

# Cascade Secure POS

Calvin Tallent

Owen Moore

Giovanni Munoz

Spring 2026

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Abstract . . . . .	3
1.2	Project Stakeholders . . . . .	3
<b>2</b>	<b>Team Members &amp; Bios</b>	<b>3</b>
2.1	Calvin Tallent . . . . .	3
2.2	Owen Moore . . . . .	3
2.3	Giovanni Munoz . . . . .	3
<b>3</b>	<b>Project Requirements Specification</b>	<b>3</b>
3.1	Project Stakeholders . . . . .	4
3.2	Use Cases . . . . .	4
3.3	Functional Requirements . . . . .	4
3.4	Non-Functional Requirements . . . . .	4
3.5	Standards . . . . .	4
<b>4</b>	<b>Software Design - Solution Approach</b>	<b>4</b>
4.1	Architecture Design . . . . .	4
4.1.1	Overview . . . . .	4
4.1.2	Subsystem Decomposition . . . . .	4
4.2	Data design . . . . .	5
4.3	User Interface Design . . . . .	5
4.4	Constraints . . . . .	5
<b>5</b>	<b>Test Case Specifications and Results</b>	<b>5</b>
5.1	Testing Overview . . . . .	5
5.2	Environment Requirements . . . . .	6
5.3	Test Results . . . . .	6
5.4	. . . . .	6
<b>6</b>	<b>Projects and Tools used</b>	<b>6</b>
<b>7</b>	<b>Description of Final Prototype</b>	<b>6</b>
<b>8</b>	<b>Social Responsibility and Broader Impacts</b>	<b>7</b>
<b>9</b>	<b>Product Delivery Status</b>	<b>7</b>
<b>10</b>	<b>Conclusions and Future Work</b>	<b>7</b>
10.1	Limitations and Recommendations . . . . .	7
10.2	Future Work . . . . .	8
<b>11</b>	<b>Acknowledgements</b>	<b>8</b>
<b>12</b>	<b>Glossary</b>	<b>8</b>
<b>13</b>	<b>References</b>	<b>8</b>
<b>14</b>	<b>Appendix A – Team Information</b>	<b>8</b>
<b>15</b>	<b>Appendix B - Example Testing Strategy Reporting</b>	<b>8</b>
<b>16</b>	<b>Appendix C - Project Management</b>	<b>8</b>

# **1 Introduction**

## **1.1 Abstract**

Many stores use point of sale (POS) systems to manage their inventory and sales. These systems help store owners track sales and manage their inventory. These are often touch-operated devices that also give the option of using a mouse pointer or keyboard shortcuts. They consist of purpose-built machines running proprietary, often custom-built, software.

Due to this proprietary nature, these systems are often expensive and are costly and time consuming to update. This incurs higher maintenance costs on businesses and a higher workload on employees due to the unreliability of outdated software and hardware.

## **1.2 Project Stakeholders**

Potential project stakeholders would include any physical retailer with the desire to use this software. The goal is to design the software such that it can be adapted to be used in any setting, offering a decent amount of customization. Although, the main target audience, in the beginning, will be traditional storefront retailers.

# **2 Team Members & Bios**

## **2.1 Calvin Tallent**

Calvin Tallent is a computer science student at Washington State University interested in a wide variety of topics, from embedded automotive systems to machine learning. He has experience in a wide variety of programming languages and tools, including Python, Javascript, Rust, Java, C/C++, Kotlin, C#, Git, and general Linux administration.

## **2.2 Owen Moore**

Owen Moore is a computer science student at Washington State University interested in machine learning, unique computer architectures and configurations, and the intersection between music and computers. He has experience programming in C/C++, Java, Python.

## **2.3 Giovanni Munoz**

Giovanni Munoz is a computer science student at Washington State University interested in software applications and machine learning. Also, systems to solve real world problems. He has experience in a variety of programming languages, including Python, Java, C/C++.

# **3 Project Requirements Specification**

Include the following:

## **3.1 Project Stakeholders**

List your stakeholders, summarize their needs.

