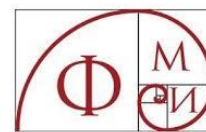


Софийски университет “Св. Климент
Охридски”
Факултет по математика и информатика



Курсов проект по “Системи за електронен бизнес” зимен семестър 2022/2023

Проект - Система за онлайн банкиране
Изработил: Веселин Славов Тодоров
Специалност: Информационни системи
Курс: 4
Факултетен номер: 71923
Рецензент: проф. гл-ас. Йоанис Патиас

Гр. София
Декември 2022

Съдържание:

1. Описание на учебния проект
2. Архитектура и дизайн на системата
 - 2.1. Потребителски слой
 - 2.2. Бизнес слой
 - 2.3. Слой на базата данни
 - 2.4. Диаграма на архитектурата
 - 2.5. Диаграма на потребителските случаи
3. Етапи на разработка:
4. Описание на е-бизнеса / е-услуга
 - 4.1. Обхват
5. Описание на цялостната реализация (на пример началната страница на сайта)
6. Описание на Базата от Данни
7. Описание на категориите и отделите
8. Описание на съхранените процедури
9. Демо сървър
10. Допълнение

1. Описание на учебния проект

Поради широкото разпространение на дигиталните услуги, финансовия сектор също бива засегнат от промените и вече почти всеки от нас има нужда от банкова сметка. Системите за онлайн банкиране обаче са прекалено сложни за използване от средностатистическия човек и целта на този проект е да имплементира система за онлайн банкиране в която лесно да можем да следим нашата сметка и да оперираме с нея лесно и удобно.

2. Архитектура и дизайн на системата

За разработката на системата за онлайн банкиране се използват съвременни технологии от Java и JavaScript света, които предоставят добра гъвкавост и лесна поддръжка на проекта. За всеки слой от архитектура се използват следните технологии описани отдолу:

2.1 Потребителски слой (Client layer)

- Angular - front end framework на javascript използващ typescript с отворен код, който подпомага по-лесна разработка на интерфейса, routing между отделните страници, имплементация на MVC архитектурата
- Bootstrap - client-side среда с отворен код, която съдържа набор от инструменти за създаване на уеб приложения и уеб сайтове. Състои от три компонента: CSS, Компоненти на потребителския интерфейс и Javascript.
- Typescript - безплатен език за програмиране с отворен код, разработен и поддържан от Microsoft. Това е строг синтактичен супернабор на JavaScript и добавя по избор статично писане на езика. Той е предназначен за разработка на големи приложения и транспилации към JavaScript. Използваме го за Angular компонентите.

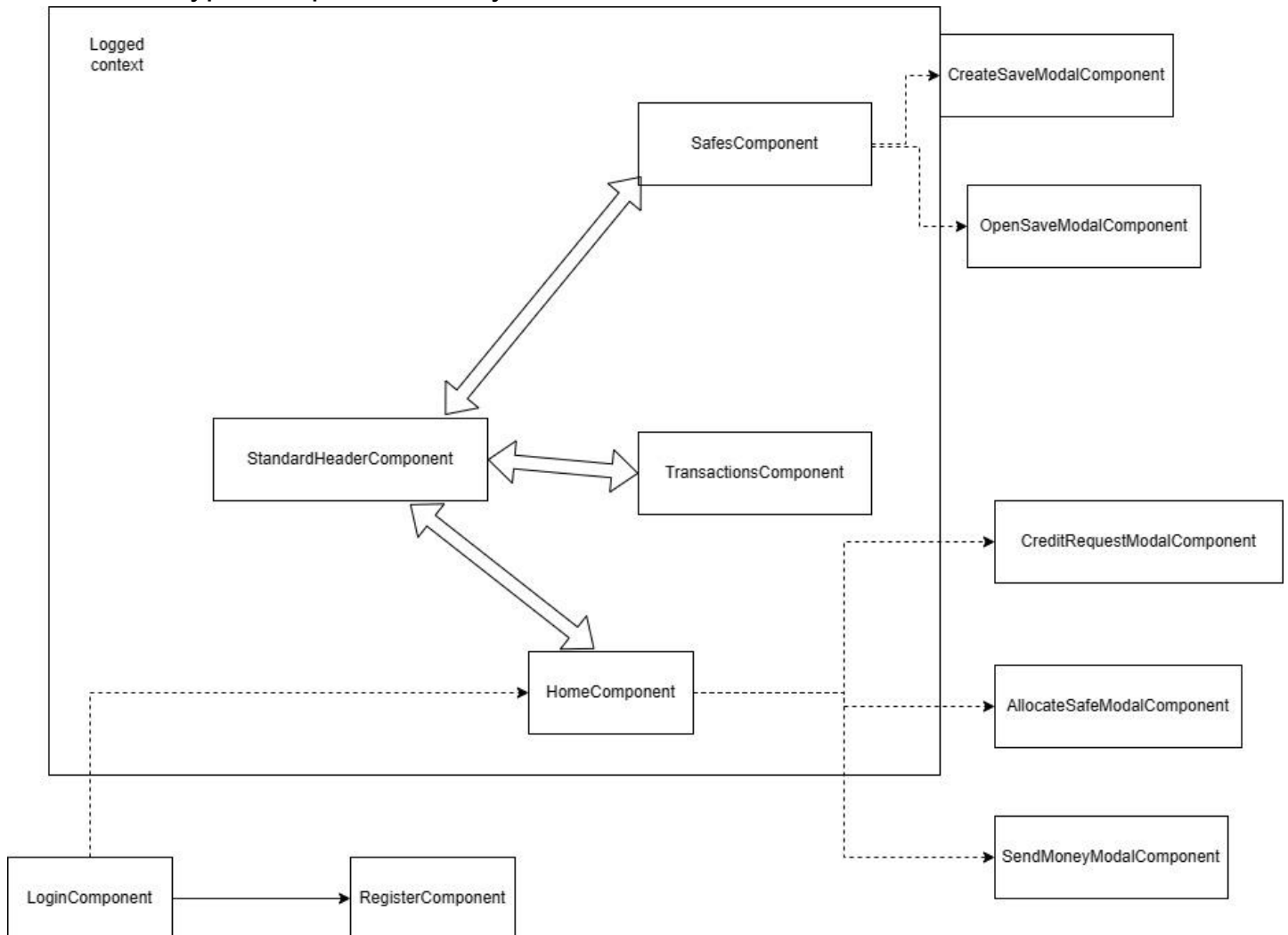
Angular компоненти в проекта:

