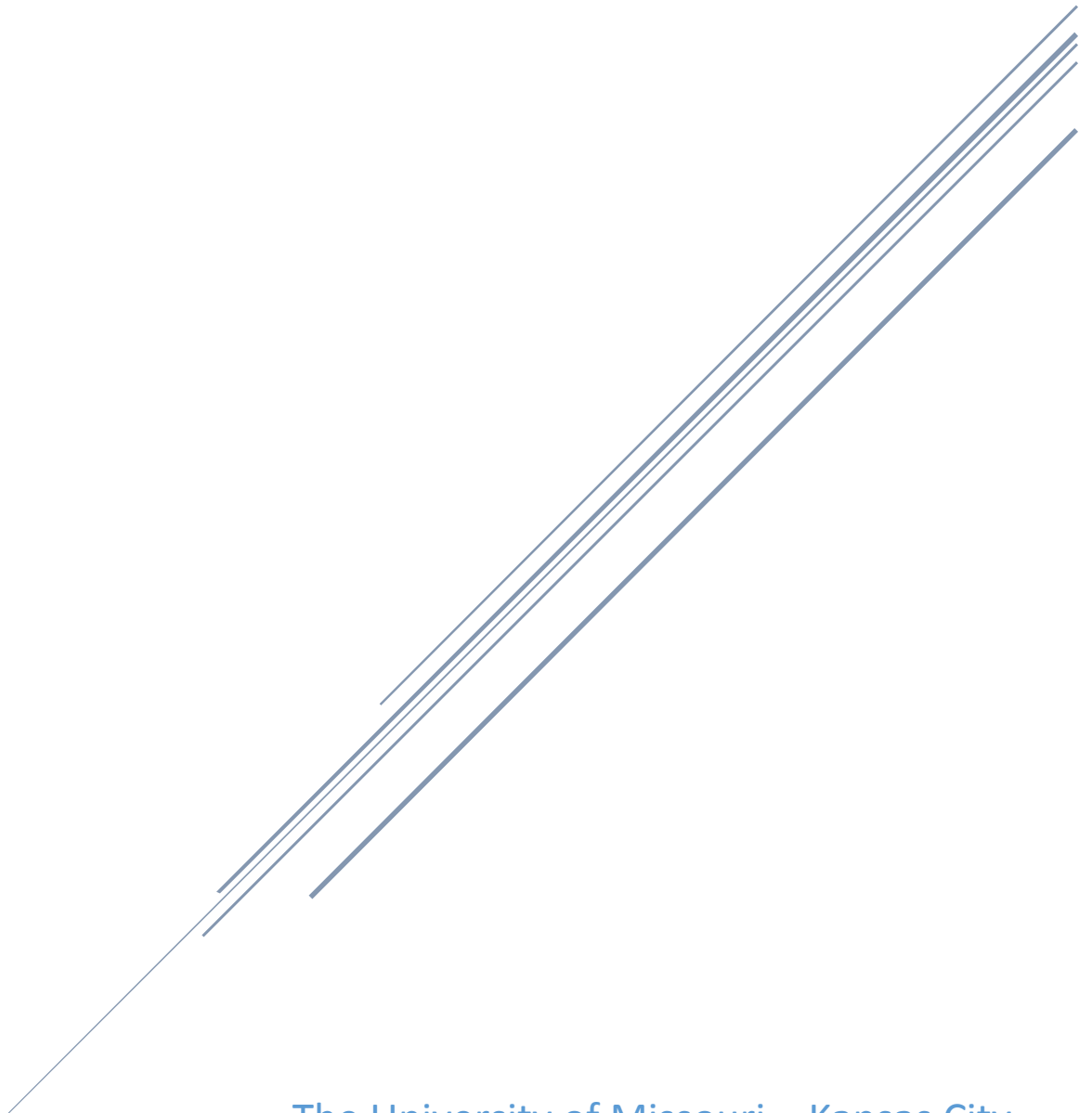


HEAL ASSIST: ITERATION 1

Hayden McParlane, Priyanka Bala and Komali Ghanta



The University of Missouri – Kansas City
CS 551 : Advanced Software Engineering

❖ Imported Services / APIs

Our system relies upon a couple of different external APIs, and the list will continue to grow in minor ways due to the applications size. In our current design, we're using the Drchrono API in order to allow for access to the Electronic Medical Record (EMR) and various other desirable features such as appointment scheduling. The Drchrono API requires use of the OAuth 2.0 protocol. As a result, we're looking into OAuth 2.0 libraries for use in our application. One that seems desirable is the Apache Oltu project. Stemming from use of Apache Oltu, as well as any alternative that was found for OAuth implementation, we're also going to use the Eclipse Development Environment for Java EE Developers along with Maven extensions and the Maven local and central repositories (the Apache Oltu project is located in the central repository). Mavens project management features seem to offset the learning curve required to use it regardless the need for the OAuth API. We're also utilizing the Metronic theme template for front-end user interface design, allowing for rich user interface design in minimal time.

❖ Detailed Design of Application Services

User Stories

The following are the user stories that we've developed thus far:

- 1) As a patient, I should be able to access the specific prescription medications that I'm taking as well as details related to that medication.
- 2) As a patient, I should be able to access physician treatment directives so that I can better maintain my health.
- 3) As a patient, I should be able to quickly contact my doctor's office so that I can efficiently make doctor's appointments and call my doctor about questions I may have.
- 4) As a parent, I should be able to access information related to the care of my child so that I can care for all my dependents and myself from the same account.
- 5) As a user (any user), I will need to log into the application via some authentication system in order to gain access to my account.
- 6) As a patient, I want to ensure that only physicians have access to my data and, therefore, that there is some sort of authorization system so that I can be as open and honest as I need to be without fear of social repercussions.
- 7) As a patient, I want to be able to view the conditions that I'm battling as well as information related to that condition.

