UNIVERSITI MALAYSIA SARAWAK

Faculty of Computer Science and Information Technology

Assignment/Report Cover Sheet

Student Name (lecturer group number)	Student Id Number	L. Group	Signature
Holifield William (L.G01)	69861	04	
Muhd Hamarudin bin Omar Ali (L.G01)	70653	04	
Christopher Sii How Chiong (L.G01)	69385	04	
Mohd Zaki bin Ahmad Mustafa (L.G01)	72368	04	
Siti Hamizah binti Zainul Abidin (L.G01)	72676	04	
Ting Lee Ji (L.G01)	72713	04	

Subject Code: **TMF2223 G1**	Subject Name: **Object Oriented Software Development**		
Assignment Title:	**Assignment 1**	Lecturer:	**Dr. Chai Soo See**
Due Date: **27/11/2020, 10:00am**		Date Submitted: **27/11/2020**	

Plagiarism and Collusion are methods of cheating that falls under Peraturan Akademik Universiti Malaysia Sarawak para 11: Etika Akademik

Plagiarism

Plagiarism is the presentation of work which has been copied in whole or in part from another person's work, or from any other source such as the internet, published books or periodicals without due acknowledgement given in the text.

Collusion

Collusion is the presentation of work that is the result in whole or in part of unauthorized collaboration with another person or persons.

Where there are reasonable grounds for believing that cheating has occurred, the only action that may be taken when plagiarism or collusion is detected is for the staff member not to mark the item of work and to report or refer the matter to the Dean. This may result in work being disallowed and given a fail grade or if the circumstances warrant, the matter may be referred to a Committee of inquiry for investigation. Such investigation may result in the matter being referred to the University Discipline Committee, **which** has the power to exclude a student.

Upon placing signature above, I certify that I have not plagiarized the work of others or participated in unauthorized collusion when preparing this assignment.

I also certify that I have taken proper case in safeguarding my work and have made all reasonable efforts to ensure that may work not be able to be copied.

MARK :

Table of Contents

1.0 Question 1

Using Visual Paradigm, develop a **use case diagram** with **use cases** (6 to 10 use cases) to reflect the requirements of the given case study. Noted that each use case must couple with correspondent **use case description**.

1.1 Use case diagram

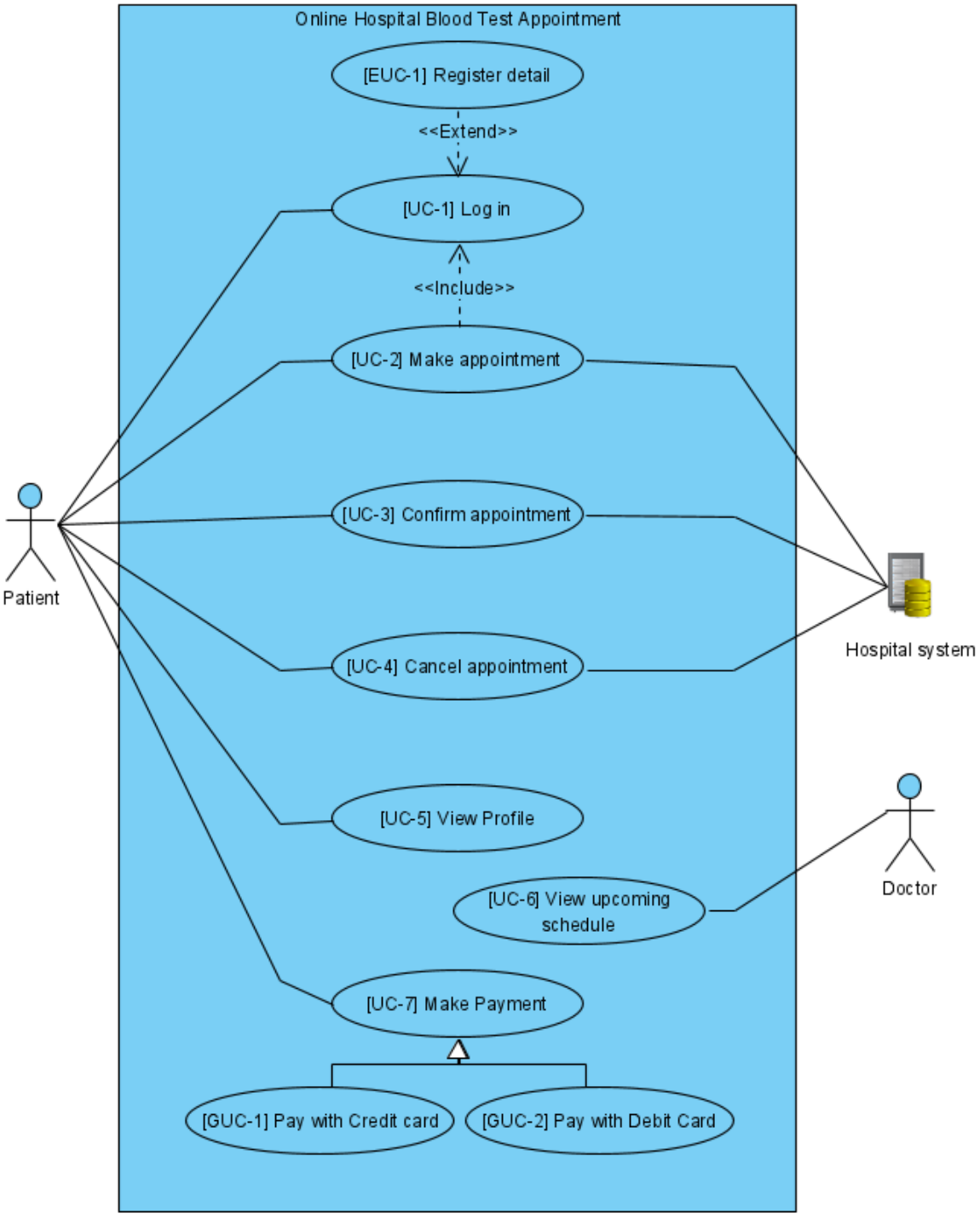

Figure 1. Use case diagram.

1.2 Use case specification

1.2.1 Use case specification for EUC-1 Register detail.

Table 1. Use case specification for EUC-1 Register detail.

