Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ

Школа бакалавриата

ОТЧЕТ

По проекту
«Разработка веб-сервиса для создания личного и командного портфолио»

по дисциплине «Проектный практикум»

Заказчик: Путинцева Т.А.

Куратор: Путинцева Т.А.

ученая степень, ученое звание, должность

Студенты команды Деловые люди

Авдеенко М.А.

Горьков В.Д.

Муравьев И.Г.

Сапегин М.Е.

Якушечкин А.В.

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3
1 Основная часть 5
1.1 Работа участников команды 5
1.1.1 Тимлид 5
1.1.2 Аналитик 5
1.1.3 Дизайнер 5
1.1.4 Бэкендер 6
1.1.5 Фронтэндер 6
1.2 Требования заказчика и составление плана действий 7
1.3 Анализ аналогов разрабатываемого продукта 7
1.4 Архитектура программного продукта 8
1.5 Описание методологии, процесс разработки и тестирования 9
1.5.1 Описание методологии 9
1.5.2 Процесс разработки 10
1.5.3 Результаты тестирования 10
1.6 Планирование деятельности в ходе разработки 11
ЗАКЛЮЧЕНИЕ 12
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 14
ПРИЛОЖЕНИЕ А (обязательное) Функциональные требования 15

ВВЕДЕНИЕ

Целью проекта является разработка веб-сервиса, предназначенного для создания и демонстрации профессиональных портфолио команд и фрилансеров, а также для удобного поиска кандидатов рекрутерами и HR-специалистами.

Задачи проекта:

1) Разработать функционал для создания индивидуальных портфолио фрилансеров с возможностью демонстрации навыков и проектов;

2) Разработать функционал для создания командных портфолио с возможностью демонстрации вклада каждого участника;

3) Обеспечить возможность загрузки и просмотра мультимедийного контента (видео, фото) без необходимости скачивания;

4) Реализовать возможность делиться ссылкой на карточку с рекрутерами;

5) Обеспечить безопасность и конфиденциальность данных пользователей.

Актуальностью проекта является важность наличия возможности наглядно демонстрировать свой опыт и выполненные проекты, так как в современном мире конкуренция на рынке труда очень высока. В частности, фрилансеры часто сталкиваются с проблемой убедительной презентации своих навыков и проектов. Грамотно составленная карточка и наглядное портфолио будут являться одним из ключевых факторов успеха при конкуренции. Студенты и выпускники университетов, которые находятся в поисках профессии, не имеют опыта в составлении портфолио и презентации своих умений, поэтому разрабатываемый сервис поможет им выделиться на фоне других кандидатов и доступно рассказать о себе рекрутерам. Многие команды и стартап-группы также испытывают трудности в создании совместного портфолио, которое бы отражало не только общую работу команды, но и вклад каждого участника по отдельности. Это добавляет трудностей при поиске работы для команды. Помимо фрилансеров, студентов

и команд, трудности также возникают и у рекрутеров, которые вынуждены тратить значительное время на обработку неструктурированной информации о кандидатах. Это снижает эффективность процесса найма сотрудников и затрудняет оценку квалификации будущих специалистов.

Областью применения продукта данного веб-сервиса может быть использования для демонстрации проектных работ студенческих команд и отдельных обучающихся. Также он может быть полезен для представления команды потенциальным работодателям или инвесторам. Помимо этого, сервис может быть использован фрилансерами для привлечения клиентов и получения новых заказов. Разрабатываемая платформа будет упрощать процесс найма сотрудников, а также будет полезна для обмена опытом и налаживания связей в профессиональной среде.

Ожидаемые результаты и планируемые достижения

1) Успешная реализация ключевых функций;
2) Получение положительных отзывов от пользователей;
3) Предоставление командам и фрилансерам возможности наглядно демонстрировать свои достижения и навыки;
4) Повышение эффективности найма сотрудников для рекрутеров;
5) Создание конкурентноспособного портфолио на рынке онлайн-сервисов для рекрутинга.

Таким образом, ожидаемый результат – разработанный и функционирующий веб-сервис, предоставляющий возможности создания и демонстрации портфолио для команд и фрилансеров, а также удобный инструмент для рекрутеров.

