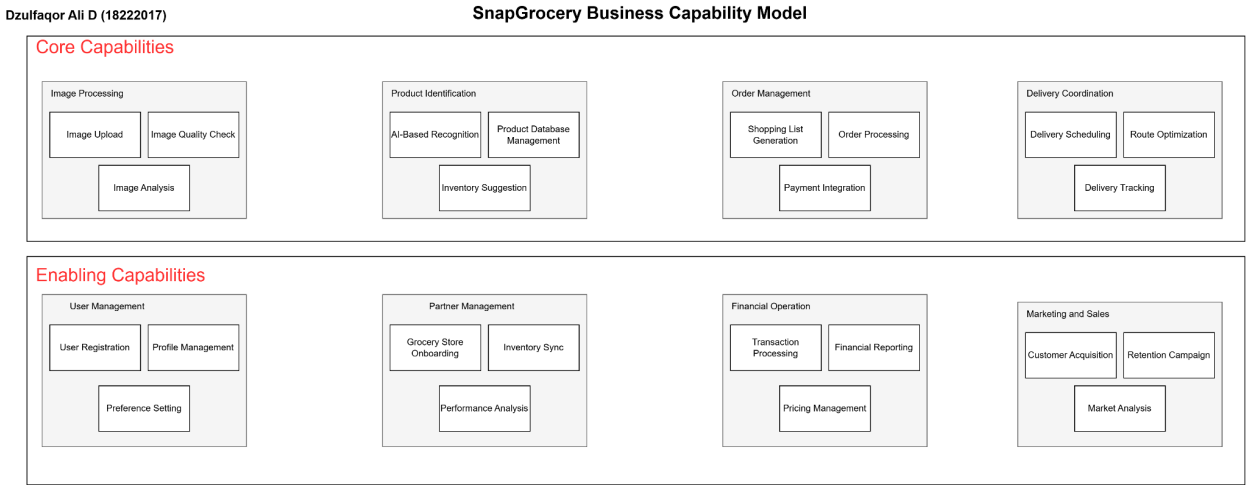
SnapGrocery - Layanan Belanja Bahan Makanan Berbasis Foto

Oleh : Dzulfaqor Ali Dipangegara (18222017)

SnapGrocery adalah aplikasi yang memungkinkan pengguna untuk berbelanja bahan makanan hanya dengan mengirimkan foto isi kulkas mereka. Sistem AI akan menganalisis foto, mengidentifikasi item yang perlu diisi ulang, dan menyusun daftar belanja otomatis yang dapat dipesan langsung melalui aplikasi.

SnapGrocery adalah bisnis inovatif yang menggabungkan teknologi AI dengan kebutuhan sehari-hari berbelanja bahan makanan. Keunikannya terletak pada kemudahan penggunaan - pengguna hanya perlu memotret isi kulkas mereka, dan sistem akan melakukan sisanya. Ini menyelesaikan masalah umum lupa membeli barang-barang penting dan menghemat waktu dalam menyusun daftar belanja.

Business Capability Map

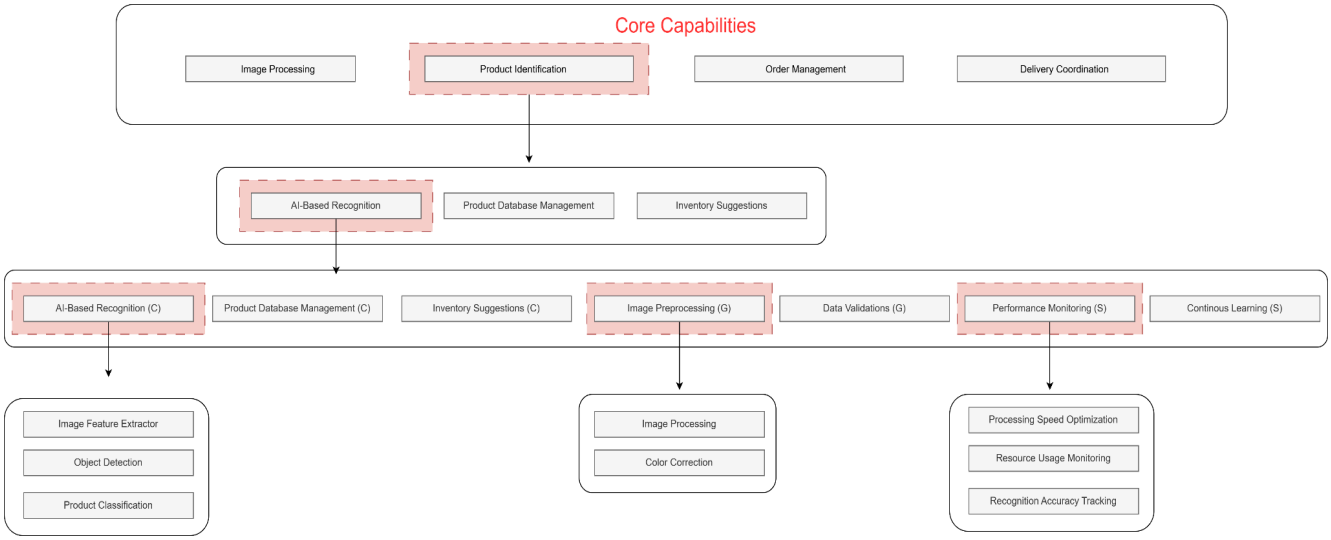


Business Capability Map (BCM) menggambarkan kemampuan utama yang dibutuhkan SnapGrocery untuk berfungsi dengan baik. BCM SnapGrocery terdiri dari dua bagian utama:

- **Core Capabilities**: Ini adalah kemampuan inti yang berkaitan langsung dengan proses pengenalan dan pemesanan bahan makanan berbasis foto. Beberapa kemampuan inti termasuk:
    - **Image Processing**: Kemampuan untuk mengolah dan menganalisis gambar yang diunggah pengguna.
    - **AI-Based Product Identification**: Kemampuan untuk mengenali produk dari gambar melalui teknologi kecerdasan buatan (AI).

    - **Order Management**: Pengelolaan pemesanan pengguna, mulai dari konfirmasi hingga penyelesaian transaksi.
- **Enabling Capabilities**: Ini adalah kemampuan yang mendukung proses inti, seperti:
    - **Data Management**: Kemampuan untuk mengelola data pengguna dan hasil analisis.
    - **Partner Integration**: Menghubungkan SnapGrocery dengan inventaris dan data produk dari mitra toko untuk memastikan ketersediaan produk.
    - **Notification Service**: Memberikan notifikasi kepada pengguna mengenai status pesanan mereka.

