Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

ИНСТИТУТ ЭЛЕКТРОННЫХ И ИНФОРМАЦИОННЫХ СИСТЕМ

**ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ**

разработка САЙТА-ПРИЛОЖЕНИЯ «MOOD TRACKER»

Курсовой проект по учебной дисциплине

«Разработка кода информационных систем»

по специальности 09.02.07 Информационные системы и программирование

ПТК.КП 2993 16.000ПЗ

|  |
| --- |
| Руководители  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ФИО  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 года |
| Студент группы 2993  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ФИО  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 года |

# Содержание

[Введение 5](#_Toc166178942)

[Раздел 1. Общая (теоретическая) часть 6](#_Toc166178943)

[1.1 Постановка задачи (ТЗ) 6](#_Toc166178944)

[1.1.1 Характеристики бизнес-процессов 7](#_Toc166178945)

[1.1.2 Технико-математическое описание задачи 9](#_Toc166178946)

[1.1.3 Требования к программе 10](#_Toc166178947)

[1.2 Обоснование проектных решений 13](#_Toc166178948)

[1.2.1 Обоснование выбора языков программирования 14](#_Toc166178949)

[1.2.2 Инструментальные средства 15](#_Toc166178950)

[1.2.3 Обоснование выбора среды программирования 16](#_Toc166178951)

[1.2.4 Информационное обеспечение 16](#_Toc166178952)

[1.3 Обзор существующих программных систем 16](#_Toc166178953)

[Раздел 2. Практическая составляющая 17](#_Toc166178954)

[2.1 Реализация 17](#_Toc166178955)

[2.1.1 Выбор технических и программных средств 17](#_Toc166178956)

[2.1.2 Практическая реализация алгоритмов 18](#_Toc166178957)

[2.1.3 Используемые методы 18](#_Toc166178958)

[2.1.4 Используемые методы (примеры) 19](#_Toc166178959)

[2.1.5 Реализация интерфейса 22](#_Toc166178960)

[2.2 Тестирование 22](#_Toc166178961)

[Заключение 30](#_Toc166178962)

[Список литературы 31](#_Toc166178963)

[Приложение А 31](#_Toc166178964)

[Приложение Б 34](#_Toc166178965)

[Приложение В 38](#_Toc166178966)

# Аннотация

Этот проект направлен на разработку комплексного веб-приложения для отслеживания настроения. Приложение позволит пользователям вносить записи о своем настроении, выявлять паттерны и получать понимание своего эмоционального благополучия.

# Введение

Наши эмоциональные сферы - это сложные паттерны, которые постоянно меняются, формируя наш личный опыт и влияя на наше взаимодействие с окружающим миром. Однако понимание этих эмоциональных колебаний часто представляет собой сложную задачу, подобную путешествию по лабиринту без географической карты. Проект “Mood Tracker” служит надежным проводником в этом очень сложном процессе познания нашего внутреннего мира, помогая четко осветить ландшафт нашего эмоционального самосознания.

Цель проекта - разработать и усовершенствовать простое, функциональное веб-приложение, предназначенное для отслеживания, анализа и понимания индивидуальных эмоциональных паттернов человека. Удобные и интуитивно понятные инструменты, такие как возможность фиксировать настроение, визуализировать собранные данные и персонализированное отслеживание, отправят пользователей в увлекательное путешествие по самопознанию и эмоциональному самоанализу. Независимо от того, сталкиваетесь ли вы с ежедневными проблемами или стремитесь улучшить общее психическое состояние, “Mood Tracker” станет верным спутником, предоставляя жизненно важную информацию и способствуя внутренней связи с собственным эмоциональным "я".

Это приложение предназначено для:

1. Желающих понять собственные эмоциональные модели;
2. Выявить триггеры и причины перепадов настроения;
3. Отслеживать прогресс в улучшении своего эмоционального состояния;
4. Разработать механизмы преодоления сложных эмоций;

Результат проекта:

1. Полнофункциональное веб-приложение с указанными выше возможностями.
2. Документация, описывающая дизайн, реализацию и функциональность проекта.
3. Руководство пользователя, объясняющее, как эффективно использовать приложение.

Для достижения поставленной цели будут решены следующие задачи:

1. Анализ предметной области и требований к приложению;
2. Выбор технологий и инструментов разработки;
3. Проектирование архитектуры и функциональности приложения;
4. Разработка пользовательского интерфейса;
5. Тестирование и отладка приложения;
6. Развертывание и публикация приложения.

# Раздел 1. Общая (теоретическая) часть

## Постановка задачи (ТЗ)

Задачи:

Регистрация настроения: Разработать систему, позволяющую пользователям легко записывать свое настроение, включая возможность выбора одной из эмоций или использования шкалы настроения.

Ведение дневника: Внедрить функцию, позволяющую пользователям вести дневник о своем дне и связывать записи с их настроением.

Визуализация: Создайте визуальные представления данных о настроении, например, графики или диаграммы, чтобы помочь пользователям выявить тенденции и закономерности.

Персонализация: Позволяйте пользователям настраивать свой опыт, устанавливая цели, выбирая цветовые темы и предпочтительные методы отслеживания настроения.

Безопасность и конфиденциальность: Обеспечьте безопасное хранение данных пользователей и разработайте четкую политику конфиденциальности в отношении использования данных.

Сфера деятельности:

Front-End Development: Проектирование и реализация пользовательского интерфейса (UI) с использованием HTML, CSS и фреймворков JavaScript (например, React, Angular, Vue.js).

Back-End разработка: Разработка логики на стороне сервера с использованием подходящего языка программирования (например, Python, Java, Node.js) и базы данных (например, MySQL, MongoDB).

Аутентификация пользователей: Реализуйте безопасную систему регистрации пользователей, входа в систему и управления сессиями.

Визуализация данных: Использование библиотек диаграмм или API для создания интерактивных визуализаций данных о настроении.

Развертывание: Разверните веб-сайт на подходящей хостинговой платформе, обеспечив доступность и масштабируемость.

## 1.1.1 Характеристики бизнес-процессов

Несмотря на то, что проект “Mood Tracker” в первую очередь ориентирован на индивидуальный опыт пользователя и его самочувствие, необходимо рассмотреть ряд бизнес-процессов:

1. Управление пользователями:

Управление пользователями: регистрация и аутентификация: безопасные системы для регистрации, входа и управления учетными записями пользователей.

Конфиденциальность и безопасность данных: меры по обеспечению конфиденциальности пользовательских данных и соблюдению соответствующих норм защиты данных.

Поддержка пользователей: обеспечение каналов, по которым пользователи могут запрашивать помощь, сообщать о проблемах и оставлять отзывы.

2. Управление данными

Сбор и хранение данных: надежное хранение эмоциональных данных пользователей, включая журналы, заметки и сопутствующую информацию.

Анализ и визуализация данных: обработка и анализ пользовательских данных для создания удобных для восприятия графиков, диаграмм и отчетов.

Резервное копирование и восстановление данных: реализация стратегий по предотвращению потери данных и обеспечению их целостности.

3. Разработка и обслуживание приложений

Непрерывная разработка: Регулярное обновление приложения новыми функциями, исправление ошибок и усовершенствование на основе отзывов пользователей и тенденций рынка.

Мониторинг производительности: отслеживание производительности приложения, выявление "узких мест" и оптимизация скорости и отзывчивости.

Обновления безопасности: Регулярное применение патчей и обновлений безопасности для предотвращения уязвимостей.

4. Монетизация (опционально):

Модель подписки: премиум-функции, такие как расширенный и персонализированный анализ данных, доступны за периодическую плату.

Free-модель: базовые функции предоставляются бесплатно, а дополнительные - за счет покупок в приложении.

Партнерские отношения: Сотрудничество с профессионалами в области психического здоровья и оздоровительными организациями для предоставления индивидуальных услуг и контента в приложении.

5. Маркетинг и продвижение

Цифровой маркетинг: Использование онлайн-каналов, таких как социальные сети, контент-маркетинг и оптимизация поисковых систем, для привлечения целевой аудитории.

Оптимизация магазинов приложений: оптимизируйте списки приложений в соответствующих магазинах приложений, чтобы повысить их узнаваемость.

Связи с общественностью и партнерство: Налаживайте отношения со СМИ и соответствующими ассоциациями для продвижения приложения и его преимуществ.