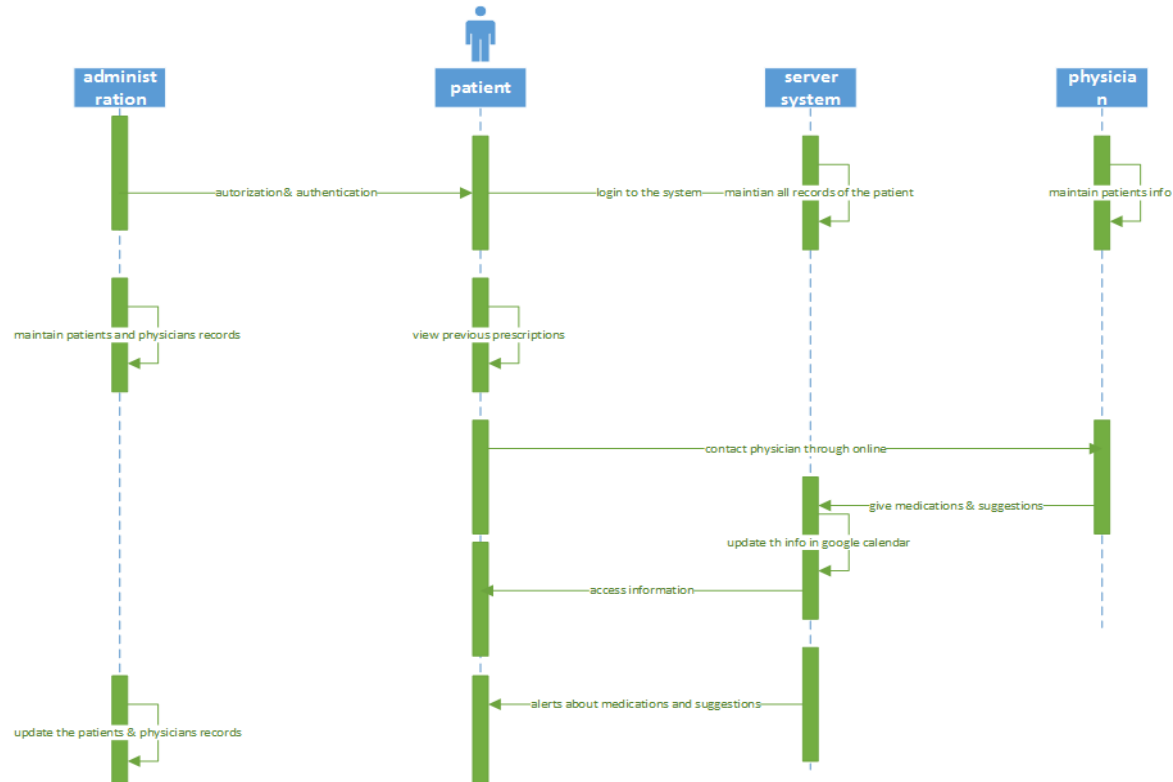
- 8) As a patient, I would like certain aspects of my treatment regimen to automatically populate my calendar if needed so that I can better ensure I keep healthy with minimal effort spent messing around in the application.
- 9) As a patient, I would like updates to my medical information to quickly synchronize with my application so that I can remain as vigilant as possible in maintaining my health.
- 10) As a parent, I would like an easy to use interface that allows me to access information pertaining to my child's health in as simple a manner as possible.
- 11) As a doctor, I should be able to view my patient's full chart on the application so that I can update the application as I interact with the patient and its records can remain as accurate as possible.
- 12) As a doctor, I should be able to view a patient's full history on the application and revert the chart to previous versions so that total information validity is ensured.
- 13) As a doctor, I should be able to access a list of prescription medications that contain information related to the medication so that I can keep up to date and refresh my memory.
- 14) As a doctor, I should be able to filter prescription medications based on various criteria (such as condition, etc) so that I can locate the desired medication information quickly.
- 15) As a doctor, I should be able to contact a patient by phone quickly and easily if the need arises so that, in emergency situations, patient contact is as easy as possible.
- 16) As a doctor, I should be able to update advice previously given to patients from the application, as well as medications recommended, prescribed, doses, etc. so that patient care remains as effective and efficient as possible.
- 17) As a doctor, I should be able to monitor a patient's adherence to the treatment regimen I've specified for them in an efficient and simple manner.
- 18) As a doctor, I should be able to access treatment information related to a particular condition directly from the device and, ideally, that information should be trustworthy (not from the internet) so that I can easily double check my treatment regimens, etc.
- 19) As a doctor, I should be able to sort patients based on various criteria so as to quicken the search process.

