***Rowan University: Team Platypus***

***Balancing The Cost Of Security***

**Design Document**

**Prepared by:** Elija Amponsah.

**Version:** 1.0

**Date:** 03/14/2020

**Sponsors:** Ron

**Table of Contents**

[1. Introduction 4](#_Toc809821)

[1.1. Purpose 4](#_Toc809822)

[1.2. Scope 4](#_Toc809823)

[1.3. System Environment 4](#_Toc809824)

[1.4. UI Design 4](#_Toc809825)

[<add snapshots here> 4](#_Toc809826)

[2. Use Case Diagram 4](#_Toc809827)

[<add use case diagram here> 4](#_Toc809828)

[3. Analysis Class Diagrams 4](#_Toc809829)

[3.1. General User 4](#_Toc809830)

[<add analysis class diagram here> 5](#_Toc809831)

[3.2. Admin 5](#_Toc809832)

[<add analysis class diagram here> 5](#_Toc809833)

[4. Class Diagrams 5](#_Toc809834)

[4.1. Overview 5](#_Toc809835)

[<add class diagram here> 5](#_Toc809836)

[4.2. Main Activity and User Classes 5](#_Toc809837)

[<add class diagram here> 5](#_Toc809838)

[4.3. Report Log and File Classes 6](#_Toc809839)

[<add class diagram here> 6](#_Toc809840)

[5. Activity Diagram 6](#_Toc809841)

[<add activity diagram here> 6](#_Toc809842)

[6. Sequence Diagrams 6](#_Toc809843)

[6.1. User actions 6](#_Toc809844)

[6.1.1 <User action> 6](#_Toc809845)

[<add sequence diagram here> 6](#_Toc809846)

[6.1.2. <User action> 6](#_Toc809847)

[<add sequence diagram here> 6](#_Toc809848)

[6.1.3. <User action> 6](#_Toc809849)

[<add sequence diagram here> 7](#_Toc809850)

[6.1.4. <Admin action> 7](#_Toc809851)

[<add sequence diagram here> 7](#_Toc809852)

[6.2. File Reading and Writing 7](#_Toc809853)

[6.2.1. Write Output Report 7](#_Toc809854)

[<add sequence diagram here> 7](#_Toc809855)

[6.2.2. Write Error Log 7](#_Toc809856)

[<add sequence diagram here> 7](#_Toc809857)

[6.2.3. Read Input File(s) 7](#_Toc809858)

[<add sequence diagram here> 7](#_Toc809859)

[6.2.4. Writing Information to the Database (if applicable) 7](#_Toc809860)

[<add sequence diagram here> 7](#_Toc809861)

[6.2.5. Removing Information from the Database (if applicable) 7](#_Toc809862)

[<add sequence diagram here> 7](#_Toc809863)

[7. Glossary of Terms 8](#_Toc809864)

[8. Revision History 8](#_Toc809865)

**1.** **Introduction**

# The Software Design Document is a document to provide documentation which will be used to aid in software development by providing the details for how the software should be built. Within the Software Design Document are narrative and graphical documentation of the software design for the project including use case models, sequence diagrams, collaboration models, object behavior models, and other supporting requirement information.

**1.1. Purpose**

The purpose of this Platypus Software Design Document is to provide a description of the design of our Platypus system fully enough to allow for software development to proceed with an understanding of what we need to be built and how it is expected to build. Platypus Software Design Document provides information necessary to provide description of the details for the software and system to be built.

**1.2. Scope**

The main goal of our Platypus application is to provide business and individual software tool that could automate the task of tracking vulnerability assessment findings, security-relevant test data, and reports of incidents over time. Platypus software should highlight hotspots on the system and predict areas for future maintenance or mitigations. Having this data would allow a business to balance whether it is more long-term cost-effective to update a piece of software to a new major, modify the architecture, or replace some hardware or software over continuously patching or, otherwise, maintaining a specific machine. This Software Design is focused on software and hardware systems. Any number of users can run the same application at the same time, whether they are a member of the same group or an admin. Ease of use is a main goal with this application, so it can be used by any user.

