

**Проект приложения «Домашняя бухгалтерия»**

**Разработал Очкасов Александр Павлович**

Концепция проекта заключается в следующем. С помощью данного приложения пользователь может вести учет своих расходов/доходов, разбивая их по различным категориям (предложенным по умолчанию приложением, а так же созданным самостоятельно) и соотнося их с определенными счетами. Счета могут быть созданы на основе разделения по статьям расходов/доходов (Например отдельный счет в котором пользователь учитывает отложенные финансы на отпуск, накопления на квартиру, машину, счет на базовые потребности и т.д.), а так же и просто как разделение по непосредственному нахождению финансов (Счет в банке «А»; деньги на карте в банке «Б»; сотка на проезд, забытая в кармане зимней куртки. Либо для семейного бюджета счета могут отражать количество денег у каждого члена семьи). В этом плане пользователю отдается свобода для его личных предпочтений и удобств.

Проект написан на языке Java с использованием таких технологий, как: SpringBoot, SpringSecurity, SpringMVC, Hibernate, SpringDataJPA, Thymeleaf, Junit, Maven, Jackson, PostgreSQL.

Ознакомиться со схемой используемой БД можно по ссылке <https://drawsql.app/teams/redw4ys-team/diagrams/home-accounting>

Проект реализован в двух версиях:

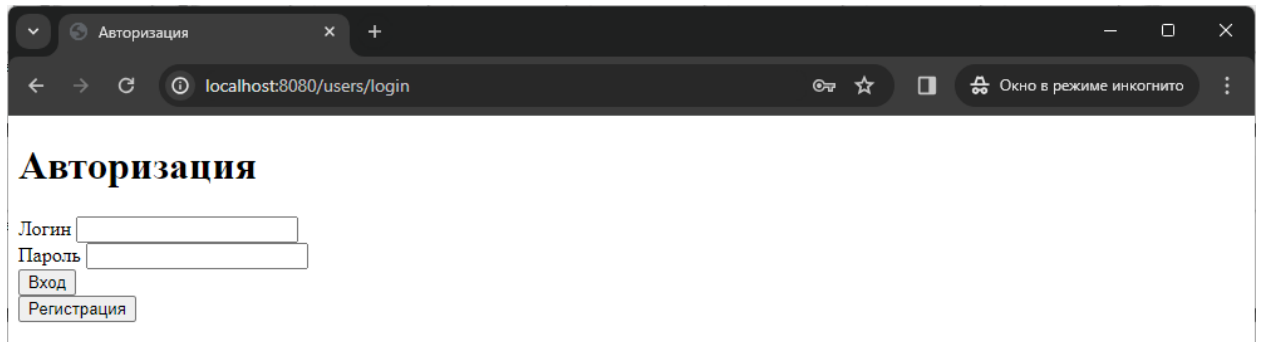
1)Web приложение с отображениями на HTML страницах с использованием шаблонизатора Thymeleaf. Страница проекта на GitHub: [https://github.com/OchkasovAP/Home\\_accounting](https://github.com/OchkasovAP/Home_accounting)

2)REST микросервис. Страница проекта на GitHub: [https://github.com/OchkasovAP/Home\\_accounting\\_REST](https://github.com/OchkasovAP/Home_accounting_REST)

## Web приложение

Рассмотрим основной функционал приложения

Не авторизованный пользователь имеет доступ только к страницам авторизации и регистрации

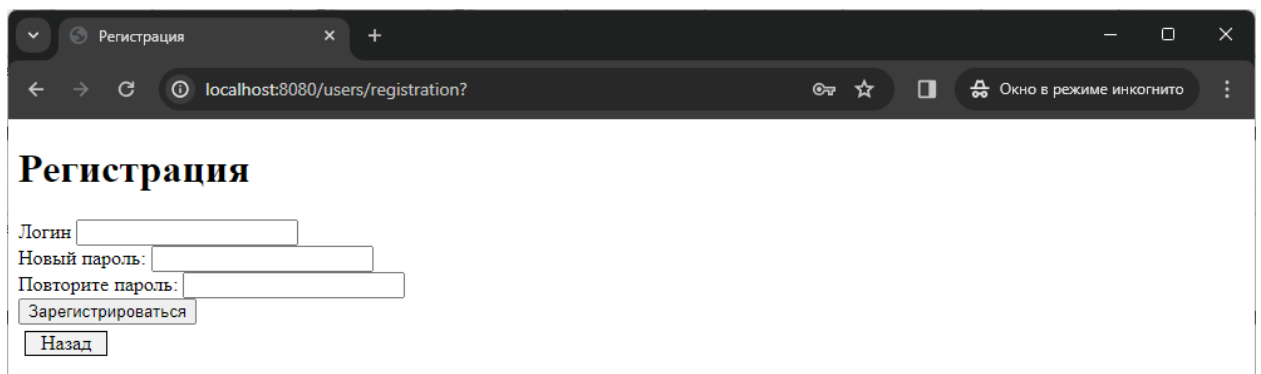


The screenshot shows a web browser window with the title 'Авторизация' and the URL 'localhost:8080/users/login'. The page content includes a heading 'Авторизация', a 'Логин' label with an input field, a 'Пароль' label with an input field, a 'Вход' button, and a 'Регистрация' button.

Авторизация

Логин

Пароль



The screenshot shows a web browser window with the title 'Регистрация' and the URL 'localhost:8080/users/registration?'. The page content includes a heading 'Регистрация', a 'Логин' label with an input field, a 'Новый пароль:' label with an input field, a 'Повторите пароль:' label with an input field, a 'Зарегистрироваться' button, and a 'Назад' button.

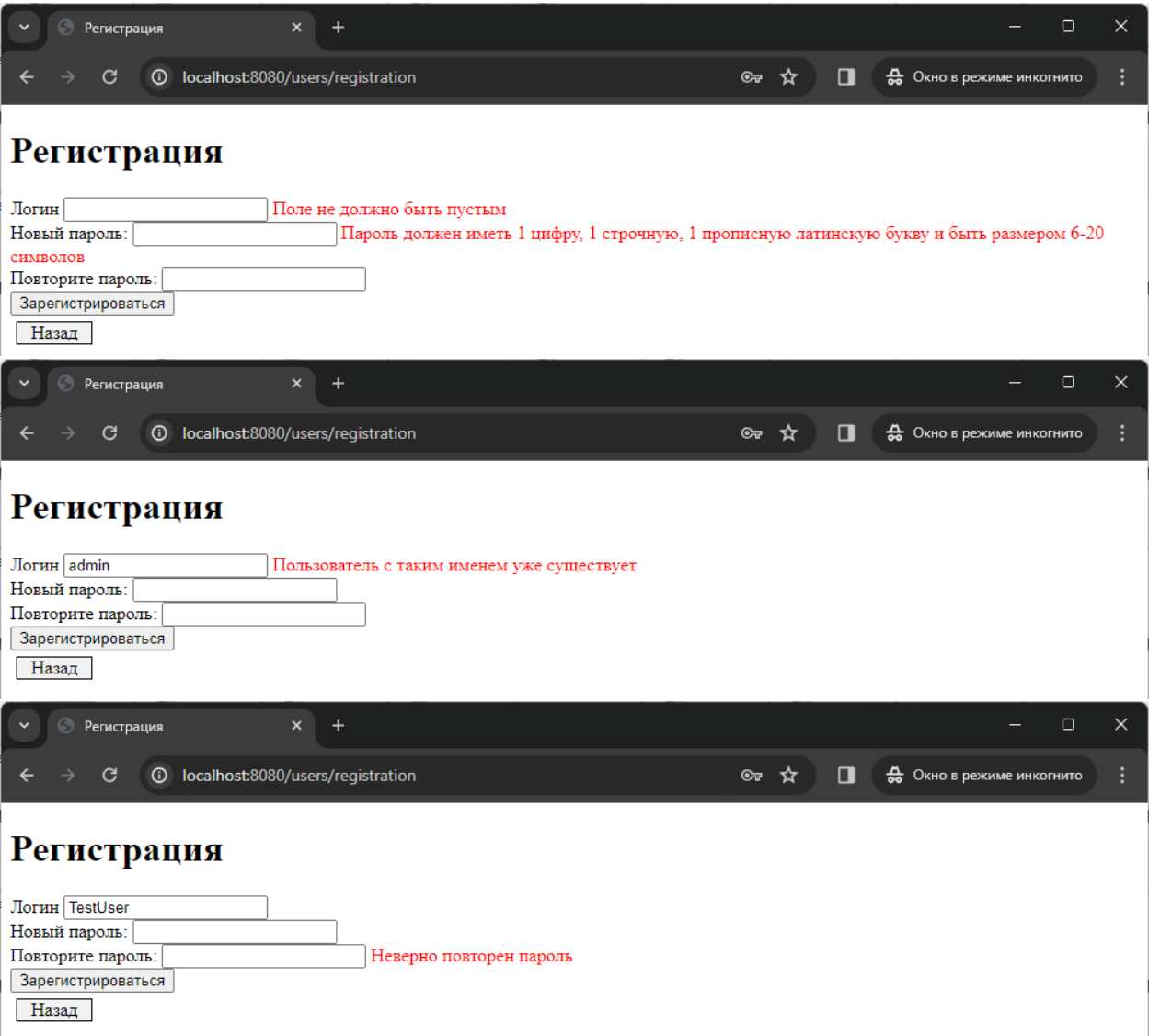
Регистрация

Логин

Новый пароль:

Повторите пароль:

При регистрации поля должны принимать валидные значения. Ниже представлены примеры валидации полей.

