МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

Кафедра информационных систем

КУРСОВАЯ РАБОТА

Тема: Разработка информационной системы управления содержимым

Работу выполнил студент: Федотенко Николай Владимирович **группы** M33122
(фамилия, имя, отчество) (номер группы)

Руководитель Приискалов Роман Андреевич, *преподаватель практики*
(фамилия, имя, отчество)

Работа защищена "_____" _____ 2022 г. с **оценкой** _____

Подписи членов комиссии: _____

_____

САНКТ-ПЕТЕРБУРГ
2022

УНИВЕРСИТЕТ ИТМО

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)

Студент Федотенко Н.В.
(Фамилия, И., О.)

Факультет Информационных технологий и программирования

Кафедра Информационных систем Группа М33122

Направление (специальность) Информационные системы и технологии

Руководитель Приискалов Р.А., Университет ИТМО, преподаватель практики.
(Фамилия, И.О., должность, ученое звание, степень)

Дисциплина Web-программирование

Наименование темы Разработка информационной системы управления содержимым

Задание Разработать веб-приложение, проанализировать и смоделировать процессы, средства автоматизации, спроектировать архитектуру информационной системы и разработать пользовательский интерфейс

Краткие методические указания В ходе выполнения работы необходимо:

1. Описать структуру разрабатываемого приложения в целом, выделить модули и конкретизировать задачи для каждого модуля (авторизация, работа с базой данных и т.д.), а также указать основные информационные объекты, которые используются в каждом модуле (модели данных, сервисы, и прочие подобные сущности фреймворка). Выбрать методологию и в соответствии с ее правилами сформировать набор диаграмм, дающих формальное описание. Сделать выводы о функциональных требованиях к средствам автоматизации со стороны смоделированных процессов. При наличии возможность описать нефункциональные требования
2. Описать типовые функциональные возможности классов информационных систем, применяющихся для автоматизации определенных на предыдущем этапе процессов, обосновать выбор конкретного набора информационных систем, детально описать их функциональные возможности и сопоставить их с функциональными требованиями, полученными на предыдущем этапе.
3. Представить функциональную и информационную архитектуры ИС, включающие все выбранные на предыдущем этапе программные средства автоматизации. Функциональная архитектура представляется как распределение операций смоделированных процессов по функциональным компонентам отдельных программных средств. В случае взаимосвязанных процессов или распределения операций одного процесса по нескольким средствам автоматизации указывается передача данных между функциональными компонентами соответствующих модулей. Информационная архитектура представляется в виде сопоставления информационных объектов, выделенных на первом этапе с информационными объектами, реализованными в выбранных средствах автоматизации. Описать интеграцию систем на уровне совместного использования преобразования данных информационных объектов, обеспечение целостности данных и синхронизации выполняемых над ними операций.

Содержание пояснительной записки

1. Определение основных понятий
2. Анализ и моделирование процессов
3. Анализ средств автоматизации процессов
4. Проектирование архитектуры ИС
5. Реализация пользовательского интерфейса

Рекомендуемая литература

1. Мартин Фаулер - Архитектура корпоративных программных приложений. Издательский дом "Вильяме'. 2006 г.
2. Флэнаган, Дэвид. JavaScript. Полное руководство, 7-е изд. : Пер. с англ. — СПб. : ООО “Диалектика”, 2021. — 720 с .
3. Янг А., Мек Б., Кантелон М. Node.js в действии. 2-е изд. — СПб.: Питер, 2018. — 432 с.
4. Браун И.Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript. 2-е издание. — СПб.: Питер, 2021. — 336 с.
5. Современный учебник JavaScript [Электронный ресурс]. – Режим доступа: https://learn.javascript.ru/. – Дата доступа: 04.05.2021.