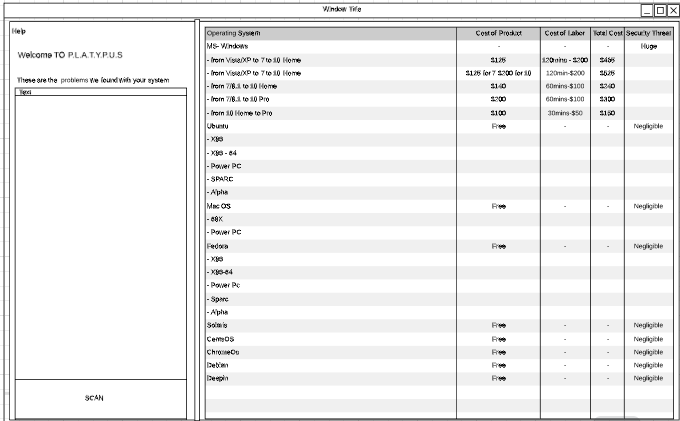
**1.3. System Environment**

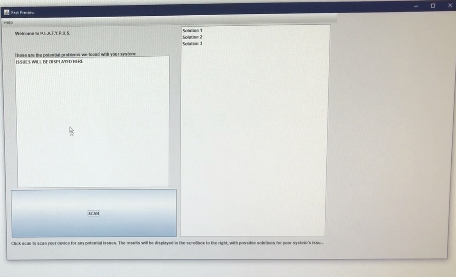
## The required Operating is Windows and Mac

* The current JavaGUI is version 1 is
* The programming language of choice is Java

**1.4. UI Design**

This section will provide a quick snapshot of how we anticipate our primary CSV files interface to look like within the Balancing the Cost of Security application. Note: the interface is developed and still under changes, so Figure 1.1 below will change and adapt further on in our development cycle.





**2. Architectural Design Views**

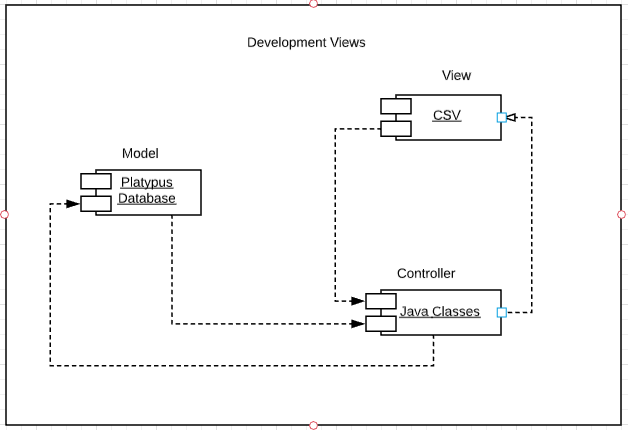
The following Architectural Design View Diagrams describe the Balancing the Cost of Security system from the viewpoint of different stakeholders, such as end-users, developers, system engineers, and project managers. These four view models illustrate the architecture of our system.

**2.1 Logical View**

The following Logical View shows key abstraction in the system as objects.

**2.2 Development View**

The following Development View shows how the software is decomposed into components for development. Figure 2.2 below is a simple Component Diagram to show how to Balance the Cost of Security was broken down into the Model-View-Controller design pattern. The main interface for end-users will be the CSV which fits the View portion. User inputs and decisions will be taken in and processed in the Controller, Java classes. It facilitates the Model, the database, based on the user decision. Nature of the MVC components made dividing up jobs for the development team very suitable.

