v rámci webových stránek v českém jazyce rozdělit do dvou podkategorií, zaměřené na přijímací zkoušky a obecný přehled matematiky na ZŠ i SŠ[1] (lze také definovat jako čistě sbírky příkladů dostupné online). Druhá část této podkapitoly pak obsahuje přehled zdrojů v angličtině.

### 2.1.1 V českém jazyce

#### Se zaměřením na přijímací zkoušky

Do této části patří především stránky, které mimo obecný přehled a tutoriály z matematiky nabízejí přímo i sekce zaměřené právě na přijímací zkoušky na SŠ z matematiky.

**Mathematicator**[2] a obecně tvorba Marka Váláška je postavena především na placených kurzech určených přímo pro přípravu na přijímací zkoušky na SŠ a VŠ[3], ale také obsahuje i jednotlivá témata na vyšších úrovních vzdělání. Na YouTube kanálu Marka Váláška lze poté najít odkazovaná videa na různá témata, která ale nejsou nijak výrazněji seřazena, a také záznamy přenosů živého řešení přijímacích zkoušek z posledních několika let.

Organizátor jednotných přijímacích zkoušek, **Cermat**[4], na své domovské webové stránce nabízí veřejně archiv zadání cvičných i ostrých zkoušek od roku 2015.

**zkousky-nanecisto**[5] je portál, který nabízí pouze placené kurzy pro přípravu na přijímací zkoušky a to jak distančně (online), tak i prezenčně na domluvených

[1] střední škola
[2] mathematicator.com
[3] vysoká škola
[4] cermat.cz
[5] zkousky-nanecisto.cz

místech s lektory. Jedná se tedy spíše o prodejní webový server.

**skolaposkole**[6] mimo obecný přehled učiva na ZŠ a SŠ také nabízí vlastní cvičné přijímací testy vytvořené na základě minulých let.

**prikladyzmatematiky**[7] podobně jako předchozí stránky nabízí vlastní krátké cvičné testy, ale také obecný přehled pro matematiku na ZŠ s poměrně velkým množstvím příkladů.

#### Obecné přehledy z matematiky

**Isibalo**[8] je vzdělávací portál, který mimo matematiku také obsahuje kurzy z biologie, chemie a fyziky. Jedná se tedy především o sbírku placených a neplacených kurzů na témata v jednotlivých předmětech, které jsou zde rozděleny podle tříd ve kterých se obecně vyučují a podle jejich vlastního obsahu. Každý kurz obsahuje několik sekcí se stupňující se obtížností a každá sekce je tvořena jedním či více vysvětlujícími videi z YouTube kanálu Isibalo, řešenými příklady a krátkým shrnujícím testem.

**Realisticky**[9] je archiv sbírek příkladů a učebnic matematiky. Mimo pokrytí všech témat ZŠ a SŠ a velkou sbírku příkladů také obsahuje občasné doplňující dokumenty k vybraným tématům.

**e-matematika**[10] je rovněž velkou sbírkou příkladů, ale obsahuje také rozdělení podle obtížnosti ve velké míře rozsahu. Je neplacenou stranou stránek *zkouskynanecisto.cz*.

**Matweb**[11] "je web zaměřený pro děti, žáky a studenty základních, středních a vysokých škol. Nabízí desítky polopaticky psaných článků o matematice pro všechny různé třídy, naleznete zde postupy, vysvětlení, obrázky, příklady a testy a vše online."[22] Témata na webu jsou zdánlivě uspořádána do kategorií a seřazena podle obtížnosti a návaznosti. Lze tak říci, že se prakticky jedná o neoficiální učebnici matematiky ve formě webové stránky.

Online je dostupné velké množství sbírek příkladů či starších učebnic. Jmenovitě tedy materiály zdarma od nakladatelství **Fraus**[12], učebnice **Matematické minutovky** vytvořené na míru RVP ZV pro matematiku a také **Matematika v kostce** pro střední školy, jejíž několik prvních kapitol je relevantních pro účely přijímacích zkoušek.

### 2.1.2 V anglickém jazyce

Zaměřením práce je specificky příprava na přijímací zkoušky v České republice, proto je výčet webových stránek v anglickém jazyce zaměřen na výuku matematiky na úrovni základní školy. Tato část tedy neobsahuje rozdělení podobné části věnované platformám v českém jazyce.

**Khan Academy**[13] je výukový web, který by se dal zařadit i do předchozí části, jelikož nabízí i verzi v češtině. Ta však nutně neobsahuje veškerý obsah, který

[6] skolaposkole.cz
[7] prikladyzmatematiky.cz
[8] isibalo.com
[9] realisticky.cz
[10] e-matematika.cz
[11] matweb.cz
[12] nakladatelství ucebnice.fraus.cz
[13] cs.khanacademy.org

původní (anglická) verze nabízí. Největším rozdílem jsou pak partnerské lekce od různých amerických institucí. Obě verze poskytují nejen výuku matematiky od lekcí pro děti v předškolním věku až po vysokoškolská témata, ale i lekce z přírodních věd, informatiky, ekonomiky a dalších. Bonusem české verze je poté i část kapitoly "Opáčko matematiky", která se věnuje JPZ[14]. Obsah je se zanedbatelnými výjimkami zdarma. S tím se pojí rozsáhlá nabídka výukových videí v anglickém jazyce na YouTube kanále Khan Academy, kde vybraná videa nabízí i titulky v češtině.

**Math.com**[15] se velice podobá výše zmíněnému webu Realisticky, nicméně není zdrojem učebnic a sbírek, ale spíše souhrnem krátkých taháků pro řešení matematických příkladů v rozsahu od ZŠ po VŠ (vystihující záložka "Homework Help"). Dále také nabízí nástroje po výpočet obsahu, objemu, průměru, kvadratických rovnic, ale i také pravděpodobnosti či určení prvočísla. Veškerý obsah je zdarma, ale neexistuje verze v češtině.

**Math Goodies**[16] nabízí kvalitně zpracované lekce z matematiky včetně velkého počtu ilustrativních obrázků. Lekce jsou ale určeny pro 1. až 8. třídu, takže podstatná část témat (především rovnice) potřebná ke splnění JPZ chybí nebo není kompletní. K procvičování předpokladů pro rovnice a podobných matematických problémů poskytuje vlastní pracovní listy s příklady a krátké učebnice zdarma. Nabízí také placený obsah, který se ale hlavně sestává z výuky a řešení vlastních příkladů lektory. Pouze verze v angličtině.

**Cliff Notes**[17] je platforma založená na *"study guides"* (v překladu studijní průvodce). To jsou krátké lekce ve formě poznámek vytvořené učiteli. Stránka nabízí jak lekce z matematiky, tak i například zběžné rozbory literárních děl. Rozsah lekcí je opravdu velký, takže lze najít lekce jak přímo na rovnice, tak i na jejich prerekvizity a jejich uplatnění v pokročilejších příkladech. Vše je až na několik bonusových lekcí zdarma, zároveň ale lze také lekce koupit ve formě fyzické učebnice. Neexistuje český překlad.

## 2.2 Mobilní aplikace

Možnosti výuky pomocí mobilních aplikací jsou poněkud omezené co se rozmanitosti týče. Existující aplikace lze rozdělit do dvou podkategorií:

- výukové aplikace od **eductify**[18] - vcelku komplexní aplikace obsahující jak teoretický výklad, tak i tisíce příkladů, které jsou případně rozřazeny do kategorií podle složitosti a komplexity (podobnou nabídku má také aplikace/stránka **Brilliant**[19], ta však především klade důraz na vizualizaci obecného řešení problémů na úkor jejich složitosti);

- výpočetní aplikace, které z fotografií pořízených kamerou telefonu dokáží nabídnout postup a výsledek úlohy, např. **Photomath**[20],

[14] Jednotná přijímací zkouška
[15] math.com
[16] mathgoodies.com
[17] cliffnotes.com
[18] eductify.com
[19] brilliant.org
[20] photomath.com

**Microsoft Math Solver**[21], **Mathway**[22], **Brainly**[23] a další.

## 2.3 Shrnutí

V tabulce 2.1 jsou jednotlivé weby a aplikace ohodnoceny v různých kategoriích, které slouží jako obecný přehled toho, co daná řešení zkoumané problematiky nabízejí. Kategorie Vzorové testy posuzuje, zdali jsou k dispozici testy ve formátu stejném či podobném testům reálné přijímací zkoušky. Dále kategorie Velký počet úloh hodnotí, zdali dané řešení nabízí úlohy k procvičování mimo úlohy připojené k různým lekcím jako vysvětlovací. Kategorie Téma rovnic značí, zdali stránka nebo aplikace obsahuje teorie a příklady na téma rovnic. Kategorie Řešení, Postup a Teorie souvisí s kategorií předchozí, tzn. nabízí-li web/aplikace mimo výčet příkladů také jejich řešení, postup (ať už pouze ve formě kroků či i se slovním vysvětlením) a případně teoretický podklad, je-li nutný. Kategorie Tutoriály, videa pouze hodnotí přítomnost lekcí ve formě videí a poslední kategorie indikuje, zdali jsou příklady obtížnostně ohodnoceny. Další informace k hodnocení se nachází v legendě tabulky - viz. titulek obrázku.

Shrnutím obsahu tabulky lze dojít k následujícím závěrům. V českém jazyce existuje mnoho zdrojů poskytujících teoretický základ a mnoho obecných sbírek příkladů a značné množství je i volně přístupné. Videotutoriály naopak výrazně chybí u většiny zdrojů, to je mimo jiné ale také způsobeno malou velikostí případného publika pro lekce v českém jazyce. Zároveň ty lekce, které jsou dostupné, jsou více než dostatečné pro samostudium (viz Isibalo (2.1.1) a další). Největším současným problémem je ve výsledku minimální dostupnost procvičování testů přímo zaměřených na JPZ. Logicky poté tento problém nelze aplikovat na vlastnosti zdrojů v anglickém jazyce. Ty ale poskytují především výborné zdroje pro teorii "ve zkratce"(viz Cliff Notes (2.1.2)) či vhodnou platformu pro komplexnější výuku všech možných témat (viz Khan Academy (2.1.2)) včetně videotutoriálů či širokého spektra příkladů.

## 2.4 Poznatky z průzkumu

Hlavním cílem průzkumu již existujících platforem bylo získání inspirací pro tvorbu vlastního řešení. V rámci webových stránek se jedná o zdroj příkladů a úloh pro vlastní užití a vlastně i o teoretický přehled jak dané příklady rozdělit pro lepší přehlednost a přístupnost studentům, tzn. obtížnost, návaznost či teoretický podklad. Z přehledu je však patrné, že více než polovina nenabízí vzorové testy ve formátu reálných testů. Zde je ale nutné podotknout, že online archiv testů organizace **Cermat** je dostačující k vytvoření jakési formy pro tvorbu vlastních testů pomocí příkladů z archivu. To je dáno především tím, že formát testů a příkladů v nich je z většiny neměnný (více v sekci 2.5). Jako zdroj teorie se nabízí webová stránka **Matweb**, kde jsou jednotlivá témata rozdělena do středně dlouhých lekcí s velkým množstvím podkladů pro studium. Pro stručnější formu teorie lze využít jednotlivých *"notes"* z webové stránky **Cliff Notes**.

[21] math.microsoft.com
[22] mathway.com
[23] brainly.com

| | Český jazyk | Placený obsah | Obsah zdarna | Vzorové testy | Velký počet úloh | Téma rovnic | Řešení | Postup | Teorie | Tutoriály, videa | Kategorie podle obtížnosti |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Webové stránky | | | | | | | | | | | |
| Mathematicator | ✓ | ✓ | ✓ | ✓* | ✓* | ✓ | ✓** | ✓** | ✓* | ✓* | ✓* |
| Cermat (web) | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| zkousky-nanecisto.cz | ✓ | ✓ | ✓ | ✓* | ✓* | ✓* | ✓** | ✓** | ✓* | ✓** | ✓* |
| skolaposkole.cz | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ |
| prikladyzmatematiky.cz | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Isibalo | ✓ | ✓ | ✓ | ✗ | ✓** | ✓ | ✓** | ✓** | ✓** | ✓** | ✓ |
| Realisticky.cz | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ |
| e-matematika | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ |
| Matweb | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ |
| Khan Academy | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Math.com | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ |
| Math Goodies | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓** | ✗ | ✓ |
| Cliff Notes | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓** | ✗ | ✓ |
| Mobilní aplikace | | | | | | | | | | | |
| eductify | ✓ | ✓ | ✗ | ✗*** | ✓* | ✓* | ✓* | ✓* | ✓* | ✓* | ✓* |
| Photomath | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓** | ✗ | ✓** | ✗ |
| Brilliant | ✓ | ✓ | ✓ | ✗ | ✓** | ✓ | ✓ | ✓* | ✓** | ✓* | ✓ |
| Mathway | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓* | ✗ | ✗ | ✗ |

**Tabulka 2.1:** Přehled webů (* - pouze v placené verzi, ** - rozdílně v placené versus neplacené verzi, *** - obsahuje náhodné testy místo testů specificky určených jako zkušební přijímací)

V rámci mobilních aplikací se poté jedná především o inspiraci grafickým vzhledem a rozvrhnutím uživatelského rozhraní. Kromě zmíněných aplikací je nutné poznamenat i jazykovou aplikaci **Duolingo**[24], která podobně jako aplikace od společnosti **eductify** nabízí předem určený průchod učivem ve vhodném pořadí (tzn. rozdělení lekcí do podoby stromu, kde pro další postup do několika lekcí - větví, je nutné nejdříve splnit lekce předchozí).

[24] Duolingo

## 2.5 Analýza obsahu testů

V sekci Poznatky z průzkumu (2.4) bylo zmíněno vytváření vlastních cvičných testů v rámci přípravy na přijímací zkoušky (více v 1.2 Jednotné přijímací zkoušky v ČR). Archiv testů společnosti Cermat[25] obsahuje testy z let 2015 až 2023 a je z něj možné vytvořit přibližný seznam testovaných témat a typů příkladů, kterými jsou tato témata u přijímacích zkoušek zastoupena. Dále je také možné určit obtížnost příkladů, které se v testech nachází. Se zaměřením na téma rovnic jsme vytvořili sbírku příkladů zadaných přímo jako rovnice, které se v archivovaných testech vyskytly. Sbírka nám poskytla přehled o různých vlastnostech rovnic v testech, ať už se jedná o jejich složitost, typy čísel v rovnicích či kolikrát se neznámá v rovnicích objevuje (více v závěru sbírky). Sbírka je součástí příloh.

Z analýzy ale vyplývá také fakt, že obtížnost starších testů oproti těm současným se výrazně liší, a toto bude potřeba zohlednit při implementaci příkladů do systému. Jako příklad pro porovnání těchto rozdílů jsou uvedeny následující 2 dvojice úloh:

**Příklady na téma rovnic:**

$$-\frac{2}{3} * \frac{x}{2} = \frac{5}{12} \tag{2.1}$$

*(JPZ 2017 - varianta A)*

$$\frac{y+10}{15} + \frac{2y}{5} = 1 - \frac{5-y}{3} \tag{2.2}$$

*(JPZ 2023 - varianta A)*

Uvedené příklady jsou rovnice se zlomky, kde ve jmenovateli jsou pouze celá čísla, která mají společný násobek v rozsahu přiměřeném výuce na základních školách. V příkladech výše lze ihned spatřit rozdíl ve složitosti zadání. Starší varianta obsahuje jen jeden výskyt neznámé a tato neznámá se vyskytuje pouze na levé straně rovnice. Varianta z roku 2023 naopak nejen že obsahuje 3 výskyty neznámé, ale tuto neznámou lze najít na levé i pravé straně rovnice.

**Příklady na téma slovní úlohy o procentech:**

"Celkem 70% z 520 důchodců používá kartu do bankomatu.
Kolik důchodců nepoužívá kartu do bankomatu?"

*(JPZ 2017 - varianta A)*

"V roce 2020 firma vyrobila 250 výrobků. Jak v roce 2021, tak v roce 2022 vyrobila firma vždy o 20% výrobků více než v předchozím roce.
Kolik výrobků vyrobila firma v roce 2022?"

*(JPZ 2023 - varianta A)*

Druhé porovnání obsahuje dva příklady na počítání s procenty zadané ve formě slovní úlohy. A stejně jako v předchozí ukázce, řešení starší verze je podstatně lehčí, tedy stačí vypočítat pouze zbývající část celku, zatímco novější verze požaduje počítání více kroků změn v uvedených údajích.

[25] archiv testů Cermat

### 2.5.1 Typy testových příkladů

Tato podkapitola obsahuje seznam témat relevantních pro navrhovaný systém spolu s typy příkladů, které se v testech mohou nacházet, spolu s tabulkou shrnující nejdůležitější údaje (viz 2.2). Seznam je vytvořen především pro účely vytyčení limitů při generování příkladů v systému.

- **Celá čísla**

  Příklady s operacemi nad celými čísly se nacházejí pouze ve starších testech, nicméně jsou základním předpokladem pro všechna následující témata.

- **Zlomky a desetinná čísla**

  Příklady se zlomky a s desetinnými čísly se bez výjimky nachází ve všech archivovaných testech a to většinou v sekcích o 3 příkladech spolu s občasným příkladem navíc, jehož zadání je slovní. Kromě jednodušších příkladů se zlomky se také v testech mohou nacházet složené zlomky.

- **Mocniny a odmocniny**

  Toto téma je v testech zastoupeno v sekcích o 2 příkladech a to většinou v kombinaci se zlomky či desetinnými čísly. Důležité je zmínit, že ačkoliv je toto téma ve většině případů omezeno na maximálně druhé mocniny deseti a příslušná desetinná čísla, tak od roku 2022 se na poslední straně testu nachází také příslušné hodnoty pro druhou mocninu čísel 11 až 20.

- **Úprava výrazů**

  Toto téma zahrnuje převod zadání obsahující neznámou do jednodušší formy. Typickým příkladem může být například násobení výrazů v závorkách či jejich umocňování. Testy vždy obsahují sekci o 2 příkladech na toto téma. Podobně jako u mocnin se od roku 2022 na poslední straně nachází 3 základní vzorce pro rozklad na součin.

- **Rovnice o jedné neznámé**

  Rovnice jsou v přijímacích testech zastoupeny sekcí o 2 příkladech. První příklad většinou obsahuje pouze desetinná čísla, zatímco druhý vždy obsahuje zlomky. Důležité je zmínit, že ve většině případů jsou jedinými operacemi s neznámými během řešení dosavadních archivovaných příkladů sčítání a odčítání. Až na vzácné výjimky nenarazí student na umocněnou neznámou (viz sbírka příkladů v příloze).

