Team FitByte

Software Design Document

(Version 2)

2/19/19

Jake Farrar
Jacob Lemon
Austin Pederson

Ana Paula Chaves Steinmacher
Dr. Eck Doerry

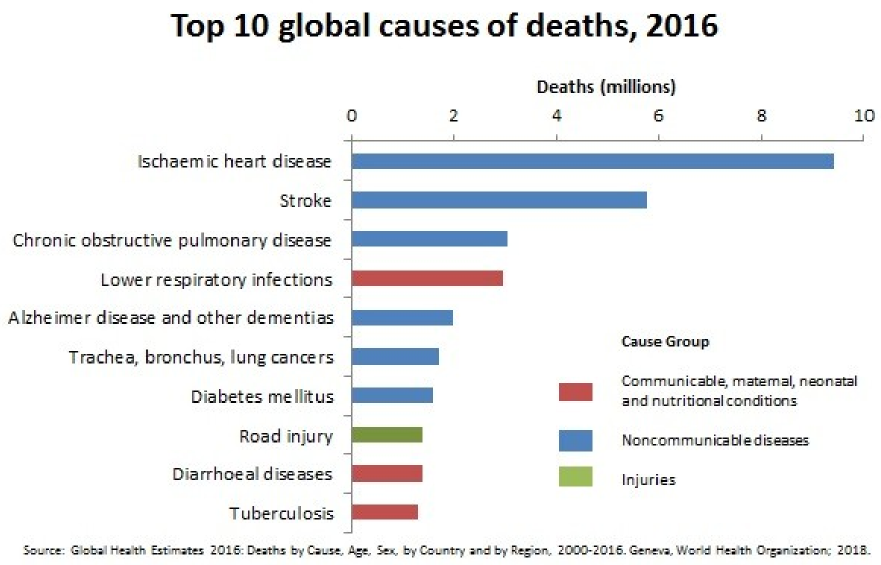
Dr. Kyle N. Winfree
Dr. Gregory Dominick

Table of Contents

1. Introduction

In 2016, the number one cause of death worldwide was ischemic heart disease (**Fig. 1**). In fact, ischemic heart disease claimed over four times as many lives as road injuries (**Fig. 1**). This statistic is shocking due to the fact that heart disease is preventable. The easiest way to help prevent ischemic heart disease is by completing at least 30 minutes of aerobic exercise five times per week. In order to help facilitate with reaching this goal, there are wearable technologies available to help motivate people to exercise, such as Fitbit.

Fig. 1: Graph detailing top 10 causes of death across the globe in 2016



Fitbit was established in 2007. Since then, Fitbit has enjoyed an enormous amount of success. In 2017 alone, Fitbit sold 15.3 million devices. This is very interesting to both of our sponsors as they are both in fields of study relating to human interaction with technology and its potential effects on fitness. Our sponsors are Dr. Kyle Winfree and Dr. Gregory Dominick. Dr. Winfree conducts his research at Northern Arizona University and focuses on the use of devices engineered for therapies and assessment of health. Dr. Gregory Dominick is a researcher at the University of Delaware whose research focuses on health-related topics. For example, *Modeling Clinically Validated Physical Activity using Commodity Hardware* is the title of a paper recently published by Dr. Dominick. This study details how Dr. Dominick and Dr. Winfree worked together to create a model that reduced Fitbit inaccuracies and made the results be closer to what would be seen if a research-grade device had been worn instead of a Fitbit.

They have been conducting research by giving their study participants Fitbits to wear for a month. This allows them to track all of the data that the wearable technology offers. Currently, they have a piece of software called WearWare to help facilitate with their research. WearWare consists of two major parts. A front end application that takes the form of a website and a back end application that deals with all of the data collection. The back end is the piece that grabs the data and exports it to a CSV. The front end is the piece that allows researchers to interface with the program conveniently, easily, and quickly. However, they are running into issues getting participant data, analyzing data for periods of inactivity, tracking progress towards goals, and returning the results to their researchers in a reasonable amount of time.

One of the glaring issues that our sponsors are facing is the sheer amount of time that it currently takes to process the participants' data in order to supply feedback. Researchers do not currently have an easy solution for dynamically analyzing the data in a reasonable amount of time in order to provide useful feedback to all of the participants in the study. Another issue is that researchers do not have an easy way to monitor their studies, the participants that are involved, or a quick way to apply Octave scripts to monitor data to find significant events. There is also currently not an easy way for new researchers to sign up to use WearWare. As a team, we are looking to enhance WearWare so that all of these issues will be resolved.

Team FitByte consists of Jake Farrar as Team Lead, Jacob Lemon as Quality Assurance (QA) and Enforcer, and Austin Pederson as Web Designer and Code Base Manager. We have been tasked with helping Dr. Kyle Winfree and Dr. Gregory Dominick create a solution that will allow them to automatically and dynamically assess their research participants' data in real time. We will be enhancing the capabilities of WearWare, implementing new features into WearWare, and creating a web Application Protocol Interface (API) that allows the team in Delaware (led by Dr. Dominick) to request and receive data in real time that is stored in WearWare's back end. The API will ideally let the Delaware team receive data from WearWare without having to go through the web application and wait for the large CSV to download. The API acts as a gateway that can get data from the database and send it via JSON to whoever requested it. The new features that our team will be implementing are a proper participant management interface, a way for researchers to manage their studies, and a way for data analysis scripts to be run on selected data on our server.

2. Implementation Overview

FitByte's vision for a solution is simple. Our goal is to use WearWare's functionality and data collection as a base for our web application. The main skeleton of WearWare will remain intact. We will then rebuild the UI and enhance its functionality using the same technologies already employed by WearWare.

WearWare, like most websites and web applications, consists of a frontend and a backend. The frontend is the piece that we are planning on stripping away and rebuilding from scratch. This piece contains all the parts that the user interacts with. This includes the UI, any animations, drop-down menus, etc. The backend is the part of the website that handles all of the interactions with any databases, other servers, and basically anything that the user does not directly interact with. This is the portion of the website that will remain mostly intact.

The single most important piece of technology that we will be using is Django Web framework. Django is an open source framework developed for the programming language Python. We will be using Django version 2.1.5 as it was the latest available release at the time we started development. This technology will be used by us for multiple reasons. First and foremost, we will be using Django because WearWare is currently written using Django. This will increase the extensibility of our software after we are no longer working on it. It will also enhance the interoperability of our improvements with the existing codebase. This will reduce the amount of time that we spend getting these two different pieces of software to work together. Consequently, this will increase the amount of time that we can spend working on new features for WearWare.

Another piece of technology that we will be using is PostgreSQL. We will be using version 10. The main reason that we will be using PostgreSQL is because it is the currently being used to store all of the data for WearWare. By using the same technology, we reduce the amount of time that we have to deal with the backend. This, in turn, increases the amount of effort we can put forth in areas that are significantly less functional.

For any tasks that need to be run on a specific schedule, we will be using Celery. We will be using Celery for the functionality of scheduling tasks to be run at specific times. Version 4.2.1 of Celery will be used as it is currently the latest version.

For web hosting, the piece of technology that we plan on using is Amazon Web Services (AWS). Specifically, we will be using an AWS Elastic Cloud Compute (EC2) instance. This is essentially a server on which our enhanced version of WearWare will run. This will make WearWare easily scalable. If the website is sluggish, the EC2 instance and be upgraded to a more powerful version easily. This will handle any issues that could occur with multiple researchers using WearWare at any given time.

On the server, we will be using two additional pieces of technology in order to assist with hosting our website. These two pieces of technology are uWSGI and NGINX. UWSGI is the piece of software that will, essentially, host WearWare locally. UWSGI will create a socket that NGINX can hook into. NGINX will be the piece of technology that will let people access our locally hosted content from the Internet. Essentially, uWSGI will run WearWare, and NGINX will make WearWare available via the Internet.

3. Architectural Overview

This section gives an overview of the proposed architecture of our software. **Fig. 2** represents our envisioned software solution. This diagram will then briefly be explained in order to give a bird's eye view of our website. The more specific details will be explained in section 4 of this document.

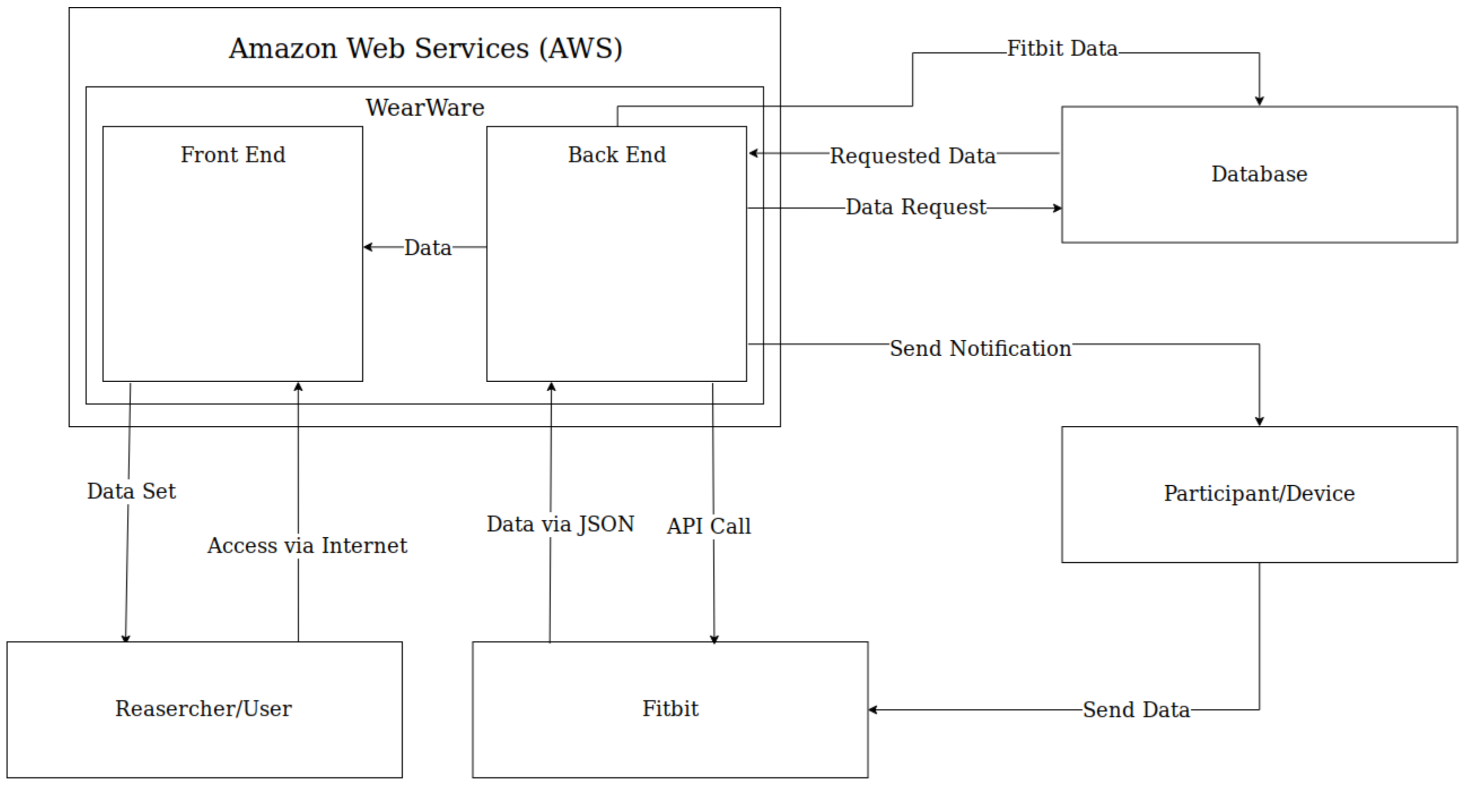


Fig. 2: Architectural overview of our software

Each component of **Fig. 2** has a certain set of responsibilities. We can lay these responsibilities out for each specific component. We will start in the upper left corner of **Fig. 2** and then work our way down and to the right in order to discuss all the responsibilities.

