

Final Report & Documentation

1. Summary -

The project focuses on addressing the needs of CSCE capstone professors who manage inventories of high-value items used in student projects. The application simplifies inventory management by providing a secure, user-friendly platform developed with Ruby on Rails. Core features include maintaining item details, tracking checkouts and returns, and enabling professors or facilities managers to manage and update inventory effectively. With SSO authentication, it ensures secure access for professors, TAs, and authorized personnel. User roles and authorizations further enhance the app's usability while ensuring proper handling of items. Key stakeholders include professors, facilities managers, and students.

Through four sprints, the team implemented foundational features such as user authentication, inventory tracking, item history logging, advanced search filters, and dynamic status updates. Notable enhancements include responsive design, note-adding functionality, and procedures for handling damaged or lost items. The project also resolved critical bugs, such as redundant fields and navigation issues, while adding stretch features like role-based permissions and item replacement alerts. Future iterations aim to optimize efficiency, support student requests, and provide better oversight for professors to track overdue items or wear-and-tear records. The application seamlessly integrates the professors' needs with a robust backend and intuitive UI.

2. Description of *all* user stories -

Sprint 1

User Story:

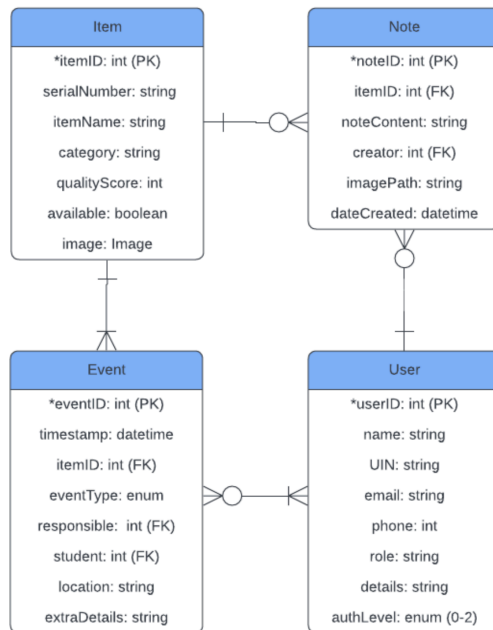
As a professor, I want to store a large list of fields for each item, including details, history, responsible persons, etc.

Points: 3

Implementation Status: Completed

- The database was designed and turned into an official UML diagram, supporting item details, history tracking, and related fields.
- **Changes:** No major changes occurred during this task.

Lo-Fi Mockup:



User Story:

As a user, I want to ensure no bugs occur while using the system.

Points: 5

Implementation Status: Completed

- Initial database and authentication tests using Cucumber validated critical fields and ensured smooth operations.
- **Changes:** None.

Images: N/A

User Story:

As a professor, I want a hosted application to manage Capstone inventory.

Points: 8

Implementation Status: Completed

- A Rails app was created, database migrations were added, and the application was deployed on Heroku.
- **Changes:** None.

Images: N/A

User Story:

As a new user, I want to create an account and log in using Google SSO.

Points: 7 (Backend), 5 (Frontend)

Implementation Status: Completed

- Google SSO was integrated for secure login/signup.
- **Changes:** The welcome path was removed, and a login button replaced the link for better UI.

Lo-Fi Mockup:



Final Screenshot:

WELCOME TO STOCK'EM

Your One Stop Solution to All Your Capstone Needs

Login with Google

Login as Admin

User Story:

As a facilities manager, I want to know all information about an item, including location, contact, and history.


Points: 5

Implementation Status: Completed

- The item information page was created, displaying fields like history and location.
- **Changes:** Improved notes formatting during implementation.

Lo-Fi Mockup:

Stock'Em

Hello, (username) 

Details About ITEM_NAME

Item Name: ITEM_NAME
Serial Number: SERIAL_NUMBER
Category: CATEGORY
Quality Score: SCORE (Good Quality)
Available: YES/NO

image

Go Back

Checkout

Item History

Relocation on 09/22/2021
Given to Professor: [Pauline Wade](#)
New Approximate Location: Peterson Storage

Checkout on 08/27/2021
Returned by Mark Sturtevant
Given to Professor: [Pauline Wade](#)
New Approximate Location: Wade's Attic
Details: Item Sustained slight damage

Checkout on 04/23/2021
Given to Mark Sturtevant
Class: CSCE 606-600
Assigned Professor: [Philip Ritchey](#)
Details: N/A

Notes

06/28/2024: Scratches on side of device, small but there


09/21/2024: Home button seems to not work very well

09/28/2024: Item was Lost :(

The Stock'Em Team, 2024

Final Screenshot:

Stock'Em

Hello, Alisha 

IPHONE8

Item Details

Item ID: 1
Serial Number: FKWK534
Item Name: iphone8
Category: Mobile Devices
Quality Score: 89
Status: Not Available
Details: IPHONE 8 128GB

Checkout

Edit

Back

Delete

Update:

Select Status:

Not Available

Update Status

Ishaan Rawal (2024-11-02):
Status Updated to Damaged

Ishaan Rawal (2024-11-02):
Status Updated to Damaged

Ishaan Rawal (2024-11-02):
Status Updated to Damaged

Ishaan Rawal (2024-11-02):
Status Updated to Damaged

Ishaan Rawal (2024-11-02):
Status Updated to Damaged

Notes

Create Note

The Stock'Em Team, 2024

User Story:

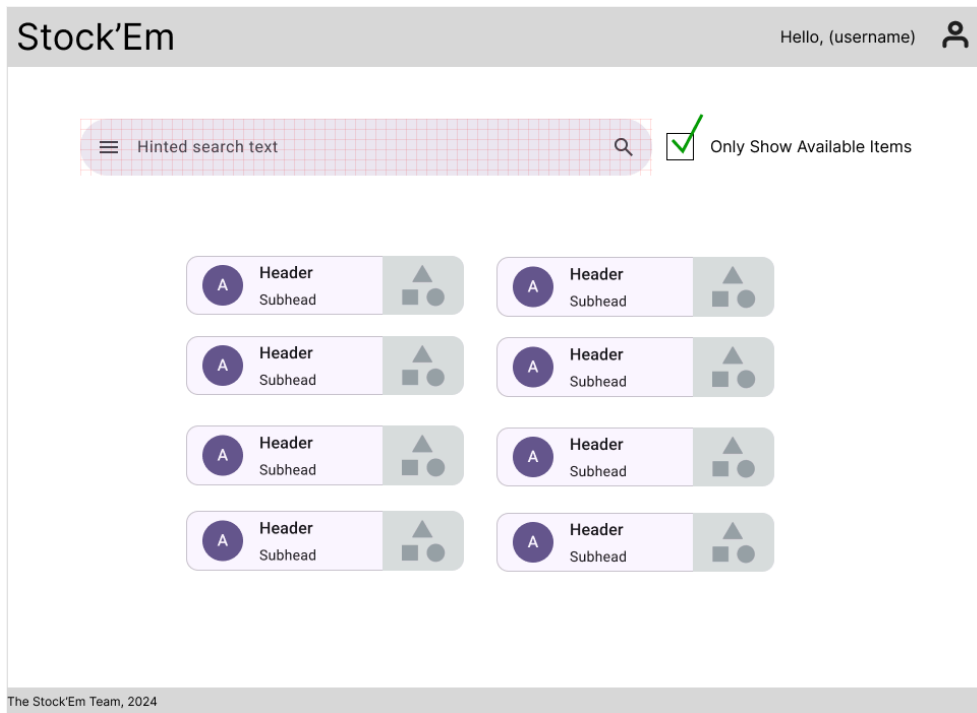
As a professor, I want to see all available items for checkout at once.

