Bilkent University

Department of Computer Engineering

CS 491: Senior Design Project I

Fall 2021

High Level Design Report

Project Name: *Laber*

Group Members:

Yiğit Gürses	21702746	yigit.gurses@ug.bilkent.edu.tr
Melisa Taşpınar	21803668	melisa.taspinar@ug.bilkent.edu.tr
Onur Oruç	21702381	onur.oruc@ug.bilkent.edu.tr
Emin Adem Buran	21703279	adem.buran@ug.bilkent.edu.tr
Mustafa Hakan Kara	21703317	m.kara@ug.bilkent.edu.tr

Supervisor: Assoc. Prof. Can Alkan

Innovation Expert: Ahmet Eren Başak

Jury Members: Assoc. Prof. Can Alkan

Asst. Prof. Dr. Shervin Arashloo

Asst. Prof. Dr. Hamdi Dibeklioğlu

Project Webpage: https://eminademburan.github.io/laber/

Table of Contents

1 Introduction

1.1 Purpose of the System

With the coming of the information age and rise of social media, there is an increasing quantity of information available on the internet that can be taken advantage of [1]. Companies try to take advantage of this using natural language processing and human analysts but they both come with some disadvantages. Machine learning models try to approximate human expertise and always come with a margin of error [2], while processing a large amount of data with human experts can be very costly. In order to reduce this cost, crowdsourcing platforms such as Amazon's Mturk [3] can be used but such unspecialized platforms require expertise and investment on the side of the client. For these reasons, there exists a niche to be filled by specialized platforms that are easy to use for both the clients and human experts.

Laber will be a mobile based platform that aims to provide real time social media analysis by human experts. Clients will be able to crowdsource their work with minimal knowledge of the system and have access to a pool of human experts at all times. The obtained analytics will be available to the clients through our website. The experts will be able to do all of their work through a mobile application allowing them to work remotely and efficiently. Various features will be implemented to ensure the reliability of our experts and provide them the optimal environment to do their work. Gamification techniques will be utilized to incentivize regular work schedules.

1.2 Design Goals

1.2.1 Usability

- Clients should be able to create tasks and see the analytics through a browser without any technical knowledge.
- Experts should be able to do all of their work through our mobile application without any technical knowledge.
- Experts should be able to communicate with each other well using the discussion threads and voice chats. To maintain the quality of the audio communication as well as the usability of the voice chat, there should not be any large delays or loss of data.

- The application should be efficient and run smoothly to not interrupt the workflow of experts.

1.2.2 Marketability

- Human experts are incentivized to use our app through monetary compensation and gamification.
- *Laber* provides the Clients real time access to social media analysis through human experts, with minimal cost.

1.2.3 Cost

- All services will be free of charge except the payments of human experts, which will be provided by the clients.

1.2.4 Reliability

- There should not be any failures that cause loss of work or payments.
- The application should run smoothly to not interrupt Experts' workflow.

1.2.5 Scalability

- The back end should be implemented keeping in mind the fact that the data to be handled can become arbitrarily large.

1.2.6 Efficiency

- The application should be efficient enough to run on low end mobile devices.
- The back end should be efficient in terms of memory and processing complexity in order to minimize the costs associated with servers.

1.2.7 Security

- SMS confirmation will be required for human experts when signing up, in an effort to eliminate bots and malicious actions.
- There should not be any bugs or security flows that may cause the leakage of sensitive information.

1.2.8 Extendibility

- The implementation should allow the addition of new task types. These might include image/audio transcription, text completion and other services that might be compatible with our crowdsourcing infrastructure.

1.2.9 Portability

- The system should support various mobile devices and web browsers.
- The mobile application should be designed to be easily ported to different operating systems.

1.3 Definitions, Acronyms, and Abbreviations

- Client: Companies and institutions that will be specifying tasks to be completed by Experts using our website.
- Expert: Human experts that will be evaluating social media posts based on specified labels using our mobile application.
- Label: A metric based on which Experts will be evaluating social media posts. These will be specified by Clients.
- Project: Set of specifications, set by a Client, that determines how and when the tasks will be created and distributed.
- Task: A unit of work to be completed by an Expert.
- Voice Chat: A service that allows Experts to discuss in a group voice call, in real time, regarding a task.

1.4 Overview

In this report, we first provide a description of our current software architecture. Then, we elaborate further on the architecture we are proposing, specifically on subsystem decomposition, hardware/software mapping, persistent data management, access control and security, global software control and boundary conditions. Afterwards, we provide a section for subsystem services. Towards the end of the report, before providing a glossary and our references, we talk about other analysis elements such as the consideration of various factors in engineering design, and contributing and functioning effectively on the team. In the latter,

we focus on what we have done to help create a collaborative and inclusive environment, take the lead role and share leadership on the team.

2 Current Software Architecture

The functionality of the software architecture we made is similar to the Amazon Mechanical Turk (MTurk) that has been implemented so far. However, the application we will make has a more specialized software architecture in terms of its purposes compared to MTurk. Since the *Laber* includes a mobile application and a website, we used different technologies and languages for both parties. Android Studio and Java have been used for the mobile application, and HTML, CSS, Bootstrap and Javascript were used for the web part. To implement the server, we started to use Java to ensure compatibility between mobile application and server side.

3 Proposed Software Architecture

3.1 Overview

There will be two types of users in the system. Clients will be creating projects and will expect to see the results of the expert evaluations through a web browser. Experts will be working on a mobile device and do the labeling on the tasks given to them by the system. These two will interact with a central server to pull and push the necessary data. Experts will also be communicating with each other through voice chat from time to time. Our server will be responsible for web scraping to generate new tasks, distributing the tasks to active Experts, evaluating Experts' answers via cross checking, giving the Experts ratings and rewards based on their performance, and finally, storing all the necessary data to later be shown to the Client when requested.

