CmpE451

Oct 2019

Group10

Milestone 1: Group Deliverables

Contents

1 Executive Summary **3**
- 1.1 Project Information . . . 3
- 1.2 Summary of Project Status . . . 3
- 1.3 What is next? . . . 3

2 List and Status of Deliverables **4**

3 Evaluation of the Status of Deliverables and Their Impact on Plan **4**

4 Work-Done by Team Members **5**

5 Requirements **7**
- 5.1 Glossary . . . 7
- 5.2 Functional Requirements . . . 7
  - 5.2.1 User Requirements . . . 7
  - 5.2.2 System Requirements . . . 9
- 5.3 Non-Functional Requirements . . . 9

6 API Documentation **10**
- 6.1 Authentication and Registration . . . 10
- 6.2 /authenticate . . . 10
- 6.3 /register . . . 10
- 6.4 Member Operations . . . 11
- 6.5 /member/profile . . . 11
- 6.6 Quiz Operations . . . 12
- 6.7 /quiz/quizId . . . 12
- 6.8 /quiz/level . . . 13
- 6.9 /quiz/quizId/submit . . . 14

7 Project Plan **15**

8 User Scenarios **16**
- 8.1 Android Scenario . . . 16
- 8.2 Frontend Scenario . . . 16

9 Code Structure and Group Process **17**
- 9.1 Backend . . . 17
- 9.2 Frontend . . . 18
- 9.3 Android . . . 18

10 Evaluation of Tools and Processes **19**

1 Executive Summary

1.1 Project Information

Our app, "YALLP", is a platform where people can improve their language skills via learning from the materials which are carefully selected from the creators of the app. All the materials are analyzed and inspected by site admins. The solely goal of this app is transforming learning of a language into something as easy as playing candy crush. Users will be able to learn language while using the metro, waiting a friend, spending time in the bathroom, anywhere you can imagine. Any user can contact with an expert to talk or discuss about anything. Since, daily speech is a part of learning, messaging between users is also another benefit "YALLP" provides to its users. We all know the famous lines from Cem Yilmaz "Speak English? I live in English". That's exactly what we are aiming for.

1.2 Summary of Project Status

At the start of this project, we began from the bottom. What we do first is arranging teams. Afterwards, each week task by task we developed our application in every aspects. On the other hand, we discussed what we should not develop. This was also crucial since we have limited time, requirements had to be renewed. Early weeks we could not decide what to implement what to discard. Once we got developed it appeared clearer. Each team worked hard and did their parts so far. In this section, the topic will be what we have done so far.

At first, we all decided on platforms that we are going to write our code. Since most of us are new to these platforms, first week was the learning week. We analysed some examples on what we are about to develop. Afterwards our first task was to implement login and register functions on both front-end and android which are connected to back-end. We also implemented authentication and token mechanism so that users will be able to stay logged in as long as their tokens are valid.

Next, we implemented quiz feature, which is our main instrument to teach users a language. After a user enters the application, in order to determine their levels they enter quizzes. Users are able to choose any quiz that is allowed at their level. Currently, we are delivering quizzes and after a quiz, we are showing the score they get. If the exam is a placement test then we also show which level they are in.

There is a profile page also where users can see their levels in each language. Right now there is only English and the level is determined by the placement test. They also can take level up exams later. They have a bio in their profile page where they can write something about themselves.

We successfully deployed both backend and frontend. We user Amazon for that purpose. We have a fully running Postgres database which is also deployed to Amazon. We have not merged our different branches however each branch is developing steadily.

1.3 What is next?

We are aiming to put our app into Google Play too. Also our website will be publicly accessible and will have a better name.

We are planning to add different types of quizzes such as writing, listening or reading. Administrative specifications will be added later also. For now we only have a profile page, we will make it open for modification. There will be forgot password button also. Website and android designs going to be updated and become prettier.

