

ПРОФИЛИРАНА ПРИРОДОМАТЕМАТИЧЕСКА ГИМНАЗИЯ

“АКАДЕМИК НИКОЛА ОБРЕШКОВ” – ГРАД РАЗГРАД

Ул.Дъбрава 2, тел:0878684362, 0876207320, e-mail:pmgrz@abv.bg,

<http://www.pmgrz.net>

Тема: РАЗРАБОТКА НА УЕБ САЙТ НА Система за банково управление

Изисквания за разработката на дипломния проект (входни данни, съдържание, оформяне, указания за изпълнение, инструкции):

Разработване на система за банково управление, която да поддържа следните основни модули:

1. Управление на клиенти

- регистрация
- редакция
- изтриване на данни

2. Управление на банкови сметки

- отваряне
- затваряне
- обновяване на информацията за сметки

3. Операции със сметки

- депозити
- тегления
- прехвърляния

4. Справки и отчети

- възможност за генериране на информация за транзакции
- състояние на сметки

5. Системата трябва да може да се администрира с администраторски панел

Ученик: Сюлейман Синанов Писанцалиев

Клас: 12. г

GitHub хранилище: <https://github.com/wetto59/BankManangementSystem>

Ръководител-консултант: Венцислав Кочанов

Разград

2025 година

Задание

Да се разработи **уеб базирана банкова система**, която да осигурява на потребителите възможност за:

- Регистрация и вход в системата.
- Създаване и управление на банкови сметки в различни валути.
- Извършване на основни банкови операции като депозити, тегления и парични преводи.
- Преглед на историята на транзакциите.

Системата трябва да предоставя и администраторски панел за управление на потребители и техните сметки.

За реализацията на проекта да се използват:

- Python с уеб фреймуърка Flask
- HTML, CSS, JavaScript за визуализация на интерфейса
- MySQL за съхранение на данните

Проектът трябва да бъде функционален, защитен, с отзивчив дизайн и с добре структурирана база данни.

Съдържание

Увод	стр. 4
• Основна част	стр. 5
2.1. Формулиране на целта и задачите на проекта	стр. 5
2.2. Анализ на съществуващи решения	стр. 7
2.3. Описание на разработената система	стр. 10
2.4. Архитектура на приложението	стр. 13
2.5. Функционалности на потребителя	стр. 16
2.6. Функционалности на администратора	стр. 19
2.7. База данни: структура и връзки	стр. 22
2.8. Безопасност и защита на данните	стр. 25
2.9. Тестване и валидиране на системата	стр. 27
• Заключение	стр.

29

• Списък на използваната литература	стр. 30
• Приложения	стр.

31

Увод

В съвременния дигитален свят електронните банкови услуги заемат все по-важна роля в живота на хората и бизнеса. Необходимостта от сигурно, бързо и удобно управление на личните и бизнес финанси налага разработването на иновативни банкови системи. В тази връзка настоящият проект е насочен към създаването на уеб базирана банкова система — **Bank Management System**, която отговаря на съвременните изисквания за ефективност, сигурност и лесна употреба.

Основната цел на проекта е разработването на система, която предоставя на потребителите възможност за регистриране, създаване и управление на банкови сметки, извършване на транзакции (депозити, тегления и преводи), както и възможност за администраторите да управляват потребителите и контролират дейността в системата. Разработката обединява знания и умения в областта на уеб програмирането, базите данни и информационната сигурност.

Проектът използва популярни технологии като Python (Flask) за сървърната логика, HTML5, CSS3 и JavaScript за клиентската част, и MySQL за съхраняване на данни. Особено внимание е отделено на удобството на потребителския интерфейс и защитата на личните данни. Всички модули са разработени в съответствие със съвременните добри практики за уеб разработка.

Основните търсени резултати от разработването на системата са:

- Осигуряване на функционалност за управление на банкови сметки;
- Възможност за извършване на финансови транзакции;
- Разработване на административен панел за управление на потребители;
- Осигуряване на високо ниво на защита на потребителските данни.

Този проект е реален пример за приложение на придобитите знания в областта на информационните технологии и представлява основа за бъдещо надграждане и развитие в сферата на електронните финансови услуги. Системата е проектирана

да бъде лесна за използване както от индивидуални потребители, така и от администратори, с оптимизиран интерфейс за настолни и мобилни устройства.

