MOSS

Multiple Operating System Superintendent

Table of Contents

Executive Summary

for MOSS Project (version 1.0)

Computers with multiple operating systems installed, otherwise known as **multiboot systems**, have become more common over the last few years. Despite being advantageous in many ways, multiboot systems have a significant flaw.

The issue relates to navigating between the local operating systems. There is no elegant way to change the order in which your operatings systems boot or even which operating system to boot into once you restart the system. The only current solutions involve navigating the Unified Extensible Firmware Interface (UEFI) or executing commands on the terminal. Both of these methods are slow, and both are too complex to be easily utilized by the general public.

There is, however, a third option. The Multiple Operating System Superintendent, acronym MOSS, is a desktop application that provides a fast and efficient interface for managing the boot configuration. With MOSS, you can change boot order quickly and simply, saving time that would normally be spent on unnecessary boot screens and navigating the UEFI interface.

Our hope is that our effort in bettering the multi-boot experience will work to make multibooting more manageable for the general population. Advances in computer hardware have allowed the technologically proficient to implement multi-boot systems for years. We believe it is time that everyone is able to receive the benefits entailed in multibooting, not just the computer proficient.

Project Definition Document

for MOSS Project (version 2.0)

Purpose

Multiple operating system environments have become prevalent both in industry and in the home, having application on enterprise mainframes as well as on personal computers. One of the major deficiencies of these environments is boot speed, and consumers are often slowed by this schema. Multiple OS environments also suffer from over-complexity, being too difficult for the general audience. The Multiboot Operating System Superintendent (MOSS) project is designed to combat both deficiencies. The MOSS application provides an advanced UEFI management menu that both minimizes boot time and removes complexity from multiboot environments.

Goals and Objectives

Multiple operating system environments have become prevalent both in industry and in the home, having application on enterprise mainframes as well as on personal computers. One of the major deficiencies of these environments is boot speed, and consumers are often slowed by this schema. Multiple OS environments also suffer from over-complexity, being too difficult for the general audience. The Multiboot Operating System Superintendent (MOSS) project is designed to combat both deficiencies. The MOSS application provides an advanced UEFI management menu that both minimizes boot time and removes complexity from multiboot environments.

Project Context

Although there exists other UEFI managers (i.e. Visual BCD Editor) they are all lacking in some regard. MOSS is unique in that it not only provides a boot sequence editor, but it also provides a quick-boot panel for swiftly booting into another operating system or drive partition. MOSS is also user-friendly, with menus that are easy to navigate.

Expected Benefits

1. Decreased boot time and complexity on multiboot systems – Making multiboot systems more useful to consumers, both in industry and in the home.
2. Encourage users to use multiboot systems by increased efficiency and user-friendliness.
3. Potentially adding business value to the Linux Foundation and/or Microsoft Corporation

Scope

The MOSS project interacts with local operating systems and superintends UEFI procedure. As such, this project is highly dependent on the concurrency of operating system protocol, especially in any future OS updates. MOSS will take input from the user specifying boot preference and communicate that to the local operating system. MOSS will not supersede the operating system's protocol. Instead, MOSS utilizes existing operating system functionality to the user's benefit, by increasing efficiency and minimizing complexity. The MOSS project will also allow the user to fine-tune UEFI settings to better reflect their preferences.

Stakeholders

1. The Computer Science Department.
2. The Linux Foundation.
3. Microsoft Corporation.
4. Industries that use multiple operating systems on a day-to-day basis.
5. Individuals with personal computers.

Success Criteria

The MOSS project will have been a success if the following criteria are met according to the listed specifications.

- Compatible with multiple operating systems (Minimum: Windows 10 and Ubuntu 18.04).
- Decreases boot time on all compatible operating systems.
- Project meets all criteria by May 7th, 2020.

If the MOSS project does meet any one of the above criteria, it has failed.

Completed By:	**Jeremiah Gage**
Date:	**4/28/2020**
Project Title:	**MOSS**
Reviewed By:	**Cheri Kembell**

Technical Requirements Document

for MOSS Project (version 3.0)

1. Introduction

This document contains the system requirements for *MOSS*. These requirements have been derived from several sources, including comments/requests from dual-boot users.

1.1 Purpose of This Document

This document is intended to guide development of MOSS. It will go through several stages during the course of the project:

1. **Draft:** The first version, or draft version, is compiled after requirements have been discovered, recorded, classified, and prioritized.
2. **Proposed:** The draft document is then proposed as a potential requirements specification for the project. The proposed document should be reviewed by several parties, who may comment on any requirements and any priorities, either to agree, to disagree, or to identify missing requirements. Readers include end-users, developers, project managers, and any other stakeholders. The document may be amended and re-proposed several times before moving to the next stage.
3. **Validated:** Once the various stakeholders have agreed to the requirements in the document, it is considered validated.
4. **Approved:** The validated document is accepted by representatives of each party of stakeholders as an appropriate statement of requirements for the project. The developers then use the requirements document as a guide to implementation and to check the progress of the project as it develops.

1.2 How to Use This Document

We expect that this document will be used by people with different skill sets. This section explains which parts of this document should be reviewed by various types of readers.

Types of Reader

- **End Users:** Personal computer owners, ETC.
- **Enterprise Representatives**: Industry leaders, heads of development, ETC.

Technical Background Required

Some experience with operating systems is required. In particular, some familiarity with multi-boot systems is necessary for understanding the MOSS project and its applications.

Overview Sections

- 1.3 Scope of the Product
- 2. General Description
- 2.1 Product Perspective

1.3 Scope of the Product

The MOSS project interacts with local operating systems and superintends UEFI procedure. As such, this project is highly dependent on the concurrency of operating system protocol, especially in any future OS updates. MOSS will take input from the user specifying boot preference and communicate that to the local operating system. MOSS will not supersede the operating system's protocol. Instead, MOSS utilizes existing operating system functionality to the user's benefit, by increasing efficiency and minimizing complexity. The MOSS project will also allow the user to fine-tune UEFI settings to better reflect their preferences.

1.4 Business Case for the Product

Although there exists other UEFI managers (i.e. Visual BCD Editor) they are all lacking, either being too technical or inefficient to be suited to a general audience. MOSS is unique in that it not only provides a boot sequence editor, but it also provides a quick-boot panel for swiftly booting into another operating system or drive partition. MOSS is also user-friendly, with menus that are easy to navigate.

MOSS is both applicable and available to all users. Project stakeholders include:

1. The Computer Science Department.

2. The Linux Foundation.
3. Microsoft Corporation.
4. Industries that use multiple operating systems on a day-to-day basis.
5. Individuals with personal computers.

1.5 Overview of the Requirements Document

Benefits of the MOSS project include:

1. Decreased boot time and complexity on multiboot systems – Making multiboot systems more useful to consumers, both in industry and in the home.
2. Encourage users to use multiboot systems by increased efficiency and user-friendliness.
3. Potentially adding business value to the Linux Foundation and/or Microsoft Corporation.

2. General Description

This section will give the reader an overview of the project, including why it was conceived, what it will do when complete, and the types of people we expect will use it. We also list constraints that were faced during development and assumptions we made about how we would proceed.

Multiple operating system environments have become prevalent both in industry and in the home, having application on enterprise mainframes as well as on personal computers. One of the major deficiencies of these environments is boot speed, and consumers are often slowed by this schema. Multiple OS environments also suffer from over-complexity, being too difficult for the general audience. The Multiboot Operating System Superintendent (MOSS) project is designed to combat both deficiencies. The MOSS application provides an advanced UEFI management menu that both minimizes boot time and removes complexity from multiboot environments.

2.1 Product Perspective

MOSS is a GUI application that functions to increase the accessibility of multiboot environments by providing an enhanced management perspective for the Unified Extensible Firmware Interface (UEFI). MOSS is designed to ease users' workload by providing quick navigation between operating systems and creating boot behavior that is tailored to the user’s needs. The MOSS program will allow access to system functionality that is currently only available to users via the terminal or UEFI directly. This includes, but is not limited to, changing boot sequence, adding a temporary boot preference, and enabling/disabling secure boot. MOSS minimizes boot time by removing unnecessary steps in the boot process.

2.2 Product Functions

MOSS allows end-users to switch between a set number of operating systems located on the quick-launch panel. Local operating systems can be added to the quick-launch panel via the edit menu panel. From the edit menu, users can select an operating system from a list of populated boot options to add to the quick-launch panel. From this menu users may also set boot preferences, such as disabling secure boot or legacy mode.

2.3 User Characteristics

The MOSS software is designed for all computer users who either own a multiboot system or are interested in creating one. MOSS is also open to enterprise representatives who utilize multiple operating systems in the industry environment. MOSS comes with easy-installation features. However, end-users must maintain administrative access in order to install operating systems and manage boot settings on the local machine. Some knowledge of operating systems, mainly installation, is also required to use the MOSS system.

