Санкт-Петербургский государственный университет

Кафедра Системного программирования

Группа 21.Б10-мм

Разработка системы управления тестированием TestY

Каргин Глеб Павлович

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
доцент кафедры системного программирования, к.т.н. Ю. В. Литвинов

Санкт-Петербург
2023

Оглавление

Введение

Тестирование является неотъемлемой частью цикла разработки программного обеспечения, однако данный процесс устроен крайне сложно и требует отдельного внимания. Во время работы тестировщики используют понятия, такие как тест-кейсы, сьюты, тест-планы и прогоны. Прогоны содержат результаты исполнения тест-планов или отдельных тестов, тест-планы состоят из различных тест-кейсов из сьют, сьюты группируют тест-кейсы по признакам, а тест-кейсы описывают сценарии тестирования одной конкретной функциональности. С целью облегчения работы при тестировании используются системы управления тестированием, которые позволяют структурировать и систематизировать данный процесс. Таким образом, система управления тестированием является незаменимым инструментом особенно для крупных компаний, занимающихся разработкой программного обеспечения.

В условиях реальной разработки крупные проекты насчитывают тысячи различных тест-кейсов, сотни конфигураций. При создании тест-плана выбираются тест-кейсы из сьют, параметры запуска из категорий, таких как операционная система, браузер, разрешение экрана и так далее. Затем выполняется перемножение выбранных параметров, и в конечном итоге создаются тест-планы с наборами тестов, унаследованных от одинаковых тест-кейсов, но имеющих различные параметры запуска. При дальнейшей работе всё вышесказанное вызывает трудности при поиске результата запуска теста или тест-плана с определенными параметрами.

На данный момент существует несколько систем управления тестированием, таких как TestRail[1], Test IT[2], Kiwi TCMS[3]. Однако они содержат ряд недостатков и решений, которые не удовлетворяют требованиям компании YADRO[4]. В связи с этим возникает необходимость в системе управления тестированием, которая бы в первую очередь удо-

[1] https://www.gurock.com/testrail/ — [Дата обращения: 2022-12-13]
[2] https://testit.software/ — [Дата обращения: 2022-12-13]
[3] https://kiwitcms.org/ — [Дата обращения: 2022-12-13]
[4] YADRO — российская ИТ-компания. https://yadro.com — [Дата обращения: 2022-12-13]

влетворяла требованиям и использовалась внутри компании, но при этом был также доступен вариант с открытым исходным кодом с другим дизайном. Поэтому было принято решение создать новую систему управления тестированием.

Данная система управления тестированием крайне важна для YADRO и в первую очередь предназначена для использования в целях компании. В связи с этим основная разработка будет принадлежать YADRO, в том числе корпоративного пользовательского интерфейса, написанного на внутренней библиотеке компании. Поскольку проект также предполагает вариант с открытым исходным кодом, то было принято решение делегировать реализацию публичного пользовательского интерфейса, а также ряд других подзадач команде из студентов СПбГУ.

Таким образом, необходимо спроектировать, реализовать пользовательский интерфейс системы управления тестированием и интегрировать серверную часть к реализованному пользовательскому интерфейсу. Тема данной учебной практики была предложена компанией YADRO.

1. Постановка задачи

Целью работы является проектирование, реализация пользовательского интерфейса для варианта с открытым исходным кодом системы TestY и интеграция с серверной частью. Для её выполнения были поставлены следующие задачи:

1. Спроектировать мокапы страниц отображения тест-планов, создания тест-планов и страницы проекта;
2. Реализовать заголовочную часть веб-приложения, страницу проекта с возможностью фильтрации и страницу профиля пользователя;
3. Интегрировать серверную часть и базу данных;
4. Реализовать локализацию приложения;
5. Добавить тёмную тему интерфейса;
6. Протестировать реализованный интерфейс и провести его апробацию среди сотрудников компании YADRO;

2. Обзор существующих решений

Были рассмотрены наиболее популярные системы управления тестированием и проанализированы исходя из требований компании YADRO. К основным требованиям компании можно отнести: наличие тест-кейсов как отдельных сущностей с приоритетами, группировки тест-кейсов в наборы и возможности импортирования их и наборов из других проектов, создание тест-планов и произвольное объединение выбранных пользователем тест-кейсов в тест-план, поддержка конфигураций, специально обрабатываемых системой меток тестов, наличие фильтрации тестов в тест-плане, настраиваемости вердиктов результатов тестов.

- TestRail — популярная система, которая содержит много устаревших решений, начиная от дизайна и заканчивая сущностями. К плюсам данной системы можно отнести полную документацию тестовых шагов, наличие наборов тест-кейсов, интеграция с Jira[5] и генерирование отчетов по наиболее востребованным показателям. Однако, система включает в себя три отдельные, но близкие по функциональности сущности: test plan, test run, milestone, отсутствует возможность добавления ролей для групп пользователей, а также фильтрация тестов по параметрам запуска.

- Test IT — российская система, предназначенная под автоматизацию. Современная система управления тестированием, которая позволяет импортировать тест-кейсы, имеет систему уведомлений, а также интеграцию с Jira. Приятной особенностью является персонализация текста при создании тест-кейсов, тест-планов. К минусам можно отнести невозможность добавления пользовательских статусов, отсутствие пользовательских полей для результатов тестов, неприменимость REST API для ручных тестов.