Руководитель

Подпись, дата

Студент

Подпись, дата

УНИВЕРСИТЕТ ИТМО

АННОТАЦИЯ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)

Студент Федотенко Н.В.
(Фамилия, И.О.)
Факультет Информационных технологий и программирования
Кафедра Информационных систем Группа М33122
Направление (специальность) 09.03.02 «Информационные системы и технологии»
Руководитель Приискалов Р.А., Университет ИТМО, преподаватель практики.
(Фамилия, И.О., место работы, должность, ученое звание, степень)

Дисциплина Web-программирование
Наименование темы Разработка информационной системы управления содержимым

ХАРАКТЕРИСТИКА КУРСОВОГО ПРОЕКТА (РАБОТЫ)

1. Цель и задачи работы

☐ Предложены студентом
☐ Сформулированы при участии студента
☐ Определены руководителем

Цель: Разработать веб-приложение.
Задачи:
1) Сформировать функциональные требования к веб-приложению.
2) Проанализировать функциональные возможности и выбрать набор средств автоматизации.
3) Сформировать функциональную и информационную архитектуру решения.

2. Характер работы

☐ Расчет
☐ Моделирование
☐ Конструирование
☐ Другое, _____

4. Содержание работы

1) Определение основных понятий
2) Анализ и моделирование процессов
3) Анализ средств автоматизации процессов
4) Проектирование архитектуры ИС
5) Реализация пользовательского интерфейса

5. Выводы

Студент_____
(подпись)
Руководитель_____
(подпись)

«_____»_____2022 г.

УНИВЕРСИТЕТ ИТМО

О Т З Ы В
РУКОВОДИТЕЛЯ
о выполнении курсового проекта (работы)

Студент Федотенко Н.В.
(Фамилия, И.О.)

Факультет Информационных технологий и программирования

Кафедра Информационных систем Группа М33122

Направление (специальность) 09.03.02 «Информационные системы и технологии»

Руководитель Приискалов Р.А., Университет ИТМО, преподаватель практики
(Фамилия, И.О., место работы, должность, ученое звание, степень)

Дисциплина Web-программирование

Наименование темы Разработка информационной системы управления содержимым

ОЦЕНКА КУРСОВОГО ПРОЕКТА (РАБОТЫ)

№ п/п	Показатели	Оценка			
		5	4	3	0*
1.	Способность к работе с литературными источниками, справочной литературой, Интернет-ресурсами и т. п.				
2.	Использование иностранных источников				
3.	Способность к анализу и обобщению информационного материала				
4.	Владение базовыми знаниями в профессиональной области				
5.	Владение базовыми знаниями в смежных областях				
6.	Владение навыками решения технических задач				
7.	Способность применять знания на практике				
8.	Уровень и корректность использования в работе методов численного моделирования, инженерных расчетов и статистической обработки данных				
9.	Владение навыками использования современных пакетов компьютерных программ и технологий				
10.	Владение навыками оформления отчетных материалов с применением современных пакетов программ				
11.	Качество оформления пояснительной записки (общий уровень грамотности, стиль изложения, качество иллюстраций, корректность цитирования и пр.**)				
12.	Качество оформления презентации				
13.	Владение навыками публичного выступления и межперсональной коммуникации				
14.	Владение навыками планирования и управления временем при выполнении работы				
ИТОГОВАЯ ОЦЕНКА					

* - не оценивается (трудно оценить)

Отмеченные достоинства:_____

Отмеченные недостатки:_____

Заключение _____

Руководитель _____
(подпись)

Дата «_____»_____2022 г.

Оглавление

Введение

Объектом разработки является система управления содержимым на языке программирования TypeScript с использованием фреймворка Nest – для серверной части приложения и языка программирования JavaScript, без использования фреймворков – для клиентской части приложения. Выбранный фреймворк один из самых поддерживаемых и распространённых.

Целью работы является разработка веб-приложения и пользовательского интерфейса, анализ требования и моделирование процессов, средств автоматизации и архитектуры информационной системы.

В ходе работы были получены следующие результаты:

- Серверная часть системы, принимающая запросы.
- Клиентская часть системы, предоставляющая интерфейс пользователя.
- База данных для хранения информации о пользователях, файлах, а также служебной информации внутри системы.

Определения, обобщения и сокращения

Браузер – прикладное программное обеспечение для просмотра страниц, содержания веб-документов, компьютерных файлов и их каталогов; управления веб-приложениями; а также для решения других задач.