### **3.2 Use Cases**

Describe the major use cases

### **3.3 Functional Requirements**

Include functional requirements

### **3.4 Non-Functional Requirements**

Include non-functional requirements

### **3.5 Standards**

Software engineering design is a process of devising a system, component, or process to meet desired needs and specifications within constraints. It is an iterative, creative, decision-making process in which the basic sciences, mathematics, and engineering sciences are applied to convert resources into solutions. Software engineering design involves identifying opportunities, developing requirements, performing analysis and synthesis, generating multiple solutions, evaluating solutions against requirements, considering risks, and making trade-offs, for the purpose of obtaining a high-quality solution under the given circumstances.

Standards vary by project. Examples of possible standards include W3C WCAG, GDPR, ECPA, FERPA, HIPAA, PCI DSS, IEEE 802, ACM SIGSOFT, IEEE 830, IEEE 1016, IEEE 829, and IEEE12207.

## **4 Software Design - Solution Approach**

This section should describe the final design of your software system.

Please review these sections and make the necessary updates to reveal the changes in your project design since your revision.

### **4.1 Architecture Design**

Include Section II from your Solution Approach report here. Provide the block diagram of your architecture and give a brief description of it.

#### **4.1.1 Overview**

This section should describe the overall architecture of your software. The architecture provides the top level design view of a system and provides a basis for more detailed design work.

#### **4.1.2 Subsystem Decomposition**

This section explains how you decomposed your system into subsystems.

### **4.2 Data design**

Describe all data structures (including the internal and temporary data structures), and the database(s) created as part of the application.

### **4.3 User Interface Design**

Provide a detailed description of user interface. The information in this section should be accompanied with proper images of your software's GUI.

The emphasis of this section should be on the design decisions of your GUI rather than the final design. You will explain your final GUI in detail in section VI (Description of Final Prototype). Please avoid redundancy as much as possible in these two sections.

### **4.4 Constraints**

Software engineering design is a process of devising a system, component, or process to meet desired needs and specifications within constraints. It is an iterative, creative, decision-making process in which the basic sciences, mathematics, and engineering sciences are applied to convert resources into solutions. Software engineering design involves identifying opportunities, developing requirements, performing analysis and synthesis, generating multiple solutions, evaluating solutions against requirements, considering risks, and making trade-offs, for the purpose of obtaining a high-quality solution under the given circumstances.

Constraints vary by project. Examples of possible constraints include aesthetics, codes, constructability, cost, ergonomics, extensibility, functionality, interoperability, legal considerations, maintainability, manufacturability, marketability, policy, regulations, schedule, sustainability, or usability.

## **5 Test Case Specifications and Results**

### **5.1 Testing Overview**

Include a summary of your test objectives and test plans.

Describe the overall approach to testing and provide the overall flow of the testing process. An example is provided in an Appendix.

Are you using Continuous Integration (CI) and/or Continuous Delivery (CD) in your testing? What tools are you using, what kinds of testing do they provide? Unit tests, integration, system, user interface, speed/non-functional requirement testing, user testing, etc.

In all of these major categories, include at least some material about your implementation, how it is currently executing, and/or what future work would be needed to ensure it happens for this project

- Unit testing
- Integration testing
- System testing
- Functional testing
- Performance testing
- Standards and constraints testing
- User acceptance testing

## 5.2 Environment Requirements

Specify both the necessary and desired properties of the test environment. The specification should contain the physical characteristics of the facilities, including the hardware, the communications and system software, the mode of usage (for example, stand-alone), and any other software or supplies needed to support the test. Identify special test tools needed.

## 5.3 Test Results

Include your prototype test results from your “Test Case Specifications and Results” document as updated with your current project status.

## 5.4

## 6 Projects and Tools used

Include a summary of the libraries, frameworks, and tools you used to implement your project. For example, you could have used some web frameworks, a database, a network message passing tool, graphics generation libraries, and various operating system platforms. Please list these out with a short one sentence note about what it was used to build/support in your project.

Tool/library/framework	Quick note on what it was for
Bootstrap	Generating layout and visual rendering on web interface
RabbitMQ	Intercommunication between our modules

As a quick survey, let me know what languages you wrote some of the project in. This is anything you wrote yourself, not just used in libraries. This includes both programming languages and markup languages.

Languages Used in Project			
C	C++	JavaScript	Erlang
Java	Python	PERL	Go
HTML5	LaTeX	BASH shell	

## 7 Description of Final Prototype

In this section provide a detailed description of your final prototype.