- **Slovní úlohy**

  Slovní úlohy se v přijímacích testech vyskytují ve dvou typech, prvním z nich jsou slovní úlohy na **počítání s procenty**. Ty se ve všech testech vyskytují buď jako sekce o 3 různých úlohách nebo jako jedna větší úloha, ve které má žák za úkol vypočítat 3 různé úkoly v rámci jednoho zadání. Žáci se v testech mohou setkat jak s úlohami s jednou procentuální změnou na zadaných hodnotách, tak i se složitějšími příklady se dvěma takovými změnami (např. "produkt byl dva roky po sobě zlevněn o X procent").

Druhým typem jsou slovní úlohy pro jejichž řešení je nejvhodnější použít rovnice. Tento typ lze nadále rozdělit na slovní úlohy **o směsích**, slovní úlohy **o pohybu**, slovní úlohy **o společné práci** a nakonec slovní úlohy **o rozdělení na části**. Slovní úlohy o směsích typicky obsahují výčet hodnot (cena, počet, váha) pro jednotlivé součásti směsi a úkolem žáka je zjistit vlastnosti směsi obsahující více takových součástí. Slovní úlohy o pohybu jsou o pohybu aktérů úlohy proti sobě, za sebou a se společným startem. Aktéři se od sebe liší rychlostí a/nebo délkou dráhy a úkolem žáků je zjistit rozdíly mezi nimi po určité době, vzdálenosti apod. Požadovanými výsledky slovních úloh o společné práci (také sem spadají slovní úlohy o spotřebě například zásob) jsou typicky vlastnosti práce různých aktérů pracují-li společně nebo pracují-li naopak odděleně či v rozdílném počtu nebo časovém úseku (závisí na zadání úlohy). Posledním typem jsou nejčastěji vyskytující se slovní úlohy o částech, které lze jednoduše řešit vhodným vybrání neznámé a přepsáním textového zadání do podoby rovnice. Celý tento druhý typ slovních úloh se v testech vyskytuje ve 2-3 příkladech.

| Téma | Počet | Typy |
|---|---|---|
| Celá čísla | 0-1 | jednoduchý příklad s operacemi +, -, *, : |
| Zlomky a desetinná čísla | 3-4 | jednoduchý příklad s operacemi +, -, *, : |
| | | složený zlomek |
| Mocniny a odmocniny | 2 | druhá (od)mocnina se vztahuje pouze na jedno číslo |
| | | více čísel pod stejnou druhou odmocninou |
| Úprava výrazů | 2 | úprava podle vzorce (rozklad na součin) |
| Rovnice o jedné neznámé | 2 | příklad omezený na celá a desetínná čísla |
| | | příklad s více zlomky, neznámá pouze v čitateli |
| Slovní úlohy o procentech | 3 | slovní úloha na změnu hodnot o X% |
| | | slovní úloha s vícenásobnými změnami hodnot |
| Slovní úlohy o rovnicích | 2-3 | slovní úloha o počítání se směsmi |
| | | slovní úloha o pohybu (za sebou, proti sobě, spolu) |
| | | slovní úloha o společné práci (nebo spotřebě) |
| | | slovní úloha o rozdělení na části |

**Tabulka 2.2:** Shrnutí typů příkladů

# Kapitola 3

# Analýza

Tato kapitola představí požadavky na obsah a funkcionalitu výukového systému, které byly posbírány na základě analýzy z předchozích kapitol. Obsahuje slovní popis aplikace, dále list požadavků na funkcionalitu aplikace, který je poté převeden do grafické podoby ve formě diagramu případů užití.

## 3.1 Záměr

Záměrem projektu je vytvořit aplikaci nabízející jak procvičování testů vytvořených pomocí dostupných testů z minulých let, tak i výuku a procvičování úloh pro přijímací testy typických či hromadné generování úloh podle vybraného typu. Aplikace nabídne žákům doporučený průchod výukou, především rovnic, a to od jednoduchých rovnic s jednou neznámou až po slovní úlohy či soustavy rovnic. Dodrženy by měly být požadavky pro výuku stanovené v RVP ZV MŠMT pro žáky 2. stupně se zaměřením především na úroveň znalostí žáků 9. třídy ZŠ (popř. odpovídající třídě osmi- a šestiletých gymnázií) (viz 1.5.3). Typ a náročnost příkladů v testech aplikace by měly odpovídat nebo se co nejvíce přiblížit obsahu skutečných testů (viz 2.5). Výsledný produkt by měl být poté otestován ve skutečném třídním prostředí během vyučovací hodiny matematiky, popř. školním kroužku pro přípravu na JPZ z matematiky.

## 3.2 Požadavky

Následující výčet požadavků je rozdělen do dvou kategorií: funkční požadavky - souhrn funkcionalit aplikace a nefunkční požadavky - vlastnosti aplikace z pohledu kvality, spolehlivosti atd. Počítáme se dvěma typy uživatelů - běžný uživatel a registrovaný uživatel.

### 3.2.1 Funkční požadavky

1. **Správa uživatelského účtu**

   Systém umožní každému uživateli vytvoření vlastního účtu.

2. **Správa dat uživatele**

Systém umožní každému uživateli resetovat svá vlastní data ukládaná aplikací (např. resetování postupu aplikací).

3. **Přeskočení sekce výuky**

   Systém umožní každému uživateli otevřít si uzamčenou sekci výuky splněním kontrolního testu.

4. **Generování příkladů**

   Systém umožní každému uživateli generovat matematické příklady podle zvolených možností (typ, složitost) a ve zvoleném počtu.

5. **Uložení vygenerovaných příkladů**

   Systém umožní registrovanému uživateli uložit dosavadní stav řešení příkladů pro příští otevření.

6. **Vyhodnocení řešených příkladů**

   Systém vyhodnotí řešení každého uživatele.

7. **Testování nanečisto**

   Systém umožní každému uživateli generovat vzorové testy ve formátu reálných testů.

8. **Vyhodnocení vzorového testu**

   Systém vyhodnotí uživatelem vyplněný test.

9. **Odkazování do výuky**

   Systém poskytne všem uživatelům možnost u každého příkladu přejít do relevantní sekce výuky.

10. **Náhled historie**

    Systém poskytne všem uživatelům přehled plnění příkladů/testů či průchodu sekcemi výuky.

11. **Obnovení zapomenutého hesla**

    V případě ztráty stávajícího hesla systém umožní registrovaným uživatelům nastavení si hesla nového přes email použitý při registraci.

12. **Zálohování dat**

    Pro případy ztráty lokálně uložených dat budou tato data také ukládány v cloudové databázi jako záloha.

### 3.2.2 Nefunkční požadavky

1. **Multiplatformní nasazení**

   Aplikace bude primárně nasazena na nejpoužívanější operační systémy přenosných zařízení (Android, iOS). Dále také bude vytvořena záložní webová aplikace.

2. **Uživatelské prostředí v češtině**

   S ohledem na cílovou skupinu aplikace bude uživatelské prostředí zcela v českém jazyce.

3. **Více návrhů uživatelského prostředí**

   Všichni uživatelé budou mít k dispozici výběr z více návrhů UI[1], např. světlý a tmavý motiv.

4. **Povinná změna hesla**

   S ohledem na bezpečnost bude mít registrovaný uživatel povinnost změnit své heslo po určité době (např. 6 měsících).

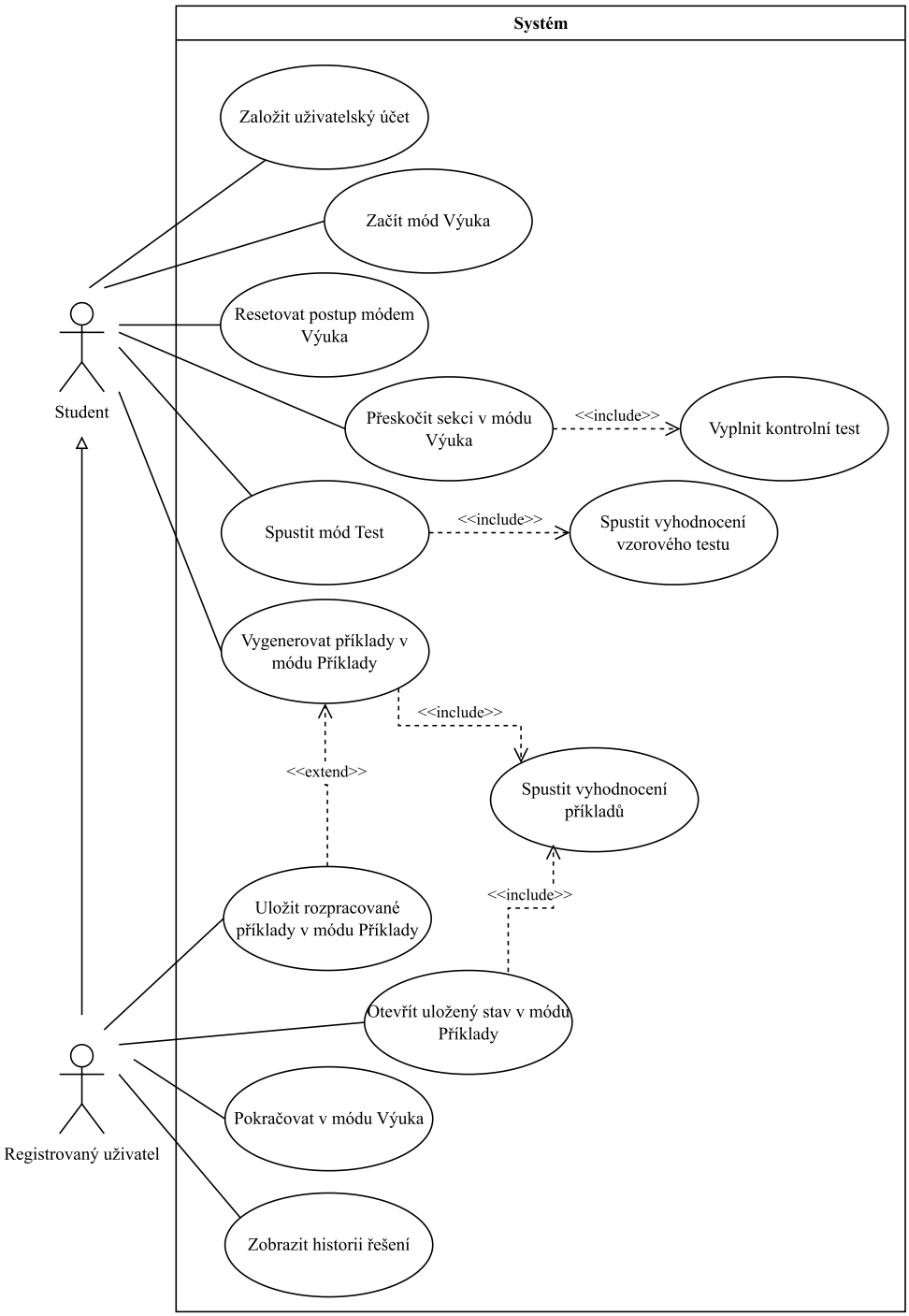
5. **Zálohování a bezpečnost dat**

   Databáze uživatelů a databáze záložních dat budou dostatečně zabezpečeny proti útokům zvenčí (např. SQL injection).

[1] user interface (překl. uživatelské rozhraní)

## 3.3 Případy užití

Na obrázku 3.1 jsou v návaznosti na systémové požadavky zobrazeny ve formě UML[2] diagramu případy užití navrhovaného řešení. Aktéry, typy uživatelů, kteří budou systém využívat jsou běžný uživatel (student) a student, který má v aplikaci vytvořený uživatelský účet (tzn. je registrován) a má tak přístup k rozšířeným funkcím, především tedy k ukládání postupu aplikací.




**Obrázek 3.1:** Případy užití

[2]Unified Modeling Language

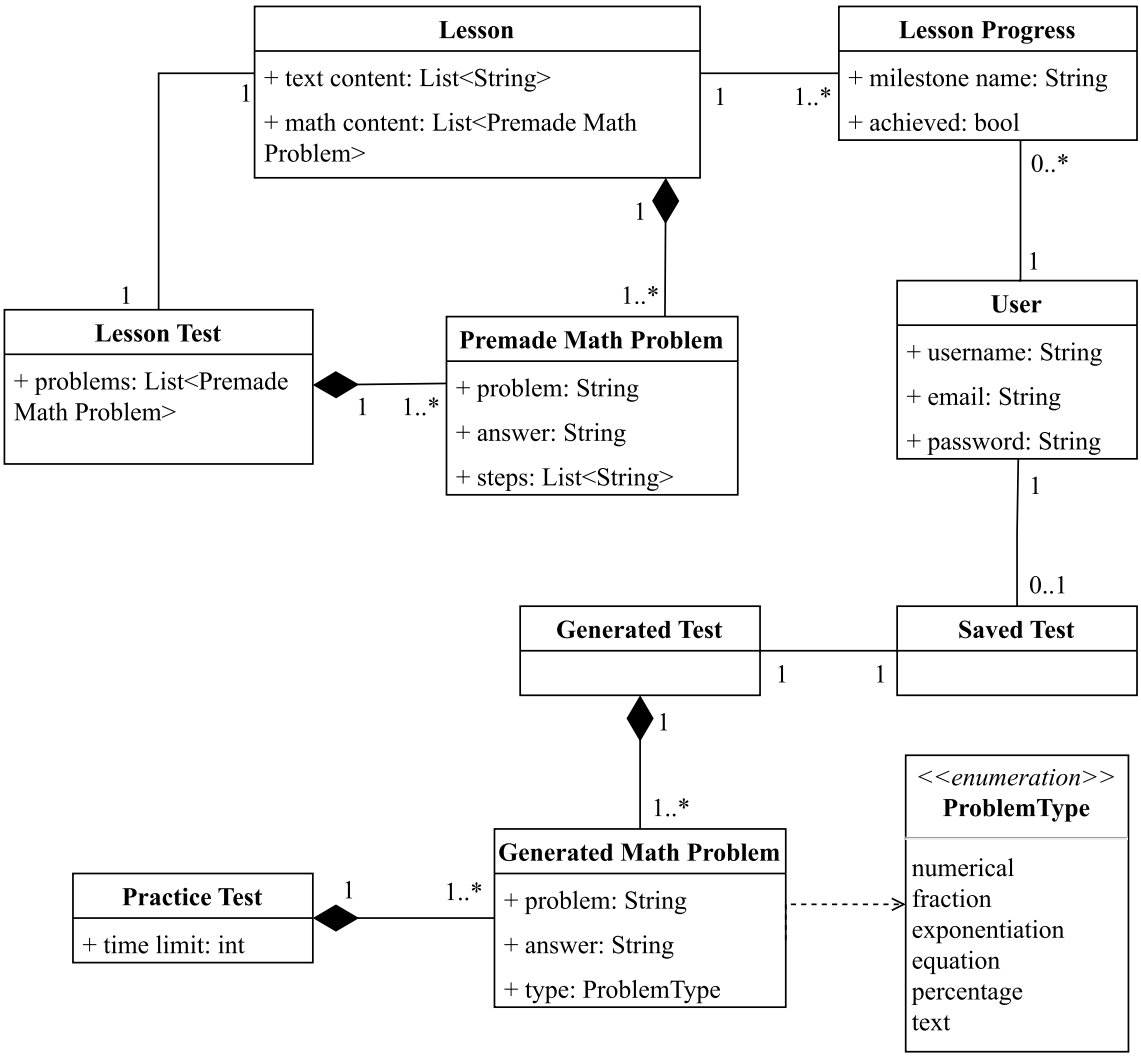
# Kapitola 4

# Návrh řešení

Tato kapitola se zabývá představením návrhu výukové aplikace, jejíž vlastnosti a obsah vycházejí z předchozí kapitoly. Obsahuje návrh struktury objektů v aplikaci, návrh platformy a návrh technologií, které budou k vývoji na vybranou platformu či platformy použity. Podle výběrů z příslušných podkapitol byly poté vytvořeny návrhy architektury aplikace spolu s diagramem nasazení. Závěr kapitoly obsahuje diagram hierarchie obrazovek a s ním související grafické návrhy vzhledu uživatelského rozhraní.

## 4.1 Doménový model

Diagram na obrázku 4.1 zobrazuje konceptuální návrh struktury systému. Každý uživatel si drží u svého účtu připojené informace o postupu výukou a vygenerované příklady, které si uložil. Příklady v systému jsou rozděleny na ručně vytvořené a vygenerované. Existují také celkem tři druhy testů: test pro přeskočení lekce obsahující několik ručně vytvořených příkladů, "test"(v tomto případě spíše sada příkladů), který si uživatel podle zvolených parametrů nechá systémem vygenerovat a konečně cvičný test s předem určeným časovým limitem.




**Obrázek 4.1:** Doménový model

## 4.2 Forma aplikace

V dnešní době mimo počítačovou učebnu mají školy často k dispozici také různá dotyková zařízení, ať už tablety s operačním systémem Android, tak i iPady s operačním systémem iPadOS. Zároveň většina žáků v adolescentním věku tráví průměrně přes 4 hodiny denně na mobilních zařízení. [15] Nabízejí se tak 3 cesty realizace řešení, webová aplikace dostupná ze všech zařízení, mobilní aplikace pro zařízení Android nebo mobilní aplikace pro zařízení od firmy Apple. Případnou volbou je poté i kombinace výše uvedených možností.

Začněme s mobilními aplikacemi (pro mobilní telefony a tablety obou typů operačního systému). Hlavní výhodou takového řešení by bylo využití již existujícího aktivního používání mobilních zařízení žáky. Dále také plná funkčnost aplikace bez přístupu k internetu. Příkladovou situací by poté bylo například zběžné řešení příkladů ráno během dojíždění do školy. Toto řešení je spíše vhodné pro samostudium než přímo pro školní prostředí, nicméně je možné toto obejít například sdílením jednoho zařízení celou třídou a společným řešením.

Naopak webová aplikace by byla dostupná pro všechna zařízení přes webový prohlížeč a aplikace by tak byla vhodná jak pro učitelem řízenou výuku ve škole, tak i pro samostudium doma. S tím se pojí také fakt, že v každé škole je k dispozici alespoň počítačová učebna v případě, že škola nedisponuje tablety nebo podobnými zařízeními. Další výhodou je možnost využití aplikace jako předváděcí za pomoci promítacího zařízení v učebně. Nicméně, s počítači se pojí omezená mobilita v porovnání s mobilními zařízení a forma aplikace v prohlížeči omezí možnosti užívání bez přístupu k internetu.

