Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский политехнический университет»

Кафедра «Инфокогнитивные технологии»

Образовательная программа «Веб-технологии»

Отчет по курсовому проекту

по дисциплине «Инженерное проектирование»

Тема: «Разработка информационной системы для салона красоты»

**Выполнил:**

Студент группы 191-361

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Козлов А.В.

подпись, дата

**Принял:**

Старший преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Даньшина М.В.

подпись, дата

Москва 2020

ВВЕДЕНИЕ

В рамках данного курсового проекта была разработана информационная система для салона красоты. В состав разработанной системы входят frontend, backend и база данных. Система размещена на хостинге факультета информационных технологий, поэтому доступ к ней может быть получен из внутренней сети вуза.

Целью данного курсового проекта является расширение информационного поля небольшого салона красоты, а именно формирование информационной системы для более удобного способа информирования клиентов об оказываемых услугах и способа ведения учёта заказов.

Задачи данного курсового проекта:

* провести анализ отечественных и зарубежных аналогов;
* составить формальные модели для формирования представления о системе;
* составить модель данных для создания на её основе базы данных;
* создать веб-приложение и серверное приложение, ведя разработку в репозитории и использованием веток;
* для каждого приложения разъяснить структуру, описать особенности функционирования и провести тестирование.

Объектом исследования данного курсового проекта является небольшой салон красоты, а предметом исследования является формируемая информационная система для салона красоты.

В данной курсовой работе содержатся X таблиц, X рисунков, X приложения и состоит она из X страницы.

# АНАЛИТИЧЕСКАЯ ЧАСТЬ

## Анализ аналогов

<http://www.whitefox.ru/beauty-medical-knjazhe-ozero/kosmetologija-lica/kosmeticheskaja-chistka/>

<http://www.gorod-krasoti.com/prices/ekspress-uslugi-v-4-ruki-ekspress-avenyu/manikyur-ekspress-avenyu/>

<https://www.beautifysalon.ie/hand-care>

Придумать критерии

## Формальные модели

В качестве формальной модели была построена функциональная модель для бизнес-процесса «Записаться на приём». Для создания модели использовалась методология IDEF0. В составе IDEF0-модели была построена графическая диаграмма (Рисунок 1.1) с декомпозицией до первого уровня (Рисунок 1.2). Для построения диаграмм использовалось программное обеспечение ERwin Process Modeler.

Коротко говоря, данная модель отображает структуру и функции системы, так же потоки информации и материальные объекты, преобразуемые функциями.