1 Основная часть

1.1 Работа участников команды

1.1.1 Тимлид

Использовал YouGile для управления задачами, отслеживания прогресса и поддержания коммуникации в команде. Он активно участвовал в планировании распределении задач между членами команды, и обеспечении соблюдения сроков. Для корректировки сроков выполнения задач была создана диаграмма Ганта и подробный бэклог.

Тимлид оказывал поддержку другим членам команды, в частности аналитику. Его помощь в анализе целевой аудитории и корректировке ТЗ повысила качество проекта. Это позволило команде избежать потенциальных ошибок и недоработок на ранних стадиях проекта.

Также тимлид периодически проводил тщательное code review, что позволило улучшить качество кода, а также избежать ошибок при тестировании.

Также он создавал презентации для контрольных точек, демонстрируя ход работы и прогресс проекта.

1.1.2 Аналитик

Провел анализ требований заказчика, определив проблему целевой аудитории и её потребности. Подробно изучил возможные аналоги сервиса, выявив их сильные и слабые стороны. На основе анализа целевой аудитории и конкурентов, а также собранных требований заказчика, было сформулировано детальное техническое задание и постановка задачи для разработчиков.

Также аналитик разработал сценарии использования, описал дополнительные функции сервиса. Он активно принимал участие в процессе разработки, корректируя техническое задания и учитывая возможности разработчиков. Помимо этого, аналитик помогал при тестировании проекта, учитывая возможные ошибки и проблемы в работе сервиса.

1.1.3 Дизайнер

Дизайнер разработал макеты всех страниц, которые необходимы при работе сервиса. В процессе разработки дизайна интерфейса веб-сервиса было проведено юзабилити тестирование, которое доказало простоту и доступность сервиса. Также дизайнер обеспечил визуальную привлекательность сервиса для пользователей.

1.1.4 Бэкендер

За всё время работы над проектом Backend разработчик реализовал API для создания личного и командного портфолио на языке программирования Java с использованием фреймворка Spring, библиотеки Hibernate и базы данных PostgreSQL. В первую очередь, была реализована возможность авторизации, чтобы пользователи могли создавать новые аккаунты, а также входить в существующие для выполнения каких-либо действий на сайте. Была добавлена возможность редактирования профиля пользователя, а также реализована возможность для пользователей объединяться в команды и редактировать их состав. Помимо этого, была создана система уведомлений, с помощью которой можно приглашать других пользователей в команды. Также было реализовано создание, получение, редактирование и удаление личных и командных карточек и проектов. Карточки и проекты включают в себя работу с файлами, что также было реализовано со стороны Backend разработчика. Также была спроектирована и реализована база данных.

1.1.5 Фронтэндер

Разработал пользовательский интерфейс проекта с использованием React.js: сверстал страницы и написал стили в соответствии с дизайн-макетами и техническим заданием, обеспечив адаптивность и визуальную целостность. Реализовал логику отображения и взаимодействия компонентов, добившись удобного и интуитивно понятного интерфейса. Реализовал обработку пользовательских действий и взаимодействие с данными.

Активно участвовал в тестировании: находил и устранял ошибки, улучшал поведение интерфейса на разных устройствах. Также участвовал в

развертывании проекта на хостинге, проверял его работоспособность и помогал команде с подготовкой видеоотчётов к контрольным точкам.

1.2 Требования заказчика и составление плана действий

Требования заказчика были уточнены и изложены в ТЗ в подробной форме. Они включают в себя функциональные требования, которые описывают алгоритмы регистрации, авторизации, создания и управления личными и командными портфолио, работу с проектами, а также карточками. Также были описаны нефункциональные требования, которые касаются производительности, масштабируемости и юзабилити.

На основе ТЗ был составлен backlog, который включил в себя следующие разделы: проведение подробной аналитики, разработка дизайнов, реализация алгоритма регистрации и авторизации, разработка модуля профиля пользователя, страницы личных и командных проектов, раздела карточек. Также в backlog были включены этапы тестирования и развертывания на хостинге компании заказчика.

1.3 Анализ аналогов разрабатываемого продукта

В качестве аналогов сервиса были рассмотрены как иностранные, так и отечественные сервисы.

