**Enums and Their API Usage**

1. **RoleType**
   * **API:** /api/enums/roles
   * **Use:** Represents user roles (Admin, Employee, Agent, Customer) for assigning roles to users.
2. **PolicyStatus**
   * **API:** /api/enums/policy-status
   * **Use:** Indicates the status of a policy (Active, Lapsed, Surrendered).
3. **ClaimStatus**
   * **API:** /api/enums/claim-status
   * **Use:** Represents the progress of a claim (Submitted, InProgress, Approved, Rejected).
4. **TransactionType**
   * **API:** /api/enums/transaction-types
   * **Use:** Defines the types of transactions (Issuance, Renewal, Cancellation, Refund).
5. **PremiumType**
   * **API:** /api/enums/premium-types
   * **Use:** Specifies the frequency of premium payments (Monthly, Quarterly, Yearly).
6. **InstallmentStatus**
   * **API:** /api/enums/installment-status
   * **Use:** Indicates the status of an installment (Pending, Paid, Overdue).
7. **RefundStatus**
   * **API:** /api/enums/refund-status
   * **Use:** Tracks the state of refund requests (Pending, Approved, Rejected).
8. **NotificationType**
   * **API:** /api/enums/notification-types
   * **Use:** Describes the type of notifications (Email, SMS, Push).
9. **DepartmentType**
   * **API:** /api/enums/department-types
   * **Use:** Categorizes employee departments (Claims, Underwriting, Support).
10. **RiderType**
    * **API:** /api/enums/rider-types
    * **Use:** Lists available rider types (Accidental, CriticalIllness, TermExtension).
11. **EarningType**
    * **API:** /api/enums/earning-types
    * **Use:** Represents types of agent earnings (Registration, Installment, Rider).
12. **PaymentStatus**
    * **API:** /api/enums/payment-status
    * **Use:** Tracks the state of a payment (Pending, Completed, Failed).



Here’s the list of APIs we created for the **User** entity, along with their usage:

**User API Endpoints**

1. **GET /api/user**
   * **Use:** Fetch all users in the system.
2. **GET /api/user/{id}**
   * **Use:** Retrieve details of a specific user by their unique UserId.
3. **POST /api/user**
   * **Use:** Add a new user to the system. Accepts user details in the request body and hashes the password before storing.
4. **PUT /api/user**
   * **Use:** Update an existing user’s details (e.g., email, username, password).
5. **DELETE /api/user/{id}**
   * **Use:** Remove a user from the system by their unique UserId.