CmpE 352

Milestone II Report

Spring 2021

Group 8

Table of Contents

Executive Summary

a. Summary of the Project and Overall Status

In this project, we built an app about amateur sports. Our app has two parts: one for general usage, that is, a front end that is run by a browser and can be viewed, and a bunch of RESTful APIs that serve for basic operations. These basic operations are elements such as creating a new event, viewing a badge with its name etc. We deployed our app into AWS and used Docker Hub. You can reach it through this url. In this milestone report, everything we did to create our practice-app and our approach to the problems we faced is documented.

b. Deployed Application

We write a Dockerfile. Afterwards, we created our image with this file. You can find Dockerfile here. After creating our docker image we created a docker hub account to push our image in order to ease our deployment process.

In order to deploy our web application into a server we created a AWS EC2 instance. After creation and configuration of EC2 instance we connected into it through ssh. Deployment has been done through pulling of docker image from docker hub and running it. After deployment has been done configuring firewall rules in AWS so as to open our web application into the whole internet.

Application URL: http://52.15.179.49:8000/
API Wiki Documentation:
https://github.com/bounswe/2021SpringGroup8/wiki/Practice-App-Documentation

c. Functionality

Our project consists of independently working APIs that are created using the functions of pre-existing APIs. The APIs we created range from returning the exchange rates of currencies to calculating the distance between two places. We tried to make such APIs that will possibly be useful for our project of Purposeful Community. Thus, these APIs can be added to some certain communities for ease of use and they can also be utilized for more specific purposes. We mainly have 8 functional parts in the practice-app.

1. Register User : Takes name, surname, email and passwords and creates a user model than adds it to database.
   http://52.15.179.49:8000/registeruser

2. Search User: Only takes user name as parameter and searches across the database if such user exists.
3. Economy: Takes two parameters, from and to, and shows the latest exchange rate between those currencies. http://52.15.179.49:8000/economy
4. Literature: Takes first name and last name as a parameter of some famous person and returns his/her famous quotes. http://52.15.179.49:8000/literature
5. Cooking : It has 2 sub-functionality. It either returns a random meal recipe, or one specific meal recipe can be searched. http://52.15.179.49:8000/cooking
6. Create Event: It takes 3 arguments; title, description and city of the event and then saves it to the database. And then by using the given city name, it informs about the weather forecast for that city. http://52.15.179.49:8000/apis/create-event
7. Find Event Distance : It takes your country, city and county locations along with the event parameters and returns distance. http://52.15.179.49:8000/distance
8. Flag : Takes one parameter that codes for the country and returns the flag image of that country.

d. Challenges We Met

- Learning Django was challenging. Since it is a framework for web pages, we needed to integrate python script with html scripts. It was the first time for most of us to use the Django framework. So we had to first learn it and then implement our part of the project independently and merge them.

- Using Git for the store our code versions was another challenge. Even though we write our code separately, we had to merge our codes as one big project. Every time we sent pull requests and tried to merge our codes, there were conflicts between codes. Some of us had to cancel their pull requests, fix their codes and re-sent the pull requests. Again, using Git as Version Control was something new for some of us, so it became a challenging task to write part of a project separately.

- Writing HTML codes was challenging a bit for some of us since again it was new for some of us. Routing urls from one page to another sometimes becomes tricky and we get a lot of errors while designing it.

- Dockerization of the application was new and a bit complex to understand thus it required a lot of effort and time. After dockerization has been done and tested locally by some members of the group who are responsible for

dockerization, we pushed created and tested docker image into docker hub to be used in deployment of the application.

- Using environmental variables was especially challenging while trying to deploy our application with the use of docker. Docker wasn't able to read .env file and we were not able to solve the issue and decided for the first iteration to be deployed without a mechanism to store sensitive data securely. This was a major setback for the project.

- Creating an AWS EC2 instance and connecting into it through ssh was new and required a lot of work and effort. After the AWS EC2 instance was set up, we deployed our web application into the instance using docker image that we pulled from docker hub. After deployment has been done, we needed to configure firewall rules in AWS so as to open the port we use (8000) and the web application is allowed to be accessed by what whole internet traffic.

- Maybe the most challenging part was that we had to complete a project with 6 people if we consider there are 10 people groups. We were 8 people initially but 2 of us could not help us due to emergencies of their own. So it become troublesome for us to make the project to its deadline because in addition to all of these, the due date of another project and midterm of other courses overlapped with that project.