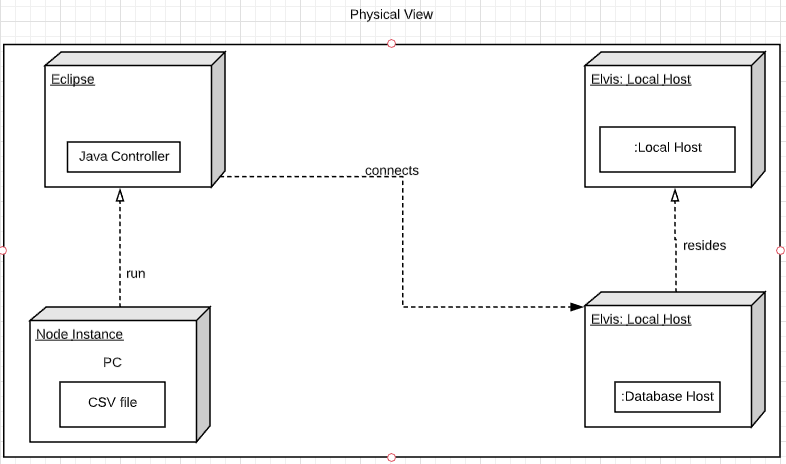


**2.3. Process View**

The following Process View shows the interactive processes of the system at runtime. Figure 2.3 below is a Communication Diagram showing how objects interact. It shows the process flow when an end-user begins to Balance the Cost of Security application.

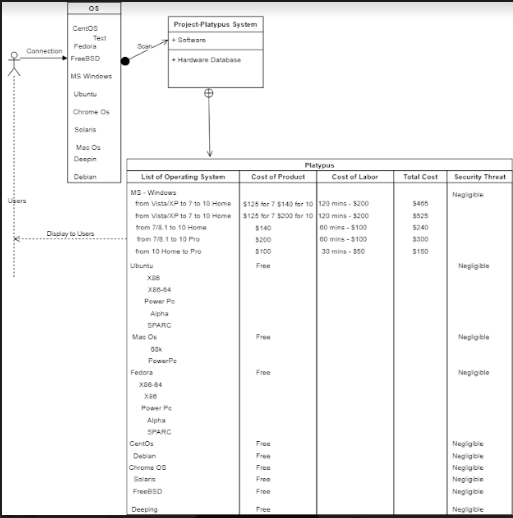
**2.4**. **Physical View**

The following Physical View shows the system from a system engineer’s perspective. It shows system hardware and how software components are distributed across the hardware. Figure 2.4 below is a Deployment Diagram which are great for visualizing the topology of the physical components of our Balancing the Cost of Security system. Our database will be residing in Elvis, Rowan University’s Sun Solaris based server that supports Unix. The main program, written in Java, will serve as the main bridge between the CSV interface and the database.



**3. Use Case Diagram**

The following use Case Diagram shows all the actions that the Actors above can perform while using the Balancing the Cost of Security application. Figure 3.1 Below helps visualize the different types of roles and show how these roles interact within Balancing the Cost of Security application. A General User can scan their operating system or their hard drive, our system will be tracking vulnerability assessment findings, security-relevant test data, and reports of incidents over time



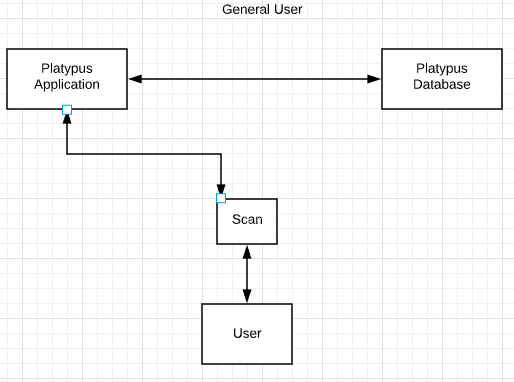
***4*. Analysis Class Diagrams**

# The following Analysis Class Diagrams visually represents a use case’s behavior, showing both participating classes and software behaviors. Like many other UML diagrams, Analysis Class Diagrams depict how stereotyped objects talk to each other.