- Kiwi TCMS — крайне простая система с открытым исходным кодом, подходящая для небольшого проекта. Несмотря на это, си-

[5]Jira. https://www.atlassian.com/ru/software/jira — [Дата обращения: 2022-12-13]

стема содержит гибкую настройку прав групп пользователей, интеграцию с Jira и плагины для сбора результатов автотестов. Однако присутствует огромное количество критических для компании YADRO недостатков, таких как отсутствие проектов, сьют, логики объединения разных кейсов и REST API.

3. Метод

3.1. Проектирование

Для реализации пользовательского интерфейса потребовалось спроектировать систему управления тестированием. В качестве сервиса для прототипирования был использован Ninjamock [7], так как он гораздо проще в использовании и визуальном представлении последовательных действий пользователя по сравнению с популярным онлайн-сервисом Figma [3].

При проектировании были рассмотрены указанные ранее системы управления тестированием, проанализированы требования компании YADRO и отрисованы страницы мокапов (представлено на Рис. 1–3). Затем последовало несколько итераций, включающих в себя получение обратной связи от заказчиков и внесение поправок.

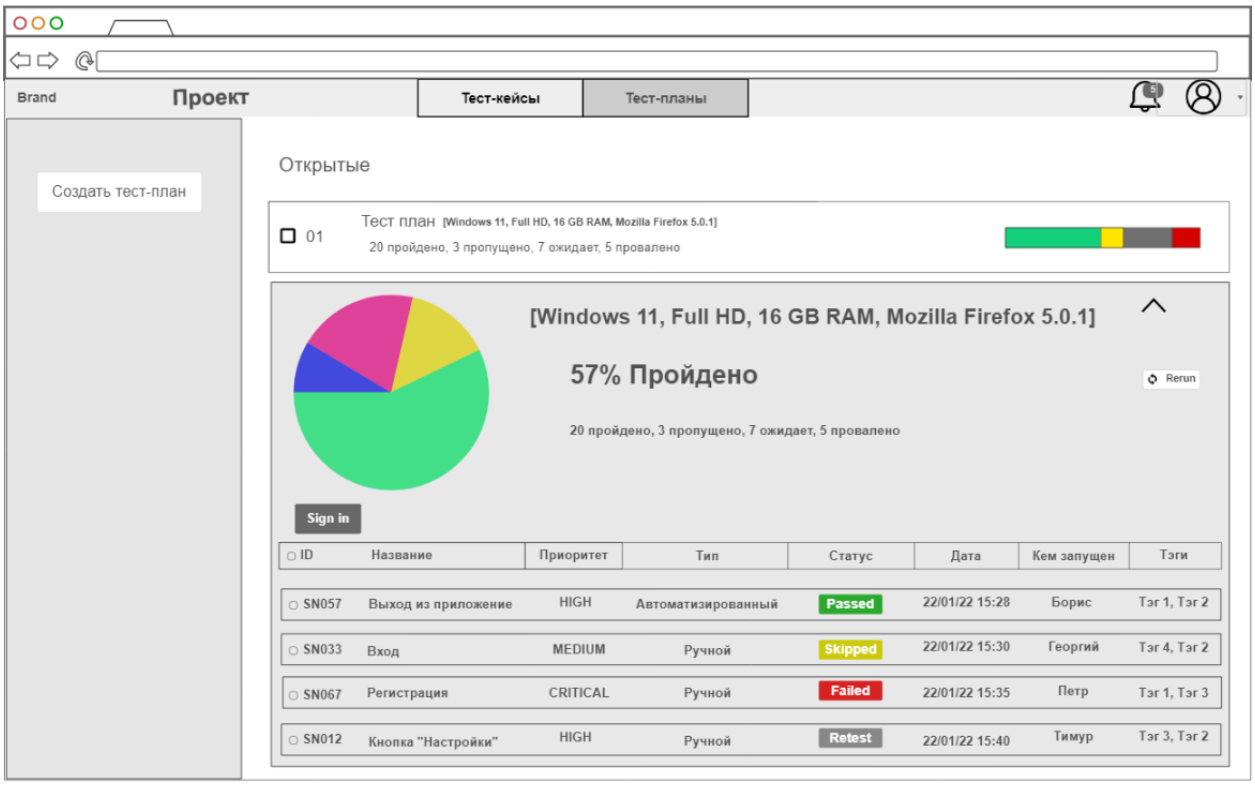

Рис. 1: Первая версия мокапа страницы тест-планов

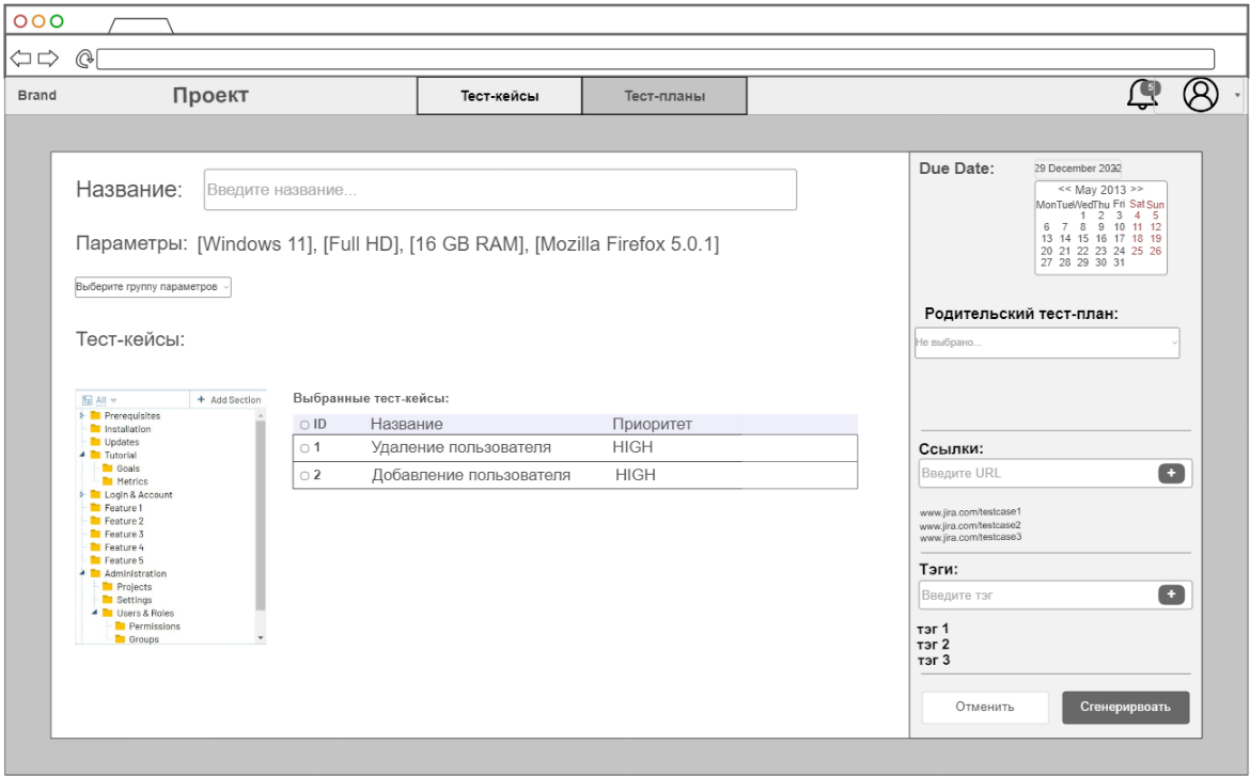

Рис. 2: Первая версия мокапа создания тест-плана

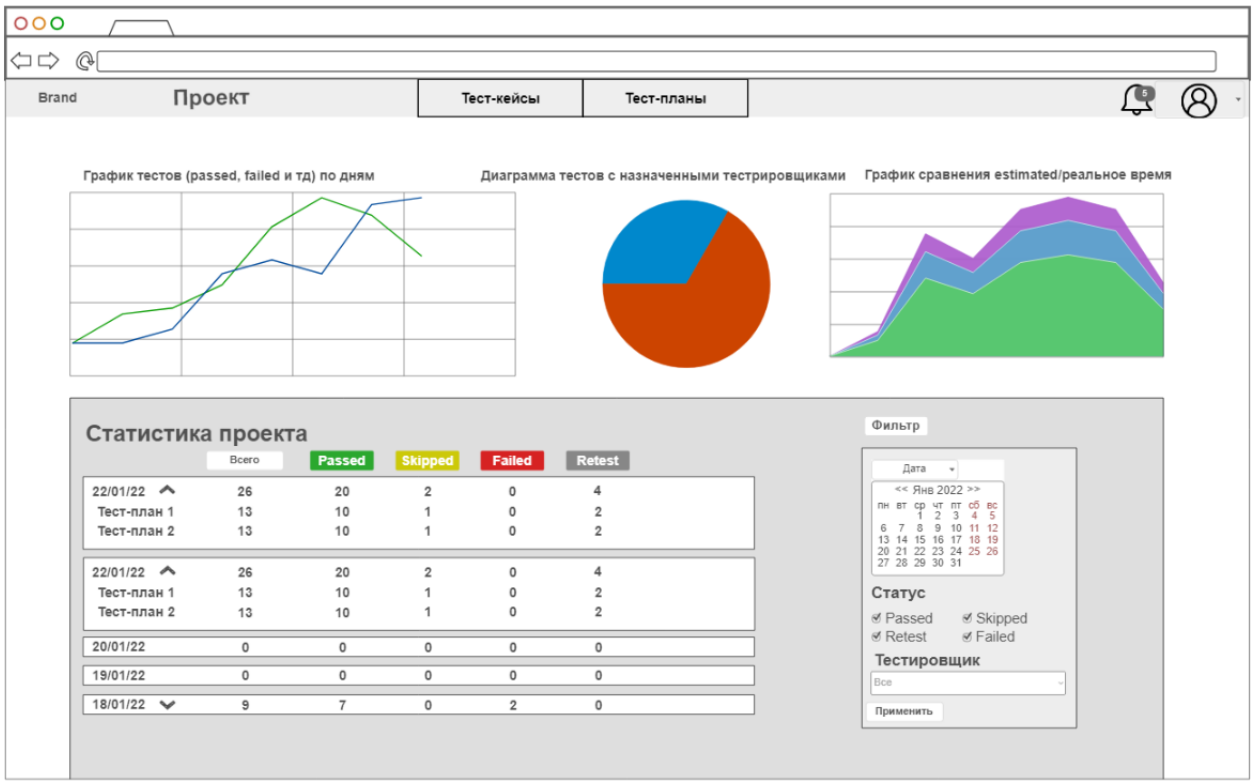

Рис. 3: Мокап страницы проекта

3.2. Реализация

При реализации элементов пользовательского интерфейса в качестве технологий использовались язык TypeScript, фреймворк React [8], так как он прост в изучении и обладает большим количеством библиотек, библиотека Material-UI [5], которая содержит множество различных компонент, библиотека для построения графиков Recharts [9], библиотека для работы с датой и временем Moment.js [6], а также библиотека Axios [1] для работы с HTTP-запросами. В ходе работы были