МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информатики и вычислительной техники

Кафедра информационной безопасности

Отчёт к курсовому проекту

по дисциплине «Безопасность систем баз данных»

**Разработка базы данных для книжного магазина**

Выполнили: студенты группы БИ-31

Урванцев Д.Н., Петухова Э.Э.,

Филонова М.В., Орехова А.В.

Проверил: доцент кафедры

ИБ Сучков Д.С.

Йошкар-Ола

2020 г.

**СОДЕРЖАНИЕ**

**Введение3**

**1. Техническое задание4**

1.1 Требования к курсовой работе4

1.2 Требования к базе данных4

1.3 Требования к API (минимальное количество реализованных методов)4

**2. Порядок выполнения работы5**

2.1 Этапы разработки базы данных5-9

2.2 Этапы разработки API10-23

**3. Приложения24**

3.1 ER-диаграмма24

3.2 Ссылка на github.com24

**4. Вывод25**

**Введение**

В курсовой работе рассматривается создание базы данных, предназначенной для автоматизации работы книжного магазина. База данных позволяет продавцу заключать сделки, создавать покупателей, узнавать об ассортименте книг. Также реализована автоматизация продажи книги, где можно выбрать книгу, узнать ее стоимость, дату покупки и данные о продавце.

## **Техническое задание**

## ***1.1 Требования к курсовой работе:***

* Получить структуру данных из файла, согласно варианту. Привести к 3й нормальной форме. Добавить недостающие таблицы.
* Составить ER-диаграмму, применяя mySQL Workbench или Dbearer.
* Разработать API для базы данных на любом языке, выполняющемся на стороне сервера (php, ASP.NET, Java, python, node.js, etc).
* Взаимодействие должно осуществляться по клиент-серверной архитектуре, подключение с клиентской программы недопустимо.
* Провести настройку пользователей базы данных для разграничения прав доступа, привести пример конфигурации.
* Все документы и исходные коды для курсовой работы должны храниться под контролем системы контроля версий — git или mercurial (https://github.com/, https://bitbucket.org/).
* Во время сдачи курсового проекта необходимо предоставить отчет о проделанной работе в печатном виде (отчет).

## ***1.2 Требования к базе данных***

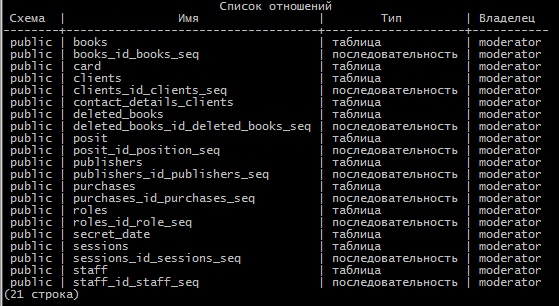
* Наличие не менее 7 таблиц, в том числе таблицы сессий и пользователей.
* Структура таблицы должна содержать не менее 3-х полей, одно из которых ключевое.
* Правомерное использование типов данных.
* Обязательно использование триггеров и/или хранимых процедур.
* Форма нормализации не менее 3NF.
* Индексирование по полям поиска.

## **1.3 *Требование к API (минимальное количество реализованных методов)***

* аутентификация пользователя (создание сессии);
* добавление/удаление/изменение данных в таблицах;
* выборка данных их ключевых таблиц по запросам;
* выборка данных из таблиц с объединением результатов.

1. **Порядок выполнения работы**
   1. Этапы разработки базы данных

Разработана база данных, содержащая 12 таблиц, в каждой таблице есть ключевое поле. Владельцем всех таблиц является moderator, который делегирует права на изменение данных editor. Это будет хорошо в том случае, если будет совершена sql-инъекция, направленная на изменение структуры базы данных.



В том числе созданы таблицы сессий (*sessions)* и пользователей (*clients*). Также есть таблицы, содержащие информацию о данных пользователей –  *card, secret\_date, contact\_details\_clients.*

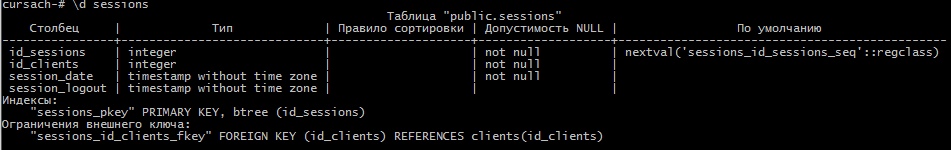
Информацию о книгах можно посмотреть в таблице *books*, об удаленных книгах – в таблице *deleted\_books*.

Отдельно можно посмотреть информацию об авторах в таблице *publishers*, а также информацию о сделках в таблице *purchases.*

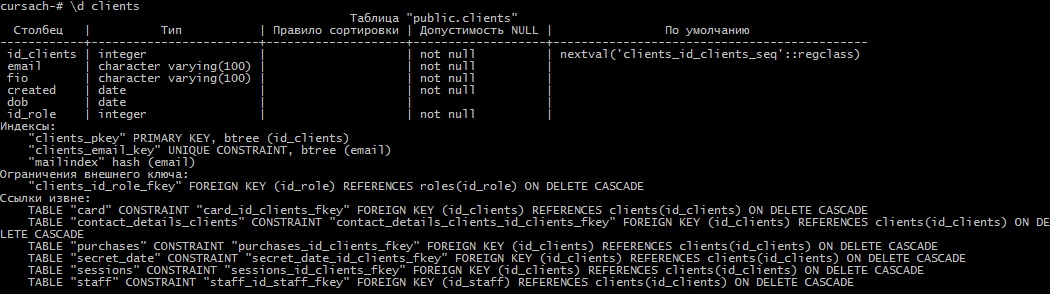
В таблице *roles* показывается информация о ролях, в таблице *posit* – информация о существующих должностях, в таблице *staff* – информация о том, какую должность имеет работник.

Структуры реализованных таблиц:

* таблица s*essions*



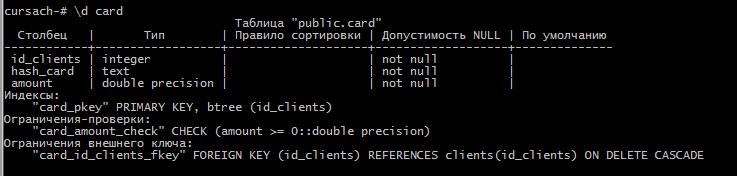
* таблицa *clients*



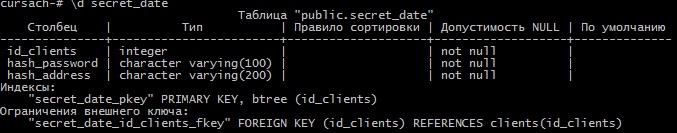
Так как аутентификация производится по email, был создан hash-индекс на это поле для более быстрого поиска.

https://sun9-8.userapi.com/impg/ZIEYAqScaDeZRWRoI56sXP46uaPBMguMvHiI6w/AHyHFTMshE4.jpg?size=459x32&quality=96&proxy=1&sign=0a4a81363585f04d8c2476e8f094cea7&type=album

