

Google Summer of Code

Proposal - Google Summer of Code 2025

DocPilot

Mentor: Jaideep

By

Prakhar Singh



Table of contents

GSoC 2025 Proposal

Table of contents

1. Project Summary

2. Project Vision & Motivation

3. Biographical Information

Personal information

Commitments

Past Experiences in Software development

Open source contributions

4. Introduction to the Project

Problem facing due to other softwares

Solution Description

5. Implementation of the Project Goals

Project Objectives

Expected Deliverables

Future Work

6. Implementation Plan

Project Methodology

Architecture of the Project

User Interface(UI) of the Project

Technical Elements

Challenges & Solutions

7. Project Timeline

Deliverables Schedule

Availability

8. Conclusion

Reference links

Why are you the best person to execute this proposal?

Post GSoC plans

Project Summary

Project Idea

Docpilot: A conversational AI powered EMR application that automates prescription generation & integrates OPD management by transcribing doctor-patient consultations in real time, enhancing efficiency & accuracy in healthcare documentation.

Project size - Large (350 hours)

Project Length - 18 weeks

Project Vision & Motivation

Docpilot aims to replace the outdated EMR (Electronic Medical Records) applications, vision for docpilot is simple yet impactful: to create a tool that makes the lives of doctors easier by automating the boring & dull parts of their job, like documentation and prescription writing, so they can focus on what truly matters—caring for their patients.

The motivation behind DocPilot stems from the need to alleviate the burdens faced by healthcare providers due to outdated and inefficient EMR systems. By automating documentation and prescription generation, we aim to reduce physician burnout and allow doctors to focus more on patient care. Our goal is to create a user-friendly tool that enhances efficiency and improves the overall healthcare experience for both providers and patients.

Biographical Information

Personal Information

Name: Prakhar Singh

Email Address: prakharsingh7014@gmail.com

Discord Username: blastoise5961

Phone:

Address:

University: Dr. APJ Abdul Kalam Technical University

Degree: Bachelor of Technology

Year: 3rd

Field of Study: Computer Science Engineering specialization in AI & ML

Date of Enrollment: November 2022

Expected Graduation date: September 2026

Github: <https://github.com/prakharsingh-74>

LinkedIn: <https://www.linkedin.com/in/prakharsingh74/>

Major Courses: DBMS, Computer Networks, Operating system, Data structures and Computer Organisation and Architecture.

Commitments

- **How many hours will you work per week on your GSoC project?**

I am planning to spend 40 - 50 hours or more on the project per week.

- **Other Commitments**

I have no other commitments during the GSoC period.

- **Do you plan to apply for any other organisation for GSoC'25?**

I am only applying to AOSSIE for GSoC'25 and have no plans to contribute to any other organisation

- **If you're selected as a GSoC student, would you like to work on other tasks besides the projects of your choice?**

Yes, I would love to work on other tasks that are not related to my GSoC project.

- **If you're not selected as a GSoC student, would you like to work on the projects as a general contributor?**

Yes, even if I'm not selected as a GSoC participant, I would happily continue working as a general contributor.

- **Would you like to contribute to AOSSIE in the long term, after the GSoC program ends?**

Yes, I would like to contribute to AOSSIE even after GSoC ends.

- **What motivated you the most towards applying for GSoC?**

There are various reasons for which I wanted to apply for GSoC but my main motive was to get recognised as a GSoC participant. Also, the stipend was not a motivating factor but an opportunity to work with a big organisation like AOSSIE was.

- **Other commitments**

I will have my 6th Semester examination from 10th - 30th June but I'm really good at time management as I had completed several personal projects during my previous semester exams.

Past Experiences in Software Development

- **Machine Learning Center of Excellence | (Data Analyst & Frontend Developer) (Oct. 2023 - Dec. 2023)**
 - Performed Exploratory Data Analysis (EDA) on various datasets.
 - Developed models using linear & logistic regression, improving prediction accuracy by 15%.
 - Successfully deployed a Face detection model using Flask and Vercel, reducing inference time by 20% and optimizing the model for real-time face detections with 95% accuracy.
- **eRx | Flutter, Dart, Express js, Firebase - [Github](#)**
 - The standardization and digitalization of the prescription offer several benefits to the involved parties. As all the prescriptions are digitized, the patient does not need to maintain a record of their previous prescriptions as it is readily available in the system. This also ensures that the doctor can look at the previous prescriptions to give an informed diagnosis.

