CIS 4911 - Senior Project (U01)

Web Dashboard for Addigy

Javier Carmona

Francisco Marcano

September 8, 2014

Jason Dettbarn

**Requirements Document**

# 

# Copyright and Trademark Notices

All Addigy specific logos and slogan material is copyright Addigy, Inc. All other materials developed for this project is available open source under MIT License:

http://opensource.org/licenses/MIT

# 

# Abstract

This document will help to outline and explain all of the functional components for the Web Dashboard for Addigy project. The document will be broken down into different chapters each of which will be in charged of explaining different aspects.

The first chapter includes an introduction, which will serve to explain the problem definition, the scope of the project, any terminology used and an overview of the rest of the document. Chapter two will explain the current system that is implemented. The third chapter talks about the project plan which helps to define the project’s organization, work breakdown, and cost. The fourth chapter talks about system requirements and functional dependencies, as well as any models in the system. Chapter five includes a glossary of any terminology used in this document. Finally, chapter six includes all of the different diagrams used to visualize the system.

# 

# Table of Contents

|  |  |
| --- | --- |
| **Chapter** | **Page** |
| **1. Introduction** | **3** |
| 1.1. Problem Definition. | 3 |
| 1.2. Scope of system. | 3 |
| 1.3. Terminology | 3 |
| 1.4. Overview of document | 3 |
| **2. Current System** | **5** |
| **3. Project Plan** | **5** |
| 3.1. Project organization | 5 |
| 3.2. Work breakdown | 5 |
| 3.3. Cost Estimate | 5 |
| **4. Proposed System Requirements** | **6** |
| 4.1. Functional Requirements | 6 |
| 4.2. Analysis of System Requirements | 6 |
| 4.2.1. Scenarios | 7 |
| 4.2.2. Use case model | 7 |
| 4.2.3. Static model | 7 |
| 4.2.4. Dynamic model | 7 |
| **5. Glossary** | **8** |
| **6. Appendix** | **8** |
| 6.1. Appendix A - Complete use cases | 8 |
| 6.2. Appendix B - Use case diagram using UML | 8 |
| 6.3. Appendix C - Static UML diagram | 8 |
| 6.4. Appendix D - Dynamic UML diagrams | 8 |
| 6.5. Appendix E - User Interface designs. | 9 |
| 6.6. Appendix F - Diary of meeting and tasks. | 9 |
| **7. References** | **10** |

# 

# 1. Introduction

The WDA project will be a client side web application. Since the project is mostly client side based, a variety of web frameworks will be used in order to provide high level functional structures (such as MVC) in order to create a flexible and scalable application. Also, a variety of agile development techniques will be used in the development of this project in order to ensure a fast delivery time and more client-developer interaction.

The Web Dashboard itself will contain statistical data gathered from computer clusters under a unified network. This provides high level professionals with important information about the computers that are currently running in the network. Using this data, these professionals can better assess different problems that may arise in the lifetime of their networks.

## 1.1. Problem Definition

Many industry professionals in Information Technology fields lack the tools necessary to successfully manage mac computers in their networks. The Addigy product aims to solve that problem by providing statistical data on any machine that is present in their network.

Currently the Addigy product does not have any user friendly way of showcasing their statistical data to these high level professionals in the industry. The Web Dashboard for Addigy aims to solve that problem by showing the aforementioned statistical data in a nice and clean format, that is easy for any professionals in the industry to follow.

## 1.2. Scope of System

The scope of the system for this project covers minor changes to the existing system’s implementation to provide meaningful data to the users if not currently present. Also, all changes needed to the Web Dashboard in order to deliver a product that meets the client standards. Finally, any revisions needed in order to provide the client with a product they are happy with without going into scope creeping.

## 1.3. Terminology

Some of the acronyms or abbreviations used throughout this document can be found detailed here:

* **WDA**: Web Dashboard for Addigy
* **REST**: Representational State Transfer
* **API**: Application Programming Interface

## 1.4. Overview of Document

This document will serve to outline and describe all of the project’s requirements. Things to be expected include the following:

* Functional breakdown of the project and all of its different components
* Any functional costs that show up on the different aspects of the project
* Work breakdown based on team members to ensure everyone on the team is contributing equally to the product
* Use cases that match all of the possible uses that high level professionals are interested in when using the WDA product
* High level models to visually represent the system
* The project timeline along with some information relating to due dates for deliverables.

# 

# 2. Current System

REST API Calls will be made via the front-end AngularJS dashboard for updated data.

# 

# 3. Project Plan

This chapter will discuss how the work for the Addigy Web Dashboard will be organized among the team members. It will break down all tasks needed for this project and make assignments. Lastly, it will estimate the cost of developing this project.

## 3.1. Project Organization

The work for this project will be organized among Javier Carmona and Francisco Marcano. Both will serve as developers and project managers for each other.