Другие соображения.

Доступность: Убедитесь, что приложение доступно для людей с ограниченными возможностями.

Этические аспекты: рассмотрите возможные этические проблемы, связанные с конфиденциальностью данных, манипулированием пользователями и влиянием отслеживания настроения на психическое здоровье.

Соблюдение правовых норм: соблюдайте соответствующие законы и правила, касающиеся защиты данных, прав потребителей и онлайн-сервисов.

Эффективное управление этими бизнес-процессами позволит создать устойчивое и успешное приложение, которое принесет пользу как пользователям, так и самому проекту.

## 1.1.2 Технико-математическое описание задачи

Для реализации были использованы следующие средства:

HTML - это стандартизированный язык разметки для документов в Интернете. Большинство веб-страниц содержат описания разметки HTML; HTML интерпретируется браузером, позволяя отображать форматированный текст на экранах компьютеров и мобильных устройств. До версии 5 HTML определялся как реализация SGML (Standard Generalised Markup Language согласно ISO 8879); спецификация HTML5 сформулирована в терминах DOM (Document Object Model). В Интернете HTML-страницы обычно отправляются с сервера в браузер в виде обычного текста или в зашифрованном виде по протоколам HTTP или HTTPS.

CSS - это формальный язык, используемый для описания внешнего вида документов, написанных с помощью языков разметки. В основном он используется для описания внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но также может применяться к XML-документам, таким как SVG и XUL.

HTTP (HyperText Transfer Protocol) - протокол прикладного уровня для передачи данных; HTTP основан на технологии клиент-сервер; хотя основной целью HTTP является передача веб-страниц (текстовых файлов, содержащих разметку HTML), он может использоваться для передачи файлов, связанных с веб-страницами (изображений и приложений), а также не связанных с ними файлов. HTTP позволяет программе-клиенту, т. е. веб-браузеру, отображать гипертекстовые веб-страницы и другие типы файлов удобным для пользователя способом. Предполагается, что программа-клиент, т. е. веб-браузер, способна отображать гипертекстовые веб-страницы и другие типы файлов удобным для пользователя способом

## Требования к программе

* + - 1. Функциональные требования:
         1. Аутентификация пользователей:

Система должна позволять пользователям создавать учетные записи и безопасно входить в систему.

Система должна предоставлять возможности входа с помощью сторонних социальных сетей (например, Google, Facebook, ВК, Mail.ru).

* + - * 1. Регистрация настроения:

Пользователи должны иметь возможность регистрировать свое настроение в любое время.

Система должна предоставлять различные варианты ввода настроения (например, выбор эмодзи, текстовое описание, голосовой ввод).

Пользователи могут добавлять примечания к записям настроения.

* + - * 1. Отслеживание настроения:

Система должна отображать записи о настроении в виде календаря.

Пользователь может просматривать прошлые записи о настроении и заметки.

Система должна иметь возможность фильтровать записи настроения по диапазону дат и типу настроения.

* + - * 1. Визуализация:

Система должна отображать тенденции настроения с помощью графиков или диаграмм.

Пользователь может выбирать различные типы визуализации (гистограммы, линейные графики и т. д.).

* + - * 1. Ведение журнала:

Пользователь может создавать и редактировать записи в журнале.

Записи в журнале могут быть связаны с конкретными записями настроения.

* + - * 1. Персонализация:

Пользователи могут устанавливать напоминания для записей настроения.

Пользователь может настраивать параметры записи настроения и метки настроения.

Пользователь может выбирать различные темы для пользовательского интерфейса.

* + - 1. Нефункциональные требования
         1. Производительность:

Система должна быстро загружаться и реагировать на действия пользователя.

Система должна быть способна обрабатывать большое количество записей настроения без снижения производительности.

* + - * 1. Безопасность:

Пользовательские данные должны быть зашифрованы и надежно храниться.

Система должна принимать меры для предотвращения несанкционированного доступа.

* + - * 1. Удобство использования:

Система должна быть простой в использовании и навигации.

Пользовательский интерфейс должен быть интуитивно понятным и визуально привлекательным.

Система должна предоставлять полезные инструкции и сообщения об ошибках.

* + - * 1. Доступность:

Система должна быть доступна для пользователей с ограниченными возможностями.

Система должна соответствовать стандартам доступности (например, WCAG).

Система должна быть доступна 24 часа в сутки 7 дней в неделю с минимальным временем простоя.

Система должна иметь план аварийного восстановления.

* + - * 1. Другие соображения.

Экспорт данных: пользователи должны иметь возможность экспортировать данные о своем настроении для личного анализа и обмена с медицинскими работниками.

Интеграция: необходимо предусмотреть интеграцию с другими приложениями для здоровья и носимыми устройствами.

Конфиденциальность: политика конфиденциальности и методы обработки данных должны быть четко разъяснены.

## 1.2 Обоснование проектных решений

В связи с растущей важностью осведомленности о психическом здоровье и потребностью в доступных инструментах для мониторинга эмоционального благополучия и управления им разработка веб-сайта для отслеживания настроения представляет собой своевременную и ценную возможность.

Дизайн, ориентированный на пользователя: Основной целью проекта является создание удобной платформы, которая учитывает потребности и предпочтения своих пользователей. Проводя тщательные исследования пользователей, включая опросы, интервью и юзабилити-тестирование, мы получили представление о конкретных требованиях и ожиданиях людей, стремящихся отслеживать их настроение.

Многофункциональность: Веб-сайт будет предлагать широкий спектр функций, предназначенных для удовлетворения разнообразных потребностей пользователей.

Включает в себя:

Настраиваемый журнал настроений: пользователи смогут гибко записывать свои настроения на основе предопределенных категорий или настраивать свои собственные дескрипторы.

Визуализация данных: Используя диаграммы и графики, пользователи смогут визуализировать тенденции своего настроения с течением времени, что способствует повышению самосознания и проницательности.

Постановка целей и отслеживание прогресса: Платформа позволит пользователям ставить личные цели, связанные с эмоциональным благополучием, и отслеживать прогресс в их достижении.

Ведение журнала и рефлексия: Интегрированные функции ведения журнала позволят пользователям записывать свои мысли и чувства, способствуя саморефлексии и обработке эмоций.

Доступность и инклюзивность: Учитывая важность доступности, веб-сайт будет разработан с учетом потребностей пользователей с различными потребностями, в том числе с ограниченными возможностями. Это предполагает соблюдение стандартов веб-доступности (например, WCAG) и внедрение таких функций, как альтернативный текст для изображений и навигация с клавиатуры.

Конфиденциальность и безопасность: Защита конфиденциальности пользователей и обеспечение сохранности данных имеют первостепенное значение при разработке веб-сайта. Для защиты пользовательской информации будут приняты надежные меры безопасности, включая протоколы шифрования и соблюдение правил защиты данных (например, GDPR).

Масштабируемость и производительность: Предвидя потенциальный рост числа пользователей в будущем и расширение функциональных возможностей, проектные решения будут уделять приоритетное внимание масштабируемости и оптимизации производительности. Это предполагает использование масштабируемой инфраструктуры, эффективных методов программирования и регулярного тестирования производительности для обеспечения бесперебойной работы при различных нагрузках.

## 1.2.1 Обоснование выбора языков программирования

Выбор языков программирования зависит от многих факторов, например, задачи, которые нужно решить. Выбор языков программирования для разработки веб-приложения “Mood Tracker” обусловлен следующими факторами:

HTML (язык гипертекстовой разметки):

Основа веб-страниц: HTML обеспечивает базовую структуру и содержание интерфейса Mood Tracker. Он определяет такие элементы, как заголовки, абзацы, формы и кнопки, которые составляют макет. Семантические элементы HTML обеспечивают доступность приложения для пользователей с ограниченными возможностями и устройств чтения с экрана. Правильная структура HTML помогает поисковым системам понимать и индексировать содержимое приложения.

CSS (каскадные таблицы стилей):

CSS управляет внешним видом и функциональностью “Mood Tracker”. Он стилизует элементы, определяет макеты и создавает визуально привлекательные интерфейсы. Персонализирует пользовательский опыт. Реализует принципы отзывчивого дизайна, чтобы Mood Tracker хорошо смотрелся на различных устройствах (настольных компьютерах, планшетах, мобильных телефонах).