2 List and Status of Deliverables

Name	Delivery Date	Delivery Status
Revised Requirements	Oct 14, 2019	Delivered
Backend/Frontend/Android Codes	Oct 21, 2019	Delivered
User Scenarios (Demo)	Oct 22, 2019	Delivered
Project Plan	Oct 26, 2019	Delivered
API Documentation	Oct 27, 2019	Delivered

3 Evaluation of the Status of Deliverables and Their Impact on Plan

- **User Scenarios (Demo):** As Group 10, we know how important user scenarios for our project. Because, user scenarios remind us why we are struggling to develop this project and how our project can be important for the users' lives. Therefore, we prepared two different user scenarios which were realistic for our presentation.
- **Backend/Frontend/Android Codes:** Each week, we have delivered tasks which have required mostly coding on every platform. At first, we are failed to deliver fully working codes since platforms we are using are new to most of us, we struggled in the beginning. Later, we have successfully delivered our coding tasks. Each week, we implemented a new feature which will play key role in our app such as login, register, quiz, profile. Currently we are up do date on code parts, and we are ready to go further.
- **Revised Requirements:** We have updated requirements week by week. At first, we could not make any changes since we did not have any idea about how this project will shape. Later, we learned how to evaluate our skills and abilities. Afterwards, we analyzed the requirements one more time and deleted the ones which are seemed hard to implement in this given time. In the end, we are end up with the requirements we have now. We will make our developments around these requirements from now on.
- **Project Plan:** Planning is important when we talk about group projects. So, we created a project plan that outlines the major activities and milestones of our project. We update our plan to keep track of the progression. It is a high level plan.
- **API Documentation:** We implemented backend, frontend and android parts as much as we can for the start. We have deployed both backend and frontend to Amazon. We are kind of failed to deliver deploy of frontend on time however it did not affect our development significantly. We used postman for backend api documentation which helped us a lot while communicating with frontend and android about endpoints.

4 Work-Done by Team Members

Team Member	Contributions
Yavuz Demir	I helped the creation of sign in and sign up logics in the backend. I fixed major bugs in authentication, token and authority logics. I fixed the "CORS" problem which affected frontend. I created the endpoint for placement exam. I am actively maintaining the database and deploys with my backend teammates. I often created github issues and projects.
Murat Buldu	I joined the frontend team. I created opening page of the web application. Then I implemented quiz page for frontend. I connected quiz page with the backend. I merged and combined pages done by other team members, so that the user could surf through the pages to take a quiz. I updated the project plan for milestone.
Fahri Can Şanlı	I attended all the meetings (both with the customer and with the team) actively. I am developing project YALLP in android team. I started to learn android development from the ground. I played active role of generating project structure for android platform. I implemented XML and Java code for sign-up page by myself and I helped to my teammates for creating Login pages.I created all the necessary classes of retro client and api for linking up our app with backend. I also implemented the quiz page and a pop-up screen for the case that if a user wants to cancel the quiz. I tried to share my knowledge and researches with my teammates throughout the development process. I also managed to store user token in the app when user both logs in and signs up. And I found the way to add user token as header of request sent by our app.
Hasan Basri Balaban	I got into frontend team for developing a web page for our project. I started to learn web development by using React and Redux from the ground. I helped my team to find necessary techs for frontend development. I created user interfaces for homepage, profile page and opening page by using Antd. I reviewed my teammates codes, but we usually worked on the same piece of code together. I also created sign-up, login modals. I edited Work Done By Team Members part of this report.
Sümeyra Yılmaz	We set up the project together as Android team. I created login page for Android, also created login activity, then connected it with back end. I also create a page for quiz result, also its activity. Backup connection for quiz result made in quiz activity, so I edited it. I reviewed all other Android pull requests, and merge most of them. I edited the whole Android project before milestone demos, the naming, code structure etc. in order to clear the warnings, make it more readable, and be systematic in terms of Android coding conventions. I edited requirements after customer meetings, and wrote requirement section of this report.

Table 1: Work-done by Each Team member

Team Member	Contributions
Selim Karaduman	I helped with the authentication mechanisms for registration and logging in. I created quiz evaluation endpoint. For that I implemented the corresponding models and services. I also dumped the questions to the database. I had written the API documentation. I solved some issues such as sanity checkings for email and authentication with email. Other than those, I created some issues on github related to bugs and enhancements for the project.
Melih Demir	I have been developing the Android part of the project. In particular, I have been working on the profile page part of the application. For the profile page's layout I've decided to use different types of views on different parts. For example, a list of GridView structures where you can show content both top to bottom and side by side was really useful for a dashboard of languages. I also did the API connection for said page. For this, I've created a user information model on our side to correlate with the response body they have been returning to our requests. Moreover, I have been constantly attending the meetings to make sure we are on the same page with both frontend and backend teams. I also helped on the modification of the requirements page of our wiki page.
Hatice Melike Ecevit	I set up the database and also created the new tables for each feature. I maintain the database relations between the entities and decide for which is best. I created the backend application and implemented the member model-repository-service and controller. I compiled and deployed our application using AWS Elastic Beanstalk also I connected our backend team to our database with editing configuration. I helped a little to frontend team to deploy the frontend. I implemented the quiz feature and also I dealed with the CORS problem on frontend with Yavuz. Also I created the DTO classes and DTO Converter Services for clients and I reviewed and opened pull requests with the backend team.
Burak Tepedelen	Worked with the frontend team. I created page templates. Created loggin/signup system and helped with the backend connection. Assisted with homepage design. Handled error and warning system. Wrote the code structure documentation.