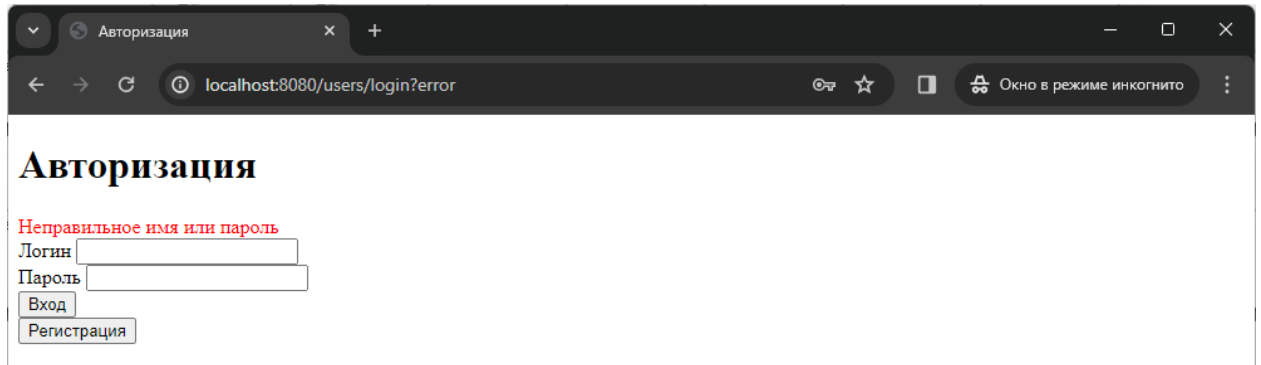


В приложении используется BCrypt шифрование паролей

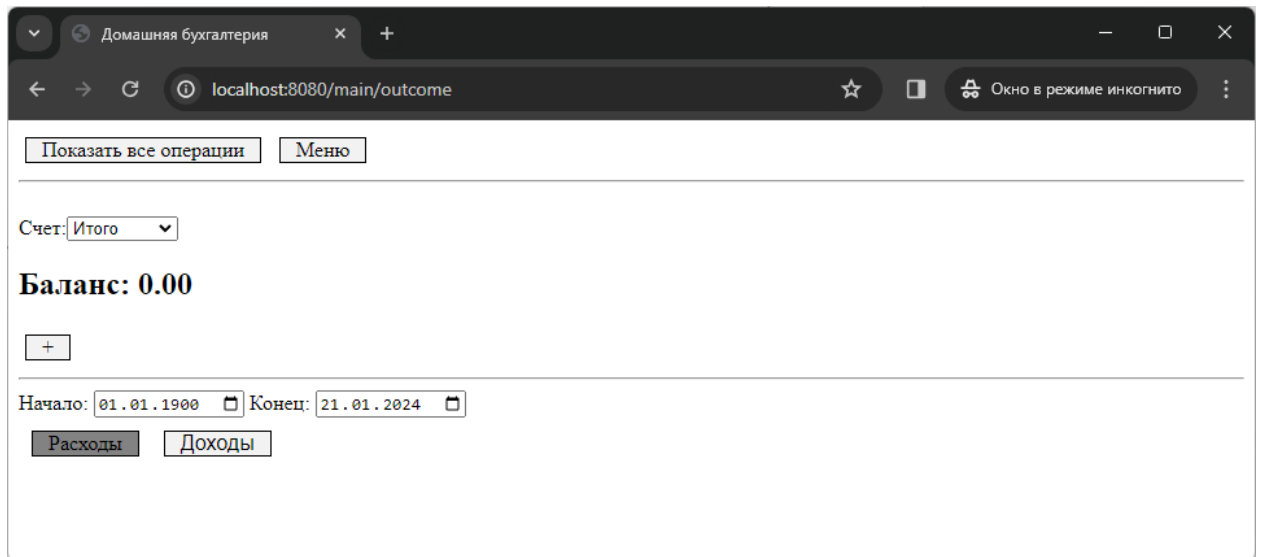
|   | id<br>[PK] integer | login<br>character varying (20) | password<br>character varying (100)                             | role_id<br>integer |
|---|--------------------|---------------------------------|---|--------------------|
| 1 | 33                 | admin                           | \$2a\$10\$hwA5uRU4VqA362s.WA/I4eQEah5xfN.1zQrK1FQBk2DIVMSejGok. | 1                  |
| 2 | 52                 | user                            | \$2a\$10\$sOuuW9JCbl7qCF91PxmaDeeQ3X5vyEUonL4y/rLAWEsANUZriopXm | 2                  |
| 3 | 79                 | TestUser                        | \$2a\$10\$Pgcc55Ciwj53SFxoH8cfOOqb8T/Z5PcPBHJpYw4AzJJOR1leL8Oue | 2                  |

По умолчанию новому пользователю присваивается роль «USER». В дальнейшем корректировать роли может только пользователь с правами администратора. После регистрации пользователя перенаправляет на страницу авторизации.

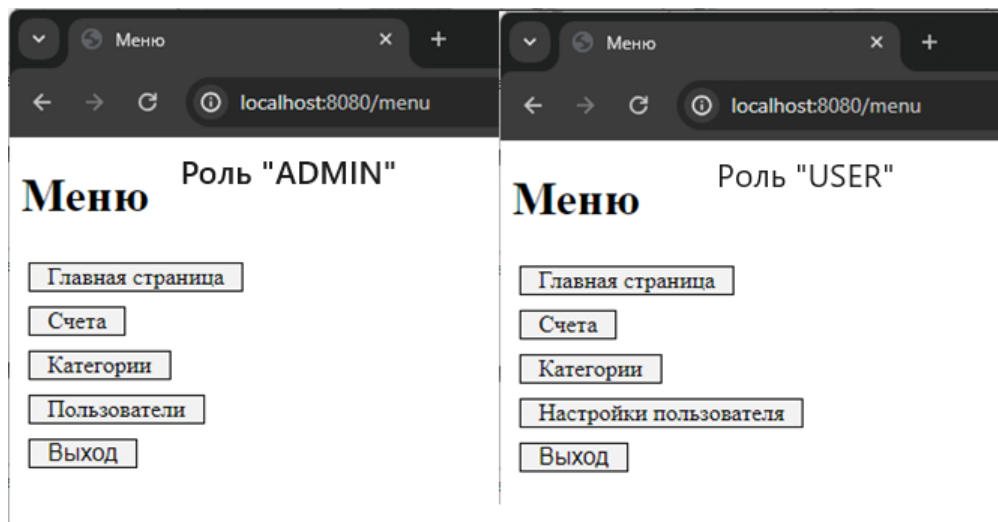
При неудачной аутентификации так же выводится ошибка.



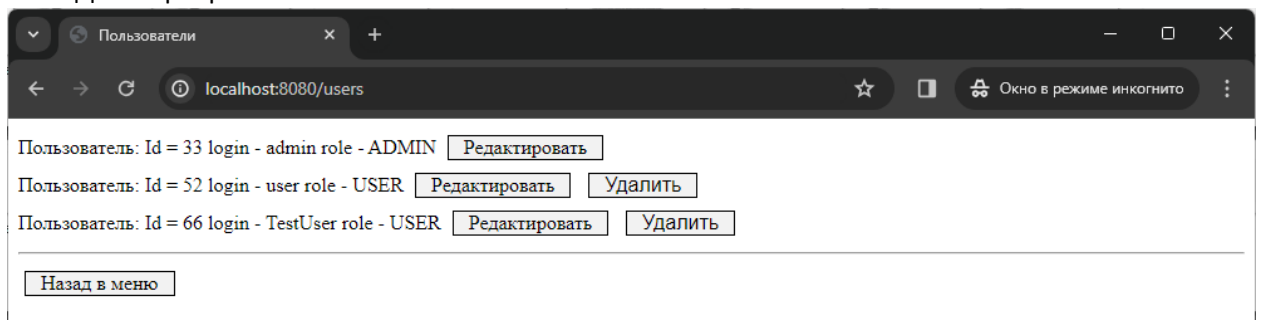
При удачной аутентификации производится переход на главную страницу приложения.



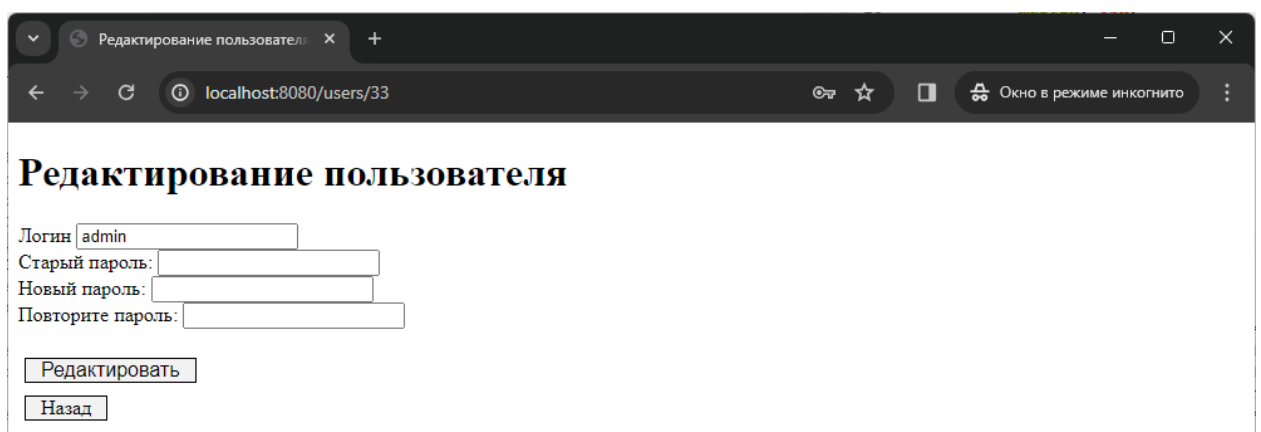
Рассмотрим ее чуть позже. Для начала перейдем в меню приложения. В зависимости от роли пользователя, меню будет отличаться. Каждый пользователь имеет доступ к своим счетам, категориям расходов/доходов и главной странице с операциями расходов/доходов. Пользователи с ролью «ADMIN» имеют доступ к списку пользователей с дальнейшим редактированием и удалением пользователей. Пользователь с ролью «USER» же имеет доступ только к личным настройкам



Рассмотрим страницу со списком пользователей. Здесь мы видим имена зарегистрированных пользователей, их id и роль. На контроллерном уровне реализована защита от удаления пользователем с ролью «ADMIN» самого себя, чтобы избежать ситуации, когда в БД не осталось администраторов. В таком случае присваивание роли «ADMIN» будет возможно только напрямую в СУБД на сервере.



Перейдем на страницу редактирования пользователя. Отмечу, что во избежание вышеуказанного состояния пользователю с ролью «ADMIN» запрещено изменять роль самому себе.