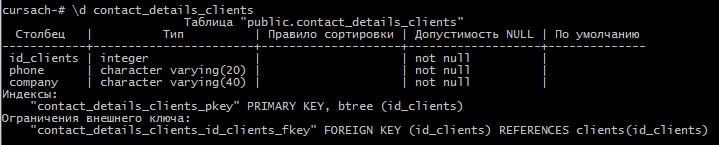
* таблица *card*



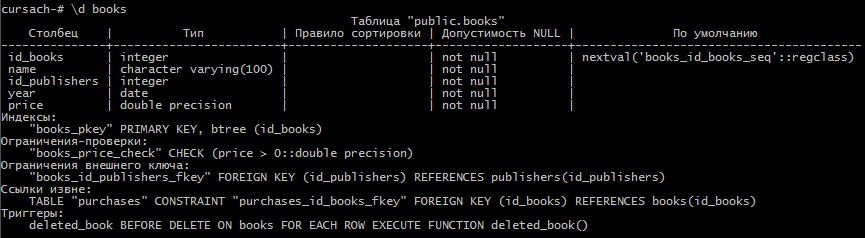
* таблица *secret\_date*



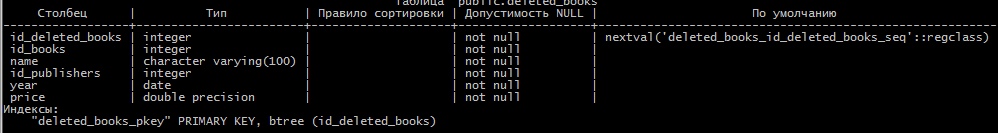
* таблица *contact\_details\_clients*



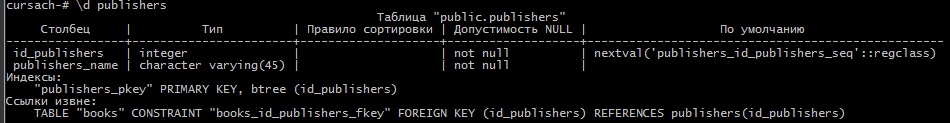
* таблица *books*



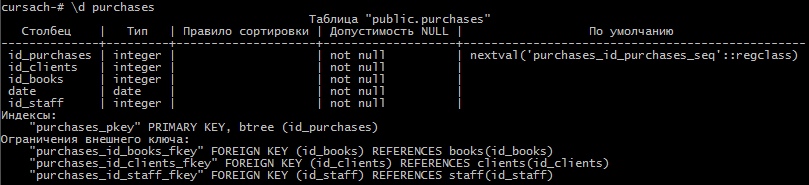
* таблица *deleted\_books*



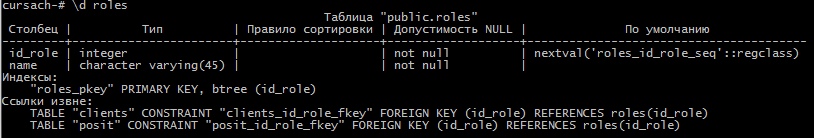
* таблица *publishers*



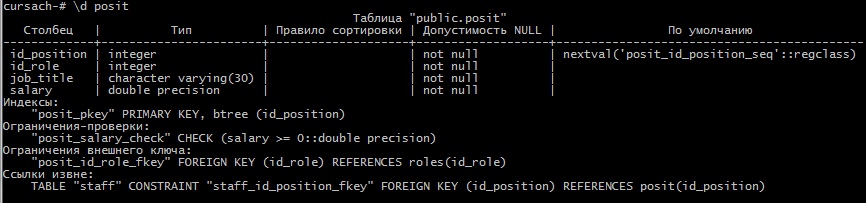
* таблица *purchases*



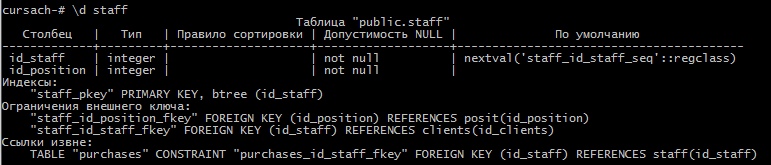
* таблица *roles*



* таблица *posit*



* таблица *staff*

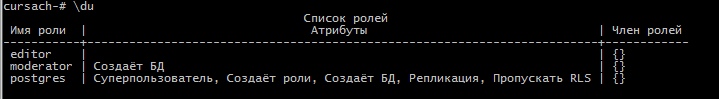


Используемый триггер и функция для него:

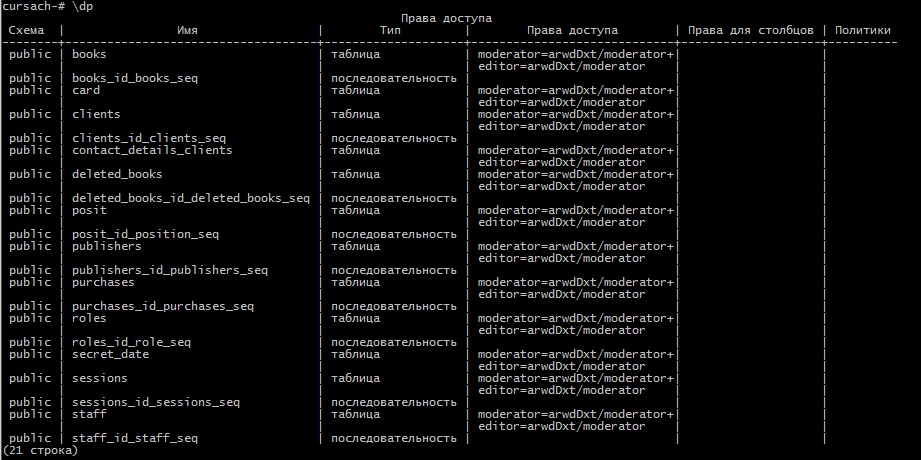
* *deleted\_book* – триггер, отвечающий за удаление книг:

CREATE FUNCTION deleted\_book() RETURNS trigger AS $deleted\_book$  
BEGIN  
INSERT INTO deleted\_books( id\_books, name, id\_publishers, year, price) values ([OLD.id](https://vk.com/away.php?to=http%3A%2F%2FOLD.id&cc_key=" \t "_blank)\_books, [OLD.name](https://vk.com/away.php?to=http%3A%2F%2FOLD.name&cc_key=), [OLD.id](https://vk.com/away.php?to=http%3A%2F%2FOLD.id&cc_key=" \t "_blank)\_publishers, OLD.year, OLD.price);  
RETURN OLD;  
END;  
$deleted\_book$ LANGUAGE plpgsql;  
  
CREATE TRIGGER deleted\_book BEFORE DELETE ON books  
FOR EACH ROW EXECUTE PROCEDURE deleted\_book();

Проведена настройка пользователей базы данных для разграничения прав доступа и прав на редактирование структуры базы данных:



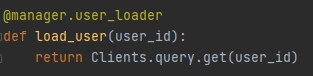
А также на изменение данных таблицы:

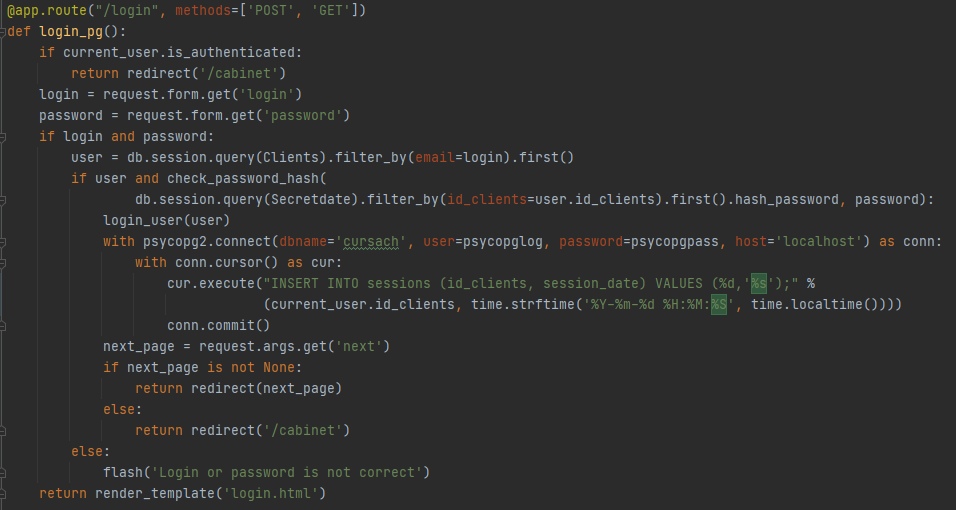


2.2 Этапы разработки API

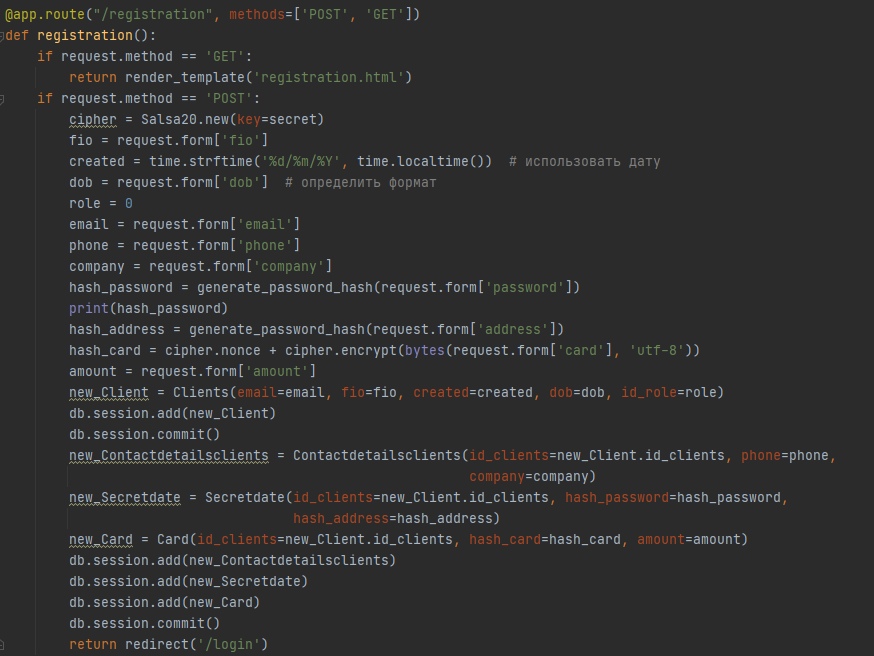
Было разработано API для логинации, аутентификации пользователей, удаления книг и регистрации, написанное на языке Python.