Table 2: Work-done by Each Team member

5 Requirements

5.1 Glossary

- **Account:** A set of information related to application and different for each user
- **Admin User:** A person who is responsible for system sustainability and management in general
- **Annotation:** A feature that allows users to attach some explanatory content to items
- **Availability:** Application must be accessible from different platforms
- **Communication:** Interaction between users inside application by sending message
- **Guest User:** A person who does not have an account and have restricted access to the application
- **Language Dashboard:** A part of profile page which contains names of languages being learned by the user and progress bar
- **Learning Material:** A set of exercises related to a language
- **Profile:** A page specific for each user and has user related information
- **Recommendation:** A suggestion system for users to add new learning materials
- **Reliability:** System must have secure the database and be prepared for any type of incident
- **Scalability:** The ability of the application to handle a growing number of users
- **Search:** A tool to help the users find the relevant contents for given input words in the application
- **Settings:** A set of choices specific to each user
- **Sign In:** Entering to the application by providing correct email and password
- **Sign Up:** Creating an account to be a member of application
- **User:** A person that interacts with the application

5.2 Functional Requirements

5.2.1 User Requirements

- User Types
  - Guest User
    - Guest users shall be able to sign up anytime in the site.
    - Guest users shall be able to see a tutorial on each language.
    - Guest users shall be able to see options to sync with Google.
    - Guest users shall not be able to start to learn a language.
  - Admin User
    - Admin users shall be able to block or remove any registered user.
    - Admin users shall be able to view messages between any users.
    - Admin users shall be able to view any student user's progress.
  - Registered User
    - Users shall be able to view the progress of the other users that are interacted with through a writing assignment or communication mechanism(message).
    - Users shall be found on the search bar.
    - Users shall be able to learn as much new languages as they want.

        * Users shall post comments to writing materials that they receive from other users.
        * Users shall be able to annotate writing exercises.
        * Users shall be able to select a language to learn.
        * Users shall be able to learn multiple languages simultaneously.
        * Users shall be able to select another user for assigning the writing exercise to be evaluated.
        * Users shall be able to use materials which are presented for their current level and materials from previous levels.
        * Users shall be able to see their completed tasks, number of wrong-correct answers, completed percent of the course.
        * Users shall be able to search the content they are looking. This can be a user or a language specific writing on a topic.
        * Users shall be able to annotate images and texts.
        * Users shall be able to post comments to other users.
        * Users shall be able learn and evaluate others' writing exercises simultaneously.
        * User can select one of the materials he prefers to do.
        * User can select the user who will review his writing exercise.
        * User can see the notifications after he logs in.

- Authentication
    - Sign Up
        * The users shall be able to sign up by providing a unique username, an appropriate password and a valid email any time.
        * The users shall be able to sign up via Google.
    - Sign In
        * The users shall be able to sign in anytime by using username-password pair or email-password pair.
        * The users shall be able to sign in via Google if they signed up via Google.

- Profile
    - Personal Information
        * The users shall have a profile page.
        * The users shall have the option to add name and surname to his/her profile.
        * The users shall be able to load a profile picture.
    - Language Dashboard
        * The users shall be able to add new languages to learn.
        * The users shall be able to check his/her progress on each language he/she started to learn.
        * The users shall be able to drop any language course they started.
    - Settings
        * The users shall be able to change notifications preferences.
        * The users shall be able to change their passwords.
        * The users shall be able to delete their accounts permanently.

- Communication
    - The users shall be able to send messages to each other if the receiver user accepts the sender's message.
    - The users shall get notifications when they get a message from other users if they have chosen to get notifications.

- Recommendation and Contribution
    - The users shall be able to recommend new learning materials to the system.

5.2.2 System Requirements

- Search
  - System shall provide a search algorithm through keywords and show the related content.
  - System shall provide semantic search by an algorithm that selects materials by topic, type, difficulty and scope, and show the related content.
- Recommendation
  - System shall recommend users to send review.
- Annotation
  - System shall support the W3C Web Annotation Data Model.
- Account
  - System shall have a hide option to allow users to disable sharing functionality of their profile pages.
- Learning Material
  - System shall have learning materials for English, French, German and Spanish languages and if any new language is added, new materials will come along with the addition.
- Notifications
  - System shall send notifications to the user when the user gets a message, a feedback from an expert or a comment about herself/himself.

