

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ
по дисциплине «Введение в нереляционные базы данных»
Тема: Агрегатор пошивочных компаний и заказчиков

Студенты гр. 0381, 0383

Преподаватель

Захаров Ф. С.

Козлов Т. В.

Котов Д. А.

Заславский М.М.

Санкт-Петербург

2023

ЗАДАНИЕ

Студенты

Захаров Ф. С.

Козлов Т. В.

Котов Д. А.

Группа 0381, 0383

Тема работы: Разработка агрегатора для пошивочных компаний и заказчиков.

Исходные данные:

Необходимо разработать приложение, которое объединит пошивочные компании и заказчиков спецодежды в одном месте с возможностью конструирования спецодежды для заказчиков.

Содержание пояснительной записки:

Перечисляются требуемые разделы пояснительной записки (обязательны разделы «Содержание», «Введение», «Заключение», «Список использованных источников»)

Предполагаемый объем пояснительной записки:

Не менее 10 страниц.

Дата выдачи задания:

Дата сдачи реферата:

Дата защиты реферата:

Студенты гр. 0381, 0383

Захаров Ф. С.

Козлов Т. В.

Котов Д. А.

Преподаватель

Заславский М.М.

АННОТАЦИЯ

В рамках данного курса предполагалось разработать какое-либо приложение в команде на одну из поставленных тем. Была выбрана тема создания приложения – агрегатора для пошивочных компаний и заказчиков. Для разработки была выбрана база данных MongoDB, язык разработки фреймворк для бэкенда – Python и FastAPI соответственно. Для фронтенда использовался Vue3. Был реализован конструктор и несколько основных эндпоинтов.

Исходный код можно найти на <https://github.com/moevm/nosql2h23-sewing>

Эндпоинты бэкенда на <https://www.postman.com/dark-escape-331376/workspace/lensizsewing>

Результат разработки на <https://конструктор.ленсиз.рф/>

SUMMARY

As part of this course, it was supposed to develop an application in a team on one of the set topics. The topic of creating an aggregator application for sewing companies and customers was chosen. The MongoDB database was chosen for development, and the backend framework development language was Python and FastAPI, respectively. Vue 3 was used for the frontend. A constructor and several basic endpoints were implemented.

The source code can be found at <https://github.com/moevm/nosql2h23-sewing>

Backend endpoints can be found at <https://www.postman.com/dark-escape-331376/workspace/lensizsewing>

The result of the development can be found at <https://конструктор.лениз.рф/>

СОДЕРЖАНИЕ

1.	Введение	7
2.	Сценарии использования	8
3.	Модель данных	15
3.1.	Нереляционная модель данных	15
3.2.	Реляционная модель данных	21
4.	Разработанное приложение	29
4.1.	Краткое описание	29
4.2.	Использованные технологии	29
4.3.	Снимки экрана приложения	29
5.	Выводы	31
6.	Приложения	32
7.	Литература	32

1. ВВЕДЕНИЕ

Цель работы – создать решение для объединения пошивочных компаний, которые готовы брать заказы на пошив спецодежды и компаний, которым нужна спецодежда на производство.

Было решено разработать веб-приложение с последующим переходом к Desktop-версии, которое позволит хранить данные о компаниях, объединять их, принимать заказы и конструировать модель спецодежды, необходимую для пошива.

Было решено сделать конструктор с помощью фреймворка Three.js, который позволяет моделировать 3D сцену, на которой уже и будет отображаться модель.

Для каждой роли пользователя реализованы свои методы, отвечающие за все действия. Frontend реализован на Vue3, backend реализован на FastAPI, используя базу данных MongoDB.