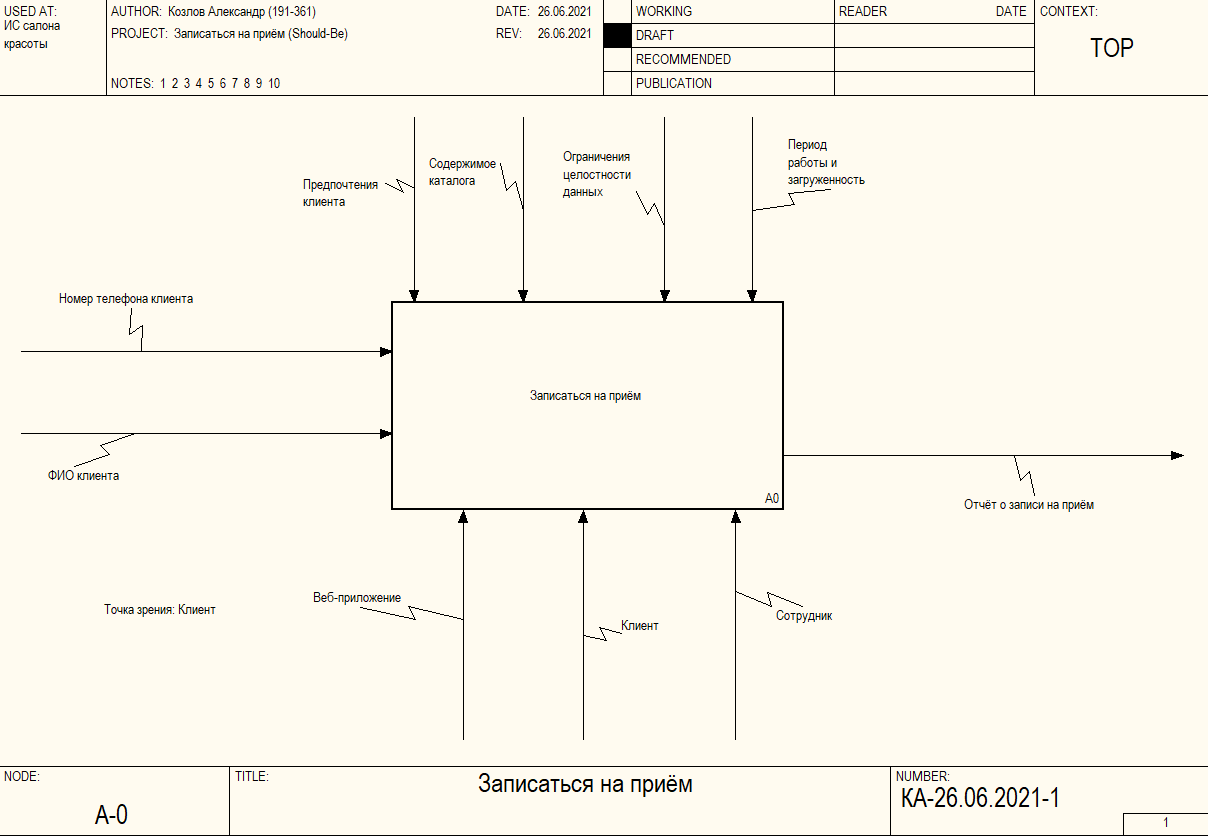


Рисунок 1.1 – Обязательная контекстная диаграмма верхнего уровня

Данная диаграмма определяет один из процессов, который должен быть внедрён в систему в конечном итоге. Сам же процесс описан с точки зрения клиента, поэтому некоторые технические подробности, например, участие backend`а, опущены.

В итоге клиент должен получить отчёт от успехе или провале подтверждения той заявки, которую он сформирует. Сам же отчёт выражен в устном подтверждении сотрудника салона красоты по телефону.

