Project Eir

Work done by Clayton Damon

(over the span of multiple sprints)

Software Requirements Specification

Version 1.0

February 26, 2020

Rami Alsibai

Leonardo Serrano

Clayton Damon

Keith Bragg

Taylor Pedretti

Development Team

Prepared for

# **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 02.19.2020 | Version 1.0 | Team Pantheon | Rough Draft of SRS Document |
| 03.1.2020 | Version 1.1 | Team Pantheon | SRS Improved  Login Working |
|  |  |  |  |
|  |  |  |  |

# **Document Approval**

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | <Your Name> |  |  |

**Table of Contents**

Revision History........................................................................................................................................... ii

Document Approval........................................................................................................................................ ii

1. Introduction................................................................................................................................... 1

1.1 Purpose.................................................................................................................................. 1

1.2 Scope........................................................................................................................................... 1

1.3 Definitions, Acronyms, and Abbreviations............................................................................................................................... 1

1.4 References.................................................................................................................................. 1

1.5 Overview..................................................................................................................................... 1

2. General Description................................................................................................................................. 2

2.1 Product Perspective................................................................................................................................ 2

2.2 Product Functions................................................................................................................................... 2

2.3 User Characteristics.......................................................................................................................... 2

2.4 General Constraints............................................................................................................................... 2

2.5 Assumptions and Dependencies.......................................................................................................................... 2

3. Specific Requirements........................................................................................................................... 2

3.1 External Interface Requirements........................................................................................................................... 3

3.1.1 User Interfaces................................................................................................................................. 3

3.1.2 Hardware Interfaces................................................................................................................................. 3

3.1.3 Software Interfaces................................................................................................................................. 3

3.1.4 Communications Interfaces................................................................................................................................. 3

3.2 Functional Requirements.......................................................................................................................... 3

3.2.1 <Functional Requirement or Feature #1>.......................................................................................................................................... 3

3.2.2 <Functional Requirement or Feature #2>.......................................................................................................................................... 3

3.3 Use Cases...................................................................................................................................... 3

3.3.1 Use Case #1............................................................................................................................................ 3

3.3.2 Use Case #2............................................................................................................................................ 3

3.4 Classes / Objects.................................................................................................................................... 3

3.4.1 <Class / Object #1>.......................................................................................................................................... 3

3.4.2 <Class / Object #2>.......................................................................................................................................... 3

3.5 Non-Functional ..Requirements............................................................................................................................ 4

3.5.1 Performance............................................................................................................................... 4

3.5.2 Reliability........................................................................................................................... 4

3.5.3 Availability................................................................................................................................... 4

3.5.4 Security....................................................................................................................................... 4

3.5.5 Maintainability.................................................................................................................... 4

3.5.6 Portability.................................................................................................................................... 4

3.6 Inverse Requirements.......................................................................................................... 4

3.7 Design Constraints............................................................................................................... 4

3.8 Logical Database Requirements.......................................................................................... 4

3.9 Other Requirements............................................................................................................................. 4

4. Analysis Models........................................................................................................................................ 4

4.1 Sequence Diagrams.................................................................................................................................... 5

4.3 Data Flow Diagrams (DFD)......................................................................................................................................... 5

4.2 State-Transition Diagrams (STD)......................................................................................................................................... 5

5. Change Management Process..................................................................................................................................... 5

A. Appendices........................................................................................................................... 5

A.1 Appendix 1........................................................................................................................... 5

A.2 Appendix 2........................................................................................................................... 5

# **1. Introduction (DONE)**

Have you ever been to the hospital and was shocked about how much you had to pay to treat your injuries or illnesses? We're sure you have. Project Eir is designed to help you find the cheapest price for whatever procedure you need at your local hospitals. We wanna keep money in your pocket while hospitals wanna take it. We take the average prices that each hospital has to release every year and put them all in an easy to use website that will go through and find the cheapest hospital for you. (Rami Alsibai)

## **1.1 Purpose (DONE)**

Project Eir seeks to allow users to easily view the price of procedures to ensure they can obtain the lowest price. Our target audience is anyone who needs a medical procedure especially those without insurance.(Keith Bragg)

## **1.2 Scope (DONE)**

The scope for this project is to create a free website that will allow people to get the best bang for their buck when visiting the hospital. These days a lot of people don't have insurance and have to pay out of pocket or they have insurance and have high deductibles. Project Eir will show them the average cost of their specific medical procedure at local hospitals to help them keep some money in their pocket. This product will of course not do the medical procedure for them or book an appointment for them, it will just show you the average cost from the hospitals yearly report of costs. (Rami Alsibai)

## **1.3 Definitions, Acronyms, and Abbreviations (DONE)**

Project Eir strives to provide the most user friendly experience, our program searches through difficult medical definitions to best match your medical related search.

