



HACKATHON

HACKATHON: MILESTONE-BASED E-COMMERCE APP WITH NEXT.JS

Event Date: 6th September 2024
Duration: 24 Hours (Starting Midnight after the event)
Submission Deadline: 7th September 2024, Midnight

Ameen Alam
Dean of Faculty

Hackathon: Advanced Inventory Management System with Next.js

Event Date: 06, Sep 2024

Duration: 48 Hours

Submission Deadline: 07 Sep 2024

Overview:

This hackathon challenges you to build an **Inventory Management System** using **Next.js** with advanced features such as user role management, real-time stock updates, and multi-warehouse handling. The system will allow different users (admins, warehouse managers, and customers) to interact with the inventory data. Participants are encouraged to use **React Context API**, **NextAuth.js**, and **Next.js API routes** for a robust and secure application.

Hackathon Structure:

- The hackathon is divided into 6 advanced milestones.
 - Each milestone carries 150 points.
 - You can submit your progress for each completed milestone.
 - Aim to complete as many milestones as possible within the 48-hour limit.
-

Milestone 1: Inventory Dashboard with Real-Time Stock Updates (150 Points)

Objective:

Create an inventory dashboard with real-time stock updates using **Next.js** and **WebSockets**.

Requirements:

- Build a dashboard showing a list of products, current stock levels, and warehouse locations.
- Use **getServerSideProps()** to fetch inventory data initially, and **WebSockets** to reflect real-time stock changes.
- Ensure the UI dynamically updates stock levels when changes occur.

Scoring Breakdown:

- **Real-Time Stock Updates (60 points):**
Does the dashboard update in real-time without page refresh using WebSockets?
- **Responsive and Functional UI (50 points):**
Is the dashboard design responsive, and does it display key data clearly?

- **Data Fetching and Integration (40 points):**
Is `getServerSideProps` used properly for server-side rendering, and is WebSocket integration efficient?
-

Milestone 2: Multi-Warehouse Support with Dynamic Routing (150 Points)

Objective:

Implement support for managing multiple warehouses with dynamic routing for different locations.

Requirements:

- Use **Next.js Dynamic Routing** to create pages for different warehouses, such as `/warehouse/[warehouseId]`.
- Allow users to view products specific to each warehouse.
- Include a feature to transfer stock between warehouses and reflect the change across relevant pages.

Scoring Breakdown:

- **Dynamic Routing (50 points):**
Is dynamic routing properly implemented for multiple warehouses?
 - **Stock Transfer Functionality (50 points):**
Is the stock transfer feature working, and does it reflect in real-time on both warehouse pages?
 - **Warehouse-Level Data Management (50 points):**
Can users view and manage inventory separately for each warehouse?
-

Milestone 3: Role-Based Access Control with NextAuth.js (150 Points)

Objective:

Implement user authentication and role-based access control for admins, warehouse managers, and customers.

Requirements:

- Use **NextAuth.js** to handle authentication.
- Define different roles (Admin, Manager, Customer) and restrict access to certain pages or features.
 - Admins: Full access to all features.
 - Managers: Can manage inventory but not users.
 - Customers: Can only view stock levels, but not modify them.

- Redirect unauthorized users to a "403 Forbidden" page.

Scoring Breakdown:

- **Role-Based Access Control (70 points):**
Is role-based access control implemented properly, with different access levels for different users?
 - **Authentication Flow (50 points):**
Is authentication secure and well-integrated with NextAuth.js?
 - **User Experience and Error Handling (30 points):**
Is unauthorized access handled gracefully with redirection or error messages?
-

Milestone 4: Product Management with CRUD Operations (150 Points)

Objective:

Create CRUD operations (Create, Read, Update, Delete) for managing products in the inventory.

Requirements:

- Implement product creation, editing, deletion, and viewing functionality using **Next.js API Routes**.
- Ensure that only admins and managers can perform these actions.
- Use client-side validation to prevent invalid data input.

Scoring Breakdown:

- **CRUD Operations (70 points):**
Are the create, read, update, and delete functions implemented efficiently with **Next.js API Routes**?
 - **Role-Based Restrictions (40 points):**
Are product management actions restricted based on user roles (admins and managers)?
 - **Data Validation and Error Handling (40 points):**
Is there proper validation on the client side, and are errors handled gracefully?
-

Milestone 5: Real-Time Stock Alerts and Notifications (150 Points)

Objective:

Implement a notification system for stock level alerts, triggered when product levels fall below a certain threshold.

Requirements:

- Use WebSockets or a mock notification API to trigger real-time alerts when stock levels are low.
- Send alerts to managers and admins when stock needs to be replenished.
- Display notifications on the dashboard with clear action items (e.g., "Reorder Product").

Scoring Breakdown:

- **Real-Time Notifications (70 points):**
Are stock alerts triggered and displayed in real-time using WebSockets or API?
 - **Notification Display and UX (50 points):**
Are the notifications displayed clearly on the dashboard with actionable items?
 - **Stock Monitoring Efficiency (30 points):**
Does the system accurately monitor stock levels and notify the right users when necessary?
-

Milestone 6: Reporting and Analytics Dashboard (150 Points)

Objective:

Build a reporting dashboard that provides insights on inventory trends, such as best-selling products, stock turnover rate, and warehouse performance.

Requirements:

- Create charts and graphs using a library like **Chart.js** or **Recharts** to display key metrics.
- Implement filtering options by date range, warehouse, or product category.
- Include export functionality to download reports as CSV or PDF.

Scoring Breakdown:

- **Reporting Functionality (60 points):**
Are key inventory metrics accurately displayed on the dashboard using visualizations?
 - **Filters and Export Options (50 points):**
Can users filter the reports and export them as CSV or PDF?
 - **UI Design and Data Presentation (40 points):**
Is the reporting dashboard user-friendly, with a clear display of trends and performance?
-

Hackathon Timeline:

Start Time: Midnight [Insert Date]

End Time: Midnight [Insert Date]

Submission Deadline: Midnight [Insert Date]

Submission Guidelines:

- **Milestone 1:** Submit your Next.js code for the inventory dashboard with real-time stock updates.
 - **Milestone 2:** Submit the code for multi-warehouse functionality and stock transfer features.
 - **Milestone 3:** Share your role-based access control system with NextAuth.js implemented.
 - **Milestone 4:** Submit the CRUD functionality for product management.
 - **Milestone 5:** Submit the real-time stock alert system.
 - **Milestone 6:** Submit the reporting and analytics dashboard with export functionality.
-

Awards and Recognition:

Participants who complete all milestones will receive the title of "**Master Developer**" and special recognition. Exceptional projects may be featured on the organization's platform or social media.

Evaluation Criteria:

- **Completeness:** Did you meet the objectives of each milestone?
 - **Code Quality:** Is your code clean, well-organized, and efficient?
 - **Functionality:** Does the inventory management system work as expected, including all features?
 - **Design and User Experience:** Is the app visually appealing, responsive, and user-friendly?
 - **Innovation and Creativity:** Did you implement any additional features or show creative problem-solving?
-

Good luck, and we look forward to seeing your advanced Inventory Management Systems built with Next.js!