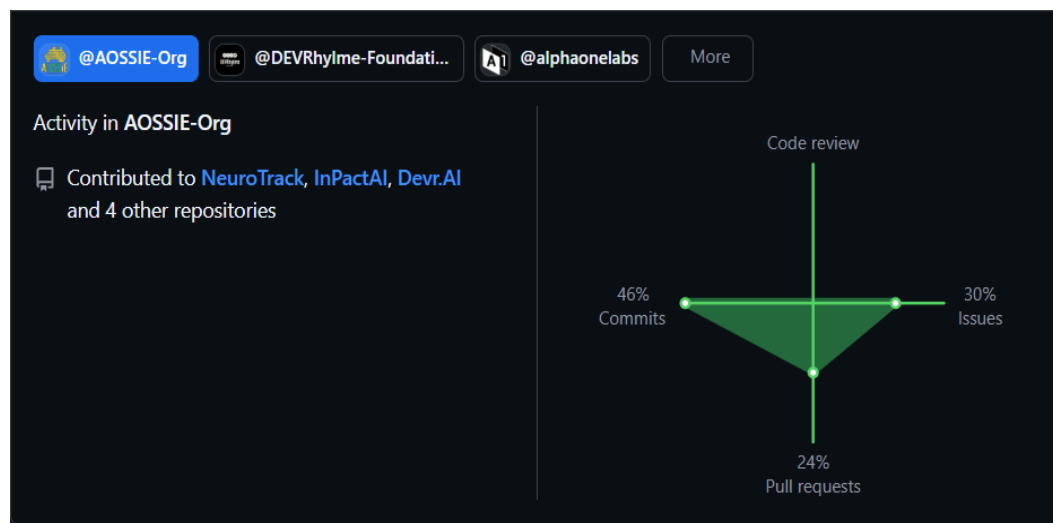
- Optimized API response times by 20% for smoother, uninterrupted sessions, improving the user experience.
 - Optimized application performance and scalability by deploying on AWS cloud infrastructure.
 - **LinkUp | Nextjs, Clerk, Stream, TailwindCSS, Typescript** - [Github Deployed Link](#)
 - Implemented a scalable architecture with server-side rendering in Next.js and optimized deployment using Vercel & efficiently handling up to 40+ users across multiple sessions with fast load times.
 - Integrated Clerk for secure user authentication with multi-factor authentication & Stream API for real-time video streaming & messaging, ensuring low latency, high-quality video calls & role-based access control.
-

Open source contributions

- #33 : Add HomeProvider and screens for Profile, Patients, and Sessions : <https://github.com/AOSSIE-Org/NeuroTrack/pull/33>
- #49 : Therapist Schedule Screen : <https://github.com/AOSSIE-Org/NeuroTrack/pull/49>
- #24 : Created splash and Auth screen : <https://github.com/AOSSIE-Org/NeuroTrack/pull/24>
- #41 : feat: Implement auto-update for project_structure.txt via GitHub Actions : <https://github.com/AOSSIE-Org/InPactAI/pull/41>
- #21 : SR No. 3 improvement of the Home Page & Adding Image : <https://github.com/AOSSIE-Org/InPactAI/pull/21>
- #43 : (SR No. 8) Added Analytics & Support Page in the Dashboard : <https://github.com/AOSSIE-Org/Devr.AI/pull/43>
- #247 : Add Firebase authentication and update routing for user signup and login : <https://github.com/AOSSIE-Org/EduAid/pull/247>

