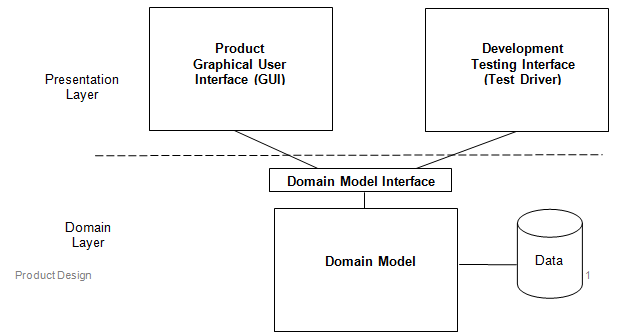
**Product Design**

|  |  |
| --- | --- |
| **Team** | Selfie |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Revision Number*** | ***Revision Date*** | ***Summary of Changes*** | ***Author(s)*** |
| 1 | Feb 21, 2015 | Document first filled out. | Everyone |
| 2 | 7 Mar 2015 | Updated the document based on Professor Martinez’s mycourses comments. | Mike |
|  |  |  |  |

# **Architectural Model**

This diagram represents the major subsystems of the product. The architectural model is primarily driven by the use cases. All tests collected and run are the result of the requirements gathering and they influence the architectural model as well. The entities within HealthNet itself are a high level specification and make no promises about specific interface, data, or component design.



|  |  |
| --- | --- |
|  |  |
|  |  |

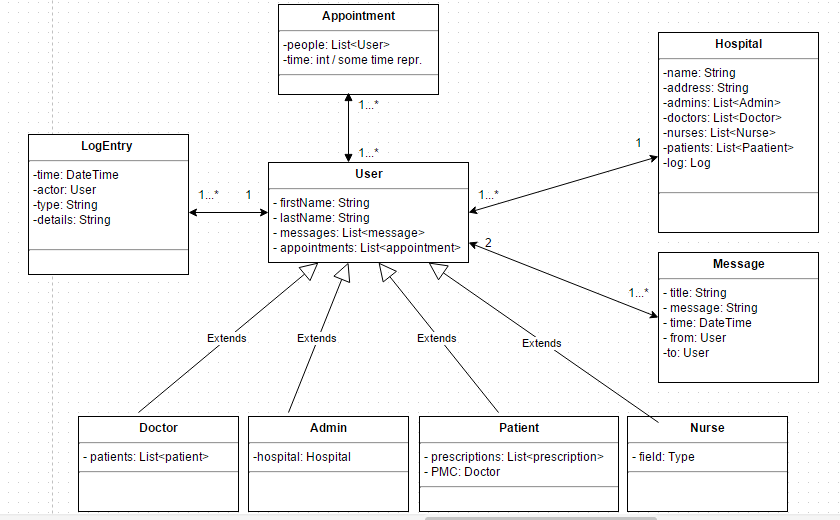
# **Components and Functions**

|  |  |
| --- | --- |
| Communication | Component State:   * This component enables users of the HealthNet system to communicate with each other through various services such as messaging and appointment scheduling.   Component Behavior:   * The messaging service can be used not only for user-user communication, but it can also be used as a method to notify users of new activity (i.e. a patient will receive a message when they have a confirmed appointment with their doctor). |
| Schedule | Component state:   * Holds all relevant upcoming information in a calendar that can be viewed by Doctors, Nurses, Patients, and Administrators.   Component behavior:   * The view of the calendar can be filtered by view type, i.e. day view, week view and month view. There’s a possibility that you can access options to modify, cancel and create events (appointments) from the calendar. |
| Activity Log | Component state:   * A log of all relevant recent system activity to keep track of changes within HealthNet to ensure security and tracking necessary changes.   Component behavior:   * The system will have a built-in method to log all relevant system activity occurring in HealthNet that we deem is important to log. Only administrators are allowed to view the activity log, so permissions need to be checked when the activity log is being accessed. Each log entry will have a variety of important information inside of it, for example, time, type, actor, etc. That way, especially for security purposes, we can provide ourselves with a plethora of information to sort out any issues and avoid conflict. |

|  |  |
| --- | --- |
| Users | Component State:   * Grouping of overall users of HealthNet. Keeps simple relevant information to identify the type of user and name of user inside the system.   Component Behavior:   * Overall superclass that keeps track of type of user, messages and appointments. This component is important because it allows us to have a way to keep track of every user in the system (or hospital) which will be useful down the road. Without this, it would be tough to keep track of all the users in a centralized place, which would decrease efficiency and cause problems down the road. |

