

Cuvette Backend Module Test – Project Documentation

Student Details

- **Student Name:** Hariharan P
 - **Project Name:** Backend Module API
 - **GitHub Repository:** <https://github.com/Hariharanpugazh/backend-module>
 - **Backend Deployment:** <https://backend-module-go8p.onrender.com>
-

1. Project Overview

The Backend Module project is a RESTful API application designed to demonstrate authentication, authorization, and structured backend architecture. The application is built following modular coding practices with proper validation, error handling, and security mechanisms.

The backend provides API endpoints for managing user authentication workflows and ensures secure handling of user data while maintaining scalable and maintainable code architecture.

2. Features Implemented

Authentication & Authorization

- User Registration
 - User Login
 - Password Encryption
 - Token-Based Authentication (JWT)
 - Protected Routes requiring valid authentication tokens
-

API Functionalities

- Secure user data handling
- Structured API response format (JSON)
- Validation checks for request data
- Proper HTTP status code usage

- Centralized error handling
 - Environment-based configuration
-

3. Technologies Used

- **Node.js / Backend Runtime**
 - **Express.js / Server Framework**
 - **MongoDB / Database**
 - **JWT (JSON Web Token)** – Authentication and Authorization
 - **Bcrypt** – Password Encryption
 - **Dotenv** – Environment Variable Management
 - **REST API Architecture**
-

4. Project Architecture

The project follows modular backend architecture with clear separation of responsibilities.

Folder Structure

backend-module/

- controllers/ – Handles request logic
 - models/ – Database schemas
 - routes/ – API route definitions
 - middleware/ – Authentication & error handling
 - utils/ – Helper functions
 - config/ – Database and environment configuration
 - server.js / app.js – Application entry point
-

5. Validation and Error Handling

The application includes strong validation and error management:

- Request body validation for user inputs
 - Protection against invalid or missing data
 - Graceful error responses with meaningful messages
 - Centralized middleware for handling client and server errors
 - Proper HTTP status codes returned for all responses
-

6. Security Measures

- Passwords are encrypted using bcrypt before storing in database
 - JWT tokens are used for secure session handling
 - Sensitive data stored using environment variables
 - Protected routes prevent unauthorized resource access
 - Secure backend configuration practices followed
-

7. API Response Standardization

All API responses follow a structured JSON format including:

- Status indicator
 - Message description
 - Response data payload (if applicable)
 - Error information for failed requests
-

8. Compliance With Assignment Instructions

Do's Checklist

- ✓ Modular code structure implemented
 - ✓ Authentication using JWT implemented
 - ✓ Passwords encrypted using hashing library
 - ✓ Proper validation and error handling implemented
 - ✓ User input validated and sanitized
 - ✓ JSON-based standardized API responses
 - ✓ Clean and maintainable JavaScript backend code
 - ✓ Status codes used appropriately
 - ✓ Environment variables used for configuration
 - ✓ Sensitive data hidden from production code
 - ✓ Meaningful error messages returned
-

Don'ts Checklist

- ✓ No passwords stored as plain text
- ✓ No sensitive environment variables exposed
- ✓ No direct client-side error stack exposure
- ✓ No unauthorized route access allowed
- ✓ No copied backend code from external sources

9. How to Run the Project Locally

Step 1: Clone Repository

git clone <https://github.com/Hariharanpugazh/backend-module.git>

Step 2: Install Dependencies

npm install

Step 3: Configure Environment Variables

Create `.env` file and configure:

```
PORT=  
MONGO_URI=  
JWT_SECRET=
```

Step 4: Run Server

npm run dev

10. API Deployment

The backend API is deployed and accessible at:

👉 <https://backend-module-go8p.onrender.com>

11. Evaluation Readiness

This project satisfies backend evaluation parameters by:

- Implementing authentication and authorization workflows
- Maintaining structured and scalable backend architecture
- Ensuring secure password handling and token management
- Following REST API best practices
- Handling edge cases and validation effectively
- Returning consistent and clean response formats

12. Conclusion

The Backend Module project demonstrates strong understanding of backend development principles including authentication, security practices, modular design, and API structuring. The project meets all assignment requirements and showcases production-level backend development standards.
