Write-up of CARS's architecture and design

Name
Loon Hai Qi
Madeline Tooh Weiping
Nurul Afiqah Binte Rashid

Table Of Contents

High Level Architecture and Tiers

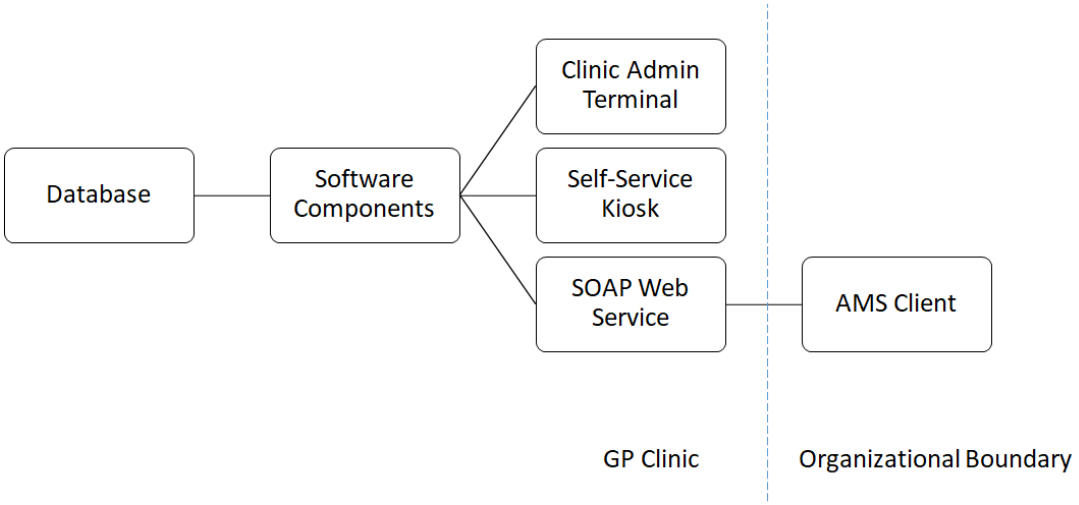

Figure 1: High Level Architecture of Clinic Appointment Registration System (CARS)

Database Tier

The database tier is typically the database server in conjunction with Relational Database Management System(RDBMS) such as MySQL. All information related to staff, patients, doctors, appointments, as well as consultations is stored here.

Logic Tier

The logic tier is where the business logic layer operates. Also, the software elements and components can be found in this layer to carry out the administration, appointment and registration operations of CARS using the Java Platform Enterprise Edition (Java EE), together with Enterprise JavaBeans (EJB) and Java Persistence API (JPA). Other than that, the SOAP web services or framework also resides in this layer.

Presentation Tier

The Command-line Interface (CLI) client applications such as Clinic Admin Terminal, Self-Service Kiosk Terminal and AMS Client resides in the presentation tier.

Below is a illustration of the tiers in CARS.

Presentation Tier	CLI	Clinic Admin Terminal
		Self-Service Kiosk Terminal
		AMS Client

	Web Browser	
Logic Tier	Java EE	EJB
		JPA
		JNDI
	SOAP Web Services	
Database Tier	RDBMS	MySQL

