**1. Problem Identification**

The need for immediate, accurate medical guidance and efficient management of first-aid supplies

in emergencies is crucial. Traditional first-aid solutions lack the integration of smart technology,

leading to missed opportunities for timely and informed assistance.

**2. Solution Overview**

SmartMedKit is an innovative IoT-enabled first-aid kit equipped with real-time tracking and AI-based

analysis features. It provides instant medical advice through a user-friendly web application,

revolutionizing the way first-aid assistance is administered.

**3. Existing Alternatives**

Conventional first-aid kits are static and lack interactivity, while standalone medical apps do not offer

integration with physical medical supplies. These limitations hinder effective emergency response.

**4. User Demographics**

SmartMedKit is designed for a broad audience, including households, educational institutions, and

workplaces. It serves anyone seeking an efficient, smart emergency response tool.

**5. Features and User Flows**

**Inventory Management:**

Automated tracking of supplies via weight sensors, ensuring items are always stocked.

**AI Analysis:**

A cloud-based service that analyzes injury images to provide specific first-aid advice, enhancing the

quality of care.

**User Interface:**

A simple and intuitive web app allows users to easily navigate through guidance options and supply

management.

**Admin Dashboard:**

Enables administrators to monitor the kit's status, manage inventory, and receive restock alerts.

**Automated Alerts:**

Notifies administrators when supplies are low or nearing expiry, facilitating timely replenishment.

**6. Dependencies**

SmartMedKit's functionality relies on a stable internet connection, compatibility with a range of

mobile devices, access to medical image datasets for AI analysis, and the seamless operation of

integrated sensors.

**7. Technology Stack**

Backend: Python, Flask

Frontend: React

Database: PostgreSQL

Machine Learning: TensorFlow, OpenCV

Containerization: Docker

Cloud Services: AWS/Google Cloud

Hardware: Raspberry Pi OS, weight sensors with HX711 Amps

**API Endpoints Overview**

**Injury Analysis Endpoints**

* **POST /api/injuries/analyze** - Analyze injury from image.

**Admin Authentication and Registration**

* **POST /api/admins/register** - Register new admin.
* **POST /api/admins/login** - Authenticate admin.

**Admin and Kit Management**

* **POST /api/admins/{admin\_id}/kits** - Assign kit to admin.
* **GET /api/admins/{admin\_id}/kits** - List kits for admin.
* **DELETE /api/admins/{admin\_id}/kits/{kit\_id}** - Unassign kit from admin.

**Kit Management**

* **POST /api/kits** - Register new kit.
* **GET /api/kits** - List all kits.
* **GET /api/kits/{kit\_id}** - Get kit details.
* **PUT /api/kits/{kit\_id}** - Update kit details.
* **DELETE /api/kits/{kit\_id}** - Delete a kit.

**Compartment Management**

* **POST /api/kits/{kit\_id}/compartments** - Add compartment to kit.
* **GET /api/kits/{kit\_id}/compartments** - List compartments in kit.
* **GET /api/kits/{kit\_id}/compartments/{compartment\_id}** - Get compartment details.
* **PUT /api/kits/{kit\_id}/compartments/{compartment\_id}** - Update compartment details.
* **DELETE /api/kits/{kit\_id}/compartments/{compartment\_id}** - Remove compartment.

**Inventory Management**

* **POST /api/compartments/{compartment\_id}/items** - Add items to compartment.
* **GET /api/compartments/{compartment\_id}/items** - List items in compartment.
* **GET /api/compartments/{compartment\_id}/items/{item\_id}** - Get item details.
* **PUT /api/compartments/{compartment\_id}/items/{item\_id}** - Update item details.
* **DELETE /api/compartments/{compartment\_id}/items/{item\_id}** - Remove item from compartment.

**Database Tables**

1. **Kits**: Stores each medical kit's location and identifier.
2. **Admins**: Contains details about kit administrators.
3. **AdminKit**: Links admins to kits they manage.
4. **Compartments**: Tracks compartments within kits.
5. **Items**: Details items stored in compartments.
6. **CompartmentItems**: Connects items to compartments, tracks quantity.
7. **LowStockAlerts**: Alerts for low stock in compartments.

CREATE TABLE Kits (

kit\_id SERIAL PRIMARY KEY,

location TEXT

);

CREATE TABLE Admins (

admin\_id SERIAL PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

hashed\_password VARCHAR(255) NOT NULL,

email VARCHAR(255) UNIQUE NOT NULL

);

CREATE TABLE AdminKit (

admin\_id INTEGER REFERENCES Admins(admin\_id),

kit\_id INTEGER REFERENCES Kits(kit\_id),

PRIMARY KEY (admin\_id, kit\_id)

);

CREATE TABLE Compartments (

compartment\_id SERIAL,

kit\_id INTEGER REFERENCES Kits(kit\_id),

PRIMARY KEY (kit\_id, compartment\_id)

);

CREATE TABLE Items (

item\_id SERIAL PRIMARY KEY,

item\_name VARCHAR(255) NOT NULL,

description TEXT

);

CREATE TABLE CompartmentItems (

compartment\_id INTEGER,

kit\_id INTEGER,

item\_id INTEGER REFERENCES Items(item\_id),

quantity INTEGER NOT NULL,

current\_weight FLOAT,

PRIMARY KEY (compartment\_id, kit\_id, item\_id),

FOREIGN KEY (compartment\_id, kit\_id) REFERENCES Compartments(compartment\_id, kit\_id)

);

CREATE TABLE LowStockAlerts (

alert\_id SERIAL PRIMARY KEY,

kit\_id INTEGER REFERENCES Kits(kit\_id),

compartment\_id INTEGER REFERENCES Compartments(compartment\_id),

alert\_time TIMESTAMP NOT NULL,

message TEXT

);