При редактировании пользователя действуют те же правила валидации, что и при его регистрации. Добавляется лишь наличие проверки на соответствие введенного старого пароля действительному. При этом для изменения имени или роли не требуется изменение пароля. Изменим роль TestUser на «ADMIN»

Редактирование пользователя

Логин

Старый пароль:

Новый пароль:

Повторите пароль:

Роль:

Пользователи

Пользователь: Id = 33 login - admin role - ADMIN

Пользователь: Id = 52 login - user role - USER

Пользователь: Id = 66 login - TestUser role - ADMIN

Теперь удалим пользователя

Пользователи

Пользователь: Id = 33 login - admin role - ADMIN

Пользователь: Id = 52 login - user role - USER

Авторизируемся под пользователем «user» и перейдем в меню на страницу с настройками. Как видим пользователь с правами «USER» имеет возможность корректирования собственного логина и пароля без изменения роли.

Редактирование пользователя

Логин

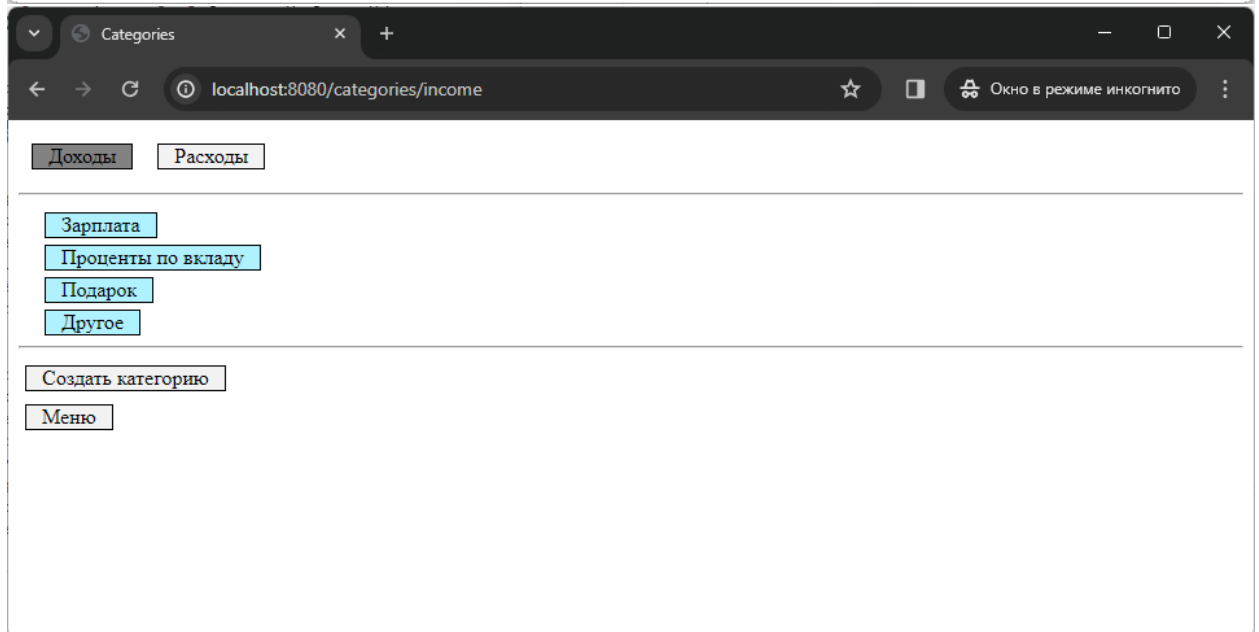
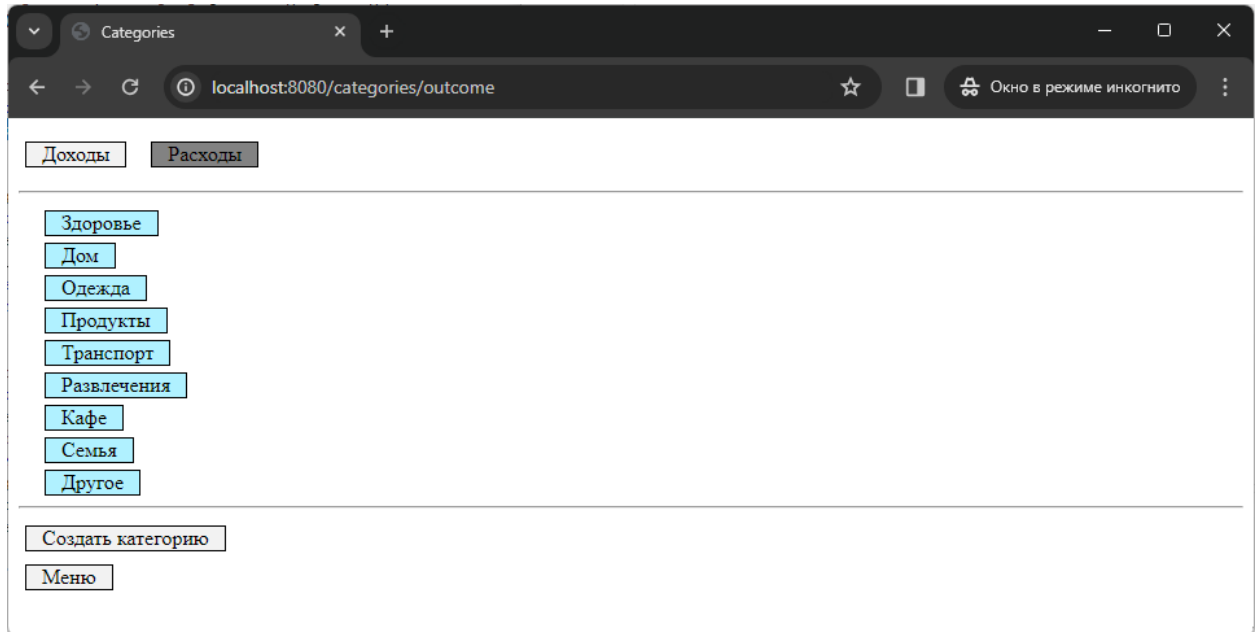
Старый пароль:

Новый пароль:

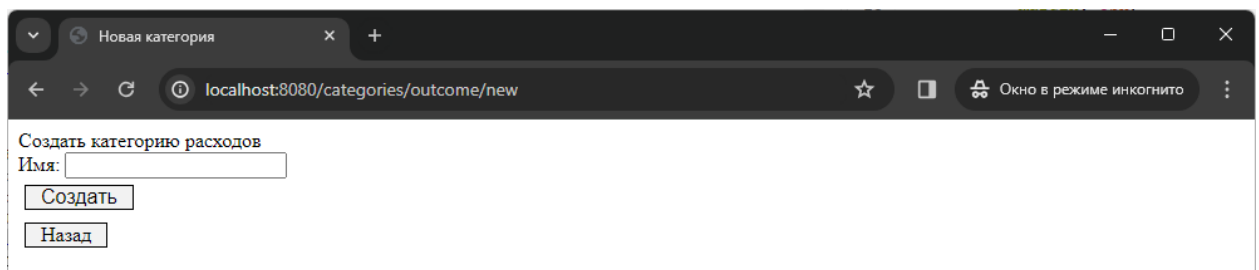
Повторите пароль:

Вот мы успешно зарегистрировались и прошли авторизацию и готовы перейти к самой сути приложения.

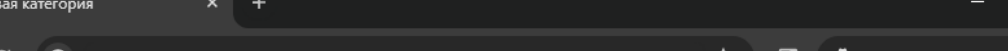
В разделе «Категории» мы видим категории расходов и доходов пользователя. Пользователю при регистрации создаются распространенные категории по умолчанию



Пользователь может создавать новые категории, редактировать и удалять существующие (в том числе созданные по умолчанию). При создании и редактировании категорий так же предусмотрена валидация полей.



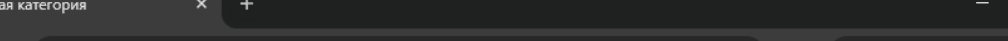




Создать категорию расходов

Имя:

Категория с таким именем уже существует



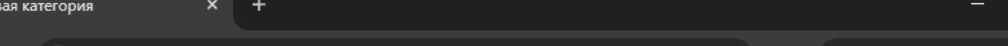
Создать категорию расходов

Имя:  Поле не должно быть пустым

A screenshot of a web browser window. The address bar shows "localhost:8080/categories/outcome". The page title is "Создать категорию расходов". Below the title, there is a label "Имя:" followed by a text input field containing "ОченьОченьОченьОченьОч". To the right of the input field, there is a red error message: "Поле не должно превышать 20 символов". Below the input field, there are two buttons: "Создать" and "Назад".