❖ Service Description

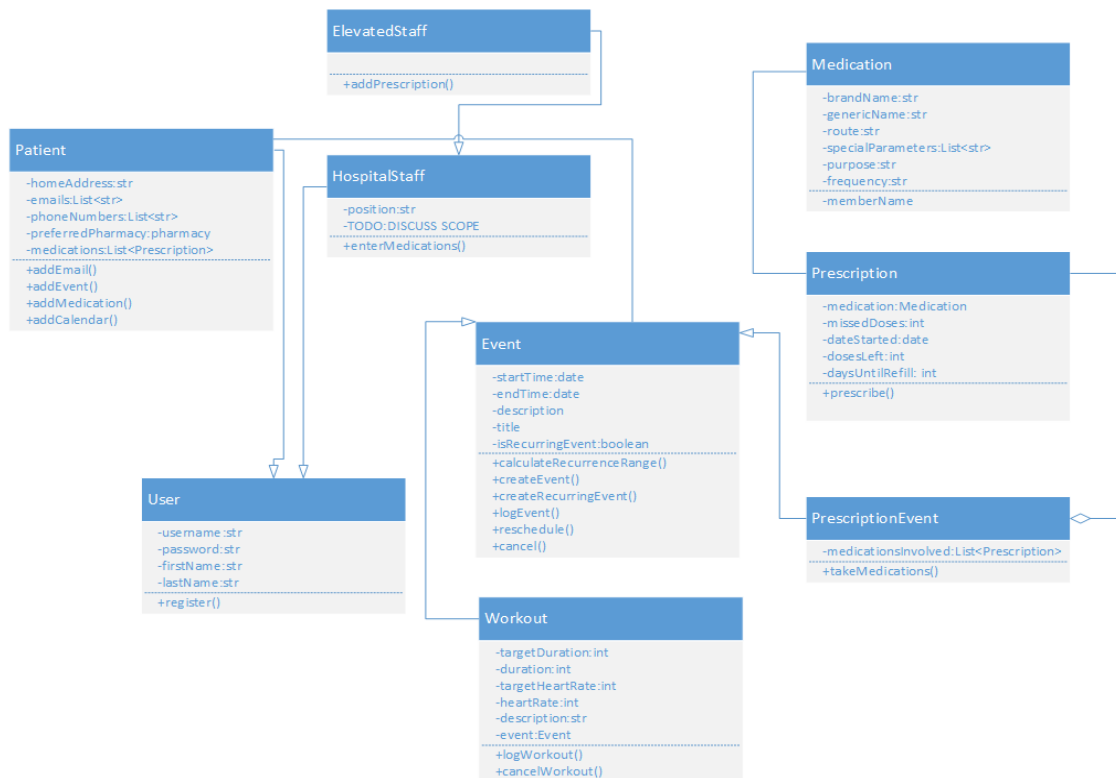
The drive to develop HealAssist comes in large part from the safety and security one would feel having one's own health data accessible on one's mobile device. Our application is intended to allow patients to locate helpful information related to their ailments. For example,

simple things such as the dosage of one's medication, physical appearance of a prescription pill or liquid, frequency of administration of the medication and support groups for various medical conditions are a few of the features that we plan on including in our application. Furthermore, we're designing the system such that it can be extended to include Electronic Medical Records (EMR) other than Drchrono by use of the abstract factory pattern. This means, once physicians start to pick up on the value of linking patients to select aspects of their medical record, APIs other than Drchrono will be usable with our system. This is beneficial because it'll allow parents to monitor the health of their children by linking their HealAssist account to additional EMRs. In short, the service we're offering allows one to have greater control over their health.

System-level Sequence Diagram



System-level Class Diagram



❖ Implementation Details

Implementation of REST Services.

Our implementation of REST services will be achieved through use of a dynamic web application within Apache Maven developed through use of the Eclipse IDE. Our front-end design will consist of customized JavaScript sending REST resource requests to our servlet present on the server. For diagrams detailing system design see the report section.

