Документация Alexander

Финальный проект	**Исполнитель:**
Гладких А.А.	

Ссылка на задчу	JUS-234: Финальный проект TO DO
Статус задачи	В работе
Описание	Тестирование веб приложения банка Ba nk Saint-Petersburg с целью обеспечения его корректной работы и удобства использования вышеуказанного продукта.

Тест-кейсы
JUS-871: Авторизация в личном кабинете интернет банка "BSPB". TO DO
JUS-870: Проверка перевода с банковской карты 4797 72** ****2224 - Золота на банковскую карту 497 29** **** 6192 - Проезд на сумму 100 000 тысяч рублей 00 копеек. TO DO
JUS-868: Открытие вклада сроком 1 года со ставкой 14.15% годовых на сумму 10 000 тысяч рублей. TO DO
JUS-867: Проверка оформление кредита на сумму 100 000 тысяч рублей 00 копеек со ставкой 11% на срок 6 месяцев. TO DO

Баг-репорты
JUS-856: В разделе "Платежи и переводы" функция перевода по номеру телефона недоступна. ГОТОВО
JUS-857: В личном кабинете некоработает функция "оформление кредитной карты". ГОТОВО
JUS-855: В разделе "Валюты" не проходит автоматическая конвертация денежных средств. ГОТОВО

Описание фичей, с которыми предоставлены требования и описания требований.

Описание фичей

Описание фичей

1. Страница авторизации подьзователя: ввод логина и пароля - demo/demo
2. Проверка корректности введенных данных - кнопка “Войти” должна быть кликабельной только при заполненных корректных данных.
3. Подтверждение кода для входа в личный кабинет - поле для ввода кода “0000”.
4. Экран “Переводы”, раздел “На карту” кнопка “Дальше”.
5. Раздел “ВКЛАДЫ” заполнение формы с указанием суммы 10 000 тысяч рублей 00 копеек сроком на 1 год, подтверждение условий, открытие вклада.
6. Раздел “КРЕДИТЫ” заполнение формы с указанием суммы 100 000 тысяч рублей 00 копеек, процентной ставки 11%, сроком на 6 месяцев, проверка и ознакомление с условиями кредита, получение результата.

Требования

Требования

1.1. Логин

- **Требование**: Логин должен содержать от 4 до 20 символов.
- **Дополнительно**: Логин должен состоять из латинских букв.

1.2. Пароль

- **Требование**: Пароль должен содержать от 5 до 15 символов.
- **Дополнительно**: Пароль должен состоять из латинских букв.

1.3. Переход к окну входа через SMS

- **Требование**: При введении валидных значений логина и пароля осуществляется переход к окну входа через SMS.

1.4. Блокировка аккаунта

- **Требование**: При трех неудачных попытках входа аккаунт блокируется на 2 часа.

1.5. Верификация номера для SMS

- **Требование**: Номер для SMS должен быть верифицирован в настройках аккаунта.

2. **Требования к коду SMS**

2.1. Код

- **Требование**: Код должен состоять из четырех цифр.

2.2. Повторная отправка кода

- **Требование**: Повторная отправка кода осуществляется каждые 120 минут.

2.3. Блокировка аккаунта при неудачных попытках

- **Требование**: При трех неудачных попытках ввода кода аккаунт блокируется на 2 часа.

2.4. Успешный вход

- **Требование**: При удачном введении кода и нажатии кнопки “Войти” осуществляется переход на главную страницу веб-приложения.

3. **Переход на страницу "Платежи и переводы"**

3.1. Переход на страницу

- **Требование**: Переход с главной страницы веб-приложения на страницу “Платежи и переводы” осуществляется посредством нажатия на иконку в правом верхнем углу “в виде подарка”.

4. **Раздел "Нам важно знать ваше мнение"**

4.1. Переход в блок

- **Требование**: Переход в блок “Переводы” осуществляется путем нажатия на вкладку “Платежи и переводы”.

4.2. Отображение информации

- **Требование**: При переходе в раздел “Переводы” отображается информация о вкладке “На карту”.

4.3. Язык отображения

- **Требование**: При переходе в раздел “Переводы” текст отображается на русском языке.

