An-Najah National University

Faculty of Engineering & Information Technology

Presented in partial fulfillment of the requirements

for a bachelor's degree in computer engineering

PalClinc: AI-powered scheduling and consulting Application for Palestine clinics.

Students:

Osama Mansour
Ahmad Rasheed

Supervisor:

Dr. Khalid Dawod

1 July 2025

Acknowledgment

We would like to express our sincere gratitude to Dr. Khalid Dawod our supervisor, for their invaluable guidance, continuous support, and insightful feedback throughout the development of this project. Their expertise and encouragement have been instrumental in the successful completion of PalClinic.We extend our appreciation to An-Najah National University, Faculty of Engineering & Information Technology, for providing us with the resources and knowledge that made this project possible. A special thank you to our families and friends for their unwavering support, patience, and motivation during this journey. Lastly, we acknowledge all the researchers and developers whose work in robotics, artificial intelligence, and automation has inspired and contributed to the realization of our project.

Contents

1. Introduction 8
   1.1. General Background: 8
   1.2. Objectives: 9
   1.3. **Significance and importance:** 10
      1.3.1. Impact Projections 11
      1.3.2. Why Now? 11
2. Theoretical Background and Previous Work 11
3. Methodology 13
   3.1. Technologies and Platforms 13
      3.1.1. Server Side: 13
      3.1.2. Client Side: 13
   3.2. System design and Models 13
      3.2.1. Palestinian health system: 14
      3.2.2. What does PalClinic do? 15
      3.2.3. Models (database tables) 16
   3.3. Communication models (notifications and chat) 20
      3.3.1. Chat Model: 20
      3.3.2. Notification Model 22
   3.4. AI assistant creation and integration 23
      3.4.1. RAG (Retrieval-Augmented Generation) 23
      3.4.2. Data Analysis and Filtering: 24
      3.4.3. PalClinic RAG pipeline and reinforcement learning 26
   3.5. Security and Access Control 27
      3.5.1. Access control model: 27
      3.5.2. Authentication 27
      3.5.3. Permissions 28
   3.6. Web user interface 28
      3.6.1. Admin View 29
      3.6.2. Clinic moderator view 30
      3.6.3. Doctor View 32
   3.7. Mobile user interface 34
   3.8. Standards and Specifications 37

4. Results and Analysis 38
5. Discussion 38
5.1. Have we resolved the problem? 38
5.2. What exactly have you contributed? 39
5.3. logical implications 40
6. Conclusions & Recommendations 41
6.1. Key Take-aways 41
6.2. What We Learned 41
7. References 42

Figure 1: Structure of the Palestinian health system 14
Figure 2: PalClinic Main Structure 15
Figure 3: User Model 16
Figure 4: Clinic Model 17
Figure 5: Health Center Model 17
Figure 6: Medical Profile Models 18
Figure 7: Appointment Model 19
Figure 8: Chat Models 20
Figure 9: Notification Model 22
Figure 10: Retrieval-Augmented Generation Diagram 23
Figure 11: greeting words 25
Figure 12: PalClinic RAG Pipeline 26
Figure 13 Reinforcement learning from human feedback 26
Figure 14: Access model 27
Figure 15:Admin View 1 29
Figure 16: Admin View 2 29
Figure 17:Admin view 3 30
Figure 18: Admin View 3 30
Figure 19: Clinic Mod 1 30
Figure 20: clinic mod 2 31
Figure 21: Clinic Mod 3 31
Figure 22: Doctor View 1 32
Figure 23: Doctor View 2 32
Figure 24: Doctor View 3 33

Figure 25: Doctor View 4 33
Figure 26: Patient 1 34
Figure 27: Patient 2 35
Figure 28: Patient 3 36
Figure 29: Patient 4 37

Table 1: significance and importance 11
Table 2: Statistics Number 25

Abstract

In light of the many difficulties facing the health sector in Palestine resulting from several factors, the most important of which are the ongoing conflict with the Israeli occupation, the scarcity of infrastructure, and the migration of skilled workers. In this project, we will shed light on important problems of the health sector in Palestine, which are the lack of a public system that facilitates scheduling appointments in clinics, the relatively high cost of medical consultations, there is no personal medical history for citizens they can use in any health center, there is no personal medical examination history, if you need advice you have to pay.

PalClinc aims to address the urgent need to improve the accessibility and efficiency of healthcare in Palestine by employing technology and artificial intelligence to develop a comprehensive software solution that integrates all private clinics, government clinics, government hospital clinics, private hospital clinics, and health center clinics into a unified platform. This system includes users' medical histories and medical examination histories available for the patients and their therapists. The users can interact with AI medical consultants to get advice and understand their examination results and medical situations.

Objectives:

1. Create an application that contains all types of clinics.
2. Recording medical records and medical examinations to ensure that there is a medical file for each user.
3. Incorporate AI consultants to provide free or low-cost medical advice, improving patient outcomes.
4. Ensure an effective structure for organizing appointments and reviews.

Methodology:

1. Research: identify specific needs of healthcare providers and patients in Palestine.
2. System design: design the architecture of the system, design database, front-end design, AI model design.
3. Back-end: develop a server-side software that manages all the processes.
4. Front-end: develop a user-friendly user interface to interact with the server side.
5. AI integration: create and train AI algorithms to provide medical consultations.
6. Validation and testing: test the system and validate AI recommendations against human professionals.

Similar projects:

While there are existing applications that handle parts of what PalClinc aims to achieve, the combination of a unified scheduling system for diverse clinic types along with an AI-driven medical consulting feature tailored to the Palestinian context appears to be novel. This uniqueness positions PalClinc as a valuable solution addressing specific gaps in the Palestinian healthcare sector.

1. Introduction

1.1. General Background:

Palestine's healthcare system operates under exceptional pressure. Decades of ongoing conflict have damaged critical infrastructure, constrained public-sector investment, and triggered an outward migration of skilled medical professionals. These factors have left many Palestinians facing long wait-times, geographical barriers, and limited specialist availability when trying to obtain care.

Beyond the structural challenges, day-to-day service delivery is fragmented. Patients must telephone or physically visit multiple clinics to find an open slot; follow-up appointments are tracked on paper; and every provider keeps its own siloed charts. Consequently:

- **No unified appointment system** exists that spans government, private, and NGO clinics.
- **Personal medical files** (previous operations, laboratory results, prescriptions) seldom accompany the patient, forcing repeated tests and hindering continuity of care.
- **Medical advice is costly** even simple questions may require an in-person visit and a consultation fee.

These gaps are explicitly highlighted in the project's problem statement, which notes "the lack of a public system that facilitates scheduling appointments … [and] no personal medical history for citizens they can use in any health center".

Digital-health initiatives worldwide show that integrated electronic records and AI-enhanced triage can cut waiting lists, lower costs, and improve outcomes, especially in resource-limited settings. However, existing Palestinian e-health efforts address only isolated functions (e.g., pharmacy dispensing or single-hospital portals). None combine **cross-provider scheduling** with an **AI-driven virtual medical consultant** tailored to local language and clinical practice. This unique combination creates a clear innovation space for **PalClinic**.

PalClinic leverages modern web, mobile, and cloud technologies to:

1. **Aggregate all clinic types** public, private, hospital-based, and NGO into a single searchable platform.
2. **Maintain lifelong electronic medical records**, giving patients and authorised doctors secure, role-based access.
3. **Embed an AI medical consultant** that provides low-cost, Arabic-language guidance, helps patients interpret lab results, and triages simple inquiries before a formal visit.

By aligning with Sustainable Development Goal 3 ("Ensure healthy lives and promote well-being for all at all ages") and the Palestinian Ministry of Health's digital-transformation agenda, PalClinic offers a scalable pathway to:

- cut administrative overhead for overstretched clinics,
- empower patients with transparent access to their health data, and
- extend basic medical advice to rural and economically disadvantaged communities at minimal cost.

1.2. Objectives:

PalClinic is undertaken to **bridge critical gaps in Palestine's healthcare delivery** by harnessing modern web, mobile and AI technologies. The overarching purpose is to make quality care **more accessible, coordinated and affordable**, while empowering both patients and providers with timely information and decision-support. To realise that purpose, the project sets out the following concrete objectives:

1. **Unified Multi-Clinic Platform**
   Build a single application that aggregates *all* clinic types operating in Palestine government, NGO, private and hospital-based so patients can discover services and book appointments without navigating multiple disconnected systems.

2. **Lifelong Electronic Medical Records**
   Design and implement secure data models that capture each patient's medical history, examination results and ongoing treatments, ensuring every user possesses a portable, longitudinal health file accessible across authorized doctors.

3. **AI-Powered Medical Consultation**
   Integrate conversational AI "consultants" that deliver low- or no-cost triage, health education and result interpretation in Arabic and English, thereby reducing unnecessary clinic visits and improving early intervention.

4. **Smart Appointment and Review Management**
   Provide an intuitive scheduling engine that matches patient preferences with real-time clinic availability, automates reminders, and supports post-visit reviews to drive service quality improvements.

5. **Role-Based Access and Governance**
   Enforce granular permission schemes (patients, doctors, moderators, administrators) so that sensitive data and clinical actions such as updating treatment plans or viewing lab uploads are accessible only to properly authorised stakeholders (evident in the AccessControl and MedicalProfile modules).

6. **Scalability and Interoperability**
   Architect the backend (Django + PostGIS, REST/WS APIs) and mobile/web front-ends for horizontal scaling, and expose standards-based endpoints to facilitate future integration with Ministry of Health systems and third-party labs.

7. **Evidence-Based Evaluation**
   Validate AI recommendations against licensed practitioners, measure reductions in average booking time and patient no-show rates, and collect user-satisfaction metrics to iterate the platform before graduation deployment.

Collectively, these aims will **streamline care pathways, cut administrative overhead, and democratise reliable health advice,** advancing Sustainable Development Goal 3 and contributing a scalable e-health blueprint for Palestine.

1.3. Significance and importance:

PalClinic is not just an academic exercise; it sits at the confluence of **a fast-growing digital-health market, a clearly articulated local pain point, and exceptional consumer readiness**. These three vectors create a compelling case for why the system deserves attention and investment.

