1. Executive Summary

1.1. Summary of Project and Overall Status

- For this practice-app, we decided to build an app which mainly focuses on the financial data presentation to the users. Besides, there are additional functionalities not only related to finance but could be needed in any web app. To sum up, we can say that the project consists of two parts of functionalities: the first is the financial APIs and the second is the general-usage APIs. Stating the overal status of our project, even though Docker works successfuly in our local, we could not solve the problem while we were trying to deploy our app into AWS despite our team members' long time of try. Therefore, we cannot provide an AWS URL for our deployed app but there is an alternative we mention below.

1.2. URL of our deployed app

- https://2021springgroup11-1.emindeniz99.repl.co/ We deployed our application to replit.com, and It uses a PostgreSQL DB at another server. We could not deploy our application to AWS successfully.

1.3. API URL(documented)

- https://github.com/bounswe/2021SpringGroup11/wiki/Practice-App-Documentation
- Logo API Readme.md

1.4. Basic Functionalities

- Even though our functionalities works independently by using various APIs, as we mentioned above in the summary of project section, they meet in a common goal, which is to serve as a financial data app. Consequently, we have 5 different functionalities can be used by users.
    - a. /ETFSector: Gets ETF symbol from the user and returns the sector exposure data. At the same time, saves this to database.
    - b. /FXRates?
    - c. /Stocks: Gets stock prices of 8 companies in the US market for the given period.
    - d. /Spam: checks if email is spam or not and append the status to the list.
    - e. /CryptoAPI: Gets crypto parities, shows graphical past data. It doesn't exist.
    - f. /logosearch: Gets the logo from the public api or db, change or add new logo to db.

1.5. Challenges we met as a group

- In general, learning and implementing new concepts, libraries, programming languages in some of which we did not have so much experience before was quitely challenging. Specifically,

dockerization and deployment of the app was one of the most confusing and challenging parts so that we could not achieve to deploy it. The detailed comments including the challenges we met about the tools we use is in the section 4. In addition to the technical difficulties, another challenge was the unexpected contribution rates of the group members. Considering that we were assigned as 9 members at the beginning, we tried to complete this practice-app as 6 people.

1.6. List of status of deliverables

- Practice-app Implementation -Success
- Dockerizing the app -Success
- Deployment of the app -Failure
- Documentation of the app -Success

2. Project Plan

PDF and pod of the pdf file can be found in zip file.

3. Summary of Work Done

Team Member	Summary of Work Done
Alper Canberk Balcı	Attended meetings, researched about financial API's, decided on crypto parities. Failed to implement due to technical difficulties, couldn't install dotenv. Helped in milestone2.
Bekir Tatar	Made research about some public APIs. Learned Django and Django REST Framework from theirs own websites Django, Django REST Framework. Implemented the stocks API using tutorials on the these websites. Made research about Docker containers and AWS. Reviewed Fxrates and spam API and the pull request about Dockerize the application. Deployed the docker containers of application to AWS ECS and they seems to be running without any problem in the cluster but application does not respond due to an unknown reason. Attended all the meetings. Contributed the reports.
Eren Altunoğlu	
Fikri Cem Yılmaz	Beside my communicator duties. Researched the public API's to find suitable one to use GET and POST methods. Clearbit is the suitable one. Then followed the tutorial on this link. After implementing proccesses, helped my teammates on code reviews and tried to dockerize and deploy apps in the meetings. The resulting API documentation can be found on this URL and my individual report.

Team Member	Summary of Work Done
Mehmet Yasin Şeremet	Attended all meetings. Researched about the API and proper usage of it. Created ETFSector API. Created pull request related to this. Reviewed codes of some of our group members. I wrote my API documentation in here. Contributed to the practice-app submission deliverable. Contributed to the Milestone 2 report. Completed my own Milestone 2 individual report.
Muhammed Emin Deniz	In addition to participating the meetings, I done my tasks. Firstly, I tried to dockerize our application and shared my researchs with my team at next meeting. Afterthat, I search free and good APIs. I developed my "spam" API with using https://eva.pingutil.com/. I got reviews and improve my works. I also tried to give feedback and review to my teammates. I also made contribution our reports and wikis.
Selman Berk Özkurt	Initiated the finance application bundle with its intended functionality and API provider. Made the Forex price API (called FXrates here). Prepared the working templates for the front end and the Django server. Involved in the overall architecture for the finance bundle.

4. Evaluation of Tools and Processes

BackEnd

We've made researches and decided Django could be a good choice. Django is a free and open source web framework and this qualities played a big part in our decision, other than easy to learn of course. Most of us followed this link to learn Django and implement our app.

FrontEnd

In the beginning, we've decided to use React as our front end. While it seems still a good choice for our general app, it was a bit unnecessary and time consuming for this project. So, we've decided abandon React and move to Django views. Django views is simple yet powerful choice. Django views uses basic HTML and handles the requests by itself. It may change in the big project, but it was very helpful in the mock project.

DataBase

Database decision is maybe the worst call of this project. We haven't used Postgres' any advantage, we could easily done it with SQLite and maybe then, uploading it to AWS would be much easier. While using Docker,two image has been made one for Django and one for Postgres. Without Postgres we would only have one image and maybe we can handle the uploading process easier.

Management

For the project Management we used Github. Its code management, issue management and version control make the Github perfect for project management. We've used different branches and implement our practice-app simultaneously. We've created issues (not enough, still progressing) to record the requirements and development road of this project. These records are very helpful to others who will develop same kind projects or future ourselves. Version controlling is a good way to see the progress and improve efficency.

Communication

Communication plays big part in project management. A team project can not succeed without communication. So, we've used Slack and Microsoft Teams. Both of them did their job just fine. Teams has no time limit but a bit harder when inviting someone outside from team. So, not the perfect choice but still very good. On the other hand Slack gets more mixed comments, some of the team members find itself very useful, while others think it has problems on its notification systems and sometimes it is hard to reach someone while it is emergency.