2.4 Assumptions and Dependencies

After the conclusion of the MOSS project, distribution of the finished product will occur over an internet medium. The exact website is still to be determined. Execution and installation can only be completed by the local machine administrator. This application is not supported for those without administrator access. See 3.1-3.3 for full breakdown of system/user requirements.

3. Specific Requirements

This section of the document lists specific requirements for MOSS. Requirements are divided into the following sections:

1. User requirements. These are requirements written from the point of view of end users, usually expressed in narrative form.
2. System requirements. These are detailed specifications describing the functions the system must be capable of doing.
3. Interface requirements. These are requirements about the user interface, which may be expressed as a list, as a narrative, or as images of screen mock-ups.

3.1 User Requirements

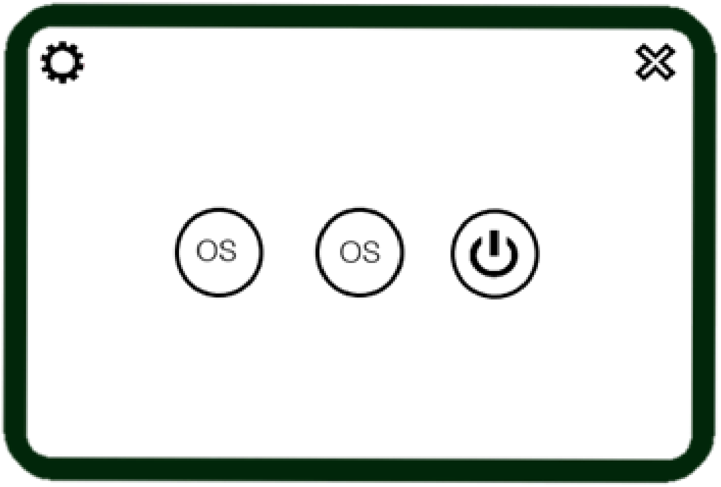
- Administrator access.
- OS installation familiarity.

3.2 System Requirements

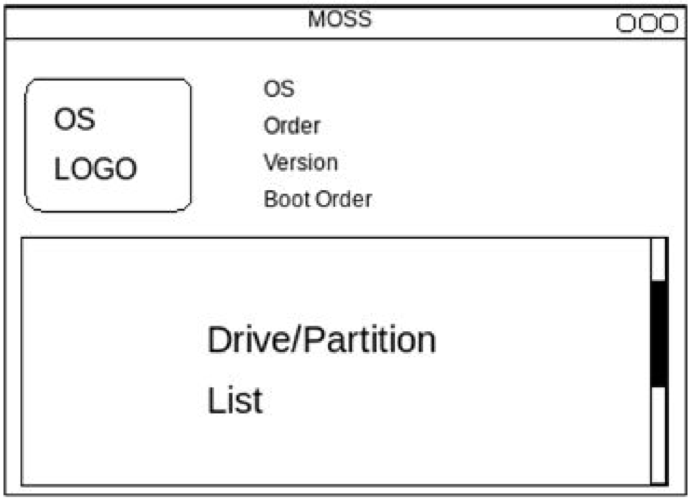
- Primary Operating System: Ubuntu 18.04 or newer.
- UEFI system with EFI variables enabled.
- Java JRE 11+

4. Mock-Ups

The GUI is comprised of two pages. The first page, the quick-start page, consists of a simple display with the MOSS system's base functionality. This page is designed to allow the user to quickly switch between operating systems.



The second page relates to the settings and preferences of the MOSS application. This page is equipped with a scrolling list of the system's drives and partitions, and fields illustrating the selected drive\partition's information.



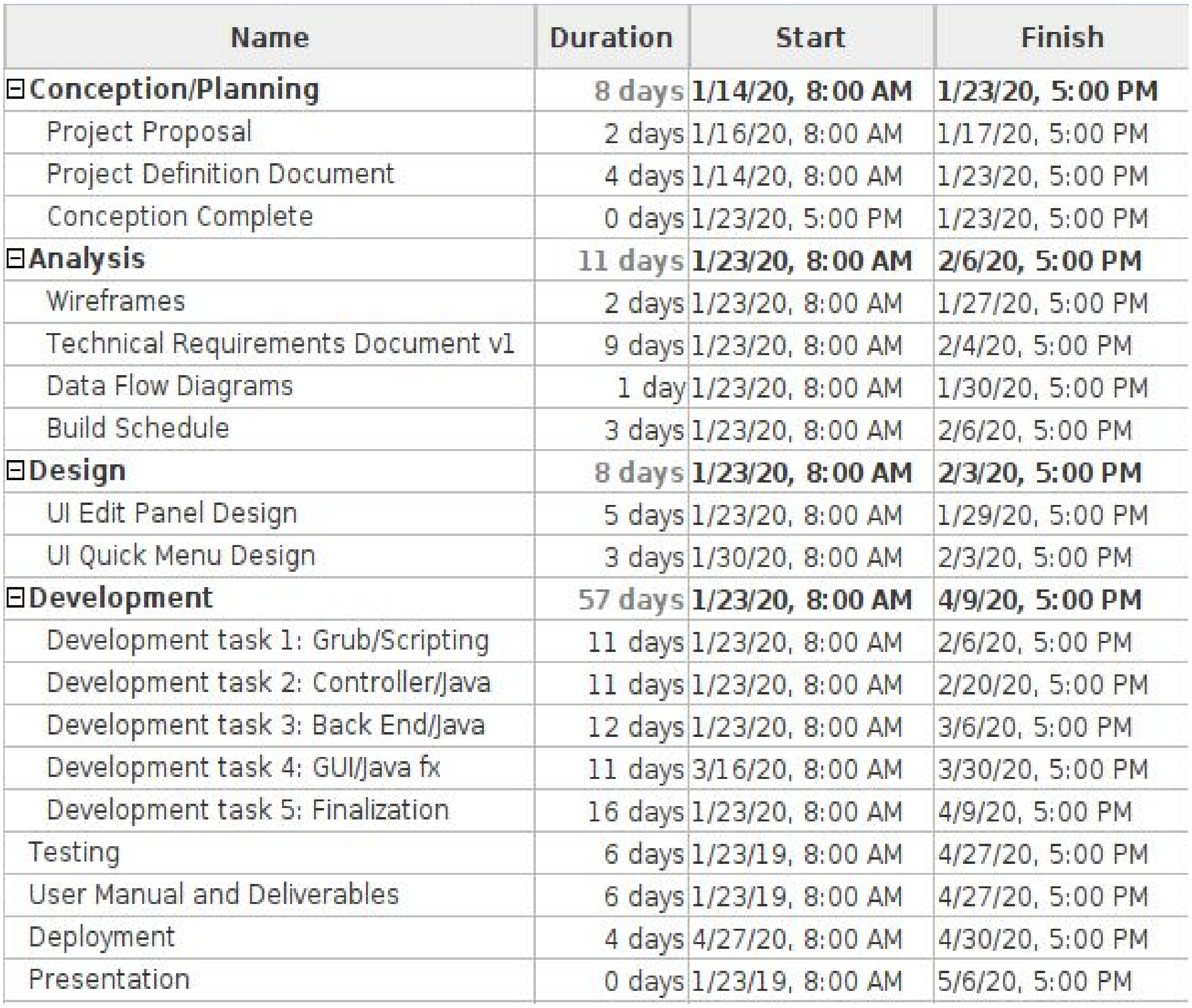
5. Glossary

- MOSS – Multiboot Operating System Superintendent.
- OS – Operating System.
- UEFI—Unified Extensible Firmware Interface.
- GRUB – GRand Unified Bootloader

Project Name:	**MOSS**
Date:	**4/28/2020**
Prepared By:	**Jeremiah Gage**
Document Status:	**Approved**

Project Schedule

for MOSS Project (version 2.0)



