Product Design

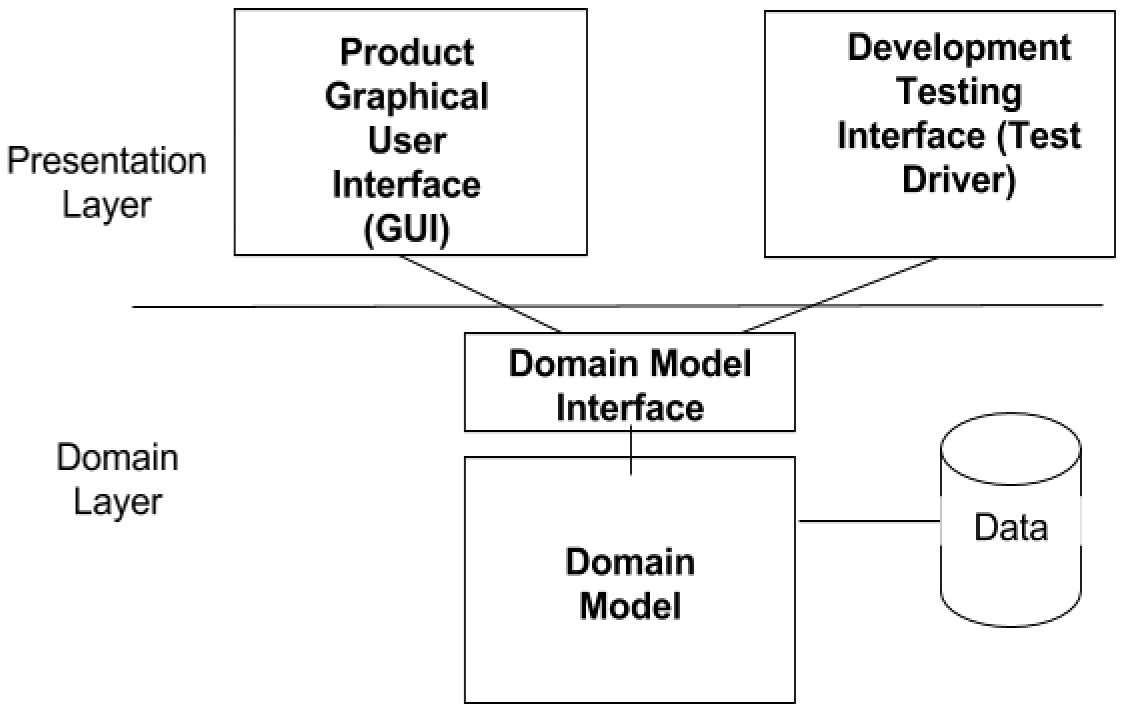
Team D: GoldTeamRules
Stephen Cook, Xixiao Yue, Chris Letourneau,
Oren Rosin, Georgiy Rozenshteyn

Revision Number	***Revision Date***	***Summary of Changes***	***Author(s)***
0.1	*02/11/2017*	*Initial creation of project plan document*	*Stephen Cook*
0.2	*02/11/2017*	*Creation of initial components and class diagrams*	*Whole Team*
0.3	*02/14/2017*	*Updates to Appointment, Appointment Calendar, Patient, and Doctor Classes*	*Whole Team*
0.4	*02/16/2017*	*Addition of more Release One states*	*Whole Team*
0.5	*02/17/2017*	*Addition of new Release 2 states and creation of early relationship states*	*Stephen Cook*
0.6	*02/18/2017*	*Addition of final UML Diagram*	*Whole Team*
0.7	*02/20/2017*	*Addition of Nurse and Admin Sequence Diagrams*	*Oren Rosin*
0.8	*02/20/2017*	*Addition of Patient Sequence Diagram*	*Xixiao Yue*
0.9	*02/20/2017*	*Addition of Doctor Sequence Diagram*	*Chris Letourneau*
0.10	*02/23/2017*	*Edits to Design Rational (User Field)*	*Stephen Cook*
0.11	*02/25/2017*	*Edits to Design Rational (Activity Log and Appointment Calendar)*	*Stephen Cook*
0.12	*03/01/2017*	*Edits to Components to reflect Changes to design*	*Stephen Cook*
0.13	*03/05/2017*	*Update of Class Diagram to reflect changes*	*Stephen Cook*
1.0	*03/06/2017*	*Finalization for Release 1*	*Stephen Cook, Georgiy Rozenshteyn*
1.1	*03/26/2017*	*Update to UML Diagram and Components for Release 2*	*Stephen Cook*

