Gymnázium, Praha 6, Arabská 14

Programování

Ročníková práce

Duben 2025

Jakub Vagera

Gymnázium, Praha 6, Arabská 14

Arabská 14, Praha 6, 160 00

Ročníková práce

Předmět: Programování

Téma: Plánování směn

Školní rok: 2024 / 2025

Autoři: Jakub Vagera

Třída: 4.E

Vedoucí práce: Mgr. Jan Lána

Třídní učitel: Mgr. Blanka Hniličková

Čestné prohlášení

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 31. března 2025 _____

Anotace

Tento projekt je nástroj pro efektivní plánování a evidenci směn. Jeho hlavním cílem je usnadnit správu rozvrhů zaměstnanců. Program umožňuje ruční úpravy směn prostřednictvím webového rozhraní nebo mobilní aplikace. Uživatelé mohou měnit své pracovní dny, požadovat výměny směn, upravovat pracovní hodiny a potvrzovat změny v reálném čase.

Obsah

1 Úvod

Cílem tohoto projektu je navázat na ročníkovou práci z minulého školního roku 2023/2024 [VV23], ve které jsem vytvořil systém na plánování a evidenci směn. Tento projekt importuje již existující systém do PHP frameworku Laravel, zlepšuje UI/UX rozhraní systému, stávající funkce - emailovou verifikaci, algoritmus výběru směn, a přidává nové funkce či nástroje jako např. chatovací místnosti, nabídku volných směn atd.. Projekt vytváří mobilní aplikaci v Javě v operačním systému Android, přes který je možné se přihlásit do systému na plánování směn. Vedlejší cílem projektu je možnost vytvoření a prozkoumání hardwaru na přihlašování pomocí čtečky otisků prstů.

2 Použité technologie

Webové rozhraní projektu bylo vytvořeno v editoru Visual Studio Code, aplikace pro operační systém Android pak v editoru Android Studios (verze Ladybug). Pro webové rozhraní je v projektu využit PHP open-source framework Laravel (verze 11). [Lar] Dále byly k tvorbě webových stránek využity jazyky Javascript, HTML a CSS a grafické knihovny Bootstrap a SweetAlert2. Databáze projektu je sestavena v relačním databázovém modelu MySQL. Pro snazší přístup a jednodušší správu databáze jsem využil administrační systém PhpMyAdmin. Aplikace pro operační systém Android je napsána v jazycích Java a XML a využívá grafickou knihovnu Material 3. Pro zasílání emailů je v programu využita služba Mailtrap a pro zasílání chatovacích zpráv služba Pusher [Pus].[1] Pro chatovací místnosti ve webové aplikaci je využit framework Chatify. [Mun24]

[1] Obě služby vyžadují vytvoření vlastního účtu, nastavení těchto služeb je uvedeno v souboru README.md

3 Struktura Programu

Webová aplikace je postavena ve frameworku Laravel (verze 11). Přístup do programu mají jen uživatelé, kteří jsou zaregistrovaní v databázi v tabulce *users*. Systém k přihlašování využívá vestavěnou Laravel knihovnu Breeze. Každý registrovaný uživatel má jednu ze čtyř rolí. Role v programu jsou : Administrátor, manažer, pracovník na plný úvazek a pracovník na částečný úvazek. Tyto role určují oprávnění a možnosti, které může uživatel v systému vykonávat. Na základě těchto rolí je v programu implementován middleware. [Kno] Jednotlivá práva jsou sepsána v tabulce 1.

