Dokumentasi Proyek: UnionSpace CRM

1. Pendahuluan

Dokumen ini memberikan gambaran teknis mengenai proyek **UnionSpace CRM**, sebuah sistem manajemen hubungan pelanggan (CRM) yang dirancang khusus untuk kebutuhan co-working space. Aplikasi ini dibangun di atas platform Firebase, dengan frontend berbasis React dan backend menggunakan Firebase Cloud Functions.

Tujuan utama aplikasi ini adalah untuk memfasilitasi manajemen operasional co-working space, termasuk: - Manajemen data pelanggan (Customers) - Pengelolaan gedung (Buildings) dan ruangan (Spaces) - Pemesanan (Orders) dan penjadwalan - Keuangan, termasuk pembuatan invoice dan pembayaran - Manajemen konten seperti artikel dan promo

2. Teknologi Utama

Kategori	Teknologi	Keterangan
Platform	Google Firebase	Digunakan untuk hosting, database, functions (backend), dan otentikasi.
Frontend	React.js (Vite)	Library JavaScript untuk membangun antarmuka pengguna yang interaktif.
Backend	Node.js	Lingkungan eksekusi untuk Firebase Cloud Functions.
Database	Firestore	Database NoSQL real-time yang fleksibel dan skalabel.
Styling	CSS Murni	Styling dasar tanpa framework CSS spesifik.

3. Struktur Proyek

Proyek ini dibagi menjadi dua bagian utama: `frontend` untuk aplikasi sisi klien dan `functions` untuk logika sisi server.

```
unionspace_crm/
  frontend/         # Kode sumber aplikasi React (sisi klien)
     src/
         components/ # Komponen UI yang dapat digunakan kembali
         config/     # Konfigurasi koneksi Firebase
```

```
        contexts/   # React Context untuk state management global
        hooks/      # Custom hooks untuk logika bisnis (mis: fetch data)
        services/   # Layer untuk berinteraksi dengan API backend
        ...
    package.json    # Dependensi dan skrip untuk frontend

functions/        # Kode sumber untuk backend (Firebase Cloud Functions)
    src/
        services/   # Servis pendukung (mis: image service)
        utils/      # Utilitas (mis: konfigurasi CORS)
        *.js        # File-file yang mendefinisikan setiap fungsi/endpoint API
        ...
    package.json    # Dependensi dan skrip untuk backend

firebase.json     # Konfigurasi deploy Firebase (hosting, functions, rules)
firestore.rules   # Aturan keamanan untuk database Firestore
...
```

3.1. Frontend (`/frontend`)

- **`src/components`**: Direktori ini berisi semua komponen React yang membangun antarmuka aplikasi. Komponen dikelompokkan berdasarkan fitur, seperti `auth`, `buildings`, `customers`, `orders`, dll. Ini membuat struktur UI menjadi modular dan mudah dikelola.
- **`src/hooks`**: Custom hooks digunakan untuk mengisolasi dan menggunakan kembali logika stateful. Contohnya, `useBuildings.js` mungkin berisi logika untuk mengambil (fetch), menambah, dan mengubah data gedung dari backend.
- **`src/services`**: Bertanggung jawab sebagai jembatan antara frontend dan backend. File seperti `buildingApi.js` berisi fungsi-fungsi yang melakukan panggilan HTTP (misalnya menggunakan `fetch` atau `axios`) ke Cloud Functions yang sesuai.
- **`src/contexts`**: Digunakan untuk menyediakan data atau state secara global ke seluruh pohon komponen tanpa perlu melewati props secara manual. Contohnya adalah `GlobalRefreshContext` yang kemungkinan digunakan untuk memicu pembaruan data di berbagai komponen secara bersamaan.

3.2. Backend (`/functions`)

- **`index.js`**: Titik masuk utama untuk semua Cloud Functions. File ini mengimpor dan mengekspor semua fungsi yang didefinisikan di dalam direktori `src`.
- **`src/`**: Setiap file `.js` di dalam direktori ini (misalnya `buildings.js`, `customers.js`, `orders.js`) mendefinisikan satu atau lebih Cloud Functions yang terkait dengan fitur tersebut. Umumnya, ini adalah fungsi

HTTP yang berfungsi sebagai endpoint API untuk operasi CRUD (Create, Read, Update, Delete).
- **`src/authTriggers.js`**: File ini kemungkinan berisi fungsi yang dipicu oleh event otentikasi Firebase, seperti saat pengguna baru dibuat (`onCreate`) atau dihapus (`onDelete`).
- **`src/database.js`**: Berisi fungsi-fungsi pembantu untuk berinteraksi dengan Firestore, menyederhanakan query dan manipulasi data.