1.2	*03/27/2017*	*Update to Design Rationale for R2 Planning*	*Georgiy Rozenshteyn*
1.3	*03/27/2017*	*Update to Patient Sequence Diagram*	*Xixiao Yue*
1.4	*03/27/2017*	*Update of Admin Sequence Diagram*	*Stephen Cook*
1.5	*03/27/2017*	*Update of Doctor Sequence Diagram*	*Chris Letourneau*
1.6	*03/28/2017*	*Update of Nurse Sequence Diagram*	*Oren Rosin*
1.7	*04/15/2017*	*Update of Components to reflect current design of product*	*Stephen Cook*
2.0	*04/26/2017*	*Addition of Message Design Rationale*	*Stephen Cook*
2.1	*04/28/2017*	*Additions to Design Rationale*	*Stephen Cook*
2.2	*05/03/2017*	*Additions to Design Rational*	*Stephen Cook*
2.3	*05/08/2017*	*Added Final UML and updated Components and Functions*	*Stephen Cook*
2.4	*05/08/2017*	*Finalize Design Document for R2*	*Georgiy Rozenshteyn*

Architectural Model

This diagram represents the major subsystems of the product. Initially focus on the domain layer and its components before decomposing the user interface component. Note that a common interface allows both the GUI and a Command Line Interface to access the domain model in the same manner without regard to the type of presentation technique.


Presentation Layer
Product Graphical User Interface (GUI)
Development Testing Interface (Test Driver)
Domain Model Interface
Domain Layer
Domain Model
Data

Components and Functions

*- indicates release two function

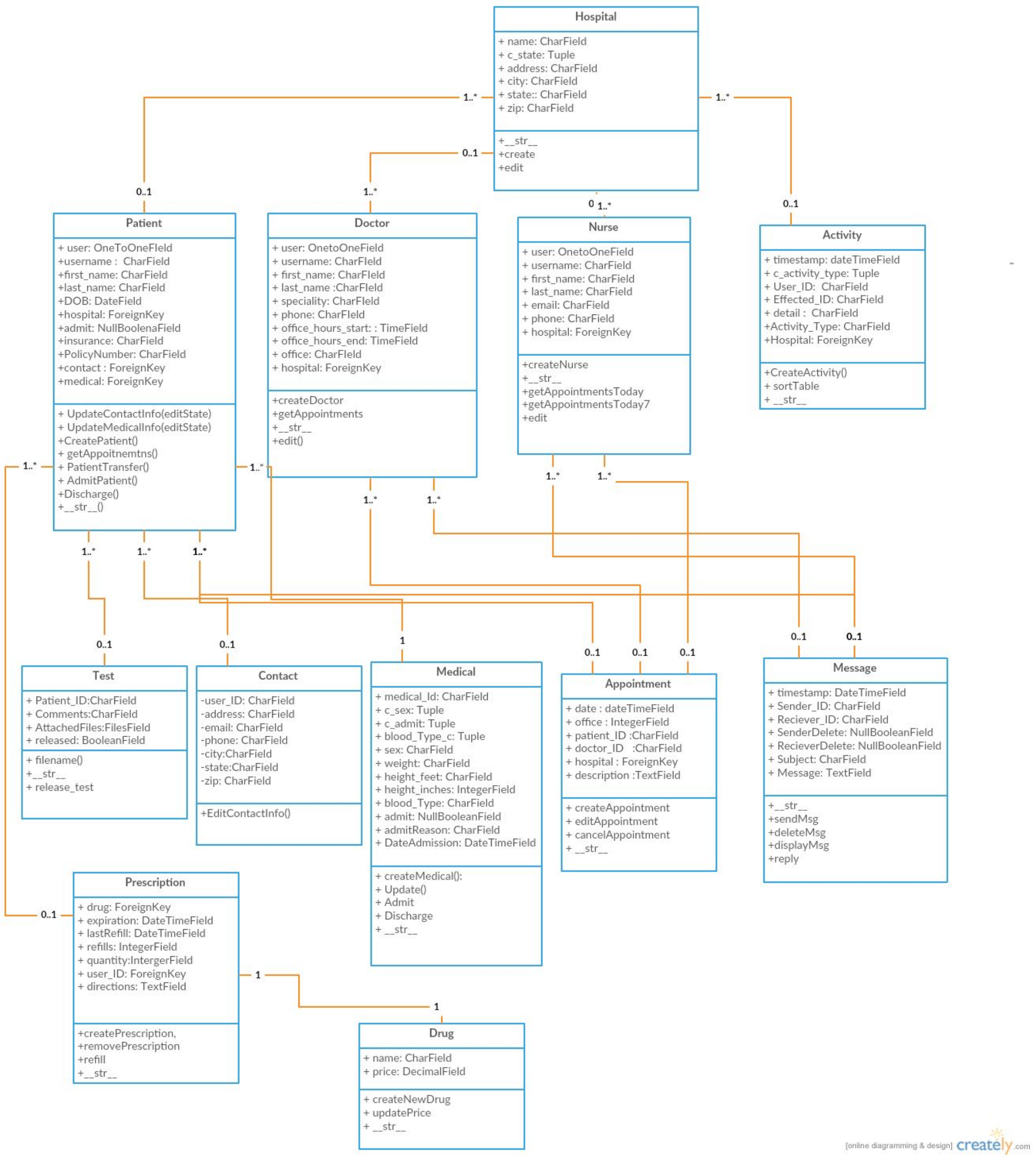
Appointment	Component state: ● date ● office ● patient_ID ● doctor_ID ● hospital ● description Component behavior: ● createAppointment() ● editAppointment() ● cancelAppointment() ● __str__
Patient Profile	Component state: ● user ● username ● first_name ● last_name ● hospital ● admit ● contact information ● medical information ● insurance ● PolicyNumber ● DOB Component behavior: ● __str__() ● createPatient() ● UpdateContact() ● UpdateMedical() ● getAppointments() ● PatientTransfer() ● AdmitPatient() ● Discharge()