2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

2.1 Макет UI

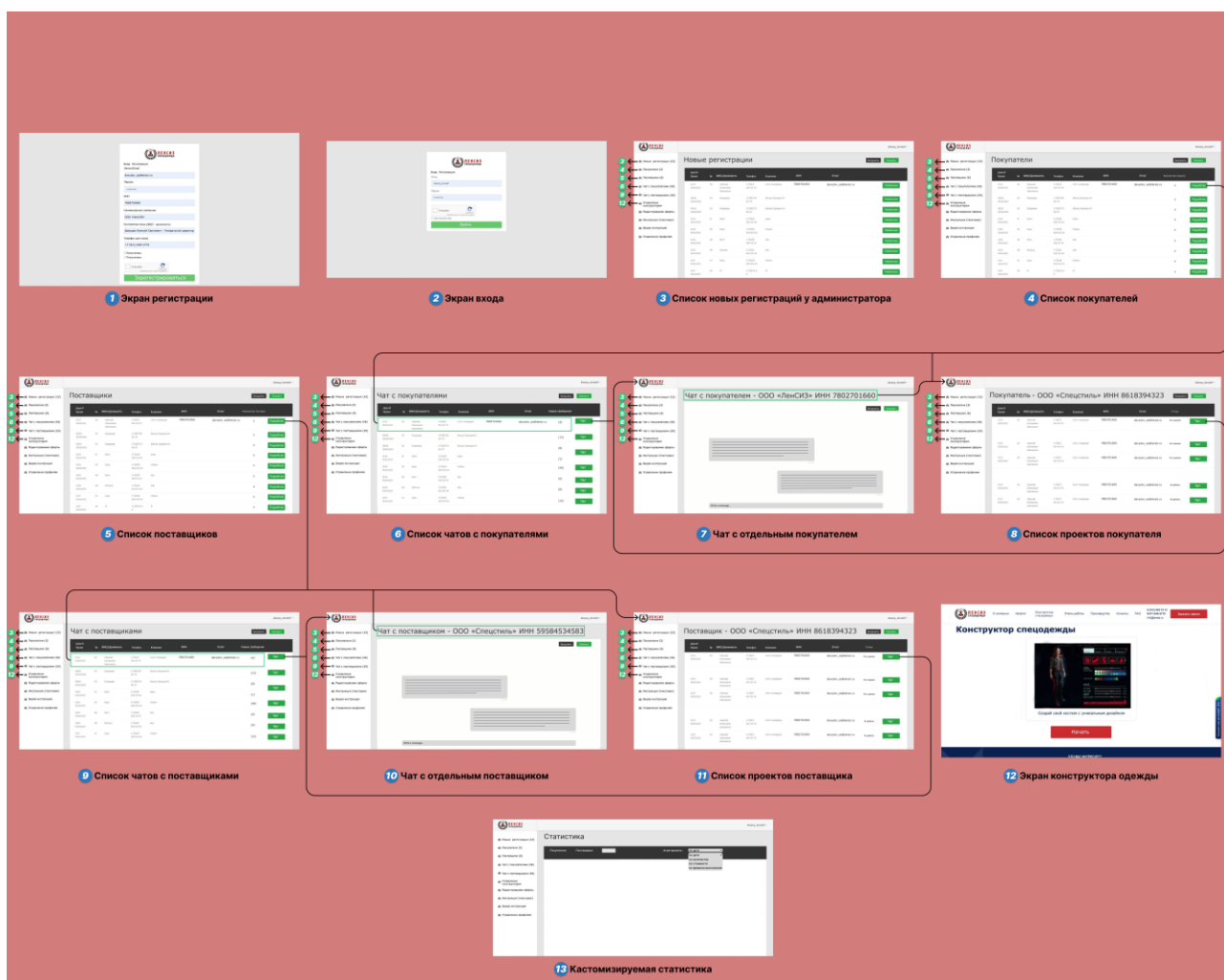


Рисунок 1 - Макет UI

2.2 Окно регистрации

The screenshot shows a registration form for 'ЛЕНСИЗ СПЕЦДЕЖДА'. At the top is the company logo. Below it are links for 'Вход' and 'Регистрация'. The form includes fields for 'Логин/Email' (filled with 'davydov_as@lensiz.ru'), 'Пароль', 'ИНН' (filled with '7802701660'), 'Наименование компании' (filled with 'ООО «ЛЕНСИЗ»'), 'Контактное лицо (ФИО - должность)' (filled with 'Давыдов Алексей Сергеевич - Генеральный директор'), and 'Телефон для связи' (filled with '+7 (911)-849-3770'). There are two checkboxes labeled 'Покупатель', a CAPTCHA section with 'Я не робот' and a 'ПАСПОРТ' logo, and a large green 'Зарегистрироваться' button at the bottom.

Рисунок 2 - Окно регистрации

2.3. Окно авторизации

The screenshot shows an authorization form for 'ЛЕНСИЗ СПЕЦДЕЖДА'. It features the company logo and links for 'Вход' and 'Регистрация'. The form has fields for 'Логин' (filled with 'alexey_lensizrf') and 'Пароль'. Below these are a CAPTCHA section with 'Я не робот' and a 'ПАСПОРТ' logo, and a 'Remember Me' checkbox. A large green 'Войти' button is at the bottom.

Рисунок 3 - Окно авторизации

2.4. Окно заявок у администратора

Новые регистрации

Дата и Время	№	ФИО/Должность	Телефон	Компания	ИНН	Email	Обработано
07:27 18.09.2023	54	Николай Евгеньевич Шапогалов	+7 (867) 340-40-70	ООО «Никофарм»	7802701660	davydov_as@lensiz.ru	Обработано
08:09 29.06.2023	53	Владимир	+7 (861) 70- 50-78	Ватер Премьер Юг			Обработано
08:04 29.06.2023	52	Владимир	+7 (861) 07- 06-07	Ватер Премьер Юг			Обработано
14:30 16.04.2023	51	Митя	+7 (439) 239-32-94	Асис			Обработано
14:29 16.04.2023	50	Боря	+7 (865) 454-65-43	Габлян			Обработано
14:28 16.04.2023	49	Витя	+7 (838) 239-23-01	Акс			Обработано
14:25 16.04.2023	48	Мюстр	+7 (923) 932-92-99	Акс			Обработано
14:27 16.04.2023	47	Боря	+7 (865) 454-66-54	Габлян			Обработано
14:07 16.04.2023	46	И	+7 (861) 09-0- 11	И			Обработано

Рисунок 4 - Окно заявок у администратора

2.5. Окно покупателей

Покупатели

Дата и Время	№	ФИО/Должность	Телефон	Компания	ИНН	Email	Количество покупок	Подробнее
07:27 18.09.2023	54	Николай Евгеньевич Шапогалов	+7 (867) 340-40-70	ООО «Никофарм»	7802701660	davydov_as@lensiz.ru	5	Подробнее
08:09 29.06.2023	53	Владимир	+7 (861) 70- 50-78	Ватер Премьер Юг			5	Подробнее
08:04 29.06.2023	52	Владимир	+7 (861) 07- 06-07	Ватер Премьер Юг			5	Подробнее
14:30 16.04.2023	51	Митя	+7 (439) 239-32-94	Асис			5	Подробнее
14:29 16.04.2023	50	Боря	+7 (865) 454-65-43	Габлян			5	Подробнее
14:28 16.04.2023	49	Витя	+7 (838) 239-23-01	Акс			5	Подробнее
14:25 16.04.2023	48	Мюстр	+7 (923) 932-92-99	Акс			5	Подробнее
14:27 16.04.2023	47	Боря	+7 (865) 454-66-54	Габлян			5	Подробнее
14:07 16.04.2023	46	И	+7 (861) 09-0- 11	И			5	Подробнее

Рисунок 5 - Окно покупателей

2.6. Окно поставщиков

Поставщики

Дата и время	№	ФИО/Должность	Телефон	Компания	ИНН	Email	Количество поставок	Подробнее
07:27 18.06.2023	54	Николай Евгеньевич Шаталов	+7 (907) 341-40-70	ООО «ЛенСИЗ»	7802701660	davydov_as@lensiz.ru	5	Подробнее
06:09 20.05.2023	53	Владимир	+7 (810) 790- 50-78	Валтер Премьер Юг			5	Подробнее
06:04 20.05.2023	52	Владимир	+7 (810) 171- 06-07	Валтер Премьер Юг			5	Подробнее
14:30 16.04.2023	51	Митя	+7 (439) 219-32-94	Акс			5	Подробнее
14:29 16.04.2023	50	Боря	+7 (855) 454-05-43	Голдман			5	Подробнее
14:29 16.04.2023	49	Витя	+7 (839) 239-23-01	Акс			5	Подробнее
14:29 16.04.2023	48	МОНСТР	+7 (923) 932-92-39	Акс			5	Подробнее
14:27 16.04.2023	47	Боря	+7 (855) 454-05-54	Голдман			5	Подробнее
14:07 16.04.2023	46	М	+7 (81) 11-11- 11	М			5	Подробнее

Рисунок 6 - Окно поставщиков

2.7. Чат с покупателем

Чат с покупателем - ООО «ЛенСИЗ» ИНН 7802701660

Write a message...