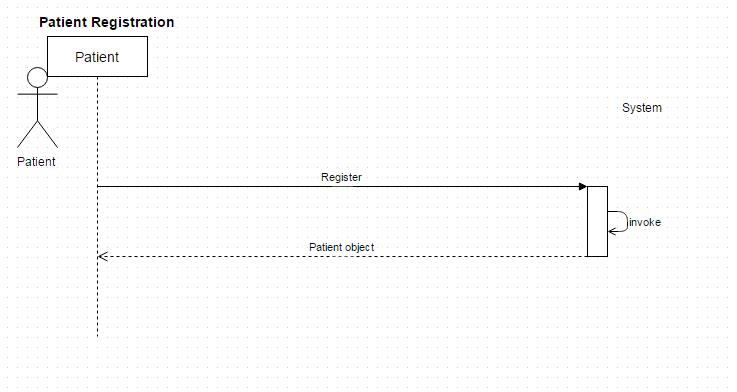
|  |  |
| --- | --- |
| Hospital | Component State:   * This component is basically a super object that tracks everything. It has the hospital information and access to the log and users. This makes it efficient because we know from talking to Dr. House and Professor Martinez that HealthNet will be the system for many hospitals so this increases database efficiency and easy understanding.   Component Behavior:   * The hospital will track it’s individual information to increase efficiency and have a centralized place with lists of all users. This allows us to have easy access to whatever we need to when we need to. This makes it easier so we’re never accessing the wrong hospitals information when we’re accessing specific users in specific systems. |

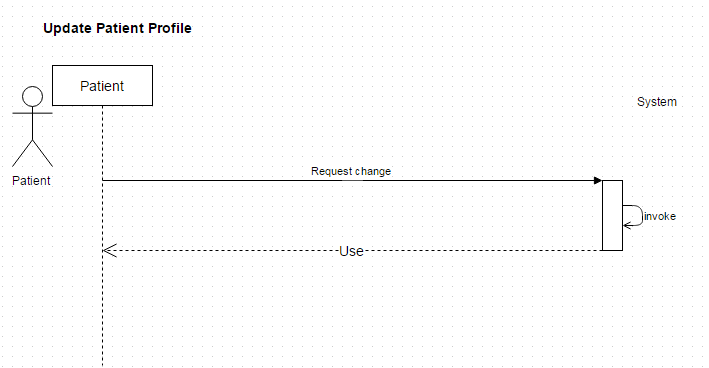
# **Class Diagram(s)**

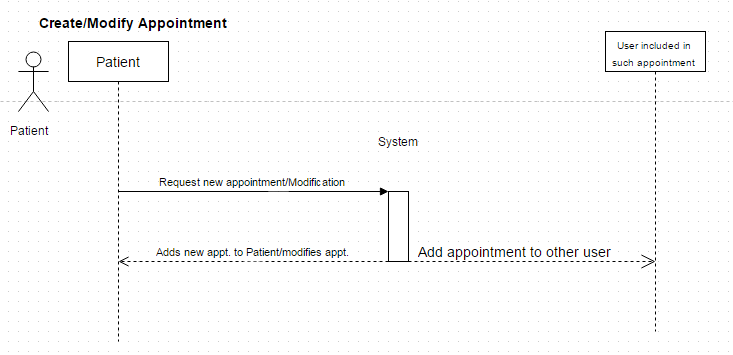
Methods (mostly setters and getters) are omitted for space.

