MedBook

Searching for a doctor made easy.

User Requirements and System Specifications

Version 1.0

Gerogiannis Konstantinos 9638 konsgero@ece.auth.gr

Gerontopoulos Anastasios 9682 ganastas@ece.auth.gr

Mparmpounakis Konstantinos 9759 kmparmpou@ece.auth.gr

Antoniadis Prodromos 9911 piantoni@ece.auth.gr

25/04/2022

Table of Contents

List of Figures

1 System Requirements

1.1 Functional Requirements (Use Scenarios)

<FR- 1>

The user should be able to enter their details.

Description: The patient should be able to enter their personal information, while the doctor should be able to enter their professional information into the system.

User Priority: 3/5 It is essential for building trust relationships between users, as they need to have basic information about the other users they will interact with.

Technical Priority: 5/5 Some details are necessary for certain system functions.

< FR - 2>

The user should be able to edit their details.

Description: The patient should be able to edit their personal information, while the doctor should be able to edit their professional information.

User Priority: 3/5 It is important for the user to be able to make changes to their details, as some information may no longer be valid.

Technical Priority: 3/5 The system can function without user editing of details since their information has already been entered into the system. However, the presence of invalid information may lead to malfunctions.

< FR - 3>

The notification system should be able to send notifications to the user.

Description: The notification system should be able to send notifications to a user to inform them about the cancellation of an appointment.

User Priority: 4/5 It is essential as users need to be informed about their actions within the system, particularly helpful in facilitating users by reminding them of their upcoming appointments.

Technical Priority: 1/5 The system can function without user notifications, as appointments can be selected, scheduled, and canceled normally, whether there is subsequent user notification or not.

< FR - 4>

The patient should be able to schedule an appointment.

Description: The patient should be able to select an appointment from the available appointments of a doctor and schedule it.

User Priority: 5/5 It is essential as it constitutes the primary use case of the application for all types of users.

Technical Priority: 5/5 It is essential for the system, as it is considered the main purpose of its existence. Without this functionality, the application would be practically useless.

< FR - 5>

The user should be able to cancel a booked appointment.

Description: The user should be able to cancel a booked appointment in case they are unable or do not wish to attend it.

User Priority: 5/5 It is very important for the user to be able to cancel a booked appointment.

Technical Priority: 2/5 The ability to cancel an appointment is not necessary for the smooth operation of the system.

< FR - 6>

The patient should be able to submit a review for a doctor they have visited.

Description: The patient should have the ability to submit a review to the system after completing an appointment with a specific doctor.

User Priority: 3/5 Reviews for doctors are likely to influence the decision of patients, as the opinions of other users are definitely an indicative criterion for choosing the most suitable doctor.

Technical Priority: 1/5 The existence of the rating system does not improve the system's performance.

< FR - 7>

The patient should be able to view the reviews of a doctor.

Description: The patient should be able to view the reviews (if available) submitted by other patients for any doctor in the system.

User Priority: 3/5 Reviews for doctors are likely to influence the decision of patients.

Technical Priority: 1/5 For the system, the existence of reviews and the ability to display them is not a necessary function.

< FR - 8>

The user should be able to see their appointments.

Description: The doctor should be able to see their scheduled appointments and the available appointment slots for any day they choose. The patient should be able to see their completed appointments and their scheduled appointments.

User Priority: 4/5 It is important for the doctor as it provides them with oversight of their appointments, and for the patient, who can review the appointments they have scheduled.

Technical Priority: 2/5 Appointment oversight is not a significant requirement for the smooth operation of the system.

< FR - 9>

The patient must be able to search for doctors in the system.

Description: The patient should be able to search for doctors in the system using one or more search criteria.

User Priority: 5/5 It is important for the patient because it is the main reason why they would want to use the system.

Technical Priority: 5/5 The system cannot function without the search step, as the user cannot access a doctor's profile and therefore cannot make an appointment.

< FR - 10>

The patient must be able to sort the doctors in the search results.

Description: The patient should be able to sort the doctors in the search results using a sorting criterion.

User Priority: 3/5 It is important for the patient as it facilitates their search, but it is not essential for the service.

Technical Priority: 1/5 It is not a necessary function for the system.

< FR - 11>

The user must be able to submit a report to the system.

Description: The user should be able to submit a report and thus communicate with the system administrators.

User Priority: 3/5 It is considered an important function for users since it is their only form of communication with the system administrators.

Technical Priority: 1/5 It is not necessary for the operation of the system.

< FR - 12>

The doctor must be able to modify the available appointment slots.

Description: The doctor should be able to modify the available appointment slots by specifying the working days and hours during which their clinic is open and accepts appointments. Initially, the doctor has no available appointment slots.

User Priority: 5/5 It is considered an important function for users since clinics often have different operating hours.

Technical Priority: 3/5 The system can still operate even if it assumes arbitrarily that all clinics have the same operating hours.

< FR - 13>

The user should be able to edit the photos related to their profile.

Description: The doctor should be able to edit the profile photo of themselves as well as the photo of their clinic, and the patient should be able to edit their profile photo. Initially, these photos have the same icons for all users.

User Priority: 2/5 It is not considered important for users, but it provides an image of the doctor and the clinic they will visit.

Technical Priority: 1/5 It is not necessary for the system to function.

< FR - 14>

The doctor should be able to change the status of an appointment from available for booking to closed.

Description: The doctor should be able to inform the system that an appointment is no longer available because it has been booked outside the system.

User Priority: 5/5 It is important for users, as otherwise, double booking for the same appointment would be inevitable.

Technical Priority: 2/5 It is not necessary for the system to function.

1.2 Users and External Systems

1.2.1 <Patient>

The user, as a patient, has an account and enters the application with the purpose of finding the suitable doctor for their needs. The patient has the ability to edit their personal information. Additionally, they can schedule appointments with a doctor of their choice and view a summary of the appointments they have booked.

1.2.2 <Doctor>

The user, as a doctor, provides their medical services within the application. They have the ability to edit their personal and professional information and can also view the list of their appointments.

1.2.3 <Notification System>

The notification system is used to send notifications to users' devices in various cases, such as reminding them of an upcoming appointment or informing them of the cancellation of a booked appointment.

1.2.4 <doctors DB>

Database where all the information related to doctor-type users is stored. This includes their professional details, the reviews written on their profiles, as well as their available/closed appointments.

1.2.5 < patients DB>

Database where all the information related to patient-type users is stored. This includes their personal details, as well as their closed appointments.

1.2.6 <Βάση δεδομένων αναφορών: reports DB>

Database where all the reports made by any user are stored. Reports can be about a user of the system or even the system itself if it experiences functional issues.

1.2.7 <Google Maps API>

Use of this external system to find addresses on the map as well as to calculate distances based on a specific address. No map is displayed within the application.

1.2.8 <Photos Collection API>

Use of this external system to display the photos stored on the device and the ability to select from them.

1.3 Significant Non-Functional Requirements

Performance requirements

<NFR- 1>

The system must have a response time of less than 200 milliseconds.

Description: The response time must be kept small so that the user's navigation within the system is pleasant and with minimal delays.

User priority: 4/5 The user wants to use a system without unnecessary delays.

Technical priority: 2/5 The application operates normally even if there are some delays, as long as they do not significantly exceed the limit.

Security requirements

<NFR- 2>

The system must comply with the current data protection regulations set by GDPR.

Description: The system should protect users' personal data.

User priority: 3/5 The user wants their personal data to be protected and to feel secure while using the application, although they do not directly benefit from any additional functionality.

Technical priority: 5/5 Failure to meet this requirement will force the application to be disabled.

Usability requirements

<NFR- 3>

The system must work without issues on any device running Windows 7 or a newer version, and Android 8 or a newer version.

Description: Any device using Windows 7 or newer / Android 8 or newer should be able to use the system without encountering compatibility issues or any technical problems.

User priority: 3/5 Users who own certain devices should be able to be served by the system without facing problems.

Technical priority: 3/5 The system can still operate even if it runs on a single operating system.

<NFR- 4>

The system must be able to display the user's appointments in a calendar format.

Description: .

User priority: 3/5 The user benefits from viewing appointments in a calendar format, but they could also schedule appointments in other presentation formats.

Technical priority: 1/5 The system could operate just as smoothly without adding the appointment display in calendar format.

Reliability requirements

<NFR- 5>

The system must be able to operate without problems for up to 10,000 concurrent users.

Description: The system should be capable of handling a specific number of users who wish to navigate it.

User priority: 3/5 It affects the user's experience, as they want the system to be reliable and able to serve them without delays, regardless of external factors.

Technical priority: 3/5 It affects the system in cases where the load is so high that it cannot cope and becomes unavailable.

1.10 Glossary

User: Anyone who enters the system with the purpose of using it.

System: All the components and systems that work together for the smooth operation of the application.

Personal Information: Name, surname, gender, age, mobile phone number, home address.

Professional Information: Name, surname, gender, age, office landline number, specialization, office address, consultation cost, and a brief description of oneself.

Appointment: A specific date and time at which a meeting between a patient and a doctor is scheduled.

Closed Appointment: An appointment selected by a patient from the list of available appointments of a doctor to visit them.

Available for Reservation Appointment: An appointment that the doctor has designated as available for booking, regardless of whether it has been booked by a patient.

Sending Notifications: Done through push notifications.

Appointment Completion: When the predetermined date of a reserved appointment passes without cancellation, the system considers the appointment to be successfully completed.

Appointment Cancellation: The cancellation of a closed appointment by the patient, making it available again.

Closest Available Appointment: The date for the nearest available appointment with a doctor.

Appointment Status: Designates an appointment as available for booking or closed.

Review: An evaluation of the doctor by the patient. The review includes a rating on a scale from 1 to 5 and can optionally be accompanied by a brief text.

Search Criteria: Doctor's address, consultation cost, doctor's rating, appointment availability, doctor's specialization, doctor's name, distance from the patient's address to the doctor's address.

Sorting Criteria: The sorting criteria can be one of the following: by consultation cost in ascending order, by consultation cost in descending order, by doctor's rating.

Search Result: The set of doctors in the system that meet all the criteria of a patient's search, presented in a way that satisfies the sorting criteria.

Report: Text that can refer either to the behavior of another user that violates the rules or to an issue that the user encountered while using the application.

Terms: Refers to the Terms of Service of the application, which are the legally binding terms that the user accepts upon entering the system.

Response Time: The total time to complete a function in the application, triggered by a user's action.

Concurrent Users: The total number of users, both doctors and patients, who have the application active and are performing actions in it at a specific point in time.

2 Use Cases

2.1 Use Case Diagrams

2.2 Feature 1: Enter user details

Background:
Given that I am logged in user
And I have just logged in for the first time
And I am in "User Profile" page
Scenario: Successfully entering details as patient
Given that I am logged in as patient
And that I am asked to enter my personal details
When I enter my personal details
And they are compliant with the "Patient's profile details form":
First name	String	'George'	at least 3 chars and at most 30 chars
Last name	String	'Brown'	at least 3 chars and at most 30 chars
Gender	String	'Male'	
Age	Integer	21	between 18 and 120

| Phone number | String |'6948083247' | | |
| Home address | String |'Mpoumpoulinas 30, 62222' | | |
And I click the "Done" button
Then I should be redirected to "Home" page

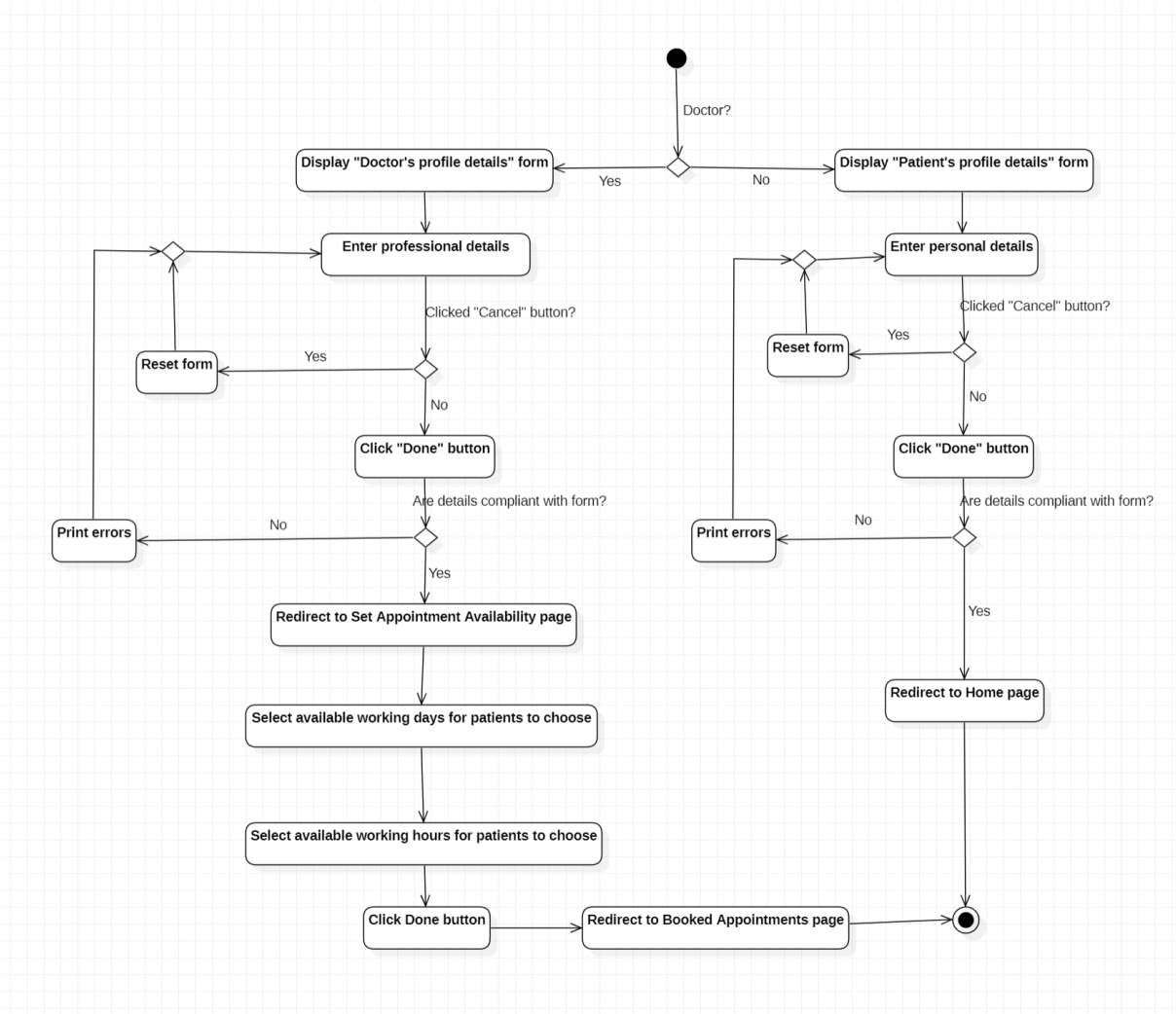
Scenario: Unsuccessfully entering personal details as patient
Given that I am logged in as patient
And that I am asked to enter my personal details
When I enter my personal details
And they are not compliant with the "Patient's profile details form"
And I click the "Done" button
Then I should be prompted to correct my mistakes

Scenario: Successfully entering details as doctor
Given that I am logged in as doctor
And that I am asked to enter my professional details
When I enter my professional details
And they are compliant with the following "Doctor's profile details form":
First name	String	'George'	at least 3 chars and at most 30 chars
Last name	String	'Brown'	at least 3 chars and at most 30chars
Gender	String |'Male'		
Age	Integer| 21	between 18 and 120	
Phone number	String |'6948083247'		
Clinic address	String |'Mpoumpoulinas 30, 62222'		
Specialization	String |'Eye Specialist'		
Visit Cost	Number	30	minimum of 30	
And I click the "Done" button
Then I should be redirected to the "Set Appointment Availability" Page
When I select all the available working days for patients to choose
And I select all the available working hours for patients to choose
And I press the "Done" button
Then I should be redirected to the "Booked Appointments" Page

Scenario: Unsuccessfully entering details as doctor
Given that I am logged in as doctor
And that I am asked to enter my professional details
When I enter my professional details
And they are not compliant with the "Doctor's profile details form"
And I click the "Done" button
Then I should be prompted to correct my mistakes

Scenario: Clicking on "Cancel" button
When I click "Cancel" button
Then the form is getting empty

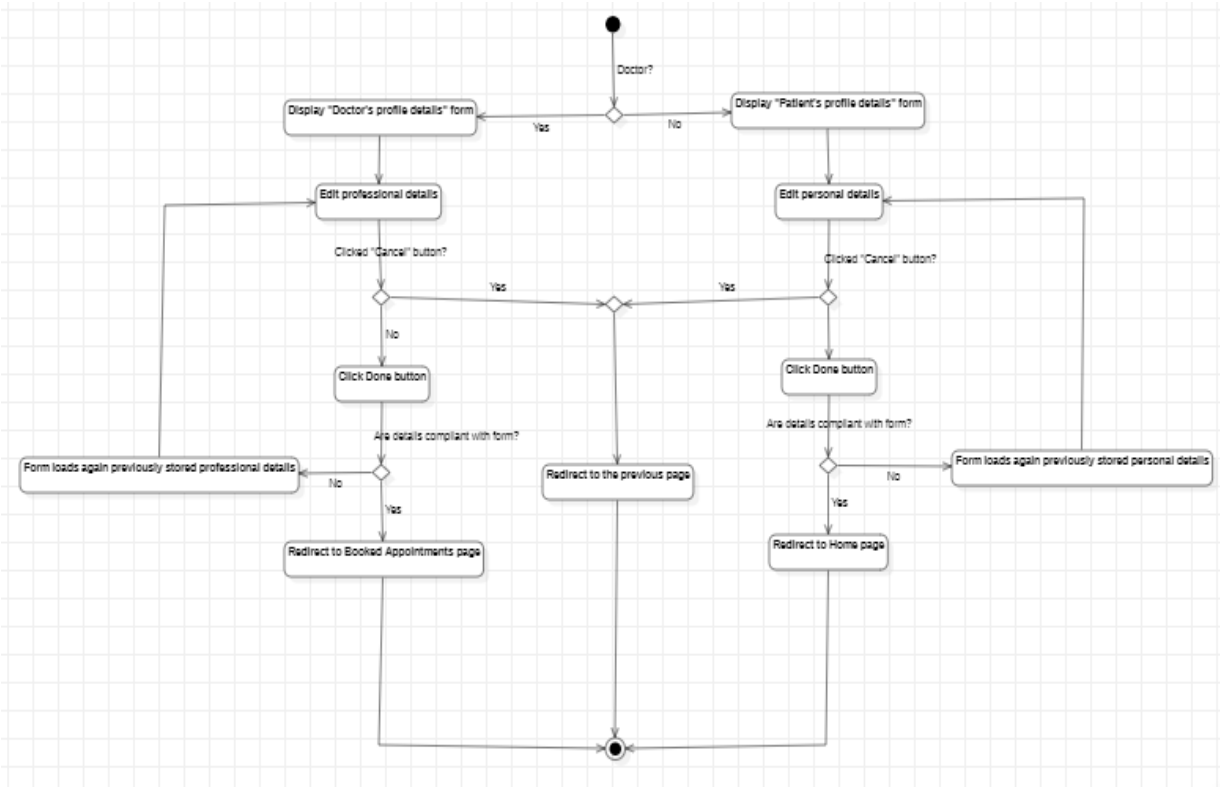
And I should be prompted to fill the form again



2.3 Feature 2: Edit user details

Background:
And I am in "User Profile" page
And That I clicked on the "Edit details" button
Scenario: Successfully editing details as patient
Given that I am logged in as patient
And that I want to change my personal details
When I make my changes
And they are compliant with the "Patient's profile details form":
First name	String	'George'	at least 3 chars and at most 30 chars
Last name	String	'Brown'	at least 3 chars and at most 30 chars
Gender	String	'Male'	

Age	Integer	21	between 18 and 120
Phone number	String	'6948083247'	
Home address	String	'Mpoumpoulinas 30, 62222'	
And I click the "Done" button
Then I should be redirected to the "Home" Page
Scenario: Unsuccessfully editing details as patient
Given that I am logged in as patient
And that I want to change my personal details
When I make my changes
And they are not compliant with the "Patient's profile details form"
And I click the "Done" button
Then I should be prompted to correct the mistakes
Scenario: Successfully editing details as doctor
Given that I am logged in as doctor
And that I want to change my professional details
When I make my changes
And they are compliant with the "Doctor's profile details form":
First name	String	'George'	at least 3 chars and at most 30 chars
Last name	String	'Brown'	at least 3 chars and at most 30 chars
Gender	String	'Male'	
Age	Integer	21	between 18 and 120
Phone number	String	'6948083247'	
Clinic address	String	'Mpoumpoulinas 30, 62222'	
Specialization	String	'Eye Specialist'	
Visit Cost	Number	30	minimum of 30
And I click the "Done" button
Then I should be redirected to the "Booked Appointments" Page
Scenario: Unsuccessfully editing details as doctor
Given that I am logged in as doctor
And that I want to change my professional details
When I make my changes
And they are not compliant with the "Doctor's profile details form"
And I click the "Done" button
Then I should be prompted to correct my mistakes
Scenario: Exiting from editing profile details
When I press "Cancel" button
Then I am redirected to the previous page



2.4 Feature 3: Book Appointment

Background:

Given that I am a patient

And I have selected a doctor

And I have clicked on the "Appointments" button

And I am redirected to the "Book Appointment" page

Scenario: Successfully Booking Appointment

When I choose one or multiple dates from the calendar

And I select one of the available appointments listed that match the dates selected