- **LoginComponent** - използва се за началната страница в приложението, където има възможност да се логнем с потребителско име и парола и така да бъдем пренасочени към HomeComponent-а или директно към RegisterComponent.
- **RegisterComponent** - използва се страницата за регистрация в приложението, където има възможност да си създадем акаунт с потребителско име, парола, държава и така да бъдем пренасочени към HomeComponent-а директно логнати с новосъздадения профил
- **StandardHeaderComponent** - използва се header-а на HomeComponent, TransactionsComponent и SafesComponent, като има възможност да се пренасочим към всяка от тях както и към LoginComponent-а ако натиснем надписа Online Banking System.
- **HomeComponent** - използва се за home страницата в приложението, където спрямо потребителския профил с който сме влезли в системата се зарежда информация свързана с дебита по сметката, iban-а, сумата на всички кредити изтеглени от потребителя, сумата на сумите във всички сейфове на потребителя, баланс на сметка и всички транзакции, както и бутони отварящи модал за теглене на кредит, изпращане на пари по сметка (банков превод) и прехвърляне на пари по сейф. Във всички тези модали се изисква ауторизация с дебитна карта.
 - **CreditRequestModalComponent** - модал който се появява когато натиснем Request Credit бутона, това ни позволява да изтеглим кредит, като за да го изтеглим трябва да въведем сумата която искаме и за колко години искаме да го върнем. Също така има стандартна ауторизация на дебитна карта, която валидира дали сме въвели съществуваща карта от външната система за която сме свързани, както и дали това е нашата карта свързана с акаунта. При успешна валидация се зареждат изтеглените пари в нашия акаунт и се добавя транзакция с получател CREDIT

- **AllocateSafeModalComponent** - модал който се появява когато натиснем Allocate to safe бутона, това ни позволява да сложим пари в сейф, който да ги държи докато не го отворим, като за да ги изпратим трябва да въведем сумата която искаме и в кой сейф искаме да ги изпратим. Отново се прави ауторизацията с дебитна карта. При успешна валидация се премахват парите от нашия акаунт и се добавят в сейфа, ако имаме достатъчен баланс.
- **SendMoneyModalComponent** - модал който се появява когато натиснем Send money бутона, това ни позволява да изпратим пари на съществуващ потребител в системата, като за да ги изпратим трябва да въведем сумата която искаме да изпратим и iban на потребителя на когото искаме да ги изпратим. Отново се прави ауторизацията с дебитна карта. При успешна валидация се премахват парите от нашия акаунт и се добавят в акаунта на другия потребител, ако имаме достатъчен баланс.
- **TransactionsComponent** - използва се за transactions страницата в приложението, където спрямо потребителския профил с който сме влезли в системата се зарежда информация свързана с всички транзакции и пълната информация за тях, както и критерии по които да търсим транзакции, които са Receiver Iban, интервал от дати и дали системата да търси само входящи трансфера.
- **SafesComponent** - използва се за saves страницата в приложението, където спрямо потребителския профил с който сме влезли в системата се зарежда информация свързана с нашите сейфове, както и бутони за създаване на нов празен сейф и такъв за отваряне на сейф.
 - **CreateSaveModalComponent** - модал който се използва за създаване на нов сейф, чрез въвеждане на име и парола на сейфа. Когато се създаде той има 0 долара наличност.
 - **OpenSaveModalComponent** - модал който се използва за отваряне на съществуващ сейф, чрез въвеждане на

неговата парола. Ако паролата е валидна, то всички пари в сейфа се алокират към баланса и сейфа се изтрива от системата.