4.4. Перевод текста

- **Требование**: При помощи кнопки RUS/ENG, расположенной внизу каждой страницы сайта, текст переводится на русский или английский языки по выбору пользователя.

5. **Корректное отображение дропдауна в разделах "С карты", “На карту”**

5.1. Переход в дропдаун

- **Требование**: при нажатии на дропдаун в разделах “С карты” и “На карту отображаются доступные счета пользователя.
- **Дополнительно:** номера карт должны отображаться цифрами.

5.2. Отображение текстового поле “Сумма”

- **Требование**: В разделе “Сумма” ввод осуществляется посредством цифр.

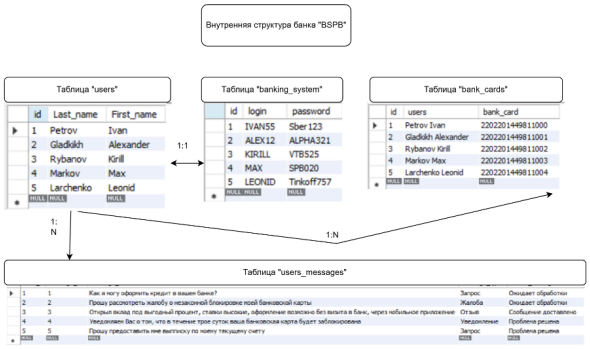
5.3. Переход на страницу подтверждения перевода посредством смс-кода.

- **Требование**: При переходе на страницу “Перевод” отображается информация реквизитов карт пользователя, осуществляющего дальнейший перевод которые необходимо подтвердить вводом в текстовое окно четырехзначного кода поступившего на привязанный номер мобильного телефона вышеуказанного пользователя.
- 6. **Переход на страниу “Перевод”**
- **Требование:** После успешного перевода пользователя автоматически перенаправляется на страницу “Перевод” с уведомлением о выполненном переводе.

База данных

☐ База данных

<details open>
<summary>Внешняя структура банка "BSPB"</summary>



Соединение таблиц
1. Объединяем таблицу "users" с таблицей "banking_system" по полю "id".
2. Объединяю таблицу "bank_cards" с таблицей "users" по полю "id"
3. Объединяем таблицу "users_messages" с таблцей "users_message". по полю "id".

Типы связей между таблицами
1. "users" - "banking_system" 1 - 1. (Каждый пользователь имеет только свой индивидуальный логин и пароль).
2. "users" - "bank_cards" (Каждый пользователь может иметь несколько банковских карт).
3. "users" - "users_messages" (Каждый пользователь имеет возможность отправлять большое количество сообщений).
</details>

<details open>
<summary>Создание базы данных</summary>

Создание базды данных банка “Санкт-Петербург” в MySQL

1. Создание базы данных под названием “banking_system”
   CREATE database banking_system;
2. Осуществляю вход в свою вышеуказанную базу данных.
   USE banking_system;
</details>

