

Федеральное государственное автономное образовательное учреждение  
высшего образования «Санкт-Петербургский национальный  
исследовательский университет информационных технологий, механики и  
оптики»

Факультет Программной Инженерии и Компьютерной Техники

Дисциплина: Информационные системы и базы данных

## Курсовая работа

### Этап №1

Выполнили: Тучин Артём Евгеньевич

Емельянов Дмитрий Сергеевич

Группа: Р33111

Преподаватель: Харитонов Анастасия Евгеньевна

Г. Санкт-Петербург, 2023г.

## **Предметная область**

Приложение для ремонта бытовой техники. Упрощает процесс ремонта (в приложении выбирается поломка и желаемая дата, далее сервис сам подбирает мастера, также используется модель подписки - не нужно платить за каждый ремонт). Возможна и оплата конкретных заказов. Две группы пользователей - мастера и владельцы техники.

## **Бизнес-процессы**

### Для владельцев техники

1. Оформление подписки (аналог страхования для техники на определенный срок. Включает в себя тех. осмотр, ремонт и рекомендации по эксплуатации от мастера)
2. Создание заказа (в случае поломки техники пользователь оставляет заказ, под который автоматически выбирается мастер. В заказе указывается тип товара(напр. кухонный), название, описание поломки и дата)
3. Создание заказа с выбором мастера (на специальной вкладке можно выбрать конкретного мастера и создать заказ, вызвав его)
4. Просмотр статуса заказа (новый, выполняется, завершен, завершенные заказы можно повторять и оценивать)
5. Оставление отзыва (после получения услуг по ремонту техники владелец может написать отзыв о мастере, который оказывал услугу. В отзыве можно оценить мастера по 5-ти бальной шкале, а так же написать комментарий)
6. Поиск ответов на частые вопросы (во вкладке Q&A)

### Для мастеров

1. Составление и просмотр своего расписания (У каждого мастера есть расписание рабочих дней, для реализации поиска мастера для заказа)
2. Просмотр текущих заказов (мастер должен знать какие-заказы ему предстоит выполнить)
3. Просмотр истории заказов (мастер может посмотреть прошлые заказы для просмотра отзывов, проверки расчетов по заказу)
4. Прием/отклонение заказов (при автоматическом определении мастера к заказу мастер получает уведомление об этом с возможностью отказаться от выполнения заказа, тогда системой будет подобран другой мастер)
5. Просмотр и редактирование своего профиля (в профиле указаны данные о мастере, его стаж работы, квалификации, фотография)

## **Сущности**

1. Человек (личные данные о человеке: ФИО, телефон, эл. почта)
2. Заказы (информация о заказах: дата, прибор, мастер, цена, статус)
3. Тип подписки (виды подписок в приложении)
4. Мастер (Информация о мастере: стаж, квалификация, рейтинг, фото)
5. Клиент (Информация о клиенте: адрес)
6. Тип техники (тип бытовой техники, на которые распространяются услуги)
7. Логин, пароль(шифрованные)
8. Вопросы – Ответы (данные для соответствующего функционала приложения)
9. Отзывы (информация об отзывах: от кого, кому, комментарий, рейтинг, заказ)
10. Заявки в поддержку (Информация о запросах: отправитель, тема, сообщение, дата)
11. Способы оплаты (Способы оплаты, поддерживаемые в приложении)
12. Подписка у пользователя (информация о подписках пользователей: тип подписки, пользователь, дата оформления, дата отключения)

13. Техника пользователя (информация о приборах у пользователей: тип прибора, пользователь, дата покупки)
14. Дни (дни недели, время работы)
15. Расписание (отношение между днями и мастерами)

Федеральное государственное автономное образовательное учреждение  
высшего образования «Санкт-Петербургский национальный  
исследовательский университет информационных технологий, механики и  
оптики»