Amazon Web Services is the first component of which we will discuss the responsibilities. Amazon Web Services will be responsible for two crucial pieces of our software. First, it will host the data collection side of WearWare. Second, Amazon Web Services will host the website with which researchers can interact.

WearWare is the next component. The key responsibilities of WearWare are twofold and each responsibility is addressed by each part. The back end must collect data from Fitbit for users in a study and store that data in such a way that it can be retrieved later. The front end will receive data from the back end and present it to any user logged into WearWare's website.

The database is also a component in this system. Its responsibilities are to store and serve data. These responsibilities are pretty standard for a database. These responsibilities should be relatively easy to enforce.

Fitbit is the next component of our architecture. It only has one responsibility, which is is to serve data based on requests from WearWare. This component is not controlled by us, so we will assume that it is always working.

The final component in our architecture is a person/device. This is any research study participant or their wearable fitness tracking device. The responsibility of the person/device is to ensure that their data is sent to Fitbit regularly. This can be attempted to be controlled as we will discuss later.

The communication methods/information control among components will be carried out in the following ways. The person/device will send their data to Fitbit via some method that we are not sure of as the software is closed source. The basic flow of data is that WearWare will request data from Fitbit using an API call, Fitbit will send a response to WearWare, and that data will be analyzed and any SMS messages will be sent to the required people. All of these communications and the flow of information is represented as arrows in **Fig. 2**.

4. Module and Interface Descriptions

The architecture and interface of our website starts with the Login Page (see **Fig. 3** below). Starting with the Login Page shown by **Fig. 4**, a user is able to login to the application as an administrator or a user (depending on their account type), sign up, or reset their password. Anyone is allowed to sign up, but there is an administrator option to require approval before gaining access to WearWare. If the administrator approval option is enabled, anyone who signs up will be put in a queue to be approved by an administrator. As seen by the arrows in **Fig. 3**, a user can travel to the Forgot Password Page and back to the Login Page, to the Sign up Page and back to the Login Page, or to the Administrator Dashboard or User Dashboard, based upon their account type.

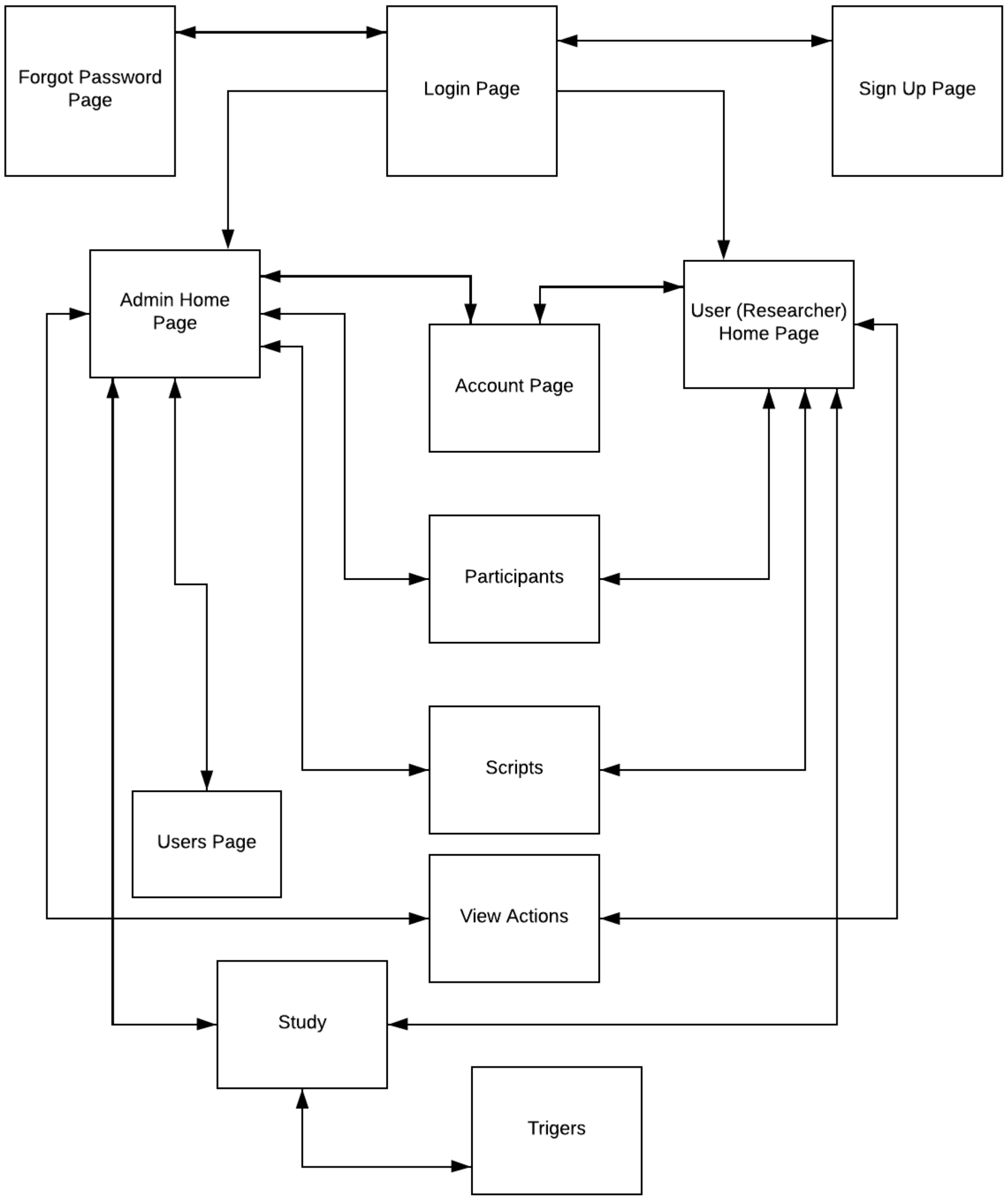


Fig. 3: Diagram showing the architecture of our website

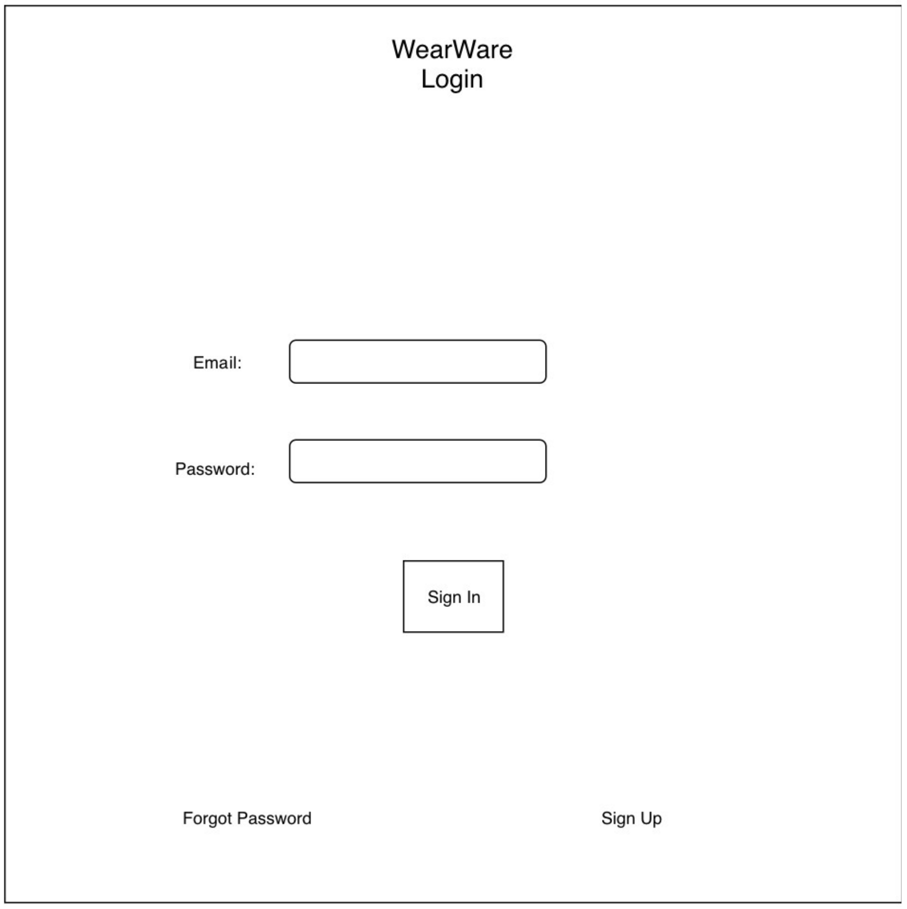


Fig 4. Login Page

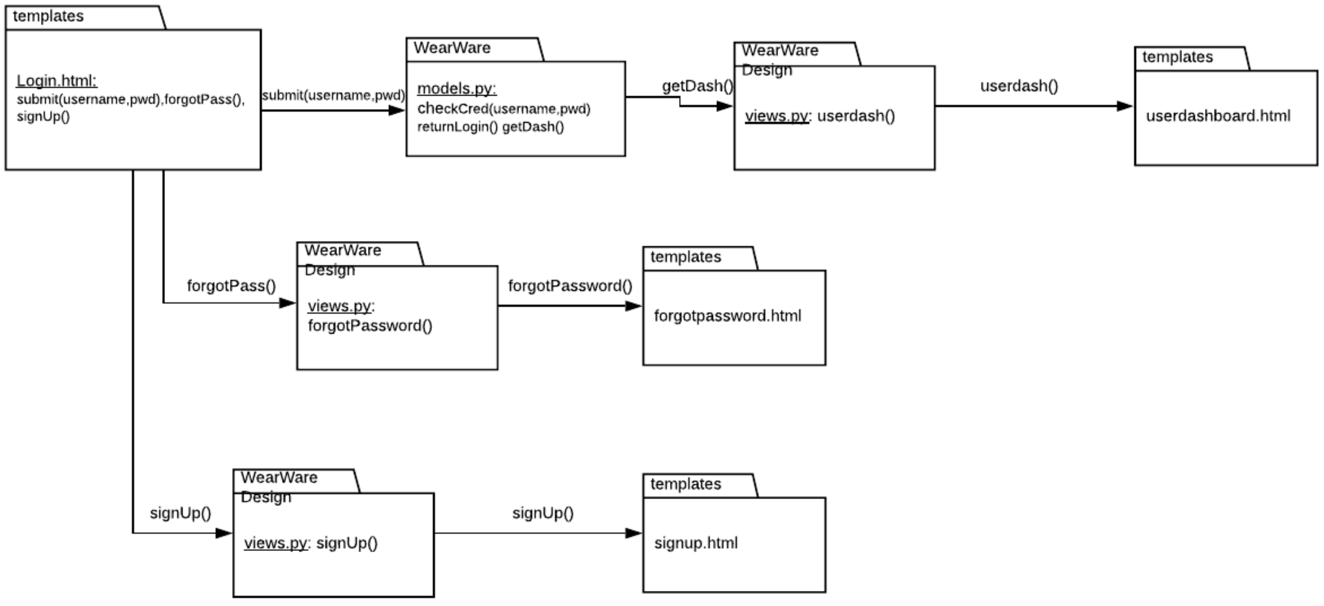


Fig 5. UML of Login Page

The Forgot Password Page as shown by **Fig 6,** allows users a way to reset their password if it is ever forgotten. The page has a form that asks for the email that is associated with the account and a submission button. The user can then travel back to the Login Page as shown by **Fig 7**. The Sign Up Page allows users to create an account so they can participate in research studies. The page has a form that asks for first name, last name, email, cell phone number, and a submission button. The user can travel back to the login page after they have followed the sign up flow as seen in **Fig. 8**.