JavaScript:

JavaScript добавляет динамическое поведение в ваш Mood Tracker. Он позволяет реализовать такие функции, как валидация пользовательского ввода, регистрация настроения, обработка данных и интерактивные визуализации. Он взаимодействует с API и внутренними службами для хранения и получения данных о настроении.

## 1.2.2 Инструментальные средства

Для разработки приложения использовались следующие инструментальные средства:

* Visual Studio Code - кроссплатформенная интегрированная среда разработки (IDE), обеспечивающая удобный редактор кода, отладчик и множество полезных расширений для веб-разработки. Она предоставляет широкий набор инструментов для отладки, рефакторинга кода и автоматизации задач.
* Git - распространенная система контроля версий, позволяющая отслеживать изменения в коде, производить откаты и совместную работу над проектом.
* Node.js - программная платформа для выполнения JavaScript-кода на сервере, необходимая для работы с различными инструментами и библиотеками.
* Figma - это онлайн-редактор дизайна, который позволяет создавать макеты и прототипы веб-сайтов, приложений и других пользовательских интерфейсов.
* npm - менеджер пакетов для Node.js, позволяющий устанавливать и управлять зависимостями проекта.

## 1.2.3 Обоснование выбора среды программирования

Браузер был выбран в качестве среды выполнения для данного веб-приложения, так как он предоставляет все необходимые возможности для отображения и выполнения HTML, CSS и JavaScript. Современные веб-браузеры поддерживают новейшие стандарты и технологии, что делает их идеальной средой для разработки и запуска веб-приложений.

## 1.2.4 Информационное обеспечение

Для разработки логики работы веб-приложения используется язык программирования JavaScript.

## 1.3 Обзор существующих программных систем

На рынке существует ряд веб-приложений для отслеживания эмоций. Некоторые из наиболее популярных и широко используемых:

* Дневник эмоций и Осознанность

Мобильное приложение: https://play.google.com/store/apps/details?id=ru.harlion.psy

Мобильное приложение «Дневник эмоций и Осознанность»

предлагает курсы по самопознанию и отслеживание эмоций. Отличается разбиением личного дневника для каждой внутренней личности (внутренний ребенок, внутренний взрослый, внутренний родитель)

* Дневник - Трекер Настроения

Мобильное приложение: https://play.google.com/store/apps/details?id=net.daylio

Это мобильное приложение предоставляет отслеживание эмоций. Отличается наличием легкодоступной статистикой данных в разных формах.

* EMMO

Мобильное приложение: https://play.google.com/store/apps/details?id=emmo.diary.app

Веб-приложение предоставляет отслеживание эмоций и личный дневник. Отличается возможностью нарисовать собственные иконки для эмоций.

Преимущества разрабатываемого приложения заключаются в том, что оно легкое, быстро загружаемое и адаптируемое по дизайну, что позволяет использовать его на различных устройствах и при медленном интернет-соединении. Еще одним преимуществом является возможность гибкой настройки приложения в соответствии с потребностями пользователя.

# Раздел 2. Практическая составляющая

## 2.1 Реализация

## 2.1.1 Выбор технических и программных средств

Для разработки приложения прогноза погоды использовались следующие технические и программные средства:

* Операционная система: Windows 11
* Браузер: Opera GX (для тестирования и отладки)
* IDE: Visual Studio Code
* Система контроля версий: Git
* Инструменты разработки: Node.js, npm (менеджер пакетов для Node.js)

## 2.1.2 Практическая реализация алгоритмов

Алгоритм - это точное предписание, описывающее порядок действий (план действий), определяющих процесс решения задачи или достижения цели.

Основным алгоритмом, реализованным в приложении, является отображение записей пользователя о настроении. Этот процесс включает следующие шаги:

1. Формирование запроса к локальному хранилищу для получения данных;
2. Преобразование данных к удобному для работы формату;
3. Ввод полученных данных в конструкцию HTML с помощью манипуляций с DOM

## 2.1.3 Используемые методы

1. Получение данных: Код использует манипуляции с DOM для получения и хранения введенных пользователем данных.
2. Отображение данных: Полученные данные отображаются на веб-странице с помощью манипуляций с DOM. Код обновляет различные элементы HTML, такие как Календарь эмоций и строка «последние использованные».
3. Настройки приложения: Код позволяет пользователю изменять тему оформления (светлая или темная). Эти настройки сохраняются в локальном хранилище браузера.
4. Дополнительные функции: Веб-приложение также включает в себя вспомогательную функцию определения даты и времени во время входа в аккаунт и создания новых записей в календарь.
5. Обработка событий: Код обрабатывает различные события, такие как отправка формы регистрации, нажатие на кнопки настроек, изменения пароля аккаунта, отправка формы записи эмоции и нажатие на день в календаре для отображения записанных эмоций.

В целом, код демонстрирует различные аспекты веб-разработки, включая манипуляции с DOM, обработку событий, использование локального хранилища и реализацию различных функций для обработки данных и отображения информации на веб-странице.

## 2.1.4 Используемые методы (примеры)

1. Регистрация нового пользователя:

let btn\_reg = document.getElementById("btn\_reg");

function registerUser(){

email = document.getElementById('signUpEmail').value;

password = document.getElementById('signUpPassword').value;

confPassword = document.getElementById('signUpConfirmPassword').value;

if(password == confPassword) {

localStorage.setItem('newUserEmail', email);

localStorage.setItem('newUserPassword', password);

}

btn\_reg.addEventListener('click', () => registerUser());

В этом примере добавляется обработчик события на кнопку регистрации. Используются методы getElementById() для обращения к DOM дереву и получения данных из документа. Полученные данные затем фиксируются в локальном хранилище для последующего использования.

1. Обработка входа зарегистрированного пользователя:

let btn\_log = document.getElementById("btn\_log");

function loginUser(){

email = document.getElementById('signInEmail').value;

password = document.getElementById('signInPassword').value;

const registeredUser = localStorage.getItem("newUserEmail");

const registeredPassword = localStorage.getItem("newUserPassword");

const validUser = email == registeredUser;

const validPassword = password == registeredPassword;

if(validUser && validPassword){

console.log('u did it');

document.location.replace('pages/recEmotion.html');

}

else if(!validUser) alert(`email ${email} has not been registered. `);

else if(!validPassword) alert(`Incorrect password for email ${email}`);

}

btn\_log.addEventListener('click', () => loginUser());

Этот код добавляет обработчик события для формы авторизации. Также он проверяет правильность введенных данных, сверяясь с локальным хранилищем. При ошибке выводит сообщение пользователю. Если ошибок не было допущено, то код отсылает на страницу записи эмоций в календарь.

1. Взаимодействие с главным экраном веб-приложения:

let blob = document.getElementById("blob");

let textsWelcome = document.querySelectorAll(".textOnWelcome");

let signUpForm = document.querySelector("#main");

let card = document.getElementById('card');

let swapCard = document.getElementById('formToSignUp');

blob.addEventListener('click', () => {

textsWelcome.forEach(element => {

element.classList.add('fade-out')

});

signUpForm.setAttribute('style', 'z-index: 2');

blob.setAttribute('style', 'width: 35vw; height: 60vh; animation: none; border-radius: 50px;');

blob.classList.add('no-pseudo');

let active = document.querySelectorAll('.fade-in');

active.forEach((el) => {

el.classList.add('active')

});

});

swapCard.addEventListener('click', () => {

if(card.classList.contains('container') !== false){

card.classList.remove('container');

swapCard.innerText = 'first time';

swapCard.classList.remove('backTitle');

swapCard.setAttribute('style', 'transform: rotateY(0)');

blob.setAttribute('style', 'width: 35vw; height: 60vh; animation: none; border-radius: 50px;');

} else {

swapCard.innerText = 'already registered';

swapCard.setAttribute('style', 'transform: rotateY(180deg)');

card.classList.add('container');

swapCard.classList.add('backTitle');

blob.setAttribute('style', 'width: 35vw; height: 70vh; animation: none; border-radius: 50px;');

};

})

Эта функция добавляет интерактивный элемент на главную страницу веб-приложения. Когда пользователь только заходит на страницу перед ним появляется хаотичная клякса с обработчиком событий. Благодаря вышеуказанному коду эта клякса по нажатию выполняет анимацию превращения в окно-карточку авторизации. Если нажать на надпись «впервые здесь», то произойдет переворот карточки и окно авторизации сменится окном регистрации.

