# 1. СИСТЕМНЫЙ АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Предметная область – книжный интернет-магазин.

АИС предназначена для автоматизации деятельности сотрудников магазина и упрощения процесса приобретения нужной покупателю литературы.

Схема работы интернет-магазина включает в себя:

– Авторизация/Регистрация в системе;

– Получение заказа от клиента;

– Получение оплаты;

– Резервирование товаров (экземпляров книг) на складе;

– Упаковка и доставка товаров в пункт выдачи заказа;

– Выдача товара.

Таким образом, основными информационными объектами предметной области будут: заказ, клиент, товар/книга (поскольку магазин торгует только ими), экземпляр книги, а также сотрудник и пункт выдачи.

Каждая книга может присутствовать в нескольких экземплярах, поэтому экземпляру должны быть присвоены уникальный инвентарный номер и адрес склада. Книга характеризуется следующими параметрами:

– Уникальный шифр (ISBN);

– Название;

– Фамилии и инициалы авторов (могут отсутствовать);

– Город издания;

– Издатель;

– Год издания;

– Количество страниц;

– Раздел;

– Стоимость.

За каждым пунктом выдачи (код пункта, адрес, оператор) закреплены операторы пункта выдачи – по сути, те же кладовщики, только, в отличие от последнего, именно они непосредственно взаимодействуют с клиентом (код клиента, фамилия, имя, отчество, телефон, e-mail, пароль).

Состав ролевой системы:

Администрация («администратор») должна вести учет всех сотрудников и контролировать их деятельность.

Складская служба («кладовщик») должна вести учет и контроль выдачи экземпляров и следить за поступлением новых. Учет должен производиться для планирования складских остатков и закупок.

Операторы пункта выдачи («оператор») отвечают за выдачу товара клиенту («клиент»).

В данной системе существуют и ограничения:

– Клиенту предоставляется информация о книгах, экземплярах и его заказах только на чтение;

– Клиент может делать несколько заказов, но каждому заказу соответствует только один клиент;

– При регистрации в системе клиент может не указывать фамилию и отчество, телефон и e-mail являются обязательными пунктами;

– Стоимость заказа указывается только в рублях;

– Администратору доступна вся информация о сотрудниках на чтение и запись;

– Кладовщику доступна вся информация об экземплярах на чтение и запись, а также частично информация о заказах;

– Оператору пункта выдачи доступна лишь информация о заказах на чтение;

– К каждому пункту выдачи привязан лишь один сотрудник (оператор пункта выдачи).

Возможны другие ограничения, которые будут выявлены на последующих этапах проектирования и реализации системы.

С информацией о деятельности всего магазина, которая отражена в совокупности всех каталогов и учетных записей всех отделов и сотрудников, может взаимодействовать (чтение/запись) напрямую лишь администратор базы данных, другим доступна лишь та информация, которая необходима для осуществления своей деятельности, например, кладовщику недоступна информация о пункте выдачи.

Типовые запросы к данным:

– Информация о заказе (все его атрибуты);

– Информация о сотруднике (все его атрибуты);

– Информация о заказах, которые находятся в конкретном пункте выдачи;

– Фильтрация информации о книгах, сотрудниках и других сущностях.

# 2. ИНФОЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

Перечень атрибутов

Таблица 1:

|  |  |  |
| --- | --- | --- |
| Обозначение | Описание | Примечание |
| x1 | ISBN |  |
| x2 | Название книги | Книги |
| x3 | Фамилия и инициалы авторов | Книги |
| x4 | Город издания | Книги |
| x5 | Издатель | Книги |
| x6 | Год издания | Книги |
| x7 | Количество страниц | Книги |
| x8 | Раздел | К которым принадлежит книга, даже если книгу сложно отнести к какому-либо разделу (области знаний), книга все равно должна иметь хотя бы один раздел |
| x9 | Стоимость | Книги |
| x10 | Код экземпляра книги |  |
| x11 | Уникальный инвентарный номер | Экземпляра книги, код, в котором зашифровано местоположение экземпляра на складе |
| x12 | Склад | Экземпляра книги, адрес склада, в котором хранится экземпляр |
| x13 | ISBN | Экземпляра книги, ссылка на книгу |
| x14 | Код заказа |  |
| x15 | Дата | Заказа |
| x16 | Время | Заказа |
| x17 | Статус | Заказа |
| x18 | Стоимость | Заказа |
| x19 | Способ оплаты | Заказа |
| x20 | Клиент | Заказа, ссылка на клиента |
| x21 | Пункт выдачи | Заказа, ссылка на пункт выдачи |
| x22 | Код клиента |  |
| x23 | Фамилия | Клиента |
| x24 | Имя | Клиента |
| x25 | Отчество | Клиента |
| x26 | Телефон | Клиента |
| x27 | E-mail | Клиента |
| x28 | Пароль | Клиента |
| x29 | Код сотрудника |  |
| x30 | Отдел | Сотрудника, ссылка отдел, в котором он работает |
| x31 | Должность | Сотрудника |
| x32 | Фамилия | Сотрудника |
| x33 | Имя | Сотрудника |
| x34 | Отчество | Сотрудника |
| x35 | Дата рождения | Сотрудника |
| x36 | Домашний адрес | Сотрудника |
| x37 | Телефон | Сотрудника |
| x38 | E-mail | Сотрудника |
| x39 | Пароль | Сотрудника |
| x40 | Примечания | Сотрудника |
| x41 | Код пункта выдачи |  |
| x42 | Адрес | Пункта выдачи |
| x43 | Оператор | Пункта выдачи, ссылка на сотрудника, который работает в данном пункте выдачи |
| x44 | Код отдела |  |
| x45 | Название отдела |  |
| x46 | Код заказа |  |
| x47 | Код экземпляра |  |

Перечень сущностей

Таблица 2:

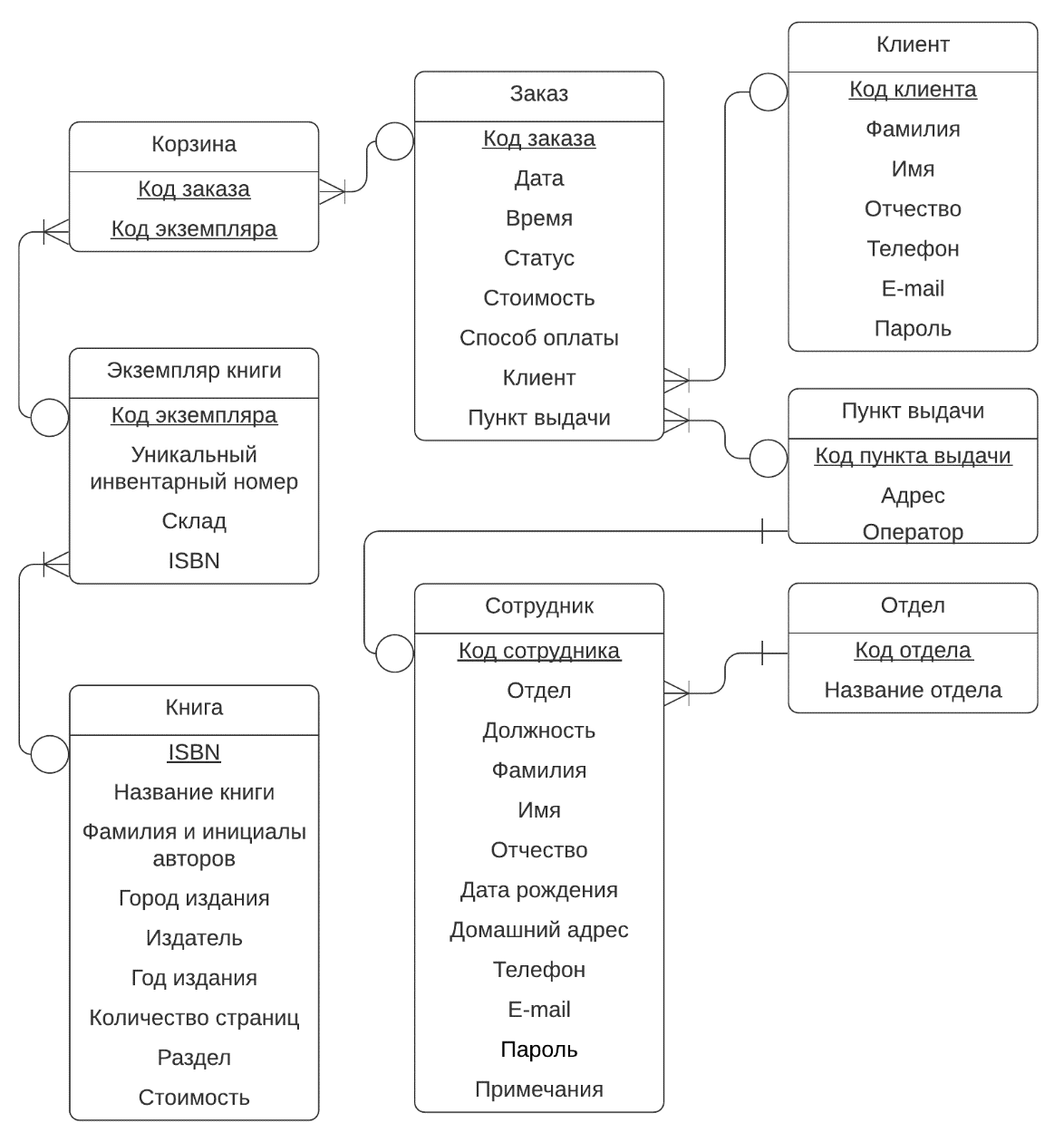
|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение | Описание | Состав атрибутов | Первичный ключ |
| Y1 | Книга | x1 – x9 | x1 |
| Y2 | Экземпляр книги | x10 – x13 | x10 |
| Y3 | Заказ | x14 – x21 | x14 |
| Y4 | Клиент | x22 – x28 | x22 |
| Y5 | Сотрудник | x29 – x40 | x29 |
| Y6 | Пункт выдачи | x41 – x43 | x40 |
| Y7 | Отдел | x44 – x45 | x44 |
| Y8 | Корзина | x46 – x47 | x47 |