Фигура 1. Връзки между компонентите



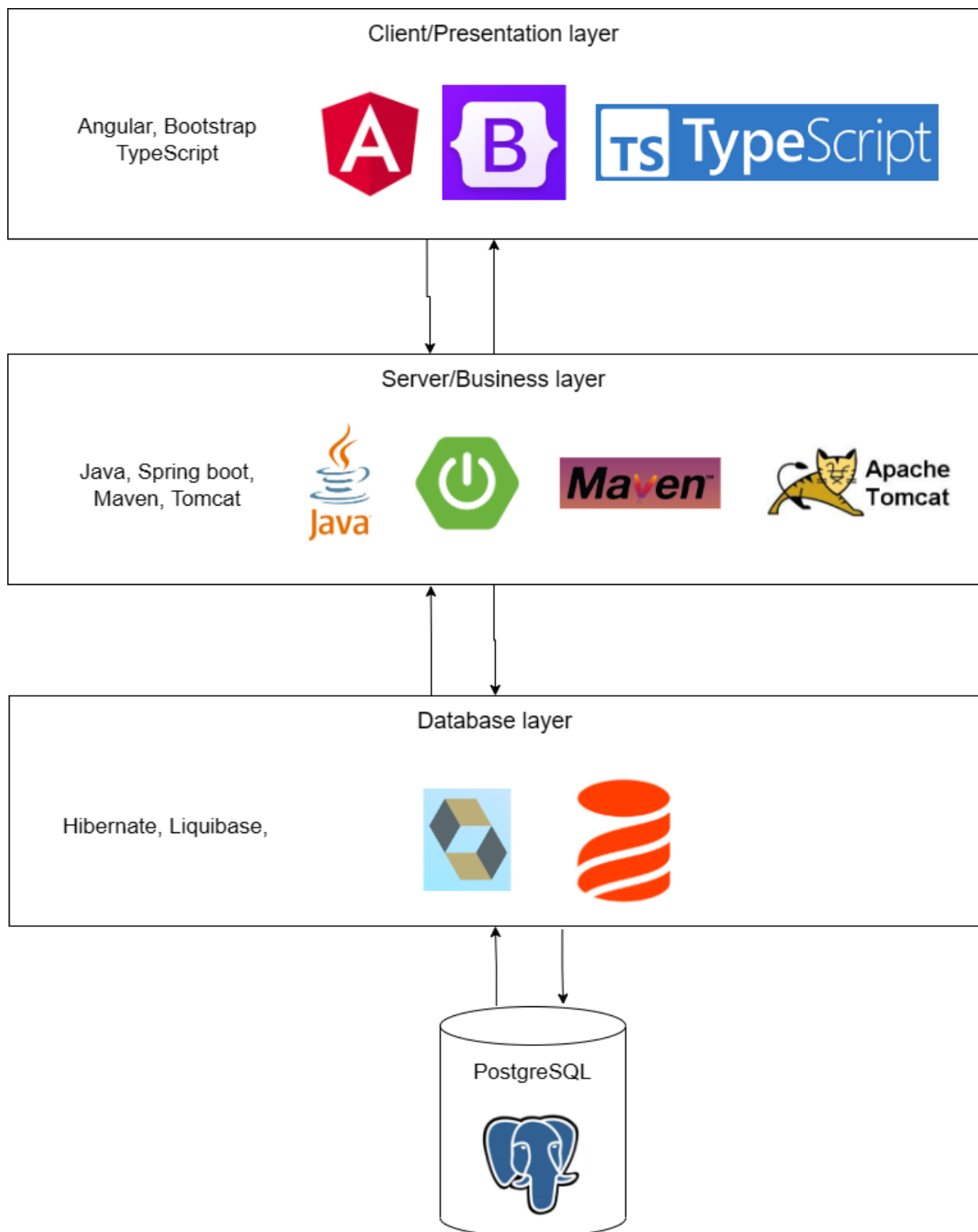
2.2 Бизнес слой (Service layer)

- Java 8 - безплатен език за програмиране с отворен код, поддържан от Oracle. Това е обектно ориентиран език от много високо ниво, поради поддръжката на концепции от функционалното програмиране. Използва се за основен език на бизнес сля
- Spring boot - open source backend framework на java, който подпомага по-лесна разработка на REST API на java (която архитектура се използва в настоящия проект). Избран е поради големината на проекта и факта че е надграден над Spring, като елиминира сложните конфигурации и допълнителните файлове използвани за Inversion of control и dependency injection. Използва само java анотации
- Maven - инструмент който се използва за build на проекта, и за управление на системите които използваме (например spring boot, hibernate, jpa и тн) и външните системи с които имаме връзка (в случая [BIN Checker API - API Layer](#)).
- Tomcat - инструмент който ни помага да пуснем приложението на локалния сървър. Използваме го тъй като spring boot има директна конфигурация за него