Points: 7

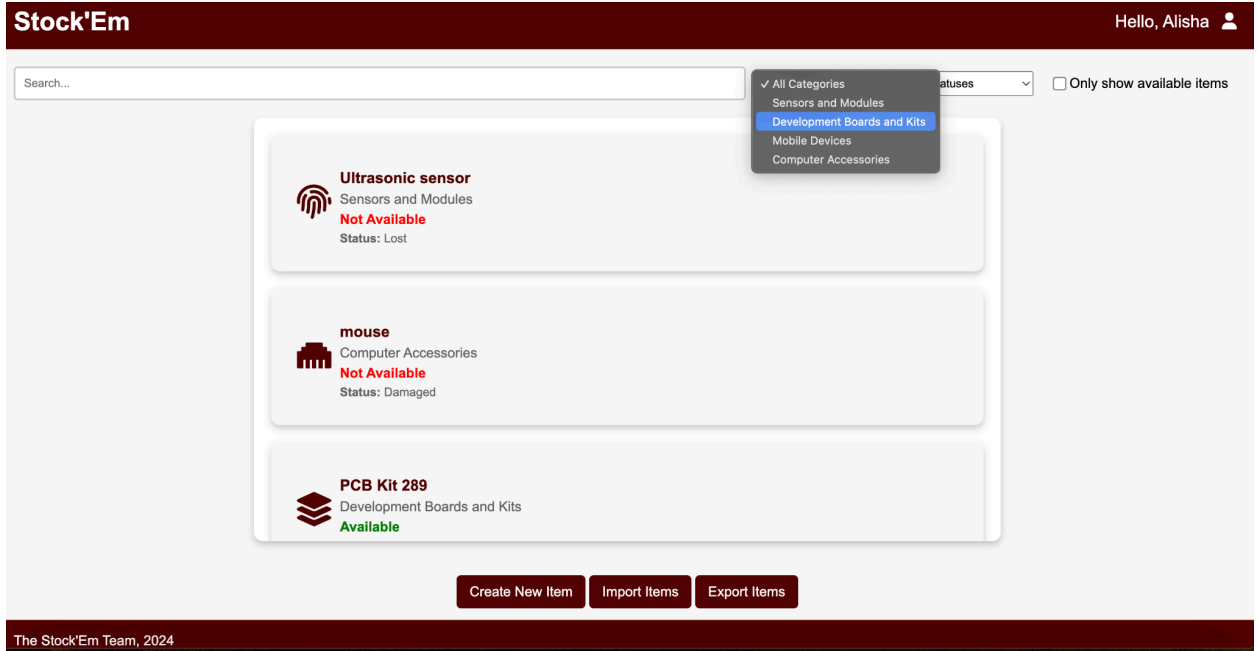
Implementation Status: Completed

- A home page displaying all items with filtering options was implemented.
- **Changes:** Filters were added based on feedback for usability.

Lo-Fi Mockup:



Final Screenshot:



Sprint 2

User Story:

As a user, I want to create new items from the home page.

Points: 5

Implementation Status: Completed

- Users can add items with required fields via a streamlined interface.
- **Changes:** Validation and error handling were improved during implementation.

Final Screenshot:

Stock'Em

Hello, Alisha

Create New Item

Item Name *

Category *

Quality Score *

Currently Available *

Serial Number *

Details

Create Item

User Story:

As a user, I want to add notes to new or existing items in the item details page.

Points: 5

Implementation Status: Completed

- Notes functionality was added, allowing users to input and save notes.
- **Changes:** Notes were made scrollable for better usability.

Final Screenshot:

IPHONE8

Item History

Neil Gautam (2024-11-25):

Status Updated to Intact

Neil Gautam (2024-11-25):

Checked in on 2024-11-25 22:34:49 to Prof. Tony with the new location being EABA. HEHE

Alisha Raj (2024-11-25):

Item attributes have been updated.

Alisha Raj (2024-11-25):

Status Updated to Lost

Notes

Publish

Neil Gautam (2024-11-25):

HOLA AMIGO

Neil Gautam (2024-11-25):

HOLA AMIGO

User Story:

As a user, I want to view added notes on the item details page.

Points: 3

Implementation Status: Completed

- Notes were displayed in a well-formatted manner in the item details section.
- **Changes:** Formatting was updated based on team feedback.

Final Screenshot: (Placeholder for Image)

IPHONE8

Item History

Neil Gautam (2024-11-25):

Status Updated to Intact

Neil Gautam (2024-11-25):

Checked in on 2024-11-25 22:34:49 to Prof. Tony with the new location being EABA. HEHE

Alisha Raj (2024-11-25):

Item attributes have been updated.

Alisha Raj (2024-11-25):

Status Updated to Lost

Notes

Publish

Neil Gautam (2024-11-25):

HOLA AMIGO

Neil Gautam (2024-11-25):

HOLA AMIGO

User Story:

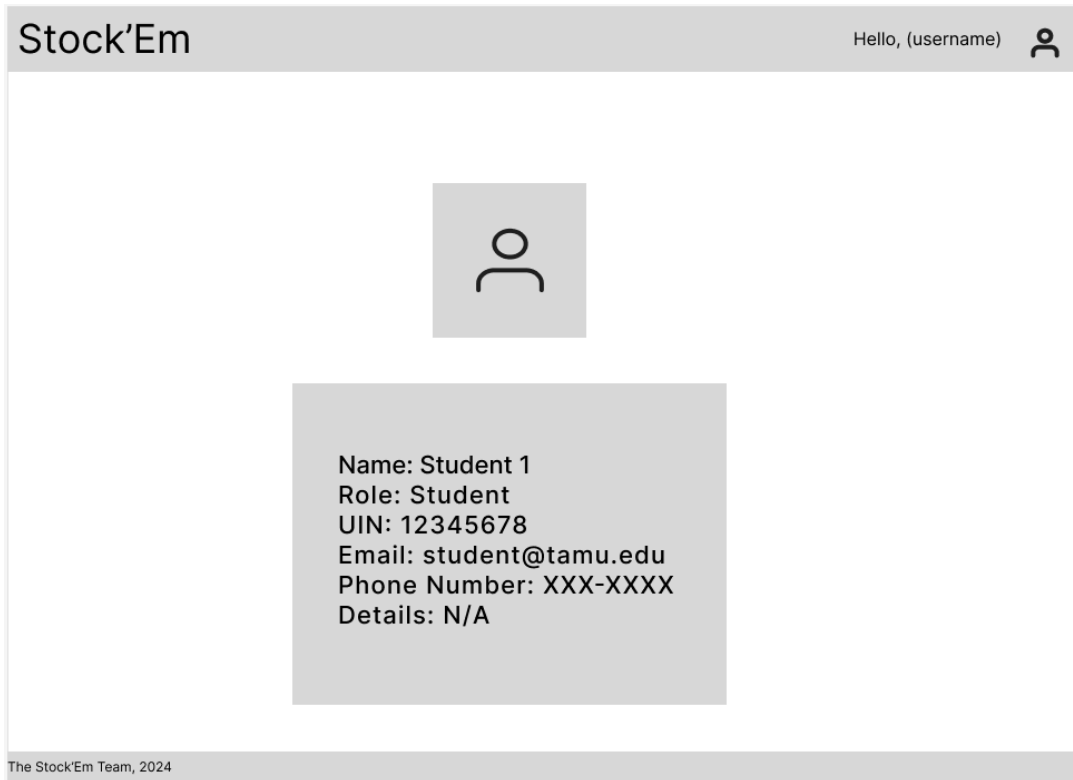
As a user, I want to add or edit details to my profile.