And I click on the "Book Appointment" button

Then selected appointments have been tagged as "Booked"

And I am redirected to the "Cancel Appointment" page

Scenario: Unsuccessfully booking appointment because no available appointment is selected

When I choose one or multiple dates from the calendar

And I don't select any of the available appointments

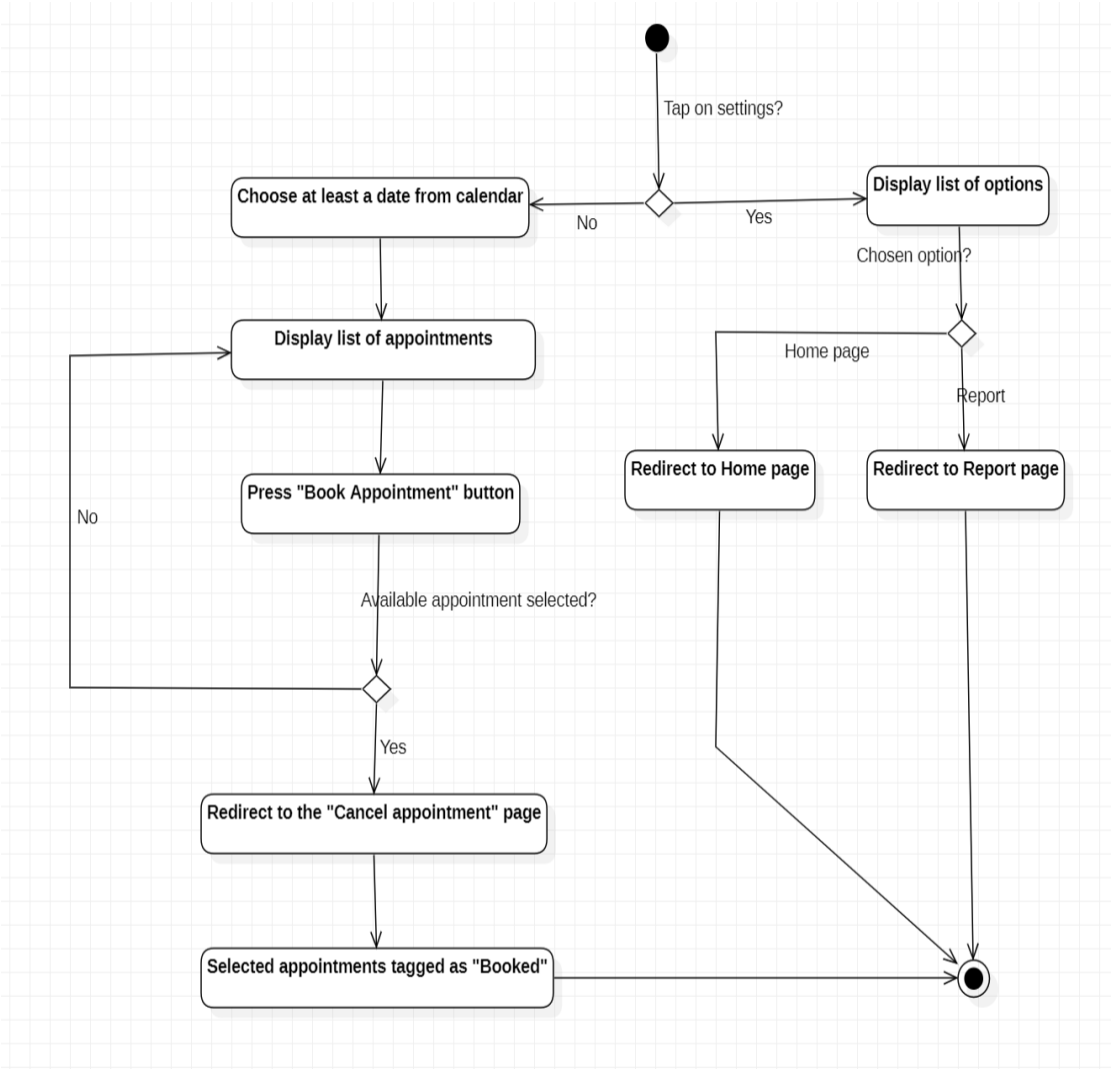
And I click on the "Book Appointment" button

Then I should be prompted to select an appointment

Scenario: Changed my mind about booking an appointment with this doctor

When I tap on "Settings" button

And I can view the list of available options
And I press the "Home Page" option
Then I am redirected to the "Home" page
Scenario: Reporting an event
When I tap on "Settings" button
And I can view the list of available options
And I press the "Report" option
Then I am redirected to the "Report" page



2.5 Feature 4: Submit Report

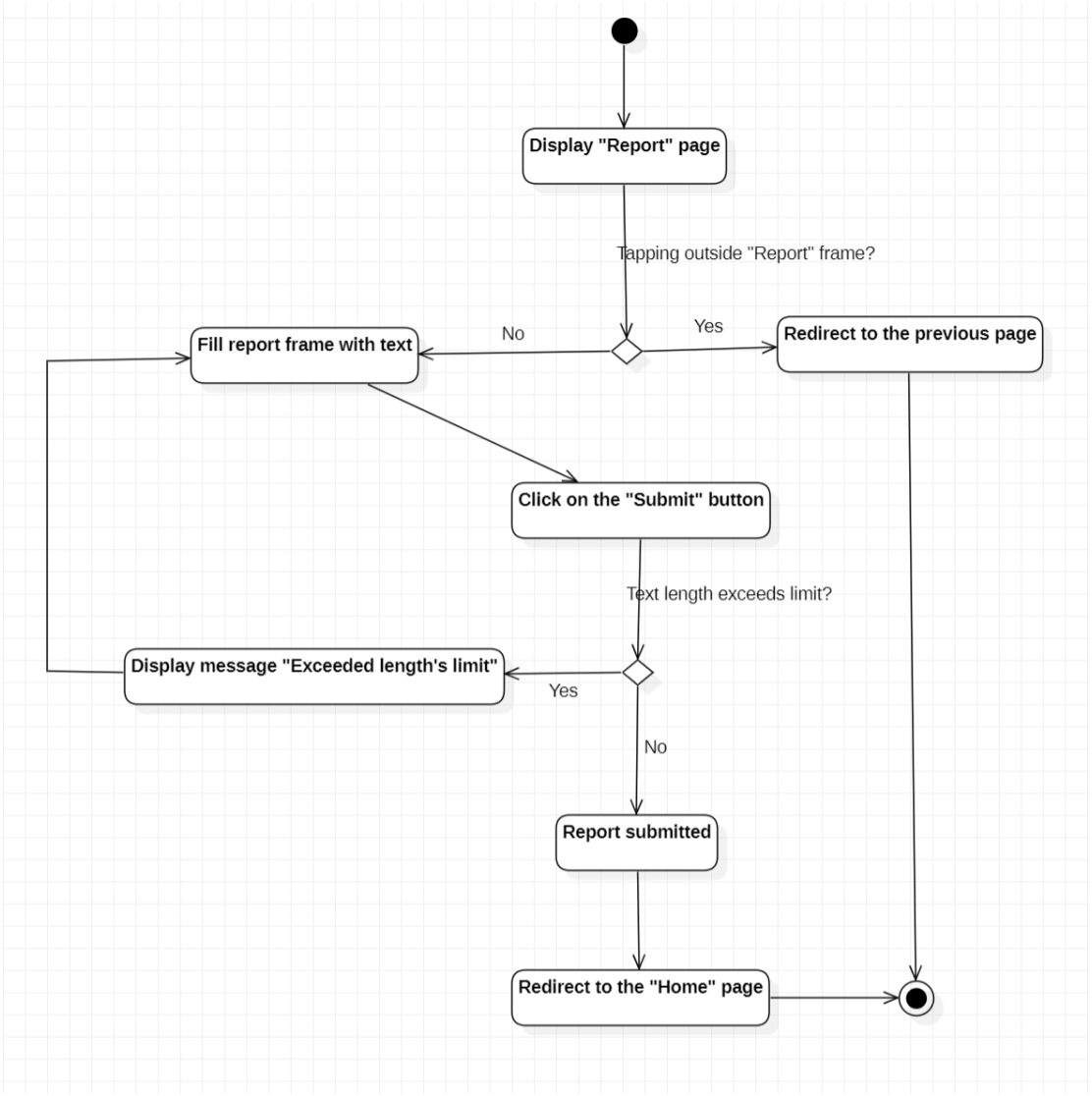
Background:
Given that I am a logged in user

And I have clicked on a "Report" button
And I am redirected to the "Submit Report" page

Scenario: Accidentally clicking on the "Report" button
Given that I do not want to submit a report
When I tap outside the margin of the "Report" frame
Then I should be redirected to the previous page

Scenario: Exceeding the character limits
Given that I have selected the report frame where writing is allowed
When I click on the "Submit" button
And I have exceeded the 400 characters limit
Then there should be a message displaying "Report must be less than 400 characters long"

Scenario: Successfully Submitting a user's report
Given that I have selected the report frame where writing is allowed
And that I have filled it with my report
When I click on the "Submit" button
Then I should be redirected to the "Home" page
And my report should be sent to the administrators of the system

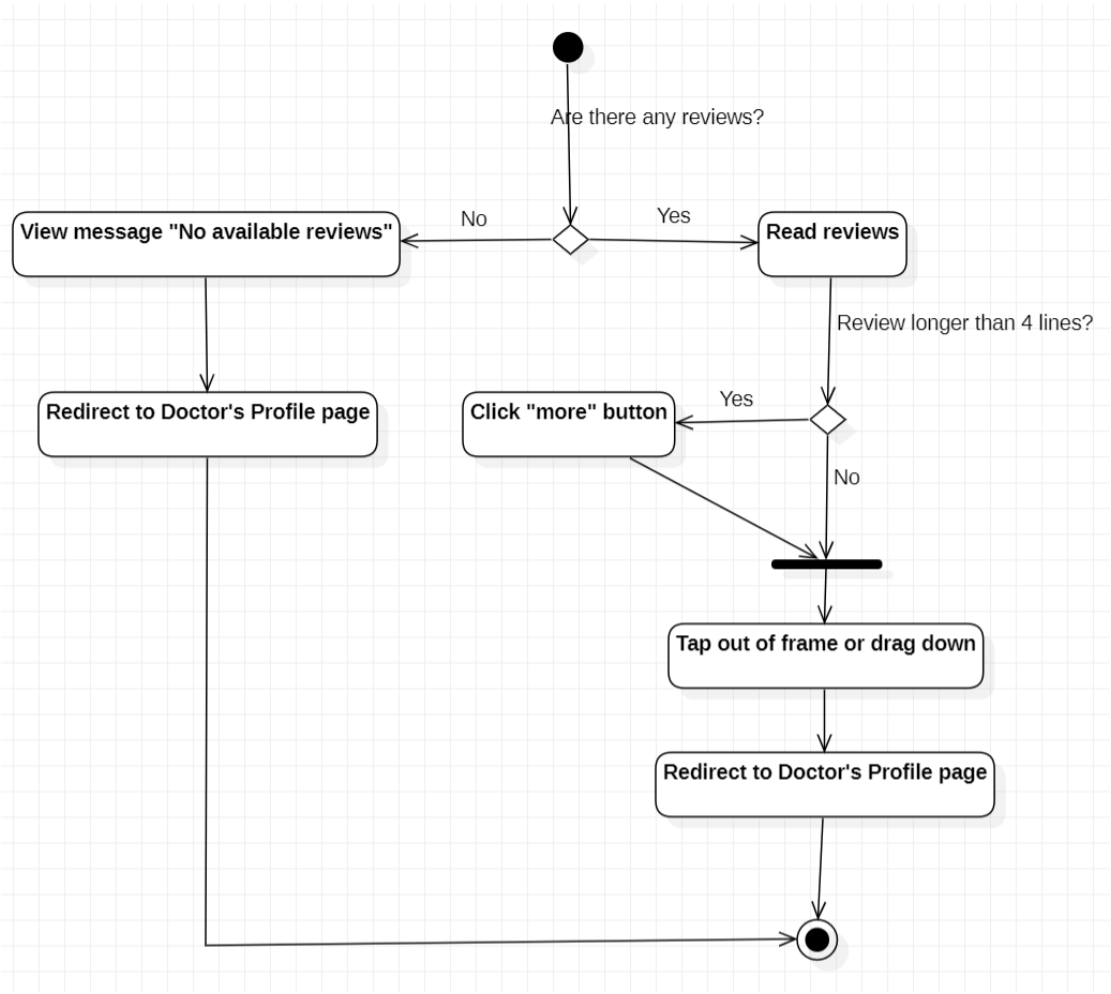


2.6 Feature 5: Read doctor reviews

Background:
Given that I am a patient
And I have reached a "Doctor's Profile" Page
And I have clicked on the "Reviews" button
Scenario: Reviews list empty
When I click Reviews button but there are not reviews
Then I should see the message "No availiable reviews"
And I should be redirected to the "Doctor's Profile" Page
Scenario: Text expand
Given I am reading a review that is longer than 4 lines
When I click on the "more" button

Then the text should expand and the rest of the review should appear

And I should be able to go back to the "Doctor's Profile" Page by tapping out of the frame or dragging down



2.7 Feature 6: Write review for a doctor

Background:

Given that I am a patient

And the date of my booked appointment with a doctor has passed

And I am in the "List of Booked Appointments" page

And I clicked on the "Review" button of the completed appointment

Scenario: Successfully writing a review for a doctor

When I tap on one of the five heart symbols, rating the doctor in a scale from one to fivee

And I select the review frame where writing is allowed

And I fill it with my review between 0 and 400 characters

And I press the "Submit" button

Then I should be redirected to the "Doctor's Reviews" page

And my review should be added and displayed first in the list of reviews in the "Doctor's Reviews" Page

Scenario: Unsuccessfully writing a review for a doctor due to no heart symbol chosen

When I don't select one of the five heart symbols

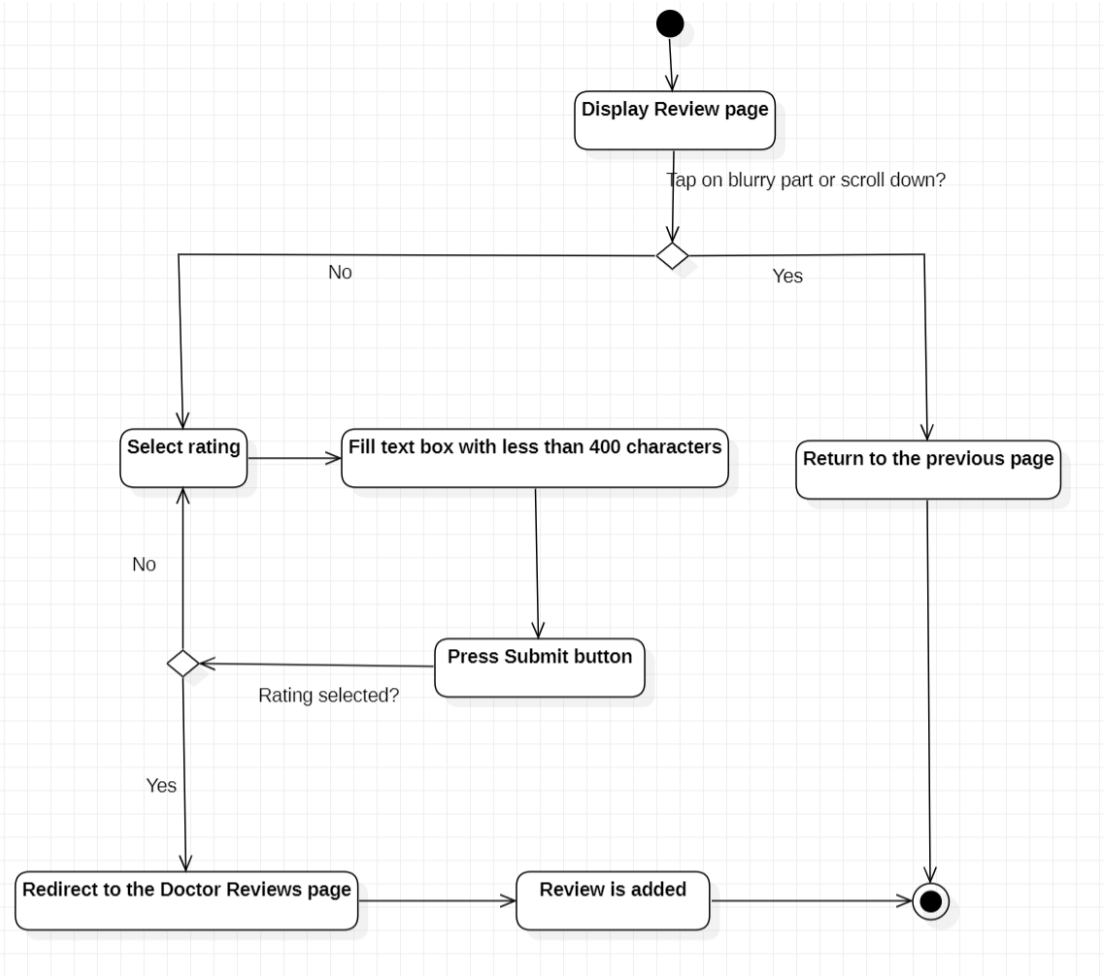
And I press "Submit" button

Then I should be prompted to rate by selecting a heart as it is mandatory in order to submit my review

Scenario: Cancel writing a review for a doctor

When I tap on the blurry part of the screen or scroll down

Then I should be redirected to the previous page



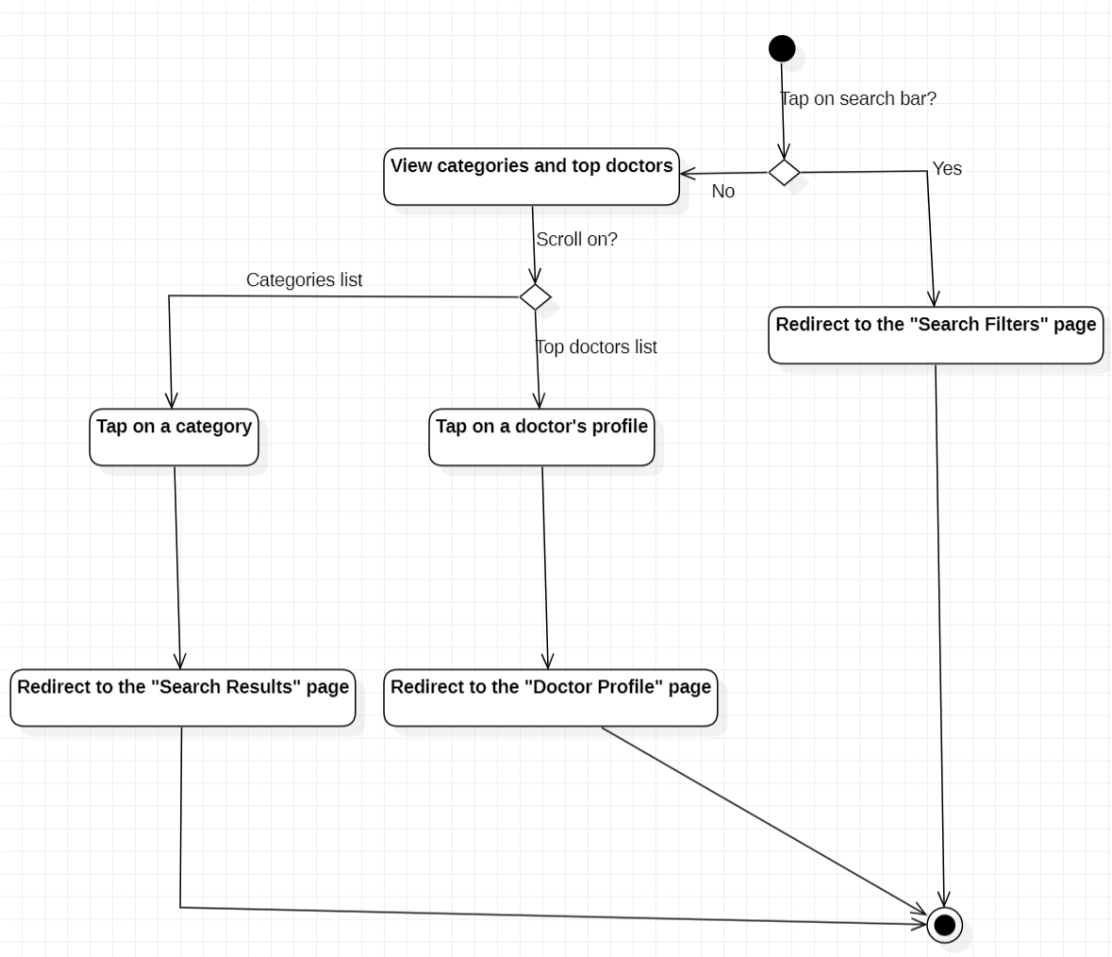
2.8 Feature 7: Search for a doctor

Background:

Given that I am a patient

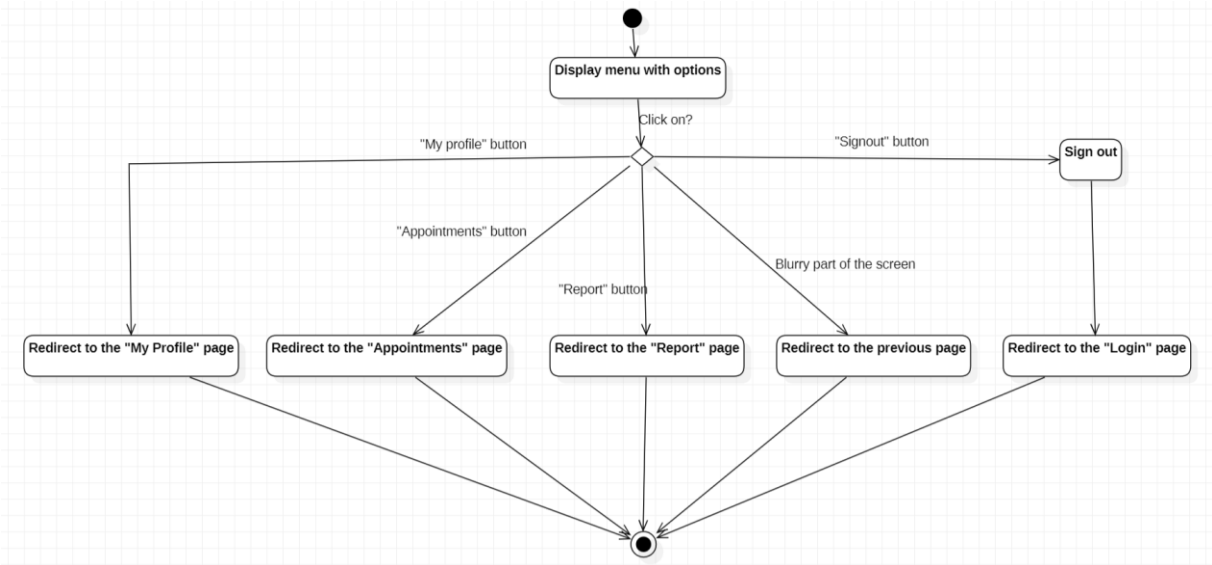
And I am in the "Home" page

Scenario: Choosing a doctor from the "Top Doctors" list
Given that I am scrolling through the "Top Doctors" list
When I tap on a doctor's profile from the "Top Doctors" list
Then I should be redirected to the "Doctor's Profile" page
Scenario: Choosing a doctor from the "Categories" list
Given that I am horizontally scrolling through the "Categories" list
When I select a category
Then I should be redirected to the "Search results" page
Scenario: Searching for a doctor using filters
When I tap on the "Search bar"
Then the "Search Filters" frame pops up
Scenario: exiting the filters frame without searching
Given that I tapped on the "Search bar"
And the "Search Filters" frame popped up



2.9 Feature 8: Patient menu roaming

Background:
Given that I am a patient
And I have clicked on the "Profile Picture"
And I am redirected to the "Menu" page
Scenario: Clicking on the "My Profile" button
When I click on the "My Profile" button
Then I am redirected to the "My Profile" page
Scenario: Clicking on the "Appointments" button
When I click on the "Appointments" button
Then I am redirected to the "Appointments" page
Scenario: Clicking on the "Report" button
When I click on the "Report" button
Then I am redirected to the "Report" page
Scenario: Clicking on the "Sign out" button
When I click on the "Sign out" button
Then I am signed out from the app
And I am redirected to the "Login" page
Scenario: Clicking on the blurry part of the screen
When I click on the blurry part of the screen
Then I am redirected to the previous page where I had clicked on the "Profile Picture"



2.10 Feature 9: Search results

Background:

Given that I am a patient

And I am in "Search Results" page

Scenario: Selecting a doctor

When that I am scrolling in list of doctors

And I tap on a doctor that I like

Then I am redirected to the "Doctor's Profile" page

Scenario: Applying new filters

When I tap on the search bar

Then I am redirected to the "Search filters" page

Scenario: Switching to another sorting method

When I check on another sorting method from the options "Rating/Decreasing cost/Increasing cost"

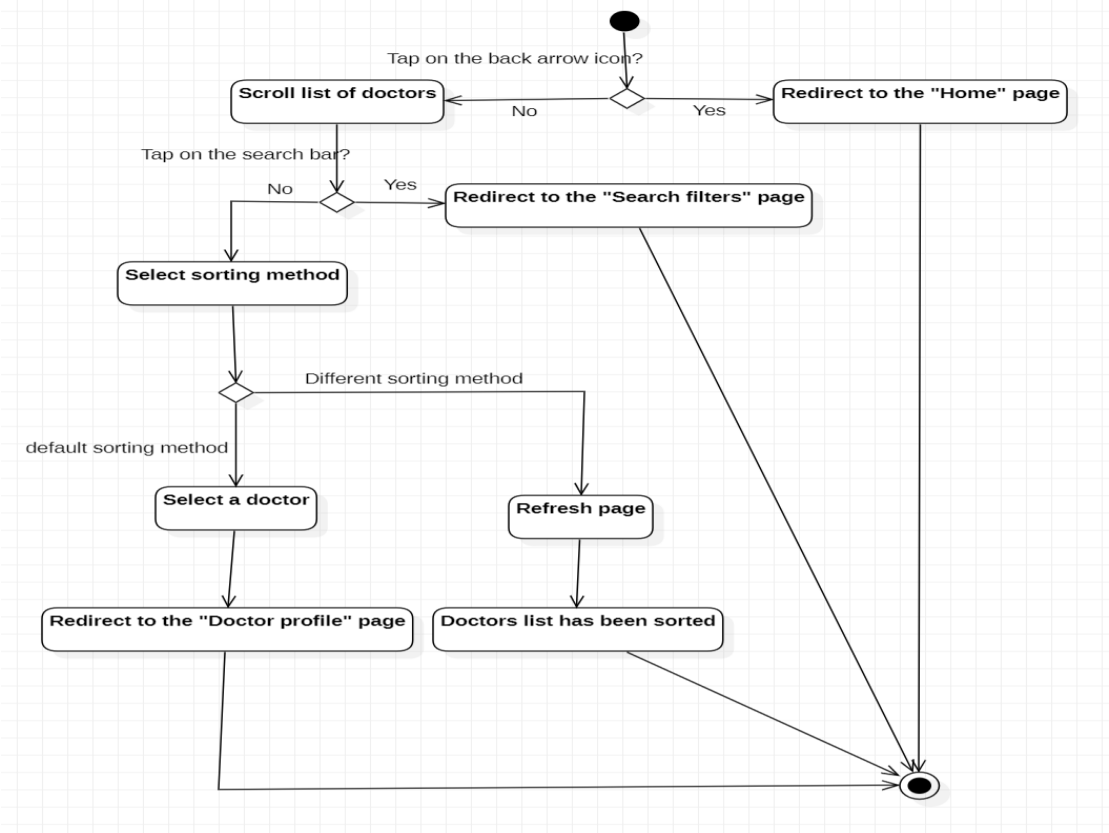
Then page is refreshed

And sorting method is applied

Scenario: Returning to "Home" page

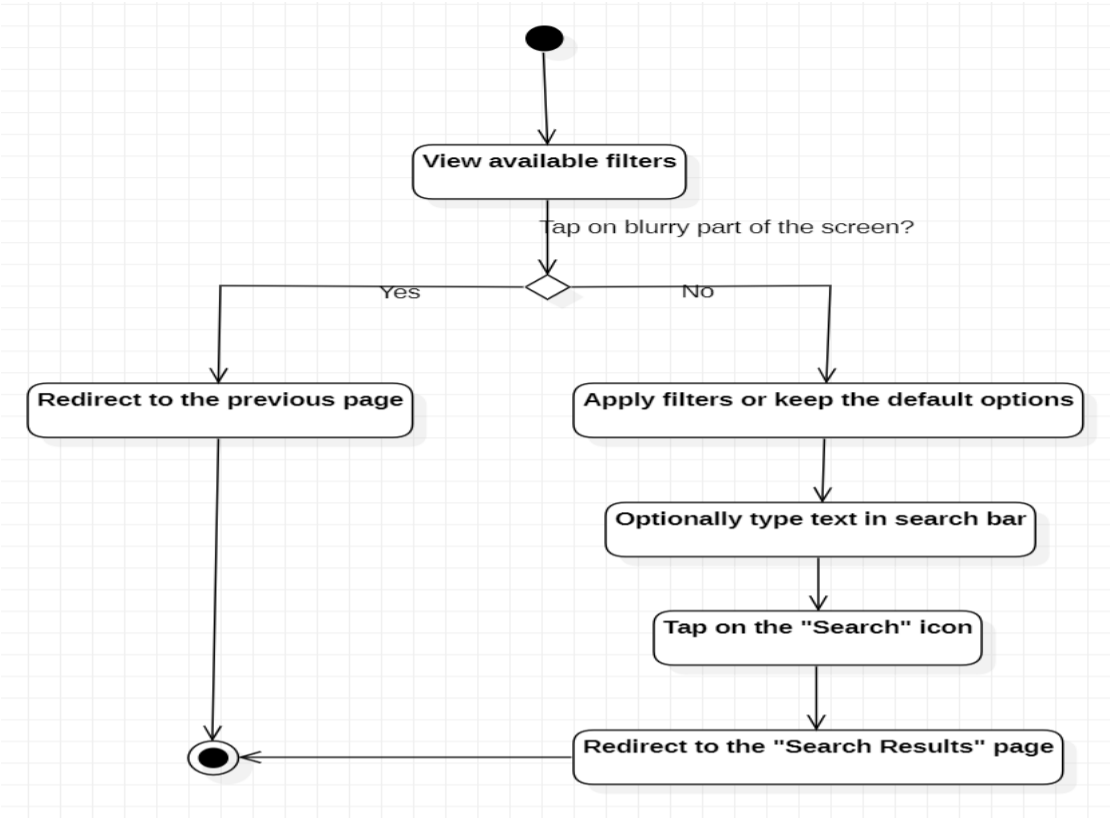
When I tap on "Back arrow" icon in upper left corner of the screen

Then I am redirected to the "Home" page



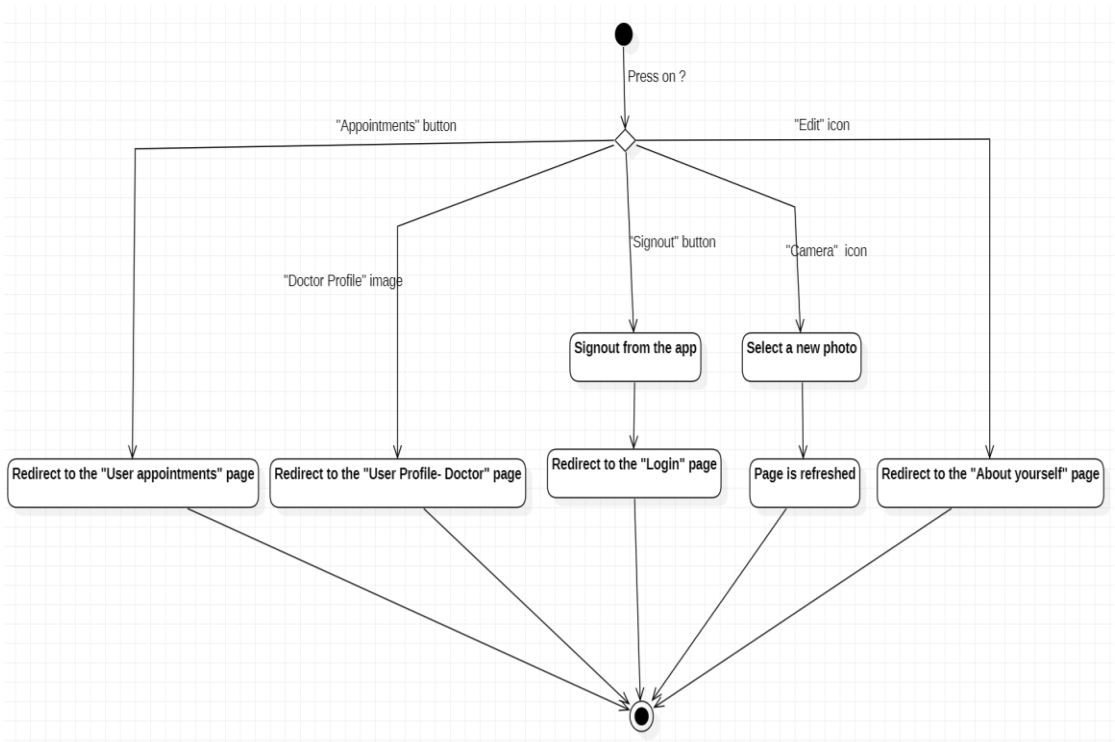
2.11 Feature 10: Apply filters

Background:
Given that I am a patient
And I am in "Search filters" page
Scenario: Searching for a doctor using filters and text
When I apply one or more of the available filters or left them as default
And I type some text in the search bar that can match one or more of the following:
Specialty	String	'Pathologist'
First Name	String	'Giorgos'
Last Name	String	'Papadopoulos'
Address	String	'Mpoumpoulinas 30, 62222'
And I tap on the "Search" icon
Then I should be redirected to the "Search results" page
Scenario: Searching for a doctor using filters and no text
When I apply one or more of the available filters or left them as default
And I tap on the "Search" icon
Then I should be redirected to the "Search Results" page
Scenario: Exiting from "Search filters" page
When I tap on the blurry part of the screen
Then I should be redirected back to the previous page



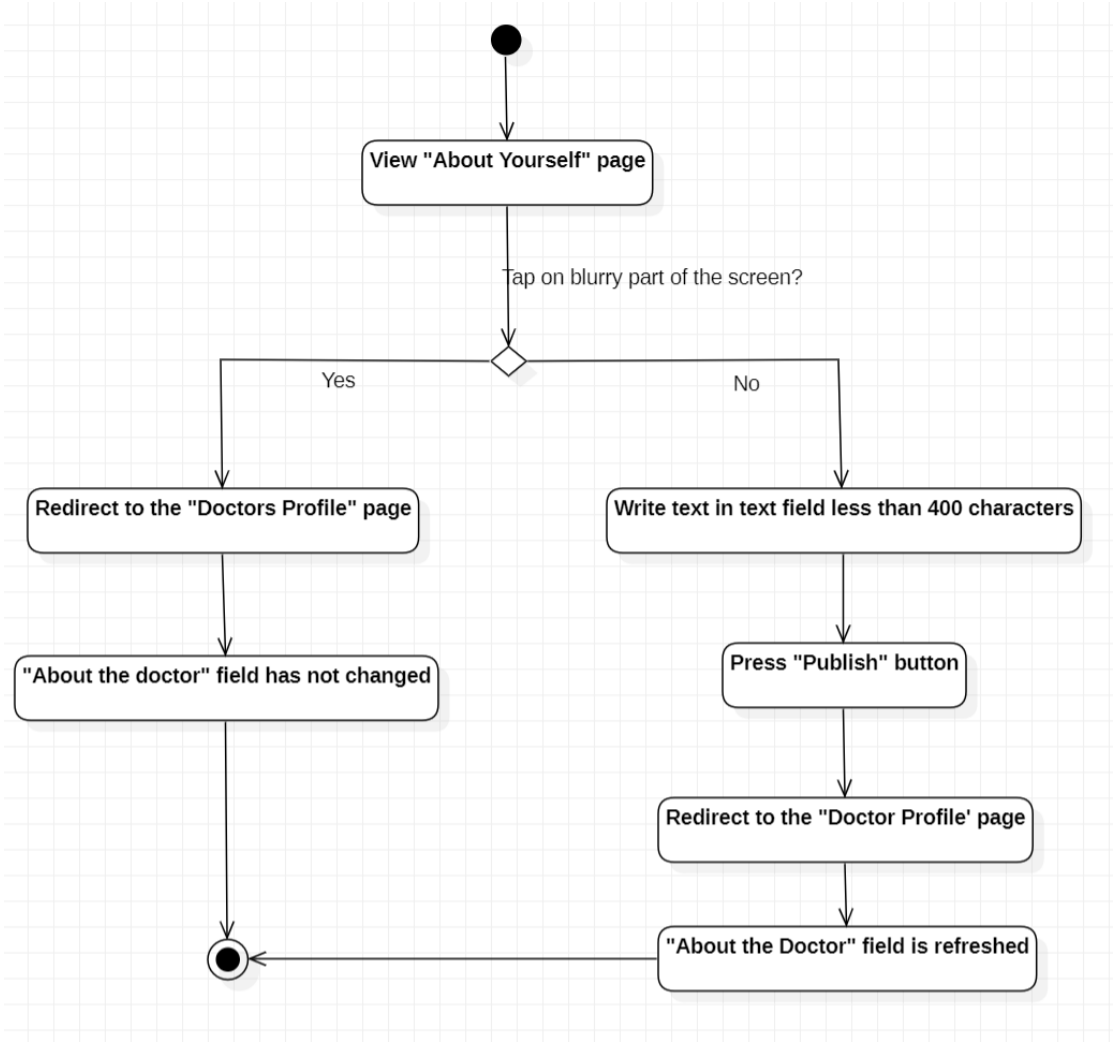
2.12 Feature 11: Doctor profile navigation

Background:
Given that I am a doctor
And I am in "Doctors Profile" page
Scenario: Previewing available and booked appointments
When I press "Appointments" button
Then I am redirected to the "User appointments" page
Scenario: Editing professional details
When I press the "Doctor's Profile" image
Then I am redirected to the "User Profile- Doctor" page
Scenario: Signing out
When I press the "Sign out" button
Then I sign out
And I am redirected to the "Login" Page
Scenario: Changing office image
When I press the "Camera" icon
Then I can select a new photo for the office's photo slot
And the page is refreshed
Scenario: Editing "About the Doctor" text
When I press the "Edit" icon
Then the "About yourself" page pops up



2.13 Feature 12: Doctor's About Yourself text

Background:
Given that I am a doctor
And I am in "About Yourself" page
Scenario: Successfully editing "about yourself" text
Given that i have written my text in text field
And it contains less than 400 characters
When I press "Publish" button
Then I am redirected to the "Doctors Profile" page
And "About the doctor" field is refreshed
Scenario: Exiting from "About Yourself" page
When I tap on blurry part of the screen
Then I am redirected to the "Doctors Profile" page
And "About the doctor" field has not changed

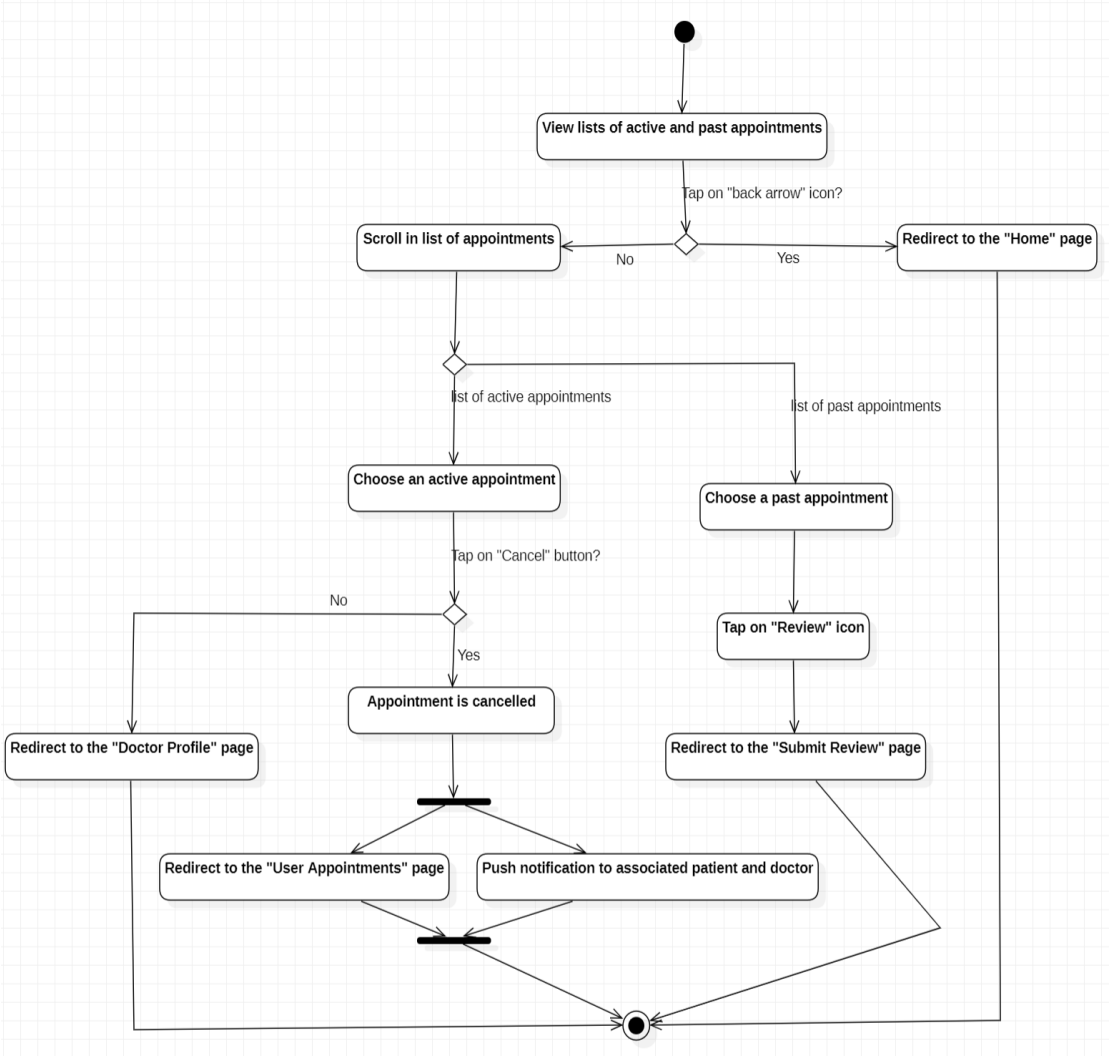


2.14 Feature 13: Patient – View appointments

Background:
Given that I am a patient
And I am in "User Appointments" page
And I can see list of active and past appointments
Scenario: Reviewing an appointment
Given that I am navigating in list of past appointments
When I tap on the "Review" icon of a doctor
Then I am redirected to the "Submit Review" page
Scenario: Selecting an active appointment
When I press on a doctor from the "Active appointments" list
Then I am redirected to the "Doctors Profile" page
Scenario: Cancelling an active appointment