Name	Duration	Start	Finish
⊟**Conception/Planning**	**8 days**	**1/14/20, 8:00 AM**	**1/23/20, 5:00 PM**
Project Proposal	2 days	1/16/20, 8:00 AM	1/17/20, 5:00 PM
Project Definition Document	4 days	1/14/20, 8:00 AM	1/23/20, 5:00 PM
Conception Complete	0 days	1/23/20, 5:00 PM	1/23/20, 5:00 PM
⊟**Analysis**	**11 days**	**1/23/20, 8:00 AM**	**2/6/20, 5:00 PM**
Wireframes	2 days	1/23/20, 8:00 AM	1/27/20, 5:00 PM
Technical Requirements Document v1	9 days	1/23/20, 8:00 AM	2/4/20, 5:00 PM
Data Flow Diagrams	1 day	1/23/20, 8:00 AM	1/30/20, 5:00 PM
Build Schedule	3 days	1/23/20, 8:00 AM	2/6/20, 5:00 PM
⊟**Design**	**8 days**	**1/23/20, 8:00 AM**	**2/3/20, 5:00 PM**
UI Edit Panel Design	5 days	1/23/20, 8:00 AM	1/29/20, 5:00 PM
UI Quick Menu Design	3 days	1/30/20, 8:00 AM	2/3/20, 5:00 PM
⊟**Development**	**57 days**	**1/23/20, 8:00 AM**	**4/9/20, 5:00 PM**
Development task 1: Grub/Scripting	11 days	1/23/20, 8:00 AM	2/6/20, 5:00 PM
Development task 2: Controller/Java	11 days	1/23/20, 8:00 AM	2/20/20, 5:00 PM
Development task 3: Back End/Java	12 days	1/23/20, 8:00 AM	3/6/20, 5:00 PM
Development task 4: GUI/Java fx	11 days	3/16/20, 8:00 AM	3/30/20, 5:00 PM
Development task 5: Finalization	16 days	1/23/20, 8:00 AM	4/9/20, 5:00 PM
Testing	6 days	1/23/19, 8:00 AM	4/27/20, 5:00 PM
User Manual and Deliverables	6 days	1/23/19, 8:00 AM	4/27/20, 5:00 PM
Deployment	4 days	4/27/20, 8:00 AM	4/30/20, 5:00 PM
Presentation	0 days	1/23/19, 8:00 AM	5/6/20, 5:00 PM

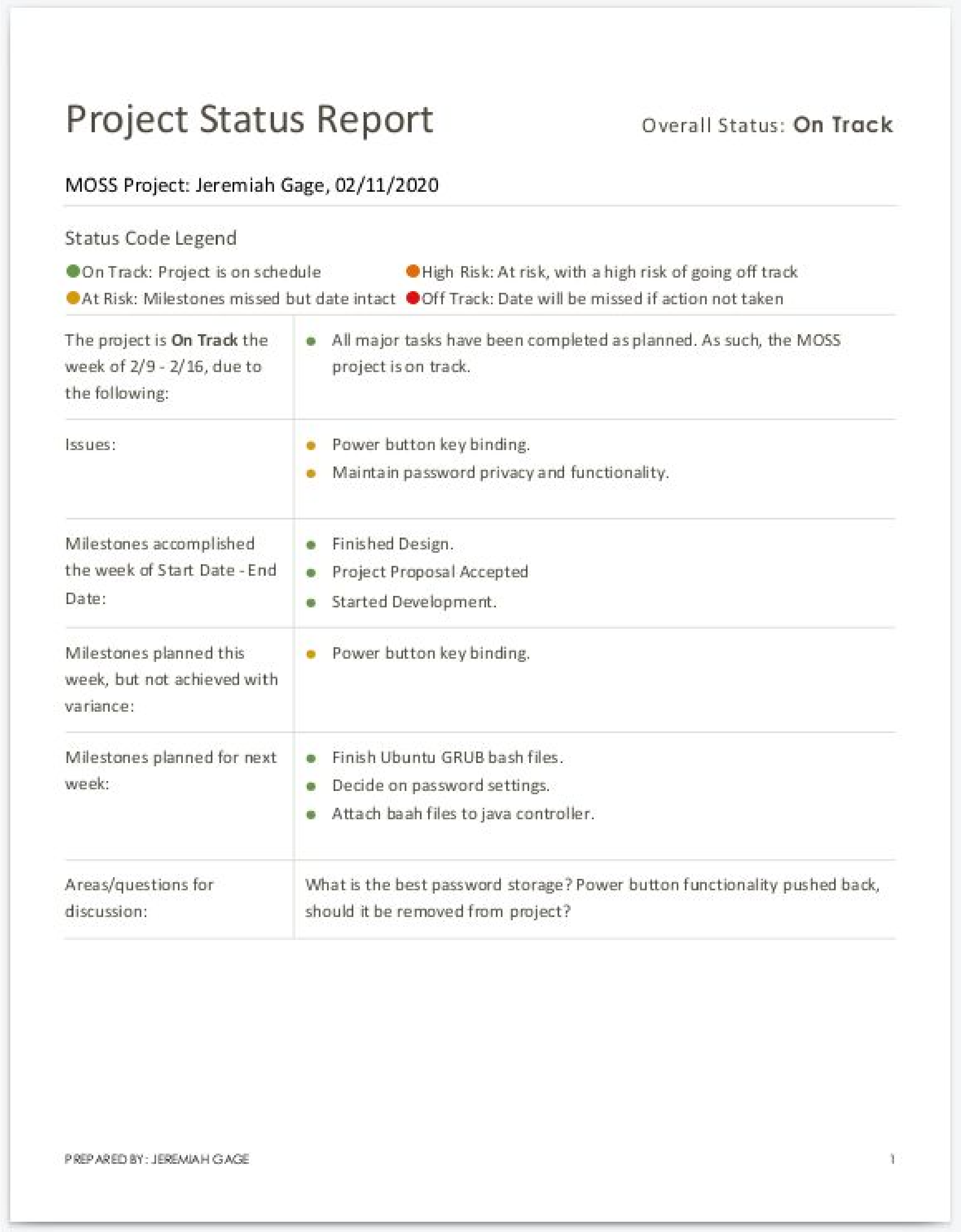
Status Reports

for MOSS Project

The following is a collection of status reports for the MOSS project. These reports were submitted intermittently, upon request, between the dates 1/23/2020 and 4/30/2020. These reports were used as progress checks of the MOSS software during its development. They were submitted by Jeremiah Gage in the duration of Senior Project in Computer Science (CSC 463-01) at College of the Ozarks. The reports were assessed by Cheri Kembell over the course of the Spring 2020 semester.

All documents were recorded here as they were originally submitted. Minimal formatting changes were made for inclusion here.

Status Report #1



Project Status Report

Overall Status: **On Track**

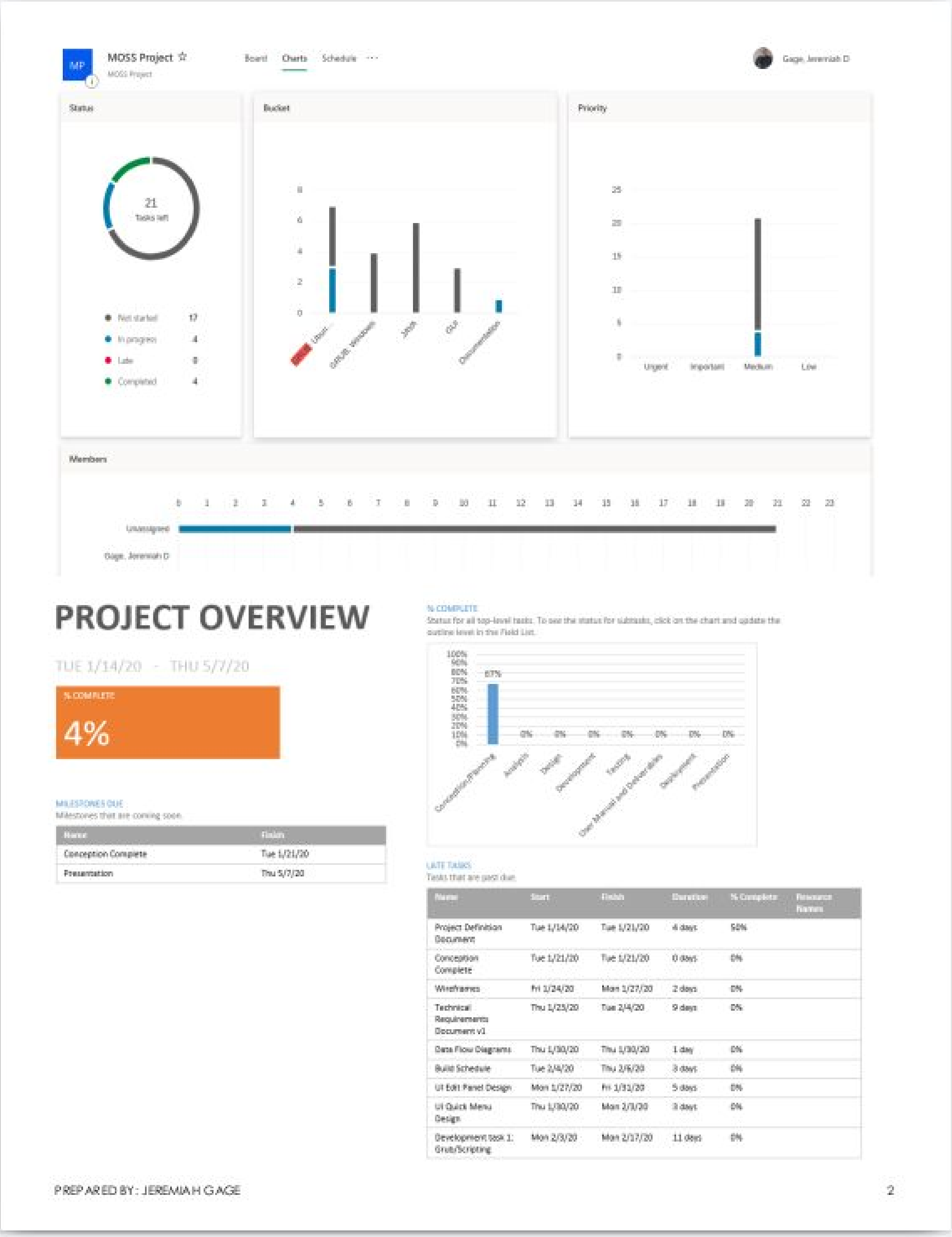
MOSS Project: Jeremiah Gage, 02/11/2020

Status Code Legend

🟢On Track: Project is on schedule 🟠High Risk: At risk, with a high risk of going off track
🟡At Risk: Milestones missed but date intact 🔴Off Track: Date will be missed if action not taken