# 

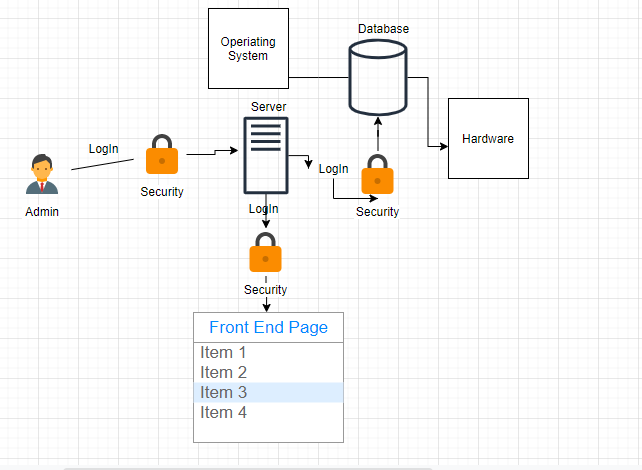
# 4.1. General User

**Figure 4.1** Below show all classes a General User will interact with.



***4*.2. Admin**

The Following Analysis Class Diagram shows all the classes a Group Admin will interact with.

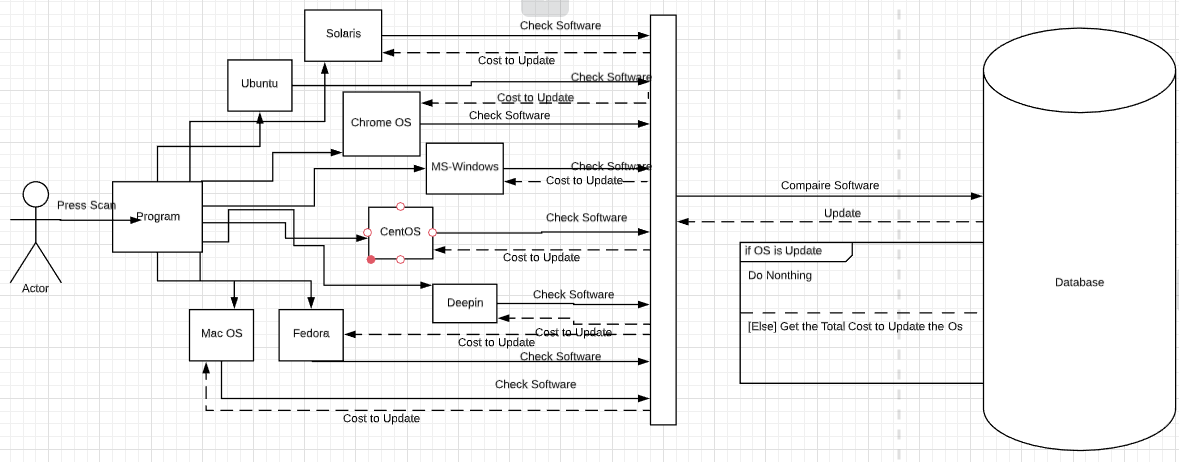


## 

## 5. Class Diagrams

## 5.1. Overview

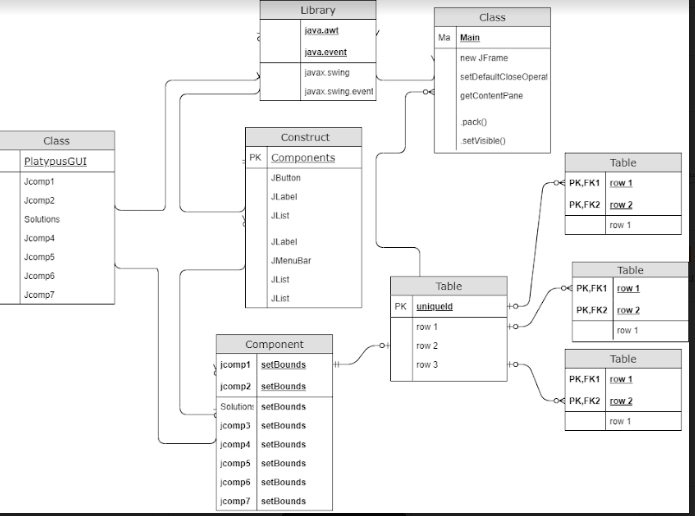
This is an overview of all the Classes used within the Balancing the Cost of Security application. This overview does not include fields and functions to give a quick insight on how the classes interact.



