# Software Requirements Specification (SRS)

**For**
**Automated Mess Token Booking System (MTBS)**

## 1. Introduction

### 1.1 Purpose

The purpose of this document is to outline the requirements, features, and design for the development of an Automated Mess Token Booking System (MTBS). This system aims to replace the current manual, desk-based token booking process with a digital, efficient, and user-friendly web application.

### 1.2 Project Scope

The MTBS will allow students to pre-book their mess meals online by providing necessary details and making a payment. The system will automatically generate a digital ticket with a unique barcode, which will be sent to the student via email and SMS. An admin portal will allow authorized personnel to view booking history, manage the system, and receive notifications for new bookings. The system will enforce a booking deadline (configurable by the admin, default 10 AM) before the mess timing.

### 1.3 Definitions, Acronyms, and Abbreviations

**MTBS:** Mess Token Booking System
**SRS:** Software Requirements Specification
**User/Student:** An individual who uses the system to book a mess token.
**Admin:** An authorized person who manages the system, views reports, and configures settings.
**UI:** User Interface
**API:** Application Programming Interface

## 2. Overall Description

### 2.1 User Needs

Students need a convenient way to book mess tokens from anywhere, without standing in line.
The mess management needs an accurate count of attendees for efficient food preparation.
The administration needs a digital record of all transactions for auditing and analysis.
The system must prevent double bookings and ensure payment is received before a token is issued.

### 2.2 System Features (High-Level)

**Student Portal:** For token booking, payment, and ticket download.
**Admin Portal:** For monitoring bookings, generating reports, and system configuration.
**Automated Communication:** Sends tickets via email and SMS.
**Payment Gateway Integration:** Securely processes online payments.
**Booking Window Management:** Automatically closes booking at a predefined time.

### 2.3 User Classes and Characteristics

**Student:**

Can register, log in, book tokens, view their booking history, and download tickets.
**Administrator:**

Has full access to the admin portal.
Can view all bookings, manage users, configure system settings (like booking close time), and view financial summaries.
Technically proficient.

## 3. Specific Requirements

### 3.1 Functional Requirements

**Student-Facing (Frontend):**

**FR-1: User Registration & Login**

The system shall allow new students to register using their Email, Phone Number, Name, and Student ID.
The system shall authenticate registered users via Email and Password.

**FR-2: Token Booking**

The system shall provide a form for booking a token. The form shall collect:

Date of Meal
Meal Type (e.g., Lunch, Dinner - fetched from system settings)
Number of Persons
(Email and Phone will be auto-filled from the user's profile).
The system shall only show future dates for which booking is open.

**FR-3: Payment Integration**

The system shall redirect the user to a secure payment gateway (e.g., Stripe, Razorpay) after form submission.
The system shall verify payment success before generating a ticket.

**FR-4: Ticket Generation & Delivery**

Upon successful payment, the system shall generate a unique ticket with:

Unique Ticket ID
Student Name, Email, Phone
Date, Meal Type, Number of Persons
A unique Barcode/QR code representing the Ticket ID.
The system shall allow the user to download the ticket as a PDF.
The system shall automatically send the ticket PDF to the user's registered email and a confirmation SMS to their phone.

**FR-5: Booking History**

The system shall allow a student to view their past and upcoming bookings.

**Admin-Facing (Backend):**

**FR-6: Admin Dashboard**

The system shall provide a dashboard showing key metrics: Today's Bookings, Total Revenue, etc.

**FR-7: View Bookings & History**

The system shall allow the admin to view all bookings, filterable by date, meal type, etc.

**FR-8: System Configuration**

The system shall allow the admin to set the **default token booking closing time** (e.g., 10:00 AM).
The system shall allow the admin to override the closing time for specific dates (e.g., holidays).
The admin shall be able to set meal types and their respective timings and prices.

**FR-9: Admin Notifications**

The system shall display a real-time notification on the admin dashboard for every new booking.

**FR-10: Reporting**

The system shall allow the admin to generate reports (e.g., Daily Collection Report, Attendance Report).

**System-Level:**

**FR-11: Booking Window Logic**

The system shall prevent token booking for a specific meal if the current time is past the configured closing time for that day.

## 4. System Design

### 4.1 System Architecture

A typical 3-tier web application architecture is proposed:

**Presentation Layer (Frontend):** React.js / Angular for a dynamic and responsive UI.
**Application Layer (Backend):** Node.js (Express) to handle business logic, API endpoints, and user authentication.

**Data Layer (Database):** A relational database like MongoDB to store all persistent data.

### 4.2 UI/UX Wireframes (Conceptual)

**Student Booking Flow:**

**Login Screen:** Standard email/password login with a "Register" link.

**Dashboard:** A welcome message and a prominent "Book New Token" button. A list of upcoming bookings.

**Booking Form:**

Date Picker (only future, open dates enabled)

Dropdown for Meal Type (populated from system settings)

Input for Number of Persons (with a max limit, e.g., 5)

A "Proceed to Pay" button.

**Payment Page:** (Hosted by Payment Gateway or custom).

**Success Page:** Displays the generated ticket with a "Download PDF" button and a confirmation message that it was sent via email/SMS.

**Admin Portal:**

**Admin Login:** Separate login URL.

**Admin Dashboard:**

Cards showing: "Bookings Today", "Revenue Today", "Total Active Users".

A live notification feed/list of recent bookings.

A chart showing daily bookings for the week.

Navigation menu: "All Bookings", "Reports", "System Settings".

## 6. External Interface Requirements

**Payment Gateway:** Integration with a third-party API (Stripe, Razorpay) using their SDKs and webhooks for payment verification.

**Email Service:** Integration with node mailer.
**SMS Service:** Integration with .