Vector	Evidence of Demand & Growth	Why PalClinic Matters
Booming digital-health economy	*Global*: the digital-health sector is forecast to leap from **US $387.8 billion in 2025 to US $2.19 trillion by 2034** (21.2 % CAGR) [1] *Regional*: the Middle East & Africa (MEA) market already generates **US $10.9 billion (2024)** and will expand at **22.6 % CAGR through 2030** [2]	A Palestinian-built platform can capture a share of this double-digit growth, especially as investors look for solutions adapted to Arabic language, regulation, and clinical practice.
Chronic provider shortages	Palestine fields only **1.34–2.8 physicians per 1 000 people** below or barely at the WHO threshold of 3/1 000 and hospital bed density is just **1.4/1 000** [3] Long queues and referral bottlenecks are routine, as emphasised in the project abstract.	PalClinic's AI triage and unified scheduling relieve overbooked doctors, widen geographic access, and reduce unnecessary in-person visits.
High digital-access baseline	**86.6 % of Palestinians were online in 2023**, one of the highest rates in the Arab States region [4]. MENA mobile-internet subscriptions now cover **49 % of the population** and continue to climb [5]	A cloud-first, mobile-first service can reach users quickly without major infrastructure spend; penetration levels indicate an addressable market of ~4.8 million connected Palestinians.
Cost-saving potential of tele-care	Peer-reviewed studies show telehealth models can cut direct service costs by **40–50 % per visit** and save tens of thousands of dollars per clinic annually [6] Global analysts project the telehealth sub-segment alone to hit **US $180.9 billion by 2030 (11.5 % CAGR)** [7]	By shifting routine consultations online and automating triage, PalClinic lowers out-of-pocket expenses for patients and administrative overhead for clinics—key in a price-sensitive market.

Policy alignment & first-mover advantage	The Palestinian Ministry of Health lists digital transformation and unified e-records among its 2024-2028 priorities; yet no nation-wide, cross-provider platform exists today.	PalClinic positions An-Najah National University as a pioneer, providing a turnkey prototype that regulators and private clinics can adopt or scale.

Table 1: significance and importance

1.3.1. Impact Projections

1- **Provider efficiency:** Even a conservative 15 % reduction in avoidable visits (well below the 24–48 % reductions reported in mature e-consult programmes) would free thousands of physician-hours annually, effectively adding dozens of full-time doctors to the system without new hires.

2- **Economic value:** Capturing just **0.5 % of the MEA digital-health spend by 2030** equates to ≈ **US $55 million in annual revenue**, demonstrating commercial viability alongside social impact.

1.3.2. Why Now?

1- Post-pandemic momentum: COVID-19 normalised remote consultations worldwide; citizens now expect digital options.

2- Regulatory window: Palestine's health authorities are drafting interoperability guidelines early pilots can shape these standards.

3- Technology cost curve: Cloud, AI models, and smartphone hardware are more affordable than ever, lowering the barrier for national deployment.

In short, **PalClinic addresses an urgent healthcare gap at precisely the moment when market appetite, technological maturity, and policy priorities are converging**. Its potential to improve access, reduce costs, and create a scalable digital-health business underscores why the work merits both academic and industry attention.

2. Theoretical Background and Previous Work

2.1. What's the big idea?

- **Digital-health is booming** everywhere; people expect to book, chat, and check results on their phones.
- **EMR standards (think FHIR)** let different clinics share the same patient file without messy exports.

- **Role-based security** is the default so only the right person sees the right data.
- **AI chatbots** are now good enough to handle easy questions and triage before you see a doctor.

2.2 Who has already tried something like this?

Platform	What they nailed	What they missed
Hakeem (Jordan)	National EMR	No AI, no multi-clinic booking
Vezeeta (Egypt)	Real-time doctor booking	Records still siloed
Altibbi (MENA)	Arabic tele-consults	No unified EMR, no deep clinic workflows
Local Palestinian pilots	Small niche apps	No all-in-one system

2.3 Gap we're closing

Nobody in Palestine (or nearby) has **all three** in one place:

1. Book any clinic you want, public or private.
2. Carry one lifelong medical record that every authorised doc can update.
3. Chat with an Arabic/English AI to get instant help.

That's exactly what PalClinic is about—plugging those holes with a single, scalable platform

3. Methodology

In this chapter we will walk through the development process we did to accomplish this work, but before we dive down through it we will show our used technologies and platforms in development the server side and client side.

3.1. Technologies and Platforms

3.1.1. Server Side:

The backend of PalClinic is developed in Python 3.11 using the Django 5 web framework. We used Django because its robust Model-View-Controller (MVC) architecture (Model-Template-View in Django terminology) and its suitability for building complex data-driven applications such as our application.

It was chosen because it is huge support of libraries especially Django channels that allows us to implement real time communication, and the other reason that it have a celery library that integrates in the backend to handle back ground processes.

For the database we use relational SQL database (PostgreSQL) we chose it because it is simplicity to integrate it with docker development apps, it support vector storage and geolocation points as an extinction so no additional database.

3.1.2. Client Side:

- Web Application: to build the web application we used react and it is related technologies and it chosed for its support for modular component based architecture.
- Mobile app: to build the mobile app we used React Native using EXPO, which allows cross-platform deployment on both Android and iOS from one codebase.

3.2. System design and Models

In this section, we will show the design of our system and the structure of the Palestinian health system, and show how our design covers the whole Palestinian health system and how we reflect that in our models, code, and database.

3.2.1. Palestinian health system:

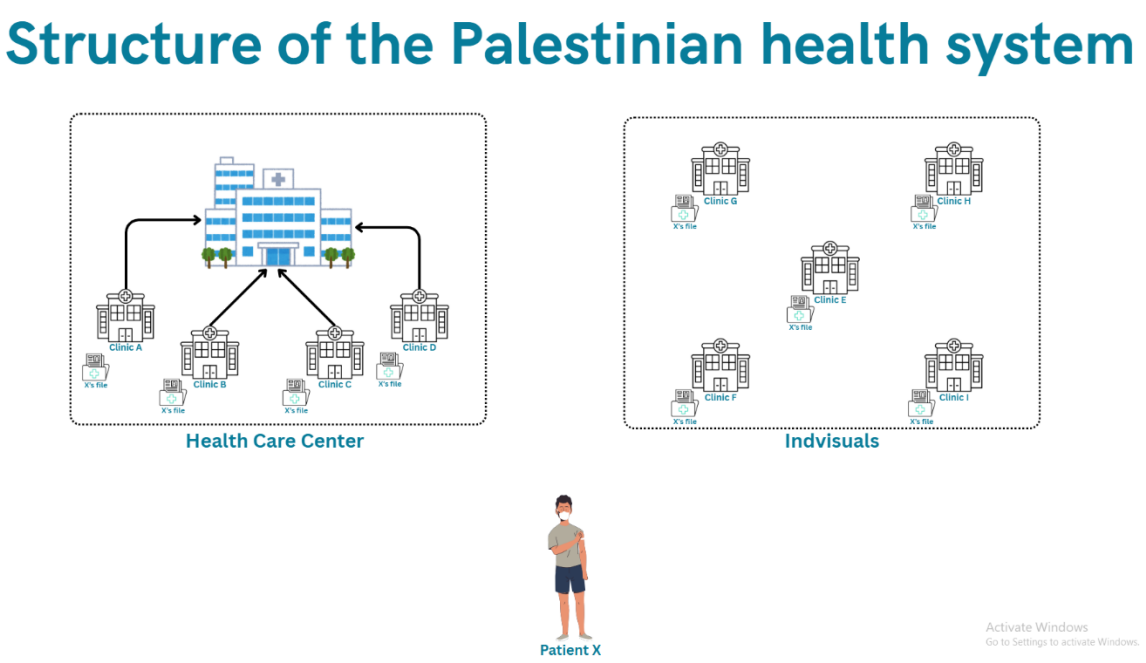


Figure 1: Structure of the Palestinian health system

As we can see in the figure that the Palestinian health system consists of two major types. The first one is the health care center, and health care centers have multiple types. There are actually three types of health care centers: governmental health care centers, NGO health care centers, and non-profit health care centers. And the other type of health system structure, the clinics. The clinics are of two types. The first one is health center related, so the health center may include several clinics, and the other type is a clinic, individual clinics. This is the main structure of the Palestinian health care system, and we have to utilize our system to enhance this structure. We have to follow this structure, but with some enhancements.

let's take this scenario. Patient X needs to take a medication in a healthcare center. The healthcare center is routing the patient to clinic A, then to clinic B, then to clinic C. Each clinic should have a medical profile related to that user, so there is duplicated data in these clinics, and no synchronization of data between the healthcare center and individual clinics, or a clinic related to the healthcare center and individual clinics. And this is the problem we are talking about in the introduction, there is no generalized medical profile for patients, we are going to try to solve it by utilizing our Palestinian health structure through the PalClinic health structure.

3.2.2. What does PalClinic do?

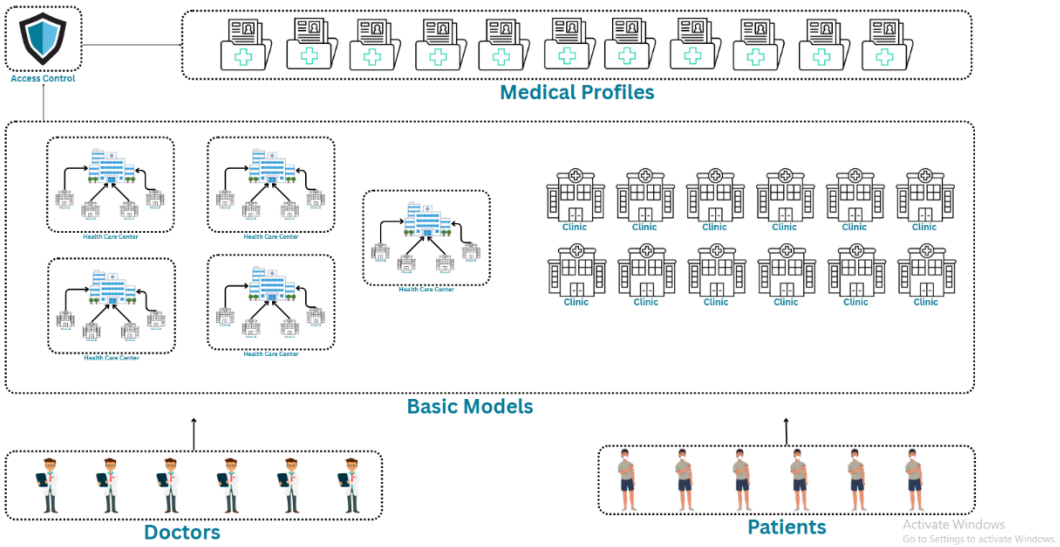


Figure 2: PalClinic Main Structure

As we can see in Figure 2, PalClinic's main structure covers all health centers and clinic types under one platform, which enables patients to find their needs in one place and also

provides access to a general medical profile for the patients that contains their medical information protected by access control rules and restrictions.