Използването на технологии като Flask и MySQL осигурява бърза обработка на заявките и надеждно съхранение на данните. Проектът демонстрира прилагането на добри практики в програмирането — като структуриран код, използване на шаблони (Jinja2) и защита от SQL инжекции. Освен това системата е разработена с мисъл за бъдещо разширяване на функционалността чрез интегриране на допълнителни услуги като двуфакторна автентикация и аналитични инструменти за следене на активността.

ОСНОВНА ЧАСТ

1. ЦЕЛ И ЗАДАЧИ НА ДИПЛОМНИЯ ПРОЕКТ

Целта на дипломния проект е разработването на уеб базирана система за управление на банкови операции, която да предоставя удобен, сигурен и лесен за използване интерфейс за клиенти и администратори. Системата трябва да поддържа създаване и управление на банкови сметки, извършване на основни транзакции, както и администрация на потребители.

Основните задачи, поставени за решаване, са:

- Реализация на потребителска регистрация и вход;
- Създаване на възможност за откриване на различни типове банкови сметки;
- Осигуряване на механизми за депозиране, теглене и трансфер на средства;
- Разработване на административен панел за контрол върху потребителите;
- Гарантиране на сигурност при съхранение и обработка на чувствителна информация;
- Осигуряване на адаптивен (responsive) дизайн за работа на различни устройства.

2. АНАЛИЗ НА ИЗВЕСТНИ РЕШЕНИЯ

2.1. Съществуващи банкови системи

На пазара съществуват множество решения за електронно банкиране, като например:

- **PayPal** — една от най-популярните платформи за парични преводи и разплащания в света. [Източник: PayPal Official Website]
- **Revolut** — дигитална банкова услуга, предоставяща управление на сметки и разплащания чрез мобилно приложение. [Източник: Revolut Official Website]
- **Wise (TransferWise)** — платформа за международни парични преводи с по-ниски такси от традиционните банки. [Източник: Wise Official Website]

Тези решения са широко използвани, но често включват сложни функционалности, които затрудняват някои потребители. Настоящият проект цели създаване на опростен модел за управление на сметки, насочен към базови банкови операции.

2.2. Технологични избори

Избраната архитектура и технологии:

- **Flask** — лек уеб фреймуърк за Python, осигуряващ гъвкавост и бърза разработка;
- **MySQL** — стабилна релационна база данни с висока производителност;
- **HTML5/CSS3/JavaScript** — за създаване на приятелски към потребителя интерфейс;
- **Bootstrap** — за бърза разработка на адаптивен дизайн;
- **FontAwesome** — за визуално обогатяване на потребителския интерфейс.

Този технологичен стек е избран заради своята надеждност, популярност и съвместимост с целите на проекта.

3. ПРИНОСИ НА ДИПЛОМНИЯ ПРОЕКТ

Проектът реализира следните основни приноси:

3.1. Реализация на защитена потребителска регистрация и вход

Системата използва валидиран вход, защита срещу SQL инжекции и управление на сесии.

3.2. Управление на банкови сметки и транзакции

Потребителите могат лесно да създават сметки, да извършват депозити, тегления и преводи с минимални усилия.

3.3. Разработване на административен панел

Администраторите имат възможност да управляват потребители, да редактират данни, да преглеждат сметки и транзакции.

3.4. Осигуряване на високо ниво на сигурност

Използвани са защитени сесии, проверки на права за достъп и валидиране на потребителски вход.

3.5. Оптимизация за различни устройства

Проектът включва напълно адаптивен дизайн, гарантиращ лесна употреба както от компютър, така и от мобилни устройства.

Архитектура на проекта:

Backend (Python Flask):

- Контролери (routes) за всички действия: вход, регистрация, транзакции, създаване на сметки, управление на потребители.
- Модели за базата данни: Users, Accounts, Transactions.
- Форми с WTForms за въвеждане и валидация на данни.

