NYU Abu Dhabi
CS-UH 2012: Software Engineering

Assignment 3: Software Design and Architecture

by

Sohaila Mohammed (sm10688)

Faizan Raza (mr5985)

Khadija Khalid (kk4597)

Ajla Šačić (as15471)

Prepared for

Professor Mohamad Kassab

Instructor Dena Ahmed

Table of Contents:

Section 1: Use Cases

1. Use case title: Book an Appointment

Main success scenario:

1. The patient navigates to the appointments booking section of their profile.
2. The system presents to the patient the doctors connected to their account.
3. The patient selects the doctor they want to reach out to from their personal doctors.
4. The system presents to the patient the doctor's available appointment slots.
5. The patient selects an available time slot and chooses the preferred mode of consultation (physical meeting or video call) as prompted.
6. The system prompts the patient to provide details of why they are scheduling an appointment.
7. The patient provides information of the reason for scheduling an appointment.
8. The system confirms the appointment booking to the patient.

Extensions:

2a. The patient does not have any personal doctors connected to their account.

1. The system will not display any doctor options on the appointment booking page.
2. The system instead provides an option for the user to start adding doctors to their account.

4a. The doctor has no available time slots.

1. The user is informed by the system and redirected back to the appointments booking page.

5a. The user attempts to select an already booked time slot.

1. The system does not process the selection and alerts the user that they can only choose free time slots.

5b. The user does not choose a mode of consultation.

1. The system reminds the user that they have to choose a mode of consultation before proceeding.
2. The system asks the user again to choose a mode of consultation.

5c. The user decides he/she does not want to complete the booking anymore.

1. The user can navigate all the way back to the main page without interruptions.
2. The system disregards the user’s actions during the appointment booking and does not confirm the appointment.

7a. The patient does not provide the appointment reason.

1. The system informs the patient that the appointment cannot be booked without providing the needed details.
2. Use case continues at step 6.

2. Use case name: Issue Medical Prescription

Main Success Scenario:

1. The doctor selects the patient they want to issue a prescription for from the patients on their profile.
2. The system redirects the doctor to that patient's profile.
3. The doctor chooses to issue a prescription for the patient.
4. The system redirects the doctor to a page where they fill out the prescription details for the patient.
5. The doctor fills in the necessary details.
6. The doctor confirms the prescription.
7. The system confirms successful upload of the prescription to the patient's profile.

Extensions:

1a. The doctor does not have any patients connected to their account.

1. The system will not display any patient options on the doctor's page.

5a. The doctor does not provide all the prescription details.

1. The system displays an error message, highlighting the missing details and prompting the doctor to provide them.
2. Use case continues at step 5

7a. The system detects an internal error or downtime preventing the prescription from being uploaded.

1. The system displays an error message to the doctor, indicating that the upload was unsuccessful and suggesting to try again later.
2. The system redirects the doctor back to their home page

3. Use case title: Upload a Medical Record

Main Success Scenario:

1. The doctor selects the patient they want to upload a medical record for from the patients on their profile.
2. The system redirects the doctor to that patient's profile.
3. The doctor selects the option to upload medical records.
4. The system provides an interface for uploading medical records.
5. The doctor uploads the relevant medical records and confirms their uploading.
6. The system confirms successful upload of medical records to the patient's profile.

Extensions:

1a. The doctor does not have any patients connected to their account.

2. The system will not display any patient options on the doctor's page.

5a. The doctor attempts to proceed without having uploaded a file.

1. The system displays an error message, informing the doctor that they need to provide a file to upload a medical record for the patient.
2. Use case continues at step 4.

5b. The doctor uploaded a file format that is not supported.

1. The system informs the doctor of the error and prompts them to upload the medical record in one of the website's accepted file formats.
2. Use case continues at step 4.

5c. The doctor uploaded a file that is too large.

1. The system informs the doctor of the error and prompts them to upload a file within the size limit.
2. Use case continues at step 4.

6a. The system detects an internal error or downtime preventing the medical record from being uploaded.

1. The system displays an error message to the doctor, indicating that the upload was unsuccessful and suggesting to try again later.
2. The system redirects the doctor back to their home page.

4. Use case title: Request Intelligent Diagnostic Support

Main success scenario:

1. A doctor navigates to the symptom checker feature available on their profile.

2. The system redirects the doctor to the interaction interface and prompts the doctor to enter the relevant symptoms to diagnose.

3. The doctor enters the symptoms.

4. The system processes the input through the ApiMedic API.

5. ApiMedic returns to the system the potential diagnoses based on the inputted symptoms.

6. The system displays the ApiMedic diagnosis results for the doctor to review and consider based on his/her own professional judgment.

Extensions:

3a. The doctor does not provide the symptoms.

1. The system informs the doctor that it cannot provide diagnostic support without symptoms.
2. The system re-prompts the doctor to enter the relevant symptoms to diagnose.

4a. The system detects that the ApiMedic service is unavailable.

1. The system notifies the doctor of the issue and suggests retrying later.
2. The system redirects the doctor back to their home page.

5. Use case title: Initiate Video Call Appointment

Main Success Scenario:

1. The patient selects the appointment from their scheduled appointments on their profile.
2. The system redirects the patient to the details page of the selected appointment.
3. The patient chooses the option to join the video call.
4. The system initiates the video call session with the selected doctor through Zoom.
5. Zoom establishes the connection between the patient and the doctor.
6. The patient and doctor conduct the consultation virtually.

Extensions:

1a. The patient does not have any scheduled appointments on their profile.

1. The system will not display any appointment options on the patient's page.

3a. The patient decides not to join the video call

1. The system allows the patient to navigate back to their profile dashboard.

3b. The appointment was originally scheduled to be in-person.

1. The system informs the patient that there is no video call to be joined because the appointment was originally scheduled to be in-person.

4a. The system detects that Zoom's services are unavailable.

1. The system informs the patient of the error and prompts them to try again later.

5a. The doctor is not available at the video call appointment.

1. Through Zoom, the patient can wait for the doctor or exit the video call.
2. The system allows the user to reschedule the appointment.

Section 2 : *System Sequence Diagram (SSD)*

1. Book an Appointment:

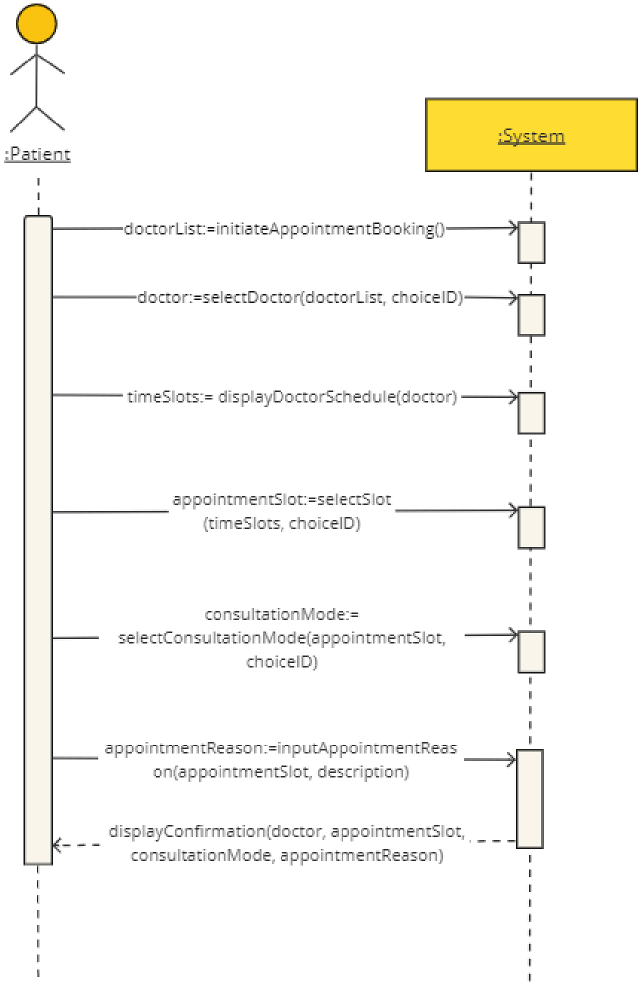

:Patient
:System
doctorList:=initiateAppointmentBooking()
doctor:=selectDoctor(doctorList, choiceID)
timeSlots:= displayDoctorSchedule(doctor)
appointmentSlot:=selectSlot (timeSlots, choiceID)
consultationMode:= selectConsultationMode(appointmentSlot, choiceID)
appointmentReason:=inputAppointmentReason(appointmentSlot, description)
displayConfirmation(doctor, appointmentSlot, consultationMode, appointmentReason)

