CmpE 352 Spring 2020 Group 6
Milestone 2

Group Members

Ahmet Berat Can
Doğukan Kalkan
Emilcan Arıcan
Muhammed Enes Toptaş
Ezgi Gülperi Er
Tuba Çibuk
Hamza Işıktaş
Hazer Babur
Ömer Faruk Özdemir
Salih Can Özçelik
Cemre Efe Karakaş

1. Executive Summary

a. Introduction :

Akademise is an academic collaboration platform that provides an environment for academicians to collaborate and share ideas. They can publish papers and/or work on projects together with other academicians.
Our platform helps them to manage this process in a user-friendly way.

b. Work done so far:

Before explaining the work done so far, let us state that the work until the milestone 1, please refer to Milestone 1 in the wiki page.

In Assignment 7 , we were given a task to implement a practice application that helps us familiarize with the use of third-party services. The task also creates an environment for the team members to work as a team and to cooperate with each other.

We started the task by splitting the team into two groups, namely back end team and front end team. Both teams made some research about their respective fields and informed each other about the conclusions. We initially, as a team, decided to work with an API that is related to our main project and picked a Python package named *scholarly.* Although it is not a real API, it works pretty well for our task.

The team, afterwards, decided to implement different APIs for every member to contribute to the implementation. As a result, we implemented two more APIs about Covid-19 and Bitcoins.

Each team member was assigned to create at least one endpoint of his/her choice for the corresponding API and also each member was supposed to inform others, especially the front end team about their work. The front end team created a base template that is used for each page of our application.

During the implementation of our application, each team member worked as hard as possible to provide reviews and observations about work of others as well as to contribute to the work overall.

And finally, We deployed our onto an AWS Server.

c. Road ahead:

Our system will support web and android platforms. We have formed sub-groups to work on related areas. Those are backend and web-development groups. The former group worked on the API implementation and integration. The latter group worked on the frontend for the web-application. Next semester there is going to be also an android sub-group as well. We implemented an example practice-app api. Next semester we can start on the real project as we experienced created a web app together.

d. Challenges you met as a group:

During implementation, our first challenge was task distribution. Since none of the tasks were independent of each other it was hard to separate them and assign them to different individuals. For example at first, we had different teams for dealing with

backend and obtaining data from APIs. After starting implementation, we have realized that merging these two groups would make the workflow easier.
In the beginning of the implementation phase, we had some commit / merge conflicts. There have been some accidental commits to master branch that caused method deletions and overridings. We tackled this problem by creating development branches by groups (frontend, coronavirus api, scholarly api).
Regarding the technical part, the biggest problem we faced was to find an API to obtain data from Google Scholar. Since there is no official API for the platform, we had to use a package named Scholarly. Although it offers the functionalities we were looking for, it has performance issues. Also since it is still in development phase, we had some problems about reliability.
Another challenge we faced was deploying our application onto Amazon server, which we eventually solved, thanks to our dedicated group members,

2. List and Status of Deliverables

2.1 Communication plan

Where	Purpose	Frequency	Audience	Channel
Slack	Keeping each other informed about progress	Anytime	All members	Online
Github	Following Overall Work / Issue Management	Every Thursday @ 12 PM	All members	Face-to-face
Bogazici University, Cmpe Building, BM A5	Weekly Review	Every Thursday @ 12 PM	All members	Face-to-face
Bogazici University, Cmpe Building, BM Classes	Customer Meeting	When Needed	Some members	Face-to-face
Whatsapp	Urgent Communication	Anytime	All members	Online
Piazza	Communication With The Customer	Anytime	All members	Online
Zoom	Weekly Review	Every Thursday @ 12 PM	All members	Online

2.2 Requirements

1. Functional Requirements

- **1.1 User Requirements**
    - **1.1.1** Users shall be able to search and view users, public papers and projects
    - **1.1.2** Users shall be able to view profile pages of authors, and collaborators of papers and projects
    - **1.1.3** Users shall be able to search related information about the upcoming conferences or journal special issues
    - *1.1.4 Guests*
        - **1.1.4.1** Guests shall not be able to view private papers or projects.
        - **1.1.4.2** Guests shall not be able to join papers/projects, comment to papers/projects or post new papers/projects.
    - *1.1.5 Registration*
        - **1.1.5.1** Users shall be able to register by providing name, surname, unique email address and password
        - **1.1.5.2** Users shall validate the e-mail address for completing the registration phase
        - **1.1.5.3** Users shall be able to specify the research interests
        - **1.1.5.4** Users shall provide the information about affiliation
        - **1.1.5.5** Users shall be able to identify themselves as student or academician
    - *1.1.6 Registered Users*
        - **1.1.6.1** Users shall login by email and password provided
        - **1.1.6.2** Users shall be able to state whether s/he is looking for a project to collaborate
        - **1.1.6.3** Users shall be able to send request to join the team
        - **1.1.6.4** Users, as the owner of the projects, shall be able to accept or reject requests to join the team
        - **1.1.6.5** Users shall be able to invite any user to participate
        - **1.1.6.6** Users shall be able to accept or reject invitations

- **1.1.6.7** Users shall be able to rate and comment other users that s/he collaborated with during the collaboration and after the project is completed
- **1.1.6.8** Users shall be able to collaborate on more than one paper/project
- **1.1.6.9** User shall be able to disable join requests to end call for collaboration phase
- **1.1.6.10** Users shall be able to have more than one paper/project topic posted
- **1.1.6.11** Users shall be able to specify a deadline for the paper/project
- **1.1.6.12** Users shall not be able to delete a paper/project that is in progress
- **1.1.6.13** Users shall be able to file complaints
    - **1.1.6.13.1** Users shall be able to report another user, a project or a paper for unwanted/inappropriate behavior
    - **1.1.6.13.2** Users shall be able to mark a publication as unrelated
- **1.1.6.14** Users shall be able to follow other users on the platform.
    - **1.1.6.14.1** Users shall be able to approve/reject follow requests.
    - **1.1.6.14.2** Users shall be able to set visibility of their projects/papers for the users not following them.
    - **1.1.6.14.3** Users shall be able to request to join a project/paper if they can see it.
    - **1.1.6.14.4** Users shall be able to invite other uses to collaborate, without following them.
- **1.1.6.15** Users shall be able to set their profiles to public or private.
- **1.1.6.16** Users shall be able to log out from his account.
- **1.1.6.17** Users shall be able to delete his account.
- **1.1.6.18** Users shall be able to delete or update a comment s/he made before.
- **1.1.6.19** Users shall be able to delete or update a rate s/he gave before.
- **1.1.6.20** Users shall be able to add additional files to the collaborated projects/papers.
- **1.1.6.21** Users shall be able to add additional files to their projects/papers in progress.

- *1.1.7 Registered Users Requirements*

- *1.1.7.1 Profile Page*
    - **1.1.7.1.1** Users shall be able to provide information about the research area, recent publications and affiliation.
    - **1.1.7.1.2** Users shall be able to link their Google Scholar or ResearchGate accounts.
    - **1.1.7.1.3** Users shall be able to edit their own profile page.
    - **1.1.7.1.4** Users shall be able to see invitations that are sent from the other users.
    - **1.1.7.1.5** Users shall be able to see messages that are sent from the other users.
    - **1.1.7.1.6** Users shall be able to see ratings and comments that are made by other users.
    - **1.1.7.1.7** Users shall be able to see their followers.
    - **1.1.7.1.8** Users shall be able to see people who they follow.
- *1.1.7.2 Creating / Editing Page*
    - **1.1.7.2.1** Users shall be able to provide information about: topic of the research, deadline of submission, milestones, codes, documents, result plots / figures, required skills in order to apply and if the the project is funded or not.
    - **1.1.7.2.2** Users shall be able to add a summary(abstract) part.
    - **1.1.7.2.3** Users shall be able to state the type of the content (Paper or Project).
    - **1.1.7.2.4** Users shall be able to state whether project/paper is private or public.
    - **1.1.7.2.5** Users shall be able to add co-authors and collaborators during creation of a project.
    - **1.1.7.2.6** Users shall be able to add tags related to project/paper.
    - **1.1.7.2.7** Collaborators shall be able to track the information shared by corresponding users.
    - **1.1.7.2.8** Collaborators shall be able to prepare document or article for submission simultaneously.
- *1.1.7.3 Paper/Project Page*
    - **1.1.7.3.1** Users shall be able to create a new paper.
    - **1.1.7.3.2** Users shall be able to create a new project.
    - **1.1.7.3.3** Users shall be able to see the status of his/her paper(s).

        - **1.1.7.3.4** Users shall be able to see the status of his/her project(s).
        - **1.1.7.3.5** Users shall be able to edit the specifications of his/her paper(s).
        - **1.1.7.3.6** Users shall be able to edit the specifications of his/her project(s).
        - **1.1.7.3.7** Users shall be able to see his/her paper(s) in progress.
        - **1.1.7.3.8** Users shall be able to see his/her project(s) in progress.
        - **1.1.7.3.9** Users shall be able to edit the specifications of his/her paper(s) in progress.
        - **1.1.7.3.10** Users shall be able to edit the specifications of his/her project(s) in progress.
        - **1.1.7.3.11** Users shall be able to invite another user to collaborate on his/her paper(s) in progress.
        - **1.1.7.3.12** Users shall be able to invite another user to collaborate on his/her project(s) in progress.
        - **1.1.7.3.13** Users shall be able to accept or reject another users invitation to collaborate on his/her paper or project

- *1.1.8 Search Page*

    - **1.1.8.1** Users shall be able to search; other users, papers, projects, conferences and journals within the system.
    - **1.1.8.2** Users shall be able to filter search results with regards to the research area, topic, scope and difficulty.
    - **1.1.8.3** Users shall be able to find related posts searching related keywords.
    - **1.1.8.4** Users shall be able to view previews of papers/journals/projects/conferences related to their activities, profile and interests.

- *1.1.9 Events Page*

    - **1.1.9.1** Users shall be able to see about upcoming conferences related to joined projects/papers.
    - **1.1.9.2** Users shall be able to see closing deadlines of joined projects/papers.
    - **1.1.9.3** Users shall be able to travel in the timeline to see past events.
    - **1.1.9.4** Users shall be able to travel in the timeline to see far future events.

        - **1.1.9.5** Users shall be able to synchronize this calendar with other calendar applications.
        - **1.1.9.6** Users shall be able to view particular event more detailed when clicked/tapped.

- **1.2 System Requirements**

    - *1.2.1 Search & Recommendation*

        - **1.2.1.1** System shall support basic search of the available content.
        - **1.2.1.2** System shall support semantic search.
        - **1.2.1.3** System shall provide a tagging system.
        - **1.2.1.4** System shall provide a filtering system. This system filters the content such as projects/papers/profiles showed to the registered user.
        - **1.2.1.5** System shall provide a recommendation mechanism that recommends papers/journals/projects/conferences to the registered user based on the user activities and profile.
        - **1.2.1.6** System shall provide a recommendation mechanism for project owners, to recommend possible collaborators based on their research interests, or previous works in that area.

    - *1.2.2 Homepage*

        - **1.2.2.1** System shall provide a homepage for each registered user, showcasing recent sharings.

    - *1.2.3 Profile Page*

        - **1.2.3.1** System shall provide a profile page for each registered user, showcasing their interests, past contributions to projects and papers, their contacts and their related links (example: Google Scholar).
        - **1.2.3.2** System shall retrieve information (example: citations, h-index) of a user from their Google Scholar/ResearchGate pages.

    - *1.2.4 Notifications*

        - **1.2.4.1** System shall provide a notification mechanism for informing users about the incoming invitations
        - **1.2.4.2** System shall provide a notification mechanism for informing users about whether their request to join a team accepted or rejected
        - **1.2.4.3** System shall provide a notification mechanism for informing users about whether someone accepted or rejected their invitations

        - **1.2.4.4** System shall provide a notification mechanism for informing users about whether someone has commented on their profile
        - **1.2.4.5** System shall provide a notification mechanism for warning users if they perform an unauthorized activity

    - *1.2.5 Events*

        - **1.2.5.1** System shall provide a page of Events for every registered user.
        - **1.2.5.2** System shall fill this page with the events related to registered user.

2. Nonfunctional Requirements

- **2.1 Activity Streams**

    - **2.1.1** System shall obey W3C HYPERLINK "https://www.w3.org/standards/"Standards
    - **2.1.2** System shall follow W3C Activity HYPERLINK "https://www.w3.org/TR/activitystreams-core/"Streams HYPERLINK "https://www.w3.org/TR/activitystreams-core/" Model

- **2.2 Availability & Accessibility**

    - **2.2.1** The system shall be available as a website and as an Android application.
    - **2.2.2** The website and the application shall be available in English.
    - **2.2.3** The system shall notify users in case of a failure.
    - **2.2.4** Any failure in the system shall be handled in at most half an hour.

- **2.3 Perfomance & Scalability**

    - **2.3.1** The system shall use queuing algorithms and caching mechanism to reduce response time.
    - **2.3.2** The system shall respond to a request at most 5 seconds.
    - **2.3.3** The system shall be able to res'pond up to 1000 requests per second.

- **2.4 Security**

    - **2.4.1** The system shall be protected from unauthorized accesses and attacks to the system and its stored data according to KVKK
    - **2.4.2** The system shall force users to use strong passwords that contain a lowercase letter, an uppercase letter, a number and a special character and contain 6-24 characters in total.

- **2.4.3** The system shall store only the hashed version of passwords.
- **2.4.4** The system shall force users to change their passwords once a year.
- **2.4.5** The system shall deny all the access attempts to an account after three failed access attempts until user re-validates his/her account via a link sent in an email.
- **2.4.6** The system shall provide a validation code for users during registration.
- **2.4.7** The system shall be protected against SQL injection and DDOS attacks.

2.3 Scenarios and Mockups

GUEST USER / Scenario 1

Persona (Ertugrul Düldül)

- 23 years old
- Student
- Currently pursuing bachelor's degree
- Enthusiastic about basketball
- Playing, watching, and thinking about basketball

Goals

- Being an academic basketball researcher
- Learning physics such that he can nail basketball
- Learning mathematics like his mother tongue to be a gambler

Preconditions

- Ertugrul is not a member of the platform.
- He visits the platform as a guest.

User story

- Ertugrul is a junior computer engineering student at Bogazici University. He wants to combine his computing and scientific skills with basketball and be the one who rules the whole game. He has been into basketball for as long as he can remember.

Scenario

- Ertugrul first visits the web page as a guest. Then he looks for some mechanical physics papers to learn more about the movement of a basketball. He finds a paper and starts reading it. When he wants to comment, he isn't able to do it since he is not registered. After that he decides to be a registered user so that he can comment to posts, see what he has read up to now, publish papers and projects through his profile page, invite people to contribute to them and join academic collaborations; he provides his full name, password, and an e-mail address to register and validate his account. Since he is not an academician, he does not have to provide university-specific information or his social security number.

Action List

- 1 . Ertugrul visits the web page as a guest.
- 2 . He searches for "mechanical physic".
- 3 . He finds a paper related to it.
- 4 . He opens the document.
- 5 . He wants to write a comment after reading.
- 6 . He decides to register since writing a comment requires to be a registered user.
- 7 . He fills the registration form.