Так, к примеру, конкурентом является сервис Linkedin и Indeed. Это достаточно известные сервисы среди англоязычной аудитории, которые допускают публикацию статей, работ, предоставляют возможности для нетворкинга и осуществляют связь с потенциальными рекрутерами. Но данные сервисы не позволяют публиковать проекты и различные файлы, а также имеют ограниченные возможности для кастомизации. Также был рассмотрен сервис Behance, который предоставляет возможность создавать красивые портфолио для дизайнеров. Он не подходит для многих профессий, не поддерживает командную работу, а также ориентирован на англоязычную аудиторию.

В качестве русских аналогов можно привести такие сервисы, как Portfoliobox, HeadHub. Они достаточно просты в использовании,

присутствуют интеграции с рекрутинговыми сервисами, а также с сервисами для продажи своих работ. Оба эти сервиса предоставляют достаточно ограниченные возможности кастомизации портфолио, а также большинство функций открываются для пользователей, оплативших годовой тариф.

Помимо представленных выше сервисов, также был проанализирован GitHub Pages. Он ориентирован исключительно на разработчиков, требует технических навыков для работы с ним. Работа с сервисом бесплатна, а также он предоставляет достаточно большой функционал для программистов, а также допускает командную работу.

В качестве нашего аналога также можно рассмотреть сервис Teamproject. Он позволяет удобно организовывать документы и папки, предоставляет возможность командной работы, но при этом не дает возможности создавать личные проекты и имеет ограниченный доступ.

1.4 Архитектура программного продукта

Веб-сервис разработан с использованием трехзвенной архитектуры (клиент-сервер). Каждое звено выполняет свою четко определенную функцию, что позволяет независимо разрабатывать, тестировать и развертывать компоненты системы.

Клиентский уровень представляет собой пользовательский интерфейс веб-сервиса. Реализован на базе библиотеки React, что обеспечивает высокую интерактивность и отзывчивость интерфейса. Этот уровень отвечает за отображение данных на страницах, сбор информации от пользователя через кнопки и ссылки, а также отправку HTTP-запросов на сервер. Помимо этого, клиентский уровень валидирует введенные пользователем данные. Взаимодействие с бэкендом происходит через REST API, данные отправляются и получаются в формате JSON.

Серверный уровень реализован на языке Java с использованием фреймворка Spring. Он отвечает за обработку запросов от клиента, бизнес-логику приложения, взаимодействие с базой данных и обеспечение безопасности. Серверный уровень принимает HTTP-запросы от фронтенда и

обрабатывает их. С помощью этого уровня реализована регистрация и авторизация пользователей, создание, редактирование, удаление проектов и карточек. Серверный уровень взаимодействует с фронтендом через REST API, работа с базой данных происходит через Spring Data JPA.

Уровень данных представляет собой систему управления базами данных. В сервисе используется PostgreSQL, так как это достаточно надежная и масштабируемая реляционная база данных. Уровень данных хранит данные в структурированном виде, обеспечивает их целостность и безопасность. Также этот уровень выполняет запросы на выборку, добавление, изменение и удаление данных.

Данное архитектурное решение позволяет легко изменять каждый уровень независимо друг от друга. Это означает, что при необходимости внесения изменений в один уровень, разработчики не затрагивают остальные уровни. Такой подход значительно ускоряет процесс внедрения новых функций, исправления ошибок и адаптации системы под новые требования, поскольку влияние изменений локализовано и легко контролируется. Помимо этого, разделение системы на отдельные уровни делает возможным изолированное тестирование каждого компонента. Это позволяет выявлять и устранять ошибки на ранних стадиях, не затрагивая остальные части системы. Также это позволяет разделить ответственность – каждый член команды может специализироваться на разработке одного из уровней. Это повышает качество работы, ускоряет процесс разработки и облегчает поддержку системы в будущем.

1.5 Описание методологии, процесс разработки и тестирования

1.5.1 Описание методологии

Для разработки веб-сервиса была использована методология Agile. Использование этой методологии позволяет гибко реагировать на изменение требований и обеспечивает быстрое создание минимально жизнеспособного продукта. Разработка велась короткими итерациями продолжительностью 14 дней. Команда самостоятельно принимала решения о том, как лучше

выполнять поставленные задачи. В конце каждого спринта работа демонстрировалась заказчику, проводились ретроспективы для обсуждения улучшений при разработке сервиса. Задачи были организованы в backlog, что позволило выделить самые важные элементы разработки для заказчика и выполнить их в первую очередь.

