

# Final project (BSPB)

|                  |  |
|------------------|--|
| Ссылка на задачу | <input checked="" type="checkbox"/> JUS-234: Финальный проект <span style="border: 1px solid #ccc; padding: 2px;">TO DO</span> |
| Статус страницы  | In progress  |
| Описание         | Проектная документация для тестирования веб-приложения <a href="#">Bank Saint-Petersburg</a>                                   |

## i Описание

- Описание требований...

### Экран авторизации: ввод логина и пароля

#### 1. Функциональные требования:

- Пользователь должен иметь возможность ввести логин и пароль для авторизации в системе.
- При вводе некорректных данных (неверный логин или пароль) система должна отображать сообщение об ошибке: "Неверный логин или пароль".
- Кнопка "Войти" должна быть активна только при заполнении обоих полей (логин и пароль).
- Должна быть предусмотрена возможность восстановления пароля через ссылку "Забыли пароль?".

#### 2. Нефункциональные требования:

- Пароль должен быть скрыт символами (например,  ) при вводе.
- Система должна блокировать вход после 3 неудачных попыток авторизации на 15 минут.
- Данные должны передаваться по защищенному протоколу (HTTPS).

### 2. Экран авторизации: ввод 2FA из SMS

#### 1. Функциональные требования:

- После успешного ввода логина и пароля пользователь должен получить SMS с кодом подтверждения.
- Пользователь должен ввести код из SMS в поле ввода.
- При вводе некорректного кода система должна отображать сообщение об ошибке: "Неверный код подтверждения".
- Должна быть предусмотрена возможность повторной отправки SMS с кодом через кнопку "Отправить код повторно".

#### 2. Нефункциональные требования:

- Код из SMS должен быть действителен в течение 5 минут.
- После 3 неудачных попыток ввода кода система должна заблокировать возможность ввода на 10 минут.

### 3. Экран "Персональные предложения", блок "Нам важно знать ваше мнение"

#### 1. Функциональные требования:

- На экране "Персональные предложения" должен отображаться блок с заголовком "Нам важно знать ваше мнение".

b. В блоке должна быть кнопка “Оставить отзыв”, которая перенаправляет пользователя на форму обратной связи.

c. Форма обратной связи должна включать:

i. Поле для ввода текста отзыва (максимум 500 символов).

ii. Оценку от 1 до 5 звезд.

iii. Кнопку “Отправить”.

d. После отправки отзыва система должна отображать сообщение: “Спасибо за ваш отзыв!”.

## 2. Нефункциональные требования:

a. Отзывы должны сохраняться в базе данных для дальнейшего анализа.

b. Форма должна быть доступна только авторизованным пользователям.

## 4. Экран “Карты”, раздел “Бизнес-карты”, кнопка “Открыть новый счет” ☰

### 1. Функциональные требования:

a. В разделе “Бизнес-карты” должна быть кнопка “Открыть новый счет”.

b. При нажатии на кнопку пользователь должен быть перенаправлен на форму открытия счета.

c. Форма должна включать:

i. Выбор типа счета (например, текущий, накопительный).

ii. Валюту счета (RUB, USD, EUR).

iii. Подтверждение согласия с условиями обслуживания (чекбокс).

iv. Кнопку “Открыть счет”.

d. После успешного открытия счета система должна отображать сообщение: “Счет успешно открыт”.

### 2. Нефункциональные требования:

a. Форма должна быть доступна только для авторизованных пользователей с подтвержденным номером телефона.

b. Данные о новом счете должны быть сохранены в системе и отображаться в личном кабинете пользователя.

## Дополнительные требования: ☰

### 1. Удобство использования (UX):

a. Все экраны должны быть адаптированы для мобильных устройств и ПК.

b. Кнопки и поля ввода должны быть достаточно крупными для удобного использования на сенсорных экранах.

c. Должна быть предусмотрена возможность возврата на предыдущий экран через кнопку “Назад”.

### 2. Безопасность:

a. Все данные пользователя должны быть защищены в соответствии с нормативными требованиями.

b. Логи и попытки входа должны сохраняться для аудита.

### 3. Производительность:

a. Время загрузки экранов не должно превышать 3 секунды.

b. SMS с кодом должно отправляться не позднее 10 секунд после запроса.

## ✓ Тест-кейсы

⌚ JUS-911: Открытие рублевого вклада на 2 недели на сумму 50 000.01 руб. с ежемесячной выплатой процентов по ставке 12.05%. **ГОТОВО**

⌚ JUS-912: Распечатка выписки по счету "Зарплатный" за текущий месяц по общему расширенному фильтру. [Готово](#)

⌚ JUS-912: Распечатка выписки по счету "Зарплатный" за текущий месяц по общему расширенному фильтру. [Готово](#)

## ✖ Баг-репорты

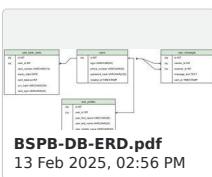
⌚ JUS-889: При переходе в раздел Бизнес-карты отображается сообщение-alert "Access Denied" при использовании ИП "Морозов Павел Изяславович". [Готово](#)

⌚ JUS-890: В разделе "Нам важно знать ваше мнение" при использовании русской локализации сайта отображается опрос на английском языке, не связанный с банковской тематикой. [Готово](#)

⌚ JUS-891: В разделе часто задаваемых вопросов (FAQ) дублируется информация для физических и юридических лиц. [Готово](#)

## ℹ База данных

 BSPB-DB-ERD.drawio



- Справка по запуску MySQL в докер-контейнере...

