Software Requirements Specification (SRS) for SkinAI

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the functionalities, requirements, and constraints of the **SkinAI** application. **SkinAI** aims to provide users with an AI-powered platform to analyze their skin conditions and provide immediate insights and precautions based on the analysis.

1.2 Scope

SkinAI will be a web application built using the MERN (MongoDB, Express.js, React, Node.js) stack integrated with a machine learning (ML) service for skin condition analysis. The system will allow users to upload images of their skin, receive an analysis report, and obtain preliminary suggestions regarding their skin health.

1.3 Definitions, Acronyms, and Abbreviations

- **AI**: Artificial Intelligence
- **ML**: Machine Learning
- **MERN**: MongoDB, Express.js, React, Node.js
- **SRS**: Software Requirements Specification
- **API**: Application Programming Interface
- **UI**: User Interface

1.4 References

- MERN Stack Documentation
- TensorFlow and PyTorch Documentation
- MongoDB Documentation

1.5 Overview

This SRS document includes detailed descriptions of the system's functionalities, user interface, hardware and software requirements, system constraints, and other specifications.

2. Overall Description

2.1 Product Perspective

SkinAI will function as a standalone web application. It will leverage the MERN stack for the application logic and user interface and integrate with an external machine learning service for image analysis.

2.2 Product Functions

- **User Registration and Authentication**: Secure user registration and login functionalities.
- **Image Upload**: Allow users to upload images of their skin for analysis.
- **ML-Based Analysis**: Analyze uploaded images using a trained machine learning model.
- **Results Display**: Provide users with detailed analysis reports and preliminary suggestions.
- **History Tracking**: Maintain a record of all past uploads and analysis results for each user.

2.3 User Classes and Characteristics

- **General Users**: Individuals looking to analyze their skin conditions. They may have limited technical knowledge and seek an easy-to-use interface.
- **Administrators**: Individuals responsible for managing the application, ensuring system health, and monitoring user activity.

2.4 Operating Environment

- **Frontend**: Modern web browsers (Chrome, Firefox, Safari, Edge)
- **Backend**: Node.js environment, MongoDB database
- **ML Service**: Python environment with Flask and TensorFlow/PyTorch

2.5 Design and Implementation Constraints

- The application must comply with relevant data privacy and security regulations.
- The ML model must be trained and validated with a suitable dataset to ensure accuracy.
- The system should be scalable to handle an increasing number of users and image uploads.

2.6 Assumptions and Dependencies

- Users have access to a device with a camera or means to upload images.
- The system relies on the availability of external machine learning services for image analysis.
- Internet connectivity is required for accessing the application and its features.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 User Registration and Authentication

- Users shall be able to register an account using their email and password.
- Users shall be able to log in to their account using their email and password.
- The system shall send a verification email upon user registration.

3.1.2 Image Upload

- Users shall be able to upload images of their skin.
- The system shall accept image files in common formats (JPEG, PNG).

3.1.3 ML-Based Analysis

- The system shall send uploaded images to the ML service for analysis.
- The system shall receive and display analysis results, including identified skin conditions and suggested precautions.

3.1.4 Results Display

- Users shall be able to view detailed analysis reports.
- The system shall provide preliminary suggestions based on the analysis.

3.1.5 History Tracking

- Users shall be able to view a history of their past uploads and analysis results.
- The system shall securely store user data and analysis results.

3.2 Non-Functional Requirements

3.2.1 Performance

- The system shall process and return analysis results within a reasonable time frame (e.g., under 60 seconds).

3.2.2 Usability

- The user interface shall be intuitive and easy to navigate.
- The system shall provide clear instructions and feedback to users.

3.2.3 Security

- The system shall use secure protocols (e.g., HTTPS) for data transmission.
- User data shall be stored securely, and sensitive information (e.g., passwords) shall be encrypted.

3.2.4 Scalability

- The system shall be able to handle a growing number of users and image uploads without performance degradation.

3.2.5 Reliability

- The system shall have a high uptime (e.g., 99.9%) and be resilient to failures.

3.3 Interface Requirements

3.3.1 User Interfaces

- **Login/Registration Page**: Forms for user authentication.
- **Dashboard**: Overview of user activity and options for image upload.
- **Upload Page**: Interface for uploading images and viewing analysis results.
- **History Page**: List of past uploads and analysis reports.

3.3.2 Hardware Interfaces

- No specific hardware requirements other than a device capable of running a modern web browser.

3.3.3 Software Interfaces

- Integration with external ML services for image analysis.
- Use of RESTful APIs for communication between the frontend, backend, and ML services.