<details open>
<summary>Создание таблиц в базе данных</summary>

1. Создание четырех таблиц в моей базе данных “banking_system”. Создание первой таблицы “banking_system” с тремя столбцами “id”, “login”, “password”. Столбцу “id” присвоил первичный ключ.
   Create table banking_system
   (id INT(12) NOT NULL PRIMARY KEY,
   login VARCHAR(55) NULL,
   password VARCHAR(55) NULL);
2. Просмотрим структуру моей первой таблицы
   DESC banking_system;
3. В таблицу “banking_system” добавляю пять новых логинов и паролей.
   INSERT INTO banking_system VALUES (1, “IVAN55” , “Sber123”);
   INSERT INTO banking_system VALUES (2, “ALEX12” , “ALPHA321”);
   INSERT INTO banking_system VALUES (3, “KIRILL” , “VTB525”);
   INSERT INTO banking_system VALUES (4, “MAX” , “SPB020”);
   INSERT INTO banking_system VALUES (5, “LEONID” , “Tinkoff757”);
4. Теперь просмотрим все записи моей первой таблицы “banking_sysyem”
   SELECT * FROM banking_system;
5. Создание второй таблицы “users” с тремя столбцами “id”, “Last_name”, “First_name”. Столбцу “id” присвоил первичный ключ.
   Create table banking_system
   (id INT(12) NOT NULL PRIMARY KEY,
   Last_name VARCHAR(55) NULL,
   First_name VARCHAR(55) NULL);
6. В таблицу “users” добавляю пять новых пользователей - фамилию и имя.
   INSERT INTO users VALUES (1, “Petrov” , “Ivan”);
   INSERT INTO users VALUES (2, “Gladkikh” , “Alexander”);
   INSERT INTO users VALUES (3, “Rybanov” , “Kirill”);
   INSERT INTO users VALUES (4, “Markov” , “Max”);
   INSERT INTO users VALUES (5, “Larchenko” , “Leonid”);
7. Создание третьей таблицы “users_messages” содержащая в себя пять столбцов “id”, “message_id”, “messages_text”, “message_type”, status_message”. Столбцу “user_id” присвоил первичный ключ.
   Create table users_messages
   (user_id INT(12) NOT NULL PRIMARY KEY,
   message_id INT (22) NOT NULL,
   message_text VARCHAR(255) NULL,
   message_type VARCHAR(255) NULL,
   status_message VARCHAR(255);
8. В таблицу “users_messages” добавляю 5 новых записей: id сообщение пользователей, id сообщений, текст сообщений, тип сообщений, статус сообщений.
</details>

INSERT INTO users_messages VALUES (1, 1 ,“Как я могу оформить кредит в вашем банке?” ,”Запрос” ,“Ожидает обработки”);
INSERT INTO users_messages VALUES (2, 2 ,“Прошу рассмотреть жалобу о незаконной блокировки моей банковской карты” ,”Жалоба” ,”Ожидает обработки”);
INSERT INTO users_messages VALUES (3, 3 ,“Открыл вклад под выгодный процент, ставк высокие, оформление возможно вбез визита в банк, через мобильное приложение” ,”Отзыв” ,”Сообщение доставлено”);
INSERT INTO users_messages VALUES (4, 4 ,“Уведомляем Вас о том, что в течение трое суток ваша банковская карта будет заблокирована” ,”Уведомление” ,”Проблема решена”);
INSERT INTO users_messages VALUES (5, 5 ,“Прошу предоставить мне выписку по моему текущему счету” ,”Запрос” ,”Проблема решена”);

9. Создание четвертой таблицы “bank_cards” с тремя столбцами “id”, “users”, “bank_card”. Столбцу “id” присвоил первичный ключ.
   Create table bank_cards
   (id INT(25) NOT NULL PRIMARY KEY,
   users VARCHAR(56) NULL,
   bank_cards VARCHAR(55) NULL);
10. В таблицу “bank_cards” добавляю пять пользователей и пять банковских карт принадлежащих им.
    INSERT INTO bank_cards VALUES (1, “Petrov Ivan” , “2202201449811000”);
    INSERT INTO bank_cardsVALUES (2, “Gladkikh Alexander” , “2202201449811001”);
    INSERT INTO bank_cards VALUES (3, “Rybanov kirill” , “2202201449811002”);
    INSERT INTO bank_cards VALUES (4, “Markov Max” , “2202201449811003”);
    INSERT INTO bank_cards VALUES (5, “Larchenko Leonid” , “2202201449811004”);

<details>
<summary>Добавление внешних ключей в таблицы</summary>

1. Добавляю внешний ключ к таблице “bank_cards”, указывю, что столбец “id” в вышеуказанной таблице будет внешим ключем.
   Alter table bank_cards
   ADD constraint fk_user
   FOREIGH KEY (id)
   REFERENCES users(id);
2. Добавляю внешний ключ к таблице “users_messages”, указывю, что столбец “user_id” в вышеуказанной таблице будет внешим ключем.
   Alter table users_messages
   ADD constraint fk_user
   FOREIGH KEY (user_id)
   REFERENCES users(id);
3. Добавляю внешний ключ к таблице “banking_system”, указывю, что столбец “id” в вышеуказанной таблице будет внешим ключем.
   Alter table banking_system
   ADD constraint fk_user
   FOREIGH KEY (id)
   REFERENCES users(id);
</details>