Frontend (HTML, CSS, JS):

- Bootstrap за бърза стилизация
- FontAwesome за добавяне на икони към бутони
- Респонсив дизайн за мобилни устройства
- JavaScript за допълнителна интерактивност

Структура на проекта:

BankManagementSystem/

```
├─ static/
|   ├─ css/
|   └─ js/
├─ templates/
|   ├─ base.html
|   ├─ login.html
|   ├─ register.html
|   └─ dashboard.html
|   └─ ...
├─ app.py
├─ models.py
├─ forms.py
└─ database.sql
```

Описание на базата данни:

- users (id, username, email, password, role, status, first_name, last_name, date_of_birth, national_id, phone_number, address, citizenship)
- accounts (account_id, user_id, iban, balance, currency, account_type, created_at)
- transactions (transaction_id, sender_iban, recipient_iban, amount, transaction_type, description, timestamp)
- closed_accounts (archived information for closed accounts)

Примери за код:

Регистрация на потребител (Flask Route)

```
@app.route('/register', methods=['GET', 'POST'])
```

```
def register():
```

```
    form = RegistrationForm()
```

```
    if form.validate_on_submit():
```

```
        hashed_password =
generate_password_hash(form.password.data)

        user = User(username=form.username.data,
email=form.email.data, password=hashed_password)

        db.session.add(user)

        db.session.commit()

        flash('Регистрацията е успешна!', 'success')

        return redirect(url_for('login'))

    return render_template('register.html', form=form)
```

Превод на средства (Transfer Money)

```
@app.route('/transfer', methods=['POST'])
def transfer():

    sender_account =
Account.query.get(request.form['sender_account_id'])

    amount = float(request.form['amount'])

    recipient_iban = request.form['recipient_iban']


    if sender_account.balance >= amount:

        recipient_account =
Account.query.filter_by(iban=recipient_iban).first()

        if recipient_account:

            sender_account.balance -= amount

            recipient_account.balance += amount

            db.session.commit()

            flash('Успешен превод!', 'success')

        else:

            flash('Получателят не съществува.', 'danger')

    else:

        flash('Недостатъчна наличност.', 'danger')


    return redirect(url_for('dashboard'))
```

Описание на екраните:

Екран за вход

- Потребителят въвежда имейл/потребителско име и парола
- Визуални съобщения за грешка при неуспешен вход

Табло за клиента

- Преглед на всички негови сметки
- Бързи бутони за депозит, теглене, трансфер, затваряне на сметка

Администраторски панел

- Таблица със списък на всички потребители
- Филтър по статус: активен, деактивиран
- Преглед и редакция на потребителски профили

Анализ на сигурността

- Сесиите се валидират и имат тайм-аут
- Всички форми са защитени от CSRF атаки чрез WTForms
- Паролите се съхраняват хеширани с bcrypt
- SQL инжекциите са предотвратени чрез ORM (SQLAlchemy)

План за тестове

- Регистрационен процес — проверка за валидност на данни
- Вход с правилна/грешна парола
- Превод на средства с достатъчна и недостатъчна наличност
- Теглене и депозит
- Достъп до админ панел само за администратори

Бъдещи подобрения

- Добавяне на известия по имейл за всяка операция
- Двухфакторна автентикация при вход
- Мобилна апликация
- Генериране на банкови извлечения в PDF формат
- Отваряне на валутни депозити

Примери за тестови случаи

Тест	Очакван резултат	Статус
Регистрация на нов потребител	Успешна регистрация и вход	✓
Превод с недостатъчна наличност	Грешка "недостатъчна наличност"	✓
Вход с грешна парола	Съобщение за грешка	✓
Създаване на нова сметка	Успешно създаване	✓
Схема на базата данни (ERD)		

Заклучение

В резултат на разработването на дипломния проект беше създадена напълно функционираща уеб базирана система за управление на банкови операции – **Bank Management System**. Проектът успешно изпълни поставените цели и задачи, като осигури платформа, която комбинира удобство за потребителите, сигурност на данните и ефикасност на процесите.

Системата предоставя всички основни функции, очаквани от съвременна банкова платформа, включително регистрация на потребители, създаване и управление на сметки, извършване на финансови операции (депозити, тегления и преводи) и административно управление на потребители и транзакции. Особено внимание бе отделено на осигуряването на стабилна защита на данните и на създаването на адаптивен интерфейс, достъпен от различни устройства.