Рисунок 7 - Чат с покупателем

2.8. Конструктор спецодежды

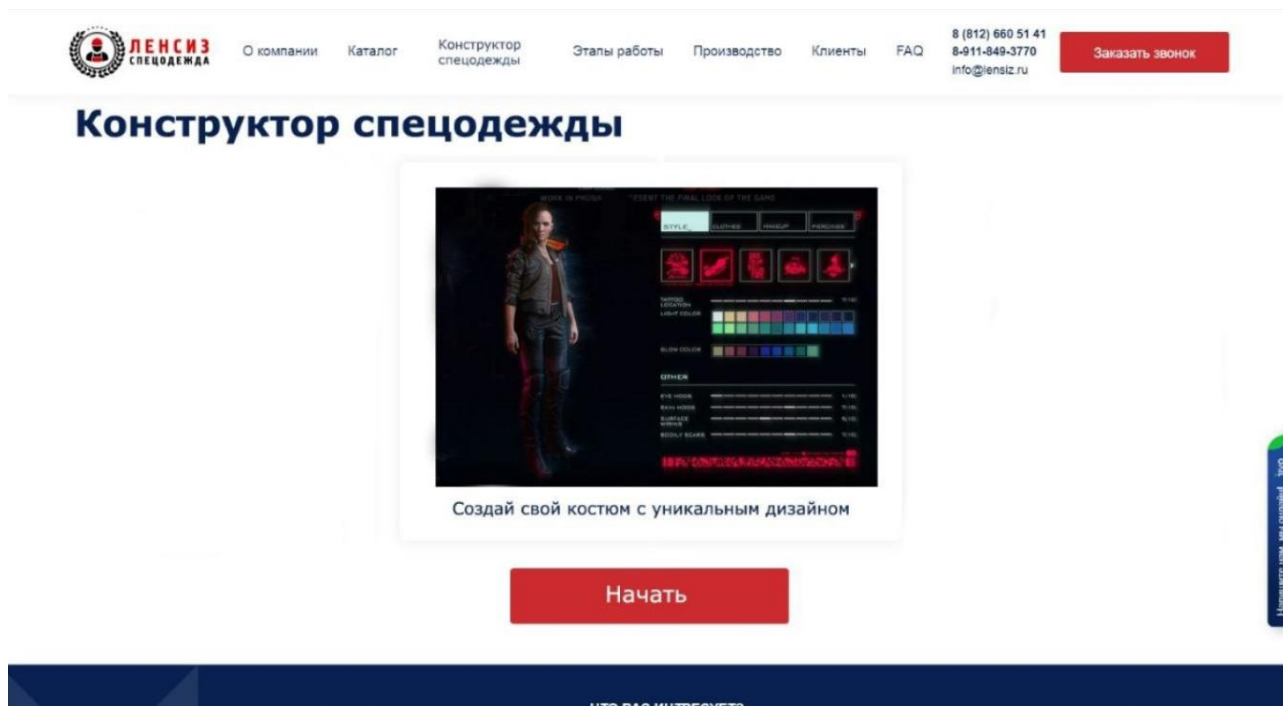


Рисунок 8 - Конструктор спецодежды

Всего существует 3 личных кабинета:

- ЛенСИЗ – администратор
- Покупатель
- Поставщик

Кабинет ЛенСИЗ должен содержать следующие разделы меню и возможности:

- Новые регистрации – в данном разделе отображаются новые заявки на регистрацию и статус (кнопка) их подтверждения. По получению заявки обязательное её подтверждение и активация менеджером ЛенСИЗ. После верификации и подтверждения контрагент попадает в раздел – покупатель/поставщик в зависимости от его статуса. (статус можно менять администратору, тк многие компании могут хитрить – будут проверяться звонком менеджера и через налоговую)
- Покупатели – содержит список покупателей, их данных (№, дата регистрации, Наименование, ИНН, ФИО/Должность, Телефон, Email)

При переходе по кнопке Подробнее – администратор попадает в карточку Покупателя.

В ней мы видим Данные на компанию – В шапке основные, ниже подробные аналогично тем, что в списке компании. Данные можно редактировать, тк лицо отвечающее за работу может меняться. Ниже мы можем видеть 4 раздела – Общий чат, Проекты на оценке, Проекты в работе, Выполненные проекты – Каждому проекту должно быть присвоено Название (на макете дизайне оно не отображено) Статус имеет право менять только администратор – переводя его уже внутри каждого проекта/чата.

Администратор видит количество новых сообщений по каждому покупателю. Внутри покупателя он также видит количество новых сообщений по каждому чату/проекту

- III. Раздел Поставщики – полностью аналогичен разделу Покупатели. Однако поставщик не имеет доступ к конструктору спецодежды и в случае его использования без согласования с ЛенСИЗ для своих целей нарушает Оферту и несет административную/уголовную ответственность. Поставщик лишь видит существующий проект созданный покупателем, отредактированный администратором.
- IV. Управление конструктором – имеет возможность добавлять новые элементы 3Д дизайна. Они постоянно будут дорабатываться и добавляться. Добавление элементов осуществляется по вышеописанной структуре. Администратор имеет возможность, как добавлять, так и удалять уже загруженные/используемые элементы одежды (манекена – мужчины и женщины)
- V. Конструктор – Содержит сам конструктор, чтобы администратор мог опробовать обновления, создать свои проекты. Они должны будут содержаться в следующем разделе.

- VI. Проекты ЛенСИЗ – содержит список созданных проектов администратором (название компании, название проекта, контактные данные ФИО телефон почта должность – аналогично проектам Клиента) Проекты можно редактировать и удалять, добавлять.
- VII. Редактирование оферты – администратор имеет возможность редактировать оферты для Клиента и Поставщика отдельно.
- VIII. Текстовая инструкция – Администратор может создавать и редактировать инструкцию для пользователей – Клиентов и Поставщиков отдельно, с возможностью вставки набора текста, картинок.
- IX. Видео-инструкция – раздел имеет возможность загрузить видео, которое будет проигрываться и показывать возможности конструктора, личного кабинета – отдельно для Клиента и Поставщика
- X. Управление профилем – редактирование информации о Администраторе – ФИО, должность, контактные данные – телефон, email, смена пароля к ЛК с какой-то защитой (доступна только владельцу компании – получается это будет отдельно верифицированный пользователь с супер-возможностями)