Table 1: Breakdown of Tiers in CARS

Overview of Operations in Clients

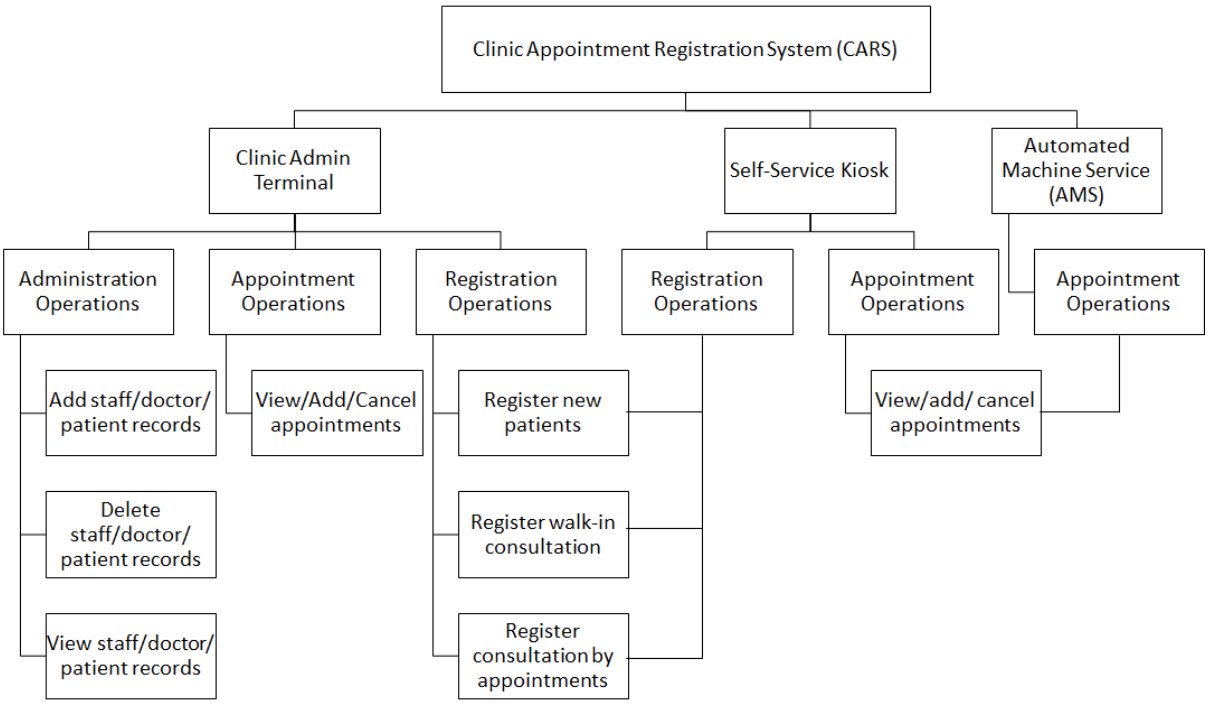

Figure 2: Overview of Operations in Clients under CARS

Clients

The Clinic Admin Terminal supports the clinic appointment, registration, as well as administration workflow.
The purpose of the Self-Service Kiosk Terminal is for patients to do self-registration and appointment booking.
The AMS Client is created for testing purposes of the web services and they are used for interfacing with external vendors.

Administration Operations

The administration operations are intended for the clients to add, delete and view staffs, doctors and patients records from the database. Starting from the clients' interfaces, the Registration Module provides a platform for the clients to interact with the staffs, doctors and patients' data and make an changes to the database using local and remote controller interfaces from the respective entities' remote controllers (DoctorEntityController, PatientEntityController, StaffEntityController).

Registration Operations

The registration operations does the registering of patients, consultations and appointments by accessing local and remote controller interfaces from the respective entity's remote

controllers (DoctorEntityController, PatientEntityController, AppointmentEntityController and ConsultationEntityController). Creation of consultation entities also involves @OneToMany and @ManyToOne relationships to be invoked in the operations.

Appointment Operations

The appointment operations allows users to view, add and cancel appointments by accessing local and remote controller interfaces from the respective entity's remote controllers (DoctorEntityController, PatientEntityController and AppointmentEntityController). Creation of appointment entities also involves @OneToMany and @ManyToOne relationships to be invoked n the operations.

Project Structure

Below is a table representing the NetBeans enterprise application project structure. The project structure is briefly specified with its file type and description of activities happening in each module.