❖ Deployment Sites

ScrumDo. <https://www.scrumdo.com/organization/healassist/dashboard#>

GitHub. <https://github.com/HealAssist/HealAssist>

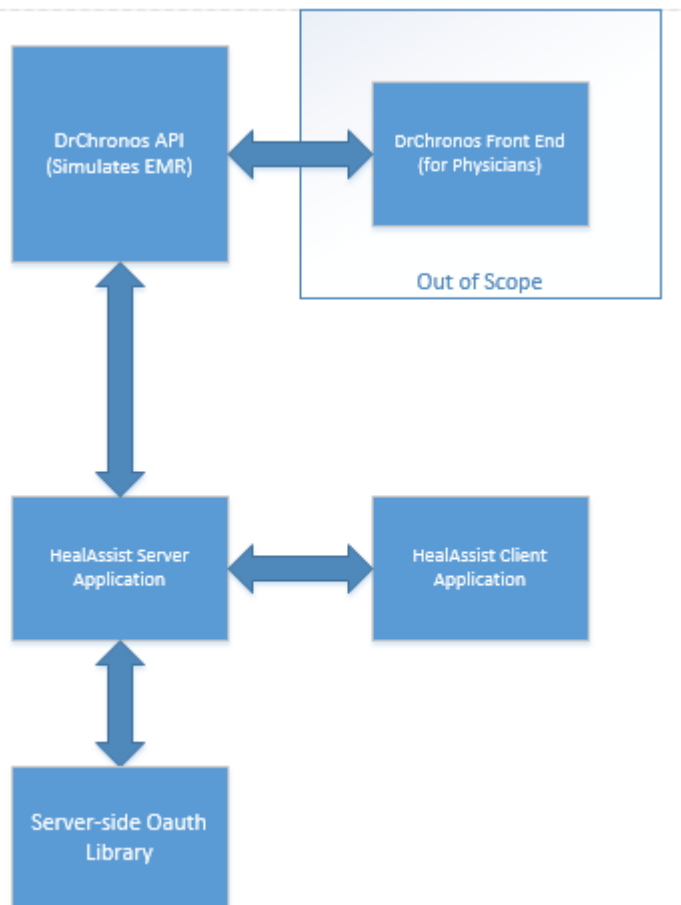
❖ System Report

In its most general form this project is a client-server application that will use Twitter Bootstrap, JQuery and JavaScript at its front-end and a servlet on its back-end. Use of the Metronic theme template will beautify our user interface using Bootstrap and JQuery, which will be integrated into the whole using custom-written JavaScript. Communication with our servlet will take place by use of REST. In other words, URLs will be sent to our servers address requesting access to resources residing on our server. While the front-end will be relatively simple, the back-end will be far more complex.

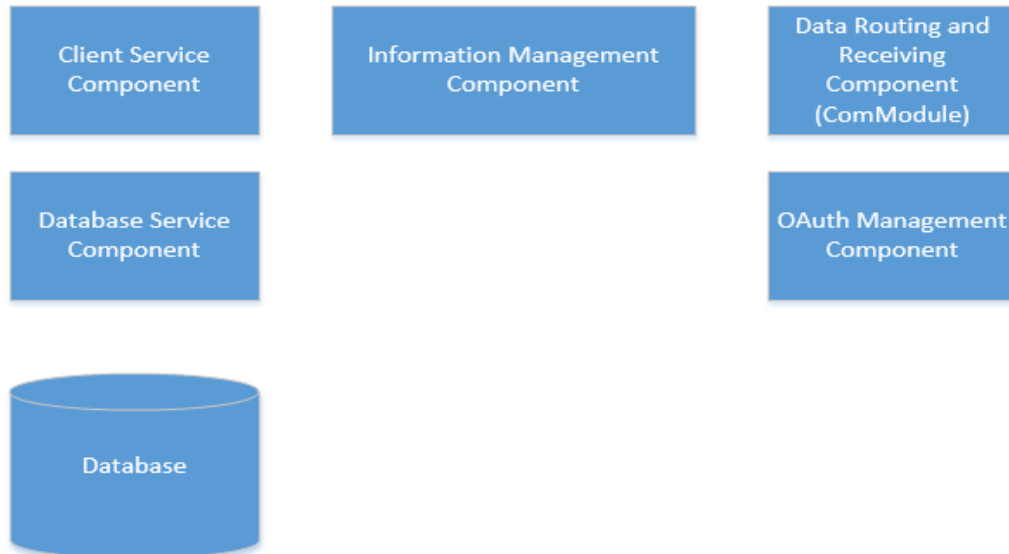
Our back-end system, as it currently stands, will consist of multiple high-level components. The *Client Service Component* will be used as the interface by which clients can retrieve their data. Data manipulation and access will be controlled by a *Data Service Component*, which will utilize the adapter pattern to decouple the data access interface from the underlying data technology implemented. Due to the important need for information to be available and consistent in health applications, our server will have on it an *Information Management Component* which will manage information-related tasks, including pulling data from the EMR at certain intervals to ensure data consistency between the remote EMR and the servers internal storage. Assisting communication with external APIs that rely upon the OAuth authentication protocol, an *OAuth Management Component* will be designed. As extraneous as this seems, this is necessary because communication needs to occur between the Googles OAuth servers and our server during the authentication process. Along with refreshing the access tokens granted to our