## 

## 5.2. Main Activity and User Classes

Shows Class Diagrams for the Main Activity and all Classes associated with a User to show how these classes interact.



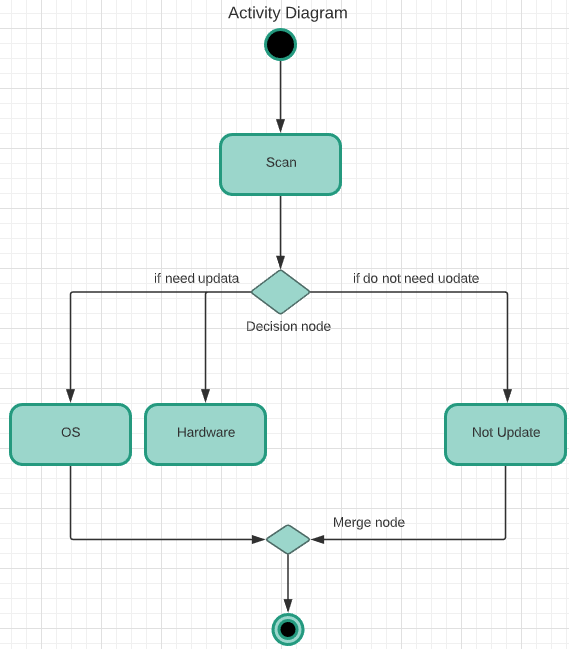
## 5.3. Report Log and File Classes

This Balancing the Cost of Security application Class Diagrams for all Classes associated with the Report Log to show how these classes interact.

#### 

# 6. Activity Diagram

This Activity Diagram is used to model system behaviors throughout Balancing the Cost of Security application. Shows step-by-step activities a user can perform while using this application.