3. МОДЕЛЬ ДАННЫХ

3.1 Нереляционная модель данных

3.1.1 Графическое представление данных

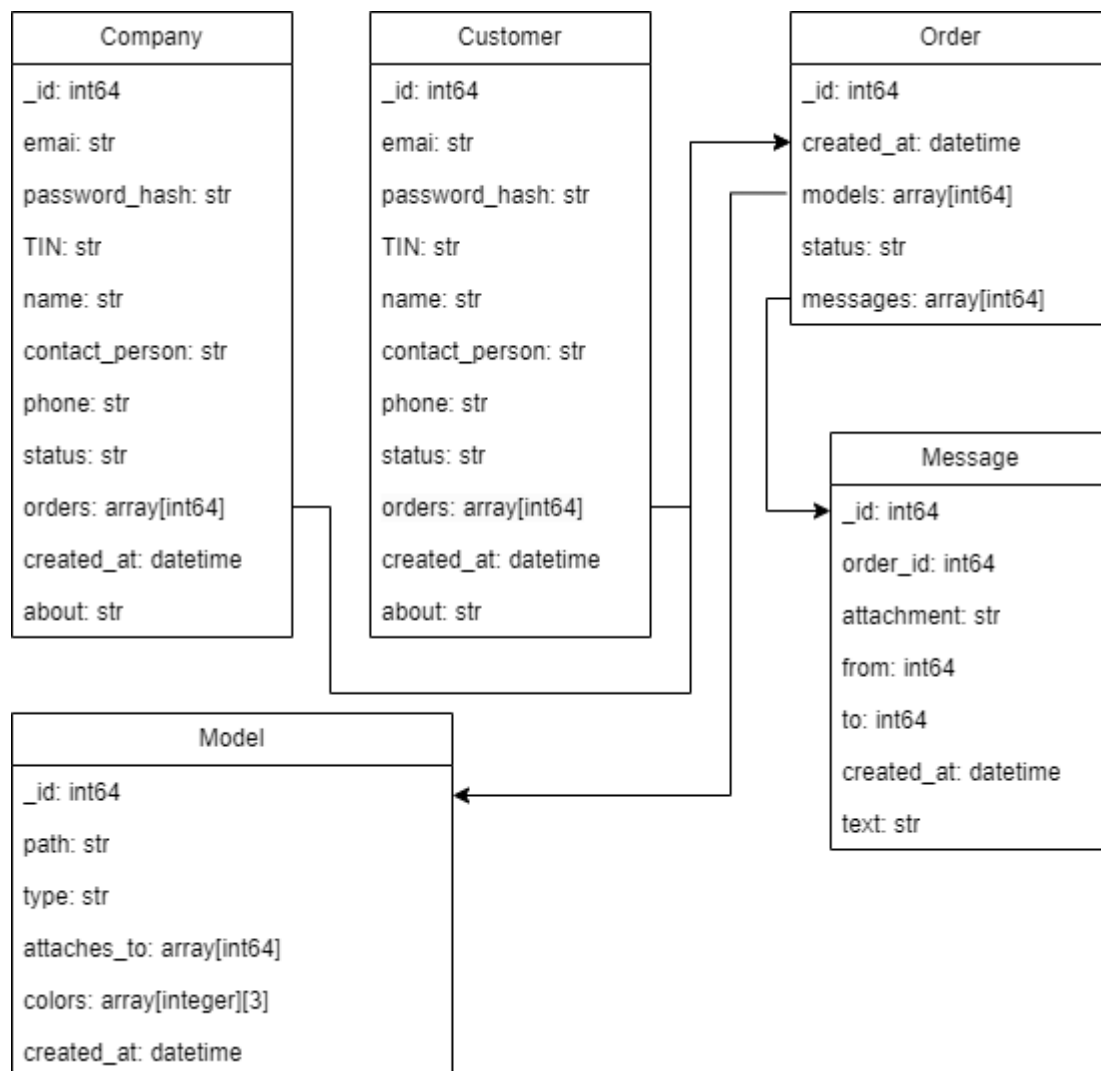


Рисунок 9 - Графическое представление нереляционной базы данных

3.1.2. Подробное описание коллекций и сущностей

БД содержит пять коллекций:

- companies - аккаунт компании, которая принимает заказы
- customers - аккаунт пользователя, формирует заказы
- orders - информация о заказе, какая модель спецодежды рассматривается для пошива, а также чат внутри заказа.
- models - информация о 3Д модели

- *messages* - сообщение в чате заказа

2.2.1 Коллекция *companies*

- *_id* - уникальный идентификатор компании
- *email* - почта
- *password_hash* - хэш пароля
- *TIN* - ИНН компании
- *name* - наименование компании
- *contact_person* - контактное лицо (ФИО - должность)
- *phone* - телефон для связи
- *status* - статус аккаунта
- *orders* - принятые заказы
- *created_at* - дата регистрации
- *about* - о компании

2.2.2 Коллекция *customers*

- *_id* - уникальный идентификатор пользователя
- *email* - почта
- *password_hash* - хэш пароля
- *TIN* - ИНН компании
- *name* - наименование компании, в которой работает
- *contact_person* - контактное лицо (ФИО - должность)
- *phone* - телефон для связи
- *status* - статус аккаунта
- *orders* - созданные заказы
- *created_at* - дата регистрации
- *about* - о компании

2.2.3 Коллекция *orders*

- *_id* - уникальный идентификатор заказа
- *created_at* - дата создания заказа
- *updated_at* - дата обновления информации о заказе

- *models* - модели, необходимые к пошиву в заказе
- *status* - статус заказа
- *messages* - сообщения в заказе

2.2.4 Коллекция models

- *_id* - уникальный идентификатор модели
- *id* - уникальный идентификатор модели для отображения администратору сайта
- *path* - путь к модели на сервере
- *type* - тип модели
- *attaches_to* - к чему можно прикрепить модель
- *key* - *id* модели, к которой можно прикрепить
- *value* - координаты крепления
- *colors* - возможные цвета модели
- *created_at* - дата добавления модели

2.2.5 Коллекция messages

- *_id* - уникальный идентификатор сообщения
- *order_id* - *id* чата, в котором написано сообщение
- *from* - *id* отправителя
- *to* - *id* получателя
- *created_at* - дата отправки сообщения
- *text* - текст сообщения
- *attachment* – вложения

3.1.3 Оценка удельного объема

3.1.3.1. Коллекция companies

_id – *Int64* (8 bytes)

email – *string* (30 bytes) – средняя длина почты 15 символов

password_hash – *string* (16 bytes)

TIN – *string* (24 bytes)

name – string (50 bytes) – средняя длина наименования компании 25 символов
contact_person – string (100 bytes) – средняя длина ФИО + должности 50 символов
phone – string (24 bytes)
status – string (10 bytes)
orders – array (28 bytes) – у компании в среднем 7 заказов с ID заказов
created_at – datetime (8 bytes)
about – string (200 bytes) – среднее описание компании 100 символов
Средний объем: 498 байт

3.1.3.2. Коллекция customers

_id - Int64 (8 bytes)
email - string (30 bytes)
password_hash - string (16 bytes)
TIN - string (24 bytes)
name - string (50 bytes)
contact_person - string (100 bytes)
phone - string (24 bytes)
status - string (10 bytes)
orders - array (14 bytes)
created_at - datetime (8 bytes)
about - string (200 bytes)
Средний объем: 484 байта

3.1.3.3. Коллекция orders

_id - Int64 (8 bytes)
created_at - datetime (4 bytes)
updated_at - datetime (4 bytes)
models - array (32) - в одном заказе по 8 моделей с ID
status - string (10 bytes)

*messages - array (80 * 8 = 640 bytes) - по 80 сообщений в заказе (Средний объём одного сообщения ниже)*

Средний объём: 698 байт

3.1.3.4. Коллекция models

_id - Int64 (8 bytes)

path - string (30 bytes) - средняя длина пути до модели 15 символов

type - string (20 bytes) - средняя длина типа 10 символов

*attaches_to - будет расти по экспоненте от мере роста количества моделей (100 * 28 = 2800 bytes)*

key - int (4 bytes)

value - array[int] (24 bytes) - 6 координат

colors - array[int] (96 bytes) - по 8 цветов RGB в каждой модели

created_at - datetime (4 bytes)

Средний объём: 2994 байта

3.1.3.5. Объекты messages

_id - Int64 (8 bytes)

order_id - Int64 (8 bytes)

from - Int64 (8 bytes)

to - Int64 (8 bytes)

created_at - datetime (4 bytes)

text - string (200 bytes) - одно сообщение в среднем 100 символов

attachment - string (30) - 15 символов путь к файлу вложения

Средний объём: 266 байт

Объём всех коллекций увеличивается линейно в зависимости от количества пользователей сайта, за исключением коллекции моделей, которые добавляет администратор. Размер коллекции увеличиваются по экспоненте из-за взаимосвязи друг с другом.

Подсчёт данных при регистрации двух пользователей - заказчика и исполнителя, 10 моделях в конструкторе и 120 сообщениях в заказе.

Заказчик - 484 байт

Исполнитель - 498 байт

Заказ - 1026 байт (отклонение от среднего +2 модели, +40 сообщений)

Модели - 29940 байт

Сообщения - 31920 байт

Итого: 63868 байт.

