# Общие данные

Проекта «Домашняя бухгалтерия» это средство для контроля семейной бухгалтерии. Основная идея проекта:

– все члены семьи, у которых есть доступ к семейному бюджету имеют доступ к общим данным о финансовых средствах, которыми семья манипулирует.

- пользователь может хранить все данные в одном месте. Не нужно устанавливать несколько приложений. Домашняя бухгалтерия позволяет планировать движение финансовых средств и полностью их контролировать.

В приложении доступны следующие модули:

1. [Планирование расходов](#_Планирование_расходов:)
2. [Планирование доходов](#_Планирование_доходов:)
3. [Список покупок](#_Обмен_валюты:)
4. [Обмен валюты](#_Список_продуктов:)

## Планирование расходов:

Тесно связан с модулем «список покупок». Пользователь может выбрать оптимальный список товаров, которые ему необходимо приобрести, используя оптимизацию расходов на покупки по бизнес логике совокупности расходов на проезд, доставку продуктов. Стоимости продукта и его наличия в магазине.

Кроме этого, пользователь может создать список планируемых расходов и доходов на месяц/год/неделю и увидеть какое количество средств у него остается на финансовом счету. Это позволит более продуктивно тратить финансовые средства и контролировать их движение.

Функции модуля планирование расходов:

## Планирование доходов:

Пользователь может спланировать свои основные доходы (такие, как зарплата, наследство и другие варианты гарантированного появления средств на счету пользователя) это поможет более грамотно планировать расходы, понимая какой приход и расход финансовых средств ожидает пользователя в текущем периоде времени.

## Обмен валюты:

Контроль средств пользователя в различных мировых валютах.

Реализация:

Пользователь может выбирать валюту, которой он хочет оперировать при заполнении расходов – доходов. Выбор валюты возможен из списка «дефолтных» валют в приложении.

Список дефолтных валют загружается в приложение из Json файла (currency.json – в корне проекта). После того, как пользователь выбрал основную валюту, для каждой конкретной операции он может перевести основную валюту в любую другую. Перевод валюты происходит га основе национального курса Банка Израиля.

Для обеспечения релевантных данных по обмену валют, каждую ночь в 3:00 приложение получает актуальные данных с сайтов национальных банков выбранных валют (в бесплатной версии доступны валюты:

* Израильский шекель
* Американский доллар
* Украинская гривна

В платной версии пользователь может связаться с администратором и «заказать» другие виды валют, если они ему необходимы)

Таким образом пользователь в любой момент может изменить валюту каждой конкретной операции, и при этом данные будут релевантные.

## Список продуктов:

Парсер в связке с со списком продуктов позволяет выбрать оптимальное решение на предмет того, где, когда и за какие средства нужно покупать тот или иной товар.

### Функции модуля списка покупок:

1. Составлять список покупок
2. Хранить ранее созданных списков
3. Учет расходов на покупки
4. Возможность составлять заметки об определенном продукте
5. Возможность выбирать самый удобный магазин (зависит от наличия товара из списка покупок, а так же от цены товара в конкретном магазине)
6. Сравнивать наличие товара и его стоимость в разных магазинах
7. Планировать меню на определенный период, и исходя из меню – составлять список продуктов (не реализовано)
8. Хранить «кулинарную книгу», в которую может собирать необходимые рецепты вместе со списком продуктов, необходимых для приготовления блюда. (не реализовано)

### Реализация:

Для того, что бы гарантировать пользователю релевантные данные о наличии продуктов в конкретном магазине или о стоимости продукта, каждую ночь парсер проходит по интернет сайтам крупных торговых сетей и собирает информацию о товаре, его стоимости и наличии в магазине.