Dekomposisi Subdomain



1. Subdomain Core

Subdomain core adalah bagian utama yang mendukung fungsi inti aplikasi SnapGrocery, yaitu pengenalan produk dari gambar yang diunggah dan pengelolaan pesanan.

- **AI-Based Recognition**:
    - **Image Feature Extractor**: Mengekstraksi fitur gambar yang penting untuk memudahkan AI mengenali produk.
    - **Object Detection**: Mendeteksi objek di dalam gambar yang diunggah pengguna. Algoritma ini mengenali produk-produk spesifik di dalam foto.
    - **Product Classification**: Mengklasifikasikan objek yang terdeteksi ke dalam kategori produk (misalnya, sayuran, buah-buahan, daging).
- **Product Database Management**:
    - **Product Catalog Updates**: Memastikan bahwa katalog produk yang ada selalu diperbarui dan sinkron dengan data terbaru dari mitra toko.

    - **Price Information Management**: Mengelola informasi harga produk yang ada di dalam katalog, memberikan pengguna harga yang akurat.
- **Inventory Suggestions**:
    - **Stock Level Predictor**: Memprediksi ketersediaan stok untuk memastikan bahwa pengguna tidak memesan produk yang stoknya habis.
    - **Replacement/Reservation**: Menyediakan opsi untuk mengganti produk yang stoknya habis atau melakukan reservasi produk ketika stok tersedia kembali.

2. Subdomain General

Subdomain ini mendukung proses utama dengan melakukan pengolahan dan validasi data serta memastikan sistem berjalan dengan baik.

- **Image Preprocessing**:
    - **Image Processing**: Memproses dan mengoptimalkan gambar yang diunggah oleh pengguna, sehingga kualitasnya lebih baik untuk dideteksi oleh AI.
    - **Color Correction**: Mengoreksi warna pada gambar untuk memastikan produk yang dikenali memiliki warna yang sesuai dengan kenyataan.
- **Data Validation**:
    - **Input Data Validator**: Memeriksa validitas data yang masuk, termasuk gambar yang diunggah dan data produk dari inventaris toko.
    - **Output Consistency Check**: Memastikan data yang dikeluarkan oleh sistem, seperti daftar produk, konsisten dengan inventaris mitra toko.
    - **Error Handling**: Menangani kesalahan yang terjadi selama proses pengenalan produk atau pengelolaan pesanan.
- **Authentication**:
    - **User Registration**: Pengguna dapat membuat akun baru menggunakan email, nomor telepon, atau media sosial yang terhubung, memastikan mereka memiliki kredensial sah untuk mengakses aplikasi.
    - **Login/Logout**: Pengguna harus login menggunakan kredensial mereka untuk mengakses fungsi inti aplikasi. Sesi pengguna akan dikelola dengan aman, memungkinkan pengguna untuk keluar dengan mudah.
    - **Token-based Authentication (JWT)**: Sistem akan menggunakan **JSON Web Token (JWT)** untuk mengamankan sesi login pengguna, sehingga mereka dapat mengakses API backend secara aman.

3. Subdomain Supporting

Subdomain pendukung bertugas untuk menjaga kinerja dan perkembangan sistem secara keseluruhan, termasuk pelatihan ulang model AI dan pemantauan kinerja.

- **Performance Monitoring**:
    - **Processing Speed Optimization**: Memastikan bahwa proses pengenalan gambar dan pengelolaan pesanan berjalan dengan cepat dan efisien.

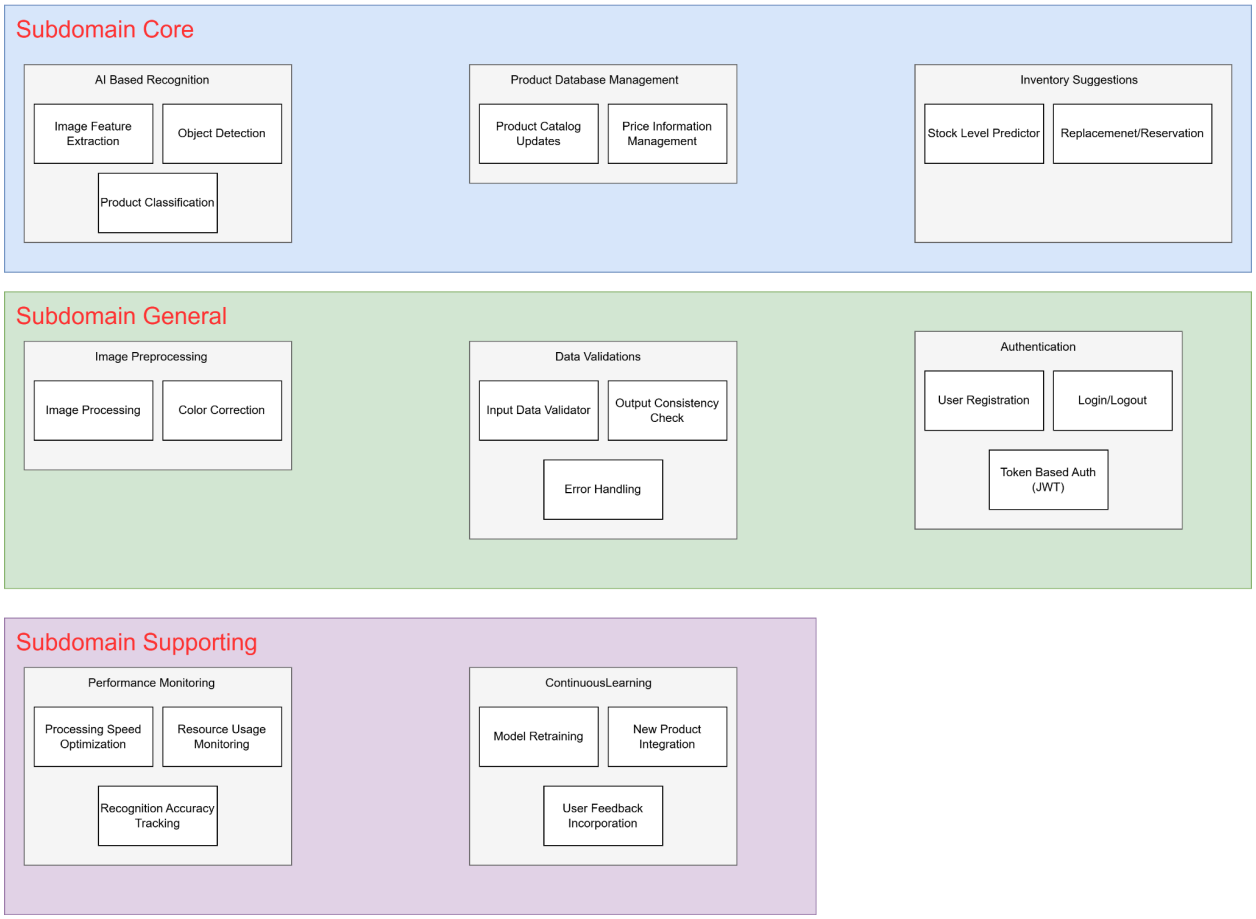
    - **Resource Usage Monitoring**: Memantau penggunaan sumber daya sistem seperti CPU, memori, dan kapasitas penyimpanan untuk menjaga efisiensi sistem.
    - **Recognition Accuracy Tracking :** Memantau akurasi pengenalan produk oleh sistem AI untuk memastikan bahwa model pengenalan produk terus berfungsi dengan baik. Ini termasuk melacak berapa kali sistem melakukan kesalahan dalam mengenali produk dan tingkat kesuksesan dalam mengenali produk yang tepat.
- **Continuous Learning**:
    - **Model Retraining**: Melakukan pelatihan ulang model AI untuk meningkatkan akurasi pengenalan produk berdasarkan data baru yang diterima.
    - **New Product Integration**: Mengintegrasikan produk baru yang belum ada di katalog produk sebelumnya.
    - **User Feedback Incorporation**: Menggunakan umpan balik dari pengguna untuk memperbaiki sistem dan menambah fungsionalitas baru yang dibutuhkan.