Points: 3

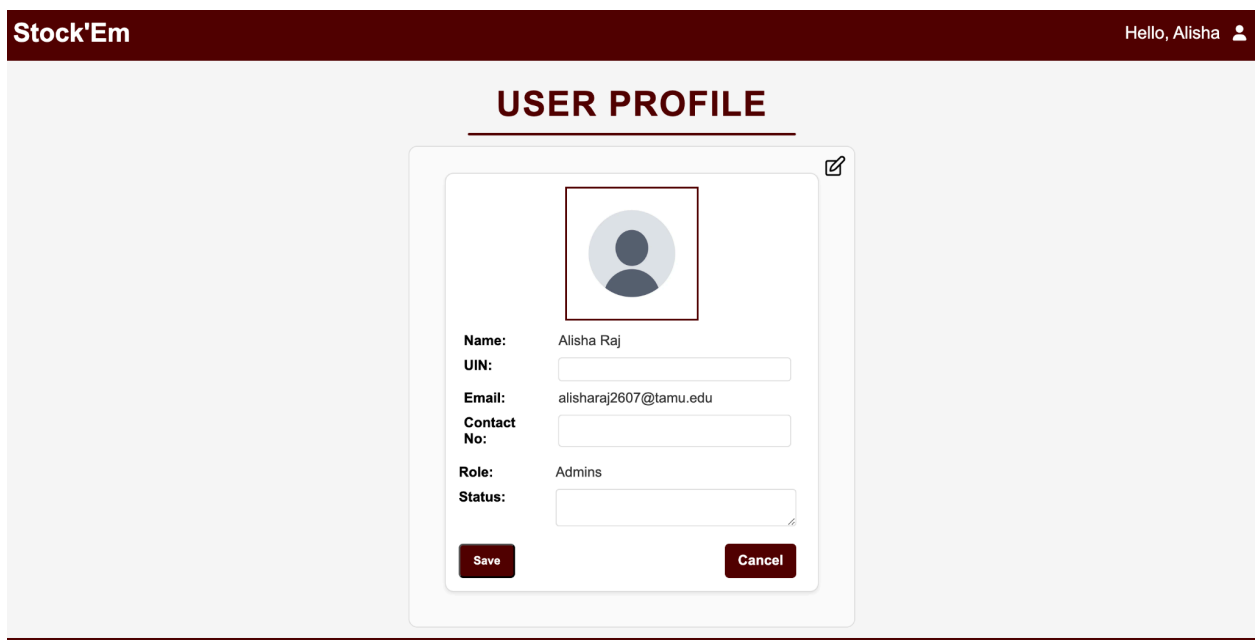
Implementation Status: Completed

- Profile functionality was implemented, supporting updates like contact and location information.
- **Changes:** Validation for illegal fields was added during testing.

Lo-Fi Mockup:



Final Screenshot:



User Story:

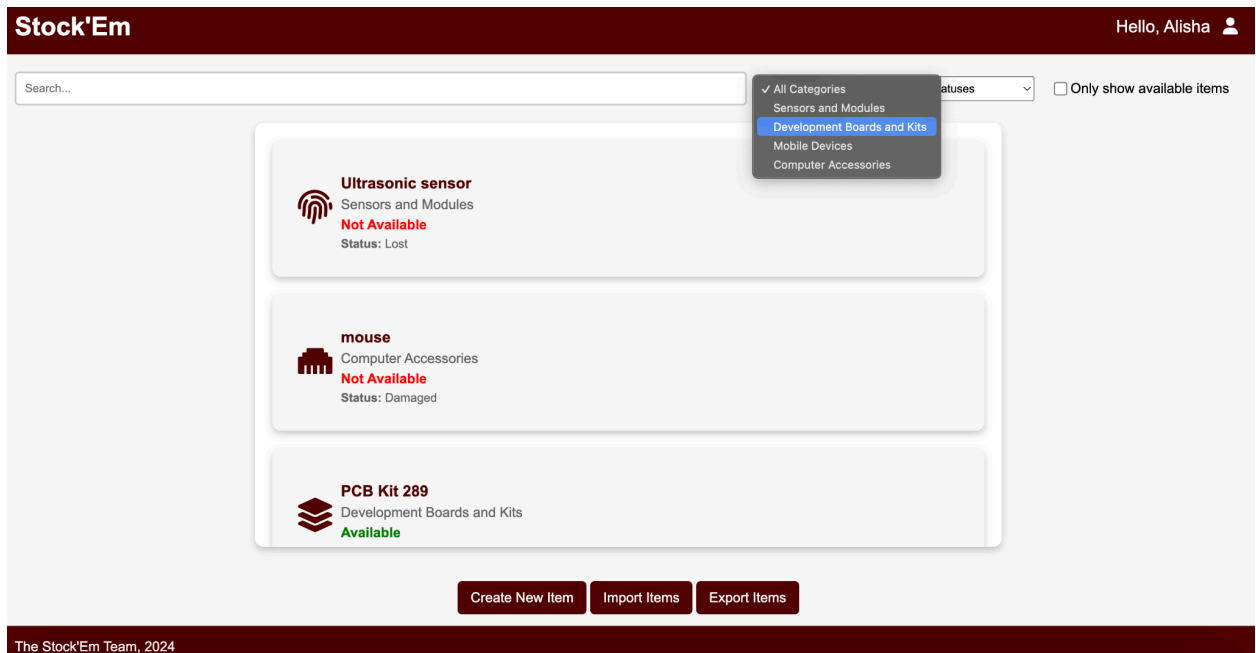
As a user, I want to use a search filter on the items list page.

Points: 5

Implementation Status: Completed

- A search bar allowed users to filter items dynamically.
- **Changes:** Dropdown filters were updated during Sprint 3 for better usability.

Final Screenshot:



User Story:

As a user, I want to change attributes (e.g., damaged/lost) of a particular item.

Points: 3

Implementation Status: Completed

- Users could modify item status directly from the item details page.
- **Changes:** Validation was added to prevent redundant updates.

Final Screenshot:

The screenshot shows a web form titled "Item Details" with the following fields: Item ID: 1, Serial Number: FKWK534, Item Name: iphone8, Category: Mobile Devices, Quality Score: 89, Status: Intact, and Details: IPHONE 8 128GB. Below these fields are four buttons: Checkout, Edit, Back, and Delete. At the bottom, there is an "Update:" section with a "Select Status:" dropdown menu. The dropdown menu is open, showing options: Clear Status, Damaged, Lost, Not Available, and Intact (which is selected with a checkmark). To the right of the dropdown is an "Update Status" button.

Item Details

Item ID: 1

Serial Number: FKWK534

Item Name: iphone8

Category: Mobile Devices

Quality Score: 89

Status: Intact

Details: IPHONE 8 128GB

Checkout Edit Back Delete

Update:

Select Status: Clear Status
Damaged
Lost
Not Available
✓ Intact Update Status

User Story:

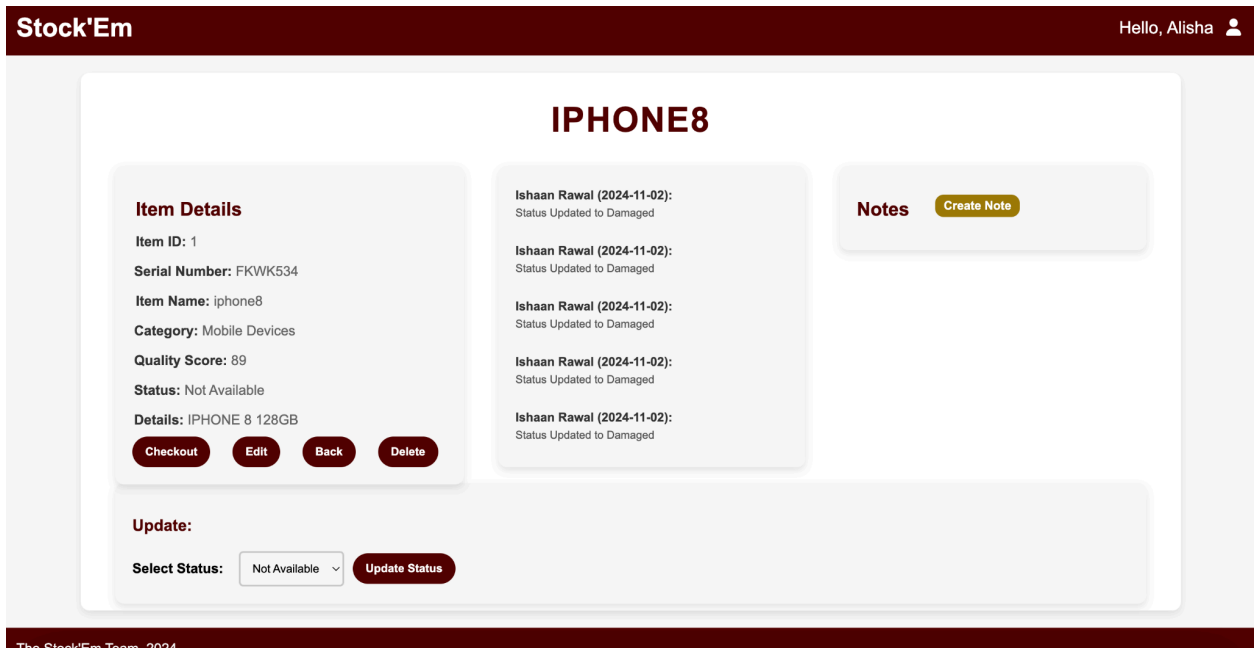
As a user, I want clear, formatted information pages describing items.

