

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

Кафедра информационных систем

КУРСОВАЯ РАБОТА

Тема: Разработка системы управления контентом

Работу выполнил студент: Триголос Алексей Павлович **группы** М33041
(фамилия, имя, отчество) (номер группы)

Руководитель Плохотнюк Вадим Станиславович, *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	<u>Триголос А. П.</u>		
	(Фамилия, И., О.)		
Факультет	<u>Информационных технологий и программирования</u>		
Кафедра	<u>Информационных систем</u>	Группа	<u>М33041</u>
Направление (специальность)	<u>Информационные системы и технологии</u>		
Руководитель	<u>Плохотнюк В. С., Университет ИТМО, преподаватель практики.</u>		
	(Фамилия, И.О., должность, ученое звание, степень)		
Дисциплина	<u>Web-программирование</u>		
Наименование темы	<u>Разработка системы управления контентом</u>		
Задание	<u>Разработать веб-приложение, проанализировать и смоделировать процессы, средства автоматизации, спроектировать архитектуру информационной системы и разработать пользовательский интерфейс</u>		
Краткие методические указания	<u>В ходе выполнения работы необходимо:</u>		
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 "Вильямс". 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 Триголос А. П.

(Фамилия, И.О.)

Факультет Информационных технологий и программирования

Кафедра Информационных систем Группа М33041

Направление (специальность) 09.03.02 «Информационные системы и технологии»

Руководитель Плохотнюк В.С., Университет ИТМО, преподаватель практики.

(Фамилия, И.О., место работы, должность, ученое звание, степень)

Дисциплина Web-программирование

Наименование темы Разработка системы управления контентом

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

☐ Конструирование

☐ Моделирование

☐ 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 Тригонос А.П.
(Фамилия, И.О.)

Факультет Информационных технологий и программирования

Кафедра Информационных систем Группа М33041

Направление (специальность) 09.03.02 «Информационные системы и технологии»

Руководитель Плохотнюк В.С., Университет ИТМО, преподаватель практики
(Фамилия, И.О., место работы, должность, ученое звание, степень)

Дисциплина Web-программирование

Наименование темы Разработка системы управления контентом

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)

ДОСТОИНСТВА: _____

[illegible]

недостатки:

[illegible]

Заключение

(ПОДПИСЬ)

(подпись)

Дата «_____» _____ 2022 г.

Оглавление

Введение.....	8
Определения, обобщения и сокращения.....	9
Описание предметной области.....	9
Описание прикладного процесса.....	10
Формирование требований.....	10
Проектирование.....	12
Используемый стек технологий.....	12
Системная архитектура.....	12
Архитектура данных	13
Программная архитектура.....	14
Разработка	15
Реализация серверного API.....	15
Реализация пользовательского интерфейса.....	21
Заключение	23
Список использованной литературы.....	24

Введение

Объектом разработки является создание своего сайта 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.

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

Model-View-Controller (далее, как MVC) — схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо.

View — это, как правило HTML файлы, которые находятся в папке `views`. Внутренняя структура зависит от названия контроллера. Допустим, если контроллер называется `SiteController`, то берется часть до `Controller` — «`Site`», меняется в нижний регистр, и, тем самым, путь до шаблонов этого контроллера становится — `views/site`.

Controller — это класс, который указывает приложению «что нужно делать». С его помощью вы создаёте action'ны и каждый из них будет выводить нужный шаблон (HTML файл — `view`), но не всегда. Зачастую, в шаблоны передаются данные полученные, используя бизнес логику, написанную в моделях (Model).

Описание предметной области

Описание прикладного процесса

Разрабатываемое веб-приложение, это в основном личный сайт. Кроме личной страницы можно увидеть, как страницы с задачами, которые можно добавлять отмечать сделанными и удалять, а также страница, на которой можно добавлять предметы, их тип, время пары и т. д. для удобного понимания в какое время какая пара проходит. Так же на сайте есть регистрация пользователей, которые в последствии могут создавать посты.

