Product Requirements & Specification Document

Project Name

QuickServe - Restaurant Order API

Description

QuickServe is a backend API for restaurant order management, including menu management, order tracking, and user authentication. Built with Node.js, Express, and MongoDB, it enables efficient handling of restaurant operations.

---

1. Goals & Objectives

Goal	Description
Menu Management	CRUD operations for menu items
Order Management	Create, update, and track orders
User Authentication	Secure access for staff and admins
Simple, RESTful API	Easy integration with frontend or POS systems

---

2. Stakeholders

Role	Responsibility
Owner	Product vision, requirements
Developer	Implementation, testing
Restaurant Staff	API usage, feedback

---

3. Functional Requirements

3.1 User Authentication

Feature	Description
Register	Staff/admin registration
Login	JWT-based authentication
Role Management	Staff vs. admin permissions

3.2 Menu Management

Feature	Description

Create Item	Add new menu item
Read Items	List all menu items
Update Item	Edit menu item details
Delete Item	Remove menu item

3.3 Order Management

Feature	Description
Create Order	Place new order
Read Orders	List/filter orders
Update Order	Change order status (e.g., pending, preparing, served, completed)
Delete Order	Remove/cancel order

---

4. Non-Functional Requirements

Requirement	Description
Security	JWT authentication, password hashing
Performance	<200ms response time for standard requests
Scalability	Support for multiple concurrent users
Documentation	API docs (OpenAPI/Swagger)
Error Handling	Consistent error responses

---

5. API Endpoints (Summary)

5.1 Authentication

Method	Endpoint	Description	Auth Required
POST	/api/auth/register	Register user	No
POST	/api/auth/login	Login user	No

5.2 Menu

Method	Endpoint	Description	Auth Required
GET	/api/menu	List menu items	Yes
POST	/api/menu	Create menu item	Yes (Admin)
PUT	/api/menu/:id	Update menu item	Yes (Admin)
DELETE	/api/menu/:id	Delete menu item	Yes (Admin)

5.3 Orders

Method	Endpoint	Description	Auth Required
GET	/api/orders	List orders	Yes
POST	/api/orders	Create order	Yes
PUT	/api/orders/:id	Update order	Yes
DELETE	/api/orders/:id	Delete order	Yes (Admin)

---

6. Data Models (Simplified)

```
// User
{
  _id: ObjectId,
  username: String,
  password: String (hashed),
  role: String // 'staff' | 'admin'
}

// MenuItem
{
  _id: ObjectId,
  name: String,
  description: String,
  price: Number,
  available: Boolean
}

// Order
{
  _id: ObjectId,
  items: [{ menuItemId: ObjectId, quantity: Number }],
  status: String, // 'pending' | 'preparing' | 'served' | 'completed'
  createdBy: ObjectId,
  createdAt: Date
}
```

---

7. Technology Stack

Component	Technology
Language	Node.js
Framework	Express
Database	MongoDB
Auth	JWT, bcrypt
Docs	Swagger/OpenAPI

---

8. Security & Compliance

- Passwords hashed with bcrypt
- JWT for session management
- Role-based access control
- Input validation and sanitization

---

9. Milestones & Deliverables

Milestone	Deliverable
API Design	OpenAPI/Swagger spec
Authentication Module	Register/Login endpoints
Menu Management Module	CRUD endpoints for menu
Order Management Module	CRUD endpoints for orders
Documentation	API usage guide
Testing	Unit & integration tests

---

10. Acceptance Criteria

- All endpoints function as specified
- Auth and role-based access enforced
- API returns correct, consistent responses
- Documentation is complete and accurate

---

11. Out of Scope

- Payment processing
- Customer-facing frontend
- Analytics/dashboard features

---

12. Appendix

- [OpenAPI/Swagger Spec] (to be delivered)
- [Sample Postman Collection] (to be delivered)

End of Document