1.5.2 Процесс разработки

В начале каждого спринта команда обсуждала задачи, описанные в backlog и определяла цель на следующие 2 недели. Каждые 3 дня команда проводила короткое собрание, на котором каждый участник отчитывался о своей работе за последнее время, озвучивал трудности в работе и планировал свои дальнейшие действия. В конце спринта команда демонстрировала выполненную работу заказчику и получала обратную связь. После этого команда обсуждала положительные и отрицательные результаты прошедшего спринта, а также планировала будущую работу.

1.5.3 Результаты тестирования

В процессе разработки использовалось несколько методов для тестирования.

Тимлид периодически проводил code review для выявления ошибок и улучшения качества кода. В процессе code review были выявлены некоторые ошибки, например, неправильная обработка исключений в модуле аутентификации, а также неоптимальный SQL-запрос для получения списка проектов.

Также разработчики и аналитик самостоятельно проводили ручное тестирование продукта после внесения изменений. Так были выявлены и исправлены ошибки валидации при вводе email, пароля и ФИО пользователя, а также решена проблема с отображением изображения в профиле пользователя.

Помимо этого, проводилось коридорное тестирование при участии 5 человек. Это тестирование, при котором прототип продукта демонстрируется людям, которые видят его впервые. Им было предложено создать проект,

добавить в него файлы, а затем создать карточку. Некоторые люди испытали сложности при создании папок из-за неочевидного расположения кнопки. Также тестируемые отметили, что меню сортировки и фильтрации карточек выглядит негармонично.

Все выявленные ошибки и недочеты были исправлены и учтены в дальнейшей разработке. Для предотвращения подобных проблем в будущем было решено уделять больше внимания code review, а также чаще проводить ручное тестирование, так как оно позволяет заметить ошибку сразу после ее появления. Также было принято решение использовать коридорное тестирование в дальнейшей работе.

1.6 Планирование деятельности в ходе разработки

Планирование деятельности и отслеживание прогресса разработки осуществлялось при помощи сервиса Yougile. Таким образом задачи были визуально понятно распределены между участниками. На доске задачи располагались в колонках: «К выполнению», «В работе», «На проверке», «Выполнено». Задачи распределялись между участниками команды с учетом их роли и навыков.

Для управления контроля версиями кода использовалась система Git и репозиторий GitHub. Таким образом команда смогла совместно работать над проектом, откатываться к предыдущим версиям в случае возникновения ошибок. Также использовались ветки для разработки и тестирования новых функций.

ЗАКЛЮЧЕНИЕ

В результате проделанной работы был разработан веб-сервис для создания личного и командного портфолио, который соответствует требованиям, изложенным в техническом задании (ТЗ). Проведенное тестирование позволило выявить и устранить ряд ошибок, обеспечив приемлемый уровень качества продукта.

Разработанный веб-сервис успешно реализует основные функциональные требования, определенные в техническом задании. Пользователи могут успешно зарегистрироваться на платформе, указав необходимые данные и аутентифицироваться для доступа к своим портфолио. Система обеспечивает проверку уникальности электронной почты, как и требовалось. Также пользователи могут заполнять и редактировать данные профиля, добавлять контактные данные.

Пользователи могут создавать проекты, добавлять к ним описание, создавать папки и добавлять туда файлы. Также реализована возможность создания карточек и передача доступа к ней по ссылке незарегистрированному человеку. Помимо этого, реализован доступ к командам и командным проектам.

Из-за ограниченности во времени мы не успели реализовать страницу достижений. Это не является ключевым элементом разрабатываемого сервиса, поэтому мы решили не включать этот элемент в минимально жизнеспособный продукт. Также было не полностью реализовано разграничение прав доступа для различных ролей в команде, так как управлять командой может только ее создатель. Тем не менее, реализованный функционал обеспечивает базовые потребности пользователей и полностью соответствует требованиям заказчика.

По выявленным и устраненным в процессе работы ошибкам можно судить о том, что продукт находится в работоспособном состоянии и выполняет основные функции. Отсутствие формального тестирования не позволяет дать объективную оценку качества продукта. Возможны проблемы

при большом количестве пользователей (более 1000 человек одновременно), а также уязвимости в системе безопасности.