3.1.4. Примеры запросов

3.1.4.1. Добавление нового пользователя

```
db.customers.insert({  
  'email': email,  
  'password_hash': password_hash,  
  'TIN': TIN,  
  'name': name,  
  'contact_person': contact_person,  
  'phone': phone,  
  'status': status,  
  'password_hash': password_hash,  
  'about': about  
})
```

3.1.4.2. Добавление заказа

```
db.orders.insert({  
  'id': id,  
  'models': models,  
  'created_at': created_at,  
  'updated_at': updated_at,  
  'status': status,  
  'messages': messages
```

```
})
```

```
db.customers.updateOne(
```

```
  {"_id": _id},
```

```
  {$push: {"orders": id}}
```

```
)
```

3.1.4.3. Написание сообщения

```
db.orders.updateOne(
```

```
  {"_id": _id},
```

```
  {$push: {"messages": message}}
```

```
)
```

3.2. Реляционная модель данных

3.2.1. Графическое представление

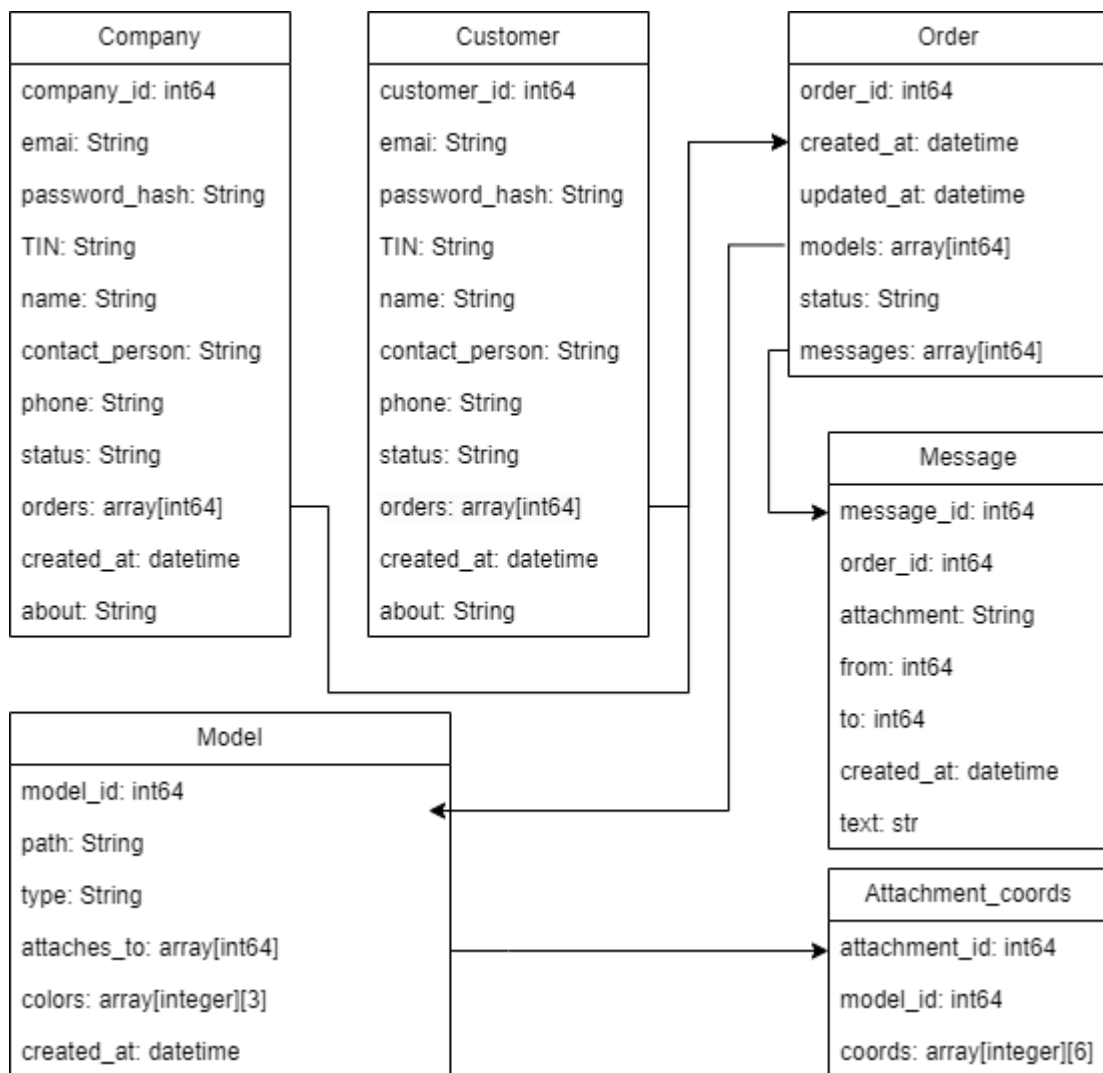


Рисунок 10 - Графическое представление реляционной базы данных

3.2.2. Подробное описание коллекций и сущностей

БД содержит пять таблиц:

companies - аккаунт компании, которая принимает заказы

customers - аккаунт пользователя, формирует заказы

orders - информация о заказе, какая модель спецодежды рассматривается для пошива, а также чат внутри заказа.

models - информация о 3Д модели

messages - сообщение в чате заказа

attachment_coords - соответствие моделей, которые могут крепиться к другим моделям и их координаты и углы поворота

3.2.2.1. Коллекция *companies*

- *company_id* - уникальный идентификатор компании
- *email* - почта
- *password_hash* - хэш пароля
- *TIN* - ИНН компании
- *name* - наименование компании
- *contact_person* - контактное лицо (ФИО - должность)
- *phone* - телефон для связи
- *status* - статус аккаунта
- *orders* - принятые заказы
- *created_at* - дата регистрации
- *about* - о компании

3.2.2.2. Коллекция *customers*

- *customer_id* - уникальный идентификатор пользователя
- *email* - почта
- *password_hash* - хэш пароля
- *TIN* - ИНН компании
- *name* - наименование компании, в которой работает

- *contact_person* - контактное лицо (ФИО - должность)
- *phone* - телефон для связи
- *status* - статус аккаунта
- *orders* - созданные заказы
- *created_at* - дата регистрации
- *about* - о компании

3.2.2.3. Коллекция orders

- *order_id* - уникальный идентификатор заказа
- *created_at* - дата создания заказа
- *updated_at* - дата обновления информации о заказе
- *models* - модели, необходимые к пошиву в заказе
- *status* - статус заказа
- *messages* - сообщения в заказе

3.2.2.4. Коллекция models

- *model_id* - уникальный идентификатор модели
- *path* - путь к модели на сервере
- *type* - тип модели
- *attaches_to* - к чему можно прикрепить модель
- *colors* - возможные цвета модели
- *created_at* - дата добавления модели

3.2.2.5. Коллекция attachment_coords

- *attachment_id* - уникальный идентификатор крепления
- *model_id* - уникальный идентификатор модели, к которой крепится текущая модель
- *coords* - координаты и угол поворота крепления