Figure 1: SSD for Booking an Appointment

2. Upload a New Prescription:

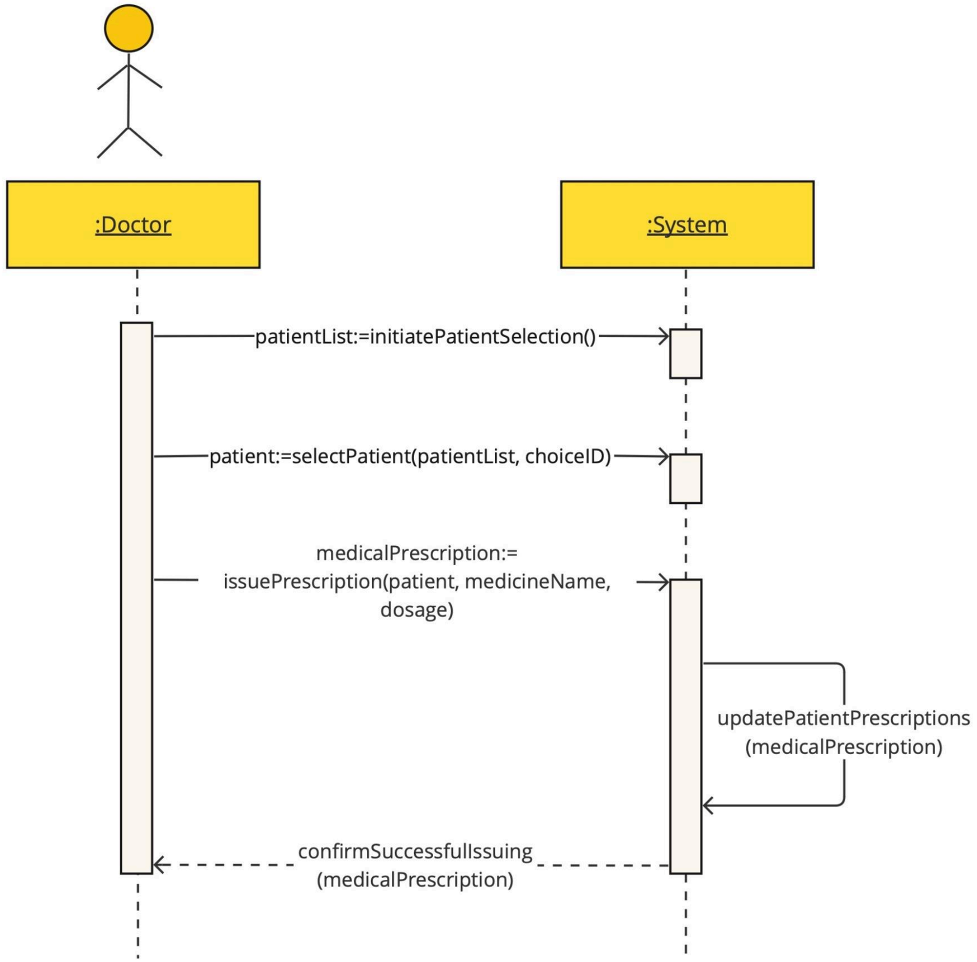

:Doctor
:System
patientList:=initiatePatientSelection()
patient:=selectPatient(patientList, choiceID)
medicalPrescription:= issuePrescription(patient, medicineName, dosage)
updatePatientPrescriptions (medicalPrescription)
confirmSuccessfulIssuing (medicalPrescription)

Figure 2: SSD for Uploading a New Prescription

3. Upload a Medical Record:

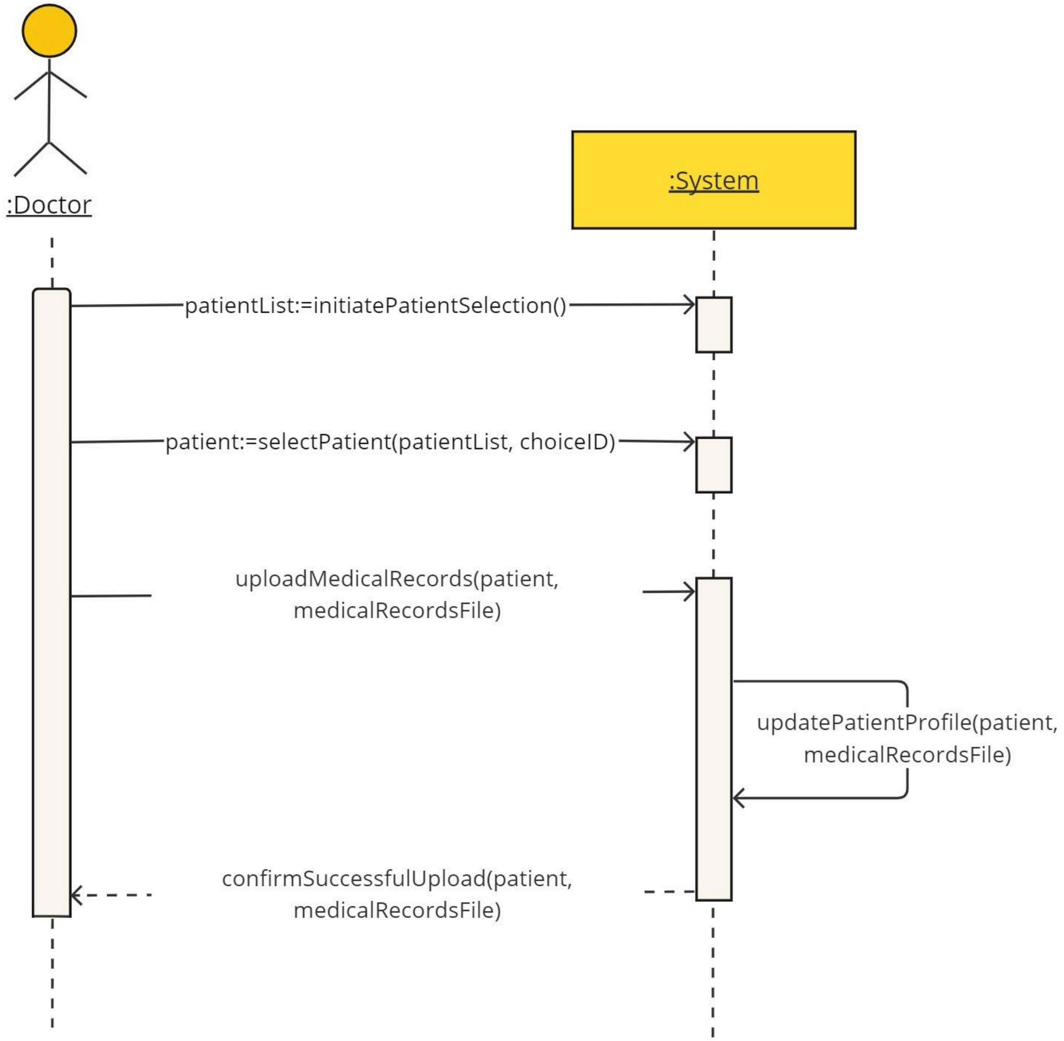

:Doctor
:System
patientList:=initiatePatientSelection()
patient:=selectPatient(patientList, choiceID)
uploadMedicalRecords(patient, medicalRecordsFile)
updatePatientProfile(patient, medicalRecordsFile)
confirmSuccessfulUpload(patient, medicalRecordsFile)

Figure 3: SSD for Uploading a Medical Record

4. Request Intelligent Diagnostic Support:

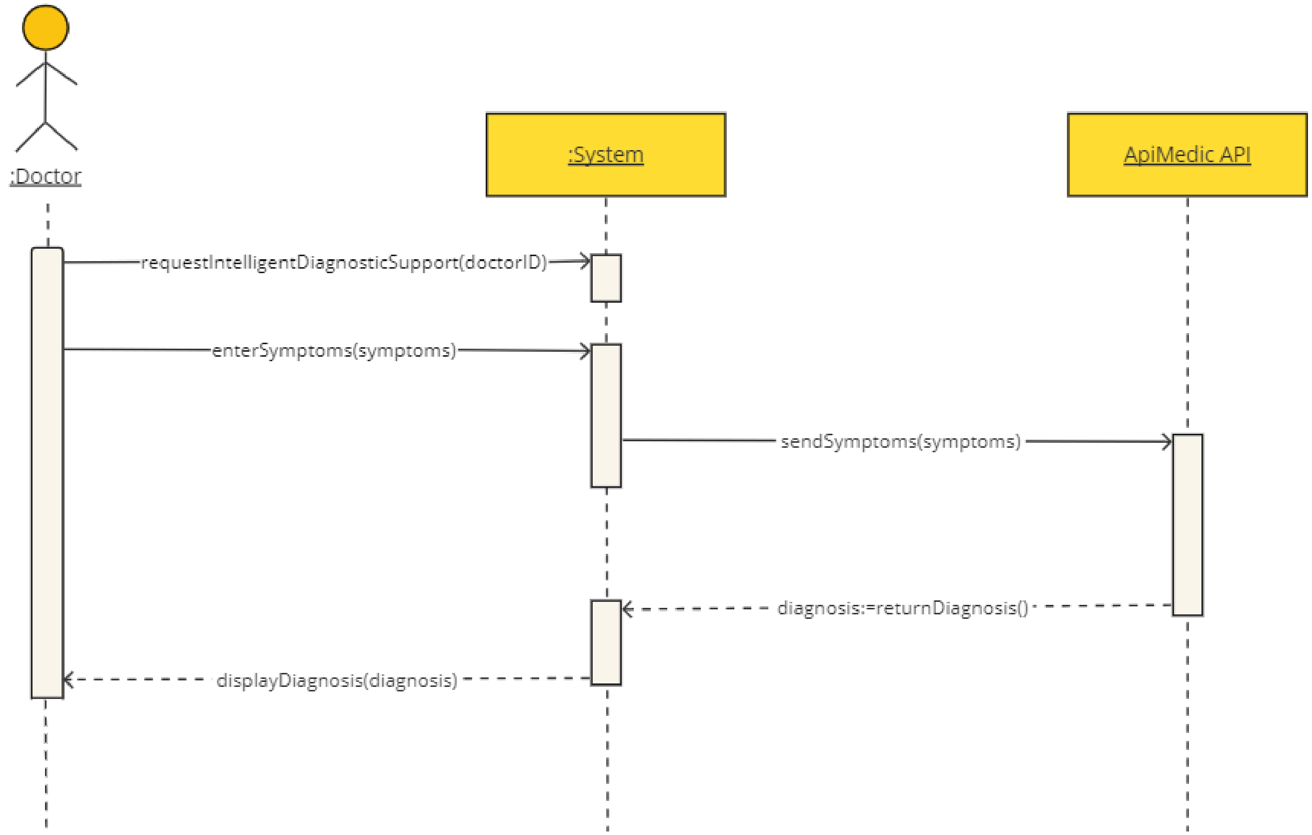

:Doctor
:System
ApiMedic API
requestIntelligentDiagnosticSupport(doctorID)
enterSymptoms(symptoms)
sendSymptoms(symptoms)
diagnosis:=returnDiagnosis()
displayDiagnosis(diagnosis)

