Частное образовательное учреждение высшего образования
«Международный Институт Дизайна и Сервиса»
(ЧОУВО МИДиС)

Кафедра математики и информатики

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ (ПО ПРОФИЛЮ СПЕЦИАЛЬНОСТИ)

Специальность: 09.02.07 Информационные системы и программирование

Квалификация программист

Форма обучения: очная

Выполнил: Макаркина В.М.

Группа П-41

Проверил: Кондаков С.А.

Челябинск 2025

СОДЕРЖАНИЕ

Введение 3

Глава I. Характеристика базы практики 5

1.1 Общая информация 5

Глава II. Задачи практики 8

2.1 Изучение целей и задач практики 8

2.2 Выбор технологий 10

2.3 Выполнение задания 12

Глава III. Анализ предметной области 19

3.1 Описание предметной области 19

3.2 Определение организации учебного процесса 20

3.3 Анализ существующих решений 22

3.4 Формирование требований к веб-приложению 28

3.5 Анализ и выбор технологий для разработки веб-приложения 30

Заключение 38

Список использованных источников 39

ВВЕДЕНИЕ

Производственная практика направлена на формирование у обучающихся практических профессиональных умений, приобретение первоначального практического опыта, реализуется в рамках профессиональных модулей образовательной программы по основным видам профессиональной деятельности для последующего освоения ими общих и профессиональных компетенций по специальности 09.02.07 Информационные системы и программирование.

Производственная практика организуется в форме практической подготовки и предусмотрена календарным учебным графиком в течение 9 недель в рамках профессиональных модулей по специальности 09.02.07 Информационные системы и программирование.

Главными целями производственной практики являются:

– комплексное освоение обучающимися всех видов профессиональной деятельности по специальности 09.02.07 Информационные системы и программирование;

– формирование общих и профессиональных компетенций, а также приобретение необходимых знаний, умений и опыта практической работы по специальности 09.02.07 Информационные системы и программирование;

– повышение качества использования в практической деятельности новых знаний и умений, стремления к саморазвитию;

– осознание социальной значимости своей будущей профессии и мотивации к выполнению профессиональной деятельности.

Задачи производственной практики:

1) ознакомиться с требованиями заказчика и спецификацией проекта;

2) спроектировать и разработать клиентскую часть веб-приложения с использованием современных технологий;

3) разработать серверную часть, реализовать CRUD-операции и добавить поддержку загрузки изображений;

4) подготовить отчет по прохождению практики и представить результаты работы.

Производственная практика является частью образовательной программы по специальности 09.02.07 Информационные системы и программирование в части освоения основных видов профессиональной деятельности:

- Проектирование и разработка информационных систем;
- Проектирование, разработка и оптимизация веб-приложений.

ГЛАВА I. ХАРАКТЕРИСТИКА БАЗЫ ПРАКТИКИ

1.1 Общая информация

Компания ООО Эшелон 370 была основана в 2023 году, с целью обеспечения надежного снабжения авиационной отрасли России качественными запчастями и комплектующими. С первых дней своего существования, компания стремилась к созданию прочных партнерских отношений с производителями и поставщиками, что позволило ей быстро занять свою нишу на рынке.

Компания занимает важную позицию на рынке поставок авиационных запчастей и комплектующих, обеспечивая надежное снабжение для ведущих авиакомпаний России. С момента своего основания, компания зарекомендовала себя как надежный партнер, предлагая широкий ассортимент продукции, включая оригинальные запчасти и компоненты для современных пассажирских самолетов.

На сегодняшний день клиентами компании являются все эксплуатанты отечественных пассажирских самолетов нового поколения, а также организации, занимающиеся их техническим обслуживанием. Среди них можно выделить как крупные авиакомпании, так и небольшие региональные перевозчики, что позволяет компании охватывать разнообразные сегменты рынка. Эшелон 370 активно сотрудничает с техническими службами и ремонтными заводами, что обеспечивает высокий уровень сервиса и оперативность в выполнении заказов.

Компания предлагает не только запчасти, но и комплектующие для различных систем самолетов, включая двигатели, системы управления и электронику. Это позволяет клиентам получать все необходимое в одном месте, что значительно упрощает процесс закупок и снижает время простоя воздушных судов. Эшелон 370 также занимается поставками специализированного оборудования для технического обслуживания самолетов, что делает ее незаменимым партнером для авиапредприятий.

Эшелон 370 располагает обширной базой данных по всем поставляемым запчастям, что позволяет быстро находить необходимую информацию и обеспечивать высокую скорость обработки заказов. Также компания активно работает над внедрением новых технологий для улучшения взаимодействия с клиентами, включая онлайн-платформы для заказа запчастей.

Важным аспектом работы компании является строгое соблюдение международных стандартов качества. Все поставляемые изделия проходят тщательную проверку на соответствие требованиям безопасности и надежности. Эшелон 370 активно внедряет инновационные технологии в процесс поставок, что позволяет не только улучшить качество продукции, но и оптимизировать логистические процессы.

В компании работает команда высококвалифицированных специалистов с большим опытом в авиационной отрасли. Это позволяет не только поддерживать высокий уровень сервиса, но и предлагать клиентам профессиональные консультации по всем вопросам, связанным с выбором и использованием авиационных запчастей.

Компания активно участвует в экологических инициативах, направленных на снижение негативного воздействия на окружающую среду. Эшелон 370 стремится использовать экологически чистые технологии в своей деятельности и поддерживает проекты по утилизации старых запчастей.

С учетом растущего спроса на авиационные услуги в России, компания планирует расширение своего ассортимента и географии поставок. Эшелон 370 рассматривает возможность выхода на международный рынок, что открывает новые горизонты для бизнеса. В рамках стратегии развития предусмотрено также увеличение объемов производства и улучшение сервисного обслуживания клиентов.

Таким образом, ООО Эшелон 370 не только обеспечивает поставки авиационных запчастей для ведущих авиакомпаний России, но и активно развивает свои внутренние процессы, внедряет инновационные решения и

заботится о своих клиентах. Это делает компанию надежным партнером в сфере авиационного обслуживания и поставок.

ГЛАВА II. ЗАДАЧИ ПРАКТИКИ

2.1 Изучение целей и задач практики

Целью производственной практики являлась разработка веб-приложения для компании ООО «Эшелон 370», занимающейся поставками авиационных запчастей и комплектующих для ведущих авиакомпаний России. Приложение должно автоматизировать процессы учета и управления запасными частями, обеспечивая удобный доступ к информации для сотрудников компании.

Для достижения поставленной цели были сформулированы следующие задачи:

Выявление ключевых требований к веб-приложению.

1. Выбор технологий и инструментов – определение оптимального стека для разработки с учетом требований к функциональности и производительности.
2. Проектирование архитектуры веб-приложения – разработка структуры базы данных, клиентской и серверной частей, а также API для обмена данными.
3. Реализация CRUD-функционала – создание интерфейсов и логики для управления каталогом авиационных запчастей (добавление, просмотр, редактирование, удаление данных).
4. Разработка системы загрузки изображений – внедрение механизма загрузки фотографий запчастей на сервер.
5. Интеграция клиентской и серверной частей – настройка взаимодействия между фронтендом и бекендом.

2.1.1 Описание задания на бэкенд

Одной из задач на практику является разработка API для обмена данными. API (англ. Application Programming Interface – программный интерфейс приложения) – это набор способов и правил, по которым различные программы общаются между собой и обмениваются данными. [23]

Типы API по способу работы:

– REST API. Его особенность в том, что он не сохраняет клиентские

данные между запросами. Обычно REST API используют для связи приложения или сайта с сервером. Приложение отправляет запрос на сайт в формате, похожем на ссылку, а ответ получает в виде набора данных.

– SOAP API. Он похож на REST, но у него более строгий принцип работы. Он запрашивает информацию о безопасности и требователен к тому, как именно приложение или сайт отправляет сообщения.

