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 section of their profile.
2. The system presents to the patient the booked appointments (if any) and provides the option to schedule a new appointment.
3. The patient selects the option to schedule a new appointment.
4. The system presents the patient with an interface to provide the referral code of the doctor to book an appointment with, which the patient fills and then selects the retrieved doctor accordingly.
5. The system presents to the patient the selected doctor's available appointment slots. From which the patient selects an available time slot.
6. The system prompts the patient to choose the preferred mode of consultation (physical meeting or video call) and to provide details of why they are scheduling an appointment, and the patient provides both accordingly.
7. The patient confirms the appointment details, and the system confirms the appointment booking success.

Extensions:

4a. The patient does not have a referral code for a doctor on the platform.

1. The system will not display any doctor options on the doctor selection page, essentially disallowing the patient from proceeding.

5a. The doctor has no available time slots.

1. The system will not display any time slots for the selected doctor, essentially disallowing the patient from proceeding.

5b. 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.

6a. The patient does not choose a mode of consultation.

1. The system highlights to the user that they have to choose a mode of consultation before proceeding.
2. Use case continues at step 6.

6b. The patient does not provide the appointment reason.

1. The system highlights to 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 navigates to the medical prescription issuing section of their profile.
2. The system presents to the doctor the patients with which he/she has booked appointments.
3. The doctor selects the patient they want to issue a prescription for.
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:

2a. The doctor does not have any booked appointments with patients on the platform.

1. The system will not display any patient options on the medical prescription issuing 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

3. Use case title: Upload a Medical Record

Main Success Scenario:

1. The doctor navigates to the medical record uploading section of their profile.
2. The system presents to the doctor the patients with which he/she has booked appointments along with the option to upload a medical record for each.
3. The doctor chooses to upload a medical record for the intended patient, provides the record, and confirms their uploading.
4. The system confirms successful upload of the medical record to the patient's profile.

Extensions:

2a. The doctor does not have any booked appointments with patients on the platform.

2. The system will not display any patient options on the medical prescription issuing page.

3a. The doctor attempts to confirm the uploading 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 3.

3b. The doctor attempts to upload 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 3.

3c. 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 3.

4. Use case title: Request Intelligent Diagnostic Support

Main success scenario:

1. The 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.

3a. The doctor provides more symptoms than the system accepts.

1. The system informs the doctor that they have inputted too many symptoms.
2. The system re-prompts the doctor to enter the symptoms to diagnose.

5. Use case title: Initiate Video Call Appointment

Main Success Scenario:

1. The patient navigates to the appointments section of their profile.
2. The system presents to the patient his/her booked appointments, and for the appointments booked for virtual consultation, provides the option to join a video call.
3. The patient chooses to join the intended virtual appointment over a video call.
4. The system initiates the video call session with the selected doctor through ZegoCloud and presents the patient the option to enter the video call.
5. The patient enters the video call, and the consultation is conducted virtually.

Extensions:

2a. 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.

2b. The patient does not have any appointments scheduled for virtual consultation on their profile.

1. The system will not display the option to join a video call for the non-virtual appointments.

3a. The patient attempts to join a video call not scheduled for the current date.
1. The system informs the patient that they cannot join a video call not scheduled for the current date.

4a. The patient decides not to enter the video call.
1. The system allows the patient to navigate back to their profile dashboard.

5a. The doctor is not available at the video call appointment.
1. Through ZegoCloud, the patient can wait for the doctor or exit the video call.

Section 2 : *System Sequence Diagram (SSD)*

1. Book an Appointment:
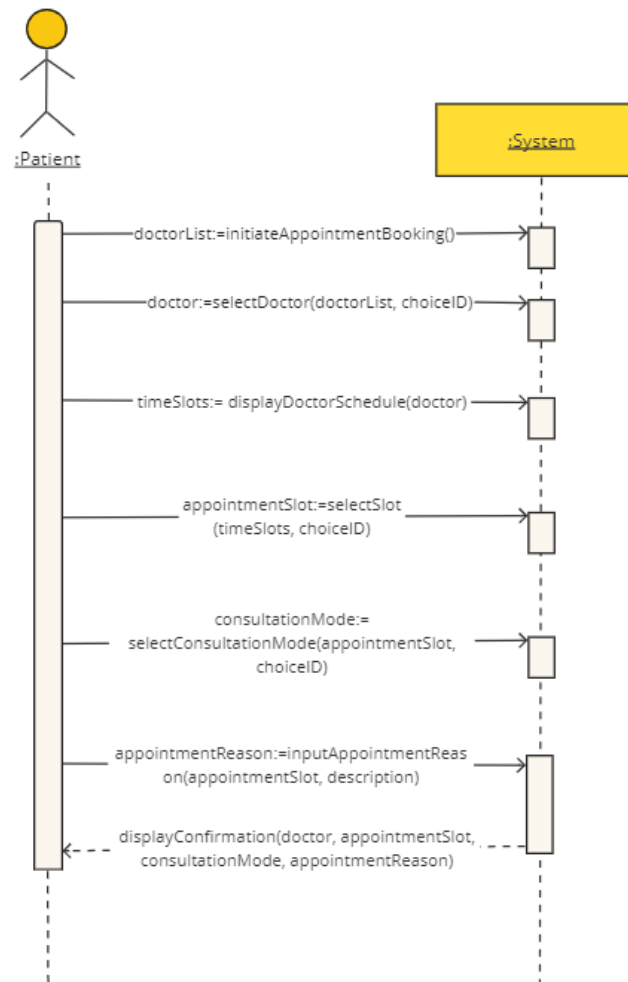
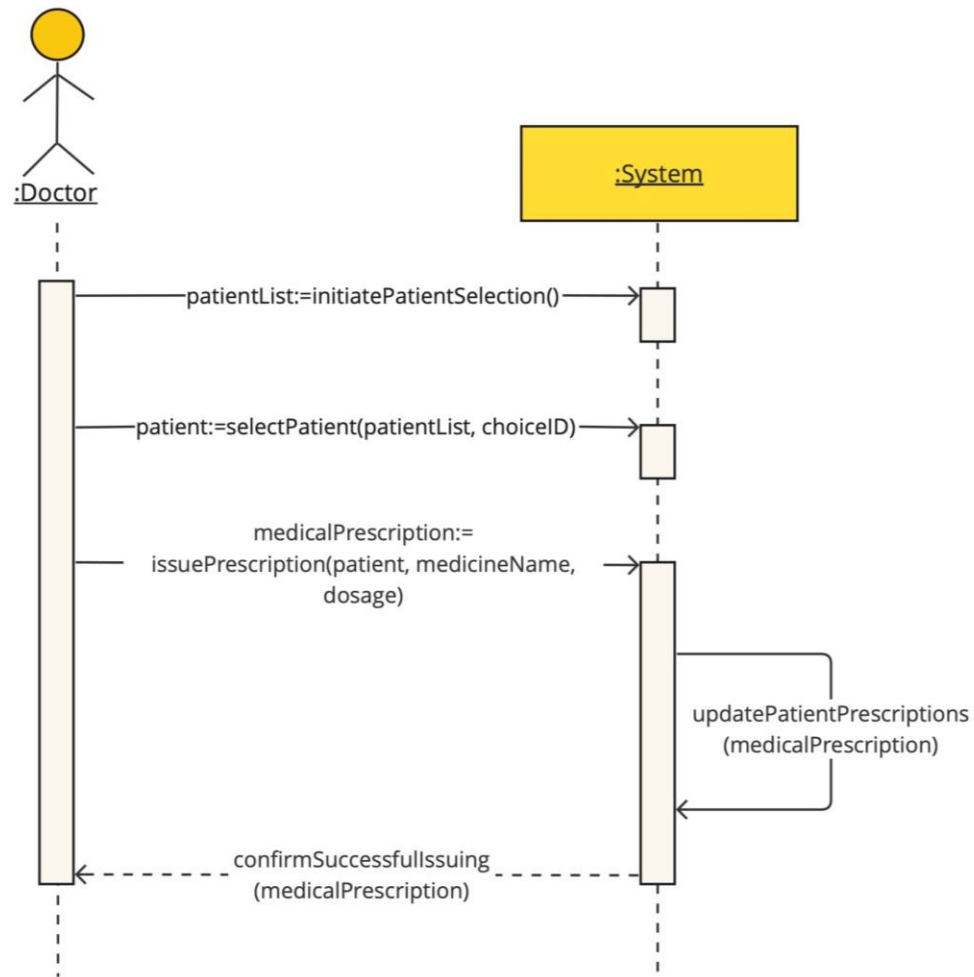*Figure 1: SSD for Booking an Appointment*