The project is **On Track** the week of 2/9 - 2/16, due to the following:	🟢 All major tasks have been completed as planned. As such, the MOSS project is on track.
Issues:	🟡 Power button key binding. 🟡 Maintain password privacy and functionality.
Milestones accomplished the week of Start Date - End Date:	🟢 Finished Design. 🟢 Project Proposal Accepted 🟢 Started Development.
Milestones planned this week, but not achieved with variance:	🟡 Power button key binding.
Milestones planned for next week:	🟢 Finish Ubuntu GRUB bash files. 🟢 Decide on password settings. 🟢 Attach baah files to java controller.
Areas/questions for discussion:	What is the best password storage? Power button functionality pushed back, should it be removed from project?

PREPARED BY: JEREMIAH GAGE 1



PROJECT OVERVIEW

TUE 1/14/20 - THU 5/7/20

% COMPLETE

4%

% COMPLETE

Status for all top-level tasks. To see the status for subtasks, click on the chart and update the outline level in the Field List.

MILESTONES DUE

Milestones that are coming soon.

Name	Finish
Conception Complete	Tue 1/21/20
Presentation	Thu 5/7/20

LATE TASKS

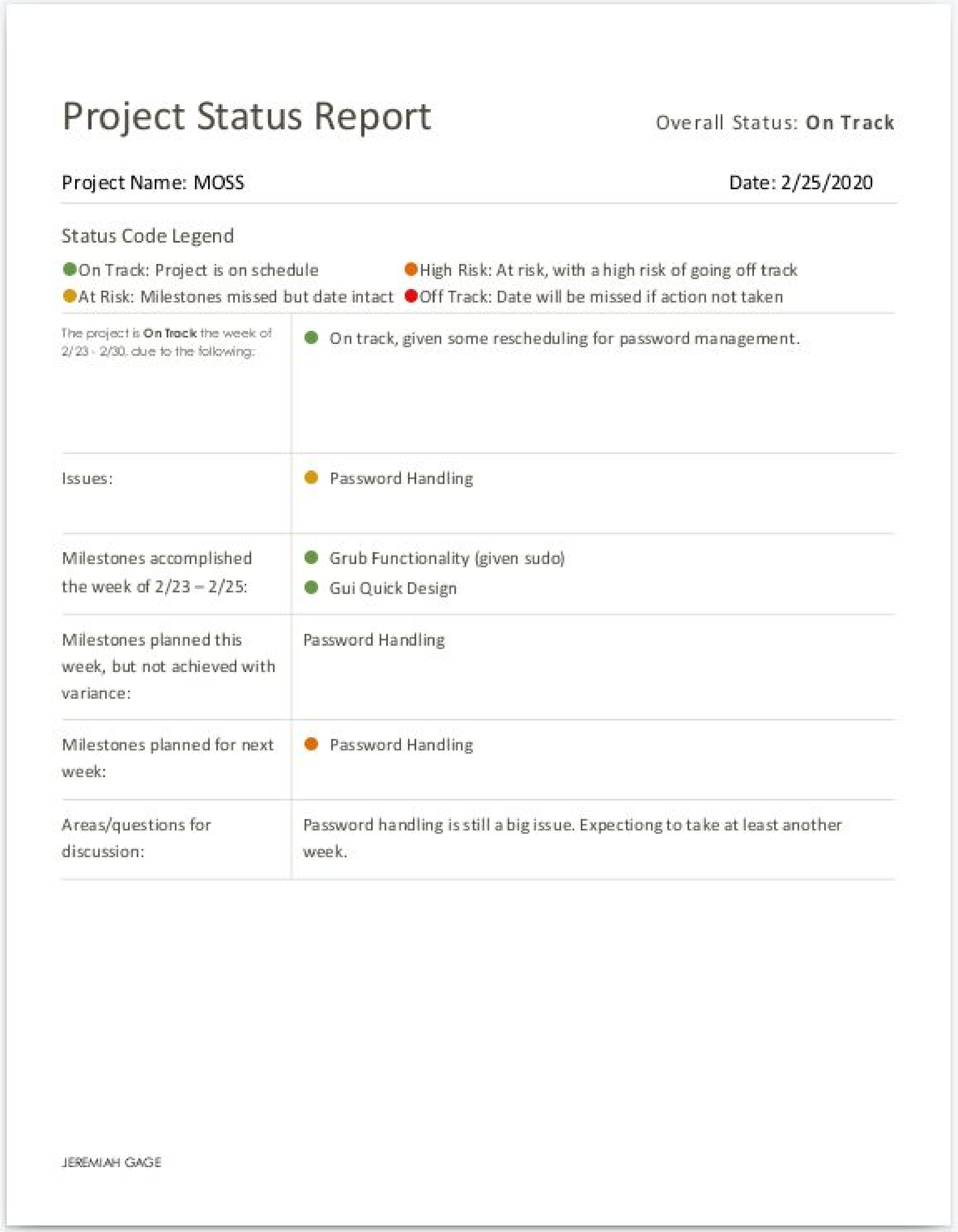
Tasks that are past due.

Name	Start	Finish	Duration	% Complete	Resource Names
Project Definition Document	Tue 1/14/20	Tue 1/21/20	4 days	50%	
Conception Complete	Tue 1/21/20	Tue 1/21/20	0 days	0%	
Wireframes	Fri 1/24/20	Mon 1/27/20	2 days	0%	
Technical Requirements Document v1	Thu 1/23/20	Tue 2/4/20	9 days	0%	
Data Flow Diagrams	Thu 1/30/20	Thu 1/30/20	1 day	0%	
Build Schedule	Tue 2/4/20	Thu 2/6/20	3 days	0%	
UI Edit Panel Design	Mon 1/27/20	Fri 1/31/20	5 days	0%	
UI Quick Menu Design	Thu 1/30/20	Mon 2/3/20	3 days	0%	
Development task 1: Grub/Scripting	Mon 2/3/20	Mon 2/17/20	11 days	0%	