**Eir**- In Norse mythology, Eir is the goddess associated with healing.

**Pantheon**- a group of particularly respected, famous, or important people.

(Clayton Damon)

## 

## **1.4 References**

Graphic User Interface Images:  
 1) <https://www.pexels.com/photo/cash-dollars-hands-money-271168/>

2)<https://www.google.com/url?sa=i&url=http%3A%2F%2Fwww.josephkeen.com%2FDnD%2FQuick%2520Reference%2FBackground%2520Quick%2520Reference.pdf&psig=AOvVaw19ajT9JiUgJ8rbvOtkqzrI&ust=1584392734833000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCKjHwYOxnegCFQAAAAAdAAAAABAT>

3)<https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.pinterest.com%2Fpin%2F118430665173652740%2F&psig=AOvVaw0UdBge7mZMrsUvr7F8_NL_&ust=1584392996223000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCLCwpoKynegCFQAAAAAdAAAAABAK>

(Rami)

## **1.5 Overview (DONE)**

The rest of the SRS document will include all of our UML diagrams/flowchart, our progress with each sprint, and explanations for paths we took along the way.(Rami Alsibai)

# **2. General Description (DONE)**

Our software is easily used, needing only an internet connection and web browser. We also require data from hospitals but since a recent law change requires hospitals to display this information, making it much easier. (Keith Bragg)

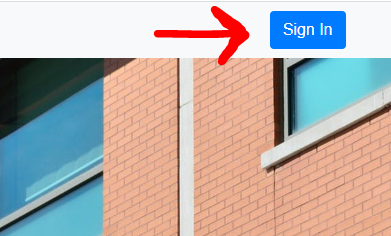
## **2.1 Product Perspective (DONE)**

Similar products online do exist but they are convoluted and hard to use. Healthcare Bluebook is a similar product but requires an access code to use and Fair Health requires an extensive survey and sensitive information. Project Eir however is a free service and does not require an access code. Project Eir is the only service of its kind that is dedicated to the fast and friendly support when searching for a medical procedure.(Clayton Damon)

## **2.2 Product Functions (DONE)**

Project Eir includes many functions that are easy to use and navigate.

\*Sign up or Sign in buttons allow users to create an account or log in using an existing account.



\*Search Bar will allow users to compare prices of medical procedures.

\*Safely stores passwords in our database to protect user information.

\*Customer support page that provides users with dedicated help all hours of the day.

(Clayton Damon)

## **2.3 User Characteristics (DONE)**

The characteristics of our target users is anyone who needs an operation or would simply like to compare prices for a future procedure. The team at Project Eir understands that many of our users may be over the age of 50, and may have limitations in their technical ability, this affects the visual aspect of our program and as such we have made an easy to use interface with a large search bar and button to accommodate these users. (Clayton Damon)

## **2.4 General Constraints (DONE)**

For this Project the biggest constraint is time, there's only so much you can do in such a short development time, because of the time constraint we had to make changes to our database and use a smaller procedure list and filter out any medical names that were too vague and hard to implement in our search algorithm. (Clayton Damon)

## **2.5 Assumptions and Dependencies (DONE)**

It is assumed that the user is familiar with an internet browser and also familiar with handling the keyboard and mouse. Since the application is a web based application there is a need for the internet browser. It will be assumed that the users will possess decent internet connectivity. (Clayton Damon)

# **3. Specific Requirements** **(DONE)**

1. Database
2. Server
3. Hospital Data
4. Log-in and log-out system
5. Search bar that queries the database
6. A system to allow general search terms to show results
7. Support System for users having problems

(Keith Bragg)

## **3.1 External Interface Requirements (DONE)**

### **3.1.1 User Interfaces**

1. Main Page with search bar and login/create account button.
2. Result page with that displays results with another search bar
3. Login Page
4. Registration Page
5. Support page

<Keith Bragg>

### **3.1.2 Hardware Interfaces (DONE)**

1. Keyboard or equivalent
2. Mouse or equivalent
3. Display

### <Keith Bragg>

### **3.1.3 Software Interfaces**

1. An interface for searching.
2. An interface for logging in.
3. An interface for registering new users.
4. An interface for submitting support tickets

<Keith Bragg>.