# ДОКУМЕНТАЦИЯ API

<https://documenter.getpostman.com/view/5121731/SzzoaFg1?version=latest>

# Сервисы, из которых состоит приложение.

|  |  |  |
| --- | --- | --- |
| Название микро  сервиса | Описание микросервиса | порт |
| [config-service](#_config-service) | Сервис конфигурации. Хранит ссылку на git-hub. По ссылке хранятся все application-properties всех микро сервисов |  |
| eureka-service | Сервис Эврики. Приложение, в котором хранится информация обо всех клиентских сервисных приложениях. Каждая служба Micro регистрируется на сервере Eureka, и сервер Eureka знает все клиентские приложения, работающие на каждом порту и IP-адресе. |  |
| [zuul-service](#_zuul-service) | Приложение-шлюз, которое обрабатывает все запросы и выполняет динамическую маршрутизацию приложений микросервисов  Этот же приложение проводит авторизацию пользователя и проверяет наличие у него разрешений для выполнения тех или иных действий |  |
| [user-service](#_user-service) | Сервис, который хранит информацию о пользователе, наличии его родственников, зарегистрированных в приложение, а так же предоставляет возможность регистрации пользователя, подтверждения его электронной почты и смены пароля, в случае, 6сли пользователь его забыл. + CRUD |  |
| [bills-service](#_bills-service) | Сервис, который хранит всю информацию о счетах пользователя, и обо всем, что связанно с этими счетами. Тут же происходит добавление операций, категорий и подкатегорий расходов/доходов. |  |
| [currency-service](#_currency-service) | Сервис, который хранит информацию о валютах и об их курсе. |  |
| [crawler-service](#_crawler-service) | Сервис, который занимается парсингом. Он парсит: - национальные банки, что бы иметь актуальную информацию о курсе валют -магазины что бы иметь актуальную информацию о наличии товара и его стоимости |  |
| send-email-service | Сервис, который занимается отправкой писем |  |

# Порядок загрузки

1. Config-service
2. Eureka-service (берет данные для загрузки из config-service)
3. Zuul-service
4. Остальные сервисы

# config-service

хранит в себе ссылку на git репозиторий, в котором хранятся application-properites для все сервисов (<https://github.com/smilyk/home-accounting-2-repo>)

# zuul-service

## безопасность

|  |  |
| --- | --- |
| Подтверждение электронной почты | /user-service/users/send-email-service/\*\* |
| Регистрация пользователя | /user-service/users |
| логин | /user-service/users/login |
| обновление пароля пользователя | /user-service/users/password-reset-request |
| новый пароль пользователя | /user-service/users/password-reset |

Все остальные методы требуют аутентификации.

Сервис проверяет наличие и релевантность токена, а так же – какие действия разрешены конкретному пользователю(Пакет security)

# user-service

Часть микро сервиса, в которой собраны все операции с пользователем.

## Микро сервис содержит:

### База данных:

**users**

### Список Таблиц:

|  |  |  |
| --- | --- | --- |
| **Название таблицы** | **Что хранится** | **Примечания** |
| users | Хранится список пользователей | [Когда пользователь регистрируется, его данные добавляются в таблицу](#_Описание_и_логика) |
| password-reset-token | Хранятся токены, необходимые для изменения пароля | [Токены отправляются пользователю в письме, когда он хочет сменить свой пароль](#_Смена_пароля:) |