Finální výběr záleží nakonec na dostupných zařízeních na místě testování systému. Na základní škole, kde bude po dohodě s ředitelem školy a individuálními učiteli probíhat testování, jsou dostupné buď stolní počítače nebo tablety od společnosti Apple. Primárně jsme tedy zvolili vývoj pro operační systém iOS, ale zároveň jako záložní verze bude vyvíjena i aplikace webová. V případě nemožnosti nasazení jako iOS aplikace či jiných komplikací, tak budou moci žáci otestovat aplikaci v prohlížeči a to jak přes tablety, tak i stolní počítače.

## 4.3 Nabídka technologií

Aplikace je zamýšlena zejména pro uživatele mobilních telefonů a tabletů, obecně dotykových zařízení, která jsou přenosná. Pro vývoj se tedy nabízí několik možných technologií, které jsou pro taková zařízení určená. Tato podkapitola nejdříve několik technologií představí a vybrané technologie pak budou mezi sebou porovnány v následující podkapitole.

Platform pro vývoj mobilních aplikací na trhu je v dnešní době více než dost. Je proto občas těžké pro nové programátory najít vhodnou technologii pro svůj projekt. Může se jednat o čistě programovací jazyky určené pro vývoj aplikací pro Android a iOS (Kotlin a Swift respektive) nebo o různé frameworky, tzn. již připravené balíčky funkcionalit (např. Flutter, React Native a další). Cílem této podkapitoly je představit současné nejznámější technologie, které by mohly být využity pro účely tohoto projektu.

- **Kotlin** a **Java** jsou programovací jazyky nejvíce používané pro vývoj mobilních aplikací pro operační systém Android. Java byla oficiálním jazykem pro

Android, ale od představení Kotlinu, který nabízí modernější prvky a je zároveň s Javou zaměnitelný, jím byla nahrazena.[8] Kotlin lze také díky technologii Kotlin Multiplatform použít pro vývoj aplikací pro Android i iOS.[9]

- **Swift** je podobně jako Kotlin oficiální jazyk pro vývoj pro určitý operační systém a to iOS (spolu s ostatními operačními systémy společnosti Apple).[27] Aplikace pro Android lze pomocí Swiftu vytvářet, ale je nutné podotknout, že podmínkou je využití knihoven jako např. SCADE.[10]

- **React Native** je framework od společnosti Meta (dříve Facebook) založený na frameworku React, který využívá programovací jazyk JavaScript. Hlavní funkcionalitou React Native je možnost kód psaný jedním jazykem nasadit na více platforem (Android, iOS, webové aplikace a další). Mobilní aplikace je možné vytvořit i pomocí původního frameworku React, ale ten je především určen pro vývoj na webu.[12]

- **Flutter** je taktéž framework, kterým lze pomocí jednoho jazyka vytvořit aplikaci pro více platforem. Framework je udržován společností Google a jeho základním programovacím jazykem je Dart, který je podobný Javě či C#. Stavebními prvky Flutteru je rozdělení aplikace do widgetů (komponent), které tvoří nejen vizuální ale i funkční prvky. Patří mezi ně např. funkční widgety jako Row (překl. řádka) nebo Container (překl. kontejner) a primárně vizuální widgety jako Checkbox nebo Slider (překl. posuvník).[19]

- **Xamarin** je dalším frameworkem, který je tentokrát udržovaný společností Microsoft a jeho základními programovacími jazyky jsou C# a F#. Xamarin narozdíl od Flutteru pracuje s nativními UI prvky a API (Flutter naopak řídí vše pomocí vlastních widgetů). Tak jako ostatní lze nasadit Xamarin kód na více platforem.[13]

- **Apache Cordova** je platforma, která umožňuje programátorům pomocí webových komponent a jazyků JavaScript, HTML a CSS vytvářet multiplatformní aplikace. Framework zprostředkovává sadu API pro přístup k nativním funkcím zařízení. Cordova umožňuje nasazení aplikace zabalením vytvořené webové aplikace do nativního kontejneru pro různé operační systémy.[3]

- **Adobe PhoneGap** je variantou Apache Cordova od společnosti Adobe. Nabízí několik funkcionalit navíc jako například vyvíjení v cloudu. PhoneGap je také především podporován Adobe[3], zatímco podpora Apache Cordova je spíše komunitně založená (software je open-source).

- **NativeScript** je framework založený na integraci JavaScriptových frameworků Angular a Vue do mobilního vývoje. NativeScript aplikace lze psát jazyky JavaScript, ale i TypeScript. Podobně jako výše uvedené frameworky poskytuje přístup k nativním API a UI komponentám. Používaný je především pro aplikace, které vyžadují vyšší výkonost.[5]

Tabulka 4.1 slouží jako zběžné porovnání technologií na seznamu výše. Obsahuje informace o jazycích, které technologie používají, dále také jestli daná technologie nabízí nasazení na více platforem. V posledním sloupci je poté pro každou technologii uvedeno pár zajímavých informací.

| | Jazyk(y) | Multi-platformní | Důležité detaily |
|---|---|---|---|
| Kotlin/Java | Kotlin/Java, XML (UI) | ano (Kotlin Multiplatform) | oficiální jazyky pro Android, Kotlin zaměnitelný s Javou |
| Swift | Swift | ano (pouze pomocí knihoven) | oficiální jazyk pro iOS, limitované využití pro ostatní platformy |
| React Native | JavaScript (TypeScript), JSX (UI) | ano | vhodný přechod z frameworku React, závislost na komunitě pro dokumentaci a rozšíření |
| Flutter | Dart | ano | oficiální dokumentace a rozsáhlá knihovna v základu, limitované využití komunitou (tzn. např. méně rozšíření) |
| Xamarin | C#/F#, XAML (UI) | ano | propojení s ekosystémem .NET frameworku, omezený přístup k rozšířením |
| Apache Cordova | JavaScript, HTML + CSS (UI) | ano | vhodné pro zkušené webové vývojáře, způsob tvorby aplikace (zabalení webové aplikace do komponenty pro mobilní zařízení) může způsobovat problémy s kompatibilitou a výkonem aplikace |
| Adobe PhoneGap | | | |
| NativeScript | JavaScript (TypeScript), XML + CSS (UI) | ano | vhodný přechod z frameworků Angular a Vue.js, podobně jako Flutter menší komunitní podpora |

**Tabulka 4.1:** Přehled technologií

## 4.4 Výběr technologie

Aplikace je zamýšlena pro Android i iOS tudíž nejvhodnější cesta vývoje by byla přes jeden z výše uvedených frameworků, které jsou koncipované přímo pro multiplatformní vývoj. Vzhledem k předchozím zkušenostem s Javou a JavaScriptem pak lze ze seznamu ideálních technologií vyřadit Xamarin. Z důvodu nezkušenosti s frameworky Angular a Vue.js a i frameworky v JavaScriptu obecně vyřazujeme také NativeScript. Ve výsledku je tak nutné provést výběr podle osobních preferencí, především k používaným jazyků a funkcionalitám frameworku. Při přihlédnutí k faktu, že jedním z důvodů zaměření se na vývoj pro mobilní zařízení bylo i oddálení se od webových technologií, jsme se rozhodli vyřadit také frameworky, které využívají HTML a CSS (tzn. Cordova a PhoneGap). Tomu podobné jsou i XML (používané pro vývoj v Kotlinu) a JSX (používané pro vývoj v React Native).

Mimo tuto skupinu se pak drží Flutter, který, jak bylo výše zmíněno, celou svou strukturu udržuje pomocí widgetů v jazyce Dart a nevyužívá tak k budování vzhledu aplikace značkovací jazyky jako XML a další. V porovnání s ostatními frameworky je také Flutter vhodný pro začátečníky, především díky jednoduchosti jazyka Dart, který lze použít jak na vývoj frontendu tak i backendu. Dále pro mnoho začátečníků

je využívání Flutter widgetů (zabalené nativní komponenty) jednodušší a intuitivnější než nutnost znalosti API a dalších detailů jednotlivých platforem. V porovnání s React Native také Flutter už v základu přichází s rozsáhlou knihovnou widgetů, zatímco React Native poskytuje základní funkcionalitu, ale další rozšíření nechává na open-source komunitě, pro začátečníka může být tak jednodušší mít vše na jednom místě než využívat mnoho knihoven pro svůj projekt.[2] V neposlední řadě funkce *"Hot Reload"* (rychlá aktualizace změn v kódu bez ztráty stavu aplikace) umožňuje efektivnější experimentování s projektem a je tak ideální pro učení se nové technologie. Většina ostatních frameworků tuto funkci také nabízí (např. *"Fast Refresh"* ve frameworku React Native), ale rychlost a spolehlivost se mohou lišit.[23][21]

V porovnání s Kotlin Multiplatform pak Flutter vyniká především v úrovni dokumentace, což je pro začátečníky s mobilním vývojem důležité. Dále je také Kotlin Multiplatform spíše vhodnější pro ty, kteří chtějí svou již existující nativní aplikaci nasadit i na další platformy anebo již mají předchozí zkušenosti s nativním vývojem. Flutter se navíc více hodí pro menší projekty, jejichž hlavním cílem je poskytnutí jednoduché funkcionality a sběrem zpětné vazby pro další vývoj.[24]

Z důvodu předchozích nezkušeností s vývojem pro mobilní zařízení a oddálení se od vývoje pro web jsme se tak rozhodli pro framework Flutter, který nás především zaujal svými widgety a jazykem Dart. Mimo jiné má také kompletní dokumentaci poskytnutou tvůrci frameworku. Vzhledem k úrovni složitosti aplikace je nepravděpodobné, že by bylo nutné se mimo základní framework zaměřit i na technologie pro manipulaci s daty mimo základní ukládání do souborů v paměti zařízení a načítání z nich (což sám Flutter umožňuje). Případným propojením s externí cloudovou službou se zabývá následující podkapitola.

### 4.4.1 Výběr backend služby

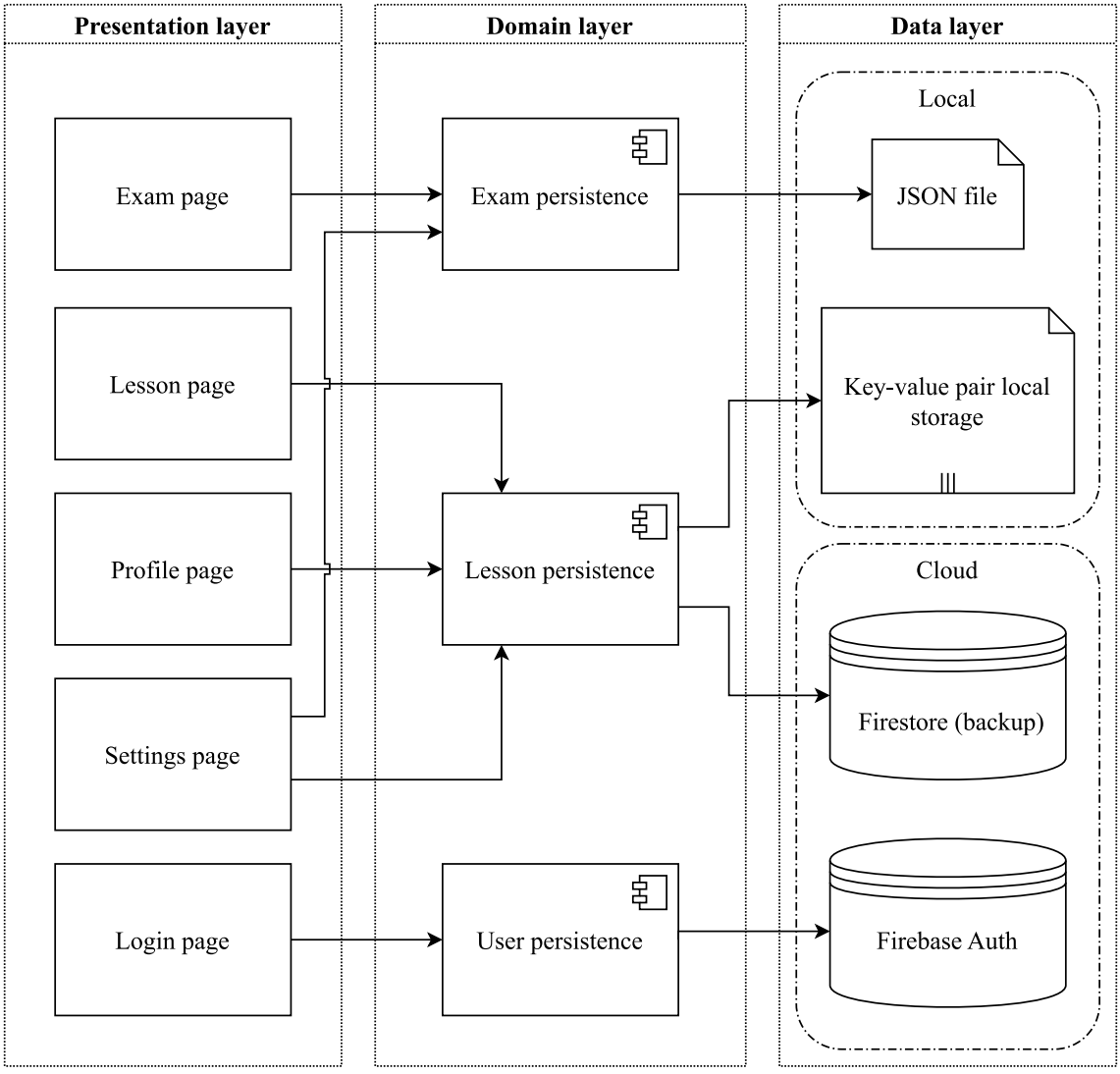
Podle požadavků definovaných v podkapitole 3.2 by systém měl poskytovat správu uživatelských účtů spolu se zálohováním data mimo zařízení. Vhodným řešením těchto požadavků, které je ale zároveň jednoduché na propojení s aplikací a nezávisí na platformě, je BaaS[1]. BaaS neboli Backend-as-a-Service je služba, která poskytuje základní funkcionalitu backendové části aplikace (tzn. ukládání dat v databázi, hosting, autentikaci uživatelů a další) jako balíček, který vývojář připojí ke svému frontendu.[4] Při výběru takové služby budou hlavními faktory cena za poskytnutí služby a konektivita s vybranou technologií samotné aplikace, tedy frameworkem Flutter.

Podle výše zmíněných faktorů mezi všemi možnými službami vyniká a byl zvolen **Firebase**, který je od stejného vývojáře jako je Flutter, tedy od společnosti Google. Ostatní nejpoužívanější služby jsou spíše vhodnější pro větší projekty (např. služba **AWS Amplify**), jsou složitější na propojení s frameworkem Flutter (např. **Backendless**) nebo jsou spíše určené zkušenějším backend vývojářům (např. **Parse/Back4App**)[20]. Firebase lze lehce připojit k existujícímu Flutter projektu a jeho základní plán zdarma vyhovuje požadavkům našeho projektu. Mimo to lze Firebase používat pomocí stejného jazyka jako Flutter, tedy jazyka Dart.[25]

[1] Backend as a Service

## 4.5 Návrh architektury

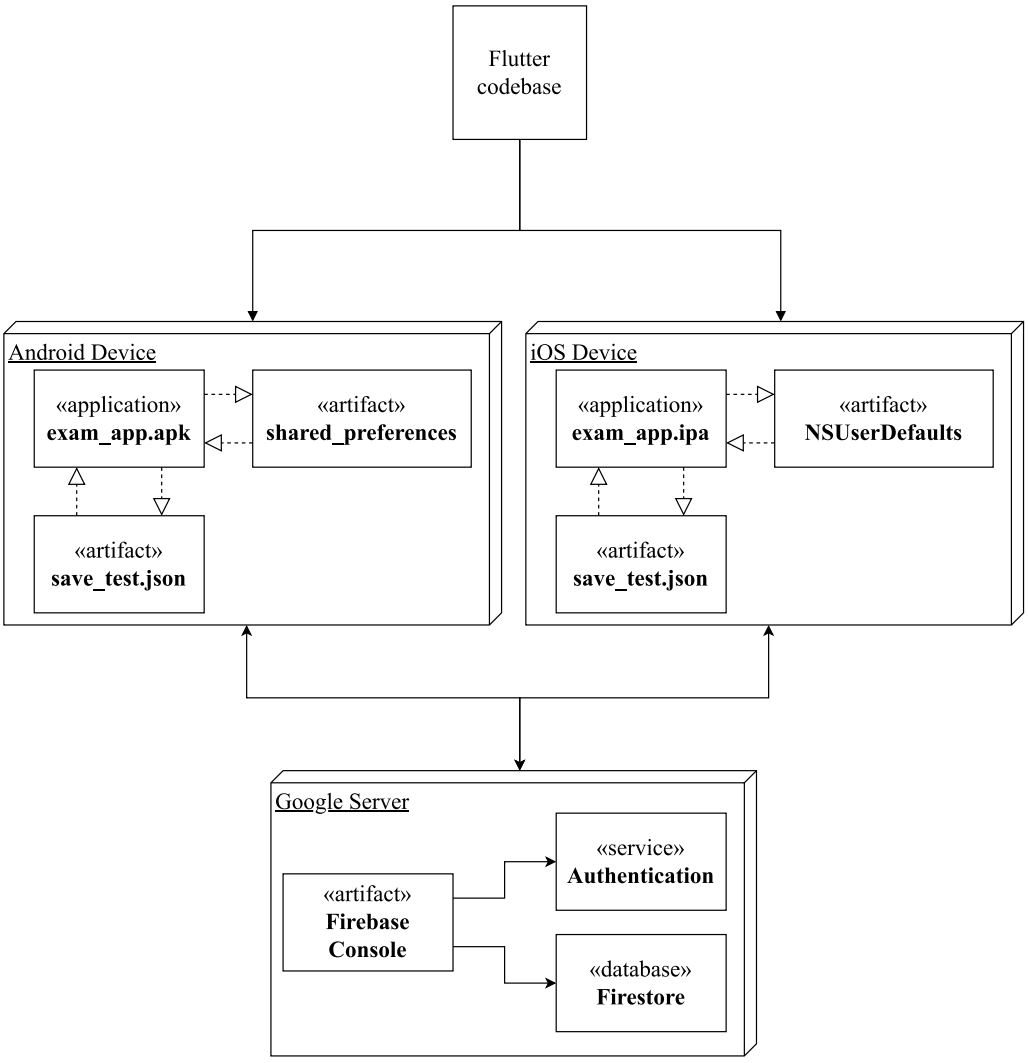
Se zvolením technologií pro vývoj aplikace lze také položit základ návrhu pro architekturu aplikace. Diagram na obrázku 4.2 zobrazuje interakci jednotlivých vrstev a komponent v nich. Celá aplikace bude vyvíjena pomocí frameworku Flutter, jehož vývojář, společnost Google, také poskytuje propojení s vlastní cloudovou (BaaS) službou Firebase. Základem aplikace bude manipulace s daty lokálně, ať už ve formě lokálních JSON souborů, nebo ukládáním klíč-hodnota párů do poskytované paměti zařízení (např. NSUserDefaults pro iOS, shared_preferences pro Android a další).[11] Propojení se službou Firebase bude poté sloužit především k zálohování ukládaných souborů a řízení uživatelských účtů.