PREPARED BY: JEREMIAH GAGE

2

Status Report #2



Project Status Report

Overall Status: **On Track**

Project Name: MOSS

Date: 2/25/2020

Status Code Legend

🟢On Track: Project is on schedule

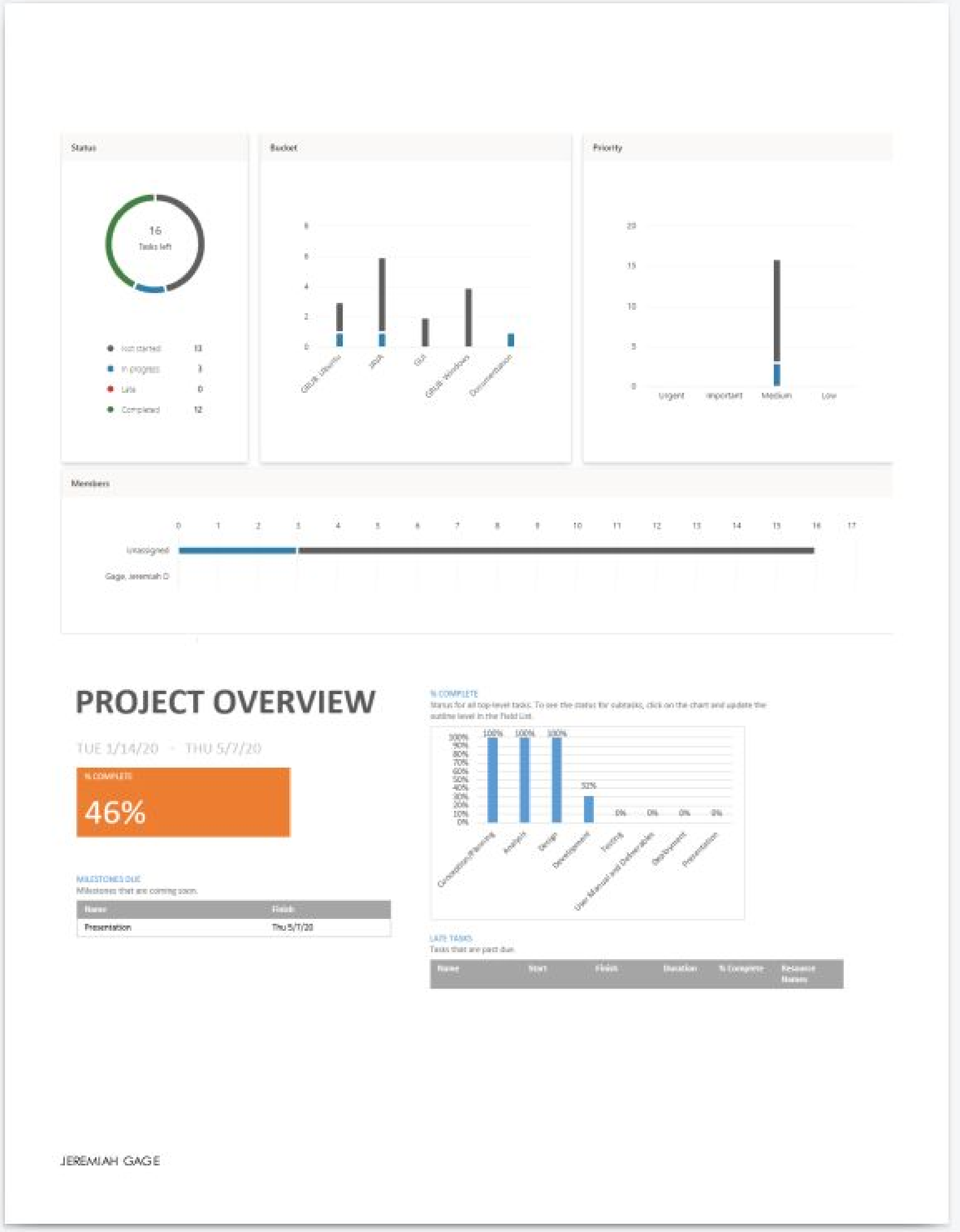
🟡At Risk: Milestones missed but date intact

🟠High Risk: At risk, with a high risk of going off track

🔴Off Track: Date will be missed if action not taken

The project is **On Track** the week of 2/23 - 2/30, due to the following:	🟢 On track, given some rescheduling for password management.
Issues:	🟡 Password Handling
Milestones accomplished the week of 2/23 – 2/25:	🟢 Grub Functionality (given sudo) 🟢 Gui Quick Design
Milestones planned this week, but not achieved with variance:	Password Handling
Milestones planned for next week:	🟠 Password Handling
Areas/questions for discussion:	Password handling is still a big issue. Expectiong to take at least another week.

JEREMIAH GAGE



PROJECT OVERVIEW

TUE 1/14/20 - THU 5/7/20

% COMPLETE

46%

MILESTONES DUE

Milestones that are coming soon.

Name	Finish
Presentation	Thu 5/7/20

% COMPLETE

Status for all top-level tasks. To see the status for subtasks, click on the chart and update the outline level in the Field List.

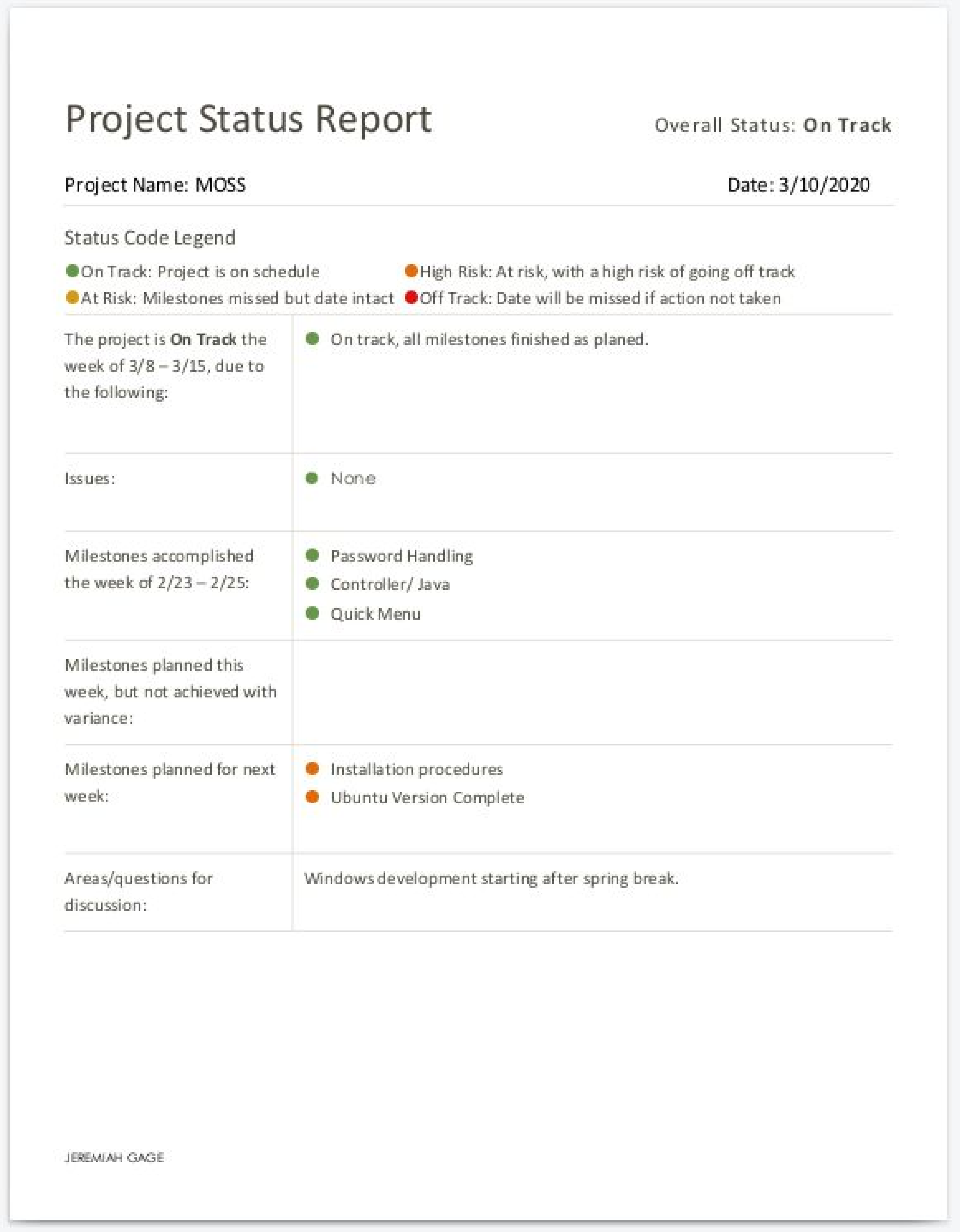
LATE TASKS

Tasks that are past due.

Name	Start	Finish	Duration	% Complete	Resource Names

JEREMIAH GAGE

Status Report #3



Project Status Report

Overall Status: **On Track**

Project Name: MOSS

Date: 3/10/2020

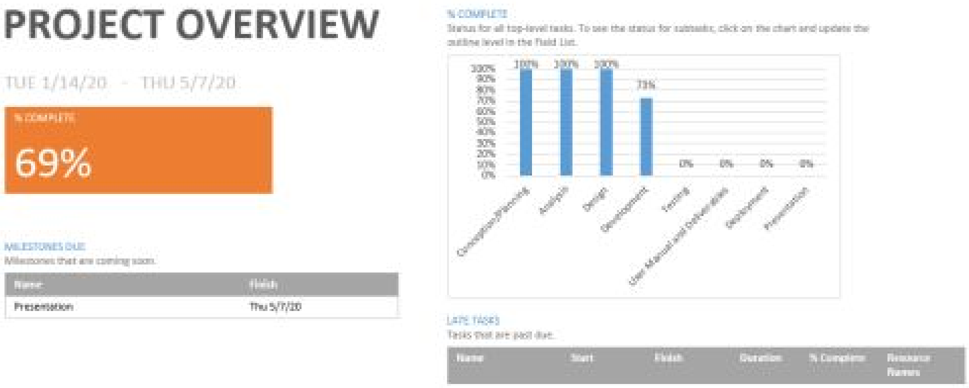
Status Code Legend

● On Track: Project is on schedule ● High Risk: At risk, with a high risk of going off track

● At Risk: Milestones missed but date intact ● Off Track: Date will be missed if action not taken

The project is **On Track** the week of 3/8 – 3/15, due to the following:	● On track, all milestones finished as planed.
Issues:	● None
Milestones accomplished the week of 2/23 – 2/25:	● Password Handling ● Controller/ Java ● Quick Menu
Milestones planned this week, but not achieved with variance:	
Milestones planned for next week:	● Installation procedures ● Ubuntu Version Complete
Areas/questions for discussion:	Windows development starting after spring break.