...

### 1. Установить Docker следуя [официальной документации](#).

```
fuwa@fuwa-Nest:~/repos/qa-test-mysql-db-dkr-loc$ sudo apt update && sudo apt install docker.io -y
[sudo] password for fuwa:
Hit:1 http://ru.archive.ubuntu.com/ubuntu oracular InRelease
Get:2 http://ru.archive.ubuntu.com/ubuntu oracular-updates InRelease [126 kB]
Hit:3 https://dl.google.com/linux/chrome/deb stable InRelease
Get:4 http://ru.archive.ubuntu.com/ubuntu oracular-backports InRelease [126 kB]
Hit:5 http://security.ubuntu.com/ubuntu oracular-security InRelease
Hit:6 https://packages.microsoft.com/code stable InRelease
Hit:7 https://ppa.launchpadcontent.net/git-core/ppa/ubuntu oracular InRelease
Hit:8 https://www.charlesproxy.com/packages/apt charles-proxy InRelease
Fetched 252 kB in 1s (185 kB/s)
114 packages can be upgraded. Run 'apt list --upgradable' to see them.
Notice: Missing Signed-By in the sources.list(5) entry for 'http://ru.archive.ubuntu.com/ubuntu'
Notice: Missing Signed-By in the sources.list(5) entry for 'http://ru.archive.ubuntu.com/ubuntu'
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.

Installing:
  docker.io

Installing dependencies:
  bridge-utils containerd pigz runc ubuntu-fan

Suggested packages:
  ifupdown  cgroupfs-mount docker-buildx  rinse
  aufs-tools | cgroup-lite docker-compose-v2 zfs-fuse
  btrfs-progs debootstrap  docker-doc  | zfsutils

Summary:
  Upgrading: 0, Installing: 6, Removing: 0, Not Upgrading: 114
  Download size: 71.2 MB
  Space needed: 271 MB / 201 GB available

Get:1 http://ru.archive.ubuntu.com/ubuntu oracular/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://ru.archive.ubuntu.com/ubuntu oracular/main amd64 bridge-utils amd64 1.7.1-2ubuntu1 [34.1 kB]
Get:3 http://ru.archive.ubuntu.com/ubuntu oracular/main amd64 runc amd64 1.1.12-0ubuntu4 [8,582 kB]
Get:4 http://ru.archive.ubuntu.com/ubuntu oracular-updates/main amd64 containerd amd64 2.0.0-rc3-0ubuntu1 [30.2 kB]
Get:5 http://ru.archive.ubuntu.com/ubuntu oracular-updates/universe amd64 docker.io amd64 26.1.3-0ubuntu1.1 [32.2 MB]
Get:6 http://ru.archive.ubuntu.com/ubuntu oracular/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]
Fetched 71.2 MB in 9s (7,511 kB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 27326 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package bridge-utils.
Preparing to unpack .../1-bridge-utils_1.7.1-2ubuntu1_amd64.deb ...
Unpacking bridge-utils (1.7.1-2ubuntu1) ...
Selecting previously unselected package runc.
Preparing to unpack .../2-runc_1.1.12-0ubuntu4_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu4) ...
Selecting previously unselected package containerd.
```

```
fuwa@fuwa-Nest:~/repos/qa-test-mysql-db-dkr-loc$ sudo docker --version
Docker version 26.1.3, build 26.1.3-0ubuntu1.1
```

```
fuwa@fuwa-Nest:~/repos/qa-test-mysql-db-dkr-loc$ sudo docker version
[sudo] password for fuwa:
Client:
  Version:          26.1.3
  API version:     1.45
  Go version:      go1.22.8
  Git commit:      26.1.3-0ubuntu1.1
  Built:           Thu Dec 12 08:27:10 2024
  OS/Arch:         linux/amd64
  Context:         default

Server:
  Engine:
    Version:          26.1.3
    API version:     1.45 (minimum version 1.24)
    Go version:      go1.22.8
    Git commit:      26.1.3-0ubuntu1.1
    Built:           Thu Dec 12 08:27:10 2024
    OS/Arch:         linux/amd64
    Experimental:   false
  containerd:
    Version:          2.0.0~rc3
    GitCommit:
  runc:
    Version:          1.1.12-0ubuntu4
    GitCommit:
  docker-init:
    Version:          0.19.0
    GitCommit:
```

2. Для установки и запуска MySQL в контейнере выполнить команду:

```
1 docker run --name mysql-container -e MYSQL_ROOT_PASSWORD=your_password -p 3306:3306 mysql:latest
```

где:

`--name mysql-container` – ИМЯ контейнера

`-e MYSQL_ROOT_PASSWORD=your_password` – пароль для root

`-p 3306:3306` – проброс стандартного MySQL порта для подключения.

```
fuwa@fuwa-Nest:~$ sudo docker run --name mysql-container -e MYSQL_ROOT_PASSWORD=qa_test_pw -p 3306:3306 mysql:latest
[sudo] password for fuwa:
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
2c0a233485c3: Pull complete
21577e00f2ba: Pull complete
c294da35c13e: Pull complete
faccdf3107c1: Pull complete
de4342aa4ad8: Pull complete
4643f1cf56c2: Pull complete
139aca660b47: Pull complete
b10e1082570e: Pull complete
26313a3e0799: Extracting [=====] 36.21MB/135.7MB
d43055c38217: Download complete
S
```

3. Для проверки работы контейнера выполнить команду:

```
1 docker ps
```

```
fuwa@fuwa-Nest:~$ sudo docker ps
[sudo] password for fuwa:
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                 NAMES
fb6b465e0ac        mysql:latest        "docker-entrypoint.s..."   About a minute ago   Up About a minute   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   mysql-container
fuwa@fuwa-Nest:~$
```

4. Для подключения к MySQL внутри контейнера через CLI выполнить команду:

```
1 docker exec -it mysql-container mysql -u root -p
2 # (Введите пароль, указанный в MYSQL_ROOT_PASSWORD)
3
```

```
fuwa@fuwa-Nest:~$ sudo docker exec -it mysql-container mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
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> 
```

5. Для подключения к MySQL внутри контейнера через GUI использовать данные:

- 1 Хост: localhost
- 2 Порт: 3306
- 3 Пользователь: root
- 4 Пароль: your\_password

6. После подключения использовать стандартные SQL запросы, в случае необходимости воспользоваться справкой через команду `\h`.

```
mysql> status
-----
mysql Ver 9.2.0 for Linux on x86_64 (MySQL Community Server - GPL)

Connection id:          10
Current database:        -
Current user:           root@localhost
SSL:                   Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:         ;
Server version:         9.2.0 MySQL Community Server - GPL
Protocol version:       10
Connection:             Localhost via UNIX socket
Server characterset:    utf8mb4
Db   characterset:      utf8mb4
Client characterset:    latin1
Conn. characterset:     latin1
UNIX socket:            /var/run/mysqld/mysqld.sock
Binary data as:          Hexadecimal
Uptime:                 5 min 24 sec

Threads: 2 Questions: 6 Slow queries: 0 Opens: 119 Flush tables: 3 Open tables: 38 Queries per second avg: 0.018
-----
mysql> 
```

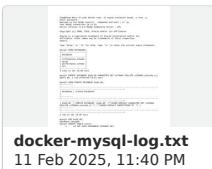
7. Остановка и запуск контейнера:

- 1 `docker stop mysql-container` # Остановить
- 2 `docker start mysql-container` # Запустить
- 3 `docker restart mysql-container` # Перезапустить

8. Удаление контейнера:

- 1 `docker stop mysql-container` # Остановить
- 2 `docker rm mysql-container` # Удалить контейнер
- 3 `docker rmi mysql:latest` # Удалить образ

✓ Запуск MySQL в докер-контейнере. Создание БД. Создание таблиц. Добавление данных. Лог файл.



▼ Описание...

Коллекция restful-booker предназначена для работы с API бронирования [restful-booker](#). Коллекция автоматизирует процесс работы с API бронирования, начиная с аутентификации и заканчивая удалением бронирования. Она использует переменные для передачи данных между запросами и динамически формирует URL.

1. [POST] - Auth - CreateToken:

- Запрос получает токен авторизации.
- В ответе извлекается токен и сохраняется в переменную коллекции auth\_token. Токен используется для авторизации в последующих запросах.

2. [POST] - Booking - CreateBooking

- Запрос создает новое бронирование.
- В ответе извлекаются данные бронирования (firstname, lastname, bookingid) и сохраняются в переменные коллекции: var\_firstname, var\_lastname, var\_bookingid.

3. [PUT] - Booking - UpdateBooking

- Запрос изменяет данные ранее созданного бронирования.
- В скрипте перед запросом извлекается bookingid из переменной коллекции. Формируется URL с подставленным bookingid. Добавляется заголовок Cookie с токеном аутентификации (token= {{auth\_token}}).

4. [GET] - Booking - GetBooking

- Запрос проверяет, что данные бронирования изменены.
- В скрипте перед запросом извлекается bookingid из переменной коллекции. Формируется URL с подставленным bookingid.

5. [DELETE] - Booking - DeleteBooking

- Запрос осуществляет удаление ранее созданного и измененного бронирования.
- В скрипте перед запросом извлекается bookingid из переменной коллекции. Формируется URL с подставленным bookingid. Добавляется заголовок Cookie с токеном аутентификации (token= {{auth\_token}}).

6. [GET] - Booking - CheckGetBookingAfterDeleteAction

- Запрос проверяет, что бронирование удалено.
- В скрипте перед запросом извлекается bookingid из переменной коллекции. Формируется URL с подставленным bookingid.

▼ JSON...



▼ Newman collection execution report...