**Obrázek 4.2:** Model vrstev

## 4.6 Vizualizace nasazení

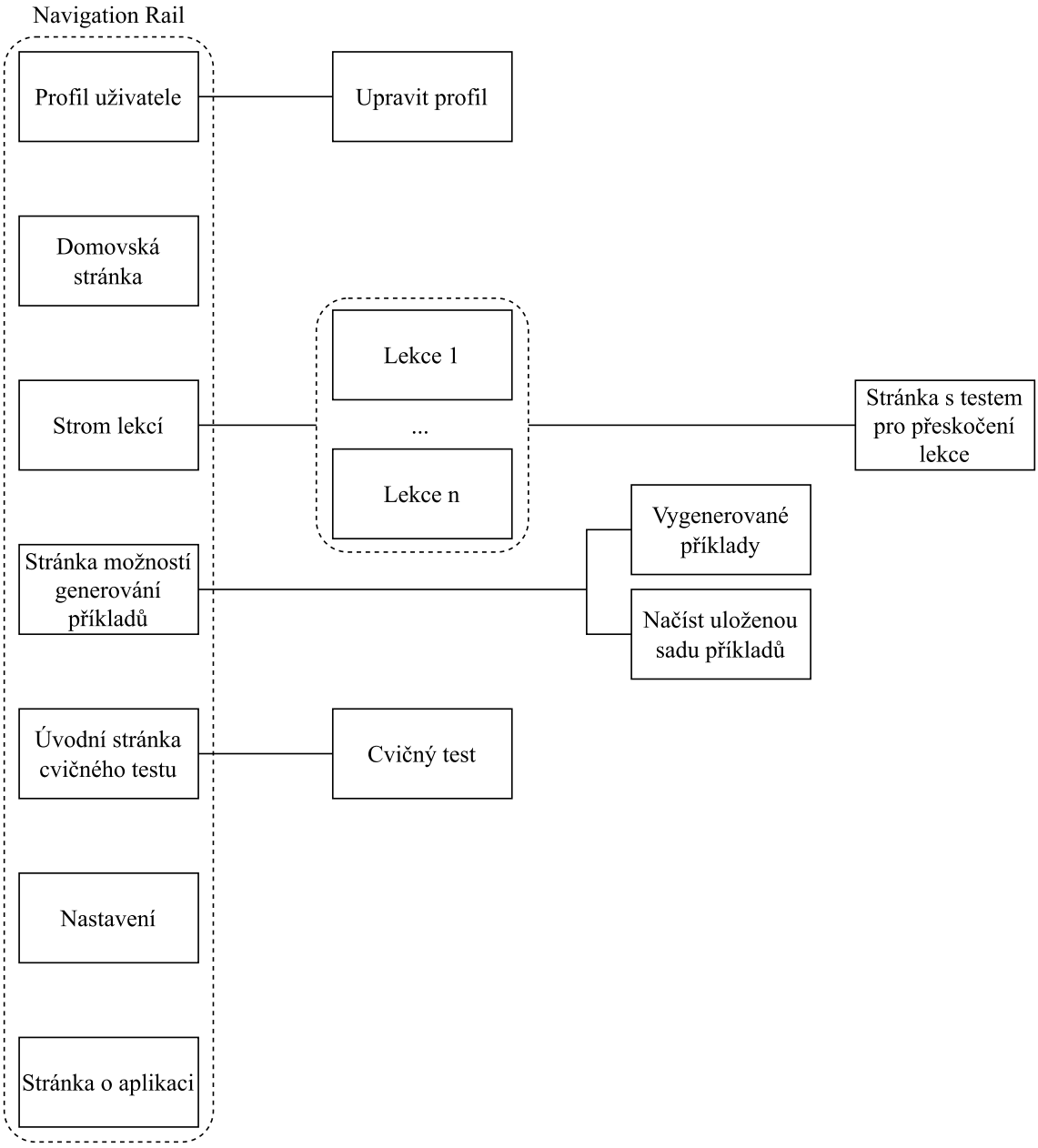
Diagram na obrázku 4.3 zobrazuje návrh nasazení aplikace při použití frameworku Flutter spolu se službou Firebase. Vzhledem k jednoduchosti aplikace bude na individuálním zařízení pouze samotná aplikace a v lokaci, kterou operační systém přiřadí aplikaci, pak také JSON soubor obsahující uložený test. Stav aplikace bude ukládán v klíč-hodnota úložišti zařízení (název se podle platformy liší). Zálohy dat a správa uživatelů pak budou starostí připojené služby Firebase.




**Obrázek 4.3:** Diagram nasazení navrhované aplikace

## 4.7 Diagram obrazovek

Obrázek 4.4 zobrazuje diagram hierarchie obrazovek v aplikaci. Základním prvkem je *Navigation Rail* (popř. *Drawer* pro menší obrazovky), který obsahuje list tlačítek, pomocí kterých se dá navigovat mezi hlavními obrazovkami. Vybrané obrazovky poté poskytují hlubší navigaci do stránek jako je například detail lekce, stránka s vygenerovanými příklady a další.



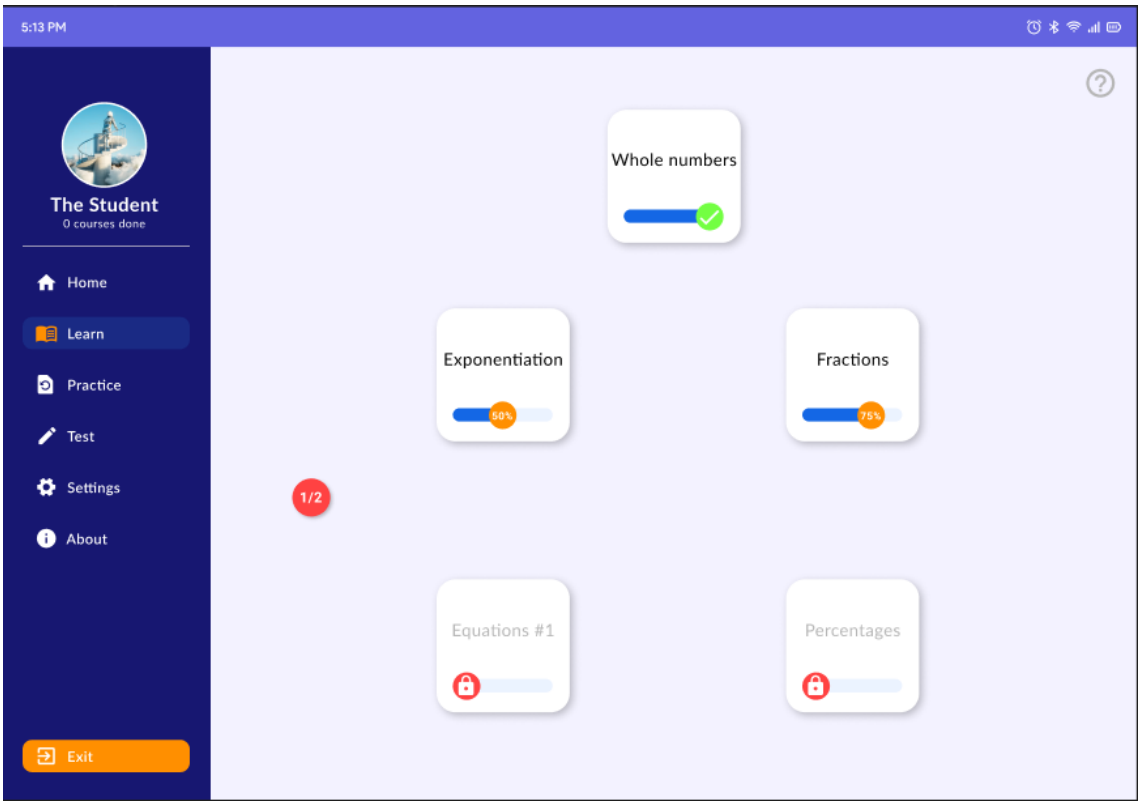

**Obrázek 4.4:** Diagram obrazovek

## 4.8 Návrh obrazovek

Tato podkapitola obsahuje vizualizaci některých částí navrhovaného systému ve formě návrhů obrazovek. Pro účely této ukázky byla zvolena varianta aplikace pro tablety.

### 4.8.1 Průchod výukou

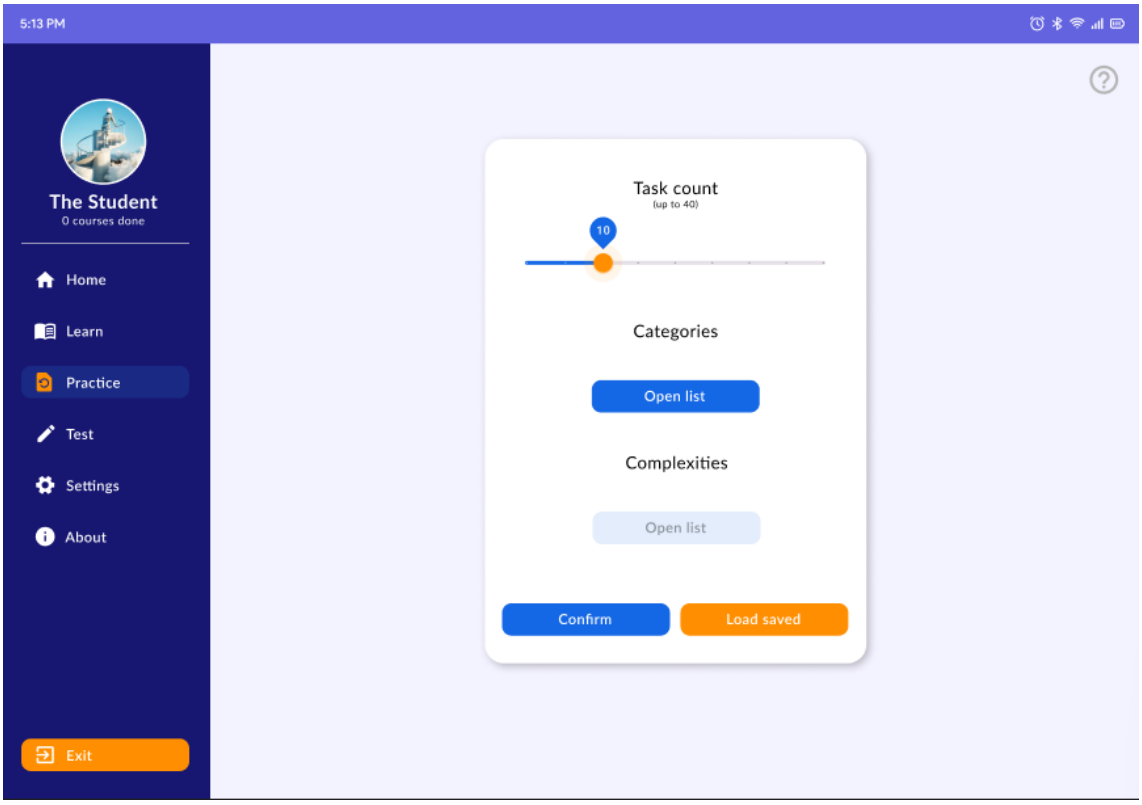
Průchod výukou je inspirován aplikací **Duolingo** zmíněné v podkapitole 2.4 Poznatky z průzkumu.




**Obrázek 4.5:** Návrh obrazovky stromu lekcí

### 4.8.2 Vygenerování sady příkladů

Uživatel může zvolit počet příkladů, téma (popř. témata) a k nim připojené obtížnosti. Pokud je registrovaným uživatelem, tak má možnost načíst si předchozí sady v případě, že si je uložil.




**Obrázek 4.6:** Návrh obrazovky výběru možností pro generování příkladů

# Kapitola 5

# Technologické provedení

Tato kapitola se zabývá vlastním vývojem aplikace, specificky výběrem vývojového prostředí a popisem práce na vývoji aplikace. Podkapitola o vývoji také obsahuje náhled do kódu aplikace spolu s krátkým popisem použitých externích knihoven a zmínku zajímavostí, se kterými jsme se setkali během vývoje. Tato sekce obsahuje dále několik ukázek obrazovek aplikace, další jsou umístěny v příloze B. Poté následuje část o průběhu nasazení a v závěru jsou uvedeny rozdíly v současném stavu aplikace oproti návrhu.

## 5.1 Vývojové prostředí

Jako nástroj pro vývoj aplikace jsme měli na výběr z několika vývojových prostředí, která nejen podporují vývoj pro mobilní zařízení, ale také mají dostupná rozšíření pro framework Flutter. Prvním z nich je **Visual Studio Code** od společnosti Microsoft, které vyniká především univerzálností, tedy pomocí různých rozšíření lze v tomto nástroji vyvíjet aplikace ve většině nejpoužívanějších programovacích jazyků. Tato vlastnost ale zároveň může být i slabinou vyžaduje-li projekt specifické zaměření.[14] Z tohoto důvodu jsme nakonec zvolili druhou možnost, **Android Studio**. Toto prostředí je také oficiálním nástrojem pro vývoj Android aplikací. Je založené na dříve používaném IntelliJ IDEA od společnosti JetBrains, které je zaměřeno na vývoj v Javě nebo Kotlinu (tedy v oficiálních jazycích pro vývoj Android aplikací). Prostředí nabízí plné propojení s frameworkem Flutter a jazykem Dart spolu se širokou škálou nástrojů pro vývoj mobilních aplikací. V prostředí se nachází také emulátor Android zařízení v různých verzích a typech zařízení.

## 5.2 Vývoj aplikace

Tato podkapitola je rozdělena do tří částí. První z nich je zaměřená na část aplikace, která je vyvíjena pomocí frameworku Flutter. Jedná se tedy především o vzhled uživatelského rozhraní, navigaci v aplikaci a lokální ukládání dat. Druhá část je poté věnována vlastní knihovně v jazyce Dart, kterou aplikace volá pro generování matematických příkladů v celkem 6 různých tématech. Poslední část popisuje způsob jakým je k aplikaci připojena služba Firebase a jakou funkcionalitu aplikace využívá. Každá z těchto částí obsahuje nejen obecný popis komponenty, ale také náhled do kódu, popř. grafickou vizualizaci algoritmů nebo struktury kódu, spolu s několika zajímavostmi, se kterými jsme se během vývoje setkali.

### 5.2.1 Mobilní aplikace

Základem vývoje ve frameworku Flutter je rozdělení jakékoliv části aplikace na "widgety", což jsou ve Flutteru předvytvořené funkční a vizuální komponenty. Což znamená, že cokoliv od navigačního panelu po jednotlivá tlačítka a text, který tlačítka obsahují, je widget. Taktéž i například uspořádání tlačítek v řádce, je widget, jmenovitě widget Row. Jednotlivé widgety poté obsahují několik vlastností, které lze libovolně upravovat, což ovlivní nejen umístění a vzhled, ale také chování widgetu v kontextu celé aplikace. Příkladem může být nastavení barvy pozadí tlačítka, ale také nastavení funkce, která bude při stisku tlačítka zavolána. Při zmínce umístění a vzhledu komponent je nutné dodat, že Flutter umožňuje pro vzhled prvků jak vzhled podle vizuálních pravidel Androidu, tak i podle iOS. Pro jednoduchost práce jsme ale zvolili widgety se základním vzhledem podle Android pravidel[1].

#### Základ aplikace a navigace obrazovek

Většina widgetů typicky dědí z třídy StatelessWidget nebo StatefulWidget. Jak už názvy napovídají, StatelessWidget se používá pro widgety, které si nepotřebují udržovat stav a během svého životního cyklu se dynamicky nemění. Pro takový účel se poté využívá StatefulWidget, který si drží stav, který lze aktualizovat funkcí *setState()*. Mimo konstruktor mají oba tyto typy několik dalších funkcí. Nejčastěji bude vývojář přepisovat funkci *build(BuildContext context)*, která vrací widget, popř. strom widgetů, které tato komponenta (tzn. nadřazený widget) bude obsahovat a popř. jimi manipulovat. Takto zabalené komponenty je pak možné různě skládat, ať do sebe, vedle sebe (widget Row, řádka), pod sebe (widget Column, sloupec) atd. Při použití pravidel pro Android pak hlavním widgetem, do kterého se ostatní vkládají je widget MaterialApp. Spuštění aplikace probíhá zavoláním *runApp(MyApp())* v metodě *main()*. Mimo funkci *build()* jsou také často přepisovány funkce *initState()* a *dispose()*. V první je například vhodné při vytvoření widgetu načíst externí data ze souboru, která bude widget používat, zatímco ve druhé se nejčastěji volají destruktory například ovladačů animací uvnitř widgetu.

V dříve zmíněném widgetu MaterialApp se typicky jako "body"vlastnost (tzn. obsah widgetu) používá widget Scaffold. V něm jsou, alespoň ve verzi widgetu pro Android, již vestavěné vlastnosti pro připojení navigačních prvků jako jsou NavigationDrawer (šuplík), který lze jednoduše identifikovat pomocí tří čar pod sebou, dále NavigationBar (persistentní řádek navigačních ikonek na spodu displeje) nebo TabBar (podobně jako NavigationBar, ale na horní části displeje). Námi zvoleným navigačním prvkem byl již při návrhu hierarchie obrazovek (viz diagram 4.4) a v grafickém návrhu (viz 4.8 Návrh obrazovek) Navigation Rail, tedy další z persistentních navigačních prvků, tentokrát podél levého kraje obrazovky.

