Summer Fellowship Report

On

Connecting Yaksh and FOSSEE Workshop Site

Submitted by

Vivek Kumar

Under the guidance of

Prof. Prabhu Ramachandran, Department of Aerospace Engineering IIT Bombay

Prof. Kannan M. Moudgalya, Department of Chemical Engineering, IIT Bombay

June 20, 2020

Acknowledgment

It was a wonderful experience working as a FOSSEE Python remote intern from April 2020 to June 2020. I could discover my true potential as a professional. I believe, this exposure and lessons that I have learnt here will never come short for my career growth.

My task would not have been accomplished successfully without the contribution and collaboration of others.
My sincere gratitude:

To Prof. Kannan M. Moudgalya, Department of Chemical Engineering, IIT Bombay), the PI of FOSSEE, for providing us the opportunity to do an internship within the organization.

To Prof. Prabhu Ramchandran, Department of Aerospace Engineering, IIT Bombay our internship guide for his encouragement and guidance.

To the Senior Project Manager, FOSSEE, Mrs. Usha Viswanathan along with the FOSSEE team at IIT Bombay. With their patience and openness they created an enjoyable working environment.

To the Project mentor Mr. P. Aditya for believing in my capability to complete the task in time. His open-mindedness and out-of-box thinking encouraged me to look for better ideas for solutions. Without his help I couldn’t have easily corrected my mistakes and learnt things. His valuable time spent on teaching me things is unforgettable. I would also like to thank Mr. Ankit J and Mr. Akash Chavan for being there as a mentor everytime in need.

To my fellow colleague, Prathamesh Shiralkar with whom I have completed the fellowship. We experienced great things together and learnt a lot.

To all of you, I extend my deepest gratitude.

Contents

Chapter 1

Introduction

Yaksh is an Online Test Interface for creating various courses and conducting online programming quizzes. It supports various programming languages like :- C, C++, Python and simple Bash. Users can solve any questions by using these languages. Yaksh uses "test cases" to test the implementations of the students. It also supports simple multiple choice questions and le uploads so that users can easily submit their code.

FOSSEE workshop website is home for all workshops conducted in association with FOSSEE across the country. It facilitates creating of workshops by the instructor and the same can be approved by the admin and on success the workshop takes place.

Each workshop has tutorial files linked with it which the instructor uses during workshop hours. The thing that we wanted to improve on was to have a systematic approach of creating a course corresponding to each workshop. For each workshop there will be a corresponding json file for tutorials. We need a way to fetch all the upcoming workshop details and create respective courses.

1.1 Vision

An automated system to connect FOSSEE workshop and Yaksh website.

1.2 Approach

1. Create a Post request API handler on Yaksh website that accepts the formatted data and create new courses.

2. Test the above POST API using custom python script and JSON data.

3. Use a Scheduler service to automate the task to fetch data from Workshop site at a particular time everyday(12 PM) and create courses on Yaksh site through above API.

4. Create an intermediary app to run periodic tasks to fetch data from the Workshop site and create post requests to create new courses on Yaksh website

Chapter 2

Requirements and relevant Solutions

Requirements:

1. The first requirement was to have a POST API handler that can take the JSON formatted data and create a new course on Yaksh website from the given data.

2. The second requirement was to get the above JSON formatted date from the workshop site as API request.

3. Third requirement was to have an intermediary app where both of above things can happen seamlessly.

4. The fourth and most important requirement was to have a scheduler that can call the above task periodically, in order to accommodate the new workshop added on to the FOSSEE workshop site and create new courses correspondingly.

5. To cache the workshop data received so that we don’t necessarily create duplicate courses for a given workshop.

6. To Map Workshop user profile and Yaksh user profile.

Solutions:

1. Created POST API handler using Django Rest Framework(DRF) which accepts formatted JSON data and creates new course and returns the newly created course data along with 201 HTTP status code.