Запуск коллекции через Newman:

```
1 newman run
2 path/to/your/restful-booker.postman_collection.json -e
3 path/to/your/restful-booker.postman_environment.json
```

Результат запуска:



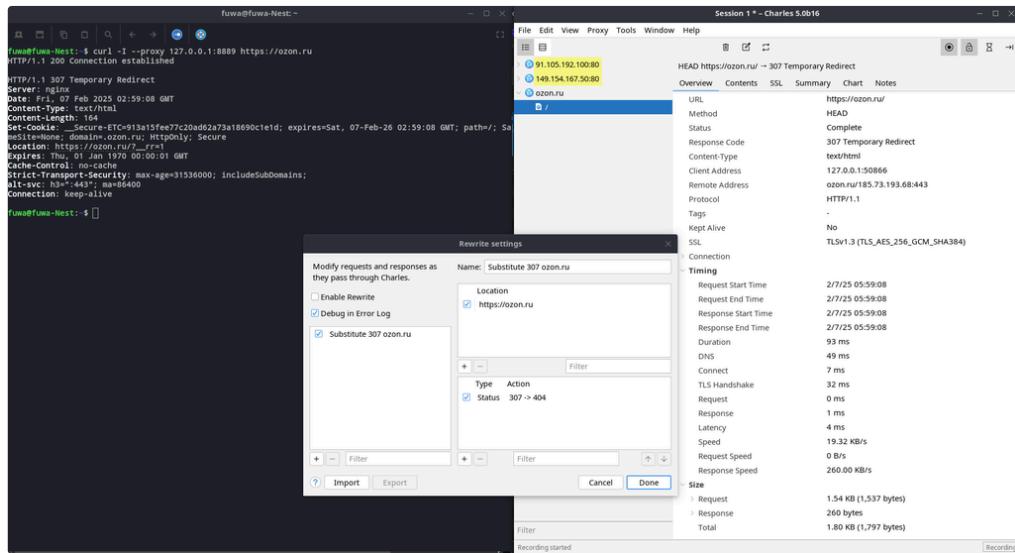


## Charles Proxy

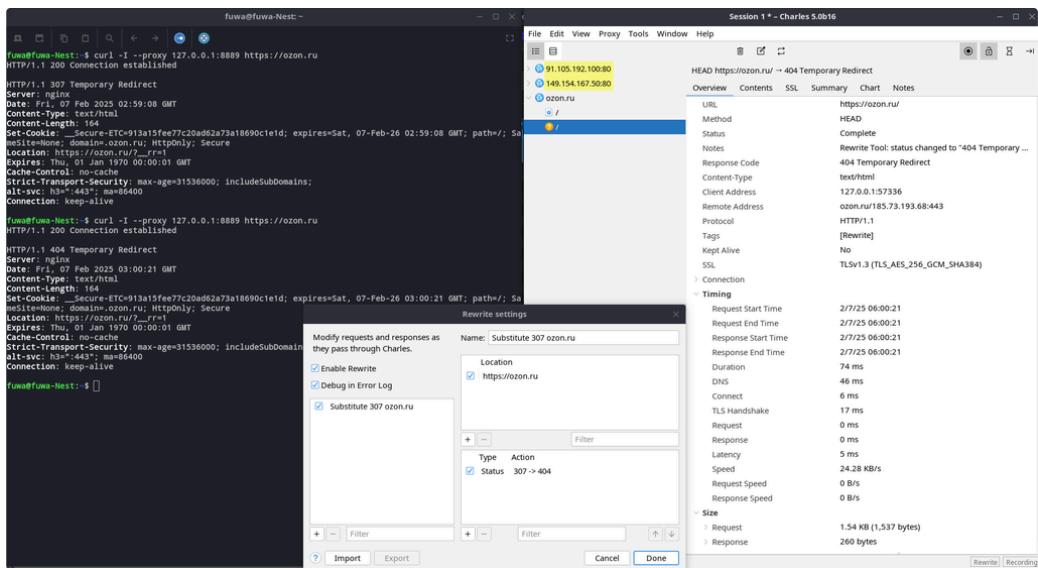
- ✓ Rewrite tool...

### Использование Rewrite tool:

1. В меню сверху нажать **Tools > Rewrite**.
2. Поставить галочку **Enable Rewrite** в появившемся окне.
3. Нажмите кнопку **Add**.
4. Ввести название правила.
5. Перейдите во вкладку **Locations**.
6. Нажать **Add** и указать, запросы для работ
7. Перейдите во вкладку **Rules**.
8. Нажмите **Add** и настроить правило:
  - **Type:** выбрать **Response Status**.
  - В поле **Match** ввести статус код, который нужно изменить.
  - В поле **Replace** написать статус код для замены
9. Нажать **Done**.
10. Отправить запрос снова.



Rewrite выключен. Для теста выполняем запрос через curl в командной строке.



