

Milestone-1 Report

Cultural Heritage

Group-11

CONTENTS

I	Executive Summary: Summary of project status and any changes that are planned for moving forward.	1
II	List and status of deliverables	1
III	Evaluation of the status of deliverables and its impact on plan	1
IV	A summary of coding work done by each team member (In tabular format)	2
V	Requirements	2
VI	Project plan (In tabular format; you may omit Gantt time lines and display dates/durations only as text for readability)	3
VII	code structure and group process	4
VIII	Evaluation of tools and managing the project	4
IX	design	5

I. EXECUTIVE SUMMARY: SUMMARY OF PROJECT STATUS AND ANY CHANGES THAT ARE PLANNED FOR MOVING FORWARD.

Frontend: Basic content upload (title, description, image) can be done. Uploaded items can be displayed in newsfeed page and item detail page. Register/Login features are implemented. Since every member is new to react js, the coding was somewhat problematic, and will be reviewed with proper react js knowledge. Routing problems showed up after deployment will be solved. Profile page and edit feature will be done. Registration through facebook and google+ will be implemented.

Android: First, learned how to develop in android from Udemmy lessons and application that Udemmy provides such as Miwok and quakeReport. Then integrated what we learned, to our project. Signup and login pages are designed and the functional features are working well. On NewsFeed page, basic operations such as getting heritage items from database and showing these items on NewsFeed page are implemented. Upload Page is implemented and the functional features are working well. Profile Page is implemented and getting profile information from database is working well. Yet, there are some bugs on saving changes on profile while trying to change some of the information. Registration through facebook and google+ will be implemented on signup and login pages.

Backend: We setup Django and installed required projects. Then we deployed backend project to Amazon AWS. We designed database models all together. Ezgi created corresponding tables and their features in Django models and we updated them when it is required. After all these are done, we started to implement endpoints. For first milestone, we implemented register, login, get all heritage items, get heritage item by id, and upload heritage item endpoints. Also, we implemented get profile informations and update profile informations even though they are not in our first milestone. We only need to write endpoints for second and final milestones for future because initial setups and database models are done in the first milestone.

II. LIST AND STATUS OF DELIVERABLES

Frontend:

User registration: user can register and log in

Heritage item upload: logged in users can upload heritage items with title, description, image (tags not working properly)

Timeline view: only the time of upload is displayed

Profile page: profile page is not implemented yet

Android:

User registration: user can register and log in

Heritage item upload: logged in users can upload heritage items with title, description, image, tags, date, location

NewsFeed: logged in users can see newsFeed page item list. Yet, comment, like and search features are not implemented

Profile page: get function is working well, yet there are some tiny bugs with saving profile.

Backend:

User registration: User model is created and they can registered easily.

Heritage item upload: Heritage Item model is created. Corresponding endpoints for getting and uploading heritage items are implemented.

Profile page: Profile model is created and integrated with Django's default user model. Endpoints for viewing and updating profile page are implemented.

Timeline View: Location and time of heritage items can be seen in items but Timeline models are not created, yet.

III. EVALUATION OF THE STATUS OF DELIVERABLES AND ITS IMPACT ON PLAN

Frontend:

Unregistered Users : requires mandatory fields to be able to register the system, non-mandatory fields and registration via Facebook and Google accounts have not been implemented yet.

Registered Users: There is no profile page yet. What it can be done is that users can add text and image to their account and see what they post on news feed.

Search Requirements: Search function has not been implemented yet. Homepage Requirements: both users (registered or not) can see homepage and when they click the item, it can be seen in more detail. Search Bar, Hot Topics and Recommend Items have not been implemented yet. For now, it does not been asked to user to give a start-end time and location and several names for the item. Related with system requirements, there is no annotation, discussion forum, and recommendations from the users.

Android:

User Registration: Mandatory fields to be able to register the system are implemented well, non-mandatory fields and registration via Facebook and Google accounts have not been implemented yet.

Heritage item upload: Functional features such as upload an item with image, title, description, location and date are working fine.

NewsFeed: Requirements for first milestone are implemented such as listing items on NewsFeed page with the properties are working well. Requirements for second milestone such as annotation, comment, like and search are not implemented yet.

ProfilePage: Profile page is implemented with minor bugs which is not necessary until second milestone.