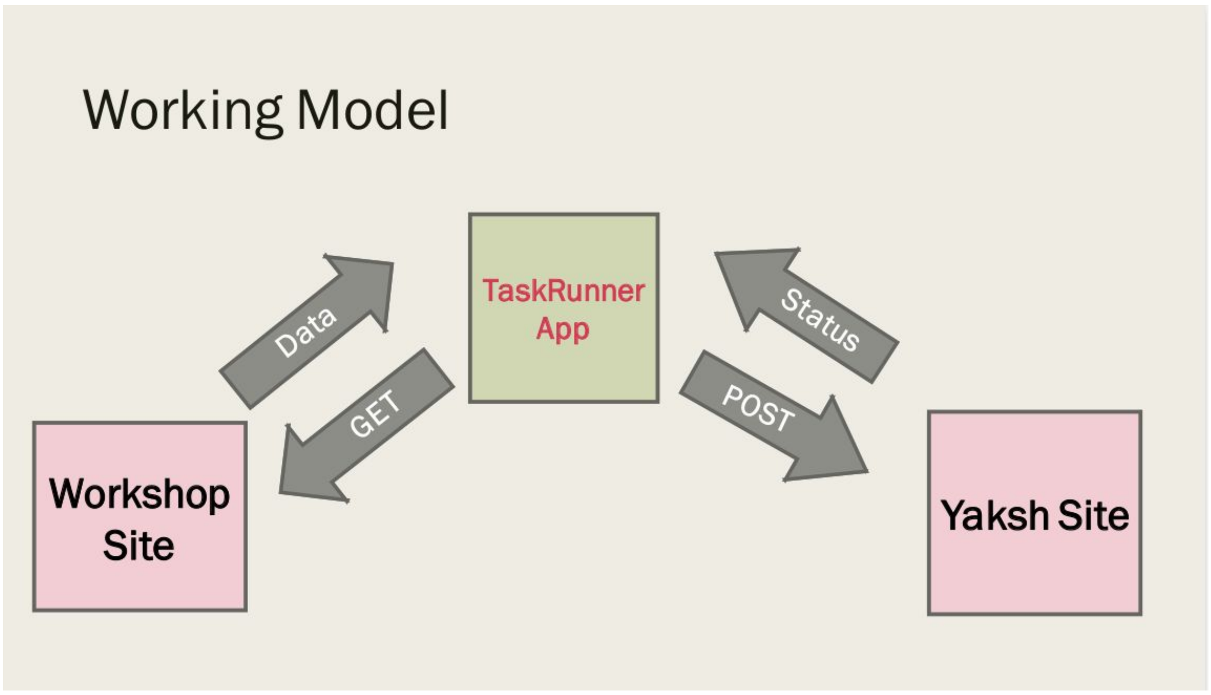
2. The second solution was covered by my fellow colleague Prathamesh Shiralkar, where I helped him with the required fields in the JSON formatted response.

3. The third blockhead was overcome by creating a new project “taskrunner”, a Django based project that with the help of “requests” library in Python, calls the above API end-points.

4. The fourth solution was suggested by the project mentor Mr. P. Aditya, we used “celery” library in Python that helps to schedule various tasks. We created a task that is scheduled to run at 12PM everyday. It first fetches the data from the FOSSEE workshop site and on successfully receiving the data, it creates a POST request to Yaksh API handler to create a new course based on the given data.

5. We used a separate model “WorkshopCached” to store the workshop id and the status, if the POST API call to Yaksh site goes well we update the status to 1(Success), else to 2. And everytime we receive data from FOSSEE workshop site we first check the status in WorkshopCached model and if present and status is 1, we continue with rest of the data.

6. This step was necessary as the username on both the platforms might not be the same for a given user, so we added a new model “UserMap”, that stores the information related to user name mapping across both the platforms.