Doctor Profile	Component state: ● user ● username ● first_name ● last_name ● speciality ● phone ● office_hours_start ● office_hours_end ● office ● hospital Component behavior: ● createDoctor ● getAppointments ● __str__ ● edit()
Nurse Profile	Component state: ● user ● username ● first_name ● last_name ● email ● phone ● hospital Component behavior: ● createNurse ● __str__ ● getAppointmentsToday ● getAppointmentsToday7 ● edit
Medical Information	Component state: ● medical_Id ● c_sex ● c_admit ● blood_Type_c ● sex ● weight ● height_feet ● height_inches ● blood_Type ● admit ● admitReason ● DateAdmission Component behavior ● createMedical(): ● Update() ● Admit ● Discharge ● __str__

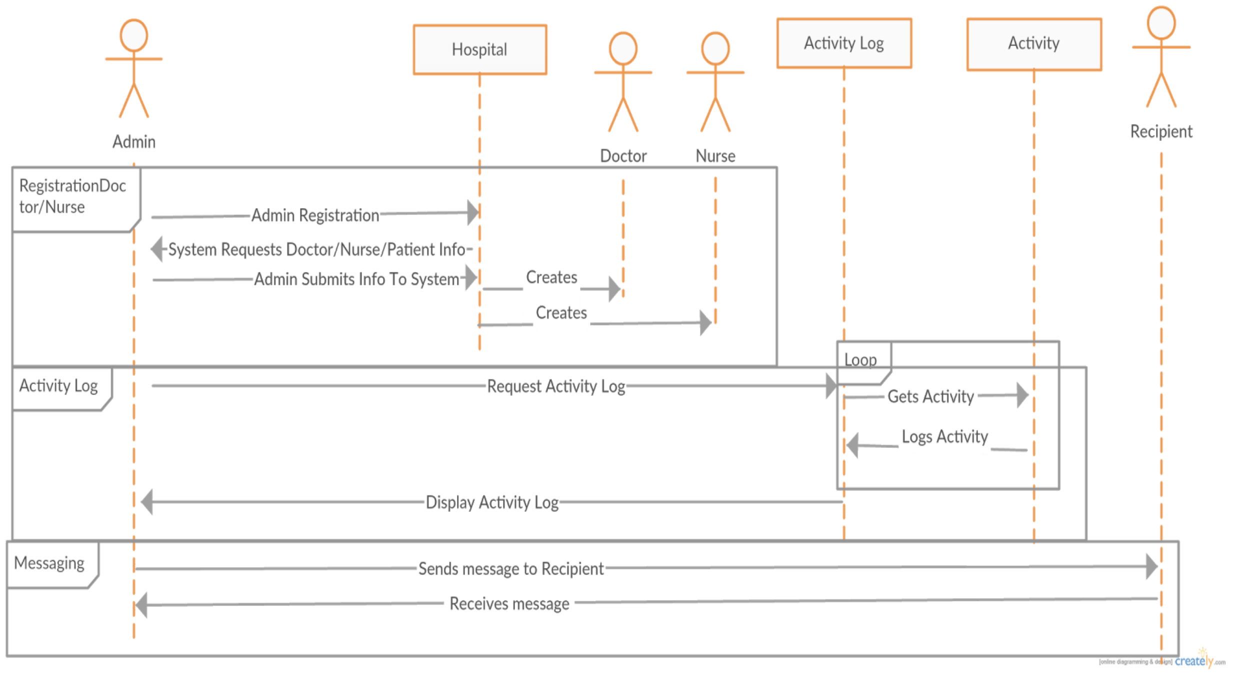
Activity	Component state: ● timestamp ● c_activity_type ● User_Id ● Effected_User ● Activity_Type ● Activity_Details ● Hopsital Component behavior: ● CreateActivity() ● sortTable() ● __str__()
Contact Information	Component state: ● patient_ID ● address ● email ● c_states ● phone ● city ● state ● zip Component behavior: ● createContact() ● editContact() ● __str__()
Prescription	Component state: ● drug ● expiration ● lastRefill ● refills ● quantity ● user_ID ● directions Component behavior: ● createPrescription, ● removePrescription ● refill ● __str__
Drug	Component state: ● name ● price Component behavior: ● createNewDrug ● updatePrice ● __str__
Hospital	Component state: ● name ● c_states ● state ● address ● city ● zip Component behaviors:

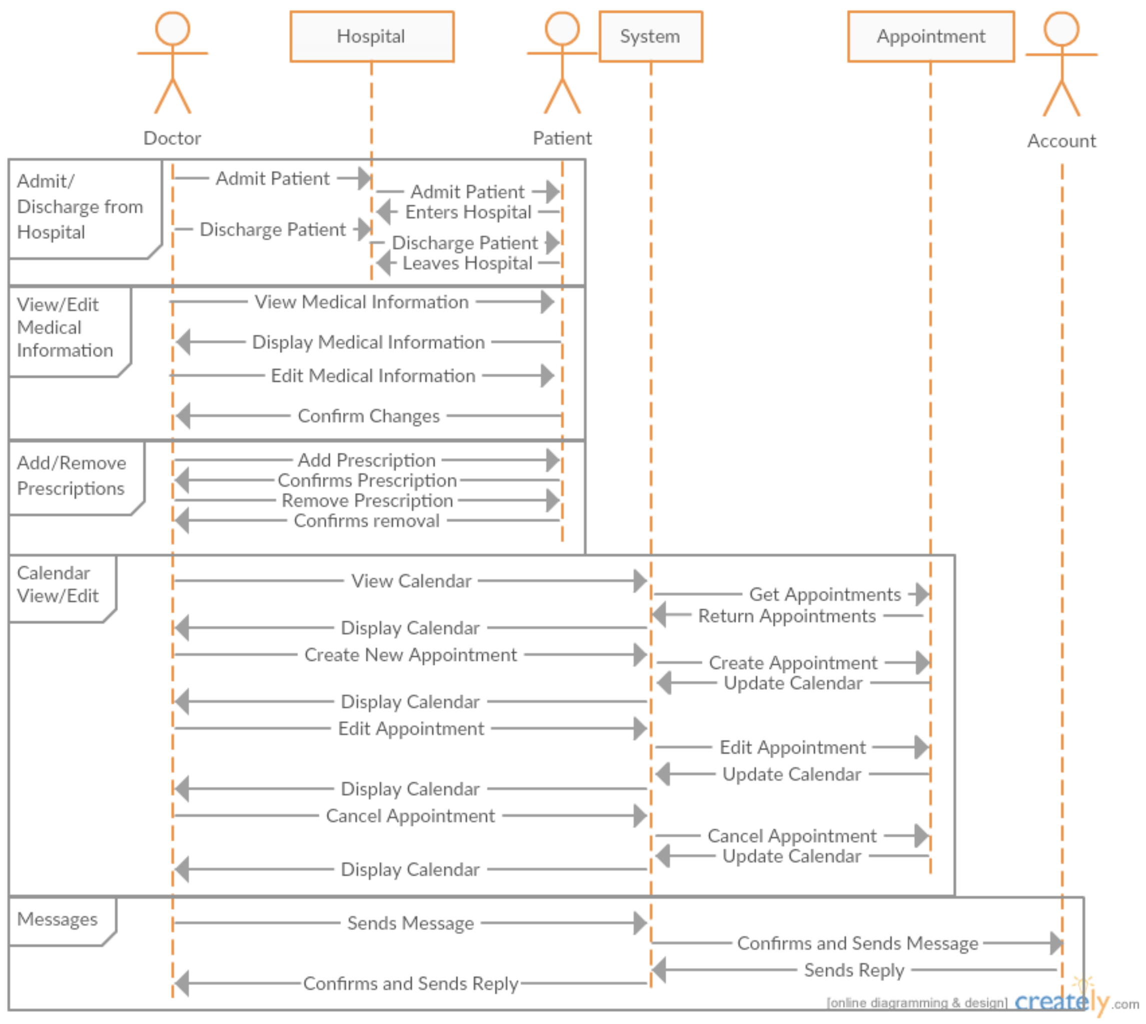
	• __str__() • create • edit
MedTest	Component state: • Patient_ID • Comments • AttachedFiles • released Component behaviors • filename() • __str__ • release_test()
Message	Component state: • timestamp • Sender_ID • Receiver_ID • SenderDelete • ReceiverDelete • Subject • Message Component behaviors: • __str__ • sendMsg • deleteMsg • displayMsg • reply

