CmpE 451 - Milestone I Report

Bonibon - Language Learning Platform

Group 5

October 25, 2019

Contents

1 Executive Summary

1.1 Project Introduction

Bonibon is language learning platform open to anyone. You can sign up for free, learn languages you don't know, do exercise anytime you want and help other people with languages you are good at. There are four types of exercises: listening, reading, writing, vocabulary. Bonibon also gives you an opportunity for chatting with other learners and experts or sending them essays to improve writing skills. You can follow up your progress and level. Bonibon accepts contributions from the community. If you want to contribute our system, you can suggest new exercises so that other people can make use of your language skills. Contribution is not only limited by the suggestions, you can also have a conversation with someone whose level is lower than you or evaluate his/her essays and give feedback.

1.2 Project Status

API: We have designed and implemented the interfaces pertaining to the goals of the first milestone.
Requirements: We defined functional and non-functional requirements. Functional requirements
has user and system requirements. Non-functional requirements has availability and accessibility,
performance and response time, annotations, reliability, security, privacy and GDPR.
Mockup - User Stories - Personas: We defined two different personas who will use the system. We created scenarios for each one of them.
Backend: Is deployed and capable of performing tasks pertaining to the goals of the first milestone. Uses continuous integration.
Android App: Is capable of performing tasks pertaining to the goals of the first milestone. Some features that have not been implemented yet exist only as non-functional placeholders. Is written in Java, using the native Android framework.
Frontend: Is capable of performing tasks pertaining to the goals of the first milestone. Some features that have not been implemented yet exist only as placeholders. Is written using React.

1.3 Future Plans

We need to design and implement the remaining part of the API to satisfy our requirements. We also need to develop the Android app and the frontend to take advantage of them.

2 List and Status of Deliverables

Name	Delivery Date	Status
Android: Login Page	October 22, 2019	Delivered
Android: Registration Page	October 22, 2019	Delivered
Android: Language Selection Page	October 22, 2019	Delivered
Android: Proficiency Exam Page	October 22, 2019	Delivered
Android: Proficiency Exam Result Page	October 22, 2019	Delivered
Android: Main Menu Page	October 22, 2019	Delivered
Android: Profile Page	October 22, 2019	Delivered
Web: Home Page	October 22, 2019	Delivered
Web: Login Page	October 22, 2019	Delivered
Web: Registration Page	October 22, 2019	Delivered
Web: Language Selection Page	October 22, 2019	Delivered
Web: Proficiency Test Page	October 22, 2019	Delivered
Web: Test Result Page	October 22, 2019	Delivered

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

This is our first milestone in Cmpe 451 course. Since, our project is the continuation of Cmpe 352 course, we started to examine from the last status of our project. Then, we have evaluated our status on each deliverable one by one. All of our application pages have both Web and Android versions and their integration to the backend server is done efficiently. Both Frontend and Backend are deployed on different AWS servers.

3.1 Requirements

As I mention at the beginning of the evaluation, we started to examine our requirements by eliminating unnecessary parts and adding some required parts into the requirements. After reviewing is done, we constructed the second version of our project's requirements and started to put our effort on the project according to this requirements.

3.2 Project Plan

Working as a team requires partition of overall work on each group member. So, from the beginning of this project, we first divided into three subgroups as Frontend, Backend and Android. In order to perform our job successfully, we have learned required programming languages and frameworks in advance. And then, on each week's assignment, each subgroup did their jobs as it planned in the project plan.

3.3 Home Page

Both Frontend and Android teams designed Home page while Backend team was constructing the backend server of our project. We tried to create an easy-to-navigate homepage in order to help users.

3.4 Login & SignUp Page

We created our registration pages as described in the requirements. After giving their credentials, users can be able to get into the application and start learning. At the backend, the required APIs and tests for registration were implemented.

3.5 Language Selection Page

After registration is completed, we direct our user into a Language Selection page in order to ask which language he/she wants to learn. Our application supports three languages that are Turkish, English and German.

