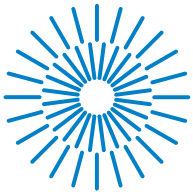
FAKULTA
PŘÍRODOVĚDNĚ-HUMANITNÍ
A PEDAGOGICKÁ TUL



Bakalářská práce

Mobilní aplikace na procvičování slovní zásoby k učebnici anglického jazyka Project explore 1

Studijní program: B0114A300065 Informatika se zaměřením na vzdělávání

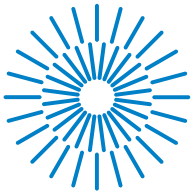
Studijní obory: Informatika se zaměřením na vzdělávání
Anglický jazyk se zaměřením na vzdělávání

Autor práce: **Benjamin Boš**

Vedoucí práce: Ing. Jindra Drábková, Ph.D.
Katedra geoinformatiky a didaktiky informatiky

Liberec 2023

FAKULTA
PŘÍRODOVĚDNĚ-HUMANITNÍ
A PEDAGOGICKÁ TUL



Zadání bakalářské práce

Mobilní aplikace na procvičování slovní zásoby k učebnici anglického jazyka Project explore 1

Jméno a příjmení:	**Benjamin Boš**
Osobní číslo:	P19000023
Studijní program:	B0114A300065 Informatika se zaměřením na vzdělávání
Specializace:	Informatika se zaměřením na vzdělávání Anglický jazyk se zaměřením na vzdělávání
Zadávající katedra:	Katedra aplikované matematiky
Akademický rok:	2021/2022

Zásady pro vypracování:

Cílem práce je vytvořit mobilní aplikaci pro operační systém Android, která by měla studentům pomoci s procvičováním slovní zásoby a jejím následným zapamatováním. V teoretické části student popíše některé nástroje, které by bylo možné použít k vytváření této aplikace. Dále porovná různé metody procvičování slovní zásoby na digitálním zařízení a popíše specifika výukové aplikace. V praktické části student vytvoří mobilní aplikaci pro operační systém Android na procvičování slovní zásoby, ve které bude možno alespoň třemi různými způsoby procvičovat slovní zásobu obsaženou ve slovníku pracovního sešitu k učebnici Project explore 1. Student vytvořenou aplikaci otestuje a upraví na základě zpětné vazby.

Rozsah grafických prací:	podle potřeby
Rozsah pracovní zprávy:	40
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	čeština

Seznam odborné literatury:

PHILLIPS, Sarah, Paul SHIPTON, Michaela TRNOVÁ a Amanda BEGG. Project explore 1. Oxford: Oxford University Press, 2019. ISBN 978-0-19-425574-5
Dart [online]. Google [cit. 28. 4. 2022]. Dostupné z: https://dart.dev
Flutter [online]. Google [cit. 28. 4. 2022]. Dostupné z: https://flutter.dev
MATYASOVA, Jana. Design výukových nástrojů. *EDTECH KISK* [online]. 2020 [cit. 3. 6. 2021]. ISSN ISSN 2570-9364. Dostupné z: https://medium.com/edtech-kisk/design-
-v%C3%BDukov%C3%BDch-n%C3%A1stroj%C5%AF-9ccbb9da9b94

Vedoucí práce:	Ing. Jindra Drábková, Ph.D. Katedra geoinformatiky a didaktiky informatiky
Datum zadání práce:	27. května 2022
Předpokládaný termín odevzdání:	30. července 2023

L.S.

prof. RNDr. Jan Picek, CSc.
děkan

doc. RNDr. Miroslav Koucký, CSc.
vedoucí katedry

V Liberci dne 6. června 2022

Prohlášení

Prohlašuji, že svou bakalářskou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědom toho, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

26. srpna 2023 Benjamin Boš

Poděkování

Rád bych poděkoval vedoucí mé bakalářské práce paní doktorce Jindře Drábkové za její nekonečnou trpělivost a spoustu času věnovanému této práci. Dále bych chtěl poděkovat všem účastníkům testování aplikace, a to jak pedagogům, tak žákům.

Anotace

Bakalářská práce, která nese název „Mobilní aplikace na procvičování slovní zásoby k učebnici anglického jazyka Project explore 1“, je zaměřena na vývoj mobilních aplikací.

V teoretické části práce jsou rozebrány přístupy k vývoji mobilní aplikace a jsou v ní popsány vybrané nástroje pro různé způsoby vývoje. Dále jsou v ní popsány různé metody procvičování slovní zásoby na digitálním zařízení.

Praktická část práce popisuje postup vývoje mobilní aplikace na procvičování slovní zásoby k učebnici anglického jazyka Project explore 1. Je v ní detailně popsán návrh, tvorba a testování aplikace.

Klíčová slova

mobilní zařízení, programový kód, aplikace, vývoj, vývojový nástroj, uživatel, Android, slovíčko, učebnice

Annotation

This bachelor's thesis called „Mobile Application to learn vocabulary from English textbook Project explore 1" deals with the development of mobile applications.

The theoretical part of the thesis discusses various approaches to mobile Application development and describes selected tools for various ways of development. It also describes various methods of learning vocabulary on a digital device.

Practical part of the thesis describes progress of mobile application development. It describes in detail the design, creation and testing of the application.

Key words

mobile device, programming code, application, development, development tool, user, Android, word, textbook

Obsah

Seznam obrázků, tabulek a grafů

Seznam zkratek a symbolů

API – rozhraní pro programování aplikací

App – aplikace

BIOS – základní systém vstupu a výstupu

CLI – rozhraní pro příkazový řádek

IDE – integrované vývojové prostředí

OS – operační systém

PWA – progresivní webová aplikace

QWERTY – způsob rozložení klávesnice

SDK – sada nástrojů pro vývoj softwaru

Úvod

Učení slovíček je nedílnou součástí osvojování cizího jazyka. Bez dostatečné slovní zásoby není možné začít v cizím jazyce komunikovat. Učitelé cizích jazyků bojují s tím, jak žáky to velké množství slovíček stihnout naučit. Protože časová dotace cizích jazyků je nedostačující, většinou se uchylují k zadávání domácích úkolů. Žáci dostanou za úkol se doma naučit určité množství slovíček. Učení slovíček pouze čtením z učebnice je nedostačující, a tak je vhodné použít nějakou pomůcku. Protože většina dnešních žáků vlastní mobilní zařízení, nabízí se možnost vytvořit aplikaci pro procvičování slovíček na tomto mobilním zařízení, která by jim usnadnila a zároveň zpříjemnila práci.

Cílem této práce je vytvořit mobilní aplikaci pro operační systém Android, která má studentům pomoci s procvičováním slovní zásoby z učebnice anglického jazyka Project explore 1. Tvorba aplikace je nekonečný proces inovací. Cílem práce je tedy vytvořit první verzi této aplikace a její další vývoj a budoucí plány jsou popsány v závěrečné části této práce.

Teoretická část práce popisuje jednotlivé přístupy k vývoji mobilní aplikace a popisuje vybrané nástroje, pomocí nichž je možné mobilní aplikace vytvářet. Dále jsou v ní rozebrány různé metody procvičování slovní zásoby na mobilním zařízení.

V praktické části této práce je popsán vývoj mobilní aplikace. Detailně jsou v ní rozebrány fáze návrhu, tvorby, testování a ladění aplikace. Testování aplikace je provedeno ve dvou fázích.

1 Nástroje pro vývoj mobilních aplikací

V této kapitole jsou rozebrány různé nástroje pro tvorbu mobilních aplikací. Nejdříve jsou definovány pojmy mobilní aplikace a mobilní zařízení. Poté je definováno rozdělení mobilních aplikací z hlediska jejich vývoje a dále jsou popsány nejpoužívanější programovací jazyky a vývojové nástroje pro tvorbu mobilních aplikací.

1.1 Mobilní zařízení

Committee on National Security Systems definuje mobilní zařízení jako přenosné výpočetní zařízení menších rozměrů s následujícími vlastnostmi:

- může být přenášeno jedním člověkem,
- funguje bez fyzického připojení (disponuje bezdrátovým připojením pro vysílání nebo přijímání informací),
- má vlastní odpojitelné/integrované paměťové úložiště,
- obsahuje vlastní napájecí zdroj.

Mobilní zařízení může být schopné zprostředkovat hlasovou komunikaci, mít vestavěné senzory pro snímání informací nebo být schopné synchronizace vlastních dat se vzdáleným úložištěm. Například se jedná o chytré telefony, tablety nebo elektronické čtečky. Jiný název pro mobilní zařízení je přenosné výpočetní zařízení [1, s. 133].

Dle National Institute of Standards and Technology je mobilní zařízení definováno jako malé přenosné zařízení s obrazovkou, které je ovládané dotykem a/nebo QWERTY klávesnicí. Toto zařízení může poskytovat telefonní služby [2, s. 69].

Vzhledem ke skutečnosti, že tyto definice zahrnují širší okruh zařízení včetně například notebooků nebo elektronických čteček, je třeba si tuto definici pro budoucí potřeby této práce ještě více zúžit. Protože cílem praktické části této práce je vytvořit mobilní aplikaci, kterou by mohli využívat žáci základní školy nejlépe kdekoli a kdykoli, je potřeba vzít v potaz hlavně ta mobilní zařízení, ke kterým má přístup většina z nich a mohou si je vzít s sebou téměř kamkoli. Taková zařízení jsou převážně chytré mobilní telefony, tablety a notebooky. A protože z hlediska přenosnosti jsou, díky své velikosti, nejpraktičtější chytré mobilní telefony, pokud je níže v práci zmiňováno mobilní zařízení, je pro potřeby této práce myšlen chytrý mobilní telefon (anglicky smartphone).

1.2 Mobilní aplikace

Mobilní aplikace je software, který je navržen pro provoz na mobilním zařízení (například chytrém telefonu nebo tabletu). Často poskytuje uživateli podobné služby, které jsou dostupné na osobním počítači. Tyto aplikace jsou obvykle menší individuální softwarové jednotky s omezenou funkcí. Mobilní aplikace je často označována zkráceně app [3].

Jak je popsáno výše, pro potřeby této práce je za mobilní zařízení považován chytrý mobilní telefon. Ty se ale liší operačním systémem, který je na nich nainstalován. Občas je termín operační systém nahrazován slovem platforma. Aplikace z jednoho mobilního operačního systému nefungují na jiném. Protože ústředním tématem práce je vývoj mobilních aplikací, je třeba definovat, pro jaké mobilní operační systémy je vývoj popisován.

Mezi nejznámější mobilní operační systémy patří Android, iOS, Tizen, Windows Phone a další operační systémy založené na Linuxu. K březnu roku 2023 Android zastává 65,14 % podílu trhu v Evropě, iOS zastává 34,3 %, Tizen 0,5 %, Windows Phone 0,01 % a Linux 0,01 % [4]. Vzhledem k tomu, že operační systémy Android a iOS dohromady zastávají přibližně 99 % z celého evropského trhu s mobilními zařízeními, je práce zaměřená na vývoj pro tyto dvě platformy.

1.3 Rozdělení aplikací z hlediska vývoje

Aplikace je možné z hlediska vývoje rozdělit na nativní, hybridní, multiplatformní a PWA. Nativní aplikace je vyvíjena pro jednu konkrétní platformu (například Android). Pokud je třeba aplikaci spustit na jiné platformě (například iOS), je nutné vytvořit úplně novou aplikaci. Hybridní aplikace využívají spojení nativních prvků a nástrojů mobilního zařízení a webových technologií (například HTML, CSS nebo JavaScript). Multiplatformní aplikace využívají, alespoň z většiny, společný programový kód, který je možné kompilovat pro více různých platforem. PWA neboli progresivní webové aplikace fungují jako běžná webová stránka, ale vzhledem a funkčností připomínají mobilní aplikace. PWA je možné spustit v telefonu nebo dokonce v některých případech stáhnout z obchodu s aplikacemi (například z Google Play na platformě android) [5]. Způsoby vývoje aplikací jsou podrobněji popsány v následující sekci *Vývojové nástroje*.

1.4 Vývojové nástroje

V této sekci je blíže popsané rozdělení vývoje mobilních aplikací a jsou zde rozebrány nejznámější vývojové nástroje a jazyky pro vývoj mobilních aplikací.

1.5 Nativní vývoj

Jak je již popsáno výše, nativní vývoj aplikace spočívá v tom, že aplikace je vyvíjena pouze pro jednu platformu (například pouze pro iOS). Pokud je třeba vytvořit tu samou aplikaci i pro jinou platformu, musí se s vývojem začít znovu a je třeba vytvořit aplikaci novou. Každá platforma má vlastní programovací jazyky pro nativní vývoj. Pro systém iOS je možné vytvářet aplikace například v programovacím jazyku Swift nebo Objective-C a pro systém Android se často používá jazyk Java nebo Kotlin [5].

Jednou z hlavních výhod nativního vývoje je přímý přístup k modulům, nástrojům a funkcím, které daná platforma nabízí, a to skrze takzvané API (rozhraní pro programování aplikací). U Androidu mezi takové funkce patří třeba fotoaparát, kalendář nebo také Google Pay (služba pro mobilní platby). Protože je aplikace vytvářena přímo pro konkrétní platformu, další výhodou je zpravidla vyšší výkon a lepší optimalizace. Na druhou stranu hlavními nevýhodami nativního vývoje je vyšší cena a vyšší časová náročnost. Pokud je třeba vytvářet aplikaci pro dvě platformy, je třeba dvou vývojářských týmů [6].

Dále je nutné zmínit, že vývoj pro Android má vlastní specifika. Zatímco iOS je operační systém vyvinutý společností Apple a je určen pouze pro jejich zařízení, Android je open-source operační systém, který je adaptován a upravován jednotlivými výrobci mobilních zařízení [5]. Tato různorodost zařízení, na kterých je instalován, a nadstaveb operačních systémů, komplikuje samotný vývoj a optimalizaci aplikací. Obecně je možné říct, že optimalizování aplikací pro iOS je jednodušší z důvodu menšího množství zařízení, pro které aplikaci vyvíjíme.

Nativní vývoj aplikace zvolí zadavatel, který buď potřebuje aplikaci pouze pro jeden operační systém, nebo mu záleží na výkonu a optimalizaci aplikace a nevadí mu si za to připlatit. Příkladem takových aplikací jsou třeba mobilní hry, u kterých záleží na výkonu a optimalizaci, aby bylo možné je hrát i na starších a méně výkonných zařízeních.

