**Full Workflow Explanation for AI Understanding**

**1️⃣ User Registration & Login**

* **Actors:** Traveler, Driver, Admin (all users)
* **Process:**
  1. User submits registration form with personal info (name, email, password, phone, role).
  2. System creates a new record in **Users** table.
  3. Generates a unique email verification token stored in **EmailVerification**.
  4. Sends email with verification link.
  5. User clicks link to activate account (is\_verified flag updated).
  6. After verification, users can log in with credentials.
  7. System validates email verified and account active before granting access.

**2️⃣ Tourist Profile Setup**

* **Actor:** Traveler
* **Process:**
  1. After login, traveler fills detailed profile (DOB, gender, country, emergency contacts, travel preferences).
  2. These details are stored in the **Tourists** table linked to their UserID.
  3. Travelers can update this profile anytime (updated\_at field tracks changes).
  4. Preferences (like adventure, beach, culture) help personalize suggested trips.

**3️⃣ Driver Registration and Verification**

* **Actor:** Driver
* **Process:**
  1. Driver registers by submitting personal info plus legal docs:
     + NIC number and images
     + Driver license number and image
     + Profile picture
  2. Driver info saved in **Drivers** table linked to their UserID.
  3. Driver submits vehicle details:
     + Vehicle type, make/model, registration number
     + Vehicle photo, insurance expiry
  4. Vehicle info saved in **Vehicles** linked to DriverID.
  5. Driver account status is initially "Pending".
  6. Admin reviews submitted documents and either approves or rejects.
  7. On approval, driver status becomes "Approved".
  8. Only approved drivers can login and receive trip requests.

**4️⃣ Driver Availability Management**

* **Actor:** Driver
* **Process:**
  1. Driver sets their availability dates via dashboard.
  2. Dates marked available/unavailable in **DriverAvailability** table.
  3. Availability is used during trip planning to match drivers who can serve requested trip dates.

**5️⃣ Destination and Distance Management**

* **Actor:** Admin (mostly)
* **Process:**
  1. Admin maintains a list of tourist destinations in **Destinations** table.
  2. Admin inputs known distances and estimated travel durations between destination pairs in **Distances** table.
  3. This dataset enables trip planning without external APIs.

**6️⃣ Trip Planning**

* **Actor:** Traveler
* **Process:**
  1. Traveler chooses origin, one or more destinations, travel dates.
  2. System uses **Destinations** and **Distances** to compute:
     + Total distance
     + Estimated travel time per segment
     + Multi-day trip splits based on max travel per day.
  3. Traveler customizes itinerary with stops, overnight stays, and notes.
  4. Traveler inputs trip preferences (vehicle type, budget, guide).
  5. Trip plan stored in **Trips**, with individual stops stored in **TripStops**, and preferences saved in **TripPreferences**.

**7️⃣ Driver Matching and Booking**

* **Actors:** System (backend), Driver, Traveler
* **Process:**
  1. System queries **DriverAvailability** to find drivers available for the trip dates.
  2. Filters drivers by vehicle type, status ("Approved"), and proximity if applicable.
  3. Sends booking request to eligible drivers.
  4. Driver can accept or reject trip request.
  5. When a driver accepts, system creates a record in **Bookings**, associating trip, driver, and vehicle.
  6. Booking status set to "confirmed" and fare is calculated.
  7. Traveler receives booking confirmation with driver and trip details.

**8️⃣ Trip Execution**

* **Actor:** Driver
* **Process:**
  1. Driver marks trip statuses during trip lifecycle:
     + "En Route"
     + "Started"
     + "Completed"
  2. Traveler can track trip progress via trip status updates.
  3. System logs trip activity (optionally stored in extended logs or audit tables).

**9️⃣ Post-Trip Review**

* **Actor:** Traveler
* **Process:**
  1. After trip completion, traveler submits a review and rating.
  2. Review and rating saved in **TripReviews** table, linked to trip, traveler, and driver.
  3. Ratings contribute to driver reputation and influence future matches.

**10️⃣ Admin Oversight**

* **Actor:** Admin
* **Process:**
  1. Reviews and approves driver registrations and vehicles.
  2. Manages destination and distance data.
  3. Monitors trips, bookings, and user reports.
  4. Can block or deactivate users or drivers violating policies.