- Some of us were responsible for creating the home page and bringing different functionality of codes together. Since everybody did their function separately, it became tedious to bring code together. But we later overcome that problem by creating different HTML pages for all functions and then putting separate buttons for all of them.

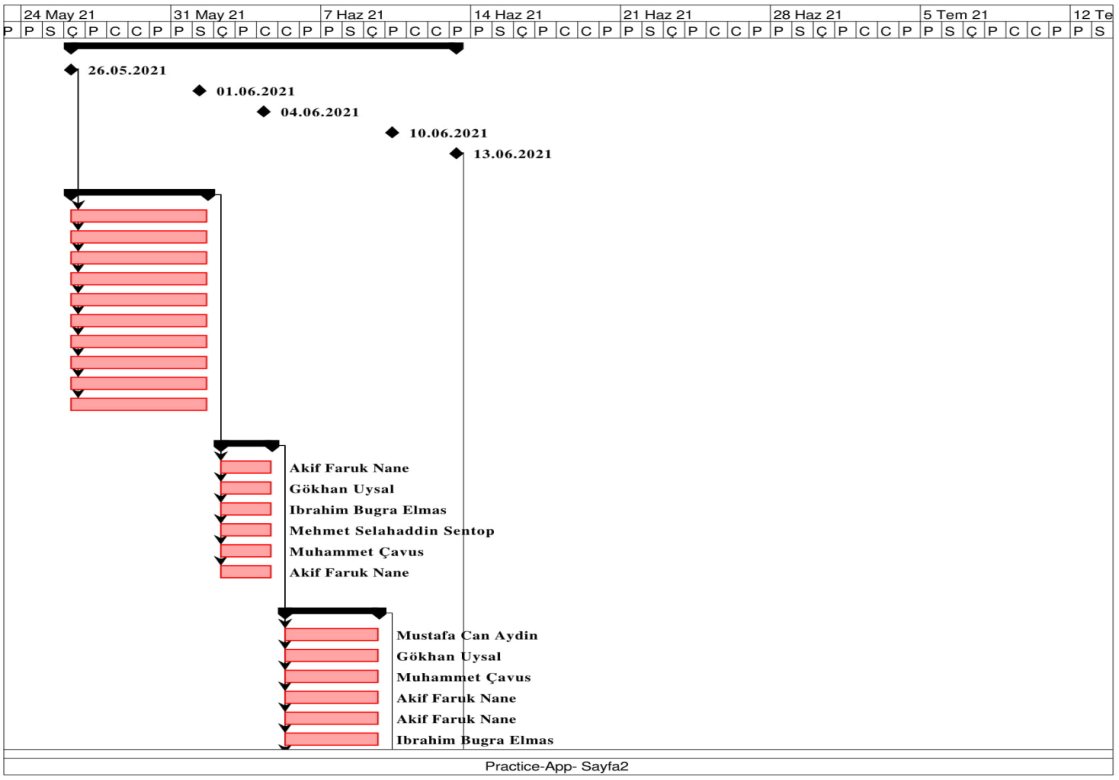
e. Status of Deliverables

Deliverable Name	Status	Delivery Date
API and Web application implementation	Success	10.06.2021
Dockerizing the application	Success	10.06.2021
Deployment of the application	Success	10.06.2021
Documentation of the application	Success	13.06.2021

