# Summary

The Home Accounting Project is a means to oversee family accounting.:

- all family members who have access to the family budget have access to shared data on the financial resources that the family manipulates.

The user can store all the data in one place. You don't need to install multiple applications.

The following modules are available in the app:

1. [Planning costs](#_Планирование_расходов:)
2. [Income planning](#_Планирование_доходов:)
3. [Shopping list](#_Обмен_валюты:)
4. [Currency exchange](#_Список_продуктов:)

## Planning costs:

Closely related to the "shopping list" module. The user can choose the optimal list of goods that he needs to purchase, using the optimization of purchase costs according to the business logic of the total cost of travel, delivery of products.

In addition, the user can create a list of planned expenses and income for the month/year/week and see how much money he has left in the financial account. This will allow more productive spending and control of their movement.

Cost planning functions:

## Income planning:

The user can plan their basic income (such as salary, inheritance and other options for guaranteed appearance of funds in the user's account) this will help to more correctly plan expenses, understanding what arrival and expenditure of funds awaits the user in the current period of time.

## Currency exchange:

Control of the user's funds in various world currencies.

Implementation:

The user can choose the currency he wants to use when filling in expenses - income. The choice of currency is possible from the list of "default" currencies in the application.

The list of defaulted currencies is uploaded to the app from the Json file (currencycurrency. json is at the root of the project). Once the user has chosen the main currency, for each particular transaction he can transfer the main currency to any other.

To provide relevant currency exchange data, every night at 3:00 a.m. the app receives up-to-date data from the websites of the national banks of selected currencies (in the free version are available currencies:

* Israeli Shekel
* Us dollar
* Ukrainian hryvnia

In the paid version, the user can contact the administrator and "order" other types of currencies, if he needs them)

Thus, the user can change the currency of each particular transaction at any time, and the data will be relevant.

## Product list:

Parser in conjunction with the list of products allows you to choose the best solution for where, todand for what means you need to buy a particularproduct.

### Shopping list module features:

1. Make a shopping list
2. Keep previously created lists
3. Accounting for purchase costs
4. The ability to make notes about a particular product
5. The ability to choose the most convenient store (depending on the availability of the goods from the shopping list, as well as the price of the goods in a particular store)
6. Compare the availability of goods and its cost in different stores
7. Plan the menu for a specific period, and based on the menu - make a list of products (not implemented)
8. Store a "cooking book" in which you can collect the necessary recipes along with a list of foods needed to prepare a dish. (not implemented)

### Implementation:

In order to guarantee the user relevant data about the availability of products in a particular store or about the cost of the product, every night the parser goes through the Internet sites of major retail chains and collects information about the product, its cost and availability in the store.

# API DOCUMENTATION

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

# The services that make up theapp.

|  |  |  |
| --- | --- | --- |
| The name of the micro  Service | Description of the microservice | Port |
| [config-service](#_config-service) | Configuration service. ссылке Keeps a link to git -properties микро сервисов -hub. |  |
| eureka-service | Eureka service. Each Micro service is registered on the Eureka server, and the Eureka server knows all the client applications running at each port and IP address. |  |
| [zuul-service](#_zuul-service) | Agateway that handles all requests and performs dynamic routing of microservice applications  The same app authorizes the user and checks if they have permissions to perform certain actions |  |
| [user-service](#_user-service) | A service that stores information about the user, the presence of his relatives registered in the application,as well as provides the ability to register theuser, confirm his email and change the password, in case 6 if the user forgot it. + CRUD |  |
| [bills-service](#_bills-service) | A service that stores all information aboutthe user's accounts and everything related to these accounts. |  |
| [currency-service](#_currency-service) | A service that stores information about currencies and their rate. |  |
| [crawler-service](#_crawler-service) | The service that deals with - национальные банки, что бы иметь актуальную информацию о курсе валют -магазины что бы иметь актуальную информацию о наличии товара и его стоимости parsing.. Он парсит: |  |
| send-email-service | A service that sends emails |  |

# How to download

1. Config-service
2. Eureka-service (takesdata to download from config-service))
3. Zuul-service
4. Other services

# config-service

keeps a link to the git repository, which stores application-properites for all services[(https://github.com/smilyk/home-accounting-2-repo)](https://github.com/smilyk/home-accounting-2-repo))

# zuul-service

## Security

|  |  |
| --- | --- |
| Confirmation of e-mail | /user-service/users/send-email-service/\*\* |
| User registration | /user-service/users |
| Login | /user-service/users/login |
| updating your user's password | /user-service/users/password-reset-request |
| new user password | /user-service/users/password-reset |

All other methods require authentication.

The service checks the presence and relevance of the token,as well as - what actions are allowed to a particular user (Security пользователю(Package) )

# user-service

Part of the micro servicethatcollects all user transactions.

