МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра технологий программирования

РАЗРАБОТКА ИНТЕРНЕТ-СЕРВИСА ДЛЯ ТЕСТИРОВАНИЯ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Митина Дмитрия Александровича

Научный руководитель
зав. каф., доцент, к. ф.-м. н. _____ И. А. Батраева

Заведующий кафедрой
зав. каф., доцент, к. ф.-м. н. _____ И. А. Батраева

Саратов 16 июня 2024 г.

ВВЕДЕНИЕ

С развитием информационных технологий и интернета образование стало доступным в любой точке мира. Сайты, предлагающие образовательные материалы и курсы, стали неотъемлемой частью современной образовательной среды. Однако существующие образовательные платформы часто имеют недостатки и ограничения, которые могут затруднять процесс обучения.

Целью данной дипломной работы является разработка интернет-сервиса для тестирования. Важно отметить, что сайт был разработан специально для преподавателя Саратовского национального исследовательского государственного университета имени Н. Г. Чернышевского. По части разработки функционала был процесс согласования требований. Веб-приложение в настоящее время активно используется как преподавателями, так и студентами. Оно основано на современных технологиях и инструментах веб-разработки, что обеспечивает удобный интерфейс и широкий функционал.

Исходя из поставленной цели необходимо решить следующие задачи:

1. изучить основные принципы разработки веб-приложений;
2. изучить технологии, необходимые для создания веб-приложения;
3. реализовать удобный и понятный пользователям функционал.

Структура и объём работы.

Для решения поставленных задач выполнена выпускная квалификационная работа, включающая в себя введение, 2 основные главы, заключение, список использованных источников из 20 наименований и 2 приложений. Работа изложена на 49 страницах, содержит 33 рисунка.

Первая глава имеет название «Web-разработка» и содержит основную теоретическую информацию о технологиях и подходах, которые использовались при разработке приложения.

Вторая глава имеет название «Реализация программной части», данная глава содержит подробное описание процесса выполнения работы.

Выпускная квалификационная работа заканчивается заключением, списком использованных источников, а также приложением с кодом.

1 Основное содержание работы

1.1 Теоретическая часть

История создания сайтов

Веб-сайт info.cern.ch, созданный Тимом Бернерсом-Ли в 1991 году, стал первым сайтом в мире и включал определение технологического процесса World Wide Web. Этот процесс основывался на HTTP, URI и HTML. Также на сайте были представлены концепции работы серверов и браузеров. Тим Бернерс-Ли также разработал первый интернет-браузер WorldWideWeb, первый сервер и веб-страницы ещё в 1990 году. Веб-сайт также стал первым интернет-каталогом, в который Бернерс-Ли добавил перечень гиперссылок. Его ранние работы на гипертекстовым программным обеспечением Enquire и предложения по использованию гипертекста в CERN также способствовали развитию интернет-технологий.

Многостраничное приложение

Многостраничное приложение (MPA) — это веб-приложение, которое загружает новую страницу для каждого действия, выполняемого пользователем. Многостраничное приложение состоит из нескольких статических страниц, которые загружают новую страницу на сервер, а затем обновляют содержимое этой страницы по мере необходимости. Многостраничное приложение часто используется, когда приложению необходимо иметь разные страницы для разных целей. Например, MPA имеет домашнюю страницу, каталог продукции, контактную форму и профиль пользователя. Каждая страница будет иметь свой собственный макет и функциональность, в зависимости от ее назначения.

Одностраничное приложение

Одностраничное приложение (SPA) относится к типу приложений, которые полностью запускаются в веб-браузере и не требуют перезагрузки всей страницы во время использования. Цель — ускорить переходы, чтобы веб-сайт больше походил на нативное мобильное приложение.

Клиентская часть

Любое веб-приложение состоит из двух частей — клиентской и серверной. Клиентская сторона охватывает всё, что пользователи могут видеть на экране.

Важнейшие элементы технологического стека в клиентской части:

1. язык разметки веб-страниц HTML, который отвечает за отображение содержимого в браузере;

2. язык описания внешнего вида документа CSS, который стилизует контент;
3. язык для доступа к объектам приложений Javascript — отвечает за интерактивную часть веб-приложения.

Основные концепции, используемые в веб-разработке

HTML, CSS и JavaScript — это основные языки, используемые в веб-разработке, каждый из которых играет свою уникальную роль в создании веб-сайтов. HTML задает структуру и содержание страницы, CSS отвечает за стиль и внешний вид, а JavaScript добавляет интерактивные элементы, делая сайты более динамичными и функциональными. Вместе эти технологии обеспечивают разработку современных, адаптивных и взаимодействующих веб-приложений, которые удовлетворяют потребности пользователей и технические требования сегодняшнего дня.