Fungsi dari Masing-masing Subdomain

- **Subdomain Core** berfungsi untuk menjalankan fitur utama SnapGrocery, seperti mengidentifikasi produk dari foto, mengelola katalog produk, dan memastikan ketersediaan produk.
- **Subdomain General** mendukung fungsi inti dengan melakukan validasi data dan pemrosesan gambar untuk memastikan pengenalan produk akurat.
- **Subdomain Supporting** memastikan bahwa sistem dapat berjalan dengan efisien, melakukan pembelajaran berkelanjutan, dan memantau kinerja keseluruhan aplikasi.

Domain dan Subdomain

- **Domain Product Identification** yang dipilih memang menjadi fokus utama karena seluruh alur bergantung pada kemampuan AI untuk mengenali produk dari foto pengguna. Subdomain seperti **Image Processing** dan **Object Detection** sangat relevan karena itu adalah kunci keberhasilan proses identifikasi produk.
- Subdomain seperti **Order Management** dan **Inventory Integration** sangat penting untuk memastikan bahwa setelah produk dikenali, pesanan dapat diproses dengan baik, dan ketersediaan produk diverifikasi.

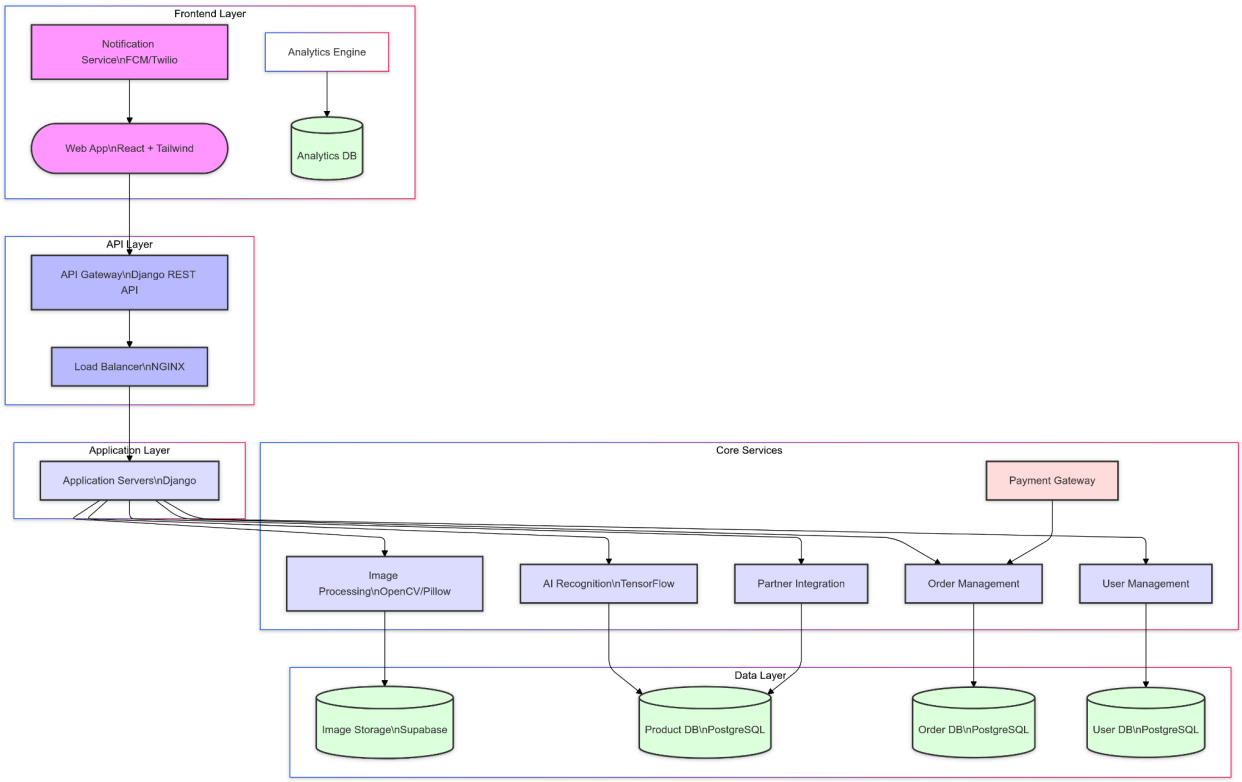


Software Architecture

Arsitektur perangkat lunak SnapGrocery dirancang untuk memastikan semua layanan utama dapat berjalan dengan lancar. Komponen utama dalam arsitektur meliputi:

- **Web App**: Antarmuka utama yang digunakan oleh pengguna untuk mengunggah foto, melihat daftar belanja, dan mengonfirmasi pesanan.
- **API Gateway**: Berfungsi sebagai penghubung antara aplikasi web dan layanan back-end, termasuk pemrosesan gambar dan pengelolaan pesanan.
- **Image Processing Service**: Layanan yang bertugas memproses dan meningkatkan kualitas gambar sebelum diidentifikasi oleh AI.
- **AI Recognition Service**: Layanan ini menggunakan algoritma kecerdasan buatan untuk mengenali produk dalam gambar dan mengirimkan data produk ke layanan pengelolaan pesanan.
- **Order Management Service**: Layanan ini menangani pengelolaan pemesanan, mulai dari membuat hingga mengonfirmasi pesanan.
- **Partner Integration Service**: Layanan yang berfungsi menghubungkan SnapGrocery dengan inventaris dari toko mitra untuk memastikan ketersediaan produk.

