ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет Информационных технологий
Кафедра Информатики и информационных технологий

направление подготовки
09.03.02 «Информационные системы и технологии»

КУРСОВОЙ ПРОЕКТ

Тема: Календарь событий. Веб-приложение для создания и управления мероприятиями, с возможностью делиться событиями с друзьями.
Дисциплина: Технологии прикладного программирования

Выполнила: студентка группы 241-335
Варивода М.Ю.
(Фамилия И.О.)

Дата, подпись _____ _____
(Дата) (Подпись)

Проверил(-а): _____

Дата, подпись _____ _____
(Дата) (Подпись)

Замечания:

_____

_____

Москва
2025

Оглавление

Введение

Постановка задачи

Веб-приложение "Календарь событий" разработано для удобного планирования личных и групповых мероприятий. Было необходимо создать интуитивно понятный инструмент с функционалом добавления, редактирования и поиска событий, а также возможностью условного обмена информацией с друзьями. Приложение ориентировано на пользователей, которым важно организовать свое время без использования сложных систем.
Требовалось разработать клиентское веб-приложение с использованием чистого JavaScript, HTML и CSS, без привлечения сторонних фреймворков. Основной упор делался на работу с DOM и локальное хранение данных через LocalStorage. Ключевые требования включали: реализацию интерактивного календаря с возможностью переключения месяцев и выбора дат, добавление событий с указанием названия, даты, времени, категории и описания, редактирование и удаление существующих событий, фильтрацию по категориям (работа, личное, семья, друзья), поиск по названию и описанию, имитацию авторизации пользователя с сохранением статуса в LocalStorage, а также демонстрационную функцию добавления друзей и совместного использования событий без реальной серверной части.

Цели и задачи проекта

Основная цель проекта заключается в создании полнофункционального веб-приложения "Календарь событий", которое позволит пользователям эффективно организовывать и управлять своими личными и групповыми мероприятиями. Приложение должно обеспечивать удобное взаимодействие с событиями через интуитивно понятный интерфейс, предоставляя возможности для планирования, категоризации и поиска мероприятий. Разрабатываемый продукт предназначен для широкого круга пользователей, которым необходимо систематизировать свое время и координировать планы с другими людьми.
Для достижения поставленной цели были определены следующие ключевые задачи, охватывающие все аспекты разработки:
Разработка клиентской части на чистом JavaScript
Необходимо создать одностраничное приложение, которое будет работать без перезагрузки страницы. Основной акцент делается на использование нативного JavaScript вместо фреймворков для полного контроля над DOM-элементами и оптимизации производительности.
Реализация интерактивного календаря
Требуется разработать систему отображения месяцев с возможностью переключения между ними, выделением текущей и выбранной даты, а также визуальным отображением дней, содержащих события. Календарь должен динамически перестраиваться при изменении периода.
Организация системы хранения данных
Для сохранения информации о событиях и пользователях необходимо реализовать работу с LocalStorage. Это включает создание структуры данных, методов сохранения, загрузки и обновления информации без использования серверной части.
Создание функционала управления событиями
Приложение должно предоставлять возможности:

- Добавления новых событий с указанием названия, даты, времени, категории и описания
- Редактирования существующих мероприятий

- Удаления запланированных событий
- Быстрого просмотра событий на выбранную дату

Реализация системы фильтрации и поиска
Необходимо разработать механизмы:

- Поиска мероприятий по названию и описанию
- Группировки событий по датам при отображении результатов фильтрации

Разработка системы авторизации
Требуется создать имитацию процесса входа и регистрации пользователей с сохранением статуса авторизации в LocalStorage. Система должна ограничивать функциональность для неавторизованных пользователей.

Создание демонстрационного функционала для работы с друзьями
Необходимо реализовать интерфейс для условного добавления друзей и обозначения событий как доступных для других пользователей, без разработки реальной серверной части для обмена данными.

Обеспечение адаптивности интерфейса
Интерфейс должен корректно отображаться на устройствах с разными размерами экранов, сохраняя удобство использования как на компьютерах, так и на мобильных устройствах.

Тестирование и отладка приложения
После реализации всех функций требуется провести комплексное тестирование:

- Проверку работы календаря и системы событий
- Тестирование фильтрации и поиска
- Проверку корректности работы авторизации
- Тестирование адаптивности интерфейса
- Проверку сохранения данных при перезагрузке страницы

Документирование кода и подготовка отчета
Необходимо обеспечить читаемость кода через комментарии и соблюдение стиля написания, а также подготовить подробную документацию по функционалу приложения в рамках курсовой работы.

Реализация этих задач позволит создать полноценное клиентское веб-приложение для управления событиями, демонстрирующее навыки работы с DOM, JavaScript и клиентским хранением данных, а также заложит основу для возможного дальнейшего развития проекта с добавлением серверной части и расширенного функционала.

Актуальность темы

В современном мире, характеризующемся высоким темпом жизни и постоянным увеличением объема информации, проблема эффективного управления временем становится особенно острой. Разработка цифровых инструментов для организации личного и рабочего времени представляет собой важную задачу, имеющую значительную практическую ценность. Актуальность создания веб-приложения "Календарь событий" обусловлена несколькими ключевыми факторами.

Во-первых, наблюдается устойчивый рост спроса на цифровые решения для планирования времени. По данным исследований, более 60% пользователей интернета

регулярно используют различные приложения для организации своего расписания. При этом существует явный запрос на простые и интуитивно понятные инструменты, которые не перегружены избыточным функционалом, характерным для многих профессиональных систем управления проектами. Разрабатываемое приложение отвечает этой потребности, предлагая базовый, но достаточный для большинства пользователей набор функций в удобной форме. Во-вторых, современные тенденции цифровизации всех сфер жизни обуславливают необходимость переноса традиционных бумажных органайзеров и планировщиков в цифровую среду. Цифровые календари обладают неоспоримыми преимуществами: они доступны с различных устройств, позволяют быстро вносить изменения, устанавливать напоминания и автоматически упорядочивать события. Однако многие существующие решения либо слишком сложны для повседневного использования, либо требуют постоянного подключения к интернету. Предлагаемое приложение решает эту проблему, сочетая простоту использования с возможностью работы в оффлайн-режиме благодаря хранению данных в LocalStorage.

В-третьих, особую актуальность приобретает вопрос синхронизации личных и рабочих графиков. По статистике, около 40% работающих людей испытывают сложности с балансом между профессиональной и личной жизнью. Календарь событий с системой категорий (работа, личное, семья, друзья) позволяет визуализировать это распределение и способствует более осознанному планированию времени. При этом реализованная система фильтрации дает возможность сосредоточиться на событиях определенного типа, что особенно важно для пользователей с насыщенным графиком.
С технической точки зрения, проект демонстрирует актуальные подходы к веб-разработке. Использование чистого JavaScript вместо фреймворков соответствует современным тенденциям оптимизации веб-приложений и уменьшения их зависимости от сторонних библиотек. Это особенно важно в условиях, когда требования к производительности веб-приложений постоянно растут, а внимание к скорости загрузки и отзывчивости интерфейса становится критически важным фактором пользовательского опыта.

Разработка полнофункционального веб-приложения охватывает широкий спектр современных веб-технологий и методик программирования:

- Работу с DOM и событиями
- Клиентское хранение данных
- Динамическую генерацию контента
- Асинхронные операции
- Валидацию данных
- Адаптивный дизайн

При этом проект имеет хороший потенциал для дальнейшего развития и масштабирования. В перспективе возможно добавление серверной части, реализация синхронизации между устройствами, внедрение системы уведомлений и других востребованных функций. Это делает выбранную тему не только актуальной на текущий момент, но и перспективной с точки зрения возможного продолжения работы.

Экономическая целесообразность разработки подобного приложения подтверждается растущим рынком productivity-приложений. По прогнозам аналитиков, мировой рынок программного обеспечения для повышения продуктивности к 2025 году достигнет 96 миллиардов долларов. При этом существует постоянный спрос на новые решения, предлагающие оригинальные подходы к организации времени. Даже в условиях высокой

конкуренции, простое и удобное приложение с продуманным пользовательским интерфейсом может найти свою нишу.
Таким образом, разработка веб-приложения "Календарь событий" представляет собой актуальную задачу, имеющую значение как с практической точки зрения (создание полезного инструмента для пользователей), так и с образовательной (демонстрация комплексных навыков веб-разработки). Проект сочетает в себе востребованность на рынке, соответствие современным технологическим тенденциям и потенциал для дальнейшего развития, что делает его отличной темой для курсовой работы по программированию.

Глава 1. Аналитическая часть

1.1. Анализ предметной области

Предметная область проекта охватывает сферу цифровых инструментов для управления временем и организации событий, включая персональное и групповое планирование, визуализацию расписания и координацию совместной деятельности. В современном мире с его высоким темпом жизни и постоянным увеличением количества задач подобные системы становятся критически важными для поддержания личной и профессиональной эффективности. Рынок предлагает множество решений - от простых мобильных приложений до сложных корпоративных систем, однако сохраняется потребность в удобных и специализированных инструментах, учитывающих специфические потребности различных групп пользователей. Основными сущностями в системе являются пользователи, события и категории. Пользователь представляет собой учетную запись с уникальными идентификаторами и аутентификационными данными. Событие - это базовый элемент планирования, содержащий информацию о времени, месте, участниках и других параметрах. Категории служат для классификации событий по тематике или важности. Дополнительными сущностями выступают уведомления (напоминания о событиях), шаблоны повторяющихся мероприятий и настройки отображения календаря. Все сущности связаны между собой отношениями "один-ко-многим" и "многие-ко-многим", что требует тщательного проектирования структуры базы данных. Система должна обеспечивать надежное хранение данных, мгновенную синхронизацию между устройствами и защиту конфиденциальной информации. Критически важными являются требования к производительности при работе с большими объемами событий и отказоустойчивости при сбоях соединения. Основные технологические ограничения связаны с необходимостью поддержки различных платформ и браузеров, включая мобильные устройства. Особую сложность представляет реализация оффлайн-режима с последующей синхронизацией данных при восстановлении соединения.
Дальнейшее развитие системы может идти по нескольким направлениям: внедрение искусственного интеллекта для анализа расписания и автоматического планирования, интеграция с IoT-устройствами для создания "умного" окружения, разработка инструментов аналитики продуктивности. С точки зрения коммерциализации перспективными представляются модели freemium с платными премиум-функциями, корпоративные подписки для организаций, а также партнерские интеграции с другими сервисами. Особый потенциал имеет разработка white-label решений для образовательных учреждений и бизнес-структур.