2. Upload a New Prescription:

Figure 2: SSD for Uploading a New Prescription
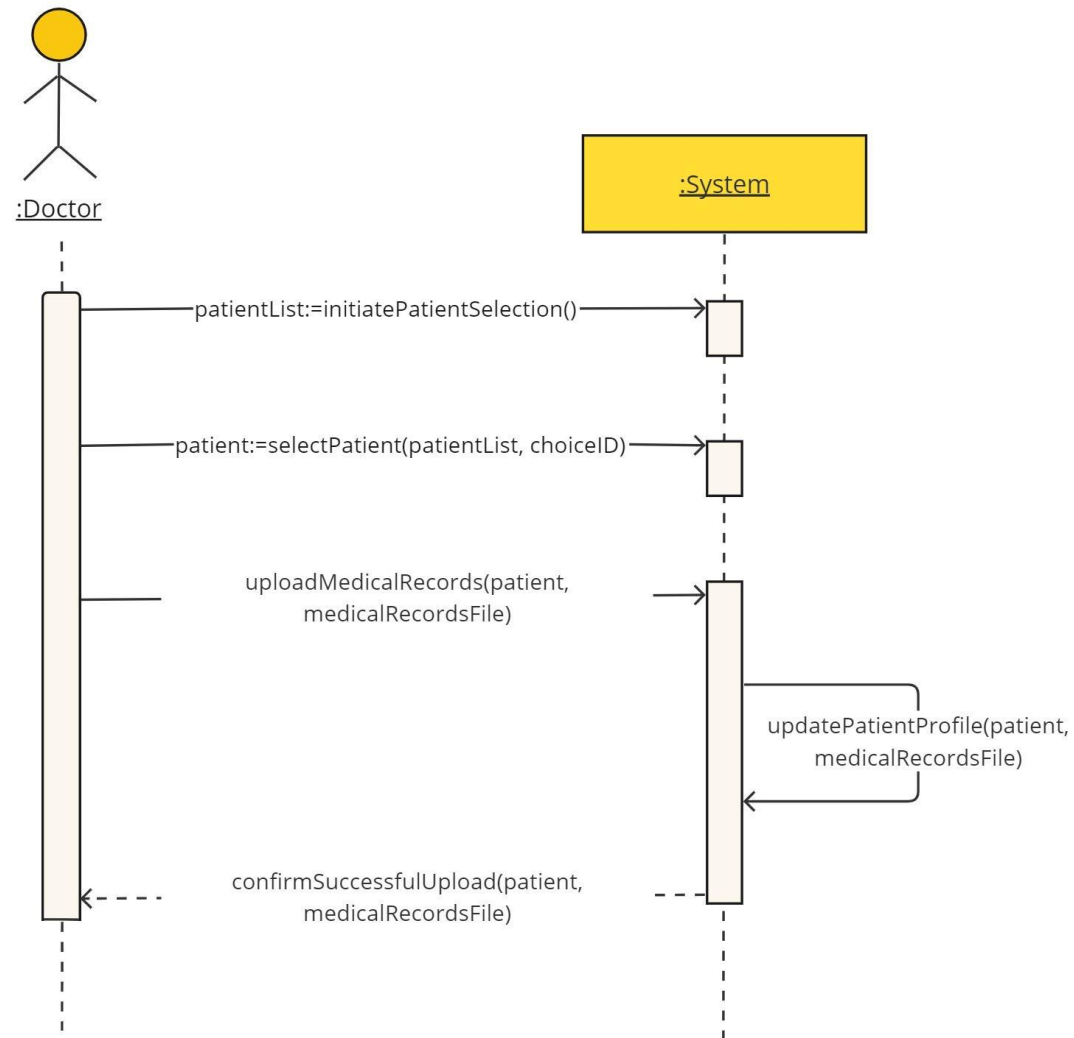
3. Upload a Medical Record:

Figure 3: SSD for Uploading a Medical Record

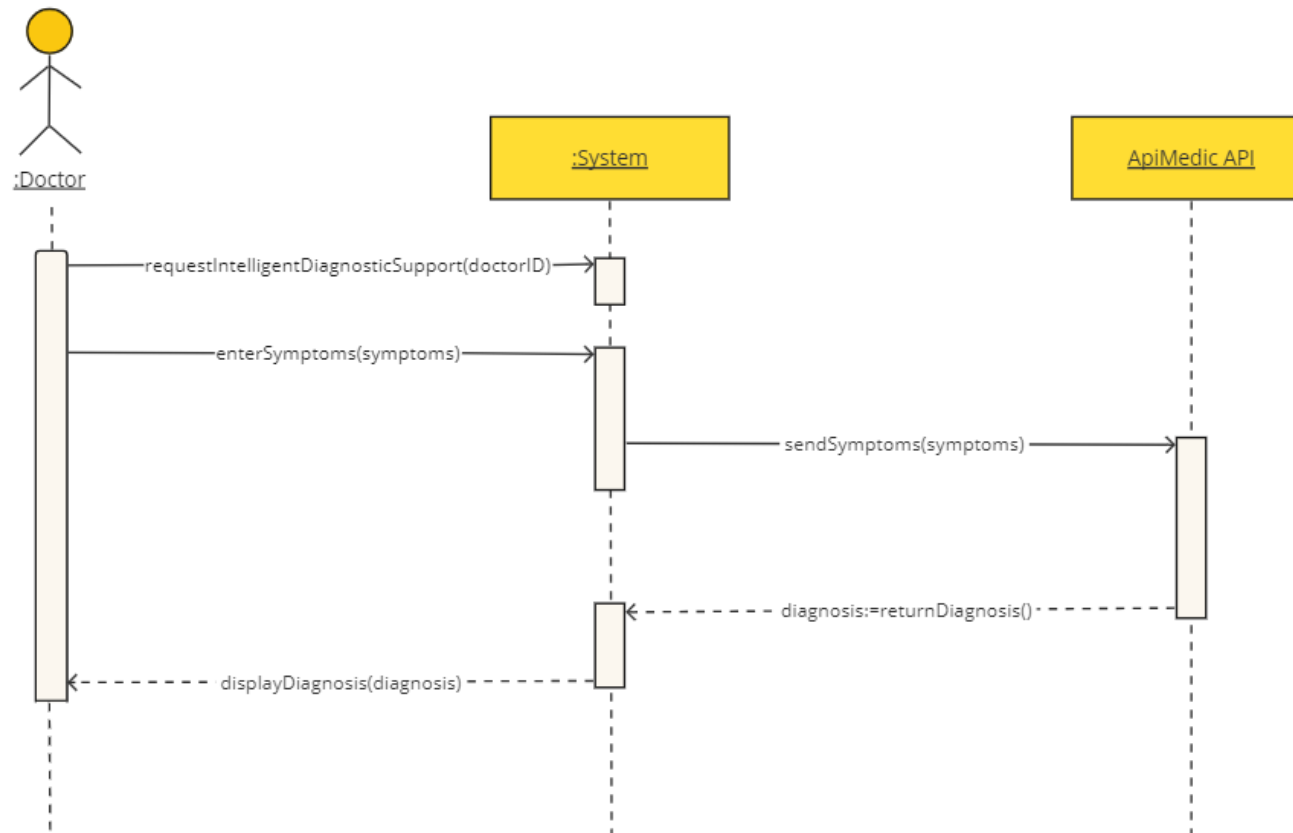**4.** Request Intelligent Diagnostic Support:

Figure 4: SSD for Requesting Intelligent Diagnostic Support
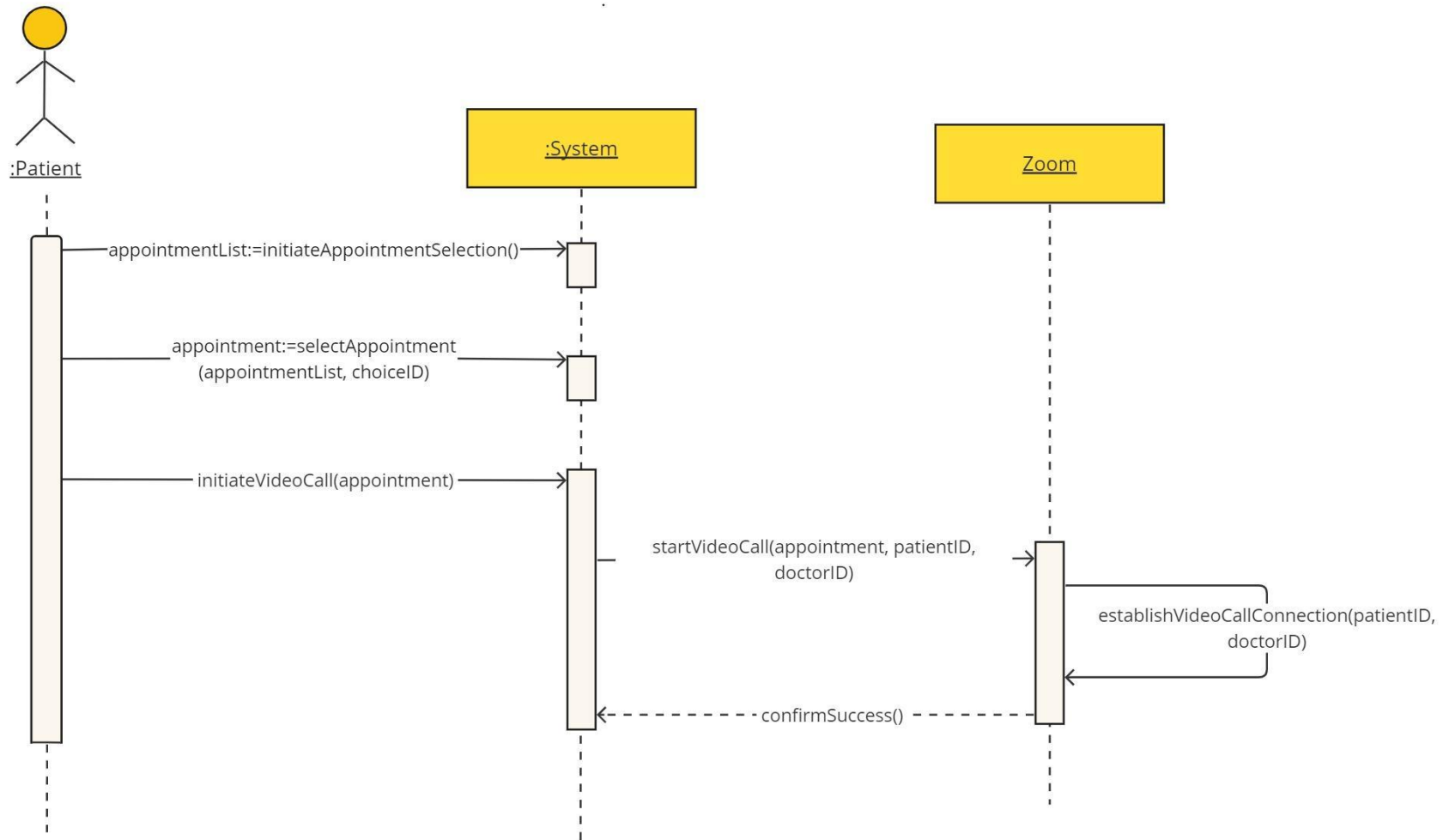
5. Initiate Video Call Appointment:

Figure 5: SSD for Initiating a Video Call Appointment

Section 3: *Domain Model*
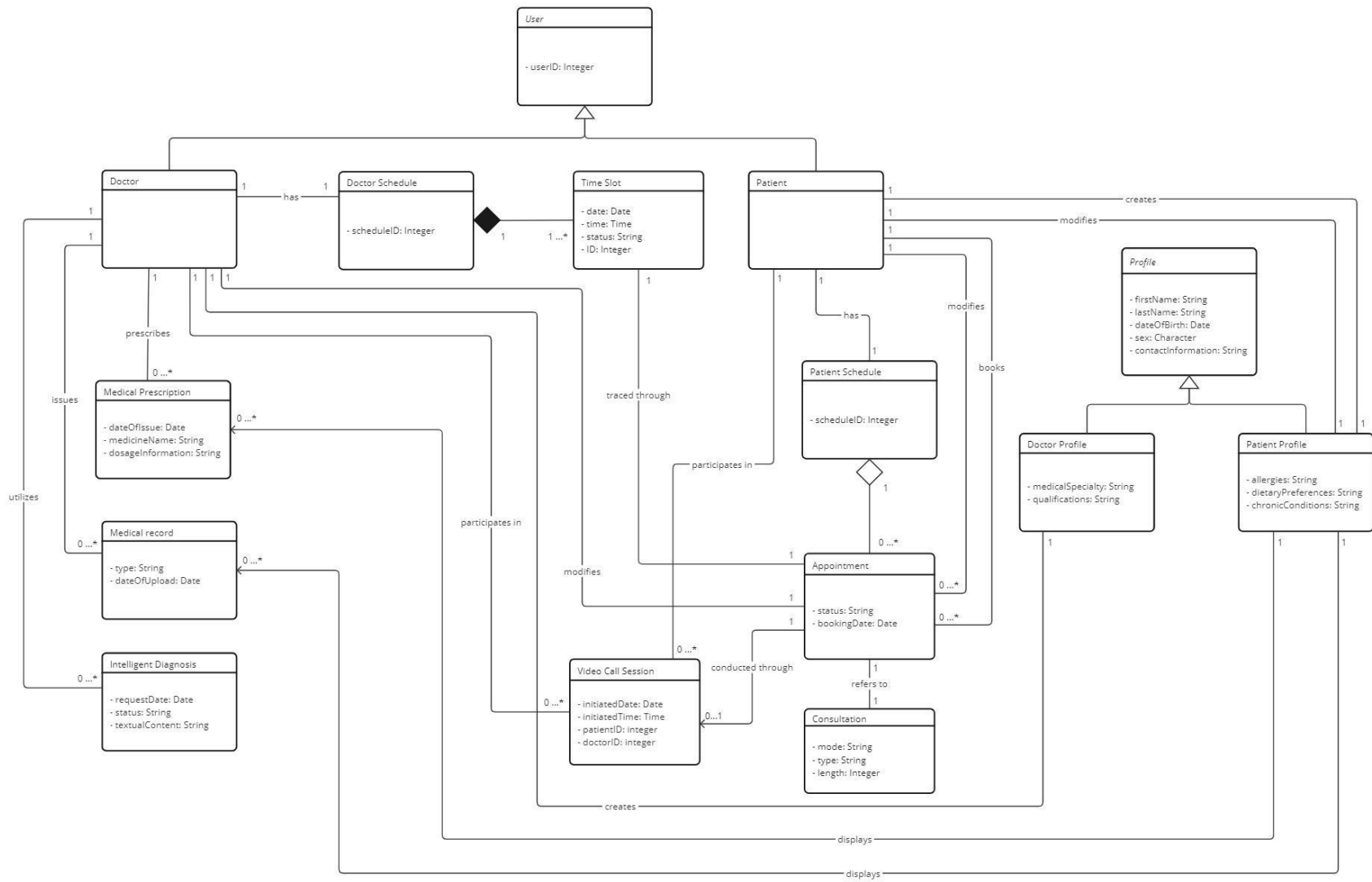
Figure 6: Domain Model Diagram