Use Case ID:	EUC-1
Use Case Name	Register detail
Short Description:	Registers an account for patient which will enable selection slot of appointment.
Trigger Action:	Patient creates a new account
Actors:	Patient
Requirements:	R1.1; R1.2; R1.3; R2.1; R2.2; R2.3
Pre-Conditions:	1. Patient accesses to the "*Register*" page
Post-Conditions:	1. Patient able to access his/her account

Main Flow:

1. Patient clicks "*Register Now*" button.
2. System display registration form.
3. Patient inputs required information (name) [E1]
4. Patient inputs required information (date of birth) [E1]
5. Patient inputs required information (IC number) [E1]
6. Patient inputs required information (address) [E1]
7. Patient inputs required information (E-mail) [E1]
8. Patient inputs required information (mobile num.) [E1]
9. Patient uploads photo of identity card. [E1]
10. Patient creates username, and password. [E1]
11. Patient clicks "*Submit*" button. [A1] [E2]
12. System displays registration successful message.
13. System adds new patient account into database.
14. Patient accesses to the relevant homepage.

Alternate Flows:

(A1.) Patient cancels registration.
Post-condition→ no account was created.

Exception Flows:

(E1.) Patient inputs any invalid or empty personal data or document.
System displays "*Invalid data entered, please try again*."
Post condition → Use Case resume at main flow (2).

(E2.) Account already exists.
Post-condition → no account was created

1.2.2 Use case specification for UC-1 Log in.

Table 2. Use case specification for UC-1 Log in.

Use Case ID:	UC-1
Use Case Name	Log in
Short Description:	Allows Patient to login into a his/her account to manage personal appointment.
Trigger Action:	Patient creates a new account
Actors:	Patient
Requirements:	R1.1; R1.3; R1.4; R2.2
Pre-Conditions:	1. Patient accesses to the "Log in" page
Post-Conditions:	1. Patient accesses to the relevant homepage.

Main Flow:

1. Patient clicks "Log in" button.
2. Patient inputs username. [A1]
3. Patient inputs password.
4. Patient clicks "Submit" button. [A2]
5. System verify username and password. [E1]
6. System displays a successful message.
7. System displays relevant homepage.

Alternate Flows:

(A1.) Patient has no account.
Post-condition→ patient accesses "*Register Now*" page.

(A2.) Patient forgot username or password.
Post-condition→ patient accesses "*forgot username/password*" page.

Exception Flows:

(E1.) Patient enters invalid username or password.
Post-condition→ system displays error message, and the Use Case resume at main flow (2).

1.2.3 Use case specification for UC-2 Make an appointment.

Table 3. Use case specification for UC-2 Make an appointment.

Use Case ID:	UC-2
Use Case Name	Make an appointment
Short Description:	The patient will use this function to select their appointment slot. Once logged on to patient account, patient will be able to access "M*ake an appointment*" page where patient can select available slot for appointment allocated which is one hour apart. There is a limit of 15 appointments for each slot.
Trigger Action:	Patient makes an appointment
Actors:	Patient
Requirements:	R1.7; R2.2; R2.3
Pre-Conditions:	1. Patient successfully logged in. 2. Patient accesses to the "M*ake an appointment*" page.
Post-Conditions:	1. Patient appointment successfully created.
Main Flow:	
1. Patient clicks "*Log in*" to access patient account. 2. Patient fills in username and password. 3. System checks username and password validity. 4. System displays the relevant homepage. 5. Patient clicks "*Make an appointment*" button. 6. System retrieves available slot. 7. Patient selects an appointment slot. 8. System displays message "*Only 15 appointments for each slot. Every appointment is one hour apart."* 9. System displays available times for selected slot. (E1) 10. Patient selects an available time. 11. Patient clicks "*Submit*" button. (A1) 12. System records selected time into the patient database. 13. System displays "*Your appointment successfully created. Please confirm the appointment at least 3 days or at most 3 months in advance."*	
Alternate Flows:	
(A1.) Discard appointment Post-Conditions→ No appointment was made	
Exception Flows:	
(E1.) All 15 appointments for selected slot are fully booked. System displays "*All available time has been booked. Select another slot.*" Post-Condition→ Use case resumes at main flow 7.	

1.2.4 Use case specification for UC-3 Confirm appointment.

Table 4. Use case specification for UC-3 Confirm appointment.

Use Case ID:	UC-3
Use Case Name	Confirm appointment
Short Description:	The patient will use this function to confirm a booked appointment. A verification email confirming the appointment will be send to the patient once "*confirm*" button clicked in the system.
Trigger Action:	Patient confirms created appointment.
Actors:	Patient
Requirements:	R.1.8; R2.2; R2.3
Pre-Conditions:	1. The patient has successfully logged in. 2. The patent has successfully made an appointment.
Post-Conditions:	1. The confirmation appointment was recorded by the system.

Main Flow:

1. Patient clicks "*Log in*" to access patient account.
2. Patient fills in username and password.
3. System checks username and password validity.
4. System displays the relevant homepage.
5. Patient clicks "*Confirmation appointment*" button to access list of appointments booked by patient.
6. Patient clicks "*Confirm*" button right next to the selected appointment list. [A1]
7. System sends a verify mail to patient.
8. System displays "*A verification mail has been sent to your E-mail.*"
9. The patient received an email notification.
10. The patient clicks the provided link to confirm the appointment.
11. The user clicks the button "*confirm".* [E1]
12. The confirmation was recorded by the system

Alternate Flows:

(A1.) Cancel booked.
Post-Conditions→ No confirmation appointment was made.

Exception Flows:

(E1.) exceed the confirmation period given by the system.
Post-Conditions→ Reject application.

1.2.5 Use case specification for UC-4 Cancel appointment.

Table 5. Use case specification for UC-4 Cancel appointment.

Use Case ID:	UC-4
Use Case Name	Cancel appointment
Short Description:	The patient will use this function to cancel a booked appointment. The patient can cancel the successful appointment 1 day before the actual date. Reason for cancellation should be given. Once a cancellation is successful, a free slot will be available.
Trigger Action:	Patient cancels created appointment.
Actors:	Patient
Requirements:	R1.9; R2.2; R2.3
Pre-Conditions:	1. The patient has successfully logged in. 2. The patent has successfully made an appointment.
Post-Conditions:	1. System frees the cancellation slot to available slot.

Main Flow:

1. Patient clicks "*Log in*" to access patient account.
2. Patient fills in username and password.
3. System checks username and password validity.
4. System displays the relevant homepage.
5. Patient clicks "*Cancel Appointment*" button to access list of appointments booked by patient.
6. Patient clicks "*Cancel*" button right next to the selected appointment list. [A1]
7. Patient clicks "*Submit*" button. [E1]
8. System updates the selected slot to available slot.