Project Structure	**Type**	**Description**
AMSTerminalClient	Project	Enterprise Application Client
amsterminalclient	Package	
Main.java	Class	Injection of @EJB for all remote controllers into MainApp.java
MainApp.java	Class	Main methods of appointment operations found in here.
ClinicAdminTerminalClient	Project	Enterprise Application Client
clinicadminterminalclient	Package	
AdministrationModule.java	Class	Main methods of administration operations found in here.
AppointmentModule.java	Class	Main methods of appointment operations found in here.
Main.java	Class	Injection of @EJB for all remote controllers into MainApp.java
MainApp.java	Class	Main method to call for main menu, which has the main operations such as administration, appointment and registration operations
RegistrationModule.java	Class	Main methods of registration operations found in here.
SelfServiceKiockTerminalClient	Project	Enterprise Application Client
selfservicekioskterminalclient	Package	
AppointmentModule.java	Class	Main methods of appointment operations found in here.
Main.java	Class	Injection of @EJB for all remote controllers into MainApp.java
MainApp.java	Class	Register patient operations found here.
RegistrationModule.java	Class	Main methods of registering consultation and appointment found here.

ClinicAppointmentRegistrationSystem-ejb	Project	EJB Module
ejb.session.singleton	Package	
DataInitializationSesssion Bean.java	Class	Initialise information in the database when running and starting up EJB
ejb.session.stateful	Package	
RegistrationController.java	Class	Keeps track of queue number and it will track without disruption since EJB is stateful.
RegistrationControllerLocal.java	Interface	Implemented by RegistrationController.java
ejb.session.stateless	Package	
AppointmentEntityController.java	Class	Uses EntityManager to create methods to add, cancel, view and retrieve AppointmentEntity entities from and to database.
AppointmentEntityControllerLocal.java	Interface	Implemented by AppointmentEntityController.java
ConsultationEntityController.java	Class	Uses EntityManager to create methods to add, view and retrieve ConsultationEntity from and to database.
ConsultationEntityControllerLocal.java	Interface	Implemented by ConsultationEntityController.java
DoctorEntityController.java	Class	Uses EntityManager to create, delete, update and retrieve DoctorEntity from and to database
DoctorEntityControllerLocal.java	Interface	Implemented by DoctorEntityController.java
PatientEntityController.java	Class	Uses EntityManager to create, delete, update and retrieve PatientEntity from and to database. Also controls patient login to check credentials
PatientEntityControllerLocal.java	Interface	Implemented by PatientEntityControllerLocal.java
StaffEntityController.java	Class	Uses EntityManager to create,

		delete, update and retrieve Staff Entity from and to database. Also controls staff login to check credentials
StaffEntityControllerLocal.java	Interface	Implemented by StaffEntityController.java
ejb.session.ws	Package	
AMSWebService.java	Class	Web Service Client
ClinicAppointmentRegistrationSystemLibrary	Project	Java Class Library
ejb.session.stateful	Package	
RegistrationControllerRemote.java	Interface	Implemented by RegistrationController.java
ejb.session.stateless	Package	
AppointmentControllerRemote.java	Interface	Implemented by AppointmentEntityController.java
ConsultationControllerRemote.java	Interface	Implemented by ConsultationEntityController.java
DoctorControllerRemote.java	Interface	Implemented by DoctorEntityController.java
PatientControllerRemote.java	Interface	Implemented by PatientEntityController.java
StaffControllerRemote.java	Interface	Implemented by StaffEntityController.java
entity	Package	
AppointmentEntity.java	Class	This method creates the Appointment Entity with its following attributes, appointmentId, time, date with relationship to PatientEntity and DoctorEntity. Getter and setter methods are also made available to access.
ConsultationEntity.java	Class	This method creates the ConsultationEntity with its following attributes, consultationId, time with relationship to PatientEntity and DoctorEntity. Getter and setter methods are also made

		available to access.
DoctorEntity.java	Class	Set primary key using doctorId. This method creates the DoctorEntity. Getter and setter methods are also made available to access.
PatientEntity.java	Class	Set primary key using identityNumber. This method creates the PatientEntity. Getter and setter methods are also made available to access.
StaffEntity.java	Class	Set primary key using staffId. This method creates the StaffEntity. Getter and setter methods are also made available to access
util.exception	Package	
AppointmentNotFoundException.java	Class	Throws exception if appointment is not found.
DoctorAddAppointmentException.java	Class	Throws exception if appointment is already added to doctor.
DoctorAddConsultationException.java	Class	Throws exception if consultation is already added to doctor.
DoctorNotFoundException.java	Class	Throws exception if doctor is not found.
DoctorRemoveAppointmentException.java	Class	Throws exception if appointment is missing from doctor.
InvalidLoginException.java	Class	Throws exception if login credentials does not match.
PatientAddAppointmentException.java	Class	Throws exception if appointment is already added to patient.
PatientAddConsultationExeption.java	Class	Throws exception if consultation is already added to patient.
PatientNotFoundException.java	Class	Throws exception if patient is not found.
PatientRemoveAppointmentE	Class	Throws exception if