- #76 : [S.NO - 6][ENHANCEMENT] Refactoring Home page :
<https://github.com/AOSSIE-Org/Perspective/pull/76>
- #159 : Auth0 Integration and allow Event data edit button to admin users only :
<https://github.com/DEVRhyme-Foundation/new-website/pull/159>
- #161 : Project Filtering & Search for projects :
<https://github.com/DEVRhyme-Foundation/new-website/pull/161>
- #404 : Landing Page of Pictopy **(Collaborated)** :
<https://github.com/AOSSIE-Org/PictoPy/pull/404>
- #341 : Control & Customization/Image Compressor + Download/Bug Fixes (small case) (SR. NO. 4) **(Collaborated)** :
<https://github.com/AOSSIE-Org/PictoPy/pull/341>

Contributions Graph to AOSSIE



Introduction to the Project

This project aims to change that by introducing DocPilot, a next-generation EMR application powered by conversational AI. DocPilot listens to the natural conversation between doctors and patients during consultations and automatically generates prescriptions that doctors can review, sign, print, and save digitally. By eliminating manual data entry and leveraging cutting-edge AI technology, DocPilot simplifies the documentation process while improving accuracy and efficiency.

Problems facing due to other platforms

- **72% physician** burnout from manual data entry.
- **\$15k-\$70k** implementation costs per clinic.
- **43%** prescription **errors** from manual systems.
- **No native OPD** integration in the current platforms.
- Excessive documentation requirements leading to after-hours work.
- Difficult to use systems overwhelm healthcare providers.

Solutions Description

1. Conversational AI for Real-Time Documentation

DocPilot employs state-of-the-art natural language processing (NLP) & speech-to-text (STT) technologies to capture and analyze conversations between doctors and patients. This feature allows the application to:-

- **Extract Key Information:** Automatically identify and categorize symptoms, diagnoses, medications, and tests mentioned during consultations.
- **Generate Prescriptions:** Create accurate prescription templates that doctors can quickly review, sign, print, or save digitally, significantly reducing the time spent on manual data entry.

2. Integrated OPD Appointment Management

DocPilot includes an OPD appointment management system that streamlines scheduling and patient tracking. This feature offers:

- **Smart Scheduling:** Automated appointment reminders and waitlist management to optimize patient flow.
- **Real-Time Updates:** Instant notifications for both doctors and patients regarding appointment changes or follow-ups.

3. Offline Functionality

Understanding that connectivity may be an issue in some healthcare environments, DocPilot incorporates offline capabilities:

- **Local Data Processing:** The application can function without internet access by utilizing lightweight AI models embedded within the app.
- **Seamless Syncing:** Once reconnected, all data captured offline syncs automatically with the cloud-based system for secure storage and compliance.

Implementation of the Project Goals

Project Objectives & community benefits

1. **Automate Documentation Processes:** Develop a conversational AI-powered EMR application that listens to real-time consultations between healthcare providers and patients, extracting key information to generate accurate prescriptions automatically.
2. **Integrate OPD Appointment Management:** Incorporate a comprehensive outpatient department (OPD) appointment management system that streamlines scheduling, follow-ups, and patient tracking, improving overall clinic efficiency.
3. **Improved Patient Care:** By automating documentation and reducing prescription errors, healthcare providers can focus more on patient interaction, leading to better care outcomes and enhanced patient satisfaction.
4. **Reduced Physician Burnout:** By alleviating the burden of manual data entry, DocPilot aims to decrease the high levels of burnout reported by physicians, allowing them to spend more time on clinical responsibilities rather than administrative tasks.
5. **Cost Savings for Clinics:** The affordability of DocPilot will enable smaller clinics to adopt modern EMR solutions without incurring prohibitive costs, thus democratizing access to advanced healthcare technology.

6. **Enhanced Workflow Efficiency:** The integrated OPD appointment management system will streamline scheduling processes, reducing wait times for patients and improving overall clinic operations.

Expected Deliverables

- A fully functional conversational AI-powered EMR application with real-time documentation capabilities.
- Integrated OPD appointment management features that streamline scheduling processes.
- Comprehensive user documentation detailing application usage and best practices.
- A final report summarizing project outcomes, challenges faced, and recommendations for future enhancements.

Futures work based on the Project

- Expanding language support for diverse patient populations.
- Incorporating machine learning algorithms for predictive analytics in patient care.
- Developing additional modules tailored to specific medical specialties or workflows.