Project Plan

		Ad	Süre	Balat	Bitirme	Ö...	Kaynak Adlar
1		**Meeting Notes**	**18 günler**	**26.05.2021 08:00**	**13.06.2021 08:00**		
2		Meeting1	0 günler	26.05.2021 08:00	26.05.2021 08:00		Mehmet Selahaddin Sentop;Ibrahim Bugra Elmas;Akif Faruk Nane;Muhammet Çavus;Güney ...
3		Meeting2	0 günler	01.06.2021 08:00	01.06.2021 08:00		Mehmet Selahaddin Sentop;Ibrahim Bugra Elmas;Akif Faruk Nane;Muhammet Çavus;Güney ...
4		Meeting3	0 günler	04.06.2021 08:00	04.06.2021 08:00		Mehmet Selahaddin Sentop;Ibrahim Bugra Elmas;Akif Faruk Nane;Muhammet Çavus;Gökha...
5		Meeting4	0 günler	10.06.2021 08:00	10.06.2021 08:00		Mehmet Selahaddin Sentop;Ibrahim Bugra Elmas;Akif Faruk Nane;Muhammet Çavus;Gökha...
6		Meeting5	0 günler	13.06.2021 08:00	13.06.2021 08:00		Mehmet Selahaddin Sentop;Ibrahim Bugra Elmas;Akif Faruk Nane;Muhammet Çavus;Gökha...
8		**Research about tools**	**7 günler**	**26.05.2021 08:00**	**01.06.2021 17:00**		**Mehmet Selahaddin Sentop;Ibrahim Bugra Elmas;Akif Faruk Nane;Muhammet ...**
9		Django	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
10		Postgresql	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
11		Docker	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
12		Git	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
13		Unittest	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
14		AWS	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
15		Django.test	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
16		Django.models & SQLlite3	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
17		Django.Forms	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
18		HTML	7 günler	26.05.2021 08:00	01.06.2021 17:00	2	
20		**Researching APIs**	**3 günler**	**02.06.2021 08:00**	**04.06.2021 17:00**		
21		Country Flags API	3 günler	02.06.2021 08:00	04.06.2021 17:00	8	Akif Faruk Nane
22		TheMealDB	3 günler	02.06.2021 08:00	04.06.2021 17:00	8	Gökhan Uysal
23		opencagedata	3 günler	02.06.2021 08:00	04.06.2021 17:00	8	Ibrahim Bugra Elmas
24		qutable.io	3 günler	02.06.2021 08:00	04.06.2021 17:00	8	Mehmet Selahaddin Sentop
25		openweathermap	3 günler	02.06.2021 08:00	04.06.2021 17:00	8	Muhammet Çavus
26		free.currconv	3 günler	02.06.2021 08:00	04.06.2021 17:00	8	Akif Faruk Nane
28		**Implementation start**	**5 günler**	**05.06.2021 08:00**	**09.06.2021 17:00**		
29		User	5 günler	05.06.2021 08:00	09.06.2021 17:00	20	Mustafa Can Aydin
30		Cooking	5 günler	05.06.2021 08:00	09.06.2021 17:00	20	Gökhan Uysal
31		Event	5 günler	05.06.2021 08:00	09.06.2021 17:00	20	Muhammet Çavus
32		Flag	5 günler	05.06.2021 08:00	09.06.2021 17:00	20	Akif Faruk Nane
33		Economy	5 günler	05.06.2021 08:00	09.06.2021 17:00	20	Akif Faruk Nane
34		Distance	5 günler	05.06.2021 08:00	09.06.2021 17:00	20	Ibrahim Bugra Elmas

