**Product Design**

|  |  |
| --- | --- |
| **Team** | **Group 4, team Unchained** |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Revision Number*** | ***Revision Date*** | ***Summary of Changes*** | ***Author(s)*** |
| 0.1 | 02/14/2017 | Created Class Diagrams | Unchained |
| 0.2 | 02/15/2017 | Components and Sequence Diagram Created | Unchained |
| 1.0 | 2/18/2017 | Finalized design diagram | Unchained |

# **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.

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

**Components and Functions**

|  |  |
| --- | --- |
| Registration | * State   + Registered Users * Behavior   + Create new user   + Confirm user exists   + Allow access to user’s personal information |
| Patient Information | * State   + Patient Medical Information   + Patient Personal Information   + Test Results   + Prescriptions * Behavior   + Update patient medical information   + Update patient personal information   + Export Information   + Add and remove prescriptions   + Add patient tests   + Release patient tests   + Viewing Patient Medical Information, Prescriptions and Tests and Results |
| Appointment Calendar | * State   + Appointment list   + Appointment schedule   + Doctors and employees present in hospital on day of appointment * Behavior   + Create appointments   + Update appointments   + Cancel appointments   + Access appointments |
| Hospital Management | * State   + Patients admitted   + Doctors at hospital   + Nurses at hospitals * Behavior   + Admitting Patient   + Tracking employes at hospital |
| Admin | * State   + System Logs   + System Statistics * Behavior   + Access system logs   + Analyze logs into statistics |
| Messaging | * State   + Messages received   + Messages sent * Behavior   + Send Message   + Receive Message |

# **Class Diagram(s)**

# 2017-03-07 (1).png

# 

# 

# 

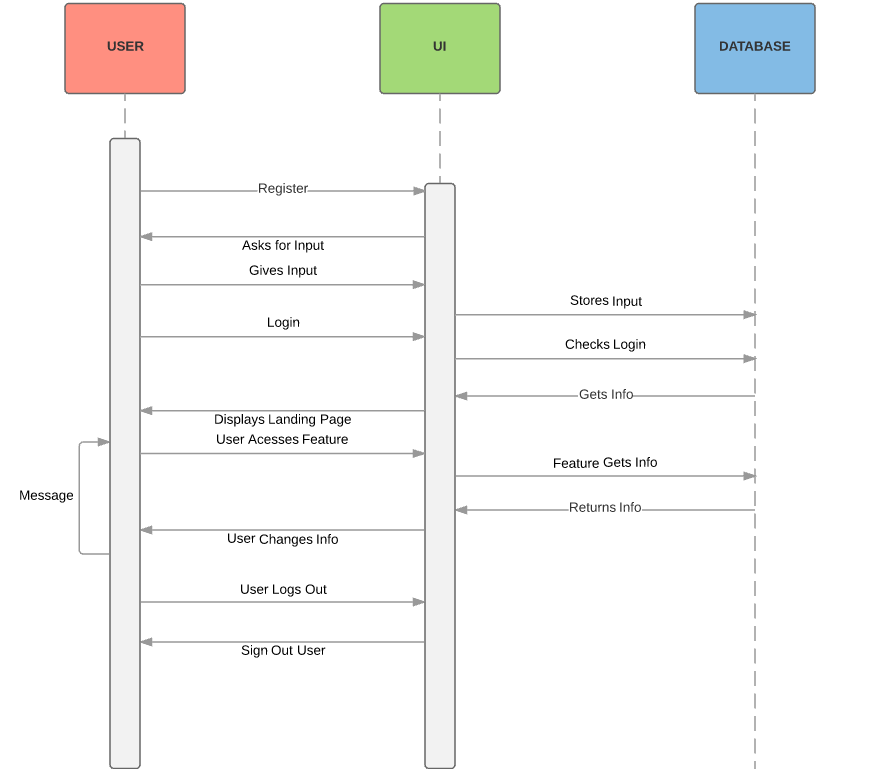
# 

# 

# 

# 

# **Sequence Diagram(s)**



# **Design Rationale**

**What alternatives did you consider?**

1. Having Doctor filling out a test form then specifying the patient vs specifying the patient then filling out a test form
2. The logs will be recorded after the action being logged is complete
3. Doctor and nurse having different landing pages
4. Use Django messages or create our own

**What are the strengths (and deficiencies) of the final design compared to the other alternatives considered?**

1. Many of our decisions were based off of usability for the user. For example the decision for the doctor to specify the patient first was so that they would have to go to the patients page they are working on to upload their test results. This is better because the doctor now has all the functions available for that patient in front of him. It way be a little more work for the doctor but saves time in the long run.
2. Having the events logged after completion allows for more a more precise list when going through the logs. It also makes sure that anyone not completing a registration doesn’t get logged so we can never get useless clutter.
3. Nurses and doctors have different abilities that each can do, however since they share many similar traits we decided for them to share the same landing page but they each will only have access to the parts of that page they have clearance for.
4. Using Django libraries would give us a wide range of already implemented functions however we’d be restricted to their abilities. Creating our own from scratch would give us a wider range of possibility but we would have to implement each feature our self and there is greater room for error.

**Why did you select the approach you finally chose?**

1. It was better for usability.
2. Prevents a log event being created if someone never completes a registration.
3. It’s more efficient use of code for nurses to just not be able to use doctor related functions rather than have a new page for themselves.
4. We decided to create our own because the wide range of functionality that we get out weighs the chance we may have more bugs.