Points: 5

Implementation Status: Completed

- Item details pages were enhanced with formatted and concise information.
- **Changes:** Improvements were based on user feedback.

Final Screenshot:



Sprint 3

User Story:

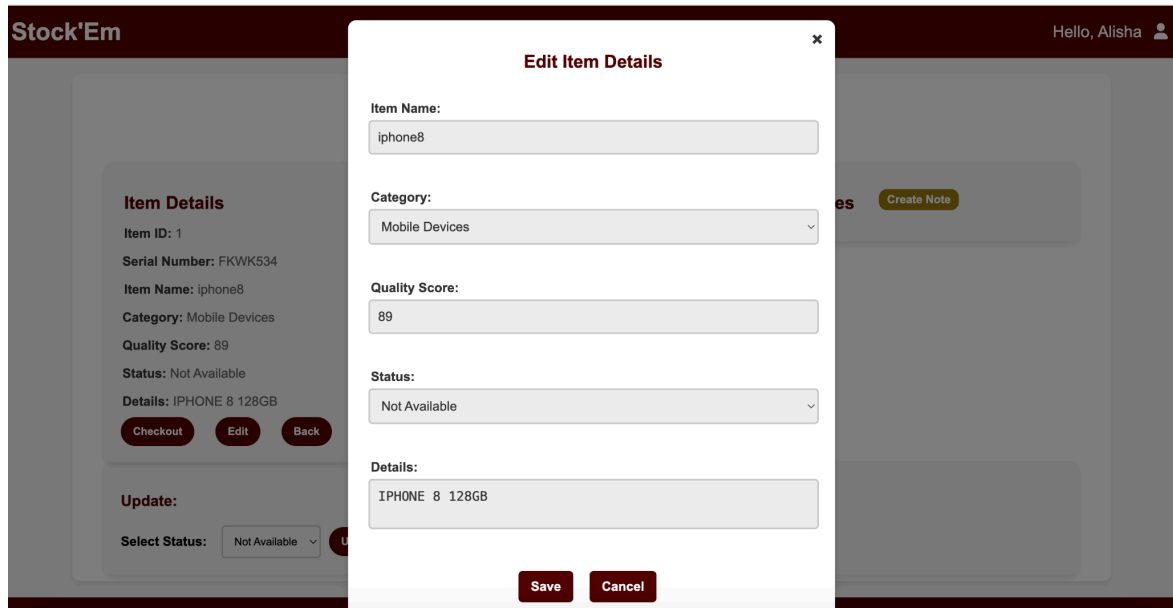
As a user, I want to edit, delete, or relocate individual items.

Points: 5

Implementation Status: Completed

- CRUD operations were implemented with proper validations.
- **Changes:** Inline editing was replaced with popups for clarity in Sprint 4.

Final Screenshot:



User Story:

As a user, I want updates and edits to items tagged in the events table.

Points: 3

Implementation Status: Completed

- Events were logged automatically during any updates or changes.
- **Changes:** None.

Final Screenshot:

Item History

Neil Gautam (2024-11-25):

Status Updated to Intact

Neil Gautam (2024-11-25):

Checked in on 2024-11-25 22:34:49 to Prof. Tony with the new location being EABA. HEHE

Alisha Raj (2024-11-25):

Item attributes have been updated.

Alisha Raj (2024-11-25):

Status Updated to Lost

User Story:

As a user, I want to check in or check out items and view history.


Points: 5

Implementation Status: Completed

- Check-in and check-out buttons were added, with corresponding history updates.
- **Changes:** UI formatting for buttons was improved.

Final Screenshot:

Stock'Em

Hello, Alisha 

Checkout Item

Responsible Professor *

Student / Teams Checked Out To

Class & Section, or Approximate Location *

Additional Comments

Publish

The Stock'Em Team, 2024

User Story:

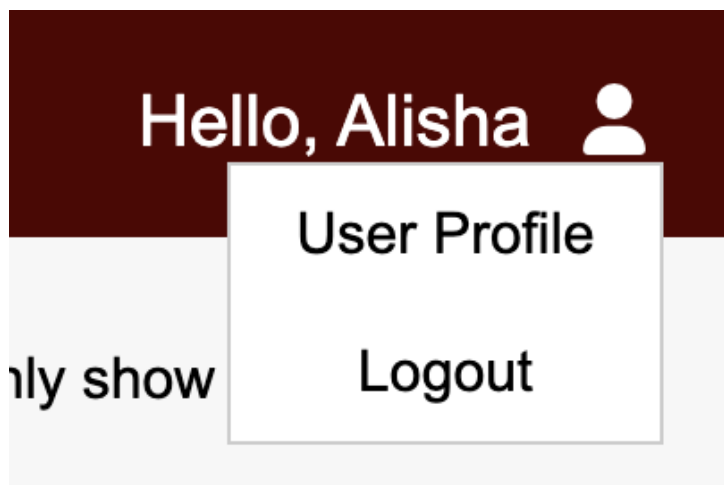
As a user, I want navigation to be intuitive and responsive.

Points: 5

Implementation Status: Completed

- Navigation issues were resolved, and pages were made fully responsive.
- **Changes:** Dropdown navigation was added for better usability.

Final Screenshot:



Sprint 4

User Story:

As an admin, I want to restrict access to editing items only to admins.

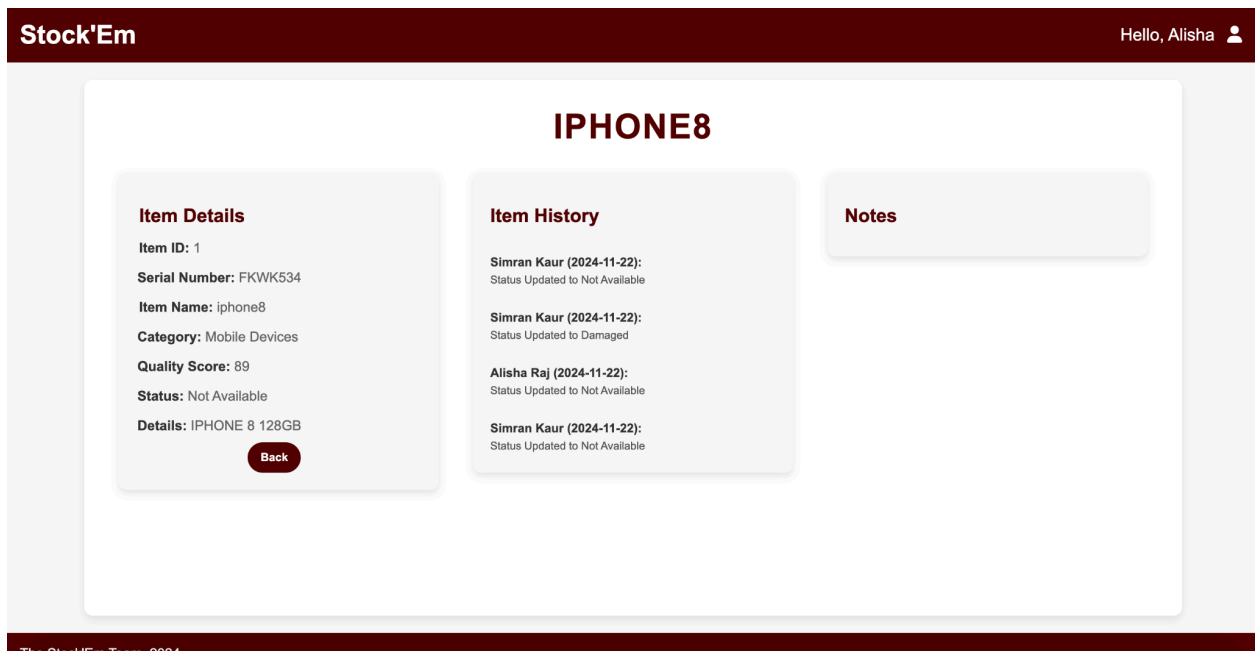
Points: 3

Implementation Status: Completed

- The edit functionality was restricted to users with admin privileges (`auth_level == 2`).
- **Changes:** None.