– API браузера. Этот вид программного интерфейса помогает связывать веб-сервер и браузер. Он использует ту же архитектуру, что и REST API и помогает пользователю воспроизводить музыку или анимации, а браузеру реагировать на движения мыши или команды клавиатуры. [7]

В данной работе предстоит разработать REST API. Api должен содержать:

– модель Деталь (id, название, категория, описание, количество, цена, url фотографии);

– модель Пользователь (id, имя, почта, пароль).

Также разработка серверной части подразумевает собой разработку эндпоинтов. Эндпоинт (Endpoint – конечная точка) – это само обращение к маршруту отдельным HTTP методом. Эндпоинт выполняют конкретную задачу, принимают параметры и возвращают данные Клиенту. [18]

Должны быть следующие эндпоинты:

– Пользователь (авторизация, получение токена)

– Товар (все товары, один товар, добавить товар, обновить товар, удалить товар).

2.1.2 Описание задания на фронтенд

Необходимо разработать клиентскую часть для админ-панели магазина с товарами по существующему дизайну, подключившись к REST API. По шаблону построения сайта веб-приложения делятся на:

– многостраничные (MPA) – запрос отправляется на сервер, а страница полностью обновляется в результате ответа, заменяется на новую;

– одностраничные (SPA) – после отправки запроса на сервер обновляется

часть той страницы, из которой состоит приложение, без полной перезагрузки;

– прогрессивные (PWA) – сохраняют свою функциональность, даже когда работают в режиме офлайн из-за отключения доступа к интернету. [31]

Соответственно, разрабатываемое приложение будет иметь многостраничный вид.

Первая страница – авторизация, содержит форму для входа, включающей в себя два поля ввода и кнопку.

Вторая страница – сами детали.

Третья страница – страница редактирования детали (изменения названия, цены, смена картинки и т.д.).

Четвертая страница – создание детали, добавление ее в список к основным деталям.

2.2 Выбор технологий

2.2.1 Выбор технологий для бэкенда

Есть множество языков программирования и фреймворков, которые позволяют создать RESTful API. Вот некоторые из них:

– JavaScript и Node.js с фреймворками Express, Koa или Nest;

– Python и фреймворки Flask или Django;

– Ruby и фреймворк Ruby on Rails;

– Java и фреймворки Spring, Jakarta EE или Micronaut. [17]

Было принято решение, что в работе будут использованы JavaScript и Node.js. Но в качестве фреймворка был выбран json-server. JSON Server – это библиотека, позволяющая получить полный фейковый REST API без предварительной настройки менее чем за 30 секунд. Также имеется возможность создания полноценного сервера. [3]

По заданию нужно реализовать функционал ролей и аутентификации. Для этого обычно используют jsonwebtoken. JSON Web Token (JWT) – это открытый

стандарт 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.

В простом понимании – это строка в специальном формате, которая содержит данные, например, ID и имя зарегистрированного пользователя. Она передается при каждом запросе на сервер, когда необходимо идентифицировать и понять, кто прислал этот запрос. [19]

В рамках задания необходимо загружать изображения на сервер. Для этого существует пакет multer. Multer – это ПО промежуточного слоя для Express.js. Пакет сохраняет промежуточные файлы либо в памяти, либо на диске и заполняет объект res.files для получения файлов. Пользователи этого пакета могут тонко управлять полями, предназначенными для загрузки. [10]

2.2.2 Выбор технологий для фронтенда

Есть несколько фреймворков для разработки веб-приложений:

– React – компонентная веб-разработка и библиотека Facebook для создания пользовательских интерфейсов и предоставления декларативных представлений, что делает код более предсказуемым и удобным для отладки. Позволяет создавать мощные, быстрые, удобные для пользователя и быстро реагирующие веб-приложения.

– Angular, ранее известен как Angular JS, представляет собой единую платформу веб-разработки, разработанную Google как для настольных, так и для мобильных веб-приложений. Angular нацелен на создание прогрессивных веб-приложений, предлагая внедрение зависимостей, которое помогает в сборке службы данных для приложений.

– Vue.js это еще один прогрессивный JavaScript-фреймворк с открытым исходным кодом для создания пользовательских интерфейсов, аналогичный React. Он также поддерживает декларативный рендеринг с использованием

синтаксиса шаблона для предоставления данных в DOM. [30]

Здесь же будет использоваться Next.js – фреймворк, основанный на React.js, но с определенными доработками, позволяющими ему выйти за рамки стандартных SPA-приложений. Благодаря этому существенно упрощается разработка многостраничных и гибридных веб-приложений. Дополнительно Next.js предоставляет и другие интересные возможности для разработчика, особенно, касающихся работы с серверной частью. [25]

Список дополнительных технологий, которые должны использоваться в работе:

1. Tailwind – CSS-фреймворк, предоставляющий набор готовых классов для стилизации веб-интерфейсов.

2. Zustand – библиотека управления состоянием с открытым исходным кодом.

3. React Hook Form – одна из самых популярных библиотек для создания форм. [12, 38, 28]

Для работы с Next.js нам понадобится установить Node. Node.js – это кроссплатформенная среда для разработки клиентских приложений, в основе которой лежит язык программирования JavaScript. Платформа превращает специализированный скриптовый язык JavaScript в язык общего назначения, поэтому на Node.js можно писать любые компьютерные программы. [27]

Далее, для обращения к Node в терминале мы будем использовать команды npm – популярный пакетный менеджер Node, позволяющий JS разработчикам быстро делиться пакетами. [22]

2.3 Выполнение задания

2.3.1 Описание процесса разработки бэкенда

Архитектура кода серверной части приложения построена на основе логики CRUD.

CRUD – это аббревиатура, обозначающая четыре основных операции управления данными: create, read, update, delete, то есть создание, чтение, обновление и удаление. Это действия, которые совершаются с любой информацией, в любых системах: на сайтах, в приложениях, базах данных.

Например, администратор интернет-магазина может создать новую карточку товара (create), открыть старую (read), изменить в ней информацию (update) или удалить (delete). [24]

Для работы сервера нужно разработать автоматическую генерацию данных. За это будет отвечать файл db.js:

1) Создается функция generateData, которая инициализирует объект data с ключом parts, представляющим собой пустой массив.

2) Массив заполняется данными о деталях.

Для работы сервера нужно создать контроллеры – наборы функций, которые обрабатывают HTTP-запросы и управляют взаимодействием между клиентом и сервером. Эти функции выполняют операции с данными, определяя логику обработки запросов и формируя ответы. [33]

Файл PartController.js:

1) Импортируются функция generateData имодуль jsonServer. С помощью jsonServer.router создается роутер, который будет использовать данные, сгенерированные функцией generateData.

2) Функция getAllParts получает список деталей с поддержкой поиска по названию. Заголовок X-Total-Count добавляется для указания общего количества продуктов.

3) Функция getOnePartById получает одну деталь по id, id передается в параметрах запроса. Если деталь найдена, она возвращается в ответе, иначе возвращается ошибка 404.

4) Функция addPart добавляет новую деталь. Данные детали берутся из тела запроса. URL фотографии создается на основе имени файла, загруженного с запросом.

5) Функция updatePartById обновляет информацию о детали по ее id, id передается в параметрах запроса. Обновленные данные берутся из тела запроса, и URL фотографии также обновляется, если был загружен новый файл.

6) Функция deletePartById удаляет деталь по ее id, id передается в параметрах запроса. Если удаление успешно, возвращается сообщение об успешном удалении.

Файл UserController.js:

1) Импортируются функции generateData, модули jsonServer и jsonwebtoken. Создается роутер, который будет использовать сгенерированные данные из generateData.

2) Функция generateToken создает JWT-токен для пользователя. В качестве полезной нагрузки (payload) используется ID пользователя. Токен подписывается секретом и имеет срок действия 1 день.

3) Функция login обрабатывает запрос на вход в систему. Она принимает email и password из тела запроса, ищет пользователя по email в базе данных, а затем проверяет пароль. Если пользователь не найден или пароль неверный, возвращается соответствующая ошибка. Если проверка успешна, генерируется JWT-токен, который возвращается вместе с данными пользователя.