Implementation Plans

Project Methodology

- To meet the project objectives, we will follow an iterative development approach(waterfall approach), starting with requirements gathering, followed by design, implementation, testing, and deployment phases.
- **Planning & Research (Phase 1):**

This Phase is divided into the below points:-

- **Requirements Gathering:** Define technical specifications, including offline AI capabilities and integration with OPD workflows.
- **Database Schema Design:** Use Appwrite's database module for defining structured collections for:
 - 1) Patients data (ID, Name, medical history, contact).
 - 2) Doctors data (ID, specialization, schedule).
 - 3) Appointments (patient ID, doctorID, datetime, status).
 - 4) Prescriptions (patientID, doctorID, medications, notes).
 - 5) Consultations (transcript, diagnosis, AI-extracted entities).
- Setting up **role-based access control (RBAC)** for doctors, patients and admins.
- Establishing relationships between collections to ensure data integrity and facilitate efficient querying.
- **Development (Phase 2):**
 1. **Frontend development:** Divide development into iterative parts focusing on core features of the frontend and make it ready to integrate with the backend. Dividing this Phase into the below points -
 - **Doctor Dashboard:**
 - ◆ Appointment scheduler (sync with backend)
 - ◆ Consultation transcript viewer
 - ◆ Prescription editor with AI suggestions
 - **Patient Portal:**
 - ◆ Secure login (Appwrite Auth)
 - ◆ Appointment booking UI
 - ◆ EMR access (read-only)
 2. **Backend Integration:** Implementing Appwrite and their functions for secure data storage and management for the storage of

sensitive medical data. By Keeping the following points in mind by me, I will going to setting up the Appwrite backend -

- Configure **authentication** (JWT tokens).
- Set up **database collections** with proper indexing.
- RBAC(Role based access control) - Doctor, Admin, Patient
- API's - (a) /appointments
(b) /upload-audio
(c) /generate-transcript
(d) /generate-prescription
- Implement **Cloud Functions** for:
 - (a) Prescription validation.
 - (b) Appointment reminders.
 - (c) Data encryption.

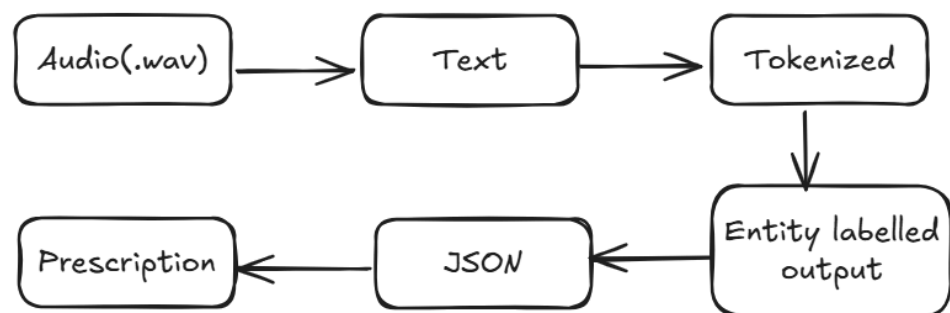
3. **AI Model Optimization & integration:** Use lightweight models like TensorFlow Lite for offline functionality while maintaining high accuracy.

AI/ML Engine Description

- **Speech-to-Text Module:** Wav2Vec 2.0 Lite (8MB) ensures real-time transcription of consultations.
- **NLP Engine:** DistilClinicalBERT extracts symptoms, diagnoses, medications, and tests from conversations with >90% accuracy.
- **Prescription Builder:** Combines rule-based logic with machine learning algorithms to auto-generate prescriptions while checking drug interactions via RxNorm API.
- Fine-tune **DistilClinicalBERT** on medical conversation datasets (i2b2, MIMIC-III) and extracting the below entities -
 - Symptoms
 - Diagnosis
 - Tests

- Medications

- Convert models using **TensorFlow Lite Converter** for on-device inference(for the local deployment process).
- **Datasets (for model training):** MIMIC-III, i2b2 clinical datasets, HealthTap Q&A
- Building a Pipeline for converting **Audio to Prescription**
(Below diagram showcasing the pipeline followup structure)