Lo-Fi Mockup: N/A

Final Screenshot:



User Story:

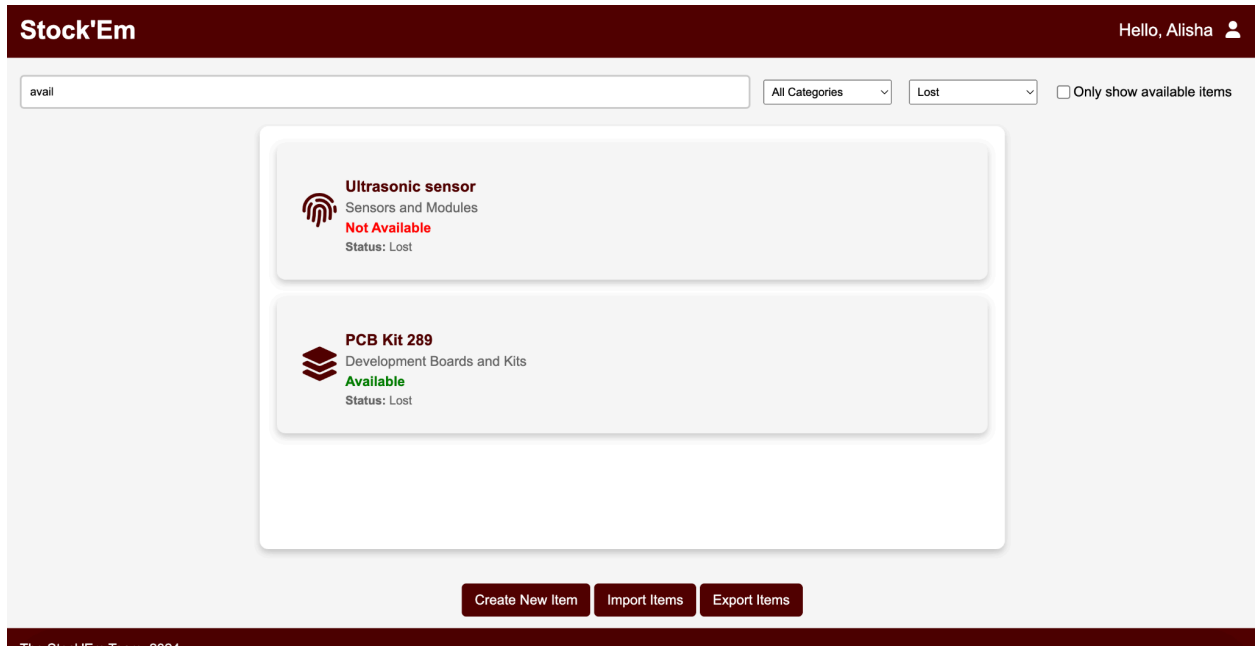
As a user, I want dynamic filters for categories and statuses on the items list page.

Points: 5

Implementation Status: Completed

- The filter dropdowns now dynamically load all available options, including an "Unknown" status.
- JavaScript auto-submits the form upon selection, replacing the earlier "Filter" button.
- **Changes:** Adjusted tests to handle the new functionality.

Final Screenshot:



User Story:

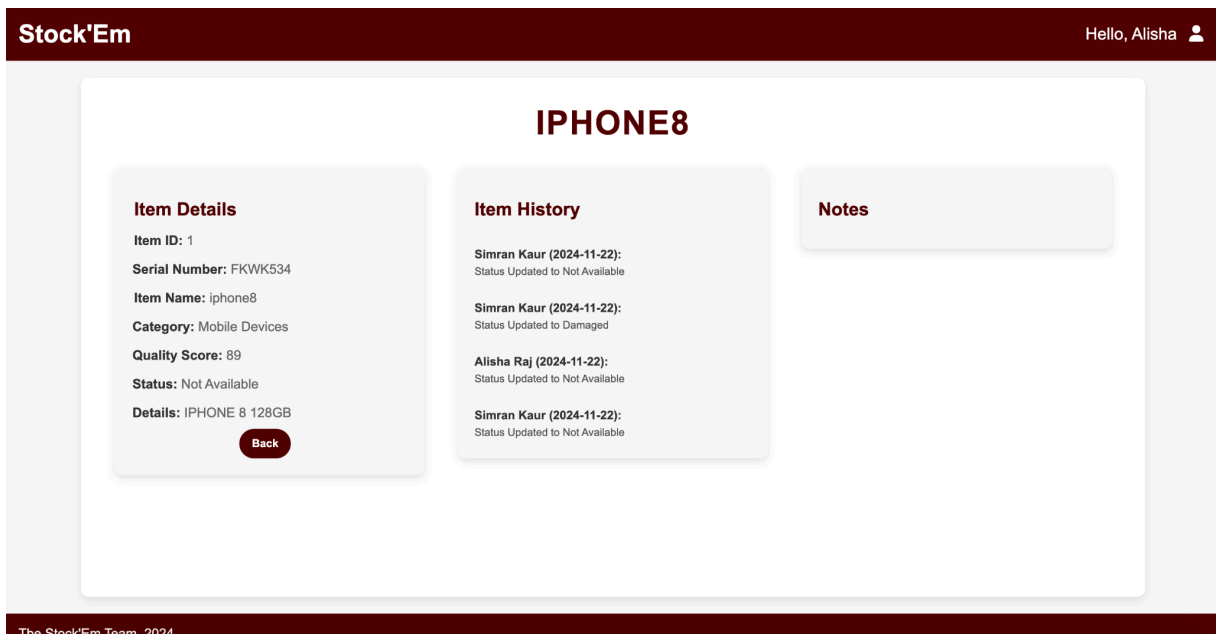
As a user, I want to add notes to the notes section, with access limited to certain roles.

Points: 5

Implementation Status: Completed

- Notes functionality allows `auth_level == 1` (assistant) or `auth_level == 2` (admin) users to add notes to items.
- **Changes:** Adjusted authorization to match user roles accurately.

Final Screenshot: "create note" option not present for students



User Story:

As a developer, I want clear error messages when updating item statuses to streamline debugging.

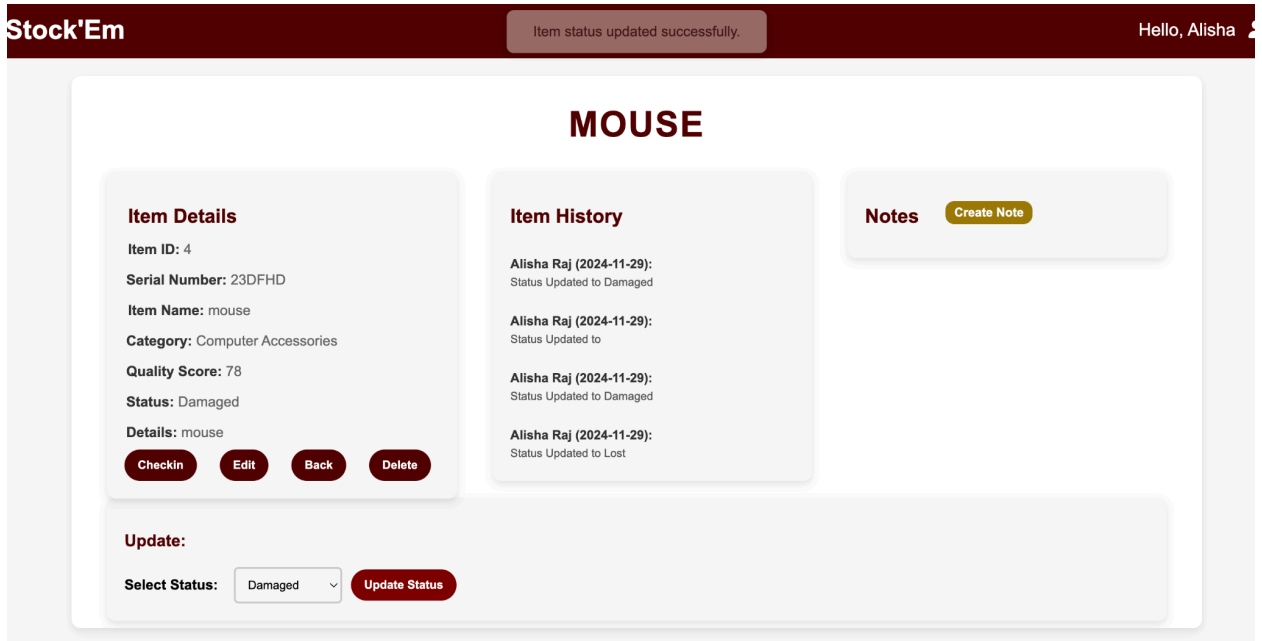
Points: 3

Implementation Status: Completed

- Flash messages now differentiate between successful status updates and validation errors.
- **Changes:** Updated JavaScript to handle status-specific error displays dynamically.