Role	**Práva**
Administrátor	- Upravovat pobočkový systém - Upravovat modely směn - Plánovat a měnit směny uživatelů - Náhledu do statistiky všech uživatelů - Náhledu do časových možností všech uživatelů - Schvalovat nabídky volných směn - Vytváření informační tabule - Registrovat zařízení do systému pro přihlášení - Měnění nastavení uživatelů - Přihlášení na směny
Manažer	- Plánovat a měnit určité směny uživatelů - Upravovat určité modely směn - Schvalovat určité nabídky volných směn - Plánovat a měnit směny uživatelů - Měnit nastavení uživatelů - Přihlášení na směny
Pracovník na plný úvazek	- Náhledu do svých statistik - Náhledu do svých časových možností - Přihlášení na směny
Pracovník na plný úvazek	- Náhledu do svých statistik - Náhledu do svých časových možností - Plánování svých časových možností - Přihlášení na směny

Tabulka 1: Práva uživatel

Jediný, kdo může do systému přidávat nové uživatele je uživatel v roli administrátora. Nově je do systému implementován systém ikon, kde každý uživatel si může zvolit svou vlastní profilovou fotku. Ty se v systému ukládají do adresáře */storage/profile-images*. Jednotlivé webové HTML soubory pro webové stránky jsou uloženy v adresáři */resources/views* a jednotlivé PHP soubory pro komunikaci s webovými stránkami a backend serverem jsou uloženy v souboru */app/Http/Controller*s.

3.1 Pobočkový systém

Pro lepší orientaci v programu je zde uplatněn pobočkový/objektový systém, na který se následně vážou směny. Základní princip pobočkového systému oproti předchozí verzi zůstává nezmněněn. V kódu jsou proměnné a funkce spjaté s pobočkovým systémem pojmenované jako objects. Pobočkový systém tvoří stromovou strukturu (nejsou zde cykly a každá pobočka má své potomky a jednoho rodiče s výjimkou kořene). Celý pobočkový systém je uložen v databázi v tabulce *object_model*. Tabulka se skládá z primárního klíče *id_object*, *object_name*, ve kterém je uložen název pobočky. Proměnná *superior_object_id* odkazuje na *id_object* svého rodiče (kořenové prvky mají nastavenou proměnnou na 0). Nově jsou zde v tabulce přidány sloupce: *created_at* & *created_by* pro určení kdy a kdo tabulku vytvořil. Nakonec je zde vložen sloupec *object_icon*, ve kterém se nachází odkaz na Bootstrap ikonu pro lepší grafické rozhraní. [2] Na jednotlivé pobočky se pak vážou směny. Schopnost měnit a upravovat pobočky má pouze uživatel v roli administrátora, ale administrátor může jednotlivým uživatelům v rolí manažera přiřadit práva na správu. Tyto práva neslouží k upravování poboček, ale dávají manažerům práva na úpravy směn navázaných na danou pobočku a tato práva se ukládají v databázi v tabulce *management_rights*.

[2] V kódu je celkem 14 nastavených ikon, pro přidání další ikony, musí také v mobilní aplikaci být přidána ikona ve vectorovém formátu do balícku textit.drawables s korespondujícím názvem

3.2 Model směn

Směna se vždy musí vázat na určitou pobočku. Směny se v databázi ukládají v tabulce *shift_model*. Tabulka se skládá z primárního klíče *id_shift*, proměnné *start_shift* udávající datum vytvoření směny, proměnné *color* udávající barvu směny, proměnné *shift_name* udávající jméno směny a proměnné *id_object* odkazující na primární klíč pobočky, ve které se směna nachází. Dále se v tabulce nacházejí proměnné *monday, tuesday, wednesday, thursday, friday, saturday* a *sunday*, které udávají, zda-li daná směna, probíhá v odpovídajícím dnu v týdnu. Nově je v tabulce vložen sloupec *description* do něhož je možné přidat komentář ke směně, sloupec *created_by*, ve kterém se udává, kdo směnu vytvořil a sloupec *rep_non* udávající activitu směny . Modely směn slouží jako šablona, která se používá pro vytváření jednotlivých směn ke konkrétnímu dni v kalendáři.