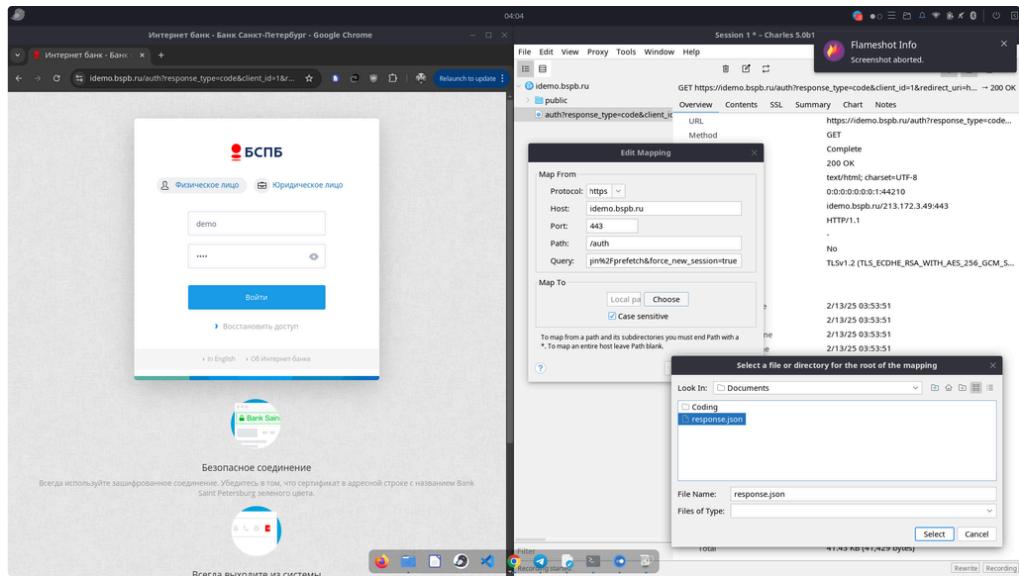
Rewrite включен. Повторяем запрос, чтобы убедиться, что подмена кода успешно выполнена.

▼ Map local tool...

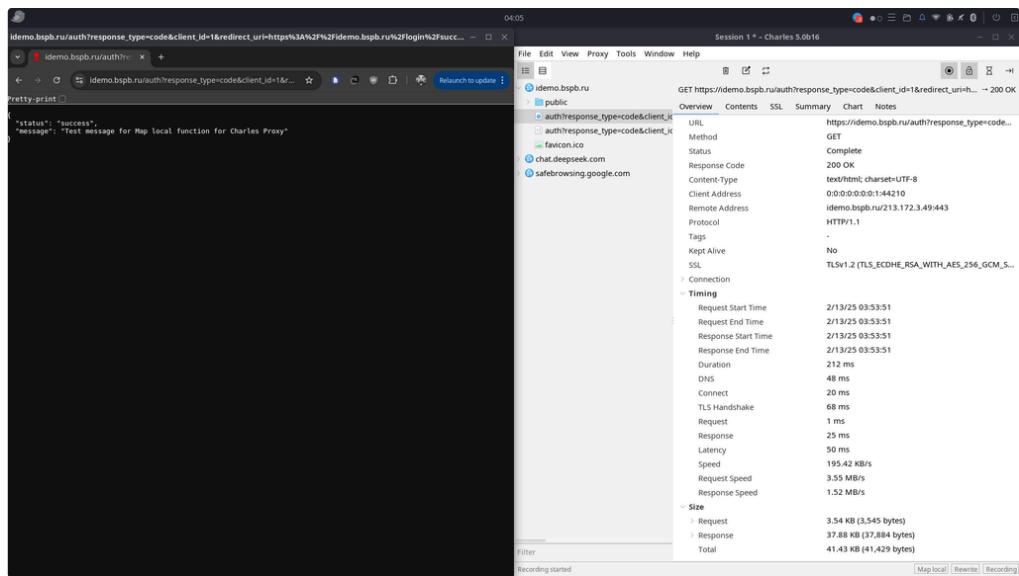
### Использование Map local tool:

- Отправить запрос в приложении/браузере, чтобы он отобразился в Charles. В списке запросов найдите нужный.
- Кликнуть правой кнопкой на запрос и выберите `Map Local`.
- Нажать на кнопку `Choose...` и выбрать файл для замены тела ответа (JSON, HTML, XML и др.).
- Проверить условия подмены:
  - В Charles подмена срабатывает, если путь запроса, метод (GET/POST), заголовки и другие параметры совпадают. Если запрос содержит динамические параметры (например, query string или заголовки), убедиться, что они остаются одинаковыми при каждом вызове.
- Сохранить настройки и убедиться, что подмена активна (галочка напротив `Map Local` включена).
- Отправить запрос снова и проверить вкладку `Response`. Charles заменит тело ответа содержимым локального файла.

Выбираем запрос для подмены.



Настраиваем Mapping и выбираем файл для подмены.