HTML: структура страницы

Основная информация текста заключается в том, что HTML (HyperText Markup Language) является основным языком разметки для создания веб-страниц. HTML использует систему тегов, которые определяют структуру и функции элементов на странице, таких как текст, картинки, кнопки. Ключевые теги включают <html>, <head>, <body>, , <h1>, <p>, и <button>. Также HTML включает теги для навигации (<nav>), структурирования контента (<article>, <section>), вставки изображений (<img>), создания гиперссылок (<a>), таблиц (<table>), списков (<ul>, <ol>, <li>), а также мультимедийного контента (<video>, <audio>, <iframe>). Эти элементы обеспечивают не только визуальное и функциональное обогащение веб-страниц, но и их семантическую структуру и доступность.

CSS: стилизация элементов страницы

Cascading Style Sheets (CSS) — это язык стилей, используемый для оформления веб-страниц. Он позволяет стилизовать текст, изменять цвета, размеры, расположение элементов и добавлять анимации и эффекты перехода. CSS поддерживает три основных типа селекторов: по тегу, классу и идентификатору, которые могут быть использованы для применения стилей к различным HTML элементам.

Стили можно описывать непосредственно в HTML через атрибут style (inline стили) или внутри тега <style>. Однако наиболее эффективный способ управления стилями — это использование внешних CSS файлов, которые подключаются в HTML. Это улучшает поддержку и управляемость сайта, позволяя

изменять внешний вид всех страниц, подключающих эти стили, централизованно.

CSS также включает продвинутые техники, такие как медиазапросы, которые адаптируют внешний вид страницы к различным условиям отображения. Это основа создания отзывчивых дизайнов, которые обеспечивают корректное отображение контента на разных устройствах. Каскадность и наследование в CSS позволяют создать визуальную иерархию и единообразный дизайн, минимизируя при этом количество кода.

JavaScript: добавление интерактивности

JavaScript играет ключевую роль в создании интерактивных и динамических веб-приложений, исполняясь непосредственно в браузере и взаимодействуя с элементами страницы через подключаемые скрипты. Этот язык программирования является основой для многих современных библиотек и фреймворков, таких как React, Angular, и Vue.js, которые облегчают разработку сложных интерфейсов и одностраничных приложений.

JavaScript не ограничивается только веб-разработкой; он также применяется в серверных и мобильных приложениях, а также в программировании игр. Язык поддерживает различные парадигмы программирования и предлагает механизмы для асинхронного программирования, такие как промисы, что делает его незаменимым инструментом для создания быстрых и отзывчивых приложений. Постоянное развитие стандартов ECMAScript гарантирует, что JavaScript остается в авангарде технологических инноваций, обеспечивая разработчикам передовые инструменты для решения актуальных задач.

Фреймворки и библиотеки для веб-разработки

Фреймворк облегчает разработку приложений и сайтов, предоставляя готовые библиотеки и код. Он автоматически управляет базами данных, файловой системой и другими ключевыми элементами, что позволяет разработчику сосредоточиться на создании структуры приложения, добавлении нужных компонентов и их интеграции.

Хотя фреймворки и библиотеки часто путают из-за их готовых решений, ключевое отличие в том, что фреймворки задают архитектуру приложения, а библиотеки — нет. Другое важное различие заключается в том, как они используются: в случае с библиотекой вызываются её функции, в то время как фреймворк требует написания функций в специфическом стиле и сам вызывает

код. написания функций в определённом стиле.

Фреймворк Vue

Vue.js — это JavaScript-фреймворк, разработанный для создания интерактивных веб-интерфейсов. Он обладает простой моделью данных и предоставляет удобные инструменты для построения динамичных приложений. Основная логика работы Vue заключается в реактивности данных, что позволяет автоматически обновлять DOM при изменениях данных. Это значительно упрощает разработку интерактивных интерфейсов, так как разработчик может сосредоточиться на логике данных, а не на манипуляциях с DOM.

Фреймворк Angular

Angular — это мощный фреймворк для разработки веб-приложений, созданный и поддерживаемый компанией Google. Он предоставляет разработчикам полный набор инструментов для построения как простых, так и сложных веб-приложений. Особенностями Angular являются его модульная структура, поддержка двустороннего связывания данных, а также использование компонентов и сервисов, которые помогают организовать код и повысить его переиспользуемость.

Angular применяет шаблон MVC (Model-View-Controller), что облегчает разделение данных (модели), пользовательского интерфейса (вид) и логики управления (контроллер). Это делает приложения, написанные на Angular, легко масштабируемыми и поддерживаемыми. Фреймворк также активно использует TypeScript — строго типизированный язык программирования, который добавляет множество преимуществ для больших и сложных проектов, таких как автодополнение кода и повышенная читаемость.