При работе в данной системе можно выделить такие типовые задачи как редактирование и удаление текущих, создание новых пользователей или постов

Формирование требований

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

Модуль работы с пользователем должен иметь:

- Возможность создание нового пользователя
- Возможность удаления пользователя
- Возможность получения всех пользователей
- Возможность изменения имени пользователю
- Возможность добавления поста пользователю

Модуль работы с постом должен иметь:

- Возможность создать пост
- Возможность изменить контент поста
- Возможность опубликовать пост
- Возможность получить все посты

Нефункциональные требования

Разрабатываемая система не является публичной, но и не является чересчур секретной, поэтому только некоторые действия доступны зарегистрированным пользователям для подтверждения того, что такое сделать возможно.

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

В первую очередь для веб-приложения необходимо создать веб-сервер, с которым клиент общается по протоколу HTTPS. Так как для общения по сети, мне пришлось создавать отдельный WebSocket-сервер. С ним клиент общается по протоколу WSS. В своем приложении я активно работаю с базой данных, поэтому ее тоже пришлось развернуть. Общение веб-сервера с базой данных осуществляется по протоколу ODBC.

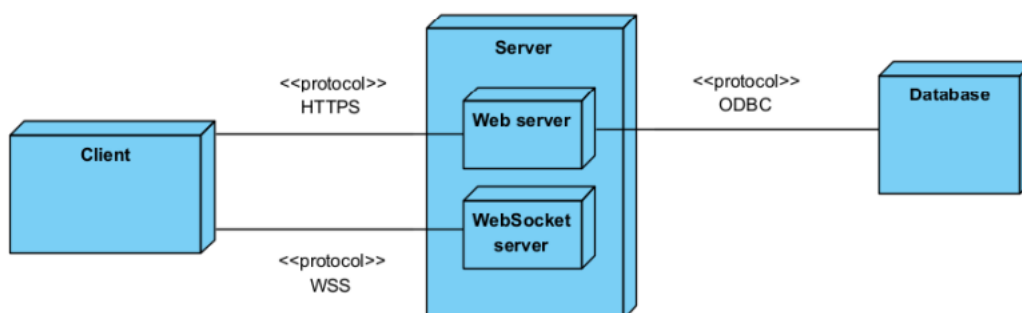


Рисунок 1. Системная архитектура приложения.

Архитектура данных

Сущность User представляет из себя пользователя. Ключом является самоинкрементирующийся id, так же user имеет уникальный E-mail, имеет имя и массив записей (Post)

Сущность Post представляет из себя запись. Ключом является самоинкрементирующийся id, так же post имеет заголовок, описание, автора, Id автора и статус опубликован или нет.

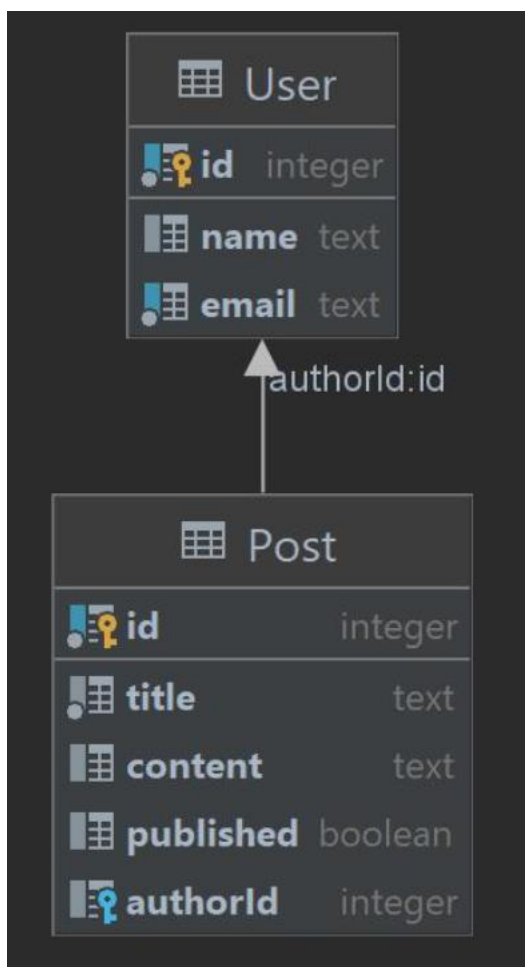


Рисунок 2. Схема таблиц базы данных

User – Представляет информацию о пользователях

Post – Представляет информацию о постах

Программная архитектура

В своем проекте я использовал схему MVC.