Создать категорию расходов

Имя:  Поле не должно превышать 20 символов



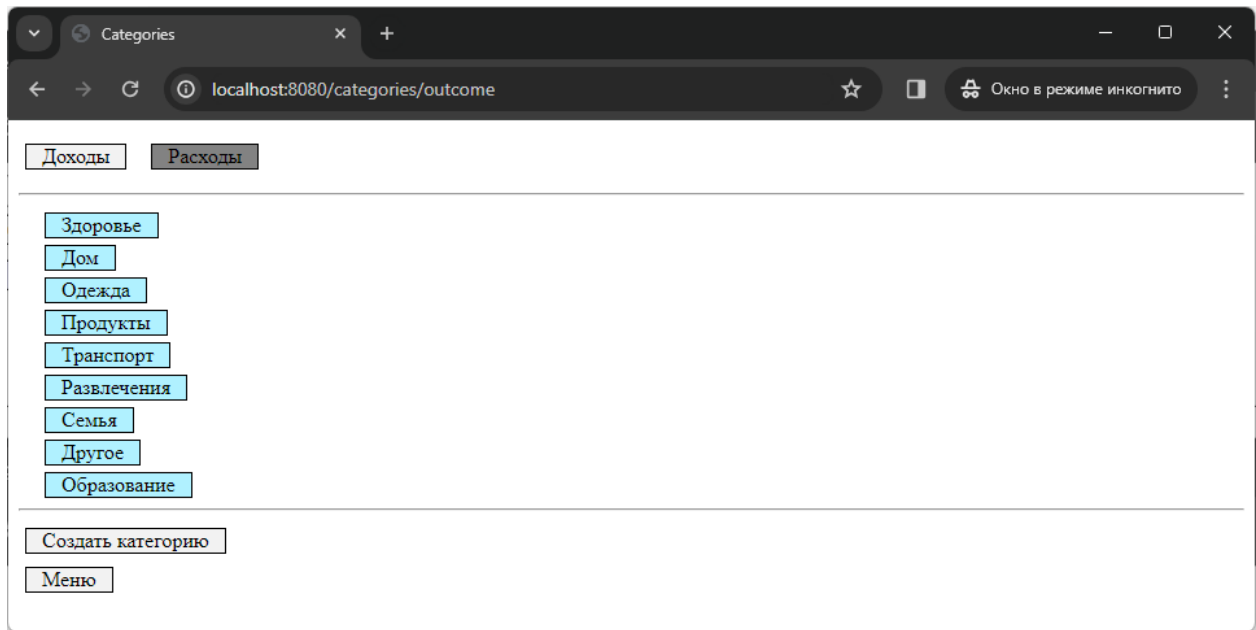
Создать категорию расходов

Имя:



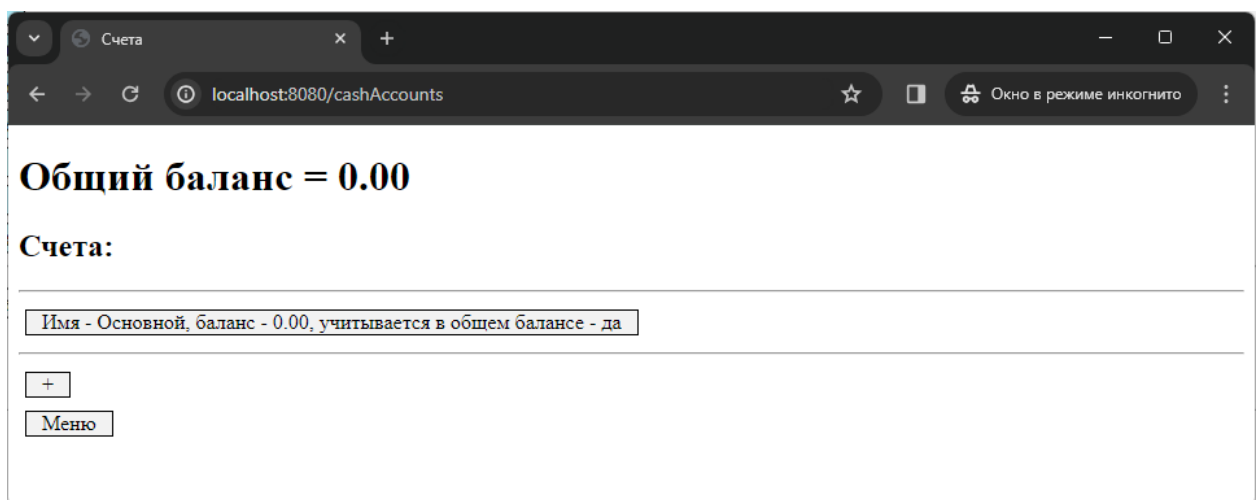
The screenshot shows a web browser window with the address bar displaying 'localhost:8080/categories/outcome'. The page has two tabs: 'Доходы' (Income) and 'Расходы' (Expenses). The 'Расходы' tab is active. Below the tabs, there is a list of categories: 'Здоровье', 'Дом', 'Одежда', 'Продукты', 'Транспорт', 'Развлечения', 'Кафе', 'Семья', 'Другое', and 'Образование'. At the bottom, there are two buttons: 'Создать категорию' (Create category) and 'Меню' (Menu).

Для примера удалим категорию расходов «Кафе»



Все операции с категориями доходов аналогичные

Далее перейдем на страницу со счетами. Базово пользователю создается пустой счет с именем «Основной».



Счета помимо имени и баланса также обладают булевой характеристикой «Учитывается в общем балансе». Необходимо это для фильтрации отображения общего баланса. Таким образом можно разграничивать счета накопительные от счетов бытового потребления. Приведу пример:

Счета

localhost:8080/cashAccounts

Окно в режиме инкогнито

# Общий баланс = 33000.00

## Счета:

Имя - Основной, баланс - 15000.00, учитывается в общем балансе - да

Имя - Копим на машину, баланс - 200000.00, учитывается в общем балансе - нет

Имя - На продукты, баланс - 10000.00, учитывается в общем балансе - да

Имя - Копим на отпуск, баланс - 50000.00, учитывается в общем балансе - нет

Имя - На топливо, баланс - 8000.00, учитывается в общем балансе - да

+

Меню

При создании счетов так же существуют правила валидации полей. Поля не должны быть пустым, имя должно быть определенного размера, а так же у одного пользователя не может быть несколько счетов с одинаковыми именами

Создать счет

localhost:8080/cashAccounts/new

Окно в режиме инкогнито

Имя:

Баланс:

Учитывать в общем балансе: ☐ Да ☒ Нет

Создать

Назад

Создать счет

localhost:8080/cashAccounts

Окно в режиме инкогнито

Имя:  Поле не должно быть пустым

Баланс:  Поле не должно быть пустым

Учитывать в общем балансе: ☐ Да ☒ Нет

Создать

Назад

Создать счет

localhost:8080/cashAccounts

Имя: Основной Такой счет уже существует

Баланс: Поле не должно быть пустым

Учитывать в общем балансе: ☐ Да ☒ Нет

Создать

Назад

Создать счет

localhost:8080/cashAccounts

Имя: ОченьОченьОченьОченьОч Поле не должно превышать 20 символов

Баланс: Поле не должно быть пустым

Учитывать в общем балансе: ☐ Да ☒ Нет

Создать

Назад

При этом баланс может принимать отрицательные значения. Таким образом можно вести учет займов не боясь потерять эту информацию в своих чертогах разума.

Создать счет

localhost:8080/cashAccounts/new

Имя: Занял Серге

Баланс: -3000

Учитывать в общем балансе: ☐ Да ☒ Нет

Создать

Назад

Так же, как все остальное, счета мы можем редактировать и удалять

Счет Занял Серге

localhost:8080/cashAccounts/48

Имя: Занял Серге

Баланс: -3000.00

Учитывать в общем балансе: ☐ Да ☒ Нет

Редактировать

Удалить

Назад

И вот собрав все во едино мы переходим к главной сущности приложения. К операциям доходов и расходов. Рассмотрим главную страницу приложения.

Домашняя бухгалтерия

localhost:8080/main/outcome

Показать все операции Меню

Счет: Итого

**Баланс: 33000.00**

+

Начало: 01.01.1900 Конец: 21.01.2024

Расходы Доходы

На данной странице появляется возможность создавать операции и получать их представление с различными видами фильтрации. Для наглядности создадим несколько операций в различных категориях, счетах и в различные дни

Создание операции

localhost:8080/operations/outcome/new

Расход:

Сумма:

Счет: Основной

Категория: Здоровье

Дата:

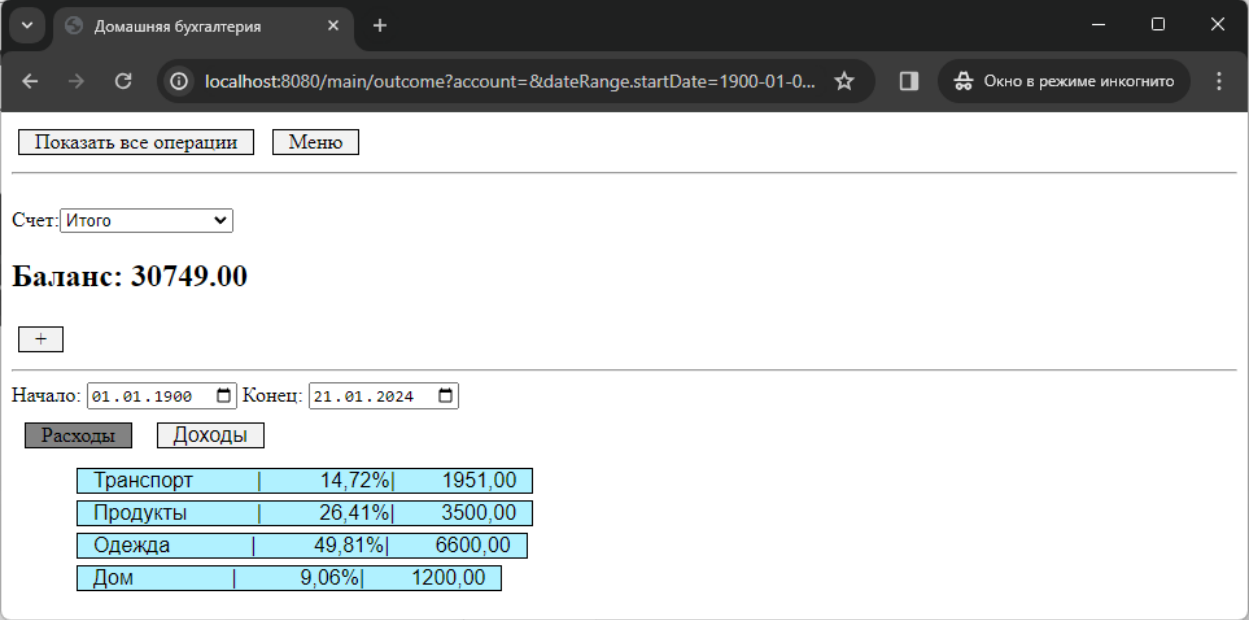
Комментарий:

Создать

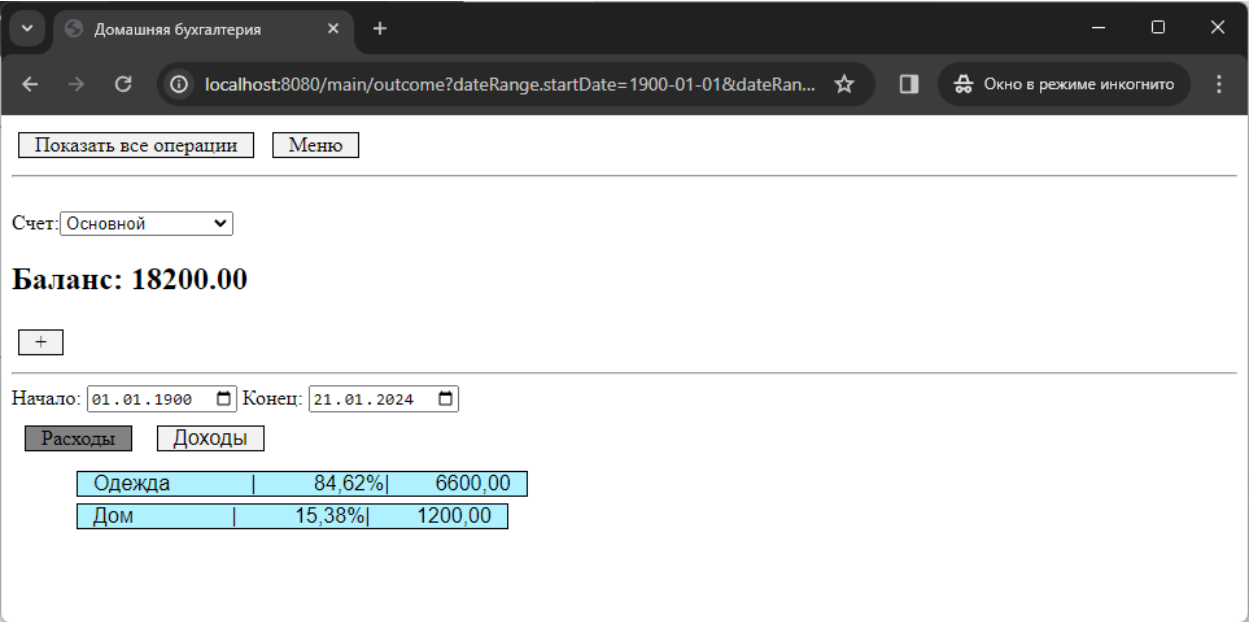
Главная страница

На странице создания мы должны выбрать счет и категорию из существующих, указать дату и сумму операции. Данные поля не могут быть пустыми, дата к тому же не может быть позже текущей. Комментарий является опциональным, но он так же должен быть валидным по размеру

Создав несколько операций мы видим, что на главной странице появляется отображение того, сколько было потрачено/приобретено для определенного счета по категориям, с расчетом процентного соотношения. По умолчанию информация отображается для всех счетов суммарно в период с 01.01.1900 (предполагается, что у пользователя нет доходов или расходов до этой даты, хотя установить можно дату и меньше. По низу ограничений нет). Баланс указан в соответствии с расставленными нами фильтрами в разделе «Счета»



Допустим мы хотим посмотреть все операции на основном счете за все время. Выбираем основной счет и видим баланс конкретно на этом счету и сумма операций по категориям, произведенных за все время по этому счету.



Категории являются кликабельными и переводят нас на отображение перечня операций, отфильтрованных по дате, счету и категории.

Перейдем к операциям по категории «Дом»

Операции

Начало: 01.01.1900 Конец: 21.01.2024

[дата=2024-01-07, расход = 1200.00, счет: Основной, категория: Дом, комментарий: Обои]

+

Главная страница

Из окна отображения мы так же можем перейти в форму создания операции. И кликнув по самой операции мы можем перейти в форму ее редактирования и удаления

Редактирование операции

localhost:8080/operations/outcome/36

Расход:

Сумма: 1200.00

Счет: Основной

Категория: Дом

Дата: 07.01.2024

Комментарий: Обои

Редактировать

Удалить

Главная страница

Так же мы можем получить отображение всех операций по отдельным счетам, либо по всем счетам сразу. Для этого на главной странице необходимо выбрать счет отображения и кликнуть на кнопку «Показать все операции». Например отобразим все операции расхода

Операции

Начало: 01.01.1900 Конец: 21.01.2024

[дата=2024-01-05, расход = 6600.00, счет: Основной, категория: Одежда, комментарий: Ботинки]

[дата=2024-01-07, расход = 1200.00, счет: Основной, категория: Дом, комментарий: Обои]

[дата=2024-01-10, расход = 1500.00, счет: На продукты, категория: Продукты, комментарий: ]

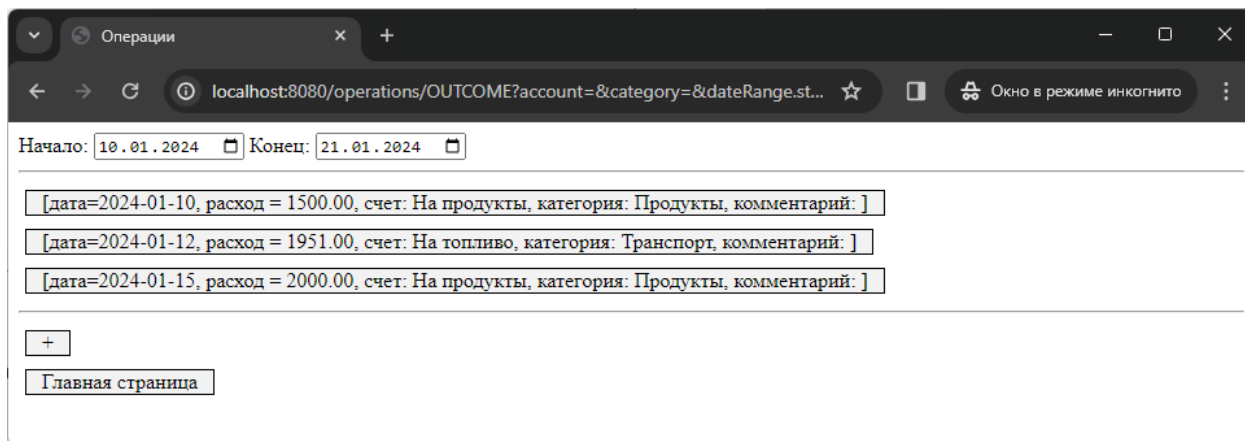
[дата=2024-01-12, расход = 1951.00, счет: На топливо, категория: Транспорт, комментарий: ]

[дата=2024-01-15, расход = 2000.00, счет: На продукты, категория: Продукты, комментарий: ]

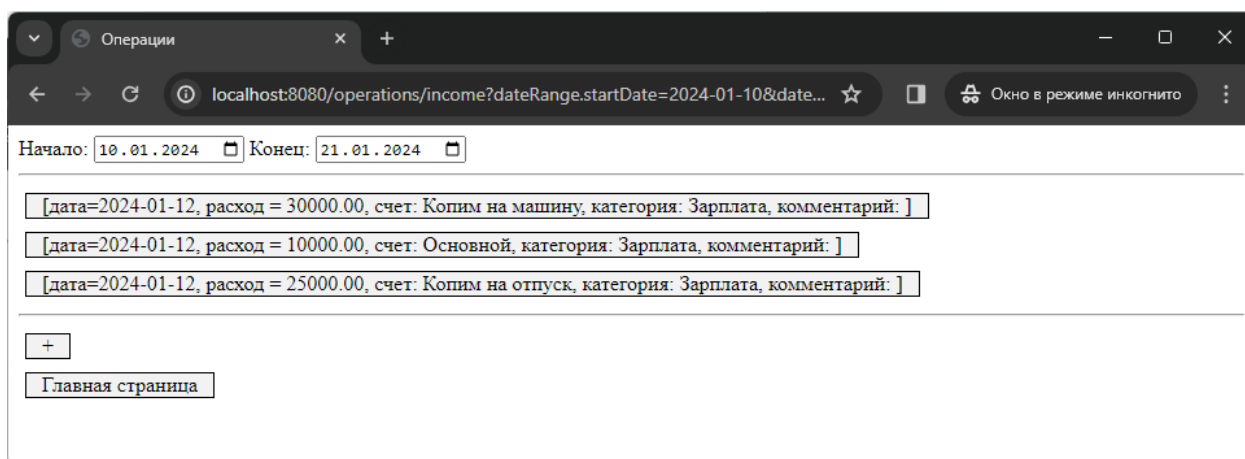
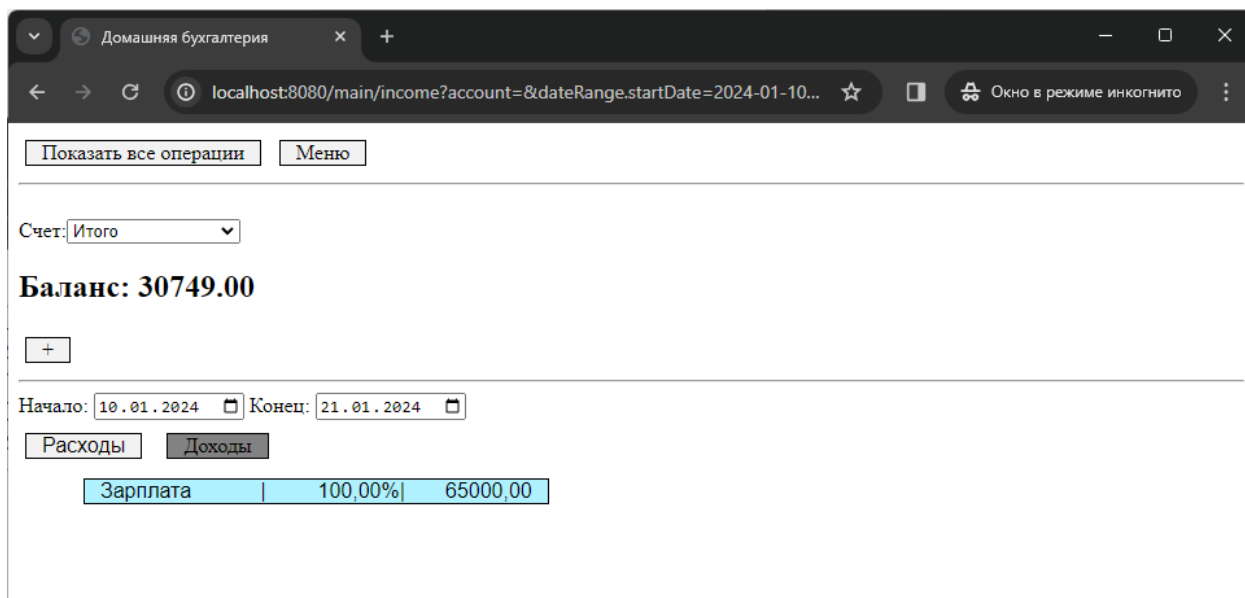
+