3.6 Proficiency Test Page

After Language Selection is completed, we canalize our user into a Proficiency Test in order to measure his/her language level. The questions are fetched from the Proficiency Test API.

3.7 Test Result Page

There are six levels starting from A1 to C2 in the application. After Proficiency Test is completed, we lead our user into a Test Result page where he/she can be able to see his/her level.

4 A Summary of Work Done by Each Team Member

Team Member	Contribution
Hasan Öztürk	In the Android team, we have divided our work in the basis of activities. I have created a bridge activity on which a user can land after a successful login and sign up. In this activity I have created buttons for each language (English, German, and Turkish) and I created data carrier intents to the proficiency activity. This intents carry the chosen language to the proficiency activity and that activity presents the corresponding exam accordingly. I have also created a Profile Page activity even if it is beyond the scope of milestone 1. For profile page I have tried to present a good UI where user can view his/her name, surname, username, average rating, attended languages and the received comments. Since the backend of the profile utility is not ready, I did not connect it to the server, but I parsed the json input as if it’s ready.

Furkan Kadıoğlu	- Login, register endpoints. - Test cases for login, register endpoints - User authentication - Api Specification for other teams - Proficiency exam endpoint and test cases (after mert has done remarkable changes and test cases are not valid) - Backend code design and directory structure - Api documentation - Remarkable contributions to other backend members about django and drf
Kartal Kaan Bozdoğan	- Contributed to the milestone presentation - Created some of the English proficiency exam questions - Created the Android app proficiency exam page - Solved merge conflicts - Reviewed merge requests - Created the Android app skeleton
Ali Meriç Deşer	I have contributed the project as a member of Android team.I have created the registration page for signing up the mobile application of the project. Worked on taking string inputs and convert them to JSON and using Volley, have handled the REST API requests.Designed it and handled the backend integration of this page. I have worked for token sharing issues. Solved some merge conflicts and reviewed pull requests. Working on validations and button visualization. Deployed the mobile skeleton
Meltem Suiçmez	- Contributed as a frontend team member, using react.js - Created language selection and guest-login pages - Visual design and style of the login, sign-up, home, about, guest and

	language selection pages. using paint for logos and backgrounds, and react-bootstrap and material-ui libraries. - Added a question to english and turkish prof exams. -Reviewed the frontend branch pull request and merged it to master. -Did some manual tests of some scenarios with Selamettin.
Nevzat Ersoy	- Contributed as a backend team member, using django on python. - Created models for database such as user model, language model and comment model, along with their serializers. - Created the ‘profile’ endpoint of the API and prepared its view in django. - Wrote tests for the profile view. - Reviewed the material which has been done by other members of backend.
Selamettin Dirik	I am in the Frontend team. At first, I implemented SignUp page for registration and Test Result page for Proficiency Test takers. And partially, I took participate in Proficiency Test page by reorganizing it according to new API-Specifications. In addition, I added CSS into these pages. I also partially rearranged the Home page by throwing out unnecessary parts. I fixed two bugs. One of them was not correctly showing the Test Result of the users and the other one was not properly redirecting the logged out users to the Home page. During creation of all these pages, I also implemented their Action-Creators and Reducers.
Mahmut Uzunpostalcı	I have contributed to the Android project as I am in Android team. I have created proficiency exam result page that showcases the correct, incorrect and

	language level with respect to the exam results. I have also designed login page and connected it to the backend, using Volley and JSONObject. I have created Main menu page that a user will land after login or after proficiency exam result page. I also have connected these pages to the according pages. I have also solved some merge conflicts of my teammates’ pull requests.
Abdullah Enes ÖNCÜ	I am from the Frontend Team. First of all, I builded the architecture of our Web App by constructing Router, Redux, and Redux-Thunk. After that, I wrote login page and prof test page without design and I connected them to Api. On the other hand, I searched about state management of project and added something for cookies to our App. I don’t have a finger on graphics and visual design of our project, but I mostly interested with architectural structure of our project.
Yusuf Mert Bila	I am in the backend team, here are my contributions to the backend: I dockerized the backend I integrated the backend with Travis CI to run tests at each pull request to backend branch I deployed the backend on an AWS EC2 instance I redesigned proficiency endpoint along with exam, proficiency exam, question and question option models. I also wrote serializers for these models and implemented all these changes I actively took part in code reviews in pull requests I helped frontend and android teams for api specifications