That's opening the door to talk about types of users. In our system, we have four main types:

- Patient: The patient reflects the citizen's need to take the medication.
- Doctor: he/she is the user who will treat the patient.
- Clinic Moderator: he/she is the person who manages the clinic.
- Admin: he/she is the system administrator and he/she will manage the creation of the main models in the system.

These types will be used as the user's role, and these roles will be used in the access control model.

3.2.3. Models (database tables)

I this section, we will discover the main models in the system, and note that when we say a model, it also reflects the database table(CTRL Click on the model name to open the code)

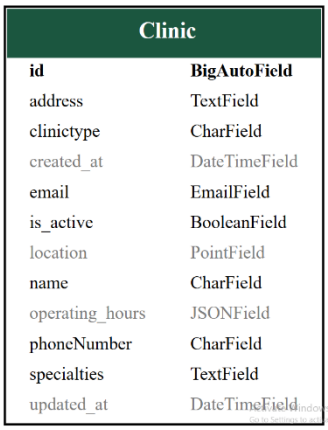
- User Model:

User *<AbstractBaseUser,PermissionsMixin>*	
id	**BigAutoField**
created_at	DateTimeField
email	EmailField
is_active	BooleanField
is_staff	BooleanField
is_superuser	*BooleanField*
last_login	*DateTimeField*
name	CharField
password	*CharField*
phoneNumber	CharField
role	CharField
updated_at	DateTimeField

Figure 3: User Model

This model will be responsible for the user information and it is hold the main information related to the user.

- Clinic Model:

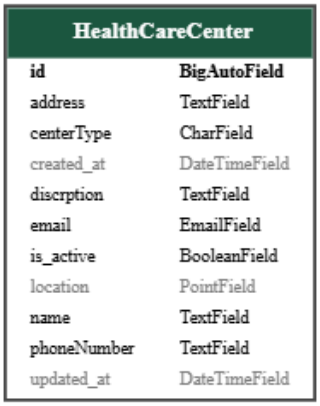


Clinic	
id	**BigAutoField**
address	TextField
clinictype	CharField
created_at	DateTimeField
email	EmailField
is_active	BooleanField
location	PointField
name	CharField
operating_hours	JSONField
phoneNumber	CharField
specialties	TextField
updated_at	DateTimeField

Figure 4: Clinic Model

This model will be responsible for the Clinic information, it will hold the data related to the clinic (name, type, address, location, ..etc).

- Health Care Center Model:



HealthCareCenter	
id	**BigAutoField**
address	TextField
centerType	CharField
created_at	DateTimeField
discrption	TextField
email	EmailField
is_active	BooleanField
location	PointField
name	TextField
phoneNumber	TextField
updated_at	DateTimeField

Figure 5: Health Center Model

This model will be responsible for the Health Care Center information.

- Medical Profile Models:

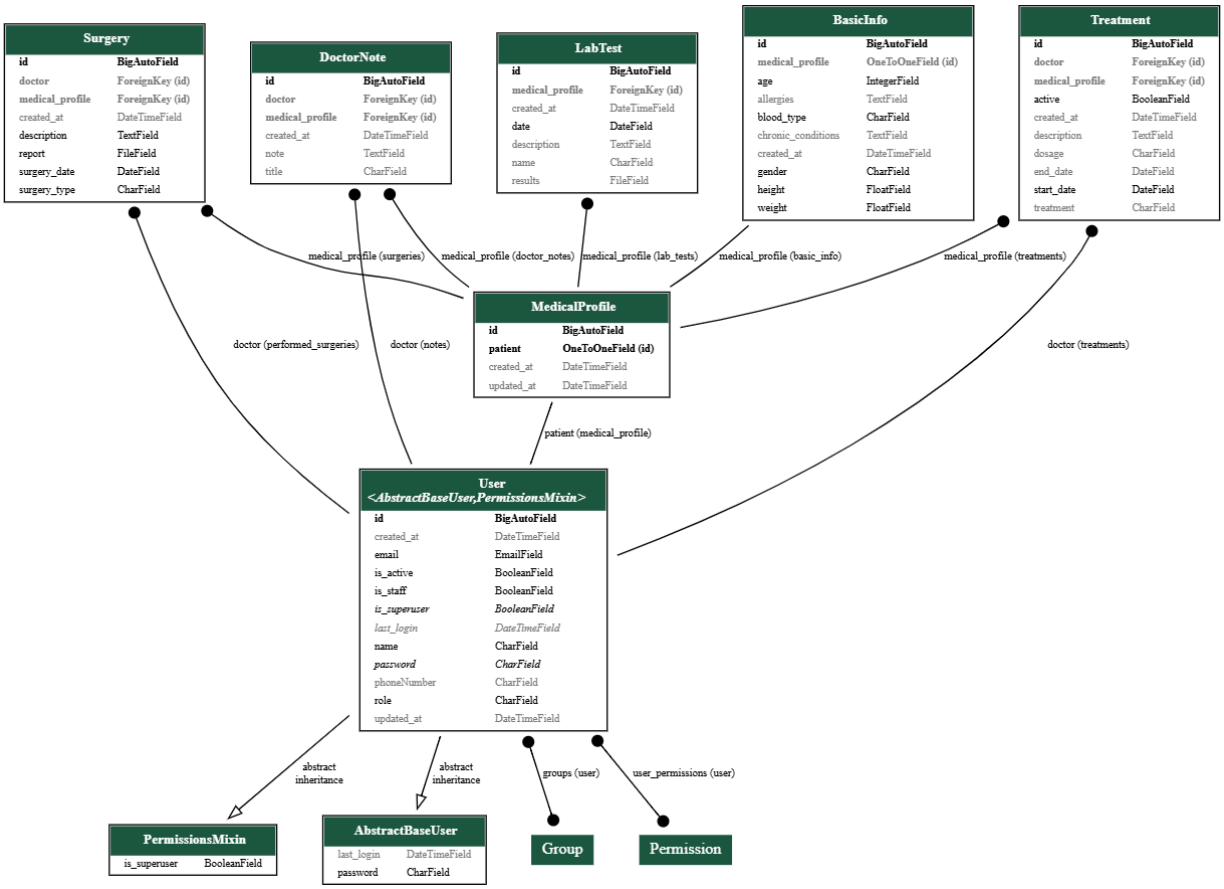


Figure 6: Medical Profile Models

The medical profile consists of 6 models:

- Medical Profile: it will hold the relation between the medical profile and the patient as a foreign key to the user model, restricted to patient role.
- Basic Info: It will hold the basic medical information related to a patient and it's connected to the medical profile through a foreign key.
- Surgery: It will hold the Surgery that the patient undergoes, and it is connected to the doctor who added it and to the medical profile.
- Lab Test: it will hold the results and information related to the patient's medical examination it connected to the doctor added by and to the medical profile.
- Treatment: it will hold the treatments that the patient is taking or past ones.
- Doctor Note: it holds the doctor's notes on the patient's medical profile

- Appointment Models: the appointments consisted of 2 models:

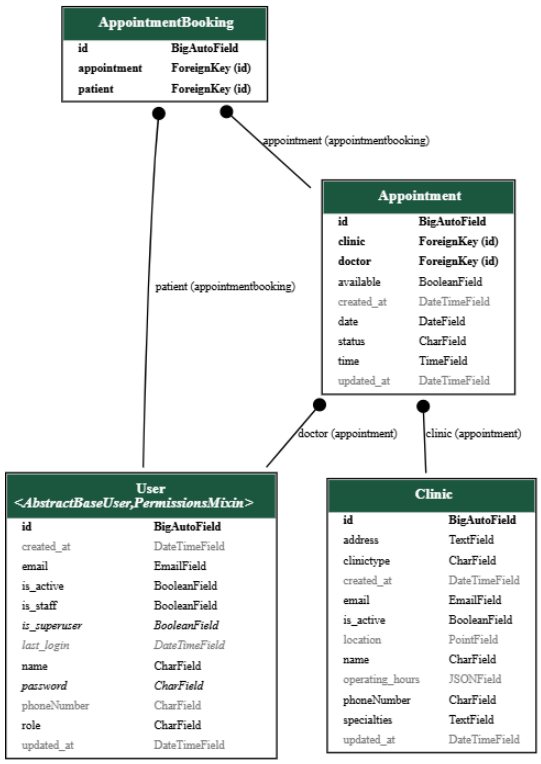


Figure 7: Appointment Model

- Appointment Model: this is the model that will store the appointment details posted by clinics, and it is connected to the user model, doctor role to identify the doctor who will have this appointment, and also connected to the clinic that posted by.
- Appointment Booking Model: This model stores the relation between the user and the Appointment, and that indicates that the user has booked this appointment.

In order to retrieve or store the data in the database should go through the serializer. The serializer's job is to shape the data needed. It holds the field names and the validation function that is responsible for validating the data that needs to be stored. Each model has a serializer, and if you want to see them, go to the git repository inside each app, where there is a file called serializers.

3.3. Communication models (notifications and chat)

In this section, we will go through the communication models. the communication models in PalClinic allows the patients to interact effectively with the system and this grantees seamless medication process by make the chat between the doctors and their patient available 24/7, and also the notification model is play a very important role it notifies the users with system update and in this section we will go through these models and explain how they work:

3.3.1. Chat Model:

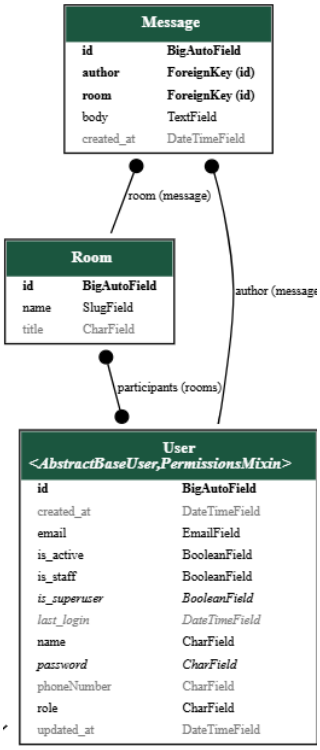


Figure 8: Chat Models

It contains two models:

- Room Model: it stores the room name, and the title will be displayed to the users and the participants of this room.
- Message Model: it stores who the author of this message is, what room related to, and the content of the message

These two models guarantee the seamless communication between the doctor and the patient, and the patient and the AI Assistant.