Главная страница

Отфильтруем их по дате, отобразив все операции позже 10.01.2024



Аналогично мы можем увидеть и доходы



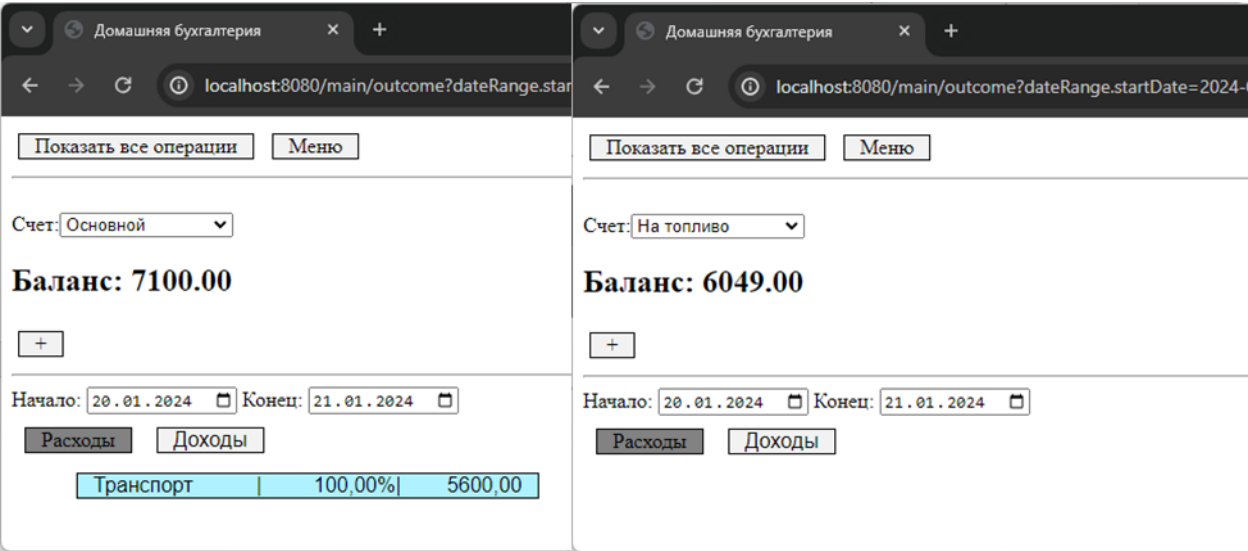
Таким образом видим, что 12.01.2024 пользователь получил зарплату и распределил ее по категориям. Часть отложил на бытовые нужды, часть определил на накопительные счета



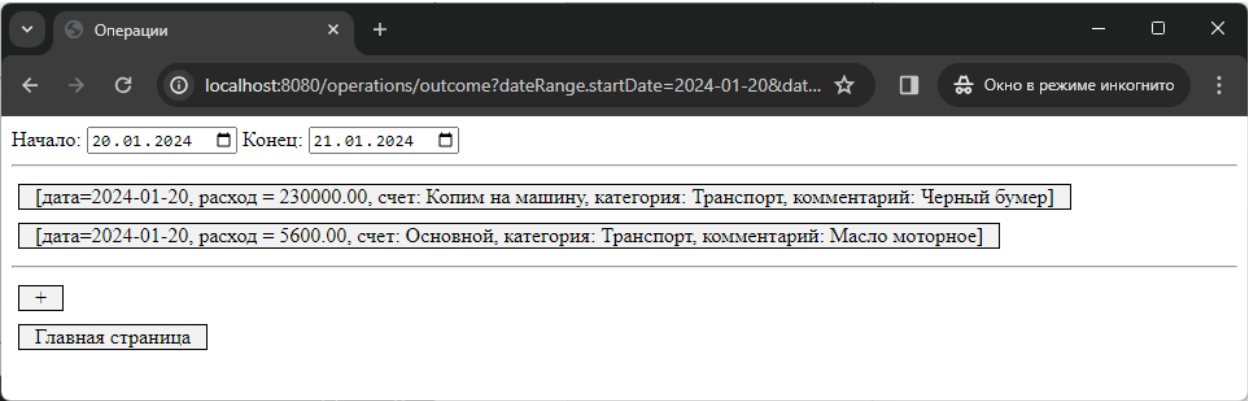
Так же хотелось бы продемонстрировать как редактирование операций влияет на изменение в балансе счетов. Смоделируем ситуацию. Пользователь, не стерпев мук ожидания решил, что ему хватит копить на машину и взяв свои честно накопленные 230 000р. по совету соседа купил себе новенький двадцатипятилетний премиальный немецкий автомобиль. По пути домой он купил моторное масло на АЗС по скидке. Сев на кожаное сидение, он открыл приложение и занес в расход покупку, отнеся ее к счету «Основной» и категории трат «Транспорт». Но доехав до дома он был слегка разочарован, поняв, что, судя по расходу масла, придется отнести покупку в категорию «Топливо». Благо хоть кэшбэк 600р. пришел и получилось немного сэкономить.

Рассмотрим эту ситуацию в приложении.

Имеем баланс на основном счете 7100р, на топливо 6049р и покупку в 5600р



Переходим в отображение операций, затем в редактирование, меняем счет, уменьшаем сумму, жмем редактировать и наблюдаем магию



Редактирование операции

localhost:8080/operation

Расход:

Сумма: 5600,00

Счет: Основной

Категория: Транспорт

Дата: 20.01.2024

Комментарий: Масло моторное

Редактировать

Удалить

Главная страница

Редактирование операции

localhost:8080/operations/outcome/39

Расход:

Сумма: 5000,00

Счет: На топливо

Категория: Транспорт

Дата: 20.01.2024

Комментарий: Масло моторное

Редактировать

Удалить

Главная страница

Как мы видим, на основном счете баланс увеличился на 5600р. и пропала операция расхода  
При этом на счете «На топливо» появился расход в 5000р. и ровно столько же отнялось от баланса

Домашняя бухгалтерия

localhost:8080/main/outcome?account=Oc

Показать все операции

Меню

Счет: Основной

Баланс: 12700.00

+

Начало: 20.01.2024

Конец: 21.01.2024

Расходы

Доходы

Домашняя бухгалтерия

localhost:8080/main/outcome?dateRange.startDate=2024-01

Показать все операции

Меню

Счет: На топливо

Баланс: 1049.00

+

Начало: 20.01.2024

Конец: 21.01.2024

Расходы

Доходы

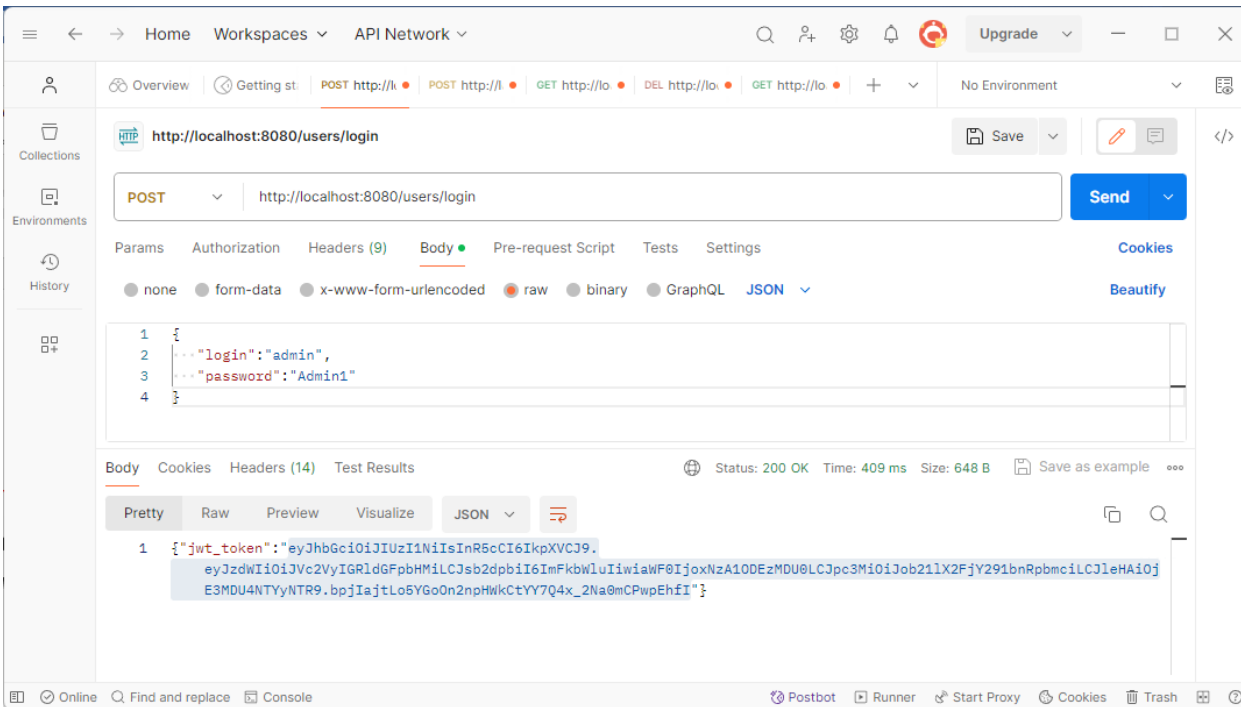
|           |         |         |
|-----------|---------|---------|
| Транспорт | 100,00% | 5000,00 |
|-----------|---------|---------|

Мораль пользователь усвоил прекрасно: «Приложение 10/10, сосед 1/10»