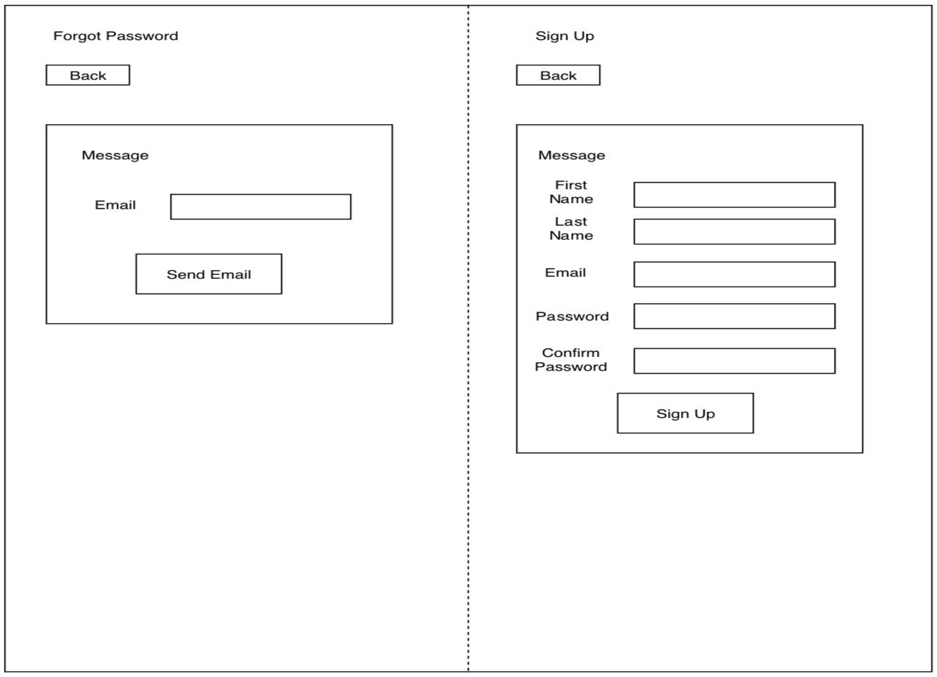


Fig. 6 Sign Up and Forgot Password

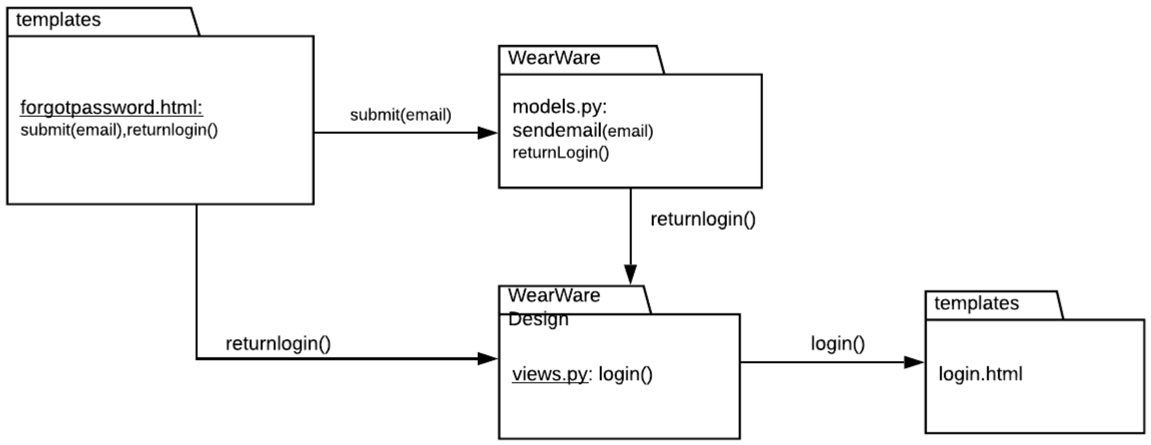


Fig. 7 Forgot Password UML

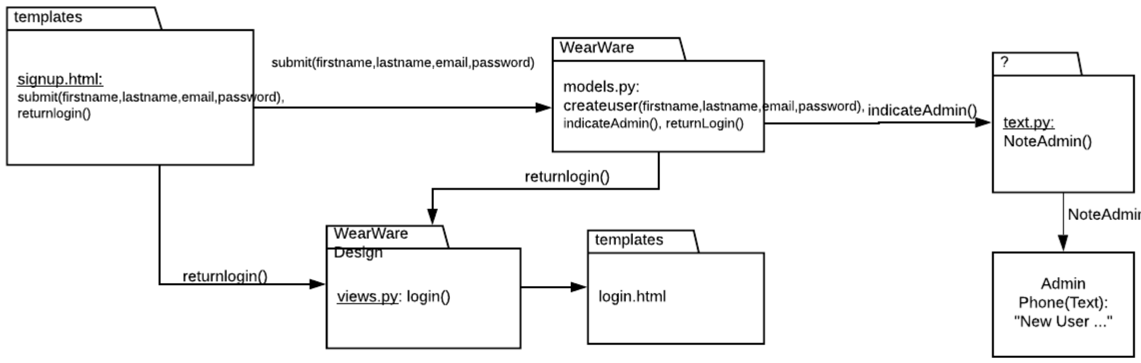


Fig. 8 Sign Up UML

Two different dashboard pages can be loaded depending on the type of account that is logging in (see **Fig. 9** and **Fig. 10**). The administrator accounts have access to everything while the user (researcher) accounts can see a limited version of the administrator account. The admins have access to the Users Page, Account Page, and Study Page. The researchers have access to all but the Users Page. This can be seen in **Fig. 3**. The dashboard offers a view of the users account and some basic info, a message of the day, the different studies that they are involved in, and links to other pages. Each page that can be accessed from the dashboard has a way to return to the dashboard via the navigation bar at the side of the webpage. This is shown in **Fig. 3** with the double arrows. **Fig. 11** and **Fig. 12** show the flow of both dashboards.