Фреймворк – программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Раздел – модуль (таблица) из Prisma с соответствующим ему блоком из клиентской части.

MVC – архитектурный шаблон (паттерн), разделяющий данные приложения и управляющей логики на 3 отдельных независимо модифицируемых компонента: *модель* (набор классов, реализующих всю бизнес-логику), *представление* (набор классов и шаблонов, отвечающих за взаимодействия с пользователями) и *контроллер* (связующее звено между первыми двумя компонентами).

DDD (Domain Driven Design) – набор принципов и схем, направленных на создание систем объектов. Основные определения: *Область (Domain)* – предметная область, к которой применяется ПО; *Модель (Model)* – описывает отдельные аспекты области; *Язык описания* – используется для единого стиля описания домена/модели.

Модуль (в Nest) – это набор классов (контроллеров, сервисов, моделей и прочее), решающих одну конкретную задачу или конкретный вариант использования.

Контроллер (в Nest) – компонент, отвечающих за обработку входящих запросов и возврат ответов клиенту. Для создания контроллера используется класс и декоратор.

DTO (Data Transfer Object) – один из шаблонов проектирования, используется для передачи данных между подсистемами приложения. Не должен содержать какого-либо поведения.

Валидация – процесс проверки данных по критериям корректности и полезности для конкретного применения. Nest предлагает ValidationPipe как инструмент для проверки входных данных.

Аутентификация – процедура проверки подлинности, в контексте данной работы подразумевается проверка подлинности пользователя путем сравнения введенного им пароля (для указанного логина) с паролем, сохраненным в базе данных пользовательских логинов.

JWT – открытый стандарт (RFC 7519) для создания токенов доступа, основанный на формате JSON. Как правило, используется для передачи данных для аутентификации в клиент-серверных приложениях. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения своей личности.

WebSocket – протокол связи поверх TCP-соединения, предназначенный для обмена сообщениями между браузером и веб-сервером в режиме реального времени.

Gateway (в Nest) – класс, аннотированный декоратором *@WebSocketGateway*; платформенно независим, что позволяет ему быть совместимым с любой WebSocket библиотекой при создании адаптера.

Описание предметной области

Описание прикладного процесса

Разрабатываемое веб-приложение, это система управления содержимым. Благодаря таким системам можно, не имея навыков программирования, и не обращаясь к веб-разработчикам, самостоятельно управлять содержимым внутри сайта.

При работе с классическими системами управления контентом можно выделить такие типовые задачи как редактирование и удаление текущих, создание новых страниц или разделов.

Формирование требований

В ходе анализа прикладного процесса был получен следующий список функциональных требований:

- Возможность добавления новой информации в раздел:
  Представлено 3 раздела: "*Главные факты обо мне*" (факты), "*Мой стэк*" (навыки) и "*Мои проекты*" (проекты);
- Отображение имеющейся информации;
- Удаление имеющейся информации;
- Аутентификация пользователей;
- Возможность общения между пользователями посредством анонимного чата.

Нефункциональные требования

Разрабатываемая система не является публичной, поэтому основным требованием выступает её закрытость, путем обязательного прохождения процесса авторизации и аутентификации.

Так же должна быть возможность работы с приложением напрямую, через общие программные интерфейсы, описанные по спецификации OpenAPI версии не ниже 3.0.

Проектирование

Используемый стек технологий

Решение создания именно веб-приложения обусловлено тем, что необходимо было обеспечить доступ к системе с любого устройства, в любое время. Веб-приложение решает этот вопрос, а также снимает вопрос обновлений на стороне клиента.

Проект использует стек стандартных технологий, характерный для большинства веб-приложений: HTML, CSS, Javascript.

В качестве веб-сервера используется Express.

В качестве базы данных используется PostgreSQL, ввиду того, что данная СУБД является самой стабильной в данной связке. Библиотекой для работы с данными была выбрана Prisma.