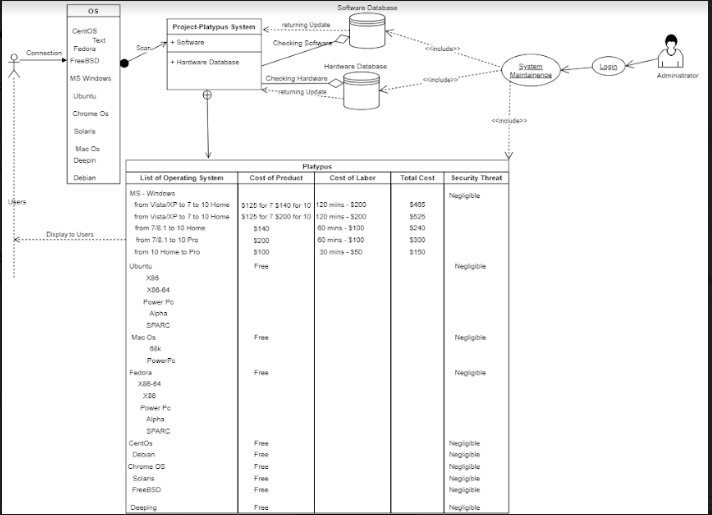


**7. Sequence Diagrams**

#### 7.1. User Actions

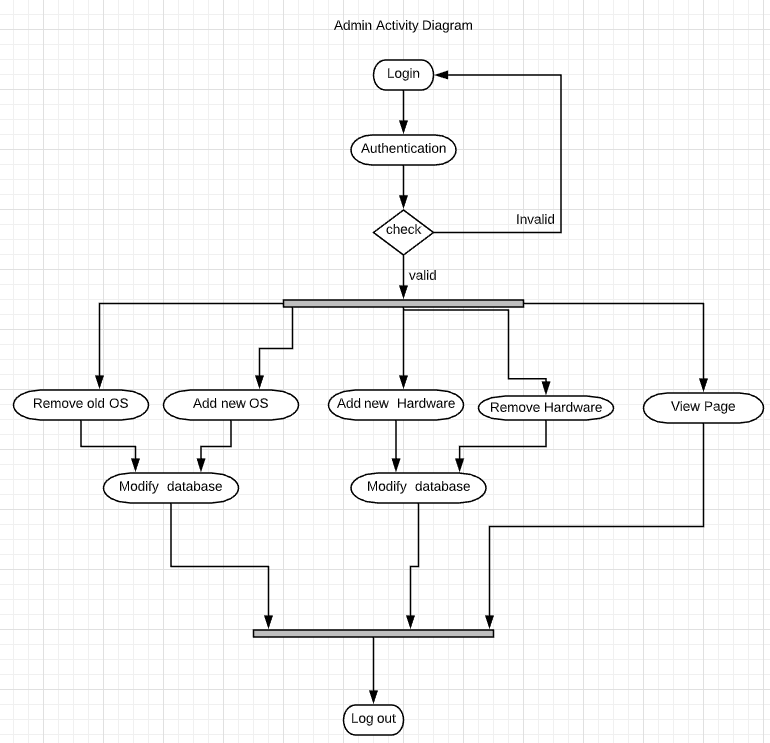
### 7.1.1 User Action

This is a User action Sequence Diagram of Balancing the Cost of Security application process



### 7.1.2. Admin action

The Sequence Diagram of Balancing the Cost of Security application Admin action process.

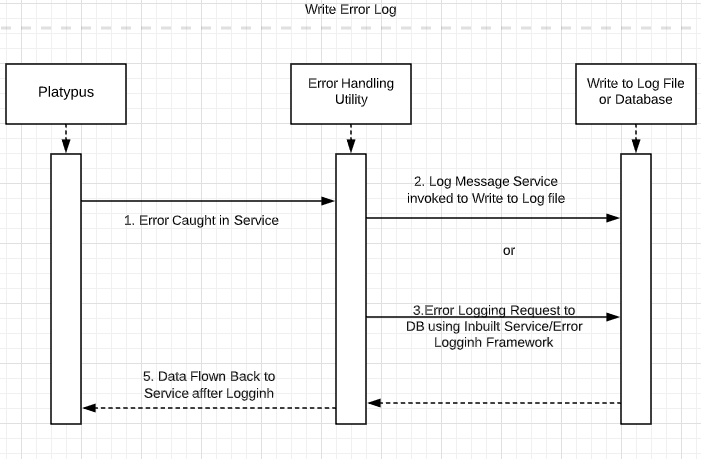


## 

## 7.2. File Reading and Writing

### 7.2.1. Write Error Log

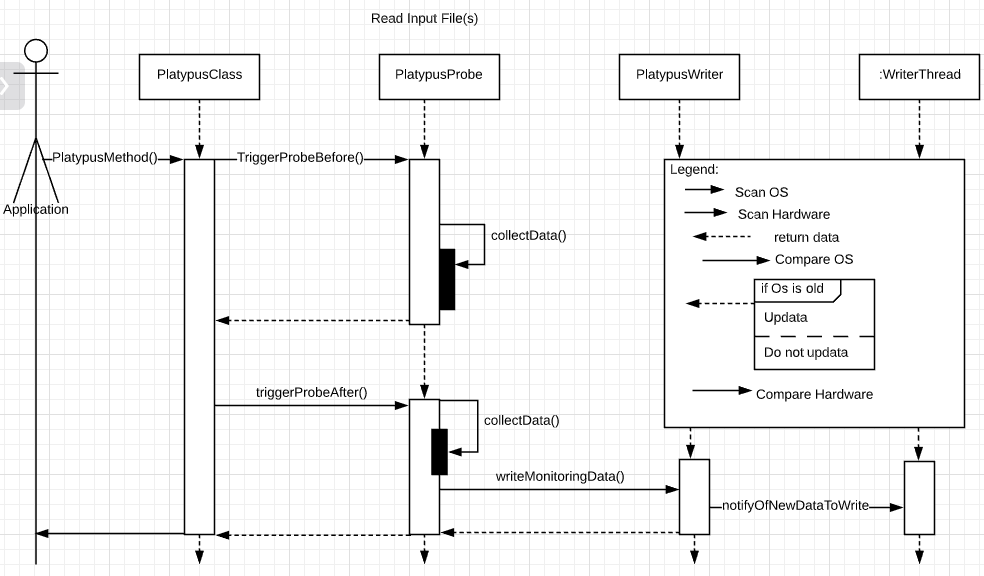
Sequence Diagram shows the Write error log file process within the Balancing the Cost of Security application.