Таблица 1. Отношения модулей и классов

Название модуля	Название класса	Назначение класса
MyPostModule	MyPostController	Сервис (Контроллер)
	MyPostRequestDto	Модель передачи данных
UserModule	UserController	Сервис (Контроллер)
	UserRequestDto	Модель передачи данных


Таблица 2. Описание классов

Название класса	Описание класса
UserService	Класс используется для работы с пользователями. Осуществляет поиск сущностей, отображение полного списка пользователей, даёт удалить или изменить пользователя.
MyPostService	Класс используется для работы с постами. Осуществляет поиск сущностей, отображение полного списка постов, даёт удалить или изменить пост.

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.

 **Swagger**
Supported by SMARTBEAR

Example 1.0 OAS3

The API description

Authorize 

app

GET / Main page

GET /second Register page

POST /second Change user

GET /third Just page

GET /fourth Lessons page

GET /tasks Tasks page

users

POST /users Create user

DELETE /users Delete user

GET /users Get all users

PUT /users Set user name

PUT /users/postId Add post for user

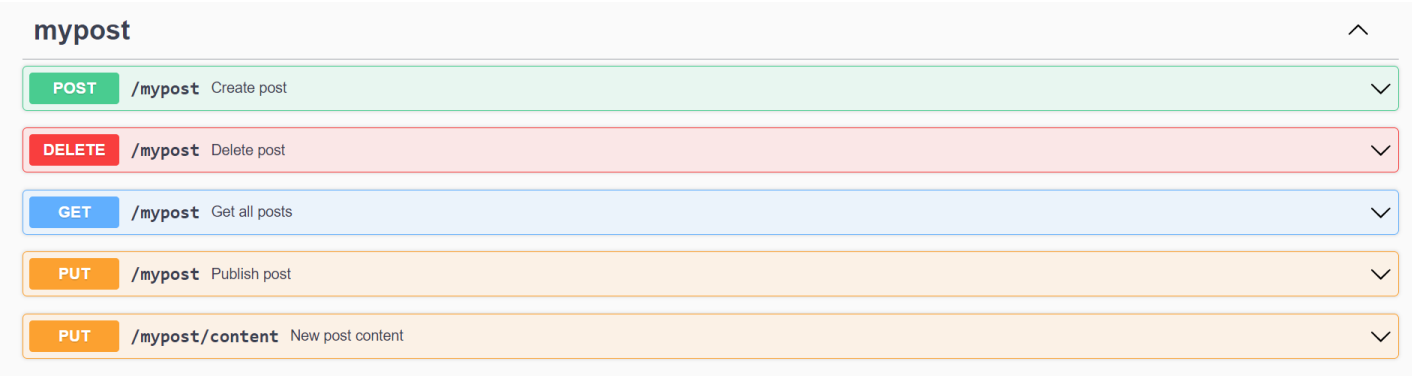


Рисунок 3.1. Программный интерфейс серверного API.