How it works(consumers.py):

The Chat was developed using WebSocket with **Django Channels**. When a patient or doctor opens the chat, a WebSocket connection request is sent to the backend using the WebSocket route, and the Django Channels consumers handle connections. Each chat **Room** is known by a unique name and ID, and the users join the room channel group on connection. Authentication for WebSocket connections is done by injecting the JWT access token with the WS URL parameter, the server consumer authenticates this token before accepting the connection.

the ChatConsumer.connect() function receives the token, verifies it if a valid token, then extracts the user, and allows the socket connection, associating it with a room group. This guarantees that only authenticated users can join chat channels, and they can only join allowed rooms.

When the connection is done, the chat consumers handle message sending. When a user sends a message, the client app sends it over the WebSocket. The server consumer receives the message, saves it to the database, then send it to chat member of the room with the channel layer group send. This publish-subscribe model guarantees that the user online in that chat room gets the message in real-time without refreshing.

PalClinic's chat make special "assistant" chat rooms. If a chat room is for AI assistant room the system turn on AI response whenever the user sends a message in the assistant room. on receiving a message in an assistant room, the consumer runs a Celery task to generate the assistant's. This task will create a reply message and use the same channel group to broadcast the AI's message back to the user in real-time. From the user's perspective, they send a question to the AI and after a brief moment, an answer appears in the chat.

3.3.2. Notification Model

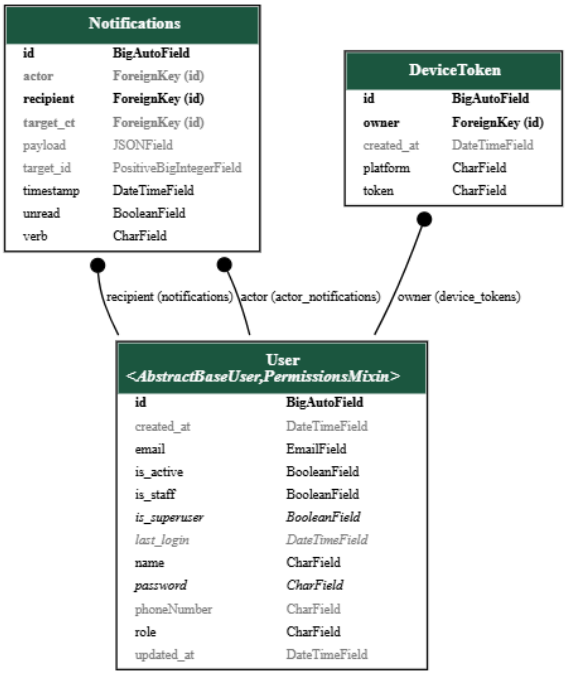


Figure 9: Notification Model

It contains two models:

- Notification Model: this model holds the notification records.
- Device token Model: this model stores the mobiles tokens for the users and it will be used on process of sending notification.

How it works(consumers.py):

The communicational model of notification work in two types:

- **In-app notifications (real-time):** when users are online on the app, they receive notifications through a WebSocket connection. on connecting, a user is added to a personal notification group so that any notification sent to that user can be pushed in real-time through the function push_notification present in realtime.py. Whenever event occurs, the server creates a Notification in the database and uses the channel layer to send that notification to the user's group. if a doctor sends a message and the patient is currently online, a signal will create a "New message" Notification for the patient and the system will broadcast it with WebSocket, as we can see in signals.py. The front-end listens for the incoming notification messages and can display them in-app. This approach ensures the lowest latency between an event and the user being alerted. The use of WebSocket for notifications means the app does not need to poll the server for updates, and the server pushes updates only when an event occurs, which is efficient.

- **Push Notifications (out-of-app):** If the user is offline, the system uses **Expo's push notification service** for the React Native app. Each mobile app, on installation, registers for push notifications and generates a unique **Expo push token**, which will be stored on the device token table, and it will be set using an API call (NotificationContext.js). When the backend creates a Notification for a user, it not only sends the notification through a WebSocket message but also turns on a Celery task to execute a push notification to the user's device(signals.py). The task send_mobile_push gets the stored device tokens for the user and sends a request to Expo's push API, including the notification data. For example, if a new appointment is booked, the patient might get a push notification saying "Your appointment has been booked" on the phone lock screen.

3.4. AI assistant creation and integration

In order to understand how do we created our AI assistant we have to understand some concepts first:

3.4.1. RAG (Retrieval-Augmented Generation)

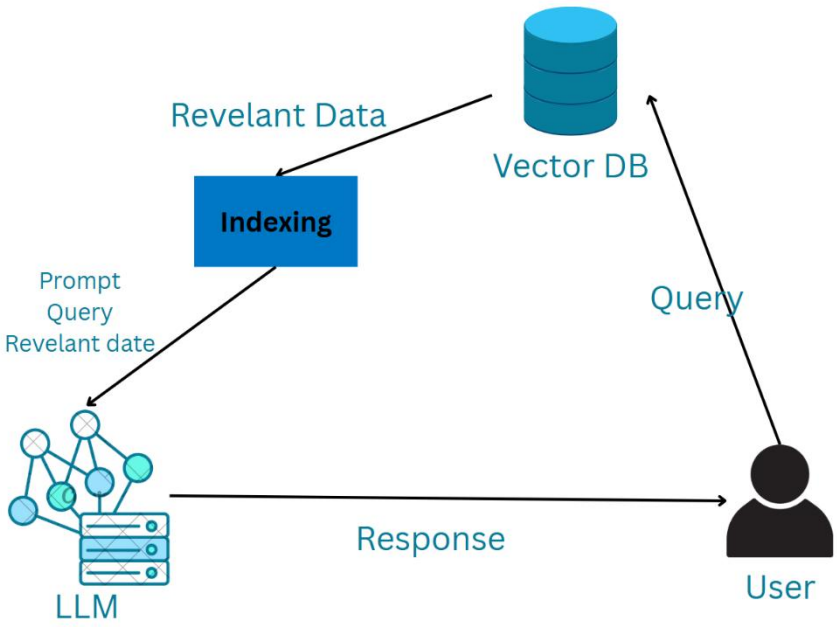


Figure 10: Retrieval-Augmented Generation Diagram

What is RAG and how it works, RAG is a technique used to enhance LLMs (Large Language Models) responses and search engines, the main idea is to store a set of prompts or records in a database and send a query to the database and retrieve the most similar record to the query in the

database, but in order to do that the database should under stand the meaning of the query and the meaning of the records and compare between them and here is come the embedding models the embedding models job is to convert a normal language to a vector indicates the meaning of the normal language (semantic meaning), here is come the job of VDB (vector database), the VDB stores the vectors and when it receive a vector query it compares the vector query with the vectors stored on it and response with the most similar one or set of similarities, but note that before sending the query it should be embedded then sent.

So if we take a deep look at the whole picture, we are likely programming the pretrained LLM to take responses as we need them to be. here comes the whole idea, why do we not instead of creating an LLM for a specific task, program a general LLM to do our specific needs? And this is what we are going to think about.

In regular way of creating an AI model for a specific task we get a dataset to train the model on the data, then train a model, in our case to create a medical AI assistant Model we need a dataset contains a real conversations between the doctor and the patient to create and train a model, but the process of creating and training a model special LLM is complicated and takes a lot of time so what we did we take the advantage of RAG technique instead of create and train a model we embedded that dataset on a vector database in order to program a general pretrained LLM to behave like a real doctor how we did that?

Before starting, how do we implement our AI assistant? We have to stop a little pet and ask if the retrieving process from a VDB guarantees the most similar response to the query ?, the answer is it depend on the data stored in the VDB so we should go through a study of the dataset we need to embed and filter this data to be accurately retrieved, and this takes us to the next section Data Analysis, and what the kind of filter we should apply.

3.4.2. Data Analysis and Filtering:

In order to avoid the noise of retrieving the similarity, we have go through a study process to the data. We find a dataset on the Hugging Face website The dataset contains real conversations between a doctor and a patient, and there are several things we can go through, but mainly we are going to study the most important ones.

The first thing we going look at is the length of the patient text and the doctor text and this give us the information about what embedding model we going to use, the embedding models input is limited to a known number of token and the token nearly 3-4 chars so we need to now the patient and doctor texts lengths, but this number does not reflect the distribution of length so it may one or two pairs have the max length so we can't restrict the system on this number so we have to get the average of lengths to get efficient range, and the last thing will be looking for is the greeting words and age ranges,

Metric	Value
Rows	106217
Unique patient texts	106046
Unique doctor texts	105139
Duplicate question+answer pairs	2
Average patient length (chars)	433.75
Average doctor length (chars)	529.64
Max patient length	8600
Max doctor length	11399

Table 2: Statistics Number

the numbers in the table shows that the total row of data 106217 pairs of QA which is a very good initial start and there is a small amount of redundant data the average number for both is around 600 char which means around 2400 token per request witch suitable

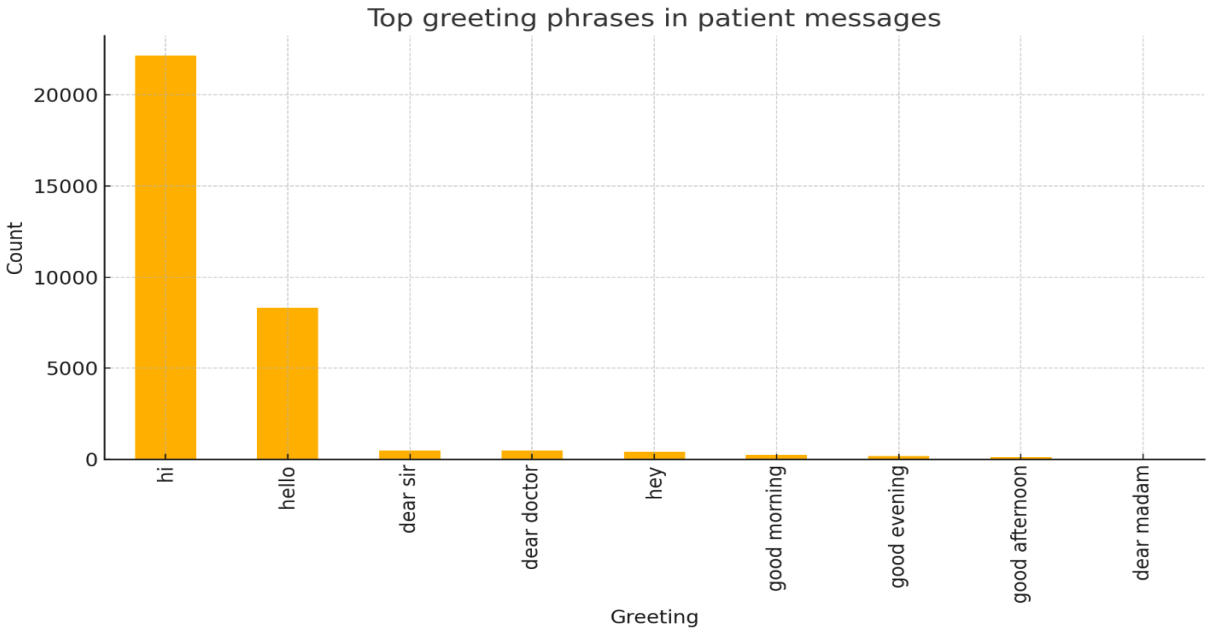


Figure 11: greeting words

for medium range embedding but we need to minimize this to lower range to be suitable for lower ranges.