## 3.2. Work Breakdown

The dashboard will be divided into two main components. The main dashboard, displaying general information about groups of computers in a network of an organization, and the individual computer which shows more in depth, real time, information regarding one specific computer. Both of these will be divided into different modules to work on with Francisco Marcano taking the majority of the modules of the main dashboard, and Javier Carmona taking on the majority of the modules in dashboard for one specific computer. Both members will work together on all the deliverables of this project.

## 3.3. Cost Estimate

The Addigy Web Dashboard will be developed using an underlying infrastructure that is already in place. There will be no costs for the dashboard itself.

# 

# 4. Proposed System Requirements

In this chapter, this document will explore the different functional requirements of the system being proposed. The requirements will be analyzed using different scenarios and models.

## 4.1. Functional Requirements

* **Functional requirements:**
  + The system shall display what status the computers are in from the map (green/healthy or red/issue)
  + The system shall show how many machines have encrypted disk or are unencrypted
  + The system shall show what is the make-up of the Mac assets such as macbook air, macbook pro, mac mini, etc.
  + The system shall show the uptime of the machines such as the time that have been online in the last hour / day / week.
  + The system shall show Apple warranty data such as whether the machines are still under warranty, warranty expiring within 30 days, or out of warranty.
  + The system shall display where are the computers located geographically.
  + The system shall display alerts based on critical machine states.
  + The system shall allow users to mark alerts as acknowledged.
  + The system shall allow users to mark alerts as resolved.
  + The system shall allow users to re-assign alerts to other Admin users.
  + The system shall be able to show real time information for a single computer.
  + The system shall show the different processes running on a single computer.
  + The system shall be able to start or stop different processes running on a single computer.
  + The system shall be able to run and report bandwidth stats such as download and upload speeds.
  + The system shall report what ports are being used in the target machine and report the ports with the used protocols.
  + The system shall display what network ports are being used in the machine and what protocols are being utilized.
  + The system shall provide a chat for the administrator to interact with the user of the machine.
* **Non-functional requirements:**
  + The system shall use HTML, CSS, and JavaScript.
  + The system must keep itself up to date to no more than 30 minutes delay.
  + The system shall poll the Addigy web service for updated data, and render the changes on the dashboard.
  + The system shall be able to understand HTTP/HTTPS protocols.
  + The system shall be able to interpret JSON notation.
  + The system shall use Google Map APIs.

**4.2. Analysis of System Requirements - Analysis models – contains the complete functional specification and is mainly for the designers and programmers. This section describes the diagrams in the Appendices B - D and validates the models against the use cases.**

In this section, we will look at the different fictional scenarios that could occur while using the Addigy Web Dashboard. Then we will examine the Use Case Model of the dashboard. Lastly, we will go over the static and dynamic models of the software.

## 4.2.1. Scenarios

This section will contain some different scenarios the user might find themselves in when using the WDA product. This scenarios are listed in no particular order based on weight or importance. After the first scenario, which serves to show the process from a logged out state, users will be assumed to have logged into the application.

1. Bob uses his web browser and navigates over to the WDA project website. Once in the login screen, Bob uses his credentials to log in. Once logged in, Bob is presented with the Dashboard showing information.
2. Chris is in the home page for the application and decides to change some settings/preference. On the home screen, Chris clicks on the gear icon and he is taken to the settings/preferences page in which he is able to change any settings and/or preferences listed in the page. Once Chris is happy with the changes he made, he can save the changes by clicking on the “Save” button.
3. Mike wants to geolocate the computers that are having problems in order to send IT professionals to fix them. Once in the home page, Mike can look for the map feature in the page in order to see where in the map the computers are located. Hovering or clicking on the computers on the map will change the dashboard to showcase individual data for that particular computer. At this time, Mike can quickly see crucial information about the computer and successfully assess the problem and come up with a solution that matches the statistical data provided by the application.
4. John is looking at the dashboard and sees that on the map feature, there is a group of computers on one location with one computer having issues. He wants to only look at the specific group of computers in order to help him troubleshoot the issue. John clicks on the problematic location of the map, which makes the information on the dashboard be about the computers in the selected location only.
5. Michael is looking at the group of computers in a specific location of the map and sees that one computer is struggling with the available memory. He decides that he wants to see statistics about this computer only in order to help him identify what is causing the memory issues. Michael clicks on the name of the specific machine and the dashboard starts displaying real time information about the selected machine.
6. Ariel is looking at a specific computer that is having issues with memory. As he looks through the processes running in that specific machine, he notices that a non-important process has allocated most of the memory of the system. Ariel decides that he will terminate the process he identified by clicking the “kill button” for the specific process.
7. Ralph wants to backup an important folder for a specific machine that he is looking at from the dashboard. He enters the command he wants to execute in order to backup the target folder and it is executed on the machine through the dashboard.

**4.2.2. Use case model**

Refer to **Appendix A** and **Appendix B.**

The use cases show how a system administrator and the user of a machine interact with the different features of the Addigy dashboard. System administrators deal directly with the main Addigy dashboard in order to see commulative information about all the machines of an organization, and the single machine dashboard to deal with features that are specific to one machine such as bandwidth tests and processes. The user of the machines interact with the system through the chat capabilities provided by the single machine agent that was installed in the machine through the Addigy services.