4. Alur Kerja Umum (Contoh: Menambah Gedung Baru)

1. **Pengguna** berinteraksi dengan UI di `frontend/src/components/buildings/Buildings.jsx`.
2. Saat tombol "Tambah Gedung" diklik, sebuah modal (`BuildingModal.jsx`) akan muncul.
3. Setelah mengisi form dan menekan "Simpan", fungsi yang ada di dalam komponen akan memanggil *custom hook* atau *service* yang relevan, misalnya `addBuilding()` dari `frontend/src/services/buildingApi.js`.
4. Fungsi `addBuilding()` melakukan panggilan `fetch` ke endpoint API backend, misalnya `https://<region>-<project-id>.cloudfunctions.net/api/buildings`.
5. Permintaan ini diterima oleh Cloud Function yang didefinisikan di `functions/src/buildings.js`.
6. Fungsi tersebut memvalidasi data yang masuk, kemudian menyimpannya sebagai dokumen baru di koleksi `buildings` di Firestore.
7. Setelah berhasil, fungsi akan mengirimkan respons sukses kembali ke frontend.
8. Frontend menerima respons, menutup modal, dan memperbarui daftar gedung yang ditampilkan di layar, sering kali dipicu oleh `GlobalRefreshContext`.

5. Instalasi dan Menjalankan Proyek

Prasyarat

- Node.js (v18 atau lebih baru)
- Firebase CLI (`npm install -g firebase-tools`)

Instalasi Dependensi

Proyek ini memiliki dua `package.json`, sehingga instalasi perlu dilakukan di kedua direktori.

```
# Instal dependensi untuk frontend
cd frontend
npm install

# Kembali ke root dan instal dependensi untuk functions
cd ../functions
npm install
```

Menjalankan Secara Lokal

1. **Menjalankan Frontend (Vite Dev Server):** `bash cd frontend npm run dev` Aplikasi frontend akan tersedia di `http://localhost:5173` atau port lain yang tersedia.
2. **Menjalankan Backend (Firebase Emulator):** Untuk menjalankan backend secara lokal, gunakan Firebase Emulator Suite. `bash # Dari direktori root firebase emulators:start` Emulator akan menjalankan functions, firestore, dan layanan lainnya secara lokal. Frontend perlu dikonfigurasi untuk menunjuk ke emulator saat dalam mode pengembangan.

Deploy ke Firebase

Development Environment Untuk menjalankan dalam mode development dengan emulator:

```
# Start Firebase emulators
firebase emulators:start --project demo-unionspace-crm

# Di terminal terpisah, jalankan frontend
cd frontend
npm run dev
```

Production Deployment Untuk mendeploy ke production environment:

1. **Build Frontend untuk Production**

   ```
   cd frontend
   npm run build
   ```

2. **Deploy ke Firebase Production**

   ```
   # Kembali ke root project
   cd ..

   # Deploy semua (hosting + functions)
   firebase deploy --project unionspace-w9v242

   # Atau deploy terpisah:
   firebase deploy --only functions --project unionspace-w9v242
   firebase deploy --only hosting --project unionspace-w9v242
   ```

3. **Quick Production Deploy (Single Command)**

   ```
   # Build frontend + deploy everything
   cd frontend && npm run build && cd .. && firebase deploy --project unionspace-w9v242
   ```

Environment Configuration Proyek ini menggunakan konfigurasi environment yang dynamic:

Development Environment: - Frontend: File `.env.development` - `VITE_FIREBASE_PROJECT_ID=demo-unionspace-crm` - `VITE_API_BASE="http://localhost:5555/demo-unio` - Backend: Firebase Emulator dengan project `demo-unionspace-crm`

Production Environment: - Frontend: File `.env.production` - `VITE_FIREBASE_PROJECT_ID=unionspace-w9v242` - `VITE_API_BASE="/api"` (menggunakan Firebase Hosting rewrites) - Backend: Firebase Functions dengan project `unionspace-w9v242`

Environment Switching Commands Development:

```
# Start emulators
firebase emulators:start --project demo-unionspace-crm

# Frontend dev server
cd frontend && npm run dev
```

Production Build Test:

```
# Build frontend
cd frontend && npm run build

# Serve production build locally
npm run preview

# Test with production Firebase
firebase hosting:channel:deploy preview --project unionspace-w9v242
```

Monitoring & Verification Setelah deploy production, verifikasi: - **Frontend**: `https://unionspace-w9v242.web.app` - **API**: `https://unionspace-w9v242.web.app/api/dashboar` - **Functions**: Firebase Console > Functions - **Database**: Firebase Console > Firestore

Rollback (jika ada masalah)

```
# Lihat deployment history
firebase hosting:releases:list --project unionspace-w9v242

# Rollback ke versi sebelumnya
firebase hosting:releases:rollback <RELEASE_ID> --project unionspace-w9v242
```

Firebase Hosting Rewrites File `firebase.json` sudah dikonfigurasi untuk redirect API calls di production:

```
"rewrites": [
  { "source": "/api/dashboard/**", "function": "dashboard" },
```

```
  { "source": "/api/spaces/**", "function": "spaces" },
  { "source": "/api/orders/**", "function": "orders" },
  { "source": "/api/customers/**", "function": "customers" },
  // ... dll
  { "source": "**", "destination": "/index.html" }
]
```

6. Rekomendasi

1. **Buat `README.md`**: Tambahkan file `README.md` di direktori root yang berisi instruksi instalasi singkat dan deskripsi proyek.
2. **Manajemen Variabel Lingkungan**: Formalisasikan penggunaan variabel lingkungan dengan membuat file `.env` dari `env.example` di direktori `frontend` untuk menyimpan kunci API dan konfigurasi lainnya.
3. **Pengujian**: Kembangkan strategi pengujian otomatis (unit test, integration test) untuk `frontend` dan `functions` untuk memastikan kualitas dan stabilitas kode.