Software Architecture Models

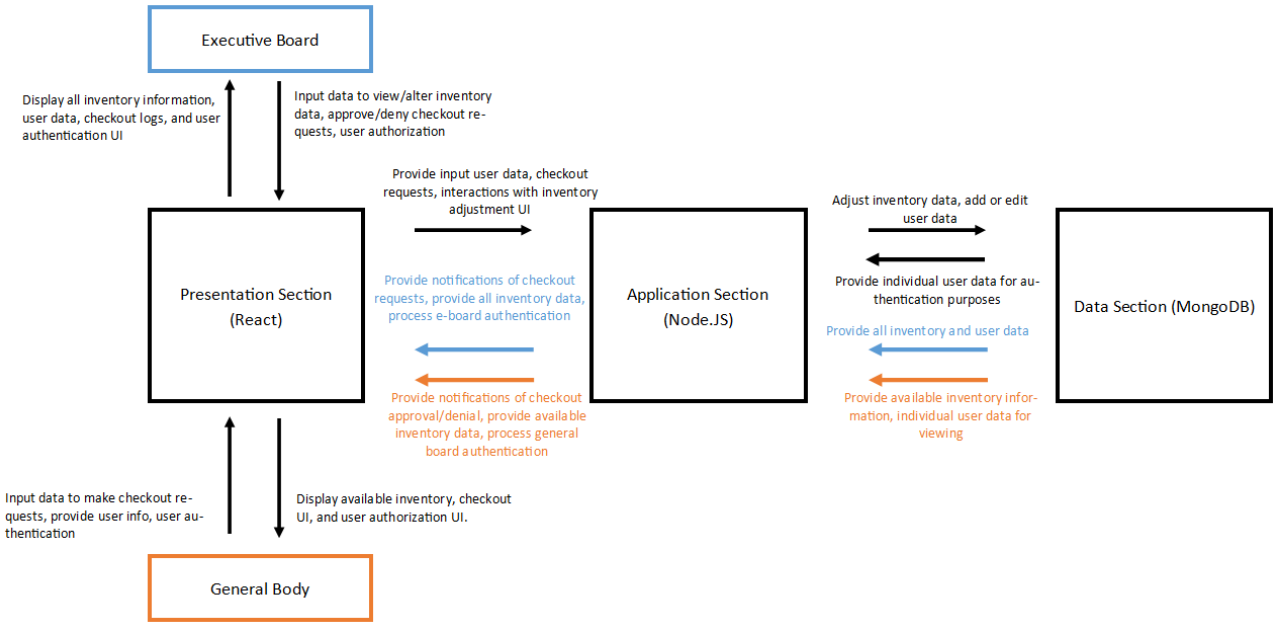
Architecture pattern:

- Hardware Hive will be built on a 3-tiered layered system, utilizing a presentation layer (as part of the front end) to display the UI to all users, an application layer (as part of the back end) to conduct inventory operations and manage authentication and notification services, and a data layer (also as part of the back end) to store the inventory and user data.

Decomposition:

- The presentation layer will allow all users to view the inventory of available parts for checkout, allow users to put in their information for authentication (specifically to determine their club role), and will provide the executive board with the UI to add or remove items from the inventory, add item details, or group items into projects. It will also display a notification system for the executive board when members request to check out parts. The application layer will process authentication and role confirmation for users, conduct operations on the inventory, and confirm availability for various parts and eligibility for checkout. This section will also be responsible for sending email notifications to board members when users request to check out parts and to users when their requests to check out parts are approved/denied. The data layer will store all inventory and user data, allowing the application layer to access and make changes to all items present in the inventory. It will also be responsible for providing the application layer with the user data necessary for user authentication and role confirmation. The presentation layer will be built using React, and will be streamlined for use with Node.js, which will make up the application layer. The data layer will be built using MongoDB, with Python as our language.

Context Model:


Executive Board
Display all inventory information, user data, checkout logs, and user authentication UI
Input data to view/alter inventory data, approve/deny checkout requests, user authorization
Provide input user data, checkout requests, interactions with inventory adjustment UI
Presentation Section (React)
Provide notifications of checkout requests, provide all inventory data, process e-board authentication
Provide notifications of checkout approval/denial, provide available inventory data, process general board authentication
Application Section (Node.JS)
Adjust inventory data, add or edit user data
Provide individual user data for authentication purposes
Provide all inventory and user data
Provide available inventory information, individual user data for viewing
Data Section (MongoDB)
Input data to make checkout requests, provide user info, user authentication
Display available inventory, checkout UI, and user authorization UI.
General Body