JEREMIAH GAGE



PROJECT OVERVIEW

TUE 1/14/20 - THU 5/7/20

% COMPLETE

69%

MILESTONES DUE

Milestones that are coming soon.

Name	Finish
Presentation	Thu 5/7/20

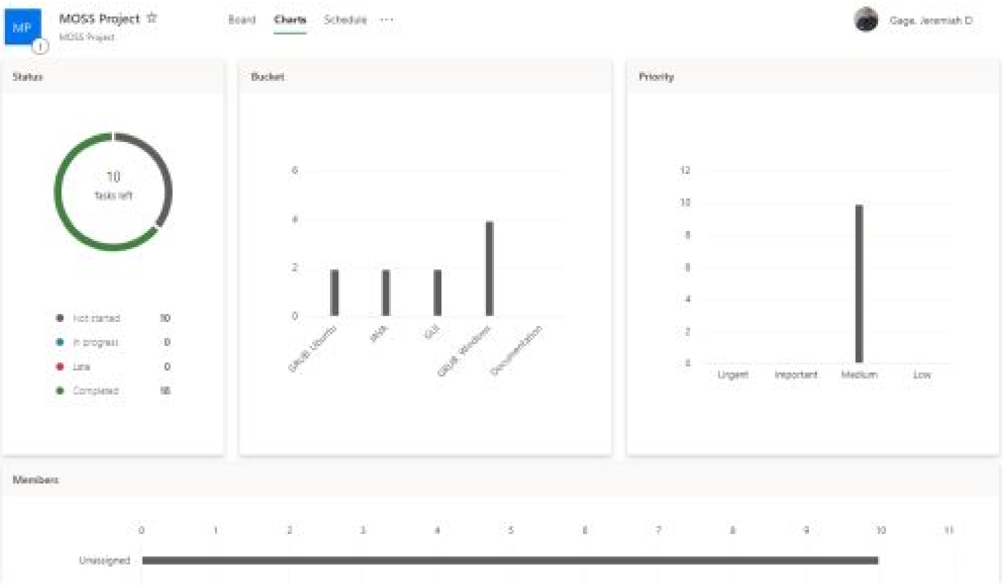
% COMPLETE

Status for all top-level tasks. To see the status for subtasks, click on the chart and update the outline level in the Field List.

LATE TASKS

Tasks that are past due.

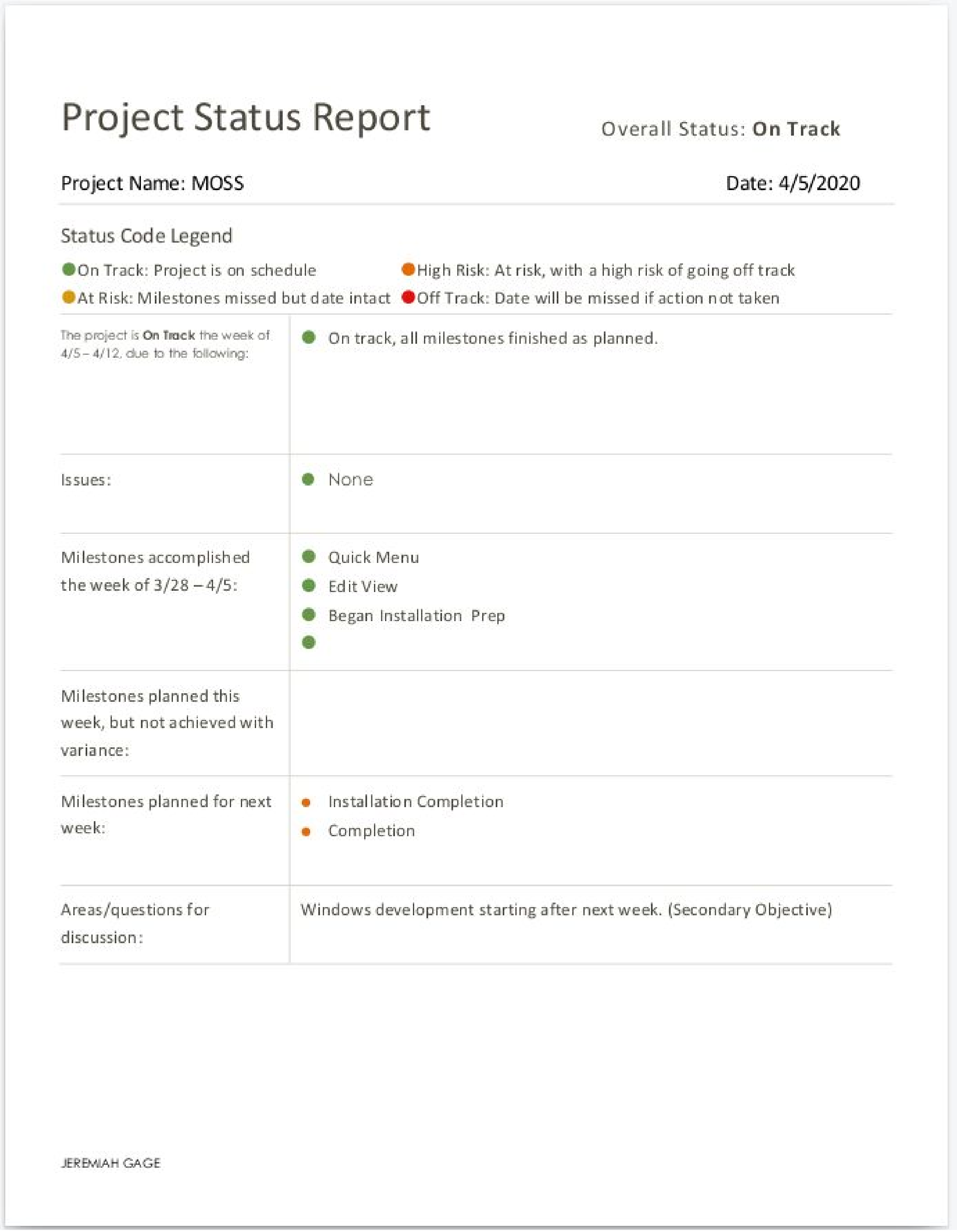
Name	Start	Finish	Duration	% Complete	Resource Names



●	Not started	10
●	In progress	0
●	Late	0
●	Completed	18

JEREMIAH GAGE

Status Report #4



Project Status Report

Overall Status: **On Track**

Project Name: MOSS

Date: 4/5/2020

Status Code Legend

● On Track: Project is on schedule

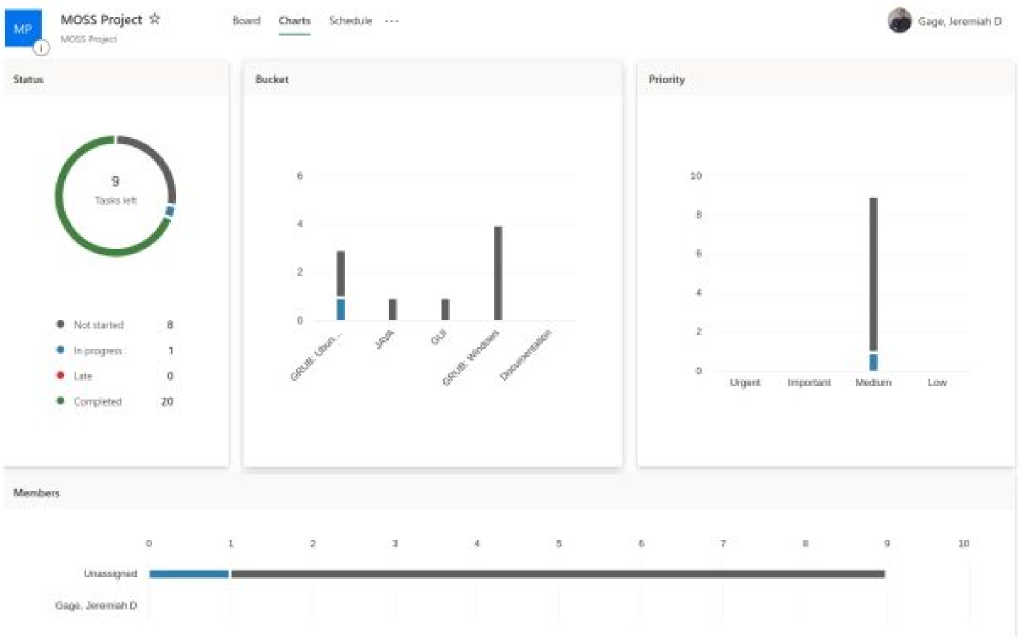
● High Risk: At risk, with a high risk of going off track