4) Функция getMe возвращает информацию о текущем пользователе на основе userId, который должен быть извлечен из проверенного JWT. Она ищет пользователя по id, а затем извлекает данные пользователя из базы данных. Если пользователь не найден, возвращается ошибка 404. Если запрос успешен, возвращается информация о пользователе.

В сумме, этот код обеспечивает: аутентификацию пользователей с помощью электронной почты и пароля; генерацию и использование JWT для аутентификации, получение информации о текущем пользователе, включая его роли.

Для проверки JWT нужно реализовать middleware. Middleware (промежуточное или связующее программное обеспечение) – это фрагмент кода в конвейере приложения, используемый для обработки запросов и ответов. [34]

Здесь это будет файл checkAuth.js:

1) Импортируется jsonwebtoken для работы с JWT.

2) Экспортируется функция с параметрами req, res, next.

3) Из заголовка Authorization извлекается токен (без Bearer).

4) Если токен существует, он проверяется с секретным ключом. При валидности извлекается userId, добавляется в req, вызывается next(). Иначе возвращается ошибка 403 «Нет доступа».

Наконец, идет связь всех этих модулей в файле index.js:

1) Импортируются библиотеки и модули, необходимые для работы сервера, маршрутизации, авторизации и загрузки файлов.

2) Создаются экземпляры сервера и роутера. Роутер будет использовать данные, сгенерированные функцией generateData. Также добавляются стандартные middlewares jsonServer.

3) Настраивается хранилище для загружаемых файлов с использованием multer. Файлы будут сохраняться в директории ./public, и каждому файлу будет присвоено уникальное имя на основе текущей даты и времени. Подключаются стандартные middlewares jsonServer и парсер тела запроса для обработки JSON-данных.

4) Определяются маршруты для входа в систему и получения информации о текущем пользователе. Для маршрута /me используется middleware checkAuth для проверки авторизации. Определяются маршруты, связанные с деталями. Для добавления и обновления продуктов используется middleware upload для обработки загрузки файлов.

2.3.2 Описание процесса разработки фронтенда

При разработке веб-приложения необходимо учитывать несколько ключевых аспектов дизайна:

– Удобная навигация. Приложение должно иметь чёткую и интуитивно понятную структуру, а также легко находимые ссылки на важную информацию.

– Брендинг и эстетика. Приложение должно соответствовать брендингу и

отражать ценности и миссию учреждения.

– Структура контента. Макет приложения должен быть легко читаемым, с чёткими заголовками, разделами и иерархией информации.

– Высококачественные изображения и видео. Приложение должно содержать высококачественные изображения. [16]

Приложение состоит из страниц, а сами страницы из компонентов.

Компоненты позволяют разбить интерфейс на независимые части, про которые легко думать в отдельности. Во многом компоненты ведут себя как обычные функции JavaScript. Они принимают произвольные входные данные (так называемые «пропсы») и возвращают React-элементы, описывающие, что мы хотим увидеть на экране. [2]

Первая страница представляет собой форму авторизации в систему. Форма состоит из:

– Заголовок «Авторизация».

– Два поля ввода: Email (с меткой «Почта»), пароль (с меткой «Пароль»).

– Кнопка «Войти».

Визуальное оформление:

– Фон страницы состоит из двух частей: верхняя треть закрашена синим цветом (bg-blue-800), остальная часть имеет светло-серый фон (bg-slate-200).

– Вся структура выровнена по центру экрана (flex justify-center items-center min-h-screen).

– Форма входа (Form) размещена поверх фона в виде карточки с тенями (shadow-lg), закруглёнными углами и белым фоном.

Функциональность:

– Используется useState для управления состоянием email и пароля.

– При отправке формы (handleSubmit) выполняется запрос на сервер (api.post(/login, { email, password })). Если логин успешен:

1) Данные пользователя сохраняются в Zustand store (setUser и setToken).

2) Токен сохраняется в localStorage.

3) Происходит редирект на страницу /parts.

В случае ошибки выводится alert «Ошибка. Неверный логин или пароль».

Вторая страница реализует отображение списка деталей, получение данных с сервера, а также CRUD-операции (создание, обновление, удаление деталей). Основные компоненты:

1. Parts.tsx (главный компонент страницы)

– Использует usePartsStore (Zustand) для работы с деталями.

– При монтировании (через useEffect) запрашивает список деталей (getAllParts).

– Отображает заголовок (Header) и список карточек (Cards), либо текст «Загрузка...», если данные еще не получены.

2. Cards.tsx (список карточек деталей)

– Получает массив parts и рендерит Card для каждой детали.

– Оборачивает карточки в контейнер (Container) с адаптивным расположением.

3. Card.tsx (карточка отдельной детали)

– Отображает название, изображение, цену, количество на складе.

– Есть кнопки редактирования (handleEditPart) и удаления (handleDeletePart).

– Удаление детали требует подтверждения (window.confirm), после чего вызывается deletePart.

4. usePartsStore.ts (глобальное состояние деталей через Zustand)

– getAllParts() – загружает список деталей с сервера.

– getOnePart(id) – загружает одну деталь по ID.

– createPart(data) – создаёт новую деталь через POST /parts.

– updatePart(data, id) – обновляет деталь через PATCH /parts/{id}.

– deletePart(id) – удаляет деталь через DELETE /parts/{id} и обновляет parts.

Третья и четвертая страницы почти аналогичны. Одна из них – форма редактирования детали, а другая – форма для добавления.

2.3.3 Автоматизирование запуска веб-приложения в режиме разработки

Чтобы одновременно запускать сервер и клиентскую часть, нужно установить Concurrently. Concurrently – это инструмент командной строки, который позволяет запускать несколько команд или скриптов одновременно из одного окна терминала. Он часто используется в контексте запуска React и Express вместе для упрощения рабочего процесса разработки. [6]

Файл package.json описывает корневой проект echelon, который объединяет два подпроекта:

echelon-client (клиентская часть на Next.js)

echelon-server (серверная часть на JSON-server)

Связь между этими проектами обеспечивается с помощью concurrently.

Разбор scripts из package.json:

1. init – Устанавливает зависимости для всех частей проекта:

– Устанавливает зависимости в корневом проекте (npm i).

– Переходит в echelon-client и устанавливает зависимости с флагом --legacy-peer-deps (для совместимости с зависимостями).

– Затем переходит в echelon-server и тоже устанавливает зависимости (npm install).

2. next – Запускает клиент и сервер одновременно:

– cd echelon-server && npm run start – Запускает сервер.

– cd echelon-client && npm run dev – Запускает клиент.

Вывод по второй главе

По итогам второй главы было разработано веб-приложение для управления запчастями самолетов, создан фронтенд на Next.js, сервер с помощью json-server, описана работа над данными проектами.

ГЛАВА III. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

3.1 Описание предметной области

В наши дни все больше и больше современных людей постигает дары информационного пространства, а наша молодежь поголовно погрузилась в инфосферу. Практически каждое сообщество имеет свою цифровую репрезентацию, что подчеркивает важность онлайн-присутствия. Интернет предоставляет научным центрам колоссальные перспективы, стирая границы, как пространственные, так и временные, и тем самым значительно увеличивая производственные ресурсы и возможности для обмена знаниями и опытом. [36]

До того, как приняли закон «Об образовании в Российской Федерации», публикация информации вузами о своей деятельности в интернете скорее считалось правилом хорошего тона, чем как обязательной практикой. Но спустя время в сферу образования была добавлена ответственность за обеспечение создания и ведения официального сайта учреждения. Это изменение стало важным шагом к повышению прозрачности и доступности информации для студентов и их родителей. [21]

В настоящее время вузы находятся в условиях активной конкуренции друг с другом. Доступность информации на их веб-сайтах делает их более привлекательными для студентов из других городов, количество которых постепенно продолжает расти. Эффективность вузовских сайтов оценивается по тем же критериям, что и эффективность любых других интернет-порталов, и зависит от таких факторов, как содержание, навигация, видимость в поисковых системах и дизайн. Чтобы сайты полностью соответствовали требованиям регулирующих органов, их разработкой и наполнением должны заниматься квалифицированные специалисты. [39]

Однако важно отметить, что речь идет не только о сайтах-визитках, где размещается информация о специальностях и условиях поступления. Область

темы охватывает более широкие аспекты управления вузом с помощью цифровых технологий. Внедрение современных информационных систем позволяет не только оптимизировать учебный процесс, но и улучшить взаимодействие между всеми участниками образовательного процесса – студентами, преподавателями и администрацией.