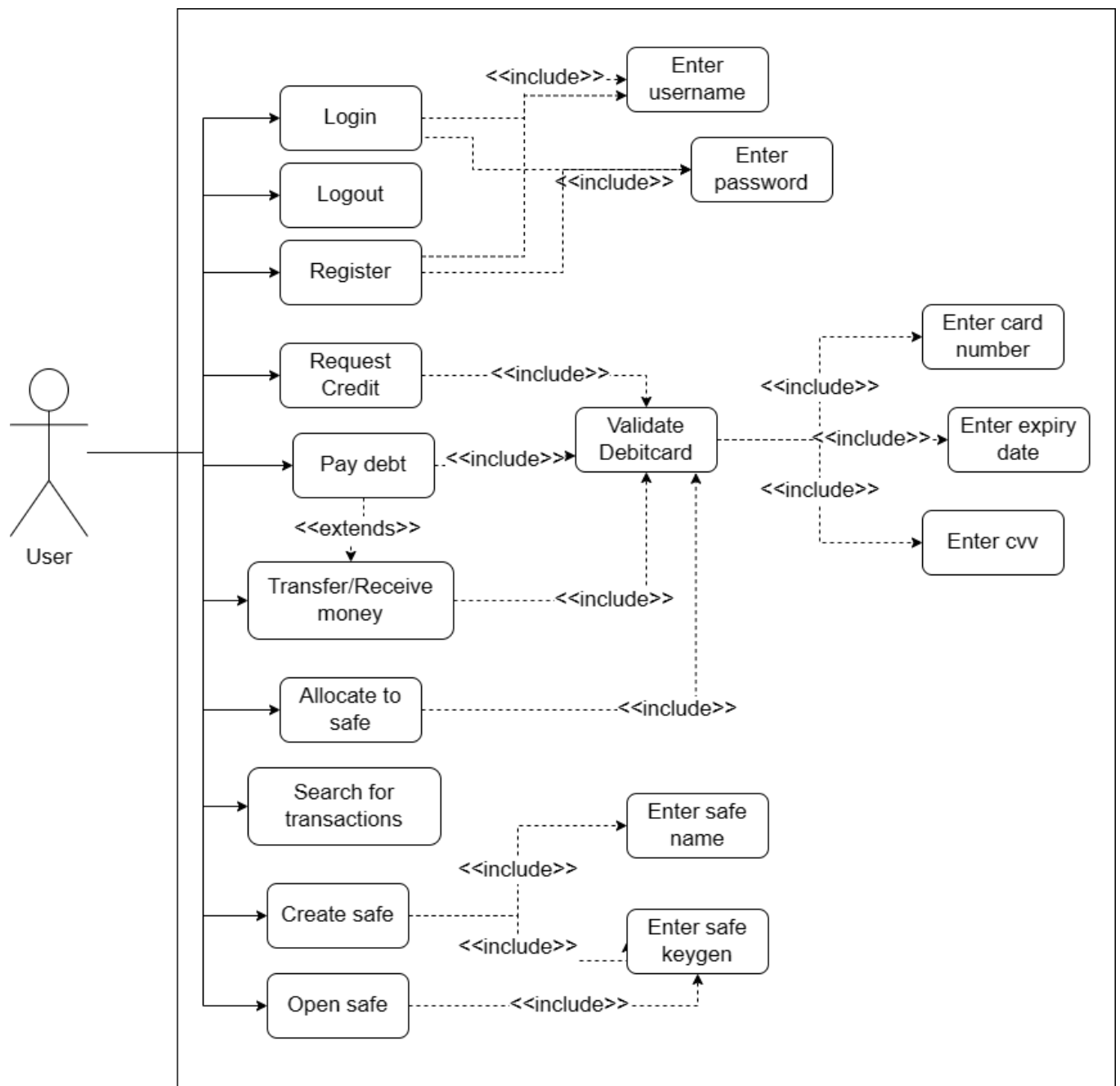
2.3 Слой на базата данни (Persistence layer)

- Hibernate - инструмент надграждащ JDBC използван от Java, за извършване на object relational mapping (ORM)
- PostgreSQL - релационна база от данни
- Liquibase - инструмент за управление на миграции на базата от данни

2.4 Архитектура на приложението



2.5 Диаграма на потребителските случаи



3. Етапи на разработка

- Първи етап - определяне на обхвата, създаване на REST API, реализиране на категориите на услугите, създаване на инстанция на базата данни и дефиниране на моделите/таблиците в базата данни.
- Втори етап - създаване на потребителски интерфейс и връзката между потребителския интерфейс и бизнес логическия слой
- Трети етап – Провеждане на тестове, интегриране на приложението и въвеждане на тестови данни

4. Описание на е-бизнеса / е-услуга

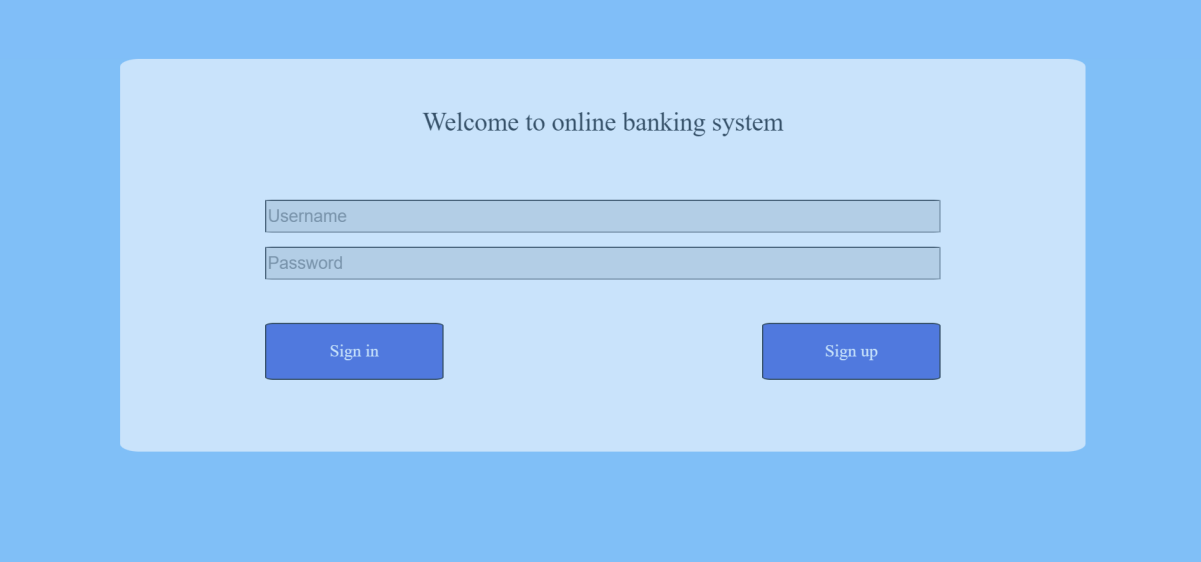
Съвременните банкови институции имат системи за онлайн банкиране. Проблемата при тези системи обаче е че са много трудни за използване от средностатистическия човек, който ги използва. Този проблем решава текущата система. При нея са оставени само най-простите и нужни операции, както и няколко новости свързани с управлението на личните финанси. Поддържа и връзка със система за верификация дали дебитна карта е валидна и издадена от банкова институция.