Chapter 3

Implementation

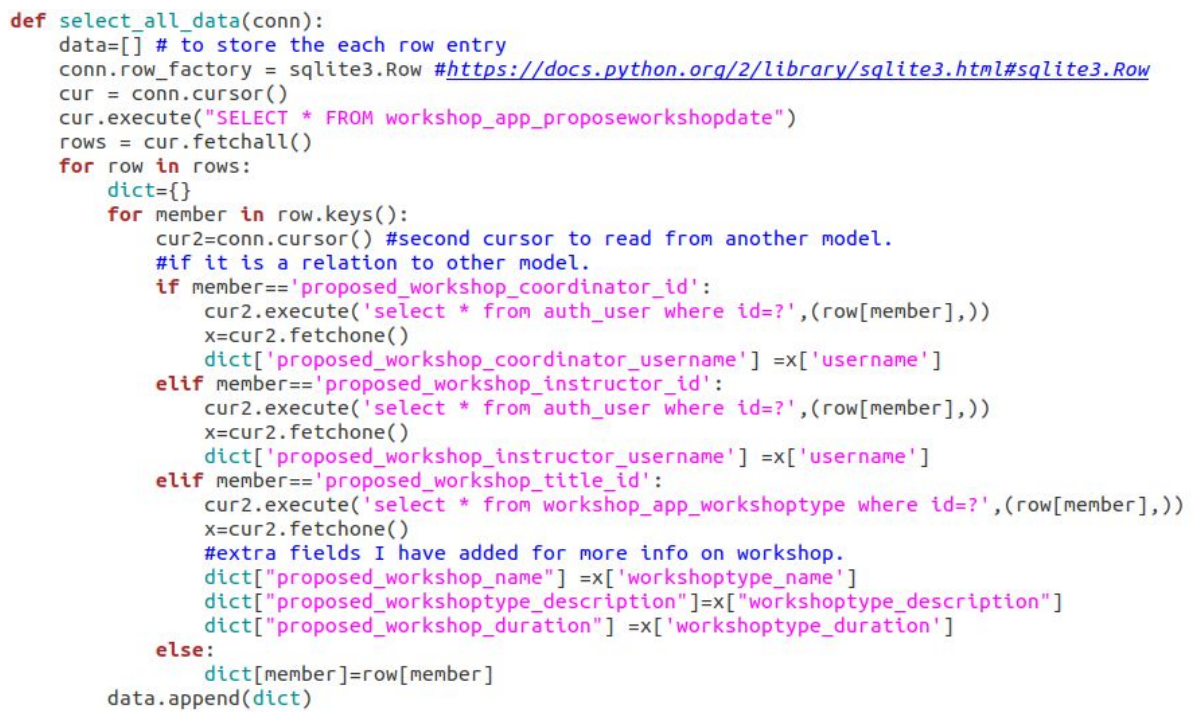
3.1 Used Technology

- Django
- Django Rest Framework
- Requests
- Celery
- JSON
- Postman
- Travis CI

3.2 First Phase

- The first phase of the implementation went into learning about interacting with Sqlite database with Python. Wrote a custom serializer that reads a Sqlite file and converts the data in Python Objects.

- This was necessary as we were about to create the POST API handler for the Yaksh website that receives the formatted JSON data and create a new course based on that.



```
def select_all_data(conn):
    data=[] # to store the each row entry
    conn.row_factory = sqlite3.Row #https://docs.python.org/2/library/sqlite3.html#sqlite3.Row
    cur = conn.cursor()
    cur.execute("SELECT * FROM workshop_app_proposeworkshopdate")
    rows = cur.fetchall()
    for row in rows:
        dict={}
        for member in row.keys():
            cur2=conn.cursor() #second cursor to read from another model.
            #if it is a relation to other model.
            if member=='proposed_workshop_coordinator_id':
                cur2.execute('select * from auth_user where id=?',(row[member],))
                x=cur2.fetchone()
                dict['proposed_workshop_coordinator_username'] =x['username']
            elif member=='proposed_workshop_instructor_id':
                cur2.execute('select * from auth_user where id=?',(row[member],))
                x=cur2.fetchone()
                dict['proposed_workshop_instructor_username'] =x['username']
            elif member=='proposed_workshop_title_id':
                cur2.execute('select * from workshop_app_workshoptype where id=?',(row[member],))
                x=cur2.fetchone()
                #extra fields I have added for more info on workshop.
                dict["proposed_workshop_name"] =x['workshoptype_name']
                dict["proposed_workshoptype_description"]=x["workshoptype_description"]
                dict["proposed_workshop_duration"] =x['workshoptype_duration']
            else:
                dict[member]=row[member]
        data.append(dict)
```

