Purdue Electronic Checkout (PEC)

Team Members: Bor Shiuan Chen, Brian Chan, Hongda Zeng, Xu Han, Nick Saggese

Problem Statement

As technology continues to grow, keeping track of electronic devices using a whiteboard or by paper is slow, error prone, and inefficient. The need for an electronic system will reduce these problems and help entities better managing their technology.

Background Information

You have many electronic devices that you need to keep track of. You are tired of having to use a whiteboard or paper to know who has what device. There is also no quick way of checking status other than to be at the actual location. With our Purdue Electronic Checkout system, users will be able to quickly check in/out electronic devices from any place. People can also view a complete log of previous users of equipment, as well as statistics.

Environment

We will be developing our system using an HTML5 client web app front end, node js server and postgresql database. A RESTful api will also handle network communications between the client and server.

Requirements

Functional:

ID	Description	Hours
1	As a user, I want to be able to check out a piece of equipment.	30
2	As a user, I want to be able to check in a piece of equipment.	30

3	As a user, upon check-in, I want to be able to submit a comment as well as an overall rating that dictates the condition of a piece of equipment upon my return of it.	30
4	As a user, I want to see the entire library of equipment.	15
5	As a user, I want to see the library of available equipment.	20
6	As a user, I want to be able to see the details of a particular piece of equipment.	20
7	As a user, I want to have an account where I can see what equipment I have currently checked out and when it is due.	10
8	As an administrator, I want to be able to register a piece of equipment.	10
9	As an administrator, I want to be able to retire a piece of equipment from the library.	10
10	As an administrator, I want to be able to view a specific user's history of equipment usage, including statistical breakdowns of lateness of returns, item condition, and types of equipment used.	10
11	Al	10
12	As a developer, I want the database have a unique index.	5
13	As a developer, I want to be able to manipulate the database through the API.	10

Non-functional:

1. As developers, we want our application to be simple, straightforward, and easy to use with little to no introduction.
2. As developers, we want our application to utilize a clean and professional user interface.
3. As developers, we want our application to run as lightly as possible reducing system resource costs.
4. As developers, we want our application to have no bugs if possible.

5. As users, we want our application to be as fast as possible without sacrificing features of the app.
6. As developers, we must be able to accommodate as many users as possible.
7. As developers, we want to store as little data on the client device as possible.
8. As developers, we want our database system to be scalable.
9. As a developer, I want to minimize click depth.
10. As a user, I want to always be able to access the app.
11. As a developer, I want to reduce server downtime as much as possible.
12. As a developer, I want the API to be as simple as possible.
13. As a developer, I want the database to be as compact as possible.
14. As a developer, I want there to be as few database as possible.

Use Cases

Case	As a user, I want to be able to check IN a piece of equipment.
Action	**System Response**
1. Click status combobox	
	2. Combobox drop down appears with In/Out options
3. Select “Check In” status	
4. Click apply button	
	5.Confirmation dialog appears
6. Click OK/Cancel	
	7. Dialog disappears
	8. Page and logs are updated

Case	As a user, I want to be able to check OUT a piece of equipment.
Action	**System Response**
1. Click status combobox	
	2. Combobox drop down appears with In/Out options
3. Select Out status	
4. Click user combobox	
	5. Combobox drop down appears with available users options
6. Select user	
7. Click apply button	
	8.Confirmation dialog appears
9. Click OK/Cancel	
	10. Dialog disappears
	11. Page and logs are updated

Case	As a user, upon check-in, I want to be able to submit a comment as well as an overall rating that dictates the condition of a piece of equipment upon my return of it.
Action	**System Response**
1. [Check In Case Completed]	1.Condition Entry Page is displayed.

2. User selects condition rating 1-10 using available slider.	
User adds any comments in text box.	
3.User selects submit button	2. Confirmation box is displayed.
4. User clicks OK or CANCEL	3. Dialog Disappears
	4. System/Logs Updated.
	5. User redirected to main app page.

Case	As a user, I want to see the entire library of equipment.
Action	**System Response**
1. User authenticates by entering username and password.	
2. Click LOGIN button.	1. System redirects to main application page if successful.
	2. If login incorrect, message is displayed to user on login screen and they have the opportunity to try again.
3. Table view of available devices is shown by default. Click filter button to see entire library of equipment.	3. System refreshes page to show the entire library of equipment.

Case	As a user, I want to see the library of available equipment.
Action	**System Response**

1. User authenticates by entering username and password.	
2. Click LOGIN button.	2. System redirects to main application page if successful.
3. Table view of available devices is shown by default.	

Case	As a user, I want to be able to see the details of a particular piece of equipment.
Action	**System Response**
1. Click a devices icon from the table view of devices.	1. System directs to that devices individual page.

Case	As a user, I want to have an account where I can see what equipment I have currently checked out and when it is due.
Action	**System Response**
1. [User is already authenticated]	
2. Click on profile button.	2. System redirects to profile page.

Case	As an administrator, I want to be able to register a piece of equipment.

Action	**System Response**
1. Administrator authenticates by entering username and password.	
2. Clicks on Add new devices	3. Add new devices appears (Alternative: Page change)
4. Administrator enters device details (Device name, UID, description)	
5. Administrator selects OK or Cancel	6. System adds device to database and log
	7. Dialog closes (Alternative: Page clears and Green confirmation text)

Case	As an administrator, I want to be able to retire a piece of equipment from the library.
Action	**System Response**
1. Administrator authenticates by entering username and password.	
2. Clicks on List of Devices	3. Page changes to List of Devices
4. Clicks on Manage	
5. Clicks delete button (little X)	6. A dialog appear with input for reason of retirement
7. Input Reasons	
8. Selects OK	9. System marks device as retired
	10. Confirmation

Case	As an administrator, I want to be able to reserve a specific piece of equipment for a specific user on a date range.
Action	**System Response**
1. Administrator authenticates by entering username and password.	
2. Clicks on List of Devices	3. Page changes to List of Device
4. Clicks on a device	5. Click on Reserve this Device
	6. Dialog appears
7. Selects date, Selects user, Input optional comment	
8. Administrator clicks OK	9. Dialog disappears
	10. Confirmation

Case	As an administrator, I want to be able to view a specific user’s history of equipment usage, including statistical breakdowns of lateness of returns, item condition, and types of equipment used.
Action	**System Response**
1. Administrator authenticates by entering username and password.	
2. Clicks on List of Users	3. Page change to List of Users
4. Selects a User	4. Dialog appears with User Information
5. User selects OK	6. Dialog closes

Case	As a developer, I want to be able to access the Postgres database through nodejs server with a RESTful API.
Action	**System Response**
1. Website calls API	2. Server process website message
	3. Server response
4. Website receives response	

Case	As a developer, I want to be able to manipulate the database through the API.
Action	**System Response**
1. Developer calls API	2. Server sends query to database
	3. Database responds.

Case	As a developer, I want the database have a unique index.
Action	**System Response**
1. User reference a specific item ID	2. Server query database with specific item ID
	3. Database correctly identify equipment in question