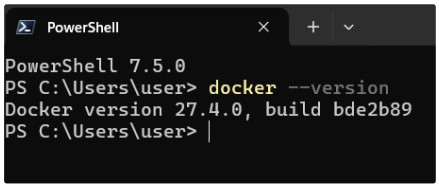
<details>
<summary>Просмотр всех существуюших таблиц</summary>

Просматриваю все свои существующие таблицы в базе данных “banking_system”
show tables;
</details>

<details>
<summary>Соединение таблиц</summary>

Соединяю таблицы, а именно соединяю таблицу “users” с таблицей “banking_system”, “id” из таблицы “users” соединяю с “id” из таблицы “banking_system”. Далее соединяю таблицы “bank_cards” и “users” по полю “id” в таблице “users”. Таблица “users_messages” соединяю с таблицей “users” по полю “user_id”.
select * from banking_system
JOIN users ON users.id = banking_system.id
JOIN bank_cards ON bank_cards.id = users.id
JOIN users_messages on users_messages.user_id = users.id;
</details>

<details>
<summary>Установка Docker</summary>

1. Установка Docker.
   Скачать Docker с сайта Docker Desktop: The #1 Containerization Tool for Developers | Docker
2. Проверяем установку Docker. В терминале прописываем команду docker --version
3. 

</details>

<details>
<summary>Запуск MySQL в контейнере</summary>

Запускаем MySQL в контейнере Для того чтобы запустить

1. docker run --name ######### -e MYSQL_ROOT_PASSWORD=****** -d -p 3306:3306 mysql:latest

   Параметры команды:
</details>

--name ##########: задает имя контейнера (можно выбрать любое имя).

-e MYSQL_ROOT_PASSWORD=**********: задает пароль для пользователя root. Нужно заменить ********** на безопасный пароль.

-d: запускает контейнер в фоновом режиме (detached mode).

-p 3306:3306: проксирует порт 3306 на локальной машине к порту 3306 в контейнере (это стандартный порт MySQL).

mysql:latest: указывает, что мы используем последнюю версию MySQL

2. Проверка запущенного контейнера в docker, в терминале набираем команду docker ps -a

Подключение к серверу MySQL

Запускаем оболчку bash **docker exec -it GladkikhAlexander /bin/bash** , далее подключаемся к серверу MySQL **mysql -u root -p**. Вводим пароль.

```
PS C:\Users\user> docker exec -it GladkikhAlexander /bin/sh
sh-5.1# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 9.2.0 MySQL Community Server - GPL

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Postman Swagger UI

Users Copy.postman_collection.json

Users.postman_environment.json

Запуск коллекции

Устанавливаем Newman на устройство с помощью npm.

Скачиваем с официального сайта Node.js — Run JavaScript Everywhere Node.js

1. Запускаем командную строку
2. Запускаем команду для установки Newman - инструмент командной строки для запуска коллкций postman. npm install -g newman.
3. Проводим команду newman -v для проверки установки newman.

Свою коллецию и окружение из Postman я экспортировал в C:\Postman. Далее провожу команду для запуска коллекции Postman с помощью newman, указывая путь к файлу коллекции и файлу окружения.

- **newman run** - это команда для запуска коллекции с помощью Newman.
- **"C:\Postman\Users Copy.postman_collection.json"** - это путь к моему файлу коллекции Postman, который я запущу.
- **-e "C:\Postman\Users.postman_environment.json"** - эта команда указывает файл окружения, который я использую при выполнении коллекции.

```
PS C:\Users\user> newman run "C:\Postman\Users Copy.postman_collection.json"
 -e "C:\Postman\Users.postman_environment.json"
newman

Users Copy

→ Запросить пользователя
  GET https://petstore.swagger.io/v2/user/Gladkikh1996 [200 OK, 513B, 902ms]
  ✓  Status code is 200
  1. Response time is less than 200ms
  ✓  Content-Type is present
  ✓  Content-Type is application/json

→ Обновить пользователя
  PUT https://petstore.swagger.io/v2/user/Gladkikh1996 [200 OK, 387B, 125ms]
  ✓  Status code is 200
  ✓  Response time is less than 200ms
  ✓  Content-Type is present
  ✓  Content-Type is application/json