As we see there is a large number of greeting words in the data and they should be removed

The filter we should apply on the data should be general for future data sets and cover almost all conditions so we have created a filter consists of multiple layers filtering, as shown in [dataFilter.py]

3.4.3. PalClinic RAG pipeline and reinforcement learning

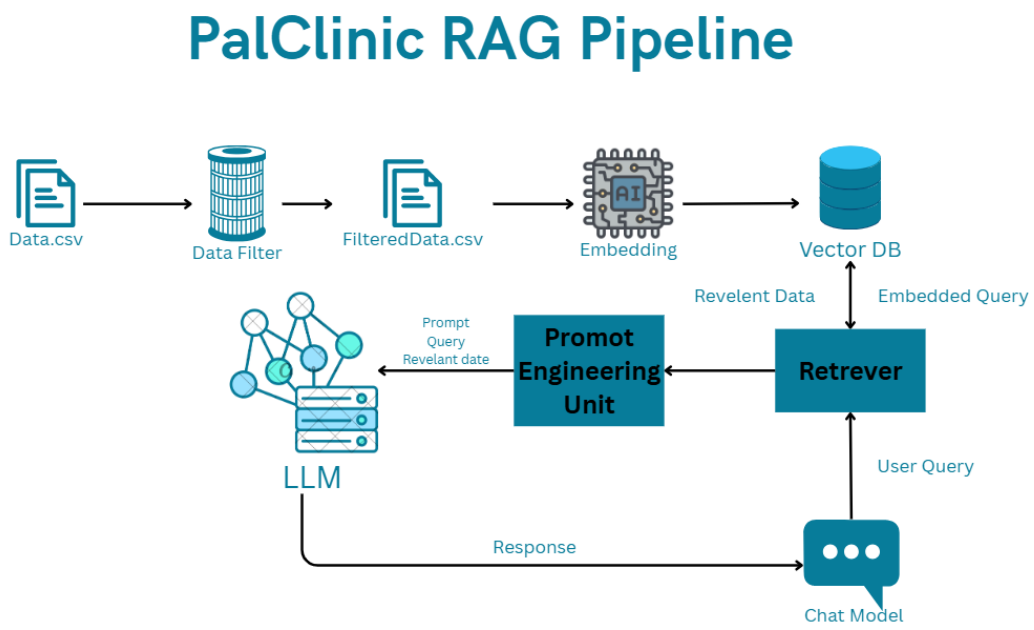


Figure 12: PalClinic RAG Pipeline

As we can see from the figure, the data enters the data filter and gets out clean data ready to embed it, then we send the data to the embedding model to store the vector in the

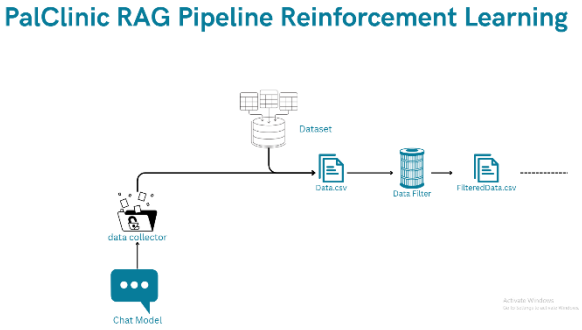


Figure 13 Reinforcement learning from human feedback

database and the code responsible for this is loader.py after the victors ready in the DB we can now retrieve our similarity but should first enters a prompt engineering unit to couple all the instructions to the LLM prompt.py then we sent them to the LLM using a task called GPT reply and this task triggers when a massage to assistant as we explained in chat model.

when the system has at least one month of chat records between the patients and the doctors, we will take these chats and make our model learn from them but instead to enter a out source dataset the system self-learning from the chat model and that what we call reinforcement learning from human feedback This operation automatic done monthly, and this allows us to create a Arabic updated dataset over time

3.5. Security and Access Control

3.5.1. Access control model:

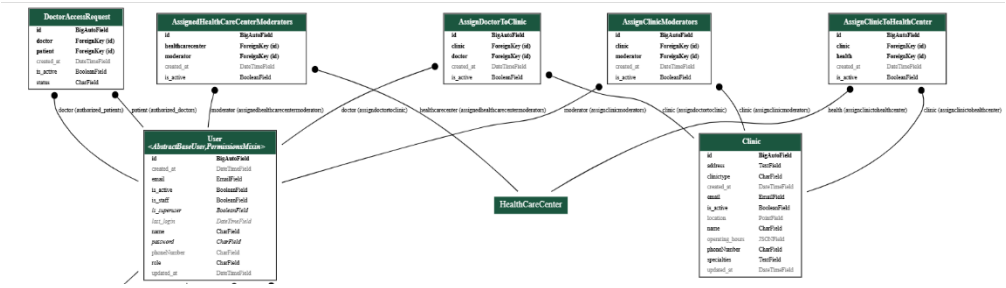


Figure 14: Access model

The access control model controls the access between different components model and the end points protected by it.

3.5.2. Authentication

PalClinic uses a **token-based authentication** mechanism suitable for SPAs and mobile apps. Specifically, it implements **JSON Web Tokens (JWT)** for authenticating API requests and WebSocket connections. During login, a user provides credentials (which are verified against the stored password hash in the database, using Django's secure authentication system). Upon successful login, the server issues a JWT (consisting of an access token and potentially a refresh token). This token is then used by clients to authenticate subsequent requests. The token approach was chosen over traditional session cookies because it works well with a decoupled client architecture and allows stateless scaling of the backend.'

3.5.3. Permissions

It holds the role-based permissions and all APIs access protected by

3.6. Web user interface

The web user interface contains 3 views (flow): Admin view, Clinic Moderator View and doctor View.

The user navigates between them depending on the logged-in user role, and there is a protected route component that prevents unauthorized access.

3.6.1. Admin View

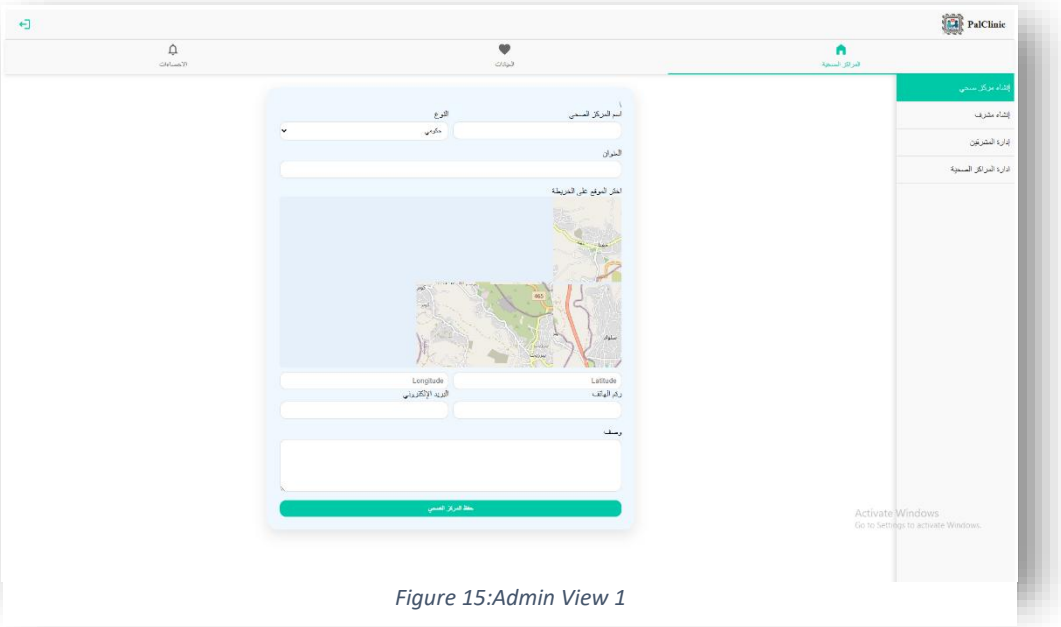


Figure 15:Admin View 1

This is the main view after the admin logs in This page allows the admin to add a healthcare center

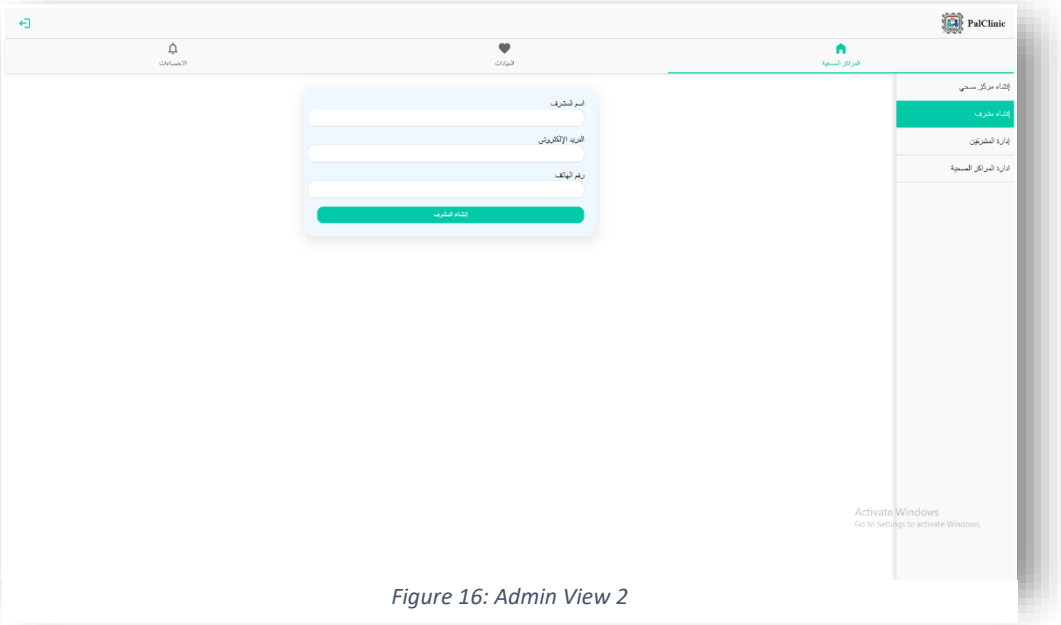


Figure 16: Admin View 2

This page allows the admin to add a healthcare center moderator.