5. Requirements

5.1. Functional Requirements

- 1.1. User Requirements
    - 1.1.1. Registration and Login
        - 1.1.1.1. Users shall be able to either create an account or continue as a guest user.
        - 1.1.1.2 Registered User
            - 1.1.1.2.1. Users shall be able to sign up by providing a username with at most 10 characters, an email, at least six character long password and native language.
            - 1.1.1.2.2. Users shall be able to login to the application by providing his/her username or email and password.
        - 1.1.1.3 Guest User
            - 1.1.1.3.1. Guest users shall have limited access to content.(Only first exercises of exercise types)
        - 1.1.1.4 Admin User
            - 1.1.1.4.1. Admin users shall be able to delete abusive content and users.
            - 1.1.1.4.2. Admin users shall be able to alter the content of exercises.
    - 1.1.2. Course Material
        - 1.1.2.1. Users shall be able to apply to any language he/she wants to learn.
        - 1.1.2.2. Users shall be able to take the proficiency exam.
        - 1.1.2.3. Users shall be able to take listening, reading, grammar, vocabulary exercises.
        - 1.1.2.4. Users shall be able to access the lower level contents.
        - 1.1.2.5. Users shall be able to suggest new material for a language.
        - 1.1.2.6. Admins shall review and approve suggested new materials to the system.
        - 1.1.2.7. Users shall be able to see the *correct* answers after the *exercise*.
    - 1.1.3 Writing Assignment
        - 1.1.3.1. Users shall be able to upload a text file or image file that includes text as a writing exercise

        - 1.1.3.2. Users shall be able to select one user and see recommended users who are provided by the application.
        - 1.1.3.3. Users shall be able to communicate with the users that are currently reviewing his/her writing exercise through a built-in communication mechanism
        - 1.1.3.4. Users shall be able to reject an essay request.
    - 1.1.4 Annotation in writing exercise
        - 1.1.4.1 Users shall be able to annotate essays that are in either text or image form, as a feedback for the writing exercises
        - 1.1.4.2 Users shall be able to annotate a text or image in a course content.
        - 1.1.4.3 Users shall be able to annotate a text or image in a course content
    - 1.1.5 Chat
        - 1.1.5.1 Users shall be able to chat with other users.
    - 1.1.6 Tracking for progress
        - 1.1.6.1 Users shall be able to track progress and what she/he accomplished so far.
        - 1.1.6.2 Users shall be able to see statistics about the ratio of completed exercises.
        - 1.1.6.3 Users shall be able to view average grade.
    - 1.1.7 Rating & Comment
        - 1.1.7.1 Users shall be able to view user's rates and comments about her/him.
        - 1.1.7.2 Users shall be able to vote users to whom they've sent an essay.
        - 1.1.7.3 Users shall be able to comment about users to whom they have sent an essay.
    - 1.1.8 Search
        - 1.1.8.1. Users shall be able to search a user and a content semantically through keywords
        - 1.1.8.2. Users shall be able to do advanced search a user and a content through tags.
        - 1.1.8.3. Users shall be able to suggest tags and keywords about material which they suggest
        - 1.1.8.4. Admin users shall be able to approve or reject the suggested tags for the materials
- 1.2. System requirements
    - 1.2.1 Recommendation