## 2.1.5 Реализация интерфейса

Интерфейс приложения состоит из следующих основных компонентов:

* Поле ввода для местоположения (город или координаты);
* Блок с текущей погодой, включающий температуру, скорость ветра, вероятность осадков и другие данные;
* Блок с прогнозом погоды на ближайшие дни.

Структура интерфейса создана с использованием HTML и семантических тегов. Стилизация выполнена с помощью CSS, обеспечивая современный и адаптивный дизайн. JavaScript используется для обработки взаимодействия с пользователем (ввод записей, регистрация, авторизация, настройки учетной записи) и отображения введенных данных.

## 2.2 Тестирование

В современном цифровом ландшафте хорошо функционирующий веб-сайт имеет решающее значение для процветания предприятий и организаций. Однако создание веб-сайта - это только половина дела; не менее важно обеспечить его качество и функциональность путем тщательного тестирования. Тестирование веб-сайта - это систематический процесс оценки производительности, удобства использования, безопасности и доступности веб-сайта, чтобы убедиться, что он соответствует ожиданиям пользователей и бизнес-целям.

Значение тестирования веб-сайтов

Тестирование веб-сайтов - это не просто вариант, а необходимость по нескольким веским причинам:

Улучшение пользовательского опыта (UX): Успех веб-сайта зависит от его способности обеспечить положительный пользовательский опыт. Тестирование помогает выявить проблемы с юзабилити, навигацией и неработающими функциями, которые препятствуют удовлетворению пользователей. Решив эти проблемы, вы создадите беспроблемный и приятный опыт, что приведет к повышению вовлеченности пользователей и конверсии.

Раннее обнаружение ошибок: Ошибки и дефекты неизбежны при разработке программного обеспечения, но их раннее обнаружение имеет решающее значение для предотвращения дорогостоящих последствий. Тестирование выявляет ошибки на разных этапах, что позволяет своевременно устранить их до того, как они повлияют на пользователей и нанесут ущерб репутации сайта.

Повышение производительности и надежности: Тестирование производительности оценивает такие факторы, как время загрузки страниц, отзывчивость и стабильность при интенсивном трафике. Это помогает обеспечить оптимальное функционирование сайта, предотвратить сбои и замедления, которые расстраивают пользователей и отталкивают их.

Повышенная безопасность: Тестирование безопасности выявляет уязвимости, которыми могут воспользоваться злоумышленники, например, SQL-инъекции, межсайтовый скриптинг или утечка данных. Устранение этих уязвимостей позволяет защитить данные пользователей и целостность веб-сайта.

Доступность для всех: Тестирование на доступность гарантирует, что сайт будет удобен для людей с ограниченными возможностями, в соответствии со стандартами доступности, такими как WCAG. Это способствует инклюзивности и расширяет охват веб-сайта для более широкой аудитории.

Виды тестирования веб-сайтов

Тестирование веб-сайтов включает в себя различные типы, каждый из которых фокусируется на определенных аспектах:

Функциональное тестирование: Этот основной тип тестирования проверяет, что все функции и возможности работают так, как задумано, в соответствии со спецификациями. Оно включает в себя тестирование регистрации пользователей, входа в систему, ввода данных, поиска и других основных процессов.

Тестирование удобства использования: Этот тип тестирования оценивает простоту использования, интуитивность и общий пользовательский опыт. Такие методы, как наблюдение за пользователями, опросы и A/B-тестирование, помогают выявить области для улучшения и обеспечить удобство использования сайта.

Тестирование совместимости: Убеждаются в том, что сайт корректно отображается и функционирует в различных браузерах, операционных системах, устройствах и разрешениях экрана.

Тестирование производительности: Оценивается отзывчивость, стабильность и масштабируемость сайта при различных условиях нагрузки. При этом измеряются такие показатели, как время загрузки страницы, использование ресурсов и одновременная работа пользователей.

Тестирование безопасности: Выявление уязвимостей и слабых мест, которые могут быть использованы злоумышленниками. Оно включает в себя тестирование на проникновение, сканирование уязвимостей и аудит безопасности, чтобы обеспечить защиту пользовательских данных и целостности сайта. Тестирование приложения проводилось на различных устройствах и в разных браузерах (Opera GX, Mozilla Firefox, Microsoft Edge) для проверки корректности отображения и функционирования. Были выявлены и исправлены следующие проблемы:

Таблица

|  |  |  |
| --- | --- | --- |
|  |  | Комментарии |
| Тест № | TC\_MT\_1 | Расшифровка: TestCase\_MoodTracker\_1 |
| Приоритет тестирования | Высокий |  |
| Заголовок/название теста | Проверка ошибки в Email |  |
| Краткое изложение теста | Ввести в форму авторизации некорректные данные для поля Email(незарегистрированные) |  |
| Этапы теста | * 1. Зайти на главную страницу сайта;   2. Нажать левой кнопкой мыши по Кляксе в середине экрана   3. Внести ложные данные в поле Email   4. Внести данные в поле Password   5. Отправляем форму нажатием левой кнопки мыши по кнопке «Sign In» |  |
| Тестовые данные | Ложные данные для поля Email: [321232@yandex.com](mailto:321232@yandex.com)  Данные для поля Password: 4421 |  |
| Ожидаемый результат | Уведомление «email [321232@yandex.com](mailto:321232@yandex.com) has not been registered» |  |
| Фактический результат | Уведомление «email [321232@yandex.com](mailto:321232@yandex.com) has not been registered» |  |
| Статус | Выполнено |  |
| Предварительное условие | Запустить файл index.html из корневой папки проекта с помощью браузера Opera GX |  |
| Постусловие | После выполнения теста поле должно быть снова очищено для дальнейшего тестирования |  |

Таблица

|  |  |  |
| --- | --- | --- |
|  |  | Комментарии |
| Тест № | TC\_MT\_2 | Расшифровка: TestCase\_MoodTracker\_2 |
| Приоритет тестирования | Высокий |  |
| Заголовок/название теста | Проверка ошибки в Password |  |
| Краткое изложение теста | Ввести в форму авторизации некорректные данные для поля Email(незарегистрированные) |  |
| Этапы теста | 1. Зайти на главную страницу сайта; 2. Нажать левой кнопкой мыши по Кляксе в середине экрана 3. Внести данные в поле Email 4. Внести ложные данные в поле Password   Отправляем форму нажатием левой кнопки мыши по кнопке «Sign In» |  |
| Тестовые данные | Данные для поля Email: [karina.slake.koo@gmail.](mailto:karina.slake.koo@gmail.)com  Данные для поля Password: 3212 |  |
| Ожидаемый результат | Уведомление «Incorrect password for email karina.slake.koo@gmail.com» |  |
| Фактический результат | Уведомление «Incorrect password for email karina.slake.koo@gmail.com» |  |
| Статус | Выполнено |  |
| Предварительное условие | Запустить файл index.html из корневой папки проекта с помощью браузера Opera GX |  |
| Постусловие | После выполнения теста поле должно быть снова очищено для дальнейшего тестирования |  |

Таблица

|  |  |  |
| --- | --- | --- |
|  |  | Комментарии |
| Тест № | TC\_MT\_3 | Расшифровка: TestCase\_MoodTracker\_3 |
| Приоритет тестирования | Высокий | Сильно влияет на пользовательский опыт, а также не позволяет новым желающим зарегистрироваться в сервисе |
| Заголовок/название теста | Проверка работы кнопки «first time» из авторизации в регистрацию |  |
| Краткое изложение теста | Проверка работоспособности функции переворачивания карточки, если пользователь не зарегистрирован |  |
| Этапы теста | * 1. Зайти на главную страницу сайта;   2. Нажать левой кнопкой мыши по Кляксе в середине экрана;   3. Нажать левой кнопкой мыши по надписи «first time» под карточкой авторизации |  |
| Тестовые данные | - |  |
| Ожидаемый результат | Происходит анимация переворачивания карточки |  |
| Фактический результат | Происходит анимация переворачивания карточки |  |
| Статус | Выполнено |  |
| Предварительное условие | Запустить файл index.html из корневой папки проекта с помощью браузера Opera GX |  |
| Постусловие | Перезагрузить страницу для последующего тестирования |  |