Современные технологии открывают новые горизонты для образования: от дистанционного обучения до активного использования социальных сетей для обмена знаниями и опытом. Это создает уникальные возможности для студентов из разных регионов, позволяя им учиться в удобном для них темпе и формате. В результате образовательный процесс становится более доступным и инклюзивным, что особенно важно в условиях глобализации и быстрого изменения требований к знаниям и навыкам.

Таким образом, внедрение веб-приложений и информационных технологий в управление вузами не только отвечает современным требованиям общества, но и способствует созданию более эффективной образовательной среды. Это позволяет вузам не только привлекать новых студентов, но и обеспечивать высокое качество образования для всех участников процесса.

3.2 Определение организации учебного процесса

Организация учебного процесса в вузе играет ключевую роль в обеспечении качественного образования. Для дальнейшего плодотворного проектирования веб-проекта, способствующего поддержанию уровню учебной деятельности, необходимо четко разобрать такие понятия, как учебный процесс и его организация. Также нужно определить его структуру, основные компоненты и процессы.

Учебный процесс в вузе представляет собой систему мероприятий, направленных на достижение образовательных целей в соответствии с государственными стандартами и внутренними регламентами учебного

заведения. Он основан на количественных и качественных подходах к уровню образования студентов [9].

Организация учебного процесса – это система планирования, реализации и контроля обучения, включающая нормативные положения, методические подходы и педагогические технологии, обеспечивающие эффективное взаимодействие между преподавателем и студентом.

Ключевыми элементами учебного процесса являются:

- Учебный план – определяет перечень и последовательность изучаемых дисциплин, устанавливает объем учебной нагрузки и регламентирует образовательный процесс.
- Учебная программа – конкретизирует содержание дисциплин, цели и задачи изучения, формы контроля знаний.
- Учебные дисциплины, которые группируются в модули для структурирования образовательного процесса.

Процесс обучения включает формирование знаний, навыков и умений студентов, промежуточную и итоговую аттестацию. Его основой является непосредственное взаимодействие преподавателя и обучающегося, где преподаватель передает знания, а студент их усваивает.

Основной целью организации образовательного процесса является достижение запланированных образовательных результатов, соответствующих требованиям федеральных образовательных стандартов и ожиданиям работодателей. Для этого используются:

- Определение целей и задач обучения.
- Применение методологических подходов и дидактических принципов.
- Развитие теоретических знаний и практических навыков.
- Контроль и оценка успеваемости студентов.

Таким образом, организация учебного процесса обеспечивает не только реализацию образовательных программ, но и их соответствие современным требованиям профессионального рынка [37].

3.3 Анализ существующих решений

Для создания учебной среды существуют различные платформы, включающие в себя технологические и педагогические инструменты для поддержки образовательного процесса в виртуальном формате. Система управления обучением (LMS) представляет собой программное решение, которое помогает организовывать и контролировать процесс обучения и преподавания. Такая система позволяет регистрировать студентов, структурировать курсы в каталоге, отслеживать их успеваемость и информировать преподавателей или кураторов о ходе обучения. Кроме того, LMS дает возможность не только создавать и хранить учебные материалы, но и использовать их повторно в разных дисциплинах и для различных групп студентов. К таким универсальным ресурсам относятся текстовые документы, презентации, анимации и изображения. [13]

3.3.1 Программный продукт 1С:Университет

1С:Университет – это комплексная система автоматизации управления образовательным учреждением. Она предназначена для решения задач административного и учебного управления. Программный продукт разработан на технологической платформе 1С:Предприятие 8.3. [1]

Функционал:

– Реализация различных моделей ведения приемной кампании (обработка личных дел, ранжирование абитуриентов и их зачисление).

– Ведение электронных учебных планов.

– Управление студенческими данными (учет групп, индивидуальных достижений, посещаемости).

– Автоматизация расписания занятий и экзаменов.

– Ведение финансового учета (оплата за обучение, бюджетирование).

– Электронный документооборот (зачетки, ведомости, приказы).

– Интеграция с другими сервисами 1С (1С:Бухгалтерия, 1С:Зарплата и кадры).

Решение позволяет автоматизировать учет, хранение, обработку и анализ информации об основных процессах высшего учебного заведения: поступление в вуз, обучение, оплата за обучение, выпуск и трудоустройство выпускников, расчет и распределение нагрузки профессорско-преподавательского состава, деятельность учебно-методических отделов и деканатов, поддержка ФГОС-3 и уровневой системы подготовки (бакалавр, специалист, магистр) на уровне учебных планов и документов государственного образца об окончании вуза, формирование отчетности.

Интерфейс 1С:Университет выполнен в классическом стиле продуктов 1С: панель навигации слева, основное рабочее поле в центре. Дизайн достаточно устаревший, ориентирован на бухгалтерский стиль, что делает систему не самой удобной для рядового пользователя (рис. 3.1).

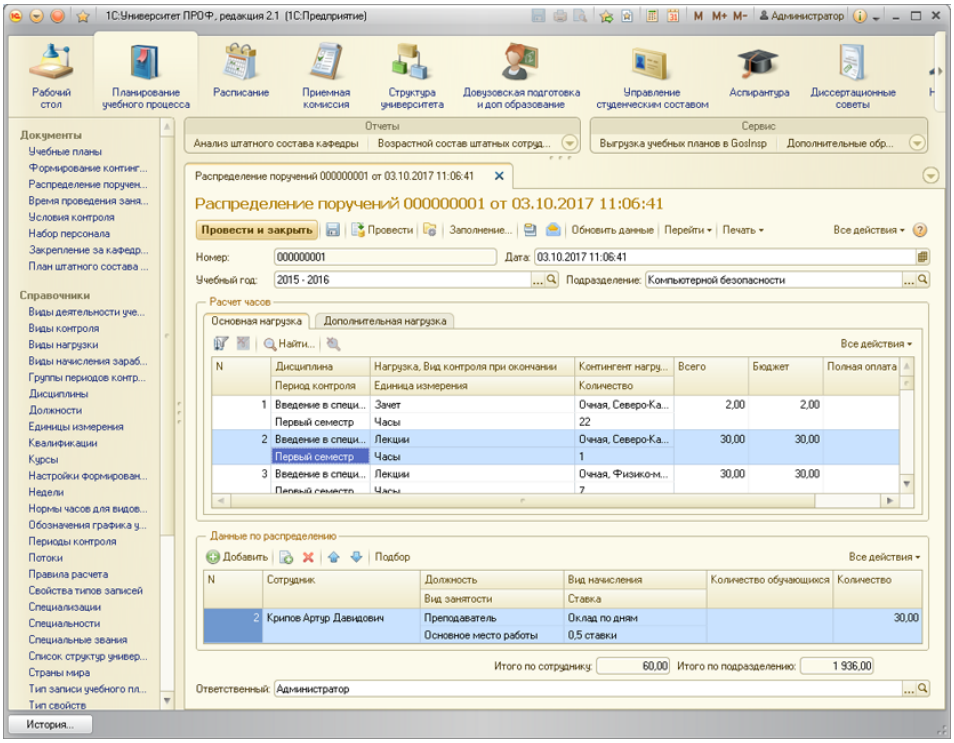

Рисунок 3.1 – Планирование учебного процесса в 1С: Университет

3.3.2 LMS Moodle

Moodle – одна из самых популярных систем управления обучением (LMS), используемая во многих вузах для организации дистанционного обучения и электронных курсов.

Функционал:

– Разработка интерактивных курсов. Moodle поддерживает видео, аудио, тексты, презентации, SCORM-материалы и изображения, позволяя создавать пошаговые курсы с разграниченным доступом.

– Анкетирование, опросы и тестирование. Позволяют оценивать знания студентов и получать обратную связь. В тестах настраиваются дедлайн, время, количество попыток, оценки и доступ.