### API:

http://.../users-service/users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **API** | **метод** | **описание** | **авторизация** | **JSON** |
| / | POST | [Метод для регистрации нового пользователя](#_Регистрация_нового_пользователя:) | Не требуется | {  "firstName":"User",  "lastName":"Khr",  "email":"nu@mail.co”,  "password":"123",  "lang":"RUS",  "myCountry":"ISRAEL"  } |
| /login | POST | Метод входа в приложение |  | {  "firstName":"User",  "password":"123"  } |
| /create-admin | GET | [Метод, необходим для scheduler-service](#_Создание_супер_администратора:) | Не требуется | Данные доступны в свойствах аппликации. |
| /delete-admin/{adminId} | DELETE | Метод, который удаляет супер администратора | Не требуется |  |
| /{id} | GET | Получение данных пользователя | Требуется |  |
| /{id} | PUT | [Обновление данных пользователя](#_Обновление_своих_данных:) | Требуется | {  "firstName":"User",  "lastName": "Khr",  "lang":"RUS",  "myCountry":"ISRAEL"  } |
| /{id} | DELETE | [Удаление пользователя](#_Удаление_учетных_данных) | Требуется |  |
| / | GET | [Получение данных всех пользователей системы](#_Получение_всех_пользователей:) | Требуется |  |
| /email-verification | GET | Метод для подтверждения электронной почты. Вызывается, когда пользователь нажимает на ссылку в письме подтверждения своего электронного адреса | Требуется |  |
| /password-reset-request | POST | [Метод для смены пароля. Первый шаг – пользователь вводит свой электронный адрес, и получает на него письмо с токеном для изменения пароля](#_Смена_пароля:) | Не требуется | {  "email":"nu@mail.co”,  } |
| /password-reset | POST | [Второй шаг метода «обновить пароль»](#_Смена_пароля:) | Не требуется | {  "password":"123",  "token": "пеУ№пвапл0"  } |
| **Hystrix – методы, отправляют запрос в send-email-service** | | | | |
| /verification-email | POST | Запрос на оправку письма для подтверждения почты | | |
| /password-reset-request | POST | Запрос на отправку письма для смены пароля | | |
| /clean-not-confirmed-mail | POST | Запрос на отправку письма о том, что база данных отчищена от пользователей с неподтвержденным e-mail | | |
| /error-mail | POST | Запрос на отправку письма об ошибке | | |

### **Описание и логика методов:**

#### **Регистрация нового пользователя:**

При регистрации новый пользователь вводит необходимые данные:

* имя (firstName),
* фамилия(lastName),
* электронный адрес(email),
* пароль(password),
* язык интерфейса(lang)
* страну проживания (myCountry)

язык интерфейса – это язык, на котором будет отображаться интерфейс приложения

страна проживания нужна для того, чтобы в дальнейшем, когда будет реализована функция обмена валюты пользователь сможет «поменять» свою валюту исходя из официального курса той страны, которая указана в пункте «страна проживания».

Пользователю на выбор доступны языки:

* RUS (по умолчанию)
* ENG
* HEB

В бесплатной версии будет возможность выбрать страну:

* ISRAEL (по умолчанию)
* UKRAINE

Но в платной версии можно связаться с администратором и попросить о добавлении необходимой пользователю страны.

Электронный адрес должен быть релевантным. После регистрации на него будет отправлено письмо для подтверждения и окончания регистрации. Пользователи с неподтвержденным e-mail будут считаться незарегистрированными.

Раз в неделю (в воскресенье в 00:00 «чистильщик» и удалять из базы данных всех пользователей, у которых не подтверждена регистрация.

После того, как пользователь введет свои данные и нажмет кнопку «зарегистрироваться» его данные будут добавлены в базу данных, а на почту придёт письмо для подтверждения регистрации. В письме будет ссылка, на которую пользователь должен нажать и подтвердить регистрацию.

После того как пользователь подтвердит регистрацию, он сможет зайти в приложение под своим логином, используя пароль.

#### **Обновление своих данных:**

Пользователь может обновить свои данные. А именно:

* имя (firstName),
* фамилия(lastName),
* язык интерфейса(lang)
* страну проживания (myCountry)

#### **Удаление учетных данных**

#### **Получение всех пользователей:**

Администратор может посмотреть список всех пользователей, зарегистрированных в приложении вместе с их учетными данными.

#### **Смена пароля:**

Если пользователь забыл пароль – он может нажать на кнопку «забыл пароль». В этом случае на его электронную почту придет письмо с временным токеном и ссылкой для изменения пароля.

Пользователь может пройти по ссылке из письма и изменить свой пароль в базе данных.

#### **Добавление члена семьи:**

Когда пользователь регистрируется в приложении он может выбрать, хочет ли он быть «независимым пользователем» или хочет присоединиться к семье.

Если пользователь хочет присоединиться к семье, он должен в списке пользователей выбрать того пользователя, в чью семью он хочет присоединиться.

Пользователю «хозяину семьи» будет отправлено письмо с вопросом – действительно ли этот пользователь является «членом семьи». Если да, и «хозяин семьи» готов открыть свои счета для пользователя, пославшего запрос – он должен просто перейти по указанной в письме ссылке.