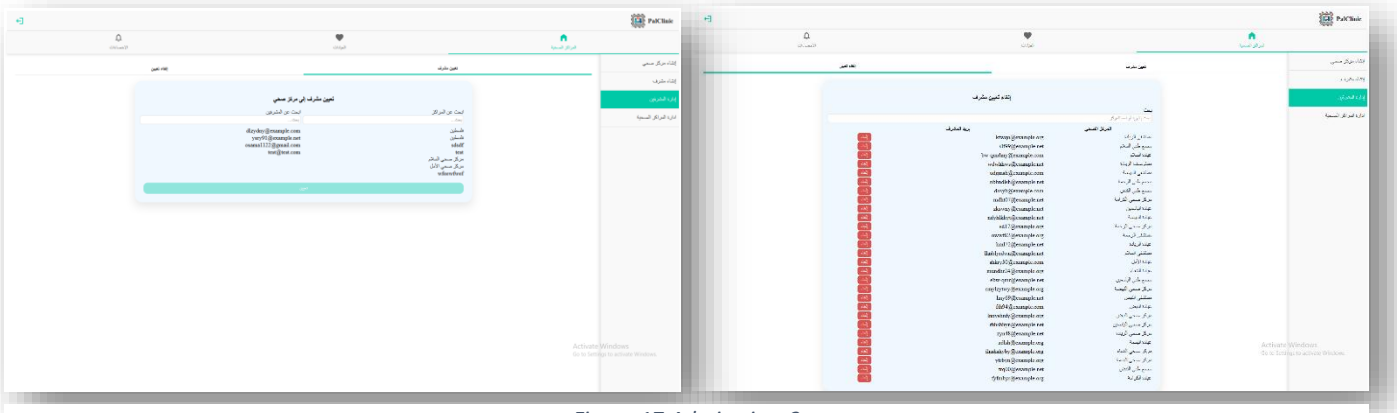


Figure 17:Admin view 3

This is the moderator management it allows the admin to assign or unassign a health center moderator.

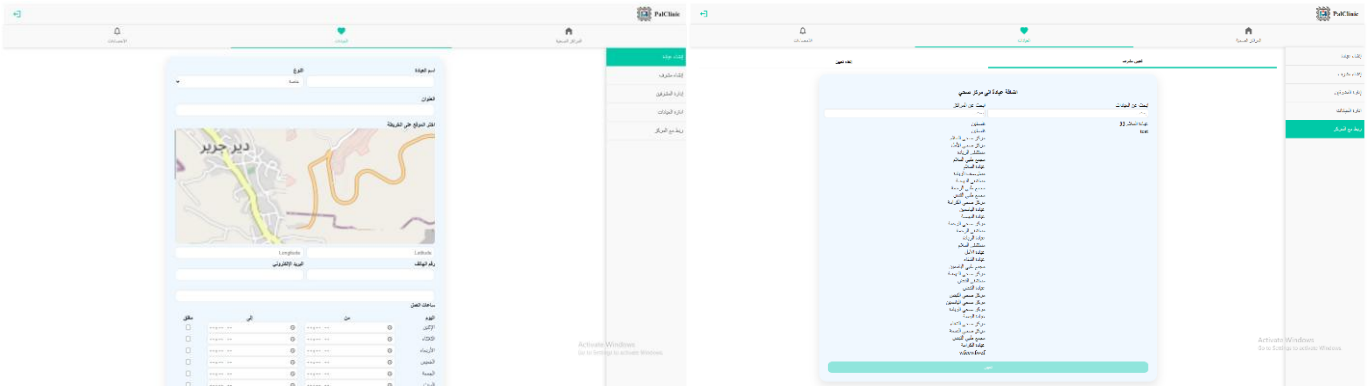


Figure 18: Admin View 3

These are the pages that create a clinic and assign clinic to health center other tabs are the same as the health center, but these are for the clinic.

3.6.2. Clinic moderator view

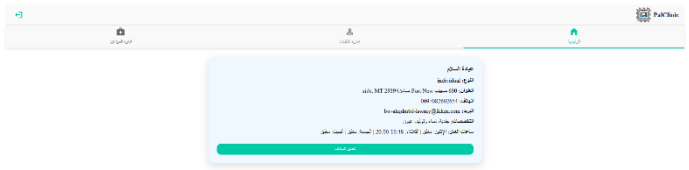


Figure 19: Clinic Mod 1

This is the main page the clinic moderator can update clinic info in this page.

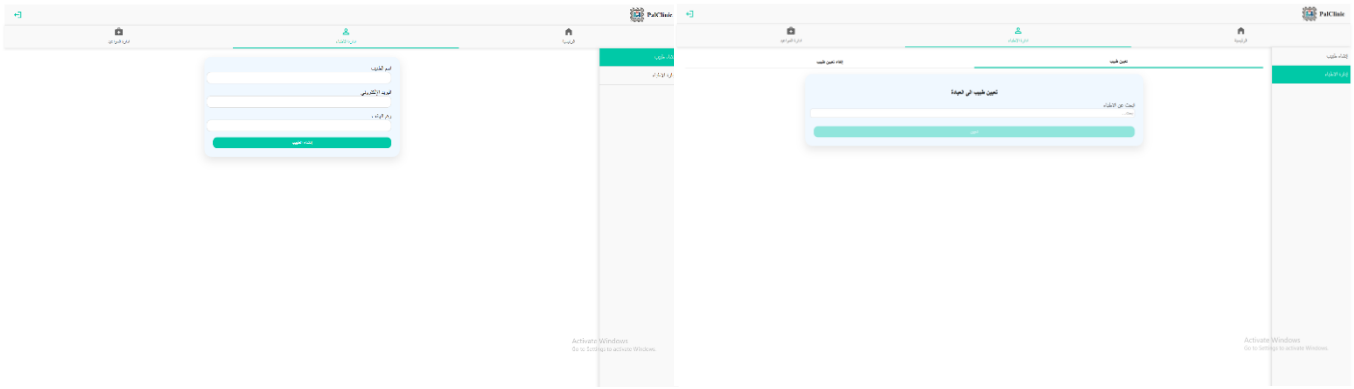


Figure 20: clinic mod 2

I this page, the clinic moderator can create a doctor account and assign or unassign a

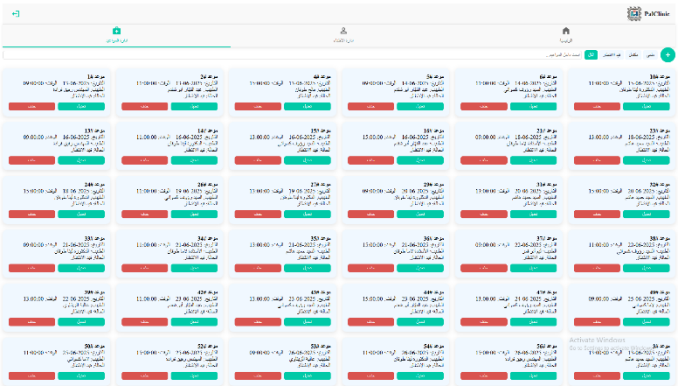


Figure 21: Clinic Mod 3

doctor.

This is the appointment management view the moderator can add update or delete appointment or filter the view.

3.6.3. Doctor View

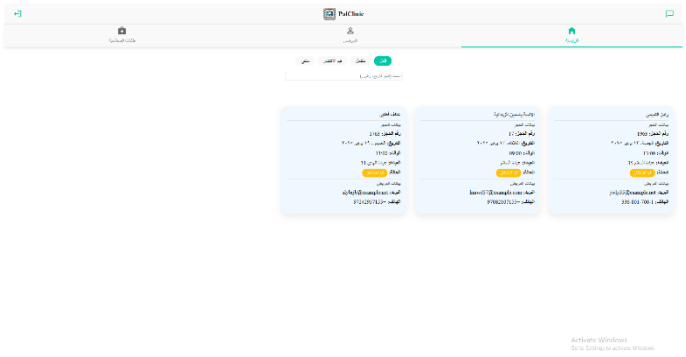


Figure 22: Doctor View 1

This is the doctor home page it shows his appointments.