1.5.1 Java

Java je vyšší univerzální objektově orientovaný programovací jazyk. Aplikace vytvořené v Javě je možné spustit téměř na jakémkoli zařízení, například aplikaci vytvořenou pro operační systém Windows, je možné spustit i na jiném operačním systému. Java poskytuje mnoho programovacích nástrojů, díky kterým je programování jednodušší. Součástí Javy je spousta open-source knihoven, které zjednodušují práci vývojářům aplikací. Jako jeden z nejrozšířenějších programovacích jazyků má obrovskou vývojářskou komunitu, která může pomoci s hledáním řešení problémů, které mohou nastat během vývoje [7].

Vývoj mobilní aplikace je čistě teoreticky možný pouze s textovým editorem, ale pro značné zjednodušení práce je možné nainstalovat nějaké vývojové prostředí (dále jen IDE). Společnost Google, která je tvůrcem operačního systému Android, vytvořila vlastní IDE s názvem Android studio. Toto studio obsahuje vše potřebné pro vývoj aplikace. Součástí balíčku je editor programového kódu, šablony pro aplikace, nástroje pro vývoj v Javě (OpenJDK), testování a ladění kódu a emulátor mobilního zařízení. Android studio je dostupné pro počítače s operačním systémem Windows, Linux a macOS [8]. Pro vývoj mobilní aplikace psané v Javě existuje více vývojových prostředí. Jedno mají ale společné, usnadňují vývoj. Největší usnadnění poskytují šablony, které zpravidla obsahují kostru aplikace, kterou tvoří spousta již předpřipravených souborů obsahujících nezbytné součásti aplikace. Dalším velkým pomocníkem je emulátor mobilního zařízení, který umožňuje vytvořit několik virtuálních mobilních zařízení, na kterých je možné testovat a ladit aplikaci bez nutnosti připojování fyzického zařízení [8].

Pro tvorbu mobilní aplikace psané v Javě za použití Android studia je potřeba umět programovací jazyk Java, který slouží ke zpracování logiky programu (například, co se má stát, když uživatel zmáčkne tlačítko), a značkovací jazyk XML, který slouží k popsání rozložení (vzhledu) aplikace [8].

1.5.2 Kotlin

Kotlin je stejně jako Java vyšší univerzální objektově orientovaný programovací jazyk, který byl navrhnut tak, aby byl schopný spolupracovat s Javou. Programový kód Kotlinu je možné spouštět v Javě a naopak. Knihovny funkcí, frameworky a IDE vytvořené specificky pro Javu je možné využít při psaní programového kódu Kotlinu. Na rozdíl od Javy, která je vyvíjena i pro jiné operační systémy, je již od začátku specificky vyvíjen pro psaní aplikací pro operační

systém Android. Další jeho výhodou je zkrácení a zjednodušení zápisu programového kódu, který je v Javě zdlouhavý a složitý. Díky tomu je možné při využití Kotlinu šetřit čas a náklady potřebné na vývoj aplikace. Hlavní nevýhodou oproti Javě je menší vývojářská komunita, a tedy menší podpora při řešení problému při vývoji aplikace. Kromě nativních aplikací je možné Kotlin využít i pro multiplatformní vývoj [9]. Další výhodou Kotlinu je, že android v roce 2019 na konferenci Google I/O oznámil, že bude čím dál více „Kotlin-first“. Tím je míněno, že další vývoj operačního systému bude čím dál tím více uzpůsobován kompatibilitě s programovacím jazykem Kotlin [10].

Jak je již výše zmíněno Kotlin je plně kompatibilní s Javou a při vývoji aplikací v něm je možné využít některého IDE. I zde je možné využít například výše popsané Android studio. Na rozdíl od Javy je možné při vývoji využít Jetpack Compose, moderní nástroj pro tvorbu nativních uživatelských rozhraní, který opět oproti XML usnadňuje vývoj [10].

Vzhledem k výše popsaným výhodám Kotlinu, dává smysl zvolit pro vývoj nativních aplikací jazyk Kotlin z důvodu ušetření času a nákladů na vývoj. Další nespornou výhodou je kompatibilita a vzájemná spolupráce mezi jazyky Java a Kotlin, která umožňuje využití Kotlinu pro údržbu a úpravu již existujících aplikací napsaných v Javě.

1.5.3 Objective C

Objective C je univerzální objektově orientovaný programovací jazyk, který patří mezi vyšší programovací jazyky, ale jeho kódový zápis je pro počítač jednoduše zpracovatelný. Tato vlastnost ho dělá rychlým a výkonným. Objective C vychází z programovacího jazyka C a většina jeho funkcionalit a konceptů je použitelných i v něm. Oproti jazyku C podporuje veškeré koncepty objektově orientovaného programování, což jej dělá velice univerzálním programovacím jazykem [11].

Pro vývoj mobilní aplikace v Objective C je nutné využít IDE Xcode, které je vydáváno společností Apple, která je tvůrcem operačního systému iOS. Xcode je aplikace vytvořená pouze pro počítačový operační systém macOS, takže ji není možné nainstalovat a spustit na Linuxu nebo ve Windows. Veškeré potřebné nástroje včetně editoru kódu, nástrojů pro testování a ladění, emulátoru iOS zařízení a další jsou zde plně integrované. Jako důsledek je sice možné psát programový kód v jiném operačním systému než macOS, ale samotné vytvoření a testování aplikace je možná pouze na zařízení s operačním systémem macOS. Pro tvorbu a úpravu vizuální stránky aplikace slouží takzvaný Interface Builder (součást Xcode),

který obsahuje jednotlivé součásti uživatelského rozhraní, které je možné využít pro tvorbu aplikace [12].

Existují možnosti, jak spustit nebo zpřístupnit operační systém macOS na zařízení s operačním systémem Windows nebo Linux (například virtuální stroj nebo cloudové služby). Tyto řešení jsou však nepraktické a mnohdy složité, a proto nejsou popsány v této práci.

1.5.4 Swift

Swift je univerzální vyšší objektově orientovaný programovací jazyk vytvořený v roce 2014 společností Apple za účelem vývoje mobilních aplikací pro operační systém iOS. Swift si sám řídí správu paměti a díky tomu je jeho práce s pamětí vysoce efektivní. Po vzoru moderních jazyků je přehledný a má krátký zápis, tím pádem je jednodušší se v programovém kódu vyznat [11]. Jazyk je navržen tak, aby byl schopen spolupracovat s Objective C. Je ho možné použít k přidávání nových prvků a funkcionalit do projektu napsaném v Objective C [13].

Ve Swiftu, na rozdíl od Objective C, při tvorbě proměnných není třeba definovat datové typy, což přispívá k rychlejšímu psaní programového kódu, takto ale mohou vznikat těžce odhalitelné chyby z nepozornosti programátora. Protože Objective C má složitější zápis, psaní programového kódu ve Swiftu je rychlejší a efektivnější. Oba programové jazyky jsou považované za rychlé z hlediska výkonu, ale obecně je možné říci, že Swift je z těchto dvou výkonnější a tedy rychlejší. Objective C je přibližně o 30 let starší jazyk než Swift, díky tomu má větší vývojářskou komunitu. Swift na rozdíl od Objective C nepodporuje zpětnou kompatibilitu, proto je méně stabilní a při přechodu na novou verzi je nutné v programovém kódu zakomponovat příslušné změny [11]. Swift je v kontextu programovacích jazyků mladý ale moderní. Jeho obrovskou výhodou je přímý vývoj a podpora společností Apple, která je zároveň společností, co vyvíjí operační systém iOS. Jeho hlavní nevýhodou je zpětná nekompatibilita, která přidělává práci vývojářům při zavádění nových verzí.

Pro tvorbu aplikací v programovacím jazyce Swift je opět nejlepší využít IDE Xcode, jehož součástí jsou veškeré potřebné vývojové nástroje [13].

1.6 Multiplatformní vývoj

Většina zadavatelů mobilních aplikací potřebuje vytvořit jednu aplikaci pro více platforem (operačních systémů). Při multiplatformním vývoji můžou programátoři využít většinu programového kódu pro více operačních systémů. Tento přístup zadavateli aplikace ušetří čas

a peníze, protože na rozdíl od nativního vývoje není třeba dvou vývojářských týmů a zároveň sdílený programový kód může ušetřit velké množství času potřebného pro psaní kódu aplikace a její následnou údržbu (například přidávání nových funkcionalit). Další nespornou výhodou menšího množství programového kódu je méně potencionálních chyb, čehož výsledkem je snížení času potřebného pro testování a ladění programu. Jednou z hlavních nevýhod multiplatformního vývoje je slabší optimalizace a výkon oproti nativním aplikacím, což může mít negativní dopad na prožitek uživatele. S postupným vývojem multiplatformních nástrojů je možné dosáhnout zážitku srovnatelného s nativními aplikacemi [14]. Nástrojů pro multiplatformní vývoj je velké množství a stále přibývají. V dalších podsekcích jsou popsány dva, které patří mezi nejznámější a nejvyužívanější, a to React Native a Flutter [17].

Multiplatformní vývoj zvolí zadavatel, který potřebuje aplikaci pro více platforem a zároveň potřebuje ušetřit náklady potřebné na vývoj aplikace nebo potřebuje vývoj co nejvíce urychlit.

1.6.1 React Native

React Native je nástroj na tvorbu aplikací pro Android a iOS, který je založený na Reactu (jedné z nejrozšířenějších a nejznámějších JavaScriptových knihoven). Velká část programového kódu napsaného v React Nativu může být využita pro více platforem, což je ostatně princip multiplatformního vývoje. Součástí nástroje je spousta komponent, které jsou následně převedené do nativního uživatelského rozhraní dané platformy, díky tomu působí na uživatele podobně jako nativní aplikace. Předpokladem pro práci s React Nativem je alespoň základní znalost programovacího jazyka JavaScript [15]. Multiplatformní aplikace sice může po uživatelské stránce působit podobně jako nativní, nikdy ale nemůže dosáhnout srovnatelného výkonu a optimalizace.

Pro vývoj mobilní aplikace pomocí React Native je možné zvolit jednu z následujících možností. Jednodušší a rychlejší volba je Expo Go, složitější, ale pokročilejší volba je React Native CLI [15].

Expo Go je mobilní aplikace obsahující balíček nástrojů a služeb potřebných pro tvorbu React Native aplikace. Kromě něj je třeba mít nejnovější Node.js (nástroj na spouštění JavaScriptového kódu), libovolný editor programového kódu (například Visual Studio Code) a mobilní zařízení nebo jeho emulátor, na němž je nainstalované Expo Go [15].

React Native CLI je rozhraní pro příkazový řádek, které slouží pro vytvoření a spouštění aplikace. Pro vývoj mobilní aplikace pro Android je kromě něj ještě potřeba Node.js, JDK

(soubor nástrojů pro vývoj aplikací v jazyce Java) a Android Studio (vývojové prostředí pro Android). Pro vývoj mobilní aplikace pro platformu iOS je třeba zařízení s macOS (protože vývoj na operačních systémech Windows a Linux není podporován), Node.js, Ruby (univerzální programovací jazyk, který je potřeba pro některé skripty React Nativu), Ruby's Bundler (manažer balíčků pro jazyk Ruby) a Xcode (vývojové prostředí pro iOS) [15].

1.6.2 Flutter

Flutter je soubor nástrojů pro vývoj mobilních, webových, desktopových a vestavěných aplikací založený na moderním programovacím jazyce Dart. Programový kód napsaný v Dartu je následně zkompilován do nativního kódu pro danou platformu, díky čemuž jsou aplikace vytvořené ve Flutteru rychlé. Stejně jako u React Nativu velká část programového kódu je společná pro více platforem. Jednou z hlavních výhod je, že jak programovací jazyk Dart, tak framework Flutter jsou vyvíjeni společností Google, která zároveň vyvíjí operační systém Android. Lze tedy předpokládat, že kompatibilita s platformou Android a budoucnost nástroje je zajištěna [16]. Další nespornou výhodou Flutteru oproti ostatním nástrojům je fakt, že zatímco oblíbenost jiných nástrojů klesá, tak jeho oblíbenost stoupá. Dle průzkumů v roce 2019 využívalo Flutter pouze 30 % vývojářů, toto číslo v roce 2021 stouplo na 42 % [17]. Je možné tedy předpokládat, že multiplatformní vývoj se přesouvá směrem k Flutteru, který stále nabývá na popularitě.

Pro vývoj mobilní aplikace ve Flutteru je třeba nainstalovat Flutter SDK (sadu nástrojů na vývoj softwaru ve Flutteru). Pro vývoj mobilních aplikací na platformu Android je třeba nainstalovat Android Studio. Stejně jako u React Nativu, vývoj pro platformu iOS není na operačních systémech Windows a Linux podporován. Ten je možný pouze na zařízení s macOS, na kterém je nainstalován Xcode [16].

1.7 Hybridní vývoj

Hybridní aplikace jsou vytvořené pomocí jazyků pro vývoj webových aplikací (HTML5, CSS a JavaScript), ale jsou zabalené do nativního kontejneru (takzvaného WebView), který umožňuje zobrazování webového obsahu a pomocí doplňkových modulů umožňuje přístup k nativním funkcím zařízení (například fotoaparátu nebo GPS). Aplikace tedy vypadá nativně, ale funguje jako webová [18].

Jednou z hlavních výhod hybridních aplikací je kompatibilita napříč platformami. Stejně jako u multiplatformního vývoje, většina programovacího kódu může být využita pro více platforem. Tím je ušetřen čas potřebný na vývoj, a tedy i celkové náklady. Hybridní aplikace má podobné vlastnosti jako multiplatformní aplikace, ale zásadním rozdílem jsou webové technologie, které jsou použité k jejímu vývoji. Jednou z hlavních nevýhod je slabší výkon hybridních aplikací, protože mezi webovou aplikací a operačním systémem je „komunikační“ vrstva, která může omezovat rychlost aplikace [18]. Protože hybridní aplikace jsou založené na webových technologiích, další nespornou výhodou je potenciální konverze již existující webové aplikace na hybridní.

I když je možné vytvořit hybridní aplikaci pomocí webových technologií a následně ji implementovat pomocí jazyků pro nativní vývoj, existují nástroje pro usnadnění takové práce. Mezi ně patří například Ionic nebo Cordova [18].

Hybridní vývoj s příchodem progresivních webových aplikací pomalu ztrácí význam. Oba přístupy jsou si velice podobné, mají stejné výhody, ale PWA aplikace mají navíc větší dosah, což je podrobněji vysvětleno v kapitole PWA.

1.7.1 Ionic

Ionic je sada nástrojů pro vývoj mobilních aplikací za pomoci webových technologií. Pro vývoj aplikací v Ionicu je možné použít nejpoužívanější vývojové platformy pro webové aplikace, kterými jsou Angular, React a Vue [19].

