Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Vývoj aplikácií s viacvrstvovou architektúrou - Vlastný projekt

RayBank internet banking

Michaela Gubovská a Jakub Hajdu
Pondelok 16:00
Cvičiaci: PhDr. Ing. Mgr. Miroslav Reiter, Dis., MBA, MPA, MSC, DBA, Ing. Paed. IGIP

Michaela Gubovská a Jakub Hajdu
VAVA - Semestrálny projekt
Pondelok 16:00

1. Vízia

RayBank je desktopová aplikácia internetového bankovníctva. Aplikáciu môže používať každý, kto sa rozhodol založiť si v danej banke účet. Používateľ si môže vytvoriť jeden z dvoch typov účtu a na základe neho môže využívať internetové bankovníctvo. Túto aplikáciu bude môcť takisto používať zamestnanec pobočky banky, ako aj správca.

Aplikácia ponúka komfortnú prácu s bankovým účtom, nakoľko je prehľadná a intuitívne navigovateľná. Nevyžaduje fyzický kontakt so zamestnancom banky a šetrí čas zjednodušením a digitalizáciou základných úkonov, nakoľko kvôli nim nie je majiteľ účtu nútený navštíviť pobočku svojej banky. Pri hotovostných vkladoch samozrejme nie je možné úplne oddeliť činnosť zákazníka a zamestnanca, avšak úkon potvrdzovania vkladu sa dá našou aplikáciou výrazne zjednodušiť, nakoľko sa zrýchli a zefektívni autorizácia, čo platí aj o výberoch z bankomatov.

Aplikácia je rovnako prehľadná a užitočná aj pre zamestnancov banky, nakoľko majú okamžitý prístup k údajom o každom existujúcom účte. Pre zamestnanca je jednoduchší úkon spracovania vkladu hotovosti od zákazníka, nemusí overovať a vypĺňať údaje, ktoré sú štandardne spracovávané práve manuálne zamestnancom.

2. Využitie

Aplikácia RayBank je využiteľná v kontexte internetového bankovníctva. Je cielená pre banku, ktorá chce ponúknuť svojim zákazníkom jednoduché založenie účtu a vykonávanie finančných operácií z pohodlia domova. Vďaka desktopovej aplikácii je jednoduché spravovať svoj účet hlavne z dôvodu online potvrdzovania vkladov na účet, ktoré nevyžaduje fyzický kontakt so zamestnancom banky a čakanie na spracovanie vkladu ním.

3. Poznaj svojich používateľov

Počet používateľov aplikácie nie je nijako obmedzený. Každý používateľ vie aplikáciu používať u seba lokálne a jej používanie je obmedzené na základe typu používate%la (zamestnanec, admin alebo zákazník). Čas strávený v aplikácii je pre bežného zákazníka v rozsahu niekoľkých minút, teda aplikáciu využíva iba chvíľkovo na vykonanie finančnej operácie, neobmedzený počet dní v týždni. Zamestnanci aplikáciu využívajú 8 hodín týždenne v rámci pracovnej doby.

Po spustení aplikácie je potrebné sa prihlásiť vyplnením používateľských údajov. Pokiaľ s aplikáciou pracuje nový používateľ, ktorý si chce v banke založiť účet a ešte nemá pridelený prístup, použije registračný formulár, kde si vytvorí konto, do ktorého sa následne môže prihlásiť. Keďže každý zákazník by mal mať v banke svoj účet, bude umožnené si ho pri registrácii založiť výberom z ponuky typov účtov (napríklad bežný alebo študentský).

Zákazník s existujúcim kontom sa po prihlásení dostáva do prostredia s prehľadom o transakciách na svojom účte a možnosťami na realizáciu platby, vkladu alebo výberu. Peniaze z vkladu sa zákazníkovi zobrazia na účte až po jeho potvrdení niektorým zo zamestnancov. Zákazník si môže tiež exportovať dokument s výpisom z účtu za zvolené obdobie.

Zamestnanec pobočky má vo svojom prostredí prehľad o pohyboch na účte zvoleného zákazníka a zároveň vidí všetky vklady na účty čakajúce na potvrdenie. Tieto

vklady môže postupne spracovávať a potvrdzovať. Okrem toho si vie pozrieť detail zákazníkov, teda ich meno s priezviskom, dátum narodenia, emailovú adresu, bydlisko a používateľské meno, typ účtu vedený v banke, IBAN a zostatok na účte.

Administrátor má vo svojom prostredí možnosť vytvorenia konta pre nového zamestnanca. Okrem toho vidí všetkých zamestnancov, ktorým môže v prípade potreby zrušiť konto v aplikácii. Tak isto vidí všetky informácie o zamestnancoch aj zákazníkoch. Vie rušiť účty zákazníkov, ktorí si želajú, aby tak spravil. Admin vidí prihlasovacie údaje všetkých používateľov a nikto iný k nim z bezpečnostného hľadiska nemá prístup.