5.3 Non-Functional Requirements

- Availability
  - The system shall have a Web application that supports:
    - Chrome browser version 63 and up.
    - Firefox browser version 58 and up.
    - Opera browser version 50 and up.
  - The system shall have an Android application.
- Security
  - The system shall protect user passwords in a database using SHA-256 encryption.
  - The system shall check user queries against SQL injection attacks.
  - User account email shall be unique.
  - User account password shall be valid.
- Scalability
  - The system should handle at least 5000 online registered users at any time.
- Reliability
  - The database shall be backed up daily and weekly.
  - The system should have monthly maintenance period of 3 hours.
  - The system shall recover fully from a crash in at most 24 hours.

6 API Documentation

6.1 Authentication and Registration

6.2 /authenticate

Description

This endpoint is used for a client to sign-in. It is a POST request. The client will get a session token if he/she gives correct credentials. The details are given below.

Request

POST /authenticate
Content-Type: application/json
{
"username":"ali123",
"password":"——-"
}

Response

200
Content-Type: application/json
{
"token": "eyJhbGciOiJIUzUxMiJ9...."
}

6.3 /register

Description

This endpoint is used for a client to register. It is a POST request. The details are given below. The user needs to choose a valid username, mail and password to register. In case of success the server responses to the user with the session token.

Request

POST /register
Content-Type: application/json
{
username:"ali123",
password:"——-"
mail:"ali123@gmail.com" }

Response

200
Content-Type: application/json
{
"token": "eyJhbGciOiJIUzUxMiJ9...."
}

6.4 Member Operations

6.5 /member/profile

Description

This endpoint is used for an authenticated user to see his/her profile. It is a GET request. The client can get details of his/her profile, such as bio, mail, name, surname and languages.

Request

GET member/profile
Content-Type: application/json

Response

Content-Type: application/json


```
{
"id": 7,
"bio": null,
"password": "—",
"username": "ali",
"mail": "ali@123.com",
"role": "USER",
"name": "ali",
"surname": "kaya",
"memberLanguages": [
{
"id": 4,
"memberId": 7,
"language": {
"id": 1,
"languageName": "ENGLISH",
"hibernateLazyInitializer": {}
},
"languageLevel": 2
}
],
"expert": null,
"enabled": false
}
```

6.6 Quiz Operations

6.7 /quiz/quizId

Description

This endpoint is used for an authenticated user to get a quiz from the server. It is a GET request. The response will contain questions, the choices and the answers. The details are given below

Request

GET /quiz/quizId
Content-Type: application/json

Response

Content-Type: application/json


```
{
"id": 5,
"level": 1,
"quizType": "phrasal",
"questions": [
{
"id": 49,
"questionText": "I ..... could later today to collect the books",
"firstChoice": "check in",
"secondChoice": "broke down",
"thirdChoice": "call back",
"correctChoiceId": 3,
"quizId": 5
},
{
"id": 50,
"questionText": "Mari ..... the wedding at the very last minute!",
"firstChoice": "broke down",
"secondChoice": "called off",
"thirdChoice": "check in",
"correctChoiceId": 2,
"quizId": 5
},
{
"id": 51,
"questionText": "We have to ..... the search.",
"firstChoice": "call off",
"secondChoice": "check in",
"thirdChoice": "broke down",
"correctChoiceId": 1,
"quizId": 5
},
}
```

6.8 /quiz/level

Description

This endpoint is used for an authenticated user to get a placement test. It is a GET request. The details are given below.

Request

GET /quiz/level
Content-Type: application/json

Response

Content-Type: application/json


```
{
"id": 66,
"level": 1,
"quizType": "level",
"questions": [
{
"id": 660,
"questionText": "Choose the correct suffix for terr.....",
"firstChoice": "ify",
"secondChoice": "ize",
"thirdChoice": "ly",
"correctChoiceId": 1,
"quizId": 66
},
{
"id": 661,
"questionText": "Choose the correct suffix for ver.....",
"firstChoice": "ly",
"secondChoice": "ify",
"thirdChoice": "ize",
"correctChoiceId": 2,
"quizId": 66
},
}
```

6.9 /quiz/quizId/submit

Description

This endpoint is used for an authenticated user to submit his/her quiz. It is a POST request.
The user sends the details of the question in json format. The user includes the answers to each question
In response the server returns the quiz in json format with additional fields. Those fields are: Score of the user, which is calculated using the number correct answers of the user. Level, which is returned only if the quiz is a placement quiz. And finally, for each question the server tells whether or not the question is correct.