(If applicable) Include a brief user manual for your final software where you provide step by step instructions for using your system. Explain the major use case scenarios. You may include screenshots of the user interface.

Describe your final prototype implementation. Please format this section according to what you think is the best way to describe your prototype. The following is just a suggestion.

I recommend to include plenty of images and pictures of the following where appropriate:

- any diagrams/figures that visualize various features of your prototype;
- the screenshots of your user interfaces;
- the screenshots of your test programs;
- pictures of your team testing and debugging the devices, programs, etc.

A well-thought and clear diagram is better than long and descriptive text.

If your document starts to be very long due to screenshots and diagrams, please put at least some of them into an appendix to this document.

## 8 Social Responsibility and Broader Impacts

Social responsibility: identify informed judgements that you made for this project based on legal and ethical principles. Discuss if those judgements are in line with your responsibilities as a computing / software / cyber engineer. If not, what prompted you to make judgements that are contradictory to your professional responsibilities?

Broader impacts: highlight how your work aligns with the WSU EECS's goals of benefiting society and broadening participation in STEM. Describe not only how your project addressed a real-world problem (briefly state the problem) but also emphasized accessibility, inclusivity, and/or educational outreach. If you engaged with local communities through user testing and feedback sessions, mention that. If you collaborated with multiple teams from various departments or groups, mention that. If you documented your project and released it freely to support open-source missions, mention that. If your project outcomes contribute to technological innovation and provide a foundation for future student-led research, community-based applications, mention that.

## 9 Product Delivery Status

When was/will the project be delivered to your client? Was it demonstrated and who did you hand it off to?

Also include project location information here. This is for both the instructor and your client as future reference. This should include:

- 1) Source repositories
- 2) Equipment storage location - where did you leave any client or course materials
- 3) Any other materials needed to rebuild your project from the ground up

Ensure that either here or in your detailed description you include any instructions or where the instructions are to install and setup your project so future users know where to start.

## 10 Conclusions and Future Work

### 10.1 Limitations and Recommendations

Include a brief description of the limitations of your current prototype that you are aware of, and discuss possible solution approaches to overcome those limitations.

Please keep the discussion brief. 1 page text should be sufficient for this section.

Note that there might be overlap between the Limitations and Recommendations section and the Future Work section. “Future Work” should discuss the possible ways to extend the project, whereas “Limitations and Recommendations” should be more detailed and should discuss the problems and missing features in the final prototype and should describe the possible solution approaches.

## **10.2 Future Work**

Include a conclusion and discuss the future work. Future work should include discussion on how to extend this project. For those of you who are self or EECS sponsored, discuss commercialization possibilities.

## **11 Acknowledgements**

Thank the people who have contributed to your project. Also thank to your sponsors.

## **12 Glossary**

Define technical terms, acronyms, and anything else the reader might need to look up as they go through your document

## **13 References**

Cite all your references here. For the papers you cite give the authors, the title of the article, the journal name, journal volume number, date of publication and inclusive page numbers. Giving only the URL for the journal is not appropriate.

Use either Chicago, IEEE, or ACM style citations

-- Note: you can find many articles on scholar.google.com, which includes a link for each article's citation in various formats.

## **14 Appendix A – Team Information**

List your team members here and provide a team photo.

## **15 Appendix B - Example Testing Strategy Reporting**

- 1) Identify the requirements being tested
- 2) Either link to available online test results and/or take screenshots of the various system testing results
- 3) User testing can be demonstrated via survey results or quotes and a discussion of the feedback received

## **16 Appendix C - Project Management**

Describe your team's weekly schedule, i.e.,

- weekly meetings with the instructor/ mentor
- weekly meetings only with the team members
- other meetings and project related team activities.

Explain the purpose of each of the above activities and describe the routine agenda for each.  
Please comment on which team activities/meetings were the most beneficial to your team.  
Please include any planning documents you may have used. Examples could include:

- Gantt charts
- GitHub projects - Screenshots & a figure description
- GitHub issues - Screenshots & a figure description
- Notes on team tools used:
  - Email
  - SMS/IM
  - Slack
  - appear.in or other video conference tools
  - Trello
  - etc.