**Firebase Configuration**

Файл firebase-config.json содержит конфигурацию Firebase для подключения к проекту dbprivat-ee27e. Конфигурация включает ключи и идентификаторы, необходимые для аутентификации и взаимодействия с Firebase Firestore.

**HTML**

Файл index.html описывает структуру веб-приложения:

* Основной заголовок Repair Management System.
* Кнопки для добавления акта (Внести Акт), поиска устройства (Поиск устройства), и добавления устройства (Добавить устройство).
* Формы для добавления акта (form-container), поиска устройства (search-container), и добавления устройства (add-device-container).
* Скрипты utils.js, search.js, и app.js подключены для реализации функционала.

**CSS**

Файл styles.css содержит стили для оформления веб-приложения.

**JavaScript**

Файл app.js включает основную логику приложения:

1. **Инициализация элементов и переменных:**

javascript

Копировать код

import { searchDeviceRepairs } from './search.js';

import { createTable } from './utils.js';

1. **Обработчики событий:**
   * Кнопки для перехода между формами (addActButton, searchDeviceButton, addDeviceButton, backButton, searchBackButton, addDeviceBackButton).
   * Обработчики событий для отправки форм (actForm, device-form).
   * Обработчики для динамической загрузки данных и обновления DOM.
2. **Функции для взаимодействия с Firebase:**
   * loadDeviceNumbers: Загрузка номеров устройств.
   * loadDeviceType: Загрузка типа устройства.
   * loadPreviousRepairs: Загрузка предыдущих ремонтов.
   * fetchWorkTypes: Загрузка типов работ.
   * updateTotalCost: Обновление общей стоимости.
3. **Функции для управления состоянием формы:**
   * clearForm, clearSearch, clearAddDeviceForm: Очистка форм и сброс состояния.
   * populateRegionSelect: Заполнение списка регионов.
4. **Обработчики для добавления и удаления ремонтных работ:**

javascript

Копировать код

addRepairButton.addEventListener('click', async () => {

// Логика для добавления ремонта

});

**Дополнительные файлы**

* search.js: Функции для поиска и отображения результатов поиска.
* utils.js: Вспомогательные функции, такие как создание таблиц (createTable).

**Ключевые моменты и взаимодействие:**

1. **Добавление акта:**
   * Форма act-form собирает данные акта, включая номер акта, номер устройства, дату ремонта, типы ремонта и их количество.
   * Данные отправляются на сервер для сохранения в базе данных.
2. **Добавление устройства:**
   * Форма device-form собирает данные устройства, включая номер устройства, модель, серийный номер и регион.
   * Данные отправляются на сервер для сохранения в базе данных.
3. **Поиск устройства:**
   * Форма поиска позволяет пользователю искать устройства по их ID.
   * Результаты отображаются в виде таблицы с информацией о предыдущих ремонтах и доступных типах работ.
4. **Динамическое обновление интерфейса:**
   * Обработчики событий и функции обновляют интерфейс на основе взаимодействий пользователя, таких как выбор устройства, добавление или удаление ремонтных работ.

Эта информация предоставляет полное представление о проекте и его функциональности, что поможет вам в будущем взаимодействовать с другими разработчиками или автоматизированными системами.