Аутентификация:



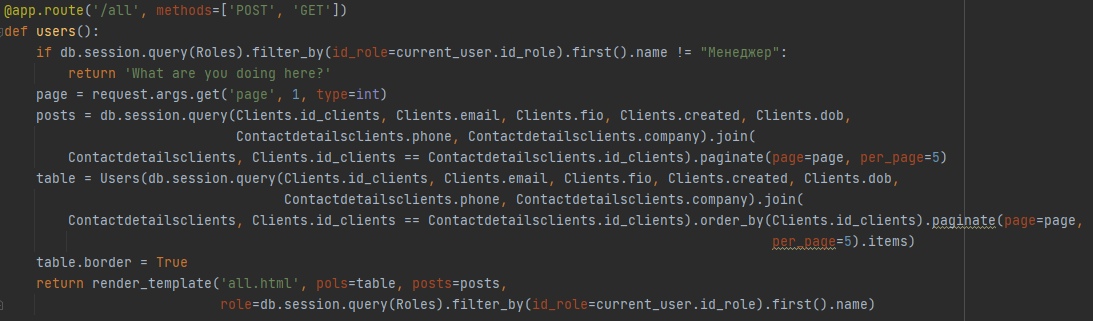


Регистрация:



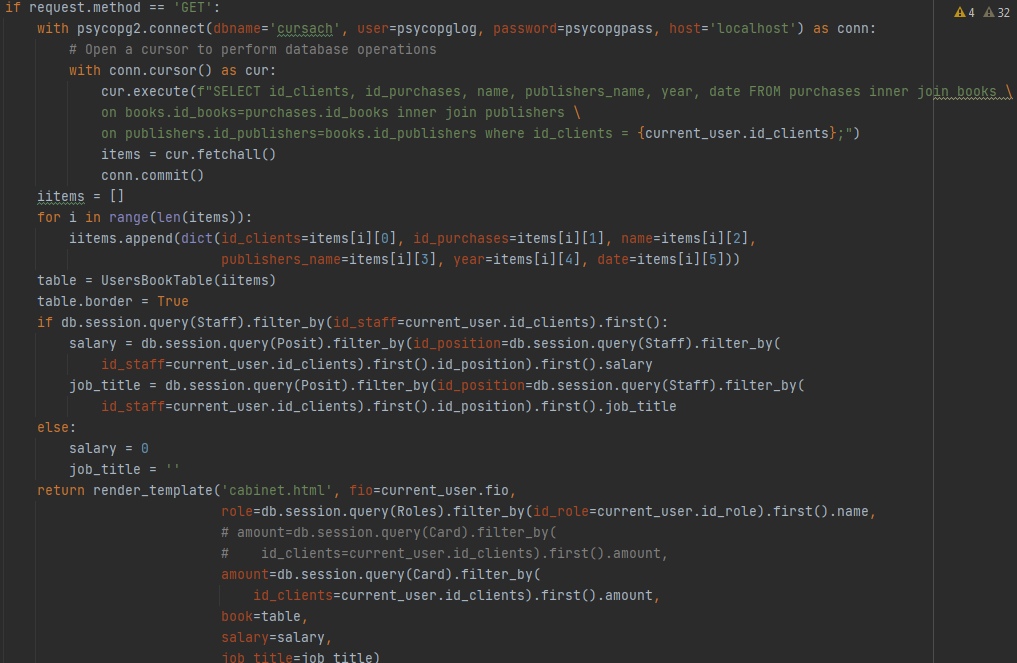
**Функции вывода из ключевых таблиц:**

Вывод пользователей:

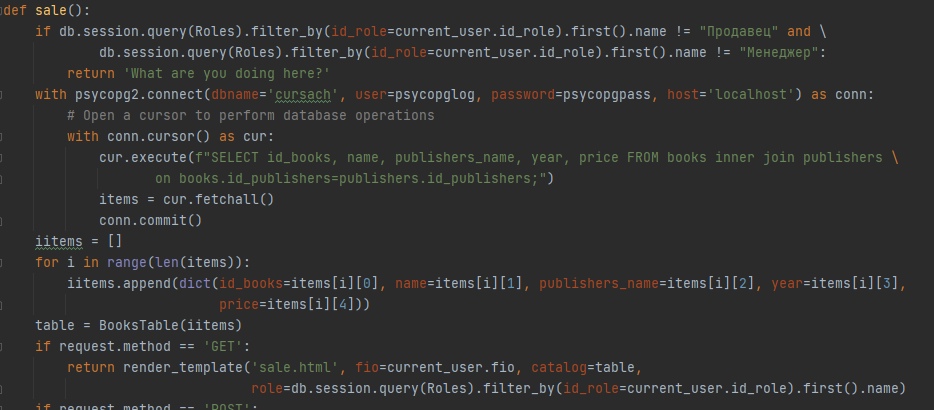


Пагинация реализована в этой функции при помощи метода «paginate».

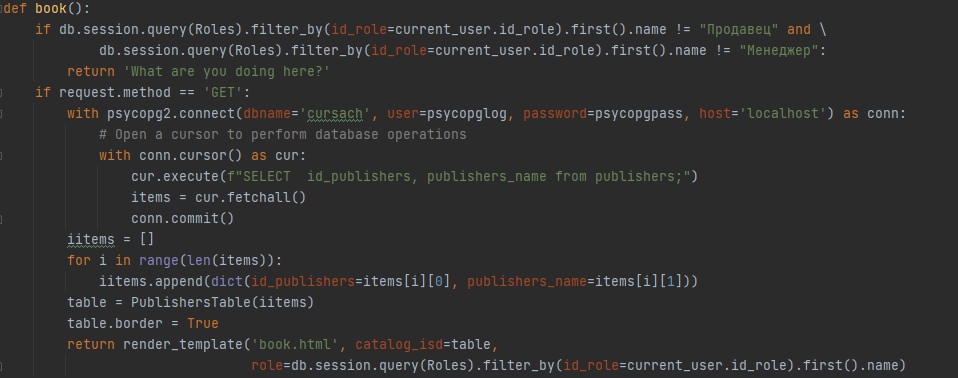
Вывод данных пользователя:



Вывод данных по сделке:

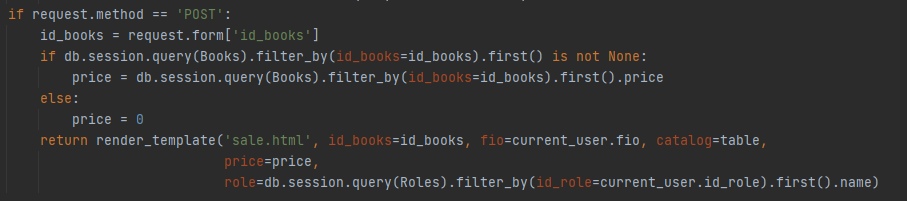


Вывод данных по книге:



**Добавление данных в БД:**

Добавление сделки:



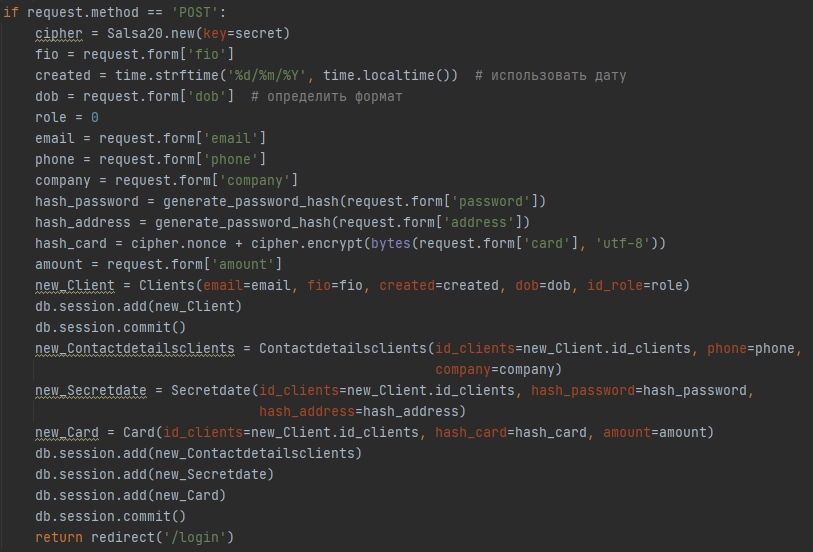
Добавление денег на счёт:



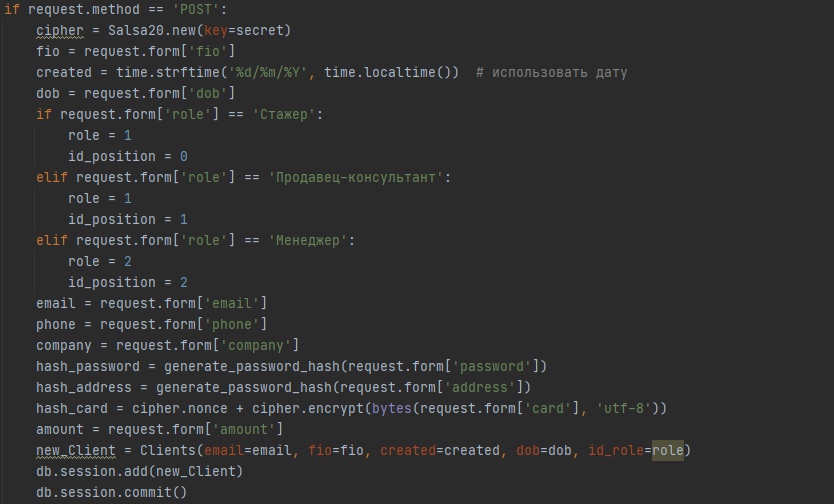
Добавление издателя:

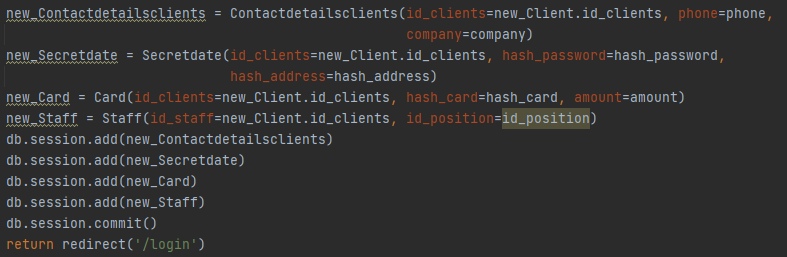


Добавление нового покупателя:

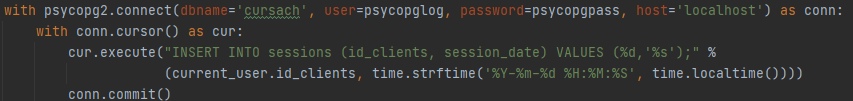


Добавление нового работника:



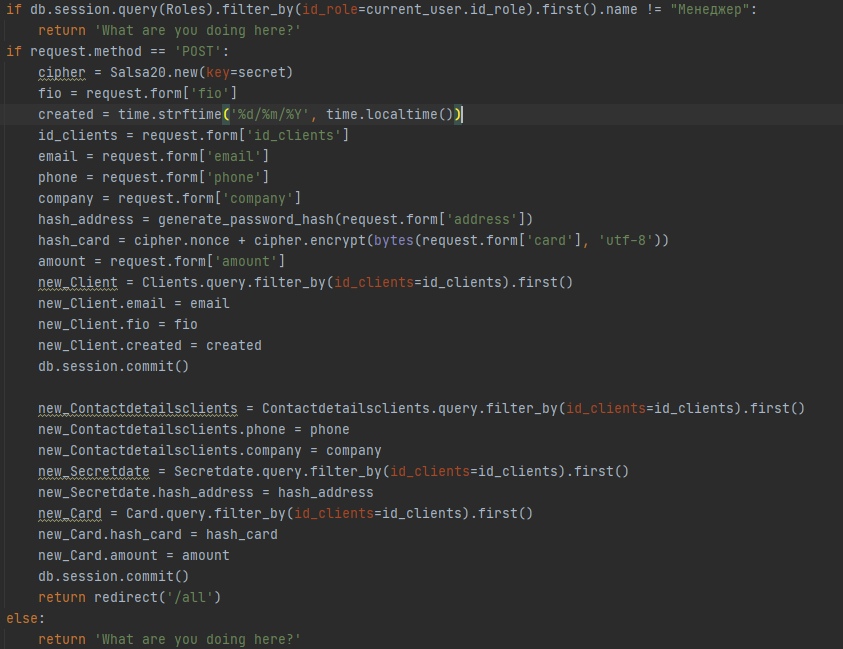


Добавление сессии:

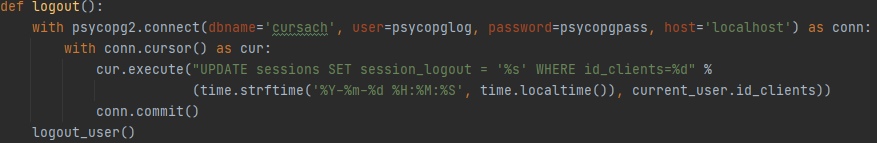


**Изменение данных в БД:**

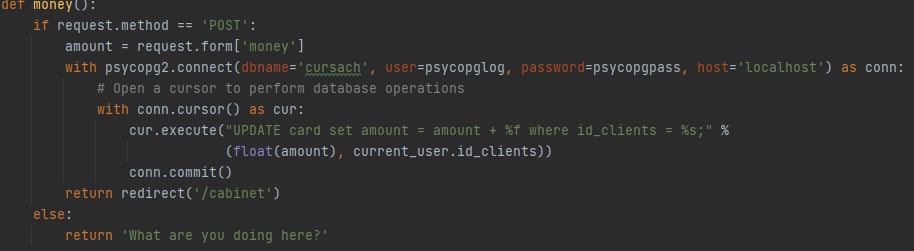
Изменение данных пользователей:



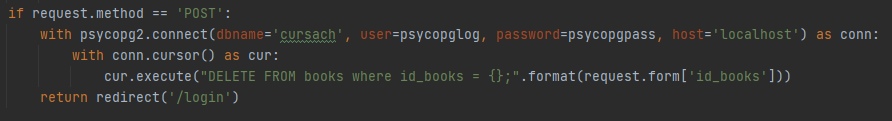
Обновление данных сессии:



Пополнение баланса:



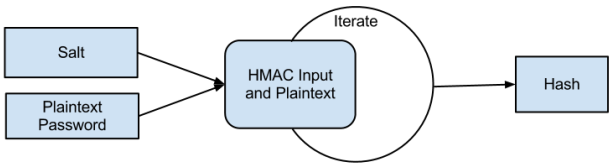
Удаление книги:



При помощи алгоритма Salsa20 происходит шифрация номеров карт. Данная система является системой поточного шифрования и разработана Д. Бернштейном. Благодаря тому, что преобразования каждого столбца и каждой строки не зависят друг от друга, вычисления, необходимые для шифрования, легко [распараллеливаются](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F). Сложность «ручного» восстановления зашифрованного диска в том, что этот процесс занимает значительное время (часы) и требует больших объемов свободного дискового пространства.

Хэширование паролей происходит с помощью pbkdf2:sha256.PBKDF2 (Password-Based Key Derivation Function) – это функция получения ключа, разработанная RSA Laboratories, используемая для получения стойких ключей на основе хэша. Она работает путем применения псевдослучайной хэш-функции SHA-256 к строке, в нашем случае – к паролю, вместе с солью (модификатором входа хэш-функции) и повторением этого процесса большое число раз.