xception.java		appointment is missing from patient.
PatientRemoveConsultationException.java	Class	Throws exception if consultation is missing from patient.
StaffNotFoundException.java	Class	Throws exception if staff is not found.

Table 2: Project Structure of CARS in NetBeans

Entity Classes, Attributes and Relationships

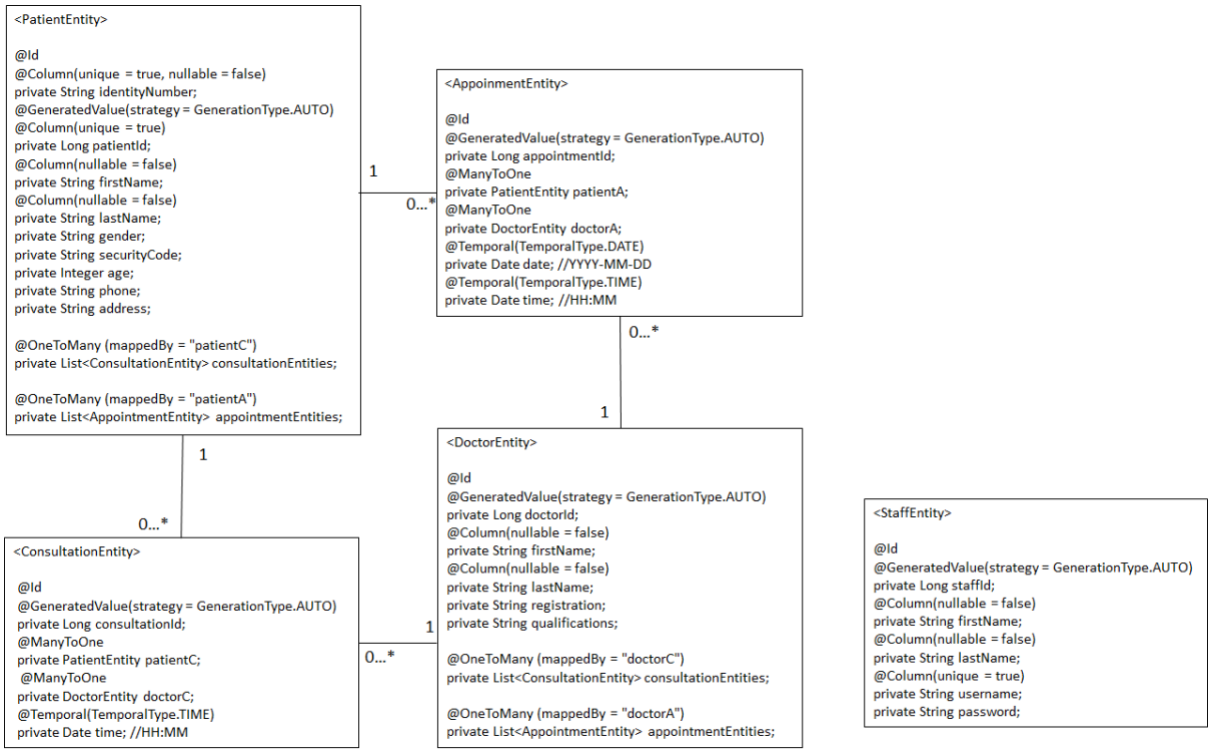

Figure 3: Entity Classes, Attributes and Relationships among the Entity Classes

Figure 3 shows the overall relationships between the entities classes in CARS. There are five entities, which are ConsultationEntity, AppointmentEntity, StaffEntity, DoctorEntity and PatientEntity.