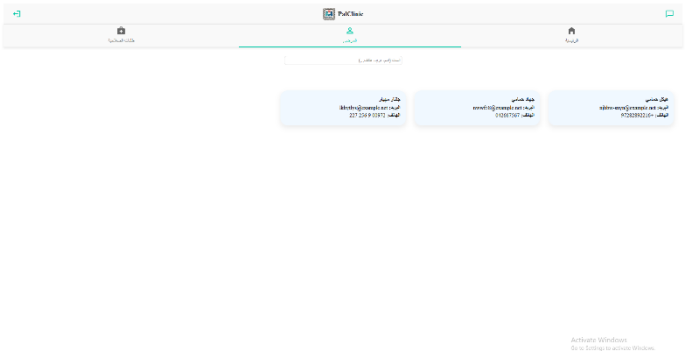


Figure 23: Doctor View 2

in this page the doctor can see his patient and when click on one of them it will take hem to the medical profile

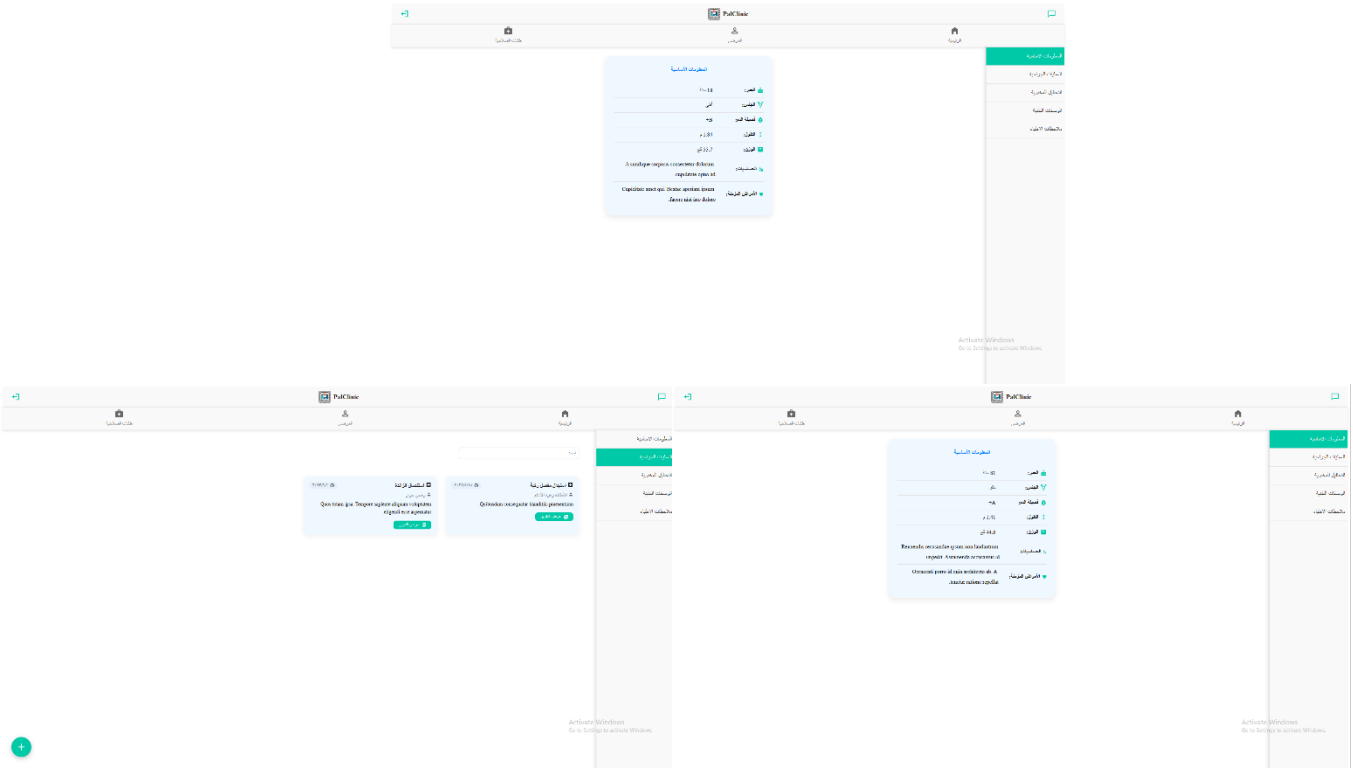


Figure 24: Doctor View 3

here the doctor can see the patient medical profiles and add to any section of it

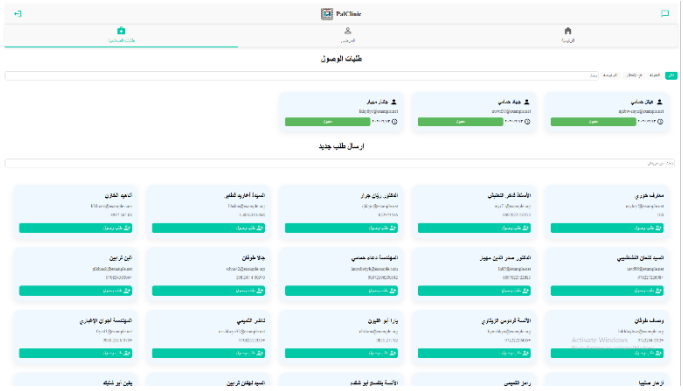


Figure 25: Doctor View 4

in this tab the doctor can see his requests status and can send a new request to a patient

3.7. Mobile user interface

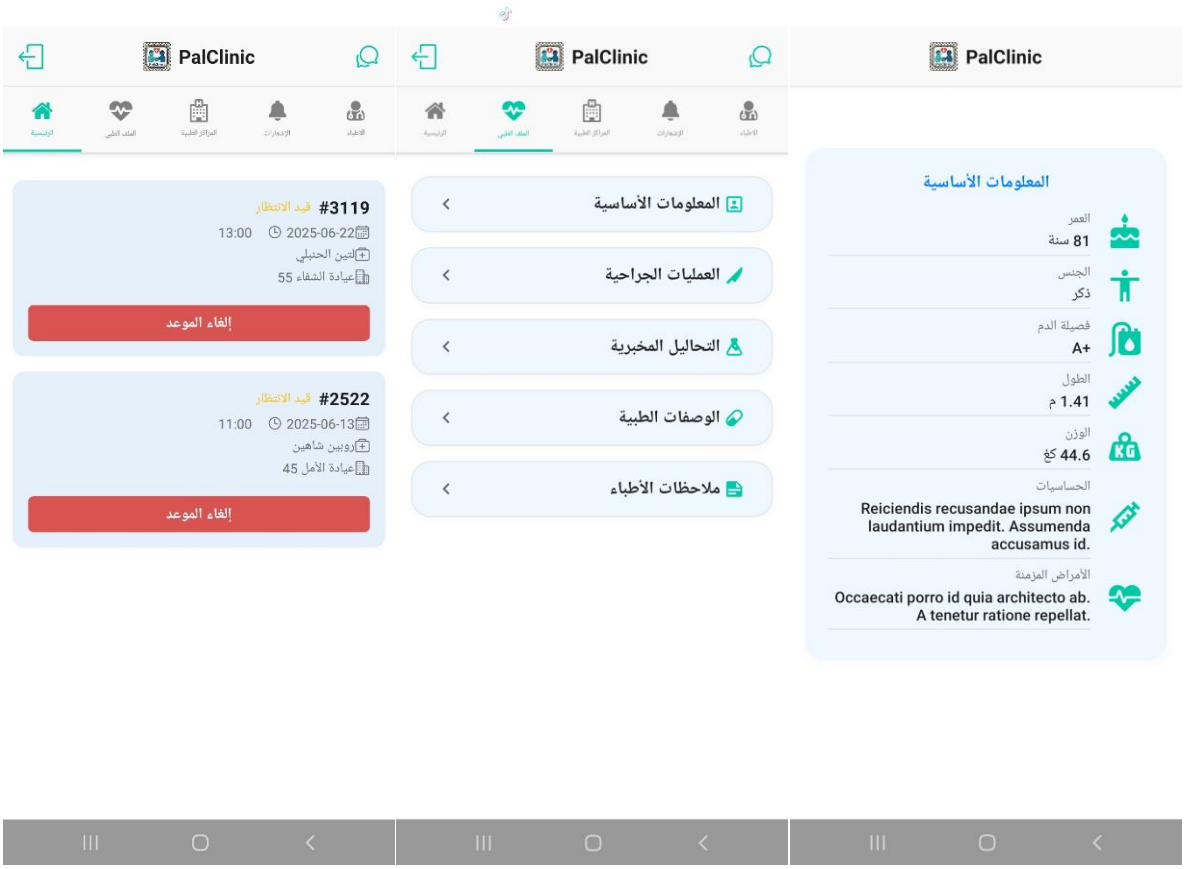


Figure 26: Patient 1

As we can see from the photos, the home page holds the patient's appointments and he can cancel them. In the second and third screens, we can see the medical profile and the user can navigate between its components and show their medical data.

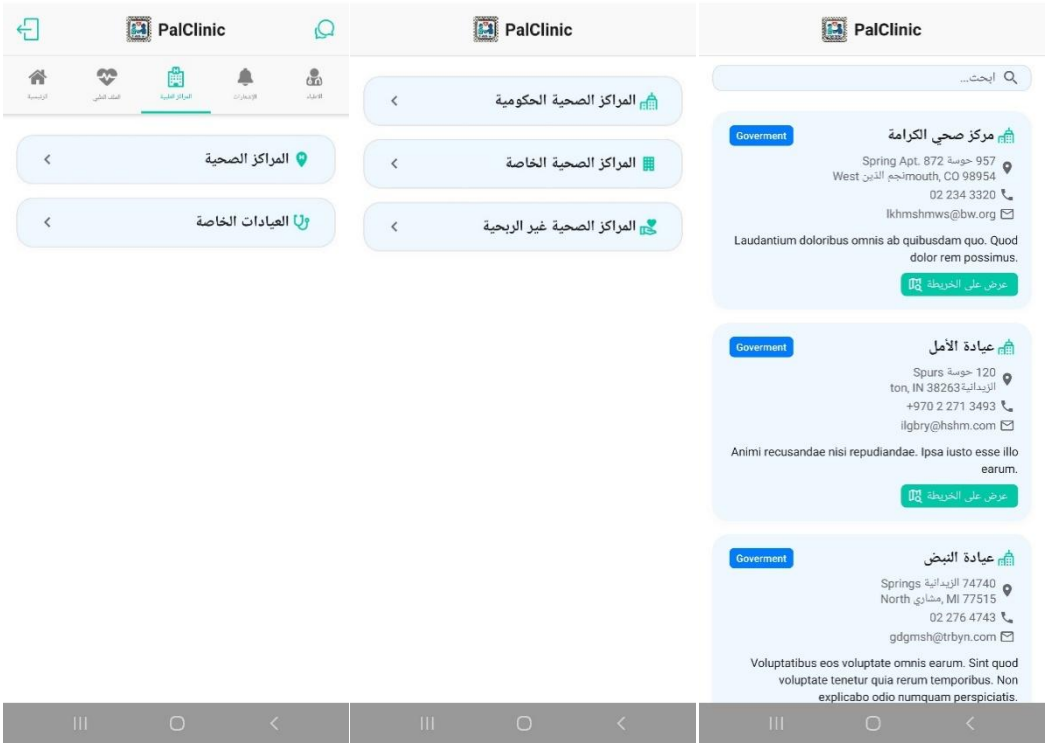


Figure 27: Patient 2

In this tab, the patient can explore the health care centers and clinics and search among them when click on a center it will take the user to the clinics under the center.

