Denise Nguyen
Jason Sneddon
Jonathan Tapia
Nick Wallingford

CS 441 Final Project Report

ABSTRACT

Overview

We would like to develop a web application to manage equipment and inventory. This inventory management system would improve efficiency by providing the employees with a secure and user-friendly interface to interact with their equipment stored in a database.

Problem Statement

Many organizations manage their inventory through custom spreadsheets or text documents, which are error prone and time consuming. Commercial solutions to this problem exist, but they are expensive. Open source solutions also exist, but most of them lack essential features such as the ability to check out equipment. Therefore, we will create our own system to manage inventory.

Target Users

The target users for this inventory management system are organizations that need to manage an inventory of equipment. Depending on the success of development, the system will be implemented and used by the CSUSM Audio/Visual department.

Expected Outcome

The deliverable for this project will include:

- Database to store inventory data in an organized fashion.
- Login page to securely access the inventory management system.
- Intuitive pages to add, delete, and update their equipment efficiently.
- Pages that allow the user to check-in and check-out equipment.
- Database will be designed with optimal performance in mind.

AVtory Table of Contents

TEAM STRUCTURE

Team Members, Roles, and Contributions

NAMES & ROLES	CONTRIBUTIONS
 **Denise Nguyen** Product Owner Technical Writer Web Developer Overall Design Aesthetic Lead	- Created initial project plan on Project. - Ensured project milestones and deadlines were on met. - Technical documentation. - Created initial wireframes for user interface design. - Created sequence diagram and updated as necessary. - Implemented initial HTML/CSS for web pages. - Quality Assurance Tester after initial connection to database. - Helped development team where needed, answering questions, keeping everyone up to date as much as possible.
 **Jason Sneddon** Project Manager Technical Writer AWS Architect Database Designer Web Developer	- Suggested and refined initial project subject. - Created initial project plan. - Managed project plan and maintained sprint meeting logs. - Created web server in EC2 console - Created database on RDS instance - Created infrastructure to support web server and database in VPC - Designed, implemented, and maintained database - Quality Assurance Tester after initial connection to database
 **Jonathan Tapia** Technical Writer	- Assisted with project reports - Attended meetings
 **Nick Wallingford** Web Developer Back-end Developer	- Created initial UML diagrams - Created partial analysis models and diagrams - Contributed to database design - Established back-end server - Successfully connected backend to frontend via Python - Maintained Github repository - Implemented website functionalities - Quality Assurance Tester after initial connection to database

Team Contact Information

Name	Campus Email Address	Phone Number	Laptop Model
Denise Nguyen	nguye208@cougars.csusm.edu	(760) 498-9502	MacBook Pro
Jason Sneddon	snedd001@cougars.csusm.edu	(972) 838-7254	Dell Inspiron
Jonathan Tapia	tapia033@cougars.csusm.edu	(760) 712-9591	Microsoft Surface Pro 4
Nick Wallingford	walli020@cougars.csusm.edu	(858) 408-5722	Dell XPS

Team Norms

The following team norms are rules and guidelines that AVtory have developed. These represent the expectations that we agree to abide by for the duration of the project.

- Team members will be required to attend all scheduled meetings.
- All communication between team members, and the professor will be honest and respectful.
- All activities and responsibilities are clearly outlined and made known to each member contributing to the project. Upon decline of efficiency or problematic occurrences, the team will be responsible for addressing them as it pertains to their contribution.
- Each team member has been assigned specific tasks and critical roles for the development of the system.
- If a team member is struggling with their task, they will make it known to the other team members in a timely manner and well before any deadline. This will allow team members to receive any necessary assistance, while also ensuring all deadlines are met.
- Obligations tend to change week to week as different aspects of work required arise. People have different strengths and people try to align to those strengths, but flexibility was still kept important.

Product Owner

Our software development team worked in conjunction with the AV department on campus, who acted as beta testers for our system. This allowed us to get real-world feedback and develop a software system that meets the needs of customers that use the system on a regular basis.

Since Denise works at the AV department, she acted as the Product Owner and assisted with communication between the AV department and our software development team. Having one product owner simplified the project because we did not have overlapping customer requirements, which allowed us to focus our efforts on building a great system.

Project Manager

We chose to maintain a consistent project manager for the duration of the project because it allowed our team members to focus on the development and implementation of the project.

Quality Assurance & Tester

To align with our sprint goals, Denise, Nick, and Jason acted as quality assurance testers throughout the project. This allowed us to test each functionality as it was being developed, which made development more efficient.

Technical Writer

The thorough documentation of the software development life cycle was maintained by the team members who contributed to product development.

PROJECT PROPOSAL

User Requirements

At minimum, we require the following capabilities:

- Secure login
- Add, remove, and track equipment
- Process to check-in and check-out equipment
- Modern, user-friendly interface

Project Objectives

Our ***top-level objective*** is to design and develop a database and web application, using a cloud-based architecture, to store and visualize inventory.

The ***primary objectives*** for the design and development of the inventory management system are as follows:

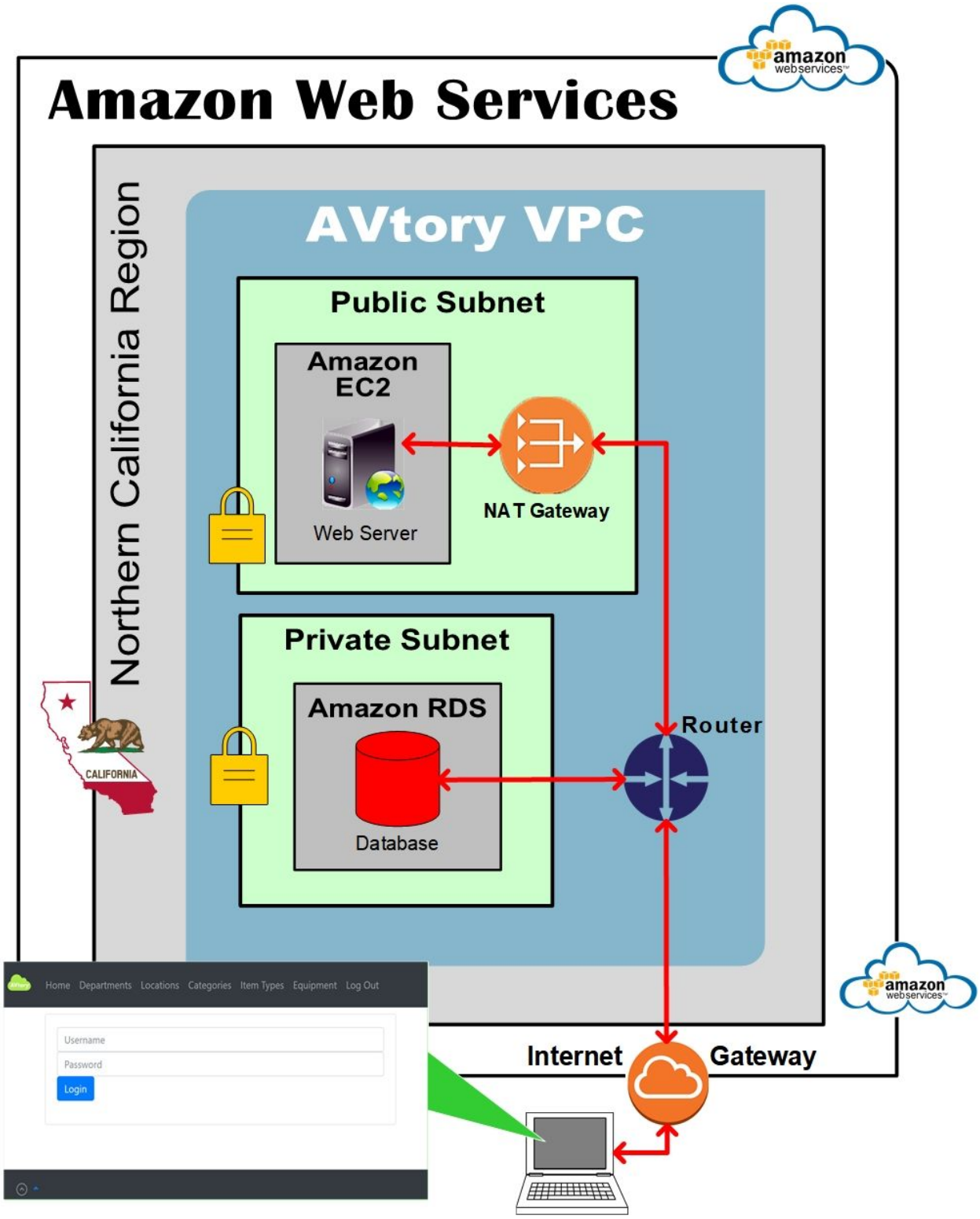
1. Design a website that has the ability to efficiently:
    a. Add and delete equipment and other materials
    b. Check-in and Checkout equipment
2. Create a modern website that employees can easily use to login and manage equipment.
3. Develop a scalable database that will align with future growth predictions of businesses or other organizations.

The ***secondary objectives*** may be implemented if time permits:

1. Remote access - off campus access to the system
2. RFID or barcode scanning - to improve the check-in and checkout process of equipment