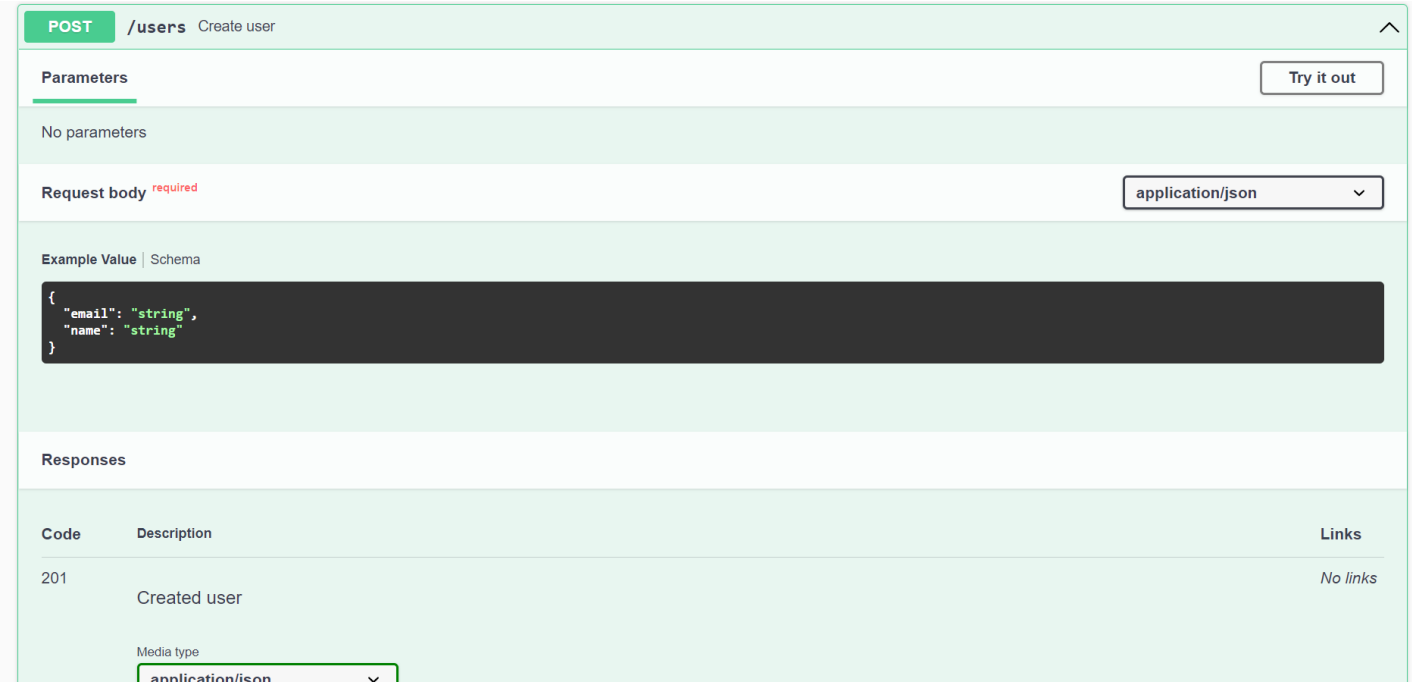


Рисунок 3.2. Программный интерфейс серверного API.



Рисунок 3.3. Программный интерфейс серверного API.

GET

/users

Get all users

⌵

🔒

Parameters

Try it out

Name	Description
page number (query)	Page number

page

Responses

Code	Description	Links
200	All Users	No links

Media type

application/json

Controls Accept header.

Рисунок 3.4. Программный интерфейс серверного API.

PUT

/users

Set user name

⌵

Parameters

Try it out

No parameters

Request body required

application/json

Example Value | Schema

```
{  "email": "string",  "name": "string"}  
```

Responses

Code	Description	Links
200	Updated User	No links

Media type

application/ison

Рисунок 3.5. Программный интерфейс серверного API.

PUT

/users/postId

Add post for user

⌵

Parameters

Try it out

No parameters

Request body

required

application/json

⌵

Example Value

Schema

```
{
  "email": "string",
  "postId": 0
}
```

Responses

Code	Description	Links
200	Updated User	No links

Media type

application/json

Рисунок 3.6. Программный интерфейс серверного API.

POST

/mypost

Create post

⌵

Parameters

Try it out

No parameters

Request body

required

application/json

⌵

Example Value

Schema

```
{
  "title": "string",
  "content": "string",
  "authorId": 0
}
```

Responses

Code	Description	Links
201	Created post	No links

Media type

application/json

Рисунок 3.7. Программный интерфейс серверного API.