Lo-Fi Mockup: N/A

Final Screenshot: “pop up” will show if the update is successful or not



User Story:

As a user, I want to delete an item and confirm the deletion with a clear message.

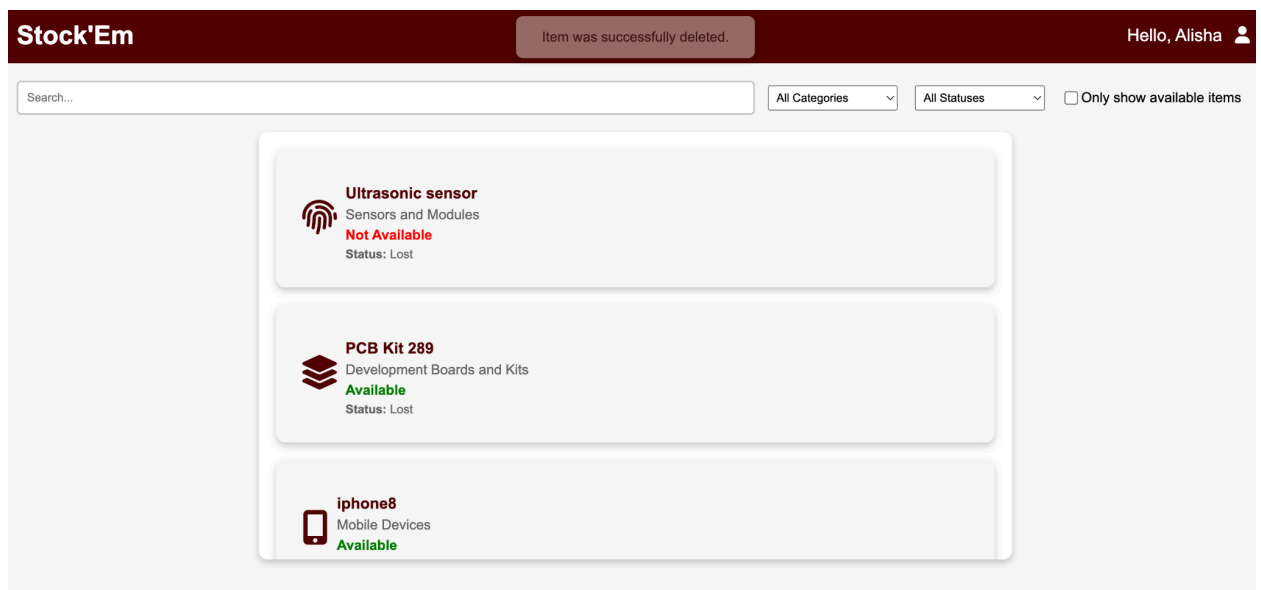
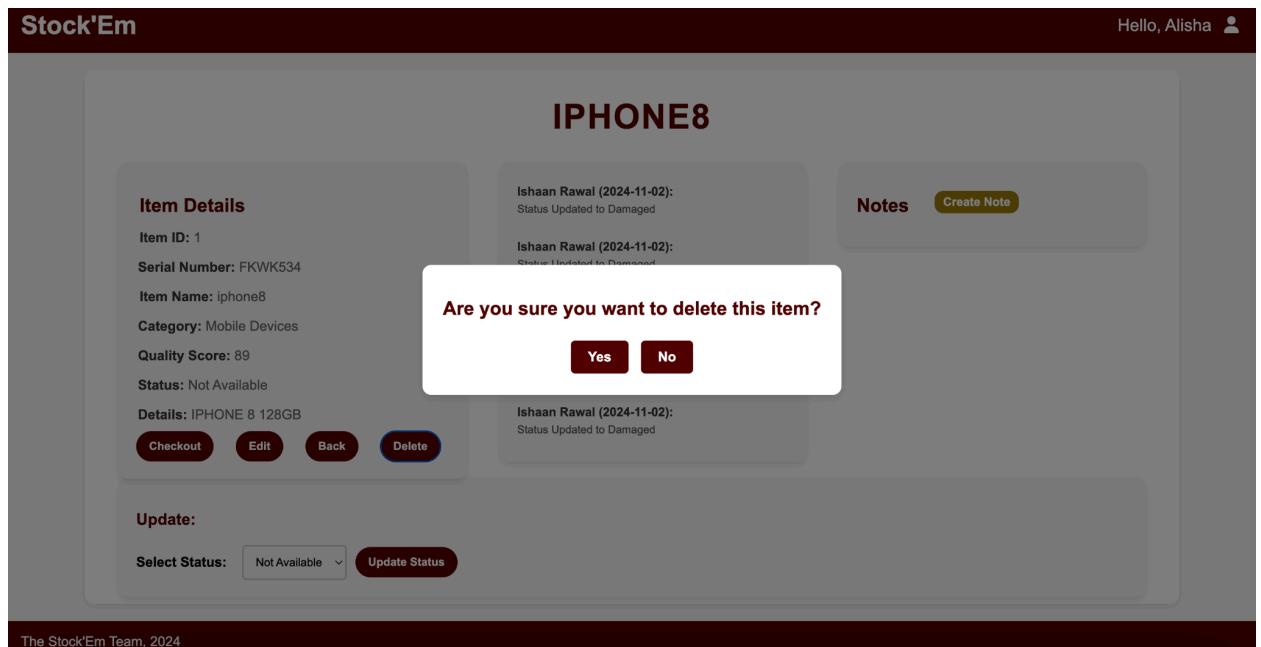
Points: 5

Implementation Status: Completed

- The delete functionality was restricted to admins, with flash messages confirming success or failure.
- **Changes:** Added confirmation dialogs for user clarity.

Lo-Fi Mockup: N/A

Final Screenshot:



User Story:

As a developer, I want testing coverage to exceed 90% to ensure code quality.

Points: 8

Implementation Status: Completed

- Code coverage tools like **SimpleCov** were integrated, achieving over 90% coverage.
- **Changes:** Additional edge cases were covered after initial testing.

