**Full Stack Development with MERN**

**API Development and Integration Report**

|  |  |
| --- | --- |
| Date | 18-07-24 |
| Team ID | SWTID1720076124 |
| Project Name | Online Complaint Registration and Management System |
| Maximum Marks |  |

**Project Title: Online Complaint Registration and Management System**

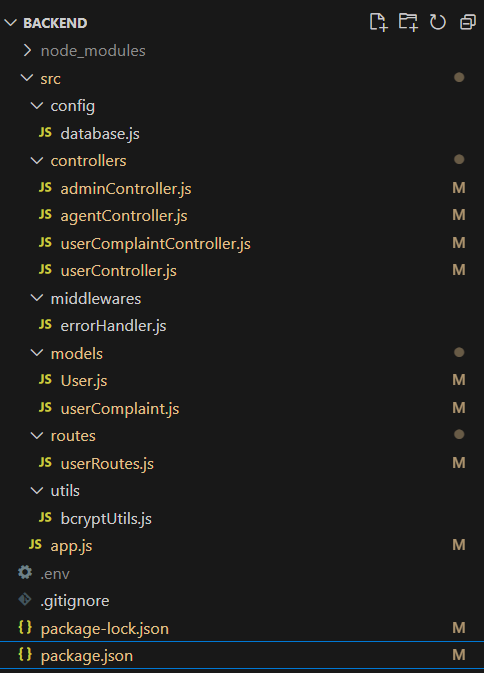
**Date**: 18-07-2024

**Prepared by**: Divyanshi Mittal, V. Sree Teja Pradeep, Manas Kamal Das, Shreya Mahakur

**Objective**  
The objective of this report is to document the API development progress and key aspects of the backend services implementation for the Online Complaint Registration and Management project.

**Technologies Used**

* **Backend Framework:** Node.js with Express.js
* **Database:** MongoDB

**Project Structure**  


**Key Directories and Files**

* **src/config/database.js**:

Manages database configuration and connection setup.

* **src/controllers/adminController.js**:

Handles admin-related API requests and business logic.

* **src/controllers/agentController.js**:

Manages agent-related API requests and business logic.

* **src/controllers/userComplaintController.js**:

Processes user complaint-related API requests and business logic.

* **src/controllers/userController.js**:

Handles user-related API requests and business logic.

* **src/middlewares/errorHandler.js**:

Contains middleware for handling errors across the application.

* **src/models/User.js**:

Defines the User model schema and interactions with the database.

* **src/models/userComplaint.js**:

Defines the UserComplaint model schema and interactions with the database.

* **src/routes/userRoutes.js**:

Sets up and manages user-related API routes.

* **src/utils/bcryptUtils.js**:

Provides utility functions for password hashing and validation using bcrypt.

* **src/app.js**:

Main entry point of the application, sets up the server and integrates middleware.

* **.env**:

Stores environment variables.

* **.gitignore**:

Specifies files and directories to be ignored by Git.

* **package-lock.json**:

Manages the exact versions of installed Node.js dependencies.

* **package.json**:

Contains metadata about the project and lists dependencies and scripts.

**API Endpoints**

A summary of the main API endpoints and their purposes:

**User Authentication**

**- POST /api/signup -** Registers a new user.

**- POST /api/signin -** Authenticates a user and returns a token.

**User Profile**

**- GET /api/getprofile/:userId -** Retrieves user profile information by user ID.

**User Complaints**

**- POST /api/complaintReg -** Registers a new user complaint.

**- GET /api/getusercomplaints/:userId** - Retrieves complaints made by a specific user.

**- GET /api/getallcomplaints-** Retrieves all user complaints.

**- GET /api/pendingcomplaints -** Retrieves all pending complaints.

**Agent Management**

**- GET /api/getagents -** Retrieves all agents.

**- POST /api/assignagent -** Assigns an agent to a complaint.

**- GET /api/viewagentcomplaints/:agentId -** Retrieves complaints assigned to a specific agent.

**- POST /api/updatecomplaintstatus -** Updates the status of a complaint. Integration with Frontend

The backend communicates with the frontend via RESTful APIs. Key points of integration include:

* **Data Fetching:** Frontend components make API calls to fetch necessary data for display and interaction.

**Error Handling and Validation**  
Describe the error handling strategy and validation mechanisms:

* **Error Handling:** Centralized error handling using middleware.

**Security Considerations**  
Outline the security measures implemented:

**Data Encryption:** Encrypt sensitive data at rest and in transit, encrypted using the bcrypt node module.