Group 9 Milestone 1 Report

Executive Summary

We are designing a product, which will be used to teach, learn and make exercises on languages. Mainly, it is a language learning platform. Our project mainly consists of two parts: designing the project on an abstract level and implementation. We are using a waterfall development model during the process. So far we have designed the basics of our project according to the need and requirements of the user and now developing it, writing code.

We are using Java Spring Framework for our backend development, React for frontend and also Java for Android. We are divided into three sub-groups for each of these development lines and merging our deployments with each other in our main application. We scheduled a three months development process from now. (this milestone)

We are all bunch of friends taking that course and developing the project simultaneously. So far we have achieved most of our deadlines and goals. We hope to keep it up and stay tuned for new things that come in the way.

2.md

List Of Deliverables

Team	Deliverables
Android	- Register Page - Login Page - Language selection page - Proficiency exam page - Exam result page
Frontend	- Register Page - Login Page - Language selection page - Proficiency exam page - Exam result page
Backend	- Register User Endpoint - Login user endpoint - Greeting endpoint - Content endpoints - Grading Endpoints - Backend API documentation and swagger UI

Status and Evaluation of Deliverables

Android

Register Page

The register page in Android works as expected. Items 1.1.3.1.1. (Users shall give their email address while registering.), 1.1.3.1.2. (Users shall give their names while registering.), 1.1.3.1.3. (Users shall set up a password while registering to the system) of project requirements list have been implemented in this page. It is connected to the backend and the input is checked.

Login Page

The login page in Android works as expected. The item 1.1.3.2.1.(Users shall be able to login to the system using their email address and their passwords.) of project requirements list has been implemented in this page. The only problem is that when mobile data is used instead of wifi, there are some connection issues. The bug about is opened and the Android team will solve this question shortly.

Language Selection Page

This page is also working perfectly. Only some UI changes need to be done on the language icons. With this page one of the use cases has been realized and item 1.2.3.1. (The system shall support multiple languages.) of requirements has been partly realized; however, for Milestone-1 there is only a test for English Language.

Proficiency Exam Page

Questions are retrieved from the backend and displayed with no problem. The only problem is that this page is not consistent with the frontend version and also requirement 1.1.1.1.2.3. (After every question, the users shall be able to see the correct answer). In Android, the correct answers are not shown and the user only sees the result. Android team will change it and make it consistent with frontend version.

Exam Result Page

Exam results are calculated and updated in the backend. We need to show the result in a more detailed way. Currently, only the A1, A2, etc. level is shown but some level information like intermediate, advanced etc needs to shown as well.

Frontend

Register Page

The register page in the frontend works as expected. Items 1.1.3.1.1. (Users shall give their email address while registering.), 1.1.3.1.2. (Users shall give their names while registering.), 1.1.3.1.3. (Users shall set up a password while registering to the system) of project requirements list have been implemented in this page. It is connected to the backend and the input is checked.

Login Page

The login page in the frontend works as expected. The item 1.1.3.2.1.(Users shall be able to login to the system using their email address and their passwords.) of project requirements list has been implemented in this page. Some UI polish is needed in the frontend generally as the current version seems a little simple.

Language Selection Page

This page is also working perfectly. With this page one of the use cases has been realized and item 1.2.3.1. (The system shall support multiple languages.) of requirements has been partly realized. Currently, we are giving three options to users but only English is working so we need to work on that as well.

Proficiency Exam Page

Questions are retrieved from the backend and displayed with no problem. Unlike the android version, the correct answers are shown after each question by making the correct answer red in the frontend. Thus, the item 1.1.1.1.2.3. (After every question, the users shall be able to see the correct answer) of project requirements list has been fulfilled.

Exam Result Page

Exam results are calculated and updated in the backend. We need to show the result in a more detailed way. Currently, only the A1, A2, etc. level is shown but some level information like intermediate, advanced etc needs to shown as well. The result is shown in a pop-up so.

Backend

Test/Greeting Endpoint

This endpoint was for testing if the API connection works and is no longer needed. But it was very helpful and still actively working.

User Endpoints

Here and endpoint for registering and login. It works as expected and returns the user ID. In case of a wrong input, a response with error code is returned.

Content Endpoints

Content endpoints are currently our most functional endpoints. Here we can get all the languages list, get all exercises, get the proficiency exam questions of one language, create/delete exercises. The endpoint is working and is used actively by frontend and Android.

Grading Endpoints

The users' grades on a language are available in this endpoint. Also, when the user solves the test, the grades posted to this endpoint.

API Documentation

Our API documentatşon can be found in our master branch under Rest API Documentation section and the documentation is updated regularly to contain the latest changes. There are examples of successful and unsuccessfull requests.

5.md

Member	Work
Egemen	Worked solely on web application. - Implemented register page. - Linked register and login pages together. - Implemented "forgot password" page. This is a mock form. - Created a simple page to check API by hitting `/greeting` endpoint. This is the first API communication.
Halit	Worked on web application and devops. - Implemented frontend code structure, navigation flows, API calls. - Implemented Login, Language Selection, Proficiency Exam Pages. - Helped Ahmet deploy backend and create a AWS deployment flow for backend. - Deployed frontend (master/frontend) and next@frontend (frontend/frontend) on AWS S3. - Created Travis configuration for CI/CD for both frontend, next@frontend, backend.
İbrahim	Worked on Android application. - Created a simple greeting page for testing backend's `/greeting` endpoint for first week. - Implemented initial structure of the android app which are login and register pages. - Connected login&register pages to backend via `/users/login` , `/users/register` . - Implemented structure(not UI) of QuestionView and GradeView pages which performs Proficiency Exam's activity and grade display. - Connected Proficiency Exam's results to backend via `/grades/add` .
İrem	Worked on Android application. - Implemented language list page with Gamze. - Fixed the bug about displaying the language list. - Changed and set up a presentable UI for all of the application pages including main activity, login-register pages, language list display, and proficiency exam pages.
Gamze	Worked on Android application. - In the greeting page, the response was taken as JSONObject and parsed. - Implemented language list page with Irem.
Ahmet	Worked on Backend. - Created the initial basic backend project with Spring Framework. - Implemented a testing endpoint for Android and Frontend developers to connect. - Created a Postgresql DB instance on AWS RDS. - Deployed the project and created an AWS deployment flow for backend, with Halit. - Implemented login and register endpoints, services. - Implemented languages endpoint. - Implemented grade submitting and retrieving endpoints, services. - Implemented user retrieving endpoint for profile page. - Implemented whole exercise controller with exercise retrieving, creating and deleting endpoints, services. - Implemented proficiency exam retrieve endpoint. - Added Swagger to the project. - Created the DB structure and wrote appropriate sql queries on project repositories.
Emirhan	Worked solely on web application / frontend. - Implemented some of login page. - Reviewed code in frontend branch. - Helped teammates.