В процессе ручного тестирования было обработано много редких сценариев использования, что помогло устранить дополнительные ошибки, возникающие при работе сервиса. Влияние устраненных ошибок было значительным. Например, исправление ошибки в модуле аутентификации предотвратило несанкционированный доступ к аккаунтам пользователей. Оптимизация SQL-запросов позволила повысить скорость загрузки данных на странице проектов.

Для улучшения качества разработанного продукта можно реализовать страницу «Достижения», которые позволят пользователям лучше ориентироваться при использовании сервиса. Также стоит провести юзабилити тестирование с большим количеством реальных пользователей. Для улучшения пользовательского опыта стоит также реализовать расширенную фильтрацию, которая позволит пользователям более точно находить нужный ему проект.

Развитие продукта и доработки позволят создать более качественный и функциональный веб-сервис для создания личного и командного портфолио.

Таким образом, несмотря на некоторые ограничения и сложности, возникавшие в процессе работы, разработанный веб-сервис является успешным результатом работы команды. Он соответствует требованиям заказчика, обладает приемлемым уровнем качества, а также имеет потенциал для дальнейшего развития. Полученный опыт разработки и тестирования будет полезен в будущем и позволит избежать ошибок в дальнейших проектах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Вигерс, К. Разработка требований к программному обеспечению / К. Вигерс, Дж. Битти. – Санкт-Петербург : Microsoft Press, 2024. – 736 с.

2. Унгер, Р. UX-дизайн. Практическое руководство по проектированию опыта взаимодействия / Р. Унгер, К. Чендлер. – Санкт-Петербург : Символ-Плюс, 2017. – 327 с.

3. Кобёрн, А. Современные методы описания функциональных требований к системам / А. Кобёрн. – Москва : Лори, 2014. – 288 с.

4. Паттон, Д. Пользовательские истории. Искусство гибкой разработки ПО / Д. Паттон. – Москва : Айлиб, 2017. – 393 с.

5. Ильяхов, М. Пиши, сокращай 2025 / М. Ильяхов, Л. Сарычева. – Москва : Альпина Паблишер, 2025. – 427 с.

ПРИЛОЖЕНИЕ А

(обязательное)

Функциональные требования

1. Регистрация и аутентификация

1.1. Регистрации

Пользователь должен иметь возможность зарегистрироваться на сайте, указав:

а) Электронную почту (обязательное поле),

б) Имя пользователя (обязательное поле),

в) Пароль (обязательное поле).

Система должна:

а) Проверять уникальность электронной почты в базе данных.

б) Валидировать пароль на соответствие минимальным требованиям (длина не менее 8 символов).

в) Создавать учетную запись пользователя в базе данных.

г) Отправлять письмо с подтверждением регистрации на указанный адрес электронной почты (доп.требование).

В случае успешной регистрации, пользователь получает уведомление об успешной регистрации.

1.2. Подтверждении регистрации (при реализации подтверждения регистрации).

Пользователь должен иметь возможность подтвердить свою учетную запись, перейдя по ссылке в письме, отправленном на указанный при регистрации адрес электронной почты. После подтверждения учетной записи, пользователь должен быть перенаправлен на главную страницу сайта.

1.3. Аутентификация

Пользователь должен иметь возможность войти в систему, используя свой адрес электронной почты и пароль. Система должна:

а) Проверять введенные учетные данные в базе данных.

б) В случае успешной аутентификации, перенаправлять пользователя на главную страницу сайта.

в) В случае неудачной аутентификации, пользователь получает уведомление о неверных учетных данных.

2. Главная страница

2.1. Общая структура:

Верхнее навигационное меню:

а) Проекты (активная вкладка по умолчанию).

б) Команды.

в) Карточки.

г) Уведомления.

д) Изображение пользователя (ссылка на профиль пользователя).

3. Профиль пользователя

3.1. Отображение информации:

1) Отображение информации, введенной при регистрации:
   а) Электронная почта.
   б) Имя пользователя.
2) Отображение пустых полей (для заполнения):
   а) ФИО.
   б) Дата рождения.
   в) Фото пользователя.
   г) Номер телефона.
   д) Контактные данные: telegram, github, ВК и т.д.