Проведенный анализ показывает, что разработка календарного приложения требует тщательного учета множества факторов: от понимания потребностей целевой аудитории до грамотного технического проектирования. Ключевыми конкурентными преимуществами могут стать специализация на конкретных пользовательских сценариях, интуитивно понятный интерфейс и надежная работа в условиях нестабильного интернет-соединения. Особое внимание следует уделить механизмам синхронизации данных и разрешения конфликтов, так как именно эти аспекты чаще всего вызывают проблемы у пользователей. Дальнейшая

разработка должна вестись с учетом возможностей масштабирования системы и добавления новых функций без нарушения работы существующего функционала.

1.2. Обзор существующих решений

Современный рынок предлагает множество решений для управления событиями и планирования времени, каждое из которых имеет свои особенности и целевую аудиторию. Наиболее распространенными являются корпоративные календари, такие как Microsoft Outlook Calendar и Google Calendar, которые интегрированы с офисными пакетами и почтовыми сервисами. Эти решения обеспечивают синхронизацию между устройствами, поддержку группового планирования и напоминания, но часто перегружены функциями, что делает их неудобными для личного использования. Персональные планировщики типа Fantastical и Any.do предлагают более простой интерфейс и специализированные функции, такие как обработка естественного языка для ввода событий, но имеют ограниченные возможности для командной работы. Специализированные инструменты вроде Calendly ориентированы на конкретные сценарии, например, бронирование встреч, и предлагают глубокую интеграцию с CRM-системами и другими бизнес-инструментами. Отдельную нишу занимают дизайн-ориентированные решения, такие как Timepage, которые делают акцент на визуальном представлении данных и пользовательском опыте.

При детальном сравнении функциональности выделяются ключевые различия между популярными решениями. Google Calendar и Outlook Calendar обеспечивают полную синхронизацию через облачные сервисы и поддерживают совместную работу, но требуют постоянного подключения к интернету. Fantastical, ориентированный на экосистему Apple, предлагает более продвинутые функции управления повторяющимися событиями и интеграцию с задачами, но ограничен в кросс-платформенности. Calendly, как узкоспециализированный инструмент, идеально подходит для организации встреч, но не заменяет полноценный календарь. Общей проблемой для большинства решений остается слабая поддержка оффлайн-режима и ограниченные возможности кастомизации интерфейса.

Анализ существующих продуктов выявляет несколько перспективных направлений для разработки новых решений. Во-первых, это создание гибридных систем, сочетающих простоту персональных планировщиков с функциональностью корпоративных календарей. Во-вторых, развитие технологий офлайн-доступа с последующей синхронизацией данных. В-третьих, улучшение интеграции между календарями и системами управления задачами. Особый потенциал имеют решения, ориентированные на конкретные профессиональные группы (например, врачи, преподаватели или фрилансеры), с учетом их специфических потребностей в планировании. Также перспективным направлением является развитие технологий искусственного интеллекта для автоматического анализа расписания и оптимизации времени.

Глава 2. Технологическая часть

2.1. Выбор технологий

Для разработки веб-приложения "Календарь событий" были выбраны современные технологии, обеспечивающие высокую производительность, надежность и удобство работы как для разработчиков, так и для пользователей. Выбор технологий основывался на требованиях проекта, таких как необходимость создания интерактивного интерфейса, управления событиями, обеспечения безопасности и кроссбраузерной совместимости. Ниже представлено описание выбранных технологий и обоснование их использования:

- HTML5 — использовался для создания структуры веб-страниц. HTML5

обеспечивает семантическую разметку, что важно для доступности и SEO-оптимизации приложения. Он также поддерживает современные элементы, такие как `<input type="date">`, которые упрощают работу с датами в календаре [3].

- CSS3 — применялся для стилизации интерфейса. CSS3 позволил реализовать адаптивный дизайн, анимации и сложные визуальные эффекты, такие как тени, градиенты и переходы. Использование Flexbox и Grid упростило создание макетов, которые корректно отображаются на устройствах с разными размерами экранов [3].
- JavaScript (ES6+) — был выбран для реализации интерактивных элементов, таких как динамическое отображение событий, модальные окна и обработка пользовательских действий. JavaScript обеспечил работу календаря, фильтрацию событий и взаимодействие с локальным хранилищем данных [3].
- LocalStorage — использовался для хранения данных о событиях и статусе авторизации пользователей. Это позволило реализовать автономную работу приложения без необходимости подключения к серверу, что упрощает использование календаря в условиях отсутствия интернета [1].
- Font Awesome — библиотека иконок, которая была интегрирована для улучшения визуального восприятия интерфейса. Иконки обеспечивают интуитивно понятную навигацию и делают интерфейс более современным и удобным для пользователей [4].
- Google Fonts (Segoe UI) — использовался для подключения современных и читаемых шрифтов, что улучшило пользовательский опыт и визуальную привлекательность приложения.
- Адаптивный дизайн — реализован с помощью медиа-запросов CSS, что обеспечивает корректное отображение приложения на различных устройствах, включая мобильные телефоны, планшеты и десктопы [3].
- Модульная структура кода — JavaScript-код был организован в виде модулей, что упростило поддержку и масштабируемость приложения. Это также

позволило разделить логику работы календаря, событий и авторизации.

Выбранные технологии позволяют создать полнофункциональное веб-приложение с интерактивным интерфейсом, которое работает без необходимости подключения к серверу. Использование локального хранилища (LocalStorage) обеспечивает сохранность данных между сеансами, а современные инструменты HTML5, CSS3 и JavaScript делают приложение отзывчивым и удобным для пользователей. Интеграция иконок Font Awesome и качественных шрифтов улучшает визуальную составляющую, что важно для пользовательского опыта. Эти технологии были выбраны из-за их широкой поддержки, простоты использования и возможности создания эффективного решения, отвечающего всем требованиям проекта.

2.2. Структура проекта

1. static/ - содержит все файлы фронтенда:
index.html - главная страница с календарем событий
friends.html - страница управления друзьями
styles.css - общие стили для всего приложения
script.js - основной JavaScript для календаря
friends.js - JavaScript для страницы друзей

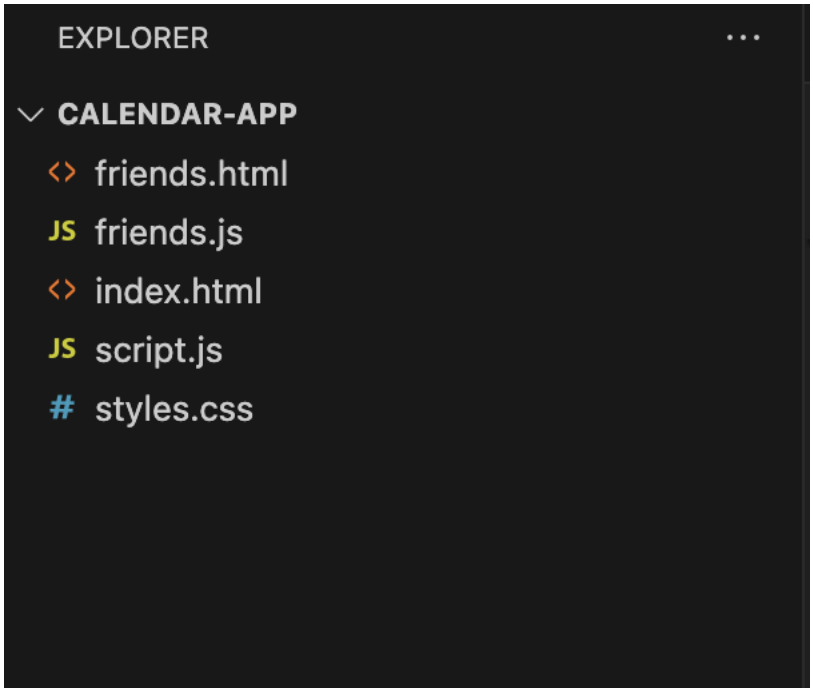

Рисунок 2.1. Структура проекта

2.3. Описание модулей

Модуль `**static**` содержит фронтенд-часть приложения "Календарь событий" и включает следующие файлы:
1. index.html: Главная страница приложения с календарем событий, формами для входа и регистрации, а также возможностью добавления, редактирования и удаления событий. После успешной авторизации пользователь получает доступ к своему календарю. Код представлен в приложении (см. приложение №1).

2. friends.html: Страница управления списком друзей, где пользователь может добавлять, искать и удалять друзей, а также видеть их статус (online/offline). Код представлен в приложении (см. приложение №2).
3. styles.css: Общие стили для всех страниц, обеспечивающие единообразный и адаптивный дизайн. Включает стили для календаря, модальных окон, карточек событий и друзей. Код представлен в приложении (см. приложение №3).
4. script.js: Основной скрипт для работы с календарем, включая рендеринг событий, обработку форм, управление модальными окнами и взаимодействие с `localStorage`. Код представлен в приложении (см. приложение №4).
5. friends.js: Скрипт для управления списком друзей на странице `friends.html`, включая поиск, добавление и удаление друзей. Код представлен в приложении (см. приложение №5).
Модуль обеспечивает:

- Авторизацию и регистрацию пользователей.
- Визуализацию календаря с событиями по категориям.
- Добавление, редактирование и удаление событий.
- Возможность делиться событиями с друзьями.
- Управление списком друзей и их статусами.

Все страницы связаны между собой, а данные сохраняются в `localStorage` для обеспечения работы без backend-части.

2.4 Тестирование

Для выполнения практической части была выбрана среда разработки Visual Studio Code, так как идеально для этого подходит, является бесплатной, поддерживает js.

Для запуска используем LiveServer, переходим по адресу *http://127.0.0.1:5500* и перед нами главная страница (рис. 2.1), однако создавать и просматривать заметки можно только после прохождения аутентификации (рис. 2.2).