Aplikácia je ošetrená voči chybným používateľským vstupom, čiže nie je možné zabrániť jej správnemu používaniu pri jej behu. Napríklad je ošetrená potreba zadania všetkých povinných polí, nastavenie obdobia pri výpise z účtu, potreba zadania príjemcu pri platbe alebo limitovanie platobných operácií na základe aktuálneho zostatku na účte zákazníka.

4. Najbežnejšie úlohy

Čo chce Admin:

- Vytvárať nových zamestnancov tak, aby mali prístup do systému a mohli spracovávať požiadavky na vklady
- Rušiť účty zákazníkov, ktorí to vyžiadajú, aby sa zachovala dobrovoľná báza vedenia účtu v RayBank

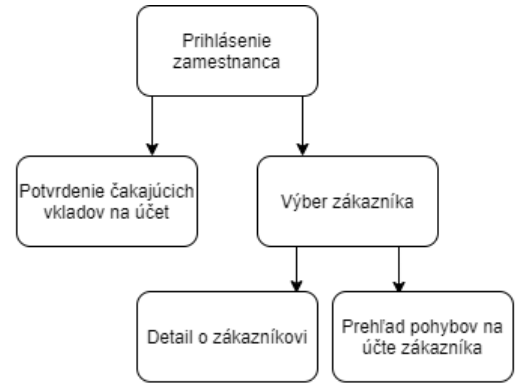
Čo chce Zamestnanec:

- Potvrdzovať vklady na účty zákazníkov, aby tým zabezpečil pripísanie vloženej sumy peňazí na účet zákazníka
- Vidieť detaily o pohybe na účte zákazníka, aby v prípade potreby bolo možné zaznamenať nevyžiadanú aktivitu
- Vidieť detail o konte zákazníka, aby bolo možné zákazníka kontaktovať prostredníctvom emailu

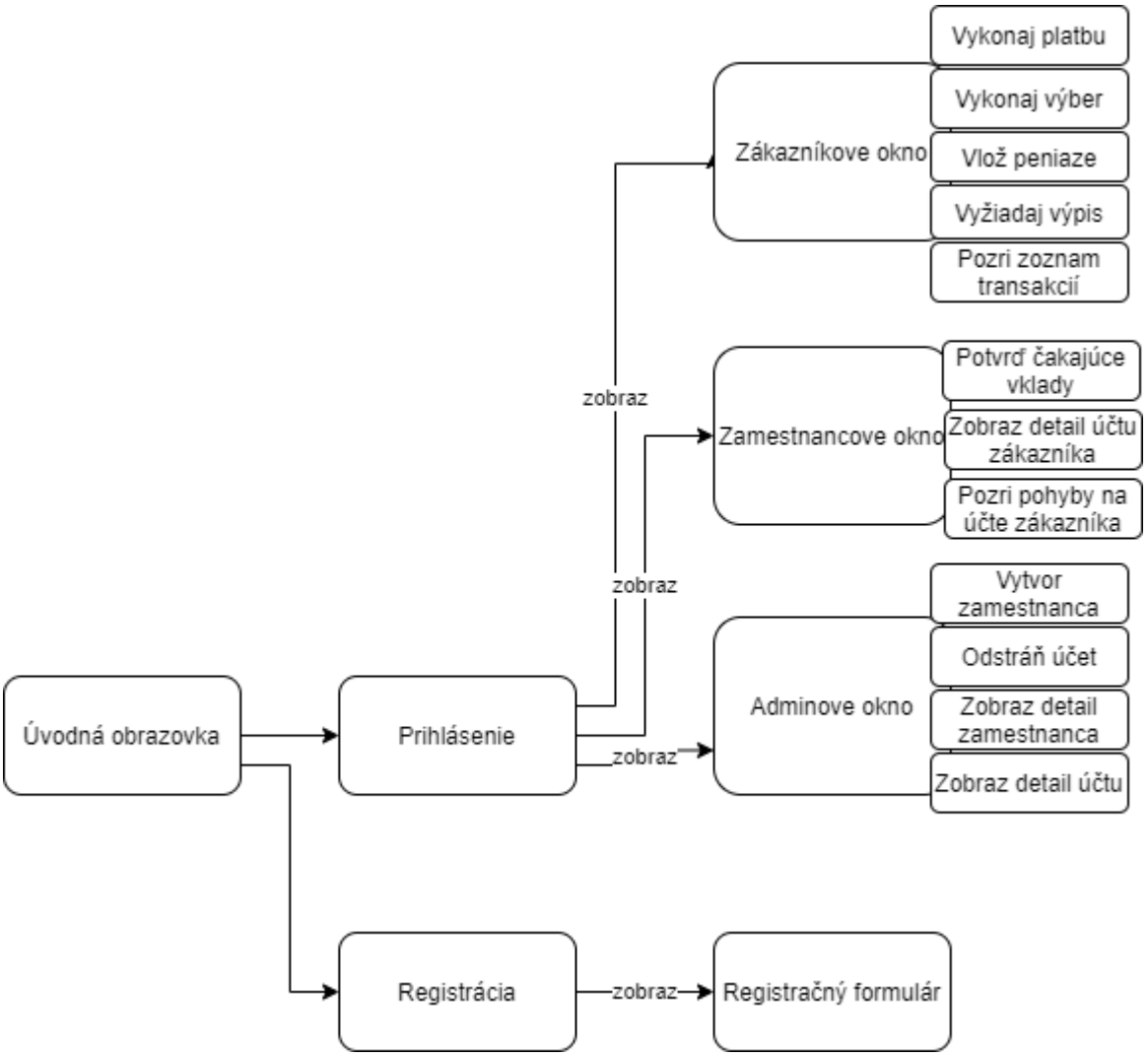
Čo chce Zákazník:

- Rýchlo sa zaregistrovať, aby si mohol v RayBank viesť účet
- Mať prístup ku pohybom na účte a vykonávaniu platobných operácií online
- Vyžiadať vygenerovanie výpisu z účtu za zvolené obdobie pre udržiavanie prehľadu

5. Hlavné procesy

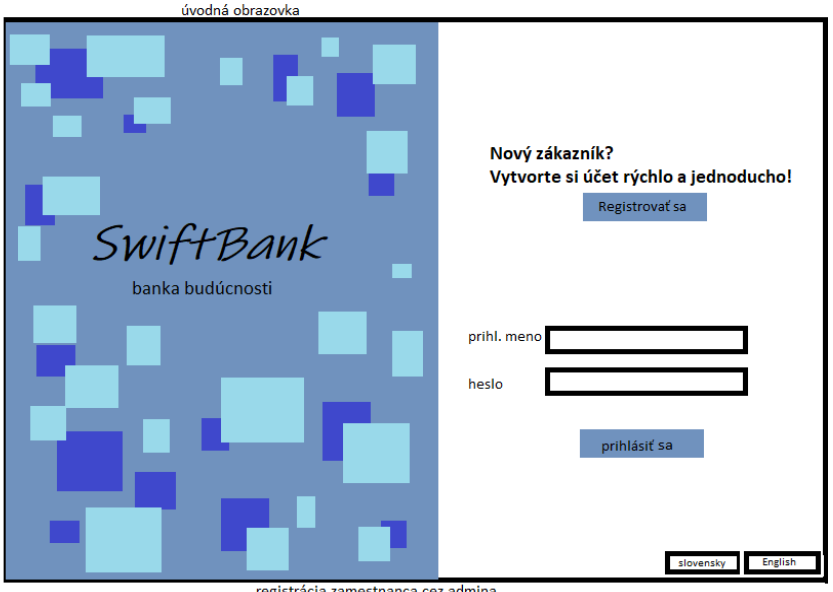


6. Navigácia

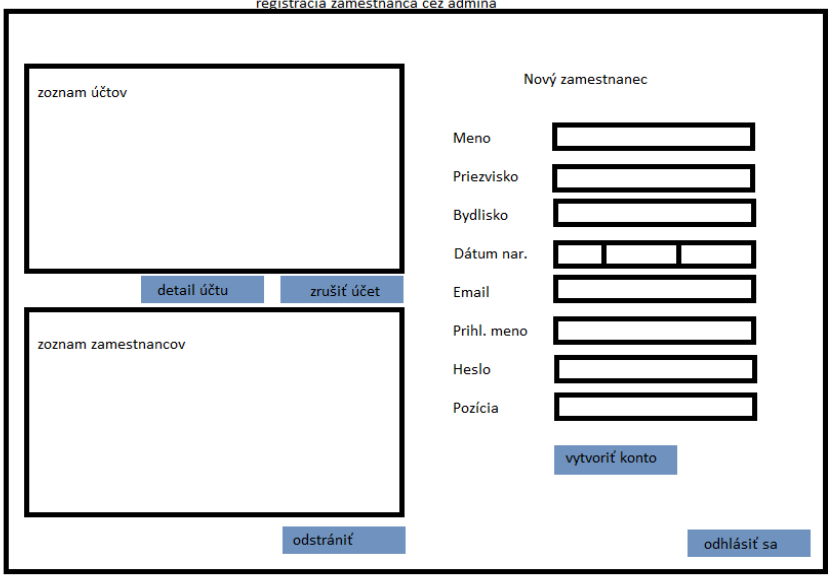

Úvodná obrazovka
Prihlásenie
Registrácia
Registračný formulár
zobraz
zobraz
zobraz
zobraz
Zákazníkove okno
Vykonaj platbu
Vykonaj výber
Vlož peniaze
Vyžiadaj výpis
Pozri zoznam transakcií
Zamestnancove okno
Potvrď čakajúce vklady
Zobraz detail účtu zákazníka
Pozri pohyby na účte zákazníka
Adminove okno
Vytvor zamestnanca
Odstráň účet
Zobraz detail zamestnanca
Zobraz detail účtu

