**Тестовое задание для собеседования в компанию:**

**ООО Смартсофт**

**Задача:**

Сделать конвертер валют

**Исполнитель:**

Фролов Сергей Анатольевич

1. **Описание задачи:**

При запуске приложения, необходимо получить список актуальных валют и их курсов с сайта ЦБРФ <http://www.cbr.ru/scripts/XML_daily.asp> (дополнительная информация <https://cbr.ru/development/sxml/>) и записать их в базу данных (индентификаторы, коды, названия), а так же курсы (привязанные к валюте) на дату запроса. В конвертере должна быть авторизация по логину и паролю. Пользователь пройдя авторизацию попадает на главный экран, где может выбрать из какой валюты и в какую будет конвертация, указывает количество переводимых средств и нажимает кнопку "Конвертировать". После чего происходит запрос в БД на получение актуального курса на \*\*\*текущую дату \*\*\*, если данные на текущую дату отсутствуют, необходимо, произвести получение курсов с сайта ЦБ и добавить новые записи в базу данных. Также в конвертере должна вестись история произведенных конвертаций с записью в базу данных со ссылкой на курс по которой была произведена конвертация. Историю можно посмотреть на той же странице конвертера или отдельной вкладке (возможна реализация базовых фильтров). Остальная функциональность и визуал по желанию.

1. **Использованные технологии:**

* Java 11
* Maven
* Spring boot 2
* PostgresSQL 12
* Html 5\CS, Thymeleaf

1. **Описание функционала и интерфейса программы:**

**Общие сведения:**

Хранение информации осуществляется в базе данных (далее БД). Используемые таблицы: таблица пользователей, таблица ролей, таблица связи ролей и пользователей, таблица истории операций, таблица доступных валют, таблица курсов валют. Подробный скрипт БД представлен в Приложении 1.

Программа имеет панель навигации в верхней части страницы, вкладки подсвечиваются в зависимости от пункта меню, в котором пользователь находится на текущий момент.

Предусмотрена двуязычность программы.

**Первый старт программы:**

Во время первого запуска проверяется наличные ролей в БД, если роли отсутствуют, это значит что в БД нет данных о пользователях. В момент первого запуска проводится добавление ролей USER и ADMIN в БД и добавляется главный администратор с именем admin и паролем 100

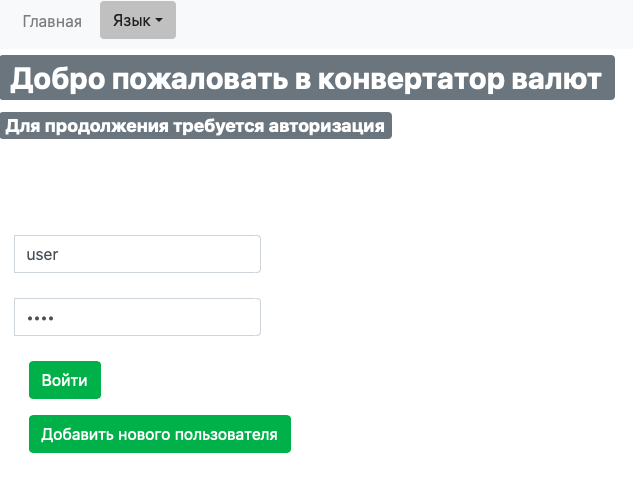
USER – обычный пользователь

ADMIN – пользователь с расширенным функционалом

При каждом старте программы проверяется наличие актуального списка валют и курса в БД, в случае отсутствия данных на текущую дату, запрос отправляется на сайт Центрального Банка РФ и добавляется в БД