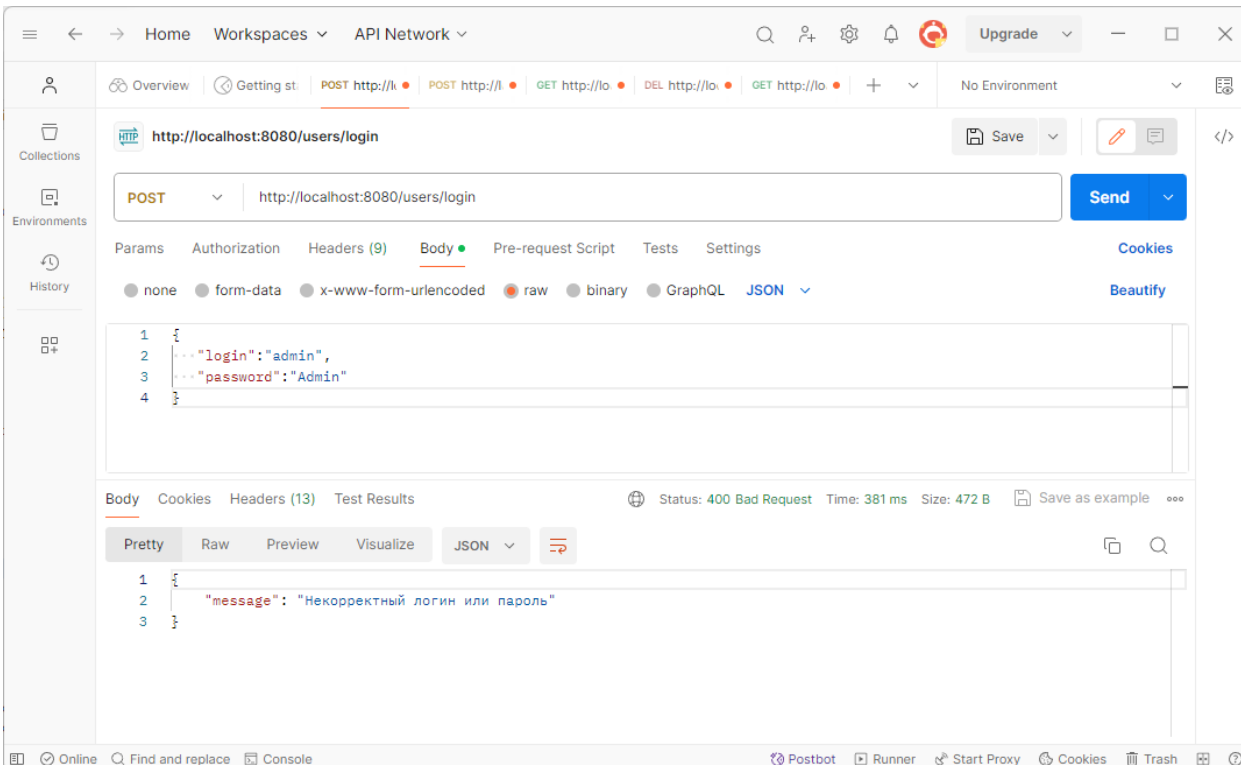
## HomeAccounting REST

Обе версии приложения отличаются лишь на контроллерном уровне. В REST микросервисе контроллеры принимают HTTP запросы с теми же маппингами, только данные теперь передаются не в параметрах запроса, а в его теле в формате JSON.

При регистрации или аутентификации в теле запроса передается JSON с необходимыми данными и при удачной регистрации/аутентификации приложение возвращает JSON с JWT токеном, необходимым для дальнейшей авторизации пользователя. Все последующие запросы посылаются с полученным токеном в заголовке запроса Authorization.

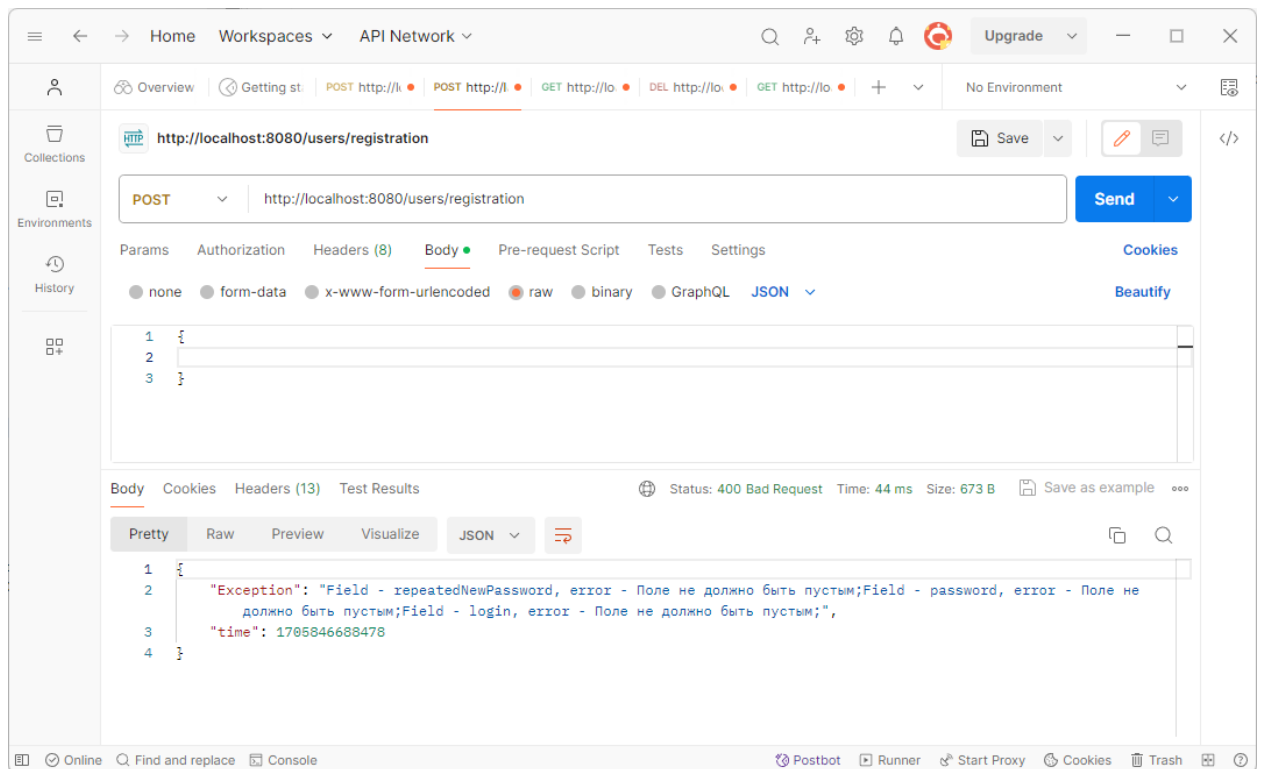


Ниже приведен пример неудачной аутентификации



При регистрации действуют те же правила валидации, нарушение которых возбуждает исключение наследника класса `FieldNotFoundException` и обработчик исключений в абстрактном родительском классе контроллеров отдает клиенту ответ с информацией об ошибке.

Ниже приведен пример передачи запроса на регистрацию с пустым телом



Все остальные запросы возвращают ту же информацию, что и в Web приложении. Различием является разве что отсутствие главной страницы и как следствие `HomePageService` и `MainController`. Приведу пример нескольких представлений для используемого ранее в примерах пользователя `TestUser`:

Отображение счетов и общего баланса

Overview

Getting st

POST http://lo

POST http://lo

GET http://lo

DEL http://lo

GET http://lo

+

No Environment

http://localhost:8080/cashAccounts

Save

</>

GET

http://localhost:8080/cashAccounts

Send

Params

Authorization

Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ...

TestUser authorization

Key

Value

Description

Body

Cookies

Headers (14)

Test Results

Status: 200 OK

Time: 90 ms

Size: 901 B

Save as example

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

{

"generalBalance": 20249.00,

"accounts": [

{

"id": 42,

"balance": 12700.00,

"containInGenBalance": true,

"name": "Основной"

}

,

{

"id": 43,

"balance": 0.00,

"containInGenBalance": false,

"name": "Копим на машину"

}

,

{

"id": 44,

"balance": 6500.00,

"containInGenBalance": true,

"name": "На продукты"

}

,

{

"id": 45,

"balance": 75000.00,

"containInGenBalance": false,

"name": "Копим на отпуск"

}

,

{

"id": 46,

"balance": 1049.00,

"containInGenBalance": true,

"name": "На топливо"

}

]

}

Online

Find and replace

Console

Postbot

Runner

Start Proxy

Cookies

Trash

Категории расходов

HomeWorkspacesAPI Network

Search Postman

Invite

Upgrade

OverviewGetting startedPOST http://localh...POST http://localh...GET http://localh...DEL http://localh...GET http://localh...+No Environment

http://localhost:8080/categories/outcome

GEThttp://localhost:8080/categories/outcomeSend

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettingsCookies

Connectionkeep-alive

AuthorizationBearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOi...