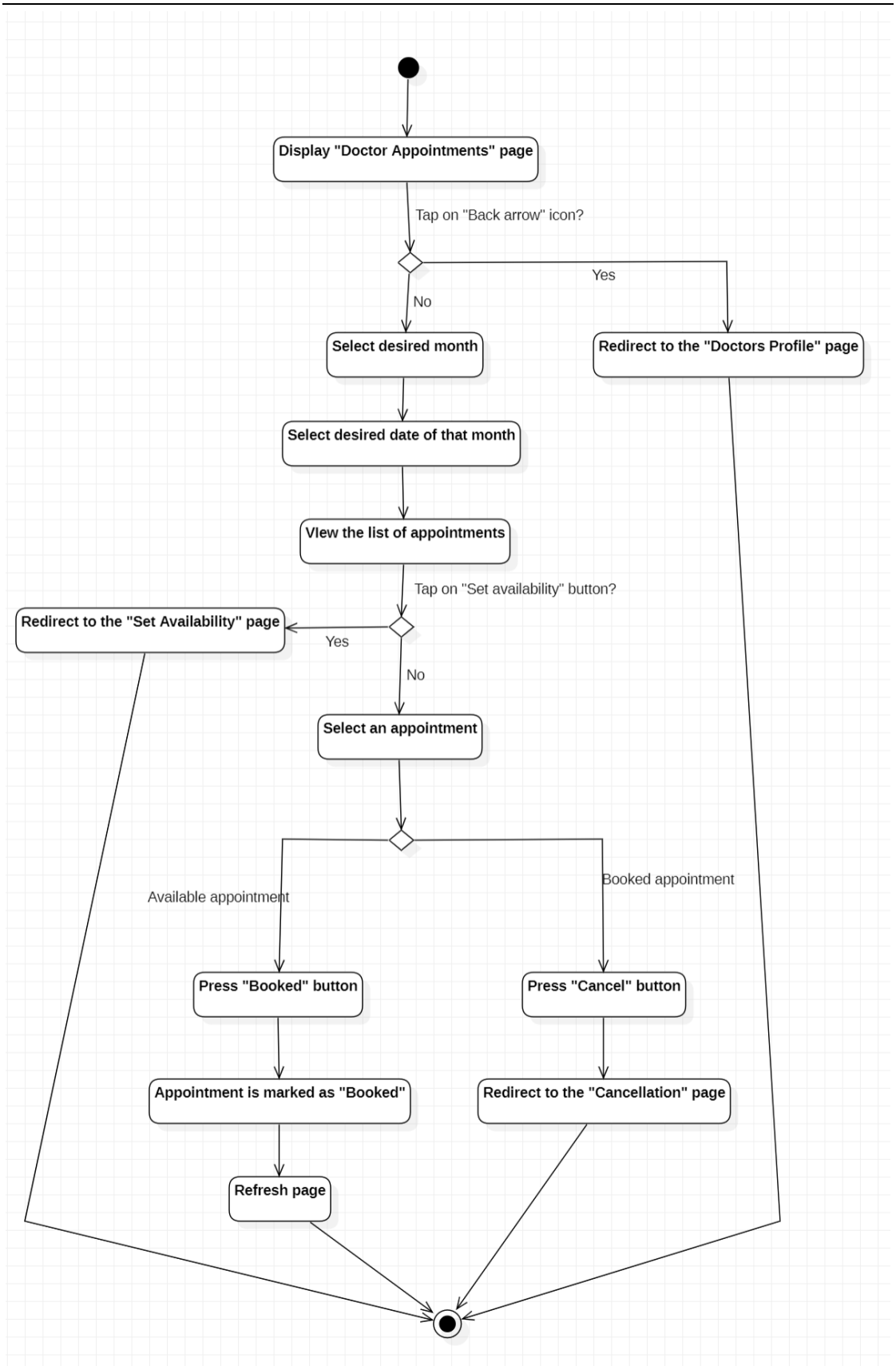
Given that I am navigating in list of active appointments
And I have selected an appointment
When I tap on "Cancel" button of that appointment
Then appointment is cancelled
And I am redirected to the "User appointments" page
And notification system sends notification to associated patient and doctor

Scenario: Returning to home page
When I tap on "Return" icon
Then I am redirected to the "Home" page



2.15 Feature 14: Doctor – View appointments

Background:
Given that I am a doctor
And I am in "Doctor Appointments" page
And I have selected the desired month
And I have selected the desired date of that month
And I can see the list of appointments of that day
Scenario: Changing available working hours
When I press on "Set Availability" button
Then I am redirected to the "Set Availability" page
Scenario: Cancelling a booked appointment
When I select an booked appointment
And I press "Cancel" button
Then I am redirected to the "Cancelation" page
Scenario: Booking an available appointment
When I select an available appointment
And I press "Booked" button
Then page is refreshed
And the appointment is marked as "Booked"
Scenario: Returning to profile page
When I tap on "Back arrow" icon
Then I am redirected to the "Doctors Profile" page


Display "Doctor Appointments" page
Tap on "Back arrow" icon?
Yes
No
Select desired month
Redirect to the "Doctors Profile" page
Select desired date of that month
Vlew the list of appointments
Tap on "Set availability" button?
Redirect to the "Set Availability" page
Yes
No
Select an appointment
Available appointment
Booked appointment
Press "Booked" button
Press "Cancel" button
Appointment is marked as "Booked"
Redirect to the "Cancellation" page
Refresh page

2.16 Feature 15: Doctor Cancel appointment

Background:

Given that I am a patient

And I am in "Cancellation" page

Scenario: Successfully canceling an appointment with that doctor

When I press on "Yes" button

Then I am redirected to the "Cancel Appointment" page

And the previously booked appointment has been tagged as "Available"

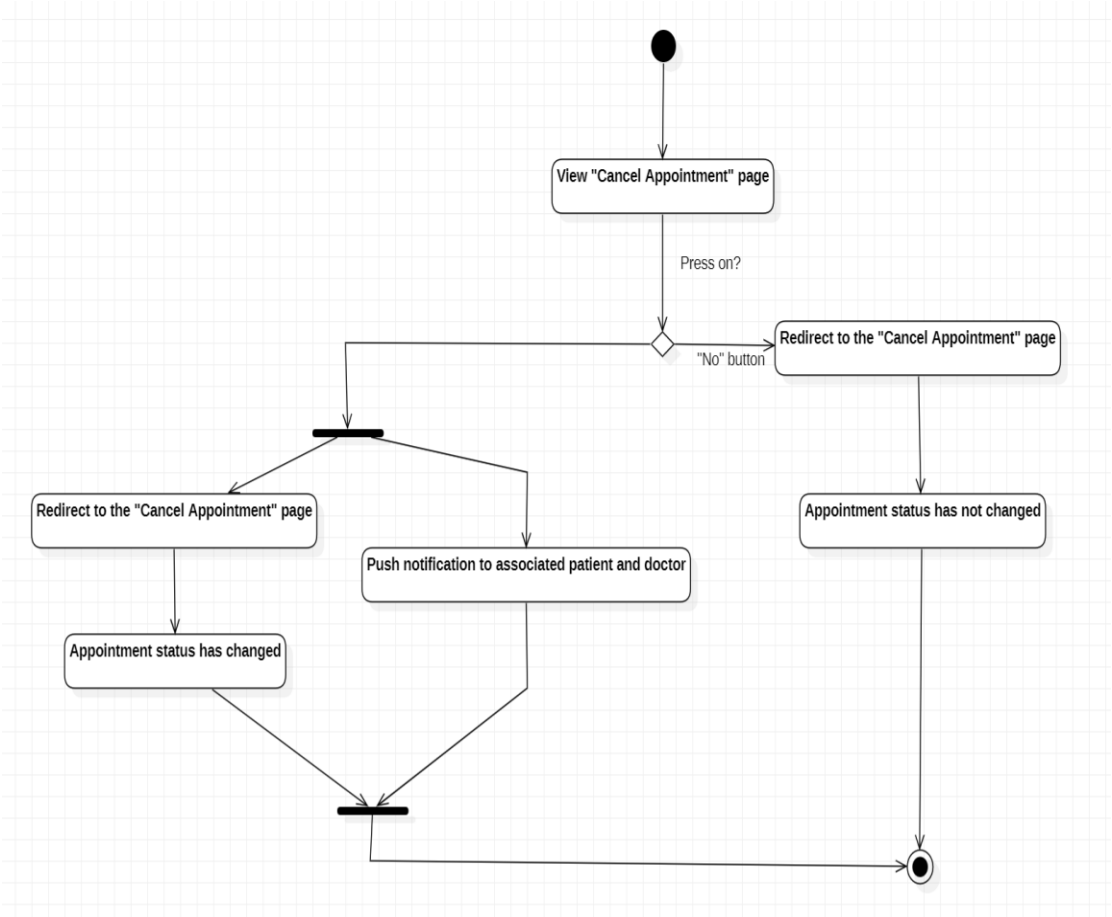
And the notification system pushes a notification to the associated patient and doctor

Scenario: Returning to previous page

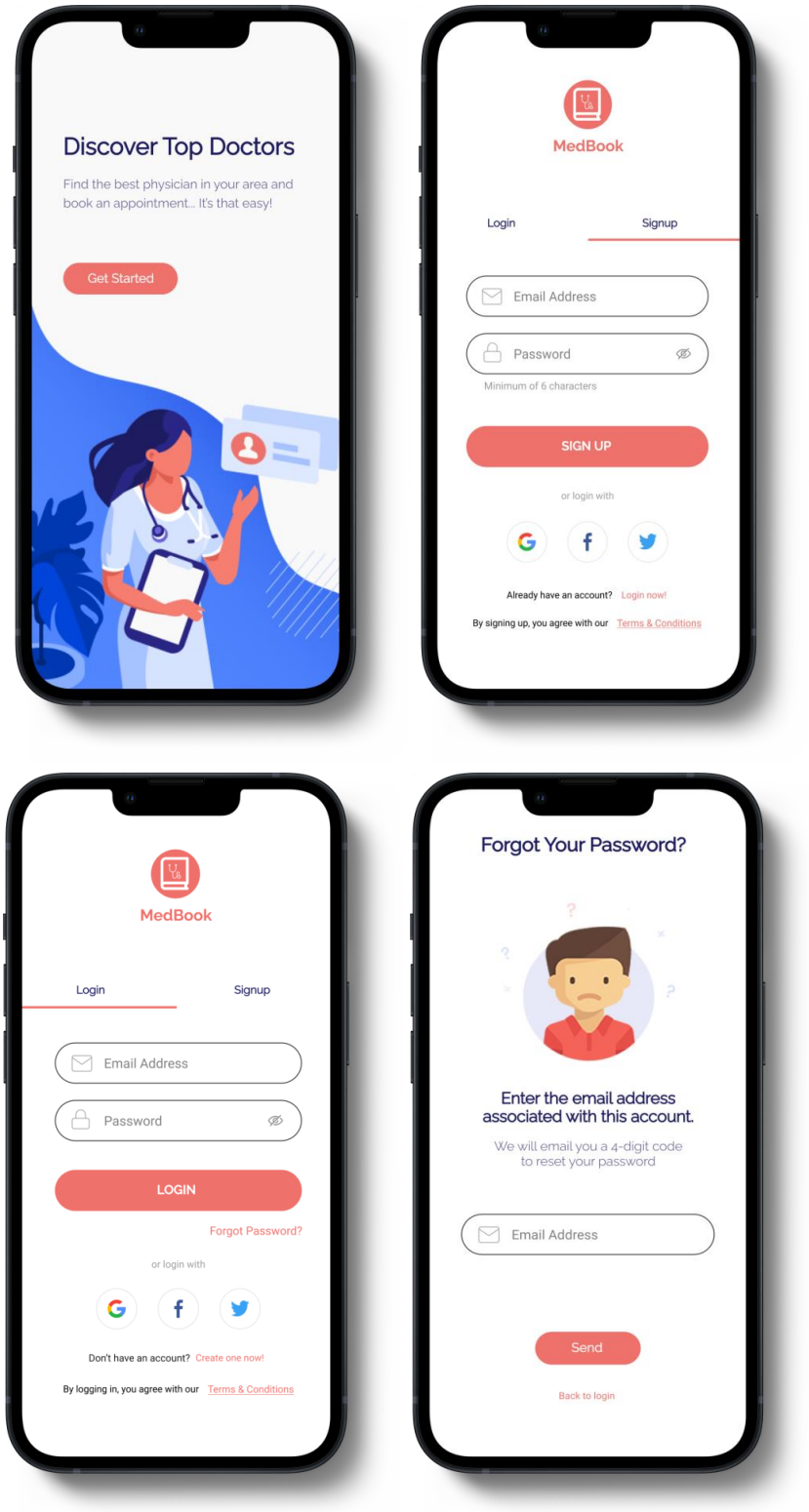
When I press on "No" button

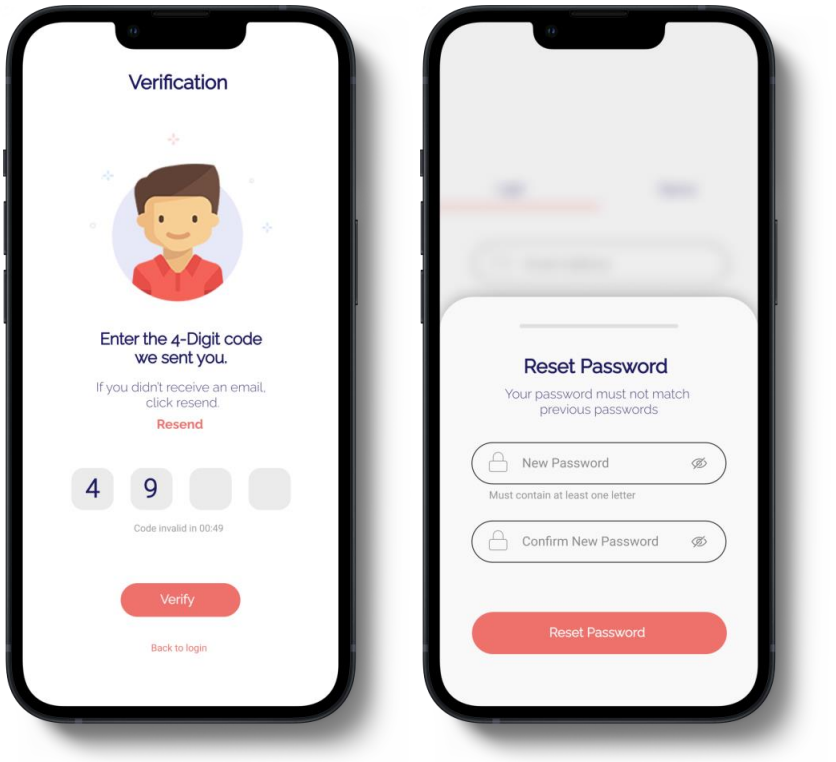
Then I am redirected to the "Cancel Appointment" page

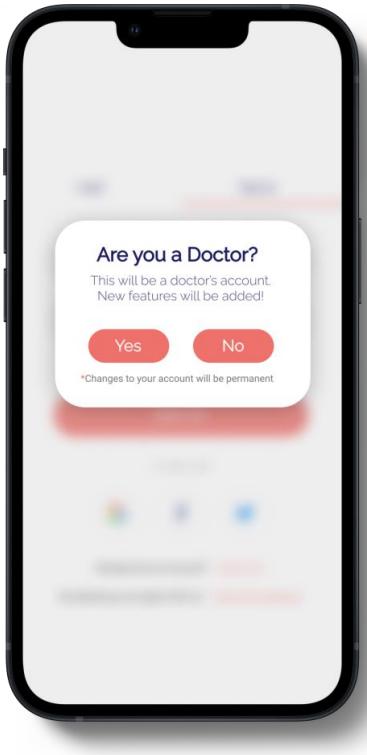
And appointment status has not changed

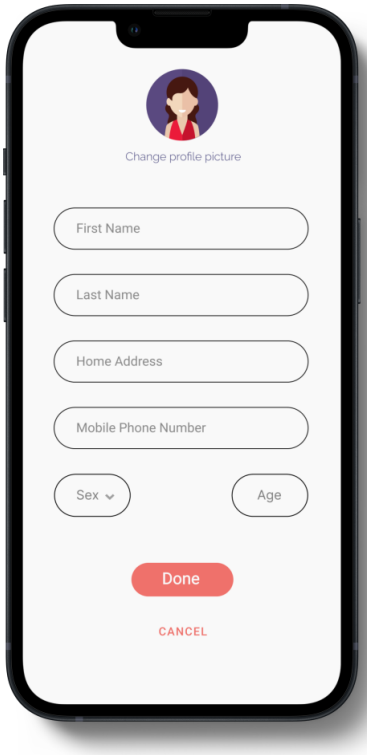


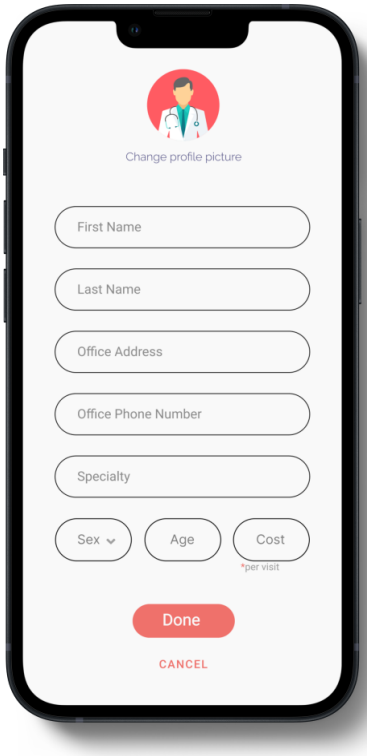
3 User Interface Mockups

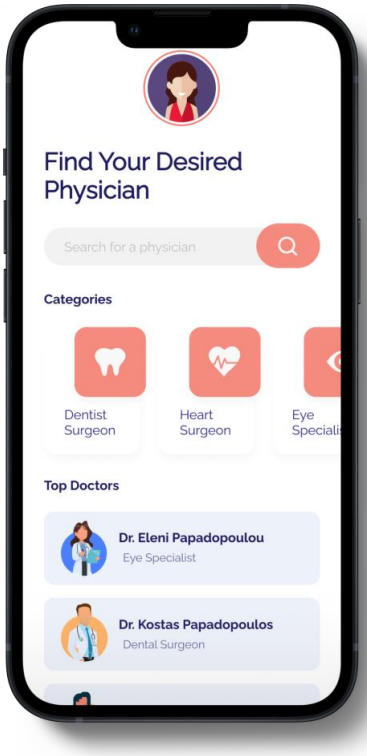


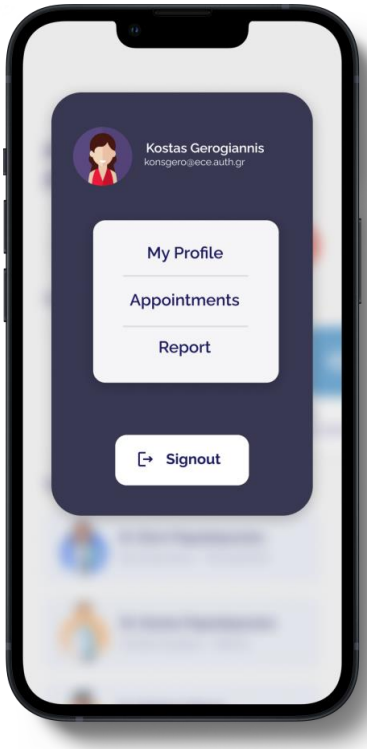

Verification
Enter the 4-Digit code we sent you.
If you didn't receive an email, click resend.
Resend
4
9
Code invalid in 00:49
Verify
Back to login
Reset Password
Your password must not match previous passwords
New Password
Must contain at least one letter
Confirm New Password
Reset Password

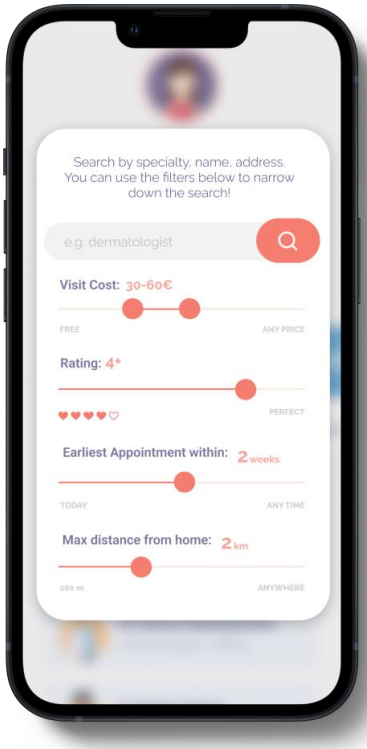

Are you a Doctor?
This will be a doctor's account.
New features will be added!
Yes
No
*Changes to your account will be permanent