StaffEntity is not bounded by any relationship constraints. However, DoctorEntity and PatientEntity has a @OneToMany relationship with ConsultationEntity and AppointmentEntity, while ConsultationEntity and AppointmentEntity have a @ManyToOne relationship with PatientEntity and DoctorEntity.

The relationship keeps track that every creation of ConsultationEntity and AppointmentEntity have one Patient Entity and one DoctorEntity set to it. This also ensures that the DoctorEntity or PatientEntity can keep track of the list of consultations and appointments they have. For instance, when both consultation and appointment are created, they are both binded with a PatientEntity and a DoctorEntity.

With the @OneToMany and @ManyToOne relationships occurring, there is a need to make set methods when creating the Appointment Entity or Consultation Entity. When adding these two entities, the PatientEntity and DoctorEntity have to be set. When cancelling appointments, the AppointmentEntity is removed by EntityManager, and the appointment binded within the Patient and Doctor Entity lists are removed.

For each entity, it has unique ID (primary key) to search for the respective entities. For PatientEntity, it is retrieved using identityNumber as the primary key. For StaffEntity, it is

retrieved using staffId as the primary key. For DoctorEntity it is retrieved using doctorId as the primary key. For ConsultationEntity, the @Id is set as consultationId while for Appointment Entity, the @Id is set as appointmentId. Furthermore, these respective Ids are also set to be increased automatically by the database when it is created.

Business Rules, Rational and Assumptions

Clinic Admin Terminal

Use Case	Description/ Business Rules	Assumptions/ Solutions
Login	● May only be performed if staff is not currently logged into the system. ● Staff must be currently logged into the system to perform all other use cases.	● Staff Not Found exception caught. ● InvalidLoginException caught.
Register New Patient	● If a patient is not registered, admin staff registers a new patient record in the system. Admin staff needs to ask and enter patient information: ○ Identity Number (NRIC or Passport) ○ Security Code ○ First Name ○ Last Name ○ Gender ○ Age ○ Phone ○ Address	● Invalid inputs may happen. ● Patient should not already exist in the database.
Register Walk-in Consultation	● Admin staff registers a walk-in consultation for a registered patient. ● The system provides admin staff with 3-hour availability of all doctors for registration. ● A queue number will be given to the patient once the consultation registration is done.	● Checks if time is within operating hours and printing the right time slots available. ● Check if consultation available time slots are within 3 hour frame. ● Add consultation concurrently while setting doctor and patient. ● Sets queue number using a stateful EJB controller.
Register Consultation by Appointment	● Admin staff view appointments with the patient identity number ● The system allows admin staff to select which appointment to register. ● A queue number will be given to the patient once the consultation registration is done.	● Appointment has to exist to be added to consultation. ● Consultation slot has to be empty during the appointment time frame.

View Appointments	● The system list down all appointments for the patient using his/her identity number.	● Retrieves all appointments using patient's identity number.
Add Appointment	● The system shows list of the doctors, and allows admin staff to select doctor, and select date for adding an appointment. ● The system provides the availability of the doctors for the date. ● Admin staff enters the time and patient identity number for finalising the appointment.	● Add appointment concurrently while setting doctor and patient.
Cancel Appointment	● The system shows the list of patient appointments, given his/her patient identity number. ● Admin staff enters the appointment id to cancel the appointment.	● Appointment Not Found exception caught. ● Retrieves all appointments using patient's identity number. ● Cancels appointment by appointmentId
Manage Patients	● The system provides functionalities for admin staff to perform: ○ Add Patient ○ View Patient Details ○ Update Patient ○ Delete Patient ○ View All Patients	● PatientNotFoundExcepti on caught.
Manage Doctors	● The system provides functionalities for admin staff to perform: ○ Add Doctor ○ View Doctor Details ○ Update Doctor ○ Delete Doctor ○ View All Doctors	● DoctorNotFoundExcepti on caught.
Manage Staff	● The system provides functionalities for admin staff to perform: ○ Add Staff ○ View Staff Details ○ Update Staff ○ Delete Staff ○ View All Staff	● StaffNotFoundException caught.
Logout	● Logout if staff is currently	● Breaks while (true) loop