Figure 4: SSD for Requesting Intelligent Diagnostic Support

5. Initiate Video Call Appointment:

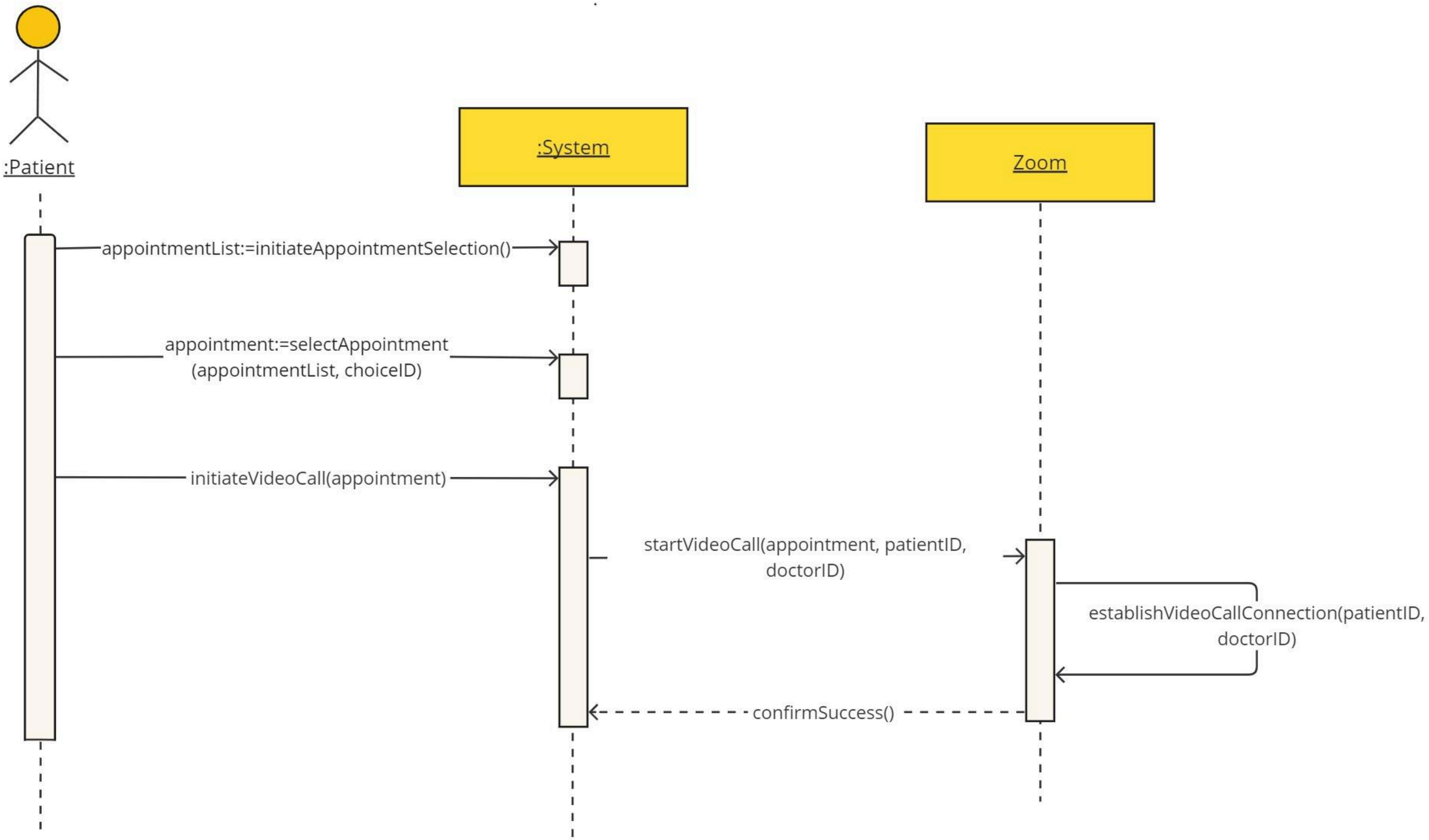

:Patient
:System
Zoom
appointmentList:=initiateAppointmentSelection()
appointment:=selectAppointment
(appointmentList, choiceID)
initiateVideoCall(appointment)
startVideoCall(appointment, patientID,
doctorID)
establishVideoCallConnection(patientID,
doctorID)
confirmSuccess()

Figure 5: SSD for Initiating a Video Call Appointment

Section 3: *Domain Model*

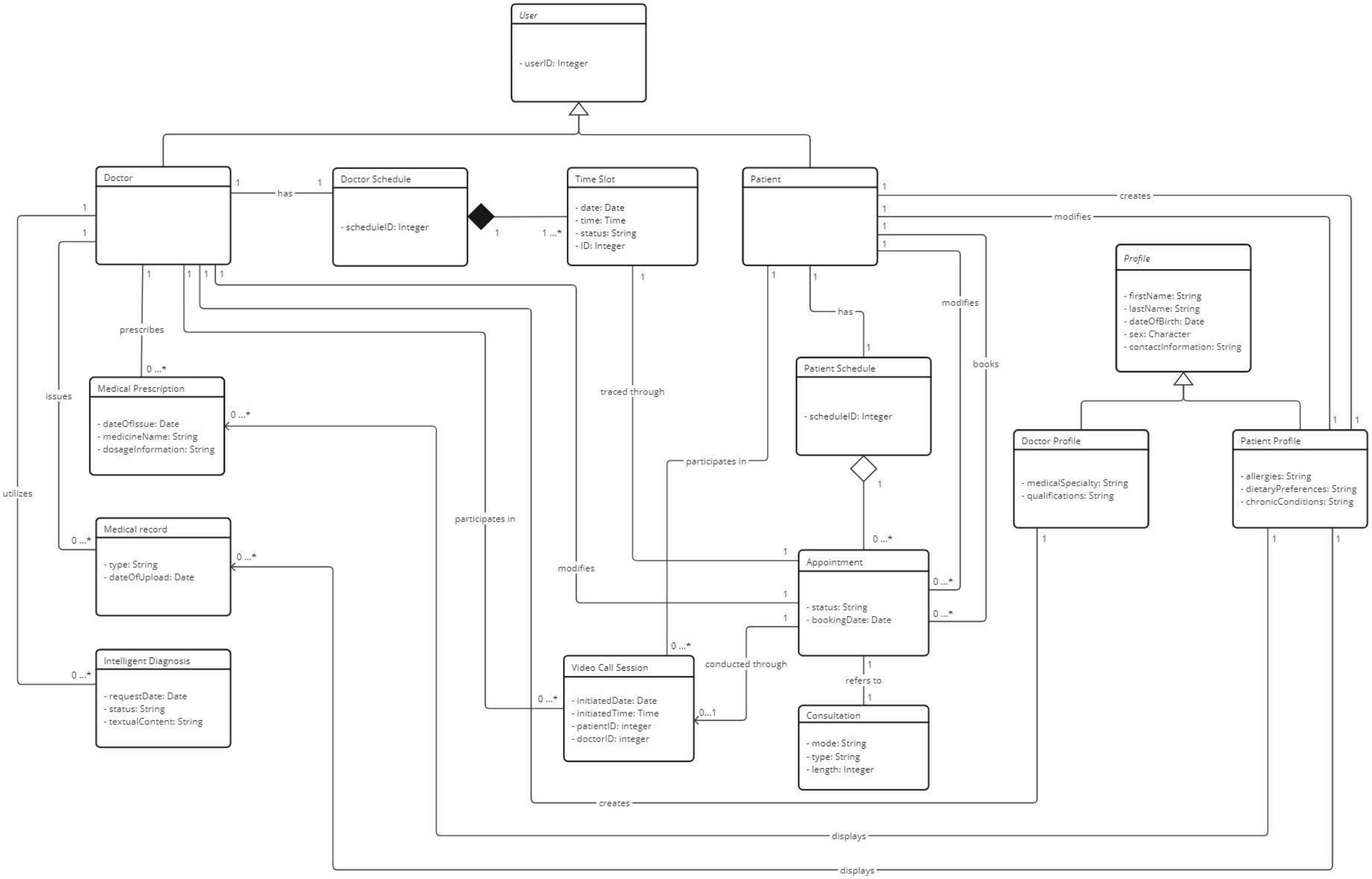


Figure 6: Domain Model Diagram

Section 4: *Operation Contracts*

1. Book an Appointment:

- Name: selectSlot(timeSlots, choiceID)

- Responsibilities: To allow a patient to select a specific slot for an appointment based on the personal doctor’s availability schedule.

- Type: System

- Cross References: System functions: 2.1, Use case: Book an Appointment

- Notes:

- Exceptions: If the “choiceID” does not correspond to a free, available slot in the “timeSlots” of the doctor’s schedule, the system should indicate that the selection was invalid.

- Output:

- Pre-conditions:

1. The patient is logged in and connected on the system to the doctor with whom they’re booking an appointment.
2. The “timeSlots” must be populated with the doctor’s available appointment times for every date based on his/her schedule.

- Post-conditions:

1. An object instance of “Appointment” is created. *(instance creation)*
2. An association is formed between the “Appointment” instance and the “TimeSlot” selected by the user. *(association formed)*

3. An association is formed between the “patientSchedule” and the new “Appointment” instance. *(association formed)*
4. The “Appointment” instance’s “bookingDate” attribute is modified to be the current date. *(attribute modification)*
5. The “Appointment” instance’s “status” attribute is modified to be “pending”. *(attribute modification)*
6. The status of the “TimeSlot” the patient selected will be updated to “unavailable” to signify that it has been chosen by a patient. *(attribute modification)*

2. Upload a New Prescription:

Name: issuePrescription(patient, medicineName, dosage)

Responsibilities: To allow a doctor to issue a digital medical prescription with all its details for one of his/her patients.

Type: System

Cross References: System functions: 4.1, Use case: Issue Medical Prescription

Notes:

Exceptions: If the “patient” object does not correspond to an existing patient in the system, the system should indicate the error accordingly.

Output:

Pre-conditions:

1. The doctor is logged in, authenticated, and connected on the system to the patient for whom they’re issuing a prescription.
2. The “patient” corresponding to a patient must exist in the system.

Post-conditions:

1. An instance of the “medicalPrescription” is created. (*instance creation*)
2. The “medicalPrescription” instance’s “medicineName” attribute is modified based on the doctor’s input. *(attribute modification*)
3. The “medicalPrescription” instance’s “dosage” attribute is modified based on the doctor’s input. *(attribute modification*)
4. An association is formed between the “patientProfile” and the new “medicalPrescription”. (*association formed*)

3. Upload a Medical Record:

Name: uploadMedicalRecords(patient, medicalRecordsFile)

Responsibilities: To allow a doctor to upload the medical records file associated with a patient to the patient's profile.