### **3.1.3 Software Interfaces**

### **3.1.4 Communications Interfaces**

The only communication interface in this project is the support ticket system which allows users to submit problems and receive solutions via email. <Keith Bragg>

## **3.2 Functional Requirements**

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

### **3.2.1 <Functional Requirement or Feature #1>**

3.2.1.1 Introduction

There are several requirements needed for a functional website and they must be implemented correctly to ensure full functionality without errors. Below are the requirements for Project Eir to function; they will expand as we add more functionality to satisfy our customers and provide them with even more access to procedures they need. <Keith Bragg>

3.2.1.2 Inputs

1. Input from user for searching
2. Input from user to log in
3. Input for user to register
4. Input for user to submit problems with website
5. Input from user to filter search results
6. Input from user to save procedures to profile if they are registered and logged in

<Keith Bragg>

3.2.1.3 Processing

1. Processing search terms to find synonyms
2. Processing search terms to find procedures
3. Processing hospital data to insert into database
4. Processing user passwords using salt and hashing algorithms to protect data
5. Processing user filters when searching.

<Keith Bragg>

3.2.1.4 Outputs

1. Results from searches
2. Notifying user they input an incorrect password or username
3. Notifying users no results were found from search term
4. Outputting user’s saved procedure

<Keith Bragg>

3.2.1.5 Error Handling

Errors are output to the console for the site administrator to review and fix. If a user encounters an unexpected error then they can submit a support ticket and receive help from trained support experts.