реализованы заголовочная часть веб-приложения и страница проекта (представлено на Рис. 4), а также страница профиля и настроек (представлено на Рис. 5).

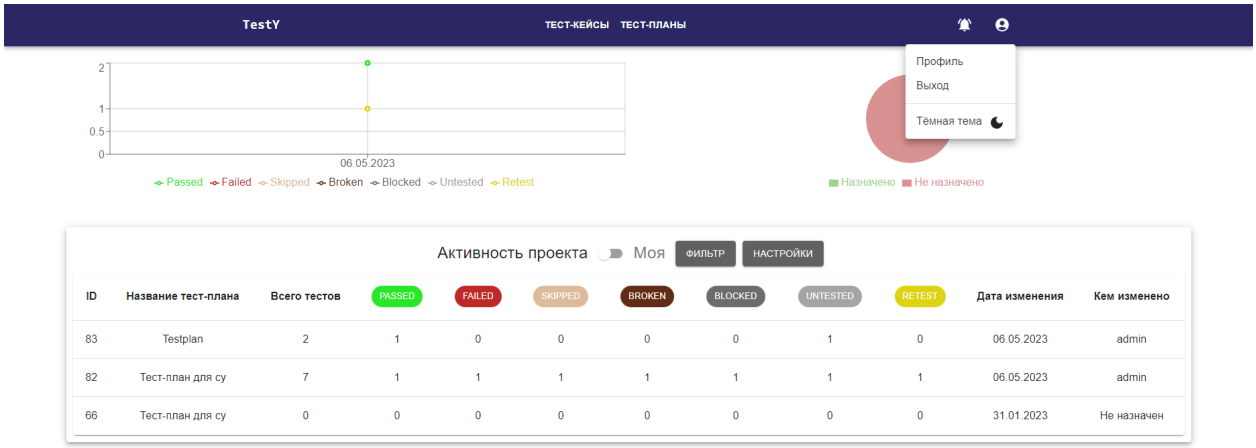

Рис. 4: Страница проекта с реализованной заголовочной частью

Заголовочная часть является основным навигационным элементом данного веб-приложения. При нажатии на логотип системы «TestY», открывается страница выбора проекта, при нажатии на кнопки «Тест-кейсы», «Тест-планы», пользователь переходит на страницы создания, редактирования тест-кейсов и сьют или создания, редактирования тест-планов, тестов и результатов. Также пользователь может перейти на страницу профиля и настроек, переключить тёмный или светлый режимы интерфейса или выйти из текущего аккаунта (представлено на

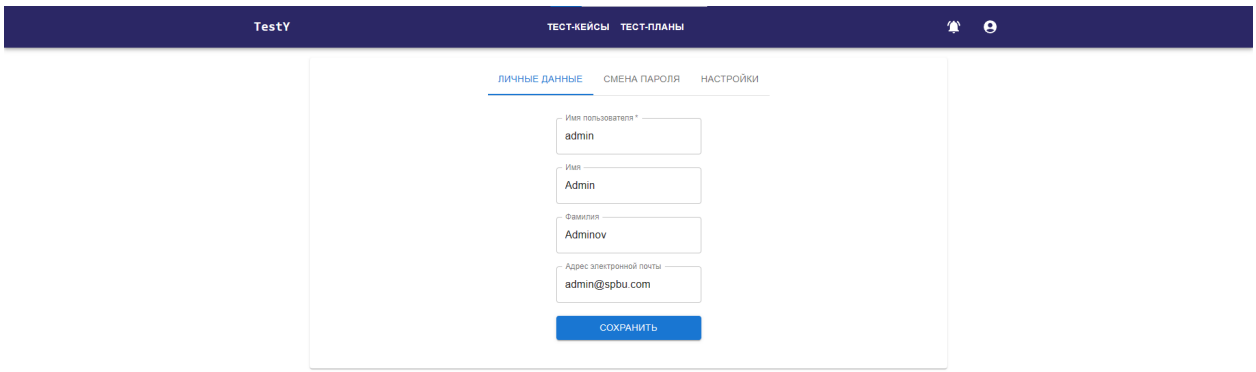

Рис. 5: Страница профиля и настроек

Рис. 5), нажав на иконку пользователя в правом верхнем углу, а затем выбрав соответствующую кнопку.