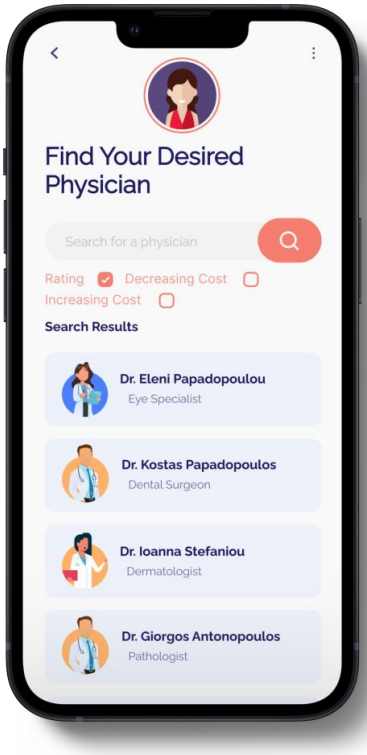

Change profile picture
First Name
Last Name
Home Address
Mobile Phone Number
Sex
Age
Done
CANCEL

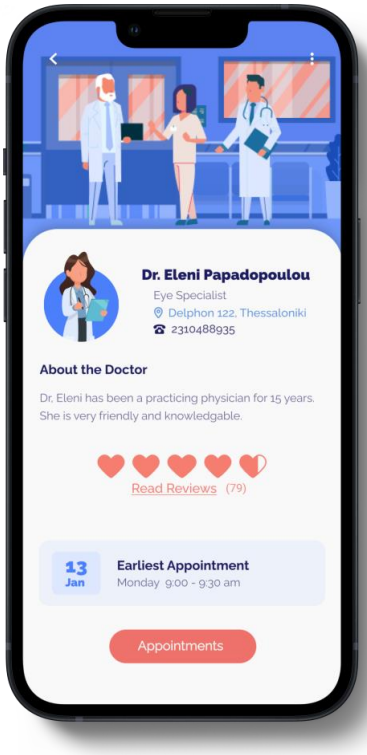

Change profile picture
First Name
Last Name
Office Address
Office Phone Number
Specialty
Sex
Age
Cost
*per visit
Done
CANCEL

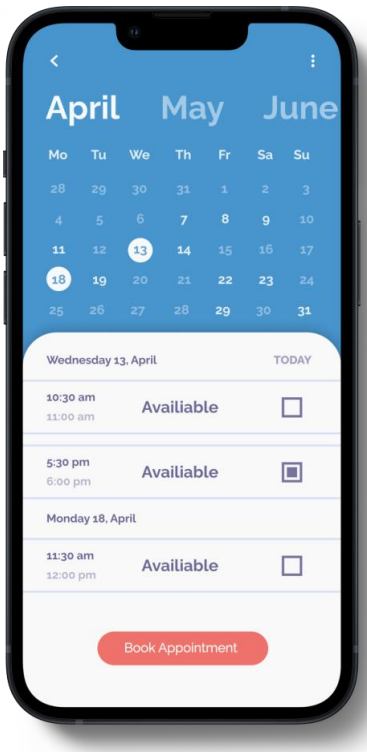

Find Your Desired Physician
Search for a physician
Categories
Dentist Surgeon
Heart Surgeon
Top Doctors
Dr. Eleni Papadopoulou
Eye Specialist
Dr. Kostas Papadopoulos
Dental Surgeon

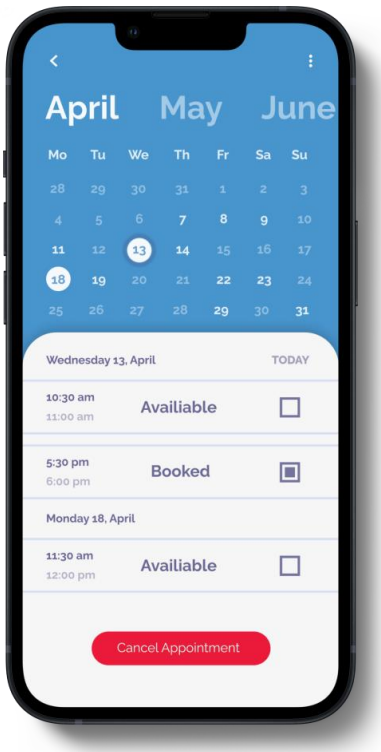

Kostas Gerogiannis
konsgero@ece.auth.gr
My Profile
Appointments
Report
Signout

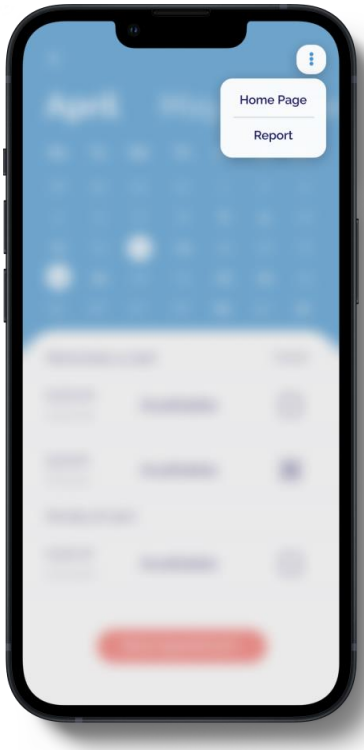

Search by specialty, name, address.
You can use the filters below to narrow down the search!
e.g. dermatologist
Visit Cost: 30-60€
FREE
ANY PRICE
Rating: 4+
PERFECT
Earliest Appointment within: 2 weeks
TODAY
ANY TIME
Max distance from home: 2 km
200 m
ANYWHERE

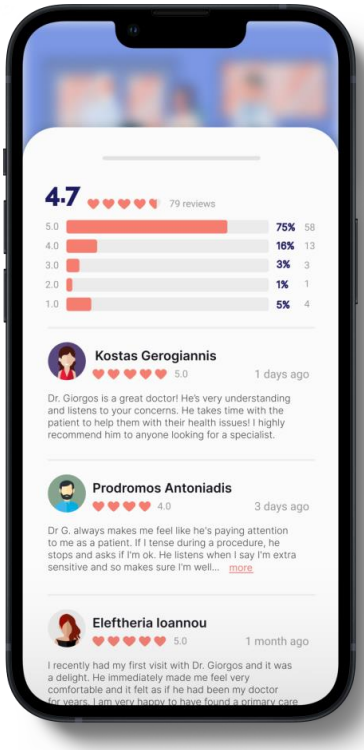

Find Your Desired Physician
Search for a physician
Rating
Decreasing Cost
Increasing Cost
Search Results
Dr. Eleni Papadopoulou
Eye Specialist
Dr. Kostas Papadopoulos
Dental Surgeon
Dr. Ioanna Stefaniou
Dermatologist
Dr. Giorgos Antonopoulos
Pathologist

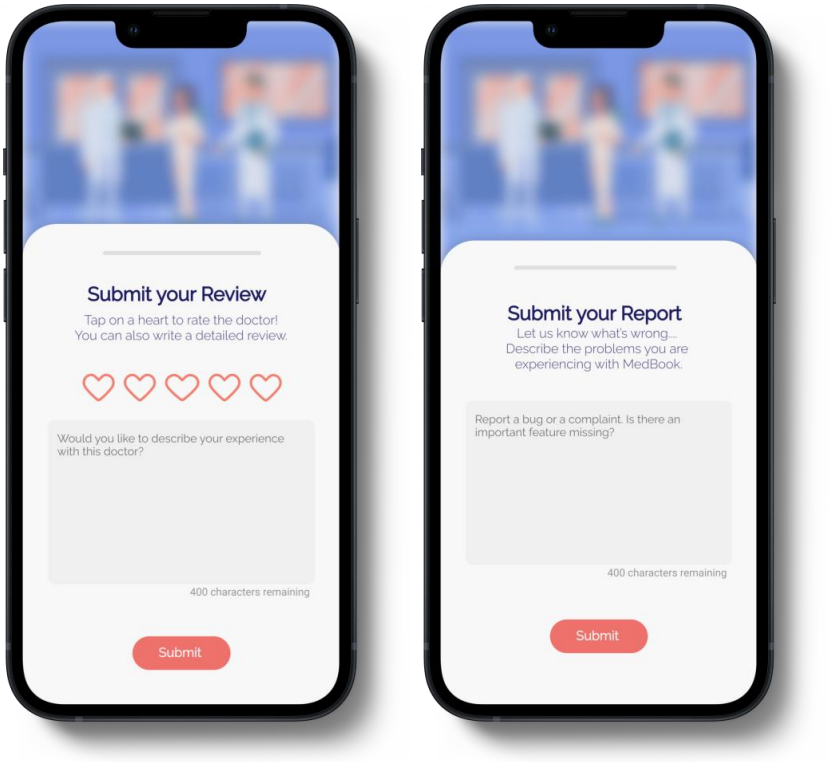

Dr. Eleni Papadopoulou
Eye Specialist
Delphon 122, Thessaloniki
2310488935
About the Doctor
Dr. Eleni has been a practicing physician for 15 years. She is very friendly and knowledgable.
Read Reviews (79)
13 Jan
Earliest Appointment
Monday 9:00 - 9:30 am
Appointments

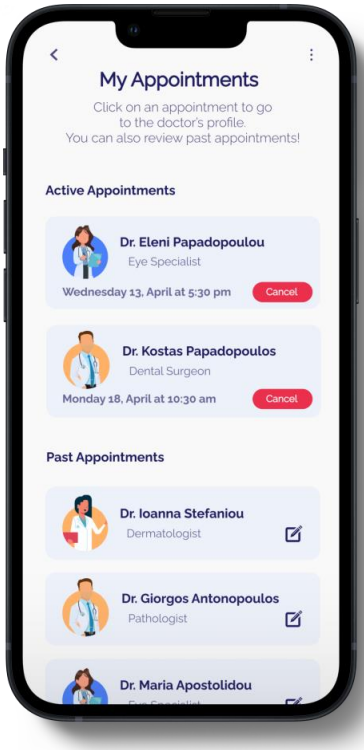

April May June
Wednesday 13, April
TODAY
10:30 am 11:00 am
Availiable
5:30 pm 6:00 pm
Availiable
Monday 18, April
11:30 am 12:00 pm
Availiable
Book Appointment

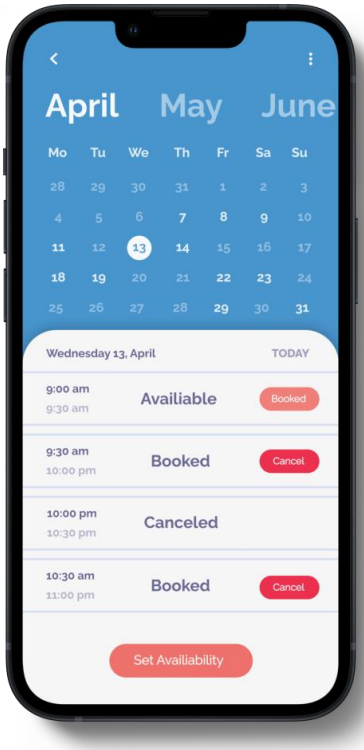

April May June
Wednesday 13, April
TODAY
10:30 am 11:00 am
Availiable
5:30 pm 6:00 pm
Booked
Monday 18, April
11:30 am 12:00 pm
Availiable
Cancel Appointment

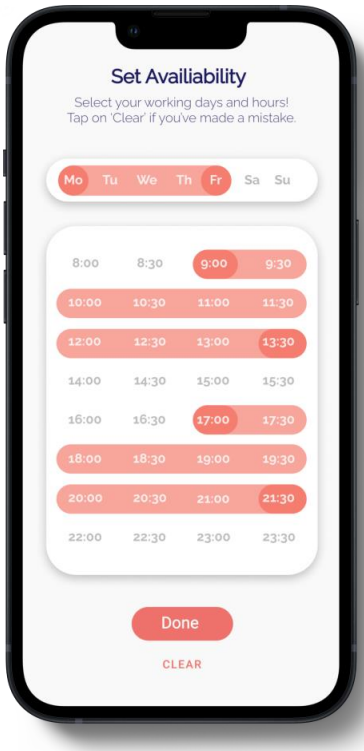

Home Page
Report

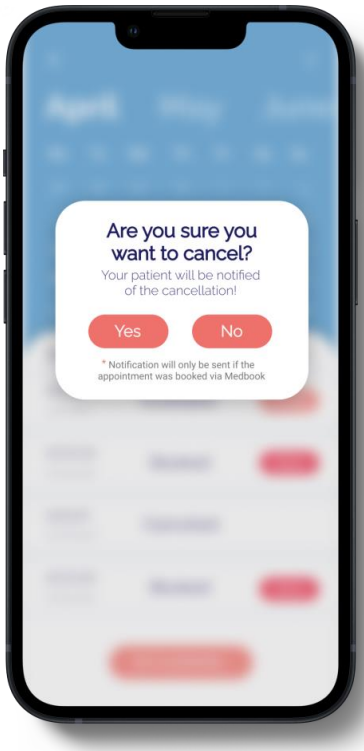

4.7
79 reviews
5.0 75% 58
4.0 16% 13
3.0 3% 3
2.0 1% 1
1.0 5% 4
Kostas Gerogiannis
5.0
1 days ago
Dr. Giorgos is a great doctor! He's very understanding and listens to your concerns. He takes time with the patient to help them with their health issues! I highly recommend him to anyone looking for a specialist.
Prodromos Antoniadis
4.0
3 days ago
Dr G. always makes me feel like he's paying attention to me as a patient. If I tense during a procedure, he stops and asks if I'm ok. He listens when I say I'm extra sensitive and so makes sure I'm well... more
Eleftheria Ioannou
5.0
1 month ago
I recently had my first visit with Dr. Giorgos and it was a delight. He immediately made me feel very comfortable and it felt as if he had been my doctor

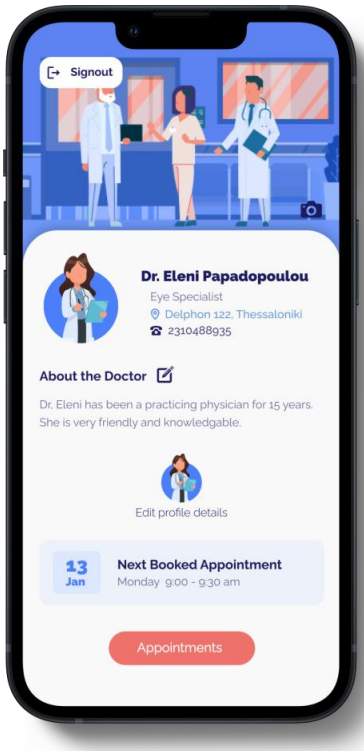

Submit your Review
Tap on a heart to rate the doctor!
You can also write a detailed review.
Would you like to describe your experience with this doctor?
400 characters remaining
Submit
Submit your Report
Let us know what's wrong...
Describe the problems you are experiencing with MedBook.
Report a bug or a complaint. Is there an important feature missing?
400 characters remaining
Submit

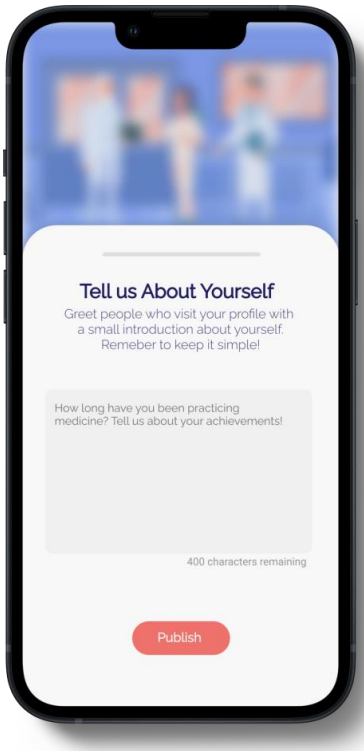

My Appointments
Click on an appointment to go to the doctor's profile.
You can also review past appointments!
Active Appointments
Dr. Eleni Papadopoulou
Eye Specialist
Wednesday 13, April at 5:30 pm
Cancel
Dr. Kostas Papadopoulos
Dental Surgeon
Monday 18, April at 10:30 am
Cancel
Past Appointments
Dr. Ioanna Stefaniou
Dermatologist
Dr. Giorgos Antonopoulos
Pathologist
Dr. Maria Apostolidou


April
May
June
Mo Tu We Th Fr Sa Su
Wednesday 13, April
TODAY
9:00 am
9:30 am
Availiable
Booked
9:30 am
10:00 pm
Booked
Cancel
10:00 pm
10:30 pm
Canceled
10:30 am
11:00 pm
Booked
Cancel
Set Availiability


Set Availiability
Select your working days and hours!
Tap on 'Clear' if you've made a mistake.
Mo Tu We Th Fr Sa Su
8:00 8:30 9:00 9:30
10:00 10:30 11:00 11:30
12:00 12:30 13:00 13:30
14:00 14:30 15:00 15:30
16:00 16:30 17:00 17:30
18:00 18:30 19:00 19:30
20:00 20:30 21:00 21:30
22:00 22:30 23:00 23:30
Done
CLEAR

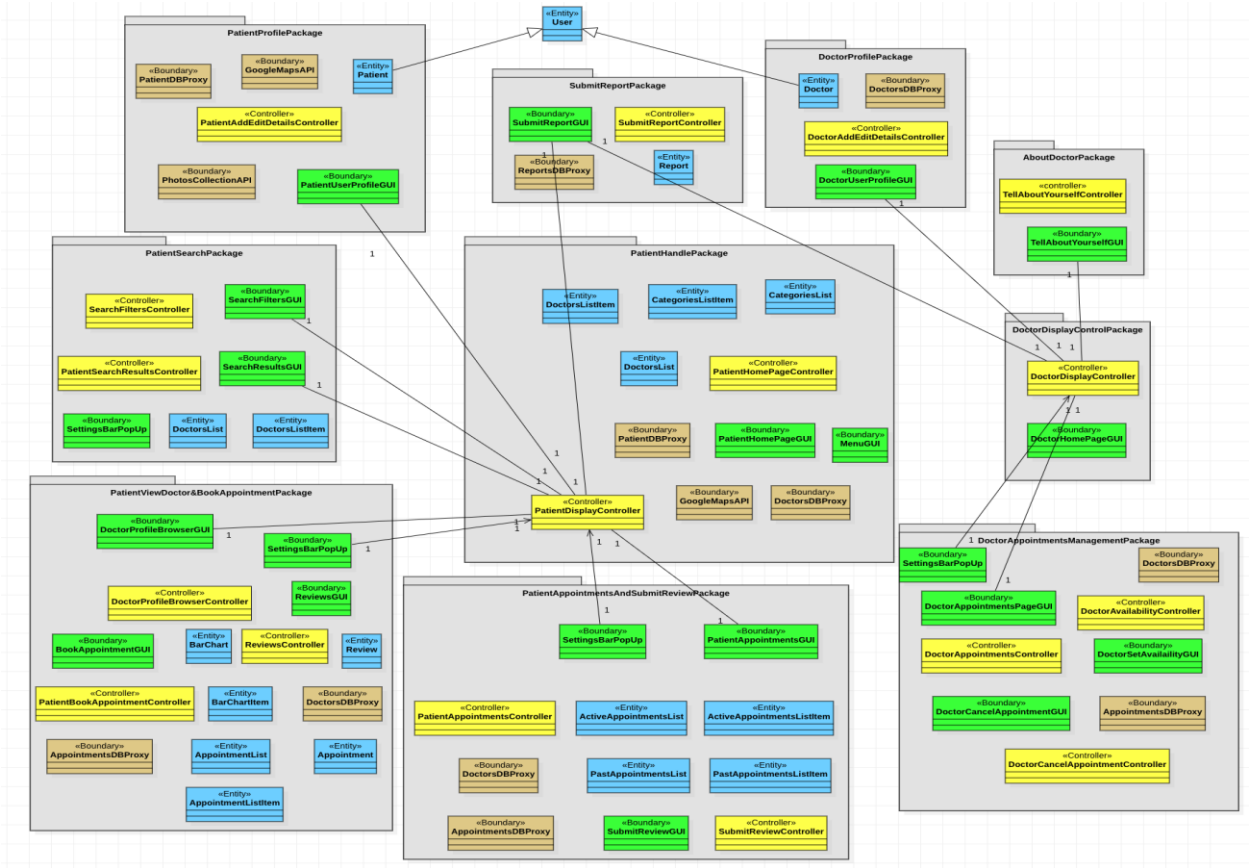

Are you sure you want to cancel?
Your patient will be notified of the cancellation!
Yes
No
* Notification will only be sent if the appointment was booked via Medbook


Signout
Dr. Eleni Papadopoulou
Eye Specialist
Delphon 122, Thessaloniki
2310488935
About the Doctor
Dr. Eleni has been a practicing physician for 15 years.
She is very friendly and knowledgable.
Edit profile details
13
Jan
Next Booked Appointment
Monday 9:00 - 9:30 am
Appointments


Tell us About Yourself
Greet people who visit your profile with
a small introduction about yourself.
Remeber to keep it simple!
How long have you been practicing
medicine? Tell us about your achievements!
400 characters remaining
Publish

4 Static Modeling

Class Diagram



«Entity»
User

-email: String
-password: String
-firstName: String
+lastName: String
-age: int
-sex: String
-profileImage: Image
-phoneNumber: String
-address: String

+User(email: String, password: String, firstName: String, lastName: String, age: int, address: String, phoneNumber: String, sex: String, profileImage: Image): void
+setEmail(email: String): void
+getEmail(): String
+setPassword(password: String): void
+getPassword(): String
+setFirstName(firstName: String): void
+getFirstName(): String
+setLastName(lastName: String): void
+getLastName(): String
+setAge(age: int): void
+getAge(): Int
+setSex(sex: String): void
+getSex(): String
+setProfileImage(profileImage: Image): void
+getProfileImage(): Image
+getPhoneNumber(): String
+setPhoneNumber(value: String): void
+setAddress(value: String): void
+getAddress(): String

Class attributes:

- email: The user's email.

