МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ ДЛЯ ERP СИСТЕМЫ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Климова Алексея Дмитриевича

Научный руководитель
доцент, к. ф.-м. н. _____ В. М. Соловьев

Заведующий кафедрой
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В силу постоянно растущих требований к пользовательским интерфейсам, технологии для их создания развиваются с высокой скоростью. Ежегодно появляются инструменты, предоставляющие новые концепции и возможности для разработки клиентских приложений. При этом, зачастую, многие проекты, направленные на решение проблем бизнеса, не обновляют инструментальную базу с момента начала разработки. Переход таких систем на новые технологии является трудоемкой задачей.

Целью настоящей работы является создание клиентского приложения, которое будет являться основой для перехода ERP системы компании Exactpro на актуальные технологии построения пользовательских интерфейсов. В связи с поставленной целью, в данной работе решаются следующие задачи:

- анализ и выбор инструментальной базы;
- создание клиентского приложения с реализацией части интерфейса основного проекта;
- частичное интегрирование нового интерфейса в основную систему;
- интеграция нового приложения в процесс разработки и непрерывной интеграции основной системы;
- сравнение производительности аналогичного функционала в новой и старой системе.

1 Теоретическая часть

1.1 Клиент-серверная архитекутра

Данная работа строится на понятии клиент-серверной архитектуры. Это модель разработки, в которой веб-приложение разделяется на две составляющие - клиент и сервер. Сервер является вычислительном ядром, управляет большинством ресурсов, служб и данных. Каждый пользователь имеет собственный клиент, который в свою очередь занимается только отображением данных, получаемых от сервера и отправкой новых данных на этот же сервер. [1]

Самой популярной реализацией этой модели является REST API, в которой все данные передаются с помощью HTTP протокола и каждая сущность данных представляет собственный ресурс.

1.2 ERP система

ERP система компании Exactpro является основным внутренним проектом организации, который позволяет управлять различными процессами ее жизненного цикла. Проект был создан немного позднее основания компании, и с помощью использующихся в нем технологий достаточно тяжело реализовать интерфейсы, соответствующие текущим стандартам дизайна, а сама такая реализация может иметь низкую производительность. В связи с этим, было принято решения начать процесс миграции проекта на актуальные технологии разработки клиентской части.

1.3 Java EE

Основой ERP системы является Java EE. Это платформа разработки, включающая в себя полный спектр инструментов, необходимых для реализации веб-приложений. Эта архитектура помогает сосредоточиться на проблемах презентации и приложений, а не на системных проблемах. Java EE предоставляет упрощенную модель программирования, включающую следующие инструменты:

- встроенная конфигурация с аннотациями, что делает необязательным подход с использованием дескрипторов развертывания;
- внедрение зависимостей, скрытие создания ресурсов и поиска кода;
- управление данными с помощью JPA без явного использования SQL.

В данной работе рассматриваются компоненты Java EE, упрощающие взаимодействие с протоколом HTTP. Благодаря Java Security конфигурируются

роли пользователей и их разрешения для получения различных страниц и ресурсов. С помощью JAX-RS создается REST API и задаются обработчики HTTP методов. Благодаря технологии сервлетных фильтров, запросы к REST API проходят аутентификацию и конфигурируется работа с CORS. [2]

1.4 React

В качестве основной технологии для нового приложения была выбрана JavaScript библиотека React,т.к. она является самой часто используемой среди своих аналогов и имеет хорошую документацию и базу готовых решений [3]. На рисунке 1 показан график количества использований React и его аналогов. [4]

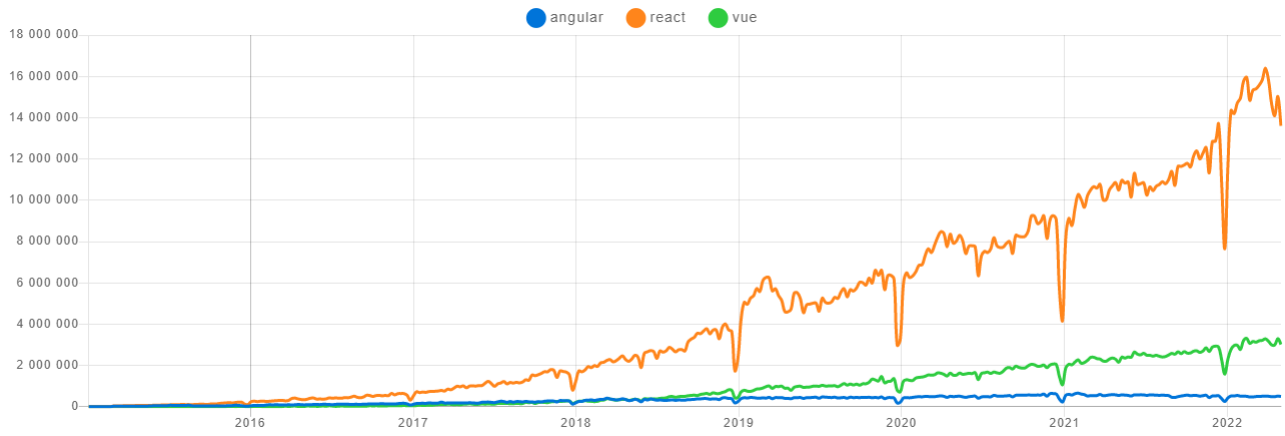

Рисунок 1 – «График количества использований React и его аналогов»