После выбора проекта пользователь переходит на страницу проекта (представлено на Рис. 4), где посредством HTTP-запроса с помощью Axios происходит парсинг тест-планов, а также результатов тестов. Полученные данные обрабатываются, по ним строятся графики и отображаются отсортированные по дате редактирования данные тест-планов. При нажатии на переключатель справа от надписи «Активность проекта» будут отображены данные тест-планов, где последние изменения были внесены текущим пользователем. При нажатии на кнопку «Фильтр» появляется панель фильтра (представлено на Рис. 6), где пользователь может убрать лишние столбцы статусов тестов, а также с помощью библиотеки Moment.js отфильтровать отображаемые данные по дате.

На странице профиля и настроек (представлено на Рис. 5) расположены три вкладки: «Личные данные», где пользователь, к примеру, может изменить адрес электронной почты, «Смена пароля», где пользователь может сменить пароль, а также «Настройки», где будут пред-

ставлены настройки интерфейса. Для загрузки текущей информации о пользователе, а также для изменения данных пользователя в базе данных были использованы HTTP-запросы с соответствующими данными при помощи ранее упомянутого Axios.

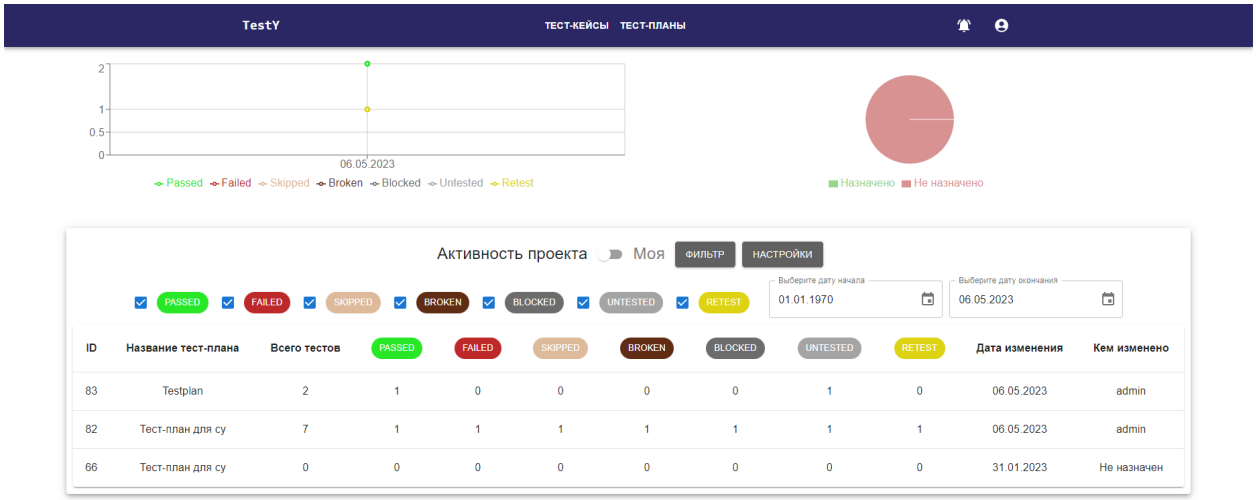

Рис. 6: Страница проекта с панелью фильтров

После открытия страницы профиля и настроек, перейдя на вкладку «Настройки» (представлено на Рис. 7), пользователь может сменить тему, а также региональные настройки интерфейса (пример применения данных параметров представлен на Рис. 8).

Большое внимание было уделено локализации, а именно, были локализованы текст с использованием react-i18next [10], даты с использованием Moment.js [6] и числа с использованием Intl [4]. В директориях `{корень проекта}/frontend/public/locales/{код языка в ISO 639-1}/` расположены JSON файлы `translation.json`, которые в формате «ключ: значение» содержат перевод. Для отображения перевода необходимо вызвать функцию, полученную из хука useTranslation(), в качестве аргумента передать ключ, которому соответствует перевод. Для поддержки большего количества регионов в `README.md` расположена инструкция по добавлению новых переводов.

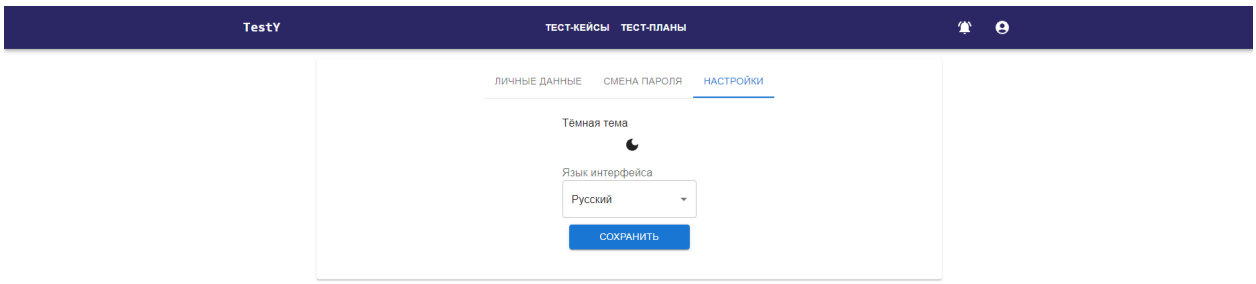

Рис. 7: Вкладка настроек на странице профиля

При проектировании тёмной темы очень важно следить за оттенками различных компонент так, чтобы они не только не сливались, но и выглядели гармонично. В Material-UI [5] по умолчанию поддерживаются два варианта интерфейса: светлый и тёмный. Однако простого копирования примера из документации недостаточно, так как интерфейс приложения содержит множество цветов и цветовых эффектов, таких как наведение, оттенки которых необходимо подбирать вручную.