## The micro service contains:

### Database:

**users**

### Table list:

|  |  |  |
| --- | --- | --- |
| **Table name** | **What's stored** | **Notes** |
| users | User list kept | [When a user registers, their data is added to the table](#_Описание_и_логика) |
| password-reset-token | Stores needed to change your password | [Tokens are sent to the user in a letter when he wants to change his password](#_Смена_пароля:) |

### API:

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Api** | **Method** | **Description** | **Authorization** | **Json** |
| / | POST | [A method for registering a new user](#_Регистрация_нового_пользователя:) | Not required | {  "firstName":"User",  "lastName":"Khr",  "email":"nu@mail. co",  "password":"123",  "lang":"RUS",  "myCountry":"ISRAEL"  } |
| /login | POST | How to log in to the app |  | {  "firstName":"User",  "password":"123"  } |
| /create-admin | Get | [Method needed for scheduler-service](#_Создание_супер_администратора:) | Not required | The data is available in applique properties. |
| /delete-admin/{adminId} | DELETE | A method that removes a super admin | Not required |  |
| /{id} | Get | Getting user data | Required |  |
| /{id} | Put | [Update user data](#_Обновление_своих_данных:) | Required | {  "firstName":"User",  "lastName": "Khr",  "lang":"RUS",  "myCountry":"ISRAEL"  } |
| /{id} | DELETE | [Removing the user](#_Удаление_учетных_данных) | Required |  |
| / | Get | [Get data from all users of the system](#_Получение_всех_пользователей:) | Required |  |
| /email-verification | Get | A method for confirming email. Called when a user clicks on a link in a confirmation letter to their email address | Required |  |
| /password-reset-request | POST | [A method for changing your password. The first step is that the user enters his email address and receives an email with a token to change the password](#_Смена_пароля:) | Not required | {  "email":"nu@mail.co",  } |
| /password-reset | POST | [The second step of the "update password" method](#_Смена_пароля:) | Not required | {  "password":"123",  "token":"PeU Pvapl0""  } |
| **Hystrix - methods, send a request to**  **send-email-service** | | | | |
| /verification-email | POST | Request for letter confirmation | | |
| /password-reset-request | POST | Request to send a letter to change your password | | |
| /clean-not-confirmed-mail | POST | Request to send a letter that the database is cleared of users with unconfirmed ee-mailmail | | |
| /error-mail | POST | Request to send an error letter | | |

### **Description and logic of methods:**

#### **Register a newuser:**

When you sign up, the new user enters the data they need:

* Name(firstName),),
* surname(lastName),),
* emailedemailaddress
* password(password)),
* interface language(lang))
* country of residence(myCountry))

Interface language is the language in which the app interface will appear

the country of residence is necessary in order that in the future, when the function of currency exchange will be implemented, the user will be able to "change" their currency based on the official rate of the country, which is specified in the "country of residence" paragraph.

Languages are available to the user:

* RUS(default))
* Eng
* Heb

The free version will be able to choose the country:

* ISRAEL (default))
* UKRAINE

But in the paid version, you can contact the administrator and ask for the addition of the country user needed.

The email address should be relevant. Once you sign up, a letter will be sent to confirm and end registration. Users with unconfirmed ee-mailmail will be considered unregistered.

Once a week (Sunday at 00:00 "cleaner" and remove from the database all users who have not confirmed registration.

Once the user enters their data and clicks the "register" button, their data will be added to the database, and a letter will arrive in the mail to confirm registration. The letter will have a link to which the user must click and confirm the registration.

Once the user confirms registration, they will be able to enter the app under their login using the password.

#### **Update yourdata:**

The user can update their data. Namely:

* Name(firstName),),
* surname(lastName),),
* interface language(lang))
* country of residence(myCountry))

#### **Delete credentials**

#### **Getting all users:**

The administrator can see a list of all users registered in the app along with their credentials.

#### **Changing password:**

If a user forgets the password, they can click on the "forgot password" button. In this case, an email with a temporary token and a link to change the password will come to his email.

The user can follow the link from the letter and change their password in the database.

#### **Adding a family member:**

When a user registers in an app, they can choose whether they want to be an "independent user" or want to join the family.

If a user wants to join a family, they must choose the person in the user list who they want to join.

A "family owner" user will be sent an email asking if the user is really a "family member." If so, and the "family owner" is ready to open his accounts to the user who sent the request - he should simply click on the link specified in the letter.

If the "family owner" has confirmed "kinship," the "relative" will be able to access all accounts and transactions that are open to the family.

### Additional methods:

##### Creating a super administrator::

The method that is called every time you run tasks from schedulerto суперадминистратора для того, что бы service.

Before creating a super administrator,the method will check - if such a user is in the database, the new one is not created.