4.1 Обхват

- Регистрация
- Вход в профила (login)
- Изход от профила (logout)
- Изпращане на пари
- Получаване на пари
- Изтегляне на кредит
- Погасяване на кредит
- Търсене на транзакция

- Валидация на дебитна карта
- Създаване на сейф
- Алокация на пари към сейф
- Отваряне на сейф

5. Описание на цялостната реализация



The image shows a login interface for an online banking system. It features a light blue background with a central white rounded rectangle. Inside the rectangle, the text "Welcome to online banking system" is centered at the top. Below this, there are two input fields: "Username" and "Password". At the bottom of the rectangle, there are two buttons: "Sign in" on the left and "Sign up" on the right.

При отваряне на приложението пред потребителя се представя начална страница, като се дава възможност за избор между:

- Регистрация, ако потребителят все още не се е регистрирал в системата
- Вход в системата, чрез въвеждане на потребителско име и парола ако потребителя се е регистрирал в системата.

Register

Username

Password

Confirm password

Country

Sign up

При избор за регистрация трябва да въведем потребителско име, парола, потвърждение за паролата и страна. След това трябва да натиснем Sign up и автоматично ни регистрира и ни праща в нашия акаунт.

Online banking system

[Home](#) [Transactions](#) [Safes](#)

Available money

Debit: 353.4 USD BG0001

Credit: -70 USD BG0001

Safes: 42 USD BG0001

Balance: 325.4 USD BG0001

Operations

Request credit

Send money

Allocate to safe

Transactions

Debit	Receiver	Date
11 USD	EN001	2022-12-14
11 USD	BG014	2022-12-02
90 USD	CREDIT	2023-01-03
-20 USD	CREDIT	2023-01-03

След успешен вход в системата пред потребителя се отваря Home страницата на неговия профил, която съдържа информация за дебит, кредит, сума на всички сейфове, баланс и транзакции.

При натискане на Request credit се отваря следния модал за изтегляне на кредит

Credit request

Operations

X

Credit size

60000

Years

12

Request credit

Credit

60000

Send money

Card number

CVV

Allocate to add

Request credit

ID	Request	Date
BGD14	CREDIT	2022-12-14
CREDIT	CREDIT	2022-12-02
CREDIT	CREDIT	2023-01-03
CREDIT	CREDIT	2023-01-03

При натискане на Send money се отваря следния модал за прехвърляне на пари

Sending money

Operations

X

Receiver IBAN

Amount

Request credit

Reason

☐ Credit payment

Allocate to safe

Card number

CVW

Send money

EN001

BG014

CREDIT

amount

2022-12-14

2022-12-02

2023-01-09

2023-01-03

При натискане на Allocate to safe се отваря следния модал за изпращане на пари към даден сейф

Allocate to safe

X

Debit

Amount

Reason

Request credit

Credit

Safe name

Amount

Send money

Safe

Reason

Allocate to safe

Balance

Card number

CVV

Debit

Receiver

Date

Allocate

2022-12-14

2022-12-02

2023-01-03

2023-01-03

CREDIT

Екранът transactions филтрира по 3 критерия. Receiver IBAN, транзакции в интервал между 2 дати и дали са само входящи трансфери (получени пари от друг).

Online banking system

[Home](#)[Transactions](#)[Safes](#)