Pro vývoj aplikací pomocí Ionic je potřeba nainstalovat Ionic CLI. Pro vývoj aplikací na operační systém Android je třeba nainstalovat Android Studio. Stejně jako u ostatních nástrojů vývoj iOS aplikací je možný pouze na zařízení z macOS, na kterém je nainstalován Xcode [19].

1.8 PWA

Na PWA neboli progresivní webové aplikace je možné nahlížet jako na moderní verzi hybridních aplikací. Tyto aplikace v sobě kombinují možnosti nativních aplikací jako je například přístup k místním souborům, přístup k připojeným zařízením nebo přístup k senzorům a dalšímu vybavení zařízení s dosahem webových aplikací, čímž je míněna možnost přístupu k aplikaci z velkého množství různých zařízení (jakékoli zařízení s moderním webovým prohlížečem). PWA jsou webové aplikace, které jsou stejně jaké hybridní aplikace vytvořené za pomoci webových technologií, kterými jsou HTML5, CSS nebo JavaScript.

Funkčnost progresivní webové aplikace je závislá na moderních funkcích webového prohlížeče, takže pro plnou funkčnost aplikace je třeba na zařízení mít nainstalovanou jeho nejnovější verzi. PWA lze nainstalovat na dané zařízení, čímž se do zařízení přidá spouštěcí ikona a zároveň je v některých případech možné aplikaci využívat i v offline režimu. Na rozdíl od hybridních aplikací, které jsou zabalené do nativního kontejneru, se jedná o stoprocentní webovou aplikaci, která po instalaci běží v samostatném okně prohlížeče bez adresního řádku [20]. PWA je multiplatformní aplikace v pravém slova smyslu, protože pokud je správně vytvořena, je možné pro všechny platformy využít stejný programový kód. Mezi hlavní nevýhody těchto aplikací patří to, že prozatím nemohou dosáhnout všech možností nativních aplikací a na starším zařízení nemusí být plně funkční z důvodu zastaralosti webového prohlížeče daného zařízení.

Dle Googlu by PWA měla splňovat následující vlastnosti:

- Měla by být rychlá.
- Pro přístup k aplikaci je možné použít jakýkoli webový prohlížeč.
- Je možné ji spustit a mít přístup ke všem jejím funkcím na každém zařízení nehledě na velikost obrazovky.
- Aplikace má vlastní offline stránku, která se uživateli zobrazí, když je bez připojení k internetu a nezobrazí univerzální offline stránku webového prohlížeče.
- Je možné ji nainstalovat na zařízení [20].

Pro tvorbu PWA je možné využít nástroje, které slouží ke zjednodušení vývoje webových aplikací jako je například Angular nebo Vue.js.

2 Učebnice Project Explore 1

Mobilní aplikace, která je k této práci vytvářena, by měla sloužit k procvičování slovní zásoby nacházející se na konci pracovního sešitu k učebnici Project Explore 1. Nejdříve v této sekci je popsán důvod, proč je zvolena právě tato učebnice. Dále vzhledem k tomu, že pracovní sešit a učebnice tvoří jeden celek, je popsána struktura učebnice i pracovního sešitu.

2.1 Výběr učebnice

Učení slovní zásoby je nedílnou součástí osvojování cizího jazyka. Cílem této práce je vytvořit aplikaci na procvičování slovní zásoby, která by mohla pomoci při výuce anglického jazyka na druhém stupni základní školy. Měla by usnadnit práci pedagogům a zpříjemnit výuku žákům.

Učitelé anglického jazyka často využívají učebnice, podle kterých přizpůsobují svoji výuku. V učebnicích bývá souhrnný seznam slovní zásoby, kterou by si měl daný žák osvojit. Aplikace, která je vytvářena jako součást této práce, by měla proces osvojování této slovní zásoby zjednodušit a zpříjemnit.

Protože důležitou součástí tvorby aplikace je její testování a ladění, je potřeba vybrat školu, kde by bylo možné toto testování aplikace provést. Jako testovací škola je zvolena Základní škola a Mateřská škola Kamenický Šenov, náměstí Míru 616, příspěvková organizace. Tato škola používá k výuce anglického jazyka učebnice Project od nakladatelství Oxford University Press, které jsou běžně užívané na českých základních školách. Je tedy rozhodnuto, že aplikace je vytvářena pro učebnici *Project Fourth edition* (čtvrtá edice).

Během tvorby této práce došlo na vybrané škole ke změně učebnice. Místo učebnice Project Fourth edition používají její novější verzi *Project Explore*. Protože cílem práce je vytvořit použitelnou pomůcku, kterou je třeba otestovat, je změněno zadání práce a místo mobilní aplikace na procvičování slovní zásoby z učebnice Project Fourth edition je v praktické části této práce vytvářena aplikace na procvičování slovní zásoby z učebnice Project Explore 1, která je určena převážně pro šestou třídu základní školy.

2.2 Učebnice Project Explore 1

Učebnice Project Explore 1 je rozdělena na sedm tematicky zaměřených jednotek (kapitol), z nichž první je úvodní:

0) Introduction
1) My life
2) I live here
3) Animals everywhere
4) Be active!
5) Traveling
6) Food is fun

Na konci každé jednotky se nacházejí cvičení na opakování. Jednotky jsou dále dělené na lekce. Například první jednotka *Introduction* je rozdělena na lekce:

A. New friends
B. The Exchange students
C. Winston's week

Každá jednotka procvičuje jiné jazykové prostředky a dovednosti. Pro potřeby učebnice jsou za prostředky považovány gramatika, slovní zásoba a výslovnost. Za jazykové dovednosti čtení, psaní, poslech a mluvení. V obsahu učebnice jsou pak specifikovány konkrétní příklady jazykových prostředků a dovedností, které jednotky obsahují. Například jednotka *Introduction* se zaměřuje na gramatiku příslovcí frekvence, slovní zásobu *představování* a *místa ve škole*, tématy pro čtení a psaní jsou *vyhrajte robomazlíčka* a *vítejte v Three Oaks Secondary School* a témata pro poslech a mluvení jsou *rozhovory s datumy* a *Dialogy: otázky na školu.*

Učebnice je dodávána společně s odpovídajícím pracovním sešitem Project Explore 1 [21].

2.3 Pracovní sešit Project Explore 1

Pracovní sešit Project Explore 1 téměř přesně kopíruje učebnici. Je také rozdělen do jednotek a lekcí, které přesně odpovídají těm v učebnici, s tím rozdílem, že úvodní jednotka *Introduction* zde chybí. Na konci každé kapitoly se nachází takzvaný „Progress check", který má podobnou funkci jako opakování v učebnici a měl by ověřit znalosti nabyté v dané jednotce.

Na konci pracovního sešitu se nachází kapitola *Příprava na testování*, která obsahuje vzorové testy pro jednotlivé jednotky. Funkcí této kapitoly je žáky připravit na test z dané jednotky. Dále se zde nachází *Přehled mluvnice*, který slouží jako rekapitulace gramatických pravidel v jednotlivých jednotkách. Na úplném závěru pracovního sešitu najdeme seznam slovní zásoby, takzvaný *Slovníček*, který obsahuje nezbytná a doporučená slovíčka. Ta jsou rozdělena do jednotlivých jednotek a lekcí, aby uspořádání učebnice a pracovního sešitu korespondovalo.

Tučně jsou pak vyznačeny slovíčka nezbytná pro naučení. Žák tedy ví, která slovíčka se potřebuje naučit pro jednotlivé lekce. Tento *Slovníček* slouží jako hlavní podklad pro tvorbu mobilní aplikace [22].

2.4 Mobilní aplikace

V době vytváření této práce neexistuje žádná mobilní aplikace na procvičování slovní zásoby z pracovního sešitu Project Explore 1.

Existují ovšem aplikace, jako je například Quizlet, které jsou univerzální a umožňují uživateli libovolnou slovní zásobu do aplikace doplnit. Bohužel, aby uživatel tyto aplikace mohl naplno využívat, většinou vyžadují nějakou formu platby. Například již výše zmiňovaný Quizlet má prémiový plán s měsíční nebo roční platbou, bez kterého třeba není možné do aplikace nahrát obrázek nebo zvukový záznam [23]. V těchto aplikacích navíc existují již předdefinované metody a způsoby procvičování, které nelze nijak rozšířit a pouze omezeně modifikovat.

Aplikace vytvořené pro konkrétní slovní zásobu jsou oproti univerzálním omezené, protože je není možné libovolně doplňovat. Na druhou stranu některé způsoby a metody procvičování je možné implementovat pouze pokud je slovní zásoba předem dána (například doplňování slov do vět). Další výhodou aplikací pro specifickou slovní zásobu je jejich připravenost. Uživatel nemusí nic doplňovat, aplikaci pouze spustí a hned může začít procvičovat.

V praktické části této práce je vytvářena mobilní aplikace pro specifickou slovní zásobu protože:

1. Cílem práce je vytvořit pomůcku pro zjednodušení práce pedagogům a zpříjemnění učení žákům, takže je nutné, aby ji bylo možné ihned po nainstalování začít používat.
2. Specifická slovní zásoba umožňuje budoucí implementaci různých způsobů procvičování, které v univerzální aplikaci nemůžou být.

3 Metody procvičování slovní zásoby na digitálním zařízení

V této kapitole jsou popsány základní způsoby procvičování slovní zásoby. Pro potřeby této práce je „slovíčko“ považováno za libovolnou jednotku slovní zásoby (tedy například slovo nebo frázi).

3.1 Kartičky (anglicky Flash cards)

Základem tohoto způsobu jsou karty, které je možné otáčet. Na jedné straně karty je vždy slovíčko v jednom jazyce a na druhé straně v jazyce druhém. Uživatel si přečte první stranu karty, zamyslí se, zda zná výraz v druhém jazyce a kartičku otočí. Pokud hádal správně, označí kartičku za známou a pokud ne, označí ji jako neznámou. Takto pokračuje se všemi kartičkami, dokud všechny nejsou označené jako známé [23]. Existují různé varianty této metody, například je možné, aby se na jedné straně nacházela definice daného slovíčka.

3.2 Přiřazování

Při této metodě máme dvě skupiny. Na jedné straně se zpravidla vyskytují procvičovaná slovíčka a na druhé mohou být definice, obrázky, zvukové záznamy nebo slovíčka v druhém jazyce. Úkolem uživatele je vytvářet dvojice tak, že z každé skupiny vždy vybere jeden prvek [24].

3.3 Výběr z možností

V tomto způsobu procvičování má uživatel zpravidla zadané slovíčko, obrázek, definici nebo zvukový záznam slovíčka a má na výběr z několika možností, které zpravidla bývají stejného druhu (slovíčka, obrázky, definice nebo zvukové záznamy). Úkolem uživatele je vybrat z nabízených správnou možnost [23].

3.4 Pravda/nepravda

Uživatel dostane vždy dvojici, může se jednat o slovíčko, definici, obrázek nebo zvukový záznam a úkolem uživatele je určit, zda dvojice k sobě patří. Pokud k sobě patří, označí ji jako správnou, pokud k sobě nepatří, označí ji jako nesprávnou [23].

3.5 Psaní

Cílem tohoto procvičování je zpravidla zafixovat psanou formu slovíčka. Nejčastěji bývá slovíčko napsané v mateřském jazyce uživatele a jeho úkolem je ho napsat v cizím jazyce [23]. Existují i různé varianty. Například je možné, aby místo slovíčka v mateřském jazyce byla definice slovíčka nebo odpovídající obrázek.

3.6 Rozřazování slovíček do skupin

Uživatel dostane nějaké množství slovíček v cizím jazyce a názvy nebo definice skupin (například domácnost nebo letiště), ty mohou být buď v cizím nebo mateřském jazyce. Jeho úkolem je zadaná slovíčka roztřídit do správných skupin [24].

3.7 Označ nehodící

V tomto způsobu procvičování je zadána skupina slovíček v cizím jazyce, která obsahuje jedno, které nepatří do stejné skupiny jako ostatní. Úkolem uživatele je takové slovíčko identifikovat a označit [24]. Například mohou být zadána slovíčka: vařečka, lžíce, hrnec a klíč. Mezi tyto slovíčka evidentně nepatří klíč, protože ostatní spadají do skupiny kuchyňského vybavení. Uživatel tedy správně vybere slovíčko klíč.

3.8 Doplňování slov do věty

Uživatel dostane zadanou větu v cizím jazyce, kde jedno slovíčko chybí. Existují dvě varianty. Buď je úkolem slovíčko do věty dopsat, nebo je možné ho vybrat z nabízených možností [24].

3.9 Doplňování písmenek

Cílem této metody je procvičit správné psaní slov. Uživatel dostane zadané slovo, kde některá písmenka chybí. Úkolem je tato písmenka správně doplnit, aby vzniklo požadované slovo [24]. Problémem tohoto procvičování je, že občas je ze zadaného slova možné vytvořit více různých slov doplněním různých písmenek. Tvůrce procvičování by měl na tuto možnost myslet, aby se nestalo, že uživatel správně vytvoří existující slovo, které by bylo následně označené jako špatná odpověď.

3.10 Sestavení slova ze zadaných písmenek

Tento způsob procvičování uživateli zadá skupinu písmen a jeho úkolem je z nich poskládat požadované slovo v cizím jazyce [24]. Tvůrce tohoto způsobu procvičování by si měl dát pozor, aby ze slovíček nešlo vytvořit slovo jiné. Případně je možné uživateli poskytnout nápovědu, aby bylo dané slovíčko jednoznačné.

3.11 Hry

Interaktivní hry jsou další zajímavé možnosti procvičování slovní zásoby. Jednou z takových je například slovní fotbal, ve kterém uživatel dostane omezený počet písmen a jeho úkolem je z nich vytvořit co nejvíce slov. Každé zadané písmenko může uživatel použít ve více slovech, ale v každém slovu ho může použít pouze jednou. Je možné dostat zadaných více stejných písmen (například dvě písmena „a“) [24]. Nevýhodou tohoto procvičování je fakt, že neprocvičujeme konkrétní slovní zásobu, protože je možné tvořit libovolná slova.