	logged in the system	to log out

Table 3: Use Case Descriptions and Business Rules of Clinic Admin Terminal

Self-Service Kiosk Terminal

Use Case	**Description/ Business Rules**	**Assumptions/ Rational**
Login	● If a patient is registered, the patient will login into the system using: ○ Identity Number ○ Security Code	● Patient Not Found exception caught. ● Invalid Login exception caught.
Register New Patient	● If a patient is not registered, system will prompt patient to register by entering personal particulars: ○ Identity Number (NRIC or Passport) ○ Security Code ○ First Name ○ Last Name ○ Gender ○ Age ○ Phone ○ Address	● Invalid inputs may happen. ● Patient should not already exist in the database.
Register Walk-in Consultation	● The system provides the patient with 3-hour availability of all doctors for registration. ● The patient selects a doctor for consultation. ● A queue number will be given to the patient once the consultation registration is done.	● Checks if time is within operating hours and printing the right time slots available. ● Check if consultation available time slots are within 3 hour frame. ● Add consultation concurrently while setting doctor and patient. ● Sets queue number using a stateful EJB controller.
Register Consultation by Appointment	● The system shows the list of the patient appointments. ● The system allows the patient to select which appointment to register. ● A queue number will be given to the patient once the consultation	● Appointment has to exist to be added to consultation. ● Consultation slot has to be empty during the appointment time frame.

	registration is done.	
View Appointments	● The system list down all appointments for the patient.	● Retrieves all appointments using patient's identity number.
Add Appointment	● The system shows list of the doctors, and allows the patient to select doctor, and select date for adding an appointment. ● The system provides the availability of the doctors for the date. ● The patient enters the time for finalising the appointment.	● Add appointment concurrently while setting doctor and patient.
Cancel Appointment	● The system shows the list of patient appointments. ● The patient enters the appointment id to cancel the appointment.	● Appointment Not Found exception caught. ● Retrieves all appointments using patient's identity number. ● Cancels appointment by appointmentId
Logout	● Logout if patient is currently logged in the system	● Breaks while (true) loop to log out

Table 4: Use Case Descriptions and Business Rules of Self-Service Kiosk

AMS Client

Use Case	**Description/ Business Rules**	**Assumptions/ Rational**
Login	● If a patient is registered, the patient will login into the system using: ○ Identity Number ○ Security Code	● Patient Not Found exception caught. ● Invalid Login exception caught.
Register New Patient	● If a patient is not registered, system will prompt patient to register by entering personal particulars: ○ Identity Number (NRIC or Passport) ○ Security Code ○ First Name ○ Last Name ○ Gender	● Invalid inputs may happen. ● Patient should not already exist in the database.

	○ Age ○ Phone ○ Address	
View Appointments	● The system list down all appointments for the patient.	● Retrieves all appointments using patient's identity number.
Add Appointment	● The system shows list of the doctors, and allows the patient to select doctor, and select date for adding an appointment. ● The system provides the availability of the doctors for the date. ● The patient enters the time for finalising the appointment.	● Add appointment concurrently while setting doctor and patient.
Cancel Appointment	● The system shows the list of patient appointments. ● The patient enters the appointment id to cancel the appointment.	● Appointment Not Found exception caught. ● Retrieves all appointments using patient's identity number. ● Cancels appointment by appointmentId
Logout	● Logout if patient is currently logged in the system	● Breaks while (true) loop to log out

Table 5: Use Case Descriptions and Business Rules of AMS Client

Instructions to Test and Deploy

Set up database = “clinicappointmentregistrationsystem”

Clinic Admin Terminal

Login

The Login interface will be shown when the system is clicked open

1. Operation 1: To login to the system with username and password
    a. Correct username and password: “Login successful!” and system will display Main Menu CLI
    b. Wrong username and password: “Invalid login: Username does not exist or invalid password!”. Back to step 1 and try again.
2. Operation 2: To exit the system application

Main Menu

Main menu interface is shown, with user’s first name and last name

1. Operation 1: To perform registration operation
2. Operation 2: To perform appointment operation
3. Operation 3: To perform administration operation
4. Operation 4: To end user session and exit the system application

Registration Operation

Registration operation interface is shown.