Если «хозяин семьи» подтвердил «родство», то «родственник» сможет получать доступ ко всем счетам и операциям, которые открыты для данной семьи.

### Дополнительные методы:

##### Создание суперадминистратора:

Метод, который вызывается при каждом запуске задач из scheduler-service. Перед тем, как начинать выполнять задачу, сервис создает суперадминистратора с введенными заранее данными, и таким образом получает пароль и идентификатор администратора, необходимые для того, что бы создать релевантный токен.

Перед тем, как создавать суперадминистратора, метод проверят – если такой пользователь есть в базе данных, то новый не создается.

### **Структура таблиц:**

#### users

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT | + | + | + |  |
| email | VARCHAR(120) |  | + |  |  |
| email\_verification\_status | BIT(1) |  | + |  |  |
| email\_verification\_token | VARCHAR(255) |  |  |  | NULL |
| encrypted\_password | VARCHAR(255) |  | + |  |  |
| first\_name | VARCHAR(50) |  | + |  |  |
| last\_name | VARCHAR(50) |  | + |  |  |
| user\_id | VARCHAR(255) |  | + |  |  |
| lang | VARCHAR(255) |  | + |  |  |
| my\_country | VARCHAR(255) |  |  |  | NULL |

#### password-reset-token

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT | + | + | + |  |
| token | VARCHAR(255) |  |  |  | NULL |
| user\_id | BIT(1) |  | + |  | NULL |

### **Application properties:**

|  |  |
| --- | --- |
| Свойство | Описание |
| server.port |  |
| spring.application.name |  |

# bills-service

### База данных:

Bills

### Список Таблиц:

|  |  |
| --- | --- |
| **Название таблицы** | **Что хранится** |
| [bill](#_bill) | Хранится данные о счетах пользователя |
| [category](#_category) | Хранится данные о категориях расходов/доходов пользователя |
| [operation](#_operation) | Хранится данные об операциях пользователя |
| [subcategory](#_subcatogity) | Хранится данные о субкатегориях дохода/расхода пользователя |

### **Структура таблиц:**

#### bill

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT(20) | + | + | + |  |
| create\_date | DATETIME(6) |  |  |  | NULL |
| deleted | BIT(1) |  |  |  | NULL |
| user\_uuid | VARCHAR(255) |  |  |  | NULL |
| bill\_name | VARCHAR(255) |  |  |  | NULL |
| bill\_uuid | VARCHAR(50) |  |  |  | NULL |
| description | VARCHAR(255) |  |  |  | NULL |
| start\_sum | DOUBLE |  |  |  | NULL |
| bill-number | INT(11) |  |  |  |  |

#### category

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT(20) | + | + | + |  |
| create\_date | DATETIME(6) |  |  |  | NULL |
| deleted | BIT(1) |  |  |  | NULL |
| user\_uuid | VARCHAR(255) |  |  |  | NULL |
| category\_name | VARCHAR(255) |  |  |  | NULL |
| category\_uuid | VARCHAR(50) |  |  |  | NULL |
| description | VARCHAR(255) |  |  |  | NULL |
| types | VARCHAR(255) |  |  |  | NULL |

#### operation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT(20) | + | + | + |  |
| create\_date | DATETIME(6) |  |  |  | NULL |
| deleted | BIT(1) |  |  |  | NULL |
| user\_uuid | VARCHAR(255) |  |  |  | NULL |
| bill\_uuid | VARCHAR(255) |  |  |  | NULL |
| category\_uuid | VARCHAR(50) |  |  |  | NULL |
| description | VARCHAR(255) |  |  |  | NULL |
| currency | VARCHAR(255) |  |  |  | NULL |
| operation\_uuid | VARCHAR(255) |  |  |  | NULL |
| subcatogiry\_uuid | VARCHAR(255) |  |  |  | NULL |
| sum | DOUBLE |  |  |  | NULL |
| type | VARCHAR(255) |  |  |  | NULL |

