1 Abstract

The task of the assignment is to develop a CRUD backend server application using the Python programming language and the server Django REST framework. As per the description of the application, an Employee Management System is created. The employees are organized in various teams and are paid on hourly basis. Few of the members are team leader and are paid 10 percent extra salary as compared to the other employees. Not everybody in the company is a full time employee. The task is to create an API to retrieve the list of employees with their respective salary.

Three classes are created namely Employee, WorkArr (for work arrangement) and Team. Each team has a team leader. The salary is calculated based on the hourly rate and the employment type. For CRUD operations, Postman is used. Foreign keys are used to defined the relations between the classes.

2 Tools and Editors

- Visual Studio Code - for building and debugging application
- pgAdmin - for database management in PostgreSQL.
- GIT - for version control.
- Postman - for create, update, delete, get operations.
- LaTeX - for creating this document.

3 Outline of the Application

The very first task of the development of any API is to create a database schema. The outline of the assignment is shown below in fig. 1. The name of the project is EDI_EMS while the name of the application is employeeapi.

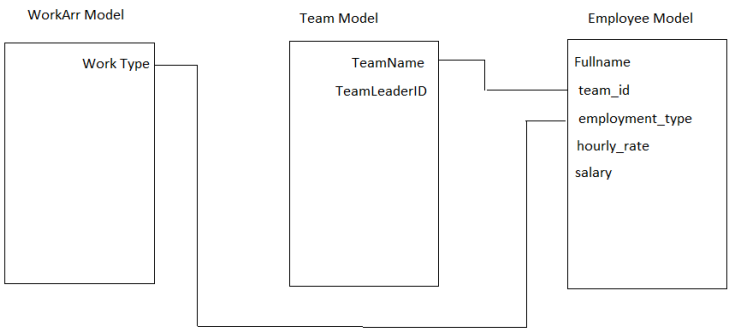

Figure 1: Database Schema for the Application

3.1 Model Description

The application has 3 model classes WorkArr, Team and Employee.

3.1.1 WorkArr Model

It has one character field WorkType having a maximum length of 50.

3.1.2 Team Model

Team Model has two fields.

- TeamName - character field having maximum length of 100.
- TeamLeaderId - ID for team leader of the team.

3.1.3 Employee Model

It has five fields.

- fullname - character field having maximum length of 100.
- Team_id - foreign key from Team model.
- employment_type - Full time or part time
- hourly_rate - float field as paid on hourly basis
- salary - float field for monthly salary (default = 10)

Monthly salary is calculated by get_salary method in which a full time employee salary is calculated by multiplying hourly_rate by 160 and part time employee salary is calculated by multiplying hourly_rate by 120.

Team Leader salary is calculated by multiplying its salary by 1.1.

3.2 Viewsets, Routers and Serializers

Django viewsets internally creates classes like list(), create() for CRUD operations. Routers are used for providing functionality for automatically mapping the URL's tothe logic that deals with incoming requests. Serializers are used for converting querysets to Python datatypes. They are also used for deserialization of Python datatypes back to the complex types.

4 Some Useful Commands and Libraries

- To install Django - pip -m install Django
- To install REST framework - pip install djangorestframework
- To install coverage - pip install coverage
- To install psycopg2 - pip install psycopg2
- To start a project - django-admin startproject <project name>

- To start an application - python manage.py startapp <application name>
- To run the server - python manage.py runserver
- To make the migrations - python manage.py makemigrations <application name> and python manage.py migrate
- To run coverage - coverage run manage.py test <application name>
- To get the coverage report - coverage report

5 Results and Observations

All the results and observations of the assignment are put in the test report.

6 References

Although it is not a good practice to put website and video links as a reference, the project implementation mainly came through online internet content. Few of the links are -

- https://docs.djangoproject.com/
- https://www.django-rest-framework.org/
- https://github.com/krumbot/django_ems
- https://medium.com/aubergine-solutions/viewsets-in-django-rest-framework-25bb0110c210
- https://www.tutorialspoint.com/dbms/dbms_data_schemas.htm
- https://docs.djangoproject.com/en/3.1/topics/testing/overview/