3.2 Subsystem decomposition

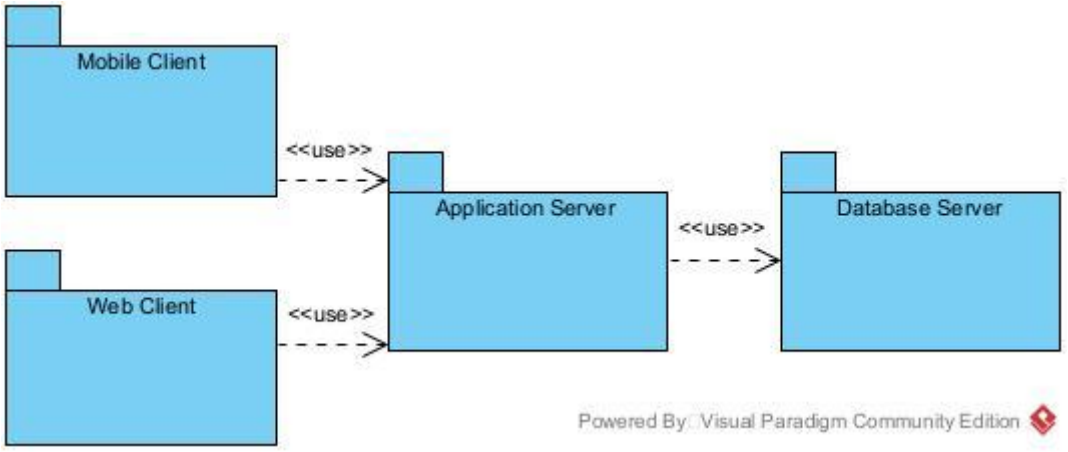

Figure 1: Subsystem Decomposition

The system, in the most abstract level, consists of three subsystems. Mobile client is the subsystem that contains the Experts' use cases and the functionality of the mobile application they will be using. In turn, the web client contains the Clients' use cases and the functionality of the website they will access through a browser. The server contains the central application logic and the database where all the necessary persistent data will be stored. In this client-server architecture, the clients may interact directly with the database to pull-push data, or for more computationally intensive use cases, make a request to the application logic part and receive the results of the server-side computations. The details of each subsystem is expanded upon the subsystem services section.

3.3 Hardware/software mapping

Experts will be using a mobile device to access our mobile application. The mobile device may have an Android or iOS operating system. Since we aim to make the application as accessible as possible, the software should be compatible with old releases of these operating systems. Also, it should not be resource intensive in order to accommodate low end devices and low bandwidths. The application will connect to the application server on our host machine which may then request data from the database server in our host machine as necessary.

Clients will be accessing our website through a browser. The user machine, in this case, might be various kinds of devices including PC and mobile phones. For this reason, the interface should be responsive to the dimensions of the screen, and also to the kind of device the user connected from. Other than that, the client will be connected to the same server on

the same host machine as the mobile application. However, it will be making different kinds of requests.

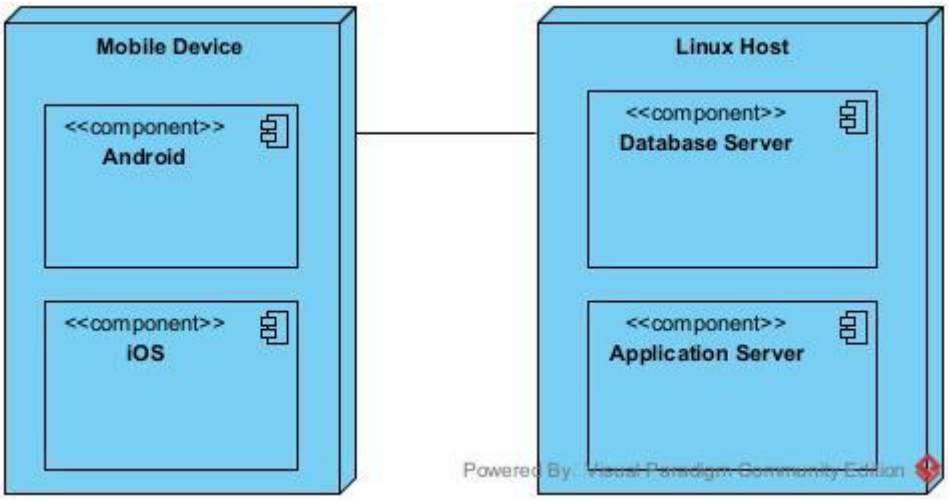

Figure 2: Mobile Client Mapping

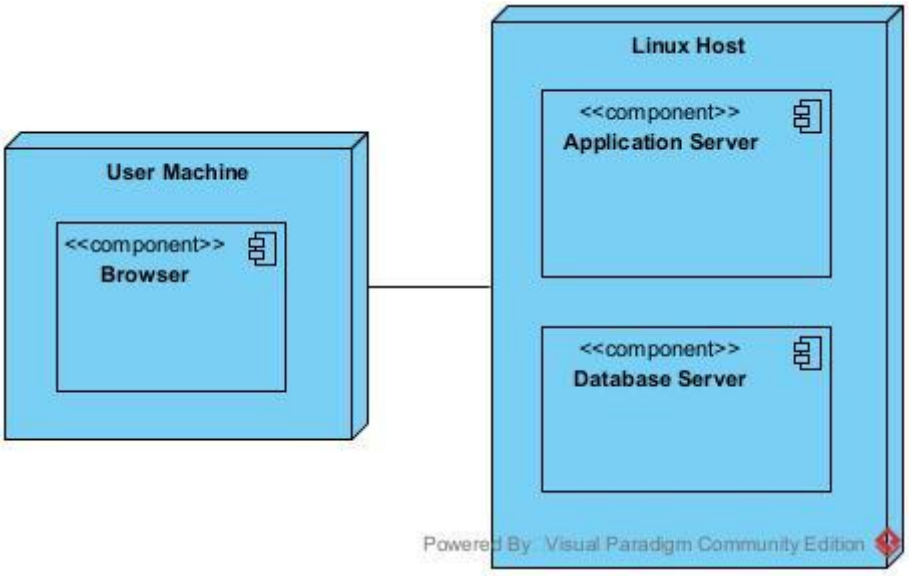

Figure 3: Web Client Mapping

3.4 Persistent data management

Loss of the work of Experts can be costly since they are to be paid by the amount of tasks they complete. For this reason, when the app is closing, any completed tasks in a session should be posted to the database and half completed tasks should be saved locally so the Expert can continue from where they left off. User information should also be persistent after a user logs in, until they log out. This part of the persistent data management will be done using the local file system of the user machine.

Other than this, there is a big amount of data that will need to be stored in the database permanently. Namely, user information, account balances, project information, tasks and expert answers and timestamped logs of user actions will need to be stored. To be extra cautious, it is possible to add a backup database to the system.

3.5 Access control and security

In order to minimize the risk of people abusing the system with multiple accounts, we will require an sms confirmation to register and log-in to the mobile application. For the web services, email authentication will be used for registration and password authentication will be used for log-in.