<Keith Bragg>

### **3.2.2 <Functional Requirement or Feature #2>**

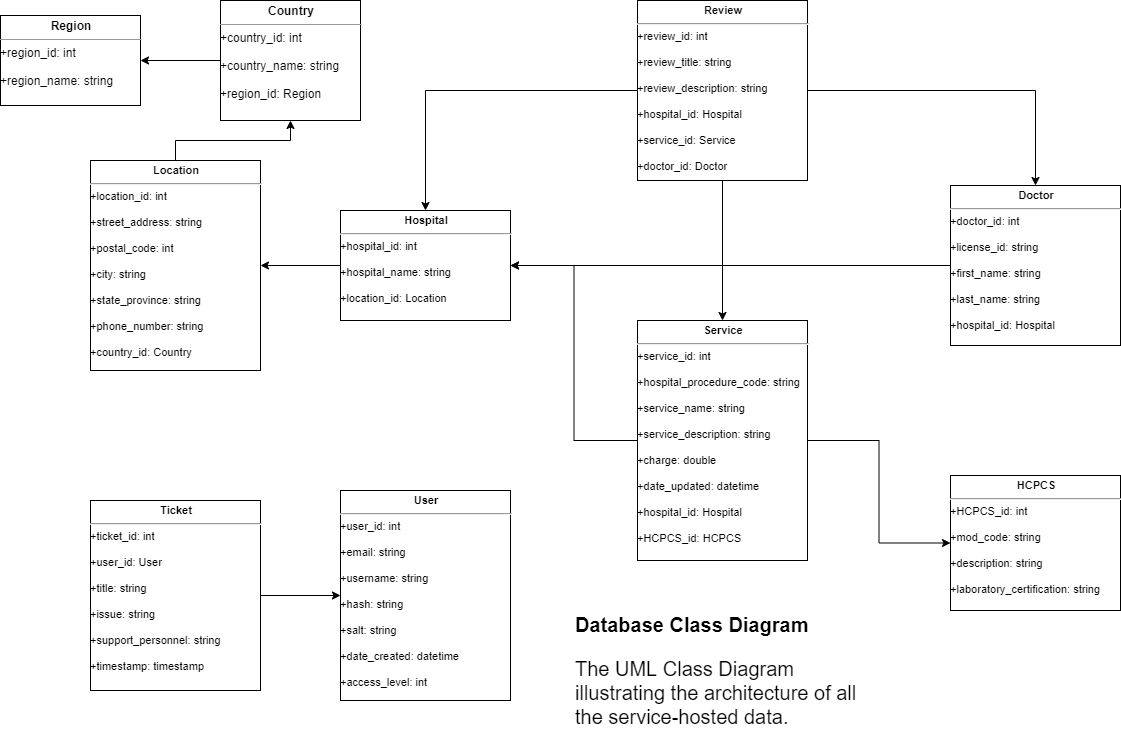
…

## **3.3 Use Cases**

The use case of our website would the user coming to the website and being greeted by the home page. On the homepage they would have multiple options to choose from, they could search right of the batt, register, login, or view the frequently asked questions page. Another feature they could utilize is the support page which they can use to submit tickets for our website.Then with the search they will be able to use alias for different medical procedures and find the results for said procedure.(Rami Alsibai)

## **3.4 Classes / Objects**

### **3.4.1 <Class / Object #1>**



3.4.1.1 Attributes

3.4.1.2 Functions

<Reference to functional requirements and/or use cases>

### **3.4.2 <Class / Object #2>**

…

## **3.5 Non-Functional Requirements**

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

### **3.5.1 Performance**

Users must be able to type in a search request and receive a response (under a normalized latency of 0ms) within 200ms. This means that the software must be able to process the request, search the database via MySQL query, and return data to the client within that amount of time.

Thus the server that the users connect to must have sufficiently-powerful hardware to facilitate this requirement. And, the code must be performant enough. Automated unit tests will ensure that common searches run under the given time constraints. (Leonardo Serrano)

### **3.5.2 Reliability**

The product must be accurately parse the user’s input and return a reliable search for medical procedures. The list of recognized procedures will grow as the product matures and the development cycle furthers.

Data stored for procedures, hospitals, departments like pricings, naming, availability, must be accurate. Continuous quality assurement must be practiced to ensure that the medical procedure data is up to date and accurate. (Clayton Damon, Leonardo Serrano)

### **3.5.3 Availability**

The product is available to anyone with a modern internet browser like Google Chrome, Microsoft Edge or Mozilla Firefox. Access to the product will require an internet-enabled device with an active connection.

It must be available in every major geographic region (Europe, North America, Asia, etc) within 50ms latency. This demands that the servers the product is hosted on be replicated and served with a geo-ip sensitive load balancer.

The product must be available to serve searches with 99.99999% uptime (Clayton Damon, Leonardo Serrano).

### **3.5.4 Security**

Users entrust the product to securely store their personal information in our database. To protect their passwords, we utilize the standard hashing and salting method to mitigate sensitive user password leaks in the event that our database is compromised. Conventional “Direct Denial of Service” attacks will be mitigated through delayed queries to the backend API; rather than blindly allowing queries to be ran, they will not run if there’s been a query ran too soon in the past.

The security of our front-facing APIs will be tested against common SQL attacks and maliciously malformed requests. (Clayton Damon, Leonardo Serrano)

### **3.5.5 Maintainability**

Quality of code will be ensured by automated unit tests and static quality analysis on code pulls to “master”. Code updates to “master” will reflect the main production server that the end users have access to. Code updates to “dev” will reflect the development server that will be used to ensure quality before changes are merged to “master”. Code will not be able to be pulled into “master” or “dev” if they fail any test cases.

To ease further contributions to the software, code standards must be established and enforced by automated “linting”. This ensures that there are no stylistic or syntactic differences in the software, such as the common “spaces” vs “tabs” indentation inconsistency.

Code reviews will periodically be performed to ensure that quality of code, even when it passes automated tests, is maintained. (Leonardo Serrano)

### **3.5.6 Portability**

The website will be available on any device that has an internet connection and a browser to view in, because it runs on a device friendly framework BootStrap that makes the page dynamic. (Rami Alsibai)

### **3.5.6 Customer Support**

TAYLOR SUPPORT STUFF HERE

## **3.6 Inverse Requirements (DONE)**

Passwords should not contain only numbers or letters.