### 

### 7.2.3. Read Input File(s)

This a Sequence Diagram shows the Read Input File process within the Balancing the Cost of Security application.

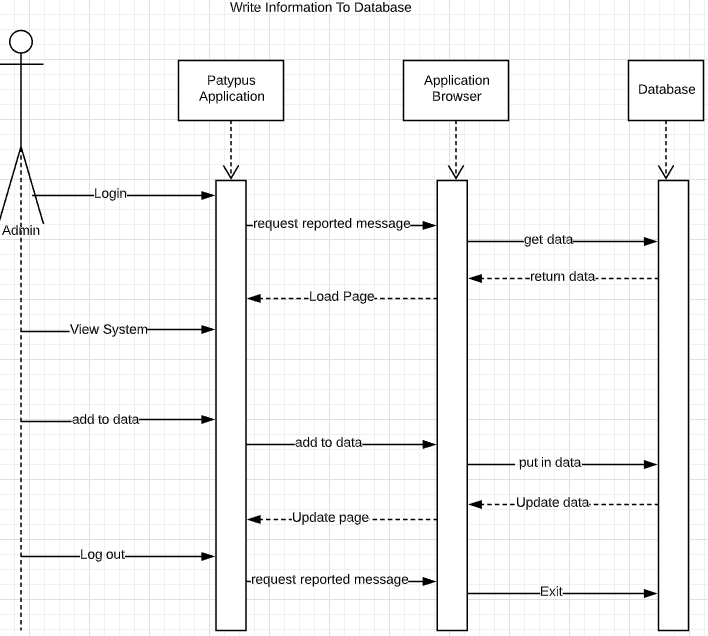


.

#### 

#### 7.2.4. Writing Information to the Database

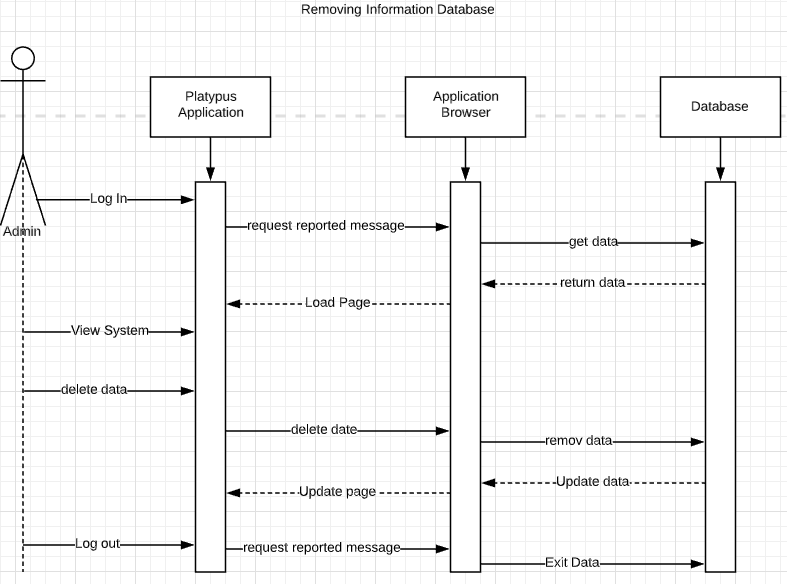
This is a Sequence Diagram on how information is added to Balancing the Cost of Security Database within the Balancing the Cost of Security application.



.

#### 7.2.5. Removing Information from the Database

This is a Sequence Diagram on how information is removed from Balancing the Cost of Security Database within the Balancing the Cost of Security application.



# 8. Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Updated By** | **Comments** |
| 1.0 | ***MM/DD/YYYY*** | ***Team Member*** | Initial document creation |
|  | 03/18/2020 | Program Owner – John Scrum Master – Patrick  Development Team - Kevin, Eli, Tyler, Isaac | Elija Amponsah |