#### subcatogity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT(20) | + | + | + |  |
| create\_date | DATETIME(6) |  |  |  | NULL |
| deleted | BIT(1) |  |  |  | NULL |
| user\_uuid | VARCHAR(255) |  |  |  | NULL |
| category\_uuid | VARCHAR(255) |  |  |  | NULL |
| description | VARCHAR(255) |  |  |  | NULL |
| subcategory\_name | VARCHAR(255) |  |  |  | NULL |
| subcatogiry\_uuid | VARCHAR(255) |  |  |  | NULL |
| sum | DOUBLE |  |  |  | NULL |
| type | VARCHAR(255) |  |  |  | NULL |

### API:

http://.../bill-service/bill

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **API** | **метод** | **описание** | **авторизация** | **JSON** |
|  | POST | Добавление нового счета | Требуется | {  "billNumber":1,  "startSum":0.0,  "userUuid":"tHwJcqM2jO33ccM3LkNWra8WqxrDx7",  "description":"Natasha`s`s bill of bank"  } |
| /allBills/{userUuid} | GET | Получение всех счетов конкретного пользователя | Требуется |  |
| /allBills/details/{userUuid} | GET | Получение подробных данных обо всех счетах конкретного пользователя | Требуется |  |
| /{userUuid}/{billUuid} | DELETE | Удаление пользователя | Требуется |  |
| /{userUuid}/{billUuid} | PUT | Обновление счета | Требуется | {  "billName":"bill B"  } |

http://.../bill-service/category

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **API** | **метод** | **описание** | **авторизация** | **JSON** |
|  | POST | Добавление категории | Требуется | {  "categoryName":"Category C",  "userUuid":"tHwJcqM2jO33ccM3LkNWra8WqxrDx7",  "type":"OUTCOME",  "decryption":""  } |
| /{userUuid}/{billUuid} | DELETE | Удаление категории | Требуется |  |

http://.../bill-service/operation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **API** | **метод** | **описание** | **авторизация** | **JSON** |
|  | POST | Добавление операцию | Требуется | { "userUuid":"tHwJcqM2jO33ccM3LkNWra8WqxrDx7",  "billUuid":"921cf1c8-5a51-4c4a-b686-0685ceaf6a4b",  "categoryUuid":"5634b81a-dc28-473b-a2f7-e840e1d2554e",  "sum":78.0,  "currency":"RUB",  "type":"OUTCOME"  } |
| /{operationUuid}/{userUuid}/{type} | DELETE | Удаление операции | Требуется |  |
| /{userUuid} | GET | Получение всех операций пользователя | Требуется |  |
| /{operationUuid}/{userUuid}/{type | PUT | Обновление операции | Требуется | {  "billUuid":"777921cf1c8-5a51-4c4a-b686-0685ceaf6a4b",  "categoryUuid":"0db78755-451c-4279-b2d1-1d63aec3619b",  "sum":7888888888888.0,  "currency":"RUB"  } |

http://.../bill-service/subcategory

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **API** | **метод** | **описание** | **авторизация** | **JSON** |
|  | POST | Добавление подкатегории | Требуется | {  "subCategoryName":"subcategory A",  "userUuid":"tHwJcqM2jO33ccM3LkNWra8WqxrDx7",  "category":{  "categoryName":"Category C",  "categoryUuid": "0db78755-451c-4279-b2d1-1d63aec3619b"  },  "type":"OUTCOME",  "decryption":""  } |
| /{uuid}/{userUuid} | DELETE | Удаление подкатегории | Требуется |  |

### **Application properties:**

|  |  |
| --- | --- |
| Свойство | Описание |
| server.port |  |
| spring.application.name |  |
| tokenSecret |  |
| appl-database | название базы данных |
| spring.datasource.url | Подключение к Mysql |

# currency-service

## Микро сервис содержит:

### База данных:

**currency**

### Список Таблиц:

|  |  |  |
| --- | --- | --- |
| **Название таблицы** | **Что хранится** | **Примечания** |
| [default\_currency](#_default_currency:) | хранится список доступных в аппликации валют | [используется при создании валюты пользователя](#_default_currency:) |
| [сurrency](#_currency:) | хранится вся валюта конкретного пользователя, которую он создает | [используется при создании счета и/или при создании операции](#_currency:) |
| [israel\_currency](#_israel_currency:) | хранятся данные относительно обменного курса валют | [Используется при обмене валют](#_israel_currency:) |
| [ukraine\_currency](#_ukraine_currency:) | хранятся данные относительно обменного курса валют | [Используется при обмене валют](#_ukraine_currency:) |