**🧠 Summary for AI:**

* **Users** are the base accounts with roles and login credentials.
* **Tourists** extend Users with travel profile details.
* **Drivers** extend Users with verified legal documents + vehicles.
* **Trips** represent planned journeys with multiple stops.
* **Bookings** link trips with driver & vehicle assignments.
* **DriverAvailability** supports scheduling and trip matching.
* **Reviews** close the feedback loop after trip completion.

The system’s behavior is **driven by status fields** (e.g., Driver.Status, Booking.Status) and **foreign keys enforce relationships** between actors and their activities.

SQL  
-- Create the database

CREATE DATABASE TripBookingSystem;

GO

USE TripBookingSystem;

GO

-- USERS TABLE

CREATE TABLE Users (

user\_id INT IDENTITY(1,1) PRIMARY KEY,

full\_name NVARCHAR(100) NOT NULL,

email NVARCHAR(100) UNIQUE NOT NULL,

password NVARCHAR(255) NOT NULL,

phone NVARCHAR(15),

role VARCHAR(10) NOT NULL CHECK (role IN ('traveler', 'driver', 'admin')),

profile\_photo NVARCHAR(255),

created\_at DATETIME DEFAULT GETDATE()

);

-- EMAIL VERIFICATION TABLE

CREATE TABLE EmailVerification (

verification\_id INT IDENTITY(1,1) PRIMARY KEY,

user\_id INT,

verification\_token NVARCHAR(255),

expiration DATETIME,

is\_verified BIT DEFAULT 0,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE

);

-- TOURISTS TABLE

CREATE TABLE Tourists (

tourist\_id INT IDENTITY(1,1) PRIMARY KEY,

user\_id INT UNIQUE NOT NULL,

full\_name NVARCHAR(100) NOT NULL,

phone\_number NVARCHAR(20),

country NVARCHAR(100),

date\_of\_birth DATE,

gender NVARCHAR(20) CHECK (gender IN ('Male', 'Female', 'Other')),

preferred\_language NVARCHAR(50) DEFAULT 'English',

emergency\_contact\_name NVARCHAR(100),

emergency\_contact\_phone NVARCHAR(20),

profile\_picture NVARCHAR(MAX),

travel\_preferences NVARCHAR(MAX),

registration\_date DATETIME DEFAULT GETDATE(),

updated\_at DATETIME NULL,

status NVARCHAR(20) DEFAULT 'Active' CHECK (status IN ('Active', 'Inactive', 'Banned')),

FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE

);

-- DRIVERS TABLE

CREATE TABLE Drivers (

driver\_id INT IDENTITY(1,1) PRIMARY KEY,

user\_id INT UNIQUE NOT NULL,

full\_name NVARCHAR(100) NOT NULL,

phone\_number NVARCHAR(20) NOT NULL,

gender NVARCHAR(10),

date\_of\_birth DATE,

nic\_number NVARCHAR(20) NOT NULL,

nic\_front\_image NVARCHAR(MAX) NOT NULL,

nic\_back\_image NVARCHAR(MAX) NOT NULL,

profile\_picture NVARCHAR(MAX),

license\_number NVARCHAR(50) NOT NULL,

license\_front\_image NVARCHAR(MAX) NOT NULL,

license\_expiry\_date DATE NOT NULL,

police\_clearance\_image NVARCHAR(MAX),

status NVARCHAR(20) DEFAULT 'Pending' CHECK (status IN ('Pending', 'Approved', 'Rejected', 'Blocked')),

registration\_date DATETIME DEFAULT GETDATE(),

updated\_at DATETIME NULL,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE

);

-- VEHICLES TABLE

CREATE TABLE Vehicles (

vehicle\_id INT IDENTITY(1,1) PRIMARY KEY,

driver\_id INT NOT NULL,

vehicle\_type NVARCHAR(10) CHECK (vehicle\_type IN ('car', 'van', 'suv', 'tuk')),

make\_model NVARCHAR(100),

registration\_number NVARCHAR(50) NOT NULL,

vehicle\_photo NVARCHAR(MAX),

insurance\_expiry\_date DATE NOT NULL,

seat\_count INT,

air\_conditioned BIT DEFAULT 1,

verified BIT DEFAULT 0,

FOREIGN KEY (driver\_id) REFERENCES Drivers(driver\_id) ON DELETE CASCADE

);