3.3 Kalendář a zaznamenávání směn

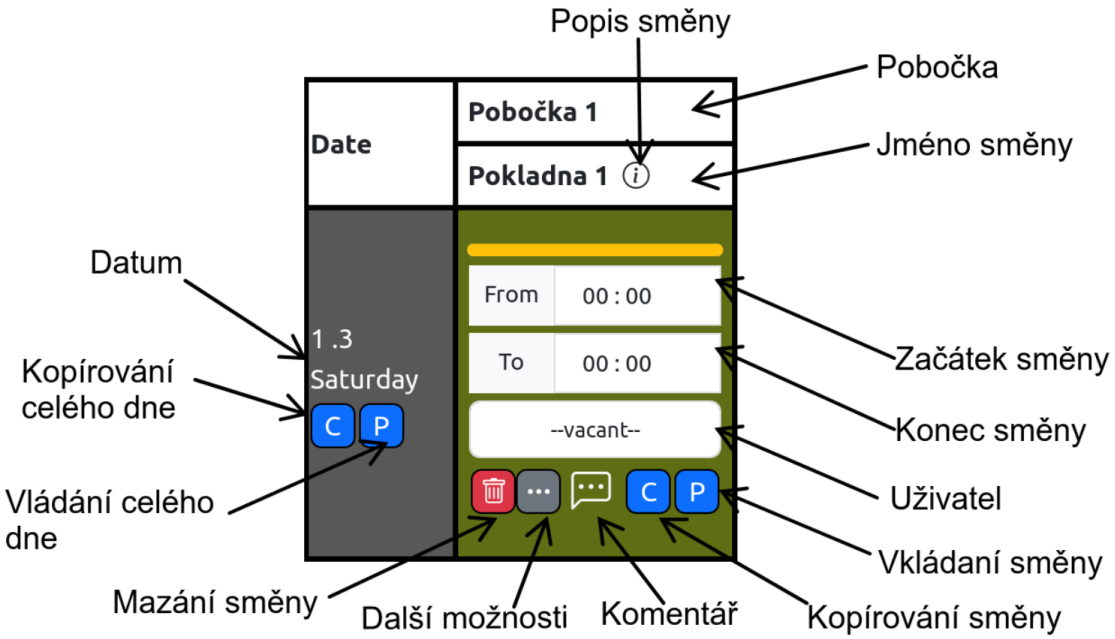
Všechny směny v systému se plánují v kalendáři. V systému jsou 2 typy kalendářů.
1. typ: Kalendář pouze pro náhled na směny.K tomuto kalendáři mají všichni uživatelé přístup.
2 typ: Kalendář pro editaci a plánování směn. K tomu kalendáři mají přístup jen uživatelé v roli administrátora a manažera.

Kalendář, ve kterém se plánují směny je formátován v tabulce, kde sloupce představují dny a řádky představují jednotlivé směny. Jednotlivé kolonky pak představují směny v jednotlivých dnech (viz Obr. 1). Kalendář je možné filtrovat za pomoci HTML multiselektu [Mha25] v horní části obrazovky a to buď podle směn, nebo podle poboček.

Ke každé směně může uživatel přiřadit pracovníka za pomoci tačítka. Po rozkliknutí má ještě uživatel možnost výběru pracovníka a to buď z celého systému, nebo ze seznamu pracovníků, kteří mají oprávnění na dané směně pracovat. Ke každé směně je pak možnost přidat komentář. Nově jsou komentáře ke směně k danému dni vytvářeny ve vyskakovacím okně, namísto v buňce tabulky kalendáře. Při novém načtení kalendáře, ve kterém nejsou žádná uložená data, se kalendář načítá podle standardního modelu nastavené směny. K ověření, že data k danému měsíci existují, se používá v databázi tabulka *shift_check*, která kontroluje, zda-li na daný měsíc a rok existují uložená data ke směně. Pokud v tabulce

existuje záznam, data k určité zvolené směně se načtou z tabulky *shift_active_data*, v níž jsou uložena všechna data směn ke konkrétnímu dni. Nově v programu již neexistuje tabulka *shift_planned_data*, do které se zálohovaly data z tabulky *shift_active_data*. Vlevo od kalendáře se pak nachází tabulka, která shrnuje jednotlivě odpracované směny zaměstnance. Nově lze k jednotlivému dni přiřadit směnu na volný výběr. To znamená, že všichni uživatelé, kteří mají oprávnění na směně pracovat, se mohou ucházet o volnou směnu k jednotlivému dni na domovské ploše nebo v nabídce volných směn. Nabídka směny se v databázi ukládá do tabulky *shift_offers* a žádosti o tuto směnu se ukládají do tabulky *shift_request*. Aby žádost uživatele o volnou směnu k určitému dni byla schválena, musí ji uživatel v roli admina nebo manažera s určitými právy potvrdit.

