

Technical Test Junior FE

Dashboard Developer Position

Summit Global Teknologi

1. Setup & Installation

- Next.js 14 dengan App Router (wajib)
- TypeScript untuk type safety (disarankan)
- Ant Design (antd) untuk UI components
- Axios untuk HTTP client (wajib untuk API calls)
- API Integration: Frontend harus call backend Next.js sendiri (internal API routes)

2. API Architecture Flow

- Frontend (Axios) → Next.js API Routes → External API (Axios) → Response

Penjelasan:

1. Frontend menggunakan Axios untuk call API routes Next.js (/api/...)
2. Next.js API Routes sebagai proxy/middleware layer
3. API Routes menggunakan Axios untuk call API external yang sebenarnya
4. Response dari external API diteruskan ke frontend

Contoh Implementation:

Frontend Call:

typescript

```
const fetchProducts = async ( ) => {  
    const response = await axios.get('/api/products')  
    return response.data  
}
```

API Routes (Proxy):

typescript

// app/api/products/route.ts

```
export async function GET( ) {  
  try {  
    const response = await  
      axios.get('https://external-api.com/api/web/v1/products')  
    return NextResponse.json(response.data)  
  } catch (error) {  
    return NextResponse.json({ error: 'Failed to fetch' }, { status: 500 })  
  }  
}
```

3. Product Management Features

3.1 Data Interface

typescript

```
interface Product {  
  product_id: string;  
  product_title: string;  
  product_price: number;  
  product_description?: string;  
  product_image?: string;  
  product_category?: string;  
  created_timestamp: string;  
  updated_timestamp: string;  
}  
  
interface ProductListParams {  
  page: number; // Current page number  
  limit: number; // Items per page  
  offset: number; // Calculate from page & limit  
  search?: string; // Search term  
}
```

3.2 API Endpoints yang Harus Diproxy

- GET /api/web/v1/products - List products dengan pagination & search
- GET /api/web/v1/product - Get single product (dengan query product_id)
- POST /api/web/v1/product - Create new product
- PUT /api/web/v1/product - Update existing product

3.3 Frontend Features

Product List Page (app/products/page.tsx)

- Antd Table dengan columns: Product Title, Price (format currency), Category, Description
- (truncated), Actions (Edit, Delete)
- Pagination dengan Antd Pagination component
- Search Box dengan real-time search capability (debounce 300ms)
- Create Button untuk open modal create product

Search Functionality

- Search akan mencari di 3 field: product_title , product_description , product_category

Modal Forms (dalam 1 file, 1 modal)

- Create/Edit Modal: Form untuk create dan update product
- Form fields: Product Title (required), Price (required, number), Description (textarea, optional),
- Category (optional), Image URL (optional)
- Modal type ditentukan berdasarkan props/state

3.4 File Structure

/app

 /products

 page.tsx // Main products page + modal

 /api

 /products

 route.ts// Handle GET /products

 /product

 route.ts // Handle GET, POST, PUT /product

3.5 Required Antd Components

- Table
- Pagination
- Input.Search
- Modal
- Form
- Button
- Space
- Typography

4. Technical Requirements

4.1 State Management

- React hooks (useState, useEffect) untuk state management
- Loading states untuk API calls
- Error handling dengan proper error messages
- Form state management dengan Antd Form

4.2 Search Implementation

- Debounce search input (300ms delay)
- Reset pagination ke page 1 saat search berubah
- Search parameter dikirim ke backend

5. Bonus Points - Firebase Integration

5.1 Firebase Setup

- Firebase Authentication untuk login system
- React Context untuk authentication state management

5.2 Authentication Context

```
typescript
interface AuthContext {
  user: User | null;
  logout: ( ) => Promise<void>;
  getToken: ( ) => Promise<string | null>;
}
```

5.3 Pages & Authentication Flow

- Login Page (app/login/page.tsx): Email/password form, auto redirect ke /products setelah login
- Protected Routes: /products require authentication, auto redirect ke /login jika belum login
- Token Management: getToken() dari context untuk authorization header di API calls (token dikirim dari client-side/CSR)
- Auth State Check: Monitor Firebase auth state untuk maintain user session

5.4 API Authorization

- Setiap API call harus include authorization token menggunakan getToken() dari context.

6. Deliverables

- Working Next.js 14 application dengan TypeScript
- Product list page dengan Antd Table, search, dan pagination
- Create/Edit modal dengan form validation
- API routes sebagai proxy ke external API
- Proper error handling dan loading states
- **Bonus Points:** Firebase authentication dengan protected routes

7. Backend API & Setup

7.1 Backend Test API

- Zip Download:

7.2 Cara Menjalankan

- Development tanpa Firebase: yarn dev atau npm run dev
- Development dengan Firebase: yarn dev:firebase atau npm run dev:firebase

7.3 Informasi Lengkap

- Untuk informasi lebih detail tentang API endpoints, authentication, dan setup, silakan baca
- README.md di repository backend.