Библиотека React

React — это библиотека JavaScript, разработанная Facebook для создания пользовательских интерфейсов. Основная идея React заключается в том, что интерфейс пользователя разбивается на независимые компоненты, каждый из которых управляет своим собственным состоянием и рендерит себя в зависимости от данных, с которыми он работает. Это позволяет разработчикам строить сложные пользовательские интерфейсы из простых частей, делая код более модульным и легким для поддержки.

React использует JSX — синтаксическое расширение для JavaScript, которое позволяет писать структуру компонента в виде, напоминающем HTML.

Это делает код более читаемым и удобным для работы. React также оптимизирует процесс рендеринга, используя виртуальный DOM, который позволяет эффективно обновлять только те части интерфейса, которые действительно изменились.

Архитектура REST

REST — это акроним, сокращение от английского Representational State Transfer — передача состояния представления.

Это архитектурный стиль взаимодействия компонентов распределенной системы в компьютерной сети. Проще говоря, REST определяет стиль взаимодействия (обмена данными) между разными компонентами системы, каждая из которых может физически располагаться в разных местах. В общем случае это происходит посредством запросов-ответов. Компоненту, которая отправляет запрос называют клиентом; компоненту, которая обрабатывает запрос и отправляет клиенту ответ, называют сервером. Запросы и ответы, чаще всего, отправляются по протоколу HTTP (HyperText Transfer Protocol). Как правило сервер — это некое веб-приложение. Клиентом же может быть мобильное приложение, которое запрашивает у сервера данные, либо браузер, который отправляет запросы с веб-страницы на сервер для загрузки данных. Приложение А может запрашивать данные у приложения Б. Тогда А является клиентом по отношению к Б, а Б — сервером по отношению к А. Одновременно с этим, А может обрабатывать запросы от В, Г, Д и т.д. В таком случае, приложение А является одновременно и сервером, и клиентом. Все зависит от контекста. Однозначно одно: компонента которая шлет запрос — это клиент. Компонента, которая принимает, обрабатывает и отвечает на запрос — сервер.

Платформа Node.js

Node.js представляет собой платформу, которая расширяет возможности JavaScript, позволяя использовать его не только на клиентской, но и на серверной стороне. Это существенно упрощает разработку, так как разработчики могут использовать один и тот же язык программирования для всего стека технологий, способствуя совместимости и переиспользованию кода.

Язык программирования Python

Python — это интерпретируемый язык программирования, который известен своей кросс-платформенностью, что позволяет выполнять программы на разных операционных системах без изменений. Он используется в веб-

разработке, научных вычислениях, анализе данных, машинном обучении и автоматизации задач. Python поддерживает различные парадигмы программирования, включая объектно-ориентированный, процедурный и функциональный стили, а также имеет большое количество библиотек и фреймворков, что упрощает процесс написания кода

Язык программирования Java

Java — это объектно-ориентированный язык программирования, который известен своей надежностью и переносимостью, что позволяет его использовать на различных платформах и устройствах. Java используется во многих отраслях, включая разработку мобильных приложений, веб-приложений. Благодаря своим возможностям для распределенных систем и сетевого программирования, Джава также широко применяется в области корпоративной разработки.

Язык программирования C#

C# — это объектно-ориентированный язык программирования, который используется преимущественно для создания приложений под платформу .NET. Он предлагает широкий набор инструментов и библиотек для создания масштабируемых, надежных и безопасных приложений. Благодаря своей интеграции с платформой .NET, C# предоставляет доступ к различным возможностям, таким как работа с базами данных, сетевое программирование и многопоточность.

Язык программирования Golang

Go — это компилируемый многопоточный язык, который применяется в таких областях, как сетевое программирование, веб-разработка и создание микросервисов, а также позволяет разрабатывать эффективное и масштабируемое программное обеспечение. Golang отличается простым синтаксисом и быстрой компиляцией, что облегчает разработку и повышает производительность приложений. Еще он предоставляет встроенные средства для работы с параллелизмом и горутинами, что делает его особенно полезным при разработке конкурентных приложений.

1.2 Практическая часть

Требования к функционалу

1. Возможность создания тестов с произвольным количеством вопросов и вариантов ответа;
2. Два режима отображения вопросов: вопросы отображаются на доске в классе или на экране студента;

3. Ручное закрытие доступа к тесту преподавателем;
4. Автоматическая проверка ответов по заранее заданному ключу;
5. Вывод статистики по результатам тестирования для преподавателя;
6. Экспорт результатов тестирования в формате Excel для дальнейшего анализа;
7. Возможность создания папок для группировки тестов;
8. Возможность прохождения тестов студентами без регистрации.