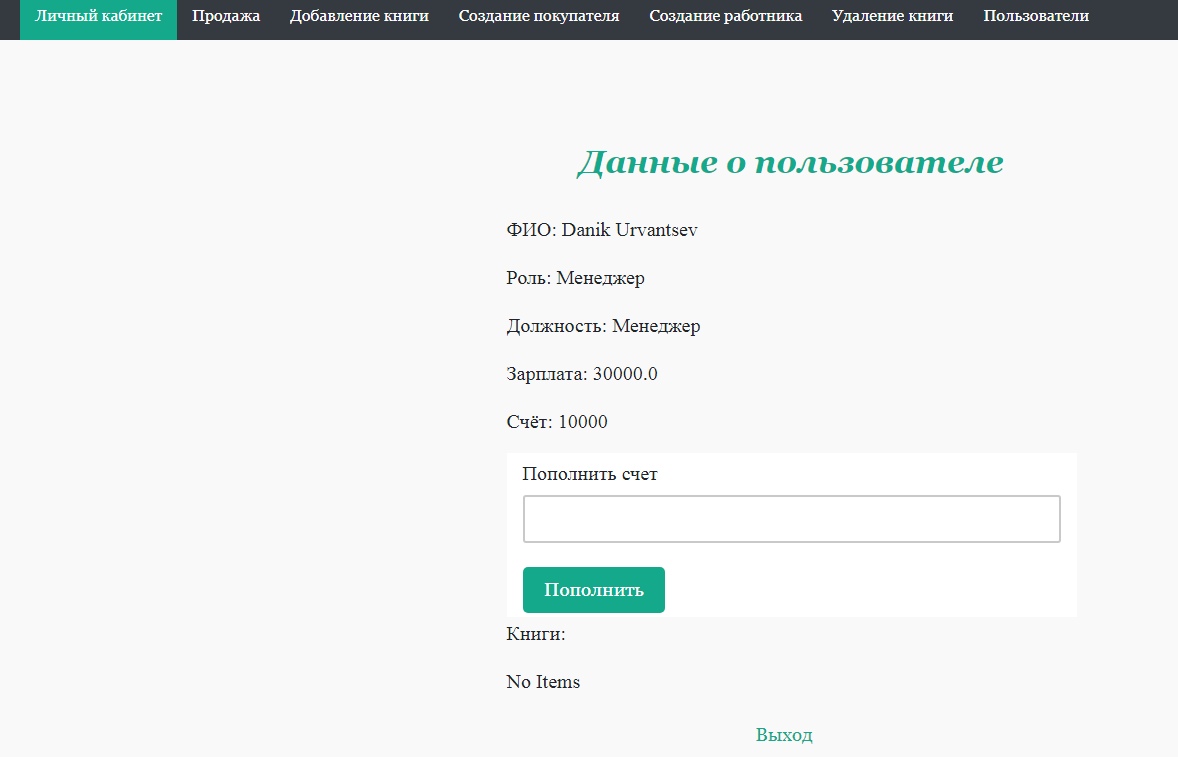
Данный процесс может быть обобщен следующей диаграммой:



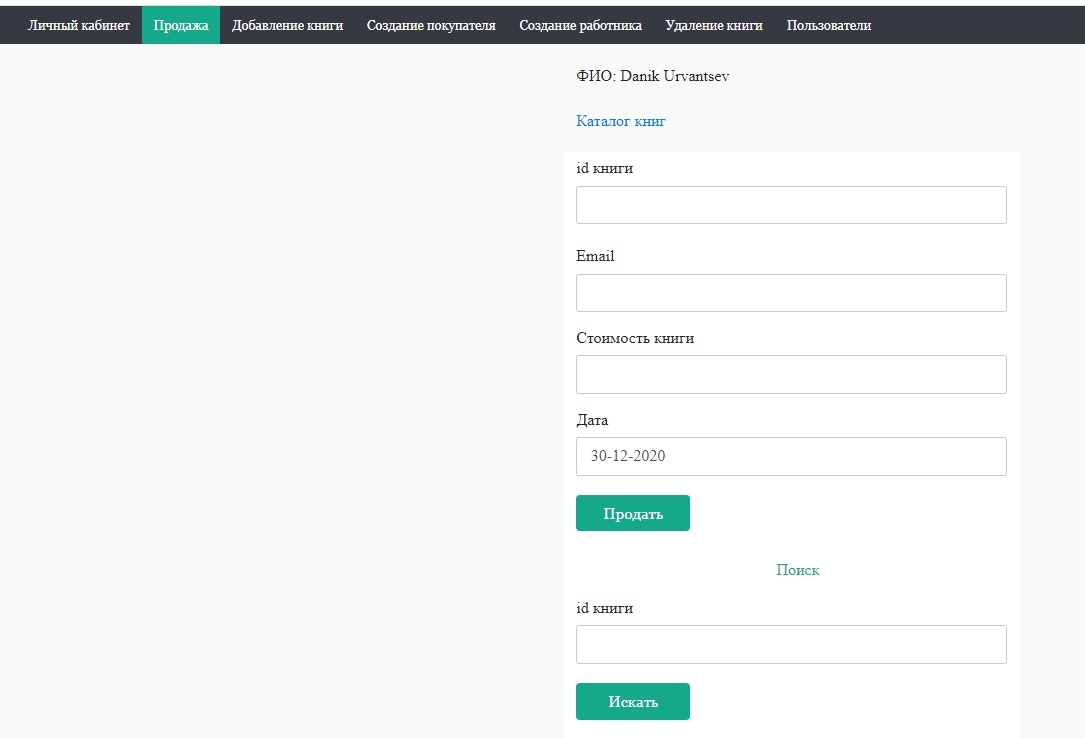
При использовании этого алгоритма хэшируется соль и открытый текст для получения первого хэша, затем в цикле тот же алгоритм используется для вычисления хэша от открытого текста и результата предыдущей итерации, после чего возвращается результат применения операции XOR ко всем вычисленным хэшам. В нашем случае данная операция на серверной стороне выполняется 150000 раз, поэтому на выходе генерируется сильный, стойкий ко взлому хэш, который можно безопасно использовать для хранения паролей.

Ниже представлены скриншоты страниц с ролью менеджера:

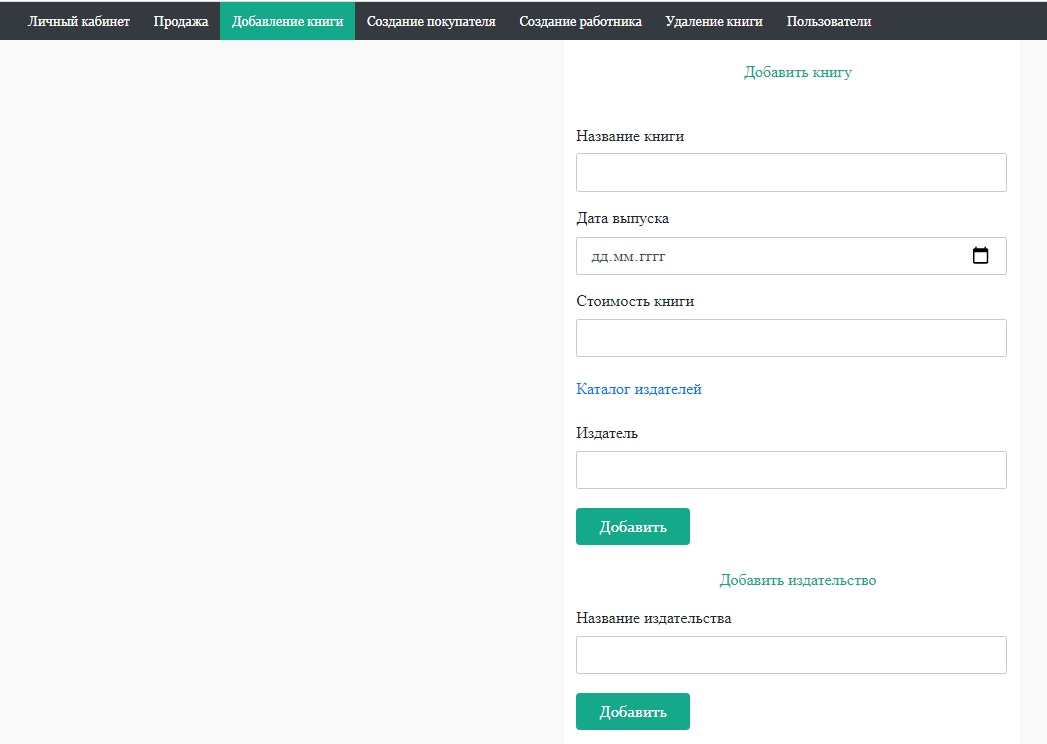
Личный кабинет:



