SwasthaSetu: Project Dependencies & Technical Specifications

Team- OptiBeamTech

Date: August 2, 2025

1. Overview

This document provides a complete list of all technical dependencies required to build, run, and deploy the **SwasthaSetu** application. The project consists of two primary components:

- **A Flutter-based mobile application (Frontend):** The user-facing app for Android that handles UI, user interactions, and communication with the backend.
- **A Python-based AI server (Backend):** A FastAPI server that hosts the machine learning models and provides API endpoints for diagnosis and AI-driven features.

This document is structured to guide developers in setting up the required environment for both components.

2. Frontend Dependencies (Flutter)

The frontend is built using the Flutter framework. The following table outlines the essential packages managed in the `pubspec.yaml` file.

Package	Version	Purpose in SwasthaSetu
`http`	^1.2.1	Essential for making API calls to the backend and Sarvam AI.
`image_picker`	^1.1.2	Selecting images from gallery or camera.
`flutter_spinkit`	^5.2.1	Provides professional loading animations during AI analysis.
`pdf`	^3.10.8	The core library for generating multi-page, multilingual PDF reports.
`printing`	^5.12.0	Enables the viewing, sharing, and printing of generated PDF reports.
`path_provider`	^2.1.3	Used to find the correct local device path to save the PDF report.
`cupertino_icons`	^1.0.8	Provides a set of standard iOS-style icons.

To install these dependencies, a developer would run `flutter pub get` in the project directory.

3. Backend Dependencies (Python)

The backend AI server is built using Python with the FastAPI framework. The following table details the libraries required for the server, managed via a `requirements.txt` file.

Library	Version	Purpose in SwasthaSetu
`fastapi`	0.111.0	A high-performance web framework for building the API endpoints.
`uvicorn`	0.29.0	The ASGI server that runs the FastAPI application.
`python-multipart`	0.0.9	Required by FastAPI to handle image file uploads.
`torch`	2.3.1	The core PyTorch library for running the trained deep learning models.
`torchvision`	0.18.1	Provides necessary image transformation functions for the models.
`Pillow`	10.3.0	An image processing library for opening and manipulating images.
`numpy`	1.26.4	A fundamental package for numerical computation with image data.
`onnxruntime`	1.18.0	Used for high-performance inference of the optimized Malaria model.
`efficientnet-pytorch`	0.7.1	Required to correctly load the trained EfficientNet model architecture.

To install these dependencies, a developer would run `pip install -r requirements.txt` in a Python environment.

4. Team Contributions Summary

The success of this project was a result of clearly defined roles and a collaborative effort. The following table highlights the key contributions from each team role.

Role	Contribution
Team Lead | AI/ML | Agentic & Gen AI | App Development	Architected the end-to-end system, developed the AI models, built the backend API, and created the complete Flutter application.
Machine Learner | UI/UX | Git	Trained the core machine learning models, designed the app's entire user interface and experience, and managed the project's codebase on GitHub.
Overall Concept & Presentation	Conceptualized the core SwasthaSetu idea and masterfully crafted the compelling narrative and presentations to articulate its vision and impact.

⁂