Alternate Flows:

(A1.) Patient clicks "Add reason of cancellation". Patient fills in reasons of cancellation.
Post-Conditions→ System records reason of cancellation

Exception Flows:

(E1.) exceed the cancellation period given by the system.
Post-Conditions→ Reject application.

1.2.6 Use case specification for UC-5 View profile.

Table 6. Use case specification for UC-5 View profile.

Use Case ID:	UC-5
Use Case Name	View profile
Short Description:	The patient will use this function to view account information. Once logged on to patient account, they will be able to access "*View profile*" page where patient can view their personal detail and appointment history.
Trigger Action:	Patient views personal detail and appointment history.
Actors:	Patient
Requirements:	R1.5; R1.6; R2.2; R2.3
Pre-Conditions:	1. The patient has successfully logged in
Post-Conditions:	1. The patient views personal details and appointment history.

Main Flow:

1. Patient clicks "*Log in*" to access patient account.
2. Patient fills in username and password.
3. System checks username and password validity.
4. System displays the relevant homepage.
5. Patient clicks "*View Profile*" button. [A1]
6. System retrieves personal details and appointment history. [E1]
7. System displays personal details and appointment history

Alternate Flows:

(A1.) Patient clicks "*Edit detail*" button.
Post-Condition→ Patient modifies personal detail.

Exception Flows:

(E1.) System database error.
Post-Condition→ No data retrieve to be view.

1.2.7 Use case specification for UC-6 View schedule.

Table 7. Use case specification for UC-6 View upcoming schedule.

Use Case ID:	UC-6
Use Case Name	View schedule
Short Description:	The doctor will use this function to view upcoming appointment schedules information. Once logged on to system with provided account, they will be able to access "*View upcoming appointment schedule*" page where doctor can view their patients' schedule.
Trigger Action:	Doctor views upcoming appointment schedule.
Actors:	Doctor
Requirements:	R2.3
Pre-Conditions:	1. The Doctor has successfully logged in with provided account.
Post-Conditions:	1. The Doctor views upcoming appointment schedule.
Main Flow:	
1. Patient clicks "*Log in*" to access patient account. 2. Patient fills in username and password. 3. System checks username and password validity. 4. System displays the relevant homepage. 5. Patient clicks "*Billing*". 6. System retrieves outstanding bills. 7. System offers option to pay either credit card or debit card. 8. Patients selects either one payment method. 9. Patients clicks "*Submit*" button. [A1] 10. System displays card credential forms. 11. Patient fills in the credential of the card. 12. Patient clicks "*Submit*" button. [E1] [E2] 13. System receives payment. 14. System generates receipt. 15. Patient receives receipt.	
Alternate Flows:	
A1. Doctor clicks "*View patient profile*" Post-Condition→ The doctor views patients' profile	
Exception Flows:	
E1. System database error Post-Condition→ No data retrieve to be view.	

1.2.8 Use case specification for UC-7 Make payment.

Table 8. Use case specification for UC-7 Make payment.

Use Case ID:	UC-7
Use Case Name	Make payment
Short Description:	The patient will use this function to make payment. Once logged on to patient account, patient will be able to access "Billing" page where patient can make payment via online using credit or debit card.
Trigger Action:	Patient pays outstanding bills.
Actors:	Patient
Requirements:	R1.10; R2.2; R2.3
Pre-Conditions:	1. The patient has successfully logged in. 2. The appointment has been confirmed.
Post-Conditions:	2. The patient receives receipt

Main Flow:

1. Patient clicks "Log in" to access patient account.
2. Patient fills in username and password.
3. System checks username and password validity.
4. System displays the relevant homepage.
5. Patient clicks "Billing".
6. System retrieves outstanding bills.
7. System offers option to pay either credit card or debit card.
8. Patients selects either one payment method.
9. Patients clicks "Submit" button. [A1]
10. System displays card credential forms.
11. Patient fills in the credential of the card.
12. Patient clicks "Submit" button. [E1] [E2]
13. System receives payment.
14. System generates receipt.
15. Patient receives receipt.

Alternate Flows:

A1. The user cancels the transaction.
Post-Condition→ No payment was made.

Exception Flows:

E1. In sufficient fund to pay for appointment fee. The user is informed and the Use Case Terminates.
Post-Condition→ No payment was made.

E2. The payment unsuccessful.
Post-Condition→ No payment was made.

1.3 Requirements.

1.3.1 Performance requirements:

R1.1. All the operations carried out in the system must respond within 2 seconds.

R1.2. The user should be able to register and manage his appointments online at any time.

R1.3. Sufficient disk space and RAM for system minimum requirement

R1.4. The user should be able to login into the system any time.

R1.5. The user can view his profile info at any time.

R1.6. The user can change profile info at any time.

R1.7. The user shall be able to choose the available time allocated for each slot.

R1.8. The user can receive an email of confirmation of appointment.

R1.9. The user can cancel the successful appointment 1 day before the actual date.

R1.10. The user should be able to pay his bill through the system.

1.3.2 Security requirements.

R2.1. The password should be at least 8 characters, 1 Upper case, 1 lower case and 1 number.

R2.2. Actions which cannot be undone should ask for confirmation.

R2.3. Database must store all the information efficiently without any information loss.

Security requirements.

To ensure Better Security, the code behind our software should use a few Object Oriented Programming (OOP) fundamentals and protocols to better protect data. Aside that, software security is the term largely associated with the process of producing reliable, stable, bug and vulnerable free software. Security is fast becoming one of the most important quality attribute in Software Engineering. Critical information and transactions need to be safeguarded against malicious users and attackers. Copyright protected Commercial Off-the Shelf Components (COTS) may be used in software development.

Encapsulation

Encapsulation is defined as binding together data and the functions that manipulate them, keeps data and code hidden from external interferences. Encapsulation hides private data members and implementation details of a class. Public users can only interact with the system, logging in, making appointments and keying in required data for blood donations such as blood type. This data can only be viewed by internal and trusted actors in the system that is the only people that require it. Public users have no way of viewing data of other users when interacting with the interface. Encapsulated objects declared as private or protected acts a safe box, protecting the objects data and access to data is only allowed by calling class member methods that dictates how data can be accessed or changed. As a result, data is safe as it can be only accessed in a safe and in a manner that is known beforehand. In the same way a bank account is secure because it is accessed in a safe manner.