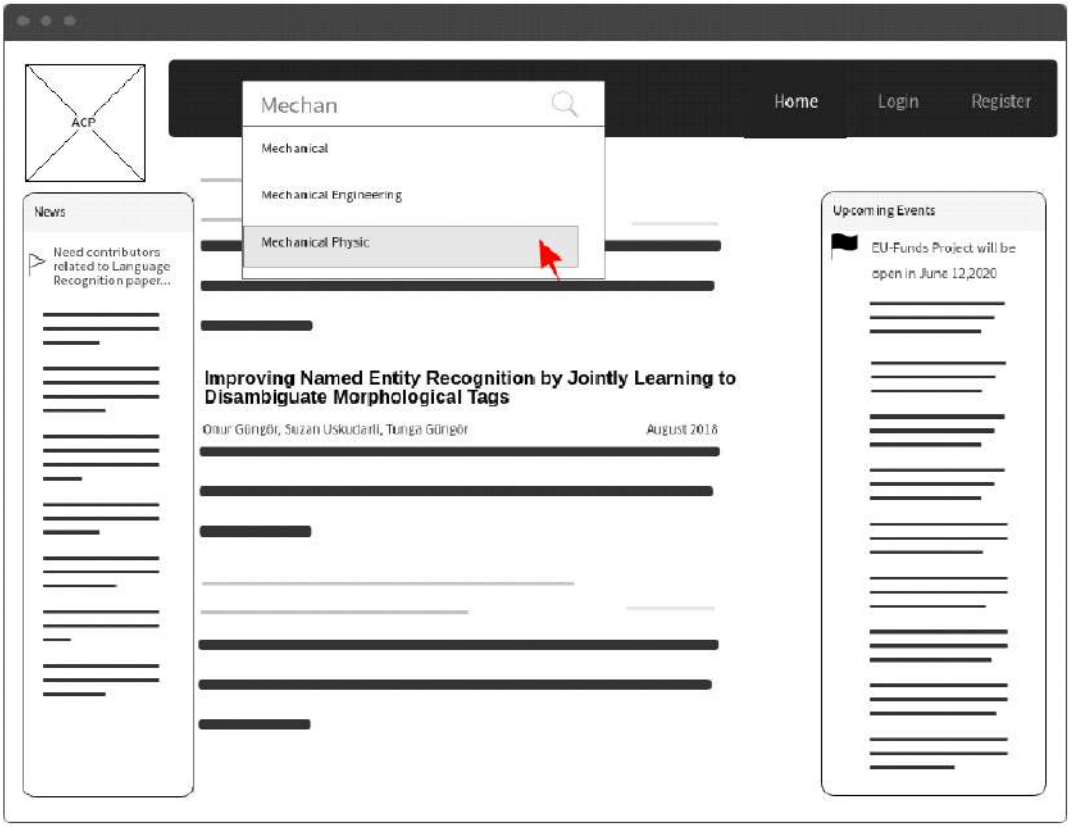
Acceptance criteria

- 1.1.1 Users shall be able to search and see public papers or projects
- 1.1.2 Guests should be able to see profile pages of authors, contributors and collaborators of papers and projects
- 1.1.3 Guests shall be able to search related information about the upcoming conferences or journal special issues
- 1.1.11 Registration
- 1.1.11.1 Users shall be able to register by providing name, surname, unique email address and password

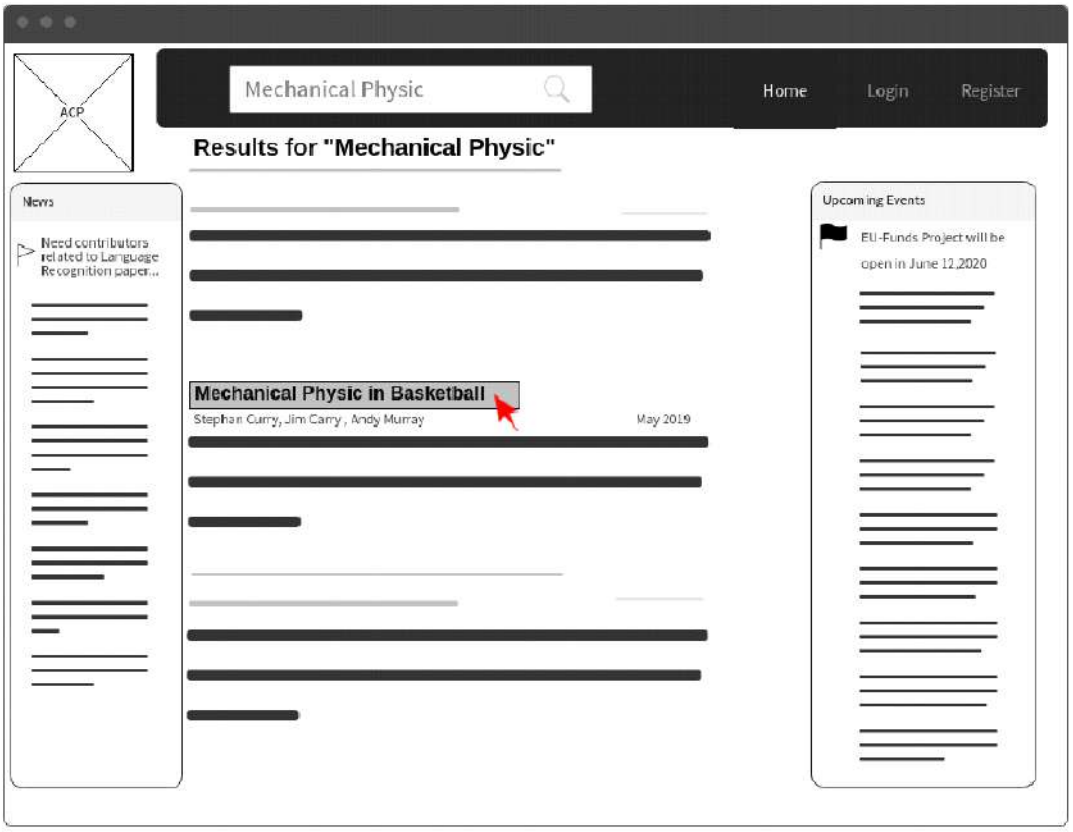
- 1.1.11.2 Users shall validate the e-mail address for completing the registration phase
- 1.1.11.3 Users shall be able to specify the research interests
- 1.1.11.4 Users shall provide the information about affiliation
- 1.1.11.5 Users shall be able to idendify themselves as student or academician

Mockup

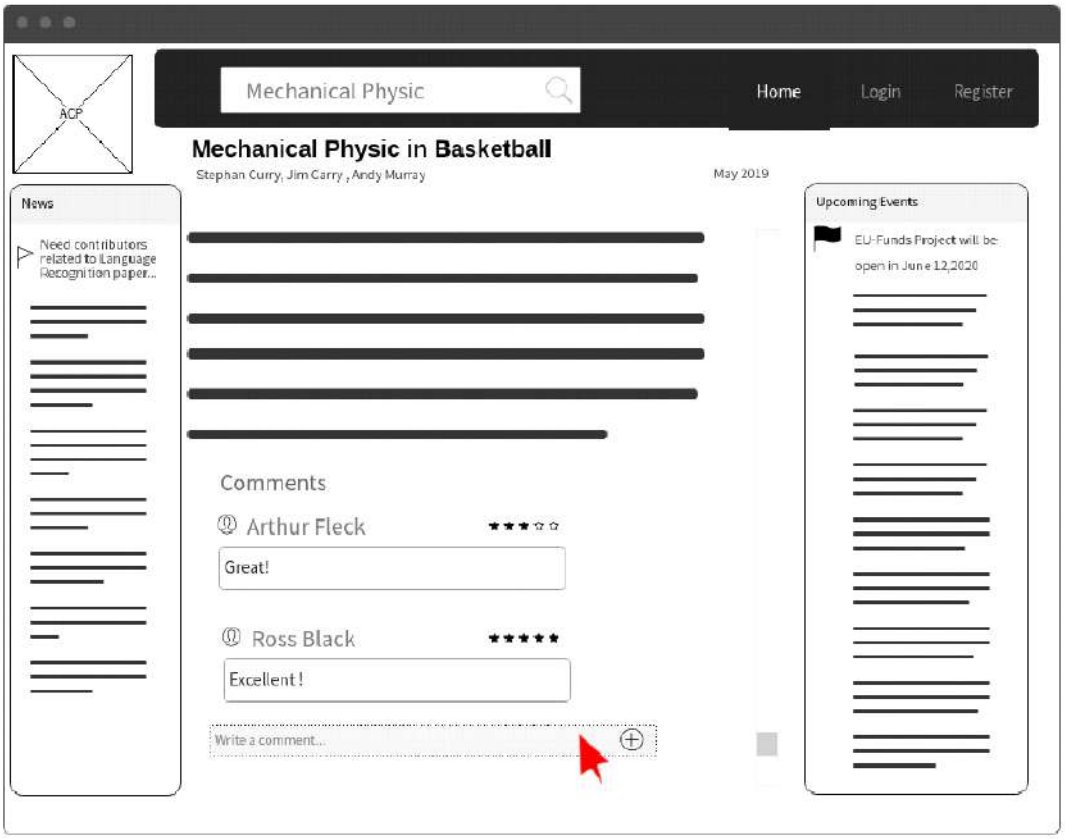
- Ertugrul wants to learn about mechanical physic in basketball. He visits the web page as a guest.
- Then search for "mechanical physic".



- Find a paper related to it.

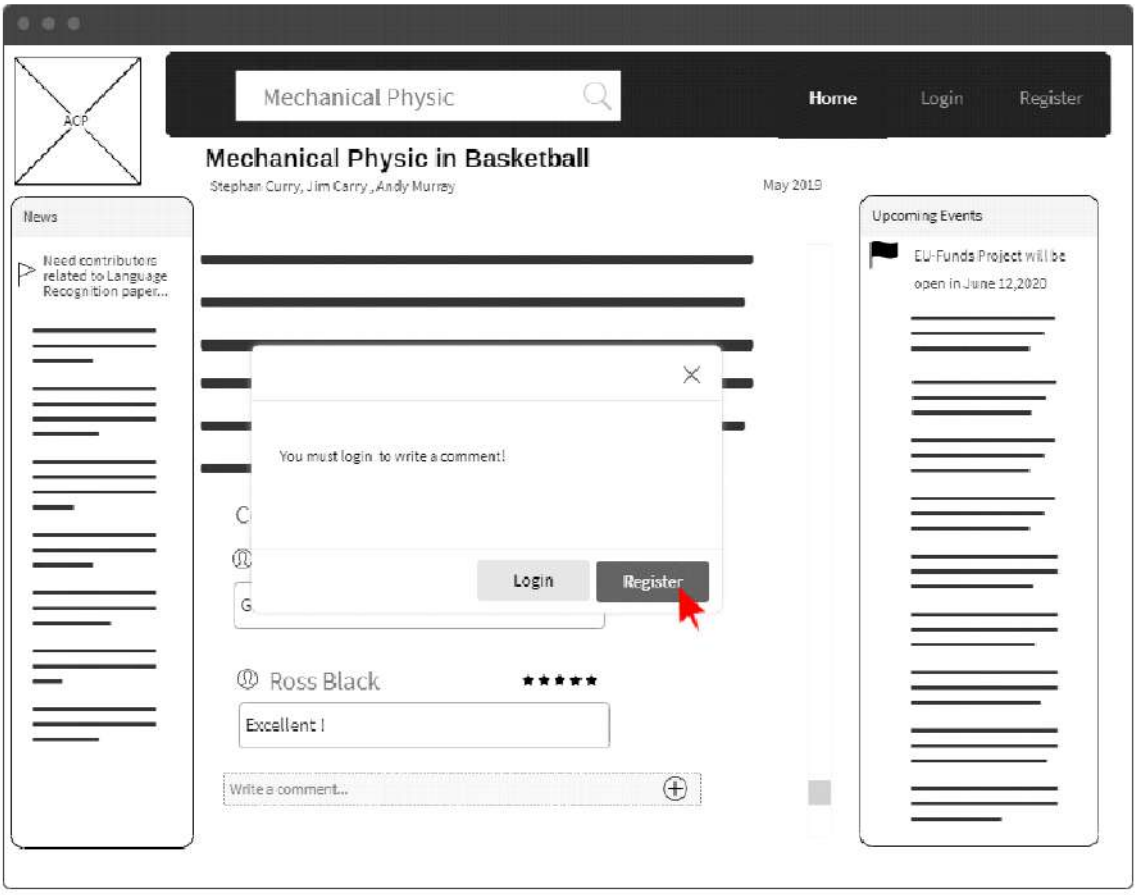

ACP
Mechanical Physic
Home
Login
Register
Results for "Mechanical Physic"
News
Need contributors related to Language Recognition paper...
Mechanical Physic in Basketball
Stephan Curry, Jim Carry , Andy Murray
May 2019
Upcoming Events
EU-Funds Project will be open in June 12,2020

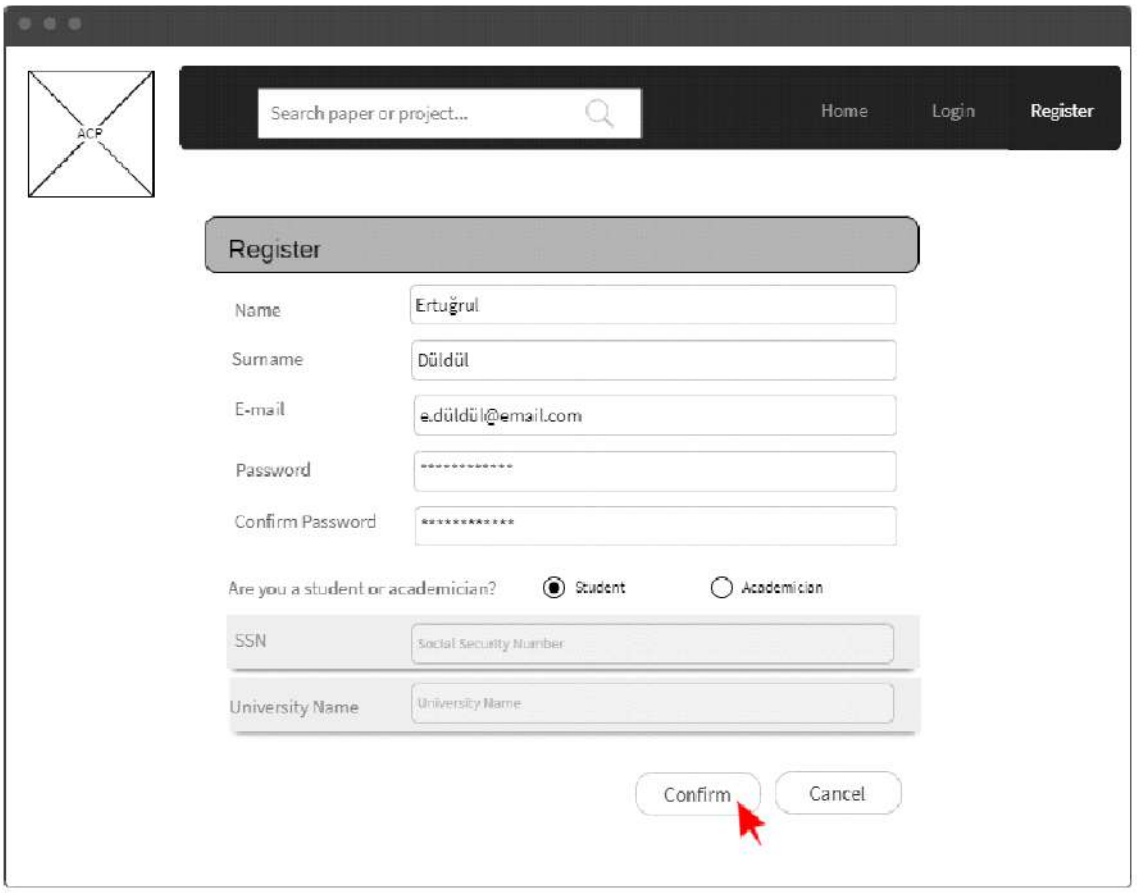
- After reading he wants to write comment