3.2. Редактирование профиля:

Наличие кнопки «Редактирование профиля».

а) При нажатии на кнопку, поля становятся доступными для редактирования.

б) После редактирования, наличие кнопки «Сохранить изменения» и «Отменить изменения».

в) После сохранения изменений, пользователь перенаправляется обратно на страницу профиля с обновленной информацией.

г) При нажатии «Отменить изменения» пользователь перенаправляется обратно на страницу профиля без изменения информации.

4. Страница «Проекты»

4.1. Отображение проектов:

1) Отображение проектов в виде карточек.
2) Каждая карточка должна содержать:
    а) Изображение проекта.
    б) Название проекта (кликабельное, ведет на страницу проекта).

4.2. Создание проекта:

1) Наличие кнопки «Создать проект».При нажатии на кнопку, отображается модальное окно с полями:
    а) Название проекта (обязательное поле).
    б) Описание проекта (необязательное поле).
    в) Фото проекта (необязательное поле).
2) Кнопки «Создать» и «Отмена».
    а) При нажатии «Создать», проект сохраняется в базе данных и отображается на странице проектов в виде карточки.
    б) При нажатии «Отмена», модальное окно закрывается без сохранения.

4.3. Сортировка проектов:

1) Наличие кнопки «Сортировать».
2) При нажатии на кнопку, отображается меню с опциями сортировки:
    а) По дате создания.
    б) По названию.

3) После выбора опции сортировки, проекты отображаются в соответствии с выбранным порядком.

5. Страница конкретного проекта

5.1. Отображение информации:

а) Отображение названия проекта.

б) Отображение описания проекта.

в) Автоматически созданная папка «Итоги».

г) Созданные пользователем папки

5.2. Редактирование информации о проекте:

а) Наличие кнопки «Редактировать информацию».

б) При нажатии на кнопку, информация о проекте становится доступной для редактирования (название, описание, фото).

5.3. Создание папки:

а) Наличие кнопки «Создать папку».

б) При нажатии на кнопку, отображается окно с полем для ввода названия папки.

в) После ввода названия папки, она создается и отображается на странице проекта.

5.4. Работа с файлами в папке:

а) Возможность добавления файлов в папку.

б) Типы файлов: документы (ppt, pptx, docx, txt), фото (jpg, png, gif), звуковые и видеофайлы (mp3, mp4, wav, avi, mkv), исполняемые файлы и архивы (exe, zip, rar, 7z).

6. Страница «Достижения»

6.1. Возможность добавления достижений:

Возможность добавлять файлы по кнопке «Добавить файл»

6.2. Работа с достижениями:

Возможность переименования достижения.

6.3. Отображение достижений:

Отображение всех достижений пользователя.

7. Страница «Команды»

7.1. Отображение команд:

а) Отображение списка команд, в которых пользователь является участником или создателем.

б) Каждая команда должна быть представлена в виде карточки, содержащей:

в) Название команды.

г) Состав команды (список участников).

д) Дата создания команды.

е) Кнопку «Командные проекты» (ведущую на страницу проектов данной команды).

7.2. Создание команды:

1) Наличие кнопки «Создать команду». При нажатии на кнопку, отображается модальное окно с полями:
   а) Название команды (обязательное поле).
   б) Состав команды (поле для добавления участников):
      Поиск пользователей по адресу электронной почты и возможность приглашения новых пользователей в команду по электронной почте.
   в) Кнопки «Создать» и «Отмена».
2) При нажатии «Создать»:
   а) Команда сохраняется в базе данных.
   б) Создатель команды автоматически назначается администратором/лидером команды.
   в) Каждому выбранному участнику отправляется уведомление-приглашение на вступление в команду.
   г) Новая команда отображается в списке команд пользователя.
3) При нажатии «Отмена», модальное окно закрывается без сохранения.

7.3. Управление командой: (Только для создателя/администратора)

а) Возможность редактирования информации о команде (название, состав).

б) Возможность назначения/изменения ролей участников команды (например, администратор, участник).

в) Возможность удаления участников из команды.

