вот краткий саммари нашего позапрошлого диалога:
1. Вы сообщили, что установили Python, VS Code и Git, и готовы начать разработку веб-приложения для Telegram Mini App.
2. Вы предоставили подробные требования к приложению, включая описание главной страницы, страницы создания и редактирования списка, страницы создания и редактирования слова, страницы режима повтора, и функциональности выбора нескольких элементов.
3. На основе ваших требований я предложил структуру файлов для проекта, которая включает:
   * Основной HTML файл
   * Директории для стилей, скриптов и компонентов
   * Отдельные файлы для различных функциональностей (списки, карточки, повторение)
   * Компоненты для повторяющихся элементов интерфейса
4. Структура разработана с учетом специфики Telegram Mini App и требований к мобильному интерфейсу.
5. Было отмечено, что следует использовать готовые компоненты фреймворка, где это возможно, и добавлять подтверждение перед удалением элементов.
6. Файлы со схемой и требованиями прилагаю, давай продолжим с начала написания кода

Вот краткий саммари нашего прошлого диалога:
Отлично! Давайте сделаем промежуточное резюме текущей реализации.

Реализованная функциональность:

1. Структура приложения:
   - Главная страница со списком наборов карточек
   - Страница списка слов
   - Страница редактирования/добавления слова
   - Страница режима повторения

2. Навигация:
   - Переход между страницами
   - Кнопка "Назад" на всех страницах, кроме главной

3. Управление списками и словами:
   - Отображение списков на главной странице
   - Отображение слов внутри списка
   - Добавление новых слов
   - Редактирование существующих слов

4. Режим повторения:
   - Настройки повторения (выбор стороны и порядка слов)
   - Отображение карточек в режиме повторения
   - Кнопки "Ответ" и "Помню"

Структура проекта:

1. `index.html`: Основной HTML файл
2. `styles/main.css`: Стили приложения
3. `scripts/main.js`: Основной JavaScript файл с объектом `app`
4. `scripts/components/`:
   - `mainPage.js`: Компонент главной страницы
   - `listPage.js`: Компонент страницы списка
   - `wordPage.js`: Компонент страницы слова
   - `repeatPage.js`: Компонент страницы повторения

Основные методы в `app`:

- `renderPage()`: Отрисовка текущей страницы
- `startRepeat()`: Запуск режима повторения
- `showAnswer()`: Показ ответа в режиме повторения
- `nextWord()`: Переход к следующему слову в режиме повторения

Нереализованные функции:

1. Возможность выбрать несколько элементов (списков или слов)
2. Удаление списков и слов (свайп или кнопка удаления)
3. Переименование списков (свайп слева направо)
4. Изменение порядка элементов в списке (перетаскивание)
5. Добавление слов, на которых был нажат "Ответ", в очередь на повторение
6. Всплывающее окно с поздравлением после завершения всех карточек
7. Подтверждение удаления элементов

Дополнительные улучшения:

1. Сохранение данных (списки и слова) в локальное хранилище браузера
2. Улучшение дизайна и пользовательского интерфейса
3. Обработка ошибок и пустых состояний (например, пустой список слов)
4. Оптимизация производительности для больших списков

Это резюме показывает, что основная структура приложения реализована, но еще есть ряд функций, которые нужно добавить для полного соответствия требованиям. Следующим шагом может быть реализация функций удаления и редактирования, а также улучшение режима повторения.

Но я предложила начать с того, чтобы сделать возможным Сохранение данных (списки и слова) в локальное хранилище браузера, мне кажется это первостепенно важно