Since Expert accounts will be tied to phone numbers, it will be possible to directly ban the phone number in case a user takes suspicious actions.

Clients will only be able to see the projects they created, and will not be able to access the analytics for other Clients' projects.

Experts can only access the tasks they are assigned by the system. During a voice-chat session, only sound data will be transmitted to the other participants. A participant will be able to close their microphone at any time.

3.6 Global software control

In both the mobile and web applications, event-driven control will be used. The events will be triggered by users when they interact with the user interface (e.g., when they label a post) and the data in the server will be updated according to the actions taken by the user. Since there is no certain time order that can be predicted for events that are occuring, events will be handled asynchronously in the server. Moreover, event-driven is convenient as it allows considerably fast responses to events to be handled.

3.7 Boundary Conditions

3.7.1 Initialization

People will need an Internet connection while using the application. When a new expert signs up for the app, he or she will undergo training on how to use the app.
Since the server client model will also be used in the structure of the application, the server must be started before the clients and be ready for future requests coming from clients.

3.7.2 Termination

When a user closes the application, the application terminates. If a user attempts to quit the application in the middle of a labeling process, the current status of the task will be stored and shown to the user when the application starts running again.

3.7.3 Failure

If the connection is lost unexpectedly, a failure will occur and the user's progress on a task, if any, will be lost and required to be done from scratch.
Since there will be discussion threads and voice chat features in the application, the servers must be robust in order to exchange information between the clients. That is, the server should be able to handle incoming requests and continue to work in case of an exception.

4 Subsystem Services

4.1 Mobile Client Subsystem

4.1.1 Task Manager Service

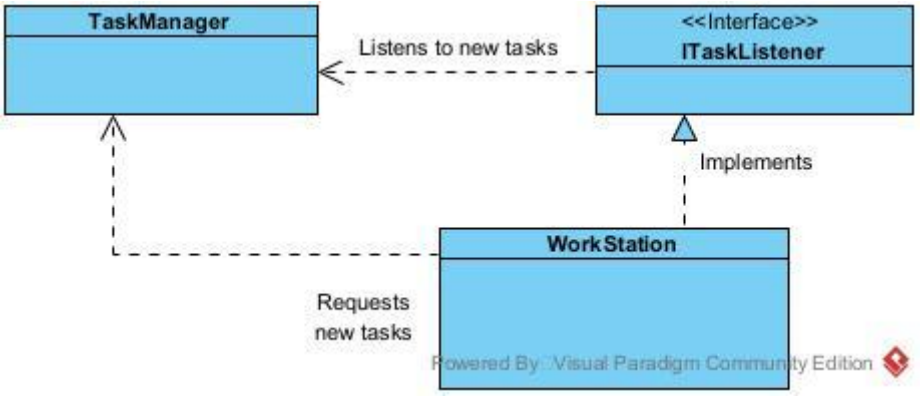

Figure 4: Workstation-TaskManager interaction

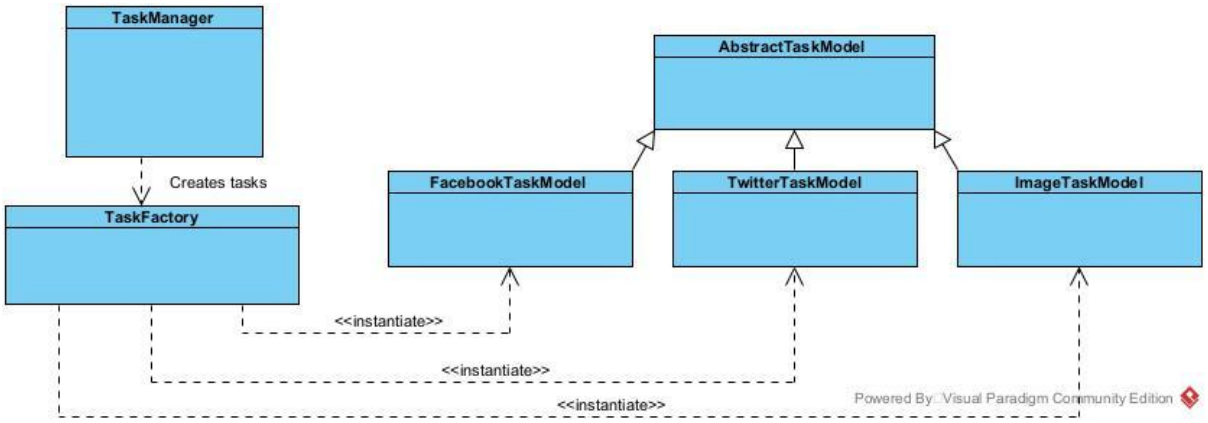

Figure 5: Task Model Creation

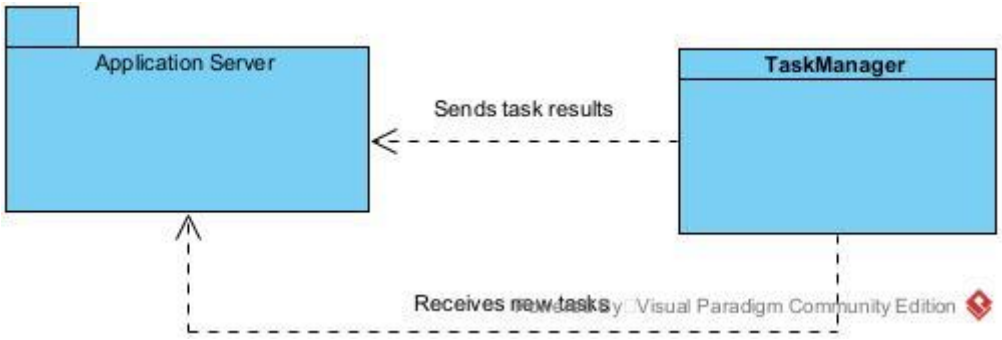

Figure 6: TaskManager-Server interaction

Task manager service requests new tasks from the application server subsystem when needed. When it receives the task data, it creates a task model using the TaskFactory class and passes the model to the Workstation Service. TaskManager will also take the Expert answers from the workstation and send them to the Application Server subsystem.

4.1.2 Workstation Service

Figure 7: Workstation receives the answers from the Expert

The Workstation service will receive tasks from the TaskManager. It will then create a view to show the necessary information to the expert. It will then take the answers of the expert and pass it back to the TaskManager as a ResultSet.