SYSTEM DESIGN

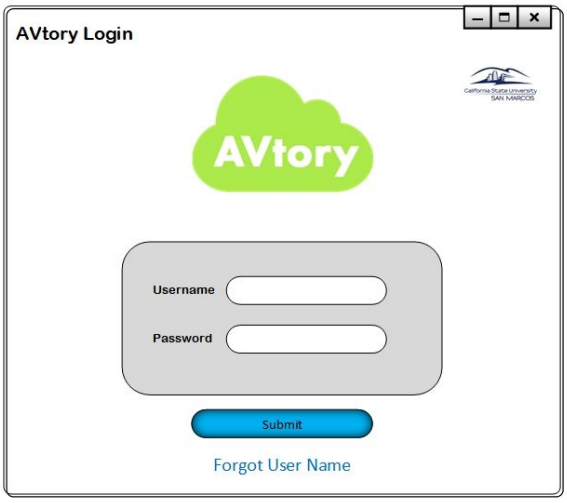
System Architecture



Link: AVtory

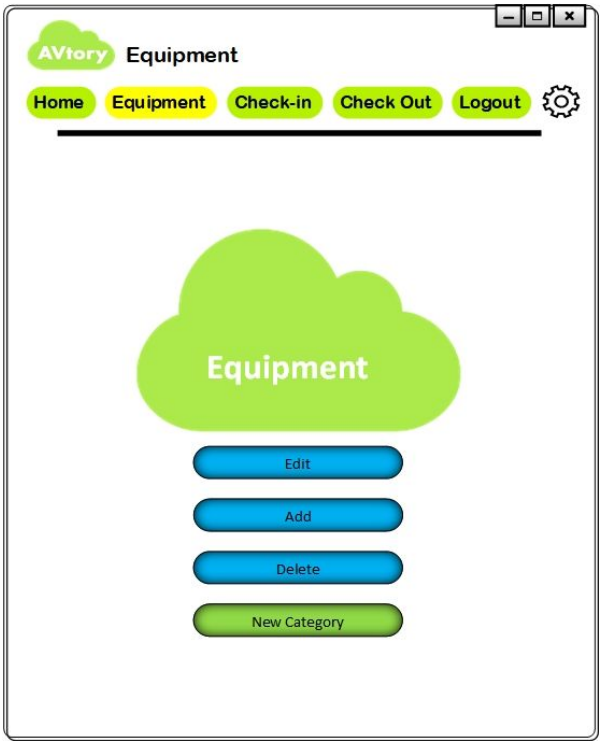
USER INTERFACE DESIGN

Wireframe - Login Page

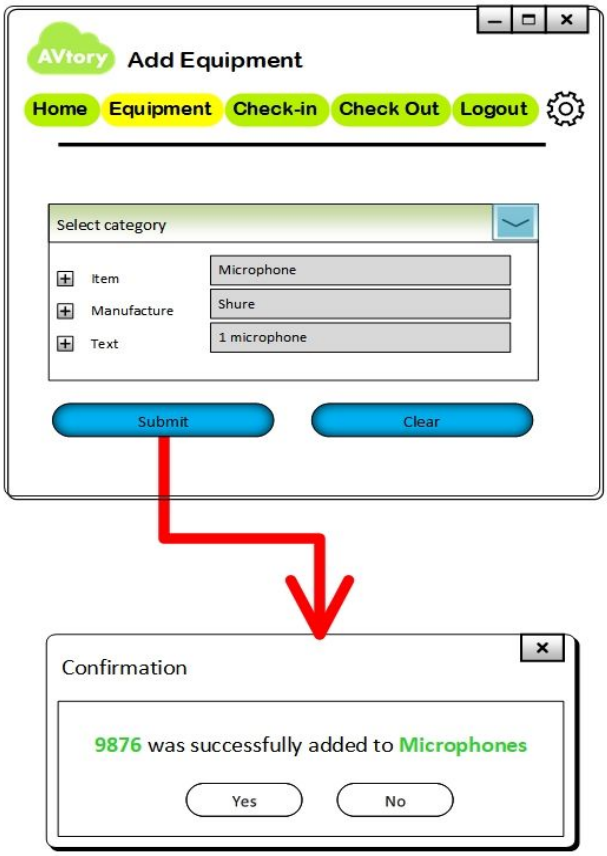


Wireframe - Home Page

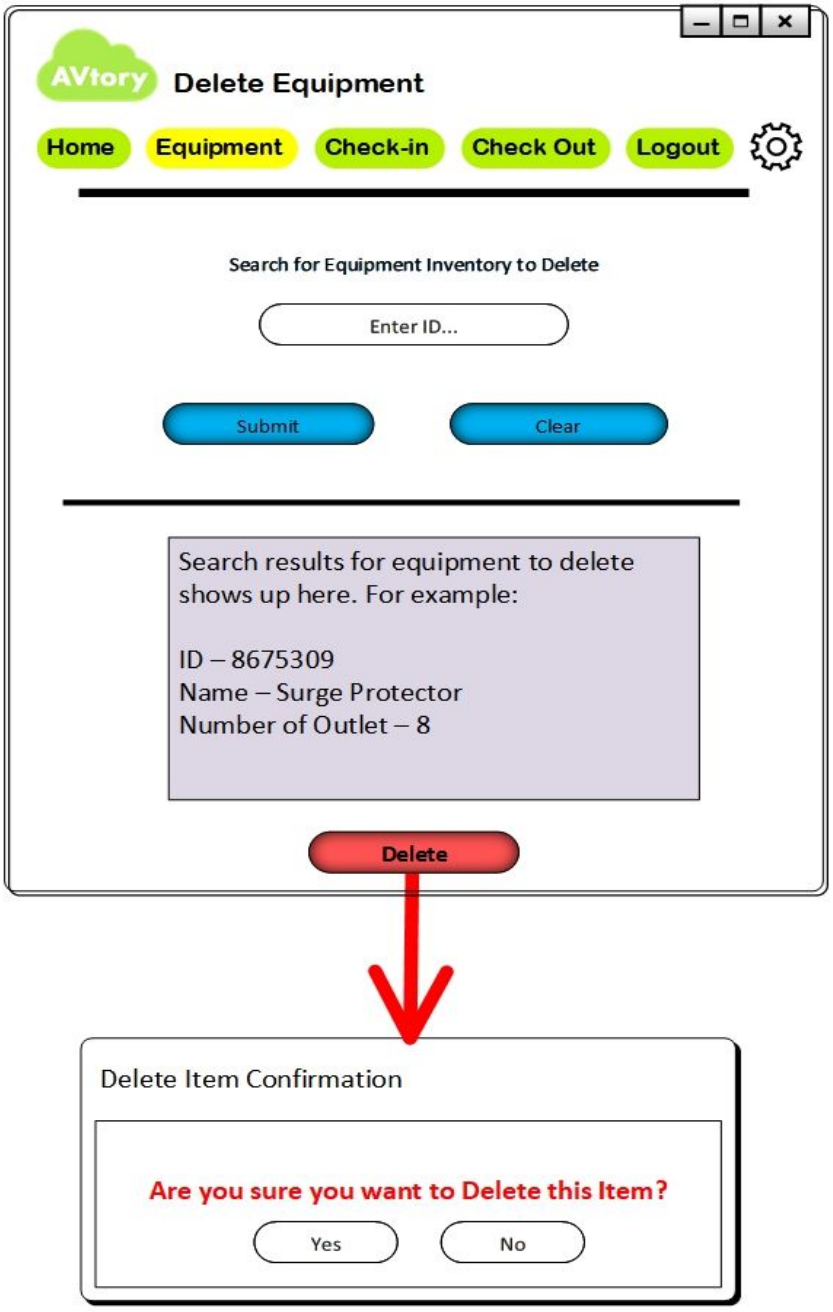
Wireframe - Equipment Page



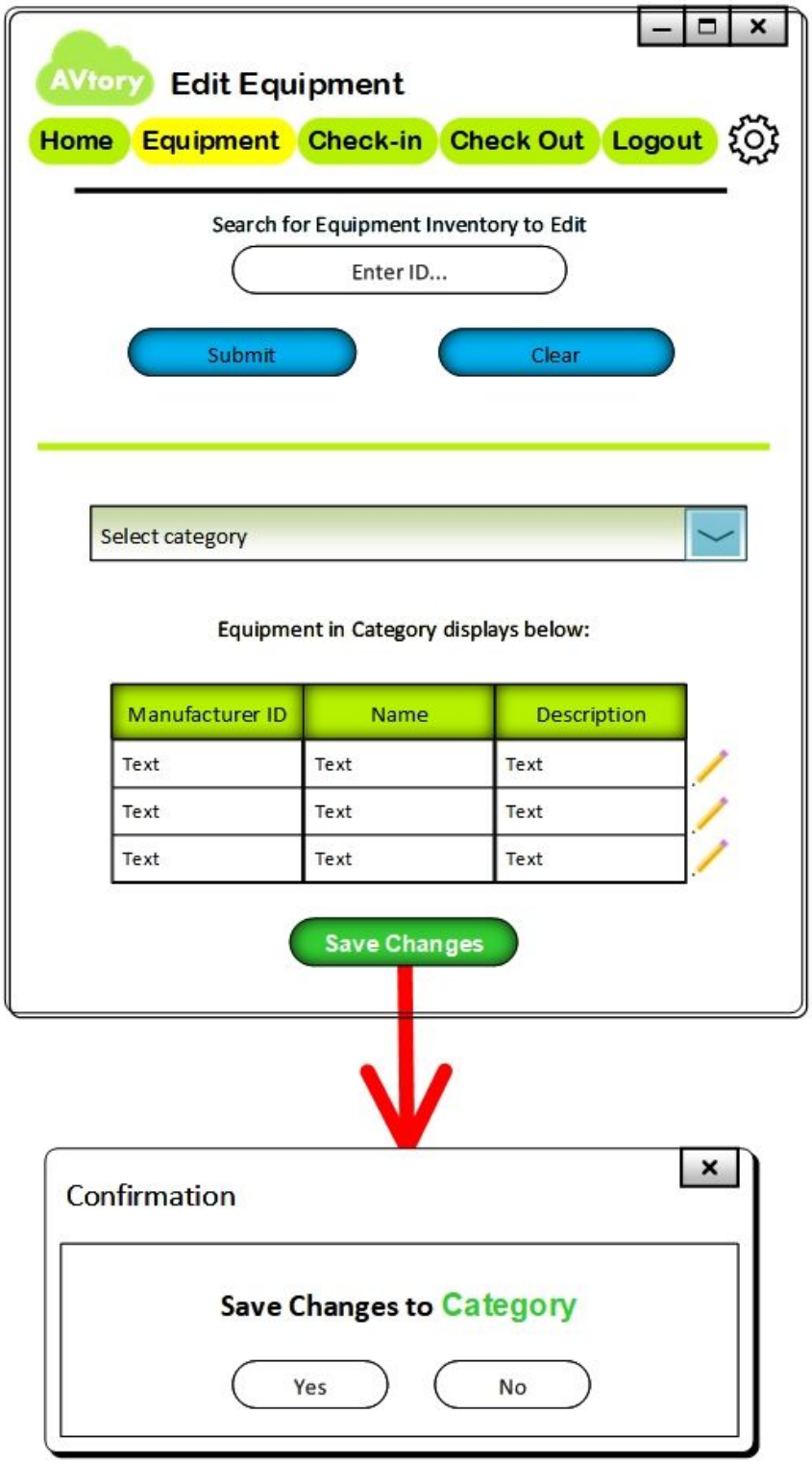
Wireframe - Add Equipment Page



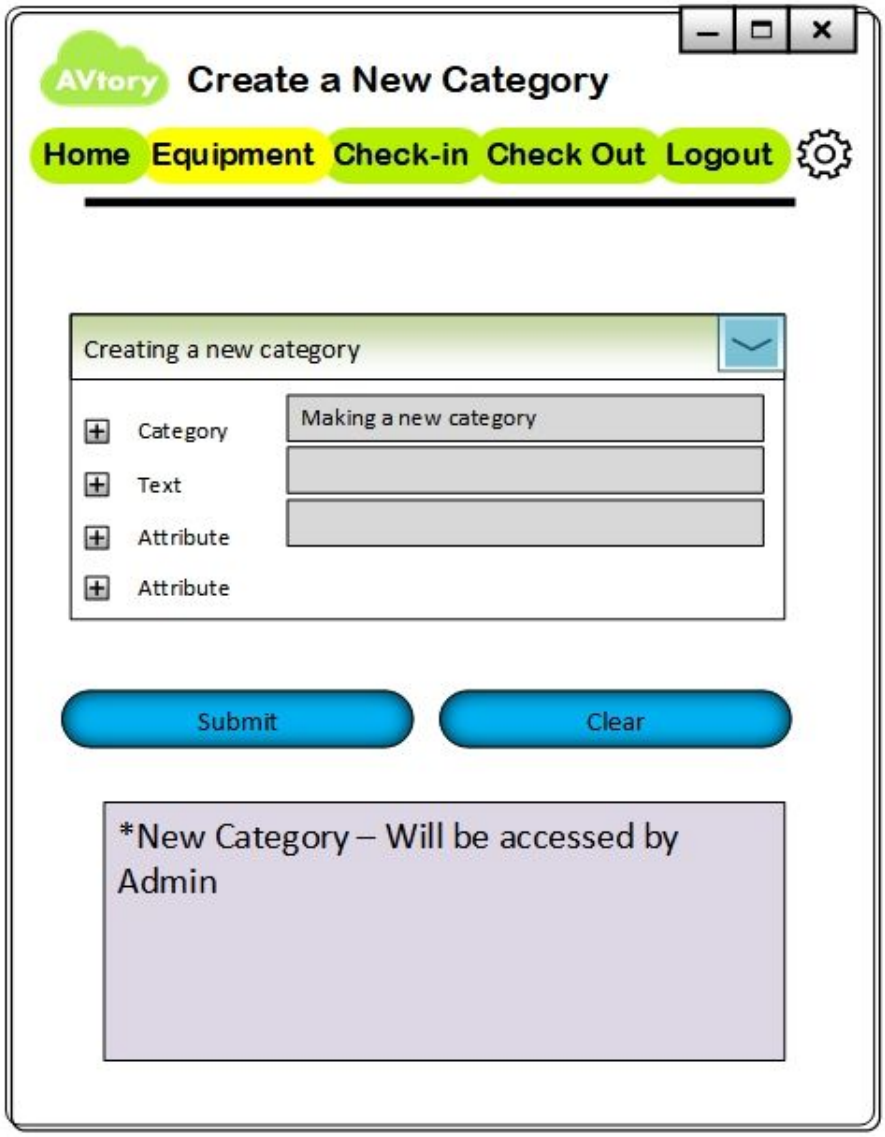
Wireframe - Delete Equipment Page