Access Control List

A table that tells a computer's operating system which user has the right to access a certain system object, individual file or file directory. Each object is given a security attribute that identifies its access control list. The list contains entries on system user privileges such as to read a file or all files in the directory, write on a file, and execute a file. So, Access Control List secures the systems discretion.

MD5 Hash Password Encryption

The MD5 message-digest algorithm, a widely used hash function designed for cryptographic hash function. MD5 digests have been widely used in software to provide assurance that a transferred file has arrived intact. MD5 was used historically to store a one-way hash of a password. Besides that, it is also used as a unique identifier for each document that is exchanged in the legal discovery process. Though the MD5 has been reported to have extensive vulnerabilities, it is useful for a checksum to verify data integrity. Despite this, MD5 is still widely used as of 2019.

2.0 Question 2

Based on the use cases you provided in Question 1, draw **TWO (2) system sequence diagrams (SSD)**. For the other use cases you did not draw SSD, draw **FOUR (4) sequence diagrams (SD)**.

2.1 Sequence diagram.

2.1.1 Register detail sequence diagram.

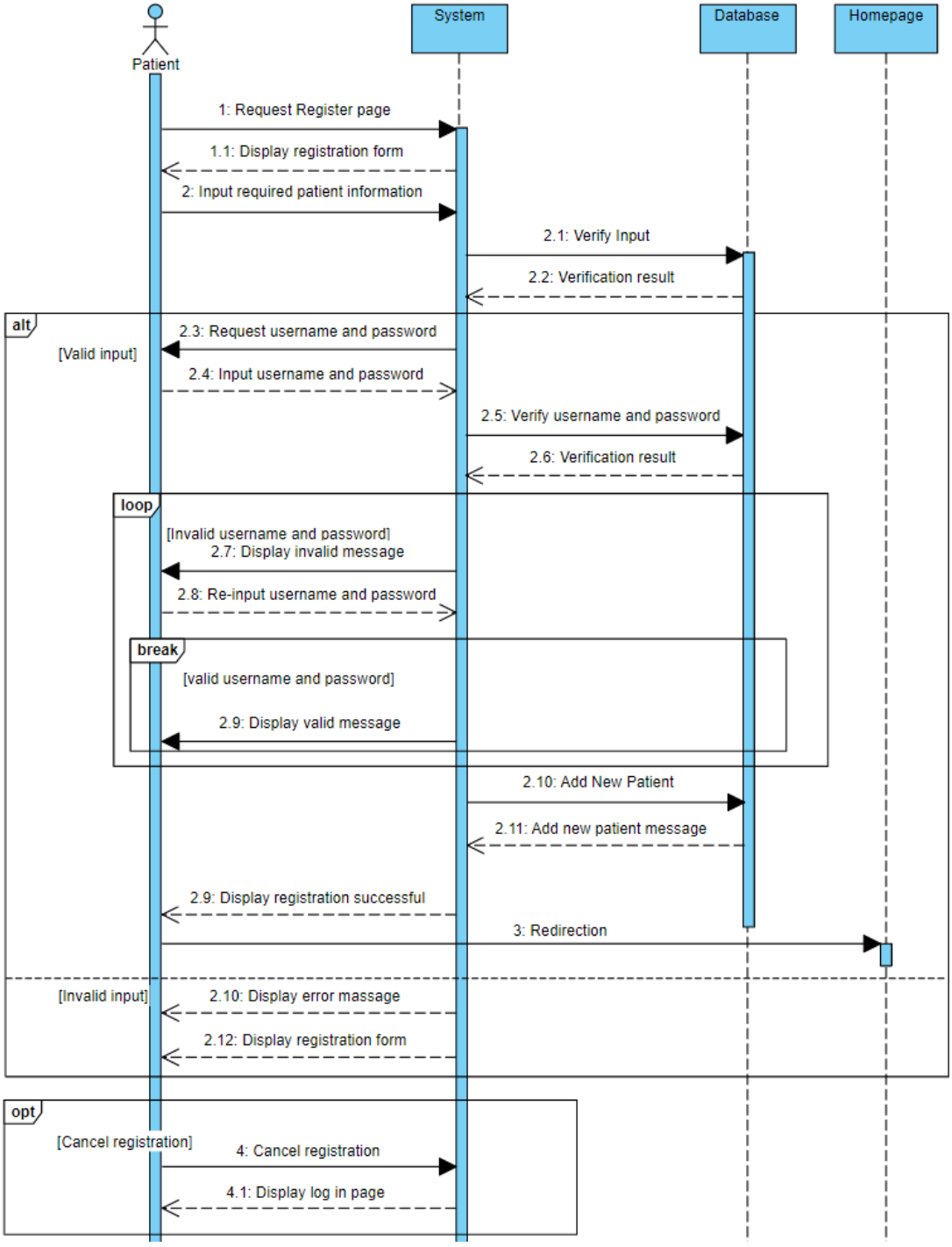

Figure 2. Register detail sequence diagram.