1.5 TypeScript

TypeScript это язык программирования, расширяющий возможности языка JavaScript. Он отличается от JavaScript возможностью явного статического назначения типов. Это позволяет избежать распространенных проблем при разработке на JS, связанных со сложностью отладки. Статическая типизация позволяет найти уязвимости и проблемы еще до запуска программы. TypeScript компилируется в JavaScript и обладает с ним обратной совместимостью. Для того чтобы запустить JavaScript код в TypeScript, нужно создать специальный файл, с определениями типов используемого кода. TypeScript используется при разработке React приложений на равне с JavaScript, в частности для явного описания типов параметров и состояний компонента, во избежание ошибок. [5]

1.6 Styled Components

Styled Components - это удобный инструмент для стилизации React компонентов, широко использующийся при разработке интерфейсов. Он позволяет

создавать из JSX HTML элементов полноценные компоненты со встроенными стилями. Стилизация таких компонентов наследует идеи препроцессора языка CSS SASS, добавляя возможность удобно описывать в одном компоненте стилизацию в зависимости от передаваемых параметров. Styled components позволяет настраивать сложные иерархии параметров, использующиеся, например, при разработке интерфейсных тем. Также большим преимуществом этого инструмента является то, что генерируемые имена классов являются уникальными, что позволяет не заботится об их наименовании.

1.7 Webpack

Webpack это инструмент, позволяющий конфигурировать и осуществлять сборку современных JavaScript приложений. Он строит граф зависимостей точек входа в систему и объединяет каждый модуль в один или несколько пакетов, являющихся статическими ресурсами. [6]

1.8 NodeJs

NodeJs представляет собой кросплатформенную программную среду для исполнения JavaScript программ. Он использует функциональное ядро веб браузера Google Chrome под названием V8. Среда NodeJs является событийно-ориентированной и поэтому основной поток исполнения практически никогда не блокируется, а затратные операции выполняются асинхронно. Благодаря этому, NodeJs используется для серверной разработки, позволяя создавать высокопроизводительные и легко масштабируемые серверные приложения с помощью JavaScript.

1.9 CI/CD

Прежде чем приступить к рассмотрению конкретных инструментов, стоит дать определение понятию CI/CD. CI/CD это методология, сочетающая в себе методы непрерывной интеграции и непрерывной доставки. CI/CD автоматизирует большую часть или все ручное вмешательство человека, традиционно необходимое для запуска нового кода из фиксации в производство, например сборку, тестирование и развертывание, а также подготовку инфраструктуры. С помощью конвейера CI/CD разработчики могут вносить изменения в код, которые затем автоматически тестируются и передаются для доставки и развертывания. [7]

1.10 GitLab CI/CD

Gitlab CI/CD это инструмент для настройки процессов CI/CD. Основными понятием Gitlab CI/CD является Pipeline. Это CI/CD конвейер, состоящий из этапов и Job. Job описывают конкретные скрипты исполнения. Этапы описывают последовательность выполнения различных Job. В зависимости от конфигурации, Job могут выполняться последовательно или параллельно, а также иметь различные способы запуска, такие, как например, ручной запуск или запуск в зависимости от результатов работы прошлого этапа. Вся конфигурация описанных функций описывается в специальном yaml файле. [8]

2 Практическая часть

Миграция является достаточно трудоемкой задачей, т.к. ERP система имеет обширный функционал, включающий в себя порядка ста страниц. Помимо этого система представляет собой монолитную архитектуру, одновременно являясь и клиентом и сервером. Поэтому переход необходимо выполнять в несколько этапов, постепенно внедряя новый функционал в текущую систему. В данной работе рассматривается первый этап, в котором ставятся следующие задачи:

— построение архитектуры отдельного клиентского приложения;
— перенос функционала страницы Employees в новый клиент;
— реализация новых элементов интерфейса для навигации и перенос их в основной проект;
— обновление конфигурации непрерывной интеграции и администрирования для поддержки новой архитектуры.