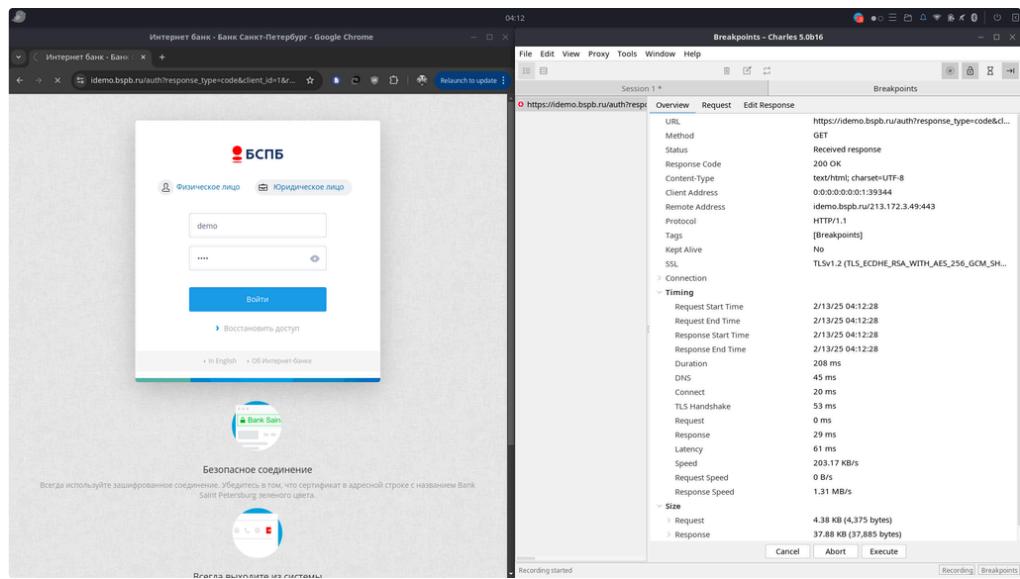


Обновляем запрос и смотрим результат.

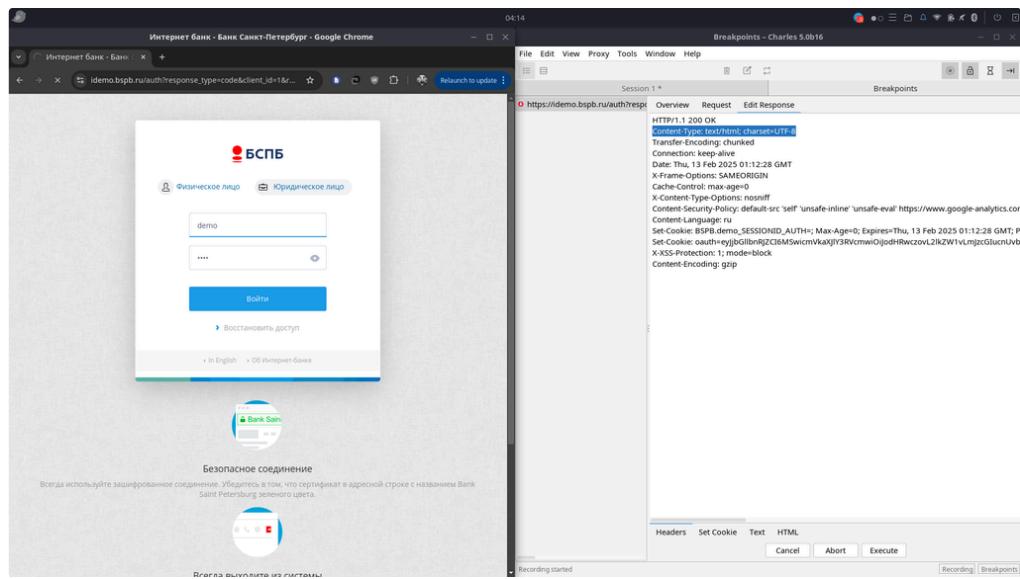
#### Breakpoint tool...

#### Использование breakpoint tool:

- Выбрать нужный запрос в списке, кликнуть правой кнопкой и выбрать **Breakpoint**.
- Отправить запрос снова. В Charles появится окно Breakpoint.
- Изменение заголовков:**
  - Если Charles остановил запрос (вкладка **Request**):
    - Перейти в **Headers**.
    - Найти нужный заголовок, чтобы изменить его, или добавить новый, нажав на кнопку **Add**.
  - Если Charles остановил ответ (вкладка **Response**):
    - В **Headers** также можно изменить заголовки ответа от сервера.
- После редактирования нажать **Execute**, чтобы запрос или ответ продолжили обработку с изменениями.



После настройки делаем запрос. В Charles открывается окно Breakpoints.



Выбираем нужный заголовок для подмены.

The screenshot shows a browser window with a login form for 'Банк Санкт-Петербург'. The form includes fields for 'Логин' and 'Пароль', a 'Войти' button, and a link to 'Восстановить доступ'. Below the form is a 'Безопасное соединение' message. The Charles proxy tool window is open, showing a captured HTTP request for the login page. The request details pane shows the following headers:

```

HTTP/1.1 200 OK
Content-Type: application/json
Transfer-Encoding: chunked
Connection: keep-alive
Date: Thu, 13 Feb 2025 01:12:28 GMT
X-Frame-Options: SAMEORIGIN
Cache-Control: max-age=0
X-Content-Type-Options: nosniff
Content-Security-Policy: default-src 'self' 'unsafe-inline' 'unsafe-eval' https://www.google-analytics.com
Content-Language: ru
Set-Cookie: BSFB_demo_SESSONID_AUTH=; Max-Age=0; Expires=Thu, 13 Feb 2025 01:12:28 GMT; Path=/
Set-Cookie: oauth=eyJhbGciOiJIUzI1NiJ9.eyJvcmluZzI6Imh0dHlzcWxvZmVyc29yL2lkZWVtLmJzGlsuJvbX...; X-XSS-Protection: 1; mode=block
Content-Encoding: gzip

```

Изменяем данные заголовка и нажимаем Execute.