### **Table structure:**

#### 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:**

|  |  |
| --- | --- |
| Property | Description |
| server.port |  |
| spring.application.name |  |

# bills-service

### Database:

Bills

### Table list:

|  |  |
| --- | --- |
| **Table name** | **What's stored** |
| [bill](#_bill) | User accounts are stored |
| [category](#_category) | Data on user expense/revenue categories are stored |
| [operation](#_operation) | User's operations data stored |
| [subcategory](#_subcatogity) | Data on user income/expense subcategories are stored |

### **Table structure:**

#### 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** | **Method** | **Description** | **Authorization** | **Json** |
|  | POST | Adding a new account | Requiresa sy | {  "billNumber":1,  "startSum":0.0,  "userUuid":"tHwJcqM2jO33ccM3LkNWra8WqxrDx7",  "description":"Natasha's's bill of bank"  } |
| /allBills/{userUuid} | Get | Getting all of a user's accounts | Requiresa sy |  |
| /allBills/details/{userUuid} | Get | Get detailed information about all user accounts | Requiresa sy |  |
| /{userUuid}/{billUuid} | DELETE | Removing the user | Requiresa sy |  |
| /{userUuid}/{billUuid} | Put | Account update | Requiresa sy | {  "billName":"bill B"  } |

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Api** | **Method** | **Description** | **Authorization** | **Json** |
|  | POST | Adding category | Requiresa sy | {  "categoryName":"Category C",  "userUuid":"tHwJcqM2jO33ccM3LkNWra8WqxrDx7",  "type":"OUTCOME",  "decryption":""  } |
| /{userUuid}/{billUuid} | DELETE | Removing the category | Requiresa sy |  |

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Api** | **Method** | **Description** | **Authorization** | **Json** |
|  | POST | Adding surgery | Requiresa sy | { "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 | Removing surgery | Requiresa sy |  |
| /{userUuid} | Get | Get all the user's operations | Requiresa sy |  |
| /{operationUuid}/{userUuid}/{type | Put | Update the operation | Requiresa sy | {  "billUuid":"777921cf1c8-5a51-4c4a-b686-0685ceaf6a4b",  "categoryUuid":"0db78755-451c-4279-b2d1-1d63aec3619b",  "sum":7888888888888.0,  "currency":"RUB"  } |

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Api** | **Method** | **Description** | **Authorization** | **Json** |
|  | POST | Adding subcategory | Requiresa sy | {  "subCategoryName":"subcategory A",  "userUuid":"tHwJcqM2jO33ccM3LkNWra8WqxrDx7",  "category":{  "categoryName":"Category C",  "categoryUuid": "0db78755-451c-4279-b2d1-1d63aec3619b"  },  "type":"OUTCOME",  "decryption":""  } |
| /{uuid}/{userUuid} | DELETE | Removal of subcategory | Requiresa sy |  |

### **Application properties:**

|  |  |
| --- | --- |
| Property | Description |
| server.port |  |
| spring.application.name |  |
| tokenSecret |  |
| appl-database | database name |
| spring.datasource.url | Connect to Mysql |

# currency-service

## The micro service contains:

### Database:

**currency**

### Table list:

|  |  |  |
| --- | --- | --- |
| **Table name** | **What's stored** | **Notes** |
| [default\_currency](#_default_currency:) | stored list of available in currency applications | [used to create a user's currency](#_default_currency:) |
| [surrency](#_currency:) | stored the entire currency of the particular user he creates | [used to create an account and/or create an operation](#_currency:) |
| [israel\_currency](#_israel_currency:) | stored data on the exchange rate of currencies | [Used in currency exchange](#_israel_currency:) |
| [ukraine\_currency](#_ukraine_currency:) | stored data on the exchange rate of currencies | [Used in currency exchange](#_ukraine_currency:) |

### Api:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Api** | **Method** | **Description** | **Authorization** | **Json** |
| currency-service/parse/all | Get | Receiving\_currency all the currenciesthat are stored in the default table | Requiresa sy |  |
| currency-service/parse | Get | adding\_currency currencies to the default table | Requiresa sy |  |
| currency-service/currency | POST | adding a user's currency to a table withurrency | Requiresa sy | {  "currencyName":"Rub", "userUuid":"11111",  "abbr":"RUB"  } |
| currency-service/ currency/{"user\_UUID"}/{abbr} | DELETE | removes a particular currency of a particular used from the sign with a | Requiresa sy |  |
| currency-service/parse/israel | Get | The isail national бb-anka website (internal method for scheduler-service) | Requiresa sy |  |
| currency-service/parse/ukraine | Get | The website of the Ukrainian National Woman (internal method for scheduler-service) | Requiresa sy |  |
| currency-service/parse/ukraine/all | Get | receiving all exchange rates from the table ukraine\_currency | Requiresa sy |  |
| currency-service/parse/israel/all | Get | GET getting all exchange rates from the table israel\_currency | Requiresa sy |  |
| currency-service/test |  | Checking the health of the app (ping) | Not required |  |
| currency-service/actuator |  |  | Requiresa sy |  |

### **Description and logic of methods:**

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

The currency hits the table default\_currency by parsing the JSON file.