2.1 Клиентское приложение

Клиент представляет собой одностраничное React приложение, с маршрутизацией для страницы функционала и информационных страниц, таких, как например, страница ошибки 404. В зависимости от типа сборки, приложение взаимодействует с API или использует встроенные тестовые данные. Статичными ресурсами приложения являются шрифты и svg изображения. [9]

2.2 Функциональное ядро ERP системы

Основная сложность интеграции нового интерфейса заключается в том, что ERP система использует Java фреймворк JSF, реализующий шаблон MVC. Поэтому для синхронизации основного проекта и нового интерфейса, который использует REST, нужно создание нового слоя функционала. Этот слой должен обрабатывать запросы и синхронизировать данные основной системы и нового приложения. [10]

2.2.1 Сборка для работы внутри ERP системы

Для работы внутри основного проекта была написана конфигурация для сборки клиента. В ней было отключено разделение кода и стилей на отдельные части, чтобы интерфейс представлял собой один JavaScript файл. При сборке ERP системы, все статичные ресурсы нового UI переносятся в соответствующие

библиотеки, поэтому в конфигурации их локальные пути заменены на URL соответствующей библиотеки с учетом механизма кеширования. [6]

2.3 CI/CD

Новое приложение пока не является основным, поэтому для разработки и тестирования какого-то функционала не всегда нужно осуществлять его сборку и развертывание. Поэтому в GitLab Pipeline передается параметр, в зависимости от которого обрабатывается только ERP система со встроенными частями нового интерфейса или же дополнительно также обрабатывается само новое приложение. Pipeline имеет 3 этапа: Сборка нового приложения, сборка основного проекта, развертывание. [11].

Для проверки производительности, были проведены тест производительности с использованием Lighthouse [12].

На рисунке 2 изображены метрики производительности страницы Employees в старом проекте.

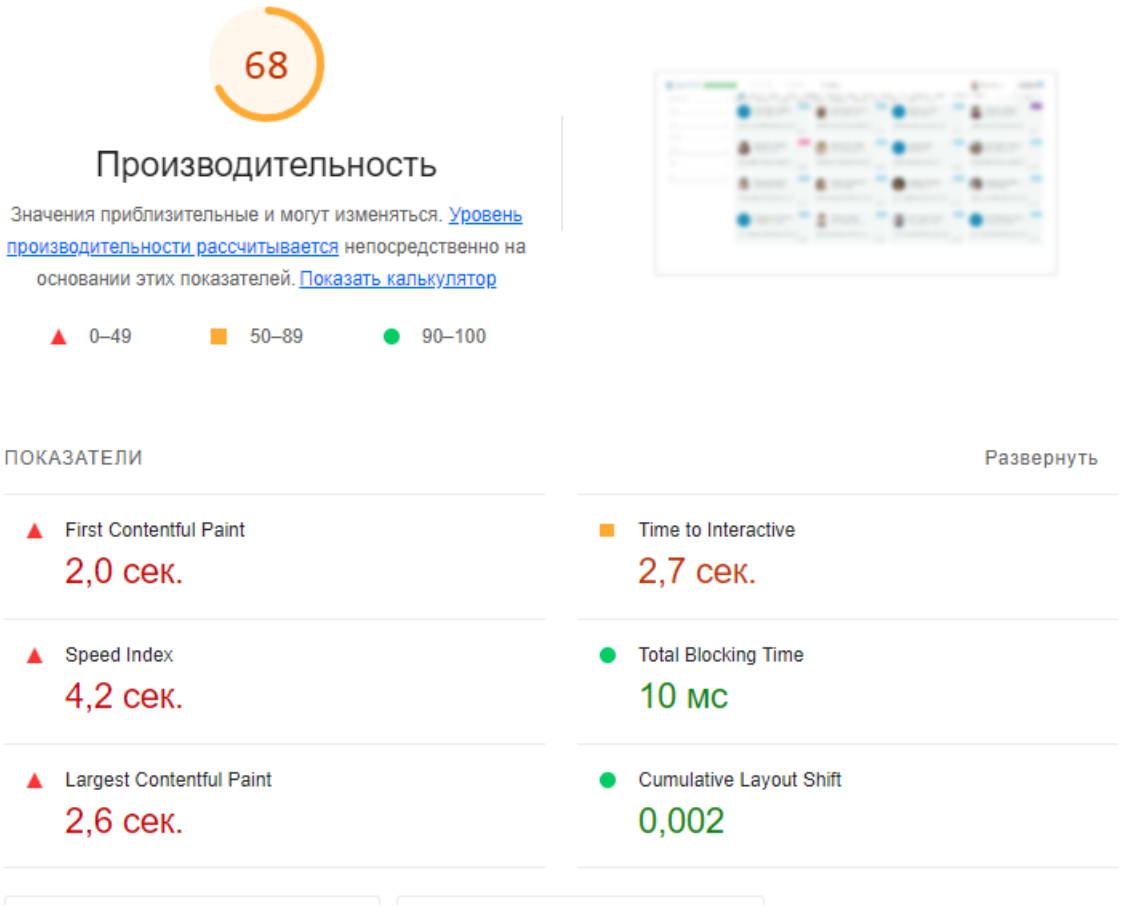

Рисунок 2 – «Показатели производительности старого UI»

На рисунке 3 изображены метрики производительности страницы Employees в новом проекте.