Проектът доказва значението на добрите практики при разработка на уеб приложения, като например използване на валидиран вход, структурирано програмиране и разделение между сървърна и клиентска логика. Съчетаването на Python, Flask, HTML5, CSS3, JavaScript и MySQL позволи създаването на стабилна и гъвкава система, отговаряща на съвременните технологични стандарти.

Бъдещи подобрения на системата биха могли да включват интеграция на двуфакторна автентикация за по-високо ниво на сигурност, автоматично известяване по имейл за нови транзакции, както и разработване на мобилно приложение за още по-голяма достъпност.

Разработката на Bank Management System допринесе значително за развитието на професионалните умения на екипа в областите на уеб програмирането, базите данни и информационната сигурност, и представлява стабилна основа за бъдещи проекти в сферата на електронните финансови услуги

Списък на използваната литература

- **Grinberg, M. Flask Web Development: Developing Web Applications with Python. O'Reilly Media, 2018.**
- **McFarland, D. HTML5 & CSS3: The Complete Reference. McGraw-Hill Education, 2014.**
- **Beighley, L. Head First SQL: Your Brain on SQL – A Learner's Guide. O'Reilly Media, 2007.**
- **Zakas, N. JavaScript for Web Developers. Wiley Publishing, 2011.**
- **MySQL Documentation. MySQL 8.0 Reference Manual. Oracle Corporation, <https://dev.mysql.com/doc/>.**
- **Flask Documentation. The Pallets Projects. <https://flask.palletsprojects.com/>**
- **W3C. HTML and CSS Standards. World Wide Web Consortium (W3C), <https://www.w3.org/>**
- **Mozilla Developer Network (MDN). JavaScript Guide. <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>**
- **Bootstrap Documentation. Bootstrap 5.0 Framework. <https://getbootstrap.com/>**
- **FontAwesome Documentation. Icons for the Web. <https://fontawesome.com/>**

ПРИЛОЖЕНИЕ 1
(Примерна структура на база данни)

Схема на базата данни на Bank Management System

- **Таблица users**

- id (INT, PRIMARY KEY, AUTO_INCREMENT) – Идентификатор на потребителя
- username (VARCHAR) – Потребителско име
- email (VARCHAR) – Имейл адрес
- password (VARCHAR) – Хеширана парола
- first_name (VARCHAR) – Собствено име
- last_name (VARCHAR) – Фамилия
- date_of_birth (DATE) – Дата на раждане
- national_id (VARCHAR) – ЕГН
- phone_number (VARCHAR) – Телефонен номер
- address (VARCHAR) – Адрес
- citizenship (VARCHAR) – Гражданство
- status (ENUM) – Статус (active, inactive, suspended)
- role (ENUM) – Роля (customer, admin)
- created_at (TIMESTAMP) – Дата на създаване

- **Таблица accounts**

- account_id (INT, PRIMARY KEY, AUTO_INCREMENT) – Идентификатор на сметка
- user_id (INT, FOREIGN KEY) – Собственик на сметката

- `iban` (VARCHAR) – IBAN номер
- `account_type` (ENUM) – Тип сметка (savings, checking, business)
- `balance` (DECIMAL) – Баланс
- `currency` (VARCHAR) – Валута
- `created_at` (TIMESTAMP) – Дата на създаване
- **Таблица `transactions`**
 - `transaction_id` (INT, PRIMARY KEY, AUTO_INCREMENT) – Идентификатор на транзакция
 - `sender_iban` (VARCHAR) – IBAN на изпращача
 - `recipient_iban` (VARCHAR) – IBAN на получателя
 - `amount` (DECIMAL) – Сума
 - `transaction_type` (ENUM) – Тип на транзакцията (deposit, withdraw, transfer)
 - `description` (TEXT) – Описание
 - `timestamp` (TIMESTAMP) – Дата и час на транзакцията
- **Таблица `closed_accounts`**
 - `closed_id` (INT, PRIMARY KEY, AUTO_INCREMENT)
 - `account_id` (INT) – ID на закритата сметка
 - `closed_at` (TIMESTAMP) – Дата на закриване

ПРИЛОЖЕНИЕ 2
(Кодови примери от проекта
Система за банково управление)