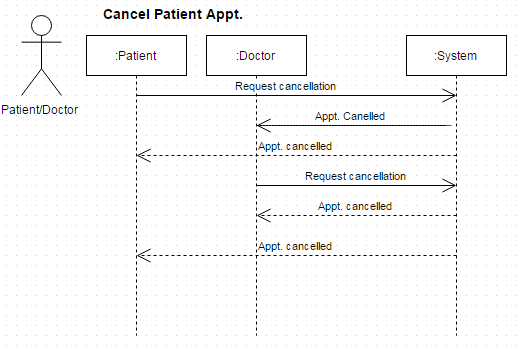
# **Sequence Diagram(s)**

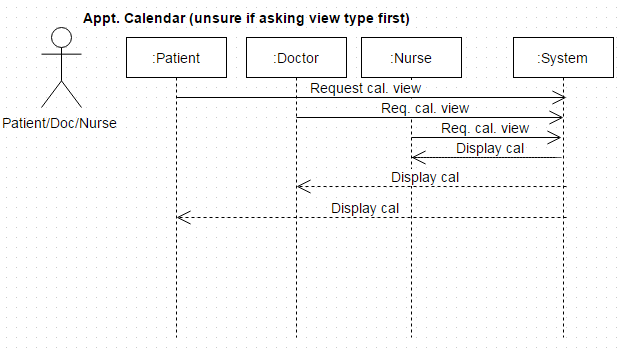
Disclaimer. The term system is used because as the product evolves it is unclear which specific parts of the system the actor is actually interacting with.