Type: System

Cross References: System functions: 3.4, Use case: Upload a Medical Record

Notes:

Exceptions: If the uploaded file format is not supported or the uploaded file is too large, the system should indicate the error to the doctor accordingly.

Output:

Pre-conditions:

- The doctor is logged in, authenticated, and connected on the system to the patient for whom they're issuing a prescription.
- The "patient" corresponding to a patient must exist in the system.

Post-conditions:

- An object instance of "medicalRecord" is created. (*instance creation*).
- The date attribute of the "medicalRecord" instance is updated to the current date. *(attribute modification*)
- An association is formed between the "patientProfile" and "medicalRecord". (*association formed*)

4. Request Intelligent Diagnostic Support:

- Name: sendSymptoms(symptoms)

- Responsibilities: To send the symptoms inputted by the doctor to the ApiMedic API for processing.

- Type: System

- Cross References: System functions: 7.1, Use case: Request intelligent diagnostic support

- Notes:

- Exceptions: If the ApiMedic API is unavailable, or if the symptoms provided are in an incorrect format, the system should indicate the error to the doctor accordingly.

- Output: Request sent to the ApiMedic API

- Pre-conditions:

1. The doctor must be logged in and authenticated in order to use the diagnostic support feature.
2. The "symptoms" must be in a format accepted by the ApiMedic API.

- Post-conditions:

1. An object instance "Intelligent Diagnosis" is created. *(instance creation)*
2. An association is formed between the "Doctor" and the new "Intelligent Diagnosis" instance. *(association formed)*
3. The "Intelligent Diagnosis" instance's "Request Date" attribute is modified to be the current date. *(attribute modification)*
4. The "Intelligent Diagnosis" instance's "status" attribute is modified to be "pending". *(attribute modification)*

5. Initiate Video Call Appointment:

Name: initiateVideoCall(appointment)

Responsibilities: To allow a patient to initiate a scheduled video call appointment with the selected doctor.

Type: System

Exceptions: If the “appointment” does not correspond to a valid appointment in the patient's list of scheduled appointments, the system should indicate that the appointment provided is invalid. Additionally, if the “appointment” corresponds to an in-person consultation, the system should indicate so to the user and inform them that there is no video call to join for this appointment.

Pre-conditions:

1. The patient is logged in and has a list of scheduled appointments.
2. The “appointment” corresponds to a valid, virtual appointment in the patient's list of scheduled appointments.

Post Conditions:

1. An object instance of "VideoCallSession" is created. *(instance creation)*
2. An association is formed between the "VideoCallSession" instance and the selected "appointment" instance. *(association formed)*
3. The “VideoCallSession” instance’s “initiatedDate” attribute is modified based on the date of its creation. *(attribute modification)*
4. The “VideoCallSession” instance’s “initiatedTime” attribute is modified based on the time of its creation. *(attribute modification)*
5. The “VideoCallSession” instance’s “patientID” attribute is modified based on the patient participating in the video call. *(attribute modification)*
6. The “VideoCallSession” instance’s “doctorID” attribute is modified based on the patient participating in the video call. *(attribute modification)*

7. An association is formed between the "VideoCallSession" instance and the corresponding “patient”. *(association formed)*
8. An association is formed between the "VideoCallSession" instance and the corresponding “doctor”. *(association formed)*

Section 5: *UML Interaction Diagram*

1. Make an Appointment

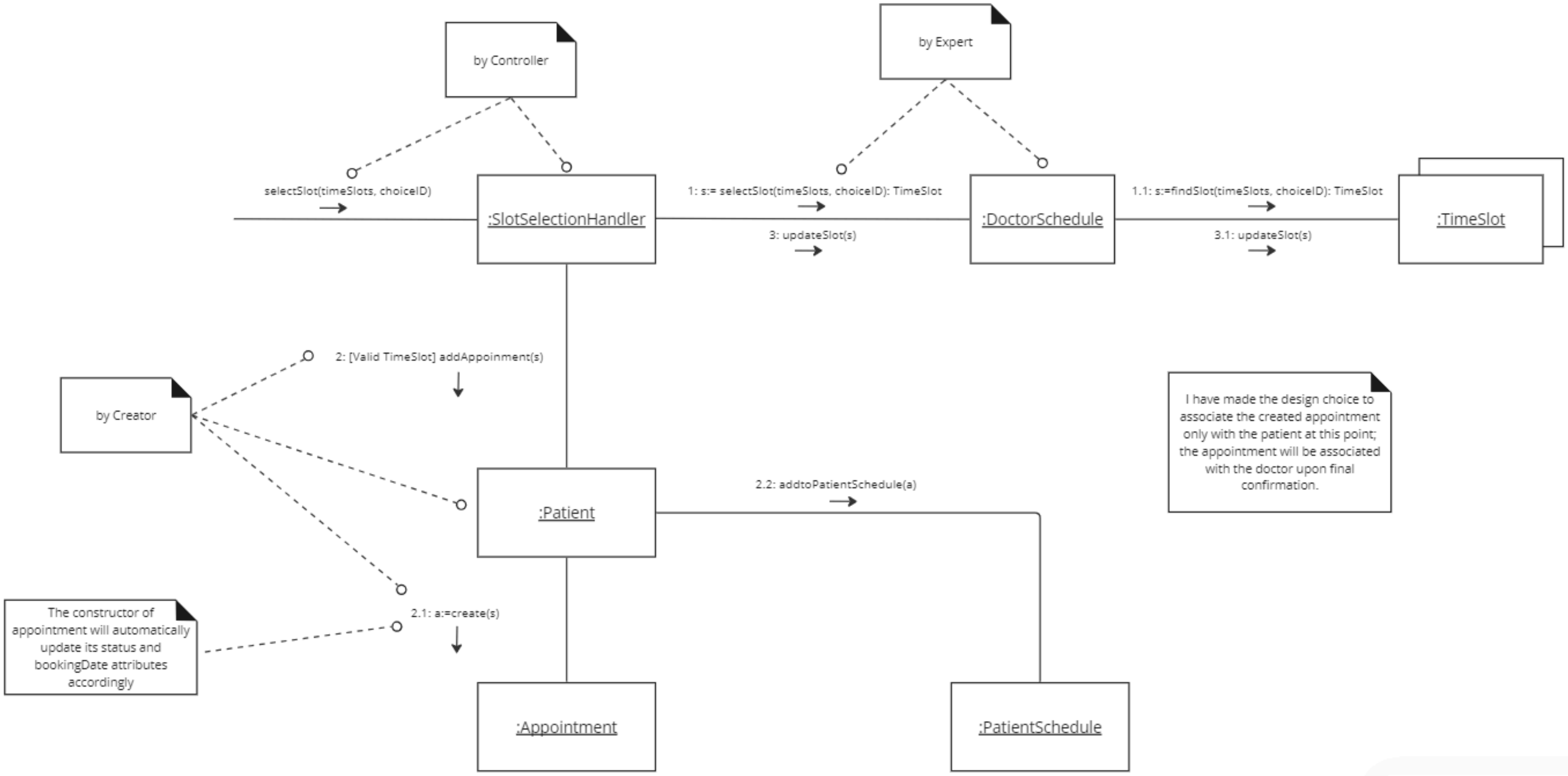

by Controller
by Expert
selectSlot(timeSlots, choiceID)
:SlotSelectionHandler
1: s:= selectSlot(timeSlots, choiceID): TimeSlot
3: updateSlot(s)
:DoctorSchedule
1.1: s:=findSlot(timeSlots, choiceID): TimeSlot
3.1: updateSlot(s)
:TimeSlot
by Creator
2: [Valid TimeSlot] addAppoinment(s)
I have made the design choice to associate the created appointment only with the patient at this point; the appointment will be associated with the doctor upon final confirmation.
:Patient
2.2: addtoPatientSchedule(a)
The constructor of appointment will automatically update its status and bookingDate attributes accordingly
2.1: a:=create(s)
:Appointment
:PatientSchedule