- **Testing, Deployment & monitoring (Phase 3):**

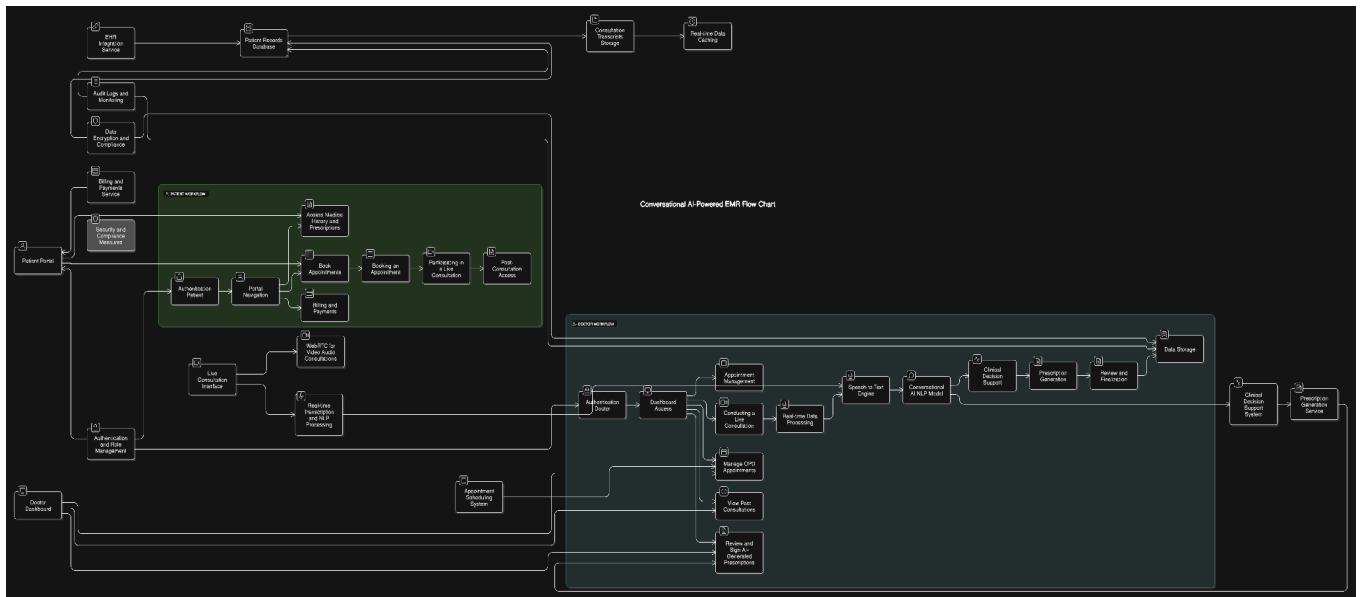
1. **Comprehensive Testing Strategy:**

- **Unit Testing:** Validate individual modules (appointment booking, AI model accuracy) using Flutter Test & Jest.
- **Integration Testing:** Ensure seamless communication between Flutter (frontend), Appwrite (backend), and AI models.
- **Performance Testing:** Simulate high-traffic conditions (100+ concurrent users) to assess latency and stability.

2. **Dockerized Deployment:**

- Containerize the Flutter app, Appwrite backend, and AI models for consistency across environments.
- Use **GitHub Actions CI/CD** for automated builds and deployments.

→ Deploying on a scalable cloud provider (AWS/Azure) with load balancing for high availability.



High level Architecture Diagram

Link(Architecture) -

<https://app.eraser.io/workspace/uH9HKVssowZErmxsQak8?origin=share>

Explanation of the Architecture-

1. Patient Portal:

- Provides patients secure access to their medical history, prescriptions, and post-consultation records.
- Enables appointment booking and payment processing through a user-friendly interface.

2. Doctor Dashboard:

- Allows doctors to access their schedules, manage OPD appointments, and view past consultations seamlessly.
- Serves as the central hub for clinical workflows.

3. Real-Time Data Processing:

- Captures audio during live consultations using WebRTC for video/audio input.
- Processes speech data through a Speech-to-Text Engine and NLP model to extract medical entities like symptoms, diagnoses, and medications.