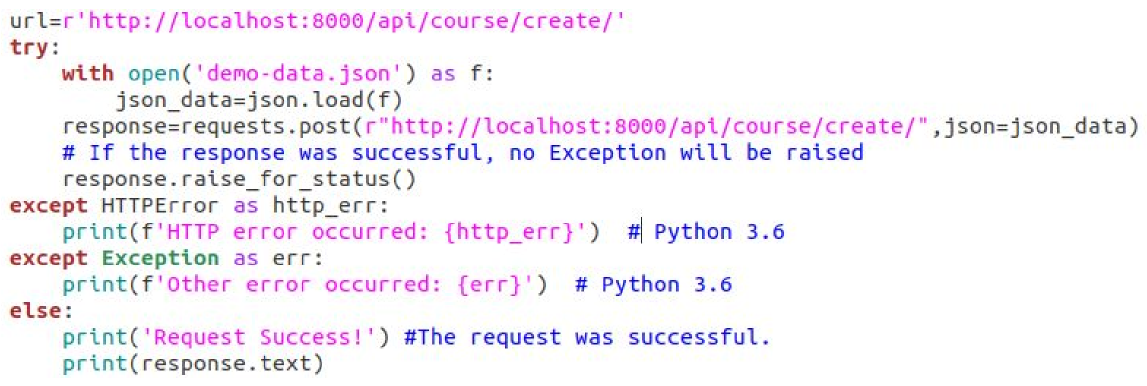
3.3 Second Phase

- In the second phase of the internship, I started working on creating the POST API handler for the Yaksh website, created serializers that serializes a Course into corresponding LearningModules, LearningUnits, Quiz and Lessons.

- Once the serializer part was over, I worked on creating a view that handles the POST request, gets the data and creates a new course and returns the newly created course data and HTTP 201 response code.

3.4 Third Phase

- Once the POST API handler was over, I wrote a custom Python script using requests and JSON module that tests the above functionality. This was a necessary step to ensure successful implementation of the API handler to create new courses.



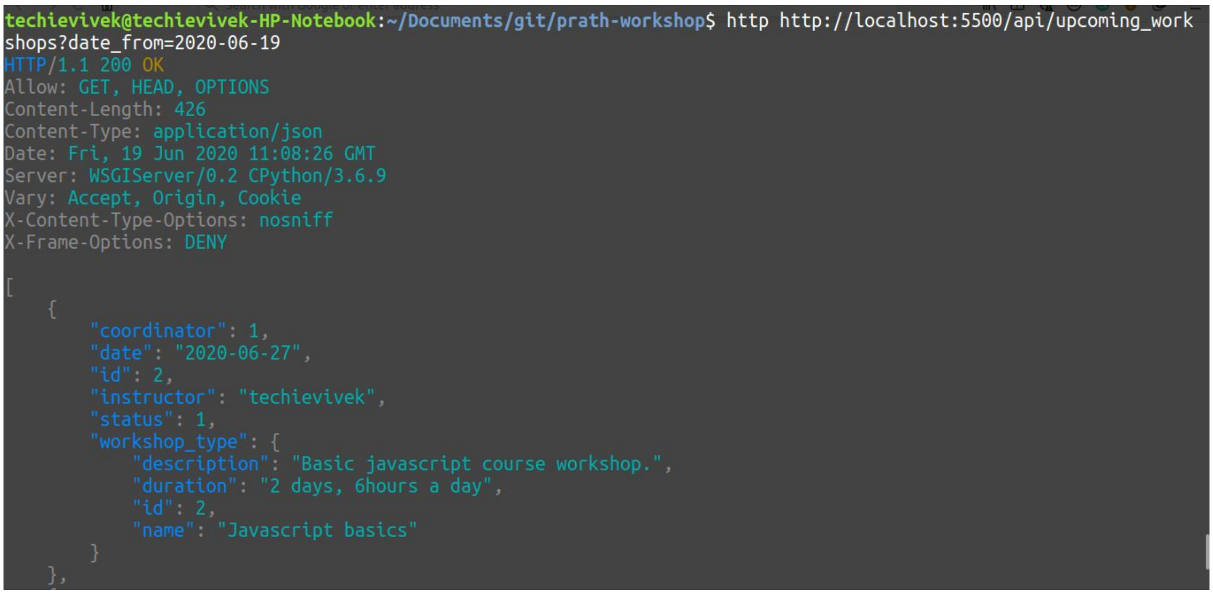
```
url=r'http://localhost:8000/api/course/create/'
try:
    with open('demo-data.json') as f:
        json_data=json.load(f)
    response=requests.post(r"http://localhost:8000/api/course/create/",json=json_data)
    # If the response was successful, no Exception will be raised
    response.raise_for_status()
except HTTPError as http_err:
    print(f'HTTP error occurred: {http_err}')  # Python 3.6
except Exception as err:
    print(f'Other error occurred: {err}')  # Python 3.6
else:
    print('Request Success!') #The request was successful.
    print(response.text)
```

- Tested the above API with Postman too once the Python Script check was successful.