В проекте используется свободная распределённая система управления версиями Git, хранилищем исходных кодов является крупнейший веб-сервис GitHub, а в качестве хостинга используется облачный сервис Heroku.

Системная архитектура

Рисунок 1. Системная архитектура приложения.

Архитектура данных

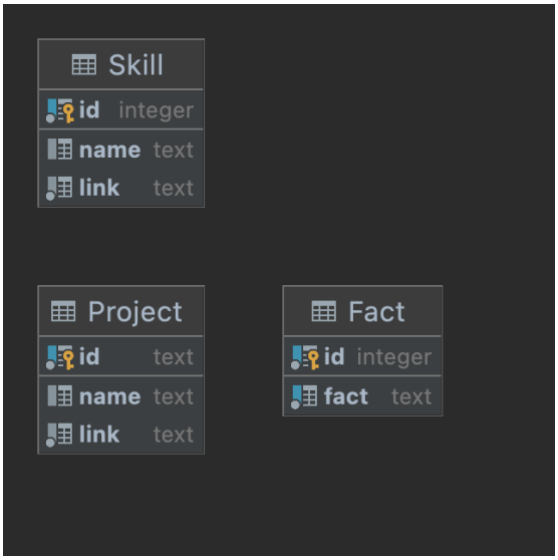

Рисунок 2. Схема таблиц базы данных

- **Fact** (раздел "*Главные факты обо мне*"):
  - **id** — уникальный идентификатор (*числовой автоинкремент*);
  - **fact** — строка с фактом, должна быть уникальной.
- **Skill** (раздел "*Мой стэк*"):
  - **id** — уникальный идентификатор (*числовой автоинкремент*);
  - **name** — название навыка, может быть пустым;
  - **link** — ссылка на картинку с навыком.
- **Project** (раздел "*Мои проекты*"):
  - **id** — уникальный идентификатор (*строка для удобства*);
  - **name** — название проекта, может быть пустым;
  - **link** — ссылка на картинку с проектом.

Программная архитектура

Таблица 1. Отношения модулей и классов

Название модуля	**Название класса**	**Назначение класса**
Fact	FactController	Сервис (Контроллер)
	FactDto	Модель передачи данных
	FactService	Сервис (Модель)
	FactModule	Корневая точка модуля
Skill	SkillController	Сервис (Контроллер)
	SkillDto	Модель передачи данных
	SkillService	Сервис (Модель)
	SkillModule	Корневая точка модуля
Project	ProjectController	Сервис (Контроллер)
	ProjectDto	Модель передачи данных
	ProjectService	Сервис (Модель)
	ProjectModule	Корневая точка модуля
Auth	PreAuthMiddleware	Связующее программное обеспечение (Ограничитель доступа)
	FirebaseApp	Класс-обертка над Сервисом Firebase
	AuthStrategy	Класс-обертка (Аутентификация по JWT)
	AuthModule	Корневая точка модуля
Chat	ChatController	Сервис (Контроллер)
	ChatGateway	Класс-обёртка (Gateway)
	ChatModule	Корневая точка модуля

Таблица 2. Описание классов

Название класса	**Описание класса**
FactController	Класс, обрабатывающий входящие запросы и возвращающий ответы.
FactDto	Класс, инкапсулирующий и валидирующий данные при передаче из одной подсистемы в другую.
FactService	Класс, используемый для работы с фактами. Осуществляет поиск фактов, отображение полного списка фактов, а также позволяет удалить тот или иной факт из общего каталога.
FactModule	Класс, использующийся фреймворком Nest для организации внутренней структуры модуля.
SkillController	Класс, обрабатывающий входящие запросы и возвращающий ответы.
SkillDto	Класс, инкапсулирующий и валидирующий данные при передаче из одной подсистемы в другую.
SkillService	Класс, используемый для работы с навыками. Осуществляет поиск навыков, отображение полного списка навыков, а также позволяет удалить тот или иной навык из общего каталога.
SkillModule	Класс, использующийся фреймворком Nest для организации внутренней структуры модуля.
ProjectController	Класс, обрабатывающий входящие запросы и возвращающий ответы.
ProjectDto	Класс, инкапсулирующий и валидирующий данные при передаче из одной подсистемы в другую.
ProjectService	Класс, используемый для работы с проектами. Осуществляет поиск проектов, отображение полного списка проектов, а также позволяет удалить тот или иной проект из общего каталога.
ProjectModule	Класс, использующийся фреймворком Nest для