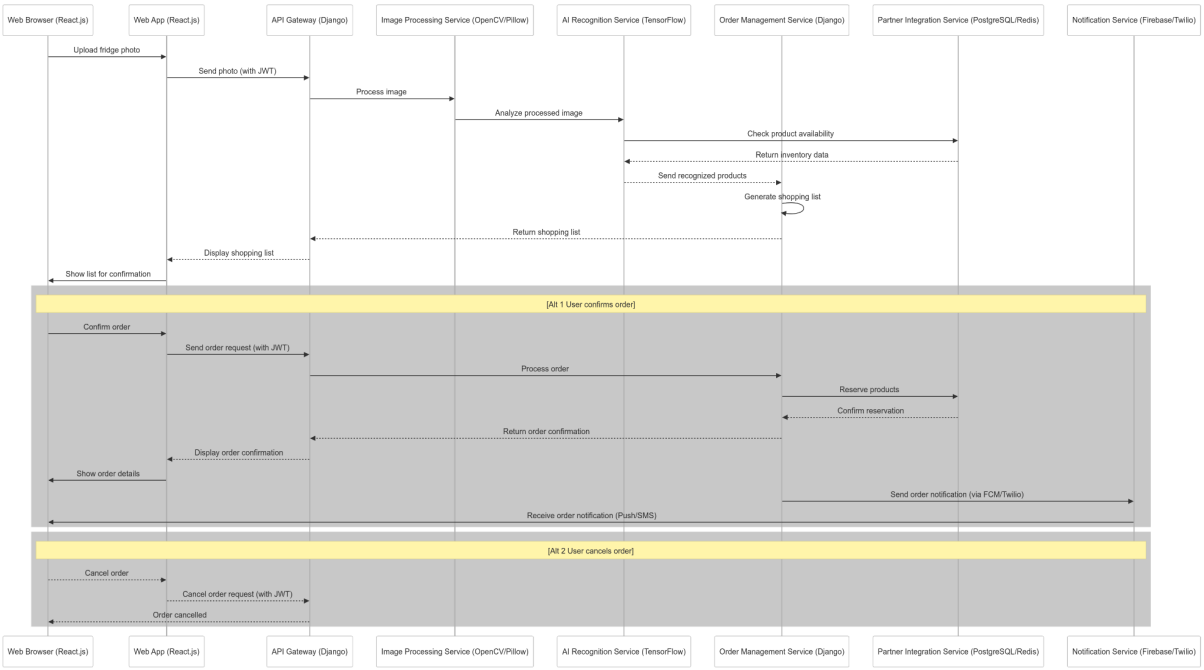
- **Notification Service**: Mengirim notifikasi terkait status pesanan kepada pengguna.



Sequence Diagram

Sequence diagram menunjukkan alur proses yang terjadi di SnapGrocery, mulai dari pengguna mengunggah foto hingga pesanan selesai:

- **Proses Utama**: Pengguna mengunggah foto isi kulkas melalui aplikasi web. Foto ini kemudian dikirim ke API Gateway dan diproses oleh **Image Processing Service**. Setelah gambar diolah, **AI Recognition Service** akan mengenali produk dalam gambar dan mengirimkan daftar produk yang teridentifikasi. Sistem kemudian memeriksa ketersediaan produk di inventaris toko mitra melalui **Order Management Service** dan mengembalikan daftar belanja ke pengguna untuk dikonfirmasi.
- **Alur Alternatif**: Pada tahap konfirmasi, pengguna memiliki opsi untuk membatalkan pesanan. Jika pesanan dibatalkan, sistem akan memproses pembatalan melalui **Order Management Service** dan memberikan notifikasi pembatalan kepada pengguna.



Penjelasan Layered:

1. **User Interface Layer:**
    - **Web Browser**: Entry point dimana pengguna berinteraksi dengan aplikasi
    - **Web App (React.js)**: Antarmuka pengguna yang menangani:
        - Upload foto kulkas
        - Menampilkan daftar belanja
        - Konfirmasi atau pembatalan pesanan
        - Menampilkan notifikasi
2. **API Layer:**
    - **API Gateway (Django)**:
        - Menangani autentikasi dengan JWT
        - Routing request ke service yang sesuai
        - Mengkoordinasikan komunikasi antar service
        - Mengembalikan response ke Web App
3. **Service Layer:**
    - **Image Processing Service (OpenCV/Pillow):**
        - Menerima dan memproses foto dari API Gateway
        - Mengoptimalkan gambar untuk analisis AI
    - **AI Recognition Service (TensorFlow):**
        - Menganalisis gambar yang telah diproses
        - Mengidentifikasi produk dalam foto
        - Berkomunikasi dengan Partner Integration Service
    - **Order Management Service (Django):**
        - Membuat daftar belanja berdasarkan produk yang dikenali

        - Memproses konfirmasi pesanan
        - Menginisiasi notifikasi pesanan
    - **Partner Integration Service (PostgreSQL/Redis):**
        - Mengecek ketersediaan produk
        - Mengelola reservasi produk
        - Sinkronisasi dengan inventaris partner
    - **Notification Service (Firebase/Twilio):**
        - Mengirim notifikasi push via Firebase Cloud Messaging
        - Mengirim SMS via Twilio
        - Memberikan update status pesanan
4. **Data Flow Layer:**
    - **Primary Flow**:
        - Upload foto → Proses gambar → Analisis AI → Cek ketersediaan → Generate daftar belanja
        - Konfirmasi pesanan → Proses order → Reservasi produk → Kirim notifikasi
    - **Alternative Flow**:
        - Pembatalan pesanan: Cancel request → Konfirmasi pembatalan
        - Error handling: Notifikasi kegagalan proses di setiap tahap

Arsitektur Perangkat Lunak

Arsitektur perangkat lunak dirancang untuk mendukung fungsi utama SnapGrocery:

- Web App: Antarmuka utama pengguna.
- Image Processing Service: Memproses dan mengoptimalkan foto yang diunggah.
- AI Recognition Service: Mengidentifikasi produk dalam foto.
- Order Management Service: Mengelola pembuatan dan pemrosesan pesanan.
- Partner Integration Service: Menghubungkan dengan inventaris toko mitra.
- Notification Service: Mengirim pemberitahuan kepada pengguna.
- Analytics Engine: Menganalisis data untuk peningkatan layanan.

