**## Overview**

**The table booking system provides a comprehensive solution for managing restaurant reservations. It handles table availability, reservation scheduling, and guest management through an intuitive interface while maintaining data consistency and preventing double bookings.**

**Table Booking Features Documentation**

1. Customer Features
   * Making a Reservation
     + Date selection (up to 30 days in advance)
     + Party size selection (1-12 guests)
     + Time slot selection with real-time availability
     + Contact information and special requests
     + Instant confirmation with email
   * Managing Reservations
     + Lookup using confirmation number
     + View/cancel existing reservations
     + Receive email notifications
     + Special requests handling
2. Admin Features
   * Reservation Management
     + View all reservations by date
     + Filter by status (pending/confirmed/cancelled/completed)
     + Update reservation status
     + Handle walk-ins and no-shows
   * Table Management
     + View table occupancy
     + Mark tables as active/inactive
     + Monitor table capacity and availability
3. System Features
   * Smart Availability
     + Real-time availability checking
     + Prevents double bookings
     + Considers reservation duration (2 hours default)
     + Handles overlapping time slots
   * Time Management
     + Operating hours: 5 PM - 11 PM
     + 30-minute time slot intervals
     + 12-hour format display (5:00 PM)
     + Minimum 2-hour notice required
4. Configuration Options
   * Customizable Settings
     + Operating hours by day
     + Maximum party size
     + Advance booking window
     + Holiday closures
     + Time slot intervals
     + Reservation duration

**## System Components**

**### 1. Database Schema**

The system uses three main tables:

- `tables`: Stores restaurant table information

- `reservations`: Manages booking details

- `configurations`: Handles system-wide settings

**#### Tables Schema**

```sql

create table public.tables (

    id uuid default uuid\_generate\_v4() primary key,

    number integer not null unique,

    capacity integer not null,

    is\_active boolean default true

);

```

**#### Reservations Schema**

```sql

create table public.reservations (

    id uuid default uuid\_generate\_v4() primary key,

    table\_id uuid references public.tables(id),

    customer\_name text not null,

    email text not null,

    phone text not null,

    date date not null,

    time time not null,

    party\_size integer not null,

    status text not null check (status in ('pending', 'confirmed', 'cancelled', 'completed')),

    notes text

);

```

**### 2. Core Functions**

**#### Table Availability Check**

```sql

function check\_table\_availability(

    check\_date date,

    check\_time time,

    required\_capacity integer,

    duration\_hours integer default 2

) returns table (

    available boolean,

    table\_id uuid,

    message text

)

```

This function checks table availability considering:

- Table capacity

- Existing reservations

- Reservation duration

- Table active status

**#### Time Slot Availability**

```sql

function get\_available\_time\_slots(

    check\_date date,

    party\_size integer,

    duration\_hours integer default 2

) returns table (

    time\_slot time,

    available boolean

)

```

Returns available time slots for a given date and party size.

**### 3. Frontend Components**

**#### ReservationForm**

- Path: `src/components/reservation/ReservationForm.jsx`

- Purpose: Main booking interface for customers

- Features:

  - Date selection

  - Party size selection

  - Time slot selection

  - Contact information collection

  - Special requests handling

**#### TimeSelector**

- Path: `src/components/reservation/TimeSelector.jsx`

- Purpose: Displays available time slots

- Features:

  - Shows only available times

  - 12-hour time format display

  - Real-time availability updates

**#### ReservationList (Admin)**

- Path: `src/components/admin/ReservationList.jsx`

- Purpose: Reservation management interface

- Features:

  - View all reservations

  - Filter by date/status

  - Update reservation status

  - Cancel reservations

**### 4. Time Handling**

**#### Time Format Utilities**

- Path: `src/utils/timeUtils.js`

- Functions:

  ```javascript

  to12Hour(time: string): string  // Converts "17:00" to "5:00 PM"

  to24Hour(time: string): string  // Converts "5:00 PM" to "17:00"

  ```

**### 5. Configuration**

**#### Reservation Settings**

- Path: `src/config/reservationConfig.js`

- Settings:

  - Operating hours

  - Time slot intervals

  - Maximum party size

  - Advance booking window

  - Minimum notice period

  - Special dates (holidays)

  - Reservation duration

**## User Flows**

**### 1. Making a Reservation**

1. User selects date and party size

2. System checks table availability

3. User selects from available time slots

4. User provides contact information

5. System validates availability again before confirming

6. Confirmation email sent to user

**### 2. Canceling a Reservation**

1. User accesses reservation using confirmation number

2. System verifies reservation status

3. User confirms cancellation

4. System updates reservation status

5. Cancellation email sent to user

**### 3. Admin Management**

1. Admin views reservations for specific date

2. Can filter by status (pending/confirmed/cancelled/completed)

3. Can update reservation status

4. Can view table occupancy

**## Error Handling**

**### 1. Double Booking Prevention**

- Real-time availability checks

- Two-phase validation (initial check and pre-confirmation check)

- Overlapping reservation detection

**### 2. Edge Cases**

- Handle walk-ins

- Late arrivals

- No-shows

- Special requests

- Large party accommodations

**## Security**

**### 1. Data Protection**

- Row Level Security (RLS) policies

- Email validation

- Phone number validation

- Secure status transitions

**### 2. Access Control**

- Admin-only functions

- Public vs authenticated routes

- Rate limiting on reservation attempts

**## Best Practices**

**### 1. Time Management**

- Store times in 24-hour format

- Display in 12-hour format

- Handle timezone considerations

- Consider reservation duration in availability checks

**### 2. Performance**

- Index critical columns

- Optimize availability checks

- Cache common queries

- Handle concurrent bookings

**### 3. User Experience**

- Clear error messages

- Intuitive interface

- Mobile-responsive design

- Real-time updates

**## API Endpoints**

**### 1. Public Endpoints**

- `GET /api/availability`: Check time slot availability

- `POST /api/reservations`: Create new reservation

- `GET /api/reservation/:id`: Get reservation details

- `PUT /api/reservation/:id/cancel`: Cancel reservation

**### 2. Admin Endpoints**

- `GET /api/admin/reservations`: List all reservations

- `PUT /api/admin/reservation/:id/status`: Update status

- `GET /api/admin/tables`: List all tables

- `GET /api/admin/occupancy`: Get table occupancy

**## Testing**

**### 1. Unit Tests**

- Time format conversions

- Availability calculations

- Validation functions

**### 2. Integration Tests**

- Reservation workflow

- Admin operations

- Email notifications

**### 3. Edge Cases**

- Concurrent bookings

- Invalid inputs

- System failures

**## Maintenance**

**### 1. Regular Tasks**

- Clean up old reservations

- Update holiday schedules

- Monitor system performance

- Backup database

**### 2. Monitoring**

- Track failed bookings

- Monitor system usage

- Track popular time slots

- Analyze cancellation patterns

**## Future Improvements**

**### 1. Planned Features**

- Waitlist management

- Table preferences

- Regular customer recognition

- Integration with POS system

**### 2. Scalability**

- Multiple location support

- Custom table layouts

- Dynamic pricing

- Advanced analytics