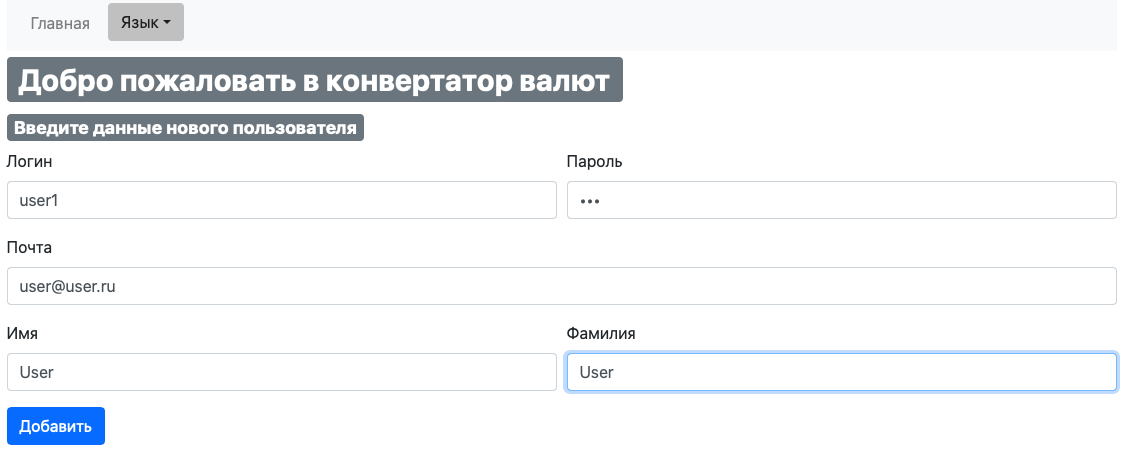
**Авторизация:**

Для входа в систему пользователю требуется пройти авторизацию, где вводится логин и пароль. При неверном вводе логина и/или пароля предусмотрено сообщение об ошибке.



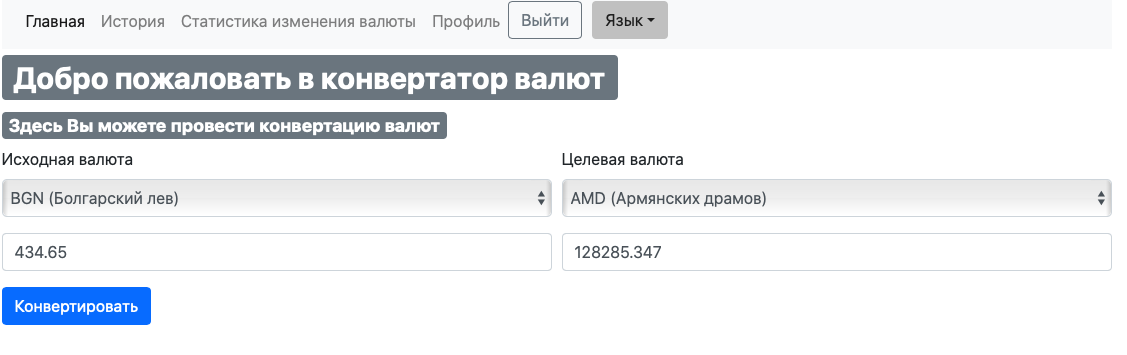
**Добавление нового пользователя:**

Для использования программы новым пользователям предлагается добавить свои данные для регистрации в системе. Требуемые данные: логин, пароль, имя, фамилия и адрес электронной почты. Все поля обязательны для заполнения. Запрещается повторное использование логина пользователя. У каждого пользователя должен быть уникальный логин, если в БД уже существует пользователь с таким логином, то выдается сообщение об ошибке.