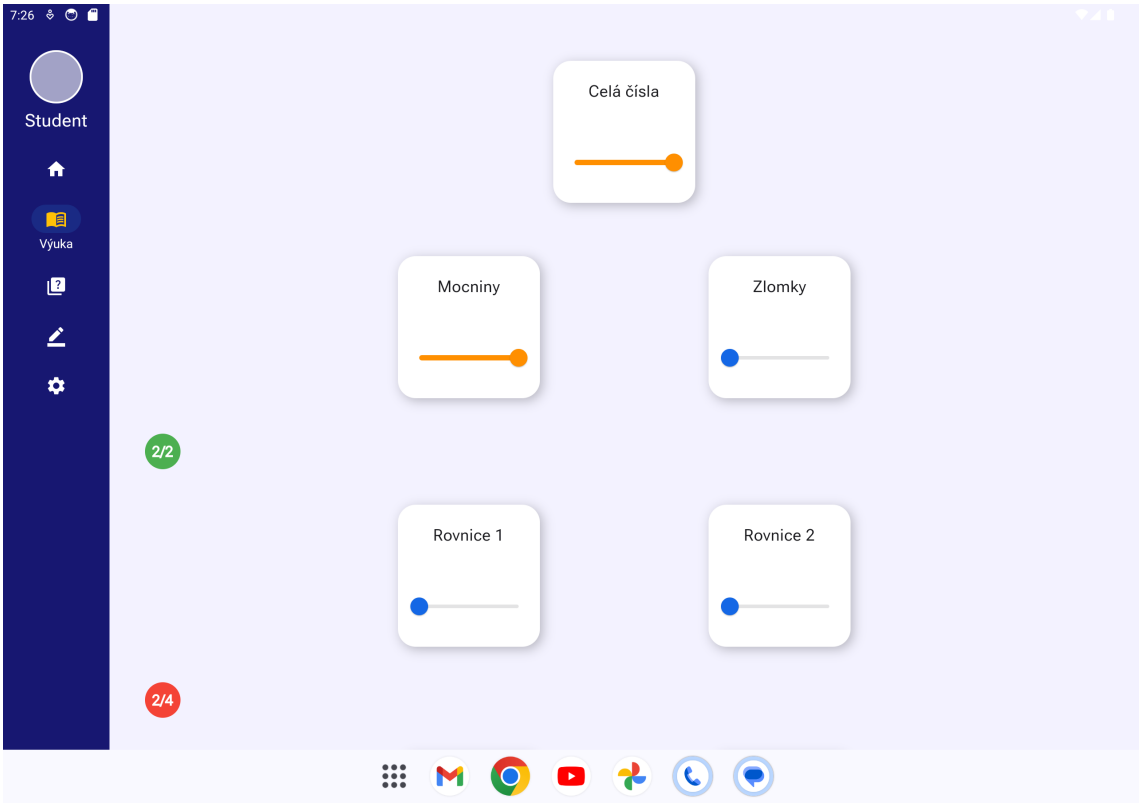
#### Stránka lekcí

Stránka se seznamem lekcí s tématy relevantními pro přijímací zkoušky se především inspirovala aplikací Duolingo, jak je zmíněno v sekci 2.4 Poznatky z průzkumu. Rozdělení lekcí bylo rozvrženo podle poznatků z kapitoly Přehled vzdělávání a samotný obsah je ve většině lekcí inspirován teorií z webových stránek Matweb (viz 2.4). Navigovační systém této stránky pracuje nezávisle na navigaci widgetu NavigationRail. Stránka se stromem lekcí odpovídá domovské adrese v navigaci, ze které

[1] m3.material.io

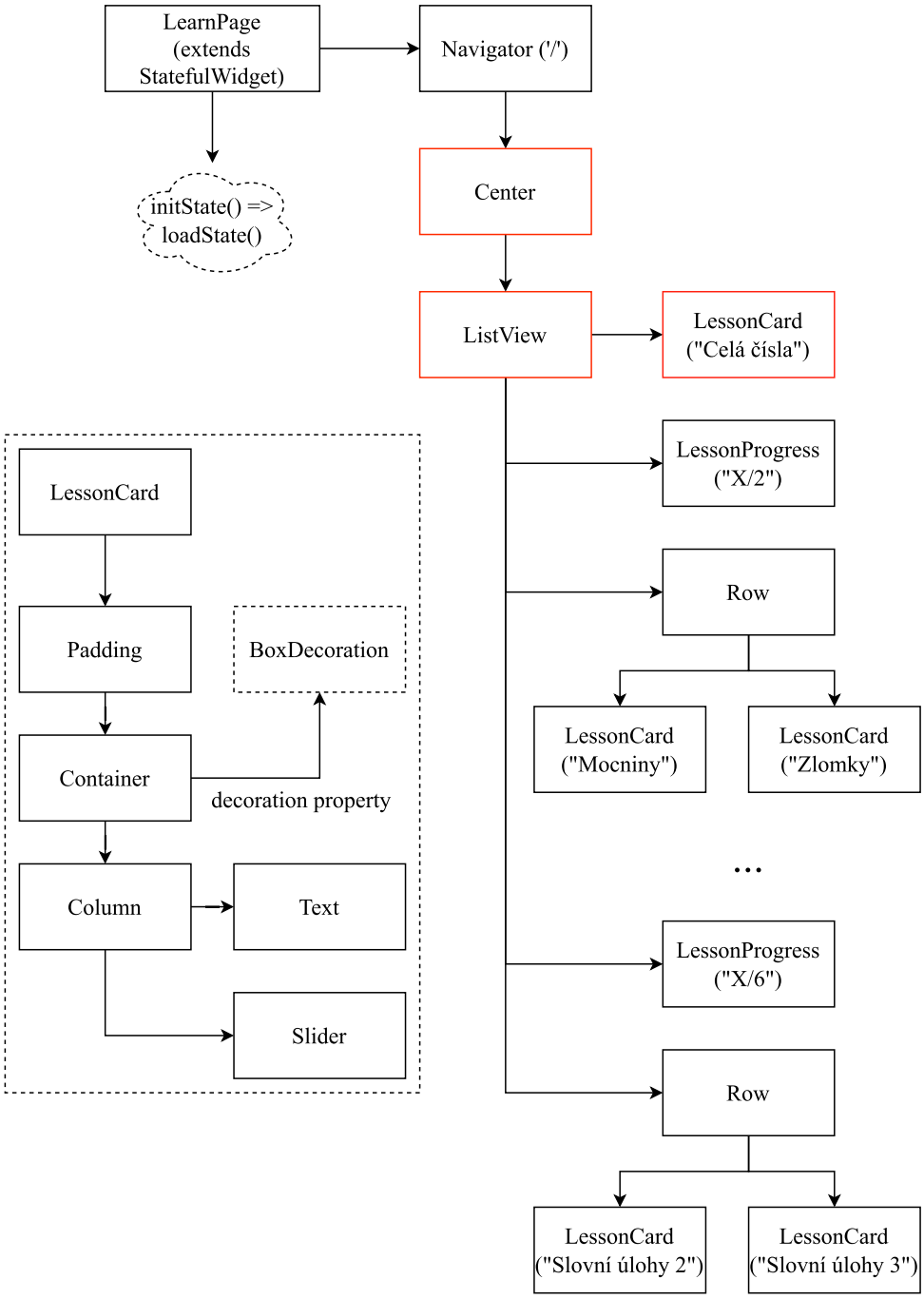
poté vznikají cesty do individuálních lekcí. Nutné je dodat, že interakce s NavigationRail má vyšší prioritu, takže pokud uživatel zvolí jinou destinaci v NavigationRail, tak současná cesta v této stránce kompletně zaniká a uživatel je přesměrován na stránku odpovídající zvolené destinaci v NavigationRail.

Screenshot této stránky z aplikace je na obrázku 5.1, pro porovnání je dostupný obrázek v kapitole Návrh řešení - 4.5. Struktura stránky, tak jak ji tvoří widgety frameworku je znázorněna na obrázku 5.2.




**Obrázek 5.1:** Screenshot obrazovky se stromem lekcí

Obrázek 5.2 zobrazuje, jak jsou do sebe či za sebou jednotlivé widgety tvořící stránku lekcí umístěny. Základem stránky je vlastní widget LearnPage, který si drží svůj stav. Při inicializaci si z paměti zařízení načte postup jednotlivých lekcí a upraví odpovídající widgety - posuvníky. Potomci tohoto widgetu jsou také zabaleni do widgetu Navigator, který řídí procházení lekcemi. Vedle vyobrazení struktury stránky je také přiložena struktura vlastního widgetu LessonCard.




**Obrázek 5.2:** Strom widgetů v obrazovce stromu lekcí, vlevo je náhled do vnitřní struktury LessonCard

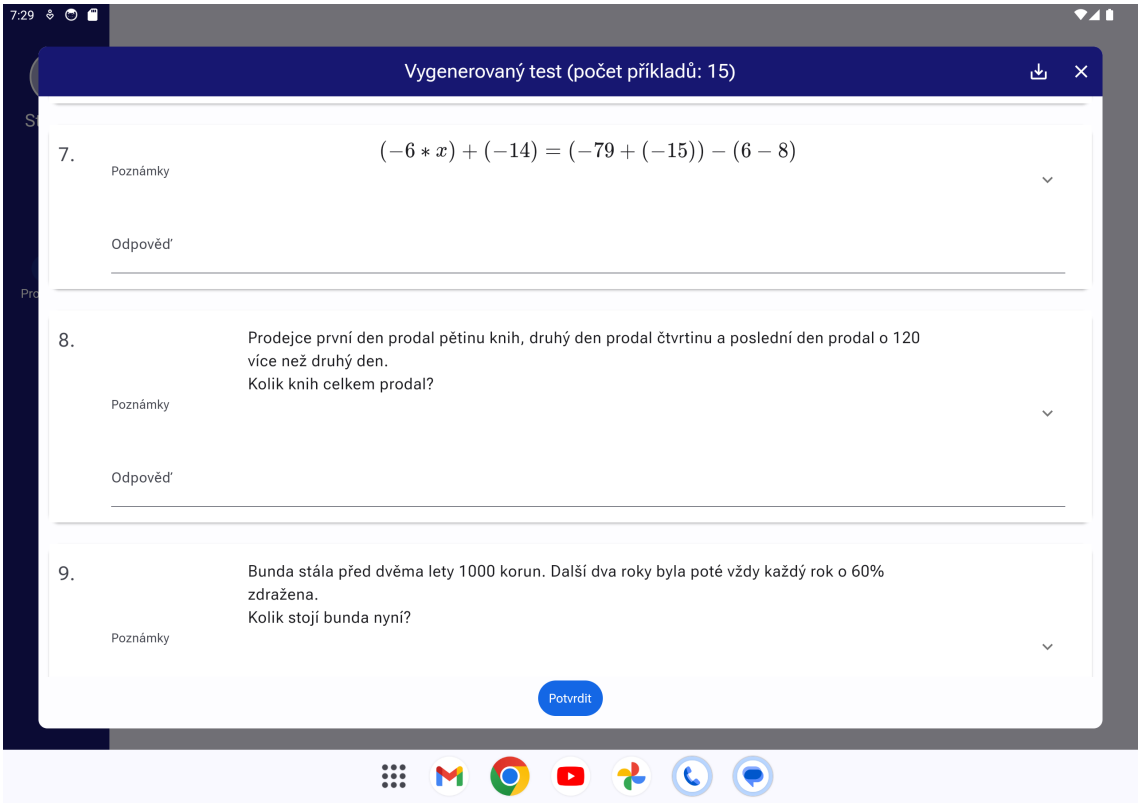
Úryvek kódu 1 obsahuje červeně vyznačené widgety Center, ListView a LessonCard z obrázku 5.2. Dále také widget GestureDetector, který poslouchá na kliknutí v rámci svého vloženého potomka (widget LessonCard). Při kliknutí zavolá widget Navigator, který přesune uživatele do obrazovky Lesson1WN (obrazovka lekce o celých číslech) a pokud se z této obrazovky vrátí (funkce *then()*, pak je zavolána funkce *loadState()*, která aktualizuje stav této obrazovky, tzn. například upraví posuvník v příslušné LessonCard podle změny v postupu lekcí.

```
return Center(
  child: ListView(
      children: [
        Center(
            child: GestureDetector(
                onTap: () {
                  Navigator.of(context).push(
                    MaterialPageRoute(builder: (context) => Lesson1WN())
                  ).then((_) {loadState();});
                },
                child: LessonCard(
                  offset: 40,
                  text: "Celá čísla",
                  slider: LessonSlider(
                    max: 3,
                    min: 0,
                    value: less1Count.toDouble(),
                    ),
                ))
            ),
```

**Listing 1:** Náhled kódu v označené části stromu widgetů

### Testový dialog

Jakýkoliv soubor příkladů, ať už se jedná o test pro přeskočení lekce, seznam vygenerovaných příkladů, nebo cvičný test na čas, je v aplikaci zobrazen ve formě dialogu, tzn. okna, které dočasně překryje současný obsah aplikace. V případě seznamu generovaných příkladů a příkladů ve cvičném testu jsou všechny příklady uloženy jako objekty třídy MathProblem (více v podkapitole 5.2.2). K nim je zároveň vytvořen i list odpovědí, který se postupně naplňuje vstupy od uživatele. V přiloženém screenshotu obrazovky (5.3) je dialog s příklady vygenerovanými podle vložených parametrů. Uživatel si může současnou sadu příkladů uložit pro budoucí řešení a nebo si může nechat vyhodnotit své odpovědi stisknutím tlačítka Potvrdit. V takovém případě je přesunut do dalšího dialogu s výčtem porovnání mezi jeho odpověďmi a správnými výsledky, které jednotlivé objekty MathProblem obsahují. Tato sekce obsahuje screenshot testu a také úryvky kódu s inicializací dialogu (2) a úryvek toho, jak vypadá widget znázorňující jeden příklad (3).




**Obrázek 5.3:** Screenshot vygenerovaného testu

Úryvek kódu 2 znázorňuje jak vzniká dialog a jak je pak řešeno volání následujícího dialogu v případě vyhodnocování výsledků. Funkce *_showTest()* je zavolána spolu s vytvořeným listem úloh a listem odpovědí, který je buď prázdný, je-li to nový test, nebo může tento list být zčásti už naplněný, jedná-li o načítání uloženého testu. Podobně jako v předchozí ukázce kódu je nejdříve zavolána funkce pro navigaci, v tomto případě *showDialog()*, a poté následuje vyhodnocení *then()*, kde pokud je vrácená hodnota true (uživatel stiskl tlačítko "Potvrdit", tedy nezavřel dialog odpovídajícím tlačítkem), tak je zavolán nový dialog, kde proběhne vyhodnocení listů s příklady s listem odpovědí.

```
void _showTest(BuildContext context,
    List<MathProblem> generatedProblems, List<String> answers){
    List<MathProblem> problems = generatedProblems;

    showDialog(
        barrierDismissible: false,
        context: context,
        builder: (context) {
          return GeneratedExam(problems: problems,
            answers: answers,);
        }).then((value) {
          if (value == true) {
            //results screen
            showDialog(
                barrierDismissible: false,
                context: context,
                builder: (context) {
                  return GeneratedExamResults(problems: problems,
                    answers: answers,);
                }
            );
          }
    });
}
```

**Listing 2:** Náhled kódu inicializace dialogu

Druhý úryvek kódu (3) znázorňuje, jak jsou jednotlivé příklady vykreslovány. Každý je nejdříve zabalen do widgetu ListTile, kde zadání příkladu je ve vlastnosti *"title"*. Tento widget pak také obsahuje sloupec (widget Column) s vlastním widgetem NoteField (pole pro poznámky, více v sekci Použité externí knihovny) a widgetem pro vkládání textu (TextField). Ten je ovládán přiřazeným ovladačem z listu ovladačů, kde každý má přiřazený své vlastní pole odpovídající jednomu z příkladů. Obsahuje-li list odpovědí (widget.answers) již nějaký neprázdný řetězec, tak právě tento ovladač je při inicializaci celého tohoto widgetu (funkce *initState()*) s tímto řetězcem vytvořen a TextField už při své inicializaci nějaký řetězec obsahuje (případ načítání uložené sady příkladů).

```
Card(
    color: Theme.of(context).colorScheme.onPrimary,
    elevation: 2,
    child: ListTile(
        contentPadding: ...,
        title: Center(...),
        subtitle: Padding(
          padding: ...,
          child: Column(
            children: [
              const NoteField(),
              TextField(
                controller: answerControllers[i],
                ...,
                decoration: const InputDecoration(
                  label: Text('Odpověď'),
                ),
                keyboardType: TextInputType.number,
                onChanged: (value) {
                  widget.answers[i] = value;
                },
              )
            ],
          ),
        ),
        leading: Text(...),
        titleAlignment: ...,
        tileColor: Theme.of(context).colorScheme.onPrimary,
        isThreeLine: true,
    ),
),
```

**Listing 3:** Náhled kódu widgetu pro zobrazení příkladu

#### Použité externí knihovny

Ačkoliv Flutter není v porovnání s například React Native na knihovny bohatý, tak jich bylo několik v tomto projektu použito. Mimo to jsme pro účely generování příkladů ve formátu odpovídajícímu přijímacím zkouškám vytvořili knihovnu vlastní, ta je předmětem další podkapitoly. Tato sekce obsahuje seznam použitých externích knihoven spolu s krátkým popisem, jak byly v aplikaci použité.

Hlavní knihovny, které stojí za zmínku jsou celkem 3. Dvě z toho slouží ukládání dat do paměti zařízení. Knihovna **path_provider**[2] slouží k identifikaci složky, která byla aplikaci operačním systémem přiřazena a do které si aplikace může ukládat soubory. Aplikace toto využívání k ukládání vygenerovaného testu jako soubor JSON. Poté aplikace také využívá knihovnu **shared_preferences**[3], která poskytuje vývojáři přístup do lokálního úložiště zařízení pro ukládání klíč-hodnota párů. Různé platformy mají různé přístupy k těmto datům, takže tato knihovna je rámci filozofie Flutteru, jeden kód pro více platforem, zabalí a poskytne vývojáři univerzální přístup. Poslední hlavní knihovnou je KaTeX renderer pro zobrazování matematických příkladů, knihovna **flutter_math_fork**[4]. Ta poskytuje vlastní widget Math.tex, který se podobá původnímu widgetu Text a lze jej tak podobně používat a upravovat. Jakýkoliv typ matematického příkladu v aplikaci mimo slovních úloh je zobrazován právě pomocí této knihovny.

Další použité knihovny jsou méně důležité, nicméně za zmínku stojí knihovna **super_rich_text**[5]. V základu je Flutter widget Text jen složitě zpětně upravovatelný, co se například tučného písma nebo kurzívy týče. Dosažitelné to je pouze určením rozsahů (spanů) ve kterých bude text psaný například tučně. Tato knihovna nabízí svůj widget SuperRichText, který se, podobně jako widget Text, inicializuje s řetězcem. SuperRichText ale předem určené znaky jako je hvězdička či podtržítko rozpoznává jako značky pro začátek a konec tučného textu, respektive kurzívy. Řetězec vkládaný do takového widgetu stačí tedy jen vhodně označkovat. Tyto značky lze zároveň dle libosti upravovat či i nové přidávat (například pro označení textu, který bude jinou barvou). Výše zmíněný widget NoteField využívá knihovnu **expandable**[6], která poskytuje funkcionalitu rozbalovacího kontejneru. Pole pro poznámky tak musí pro používání nejdříve uživatel "rozbalit".