3.5 Fourth Phase

- The Yaksh API part was final by this time, and now helped my colleague Prathamesh Shiralkar in creating the workshop API end-point to get new workshop data that will be used to create new courses on Yaksh site.

- Tested the above end-point with Postman before moving forward.



```
techievivek@techievivek-HP-Notebook:~/Documents/git/prath-workshop$ http http://localhost:5500/api/upcoming_work
shops?date_from=2020-06-19
HTTP/1.1 200 OK
Allow: GET, HEAD, OPTIONS
Content-Length: 426
Content-Type: application/json
Date: Fri, 19 Jun 2020 11:08:26 GMT
Server: WSGIServer/0.2 CPython/3.6.9
Vary: Accept, Origin, Cookie
X-Content-Type-Options: nosniff
X-Frame-Options: DENY

[
    {
        "coordinator": 1,
        "date": "2020-06-27",
        "id": 2,
        "instructor": "techievivek",
        "status": 1,
        "workshop_type": {
            "description": "Basic javascript course workshop.",
            "duration": "2 days, 6hours a day",
            "id": 2,
            "name": "Javascript basics"
        }
    },
```

3.6 Fifth Phase

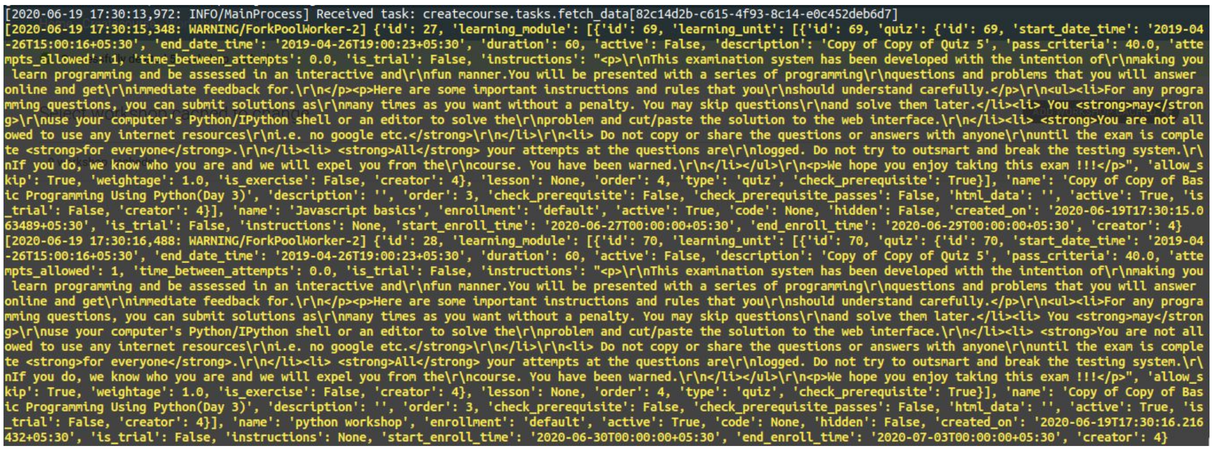
- Started working on a new Django App **taskrunner** from scratch to connectWorkshop website and Yaksh website.

- The requirement was to implement a task service that can be run periodically allowing fetching of new workshop data from the FOSSEE workshop site and creating POST requests on Yaksh API endpoint to create new courses.

- Used Celery module in Python to create periodic tasks.

- Added periodic tasks to retrieve all the accepted workshops and create corresponding courses onYaksh website.