→ Удаление пользователя
  DELETE https://petstore.swagger.io/v2/user/Gladkikh1996 [200 OK, 380B, 125ms]
  ✓  Status code is 200
  ✓  Response time is less than 200ms
  ✓  Content-Type is present
  ✓  Content-Type is application/json

→ Запрос на наличие пользователя в системе
  GET https://petstore.swagger.io/v2/user/login [200 OK, 471B, 125ms]
  ✓  Status code is 200
  ✓  Response time is less than 200ms
  ✓  Content-Type is present
  ✓  Content-Type is application/json

→ Выход из текущего сеанса
  GET https://petstore.swagger.io/v2/user/logout [200 OK, 370B, 124ms]
  ✓  Status code is 200
  ✓  Response time is less than 200ms
  ✓  Content-Type is present
  ✓  Content-Type is application/json

→ Создать пользователя
  POST https://petstore.swagger.io/v2/user/ [200 OK, 387B, 127ms]
  ✓  Status code is 200
  ✓  Response time is less than 200ms
  ✓  Content-Type is present
  ✓  Content-Type is application/json
```

	executed	failed
iterations	1	0
requests	6	0
test-scripts	6	0
prerequest-scripts	0	0
assertions	24	1

total run duration: 2s

total data received: 485B (approx)

average response time: 254ms [min: 124ms, max: 902ms, s.d.: 289ms]

#	failure	detail
1.	AssertionEr…	Response time is less than 200ms expected 902 to be below 600 at assertion:1 in test-script inside "Запросить пользователя"

Bash Script

Bash script

echo "Введите первое целое число для сложения"
read a_1

echo "Введите второе целое число для сложения"
read a_2

echo "Результат = $(($a_1 + $a_2))"

- **echo**: команда, которая выводит текст на экран. В данном случае она запрашивает у пользователя ввод первого целого числа.
- **read a_1**: команда read считывает ввод пользователя и сохраняет его в переменной **a_1.**
- **read a_2**: команда read считывает ввод пользователя и сохраняет его в переменной **a_2.**
- **$(($a_1 + $a_2))**: это арифметическая операция, которая складывает значения переменных **a_1** и **a_2.**
- **echo**: выводит результат сложения на экран в формате **"Результат = $(($a_1 + $a_2))"**

```
user@LAPTOP-VD1F23JA MINGW64 ~/portfolio/bash_script (main)
$ sh script.sh
Введите первое целое число для сложения
20
Введите второе целое число для сложения
25
Результат = 45
```

Bash команды

Баш команды

bash.txt
11 Feb 2025, 11:11 PM

Клиент-серверная архитектура

Схема клиент-серверной архитектруы

client-schema.pdf
11 Feb 2025, 11:07 PM

Описание вышеуказанной клиент-серверной архитектуры

На схеме представлена система взаимодействия между клиентами и сервером, клиенты представляют собой устройства, которые отправляют запросы к серверу.

Устройства

1. **Клиенты:**

- **ПК** (персональный компьютер)
- **Mobile** (мобильное устройство)
- **Laptop** (ноутбук)

2. **Серверы:**

- **Server** (сервер) - основной сервер, к которому подключаются клиенты.
- **Load balancer** (балансировщик нагрузки) - распределяет запросы между серверами.
- **Database** (база данных) - хранит данные, к которым обращаются серверы.

3. **Сеть:**

- **Интернет LAN/WAN:** Сеть, объединяющая клиентов, серверы и базу данных.

4. **Действия**

- Клиенты отправляют запросы к серверу.
- Запросы клиентов поступают на балансировщик нагрузки.
- Балансировщик нагрузки распределяет запросы между серверами.
- Сервер обрабатывает запросы клиентов и отправляет ответы.
- Сервер обращается к базе данных, чтобы получить информацию, необходимую для обработки запросов.

5. **Вывод**

Изображение иллюстрирует типичный сценарий взаимодействия между клиентами и сервером в сети. Балансировщик нагрузки, серверы и база данных работают вместе, чтобы обеспечить эффективную обработку запросов клиентов.