    - 1.2.1.1 The system shall provide a recommendation mechanism for users in order to suggest a set of users that are eligible to review a writing exercise
- 1.2.2 Search
    - 1.2.1.1 The system shall provide a mechanism for searching a content semantically.
    - 1.2.1.2 The system shall provide a mechanism for searching a content with exact matching.
- 1.2.3 Education Language
    - 1.2.3.1 System shall supply materials for learners in Turkish, English, and German.
- 1.2.4 Commenting and Rating
    - 1.2.4.1 System should keep comments and votes mentioned in 1.1.7 and evaluate overall ratings for the relevant user profiles.
- 1.2.5 Annotation
    - 1.2.5.1 The system shall provide an annotation mechanism for users to annotate a text, an image file or a part of an image file
- 1.2.6 Communication System
    - 1.2.6.1 System should provide an interface for communication between *users.*
- 1.2.7 Adding New Learning Materials
    - 1.2.7.1 System should redirect the materials suggested from users mentioned in 1.1.2.5 to admins in order to get approval about materials.
- 1.2.8 Apparency & Privacy
    - 1.2.8.1 System should maintain the privacy by restricting the accessibility to user profiles from guest users.
    - 1.2.8.2 System should restrict guest users from rating & commenting for the user profiles.
- 1.2.9 Progress & Statistics
    - 1.2.9.1 System should provide an interface to let user track and monitor his/her progress.
    - 1.2.9.2 System should also provide statistics of ratio of completed exercises, average grade.
- 1.2.10 Automatic Grading
    - 1.2.10.1 System shall automatically grade listening, reading, grammar and vocabulary exercises and shall be capable of

highlighting the correct answer if the provided one by the user is wrong.

2. Nonfunctional Requirements

- 2.1. Availability and Accessibility
  - 2.1.1. The system should work on any web browsers except IE10 and older versions. Chrome
  - 2.1.2. The application shall be provided for Android 7 and later versions.
  - 2.1.3. The system should inform the user if the application is not able to provide a service.
  - 2.1.4. If a failure happens, the problem should be fixed within 1 hour.
- 2.2. Performance and Response time
  - 2.2.1. The system shall block an excessive amount of access requests made in a short amount of time.
  - 2.2.2. The system shall back-up user-expert messages once a month.
  - 2.2.3. The system shall use encryption for user messages and passwords.
- 2.3. Annotations
  - 2.3.1. W3C Web Annotation Data Model shall be used for writing exercises which are annotated by learners and experts.
  - 2.3.2. Implementation of this system will follow the standards introduced by the World Wide Web Consortium (W3C).
- 2.4. Reliability
  - 2.4.1. The system shall endure up to 250 users without breaking.
  - 2.4.2. The system shall provide materials to searches within 3 seconds.
- 2.5. Security
  - 2.5.1. The system should encrypt the traffic between browser and web server by using HTTPS.
  - 2.5.2. The system should use input validation.
  - 2.5.3. The authorization should always be done on the server side.
  - 2.5.4. The system should not use redirect to prevent header injection.
  - 2.5.5. The system should sanitize the user input.
- 2.6. Privacy and GDPR
  - 2.6.1. The system should have a privacy policy and this policy should be seen easily by users.
  - 2.6.2. The user should have right to object processing of self data for direct marketing.

- 2.6.3. The user should have right to object processing that is based on profiling.
- 2.6.4. The system should not collect or try to predict users personal data like religious, gender, identity, race, political leanings.

6 API Documentation

http://18.197.149.174:8000/docs
http://18.197.149.174:8000/redocs

8 User Scenarios

8.1 Web Scenario

Persona: Ecem Varsın

Ecem studies literature at Bogazici University. She is 21 years old and her native language is Turkish. She knows English well and also she knows German a little, from her high school years. She wants to go to Germany with the Erasmus program next year, so she wants to improve her German and do some practise online. She hears about bonibon from her friend and wants to give it a try.

She goes to Bonibon website for the first time and wants to sign-up immediately. She signs up by typing her name, surname, email address and password. She chooses Turkish as her native language.

When she signs up she is redirected to the language selection page for the proficiency exam. The proficiency exam will put her into a learning level according to her knowledge. She choses German because she wants to learn German.