4. Prescription Generation System:

- Automatically generates prescriptions based on processed consultation data.
- Doctors can review, finalize, sign, and save prescriptions digitally.

5. Clinical Decision Support System:

- Provides AI-driven recommendations for diagnoses and treatments based on extracted consultation data.
- Ensures accuracy and reduces manual errors.

6. EMR Database:

- Stores patient records, consultation logs, prescriptions, and appointment details securely.
- Ensures compliance with healthcare regulations like HIPAA through encryption and secure access controls.

7. Data Encryption & Compliance:

- Protects sensitive patient data with robust encryption mechanisms.
- Adheres to healthcare standards for privacy and security.

8. OPD Appointment Management:

- Streamlines scheduling for outpatient department visits, reducing wait times and improving clinic efficiency.

9. Integration of AI Models:

- Combines conversational AI (NLP) and real-time transcription to automate documentation processes efficiently during consultations.

Technical Details

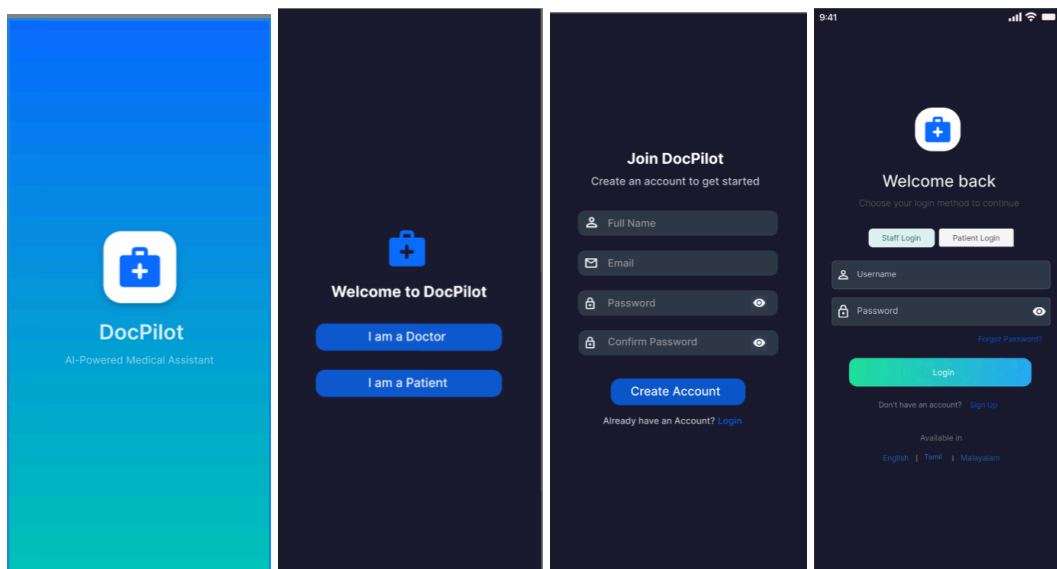
- Frontend - Flutter(Dart)
- Backend - Appwrite (Secure user Authentication & RBAC Access)
- Database - Appwrite database
- AI/ML Engine - TensorFlow Lite, DistilClinicalBERT, Wav2Vec 2.0
- Containerization & Automation - Github Actions & Docker