Request

POST quiz/quizId/submit
Content-Type: application/json


```
{
"quizId": 66,
"answers":[
{"questionId": 660, "choiceId": 2},
{"questionId": 661, "choiceId": 1},
{"questionId": 662, "choiceId": 3},
{"questionId": 663, "choiceId": 3},
{"questionId": 664, "choiceId": 3},
{"questionId": 665, "choiceId": 3},
{"questionId": 666, "choiceId": 3},
{"questionId": 667, "choiceId": 3},
{"questionId": 668, "choiceId": 3},
]
}
```


Response

Content-Type: application/json


```
{
"quizId":66,
"answers":[
{"questionId":660,"choiceId":2,"correctId":1,"true":false},
{"questionId":661,"choiceId":1,"correctId":2,"true":false},
{"questionId":662,"choiceId":3,"correctId":3,"true":true},
{"questionId":663,"choiceId":3,"correctId":2,"true":false},
{"questionId":664,"choiceId":3,"correctId":1,"true":false},
{"questionId":665,"choiceId":3,"correctId":2,"true":false},
{"questionId":666,"choiceId":3,"correctId":3,"true":true},
{"questionId":667,"choiceId":3,"correctId":2,"true":false},
{"questionId":668,"choiceId":3,"correctId":3,"true":true}
],
"level":3,
"score":3
}
```

7 Project Plan

	Name	Duration	Start	Finish	Predecessors	Resource Names
1	**Revision of The Project**	**5 days**	**9/25/19 8:00 AM**	**10/1/19 5:00 PM**		
2	Review of the Github project and creating own wiki page	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Hatice Melike Ecevit
3	Review of the Github project and creating own wiki page	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Selim Karaduman
4	Review of the Github project and creating own wiki page	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Burak Tepedelen
5	Refining questions to be asked to the customer	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Melih Demir
6	Refining questions to be asked to the customer	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Sümeyra Yilmaz
7	Revision of the class diagrams and the project plan	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Hasan Basri Balaban
8	Revision of the requirements	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Fahri Can Sanli
9	Updating the project plan	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Murat Buldu
10	Revision of the mockups	5 days	9/25/19 8:00 AM	10/1/19 5:00 PM		Yavuz Demir
11	**Backend**	**15 days**	**10/2/19 8:00 AM**	**10/22/19 5:00 PM**	**1**	**Hatice Melike Ecevit;Selim Karaduman;Yavuz Demir**
12	Authentication for backend	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Selim Karaduman
13	Necessary tables for login and sign up will be created	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Hatice Melike Ecevit
14	Login and sign up functions for backend	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM	13	Yavuz Demir
15	Authentication bugfix	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM	12	Yavuz Demir
16	Creation of necessary quiz and question classes for backend	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM		Selim Karaduman
17	Creation of quiz related tables such as questions and quizes	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	16	Hatice Melike Ecevit
18	Start of Unit Testing in backend	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM		Yavuz Demir
19	Quiz score calculations	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	16	Selim Karaduman
20	Addition of language type	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	16	Hatice Melike Ecevit
21	**Mobile**	**15 days**	**10/2/19 8:00 AM**	**10/22/19 5:00 PM**	**1**	**Fahri Can Sanli;Melih Demir;Sümeyra Yilmaz**
22	Sign up/Get Started page for Android	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Fahri Can Sanli
23	Profile page for Android	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Melih Demir
24	Home and login page for Android	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Sümeyra Yilmaz
25	Create quiz activity for Android, bug fix sign up activity	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM		Fahri Can Sanli
26	Bug fix profile page for Android	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM		Melih Demir
27	Bug fix sign in activity for Android, update requirements	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM		Sümeyra Yilmaz
28	Refactoring and developing quiz page, storing token when user sings in or signs up	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	25;26;27	Fahri Can Sanli
29	Connect profile page with backend	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	23	Melih Demir
30	Create quiz score page and connect it with backend, logout, update requirements	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	25;26;27	Sümeyra Yilmaz
31	**Frontend**	**17 days**	**10/2/19 8:00 AM**	**10/24/19 5:00 PM**	**1**	**Burak Tepedelen;Hasan Basri Balaban;Murat Buldu**
32	Login, sign up for frontend	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Burak Tepedelen
33	Opening page for frontend	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Murat Buldu
34	Home and profile page	5 days	10/2/19 8:00 AM	10/8/19 5:00 PM		Hasan Basri Balaban
35	Connecting backend to frontend	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM	32;33;34	Burak Tepedelen
36	Question page for frontend	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM		Murat Buldu
37	Revisions of the general look and profile page	5 days	10/9/19 8:00 AM	10/15/19 5:00 PM	34	Hasan Basri Balaban
38	Login/signup update in frontend	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	32	Burak Tepedelen
39	Combining frontend pages	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	32;33;34;36	Murat Buldu
40	Home page update in frontend	5 days	10/16/19 8:00 AM	10/22/19 5:00 PM	34	Hasan Basri Balaban
41	**Milestone 1**	**3 days**	**10/22/19 8:00 AM**	**10/24/19 5:00 PM**		
42	Executive summary	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Yavuz Demir
43	Project plan	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Murat Buldu
44	Work done	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Hasan Basri Balaban
45	List and status of deliverables	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Fahri Can Sanli
46	User scenarios	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Melih Demir
47	Requirements	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Sümeyra Yilmaz
48	API Documentation	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Selim Karaduman
49	Code structure of backend	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Hatice Melike Ecevit
50	Code structure of frontend	3 days	10/22/19 8:00 AM	10/24/19 5:00 PM		Burak Tepedelen