**Конвертация валюты (доступна только для авторизованных пользователей с ролью USER и ADMIN):**

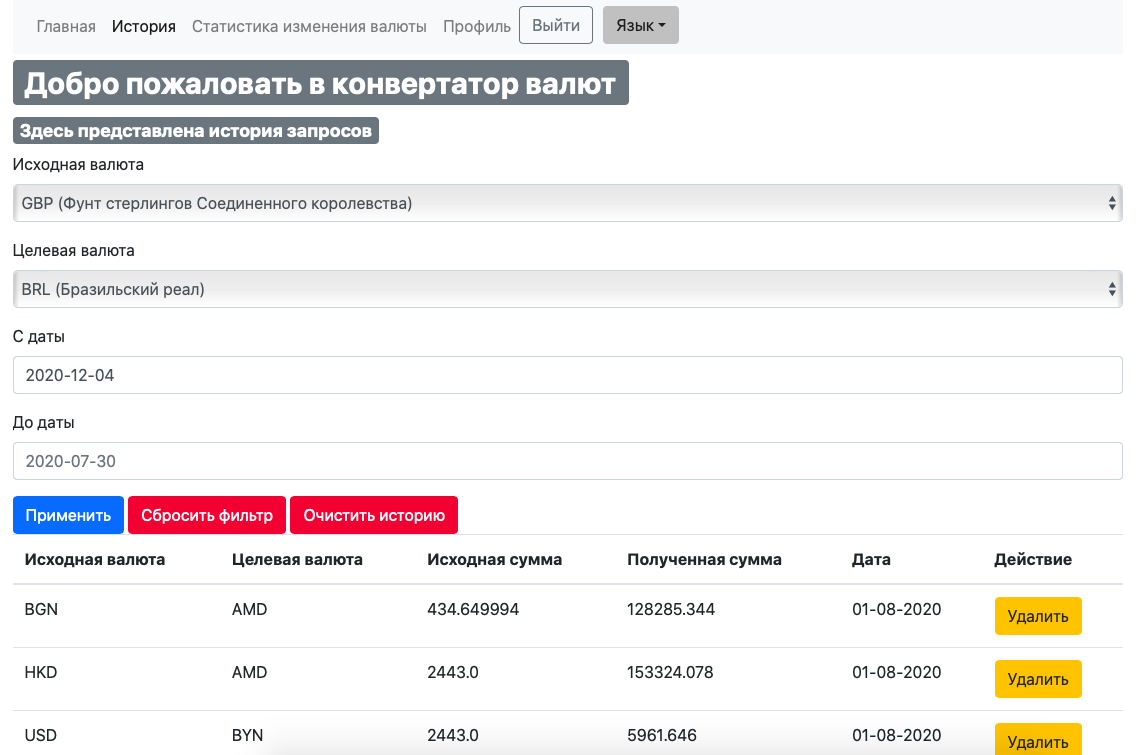
Конвертация валют производится при выборе двух валют (исходной и запрашиваемой) и введенном значении исходной валюты. Проверяется корректность введенных даных: не допускается ввода никаких значений кроме цифр.



**История выполненных операций (доступна только для авторизованных пользователей с ролью USER и ADMIN):**

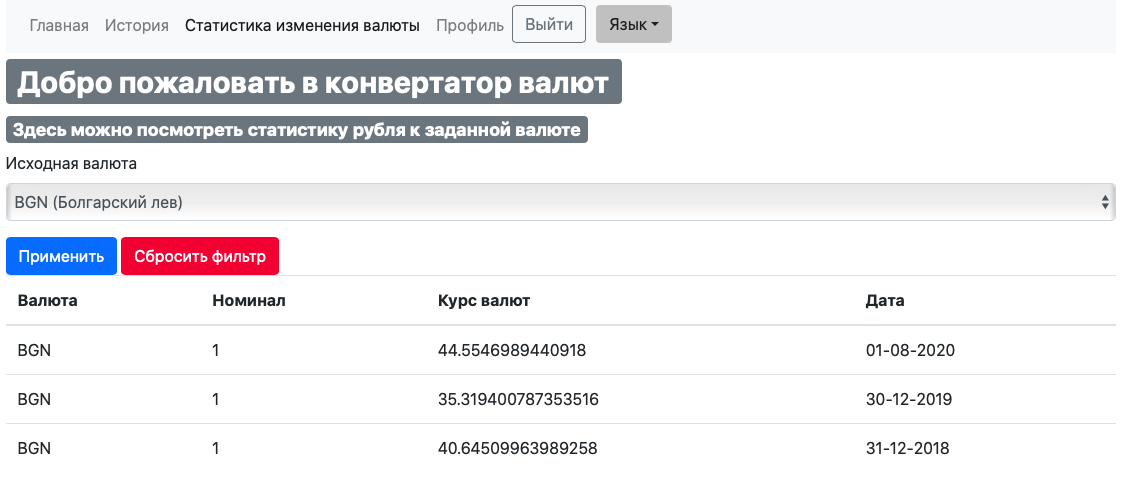
Вся история операций сохраняется в БД для каждого пользователя. Реализованы базовые фильтры: по исходной валюте, по целевой валюте и по интервалу дат.

Реализованы кнопки сброса фильтров, очистки истории операций для текущего пользователя и удаления выбранной операции отдельно.



