

| Rubric for Node.js Backend | Exceeds Expectations (10-8 marks) | Meets Expectations (7-5 marks) | Needs Improvement (4-2 marks) | Unsatisfactory (1-0 marks) | Marks |
|--|--|---|---|--|-------------|
| Project Structure | Project follows the specified structure perfectly with additional useful organization. | Project follows the specified structure correctly. | Project follows the structure with minor deviations. | Project structure is incorrect or confusing. | /10 |
| Accommodation Management: CRUD Operations | All CRUD operations for accommodations are fully functional and efficient with proper validation. | CRUD operations for accommodations are functional with basic validation. | CRUD operations for accommodations have noticeable issues or lack validation. | CRUD operations for accommodations are non-functional or poorly implemented. | /10 |
| User Authentication | User authentication is secure, fully functional, and properly validated with JWT. | User authentication is functional with JWT but has minor issues or lacks some validation. | User authentication has noticeable issues or lacks proper validation. | User authentication is non-functional or insecure. | /10 |
| Reservation Management: CRUD Operations | All CRUD operations for reservations are fully functional and efficient with proper validation. | CRUD operations for reservations are functional with basic validation. | CRUD operations for reservations have noticeable issues or lack validation. | CRUD operations for reservations are non-functional or poorly implemented. | /10 |
| Middleware Implementation | Middleware for authentication is robust, secure, and reusable. | Middleware for authentication is functional but could be improved. | Middleware for authentication has noticeable issues. | Middleware for authentication is non-functional or insecure. | /10 |
| Error Handling and Status Codes | Comprehensive and informative error handling with appropriate status codes throughout the application. | Sufficient error handling with correct status codes in most cases. | Basic error handling present but lacks in some areas or uses incorrect status codes. | Error handling is missing or incorrect, leading to confusion. | /10 |
| Database Integration with Mongoose | Efficient and optimized integration with MongoDB using Mongoose with proper schemas and relationships. | Functional integration with MongoDB using Mongoose with appropriate schemas. | Basic integration with MongoDB using Mongoose but with some inefficiencies or issues. | Poor integration with MongoDB or incorrect use of Mongoose. | /10 |
| API Documentation and Comments | Comprehensive API documentation and well-commented code throughout the project. | Sufficient API documentation and comments in the code. | Basic documentation and comments present but lacking in detail. | Poor or no documentation and comments, making the code hard to understand. | /10 |
| Security Best Practices | Adheres to security best practices with no vulnerabilities, including JWT and data validation. | Follows basic security practices with minimal vulnerabilities. | Basic security present but with some vulnerabilities. | Poor security with significant vulnerabilities. | /10 |
| Performance and Efficiency | Optimized performance with minimal load times and efficient resource management. | Good performance with acceptable load times. | Performance present but with noticeable delays or inefficiencies. | Poor performance with significant delays or inefficiencies. | /10 |
| Modular and Clean Code Structure | Highly modular and clean code with reusable components and clear separation of concerns. | Modular and clean code with some reusable components. | Basic modularity present but with some issues or lack of reusability. | Poor code structure with significant issues or lack of modularity. | /10 |
| Testing and Validation | Comprehensive testing and validation for all endpoints and functionalities. | Basic testing and validation for most endpoints and functionalities. | Limited testing and validation present but lacking in some areas. | No testing or validation, leading to potential issues. | /10 |
| Overall Functionality and Integration | All functionalities work seamlessly together with smooth integration between components. | Most functionalities work together with minor integration issues. | Some functionalities have noticeable integration issues or lack cohesion. | Significant integration issues or functionalities not working together. | /10 |
| Deployment and Environment Configuration | Well-documented deployment process with environment configurations for different stages (dev, prod). | Basic deployment process with some environment configurations. | Deployment process present but lacking in detail or missing configurations. | Poor or no deployment process, leading to potential issues. | /10 |
| Overall Presentation and Polish | Highly polished and professional presentation with attention to detail. | Good presentation with minor details attended to. | Basic presentation present but lacking in polish. | Poor presentation with major issues or lacking detail. | /10 |
| | | | | Total | /150 |