Факультет Программной Инженерии и Компьютерной Техники

Дисциплина: Информационные системы и базы данных

## Курсовая работа

### Этап №2

Выполнили: Тучин Артём Евгеньевич

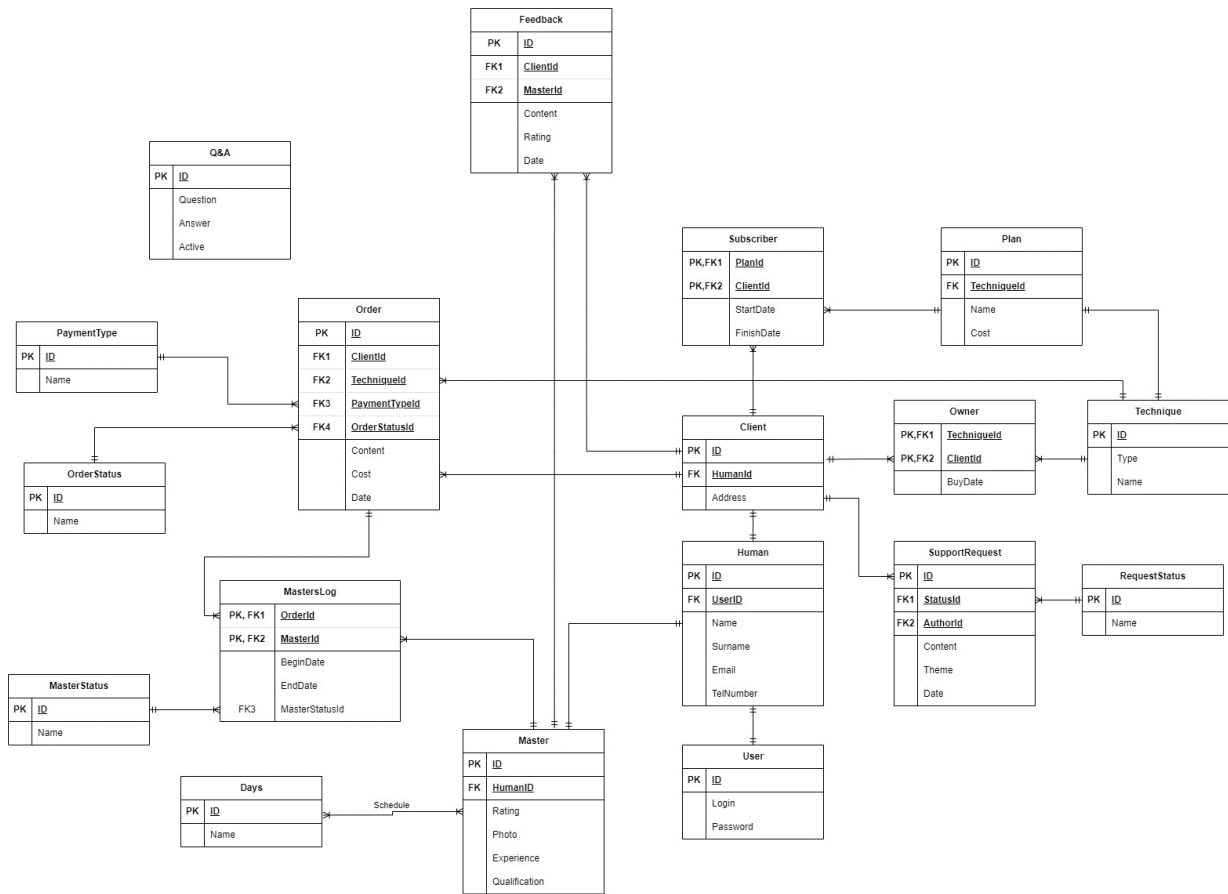
Емельянов Дмитрий Сергеевич

Группа: Р33111

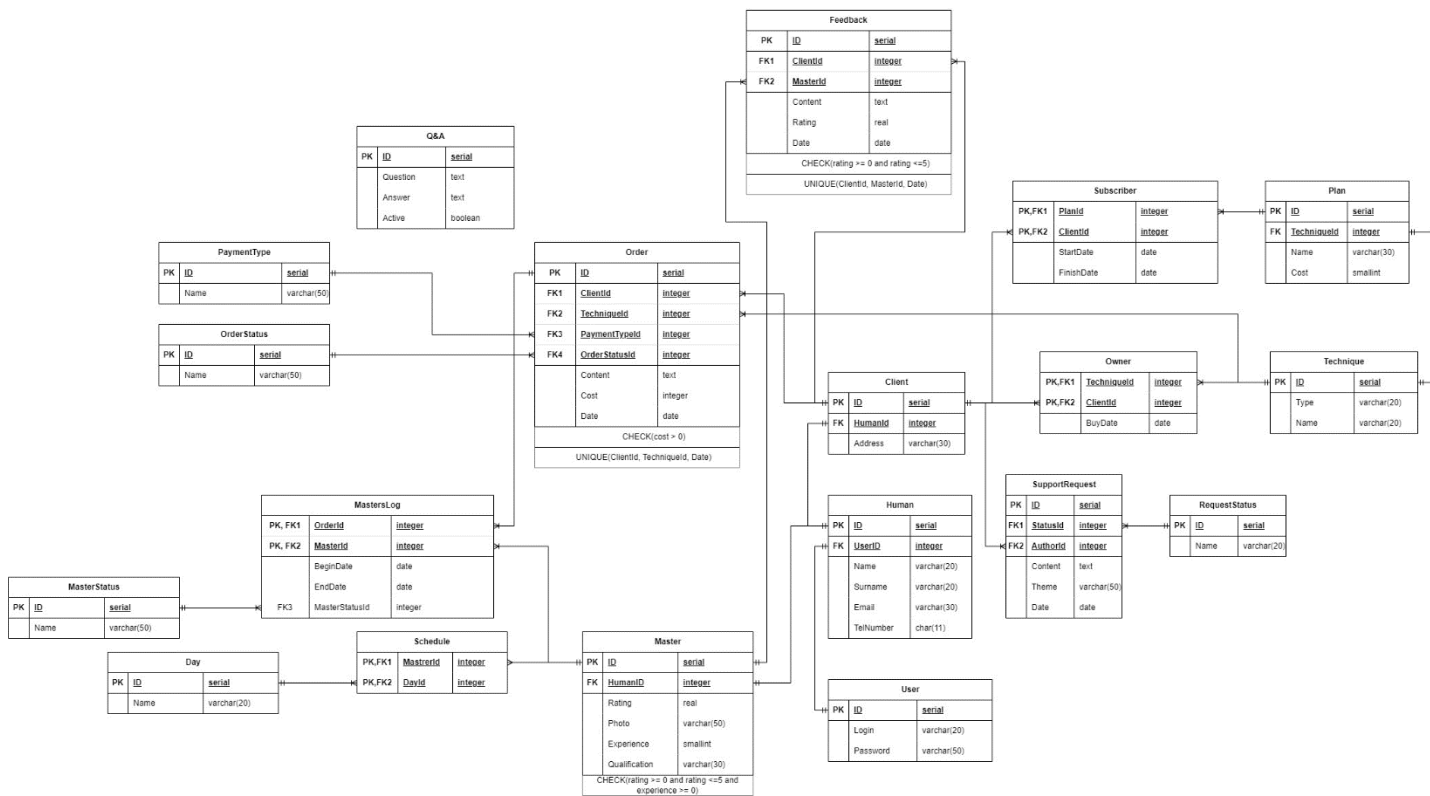
Преподаватель: Харитонова Анастасия Евгеньевна

Г. Санкт-Петербург, 2023г.

Инфологическая модель



Даталогическая модель



Федеральное государственное автономное образовательное учреждение  
высшего образования «Санкт-Петербургский национальный  
исследовательский университет информационных технологий, механики и  
оптики»

Факультет Программной Инженерии и Компьютерной Техники

Дисциплина: Информационные системы и базы данных

## Курсовая работа

### Этап №3

Выполнили: Тучин Артём Евгеньевич

Емельянов Дмитрий Сергеевич

Группа: Р33111

Преподаватель: Харитонов Анастасия Евгеньевна

Г. Санкт-Петербург, 2023г.

## Таблицы в бд

```
create table accounts(  
    id serial primary key,  
    login varchar(20) not null unique,  
    password varchar(50) not null,  
    active boolean not null default true  
);  
  
create table human (  
    id serial primary key,  
    userId integer references accounts(id),  
    name varchar(20) not null,  
    surname varchar(20) not null,  
    email varchar(30) not null unique,  
    telNumber char(11) not null unique);  
create table master (  
    id serial primary key,  
    humanId integer not null references human(id),  
    rating real default 0,  
    photo varchar(50) unique,  
    experience smallint not null,  
    qualification varchar(100) not null,  
    check(rating >= 0 and rating <= 5 and experience >= 0)  
);  
  
create table day (  
    id serial primary key,  
    name varchar(20) unique not null,  
    shortName char(2) unique not null  
);  
  
create table schedule (  
    masterId integer references master(id),  
    dayId integer references day(id),  
    primary key(masterId, dayId)  
);  
  
create table qa (  
    id serial primary key,  
    question text not null unique,  
    answer text not null,  
    active boolean not null default false  
);  
  
create table techniqueType(  
    id serial primary key,  
    name varchar(20) not null unique  
);  
  
create table technique(  
    id serial primary key,  
    typeId integer references techniqueType(id) not null,  
    name varchar(20) not null
```

```
);  
create table plan(  
    id serial primary key,  
    techniqueId integer unique references technique(id),  
    name varchar(30) not null unique,  
    cost smallint not null,  
    active boolean not null default true  
);
```

```
create table client(  
    id serial primary key,  
    humanId integer references human(id),  
    address varchar(30) not null  
);
```

```
create table owner(  
    techniqueId integer references technique(id),  
    clientId integer references client(id),  
    buyDate Date not null,  
    primary key(techniqueId, clientId)  
);
```

```
create table subscriber(  
    planId integer references plan(id),  
    clientId integer references client(id),  
    startDate Date not null,  
    finishDate Date not null,  
    primary key(planId, clientId),  
    check(startDate < finishDate)  
);
```

```
create table feedback(  
    id serial primary key,  
    clientId integer references client(id),  
    masterId integer references master(id),  
    content text not null,  
    rating real not null,  
    date Date not null  
);
```

```
create table orderStatus (  
    id serial primary key,  
    name varchar(50) not null unique  
);
```

```
create table paymentType (  
    id serial primary key,  
    name varchar(50) not null unique  
);
```

```
create table orders (  
    id serial primary key,  
    clientId integer not null references client(id),  
    techniqueId integer not null references technique(id),
```



```

        paymentTypeId integer not null references paymentType(id),
        orderStatusId integer not null references orderStatus(id),
        content text not null,
        cost integer not null check(cost > 0),
        date date not null,
        unique(clientId, techniqueId, date)
    );

create table masterStatus (
    id serial primary key,
    name varchar(50) not null unique
);

create table masterLog (
    orderId integer references orders(id),
    masterId integer references master(id),
    beginDate date not null,
    endDate date,
    masterStatusId integer not null references masterStatus(id),
    primary key (orderId, masterId)
);

create table requestStatus (
    id serial primary key,
    name varchar(20) not null unique
);

create table supportRequest (
    id serial primary key,
    statusId integer not null references requestStatus(id),
    authorId integer not null references client(id),
    content text not null,
    theme varchar(50) not null,
    date date not null
);

create table newOrders (
    masterId integer references master(id),
    orderId integer references orders(id),
    primary key(masterId,orderId)
);