● At Risk: Milestones missed but date intact

● Off Track: Date will be missed if action not taken

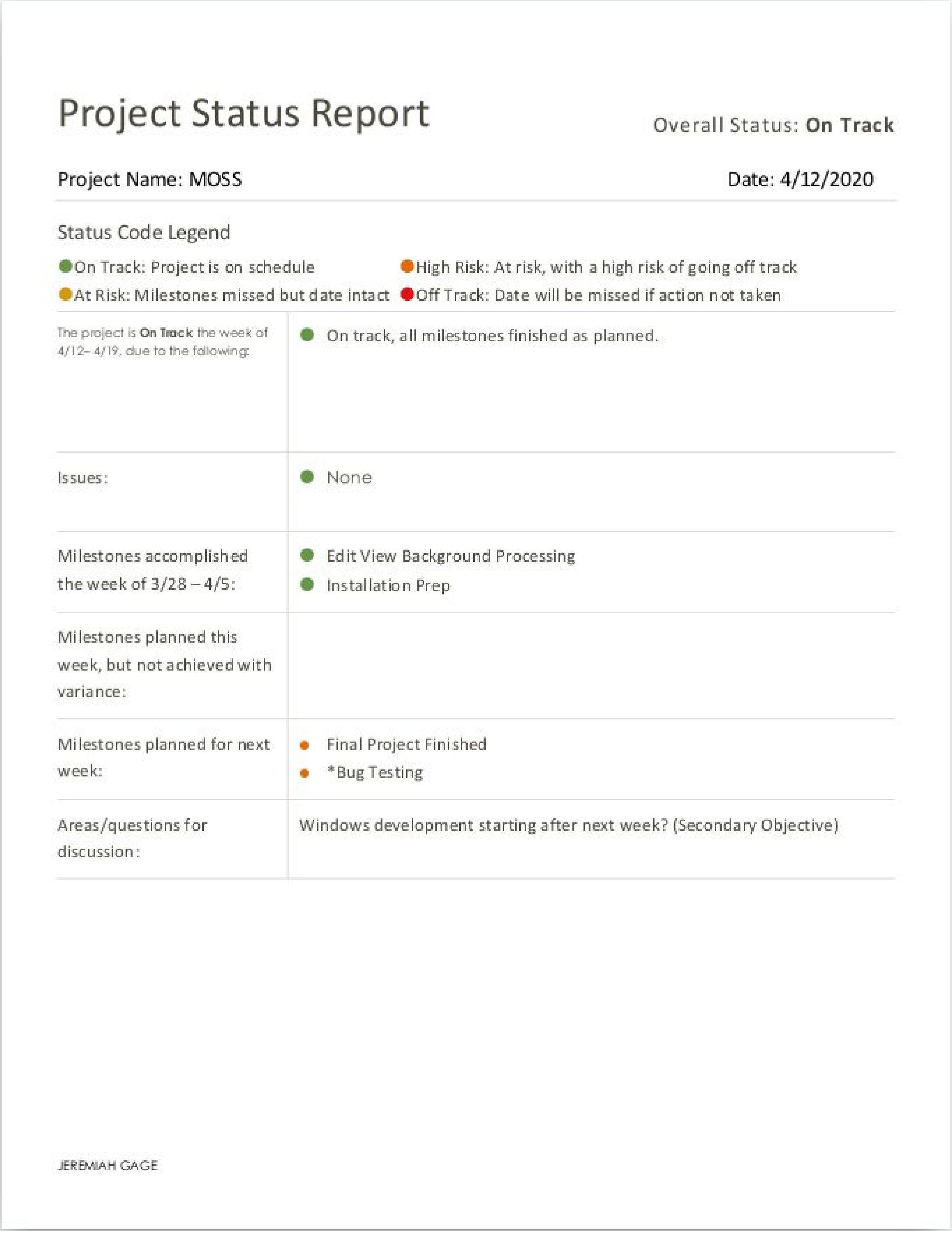
The project is **On Track** the week of 4/5 – 4/12, due to the following:	● On track, all milestones finished as planned.
Issues:	● None
Milestones accomplished the week of 3/28 – 4/5:	● Quick Menu ● Edit View ● Began Installation Prep ●
Milestones planned this week, but not achieved with variance:	
Milestones planned for next week:	● Installation Completion ● Completion
Areas/questions for discussion:	Windows development starting after next week. (Secondary Objective)

JEREMIAH GAGE



JEREMIAH GAGE

Status Report #5



Project Status Report

Overall Status: **On Track**

Project Name: MOSS

Date: 4/12/2020

Status Code Legend

🟢On Track: Project is on schedule

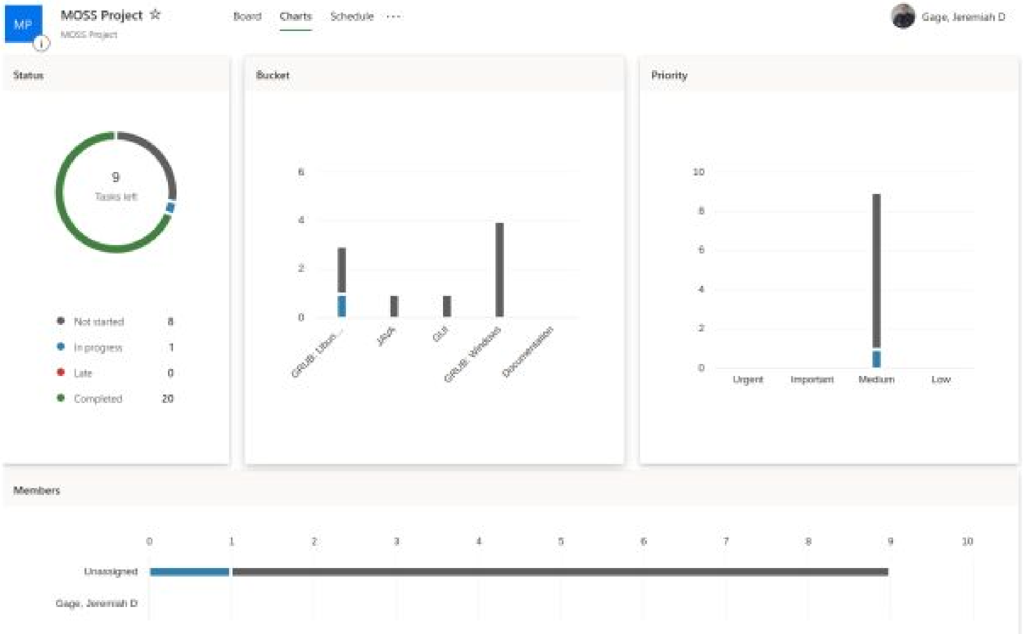
🟠High Risk: At risk, with a high risk of going off track

🟡At Risk: Milestones missed but date intact

🔴Off Track: Date will be missed if action not taken

The project is **On Track** the week of 4/12– 4/19, due to the following:	🟢 On track, all milestones finished as planned.
Issues:	🟢 None
Milestones accomplished the week of 3/28 – 4/5:	🟢 Edit View Background Processing 🟢 Installation Prep
Milestones planned this week, but not achieved with variance:	
Milestones planned for next week:	🟠 Final Project Finished 🟠 *Bug Testing
Areas/questions for discussion:	Windows development starting after next week? (Secondary Objective)

JEREMIAH GAGE



●	Not started	8
●	In progress	1
●	Late	0
●	Completed	20

User's Manual

for MOSS Project

User’s Manual: Table of Contents

Requirements

System Requirements

MOSS was designed for Ubuntu 19.10 or newer installations, but should work on similar Linux distros.

Program Dependencies

The MOSS system needs the Java Runtime Environment 11+ in order to run. If you do not have JRE 11+ or newer installed, you can install it with by opening a terminal window and entering the following text:

sudo apt install default-jre

And follow the resulting prompts.

Installation

Download the MOSS application and save the folder to a location where you can easily access it. Once saved, you can install MOSS with the following steps:

1. Open the folder and right-click in any open space.
2. Select “Open in terminal”
3. Type *./install.sh*

That’s it, MOSS is now installed! You can now run MOSS by clicking on it in the app launcher.

Note: You can uninstall MOSS anytime by repeating the steps but typing replacing *./install.sh* with *./uninstall.sh* on the third step.

Set Up

The first time you run MOSS, you may have to change a few things to make the application work effectively. If this is the first time starting the app, you should:

- Make sure Ubuntu is your Primary OS. To do that, follow the steps on **Change Boot Order.**
- Make sure your boot images are correct. See **Change Image.**

QuickMenu

The QuickMenu has five buttons. They are:

- **BootPrimary:** The first (left-to-right) of the three main buttons. It boots into the first OS on the boot order. It will not change the boot order next time you restart the PC.
- **BootSecondary:** The second (left-to-right) of the three main buttons. It boots into the second OS on the boot order. It will not change the boot order next time you restart the PC.
- **PowerOff:** The last (left-to-right) of the three main buttons. It powers off the system.
- **Exit:** Located at the top right, it closes the QuickMenu window.
- **Config:** Located at the top left of the QuickMenu pane, this button takes you to the BootMenu pane.