Then she is redirected to the proficiency exam to place her level. She takes the test, she goes to the next question by clicking next button and she can also go back to previous question from the prev button.

She finishes the test and clicks the complete test button. She sees her right and wrong answered questions, wrong answers appears red and correct answers appears green. Then she clicks newly appeared show me my results button.

She is redirected to the test results page and there she sees her test statistics and her German language level.

8.2 Android Scenario

Persona: Halil Yılmaz

Halil is a high school student living in Adıyaman, Turkey. He thinks that he cannot learn English at school, thus he looks for an alternative ways to learn. Halil spends most of this time in transportation, since his home is very far away from the school and he plans to utilize this time.

One day, he sees Bonibon mobile app from a friend of him and thinks that “Let’s give it a try”. He downloads the app in the school and opens it in his return to the home in the bus.

First he signs up the app and app redirects him to a page where he can choose a language for which he will take the proficiency test. Because he wants to learn English, he chooses English, not Turkish or German. Then he sees the proficiency test which is used to calculate his level and give exercises accordingly. He solves the test and get a score of A2 meaning that he is a beginner.

Afterwards he goes to his profile page where he can see his name, surname, username, his attended languages (English for now), his average rating (will be implemented later), and his comments (will be implemented later).

Then he logs out from the app. One day later he opens up the app again and logs in this time and spend some time in the app and logs out again. His next plan is to take exercises and start to learn English...

9. Code Structure

9.1 Branch Structure

The project has three phases as known:

- Backend
- Frontend
- Mobile (Android)

We, as a team have started to develop all of these phases from scratch. So, it Is very likely to have code congestions while working with a 10 people team. Therefore, we came up with an idea that we should create 3 branches the same Name with phases (backend,frontend,mobile) we thought that this isolation may let Us work more clearly and we can combine our work when we finish coding. This main branches can also have sub-branches for development purposes, we name them as <branch_name>_issue<#number>_<feature_summary> and we Keep our updated work in the main branches by merging when the changes are done.

9.2 Pull Request Structure

As mentioned above, we have branches and sub branches, Because of the fact that we work separately, we may have conflicts. We talk as a team and assign reviewers for pull requests and if there are conflicts, we try to resolve them with communicating each other.So far, we haven’t had faced with a disaster and handled the conflict pretty good.

And we had our presentation for Milestone 1 without having error.We will continue developing like this and at final stages we will merge 3 branches for production.

10. Evaluation of Tools

The project has many phases and thus, we needed more tools for each coding, testing deployment and documentary issues.For example the android phase is developed with Java programming language using Android Studio and also emulators. The project has evolved with the help of the tools. By holding to given requirements for tool selection, we also try to use most modern technologies for developing and deployment purposes. Some of these are changeable for person to person(IDE choice for example). But the main technologies are common. Backend is developed by using Python Django, Front-end is developed by using React. We have also deployed our project. We have considered the ease of use, scalability, collaborative(While working together) while selecting tools. So far, we have managed to work together in harmony without having problems with the help of the tools we have used.

11. Managing the Project

The project is being developed using the modern software development approaches. Working in a crowded team makes development a little challenging, that's why we have implemented some methods to make it easier.

Groups: We have 3 groups for 3 phases of the project. These are Backend, Frontend and Android teams. Keeping groups smaller makes development easier. Also communication is easier. For sure, we keep the communication constant as a whole communication group.

Meetings: We have weekly meetings inside groups and also as a whole project team. We share the current status of our phases, discuss what to do next and how to integrate together. We try to be clean as much as possible.

Other Communication: Besides our meetings, we also communicate from other channels like slack and whatsapp groups.

Git & Branches and PRs : We use Git as version control management system. As mentioned above, we have branches and sub branches just to keep our project simple and easy to control. We assign reviewers and merge requests to merge our work together without having conflicts.

All of these above are the techniques that we use to manage our project. As our instructor said, communication is the key. That's why we try to manage as a team at first.