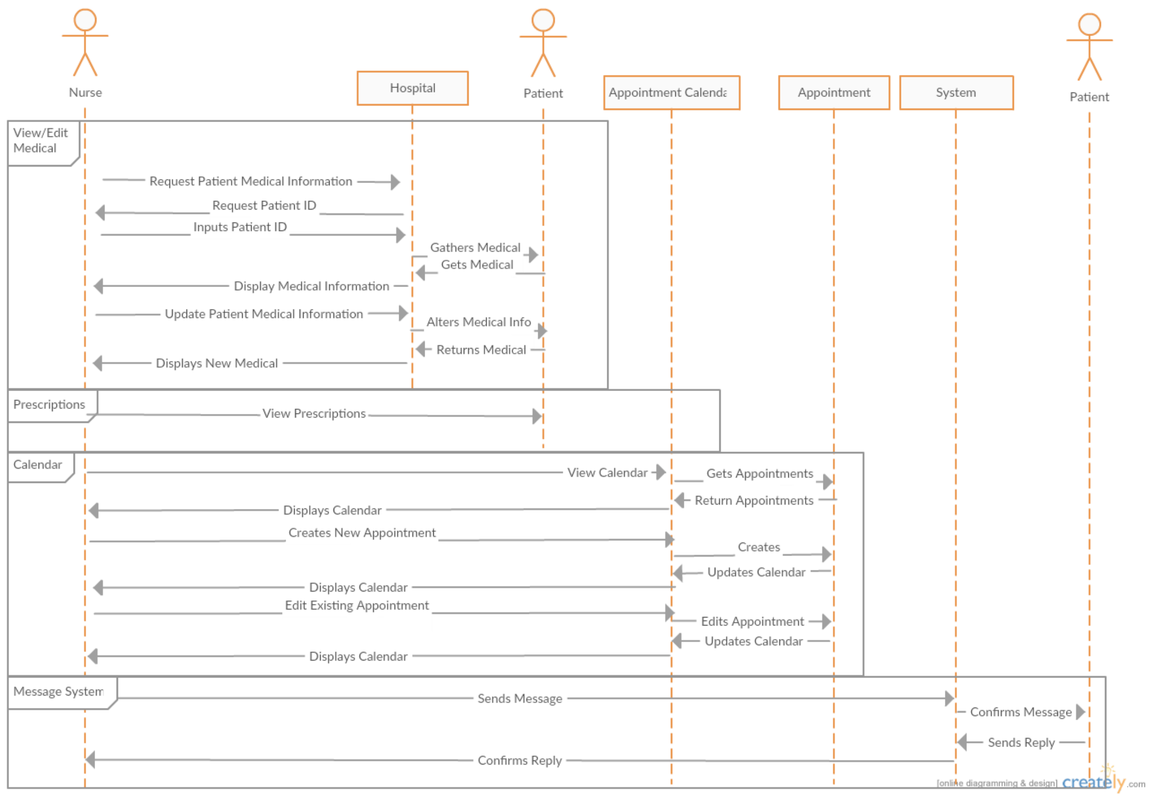
Class Diagram(s):