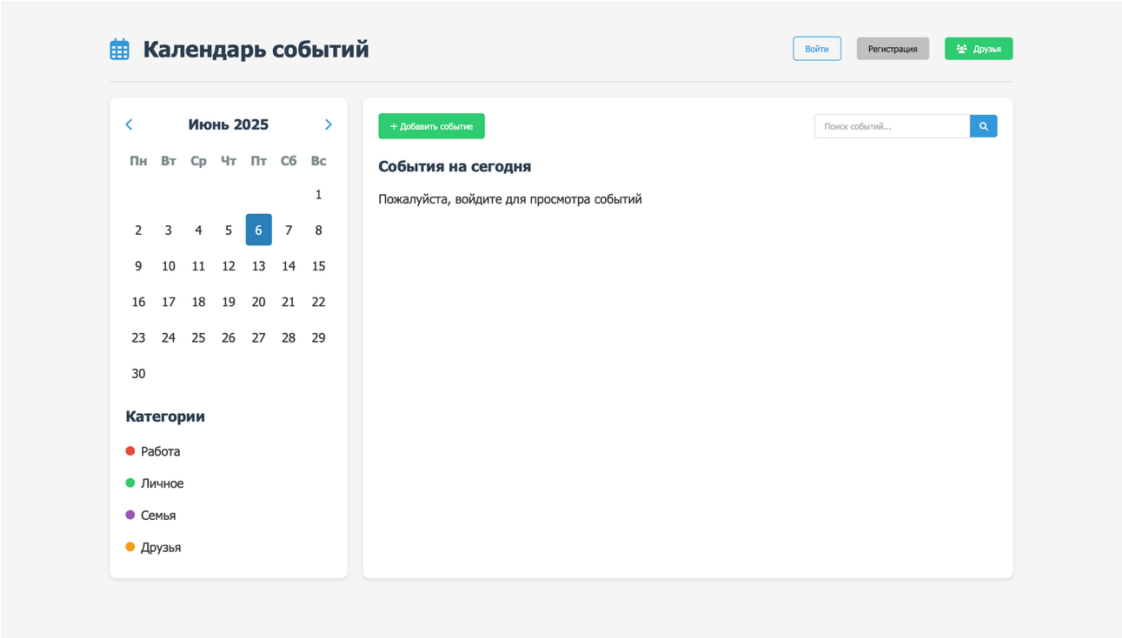

Рисунок 2.1. Главная страница

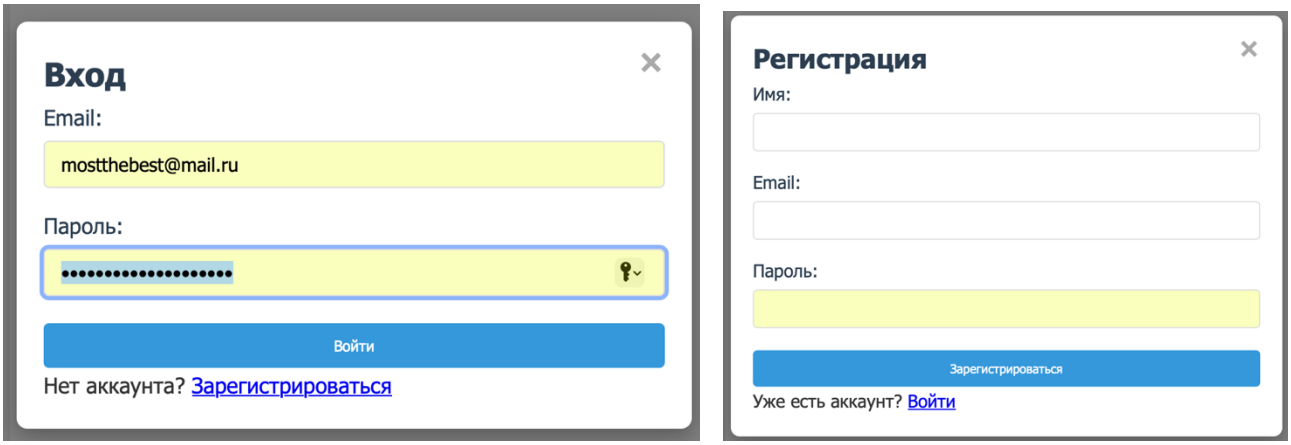

Рисунок 2.2. Авторизация

После авторизации появляется возможность увидеть сохраненные события, добавить новые и перейти на вкладку «Друзья» (рис. 2.2).

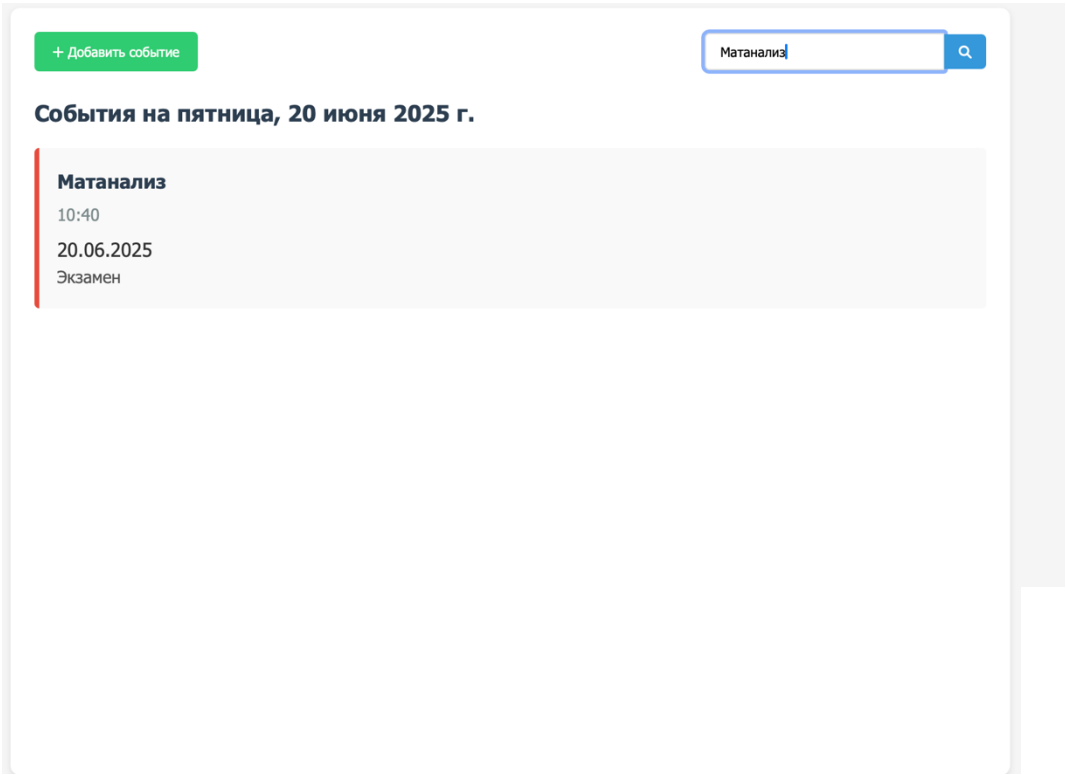

Рисунок 2.3. Главная страница зарегистрированного пользователя

После нажатия кнопки “Добавить событие ” появляется такое окно (рис. 2.4) с возможностью указать название, дату, время события, также установить категорию, описание и поделиться с друзьями.

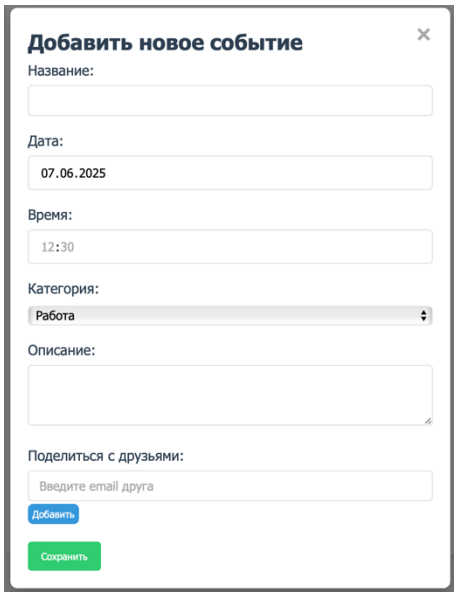

Рисунок 2.4. Панель для добавления события

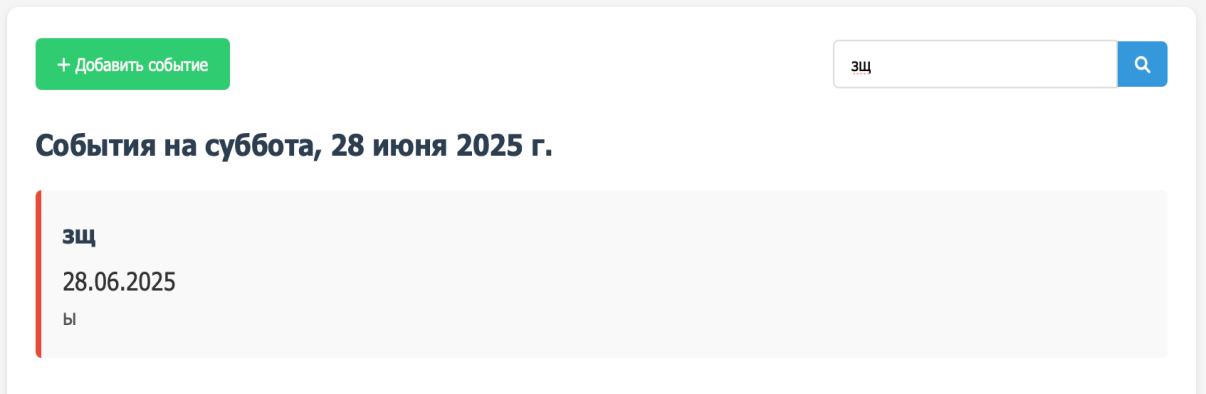

Рисунок 2.5. Демонстрация поиска события по названию

При нажатии кнопки «Друзья» открывается страница с моими друзьями, а также возможностью поиска новых друзей (рис 2.6).