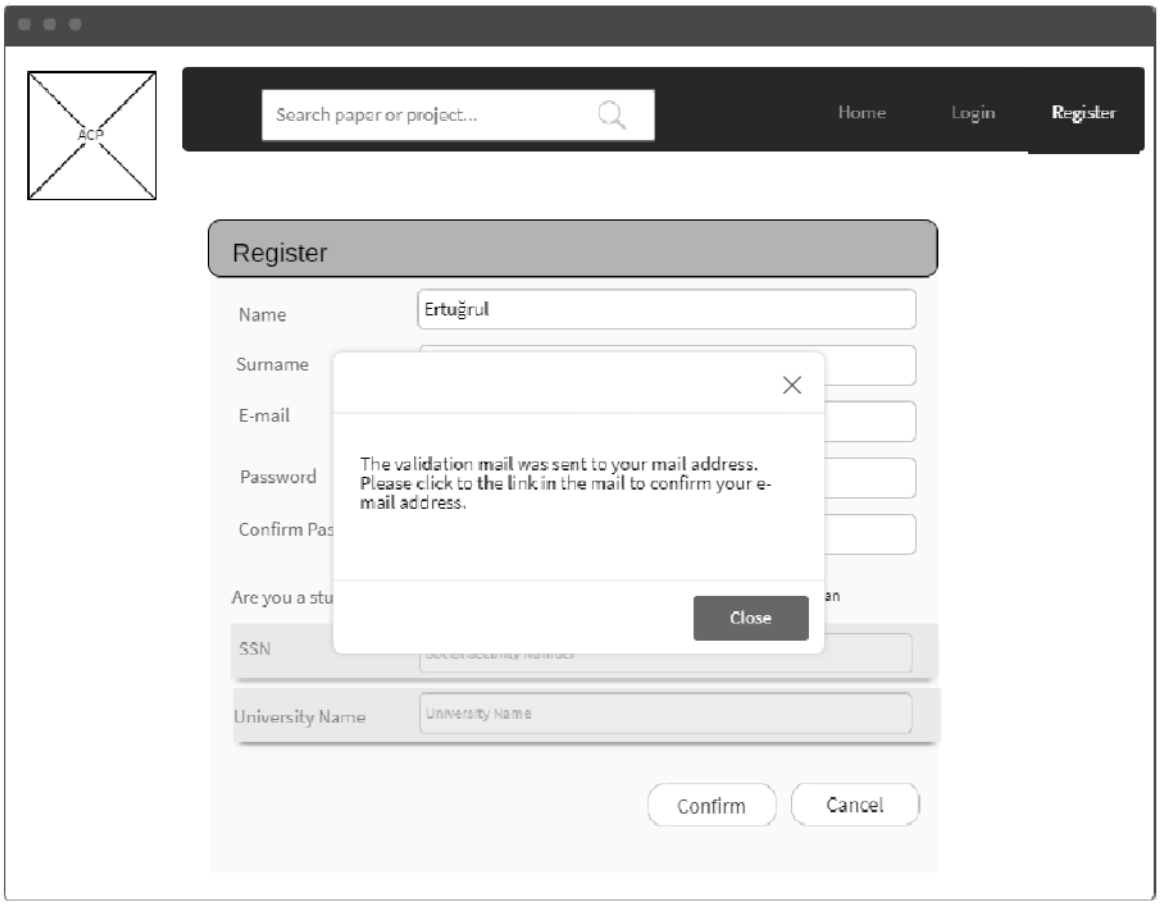
- Writing a comment requires to be login.

- In order to write a comment he decides to be a registered user.



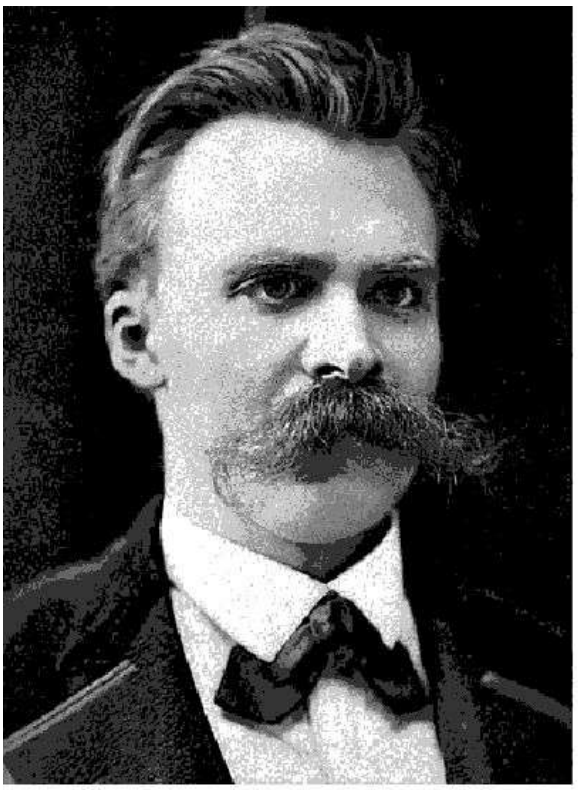


- After filling the registration form validation e-mail is sent.



Registered User / Scenario 2

Persona (Friedrich Nietzsche)



- 34 years old
- Nihilist.
- German philosopher, cultural critic, composer, poet, philologist, and scholar of Latin and Ancient Greek whose work has exerted a profound influence on modern intellectual history.
- Übermensch
- Loves horses

Goals

- Help other people comprehend the world around them
- Publish papers, create projects, and invite people to his research
- Paying respect to and supporting an independent academic platform

Preconditions

- The user has already logged in
- Friedrich Nietzsche is an old member of the platform
- He already has a great portfolio beforehand

User story

- Friedrich Nietzsche is a man who supports the independent academy. He wants to publish ideas/projects/papers on this free platform. He also stands up for every individual's right to feel free in academy and not hesitate to reveal their ideas.

Scenario

- Friedrich Nietzsche is a spirited member of academy but sick of people not appreciating good work. Since he believes his ideas are thrilling and paradigm-shifting, he applies to our platform using the web page to be a registered user so that he can freely publish his ideas/papers/projects. While publishing his new work he sets a topic of research, a deadline and some requirements. He creates a fund request and attaches some documents. He adds a summary part to his work that explains what the work is about and what type of publication it is (paper, project). He states that the project public and clicks publish. He can now see his paper is "in progress" (progress) and "awaiting funding" (status).

Action List

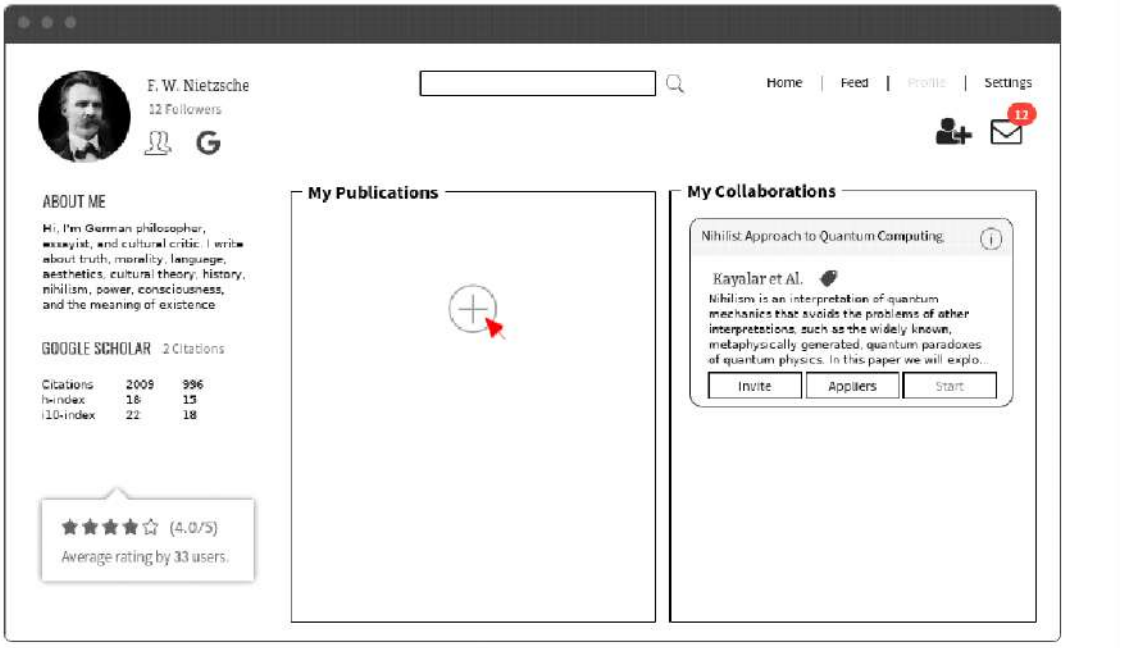
- 1 . Nietzsche is logged in and is looking at his profile page.
- 2 . He realizes it is a good time for creating a new publication and clicks "add" under publications.
- 3 . He fills out the new publication form according to his needs.
    - He sets a topic.
    - He writes a summary that explains what the work is/will be about.
    - He selects the type of publication to be "paper".
    - He skips the add files part.
    - He writes out some requirements.
    - He toggles the switch to apply for a funding.
- 4 . He clicks publish to finish creating the publication.
- 5 . Now back at his profile, he checks the info for the paper.
- 6 . He sees that the paper is in progress and is awaiting for funding.

Acceptance criteria

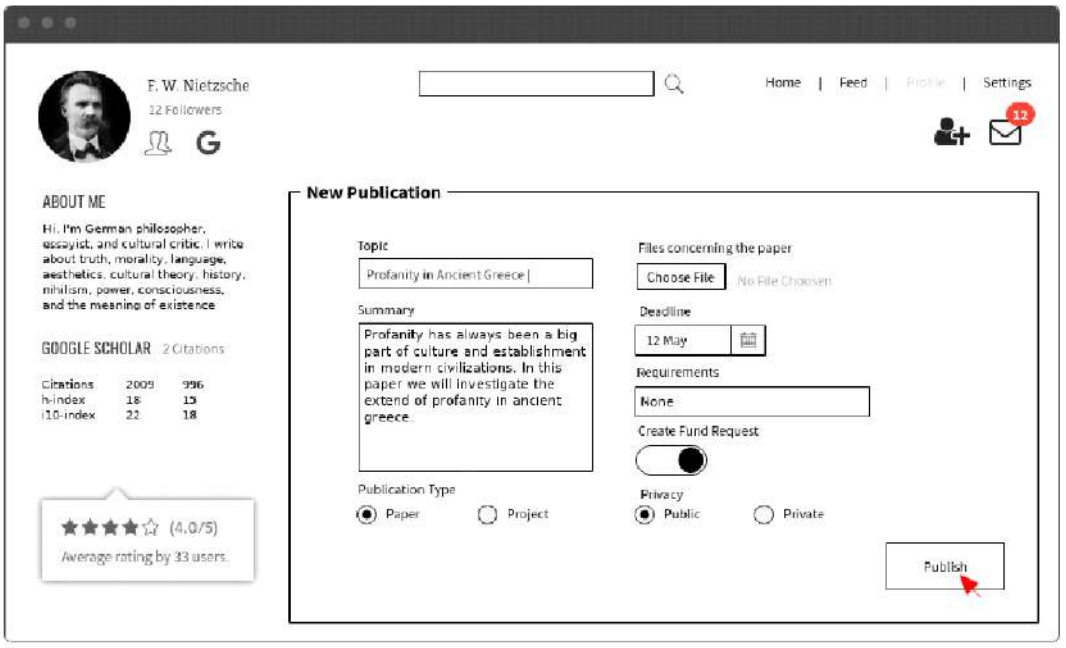
- 1.1.9.1 Users shall be able to create a new paper.
- 1.1.9.2 Users shall be able to create a new project.
- 1.1.9.3 Users shall be able to see the status of his/her paper(s).
- 1.1.9.4 Users shall be able to see the status of his/her project(s).
- 1.1.9.5 Users shall be able to edit the specifications of his/her paper(s).
- 1.1.9.6 Users shall be able to edit the specifications of his/her project(s).
- 1.1.9.7 Users shall be able to see his/her paper(s) in progress.
- 1.1.9.8 Users shall be able to see his/her project(s) in progress.

Mockup

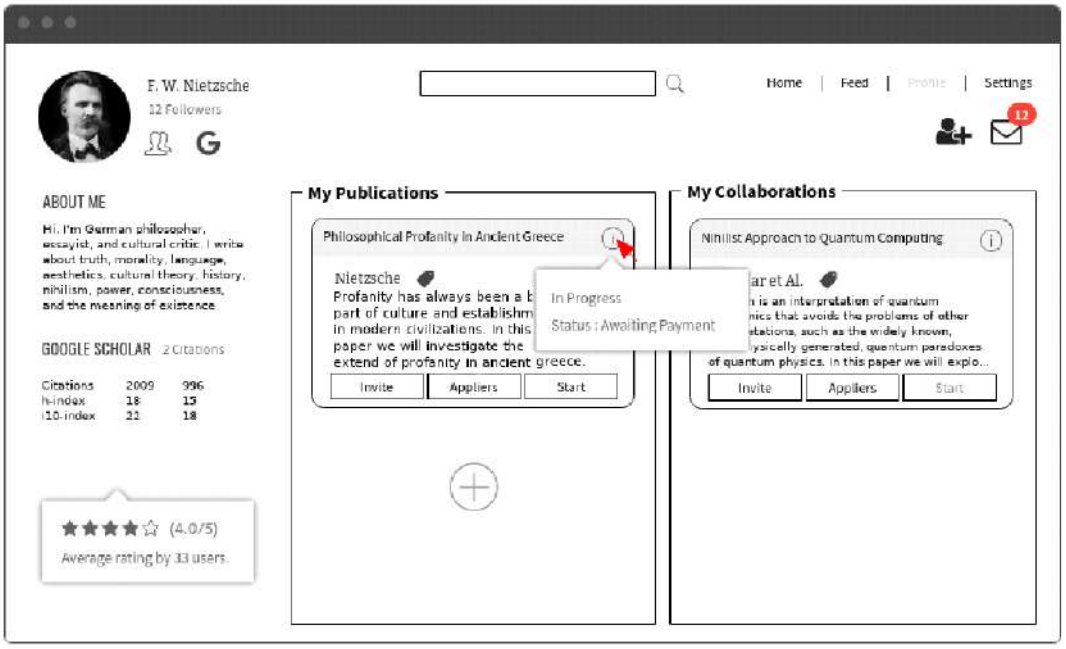
- Nietzsche is logged in and is looking at his profile page.
- He realizes it is a good time for creating a new publication and clicks "add" under publications.



- He fills out the new publication form according to his needs.
    - He sets a topic.
    - He writes a summary that explains what the work is/will be about.
    - He selects the type of publication to be "paper".
    - He skips the add files part.
    - He writes out some requirements.
    - He toggles the switch to apply for a funding.
- He clicks publish to finish creating the publication.



- Now back at his profile, he checks the info for the paper.
- He sees that the paper is in progress and is awaiting for funding.