5.md

Requirements

Glossary

- **User:** A person using the app/web platform of the project.
  - **Guest:** A user who is not registered yet.
  - **Registered:** A user who has signed up to the platform.
  - **Admin:** A user with special privileges.
- **Language:** One of English/Turkish/Chinese or other provided language in the platform.
- **Learning Material:** Content in one of the categories below, in one of the types below, in a given language used to teach users that language.
  - **Categories:** Categories of a learning material, one of Listening, Reading, Grammar, Vocabulary or Writing.
    - **Listening:** Materials relating to improving listening skills in a given language.
    - **Reading:** Materials relating to improving reading skills in a given language.
    - **Grammar:** Materials relating to improving grammar skills in a given language.
    - **Vocabulary:** Materials relating to improving vocabulary skills in a given language.
    - **Writing:** Materials relating to improving writing skills in a given language.
  - **Types:** Types of a learning material, one of Notes, Assignment, Exercise or Exam.
    - **Notes:** Materials that present notes about the content.
    - **Assignments:** Materials that require user to write a long answer(paragraph, essay, etc.) and another user to review & evaluate.
    - **Exercises:** Materials that require user to answer questions which can be automatically graded.
- **Proficiency Exam:** A special exam consisting of questions used to evaluate the expertise of a user in a given language if the user wants to start directly from any level higher than `A1` .
- **Achievement:** A user can get special labels according to accomplishing some tasks with given conditions, such as in 2 minutes.
- **Progress of Learning:** The statistics about the users' learning history of a given language i.e. accomplished exercises, assignments, duration.
- **Interaction:** Users can interact with each other in either one of the ways below.
  - **Communication:** A user can send a request to another user to chat privately.
    - **Request:** When a user sends a chat request to another user, chatting only starts after receiver user accepts the request.
  - **Review:** The process of a user grading another user's writing assignment, and providing feedback to that user related to the assignment.
    - **Feedback:** An explanation of how the user can do better or an error found in one's writing assignment.
    - **Annotation:** A user can add annotation to the writing assignment he/she is reviewing to give feedback to the reviewed.
- **Rating:** A user can rate other users he/she interacted based on the related interaction.
- **Comment:** A user can comment about other users he/she interacted based on the related interaction.
- **Level of Expertise:**: The users' level of proficiency shown as either `A1` , `A2` , `B1` , `B2` , `C1` or `C2` in a given language.
- **Search System:** A system allowing users to search for contents in a given language.
  - **Basic Search System:** A search system based on keywords and semantic search.
  - **Advanced Search System:** A search system that allows search and filter features by type, difficulty and tags.
- **Contribution System:** Users can upload new learning materials and suggest them to be added to the system, or they can suggest new tags to existing material.
  - **Verification:** Admins can verify suggested tags to existing material to add them to the system. Also, for suggested new user uploaded materials, either a specified amount of users can support the suggestion to approval or an admin user can verify the material by himself/herself.

1. Functional Requirements

1.1. User Requirements

- **1.1.1. Users**
  - 1.1.1.1. There will be three types of users.
  - **1.1.1.1.1. Guests**
    - 1.1.1.1.1.1. Guest users can access 5 exercises for each type of learning materials(except writing) before being prompted for registering.
  - **1.1.1.1.2. Registered Users**
    - 1.1.1.1.2.1. These users should access to materials anytime and anywhere.
    - 1.1.1.1.2.2. The users who want to start from higher level, shall take a proficiency exam. Users shall be able to see their exam results and see the content of corresponding level of their scores.
    - 1.1.1.1.2.3. After every question, the users shall be able to see the correct answer.
    - 1.1.1.1.2.4. Users shall be able to send their essays to other users for grading. Grading includes feedback as well as grade.
    - 1.1.1.1.2.5. If two users interact, they shall be able to rate and comment on each other.
    - 1.1.1.1.2.6. Users shall be able to see their learning process such as completed exercises, grade achievements for each language.
    - 1.1.1.1.2.7. Users should be able to upload their suggestion of some learning materials.
    - 1.1.1.1.2.8. Users should be able to declare whether they want to review essays or not.
    - 1.1.1.1.2.9. Users should be able to report inappropriate behavior and sensitive content.
    - 1.1.1.1.2.10. Users' profile pages should include the review count, rating, achievements, uploaded contents and comments about them.
  - **1.1.1.1.3. Administrators**
    - 1.1.1.1.3.1. Administrators shall handle reports and take necessary actions.
    - 1.1.1.1.3.2. Administrators shall be able to accept or reject a suggested content.
- **1.1.2. Communication**
  - 1.1.2.1. Users shall be able to communicate over a messaging channel.
    - 1.1.2.1.1. Two users can use the messaging service if and only if one sends a request for communication and the other one accepts. Users shall be able to see their message requests and accept or reject them.
- **1.1.3. Login and Sign-up**
  - 1.1.3.1. Unregistered users shall be able to register after giving the necessary information.
    - 1.1.3.1.1. Users shall give their email address while registering.
    - 1.1.3.1.2. Users shall give their names while registering.
    - 1.1.3.1.3. Users shall set up a password while registering to the system.
  - 1.1.3.2. Registered users shall be able to login to the system.
    - 1.1.3.2.1. Users shall be able to login to the system using their email address and their passwords.
    - 1.1.3.2.2. The users that had forgotten their passwords shall be able to set up a new password with a verification e-mail.