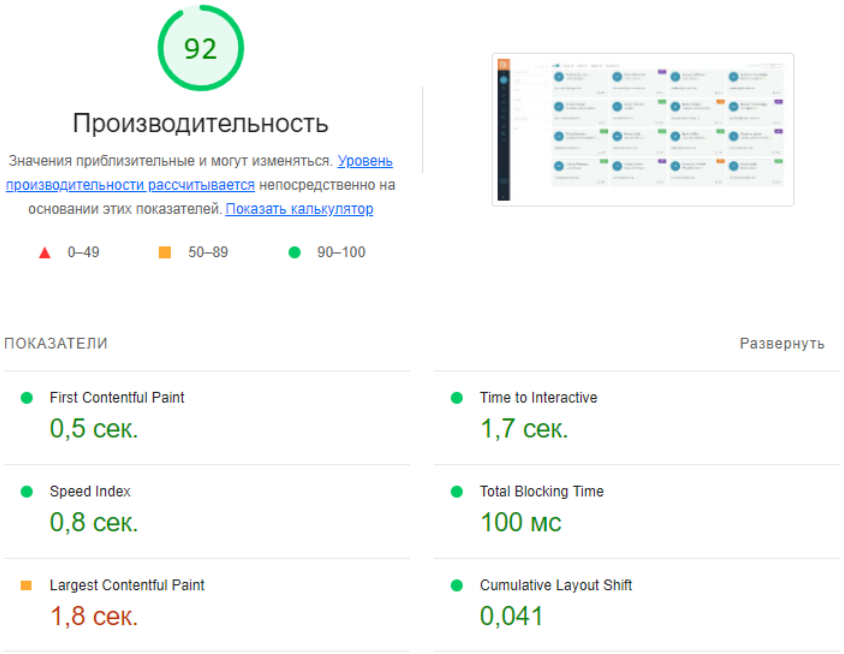

Рисунок 3 – «Показатели производительности нового UI»

Из результатов теста видно, что новое приложение имеет более высокую производительность.

ЗАКЛЮЧЕНИЕ

В ходе данной работы были изучены технологии для построения веб приложений, реализовано клиентское приложение с частью функционала ERP системы. Также новый проект был интегрирован в архитектуру основного проекта и в процессы его непрерывной интеграции и разработки.

Данное приложение будет использоваться в качестве основы, для переноса интерфейсов ERP системы компании Exactpro и позволит создавать более производительный функционал в меньшие сроки.

Таким образом, цель и задачи работы были достигнуты.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Vora, P.* Web Application Design Patterns / P. Vora. — Morgan Kaufmann, 2009.

2 *Blokdyk, G.* Java Platform Enterprise Edition Java EE Complete Self-Assessment Guide / G. Blokdyk. — 5STARCooks, 2018.

3 *Accomazzo, A.* Fullstack React: The Complete Guide to ReactJS and Friends / A. Accomazzo. — Fullstack.io, 2017.

4 Npm trends [Электронный ресурс]. — URL: https://www.npmtrends.com/@angular/core-vs-react-vs-vue (Дата обращения 20.03.2022) Загл. с экр. Яз. англ.

5 Typescript documentation [Электронный ресурс]. — URL: https://www.typescriptlang.org/docs/ (Дата обращения 25.03.2020) Загл. с экр. Яз. англ.

6 Webpack concepts [Электронный ресурс]. — URL: https://webpack.js.org/concepts/ (Дата обращения 05.04.2020) Загл. с экр. Яз. англ.

7 *Kim, G.* Руководство по DevOps / G. Kim. — Манн, Иванов и Фербер, 2018.

8 *O'Grady, A.* GitLab Quick Start Guide: Migrate to GitLab for all your repository management solutions / A. O'Grady. — Packt Publishing, 2018.

9 *Choi, D.* Full-Stack React, TypeScript, and Node. Build cloud-ready web applications using React 17 with Hooks and GraphQL / D. Choi. — Вильямс, 2020.

10 *Burke, B.* RESTful Java with JAX-RS / B. Burke. — O'Reilly Media, 2014.

11 Gitlab ci/cd documentation [Электронный ресурс]. — URL: https://docs.gitlab.com/ee/ci/ (Дата обращения 15.04.2020) Загл. с экр. Яз. англ.

12 A technical seo guide to lighthouse performance metrics [Электронный ресурс]. — URL: https://www.searchenginejournal.com/core-web-vitals/technical-seo-lighthouse/ (Дата обращения 20.04.2022) Загл. с экр. Яз. англ.