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. Upload a New Prescription:

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)*

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
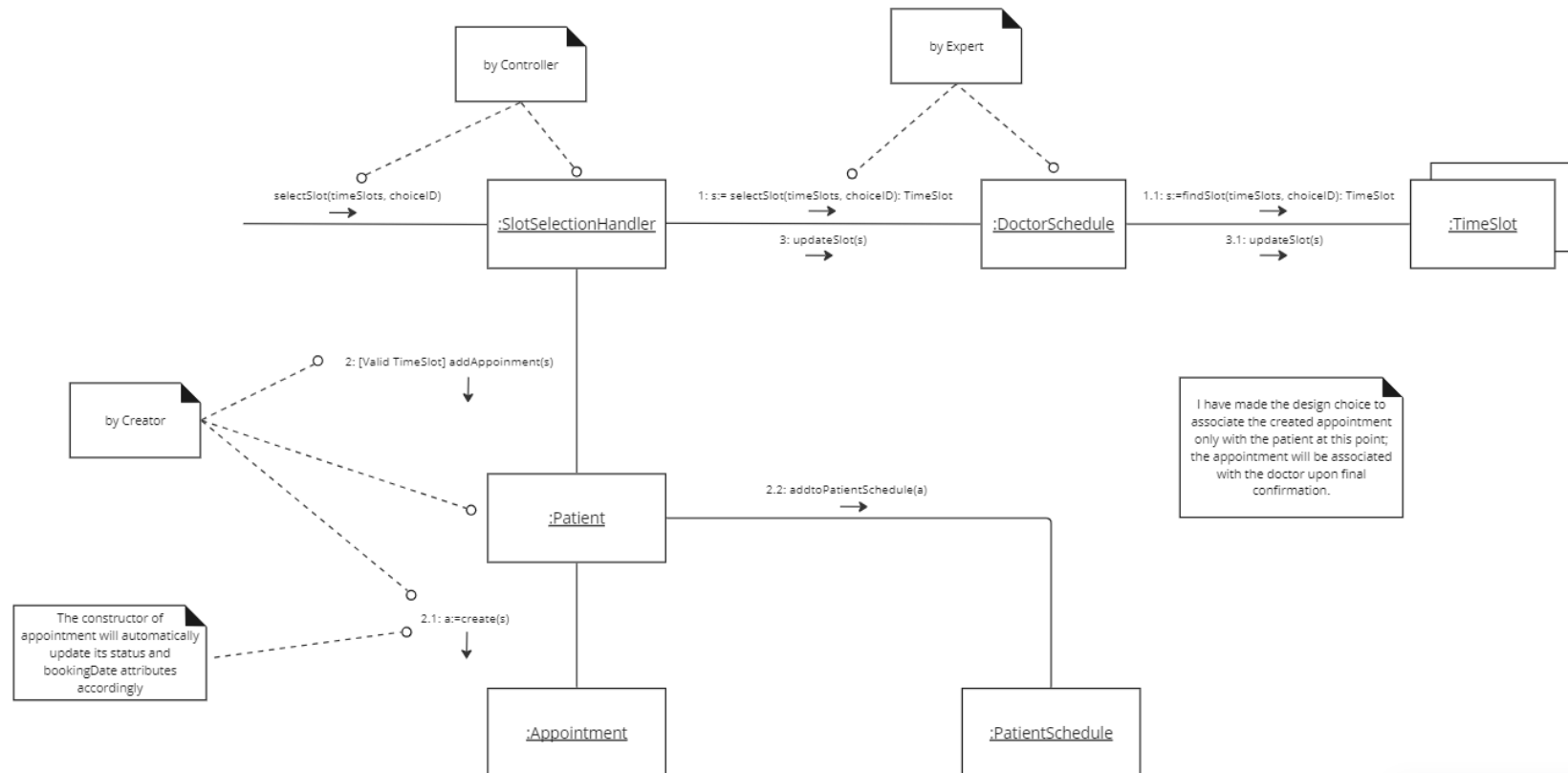
Figure 7: Interaction diagram for Making an Appointment

2. Upload a New Prescription:
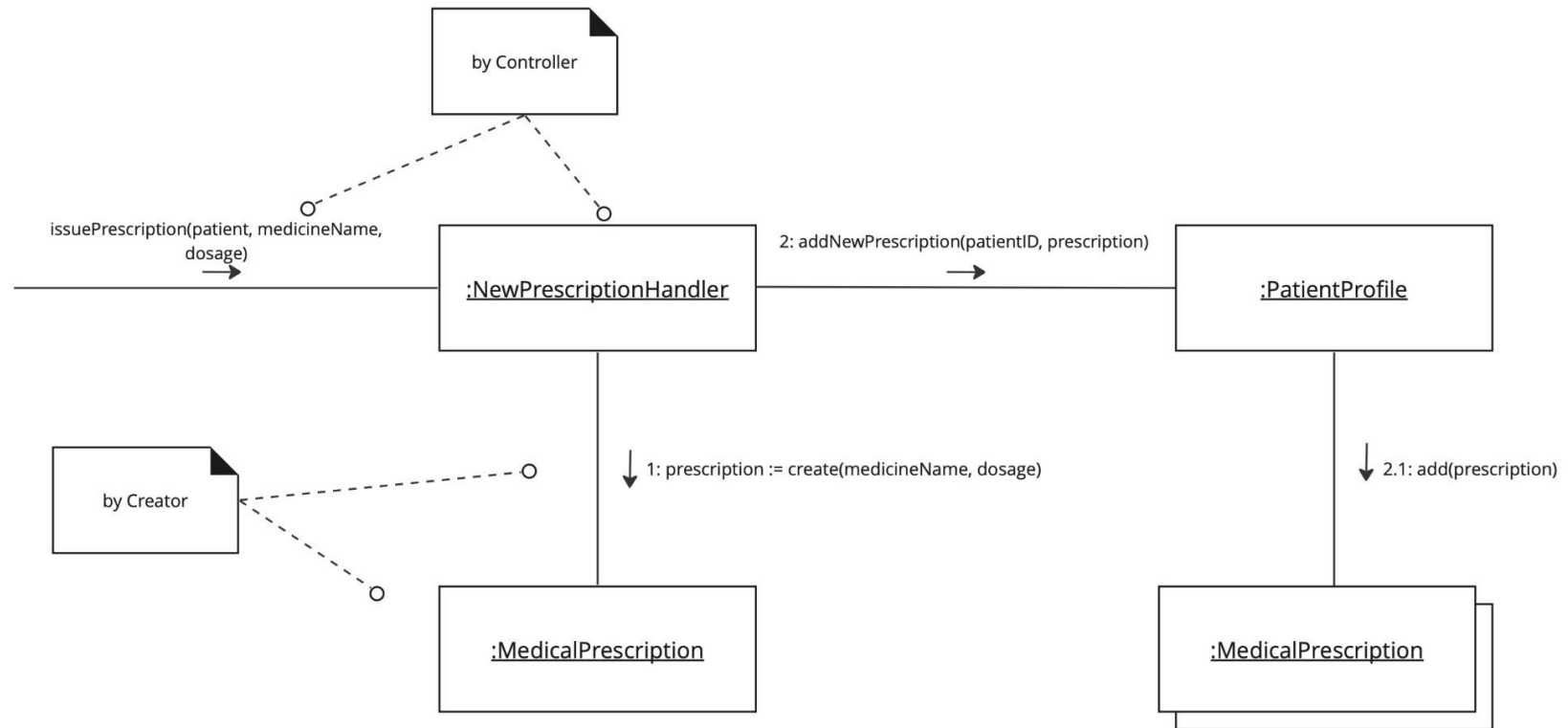
Figure 8: Interaction diagram for Uploading a New Prescription