1. Operation 1: To register new patient to the clinic database
    a. Required to fill in patient’s particulars (Identity number, security code, first name, last name, gender, age, phone, address)
        i. If patient has not registered with the clinic before: “Patient has been registered successfully!”
        ii. If patient has previously registered with the clinic: “Patient is already created.”.
2. Operation 2: To register patient as walk-in consultation
    a. System displays the list of doctors in the clinic, as well as their available time for consultation
    b. Enter doctor id
    c. Enter patient identity number
        i. If Patient Identity Number is invalid (not in database): “An error has occurred while retrieving patient: Patient Identity Number (Patient’s Identity Number) does not exist!”
        ii. Otherwise: “(Patient’s first name and last name) is going to see (doctor’s first name and last name) at (time). Queue Number is: (queue number).”
3. Operation 3: To register patient consultation by the appointment booked beforehand
    a. Enter patient identity number

b. System will display a list of appointments which this patient has previously booked
    i. An empty table if patient does not has any appointment.
c. Enter appointment id
d. A queue number is generated.
e. System will then display:
"(Patient's first name and last name) is going to see (doctor's first name and last name) at (time). Queue Number is: (queue number)."

4. Operation 4: To return to the main menu

Appointment Operation

Appointment operation interface is shown.

1. Operation 1: To view the list of appointments that a particular patient has booked
    a. Enter patient's identity number
        i. If patient's identity number is incorrectly entered: "An error has occurred while retrieving patient: Patient Identity Number (Patient Identity Number) does not exist!"
    b. System will display the list of appointments which this patient has previously booked
        i. An empty table if patient does not has any appointment.

2. Operation 2: To add appointment to a patient
    a. System displays the list of doctors in the clinic
    b. Enter doctor id
    c. Enter date for appointment to be booked
    d. System displays the list of time that the doctor is available for consultation on the particular date
    e. Enter time
    f. Enter patient identity number
    g. System will then display:
    "Appointment: (Patient's first name and last name) and (doctor's first name and last name) at (time) on (date) has been added."
3. Operation 3: To cancel patient's appointment
    a. Enter patient's identity number
        i. If patient's identity number is incorrectly entered: "An error has occurred while retrieving patient: Patient Identity Number (Patient Identity Number) does not exist!"
    b. System will display a list of appointments that the patient has booked
    c. Enter appointment id
    d. System will then display:
    "Appointment: (Patient's first name and last name) and (doctor's first name and last name) at (time) on (date) has been cancelled."
4. Operation 4: To return to the main menu

Administration Operation

Administration operation interface is shown.

1. Operation 1: Patient Management
    a. Operation 1: To add new patient

        i. System will display: “New patient created successfully!: (Patient’s Identity Number)” upon successful addition.
    b. Operation 2: To view a particular patient’s details
        i. Enter Patient Identity Number
        ii. System will display a table with patient’s particulars.
    c. Operation 3: To update patient
        i. Enter Patient Identity Number
        ii. System will display a table with patient’s particulars.
        iii. System will prompt users to update any changes and show “Patient updated successfully!”
    d. Operation 4: To delete a patient from the clinic database
        i. Enter Patient Identity Number
        ii. System will display a table with patient’s particulars.
        iii. Delete patient’s records if user confirm the deletion.
    e. Operation 5: To view all patients in the clinic
        i. System will display a table with all patients’ particulars.
    f. Operation 6: To return to administration operation interface
2. Operation 2: Doctor Management
    a. Operation 1: To add new doctor
        i. System will display: “New doctor created successfully!: (Doctor’s ID)” upon successful addition.
    b. Operation 2: To view a particular doctor’s details
        i. Enter Doctor ID
        ii. System will display a table with doctor’s particulars.
    c. Operation 3: To update doctor
        i. Enter Doctor ID
        ii. System will display a table with doctor’s particulars.
        iii. System will prompt users to update any changes and show “Doctor updated successfully!”
    d. Operation 4: To delete a doctor from the clinic database
        i. Enter Doctor ID
        ii. System will display a table with doctor’s particulars.
        iii. Delete doctor’s records if user confirm the deletion.
    e. Operation 5: To view all doctors in the clinic
        i. System will display a table with all doctors’ particulars.
    f. Operation 6: To return to administration operation interface
3. Operation 3: Staff Management
    a. Operation 1: To add new staff
        i. System will display: “New staff created successfully!!: (Staff’s ID)” upon successful addition.
    b. Operation 2: To view a particular staff’s details
        i. Enter Staff ID
        ii. System will display a table with staff particulars.
    c. Operation 3: To update staff
        i. Enter Staff ID
        ii. System will display a table with staff particulars.
        iii. System will prompt users to update any changes and show “Staff updated successfully!”