4.1.3 Voice Chat Service

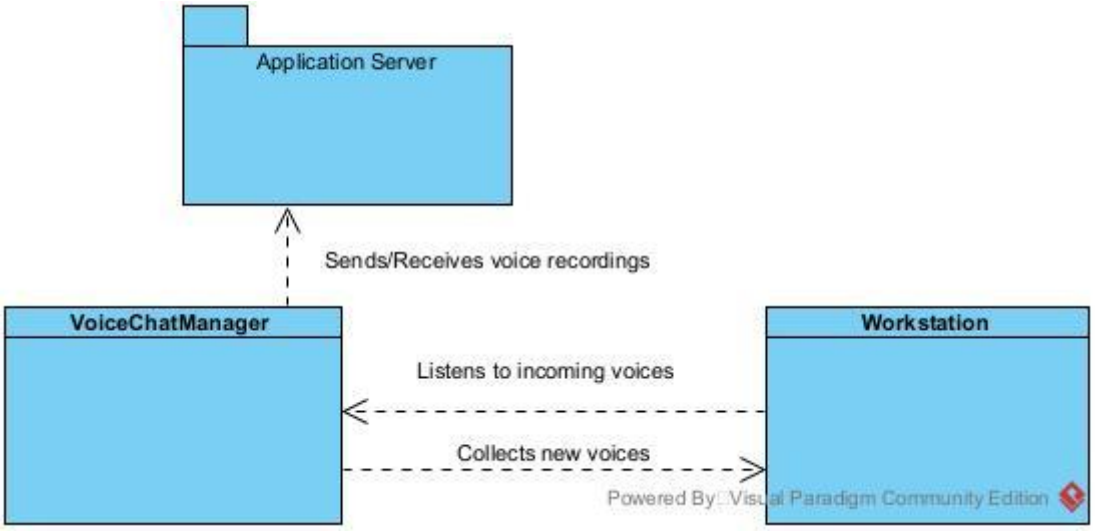

Figure 8: VoiceChatManager interactions with WorkStation and Server

Voice chat service provides an environment for Experts to achieve a quick consensus when their answers conflict dıring the cross check service within the application server subsystem.

4.1.4 Friends Service

Experts can add new friends via sending requests to their phone numbers. They can also view their friends list and the usernames and ratings of their friends.

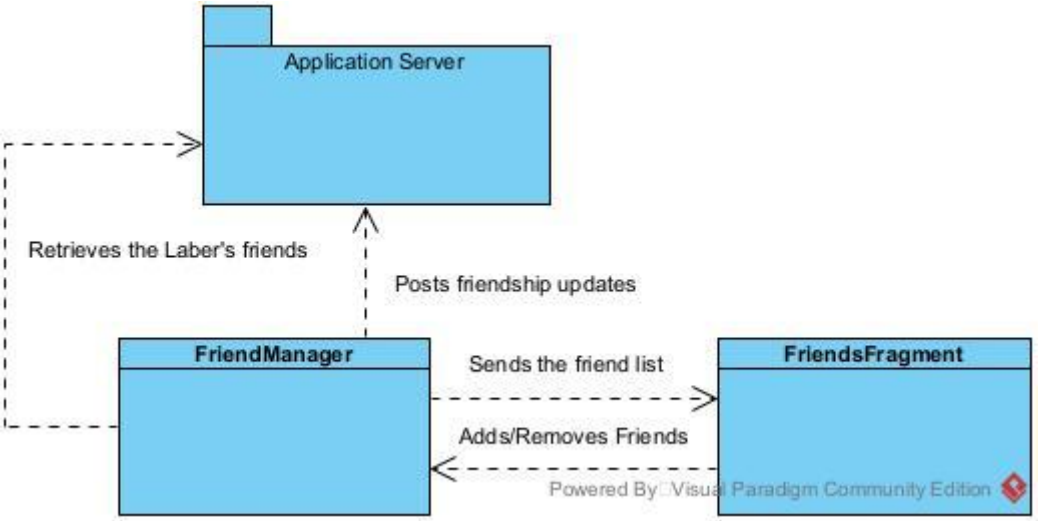

Figure 9: Friends Service Diagram

4.2.5 User Manager Service

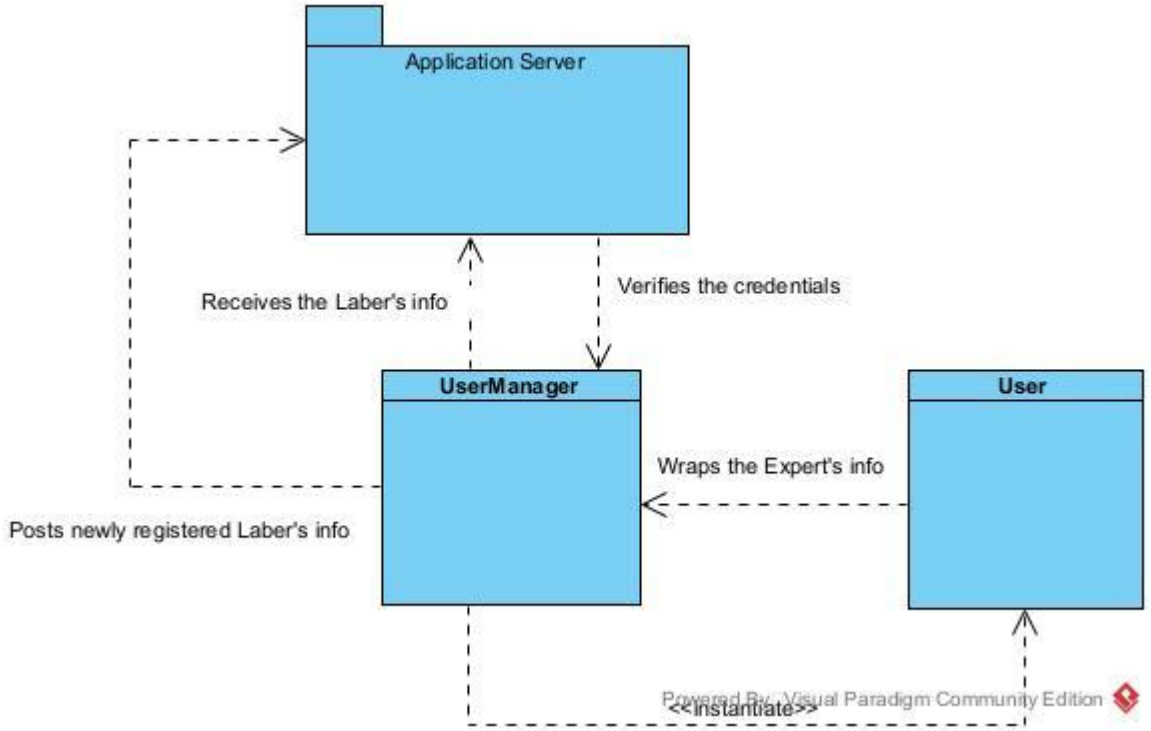

Figure 10: User Manager Service Diagram

User session info will be wrapped in this class to be used in queries. Registration and log-in authentication will also be handled by this service.