3. Upload a Medical Record:
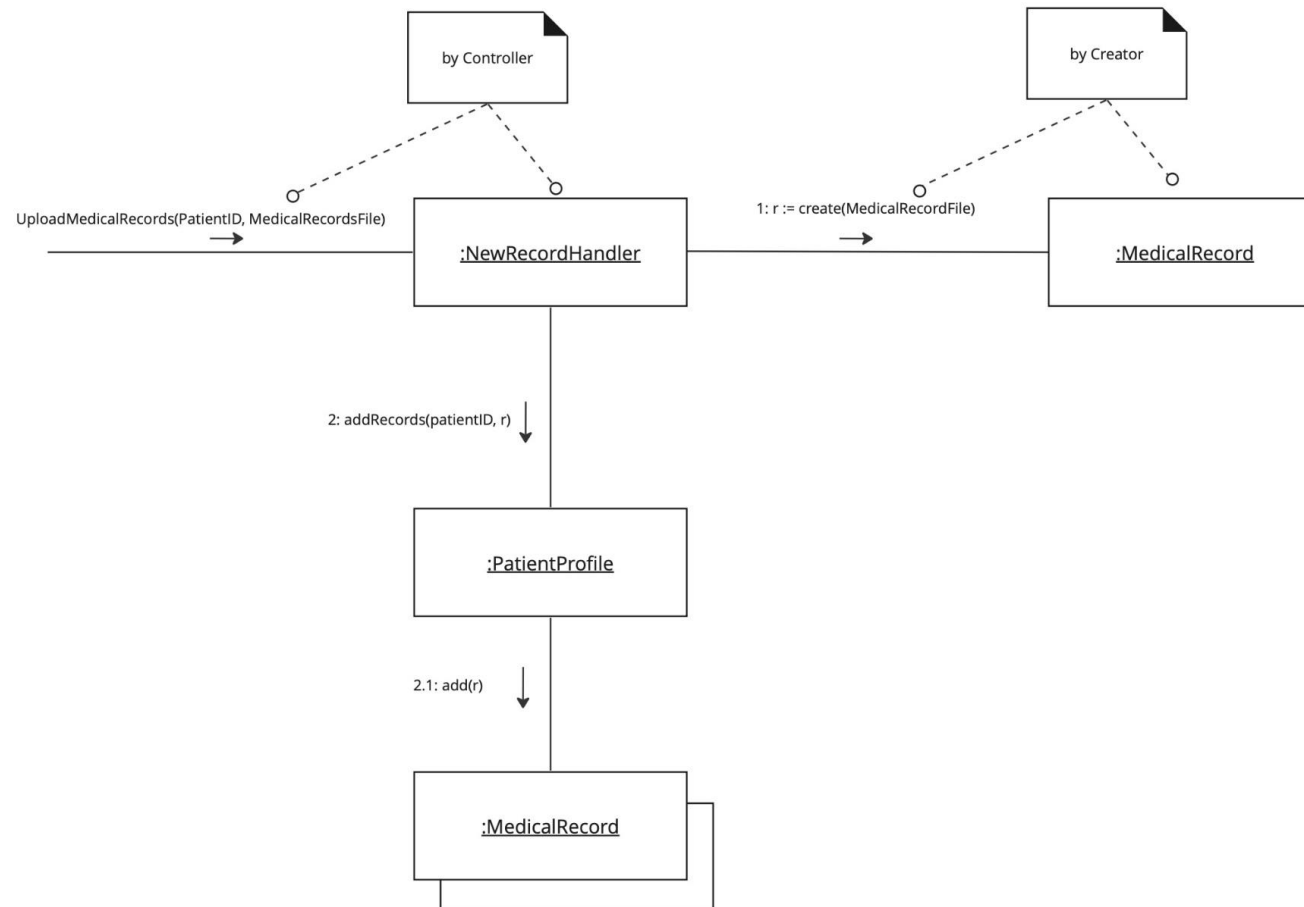
Figure 9: Interaction diagram for Uploading a Medical Record

4. Request Intelligent Diagnostic Support:
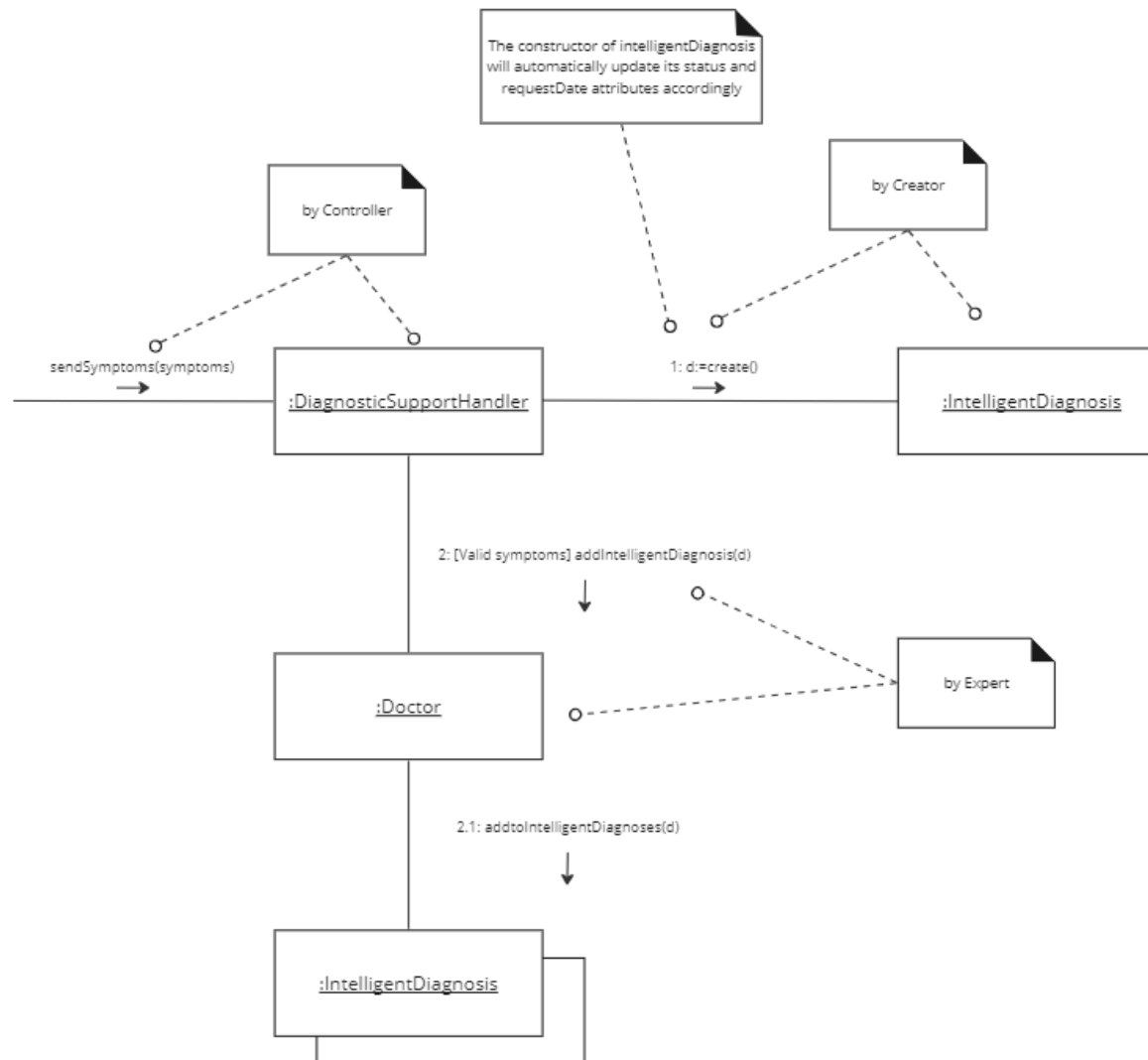
Figure 10: Interaction diagram for Requesting Intelligent Diagnostic Support