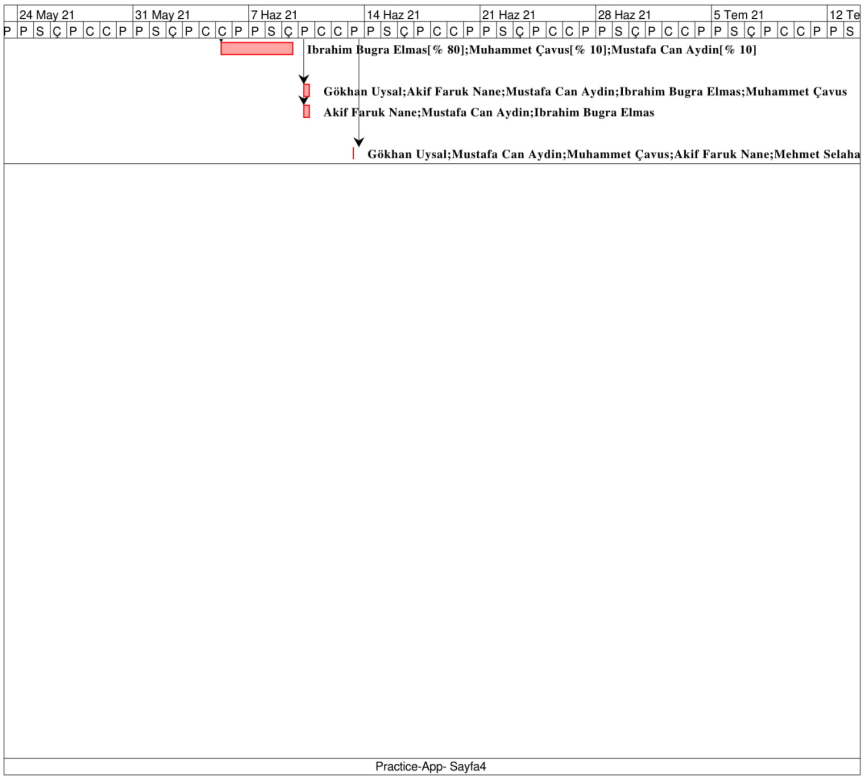
Practice-App- Sayfa1



Practice-App- Sayfa2

		Ad	Süre	Balat	Bitirme	Ö...	Kaynak Adlar
35		FrontEnd	5 günler	05.06.2021 08:00	09.06.2021 17:00	20	Ibrahim Bugra Elmas[% 80];Muhammet Çavus[% 10];Mustafa Can Aydin[% 10]
37		Merging branches and fixing bugs	1 gün	10.06.2021 08:00	10.06.2021 17:00	28	Gökhan Uysal;Akif Faruk Nane;Mustafa Can Aydin;Ibrahim Bugra Elmas;Muhammet Çavus
38		Dockerizing and AWS server	1 gün	10.06.2021 08:00	10.06.2021 17:00	28	Akif Faruk Nane;Mustafa Can Aydin;Ibrahim Bugra Elmas
40		Documenting Milestone 2 Report	0,2 günler?	13.06.2021 08:00	13.06.2021 09:36	6	Gökhan Uysal;Mustafa Can Aydin;Muhammet Çavus;Akif Faruk Nane;Mehmet Selahaddin S...

Practice-App- Sayfa3



Efforts of team members

MUSTAFA CAN AYDIN	• I created register user and search user api. • Contributed to enhancement of UI of the application by adding user register html files. • I created AWS EC2 instance and took part in deployment of the docker image to the server. • I helped Bugra and Faruk creating docker image and dockerizing the app through contributions to Dockerfile. • Attended meetings with TA. • Attended meetings with group members. • Reviewed other group members' api codes and make sure they are working and accepted some of pull requests. • Configured django and AWS firewall rules to allow web application to be accessed by

	whole internet. • Offered apis: • http://52.15.179.49:8000/apis/registeruser • http://52.15.179.49:8000/apis/searchuser?user_name=demo • • Worked on writing of deliverables report and documentation with Bugra and Faruk. • Contributed to the writing of M2 report, writing parts of tools & process, how to run application, parts of executive summary, and parts of challenges.
MUHAMMET ÇAVUŞ	• Milestone 2 individual report • I attended meetings with my teammates and TA • We exchanged some API suggestions between us. • I studied Django Framework from several channels and review some external Django examples to learn how to use Forms, Models etc. • I studied some basic HTML syntaxes to create web page for my responsibility of the project. The resulting page can be seen at create-event page • I used openweathermap api to offer to user weather forecast about his/her event location whenever he/she tries to create an event. • I decided to add created events to database. • After that I wrote a unittest to check if my api returns meaningful result. • Then, I sent a pull request and we reviewed all other pull requests together while my team mates merged them. • I reviewed getcurrency,getrandommealrecipe and register user web pages. • Utilized api : `api.openweathermap.org/data/2.5/weather?q={city name}&appid={API key}`
İBRAHİM BUĞRA ELMAS	• I attended all meetings with both the group and TA. • I created distance api which takes two addresses and returns the distance between them in km's. • I wrote most of the UI using django and html. I receive suggestions about enhancements

	from my teammates. ● I merged many pull requests. While preapring files to merge and solve conflicts, I reviwed APIs of others make some suggestions regarding functionality. Also, I got help from my teammates when solving conflicts. ● I created several branches and pull requests for my api and ui design ● I helped Faruk and Mustafa to dockerize and deploy our app to AWS. ● Created the google docs document for the submission, and contributed to writing of documentation ● Offered Api: ● http://52.15.179.49:8000/apis/finddistance?country_from=turkey&city_from=ankara&county_from=yenimahalle&country_to=turkey&city_to=istanbul&county_to=besiktas ● Name:Distance Api ● Utilized Api: ● https://api.opencagedata.com/geocode/v1/json?key= ● Name:Geocode Api
AKİF FARUK NANE	● - Had a meeting with TA two times. Attended both of them. We had a fun conversation. ● - Had many meetings with my group, all of which I attended. To determine which framework we are going to use, to dockerize the app etc. ● - I created the base project of the application with Django, created two apps, apis and ui. Then, I pushed it to GitHub. ● - I dockerized the app and registered for the docker with the id of assignmentm2. ● - I helped and guide many people in this process. Such as how to use django and how to create URLs etc. ● - I created two apis, "flag" and "currency" APIs. I used API of "https://www.countryflags.io/" and "https://free.currconv.com/api/v7/convert". ● - We reviewed conflicts of many PRs and resolve them. ● - Deployed 4 times the Docker image and fixed the bugs! ● - We prepared the deliverables report with

