Application Testing Documentation

Employee Management & Lookup System

Your Name

December 12, 2025

Contents

1 Introduction

This document describes the testing process and functional flow of the **Employee Management & Lookup System** application. The system is developed using NestJS and PostgreSQL and includes modules for Users, Lookup Types, and Lookups. The purpose is to provide clarity on endpoints, example requests/responses, and overall application workflow.

2 Modules Overview

2.1 User Module

The User module allows registration, authentication, updating, and retrieval of users.

2.1.1 Register User

Endpoint: POST /users/register

Request:

```
{
  "username": "moien",
  "password": "12345",
  "email": "moien@example.com",
  "first_name": "Moien",
  "last_name": "Uddin"
}
```

Response:

```
{
  "id": "47d4d466-...",
  "username": "moien",
  "is_active": true,
  "first_name": "Moien",
  "last_name": "Uddin",
  "email": "moien@example.com",
  ...
}
```

2.1.2 Sign In User

Endpoint: POST /users/signin

Request:

```
{
  "username": "moien",
  "password": "12345"
}
```

Response:

```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR..."
}
```

2.1.3 Update User

Endpoint: PUT /users/update/:id

Request:

```
{
  "first_name": "Moien Updated",
  "last_name": "Uddin Updated",
  "is_active": false
}
```

Response:

```
{
  "id": "47d4d466-...",
  "first_name": "Moien Updated",
  "last_name": "Uddin Updated",
  "is_active": false,
  ...
}
```

2.2 Lookup Types Module

2.2.1 Create Lookup Type

Endpoint: POST /lookup-types

Request:

```
{
  "name": "Department",
  "is_active": true
}
```

Response:

```
{
  "id": "e71987c3-...",
  "name": "Department",
  "is_active": true,
  ...
}
```

2.2.2 Update Lookup Type

Endpoint: PUT /lookup-types/:id

Request:

```
{
  "name": "Role",
  "is_active": false
}
```

Response:

```
{
  "id": "e71987c3-...",
  "name": "Role",
  "is_active": false,
```

```
  ...
}
```

2.3 Lookups Module

2.3.1 Create Lookup

Endpoint: POST /lookups

Request:

```
{
  "name": "Finance",
  "short_name": "FIN",
  "lookup_type_id": "e71987c3-...",
  "is_active": true
}
```

Response:

```
{
  "id": "1967d59b-...",
  "name": "Finance",
  "short_name": "FIN",
  "lookup_type": {
    "id": "e71987c3-...",
    "name": "Department",
    "is_active": true
  },
  "is_active": true,
  ...
}
```

2.3.2 Update Lookup

Endpoint: PUT /lookups/:id

Request:

```
{
  "name": "Human Resources",
  "short_name": "HR",
  "is_active": false
}
```

Response:

```
{
  "id": "1967d59b-...",
  "name": "Human Resources",
  "short_name": "HR",
  "lookup_type": {
    "id": "e71987c3-...",
    "name": "Department"
  },
  "is_active": false,
  ...
}
```

3 Application Flow

Flow Description:

- Users register and sign in.
- Lookup Types are created to categorize lookups (e.g., Department, Role).
- Lookups are created and linked to a Lookup Type.
- Users can be associated with lookups for additional functionality.

4 Conclusion

This document presents professional snapshots of request and response payloads for all application modules. All endpoints for Users, Lookup Types, and Lookups have been tested successfully, showing proper CRUD operations and relationships.