```

### **Заполнение таблиц**

```

insert into accounts (login, password) values
('kapibara2018','qwerty123'),
('777terminator777','qpwofn2j_83'),
('the_best_master','12345'),
('the_best_client00','987732gj'),
('anton_ivanov1999','923hfhfg2'),
('user29','qwerty100');

```

```

insert into human (userId, name, surname, email, telNumber) values
(1,'Ivan','Jsonov','kapibara2018@mail.ru','89218882021'),

```

```
(2,'Vlad','Petrov','777terminator777@mail.ru','89002652956'),
(3,'Andrew','Mikhailov','the_best_master@mail.ru','89926750567'),
(4,'Vlad','Petrov','the_best_client00@mail.ru','89543214878'),
(5,'Anton','Ivanov','anton_ivanov1999@mail.ru','89225920982'),
(6,'John','Doe','user29@mail.ru','89010001010');
```

```
insert into master (humanId, experience, qualification) values
(1, 2, 'Курсы мастера по починке кухонной техники'),
(2, 1, 'Курсы мастера по починке микроволновок'),
(3, 10, 'Высшее образование в сфере починки кофемашин');
```

```
insert into day (name, shortName) values
('Понедельник','Пн'),
('Вторник','Вт'),
('Среда','Ср'),
('Четверг','Чт'),
('Пятница','Пт'),
('Суббота','Сб'),
('Воскресенье','Вс');
```

```
insert into schedule (dayId, masterId) values
(1,1),
(2,1),
(5,1),
(3,2),
(4,2),
(1,3),
(2,3),
(6,3),
(7,3);
```

```
insert into qa (question, answer) values
('Как починить туалет, если кинул туда петарду?', 'Никак. Советуем купить новый'),
('Можно ли отмыть плитку после 10 лет использования?', 'Да, нужно всего лишь одно народное средство...');
```

```
insert into techniqueType(name) values
('Кухонная техника'),
('Техника из ванны'),
('Общее');
```

```
insert into technique (typeId, name) values
(1, 'Духовая печь'),
(1, 'Микроволновая печь'),
(1, 'Кофемашина'),
(2, 'Стиральная машина');
```

```
insert into plan (techniqueId, name, cost) values
(1, 'Защити свою духовку!', 990),
(2, 'Защити свою микроволновку!', 790),
(3, 'Защити свою кофемашинку!', 1290);
```

```
insert into client (humanId, address) values
(4, 'ул. Пушкина д. 45 кв. 23'),
```

(5, 'ул. Пушкина д. 45 кв. 22'),  
(6, 'ул. Пушкина д. 45 кв. 21');

insert into owner (techniqueId ,clientId , buyDate ) values  
(1, 1, '23.02.2012'),  
(2, 2, '01.12.2021'),  
(3, 3, '15.10.2017');

insert into subscriber (planId ,clientId ,startDate , finishDate) values  
(1, 1,'24.03.2019','24.09.2019'),  
(2, 2, '04.11.2021','04.05.2022'),  
(3, 3, '14.10.2019','14.04.2020');

insert into orderStatus (name) values  
( 'В поиске мастера'),  
( 'В процессе выполнения'),  
( 'Завершен'),  
( 'Отменен');

insert into paymentType (name) values  
( 'Подписка'),  
( 'Безналичный');

insert into orders (clientId, techniqueId, paymentTypeId , content , cost , date) values  
(1,1,2,'Сломалась духовка!!ПОМОГИТЕ!!!', 3000, '14.08.2019'),  
(2,2,1,'Сломалась микроволновка!!ПОМОГИТЕ!!!', 2500, '18.02.2022'),  
(3,3,1,'Сломалась кофемашинка!!ПОМОГИТЕ!!!', 4500, '06.06.2019');

insert into masterStatus (name) values  
( 'Работает'),  
( 'Завершил заказ'),  
( 'Был заменен');

insert into masterLog (orderId , masterId, masterStatusId ) values  
(1, 1 , 2),  
(2, 2 , 2),  
(3, 3 , 2);

insert into feedback (clientId, masterId, content, rating) values  
(1,1,'Спасибо мастеру! Все быстро починил!' , 3),  
(2,2, 'Лучший мастер евер!!!', 5),  
(3,3, 'Работа выполнена быстро и четко. Благодарю мастера за оперативность.' ,4 );

insert into requestStatus (name) values  
( 'Новый'),  
( 'В работе'),  
( 'Закрыт');

insert into supportRequest (authorId, content, theme) values  
(1,'Не получается создать заказ', 'Создание заказа'),  
(2,'Не устраивает работа мастера', 'Работа мастера');

## Триггеры

--добавления мастера

```
CREATE OR REPLACE FUNCTION master_insert() RETURNS trigger AS $master_insert$
BEGIN
    NEW.rating = 0;
    NEW.photo = concat('/include/photos/',NEW.id,'.jpg');
    RETURN NEW;
END;
$master_insert$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER master_insert BEFORE INSERT ON master
FOR EACH ROW EXECUTE PROCEDURE master_insert();
```

--создание нового заказа

```
CREATE OR REPLACE FUNCTION order_insert() RETURNS trigger AS $order_insert$
DECLARE
client integer;
BEGIN
select clientId INTO client from subscriber where clientId = NEW.clientId and planId in (select id from
plan where techniqueId = NEW.techniqueId) and startDate <= CURRENT_DATE and finishDate >=
CURRENT_DATE;
    IF client IS NOT NULL THEN
        NEW.paymentTypeID = 1;
        ELSE NEW.paymentTypeID = 2;
    END IF;
    NEW.orderStatusId = 1;
    RETURN NEW;
END;
$order_insert$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER order_insert BEFORE INSERT ON orders
FOR EACH ROW EXECUTE PROCEDURE order_insert();
```