7. Mockupy

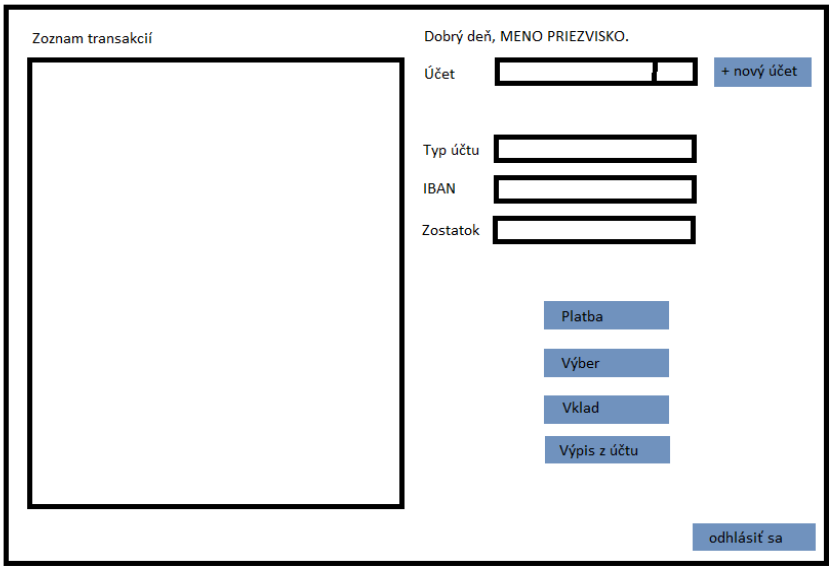
úvodná obrazovka



registrácia zamestnanca cez admina

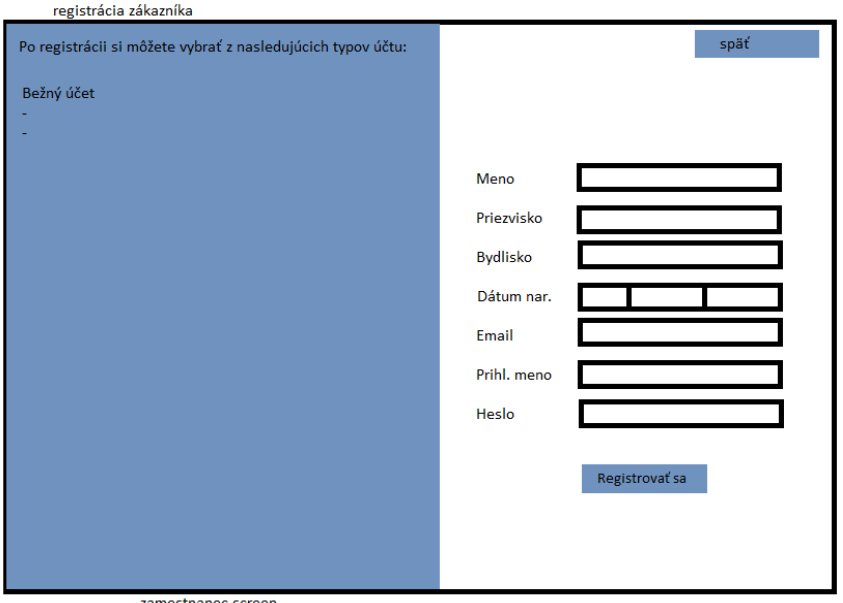


zákazník screen

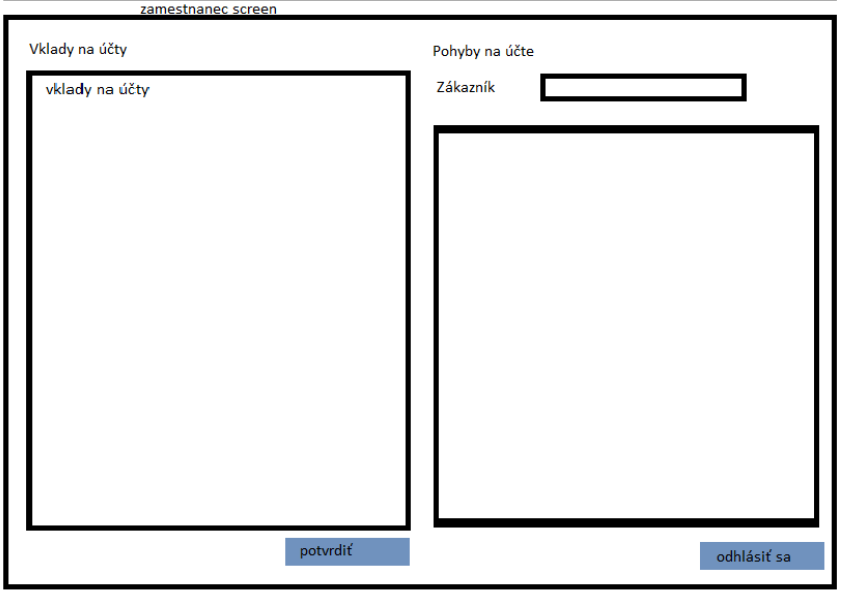


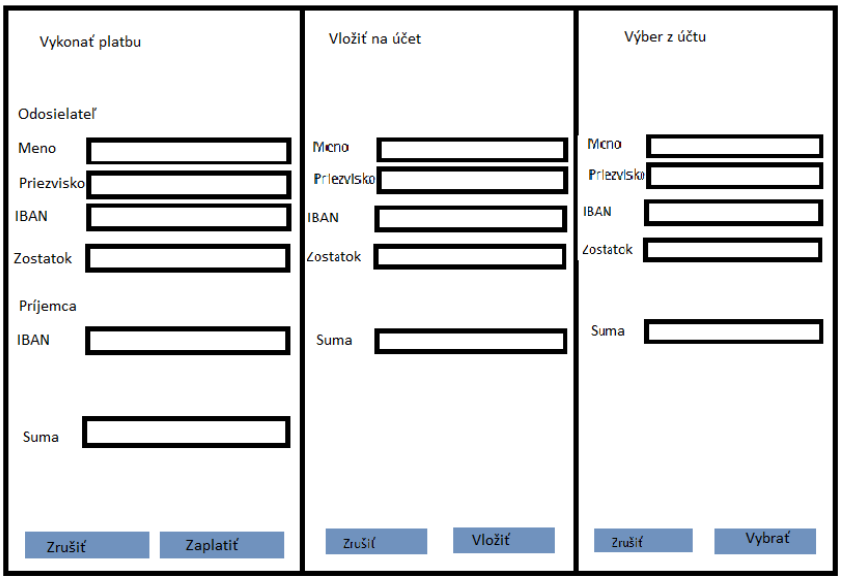
Michaela Gubovská a Jakub Hajdu
VAVA - Semestrálny projekt
Pondelok 16:00

registrácia zákazníka



zamestnanec screen





8. Nefunkčné požiadavky