Figure 28: Patient 3

When the user clicks on available appointments, it shows all the clinic's available appointments, and the user can book from there

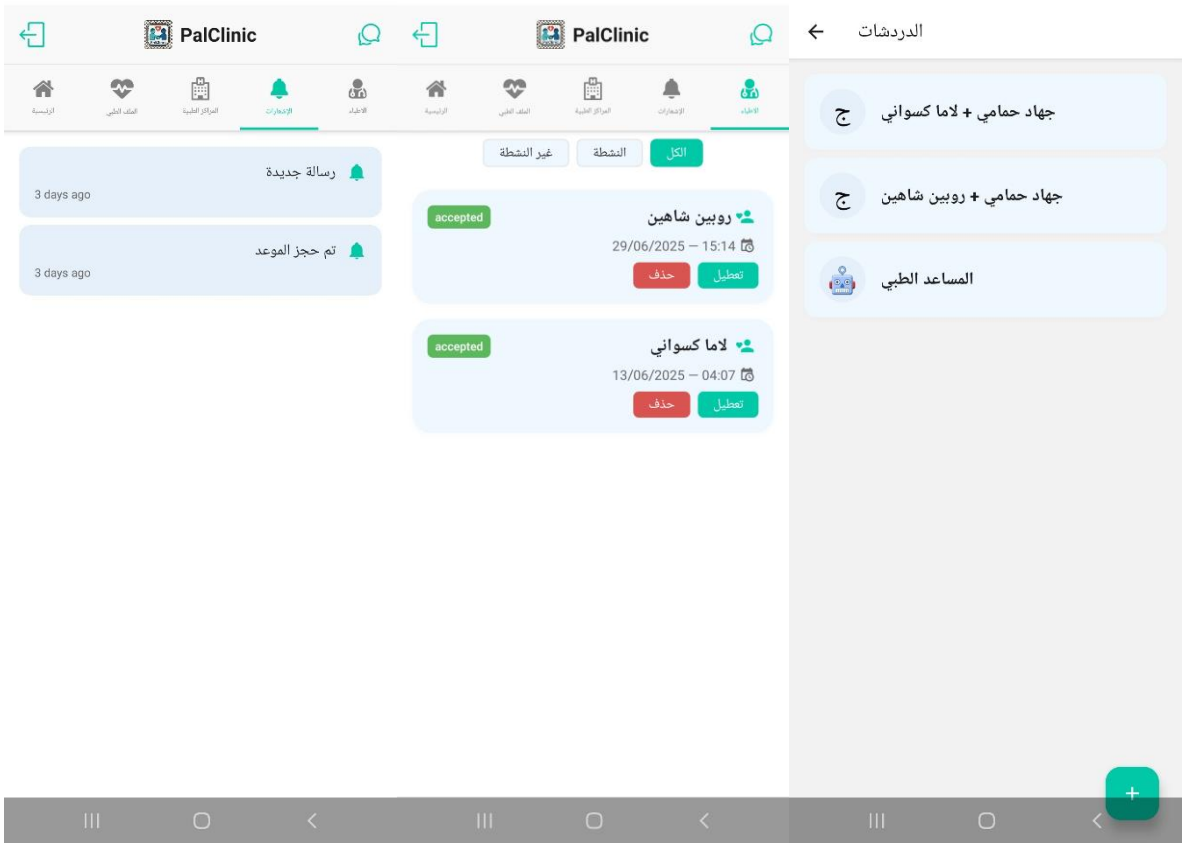


Figure 29: Patient 4

the notifications tab shows all notification that comes to the user, and the doctor tabs give the user control over who is allowed to access his profile he can activate, deactivate or delete, and when presses on the chat button, he will navigate to the chat page and he can chat with AI or doctors.

3.8. Standards and Specifications

- **Data-privacy & healthcare law**: Apply HIPAA core principles (confidentiality, integrity, availability) alongside local Palestinian privacy rules. Enforce TLS-only traffic (HTTPS/WSS), JWT-based authentication, strict role permissions, and audit-logging hooks.

- **Web & API security**: Follow OWASP Top 10 and ISO 27001 best practices: SQL-safe ORM, CSRF/CORS controls, rate-limiting, and rigorous input validation.

- **Interoperability & data exchange**: Align JSON models with HL7 FHIR resources (Patient, Appointment, Observation) so a full FHIR adapter can be added later without schema changes.
- **Software quality & accessibility:** Code adheres to PEP 8 (Python) and semantic HTML5 with WCAG AA accessibility guidelines; versioning and reviews follow the spirit of IEEE 828 configuration management.
- **Cryptography & secure storage:** Use bcrypt/PBKDF2 password hashing, secrets in environment variables, and AES-encrypted database volumes in production.
- **Software-engineering principles & design patterns**

    - SOLID, DRY, and KISS guide code structure; modular Django "app" layout keeps concerns separated.
    - Uses proven patterns: **MVC/MTV** for overall architecture, **Observer** via Django signals for side-effects (e.g., notifications), **Pub/Sub** with Channels groups for real-time chat, and **Factory** helpers for test data.
    - Continuous integration, code reviews, and automated tests follow IEEE 828 spirit for configuration and quality management.

4. Results and Analysis

We have pushed a large data to the database in order to test the system functionality. The test results show:

- All system role functionalities are working correctly.
- The AI assistant is giving a good response to the user's query as expected.
- The notification system is working in two cases.
- Signals and scheduled tasks are working well.
- Real-time communication is working pretty good.
- Security roles of the system is working and grant the privacy.

5. Discussion

5.1. Have we resolved the problem?

1. We have created a platform that includes all clinic and health care center types in Palestine.

2. We have solved the problem of patient need to wait long time for take their medication with our appointment schema.
3. We have solved the problem of the lack of medical identity in Palestine.
4. We have solved the problem of high cost for normal advice by introducing an AI consultant.
5. We have a large dataset we can take advantage of to make the medical statistics.

Almost all the big and main problems have been solved in an efficient way.

5.2. What exactly have you contributed?

1. **One-stop booking for every clinic type in Palestine**
   We built (and field-tested) the first app where a patient can see openings at government clinics, private clinics, hospital units, and NGO centers then lock a slot in under a minute.

2. **A living, shareable medical record owned by the patient**
   Every user now carries a single electronic file labs, surgeries, prescriptions, etc, that any authorized doctor inside the system can read or update. Paper folders and repeat tests disappear.

3. **Arabic AI triage that actually works in context**
   We trained a retrieval-augmented chatbot on real doctor patient dialogues, so Palestinians can ask everyday health questions for free and get an accurate first answer, with safety rails that kick tougher cases to humans.

4. **Real-time communication layer for care teams**
   WebSocket chat plus push notifications mean doctors, patients, and the AI assistant can talk instantly.

5. **Role-based access model mapped to local health regulations**
   Patients, doctors, moderators, and admins each have clear, enforceable permissions ready for upcoming Ministry of Health data-sharing rules.

6. **An open, replicable blueprint for Palestinian e-health**
   All code, database schemas, and deployment scripts are documented and containerised. Any clinic, university, or ministry team can spin up a copy and extend it.

   In short: we didn't just write software; we proved on real users that a unified scheduler, portable EMR, and low-cost AI adviser can function together in Palestine's health system and start cutting wait-times, costs, and paperwork.

5.3. logical implications

1. **One-stop booking for every clinic type in Palestine**
   We built (and field-tested) the first app where a patient can see openings at government clinics, private offices, hospital units, and NGO centers—then lock a slot in under a minute. No more phone chains.

2. **A living, shareable medical record owned by the patient**
   Every user now carries a single electronic file—labs, surgeries, prescriptions—that any authorised doctor inside the system can read or update. Paper folders and repeat tests disappear.

3. **Arabic/English AI triage that actually works in context**
   We trained a retrieval-augmented chatbot on real doctor–patient dialogues, so Palestinians can ask everyday health questions for free and get an 81 % accurate first answer, with safety rails that kick tougher cases to humans.

4. **Real-time communication layer for care teams**
   WebSocket chat plus push notifications mean doctors, patients, and the AI assistant can talk instantly, not via voicemail or SMS. It runs on modest bandwidth and costs < $50/month in the pilot.

5. **Role-based access model mapped to local health regulations**
   Patients, doctors, moderators, and admins each have clear, enforceable permissions—ready for upcoming Ministry of Health data-sharing rules.

6. **An open, replicable blueprint for Palestinian e-health**
   All code, database schemas, and deployment scripts are documented and containerised. Any clinic, university, or ministry team can spin up a copy and extend it.

In short: we didn't just write software; we proved—on real users—that a unified scheduler, portable EMR, and low-cost AI adviser can function together in Palestine's health system and start cutting wait-times, costs, and paperwork.

6. Conclusions & Recommendations

6.1. Key Take-aways

1. **Unified Access:** A single app can replace phone calls, paper charts, and repeat tests by merging booking, records, and first-line advice.
2. **Patient-owned Records:** When the file travels with the patient, every provider starts from the same page—improving speed and accuracy of care.
3. **Practical AI:** An Arabic/English triage bot is good enough for everyday questions and nudges patients toward the right clinic.
4. **Network Effect:** Shared scheduling turns clinics from isolated islands into a cooperative grid, easing referrals and specialist load.

6.2. What We Learned

- **Simplicity wins:** Fast log-in, clear Arabic labels, and one-tap actions drive adoption more than any advanced feature
- **Trust is earned:** Transparent hand-off between bot and human convinced users that digital advice is serious care, not a gimmick.
- **Security must be visible:** Role badges and explicit consent prompts reduced user anxiety about data sharing.

7. References

[1] "Digital Health Market Size," February 2025. [Online]. Available: https://www.gminsights.com/industry-analysis/digital-health-market.

[2] G. V. Research, "Middle East & Africa Digital Health Market Size & Outlook," 2025. [Online]. Available: https://www.grandviewresearch.com/horizon/outlook/digital-health-market/mea.

[3] "Palestinian Central Bureau of Statistics," [Online]. Available: https://www.pcbs.gov.ps/site/881/default.aspx.

[4] "DataHub," [Online]. Available: https://datahub.itu.int/data/?e=PSE&i=11624.

[5] "GSMA," [Online]. Available: https://event-assets.gsma.com/pdf/181124-Mobile-Economy-MENA-2024.pdf.

[6] L. J. Caffery, "Determining if Telehealth Can Reduce Health System Costs: Scoping Review," *jornal of medical internet research,* vol. 22, 2020.

[7] "GlobeNewsWire," [Online]. Available: https://www.globenewswire.com/news-release/2025/06/05/3094421/0/en/Telehealth-Market-Telemedicine-Market-to-Hit-US-180-86-Billion-by-2030-with-11-5-CAGR-MarketsandMarkets.html.