--создание подписки

```
CREATE OR REPLACE FUNCTION subscriber_insert() RETURNS trigger AS $subscriber_insert$
DECLARE
planA boolean;
start_date date;
finish_date date;
BEGIN
select active INTO planA from plan where id = NEW.planId;
    IF NOT planA THEN
        RAISE EXCEPTION 'Subscribe is unavailable now';
    END IF;
    IF NEW.startDate < CURRENT_DATE THEN
        RAISE EXCEPTION 'Incorrect date';
    END IF;
    IF NEW.startDate > NEW.finishDate THEN
        RAISE EXCEPTION 'Incorrect date';
    END IF;
    select finishDate into finish_date from subscriber where clientId = NEW.clientId and planId =
NEW.planId;
```

```

        select startDate into start_date from subscriber where clientId = NEW.clientId and planId =
NEW.planId;
        IF NEW.finishDate < finish_date THEN
            NEW.finishDate = finish_date;
        END IF;
        IF NEW.startDate > finish_date and finish_date > CURRENT_DATE THEN
            NEW.startDate = start_date;
        END IF;
        IF start_date IS NOT NULL THEN
            UPDATE subscriber SET finishDate = NEW.finishDate, startDate = NEW.startDate
where clientId = NEW.clientId and planId = NEW.planId;
            RETURN NULL;
        END IF;
        RETURN NEW;
    END;
$subscriber_insert$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER subscriber_insert BEFORE INSERT ON subscriber
FOR EACH ROW EXECUTE PROCEDURE subscriber_insert();

```

--создание отзыва

```

CREATE OR REPLACE FUNCTION feedback_insert() RETURNS trigger AS $feedback_insert$
DECLARE master integer;
BEGIN
select masterId INTO master from masterLog where masterId = NEW.masterId and orderId in (select id
from orders where clientId = NEW.clientId);
    IF master IS NULL THEN
        RAISE EXCEPTION 'This client doesn't have any orders with that master';
    END IF;

        NEW.date = current_timestamp;
    RETURN NEW;
END;
$feedback_insert$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER feedback_insert BEFORE INSERT ON feedback
FOR EACH ROW EXECUTE PROCEDURE feedback_insert();

```

--обновление рейтинга при появлении отзыва

```

CREATE OR REPLACE FUNCTION feedback_after_insert() RETURNS trigger AS $feedback_after_insert$
DECLARE newRating real;
BEGIN
        select AVG(rating)into newRating from feedback where masterId = NEW.masterId;
        update master SET rating = newRating where id = NEW.masterId;
        RETURN NEW;
    END;
$feedback_after_insert$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER feedback_after_insert AFTER INSERT ON feedback
FOR EACH ROW EXECUTE PROCEDURE feedback_after_insert();

```

--завершение заказа или смена мастера в заказе

```

CREATE OR REPLACE FUNCTION masterLog_before_update() RETURNS trigger AS
$masterLog_before_update$

```

```

BEGIN
    IF NEW.masterStatusId = 1 THEN
        RAISE EXCEPTION 'This status wrong';
    ELSEIF NEW.masterStatusId = 3 THEN UPDATE orders SET orderStatusId = 1 where id =
NEW.orderId;
    ELSEIF NEW.masterStatusId = 2 THEN UPDATE orders SET orderStatusId = 3 where id =
NEW.orderId;
    END IF;
    NEW.endDate = current_timestamp;
    RETURN NEW;
END;
$masterLog_before_update$ LANGUAGE plpgsql;

CREATE TRIGGER masterLog_before_update BEFORE UPDATE ON masterLog
FOR EACH ROW EXECUTE PROCEDURE masterLog_before_update();

--назначение мастера на заказ
CREATE OR REPLACE FUNCTION masterLog_before_insert() RETURNS trigger AS
$masterLog_before_insert$
DECLARE
id integer;
BEGIN
select masterStatusId into id from MasterLog where orderId = NEW.orderId and (masterStatusId=1 or
masterStatusId=2);
    IF id IS NOT NULL THEN
        RAISE EXCEPTION 'Order already in work';
    END IF;
    NEW.masterStatusId = 1;
    NEW.beginDate = current_timestamp;
    RETURN NEW;
END;
$masterLog_before_insert$ LANGUAGE plpgsql;

CREATE TRIGGER masterLog_before_insert BEFORE INSERT ON masterLog
FOR EACH ROW EXECUTE PROCEDURE masterLog_before_insert();

--мастер назначен на заказ
CREATE OR REPLACE FUNCTION masterLog_after_insert() RETURNS trigger AS $masterLog_after_insert$
BEGIN
    UPDATE orders SET orderStatusId = 2 where id = NEW.orderId;
    RETURN NEW;
END;
$masterLog_after_insert$ LANGUAGE plpgsql;

CREATE TRIGGER masterLog_after_insert AFTER INSERT ON masterLog
FOR EACH ROW EXECUTE PROCEDURE masterLog_after_insert();

--создание обращения ТП
CREATE OR REPLACE FUNCTION supportRequest_before_insert() RETURNS trigger AS
$supportRequest_before_insert$
BEGIN
    NEW.statusId = 1;
    NEW.date = current_timestamp;
    RETURN NEW;

```

```
END;  
$supportRequest_before_insert$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER supportRequest_before_insert BEFORE INSERT ON supportRequest  
FOR EACH ROW EXECUTE PROCEDURE supportRequest_before_insert();
```

--удаление обращения ТП

```
CREATE OR REPLACE FUNCTION supportRequest_before_delete() RETURNS trigger AS  
$supportRequest_before_delete$  
BEGIN
```

```
    IF OLD.requestStatus != 3 THEN  
        RAISE EXCEPTION 'Request is not done.';  
    END IF;
```

```
    RETURN OLD;
```

```
END;
```

```
$supportRequest_before_delete$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER supportRequest_before_delete BEFORE DELETE ON supportRequest  
FOR EACH ROW EXECUTE PROCEDURE supportRequest_before_delete();
```

--удаление подписки пользователя

```
CREATE OR REPLACE FUNCTION subscriber_before_delete() RETURNS trigger AS  
$subscriber_before_delete$  
BEGIN
```

```
    IF OLD.endDate >= current_timestamp THEN  
        RAISE EXCEPTION 'Subscribe is still active.';  
    END IF;
```

```
    RETURN OLD;
```

```
END;
```

```
$subscriber_before_delete$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER subscriber_before_delete BEFORE DELETE ON subscriber  
FOR EACH ROW EXECUTE PROCEDURE subscriber_before_delete();
```