– Средства коммуникации с учащимися. Включают чат, видеочат и комментарии для взаимодействия с учащимися.

– Хранилище учебных материалов. Moodle также выполняет функцию базы знаний, обеспечивая постоянный доступ к загруженным учебным ресурсам для студентов.

– Статистика успеваемости и автоматизированные отчеты. Система дистанционного обучения (СДО) отслеживает посещаемость, анализирует успехи студентов и формирует отчеты. Преподаватели могут просматривать время, затраченное на прохождение курсов, а также ошибки, допущенные в тестах. Некоторые плагины предоставляются бесплатно, другие – на платной основе.

– Мобильное приложение. Moodle поддерживает обучение и преподавание на планшетах и смартфонах. В мобильной версии доступны все основные функции, а также настройка push-уведомлений.

– Автоматизированная рассылка сообщений. Позволяет планировать отправку сообщений студентам без использования сторонних сервисов. [15]

Интерфейс Moodle относительно удобен, но сложен в настройке. Он имеет множество функций, однако требует адаптации и настройки под нужды

конкретного учебного заведения. Навигация может быть сложной для неподготовленного пользователя (рис. 3.2).

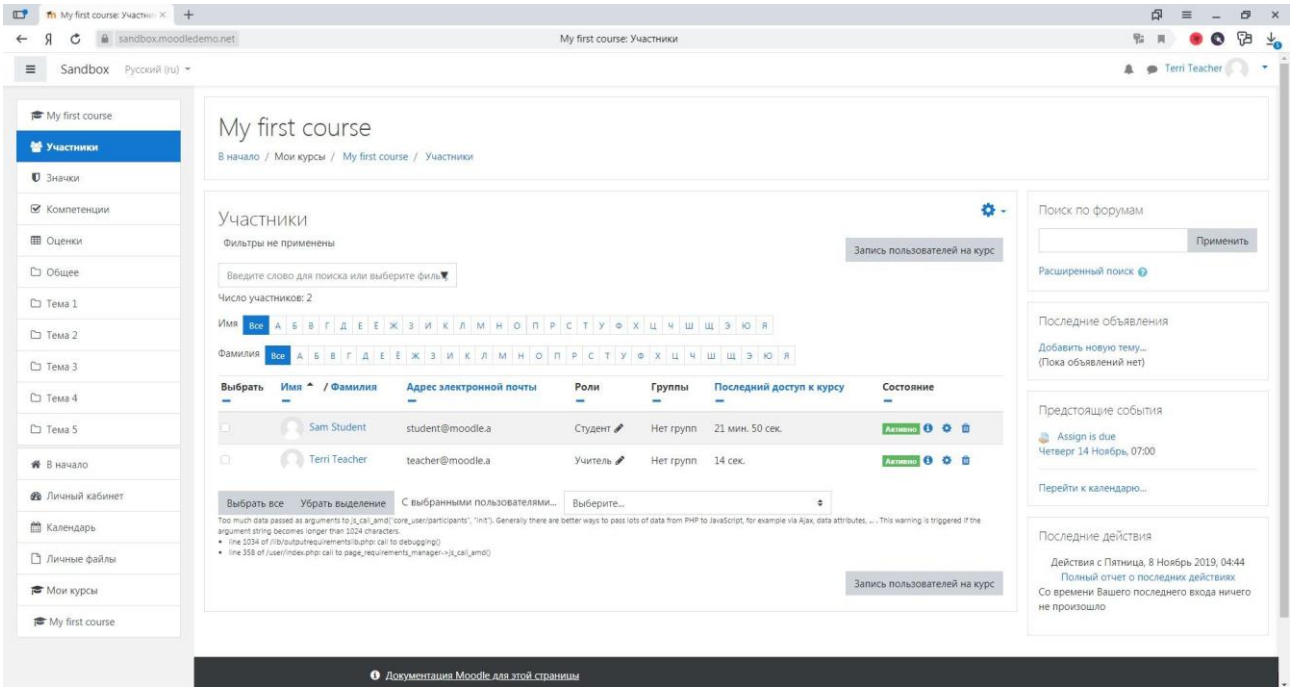

Рисунок 3.2 – Отображение участников курса в Moodle

3.3.3 LMS HSE

LMS HSE – это автоматизированный сервис управления учебными процессами со встроенной платформой дистанционного обучения, разработанная специально для ВШЭ. Система представляет собой облачное решение, предназначенное для автоматизации процессов обучения по всем направлениям производственной деятельности.

Каждый студент может получить персонализированную информацию о:

– индивидуальном учебном плане;

– успеваемости (электронная зачётная книжка);

– личном расписании.

LMS предоставляет следующие возможности:

– обмен сообщениями между студентами и преподавателями;

– доступ к учебным материалам, загруженным преподавателями;

– ознакомление с программами учебных дисциплин;

– проведение тестирования для промежуточного и итогового контроля знаний;

– организация сдачи проектов и домашних заданий.

Кроме того, в системе LMS реализованы общеуниверситетские сервисы:

– проверка студенческих работ на заимствования с помощью модуля «Антиплагиат»;

– самостоятельное планирование образовательной траектории через модули «Дисциплины по выбору» и «Выбор майнора», что позволяет студентам подавать заявки на курсы как из своей образовательной программы, так и из других программ, включая общеуниверситетские проекты;

– доступ к модулю «Оцени свои курсы», который формирует анонимные анкеты для студентов с возможностью оценки преподавателей по различным критериям;

– подача темы курсовой работы для согласования с руководителем;

– оформление заявок на перевод в другую образовательную программу, получение стипендии и прохождение практики.

Кроме того, система LMS помогает бороться со списыванием, риск которого увеличивается в период дистанционного обучения. В LMS организуется проверка студенческих письменных работ на уровень заимствований посредством системы «Антиплагиат». [11]

Интерфейс: система имеет современный и минималистичный дизайн, удобную навигацию и адаптивность под мобильные устройства. Основной упор сделан на удобство работы с образовательным контентом, а не на административные функции (рис. 3.3).

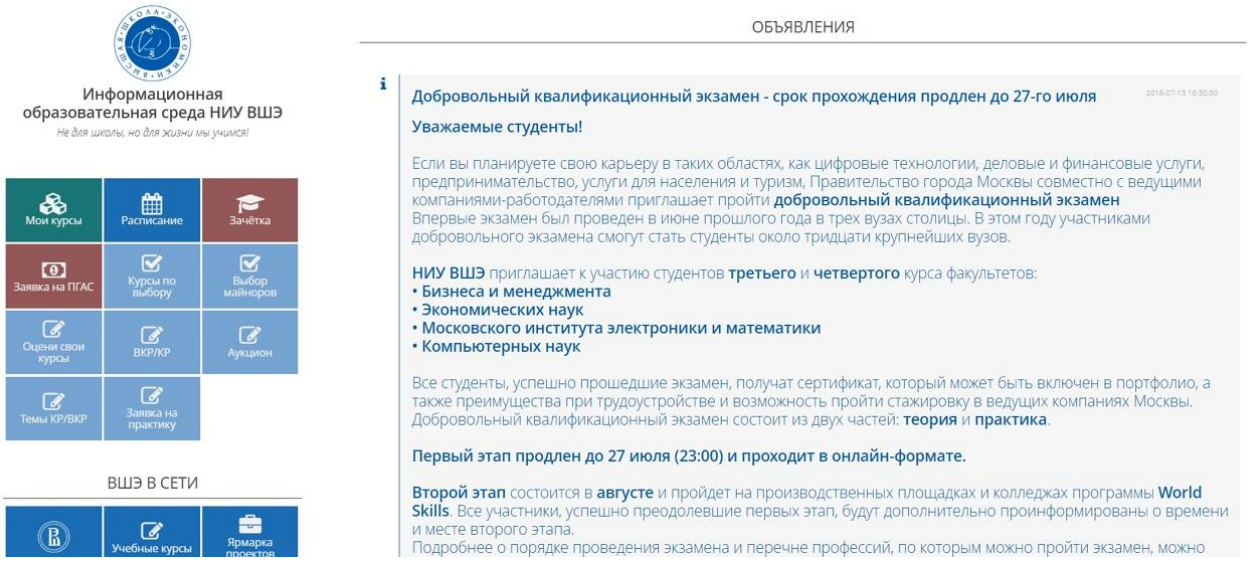

Рисунок 3.3 – Часть интерфейса LMS HSE

В таблице 1 представлены основные характеристики рассматриваемых систем.