- password: The user's password.

- firstName: The user's first name.

- lastName: The user's last name.

- age: The user's age.

- sex: The user's gender.

- profileImage: The user's profile image.

- phoneNumber: The user's phone number.

- address: The address declared by the user.

Class methods:

- [User(email: String, password: String, firstName: String, lastName: String, age: int, address: String, phoneNumber: String, sex: String, profileImage: Image): void]: The class constructor. All arguments are copied to the corresponding class attributes.

- [setEmail(email: String): void]: Changes the email variable with the value provided as an argument.

- [getEmail(): String]: Returns the email variable.

- [setPassword(password: String): void]: Changes the password variable with the value provided as an argument.

- [getPassword(): String]: Returns the password variable.

- [setFirstName(firstName: String): void]: Changes the firstName variable with the value provided as an argument.

- [getFirstName(): String]: Returns the firstName variable.

- [setLastName(lastName: String): void]: Changes the lastName variable with the value provided as an argument.

- [getLastName(): String]: Returns the lastName variable.

- [setAge(age: int): void]: Changes the age variable with the value provided as an argument.

- [getAge(): int]: Returns the age variable.

- [setSex(sex: String): void]: Changes the sex variable with the value provided as an argument.

- [getSex(): String]: Returns the sex variable.

- [setProfileImage(profileImage: Image): void]: Changes the profileImage with the image provided as an argument.

- [getProfileImage(): Image]: Returns the profileImage variable.

- [setPhoneNumber(phoneNumber: String): void]: Changes the phoneNumber variable with the value provided as an argument.

- [getPhoneNumber(): String]: Returns the phoneNumber variable.

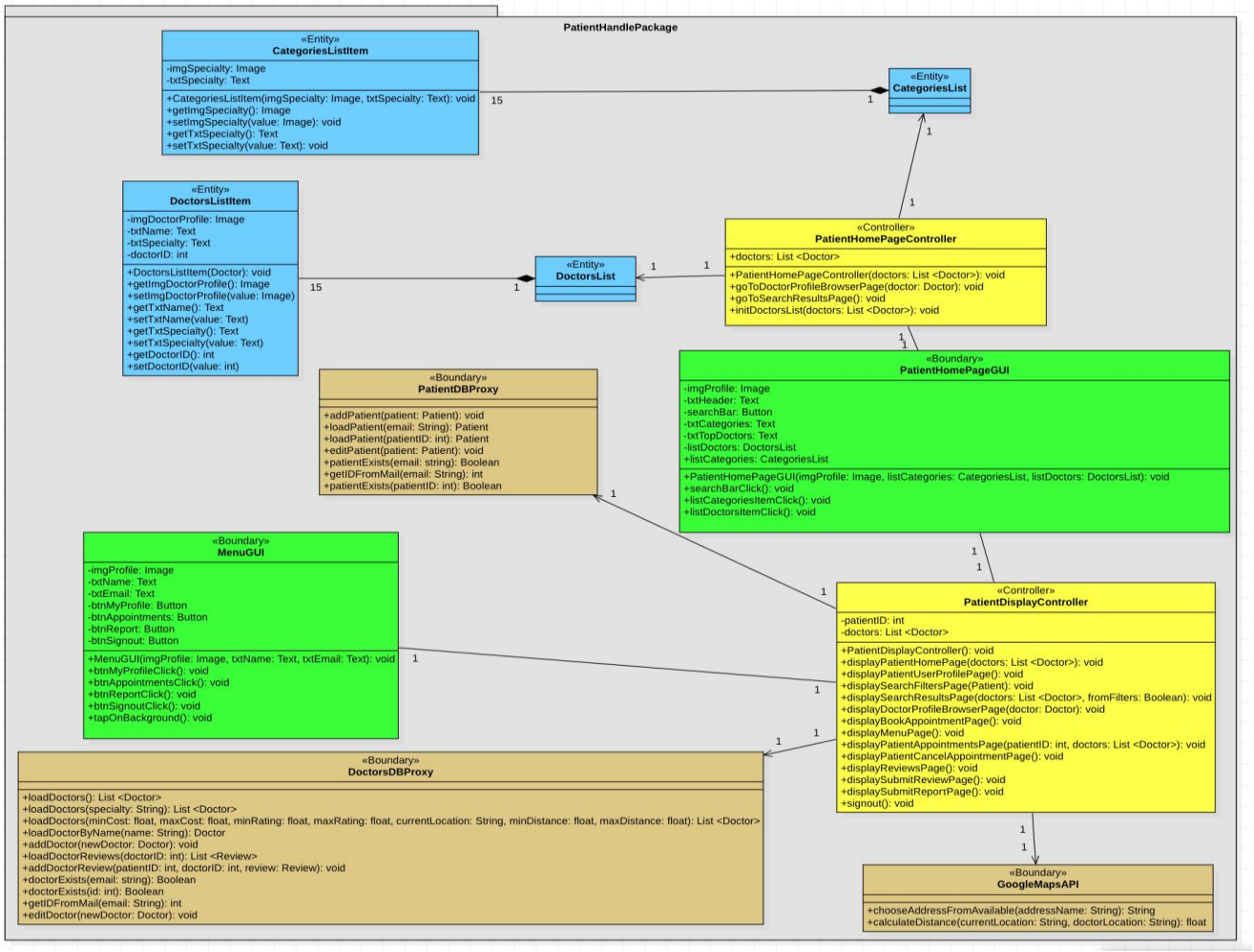
- [setAddress(address: String): void]: Changes the address variable with the value provided as an argument.

- [getAddress(): String]: Returns the address variable.

NOTE: From here onwards, in any class where there are getter and setter functions, they will not be listed to save space.

4.1 <Πακέτα λεξιλογίου σεναρίων υψηλής προτεραιότητας>

4.1.1 PatientHandlePackage



<<Boundary>> MenuGUI

Class attributes:

- imgProfile: The user's profile image.
- txtName: Label displaying the user's name and last name.
- txtEmail: Label displaying the user's email.
- btnMyProfile: Button for viewing the user's profile.
- btnAppointments: Button for viewing the user's appointments.
- btnReport: Button for submitting a report by the user.
- btnSignout: Button for signing out the user from the application.

Class methods:

- [MenuGUI(imgProfile: Image, txtName: Text, txtEmail: Text): void]: The class constructor. Initializes the class attributes based on the arguments.

- [btnMyProfileClick(): void]: Calls the displayPatientUserProfilePage() method of the PatientDisplayController controller when the btnMyProfile button is clicked.
- [btnAppointmentsClick(): void]: Calls the displayPatientAppointmentsPage() method of the PatientDisplayController controller when the btnAppointments button is clicked.
- [btnReportClick(): void]: Calls the displaySubmitReportPage() method of the PatientDisplayController controller when the btnReport button is clicked.
- [btnSignoutClick(): void]: Calls the signout() method of the PatientDisplayController controller when the btnSignout button is clicked.
- [tapOnBackground(): void]: Calls the displayPatientHomePage() method of the PatientDisplayController controller when the background overlay is tapped.

<<Controller>> PatientHomePageController

Class attributes:

- doctors: A list of objects of type Doctor.

Class methods:

- [PatientHomePageController(doctors: List<Doctor>): void]: The class constructor. Calls the initDoctorsList(doctors) method.
- [goToDoctorProfilePage(): void]: Calls the displayDoctorProfileBrowserPage() method of the PatientDisplayController controller when a DoctorsListItem is clicked.
- [goToSearchResultsPage(): void]: Calls the displaySearchResultsPage() method of the PatientDisplayController controller when a CategoriesListItem is clicked.
- [initDoctorsList(doctors: List<Doctor>): void]: Creates the DoctorsList and initializes the DoctorsListItem based on the doctors list.

<<Boundary>> PatientDBProxy

Class methods:

- [addPatient(patient: Patient): void]: Adds a patient to the patient database.
- [loadPatient(email: String): Patient]: Searches, loads, and returns a patient from the database based on their email.
- [loadPatient(patientID: int): Patient]: Searches, loads, and returns a patient from the database based on their ID.
- [editPatient(patient: Patient): void]: Calls the patientExists(email: String) function, and if the patient exists, replaces them with the input patient.
- [patientExists(email: String): Boolean]: Returns True if the patient with the provided email (argument) exists in the patient database.
- [getIDFromMail(email: String): int]: Returns the ID of the patient with the specified email from the patient database.
- [patientExists(patientID:int): Boolean]: Returns True if the patient with the provided ID (argument) exists in the patient database.

<<Boundary>> DoctorsDBProxy

Class methods:

- [loadDoctors(): List<Doctor>]: Returns a list of the best doctors based on their ratings. The list is refreshed in the database every 24 hours.
- [loadDoctors(specialty: String): List<Doctor>]: Returns a list of the best doctors in a specific specialty based on their ratings. The list is refreshed in the database every 24 hours.
- [loadDoctors(minCost: float, maxCost: float, minRating: float, maxRating: float, currentLocation: String, minDistance: float, maxDistance: float): List<Doctor>]: Returns a list of doctors that meet the specified criteria. To calculate the distances of doctors from the current location, the calculateDistance function of the GoogleMapsAPI is called.
- [loadDoctorByName(name: String): Doctor]: Searches, loads, and returns a doctor from the doctor database based on their name.
- [addDoctor(newDoctor: Doctor): void]: Adds a doctor to the doctor database.
- [loadDoctorReviews(doctorID: int): List<Review>]: Searches, loads, and returns a list of reviews for a doctor from the doctor database based on their ID.
- [addDoctorReview(patientID: int, doctorID: int, review: Review): void]: Adds a review for the doctor with the 'doctorID' ID to the doctor database by the patient with the patientID.
- [doctorExists(email: String): Boolean]: Returns True if the doctor with the provided email exists in the doctor database.
- [doctorExists(doctorID: int): Boolean]: Returns True if the doctor with the provided ID exists in the doctor database.
- [getIDFromMail(email: String): int]: Returns the ID of the doctor with the specified email from the doctor database.
- [editDoctor(newDoctor: Doctor): void]: Calls the doctorExists(email: String) function, and if the doctor exists, replaces them with the input doctor.

<<Boundary>> GoogleMapsAPI

Class methods:

- [chooseAddressFromAvailable(addressName: String): String]: Displays a list of possible available addresses based on the addressName argument and returns the user's selection as a String.
- [calculateDistance(currentLocation: String, doctorLocation: String): float]: Takes as input the addresses of the patient and the doctor and calculates the distance between them, returning it as a float.

<<Controller>> PatientDisplayController

Class attributes:

- patientID: The unique identifier for each patient.
- doctors: A list of objects of type Doctor.

Class methods:

- [PatientDisplayController(): void]: The constructor function of the class. It does not have any functionality.
- [displayPatientHomePage(doctors: List <Doctor>): void]: Displays the Patient Home Page, taking the list of recommended doctors as an argument.

- [displayPatientUserProfilePage(): void]: Displays the Patient User Profile Page.
- [displaySearchFiltersPage(): void]: Displays the Search Filters Page.
- [displaySearchResultsPage(doctors: List <Doctor>, fromFilters:Boolean): void]: Displays the Search Results Page with the list of doctors retrieved from the DoctorsDBProxy. If fromFilters is true, it calls the function loadDoctors(minCost:float, maxCost:float, minRating:float, maxRating:float, minDistance:float, maxDistance:float): List <Doctor>), otherwise, it calls the function loadDoctors(specialty: String).
- [displayDoctorProfileBrowserPage(doctor: Doctor): void]: Displays the Doctor Profile Browser Page.
- [displayBookAppointmentPage(): void]: Displays the Book Appointment Page.
- [displayMenuPage(): void]: Displays the Menu Page.
- [displayPatientAppointmentsPage(patientID: int, doctors: List <Doctor>): void]: Displays the Patient Appointments Page and transfers the unique user ID.
- [displayCancelAppointmentPage(): void]: Displays the Patient Cancel Appointment Page.
- [displayReviewsPage(): void]: Displays the Reviews Page.
- [displaySubmitReviewPage(): void]: Displays the Submit Review Page.
- [displaySubmitReportPage(): void]: Displays the Submit Report Page.
- [signout(): void]: Logs the user out of the application.

<<Boundary>> PatientHomePageGUI

Class attributes:
- imgProfile: The user's profile picture.
- txtHeader: A header with the string "Find Your Desired Physician."
- searchBar: The search bar that functions as a button.
- txtCategories: A label with the string "Categories."
- txtTopDoctors: A label with the string "Top Doctors."
- listDoctors: An object of type DoctorsList.
- listCategories: An object of type CategoriesList.

Class methods:
- [PatientHomePageGUI(imgProfile: Image, listCategories: CategoriesList, listDoctors: DoctorsList): void]: The constructor function of the class.
- [SearchBarClick(): void]: Calls the method displaySearchFiltersPage() of the PatientDisplayController when the searchBar is clicked.
- [listCategoriesItemClick(): void]: Calls the displaySearchResultsPage method of the PatientDisplayController class when a click event occurs on an item of the CategoriesListItem entity.
- [listDoctorItemClick(): void]: Calls the displayDoctorProfileBrowserPage method of the PatientDisplayController class when a click event occurs on an item of the DoctorListItem entity.

<<Entity>> CategoriesListItem

Class attributes:
- imgSpecialty: The icon representing the specialty category of doctors.

- txtSpecialty: A label with the title of the specialty category of doctors.

Class methods:

- [CategoriesListItem(imgSpecialty: Image, txtSpecialty: Text): void]: The constructor function of the class.

<<Entity>> CategoriesList

This list consists of 15 objects of the CategoriesListItem class.

<<Entity>> DoctorsListItem

Class attributes:

• imgDoctorProfile: The profile picture of the respective doctor.
• txtName: Label with the name and surname of the doctor.
• txtSpecialty: Label with the doctor's specialty.
• doctorID: The ID of the respective doctor.

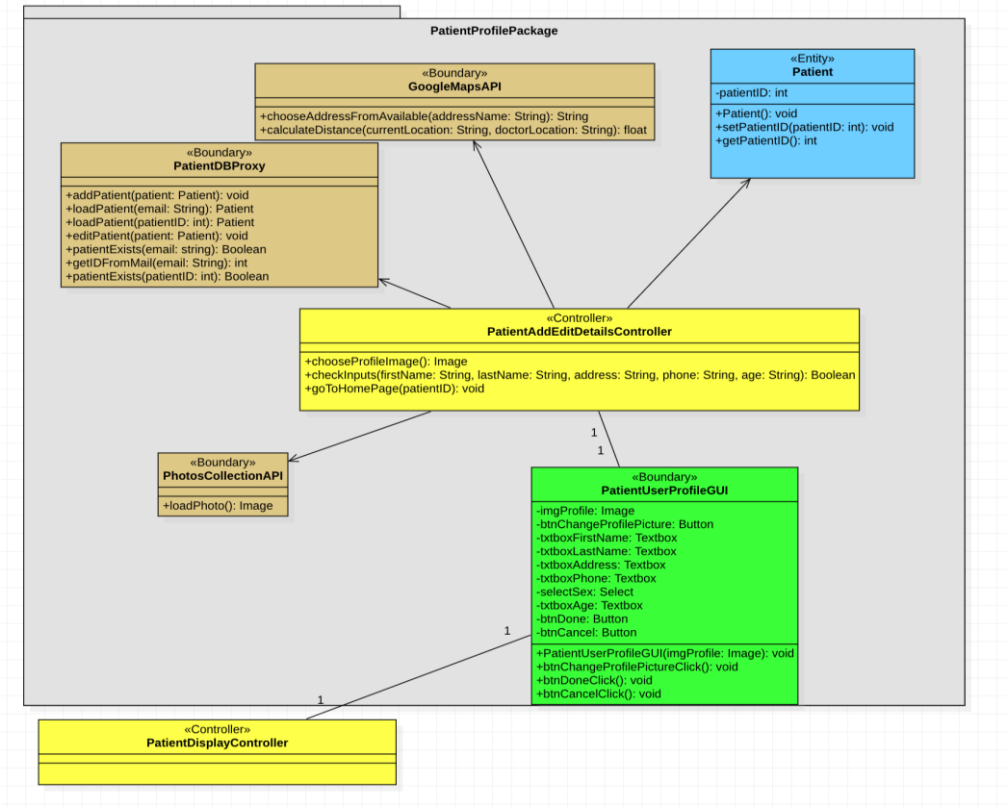
Class methods:

• [DoctorsListItem(Doctor): void]: The constructor of the class.

<<Entity>> DoctorsList

This list consists of an indefinite number of objects of the DoctorsListItem class.

4.1.2 PatientProfilePackage



<<Boundary>> PatientUserProfileGUI

Class Features:

• imgProfile: The user's profile picture.

• btnChangeProfilePicture: A button labeled "change profile picture" to change the user's profile picture.

• txtboxFirstName: A text field for entering the user's first name.

• txtboxLastName: A text field for entering the user's last name.

• txtboxAddress: A text field for entering the user's address.

• txtboxPhone: A text field for entering the user's phone number.

• selectSex: A list of options for the user's gender.

• txtboxAge: A text field for entering the user's age.

• btnDone: A button labeled "Done" to save the entered information.

• btnCancel: A button labeled "Cancel" to exit the page.

Class Methods:

• [PatientUserProfileGUI(imgProfile: Image): void]: The class constructor. The text fields and the imgProfile image are initialized with the previous values of the patient (if they exist).

• [btnChangeProfilePictureClick(): void]: Calls the chooseProfileImage() method of the PatientAddEditDetailsController controller when btnChangeProfilePicture is clicked.

• [btnDoneClick(): void]: Calls the goToHomePage(patientID) method of the PatientAddEditDetailsController controller when btnDone is clicked.

• [btnCancelClick(): void]: Calls the goToHomePage(patientID) method of the PatientAddEditDetailsController controller when btnCancel is clicked if there is already a user registration; otherwise, the button is inactive.

<<Controller>> PatientAddEditDetailsController

Class Methods:

• [chooseProfileImage(): Image]: Calls the loadPhoto() function of the external system PhotoCollectionGUI and returns the selected image.

• [checkInputs(firstName: String, lastName: String, address: String, phone: String, age: String): Boolean]: Checks if the arguments follow certain rules defined by the system.

• [goToHomePage(patientID): void]: Calls the checkInputs function and validates the information provided by the user. If everything is okay, the addPatient function is called, and the HomePage is displayed. Otherwise, the user remains on the current page.

<<Entity>> Patient

The class inherits from the <<Entity>> User class that we have already defined.

Class Attributes:

• patientID: The unique number of the patient.

Class Methods:

• Patient(): The class constructor. It calls the constructor of the User class and also assigns a random value to the patientID variable.

<<Boundary>> PhotosCollectionAPI

Class Methods:

• [loadPhoto(): Image]: Invokes an external window of the device's operating system that prompts the user to select the desired image, which it then returns.

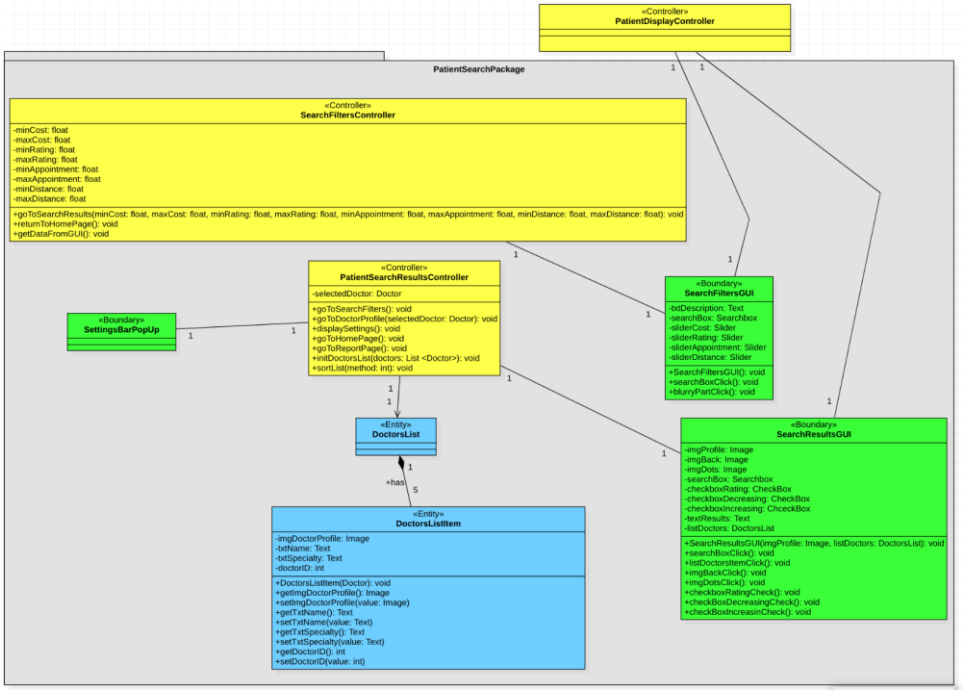
<<Boundary>> DoctorsDBProxy

This classs has already been defined on chapter 4.1.1.

<<Boundary>> GoogleMapsAPI

This classs has already been defined on chapter 4.1.1.

4.1.3 PatientSearchPackage



<<Boundary>> SearchFiltersGUI

Class Features:

• txtDescription: Label with the string "Search by specialty, name, address. You can use the filters below to narrow down the search."

• searchBox: Search box where the user can enter the specialty of the doctors they want to search for.