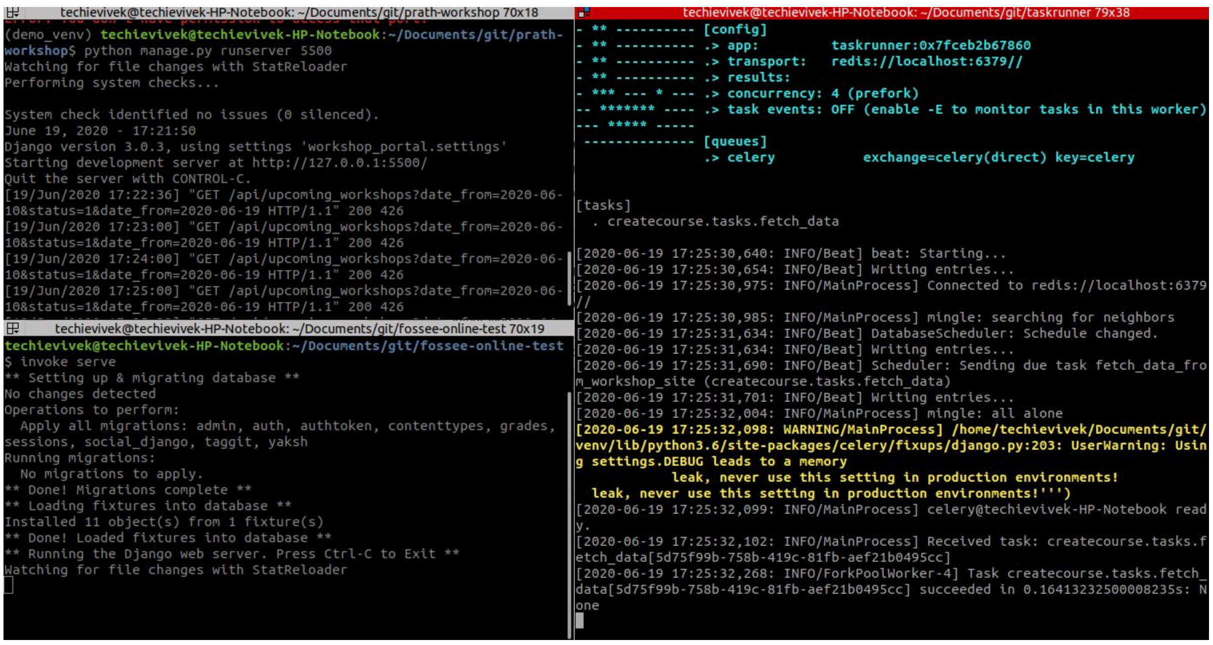
- Added two new models in the app to store the workshop id and the status, and a model to store username for both Workshop and Yaksh website.



```
[2020-06-19 17:30:13,972: INFO/MainProcess] Received task: createcourse.tasks.fetch_data[82c14d2b-c615-4f93-8c14-e0c452deb6d7]
[2020-06-19 17:30:15,348: WARNING/ForkPoolWorker-2] {'id': 27, 'learning_module': [{'id': 69, 'learning_unit': [{'id': 69, 'quiz': {'id': 69, 'start_date_time': '2019-04
-26T15:00:16+05:30', 'end_date_time': '2019-04-26T19:00:23+05:30', 'duration': 60, 'active': False, 'description': 'Copy of Copy of Quiz 5', 'pass_criteria': 40.0, 'atte
mpts_allowed': 1, 'time_between_attempts': 0.0, 'is_trial': False, 'instructions': "<p>\r\nThis examination system has been developed with the intention of\r\nmaking you
 learn programming and be assessed in an interactive and\r\nfun manner.You will be presented with a series of programming\r\nquestions and problems that you will answer
online and get\r\nimmediate feedback for.\r\n</p><p>Here are some important instructions and rules that you\r\nshould understand carefully.</p>\r\n<ul><li>For any progra
mming questions, you can submit solutions as\r\nmany times as you want without a penalty. You may skip questions\r\nand solve them later.</li><li> You <strong>may</stron
g>\r\nuse your computer's Python/IPython shell or an editor to solve the\r\nproblem and cut/paste the solution to the web interface.\r\n</li><li> <strong>You are not all
owed to use any internet resources\r\ni.e. no google etc.</strong>\r\n</li>\r\n<li> Do not copy or share the questions or answers with anyone\r\nuntil the exam is comple
te <strong>for everyone</strong>.\r\n</li><li> <strong>All</strong> your attempts at the questions are\r\nlogged. Do not try to outsmart and break the testing system.\r\
nIf you do, we know who you are and we will expel you from the\r\ncourse. You have been warned.\r\n</li></ul>\r\n<p>We hope you enjoy taking this exam !!!</p>", 'allow_s
kip': True, 'weightage': 1.0, 'is_exercise': False, 'creator': 4}, 'lesson': None, 'order': 4, 'type': 'quiz', 'check_prerequisite': True}], 'name': 'Copy of Copy of Bas
ic Programming Using Python(Day 3)', 'description': '', 'order': 3, 'check_prerequisite': False, 'check_prerequisite_passes': False, 'html_data': '', 'active': True, 'is
_trial': False, 'creator': 4}], 'name': 'Javascript basics', 'enrollment': 'default', 'active': True, 'code': None, 'hidden': False, 'created_on': '2020-06-19T17:30:15.0