Таблица

|  |  |  |
| --- | --- | --- |
|  |  | Комментарии |
| Тест № | TC\_MT\_4 | Расшифровка: TestCase\_MoodTracker\_4 |
| Приоритет тестирования | Средний |  |
| Заголовок/название теста | Проверка работы кнопки «first time» из регистрации в авторизацию |  |
| Краткое изложение теста | Проверка работоспособности функции переворачивания карточки, если пользователь только что зарегистрировался и хочет войти в сервис |  |
| Этапы теста | * + 1. Зайти на главную страницу сайта;     2. Нажать левой кнопкой мыши по Кляксе в середине экрана;     3. Нажать левой кнопкой мыши на надпись «first time»;     4. Ввести тестовые данные в поля регистрации;     5. Нажать левой кнопкой мыши по кнопке «Sign Up»;     6. Нажать левой кнопкой мыши по надписи «already registered» |  |
| Тестовые данные | Email: [karina.slake.koo@gmail.com](mailto:karina.slake.koo@gmail.com)  Password: 32122111  Confirm Password: 32122111 |  |
| Ожидаемый результат | Происходит анимация переворачивания карточки |  |
| Фактический результат | Происходит анимация переворачивания карточки |  |
| Статус | Выполнено |  |
| Предварительное условие | Запустить файл index.html из корневой папки проекта с помощью браузера Opera GX |  |
| Постусловие | Перезагрузить страницу для последующего тестирования |  |

Таблица

|  |  |  |
| --- | --- | --- |
|  |  | Комментарии |
| Тест № | TC\_MT\_5 | Расшифровка: TestCase\_MoodTracker\_5 |
| Приоритет тестирования | Высокий |  |
| Заголовок/название теста | Проверка формы регистрации |  |
| Краткое изложение теста | Проверка сохранения данных формы регистрации в локальном хранилище |  |
| Этапы теста | * + - 1. Зайти на главную страницу сайта;       2. Нажать левой кнопкой мыши по Кляксе в середине экрана;       3. Нажать левой кнопкой мыши на надпись «first time»;       4. Ввести тестовые данные в поля регистрации       5. Нажать левой кнопкой мыши по кнопке «Sign Up»;       6. Проверить локальное хранилище на наличие новых записей |  |
| Тестовые данные | Email: [moonl@yandex.ru](mailto:moonl@yandex.ru)  Password: m00nly123  Confirm password: m00nly123 |  |
| Ожидаемый результат | Наличие следующих записей в локальном хранилище:  newUserEmail: [moonl@yandex.ru](mailto:moonl@yandex.ru)  newUserPassword: m00nly123 |  |
| Фактический результат | Наличие следующих записей в локальном хранилище:  newUserEmail: [moonl@yandex.ru](mailto:moonl@yandex.ru)  newUserPassword: m00nly123 |  |
| Статус | Выполнено |  |
| Предварительное условие | Запустить файл index.html из корневой папки проекта с помощью браузера Opera GX |  |
| Постусловие | Перезагрузить страницу для последующего тестирования |  |

Таблица

|  |  |  |
| --- | --- | --- |
|  |  | Комментарии |
| Тест № | TC\_MT\_6 | Расшифровка: TestCase\_MoodTracker\_6 |
| Приоритет тестирования | Средний |  |
| Заголовок/название теста | Проверка формы регистрации на вывод ошибок |  |
| Краткое изложение теста | Проверка вывода ошибки при попытке зарегистрировать аккаунт с паролем короче 8 символов |  |
| Этапы теста | * + - * 1. Зайти на главную страницу сайта;         2. Нажать левой кнопкой мыши по Кляксе в середине экрана;         3. Нажать левой кнопкой мыши по надписи «first time»;         4. Ввести тестовые данные в поля регистрации;         5. Нажать левой кнопкой мыши по кнопке «Sign Up» |  |
| Тестовые данные | Email: [moonl@yandex.ru](mailto:moonl@yandex.ru)  Password: R213  Confirm password: R213 |  |
| Ожидаемый результат | Уведомление «Password is shorter than 8 characters» |  |
| Фактический результат | Уведомление «Password is shorter than 8 characters» |  |
| Статус | Выполнено |  |
| Предварительное условие | Запустить файл index.html из корневой папки проекта с помощью браузера Opera GX |  |
| Постусловие | - |  |

Таблица

|  |  |  |
| --- | --- | --- |
|  |  | Комментарии |
| Тест № | TC\_MT\_7 | Расшифровка: TestCase\_MoodTracker\_7 |
| Приоритет тестирования | Средний |  |
| Заголовок/название теста | Проверка формы регистрации на вывод ошибок |  |
| Краткое изложение теста | Проверка вывода ошибки при попытке зарегистрировать аккаунт вводя несовпадающие пароли |  |
| Этапы теста | * + - * 1. Зайти на главную страницу сайта;         2. Нажать левой кнопкой мыши по Кляксе в середине экрана;         3. Нажать левой кнопкой мыши по надписи «first time»;         4. Ввести тестовые данные в поля регистрации;         5. Нажать левой кнопкой мыши по кнопке «Sign Up» |  |
| Тестовые данные | Email: [moonl@yandex.ru](mailto:moonl@yandex.ru)  Password: R2134  Confirm password: R2136 |  |
| Ожидаемый результат | Уведомление «Passwords don't match» |  |
| Фактический результат | Уведомление «Passwords don't match» |  |
| Статус | Выполнено |  |
| Предварительное условие | Запустить файл index.html из корневой папки проекта с помощью браузера Opera GX |  |
| Постусловие | - |  |

# Заключение

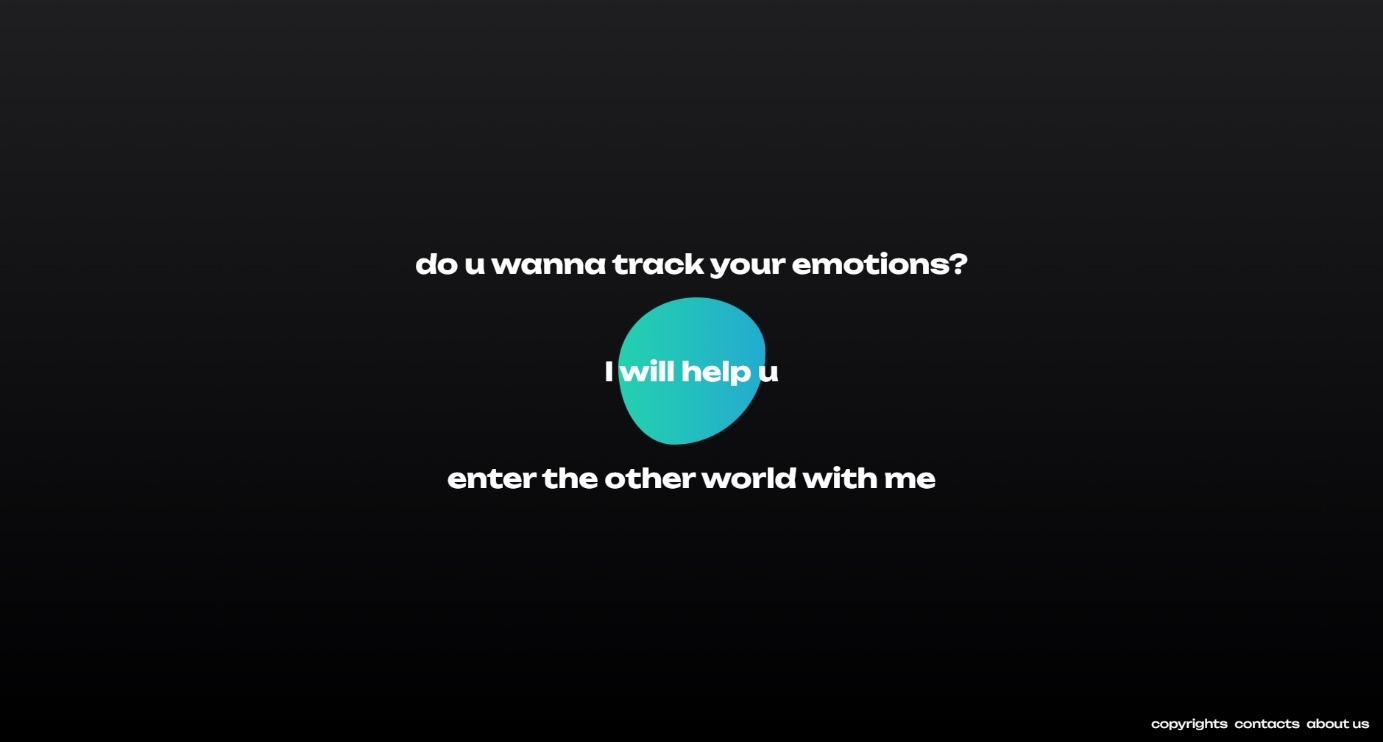
В рамках этого проекта было разработано полнофункциональное веб-приложение для отслеживания настроения и календаря настроения, которое отвечало всем требованиям. Приложение позволяет пользователям в удобном формате просматривать информацию об эмоциях и чувствах, которые они испытывали в прошлом, и читать заметки, связанные с этими эмоциями.