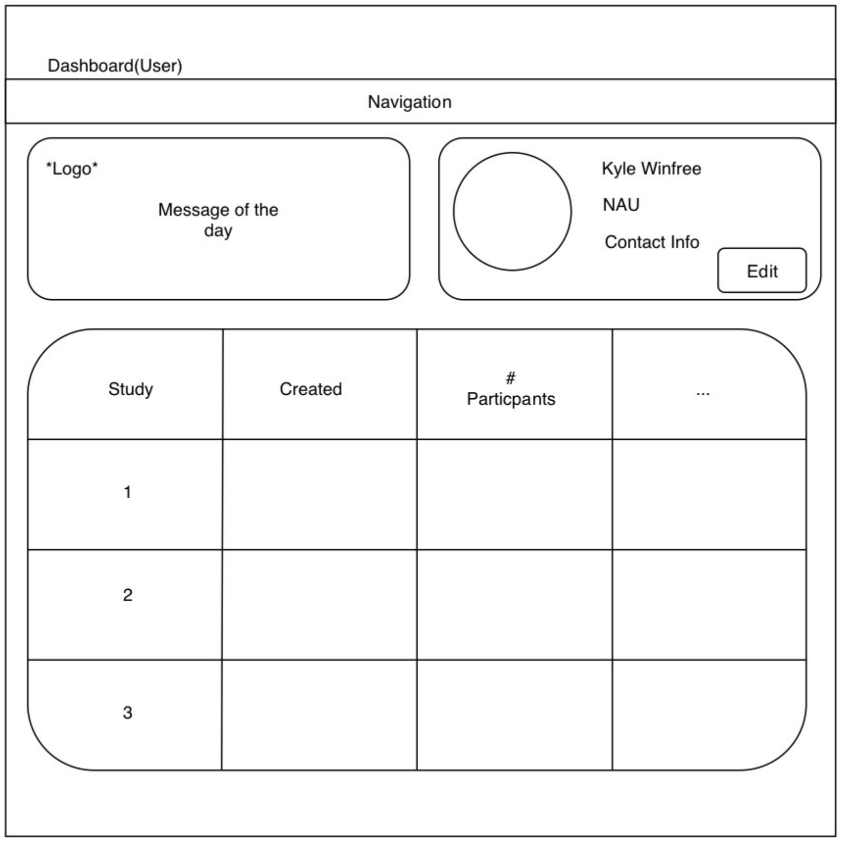


Fig. 9 User Dashboard

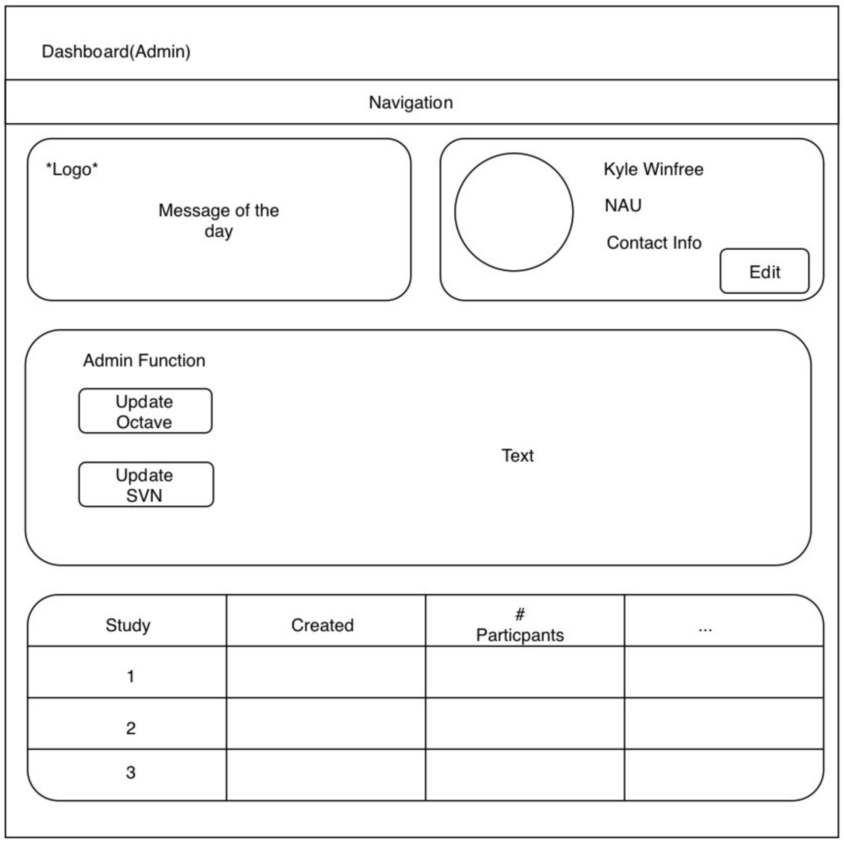


Fig 10. Admin Dashboard

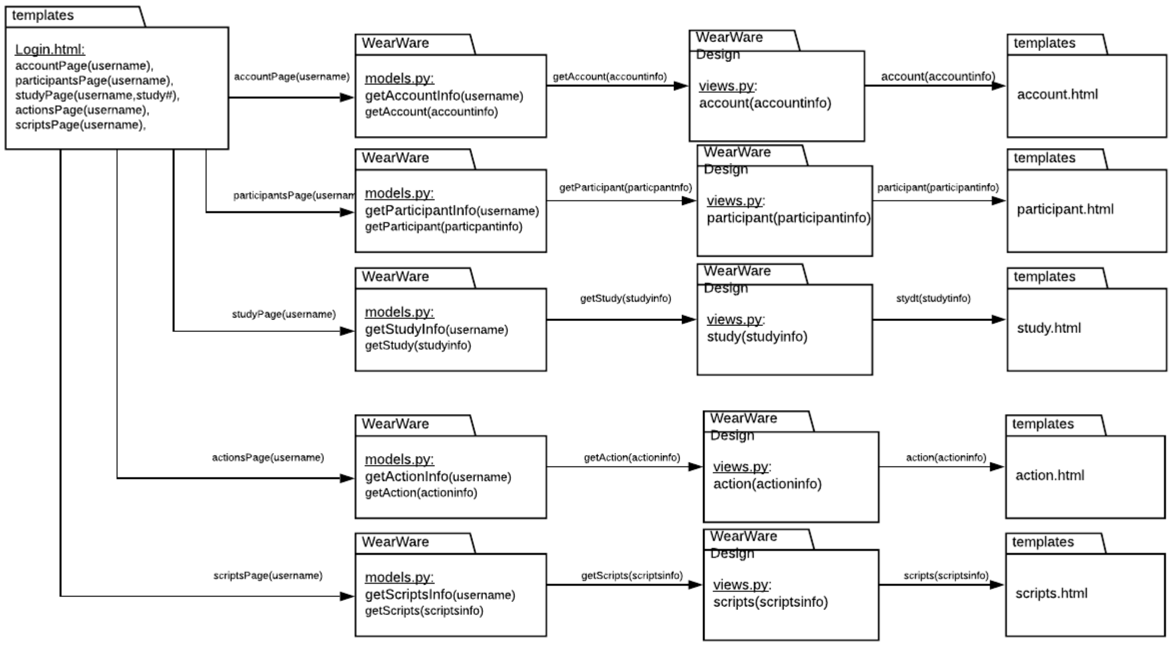


Fig 11. UML of User Dashboard

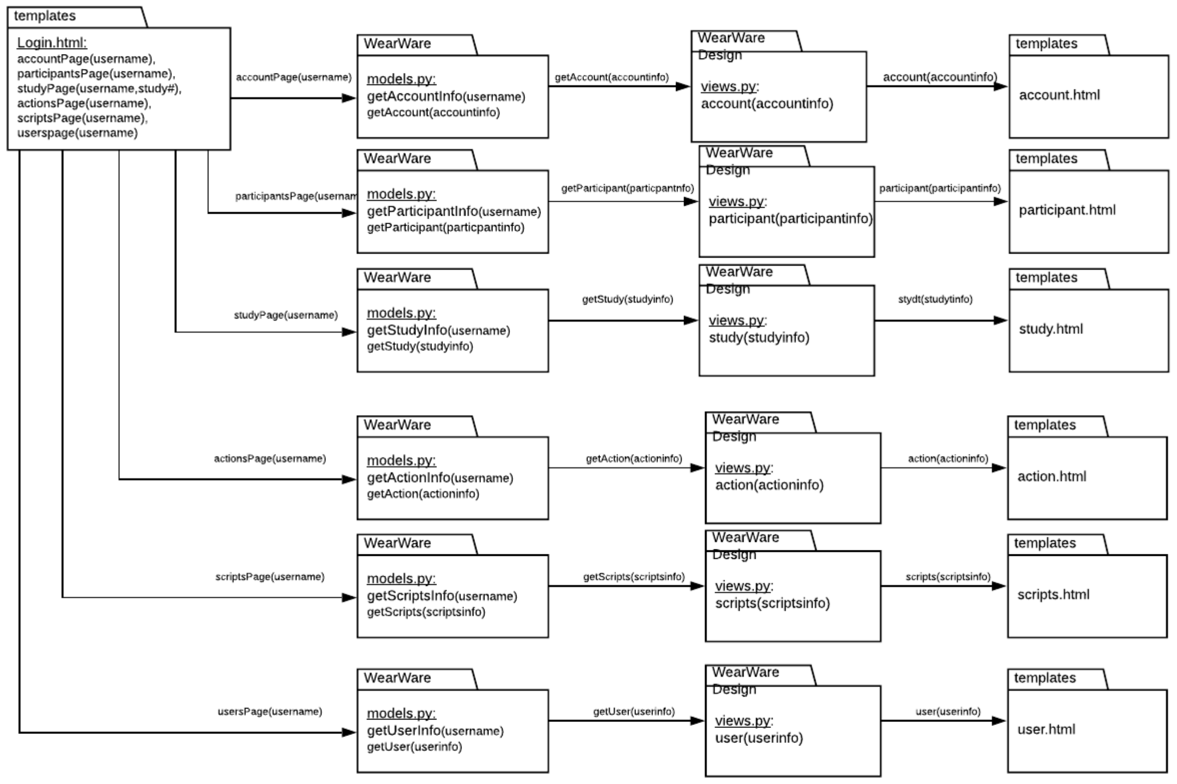


Fig 12. UML of Admin Dashboard

The Account Page shown by **Fig. 13** can be accessed from the edit account link from the dashboard. This gives the users the ability to change basic information about their Account such as first name, last name, and password. To change their information, they will need to be able to verify their existing password. This is done when the user clicks the “save” button. The dashboard can be accessed again using the navigation bar at the side of the screen. The flow of the page is shown by **Fig. 14.**

The nav bar at the side of the screen will be in a box that contains buttons that link to other pages within the web application and will be on most pages of the web application. The message of the day will also be in a box that contains a logo which is determined by Dr. Winfree or the current admin. It will be constantly changing but the box will stay the same. If the message is too long then the box will be able to scroll. The user profile info will also be in a box. It will contain a photo chosen by the user within the about page. There will also be a edit button which links to the edit information page. The studies will be in a table that will grow depending on the number of studies that are able to be view. It will contain multiple columns that will be centered on the page. The admin dashboard will have an extra box that contains a few buttons that update certain parts of WearWare.