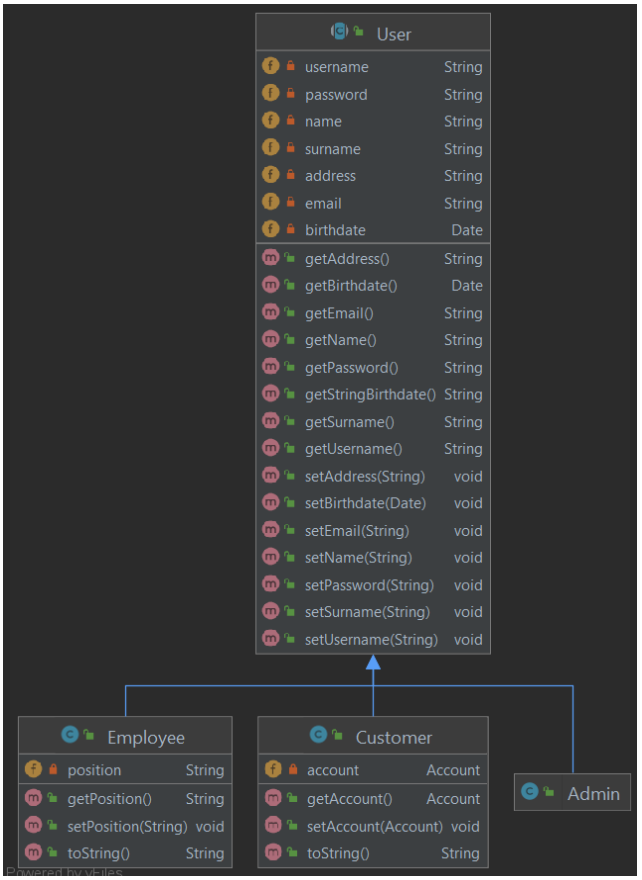
- **Prístupnosť**
  Aplikáciu musí byť možné spustiť na každom desktopovom zariadení.

- **Zabezpečenie používateľských údajov**
  Citlivé údaje o používateľoch vidia iba zamestnanci a admin. Admin môže vidieť aj nastavené heslo používateľa. Iní zákazníci nemôžu za žiadnych okolností získať prístup do konta iného zákazníka alebo získať údaje o kontách iných zákazníkov.