Таблица 1 – Сравнительный анализ представленных решений

Характеристика	1С:Университет	Moodle	LMS HSE
Основное назначение	Административное управление	Дистанционное обучение	Управление учебным контентом
Гибкость настроек	Средняя	Высокая	Высокая
Дружелюбность интерфейса	Низкая	Средняя	Высокая
Функции электронного обучения	Ограниченные	Полноценные	Полноценные
Поддержка мобильных устройств	Частичная	Полная	Полная
Интеграция с другими системами	Хорошая	Отличная	Отличная

На основе анализа можно сделать вывод, что 1С:Университет лучше всего подходит для административного управления вузом, Moodle – для организации дистанционного обучения и управления учебными материалами, а LMS HSE оптимально решает задачи взаимодействия студентов и преподавателей в рамках учебного процесса.

Цель нашей работы разработать так называемую социальную сеть, в которой студенты смогут без проблем взаимодействовать друг с другом, организовывать мероприятия, собирать группы по интересам, при этом не отвлекать от учебного процесса.

3.4 Формирование требований к веб-приложению

Разработка веб-приложения для организации учебного процесса в вузе требует четкого определения требований, которые обеспечат его функциональность, удобство использования и безопасность.

Для организации обучения через Интернет применяются технологии Web 2.0, включающие различные инструменты и сервисы, такие как блоги, мгновенные сообщения, социальные сети и справочные материалы, необходимые для образовательного процесса. Эти технологии не только расширяют возможности преподавателей в создании и распространении учебного контента, но и способствуют развитию новых форм взаимодействия между студентами и преподавателями.

Дизайн играет важнейшую роль в электронном обучении. Эффективность онлайн-образования определяется качеством учебного материала и его структурой, и оба эти аспекта должны быть тщательно проработаны. Система управления обучением должна быть интуитивно понятной, адаптивной, доступной и минимизировать препятствия, обеспечивая разнообразность. Образовательные платформы могут сделать процесс обучения более удобным и увлекательным, если предоставляют инструменты для самостоятельного развития. [8]

Функциональные требования описывают, какие возможности должно предоставлять веб-приложение:

– Регистрация и аутентификация пользователей (студенты, преподаватели, администраторы) с использованием логина и пароля.

– Управление пользователями: добавление, редактирование, удаление учетных записей.

– Формирование расписания.

– Возможность просмотра расписания занятий пользователями.

– Функционал загрузки и проверки сданных работ студентами.

– Выставление и просмотр оценок.

– Обмен сообщениями между пользователями.

– Генерация отчетов и статистики по успеваемости студентов.

– Оповещения и уведомления о предстоящих событиях и изменениях в расписании.

Нефункциональные требования определяют качество работы веб-приложения:

– Производительность – время отклика системы не должно превышать 3 секунд при стандартных запросах.

– Масштабируемость – возможность поддержки увеличивающегося количества пользователей без значительного снижения производительности.

– Кроссплатформенность – поддержка различных устройств и браузеров.

– Удобство использования – интуитивно понятный интерфейс, адаптированный под различные категории пользователей.

Веб-приложение будет использоваться различными категориями пользователей, каждая из которых обладает определенными ролями и возможностями:

– Администратор – управляет пользователями, назначает роли, регулирует доступ к функциям, управляет учебными планами и расписанием.

– Преподаватель – добавляет материалы к занятиям, проверяет сданные работы, выставляет оценки, отправляет уведомления студентам.

– Студент – просматривает расписание, отправляет работы, получает оценки, взаимодействует с преподавателями через систему сообщений.

Веб-приложение должно обеспечивать высокий уровень защиты данных и предотвращать несанкционированный доступ:

– Аутентификация и авторизация – использование JWT-токенов и bcrypt для шифрования паролей.

– Разграничение прав доступа – четкое разграничение прав пользователей в зависимости от их ролей.

– Защита данных – шифрование передаваемой информации, защита от SQL-инъекций и межсайтовых атак (XSS, CSRF).

3.5 Анализ и выбор технологий для разработки веб-приложения

3.5.1 Выбор системы управления базами данных (СУБД)

Система управления базами данных (СУБД) – это набор программ, которые управляют структурой БД и контролируют доступ к данным, хранящимся в БД.

Допустим, у нас есть большая библиотека с книгами, но все эти книги зашифрованы. А СУБД – это некий библиотекарь, и только он способен предоставить доступ к книгам. Когда нужно найти какую-то информацию, вместо того, чтобы копаться в зашифрованных книгах, можно просто спросить библиотекаря, и он быстро найдет запрашиваемые данные, переводит в понятный вид и предоставляет пользователю. Так работает СУБД: она принимает запросы от программ (или пользователей), ищет нужную информацию в хранилище, обрабатывает её и выдаёт результат.

Когда между программами пользователя и базой данных есть система управления базами данных, это даёт несколько важных преимуществ.

Во-первых, СУБД позволяет нескольким приложениям и пользователям одновременно работать с одними и теми же данными. Во-вторых, она объединяет информацию из разных источников в одно целостное хранилище, что делает работу с данными удобнее. Вот основные плюсы использования СУБД:

– Лучший обмен данными. СУБД создаёт удобную среду, в которой пользователи могут безопасно работать с информацией.

– Повышенная безопасность. Чем больше людей получают доступ к данным, тем выше риск утечки информации. СУБД помогает защищать данные и соблюдать правила безопасности.

– Хорошая интеграция данных. Централизованное управление информацией позволяет видеть полную картину деятельности организации.

– Быстрый доступ к данным. СУБД обрабатывает запросы пользователей и мгновенно выдаёт нужную информацию, например, для просмотра или обновления данных. [26]

Далее, нам необходимо определиться с типом (модели) базы данных. Модель данных – это набор правил, которые определяют, как хранится и организуется информация в базе данных. Она включает в себя структуру данных, возможные операции с ними, а также ограничения, которые обеспечивают правильные связи между данными и их допустимые значения. Мы можем воспользоваться данным инструментом моделирования произвольной предметной области. [14]

Основные типы баз данных и их краткая характеристика:

– Реляционные (SQL) базы данных. Основаны на таблицах с четко определенными связями.

– Графовые базы данных. Ориентированы на хранение данных в виде узлов и связей между ними.

– Объектно-ориентированные базы данных. Хранят данные в виде объектов, как в ООП.

В разработке нашего веб-приложения предстоит спроектировать четкую структуру и учитывать множество связей между сущностями. И для этих целей отлично подойдет такой тип базы данных, как реляционная.

В реляционных базах данных используется следующая терминология:

1) Таблица – хранилище данных.

2) Записи – строки таблицы.

3) Поля – столбцы таблицы.

4) Имена полей – имена столбцов таблицы.

5) Ключ – уникальное поле или совокупность полей таблицы, позволяющее однозначно идентифицировать запись. Каждая таблица РБД должна иметь ключ, который называется первичным ключом. [20]

В реляционных базах данные имеют четкую структуру и связи. Это важно в контексте темы диплома, так как в учебном процессе много связанных сущностей: студенты, группы, преподаватели, дисциплины, расписание, оценки. Также реляционная модель позволяет использовать ключи и нормализацию.

Реляционная модель гарантирует, важную в проектировании и разработке, целостность данных. Благодаря этому, как пример, студент не сможет быть записан на несуществующий курс, а оценка не сможет быть выставлена вне контекста предмета.

При помощи SQL-запросах, которые поддерживает реляционная модель, позволено быстро фильтровать и находить необходимую информацию, например, оценки всех студентов группы за определенный период.

В таблице 2 поведены альтернативные типы БД и почему они не подходят.

