# Codex API Documentation

## Base URL

All endpoints are relative to the base URL: `http://127.0.0.1:8000/api/`

## Authentication

API authentication is required for certain endpoints. Include the `Authorization` header with a valid token.

## Endpoints

# User API

## Endpoints

### 1. GET /users

Returns a list of all users.

**Request**

No request body required.

**Response**

```
[
  {
    "id": 1,
    "first_name": "John",
    "last_name": "Doe",
    "email": "john.doe@example.com",
    ...
  },
  ...
]
```

PROF

### 2. GET /users/{id}

Returns a single user by ID.

**Request**

No request body required.

**Response**

```
{
  "id": 1,
  "first_name": "John",
  "last_name": "Doe",
  "email": "john.doe@example.com",
  ...
}
```

### 3. POST /users

Creates a new user.

**Request**

```
{
  "first_name": "Jane",
  "last_name": "Doe",
  "email": "jane.doe@example.com",
  ...
}
```

**Response**

```
{
  "id": 2,
  "first_name": "Jane",
  "last_name": "Doe",
  "email": "jane.doe@example.com",
  ...
}
```

PROF

### 4. PUT /users/{id}

Updates a user by ID.

**Request**

```
{
  "first_name": "Updated John",
  "last_name": "Updated Doe",
```

```
  ...
}
```

**Response**

```
{
  "id": 1,
  "first_name": "Updated John",
  "last_name": "Updated Doe",
  ...
}
```

5. DELETE /users/{id}

Deletes a user by ID.

**Request**

No request body required.

**Response**

```
{
  "message": "User deleted successfully."
}
```

## User Object

PROF

The User object has the following properties:

- id: The user's unique identifier.
- first_name: The user's first name.
- last_name: The user's last name.
- date_of_birth: The user's date of birth.
- gender: The user's gender.
- national_id: The user's national ID.
- phone_number: The user's phone number.
- address: The user's address.
- city: The user's city.
- marital_status: The user's marital status.
- profile_photo_path: The path to the user's profile photo.
- email: The user's email.
- email_verified_at: The timestamp when the user's email was verified.
- password: The user's password.

- supervisor_id: The ID of the user's supervisor.
- salary_reference_number: The user's salary reference number.
- section_id: The ID of the user's section.
- years_of_service: The number of years the user has been in service.
- current_team_id: The ID of the user's current team.
- role_id: The ID of the user's role.
- position: The user's position.
- tax_identification_number: The user's tax identification number.
- social_security_number: The user's social security number.
- bank_account_number: The user's bank account number.
- bank_name: The name of the user's bank.
- bank_branch: The branch of the user's bank.
- salary_scale: The user's salary scale.
- basic_salary: The user's basic salary.
- housing_allowance: The user's housing allowance.
- transport_allowance: The user's transport allowance.
- other_allowance: The user's other allowance.
- total_salary: The user's total salary.
- education: The user's education.
- created_at: The timestamp when the user was created.
- updated_at: The timestamp when the user was last updated.
- date_of_employment: The user's date of employment.

# Organisation API

## Endpoints

### 1. GET /organisations

Returns a list of all organisations.

PROF

**Request**

No request body required.

**Response**

```
[
  {
    "id": 1,
    "name": "Example Corp",
    "location": "Cityville",
    ...
  },
  ...
]
```

## 2. GET /organisations/{id}

Returns a single organisation by ID.

**Request**

No request body required.

**Response**

```
{
  "id": 1,
  "name": "Example Corp",
  "location": "Cityville",
  ...
}
```

## 3. POST /organisations

Creates a new organisation.

**Request**

```
{
  "name": "New Corp",
  "location": "Townsville",
  ...
}
```

PROF

**Response**

```
{
  "id": 2,
  "name": "New Corp",
  "location": "Townsville",
  ...
}
```

## 4. PUT /organisations/{id}

Updates an organisation by ID.

**Request**

```
{
  "name": "Updated Corp",
  "location": "Updated City",
  ...
}
```

**Response**

```
{
  "id": 1,
  "name": "Updated Corp",
  "location": "Updated City",
  ...
}
```

5. DELETE /organisations/{id}

Deletes an organisation by ID.

**Request**

No request body required.