The database should not allow for unsanitized queries.

Users will not be forced to log in to search.

Our website will not book appointments or contact doctors in anyway

<Keith Bragg>

## **3.7 Design Constraints (DONE)**

Specify design constraints imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

One major constraint of our project is our data collection. Since most hospitals attempt to make their data as hard as legally possible to access, we cannot create a standard program to collect it. Instead, we are forced to manually gather data. Another limitation is that many hospitals name their procedures differently which makes searching much more difficult. Along the same line, the sheer amount of data makes it extremely difficult to make searching easier for the layman. The last constraint is the amount of time we have, forcing us to cut planned features.

<Keith Bragg>

## **3.8 Logical Database Requirements (DONE)**

Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc

Our database requirements are not much since we are only storing text and numbers, which is why we chose MYSQL for our databases. The user database will require salted and hashed passwords and an email. The procedure table will require a procedure name, price, and hospital id as varchars, doubles, and ints respectively. The synonym table requires an ID and Name which are stored as varchars.

<Keith>

## **3.9 Other Requirements**

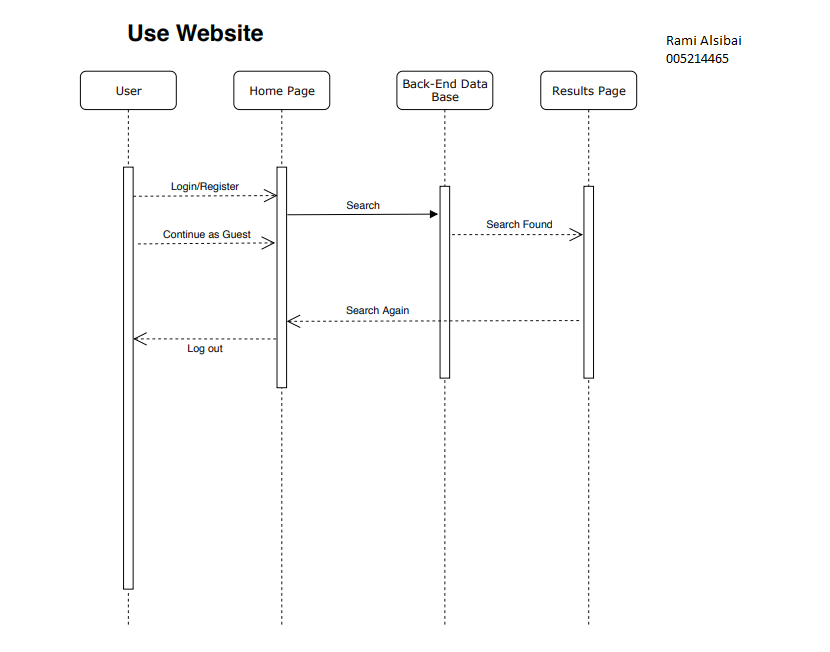
Catchall section for any additional requirements.