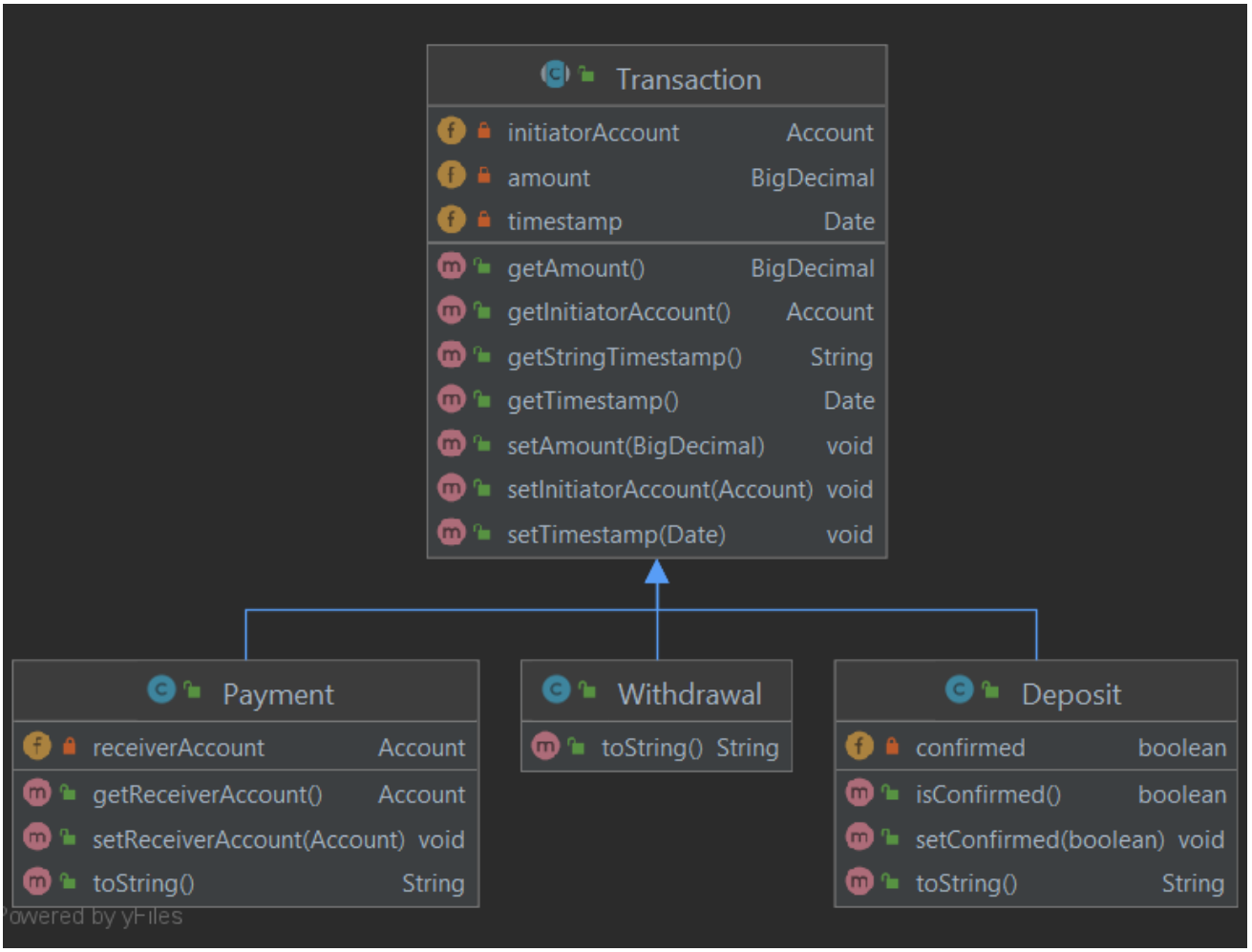
- **Intuitívna navigácia**
  Používateľa je schopný navigovať sa aplikáciou bez väčších problémov. Každú akciu je schopný vykonať s najviac tromi nesprávnymi kliknutiami. Aplikácia používateľa upozorní na každý nesprávny vstup a jasne vysvetlí, kde nastala chyba.

9. UML diagramy tried

Hierarchia používateľov je v aplikácii tvorená abstraktou triedou **User**, od ktorej dedia triedy **Admin**, **Employee** a **Customer**. Používatelia preto môžu byť inštanciou jednej z týchto troch tried. Všetci používatelia majú od triedy User spoločné atribúty meno, priezvisko, používateľské meno, heslo, adresa, email a dátum narodenia.
Ostatné atribúty používateľa potom závisia od toho, ktorej triedy je inštanciou. Zákazníkovi je navyše pridelený účet Account, zamestnancovi zasa pozícia a admin nemá nič navyše, je však potrebné pamätať si, ktorý používateľ je zrovna adminom a má tak prístup k citlivým údajom o každom používateľovi.