Registered User / Scenario 3

Persona (Atakan Kayalar)

- 10 years old
- Kid
- Primary school student
- Part time philosopher
- Enthusiastic about all kinds of books, papers, and projects
- Currently holding the Guinness Record for most books read in 5 months (250)

Goals

- Wants to expand his knowledge about academic philosophy
- Wants to publish papers, create projects, and invite people to his research
- Wants to join other academicians' research

Preconditions

- The user has already logged in
- Atakan is a new member of the platform
- He has a great portfolio beforehand
- He has many followers who are interested in his research and quotas.
- He linked his Google Scholar account to his profile page.

User story

- Atakan is a primary school student who wants to broaden his academic skills. His main goal is to share his knowledge with the world so that a better world could be established with academy.

Scenario

- The little philosopher visits his profile page using his android phone. He sees that he has 150k followers and by clicking his google scholar page link he sees that he has 98724 citations. There is an invite button that he can invite someone to his papers/projects. He also has an invitation box wherein he can see all invitations for him. He can accept or reject the invitations.

Action List

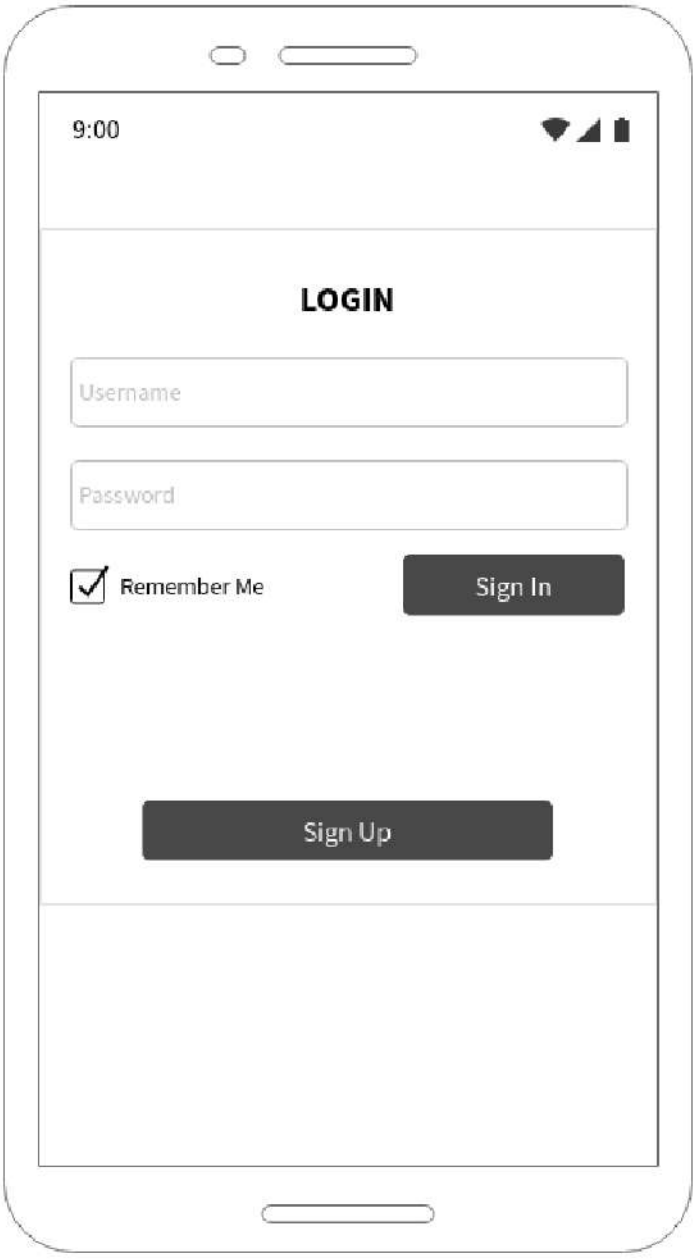
- 1 . The little philosopher logs in using his android phone.
- 2 . The little philosopher visits his profile page. He sees that he has 150k followers and he sees that he has 98724 citations.
- 3 . He looks at his pending paper and sees who is invited.
- 4 . He searches for one specific user and invites him.
- 5 . He checks the appliers for his paper.

Acceptance criteria

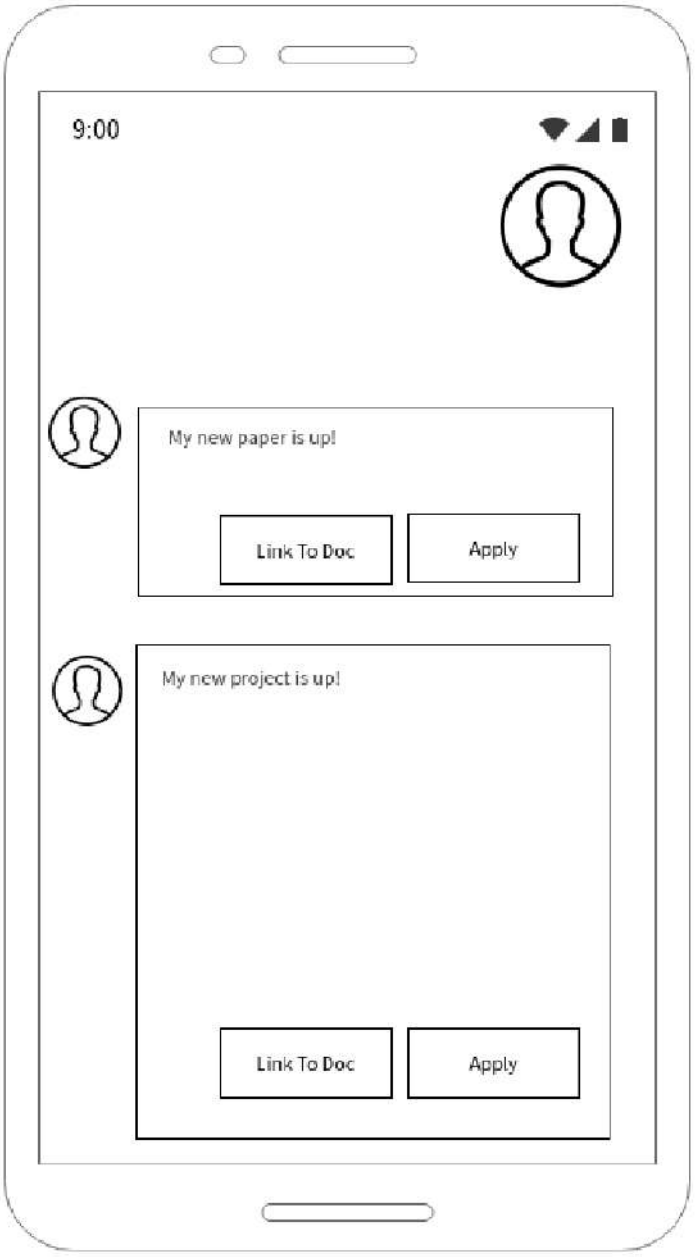
- 1.1.5.1 Users shall login by email and password provided
- 1.1.5.2 Users shall be able to send request to join the team of authors
- 1.1.5.3 Users shall be able to accept or reject requests to join the team of authors
- 1.1.5.4 Users shall be able to invite any academicians to participate
- 1.1.5.5 Users shall be able to search related information about the upcoming conferences or journal special issues
- 1.1.5.6 Users shall be able to rate and comment other users that s/he collaborated with
- 1.1.5.7 Users shall be able to collaborate on more than one paper/project
- 1.1.5.8 Users shall be able to have more than one paper/project topic posted
- 1.1.5.9 Users shall be able to specify a deadline or make the deadline open for the paper/project
- 1.1.5.10 Users shall not be able to delete a paper/project that is in progress
- 1.1.6.2 Users shall be able to link their Google Scholar or ResearchGate accounts.
- 1.1.6.4 Users shall be able to see invitations that are sent from the other users

Mockup

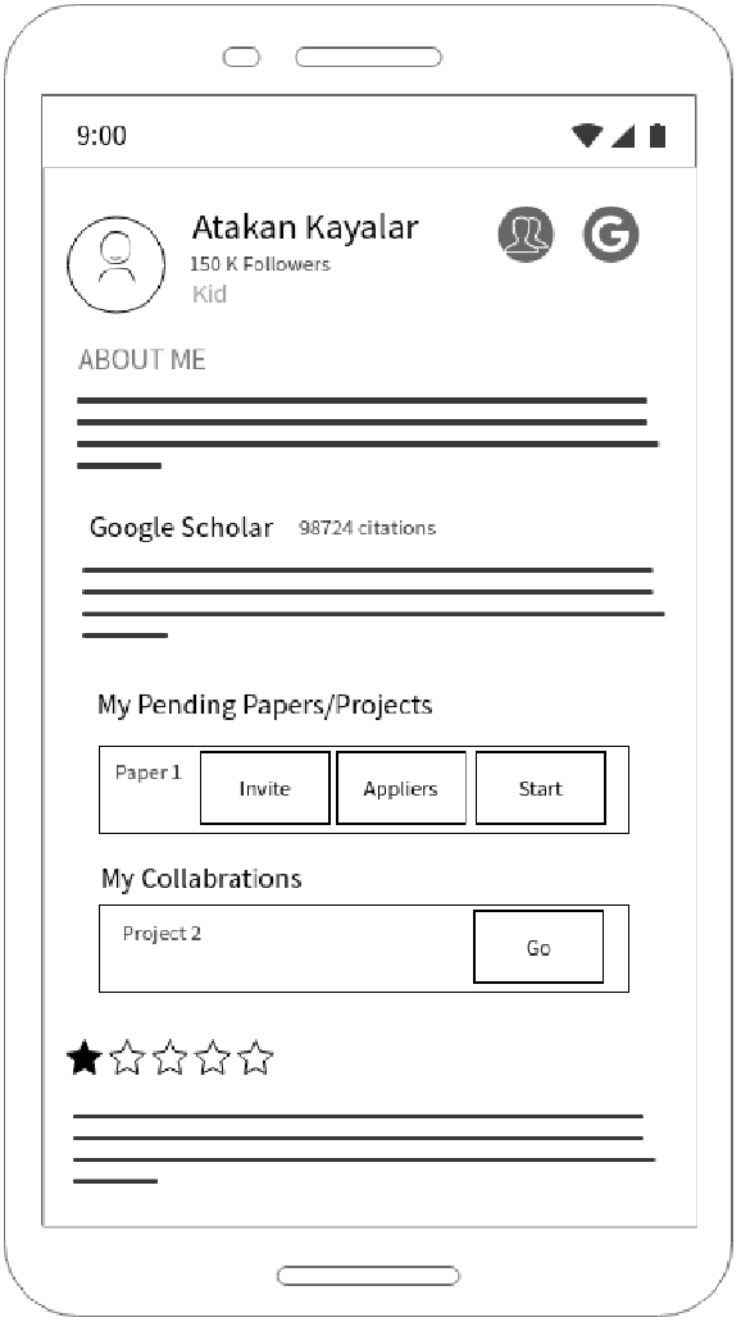
- Atakan first logs in with his e-mail and password.



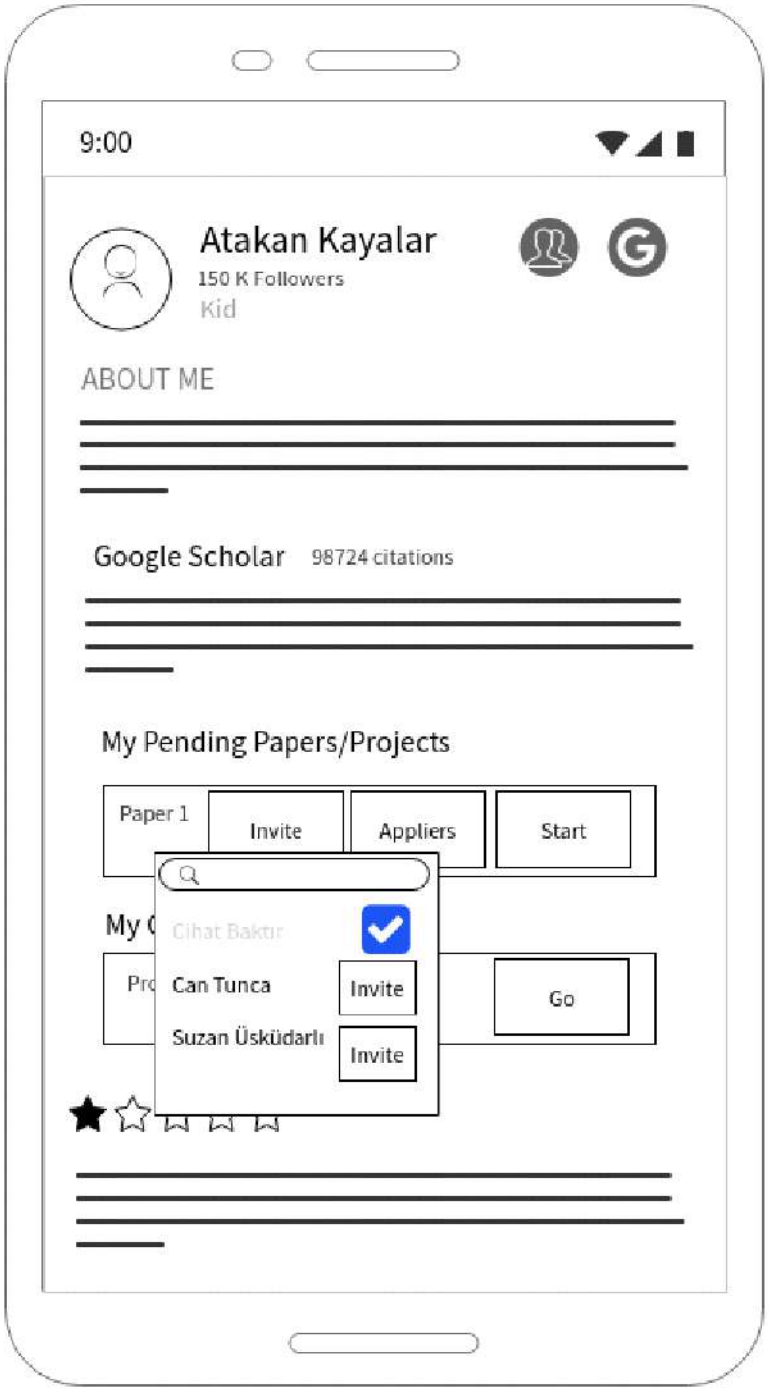
- Then home page is opened.
- He goes to his profile page by clicking his own profile photo. (After closing the document)