### API:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **API** | **метод** | **описание** | **авторизация** | **JSON** |
| currency-service/parse/all | GET | получение все валют, которые хранятся в таблице default\_currency | Требуется |  |
| currency-service/parse | GET | добавление валют в таблицу default\_currency | Требуется |  |
| currency-service/currency | POST | добавление валюты пользователя в таблицу сurrency | Требуется | {  "currencyName":"Rub", "userUuid":"11111",  "abbr":"RUB"  } |
| currency-service/ currency/{"user\_UUID”}/{abbr} | DELETE | удаляет конкретную валюту конкретного пользоватял из таблички сurrency | Требуется |  |
| currency-service/parse/israel | GET | парсит сайт Изаильского национального банка (внутренний метод для scheduler-service) | Требуется |  |
| currency-service/parse/ukraine | GET | парсит сайт Украинского национального юанка (внутренний метод для scheduler-service) | Требуется |  |
| currency-service/parse/ukraine/all | GET | получение всех курсов валют из таблицы ukraine\_currency | Требуется |  |
| currency-service/parse/israel/all | GET | GET получение всех курсов валют из таблицы israel\_currency | Требуется |  |
| currency-service/test |  | проверка работоспособности приложения (ping) | Не требуется |  |
| currency-service/actuator |  |  | Требуется |  |

### **Описание и логика методов:**

#### **default\_currency:**

Валюта попадает в таблицу default\_currency путем парсинга JSON файла.

Пример файла храниться в приложении currency-service : /Users/smilykq/Desktop/home\_ACC/home-acc-beta-2/currency-service/src/main/java/currencyservice/jsonFile/currency.json

Структура JSON файла:

{

"global\_id":62838463,

"system\_object\_id":"840",

"CODE":"840",

"STRCODE":"USD",

"NAME":"Доллар США",

"COUNTRY":"Соединенные Штаты Америки"

}

Парсинг JSON файла доступен в API : currency-service/parse

!!! требуется авторизация!!!

Имя и пусть JSON файла могут быть изменены в файле currency-service.peoperties (в приложении config-server-repo) в свойстве: currency.file

Перед тем, как добавить данные в таблицу метод удаляет те данные, которые были в ней раньше

#### **currency:**

Когда пользователь создает операцию или счет в приложении, он должен выбрать валюту этой операции или счета. Пользователь может выбрать только тот вид валюты, который храниться в таблице default\_currency.

#### **israel\_currency:**

Каждый день в 3:00 ночи scheduler-service создает задачу. Согласно этой задаче, вызывается API currency-service/parse/israel

Согласно этой задаче, приложение обращается по ссылке <https://www.boi.org.il/currency.xml>

 и добавляет в БД курс валют, согласно данным национального банка Израиля.

После проведения операции парсинга администратор получает письмо о том, что процедура прошла успешно

В случае ошибки – получает письмо об ошибке

url может быть изменен в файле currency-service.peoperties (в приложении config-server-repo) в свойстве: israel.currency.xml

Перед тем, как добавить данные в таблицу метод удаляет те данные, которые были в ней раньше!!!

#### **ukraine\_currency:**

Каждый день в 3:00 ночи scheduler-service создает задачу. Согласно этой задаче, вызывается API currency-service/parse/ukraine

Согласно этой задаче, приложение обращается по ссылке <https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange>

и добавляет в БД курс валют, согласно данным национального банка Украины.

url может быть изменен в файле currency-service.peoperties (в приложении config-server-repo) в свойстве: ukraine.currency.xml

Перед тем, как добавить данные в таблицу метод удаляет те данные, которые были в ней раньше!!!