The screenshot shows a browser with the same login page. The Charles proxy tool window shows the modified request. The 'Content-Type' header has been changed to 'application/json'. The response from the server is shown in the 'Contents' tab of the Charles interface.

Видим результат подмены заголовков.

## Bash команды

### Список команд...

- 1 а. Перейти в домашнюю директорию:  
cd ~
- 3
- 4
- 5 б. Создать новую папку:  
mkdir my\_new\_folder
- 7
- 8
- 9 с. Перейти в новую папку:  
cd my\_new\_folder
- 11
- 12
- 13 д. Создать в новой папке новый файл:

```
14 touch my_file.txt
15
16
17 е. Перейти в текстовый редактор (например, nano) и написать текст:
18 nano my_file.txt
19 -> ввод текста
20
21
22 f. Выйти и сохранить этот текст:
23 В редакторе nano:
24 Нажмите Ctrl + O для сохранения файла.
25 Нажмите Enter для подтверждения.
26 Нажмите Ctrl + X для выхода из редактора.
27
28
29 g. Посмотреть конкретное словосочетание в файле:
30 grep "ваше_словосочетание" my_file.txt
31 -> замените "ваше_словосочетание" на нужное слово или фразу.
32
33
34 h. Посмотреть конкретное словосочетание в файле и перенаправить поток в новый файл:
35 grep "ваше_словосочетание" my_file.txt > new_file.txt
36
37 i. Посмотреть, что находится в вашей директории:
38 ls
39
40
41 j. Как посмотреть, что находится в вашей директории, Но, включая скрытые файлы:
42 ls -a
43
44
45 k. Посмотреть, где вы находитесь (в плане пути):
46 pwd
47
48
49 l. Вы в домашней директории ->
50 создать 2 папки ->
51 перейти в первую любую папку ->
52 создать там файл ->
53 скопировать и перенести этот файл во вторую папку:
54
55 cd ~
56 mkdir folder1 folder2
57 cd folder1
58 touch file_in_folder1.txt
59 cp file_in_folder1.txt ../folder2/
60 Теперь файл file_in_folder1.txt будет скопирован в папку folder2.
```

## Bash скрипты

▼ Список скриптов...

### [Скрипт мониторинга доступности URL с преобразованием в curl](#) ↗

▼ Описание...

Назначение: мониторинг доступности веб-сервисов или API, включая учет редиректов.

1. Пользователь вводит URL для проверки.
2. Пользователь вводит количество попыток (MAX\_ATTEMPTS).
3. Цикл с попытками - Для каждой попытки отправляется запрос с помощью curl.
4. Тайм-аут 5 секунд на каждый запрос.
  - Если сервис доступен (код ответа 2XX), выводится сообщение об успешном подключении, и выполнение скрипта завершается.
  - Если сервис выполняет редирект (код ответа 3XX), скрипт сообщает об этом и продолжает следовать за редиректом.
  - Если код ответа другой (например, ошибка 4XX или 5XX), скрипт повторяет запрос, пока не исчерпает количество попыток.
5. Ожидание ввода:
  - После завершения работы скрипта (успех или неудача) пользователь должен нажать Enter, чтобы закрыть окно.

▼ CODE - service-availability-checker.sh



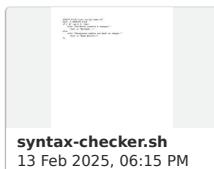
## Проверка синтаксиса Bash-скриптов

▼ Описание...

Назначение: предварительная проверка синтаксиса другого bash-скрипта на отсутствие ошибок перед его выполнением.

1. Переменная `SCRIPT_FILE` указывает на файл скрипта, который будет проверяться.
2. Команда `bash -n $SCRIPT_FILE` запускает проверку синтаксиса указанного скрипта, не выполняя его.
3. Если синтаксис правильный (код возврата `$?` равен 0), выводится сообщение "Синтаксис скрипта в порядке."
4. Если есть ошибки (код возврата не равен 0), выводится сообщение "Обнаружены ошибки в синтаксисе."

▼ CODE - syntax-checker.sh



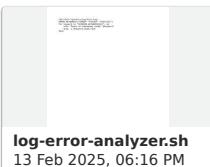
## Проверка логов на наличие ошибок

▼ Описание...

Назначение: поиск ключевых слов в лог-файле. Скрипт проверяет наличие слов, которые могут указывать на ошибки или критические события, и выводит соответствующие строки из лог-файла.

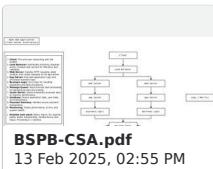
- Переменная `LOG_FILE="/var/log/app/error.log"` — путь к лог-файлу, который будет анализироваться.
- Переменная `ERROR_KEYWORDS=( "ERROR" "FAILED" "CRITICAL" )` — массив ключевых слов, которые будут искааться в лог-файле.
- Цикл `for` перебирает каждое ключевое слово из массива `ERROR_KEYWORDS`. Для каждого ключевого слова выполняется поиск в лог-файле.
- Команда `grep -i $keyword $LOG_FILE` используется для поиска строк в файле, которые содержат указанное ключевое слово, без учета регистра.
- Перед поиском каждого ключевого слова выводится сообщение: Поиск по ключевому слову: `<ключевое_слово>`. Затем выводятся строки из лог-файла, содержащие это ключевое слово.