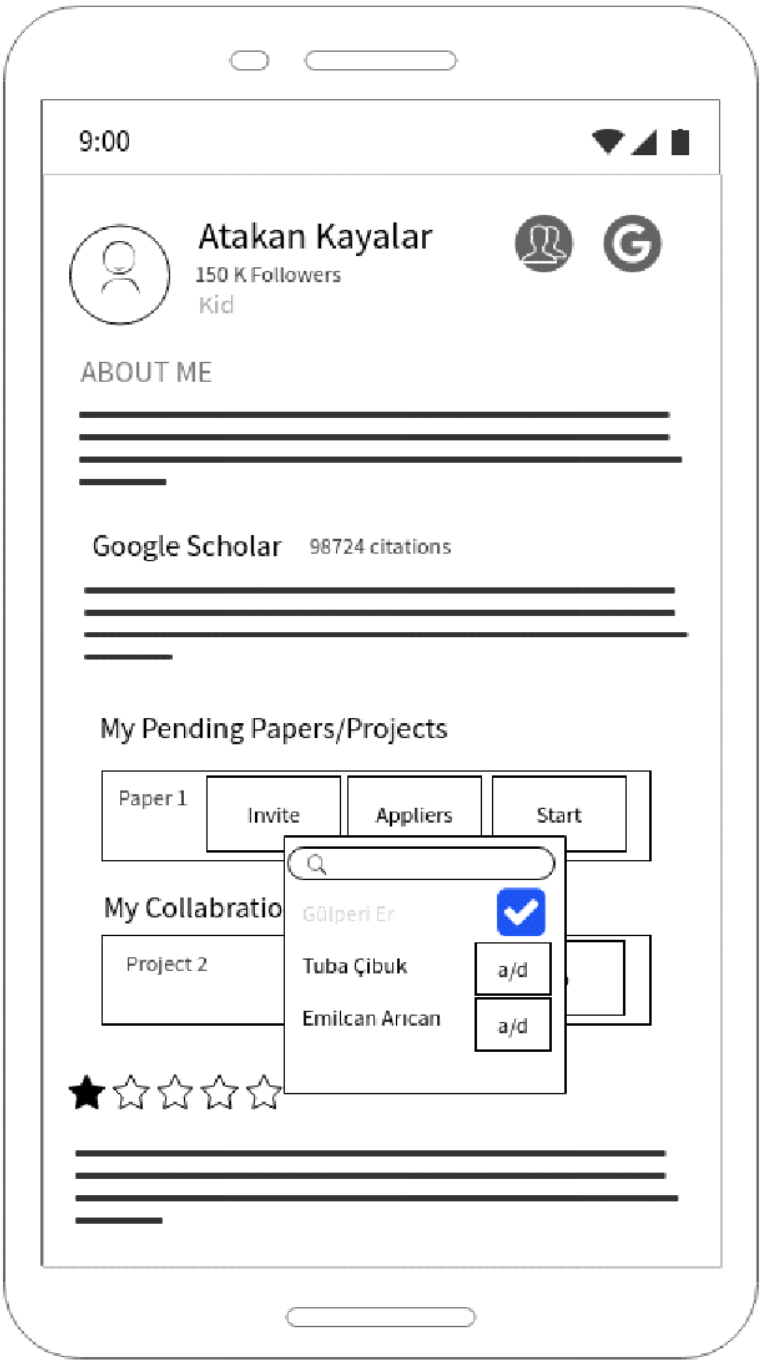
- When he clicks on his own profile photo, his profile page is opened.
- He sees his followers, buttons for linking Google Scholar or ResearchGate Accounts, and information about himself he wrote.
- He looks at his pending papers.
- He invites users by invite button, looks at applicants by appliers button.



- When he clicks invite button, this menu is opened.
- He searches users to invite them.
- He sees who is invited before.
- He invites a user by clicking buttons.

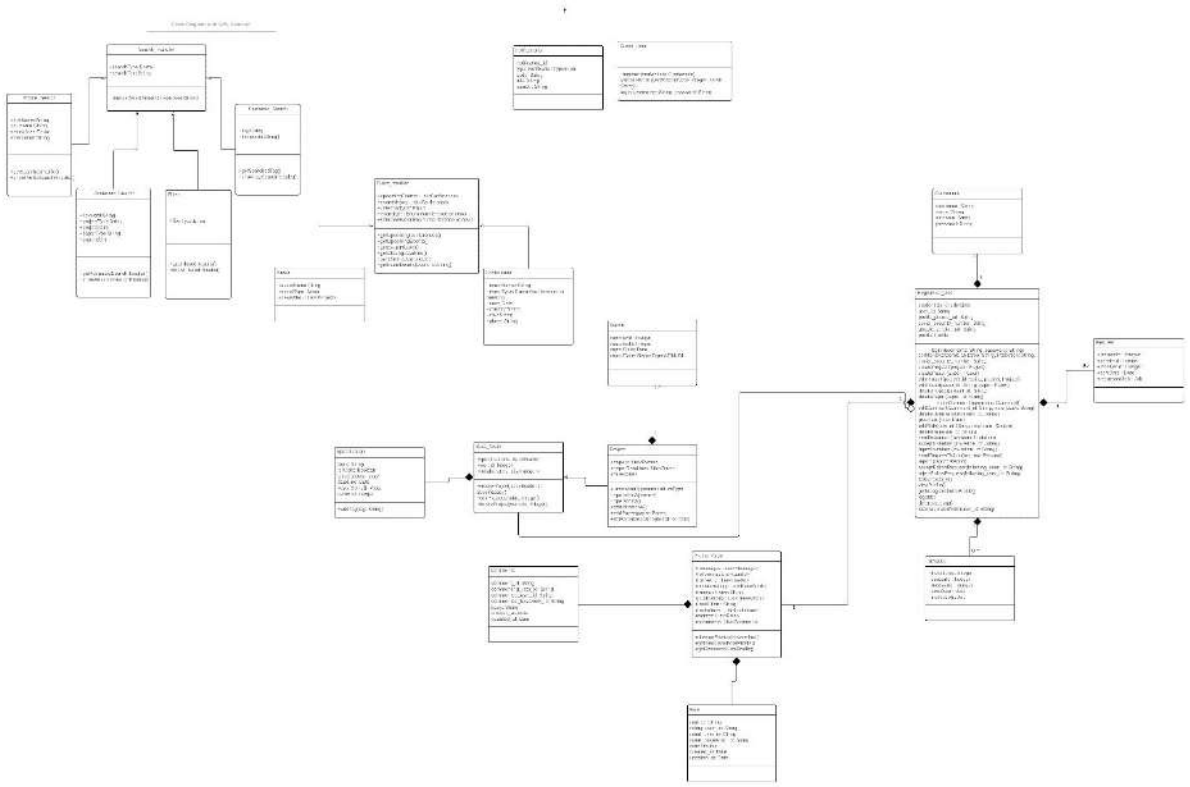


- When he clicks Appliers button, this menu is opened.
- He closes the menu touching the screen. (not on the menu but anywhere else)

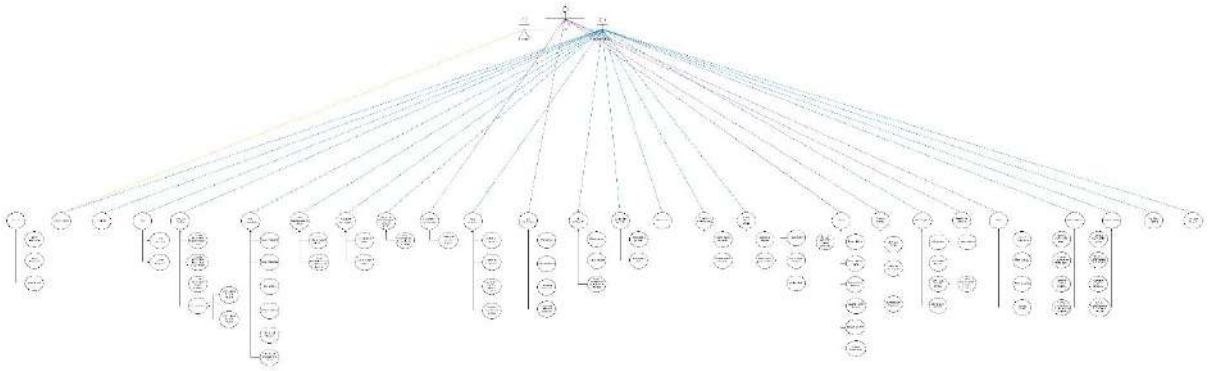

9:00
Atakan Kayalar
150 K Followers
Kid
ABOUT ME
Google Scholar
98724 citations
My Pending Papers/Projects
Paper 1
Invite
Appliers
Start
My Collabratio
Project 2
Gülperi Er
Tuba Çibuk
a/d
Emilcan Arıcan
a/d