8. Страница «Командные проекты»

8.1. Отображение командных проектов:

1) Отображение списка проектов конкретной команды в виде карточек.
2) Каждая карточка проекта должна содержать:
   а) Название проекта (кликабельное, ведет на страницу проекта).
   б) Статус проекта (не начат, в процессе, завершен, приостановлен, ожидает утверждения).
   в) Даты проекта (начало, окончание или дедлайн, либо пустое поле).
   г) Фото проекта (превью).

8.2. Управление командными проектами:

1) При нажатии на кнопку «Создать проект», отображается модальное окно с полями: (Только для создателя/администратора команды)
   а) Название проекта (обязательное поле).
   б) Описание проекта.
   в) Заказчик проекта.
   г) Кнопки «Создать» и «Отмена».
   - При нажатии «Создать», проект сохраняется в базе данных и отображается в списке проектов команды.
   - При нажатии «Отмена», модальное окно закрывается без сохранения.
2) Редактирование проекта: (Только для создателя/администратора команды)
   а) Наличие кнопки «Редактировать» на карточке проекта.

б) При нажатии на кнопку, открывается форма редактирования информации о проекте (название, описание, заказчик, даты, статус).

3) Создание карточки команды: (Только для создателя/администратора команды)
   а) Наличие кнопки «Создать карточку команды».
   б) При нажатии на кнопку, создается карточка команды, которая отображается у всех участников команды в разделе «Карточки -> Командные».
4) Покинуть команду: (Доступно всем участникам команды)
   а) Наличие кнопки «Покинуть команду».
   б) При нажатии на кнопку, отображается окно подтверждения.
   в) После подтверждения, команда удаляется из списка команд пользователя.

8.3. Разграничение прав доступа.

Четкое определение ролей в команде (создатель, администратор, участник) и их прав доступа к функциональности управления проектами.

9. Страница «Карточки»

9.1. Общий вид:

а) Фильтрация карточек: «Личные» / «Командные».
б) Наличие кнопки «Создать карточку» (только при выбранном значении фильтра «Личные»).

9.2. Личные карточки:

а) Отображение всех личных карточек пользователя.
б) Личные карточки видны только создателю.

9.3. Командные карточки:

а) Отображение карточек, созданных тимлидом/администратором команды (им самим, если он является тимлидом).
б) Командные карточки видны всем участникам команды.

9.4 Создание карточки:

1) При нажатии на кнопку «Создать карточку» открывается форма с полями:
   а) Название карточки (обязательное поле).
   б) Фото карточки.
   в) Описание карточки.
   г) Файлы.
   д) Описание к каждому файлу.
   е) Ссылки на проекты (выбор из загруженных проектов).
2) После создания карточки, она отображается в разделе «Личные».

9.5. Функциональность карточки:

1) Возможность копирования ссылки на карточку для предоставления рекрутерам (просмотр без регистрации).
2) Предоставление возможности просмотра файлов.
3) Предоставление возможности просмотра проектов.

10. Вкладка «Уведомления»

10.1. Общий вид:

Отображение списка уведомлений пользователя в хронологическом порядке (от новых к старым).

10.2. Типы уведомлений:

1) Приглашения в команду:
   а) Уведомление о приглашении в команду от другого пользователя.
   б) Должно содержать:
   - Название команды.
   - Имя пользователя, пригласившего в команду.
   - Кнопки «Принять» и «Отклонить».
   - Подтверждение/Отклонение заявки на вступление в команду (для создателя/администратора команды):
2) Другие уведомления:
   а) Уведомления об изменениях в статусе проектов

б) Уведомления о создании карточки проекта (личной/командной)

10.3. Функциональность:

1) Принятие приглашения в команду:

При нажатии на кнопку «Принять» рядом с уведомлением о приглашении в команду:

а) Пользователь становится участником команды.

б) Команда добавляется в список команд пользователя на странице «Команды».

в) Уведомление удаляется из списка.

2) Отклонение приглашения в команду:

При нажатии на кнопку «Отклонить» рядом с уведомлением о приглашении в команду:

а) Пользователь отклоняет приглашение.

б) Уведомление удаляется из списка.

в) Принятие заявки на вступление в команду (для администратора):

3) Просмотр других уведомлений:

Пользователь может просматривать и отмечать прочитанными другие типы уведомлений.