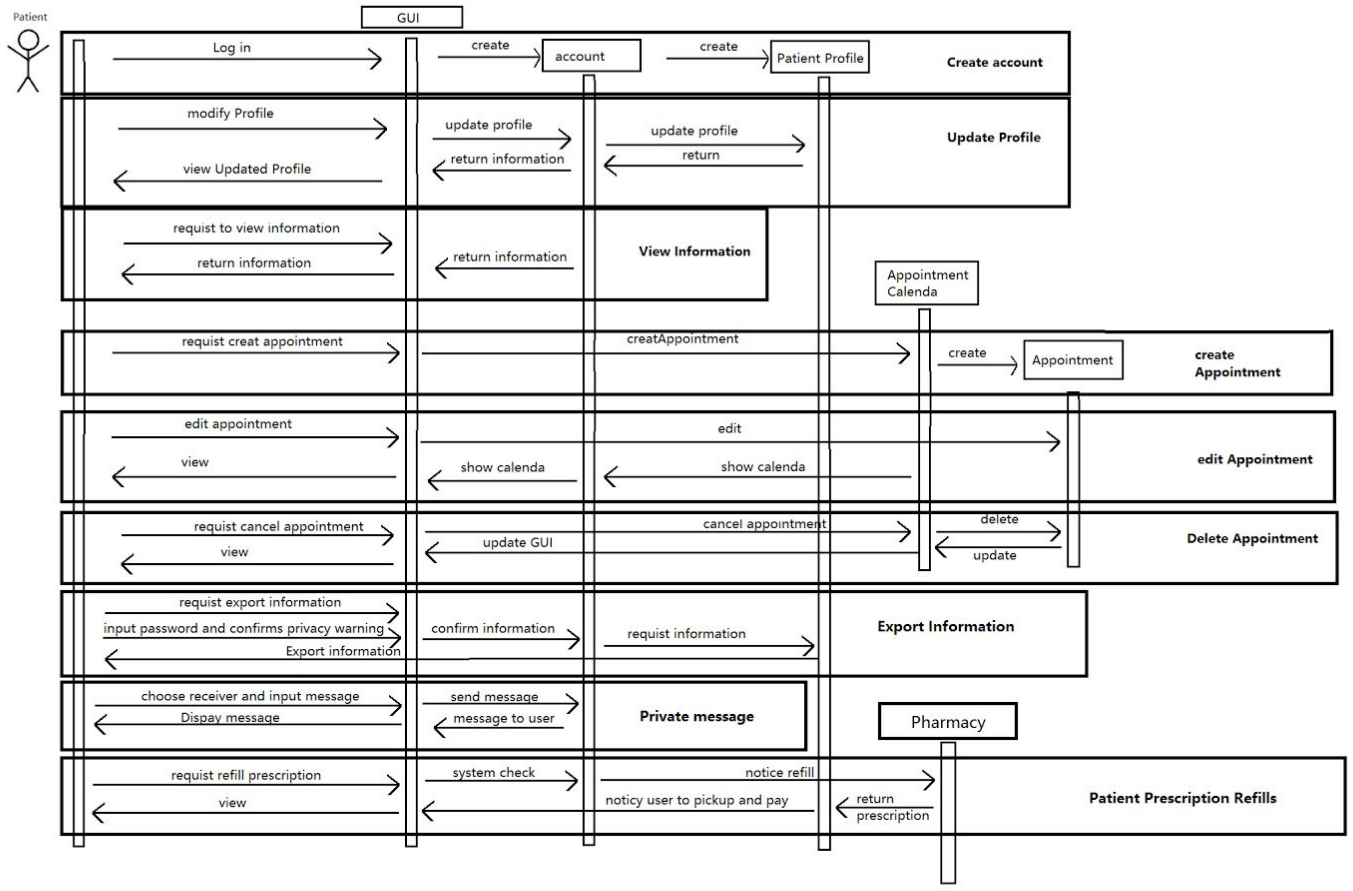


Sequence Diagram(s)