--удаление логов мастера

```
CREATE OR REPLACE FUNCTION masterLog_before_delete() RETURNS trigger AS  
$masterLog_before_delete$
```

```
    DECLARE masterA boolean;
```

```
    BEGIN
```

```
        IF OLD.masterStatusId != 3 THEN  
            RAISE EXCEPTION 'Log with that status unavailable to delete.';  
        END IF;
```

```
        select active into masterA from accounts where id in (select userId from human where id in  
(select humanId from master where id = OLD.masterId));
```

```
        IF masterA THEN  
            RAISE EXCEPTION 'User is still active.';  
        END IF;
```

```
    RETURN OLD;
```

```
END;
```

```
$masterLog_before_delete$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER masterLog_before_delete BEFORE DELETE ON masterLog  
FOR EACH ROW EXECUTE PROCEDURE masterLog_before_delete();
```

```

--удаление комментария
CREATE OR REPLACE FUNCTION feedback_before_delete() RETURNS trigger AS
$feedback_before_delete$
    DECLARE
        masterA boolean;
        clientA boolean;
    BEGIN
        select active into masterA from accounts where id in (select userId from human where id in
(select humanId from master where id = OLD.masterId));
        IF masterA THEN
            RAISE EXCEPTION 'Master is still active.';
        END IF;
        select active into clientA from accounts where id in (select userId from human where id in
(select humanId from client where id = OLD.clientId));
        IF clientA THEN
            RAISE EXCEPTION 'Client is still active.';
        END IF;

        RETURN OLD;
    END;
$feedback_before_delete$ LANGUAGE plpgsql;

CREATE TRIGGER feedback_before_delete BEFORE DELETE ON feedback
FOR EACH ROW EXECUTE PROCEDURE feedback_before_delete();

```

## Функции

-- оформление подписки

```

CREATE OR REPLACE FUNCTION subscribe_client_plan(client_id INT, plan_id INT, start_date DATE,
finish_date DATE)
RETURNS VOID AS
$$
BEGIN
    INSERT INTO subscriber(planId, clientId, startDate, finishDate)
    VALUES (plan_id, client_id, start_date, finish_date);
END;
$$ LANGUAGE plpgsql;

```

-- получит активные планы подписки

```

CREATE OR REPLACE FUNCTION get_subscribe_plans()
RETURNS TABLE (
    id integer,
    name varchar(30),
    cost smallint,
    techniqueName varchar(20)
)AS
$$
BEGIN
    RETURN QUERY
    select plan.id, plan.name, plan.cost, technique.name from plan

```



```

        join technique on plan.techniqueId = technique.id
        where plan.active = true order by plan.id;
    END;
$$ LANGUAGE plpgsql;

-- получить доступные в сервисе приборы
CREATE OR REPLACE FUNCTION get_technique()
RETURNS TABLE (
    techniqueId integer,
    tecniqueName varchar(20)
)AS
$$
BEGIN
    RETURN QUERY
    select id, name from technique;
END;
$$ LANGUAGE plpgsql;

-- добавить прибор клиента
CREATE OR REPLACE FUNCTION add_technique(client_id INT, date_d DATE, technique_id INT)
RETURNS VOID AS
$$
BEGIN
    INSERT into owner (techniqueId,clientId, buyDate) VALUES(technique_id, client_id,date_d);
END;
$$ LANGUAGE plpgsql;

--получить приборы клиента
CREATE OR REPLACE FUNCTION get_owners(client_id INT)
RETURNS TABLE (
    techniqueId integer,
    buyDate date,
    tecniqueName varchar(20)
)AS
$$
BEGIN
    RETURN QUERY
    select technique.id, owner.buyDate, technique.name from owner
    join technique on owner.techniqueId = technique.id and owner.clientId = client_id;
END;
$$ LANGUAGE plpgsql;

-- получить мастеров, доступных в текущую дату
CREATE OR REPLACE FUNCTION get_masters(date DATE)
RETURNS TABLE (
    masterId integer,
    rating real
)AS
$$
DECLARE
    day smallint;
BEGIN
    day = extract(isodow from date);
    RETURN QUERY

```

```

        select master.id, master.rating from master
        join schedule on schedule.masterId = master.id and schedule.dayId = day;
    END;
$$ LANGUAGE plpgsql;

-- создание заказа
CREATE OR REPLACE FUNCTION create_order(client_id INT, technique_id INT, order_content TEXT,
order_cost INT, date Date)
RETURNS VOID AS
$$
BEGIN
    INSERT INTO orders(clientId, techniqueId, content, cost, date)
    VALUES (client_id, technique_id, order_content, order_cost, date);
END;
$$ LANGUAGE plpgsql;

-- создание заказа с выбором мастера
CREATE OR REPLACE FUNCTION create_order_with_master(client_id INT, technique_id INT,
order_content TEXT, order_cost INT, date1 DATE, master_id INT)
RETURNS VOID AS
$$
BEGIN
    INSERT INTO orders(clientId, techniqueId, content, cost, date)
    VALUES (client_id, technique_id, order_content, order_cost, date1);

    INSERT INTO masterLog(orderId, masterId)
    VALUES ((SELECT id FROM orders WHERE clientId = client_id AND techniqueId = technique_id
AND date = date1), master_id);
END;
$$ LANGUAGE plpgsql;

-- просмотр статуса заказа
CREATE OR REPLACE FUNCTION get_order_status(client_id INT, technique_id INT, order_date DATE)
RETURNS VARCHAR(50) AS
$$
DECLARE
    status_name VARCHAR(50);
BEGIN
    SELECT os.name INTO status_name
    FROM orders o
    JOIN orderStatus os ON o.orderStatusId = os.id
    WHERE o.clientId = client_id AND o.techniqueId = technique_id AND o.date = order_date;

    RETURN status_name;
END;
$$ LANGUAGE plpgsql;

-- оставление отзыва
CREATE OR REPLACE FUNCTION leave_feedback(client_id INT, master_id INT, feedback_content TEXT,
feedback_rating REAL)
RETURNS VOID AS
$$
DECLARE
    masterid1 integer;

```

```

BEGIN
    SELECT masterId into masterid1 FROM masterLog
    WHERE orderId = master_id and masterStatusId = 2;

    INSERT INTO feedback(clientId, masterId, content, rating)
    VALUES (client_id, masterid1, feedback_content, feedback_rating);
END;
$$ LANGUAGE plpgsql;

-- задать вопрос в тп
CREATE OR REPLACE FUNCTION leave_question(client_id INT, theme_v TEXT, comment TEXT)
RETURNS VOID AS
$$
BEGIN
    INSERT INTO supportRequest(authorId, content, theme)
    VALUES (client_id, comment, theme_v);
END;
$$ LANGUAGE plpgsql;

-- поиск ответов на частые вопросы
CREATE OR REPLACE FUNCTION search_faq(question_text TEXT)
RETURNS TEXT AS
$$
DECLARE
    answer_text TEXT;
BEGIN
    SELECT answer INTO answer_text
    FROM qa
    WHERE question = question_text;

    RETURN answer_text;
END;
$$ LANGUAGE plpgsql;

-- просмотр своего расписания
CREATE OR REPLACE FUNCTION get_master_schedule(master_id INT)
RETURNS TABLE (
    day_name VARCHAR(20),
    day_short CHAR(2)
) AS $$
BEGIN
    RETURN QUERY
    SELECT day.name, day.shortName
    FROM day
    JOIN schedule ON day.id = schedule.dayId
    WHERE schedule.masterId = master_id;
END;
$$ LANGUAGE plpgsql;

-- получить новые заказы мастера
CREATE OR REPLACE FUNCTION get_master_new_orders(master_id INT)
RETURNS TABLE (
    order_id INT,
    content TEXT,