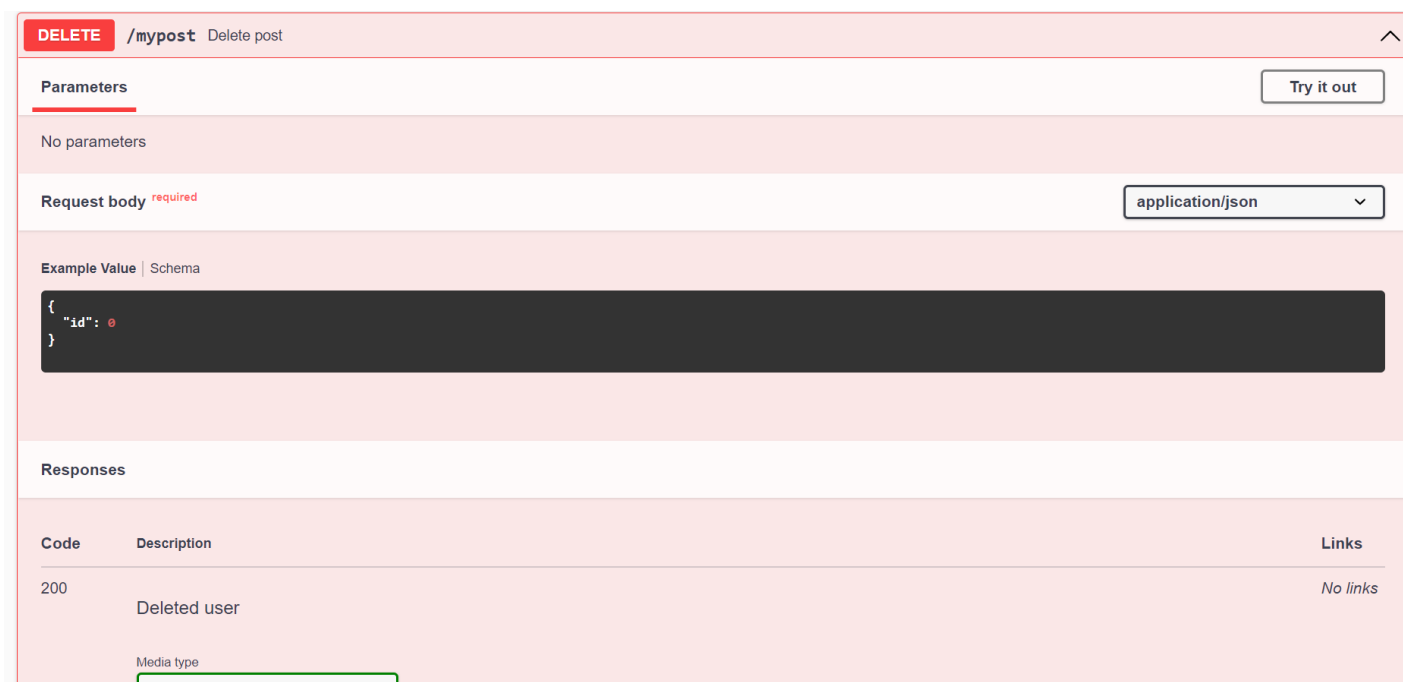


Рисунок 3.8. Программный интерфейс серверного API.