2.4 UML Software Designs

Class Diagram

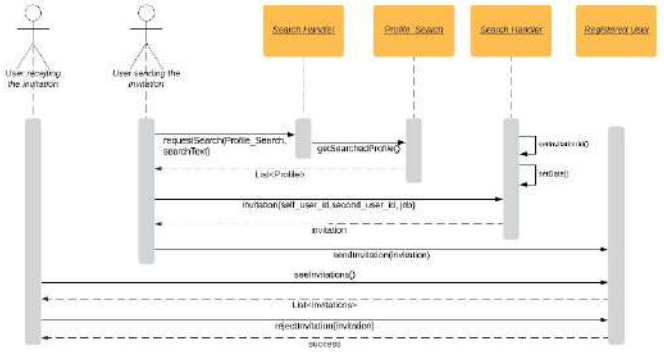


Use-Case Diagram

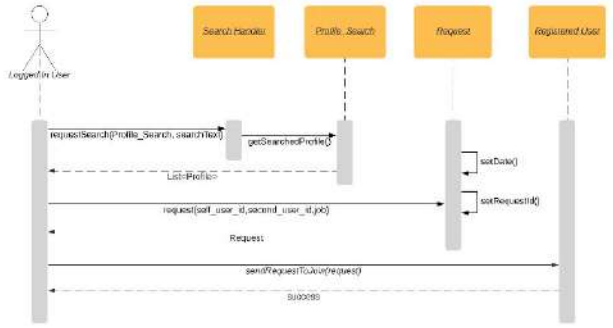


Sequence Diagrams

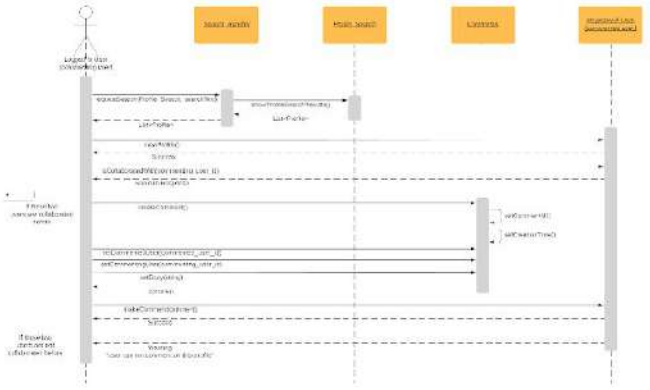
Reject Collab Invitation



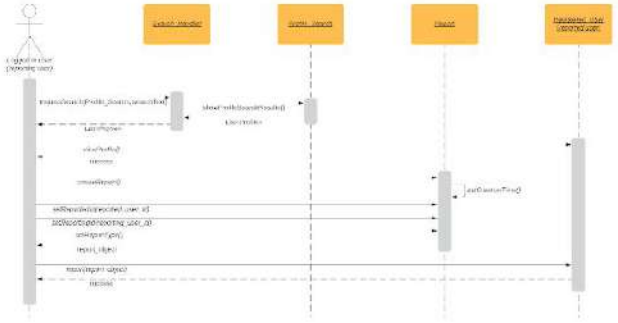
Send Collab Request



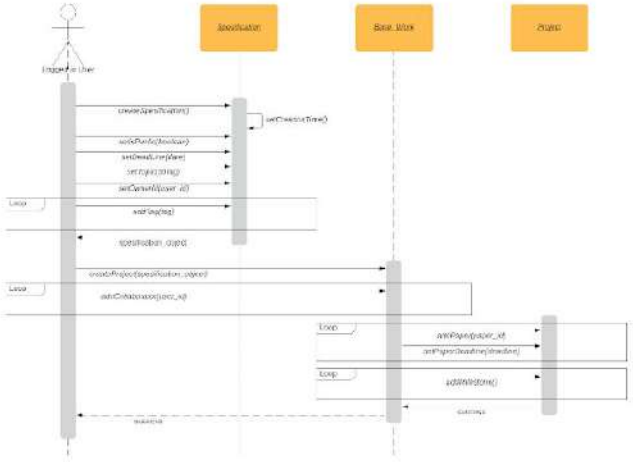
Make Comment



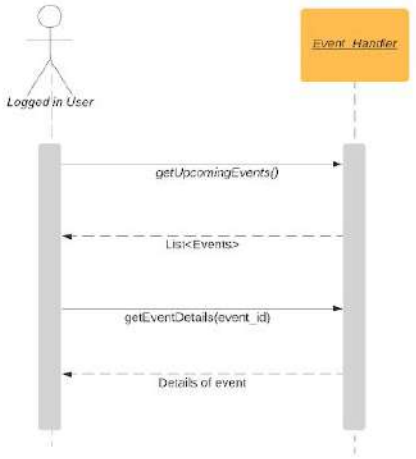
File Complaints



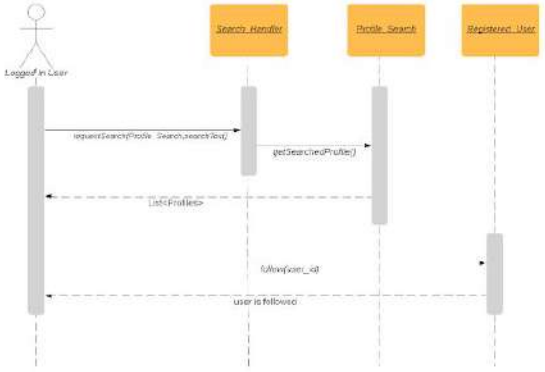
Create Project



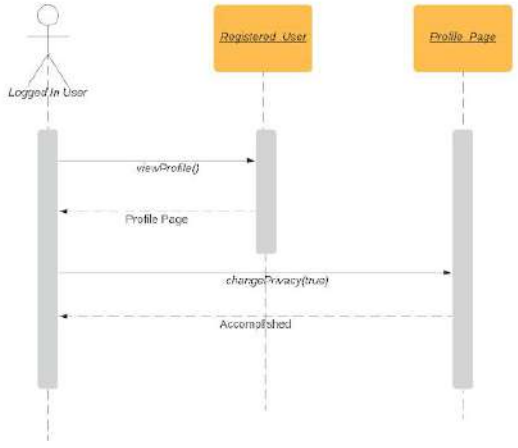
See Upcoming Events in Detail



Search and Follow

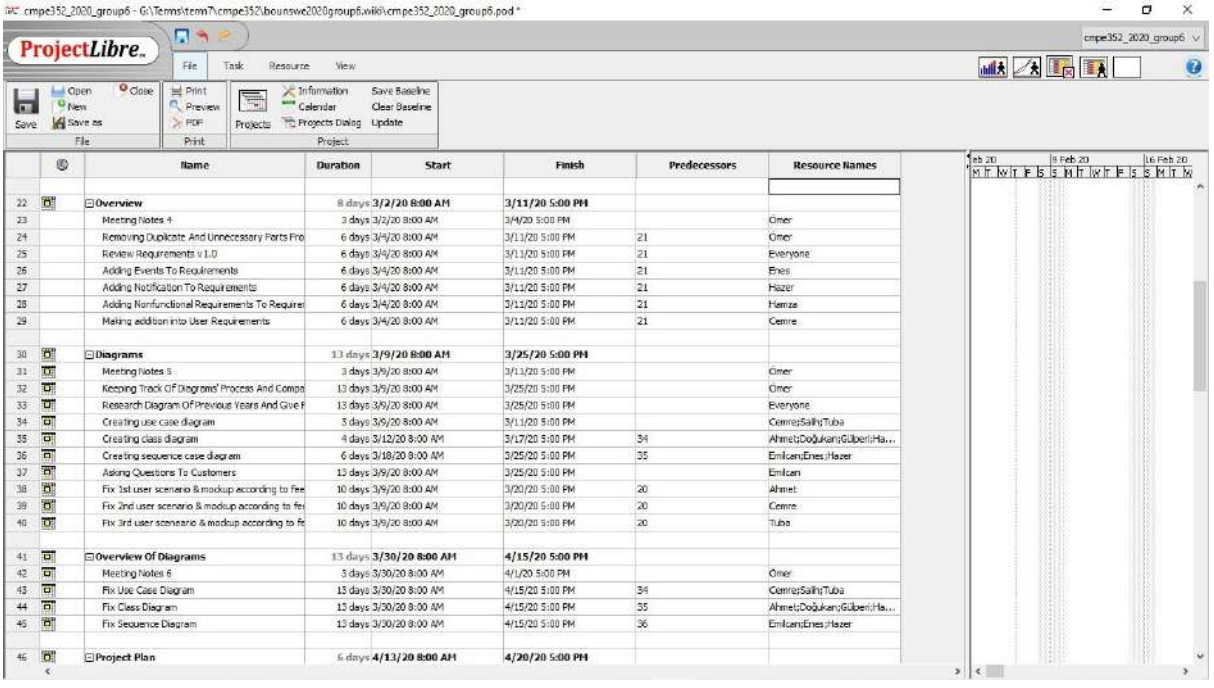


Be private

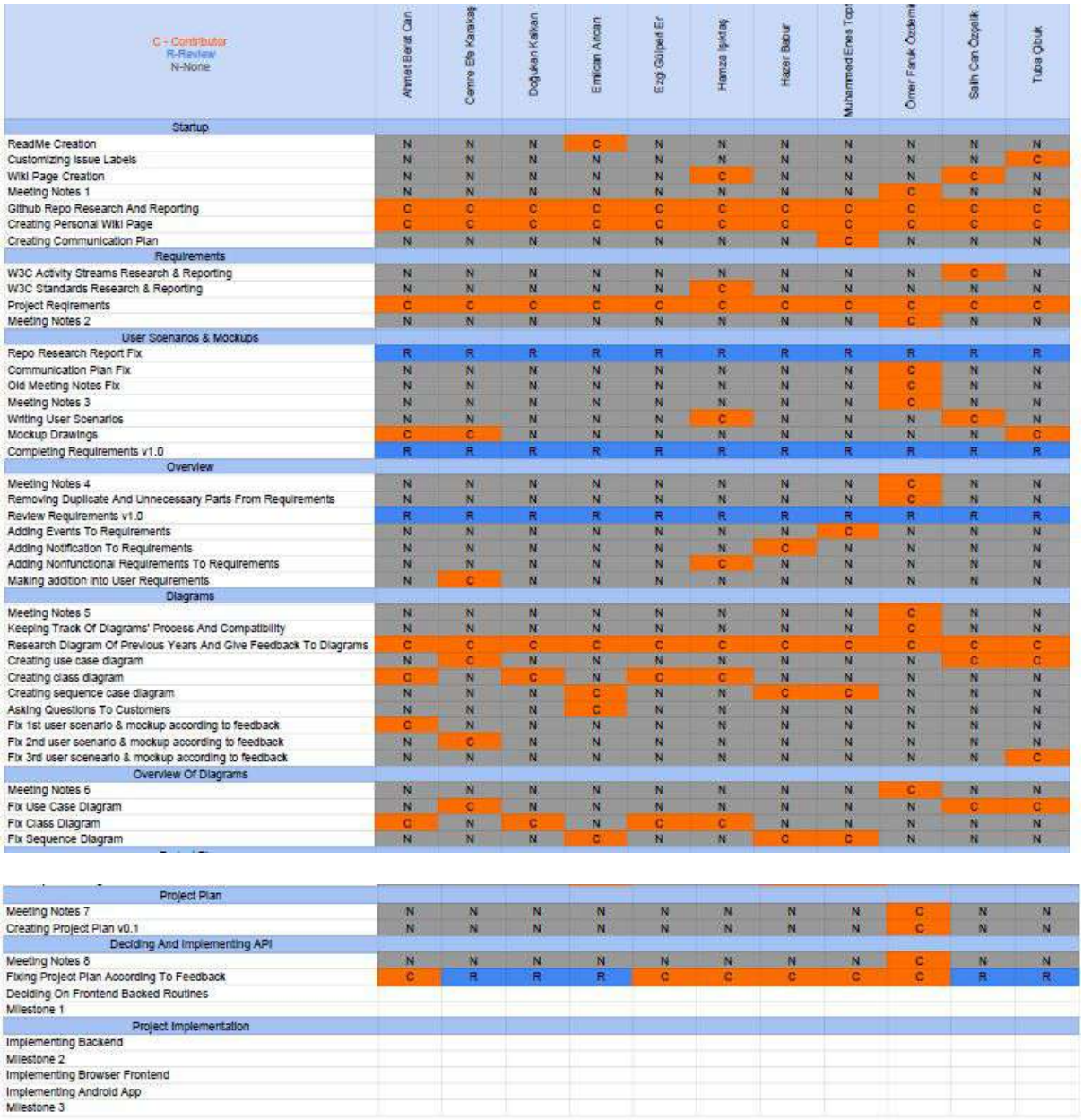


2.5 Project plan & RAM

1		**Startup**	6 günler	**10.02.2020 08:00**	**17.02.2020 17:00**		
2		ReadMe Creation	6 günler	10.02.2020 08:00	17.02.2020 17:00		Emlcan
3		Customizing Issue Labels	6 günler	10.02.2020 08:00	17.02.2020 17:00		Tuba
4		Wiki Page Creation	6 günler	10.02.2020 08:00	17.02.2020 17:00		Hamza;Salih
5		Meeting Notes 1	3 günler	10.02.2020 08:00	12.02.2020 17:00		Ömer
6		Github Repo Research And Reporting	6 günler	10.02.2020 08:00	17.02.2020 17:00		Everyone
7		Creating Personal Wiki Page	6 günler	10.02.2020 08:00	17.02.2020 17:00		Everyone
8		Creating Communicator Plan	6 günler	10.02.2020 08:00	17.02.2020 17:00		Enes
9		**Requirements**	6 günler	**17.02.2020 08:00**	**24.02.2020 17:00**		
10		W3C Activity Streams Research & Reporting	6 günler	17.02.2020 08:00	24.02.2020 17:00		Salih
11		W3C Standards Research & Reporting	6 günler	17.02.2020 08:00	24.02.2020 17:00		Hamza
12		Project Requirements	6 günler	17.02.2020 08:00	24.02.2020 17:00		Everyone
13		Meeting Notes 2	3 günler	17.02.2020 08:00	19.02.2020 17:00		Ömer
14		**User Scenarios & Mockups**	7 günler	**24.02.2020 08:00**	**03.03.2020 17:00**		
15		Repo Research Report Fix	6 günler	24.02.2020 08:00	02.03.2020 17:00	6	Everyone
16		Communication Plan Fix	6 günler	24.02.2020 08:00	02.03.2020 17:00	8	Ömer
17		Old Meeting Notes Fix	6 günler	24.02.2020 08:00	02.03.2020 17:00	5;13	Ömer
18		Meeting Notes 3	3 günler	24.02.2020 08:00	26.02.2020 17:00		Ömer
19		Writing User Scenarios	3 günler	25.02.2020 08:00	27.02.2020 17:00	12	Hamza;Salih
20		Mockup Drawings	3 günler	28.02.2020 08:00	03.03.2020 17:00	19	Ahmet;Cemre;Tuba
21		Completing Requirements v1.0	6 günler	25.02.2020 08:00	03.03.2020 17:00	12	Everyone
22		**Overview**	8 günler	**02.03.2020 08:00**	**11.03.2020 17:00**		
23		Meeting Notes 4	3 günler	02.03.2020 08:00	04.03.2020 17:00		Ömer
24		Removing Duplicate And Unnecessary Parts Fro	6 günler	04.03.2020 08:00	11.03.2020 17:00	21	Ömer
25		Review Requirements v1.0	6 günler	04.03.2020 08:00	11.03.2020 17:00	21	Everyone
26		Adding Events To Requirements	6 günler	04.03.2020 08:00	11.03.2020 17:00	21	Enes
27		Adding Notification To Requirements	6 günler	04.03.2020 08:00	11.03.2020 17:00	21	Hazer
28		Adding Nonfunctional Requirements To Requirer	6 günler	04.03.2020 08:00	11.03.2020 17:00	21	Hamza
29		Making addition into User Requirements	6 günler	04.03.2020 08:00	11.03.2020 17:00	21	Cemre



	Project Plan	**6 days**	**4/13/20 8:00 AM**	**4/20/20 5:00 PM**		
	Meeting Notes 7	3 days	4/13/20 8:00 AM	4/15/20 5:00 PM		Ömer
	Creating Project Plan v0.1	6 days	4/13/20 8:00 AM	4/20/20 5:00 PM	25	Ömer
	Deciding And Implementing API	**35 days**	**6/2/20 8:00 AM**	**7/20/20 5:00 PM**		
	Meeting Notes 8	3 days	6/2/20 8:00 AM	6/4/20 5:00 PM		Ömer
	Fixing Project Plan According To Feedback	6 days	6/2/20 8:00 AM	6/9/20 5:00 PM		Everyone
	Deciding On Frontend Backed Routines	35 days	6/2/20 8:00 AM	7/20/20 5:00 PM		
	Milestone 1	0 days	7/20/20 5:00 PM	7/20/20 5:00 PM	52	
	Milestone 2	0 days	6/2/20 8:00 AM	6/2/20 8:00 AM		
	Project Implementation Planning	**5 days**	**10/23/20 8:00 AM**	**10/29/20 5:00 PM**		
	Review Use Cases	5 days	10/23/20 8:00 AM	10/29/20 5:00 PM		Everyone
	Review Mock ups	5 days	10/23/20 8:00 AM	10/29/20 5:00 PM		Everyone
	Review Diagrams	5 days	10/23/20 8:00 AM	10/29/20 5:00 PM		Everyone
	Reorganizing Frontend & Backend Teams	5 days	10/23/20 8:00 AM	10/29/20 5:00 PM		
	Determining Implementation Process and timing	5 days	10/23/20 8:00 AM	10/29/20 5:00 PM		
	Milestone 1 & 2 Implementation	**45 days**	**10/30/20 8:00 AM**	**12/31/20 5:00 PM**		
	Implementing sign up facility	45 days	10/30/20 8:00 AM	12/31/20 5:00 PM		Backend Team
	Implementing search functionality	45 days	10/30/20 8:00 AM	12/31/20 5:00 PM		Backend Team
	Implementing creating & editng page functional	45 days	10/30/20 8:00 AM	12/31/20 5:00 PM		Backend Team
	Implementing notification and recommendation f	45 days	10/30/20 8:00 AM	12/31/20 5:00 PM		Backend Team
	Implementing follow & invitation mechanism	45 days	10/30/20 8:00 AM	12/31/20 5:00 PM		Backend Team
	Implementing other minor parts	45 days	10/30/20 8:00 AM	12/31/20 5:00 PM		Everyone
	Implementing frontend part	45 days	10/30/20 8:00 AM	12/31/20 5:00 PM		Frontend Team
	Mobile App Implementation	**30 days**	**10/30/20 8:00 AM**	**12/10/20 5:00 PM**		
	Deciding process	5 days	10/30/20 8:00 AM	11/5/20 5:00 PM		
	Implementing sign up page	30 days	10/30/20 8:00 AM	12/10/20 5:00 PM		Backend Team;Frontend Team
	Implementing search page	30 days	10/30/20 8:00 AM	12/10/20 5:00 PM		Backend Team;Frontend Team
	Implementing event page	30 days	10/30/20 8:00 AM	12/10/20 5:00 PM		Backend Team;Frontend Team
	Implementing notification and recommendation f	30 days	10/30/20 8:00 AM	12/10/20 5:00 PM		Backend Team;Frontend Team
	Implementing profile page	30 days	10/30/20 8:00 AM	12/10/20 5:00 PM		Backend Team;Frontend Team
	Implementing other minor parts	30 days	10/30/20 8:00 AM	12/10/20 5:00 PM		Backend Team;Frontend Team



C - Contributor R-Review N-None	Ahmet Berat Can	Cemre Efe Karakaş	Doğukan Kalkan	Emircan Arıcan	Ezgi Gülşen Er	Hamza Işıktaş	Hazer Babur	Muhammed Enes Topt	Ömer Faruk Özdemir	Salih Can Özçelik	Tuba Çıbık
Startup											
ReadMe Creation	N	N	N	C	N	N	N	N	N	N	N
Customizing Issue Labels	N	N	N	N	N	N	N	N	N	N	C
Wiki Page Creation	N	N	N	N	N	C	N	N	N	C	N
Meeting Notes 1	N	N	N	N	N	N	N	N	C	N	N
Github Repo Research And Reporting	C	C	C	C	C	C	C	C	C	C	C
Creating Personal Wiki Page	C	C	C	C	C	C	C	C	C	C	C
Creating Communication Plan	N	N	N	N	N	N	N	C	N	N	N
Requirements											
W3C Activity Streams Research & Reporting	N	N	N	N	N	N	N	N	N	C	N
W3C Standards Research & Reporting	N	N	N	N	N	C	N	N	N	N	N
Project Reqirements	C	C	C	C	C	C	C	C	C	C	C
Meeting Notes 2	N	N	N	N	N	N	N	N	C	N	N
User Soenarios & Mockups											
Repo Research Report Fix	R	R	R	R	R	R	R	R	R	R	R
Communication Plan Fix	N	N	N	N	N	N	N	N	C	N	N
Old Meeting Notes Fix	N	N	N	N	N	N	N	N	C	N	N
Meeting Notes 3	N	N	N	N	N	N	N	N	C	N	N
Writing User Soenarios	N	N	N	N	N	C	N	N	N	C	N
Mockup Drawings	C	C	N	N	N	N	N	N	N	N	C
Completing Requirements v1.0	R	R	R	R	R	R	R	R	R	R	R
Overview											
Meeting Notes 4	N	N	N	N	N	N	N	N	C	N	N
Removing Duplicate And Unnecessary Parts From Requirements	N	N	N	N	N	N	N	N	C	N	N
Review Requirements v1.0	R	R	R	R	R	R	R	R	R	R	R
Adding Events To Requirements	N	N	N	N	N	N	N	C	N	N	N
Adding Notification To Requirements	N	N	N	N	N	N	C	N	N	N	N
Adding Nonfunctional Requirements To Requirements	N	N	N	N	N	C	N	N	N	N	N
Making addition into User Requirements	N	C	N	N	N	N	N	N	N	N	N
Diagrams											
Meeting Notes 5	N	N	N	N	N	N	N	N	C	N	N
Keeping Track Of Diagrams' Process And Compatibility	N	N	N	N	N	N	N	N	C	N	N
Research Diagram Of Previous Years And Give Feedback To Diagrams	C	C	C	C	C	C	C	C	C	C	C
Creating use case diagram	N	C	N	N	N	N	N	N	N	C	C
Creating class diagram	C	N	C	N	C	C	N	N	N	N	N
Creating sequence case diagram	N	N	N	C	N	N	C	C	N	N	N
Asking Questions To Customers	N	N	N	C	N	N	N	N	N	N	N
Fix 1st user scenario & mockup according to feedback	C	N	N	N	N	N	N	N	N	N	N
Fix 2nd user scenario & mockup according to feedback	N	C	N	N	N	N	N	N	N	N	N
Fix 3rd user sceneario & mockup according to feedback	N	N	N	N	N	N	N	N	N	N	C
Overview Of Diagrams											
Meeting Notes 6	N	N	N	N	N	N	N	N	C	N	N
Fix Use Case Diagram	N	C	N	N	N	N	N	N	N	C	C
Fix Class Diagram	C	N	C	N	C	C	N	N	N	N	N
Fix Sequence Diagram	N	N	N	C	N	N	C	C	N	N	N

Project Plan											
Meeting Notes 7	N	N	N	N	N	N	N	N	C	N	N
Creating Project Plan v0.1	N	N	N	N	N	N	N	N	C	N	N
Deciding And Implementing API											
Meeting Notes 8	N	N	N	N	N	N	N	N	C	N	N
Fixing Project Plan According To Feedback	C	R	R	R	C	C	C	C	C	R	R
Deciding On Frontend Backed Routines											
Milestone 1											
Project Implementation											
Implementing Backend											
Milestone 2											
Implementing Browser Frontend											
Implementing Android App											
Milestone 3											

3. Evaluation of the status of deliverables and its impact on your project plan

COMMUNICATION PLAN

Firstly, we talked that we need a Slack group via Whatsapp. Then, we decided the time of the first meeting on Slack. After the first meeting in CmpE, we concluded on meeting time and communication plan. One of the our members created the wiki page of communication plan. We track the deadlines looking at issues on Github. If we have to say something urgent, we use Whatsap. After the Coronavirus break, we started to use Zoom. It helps us to share screen and communicate effectively. So that, we can plan our next works easily.