TechStack yang Digunakan

1. Frontend (Web App - React.js)

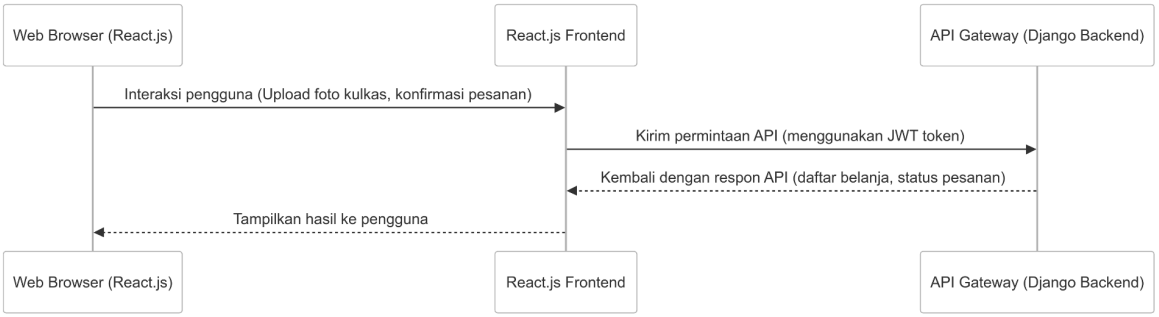
Tech Stack:

- **React.js**: Untuk membangun antarmuka web yang dinamis dan interaktif.
- **Axios**: Untuk komunikasi antara frontend dan backend melalui permintaan **API**, termasuk otentikasi dengan **JWT**.
- **Tailwind CSS**: Untuk styling halaman secara modern dan responsif.

Rasionalisasi:

- **React.js** cocok untuk membangun aplikasi **Single Page Application (SPA)** yang responsif dan mudah di-maintain.
- **Axios** memudahkan pengiriman permintaan API ke backend, terutama untuk komunikasi **stateless** dengan token JWT.
- **Tailwind CSS** mempercepat pengembangan UI tanpa menulis kode CSS dari awal.

Frontend Sequence Diagram:



---

2. Backend (Django)

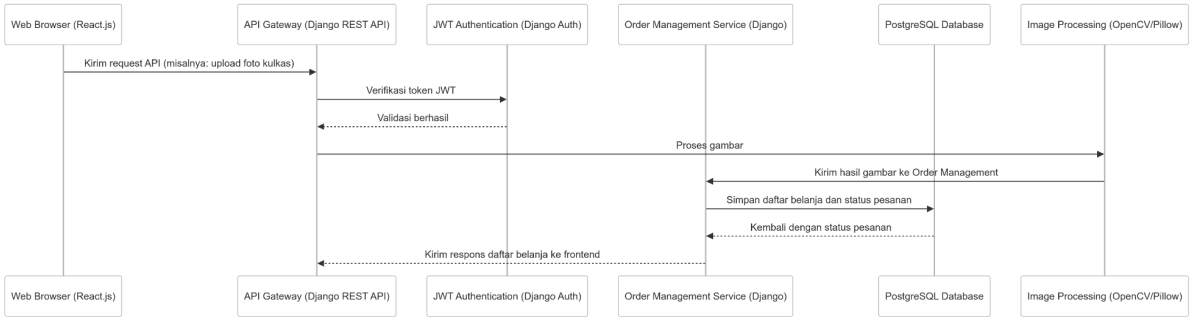
Tech Stack:

- **Django**: Untuk membangun backend yang mengelola API, otentikasi, order management, dan pemrosesan gambar.
- **Django REST Framework (DRF)**: Untuk membangun RESTful API yang melayani frontend.
- **JWT (JSON Web Token)**: Untuk otentikasi berbasis token, memastikan setiap permintaan API diautentikasi.
- **OpenCV atau Pillow**: Untuk melakukan pemrosesan gambar yang diunggah pengguna.

Rasionalisasi:

- **Django** sangat kuat untuk pengelolaan backend berbasis Python, mendukung integrasi AI dengan mudah.
- **Django REST Framework (DRF)** memudahkan pembangunan API yang cepat dan aman untuk komunikasi dengan frontend React.js.
- **JWT** menawarkan **stateless authentication**, sehingga setiap permintaan API aman tanpa perlu penyimpanan sesi di server.
- **OpenCV atau Pillow** sangat baik untuk memproses gambar sebelum digunakan untuk pengenalan produk.

Backend Sequence Diagram:



---

3. Autentikasi (JWT Token)

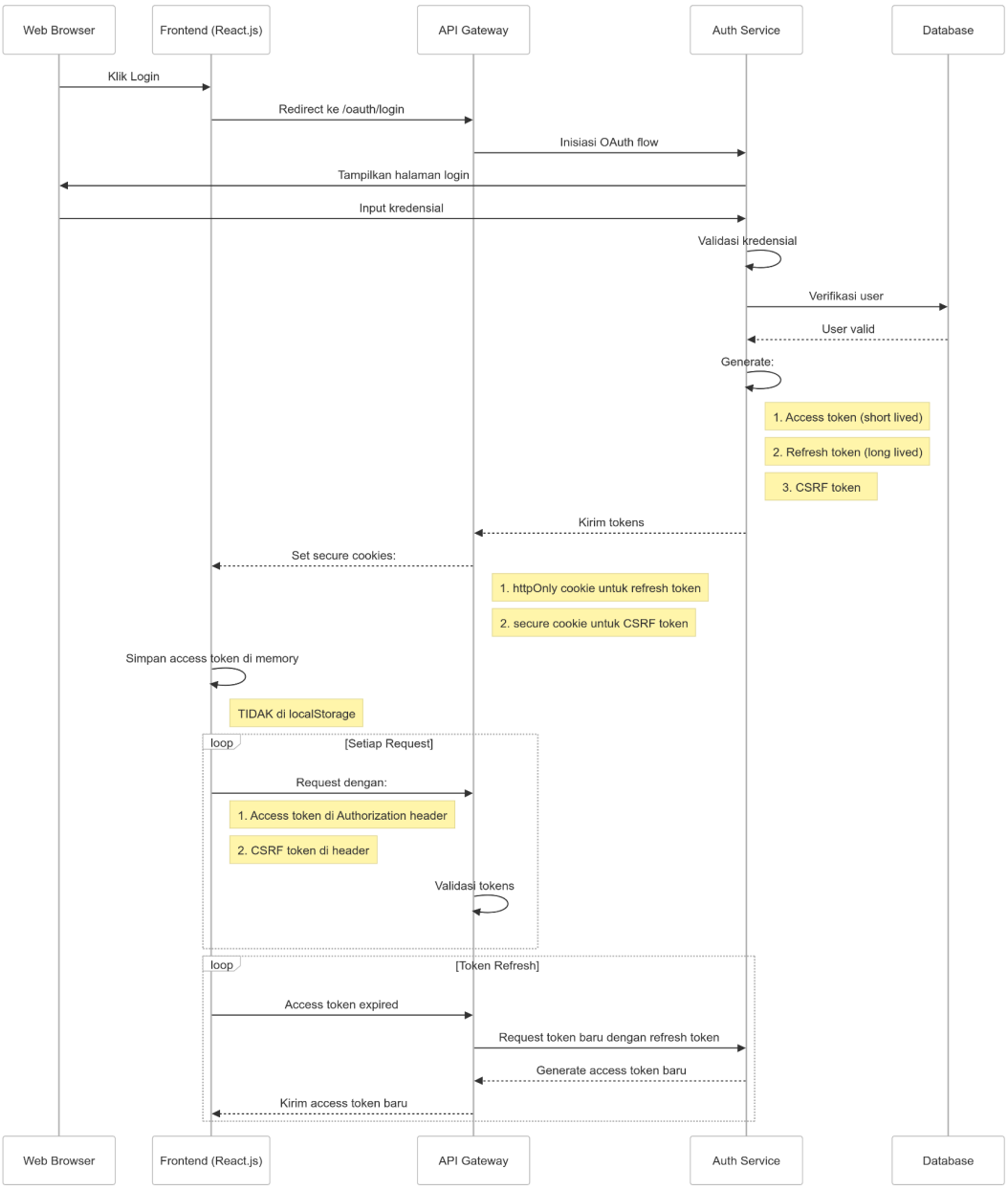
Tech Stack:

- **Supabase Auth**: Untuk otentikasi berbasis email dan password serta penerbitan **JWT** untuk setiap pengguna yang login. Bisa juga menggunakan OAuth

Rasionalisasi:

- **Supabase Auth** sangat mudah diintegrasikan dengan sistem yang sudah ada, memudahkan otentikasi dengan **email dan password**.
- **JWT** memungkinkan autentikasi stateless, memverifikasi setiap permintaan API dengan token tanpa menyimpan sesi di server.

Autentikasi Sequence Diagram:



---

4. Image Processing

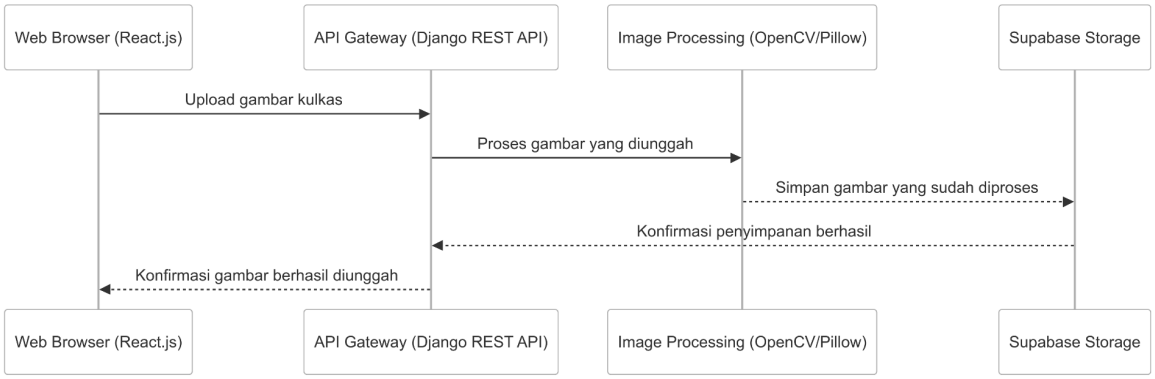
Tech Stack:

- **OpenCV atau Pillow**: Untuk memproses gambar yang diunggah pengguna, seperti resizing, kompresi, atau cropping.
- **Supabase Storage**: Untuk menyimpan gambar kulkas yang diunggah oleh pengguna.

Rasionalisasi:

- **OpenCV atau Pillow** menawarkan fleksibilitas dalam pemrosesan gambar yang diperlukan sebelum digunakan untuk pengenalan produk oleh AI.
- **Supabase Storage** terintegrasi dengan baik ke backend Django, memungkinkan akses gambar yang diunggah dengan mudah dan aman.

Image Processing Sequence Diagram:



---

5. AI Recognition (TensorFlow atau PyTorch)

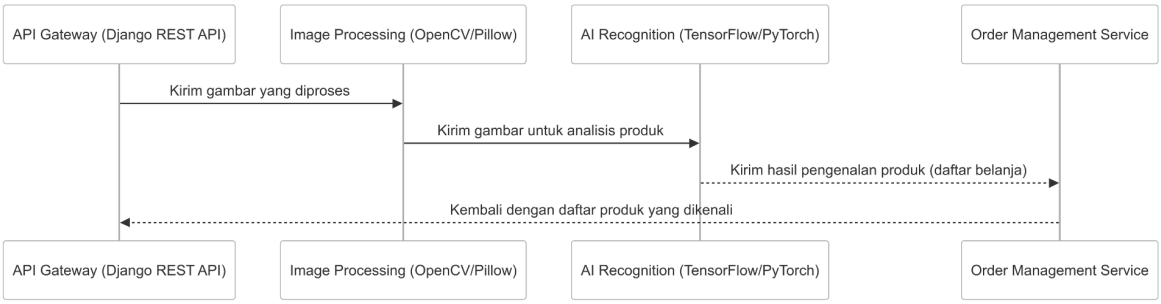
Tech Stack:

- **TensorFlow atau PyTorch**: Untuk menjalankan model **pengenalan produk** berbasis gambar dari foto kulkas yang diunggah.