Co se týče potvrzení příchodu na směny, program dovoluje uživatelům potvrzení příchodu na směny, pokud jejich směna neskončila. Pokud má uživatel v daný den vícero směn, program automaticky přihlásí uživatele na nejbližší danou směnu podle současného času na serveru. Pro úspěšné přihlášení je ještě potřeba, aby zařízení, ze kterého se přihlašuje, bylo ověřeno administrátorem v systému. (Toto omezení neplatí v mobilní aplikaci). Program zvládá také přihlášení i odhlášení ze směn, které začínají a končí v jiné dny (noční směny). Všechny záznamy o příchodech a odchodech se v databázi ukládají do tabulky *attendance*.



Obr. 1: Směna v kalendáři.

3.4 Algoritmus na výběr směn

V kalendáři je možnost přiřadit k volným směnám uživatele pomocí algoritmu. Algoritmus prochází a přiděluje v kalendáři jednotlivé směny po dnech (prochází tabulku po řádcích) a na základě několika filtrací vybere nejlepší uživatele. První filtrace spočívá v tom, že algoritmus k dané směně v daný den vybere všechny zaměstnance, kteří na ní mohou pracovat. Uživatelé, kteří mohou na směně pracovat musí mít oprávnění na této směně pracovat. Jejich časové možnosti nebo trvalé časové možnosti musí vyhovovat času, kdy se směna koná a nesmí mít směnu, která by se již překrývala s touto směnou. Po této filtraci se z dat vytvoří všechny možné kombinace uživatelů k dané směně v rámci celého dne - opět tak, aby nedošlo k překrývání směn viz následující kód:

```

smena_1 = [uzivatel 1, uzivatel 2],
smena_2 = [uzivatel 3]
kombinace [
     [uzivatel 1, uzivatel 3] // Prvni kombinace
     [uzivatel 2, uzivatel 3]  // Druha kombinace
]
```

Tento process slouží k nalezení nejvhodnější kombinace, tak aby došlo k zaplnění co nejvíce směn v daný den. Nejlepší kombinace pak projdou druhou filtrací, kde se vybere nejvhodnější kombinace na základě již přiřazených směn k danému uživateli. Pokud z druhé filtrace dostaneme vícero vhodných kombinací, použije se třetí filtrace, která nalezne kombinaci, kde se nachází nejvíce uživatelů v roli administrátora, manažera nebo pracovníka na plný úvazek. Pokud i z této filtrace dostane program vícero kombinací, vybere se jedna kombinace náhodně a ta se poté aplikuje.

Algoritmus je oproti starší verzi lépe optimalizovaný a umí lépe pracovat s většími daty. Výpočty, které algoritmus provádí jsou nově lépe rozdělené mezi klienta a server, pro větší výkon programu. Diagram algoritmu je uveden v příloze.

4 Aplikační část

Aplikace pro operační systém Android byla vytvořena v editoru Android Studios za použití jazyků Java a XML. Jedná se o novou část projektu, která nebyla součástí práce z minulého roku. Aplikace primárně slouží k ulehčení přihlašování uživatelů na směny přes mobilní zařízení. [eas] Uživatelé v aplikaci mají možnost přihlásit se na směny, ucházet se o volné směny, podívat se na své směny i na směny ostatních zaměstnanců, chatovat s ostatními uživateli a vidět informační tabule a statistiky. Uživatelé pod rolí brigádníka (pracovník na částečný úvazek) mají navíc možnost zvolit si své časové možnosti přímo v aplikaci.