**4.2.3. Static model**

Refer to **Appendix C.**

The static model shows what are the different classes present on the Addigy Dashboard system, what are their capabilities, and the relationship between them. We attempt to reveal the interactions between the Administrator, the Main Dashboard, the Single Machine Dashboard, the existing Addigy Services, and the Publish-Subscribe architecture.

## 4.2.4. Dynamic Model

Refer to **Appendix D.**

This section shows the sequence diagrams and state diagrams for the Addigy dashboard. It reveals the sequence of execution of the system as different scenarios play out. It also attempts to clarify different interactions between the different subsystems that are interacting with the dashboard such as the Addigy Web Services, PagerDuty, and PubNub.

# 

# 

# 

# 

# 5. Glossary - define terms used in document, especially domain specific terms.

**Fact**: A piece of information about a machine.

**Agent:** A program that acts on the user’s computer on behalf of the administrator.

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 6. Appendix

**6.1. Appendix A - Complete use cases**

***Log In***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user is standing in the Login page.

**Flow of Events**

The user enters his username in the username field, his password in his password field, and clicks on the login button.

**Exit Condition**

The user is successfully logged in.

**Exceptions**

An error occurred and an error message is being displayed.

**Nonfunctional Requirements**

Login form must validate user’s input.

***Examine Dashboard***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user is logged in

**Flow of Events**

The user logs in and is redirected to the dashboard page where the different statistics of the network can be seen.

**Exit Condition**

The user saw the status of the network.

**Exceptions**

None

**Nonfunctional Requirements**

The information of the dashboard must not be older than 30 minutes.

***Examine Machines in Location***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user has logged in and is looking at the main dashboard.

**Flow of Events**

User finds the location of the machines he wants to see in the map and clicks the specific location. The rest of the dashboard now displays information about the computers in the selected locations only.

**Exit Condition**

The user sees the information about the group of machines he is interested in.

**Exceptions**

None

**Nonfunctional Requirements**

None

***Acknowledge Machine Alerts***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user is looking at the errors in the main dashboard

**Flow of Events**

The user is presented with the alert in the main dashboard, the user then clicks on the acknowledge button for the alert of their interest

**Exit Condition**

The system received confirmation that the alert was acknowledge

**Exceptions**

The alert is not successfully acknowledge on the PagerDuty framework

**Nonfunctional Requirements**

The alert information should be polled every 10 seconds

***Resolve Machine Alerts***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user is looking at the errors in the main dashboard

**Flow of Events**

The user is presented with the alert in the main dashboard, the user then clicks on the resolve button for the alert of their interest

**Exit Condition**

The system received confirmation that the alert was resolved

**Exceptions**

The alert is not successfully resolved on the PagerDuty framework

**Nonfunctional Requirements**

The alert information should be polled every 10 seconds

***Re-Assign Machine Alerts***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user is looking at the errors in the main dashboard

**Flow of Events**

The user is presented with the alert in the main dashboard, the user then clicks on the assign dropdown and selects a the user he wishes to re-assign the alert to.

**Exit Condition**

The system received confirmation that the alert was re-assigned

**Exceptions**

The alert is not successfully re-assigned on the PagerDuty framework

**Nonfunctional Requirements**

The alert information should be polled every 10 seconds

***Examine Single Machine***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user has logged in and is looking at a group of computers in a specific location.

**Flow of Events**

The use identifies the machine he wants to zoom into and clicks it. The dashboard now displays real time information about that specific machine.

**Exit Condition**

The user sees specific information to one machine.

**Exceptions**

Could not connect to the machine.

**Nonfunctional Requirements**

Shall not affect performance of the dashboard or the target machine.

***Start Process in One Machine***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user has logged in and is looking at the specifics of one machine.

**Flow of Events**

The user enters a command in the command input field and submits the command for execution. The command is executed in the target machine.

**Exit Condition**

The user starts a desired process in the target machine.

**Exceptions**

Fail to deliver command.

Failed to connect to the machine.

**Nonfunctional Requirements**

The command should be executed on the target machine within 10 seconds.

***Terminate Process in One Machine***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user is looking at a specific machine and its processes. User decided that one process must be terminated.

**Flow of Events**

User clicks on the process he desires to terminate and clicks on the Terminate button for that process. The target machine executes a command to kill the target process.

**Exit Condition**

The process is terminated on the target machine.

**Exceptions**

Fail to deliver command.

Failed to connect to the machine.

**Nonfunctional Requirements**

Process must be terminated within 10 seconds of the button being clicked.

***Get Machine Bandwidth***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The user is looking at a specific machine and need to know the download and upload speeds of the machine.

**Flow of Events**

User expands the Speed Test module of the single machine dashboard. The dashboard requests a speed test from the target machine and reports the results to the user.

**Exit Condition**

The user sees the download and upload speeds of the machine he is monitoring.