# **4. Analysis Models**

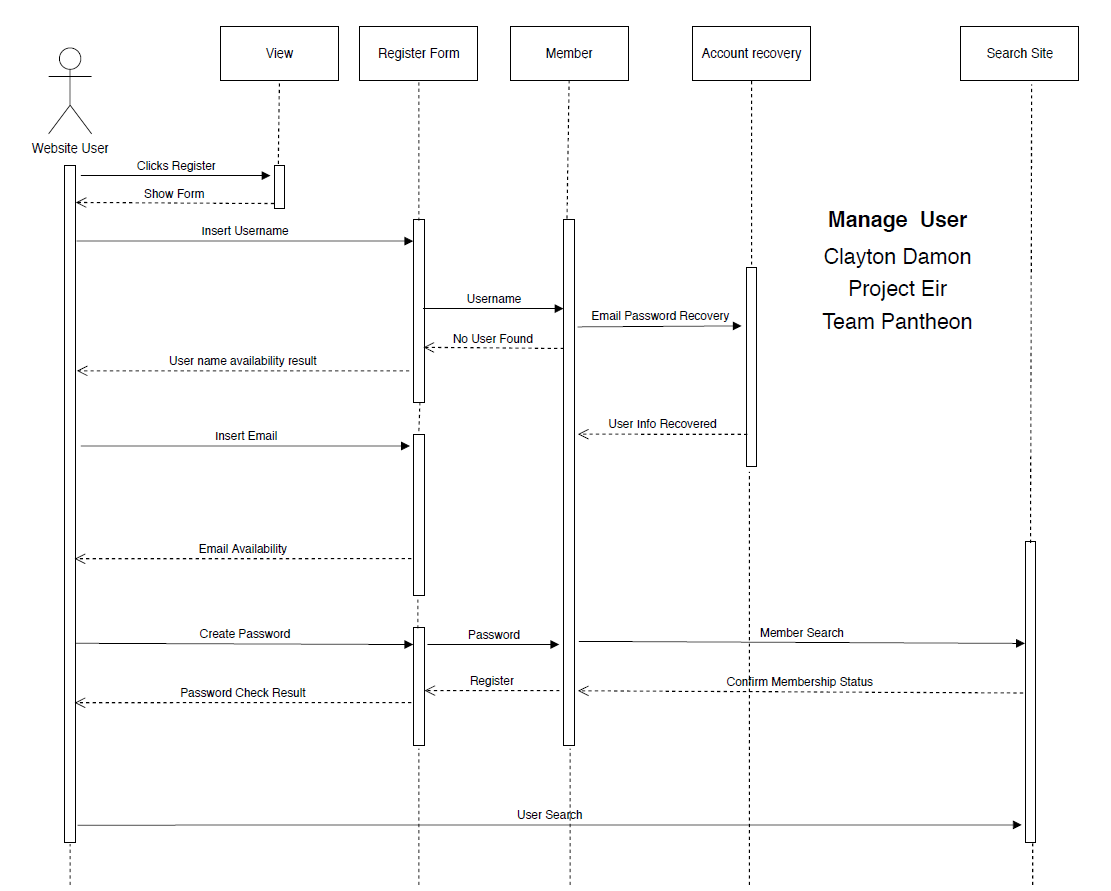
List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable to the SRS’s requirements.