Doctor
Hospital
Patient
System
Appointment
Account
Admit/ Discharge from Hospital
Admit Patient
Admit Patient
Enters Hospital
Discharge Patient
Discharge Patient
Leaves Hospital
View/Edit Medical Information
View Medical Information
Display Medical Information
Edit Medical Information
Confirm Changes
Add/Remove Prescriptions
Add Prescription
Confirms Prescription
Remove Prescription
Confirms removal
Calendar View/Edit
View Calendar
Get Appointments
Return Appointments
Display Calendar
Create New Appointment
Create Appointment
Update Calendar
Display Calendar
Edit Appointment
Edit Appointment
Update Calendar
Display Calendar
Cancel Appointment
Cancel Appointment
Update Calendar
Display Calendar
Messages
Sends Message
Confirms and Sends Message
Sends Reply
Confirms and Sends Reply
[online diagramming & design] creately.com


Nurse
Hospital
Patient
Appointment Calenda
Appointment
System
Patient
View/Edit Medical
Request Patient Medical Information
Request Patient ID
Inputs Patient ID
Gathers Medical
Gets Medical
Display Medical Information
Update Patient Medical Information
Alters Medical Info
Returns Medical
Displays New Medical
Prescriptions
View Prescriptions
Calendar
View Calendar
Gets Appointments
Return Appointments
Displays Calendar
Creates New Appointment
Creates
Updates Calendar
Displays Calendar
Edit Existing Appointment
Edits Appointment
Updates Calendar
Displays Calendar
Message System
Sends Message
Confirms Message
Sends Reply
Confirms Reply
[online diagramming & design] creately.com



Design Rationale

- **Medical Information:** We decided to create a Medical Information class so that it is more efficient to edit Medical Information's states if we ever need to.
- **Contact Information:** We decided to create a Contact Information class so that we could

modify and view it with ease.

- **Appointment Calendar**: We decided to have both an Appointment Calendar and an Appointment class so that multiple of the same Appointment objects could be on different Calendars so that if it was edited on one Calendar it would be updated on all Calendars.
- **Prescriptions**: We decided to have a Prescription class because we wanted to be able to add any number of prescriptions per patient. The prescriptions themselves we anticipate to be drawn from a database.
- **Activity Log**: we decided to have both an Activity Log and an Activity class because we wanted to be able to have any number of activities in the log.
    - **resetLog(time)**: we wanted the Admin to have the ability to clear the log of all activities before a certain time.
- **Profile Superclass**: we discussed having a Profile Superclass but decided not to because each Profile types have very little in common with each other type
- **Use of User:** It was decided that if we associate a user to each class that would be the easiest way of login and validation
- **Elimination of Appointment Calendar:** After learning more about how Django and the databases work, it was deemed unnecessary to have an appointment calendar. Instead filtering appointments based on associated ID is preferred
- **Elimination of Activity Log class:** Very similar to Appointment Calendar, this class was deemed unnecessary due to how django interacts with databases
- **Message:** We decided to have a separate class for messaging so that we can include basic functionality such as sending a message, deleting a message, and replying to a message as well as setting a character limit and selecting a recipient.
- **Test:** In order to adhere to the requirement that tests and additional files (such as X-Rays) go through a review/release period, as well as allow for comments from the doctor/patient, we decided to create a separate class for medical tests.
- **Drug:** We decided to add a drug class so the admins could easily add new drugs as needed, and so each drug type could have a price associated with it.
- **Message Idea**: 1) user clicks inbox and system gets all messages where Recipient_ID = current_user. 2) Lists them sorted by date. 3) allows user to click the message and display the message details. 4) Options to delete the message and reply to message at bottom. Reply creates a form with pre-existing information such as Sender_ID and Recipient_ID.
- **Update Appointment Cut:** Since the system checks if an appointment exists already, we had issues with editing an appointment. The system would think the appointment already existed. To fix this we delete the appointment on submit and create a new one. This means that the edit appointment method is useless and will be cut.
- **Adding of Admit to Patient Atrb:** This lets us easier identify the users admitted to a hospital
- **Adding Username Atrb:** This was used to make filtering search results easier.
- **SenderDelete and RecieverDelete:** This was added to the Message Class to allow a user to remove a message from their outbox or inbox without removing it from the other person's inbox/outbox. This also allows us to delete the message when both users delete the message
- **Stats Rational:** The stats are calculated using the activity log, since all activities for statistics exist there. This will make sorting out information by date easier.