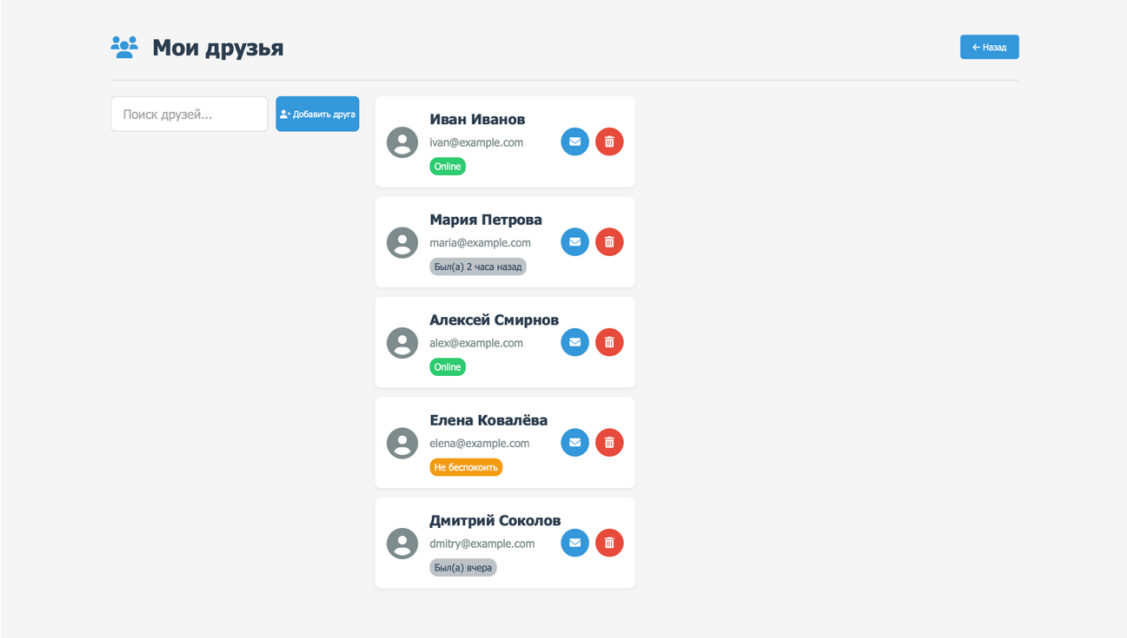

Рисунок 2.6. Страница с друзьями

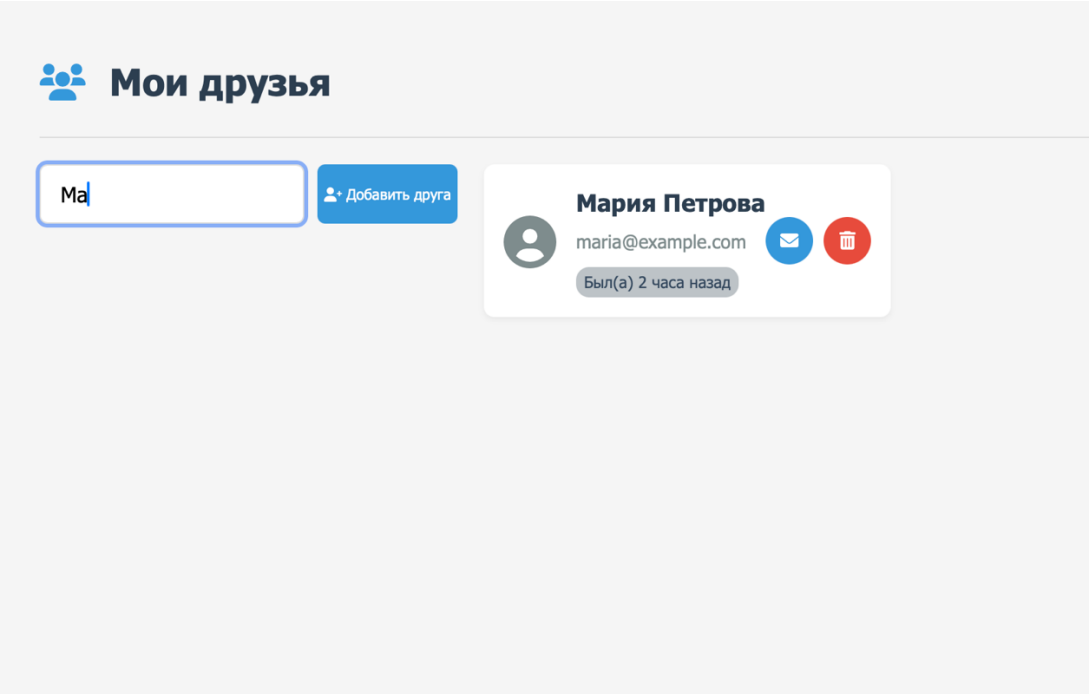

Рисунок 2.7. Демонстрация поиска друзей

При нажатии «Добавить друга» открывается окно (рис.2.8)

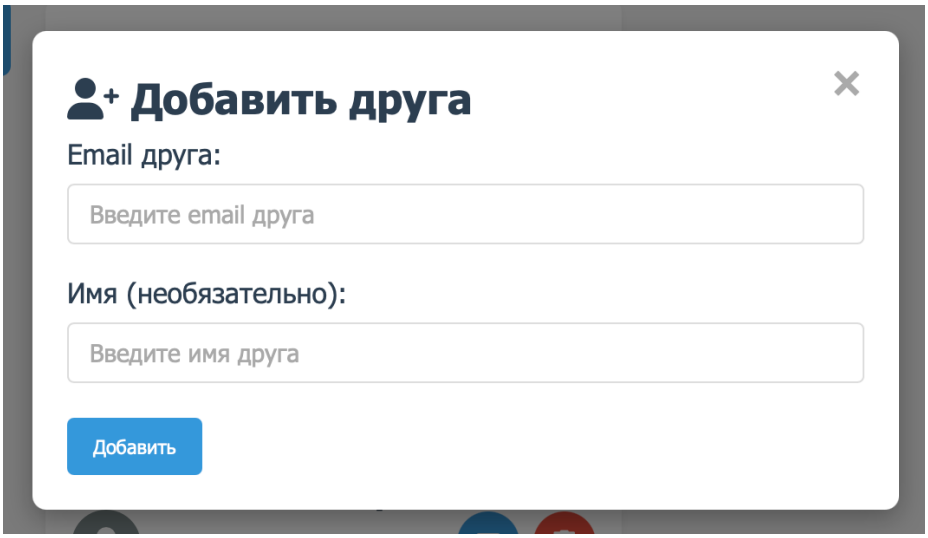

Рисунок 2.8. Демонстрация возможности добавить нового друга

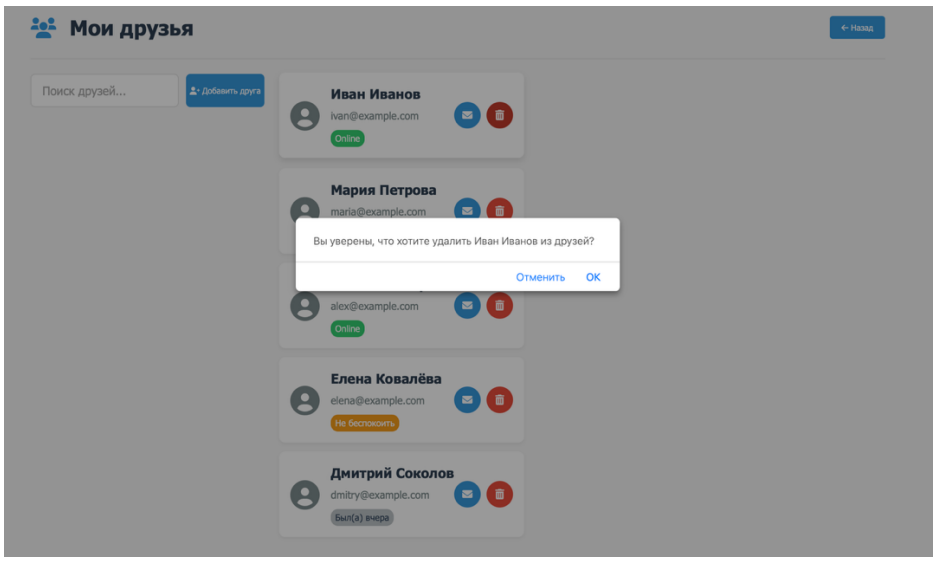

Рисунок 2.9. Демонстрация возможности удалить друга

Список литературы

1. MDN Web Docs – официальная документация по HTML, CSS и JavaScript.
   URL: https://developer.mozilla.org/ru/

2. LocalStorage API – документация по работе с Web Storage.
   URL: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage

3. Font Awesome – документация по использованию иконок.
   URL: https://fontawesome.com

4. Google Fonts – библиотека шрифтов.
   URL: https://fonts.google.com

5. Адаптивный дизайн – руководство по медиазапросам и Flexbox/Grid.
   URL: https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout/Responsive_Design

6. Современный JavaScript (ES6+) – руководство по новым возможностям JavaScript.
   URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript

7. Веб-приложения без серверной части – статьи о клиентских технологиях.
   URL: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

8. Кроссбраузерная совместимость – рекомендации по поддержке разных браузеров.
   URL: https://caniuse.com

9. Документация по Visual Studio Code – руководство по использованию среды разработки.
   URL: https://code.visualstudio.com/docs

10. Принципы UX/UI для веб-приложений – рекомендации по созданию удобных интерфейсов.
    URL: https://www.nngroup.com/articles/

11. Статьи о планировании времени и цифровых органайзерах – анализ современных решений.
    URL: https://www.researchgate.net

12. Официальная документация по HTML5 и CSS3 – спецификации W3C.
    URL: https://www.w3.org

13. Руководство по тестированию веб-приложений – методы и инструменты.
    URL: https://www.smashingmagazine.com

14. Статьи о трендах в веб-разработке – актуальные технологии и подходы.
    URL: https://css-tricks.com

15. Книги по JavaScript и фронтенд-разработке (например, "Вы не знаете JS" Кайла Симпсона).

Этот список включает ключевые ресурсы, которые помогут обосновать выбор технологий, подходов к разработке и тестированию, а также подчеркнуть актуальность темы проекта.