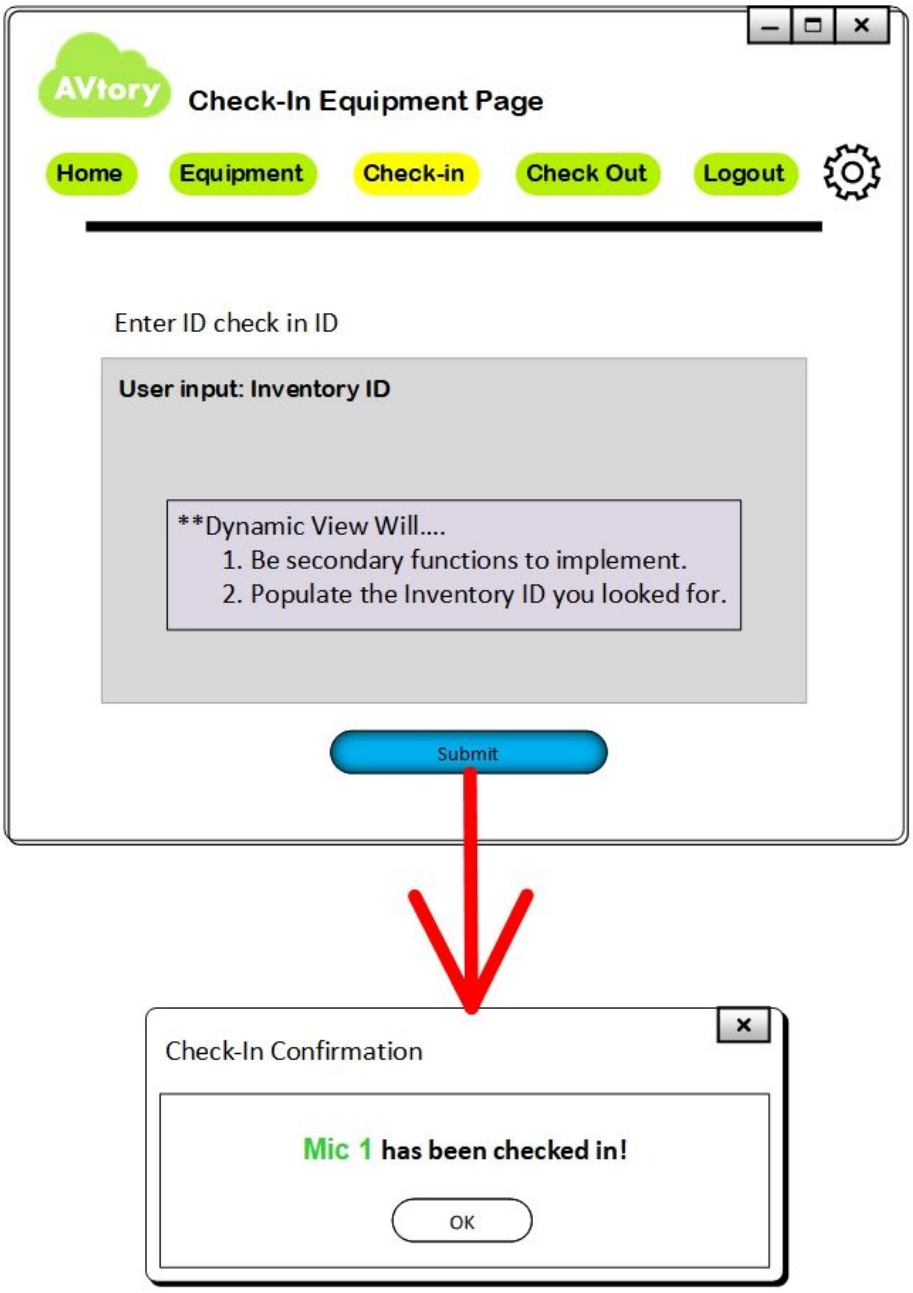
Wireframe - Edit Equipment Page



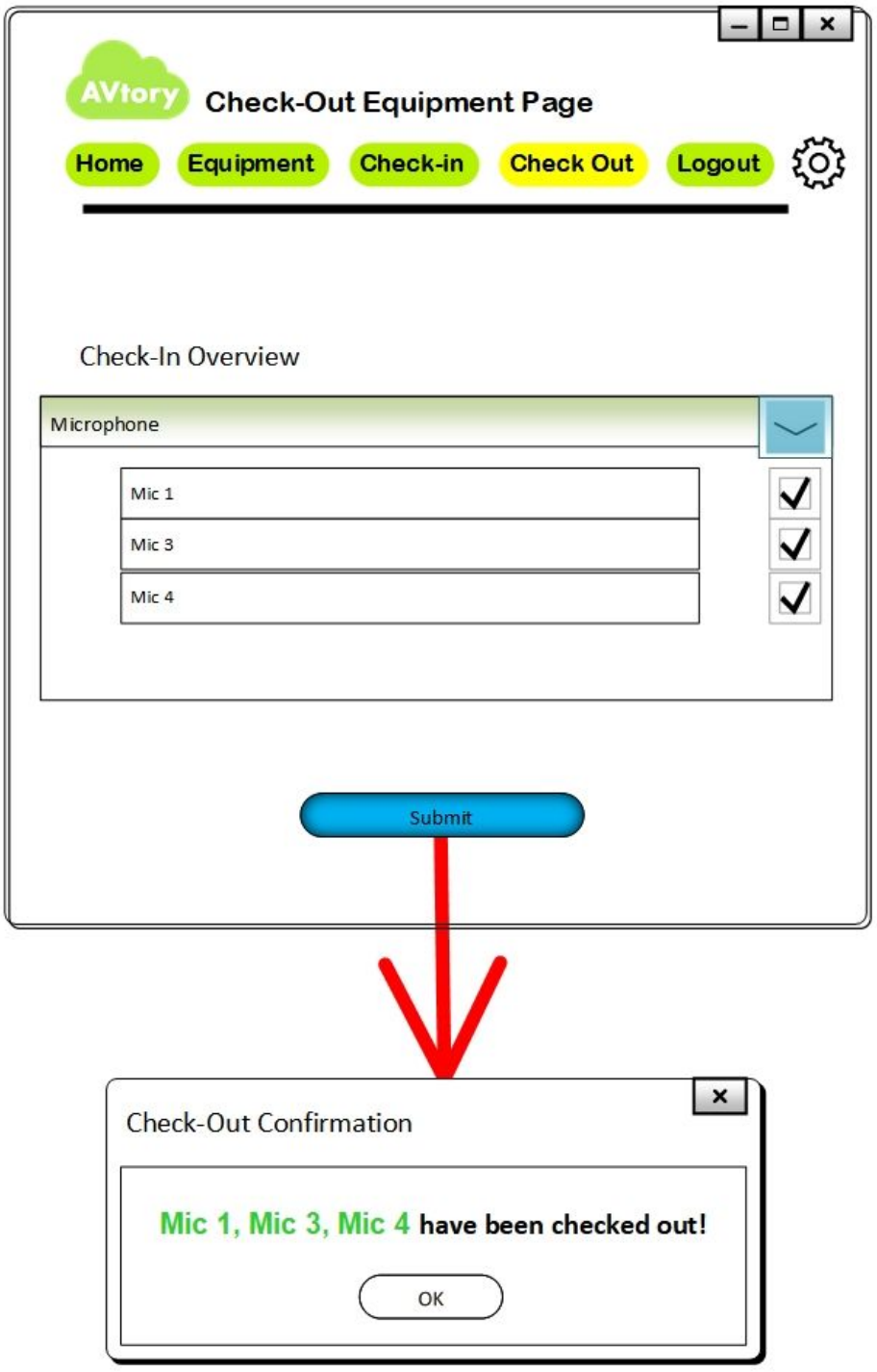
Wireframe - Create Category Page



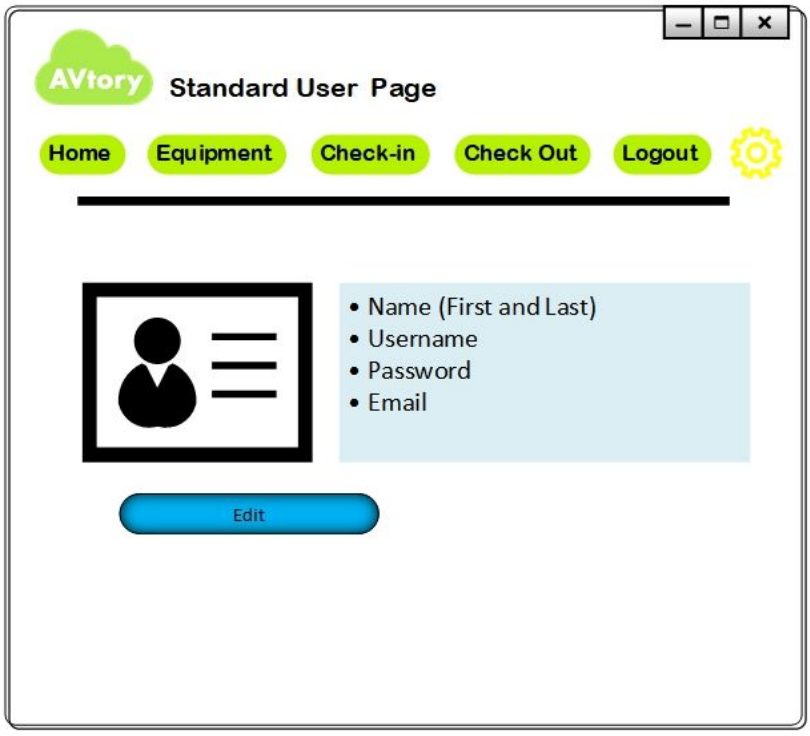
Wireframe - Check-In Equipment Page



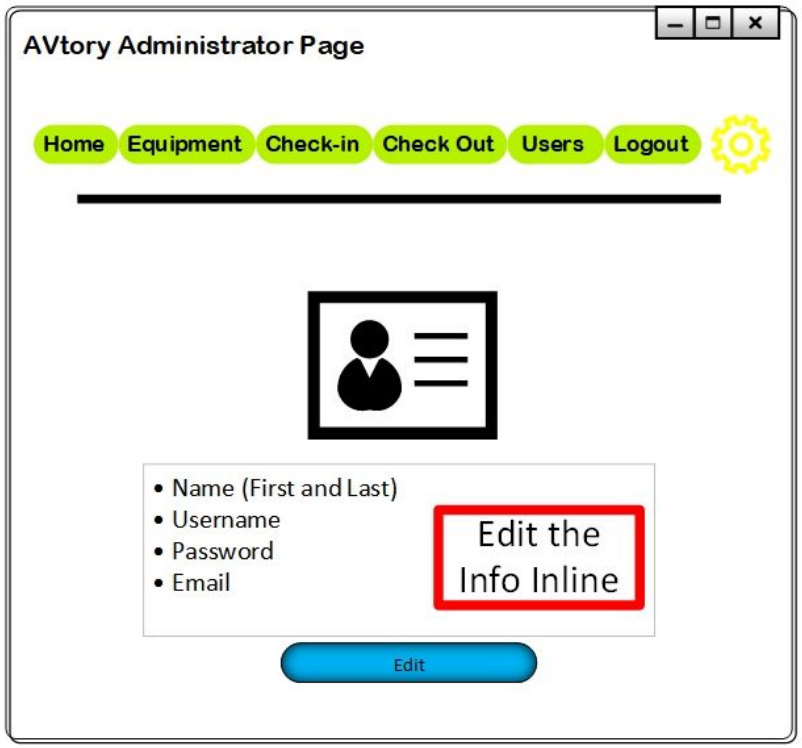
Wireframe - Checkout Equipment Page



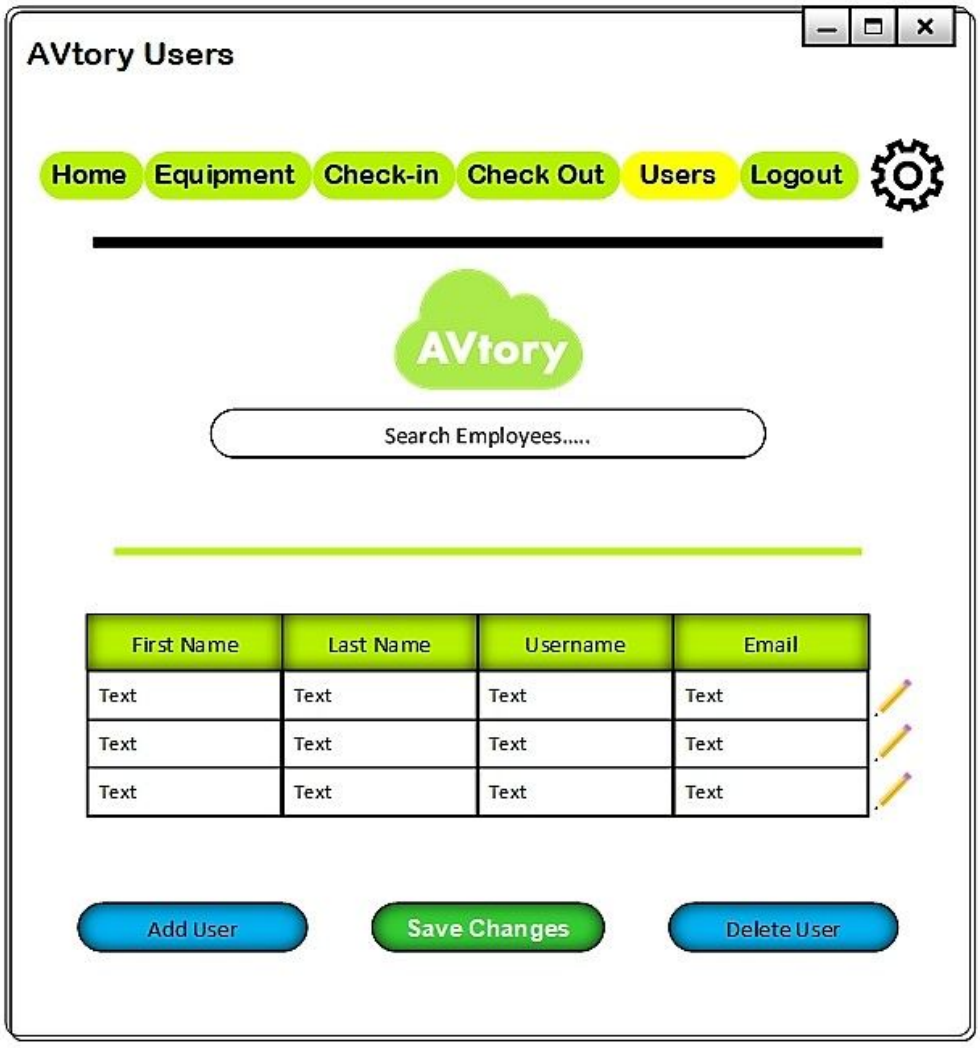
Wireframe - Standard User Profile Page



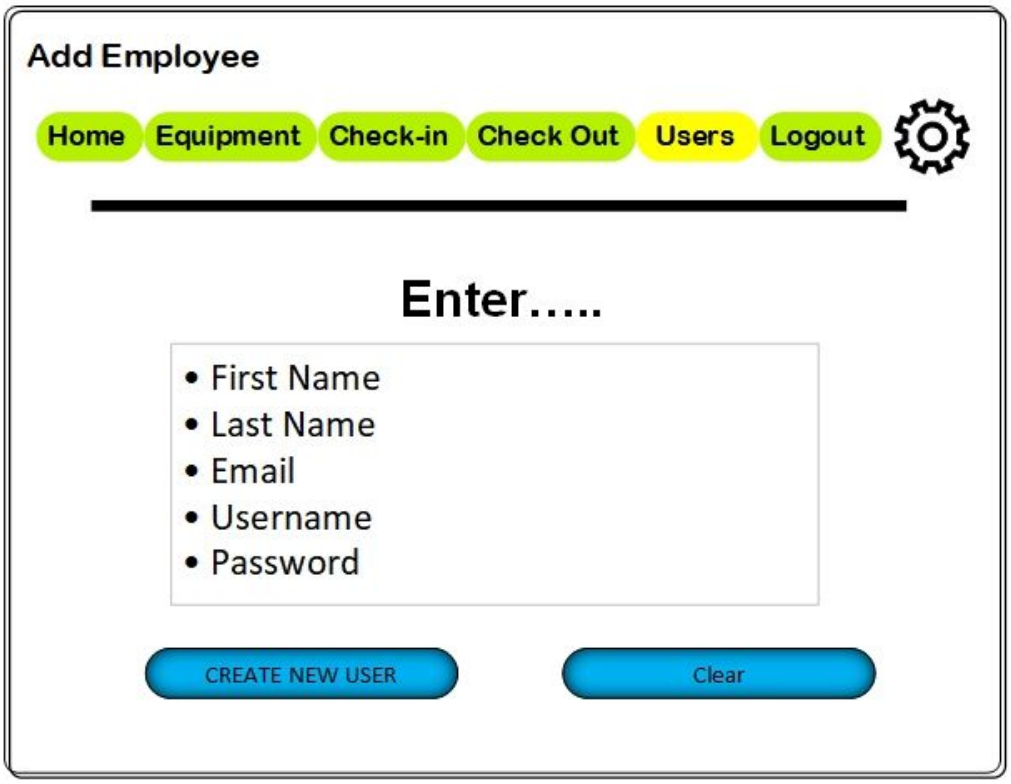
Wireframe - Administrative User Profile Page