Реализация серверной части интернет-сервиса для тестирования

Файл index выполняет роль корневого элемента сервера, отвечающего за основные настройки и подключения. В этом файле происходит подключение необходимых модулей, таких как Express для работы с сервером, CORS для обработки запросов с других доменов, cookie-parser для работы с куки и mongoose для взаимодействия с базой данных. Здесь же настраиваются middleware для обработки JSON и куки, управление CORS-запросами, подключение маршрутов приложения и обработка ошибок. Также определяется маршрут для получения изображений по их ID, что свидетельствует о готовности сервера к обработке входящих запросов после успешного подключения к MongoDB.

Управление роутерами и контроллерами

В системе предусмотрены разные роутеры для управления тестами, пользователями и файлами CSV. Роутеры, такие как testRouter, userRouter, и csvRoutes, описаны в отдельных файлах и настроены на обработку специфических запросов. Например, csvRoutes управляет загрузкой и созданием CSV-файлов, testRouter занимается операциями с тестами, а userRouter обрабатывает запросы пользователей. Эти файлы используют Express для маршрутизации и включают middleware для аутентификации и других функций, что обеспечивает централизованное и эффективное управление доступом и функциональностью.

Аутентификация и управление пользователями

Контроллеры из файла userController включают методы для аутентификации пользователя, обновления пароля, управления сеансами пользователя и обновления токенов. Методы, такие как login, updateUserPassword, logout, и refresh, поддерживают безопасность и управление пользователями в системе. Они позволяют пользователям входить в систему, обновлять свои данные, безопасно выходить и поддерживать свою сессию активной через механизмы токенов.

Управление тестами и их результатами

Файл TestController содержит множество методов для управления тестами. Это включает создание тестов, обновление, удаление, и получение данных о тестах. Контроллер также обладает функциональностью для создания папок для группировки тестов, открытия и закрытия доступа к тестам, а также экспорта результатов. Методы, такие как create, updateFolder, openAll, closeAll, и downloadTest, позволяют преподавателям эффективно управлять процессом тестирования и анализа результатов, что обеспечивает гибкость и контроль над образовательным контентом.

Реализация клиентской части интернет-сервиса для тестирования

Клиентская часть сайта была разработана с использованием React, CSS, стейт-менеджера Zustand, а также библиотек Axios и Ant Design. Это обеспечивает гибкость и масштабируемость интерфейса, поддерживает современные стандарты пользовательского взаимодействия и позволяет эффективно обрабатывать сетевые запросы и состояния компонентов.

Реализация навигации и маршрутизации

В клиентской части сайта внедрена система маршрутизации с помощью React Router, которая включает приватные и публичные маршруты. Это позволяет разграничивать доступ к функционалу сайта в зависимости от статуса пользователя (студент или преподаватель). Приватные маршруты доступны только после авторизации и включают управление тестами и просмотр детализированных данных, в то время как публичные маршруты позволяют студентам просматривать и проходить тесты.

Страницы для студентов

Для студентов были разработаны две основные страницы: страница со списком всех доступных тестов и страница прохождения теста. На первой студенты могут видеть все тесты, доступные для прохождения, что позволяет им легко выбирать и начинать тестирование. Вторая страница предназначена для непосредственного прохождения теста, где студенты отвечают на вопросы и подают свои ответы в систему.

Страницы для администратора системы (преподавателя)

Для администратора системы предусмотрены страницы для авторизации, управления тестами, просмотра результатов, изменения ключа теста и поиска результатов конкретных студентов. Эти инструменты позволяют преподавате-

лям эффективно управлять контентом тестов, отслеживать успеваемость студентов, а также обновлять и корректировать ключи ответов для повышения точности оценок.

ЗАКЛЮЧЕНИЕ

В ходе дипломной работы было реализовано web-приложение с использованием таких технологий, как библиотека React, кроссплатформенной среды для разработки клиентских приложений Node.js, стейт-менеджера Zustand, MongoDB. Разработка с использованием библиотеки React позволила достичь высокой оптимизации приложения и удобного интерфейса с простым управлением.

Умения, полученные в ходе выполнения практической части, являются важными навыками для fullstack-разработчиков, так как они позволяют создавать современные и отзывчивые приложения с использованием актуальных технологий.

Преподаватель остался доволен результатом, так как все его требования были учтены в проекте. Несмотря на успешное завершение основного этапа разработки, проект пока не может считаться полностью завершенным. Предстоит расширение функционала для обеспечения поддержки большего числа пользователей и внедрения новых возможностей, что позволит приложению масштабироваться и адаптироваться к расширяющейся аудитории.

Все поставленные задачи дипломной работы решены, цель достигнута.