server, the OAuth Management Component allows for increased cohesiveness of components and decreased coupling. Furthermore, OAuth is becoming increasingly used to allow for cross-application communication within many APIs, so the presence of a full-blown system component dedicated to the protocol seems necessary given the likelihood of using this client-server application with additional EMR APIs. The final component that is currently known is a *Data Routing and Retrieval Component* taking the responsibility of communicating with external data sources and retrieving data of multiple formats. Multi-format data retrieval will be achieved by use of abstract factories, as shown in the figures below. This is our design in its current state. However, as with most software projects, certain aspects will change. Despite that volatility, one thing is of central concern; we want the system to be as extensible and modular as possible, decreasing the likelihood of dependencies on external APIs, specific database technologies, what have you from crippling our system later on during development.

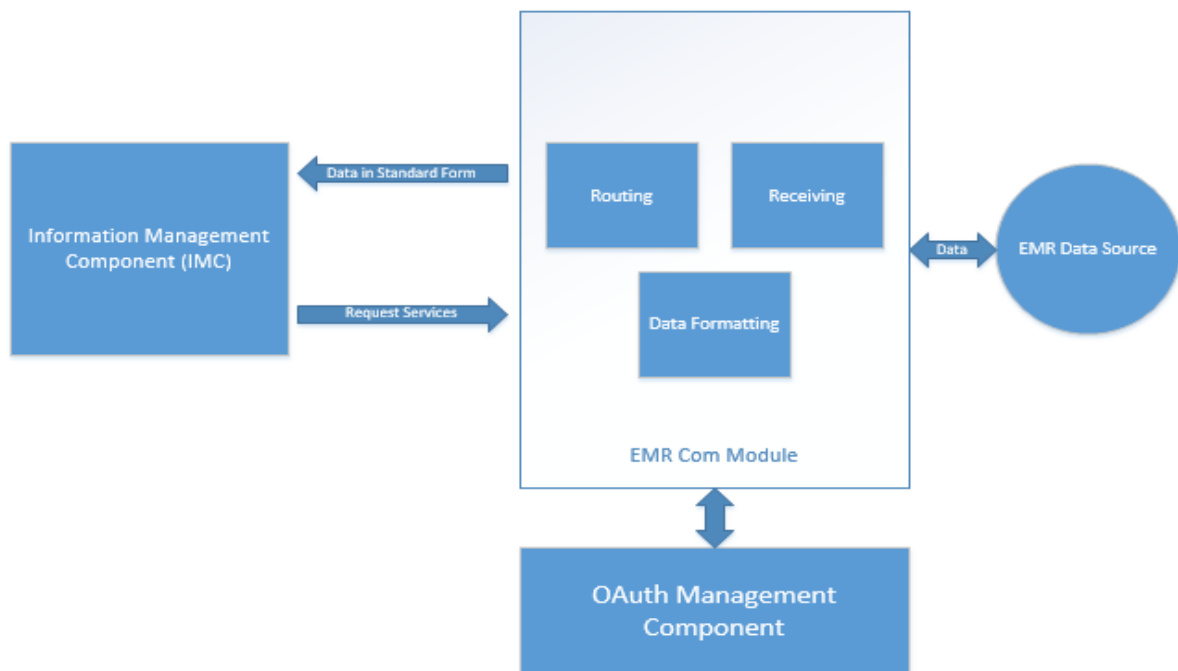