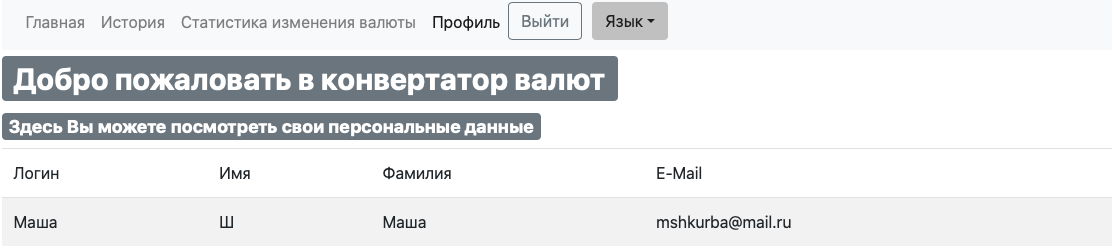
**Статистика изменения валюты (доступна только для авторизованных пользователей с ролью USER и ADMIN):**

Реализована функция отображения курса выбранной валюты относительно рубля. Требуется ввод исходной валюты. Реализована функция сброса выбранного фильтра.



**Профиль (доступна только для авторизованных пользователей с ролью USER и ADMIN):**

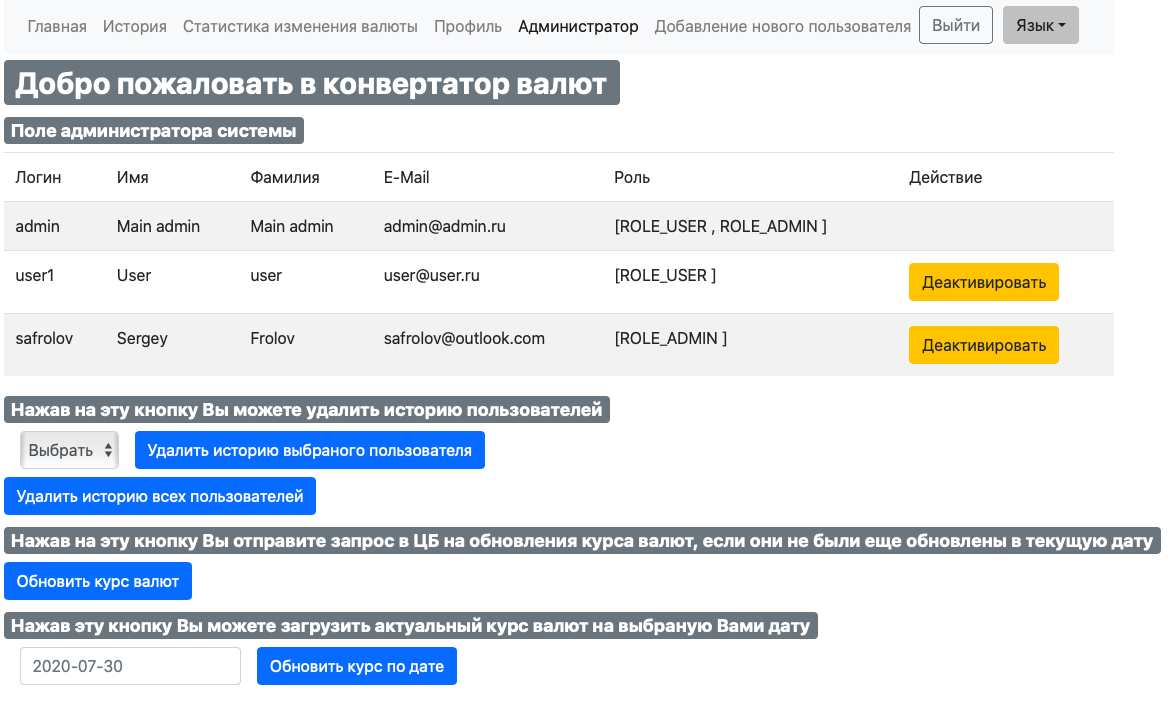
В данном разделе представляется информация о пользователе (логин, имя, фамилия и электронная почта).