**Response**

```
{
  "message": "Organisation deleted successfully."
}
```

PROF

## Organisation Object

- id: The organisation's unique identifier.
- organisation_name: The name of the organisation.
- organisation_logo: The logo of the organisation.
- organisation_mission: The mission statement of the organisation.
- organisation_vision: The vision statement of the organisation.
- created_at: The timestamp when the organisation was created.
- updated_at: The timestamp when the organisation was last updated.

# Department API

## Endpoints

### 1. GET /departments

Returns a list of all departments.

**Request**

No request body required.

**Response**

```
[
  {
    "id": 1,
    "name": "HR Department",
    ...
  },
  ...
]
```

### 2. GET /departments/{id}

Returns a single department by ID.

**Request**

No request body required.

**Response**

```
{
  "id": 1,
  "name": "HR Department",
  ...
}
```

PROF

### 3. POST /departments

Creates a new department.

**Request**

```
{
  "name": "Finance Department",
  ...
}
```

**Response**

```
{
  "id": 2,
  "name": "Finance Department",
  ...
}
```

### 4. PUT /departments/{id}

Updates a department by ID.

**Request**

```
{
  "name": "Updated Finance Department",
  ...
}
```

**Response**

```
{
  "id": 2,
  "name": "Updated Finance Department",
  ...
}
```

PROF

### 5. DELETE /departments/{id}

Deletes a department by ID.

**Request**

No request body required.

**Response**

```
{
  "message": "Department deleted successfully."
}
```

## Department Object

- `id`: The department's unique identifier.
- `department_name`: The name of the department.
- `department_location`: The location of the department.
- `organisation_id`: The ID of the organisation the department belongs to.
- `created_at`: The timestamp when the department was created.
- `updated_at`: The timestamp when the department was last updated.

# Section API

## Endpoints

### 1. GET /sections

Returns a list of all sections.

**Request**

No request body required.

**Response**

```
[
  {
    "id": 1,
    "section_name": "IT Section",
    ...
  },
  ...
]
```

PROF

### 2. GET /sections/{id}

Returns a single section by ID.

**Request**

No request body required.

**Response**

```
{
  "id": 1,
  "section_name": "IT Section",
  ...
}
```

### 3. POST /sections

Creates a new section.

**Request**

```
{
  "section_name": "Finance Section",
  ...
}
```

**Response**

```
{
  "id": 2,
  "section_name": "Finance Section",
  ...
}
```

### 4. PUT /sections/{id}

Updates a section by ID.

PROF

**Request**

```
{
  "section_name": "Updated Finance Section",
  ...
}
```

**Response**

```
{
  "id": 2,
  "section_name": "Updated Finance Section",
```

```
  ...
}
```

5. DELETE /sections/{id}

Deletes a section by ID.

**Request**

No request body required.

**Response**

```
{
  "message": "Section deleted successfully."
}
```

## Section Object

- id: The section's unique identifier.
- section_name: The name of the section.
- department_id: The ID of the department the section belongs to.
- created_at: The timestamp when the section was created.
- updated_at: The timestamp when the section was last updated.

# Role API

## Endpoints

PROF

1. GET /roles

Returns a list of all roles.

**Request**

No request body required.

**Response**

```
[
  {
    "id": 1,
    "role_title": "Chief",
    ...
  },
```

```
  ...
]
```

### 2. GET /roles/{id}

Returns a single role by ID.

**Request**

No request body required.

**Response**

```
{
  "id": 1,
  "role_title": "Chief",
  ...
}
```

### 3. POST /roles

Creates a new role.

**Request**

```
{
  "role_title": "Senior",
  ...
}
```

PROF

**Response**

```
{
  "id": 2,
  "role_title": "Senior",
  ...
}
```

### 4. PUT /roles/{id}

Updates a role by ID.

**Request**

```
{
  "role_title": "Updated Senior",
  ...
}
```

**Response**

```
{
  "id": 2,
  "role_title": "Updated Senior",
  ...
}
```

### 5. DELETE /roles/{id}

Deletes a role by ID.

**Request**

No request body required.

**Response**

```
{
  "message": "Role deleted successfully."
}
```

PROF

## Role Object

- `id`: The role's unique identifier.
- `role_title`: The title of the role.
- `created_at`: The timestamp when the role was created.
- `updated_at`: The timestamp when the role was last updated.