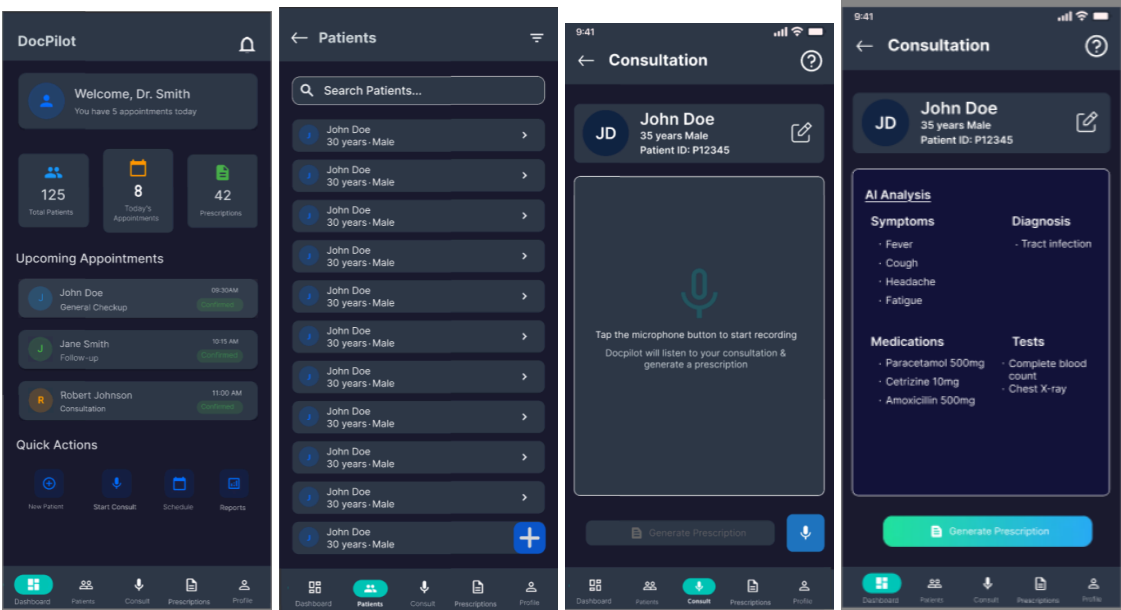
UI Design of the Project:

Figma link -

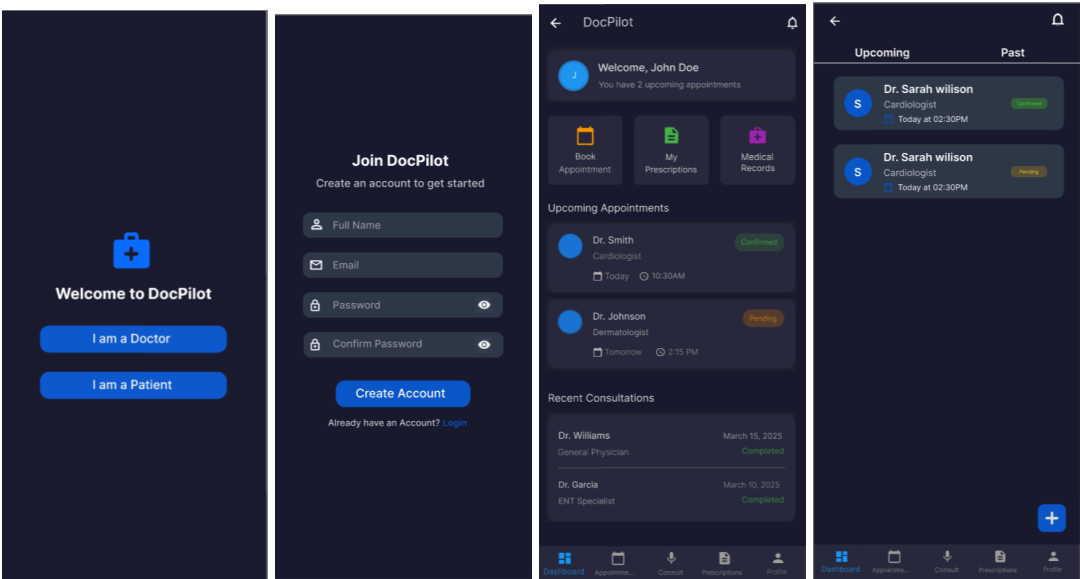
<https://www.figma.com/design/kSKG97AbVEYma2FK2gRWQq/Docpilot?node-id=0-1&t=CPTqErdq4D1AbSFX-1>