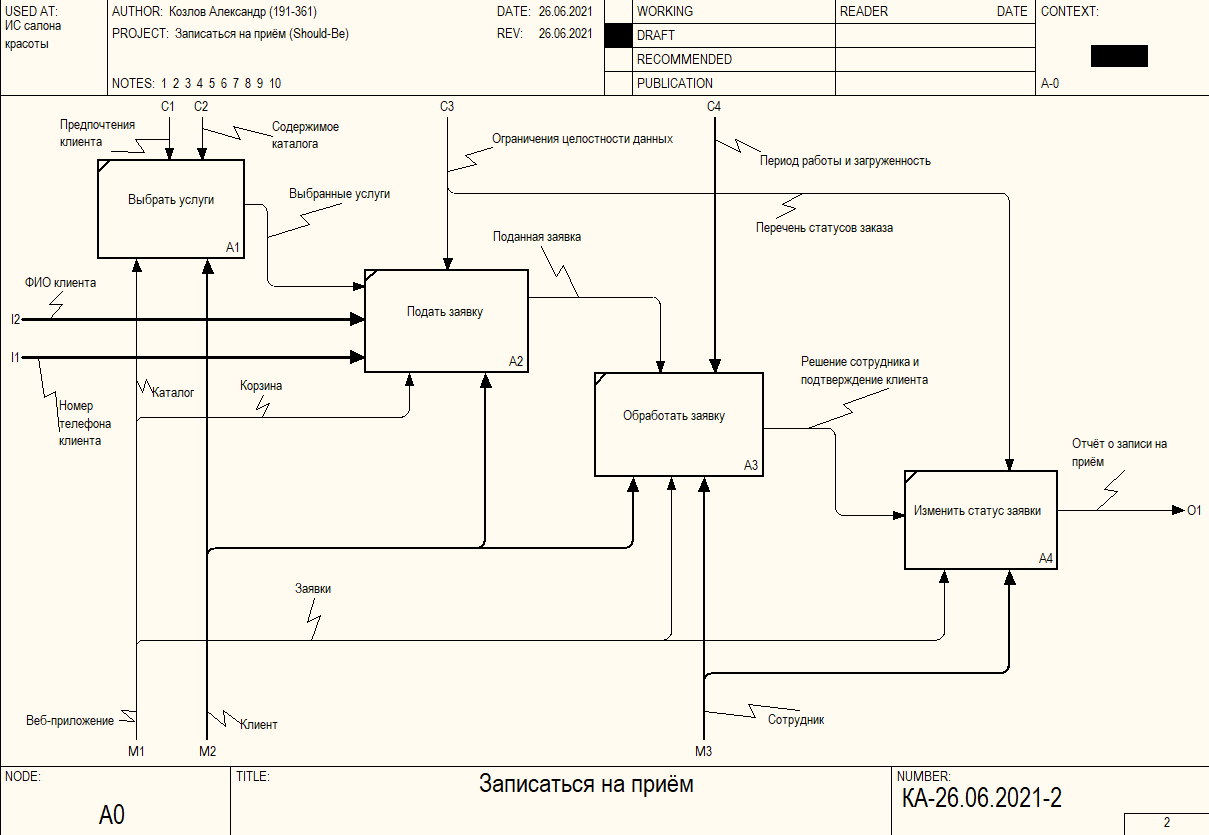


Рисунок 1.2 – Дочерняя диаграмма узла A-0

Как видно из диаграмм, все заявки должны обрабатываться сотрудниками салона красоты. Данный фактор в совокупности с накладываемыми ограничениями целостности данных позволит уменьшить число нежелательных заявок.

## Выводы по аналитической части

# ПРОЕКТНАЯ ЧАСТЬ

## Модель данных

### Концептуальная модель

При концептуальном проектировании базы данных была построена ER-модель в виде ER-диаграммы, представленной на рисунке 2.1.

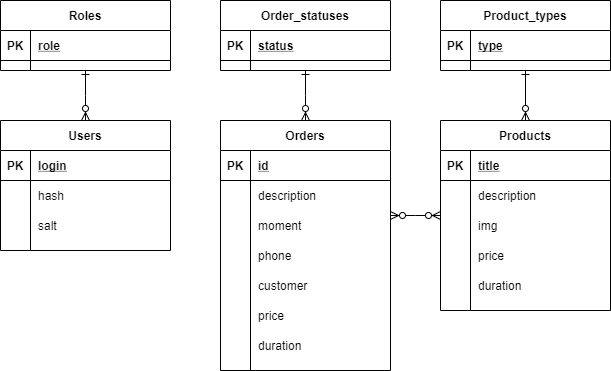


Рисунок 2.1 – Концептуальная модель данных

### Аспект манипуляции

Хранить пароли пользователей в чистом виде не безопасно, поэтому, как видно из рисунка, вместо пароля должны хранится хэш и «соль». «Соль» единожды генерируется при создании пользователя и нужна для того, чтобы хэш у пользователей с одинаковыми паролями отличался. Например, у двух пользователей пароль «123456». В таком случае в поле «hash» в таблице «Users» будет храниться не хэш от «123456», а хэш от «123456» с приписанной на конце «солью».

### Аспект структуры

В базе данных должны быть три служебные таблицы с предопределёнными значениями. Одна для хранения видов ролей пользователей, другая для хранения видов статусов заказа и последняя для хранения типов услуг салона красоты.

Между сущностями «Orders» и «Products» объявлена связь «Многие-ко-многим», поэтому для реализации этой связи в базе данных необходимо создать промежуточную таблицу.

### Аспект целостности

Для поддержания целостности данных в базе данных необходимо использовать внешние ключи между зависящими сущностями. Для простоты добавления этих ограничений у каждой сущности должно быть поле «id», по которому будет построен внешний ключ.

Логины у пользователей должны быть уникальны, поэтому необходимо добавить ограничение уникальности значений в поле «login» в таблице «Users». Аналогичное можно сказать и про названия услуг.

### Физическая модель

На основе модели, приведённой в пункте 2.1.1, была построена физическая модель данных, описанная в виде ER-диаграммы на рисунке 2.2.