Определение связей между сущностями

Таблица 3:

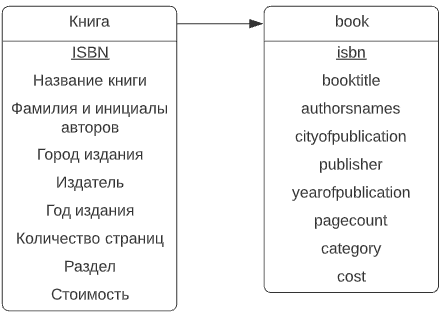
|  |  |  |  |
| --- | --- | --- | --- |
| Сущность 1 | Сущность 2 | Тип связи | Обязательность |
| Y1 | Y2 | 1:M | 0-1 |
| Y2 | Y8 | 1:M | 0-1 |
| Y3 | Y4 | M:1 | 1-0 |
| Y3 | Y6 | M:1 | 1-0 |
| Y3 | Y8 | 1:M | 0-1 |
| Y5 | Y6 | 1:1 | 0-1 |
| Y5 | Y7 | M:1 | 1-1 |

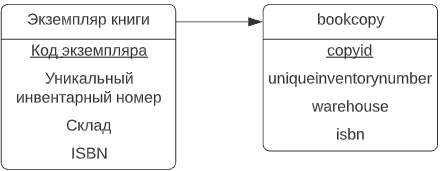
Инфологическая схема

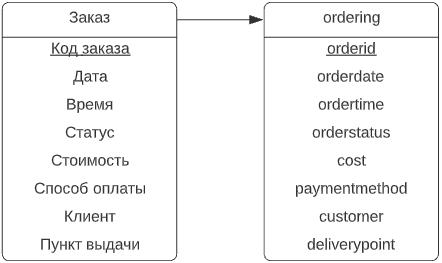


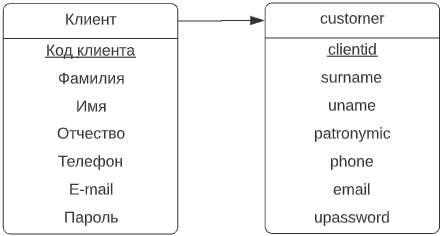
# 3. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

Преобразование сущностей в отношения

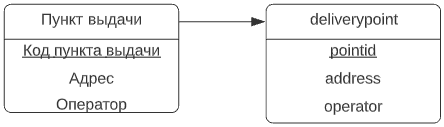


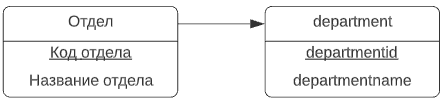


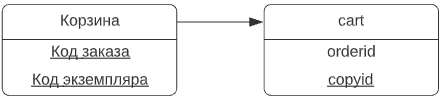












Определение свойств атрибутов отношений

Свойства атрибутов отношения book:



Свойства атрибутов отношения bookcopy:



Свойства атрибутов отношения ordering:



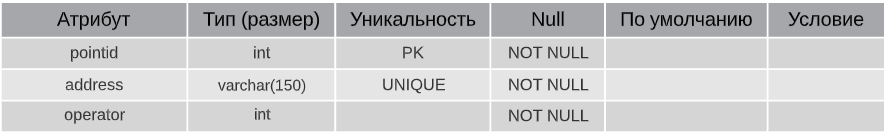
Свойства атрибутов отношения customer:



Свойства атрибутов отношения employee:



Свойства атрибутов отношения deliverypoint:



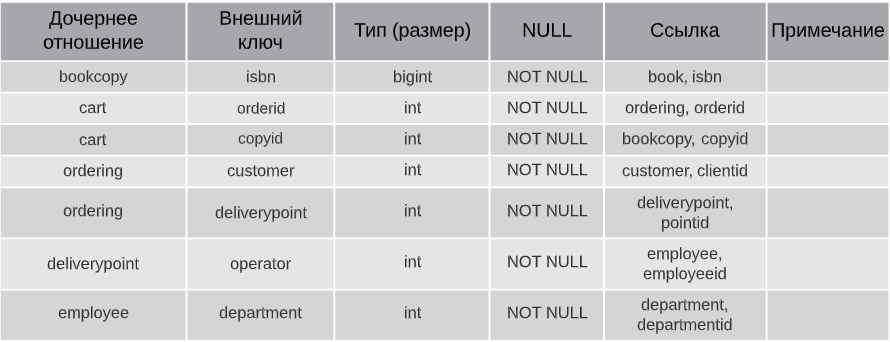
Свойства атрибутов отношения department:



Свойства атрибутов отношения cart:



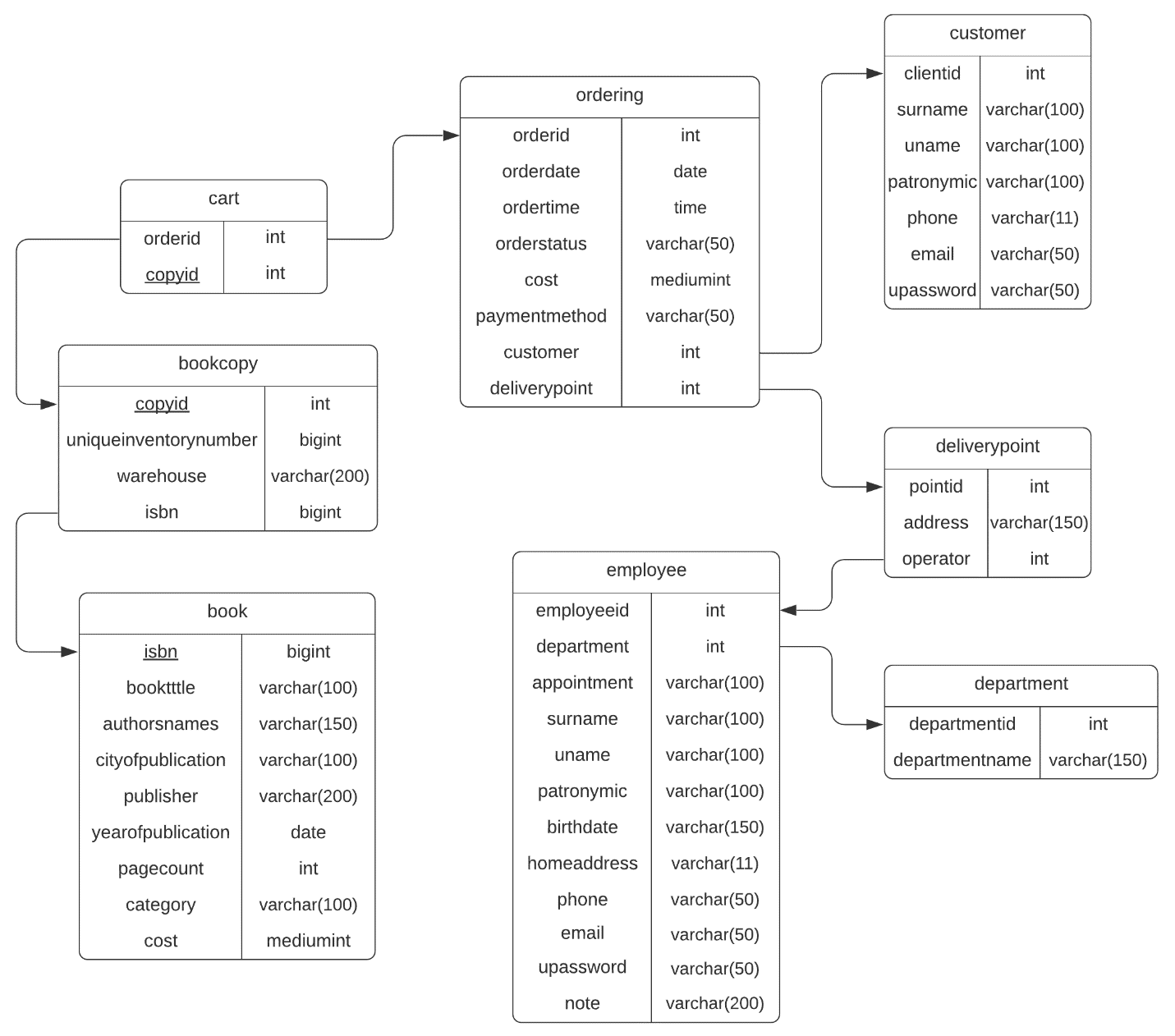
Определение внешних ключей



Реализация связей типа «многие-ко-многим»

Связи многие-ко-многим нет. Реализация не требуется.

