

**9530**

**St. MOTHER THERESA ENGINEERING COLLEGE**

**COMPUTER SCIENCE ENGINEERING**

**NM-ID: 5DD2BBC7BC998FA9F944650DFA5D8E97**

**REG NO: 953023104090**

**DATE:15-09-2025**

**Completed the project named as Phase**

**2**

**FRONT END TECHNOLOGY**

**Login Authentication System**

**SUBMITTED BY:  
J POPE JOHN ALBERT**

**6379976729**

# Login Authentication System

## Solution Design & Architecture

### 1. Tech Stack Selection

- **Frontend:** React.js / HTML5 / CSS3 / JavaScript
- **Backend:** Node.js with Express.js
- **Database:** MongoDB (NoSQL) or MySQL/PostgreSQL (SQL)
- **Authentication:** JWT (JSON Web Token) / OAuth2 (optional for scaling)
- **Security:** bcrypt/argon2 for password hashing, HTTPS, Helmet.js for securing HTTP headers
- **Hosting:** AWS / Heroku / Vercel
- **Version Control:** Git + GitHub/GitLab

### 2. UI Structure / API Schema Design

#### UI Structure (Screens)

- **Login Page** – Email, Password, Login button, Forgot Password link
- **Registration Page** – Name, Email, Password, Confirm Password
- **Dashboard (User)** – Welcome message, profile info
- **Admin Dashboard** – User list, lock/unlock accounts, reset passwords

## API Schema (JSON Structures)

### User Schema (MongoDB Example)

```
{  
  "id": "string",  
  "name": "string",  
  "email": "string",  
  "passwordHash": "string",  
  "role": "user | admin",  
  "isLocked": "boolean",  
  "createdAt": "date",  
  "updatedAt": "date"  
}
```

## 3. Data Handling Approach

- **Input Validation:** Validate fields (email format, strong password policy).
- **Password Storage:** Use bcrypt with salt to hash passwords.
- **Token Management:**
  - Short-lived JWT for authentication.
  - Refresh tokens stored securely for re-login.
- **Error Handling:** Provide meaningful error codes (400, 401, 403).
- **Logs:** Record failed login attempts for monitoring.

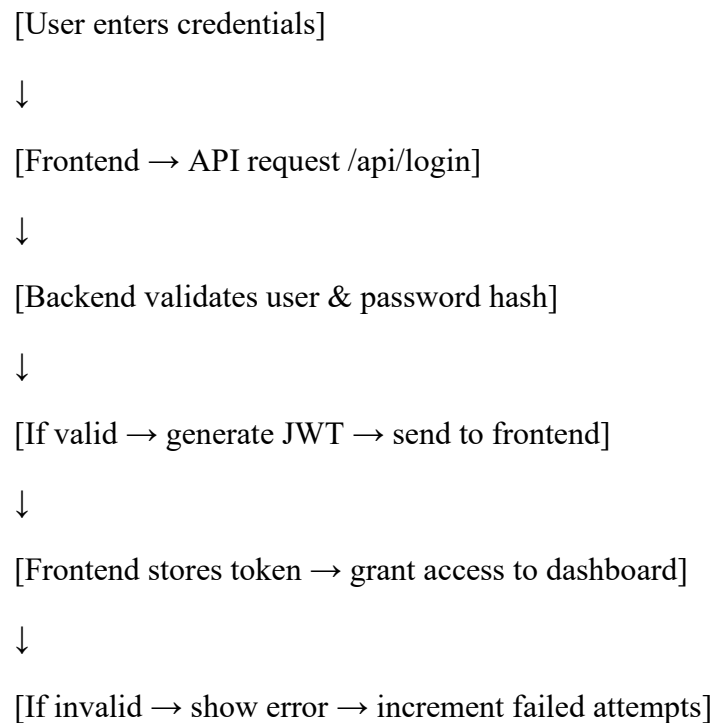
## 4. Component / Module Diagram

### Modules

- **Auth Module**
  - /api/register → Registers user
  - /api/login → Authenticates user
  - /api/logout → Ends session
- **User Module**
  - /api/profile → View/Update profile
- **Admin Module**
  - /api/admin/users → Manage users
  - /api/admin/user/:id → Lock/Unlock users

## 5. Basic Flow Diagram

### Login Flow



## Registration Flow

[User enters name, email, password]



[Frontend → API request /api/register]



[Backend validates input → hashes password → stores in DB]



[Returns success message → redirect to login]