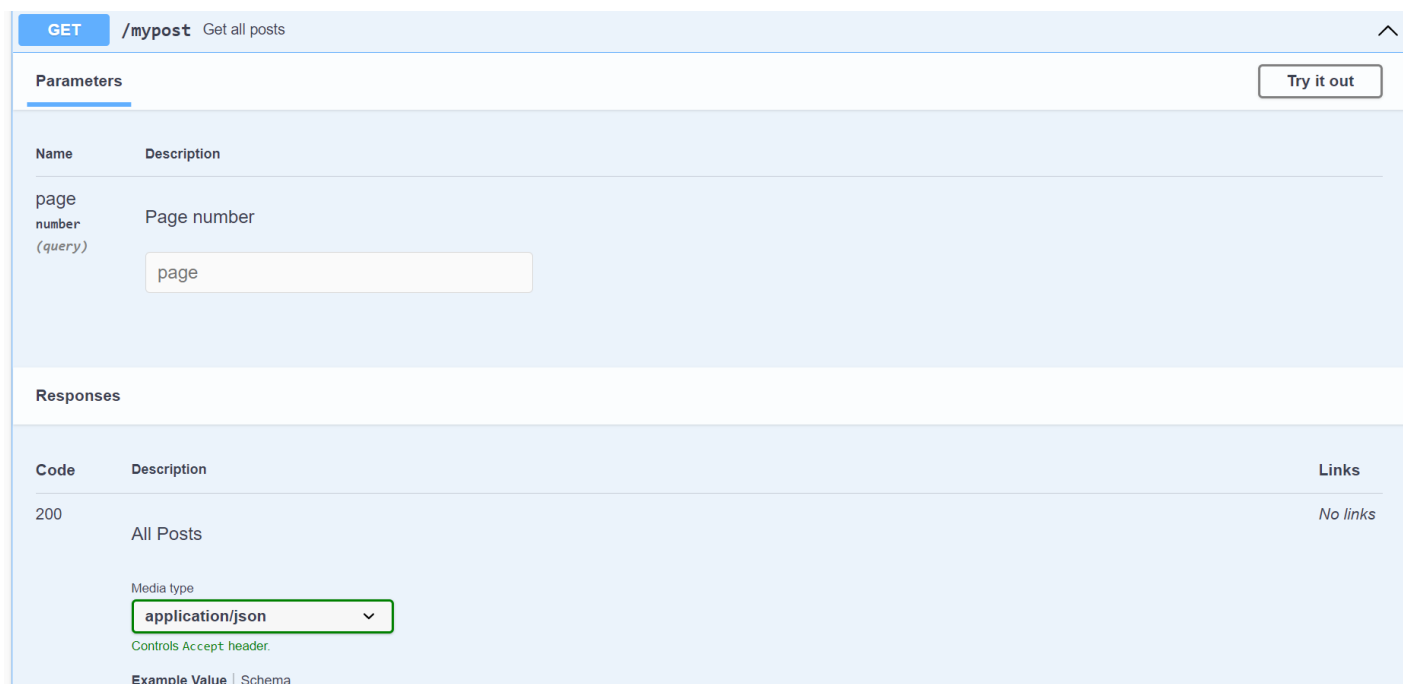


Рисунок 3.9. Программный интерфейс серверного API.

PUT

/mypost

Publish post

⌵

Parameters

Try it out

No parameters

Request body required

application/json

Example Value | Schema

```
{
  "id": 0
}
```

Responses

Code	Description	Links
200	Publish MyPost	No links

Media type

application/json

Рисунок 3.10. Программный интерфейс серверного API.

PUT

/mypost/content

New post content

⌵

Parameters

Try it out

No parameters

Request body required

application/json

Example Value | Schema

```
{
  "id": 0,
  "content": "string"
}
```

Responses

Code	Description	Links
200	New content MyPost	No links

Media type

application/json

Рисунок 3.11. Программный интерфейс серверного API.