3.2.2.5. Коллекция messages

- *message_id* - уникальный идентификатор сообщения
- *order_id* - id чата, в котором написано сообщение
- *from* - id отправителя

- *to* - *id* получателя
- *created_at* - дата отправки сообщения
- *text* - текст сообщения
- *attachment* – вложения

3.2.3. Оценка удельного объема

3.2.3.1. Таблица companies

company_id - *int64* (8 bytes)

email - *string* (30 bytes) - средняя длина почты 15 символов

password_hash - *string* (16 bytes)

TIN - *string* (24 bytes)

name - *string* (50 bytes) - средняя длина наименования компании 25 символов

contact_person - *string* (100 bytes) - средняя длина ФИО + должности 50 символов

phone - *string* (24 bytes)

status - *string* (10 bytes)

orders - *array* (28 bytes) - у компании в среднем 7 заказов с ID заказов

created_at - *datetime* (8 bytes)

about - *string* (200 bytes) - среднее описание компании 100 символов

Средний объем: 498 байт

3.2.3.2. Таблица customers

customer_id - *int64* (8 bytes)

email - *string* (30 bytes)

password_hash - *string* (16 bytes)

TIN - *string* (24 bytes)

name - *string* (50 bytes)

contact_person - *string* (100 bytes)

phone - *string* (24 bytes)

status - *string* (10 bytes)

orders - array (14 bytes)

created_at - datetime (8 bytes)

about - string (200 bytes)

Средний объем: 484 байт

3.2.3.3. Таблица orders

order_id - int64 (8 bytes)

id - int (4 bytes)

created_at - datetime (4 bytes)

updated_at - datetime (4 bytes)

models - array (32) - в одном заказе по 8 моделей с ID

status - string (10 bytes)

messages - array (80 * 8 = 640 bytes) - по 80 сообщений в заказе (int64 - идентификатор сообщения)

Средний объем: 702 байт

3.2.3.4. Таблица models

model_id - int64 (8 bytes)

path - string (30 bytes) - средняя длина пути до модели 15 символов

type - string (20 bytes) - средняя длина типа 10 символов

attaches_to - int64 (100 * 8 = 800 bytes)

colors - array[int] (96 bytes) - по 8 цветов RGB в каждой модели

created_at - datetime (4 bytes)

Средний объем: 958 байт

3.2.3.5. Таблица attachment_coords

attachment_id - int64 (8 bytes)

model_id - int64 (8 bytes)

coords - array[int] (24 bytes)

Средний объем: 40 байт

3.2.3.6. Таблица messages

message_id - int64 (8 bytes)

order_id - int64 (8 bytes)

from - int64 (8 bytes)

to - int64 (8 bytes)

created_at - datetime (4 bytes)

text - string (200 bytes) - одно сообщение в среднем 100 символов

attachment - string (30) - 15 символов путь к файлу вложения

Средний объем: 266 байт

Подсчёт данных при регистрации двух пользователей - заказчика и исполнителя, 10 моделях в конструкторе и 120 сообщениях в заказе.

Заказчик - 484 байт

Исполнитель - 498 байт

Заказ - 1086 байт (отклонение от среднего +2 модели, +40 сообщений)

*Модели - 9580 байт (объекты моделей) + 100 (крепления у модели) * 10 (кол-во моделей) * 40(вес объекта крепления) = 49580 байт*

Сообщения - 31920 байт

Итого: 83568 байт.

Избыточность данных присутствует. Есть две таблицы, в которых хранится по два идентификатора - messages и attachment_coords. При присутствии key-value типа данных в рамках одного поля, две таблицы можно было бы исключить. Из-за их присутствия, пришлось бы делать поиск по всем сообщениям базы данных для поиска сообщений для конкретного диалога.

3.2.4. Примеры запросов

3.2.4.1. Добавление нового пользователя

INSERT customers(email, password_hash, TIN, name, contact_person, phone, status, password_hash, about)

VALUES (email, password_hash, TIN, name, contact_person, phone, status, password_hash, about);

3.2.4.2. Добавление заказа

INSERT orders(models, status)

VALUES (models, status);

UPDATE customers

SET orders = ARRAY_APPEND(orders, order_id)

WHERE customer_id = customer_id;

3.2.4.3. Написание сообщения

INSERT customers(order_id, attachment, from, to, text)

VALUES (order_id, attachment, from, to, text);

UPDATE orders

SET messages = ARRAY_APPEND(messages, message_id)

WHERE order_id = order_id;

3.2.4.4. Добавление модели

INSERT attachment_coords(model_id, coords)

VALUES (model_id, coords);

INSERT models(path, type, attaches_to, colors)

VALUES (path, type, attaches_to, colors);

Получение данных для одной страницы выполняется одним запросом.

3.3. Сравнение моделей

SQL модель требует больше места за счёт двух дополнительных таблиц. Они необходимы, потому что необходимо в одном поле объекта хранить несколько пар ключ-значение (например, список сообщений заказа), а такое в SQL невозможно. Можно было бы сделать массив как с цветами, но чистота кода бы сильно пострадала.

В SQL модели требуется больше запросов для добавления объектов. Также сильно увеличивается время поиска сообщений в базе данных за счёт того, что необходимо выбирать сообщения конкретного заказа среди всех

сообщений системы, в отличие от NoSQL, где только один заказ, в котором и хранятся сообщения, привязанные к нему.

Вывод: для данного проекта выгоднее использовать NoSQL модель за счёт отсутствия дублирования и скорости запроса для получения сообщений.

4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

4.1. Краткое описание

Были разработаны страница конструктора, часть backend стороны сайта, часть frontend стороны сайта. Были созданы контейнеры для базы данных MongoDB и backend и помещены в docker-compose файл. Также на хостинге был развернут portainer и образ backend выгружен в Docker Hub.

Web-приложение имеет микросервисную архитектуру – то есть при желании можно заменить backend и frontend по отдельности, frontend использует API backend стороны.