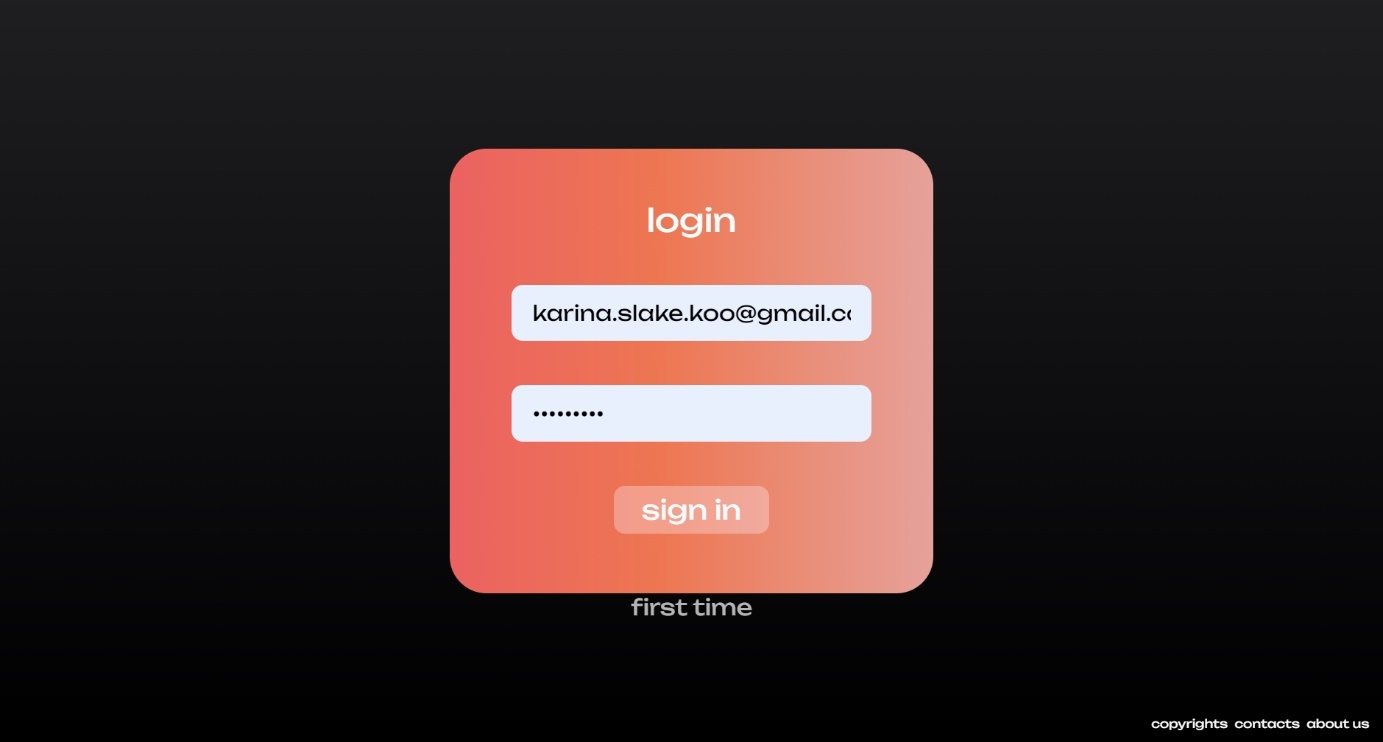
Планируется дальнейшее развитие и совершенствование приложения.

# Список литературы

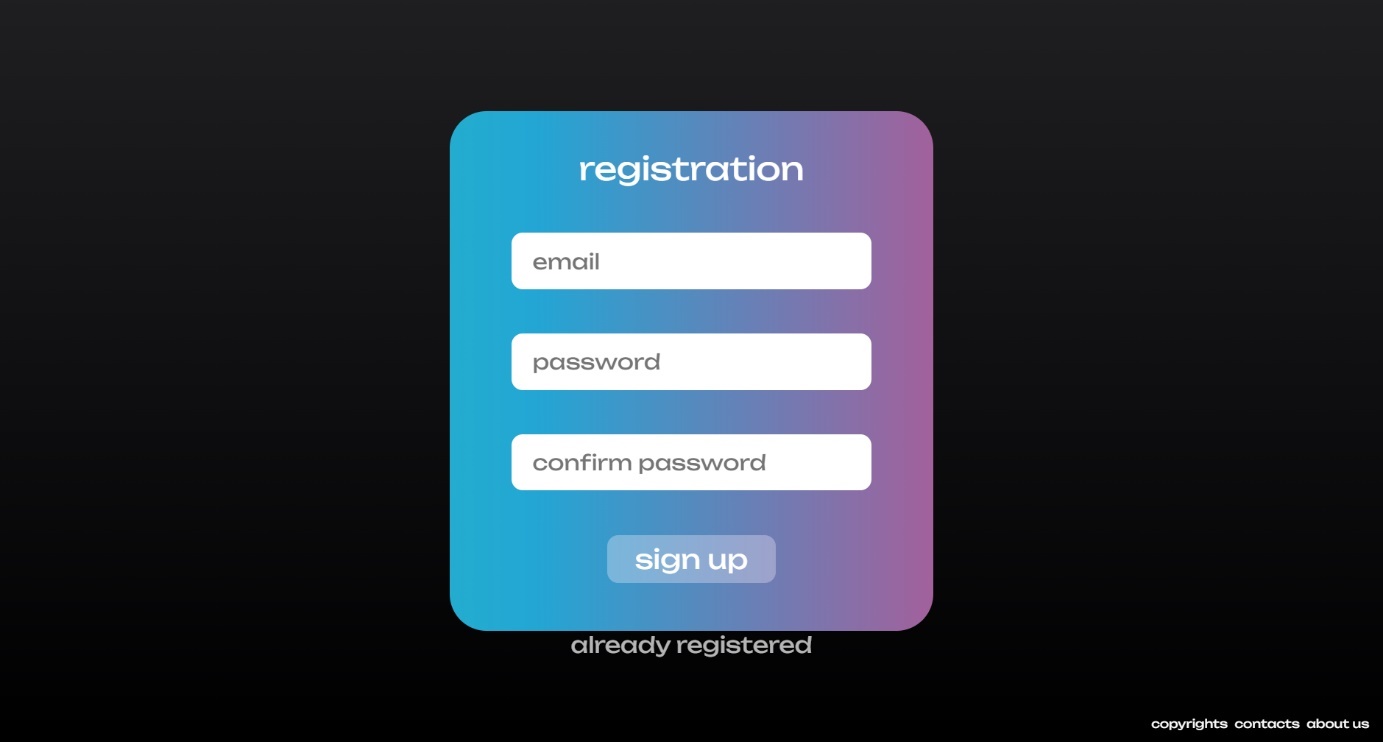
1. Документация HTML: <https://developer.mozilla.org/ru/docs/Web/HTML>
2. Руководство по JavaScript: <https://developer.mozilla.org/ru/docs/Web/JavaScript>
3. Список признанных психологами чувств и эмоций: <https://www.b17.ru/blog/feelings_and_emotions/>
4. Словарь настроений: https://docs.google.com/document/d/1L9fwMbrRMNneR3sgElCfcMA1Gs4IWxwdu2MMhLfPXT4/mobilebasic