4 Mobilní aplikace na procvičování slovíček z učebnice Project Explore 1

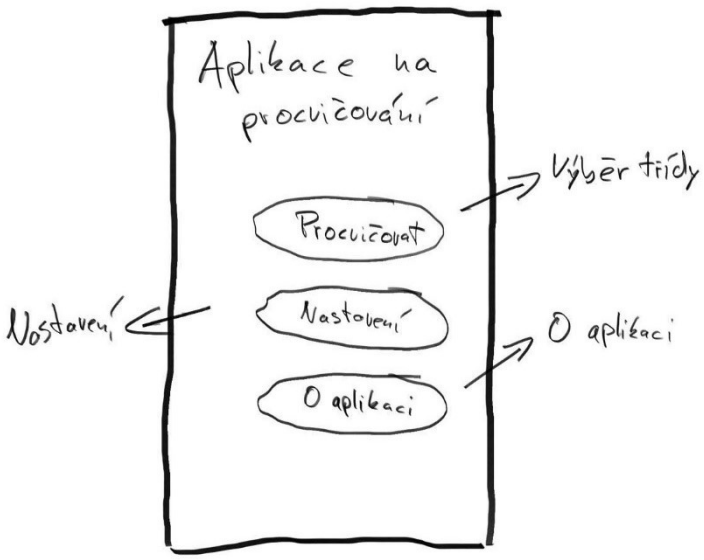
V této části je popsán návrh, tvorba a testování a ladění mobilní aplikace. Jak je již vysvětleno výše, sice se slovíčka nachází v pracovním sešitě, ten je ale nedílnou součástí učebnice. Pokud v práci jsou zmiňována slovíčka z učebnice Project Explore 1, jsou tím míněna slovíčka nacházející se v závěrečné části příslušného pracovního sešitu.

4.1 Návrh aplikace

V procesu vývoje mobilní aplikace je důležitý návrh, při kterém je potřeba naplánovat základní rozložení aplikace, vzhled a funkčnost. Návrh této mobilní aplikace je rozdělen na jednotlivé obrazovky (obsah aplikace zobrazen v jeden konkrétní okamžik naráz). Pokud obrazovka obsahuje navigační prvky (zpravidla tlačítka, která vedou na jiné obrazovky), je pomocí šipek naznačen směr navigace v aplikaci. V některých případech obrazovka zobrazuje prvky, které mohou překrýt dosavadně zobrazovaný obsah, nejedná se ale o samostatnou novou obrazovku.

4.1.1 Domovská obrazovka

Domovská obrazovka je ta, která se zobrazí po spuštění aplikace. Na této obrazovce se v horní části nachází název aplikace. Pod názvem jsou tři tlačítka. První tlačítko s názvem „Procvičovat“ vede na obrazovku *Výběr třídy*, druhé tlačítko s názvem „Nastavení“ vede na obrazovku *Nastavení* a třetí tlačítko „O aplikaci“ vede na obrazovku *O aplikaci.*



Obrázek 1 – Domovská obrazovka

4.1.2 Nastavení

Jednou z budoucích funkcionalit aplikace je, že si bude pamatovat, která slovíčka uživatel již má procvičená a která zatím ještě ne. Zatím jedinou funkcí obrazovky *Nastavení* je možnost resetovat tento postup. *Nastavení* tedy obsahuje tlačítko na resetování postupu uživatele v procvičování slovíček. Dále je zde místo pro jakékoli další potřebné nastavení, které by mohlo vyplynout z vývoje aplikace.

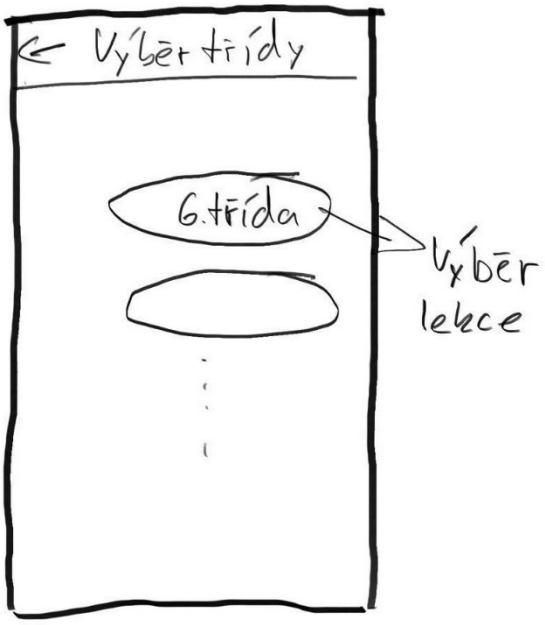
4.1.3 O aplikaci

Obrazovka *O aplikaci* obsahuje základní informace o mobilní aplikaci. Je zde zmíněno, že aplikace je vytvářena v rámci bakalářské práce na Technické univerzitě v Liberci, a že se jedná o aplikaci na procvičování slovní zásoby z učebnice Project Explore 1.

4.1.4 Výběr třídy

Výběr třídy je obrazovka, kde si uživatel vybírá ročník, ze kterého chce procvičovat slovíčka. V horní částí obrazovky se nachází takzvaný appbar (aplikační lišta), ve kterém je název obrazovky a tlačítko zpět na *Domovskou obrazovku*. Pod ním jsou tlačítka s názvy tříd (například „6. třída“), která vedou na obrazovku *Výběr lekce*. V první části vývoje aplikace se

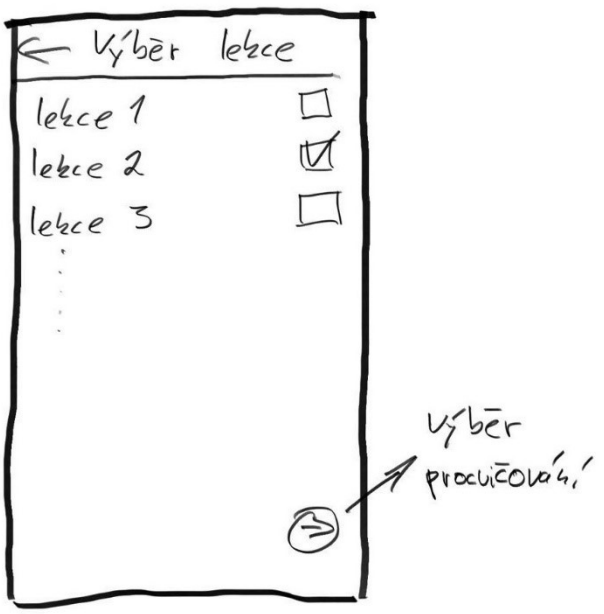
zde nachází pouze šestá třída, protože učebnice Project Explore 1 obsahuje pouze slovíčka pro šestou třídu. Aplikace je však vyvíjena univerzálně a do budoucna je možné přidat další třídy.



Obrázek 2 – Výběr třídy

4.1.5 Výběr lekce

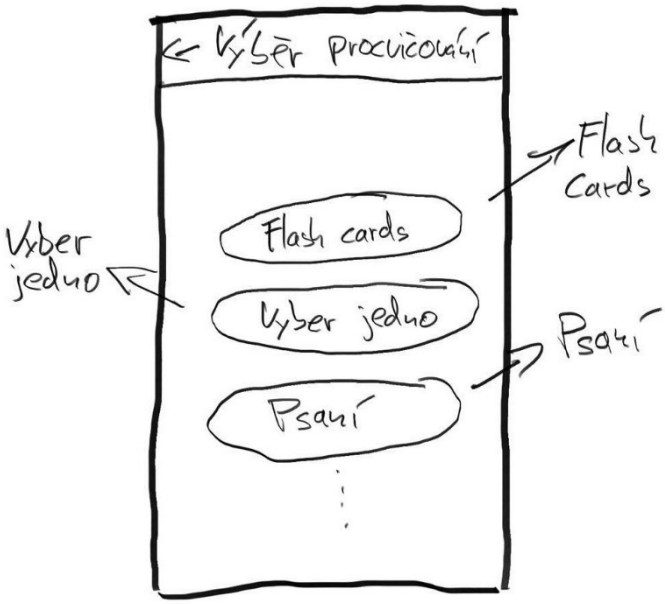
Obrazovka *Výběr lekce* generuje obsah dle zvoleného ročníku na předchozí obrazovce. Zobrazují se zde jednotlivé lekce roztříděné do jednotek tak, jak je tomu v učebnici. Uživatel má možnost si vybrat jednu nebo více lekcí k procvičování najednou. V horní části obrazovky se nachází appbar obsahující název obrazovky a tlačítko zpět, které vede na předchozí obrazovku *Výběr třídy*. Ve zbytku obrazovky je již zmiňovaný seznam lekcí. Každá lekce má v pravé části obrazovky odpovídající zaškrtávací box, který znázorňuje, zda je lekce vybraná nebo ne. V pravé spodní části obrazovky je potvrzovací tlačítko, které vede na další obrazovku *Výběr procvičování*.



Obrázek 3 – Výběr lekce

4.1.6 Výběr procvičování

Dle návrhu v první části vývoje aplikace jsou implementovány pouze tři základní způsoby procvičování. Vývoj aplikace je uzpůsoben tak, aby budoucí implementace dalších způsobů byla co nejjednodušší. První tři způsoby procvičování jsou *Flash cards, Vyber jedno a Psaní*. V horní části obrazovky výběr procvičování se nachází appbar s tlačítkem zpět na obrazovku *Výběr lekce*. Pod ním jsou tlačítka s názvy „Flash cards“, „Vyber jedno“ a „Psaní“, která vedou na příslušné obrazovky *Flash cards, Vyber jedno a Psaní*.



Obrázek 4 – Výběr procvičování

4.1.7 Flash cards

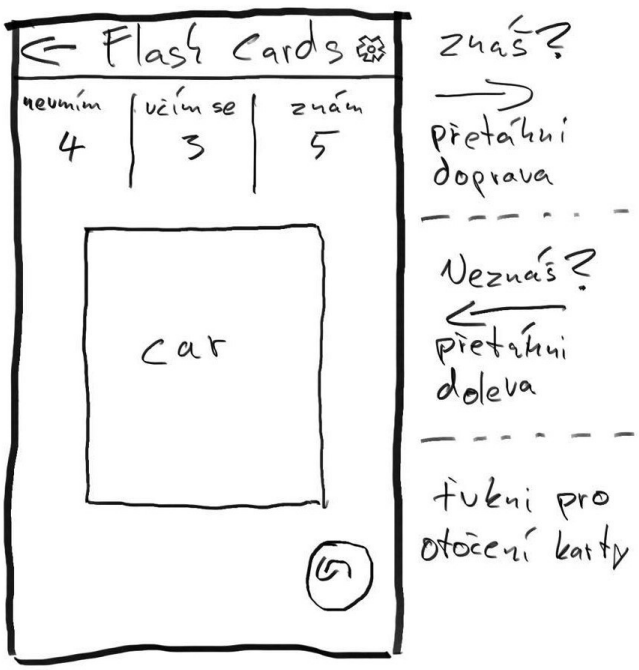
Flash cards obrazovka generuje slovíčka na procvičování v závislosti na uživatelem zvolené lekci na předchozí obrazovce *Výběr lekce*.

Ve vrchní části obrazovky se nachází appbar, který obsahuje název procvičování, tlačítko zpět na předchozí obrazovku *Výběr procvičování* a tlačítko nastavení, které vyvolává dodatečné nastavení, to je podobné pro všechny procvičování (viz níže).

Pod appbarem jsou počítadla, která zaznamenávají počty slovíček. Počítadlo „neumím“ zaznamenává počet slovíček, která uživatel ještě nemá naučená, počítadlo „učím se“, zaznamenává počet slovíček, která se uživatel právě učí, ale ještě je neumí a počítadlo „znám“ zaznamenává počet slovíček, která již uživatel umí a není třeba je dále procvičovat. Každé ze slovíček může být započítané pouze na jednom počítadle, takže celkový součet čísel na počítadlech souhlasí s celkovým počtem slovíček, která jsou vybrána pro procvičování. Aplikace si pamatuje, které slovíčko se nachází, v kterém počítadle, i po vypnutí, takže při opětovném zapnutí aplikace uživatel může pokračovat v procvičování.

Pod počítadly se nachází hromádka karet, ve které jsou zamíchaná slovíčka, která ještě uživatel nezná, nacházejí se tedy v počítadle „neumím“ nebo „učím se“. Každá karta má na jedné straně slovíčko v angličtině a na druhé straně slovíčko v češtině. Úkolem uživatele je si slovíčko přečíst, zkusit si říct, zda zná jeho překlad a ťuknout na něj, čímž se karta otočí. Pokud slovíčko uhodnul, tak přetáhne kartu doprava a pokud ne, tak přetáhne kartu doleva. Na začátku se všechna slovíčka nacházejí v počítadle „neumím“. Pokud uživatel slovíčko přetáhne doprava, tak se přesune do počítadla „učím se“. Slovíčka v počítadle „učím se“ se přetažením doleva dostanou zpátky do počítadla „neumím“ a přetažením doprava se dostanou do počítadla „znám“. Úkolem uživatele je dostat všechna slovíčka do počítadla „znám“ a tím vyprázdnit hromádku karet. V dodatečném nastavení pod tlačítkem v appbaru je možné změnit, která strana karty se ukazuje jako první (česká nebo anglická).

Pod hromádkou karet se v pravé dolní části obrazovky nachází tlačítko, které vrátí poslední přetaženou kartu zpět.

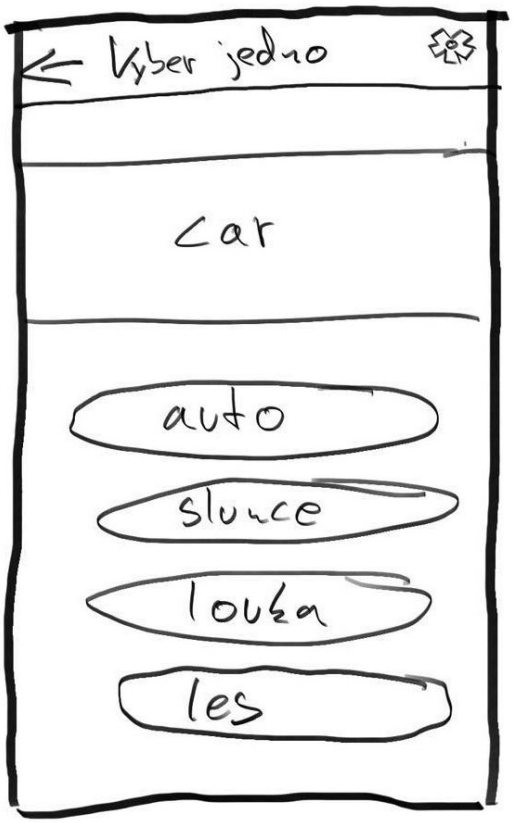


Obrázek 5 – Flash cards

4.1.8 Vyber jedno

Obrazovka *Vyber jedno* stejně jako *Flash cards* generuje slovíčka v závislosti na předchozí volbě uživatele.