1.2. System Requirements

- **1.2.1. Learning Materials**
  - 1.2.1.1. There should be learning materials in different languages.
  - 1.2.1.2. The learning materials should be mapped into 5 different categories: listening, reading, grammar, vocabulary and writing.

    - 1.2.1.3. The materials uploaded by users exist in the system after verification.
    - 1.2.1.4. The materials can have one or more resources with them such as images, sounds etc.
- **1.2.2. Assignments and Exams**
    - 1.2.2.1. Listening, reading, grammar and vocabulary categories of assignments or exams will be graded automatically by the system.
    - 1.2.2.2. A user should be able to pick the person who shall grade the writing assignment.
    - 1.2.2.3. System provides the answers after every question in exercises.
    - 1.2.2.4. Writing assignments should be uploaded by tags in order for the system to suggest reviewers.
- **1.2.3. Languages**
    - 1.2.3.1. The system shall support multiple languages.
- **1.2.4. Recommendation**
    - 1.2.4.1. For grading writing assignments, the system shall recommend a selection of users.
        - 1.2.4.1.1. When tags of the writing assignment and the tags that the reviewer user previously reviewed are similar, the system recommends that user for grading.
- **1.2.5. Annotation**
    - 1.2.5.1. The system shall allow the annotation content.
- **1.2.6. Rating**
    - 1.2.6.1. The system shall support rating and comments only between interacting users.
- **1.2.7. Contribution**
    - 1.2.7.1. New learning materials should be suggested by the users who upload their suggested materials.
    - 1.2.7.2. Acceptance of the suggested material is up to the administrator.
- **1.2.8. Searching**
    - 1.2.8.1 The system shall support two types of search for content: basic and advanced.
    - 1.2.8.1.1. Basic search only uses keywords for searching.
    - 1.2.8.1.2. Advanced search shall filter the content by difficulty, tag, type and like count.
    - 1.2.8.2 The system shall support user search.
    - 1.2.8.2.1. Users can be searched by their name, ratings' value and count, previously reviewed tags and language expertise.
- **1.2.9. Tagging**
    - 1.2.9.1 The system should support a set of tags with every writing assignment which informs the user about the topics that are discussed.
    - 1.2.9.2 The system should suggest reviewers for writing assignments considering tags of the assignment and the reviewer.
- **1.2.10. Liking and Reporting**
    - 1.2.10.1. Users should be able to like any content, this increases the content's rating.
    - 1.2.10.2. Users should be able to report any content for inappropriate, offensive or other reasons. Admins shall handle the situation from then on.

2. Non-Functional Requirements

2.1. Security

- 2.1.1. User data shall be protected and used according to LAW ON THE PROTECTION OF PERSONAL DATA
- 2.1.2. The personal information, contact information, copyrighted contents, license issues and everything related to these paradigms should be respected and considered.
- 2.1.3. System shall be protected against SQL injection.
- 2.1.4. System should use encryption for personal messages between users.
- 2.1.5. Storing of passwords should conform to security standards such as hashing.

2.2. Reliability

- 2.2.1. The system shall serve to 300 users without breaking.

2.3. Availability

- 2.3.1. Project shall be available on both Android and Web platforms.
- 2.3.2. The application should be deployable on a manually configurable remote server.

2.4. Protocols & Standards

- 2.4.1. The system shall support the W3C Web Annotation Data Model.

2.5. Performance

- 2.5.1. The system shall respond to 100 requests per second.
- 2.5.2. Maximum response time shall be at most 500 ms.

📖 6.md

Language Learning Platform REST API Documentation

Test Endpoint

Greeting

This is created for testing purpose, say hi

```
GET /greeting?name=group9
```

```
{
  "id": 3,
  "content": "Hello, group9!"
}
```

User Endpoints

Get User By Id

Request Content: id

Response Content: The user with corresponding id if it exists, null otherwise.

Example Request

```
GET /users/get?id=2
```

Example Response

```
{
  "id": 2,
  "email": "ahmettest@testtest.coam",
  "password": "9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08",
  "firstName": "ahmet",
  "lastName": "test"
}
```

Register User

Request Content: e-mail, password, firstName, lastName

Response Content: The created user with 200 status code if successful. The reason with 400(Bad Request) if unsuccessful.

Example Request-1

```
POST /users/register
```

```
{
  "email": "test@test.com",
  "password": "123456",
  "firstName": "testname",
  "lastName": "testsurname"
}
```

Example Response-1

```
{
  "status": 200,
  "explanation": null,
  "data": {
    "email": "test@test.com",
    "password": "123456",
    "firstName": "testname",
    "lastName": "testsurname"
  }
}
```

Example Request-2

```
POST /users/register
```

```
{
  "email": "testtest.com",
  "password": "123456",
  "firstName": "testname",
  "lastName": "testsurname"
}
```

Example Response-2

```
{
  "status": 400,
  "explanation": "Invalid email",
  "data": null
}
```

Example Request-3

```
POST /users/register
```

```
{
  "email": "somebodyelsesemailaddress@test.com",
  "password": "123456",
  "firstName": "testname",
  "lastName": "testsurname"
}
```

Example Response-3

```
{
  "status": 400,
  "explanation": "This e-mail has already been registered",
  "data": null
}
```

Login User

Request Content: e-mail, password

Response Content: The user with 200 status code if successful. The reason with 400(Bad Request) if unsuccessful.