Final Screenshot:

```
62 scenarios (62 passed)
541 steps (541 passed)
0m16.915s

View your Cucumber Report at:
https://reports.cucumber.io/reports/9327cf6e-a7f4-4244-9971-2d74fa4d5779

This report was published in collection "stockemCucumberReports"

Coverage report generated for Cucumber Features to /Users/alisharaj/Desktop/Stock-Em/coverage.
Line Coverage: 90.8% (306 / 337)
```

```
.....

Finished in 0.37285 seconds (files took 1.28 seconds to load)
88 examples, 0 failures

Coverage report generated for Cucumber Features, RSpec to /Users/alisharaj/Desktop/Stock-Em/coverage.
Line Coverage: 96.76% (328 / 339)
```

User Story:

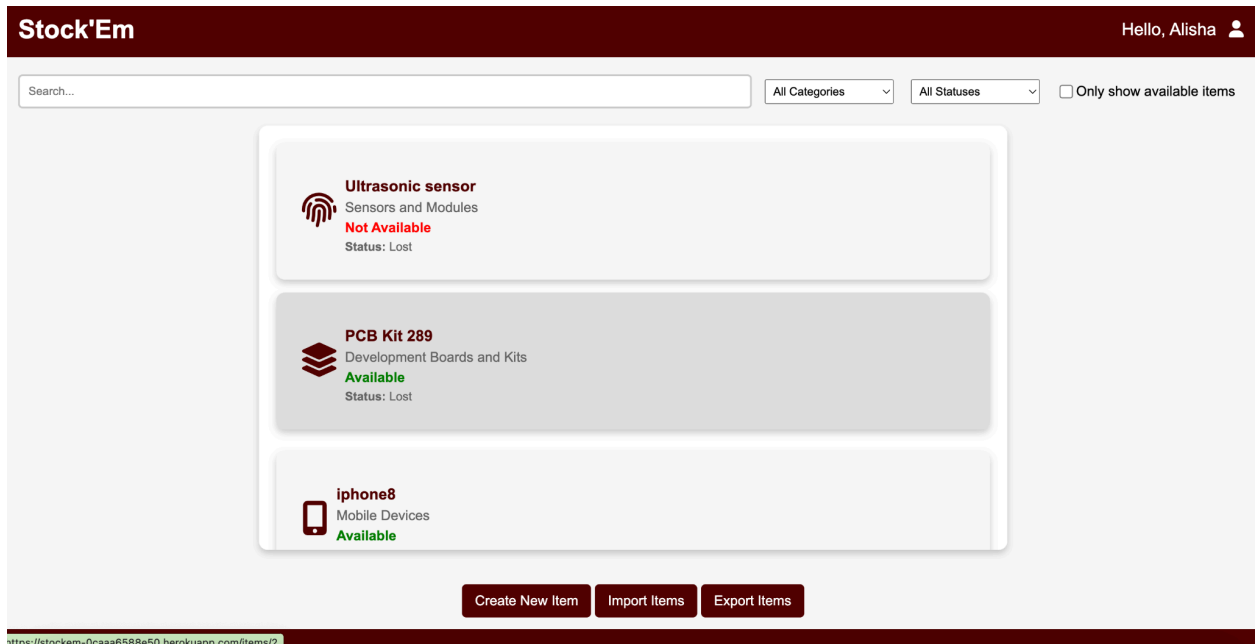
As a user, I want responsive and styled UI for all pages without using CSS libraries.

Points: 5

Implementation Status: Completed

- All views (index, new, show, edit) received additional styling for responsiveness and better structure.
- **Changes:** Adjusted to include hover effects for interactive elements.

Final Screenshot:



3. List of Roles -

Sprint 1: Mark Sturtevant (Scrum Master), Ishaan Rawal (Product Owner)
Sprint 2: Tanishq Chopra (Scrum Master), Alisha Raj (Product Owner)
Sprint 3: Simran Jot Kaur (Scrum Master), Vinay Singh (Product Owner)
Sprint 4: Simran Kaur (Scrum Master), Mark Sturtevant (Product Owner)

4. Summary of each iteration -

Sprint 1:

Users are able to log in using their TAMU Google accounts and navigate to their user profiles or view the items currently seeded in the database. We successfully completed all planned stories on time and developed and deployed the application, ensuring that all documentation, code quality, testing, and coverage met the required standards.

Points completed in sprint 1- **57**

Sprint 2:

Item Creation: Successfully implemented the feature allowing users to create new items directly from the home page, enhancing user accessibility and workflow.

Note Management: Enabled users to add notes to both new and existing items on the item details page, facilitating better organization and tracking of information.

Notes Visibility: Developed functionality for users to view added notes on the item details page, improving transparency and user experience.

Profile Updates: Implemented a feature allowing users to add and edit their profile details, promoting personalized user engagement.

Search Functionality: Integrated a search filter on the items list page, enabling users to quickly find specific items based on criteria.

Attribute Modification: Created the ability for users to change item attributes (damaged/lost), allowing for better inventory management.

Formatted Item Information Page: Launched a clear and well-structured information page for items, providing users with comprehensive details at a glance.

Points completed in sprint 2- **39**

Sprint 3:

[Bug Fix] Remove the redundant column from the users table.

[Bug Fix] Combine the Status and Availability fields into a single, unified field.

[Bug Fix] Resolve the issue with the search filter functionality.

[UI] Ensure all pages are responsive and address any navigation-related issues.

[API/UI] Enable users to edit, delete, or relocate individual items.

[API/UI] Record any updates or edits made to an item as part of the item's history in the events table.

[UI] Correct all CSS issues and optimize styling by creating a centralized CSS framework for the entire system.

[API/UI] Develop a more advanced search feature to improve item retrieval.

[API/UI] Allow users to check in and check out items, with access to each item's history.

[API/UI] Establish clear authorization rules and procedures within the system to ensure secure access.

Points completed in sprint 3- **44**

Sprint 4:

Authentication Enforcement: Certain privileged operations (such as editing or checking out items) can now only be performed by users who have the relevant credentials to do so.

UI Support for Authentication: For privileged operations a user cannot perform, any buttons to do so are automatically hidden from the user to avoid any confusion.

Improved Item Page Formatting: The “View Item” page now has separate sections for notes and events, both of which also receive rendering improvements.

Improved Edit Item Formatting: Now, the edit item page creates a pop-up with custom formatting.

Bulk Import/Export: Along with creating items, users can also pass in CSVs to add many items at once. Users may also export the current items database into a CSV to boot.

Random QOL Features: Many small, user-facing features such as warning popups for delete operations, notifications when all items are filtered, and more.

More Consistent Flash Messages: Normalization across all web pages to make the formatting more consistent.

Points completed in Sprint 4 - **39**

5. Table of stories points -

Member Name	Total points solved by individual	Total Stories Completed by individual
Tanishq Chopra	25.5	7
Simran Kaur	30	6
Simran Jot Kaur	31.5	6
Vinay Singh	29.5	6
Ishaan Rawal	22	4
Alisha Raj	26	5
Mark Sturtevant	25	6

6. Customer Meeting details -

Sprint 1 Meet: : Oct 2, 2024, 01:30 PM Central Time (US and Canada)

Summary: Paul was satisfied with the first iteration of the project. For our meeting, we went over our designs/progress for the Frontend UI, Database design / organization, and showed off our current progress on login authentication. Paul gave the green flag on our work so far, suggested one new feature i.e. (Exhaustive Search Items) for the next sprint.

Sprint 2 Meet: : Oct 16, 2024, 01:30 PM Central Time (US and Canada)

Summary: For our meeting, we went over our new additions in the second sprint including the changes he proposed in the previous meeting. We presented our new UI and backend changes in creating a new Item page, managing notes for the new and existing items, user profile page and the edit function on the same, and the functionality to change the status of the item. Paul was happy with the new changes and was looking forward to the next sprint.

Sprint 3 Meet: : Oct 30, 2024, 01:30 PM Central Time (US and Canada)

Summary: In our recent meeting, the customer reviewed the latest updates to the system and expressed satisfaction with the improvements made. Key

enhancements included the merging of redundant fields, the resolution of search filter issues, and the implementation of a responsive design that optimizes navigation across pages. The customer appreciated the new item management capabilities, such as editing, deleting, relocating items, and viewing item history. The advanced search functionality and centralized CSS were also well-received, and the customer noted that the authorization rules added valuable security to the system. In terms of additional features, the customer suggested introducing a mass item creation feature to streamline bulk entries and requested a dropdown menu to simplify updating the location of items. These suggestions will be considered in upcoming development cycles to further improve the user experience.

Sprint 4 Meet: : Nov 13, 2024, 01:30 PM Central Time (US and Canada)
Summary: After seeing the final sprint MVP, Paul was very pleased with the results and noted that we achieved everything that he desired for the application. The only caveat was one small request to change a specific exported file format; a quick change that was merged in the final sprint. Overall, Paul seems to view the project as a success and is looking forward to using this for organizing the inventory in Capstone.

7. Explain your BDD/TDD process, and any benefits/problems from it -

BDD/TDD Process for Stock-Em:

Behavior-Driven Development (BDD):

1. Feature Files:

- For every user story, a feature file was created using Gherkin syntax, describing the behavior in plain English.
- Scenarios captured user interactions, focusing on end-user experience (e.g., filtering items, adding notes, or handling privileges).