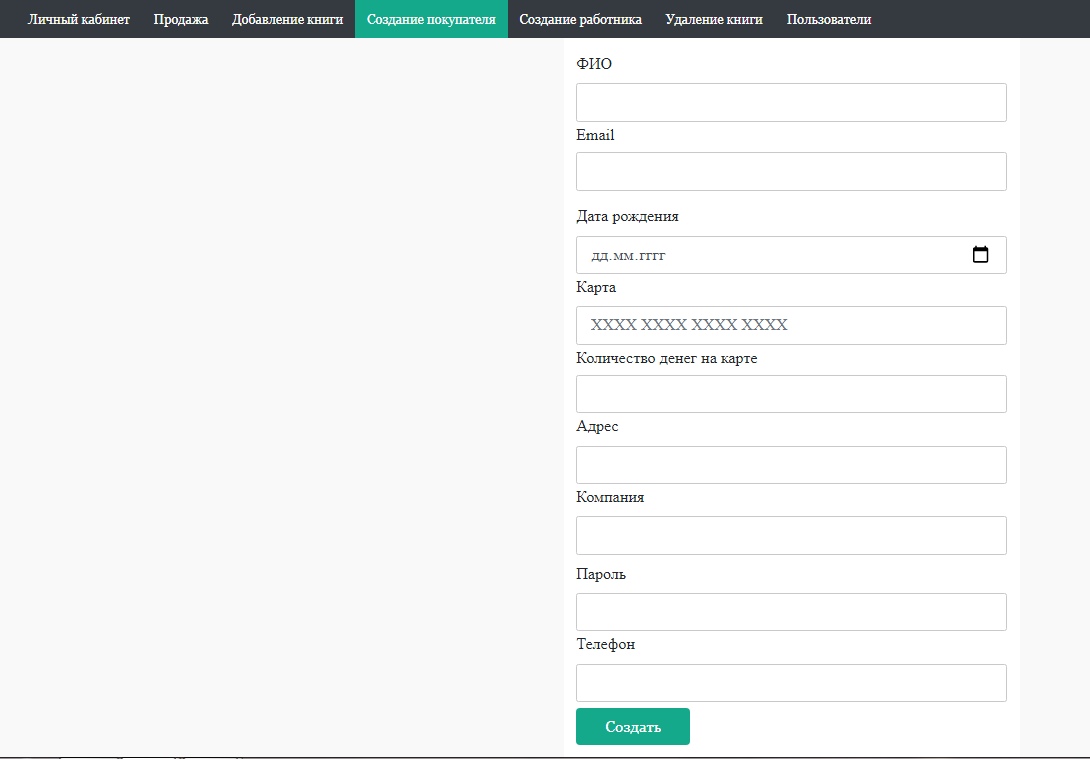
Продажа:



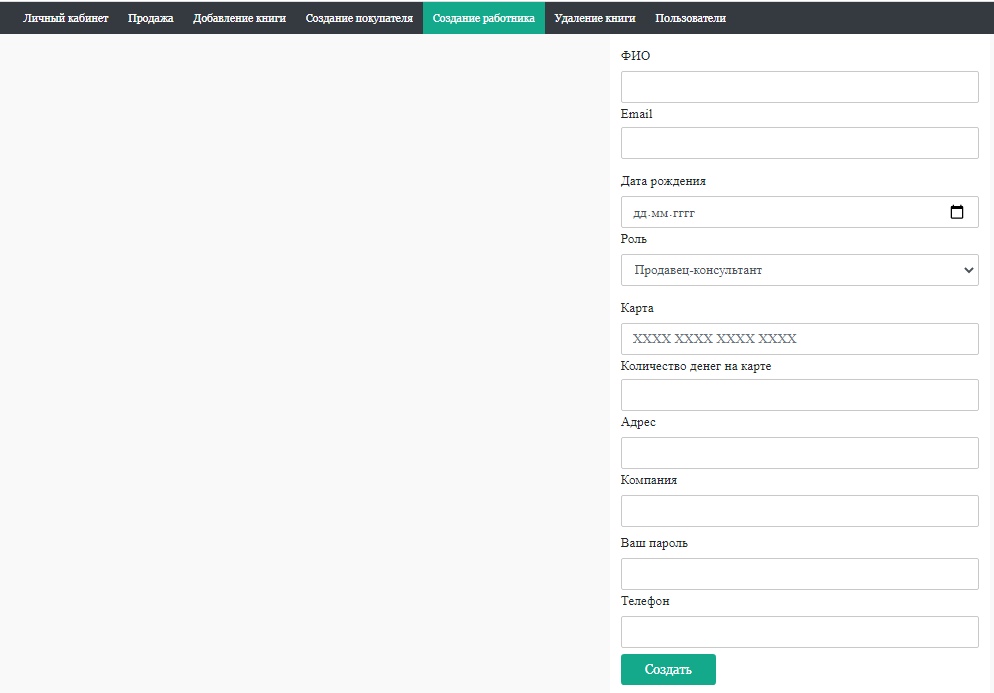
Добавление книги:



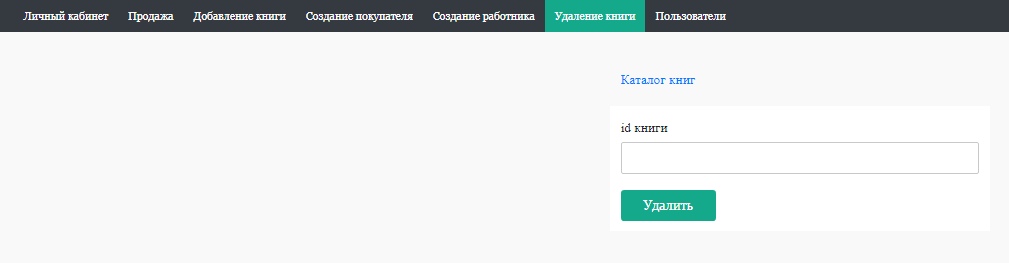
Создание покупателя:



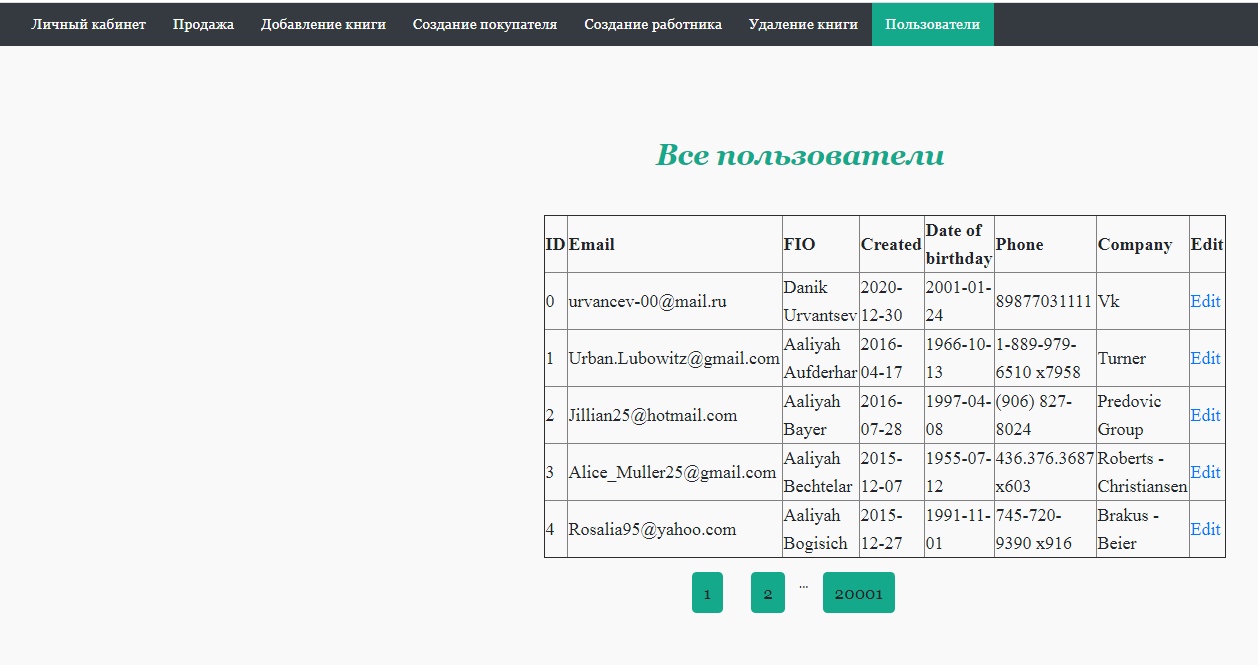
Создание работника:



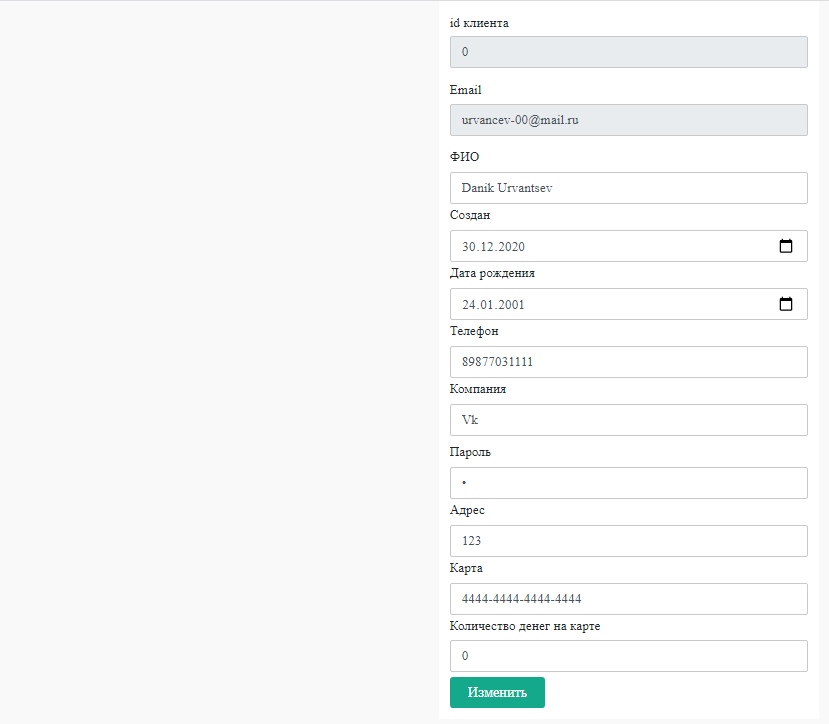
Удаление книги:



Пользователи (реализована пагинация, отображается до 5 пользователей):

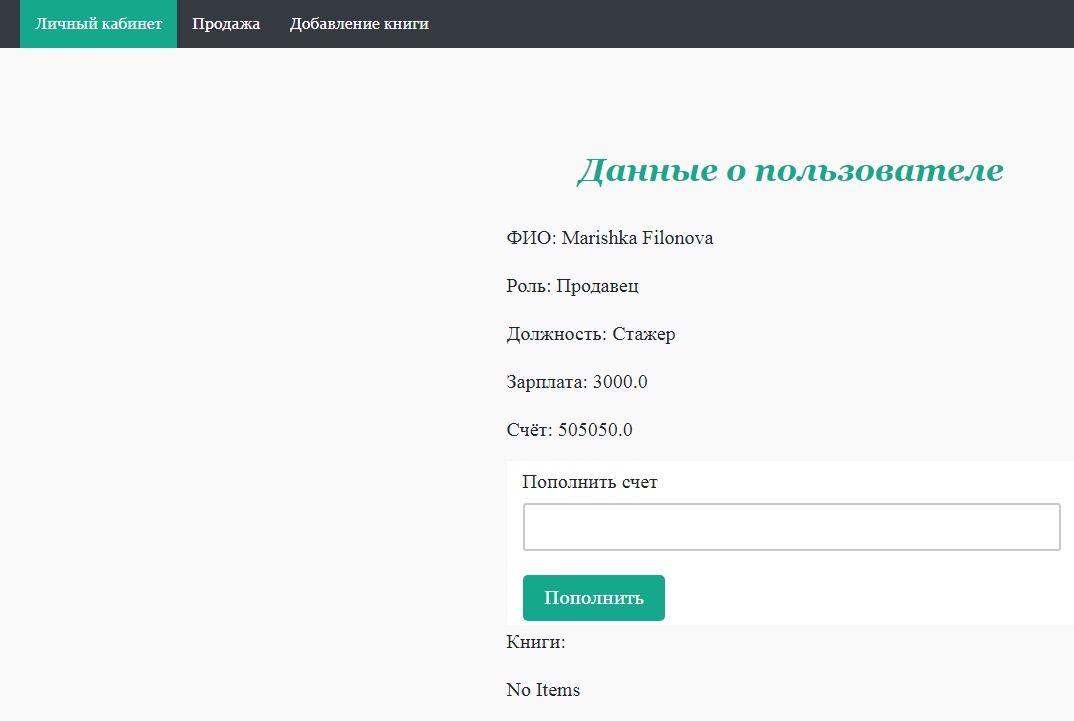


Редактирование данных пользователя при помощи кнопки edit:

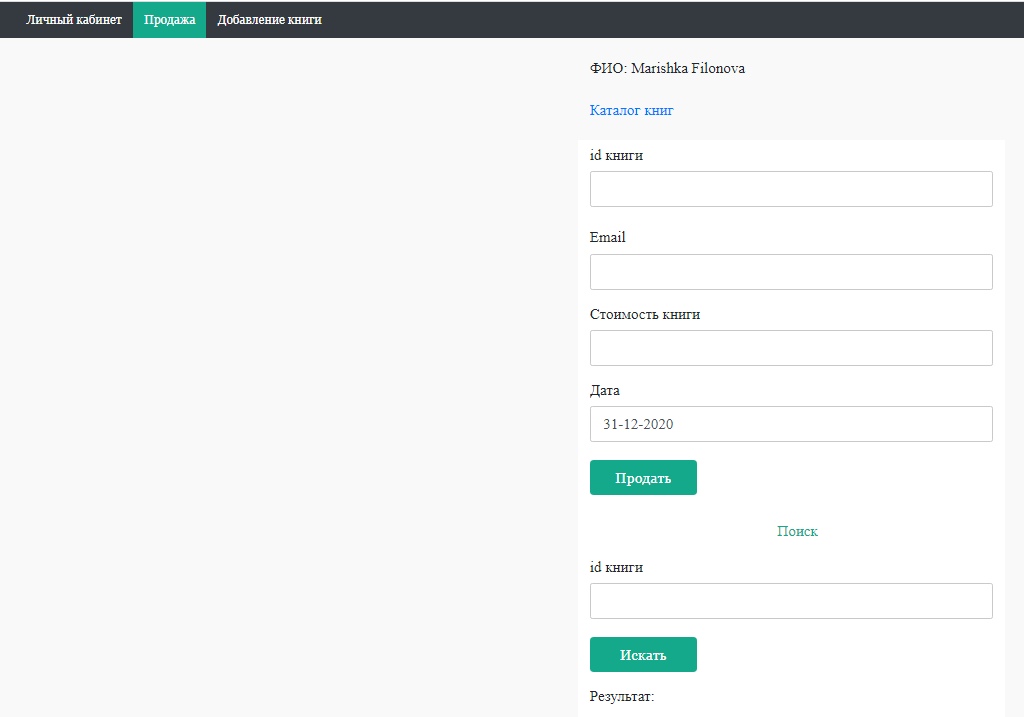


С ролью продавца:

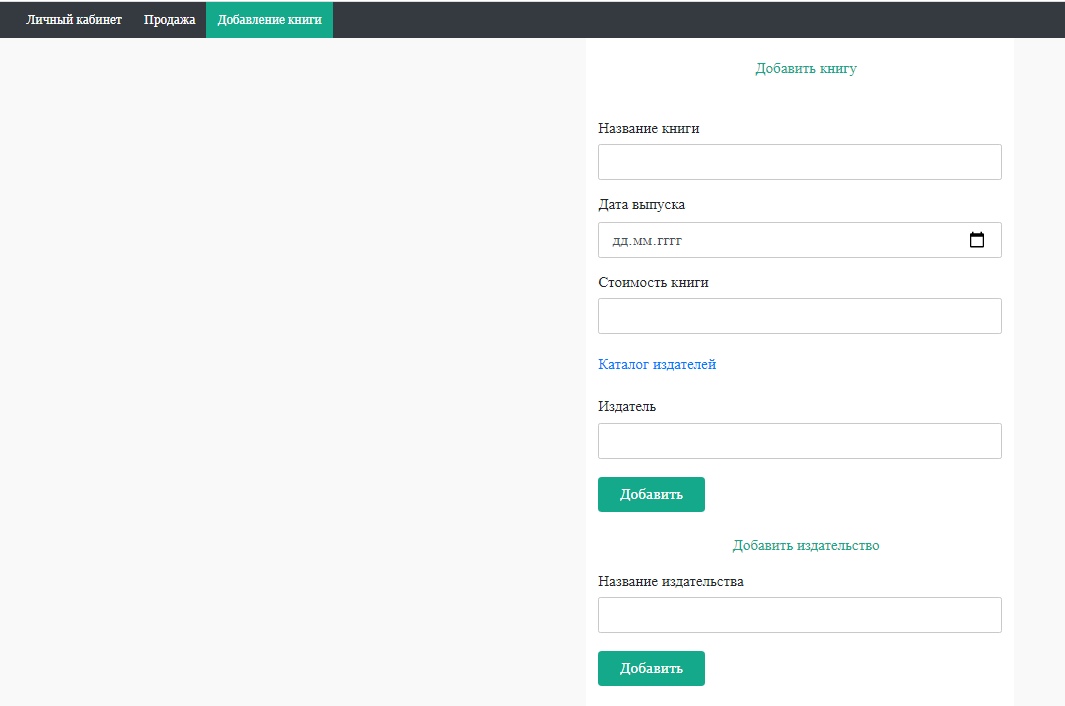
Личный кабинет:



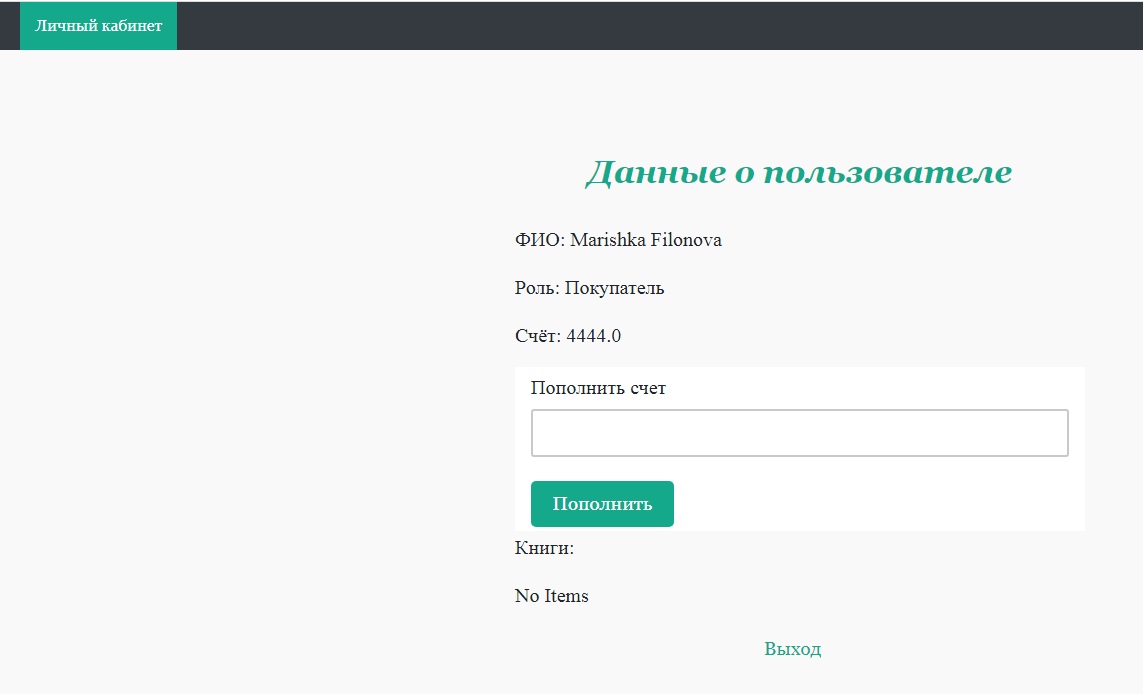
Продажа:



Добавление книги:

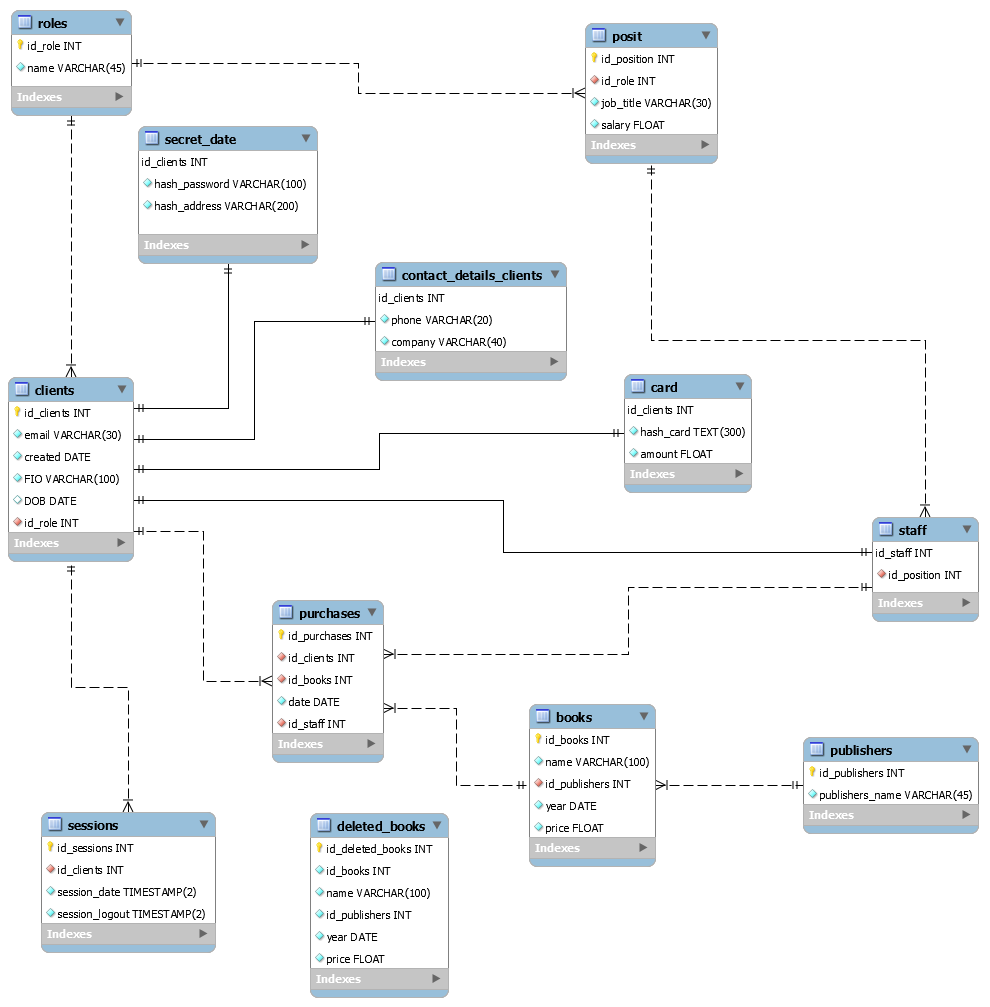


С ролью покупателя:



**Приложения**

1. ER-диаграмма



1. Исходные коды и документы:

[https://github.com/danchik1155/ServerDB](https://vk.com/away.php?to=https%3A%2F%2Fgithub.com%2Fdanchik1155%2FServerDB&cc_key=)

**Вывод**

Во время выполнения курсового проекта были изучены методы работы с базами данных, способы управления. Таким образом, в ходе работы была разработана база данных, а также API, для автоматизации работы книжного магазина.