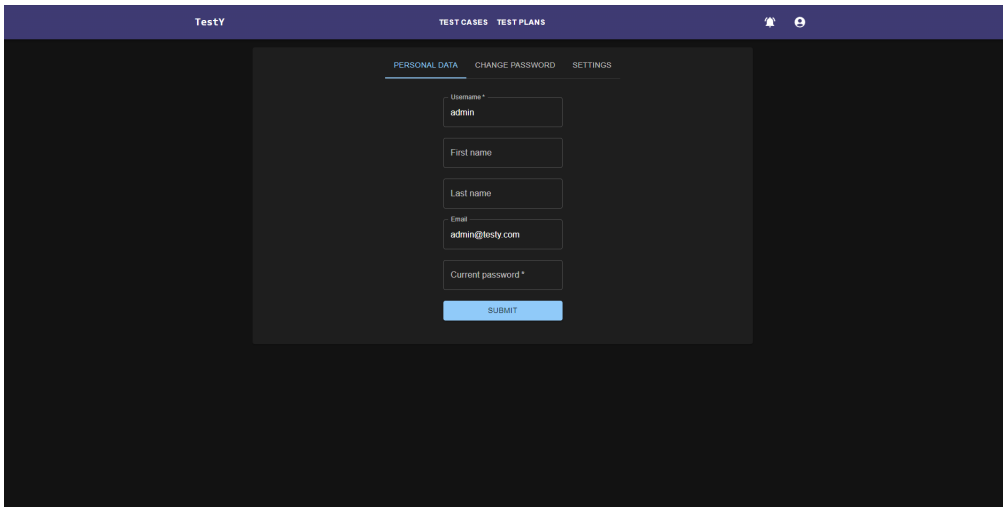

Рис. 8: Страница профиля с применённой локализацией и тёмной темой

Серверная часть проекта постоянно поддерживается и улучшается компанией YADRO, поэтому важно следить за изменениями, которые в большинстве случаев влияют на пользовательский интерфейс. Одним из нововведений было появление шагов тестирования у тест-кейсов. Для добавления поддержки новой сущности были изменены страница создания тест-кейса, его отображения, а также страница результата (представлено на Рис. 9–11). При создании тест-кейса с шагами тестирования необходим порядок следования шагов. Для этого была использована библиотека react-beautiful-dnd [11], которая предоставляет удобный механизм перетаскивания элементов для их сортировки.