**Exceptions**

Fail to deliver speed test request.

Failed to connect to the machine.

**Nonfunctional Requirements**

None

***Chat With Machine User***

**Actors**

Admin: The network administrator inspecting the user’s machine.

User: A user is the user of the machine.

**Entry Condition**

The administrator is looking at a specific machine and needs to communicate with the person using the machine.

The user is working on his machine.

**Flow of Events**

Administrator expands the Chat module of the single machine dashboard. The dashboard reveals a chat window for the administrator to type and send a message. The administrator types a message and clicks on the send button. The dashboard sends the message to the machine. The machine opens a chat window with the administrator’s message and capability for the user to enter and send a message for the administrator. The user types and message and sends it to the administrator. The machine sends the message to the dashboard. The dashboard shows the message to the administrator through the chat module.

**Exit Condition**

The user and the administrator have exchanged messages.

**Exceptions**

Fail to deliver message.

Failed to connect to the machine.

**Nonfunctional Requirements**

None

***See Active Ports in the Machine***

**Actors**

User: A user is the administrator of a network.

**Entry Condition**

The administrator is looking at a specific machine and needs to know what ports are being used.

**Flow of Events**

The user expands the Netstats module of the dashboard. The dashboard requests a request for the ports of the machine to be scanned. The machine scans its ports and reports what ports are in use and the protocols being used. The dashboard displays the results of the scan.

**Exit Condition**

The user sees what ports are open in the machine and what protocols are being used.

**Exceptions**

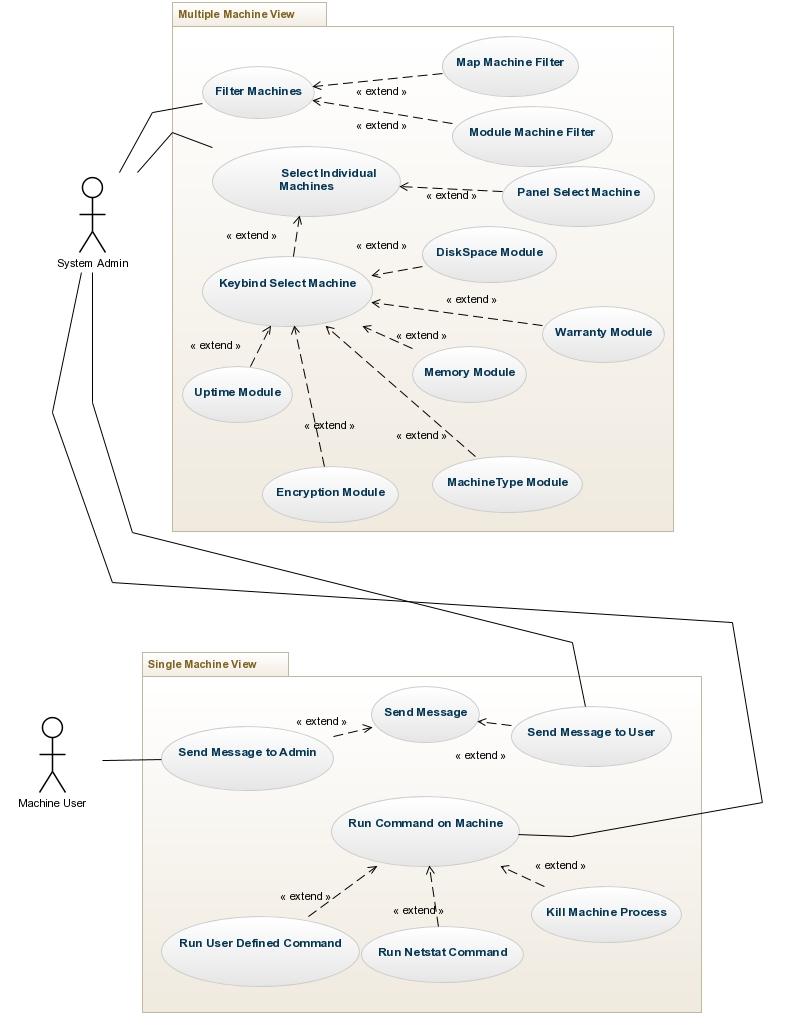
Fail to deliver Netstat request..

Failed to connect to the machine.