4.1.6 Balance Manager Service

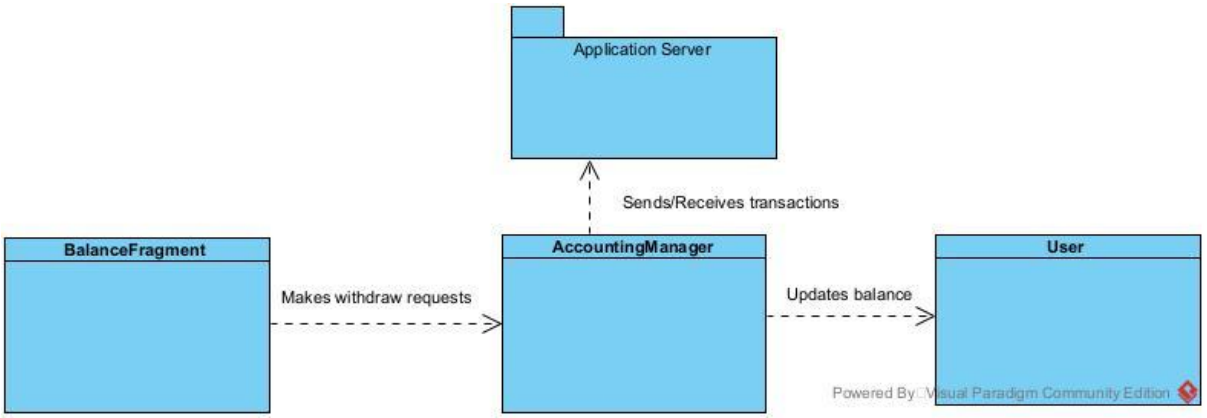

Figure 11: Balance Manager Service Diagram

Experts can view their current balance and make withdraw requests.

4.2 Web Client Subsystem

4.2.1 Balance Manager Service

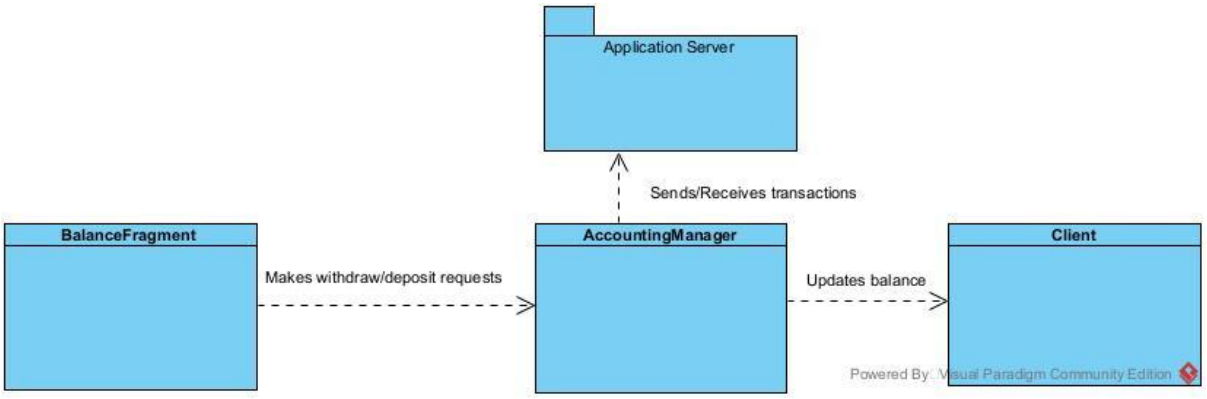

Figure 12: Balance Manager Service Diagram

The current balance is pulled from the database to be shown to the client. The client can deposit or withdraw arbitrary amounts of currency using this service.

4.2.2 Project Viewer Service

Clients can view a list of their existing projects and track their progress. They can select a project to be edited via the project manager service, or view the details of a selected project. They can also see the analytics of a project via the plotting service.

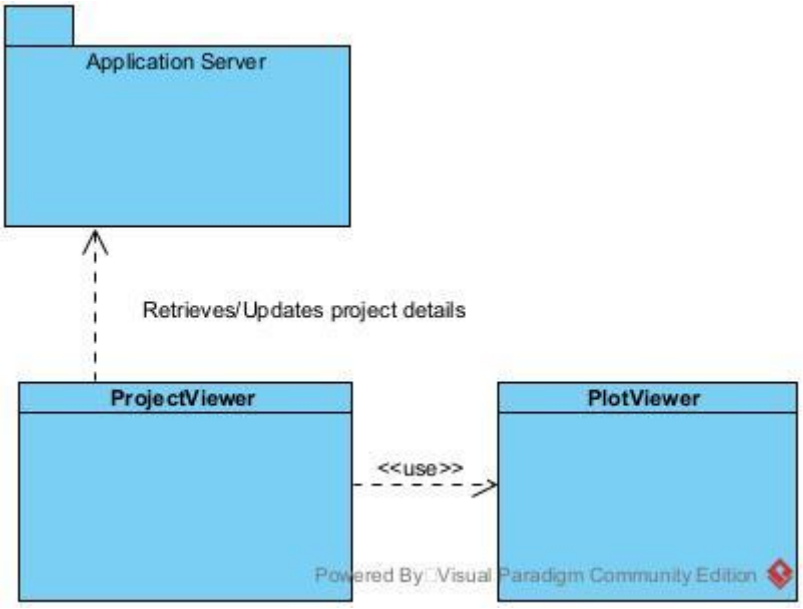

Figure 13: Balance Manager Service Diagram

4.2.3 Project Manager Service

The client can create new projects or edit existing ones using this service. The changes will be posted to the database and will be reflected on the tasks that will be created and put in the task queue.