Transaction search

Receiver IBAN

Date from: mm/dd/yyyy

Date to: mm/dd/yyyy

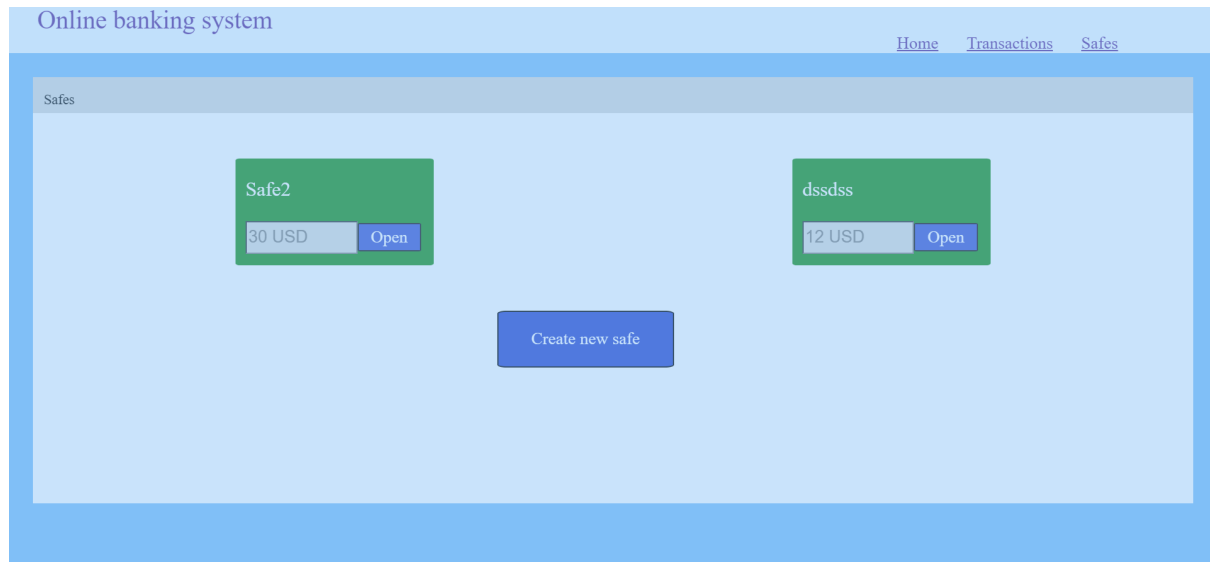
☐ Only positive debit

Search

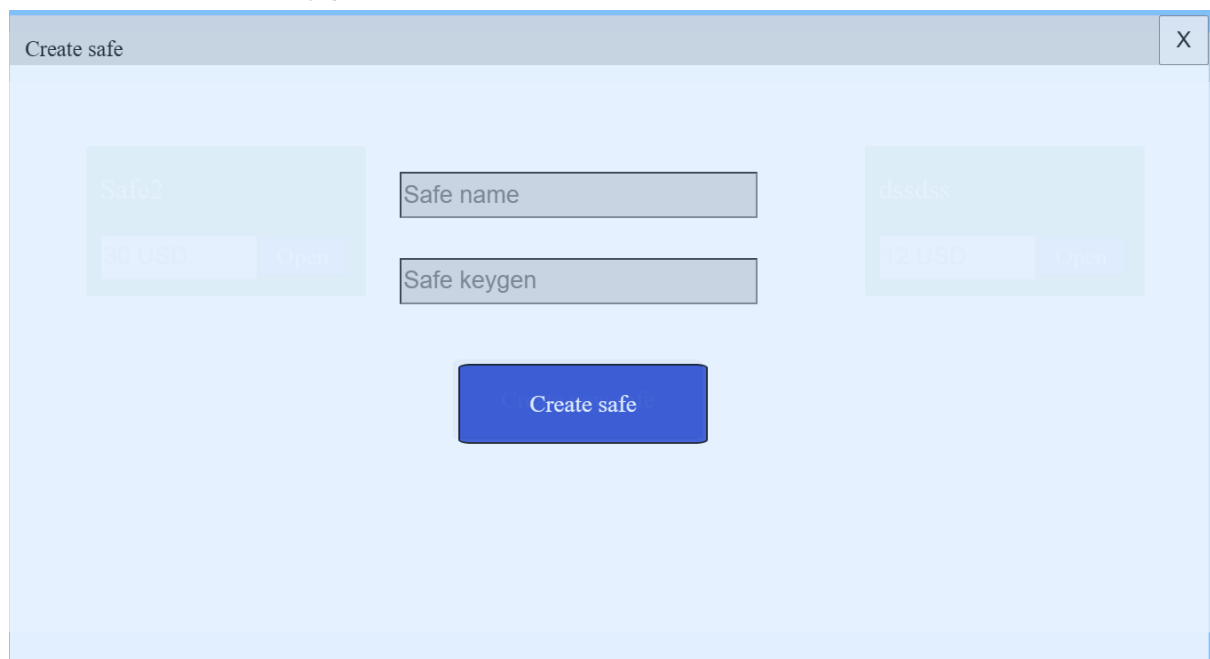
Transactions

Debit	Receiver	Reason	Date
-------	----------	--------	------

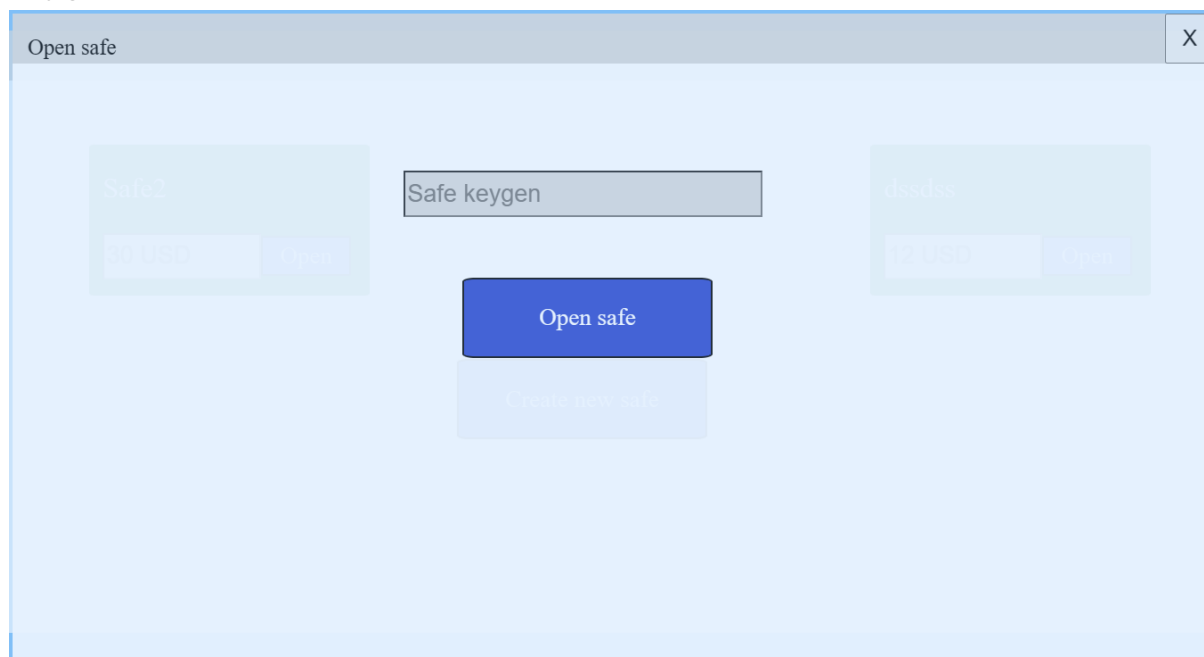
Екранът Safes държи всички сейфове на потребителя като показва имената и сумата в тях. Имаме възможност да създадем нов сейф чрез бутона Create new safe или да отворим съществуващ чрез бутона open на съответния safe.