### 5.2.2 Generátor matematických příkladů

Jedním z hlavních cílů tohoto projektu bylo umožnit žákům připravujícím se na přijímací zkoušky na střední školy procvičovat si dle libosti různé témata, se kterými se u zkoušek mohou setkat. Z podkapitoly 2.5 Analýza obsahu testů jsme získali pro tento účel celkem 7 různých témat, v jakém počtu se v testech vyskytují a jaké typy příkladů tato témata obsahují (viz tabulka 2.2). Každé téma, mimo slovní úlohy, jsme také rozdělili na několik úrovní podle obtížnosti. Do generátoru jsme se kvůli úspoře času rozhodli prozatím neimplementovat generování příkladů na úpravu výrazů. Nejenže by jejich implementace a následné vyhodnocování výsledků byly podstatně odlišné od ostatních typů příkladů, ale jak již bylo zmíněno v podkapitole 2.5, tak Cermat na konci zadání zkoušky uvádí pro žáky vzorce pro

[2] pub.dev/packages/path_provider
[3] pub.dev/packages/shared_preferences
[4] pub.dev/packages/flutter_math_fork
[5] pub.dev/packages/super_rich_text
[6] pub.dev/packages/expandable

řešení umocňování výrazů, takže procvičování tohoto tématu považujeme za méně důležité. Nicméně i přesto jsme se rozhodli do seznamu lekcí zařadit také lekci právě o úpravě výrazů.

### Implementace

Podle získaných vzorů příkladů jsme se rozhodli generátor příkladů vytvořit jako vlastní knihovnu pro jazyk Dart mimo samotnou aplikaci ve frameworku Flutter. Pro účely této knihovny jsme použili několik již existujících knihoven pro práci s matematickými výpočty, které bude uvedeny v další podkapitole.

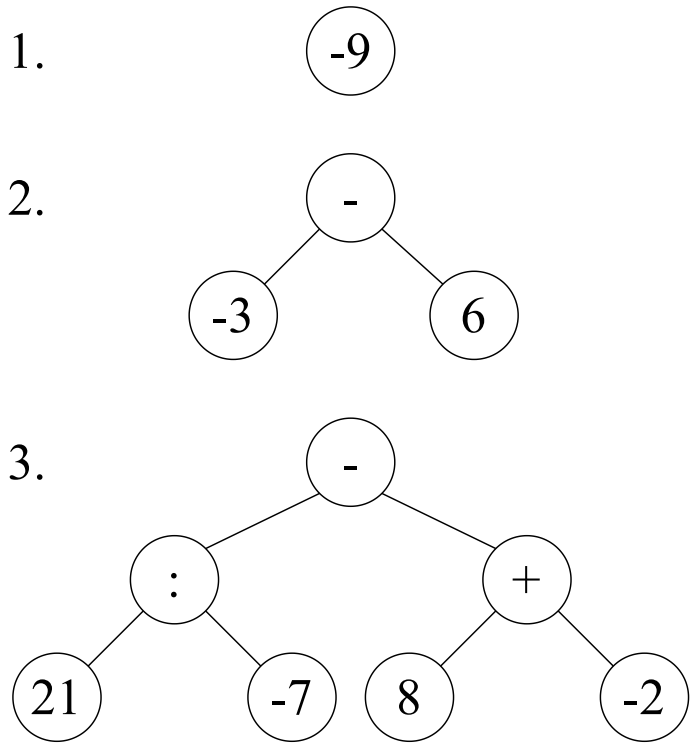
Základem celé knihovny je třída **MathProblem**, jejíž strukturu lze najít v doménovém modelu - 4.1. Mimo atributů třída obsahuje také funkce pro konverzi do a z formátu JSON, čehož naše aplikace využívá k ukládání sad generovaných příkladů. Každé téma příkladů je reprezentováno svou vlastní třídou s 1 (v případě slovních úloh) až 4 metodami odpovídajícími různým obtížnostem příkladů. Metody generování lze rozdělit do dvou hlavních typů, přičemž třída pro generování příkladů na celá čísla obsahuje oba typy a třída pro generování příkladů na rovnice obsahuje ještě jeden typ navíc.

1. Prvním typem je **generování hodnot do předem připravené formy**. Tento typ využívají především generátory slovních úloh, ale také příklady na zlomky a mocniny.

   U slovních úloh je pro každé různé zadání vytvořena základní textová forma, do které jsou poté vloženy skupiny vybraných slov a vypočtených hodnot. Každé zadání obsahuje list skupin slov, která tvoří různá zadání pro stejnou textovou formu (tzn. např. obměňování toho, co prodavač prodává ve slovní úloze na procenta, nebo prohazování jmen lidí vykonávajících určitou práci ve slovních úloh o společné práci). Hodnoty vkládané do formy jsou vypočítávány zpětně z náhodně vybraného výsledku, který je přiměřen realitě (tzn. smysluplné počty odpracovaných hodin či ceny produktů).

   Pro příklady na zlomky a mocniny pak bylo vybráno na základě archivních testů několik forem, do kterých jsou vkládány hodnoty tak, aby úroveň obtížnosti odpovídala znalostem žáka 9. třídy. Tedy například úlohy s odmocninami obsahují pouze předem určené hodnoty pod odmocninou, ačkoliv tyto hodnoty mohou být skryté pod například odčítáním (tzn. je-li vybraná hodnota vložená pod odmocninu 0.09, tedy $0.3^2$, pak výsledná odmocnina může být například $\sqrt{1 - 0.91}$).

2. Druhý typ je podobně jako slovní úlohy vytvářen zpětně podle náhodně vybraného výsledku. K vytvoření takového příkladu využívají generátory **vyjádření aritmetických výrazu jako stromu**. Takový strom má ve vnitřních uzlech operátory a listy stromu představují operandy, jak je znázorněno na obrázku 5.4.




**Obrázek 5.4:** Výraz $(21 : (-7)) - (8 + (-2))$ ve formě stromu

Pro generování příkladů na celá čísla stačí pouze určit výslednou hodnotu a hloubku rekurze. Pro rovnice s jedním výskytem neznámé je nejdříve určena hodnota neznámé, poté je náhodně vytvořen výraz na levé straně rovnice pomocí stromu. Podle hodnoty neznámé je vypočítána současná hodnota pravé strany, ze které je poté také vytvořen výraz pomocí stromu. Pokud se jedná o rovnice s neznámou na obou stranách rovnice, tak jsou nezávisle na sobě vytvořeny obě strany rovnice pomocí stromu a v závěru poté zvláštní metoda vytvoří nový výraz z rozdílu stran, který připojí k pravé straně rovnice.

3. Poslední typ je využíván pro generování rovnic se zlomky a trojnásobným výskytem stejné neznámé (viz podkapitola 2.5 Analýza obsahu testů). Pro vytváření rovnic v tomto tvaru jsme vytvořili algoritmus popsaný v krocích níže.

   (a) Vyber hodnotu neznámé.

   $x = -2$

   (b) Vytvoř lineární rovnici $ax + b = 0$.

   $4x = -8 \rightarrow 4x + 8 = 0$

   (c) Náhodně vyber $X$ hodnot, jejichž součet se rovná hodnotě $a$.

   $4 \rightarrow [+8, -9, +5]$

   (d) Náhodně vyber $X + 1$ hodnot, jejichž součet se rovná hodnotě $b$.

   $8 \rightarrow [-10, +12, +3, +3]$

   (e) Spoj vybraná čísla.

   $(8x - 10) + (-9x + 12) + (5x + 3) + 3$

(f) Vytkni největšího společného dělitele číselných hodnot pro každou závorku.

$$2(4x-5)+3(-3x+4)+1(5x+3)+3$$

(g) Najdi nejmenší společný násobek vytknutých hodnot.

$$lcm(2,3,1)=6$$

(h) Vytvoř zlomky pomocí nalezeného čísla.

$$\frac{4x-5}{3}+\frac{-3x+4}{2}+\frac{5x+3}{6}+\frac{1}{2}$$

(i) Zlomky rozmísti rovnoměrně na obě strany rovnice.

$$\frac{-3x+4}{2}+\frac{1}{2}=-\frac{4x-5}{3}-\frac{5x+3}{6}$$

**Použité knihovny**

Pro účely námi vybraných témat knihovna využívá pouze 3 externí knihovny. První je knihovna **equations**[7], která poskytuje parsování a výpočet výrazů, které jsou uloženy jako řetězce. U témat, která jsou v tomto projektu spíše jako podpůrná, tedy téma celých čísel, zlomků a mocnin, pro zjednodušení práce tato knihovna počítá přímo výsledky vygenerovaných příkladů. U rovnic, jak je zmíněno v příslušné sekci předchozí podkapitoly, tato knihovna počítá mezivýsledky, které vzniknou po vytvoření stromů.

Další knihovna, **binary_expression_tree**[8], byla původně přidána do knihovny pro tvorbu stromů pro téma rovnic. Nicméně provedení knihovny se ukázalo být nekompatibilní se zavedeným způsobem generování příkladů, takže nakonec jediné využití knihovny spočívá v používání poskytované třídy Node pro tvorbu vlastních stromů. Mimo to, knihovna poskytuje čtení stromu pouze ve způsobech zápisu prefix a postfix, které jsou pro naše účely nevhodné. Ve výsledku jsme tedy i metodu pro čtení námi vytvořených stromů, infix, který je určený spíše pro čtení lidmi, vytvořili vlastní.

Podobně jako předchozí knihovna, i z poslední knihovny **fraction**[9] zůstala většina funkcionalit nevyužitá. Již podle názvu lze poznat, že největší využití by se knihovně dostalo při generování příkladů se zlomky. Nicméně knihovna používá pouze datový typ integer v konstruktoru objektů třídy Fraction, takže by použití v příkladech s desetinnými čísly nebo složenými zlomky nebylo vhodné. Bohatě využívanými funkcemi této knihovny jsou ale funkce *toFraction()* a *reduce()*. První z nich je využívaná naší knihovnou prakticky kdykoliv, kdy výsledek příkladu vychází jako desetinné číslo. V tu chvíli jej tato funkce převede na zlomek. Není-li takový zlomek v základním tvaru, pak přichází na řadu druhá funkce, *reduce()*, která jej do něj převede.

### 5.2.3 Backend

Pro zjednodušení práce bylo místo vytvoření vlastního backendu zvoleno připojení k již existující službě, tedy backend as a service (BaaS), Firebase[10]. Jelikož jsou Firebase a framework Flutter od téhož vývojáře, tak propojení mezi nimi je vcelku

[7] pub.dev/packages/equations
[8] pub.dev/packages/binary_expression_tree
[9] pub.dev/packages/fraction
[10] firebase.google.com

jednoduché. Po založení projektu na webové stránce Firebase a připojení projektu ve frameworku Flutter v GitHub repozitáři stačí do samotného projektu pouze vložit identifikační soubory, které se zaktivují při buildu projektu na jednotlivé platformy. Dále je také potřeba nejdříve připojit do projektu knihovny odpovídající jednotlivým službám, které bude aplikace v rámci Firebase využívat, v našem případě knihovny **firebase_core**[11], která je povinná pro jakékoliv využití Firebase, dále pak **cloud_firestore**[12] pro připojení Firestore databáze a **firebase_auth**[13] pro připojení autentikační služby. Posledním krokem je inicializace připojení ke službě při startu aplikace pomocí metody *Firebase.initializaceApp(options: DefaultFirebaseOptions.currentPlatform)*, kde *DefaultFirebaseOptions* pochází právě z vložených identifikačních souborů odpovídajících současné platformě. Dále jsou uvedeny dvě ukázky použití této služby uvnitř aplikace.

### Ukázka použití autentikační služby

V částech aplikace ve kterých stav aktuálního widgetu závisí na přihlášení nějakého uživatele lze použít naslouchač na změnu stavu autentikace, který je poskytován singletonem služby FirebaseAuth. Následující úryvek kódu (4) ukazuje, jak je na stránce Nastavení (Settings) právě tato služba volána. Při vytvoření instance stránky je stránce poskytnuta instance FirebaseAuth, která v přepsané metodě *initState()* inicializuje také naslouchač na změnu autentikace. Samotné přihlašování pak probíhá pomocí zabudovaných funkcí, které aplikace volá na instanci FirebaseAuth, například anonymní přihlášení by pak odpovídalo volání *_auth.signInAnonymously()*.

```
class _SettingsPageState extends State<SettingsPage> {
    ...
    final FirebaseAuth _auth = FirebaseAuth.instance;
    User? _user;

    @override
    void initState() {
        super.initState();
        ...
        _auth.authStateChanges().listen((user) {
          setState(() {
            _user = user;
          });
        });
    }

}
```

**Listing 4:** Náhled kódu pro FirebaseAuth

[11] pub.dev/packages/firebase_core
[12] pub.dev/packages/cloud_firestore
[13] pub.dev/packages/firebase_auth

### Ukázka komunikace s databází Firestore

Databáze Firestore služby Firebase je založená na kolekcích a dokumentech (je NoSQL). V současném stavu aplikace je připojení k databázi využíváno pouze pro sběr statistik a zpětné vazby od uživatelů přihlášených anonymně. Úryvek kódu 5 obsahuje ukázku kódu pro připojení se k určité kolekci v databázi (v našem případě kolekce *UserStats*) a ukládání dat jako dokumentů do této kolekce. (Pozn. *StatisticPersistence* je vlastní třída, která se stará o ukládání dat jako je úspěšnost v testech, počet vygenerovaných příkladů a další podobné údaje o postupu uživatele aplikací)

```
Future<void> _sendAnonymousData(String name, String feedback) async {
    StatisticPersistence sp = StatisticPersistence();
    Map<String, dynamic> statistics = await sp.loadData();
    CollectionReference cr = FirebaseFirestore.instance.
                                        collection('UserStats');
    cr.doc(name).set({...statistics, "text": feedback});
}
```

**Listing 5:** Náhled kódu pro komunikaci s Firestore

### Zabezpečení

O zabezpečení aplikace lze mluvit ve 3 ohledech: bezpečnost anonymního přihlášení, přihlášení pomocí Google a zabezpečení databáze Firestore. Vzhledem k jednoduchosti aplikace je prozatím využíváno základní nastavení autentikace pro Firebase Auth a Firebase *security rules* pro Firestore (tzn. filtrovací pravidla pro jakoukoliv komunikaci s databází). Dále také aplikace nerozlišuje více rolí uživatelů, takže autorizace pro současnou aplikaci není relevantní. Při anonymním přihlášení obdrží přihlášený uživatel authentication token jako kdyby se přihlásil jiným způsobem, nicméně toto přihlášení pouze trvá buď dokud se uživatel sám neodhlásí, nebo nejsou smazána data o současné *session*, kdy uživatel přijde o autentikační token. Anonymní přihlášení také pokaždé vytváří nového unikátního uživatele, i když by se uživatel přihlašoval ze stejného zařízení. Přihlašování pomocí Google probíhá podle vestavěných funkcí, kdy je uživatel nejdříve přesměrován na obrazovku či dialog pro vybrání Google účtu k přihlášení a v případě úspěšného přihlášení je mu přiřazen JWT token. Práce s tokeny jako jejich obnovování probíhá uvnitř Firebase a vzhledem k současně primitivnímu využití tohoto typu přihlášení v aplikaci, kdy je pouze dekorační (více v 5.4 Porovnání s návrhem), není nutné jej dále rozebírat. V podobném stavu je také komunikace s databází Firestore, která je zatím využívána pouze pro sbírání dat, čtení dat z databáze aplikací tedy není implementováno. Základní nastavení zabezpečení (*security rules*) tedy nebylo nijak od vytvoření projektu ve Firebase konzoli upravováno.

Součástí služby Firebase je také hosting webové stránky, o tom více v následující podkapitole 5.4 Porovnání s návrhem.

## 5.3 Nasazení

Ačkoliv byla aplikace plánována primárně k nasazení na iOS, ve finále existuje jako webová stránka a také jako instalační .apk soubor pro zařízení s operačním systémem Android. Hlavním důvodem byly poplatky za distribuci přes App Store (iOS) a Google Play (Android) a také poněkud složitější cesta pro build aplikace bez přístupu k macOS zařízení (jednou z možností bylo využití služby CodeMagic pro vzdáleným přístup k Mac mini na kterých by byl instalační .ipa soubor pro iOS vytvořen). Nicméně Android zařízení alespoň povolují volně instalovat stažené .apk soubory bez nutnosti distribuce přes Google Play. Hlavním přístupovou cestou nakonec zůstává webová aplikace, která je hostována ve službě Firebase na adrese mathexamapp.web.app. Instalovatelný apk soubor lze stáhnout z GitHub repozitáře[14].

## 5.4 Porovnání s návrhem

Hlavním rozdílem současného stavu aplikace oproti návrhu je především absence rozdělení uživatelů. Registrovaný uživatel v tuto chvíli nezískává kromě vlastního profilového obrázku nic navíc oproti běžnému nepřihlášenému uživateli. Všechny definované požadavky (viz 3.2 Požadavky), kromě zálohování dat a správy hesel, tak prozatím platí pro jakéhokoliv uživatele aplikace. Dále jsou rozdíly oproti návrhu, ale i další poznámky k implementaci, které stojí za zmínku, rozděleny na ty technického druhu a vizuální. V závěru této podkapitoly je také porovnání plánovaného obsahu s obsahem, který v tuto chvíli aplikace nabízí.

### 5.4.1 Technické rozdíly

Významnou odlišností oproti celému návrhu je přidání webové verze aplikace. Zatímco návrh hovořil především o nasazení na platformy s operačním systémem Android a iOS, tak nakonec byla pro účely testování nasazena aplikace na web. Od toho se také odvíjí platformní omezení a to především v ztrátě možnosti ukládat vygenerovaný test do paměti zařízení, což verze pro Android umožňuje. Nicméně možnost ukládat postup uživatele aplikací jako klíč-hodnota páry do poskytnutého lokálního úložiště je na webu dostupná.