2.1.2 Log in system sequence diagram.

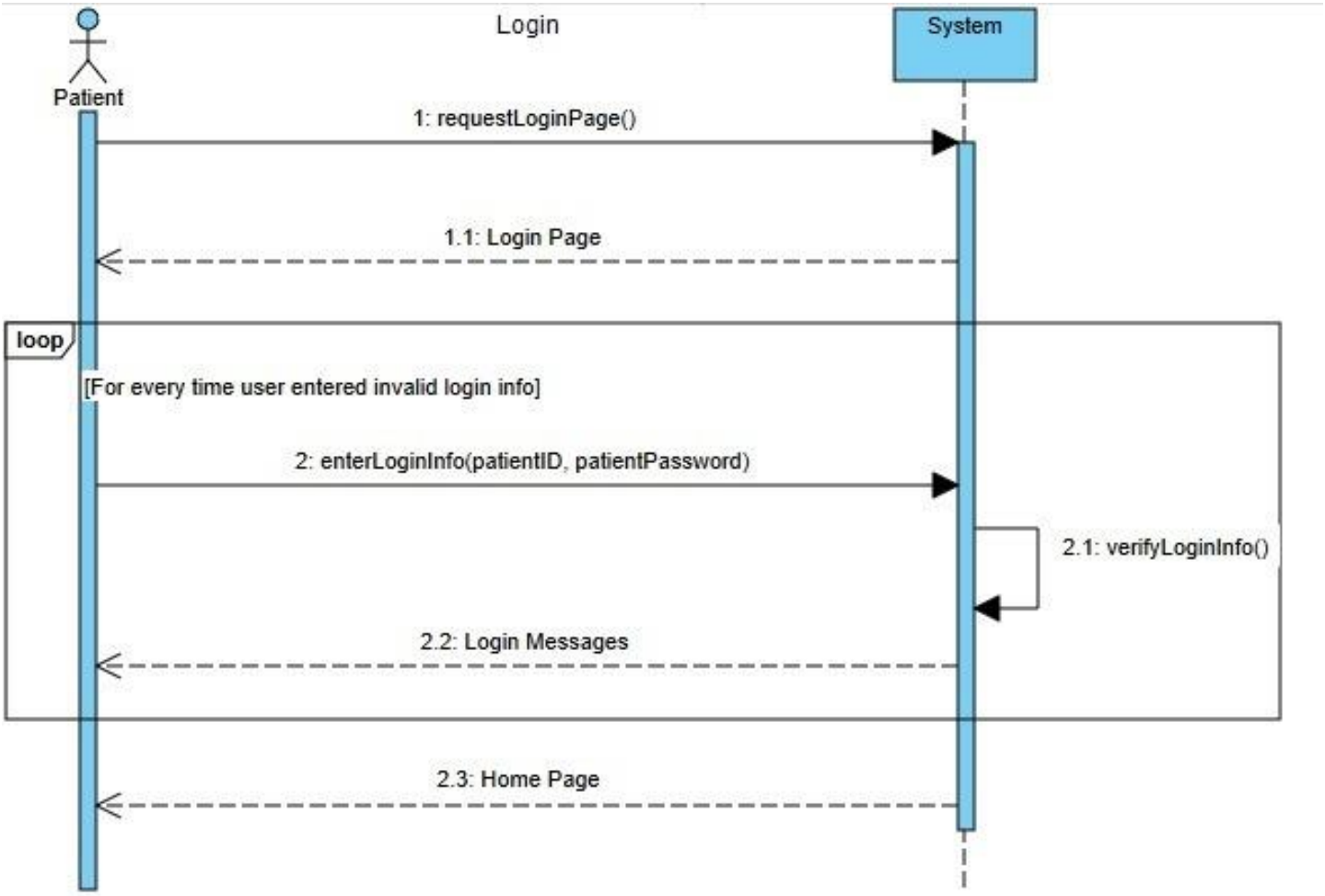

Figure 3. Log in system sequence diagram.

2.1.3 Make an appointment system sequence diagram.

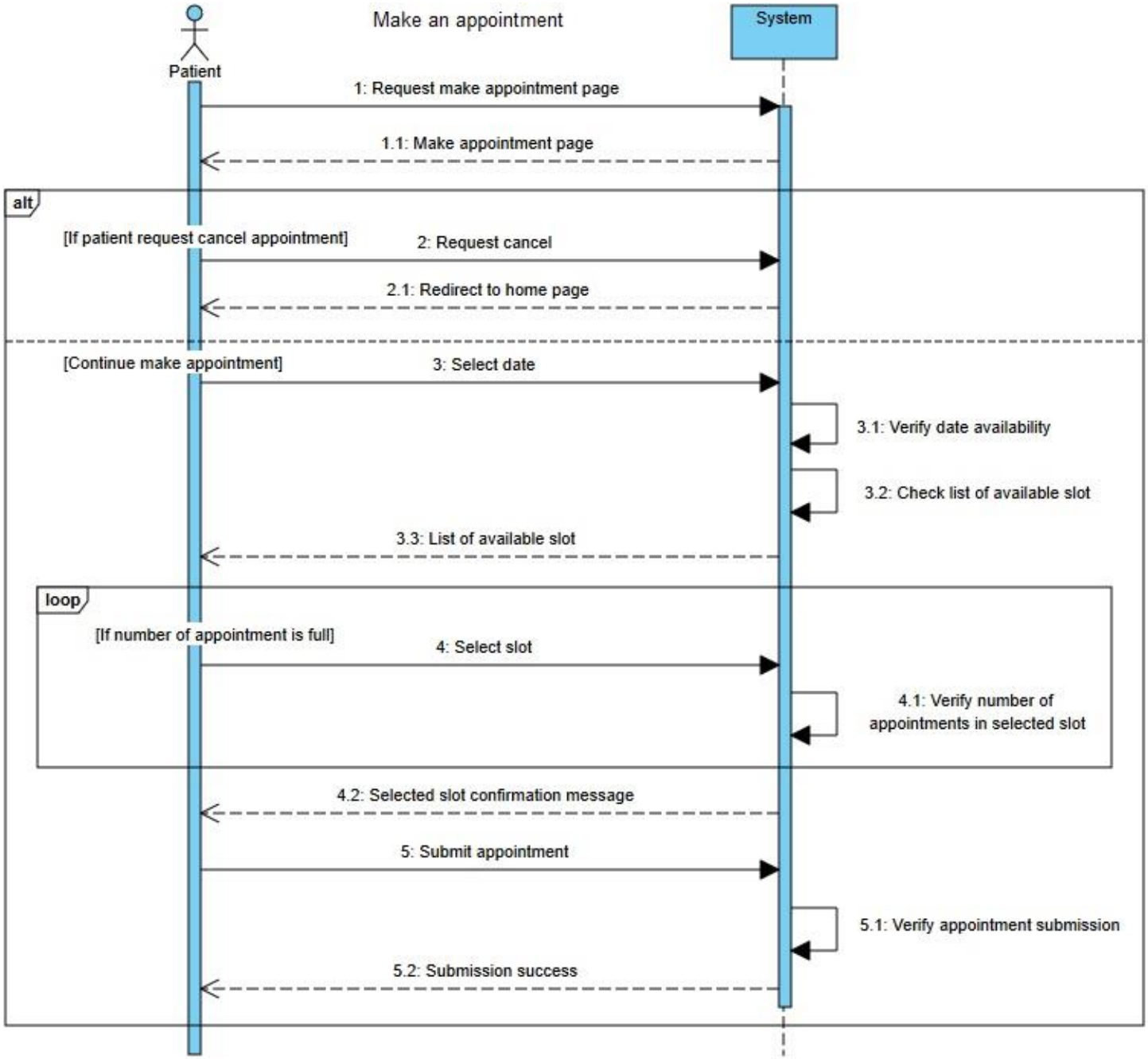

Figure 4. Make an appointment system sequence diagram.