4.1 API

Aplikace získává data z Laravel webového serveru. Ty jsou získávány přes knihovnu *Retrofit*, která je schopna připojit se k serveru. Pro připojení na klientově straně je zapotřebí vložit vlastní serverovou adresu do souboru *ConnectionFile.java*. Ověřování funguje přes Laravel balíček Sanctum. Pokud server běží na jiné adrese než localhost, je zapotřebí na serveru do souboru */config/sanctum.php* nastavit tuto adresu. Sanctum při úspěšném zadání přihlašovacích parametrů vytvoří pro uživatele token, který si uloží do tabulky *personal_access_tokens* a kopie se pošle klientovy. Aplikace následně tento token ukládá v *SharedPreferences*. K zasílání a získávání následných dat ze serveru je zapotřebí tento token vždy odesílat v hlavičce requestu. viz následující kód.

```
@POST("/targetURL")
Call<ResponseBody> sendMessage(
        @Header("Authorization") String authToken,
        @Body SendRequest request
);
```

Nastavené cesty k serveru jsou definovány v souborech */Api/ChatService.java*, *DataService.java* a *LoginService.java*. V souboru LoginService.java je definována URL cesta k přihlášení. V souboru ChatService.java jsou definovány URL cesty potřebné pro chatovací místnosti a v souboru DataService.java jsou definované URL cesty pro získávání dat ze serveru. Api requesty jsou na serveru zpracovány v souborech */routes/api.php* a */routes/chatify/api/api.php*. Jednotlivé soubory odesílající data jsou uloženy v adresáři

/app/Http/Controllers začínající slovem Api. Pro přenos živých dat v chatovacích místnostech využívá aplikace službu Pusher. Ta posílá requesty na server do souboru */routes/chatify/api.php*. Pro spuštění živého přenosu dat je zapotřebí mít vytvořený účet u Pusheru (Pusher má bezplatnou registraci do jistého počtu denních živých přenosů) a v něm vytvořený kanál. Dále je zapotřebí nastavit v souboru *PusherConnnectionFile.java* klíč ke kanálu a clusteru pro připojení a zachycení živého přenosu v chatovací aplikaci.

4.2 Struktura Android aplikace

Android aplikace je napsaná v jazyku Java. Pro grafické rozhraní využívá program jazyk XML a pro konfigurační nastavení jazyk Groovy. Prvotní soubor aplikace, který se spustí při načtení, je *LoginActivity.java*, ve kterém probíhá přihlašování do aplikace. Pokud je uživatel úspěšně přihlášen, je poslán do hlavní řídící aktivity *HomePage.java*. Úspěšné přihlášení, program zjišťuje přes existenci *SharedPreferences*. Pokud při spuštění aplikace existují *SharedPreferences*, uživatel je rovnou poslán do souboru *HomePage.java*. Aplikace je převážně postavena ve fragmentech. Řídící soubory k jednotlivým fragmentům jsou uloženy v adresáři *.ui*. Každý fragment má svůj XML soubor, který odpovídá jmennému prostoru (namespace) daného fragmentu. Mezi fragmenty se dá pohybovat za pomocí menu v levém rohu horní lišty. V pravém horním rohu lišty se pak nachází menu pro odhlášení.

Ve struktuře je hojně využíván element dynamického list *RecycleView*. Ten je například použit při vytváření a načítání kalendáře či k načítání seznamu. Jednotlivá skripta pro tyto listy jsou uložena v balíčku *.adapters*. Většina listů má vlastní objekt kvůli zamezení asynchronizace při načítání. Všechny modely, které aplikace využívá, jsou umístěny v balíčku *.model*. Jednotlivé soubory, které připojují klienta k serveru, jsou uloženy v balíčku .connection a jednotlivé URL cesty jsou uloženy v balíčku *.api_routes*

Individuální grafické prvky jsou uloženy v balíčku *.drawable*. Systém využívá grafické ikony Bootstrapu, převedené z SVG formátu do vektorového formátu. Tyto jednotlivé ikony jsou uložené v balíčku a začínají pod zkratkou bi_ . Systém také využívá grafickou paletu Bootstrapu, která je definovaná v souboru colors.xml kvůli konzistenci s webovým rozhraním. Dále systém využívá grafickou knihovnu Material 3 a knihovnu Glide pro načítání obrázků z URL v operačním systému.