Figure 7: Interaction diagram for Making an Appointment

2. Upload a New Prescription:

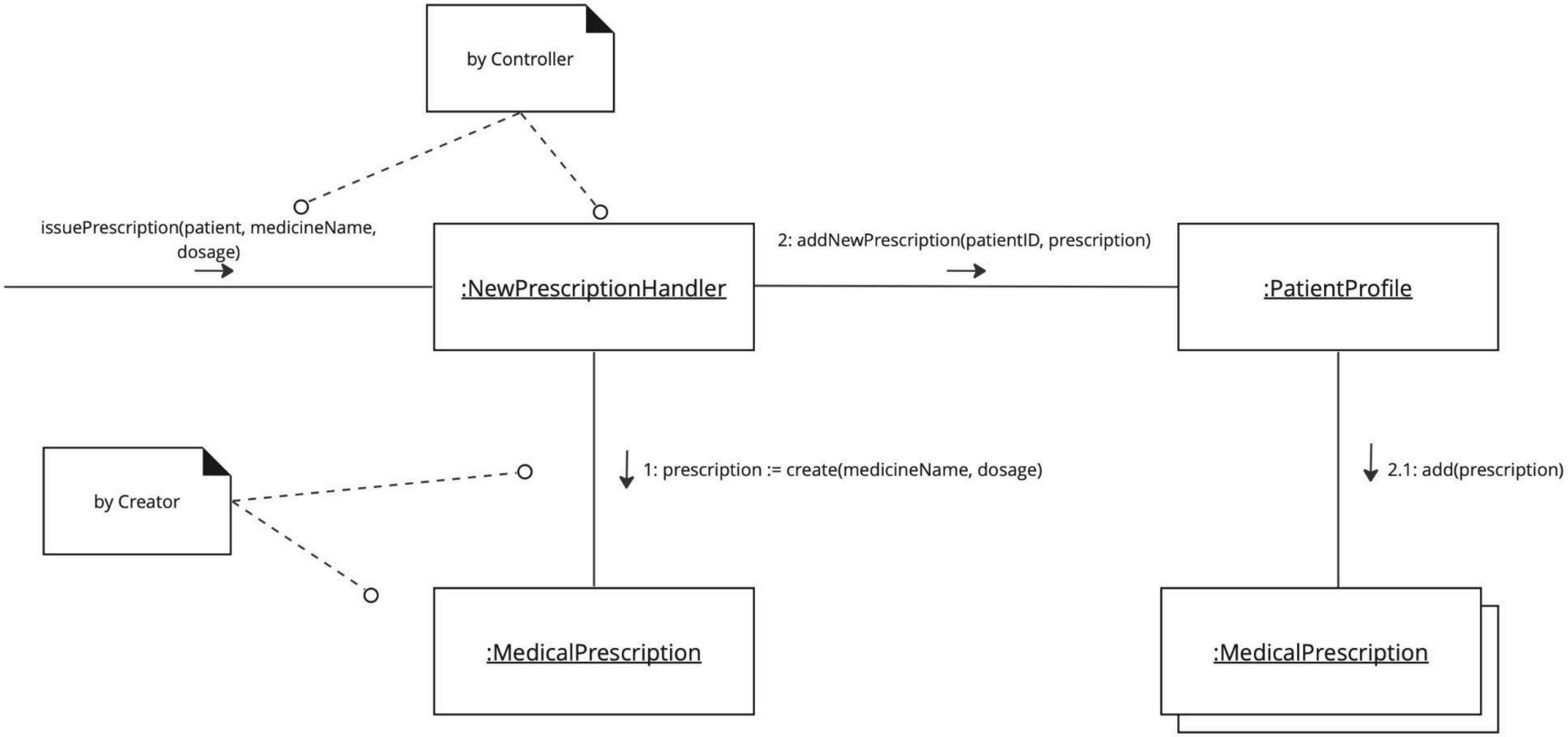

by Controller
issuePrescription(patient, medicineName, dosage)
:NewPrescriptionHandler
2: addNewPrescription(patientID, prescription)
:PatientProfile
by Creator
1: prescription := create(medicineName, dosage)
2.1: add(prescription)
:MedicalPrescription
:MedicalPrescription

Figure 8: Interaction diagram for Uploading a New Prescription

3. Upload a Medical Record:

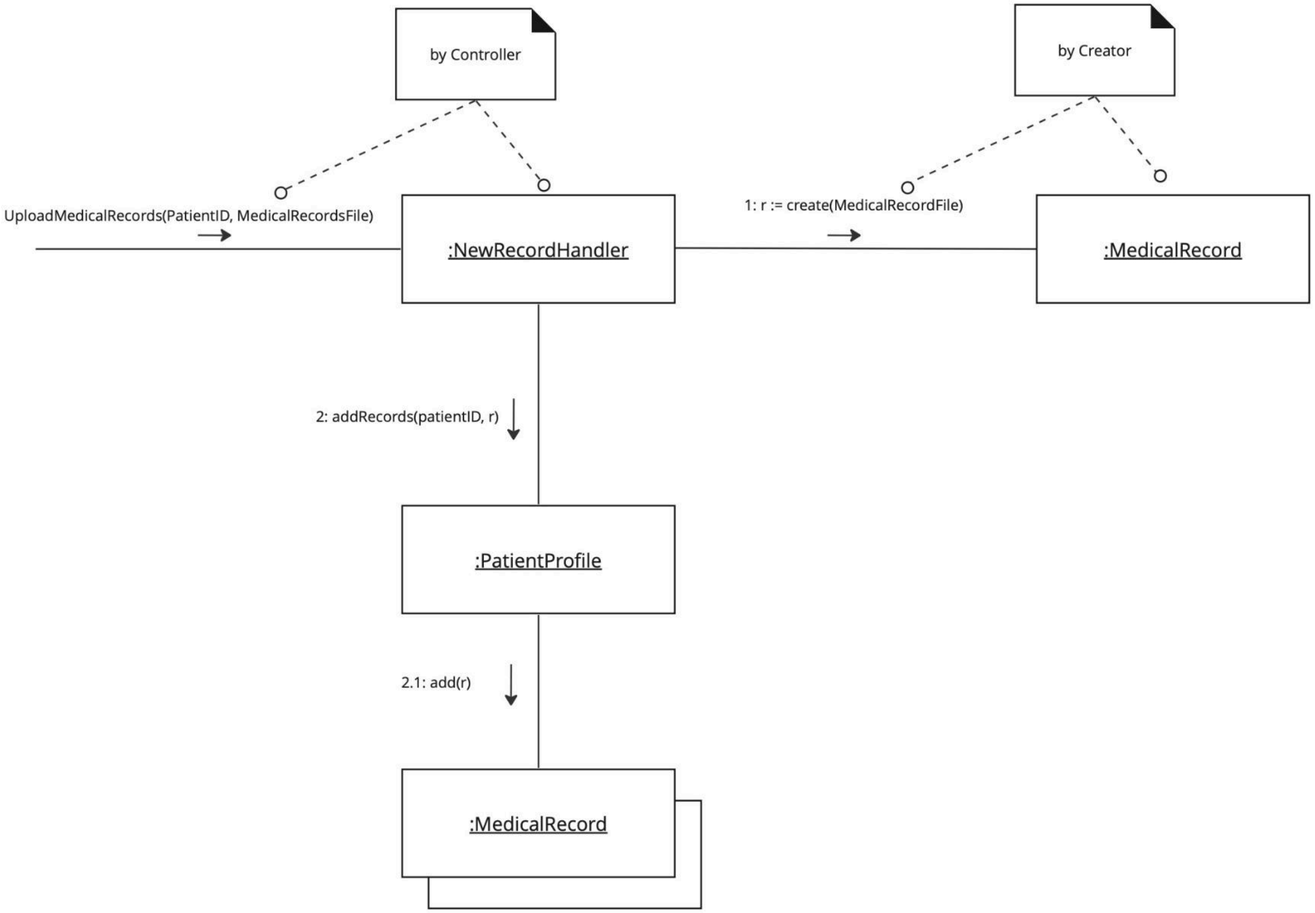

by Controller
by Creator
UploadMedicalRecords(PatientID, MedicalRecordsFile)
:NewRecordHandler
1: r := create(MedicalRecordFile)
:MedicalRecord
2: addRecords(patientID, r)
:PatientProfile
2.1: add(r)
:MedicalRecord

Figure 9: Interaction diagram for Uploading a Medical Record

4. Request Intelligent Diagnostic Support:

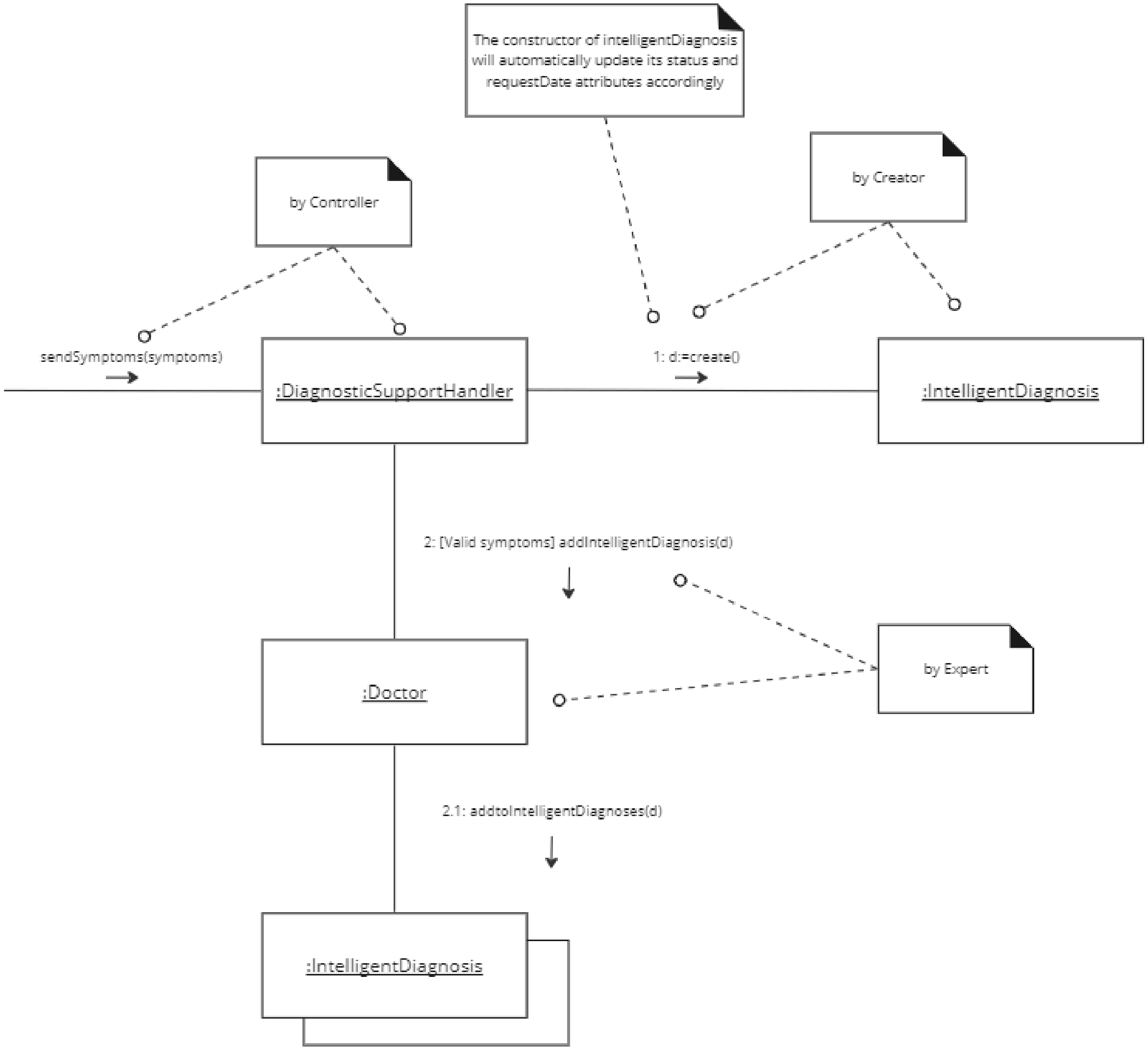

The constructor of intelligentDiagnosis will automatically update its status and requestDate attributes accordingly
by Controller
by Creator
sendSymptoms(symptoms)
:DiagnosticSupportHandler
1: d:=create()
:IntelligentDiagnosis
2: [Valid symptoms] addIntelligentDiagnosis(d)
by Expert
:Doctor
2.1: addtoIntelligentDiagnoses(d)
:IntelligentDiagnosis