Rasionalisasi:

- **TensorFlow atau PyTorch** sangat cocok untuk pengembangan dan pengoperasian model **AI** yang kompleks, terutama untuk **pengenalan gambar**. Django yang berbasis Python memungkinkan integrasi yang mulus dengan model AI ini.

AI Recognition Sequence Diagram:



---

6. Database (PostgreSQL)

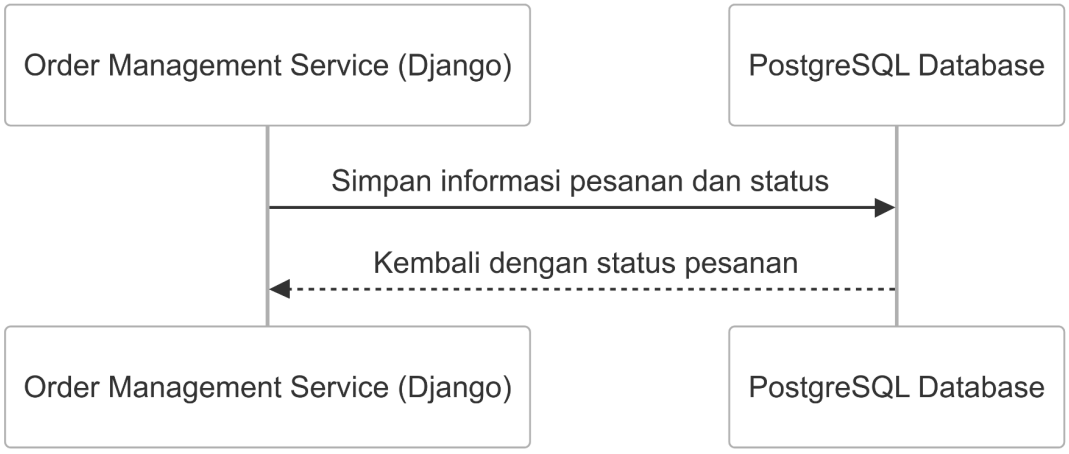
Tech Stack:

- **PostgreSQL**: Untuk menyimpan data produk, inventaris, dan pesanan.

Rasionalisasi:

- **PostgreSQL** sangat handal untuk menyimpan data terstruktur seperti daftar produk, status inventaris, dan pesanan pengguna, dengan dukungan integrasi penuh ke Django.

Database Sequence Diagram:



---

7. Notifikasi (Opsional - Firebase Cloud Messaging atau Twilio)

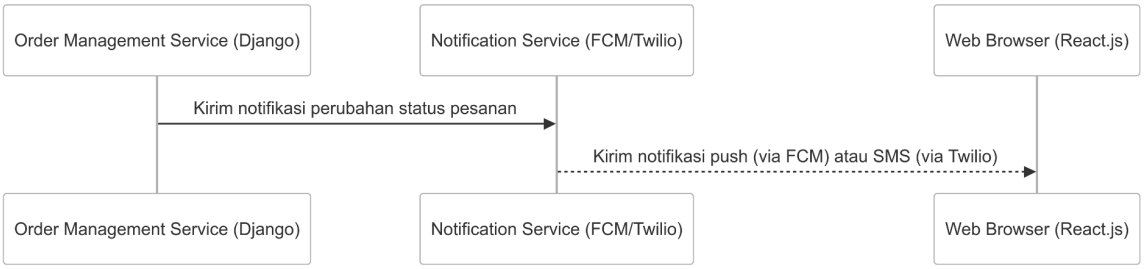
Tech Stack:

- **Firebase Cloud Messaging (FCM)**: Untuk mengirim notifikasi push real-time ke aplikasi web saat status pesanan berubah.
- **Twilio**: Untuk mengirim notifikasi **SMS** atau **email** ke pengguna.

Rasionalisasi:

- **FCM** sangat cocok untuk notifikasi **real-time** yang efisien dan langsung ke aplikasi web.
- **Twilio** menawarkan fleksibilitas untuk mengirim notifikasi berbasis **SMS** atau **email**, terutama untuk menginformasikan status pesanan atau pengiriman produk.

Notifikasi Sequence Diagram:



Kesimpulan Tech Stack dan Integrasi:

- **Frontend (React.js)**: Memberikan antarmuka pengguna yang dinamis dan responsif.
- **Backend (Django)**: Mengelola API, otentikasi, pemrosesan gambar, dan pengelolaan pesanan.
- **Database (PostgreSQL)**: Untuk penyimpanan produk, inventaris, dan pesanan.
- **Storage (Supabase Storage)**: Untuk menyimpan gambar yang diunggah pengguna.
- **Autentikasi (JWT)**: Menggunakan token berbasis JWT yang aman dan **stateless**.
- **Image Processing (OpenCV/Pillow)**: Mengoptimalkan gambar yang diunggah sebelum analisis AI.

Rencana Implementasi dan Lingkungan Pengembangan

Development Environment & Tools

1. Frontend Development Environment

- **IDE & Extensions**
    - Visual Studio Code dengan extensions:
        - ES7 React/Redux/GraphQL/React-Native snippets

        - Tailwind CSS IntelliSense
        - ESLint & Prettier
        - GitLens
        - npm Intellisense
- **Development Requirements**
    - Node.js (v16.x or higher)
    - npm (v8.x or higher)
    - React Developer Tools browser extension
- **Key Libraries & Dependencies**
    - React.js (latest stable)
    - Axios untuk API calls
    - Tailwind CSS untuk styling
    - JWT decode untuk token handling
    - Supabase Auth Client

2. Backend Development Environment

- **IDE & Tools**
    - PyCharm Professional atau VS Code dengan Python extensions
    - Python 3.9 or higher
    - virtualenv untuk isolasi environment
- **Key Components**
    - Django 4.x
    - Django REST Framework
    - OpenCV/Pillow untuk image processing
    - TensorFlow/PyTorch untuk AI recognition
    - PostgreSQL 14
- **Development Tools**
    - Postman untuk API testing
    - pgAdmin 4 untuk database management
    - Docker Desktop untuk containerization

3. Version Control & CI/CD

GitHub repository dengan branch structure:
main

├── develop

│ ├── feature/frontend/*

│ ├── feature/backend/*

│ ├── feature/ai/*

└── hotfix/*

- GitHub Actions untuk automated testing dan deployment
- Pre-commit hooks untuk code formatting dan linting