5. Initiate Video Call Appointment:
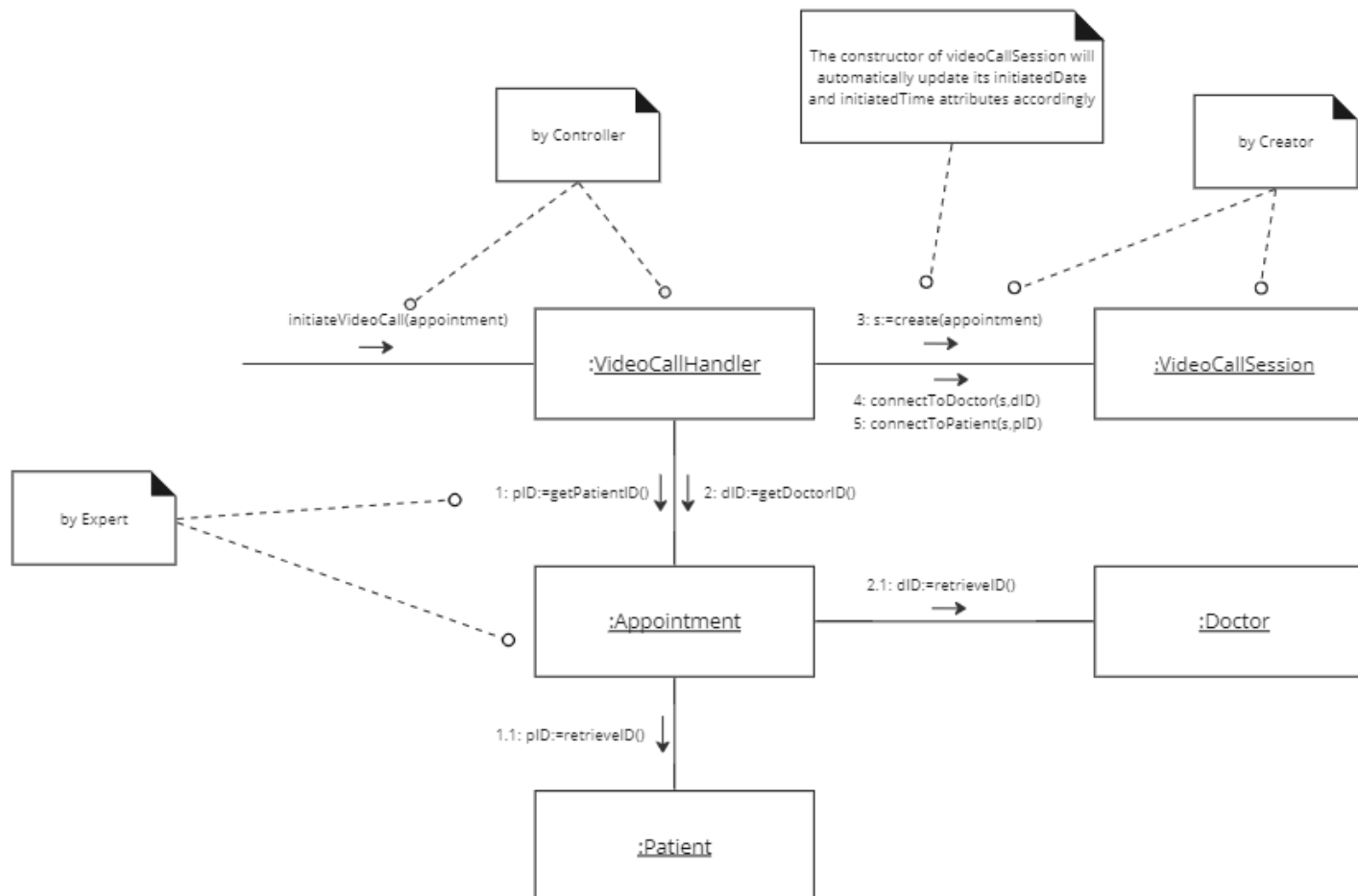
Figure 11: Interaction diagram for Initiating a Video Call Appointment