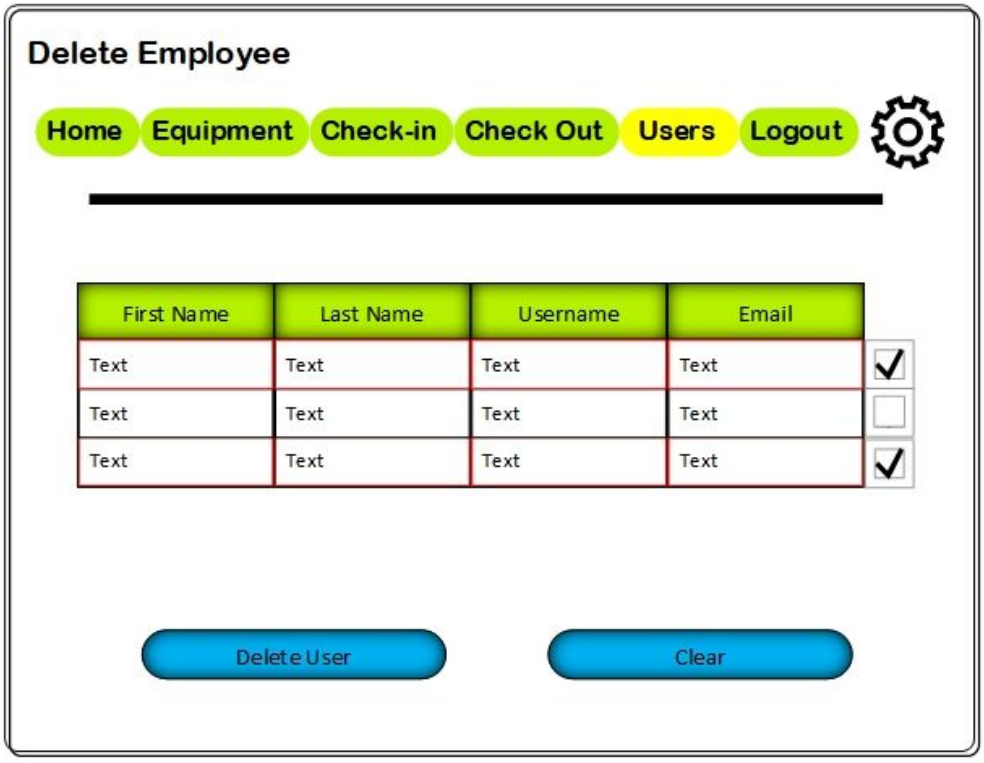
Wireframe - View Users Page



Wireframe - Add Employee Page



Wireframe - Delete Employee Page



Page Flow Diagram

The diagram below was the initial page flow diagram designed for the AVtory website, which illustrates how we envisioned a user would interact with the website.

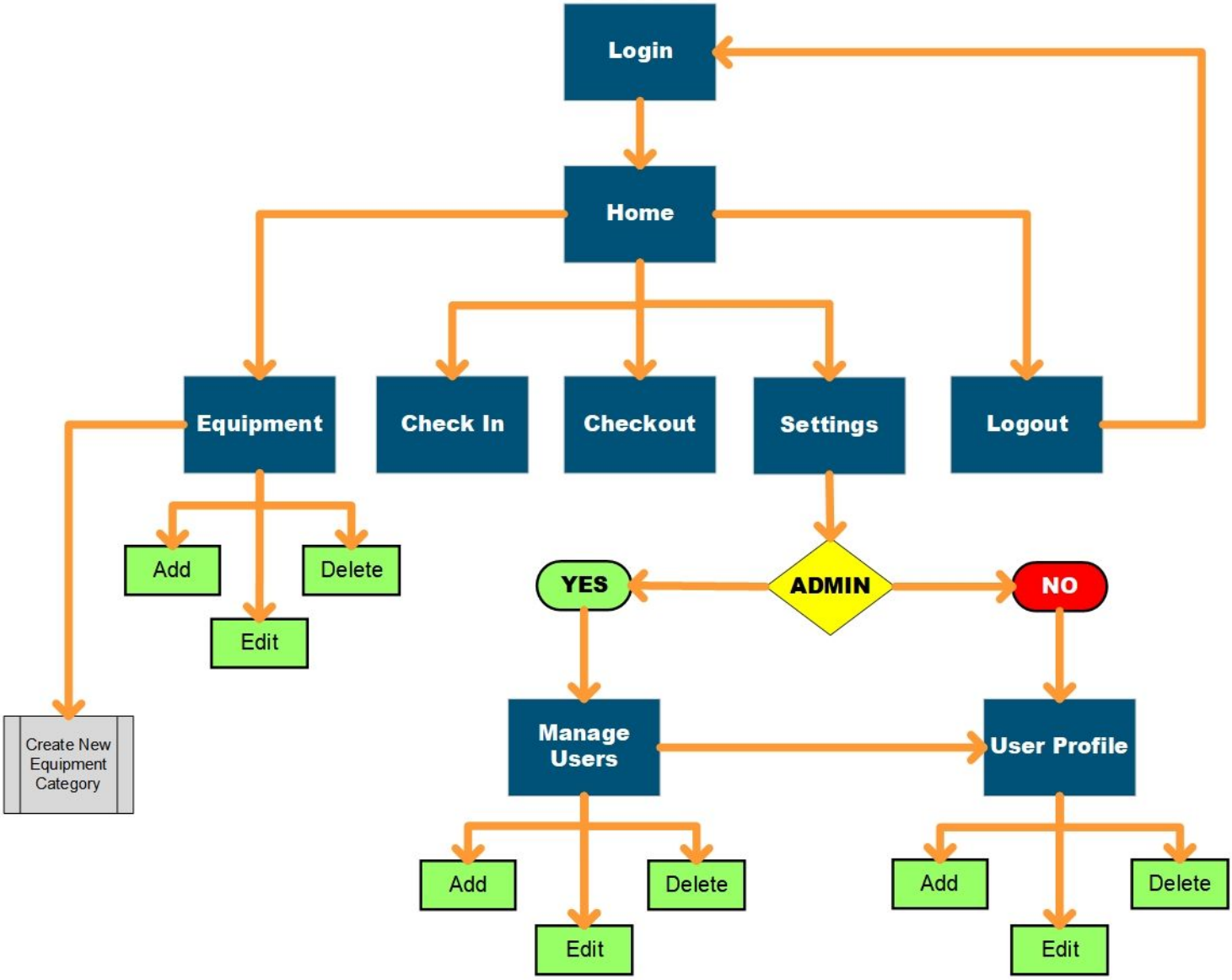
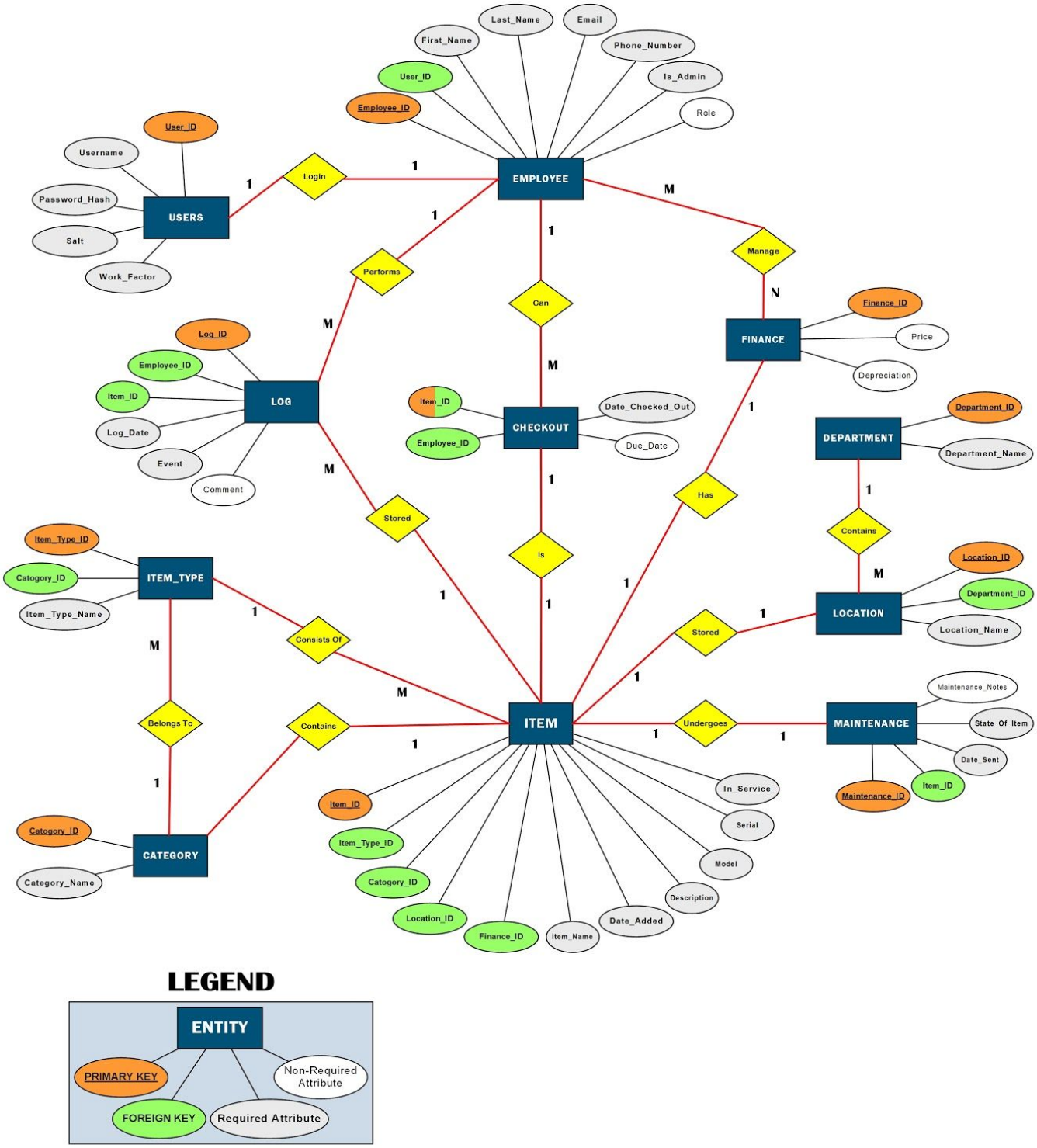