После проведения операции парсинга администратор получает письмо о том, что процедура прошла успешно

В случае ошибки – получает письмо об ошибке

### **Структура таблиц:**

#### **default\_currency:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| Id | BIGINT | + | + | + |  |
| Country | VARCHAR(255) |  |  |  |  |
| name | VARCHAR(255) |  |  |  |  |

#### **currency:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT | + | + | + |  |
| currency\_name | VARCHAR(255) |  | + |  |  |
| abbr | VARCHAR(255) |  | + |  |  |
| deleted | BIT(1) |  |  |  | NULL |
| user\_uuid | VARCHAR(255) |  |  |  |  |

#### **israel\_currency:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT | + | + | + |  |
| name | VARCHAR(255) |  |  |  | NULL |
| abbr | VARCHAR(255) |  |  |  | NULL |
| country | VARCHAR(255) |  |  |  | NULL |
| quantity | DOUBLE |  |  |  | NULL |
| rate | DOUBLE |  |  |  | NULL |
| change\_course | DOUBLE |  |  |  | NULL |

#### **ukraine\_currency:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Column | Datatype | PK | NN | AI | Defoult |
| id | BIGINT | + | + | + |  |
| name | VARCHAR(255) |  |  |  | NULL |
| abbr | VARCHAR(255) |  |  |  | NULL |
| rate | DOUBLE |  |  |  | NULL |

### **Application properties:**

|  |  |
| --- | --- |
| Свойство | Описание |
| server.port |  |
| spring.application.name |  |
| tokenSecret |  |
| appl-database | название базы данных |
| spring.datasource.url | Подключение к Mysql |
| currency.file | Пусть к JSON файлу для парсинга валют для табл [default\_currency](#_default_currency:) |
| israel.currency.xml | Линк для парсинга Израильского Национального банка |
| ukraine.currency.xml | Линк для парсинга Украинского Национального банка |

# crawler-service

### База данных:

|  |  |  |
| --- | --- | --- |
| Hystrix – методы, отправляют запрос в другие сервисы | | |
| /parse/israel | GET | Запрос на парсинг сайта Нацбанка Израиля |
| /parse/ukraine | GET | Запрос на парсинг сайта Нацбанка Украины |
| /clean-data/clean-not-conf | GET | Запрос на чистку базы данных от пользователей с неподтвержденным e-mail |
| / users/create-admin | GET | Запрос на создание супер администратора |

# send-email-service

Часть микро сервиса, которая по запросу от разных микро сервисов отправляет письма пользователям и администратору.

## Микро сервис содержит:

API:

http://.../ send-email-service

|  |  |  |  |
| --- | --- | --- | --- |
| **API** | **метод** | **описание** | **описание** |
| /verification-email | POST | отправка письма при регистрации нового пользователя | Письмо отправляется на предоставленный пользователем электронный адрес. Содержит ссылку для подтверждения + токен |
| /change-password | POST | Отправка письма при изменении пароля | Письмо отправляется на предоставленный пользователем адрес.  Содержит ссылку для отправки письма с паролем и токен |
| /clean-not-confirmed-mail | POST | Отправка письма при отчистке базы данных | Письмо отправляется администратору.  Содержит информацию о том, что пользователи с неподтвержденным email были удалены из БД |
| /error-mail | POST | Отправка письма, уведомляющего об ошибках | Письмо отправляется администратору.  Содержит информацию о любой ошибке, которая произошла |
| /parse/israell | POST | Отправка письма при парсинге Израильского банка | Письмо содержит информацию о том, что валюты национального банка Израиля были обновлены |
| /parse/ukraine | POST | Отправка письма при парсинге Украинского банка | Письмо содержит информацию о том, что валюты национального банка Украины были обновлены |

### **Application properties:**

|  |  |
| --- | --- |
| Свойство | Описание |
| server.port |  |
| spring.application.name |  |
| verification.link | Линк, которой приходит в письме о подтверждении e-mail |
| change.password.link | Линк, которой приходит в письме о смене пароля |
| email.password | Пароль для e-mail с которого отправляются письма |
| email.address | e-mail с которого отправляются письма |
| admin.email.address | e-mail администратора |