	организации внутренней структуры модуля.
PreAuthMiddleware	Класс, чей основной функционал вызывается перед обработчиком маршрута и отфильтровывает неавторизованных пользователей.
FirebaseApp	Класс, служащий оберткой над классом (сервисом) FirebaseApp из фреймворка Firebase.
AuthStrategy	Класс-обёртка над PassportStrategy из Nest, реализующий аутентификацию по JWT.
AuthModule	Класс, использующийся фреймворком Nest для организации внутренней структуры модуля.
ChatController	Класс, обрабатывающий входящие запросы и возвращающий ответы.
ChatGateway	Класс, обслуживающий WebSocket канал (обертка над WebsockerServer из Nest).
ChatModule	Класс, использующийся фреймворком Nest для организации внутренней структуры модуля.

Разработка

Реализация серверного API

В качестве описания программного интерфейса был выбран инструмент, поддерживающий стандарт OAS 3.0 – Swagger. Далее представлена полученная документация API полученная автоматически по директивам, указанным в декораторах различных методов и структурах данных внутри разрабатываемой информационной системы.

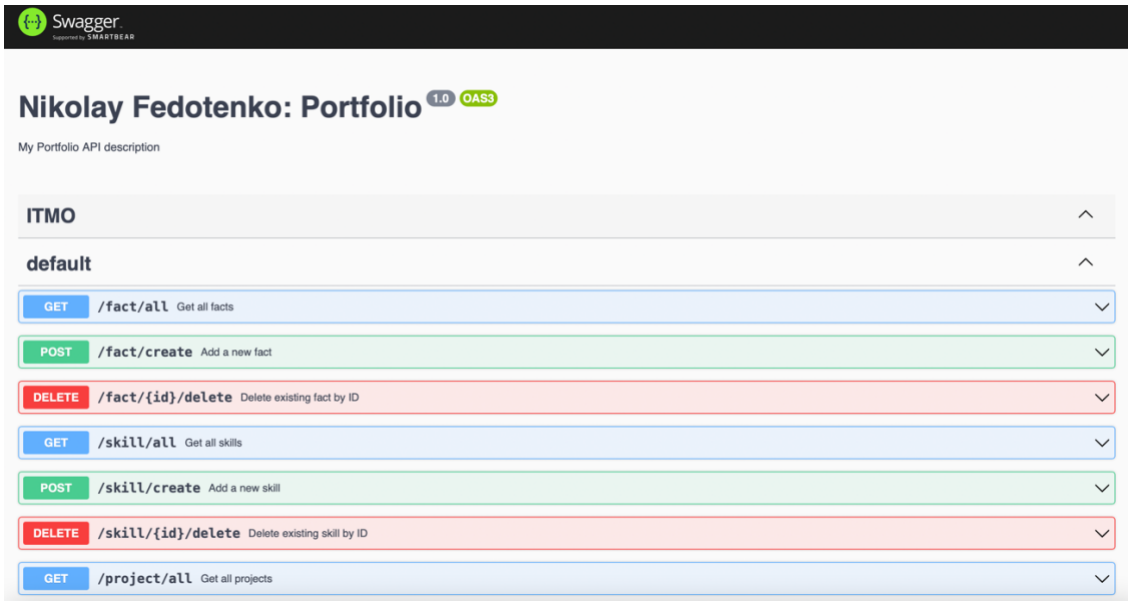

Рисунок 3.1. Программный интерфейс серверного API.