Diagram: Simplified Page Flow Diagram

DATABASE DESIGN

Database Entity Relationship Diagram



LEGEND

Diagram: ERD

Database Schema

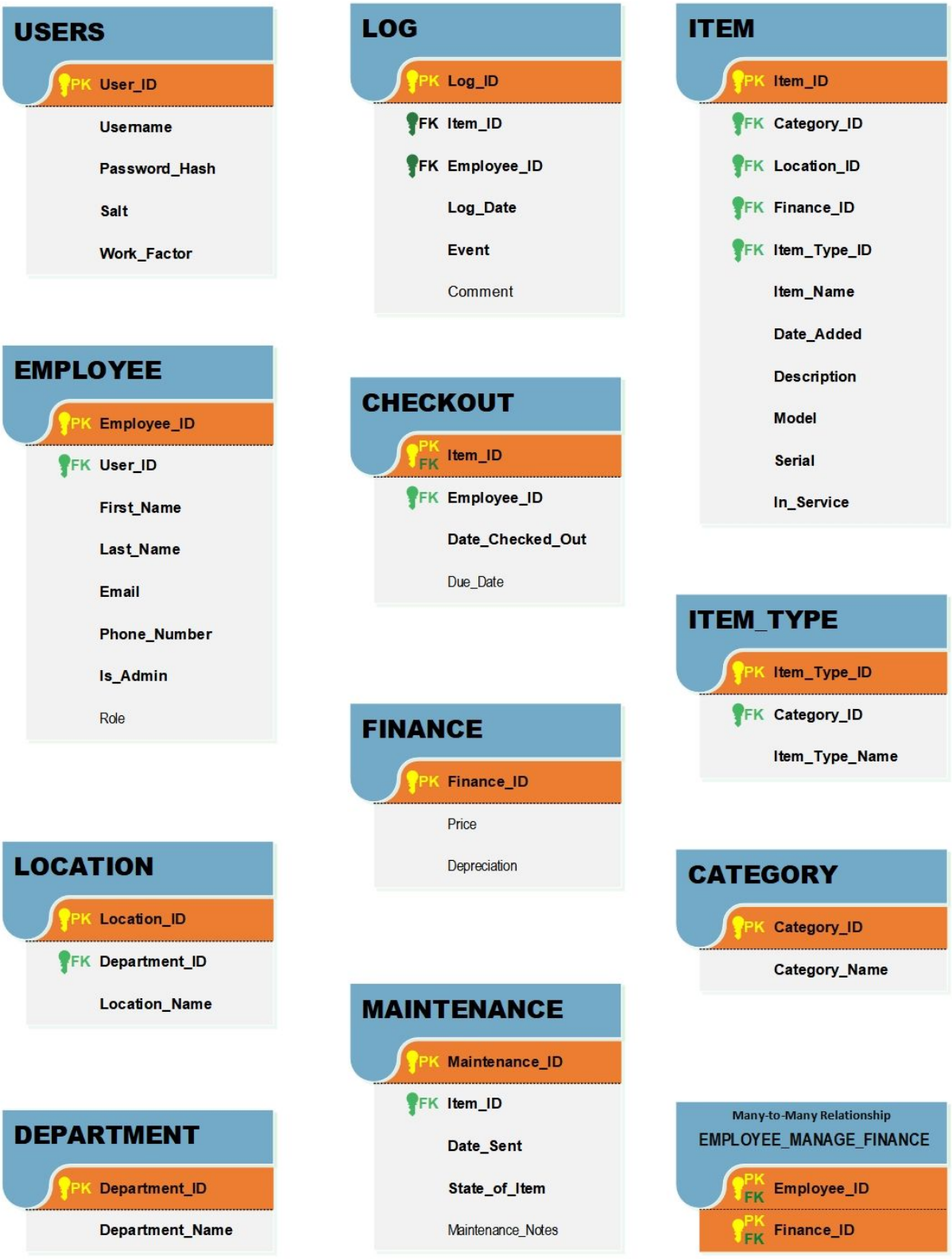


Diagram: AVtory Database Schema

DEFINITION OF ACTORS

The AVtory inventory management system has two types of users: Standard and Administrative.

Standard User

A Standard user is able to login to the website to manage all inventory by adding, editing, and deleting items from the inventory database. Standard users also have the ability to check-in and checkout items, as well as modify their user profile by editing their personal information. By default, accounts will be created as standard users.

Administrative Users

An Administrator manages all aspects of the inventory system. Along with having all privileges a standard user has, an administrator can also modify, add, and delete categories. Administrative users will also manage access to the inventory system by overseeing standard users. Administrators will have the ability to modify, add, and delete users from the system. Within the inventory management system, only administrative users will be able to add, delete, and modify categories and users.

UML

User Use Case Model

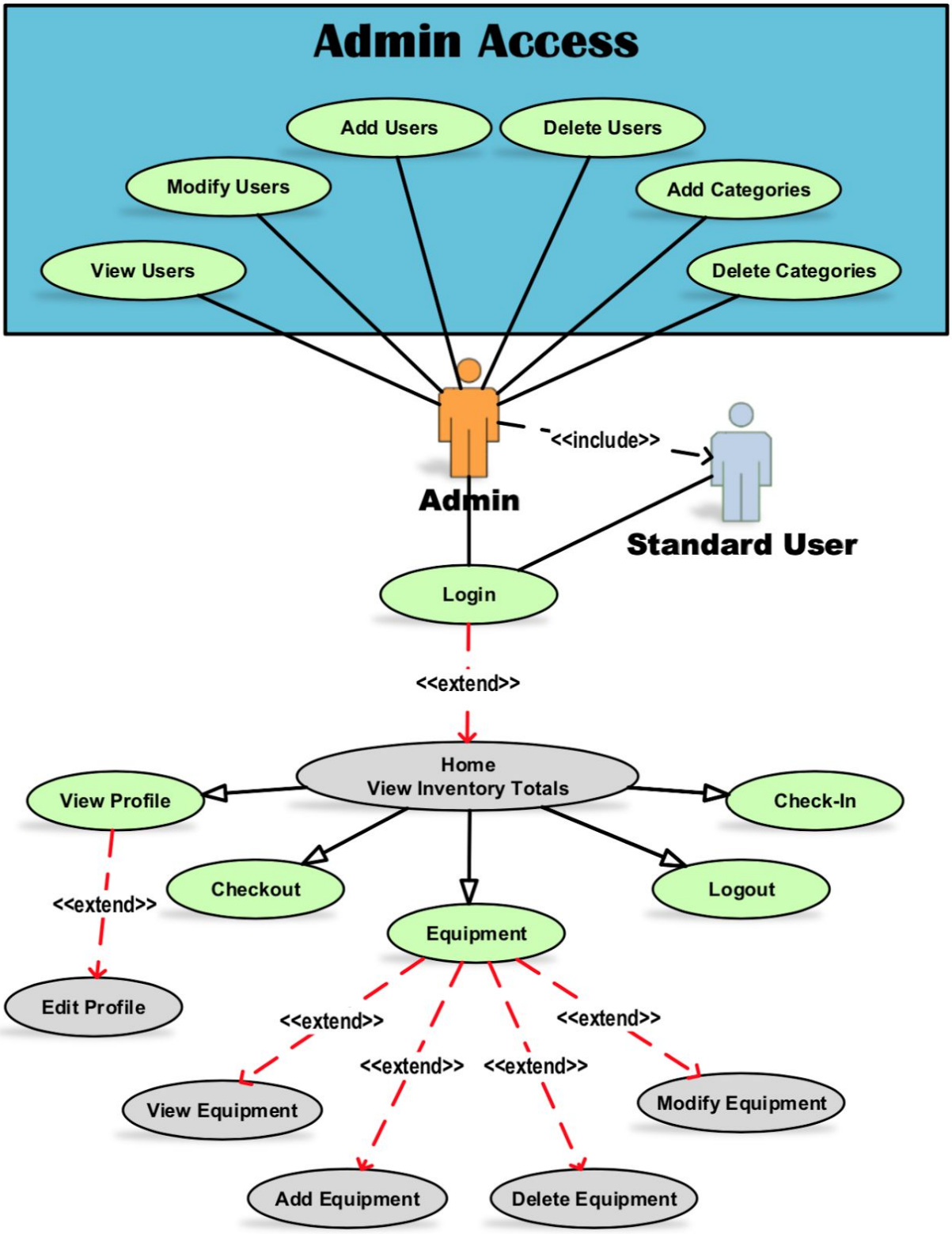


***Figure**: Use case diagram*

User Stories

As a standard user

- I can log in and out.
- I can view the equipment by category.
- I can check out a piece of equipment in stock.
- I can check in a piece of equipment that is currently checked out.
- I can add a piece of equipment to the system.
- I can delete a piece of equipment from the system.
- I can add a piece of equipment to maintenance.
- I can update a piece of equipment.

As an administrator

- I can access standard user functionality.
- I can grant users access to the system.
- I can revoke user access to the system.
- I can add or remove categories for equipment.

Formal Uses Cases

Standard User

Name: Register

Primary Actors: Standard User

Priority: High

Entry Condition: User has access to AVtory through a web browser

Exit Condition: User successfully registers to use AVtory

Flow of Events:

1. User has access to AVtory through a web browser.
2. User is not registered, user will fill out their name, email, and company information.
3. Validate user information, send to server to add into the system, prompt user to login.
4. Check what type of user: Administrator or Standard User.
5. User is acknowledged by web server to gain access.
6. AVtory displays data pertaining to that user's department.

Alternative Flows:

a. User has access, but is not recognized by the server.

b. Prompts user to register for AVtory.

Special Requirements: Login must be processed by the server quickly, at least within 30 seconds.

Name: User Login
Primary Actors: Standard User or Administrator User
Priority: High
Entry Condition: User has access to AVtory through a web browser
Exit Condition: User successfully logs in

Flow of Events:

1. User has access to AVtory through a web browser.
2. User logs in to AVtory with email and password.
3. Validate login information, server checks what type of user: Administrator or Standard User.
4. User is acknowledged by the web server and gains access.
5. AVtory displays relevant data pertaining to user's department.

Alternative Flows:

a. User has access, but is not recognized by the server.
b. Prompts user to register for AVtory.

Special Requirements: Login must be processed by the server quickly, at least within 30 seconds.

Name: View All Items
Primary Actors: Standard User or Administrator User
Priority: Medium
Entry Condition: User has privileges to access inventory and is on the equipment page
Exit Condition: User successfully adds new items

Flow of Events:

1. User clicks on “View All Items” button on the screen.
2. User can select to view by categories.
3. User can select to edit items.
4. Users can select to delete items.

Alternative Flows:

a. No data to display, will be prompted to add an item.

Special Requirements: The maximum time to display all items should not exceed 1 minute.

Name: Add Items
Primary Actors: Standard User or Administrator User
Priority: High
Entry Condition: User has privileges to access inventory and is on the equipment page
Exit Condition: User successfully adds new items

Flow of Events:

1. User clicks on the “Add Items” button on the screen.
2. User selects or enters the quantity of items to add.
3. User enters relevant information associated with adding new equipment items.

Alternative Flows:

a. User enters invalid information, prompt user with error message and the proper fields to enter.

Special Requirements: The maximum time to add a new item should not exceed 30 seconds.

Name: Edit Item
Primary Actors: Standard User or Administrator User
Priority: High
Entry Condition: User has privileges to access inventory and is on the equipment page
Exit Condition: User successfully edits an item

Flow of Events:

1. User clicks on the “Edit Items” button on the screen.
2. User selects “View All Inventory" or “View Inventory by Category” of items.
3. User selects the item to be edited, and updates the item.

Alternative Flows:

a. User inputs are invalid, prompt an error message with proper fields to enter.
b. User attempts to edit an item that does not exist.

Special Requirements: It takes no longer than 30 seconds to edit an existing item.

Name: Delete Item
Primary Actors: Standard User or Administrator User
Priority: High
Entry Condition: User has privileges to access inventory system
Exit Condition: User successfully deletes an item

Flow of Events:

1. User clicks on the “Delete Items” button on the screen.
2. User will select “View All Inventory” or “View Inventory by Category” of items.
3. User selects an item to be removed and deletes the item.

Alternative Flows:

a. User inputs are invalid, prompt an error message with proper fields to enter.
b. User attempts to delete an item that does not exist.

Special Requirements: It takes no longer than 30 seconds to delete an existing items.

Name: Edit User Profile
Primary Actors: Standard User and Administrator User
Priority: Medium
Entry Condition: User has privileges to access inventory
Exit Condition: User successfully updates profile information

Flow of Events:

1. User clicks on the “Settings” icon in the navigation bar.
2. User can modify and update profile information.

Alternative Flows:

a. User inputs are invalid, prompt an error message with proper fields to enter.

Special Requirements: It takes no longer than 30 seconds to update user information.

Name: Check Out Items
Primary Actors: Standard User and Administrator User
Priority: Medium
Entry Condition: User has privileges to access inventory
Exit Condition: User successfully checks out items

Flow of Events:

1. User clicks on the “Checkout” icon in the navigation bar or on the equipment page.
2. User inputs how many items they would like to check out.
3. Selects items to check out, either by view all, each category, or searching for an item to check out.

Alternative Flows:

a. User inputs are invalid, prompt an error message with proper fields to enter.
b. User attempts to check out an item that is either already checked out or in maintenance.

Special Requirements: It takes no longer than 30 seconds to update user information.

Name: Check In an Item
Priority: Medium
Entry Condition: User has privileges to access inventory
Exit Condition: User successfully checks out items

Flow of Events:

1. User clicks on the “Check In” icon in the navigation bar or on the equipment page.
2. User inputs items to be checked back in.

Alternative Flows:

a. User inputs are invalid, prompt an error message with proper fields to enter.
b. User is trying to check in an item that is already in inventory.

Special Requirements: It takes no longer than 30 seconds to update user information.

Administrator User

Name: View & Modify Users
Primary Actors: Administrator User
Priority: Medium
Entry Condition: User has Administrator privileges
Exit Condition: User successfully modifies existing users or adds new user

Flow of Events:

1. User clicks on the “Settings” icon in the navigation bar.
2. Administrator select users to add, edit, or delete.
3. Validate user information, update user information, or remove user from the database.

Alternative Flows:

a. User does not have administrator access.
b. User attempts to add a user that already exist, prompt gives option to edit the existing user information instead.
c. User attempts to delete a user that does not exist.