Ve vrchní části obrazovky je appbar, který obsahuje název procvičování, tlačítko zpět na předchozí obrazovku *Výběr procvičování* a tlačítko nastavení, které vyvolá dodatečné nastavení. Pod appbarem je napsané slovíčko v angličtině nebo v češtině (dle nastavení), které je náhodně vybráno ze slovíček vygenerovaných pro procvičování. Pod ním se nacházejí čtyři tlačítka se slovíčky v druhém jazyce, z nichž právě jedno je správným překladem slovíčka nad nimi. Ostatní slovíčka na tlačítkách jsou náhodně vybírána z procvičovaných slovíček. Úkolem uživatele je stisknout tlačítko se správným překladem. Po jeho stisknutí je náhodně vybráno nové hledané slovíčko a vygenerována nová tlačítka.

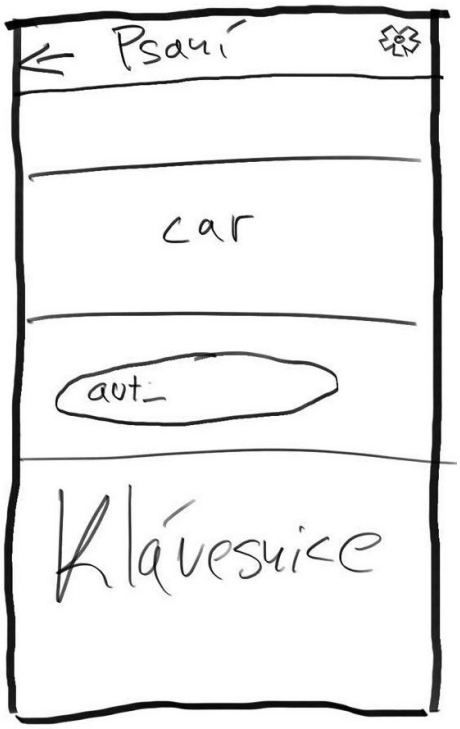


Obrázek 6 – Vyber jedno

4.1.9 Psaní

Obrazovka *Psaní* stejně jako *Flash cards* a *Vyber jedno* generuje slovíčka v závislosti na předchozí volbě uživatele.

Ve vrchní části obrazovky je appbar, který obsahuje název procvičování, tlačítko zpět na předchozí obrazovku *Výběr procvičování* a tlačítko nastavení, které vyvolá dodatečné nastavení. Pod appbarem je napsané slovíčko v angličtině nebo v češtině (dle nastavení), které je náhodně vybrané ze slovíček vygenerovaných pro procvičování. Pod ním se nachází pole pro vstup z klávesnice. Úkolem uživatele je do vstupního pole zapsat správný překlad slovíčka. Po potvrzení je vyhodnoceno, zda je překlad správně nebo ne a je náhodně vybráno nové slovíčko pro překlad.



Obrázek 7 – Psaní

Dodatečné nastavení (drawer)

Všechna procvičování mají stejné dodatečné nastavení, které se nachází v appbaru v pravém horním rohu obrazovky. Pokud uživatel klikne na tlačítko nastavení, zobrazí se mu takzvaná zásuvka (anglicky drawer), která překryje současnou obrazovku. Zde si uživatel může vybrat, zda chce slovíčka procvičovat z angličtiny do češtiny nebo obráceně. Dále zde může resetovat současné procvičování a začít od začátku, nebo vyvolat nápovědu k současnému procvičování a jeho ovládání.

4.2 +-Volba vývojového nástroje

Prvním důležitým faktorem pro volbu nástroje je platforma, pro kterou je aplikace vyvíjena. Většina žáků na zvolené základní škole k testování aplikace vlastní mobilní zařízení s operačním systémem Android. Konkrétně ze dvou šestých tříd, kterých se testování týká, pouze jeden žák má mobilní zařízení s jiným operačním systémem, a to s iOS od společnosti Apple. Pro vývoj aplikace je k dispozici počítač s operačním systémem Windows, takže vývoj mobilní aplikace pro platformu iOS stejně bez větších komplikací není možný. Z těchto důvodů je vývoj zaměřen pouze pro mobilní zařízení na platformě Android.

Dále je důležité rozhodnout jaký zvolit přístup k vývoji aplikace. Vzhledem ke skutečnosti, že multiplatformní vývoj umožňuje s drobnými změnami a úpravami aplikaci vydat pro více operačních systémů, je zvolen tento způsob vývoje. Sice je v plánu vyvíjet aplikaci pouze pro Android, ale mít budoucí možnost jednoduše aplikaci adaptovat pro druhou platformu je velké plus.

Následně je třeba zvolit konkrétní nástroj. Jak je již popsáno výše, Flutter je nástroj, jehož vývoj je přímo podporován společností Android a lze tedy předpokládat nárůst oblíbenosti tohoto nástroje mezi vývojáři. Další jeho nespornou výhodou je fakt, že Flutter je přístupný pro začátečníky ve vývoji mobilních aplikací. Je tedy rozhodnuto, že pro vývoj aplikace bude využit nástroj Flutter.

4.3 Příprava počítače pro vývoj mobilní aplikace ve Flutteru

Přesný návod, jak nainstalovat a zprovoznit Flutter na počítači je možné najít v oficiální dokumentaci na webových stránkách Flutteru. Zde je možné si vybrat specifický návod pro operační systém, na kterém bude aplikace vyvíjena. Instalace na různých počítačích se stejným operačním systémem se může lišit, ale veškeré potřebné kroky jsou popsány v oficiálním návodu. Dle tohoto návodu na zvoleném počítači s operačním systémem Windows je postup následující:

1. Nejdříve je třeba stáhnout nejnovější stabilní verzi Flutter SDK (sada nástrojů na vývoj softwaru). Ta je komprimována v zip souboru, takže ji je třeba dekomprimovat a obsah souboru vložit na libovolné místo v počítači (například doporučované *C:\src\flutter*).
2. Aby bylo možné začít používat Flutter příkazy v systémové konzoli (příkazovém řádku), je nutné upravit proměnné prostředí systému nebo pro konkrétní uživatelský účet Windows. Ty je možné upravit v ovládacích panelech pod nabídkou „Upravit

proměnné prostředí pro váš účet“. Zde je třeba vybrat proměnné Path a zvolit upravit. Dále jen stačí přidat celou cestu ke složce bin obsažené ve složce flutter z předchozího kroku (například *C:\src\flutter\bin*).

3. Nyní, když už je možné začít používat Flutter příkazy, je možné zadat do příkazového řádku příkaz „flutter doctor“, který zkontroluje, jaké další kroky instalace Flutteru jsou potřeba. V případě této instalace je jednou z chybějících věcí takzvaný Android toolchain, který je nezbytný pro vývoj mobilních aplikací na platformu Android. Dále je tedy třeba pokračovat s instalací dle oficiálního návodu v sekci „Android setup“.
4. Prvním krokem v této sekci je instalace Android Studia, jehož instalační balíček je možné stáhnout z oficiálních stránek pro vývoj pro Android. Nejdůležitější součásti balíčku jsou Android SDK, Android SDK nástroje pro příkazový řádek a Android SDK nástroje pro tvorbu instalačního balíčku.
5. V dalším kroku je třeba nastavit virtuální Android mobilní zařízení, které usnadní budoucí spouštění a testování aplikace během vývoje. Tento krok je možné přeskočit a virtuální zařízení nahradit fyzickým Android mobilním zařízením. Nejdříve je třeba povolit VM akceleraci (ta umožňuje virtuálnímu zařízení využít hardware počítače) na daném počítači (to lze udělat v BIOSu). Některé starší počítače nemusí tuto volbu podporovat. Tento krok je opět možné obejít, ale výsledné virtuální zařízení bude běžet znatelně pomaleji. Dále je třeba v Android Studiu najít správce zařízení, kde je třeba vytvořit nové virtuální zařízení. Zde je možné si zvolit jaké konkrétní zařízení má být emulováno a jaká verze Androidu na něm má být spuštěna.
6. Nyní je vše potřebné pro vývoj aplikace nainstalované a nastavené. Posledním nepovinným krokem je instalace oblíbeného editoru programového kódu. Tento krok je možné přeskočit a využít editor programového kódu, který je součástí Android Studia. Mezi nejoblíbenější a nejrozšířenější editory patří Visual Studio Code, který je možné stáhnout z oficiálních stránek. Dalším nepovinným ale doporučeným krokem je instalace Flutter rozšíření pro Visual Studio Code, které barevně rozlišuje různé části programového kódu, napovídá a pomáhá hledat chyby [16].

4.4 Tvorba aplikace

V této sekci je popsán samotný vývoj aplikace se zaměřením na změny vůči původnímu návrhu mobilní aplikace.

4.4.1 Domovská obrazovka

Při vytváření *Domovské obrazovky* jsou implementovány vůči původnímu návrhu dvě zásadní změny. Je zde navíc tlačítko „Nápověda“, které po ťuknutí vyvolá dialogové okno s obecnou nápovědou k celé aplikaci, a tlačítko „O aplikaci“ nakonec neodkazuje na novou obrazovku, ale také vyvolává dialogové okno se základními informacemi o aplikaci.

Obrázek 8 – Domovská obrazovka

4.4.2 Výběr třídy

Obrazovka pro výběr ročníku je vytvářena s myšlenkou na budoucí možné rozšíření o další ročníky. Je tedy vytvořena proměnná „vocabRegister“ datového typu slovník, kde klíčem jeho prvků je název třídy a hodnotou slovíčka pro daný ročník (více k tomuto v další sekci). Při budoucím rozšíření aplikace o další slovíčka stačí pouze vložit záznam do tohoto slovníku.

Obrázek 9 – Výběr třídy

4.4.3 Výběr lekce

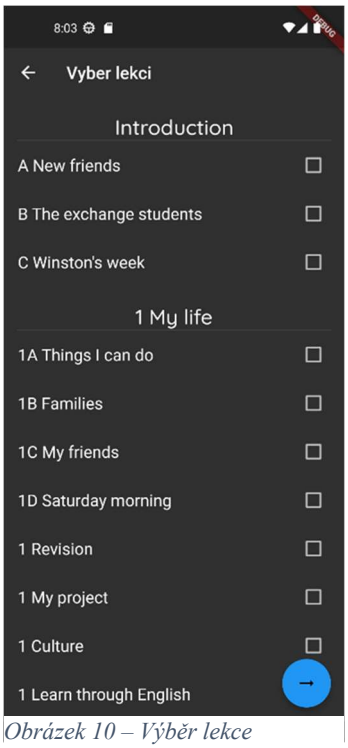
Po výběru ročníku si uživatel musí vybrat lekce, ze kterých chce slovíčka procvičovat. Tato stránka má dle návrhu obsahovat seznam lekcí. Ten je třeba vygenerovat ze zvolené třídy na předchozí obrazovce. Proto se slovíčka pro každý ročník rozdělená do jednotlivých lekcí nacházejí ve vlastní proměnné. Odkaz na ni se nachází jako hodnota v registru tříd (vocabRegister) z předchozí sekce.

Proměnná obsahující slovíčka z daného ročníku je slovníkem, jehož záznamy mají jako klíče jednotky lekcí a jako hodnoty slovníky obsahující lekce spadající do dané jednotky. Tyto vnořené slovníky mají jako klíče názvy lekcí a jako hodnoty seznamy obsahující jednotlivá slovíčka. Seznamy slovíček obsahují vnořené dvouprvkové seznamy, které vždy obsahují anglické a české slovíčko. Z této datové struktury je možné vygenerovat jednotlivé lekce rozdělené do jednotek tak, jak je tomu v návrhu obrazovky.

```
//map of units and lectures with switch to show chosen lectures
Map _lectureList = dsc.dataServiceClass.getVocabList().map((unit, lectures) =>
    MapEntry(
        unit, lectures.keys.map((lecture) => [lecture, false]).toList()));
```

Obrázek 11 – Proměnná obsahující názvy lekcí

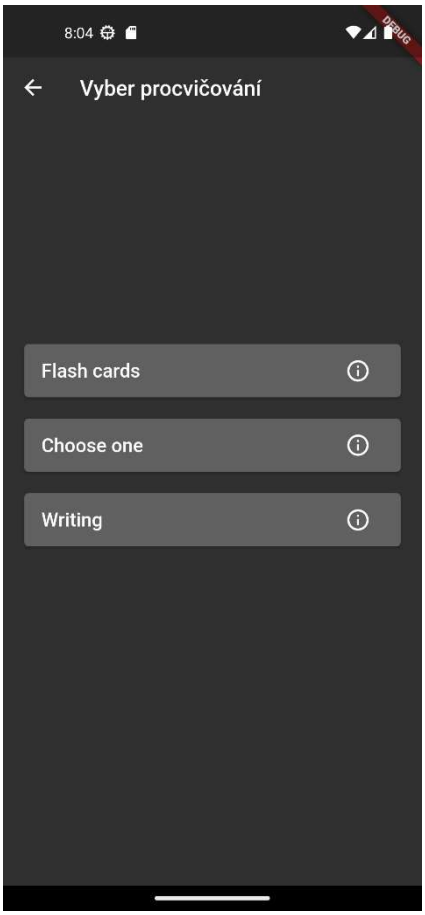
Pro zjednodušení generování lekcí, které se mají zobrazit na této obrazovce, a pro umožnění výběru lekce zaškrtáváním polí je vytvořena další proměnná datového typu slovník, ve které jsou uložené názvy lekcí rozdělené do jednotek (viz Obrázek 11). Každá lekce má příznak datového typu *bool,* který nese informaci o tom, zda je box dané lekce zaškrtnut. Slovníky v programovacím jazyce Dart mají metodu zvanou *map*, která projde daný slovník a transformuje každý jeho prvek dle předem dané funkce [16]. Pro vytvoření nového slovníku obsahujícího pouze lekce s příznaky rozdělené do jednotek je tedy volána metoda *map* patřící proměnné obsahující slovíčka z daného ročníku. Tato metoda vezme každý záznam původního slovníku a vytvoří z něj záznam slovníku nového, který jako klíč má jednotku a jako hodnotu seznam, který obsahuje vnořené dvouprvkové seznamy jednotlivých lekcí spadajících do dané jednotky. Protože u každé lekce je třeba mít informaci, zda její zaškrtávací box je zaškrtnut či nikoli, první prvek seznamu je vždy název lekce a druhý prvek je proměnná datového typu *bool*. Pokud je lekce vybrána (zaškrtnutý box), je v proměnné *bool* uložena hodnota TRUE, pokud není lekce vybrána, je v proměnné uloženo FALSE**.** Při vytváření nového seznamu jsou všechny proměnné *bool* inicializované na FALSE, a proto při načtení této obrazovky není žádné pole zaškrtnuto.



Obrázek 10 – Výběr lekce

4.4.4 Výběr procvičování

Na obrazovce *Výběru procvičování* jsou oproti návrhu pouze přidána tlačítka informací, která se nachází na každém tlačítku s druhem procvičování. Tato tlačítka po ťuknutí zobrazí návod k danému typu procvičování, takže si uživatel může ještě před spuštěním procvičování přečíst návod, jak má postupovat.