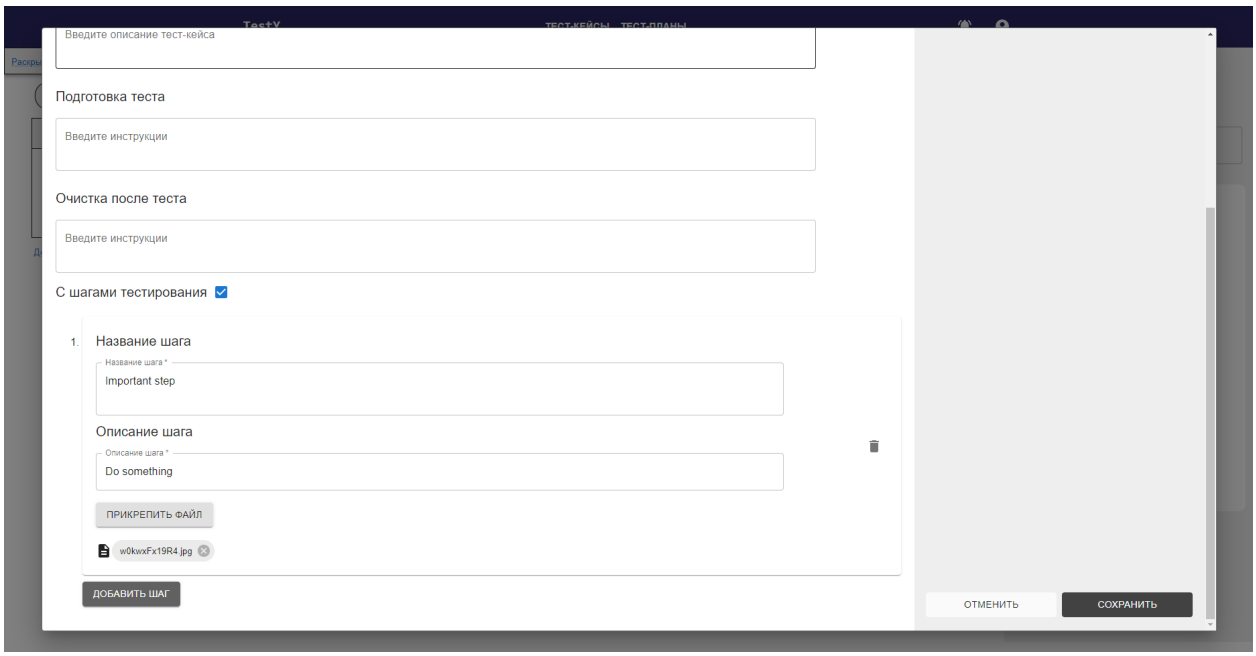

Рис. 9: Шаги тестирования при создании тест-кейса

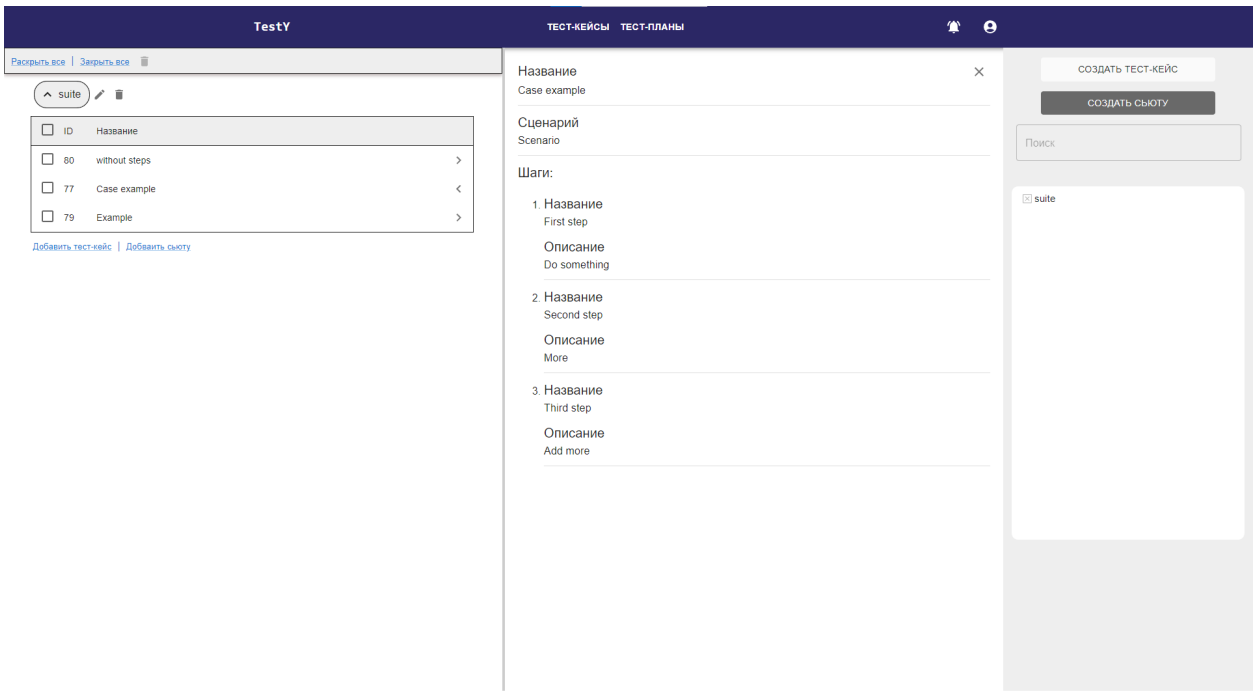

Рис. 10: Отображение шагов тестирования на странице тест-кейса

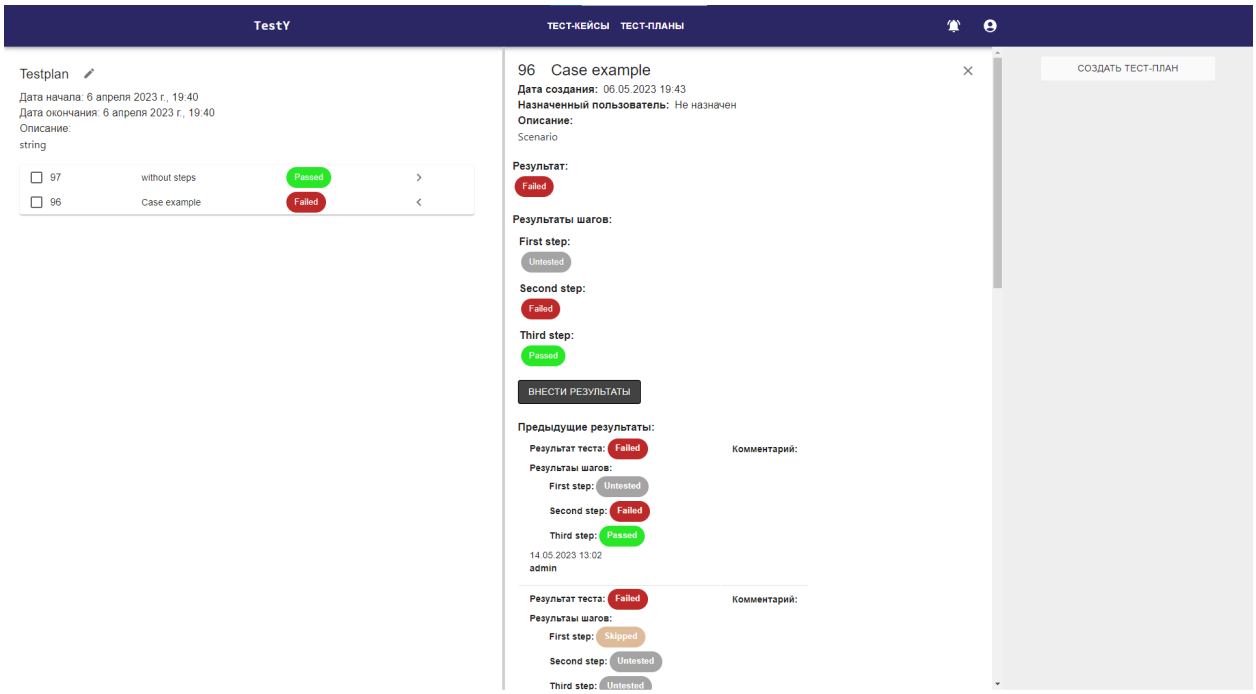

Рис. 11: Страница тестплана и результата с шагами тестирования

4. Эксперимент

Для проверки качества и работоспособности интерфейса были реализованы сквозные тесты (end-to-end) с использованием библиотеки Cypress [2] (представлено на Рис. 12). Данные тесты позволяют проводить полную проверку работы системы, от ввода данных пользователем до обработки пользовательских действий и отображения результатов. Все эти проверки позволяют обнаружить потенциальные проблемы в работе интерфейса и убедиться, что система функционирует корректно. Благодаря использованию библиотеки Cypress, можно автоматизировать выполнение этих тестов, что значительно упрощает их проведение и повышает эффективность процесса тестирования.

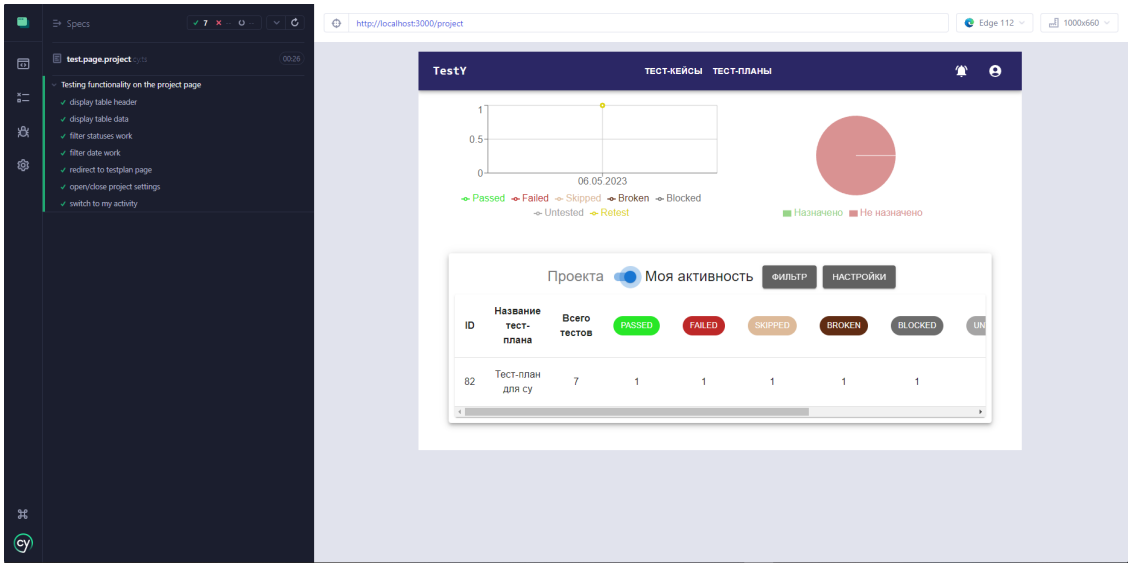

Рис. 12: Сквозные тесты в Cypress для страницы проекта

Заключение

В ходе выполнения данной работы были получены следующие результаты.

- Проектирование (мокапы):
  - Страница отображения тест-планов
  - Страница генерации тест-планов
  - Отображение результатов на странице тест-планов
  - Страница проекта