YALLP

8 User Scenarios

8.1 Android Scenario

Süreyya Yıldı is an aspiring high school graduate. She is preparing to study in a college where all lectures are given in Turkish. Due to this fact she has no way of gaining experience on English through her college. However, she wants to attend to a Work and Travel program abroad, so she really needs to find another way to teach herself the language. After doing a quick research she finds YALLP as a very suitable platform for new learners. She signs up and after being authenticated, she takes a placement test to see that she is on Beginner level.

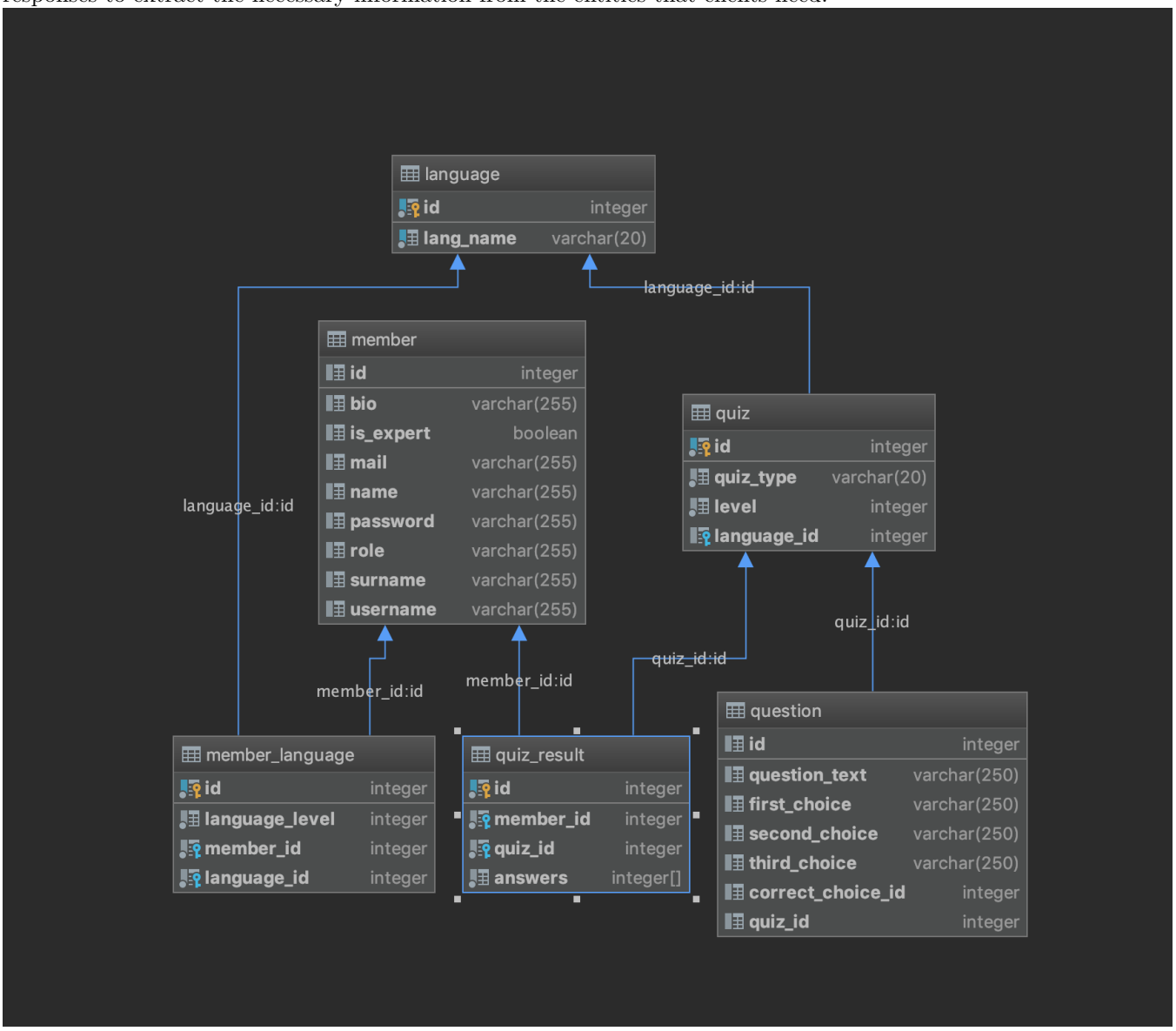
8.2 Frontend Scenario

Muhittin Topalak is a retired marine. His son works for a foreign company in England. Muhittin wants to visit his son in England this summer, so he has been studying English in order to communicate with his son's friends and colleagues better. He's been using YALLP to take daily quizzes on English. He signs in using the web client, then checks out the quiz available for the day on his dashboard.

9 Code Structure and Group Process

9.1 Backend

Backend code is composed of Java, Spring Boot framework also some other internal frameworks of Spring is used such as Spring Security and Spring Test.
It is a Maven project and project structure is as follows,
We have a config module to handle the security and other base configurations of the project also, model module to represent the entities of the project (this part will be explained in detail with the database structure), repository module to use JpaRepository to access the entities without using sql, service module to implement the logic of the platform, service module uses repository module and passes the resulting entities to controller layer, controller module takes the request and returns the response to the clients. Also we used data transfer objects (dtos) for our responses to extract the necessary information from the entities that clients need.



Our database structure as shown is very simple. We have used Postgres Relational Database on AWS and Elastic Beanstalk to deploy our backend application.

Our branch structure is as follows,
Base branch is backend-dev and for each feature we have implemented we created new branches and developed our code then we opened a pull request for each member to approve the new feature/bug/fix and merged it to backend-dev.

9.2 Frontend