24
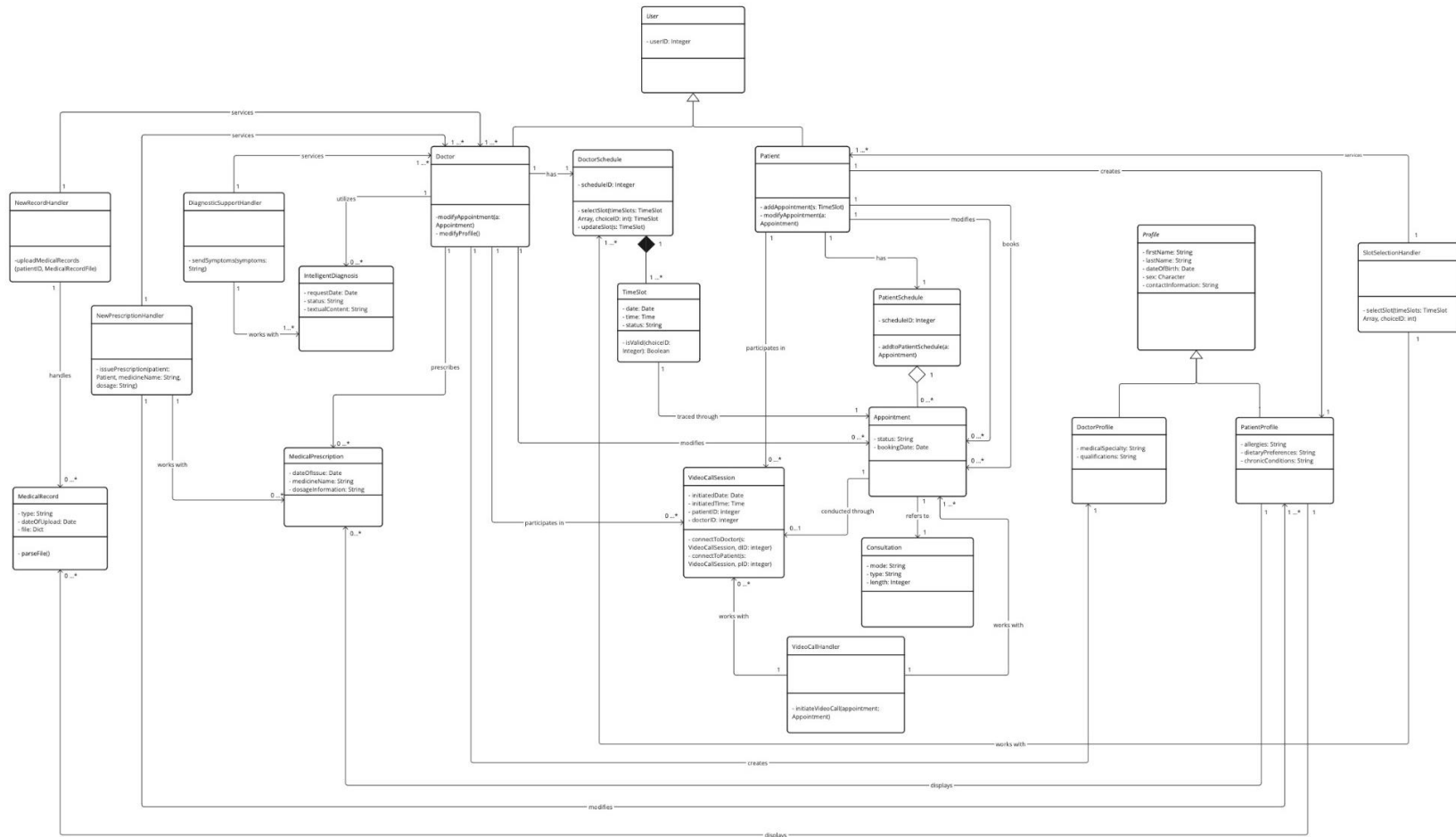
Figure 12: Class Diagram

Section 7: *User Interface (UI)*
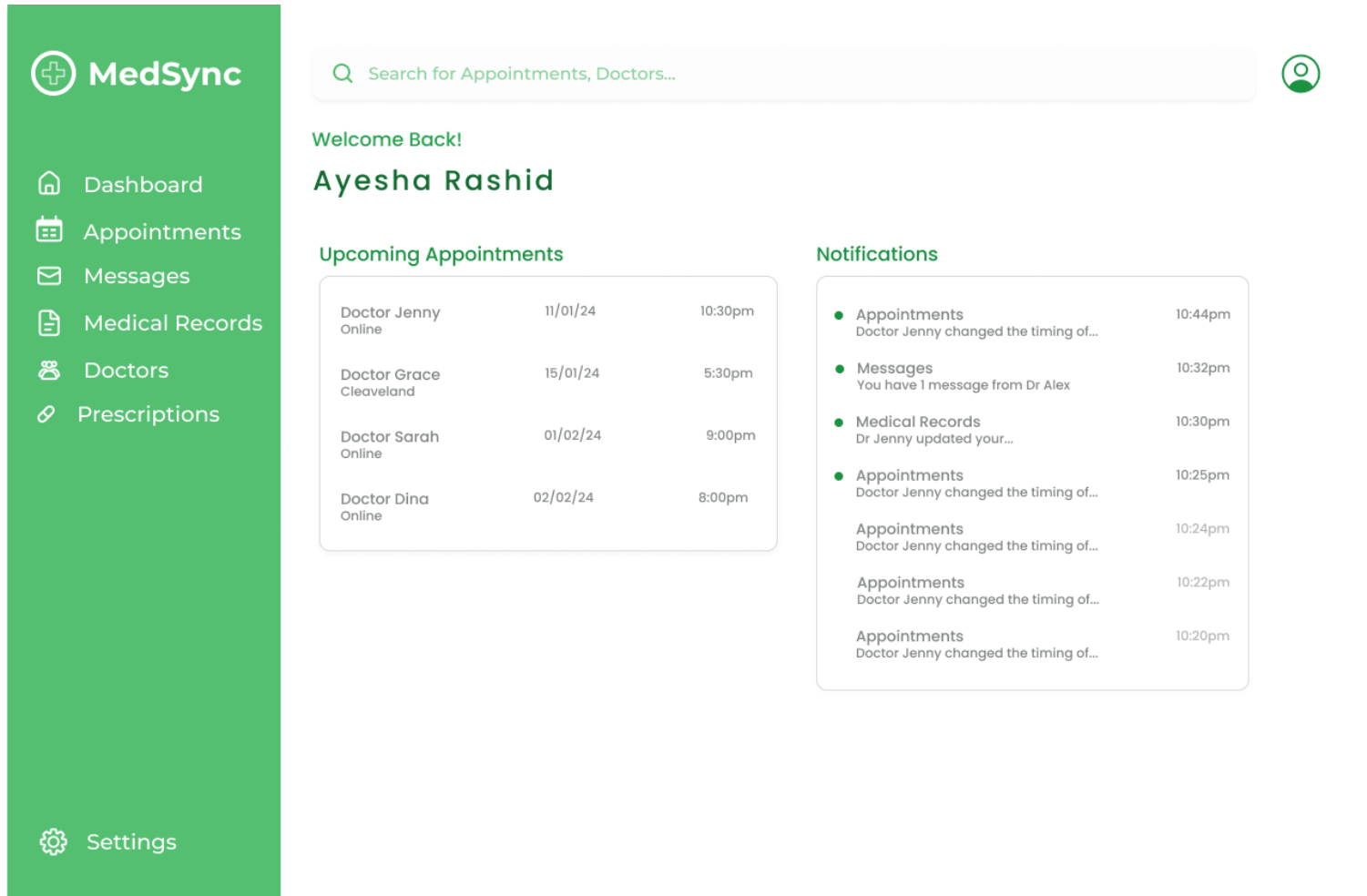
Figure 13: Patient Dashboard
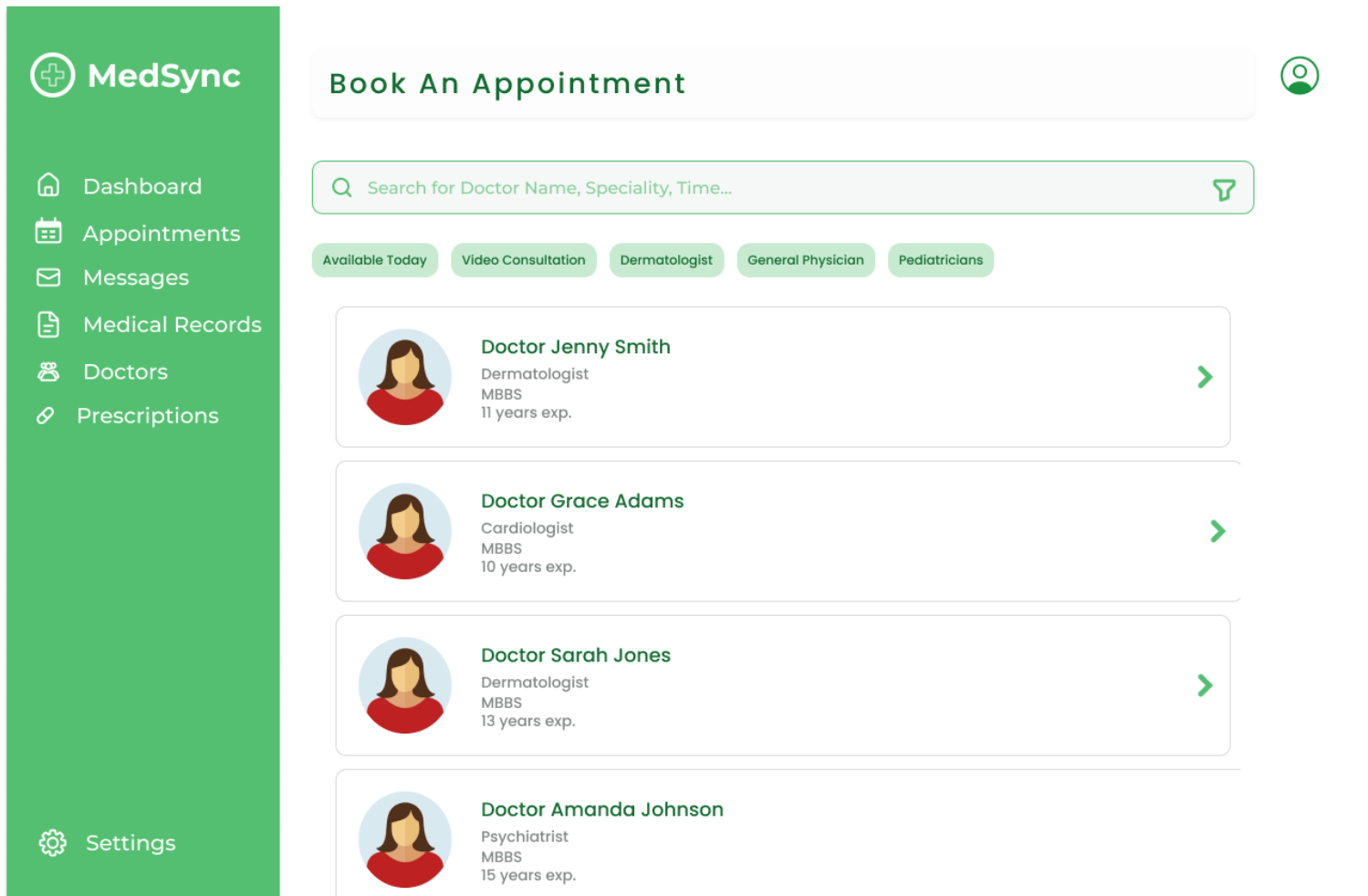
Figure 14: New Appointment Search
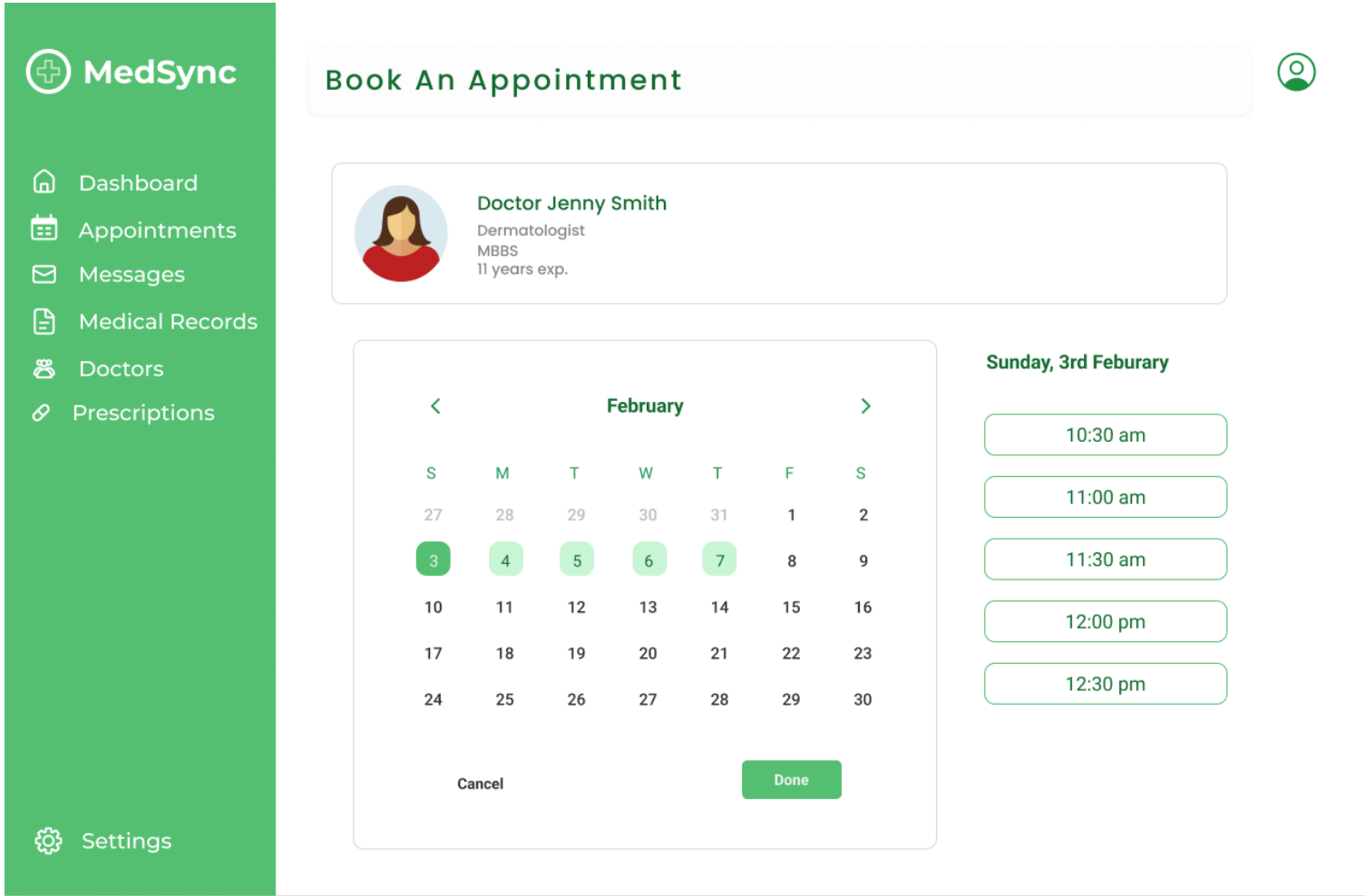
Figure 15: Doctor Availability Schedule