2.1. Регистрация на потребител (Python/Flask)

```
@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = generate_password_hash(request.form['password'])
        first_name = request.form['first_name']
        last_name = request.form['last_name']

        # Вмъкване в базата данни
        cursor = mysql.connection.cursor()
        cursor.execute("""
            INSERT INTO users (username, email, password, first_name, last_name)
            VALUES (%s, %s, %s, %s, %s)
            """, (username, email, password, first_name, last_name))
        mysql.connection.commit()
        cursor.close()

        flash('Успешна регистрация!', 'success')
        return redirect(url_for('login'))
    return render_template('register.html')
```

2.2. Вход в системата (Python/Flask)

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        login_input = request.form['login_input']
```

```

password = request.form['password']

cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
cursor.execute('SELECT * FROM users WHERE username = %s OR email = %s',
(login_input, login_input))
user = cursor.fetchone()
cursor.close()

if user and check_password_hash(user['password'], password):
    session['logged_in'] = True
    session['user_id'] = user['id']
    session['username'] = user['username']
    session['role'] = user['role']
    flash('Успешен вход!', 'success')
    return redirect(url_for('index'))
else:
    flash('Грешно потребителско име или парола.', 'danger')

return render_template('login.html')

```

2.3. Извършване на депозит (Python/Flask)

```

@app.route('/deposit/<int:account_id>', methods=['GET', 'POST'])
@login_required
def deposit(account_id):
    cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
    cursor.execute('SELECT * FROM accounts WHERE account_id = %s',
(account_id,))
    account = cursor.fetchone()

    if request.method == 'POST':
        amount = float(request.form['amount'])

```

```
# Актуализиране на баланса
new_balance = account['balance'] + amount
cursor.execute('UPDATE accounts SET balance = %s WHERE account_id = %s',
(new_balance, account_id))

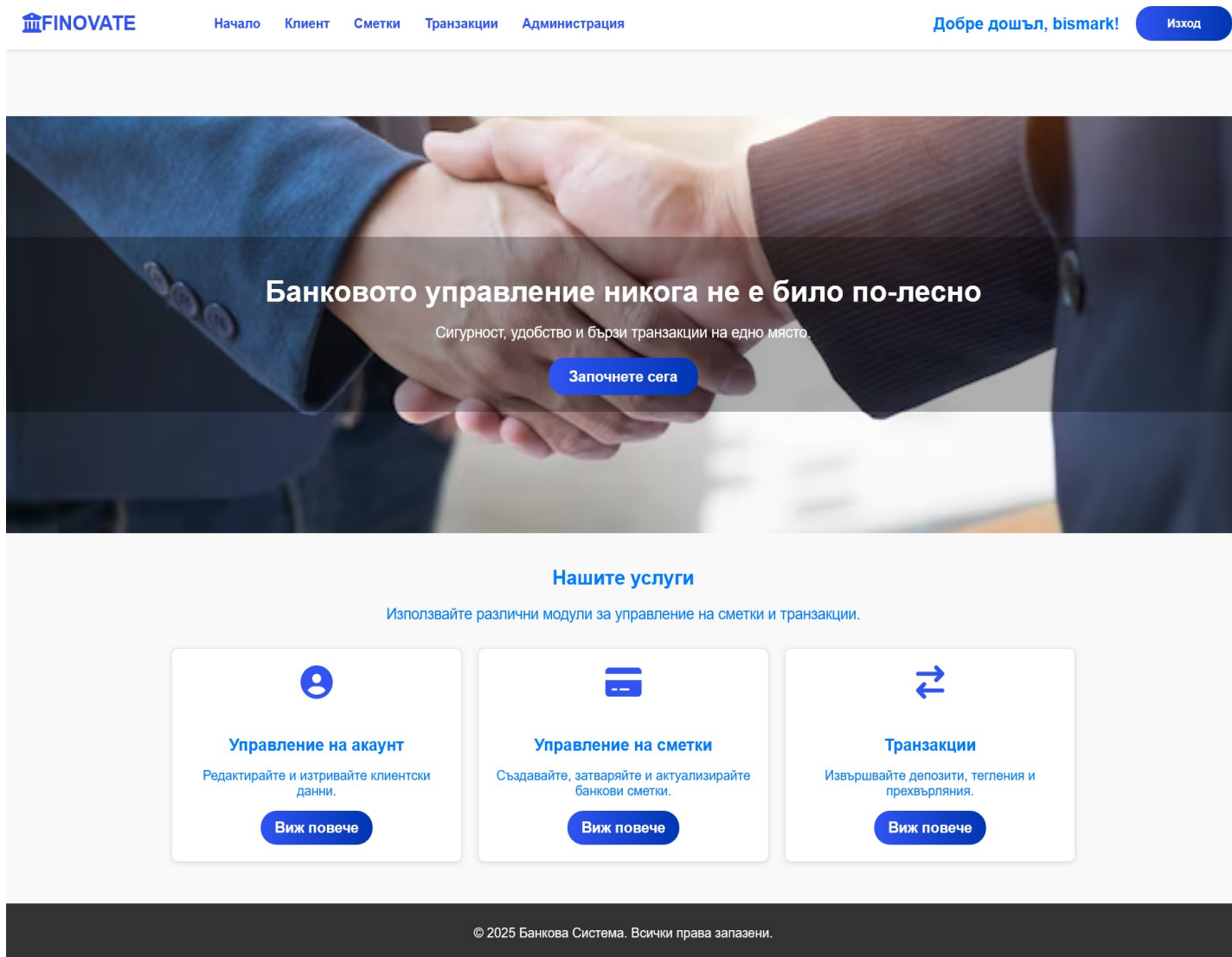
# Добавяне на транзакция
cursor.execute("""
    INSERT INTO transactions (sender_iban, recipient_iban, amount,
transaction_type, description)
    VALUES (%s, %s, %s, 'deposit', 'Депозит към собствената сметка')
    """, (account['iban'], account['iban'], amount))

