**EXPERIMENT 6**

**AIM:** To implement data security with authentication and authorization.

**THEORY:** Authentication and authorization are essential for securing user data and controlling access to specific features in a web or mobile application. Authentication verifies a user's identity, while authorization determines their level of access.

**1. Authentication**

Authentication is the process of verifying user credentials before granting access to the system. Common authentication methods include:

* **Email & Password Authentication**: Users register with an email and password, which are validated during login. The password is securely hashed before storing it in the database to prevent unauthorized access in case of a data breach.
* **JWT (JSON Web Token) Authentication**: JWT is a secure, stateless authentication mechanism commonly used in Node.js applications. When a user logs in, a JWT token is generated and sent to the client. This token is included in subsequent requests to verify the user's identity without requiring repeated logins. The token contains encoded user information and is signed with a secret key to ensure its integrity.

**2. Authorization**

Authorization defines user permissions within the application. In BlingBay:

* **Only Admins Can Add Products**: Regular users cannot add or manage products.
* **Role-Based Access Control (RBAC)**: Users are assigned roles (Admin or Customer), where only admins have access to product management features.

**Implementation in BlingBay (Node.js & MongoDB)**

We implemented authentication and authorization using Node.js, Express.js, MongoDB, and JWT.

**1. User Registration**

* Users sign up by providing:
  + **Full Name**
  + **Email** (validated for correct format)
  + **Password** (hashed using bcrypt for security, minimum 6 characters, including a capital letter and a symbol)
  + **Confirm Password** (must match the original password)
* The user details are stored in MongoDB.
* The system assigns a default role (Customer).

**2. Login Functionality**

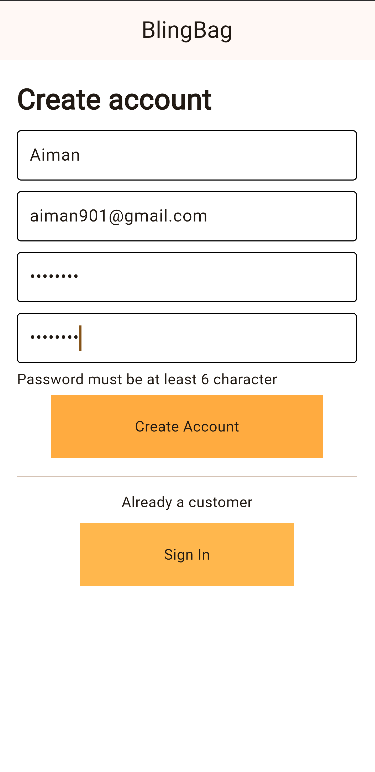
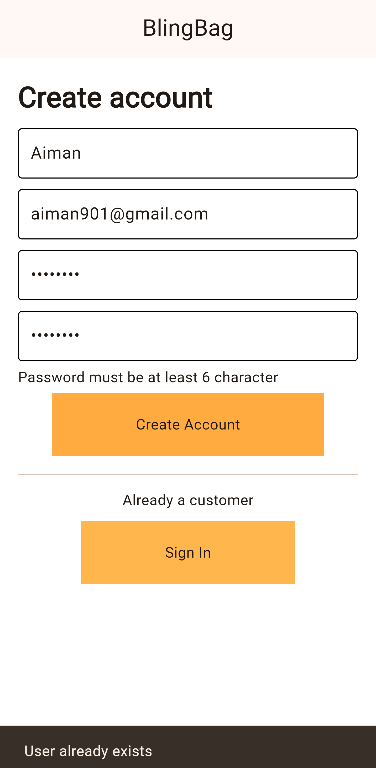
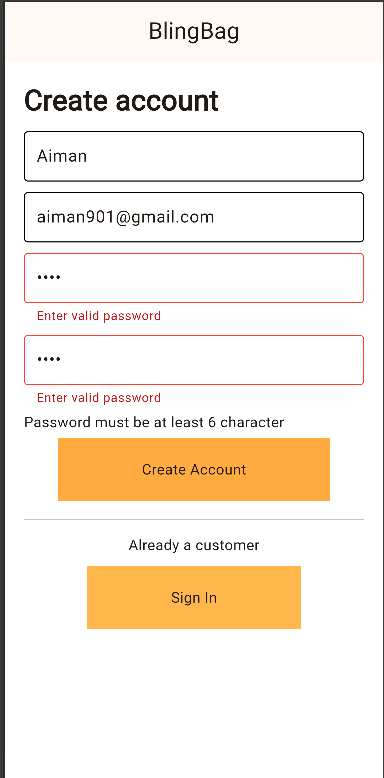
* Users enter their email and password to log in.
* Passwords are verified using bcrypt.
* If credentials match, a JWT token is generated and returned.
* Error handling includes:
  + Incorrect password error.
  + Invalid email format.
  + Unregistered email detection.

**3. Admin Role and Product Management**

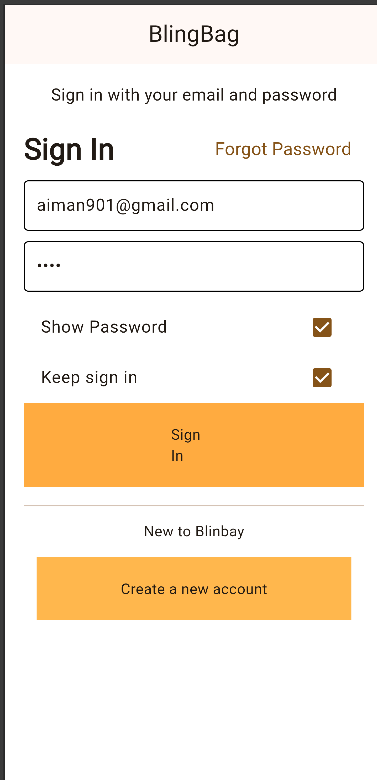
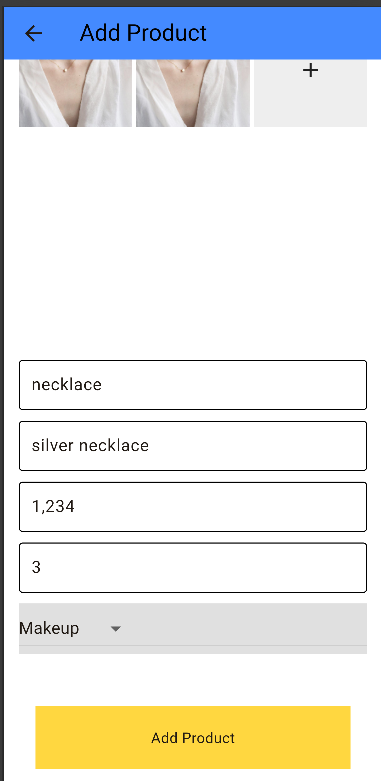
* **Role Assignment**: Admin roles are manually assigned in the database or via an admin panel.
* **JWT-Based Authorization**:
  + Users must provide a valid JWT token to access protected routes.
  + Middleware verifies the token and checks if the user is an admin.
* **Admin-Only Product Management**:
  + Only users with the Admin role can add.
  + Regular users do not have access to product management features.
  + MongoDB and Node.js enforce role-based access control.

**OUTPUT:**

(Include screenshots or logs demonstrating authentication, role assignment, and access control enforcement.)

**  **

**Login: . Authorization**

** **

**CONCLUSION:** In BlingBay, we implemented a secure authentication and authorization system using Node.js, Express.js, MongoDB, and JWT. Only authenticated admins can add or manage products, while regular users have restricted access. This experiment reinforced the importance of authentication and authorization in securing application functionality and user data.