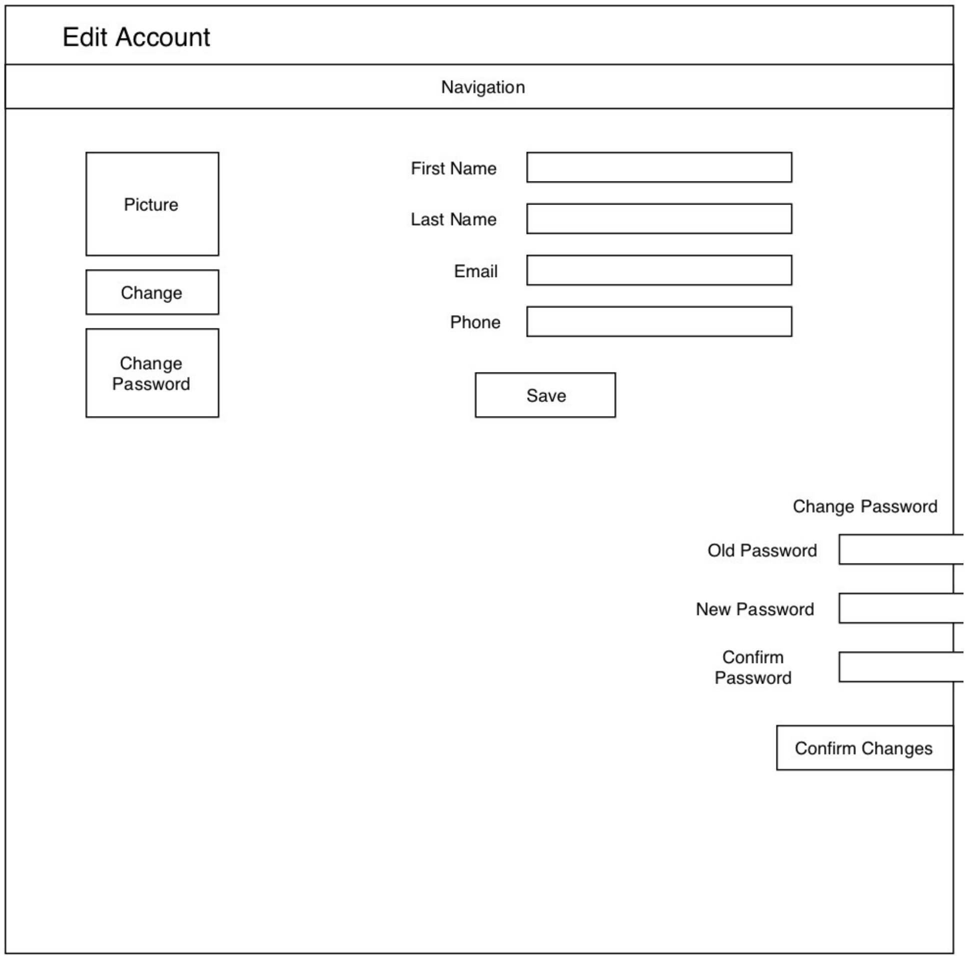


Fig. 13 Account Page

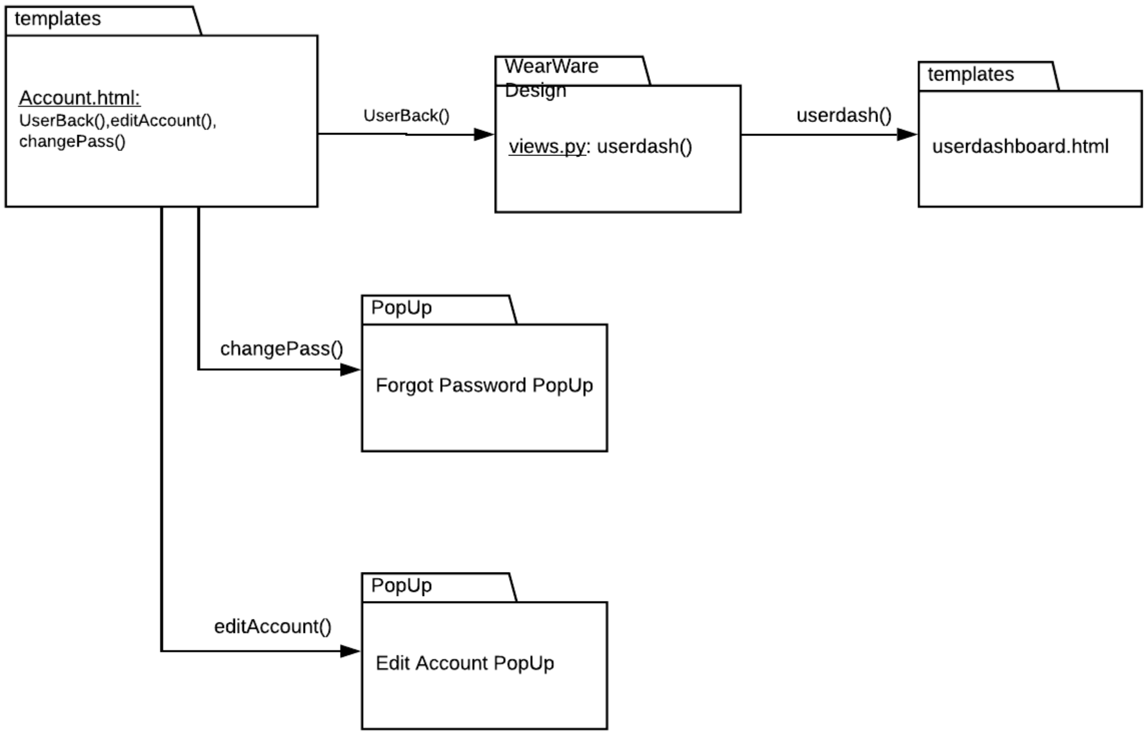


Fig 14. UML of Account Page

The Participant Page shown by **Fig. 15** can be accessed from the edit participant link from the dashboard as well as the option to return to the dashboard. It creates a table that shows the user or admin all of the participants they have access to for their studies. It allows for new participants to be added via CSV. It also allows for the current list to be exported to the user via CSV. The flow of the page is shown by **Fig. 16.**

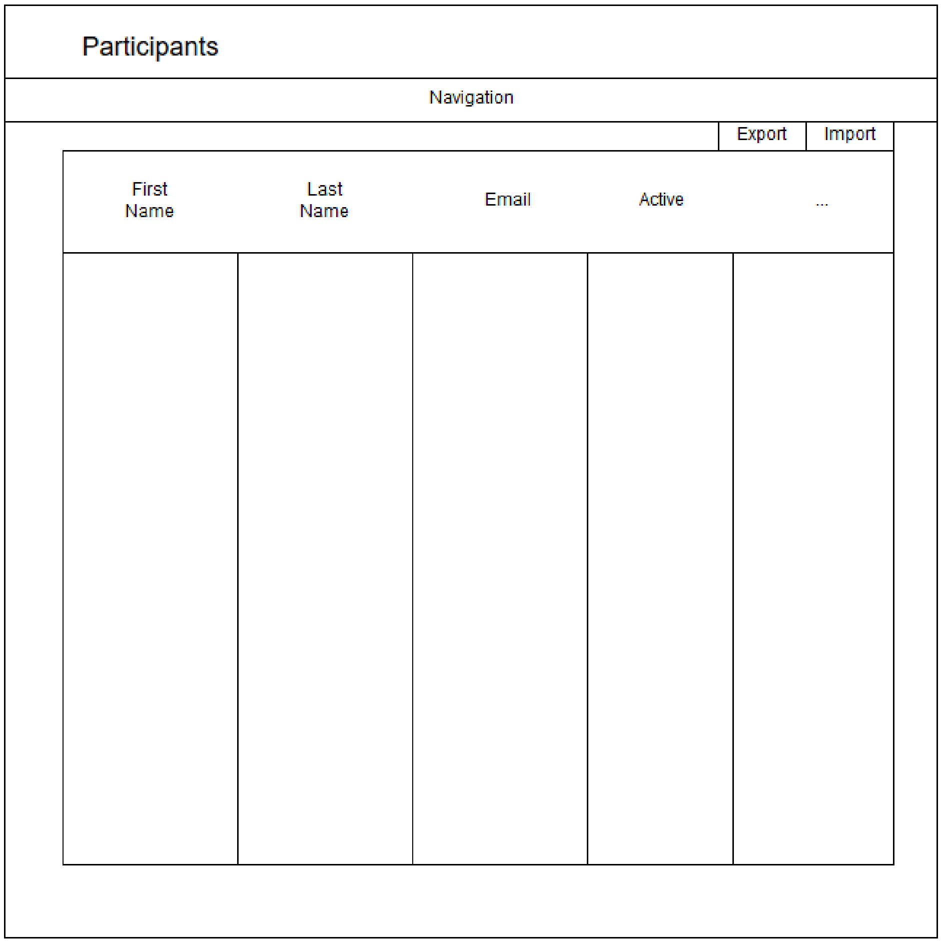


Fig. 15 Participants Page

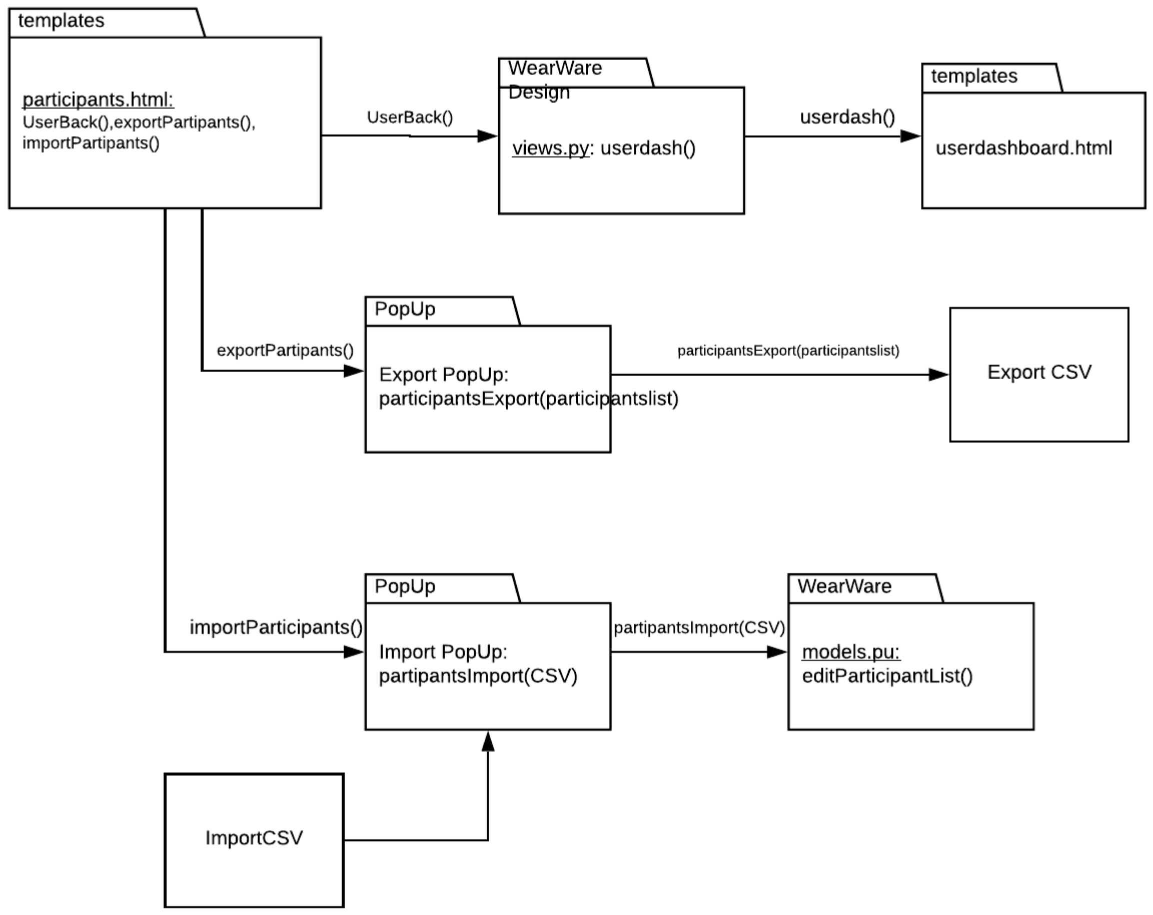


Fig 16. UML of Participants Page

The User Page shown by **Fig. 17** can be accessed from the edit participant link from the dashboard as well as the option to return to the dashboard. It creates a table that shows the admin all of the users they have in the system. It allows for new users to be added via CSV. It also allows for the current list to be exported to the admin via CSV. The flow of the page is shown by **Fig. 18.**