Приложения

Приложение 1

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Календарь событий</title>
    <link rel="stylesheet" href="styles.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
</head>
<body>
    <div class="container">
        <header>
            <h1><i class="far fa-calendar-alt"></i> Календарь событий</h1>
            <div class="user-actions">
                <button id="login-btn">Войти</button>
                <button id="register-btn">Регистрация</button>
                <button id="friends-btn" onclick="window.location.href='friends.html'"><i
class="fas fa-users"></i> Друзья</button>
            </div>
        </header>

        <div class="main-content">
            <aside class="sidebar">
                <div class="calendar-view">
                    <div class="month-header">
                        <button id="prev-month"><i class="fas fa-chevron-
left"></i></button>
                        <h2 id="current-month">Июнь 2023</h2>
                        <button id="next-month"><i class="fas fa-chevron-
right"></i></button>
                    </div>
                    <div id="calendar-days" class="calendar-days"></div>
                </div>

                <div class="event-categories">
                    <h3>Категории</h3>
                    <ul>
                        <li><span class="category-color work"></span> Работа</li>
                        <li><span class="category-color personal"></span> Личное</li>
                        <li><span class="category-color family"></span> Семья</li>
                        <li><span class="category-color friends"></span> Друзья</li>
                    </ul>
```

```
                </div>
            </aside>

            <main class="event-content">
                <div class="event-actions">
                    <button id="add-event-btn"><i class="fas fa-plus"></i> Добавить
событие</button>
                    <div class="search-box">
                        <input type="text" id="event-search" placeholder="Поиск
событий...">
                        <button id="search-btn"><i class="fas fa-search"></i></button>
                    </div>
                </div>

                <div class="events-list">
                    <h3 id="events-date">События на сегодня</h3>
                    <div id="events-container" class="events-container">
                        <!-- События будут добавляться здесь -->
                    </div>
                </div>
            </main>
        </div>
    </div>

    <!-- Модальные окна -->
    <div id="event-modal" class="modal">
        <div class="modal-content">
            <span class="close">&times;</span>
            <h2 id="modal-title">Добавить новое событие</h2>
            <form id="event-form">
                <div class="form-group">
                    <label for="event-title">Название:</label>
                    <input type="text" id="event-title" required>
                </div>
                <div class="form-group">
                    <label for="event-date">Дата:</label>
                    <input type="date" id="event-date" required>
                </div>
                <div class="form-group">
                    <label for="event-time">Время:</label>
                    <input type="time" id="event-time">
                </div>
                <div class="form-group">
                    <label for="event-category">Категория:</label>
                    <select id="event-category">
                        <option value="work">Работа</option>
                        <option value="personal">Личное</option>
                        <option value="family">Семья</option>
                        <option value="friends">Друзья</option>
                    </select>
                </div>
                <div class="form-group">
```

```
                    <label for="event-description">Описание:</label>
                    <textarea id="event-description" rows="3"></textarea>
                </div>
                <div class="form-group">
                    <label for="event-share">Поделиться с друзьями:</label>
                    <input type="text" id="event-share" placeholder="Введите email друга">
                    <button type="button" id="add-friend-btn">Добавить</button>
                    <div id="friends-list" class="friends-list"></div>
                </div>
                <div class="form-actions">
                    <button type="submit" id="save-event-btn">Сохранить</button>
                    <button type="button" id="delete-event-btn"
class="hidden">Удалить</button>
                </div>
            </form>
        </div>
    </div>

    <div id="auth-modal" class="modal">
        <div class="modal-content">
            <span class="close">&times;</span>
            <h2 id="auth-modal-title">Вход</h2>
            <form id="auth-form">
                <div id="auth-name-group" class="form-group hidden">
                    <label for="auth-name">Имя:</label>
                    <input type="text" id="auth-name">
                </div>
                <div class="form-group">
                    <label for="auth-email">Email:</label>
                    <input type="email" id="auth-email" required>
                </div>
                <div class="form-group">
                    <label for="auth-password">Пароль:</label>
                    <input type="password" id="auth-password" required>
                </div>
                <div class="form-actions">
                    <button type="submit" id="auth-submit-btn">Войти</button>
                </div>
                    <span id="auth-switch-text">Нет аккаунта? </span>
                    <a href="#" id="auth-switch-link">Зарегистрироваться</a>
                </div>
            </form>
        </div>
    </div>

    <script src="script.js"></script>
</body>
</html>
```

Приложение 2

```
<!DOCTYPE html>
<html lang="ru">
```

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Мои друзья | Календарь событий</title>
    <link rel="stylesheet" href="styles.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
</head>
<body>
    <div class="container">
        <header>
            <h1><i class="fas fa-users"></i> Мои друзья</h1>
            <div class="user-actions">
                <button onclick="window.location.href='index.html'"><i class="fas fa-arrow-
left"></i> Назад</button>
            </div>
        </header>

        <div class="main-content">
            <div class="friends-controls">
                <div class="friends-search">
                    <input type="text" id="friend-search" placeholder="Поиск друзей...">
                    <button id="add-friend-btn"><i class="fas fa-user-plus"></i> Добавить
друга</button>
                </div>
            </div>

            <div class="friends-list-container">
                <div id="friends-list">
                    <!-- Друзья будут добавляться здесь -->
                    <div class="friend-card">
                        <div class="friend-avatar">
                            <i class="fas fa-user-circle"></i>
                        </div>
                        <div class="friend-info">
                            <h3>Иван Иванов</h3>
                            <p>ivan@example.com</p>
                            <div class="friend-status online">Online</div>
                        </div>
                        <div class="friend-actions">
                            <button class="friend-action message"><i class="fas fa-
envelope"></i></button>
                            <button class="friend-action remove"><i class="fas fa-trash-
alt"></i></button>
                        </div>
                    </div>

                    <div class="friend-card">
                        <div class="friend-avatar">
                            <i class="fas fa-user-circle"></i>
                        </div>
                        <div class="friend-info">
```

```
                            <h3>Мария Петрова</h3>
                            <p>maria@example.com</p>
                            <div class="friend-status offline">Был(а) 2 часа назад</div>
                        </div>
                        <div class="friend-actions">
                            <button class="friend-action message"><i class="fas fa-
envelope"></i></button>
                            <button class="friend-action remove"><i class="fas fa-trash-
alt"></i></button>
                        </div>
                    </div>

                    <div class="friend-card">
                        <div class="friend-avatar">
                            <i class="fas fa-user-circle"></i>
                        </div>
                        <div class="friend-info">
                            <h3>Алексей Смирнов</h3>
                            <p>alex@example.com</p>
                            <div class="friend-status online">Online</div>
                        </div>
                        <div class="friend-actions">
                            <button class="friend-action message"><i class="fas fa-
envelope"></i></button>
                            <button class="friend-action remove"><i class="fas fa-trash-
alt"></i></button>
                        </div>
                    </div>

                    <div class="friend-card">
                        <div class="friend-avatar">
                            <i class="fas fa-user-circle"></i>
                        </div>
                        <div class="friend-info">
                            <h3>Елена Ковалёва</h3>
                            <p>elena@example.com</p>
                            <div class="friend-status away">Не беспокоить</div>
                        </div>
                        <div class="friend-actions">
                            <button class="friend-action message"><i class="fas fa-
envelope"></i></button>
                            <button class="friend-action remove"><i class="fas fa-trash-
alt"></i></button>
                        </div>
                    </div>

                    <div class="friend-card">
                        <div class="friend-avatar">
                            <i class="fas fa-user-circle"></i>
                        </div>
                        <div class="friend-info">
                            <h3>Дмитрий Соколов</h3>
```

```
                        <p>dmitry@example.com</p>
                        <div class="friend-status offline">Был(а) вчера</div>
                    </div>
                    <div class="friend-actions">
                        <button class="friend-action message"><i class="fas fa-
envelope"></i></button>
                        <button class="friend-action remove"><i class="fas fa-trash-
alt"></i></button>
                    </div>
                </div>
            </div>
        </div>
    </div>

    <!-- Модальное окно для добавления друга -->
    <div id="add-friend-modal" class="modal">
        <div class="modal-content">
            <span class="close">&times;</span>
            <h2><i class="fas fa-user-plus"></i> Добавить друга</h2>
            <form id="add-friend-form">
                <div class="form-group">
                    <label for="friend-email">Email друга:</label>
                    <input type="email" id="friend-email" placeholder="Введите email друга"
required>
                </div>
                <div class="form-group">
                    <label for="friend-name">Имя (необязательно):</label>
                    <input type="text" id="friend-name" placeholder="Введите имя друга">
                </div>
                <div class="form-actions">
                    <button type="submit" class="btn-primary">Добавить</button>
                </div>
            </form>
        </div>
    </div>

    <script>
        document.addEventListener('DOMContentLoaded', function() {
            // Открытие модального окна
            document.getElementById('add-friend-btn').addEventListener('click', function()
{
                document.getElementById('add-friend-modal').style.display = 'block';
            });

            // Закрытие модального окна
            document.querySelector('.close').addEventListener('click', function() {
                document.getElementById('add-friend-modal').style.display = 'none';
            });

            // Поиск друзей
            document.getElementById('friend-search').addEventListener('input', function() {
```

```
                const searchTerm = this.value.toLowerCase();
                const friends = document.querySelectorAll('.friend-card');

                friends.forEach(friend => {
                    const name = friend.querySelector('h3').textContent.toLowerCase();
                    const email = friend.querySelector('p').textContent.toLowerCase();

                    if (name.includes(searchTerm) || email.includes(searchTerm)) {
                        friend.style.display = 'flex';
                    } else {
                        friend.style.display = 'none';
                    }
                });
            });

            // Обработка добавления друга (заглушка)
            document.getElementById('add-friend-form').addEventListener('submit',
function(e) {
                e.preventDefault();
                const email = document.getElementById('friend-email').value;
                const name = document.getElementById('friend-name').value ||
email.split('@')[0];

                alert(`Запрос дружбы отправлен для ${name} (${email})`);
                document.getElementById('add-friend-modal').style.display = 'none';
                this.reset();
            });

            // Удаление друга
            document.querySelectorAll('.friend-action.remove').forEach(btn => {
                btn.addEventListener('click', function() {
                    const friendCard = this.closest('.friend-card');
                    const friendName = friendCard.querySelector('h3').textContent;

                    if (confirm(`Вы уверены, что хотите удалить ${friendName} из друзей?`))
{
                        friendCard.remove();
                    }
                });
            });
        });
    </script>
</body>
</html>
```