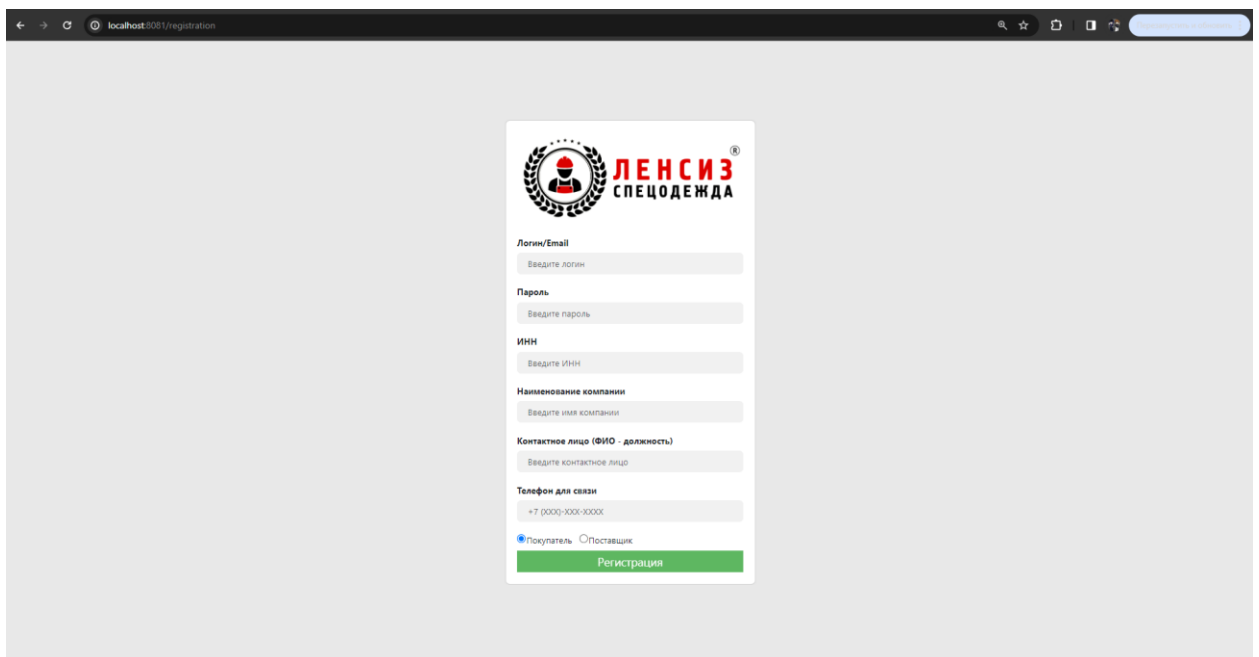
4.2. Используемые технологии

Для frontend разработки использовался Vue3 + Three.js и готовые компоненты для Vue3.

Для backend разработки использовался Python FastAPI.

База данных использовалась MongoDB в контейнере Docker.

4.3 Снимки экрана приложения



The screenshot shows a web browser window with the address bar displaying 'localhost:8081/registration'. The main content is a registration form for 'ЛЕНСИЗ СПЕЦОДЕЖДА'. The form includes the following fields and elements:

- Логин/Email:** A text input field with the placeholder 'Введите логин'.
- Пароль:** A text input field with the placeholder 'Введите пароль'.
- ИНН:** A text input field with the placeholder 'Введите ИНН'.
- Наименование компании:** A text input field with the placeholder 'Введите имя компании'.
- Контактное лицо (ФИО - должность):** A text input field with the placeholder 'Введите контактное лицо'.
- Телефон для связи:** A text input field with the placeholder '+7 (800) xxx-xxxx'.
- Роль:** Two radio buttons labeled 'Покупатель' (selected) and 'Поставщик'.
- Registration Button:** A green button labeled 'Регистрация'.