PROJECT REQUIREMENTS

Project Requirements include Glossary, Functional Requirements and Non-Functional Requirements. We divided Project Requirement topics among our team members. Then we chose 3 of our members to ask questions about the Project Requirements to our customer. After that, we finished the first version of our Project Requirements. We are also continuing to update the Project Requirements. Our members are free to change or add requirements when they see a need for it. Project Requirements are like the skeleton of the project. When we work on another new deliverable we must look at the requirements to check the consistency. Therefore, we make our project plan and other plans according to the Project Requirements.

SCENARIOS AND MOCKUPS

Scenarios and Mockups are important to understand the flow of the system. Two of our members created the scenarios according to Project Requirements. Three of our members created Mockups using MockFlow tool. We updated the scenarios and mockups according to feedback.

In one scenario, user uses the Android app version of our system. In other scenarios, users use Website version. The Scenarios and Mockups show clearly how to use and navigate on the platform. It shows basic steps. A customer can have an idea looking at these. We update the Scenarios and Mockups as well as other deliverables when there is a need.

UML SOFTWARE DESIGNS

We divided into three groups to create Use Case Diagram, Class Diagram, and Sequence Diagrams. Then we updated them according to feedback. Because of the Coronavirus Pandemic, we finished our tasks later than expected. But we were able to continue and finish our tasks.

Use Case Diagram – shows the user types and what they can do on our platform. It shows who shall access where. The directions of the arrows have a meaning. If the arrow towards to the main diagram that means this feature is extended. That means user can do one of that. If the arrow is from the main diagram that means this feature is included. User shall do every included ones to be successful.

Class Diagram – lists the classes, their fields and their functions. Also it shows the relation between classes. It will be helpful to code when we start coding. It organizes all the classes and fits them in a form. So it is very helpful for new deliverables and our project plan.

Sequence Diagrams – are like mockups but it also shows the inside of the flow. It shows the functions called when user does something. It really help us understand the structure.

PROJECT PLAN & RAM

It shows what our team members did up to now and what will be done next. It helps us to be more planned and shows us next plans so that we prepare ourselves to them.

3. Evaluation of Deliverables

Communication Plan

Firstly, we talked that we need a Slack group via Whatsapp. Then, we decided the time of the first meeting on Slack. After the first meeting in CmpE, we concluded on meeting time and communication plan. One of the our members created the wiki page of communication plan. We track the deadlines looking at issues on Github. If we have to say something urgent, we use Whatsap. After the Coronavirus break, we started to use Zoom. It helps us to share screen and communicate effectively. So that, we can plan our next works easily.

Project Requirements

Project Requirements include Glossary, Functional Requirements and Non-Functional Requirements. We divided Project Requirement topics among our team members. Then we chose 3 of our members to ask questions about the Project Requirements to our customer. After that, we finished the first version of our Project Requirements. We are also continuing to update the Project Requirements. Our members are free to change or add requirements when they see a need for it. Project Requirements are like the skeleton of the project. When we work on another new deliverable we must look at the requirements to check the consistency. Therefore, we make our project plan and other plans according to the Project Requirements.

Scenarios and Mockups

Scenarios and Mockups are important to understand the flow of the system. Two of our members created the scenarios according to Project Requirements. Three of our members created Mockups using MockFlow tool. We updated the scenarios and mockups according to feedback.

In one scenario, user uses the Android app version of our system. In other scenarios, users use Website version. The Scenarios and Mockups show clearly how to use and navigate on the platform. It shows basic steps. A customer can have an idea looking at these. We update the Scenarios and Mockups as well as other deliverables when there is a need.

UML Software Designs

We divided into three groups to create Use Case Diagram, Class Diagram, and Sequence Diagrams. Then we updated them according to feedback. Because of the Coronavirus Pandemic, we finished our tasks later than expected. But we were able to continue and finish our tasks.

Use Case Diagram – shows the user types and what they can do on our platform. It shows who shall access where. The directions of the arrows have a meaning. If the arrow towards to the main diagram that means this feature is extended. That means user can do one of that. If the arrow is from the main diagram that means this feature is included. User shall do every included ones to be successful.

Class Diagram – lists the classes, their fields and their functions. Also it shows the relation between classes. It will be helpful to code when we start coding. It organizes all the classes and fits them in a form. So it is very helpful for new deliverables and our project plan.

Sequence Diagrams – are like mockups but it also shows the inside of the flow. It shows the functions called when user does something. It really help us understand the structure.

Project Plan and RAM

It shows what our team members did up to now and what will be done next. It helps us to be more planned and shows us next plans so that we prepare ourselves to them.

4. Summary of Work Done by Members

Group Member	Contribution
Ahmet Berat Can	- Contributed in requirements. - Created mockup for scenario 2. - Contributed in creating class diagram. - Contributed in project plan - Took part in the back-end team - Integrated Search, Dashboard, and Home page to back-end. - Created a pull request named Integration_of_frontend_pages_to_backend and reviewed other members pull requests.

Doğukan Kalkan	- Helped create some parts of the Wiki Page. - Created my personal Wiki Page. - Learned Git. - Researched other GitHub Repositories. - Prepared the questions for customer meeting. - Met with the customer as the questioner. - Created the following classes for the class diagram: Request, Invitation, BaseWork, Project, Specification, Report. - Took part in the front-end team - Created the function CoronavirusByCountry in coronavirus_api.py that returns daily statistics about a country. - Created the unit test function isCoronavirusByCountryCorrect in unit_test_covid_19.py to check if the function mentioned above works properly. - Worked with team members cooperatively, informed them about my work. - Reviewed the work of my team members.
Emilcan Arıcan	-Prepared communication mediums -Attended Meetings -Researched GitHub Repositories. -Created personal wiki page -Contributed in user requirements -Contributed to and drew sequence diagram -Arranged customer meeting -Attended customer meeting -Took part in the front-end team

	-Designed and implemented; home page, academician search page and profile page of author. Also implemented the main template of the frontend pages. -Implemented the final version of the api/authorpublications endpoint. -Implemented the api/coinranks endpoint. -Prepared unit tests for api/coinranks and api/authorpublications. -Created pull requests. Made and requested reiviews.
Muhammed Enes Toptaş	-Attended meetings. -Researched informations about Github Repositories. -Contributed in creating communication plan. -Contributed in user requirements. Contributed in system requirements. -Created 2 of the sequence diagrams. -Created pull requests and issues. -Reviewed other team member's codes and pull requests. -Deployed the code onto AWS Server. -Created and then contributed to general API scheme. -Implemented first versions of `/api/search`, `/api/authorpublications`, `/api/publicationsearch` and `/api/authorstats` API endpoints. These endpoints make use of functions implemented in scholar_util. -Made some improvements to functions in scholar_util. -Further improved `/api/authorstats` API endpoint and `/search` Front-End endpoint.

	-Unit tested getAuthorCitationsStats in unit_test_scholarly.py -Contributed to unit test organization.

Ezgi Gülperi Er	-Attended meetings
	
	-Github repo research and reporting
	
	-Created personal wiki page
	
	-Contributed to user requirements
	
	-Contributed to and drew class diagram
	
	-Attended customer meeting
	
	-Contributed to Milestone 1 report
	
	-Implemented endpoints in backend.py: (/worldStats and /api/worldStats)
	
	-Contributed to html page templates (worldStats.html)
	
	-Created and reviewed pull requests
	
	-Created unit test to test getWorldStatistics() method in coronavirus_util.py
	
	-Implemented the methods (they are later extended and modified by other members) :
	
	`getAuthorsPublications(name)`
	
	`searchPublication(pub_name)`
	
	`getAuthorCitationStats(name)`

Tuba Çibuk	-Attended meetings -Created Issues -Github repo research and reporting -Drawed group logo -Created personal wiki page -Contributed in user requirements -Contributed in Mockups -Contributed in use cases -Contributed to endpoints (/coronavirusCountryLive, /api/coronavirusCountryLive) -Contributed to html pages (searchCountryName.html) -Contributed to unit tests (isJson() in unit_test_covid_19.py) -Created pull requests and made review
Hamza Işıktaş	-Attended meetings -Created personal wiki page. -Researched github repositories -Researched and created wiki page and wiki sidebar page. -Researched and shared informations about w3c standards -Contributed in requirements. -Created user scenarios. -Created class diagrams. -Contributed in creation of backend template. -Researched APIs -Created scholar_util to use scholarly API

	-Contributed to endpoints (/search, /api/profileData) -Created unittest (isGetAuthorCorrect) for endpoint /search -Contributed in the deployment of the code to AWS -Created pull requests and reviewed pull requests and merged them..
Hazer Babur	-Attended meetings -Searched Github repositories and reported interesting ones -Created personal wiki page -Contributed in user and system requirements -Created 3 of the sequence diagrams --Composed 451 related plan in Project Plan --Researched REST-APIs and contributed to thirdparty.py --Worked as a part of backend team --Implemented /api/search /coronavirus and /api/coronavirus endpoints --implemented coronavirus_summary_search() method in coronavirus.api.py --shaped final versions of scholar_unit.py methods --provided coronavirus.html file -created unit test for coronavirus_summary_search() method ---Created pull requests. Made and requested reiviews.

Ömer Faruk Özdemir	Managed Meetings, created Meeting Notes and created issues weekly. Arranged meeting topics and agendas. Contributed in creating communication plan. Contributed in user requirements. Kept track of Diagrams progress and compatibility. Created issues with distributable tasks for feedbacks. Kept track of general progress and deadlines. Created most of the Project Plan. Created backend template in flask. Helped Ahmet Berat to integrate frontend style into backend. Created structure of scholarly part of project with todos. Created a corona virus totalCaseNoPlotting endpoint, plot.png Write unit test isPlotDataFetchCorrect for coronavirus endpoint. Review some pull requests.
Salih Can Özçelik	Researched and created wiki page, wiki sidebar page. Researched and shared informations about W3C Activity Streams Research. Contributed in user requirements. Created user scenarios. Created use-case diagrams. Created template for milestone 1. Created the AWS user and tried to deploy code.

	Worked as a part of a front-end team. Researched template for frontend. Tested api_publicationsearch() via isGetAuthorCorrect() in scholarly_unit_test.py.
Cemre Efe Karakaş	-Attended meetings -Photoshopped self into the group photo -Created a personal wiki page -Created a files directory and migrated wiki images from personal repos to the project repo -Contributed in Project Requirements -Contributed in mock-ups and illustrations. -Made a research about previous years' diagrams -Contributed in the use-case diagram -Took part in the front-end team -Found base bootstrap templates to form the front-end on -Tried my best in giving a head-start to the front-end team by sharing my experiences with Html, Css and Jinja2 -Implemented a good amount of template logic and front-end pages. -Implemented /profile/<name> and api/profiledata endpoints. Helped render it on the front-end. -Modified some scholarly back-end functions to be able to link search results to user profiles. -Implemented the bitcoinprice and api/bitcoinprice endpoints, implemented its front-end. implemented a unit test to see if the api endpoint works as expected.

5. Evaluation of Tools

Github

A very well-known, widely used version control system. In our project, it is also used for communication purpose. One of the advantages is that since our project is open-source, GitHub lets us create and manage our own Wiki page which contains information and updates about our project and information about our group and its logistics. Another advantage that GitHub gives us is that group members can open issues with regards to the project and its progress.

ProjectLibre

ProjectLibre is an old fashioned Project Management software. It is hard to use and not very intuitive. Printing or saving as pdf is also very problematic. Learning and using is hard as well.

But since ProjectLibre is an open source, freely available project management software it is a great opportunity for people since it is one of the first freely available Project Management softwares.

MockFlow

It is used to make Mockups in one format. With this tool we were able to draw a basic Website User Interface and Android App user Interface. It also have different interface structures and objects based on the content, platform, and operating system.

Lucid Chart

Lucidchart is a web-based proprietary platform that is used to allow users to collaborate on drawing, revising and sharing charts and diagrams. It is quite intuitive and easy to work with. It is a well fit tool to create graphs and diagrams, especially as a group. Normally, the free product is pretty limited, but using an .edu email address you can apply to use the full product free of charge.

Slack

Slack is quite useful and robust. It was very easy to integrate it into our communication plan thanks to its short setup time. It has a user-friendly and simplistic interface. It has a long list of apps that can be added, so we can easily integrate different platforms into it (Google Drive etc.).One of the best features is channels and threads. It helps us to keep everything transparent without flooding other team members with unnecessary notifications. We can simply create channels for different topics and also use threads within conversations for running several conversations simultaneously. You can select to mute notifications for selected channels. Another great feature is the persistent state, once you log in after a break it shows you the last messages you have seen and that way you can easily catch up with all the conversations.Simplistic interface has a drawback. It is often hard to retrieve information that is buried in the sea of messages. And also sometimes multiple, simultaneous conversations can be difficult to keep up with. Even with separate channels, going back and forth through conversations can be time consuming and challenging.

Zoom

Zoom is an application that has been around for a long time now but never really got a widespread use until the recent Corona Virus pandemic. During the pandemic many workspaces and schools switched to remote classes or communication systems. Zoom was the main application used in those remote sessions and got a huge increase in user counts. We also decided to use Zoom in our meetings since due to school it was already installed in our computers and we already got familiar with it. So far it has been very useful, the connection is stabile most of the time and the application runs smoothly. Since everyone is on their computers it is also easier to reach out to important things while discussion is going on. However the chat section could be improved, it is in an archaic form now, it could and should be enriched with features.

docs.google.com

It allows us to edit same document simultaneously. In the previous milestone report that was hard to handle with using github, since commits might cause conflicts. That was hard to resolve the conflicts of an word document by using github.

Flask

It is a micro web framework written in Python. It allows setting up servers and creating APIs. If sole purpose is creating an API, then Flask_RESTful should be used as it handles all API related things and makes it quite easier to write and read and maintain. In this application, since we also served a front-end, we instead used Flask per se. Since this project was rather simple, Flask had an advantage of being smaller and easier to write compared to other frameworks, that's why we chose it. We also had some team members already knowing how to use it, so it was just the perfect solution for us.

