Problem Statement: Shipment Management System

Objective

Develop a **Shipment Management API** using Django Rest Framework (DRF) and **JWT Stateless Authentication**. The API will enable users to manage shipments based on their roles: **Admin** and **Customer**.

- **Admin** users can perform CRUD operations on shipments.
- **Customers** can view shipments associated with them.
- **Authentication** is required for all operations except login.

Models

1. **Role Model**
    - `role` (string): Defines user roles (e.g., **Admin** or **Customer**).
2. **User Model**
    - `username` (string): Unique identifier for the user.
    - `email` (string): Email address for login.
    - `password` (string): Encrypted password.
    - `roleobj` (ForeignKey to Role): Links a user to their role.
3. **Shipment Model**
    - `tracking_number` (string): Unique identifier for each shipment.
    - `shipment_status` (string): Status of the shipment (e.g., **"Delivered"**, **"In Transit"**).
    - `receiver_name` (string): Name of the shipment receiver.
    - `senderobj` (ForeignKey to User): Links a shipment to the user who created it.
    - `created_date` (DateTime): Timestamp of shipment creation.

API Endpoints with Restrictions

1. User Login

- **Endpoint**: **POST /user/login**
- **Description**: Authenticate the user using email and password to generate a JWT access token.
- **Input**:

```
{
  "email": "user@example.com",
  "password": "string"
}
```

- **Output**:
    - Success:

```
{
  "access": "jwt_token",
  "username": "user_name"
}
```

- Failure:

```
{
  "error": "Invalid credentials"
}
```

- **Restrictions**: No authentication required.

---

2. List Shipments

- **Endpoint**: **GET /shipment/list**
- **Description**: Retrieve shipment details based on a tracking number or for all shipments associated with the authenticated user.
- **Input**: Query parameter `trackingNumber` (optional).
- **Output**: List of shipments for the authenticated user.

```
[
  {
    "tracking_number": "TRK12345",
    "shipment_status": "Delivered",
    "receiver_name": "John Doe",
    "sender": "admin_username",
    "created_date": "2024-12-01T00:00:00Z"
  }
]
```

- **Restrictions**:
  - **Admin**: Can retrieve details of any shipment.
  - **Customer**: Can only retrieve shipments created by them.
  - Any **authenticated user** can view shipments by **tracking number**.

---

3. Add Shipment

- **Endpoint**: **POST /shipment/add**
- **Description**: Create a new shipment.
- **Input**:

```
{
  "tracking_number": "string",
  "shipment_status": "string",
  "receiver_name": "string"
}
```

- **Output**:
  - Success:

```
{
  "id": 1,
  "tracking_number": "TRK12345",
  "shipment_status": "Pending",
  "receiver_name": "John Doe",
  "sender": "admin_username",
  "created_date": "2024-12-01T00:00:00Z"
}
```

- **Restrictions**:
  - Only **Admin** users can create new shipments.
  - **Customers** are not allowed to create shipments.
  - There can be **multiple Admins**, and any **Admin** can perform CRUD operations on shipments.

---

4. Update Shipment

- **Endpoint**: **PUT /shipment/update/int:id**
- **Description**: Update the status of a shipment.
- **Input**:

```
{
  "shipment_status": "Delivered"
}
```

- **Output**:
  - Success:

```
{
  "id": 1,
  "tracking_number": "TRK12345",
  "shipment_status": "Delivered",
  "receiver_name": "John Doe",
  "sender": "admin_username",
  "created_date": "2024-12-01T00:00:00Z"
}
```

- **Restrictions**:
  - Only **Admin** users can update shipments.
  - Admin users can only update shipments they created.
  - **Customers** are not allowed to update shipments.

---

5. Delete Shipment

- **Endpoint**: **DELETE /shipment/delete/int:id**
- **Description**: Delete a shipment.
- **Output**:
  - Success:

```
{
  "message": "Shipment deleted successfully"
}
```

  - Failure:

```
{
  "error": "Unauthorized to delete this shipment"
}
```

- **Restrictions**:
  - Only **Admin** users can delete shipments.
  - Admin users can only delete shipments they created.
  - **Customers** are not allowed to delete shipments.

Authentication

- **JWT Authentication**:
  - All endpoints (except `/user/login`) require a valid JWT token passed in the `Authorization` header as `Bearer <token>`.
  - Tokens are role-based to enforce access control.

Validation

1. **Role and Permissions**:
   - Validate the user's role at each endpoint.
   - Restrict operations like **Add**, **Update**, and **Delete** to **Admin** users.
   - **Admin** users can perform all CRUD operations.

- **Admin** users can **Read all shipments**, but they can only **Update** and **Delete shipments** they created.

2. **Shipment Ownership**:
    - Ensure that only the Admin user who created the shipment can update or delete it.
    - **Customers** cannot modify or delete shipments, but they can view them.

Business Rules

1. **Shipment Creation**:
    - Only **Admin** users can create shipments.
    - There can be **multiple Admins**, and any **Admin** can perform CRUD operations on shipments.
2. **Shipment Updates**:
    - **Admin** users can update any shipment they created.
    - **Customers** can only view shipments associated with their username.
3. **Shipment Deletion**:
    - **Admin** users can delete any shipment they created.
    - **Customers** cannot delete shipments.
4. **Shipment Read**: -- **Admin** users can view all shipments details.
    - Any **authenticated user** can view shipment details by providing a **tracking number** in the query parameter.