Приложение 3

```
/* Общие стили */
* {
    box-sizing: border-box;
    margin: 0;
    padding: 0;
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
```

```
}

body {
    background-color: #f5f5f5;
    color: #333;
    line-height: 1.6;
}

.container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 20px;
}

header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 20px 0;
    border-bottom: 1px solid #ddd;
    margin-bottom: 20px;
}

header h1 {
    font-size: 28px;
    color: #2c3e50;
}

header h1 i {
    margin-right: 10px;
    color: #3498db;
}

.user-actions button {
    padding: 8px 15px;
    margin-left: 10px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-weight: 500;
}

.user-actions button:first-child {
    background-color: #f5f5f5;
    color: #3498db;
    border: 1px solid #3498db;
}

.user-actions button:last-child {
    background-color: #3498db;
    color: white;
}
```

```css
.main-content {
    display: flex;
    gap: 20px;
}

.sidebar {
    flex: 1;
    background: white;
    border-radius: 8px;
    padding: 20px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}

.event-content {
    flex: 3;
    background: white;
    border-radius: 8px;
    padding: 20px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}

/* Стили для страницы друзей */
.friends-controls {
    margin-bottom: 25px;
}

.friends-search {
    display: flex;
    gap: 10px;
    margin-bottom: 20px;
}

#friend-search {
    flex: 1;
    padding: 12px 15px;
    border: 1px solid #ddd;
    border-radius: 6px;
    font-size: 16px;
}

#add-friend-btn {
    padding: 5px 5px;
    background-color: #3498db;
    color: white;
    border: none;
    border-radius: 6px;
    cursor: pointer;
    font-weight: 500;
    white-space: nowrap;
}
```

```css
#add-friend-btn:hover {
    background-color: #2980b9;
}

.friend-card {
    display: flex;
    align-items: center;
    padding: 15px;
    background-color: #fff;
    border-radius: 8px;
    margin-bottom: 12px;
    box-shadow: 0 2px 4px rgba(0,0,0,0.05);
    transition: transform 0.2s, box-shadow 0.2s;
}

.friend-card:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 8px rgba(0,0,0,0.1);
}

.friend-avatar {
    margin-right: 15px;
    font-size: 40px;
    color: #7f8c8d;
}

.friend-info {
    flex: 1;
}

.friend-info h3 {
    margin: 0 0 5px;
    color: #2c3e50;
    font-size: 18px;
}

.friend-info p {
    margin: 0 0 5px;
    color: #7f8c8d;
    font-size: 14px;
}

.friend-status {
    font-size: 12px;
    padding: 2px 6px;
    border-radius: 10px;
    display: inline-block;
}

.friend-status.online {
    background-color: #2ecc71;
    color: white;
```

```css
}

.friend-status.offline {
    background-color: #bdc3c7;
    color: #34495e;
}

.friend-status.away {
    background-color: #f39c12;
    color: white;
}

.friend-actions {
    display: flex;
    gap: 8px;
}

.friend-action {
    width: 36px;
    height: 36px;
    border: none;
    border-radius: 50%;
    cursor: pointer;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 14px;
    transition: background-color 0.2s;
}

.friend-action.message {
    background-color: #3498db;
    color: white;
}

.friend-action.message:hover {
    background-color: #2980b9;
}

.friend-action.remove {
    background-color: #e74c3c;
    color: white;
}

.friend-action.remove:hover {
    background-color: #c0392b;
}


.event-categories li {
    cursor: pointer;
    transition: background-color 0.2s;
    padding: 8px 12px;
```

```
    border-radius: 4px;
}

.event-categories li:hover {
    background-color: #f0f0f0;
}

.event-categories li.active {
    background-color: #e0f7fa;
    font-weight: bold;
}

/* Модальное окно */
.modal {
    display: none;
    position: fixed;
    z-index: 100;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0,0,0,0.5);
}

.modal-content {
    background-color: #fff;
    margin: 10% auto;
    padding: 25px;
    border-radius: 8px;
    width: 90%;
    max-width: 500px;
    box-shadow: 0 5px 15px rgba(0,0,0,0.3);
}

.modal h2 {
    margin-top: 0;
    color: #2c3e50;
}

.close {
    color: #aaa;
    float: right;
    font-size: 24px;
    font-weight: bold;
    cursor: pointer;
}

.close:hover {
    color: #333;
}

.form-group {
```

```
    margin-bottom: 15px;
}

.form-group label {
    display: block;
    margin-bottom: 5px;
    font-weight: 500;
    color: #2c3e50;
}

.form-group input {
    width: 100%;
    padding: 10px 12px;
    border: 1px solid #ddd;
    border-radius: 4px;
    font-size: 16px;
}

.form-actions {
    margin-top: 20px;
    text-align: right;
}

.btn-primary {
    padding: 10px 20px;
    background-color: #3498db;
    color: white;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-weight: 500;
}

.btn-primary:hover {
    background-color: #2980b9;
}

/* Стили для кнопки "Друзья" */
#friends-btn {
    padding: 8px 15px;
    margin-left: 10px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-weight: 500;
    background-color: #2ecc71;
    color: white;
}

#friends-btn:hover {
    background-color: #2ecc71;
}
```

```css
#friends-btn i {
    margin-right: 5px;
}

.user-actions {
    display: flex;
    align-items: center;
    gap: 10px;
}

/* Календарь */
.calendar-view {
    margin-bottom: 20px;
}

.month-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 15px;
}

.month-header h2 {
    font-size: 18px;
    color: #2c3e50;
}

.month-header button {
    background: none;
    border: none;
    font-size: 16px;
    cursor: pointer;
    color: #3498db;
}

.calendar-days {
    display: grid;
    grid-template-columns: repeat(7, 1fr);
    gap: 5px;
}

.day-header {
    text-align: center;
    font-weight: bold;
    padding: 5px;
    color: #7f8c8d;
}

.day {
    text-align: center;
    padding: 8px;
```

```
    border-radius: 4px;
    cursor: pointer;
}

.day:hover {
    background-color: #f0f0f0;
}

.day.today {
    background-color: #3498db;
    color: white;
}

.day.selected {
    background-color: #2980b9;
    color: white;
}

.day.has-events::after {
    content: '';
    display: block;
    width: 5px;
    height: 5px;
    background-color: #e74c3c;
    border-radius: 50%;
    margin: 3px auto 0;
}

/* Категории */
.event-categories h3 {
    margin-bottom: 10px;
    color: #2c3e50;
}

.event-categories ul {
    list-style: none;
}

.event-categories li {
    padding: 8px 0;
    display: flex;
    align-items: center;
}

.category-color {
    display: inline-block;
    width: 12px;
    height: 12px;
    border-radius: 50%;
    margin-right: 8px;
}
```

```css
.category-color.all {
    background-color: #9e9e9e;
}

.category-color.work {
    background-color: #e74c3c;
}

.category-color.personal {
    background-color: #2ecc71;
}

.category-color.family {
    background-color: #9b59b6;
}

.category-color.friends {
    background-color: #f39c12;
}

/* События */
.event-actions {
    display: flex;
    justify-content: space-between;
    margin-bottom: 20px;
}

#add-event-btn {
    background-color: #2ecc71;
    color: white;
    border: none;
    padding: 10px 15px;
    border-radius: 4px;
    cursor: pointer;
    font-weight: 500;
}

#add-event-btn:hover {
    background-color: #27ae60;
}

.search-box {
    display: flex;
    align-items: center;
}

#event-search {
    padding: 8px 12px;
    border: 1px solid #ddd;
    border-radius: 4px 0 0 4px;
    width: 200px;
}
```

```css
#search-btn {
    padding: 8px 12px;
    background-color: #3498db;
    color: white;
    border: none;
    border-radius: 0 4px 4px 0;
    cursor: pointer;
}

.events-list h3 {
    margin-bottom: 15px;
    color: #2c3e50;
}

.events-container {
    display: flex;
    flex-direction: column;
    gap: 10px;
}

.event-card {
    background-color: #f9f9f9;
    border-left: 4px solid #3498db;
    padding: 15px;
    border-radius: 4px;
    cursor: pointer;
    transition: transform 0.2s;
}

.event-card:hover {
    transform: translateX(5px);
}

.event-card.work {
    border-left-color: #e74c3c;
}

.event-card.personal {
    border-left-color: #2ecc71;
}

.event-card.family {
    border-left-color: #9b59b6;
}

.event-card.friends {
    border-left-color: #f39c12;
}

.event-card h4 {
    margin-bottom: 5px;
```

```
    color: #2c3e50;
}

.event-time {
    font-size: 14px;
    color: #7f8c8d;
    margin-bottom: 5px;
}

.event-description {
    font-size: 14px;
    color: #555;
}

/* Модальные окна */
.modal {
    display: none;
    position: fixed;
    z-index: 1;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0,0,0,0.5);
}

.modal-content {
    background-color: white;
    margin: 10% auto;
    padding: 20px;
    border-radius: 8px;
    width: 90%;
    max-width: 500px;
    box-shadow: 0 5px 15px rgba(0,0,0,0.3);
    position: relative;
}

.close {
    position: absolute;
    right: 20px;
    top: 10px;
    font-size: 24px;
    font-weight: bold;
    color: #aaa;
    cursor: pointer;
}

.close:hover {
    color: #333;
}

.form-group {
```

```
    margin-bottom: 15px;
}

.form-group label {
    display: block;
    margin-bottom: 5px;
    font-weight: 500;
    color: #2c3e50;
}

.form-group input,
.form-group select,
.form-group textarea {
    width: 100%;
    padding: 8px 12px;
    border: 1px solid #ddd;
    border-radius: 4px;
    font-size: 14px;
}

.form-group textarea {
    resize: vertical;
}

.form-actions {
    display: flex;
    justify-content: space-between;
    margin-top: 20px;
}

.form-actions button {
    padding: 10px 15px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-weight: 500;
}

#save-event-btn {
    background-color: #2ecc71;
    color: white;
}

#delete-event-btn {
    background-color: #e74c3c;
    color: white;
}

#auth-submit-btn {
    background-color: #3498db;
    color: white;
    width: 100%;
```

```
}

.friends-list {
    margin-top: 10px;
    display: flex;
    flex-wrap: wrap;
    gap: 5px;
}

.friend-tag {
    background-color: #e0f7fa;
    color: #00838f;
    padding: 3px 8px;
    border-radius: 12px;
    font-size: 12px;
    display: flex;
    align-items: center;
}

.friend-tag button {
    background: none;
    border: none;
    color: #00838f;
    margin-left: 5px;
    cursor: pointer;
    font-size: 10px;
}

.hidden {
    display: none !important;
}

/* Адаптивность */
@media (max-width: 768px) {
    .main-content {
        flex-direction: column;
    }

    .event-actions {
        flex-direction: column;
        gap: 10px;
    }

    .search-box {
        width: 100%;
    }

    #event-search {
        width: 100%;
    }
}
```