Рисунок 11 - окно регистрации

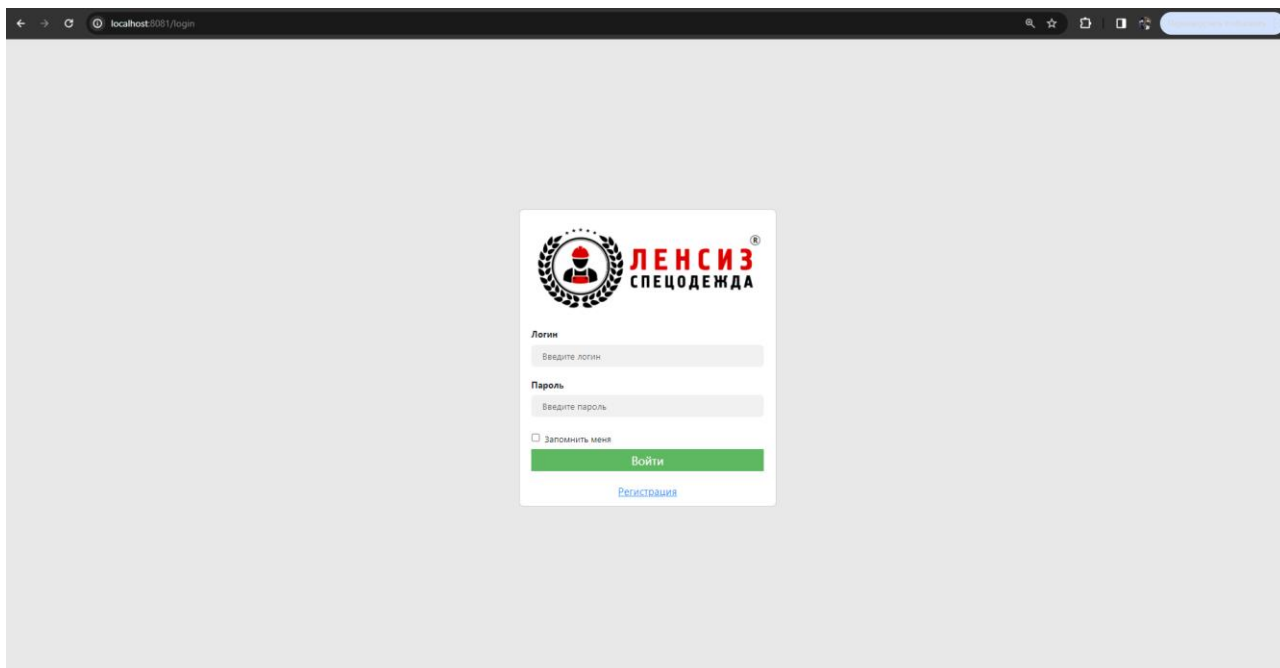


Рисунок 12 - окно авторизации

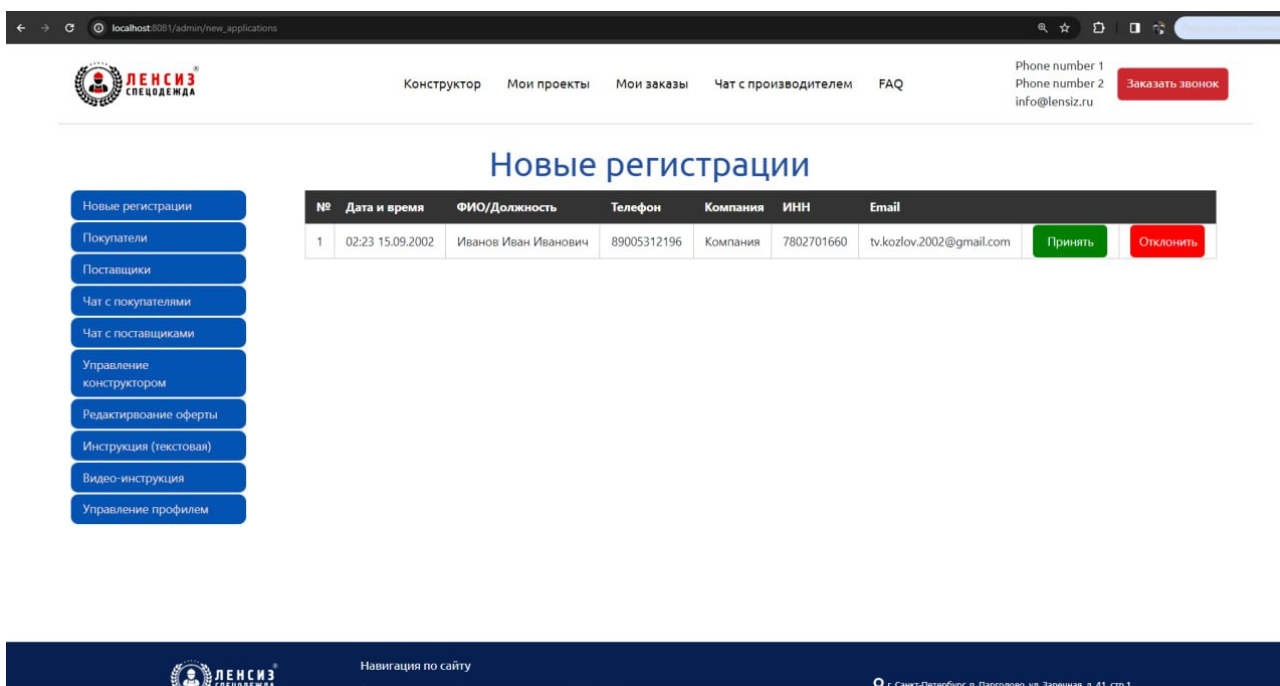


Рисунок 13 - окно новых регистраций

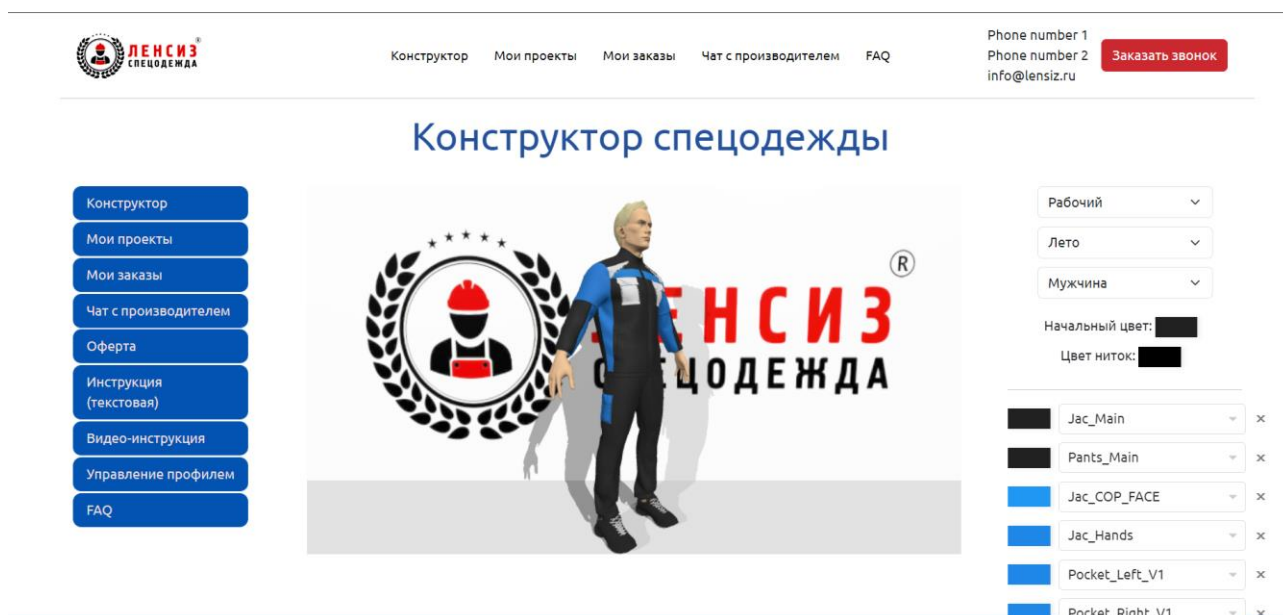


Рисунок 14 - окно конструктора

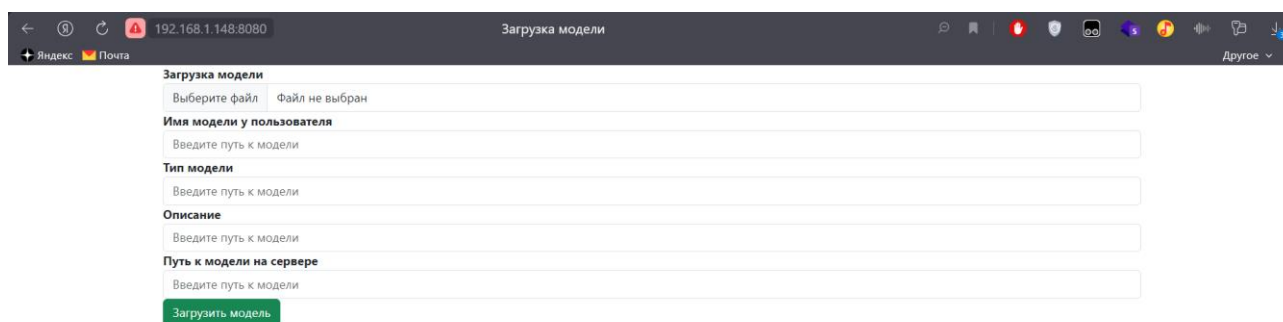


Рисунок 15 - окно загрузки модели

5. ВЫВОДЫ

5.1 Достигнутые результаты

Было разработано Web-приложение, основной частью которого является конструктор и его функционал, и он был разработан – модель людей загружаются, вся одежда на людях надевается, может быть изменен цвет самой одежды и нитей по желанию пользователя. Так же были интегрированы два типа одежды под сезон – зима и лето.

Была заготовлена архитектура для Backend и Frontend для последующей разработки.

5.2. Недостатки и пути для улучшения полученного решения

На данный момент дизайн и Frontend составляющие выглядят очень топорно, что портит общее впечатление у пользователя. Многие функции не интегрированы.

Для улучшения решения, необходимо переработать дизайн, сжать модели для более быстрой загрузки у пользователя и доработать все функции, чтоб сайт выглядел более живым.

5.3. Будущее развитие решения

Далее сайт будет дорабатываться для компании ООО «ЛенСИЗ».

6. Приложения

6.1. Документация по сборке и развертыванию приложения

1. Скачать проект из репозитория
2. Для сборки и развертывания frontend:
 - А) Перейди в папку client
 - Б) Запустить npm i для установки зависимостей
 - В) Запустить npm run build для сборки
3. Для сборки и развертывания backend:
 - А) Перейти в папку backend
 - Б) Запустить pip install -r requirements.txt для установки зависимостей
 - В) Запустить либо Dockerfile (можно пропустить пункт Б), либо main.py
4. Сервер будет доступен по адресу 127.0.0.1

7. Литература

1. Ссылка на репозиторий <https://github.com/moevm/nosql2h23-sewing>
2. Документация PyMongo: <https://www.mongodb.com/docs/drivers/pymongo/>
3. Документация FastAPI: <https://fastapi.tiangolo.com/>
4. Документация Vue: <https://vuejs.org/guide/introduction.html>
5. Документация Three.js: <https://threejs.org/docs/>