Figure 10: Interaction diagram for Requesting Intelligent Diagnostic Support

5. Initiate Video Call Appointment:

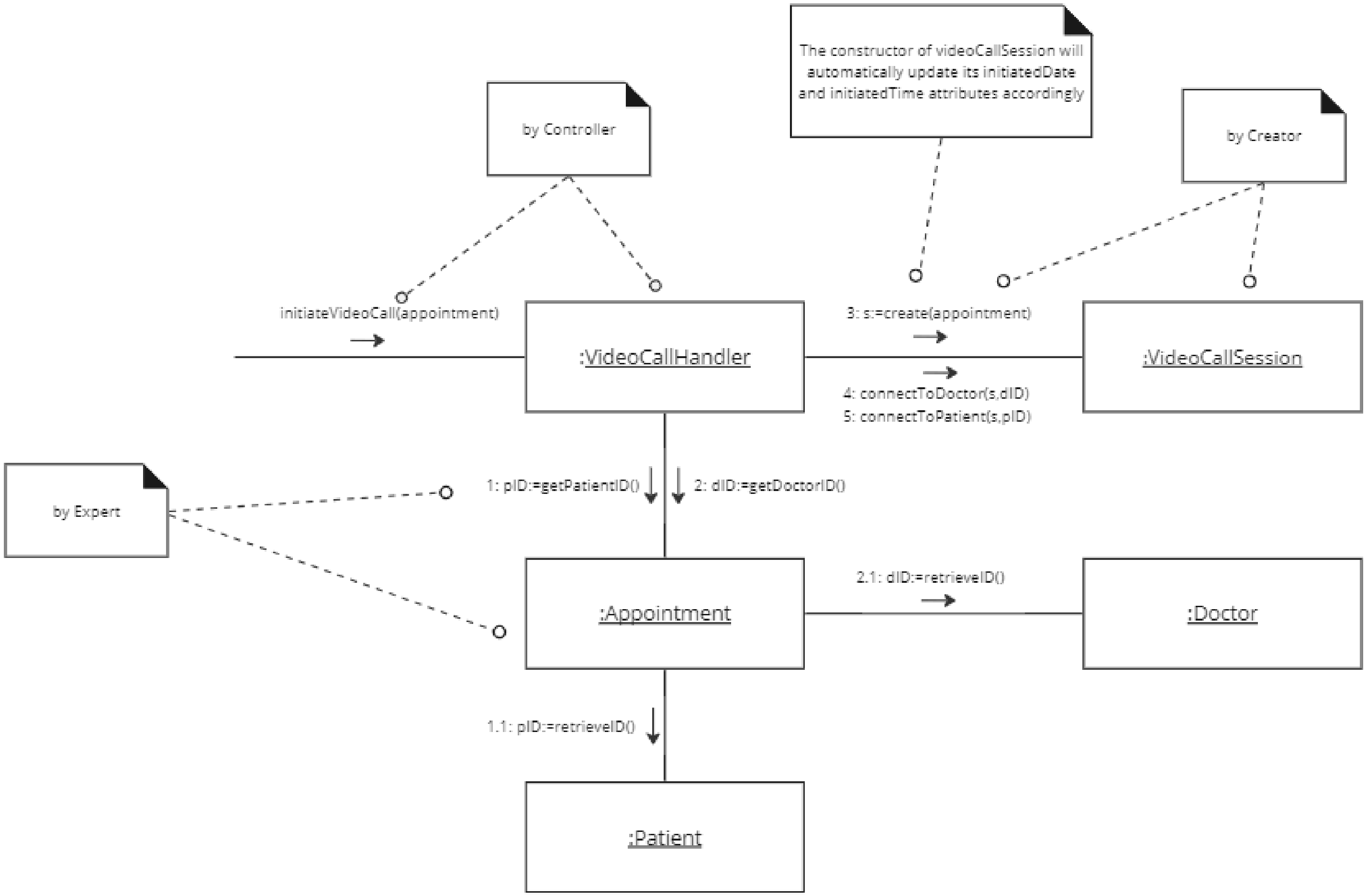

The constructor of videoCallSession will automatically update its initiatedDate and initiatedTime attributes accordingly
by Controller
by Creator
initiateVideoCall(appointment)
:VideoCallHandler
3: s:=create(appointment)
4: connectToDoctor(s,dID)
5: connectToPatient(s,pID)
:VideoCallSession
by Expert
1: pID:=getPatientID()
2: dID:=getDoctorID()
:Appointment
2.1: dID:=retrieveID()
:Doctor
1.1: pID:=retrieveID()
:Patient

Figure 11: Interaction diagram for Initiating a Video Call Appointment

Section 6: *Class diagram*

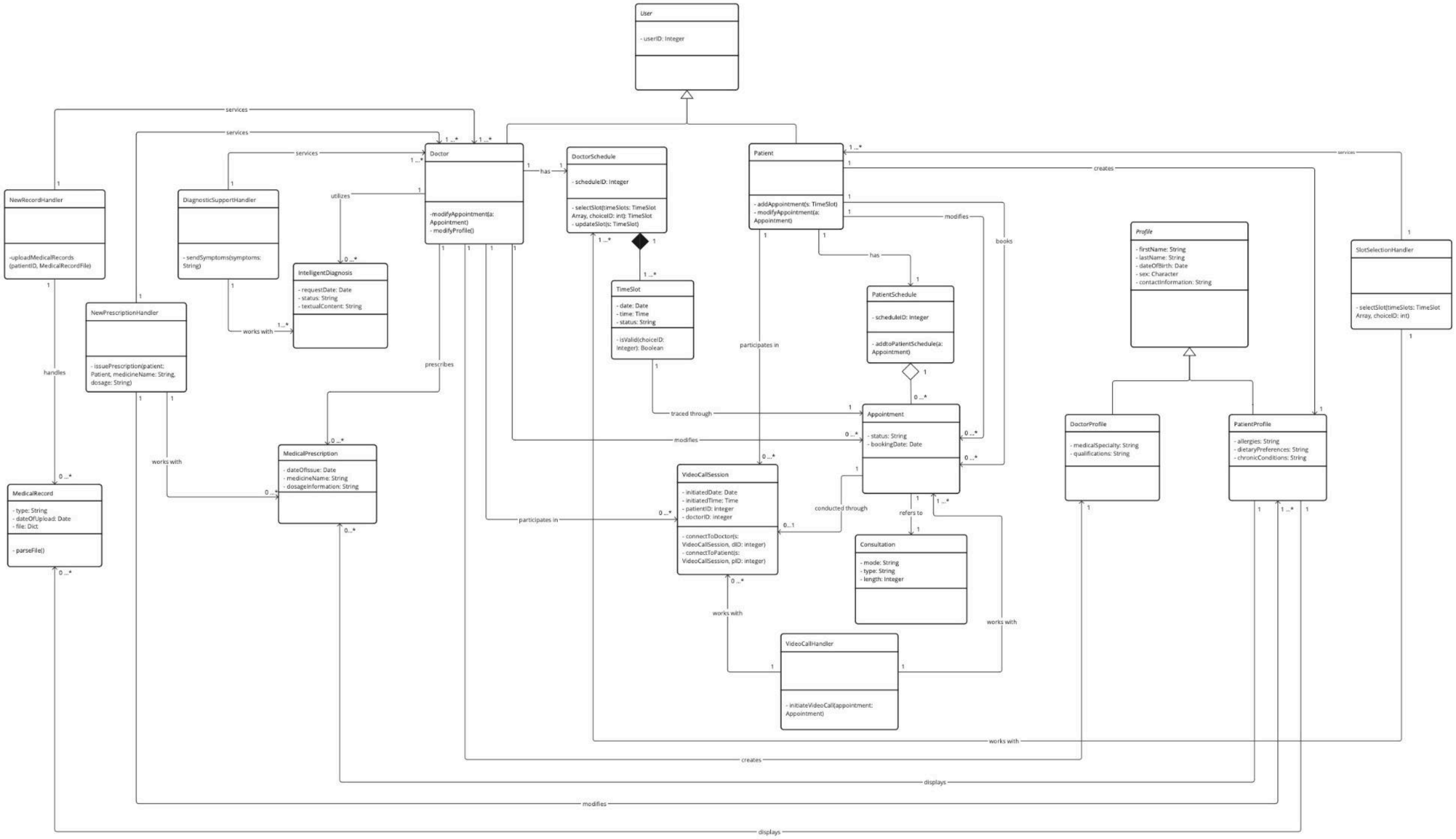
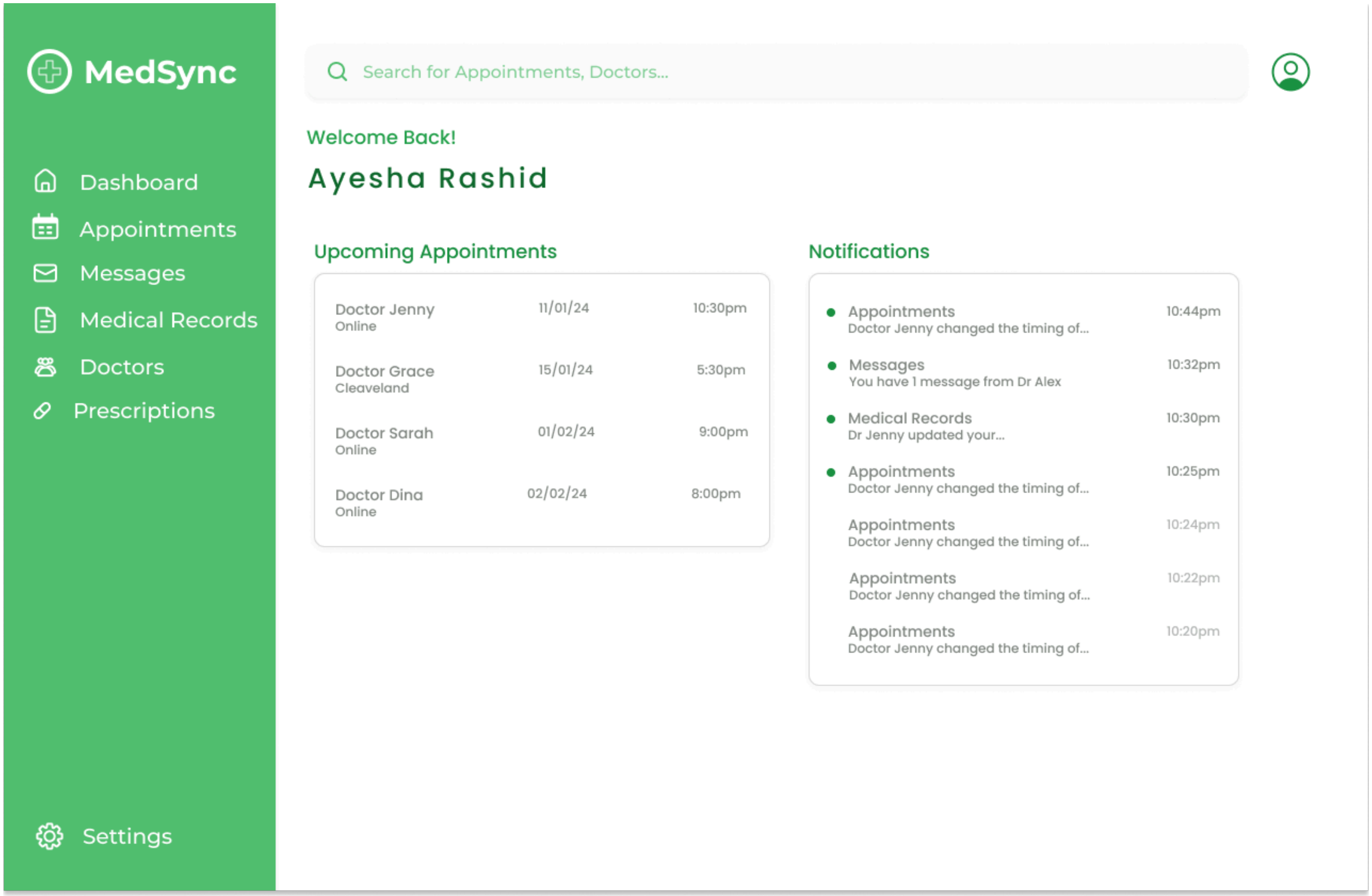