Hierarchia transakcií má na rovnakom princípu hlavnú abstraktnú triedu **Transaction** od ktorej dedia triedy **Payment**, **Deposit** a **Withdrawal**. Každá transakcia je preto inštanciou jednej z menovaných troch tried.
Od abstraktnej triedy **Transaction** má každá transakcia atribúty účet **Account** iniciujúci transakciu, suma a timestamp, teda dátum sa časom vykonania transakcie.
Trieda **Payment** má navyše pridaný atribút prijímajúci účet, nakoľko pri platbe dochádza k prevodu peňažnej čiastky medzi dvoma účtami. Trieda **Deposit** má navyše značku confirmed (potvrdený) hovoriacu o stave vkladu. Teda nadobúda hodnotu true pokiaľ bol vklad potvrdený zamestnancom (vtedy sa mení aj timestamp transakcie na čas potvrdenia zamestnancom). Hodnotu false má defaultne počas doby, kedy sa čaká na jeho potvrdenie zamestnancom banky. Trieda **Withdrawal** nemá pridaný žiadny ďalší atribút nakoľko žiadny netreba.



10. Splnenie požiadaviek na projekt

a. Kolekcie

Z kolekcií sme v aplikácii využili ArrayListy, v ktorých sú zapamätaní používatelia, transakcie zákazníkov, popup okná pre detail zákazníka alebo zamestnanca a výpis z účtu. Tieto ArrayListy sa vždy pri spustení naplnia na základe serializovaných objektov používateľov a transakcií.

```
private ArrayList<User> userList = new ArrayList<>();
private ArrayList<Transaction> transactionList = new ArrayList<>();
```

```
private final ArrayList<AccountDetail> accDetailList = new ArrayList<>();
private final ArrayList<EmployeeDetail> empDetailList = new ArrayList<>();
private final ArrayList<ReportPopup> reportPopupList = new ArrayList<>();
```

b. Logovanie

V aplikácii sa najdôležitejšie akcie používateľov zaznamenávajú v podobe logov do súboru log.txt, ale aj do konzoly. Na logovanie je využitá knižnica **java.util.logging**. Konkrétne pre zápis do .txt súboru využívame **FileHandler**. Logy sa pri každom spustení pridávajú na koniec log.txt súboru pre jednoduchšie debugovanie v procese používania, či pre možné neskoršie analyzovanie používateľského správania za nejaké obdobie.

Medzi logmi sa napríklad nachádza zaznamenanie spustenia/ukončenia aplikácie, nesprávne vyplnenie povinných polí pri registrácii používateľov alebo chýbajúce označenie používateľa, o ktorého konte si chce administrátor zobraziť detail.

```
private static final Logger logger = Logger.getLogger(MainFrame.class.getName());
private FileHandler loggerFileHandler;
```

```
logger.log(Level.WARNING, bundle.getString( key: "INCORRECT LOGIN CREDENTIALS WERE USED."));
```

```
Apr 25, 2021 6:31:46 PM view.MainFrame <init>
INFO: ---- APP SESSION OPENED ----
Apr 25, 2021 6:32:00 PM view.MainFrame btLoginActionPerformed
WARNING: Incorrect login credentials were used.
Apr 25, 2021 6:32:08 PM view.MainFrame btLoginActionPerformed
WARNING: Incorrect login credentials were used.
Apr 25, 2021 6:32:38 PM view.ReportPopup btSaveActionPerformed
WARNING: Invalid date range
Apr 25, 2021 6:32:56 PM view.MainFrame formWindowClosing
INFO: ---- APP SESSION CLOSED ----
```

c. Internacionalizácia

Aplikácia je dostupná v dvoch jazykoch - v slovenčine a angličtine. Predvolený jazyk pri spustení aplikácie je slovenčina. Jazyk sa dá kedykoľvek zmeniť na úvodnej obrazovke registrácie/prihlásenia.

Preklady aplikácie sa nachádzajú v dvoch ResourceBundle objektoch. Pre slovenskú verziu je Bundle_SVK a pre anglickú Bundle_ENG.

```
ŠTUDENTSKÝ=Študentský
BEŽNÝ=Bežný
STUDENT=Študentský
CURRENT=Bežný
SPAŤ=Späť
VYTVORENIE\ VÝPISU=Vytvorenie výpisu
```

Bundle_SVK

```
ŠTUDENTSKÝ=Student
BEŽNÝ=Current
STUDENT=Student
CURRENT=Current
SPAŤ=Back
VYTVORENIE\ VÝPISU=Create report
```

Bundle_ENG

d. XML

XML je využívané pri serializácii používateľov a transakcií.