```

```

        cost INT,
        date DATE,
        status varchar(50),
        technique varchar(50),
        client TEXT
    ) AS $$
BEGIN
    RETURN QUERY
    SELECT orders.id, orders.content, orders.cost, orders.date, orderStatus.name,
    technique.name, concat(human.surname, ' ', human.name)
    FROM newOrders
    JOIN orders ON orders.id = newOrders.orderId
    JOIN paymentType on orders.paymentTypeId = paymentType.id
    JOIN orderStatus on orders.orderStatusId = orderStatus.id
    JOIN technique on orders.techniqueId = technique.id
    JOIN client on orders.clientId = client.id
    JOIN human on client.humanId = human.id
    WHERE newOrders.masterId = master_id;
END;
$$ LANGUAGE plpgsql;

```

-- просмотр текущих заказов мастера

```
CREATE OR REPLACE FUNCTION get_master_current_orders(master_id INT)
```

```
RETURNS TABLE (
```

```

    order_id INT,
    content TEXT,
    cost INT,
    date DATE,
    status varchar(50),
    technique varchar(50),
    client TEXT

```

```
) AS $$
```

```
BEGIN
```

```
    RETURN QUERY
```

```
    SELECT orders.id, orders.content, orders.cost, orders.date, orderStatus.name,
    technique.name, concat(human.surname, ' ', human.name)
```

```
    FROM orders
```

```
    JOIN masterLog ON orders.id = masterLog.orderId
```

```
    JOIN paymentType on orders.paymentTypeId = paymentType.id
```

```
    JOIN orderStatus on orders.orderStatusId = orderStatus.id
```

```
    JOIN technique on orders.techniqueId = technique.id
```

```
    JOIN client on orders.clientId = client.id
```

```
    JOIN human on client.humanId = human.id
```

```
    WHERE masterLog.masterId = master_id AND masterLog.endDate IS NULL;
```

```
END;
```

```
$$ LANGUAGE plpgsql;
```

-- просмотр истории заказов мастера

```
CREATE OR REPLACE FUNCTION get_master_order_history(master_id INT)
```

```
RETURNS TABLE (
```

```

    order_id INT,
    content TEXT,

```

```

        cost INT,
        date DATE,
        rating REAL,
        status varchar(50),
        technique varchar(50),
        client TEXT
    ) AS $$
BEGIN
    RETURN QUERY
    SELECT orders.id, orders.content, orders.cost, orders.date, feedback.rating,
    orderStatus.name, technique.name, concat(human.surname, ' ', human.name)
    FROM orders
    JOIN masterLog ON orders.id = masterLog.orderId
    LEFT JOIN feedback ON orders.clientId = feedback.clientId
    JOIN paymentType ON orders.paymentTypeId = paymentType.id
    JOIN orderStatus ON orders.orderStatusId = orderStatus.id
    JOIN technique ON orders.techniqueId = technique.id
    JOIN client ON orders.clientId = client.id
    JOIN human ON client.humanId = human.id
    WHERE masterLog.masterId = master_id AND masterLog.endDate IS NOT NULL;
END;
$$ LANGUAGE plpgsql;

--Принять заказ мастером
CREATE OR REPLACE FUNCTION accept_order(order_id INT, master_id INT)
RETURNS VOID AS $$
BEGIN
    INSERT INTO masterLog(orderId, masterId)
    VALUES (order_id, master_id);

    DELETE from newOrders where masterId = master_id and orderId=order_id;
END;
$$ LANGUAGE plpgsql;

-- Отклонить заказ мастером
CREATE OR REPLACE FUNCTION reject_order(order_id INT, master_id INT)
RETURNS VOID AS $$
BEGIN
    UPDATE masterLog
    SET masterStatusId = (SELECT id FROM masterStatus WHERE name = 'Был заменен')
    WHERE orderId = order_id AND masterId = master_id;

    DELETE from newOrders where masterId = master_id and orderId=order_id;
END;
$$ LANGUAGE plpgsql;

-- Завершить заказ мастером
CREATE OR REPLACE FUNCTION finish_order(order_id INT, master_id INT)
RETURNS VOID AS $$
BEGIN
    UPDATE masterLog
    SET masterStatusId = (SELECT id FROM masterStatus WHERE name = 'Завершил заказ')
    WHERE orderId = order_id AND masterId = master_id;
END;

```

```
$$ LANGUAGE plpgsql;
```

```
--просмотр профиля мастера
```

```
CREATE OR REPLACE FUNCTION get_master_profile(master_id INT)
```

```
RETURNS TABLE (
```

```
    name VARCHAR(20),
```

```
    surname VARCHAR(20),
```

```
    rating REAL,
```

```
    photo VARCHAR(50),
```

```
    experience SMALLINT,
```

```
    qualification VARCHAR(100)
```

```
) AS $$
```

```
BEGIN
```

```
    RETURN QUERY
```

```
    SELECT human.name, human.surname, master.rating, master.photo, master.experience,  
master.qualification
```

```
    FROM master
```

```
    JOIN human ON master.humanId = human.id
```

```
    WHERE master.id = master_id;
```

```
END;
```

```
$$ LANGUAGE plpgsql;
```

```
-- Изменить профиль
```

```
CREATE OR REPLACE FUNCTION change_profile(master_id INT,name_ TEXT, surname_ TEXT,
```

```
experience_ TEXT, qualification_ TEXT)
```

```
RETURNS VOID AS $$
```

```
BEGIN
```

```
    IF name_ IS NOT NULL THEN
```

```
        UPDATE human
```

```
        SET name = _name
```

```
        where id in (select humanId from master where id = master_id);
```

```
    END IF;
```

```
    IF surname_ IS NOT NULL THEN
```

```
        UPDATE human
```

```
        SET surname = surname_
```

```
        where id in (select humanId from master where id = master_id);
```

```
    END IF;
```

```
    IF experience_ IS NOT NULL THEN
```

```
        UPDATE master
```

```
        SET experience = experience_
```

```
        where id = master_id;
```

```
    END IF;
```

```
    IF qualification_ IS NOT NULL THEN
```

```
        UPDATE master
```

```
        SET qualification = qualification_
```

```
        where id = master_id;
```

```
    END IF;
```

```
END;
```

```
$$ LANGUAGE plpgsql;
```

```
-- проверка логина и пароля пользователя
```

```
CREATE OR REPLACE FUNCTION check_password(
```

```
    in_username VARCHAR(255),
```

```

        in_password VARCHAR(255)
    ) RETURNS RECORD AS $$
DECLARE
    hashed_password VARCHAR(255);
    client_id integer;
    master_id integer;
    ret RECORD;
BEGIN
    SELECT client.id into client_id
    FROM accounts
JOIN human on human.userid = accounts.id
JOIN client on human.id = client.humanid
    WHERE login = in_username and password = in_password;
    SELECT master.id into master_id
    FROM accounts
JOIN human on human.userid = accounts.id
JOIN master on human.id = master.humanid
    WHERE login = in_username and password = in_password;
    IF client_id IS NOT NULL THEN SELECT client_id, 'Client' into ret;
    ELSEIF master_id IS NOT NULL THEN SELECT master_id, 'Master' into ret;
    END IF;
    RETURN RET;
END;
$$ LANGUAGE plpgsql;