Frontend code is composed of Javascript code that uses React libraries, HTML files that mainly uses CSS style and few other configuration files.
Entry to the code is through index.js file in source which initiates redux components and calls app.js in App.
Code uses Redux to handle states which ensures information transfer and changes happen in a safe and predictable manner. Necessary functions called in each page file.
Each folder in the code includes an index for calling correct javascript files inside. HTML files are with the code requires them in this folders. These folders are:

- App file that manages page routing.
- HomePage,OpeningPage,ProfilePage,QuizPage contain respective pages for website. images contain website images.
- _actions contain dispatch calls and alert returns.
- _components contain routing for logged in users.
- _constants contain constants for success and failure returns.
- _reducer contain Redux reducers for state handling.
- _servies contain backend request and other service calls.
- _helpers contain other code that doesn't fit other files.

9.3 Android

Android code is composed of mainly java classes. To make our code more readable and structured, as Android Team, we split our classes into different folders. Currently, we have activity folder where all activities are available, fragments folder in which we wrote reusable code pieces, utilities folder in which we have API folder and retro client folder. API and Retro Client folder is used to send requests to back-end and receive responses from it. We also have models folder in which we indicated the way we send our request and we expect the response to parse it easily. Other than these folders, we also have resources folder where we keep all the things related to design of our pages. In that folder, usually XML files and predefined constant string and style elements are available. Our application starts with the main activity when launched and it uses intents to navigate between activities to show different layouts to the user when needed.
As android team, we use pull requests a lot. The person developing a new feature have to open a pull request before merging his/her development branch to our dev branch and add other teammates as reviewer. The condition for one team member to be able to merge his/her branch, he/she has to get two approval from his/her team members. And if there is any comments on the changes made, the author should make change in the code accordingly and update the pull request.