Pri serializácií sa do serialization.xml súboru zaznamenávajú aktívni používatelia aplikácie, teda všetci existujúci zamestnanci vrátane admina a všetci zákazníci, ktorí majú v RayBank aktívne vedený účet. Odstránení zamestnanci a zrušené účty sa ďalej neevidujú a nie sú serializovaní.

Serializované sú dva ArrayListy - jeden s používateľmi a jeden s transakciami. Naša aplikácia si vďaka tomu po spustení vie načítať všetkých aktívnych používateľov a všetky transakcie, ktoré prebehli od prvotného nasadenia aplikácie na trh.

```java
private void serialize() {
    XMLEncoder encoder = null;
    try {
        encoder = new XMLEncoder(new BufferedOutputStream(new FileOutputStream(serializeFile)));
        encoder.setPersistenceDelegate(java.math.BigDecimal.class, encoder.getPersistenceDelegate(Integer.class));
    } catch(FileNotFoundException e) {
        System.out.println(bundle.getString( key: "NENAŠIEL SA SÚBOR PRE SERIALIZÁCIU."));
        logger.log(Level.SEVERE, bundle.getString( key: "SERIALIZATION FAILED. FILE NOT FOUND OR NOT CREATED."));
    }

    if (encoder != null) {
        encoder.writeObject(this.userList);
        encoder.writeObject(this.transactionList);
        encoder.close();
    }
}
```

e. I/O

Pri serializácii používateľov a transakcií sú na zápis a čítanie do/zo súboru serialization.xml použité knižnice:

- **java.io.BufferedInputStream**,
- **java.io.BufferedOutputStream**,
- **java.io.FileInputStream**,
- **java.io.FileOutputStream**.

```
private void deserialize() {
    XMLDecoder decoder = null;
    try {
        decoder = new XMLDecoder(new BufferedInputStream(new FileInputStream(serializeFile)));
    } catch(FileNotFoundException e) {
        System.out.println(bundle.getString( key: "NENAŠIEL SA SÚBOR PRE SERIALIZÁCIU, VYTVÁRA SA NOVÝ."));
        logger.log(Level.INFO, bundle.getString( key: "SERIALIZATION FILE WAS NOT FOUND, CREATING NEW FILE."));
    }
```

Okrem serializácie využívame I/O pri generovaní výpisu z účtu zákazníka za zvolené obdobie. Na tento zápis využívame knižnicu **java.io.FileWriter,** vďaka ktorej je umožnené zapísanie dát do .txt súboru, ktorý sa ukladá pod názvom “report_priezvisko_od-do.txt”.

```
FileWriter fw = new FileWriter(name);

fw.write(oddelovac);
fw.write( str: "| RayBank, " + currentDate + "\n|\n");
```

f. Regulárne výrazy

Regulárne výrazy si našli v našej aplikácii využitie pri overovaní validnosti zadanej emailovej adresy pri registrácii zákazníka a pridávaní nového zamestnanca. Pri emailovej adrese je totiž potrebné overiť, či zadané pole znakov má tvar “*text@text.doména*”. Rovnako nie sú v emailovej adrese povolené všetky znaky a je potrebné overiť, či zrovna používateľom zadané znaky sú povolené a tým pádom je aj zadaná emailová adresa s najväčšou pravdepodobnosťou reálna a legitímna.

Regulárny výraz na overenie emailu vyzerá nasledovne:

```
^[A-Za-z0-9'/{}|~?#:!$%_+-]+[A-Za-z0-9'/{}|~?#:!$%_+.-]*@[A-Za-
                z0-9'/{}|~?#:!$%_+-]+[.][a-z0-9-]+$
```

Aby bol zadaný výraz korektný na základe tohto regulárneho výrazu, musí na začiatku obsahovať minimálne jeden znak z množiny malých a veľkých písmen, čísel a zvyšných povolených znakov, okrem bodky. Ďalej už môže nasledovať aj bodka a potom musí nasledovať zavináč, minimálne

jeden znak z rovnakej množiny ako na začiatku, bodka a minimálne jeden znak z povolených znakov pre doménu, teda malé písmená, čísla a pomlčka.

Pokiaľ nie je zadaná emailová adresa verifikovaná týmto regulárnym výrazom, aplikácia zobrazí chybové okno s oznamom, že zadaná emailová adresa má zlý formát. Nepustí používateľa k ukončeniu registrácie/tvorby nového zamestnanca, pokiaľ email nebude zadaný v správnom formáte.

11. Zdroje

https://docs.oracle.com/javase/7/docs/api/java/beans/Encoder.html#setPersistenceDelegate(java.lang.Class,%20java.beans.PersistenceDelegate)

https://docs.oracle.com/javase/7/docs/api/java/beans/PersistenceDelegate.html

https://www.edureka.co/blog/serialization-of-java-objects-to-xml-using-xmlencoder-decoder/

http://tutorials.jenkov.com/java-regex/index.html#escaping-characters