mysql.connection.commit()
flash('Успешен депозит!', 'success')
return redirect(url_for('user_accounts'))
cursor.close()

return render_template('deposit.html', account=account)
```

ПРИЛОЖЕНИЕ 3
(Екранни снимки и описание на
интерфейса на Система за банково управление)

3.1. Начална страница на сайта (Home Page)



Описание:

На началната страница потребителят вижда навигационно меню, банер с надпис "Finovate" и основните бутони за достъп до:

- Клиенти (профил)
- Сметки

- Транзакции

При наличие на регистрация, автоматично се показва и поздравително съобщение с потребителското име.

3.2. Страница за регистрация (Register)

Регистрация

Моля, попълнете всички полета

Потребителско име

Имейл

Парола

Потвърдете паролата

Име

Фамилия

Дата на раждане

dd / mm / yyyy

ЕГН

Телефонен номер

Адрес

Гражданство

Регистрация

Вече имате акаунт? [Влезте тук](#)

Описание:

Формата съдържа полета за въвеждане на:

- Потребителско име
- Имейл
- Парола
- Име и фамилия
- Дата на раждане
- ЕГН
- Телефонен номер
- Адрес
- Гражданство

Формата използва красив и модерен дизайн с валидации на задължителните полета.

3.3. Страница за вход (Login)

Вход в системата

Моля, влезте в профила си

Потребителско име / Имейл / Телефон:

Парола:

Влез

Нямате акаунт? [Регистрирайте се тук](#)

Описание:

Потребителите въвеждат своето потребителско име, имейл или телефон и парола, за да се идентифицират в системата.

Има линк към регистрация за нови потребители.

3.4. Клиентски панел: Моите сметки (User Accounts)

FINOVATE

Добре дошъл, bismark!

Изход

Моите банкови сметки

IBAN	Тип	Баланс	Валута	Дата на създаване	Действия
BG12FINV508924882935	Бизнес	4980.00	EUR	2025-03-01 10:41:41	<div>Преглед</div> <div>Депозит</div> <div>Теглене</div> <div>Закриване</div>
BG12FINV674814794903	Спестовна	10000.00	USD	2025-03-01 10:41:47	<div>Преглед</div> <div>Депозит</div> <div>Теглене</div> <div>Закриване</div>
BG12FINV968385169502	Разплащателна	8388.00	EUR	2025-03-04 08:11:06	<div>Преглед</div> <div>Депозит</div> <div>Теглене</div> <div>Закриване</div>

Създай нова сметка


© 2025 Finovate Bank. Всички права запазени.


Описание:

Потребителите виждат списък с техните банкови сметки в таблица:

- IBAN
 - Тип на сметката (разплащателна, спестовна, бизнес)
 - Баланс
 - Валута
 - Дата на създаване
- Има бутони за Преглед, Депозит, Теглене, Закриване на сметка.


3.5. Страница за депозит в сметка (Deposit)


 **FINOVATE** Добре дошъл, bismark! Изход


 **Депозит в сметка**

IBAN: BG12FINV508924882935

Баланс: 4980.00 EUR

 Въведете сума:

 **Депозирай**


 **Назад**

© 2025 Finovate Bank. Всички права запазени.

Описание:


Форма за въвеждане на сума за депозит в избрана банкова сметка. След потвърждение, балансът на сметката се актуализира автоматично.


3.6. Страница за теглене от сметка (Withdraw)