Модалът за създаване на сейф изисква от потребителя да въведе име на сейф и keygen

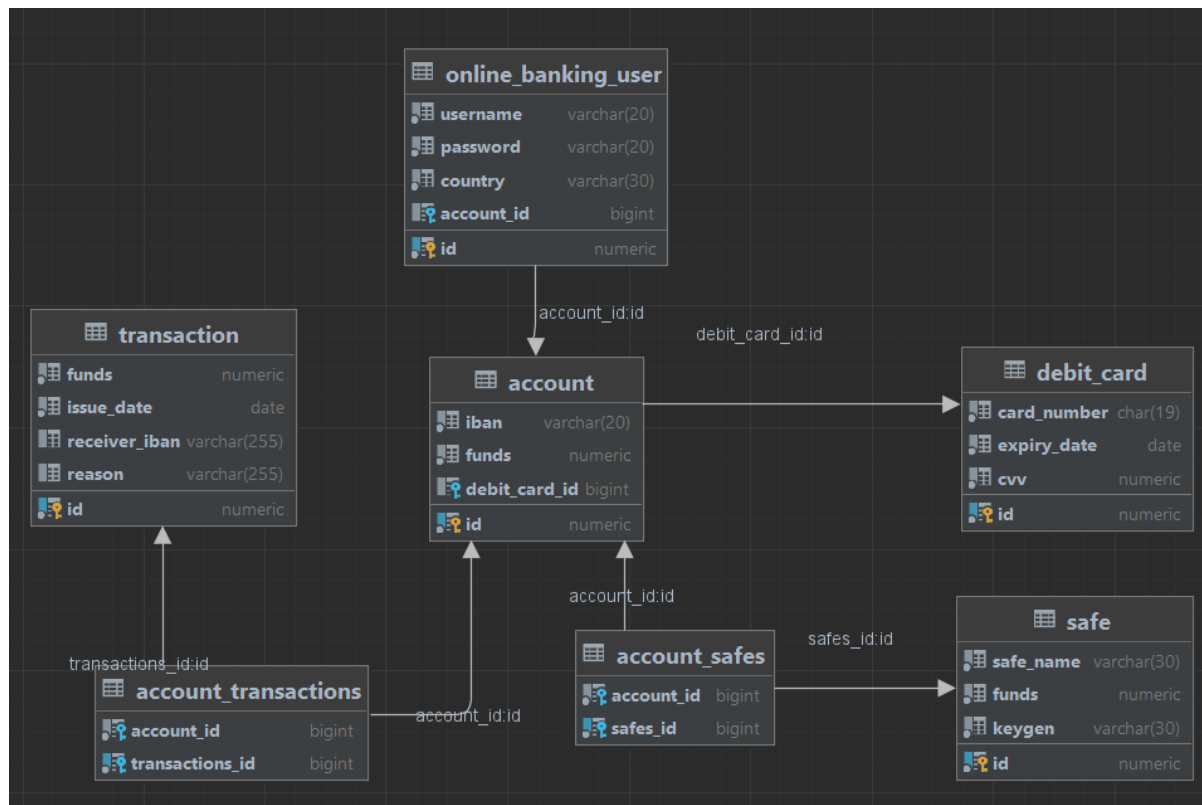


Модалът за отваряне на сейф изисква от потребителя да въведе keygen на сейф



6. Описание на Базата от Данни

6.1 E/R диаграма на базата данни



6.2 Описание на таблиците в базата данни

Таблица: Online banking user - Потребител

Име на колоната	Тип	Описание
id	number	Тази стойност се генерира автоматично от PostgreSQL.
username	string	Вписва се от потребителя при логване и регистрация. Задължително поле.