Figure 12: Class Diagram

Section 7: *User Interface (UI)*



Welcome Back!

Ayesha Rashid

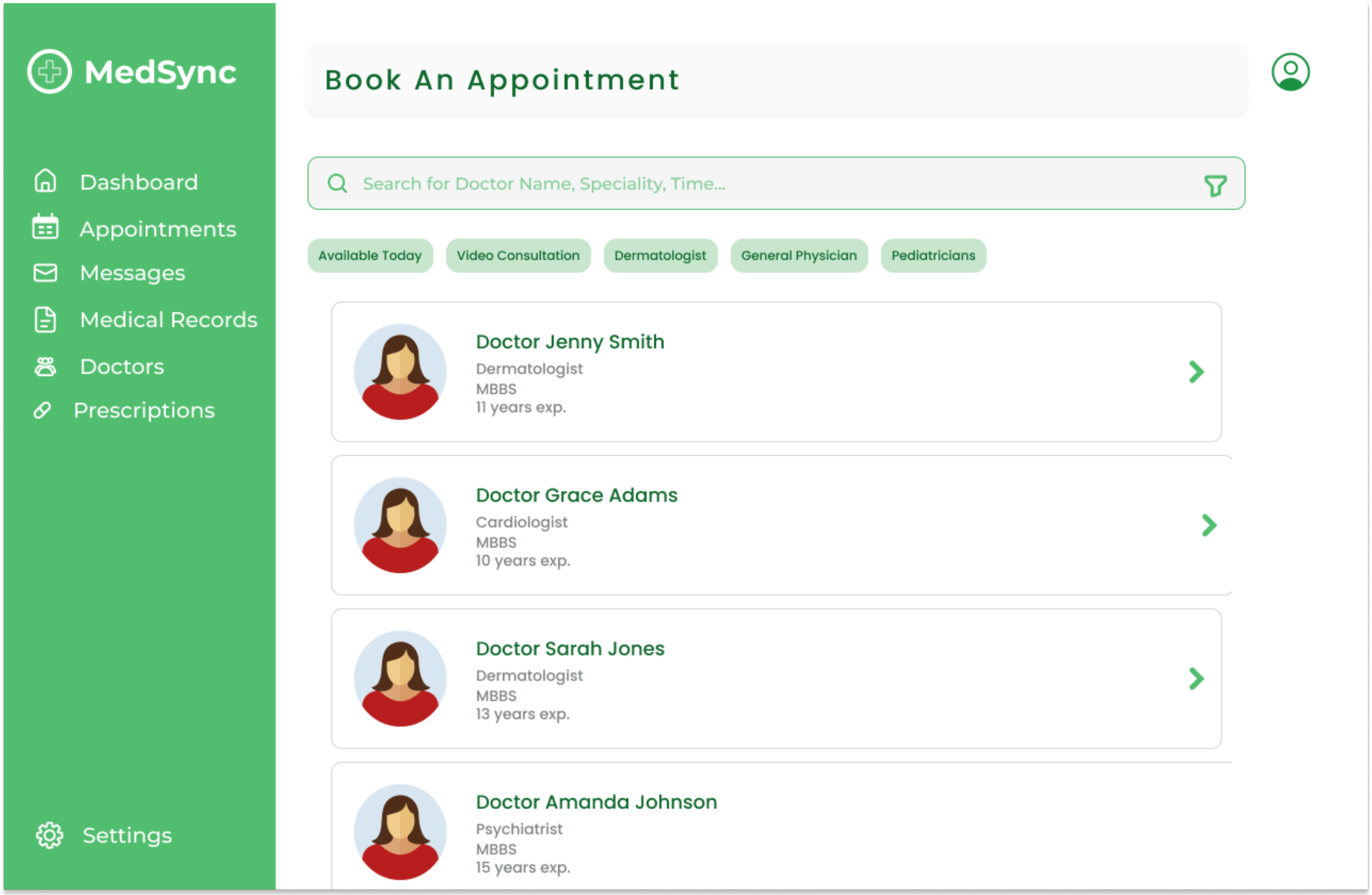
Upcoming Appointments

Doctor Jenny Online	11/01/24	10:30pm
Doctor Grace Cleaveland	15/01/24	5:30pm
Doctor Sarah Online	01/02/24	9:00pm
Doctor Dina Online	02/02/24	8:00pm

Notifications

●	Appointments Doctor Jenny changed the timing of...	10:44pm
●	Messages You have 1 message from Dr Alex	10:32pm
●	Medical Records Dr Jenny updated your...	10:30pm
●	Appointments Doctor Jenny changed the timing of...	10:25pm
	Appointments Doctor Jenny changed the timing of...	10:24pm
	Appointments Doctor Jenny changed the timing of...	10:22pm
	Appointments Doctor Jenny changed the timing of...	10:20pm

Figure 13: Patient Dashboard



Book An Appointment

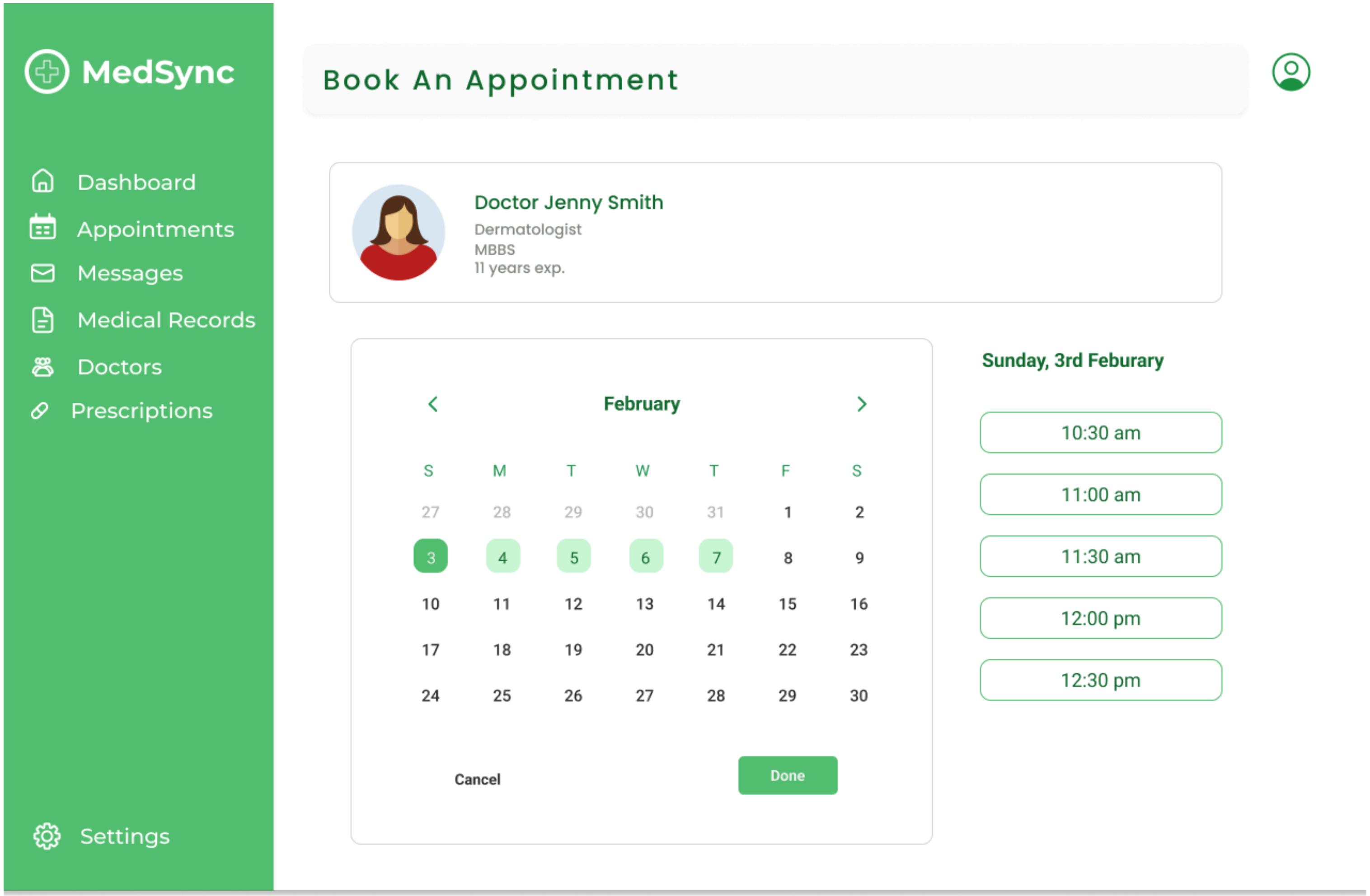
Doctor Jenny Smith
Dermatologist
MBBS
11 years exp.