Example Request-1

```
POST /users/login
```

```
{
  "email": "test@fortest.com",
  "password": "123456"
}
```

Example Response-1

```
{
  "status": 200,
  "explanation": null,
  "data": {
      "id": 5,
      "email": "test@fortest.com",
      "password": "123456",
      "firstName": "testname",
      "lastName": "testsurname"
  }
}
```

Example Request-2

```
POST /users/login
```

```
{
  "email": "wrongemailaddress@wrong.email",
  "password": "wrongpassword"
}
```

Example Response-2

```
{
  "status": 400,
  "explanation": "Wrong credentials",
  "data": null
}
```

Content Endpoints

Get All Available Languages List

Request Content: none

Response Content: The list of available languages

Example Request

```
GET /contents/languages
```

Example Response

```
{
  "status": 200,
  "explanation": null,
  "data": [
    "English",
    "Turkish",
    "Italian"
  ]
}
```

Get All Exercises List

Request Content: none

Response Content: The list of all exercises of all languages

Example Request

```
GET /contents/all
```

Example Response

```
{
    "status": 200,
    "explanation": null,
    "data": [
        {
            "languageId": 1,
            "typeId": 1,
            "imageUrl": null,
            "soundUrl": null,
            "question": "what is your name?",
            "optionA": "my name is..",
            "optionB": "your name is..",
            "optionC": "his name is...",
            "optionD": "her name is...",
            "correctAnswer": 1
        },
        {
          ...
        },

        ...
    ]
}
```

Get Proficiency Exam

Request Content: Language name as request parameter. Language names should be in English and capitalized.

Response Content: 10 questions from given language

Example Request

```
GET /contents/prof?language=English
```

Example Response

```
{
    "status": 200,
    "explanation": null,
    "data": [
        {
            "languageId": 1,
            "typeId": 1,
            "imageUrl": null,
            "soundUrl": null,
            "question": "what is your name?",
            "optionA": "my name is..",
            "optionB": "your name is..",
            "optionC": "his name is...",
            "optionD": "her name is...",
            "correctAnswer": 1
        },
        {
          ...
        },

        ...
    ]
}
```

Example Request

```
GET /contents/prof?language=english
```

Example Response

```
{
    "status": 400,
    "explanation": "Language not found.",
    "data": null
}
```

Create exercise

Request Content:

languageId(1 for English, 2 for Turkish, 3 for Italian)

typeId(1 for Grammar, 2 for vocabulary, 3 for reading, 4 for listening)

imageUrl(not necessary for proficiency, just delete)

soundUrl(not necessary for proficiency, just delete)

questionBody(As it sounds)

optionA, optionB, optionC, optionD(as it sounds)

correctAnswer(1 for A, 2 for B etc)

Only image url and sound url can be null

Response Content: Exercise itself

Example Request

```
POST /contents/add
```

```
{
  "correctAnswer": 4,
  "languageId": 1,
  "optionA": "Bean",
  "optionB": "Potato",
  "optionC": "Bread",
  "optionD": "Apple",
  "questionBody": "Which one of these is a fruit?",
  "typeId": 2
}
```

Example Response

```
{
    "status": 200,
    "explanation": null,
    "data": {
        "correctAnswer": 4,
        "imageUrl": null,
        "languageId": 1,
        "optionA": "Bean",
        "optionB": "Potato",
        "optionC": "Bread",
        "optionD": "Apple",
        "questionBody": "Which one of these is a fruit?",
        "soundUrl": null,
        "typeId": 2
```

```
      }
}
```

Delete exercise

Request Content:

Exercise id

Response Content: none

Example Request

```
GET /contents/delete?id=3
```

Example Response

```
{
    "status": 200,
    "explanation": null,
    "data": null
}
```

Grade Endpoints

Get Grade by UserId and LanguageId

Request Content: User id and language id

Response Content: Grade

Example Request

```
GET /grades/get?userId=3&languageId=1
```

Example Response

```
{
  "status": 200,
  "explanation": null,
  "data": {
    "id": 10,
    "userId": 12,
    "languageId": 1,
    "grade": 1
  }
}
```

Add Grade

Request Content: UserId, languageId, grade

Response Content: Grade itself

Example Request

```
POST /grades/add
```

```
{
  "grade": 4,
  "languageId": 1,
  "userId": 4
}
```

Example Response

```
  {
    "status": 200,
    "explanation": null,
    "data": {
      "userId": 4,
      "languageId": 1,
      "grade": 4
    }
  }
```

8.md

User Scenarios

1. Mahmut Kızıloğlu

- Short Bio:

Mahmut Kızıloğlu is a 25 year old waiter working at a doner shop, Hızlı Döner, Taksim, Istanbul. Mahmut never had a decent English education ever since his youth. Recently, with the increasing number of tourists coming to the doner shop, Mahmut's boss decided that all waiters must know English at some level. If Mahmut won't be able to improve his skills in English within 2 months, he will be fired. During a casual chat between waiters, a friend suggested Mahmut to learn English through Kereviz. Thus, Mahmut decided to try it out.

- Platform: Web
- User Type: Not Registered
- Language of Interest: English

Scenario

1. Mahmut opens her laptop and registers to the system using her mail address and password.
2. After successfully registering, the language selection display is shown. Mahmut chooses `English`
3. A proficiency exam is started, Mahmut tries to guess the answers and continues. He finishes the test and receives his language level.

2. Francisco Tárrega

- Short Bio:

Francisco is a guitar player from Mexico who occasionally performs in local bars in Guadalajara. He had a decent English education thanks to his private high school. But due to education system, he knows grammar very well, but he cannot speak with a person in daily life. He wants to improve his listening skills and daily conversation, also due to his nationality, he is a native Spanish speaker. While surfing on the Internet, he finds Kereviz and decided to try it.

- Platform: Android
- User Type: Not Registered
- Language of Interest: English

Scenario

1. Francisco downloads the app from Play Store,then registers to the app using his mail address and password.
2. After successfully registering, the language selection display is shown. Francisco chooses `English`
3. A proficiency exam is started, Francisco tries to guess the answers and continues. He finishes the test and receives his language level.

9.md

Evaluation of Tools and Managing the Project

Backend

On backend side we used `Spring Framework` with Java. Throughout the development we used free student version of `IntelliJ`. It has many different tools to provide us a fast development environment. We are glad for this choice. We used `Postman` and `Swagger UI` to test out API. Both of them are easy to use.

For deployment we are using `AWS EC2`. We created our CI/CD pipeline with `Travis`. Our database lives in a database instance on `AWS RDS`. We created an instance of `Postgresql` database and currently using it. All services of Amazon have a good documentation and performance.