6. API Documentation and URL

URL: ec2-54-229-68-140.eu-west-1.compute.amazonaws.com

POST api/search body = { "name" : "search parameter" } as JSON

Search by author name and return Author's features
Example Response:

```
{
    "author_search_result": [
        {
            "affiliation": "Universität Freiburg",
            "name": "Martin Heidegger",
            "url_picture": "https://scholar.google.com/citations?view_op=medium_photo&user=h_xwuikAAAAJ"
        },
        {
            "affiliation": "Mars44 Consulting @ Gmail . Com",
            "name": "martin alexander jones",
            "url_picture": "https://scholar.google.com/citations?view_op=medium_photo&user=ve2ce4QAAAAJ"
        },
        {
            "affiliation": "Kenneth S. Pitzer Distinguished Professor, University of California, Berkeley",
            "name": "Martin HEAD-GORDON",
            "url_picture": "https://scholar.google.com/citations?view_op=medium_photo&user=9Z52tZ4AAAAJ"
        },
        {
            "affiliation": "Full Professor of Experimental Physics, University College Dublin",
            "name": "Martin Grunewald",
            "url_picture": "https://scholar.google.com/citations?view_op=medium_photo&user=tuJ3TEcAAAAJ"
        },
        {
            "affiliation": "University of Pennsylvania",
            "name": "\"Martin Seligman\"",
            "url_picture": "https://scholar.google.com/citations?view_op=medium_photo&user=e771_EUAAAAJ"
        }
    ]
}
```


GET api/authorpublications

Actual data is gathered from a package called scholarly. Since it is still in development, there are some issues for instance it works a bit slow.

This end point returns most recent publications of a user. By default, it returns latest 5 publications. However, it can be also stated by using “range” argument. If publications of an author is less than “range” that is provided. It returns all the publications. It also returns author, title, year, summary, and, url of the publication.

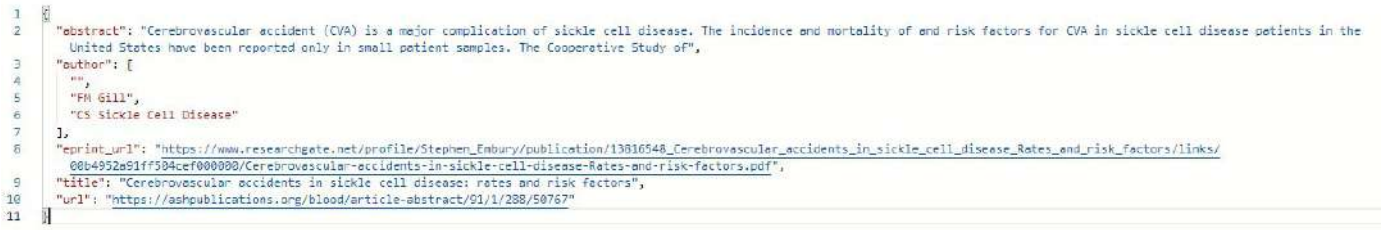
Example response:


```
▼ publications:
  ▼ 0:
      author:   "P Akan and NC Birben and M Bekbolet"
    ▼ summary:  "In this study, excitation-emission matrix (EEM) fluorescence spectral features
                were evaluated for the elucidation of surface interactions among diverse molecular
                size fractions of humic acid (0.45 µm filtered fraction, 100 kDa fraction and 30
                kDa fraction) and bare as well as anion-doped TiO2 specimens (Evonik P-25 and
                Hombikat UV-100). EEM constitute fluorescence spectra at many different excitation
                wavelengths providing a general point of view related to all features of organic
                matter within a particular spectral range. The objective of the present research
                was to compare the similarities and differences of the origin of different
                molecular size fractions of humic acid in the presence of bare and anion doped
                TiO2 (Evonik P-25 and Hombikat UV-100) specimens by excitation-emission matrix
                (EEM) fluorescence spectral properties."
    ▼ title:    "Spectrofluorometric Characterization of Adsorption of Different Molecular Size
                Fractions of Humic Acid onto Anion-Doped TiO2 Specimens"
      url:      "http://uruae.org/siteadmin/upload/AE0416249.pdf"
      year:     "unknown"
  ▶ 1:          {…}
  ▶ 2:          {…}
  ▶ 3:          {…}
  ▶ 4:          {…}
```


GET api/publicationsearch parameter(s): name= ”publication’s name”

The data is retrieved from a package scholarly, it is relatively slow so bear with us.

This endpoint searchs an article with given name and returns the info of the first one found. The returned info contains followings; abstract, authors, eprint_url, title and url to article. A name parameter must be sent with GET request



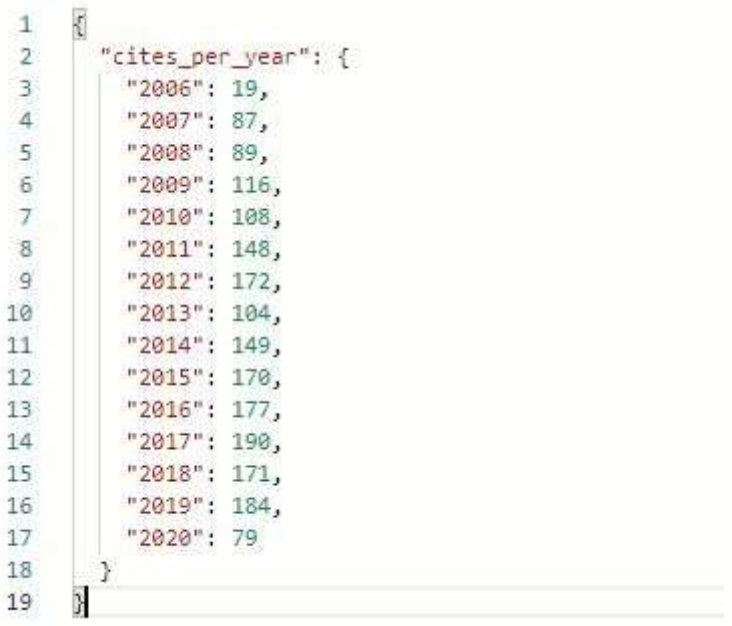
```
{
  "abstract": "Cerebrovascular accident (CVA) is a major complication of sickle cell disease. The incidence and mortality of and risk factors for CVA in sickle cell disease patients in the
    United States have been reported only in small patient samples. The Cooperative Study of",
  "author": [
    "",
    "FM Gill",
    "CS Sickle Cell Disease"
  ],
  "eprint_url": "https://www.researchgate.net/profile/Stephen_Embury/publication/13816548_Cerebrovascular_accidents_in_sickle_cell_disease_Rates_and_risk_factors/links/
    00b4952a91ff504cef000000/Cerebrovascular-accidents-in-sickle-cell-disease-Rates-and-risk-factors.pdf",
  "title": "Cerebrovascular accidents in sickle cell disease: rates and risk factors",
  "url": "https://ashpublications.org/blood/article-abstract/91/1/288/50767"
}
```


GET api/authorstats parameter(s): name= ”author’s name”

The data is retrieved from a package scholarly, it is relatively slow so bear with us.

This endpoint returns a specific authors citation numbers per year. A name parameter must be sent with GET request

Example Response: Given Name is “Can Kozcaz” -postman is used to test



```
{
  "cites_per_year": {
    "2006": 19,
    "2007": 87,
    "2008": 89,
    "2009": 116,
    "2010": 108,
    "2011": 148,
    "2012": 172,
    "2013": 104,
    "2014": 149,
    "2015": 170,
    "2016": 177,
    "2017": 190,
    "2018": 171,
    "2019": 184,
    "2020": 79
  }
}
```


.

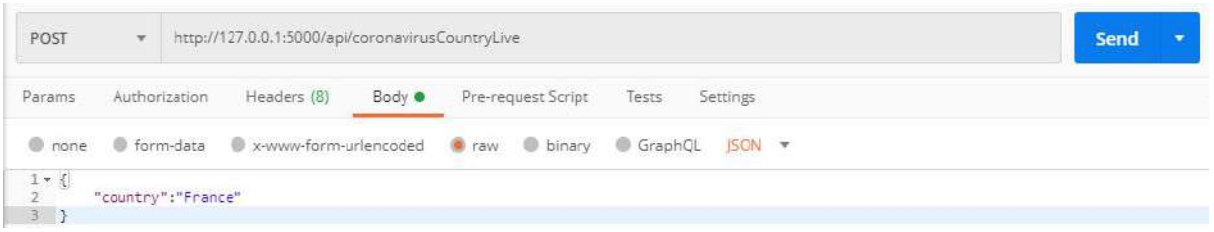
GET api/profiledata

The backend for this api gets the data from two of our own api endpoints and combines them together (getAuthors and getAuthorPublications with non-default parameters to better fit this endpoint’s needs). Then displays them on an endpoint.

See example output:

```
{
  "affiliation":"Research Assistant",
  "citedby":196,
  "id":"pAWgIFkAAAAJ",
  "interests":[
     "Advanced Oxidation Processes",
     "Photocatalysis",
     "Natural Organic Matter",
     "Spectroscopy"
  ],
  "name":"cemre birben",
  "publications":[
     {
        "author":"P Akan and NC Birben and M Bekbolet",
        "summary":"In this study, excitation-emission matrix (EEM) fluorescence spectral features were evaluated for the elucidation of surface interactions among diverse molecular
        "title":"Spectrofluorometric Characterization of Adsorption of Different Molecular Size Fractions of Humic Acid onto Anion-Doped TiO2 Specimens",
        "url":"http://uruae.org/siteadmin/upload/AE0416249.pdf",
        "year":"unknown"
     },
     {
        "author":"U\u011fur KURT and Fatih \u0130LHAN and N Cemre B\u0130RBEN and K\u00fcbra ULUCAN and M Talha",
        "summary":"Because of its complex structure and high pollutant concentrations, leachate is the leading wastewater that is difficult to be treated. It contains many pollutant
        "title":"INVESTIGATION OF TREATABILITY OF LEACHATES COMBINING WITH DOMESTIC WASTEWATER BY ELECTROCOAGULATION PROCESS",
        "url":"http://www.eds.yildiz.edu.tr/ArticleContent/Journal/sigma/Volumes/2011/Issues/TURKAY%202009%20T%C3%9CRK%C4%B0YE`DE%20KATI%20ATIK%20Y%C3%96NET%C4%B0M%C4%B0%20SEMPOZYUM
        "year":"2011"
     },
     {
        "author":"Nazmiye Cemre Birben and Miray Bekbolet",
        "summary":"Photolytic/photocatalytic degradation of synthetic ROC samples was evaluated. Role of various photocatalysts on ROC degradation was evaluated. Molecular size frac
        "title":"Molecular Size Distribution Profiles of Organic Matrix in Reverse Osmosis Concentrate Under Oxidative and Non-oxidative Conditions",
        "url":"https://link.springer.com/chapter/10.1007/978-3-030-13068-8_63",
        "year":"2020"
     }
  ],
  "url_picture":"https://scholar.google.com/citations?view_op=medium_photo&user=pAWgIFkAAAAJ"
}
```


POST api/coronavirusCountryLive



In order to get a correct response parameters shall be like above. If the country name is typed wrongly it returns nothing.

The response of this request is:


```
{
   "param": "France",
   "results": [
      {
         "Cases": 1634,
         "City": "",
         "CityCode": "",
         "Country": "France",
         "CountryCode": "FR",
         "Date": "2020-05-26T00:00:00Z",
         "Lat": "-12.83",
         "Lon": "45.17",
         "Province": "Mayotte",
         "Status": "confirmed"
      },
      {
         "Cases": 18,
         "City": "",
         "CityCode": "",
         "Country": "France",
         "CountryCode": "FR",
```

```
        "Date": "2020-05-26T00:00:00Z",
        "Lat": "-20.9",
        "Lon": "165.62",
        "Province": "New Caledonia",
        "Status": "confirmed"
      },
      {
        "Cases": 60,
        "City": "",
        "CityCode": "",
        "Country": "France",
        "CountryCode": "FR",
        "Date": "2020-05-26T00:00:00Z",
        "Lat": "-17.68",
        "Lon": "149.41",
        "Province": "French Polynesia",
        "Status": "confirmed"
      },
      {
        "Cases": 161,
        "City": "",
        "CityCode": "",
        "Country": "France",
        "CountryCode": "FR",
        "Date": "2020-05-26T00:00:00Z",
        "Lat": "16.25",
        "Lon": "-61.58",
        "Province": "Guadeloupe",
        "Status": "confirmed"
      },
      {
        "Cases": 197,
        "City": "",
        "CityCode": "",
        "Country": "France",
        "CountryCode": "FR",
        "Date": "2020-05-26T00:00:00Z",
        "Lat": "14.64",
        "Lon": "-61.02",
        "Province": "Martinique",
        "Status": "confirmed"
      }
    ]
}
```


“Cases” shows the current confirmed cases of provinces of France.

GET api/coronavirus

A summary of new and total cases per country updated daily.

Example Response


```
{
    "Country": "Afghanistan",
    "CountryCode": "https://www.countryflags.io/AF/shiny/64.png",
    "Date": "2020-05-27T11:30:14Z",
    "NewConfirmed": 658,
    "NewDeaths": 1,
    "NewRecovered": 31,
    "Slug": "afghanistan",
    "TotalConfirmed": 11831,
    "TotalDeaths": 220,
    "TotalRecovered": 1128
},
{
    "Country": "Albania",
    "CountryCode": "https://www.countryflags.io/AL/shiny/64.png",
    "Date": "2020-05-27T11:30:14Z",
    "NewConfirmed": 25,
    "NewDeaths": 1,
    "NewRecovered": 8,
    "Slug": "albania",
    "TotalConfirmed": 1029,
    "TotalDeaths": 33,
    "TotalRecovered": 803
},
```


GET api/worldStats
Returns total world statistics by case.

Example Output:


```
{
    "world_stats": [
        {
            "total_confirmed": 5677213,
            "total_deaths": 357189,
            "total_recovered": 2286305
        }
    ]
}
```


POST api/coronavirusByCountry
Returns all cases by type and by province in a specific country

```
"param": "China",
"results": [
    {
        "active": 10,
        "confirmed": 10,
        "date": "2020-01-22T00:00:00Z",
        "deaths": 0,
        "province": "Zhejiang",
        "recovered": 0
    },
    {
        "active": 2,
        "confirmed": 2,
        "date": "2020-01-22T00:00:00Z",
        "deaths": 0,
        "province": "Shandong",
        "recovered": 0
    },
```


GET api/coinranks

I have used *api.coinranking.com* in order to get the data that I need. I had to strip most of the part of the data to serve more clean and summarized version of it. And sake of simplicity. I have just took the most popular 10 cryptocurrencies to return into my own API.

Returns current ranking of cryptocurrencies. For each currency, it also returns name, price, rank, symbol (short-name), and url.

Example response:

GET api/bitcoinprice

I parsed coindesk's endpoint on https://api.coindesk.com/v1/bpi/currentprice.json to get data in json format. I then filtered out some unnecessary information.

I created an api endpoint that broadcasts the filtered out data.

See example output:

```
{
  "disclaimer": "This data was produced from the CoinDesk Bitcoin Price Index (USD).
  "rates": {
    "EUR": {
      "code": "EUR",
      "description": "Euro",
      "rate": "8,351.2948",
      "rate_float": 8351.2948,
      "symbol": "&euro;"
    },
    "GBP": {
      "code": "GBP",
      "description": "British Pound Sterling",
      "rate": "7,446.8311",
      "rate_float": 7446.8311,
      "symbol": "&pound;"
    },
    "USD": {
      "code": "USD",
      "description": "United States Dollar",
      "rate": "9,157.2721",
      "rate_float": 9157.2721,
      "symbol": "&#36;"
    }
  },
  "time_updated": "May 27, 2020 19:17:00 UTC"
}
```