TestUser authorization

KeyValueDescription

BodyCookiesHeaders (14)Test ResultsStatus: 200 OKTime: 30 msSize: 738 BSave as example

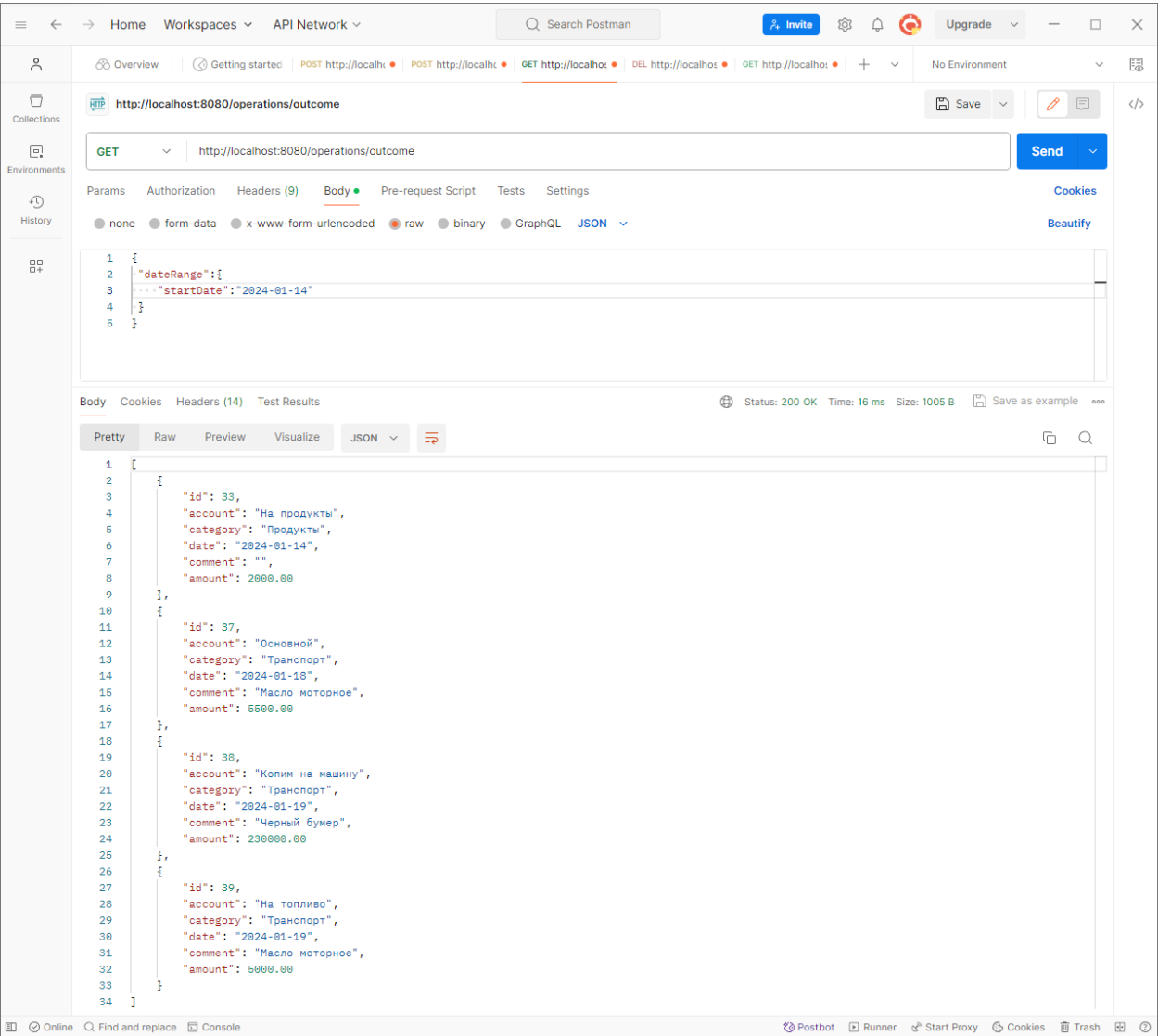
PrettyRawPreviewVisualizeJSON

```
1 [{
2   {
3     "id": 76,
4     "name": "Здоровье"
5   },
6   {
7     "id": 77,
8     "name": "Дом"
9   },
10  {
11    "id": 78,
12    "name": "Одежда"
13  },
14  {
15    "id": 79,
16    "name": "Продукты"
17  },
18  {
19    "id": 80,
20    "name": "Транспорт"
21  },
22  {
23    "id": 81,
24    "name": "Развлечения"
25  },
26  {
27    "id": 83,
28    "name": "Семья"
29  },
30  {
31    "id": 84,
32    "name": "Другое"
33  },
34  {
35    "id": 85,
36    "name": "Образование"
37  }
38 }]
```

OnlineFind and replaceConsole

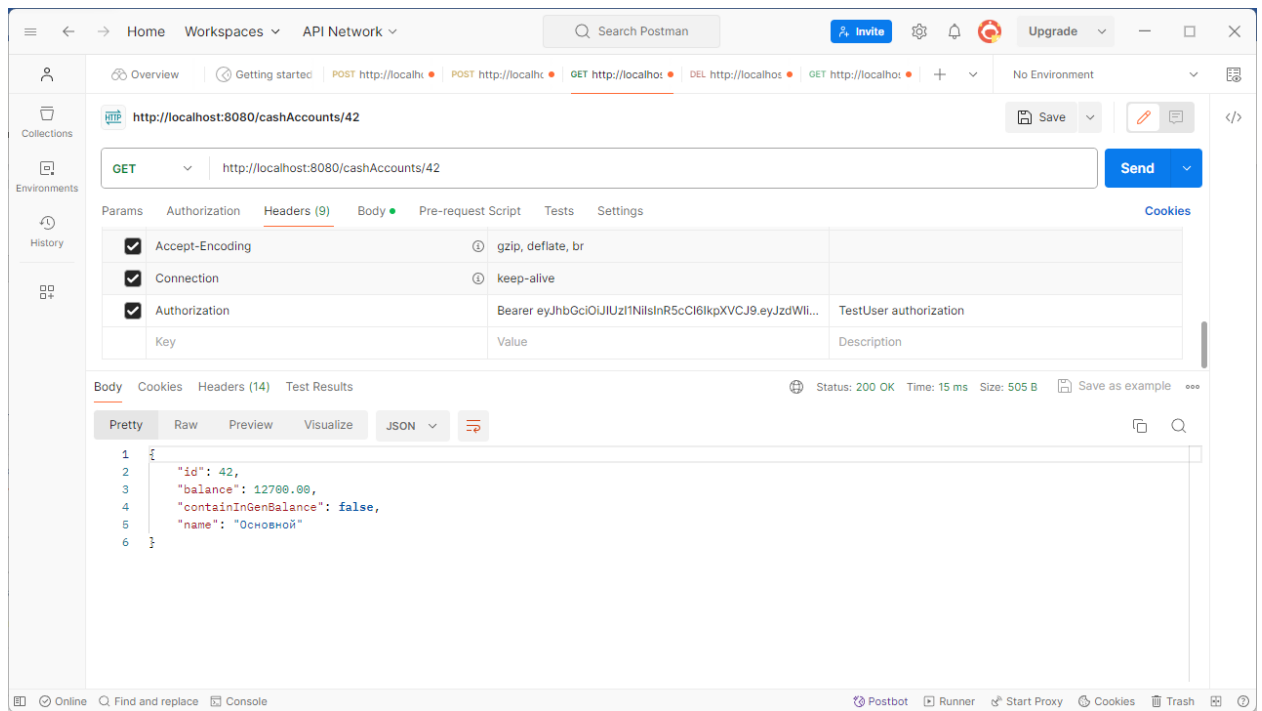
PostbotRunnerStart ProxyCookiesTrash

Расходы всех счетов и категорий с 14.01.2024г.

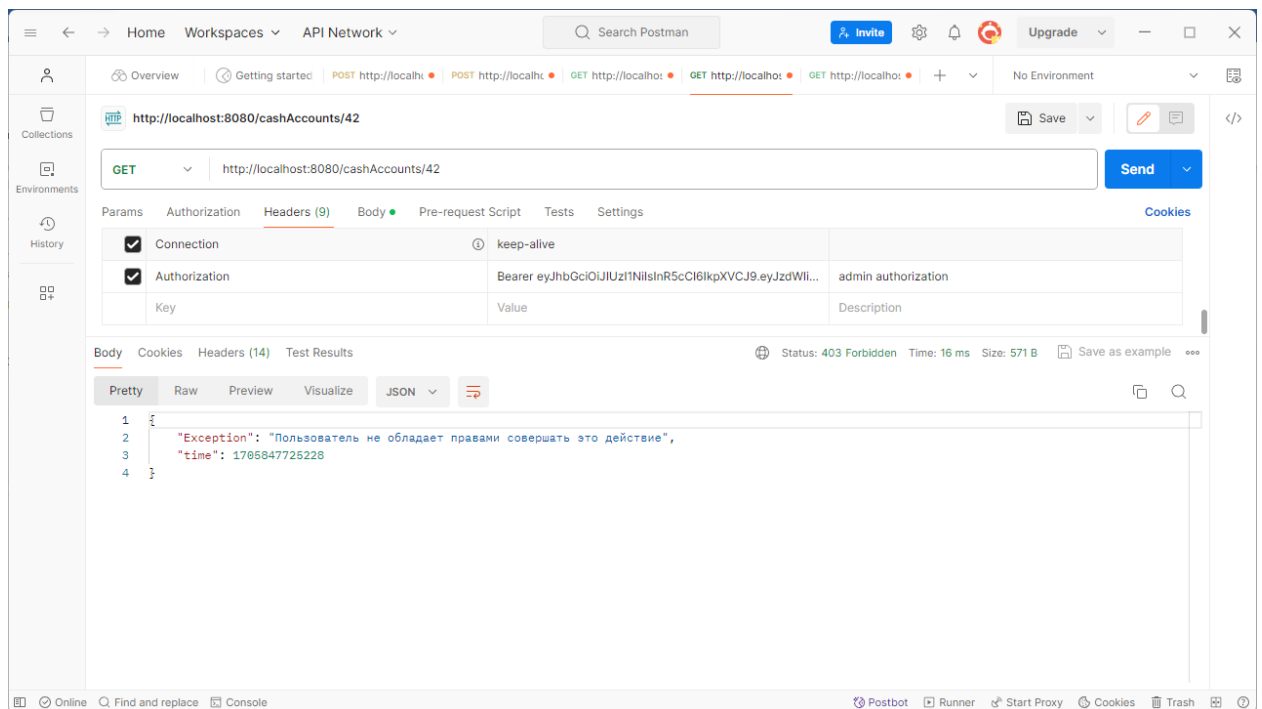


Так же именно на примере REST приложения хотелось бы продемонстрировать поведение программы при попытке манипулировать данными, к которым пользователь не имеет доступа.

Например, по запросу `"/cashAccounts/{id}"` мы можем получить информацию о конкретном счете с таким идентификатором. Убедимся, сделав запрос под пользователем – владельцем счета



Как видим вернулся ответ со статусом 200 и информацией об счете. Но что будет, если мы сделаем такой же запрос авторизуясь под другим пользователем? Пусть это будет “admin”.



Как видно чиновничеству нет места в этом приложении. Будь ты хоть администратор, хоть лорд английский, но личные данные пользователей – это их личные данные и не авторизуясь под конкретным пользователем получить его данные возможно только имея непосредственный доступ к БД на сервере. То же самое касается запросов на редактирование и удаление данных