2. Cucumber Framework:

- Steps were implemented to automate the scenarios described in the feature files.
- JavaScript-enabled tests, such as auto-submitting filters or pop-up confirmations, were run using Mozilla Firefox.

3. Iteration:

- Feature files evolved based on feedback or changing requirements during development.

Test-Driven Development (TDD):

1. Unit Tests:

- RSpec was used to write unit tests for models, controllers, and services.
- Tests were created before implementing functionalities, focusing on individual methods (e.g., privilege checks, dropdown filters).

2. Integration Tests:

- Validated workflows between controllers and views, ensuring all parts of a feature functioned together.

3. Edge Cases:

- Special emphasis was given to edge cases like invalid input, restricted actions, or empty database results.

4. Coverage:

- Tools like SimpleCov ensured over 90% test coverage, providing confidence in code robustness.
-

Benefits of BDD/TDD

1. Clear Requirements:

- Feature files in BDD acted as living documentation, aligning team understanding with stakeholder expectations.

2. Early Bug Detection:

- Writing tests first in TDD helped identify bugs early, particularly for features like dropdown filtering or privilege-based actions.

3. Confidence in Refactoring:

- Comprehensive tests allowed safe refactoring, especially when changing UI elements or optimizing controllers.

4. Improved Collaboration:

- BDD's plain-English scenarios encouraged non-developers (e.g., professors or capstone advisors) to review functionality.
-

Challenges Encountered

1. Time-Consuming:

- Writing tests before implementation added overhead during tight sprint timelines.

- JavaScript-based UI testing required more effort to configure and stabilize.
 - 2. **Dynamic Features:**
 - Auto-submitting filters caused previous button-based tests to fail, requiring updates to tests.
 - 3. **Complex Authorization:**
 - Testing role-based access (admin/assistant/student) introduced complexity, especially for edge cases like “unknown” statuses.
 - 4. **Feature Evolution:**
 - Changing requirements (e.g., UI for editing items) required frequent updates to BDD scenarios and TDD tests.
-

Lessons Learned

1. **Balance is Key:** Combining BDD for high-level behavior and TDD for granular logic ensured comprehensive coverage.
2. **Iterate on Tests:** Regularly updating feature files and unit tests prevented misalignment with evolving code.
3. **Automation Tools Matter:** Investing time in reliable tools (e.g., SimpleCov for coverage) was critical for maintaining quality and consistency.

8. Discuss your configuration management approach. Did you need to do any spikes? How many branches and releases did you have?

Branching:

- **Main branch:** Held the tested and released version of the code.
- **Dev branch:** Served as a staging branch for pre-deployment testing.
- **Feature branches:** Each feature or bug fix was developed in its own branch, merged into the dev branch after testing, and then into the main branch upon approval.

Tagging: We had four releases, each tagged according to the corresponding sprint: Sprint 1, Sprint 2, Sprint 3, and Sprint 4.

Spikes:

1. **Database Design:** We conducted an in-depth discussion on database structure, including tables, attributes, and relationships (primary and foreign keys), both within the team and with Paul.
2. **Access Control:** Since students and professors have different permissions (e.g., only professors can create or delete items), we developed access control functionality. The design and implementation were carefully coordinated within the team to ensure seamless integration with ongoing feature development.

9. Discuss any issues you had in the production release process to Heroku -

In the production release process to Heroku, there were no issues encountered. The deployment was smooth and seamless. All steps, including pushing the code to the Heroku repository, setting up environment variables, and configuring the necessary add-ons, were executed without any complications. Additionally, the app was successfully scaled and passed all necessary tests before going live. Overall, the Heroku deployment process went according to plan, and the production release was completed without any setbacks.

10. Describe the tools/Gems you used, such as GitHub, CodeClimate, SimpleCov, and their benefits and problems -

The tools we used namely GitHub, CodeClimate, and SimpleCov are widely used in software development for version control, code quality monitoring, and test coverage analysis, respectively. Each tool offers specific benefits, but also comes with some challenges.

1. GitHub: It is a cloud-based platform primarily used for version control and collaborative development. It is built on Git and allows developers to store and manage code repositories, track changes, and collaborate with other team members.

- Benefits:
 - GitHub helps track changes to code, allowing you to roll back to previous versions, understand how the code evolves, and resolve conflicts.
 - It provides features like pull requests, code reviews, and issues to facilitate team collaboration.
 - GitHub integrates with a variety of tools (e.g., CI/CD pipelines, code quality tools, project management systems).
- Problems:

- Using Git and GitHub might be tough, especially when dealing with branching, merging, and resolving conflicts.
 - If used incorrectly, GitHub workflows may cause confusion, especially for large teams or poorly managed repositories.
-

2. CodeClimate: It is a platform that integrates with your codebase to provide automated analysis on code quality, complexity, maintainability, and test coverage.

- Benefits:
 - It provides detailed reports on code quality, identifying issues such as duplicated code, complex methods, and untested code.
 - It tracks maintainability over time, helping teams ensure that their codebase is evolving in a sustainable way.
 - Users can customize the analysis rules to fit their project's needs.
 - Problems:
 - Sometimes, CodeClimate might flag issues that aren't actually problematic, leading to noise in the reports.
-

3. SimpleCov: It is a Ruby gem used to measure code coverage in a Ruby project. It provides insights into which parts of the codebase are covered by tests and which are not.

- Benefits:
 - It integrates seamlessly with Ruby on Rails and other Ruby projects.
 - SimpleCov generates a detailed report showing the percentage of code covered by tests, helping developers identify untested areas.
 - It supports setting coverage thresholds and displaying badges that indicate the test coverage percentage, helping maintain code quality standards.
 - Developers can configure the output format (HTML, JSON) and specify which parts of the codebase to exclude from coverage.
- Problems:
 - SimpleCov is only useful for Ruby projects. If you need cross-language support, other tools may be necessary.
 - High code coverage doesn't always mean high-quality tests, as coverage can be artificially inflated by trivial tests or tests that don't assess the real behavior of the code.

- Running SimpleCov can sometimes slow down test suites, especially for large codebases, as it has to track every line of code executed during tests.

Yes, We've pushed every test case in the github repo (master branch).

11. Make a separate section discussing your repo contents and the process, scripts, etc., you use to deploy your code. Make very sure that everything you need to deploy your code is in the repo. We have had problems with legacy projects missing libraries. We will verify that everything is in the repo -

1. Deployment Documentation: The README.md file includes detailed instructions for deploying the project on Heroku. This documentation gives step-by-step commands for deployment.
2. Testing Instructions: To reproduce code tests, we provide the exact commands to execute both Cucumber and RSpec tests.
3. Dependencies: The Gemfile lists all necessary libraries and dependencies required to run the code. This ensures that no components are missed during setup, addressing common legacy issues.
4. Documentation Directory (/documentation/Fall2024):
 - a. Sprint Plans: Each sprint plan summarizes meetings with Paul, highlights user stories, and includes UI screenshots representing progress.
 - b. Sprint Retrospectives: These include member contributions, sprint goals, achievements, burndown charts, and UI updates, providing a complete picture of our iterative development process.
 - c. Team Working Agreement: This document evolves with each sprint, reflecting lessons learned and adapting team strategies.
5. Code Quality Reports: We maintain a link to the latest CodeClimate report, which provides insights into code quality, maintainability, and potential issues: CodeClimate Report.

We have verified that all deployment scripts, configuration files, and documentation are included in the repository. This ensures that any future deployments will be as straightforward as possible.

12. Links to your Project Management tool page, public GitHub repo, and Heroku deployment, as appropriate. Make sure these are up-to-date -

Project Management Tool: <https://github.com/users/LOGiC31/projects/1>

Github Repo: <https://github.com/LOGiC31/Stock-Em>

Heroku: <https://stockem-0caaa6588e50.herokuapp.com/welcome/index>

13. Links to your presentation video and demo video -

PPT link:

https://docs.google.com/presentation/d/1ss7HHmnO4VMexPMSh6ozrnPA3l2g1w6f/edit?usp=drive_link&oid=107496203071613881508&rtpof=true&sd=true

Demo link: <https://www.youtube.com/watch?v=aHZVa2nnDgY>