2.1.4 Cancel appointment sequence diagram.

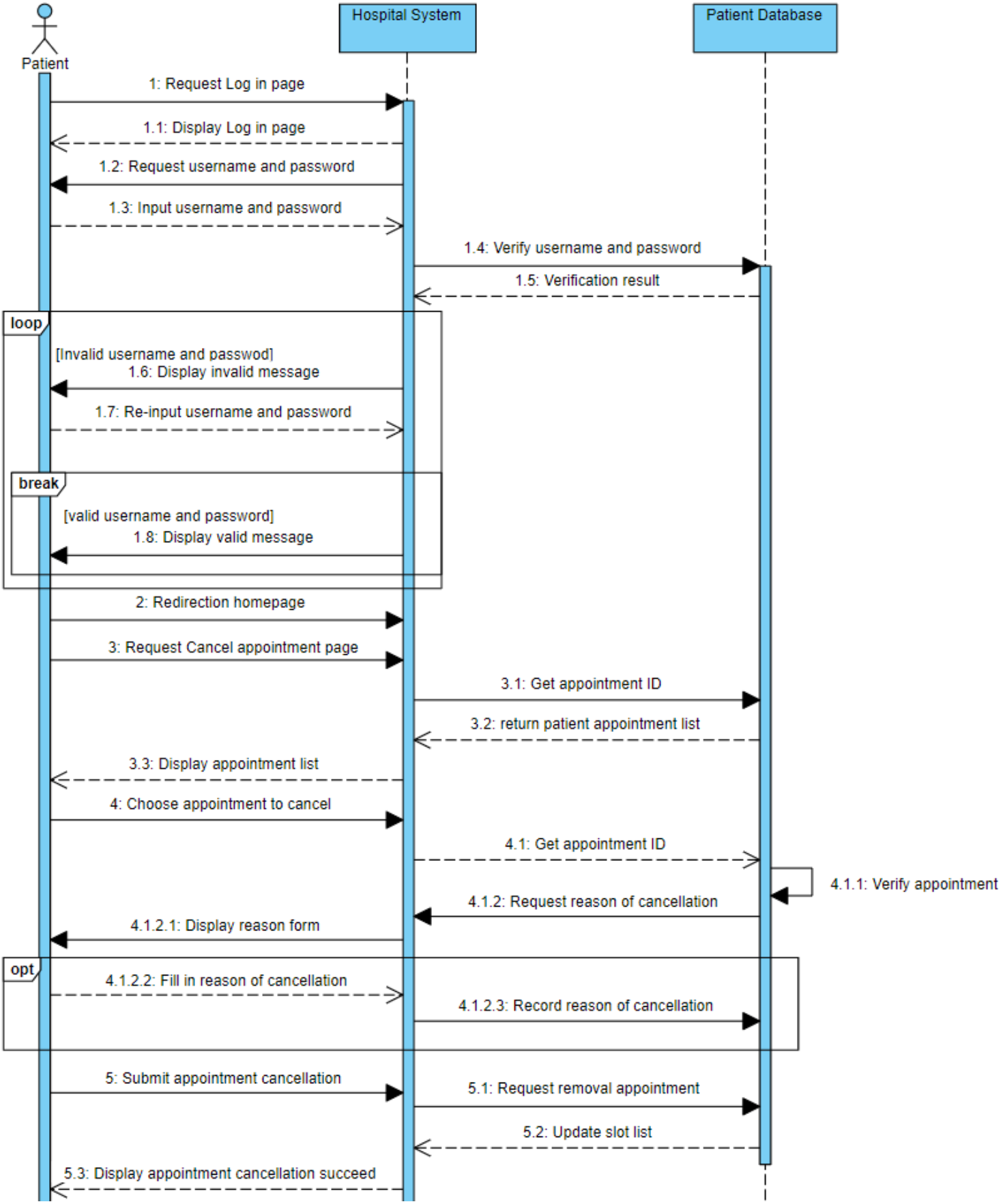

Figure 5. Cancel appointment sequence diagram.