4.2.1 Plotting Service

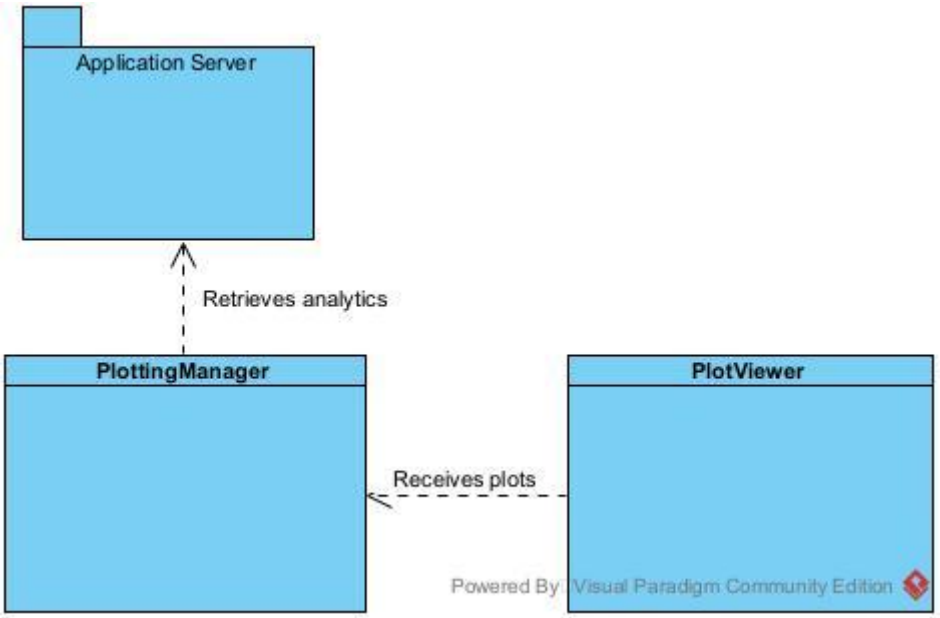

Figure 14: Balance Manager Service Diagram

This service requests data from the analytics service of the application server subsystem. It then draws plots from the given data to be shown to the Client.

4.2.5 User Manager Service

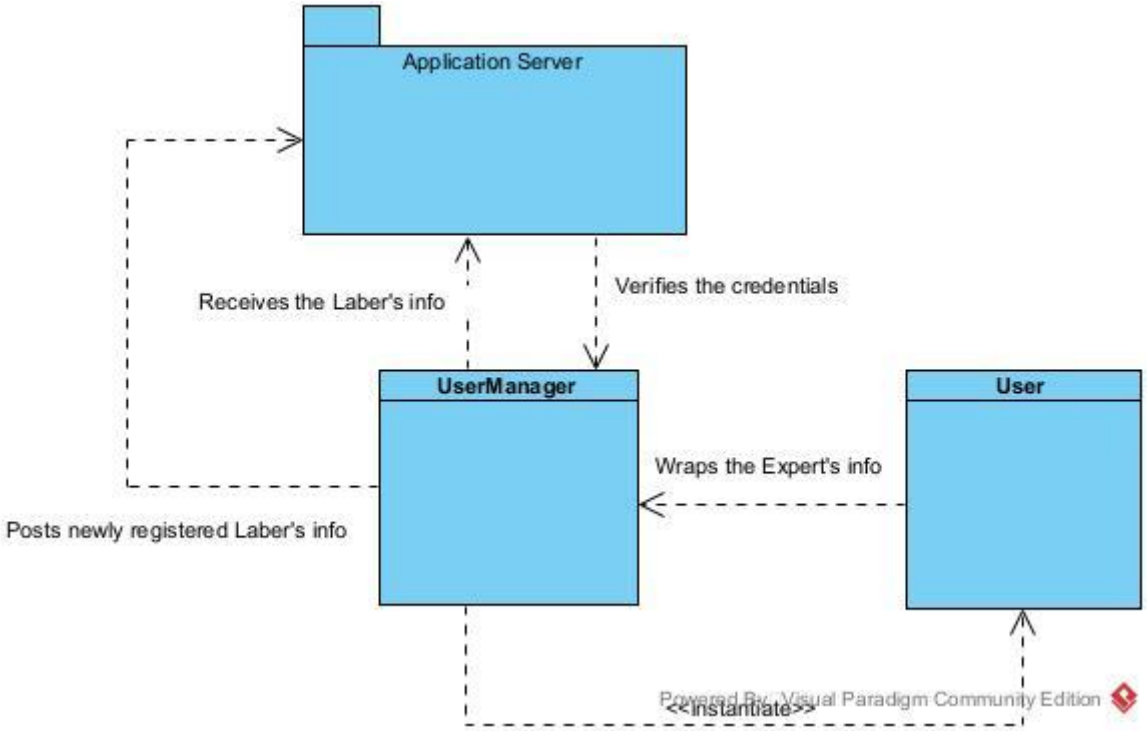

Figure 15: Balance Manager Service Diagram

User session info will be wrapped in this class to be used in queries. Registration and log-in authentication will also be handled by this service.

4.3 Application Server Subsystem

4.3.1 Distribution Service

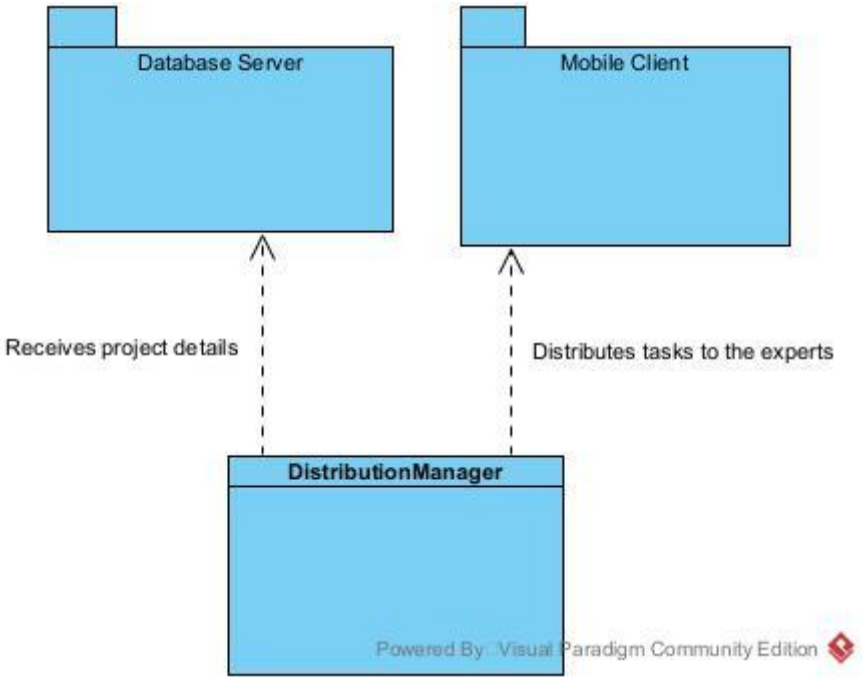

Figure 16: Balance Manager Service Diagram

The tasks in the task queue will be distributed to the active Experts by this service. This distribution will take into account Expert ratings and each task will be given to multiple Experts for cross-checking.