d. Operation 4: To delete a staff from the clinic database
    i. Enter Staff ID
    ii. System will display a table with staff particulars.
    iii. Delete staff's records if user confirm the deletion.
e. Operation 5: To view all staffs in the clinic
    i. System will display a table with all staffs particulars.
f. Operation 6: To return to administration operation interface

4. Operation 4: To return to main menu

Self-Service Kiosk

Login

The Login interface will be shown when the system is clicked open

1. Operation 1: To register patient
    a. Required to fill in patient's details (Identity number, security code, first name, last name, gender, age, phone, address)
        i. If patient has not registered with the clinic before: "Patient has been registered successfully!"
        ii. If patient has previously registered with the clinic: "Patient is already created.".
2. Operation 2: To login to the system
    a. Required to fill in identity number and security code
        i. If patient has previously registered with the clinic before: "Patient has been registered successfully!"
        ii. If patient has not registered with the clinic: "Invalid login: Identity number does not exist or invalid security code!".
3. Operation 3: To exit the system application

Patient Menu

The Patient Menu interface is shown, with user's first name and last name

1. Operation 1: To register as walk-in consultation
    a. System will display a list of doctors and their availabilities. Users cannot register if "There are no more consultations because clinic is either closed, closing soon or not opened yet." is displayed.
2. Operation 2: To register patient consultation by the appointment they have booked previously
    a. System will display a list of patient's appointments. Users cannot register if "There are no more consultations because clinic is either closed, closing soon or not opened yet." is displayed.
3. Operation 3: To view appointments
    a. Enter Patient Identity Number
    b. System will display a list of appointment under the patient.
4. Operation 4: To add appointment
    a. System will display a list of doctors and prompt patient to enter Doctor ID for appointment addition.

5. Operation 5: To cancel appointment
    a. System will display a list of doctors and prompt patient to enter Appointment ID for deletion.
6. Operation 6: To end user session and exit from the system application

Automated Machine Services

Invoke AMSWebService from EJB web service.

1. To register patients
    a. Required to fill in patient's details (Identity number, security code, first name, last name, gender, age, phone, address) and click the "registerPatient" button to register.
        i. If patient has not registered with the clinic before: It will redirect users to another web page with output "You has been registered successfully!". It also includes listing all the inputs keyed in to register.
2. To login to system
    a. Required to fill in identity number and security code
        i. If patient has previously registered with the clinic before: It will redirect users to another web page with output "Login successful". It also includes listing all the inputs keyed in to log in.
3. To add appointment
    a. Required to fill in patient's identity number, security code, doctor id, date and time to add appointment
    b. The system will redirect users to another web page and displays the new appointment entity. Also, the inputs keyed in to add appointment.
4. To cancel appointment
    a. Required to fill in appointmentId, patient's identity number and security code
    b. The system will redirect users to another web page that displays "Appointment cancelled." if cancellation of appointment is successful.
5. To views appointments
    a. Required to fill in patient's identity number and security code and system will display a list of appointments that the patient has.
    b. The system will redirect users to another web page that displays a list of the patient's appointments if cancellation of appointment is successful.