Project Structure

Frontend Structure

```
snapgrocery-frontend/
├── src/
│   ├── components/
│   │   ├── auth/
│   │   ├── camera/
│   │   ├── product/
│   │   └── order/
│   ├── services/
│   │   ├── api.service.js
│   │   ├── auth.service.js
│   │   └── image.service.js
│   ├── pages/
│   ├── utils/
│   └── App.js
└── public/
```

Backend Structure

```
snapgrocery_backend/
├── api/
│   ├── views/
│   ├── serializers/
│   └── urls.py
├── authentication/
├── image_processing/
│   ├── services/
│   └── utils/
├── ai_recognition/
```

```
│   ├── models/
│   └── services/
└── manage.py
```

Development Setup Steps

1. Frontend Setup

```
# Create and setup project
npx create-react-app snapgrocery-frontend
cd snapgrocery-frontend

# Install dependencies
npm install @supabase/supabase-js
npm install axios
npm install tailwindcss postcss autoprefixer
npm install @headlessui/react @heroicons/react
npm install jwt-decode

# Initialize Tailwind
npx tailwindcss init -p

# Setup environment variables
cp .env.example .env.local
```

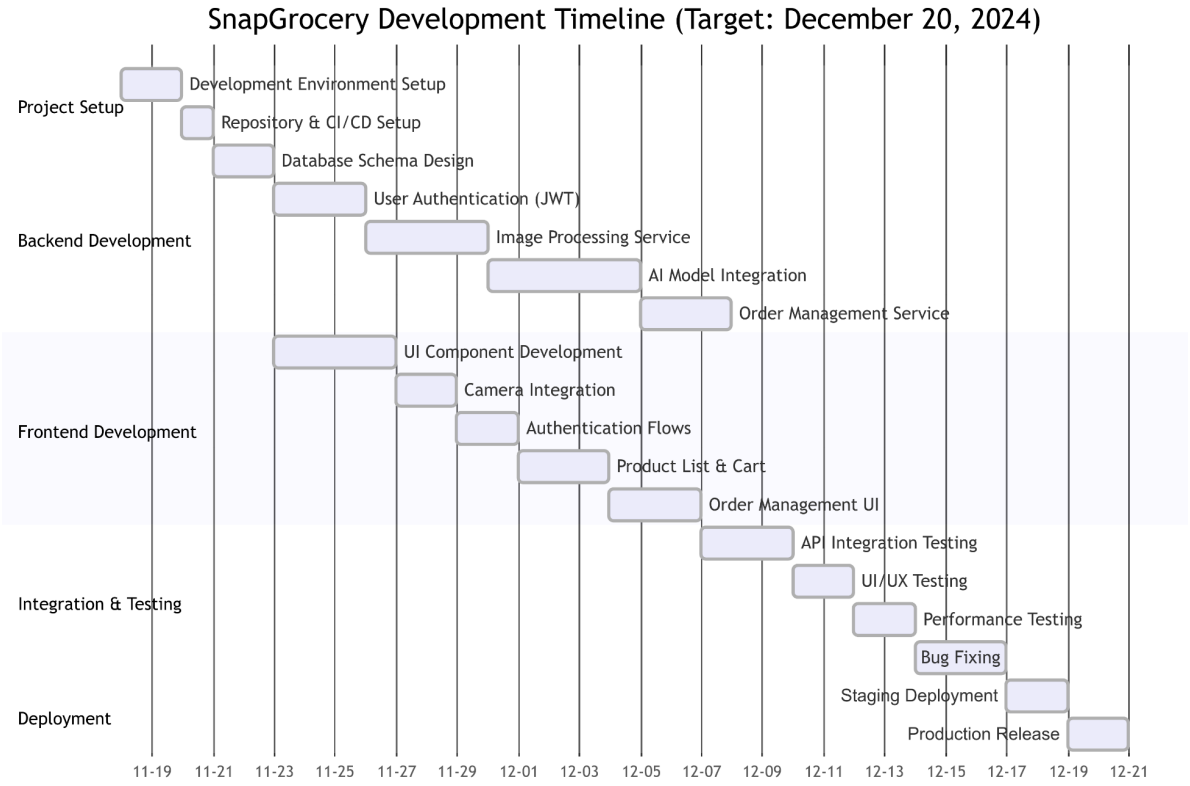
2. Backend Setup

```
# Create virtual environment
python -m venv venv
source venv/bin/activate  # Linux/Mac
.\venv\Scripts\activate   # Windows

# Install dependencies
pip install -r requirements.txt

# Initialize Django project
django-admin startproject snapgrocery_backend
cd snapgrocery_backend

# Create necessary apps
python manage.py startapp api
python manage.py startapp authentication
python manage.py startapp image_processing
```

python manage.py startapp ai_recognition

Setup database
python manage.py makemigrations
python manage.py migrate

Detailed Implementation Timeline



Implementation Phases

Phase 1: Project Setup (Nov 18-20)

- Setup development environments
- Initialize repositories
- Configure CI/CD pipeline
- Setup Docker containers

Phase 2: Backend Development (Nov 21-Dec 7)

1. **Database & Authentication (Nov 21-25)**

    - Design and implement database schema
    - Setup JWT authentication
    - Implement user management API
2. **Image Processing & AI (Nov 26-Dec 4)**
    - Implement image upload and processing
    - Setup AI model integration
    - Develop product recognition system
3. **Order Management (Dec 5-7)**
    - Implement order processing
    - Setup inventory management
    - Create notification system

Phase 3: Frontend Development (Nov 23-Dec 7)

1. **UI Components (Nov 23-26)**
    - Create reusable components
    - Implement responsive design
    - Setup routing
2. **Core Features (Nov 27-Dec 4)**
    - Implement camera integration
    - Create product listing views
    - Build shopping cart functionality
3. **Integration (Dec 5-7)**
    - Connect with backend APIs
    - Implement error handling
    - Setup state management

Phase 4: Testing & Optimization (Dec 7-16)

- Integration testing
- Performance optimization
- User acceptance testing
- Bug fixing and refinements

Phase 5: Deployment (Dec 17-20)

1. **Staging (Dec 17-18)**
    - Deploy to staging environment
    - Conduct final testing
    - Performance monitoring
2. **Production (Dec 19-20)**
    - Production deployment
    - System monitoring setup
    - Final documentation

Production Environment Architecture