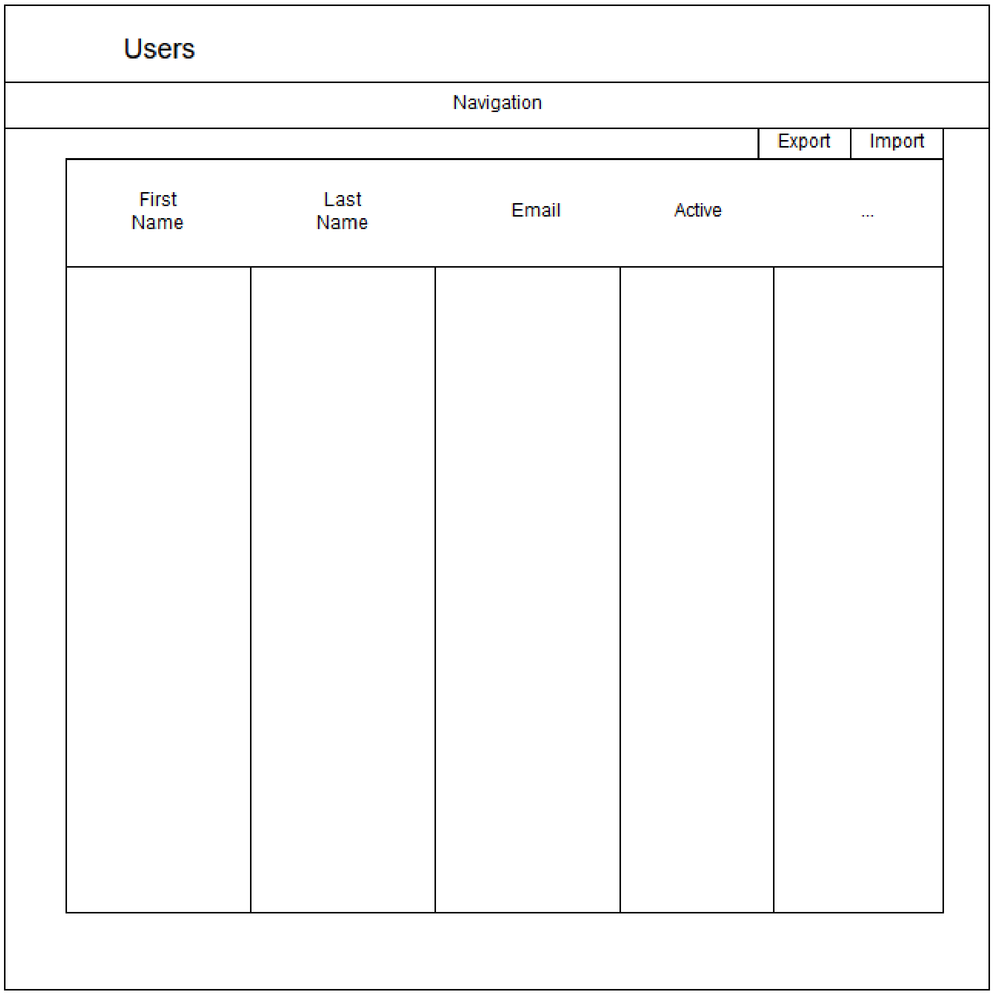


Fig. 17 Account Page

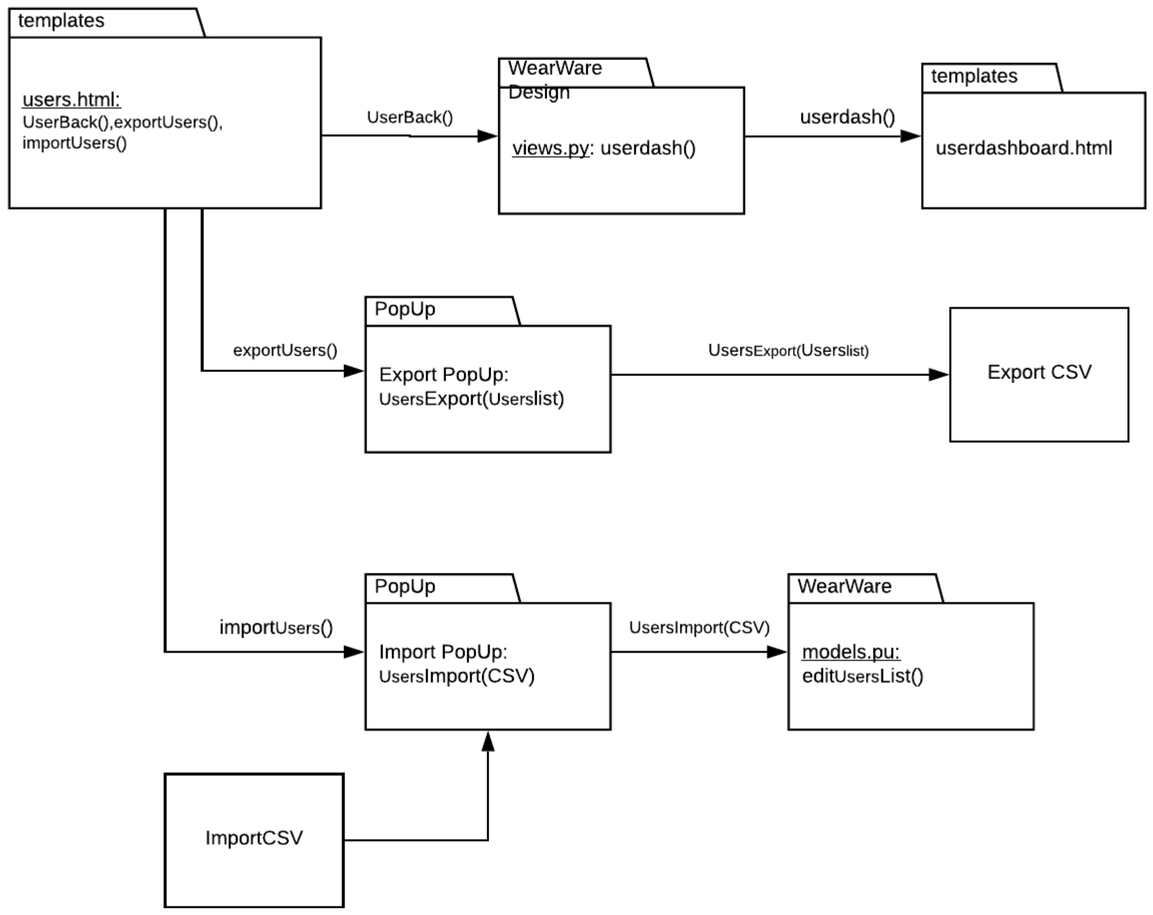


Fig 18. UML of Users Page

The Scripts Page shown by **Fig. 19** can be accessed from the edit participant link from the dashboard as well as the option to return to the dashboard. It creates a table that shows all the scripts. Scripts must be imported in to be added. Each script will have a popup that shows information about each one. The flow of the page is shown by **Fig. 20.**

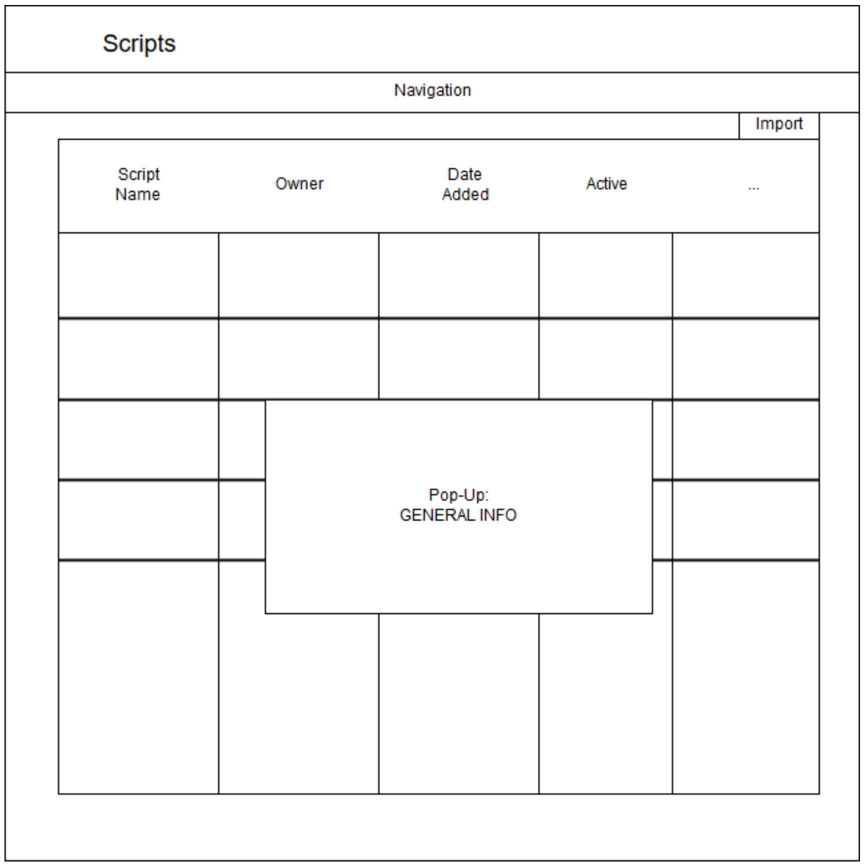


Fig. 19 Scripts Page

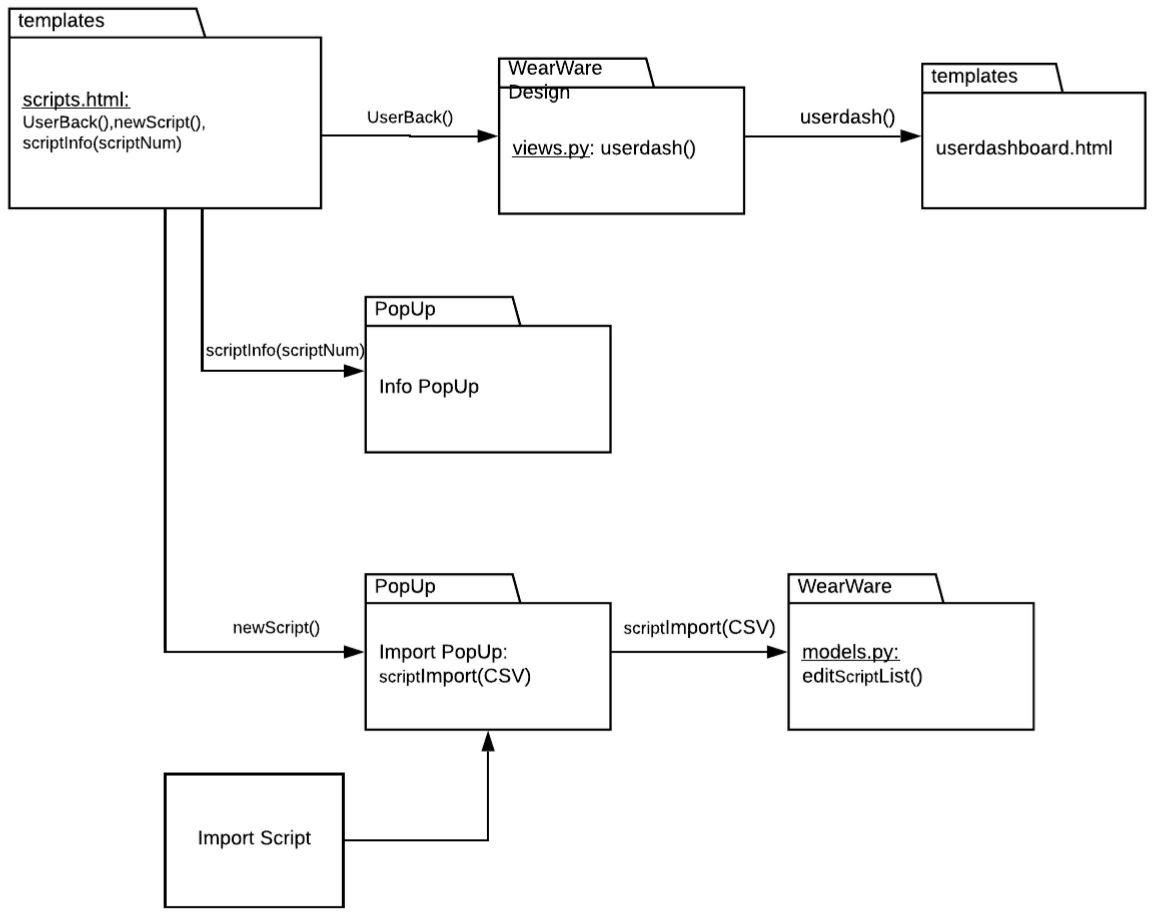


Fig 20. UML of Scripts Page

The Actions Page shown by **Fig. 21** can be accessed from the edit participant link from the dashboard as well as the option to return to the dashboard. It creates a table that shows all the action. Actions will be add via popup after clicking an add button. Each action will have a popup that shows information about each one. The flow of the page is shown by **Fig. 22.**