Приложение 4

```
document.addEventListener('DOMContentLoaded', function() {
    // Текущая дата
    let currentDate = new Date();
    let currentMonth = currentDate.getMonth();
    let currentYear = currentDate.getFullYear();
    let selectedDate = new Date();

    // Модальные окна
    const eventModal = document.getElementById('event-modal');
    const authModal = document.getElementById('auth-modal');
    const addEventBtn = document.getElementById('add-event-btn');
    const closeButtons = document.querySelectorAll('.close');
    const loginBtn = document.getElementById('login-btn');
    const registerBtn = document.getElementById('register-btn');
    const authSwitchLink = document.getElementById('auth-switch-link');
    const authModalTitle = document.getElementById('auth-modal-title');
    const authNameGroup = document.getElementById('auth-name-group');
    const authSubmitBtn = document.getElementById('auth-submit-btn');
    const authForm = document.getElementById('auth-form');

    // Форма события
    const eventForm = document.getElementById('event-form');
    const eventTitle = document.getElementById('event-title');
    const eventDate = document.getElementById('event-date');
    const eventTime = document.getElementById('event-time');
    const eventCategory = document.getElementById('event-category');
    const eventDescription = document.getElementById('event-description');
    const eventShare = document.getElementById('event-share');
    const addFriendBtn = document.getElementById('add-friend-btn');
    const friendsList = document.getElementById('friends-list');
    const saveEventBtn = document.getElementById('save-event-btn');
    const deleteEventBtn = document.getElementById('delete-event-btn');
    const modalTitle = document.getElementById('modal-title');

    // Календарь
    const currentMonthElement = document.getElementById('current-month');
    const calendarDays = document.getElementById('calendar-days');
    const prevMonthBtn = document.getElementById('prev-month');
    const nextMonthBtn = document.getElementById('next-month');

    // События
    const eventsDate = document.getElementById('events-date');
    const eventsContainer = document.getElementById('events-container');

    // Состояние приложения
    let events = JSON.parse(localStorage.getItem('events')) || [];
    let friends = [];
    let isEditing = false;
    let currentEventId = null;
    let isLoginMode = true;
    let isLoggedIn = false;
```

```
// Элементы поиска
const eventSearch = document.getElementById('event-search');
const searchBtn = document.getElementById('search-btn');

// Инициализация приложения
function init() {
    renderCalendar();
    renderEvents(selectedDate);

    // Установка текущей даты в форме
    const formattedDate = formatDateForInput(selectedDate);
    eventDate.value = formattedDate;

    // Проверяем, был ли пользователь авторизован ранее
    checkAuthStatus();

    // Добавляем обработчики для фильтрации по категориям
    setupCategoryFilters();
}


// Настройка фильтров по категориям
function setupCategoryFilters() {
    const categoryElements = document.querySelectorAll('.category-filter');

    categoryElements.forEach(element => {
        element.addEventListener('click', function() {
            const category = this.dataset.category;
            filterEventsByCategory(category);

            // Обновляем активный фильтр
            categoryElements.forEach(el => el.classList.remove('active'));
            this.classList.add('active');
        });
    });

    // Кнопка "Все" для сброса фильтра
    document.getElementById('all-categories').addEventListener('click', function() {
        renderEvents(selectedDate);
        categoryElements.forEach(el => el.classList.remove('active'));
        this.classList.add('active');
    });
}

function filterEventsByCategory(category) {
    const filteredEvents = events.filter(event => event.category === category);

    eventsContainer.innerHTML = '';

    if (filteredEvents.length === 0) {
        eventsContainer.innerHTML = '<p>Нет событий в этой категории</p>';
```

```
            return;
        }

        // Группируем события по дате
        const eventsByDate = {};
        filteredEvents.forEach(event => {
            const dateStr = new Date(event.date).toLocaleDateString('ru-RU');
            if (!eventsByDate[dateStr]) {
                eventsByDate[dateStr] = [];
            }
            eventsByDate[dateStr].push(event);
        });

         // Сортируем даты
        const sortedDates = Object.keys(eventsByDate).sort((a, b) => {
            return new Date(a.split('.').reverse().join('-')) - new
Date(b.split('.').reverse().join('-'));
        });

        // Отображаем события
        sortedDates.forEach(dateStr => {
            const dateHeader = document.createElement('h3');
            dateHeader.className = 'category-date-header';
            dateHeader.textContent = dateStr;
            eventsContainer.appendChild(dateHeader);

            eventsByDate[dateStr].forEach(event => {
                const eventElement = document.createElement('div');
                eventElement.className = `event-card ${event.category}`;
                eventElement.dataset.id = event.id;

                let timeHTML = '';
                if (event.time) {
                    timeHTML = `<div class="event-time">${event.time}</div>`;
                }

                eventElement.innerHTML = `
                    <h4>${event.title}</h4>
                    ${timeHTML}
                    <div class="event-description">${event.description || ''}</div>
                `;

                eventElement.addEventListener('click', () => editEvent(event.id));
                eventsContainer.appendChild(eventElement);
            });
        });
    }

    // Проверка статуса авторизации
    function checkAuthStatus() {
        const authStatus = localStorage.getItem('isLoggedIn');
        if (authStatus === 'true') {
```

```
            isLoggedIn = true;
            updateAuthUI();
        }
    }

    // Обновление UI в зависимости от статуса авторизации
    function updateAuthUI() {
        if (isLoggedIn) {
            loginBtn.textContent = 'Выйти';
            registerBtn.style.display = 'none';
            addEventBtn.style.display = 'block';
            // Показываем навигацию только для авторизованных пользователей
            document.querySelector('.main-nav').style.display = 'flex';
            renderCalendar();
            renderEvents(selectedDate);
        } else {
            loginBtn.textContent = 'Войти';
            registerBtn.style.display = 'inline-block';
            addEventBtn.style.display = 'none';
            // Скрываем навигацию для неавторизованных
            document.querySelector('.main-nav').style.display = 'none';
            eventsContainer.innerHTML = '<p>Пожалуйста, войдите для просмотра событий</p>';
            document.querySelectorAll('.day.has-events').forEach(day => {
                day.classList.remove('has-events');
            });
        }
    }

    // Форматирование даты для input[type="date"]
    function formatDateForInput(date) {
        const d = new Date(date);
        let month = '' + (d.getMonth() + 1);
        let day = '' + d.getDate();
        const year = d.getFullYear();

        if (month.length < 2) month = '0' + month;
        if (day.length < 2) day = '0' + day;

        return [year, month, day].join('-');
    }

    // Рендер календаря
    function renderCalendar() {
        // Очистка календаря
        calendarDays.innerHTML = '';

        // Заголовки дней недели
        const daysOfWeek = ['Пн', 'Вт', 'Ср', 'Чт', 'Пт', 'Сб', 'Вс'];
        daysOfWeek.forEach(day => {
            const dayElement = document.createElement('div');
            dayElement.className = 'day-header';
            dayElement.textContent = day;
```

```
            calendarDays.appendChild(dayElement);
        });

        // Первый день месяца
        const firstDay = new Date(currentYear, currentMonth, 1);
        // Последний день месяца
        const lastDay = new Date(currentYear, currentMonth + 1, 0);
        // День недели первого дня месяца (0 - воскресенье, 1 - понедельник и т.д.)
        const firstDayOfWeek = firstDay.getDay() === 0 ? 6 : firstDay.getDay() - 1; //
Преобразуем к 0-6 (пн-вс)

        // Обновление заголовка месяца
        const monthNames = ['Январь', 'Февраль', 'Март', 'Апрель', 'Май', 'Июнь',
                        'Июль', 'Август', 'Сентябрь', 'Октябрь', 'Ноябрь', 'Декабрь'];
        currentMonthElement.textContent = `${monthNames[currentMonth]} ${currentYear}`;

        // Добавляем пустые ячейки для дней предыдущего месяца (для выравнивания)
        for (let i = 0; i < firstDayOfWeek; i++) {
            const emptyDay = document.createElement('div');
            emptyDay.className = 'day empty';
            calendarDays.appendChild(emptyDay);
        }

        // Дни текущего месяца
        for (let i = 1; i <= lastDay.getDate(); i++) {
            const dayElement = document.createElement('div');
            dayElement.className = 'day';
            dayElement.textContent = i;

            const date = new Date(currentYear, currentMonth, i);

            // Проверка на сегодняшний день
            if (isSameDay(date, new Date())) {
                dayElement.classList.add('today');
            }

            // Проверка на выбранный день
            if (isSameDay(date, selectedDate)) {
                dayElement.classList.add('selected');
            }

            // Проверка на наличие событий (только если пользователь авторизован)
            if (isLoggedIn && hasEventsOnDate(date)) {
                dayElement.classList.add('has-events');
            }

            dayElement.addEventListener('click', () => selectDate(date));
            calendarDays.appendChild(dayElement);
        }

        // Добавляем CSS для пустых ячеек
        const style = document.createElement('style');
```

```
        style.textContent = `
            .day.empty {
                visibility: hidden;
                pointer-events: none;
            }
        `;
        document.head.appendChild(style);
    }
    // Проверка на совпадение дат (без времени)
    function isSameDay(date1, date2) {
        return date1.getFullYear() === date2.getFullYear() &&
               date1.getMonth() === date2.getMonth() &&
               date1.getDate() === date2.getDate();
    }

    // Проверка наличия событий на дату
    function hasEventsOnDate(date) {
        return events.some(event => isSameDay(new Date(event.date), date));
    }

    // Выбор даты
    function selectDate(date) {
        selectedDate = date;
        renderCalendar();
        renderEvents(date);

        // Обновление заголовка событий
        const options = { weekday: 'long', year: 'numeric', month: 'long', day: 'numeric'
};
        eventsDate.textContent = `События на ${date.toLocaleDateString('ru-RU', options)}`;
    }

    // Рендер событий
    function renderEvents(date) {
        eventsContainer.innerHTML = '';

        if (!isLoggedIn) {
            eventsContainer.innerHTML = '<p>Пожалуйста, войдите для просмотра событий</p>';
            return;
        }

        const filteredEvents = events.filter(event => isSameDay(new Date(event.date),
date));

        if (filteredEvents.length === 0) {
            eventsContainer.innerHTML = '<p>Нет событий на эту дату</p>';
            return;
        }

        filteredEvents.sort((a, b) => {
            if (a.time && b.time) {
                return a.time.localeCompare(b.time);
```

```
            }
            return 0;
        });

        filteredEvents.forEach(event => {
            const eventElement = document.createElement('div');
            eventElement.className = `event-card ${event.category}`;
            eventElement.dataset.id = event.id;

            let timeHTML = '';
            if (event.time) {
                timeHTML = `<div class="event-time">${event.time}</div>`;
            }

            eventElement.innerHTML = `
                <h4>${event.title}</h4>
                ${timeHTML}
                <div class="event-description">${event.description || ''}</div>
            `;

            eventElement.addEventListener('click', () => editEvent(event.id));
            eventsContainer.appendChild(eventElement);
        });
    }

    // Переключение месяца
    prevMonthBtn.addEventListener('click', () => {
        currentMonth--;
        if (currentMonth < 0) {
            currentMonth = 11;
            currentYear--;
        }
        renderCalendar();
    });

    nextMonthBtn.addEventListener('click', () => {
        currentMonth++;
        if (currentMonth > 11) {
            currentMonth = 0;
            currentYear++;
        }
        renderCalendar();
    });

    // Открытие модального окна для добавления события
    addEventBtn.addEventListener('click', () => {
        if (!isLoggedIn) return;

        isEditing = false;
        currentEventId = null;
        modalTitle.textContent = 'Добавить новое событие';
        eventForm.reset();
```

```
        friends = [];
        renderFriendsList();
        deleteEventBtn.classList.add('hidden');

        // Установка выбранной даты
        eventDate.value = formatDateForInput(selectedDate);

        openModal(eventModal);
    });

    // Редактирование события
    function editEvent(id) {
        if (!isLoggedIn) return;

        const event = events.find(e => e.id === id);
        if (!event) return;

        isEditing = true;
        currentEventId = id;
        modalTitle.textContent = 'Редактировать событие';
        deleteEventBtn.classList.remove('hidden');

        // Заполнение формы
        eventTitle.value = event.title;
        eventDate.value = formatDateForInput(new Date(event.date));
        eventTime.value = event.time || '';
        eventCategory.value = event.category;
        eventDescription.value = event.description || '';

        friends = event.sharedWith || [];
        renderFriendsList();

        openModal(eventModal);
    }

    // Рендер списка друзей
    function renderFriendsList() {
        friendsList.innerHTML = '';
        friends.forEach((friend, index) => {
            const friendElement = document.createElement('div');
            friendElement.className = 'friend-tag';
            friendElement.innerHTML = `
                ${friend}
                <button data-index="${index}">&times;</button>
            `;
            friendsList.appendChild(friendElement);
        });

        // Обработчики удаления друзей
        document.querySelectorAll('.friend-tag button').forEach(btn => {
            btn.addEventListener('click', (e) => {
                e.stopPropagation();
```

```
            const index = parseInt(btn.dataset.index);
            friends.splice(index, 1);
            renderFriendsList();
        });
    });
}

// Добавление друга
addFriendBtn.addEventListener('click', (e) => {
    e.preventDefault();
    const email = eventShare.value.trim();
    if (email && !friends.includes(email)) {
        friends.push(email);
        eventShare.value = '';
        renderFriendsList();
    }
});

// Сохранение события
eventForm.addEventListener('submit', (e) => {
    e.preventDefault();

    const event = {
        id: isEditing ? currentEventId : Date.now().toString(),
        title: eventTitle.value,
        date: eventDate.value,
        time: eventTime.value,
        category: eventCategory.value,
        description: eventDescription.value,
        sharedWith: [...friends]
    };

    if (isEditing) {
        // Обновление существующего события
        const index = events.findIndex(e => e.id === currentEventId);
        if (index !== -1) {
            events[index] = event;
        }
    } else {
        // Добавление нового события
        events.push(event);
    }

    // Сохранение в localStorage
    localStorage.setItem('events', JSON.stringify(events));

    // Обновление интерфейса
    renderCalendar();
    renderEvents(selectedDate);
    closeModal(eventModal);
});
```

```
// Удаление события
deleteEventBtn.addEventListener('click', () => {
    if (confirm('Вы уверены, что хотите удалить это событие?')) {
        events = events.filter(e => e.id !== currentEventId);
        localStorage.setItem('events', JSON.stringify(events));
        renderCalendar();
        renderEvents(selectedDate);
        closeModal(eventModal);
    }
});

// Управление модальными окнами
function openModal(modal) {
    modal.style.display = 'block';
}

function closeModal(modal) {
    modal.style.display = 'none';
}

closeButtons.forEach(btn => {
    btn.addEventListener('click', function() {
        const modal = this.closest('.modal');
        closeModal(modal);
    });
});

window.addEventListener('click', (e) => {
    if (e.target.classList.contains('modal')) {
        closeModal(e.target);
    }
});


// Функция переключения в режим входа
function switchToLoginMode() {
    isLoginMode = true;
    authModalTitle.textContent = 'Вход';
    authNameGroup.classList.add('hidden');
    authSubmitBtn.textContent = 'Войти';
    document.getElementById('auth-switch-text').textContent = 'Нет аккаунта? ';
    authSwitchLink.textContent = 'Зарегистрироваться';
}

// Функция переключения в режим регистрации
function switchToRegisterMode() {
    isLoginMode = false;
    authModalTitle.textContent = 'Регистрация';
    authNameGroup.classList.remove('hidden');
    authSubmitBtn.textContent = 'Зарегистрироваться';
    document.getElementById('auth-switch-text').textContent = 'Уже есть аккаунт? ';
    authSwitchLink.textContent = 'Войти';
```

```
}

// Авторизация/регистрация
loginBtn.addEventListener('click', () => {
    if (isLoggedIn) {
        // Выход из аккаунта
        isLoggedIn = false;
        localStorage.setItem('isLoggedIn', 'false');
        updateAuthUI();
        return;
    }

    switchToLoginMode();
    openModal(authModal);
});

registerBtn.addEventListener('click', () => {
    switchToRegisterMode();
    openModal(authModal);
});

authSwitchLink.addEventListener('click', (e) => {
    e.preventDefault();
    if (isLoginMode) {
        switchToRegisterMode();
    } else {
        switchToLoginMode();
    }
});

authForm.addEventListener('submit', (e) => {
    e.preventDefault();
    const email = document.getElementById('auth-email').value;
    const password = document.getElementById('auth-password').value;

    if (isLoginMode) {
        // Логика входа
        isLoggedIn = true;
        localStorage.setItem('isLoggedIn', 'true');
        alert(`Вход выполнен для ${email}`);
    } else {
        const name = document.getElementById('auth-name').value;
        // Логика регистрации
        isLoggedIn = true;
        localStorage.setItem('isLoggedIn', 'true');
        alert(`Пользователь ${name} (${email}) зарегистрирован`);
    }

    closeModal(authModal);
    updateAuthUI();
});
```

```
    // Обработчики поиска
    searchBtn.addEventListener('click', searchEvents);
    eventSearch.addEventListener('keypress', function(e) {
        if (e.key === 'Enter') {
            searchEvents();
        }
    });

    function searchEvents() {
        const searchTerm = eventSearch.value.trim().toLowerCase();

        if (!searchTerm) {
            renderEvents(selectedDate);
            return;
        }

        const filteredEvents = events.filter(event =>
            event.title.toLowerCase().includes(searchTerm) ||
            (event.description && event.description.toLowerCase().includes(searchTerm))
        );

        displaySearchResults(filteredEvents);
    }

    function displaySearchResults(filteredEvents) {
        eventsContainer.innerHTML = '';

        if (filteredEvents.length === 0) {
            eventsContainer.innerHTML = '<p>События не найдены</p>';
            return;
        }

        filteredEvents.forEach(event => {
            const eventElement = document.createElement('div');
            eventElement.className = `event-card ${event.category}`;
            eventElement.dataset.id = event.id;

            let timeHTML = '';
            if (event.time) {
                timeHTML = `<div class="event-time">${event.time}</div>`;
            }

            eventElement.innerHTML = `
                <h4>${event.title}</h4>
                ${timeHTML}
                <div class="event-date">${new Date(event.date).toLocaleDateString('ru-
RU')}</div>
                <div class="event-description">${event.description || ''}</div>
            `;

            eventElement.addEventListener('click', () => editEvent(event.id));
            eventsContainer.appendChild(eventElement);
```

```
        });
    }

    // Инициализация приложения
    init();

    // Обработчик изменения размера окна для корректного отображения кнопок
    window.addEventListener('resize', function() {
        updateAuthUI();
    });
});
```