5 Databáze

Databáze v projektu, stejně jako v předchozí verzi z minulého ročníku, je vytvořena v relačním databázovém serveru MySQL. Pro snazší přístup a jednodušší správu do databáze byl využit administrační systém phpMyAdmin. Celkem se celý databáze skládá z 32 tabulek. Jednotlivé funkce tabulek databáze jsou vypsány a vysvětleny v tabulce 2 a 3.

Tabulka	**Popis**
users	Základní tabulka programu, jsou zde uloženy všechna data o uživatelích
attendance	Ukládá se zde docházka jednotlivých uživatelů
board	Nachází se zde data o informačních tabulích, které jsou zobrazovány v domovské stránce
board_logs	Ukládá záznamy o změnách v informačních tabulích
ch_favorities	Vygenerována a používána frameworkem Chatify k zaznamenávání oblíbených uživatelů v chatovacích místnostech
ch_messages	Vygenerována a používána frameworkem Chatify. Ukládá zaslané zprávy z one-to-one konverzací
devices	Slouží k ukládání povolených zařízení pro přihlašování
edit_logs	Ukládá záznamy o změnách uživatelů jinými uživateli
management _rights	Ukládá práva uživatelů na pozici manažera k úpravám v systému
management _rights_logs	Ukládá záznamy k jednotlivým změnám v nastavení práv manažerů
object_model	Nachází se zde pobočková (objektová) struktura celého programu
password_reset_tokens	Implementovaná Laravel autentifikační knihovnou Breeze. Slouží pro reset hesla přes emailovou adresu
permanent_time_options	Slouží pro ukládání trvalých časových možností uživatel s výjimkou uživatel v roli brigádníka (pracovníka na částečný úvazek)
permanent_time_options - _logs	Ukládá záznamy o změnách trvalých časových možností uživatelů
personal_access_tokens	Implementovaná balíčkem Sanctum, který poskytuje systém pro jednoduché ověření pro aplikace SPA (single page applications)
profile_ picture	Ukládá cesty do úložiště k profilovým obrázkům
sessions	Vygenerovaná knihovnou Laravel Breeze, slouží k ukládání sessions.

Tabulka 2: Tabulky v databázi

Tabulka	**Popis**
shift_active_data	Ukládá všechny naplánované směny na konkrétní den
shift_assignment	Skladuje data o směnách přiřazených k jednotlivým uživatelům
shift_assignment_logs	Ukládá záznamy o změnách přiřazených směn
shift_check	Ověřuje, zda-li existují uložená data v kalendáři k jednotlivému měsíci a roku na směnu
shift_model	Ukládá nabídky volných směn
shift_request	Ukládá žádosti o volné směny
time_options	Ukládá data o časových možnostech uživatelů na pozici brigádníka v konkrétních dnech
users_logs	Zaznamenává registrace uživatelů v webovém rozhraní
verification_codes	Uskladňuje prvotní ověřovací kódy potřebné k přístupu do systému přes emailovou adresu