Obrázek 12 – Výběr procvičování

4.4.5 Flash cards

Procvičování s kartami je vytvořeno přesně dle návodu včetně zásuvky s dodatečným nastavením, které také odpovídá návrhu. Toto nastavení je stejné pro všechny druhy procvičování, a proto již dále není zmiňováno. Jedinou věcí, která je do procvičování doplněna je možnost na konci procvičování vše resetovat a začít od začátku.

Jak je již uvedeno v návrhu na této obrazovce se nad hromádkou s kartami nachází počítadla, která ve fázi vývoje mají změněné názvy na „neumím“, „znám“ a „umím“. Ve většině případů, když uživatel přesune kartu doleva nebo doprava, se musí změnit čísla na dvou počítadlech naráz, protože se slovíčko přesune z jednoho počítadla do druhého. Ve Flutteru pokaždé, když

se má na obrazovce něco změnit, je třeba obnovit celou stránku (zavolat její metodu build). To je samozřejmě zdlouhavé a neefektivní, existují tedy různé způsoby, jak obnovit pouze část obrazovky (v tomto případě pouze počítadla). Jedním takovým způsobem je takzvaný *ValueListenableBuilder*. Ten tvoří dvojici s *ValueNotifier*, který je proměnnou libovolného datového typu [16]. Hodnoty počítadel na této obrazovce jsou uložené v proměnné typu *ValueNotifier* a prvky obrazovky obsahující počítadla jsou obalené do *ValueListenableBuilderu*. Tato dvojice opatření zajistí, že když se změní hodnota počítadla, obnoví se pouze ta část obrazovky, která obsahuje počítadla.

Po vytvoření této obrazovky je již možné začít s implementací ukládání postupu.

Obrázek 13 – Flash cards

4.4.6 Ukládání postupu

Součástí návrhu *Flash cards* je možnost ukládat postup, čímž je míněno, že si aplikace má pamatovat, zda uživatel slovíčko zná, nezná, nebo se ho teprve učí. Tento příznak je součástí slovíčka. Aplikace si tedy ke každému slovíčku pamatuje český překlad, anglický překlad a jeho příznak (neumím/znám/umím).

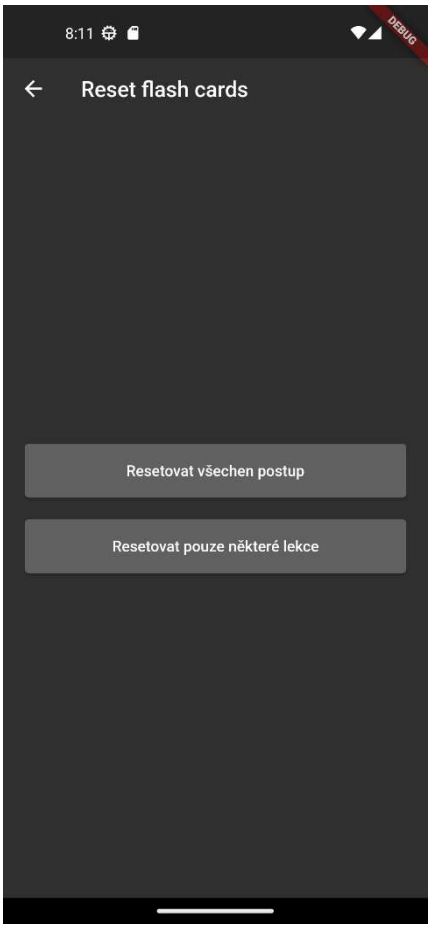
Tento způsob implementace umožňuje uživateli procházet celou aplikaci a když se vrátí zpátky k procvičování, může pokračovat tam, kde skončil. Toto bohužel nefunguje, pokud uživatel nebo operační systém aplikaci úplně zavře. Stačí, že uživatel aplikaci přesune na pozadí a tu systém následně zavře z důvodu šetření prostředků. Tímto jsou odebrány všechny prostředky alokované pro mobilní aplikaci včetně paměti RAM, ve které je uložen postup uživatele.

Za účelem vyřešení tohoto problému, je třeba implementovat ukládání postupu do stálé paměti mobilního zařízení. Nejprve je třeba nastavit, že aplikace při prvním spuštění vezme všechny proměnné obsahující slovíčka, přidá všem slovíčkům příznak „neumím“ a uloží je do trvalého úložiště, odkud pak jsou načtena kdykoliv jsou potřeba. Dále je třeba nastavit ukládání slovíček do paměti kdykoli, když uživatel aplikaci vypne nebo přesune na pozadí, nebo když se během procvičování rozhodne, že si chce procvičit slovíčka z jiného ročníku a vrátí se v aplikaci zpátky na obrazovku *Výběr lekce.* Posledním krokem je nastavení načtení slovíček z uložených souborů při generování obrazovky *Výběr lekce*.

K uložení postupu do stálé paměti mobilního zařízení je nejprve potřeba data převést do formátu *json*, který je následně uložen do souboru jako text, a je potřeba znát cílovou adresu úložiště v mobilním zařízení. Pro zjednodušení je možné využít přídavný modul *path finder* pro Flutter, který dokáže najít běžně užívaná místa k ukládání v daném souborovém systému.

4.4.7 Resetování postupu

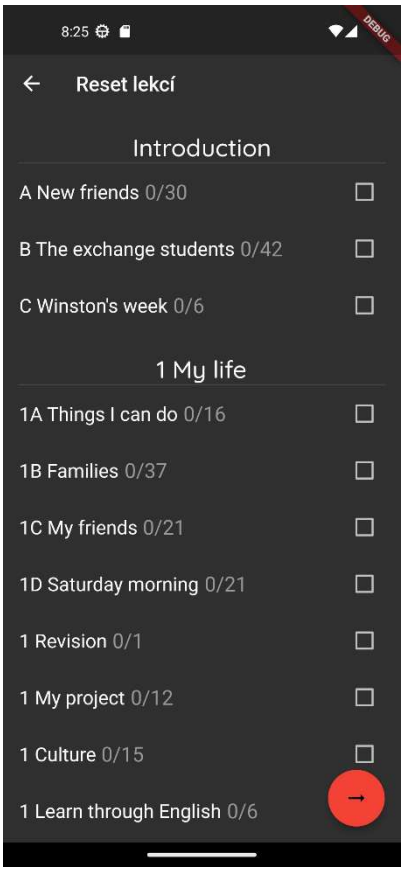
Po implementaci ukládání postupu do paměti telefonu je ještě třeba přidat funkci resetování postupu uživatele, která se nachází na obrazovce *Nastavení*. Funkce reset nabízí buď resetovat všechen postup procvičování pro všechny ročníky, nebo resetovat postup pouze pro některé lekce.



Obrázek 14 – Reset flash cards

Pokud uživatel zvolí možnost resetovat všechen postup, aplikace přepíše u všech slovíček v paměti příznak na neumím.

Pokud zvolí možnost resetovat pouze některé lekce, tak je odkázán na obrazovku *Výběr třídy*. Poté, co zvolí ročník, se před uživatelem objeví obrazovka *Reset lekcí*, která je totožná s obrazovkou *Výběr lekce* s rozdílem, že názvy lekcí obsahují informaci o tom, kolik slovíček z této lekce již uživatel umí. Zde si zaškrtávacími poli uživatel může zvolit jednotlivé lekce, které chce resetovat, tyto lekce jsou pak následně resetovány v souborech uložených ve stálé paměti mobilního zařízení.

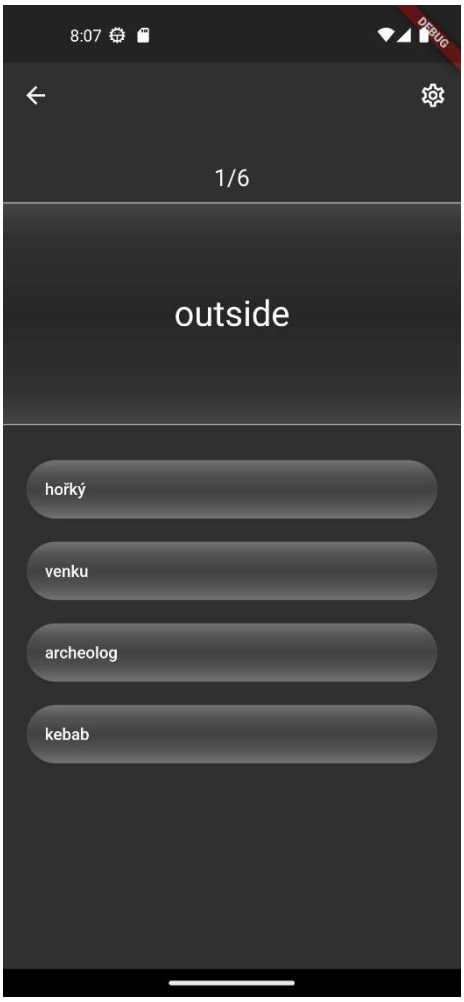


Obrázek 15 – Reset jednotlivých lekcí

4.4.8 Vyber jedno

Obrazovka procvičování *Vyber jedno* je vytvořena téměř přesně dle návrhu. Protože je však třeba, aby měl uživatel zpětnou vazbu, zda překlad zvolil správně nebo špatně, je do aplikace přidána funkcionalita, která zajišťuje, že po zvolení překladu se slovíčko na chvíli podbarví. Pokud uživatel zvolí správnou možnost, podbarví se zeleně a pokud uživatel zvolí špatnou možnost, podbarví se červeně. Dále stejně jako u flash cards je na konec procvičování doplněna možnost resetu.

Protože způsob procvičování psaní má dle návrhu podobné prvky vzhledu a chování, je možné využít velkou část programového kódu z tohoto procvičování včetně podbarvení a doplnění o možnost závěrečného resetu.

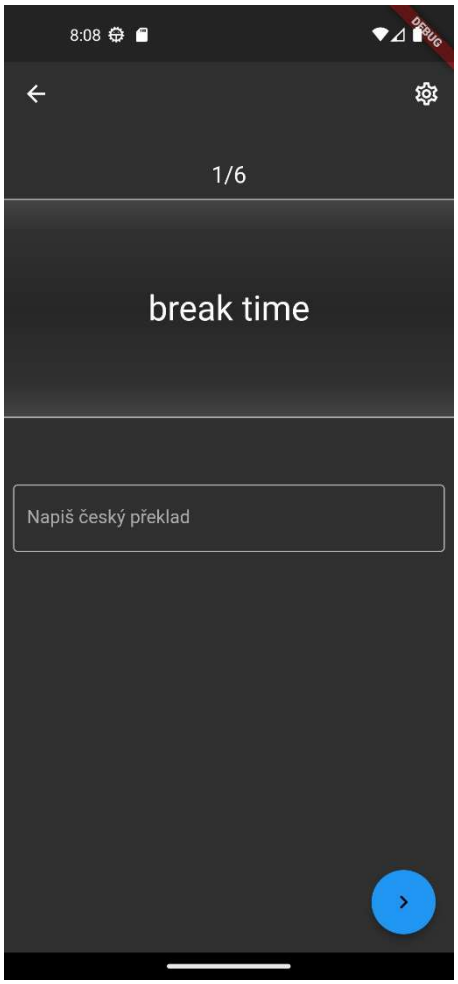


Obrázek 16 – Vyber jedno

4.4.9 Psaní

Většina obrazovky *Psaní* je vytvořena dle návrhu. Obrazovka je pouze doplněna o podbarvení a závěrečný reset (viz předchozí kapitola). Během průběžného testování však vyvstává problém s některými slovíčky, které mají ve slovníku v učebnici více významů (například *letter* je možné přeložit jako písmeno i jako dopis). Pokud by uživatel dostane slovíčko *letter*, jehož překlad má zapsat do vstupního pole, měl by napsat oba významy v tom pořadí, jak jsou v datové struktuře slovníku zapsány a to „písmeno, dopis“. Může se tedy stát, že uživatel napíše jako odpověď „písmeno“ a aplikace vyhodnotí jeho odpověď jako špatnou. Správným řešením je, když aplikace dokáže rozlišit, že se jedná o jednu ze správných možností. To vyžaduje úpravu slovníků obsahujících slovíčka, což je časově náročné. Jako dočasné řešení je zvolena možnost, že když aplikace vyhodnotí odpověď jako špatnou, tak uživateli zobrazí dialogové okno, ve kterém je napsaná správná odpověď. Tím je zajištěno, aby uživatel věděl, že příště musí napsat oba významy v daném pořadí.

Procvičování *Vyber jedno* a *Psaní* neukládají postup uživatele, protože například u procvičování *Vyber jedno* je možnost, že uživatel neúmyslně označí správnou odpověď a nelze tedy s jistotou říct, že slovíčko umí.



Obrázek 17 – Psaní

4.5 Testování a ladění aplikace

Nezbytnou součástí vývoje aplikace je její testování. Protože mobilní aplikace vytvořená jako součást této práce má být výukovým nástrojem pro žáky základních škol, bylo nezbytné její testování provést u dvou cílových skupin. První skupinou byli pedagogové, kteří budou s aplikací v budoucnu pracovat a budou žáky motivovat k jejímu používání. Učitelé dokážou nejlépe zhodnotit, zda aplikace splňuje jejich požadavky na výukový nástroj a zda jim opravdu pomůže s dosažením cílů jejich výuky. Druhou skupinou byli samotní žáci, kteří budou výukový nástroj používat a měl by tedy pro ně být intuitivní a dobře uchopitelný.

Dle Černého [26, s. 66] je vývoj výukového nástroje cyklus, jehož cílem je inovace a zlepšování. Proto bylo testování aplikace rozděleno do dvou fází. Po každé fázi proběhlo vyhodnocení zpětné vazby a následné opravy chyb a implementace nových prvků aplikace.

4.5.1 První fáze testování

V první fázi byla aplikace na testování poskytnuta pedagogům, kteří vyučují anglický jazyk v šestém ročníku základní školy. Účastníky testování byli čtyři učitelé anglického jazyka ze Základní školy a Mateřské školy Kamenický Šenov, náměstí Míru 616, příspěvková organizace.

Cíle první fáze testování

Hlavními cíli první fáze testování bylo zjistit:

1. zda pedagogové považují mobilní aplikaci pro žáky za dostatečně intuitivní a přehlednou,
2. jestli se v aplikaci nacházejí nějaké chyby (po technické i jazykové stránce),
3. jaká vylepšení by pro aplikaci navrhli,
4. co jim v aplikaci chybí a přáli by si do budoucna doplnit.