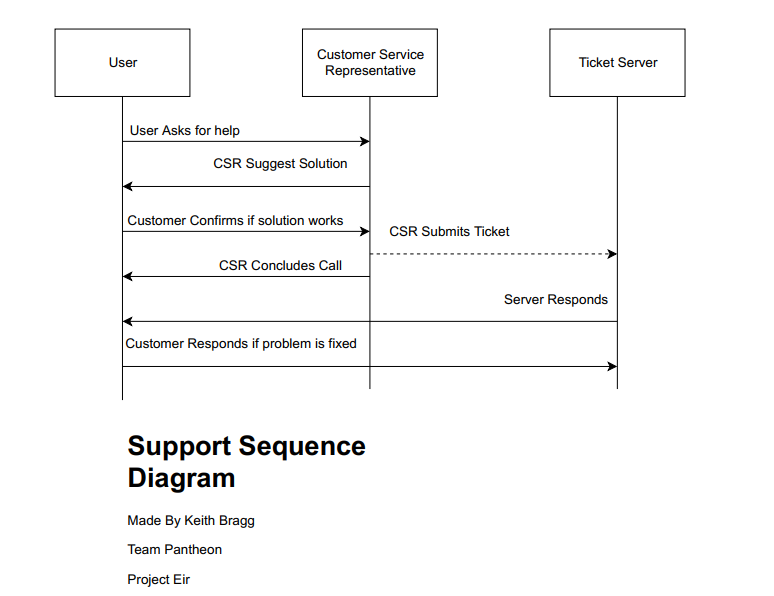
## 

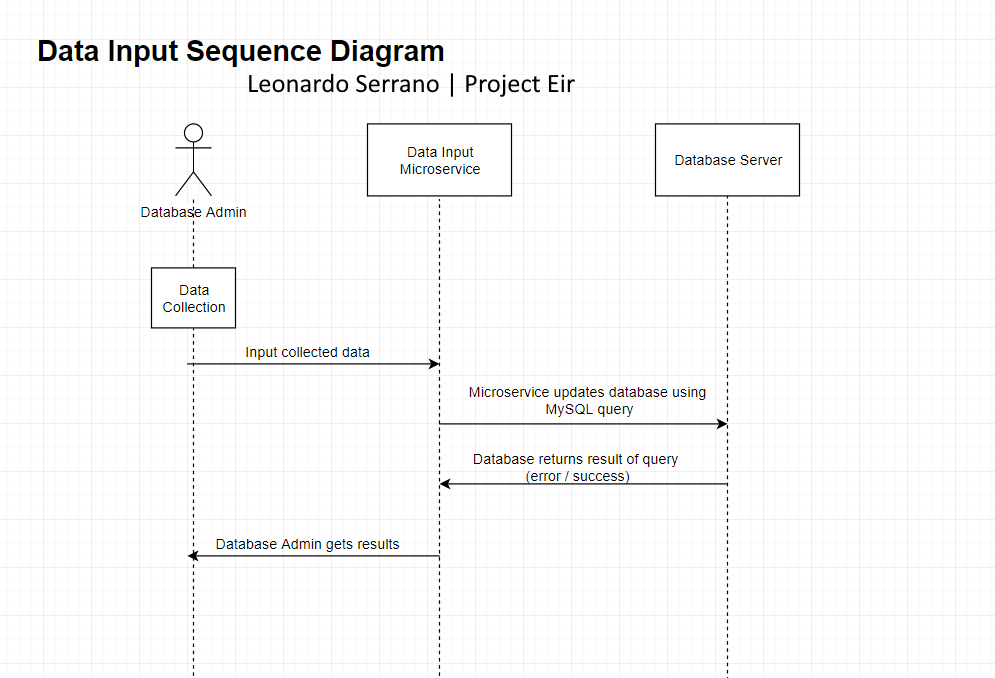
## **4.1 Sequence Diagram(s) (DONE)**

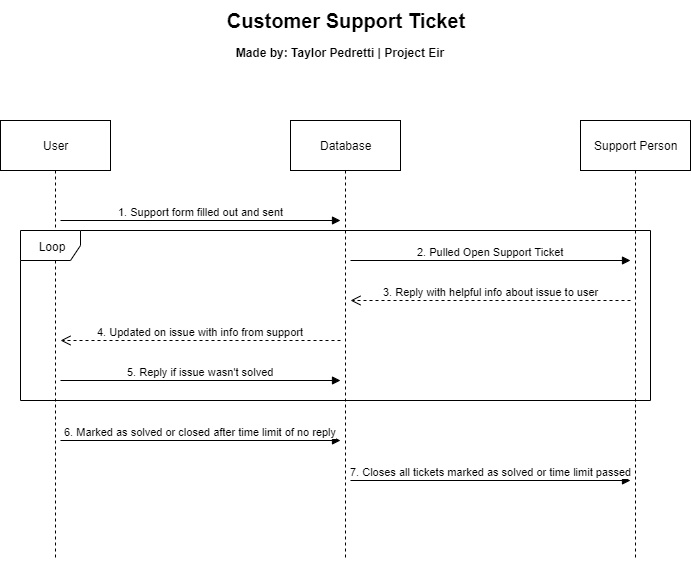


In my current sequence diagram I have the basic structure of the website and its function. The user can go through each page and will be given a new GUI with each page. I plan to add more as we get further into the project to meet the goals of the website. -Rami Alsibai

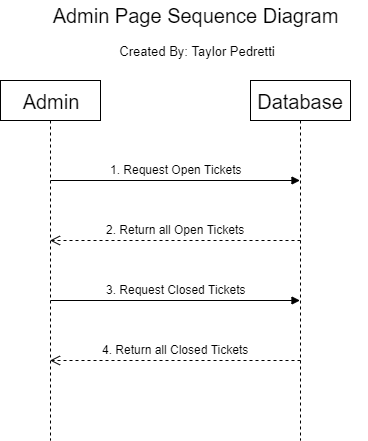
After looking back on my sequence diagram everything is still within my scope, and I will continue to work on managing the users. Sign up works and saves the users information to the database, the password is hashed to protect their information in our database. (Clayton Damon)







Support Ticket System Sequence showing the steps the user and support person have to take to get an issue solved. - Taylor Pedretti



Above is the sequence diagram of the admin page, where an admin can see open and closed support tickets.