- Реализация:
  - Заголовочная часть веб-приложения
  - Страница проекта с возможностью фильтрации
  - Страница профиля и настроек
  - Интеграция с серверной частью и базой данных
  - Локализация приложения
  - Тёмная тема интерфейса
  - Тестирование реализованного интерфейса

Код доступен в репозитории GitHub[6].

[6] https://github.com/Belgrak/testy-tms-1. [Имя аккаунта: Belgrak]

Список литературы

[1] Axios. — URL: https://axios-http.com/docs/intro (дата обращения: 2022-12-11).

[2] Cypress. — URL: https://www.cypress.io/ (дата обращения: 2022-12-11).

[3] Figma. — URL: https://www.figma.com/ (дата обращения: 2022-12-11).

[4] Intl. — URL: https://learn.javascript.ru/intl/ (дата обращения: 2022-12-11).

[5] MaterialUI. — URL: https://mui.com/ (дата обращения: 2022-12-11).

[6] Moment.js. — URL: https://momentjs.com/docs/ (дата обращения: 2022-12-11).

[7] Ninjamock. — URL: https://ninjamock.com/ (дата обращения: 2022-12-11).

[8] React. — URL: https://reactjs.org/ (дата обращения: 2022-12-11).

[9] Recharts. — URL: https://recharts.org/en-US/api (дата обращения: 2022-12-11).

[10] react-18next. — URL: https://react.i18next.com/ (дата обращения: 2022-12-11).

[11] react-beautiful-dnd. — URL: https://github.com/atlassian/react-beautiful-dnd (дата обращения: 2022-12-11).