Postup testování

Učitelé si aplikaci nainstalovali do svých mobilních zařízení. K aplikaci nedostali žádný návod ani pokyny, pouze jim bylo sděleno, k čemu by měla sloužit a na jaké věci by se měli dle cílů testování zaměřit. Poté s nimi byl proveden polostrukturovaný rozhovor, ve kterém měli odpovědět na otázky uvedené v cílech první fáze testování.

První účastník testování

První účastník uvedl, že vidí aplikaci jako dostatečně intuitivní a přehlednou. Dále sdělil, že by na obrazovce s procvičováním *Flash cards* zvětšil písmo na kartách, které dle něj není dostatečně dobře čitelné z větší vzdálenosti. Na závěr uvedl, že by bylo dobré přidat procvičování, kde by bylo možné procvičovat pouze doplňování písmenek do slov.

Druhý účastník testování

Druhý účastník aplikaci zhodnotil jako dostatečně přehlednou, ale uvedl, že mu ze začátku nebylo úplně jasné, co vše lze resetovat v nastavení a navrhl změnu názvu z „Nastavení“ na „Reset flash cards“, aby bylo jasné, že se zde pouze resetuje jeden konkrétní druh procvičování.

Třetí účastník testování

Třetí účastník pouze uvedl, že nepochopil, jak funguje procvičování *Flash cards*, protože přehlédl tlačítko s nápovědou.

Čtvrtý účastník testování

Čtvrtému účastníkovi přišla aplikace přehledná a jasná, ale při prvním spuštění se mu zobrazila prázdná stránka s výběrem lekcí a nebylo tedy možné vybrat jakoukoli lekci a začít procvičovat. Bylo nutné aplikaci restartovat a až poté začala fungovat.

Dále měl spoustu připomínek a nápadů pro vylepšení aplikace.

- Líbila by se mu krásná barevná grafika, která by působila pozitivním dojmem. Dále by aplikace mohla chválit žáka za správné odpovědi a pozitivně ho motivovat k dalšímu procvičování.
- Procvičování *Vyber jedno* vybíralo další slovíčka náhodně ze stejné lekce a občas se stalo, že u jednoslovného výrazu se mezi možnostmi na výběr objevila třeba pětislovná věta. Bylo tedy velice pravděpodobné, že to nebyla ta správná odpověď.
- Procvičování *Vyber jedno* a *Psaní* neměli na rozdíl od *Flash cards* možnost na konci procvičování pokračovat a tím znovu procvičit chybně určená slovíčka.
- Chybělo mu u procvičování *Flash cards* tlačítko reset, které by umožnilo resetovat postup uživatele přímo během procvičování, aniž by musel do hlavního menu.
- Při výběru lekce nebylo vidět, kolik slovíček má již z každé lekce naučeno a kolik ne.
- Líbilo by se mu přidání seznamu, do kterého by se automaticky přidávala často chybovaná slovíčka a následně by bylo možné je procvičit.
- Asi nejdůležitější připomínkou bylo, že někteří učitelé zadávají žákům na naučení pouze vybraná slovíčka, a nikoli celé lekce. Bylo by teda vhodné, aby si uživatel mohl vybrat, která slovíčka chce procvičovat z dané lekce.
- Posledním návrhem bylo přidat do aplikace procvičování poslechu slovíček, která by mohla být namluvená třeba rodilým mluvčím.

4.5.2 Opravy, doplnění a vylepšení v první fázi testování

Nastavení

Protože zatím v průběhu tvorby aplikace nevyvstala potřeba přidávat další možnosti nastavení na obrazovku *Nastavení,* a protože druhý účastník navrhoval změnu kvůli přehlednosti, obrazovka *Nastavení* je přejmenována na obrazovku *Reset flash cards* a tlačítko na domovské obrazovce je přejmenováno na „Resetovat postup".

Výběr lekce

Čtvrtému účastníkovi testování se při prvním načtení aplikace zobrazila prázdná obrazovka *Výběr lekce*. Tento problém se nepodařilo identifikovat, ale je pravděpodobné, že se při spuštění aplikace nevytvořil správně soubor se slovíčky ve stálé paměti telefonu, takže seznam lekcí, ze kterého se měla generovat obrazovka, byl prázdný. Proto je na této obrazovce přidán test, který, když se nepovede načíst slovíčka ze souboru a vytvořit tak nový slovník lekcí rozdělených do jednotek, načte slovíčka z proměnné (viz Obrázek 18). Tím samozřejmě nedojde k načtení postupu uživatele, který je ukládán v souborech, ale protože se chyba vyskytla pouze při prvním startu aplikace, tak to nevadí, protože v tuto dobu uživatel stejně žádný postup ještě uložen nemá. Po načtení slovíček z proměnné je samozřejmě nutné znovu vytvořit proměnnou obsahující lekce rozdělené do jednotek, aby bylo možné uživateli zobrazit seznam jednotlivých lekcí.

```
if (dsc.dataServiceClass.getVocabList() == {} || _lectureList == {}) {
  dsc.dataServiceClass.fillVocabList(
      vocabRegister[dsc.dataServiceClass.getCurrentVocab()]!);
  _lectureList = dsc.dataServiceClass.getVocabList().map((unit, lectures) =>
      MapEntry(
          unit, lectures.keys.map((lecture) => [lecture, false]).toList()));
}
```

Obrázek 18 – Oprava prázdné obrazovky Výběru lekce

Další výtkou čtvrtého účastníka bylo, že by si přál, aby při výběru lekcí pro procvičování bylo možné vidět, kolik slovíček již umí a kolik ještě ne. Nakonec tato vlastnost do aplikace přidána není, protože by se obrazovka *Výběru lekce* podobala obrazovce *Resetu lekcí,* a to by pro uživatele mohlo být matoucí. Dalším důvodem je skutečnost, že výběr lekcí neslouží pouze pro

procvičování *Flash cards*, ale i pro další způsoby procvičování, ve kterých si aplikace postup uživatele nepamatuje.

Poslední připomínkou čtvrtého účastníka testování k obrazovce *Výběru lekce* bylo, že nebylo možné vybrat na procvičování jednotlivá slovíčka, ale pouze celé lekce. V aplikaci je tedy přidána nová funkcionalita, která po vybrání lekce zobrazí dialogové okno, kde si uživatel může vybrat jednotlivá slovíčka, které chce z dané lekce procvičit. Jednotlivá slovíčka z různých lekcí lze libovolně kombinovat. Při přidávání této funkcionality se řešil následující problém. Protože aplikace vytvořená ve Flutteru je složená z takzvaných *widgetů* [16] a dialogové okno je potomkem *widgetu* obrazovky, není možné standardním způsobem (voláním funkce *setState*) při zavření dialogového okna obnovit obrazovku *Výběru lekce* a změnit její stavové hodnoty. Toto lze obejít takzvanou funkcí zpětného volání, která je definována v rodičovském *widgetu* (viz Obrázek 19) a volána z *widgetu* potomka.

```
//callback function to refresh page
void _refresh(lectureVocab, lecture) {
  if (lectureVocab.any((e) => e[1] == true)) {
    setState(() {
      lecture[1] = true;
    });
  } else {
    setState(() {
      lecture[1] = false;
    });
  }
}
```

Obrázek 19 – Definice funkce zpětného volání

Flash cards

První účastník testování navrhoval změnu velikosti písma na kartách obsahujících slovíčka. Po důkladném testování na mobilních zařízeních s různými velikostmi displejů a s různými rozlišeními je písmo upraveno tak, aby bylo dobře čitelné na jakémkoli zařízení.

Další výtkou čtvrtého účastníka testování bylo, že kvůli resetování postupu ve *Flash cards* musí jít zpátky do hlavního menu, což je zbytečně zdlouhavé. Dle něj by bylo nejlepší doplnit možnost resetu přímo do procvičování. Tato možnost se v aplikaci již v době testování vyskytovala. Je možné resetovat postup uživatele v hlavním menu, na konci procvičování, když má uživatel vše hotovo anebo v dodatečném nastavení, které se nachází v pravém horním rohu obrazovky během procvičování.

Vyber jedno

Čtvrtý účastník testování kritizoval skutečnost, že na obrazovce vyber jedno občas dochází k situaci, kdy hledané slovíčko a nabízené varianty mají různý počet slov (dále bude místo počtu slov užíván výraz délka). Díky tomu je moc jednoduché určit správnou odpověď. V aplikaci je tedy přidána funkcionalita, která se snaží do možností na výběr přidat slovíčka stejné délky.

V prvé řadě jsou všechna slovíčka ze slovníku převedena do vnořených seznamů, které umožňují jednodušší prohledávání. Seznam s jednotkami obsahuje vnořené seznamy s lekcemi, které obsahují vnořené seznamy s jednotlivými slovíčky (viz Obrázek 20). Seznamy neobsahují názvy jednotek a lekcí, ty jsou zde zastávány pouze pořadím v jednotlivých seznamech (například první jednotka je na pozici nula ve vnějším seznamu, kde najdeme všechny její lekce a slovíčka, které obsahuje).

```
//[jednotky[lekce[slovíčka]]]
final List _vocabList = dsc.dataServiceClass.getVocabList().values.map((e) {
  return e.values.map((e2) {
    return e2.map((e3) {
      return e3;
    }).toList();
  }).toList();
}).toList();
```

Obrázek 20 – Vnořené seznamy s jednotkami, lekcemi a slovíčky

Dále při výběru slovíček pro jednotlivé možnosti je tento seznam náhodně procházen a hledají se v něm tři slovíčka s podobnou délkou (se správnou odpovědí čtyři možnosti na výběr). Pokud je délka hledaného slovíčka větší než dva, mohou být vybrána libovolná slovíčka délky větší než dva. Pokud je slovíčko délky menší než tři, musí být vybrána slovíčka stejné délky. Všechny jednotky jsou procházeny v náhodném pořadí. V každé jednotce je vybrána náhodná lekce, ve které je hledáno slovíčko požadované délky. Pokud se slovíčko najde, je uloženo a hledá se další. Toto se opakuje třikrát. Na závěr se zkontroluje, zda se našly všechny tři slovíčka. Pokud ne, nahradí se chybějící slovíčka prázdnými textovými řetězci (viz Obrázek 21).

```
//creates list of shuffeled numbers of units - to avoid repetition
List unitCounter = [];
for (var i = 0; i < _vocabList.length; i++) {
  unitCounter.add(i);
}
unitCounter.shuffle();
//if total number of words in current phrase is 3 and more, then any phrase with more than two words is accepted
if (_practiceVocab[_vocabIndex][_language == "Angličtina" ? 1 : 0]
        .split(" ")
        .length >
    2) {
  for (var i = 0; i < 3; i++) {
    //goes through all units, picks random lecture
    unitLoop:
    for (var unit in unitCounter) {
      int lecture = rng.nextInt(_vocabList[unit].length);
      for (var vocab in _vocabList[unit][lecture]) {
        if ((vocab[_language == "Angličtina" ? 1 : 0].split(" ").length >
                2) &&
            !buttonTexts
                .contains(vocab[_language == "Angličtina" ? 1 : 0])) {
          buttonTexts.add(vocab[_language == "Angličtina" ? 1 : 0]);
          break unitLoop;
        }
      }
    }
  }
} else {
  //if total number of words is less than 3, then only phrase with precisely same number of words is accepted
  for (var i = 0; i < 3; i++) {
    unitLoop:
    for (var unit in unitCounter) {
      int lecture = rng.nextInt(_vocabList[unit].length);
      for (var vocab in _vocabList[unit][lecture]) {
        if ((_practiceVocab[_vocabIndex][_language == "Angličtina" ? 1 : 0]
                    .split(" ")
                    .length ==
                vocab[_language == "Angličtina" ? 1 : 0]
                    .split(" ")
                    .length) &&
            !buttonTexts
                .contains(vocab[_language == "Angličtina" ? 1 : 0])) {
          buttonTexts.add(vocab[_language == "Angličtina" ? 1 : 0]);
          break unitLoop;
        }
      }
    }
  }
}
//adds empty strings to total of 4
if (buttonTexts.length < 4) {
  int rep = 4 - buttonTexts.length;
  for (int i = 0; i < rep; i++) {
    buttonTexts.add("");
  }
}
buttonTexts.shuffle();
```

Obrázek 21 – Hledání slovíček podobné délky

Dalším požadavkem čtvrtého účastníka testovaní bylo, aby u procvičování *Vyber jedno* a *Psaní* bylo možné na konci procvičování zvolit možnost pokračovat a znovu procvičit špatně zvolená slova, tak jak je tomu u *Flash cards*. Na konec obou způsobů procvičování je přidána možnost výběru, zda chce uživatel pokračovat v procvičování nebo zda chce začít znovu.

Další návrhy

První a čtvrtý účastník testování navrhovali přidat další způsoby procvičování. V této fázi vývoje nejsou žádné další způsoby procvičování implementovány, ale jejich budoucí implementace není vyloučena.

Čtvrtý účastník navrhoval přidání barevné grafiky, která by pozitivně podporovala učení a motivovala žáky do dalšího procvičování. Grafický návrh uživatelského rozhraní není součástí této práce.

Posledním návrhem čtvrtého účastníka bylo přidat seznam, který by se postupně plnil opakovaně špatně zvolenými slovíčky. Tento seznam by mělo být možné samostatně procvičovat. Seznam zatím není z časových důvodů implementován v této části vývoje, jedná se ale o zajímavý návrh, který by určitě žákům pomohl s jejich nedostatky a je tedy zvažována jeho budoucí implementace do aplikace.

4.5.3 Druhá fáze testování

V druhé fázi testování byla aplikace poskytnuta žákům šestého ročníku Základní školy a Mateřské školy Kamenický Šenov, pro které je aplikace určena. Toto je nezbytná a důležitá část testování, ve které je aplikace testována jejími budoucími uživateli. Testování se účastnilo dvacet jedna žáků.

Cíle druhé fáze testování

Cílem druhé fáze testování bylo zjistit:

1. zda žáci zvládnou pochopit jednotlivé způsoby procvičování,
2. který způsob procvičování jim vyhovoval nejvíce,
3. jestli v aplikaci našli nějakou chybu,
4. co by v aplikaci vylepšili, změnili nebo do ní doplnili,
5. zda jim aplikace v učení pomůže.

Postup testování

Žáci si aplikaci nainstalovali do svých mobilních zařízení. K aplikaci nedostali žádný návod ani pokyny, jak ji používat. Pouze jim bylo sděleno, k čemu aplikace slouží a dostali za úkol ji zkusit použít jako pomůcku k učení slovní zásoby. V ideálním případě by bylo v této části

testování provedeno pozorování nebo uživatelské testování, ve kterém by byli žáci sledováni při používání aplikace. Protože žáci šestého ročníku jsou mladší osmnácti let, byl by třeba souhlas rodičů, což značně komplikuje testování [25]. Bylo tedy rozhodnuto, že testování provedou žáci samostatně a zpětná vazba proběhne formou dotazníku, který je vhodný pro získávání většího množství odpovědí v krátkém časovém úseku [25].

