Centre for Cloud Computing and Big Data

Cloud Based Evaluation Policies
BC_457_903_957_972
Project Report

Dhruva Kashyap Rakshith C Vikas Gowda Nikhil Ram

Abstract

This web Application is designed in order to ease the massive amounts of physical paper work involved in the process of evaluating Final-Year B.Tech Projects at the Computer Science Department.

Table of Contents

Introduction

We were tasked to build an evaluation system for the Computer Science Department at PES University. This evaluation system is designed to be used by faculty to evaluate Final Year projects as per the *evaluation procedure*.

Evaluation procedure

- Students are required to form teams and select a faculty member as a guide and register with the `project coordinator` of the Comp Sci. Department. If no guide is chosen, the project co-ordinator assigns one to them.
- Upon completion of registration of teams, the project coordinator will create **panels** and assign faculty members to each of these panels. The project coordinator will then assign one faculty member from each panel as the coordinator for a panel which we refer to as a `panel coordinator`.

- The co-ordinators of each panel are in charge of scheduling reviews for all the teams in their panel. They are also responsible for assigning faculty members to review temas for each review.
- The `evaluator` performs the job of grading each team assigned to them for every review

Structure of the solution

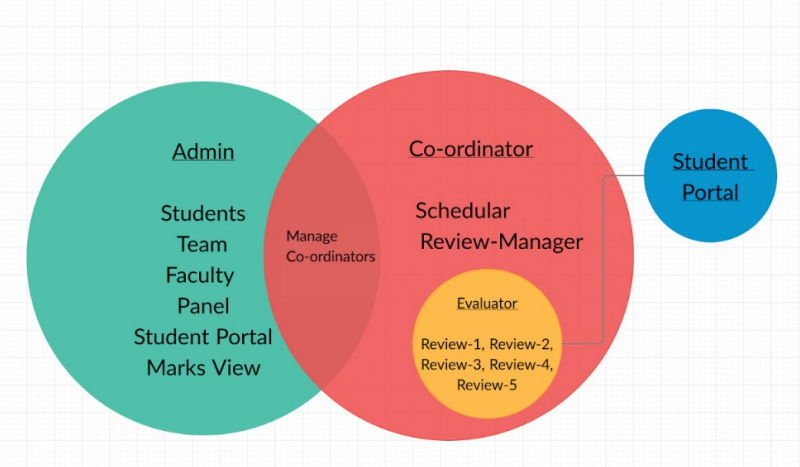

Figure 1: User Roles

1. The Application at its core is designed based on a hierarchy. Each user can access and modify the content only within the boundaries set up by their Role as discribed in Figure 1. The 3 Roles are Administrator, Coordinator and Evaluator
2. **Administrator**
    - The Administrator is incharge of inserting and maintaing the meta data for Students, Teams and Faculties.
    - The Administrator can initialise panels with faculties and teams in it.
    - The Administrator also has a few special permissions like opening/closing the Student Portal, Mail the Portal passwords to students and Download the Marks Summary for each student.
3. **Coordinator**
    - The Coordinator is incharge of setting up Project Reviews for all the teams their panel.
    - The Coordinator needs to assign a list of Faculties for each Team for each Project Review who shall evaluate the Team.
    - The coordinator must also set up open and close timings for each Project Review within which the Evalutars must evaluate the Teams.

4. **Evaluator**
    - The Evaluators' role is to simply evaluate all the teams assigned to them for each Project Review.
5. **Student Portal**
    - Students who are listed in the Web Application are given a "mini" user status. They can set up their **profile image**, project description as well as view the public comments provided by their reviewers.
    - This portal is opened by the Administrator.
6. **Other Salient Features**
    - The Interface provides in-built *E-Mail support* at various locations. eg:
        - The Coordinators can mail the Review Schedule to all the Evaluators in the panel.
        - Passwords are generated and emailed to all students to access the student portal
    - The Interface also provides *CSV Upload/Download* functionality at various Text Intensive Input/Output locations.
    - Evaluators can view profile images submitted by students.

Implementation

1. **Rest Based API**
    - The Back-End of the Web Application is a Rest Based API which interacts with the Front-End using JSON based requests and responses
2. **Security**
    - The Web Application Authenticates the user using a combination of **JWT tokens** and **Session Cookies**.
    - The *JWT Access Token* is used to authenticate the user to interact with the REST API and hence its validity is kept short.
    - The Access Token is refreshed periodically using a *Refresh Token* whose validity is substantially longer.
    - Refresh Token for the corresponding user is stored at the Back End and is accessed using the Session Cookie.
3. **Single Page Application**
    - Implementation of a **Single Page Application** using breadcrumb like structure with vanilla javascript.
    - Single Page Applications make web apps smoother to use when implemented correctly.
    - User Interface is designed with **Bootstrap v4.5**.
5. **Dockerized for cross-platform reliability**

- The Web Application is Dockerized for reliability across platforms
- To reduce hassle of installing third party libraries and easy deployment of the app,the app is dockerized into containers for REST API, Nginx, PostgreSQL

6. **Rest API incorporated with PostgreSQL and Django Web Framework**
    - The backend of the web application is developed using Django Framework.
    - PostgreSQL is the database engine used.
    - Django ORM is used to convert the relational data (postgres) to objects.

Requirements

Packages	Version	Details
PostgreSQL	12.4	Database Used With Django
Python	3.6	
psycopg2-binary	2.8.5	PostgreSQL API for python
Django	3.0.8	Python Framework
django-bleach	0.6.1	To Prevent XSS attcks
djangorestframework	3.11.0	REST API
djangorestframework-simplejwt	4.4.0	For JWT Tokens
Pillow	7.2.0	Image Processing
uWSGI	2.0.19.1	To Deploy Django With Nginx
Docker	19.03.6	
docker-compose	1.17.1	Combine Containers
Gunicorn	19.7.1	Python WSGI HTTP Server
Nginx	1.14.0	Load Balancer for HTTP server

More detailed list is available in the source code as `requirements.txt`

Run the following commands to start the server in the project directory

```
docker-compose up
```

The dockerfile is as follows

```
FROM python:3.6
EXPOSE 8000
ENV PYTHONUNBUFFERED 1
RUN mkdir /code
WORKDIR /code
ADD requrements.txt /code/
RUN pip install -r requirements.txt
ADD . /code/
LABEL creators="the NVRD team"
```

The Actual Website

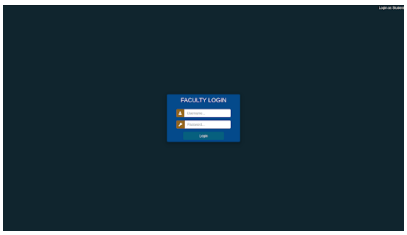

Figure 2: Faculty Login

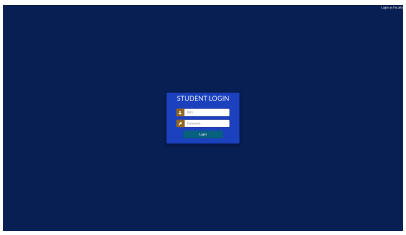

Figure 3: Student Login