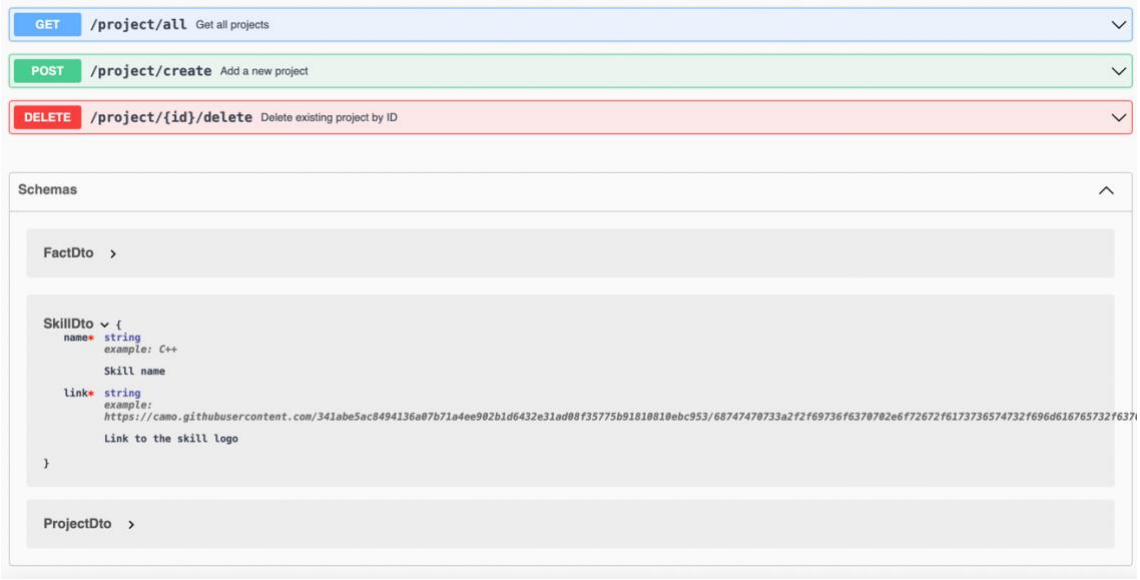

Рисунок 3.2. Программный интерфейс серверного API.

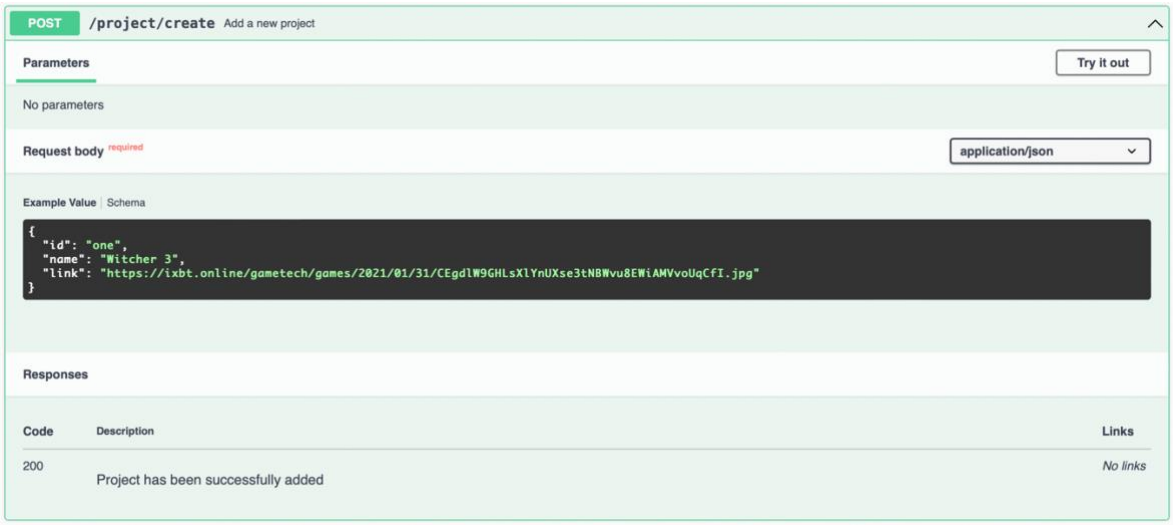

Рисунок 3.3. Программный интерфейс серверного API.