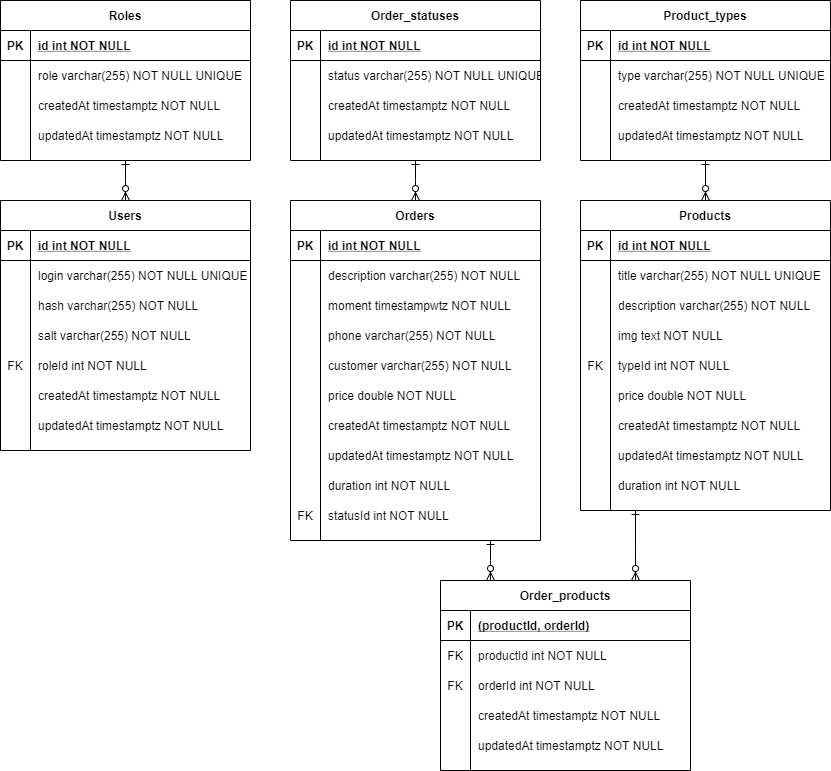


Рисунок 2.2 – Физическая модель данных

## Frontend-приложение

Веб-приложение было создано на основе фреймворка «Vue» на языке программирования «JavaScript».

Перечень используемых зависимостей в приложении:

* «axios»: используется для реализации API на frontend;
* «vue-cryptojs»: используется для хэширования паролей с «солью» перед отправной на backend;
* «vue-router»: используется для поддержки общих у нескольких страниц html-элементов и удобной навигации;
* «vuex»: используется для реализации глобального хранилища в приложении;
* «node-sass», «sass-loader», «webpack»: используются для поддержки написания стилей на scss.

### Структура проекта

Реализация API на стороне frontend находится в папке «api» и основана на «axios»: объект «axios» и его конфигурация представлены в файле «instance.js». Импорт модулей API и объекта «axios» представлены в файле «index.js». Прочие четыре файла являются модулями, между которыми разделена реализация методов API.

В папке «assets» хранятся статические файлы, такие как картинки, шрифты и стили.

Компоненты Vue-приложения распределены между папками «pages» и «components». В свою очередь, в папке «pages» хранятся компоненты, описывающие собой целую страницу, если название компонента заканчивается на «Page», или обёртку для «vue-router», если название компонента заканчивается на «Layout». В папке «components» описаны отдельные компоненты, которые могут встречаться на страницах. Также, в той же папке содержится подпапка «modal», в которую вынесены все компоненты модальных окон.

В папке «router» расположен объект «VueRouter», реализующий навигацию в приложении. Навигация в нём реализована иерархически, что позволяет выводить на экране не только компонент, в котором находится пользователь, но и элементы «родителей» этого компонента.

В папке «store» реализовано глобальное хранилище приложения на основе «vuex». Во «vuex» существует четыре основных понятия: «state» - данные, «getter» - метод получения данных, «mutation» - метод размещения данных и «action» - асинхронный метод получения данных. В проекте активно используются возможности «vuex», а некоторые элементы хранилища вынесены в отдельные модули в отдельных файлах в подпапке «modules».

### Стили

Все стили описаны в папке «assets/css» и написаны на «scss» с использованием вложенностей, миксинов и переменных.

В файле «variables.scss» описаны все переменные цветовых значений – ни одно цветовое значение не задано не через эти переменные, что позволяет очень просто интегрировать в приложение возможность переключения на другую тему, не меняя при этом другие свойства стилей.

Основные стили прописаны в файле «myStyle.scss». В начале файла импортируются переменные цветовых значений, затем сбрасываются некоторые стили по умолчанию, после этого задаются миксины: «form» применяется к каждой форме, «user-section» применяется к основным «обёрткам» с навигацией и «layout» для «обёрток», у которых снизу должен быть футер.

Вёрстка адаптивна как для широких экранов, так и для мобильных устройств.

На рисунке 2.3 представлен пример выбранной цветовой гаммы.