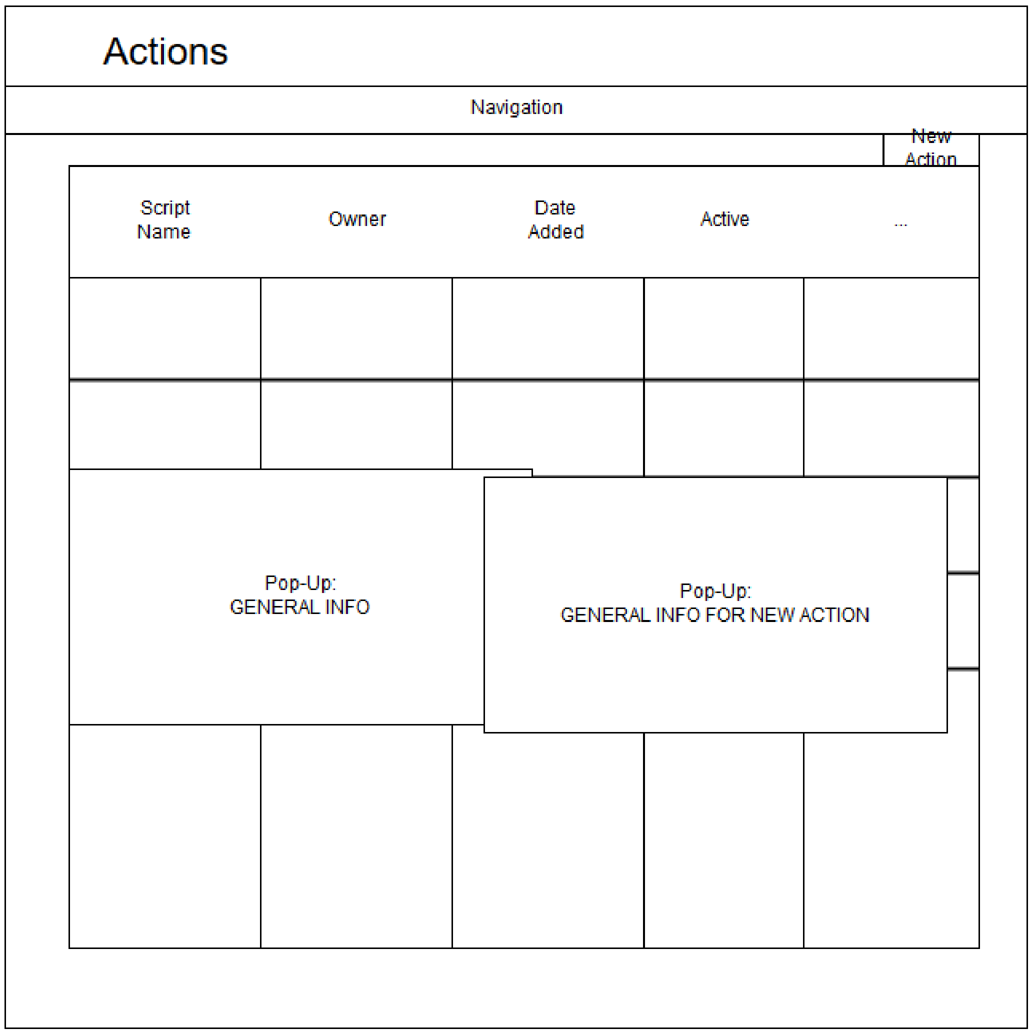


Fig. 21 Actions Page

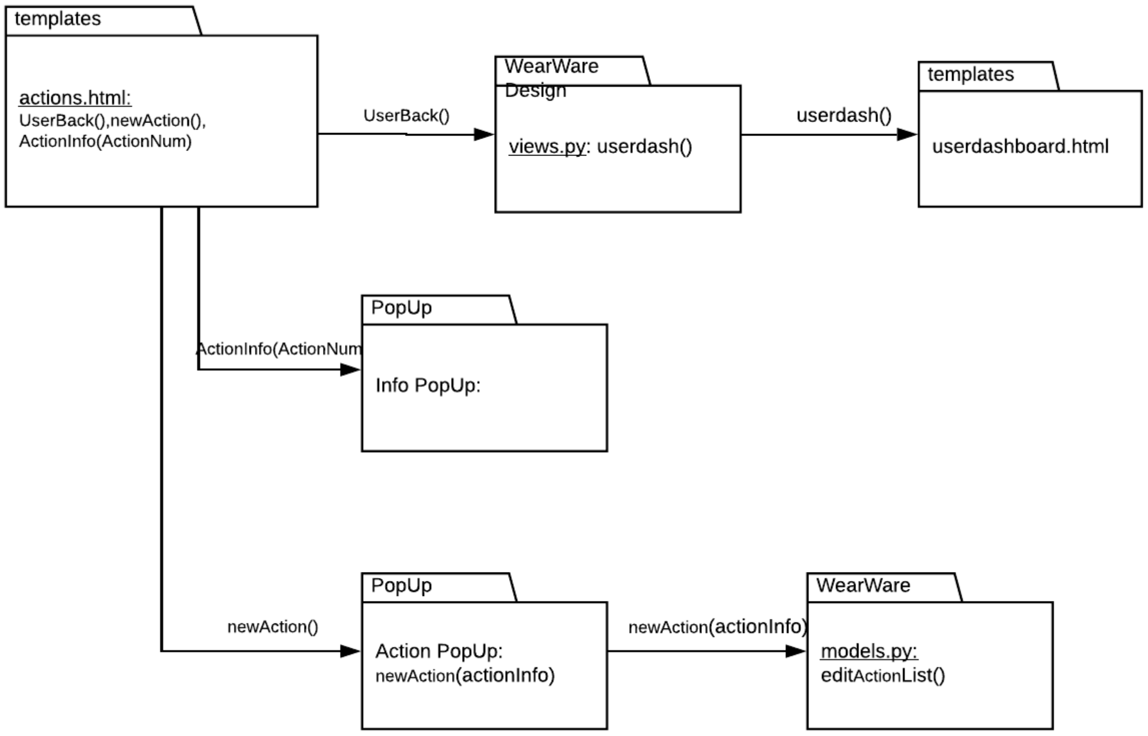


Fig 22. UML of Actions Page

The Study Page shown by **Fig. 23** can be accessed from the edit participant link from the dashboard as well as the option to return to the dashboard. It creates a page that will show information about the study. The participants in the study will be shown on the left. Actions can be done with the participants list which will be determined at a later time. The relevant data will be shown on the right. This also will be determined later. A button for viewing triggers which perform actions on the data. This will bring the user to a different page. The flow of the page is shown by **Fig. 24.**