4.3.2 Scraping Service

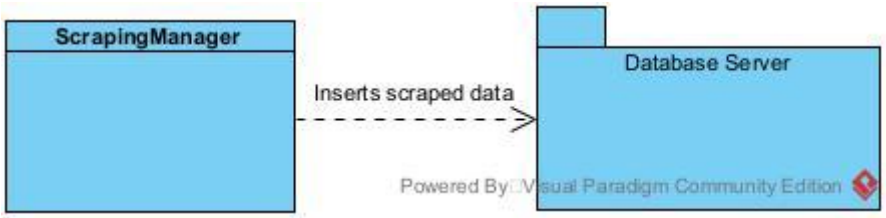

Figure 17: Balance Manager Service Diagram

This class will scrape social media posts in real time to create new tasks and push them to the task queue.

4.3.3 Payment Manager Service

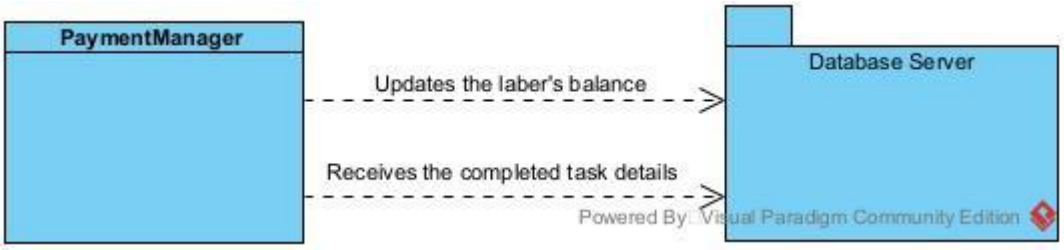

Figure 18: Payment Manager Service Diagram

This class will determine the payment for each completed task based on the Experts' ratings, cross-check results and the ratio of active Experts to the number of tasks waiting in the task queue. These payments will be logged in the database and also added to the total balance of the Experts.

4.3.4 Rating Manager Service

This service updates the rating of Experts when they complete a task. The calculation of ratings mainly depends on cross-check results and mock tasks with known answers. During a cross check, an Expert that gives an answer similar to another Expert with a high rating will have their rating increased. If the system notices random or low quality answers from an Expert, their rating will decrease.

4.3.5 Cross Check Service

A task will be given to multiple Experts if possible as mentioned in the distribution service section. This service takes a weighted average of their answers and will generate the approximate answer for the task. All answers will still be stored in the database if a Client wants to access them. However, for analytics services, the approximated answers from these cross-checks will be used.

4.3.6 Analytics Service

When a Client wants to see the analytics of one of their projects, this service will query the necessary data from the database and do the necessary computations on the server side. The results will be returned to the web client subsystem to be shown to the user. The results will also be stored in the database and flagged as valid. If any

tasks are completed for a given project, its previously computed analytics will be set to invalid and will have to be computed again on the next request.

4.3.7 Ban Manager Service

This service will track suspicious activities and user reports to decide when to ban an Expert. Banned phone numbers will be stored in the database and they will be prevented from registering new accounts or logging into their existing account.

4.3.8 Dataset Generation Service

If a client wants to access the logs corresponding to their projects (all completed tasks with timestamps and the expert information with it) they will be able to request it from the web client. This service queries the necessary data, generates a csv file containing it, compresses the file and sends it to the Client.

4.4 Database Server Subsystem

4.4.1 Backup Service

Periodically backs up the database on low traffic hours.

4.4.2 Storage Service

Database server will store all the necessary persistent data. Namely:

- Client information: client id, username, password hash, e-mail, balance, projects
- Expert information: expert id, username, phone number, balance, friends, isbanned
- Project information: project id, client id, project name, project settings
- Task queue: task id, project id, task data (texts, images etc.), source, timestamp
- Completed task logs: task id, expert id, answer, payment, timestamp
- Analytics: project id, analytics data, timestamp, isvalid

etc.

4.4.2 Query Service

Receives queries from the Application Server subsystem and returns its results.

5 Consideration of Various Factors in Engineering Design

During the analysis phase of our project, we have had to consider several factors relevant to engineering design.

- Public Health: Our approach allows people to work as much as they want when they want. These, at first glance, might sound like positive qualities and they can be. However, they can also have downsides. Extensive use of our application can disrupt a user's work life balance and can be detrimental to their mental and physical health. For this reason, in order to encourage a regular use of our application, we will be providing bonus payment for a predetermined amount of work each day. Excessive work done after the user surpasses this predetermined amount will not yield any bonus payment. We hope that this will discourage users from working in huge chunks of time. We might also consider sending the users messages that recommend taking a break after working for a long period.
- Public Safety: People that work on our application are also likely to work on other platforms that require frequent car-driving such as Uber or DoorDash. Along with that, many of our users will be driving a car daily to go to the places they need. If one of our users decide to work on our application while driving a car at the same time, this will create a safety concern for both the user and the pedestrians and drivers in the area. For this reason, we thought of using GPS to limit the user's maximum speed to be eligible to work on our application. However, with that approach, we prohibit users on buses and trains to work and that would be an unwanted consequence. Instead we decided to make the users sign an agreement form that requires them to not use our application while driving a car or doing any other dangerous activity that requires uninterrupted focus.
- Public Welfare: Our application can provide work opportunities to people, only requiring a mobile phone and internet access as an investment. Therefore, it can be used by people in need as an easy income source. However, there are a couple of considerations regarding this point. Firstly, some people may have access to only a very low end mobile phone with an older operating system. We need to make sure our application can run on such devices if we want to allow these people to work on our application. For this reason, we need to write efficient code that does not use

unnecessary resources. The application should also not be heavy on the battery usage. Secondly, the application will need to exchange data with our servers very frequently. Some users will be paying their internet providers based on the amount of data they download and upload. If our application exchanges any unnecessary data, it might be unprofitable to work on our application for many users. For this reason, we will try to minimize the data exchange via compressing data and not sending/receiving any unnecessary information.