Таблица 2 – Сравнение типов БД

Тип БД	Преимущества	Недостатки для учебного процесса
Графовая	Хороша для социальных сетей, анализа связей	Неэффективна для четко структурированных данных (студенты, группы, расписание)
Объектно-ориентированная	Хорошо подходит для работы с объектами в коде	Плохая совместимость с SQL и сложная масштабируемость
Документо-ориентированная	Гибкость структуры, быстрая запись	Отсутствие строгих связей между данными, сложность обеспечения ACID-транзакций
Функциональная	Позволяет работать с версионированием данных	Сложная интеграция, низкая популярность

Определившись с моделью базы данных, мы наконец можем перейти к выбору СУБД. По нашему мнению, при разработке веб-приложений важно учитывать интегрированность используемых технологий. Говоря простым языком, стоит изучить существующие экосистемы фреймворков, которые предлагают упрощенное внедрение в другие продукты. Исходя из данной информации и технологий, которые будут описаны далее, наш выбор пал СУБД PostgreSQL.

PostgreSQL – СУБД, которая использует реляционную модель для своих баз данных и поддерживает стандартный язык запросов SQL. PostgreSQL предоставляет множество различных возможностей, достаточно надежна и имеет хорошие характеристики по производительности. [35]

Обоснование выбора:

– PostgreSQL гарантирует целостность данных даже при сбоях, что крайне необходимо для надежного хранения информации о студентах, которое требует учебные процессы.

– PostgreSQL поддерживает сложные связи, многотабличные запросы и каскадные связи.

– PostgreSQL – open-source СУБД с широким сообществом, где легко найти решения проблем и получить поддержку.

– PostgreSQL отлично работает с TypeORM или Prisma, что упрощает интеграцию с NestJS.

Есть другие альтернативы СУБД, но по разным причинам они не подходят для наших задач.

– MySQL – хорош для небольших нагрузок, но уступает PostgreSQL в возможностях транзакций и JSONB.

– MongoDB – можно использовать для хранения больших объемов неструктурированных данных (например, сообщений и логов), но основную базу лучше строить на реляционной СУБД.

– MS SQL Server – мощная, но платная, что делает её менее привлекательной для вузов.

3.5.2 Выбор серверного фреймворка

Выше мы определились с СУБД и упомянули NestJS, как фреймфорк, который прекрасно интегрирует в себя PostgreSQL.

Так как приложение разрабатывается для браузеров, логично использовать серверные технологии, совместимые с JavaScript или его типизированной вариацией TypeScript. Это упростит разработку и дальнейшую поддержку кодовой базы.

На основе данных критериев рассмотрим три возможных подхода к разработке серверной части:

– Использование нативного Node.js позволяет работать с сервером без дополнительных фреймворков, но требует значительных усилий для настройки и реализации базовой функциональности и менее удобен в поддержке по сравнению с готовыми решениями.

– Использование фреймворка Express.js – Легкий и гибкий фреймворк с лхорошей документацией. Он широко используется в индустрии, что облегчает поиск решений и поддержку, но требует дополнительной настройки для работы с типизированными данными и обработки ошибок и не относится к реляционным базам данных.

– Использование фреймворка NestJS – основан на TypeScript, что упрощает работу с типизацией данных. Предоставляет модульную архитектуру и встроенную поддержку множества технологий.

В данном случае Node.js не подходит из-за высокой сложности настройки и поддержки. Express.js является более сбалансированным решением, однако уступает NestJS по уровню встроенных возможностей и удобству разработки.

Поэтому для серверной части веб-приложения выбран NestJS – фреймворк для разработки эффективных и масштабируемых серверных приложений на Node.js. Данный фреймворк использует прогрессивный JavaScript с полной

поддержкой TypeScript и сочетает в себе элементы объектно-ориентированного, функционального и реактивного функционального программирования. [4]

3.5.3 Выбор фронтенд-фреймворка

Мир веб-разработки полон огромным множеством различных технологий для разработки интерфейсов, которые реализуют задуманную логику. С каждым годом появляются новые фреймворки, которые основаны на своих предшественниках и специализируются на оптимизации проектах.

Веб-фреймворк – это основа для создания веб-приложений. Он задает структуру проекта, определяет принципы разработки и предоставляет необходимые инструменты.

Фронтенд-фреймворки предназначены для разработки пользовательского интерфейса. В отличие от серверных решений, они не управляют бизнес-логикой, а работают исключительно в браузере. С их помощью можно разрабатывать и улучшать интерфейсы, создавать анимации и реализовывать одностраничные приложения (SPA).

К популярным фронтенд-фреймворкам относятся:

– Angular

– Vue.js

– Svelte

– React – хотя технически это библиотека, его функциональные возможности и популярность позволяют сравнивать его с полноценными фреймворками.

Все перечисленные инструменты основаны на языке JavaScript. [32]

Next.js – это основанный на React фреймворк, предназначенный для разработки веб-приложений, обладающих функционалом, выходящим за рамки SPA, т.е. так называемых одностраничных приложений. Он позволяет создавать веб-приложения с улучшенной производительностью и улучшенным пользовательским опытом с помощью дополнительных функций

предварительного рендеринга, таких как полноценный рендеринг на стороне сервера (SSR) и статическая генерация страниц (SSG). [3, 29]

Технология SSR (Server-Side Rendering) ускоряет загрузку страниц и улучшает SEO, что полезно для публичных страниц. Static Site Generation можно использовать для редко изменяющихся страниц, например, учебных планов.

Данный фреймворк можно использовать, как фулстек инструмент, так как он поддерживает серверную разработку. Соответственно, его использование гарантирует легкую интеграцию с NestJS, используя бибилотеку axios для запросов к серверу.

Next.js поддерживает динамическую маршрутизацию и позволяет избавиться от App Router – инструмента от React. Это облегчит создание страниц, например, для профилей студентов или расписания групп. Встроенный middleware позволяет настраивать доступ, соответственно во время разработки без проблем получится запретить студентам доступ к админ-панели.

У Next.js большая экосистема, начиная от его разработчика Vercel – инструмента для развертывания веб-приложний, заканчивая такими библиотеками, как TailwindCSS (легкая стилизация интерфейса, обеспечение единого минималистичного стиля) и другими. Также в разработке планируется использовать менеджер состояний Zustand. При этом Next.js традиционно используют с TypeScript, что дает абсолютную уверенность в строгой типизации, повышении надежности кода и предотвращении ошибок на ранних этапах разработки.

Выводы по третьей главе

В данной главе был проведен анализ предметной области, рассмотрены особенности организации учебного процесса в вузе и определены ключевые сущности, участвующие в нем. Проанализированы существующие решения для автоматизации учебного процесса, выявлены их преимущества и недостатки, что позволило обосновать необходимость разработки нового веб-приложения.

На основе проведенного исследования были сформированы требования к разрабатываемому веб-приложению, включая функциональные и нефункциональные характеристики, а также определены роли пользователей. Особое внимание было уделено требованиям безопасности, учитывая работу с персональными данными.

Также был осуществлен выбор технологий для реализации веб-приложения, с учетом критериев производительности, масштабируемости и удобства разработки. Принятые решения обеспечивают соответствие требованиям и позволяют создать эффективную систему для организации учебного процесса в вузе.