63489+05:30', 'is_trial': False, 'instructions': None, 'start_enroll_time': '2020-06-27T00:00:00+05:30', 'end_enroll_time': '2020-06-29T00:00:00+05:30', 'creator': 4}
[2020-06-19 17:30:16,488: WARNING/ForkPoolWorker-2] {'id': 28, 'learning_module': [{'id': 70, 'learning_unit': [{'id': 70, 'quiz': {'id': 70, 'start_date_time': '2019-04
-26T15:00:16+05:30', 'end_date_time': '2019-04-26T19:00:23+05:30', 'duration': 60, 'active': False, 'description': 'Copy of Copy of Quiz 5', 'pass_criteria': 40.0, 'atte
mpts_allowed': 1, 'time_between_attempts': 0.0, 'is_trial': False, 'instructions': "<p>\r\nThis examination system has been developed with the intention of\r\nmaking you
 learn programming and be assessed in an interactive and\r\nfun manner.You will be presented with a series of programming\r\nquestions and problems that you will answer
online and get\r\nimmediate feedback for.\r\n</p><p>Here are some important instructions and rules that you\r\nshould understand carefully.</p>\r\n<ul><li>For any progra
mming questions, you can submit solutions as\r\nmany times as you want without a penalty. You may skip questions\r\nand solve them later.</li><li> You <strong>may</stron
g>\r\nuse your computer's Python/IPython shell or an editor to solve the\r\nproblem and cut/paste the solution to the web interface.\r\n</li><li> <strong>You are not all
owed to use any internet resources\r\ni.e. no google etc.</strong>\r\n</li>\r\n<li> Do not copy or share the questions or answers with anyone\r\nuntil the exam is comple
te <strong>for everyone</strong>.\r\n</li><li> <strong>All</strong> your attempts at the questions are\r\nlogged. Do not try to outsmart and break the testing system.\r\
nIf you do, we know who you are and we will expel you from the\r\ncourse. You have been warned.\r\n</li></ul>\r\n<p>We hope you enjoy taking this exam !!!</p>", 'allow_s
kip': True, 'weightage': 1.0, 'is_exercise': False, 'creator': 4}, 'lesson': None, 'order': 4, 'type': 'quiz', 'check_prerequisite': True}], 'name': 'Copy of Copy of Bas
ic Programming Using Python(Day 3)', 'description': '', 'order': 3, 'check_prerequisite': False, 'check_prerequisite_passes': False, 'html_data': '', 'active': True, 'is
_trial': False, 'creator': 4}], 'name': 'python workshop', 'enrollment': 'default', 'active': True, 'code': None, 'hidden': False, 'created_on': '2020-06-19T17:30:16.216
432+05:30', 'is_trial': False, 'instructions': None, 'start_enroll_time': '2020-06-30T00:00:00+05:30', 'end_enroll_time': '2020-07-03T00:00:00+05:30', 'creator': 4}
```