High-level System Architecture



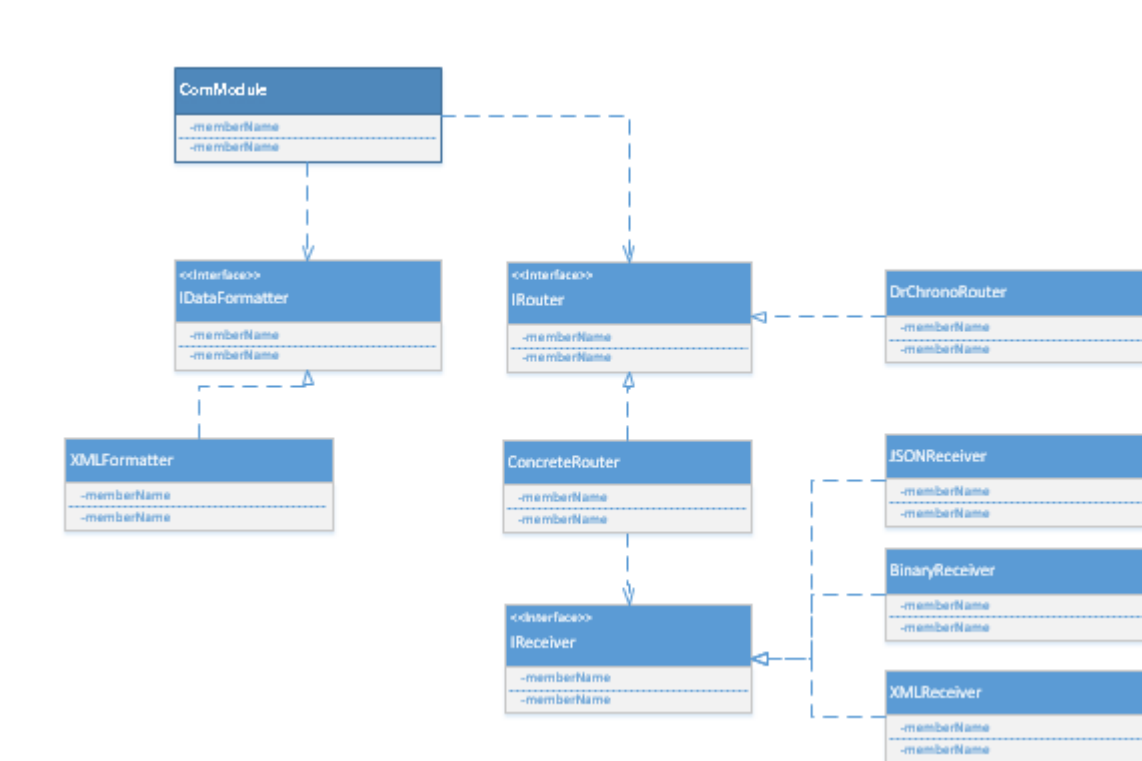
High-level Server Architecture (Additions will be made)



Data Routing and Retrieval Diagram



Routing and Data Retrieval (ComModule) Class Diagram





Current Front-end UI Screen Shots

1. Login Page for user

HEALASSIST

Login to your account

 Username

 Password

☐ Remember me

Login ➔

Forgot your password ?
no worries, click [here](#) to reset your password.

Don't have an account yet ? [Create an account](#)

2015 © Heal Assist

2. Register Page for user

HEALASSIST

Sign Up

Enter your personal details below:

Enter your account details below:

Enter your account details below:


☐ I agree to the [Terms of Service](#) and [Privacy Policy](#)

2015 © Heal Assist


3. Registration Page Input Validation

Sign Up


Enter your personal details below:

 Full Name


This field is required.

 Email


This field is required.


 Phone no

This field is required.


 -Select-

This field is required.


 -Select-

 Age

This field is required.


 Address


This field is required.


 City/Town


This field is required.


