**AmazeKart: An Online Store**

In this assignment you will create the backend for a very basic online store – AmazeKart.With this, we complete the store built in the assignment on Bootstrap and JavaScript.

**Backend for the store application**

The frontend application for this store is the one you completed as part of the Bootstrap and JavaScript assignment. To this frontend you also need to add login functionality. Since endpoints will be authenticated, you also need to add code in the frontend to pass the JWT token in the Authorization HTTP header. The “Review by” field in the reviews form on the product detail page may also be removed as the reviewer details shall be picked up by the backend from the logged in user’s claims (token).

The backend for the store application shall be built by you in this assignment. It shall replace the store backend given in the assignment on Bootstrap and JavaScript. Hence, the port and endpoints shall adhere to the server you used there.

The server you shall build shall serve products. The initial set of products that should be available on server startup have been provided in supplied-files folder (products.json). An initial set of members (users of the application) has also been provided (members.json).

The server shall run on port 4201.

The endpoints you need to implement are

1. GET /products – Returns a JSON array of products
2. GET /products/:id – Returns the JSON object with details of product with given id
3. GET /products/:id/reviews – Returns a JSON array of reviews for product with given id
4. POST /products/:id/reviews – Adds a review for the product with given id. The review details include review title, text and star rating. The reviewer name is picked up from the logged in user’s details (JWT claims object).

All of the above 4 endpoints are authenticated and should return 401 (Unauthorized) if a token is not present in Authorization HTTP header. The token is obtained by hitting the login endpoint (JWT-based authentication).

1. POST /login - Pass the username and password. If member details match one in DB, generate and return a JWT token. Else return appropriate response code

Use Mongoose to connect to a MongoDB database, perform CRUD operations and define Member, Product and Review models with appropriate validations.

Make sure you create a well-defined and maintainable project structure with modules for routing, controller, models etc. Appropriate error-handling needs to be done, with error codes as per situation.