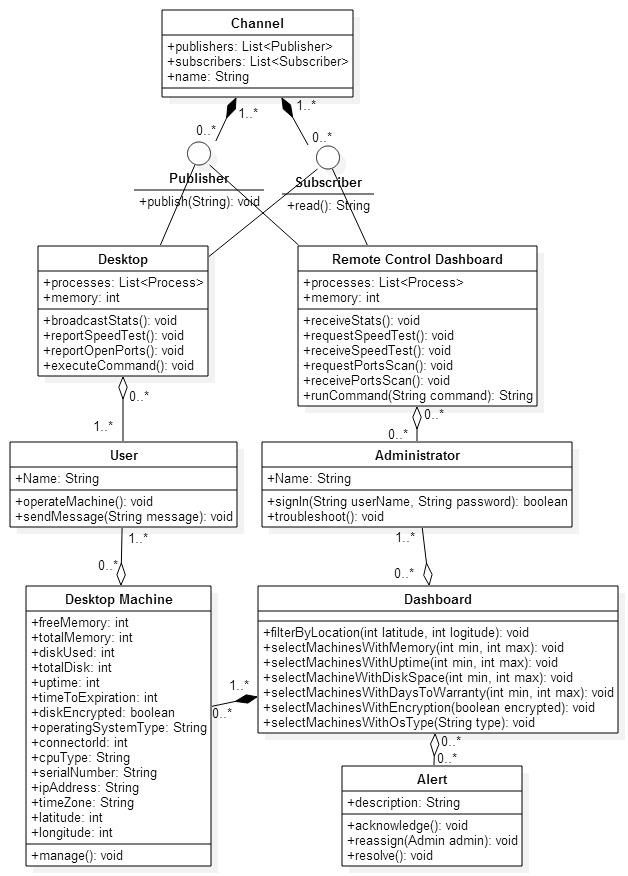
**Nonfunctional Requirements**

None

**6.2. Appendix B - Use case diagram using UML**



**6.3. Appendix C - Static UML diagram**

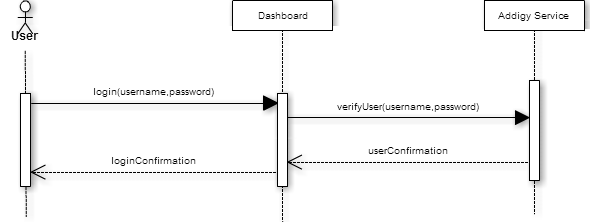


**6.4. Appendix D - Dynamic UML diagrams**

**SEQUENCE DIAGRAMS**

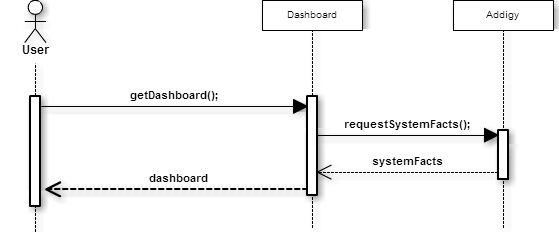
**Logging in**

The user is currently in a logged out state and successfully logs into the application.



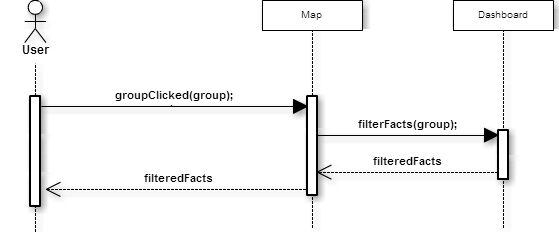
**Examine State of the System**

User is logged in and sees all the computers in the network as well as all the statistics regarding those computers in the web dashboard.



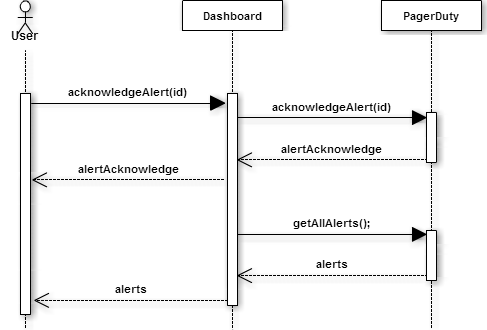
**Examine Computers in One Location**

User has seen the computers in the map and wants to see the stats for one specific computer. User clicks on one location in the map.



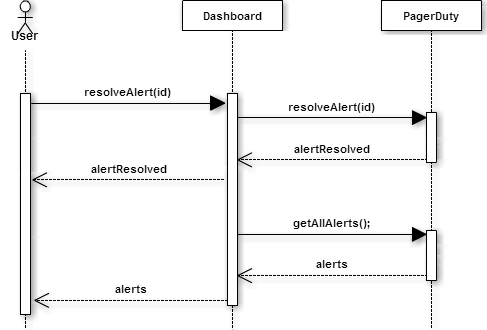
**Acknowledge a Machine Alert**

The user has seen the machine alert on the dashboard but understands it is an error that can be set aside for later.



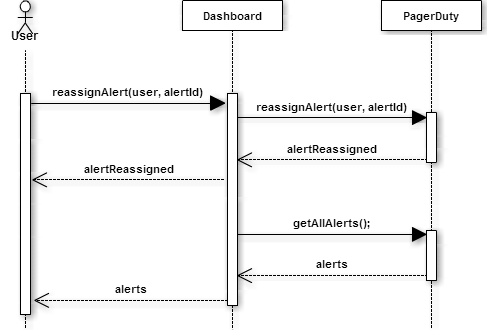
**Resolve a Machine Alert**

The user has seen the machine alert on the dashboard but understands it is an error that was already resolved.