Branch Name	Author	Brief Explanation
android-get-started	Fahri Can Şanlı	Get started activity and xml file are added. Reusable design models are added to style file and reusable variables are added to dimens file. Sign-up user model is added, user retro client and user api are created.
android-profile-page	Melih Demir	Profile page activity added. Basic profile page layout created.
android-login	Sümeyra Yılmaz	Sign-in user model created, user api updated. Sign-in activity and layout is added.
android-signupbackendconnection	Fahri Can Şanlı	Sign-up user model has been updated. Email validation and empty field validation are added.
android-loginBackendConnection	Sümeyra Yılmaz	Sign-in is connected with backend.
android-signup-fix	Fahri Can Şanlı	Sign-up model had a uncompatible field for back-end request and it is fixed.
android-quiz	Fahri Can Şanlı	Taking token from back-end and storing it in shared preferences are carried out. Quiz activity and corresponding xml files are created. Progress bar is added. Question fragment is created. Question and quiz models are created. Quiz api, quiz retro client is created. AreYouSure fragment is added. Destroying all activities and fragments when navigating is added.
android-profile-connections	Melih Demir	Profile page updated. Profile page connected with backend endpoints. User info model created to fill profile page.
android-quiz-score	Sümeyra Yılmaz	Connection for quiz result is added. Quiz api edited. Models for Answer, AnswerResult, QuizAnswers, and QuizResult added. QuizActivity edited, QuizScoreActivity and its layout is created. Log out is added. The project code is cleaned in general. String values added to string resources file. Warnings handled.
android-bugfix	Melih Demir	Updated QuizApi. Minor interface changes. Show level as a text on quiz completion added. Update level on profile page as a quiz result added. Removed debug logs. Fixed profile info updating bug.

Table 3: Android Branches

10 Evaluation of Tools and Processes

Tools And Processes	Evaluation
Github	Github is our main and essential tool while developing our application. We put our weekly meeting notes there and create issues frequently to check progress of our tasks. For this purpose, we mainly use projects part of the github. Also, since this is a group project, it is impossible to code it using only one computer. Therefore, we use github to store our actual code and create branches from it to develop multiple features at the same time.
Pull Request	As we stated above, different team member creates different branches to develop different features of our application. For that reason, we benefits from pull request to give other people a chance to review each other's code before merge them with master.
Code Review	Code review is almost one of the most important process when developing our project. Since, different team members takes responsibility for different parts of our application, people may make mistakes when using some functionality provided by their teammates. Hence, we decided to use code review for each feature we develop to prevent mistakes.
Postman	Postman is a tool that we used for developing backend part of our project.As we all know, our backend includes different endpoints and different request types such as get and post. Postman provides us the environment to test our code by sending different request with different contents to our endpoints and see the responses.
Whatsapp and Slack	Since we work as a team, we must be in touch at all times to let everyone be aware of improvements we have achieved and the difficulties we have faced. Sometimes, we also use this communication channels to make some important decisions that shows up after our meetings.
React & Redux	React is "A JavaScript library for building user interfaces". It is component based and so, we can effectively share the work among the team members. Redux is a predictable state container for JavaScript apps. So we can easily create a global state and exploit it in every component. Using react and redux together provides us a good and simple code structure and a good environment to add new features in a easy way.
Visual Studio Code	To develop frontend, we needed a development environment. We haven't decided a common IDE but Visual Studio Code is one of them. Visual Studio Code is a good environment to develop React applications since it has useful extensions for JavaScript, JSX and React itself.
Spring Boot	Spring Boot is an open source Java-based framework used to create a micro Service. We have used Spring Boot since most of our backend team members are familiar with it. It has minimum amount of configuration compared to other frameworks and it is easily deployed and easy to use with Maven as a package manager. All and all it is very efficient to develop using Spring Boot with its inner features.
IntelliJ	IntelliJ IDEA is a special programming environment or integrated development environment (IDE) largely meant for Java. IntelliJ makes it so easy to test, run and compile our Java code and also it works well with maven.
Android Studio	Android Studio provides the fastest tools to build an Android project. As Android team, we worked on it as an IDE. We created both activites, resources and our backend connection models in Android Studio. It provides us a simulator to see the app inside the phone, test its functionalities. We use this simulator at demo session too. Also, it can be connected to a phone, hence one can easily build and create a project in her/his Android phone.
Retrofit	Retrofit is a type-safe HTTP client for Android and Java. Retrofit models REST endpoints as Java interfaces, making them very simple to understand and consume. It provides a convenient builder for constructing our required objects.

Table 4: Evaluation of Tools and Processes