Графическое изображение концептуальной модели предметной области



# 4. НОРМАЛИЗАЦИЯ БАЗЫ ДАННЫХ

Перед реализацией уже самой базы данных в СУБД необходимо привести ее в 3-ю нормальную форму.

Первая нормальная форма: сущность находится в первой нормальной форме (1НФ), если все ее атрибуты можно рассматривать как неделимые элементы. В этой БД уже все сущности являются неделимыми, поэтому мы уже имеем 1НФ.

Вторая нормальная форма: сущность находится во 2НФ, если она находится в 1НФ и каждый неключевой атрибут функционально полно зависит от составного ключа. Данное условие также выполняется.

Третья нормальная форма: сущность находится в 3НФ, если она находится во 2НФ и отсутствует транзитивная зависимость каждого не ключевого атрибута от первичного ключа. Условие выполняется.

БД уже находится в 3НФ, поэтому никаких преобразований сделать не требуется.

# 5. РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ «bookshop»

Создание базы данных:

create database bookshop;

Создание таблицы customer:

use bookshop;

create table customer

(clientid int not null auto\_increment,

surname varchar(100) null,

uname varchar(100) not null,

patronymic varchar(100) null,

phone varchar(11) not null,

email varchar(50) not null,

upassword varchar(50) not null,

constraint customerpk primary key (clientid),

unique (phone),

unique (email));

Создание таблицы book:

use bookshop;

create table book

(isbn bigint not null,

booktitle varchar(100) not null,

authorsnames varchar(150) null,

cityofpublication varchar(100) null,

publisher varchar(200) not null,

yearofpublication date not null,

pagecount int not null,

category varchar(100) not null,

cost mediumint not null,

constraint bookpk primary key (isbn),

check (isbn > 0),

check (pagecount > 0),

check (cost >= 0));

Создание таблицы bookcopy:

use bookshop;

create table bookcopy

(copyid int not null,

uniqueinventorynumber bigint not null,

warehouse varchar(200) not null,

isbn bigint not null,

constraint bookcopypk primary key (copyid),

unique (uniqueinventorynumber),

constraint copybookfk

foreign key (isbn) references book (isbn));

Создание таблицы deliverypoint:

use bookshop;

create table deliverypoint

(pointid int not null auto\_increment,

address varchar(150) not null,

operator int not null,

constraint deliverypointpk primary key (pointid),

unique (address),

constraint deliverypointemployeefk

foreign key (operator) references employee (employeeid));

Создание таблицы department:

use bookshop;

create table department

(departmentid int not null,

departmentname varchar(150) not null,

constraint departmentpk primary key (departmentid),

unique (departmentname));

Создание таблицы employee:

use bookshop;

create table employee

(employeeid int not null,

department int not null,

appointment varchar(100) not null,

surname varchar(100) not null,

uname varchar(100) not null,

patronymic varchar(100) null,

birthdate date not null,

homeaddress varchar(150) null,

phone varchar(11) not null,

email varchar(50) not null,

upassword varchar(50) not null,

note varchar(200) null,

constraint employeepk primary key (employeeid),

unique (email),

unique (phone),

constraint employeedepartmentfk

foreign key (department) references department (departmentid));

Создание таблицы ordering:

use bookshop;

create table ordering

(orderid int not null auto\_increment,

orderdate date not null,

ordertime time not null,

orderstatus varchar(50) not null,

cost mediumint not null,

paymentmethod varchar(50) not null,

customer int not null,

deliverypoint int not null,

constraint ordering primary key (orderid),

check (cost >= 0),

constraint ordercustomerfk

foreign key (customer) references customer (clientid),

constraint orderdeliverypointfk

foreign key (deliverypoint) references deliverypoint (pointid));

Создание таблицы cart:

use bookshop;

create table cart

(orderid int not null,

copyid int not null,

constraint cartorderfk

foreign key (orderid) references ordering (orderid),

constraint cartcopybookfk

foreign key (copyid) references bookcopy (copyid),

unique (copyid));

Создание таблицы administrator (вспомогательная таблица, ее нет в диаграммах, поскольку она не связана с другими и хранит лишь одну запись):

use bookshop;

create table administrator

(email varchar(50) not null,

upassword varchar(50) not null,

constraint administrator primary key (email));

Заполнение таблиц:

use bookshop;

insert into book

values

(9785171333249, 'Манифест коммунистической партии', 'Маркс К., Энгельс Ф.', 'Ульяновск', 'Эксмо', '2010', 96, 'Философия', 184),

(9785171272029, 'За миллиард лет до конца света', 'Стругацкий А., Стругацкий Б.', 'Москва', 'АСТ', '2011', 256, 'Научная фантастика', 228),

(9785171733241, 'Пикник на обочине', 'Стругацкий А., Стругацкий Б.', 'Санкт-Петербург', 'АСТ', '2015', 256, 'Научная фантастика', 228),

(9785170801091, 'Заводной апельсин', 'Берджесс Э.', 'Москва', 'АСТ', '2020', 252, 'Классика', 250),

(9785170800858, 'О дивный новый мир', 'Хаксли О.', 'Москва', 'АСТ', '2020', 350, 'Классика', 220),

(9785389197541, 'Дневник писателя', 'Достоевский Ф.', 'Санкт-Петербург', 'Азбука', '2021', 1088, 'Классика', 584),

(9785041034979, 'Десять негритят', 'Кристи А.', 'Москва', 'Эксмо', '2020', 288, 'Детектив', 206),

(9785170846214, 'Коралина', 'Гейман Н.', 'Санкт-Петербург', 'АСТ', '2020', 192, 'Для детей', 459);

use bookshop;

insert into customer (surname, uname, patronymic, phone, email, upassword)

values

('Седов', 'Яков', 'Саввич', '89194561122', 'sedoy123@mail.ru', '126'),

(null, 'Александр', null, '11231234556', 'al12a@gmail.com', 'aaa4'),

('Сидорова', 'Алина', null, '74568884556', 'lina11@gmail.com', '785d!s');

use bookshop;

insert into department

values

(4, 'Администрация'),

(6, 'Курьерская служба'),

(25, 'Складская служба');

use bookshop;

insert into employee

values

(254, 6, 'Оператор пункта выдачи', 'Сергеев', 'Тимур', 'Макарович', '2010-03-04', 'г. Казань, ул. Космонавтов, 2', '89654442255', 'tim211@mail.ru', 'q1w1', null),

(111, 4, 'Администратор', 'Фадеева', 'Вероника', 'Вячеславовна', '2000-05-21', 'г. Казань, пр-кт Победы, 172', '85221237769', '123vera123@gmail.com', '1', null),

(255, 6, 'Оператор пункта выдачи', 'Филатов', 'Степан', 'Богданович', '1999-10-10', 'г. Казань, Кремлевская, 33', '84771457852', 'st11@gmail.com', '1455', null);

use bookshop;

insert into bookcopy

values

(21, 5161555, 'г. Казань, ул. Баумана, 36', 9785171333249),

(56, 5181144, 'г. Казань, ул. Баумана, 36', 9785171333249),

(12, 1111155, 'г. Казань, Московская, 20', 9785171733241);

use bookshop;

insert into deliverypoint (address, operator)

values

('г. Казань, ул. Гагарина, 93', 254)

('г. Казань, ул. Гагарина, 95', 255);

use bookshop;

insert into administrator

values

('admin@mail.ru', '1234');

use bookshop;

insert into ordering (orderdate, ordertime, orderstatus, cost, paymentmethod, customer, deliverypoint)

values

(curdate(), curtime(), 'Завершен', 1555, 'Онлайн', 2, 2),

(curdate(), curtime(), 'Завершен', 525, 'Наличные', 2, 1),

(curdate(), curtime(), 'Отменен', 200, 'Картой', 1, 2),

(curdate(), curtime(), 'Завершен', 1021, 'Онлайн', 1, 2);

# 6. ЗАПРОСЫ К БАЗЕ ДАННЫХ «bookshop»

В базу данных добавлено некоторое количество запросов, некоторые из них:

1. Авторизация пользователя.