```

--Получить активные заказы клиента

```

CREATE OR REPLACE FUNCTION get_client_current_orders(client_id INT)
RETURNS TABLE (
    order_id INT,
    content TEXT,
    payment varchar(50),
    date DATE,
    status varchar(50),
    master TEXT,
    technique varchar(20)
) AS $$
BEGIN
    RETURN QUERY
    SELECT orders.id, orders.content, paymentType.name, orders.date, orderStatus.name,
concat(human.surname, ' ', human.name), technique.name
    FROM orders
JOIN paymentType on orders.paymentTypeId = paymentType.id
JOIN orderStatus on orders.orderStatusId = orderStatus.id
JOIN technique on orders.techniqueId = technique.id
LEFT JOIN masterLog on orders.id = masterLog.orderId and masterLog.masterStatusId = 1
LEFT JOIN master on masterLog.masterId = master.id
LEFT JOIN human on master.humanId = human.id
    WHERE orders.clientId = client_id and orders.orderstatusId < 3;
END;
$$ LANGUAGE plpgsql;

```

-- Получить историю заказов клиента

```

CREATE OR REPLACE FUNCTION get_client_history_orders(client_id INT)
RETURNS TABLE (

```

```

        order_id INT,
        content TEXT,
        payment varchar(50),
        date DATE,
        status varchar(50),
        master TEXT,
        technique varchar(20)
    ) AS $$
BEGIN
    RETURN QUERY
    SELECT orders.id, orders.content, paymentType.name, orders.date, orderStatus.name,
concat(human.surname, ' ', human.name), technique.name
    FROM orders
    JOIN paymentType on orders.paymentTypeId = paymentType.id
    JOIN orderStatus on orders.orderStatusId = orderStatus.id
    JOIN technique on orders.techniqueId = technique.id
    LEFT JOIN masterLog on orders.id = masterLog.orderId and masterLog.masterStatusId = 2
    LEFT JOIN master on masterLog.masterId = master.id
    LEFT JOIN human on master.humanId = human.id
    WHERE orders.clientId = client_id and orders.orderstatusId > 2;
END;
$$ LANGUAGE plpgsql;

```

```

--Добавить новый заказ мастеру
CREATE OR REPLACE FUNCTION add_order(order_id INT, master_id INT)
RETURNS VOID AS $$
BEGIN
    INSERT into newOrders (orderId, masterId) values (order_id, master_id);
END;
$$ LANGUAGE plpgsql;

```

```

-- Отменить заказ
CREATE OR REPLACE FUNCTION cancel_order(order_id INT)
RETURNS VOID AS $$
BEGIN
    UPDATE orders
    SET orderStatusId = 4
    WHERE id = order_id;
END;
$$ LANGUAGE plpgsql;

```

```

-- Получение всех вопросов и ответов
CREATE OR REPLACE FUNCTION findall_faq()
RETURNS TABLE(
    question1 TEXT,
    answer1 TEXT)AS
$$
BEGIN
    RETURN QUERY
    SELECT question, answer from qa;
END;
$$ LANGUAGE plpgsql;

```



```

-- Получить подписки клиента
CREATE OR REPLACE FUNCTION findall_subscribes(client_id INT)
RETURNS TABLE (
    sub_type varchar(30),
    sub_start_date date,
    sub_finish_date date,
    sub_price smallint
)AS
$$
BEGIN
    RETURN QUERY
    select plan.name, subscriber.startDate, subscriber.finishDate, plan.cost from subscriber
    join plan on plan.id=subscriber.planId
    where client_id=subscriber.clientId;
END;
$$ LANGUAGE plpgsql;

```

```

--Получить всех мастеров
CREATE OR REPLACE FUNCTION findall_workers()
RETURNS TABLE (
    w_photo varchar(50),
    w_name TEXT,
    w_rating real,
    w_experience smallint,
    w_id int
)AS
$$
BEGIN
    RETURN QUERY
    select master.photo, concat(human.name, ' ', human.surname), master.rating, master.experience,
    master.id from master
    join human on human.id=master.humanId;
END;
$$ LANGUAGE plpgsql;

```

```

-- удалить расписание мастера
CREATE OR REPLACE FUNCTION delete_schedule(master_id INT)
RETURNS VOID AS
$$
BEGIN
    DELETE from schedule where masterId = master_id;
END;
$$ LANGUAGE plpgsql;

```

```

--добавить рабочий день в расписание
CREATE OR REPLACE FUNCTION add_day(master_id INT, day_id INT)
RETURNS VOID AS
$$
BEGIN
    INSERT into schedule (masterId, dayId) VALUES (master_id, day_id);
END;
$$ LANGUAGE plpgsql;

```

Федеральное государственное автономное образовательное учреждение  
высшего образования «Санкт-Петербургский национальный  
исследовательский университет информационных технологий, механики и  
оптики»

Факультет Программной Инженерии и Компьютерной Техники

Дисциплина: Информационные системы и базы данных

## Курсовая работа

### Этап №4

Выполнили: Тучин Артём Евгеньевич

Емельянов Дмитрий Сергеевич

Группа: Р33111

Преподаватель: Харитонова Анастасия Евгеньевна

Г. Санкт-Петербург, 2023г.

## Страница авторизации

Forma входа

Input field: 338923

Input field: .....

Button: Вход в аккаунт

Для входа в аккаунт необходимо ввести логин и пароль

Аккаунты клиентов:

1. 'the\_best\_client00' - '987732gj'
2. 'anton\_ivanov1999' - '923hfhfg2'
3. 'user29' - 'qwerty100'

Аккаунты мастеров:

1. 'kapibara2018' - 'qwerty123'
2. '777terminator777' - 'qpwofn2j\_83'
3. 'the\_best\_master' - '12345'

## Интерфейс клиента

### Заказы

Сервис ремонта

Заказы Вопросы Подписка Мастера Приборы Выход

### Заказы

Создать заказ

### Активные заказы

Заказ №4 Статус - "В процессе выполнения"

Тип техники: Кофемашина

Дата ремонта: 2023-12-23

Мастер: "Mikhailov Andrew"

Оплата: Безналичный

Комментарий: ghgправ

### История заказов

Оценить заказ

Заказ №3 Статус - Завершен

Тип техники: Кофемашина

Дата ремонта: 2019-06-06

Мастер: "Mikhailov Andrew"

Оплата: Безналичный

Комментарий: "Сломалась кофемашинка!! ПОМОГИТЕ!!!"

На страницы отображаются все заказы текущего пользователя, активные заказы – заказы в статусе “поиск мастера”(мастеру отправлен запрос на принятие заказа) или “выполняется”(мастер взял заказ в работу) и история заказов – заказы в статусе “завершен”(мастер завершил заказ) или “отменен”(на данную дату не было найдено мастеров). Завершенные заказы можно оценить нажатием на кнопку “оценить заказ” на карточке нужного заказа. Для создания нового заказа необходимо нажать кнопку “Создать заказ” на верху страницы.