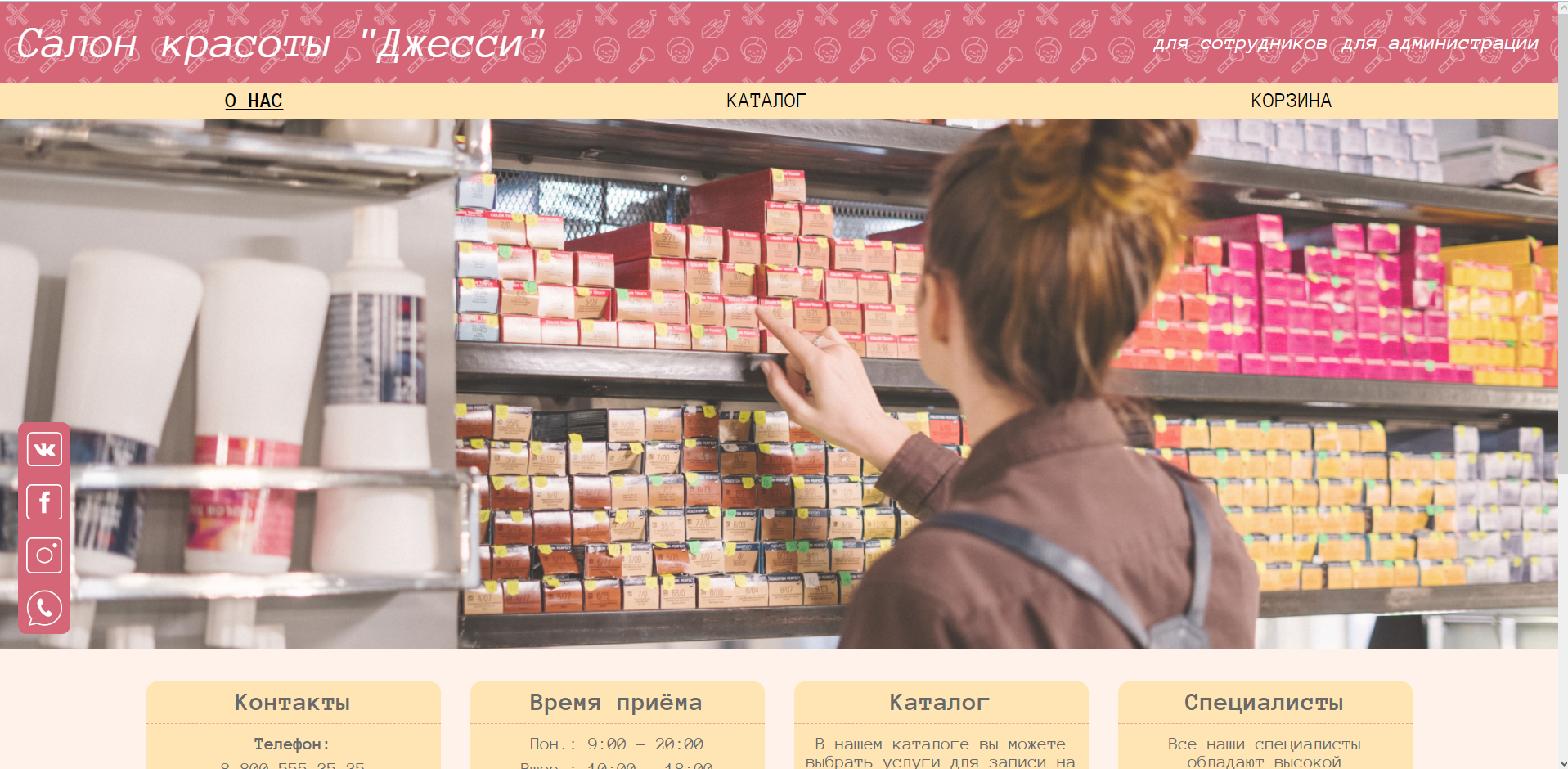


Рисунок 2.3 – Главная страница

### Особенности функционирования

Фильтр на странице каталога реализован динамически – фильтр по категориям показывает лишь те категории, которые содержатся в пришедших с backend данных, значения диапазона минимальной и максимальной цены рассчитывается на основании содержимого каталога, то же самое можно сказать и про значения диапазона продолжительности услуг.

В случае, если пользователь перешёл в корзину, но она пуста, отрисовывается компонент перенаправления в каталог, в панели навигации рядом со вкладкой корзины выводится количество добавленных товаров.

У администратора на дашборде выводится информация, агрегированная на сервере. Дашборд адаптирован для печати.

### Тестирование

## Backend-приложение

Серверное-приложение было создано на основе фреймворка «Express» на языке программирования «JavaScript».

Перечень используемых зависимостей в приложении:

* «dotenv»: используется для вынесения параметров конфигурации для подключения к базе данных;
* «jsonwebtoken»: используется для реализации получения токенов доступа и проверки токенов доступа.
* «sequelize»: используется в качестве интерфейса для обращения к базе данных, что позволило мигрировать с postgreSQL на mySQL в середине реализации проекта с минимальными затратами по времени.

### Структура проекта

### Особенности функционирования

### Тестирование

## Выводы по проектной части

ЗАКЛЮЧЕНИЕ

СПИСОК ИСТОЧНИКОВ