Frontend

On front-end we use `React` to create a dazzling web application. For placement we use a bootstrap binding called `ReactStrap`. For requests we use `axios`, for routing we use `react-router`. There is an ongoing discussion about using `Redux` for state management, we are not sure about the gains we try to have will outweigh the cost of complexity just yet.

Our tooling on front-end does not fail to satisfy at the moment, we are lucky to have considerable developer experience on our team so everything is going well, design and tooling-wise.

Android

On Android part, we used Java language which is familiar to all of us from previous years. We used Android Studio as IDE which is powered by `IntelliJ`, so IDE is also familiar to us. We tested our app both on our mobile devices and Android Studio's Emulator. We used `Volley` package for handling requests with backend side. Also we used `Postman` and `Swagger UI` for testing API's availability.

9.md

Code Structure and Group Process

We have the `master` branch and our beloved `frontend` and `android` branches for holding the best code from each user-faced subteam. There are also feature branches and issue branches.

Front-End

In front-end team, we treat the `frontend` branch like it is our master branch. We have a deployment for it, so we can see our subteam's status at all times. We use feature and issue branches and create a `pull request` from another member, so every code is checked and reviewed. Our process is working well so far.

Android

We used Github's Issues and Pull Requests effectively in our opinion. Our base branch was `/android` , we developed new features in our new branches like `/android-gradeview-enhancement` , and after completed developing a new feature, we opened pull request and our code reviewed by one peer most of the time and then merged. After completion of Milestone 1, we merged our `/android` branch to master.

We usually managed our group work by talking and dividing the work between each other. We talked about which part to be implemented and divide the part. Then we would together or on our own implement the part and send each other code reviews. We are happy about the communication between our own Android team but we need some more communication with other teams and work on it.