3.7 Final Phase

- Tested all the 3 services(Retrieve the new workshop data from the Workshop API, create POST request to Yaksh website, run the task periodically).



```
techievivek@techievivek-HP-Notebook: ~/Documents/git/prath-workshop 70x18
(demo_venv) techievivek@techievivek-HP-Notebook:~/Documents/git/prath-
workshop$ python manage.py runserver 5500
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
June 19, 2020 - 17:21:50
Django version 3.0.3, using settings 'workshop_portal.settings'
Starting development server at http://127.0.0.1:5500/
Quit the server with CONTROL-C.
[19/Jun/2020 17:22:36] "GET /api/upcoming_workshops?date_from=2020-06-
10&status=1&date_from=2020-06-19 HTTP/1.1" 200 426
[19/Jun/2020 17:23:00] "GET /api/upcoming_workshops?date_from=2020-06-
10&status=1&date_from=2020-06-19 HTTP/1.1" 200 426
[19/Jun/2020 17:24:00] "GET /api/upcoming_workshops?date_from=2020-06-
10&status=1&date_from=2020-06-19 HTTP/1.1" 200 426
[19/Jun/2020 17:25:00] "GET /api/upcoming_workshops?date_from=2020-06-
10&status=1&date_from=2020-06-19 HTTP/1.1" 200 426
```

```
techievivek@techievivek-HP-Notebook: ~/Documents/git/fossee-online-test 70x19
techievivek@techievivek-HP-Notebook:~/Documents/git/fossee-online-test
$ invoke serve
** Setting up & migrating database **
No changes detected
Operations to perform:
  Apply all migrations: admin, auth, authtoken, contenttypes, grades,
sessions, social_django, taggit, yaksh
Running migrations:
  No migrations to apply.
** Done! Migrations complete **
** Loading fixtures into database **
Installed 11 object(s) from 1 fixture(s)
** Done! Loaded fixtures into database **
** Running the Django web server. Press Ctrl-C to Exit **
Watching for file changes with StatReloader
```

```
techievivek@techievivek-HP-Notebook: ~/Documents/git/taskrunner 79x38
- ** ---------- [config]
- ** ---------- .> app:         taskrunner:0x7fceb2b67860
- ** ---------- .> transport:   redis://localhost:6379//
- ** ---------- .> results:
- *** --- * --- .> concurrency: 4 (prefork)
-- ******* ---- .> task events: OFF (enable -E to monitor tasks in this worker)
--- ***** -----
 -------------- [queues]
                .> celery           exchange=celery(direct) key=celery


[tasks]
  . createcourse.tasks.fetch_data

[2020-06-19 17:25:30,640: INFO/Beat] beat: Starting...
[2020-06-19 17:25:30,654: INFO/Beat] Writing entries...
[2020-06-19 17:25:30,975: INFO/MainProcess] Connected to redis://localhost:6379
//
[2020-06-19 17:25:30,985: INFO/MainProcess] mingle: searching for neighbors
[2020-06-19 17:25:31,634: INFO/Beat] DatabaseScheduler: Schedule changed.
[2020-06-19 17:25:31,634: INFO/Beat] Writing entries...
[2020-06-19 17:25:31,690: INFO/Beat] Scheduler: Sending due task fetch_data_fro
m_workshop_site (createcourse.tasks.fetch_data)
[2020-06-19 17:25:31,701: INFO/Beat] Writing entries...
[2020-06-19 17:25:32,004: INFO/MainProcess] mingle: all alone
[2020-06-19 17:25:32,098: WARNING/MainProcess] /home/techievivek/Documents/git/
venv/lib/python3.6/site-packages/celery/fixups/django.py:203: UserWarning: Usin
g settings.DEBUG leads to a memory
            leak, never use this setting in production environments!
  leak, never use this setting in production environments!''')
[2020-06-19 17:25:32,099: INFO/MainProcess] celery@techievivek-HP-Notebook read
y.
[2020-06-19 17:25:32,102: INFO/MainProcess] Received task: createcourse.tasks.f
etch_data[5d75f99b-758b-419c-81fb-aef21b0495cc]
[2020-06-19 17:25:32,268: INFO/ForkPoolWorker-4] Task createcourse.tasks.fetch_
data[5d75f99b-758b-419c-81fb-aef21b0495cc] succeeded in 0.16413232500008235s: N
one
```

- Wrote test cases for the Yaksh POST API to test new course creation features.

Further Improvements

Despite working heavily on making things right from every aspect, there are few things that can be improved further.

1. The Yaksh POST API end-point is open authenticated(no authentication is required to post data to the end-point), which can be improved for security and for this DRF default authentication, Token or OAuth can be used.

2. The Workshop GET API end-point is also open authenticated and data can be accessed by anyone with the URL, which can be protected similarly for better security.

Reference

Django Rest Framework- https://django-rest-framework.org/

Serializers in DRF-

https://www.django-rest-framework.org/api-guide/serializers/

Celery Documentation-

https://docs.celeryproject.org/en/stable/getting-started/introductio

n.html

Requests module- https://requests.readthedocs.io/en/master/