2.1.5 View profile sequence diagram.

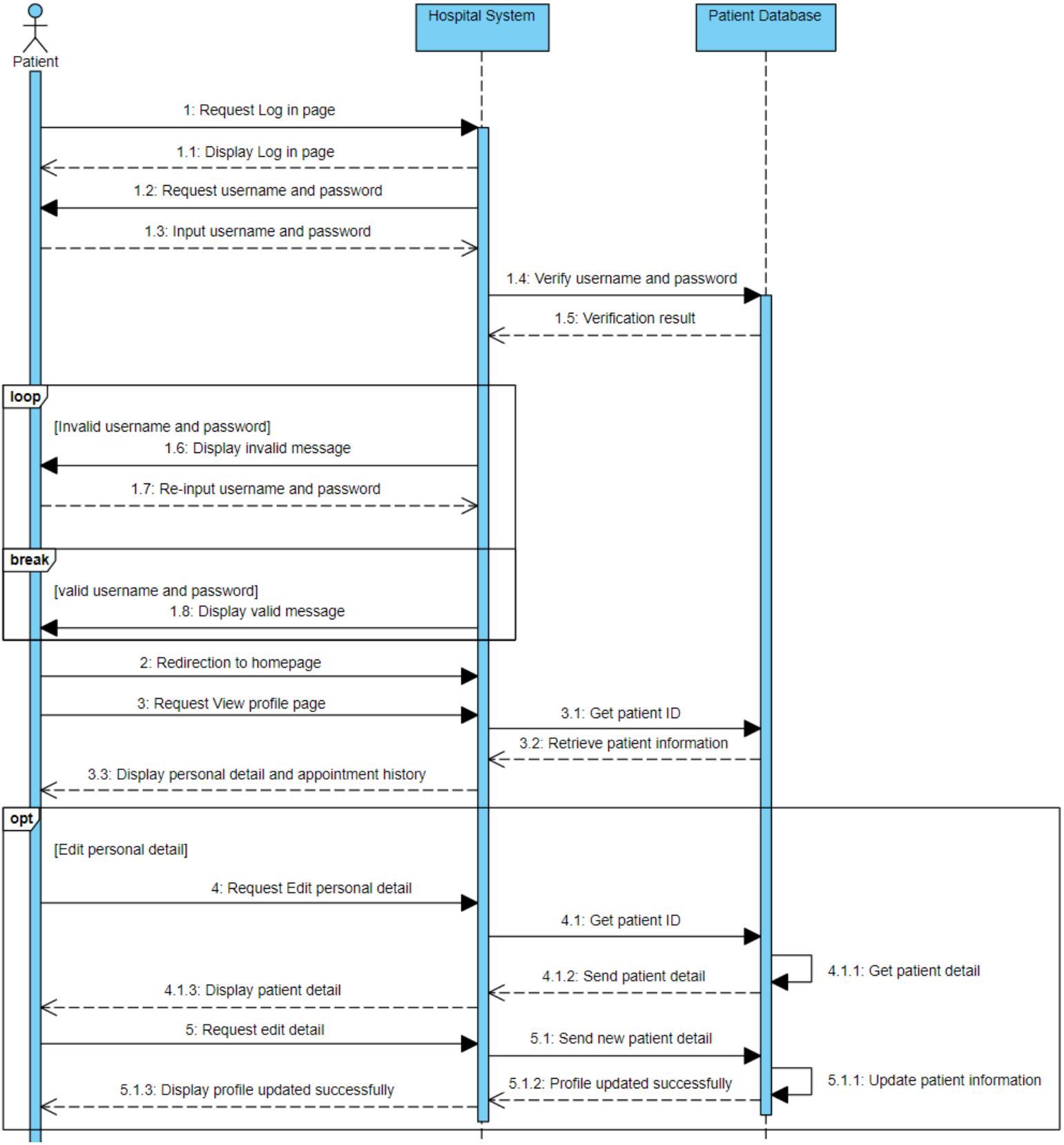

Figure 6. View profile sequence diagram.

2.1.6 Make payment sequence diagram

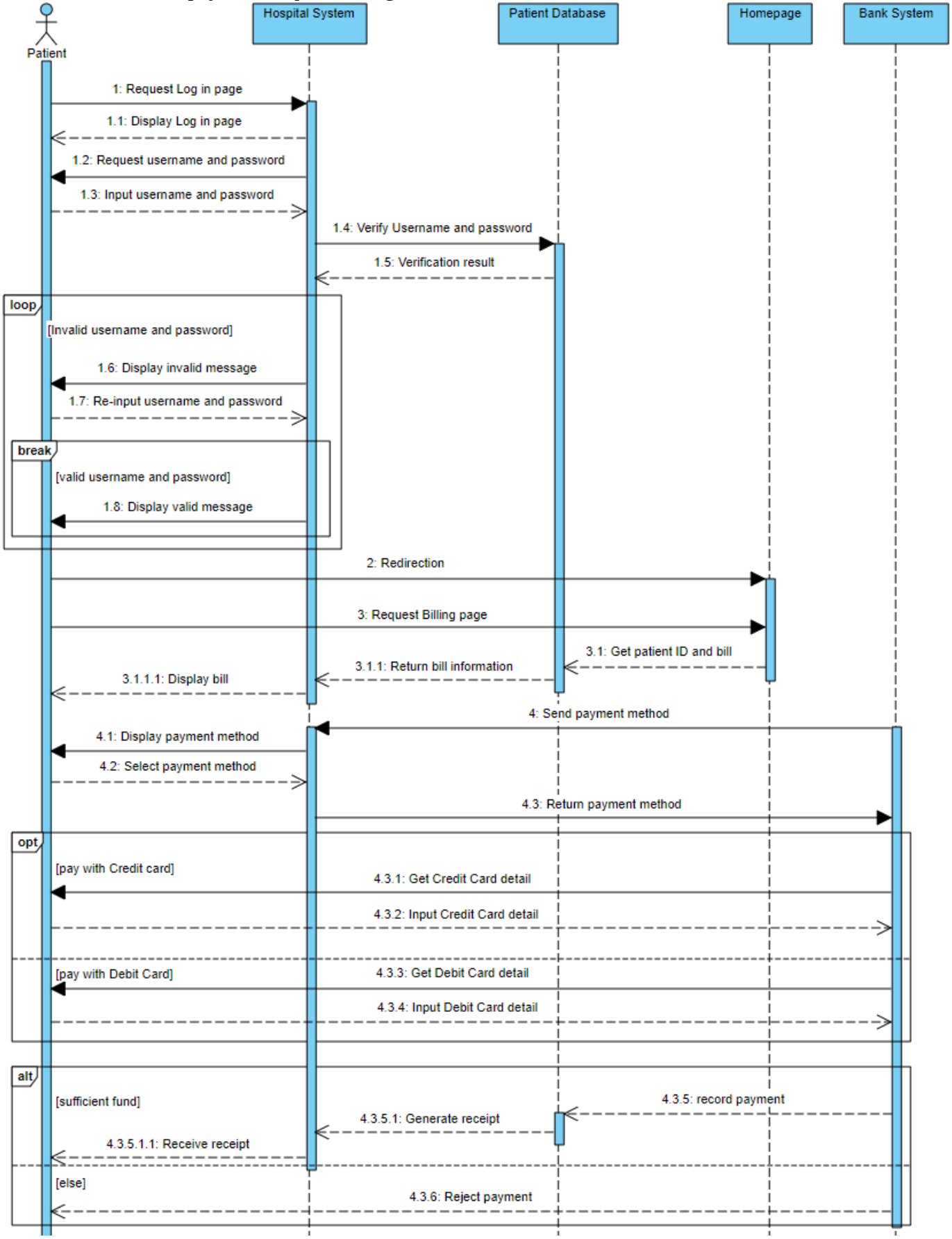

Figure 7. Make payment sequence diagram.

3.0 Question 3

Develop **ONE (1) activity diagram** to represent the business process of the proposed system.