Special Requirements: It takes no longer than 30 seconds to add, delete, or update users.

Name: Add Department or New Inventory Category
Primary Actors: Administrator User
Priority: Medium
Entry Condition: User has Administrator privileges
Exit Condition: User successfully modifies existing category or adds new category

Flow of Events:

1. User clicks on the “New Category” or “New Department” button on the Equipment page.
2. Enters in the associated fields for a new category.
3. Validate user input and update the database.

Alternative Flows:

a. User does not have administrator access.
b. User attempts to delete a category that does not exist.
c. User attempts to edit a category that does not exist.

Special Requirements: It takes no longer than 30 seconds to add, delete, or update a category.

PARTIAL ANALYSIS MODEL

Class Case Diagram

AVtory does not use an object oriented programming (OOP) paradigm. OOP aims to increase cohesion of an application by encapsulating an object's data with its associated behavior. However, our application stores its state in a SQL database, which cannot encapsulate behavior. Our application communicates with the client over HTTP, which is a stateless protocol, and therefore cannot encapsulate state. These limitations greatly limit the utility of the OOP paradigm.

We have instead adopted a model-view-controller (MVC) architectural pattern. Our model is our SQL schema, which completely describes all data used by our system, with the exception of the session id cookie used to track sessions. Our view component is a series of HTML templates, which takes data as a Python dictionary of key-value pairs corresponding to data in our model, and outputs HTML ready to be served directly to the user. Our controller component is a series of Python functions which, in general, perform three basic tasks: check permissions of the user to perform the request, send or receive data from the SQL server, and either pass data into the view component for HTML rendering if data was pulled from the database, or pass the query to another controller component if data was modified in the database.

In this way, components are to be reused. Many components may need to generate a list of items, but a component which displays a list of items is to be written only once. Consider the sequence for viewing the list of items, compared to adding a new item. The use case for adding an item ends in viewing the list of items. In this case, the component to view a list of items is reused.

Deployment Model

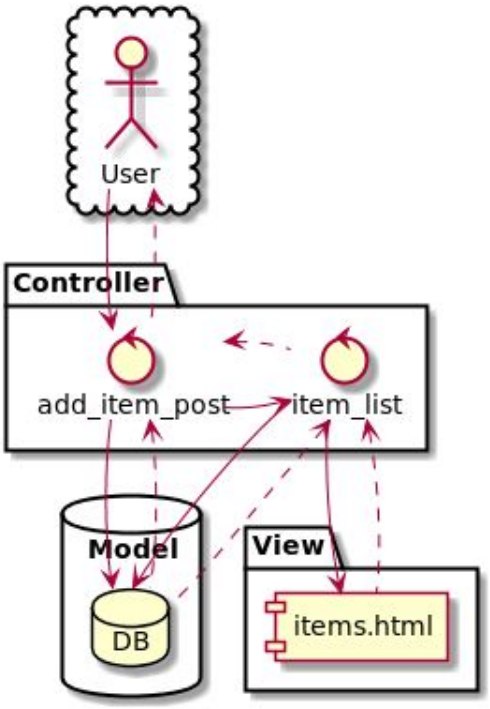


Diagram: Final step of item insertion activity, shown with MVC roles

Object Definition

	Object	Attributes & Responsibilities
Entity Classes	User	editProfile()
	Standard User Account	addItem() editItem() deleteItem()
	Administrator User Account	editUser() newCategory()
	Inventory	Item_ID: float category_ID: float checkout_ID: float inventory_ ID: string item_name: string description: string model: string serial: string in_service: string location_ID: float quantityOnHand: integer
		get_Input() update()
	Database Server	
Boundary Classes	Add new items Edit items Delete items	getInput() sendOutput() return()
Control Classes	Update inventory	

Table: Object definition

Sequence Diagram

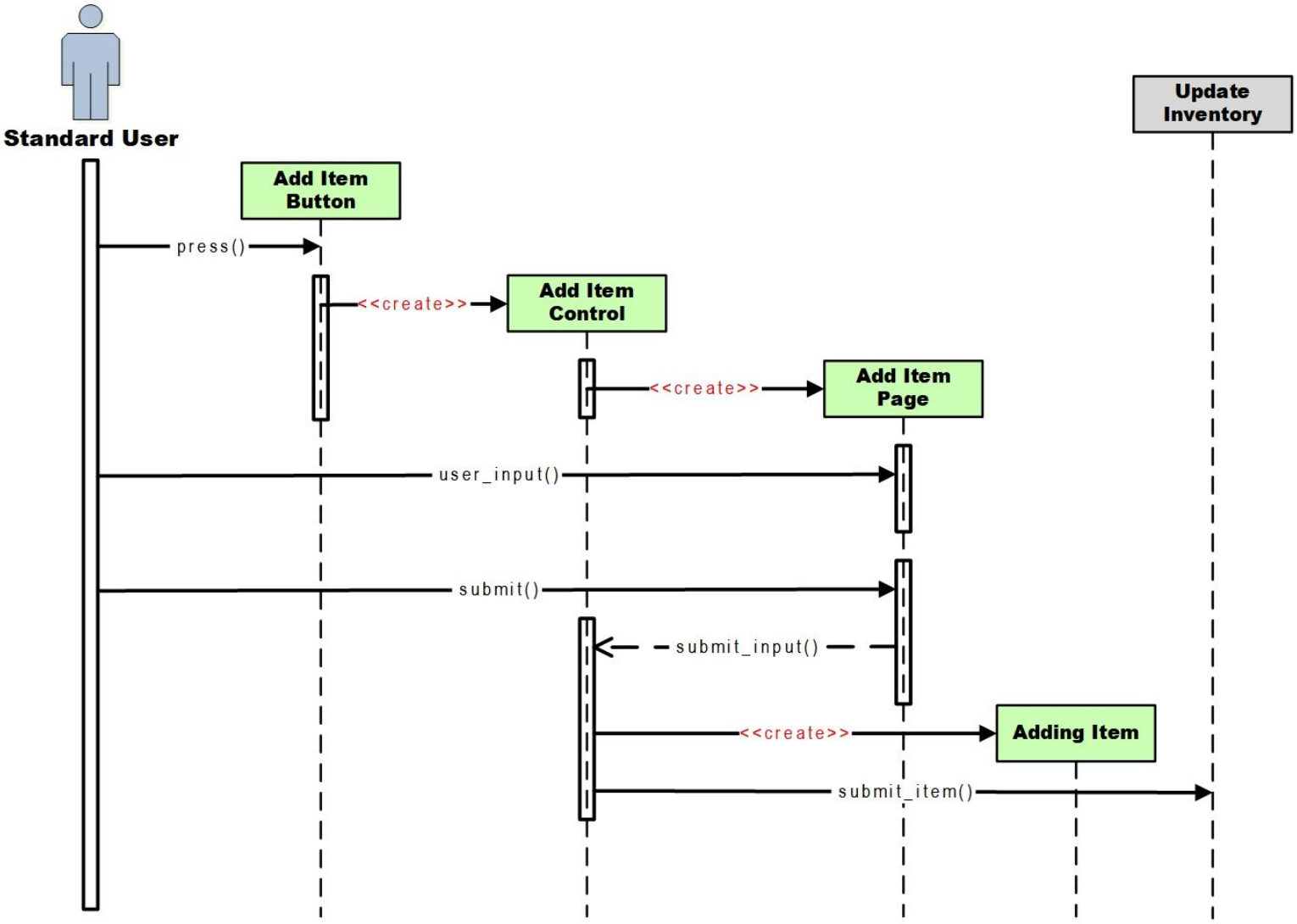


Diagram: Sequence Diagram of adding an item.

Boundary Classes: Add Item Button, Add Item Page

Control Class: Add Item Control, Update Inventory

Functions

press(): User presses the Add Item button whenever a new item needs to be added.

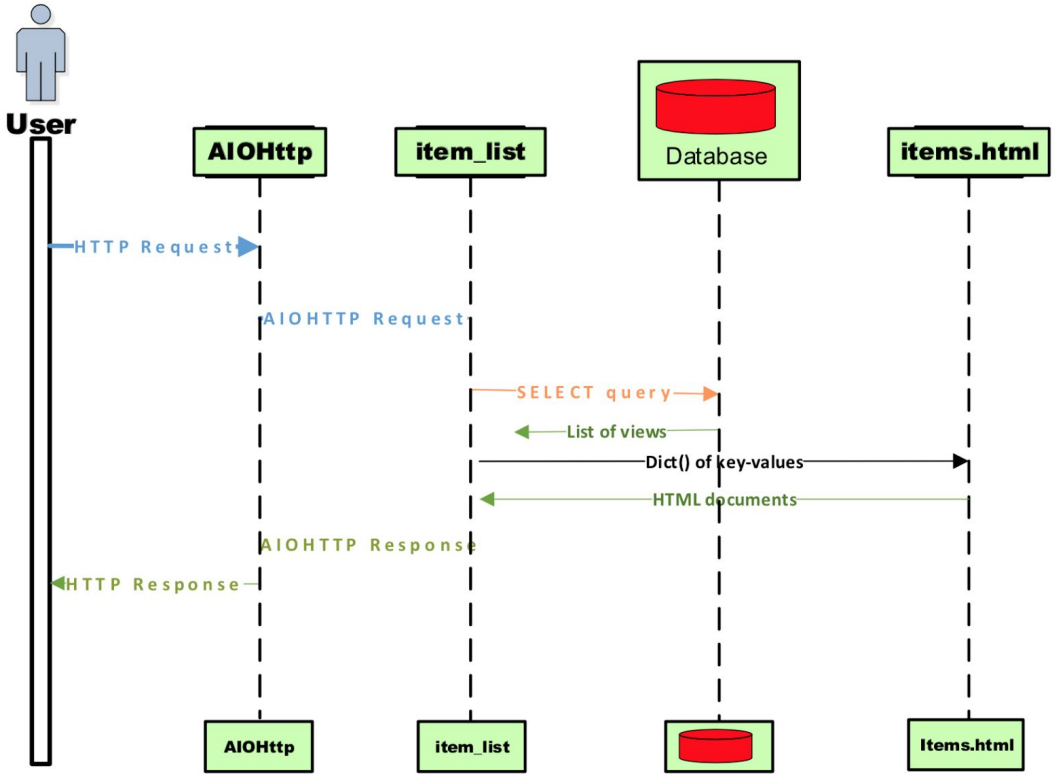
user_input(): User fills item details for new item.

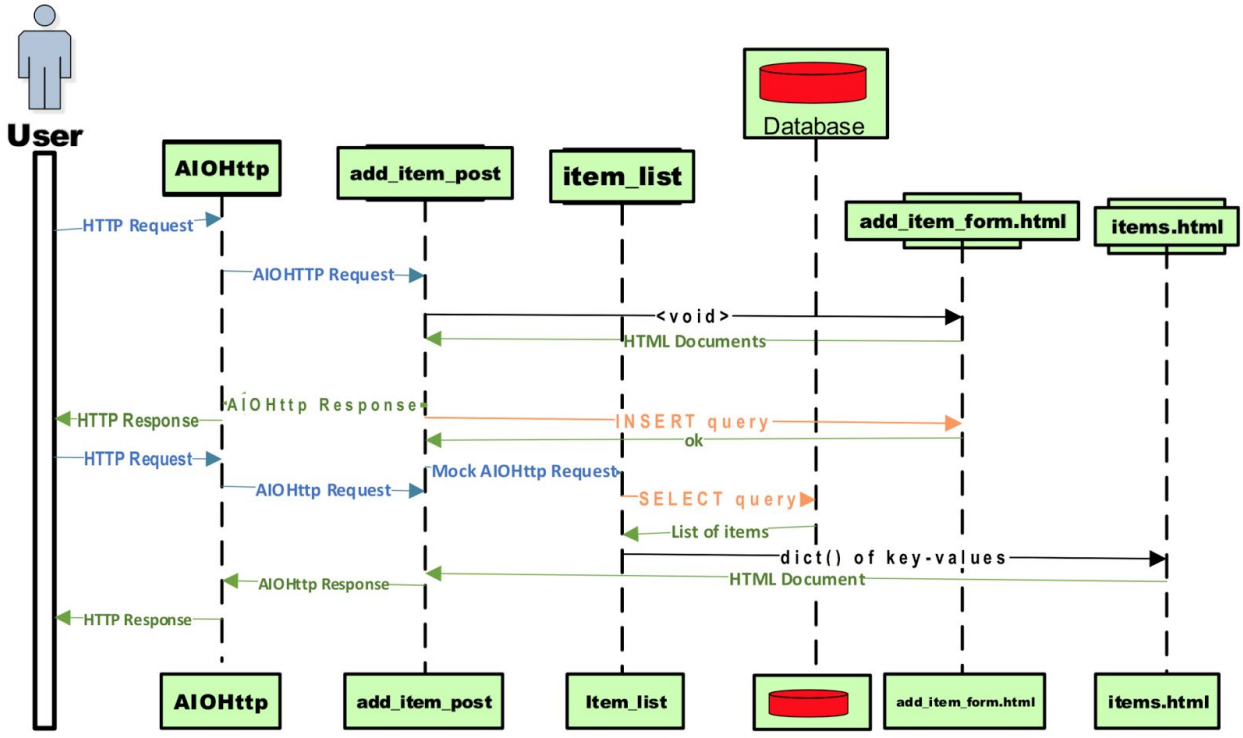
submit(): User submits the item details and it gets added to inventory.

user_item(): Is created and added to the database.