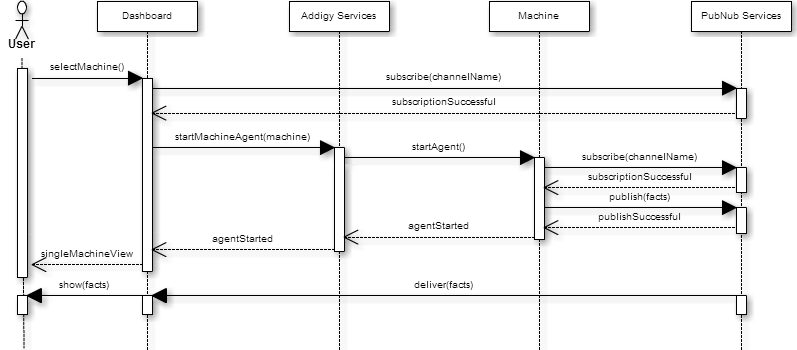
**Re-Assign a Machine Alert**

The user has seen the machine alert on the dashboard and wants to reassign it to someone else.



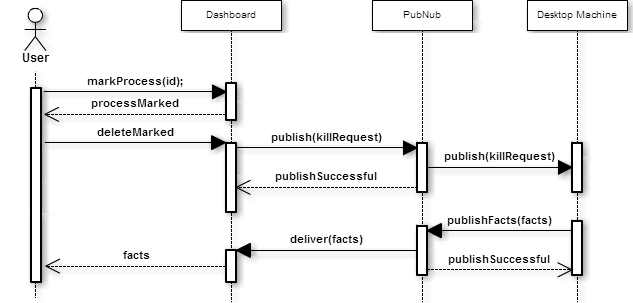
**Examine Single Computer**

User is looking at the stats for the machines in one location and wishes to narrow the stats to a single machine.



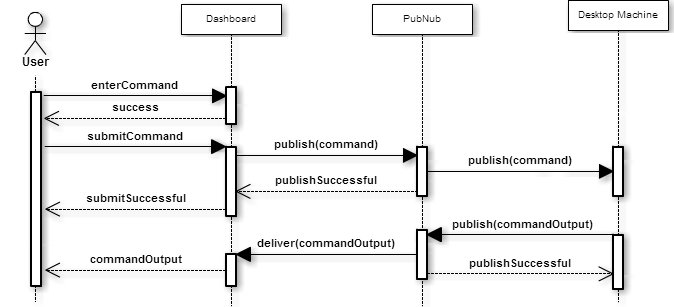
**Kill Process in One Machine**

User is looking at the stats for one specific machine and decided that one processes must be terminated.



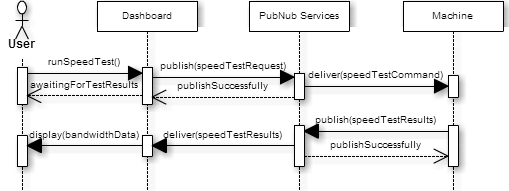
**Start Process in One Machine**

User is looking at the stats for one specific machine and wants to run a specific program in it.



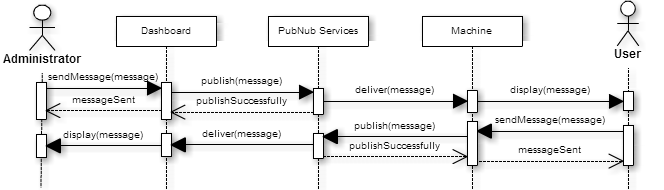
**Run speed test in one machine**

User wants to see the bandwidth speed of one machine and expands the speed test module.



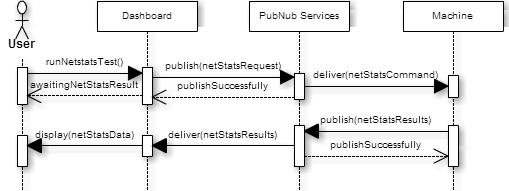
**Chat with User of the Machine**

Administrator communicates with the user of the machine through the chat module.