Создание заказа

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Заказы

Дата, когда приедет мастер

dd . mm . yyyy

Опишите проблему

Выберите прибор

Духовая плита

Сохранить

Активные заказы

Заказ №4 Статус - "В процессе выполнения"

Тип техники: Кофемашинa

Дата ремонта: 2023-12-23

Мастер: "Mikhailov Andrew"

Оплата: Безналичный

Комментарий: ghgrrrr

История заказов

Оценить заказ

Заказ №3 Статус - Завершен

Тип техники: Кофемашинa

Дата ремонта: 2019-06-06

Необходимо заполнить форму: дата, в которую мастеру необходимо приехать, кратко описать проблему и выбрать прибор из списка ваших приборов. Далее нажать кнопку сохранить. Если есть мастера, подходящие под параметры, то заказ появится в активных заказах. Если у пользователя есть действующая подписка на данный прибор, то Оплата устанавливается по подписке, иначе – безналичный расчет.

Оценка заказа

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Оценка заказа

Рейтинг

Комментарий

Сохранить

Необходимо заполнить форму: рейтинг от 0 до 5 и комментарий о мастере, нажать кнопку сохранить.

Вопросы и ответы

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Вопросы и ответы

Новый вопрос

"Как починить туалет, если канул туда петарду?"  
Никак. Советуем купить новый"

"Можно ли отмыть плитку после 10 лет использования?"  
Да, нужно всего лишь одно народное средство..."

На вкладке можно найти ответы на популярные вопросы, а также задать свой вопрос в поддержку (нажать кнопку “Новый вопрос”).

## Новый вопрос

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Вопросы и ответы

Тема

Напишите вопрос

Сохранить

"Как починить туалет, если кинул туда петарду?"  
Никак. Советуем купить новый"

"Можно ли отмыть плитку после 10 лет использования?"  
Да, нужно всего лишь одно народное средство..."

Необходимо заполнить форму: тема вашего вопроса и сам вопрос, нажать кнопку сохранить.

## Подписка

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Подписка

Оформить новую подписку

Тип: "Защити свою духовку!"

Дата оформления: 15.12.2023

Дата окончания: 30.12.2023

Цена: 990р

На вкладке отображаются все подписки пользователя. Для оформления новой подписки или продления текущей нажмите кнопку Оформить новую подписку.

## Оформление подписки

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Подписка

Дата начала подписки

Дата окончания подписки

Выберите план

Сохранить

Тип: "Защити свою духовку!"

Дата оформления: 15.12.2023

Дата окончания: 30.12.2023

Цена: 990р


Необходимо заполнить форму: дату начала и окончания подписки, выбрать тарифный план из предложенных в списке (активные в сервисе на данный момент), нажать кнопку сохранить. Если подписка уже была, то в ее карточке изменится дата, иначе появится новая карточка подписки на странице.

Мастера

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Мастера




"Ivan Isopov"

Рейтинг: 3

Стаж(год): 2

Расписание: Пн Вт Пт

[Создать заказ](#)




"Vlad Petrov"

Рейтинг: 5

Стаж(год): 1

Расписание: Ср Чт

[Создать заказ](#)



"Andrew Mikhailov"

Рейтинг: 2.5

Стаж(год): 10

Расписание: Пн Ср Чт

[Создать заказ](#)

На странице представлены все мастера сервиса с информацией о них. По нажатию на кнопку создать заказ вы будете перенаправлены на обычную страницу создания заказа, но заказ будет отправлен выбранному мастеру.

Приборы

Сервис ремонта

ЗаказыВопросыПодпискаМастераПриборыВыход

Добавление прибора

Дата покупки:

Выберите прибор: 

Духовая печь

Сохранить

Ваши приборы

Название: "Духовая печь"

Дата покупки: 2023-12-05

Название: "Микроволновая печь"

Дата покупки: 2023-12-13

Название: Кофемашинка

Дата покупки: 2017-10-15

На вкладке отображаются все приборы, добавленные клиентом. Также можно добавить еще один прибор, заполнив форму: дата покупки прибора и выбрать прибор из списка доступных в сервисе, и нажав кнопку сохранить. Если такого прибора не было, то на странице появится его карточка.

Интерфейс мастера

Заказы мастера

Сервис ремонта

Профиль мастера

Заказы мастера

Выход

Заказы мастера

Новые заказы

[Принять заказ](#) [Отклонить заказ](#)

Заказ №5 Статус - "В поиске мастера"

Тип техники: "Духовая печь"

Дата ремонта: 12.12.2023

Клиент: "Doe John"

Цена: 1000р

Комментарий: ртпапа

Активные заказы

[Завершить заказ](#)

Заказ №1 Статус - "В процессе выполнения"

Тип техники: "Духовая печь"

Дата ремонта: 14.08.2019

Клиент: Иванов Иван

Цена: 3000р

Комментарий: "Сломалась духовка!!ПОМОГИТЕ!!!"

История заказов

На вкладке отображаются все заказы текущего мастера с информацией о них. Новые заказы – заказы, которые можно взять в работу (кнопка принять заказ на карточке соответствующего заказа) или отклонить (кнопка отклонить заказ на карточке соответствующего заказа). Активные заказы – заказы, которые мастер уже взял в работу. После проведения ремонта мастер нажимает кнопку завершить заказ на карточке этого заказа. История заказов – заказы, которые мастер уже завершил.

Профиль мастера


Сервис ремонта

Профиль мастера

Заказы мастера

Выход

Профиль



Иван Иванов

Рейтинг: 3

Стаж: 2

Расписание: Пн Вт Пт

[Изменить расписание](#)

На вкладке отображается профиль мастера с информацией, которую видят клиенты. Можно поменять свои рабочие дни, нажав на кнопку изменить расписание.

Изменение расписания

Сервис ремонта

Профиль мастера

Заказы мастера

Выход

Профиль

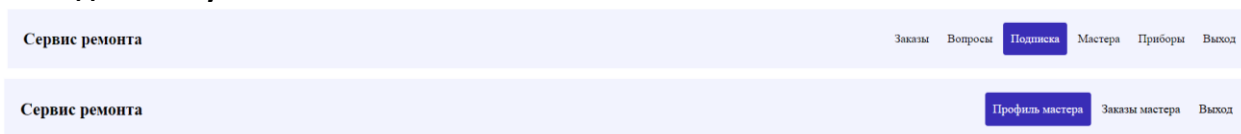
Выберите дни: 

Понедельник  
Вторник  
Среда  
Четверг

Сохранить

Для изменения расписания необходимо выбрать дни в списке и нажать кнопку сохранить. Новое расписание отобразится в вашем профиле.

## Выход из аккаунта



Для выхода из аккаунта нажмите на кнопку Выход, вы будете перенаправлены на страницу авторизации.

Исходный код - <https://github.com/artem00475/IDBD>