Figure 16: Appointment Details
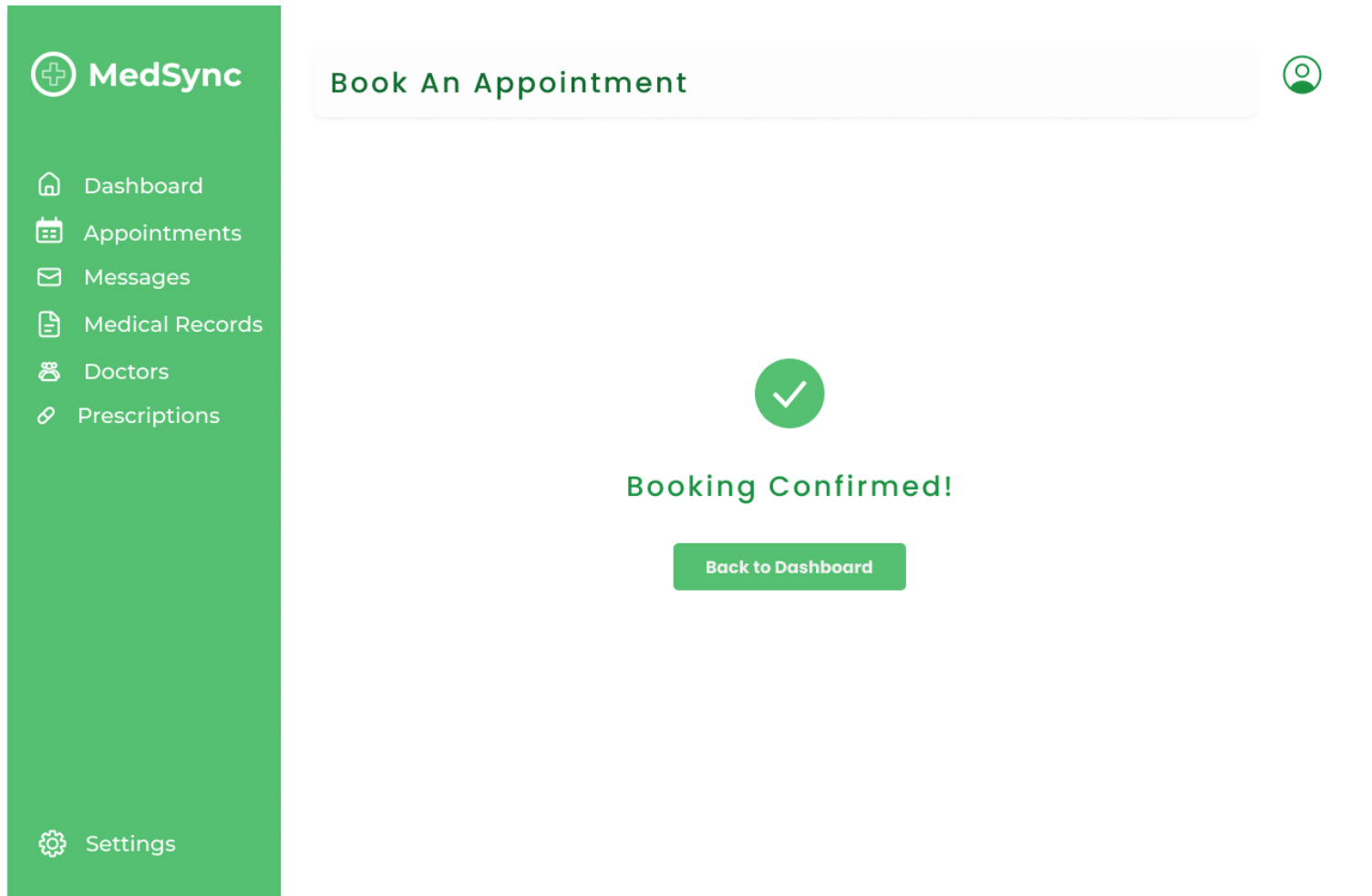
Figure 17: Appointment Confirmation

Section 8: Appendix

Domain Model Diagram:

https://miro.com/welcomeonboard/RWFGN21lWXM1b2lhamk0THRxWW5QQUtVQU90TDZGUUIjNGIZRnN4YWw3QTAy
Tm9mT01vSmNNZlZjMmN0b3djeHwzNDU4NzY0NTc3OTU3NDcxNzU3fDI=?share_link_id=248068759058

Class Diagram:

https://miro.com/welcomeonboard/RWFGN21lWXM1b2lhamk0THRxWW5QQUtVQU90TDZGUUIjNGIZRnN4YWw3QTAy
Tm9mT01vSmNNZlZjMmN0b3djeHwzNDU4NzY0NTc3OTU3NDcxNzU3fDI=?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