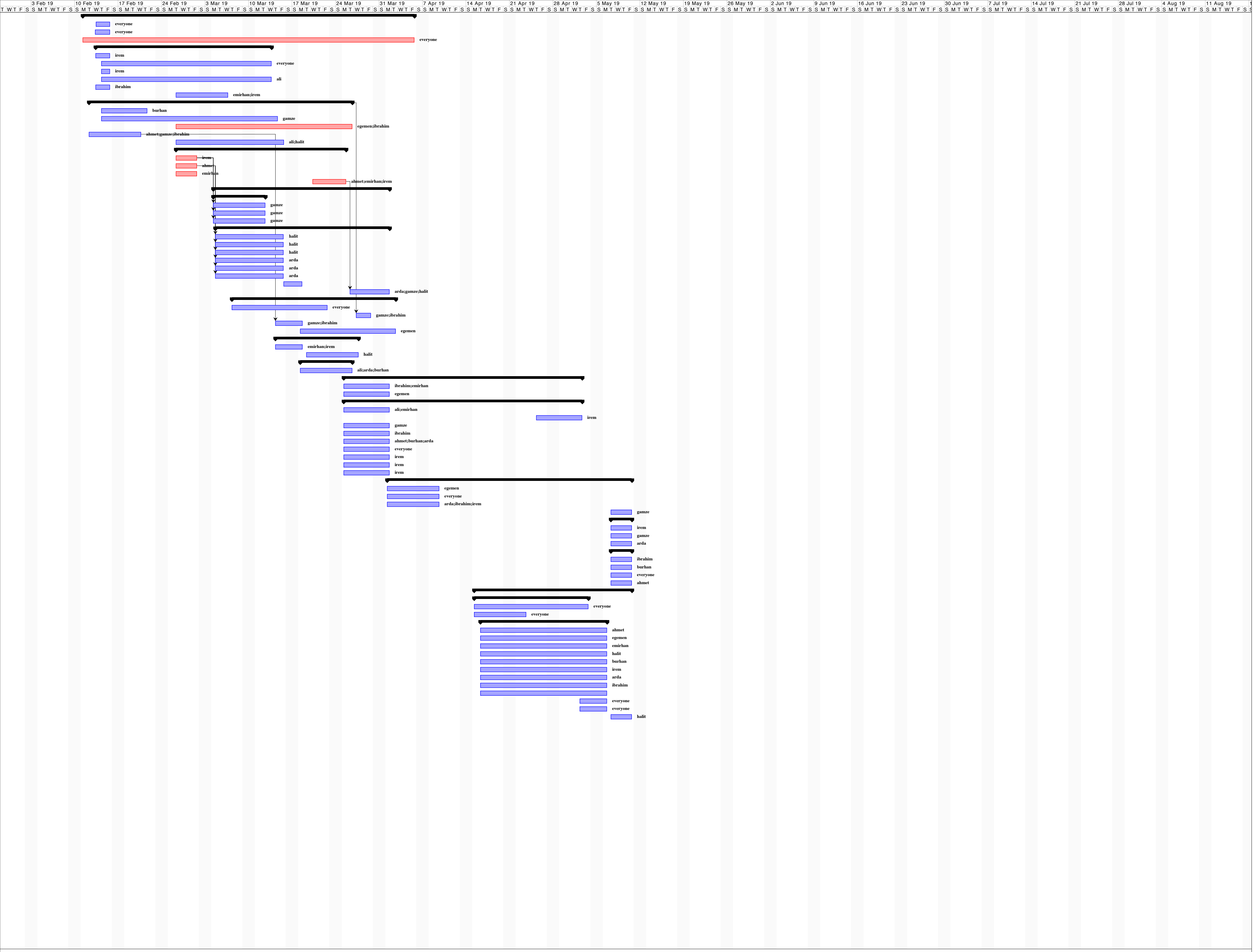
Back-End

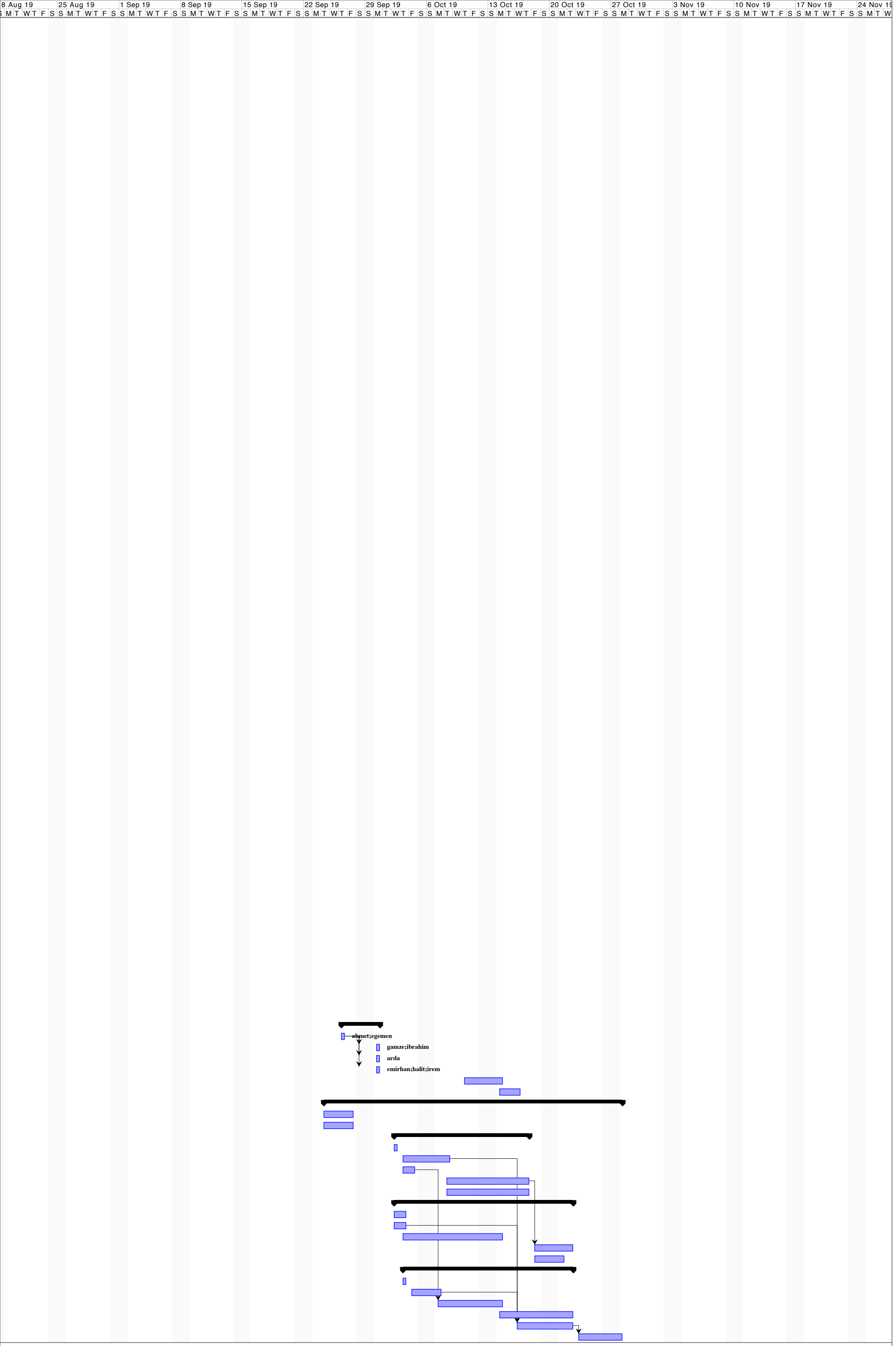
In back-end team, we don't have any main branch for development. We create a new branch for each issue, with the issue id since we want it to be unique. After the development and testing is done, we basicly merge the branch into master.