Реализация пользовательского интерфейса

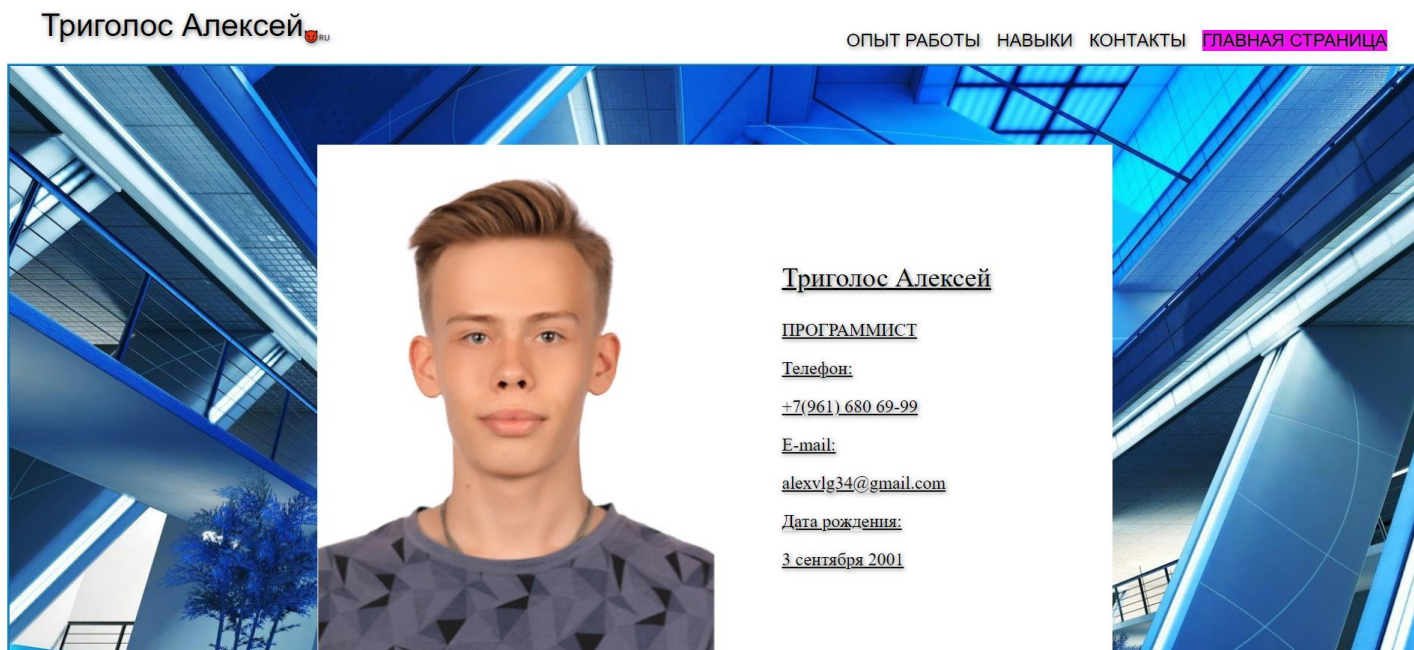


Рисунок 4.1. Главная страница.

Здесь можно посмотреть основную информацию.

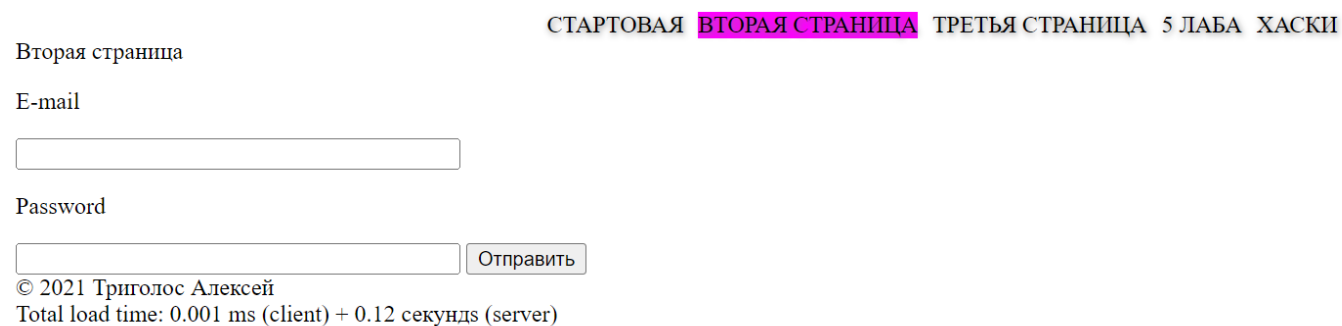


Рисунок 4.2. Страница регистрации.

