Слайд 1.
Добрый день, коллеги!

Спасибо, что предоставили слово. С темой моей работы Вы можете ознакомиться на слайде, продублирую: «Разработка клиентской части веб-системы для театральной индустрии»

Перейдем к содержанию

Слайд 2.

В презентации рассмотрим содержание: *на слайде*

Итак, перейдем к Актуальности моей работы

Слайд 3.

Веб-приложение для театральной индустрии помогает сориентироваться в большом разнообразии предложений, предлагаемых театральной индустрией.

Для ознакомления с актуальными произведениями любителю театров не нужно тратить деньги и время, пользователь заходит на веб-сервис и выбирает произведение, которое больше всего вызвало интерес.

Также пользователь может ознакомится с актерским составом и иной детальной информацией.

Театральные организации тратят достаточно большое количество денежных средств на бумажную рекламу (баннеры), демонстрацию, обновление рекламы и тд.

Веб-платформа позволит сократить издержки театров на маркетинговое развитие.

Таким образом, *текст на слайде 3*

Итак, перейдем к цели и задачам моей работы

Слайд 4.
Читаю цель со слайда

прочитал первый и второй пункт

Для грамотной работы веб-сервиса, под грамотностью мы понимаем, работает без ошибок и сбоев, а также соответствует установленным требованиям по производительности, безопасности, кроссплатформерности, эффективное программирование, реализация функциональности без ошибок и тд. Для этого нам надо провести некоторый анализ, обзор основных принципов архитектурных разработок, выбрать правильный подход к проектированию и правильные технологии.

Хочу отметить, что это очень важная задача. От ее решение зависел успех в разработке всего приложения в целом

И последняя задача связана уже непосредственно с
прочитал третий пункт

Таким образом, перейдем к следующему слайду АРХИТЕКТУРНОЕ РЕШЕНИЕ ВЕБ-ПРИЛОЖЕНИЯ

Слайд 5.

Вначале надо было определиться с выбором подходов проектирования: микросервисный или монолитный подход.
На основании анализа преимуществ и недостатков того или иного подхода было принято решение реализовать на основе монолитного подхода. Это классический паттерн проектирования ПО.

На рисунке 1 изображен подход к созданию веб-приложений на основе тонкого клиента и шаблона MVC (Model View Controller) с использованием программного каркаса. Основная логика приложения находится на серверной стороне, а клиентская часть получает готовые данные для отображения и обработки.

При проектировании используют два основных подхода: многостраничные приложения или Single Page Applications (SPA).

Данный проект будет реализован с помощью подхода Single Page Applications, так как удобен совместно с использованием монолитного подхода из-за возможности работать над клиентской стороной независимо от процессов на стороне сервера.

Многие из известных фреймворков были разработаны, следуя концепции шаблона MVC.

Таким образом, выбранные технологии: Программный каркас на основе шаблона тонкого клиента и MVC (Model View Controller), одностраничное приложение (SPA) и фреймворк.

Итак, перейдем к следующему слайду ВЫБОР ФРЕЙМВОРКА

Слайд 6.

Согласно данным из официального источников наибольшей популярностью у разработчиков пользуются javaScript-фреймворки: react.js,

Angular, Vue.js.

Рассмотрим данные фреймворки в обзоре более подробно, обозначили преимущества и недостатки, и выберал наиболее подходящий для решения задач проекта.

Таким образом, лучшим выбором в данном проекте является фреймворк react.js. Так как с помощью него мы сможем реализовать поставленные задачи, такими преимуществами как:

1. Высокая производительность: React использует виртуальный DOM и обновляет только измененные элементы, что повышает производительность приложения;
2. Модульность: React позволяет создавать многоразовые компоненты, которые можно легко переносить между проектами и использовать в разных частях приложения;
3. Легкость поддержки: React имеет большое сообщество разработчиков и активную экосистему инструментов, что делает его удобным для поддержки и дальнейшего развития;
4. Гибкость: React можно использовать для построения разнообразных типов интерфейсов - от простых форм до сложных и плавных анимаций;
5. Удобство тестирования: React имеет качественный набор инструментов для тестирования, что позволяет легко написать модульные и интеграционные тесты для веб-приложения;
6. Команда разработки клиентского части состоит из JavaScript-разработчика;
7. Прогнозируется, что взаимодействовать с платформой для театральной индустрии будет значительное количество пользователей.

Таким образом, react практически идеально подходит для решения поставленных задач.

Итак, перейдем к результатам моей работы.

Слайд 7.

Рассмотрим работу приложения на примере выбора одного из произведения. После сканирования qr кода попадаем на главную, выбираем категорию произведения, выбираем произведение и знакомимся с информацией.

Взаимодействие пользователя с другими категориями аналогично.

И конечно же в основе работы лежат, те принципы, которые мы закладывали после совершения анализа.

т. е. React строит (отрисовывает) Dom-дерево, данные получат из базы данных

от backend..

По ситуации
Когда файл со скриптовым языком программирования загружается на страницу, скрипт взаимодействует с объектной моделью документа (DOM). И внутри объектного дерева React отслеживает действия пользователя и менять дерево элементов на любом уровне, когда это нужно. Для такой работы используется быстрый виртуальный DOM.

Логика работы большинства вышеперечисленных страниц:
1) Пользователь открывает страницу;
2) Браузер отправляет запрос на сервер на получение HTML-документа по протоколу передачи данных HTTP;
3) Сервер отправляет HTML-документ и связанные ресурсы [20];
4) React.js начинает взаимодействие с UI с зафиксированного HTML-файла. Пользовательский интерфейс определяется элементом, который будет возвращать JSX. JSX - это расширение синтаксиса JavaScript;
5) В случае, если пользователь нажимает на кнопку, например, “загрузить еще” на главной странице, React запрашивает данные для отображения HTTP-запросом через REST API с backend с указанием конкретного типа запрашиваемого контента;
6) Backend возвращает HTTP-ответ с контентом, который обрабатывает React согласно написанному разработчиком коду.

Таким образом, успешно была спроектирована структура веб-приложения, разработаны адаптивный интерфейс согласно макетам дизайна, реализованы механизмы взаимодействия фреймворка React как с серверной частью, так и с пользователем веб-системы.

Слайд 8.
Ознакомиться с сайтом можете, сканировав qr-код экране

Итак, перейдем к следующему слайду ЗАКЛЮЧЕНИЕ

Слайд 9.

В ходе работы был сделан сравнительный анализ подходов к проектированию веб-приложения и JavaScript-фреймворков для разработки клиентской части веб-приложения с полным разбором преимуществ и недостатков, исходящих из целей поставленной нами задачи. Был выбран и аргументирован определенный стек технологий, с помощью которого было реализовано веб-приложение, а именно программный каркас на основе шаблона тонкого клиента и MVC (Model View Controller), принцип одностраничного приложения (SPA) и клиентский программный каркас на основе JavaScript-

фреймворка React.js.

Таким образом все поставленные задачи и цель были досигнуты в полной объеме

Также хочется отметить
Читаю последнюю фразу на слайде

Перещелкиваю слайд

Слайд 10.
Также хочу выразить благодарность своего научному руководителю, коммиисси

Спасибо за внимание!