ЗАКЛЮЧЕНИЕ

В ходе производственной практики были успешно выполнены следующие задачи: освоены основы будущей профессиональной деятельности и изучены современные информационные технологии получения и обработки данных. Также приобретены практический опыт, знания и умения для самостоятельного решения задач. Развиты практические навыки в разработке веб-приложений с помощью библиотеки Next.js, развиты практические навыки в разработке серверной части веб-приложения с помощью библиотеки json-server, развиты практические навыки в тестировании веб-приложения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. 1С:Университет [Электронный ресурс]. – Режим доступа: https:// solutions.1c.ru/catalog/university/features

2. Абрамов Д. А. Компоненты и пропсы [Электронный ресурс] // Блог react.dev – 2023. – Режим доступа: https://ru.legacy. reactjs.org/docs/componentsand -props.html

3. Агапов И.О. Next.js: подробное руководство [Электронный ресурс] // Хабр. – 2021. – Режим доступа: https://habr.com/ru/companies/timeweb/articles/ 588498/

4. Агапов И.О. Руководство по NestJS [Электронный ресурс] // Хабр. – 2022. – Режим доступа: https://habr.com/ru/companies/timeweb/articles/663234/

5. Агапов И.П. Шпаргалка по JSON Server [Электронный ресурс] // Блог MyJavaScript – 2024. – Режим доступа: https://myjs.org/docs/cheatsheet/json-server/

6. Алексеев В. А. Одновременный запуск React и Express [Электронный ресурс]. – Режим доступа: https://blog.logrocket.com/runningreact-express-concurrently/

7. Ахметшина И.С. Типы API по способу работы [Электронный ресурс] // Журнал КонтурФокус – 2023. – Режим доступа: https://focus. kontur.ru/site/news/46960-chto_takoe_api_i_kak_eto_rabotaet#header_46960_5

8. Байкова М.К. Сравнение современных LMS платформ дистанционного обучения [Электронный ресурс] // Международный журнал гуманитарных и естественных наук. – 2024. – №10. – Режим доступа: https:// cyberleninka.ru/article/n/sravnenie-sovremennyh-lms-platform-distantsionnogo-obucheniya

9. Башиева А.Х., Рашидханова П.Б., Мусаева С.Д. Внедрение цифровых образовательных ресурсов в образовательный процесс вуза [Электронный ресурс] // Проблемы современного педагогического образования. – 2023. – №81. – Режим доступа: https://cyberleninka.ru/article/n/vnedrenie-tsifrovyh-obrazovatelnyh-resursov-v-obrazovatelnyy-protsess-vuza

10. Воробьев И.С. Multer – npm-пакет для обработки файлов [Электронный ресурс] // Блог Хабр – 2020. – Режим доступа: https://habr.com/ru/companies/ruvds/articles/505760/

11. Грошева О. В. Платформы НИУ ВШЭ [Электронный ресурс] // Дзен. – 2021. – Режим доступа: https://dzen.ru/a/YYmQIH5oxg-olWhX

12. Дьяков Н.С. Tailwind не только для MVP [Электронный ресурс] // Блог Хабр – 2023. – Режим доступа: https://habr.com/ru/articles/763126/

13. Исаева Е.С. Современные LMS платформы дистанционного обучения: анализ и сравнение [Электронный ресурс] // Педагогика. Вопросы теории и практики. – 2021. – №6. – Режим доступа: https://cyberleninka.ru/article/n/sovremennye-lms-platformy-distantsionnogo-obucheniya-analiz-i-sravnenie

14. Карпова И.П. Базы данных. Курс лекций и материалы для практических заданий. – Учебное пособие. – М.: Питер, 2013. – 240 с.

15. Киргинцева Н. Н. Система Moodle: что это такое и как работает [Электронный ресурс] // Мтс-линк. – 2024. – Режим доступа: https://mts-link.ru/blog/moodle/

16. Кирсанов Д. А. Веб-дизайн – СПб: Символ-Плюс, 2001 – 376 с.

17. Кодов А.Е. Как создать RESTful API [Электронный ресурс] // Блог SkyPro – 2023. – Режим доступа: https://sky.pro/media/kak-sozdat-restful-api/

18. Комаров К.Б. Базовые понятия (знания) в REST API [Электронный ресурс] // Блог wp-kama – 2020. – Режим доступа: https://wp-kama.ru/handbook/rest/basic#routes-endpoints

19. Кочетов П.Е. JWT – как безопасный способ аутентификации и передачи данных [Электронный ресурс] // Блог vc.ru – 2020. – Режим доступа: https://vc.ru/dev/106534-jwt-kak-bezopasnyi-sposob-autentifikacii-i-peredachidannyh

20. Крикунов М.М., Поручиков А.Н. Основы баз данных: учебное пособие – Самара: Издательство Самарского университета, 2021. – 84 с.

21. Куликов И.А. Разработка проекта современного сайта факультета вуза [Электронный ресурс] // Гуманитарная информатика. – 2015. – №9. – Режим

доступа: https://cyberleninka.ru/article/n/razrabotka-proekta-sovremennogo-sayta-fakulteta-vuza

22. Кунгуров М. В. Что такое npm? [Электронный ресурс] // Блог proglib.io – 2020. – Режим доступа: https://proglib.io/p/chto-takoe-npm-gayd-ponode-package-manager-dlya-nachinayushchih-2020-07-21

23. Кучерявый Е.А. Что такое API и как он работает [Электронный ресурс] // Блог Skillbox Media – 2022. – Режим доступа: https://skillbox.ru/media/code/chto_takoe_api/

24. Левина Е. Э. Что такое CRUD [Электронный ресурс] // Блог OrbitSoft – 2023. – Режим доступа: https://orbitsoft.com/ru/blog/crud/

25. Лесников П.П. Обзор фреймворка Next.js [Электронный ресурс] // Блог LiquidHub – 2023. – Режим доступа: https://liquidhub.ru/blogs/blog/next-js

26. Мамедли Р.Э. Системы управления базами данных: учебник для СПО. – Санкт-Петербург: Лань, 2024. – 228 с.

27. Молибог Н. П. Что такое Node.js и для чего он нужен [Электронный ресурс] // Блог nic.ru – 2023. – Режим доступа: https://www.nic.ru/help/chtotakoe-nodejs-i-dlya-chego-on-nuzhen_11316.html

28. Морозов А.С. Как я использую React Hook Form [Электронный ресурс] // Блог Хабр – 2023. – Режим доступа: https://habr.com/ru/articles/746806/

29. Пашаев В.П. Что такое Next.js и для чего он нужен? [Электронный ресурс] // pxstudio – 2021. – Режим доступа: https://pxstudio.pw/blog/chto-takoe-next-js-i-dlya-chego-on-nuzhen

30. Полуэктова Н. Р. Разработка веб-приложений: учебное пособие для вузов / 2-е изд. –Москва: Издательство Юрайт, 2024. –204 с

31. Поляков Д. И. 10 лучших Frameworks и библиотек JavaScript [Электронный ресурс] // Блог zemlyanika.org – 2022. – Режим доступа: https://zemlyanika.org/10-luchshih-frameworks-i-bibliotek-javascript-dlya-izucheniya-v2022-godu/

32. Прияцелюк Н.А. Веб-фреймворки для начинающих: простое объяснение с примерами NestJS [Электронный ресурс] // Tproger – 2023. – Режим доступа: https://tproger.ru/translations/web-frameworks-how-to-get-started

33. Пушнов Г.А. Контроллеры и их истинное предназначение [Электронный ресурс] // Блог devnotes – 2023. – Режим доступа: https://www.dev-notes.ru/articles/laravel/controllers-and-their-true-purpose/

34. Рокатанский М.В. Разбираемся с middleware [Электронный ресурс] // Блог Хабр – 2020. – Режим доступа: https://habr.com/ru/companies/otus/articles/528692/

35. Стоунз Р., Мэтью Н. PostgreSQL. Основы. – Пер. с англ. – СПб.: Символ-Плюс, 2002. – 640 с.

36. Сугак Д.Б. Роль веб-сайта в научно-образовательной деятельности вуза [Электронный ресурс] // Вестник СПбГИК. – 2012. – №3. – Режим доступа: https://cyberleninka.ru/article/n/rol-veb-sayta-v-nauchno-obrazovatelnoy-deyatelnosti-vuza

37. Титова Л.Г. Методика организации и обеспечения учебного процесса: учебно-методическое пособие. – Ярославль: ЯрГУ, 2021. – 36 с.

38. Фролов П.К. Zustand – руководство по простому управлению состоянием [Электронный ресурс] // Блог Хабр – 2022. – Режим доступа: https://habr.com/ru/articles/661411/

39. Чепалов Р.Г. Что должно размещаться на сайтах вузов. [Электронный ресурс] // Academica.ru. – 2019. – Режим доступа: https://academica.ru/stati/stati-opervom-vysshem-obrazovanii-i-magistrature/866236-chto-dolzhno-razmeschatsja-na-sajtah-vuzov/