Otázky v dotazníku

1. Pochopil/a jsi, jak funguje Flash cards? Pokud ne, popiš, co ti nebylo jasné.
2. Pochopil/a jsi, jak funguje Vyber jedno? Pokud ne, popiš, co ti nebylo jasné.
3. Pochopil/a jsi, jak funguje Psaní? Pokud ne, popiš, co ti nebylo jasné.
4. Který způsob procvičování ti vyhovoval nejvíce?
    a. Flash cards
    b. Vyber jedno
    c. Psaní
5. Našel/Našla jsi v aplikaci nějakou chybu nebo něco nefungovalo? Pokud ano, napiš co.
6. Co bys na aplikaci vylepšil/a nebo změnil/a?
7. Přijde ti učení pomocí aplikace:
    a. Jednodušší
    b. Stejně náročné
    c. Složitější

Zpětná vazba

Z odpovědí na dotazníky vzešly následující připomínky a návrhy:

- Když žáci v *Psaní* napsali část slova, klávesnice mobilního zařízení jim zobrazila návrhy, ze kterých si mohli vybrat. Pokud vybrali nějaký návrh, automaticky se za něj přidala mezera. Když aplikace porovnávala správnou odpověď se zapsanou, která měla na konci mezeru, vyhodnotila je jako rozdílné, protože správná odpověď na konci mezeru neměla.
- Žáci nechápali, jak se přesouvají slovíčka mezi jednotlivými hromádkami ve *Flash cards*. V nápovědě si přečetli, že cílem je dostat slovíčka na hromádku znám, ale nevěděli, jak to udělat.
- Další připomínkou byl fakt, že pokud nějaké slovíčko má dva významy v druhém jazyce (například „oily“ může být „mastný“ i „olejnatý“), je nutné oba významy napsat

v přesném pořadí tak, jak jsou ve slovníku uvedené a nestačí zapsat pouze jeden význam.

- Posledním návrhem žáků bylo přidání dalších způsobů procvičování.

4.5.4 Opravy, doplnění a vylepšení v druhé fázi testování

Psaní

Jako řešení problému s mezerou na konci slovíčka je do aplikace přidána funkcionalita, která při porovnávání správného a zapsaného slovíčka ignoruje mezery na jeho začátku a konci.

Problém s více významy jednoho slovíčka by byl ideálně vyřešen tak, aby aplikace akceptovala zapsání libovolného správného významu v druhém jazyce. To by ale vyžadovalo úpravu slovníku a z důvodu časové náročnosti zůstalo řešení, které je již uvedeno v kapitole Tvorba aplikace. Po špatném vyhodnocení zapsaného slovíčka se zobrazí správná odpověď, která uživateli ukáže, že je potřeba zapsat oba významy ve správném pořadí.

Flash cards

Žákům nebylo jasné, jak se slovíčka přesouvají mezi jednotlivými počítadly. Je upravena nápověda k procvičování *Flash cards* tak, aby bylo zřejmé, jak slovíčka přesunout.

Další návrhy

Více žáků navrhovalo přidání dalších způsobů procvičování, ale nenapsali, jaké konkrétní způsoby by se jim líbily Jak je již uvedeno výše, v této fázi vývoje nejsou implementovány žádné další způsoby procvičování, ale jejich budoucí implementace není vyloučena.

4.6 Budoucnost aplikace a možná vylepšení

4.6.1 Databáze slovíček

V současné době jsou slovíčka v aplikaci uložená v proměnné, která se během prvního zapnutí aplikace nahraje do stálé paměti mobilního zařízení. Hlavní nevýhodou je fakt, že pro přidání nových slovíček by si uživatel musel stáhnout a nainstalovat novou verzi aplikace. Jedním z možných budoucích vylepšení je slovíčka uložit do nějakého databázového systému, který by umožnil rychlejší prohledávání. Databázový systém by mohl být přístupný online, takže by

aplikace při zapnutí mohla zkontrolovat, zda nedošlo k aktualizaci slovní zásoby a případná nová slovíčka by si mohla stáhnout sama, aniž by bylo nutné stahovat celou novou aplikaci.

4.6.2 Uživatelské účty

Dalším možným vylepšením by mohlo být přidání uživatelských účtů. Každý uživatel by si při prvním spuštění aplikace založil svůj účet. Uživatelské účty by byly zálohované online a postup uživatele by byl tedy uložen jak v zařízení uživatele, tak online. Takže když by uživatel změnil své zařízení, tak by se mohl přihlásit a veškerý postup by mu zůstal zachován.

4.6.3 Grafické rozhraní

Jak uvedl čtvrtý účastník první fáze testování, bylo by vhodné zapracovat na grafickém rozhraní aplikace, které by mohlo být zajímavější a pozitivně laděné tak, aby žáky motivovalo k dalšímu procvičování. Na druhou stranu by uživatelské rozhraní nemělo žáky rozptylovat při učení.

4.6.4 Další způsoby procvičování

Mezi další možnosti vylepšení aplikace určitě patří přidání dalších způsobů procvičování. Aplikace již obsahuje procvičování psané formy slovíček. Dalším logickým rozšířením by bylo procvičování zvukové formy slovíček. Do aplikace by bylo možné například přidat procvičování, kde by si uživatel mohl pustit namluvená slovíčka a přiřazovat k nim správné výrazy v druhém jazyce.

4.6.5 Seznam často chybovaných slovíček

Z první fáze testování vyplynul další zajímavý způsob vylepšení. Aplikace by si mohla pamatovat slovíčka, která uživatel opakovaně volil špatně. Ta by se přidávala do takzvaného chybového seznamu, který by bylo možné samostatně procvičovat.

4.6.6 IOS verze

Vytvoření druhé verze aplikace pro mobilní zařízení na platformě iOS je dalším očividným vylepšením. Toto vylepšení by zpřístupnilo aplikaci i té menšině žáků, kteří vlastní mobilní zařízení značky Apple.

Závěr

V rámci práce je vytvořena mobilní aplikace pro operační systém Android, která by měla studentům pomoci s procvičování slovní zásoby z učebnice anglického jazyka Project explore 1.

Na základě teoretických poznatků ke způsobům vývoje mobilních aplikací byl zvolen přístup multiplatformního vývoje aplikace a pro samotné psaní programového kódu byl využit nástroj Flutter.

Součástí mobilní aplikace jsou slovíčka obsažená ve slovníku pracovního sešitu k učebnici Project explore 1, která je určena převážně pro šestý ročník základní školy. Aplikace je vytvořena univerzálně tak, aby bylo možné její budoucí rozšíření o další slovní zásobu. Mobilní aplikace obsahuje tři možné způsoby procvičování, kterými jsou *Flash cards, Vyber jedno a Psaní*.

Testování a ladění aplikace probíhalo ve dvou fázích. V první fázi byla aplikace poskytnuta pedagogům. Jejich zpětná vazba byla implementována do aplikace. V druhé fázi byla poskytnuta aplikace žákům šestého ročníku, pro které je určena. Opět byla zpětná vazba implementována do aplikace.

Na tvorbu aplikace je možné pohlížet jako na nekonečný proces oprav a implementací nových funkcí. Proto aplikace po dvou fázích testování nemůže být považována za hotovou. Pro zjednodušení přidávání nových slovíček k procvičování by aplikace mohla být doplněna o online databázi slovíček. Dalšími možnými vylepšeními by mohlo být přidání uživatelských účtů, úprava grafického rozhraní, přidání dalších způsobů procvičování nebo doplnění o seznam obsahující často chybovaná slovíčka. Protože byl pro tvorbu aplikace zvolen nástroj pro multiplatformní vývoj, nabízí se možnost budoucího rozšíření aplikace o verzi pro operační systém iOS od společnosti Apple.

Seznam použitých zdrojů

[1] *Committee on National Security Systems (CNSS) Glossary* [online]. Revision No. 4009. Fort Meade: Committee on National Security Systems, 2015 [cit. 2023-04-19]. Dostupné z: www.cnss.gov/CNSS/openDoc.cfm?a=RjTc6cdtHfDUuerxV1m2cA%3D%3D&b=90585AFCA9CF11629E81E60B17609ACC3063C848F07B1F9109160506C23FA846705CA41EEB9DF1CCC5237E6140B53642

[2] AYERS, Rick, Sam BROTHERS a Wayne JANSEN. *Guidelines on Mobile Device Forensics* [online]. Gaithersburg: National Institute of Standards and Technology, 2014 [cit. 2023-04-19]. Dostupné z: nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-101r1.pdf

[3] ROUSE, Margaret. Mobile Application: What Does Mobile Application Mean? In: *Techopedia* [online]. London: Finixio, 2020 [cit. 2023-04-19]. Dostupné z: www.techopedia.com/definition/2953/mobile-application-mobile-app

[4] Mobile Operating System Market Share Europe. In: *Statcounter GlobalStats* [online]. c1999-2023, March 2023 [cit. 2023-04-24]. Dostupné z: gs.statcounter.com/os-market-share/mobile/europe

[5] SOBUSIAK, Piotr. Native vs Hybrid vs PWA mobile apps – which should you choose? In: *Applover* [online]. Wrocław: APPLOVER DIGITAL AGENCY, c2023, 13 October 2021 [cit. 2023-04-25]. Dostupné z: applover.com/blog/native-vs-hybrid-vs-pwa-mobile-apps-which-should-you-choose/?utm_term=&utm_campaign=Performance+Max-flutter&utm_source=adwords&utm_medium=ppc&hsa_acc=4213347825&hsa_cam=19029209224&hsa_grp=&hsa_ad=&hsa_src=x&hsa_tgt=&hsa_kw=&hsa_mt=&hsa_net=adwords&hsa_ver=3&gclid=CjwKCAjwov6hBhBsEiwAvrvN6Jo9eb8GR8arggRSlFy6klngPEqAopdnEjlHSx0xUmA62RdzvW2lQxoCyoIQAvD_BwE

[6] MARCHUK, Anastasiya. Native Vs Cross-Platform Development: Pros & Cons Revealed. In: *Uptech* [online]. c2016-2023 [cit. 2023-04-25]. Dostupné z: www.uptech.team/blog/native-vs-cross-platform-app-development#:~:text=The%20term%20native%20app%20development,Objective%2DC%20for%20iOS%20apps

[7] CUTHBERT, Olivia. Top 7 Reasons Why You Should Choose Java as a Programming Language for Mobile Apps in 2021. In: *Data Science Central* [online]. TechTarget,

c2023, 15 June 2021 [cit. 2023-04-27]. Dostupné z: www.datasciencecentral.com/top-7-reasons-why-choose-java-programming-language-for-mobile

[8] Build Your First Android App in Java. In: *Android for Developers* [online]. Google [cit. 2023-04-27]. Dostupné z: developer.android.com/codelabs/build-your-first-android-app#1

[9] Java Vs Kotlin. In: *Javatpoint* [online]. Noida, c2011-2021 [cit. 2023-04-28]. Dostupné z: www.javatpoint.com/java-vs-kotlin

[10] Android's Kotlin-first approach. In: *Android for Developers* [online]. Google [cit. 2023-04-28]. Dostupné z: developer.android.com/kotlin/first

[11] HATI, Spandita. Swift vs Objective C - Differences Explained in Detail. In: *KnowledgeHut* [online]. c2011-2023 [cit. 2023-04-29]. Dostupné z: www.knowledgehut.com/blog/programming/swift-vs-objective-c

[12] OS (iPhone, iPad) Tutorial. In: *Tutorials Point* [online]. c2023 [cit. 2023-04-29]. Dostupné z: www.tutorialspoint.com/ios/index.htm

[13] Swift: The powerful programming language that is also easy to learn. In: *Developer* [online]. Apple, c2023 [cit. 2023-05-02]. Dostupné z: developer.apple.com/swift/

[14] What is cross-platform mobile development? In: *Kotlin* [online]. [cit. 2023-05-03]. Dostupné z: kotlinlang.org/docs/cross-platform-mobile-development.html

[15] *React Native* [online]. Meta Platforms, c2023 [cit. 2023-05-04]. Dostupné z: reactnative.dev/

[16] *Flutter* [online]. Google [cit. 2023-05-06]. Dostupné z: flutter.dev

[17] Cross-platform mobile frameworks used by software developers worldwide from 2019 to 2021. In: *Statista* [online]. 2021 [cit. 2023-05-06]. Dostupné z: www.statista.com/statistics/869224/worldwide-software-developer-working-hours/

[18] SHIOTSU, Yoshitaka. What Is a Hybrid App? (Detailed Guide for 2023). In: *Upwork* [online]. c2015-2023, 22 November 2021 [cit. 2023-05-08]. Dostupné z: www.upwork.com/resources/hybrid-app

[19] *Ionic framework* [online]. Ionic, c2023 [cit. 2023-05-08]. Dostupné z: ionicframework.com

[20] RICHARD, Sam a Pete LEPAGE. Progressive Web Apps. In: *Web.dev* [online]. Google, 6 January 2020 [cit. 2023-05-13]. Dostupné z: web.dev/progressive-web-apps

[21] PHILLIPS, Sarah, Paul SHIPTON, Michaela TRNOVÁ a Amanda BEGG. Project explore 1. Oxford: Oxford University Press, 2019. ISBN 978-0-19-425574-5

[22] PHILLIPS, Sarah, Paul SHIPTON, Michaela TRNOVÁ a Amanda BEGG. *Project explore 1*. Oxford: Oxford University Press, 2019. ISBN 978-0-19-425638-4.

[23] *Quizlet* [online]. Quizlet, c2023 [cit. 2023-07-14]. Dostupné z: quizlet.com

[24] *British Council: LearnEnglish* [online]. British Council [cit. 2023-07-14]. Dostupné z: learnenglish.britishcouncil.org/vocabulary

[25] MATYASOVA, Jana. Design výukových nástrojů. EDTECH KISK [online]. 2020 [cit. 3. 6. 2021]. ISSN ISSN 2570-9364. Dostupné z: https://medium.com/edtech- -kisk/design-v%C3%BDukov%C3%BDch-n%C3%A1stroj%C5%AF-9ccbb9da9b94

[26] ČERNÝ, Michal. *Pedagogicko-psychologické otázky online vzdělávání*. Brno: Masarykova univerzita, 2018. ISBN 978–80–210–8925–9.

Seznam příloh

Procvičování slovíček – programový kód včetně instalačního souboru mobilní aplikace na procvičování slovní zásoby z učebnice Project explore 1