Sequence Diagrams

Diagrams: Sequences to view items compared to inserting items.





TEST AND ANALYSIS

Acceptance Testing and Functional Requirements Testing

Functions for test: User Login
Applicable Users: Test cases for Standard User and Administrative User

ID #	Test Case	Input data	Result	Satisfied?
1	Secure login	- Username - Password	User successfully logs in using their associated credentials and can access the inventory system.	Yes
2	Invalid login	- Username - Password	User attempts to login with invalid credentials. **Prompts**: Invalid username or password.	Yes

Functions for test: Registering and Editing Users
Applicable User: Test cases for Administrative User

ID #	Test Case	Input data	Result	Satisfied?
3	Administrative User Registers a Standard User	Username, first name, last name, email, phone number, admin, and role.	- Administrator User successfully creates and registers a new user.	Yes
4	Administrative User Edits a Standard User's information	Username, first name, last name, email, phone number, admin, and role.	- Administrator User successfully updates a Standard User's profile information.	Yes

Function for test: Add, Edit, and Delete Departments
Applicable Users: Test cases for Administrative Users

ID #	Test Case	Input data	Result	Satisfied?
5	Only an Administrative User has permission to add or delete a Department	- Department name to be added - Department name to be deleted	- Admin User successfully adds a new department - Admin User successfully deletes an existing department. - Standard Users do not have permission to add a new department name. - Standard Users do not have permission to delete an existing department name. - Standards Users do not have the ability to view the Add Department functionality or the Delete Department functionality.	Yes
6	Only an Administrative User has the ability to edit a Department	- Department name to edit	- Administrative User successfully updates an existing Department. - Standard Users do not have the ability to edit a Department. - Standard Users do not have the ability to view this option.	No

Function for test: Delete and Edit Locations
Applicable Users: Test cases for Administrative Users

ID #	Test Case	Input data	Result	Satisfied?
7	Only an Administrative User has permission to delete a Location	- Location name to be deleted	- Administrative User successfully deletes a Location. - Standard Users do not have the ability to delete a Location.. - Standard Users do not have the ability to view this option.	Yes
8	Only an Administrative User has the privilege to edit a Location	- Location name to edit	- Administrative User successfully updates an existing Location. - Standard Users do not have the ability to edit a Location. - Standard Users do not have the ability to view this option.	No

Function for test: Add Locations
Applicable Users: Test case for Standard User and Administrative User

ID #	Test Case	Input data	Result	Satisfied?
9	Both Standard and Administrative Users can add a new Location	- Location name to be added	- Administrative User successfully adds a new location. - Standard User successfully adds a new location.	Yes

Function for test: View Locations
Applicable Users: Test case for Standard User and Administrative Users

ID #	Test Case	Input data	Result	Satisfied?
10	Both Standard and Administrative Users can view the equipment associated with a Location	- Location name to be viewed	- Admin User successfully views all the equipment associated with the selected Location. - Standard User successfully views all the equipment associated with the selected Location.	Yes

Function for test: Add and Delete Category and Delete Item Type
Applicable Users: Test cases for Administrative Users

ID #	Test Case	Input data	Result	Satisfied?
11	Only an Administrative User has the ability to add a new Category	- Category name to be added	- Administrative User successfully adds a new Category - Standard Users do not have permission to add a new Category name. - Standards Users do not have the ability to view the Add Category option.	Yes
12	Only an Administrative User has the ability to delete an existing Category	- Category name to be deleted	- Administrative User successfully deletes an existing Category. - Standard Users do not have permission to delete an existing Category name. - Standard Users do not have the ability to view the delete Category option.	Yes
13	Only an Administrative user has permission to delete an Item Type	- Item Type name to be deleted	- Administrative User successfully deletes an existing Item Type. - Standard Users do not have permission to delete an existing Item Type name. - Standard Users do not have the ability to view the delete Item Type option.	Yes

Function for test: Add and View Item Type
Applicable Users: Test case for Standard User and Administrator Users

ID #	Test Case	Input data	Result	Satisfied?
14	Both Standard and Administrative Users can add Item Types	- Item Type name to be added.	- Administrative User successfully adds a new Item Type. - Standard User successfully adds a new Item Type.	Yes
15	Both Standard and Administrative Users can view the equipment associated with an Item Type	- Item Type name to be viewed	- Administrative User successfully views all the equipment associated with the selected Item Type. - Standard User successfully views all the equipment associated with the selected Item Type.	Yes

Function for test: Add, Delete, and Update Equipment Items
Applicable Users: Test cases for Standard User and Administrative Users

ID #	Test Case	Input data	Result	Satisfied?
16	Both Standard and Administrative Users can add equipment items.	- Department name, Location name, Category name, Item Type name, Item name, Item description, Model number, Serial number, In service, Price, and Depreciation	- Administrative User successfully adds a new Equipment Item. - Standard User successfully adds a new Equipment Item.	Yes
17	Both Standard and Administrative Users can delete equipment items.	- Item name to be deleted	- Administrative User successfully deletes an existing Equipment Item. - Standard User successfully deletes and existing Equipment Item.	Yes
18	Both Standard and Administrative Users can edit equipment items.	- Item information to be updated: Item name, description, model, or serial	- Administrative User successfully updates an existing Equipment Item. - Standard Users successfully updates an existing Equipment Item.	Yes

Function for test: Check-in and Checkout Items **Applicable Users:** Test cases for Standard User and Administrative Users				
ID #	**Test Case**	**Input data**	**Result**	**Satisfied?**
19	Both Standard and Administrative Users can check out an Item	- QR Code associated with an Item	- Administrative User successfully checks out an Item - Standard Users successfully checks out an Item	Yes
20	Both Standard and Administrative Users can check-in items	- QR Code associated with an Item	- Administrative User successfully checks in an Item - Standard Users successfully checks in an Item	Yes

Responsive Website

Function for test: Optimization for web and mobile **Applicable Users:** Test case for Standard User and Administrative Users on web browsers and mobile devices				
ID #	**Test Case**	**Input data**	**Result**	**Satisfied?**
21	Web browser responsiveness	Web browser view	- User Interface maintains a user friendly view. - Scalable to web browser.	Yes
22	Mobile Web browser responsiveness	Mobile device web browser view	- Scalable to web browser. - Dropdown navigation bar for mobile devices.	Yes

IMPLEMENTATION MILESTONES

The following are the milestones and deliverables that were implemented throughout this project.

Milestone 1 - Design User Interface

- Determined how many pages are needed & Design Wireframes.
- Created Page Flow Diagram.
- Created wireframes for webpages.
- Revised wireframes based on team feedback.
- Obtained approval to proceed.
- Finalized wireframe.

Milestone 2 - Database Design

- Obtained AV Equipment Data excel sheets.
- Cleaned up excel sheets and identify relationships between data.
- Created initial ERD and Schema based AV data.
- Asked necessary questions to normalize tables.
- Revised ERD based on team feedback.
- Finalized initial database design.

Milestone 3 - Create System Architecture

- Created AVtory Amazon Web Services account.
- Established Groups and Users using IAM for security and access management.
- Created AVtory VPC and established subnets.
- Launched EC2 instance and created web server.
- Established RDS instance for MySQL database.
- Finalized cloud-based system architecture .

Milestone 4 - Implement Initial Web Pages

- Created HTML web pages from wireframes.
- Linked web pages based on page flow.
- Revised website based on team feedback.
- Completed initial web pages.

Milestone 5 - Implement Database

- Utilized ERD to create MySQL tables.
- Created AVtory database on RDS instance.
- Generated tables and established relationships and constraints between tables.
- Populated database with initial AV data for testing.

Milestone 6 - Establish Initial Connection between Frontend and Backend

- Migrated from school server to AWS EC2.
- Configured Apache and installed Python on server.
- Utilized web pages to create a visual gateway to display user information from the database via Python.

Milestone 7 - Enhance Website Functionality

- Integrated Bootstrap to create a clean and simple interface to enhance the visual gateway of user services.
- Improved page flow functionality via buttons, navigation bar, and pop-up menu.
- Implemented forms to gather user input for specific user requirements.
- Enhanced system security by implementing login functionality to support different user views based on user access credentials.

Milestone 8 - Store and Retrieve Data from Database for Specific Pages

- Implemented add and delete functionality for the following pages: Departments, Locations, Categories, Item Types, and User Management. Successfully added department and locations for inventory items.
- Implemented user profile page which provides add, delete, and update functionality.

Milestone 9 - Refined Storage and Retrieval of Data from Databases

- Implemented edit and delete item functionalities.
- Implemented check-in and checkout process.
- Displays recent information on homepage for added items, as well as checked in and check out.

Milestone 10 - Pursue Secondary Objectives

- Generated QR codes for each item.
- Implemented QR code scanning functionality for check process.

PROJECT MANAGEMENT
Gantt Chart

	Task Name	Duration	Start	Finish	Predecessors	Resource Names
	Software Development	**64 days**	**Fri 9/15/17**	**Wed 12/13/17**		
	Planning Phase	**7 days**	**Fri 9/15/17**	**Mon 9/25/17**		
	Determine project scope	2 days	Fri 9/15/17	Mon 9/18/17		
	Define preliminary resources	4 days	Tue 9/19/17	Fri 9/22/17	2	
	Group Meet practice presentation	1 day	Mon 9/25/17	Mon 9/25/17	3	
	Scope complete Present	0 days	Mon 9/25/17	Mon 9/25/17	4	
	Design Phase	**26 days**	**Wed 9/27/17**	**Wed 11/1/17**		
	Review preliminary software specifications	2 days	Mon 10/2/17	Tue 10/3/17		Designer
	Wire Frames	**21 days**	**Wed 10/4/17**	**Wed 11/1/17**	**7**	**Designer**
	Determine how many pages are needed & Design Wireframes	14 days	Wed 10/4/17	Mon 10/23/17		Designer
	Create Page Flow Diagram	3 days	Tue 10/24/17	Thu 10/26/17	9	Management
	Incorporate feedback into functional specifications	1 day	Fri 10/27/17	Fri 10/27/17	10	Management
	Obtain approval to proceed	1 day	Mon 10/30/17	Mon 10/30/17	11	
	Design complete	2 days	Tue 10/31/17	Wed 11/1/17	12	Team
	Design Database	**19 days**	**Wed 9/27/17**	**Mon 10/23/17**		
	Obtain AV Equipment Data	1 day	Wed 9/27/17	Wed 9/27/17		
	Identify relationships between data	5 days	Thu 9/28/17	Wed 10/4/17	15	
	Create initial tables	1 day	Thu 10/5/17	Thu 10/5/17	16	
	Normalize tables	3 days	Fri 10/6/17	Tue 10/10/17	17	
	Create ERD	2 days	Wed 10/11/17	Thu 10/12/17	18	Team
	Create Schema	2 days	Fri 10/13/17	Mon 10/16/17	19	
	Populate final tables & test	2 days	Tue 10/17/17	Wed 10/18/17	20	
	Identify descrepencies & revise if necessary	3 days	Thu 10/19/17	Mon 10/23/17	21	
	Implementation Phase	**14 days**	**Thu 11/2/17**	**Tue 11/21/17**	**13**	
	Web	**14 days**	**Thu 11/2/17**	**Tue 11/21/17**	**8**	
	Review functional specifications	3 days	Thu 11/2/17	Mon 11/6/17		
	Implement view pages	1 day	Tue 11/7/17	Tue 11/7/17	25	Denise
	Implement pages	14 days	Thu 11/2/17	Tue 11/21/17		Jon
	Database	**4 days**	**Thu 11/2/17**	**Tue 11/7/17**	**14**	
	Create MySQL for each table	1 day	Thu 11/2/17	Thu 11/2/17		Jason
	Populate database	3 days	Fri 11/3/17	Tue 11/7/17	29	Denise
	Generate list of commonly used queries	3 days	Thu 11/2/17	Mon 11/6/17		
	Connect Database to Web Pages	**14 days**	**Thu 11/2/17**	**Tue 11/21/17**		
	Implement PHP or Python to establish initial connection	2 days	Wed 11/8/17	Thu 11/9/17	26,30	Nick
	Implement PHP or Python queries for specific pages	10 days	Thu 11/2/17	Wed 11/15/17		Nick
	Review, Test, and Revise as necessary	4 days	Thu 11/16/17	Tue 11/21/17	34	Team
	Software development AVtory complete	12 days	Tue 11/28/17	Wed 12/13/17		

Chart: Gantt Chart Software Development Plan

CONCLUSION

Future Work and Enhancements

AVtory's inventory management system can be improved through the following enhancements:

- Refined data validation
    - No duplicate entries for Department, Locations, Categories, or Items Type
- Search functionality
- Improved QR scanning functionality

Summary

Overall, this inventory management system meets all the user requirements by providing a secure user-friendly interface with the ability to add, delete, and track equipment via a check-in and checkout process.

Along with this, by completing the project's primary objectives ahead of schedule, we were able to implement additional features, such as the ability to access the system off campus, and a QR code scanner to enhance the check-in and checkout process.