**Вкладка администратора (доступна только для авторизованных пользователей с ролью ADMIN):**

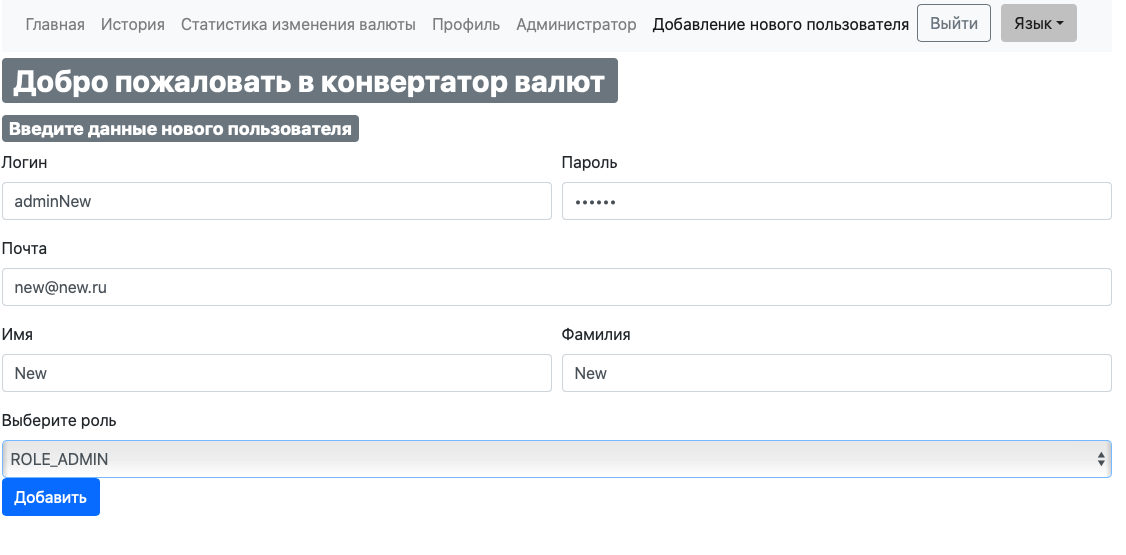
Реализованные функции администратора:

1. Управление пользователями. Активация/деактивация пользователей. Деактивированный пользователь не может войти в систему. Main Admin (создается при первом запуске системы) не может быть деактивирован.
2. Очистка истории пользователей. Возможен выбор определённого пользователя или очистка истории всех пользователей.
3. Отправка вынужденного запроса на обновление курса валют на текущую дату.
4. Отправка запроса на обновление курса валют с указанием требуемой даты. Предусмотрена выдача ошибок в случае некорректно введённой даты.



**Добавление нового пользователя (доступна только для авторизованных пользователей с ролью ADMIN):**

Реализована дополнительная функциональность добавления нового пользователя от имени администратора. В данном разделе присутствуют такие же поля, как при добавление нового пользователя. Дополнительно можно выбирать роль добавляемого пользователя (USER или ADMIN).



**Приложение 1. Скрипт используемой Базы данных.**

/\*  
Script generated by Aqua Data Studio 20.6.0-4 on Aug-02-2020 12:10:01 AM  
Database: postgres  
Schema: conversion  
Objects: TABLE  
\*/  
ALTER TABLE "conversion"."сurrency\_value"  
 DROP CONSTRAINT "constraint\_name1" CASCADE   
GO  
ALTER TABLE "conversion"."users\_roles"  
 DROP CONSTRAINT "users" CASCADE   
GO  
ALTER TABLE "conversion"."users\_roles"  
 DROP CONSTRAINT "roles\_fk" CASCADE   
GO  
ALTER TABLE "conversion"."history"  
 DROP CONSTRAINT "user\_id\_fk" CASCADE   
GO  
ALTER TABLE "conversion"."history"  
 DROP CONSTRAINT "constraint\_name2" CASCADE   
GO  
ALTER TABLE "conversion"."history"  
 DROP CONSTRAINT "constraint\_name1" CASCADE   
GO  
DROP TABLE "conversion"."сurrency\_value"  
GO  
DROP TABLE "conversion"."сurrency"  
GO  
DROP TABLE "conversion"."users\_roles"  
GO  
DROP TABLE "conversion"."users"  
GO  
DROP TABLE "conversion"."roles"  
GO  
DROP TABLE "conversion"."history"  
GO  
  