4. Patient Information


HEALASSIST 


Priyanka 

 Dashboard

 Patient Management <

 New Patient

 View Patient

 Messages <

New Patient

PATIENT DETAILS

Patient Personal Details

Full Name	<input type="text" value="Full Name"/>	E-mail	<input type="text" value="E-mail Address"/>
Gender	<input type="text" value="Male"/>	Phone No	<input type="text" value="Phone No"/>
Age	<input type="text" value="Age"/>	Address	<input type="text" value="Address"/>
City / Town	<input type="text" value="City / Town"/>	Country	<input type="text" value=""/>
Pincode	<input type="text" value="Pincode"/>	User Name	<input type="text" value="User name"/>
Password	<input type="text" value="Password"/>	Retype Password	<input type="text" value="Retype Password"/>

Patient Health Details

5. Patient Search

HEALASSIST

Priyanka

Dashboard

Patient Management

New Patient

View Patient

Messages

SEARCH PATIENT

SEARCH PATIENT

Patient Name Search Name

Gender Male

Problem Patient Problem

E-mail E-mail Address

Phone No Phone No

User Name User name

Search Cancel

❖ Project Management

Implementation Status Report

In our project HEAL-ASSIST we have a large learning curve. So we researched the different technologies, databases, APIs and Drchrono EMR API, PillFill Developer API and the Chrono library for calendar access in java. Komali and Priyanka headed studies of MongoDB and noticed that join operation wasn't supported within MongoDB itself. Although MongoDB didn't support direct joins, but we can perform joins by performing mapreduce operations for each collection. But performing mapreduce on each collection will be tough and won't be as accurate as we need. Additionally, with our current time constraints and the size of our project, it seemed better to rely on past experience. As a result, we modified our decision to use MongoDB and decided it would be better to use a relational database instead. Despite this choice, we're going to use the adapter pattern and allow our system to be modifiable if MongoDB turns out to be desirable later on.

Though we took more time learning databases, APIs and other technologies, we now have a better understanding of the project and the tasks that need to be done individually. We created some tables in the database for hospital management, patients and physicians, performed query operations using Oracle, created our website HealAssist by using the Eclipse IDE. The user can login to the hospital management system and if the user doesn't have an account then he can register for one. Internally we are maintaining the users registered information as their user IDs and passwords in the database.

Moving into the first iteration, the vast majority of our design considerations such as API research, technology research (i.e, OAuth, Maven, Drchronos, etc) have been completed or are in a sufficient enough state that further research will take far less time. As a result, our first iteration deliverable will include a huge amount of development with a solid guide as is visible in our class and high-level component diagrams above.

❖ **Work completed**

Description

- Designed the database schema and created the tables.
- Designed UI screens.
- Setting up environments.
- Analyzed the template and understood different components.
- Customized and developed login, registration, view patient, search patient screens.
- Stored registration details and authenticated user entering the system.

Individual Contributions

- Priyanka – Primary front-end developer, user interface development, prototype back-end development using a J2EE dynamic web project, MongoDB research and implementation, Relational Database implementation.
- Hayden – Primary software system engineer and architect. Researched the Drchrono API, Apache Maven, OAuth 2.0, and Apache Oltu. Created the high-level component diagrams and the data routing and retrieval class diagram.
- Komali – Assistant high-level design, research of front-end user interface design, assistant front-end developer, MongoDB designer and implementer.

Work to be completed

For the first increment we created the Website and added some features like login, registration, patient and physician information, and crude data storage of user information in the database. Additionally, tables were created in an Oracle database and query operations were successfully performed.

In the future we have to work on getting alerts for patient users and using APIs like the Google Calendar API, the Drchrono API, and we have to work on testing. We have to perform NUnit testing for each step of the project, collecting or creating databases for prescriptions and we need to maintain the information about each capsule. Working with the different APIs will be a tough task for us, so we'll have to work on using the APIs and learning how to interface our system with them such that we can access the features that we need. That needs to be added to the web application and, finally, when the user or patient logs into the system he'll need to be able to access all of his health information, as well as helpful details such as prescription dosage and appearance. Additional inclusions will be suggestions from the physician and alerts about exercises. These features will come in future increments.

Responsibility:

- (User interface designer, Front-end developer, Server-side developer, Database Design and Implementation - Priyanka Bala Guddanti)
- (System Software Engineer and Architect, Server-side Developer, Project Manager, Documentation – Hayden McParlane)
- (Front-end Developer, Server-side Developer, Database Developer, Documentation - Komali Ghanta)