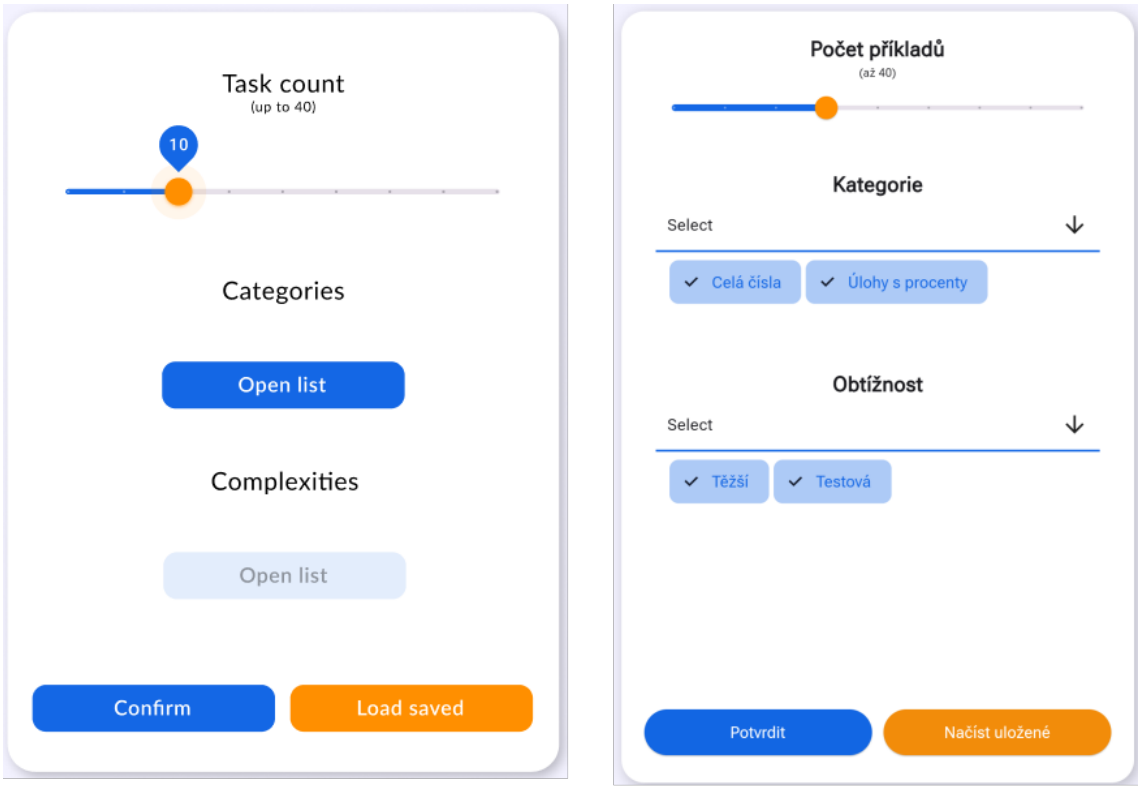
Primárním cílem aplikace je poskytnout výuku a procvičování určených témat, úroveň propojení s backendovou službou je tedy, jak již bylo výše zmíněno, minimální a bude spíše předmětem případného budoucího vývoje.

### 5.4.2 Vizuální rozdíly

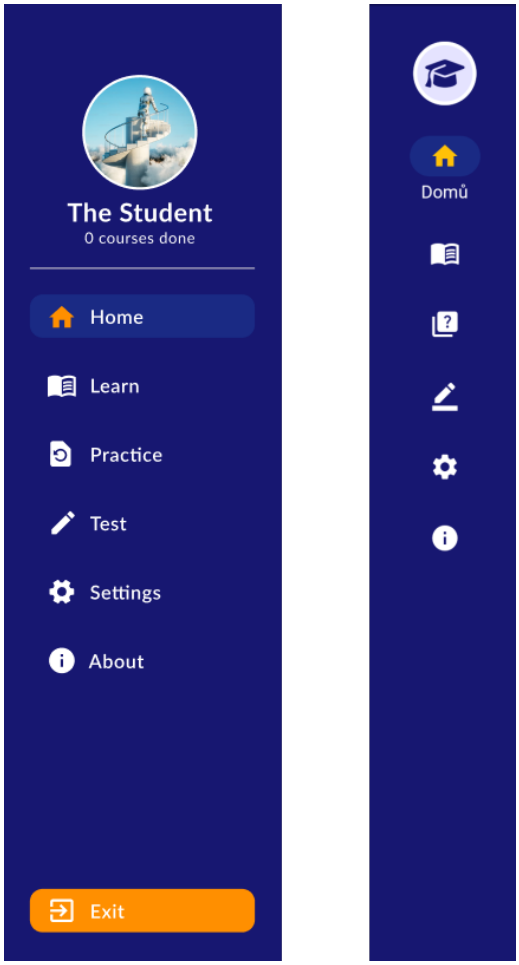
Odlišnosti mezi grafickým návrhem a reálnou podobou aplikace jsou způsobeny především limitacemi frameworku a znalostmi autora. Zároveň se ale také díky různým knihovnám poskytujícím dodatečné vizuální a funkční prvky některé části aplikace staly podle našeho názoru více uživatelsky přívětivými (viz obrázek 5.5). Pro výběr kategorií a obtížností je použita knihovna **multi_select_flutter**.[15]

[14] github.com/severfrant/exam_app_distribution
[15] pub.dev/packages/multi_select_flutter

Další větší změnou je úprava bočního panelu (Navigation Rail), ten byl pro uvolnění místa napravo od něj zmenšen a popisky jednotlivých tlačítek se nyní objeví jen u aktivního tlačítka panelu. Také jsme po uvážení odstranili tlačítko "Exit", jelikož aplikace pro mobilní zařízení (a i webové aplikace) z praktického hlediska nepotřebují dedikované ukončovací tlačítko. Porovnání změn se nachází na obrázku 5.6.




**Obrázek 5.5:** Výběr možností pro generování (verze z návrhu vlevo, verze v aplikaci vpravo)




**Obrázek 5.6:** Hlavní panel aplikace (verze z návrhu vlevo, verze v aplikaci vpravo)

### 5.4.3 Stávající obsah aplikace

Pro srovnání obsahu použijeme tabulku 2.2 z podkapitoly 2.5 Analýza obsahu testů. V knihovně pro generování příkladů chybí pouze příklady na úpravu výrazů, důvody byly zmíněny výše. Dále také chybí zhruba polovina plánovaných typů slovních úloh, zejména úlohy na pohyb. Co se obsahu lekcí v aplikaci týče, tak podobně chybí lekce na úpravu výrazů a druhá lekce na slovní úlohy (která by obsahovala právě úlohy na pohyb). Stav obsahu je zaznamenán v tabulce 5.1 níže, její struktura odpovídá tabulce 2.2 z analýzy obsahu testů.

| Téma | Typy | Lekce | Generování |
|---|---|---|---|
| Celá čísla | jednoduchý příklad s operacemi +, -, *, : | ✓ | ✓ |
| Zlomky a desetinná čísla | jednoduchý příklad s operacemi +, -, *, : | ✓ | ✓ |
| | složený zlomek | ✓ | ✓ |
| Mocniny a odmocniny | druhá (od)mocnina se vztahuje pouze na jedno číslo | ✓ | ✓ |
| | více čísel pod stejnou druhou odmocninou | ✓ | ✓ |
| Úprava výrazů | úprava podle vzorce (rozklad na součin) | ✗ | ✗ |
| Rovnice o jedné neznámé | příklad omezený na celá a desetinná čísla | ✓* | ✓* |
| | příklad s více zlomky, neznámá pouze v čitateli | ✓ | ✓ |
| Slovní úlohy o procentech | slovní úloha na změnu hodnot o X% | ✓ | ✓ |
| | slovní úloha s vícenásobnými změnami hodnot | ✓ | ✓ |
| Slovní úlohy o rovnicích | slovní úloha o počítání se směsmi | ✗** | ✓ |
| | slovní úloha o pohybu (za sebou, proti sobě, spolu) | ✗ | ✗ |
| | slovní úloha o společné práci (nebo spotřebě) | ✓ | ✓ |
| | slovní úloha o rozdělení na části | ✓ | ✓ |

**Tabulka 5.1:** Současný stav aplikace (* - pouze celá čísla, ** - součást chybějící druhé lekce o slovních úlohách spolu s úlohami na směsi)

## 5.5 Shrnutí vývoje

Na základě analýzy a návrhu z předchozích kapitol byla jako první výstup vytvořena aplikace v multiplatformním frameworku Flutter, ke které byla později připojena služba Firebase jako backend. Aplikace obsahuje většinu plánovaného obsahu a z hlediska technického ji chybí pouze hlubší propojení s backendovou službou. Aplikace byla nasazena jako mobilní aplikace pro operační systémy Android a také jako webová aplikace. Obě verze poskytují takřka identickou funkcionalitu s výjimkou ukládání vygenerované sady příkladů do paměti mobilního zařízení, webová verze toto neumožňuje. Obě aplikace tak nabízí předem vytvořené lekce s relevantními tématy pro přijímací zkoušky na střední školy, dále generování matematických příkladů podle zvolených parametrů a nakonec test nanečisto s časovým limitem, který napodobuje strukturu a obtížnost reálného testu. Aplikace je zaměřena především na zařízení s většími obrazovkami (tablety, počítače), u menších zařízení (mobilní telefony) se mohou uživatelé setkávat s grafickými problémy (více v podkapitole 6.3.1 Pilotní testování v následující kapitole).

Druhým výstupem je externí knihovna pro generování matematických příkladů ve zvolených tématech psaná v jazyce Dart. Tato knihovna byla vytvořena především pro účely výše zmíněné aplikace, nicméně je možné ji použít v budoucnu v jiných projektech.

# Kapitola 6

# Testování aplikace

Předmětem této kapitoly je popis všech úrovní testování, které aplikace podstoupila. Jedná se o testování knihovny pro generování matematických příkladů pomocí jednotkových testů, dále pak integrační testy zaměřené na ukládání a načítání dat aplikací.

Následovalo uživatelské testování, kdy byla aplikace nejdříve podrobena průzkumu lidmi, kteří nejsou cílovou skupinou - pilotní testování, a později podle zadání práce také testování žáky 9. třídy na vybrané základní škole. Jak jednotlivé úrovně testování probíhaly, jejich výsledky a vliv na aplikaci je popsáno v následujících podkapitolách.

## 6.1 Unit testy

Unit testy, nebo také jednotkové testy, se zabývají právě jednou funkcí většího systému. V případě našeho projektu se jednalo o testování výstupů knihovny pro generování matematických příkladů. Testování přímo v adresáři knihovny nám umožnila knihovna **test**[1] pro jazyk Dart. Cílem této úrovně testování bylo především ověřit správnost výstupů, tedy zdali má vygenerovaný příklad přiřazený správný výsledek. Dále také byly pro jednoduchost práce ověřovány výstupy na očekávané formáty. Pro aplikaci bylo rozhodnuto, že uživatelé budou zadávat výsledky pouze jako celá čísla nebo zlomky v základním tvaru, takže bylo nutné ověřit, že výstupy knihovny, objekty třídy MathProblem, budou vždy mít výsledky uložené v odpovídajícím formátu.

## 6.2 Integrační testy

Integrační testování se provádí na větší funkční části aplikace narozdíl od jednotkových testů. V případě naší aplikace se jedná o testy zaměřené na ukládání hodnot do paměti zařízení, tedy části aplikace, které využívají knihovny **shared_preferences** a **path_provider**. Testy jsou implementovány pomocí knihovny **integration_test**[2], k jejich spouštění je ale také nutné mít nainstalovaný nástroj pro automatizované testování **Chromedriver**[3]. Samotné testování pak probíhá v okně prohlížeče Google Chrome, instrukce pro spouštění jsou uvedeny v příslušném souboru .dart.

[1] pub.dev/packages/test
[2] github.com/flutter/flutter/tree/main/packages/integration_test
[3] chromedriver.chromium.org

## 6.3 Uživatelské testování

Testování uživateli bylo rozděleno do dvou fází. První fází bylo testování lidmi, kteří nepatří do cílové skupiny, a bylo především orientováno na otestování srozumitelnosti ovládání aplikace a obecnou kontrolu lekcí, příkladů apod. Druhá fáze pak probíhala během 2 vyučovacích hodin matematiky žáků 9. třídy. Jejím předmětem bylo posouzení přínosnosti aplikace žáky, tedy cílovou skupinou. Vzhledem k jednoduchosti aplikace, v obou případech nedostali testující žádné testovací scénáře, pouze jen obecné pokyny k ovládání aplikace a krátký popis toho, co aplikace umí.

### 6.3.1 Pilotní testování

Pilotním testováním je myšleno testování, které probíhalo po dokončení každé větší části aplikace. Hlavním cílem bylo najít problémy spojené s navigací aplikací a srozumitelností jednotlivých funkcionalit, které aplikace obsahuje. Také sem patří kontrola kvality lekcí a generovaných příkladů. Podle zpětné vazby testujících byly přidány různé nápovědy, např. vysvětlení milníků, které odemykají další lekce. Také byla rozšířena stránka "O aplikaci"o hlubší popis obsahu aplikace. Dále se jednalo o opravu chybných výsledků u ručně vytvořených příkladů v lekcích či různé opravy pravopisu ve slovních úlohách.

### 6.3.2 Testování cílovou skupinou

Finální testování probíhalo na základní škole v Červené Vodě, kdy nám byly po předchozí domluvě poskytnuty celkem 2 po sobě jdoucí vyučovací hodiny matematiky žáků 9. třídy. Testování se účastnili všichni žáci, kteří se na jaře 2024 chtějí účastnit jednotných přijímacích zkoušek, celkem tedy 20 žáků. K dispozici měli tablety s operačním systémem iOS na kterých ale spouštěli aplikaci v prohlížeči Safari. Tím jsme se, i přes absenci vývojářské licence pro distribuci přes App Store, snažili simulovat chování aplikace na dotykovém zařízení.

Žákům byly skrze projektor v místnosti na začátku první hodiny stručně ukázány obsah a ovládání aplikace, následně pak měli po zbytek hodiny volný výběr toho, co chtějí v aplikaci dělat. Nicméně jim bylo doporučeno, aby si nejdříve vyzkoušeli několik lekcí a až poté přešli na počítání generovaných příkladů. Druhá hodina pak byla celá věnována cvičnému testu, který v aplikaci má časovač celkem na 40 minut. Kdo jej odevzdal dříve, měl opět v aplikaci volnou ruku.

Sběr zpětné vazby probíhal nejenom během komunikace se žáky během testování, ale také pomocí krátkého dotazníku na přínosnost aplikace, který žáci vyplnili na konci (viz níže). Navíc pomocí anonymního přihlášení také aplikace během testování sbírala statistiky o plnění lekcí, generovaných příkladech, úspěšnosti žáků při plnění cvičného testu a dalších.

Hlavním odhaleným problémem během této fáze bylo **mizení zapsaných výsledků** ve chvíli, kdy byl příklad scrollováním odstraněn z obrazovky. Ačkoliv bylo vložené číslo uloženo, tak samotné textové pole jej při opětovném zobrazení příkladu již neobsahovalo. Pokud pak žák znovu chtěl psát do zdánlivě již vyplněného pole, tak byl předchozí výsledek kompletně vymazán. Příčinu tohoto problému jsme byli schopni najít již na místě, ale její odstranění nebylo v rámci času vymezeného na testování možné. Tuto komplikaci obsahovaly jak kontrolní testy v lekcích, tak i cvičný test na čas. Ačkoliv tak žáci mohli všechny příklady splnit, jednalo se o nepříjemnou

komplikaci. Důvodem, proč tento problém nebyl dříve odhalen, je pravděpodobně fakt, že dřívější testování bylo prováděno lidmi, kteří s příklady neměli většinou problémy, takže příklady řešili postupně a nevraceli se k těm přeskočeným, čímž by tedy odhalili chybějící již dříve zapsané výsledky v textových polích.

Tento problém se neobjevil u řešení sad vygenerovaných příkladů, kde byly hodnoty vložené do textových polí drženy v individuálních proměnných pro každé různé textové pole. Toto řešení bylo později aplikováno i do problémových sekcí a tento problém byl vyřešen.

Podobně jako u pilotního testování, další výraznou částí zpětné vazby byla **nejednoznačnost zadávání výsledků**. Žáci jsou totiž aktivně instruováni nejen převádět zlomky do základního tvaru, ale také dále na smíšená čísla, je-li to možné. Takový zápis by byl ale problémový pro jednoduché textové pole, proto byly přidány nové nápovědy do všech příkladů, které jasně vyznačují, že výsledky příkladů aplikace akceptuje pouze jako celá čísla nebo zlomky v základním tvaru (tzn. 'a/b'). S tím souvisí i poslední poznatek - u rovnic a slovních úloh byly nově, po obdržené zpětné vazbě, přidána další upozornění, že výsledek by měl být uváděn pouze jako číslo, tedy bez jednotek (pro slovní úlohy) nebo bez 'x = ...' (u rovnic).

Ačkoliv žáci tento problém nezmiňovali, a ani se zmínka tohoto problému neobjevila v níže zmíněném dotazníku, tak po konci testování žáky jsme také do aplikace přidali **potvrzovací dialog** pro vyhodnocování příkladů. Ten se zobrazí uživateli ve chvíli, kdy stiskne na tlačítko "Potvrdit"- tedy chce odevzdat své výsledky k vyhodnocení. Několik žáků totiž omylem odevzdalo výsledky dříve než si stihli projít všechny příklady v testu.

Ze zpětné vazby sbírané aplikací jsme zjistili, že 16 žáků ze 20 poctivě splnilo alespoň 1 lekci, někteří až 4. U generovaných sad příkladů bylo u všech témat zaznamenáno alespoň 10 vygenerovaných příkladů, kromě úloh na procenta, která si, alespoň zde, nevyzkoušel nikdo. Naštěstí jsou ale v počtech po 3 vždy součástí cvičného testu, takže se žáci setkali i s tímto tématem. I přes problémy se zadáváním výsledků všichni žáci odevzdali po 40 minutách práce cvičný test. V průměru se úspěšnost řešení příkladů ve cvičném testu pohybovala okolo 35%, nejlepší pak byli tři žáci s 55%, tedy 6 správných výsledků z 11. Důvodů poměrně nízké úspěšnosti může být několik, ať už se jedná o výše zmíněné problémy s mizením výsledků, nebo také zkreslení statistiky spuštěním více testů (žáci, kteří omylem odevzdali test dříve, většinou spustili test nový a jejich výsledná úspěšnost tak ve výsledku byla kombinací obou testů). Také je nutné vzít v potaz, že v době řešení testu už měli za sebou takřka hodinu počítání příkladů, takže i únava se mohla projevit na úspěšnosti řešení.
Výše zmíněný dotazník obsahoval celkem 5 otázek:

1. Na otázku, zdali se během testování setkali žáci s nějakým problémem, odpovědělo celkem 12 z 20 ano. Popis problému byl předmětu druhé otázky.

2. Každý, kdo odpověděl ano v první otázce, pak zde zmiňoval mizející výsledky. Jeden žák pak také zmínil chybné vyhodnocování výsledků rovnic (tzn. odpověď "x = 10"byla i přes správný výsledek "10"neuznána).