OFFICIAL LINKS

The resources below assisted in the development of this software system:

- AVtory GitHub
- AVtory Website
    - Username - cs441
    - Password - cs441
- AVtory Final Presentation

REFERENCES

The resources below assisted in the development of this software system:

- Bootstrap
- MySQL Workbench
- AWS
- QR Code

AVtory Final Presentation

SPRINT LOGS

Week 3 Meeting Summary

Week 3 Meeting September 17th - 11:00am to 2:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	N/A	Yes
Jason Sneddon	N/A	Yes
Jonathan Tapia	N/A	Yes

Week 3 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Determine project scope	Yes	Identified project scope
Define preliminary resources	Yes	Established required resources to complete project
Begin project proposal & presentation	No	Initial outline of project proposal and presentation.

Week 3 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
N/A	Denise Nguyen	Work on the Project Proposal and Powerpoint. Design a AWS architecture diagram.
N/A	Jason Sneddon	Established outline for Project Proposal
N/A	Jonathan Tapia	Work on the Project Proposal and Powerpoint.

Week 4 Meeting Summary

Week 4 Meeting September 24th - 12:30pm to 3:30pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	N/A	Yes
Jason Sneddon	N/A	Yes
Jonathan Tapia	N/A	Yes
Nick Wallingford	N/A	Yes - Virtual

Week 4 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Review Project Proposal Requirements	Yes	Understood project proposal assignment
Discuss Project Scope & Specifications	Yes	Defined project requirements, scope, and tools
Final Project Proposal	Yes	Completed *Project Proposal* assignment
Create & Practice Project Proposal Presentation	Yes	Created powerpoint presentation and rehearsed speaking points

Week 4 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Completed the Project Proposal and PPT - Designed a AWS architecture diagram - Created AVtory logo	Denise Nguyen	- Complete Project Plan. - Create wireframes on Visio. - Create initial ERD.
- Flushed out sections of Project Proposal - Presentation	Jason Sneddon	- Begin design for initial wireframes - Create AWS account
- Completed project proposal - Added presentation notes	Jonathan Tapia	- Determine use cases for work on ERD
- Completed basic “Hello world” application to serve content	Nick Wallingford	- Push code to github and begin producing meaningful application

Week 5 Meeting Summary

Week 5 Meeting October 1st - 12:00pm to 5:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	N/A	Yes
Jason Sneddon	N/A	Yes
Jonathan Tapia	N/A	Yes
Nick Wallingford	N/A	No

Week 5 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Review project plan	Yes	Brief project overview and updated Project Plan
Review project specifications	Yes	Discussed data for database
Start web application design	Yes	Completed initial design of wireframes for Users, Check-In, Check-Out, Equipment, New Category pages.
Distribute AWS credentials	Yes	Issued login credentials for private AVtory AWS account to team.
Gather all AV data for database	Yes	Uploaded all AV data to Google Drive for team access

Week 5 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Completed Project Plan. - Discussed wireframes; Visio, initial ERD.	Denise Nguyen	- Finalize wireframes
- Started wireframes & created AVtory AWS	Jason Sneddon	- Finalize wireframes
- Use cases for wireframes	Jonathan Tapia	- Work on preliminary ERD
- Create basic SQL schema	Nick Wallingford	- Continue writing structural code to speak between backend and database

Week 6 Meeting Summary

Week 6 Meeting October 7th - 1:30pm to 4:30pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Web Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Web Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 6 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Review wireframes	Yes	Finalized the design for web application
Review database requirements	Yes	Analyzed AV data and established inventory management system requirements
Begin initial database design	Yes	Developed initial ERD for AVtory database
Review project plan	Yes	Updated project plan and task list
Establish team roles	Yes	Defined roles for each team member

Week 6 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Completed wireframes	Denise Nguyen	- Begin implement of web app. - Start Project Requirements.
- Finalized wireframes	Jason Sneddon	- Review ERD & create AWS infrastructure - Establish report outline
- Discussed preliminary ERD	Jonathan Tapia	- Formally write up user stories
- Created github repo for back-end code - Implemented session tracking	Nick Wallingford	- Connect back-end code to database - Begin implementation of user logins

Week 7 Meeting Summary

Week 7 Meeting October 15th - 3:00pm to 6:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Web Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Web Developer	Yes
Nick Wallingford	Back-end Developer	No

Week 7 Agenda		
Agenda Item	**Completed**	**Summary**
Fill out prior week sprint logs	Yes	Updated Sprint Log
Fill out current week spring log	Yes	Team updated each other of work completed
Finalize Report 2	75%	Team members filled in sections of report
Create Use Case Presentation	No	Absent team member
Update project plan at end of meeting	Yes	Updated project plan to reflect sprint log
Meeting summary	Yes	Establish future sprint goals & meeting

Week 7 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Started web implementation.	Denise Nguyen	- Implement Menu bar, Home & Eq. page - Use Case diagrams
- Created AWS architecture & revised ERD - Created web server & MySQL database	Jason Sneddon	- Finalize ERD & create Schema - Generate MySQL tables
- User Stories for report	Jonathan Tapia	- Implement web pages
- Did UML diagrams for basic SQL schema and user stories - Implemented very basic and insecure logins	Nick Wallingford	- Do user logins correctly and securely

Week 8 Meeting Summary

Week 8 Meeting October 18th 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Web Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 8 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out week sprint logs	Yes	Updated logs
Discuss ERD for database. Possibly finalize DB	Yes	Design beginning finalization
Bootstrap	Yes	Navigation starting implementation
AWS	Yes	Minor connections made
Update project plan	Yes	Updated properly
Meeting summary - establish future sprint goals & reserve room	Yes	Goals and location set

Week 8 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Linked Web Pages	Denise Nguyen	- Continue work on web pages
- Create ERD - Create Schema - Generate MySQL tables	Jason Sneddon	- Setup MySQL Workbench - Populate tables
- NONE	Jonathan Tapia	- Work on web pages
- Log in possible	Nick Wallingford	- Update log ins and connections

Week 9 Meeting Summary

Week 9 Meeting October 25th - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 9 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out week sprint logs	Yes	Updated logs
Discuss ERD for database. Possibly finalize DB	Yes	Populating database by next week. Being able to add an item
Bootstrap	Yes	Site updated with latest changes
AWS	Yes	Set up and ready for connections
Update project plan before meeting ends	Yes	Project plan updated

Week 9 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Web page connections	Denise Nguyen	- Test websites
- Setup MySQL Workbench - Populate database with initial AV data	Jason Sneddon	- Manage access to RDS - Revise database keys and constraints
- NONE	Jonathan Tapia	- Research functionality extensions
- Preliminary site up	Nick Wallingford	- Test database connections

Week 10 Meeting Summary

Week 10 Meeting November 1st - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 10 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out week sprint logs	Yes	Logs completed
Discuss DB connections	Yes	Making sure everything works with AWS
Website	Yes	User add/edit/delete now possible
Schema	Yes	Everything working except for locations
Meeting summary	Yes	Goals and location set

Week 10 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Websites tested	Denise Nguyen	- Research SMS functionality
- Created rules to manage access to RDS - Revised database keys and constraints	Jason Sneddon	- Manage AWS
- Minor report changes	Jonathan Tapia	- Work on javascript for pages
- Connected HTML and PHP	Nick Wallingford	- Test connections

Week 11 Meeting Summary

Week 11 Meeting November 8th - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 11 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out week sprint logs	Yes	Logs completed
Database	Yes	Departments and Locations can now be added, listed, and deleted.
Website	Yes	Fully working site
Meeting summary	Yes	Goals and location set

Week 11 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Website changes	Denise Nguyen	- Preliminary report work
- Manage AWS	Jason Sneddon	- Revise database design - Update database diagrams - Update report
- NONE	Jonathan Tapia	- Update report - Additional research
- Database now handles proper functionality	Nick Wallingford	- Test database connections

Week 12 Meeting Summary

Week 12 Meeting November 15th - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 12 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out week sprint logs	Yes	Logs updated
Discuss database design and implementation	Yes	More connections made and tested
Discuss website implementation and features	Yes	Fully connected
Meeting summary	Yes	Goals set and work done

Week 12 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Updated report structure / Rough draft	Denise Nguyen	- Report - Presentation
- Updated database design and diagrams - Report	Jason Sneddon	- Report - Presentation
- NONE	Jonathan Tapia	- Report - Presentation
- Bug found - Possible feature?	Nick Wallingford	- Report - Presentation

Week 13 Meeting Summary

Week 13 Meeting November 20th - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 13 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out week sprint logs	Yes	Logs updated
Discuss database	Yes	Queries to be implemented and tested
Discuss features to implement for website	Yes	Prioritized features to be implemented
Meeting summary	Yes	Discussed project goals and work to be done

Week 13 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Attempted to work on “edit item” in Python	Denise Nguyen	- Improve aesthetics
- Managed AWS connections	Jason Sneddon	- Assist with aesthetics
- NONE	Jonathan Tapia	-
- Implemented View item functionality - Implemented Edit item feature - Cleaned up code - Beautified display of data	Nick Wallingford	- Manage Github repositories

Week 14 Meeting Summary

Week 14 Meeting December 1st - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	No
Nick Wallingford	Back-end Developer	Yes

Week 14 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out week sprint logs	Yes	Logs updated
Discuss any necessary database revisions	Yes	Established new updates for database
Discuss website functionality and features	Yes	Prioritized features to align with timeline
Meeting summary	Yes	Set project goals and agreed on tasks

Week 14 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Started to add some UI aesthetics.	Denise Nguyen	- Continue to work on aesthetics.
- Assisted with aesthetics for user interface	Jason Sneddon	- Create new database tables - Update database repository
- NONE	Jonathan Tapia	
- Managed Github repositories for cleaner version control	Nick Wallingford	- Implement interface functionalities

Week 15 Meeting Summary

Week 15 Meeting December 6th - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	No
Nick Wallingford	Back-end Developer	Yes

Week 15 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Discuss Final report	Yes	- Established work to be completed in report
Discuss Final presentation	Yes	- Established presentation requirements
Review website functionally	Yes	- Finalized priorities for website
Holistic system testing	Yes	- Agreed on a system testing approach

Week 15 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Final report and presentation	Denise Nguyen	CSS, Python, Report, Presentation Test QR code check in and checkout process.
- Updated database & repo - Managed inbound rules for access to AWS	Jason Sneddon	CSS, Python, Report, Presentation
- NONE	Jonathan Tapia	
- Cleaned up project code - Merged branches to master - Implemented QR code scanning	Nick Wallingford	CSS, Python, Report, Presentation

Week 16 Meeting Summary

Week 16 Meeting December 11th - 3:00pm to 4:00pm		
Team Members	**Roles**	**Attended Meeting**
Denise Nguyen	Front-end Developer	Yes
Jason Sneddon	AWS Architect, Database	Yes
Jonathan Tapia	Front-end Developer	Yes
Nick Wallingford	Back-end Developer	Yes

Week 16 Agenda		
Agenda Item	**Completed**	**Summary of Accomplishments**
Fill out sprint log	Yes	Sprint logs and team members updated
Discuss website features to be implemented	Yes	Established remaining project goals to fulfill
Meeting summary	Yes	Identified remaining tasks to complete

Week 16 Work Summary		
Work Completed Since Last Meeting	**Team Member**	**Work to Complete Before Next Meeting**
- Optimized website for mobile - Made website more visually appealing	Denise Nguyen	- Report - Presentation
- Updated database and repository - Improved styling for logout modal & navbar - Cleaned up code & added comments - Final report & presentation	Jason Sneddon	- Final report and presentation
- NONE	Jonathan Tapia	
- Implemented check-in & checkout process - Added “Recent Data” views to Home page - Implemented QR code functionality - Added “Edit Item” functionality - Added a checkout duration feature	Nick Wallingford	- Final presentation

- Improved presentation of data via styling		

CHANGE HISTORY

Report Change History

Date	Member	Changes Done
Project Requirements Report		
October 11	Denise Nguyen	Created initial document with abstract
October 12	Denise Nguyen	Added team info and table of contents
October 13	Jason Sneddon	Modified team contact info and added meeting summaries
October 14	Jason Sneddon	Updated table of contents and inserted weekly agendas
October 15	Jonathan Tapia	Filled out change history and wrote up team structure
October 15	Everyone	Inserted tables and images and minor overall edits
October 15	Jason Sneddon	Completed simplified ER diagram
October 16	Denise Nguyen	UML: Formal Use Cases for Users Partial Analysis: Object Definition Sequence Diagram
October 17	Denise Nguyen	Created final project paper.
Design and Implementation		
October 21	Denise Nguyen	Created initial document for report.
November 17	Jason Sneddon	Added Old Report Content to this report Updated diagrams
November 17	Denise Nguyen	Added Team Roles images Implementation Milestones
November 19	Jonathan Tapia	Edited weekly logs and change history
November 20	Nick Wallingford	Justify lack of class case diagram Document MVC design Added MVC design pattern documentation
Final Report		
November 25	Denise Nguyen	Created initial document for report.
December 6	Denise Nguyen	Test Analysis

December 10	Jason Sneddon	Updated database diagrams - ERD and Schema
December 10	Denise Nguyen	Worked on Test Analysis
December 11	Jason Sneddon	Added Conclusion, Future Work, & Summary Updated Sprint Meeting Logs Added Project Manager Section Created and added Page Flow Diagram Added wireframes