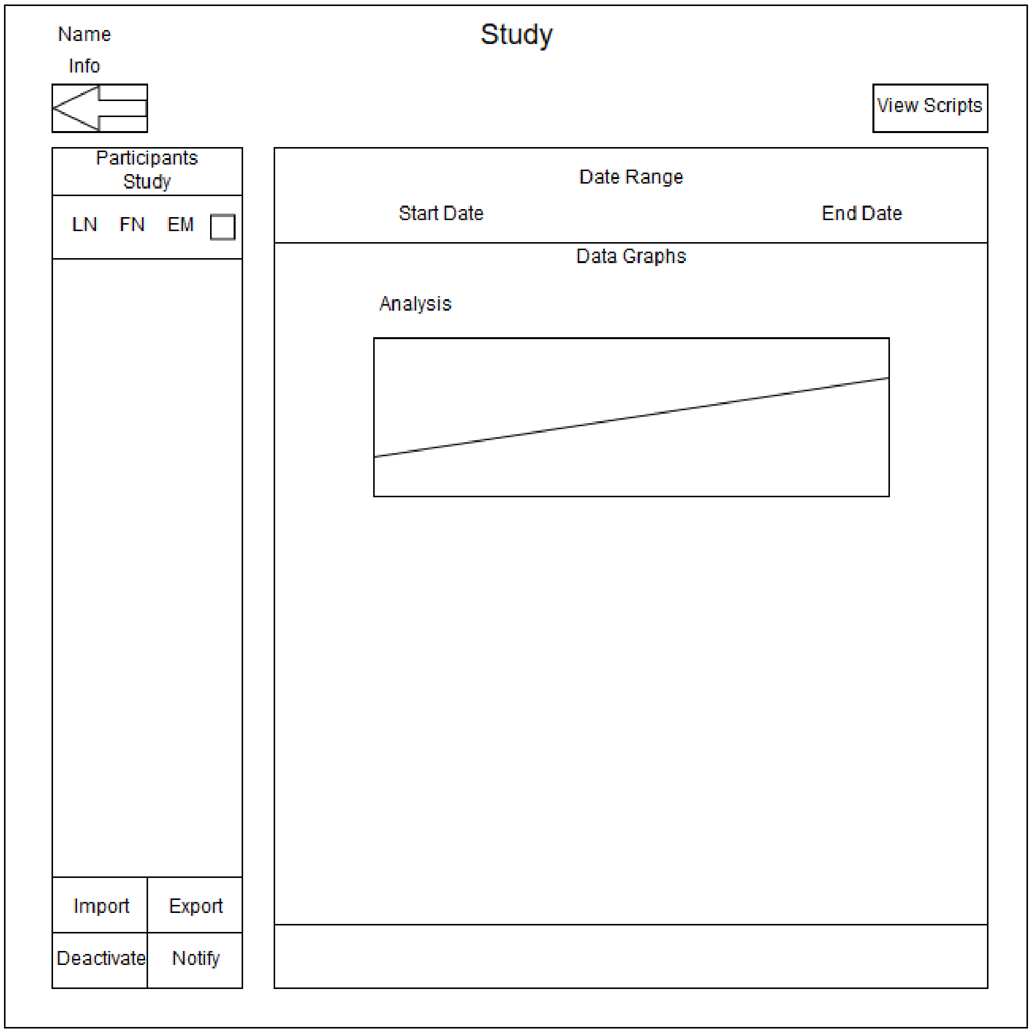


Fig. 23 Study Page

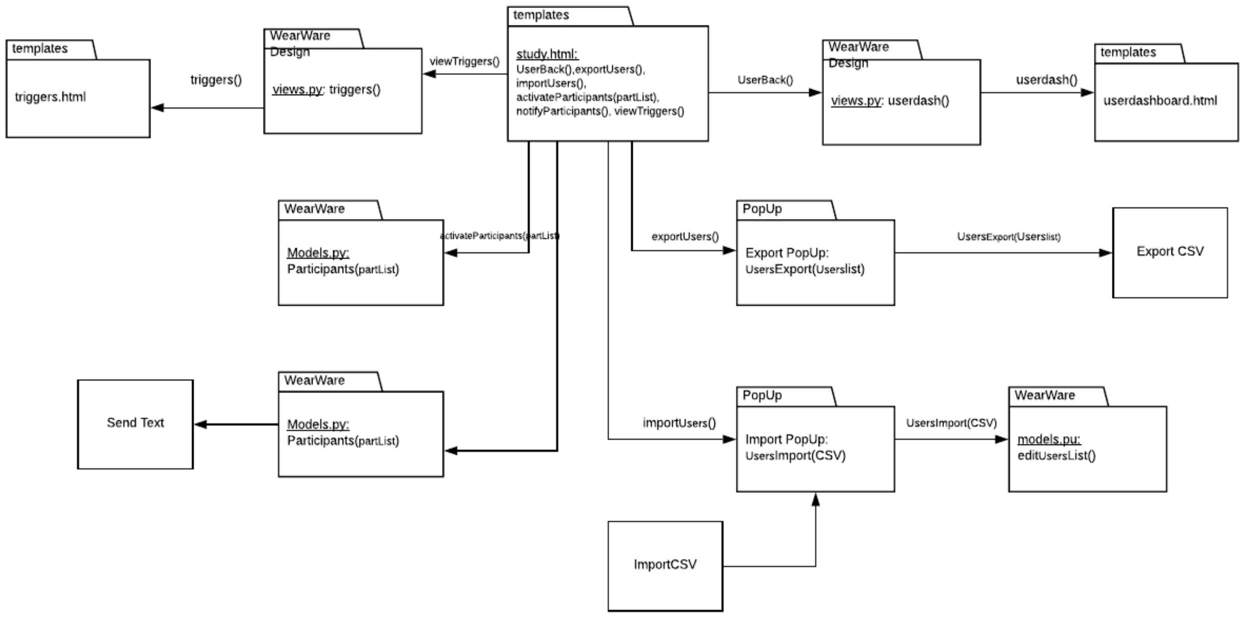


Fig 24. UML of Study Page

The Triggers Page shown by **Fig. 25** can be accessed the study page from the dashboard as well as the option to return to the study. It creates a page that will show information about the study's triggers. Buttons will be added to activate or deactivate the triggers and an option to add a new action. The flow of the page is shown by **Fig. 26.**

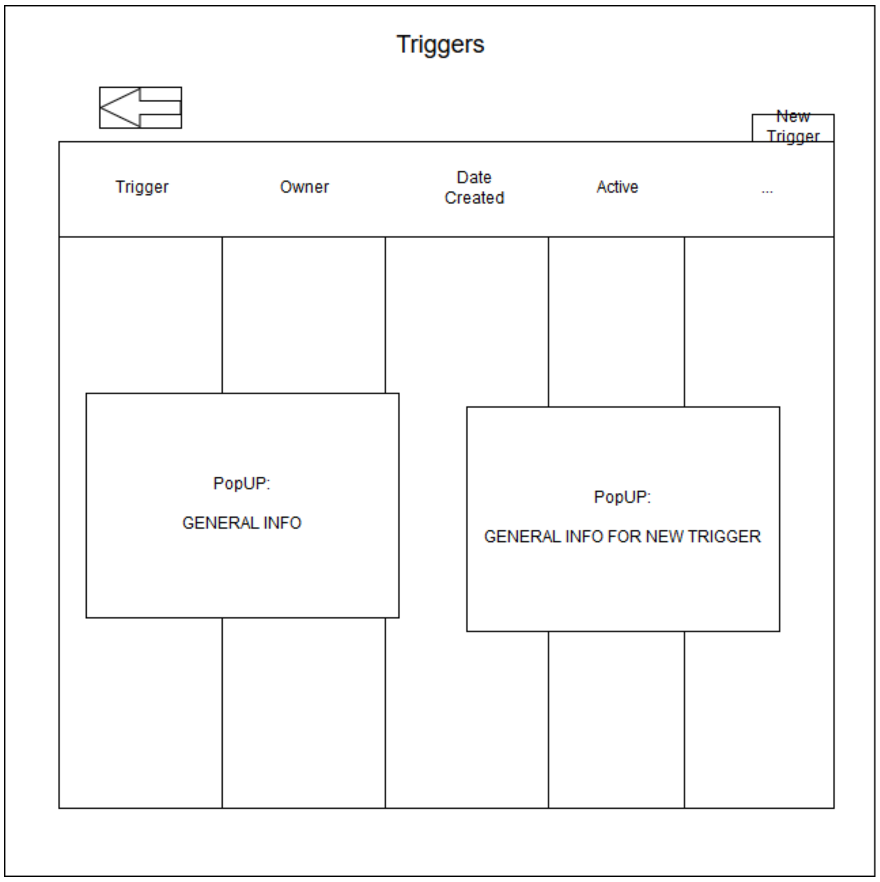


Fig. 25 Triggers Page

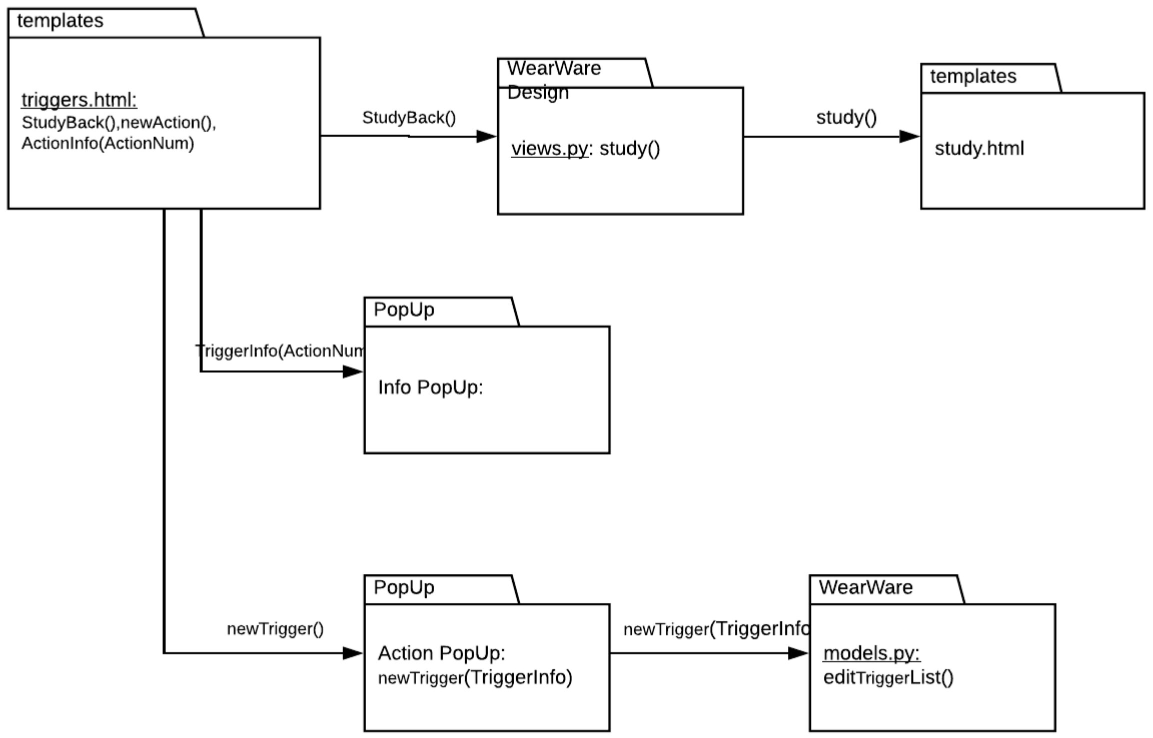


Fig 26. UML of Triggers Page

5. Implementation Plan

Team FitByte plans to use the above Gantt chart, **Fig. 27** as a tracker for milestones completed by the project. Our first goal as a team is to have the current version of WearWare up and running on an AWS server. This part of the project will be headed by Austin since he has the most experience with AWS while the others will be assisting as he assigns. We plan to have it up by Mid-February as most of the web app will be adapted from the current version. Once the server is hosting WearWare, the team will shift focus towards designing and creating web pages. This includes a standardized look in the theme as well as the functionality of the pages. This will be headed by Jake while the other members of the team will be helping as assigned by him. The team will have a week to prepare for a Beta Launch by Spring Break. After the break, the team will begin to look into bug fixes and improving the software as the client needs. The team will then launch the final product by the end of April.

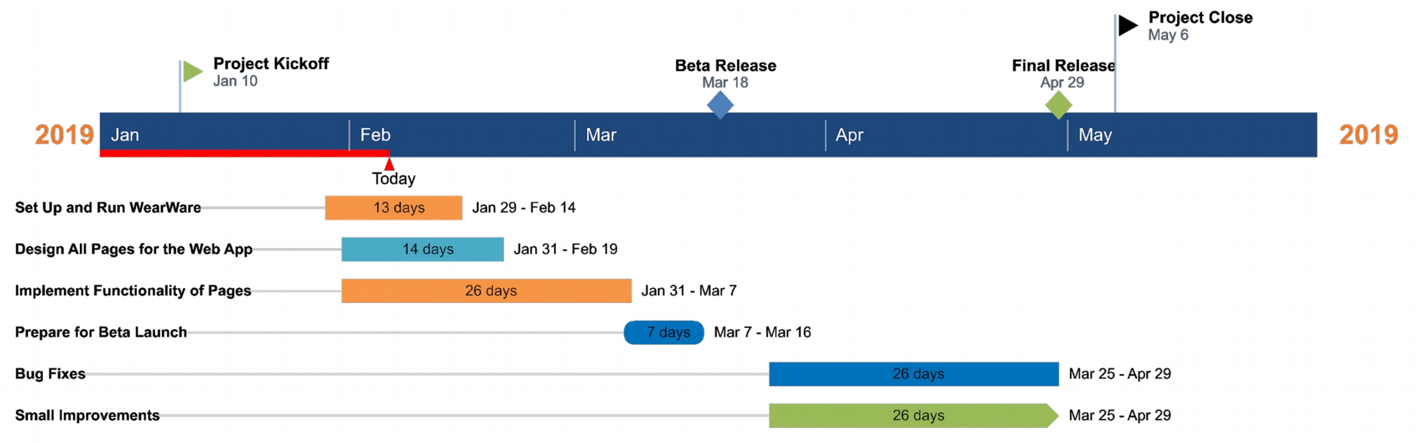


Fig. 27: Gantt Chart Detailing Project Timeline

6. Conclusion

Team FitByte plans to remedy Dr. Kyle Winfree and Dr. Gregory Dominick’s issues by extending the capabilities of WearWare. This will include moving WearWare to a more accessible location as well as developing WearWare into a more robust piece of software. We will achieve the accessibility by migrating WearWare to an AWS EC2 instance. The biggest portion of our plans will be updating and upgrading WearWare. This will include a complete redesign of the website user interface. We will also be heavily focusing on the integration of new features to WearWare. The overall goal of the web application will be to speed up and simplify the process of creating and monitoring a study for researchers using our software. Our all-inclusive website will allow them the ability to easily manage their users, studies, and analyze their data.