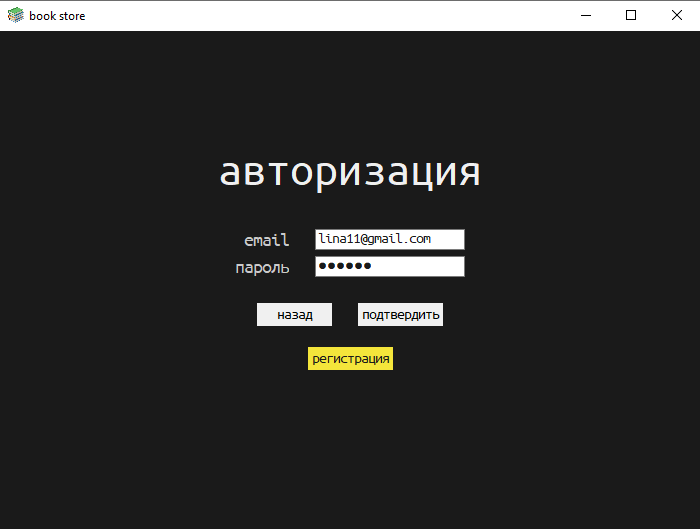
Команда SQL:

SELECT \* FROM %1 WHERE (email = :email AND upassword = :upassword);

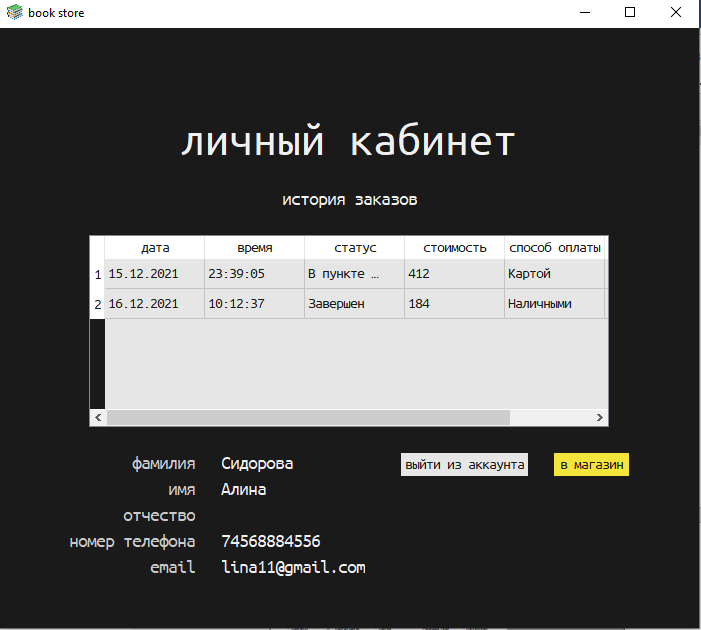
%1 – имя таблицы, в которой осуществляется поиск пользователя (customer/employee/administrator)

Результат выполнения:

Исходное состояние с введенными паролем и email:



Получаем:



2. Регистрация нового сотрудника.

Команда SQL:

INSERT INTO employee (employeeid,

department, appointment, surname,

uname, patronymic, birthdate,

homeaddress, phone, email,

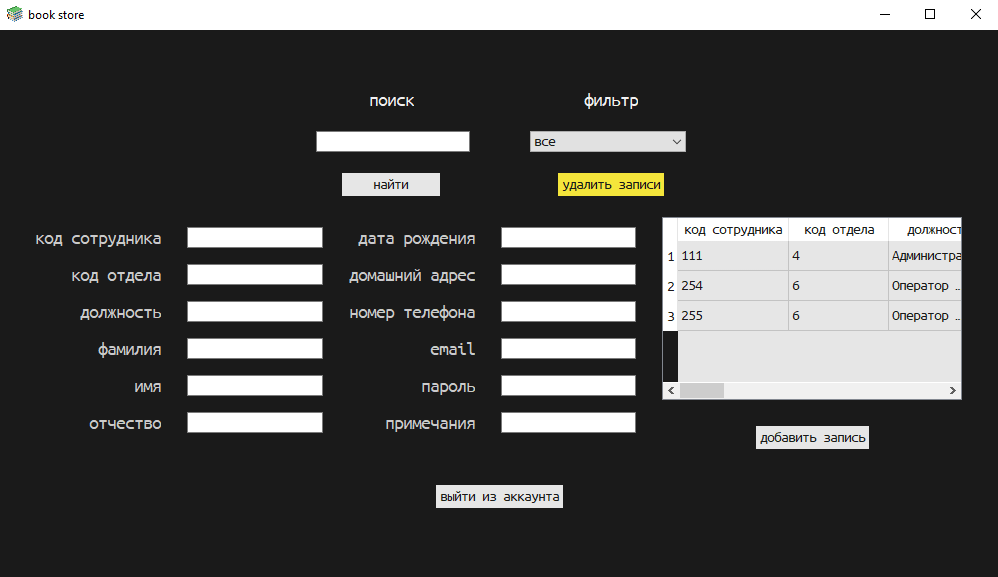
upassword, note)

VALUES (%1, %2, '%3', '%4', '%5', '%6',

'%7', '%8', '%9', '%10', '%11', '%12');

Результат выполнения:

Исходное состояние:



Заполняем поля и нажимаем на «добавить запись»:



3. Поиск книги по авторам.

Команда SQL:

SELECT \* FROM book WHERE isbn LIKE '%%1%' OR

booktitle LIKE '%%1%' OR

authorsnames LIKE '%%1%' OR

cityofpublication LIKE '%%1%' OR

publisher LIKE '%%1%' OR

yearofpublication LIKE '%%1%' OR

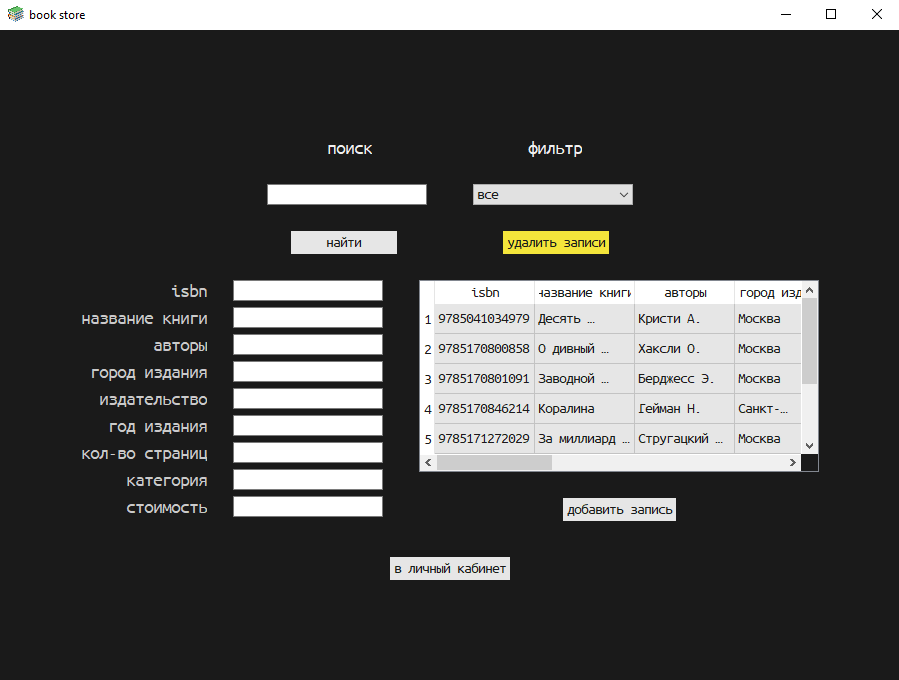
pagecount LIKE '%%1%' OR

category LIKE '%%1%' OR

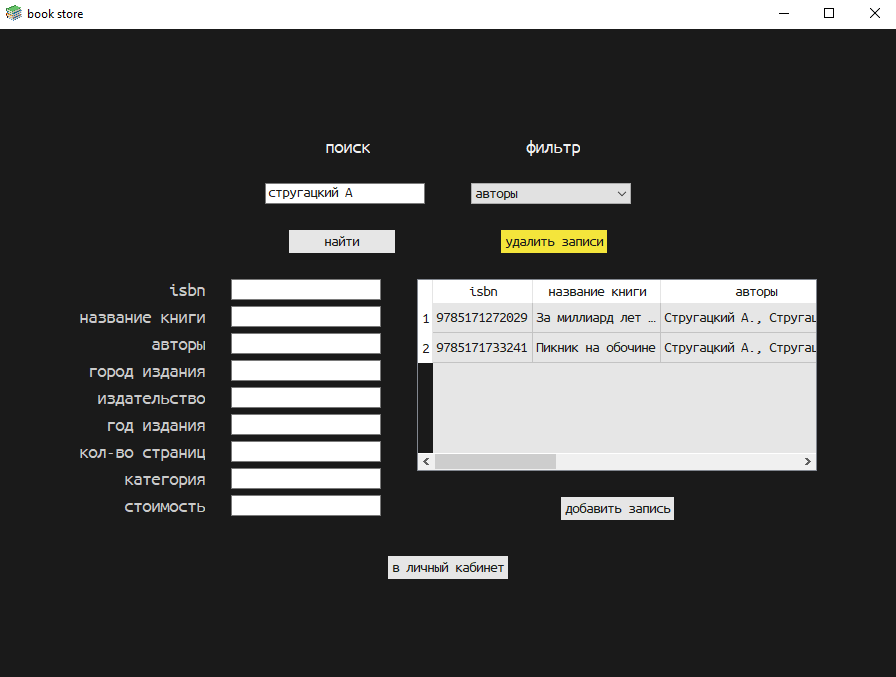
cost LIKE '%%1%';

Результат выполнения:

Исходное состояние:



Заполним поле «поиск», выберем в «фильтре» нужный нам столбец и нажмем на «найти»:



4. Отобразить все заказы со статусом «В пункте выдачи», которые находятся в данном пункте выдачи.