▼ Code - log-error-analyzer.sh



## ⓘ Клиент-серверная архитектура

BSPB-CSA.drawio

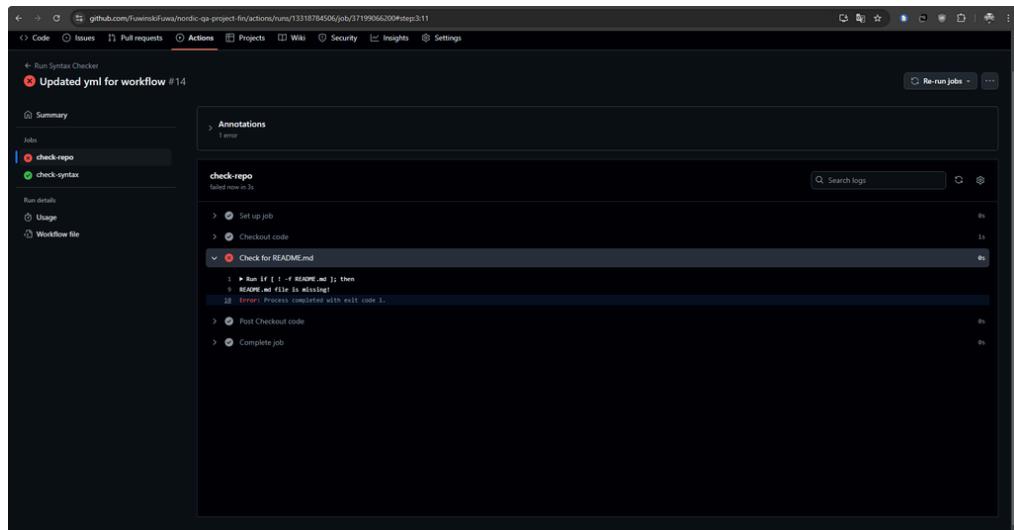


## ⓘ Github Actions

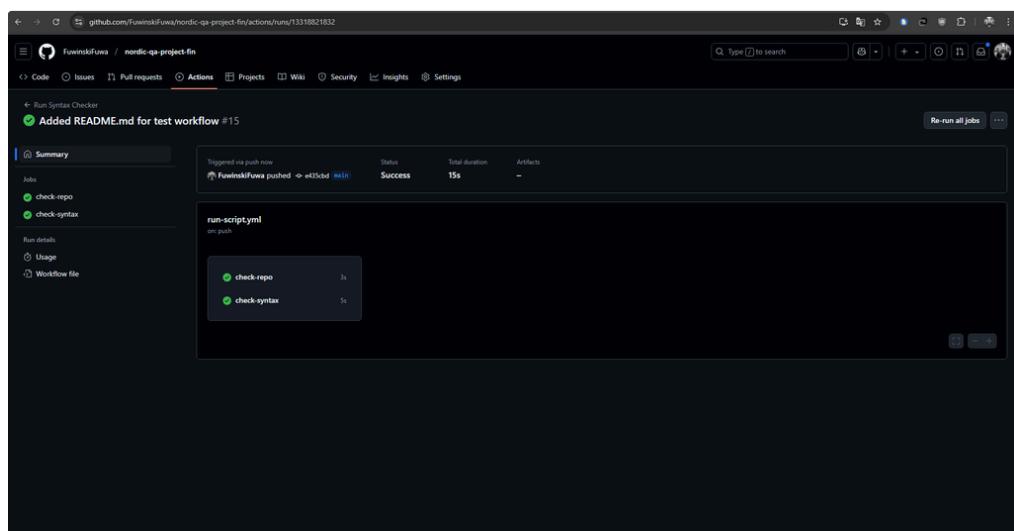
▼ Jobs...

Описание задач:

- Проверка наличия README файла в репозитории.
- Запуск shell скрипта, который проверяет синтаксис другого скрипта.



После пуша в ветку и настройки yml файла workflow первая задача выдаёт ожидаемую ошибку. Вторая задача выполняется успешно.



После добавления файла README при повторном запуске цикла обе задачи выполнены успешно.

▼ Содержание .yml файла workflow...

```
1 name: test-workflow
2
3 on:
4   push:
5     branches:
6       - main
7
8 jobs:
9   check-repo:
10    runs-on: ubuntu-latest
11
12   steps:
13     - name: Checkout code
14       uses: actions/checkout@v3
15
16     - name: Check for README.md
```

```
17     run: |
18         if [ ! -f README.md ]; then
19             echo "README.md file is missing!"
20             exit 1
21         else
22             echo "README.md file exists!"
23         fi
24
25 check-syntax:
26   runs-on: ubuntu-latest
27
28 steps:
29   - name: Checkout code
30     uses: actions/checkout@v2
31
32   - name: Make scripts executable
33     run: chmod +x bash-script/*.sh
34
35   - name: Run Syntax Checker
36     run: bash bash-script/syntax-checker-ga.sh
```