При удачной регистрации пользователь сможет использовать закрытые функции

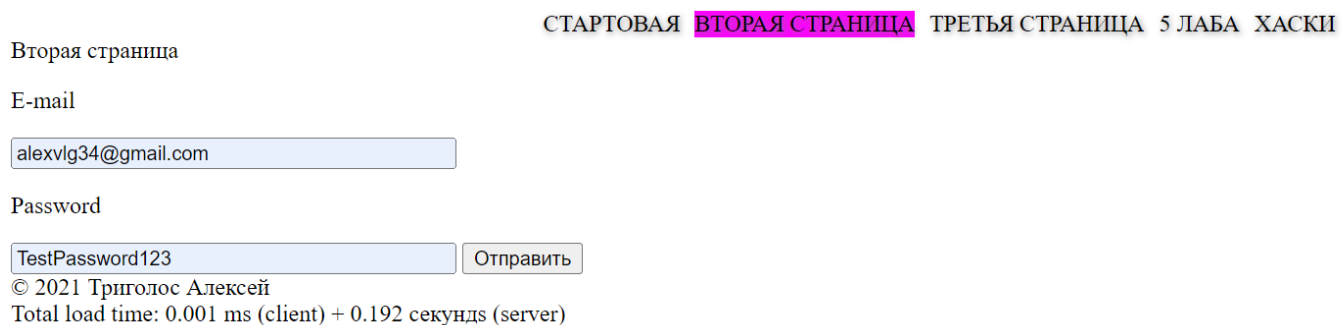


Рисунок 4.3. Вход в аккаунт.

alexvlg34@gmail.com

Вторая страница

E-mail

Password

© 2021 Триголос Алексей

Total load time: 0.001 ms (client) + 0.236 секунд (server)

Рисунок 4.4. Удачный вход в аккаунт.

Третья страница пустая.

СТАРТОВАЯ ВТОРАЯ СТРАНИЦА ТРЕТЬЯ СТРАНИЦА **5 ЛАБА** ХАСКИ

© 2021 Триголос Алексей

Total load time: 0.001 ms (client) + 0.151 секунд (server)

Рисунок 4.5. Страница с записями и добавлением предметов.

Можно добавлять предметы, ставить для них день недели, четность недели, время проведения и выбрать практика это или лекция. Добавленные предметы можно удалять.

СТАРТОВАЯ ВТОРАЯ СТРАНИЦА ТРЕТЬЯ СТРАНИЦА 5 ЛАБА **ХАСКИ**

Husky		<input type="button" value="Отправить"/>
<input type="text" value="Введите задачу"/>		
<div> <div>↓</div> <div>↑</div> </div>		
1. inventore aut nihil minima laudantium hic qui omnis	✓	<input type="button" value="Удалить"/>
2. provident aut nobis culpa	✓	<input type="button" value="Удалить"/>
3. esse et quis iste est earum aut impedit	■	<input type="button" value="Удалить"/>
4. qui consectetur id	■	<input type="button" value="Удалить"/>
5. aut quasi autem iste tempore illum possimus	■	<input type="button" value="Удалить"/>
6. ut asperiores perspiciatis veniam ipsum rerum saepe	✓	<input type="button" value="Удалить"/>
7. voluptatem libero consectetur rerum ut	✓	<input type="button" value="Удалить"/>
8. eius omnis est qui voluptatem autem	■	<input type="button" value="Удалить"/>
9. rerum culpa quis harum	■	<input type="button" value="Удалить"/>
10. nulla aliquid eveniet harum laborum libero alias ut unde	✓	<input type="button" value="Удалить"/>
11. qui ea incidunt quis	■	<input type="button" value="Удалить"/>

Рисунок 4.6. Страница с заданиями.

На странице с заданиями их можно создавать, удалять, пометить выполненными или отсортировать.

Заключение

В ходе выполнения курсовой работы был проведён анализ работы по созданию своего сайта, 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 вышеизложенные 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 "Вильямс". 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.