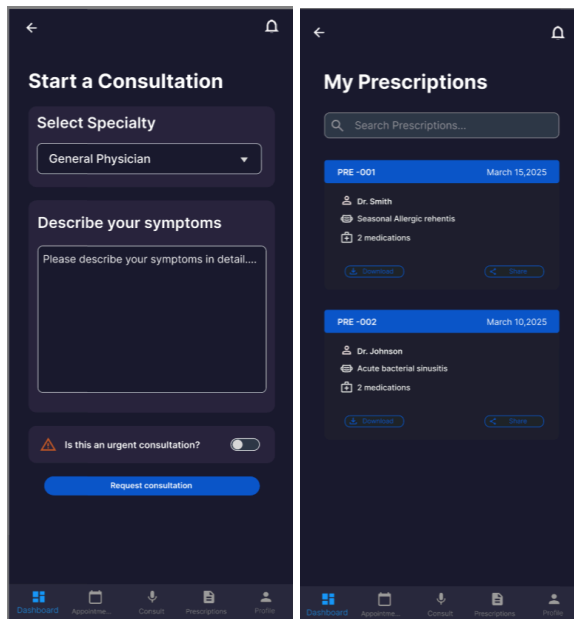
Doctor User flow Screenshots





Patient User flow Screenshots





Project Repository in which i have created the above UI -
<https://github.com/prakharsingh-74/Docpilot>

Challenges & solutions

1. Model Size vs. Accuracy

Problem: Offline AI models must be lightweight (<20MB) while maintaining high accuracy for medical entity extraction.

Solution: Use model compression techniques like quantization and knowledge distillation to reduce size without compromising performance.

2. Physician Resistance

Problem: Doctors may resist adopting new technology due to unfamiliarity about its effectiveness.

Solution: Provide gamified training modules and peer mentorship programs during deployment phases to encourage adoption.

Project Timeline

During the GSoC period:

- 10% of the time will be spent on fixing the bugs.
- 80% of the time will be allocated to adding new features to the App.
- The remaining 10% of the time will be dedicated to testing the app, preparing readme and FAQ for the app.

Time Frame	Activity	Importance
May 8 - May 20	Initial setup and project familiarization	Low
May 21 - June 7	Implementing robust login options, Enhancing UI, Developing more features	High
June 8 - June 22	[END SEMESTER EXAM BREAK]	-
June 23 - July 6	Continuing UI improvements and adding appwrite functions	Medium
July 7 - July 20	Expanding UI features, Implementing contact us and feedback	High
July 21 - August 3	Testing and Integrating AI/ML engine	High
August 4 - August 20	Bug fixing, Making adjustments and completing the unimplemented section of the projects	Medium
August 21 - August 30	Adding readme files and finalizing the documentation	Low

Evaluations

<u>Evaluation</u>	<u>Time Frame</u>	<u>Importance</u>
Phase 1 Evaluation	18 July	High
Phase 2 Evaluation	September 1 - 8	High


Availability

After reviewing the entire GSoC timeline, I can commit to dedicating approximately 40-50 hours per week to the project. However, from June 8th to June 22nd, I will be occupied with my end-sessional examinations, during which I anticipate allocating around 2-3 hours per day to GSoC. Following this period, I will be fully devoted to the project and will not have any other summer internship or job commitments. Additionally, I have no obligations after May and will dedicate all my time to GSoC.

During the project, I plan to provide daily updates on my progress via scrum emails on the mailing list. I am committed to being regular and sincere with my updates as I understand the importance of demonstrating a serious commitment and 100% devotion to the project, which is essential for selection.

Conclusion

Reference links:

- https://pub.dev/packages/lazy_load_scrollview
- <https://ai.google.dev/edge/litert>
-  Search, sort & more with Appwrite queries

Why are you the best person to execute this proposal?

I have a fair experience in building apps using Flutter, Firebase, appwrite, supabase and HTTP APIs, which is the required tech-stack for this project. I also know the importance of writing clean and scalable code. I have worked on several projects. Therefore, I do have the skillset and experience to execute this project.

Post GSoC plans

There are very few chances that some things might go unimplemented because 18 weeks is plenty of time, but still, If it happens, I'll try to complete them post GSoC.

I'll keep contributing to Monumento as much as I can and will keep the development environment running.

Regardless of GSoC, I would love to engage in discussions with the AOSSIE community to get exposure to new technologies and Ideas. I would love to be of any help even after the GSoC period.