Команда SQL:

SELECT orderid AS 'код заказа',

orderdate AS 'дата',

ordertime AS 'время',

orderstatus AS 'статус',

cost AS 'стоимость',

paymentmethod AS 'способ оплаты',

customer.clientid AS 'код клиента',

customer.uname AS 'имя клиента',

customer.email AS 'email клиента'

FROM ordering JOIN customer JOIN deliverypoint

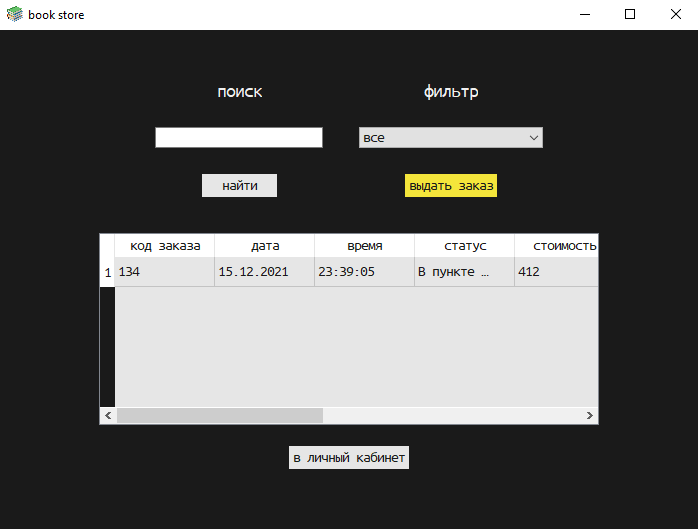
ON (ordering.customer = customer.clientid

AND ordering.deliverypoint = deliverypoint.pointid)

WHERE deliverypoint.address = '%1'

AND ordering.orderstatus = 'В пункте выдачи';

Результат выполнения:



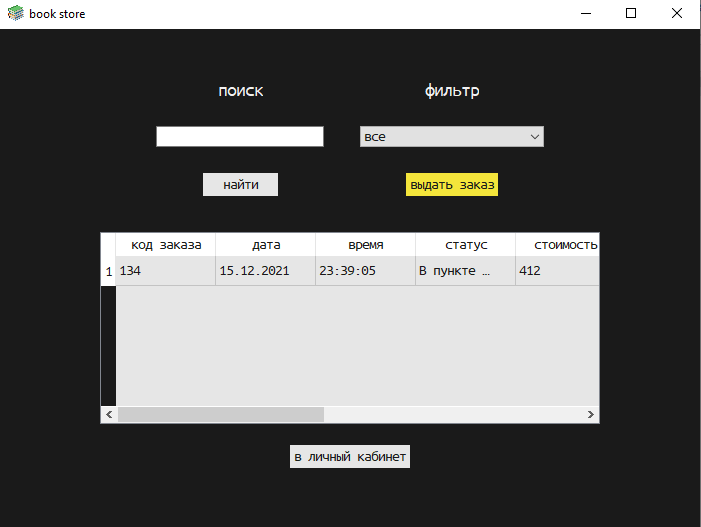
5. Обновить статус заказа, который оператор должен выдать клиенту.

Команда SQL:

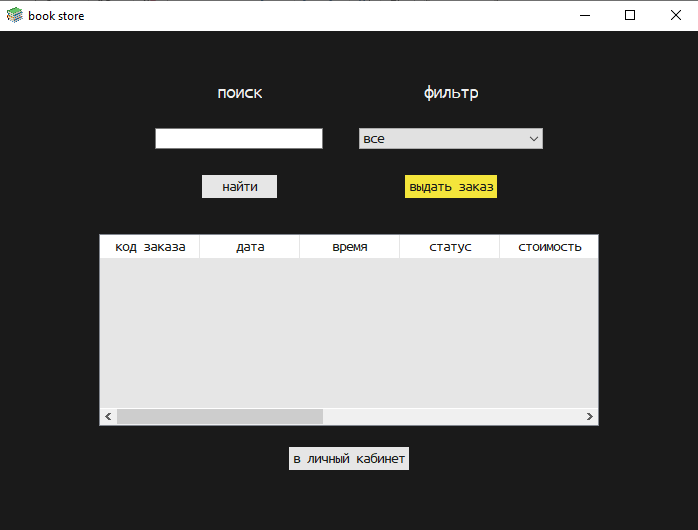
UPDATE ordering SET orderstatus = 'Завершен' WHERE orderid = %1;

Результат выполнения:

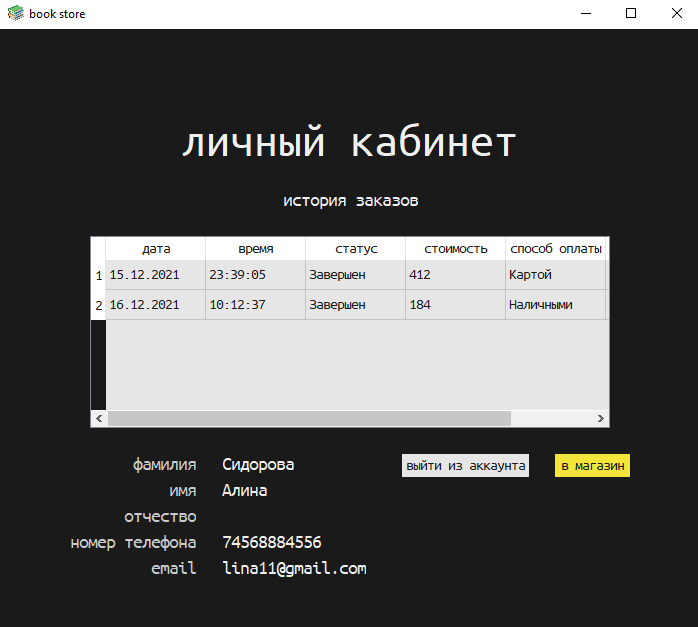
Исходное состояние:



Нажмем на «выдать заказ»:



Зайдем за этого клиента (код клиента - 3) и посмотрим статус этого заказа (код заказа – 134, стоимость – 412):



# 7. РУКОВОДСТВО ПРОГРАММИСТА

Для разработки проекта была использована среда разработки программного обеспечения Qt Creator 5.0.3.

Программный код разделен на .h и .cpp файлы. Эти файлы, в свою очередь, помещены в папки, которые структурируют их по функциональности.

Файл с расширением .pro – файл сборки проекта.

В проекте происходит соединение с локальным сервером, в котором хранится «bookshop», приложение обращается к базе данных через драйвер «MySQL».

«bookshop» разработана с использованием СУБД «MySQL».

Поскольку программа логически делится на состояния, ниже будут приведены лишь те .h файлы классов, которые отражают состояния (классы графического интерфейса ниже приведены не будут).

Состояние «Администратор»:

class AdministratorsWorkBench : public State

{

Q\_OBJECT

public:

AdministratorsWorkBench();

~AdministratorsWorkBench();

void setThisStateAsCurrent(QList<QVariant> data);

private slots:

void administratorExitsFromEmployeesSignUp();

private:

EmployeesSignUp \*employeesSignUpView;

QWidget \*currentView;

void changeView(QWidget \*view);

};

Состояние «Клиент»:

class ClientsWorkBench : public State

{

Q\_OBJECT

public:

ClientsWorkBench();

~ClientsWorkBench();

void setThisStateAsCurrent(QList<QVariant> data);

private slots:

void clientWantsToShop();

void clientExitsFromHomePage();

void orderIsMade(int orderId);

void backFromClientShopIsChosen();

void backFromClientPaymentIsChosen(bool orderWasPaidFor);

private:

void changeView(QWidget \*view);

private:

ClientsHomePage \*clientsHomePageView;

ClientsShopPage \*clientsShopPageView;

ClientPayment \*clientPaymentView;

QWidget \*currentView;

};

Состояние «Оператор пункта выдачи»:

class DPOperatorsWorkBench : public State

{

Q\_OBJECT

public:

DPOperatorsWorkBench();

~DPOperatorsWorkBench();

void setThisStateAsCurrent(QList<QVariant> data);

private slots:

void operatorExitsFromHomePage();

void operatorWantsToGoToDP(QString DPAdress);

void backFromDPIsChosen();

private:

void changeView(QWidget \*view);

private:

DPOperatorsHomePage \*dpOperatorsHomePageView;

DP \*dpView;

QWidget \*currentView;

};

Состояние «Вход в систему»:

class Entrance : public State

{

Q\_OBJECT

public:

Entrance();

~Entrance();

void setThisStateAsCurrent(QList<QVariant> data);

private slots:

void userRoleIsChosen(int userRole);

void clientSignUpIsChosen();

void backFromSignInIsChosen();

void backFromClientSignUpIsChosen();

void userSignedInSystem();

private:

void changeView(QWidget \*view);

private:

int userRole;

UserChoice \*userChoiceView;

SignIn \*signInView;

ClientSignUp \*clientSignUpView;

QWidget \*currentView;

};

Состояние «Кладовщик»:

class StoreKeepersWorkBench : public State

{

Q\_OBJECT

public:

StoreKeepersWorkBench();

~StoreKeepersWorkBench();

void setThisStateAsCurrent(QList<QVariant> data);

private slots:

void storeKeeperExitsFromHomePage();

void storeKeeperWantsToGoToWarehouse();

void storeKeeperWantsToGoToBooks();

void backFromSKWarehouseIsChosen();

void backFromSKBooksIsChosen();

private:

void changeView(QWidget \*view);

private:

StoreKeepersHomePage \*storeKeepersHomePageView;

StoreKeepersBooks \*booksView;

SKWarehouse \*warehouseView;

QWidget \*currentView;

};

Абстрактный класс State, от которого следует наследоваться, если дочерний класс хочет работать с классом StateManager (приведен ниже):

class State : public QObject

{

Q\_OBJECT

public:

~State() = default;

virtual void setThisStateAsCurrent(QList<QVariant> data) = 0;

signals:

void stateWasFinished(int newStateId, QList<QVariant> data);

protected:

State() = default;

};

Класс StateManager, который переключает состояния:

class StateManager : public QObject

{

Q\_OBJECT

public:

StateManager() = delete;

explicit StateManager(int initialStateId);

~StateManager();

private slots:

void stateIsFinished(int newStateId, QList<QVariant> data);

private:

static State \*createState(int stateId);

private:

QMap<int, State \*> states;

int currentStateId;

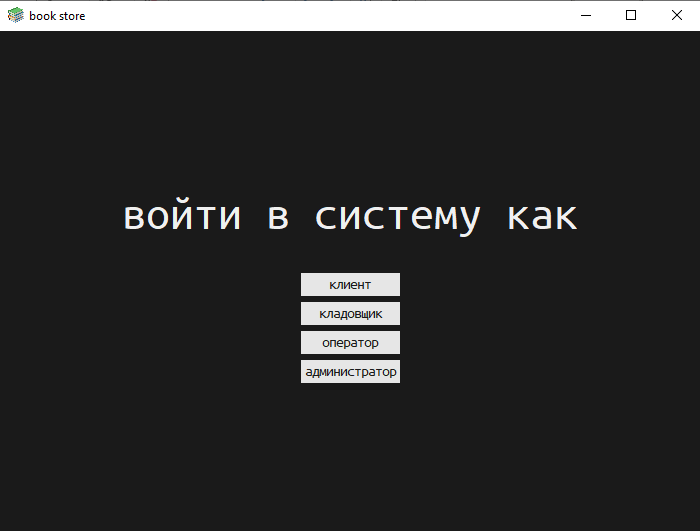
};

# 8. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

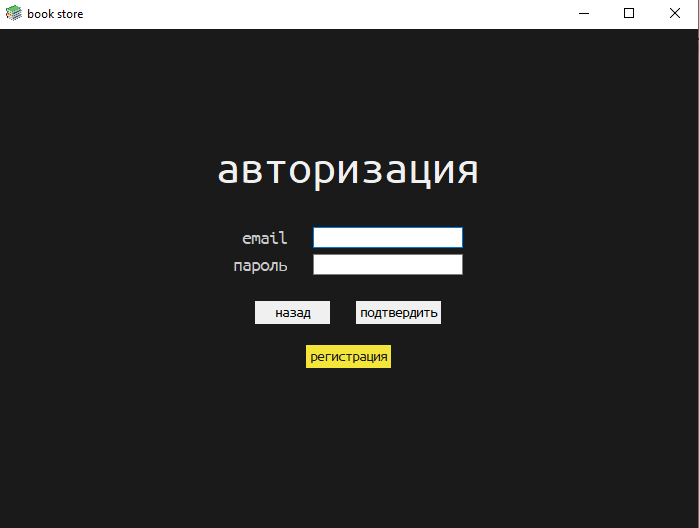
Для запуска приложения необходимо нажать на «Запустить» (в IDE Qt Creator):



Перед вами отобразится окно:

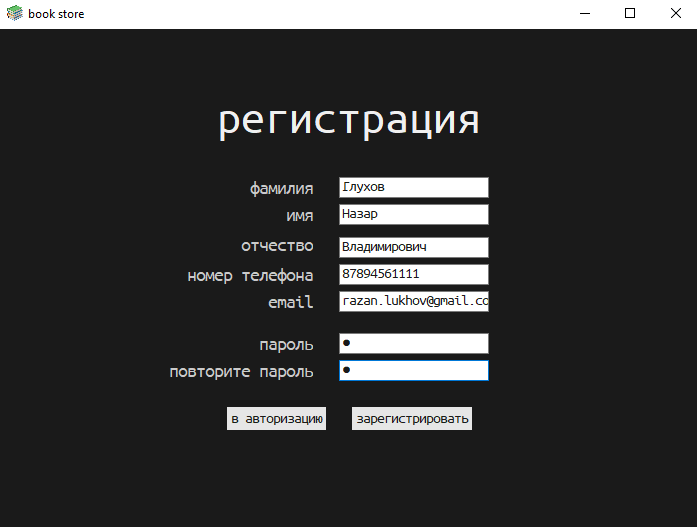


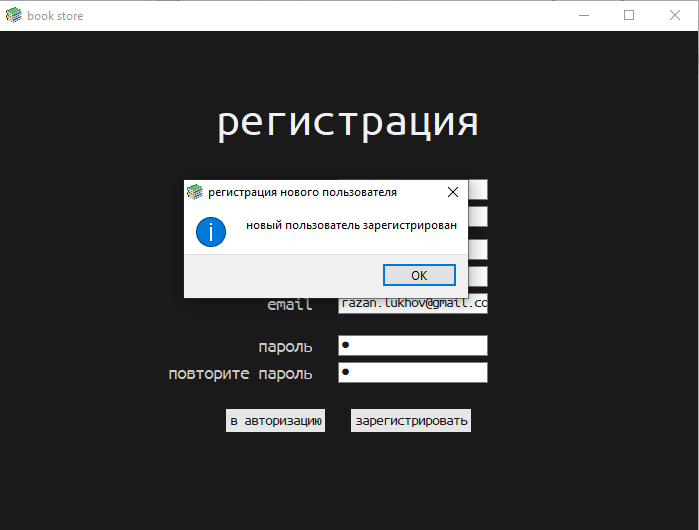
Здесь пользователь может выбрать роль, выбираем «клиент»:



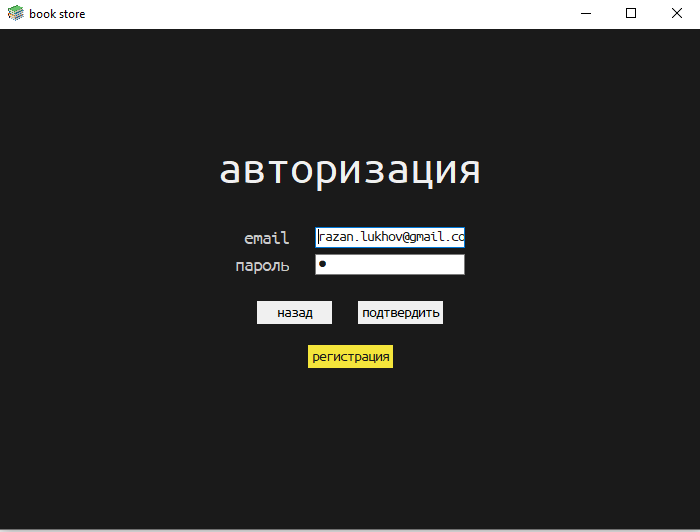
Такое же окно будет у других ролей, однако только у «клиента» есть возможность зарегистрироваться.

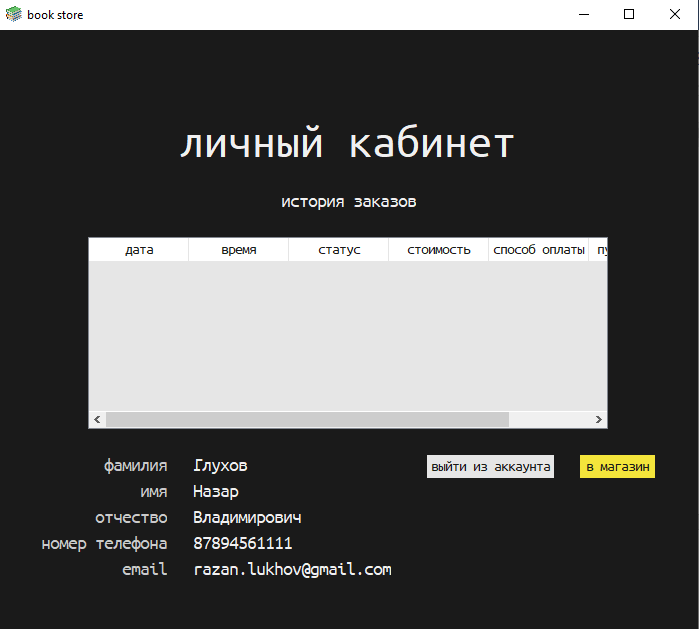
Нажмем на «регистрация», введем необходимые данные и зарегистрируемся:



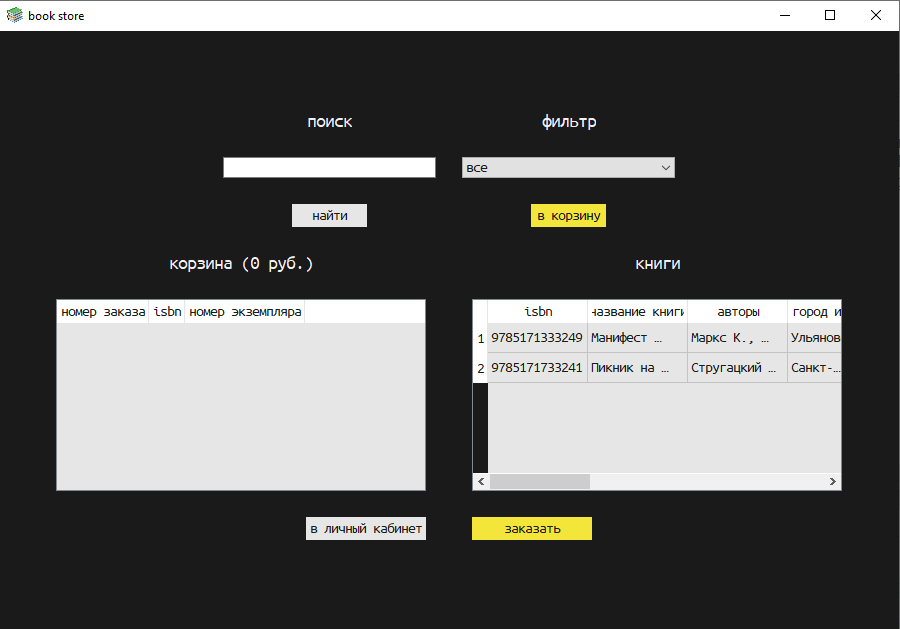


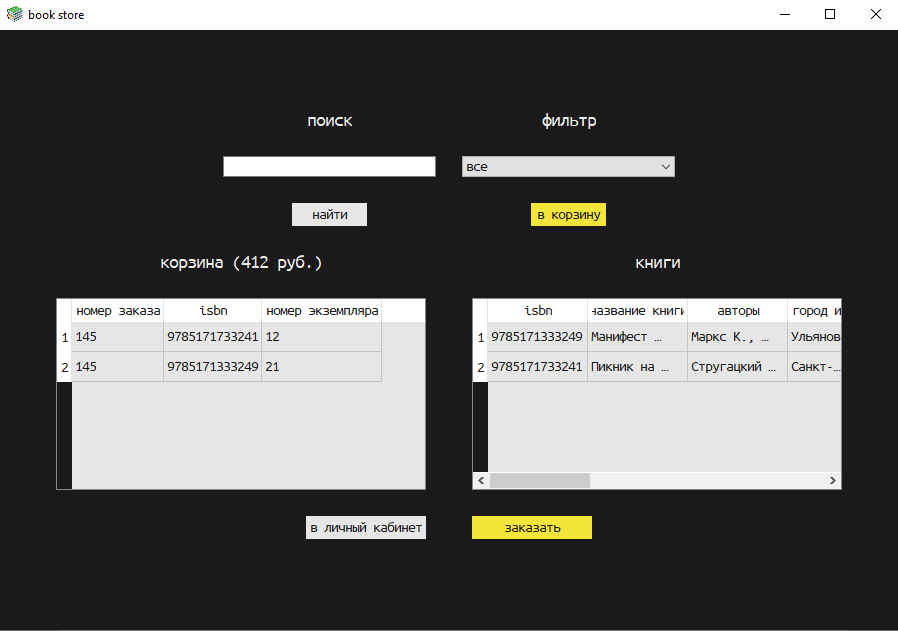
Вернемся к авторизации, введем наши email и пароль, нажмем на «подтвердить»:



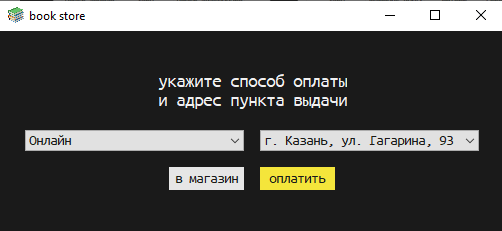


Давайте зайдем в магазин и закажем пару книг (фильтр применятся сразу, при нажатии на «в корзину», все книги, отображаемые в таблице «книги», добавятся в корзину в количестве 1 штука):

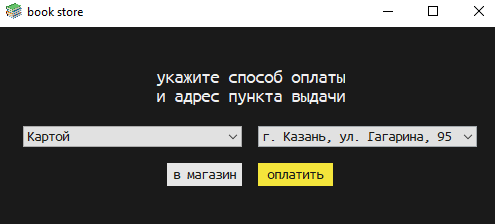




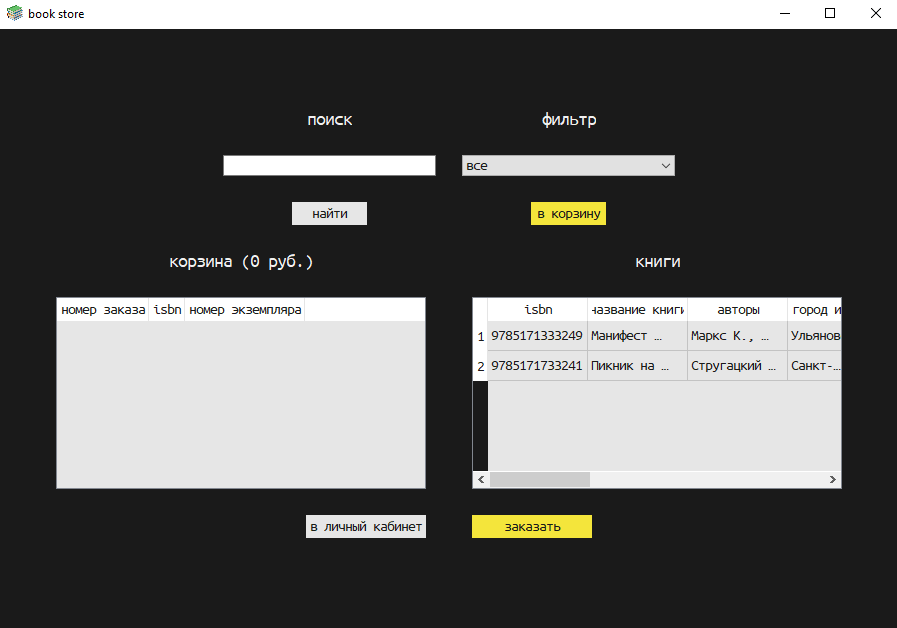
Закажем книги (кнопка «заказать»):

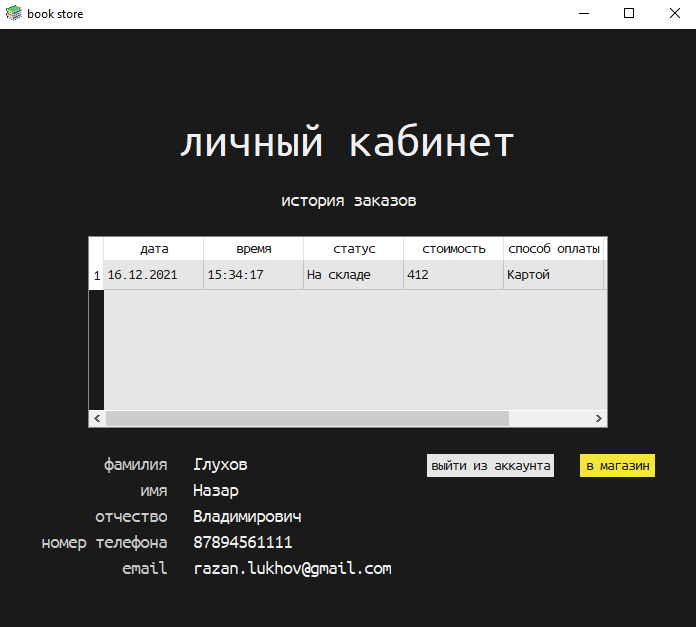


Выберем нужные способ оплаты и адрес пункта выдачи, нажмем на «оплатить»:

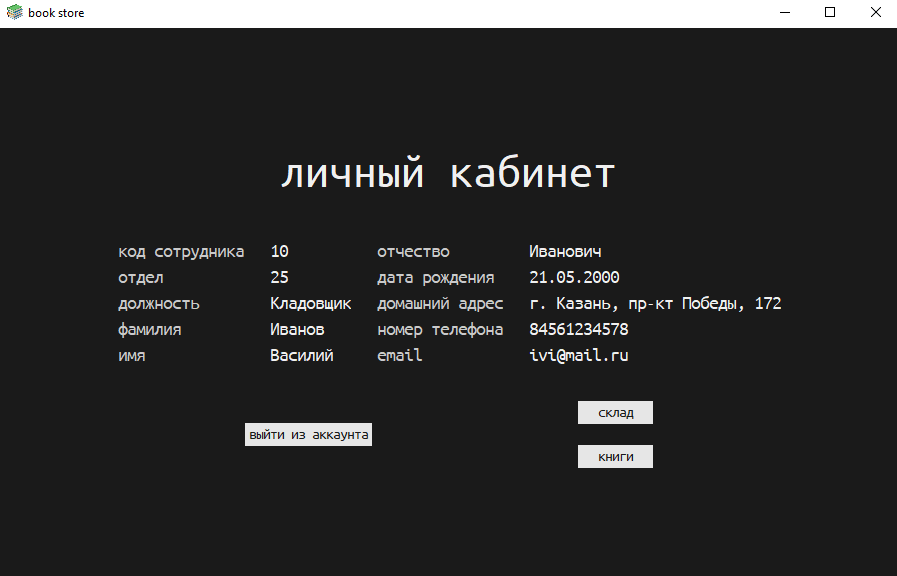


Нас перекинет обратно в магазин, выйдем в личный кабинет, посмотрим на наш заказ:

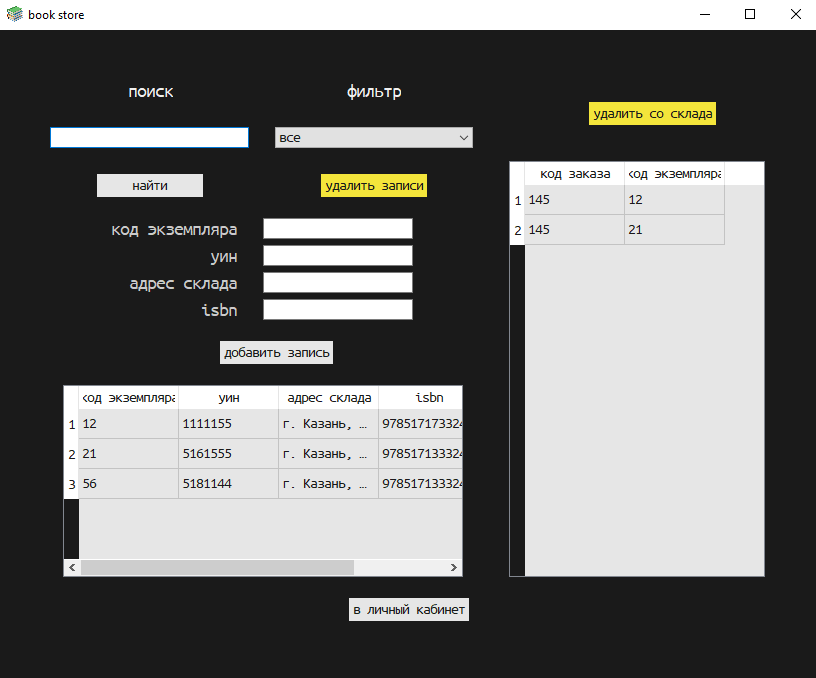




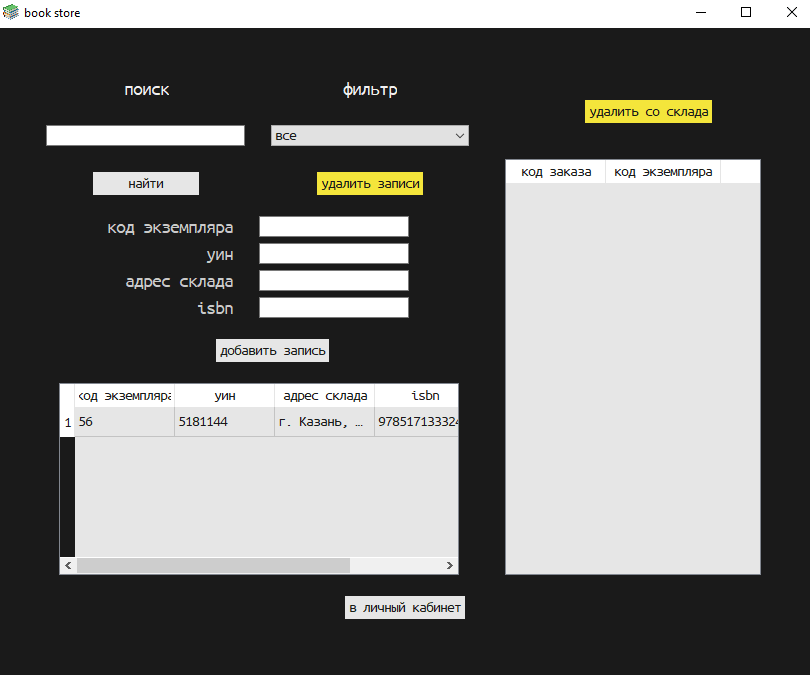
Наш заказ «На складе», зайдем за кладовщика:



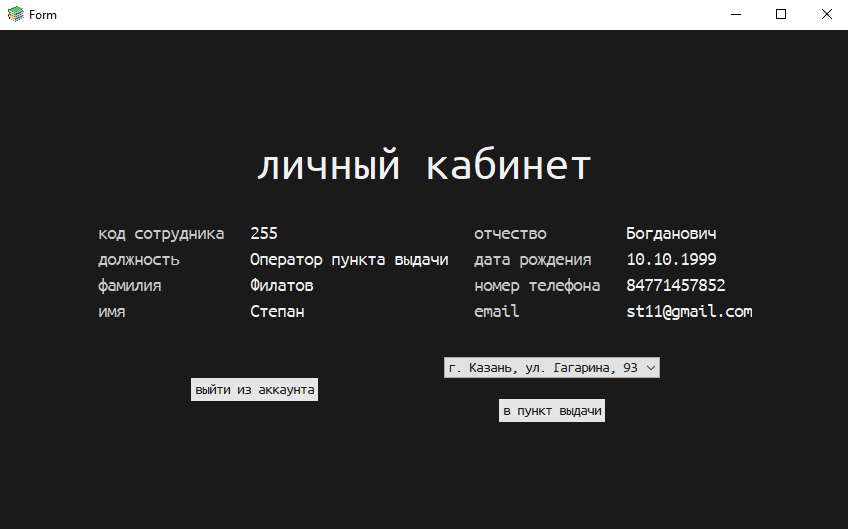
Здесь кладовщик может добавить новые книги (перейдя в соответствующее окно нажатием на «книги»), нажмем на «склад»:



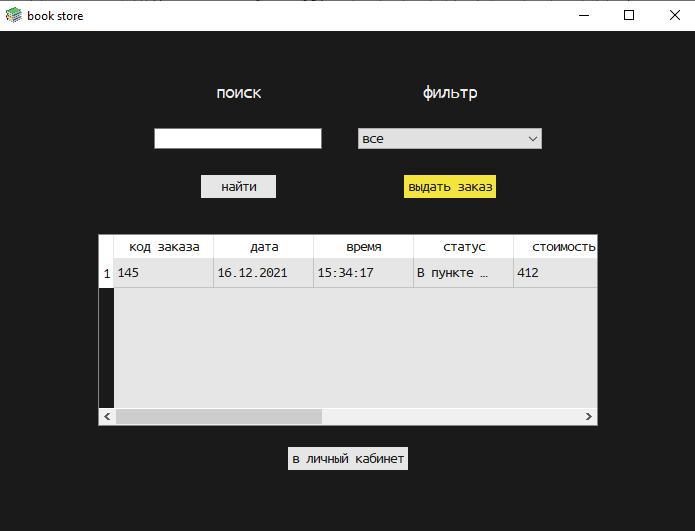
Здесь кладовщик может изменять данные об экземплярах на складе, также кладовщик обязан выдать заказ курьеру, выдадим заказ, нажмем на «удалить со склада», в этот момент информация об экземплярах удаляется из таблицы bookcopy:



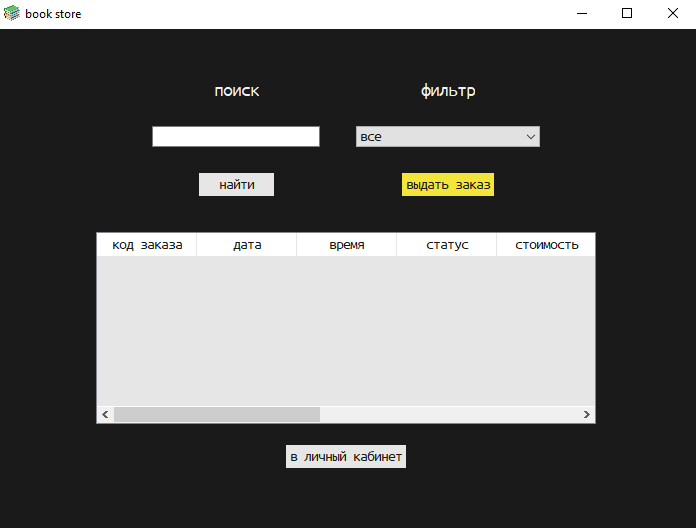
Нажали, заказ телепортируется в пункт выдачи, зайдем за оператора, выдадим наш заказ нам же:



Перейдем в наш пункт заказа:



Здесь оператор может находить нужные заказы и выдавать их:



Все, заказ завершен, если мы перейдем в наш личный кабинет, заказ будет помечен как «Завершен».