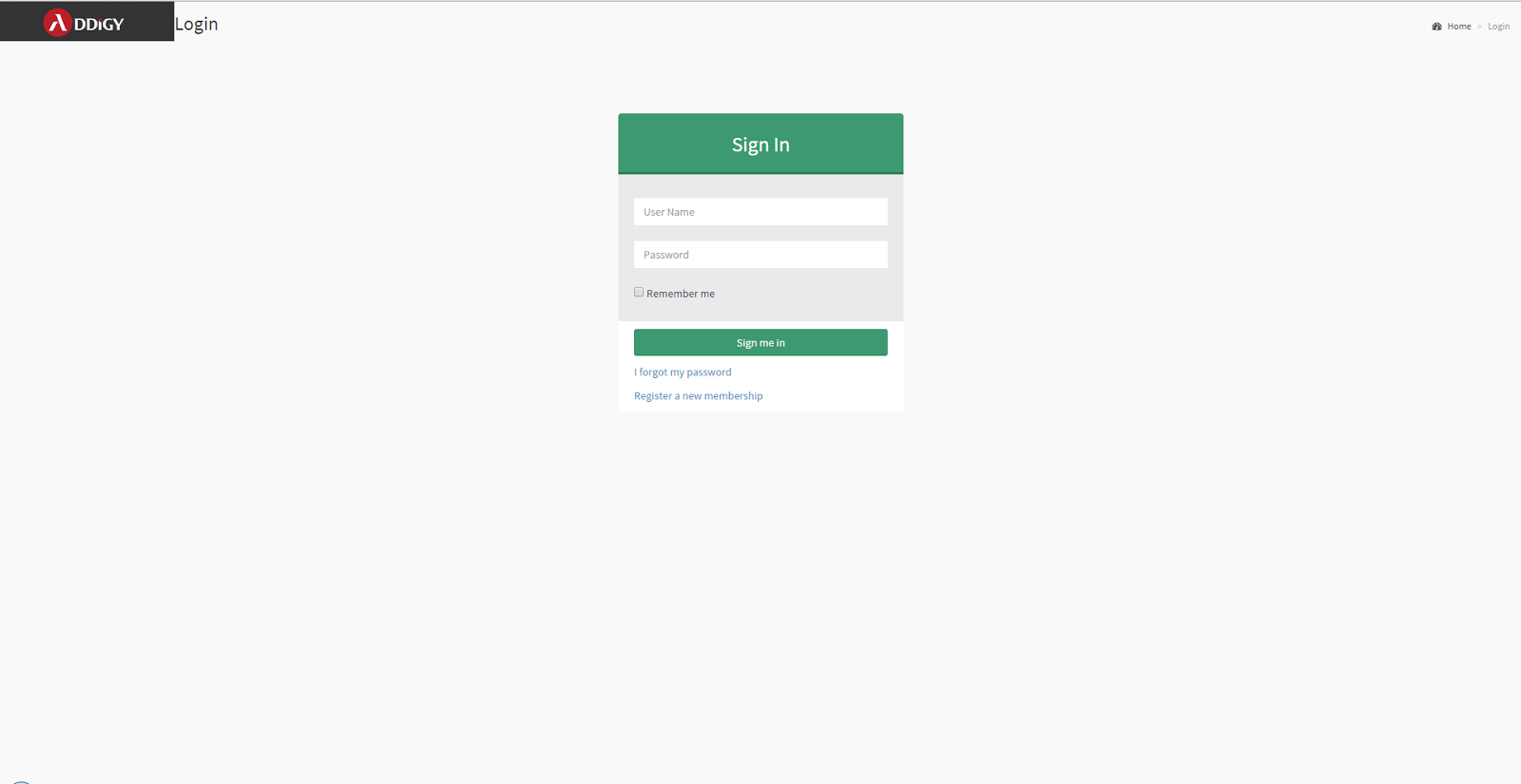
**See Active Ports in the Machine**

User looks at the active ports of the machine through the Netstats module

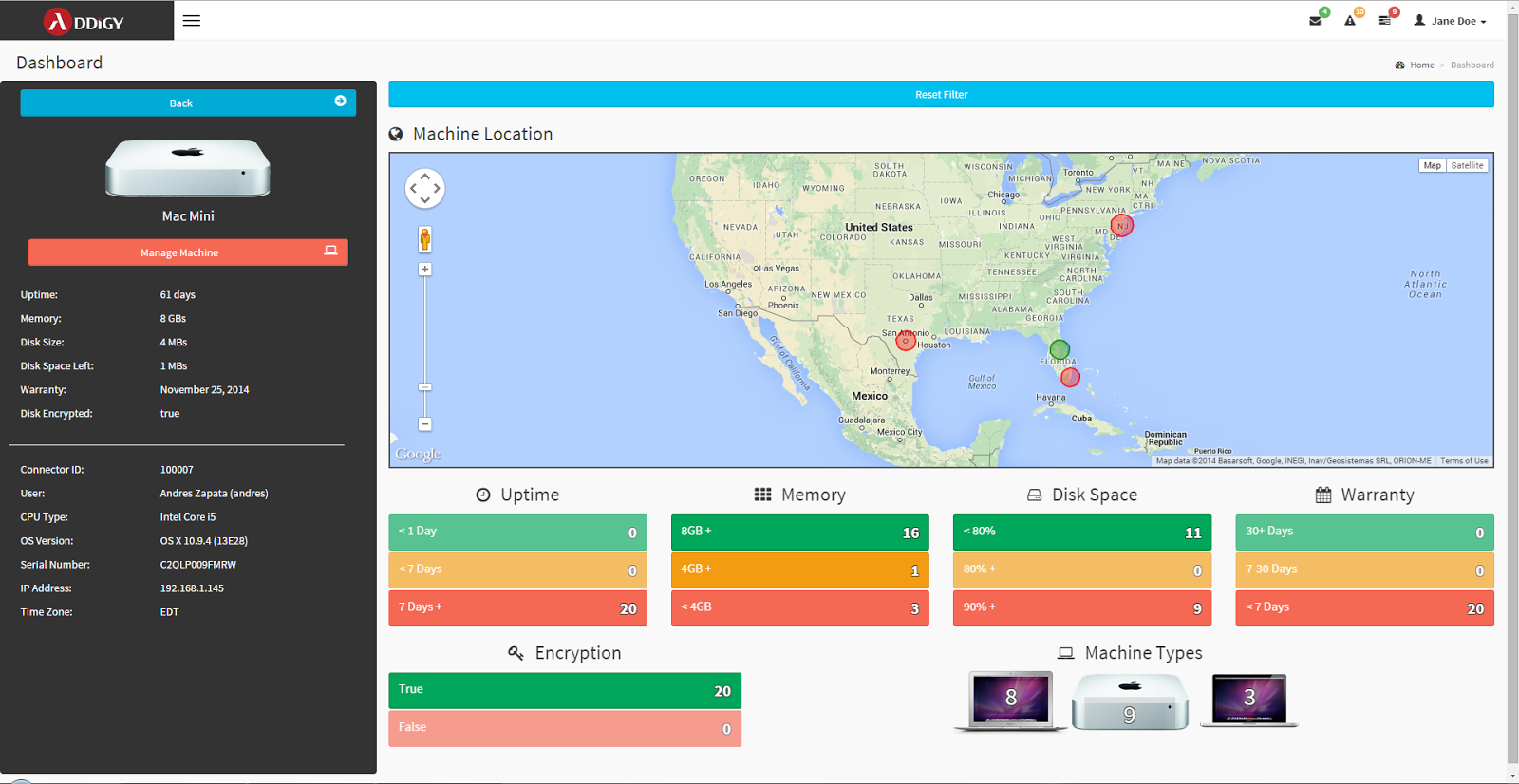


**6.5. Appendix E - User Interface designs.**

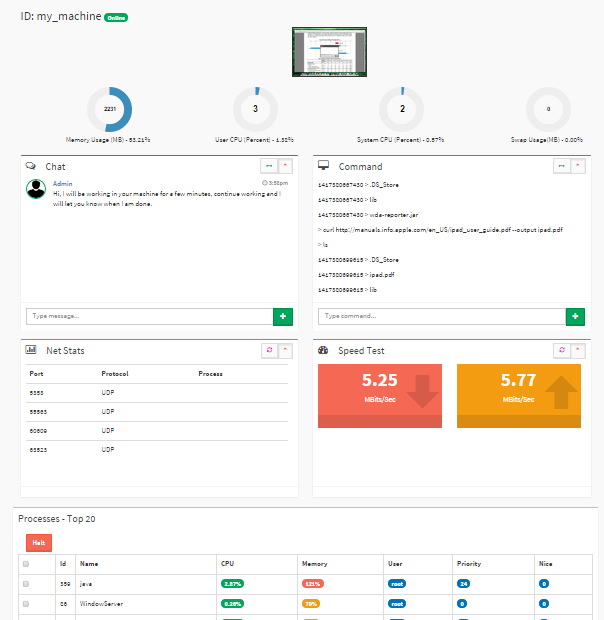
**Login**



**Multiple Machines’ Page**



**Single Machine Page**



## 

## 

## 

## 6.6. Appendix F - Diary of Meeting and Tasks.

**Diary of Meetings**

|  |  |
| --- | --- |
| **Meeting** | **Date** |
| Meeting 1 | 9/4/2014 |
| Meeting 2 | 9/9/2014 |
| Meeting 3 | 9/16/2014 |
| Meeting 4 | 9/23/2014 |
| Meeting 5 | 9/30/2014 |
| Meeting 6 | 10/7/2014 |
| Meeting 7 | 10/14/2014 |
| Meeting 8 | 10/21/2014 |
| Meeting 9 | 10/28/2014 |
| Meeting 10 | 11/4/2014 |
| Meeting 11 | 11/11/2014 |
| Meeting 12 | 11/18/2014 |
| Meeting 13 | 11/25/2014 |
| Meeting 14 | 12/2/2014 |
| Meeting 15 | 12/9/2014 |

**Project Schedule**

|  |  |
| --- | --- |
| **Task Name** | **Deadline** |
| Dev and Product servers are set up | 9/8/2014 |
| First Iteration: Scenario-Based Implementation - Unit Testing | 9/8/2014 |
| Second Iteration: Scenario-Based Implementation - Unit Testing | 9/22/2014 |
| Third Iteration: Scenario-Based Implementation - Unit Testing | 10/6/2014 |
| Fourth Iteration: Scenario-Based Implementation - System Testing | 10/20/2014 |
| Fifth Iteration: Scenario-Based Implementation - System Testing | 11/3/2014 |
| Sixth Iteration: Scenario-Based Implementation - System Testing | 11/17/2014 |
| Final deliverable | 12/11/2014 |
| Final posters | 12/5/2014 |
| Showcase | 12/12/2014 |