Приложение 5

```
            document.addEventListener('DOMContentLoaded', function() {
            // Открытие модального окна
            document.getElementById('add-friend-btn').addEventListener('click', function()
{
                document.getElementById('add-friend-modal').style.display = 'block';
            });

            // Закрытие модального окна
            document.querySelector('.close').addEventListener('click', function() {
                document.getElementById('add-friend-modal').style.display = 'none';
            });

            // Поиск друзей
            document.getElementById('friend-search').addEventListener('input', function() {
                const searchTerm = this.value.toLowerCase();
                const friends = document.querySelectorAll('.friend-card');

                friends.forEach(friend => {
                    const name = friend.querySelector('h3').textContent.toLowerCase();
                    const email = friend.querySelector('p').textContent.toLowerCase();

                    if (name.includes(searchTerm) || email.includes(searchTerm)) {
                        friend.style.display = 'flex';
                    } else {
                        friend.style.display = 'none';
                    }
                });
            });

            // Обработка добавления друга (заглушка)
            document.getElementById('add-friend-form').addEventListener('submit',
function(e) {
                e.preventDefault();
                const email = document.getElementById('friend-email').value;
```

```
                const name = document.getElementById('friend-name').value ||
email.split('@')[0];

                alert(`Запрос дружбы отправлен для ${name} (${email})`);
                document.getElementById('add-friend-modal').style.display = 'none';
                this.reset();
            });

            // Удаление друга
            document.querySelectorAll('.friend-action.remove').forEach(btn => {
                btn.addEventListener('click', function() {
                    const friendCard = this.closest('.friend-card');
                    const friendName = friendCard.querySelector('h3').textContent;

                    if (confirm(`Вы уверены, что хотите удалить ${friendName} из друзей?`))
{
                        friendCard.remove();
                    }
                });
            });
        });
```