# Приложение А

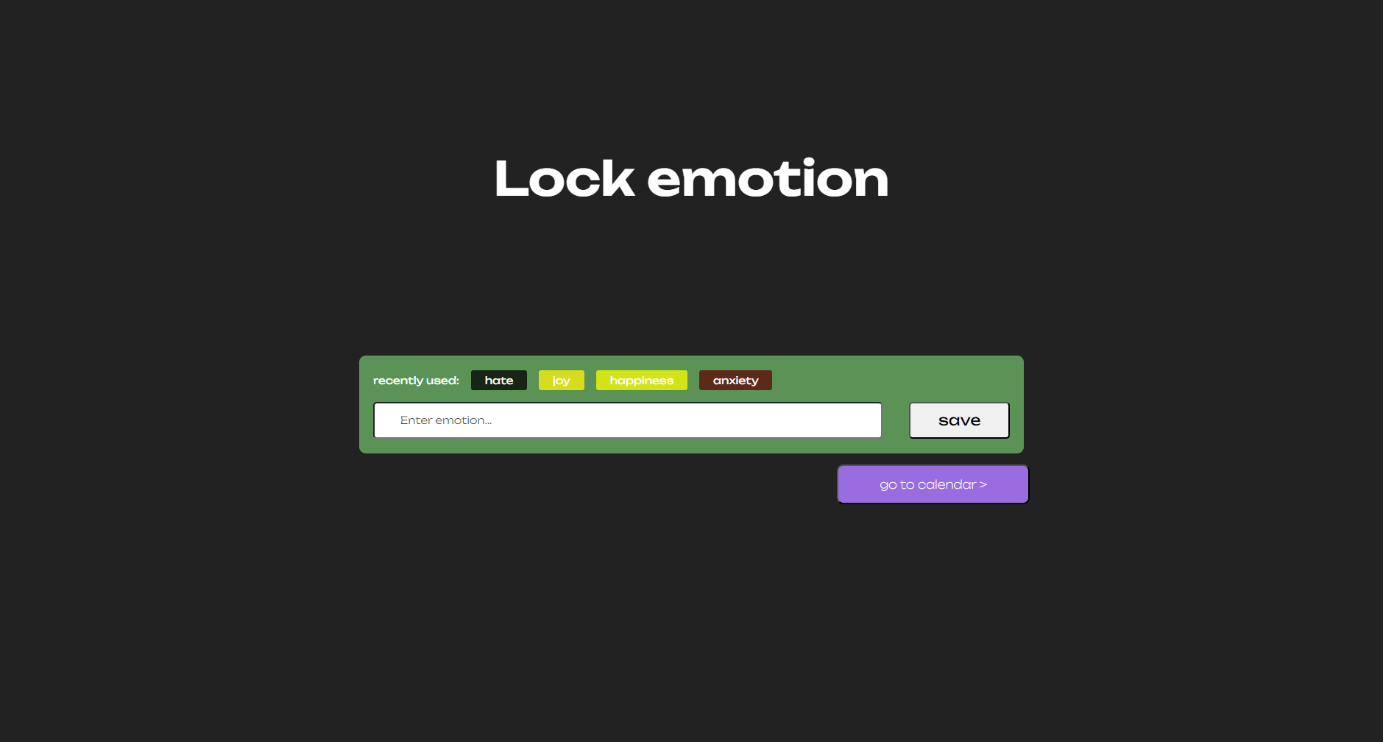
Главная страница



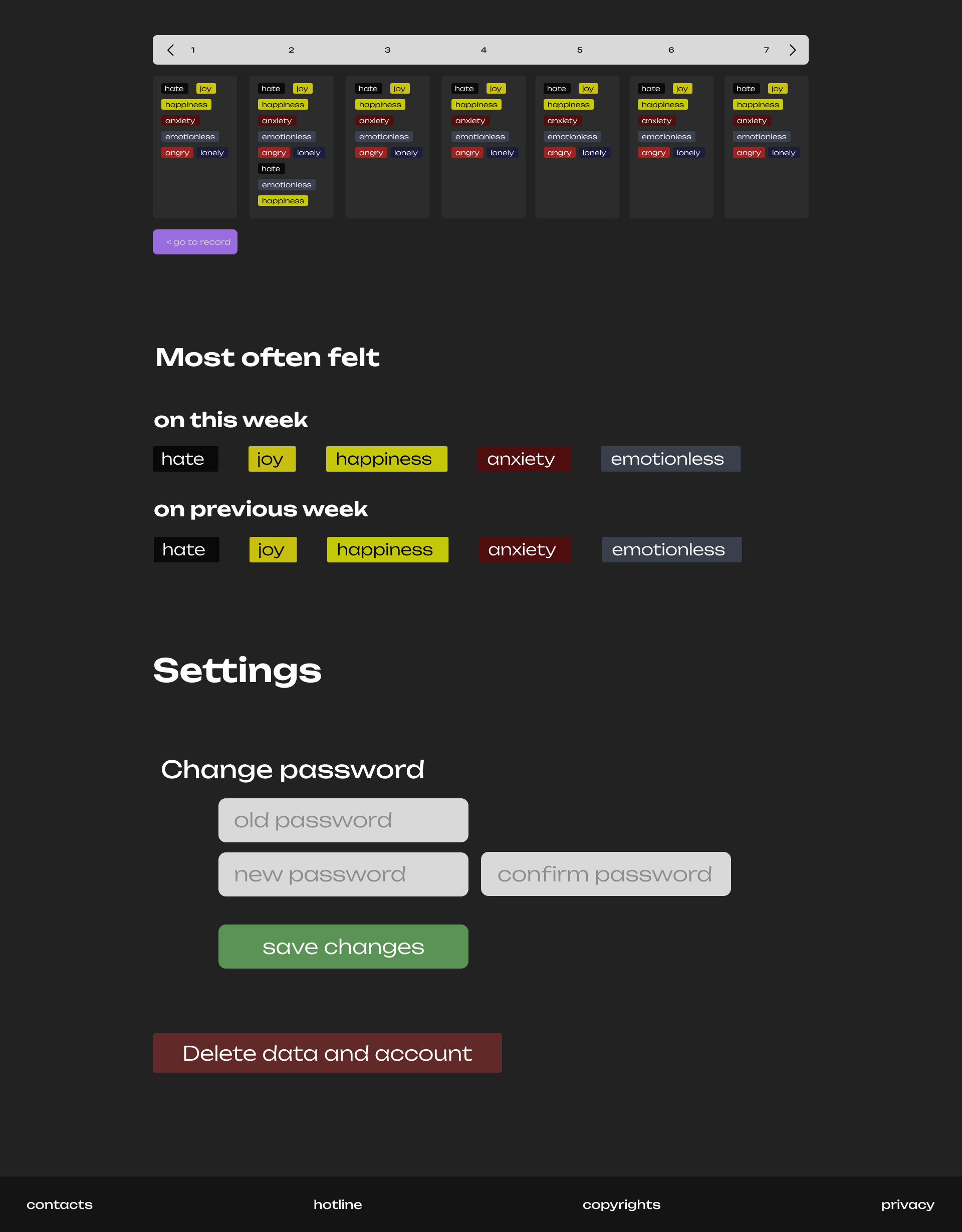
Главная страница



Главная страница



Страница внесения записей



Страница календаря

# Приложение Б

Тестируемый код в п. 2.2:

let blob = document.getElementById("blob");

let textsWelcome = document.querySelectorAll(".textOnWelcome");

let signUpForm = document.querySelector("#main");

let card = document.getElementById('card');

let swapCard = document.getElementById('formToSignUp');

blob.addEventListener('click', () => {

textsWelcome.forEach(element => {

element.classList.add('fade-out')

});

signUpForm.setAttribute('style', 'z-index: 2');

blob.setAttribute('style', 'width: 35vw; height: 60vh; animation: none; border-radius: 50px;');

blob.classList.add('no-pseudo');

let active = document.querySelectorAll('.fade-in');

active.forEach((el) => {

el.classList.add('active')

});

});

swapCard.addEventListener('click', () => {

if(card.classList.contains('container') !== false){

card.classList.remove('container');

swapCard.innerText = 'first time';

swapCard.classList.remove('backTitle');

swapCard.setAttribute('style', 'transform: rotateY(0)');

blob.setAttribute('style', 'width: 35vw; height: 60vh; animation: none; border-radius: 50px;');

