**Техническое задание на разработку веб-приложения для учета расходов**

1. **Введение**

Веб-приложение для учета расходов предназначено для помощи пользователям в систематизации своих финансов. Оно позволит пользователям ввести, отслеживать и анализировать свои расходы посредством удобного интерфейса. Проект будет включать функционал для создания отчетов по расходам и визуализацию данных.

**2. Цели проекта**

Создать удобное и простое в использовании веб-приложение, которое позволит пользователям регистрировать и отслеживать свои расходы.

Разработать систему отчетности для анализа финансовых потоков пользователя.

Обеспечить доступность стоимости на разных устройствах через адаптивный дизайн.

**3. Основные функции**

Регистрация и авторизация пользователей:

Пользователь должен иметь возможность зарегистрироваться и войти в систему, используя электронную почту и пароль.

Добавление расходов:

Пользователь сможет добавлять расходы, указывая сумму, категорию и дату.

Управление категориями расходов:

Возможность добавления, редактирования и удаления категорий расходов для удобства учета.

Отчеты:

Генерация отчетов о расходах за выбранные временные промежутки (например, за день, неделю, месяц).

Визуализация данных:

Интерактивные графики и диаграммы, показывающие распределение расходов по категориям.

Экспорт данных:

Возможность экспортировать данные о расходах в формате CSV или PDF.

**4. Технические требования**

Платформа: Веб-приложение должно быть доступно через современные браузеры (Chrome, Firefox, Safari).

Язык программирования: Использовать JavaScript (React.js для фронтенда и Node.js для бэкенда).

База данных: Хранение данных должно производиться в реляционной базе данных (например, PostgreSQL).

Адаптивный дизайн: Интерфейс приложения должен корректно отображаться на мобильных и десктопных устройствах.

**5. Этапы разработки**

Анализ требований: Сбор и анализ дополнительных требований от конечных пользователей.

Разработка прототипа: Создание прототипа интерфейса и макетов страниц.

Разработка фронтенда и бэкенда: Программирование интерфейса и серверной части приложения.

Тестирование: Проведение функционального и нагрузочного тестирования приложения.

Запуск: Вывод приложения в продакшн и начало работы с пользователями.

Обратная связь и поддержка: Собираем отзывы от пользователей и вносим изменения по мере необходимости.

**6. Сроки выполнения**

Общий срок выполнения проекта составит 4 месяца. Каждый этап будет занимать приблизительно 4 недели.

**7. Заключение**

Создание веб-приложения для учета расходов позволит пользователям легко отслеживать свои финансы, анализировать их и принимать обоснованные финансовые решения. Успешная реализация данного проекта поможет не только упростить процесс ведения личного бюджета, но и улучшить финансовую грамотность пользователей.

**1. Frontend (Клиентская часть)**

**1.1 HTML (HyperText Markup Language)**

**Описание:** HTML является основным языком разметки, используемым для создания структуры веб-страниц.

**Причины выбора:**

**Стандартная разметка:** HTML обеспечивает семантическое представление контента, что улучшает доступность и SEO (поисковую оптимизацию).

**Совместимость:** Поддерживается всеми современными браузерами.

**1.2 CSS (Cascading Style Sheets)**

**Описание:** CSS используется для стилизации HTML-документов.

**Причины выбора:**

**Визуальный дизайн:** Позволяет создать привлекательный и удобный интерфейс.

**Адаптивность:** CSS-фреймворки (например, Bootstrap или Tailwind CSS) помогают быстро создавать адаптивные дизайн-шаблоны, подходящие для разных устройств.

**1.3 JavaScript**

**Описание:** JavaScript — это язык программирования, который позволяет делать веб-страницы интерактивными и динамичными.

**Причины выбора:**

**Интерактивность:** Реализует обработку событий, динамическое обновление контента и взаимодействие с пользователем.

**Разнообразие фреймворков:** Использование библиотек и фреймворков, таких как React, Vue.js или Angular, позволяет значительно ускорить разработку сложных интерфейсов и улучшить пользовательский опыт.

**2. Backend (Серверная часть)**

**2.1 Node.js**

**Описание:** Node.js — это среда выполнения JavaScript на сервере, основанная на движке V8 от Google.

**Причины выбора:**

**Высокая производительность:** Поддержка асинхронного ввода-вывода позволяет обрабатывать множество соединений одновременно.

**Единый стек:** Использование JavaScript как на клиенте, так и на сервере упрощает процесс разработки и поддержку проекта.

**2.2 Express.js**

**Описание:** Express — это легковесный фреймворк для Node.js, который упрощает создание веб-приложений.

**Причины выбора:**

**Простота:** Позволяет быстро настроить сервер, управлять маршрутами и обрабатывать запросы и ответы.

**Гибкость:** Поддерживает множество middleware, что упрощает интеграцию различных функциональностей.

**2.3 База данных**

**MongoDB**  
**Описание:** MongoDB — это документо-ориентированная NoSQL база данных.**Причины выбора:**

**Гибкость:** Хранение данных в формате JSON позволяет легко работать с динамическими структурами данных, что удобно для приложений с изменяющимся объемом данных.

**Масштабируемость:** Легко масштабируется по горизонтали, что позволяет обрабатывать большие объемы информации.

**3. Другие технологии**

**3.1 RESTful API**

**Описание:** RESTful API позволяет фронтенду взаимодействовать с бэкендом через сетевые запросы.

**Причины выбора:**

**Стандартизированный подход:** REST принципы упрощают интеграцию между разными частями приложения.

**Гибкость:** Позволяет использовать разные клиенты (веб, мобильные приложения) для доступа к одному и тому же API.