Milestone 2

Ryan Coon

CST-391

Professor Bobby Estey

July 9, 2023

**Introduction:**

The application I will be making for this class is a Product Management System similar to a POS sales system with inventory options. This application will allow users to perform CRUD operations on products that are represented as objects with multiple properties. There will be an admin and user level. The admin level will allow that user with the credentials to edit, add, remove, and change quantities of the products in stock. This system will consist of back-end services developed using Express and Node.js. There will be 2 versions of the front end with one being developed with Angular, and the second being developed with React.

**Functionality Requirements:**

* As a user, I want to be able to login with admin privilege in order to add, update, or delete products.
* As a user, I want to be able to view a list of products so I can see what products are offered/available.
* As an admin, I want to be able to create a new product so I can add new products into the inventory.
* As an admin, I want to be able to edit an existing product in the inventory so that I can keep the product information up to date.
* As an admin, I want to be able to delete a product so that I can remove items that are no longer available.
* As a user, I want to be able to view a list of categories so that I can see how the products are organized.
* As a user, I want to be able to create a new category so that I can group products based on their similarities and characteristics.
* As a user, I want to be able to edit an existing category so that I can edit its properties if needed.
* As a user, I want to be able to delete a category if no longer needed.
* As a user, I want to be able to view the product details, including the properties and associated category.
* As a user, I want to be able to view details of the categories including its properties and the list of associated products.

**Initial Database Design:**

**A picture containing screenshot, design

Description automatically generated**

**Initial UI Sitemap:**

**A picture containing screenshot, circle, text

Description automatically generated**

**Initial UI Wireframes:**

**A screenshot of a login page

Description automatically generated with low confidence**

**A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with medium confidence**

**Initial UML Classes:**

* **User:**

Properties: id:int, username:string, password:string, level:int

Methods: getId, setId, getUsername, setUsername, getPassword, setPassword, getLevel, setLevel

* **Products:**

Properties: id:int, name:string, price:decimal, description:string, category:string, quantity:int

Methods: getId, setId, getName, setName, getPrice, setPrice, getDescription, setDescription, getCategory, setCategory, getQuantity, setQuantity

* **Category:**

Properties: id:int, name:string, description:string

Methods: getId, setId, getName, setName, getDescription, setDescription

* **DatabaseConnection:**

Methods: connect(), disconnect(), executeQuery(), executeUpdate()

* **UserController:**

Methods: loginUser(), getUserLevel()

* **ProductController:**

**Methods:** getAllProducts(), getProductById(id), createProduct(product), updateProduct(product), deleteProduct(id)

* **CategoryController:**

Methods: getAllCategories(), getCategoryById(id), createCategory(category), updateCategory(category), deleteCategory(id)

**Risks:**

This project will utilize a design report to identify and track the unknowns and risks that may arise during its development. This would ensure that we have the proper mitigation strategies in place and can be addressed effectively. This design report will contain the actions and document all of the technical decisions and designs, including UML diagrams, ER diagrams, UI designs, and all other necessary artifacts, including installation setup, to support the application design and end-to-end solution.

**API Endpoints**

* GET /api/products: Retrieves a list of all products in the system.
* POST /api/products: Creates a new product based on the provided data in the request body.
* PUT /api/product/:id: Updates the product with the specified ID using the data provided in the request body.
* DELETE /api/product/:id: Deletes the product with the specified ID from the system.