Реализация пользовательского интерфейса

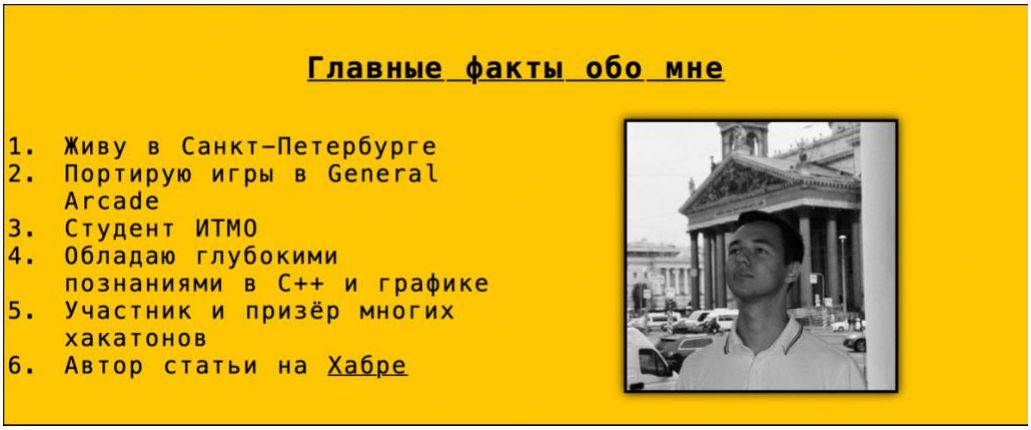

Рисунок 4. Пользовательский интерфейс взаимодействия с разделом фактов.

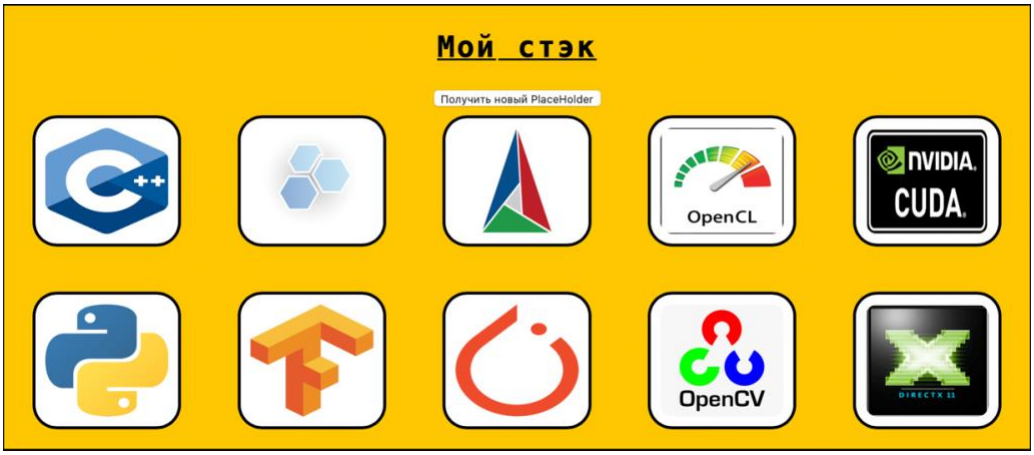

Рисунок 5. Пользовательский интерфейс взаимодействия с разделом навыков.