3. V třetí otázce ohledně používání aplikace i mimo toto testování odpověděli 2 žáci "Ano", 12 "Možná", 4 "Nevím"a celkem 2 odpověděli "Ne".

4. U otázky zda-li žákům něco v aplikaci chybělo, tak 2 zmínili úpravu výrazů.

5. Poslední otázka byla ponechána volná na další případné poznámky. Zde byly právě zmíněna smíšená čísla, ale objevily se také pochvaly pro aplikaci. Za zmínku také stojí absence rozdělení složitostí u slovních úloh a úloh na procenta, u těchto témat je ale zrovna toto obtížné hodnotit v porovnání s ostatními.

Vzhledem k otázce 3 je zajímavé zmínit, že k 31.12.2023 zaznamenala služba Firebase ode dne testování 8 nových stažení hostovaného obsahu. Je tak možné, že někteří žáci, případně učitelé, kteří byli u testování přítomni, si aplikaci sami otevřeli.

## 6.4 Shrnutí testování

Jednotkové testy byly zaměřeny na naši knihovnu pro generování matematických příkladů. Především tedy na správnost výsledků u vygenerovaných příkladů a jejich správný formát. Integrační testy naopak probíhaly na částech samotné aplikace, které ukládají či načítají data z paměti. Realizovány byly ve formě automatizovaného testování v prohlížeči.

Cílem uživatelského testování bylo získat zpětnou vazbu ohledně navigování aplikací a jejího ovládaní. Další částí zpětné vazby bylo také ověřit přínosnost aplikace z pohledu cílové skupiny, tedy žáků, kteří se připravují na přijímací zkoušky z matematiky na střední školy. Odhaleno bylo několik problémů se srozumitelností v interakci s aplikací (např. formát výsledků) a pak jeden větší technický problém související s odevzdáváním výsledků příkladů v aplikaci. Na základě získané zpětné vazby pak byly tyto problémy vyřešeny. V aplikaci tak například přibylo několik dalších pomocných oken a dalších prvků, které zmiňované nejasnosti řeší.

# Závěr

Tato bakalářská práce měla několik základních cílů. Prvním z nich byl rozbor základního vzdělávání v oblasti matematiky a analýza tématu rovnic, se kterými se žáci devátých tříd mohou setkat u jednotných přijímacích zkoušek z matematiky. V kapitole Přehled vzdělávání jsme představili základní dokumenty MŠMT, které výuku matematiky na základních školách vymezují a stanovují požadavky, které by žák při absolvování základního vzdělávání měl splňovat. Ty se poté odrážejí na obsahu přijímacích testů na střední školy. Nejdůležitější součástí této kapitoly je pak sekce Rámcové vzdělávací programy - RVP, která je věnována dokumentu stejného jména, a obsahuje právě výčet standardů a požadavků, které jsou při zaměření se na téma rovnic, pro nás podstatné. Pomocí nich pak můžeme definovat obsah, který budeme pro náš systém vytvářet.

Druhým cílem byla analýza současného stavu konkurence pro přípravu na přijímací zkoušky z matematiky, ať už se jedná o webové stránky či mobilní aplikace. Kapitola Dostupná řešení tak obsahuje seznam a porovnání nejen českých, ale i zahraničních webů a aplikací, které žáci mohou využívat. Vlastnosti všech platforem ze seznamu jsou porovnány v přiložené tabulce v podkapitole Shrnutí. Výstupem této části práce jsou především možné inspirace pro návrh vlastní výukové platformy. Závěr této kapitoly také obsahuje analýzu relevantních typů příkladů z archivních přijímacích testů společnosti Cermat z let 2015 až 2023. Tato sekce pak slouží k definování části systému, která bude uživatelům nabízet procvičování matematických příkladů.

Další cílem byl návrh a realizace vlastního systému, který by měl usnadnit žákům přípravu na přijímací zkoušky. Nabízet by měl nejen výuku jednotlivých témat, ale také jejich procvičování. V kapitole Analýza jsme nejdříve určili požadavky na funkcionalitu systému a to jak ve formě funkčních a nefunkčních požadavků, tak i jako diagram případů užití. Následující kapitola Návrh řešení pak obsahuje samotný technický návrh systému. Obsahuje výběr formy aplikace, přehled a porovnání technologií, které bychom k jejímu vývoji mohli použít, dále podle vybraných technologií také návrh architektury, diagramy nasazení a hierarchie obrazovek v aplikaci a nakonec grafický návrh uživatelského rozhraní vypracovaný v aplikaci Figma.

Popis systému a práce na něm je předmětem kapitoly Technologické provedení. V ní jsme shrnuli jak aplikace ve frameworku Flutter vypadá a jak jsme postupovali při vytváření vlastní knihovny pro generování matematických příkladů ve vybraných tématech v jazyce Dart. Kapitola obsahuje nejen popisy jednotlivých částí aplikace a knihovny spolu s krátkými ukázkami kódu, ale také seznam použitých knihoven. Poslední částí sekce o vývoji aplikace je připojení backendové služby Firebase k naší aplikaci. Součástí kapitoly Technologické provedení je také popis nasazení aplikace, ať už ve formě webové aplikace, nebo mobilní pro operační systémy Android. V závěru kapitoly jsme také uvedli rozdíly současného stavu aplikace oproti návrhu,

a to jak z technického hlediska, tak i vizuálního. Také jsme porovnali současný stav obsahu oproti tomu plánovanému.

Poslední kapitola, Testování aplikace, odpovídá poslednímu zadanému cíli práce a to je testování aplikace, ať už se jedná o interní testování či uživatelské testování. Popsány jsou zde jednotkové a integrační testy, které byly nasledovány pilotním testováním a nakonec testováním žáky základní školy v Červené Vodě během 2 vyučovacích hodin matematiky. Pro každou úroveň testování je zde uveden způsob a především u testování na základní škole je přiložena zpětná vazba žáků spolu s důsledky, které pro aplikaci měla. Jedná se například o opravení nalezených chyb či dodatečné přidání popisků a dalších pomocných elementů do aplikace.

Teoretickými výstupy práce jsou tak analýza výuky tématu rovnic během základního vzdělávání, dále analýza dostupných platforem pro mimoškolní studium matematiky v rámci přípravy na přijímací zkoušky a návrh vlastního výukového systému zaměřeného na téma rovnic. Návrh také obsahuje analýzu dostupných technologií pro multiplatformní vývoj, v našem případě primárně pro operační systémy Android a iOS, spolu s nasazením na web. Prvním praktickým výstupem práce je aplikace vytvořená pomocí frameworku Flutter a nasazená pomocí služby Firebase Hosting na web a také zkompletována jako instalační soubor pro operační systémy Android. Tato aplikace nabízí výuku vybraných témat, od příkladů na celá čísla a zlomky až po rovnice a slovní úlohy, které se objevují u přijímacích zkoušek, a procvičování témat z lekcí pomocí generovaných příkladů podle uživatelem zadaných parametrů. Druhým výstupem, který byl právě pro účely generování příkladů vytvořen, je externí knihovna v jazyce Dart, která nabízí generování příkladů v celkem 6 oblastech matematiky v několika úrovních obtížnosti, tedy kromě slovních úloh. Náročnost generovaných úloh je přiměřená obtížnosti příkladů, které se objevují u přijímacích zkoušek - více v sekci Analýza obsahu testů.

V případě pokračování práce by hlavním cílem bylo plné propojení backendové služby Firebase s aplikací, služba totiž prozatím hlavně slouží pro sběr zpětné vazby. Dále by bylo vhodné rozšířit knihovnu o generování příkladů na úpravu výrazů, které ač s tématem rovnic souvisí, zatím knihovna neobsahuje. Do knihovny lze také přidat mnoho dalších zadání slovních úloh a obecně rozšířit nabídku obtížností pro všechny typy příkladů. V poslední řadě by bylo také vhodné aplikaci plně přizpůsobit používání na menších obrazovkách. Vzhledem k zaměření se na testování na základní škole, kde byly dostupné tablety od společnosti Apple a jako případná záloha mohly být použity počítače v místní učebně informatiky, byl responzivní design upozaděn v prospěch plné funkčnosti aplikace na větších obrazovkách.

# Bibliografie


[1] *§59, §60 561/2004 Sb. Školský zákon, Přijímání ke vzdělávání ve střední škole*. 2023. URL: https://www.zakonyprolidi.cz/cs/2004-561#f2874096 (cit. 16. 12. 2023).

[2] Baisden A. *Flutter vs. React Native*. 2022. URL: https://blog.logrocket.com/react-native-vs-flutter/ (cit. 06. 11. 2023).

[3] Griffith C. *What is Apache Cordova Framework and What is the Difference from PhoneGap?* 2023. URL: https://ionic.io/resources/articles/what-is-apache-cordova (cit. 06. 11. 2023).

[4] Cloudflare. *What is BaaS?* 2023. URL: https://www.cloudflare.com/learning/serverless/glossary/backend-as-a-service-baas/ (cit. 30. 11. 2023).

[5] Asiuwhu E. *NativeScript vs. React Native*. 2021. URL: https://blog.logrocket.com/nativescript-react-native/ (cit. 06. 11. 2023).

[6] edu.cz. *RVP ZV - Rámcový vzdělávací program pro základní vzdělávání*. 2022. URL: https://www.edu.cz/rvp-ramcove-vzdelavaci-programy/ramcovy-vzdelavacici-program-pro-zakladni-vzdelavani-rvp-zv// (cit. 04. 05. 2023).

[7] edu.cz. *Vznik Strategie 2030+*. 2021. URL: https://www.edu.cz/strategie-msmt/s2030/vznik-strategie-vzdelavaci-politiky-cr-2030/ (cit. 04. 05. 2023).

[8] Lardinois F. *Google makes Kotlin a first-class language for writing Android apps*. 2017. URL: http://tcrn.ch/2rfYPCd (cit. 06. 11. 2023).

[9] Kotlin Help. *Kotlin Multiplatform Documentation*. 2023. URL: https://kotlinlang.org/docs/multiplatform.html (cit. 06. 11. 2023).

[10] *How SCADE works*. 2021. URL: https://docs.scade.io/docs/how-scade-works (cit. 06. 11. 2023).

[11] Hickson I. *sharedpreferences package*. 2023. URL: https://pub.dev/packages/shared_preferences (cit. 27. 11. 2023).

[12] Patadiya J. *React vs React Native: Which One to Choose and Why?* 2023. URL: https://radixweb.com/blog/react-vs-react-native (cit. 06. 11. 2023).

[13] Harsh K. *Flutter vs. Xamarin*. 2022. URL: https://blog.logrocket.com/flutter-vs-xamarin/ (cit. 06. 11. 2023).

[14] Nayak S. K. *Flutter Development: VS Code vs. Android Studio*. 2023. URL: https://shivamkumarnayak.medium.com/flutter-development-vs-code-vs-android-studio-9e93f4917cc (cit. 01. 12. 2023).

[15] Blahošová J. a kol. *Jak čeští adolescenti používají své mobily? Analýza dat z chytrých telefonů*. Brno: Masarykova univerzita, 2023. URL: https://irtis.muni.cz/media/3524142/wp4_report_jak-cesti-adolescenti-pouzivaji-sve-mobily.pdf (cit. 24. 05. 2023).

[16] Dvořáková J. a kol. *Standardy pro základní vzdělávání - Matematika a její aplikace*. Ministerstvo školství, mládeže a tělovýchovy České republiky, 2022. URL: https://digifolio.rvp.cz/artefact/file/download.php?file=67490&view=9832 (cit. 04. 05. 2023).

[17] Fryč J. a kol. *Strategie vzdělávací politiky ČR do roku 2030+*. Ministerstvo školství, mládeže a tělovýchovy České republiky, 2020. URL: https://www.msmt.cz/vzdelavani/skolstvi-v-cr/strategie-2030 (cit. 04. 05. 2023).

[18] Kotásek J. a kol. *Bílá kniha - Národní program rozvoje vzdělání v ČR 2002*. Ministerstvo školství, mládeže a tělovýchovy České republiky, 2013. URL: https://www.msmt.cz/vzdelavani/skolstvi-v-cr/bila-kniha-narodni-program-rozvoje-vzdelani-v-cr (cit. 04. 05. 2023).

[19] Morgan B. a kol. *FAQ | Flutter*. 2023. URL: https://docs.flutter.dev/resources/faq (cit. 06. 11. 2023).

[20] Petrovskaya V. a kol. *Flutter App Backend: Find the Best Options*. 2023. URL: https://blog.flutter.wtf/flutter-app-backend/ (cit. 30. 11. 2023).

[21] Skuza B. a kol. *Flutter vs React Native – Which is Better for Your Project?* URL: https://www.thedroidsonroids.com/blog/flutter-vs-react-native-comparison (cit. 06. 11. 2023).

[22] Havrlant L. *Učebnice matematiky — Matematika polopatě*. 2008. URL: https://www.matweb.cz (cit. 04. 05. 2023).

[23] IT Creative Labs. *Flutter vs. React Native in 2023: Which is Better for Mobile App Development?* URL: https://www.linkedin.com/pulse/flutter-vs-react-native-2023-which-better-mobile-app/ (cit. 06. 11. 2023).

[24] Lewandowski M. *Kotlin Multiplatform vs Flutter. Which Is Better for Your App?* 2023. URL: https://www.netguru.com/blog/kotlin-multiplatform-vs-flutter (cit. 30. 11. 2023).

[25] Buzdugan S. *Why Flutter Goes Best with Firebase*. 2023. URL: https://medium.com/@sebizaur/why-flutter-goes-best-with-firebase-13871e4cf468 (cit. 30. 11. 2023).

[26] *Specifikace požadavků k jednotné přijímací zkoušce*. 2023. URL: https://prijimacky.cermat.cz/menu/specifikace-pozadavku-k-jpz (cit. 16. 12. 2023).

[27] *Swift Language*. 2023. URL: https://developer.apple.com/swift/ (cit. 06. 11. 2023).

[28] vlada.cz. *Výsledky jednání vlády 19. října 2020*. 2020. URL: https://www.vlada.cz/cz/media-centrum/tiskove-zpravy/vysledky-jednani-vlady--19--rijna-2020-184316/ (cit. 15. 05. 2023).

[29] zakonyprolidi.cz. *Zákon č. 178/2016 Sb. - o předškolním, základním, středním, vyšším odborném a jiném vzdělávání (školský zákon)*. 2016. URL: https://www.zakonyprolidi.cz/cs/2016-178 (cit. 21. 05. 2023).

# Přílohy

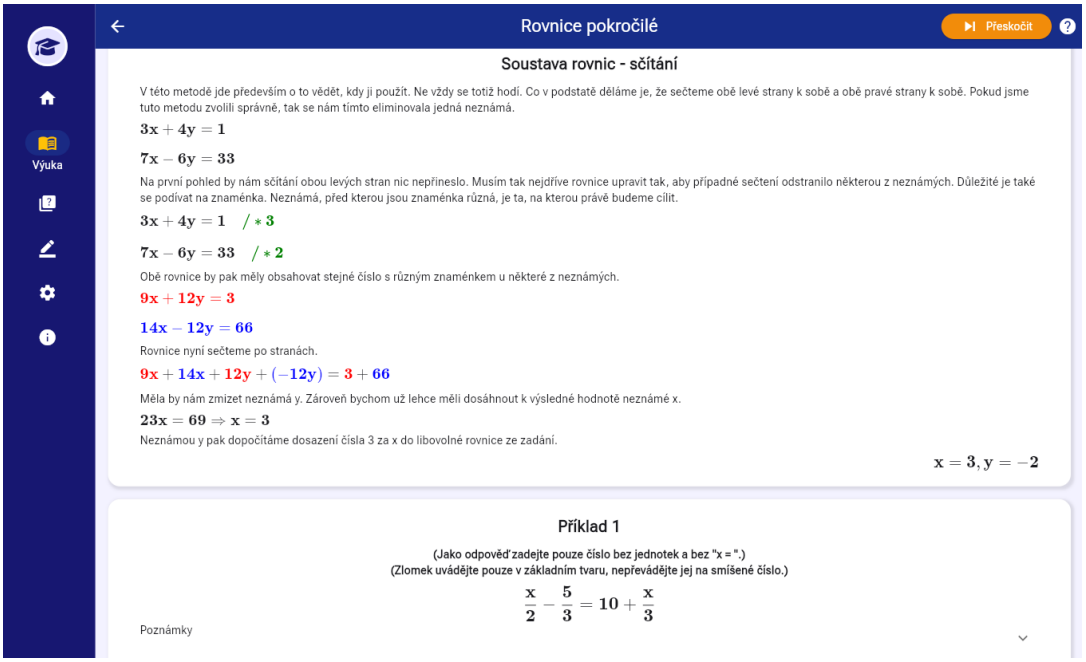
## A Zdrojové adresáře

```
exam_app ..... adresář aplikace
├── ...
├── android ..... soubory pro Android build
├── ios ..... soubory pro iOS build
├── lib
│   ├── pages ..... zdrojový kód hlavních obrazovek
│   │   └── lessons ..... zdrojový kód obrazovek lekcí
│   ├── util_classes ..... zdrojový kód pomocných tříd
│   ├── util_widgets ..... zdrojový kód vlastních widgetů
│   ├── firebase_options.dart ..... nastavení Firebase připojení
│   └── main.dart ..... spouštěcí soubor aplikace
├── public ..... složka pro výstup buildu
├── test ..... integrační testy
│   └── test_driver ..... nastavení propojení s Chromedriverem
├── text_data ..... zdrojové texty lekcí
├── web ..... soubory pro build webové aplikace
├── ...
├── README.md ..... popis aplikace
├── firebase.json ..... nastavení Firebase
├── firestore.rules ..... pravidla databáze Firestore
└── pubspec.yaml ..... metadata projektu (použité knihovny apod.)

exam_app_generator ..... adresář knihovny
├── ...
├── example ..... ukázka použití
├── lib
│   ├── src ..... zdrojový kód generátorů
│   │   └── util ..... zdrojový kód pomocných tříd
│   └── exam_app_generator.dart ..... soubor pro export zdrojového kódu
├── test ..... jednotkové testy
├── ...
├── README.md ..... popis knihovny
└── pubspec.yaml ..... metadata projektu (použité knihovny apod.)

sbirka_prikladu.pdf ..... sbírka příkladů z přijímacích zkoušek
```

# B Ukázky dalších obrazovek aplikace




**Obrázek 1:** Screenshot obrazovky lekce o rovnicích


**Obrázek 2:** Screenshot domovské obrazovky


**Obrázek 3:** Screenshot obrazovky stránky O aplikaci

# C Důležité odkazy

- Adresa webové aplikace - mathexamapp.web.app
- Repozitář s apk soubory - github.com/severfrant/exam_app_distribution