CREATE TABLE "conversion"."history" (   
 "cur1\_id" bigint NOT NULL,  
 "cur2\_id" bigint NOT NULL,  
 "date" date NOT NULL,  
 "user\_id" bigint NOT NULL,  
 "id" bigserial NOT NULL,  
 "count\_cur1" real NOT NULL,  
 "count\_cur2" real NOT NULL,  
 "currency\_value\_by\_date" real NOT NULL,  
 PRIMARY KEY("id")  
)  
GO  
ALTER TABLE "conversion"."history" OWNER TO "postgres"  
GO  
CREATE TABLE "conversion"."roles" (   
 "id" bigserial NOT NULL,  
 "name" text NOT NULL,  
 PRIMARY KEY("id")  
)  
GO  
ALTER TABLE "conversion"."roles" OWNER TO "postgres"  
GO  
CREATE TABLE "conversion"."users" (   
 "second\_name" text NOT NULL,  
 "password" text NOT NULL,  
 "name" text NOT NULL,  
 "first\_name" text NOT NULL,  
 "email" text NOT NULL,  
 "id" bigserial NOT NULL,  
 "status" text NULL,  
 PRIMARY KEY("id")  
)  
GO  
ALTER TABLE "conversion"."users" OWNER TO "postgres"  
GO  
CREATE TABLE "conversion"."users\_roles" (   
 "user\_id" bigint NOT NULL,  
 "role\_id" bigint NOT NULL   
 )  
GO  
ALTER TABLE "conversion"."users\_roles" OWNER TO "postgres"  
GO  
CREATE TABLE "conversion"."сurrency" (   
 "id" real NOT NULL,  
 "num\_code" bigint NOT NULL,  
 "char\_code" text NOT NULL,  
 "nominal" smallint NOT NULL,  
 "name" text NOT NULL,  
 PRIMARY KEY("id")  
)  
GO  
ALTER TABLE "conversion"."сurrency" OWNER TO "postgres"  
GO  
CREATE TABLE "conversion"."сurrency\_value" (   
 "value" real NOT NULL,  
 "date" date NOT NULL,  
 "currency\_id" bigint NOT NULL,  
 "id" bigserial NOT NULL,  
 PRIMARY KEY("id")  
)  
GO  
ALTER TABLE "conversion"."сurrency\_value" OWNER TO "postgres"  
GO  
  
ALTER TABLE "conversion"."history"  
 ADD CONSTRAINT "user\_id\_fk"  
 FOREIGN KEY("user\_id")  
 REFERENCES "conversion"."users"("id")  
 MATCH SIMPLE  
 ON DELETE NO ACTION   
 ON UPDATE NO ACTION   
GO  
ALTER TABLE "conversion"."history"  
 ADD CONSTRAINT "constraint\_name2"  
 FOREIGN KEY("cur2\_id")  
 REFERENCES "conversion"."сurrency"("id")  
 MATCH SIMPLE  
 ON DELETE NO ACTION   
 ON UPDATE NO ACTION   
GO  
ALTER TABLE "conversion"."history"  
 ADD CONSTRAINT "constraint\_name1"  
 FOREIGN KEY("cur1\_id")  
 REFERENCES "conversion"."сurrency"("id")  
 MATCH SIMPLE  
 ON DELETE NO ACTION   
 ON UPDATE NO ACTION   
GO  
ALTER TABLE "conversion"."users\_roles"  
 ADD CONSTRAINT "users"  
 FOREIGN KEY("user\_id")  
 REFERENCES "conversion"."users"("id")  
 MATCH SIMPLE  
 ON DELETE NO ACTION   
 ON UPDATE NO ACTION   
GO  
ALTER TABLE "conversion"."users\_roles"  
 ADD CONSTRAINT "roles\_fk"  
 FOREIGN KEY("role\_id")  
 REFERENCES "conversion"."roles"("id")  
 MATCH SIMPLE  
 ON DELETE NO ACTION   
 ON UPDATE NO ACTION   
GO  
ALTER TABLE "conversion"."сurrency\_value"  
 ADD CONSTRAINT "constraint\_name1"  
 FOREIGN KEY("currency\_id")  
 REFERENCES "conversion"."сurrency"("id")  
 MATCH SIMPLE  
 ON DELETE NO ACTION   
 ON UPDATE NO ACTION   
GO