} else {

swapCard.innerText = 'already registered';

swapCard.setAttribute('style', 'transform: rotateY(180deg)');

card.classList.add('container');

swapCard.classList.add('backTitle');

blob.setAttribute('style', 'width: 35vw; height: 70vh; animation: none; border-radius: 50px;');

};

})

let btn\_reg = document.getElementById("btn\_reg");

function registerUser(){

email = document.getElementById('signUpEmail').value;

password = document.getElementById('signUpPassword').value;

confPassword = document.getElementById('signUpConfirmPassword').value;

if(password == confPassword && password.length >= 7) {

localStorage.setItem('newUserEmail', email);

localStorage.setItem('newUserPassword', password);

}

else if(password != confPassword) alert(`Passwords don't match`);

else if(password.length <= 7) alert(`Password is shorter than 8 characters`)

}

let btn\_log = document.getElementById("btn\_log");

function loginUser(){

email = document.getElementById('signInEmail').value;

password = document.getElementById('signInPassword').value;

const registeredUser = localStorage.getItem("newUserEmail");

const registeredPassword = localStorage.getItem("newUserPassword");

const validUser = email == registeredUser;

const validPassword = password == registeredPassword;

if(validUser && validPassword){

console.log('u did it');

document.location.replace('pages/recEmotion.html');

}

else if(!validUser) alert(`email ${email} has not been registered. `);

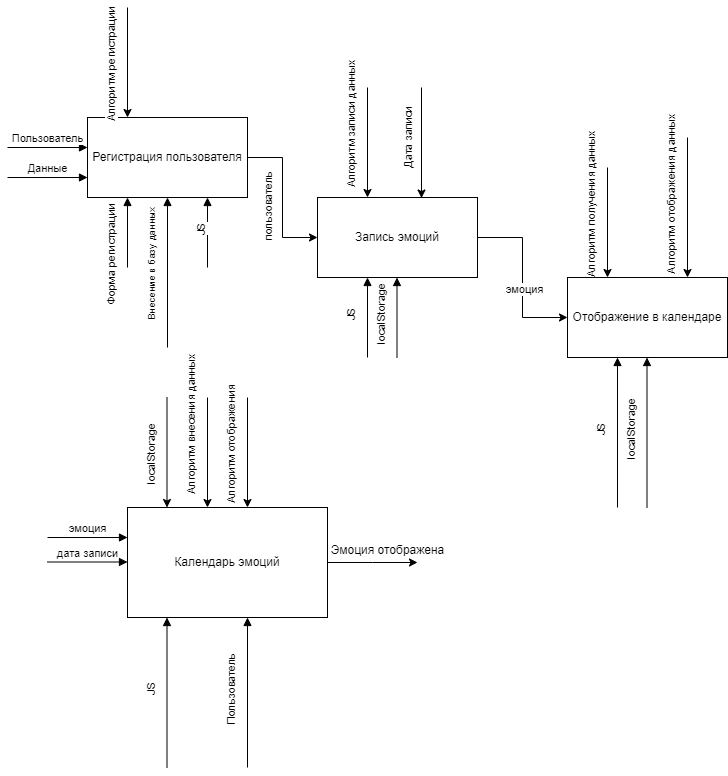
else if(!validPassword) alert(`Incorrect password for email ${email}`);

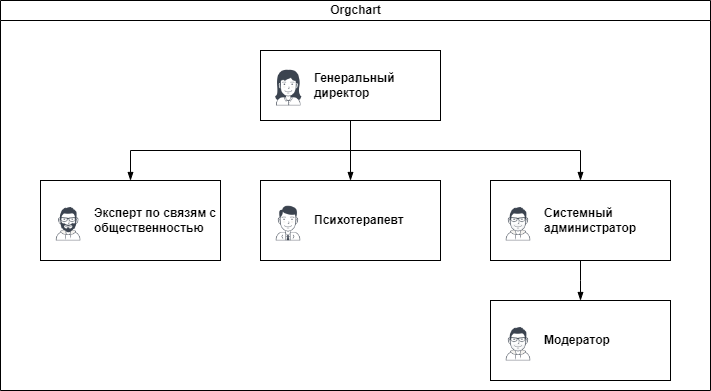
}

btn\_log.addEventListener('click', () => loginUser());

btn\_reg.addEventListener('click', () => registerUser());

# Приложение В

ДиаграммыIDEF0 и декомпозиция

Организационная диаграмма

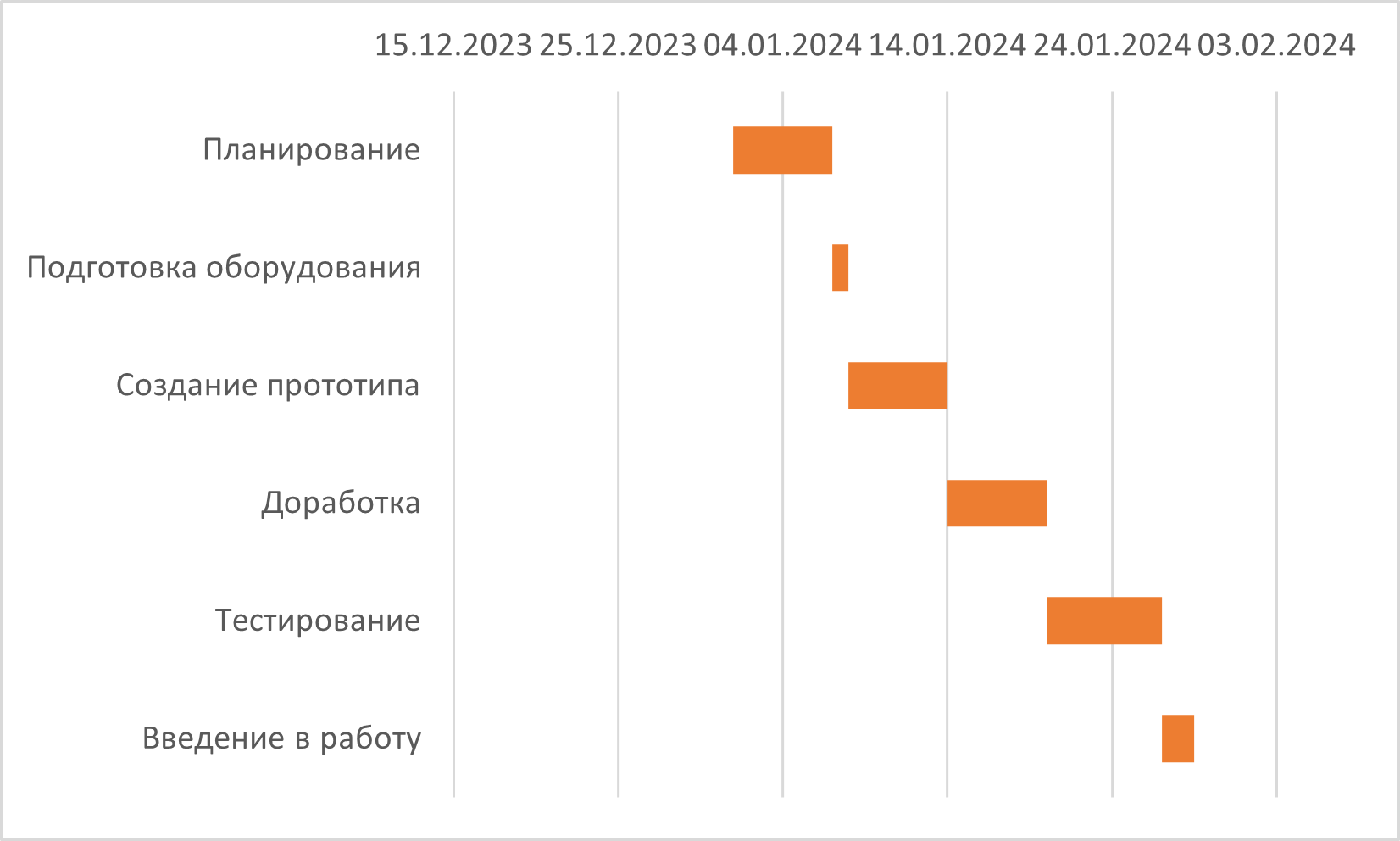


Диаграмма Ганта