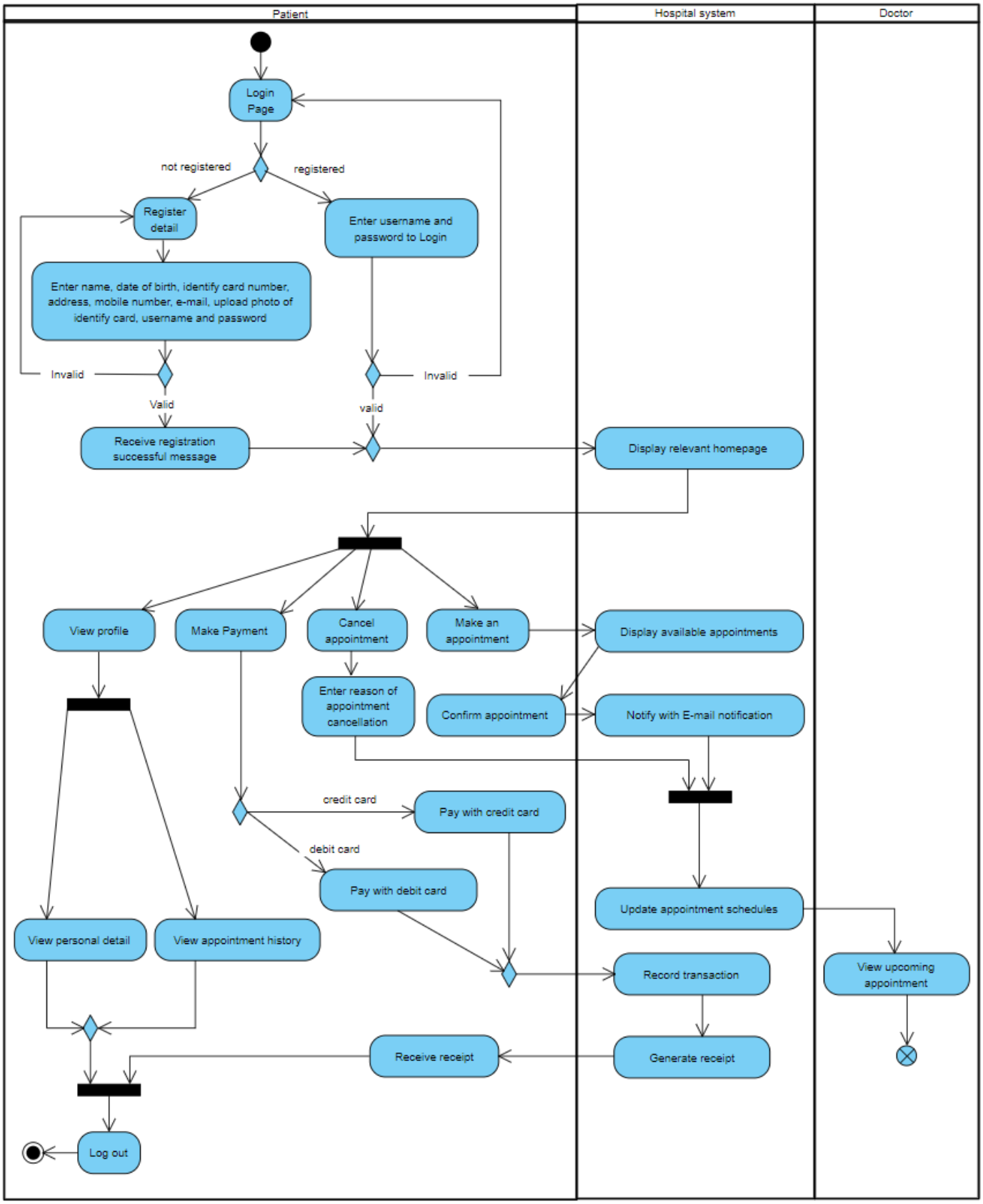

Figure 8. Business process model.

4.0 Individual Contribution

Table 9: Individual Contribution

Q	Task	Creator	Date
1	Use Case: Register detail	69385- Chris	10/11/20
	Use Case: Log in	69861- Holy	10/11/20
	Use Case: Make an appointment	69861- Holy	10/11/20
	Use Case: Confirm appointment	70653- Amar	10/11/20
	Use Case: Cancel appointment	72713- Lee	10/11/20
	Use Case: View profile	72713- Lee	10/11/20
	Use Case: View schedule	72368- Zaki	10/11/20
	Use Case: Make payment	72676- Mizah	10/11/20
	Use Case specification: Register detail	69385- Chris	17/11/20
	Use Case specification: Log in	69861- Holy	17/11/20
	Use Case specification: Make an appointment	69861- Holy	17/11/20
	Use Case specification: Confirm appointment	70653- Amar	17/11/20
	Use Case specification: Cancel appointment	72713- Lee	17/11/20
	Use Case specification: View profile	72713- Lee	17/11/20
	Use Case specification: View schedule	72368- Zaki	17/11/20
	Use Case specification: Make payment	72676- Mizah	17/11/20
	Requirement: Performance requirement	69385- Chris	17/11/20
	Requirement: Security requirement	69385- Chris	17/11/20
2	Sequence diagram: Register detail	69385- Chris	24/11/20
	Sequence system diagram: Log in	70653- Amar	24/11/20
	Sequence system diagram: Make an appointment	69861- Holy	24/11/20
	Sequence diagram: Cancel appointment	72368- Zaki	24/11/20
	Sequence diagram: View profile	72713- Lee	24/11/20
	Sequence diagram: Make payment	72676- Mizah	24/11/20
3	Business process: Login page	69385- Chris	25/11/20
	Business process: Register detail	69861- Holy	25/11/20
	Business process: Enter username & password	69385- Chris	25/11/20
	Business process: Enter patient detail	69861- Holy	25/11/20
	Business process: Receive successful message	69861- Holy	25/11/20

	Business process: Display relevant homepage	69861- Holy	25/11/20
	Business process: view profile	72713- Lee	25/11/20
	Business process: make payment	70653- Amar	25/11/20
	Business process: cancel appointment	70653- Amar	25/11/20
	Business process: make an appointment	72368- Zaki	25/11/20
	Business process: display available appointment	72368- Zaki	25/11/20
	Business process: confirm appointment	69385- Chris	25/11/20
	Business process: Notify with E-mail	70653- Amar	25/11/20
	Business process: Enter reason of cancellation	70653- Amar	25/11/20
	Business process: Update appointment schedule	72368- Zaki	25/11/20
	Business process: View upcoming appointment	72368- Zaki	25/11/20
	Business process: Pay with credit card	72676- Mizah	25/11/20
	Business process: Pay with Debit card	72676- Mizah	25/11/20
	Business process: Record transaction	72676- Mizah	25/11/20
	Business process: Generate receipt	72676- Mizah	25/11/20
	Business process: Receive receipt	72676- Mizah	25/11/20
	Business process: View personal detail	72713- Lee	25/11/20
	Business process: View appointment history	72713- Lee	25/11/20
	Business process: Log out	69385- Chris	25/11/20