- Global: We plan to allow any expert to work on any project they are eligible for. This means, an expert might be working on a project created by a client from a different country. In this case, different regulations from different regions will need to be taken into consideration.
- Cultural: Since the tasks given to the experts will be very generic, there were not any cultural factors taken into consideration when designing the project.
- Social: During their work sessions, Experts might have to interact with other Experts through our voice chat feature. This might cause problems if some mal-intentioned Experts start harassing others. In order to minimize this risk, we will implement a report functionality and also bans on phone numbers. Since the application requires an SMS confirmation for registration, a ban on a phone number should be able to reliably prevent banned users from using our application.
- Environmental: It is expected that our program will run in many different devices and might use a cryptocurrency for some of the transactions. For energy conservation, it is essential that the program runs efficiently in terms of battery usage, and the selected cryptocurrency is also environmentally friendly. It should be noted that our application will allow Experts to work remotely and this will reduce their carbon footprint by reducing their transportation needs.
- Economic: It is expected that our application will require a big number of small transactions. These transactions may be from across the border which may increase their cost. In order to reduce the number of transactions, we will set a minimum amount that can be withdrawn from an Expert's account. Also, alternative to institutions that provide transaction services like banks, we will provide an option to get paid in a cryptocurrency that our platform supports. In the case that this feature gets implemented in the final product, it will be very important to select a low fee currency that does not have a big environmental impact.

	Effect Level	Effect
Public Health	6	Daily bonuses for a limited amount of work, Warning messages for prolonged work sessions
Public Safety	7	Safe usage agreement form upon registration, Warning message when high speeds detected while using our application
Public Welfare	7	Program needs to run on low end devices, Low battery usage, Low internet usage through data compression etc.
Global Factors	3	Different regulations need to be taken into consideration for different regional versions of the application
Cultural Factors	0	None
Social Factors	7	Report functionality, SMS confirmation on registration and ban on phone numbers
Environmental Factors	8	Efficient code, Environmentally friendly transaction alternatives
Economic Factors	8	Limit on minimum transaction amount, Low fee transaction alternatives

Table 1: Factors that can affect analysis and design

6 Teamwork Details

During the implementation of the project, the group members were divided into 2 groups to work on the web side and the mobile side. Although the group was divided into two, the communication between the group members continued continuously in the parts that required synchronization. In the rest of this section, detailed information about the team work will be given.

6.1 Contributing and Functioning Effectively on the Team

Yiğit Gürses: Yiğit worked in the mobile application part of the project. Since there is voice chat in the *Laber*, he conducted research on how to provide the voice chat feature. As a result of his research, it was decided to use a server client model for the voice chat feature in the project. He also assisted in the development of graphical user interfaces for the mobile application in accordance with the mockups in the analysis report. He also did research on which database could be used for the project and then worked on modeling the project database.

Melisa Taşpınar: Melisa worked in the mobile application part of the project. Since *Laber* is a project that depends on the evaluations made by *Laber* experts, the reliability of the experts must be ensured. Melissa conducted research on how to ensure the reliability of these experts. As a result of her research, it was decided that the reliability of the experts' answers could be followed by having the experts evaluate the known tasks. It was also decided that the reliability of the experts could be increased by using a training section before using the application. She also assisted in the development of graphical user interfaces for the mobile application in accordance with the mockups in the analysis report.

Onur Oruç: Onur worked in the web part of the project. Since *Laber* is a system that works on paying customers and paying experts on the basis of customer payments**,** Onur did research on how this payment system can be made in *Laber*. As a result of his research, it was decided that payment can be received from customers' credit cards thanks to the Java Script's Payment Request API. At the same time, it was decided that the payments to the experts could be made through their bank account numbers. He also assisted in the development of graphical user interfaces for web application in accordance with the mockups in the analysis report.

Emin Adem Buran: Adem worked in the web part of the project. Since there is a discussion thread feature for tasks in the *Laber*, he conducted research on how to provide the discussion thread feature. As a result of his research, it was decided to use a server client model for the discussion thread feature in the *Laber*. He also assisted in the development of graphical user interfaces for web application in accordance with the mockups in the analysis report. He also

did research on which database could be used for the project and then worked on modeling the project database.

Mustafa Hakan Kara: Hakan worked in the mobile application part of the project. In the *Laber*, in line with the requests of the customers, the posts from the social media should be taken according to the keywords provided by customers and evaluated by the experts. Hakan did research on how to pull these posts from social media and as a result of his research, he found that there are some APIs that Twitter and Facebook allow developers to use with limited means. It was aimed to use these APIs in *Laber*. He also assisted in the development of graphical user interfaces for the mobile application in accordance with the mockups in the analysis report.

6.2 Helping Creating a Collaborative and Inclusive Environment

We share responsibilities. When someone responsible for a task has other responsibilities or s/he finds the task given to him/her hard to complete, we help him/her finish the task. When someone is confused about a task, we always help each other to clarify what should be done exactly. We periodically inform each other about the status of the tasks we are responsible for to keep each other up to date. Even if we assign different tasks to each other, we always ask other teammates' opinions while we are working on a task or after finishing the task to ensure that the tasks have been finished in the correct way. We give each other feedback for the tasks we are responsible for and make changes accordingly.

6.3 Taking Lead Role and Sharing Leadership on the Team

We always shared the leadership responsibility. When we needed to plan a meeting, determine our schedule, or do something together in general, whoever was the least busy at the moment took responsibility to lead others. The reason behind this is for both to reduce each other's burden and add different perspectives to the way we proceed. Moreover, even if we have different leaders at different times, it does not mean that the leader makes choices on his/her own. The responsibility of our leader is to organize people, combine different thoughts and come up with a plan by putting the thoughts together.

7 References

[1] Lohr, Steve. “The Age of Big Data.” *The New York Times,* 11 Feb. 2012, https://www.nytimes.com/2012/02/12/sunday-review/big-datas-impact-in-the-world.html. Accessed 5 Oct. 2021.

[2] Cavaioni, Michelle. “Machine Learning: Causes of error.” *Medium*, 1 Feb. 2017, https://medium.com/machine-learning-bites/machine-learning-causes-of-error-87ff372cd5be. Accessed 5 Oct. 2021.

[3] “Amazon Mechanical Turk.” *Amazon Mechanical Turk*, https://www.mturk.com/. Accessed 5 Oct. 2021.