Sample file stored in -service currency-service app : /Users/smilykq/Desktop/home\_ACC/home-acc-beta-2/currency-service/src/main/java/currencyservice/jsonFile/currency.json

JSON filestructure:

{

"global\_id":62838463,

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

"CODE":"840",

"STRCODE":"USD",

NAME"NAME":"U.S. Dollar"

COUNTRY"COUNTRY": "United States of America"

}

JSON file parsing is available in API : currency-service/parse/parse

!!! authorization required!!!

The name and let JSON file can be changed in the currency-service.peoperties file (in the config-server-repo app) in the property: currency.file

Before adding data to the table, the method removes the data that was in it before

#### **currency:**

When a user creates an operation or account in an app, they must choose the currency of that transaction or account. The user can only choose the kind of currency that is stored in the table default\_currency..

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

Every day at 3:00 a.m. scheduler-service creates a task. According to this task, the API is called currency-service/parse/israel

According to this task, the app clicks on the [link https://www.boi.org.il/currency.xml](https://www.boi.org.il/currency.xml)

and adds a currency rate to the database, according to the National Bank of Israel..

After the parsing operation, the administrator receives a letter stating that the procedure was successful

In case of error, receives an error letter

url can be changed in the currency-service.peoperties file (in the config-server-repo app) in the property: israel.currency.xml

Before adding data to the table, the method removes the data that was in it before!!!

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

Every day at 3:00 a.m. scheduler-service creates a task. According to this task, the API isukraine called currency-service/parse/ukraine

According to this task, the app clicks on the [link https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange](https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange)

and adds the exchange rate to the database, according to the National Bank of Ukraine..

url can be changed in the currency-service.peoperties file (in the config-server-repo app) in the property: ukraine.currency.xml

Before adding data to the table, the method removes the data that was in it before!!!

After the parsing operation, the administrator receives a letter stating that the procedure was successful

In case of error, receives an error letter

### **Table structure:**

#### **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:**

|  |  |
| --- | --- |
| Property | Description |
| server.port |  |
| spring.application.name |  |
| tokenSecret |  |
| appl-database | database name |
| spring.datasource.url | Connect to Mysql |
| currency.file | Let to JSON filey for parsing currencies for table [default\_currency](#_default_currency:) |
| israel.currency.xml | Link for parsing of the Israeli National Bank |
| ukraine.currency.xml | Link for parsing of the Ukrainian National Bank |

# crawler-service

### Database:

|  |  |  |
| --- | --- | --- |
| Hystrix - methods that send a request to other services | | |
| /parse/israel | Get | Request for parsing the website of the National Bank of Israel |
| /parse/ukraine | Get | Request for parsing the website of the National Bank of Ukraine |
| /clean-data/clean-not-conf | Get | Request to clean the database from users with unconfirmed e-mail |
| / users/create-admin | Get | Request to create a super administrator |

# send-email-service

Part of the micro servicethatsends letters to users and administrators on request from different micro services.

## The micro servicecontains:

API:

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Api** | **Method** | **Description** | **Description** |
| /verification-email | POST | Send a letter when a new user registers | The letter is sent to the email address provided by the user. |
| /change-password | POST | Send a letter when you change your password | The letter is sent to the address provided by the user. Contains a link to send a letter with a password and a token |
| /clean-not-confirmed-mail | POST | Send a letter while clearing the database | The letter is sent to the administrator.  Contains information that users with unconfirmed email have been removed from the database |
| /error-mail | POST | Sending a letter notifying you of errors | The letter is sent to the administrator.  Contains information about any error that occurred |
| /parse/israell | POST | Sending a letter at the parsing of the Israeli bank | The letter contains information that the currencies of Israel'snationalb-anka have been updated |
| /parse/ukraine | POST | Sending a letter at the parsing of the Ukrainian bank | The letter contains information that the currencies of the nationalb-an of Ukraine have been updated б |

### **Application properties:**

|  |  |
| --- | --- |
| Property | Description |
| server.port |  |
| spring.application.name |  |
| verification.link | Link, which comes in a letter of confirmation e-mailmail e |
| change.password.link | Link, which comes in a letter about changing the password |
| email.password | Password for e--mail from which letters are sent |
| email.address | e-mail from which letters are sent |
| admin.email.address | e- admin mail |