## 

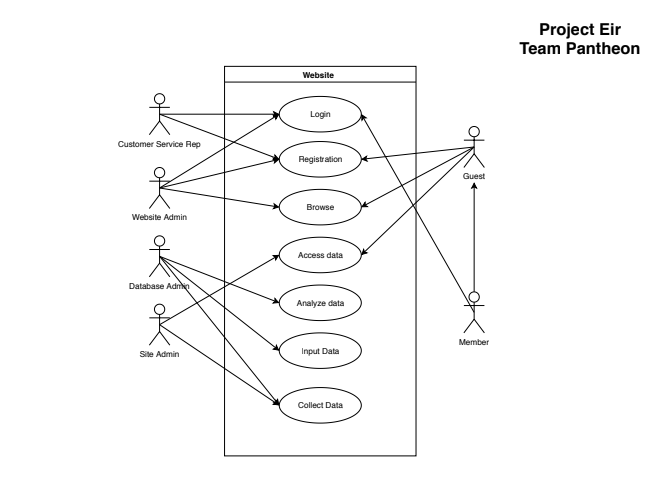
## 

## 

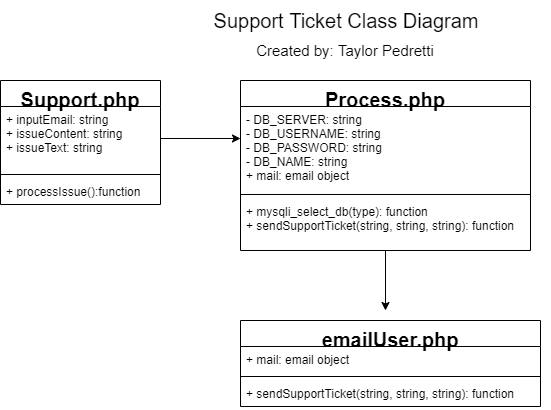
## 

## 

## **4.2 Use Case Diagram(s) (DONE)**

The Use case diagram we had for our project allowed users to choose whether or not they wanted to be a member of the website. The website is theoretically run by a few different people. We start with a customer service representative that takes care of The issues that users have with login and register. We then have a website administrator that looks over members that login and what is being browsed. Next is a Database administrator that collects, inputs, and analyzes the data for the search on the website. They ensure that the results will be as accurate as possible and that data is being pushed and pulled properly. Lastly we have a site admin that has an overview of the whole website. They ensure all parts are working properly and keep the rest in communication. (Rami Alsibai)

## **4.3 Class Diagram(s)**



Above is the class diagram for the support ticket system. It is split between three different php files so that they are modular and can be used with other files. (Taylor Pedretti)

# **5. Change Management Process**

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.

For any change that affects the scope or operation of the program

# 

# **6. Sprint Retrospective**

### **Sprint #1 Retrospective**

For sprint one most of us were still getting used to how to use draw.io for all the diagram making and figuring out what needs to be in each diagram. For the deliverable for sprint #1 Rami was working on the GUI and style of the website and sent in a basic markup on the website. The hardest thing for us during this time was the breaking down of things into smaller parts, because we’d make one diagram and then be told that the diagram could be made into multiple smaller ones.

### **Sprint #2 Retrospective**

For the second sprint we started to have a clear grasp of what we needed to do and made good progress in documenting who was doing what on the kanban board in Jira. Rami put together a design based on his previous markup. Clay further looked into PHP and became familiar with how the database works. Communication was good, professor Gonsalez in the lab helped us create clear buckets that were labeled properly. Some things we could have improved on was talking to professor earlier, so we could have had a clear understanding from the start.

### **Sprint #3 Retrospective**

By the 3rd sprint we had a working website that allowed a user to sign up and create an account that would store the user information in our database. Clay finished signup and would begin working on sign in. Rami finished the registration page layout and would add a sign in form to the GUI.

### **Sprint #4 Retrospective**

On the final sprint we all connected the final pieces that we had been working on, and had a fully functional website complete with a login and registration, Clay worked on adding error catches for misfilled or wrong information on registration and sign up. As with any project, communication and structure is key, something we could have done better as a whole is had better communication

# **7. Prototype**

# **A. Appendices**

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## **A.1 Appendix 1**

## **A.2 Appendix 2**