Doctor Grace Adams
Cardiologist
MBBS
10 years exp.

Doctor Sarah Jones
Dermatologist
MBBS
13 years exp.

Doctor Amanda Johnson
Psychiatrist
MBBS
15 years exp.

Figure 14: New Appointment Search



Book An Appointment

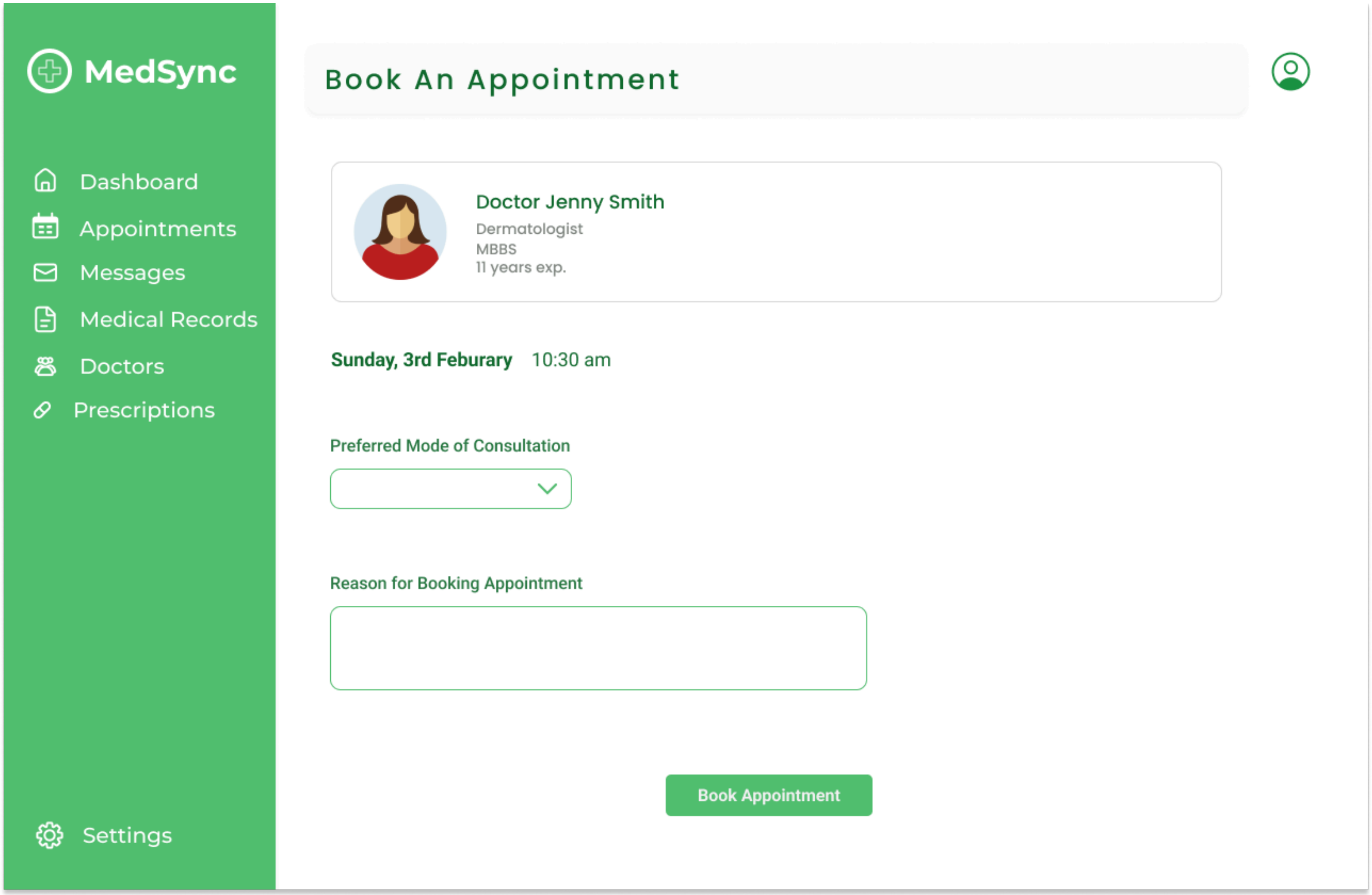
Doctor Jenny Smith
Dermatologist
MBBS
11 years exp.

February

S	M	T	W	T	F	S
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

Sunday, 3rd Feburary

Figure 15: Doctor Availability Schedule



Book An Appointment

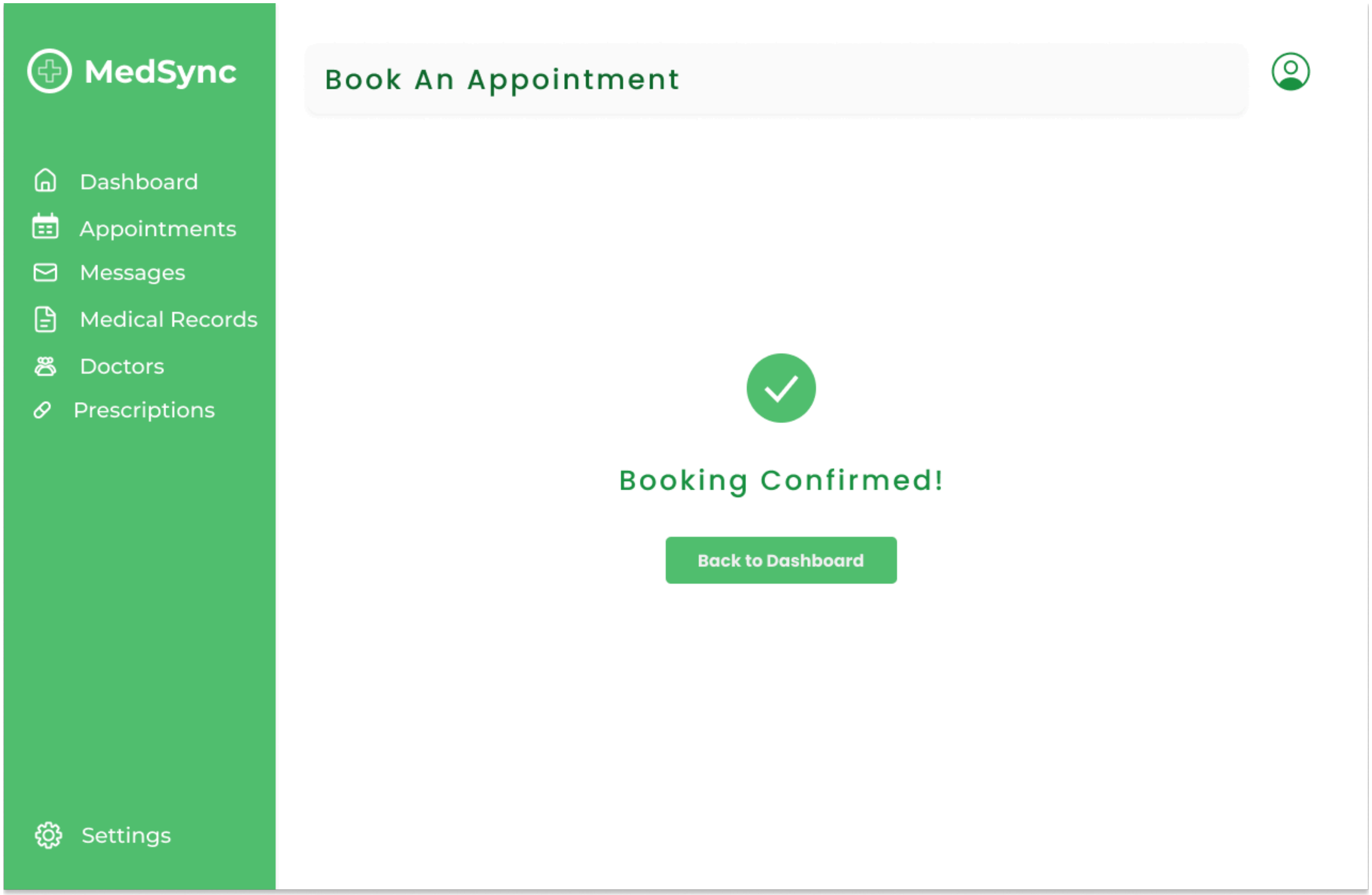
Doctor Jenny Smith
Dermatologist
MBBS
11 years exp.

Sunday, 3rd Feburary 10:30 am

Preferred Mode of Consultation

Reason for Booking Appointment

Figure 16: Appointment Details



Book An Appointment

Booking Confirmed!

Figure 17: Appointment Confirmation

Section 8: Appendix

Domain Model Diagram:
https://miro.com/welcomeonboard/RWFGN21IWXM1b2lhamk0THRxWW5QQUtVQU90TDZGUUljNGlZRnN4YWw3QTAyTm9mT01vSmNNZlZjMmN0b3djeHwzNDU4NzY0NTc3OTU3NDcxNzU3fDI=?share_link_id=248068759058

Class Diagram:
https://miro.com/welcomeonboard/RWFGN21IWXM1b2lhamk0THRxWW5QQUtVQU90TDZGUUljNGlZRnN4YWw3QTAyTm9mT01vSmNNZlZjMmN0b3djeHwzNDU4NzY0NTc3OTU3NDcxNzU3fDI=?share_link_id=248068759058

Wireframe (Prototype):
https://www.figma.com/file/xG8jAb4wqIehr7afEY854e/MedSync-user-flow-(Copy)?type=design&node-id=0%3A1&mode=design&t=zhPNNciLmwDAuPjh-1