-- DRIVER AVAILABILITY

CREATE TABLE DriverAvailability (

availability\_id INT IDENTITY(1,1) PRIMARY KEY,

driver\_id INT,

available\_date DATE,

is\_available BIT DEFAULT 1,

FOREIGN KEY (driver\_id) REFERENCES Drivers(driver\_id) ON DELETE CASCADE

);

-- DESTINATIONS TABLE

CREATE TABLE Destinations (

destination\_id INT IDENTITY(1,1) PRIMARY KEY,

name NVARCHAR(100) NOT NULL,

province NVARCHAR(50),

region NVARCHAR(50),

description TEXT

);

-- DISTANCE MATRIX

CREATE TABLE Distances (

from\_id INT,

to\_id INT,

distance\_km FLOAT NOT NULL,

duration\_hours FLOAT NOT NULL,

PRIMARY KEY (from\_id, to\_id),

FOREIGN KEY (from\_id) REFERENCES Destinations(destination\_id),

FOREIGN KEY (to\_id) REFERENCES Destinations(destination\_id)

);

-- TRIPS

CREATE TABLE Trips (

trip\_id INT IDENTITY(1,1) PRIMARY KEY,

traveler\_id INT,

start\_date DATE NOT NULL,

end\_date DATE NOT NULL,

total\_distance FLOAT,

total\_days INT,

estimated\_cost FLOAT,

created\_at DATETIME DEFAULT GETDATE(),

FOREIGN KEY (traveler\_id) REFERENCES Users(user\_id)

);

-- TRIP STOPS

CREATE TABLE TripStops (

stop\_id INT IDENTITY(1,1) PRIMARY KEY,

trip\_id INT,

destination\_id INT,

stop\_order INT,

trip\_day INT,

overnight\_stay BIT DEFAULT 0,

stop\_notes TEXT,

FOREIGN KEY (trip\_id) REFERENCES Trips(trip\_id),

FOREIGN KEY (destination\_id) REFERENCES Destinations(destination\_id)

);

-- TRIP PREFERENCES

CREATE TABLE TripPreferences (

preference\_id INT IDENTITY(1,1) PRIMARY KEY,

trip\_id INT,

vehicle\_type NVARCHAR(10) CHECK (vehicle\_type IN ('car', 'van', 'suv')),

budget\_range NVARCHAR(10) CHECK (budget\_range IN ('low', 'medium', 'luxury')),

need\_guide BIT DEFAULT 0,

need\_accommodation\_help BIT DEFAULT 0,

special\_requests TEXT,

FOREIGN KEY (trip\_id) REFERENCES Trips(trip\_id)

);

-- BOOKINGS

CREATE TABLE Bookings (

booking\_id INT IDENTITY(1,1) PRIMARY KEY,

trip\_id INT,

driver\_id INT,

vehicle\_id INT,

status NVARCHAR(15) DEFAULT 'pending' CHECK (status IN ('pending', 'confirmed', 'rejected', 'completed')),

fare FLOAT,

payment\_status NVARCHAR(15) DEFAULT 'not\_paid' CHECK (payment\_status IN ('not\_paid', 'paid', 'partial')),

created\_at DATETIME DEFAULT GETDATE(),

FOREIGN KEY (trip\_id) REFERENCES Trips(trip\_id),

FOREIGN KEY (driver\_id) REFERENCES Users(user\_id),

FOREIGN KEY (vehicle\_id) REFERENCES Vehicles(vehicle\_id)

);

-- REVIEWS

CREATE TABLE TripReviews (

review\_id INT IDENTITY(1,1) PRIMARY KEY,

trip\_id INT,

traveler\_id INT,

driver\_id INT,

rating INT CHECK (rating BETWEEN 1 AND 5),

review\_text TEXT,

created\_at DATETIME DEFAULT GETDATE(),

FOREIGN KEY (trip\_id) REFERENCES Trips(trip\_id),

FOREIGN KEY (traveler\_id) REFERENCES Users(user\_id),

FOREIGN KEY (driver\_id) REFERENCES Users(user\_id)

);