BootMenu

The BootMenu has three buttons. They are:

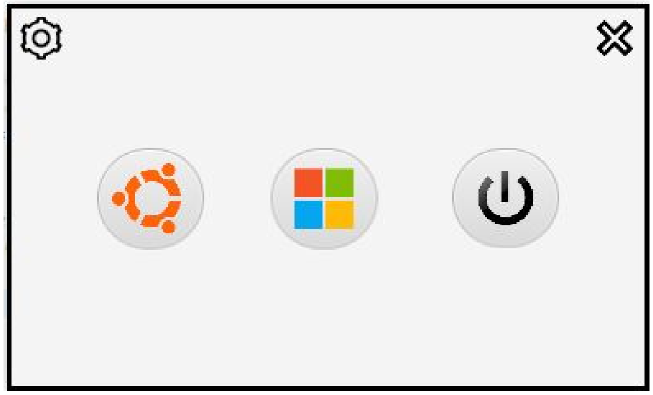
- **Load:** Loads all local boot options to the pane. You will only see boot options that were installed prior to your current boot, so you will need to restart your PC if you want to see any recent additions (i.e. bootable flash drive, etc.).

- **Boot:** Boots into whatever boot option is currently selected.
- **Set:** Saves the image path (primary and secondary only) and boot position of the selected boot.

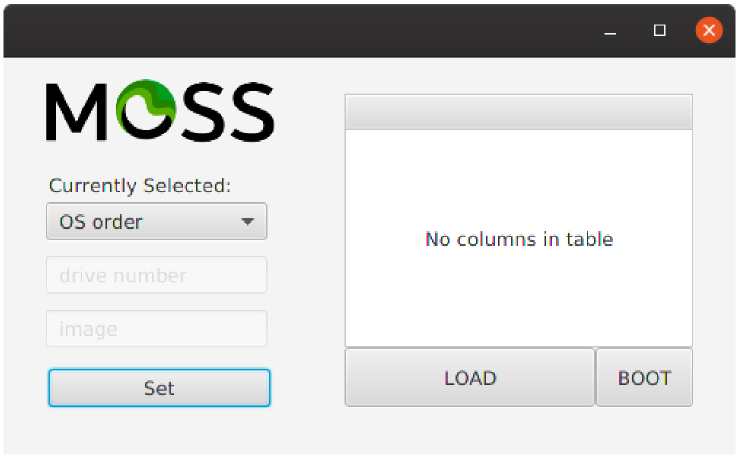
Change Image

For this example, we are going to change the image associated with our secondary boot: “Windows Boot Manager”. You can only change the images of the Primary and Secondary boots, as they are the only two that appear on the QuickMenu.

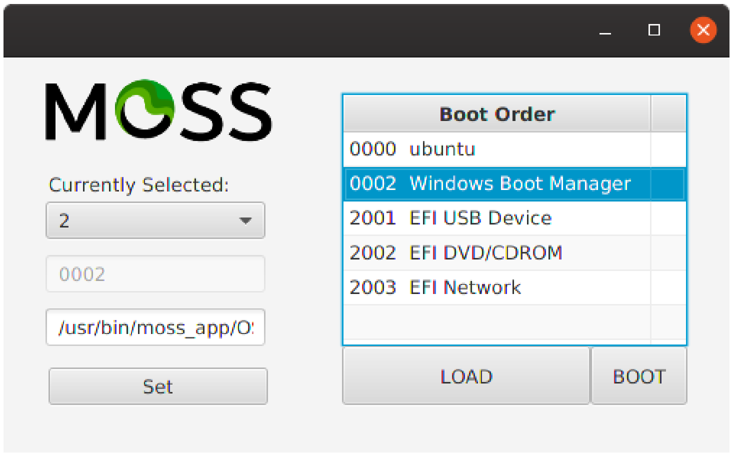
1. After opening the MOSS app, click the gear icon at the top left corner of the QuickMenu.



2. You should now see the BootMenu (as seen below).



3. Click “Load” to load all local boot options to the pane.
4. In our scenario, we have “Ubuntu”, “Windows Boot Manager”, “EFI USB Device”, “EFI DVD/CDROM”, and “EFI Network” as our boot options. “Windows Boot Manager” is in our secondary (#2) position, and we want to change the image associated with it.



Currently Selected: 2

0002

/usr/bin/moss_app/O:

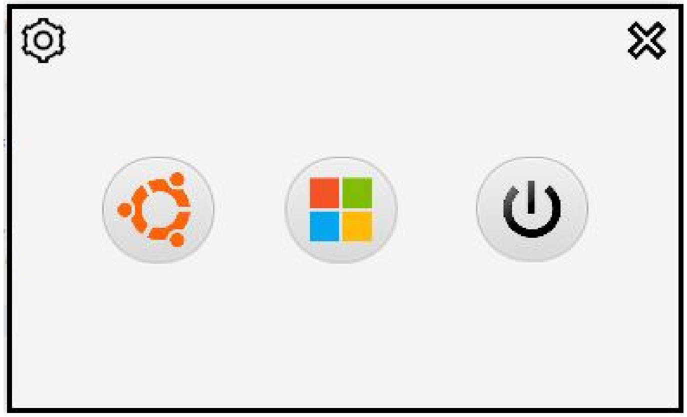
Boot Order
0000 ubuntu
0002 Windows Boot Manager
2001 EFI USB Device
2002 EFI DVD/CDROM
2003 EFI Network

5. Click on the “Windows Boot Manager” entry on the BootOrder pane.
6. While “Windows Boot Manager” is selected, navigate to the left side of the pane and paste the path to the new image in the text box. Then click “Set”.
7. Close all MOSS tabs and reopen the app. The new image should now be on the second button in the QuickMenu.

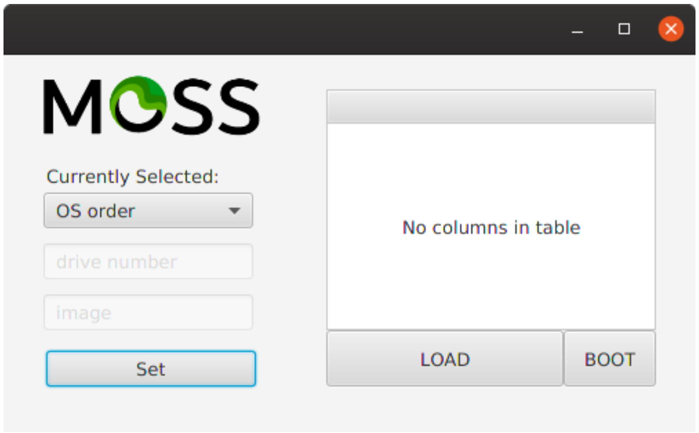
Change Boot Order

For this example, we are going to change “Ubuntu” to our primary boot, or boot #1, but you follow the same steps to change any “OS” to position #X.

1. After opening the MOSS app, click the gear icon at the top left corner of the QuickMenu (as seen below).



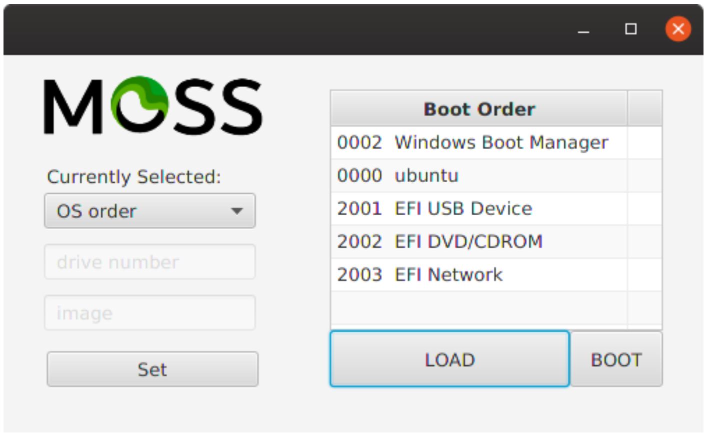
2. You should now see the BootMenu (as seen below).



Currently Selected: OS order

No columns in table

3. Click "Load" to load all local boot options to the pane.
4. In our scenario, we have "Windows Boot Manager", "Ubuntu", "EFI USB Device", "EFI DVD/CDROM", and "EFI Network" as our boot options.



Currently Selected: OS order

Boot Order
0002 Windows Boot Manager
0000 ubuntu
2001 EFI USB Device
2002 EFI DVD/CDROM
2003 EFI Network

5. Click on the “Ubuntu” entry on the BootOrder pane.
6. While “Ubuntu” is selected, navigate to the left side of the pane and select the dropdown box. Select #1 and press the “Set” button.
7. Hit “Load” again, “Ubuntu” should be at the top of the list as seen in the first photo. It is now in the Primary, or #1 position.

Project Name:	**MOSS**
Date:	**4/28/2020**
Prepared By:	**Jeremiah Gage**