	Name	Duration	Start	Finish	Predecessors	Resource Names
1	**Preliminary Search**	**40 days?**	**2/11/19 8:00 AM**	**4/5/19 5:00 PM**		
2	Learn Markdown	2.665 days?	2/13/19 10:41 AM	2/15/19 5:00 PM		everyone
3	Learn basic git commands	3 days?	2/13/19 8:00 AM	2/15/19 5:00 PM		everyone
4	Find and document fine git repos	40 days?	2/11/19 8:00 AM	4/5/19 5:00 PM		everyone
5	**Documentation**	**20.835 days?**	**2/13/19 9:19 AM**	**3/13/19 5:00 PM**		
6	Create communication plan	2.835 days?	2/13/19 9:19 AM	2/15/19 5:00 PM		irem
7	Personal wiki pages	20 days?	2/14/19 8:00 AM	3/13/19 5:00 PM		everyone
8	Meeting agenda creation	2 days?	2/14/19 8:00 AM	2/15/19 5:00 PM		irem
9	Create and manage issues every week	20 days?	2/14/19 8:00 AM	3/13/19 5:00 PM		ali
10	Create readme.md	2.835 days?	2/13/19 9:19 AM	2/15/19 5:00 PM		ibrahim
11	Add little icons to sidebar	7 days?	2/26/19 8:00 AM	3/6/19 5:00 PM		emirhan;irem
12	**Requirements**	**31 days?**	**2/12/19 8:00 AM**	**3/26/19 5:00 PM**		
13	Edit the questions to customer to create final draft	6 days?	2/14/19 8:00 AM	2/21/19 5:00 PM		burhan
14	Edit project requirements to creater final draft	21 days?	2/14/19 8:00 AM	3/14/19 5:00 PM		gamze
15	Review and update the requirements	21 days?	2/26/19 8:00 AM	3/26/19 5:00 PM		egemen;ibrahim
16	Update Requirements according to review and Q&A	7 days	2/12/19 8:00 AM	2/20/19 5:00 PM		ahmet;gamze;ibrahim
17	Create Glossary	14 days	2/26/19 8:00 AM	3/15/19 5:00 PM		ali;halit
18	**User Personas and Scenerios**	**20 days?**	**2/26/19 8:00 AM**	**3/25/19 5:00 PM**		
19	create a user persona and a scenerio for a university student	4 days?	2/26/19 8:00 AM	3/1/19 5:00 PM		irem
20	create a user persona and a scenerio for a caretaker	4 days?	2/26/19 8:00 AM	3/1/19 5:00 PM		ahmet
21	create a user persona and a scenerio for a musician	4 days?	2/26/19 8:00 AM	3/1/19 5:00 PM		emirhan
22	Split user scenarios	4 days?	3/20/19 8:00 AM	3/25/19 5:00 PM		ahmet;emirhan;irem
23	**Mock-ups**	**21 days?**	**3/4/19 8:00 AM**	**4/1/19 5:00 PM**		
24	**Web Mockup**	**7 days?**	**3/4/19 8:00 AM**	**3/12/19 5:00 PM**		
25	Login	7 days?	3/4/19 8:00 AM	3/12/19 5:00 PM	19	gamze
26	Take Exercise	7 days?	3/4/19 8:00 AM	3/12/19 5:00 PM	19	gamze
27	Advanced Search	7 days?	3/4/19 8:00 AM	3/12/19 5:00 PM	19	gamze
28	**Android Mockup**	**20.04 days?**	**3/4/19 4:41 PM**	**4/1/19 5:00 PM**		
29	Guest User	9 days	3/4/19 4:41 PM	3/15/19 4:41 PM	20	halit
30	Register	9 days?	3/4/19 4:41 PM	3/15/19 4:41 PM	20	halit
31	Beginner Exercises	9 days?	3/4/19 4:41 PM	3/15/19 4:41 PM	20	halit
32	Login	9 days?	3/4/19 4:41 PM	3/15/19 4:41 PM	20	arda
33	Chat	9 days?	3/4/19 4:41 PM	3/15/19 4:41 PM	20	arda
34	Grading Essays	9 days?	3/4/19 4:41 PM	3/15/19 4:41 PM	20	arda
35	Revise	1 day?	3/15/19 4:41 PM	3/18/19 4:41 PM		
36	Split mock-ups	5 days?	3/26/19 8:00 AM	4/1/19 5:00 PM	22	arda;gamze;halit
37	**Use Case Diagram**	**19 days?**	**3/7/19 8:00 AM**	**4/2/19 5:00 PM**		
38	Learning to create Use Case Diagram	12 days?	3/7/19 8:00 AM	3/22/19 5:00 PM		everyone
39	Use Case Diagram	3 days?	3/27/19 8:00 AM	3/29/19 5:00 PM	12	gamze;ibrahim
40	Update the Use Case Diagram as per updated Requirements	3 days?	3/14/19 8:00 AM	3/18/19 5:00 PM	16	gamze;ibrahim
41	Check and Review the Use Case Diagram	12 days?	3/16/19 8:00 AM	4/2/19 5:00 PM		egemen
42	**Class Diagram**	**10 days?**	**3/14/19 8:00 AM**	**3/27/19 5:00 PM**		
43	Create Class Diagram	3 days?	3/14/19 8:00 AM	3/18/19 5:00 PM		emirhan;irem
44	Check and Review Class Diagram	7 days?	3/19/19 8:00 AM	3/27/19 5:00 PM		halit

	Name	Duration	Start	Finish	Predecessors	Resource Names
45	**Sequence Diagrams**	**7 days?**	**3/18/19 8:00 AM**	**3/26/19 5:00 PM**		
46	Create Sequence Diagrams	7 days?	3/18/19 8:00 AM	3/26/19 5:00 PM		ali;arda;burhan
47	**Milestone1**	**29 days?**	**3/25/19 8:00 AM**	**5/2/19 5:00 PM**		
48	Executive Summary	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		ibrahim;emirhan
49	List and Status Deliverables	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		egemen
50	**Evaluation**	**29 days?**	**3/25/19 8:00 AM**	**5/2/19 5:00 PM**		
51	Evaluate requirements and mockups	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		ali;emirhan
52	Evaluate Use-Case Diagram	6 days?	4/25/19 8:00 AM	5/2/19 5:00 PM		irem
53	Evaluate Class Diagram	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		gamze
54	Evaluate Sequence Diagrams	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		ibrahim
55	Evaluate project repository	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		ahmet;burhan;arda
56	Summary of work	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		everyone
57	Communication plan	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		irem
58	Requirements and mockups	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		irem
59	Design	6 days?	3/25/19 8:00 AM	4/1/19 5:00 PM		irem
60	**Project Plan**	**30 days?**	**4/1/19 8:00 AM**	**5/10/19 5:00 PM**		
61	Create an excel file for project plan	7 days?	4/1/19 8:00 AM	4/9/19 5:00 PM		egemen
62	Add their partitioned weeks	7 days?	4/1/19 8:00 AM	4/9/19 5:00 PM		everyone
63	Merge on Project Libre	7 days?	4/1/19 8:00 AM	4/9/19 5:00 PM		arda;ibrahim;irem
64	Add new works and update according to feedback	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		gamze
65	**Milestone 2**	**4 days?**	**5/7/19 8:00 AM**	**5/10/19 5:00 PM**		
66	Executive Summary	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		irem
67	Project Plan	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		gamze
68	List and status of deliverables	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		arda
69	**Evaluation**	**4 days?**	**5/7/19 8:00 AM**	**5/10/19 5:00 PM**		
70	Evaluate the status of deliverables	4 days	5/7/19 8:00 AM	5/10/19 5:00 PM		ibrahim
71	Evaluate the tools and processes	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		burhan
72	Summary of personal work	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		everyone
73	Merge the whole Milestone report	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		ahmet
74	**Sample API Implementation**	**20 days?**	**4/15/19 8:00 AM**	**5/10/19 5:00 PM**		
75	**Preliminary Search**	**15 days?**	**4/15/19 8:00 AM**	**5/3/19 5:00 PM**		
76	Learn Django framework	15 days?	4/15/19 8:00 AM	5/3/19 5:00 PM		everyone
77	Create Amazon Account	7 days?	4/15/19 8:00 AM	4/23/19 5:00 PM		everyone
78	**Funtionality of our Api**	**15 days?**	**4/16/19 8:00 AM**	**5/6/19 5:00 PM**		
79	word of the day	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		ahmet
80	inappropriate word detection	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		egemen
81	tag detection	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		emirhan
82	translate	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		halit
83	language detection according to location	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		burhan
84	object detection (photo to foreign text)	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		irem
85	random multimedia recommendation	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		arda
86	language detection	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		ibrahim
87	text-to-speech	15 days?	4/16/19 8:00 AM	5/6/19 5:00 PM		
88	Documentation	3 days?	5/2/19 8:00 AM	5/6/19 5:00 PM		everyone