	Buğra and Mustafa. ● - I did 3 code reviews [1](https://github.com/bounswe/2021SpringGroup8/pull/48), [2](https://github.com/bounswe/2021SpringGroup8/pull/47), [3](https://github.com/bounswe/2021SpringGroup8/pull/46). ● - I contributed to the tools & processes part of the group report M2. ● - Cheers!
MEHMET SELAHADDİN ŞENTOP	● Attended all of team meetings except the last one. ● Had two meetings with TA in which we got feedback of our previous work and also asked questions about next task. ● Studied the new concepts and development environments. ● I created the api “quote” which recieves a famous persons’ name as input and returns a quote from them. ● I created a branch to work on my api. ● I reviewed the works of other team members. ● I commented on some issues and created one for myself. ● Due to the unfortunate schedule of my homeworks, I have been unable to attend the last part of this task. My thanks to the team members.
GÖKHAN UYSAL	● - Had a meeting with TA two times. Could attend one of them. ● - I attended all meetings with team except1. ● - I searched https://www.themealdb.com/api.php api and learned how to use it. ● -I created two apis, “getrandommealrecipe” and “getmealrecipebyname” APIs. ● -I reviewed some of the other team members apis and comment some of them. ● -I open more than 2 pull request ● -At last day we reviewed the general code and fixed some bugs and I make a pull request for the last version of django project. ● -Utilized search by name and random meal recipe functionalities of api:https://www.themealdb.com/api.php ● - I tried to help other team members about

	git usage. ● - I helped the dockerizing the app at the start but I couldn't help it because I had an interview at this time. ● - I created the meal recipe part of API Documentation in the wiki. ● - I contribute the deliverable submission report. ● -I contribute to the milestone2 group report. ● -I write my individual report. ● -I made the project plan with using project libre here.

Tools & Process

We utilized a few different tools and libraries, one of which is mainly Django. Django is a framework to create websites or apis basically. We had a meeting to determine which web framework that we are going to use for this application and django was exactly what we needed. It was a good choice. We benefited Django to make a simple user interface website and an API tool. We created many different APIs and while doing so, there were a few libraries required.

We used PILLOW library for downloading the flag images to a memory stream, then we converted it to base64 string and sent it as a resulting image. We used requests and urllib library to send requests to 3rd APIs to get weather data, currency exchange info, flag images etc... We also used json for the returning values of some functionalities. We learned HTML language to create a user-interface demo. These were challenging for us since some of us didn't know about html and django at all.

We needed a database to register events and users. Since we didn't communicate well enough due to extreme occupation of other assignments, two members of the team used different sql types. Then, we had to resolve this issue. We decided to use SQLite3 as a database. We used django.models to create tables & instances. This was a critical mistake honestly.

Docker is also a main tool that we utilized in our web application. We used Docker to ease deployment. Some of us also tried Docker in our local systems, these are written in personal effort pages. After the main application is dockerized, its docker

image has been deployed into docker hub to ease deployment into AWS EC2 instance.

AWS has been used to deploy our web application to a server. AWS EC2 instance has been created for deployment and Docker image of our web application has been pulled from docker hub and deployed into our EC2 instance. After deployment has been done, firewall rules of AWS EC2 instance have been configured to make the web application public and accessible for everyone.

We used GitHub as a project management system. We mainly utilized Branches & Pull Requests to overcome the complexity of collaboration. We resolved most of the conflicts with the team together. Some of us reviewed the codes as well. GitHub management stage was successful.

Code

https://github.com/bounswe/2021SpringGroup8.git

How to run web-application using Docker

In the directory Dockerfile located(practice-app/assignmentm2) running below command will create a docker image with repository name of assignment2/assignment2:

```
docker build . -t assignment2/assignment2
```

After the docker image created one can run the application at port 8000 with the command :

```
docker run -d -p 8000:8000 assignment2/assignment2
```