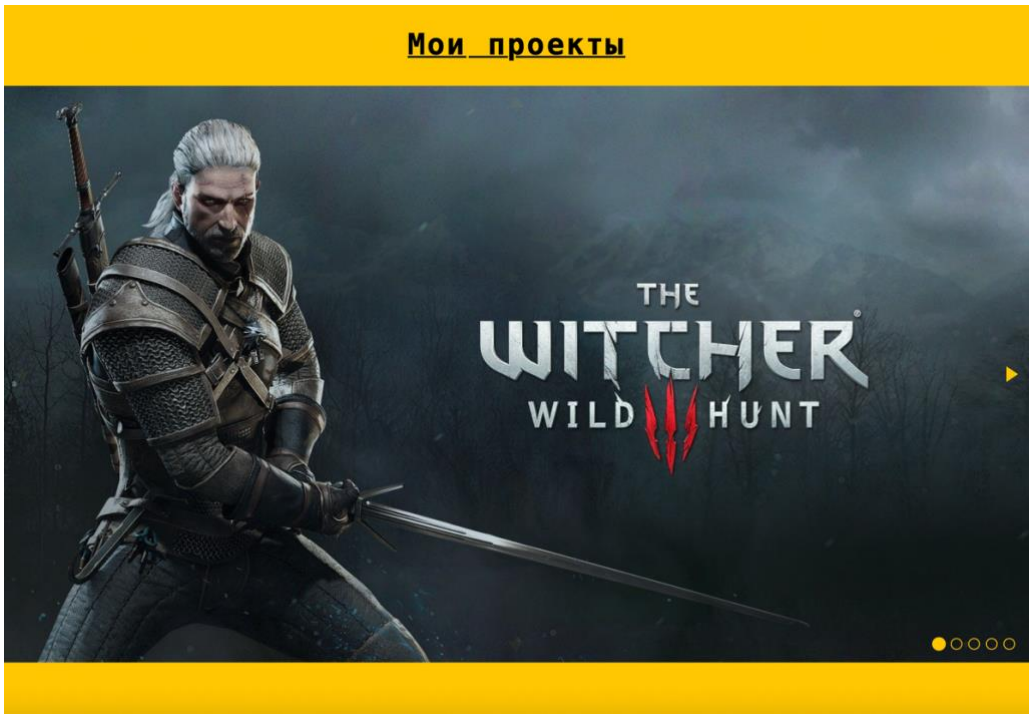

Рисунок 6. Пользовательский интерфейс взаимодействия с разделом проектов.

Рисунок 7.1. Пользовательский интерфейс авторизации.

Рисунок 7.2. Пользовательский интерфейс авторизации.

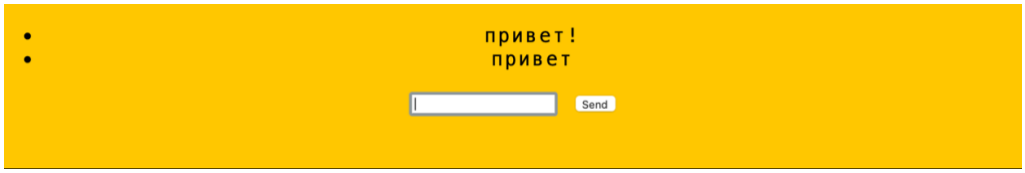

Рисунок 8. Пользовательский интерфейс чата.

Заключение

В ходе выполнения курсовой работы был проведён анализ работы классических систем управления сайтами, исходя из которого были выявлены и сформированы требования к разрабатываемому веб-приложению.

Исходя из выбранной архитектуры и наложенных ограничений были сформированы требования к используемым технологиям внутри модулей. Была спроектирована архитектура данных, программная и системная архитектура в виде набора диаграмм в нотации UML.

Опираясь на выше изложенные требования и стек технологий было разработано веб-приложение и пользовательский интерфейс в рамках дисциплины «Web-программирование».

Таким образом, все поставленные ранее цели были выполнены.

Разработанное приложение является результатом данной курсовой работы.

Список использованной литературы

1. Мартин Фаулер - Архитектура корпоративных программных приложений. Издательский дом "Вильяме'. 2006 г.

2. Флэнаган, Дэвид. JavaScript. Полное руководство, 7-е изд. : Пер. с англ. — СПб. : ООО “Диалектика”, 2021. — 720 с .

3. Янг А., Мек Б., Кантелон М. Node.js в действии. 2-е изд. — СПб.: Питер, 2018. — 432 с.

4. Браун И.Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript. 2-е издание. — СПб.: Питер, 2021. — 336 с.

5. Современный учебник JavaScript [Электронный ресурс]. – Режим доступа: https://learn.javascript.ru/. – Дата доступа: 04.05.2021.