	Name	Duration	Start	Finish	Predecessors	Resource Names
89	Review each other's codes	3 days?	5/2/19 8:00 AM	5/6/19 5:00 PM		everyone
90	Merge all functionalities	4 days?	5/7/19 8:00 AM	5/10/19 5:00 PM		halit
91	**Review Previous Work**	**3 days?**	**9/26/19 8:00 AM**	**9/30/19 5:00 PM**		
92	Review Requirements	1 day?	9/26/19 8:00 AM	9/26/19 5:00 PM		ahmet;egemen
93	Review Diagrams	1 day?	9/28/19 8:00 AM	9/30/19 5:00 PM	92	gamze;ibrahim
94	Review MockUps	1 day?	9/28/19 8:00 AM	9/30/19 5:00 PM	92	arda
95	Review Project Plan	1 day?	9/28/19 8:00 AM	9/30/19 5:00 PM	92	emirhan;halit;irem
96	Decide on a name for the project	3 days	10/10/19 8:00 AM	10/14/19 5:00 PM		
97	Create a project logo	3 days?	10/12/19 8:00 AM	10/16/19 5:00 PM		
98	**Milestone 3**	**24.001 days?**	**9/24/19 8:00 AM**	**10/28/19 8:00 AM**		
99	Research about which deployment methodologies to use	4 days	9/24/19 8:00 AM	9/27/19 5:00 PM		
100	Research technologies to use in implementation	4 days	9/24/19 8:00 AM	9/27/19 5:00 PM		
101	**Backend**	**12 days?**	**10/2/19 8:00 AM**	**10/17/19 5:00 PM**		
102	Develop a simple Spring project	1 day?	10/2/19 8:00 AM	10/2/19 5:00 PM		
103	Deploy it & Automate Deployment	4 days?	10/3/19 8:00 AM	10/8/19 5:00 PM		
104	Create Login/Register API	2 days?	10/3/19 8:00 AM	10/4/19 5:00 PM		
105	Create Language/Proficiency/Exercise API	8 days?	10/8/19 8:00 AM	10/17/19 5:00 PM		
106	Create API Docs & Swagger UI	8 days?	10/8/19 8:00 AM	10/17/19 5:00 PM		
107	**Frontend**	**15 days?**	**10/2/19 8:00 AM**	**10/22/19 5:00 PM**		
108	Develop a simple web interface	2 days?	10/2/19 8:00 AM	10/3/19 5:00 PM		
109	Deploy it & Automate Deployment	2 days?	10/2/19 8:00 AM	10/3/19 5:00 PM		
110	Create Login/Register Pages & Flow	8 days?	10/3/19 8:00 AM	10/14/19 5:00 PM		
111	Create Language Selection, Proficiency Exam Pages & API Connection	3 days?	10/18/19 8:00 AM	10/22/19 5:00 PM	105	
112	Make a second deployment & CI/CD for testing next version	2 days?	10/18/19 8:00 AM	10/21/19 5:00 PM		
113	**Android**	**14 days?**	**10/3/19 8:00 AM**	**10/22/19 5:00 PM**		
114	Develop a simple Android application	1 day?	10/3/19 8:00 AM	10/3/19 5:00 PM		
115	Take APK export and share it within team	2 days?	10/4/19 8:00 AM	10/7/19 5:00 PM		
116	Create Login/Register Screens & Flow & API Conection	6 days?	10/7/19 8:00 AM	10/14/19 5:00 PM	104	
117	Create Laguage Selection, Proficiency Exam, Grade Screens	7 days?	10/12/19 8:00 AM	10/22/19 5:00 PM		
118	Preparing Milestone 3 Demos & Presentation	5 days?	10/16/19 8:00 AM	10/22/19 5:00 PM	103;109;115	
119	Preparing Executive Summary, Deliverables and Evalutation	3.001 days?	10/23/19 8:00 AM	10/28/19 8:00 AM	118	


3 Feb 19
10 Feb 19
17 Feb 19
24 Feb 19
3 Mar 19
10 Mar 19
17 Mar 19
24 Mar 19
31 Mar 19
7 Apr 19
14 Apr 19
21 Apr 19
28 Apr 19
5 May 19
12 May 19
19 May 19
26 May 19
2 Jun 19
9 Jun 19
16 Jun 19
23 Jun 19
30 Jun 19
7 Jul 19
14 Jul 19
21 Jul 19
28 Jul 19
4 Aug 19
11 Aug 19
everyone
everyone
everyone
irem
everyone
irem
ali
ibrahim
emirhan;irem
burhan
gamze
egemen;ibrahim
ahmet;gamze;ibrahim
ali;halit
irem
ahmet
emirhan
ahmet;emirhan;irem
gamze
gamze
gamze
halit
halit
halit
arda
arda
arda
arda;gamze;halit
everyone
gamze;ibrahim
gamze;ibrahim
egemen
emirhan;irem
halit
ali;arda;burhan
ibrahim;emirhan
egemen
ali;emirhan
irem
gamze
ibrahim
ahmet;burhan;arda
everyone
irem
irem
irem
egemen
everyone
arda;ibrahim;irem
gamze
irem
gamze
arda
ibrahim
burhan
everyone
ahmet
everyone
everyone
ahmet
egemen
emirhan
halit
burhan
irem
arda
ibrahim
everyone
everyone
halit


8 Aug 19
25 Aug 19
1 Sep 19
8 Sep 19
15 Sep 19
22 Sep 19
29 Sep 19
6 Oct 19
13 Oct 19
20 Oct 19
27 Oct 19
3 Nov 19
10 Nov 19
17 Nov 19
24 Nov 19
ahmet;egemen
gamze;ibrahim
arda
emirhan;halit;irem