• sliderCost: Horizontal slider with the title ["Visit Cost: " + (string indicating the price range of the visit)], "FREE" label at the bottom left, and "ANY PRICE" label at the bottom right, where the patient can set the price range for the doctors they want to search for.

• sliderRating: Horizontal slider with the title ["Rating" + (string indicating the minimum rating)] and "PERFECT" label at the bottom right, where the patient can set the minimum rating for the doctors they want to search for.

• sliderAppointment: Horizontal slider with the title ["Earliest Appointment within:" + (string indicating the maximum time duration for the nearest appointment)], "TODAY" label at the bottom left, and "ANY TIME" label at the bottom right, where the user can set the maximum time duration for the nearest appointment available for the doctors they want to search for.

• sliderDistance: Horizontal slider with the title ["Max distance from home: " + (string indicating the maximum distance)], "200m" label at the bottom left, and "ANYWHERE" label at the bottom right, where the user can set the maximum distance (from their home) for the clinics (and by extension, the doctors) they want to search for.

Class Methods:

• [SearchFiltersGUI(): void]: The class constructor function.

• [searchBoxClick(): void]: Calls the goToSearchResults method of the SearchFiltersController when the searchBox button is clicked.

• [blurryPartClick(): void]: Calls the returntoHomePage() method of the SearchFiltersController when a click is made on the blurred background part of the screen.

<<Controller>> SearchFiltersController

Class Features:

• minCost: The minimum visit cost.

• maxCost: The maximum visit cost.

• minRating: The minimum rating.

• maxRating: The maximum rating.

• minAppointment: The earliest appointment time.

• maxAppointment: The latest appointment time.

• minDistance: The minimum distance from the patient's home.

• maxDistance: The maximum distance from the patient's home.

Class Methods:

• [goToSearchResults(minCost: float, maxCost: float, minRating: float, maxRating: float, minAppointment: float, maxAppointment: float, minDistance: float, maxDistance: float): void]: Redirects the user to the SearchResults page based on the filters set by the patient, through the central PatientDisplayController.

• [returnToHomePage(): void]: Redirects the patient to the Home Page.

• [getDataFromGUI(): void]: Returns the data obtained from the search options made by the patient on the Search Filters screen.

<<Boundary>> SearchResultsGUI

Class Features:

• imgProfile: The patient's profile picture.

• imgBack: The "back arrow" icon.

• imgDots: The icon with three dots.

• searchBox: A search box where the user can enter the specialty of the doctors they want to search for.

• checkboxRating: A checkbox that the user can select to sort doctors based on their ratings.

• checkboxDecreasing: A checkbox that the user can select to sort doctors in descending order based on visit cost.

• checkboxIncreasing: A checkbox that the user can select to sort doctors in ascending order based on visit cost.

• textResults: A label containing the string "Search Results".

• listDoctors: Panels containing information about the doctors that result from the search.

Class Methods:

• [searchResultsGUI(imgProfile: Image, listDoctors: DoctorsList): void]: The class constructor function.

• [searchBoxClick(): void]: Calls the goToSearchFilters method of the <<Controller>>PatientSearchResultsController class.

• [listDoctorsItemClick(): void]: Calls the goToDoctorProfile method of the <<Controller>>PatientSearchResultsController class when a doctor's panel is clicked.

• [imgBackClick(): void]: Calls the goToHomePage method of the <<Controller>>PatientSearchResultsController class when the imgBack icon is clicked.

• [imgDotsClick(): void]: Calls the displaySettings method of the <<Controller>>PatientSearchResultsController class when the imgDots icon is clicked.

• [checkboxCheck(): void]: When the state of a checkbox changes, the sortList method of the PatientSearchResultsController class is called.

<<Controller>> PatientSearchResultsController

Class Features:

• selectedDoctor: The doctor presented in the <<Boundary>>SearchResultsGUI panel that has been clicked.

Class Methods:

• [PatientSearchResultsController(patient: Patient): void]: The class constructor function. It uses characteristics of the patient in the PatientSearchResultsGUI class.

• [goToSearchFilters(): void]: Redirects the patient to the Search Filters screen.

• [goToDoctorProfile(selectedDoctor: Doctor): void]: Redirects the patient to the screen displaying the profile of the selected doctor (the selected doctor is passed as an argument).

• [displaySettings(): void]: Redirects the patient to the SettingsBar screen.

• [goToHomePage(): void]: Redirects the patient to the Home Page.

• [goToReportPage(): void]: Redirects the patient to the Report Page.

• [initDoctorsList(doctors: List<Doctor>): void]: Initializes the DoctorsList object with data from the doctors argument.

• [sortList(method: int): void]: Refreshes the list, applying the selected sorting method.

<<Entity>> DoctorListItem

Class Features:

• ImgDoctorProfile: The doctor's profile picture.

• txtName: Label displaying the doctor's name.

• txtSpecialty: Label displaying the doctor's specialty.

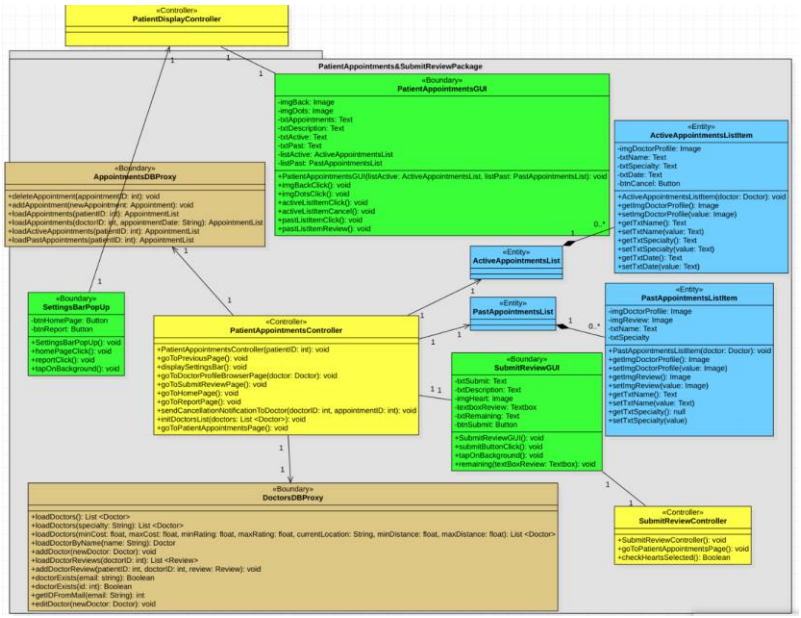
Class Methods:

• [DoctorsListItem(Doctor doctor): void]: Serves as the constructor function of the class.

<<Entity>> DoctorList

Κλάση που αποτελείται από οσαδήποτε αντικείμενα της κλάσης <<Entity>>DoctorListItem.

4.1.4 PatientAppointmentsAndSubmitReviewPackage



<<Boundary>> PatientAppointmentsGUI

Class attributes:

• imgBack: The "back arrow" icon.

• imgDots: The icon with three dots.

• txtAppointments: A label with the string "My Appointments".

• txtDescription: A label with the string "Click on an appointment to go to the doctor's profile. You can also review past appointments".

• txtActive: A label with the string "Active Appointments".

• txtPast: A label with the string "Past Appointments".

• listActive: A list of the patient's closed appointments that have not yet been completed.

• listPast: A list of the patient's closed appointments that have been completed.

Class methods:

• [PatientAppointmentsGUI(listActive: ActiveAppointmentsList, listPast: PastAppointmentsList): void]: The class constructor function.

• [imgBackClick():void]: Calls the goToPreviousPage function of the <<Controller>>PatientAppointmentsController class when the "back arrow" is clicked.

• [imgDotsClick(): void]: Calls the displaySettingsBar function of the <<Controller>>PatientAppointmentsController class when the three dots are clicked.

• [activeListIemClick(): void]: Calls the goToDoctorProfileBrowserPage function of the <<Controller>>PatientAppointmentsController class when a doctor's box in the listActive is clicked.

• [activeListIemCancel(): void]: Calls the sendCancellationNotificationToDoctor function of the <<Controller>>PatientAppointmentsController class when the "Cancel" button of a doctor's box in the listActive is clicked.

• [pastListIemClick(): void]: Calls the goToDoctorProfileBrowserPage function of the <<Controller>>PatientAppointmentsController class when a doctor's box in the listPast is clicked.

• [pastListItemReview(): void]: Calls the goToSubmitReviewPage function of the <<Controller>>PatientAppointmentsController class when the notebook icon in a doctor's box in the listPast is clicked.

<<Boundary>> AppointmentsDBProxy

Class methods:

• [deleteAppointment(appointmentID: int): void]: Deletes an appointment from the appointment database.

• [addAppointment(newAppointment: Appointment): void]: Adds a new appointment to the appointment database.

• [loadAppointments(patientID: int): Patient]: Searches, loads, and returns the list of all appointments in which the patient with the specific ID is scheduled.

• [loadAppointments(doctorID: int, appointmentDate: String): Patient]: Searches, loads, and returns the list of all appointments in which the doctor with the specific ID is scheduled for a particular date.

• [loadActiveAppointments(patientID: int): AppointmentList]: Searches, loads, and returns the list of all appointments in which the patient with the specific ID is scheduled and that have not been completed.

• [loadPastAppointments(patientID: int): AppointmentList]: Searches, loads, and returns the list of all appointments in which the patient with the specific ID is scheduled and that have been completed.

<<Controller>> PatientAppointmentsController

Class attributes:

• doctors: A list of objects of type Doctor.

Class methods:

• [PatientAppointmentsController(patientID: int): void]: The class constructor function.

• [goToPreviousPage(): void]: Redirects the user to the immediately previous screen.

• [displaySettingsBar(): void]: Redirects the user to the Settings Bar screen.

• [goToDoctorProfileBrowserPage(doctor: Doctor): void]: Redirects the user to the screen displaying the profile of the doctor passed as an argument.

• [goToSubmitReviewPage(Doctor doctor): void]: Redirects the user to the Submit Review screen to submit a review for the doctor passed as an argument.

• [goToHomePage(): void]: Redirects the user to the Home Page.

• [goToReportPage(): void]: Redirects the user to the Submit Report screen.

• [sendCancellationNotificationToDoctor(doctorID: int, appointmentID: int): void]: Sends a notification to the doctor with this ID and calls the deleteAppointment function of the Boundary AppointmentsDBProxy.

• [initDoctorsList(doctors: List <Doctor>): void]: Creates the DoctorsList and initializes the DoctorsListItem based on the doctors list.

<<Entity>> ActiveAppointmentsListItem

Class attributes:

• imgDoctorProfile: The profile picture of the respective doctor.

• txtName: Label containing the name of the respective doctor.

• txtSpeciality: Label containing the specialty of the respective doctor.

• txtDate: Label containing the (future) date when the patient has scheduled an appointment with this specific doctor.

• btnCancel: Cancellation button.

Class methods:

• [ActivateAppointmentsListItem(Doctor doctor): void]: Serves as the constructor function.

<<Entity>> ActiveAppointmentsList

It consists of objects of the class "Entity" ActiveAppointmentsListItem.

<<Entity>> PastAppointmentsListItem

Class attributes:

• imgDoctorProfile: The profile picture of the respective doctor.

• ImgReview: Note icon.

• txtName: Label containing the name of the respective doctor.

• txtSpeciality: Label containing the specialty of the respective doctor.

Class methods:

• [PastAppointmentsListItem(Doctor doctor): void]: Represents the constructor function of the class.

<<Entity>> PastAppointmentsList

Consists of objects of the class <<Entity>> AppointmentsListItem.

<<Boundary>> SubmitReviewGUI

Class attributes:

• txtSubmit: Title containing the string "Submit your Review."

• txtDescription: Label containing the string "Tap on a heart to rate the doctor! You can also write a detailed review."

• imgHeart: 5 identical heart icons.

• TextboxReview: Text box where the patient can enter their review for the doctor. Initially contains the string "Would you like to describe your experience with this doctor?" and can be filled with up to 400 characters.

• txtRemaining: Label containing the string "400 characters remaining."

• btnSubmit: Review submission button.

Class methods:

• [SubmitReviewGUI(): void]: The constructor function of the class.

• [submitButtonClick(): void]: Calls the addDoctorReview function of the <<Boundary>> DoctorsDBProxy class to add the review to the doctor's database when the btnSubmit button is clicked (handled by the sendCancellationNotificationToDoctor function of the controller).

• [tapOnBackground(): void]: Redirects the user to the PatientAppointments page.

• [remaining(textboxReview: Textbox): void]: Calculates and returns how many characters are remaining up to the limit of 400 characters in the review.

<<Boundary>> DoctorsDBProxy

The class has already been defined in section 4.1.1.

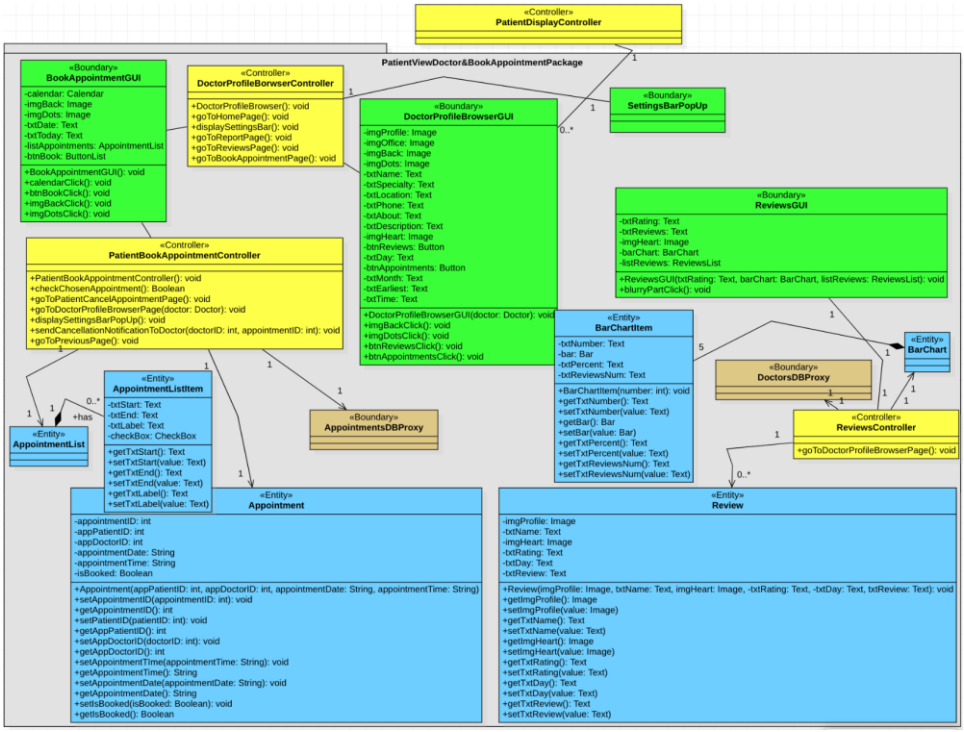
<<Boundary>> SettingsBarPopUp

Class attributes:

• btnHomePage: Button for navigating to the Home Page.
• btnReport: Button for navigating to the Report Page.

Class methods:
• [SettingsBarPopUp(): void]: The constructor function of the class.
• [homePageClick(): void]: Calls the function of the corresponding controller to redirect the user to the previous page they were on when the btnHomePage button is clicked.
• [reportClick(): void]: Calls the function of the corresponding controller to redirect the user to the previous page they were on when the btnReport button is clicked.
• [tapOnBackground(): void]: Calls the function of the corresponding controller to redirect the user to the previous page they were on when clicking on the background.

4.1.5 PatientViewDoctor&BookAppointmentPackage



<<Boundary>> SettingsBarPopUp

The class has already been defined in section 4.1.4.

<<Boundary>> AppointmentsDBProxy

The class has already been defined in section 4.1.4.

<<Boundary>> DoctorsDBProxy

The class has already been defined in section 4.1.1.

<<Entity>> Review

Class attributes:

• imgProfile: The user's profile picture.
• txtName: Label displaying the name of the patient who submitted the review.
• imgHeart: 5 identical heart icons representing the rating.
• txtRating: Label containing the rating in decimal format with one decimal place.
• txtDay: Label containing the string indicating the time elapsed since the review was submitted.
• txtReview: Text field containing the review.

Class methods:

• [Review(imgProfile: Image, txtName: Text, imgHeart: Image, txtRating: Text, txtDay: Text, txtReview: Text): void]: The constructor function of the class.

<<Entity>> Appointment

Class attributes:

• appointmentID: The unique identifier for each appointment.
• appPatientID: The patientID of the patient associated with the appointment.
• appDoctorID: The patientID of the doctor associated with the appointment.
• appointmentDate: A string containing the date of the scheduled appointment.
• appointmentTime: A string containing the time of the scheduled appointment.
• isBooked: A boolean variable that takes the value True if the appointment is booked.

Class methods:

• [Appointment(appPatientID: int, appDoctorID: int, appointmentDate: String, appointmentTime: String): void]: The constructor function of the class.

<<Entity>> BarChartItem

Class attributes:

• txtNumber: Label containing the rating (1.0 or 2.0 or 3.0 or 4.0 or 5.0) associated with the specific object.
• bar: Bar icon. The distribution of the bar's colors is determined based on the txtPercent percentage.
• txtPercent: Label containing the percentage of reviews that are for this specific object compared to the rest of the objects in the <<Entity>> BarChart class.
• txtReviewsNum: Label containing the number of different reviews submitted for a doctor with a rating equal to txtNumber.

<<Entity>> BarChart

Consists of 5 objects of the class <<Entity>> BarChartItem, where the first object has txtNumber = 1.0, the second has txtNumber = 2.0, and so on.

<<Controller>> ReviewsController

Class methods:

• [goToDoctorProfileBrowser(): void]: Redirects the patient to the doctor's profile.

<<Boundary>> ReviewsGUI

Class attributes:

• txtRating: Title containing the average rating of reviews for the respective doctor.

• txtReviews: Label containing the number of reviews submitted for the doctor associated with this object.
• imgHeart: 5 identical heart icons representing the average rating with color distribution.
• barChart: 1 object of the class <<Entity>> BarChart.
• listReviews: List containing all Review objects related to the respective doctor.

Class methods:
• [ReviewsGUI(txtRating: Text, barChart: BarChart, listReviews: ReviewsList): void]: The constructor function of the class.
• [blurryPartClick(): void]: Calls the goToDoctorProfileBrowser function of the <<Controller>> ReviewsController class when clicking on the blurred upper part of the screen.

<<Controller>> DoctorProfileController

Class methods:
• [DoctorProfileBrowser(): void]: Redirects the user to the screen with the profile of the respective doctor.
• [goToHomePage(): void]: Redirects the user to the Home Page.
• [displaySettingsBar(): void]: Redirects the user to the Settings Bar screen.
• [goToReportPage(): void]: Redirects the user to the Submit Report screen.
• [goToReviewsPage(): void]: Redirects the user to the Reviews screen.
• [goToBookAppointmentPage(): void]: Redirects the user to the Book Appointment screen.
• [goToPreviousPage(): void]: Redirects the user to the immediately previous screen.

<<Boundary>> DoctorProfileBrowserGUI

Class attributes:
• imgProfile: The profile picture of the respective doctor.
• imgOffice: The picture of the doctor's office.
• imgBack: The "back arrow" icon.
• imgDots: The icon with three dots.
• txtName: Title containing the name of the respective doctor.
• txtSpeciality: Label containing the string with the doctor's specialty.
• txtLocation: Label containing the address of the doctor's office.
• txtPhone: Label containing the doctor's contact phone number.
• txtDescription: Label containing the short biography of the respective doctor.
• imgHeart: 5 identical heart icons representing the average rating with color distribution.
• btnReviews: Read Reviews button.
• txtDay: Label containing the day (e.g., Monday, Thursday) where the earliest available appointment for the respective doctor exists.
• btnAppointments: Appointments button.
• txtMonth: Label containing the month and day (e.g., 13Jan, 20Feb) where the earliest available appointment for the respective doctor exists.
• txtEarliest: Title containing the string "Earliest Appointment."
• txtTime: Label containing the day and time (e.g., Monday 9:00-9:30 am, Thursday 10:00-10:30 am) where the earliest available appointment for the respective doctor exists.

Class methods:

• [DoctorProfileBrowserGUI(doctor: Doctor): void]: The constructor function of the class.

• [imgBackClick(): void]: Calls the goToPreviousPage function of the <<Controller>> DoctorProfileBrowser class when the "back arrow" is clicked.

• [imgDotsClick(): void]: Calls the displaySettingsBar function of the <<Controller>> DoctorProfileController class when the three dots are clicked.

• [btnReviewsClick(): void]: Calls the goToReviewsPage function of the <<Controller>> DoctorProfileController class when the btnReviews button is clicked.

• [btnAppointmentsClick(): void]: Calls the goToBookAppointmentPage function of the <<Controller>> DoctorProfileController class when the btnAppointments button is clicked.

<<Boundary>> BookAppointmentGUI

Class attributes:

• calendar: An object of type "calendar" where the user can select days and months of their choice by pressing buttons.

• imgBack: The "back arrow" icon.

• imgDots: The icon with three dots.

• TxtDate: Labels containing the strings with the dates selected from the "calendar."

• TxtToday: Labels containing the strings with the available hours for the selected dates from the "calendar."

• ListAppointments: A list of "appointment" objects.

• BtnBook: List of book buttons.

Class methods:

• [BookAppointmentsGUI(): void]: The constructor function of the class.

• CalendarClick(): Changes the color combinations of a calendar button when clicked (bold color when selected and soft color when deselected).