 **FINOVATE** Добре дошъл, bismark! Изход

Теглене от сметка

IBAN: BG12FINV508924882935
Баланс: 4980.00 EUR

 **Въведете сума:**

 **Теглене**


 **Назад**

© 2025 Finovate Bank. Всички права запазени.


Описание:

Форма за въвеждане на сума за теглене. При теглене, системата проверява дали наличният баланс е достатъчен и при успех актуализира сметката.


3.7. Страница за паричен превод (Transfer)

 **FINOVATE** Добре дошъл, bismark! Изход


Извършване на превод

 **Изберете сметка:**


Изберете сметка

 **IBAN на получателя:**

BG00XXXX123456789

 **Сума:**

100.00

 **Описание (по избор):**

Например: Наем за март

▶ ИЗПРАТИ

ОТКАЗ

© 2025 Finovate Bank. Всички права запазени.

Описание:


Форма за превеждане на сума от избрана собствена сметка към друг IBAN.

Изисква се въвеждане на:

- IBAN на получателя

- Сума
- Описание на превода (по избор)

3.8. Администраторски панел (Admin Dashboard)

 **FINOVATE** Добре дошъл, bismark! Изход

Администраторски панел

Всички статуси ▾

Добави потребител

ID	ПОТРЕБИТЕЛСКО ИМЕ	ИМЕЙЛ	ИМЕ	ФАМИЛИЯ	ДАТА НА РАЖДАНЕ	ЕГН
7	sulmann	suleyman17nn@gmail.com	sulman	pisantsaliev	1999-11-11	000000
8	bismark	bismark@gmail.com	Oto Fon	Bismark	1999-01-01	123333
10	ivann	Ivan@gmail.com	Ivann	Nakov	1999-11-11	123456
11	alyaovlu	alyaovlu@abv.bg	Ava	Alyaovlu	2007-09-01	074901
12	sude	sude12@abv.bg	sude	pisantsalieva	2012-09-06	111111

1

© 2025 Finovate Bank. Всички права запазени.

Описание:

Администраторът може:

- Да вижда списък с всички потребители

- Да редактира потребители
- Да деактивира/активира профили
- Да изтрива потребители
- Да преглежда потребителски сметки и транзакции

Панелът е с мощни филтри за търсене по статус или роля.

ПРИЛОЖЕНИЕ 3

(ERD Модел на базата данни)

ER диаграма на базата данни за "Система за банково управление"

Легенда:

- **РК** – първичен ключ (*Primary Key*)
- **FK** – външен ключ (*Foreign Key*)

Описание:

- Таблицата users съдържа информация за всички регистрирани потребители.
- Всеки потребител може да има множество банкови сметки (accounts).
- Всяка банкова сметка може да има множество транзакции (transactions) като изпращач или получател.
- При закриване на сметка, данните се прехвърлят в таблицата closed_accounts за архив.

+-----+		+-----+	
users		accounts	
+-----+		+-----+	
id (PK)		user_id (FK)	
username		account_id (PK)	
email		iban	
password		account_type	
first_name		balance	
last_name		currency	
date_of_birth		created_at	
national_id		+-----+	
phone_number			
address		v	
citizenship		+-----+	
role		transactions	
status		+-----+	
created_at		transaction_id(PK)	
+-----+		sender_account_id	
		recipient_iban	
		amount	
		transaction_type	
		description	
		timestamp	
		+-----+	
+-----+			
closed_accounts			
+-----+			
closed_id (PK)			
account_id (FK)			