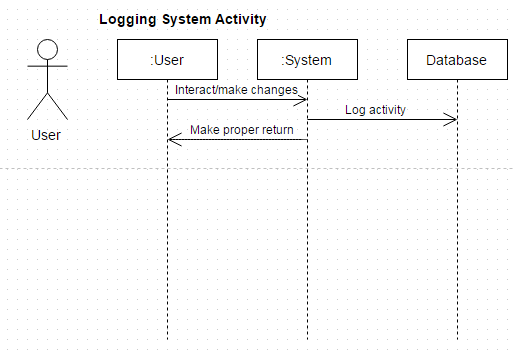


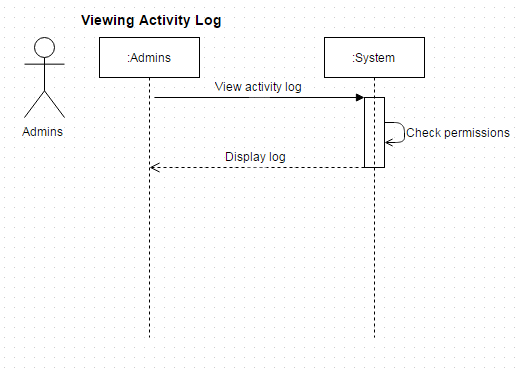


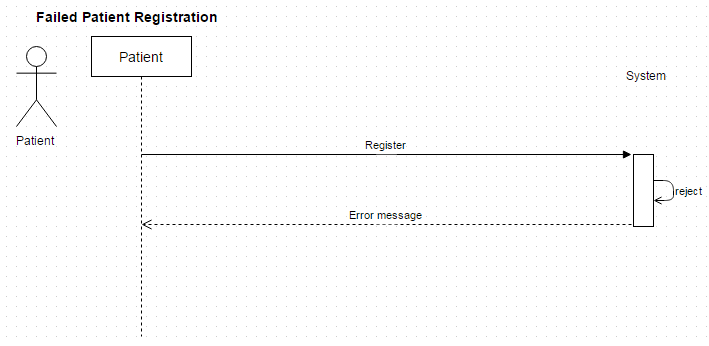


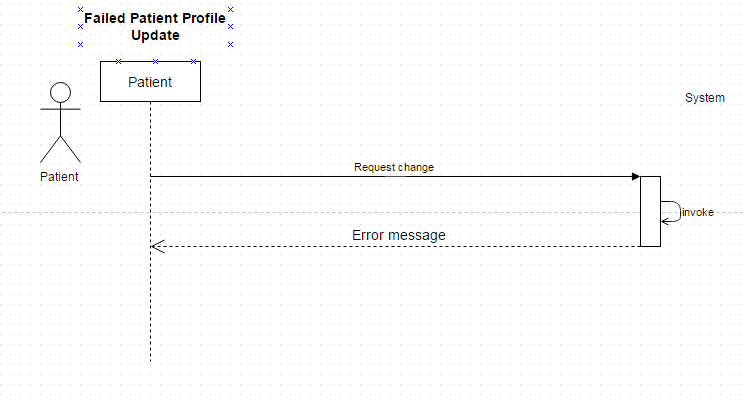


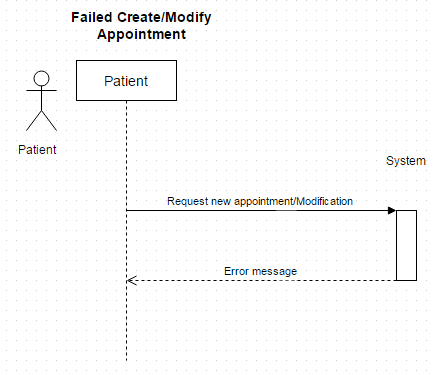


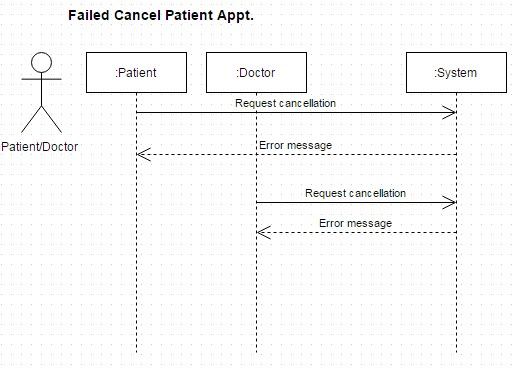


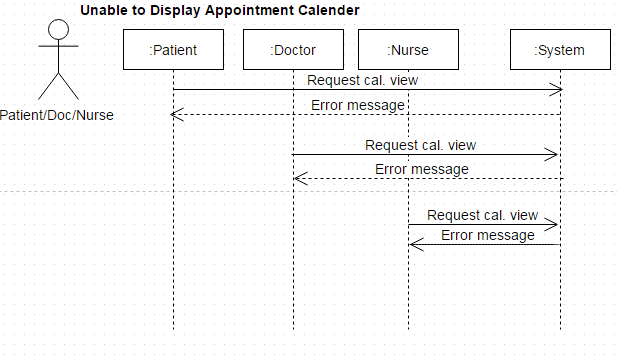


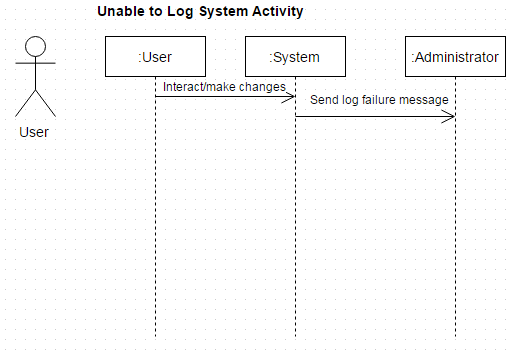


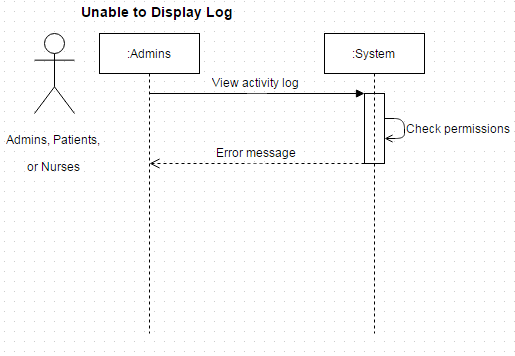












# **Design Rationale**

Other than what is listed here, everything was mutually agreed upon with no dissent.

|  |  |  |
| --- | --- | --- |
| **Choices Chosen** | **Alternatives Forsaken** | **Why** |
| Messages to be a separate object | Having messages be string | Too much info associated with a message. |
| Types of users extend the user class | Having an attribute in the user class that specifies what type it is. | Many different abilities and attributes are associated with different kinds of users. Also, it models the real world better. |
| (Still under debate)  Hospitals should have different lists for each type of user. | Hospitals should contain one list that has all users. | To be able to send messages to all of one type of user |