Tabulka 3: Tabulky v databázi

Dále se zde nachází tabulky - *jobs, failed_jobs, jobs_batches, migrations, cache* a *cache_logs*. Tyto tabulky jsou vygenerované frameworkem Laravel a pro hlavní fungování systému nemají žádnou úlohu. Oproti předchozí verzi se v databázi nachází nové tabulky, do nichž se ukládají záznamy o změnách (tabulky končící slovem logs) a tabulky *shift_request* a *shift_offer*, které slouží pro nabídku volných směn. Struktura některých tabulek byla pozměněna, aby odpovídala relacím a správnému jmennému prostoru. Nově se přihlášení do databáze na serveru provádí přes skrytý soubor */.env* oproti předchoyímu souboru
/database.php, což vylepšuje bezpečnost programu. Pro komunikaci s databází program nově používá vestavěný Laravel příkaz DB:: a částečně Laravel ORM strukturu. Celá databáze se dá zálohovat v Laravelu pomocí přidaného bash příkazu php artisan db:backup (Příkaz vytvoří snapshot databáze a uloží ho do souboru */storage/backup/*) [Lea22].

6 Hardwarová část

Vedlejší cílem mé práce byla myšlenka vytvoření a prozkoumání hardwaru na přihlašování do systému pomocí čtečky otisků prstů. K tomu by bylo zapotřebí vytvořit aplikaci, která by vyčkala na klientově straně na USB portu se zapojenou čtečkou a po naskenování by čtečka poslala na server request s biometrickými údaji. Server by pak request zpracoval a na základě porovnání otisku prstu s databází by vrátil reponse a popřípadě přihlásil uživatele na adekvátní směnu. Pro spuštění čtečky na Ubuntu je zapotřebí aktivovat vestavěný nástroj pro správu otisků prstů s názvem Fprint. Těmito příkazy:

```
sudo apt update
sudo apt install fprintd libpam-fprintd
fprintd-enroll
```

Data na server by se museli posílat v binárním formátu pro uskladnění do databáze a pro následný porovnávací algoritmus. [Cue]

Bohužel z technických a časových důvodů jsem nebyl schopen dovést tuto myšlenku do konce. Prvním důvodem byla technická nekompatibilita pořízené čtečky otisku prstu Kensington VeriMark a mého vývojového operačního systému Ubuntu 20.4. , který mi nedovolil testování a odesílání biometrických údajů a neschopnost obstarání kompatibilní čtečky. Druhým důvodem byly mé omezené časové možnosti, které jsem ve větší míře využil k dopracování a vylepšení jiných částí projektu.

Obr. 2: Čtečka otisku prstu Kensington VeriMark.

7 Závěr

I přes nedodělání hardwarové části hodnotím svou práci jako úspěšnou. Ve webové části jsem docílil stanoveného cíle importovat projekt do PHP frameworku Laravel a v aplikační části jsem úspěšně vytvořil aplikaci spustitelnou v Android operačním systému a schopnou připojit se na vzdálený server. Dále jsem docílil přidání nových prvků do systému - jako například živé chatovací místnosti a úspěšně jsem vylepšil UI/UX design celého programu.

Seznam obrázků

Seznam příloh

Literatura

[Cue] Manuel Cuevas. Fingerprint algorithm recognition. https://medium.com/@cuevas1208/fingerprint-algorithm-recognition-fd2ac0c6f5fc. *Medium*.

[eas] Code easy. Implementing user login registration in android with laravel api, https://www.youtube.com/watch?v=mqltpfdjaegt=1772s.

[Kno] Web Tech Knowledge. How to make multiple authentication in laravel 11 breeze | multi role auth in laravel tutorial, https://www.youtube.com/watch?v=g3ueclzpr4k.

[Lar] Laravel Documentation. Installation - laravel, https://laravel.com/docs/11.x/installation.

[Lea22] Learning Point. Automatic database backup daily, https://www.youtube.com/watch?v=fqgz8p35yhm, 2022.

[Mha25] Habib Mhamadi. multi-select-tag: A lightweight multi-select dropdown component, 2025. Accessed: 29 March 2025.

[Mun24] Munafio. Chatify framework v1.6.3, https://github.com/munafio/chatify/tree/master, 2024.

[Pus] Pusher. Channels documentation, https://pusher.com/docs/channels.

[VV23] J. Vagera and G. Vakula, M. a Stamenová. Gyarab/2023-3e-planovani_smen, https://github.com/gyarab/2023-3e-planovani$_smen, 2023$.