password	string	Вписва се от потребителя при логване и регистрация. Задължително поле.
country	string	Вписва се от потребителя при регистрация. Задължително поле.
account_id	number	Връзка с таблицата Account

Таблица: Account - Сметка

<i>Име на колоната</i>	<i>Тип</i>	<i>Описание</i>
id	number	Тази стойност се генерира автоматично от PostgreSQL.
iban	string	Идентификационния номер на потребителя, който се създава автоматично при създаване на потребителския профил. Задължително поле.
funds	number	Баланс по сметката Задължително поле.
debit_card_id	number	Връзка с таблицата Debit card

Таблица: Transaction - Дебитна карта

Име на колоната	Тип	Описание
id	number	Тази стойност се генерира автоматично от PostgreSQL.
card_number	string	Номер на картата Задължително поле.
expiry_date	date	Дата на изтичане на картата Задължително поле.
cvv	number	Card security code на картата Задължително поле.

Таблица: Safe - Сейф

Име на колоната	Тип	Описание
id	number	Тази стойност се генерира автоматично от PostgreSQL.
safe_name	string	Име на сейфа Задължително поле.
keygen	string	Парола на сейфа Задължително поле.
funds	number	Баланс по сейфа

		Задължително поле.
--	--	--------------------

Таблица: Transaction - Транзакция

<i>Име на колоната</i>	<i>Тип</i>	<i>Описание</i>
id	number	Тази стойност се генерира автоматично от PostgreSQL.
funds	number	Сума на транзакцията Задължително поле.
issue_date	date	Дата на извършване на транзакцията Задължително поле.
receiver_iban	string	Iban на сметката която получава трансфера Задължително поле
reason	string	Основание на трансфера Задължително поле.

Таблица: Account_Transactions - Връзка 1 към много между Account и Transactions

Име на колоната	Тип	Описание
account_id	number	Id на сметката Задължително поле.
transaction_id	number	Id на транзакцията Задължително поле

Таблица: Account_Safes- Връзка 1 към много между Account и Safes

Име на колоната	Тип	Описание
account_id	number	Id на сметката Задължително поле.
safes_id	number	Id на сейфа Задължително поле

7. Описание на категориите и отделите

- Категории за лицата, имащи отношение към системата:
 - Standard user - потребител, който има достъп в системата, чрез създаден от него профил
 - Admin - администратор
- Категории за съхранението на пари в системата:
 - Debit - баланс по картова сметка
 - Credit - баланс по кредитна сметка
 - Safes - баланс по сейфове
- Категории транзакции
 - Положителна
 - Отрицателна
 - Кредитна положителна
 - Кредитна отрицателна

8. Описание на съхранените процедури

- Online banking user
 - GET - Извличаме всички потребители, или потребител с име и парола
 - POST - Регистрация на нов потребител чрез потребителско име, парола, потвърждение на паролата и държава. Автоматично се генерират сметка и валидна дебитна карта към нея.
 - UPDATE - Обновяване на потребителя
 - DELETE - Изтриване на потребител по id
- Account
 - GET - Извличаме всички сметки, или сметка по IBAN
 - POST - Създаване на нова сметка. Генерират се IBAN спрямо данните на потребителя, funds е 0 по-подразбиране. Автоматично се генерират сметка и валидна дебитна карта към нея.
 - UPDATE - Обновяване на сметката

- DELETE - Изтриване на сметка по id
- Debit Card
 - GET - Извличаме всички дебитни карти, карта по всички характеристики. Връзка с външна система API layer, която по подаден номер на картата извлича BIN (Bank identification number) на картата и проверява дали картата е издадена от банкова институция работеща с VISA, MASTERCARD и AMERICAN CARDS.
 - POST - Създаваме дебитна карта с автоматични генерирани данни, валидни спрямо API layer системата
 - UPDATE - Обновяване на дебитната карта
 - DELETE - Изтриване на дебитна карта по id
- Transaction
 - GET - извличаме всички транзакции на потребител, както и филтрирани версии по различни критерии
 - POST - създаваме транзакция по подаден receiver iban, funds и reason.
 - UPDATE - Обновяване на транзакция
 - DELETE - Изтриване на транзакция по id
- Safe
 - GET - извличане на сейфове на даден потребител, извличане на сейф по потребител и име на сейфа
 - POST - създаване на сейф чрез въведено име и keygen, както и потребителско id
 - UPDATE - Обновяване на сейф
 - DELETE - Изтриване на сейф по id

9. Демо сървър

Приложението е разработено с Spring boot, използван е демо сървъра Tomcat, който помага с управлението на сесиите.

При настройките на сървъра той „слуша“ на порт 8080. Angular частта е на порт 4200.

10. Допълнение

За успешно стартиране на приложението

1. Клонирайте репото
2. Отваряме терминал
3. `cd client`
4. `npm install`
5. `npm start`
6. Отваряме втори терминал
7. `cd server`
8. `mvn spring-boot:run`