• [btnBookClick(): void]: Calls the checkChosenAppointment function of the <<Controller>> PatientBookAppointmentController class when a Book button is clicked.

• [imgBackClick(): void]: Calls the goToDoctorProfileBrowserPage function of the <<Controller>> PatientBookAppointmentController class when the "back arrow" is clicked.

• [imgDotsClick(): void]: Calls the displaySettingsBar function of the <<Controller>> DoctorProfileController class when the three dots are clicked.

<<Controller>> PatientBookAppointmentController

Class methods:

• [PatientBookAppointmentController(): void]: The constructor function of the class.

• [checkChosenAppointment(): Boolean]: Checks the type of the selected appointment and determines if it is suitable.

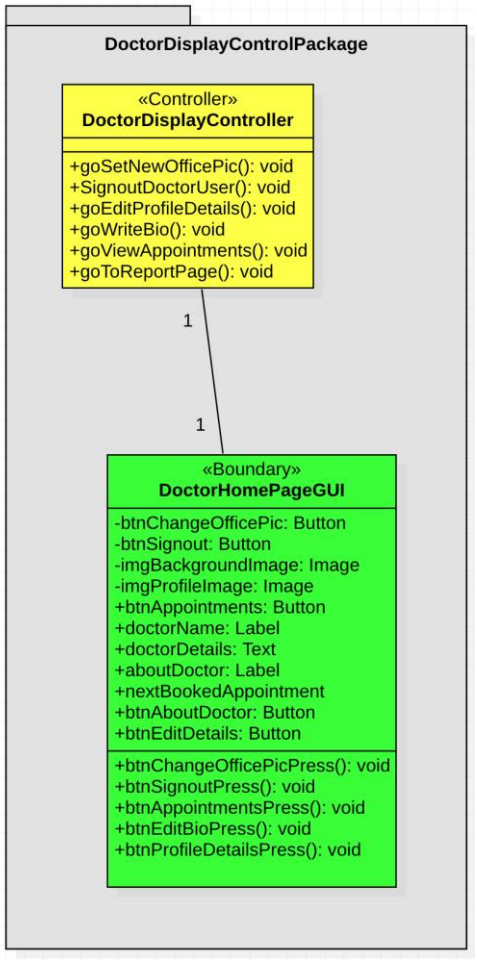
• [goToPatientCancelAppointmentPage(): void]: Redirects the user to the CancelAppointmentPage when they cancel an appointment.

• [goToDoctorProfileBrowserPage(): void]: Redirects the patient to the Doctor Profile Browser screen of the respective doctor.

• [displaySettingsBarPopUp(): void]: Redirects the patient to the Settings Bar screen.

• [sendCancellationNotificationToDoctor(in doctorID: int, in appointmentID: int): void]: Sends a notification to the patient and the doctor when an appointment is canceled through the system.

4.1.6 DoctorDisplayControlPackage



<<Controller>>DoctorDisplayController

Class methods:

• [goSetNewOfficePic(): void]: Changes the doctor's profile picture.
• [SignoutDoctorUser(): void]: Logs out the doctor from the application.
• [goEditProfileDetails(): void]: Redirects the doctor to the Doctor User Profile screen.
• [goWriteBio(): void]: Redirects the doctor to the Tell About Yourself screen.
• [goViewAppointments(): void]: Redirects the doctor to the Doctor Appointments Page.
• [goToReportPage(): void]: Redirects the doctor to the Submit Report screen.

<<Boundary>>DoctorHomePageGUI

Class attributes:

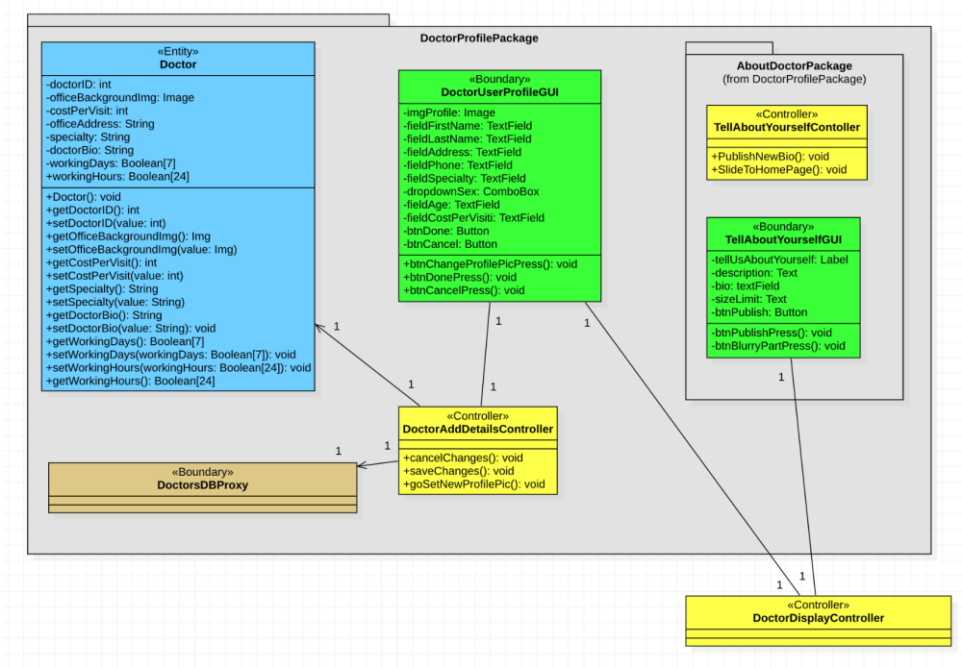
• btnChangeOfficePic: Camera icon button.
• btnSignout: Signout button.
• imgBackgroundImage: Background office image.
• imgProfileImage: Doctor's profile picture.

• btnAppointments: Appointments button.
• doctorName: Title containing the doctor's name.
• docotrDetails: Label containing the specialty, office address, and phone number of the doctor.
• aboutDoctor: Label containing the doctor's short biography.
• nextBookedAppointment: Object containing information and graphical representation of the doctor's earliest available appointment.
• btnAboutDoctor: About the Doctor button.
• btnEditDetails: Edit Profile Details button.

Class methods:

• [btnChangeOfficePicPress(): void]: Calls the goSetNewOfficePic function of the <<Controller>> DoctorDisplayController class when the btnChangeOfficePic button is pressed.
• [btnSignoutPress(): void]: Calls the SignoutDoctorUser function of the <<Controller>> DoctorDisplayController class when the btnSignout button is pressed.
• [btnAppointmentsPress(): void]: Calls the goViewAppointments function of the <<Controller>> DoctorDisplayController class when the btnAppointments button is pressed.
• [btnEditBioPress(): void]: Calls the goWriteBio function of the <<Controller>> DoctorDisplayController class when the btnAboutDoctor button is pressed.
• [btnProfileDetailsPress(): void]: Calls the goEditProfileDetails function of the <<Controller>> DoctorDisplayController class when the btnEditDetails button is pressed.

4.1.7 DoctorProfilePackage



<<Boundary>> DoctorsDBProxy

The class has already been defined in section 4.1.1.

<<Entity>> Doctor

Class attributes:

• doctorID: The unique identifier for each doctor.

• officeBackgroundImg: Image of the doctor's office.

• costPerVisit: The cost of a doctor's visit.

• officeAddress: A string containing the doctor's office address.

• speciality: The doctor's specialization.

• doctorBio: A string containing the doctor's short biography.

• workingDays: A boolean array of size 7 (one for each day of the week from Monday to Friday) where True indicates the days the doctor sets as available for appointments (e.g., if Monday and Tuesday are available, then workingDays[7] = [True True False False False False False]).

• workingHours: A boolean array of size 24 (one for each half-hour slot between 8:00 AM and 11:00 PM) following the same logic as the workingDays attribute.

Class methods:

• [Doctor(): void]: The class constructor.

<<Controller>>DoctorAddDetailsController

Class methods:

• [cancelChanges(): void]: Cancels any changes made to the doctor's information (reverts to the previous information) and redirects the doctor to the Home Page.

• [saveChanges(): void]: Saves any changes made to the doctor's information (replaces the previous information with the current information from the graphical environment) and redirects the doctor to the Home Page.

• [goSetNewProfilePic(): void]: Changes the doctor's profile picture.

<<Boundary>>DoctorUserProfileGUI

Class attributes:

• imgProfile: The doctor's profile picture.

• fieldFirstName: Text input field for the doctor's first name.

• fieldLastName: Text input field for the doctor's last name.

• fieldAddress: Text input field for the doctor's office address.

• fieldPhone: Text input field for the doctor's phone number.

• fieldSpecialty: Text input field for the doctor's specialty.

• dropdownSex: Dropdown menu for selecting the doctor's gender.

• fieldAge: Text input field for the doctor's age.

• fieldCostPerVisit: Text input field for the cost of a doctor's visit.

• btnDone: Confirmation button.

• btnCancel: Cancel button.

Class methods:

• [btnChangeProfilePicPress(): void]: Calls the goSetNewProfilePic function of the <<Controller>> DoctorAddDetailsController class when the doctor's profile picture is clicked.

• [btnDonePress(): void]: Calls the saveChanges function of the <<Controller>> DoctorAddDetailsController class when the btnDone button is pressed.

• [btnCancelPress(): void]: Calls the cancelChanges function of the <<Controller>> DoctorAddDetailsController class when the btnCancel button is pressed.

4.1.7.1 AboutDoctorPackage

(Subpackage of DoctorProfilePackage)

<<Controller>>TellAboutYourselfController

Class methods:

• [PublishNewBio(): void]: Replaces the doctor's old short biography with the new one written on the Tell About Yourself screen and redirects the doctor to the Home Page.

• [SlideToHomePage(): void]: Redirects the doctor to the Home Page.

<<Boundary>>TellAboutYourselfGUI

Class attributes:

• tellUsAboutYourself: Title containing the string "Tell us About Yourself."

• description: Label containing the string "Greet people who visit your profile with a small introduction about yourself. Remember to keep it simple!"

• bio: Text input field (up to 400 characters) for the doctor's short biography, initially containing the string "How long have you been practicing medicine? Tell us about your achievements!"

• sizeLimit: Label containing the string "400 characters remaining."

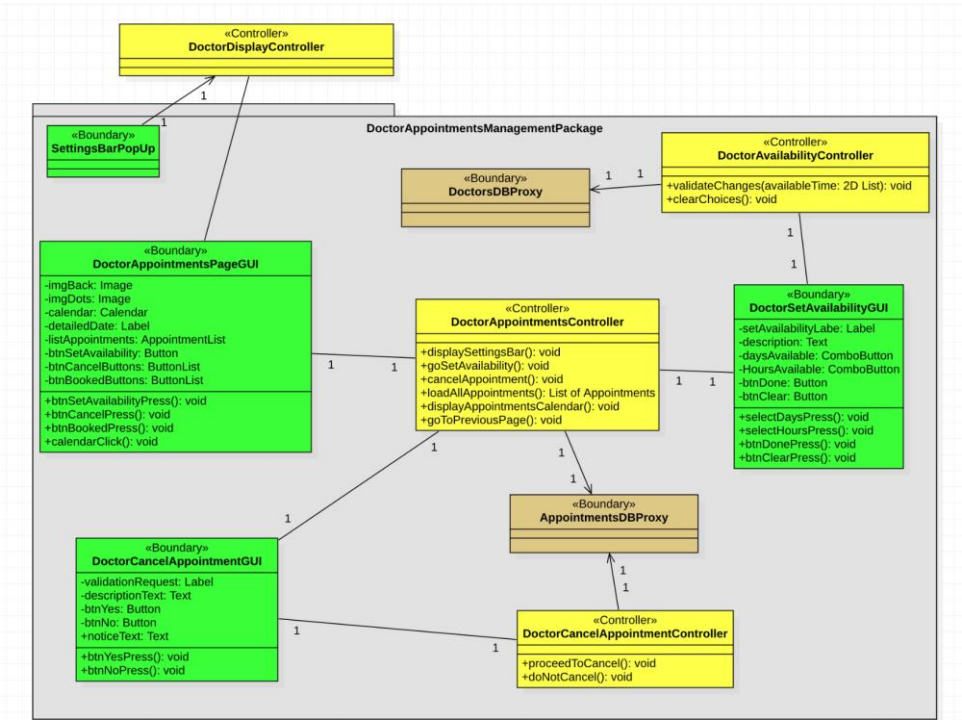
• btnPublish: Publish button.

Class methods:

• [btnPublishPress(): void]: Calls the PublishNewBio function of the <<Controller>> TellAboutYourselfController class when the btnPublish button is pressed.

• [btnBlurryPartPress(): void]: Calls the SlideToHomePage function of the <<Controller>> TellAboutYourselfController class when the upper blurry part of the screen is clicked.

4.1.8 DoctorAppointmentsManagementPackage



<<Boundary>> SettingsBarPopUp

The class has already been defined in section 4.1.4.

<<Boundary>> AppointmentsDBProxy

The class has already been defined in section 4.1.4.

<<Boundary>> DoctorsDBProxy

The class has already been defined in section 4.1.1.

<<Controller>>DoctorAppointmentsController

Class methods:

• [displaySettingsBar(): void]: Redirects the doctor to the Settings bar screen.

• [goSetAvailability(): void]: Redirects the doctor to the Doctor Set Availability screen.

• [cancelAppointment(): void]: Redirects the doctor to the Doctor Cancel Appointment screen.

• [loadAllAppointments(): List of Appointments]: Loads the appointments of the respective doctor through the <<Boundary>> AppointmentsDBProxy.

• [displayAppointmentsCalendar(): void]: Presents the personalized calendar of the doctor based on their appointments.

• [goToPreviousPage(): void]: Redirects the doctor to the exact previous screen they were on.

<<Boundary>>DoctorAppointmentsPageGUI

Class attributes:

• imgBack: The "back arrow" icon.

• imgDots: The icon with three dots.

• calendar: An "calendar" object where the user can select days and months of their choice by pressing buttons.

• listAppointments: A list of the doctor's appointments (available and booked).

• btnSetAvailability: Set Availability button.

• btnCancelButtons: A list of Cancel buttons for closed appointments (to allow the doctor to change the appointment status from closed to available).

• btnBookedButtons: A list of Booked buttons for available appointments (to allow the doctor to change the appointment status from available to closed).

Class methods:

• [btnSetAvailabilityPress(): void]: Calls the function in the <<Controller>> DoctorAppointmentsController class when the btnSetAvailability button is pressed.

• [btnCancelPress(): void]: Calls the function in the <<Controller>> DoctorAppointmentsController class when one of the btnCancelButtons is pressed.

• [btnBookedPress(): void]: Calls the function in the <<Controller>> DoctorAppointmentsController class when one of the btnBookedButtons is pressed.

• [calendarClick(): void]: Changes the color combinations of a calendar button when clicked (bold color when selected and light color when deselected).

<<Controller>>DoctorAvailabilityController

Class methods:

• [validateChanges(in availableTime: 2D List): void]: Takes the doctor's choices from the Doctor Set Availability screen as an argument, connects with the doctors' database through the

<<Boundary>> DoctorsDBProxy class, and updates the available appointment hours for the respective doctor based on the argument.

• [clearChoices(): void]: Connects with the doctors' database through the <<Boundary>> DoctorsDBProxy class and resets the status of all available appointment hours the doctor had previously (patients can no longer book appointments with the specific doctor unless the doctor reconfigures the hours for available appointments).

<<Boundary>>DoctorSetAvailabilityGUI

Class attributes:

• setAvailabilityLabel: Title containing the string "Set Availability".

• description: Label containing the string "Select your working days and hours! Tap on 'Clear' if you've made a mistake".

• daysAvailable: Combination of 7 buttons (1 for each day of the week).

• HoursAvailable: Combination of 24 buttons (1 for each half-hour from 8:00 to 23:00).

• btnDone: Confirmation button.

• btnClear: Clear button (to return to an empty selection).

Class methods:

• [selectDaysPress(): void]: Marks (changes the color in the graphical environment) a specific day when the corresponding button is pressed.

• [selectHoursPress(): void]: Marks (changes the color in the graphical environment) a specific hour when the corresponding button is pressed.

• [btnDonePress(): void]: Calls the validateChanges function of the <<Controller>>DoctorAvailabilityController class when the btnDone button is pressed.

• [btnClearPress(): void]: Calls the clearChoices function of the <<Controller>>DoctorAvailabilityController class when the btnClear button is pressed.

<<Controller>>DoctorCancelAppointmentController

Class methods:

• [proceedToCancel(): void]: Cancels a specific appointment (connects to the database and deletes it through the <<Boundary>>AppointmentsDBProxy class) and redirects the doctor to the Doctor Appointments Page screen.

• [doNotCancel(): void]: Redirects the doctor to the Doctor Appointments Page screen.

<<Boundary>>DoctorCancelAppointmentGUI

Class attributes:

• validationRequest: Title containing the string "Are you sure you want to cancel?".

• descriptionText: Label containing the string "Your patient will be notified of the cancellation".

• btnYes: Yes button.

• bnNo: No button.

• noticeText: Label containing the string "*Notification will be sent if the appointment was booked via Medbook".

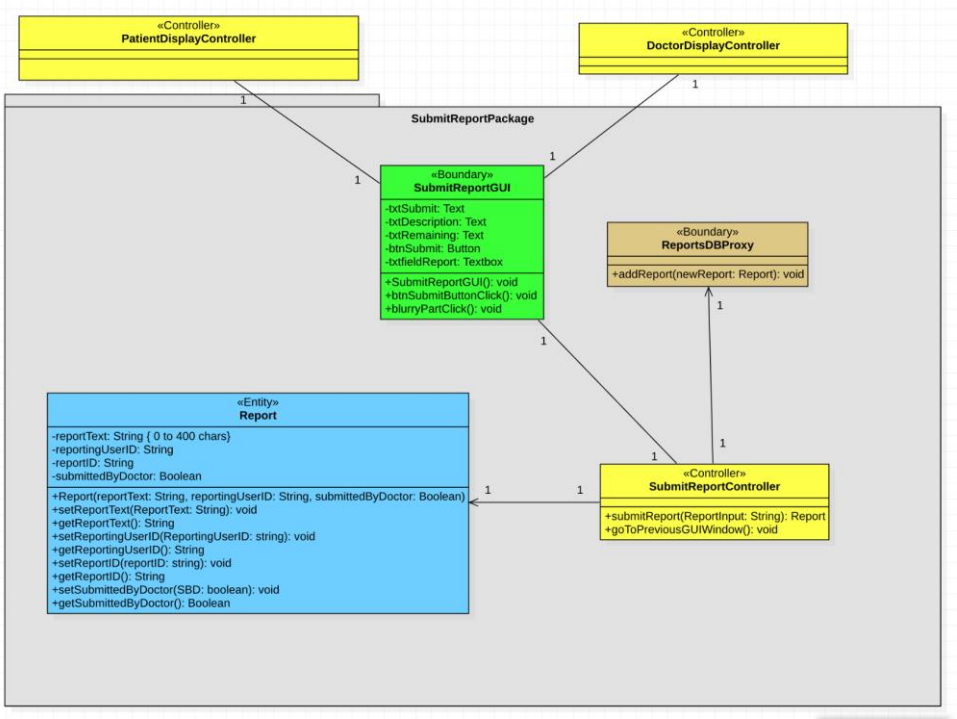
Class methods:

• [btnYesPress(): void]: Calls the proceedToCancel function of the <<Controller>>DoctorCancelAppointmentController class when the btnYes button is pressed.

• [btnNoPress(): void]: Calls the doNotCancel function of the <<Controller>>DoctorCancelAppointmentController class when the btnNo button is pressed.

4.3 <Πακέτο(α) λεξιλογίου σεναρίων χαμηλής προτεραιότητας

4.3.1 SubmitReportPackage



<<Entity>>Report

Class attributes:

• reportText: A string consisting of up to 400 characters that represents the text of the report.

• reportingUserID: A string consisting of the unique identifier of the user submitting the report.

• reportID: The unique identifier of the submitted report.

• submittedByDoctor: A variable that becomes True if the report is submitted by a doctor.

Class methods:

• [Report(reportText: String, reportingUserID: String, submittedByDoctor: Boolean)]: Constructor function of the class.

<<Boundary>>SubmitReportGUI

Class attributes:

• txtSubmit: Title containing the string "Submit your Report".

• txtDescription: Label containing the string "Let us know what's wrong... Describe the problems you are experiencing with MedBook".

• txtRemaining: Label containing the string "400 characters remaining".

• btnSubmit: Submit button.

• txtfieldReport: Text input box initially containing the string "Report a bug or a complaint. Is there an important feature missing?".

Class methods:

• [SubmitReportGUI(): void]: Constructor function of the class.

• [btnSubmitButtonClick(): void]: Calls the submitReport function of the <<Controller>>SubmitReportController class when the Submit button is pressed.

• [blurryPartClick(): void]: Calls the goToPreviousPageGUIWindow function of the <<Controller>>SubmitReportController class when the blurry part of the screen is clicked.

<<Controller>>SubmitReportController

Class methods:

• [submitReport(ReportInput: String): Report]: Function that creates and returns a Report object by setting the reportText attribute to the ReportInput string. It also calls the addReport function of the ReportsDBProxy class.

• [goToPreviousPageGUIWindow(): void]: Redirects the user to the previous screen they were on.

<<Boundary>>ReportsDBProxy

Class methods:

• [addReport(newReport: Report): void]: Connects to the report database and adds the report (Report object) received as an argument.

APPENDIX I – Glossary

The set of acronyms used in the document.

FR-*xx*	Functional Requirement *xx*
NFR-*xx*	Non-Functional Requirement *xx*