Backend:

User Registration: It is very vital for our project and we successfully implemented all required fields for user registration and login. We are completely done on this deliverable that we don't have to worry about this part in the future.

Heritage item: We created corresponding models and endpoints for heritage items. It is core of our project. We designed heritage items with one image and without timeline model but we have creation time and location for the item. We will edit item model and endpoints with one or more videos, audios, images, and also with timeline model. It is very important for our project and we completed successfully all requirements for first milestone for backend part.

Timeline View: We didn't implement timeline view but it was not vital for our first milestone because the importance of timeline depends on the change of heritage items which was not main focus on our first milestone.

ProfilePage: We implemented Profile model and corresponding endpoints for profile page. It doesn't have much importance because profile page will be needed when we implement comments and following users sections.

IV. A SUMMARY OF CODING WORK DONE BY EACH TEAM MEMBER (IN TABULAR FORMAT)

Group Member	Team	A summary of coding work done by each team member
Rıza Özçelik	Android	He has setup the android project and pushed it to the repo. Designed news feed, signup, login and upload pages. He was also responsible for connecting these pages to the API provided by the backend. He has led the task distribution job in the team and organized the team.
Barın Özmen	Android	Studied about how to use android Studio and setup android project from the repo. Designed profile page. Connected profile page with the API to be able to get data of registered user or to change profile information.
Giray Eryılmaz	Android	Learned android from scratch, designed skeleton of heritage item view (later some design changes are made), connected newsfeed and some of heritage item view to network.
M. Enes Çakır	Backend	Setup Django project and installed required projects. Made initial setups. Deployed backend project to Amazon AWS with Ezgi. Built JWT authentication system for security. Wrote Item serializer and created end points for Item CRUD. Created API documentation at Swagger.
N. Ezgi Yüçetürk	Backend	Learned a bit of Django from scratch, process is on progress. Created the chart of database table documentation. Deployed backend project to Amazon AWS with Enes. Implement models in the Django.
Abdullatif Köksal	Backend	Learned Django Designed Database Model with Ezgi. Created new model called Profile, and edited all models Default Django user model and our profile model is integrated. Created 2 endpoints which are getProfile and updateProfile. Fixed bugs of endpoints.
Kaan Uzdoğan	Frontend	Integrated the existing login page desing into React.js. Made login calls to backend and passed the login token to homepage. Store token upon login and remove on logout. Added roting to main.js. Link creation for each item in the feed. Merged Milestone 1 frontend to master and deployed to Amazon Server.
F. Hilal Benzer	Frontend	Creating news feed page and navbar using React.js. Fetching data as list from api and delivering items to smaller components. Receiving tokens as cookies and updating navbar (for anonymus users and logged in users) using them.
Halil Kalkan	Frontend	Designing login page and adding css without converting it to React.js object. After many attempts, making a proper connection with backend via fetch api. Designing and implementing item page in React.js format. Populating item page with id coming from new feed.
Melike Ermiş	Frontend	Upload page is created and implemented with React. Basically the aim of the page is that it takes input from the registered user as an image, title, description and tags and sends to database in JSON format. Two buttons which are "Login with Facebook" and "Login with Google" are implemented in the login page. Their functionalities have not implemented yet.

V. REQUIREMENTS

Implemented requirements

1.1.1.1 Unregistered users shall be able to register to the system.

1.1.1.1.1 Registration form contains mandatory fields which are username, password and e-mail address.

1.1.1.2 Unregistered users shall be able to search and view the heritage items.

1.1.2.1 Registered users shall login to the system via their registered email address/username and password

1.1.2.2 Registered users shall have a profile page which includes username, name, surname, date of birth, user photo, followed heritage items, email address, information about himself/herself, and privacy options.

1.1.2.3 Registered users shall be able to edit their profile page.

1.1.2.5.1 Registered users shall be able to add title of the heritage item.

1.1.2.5.1 Registered users shall be able to add text.

1.1.2.5.2 Registered users shall be able to add image.

1.1.3.1 Admins/Moderators shall be able to manage user permissions.

1.1.3.1.1 Admins/Moderators shall be able to give admin/moderator permissions to registered users.(Registered user shall be able to become admin/moderator with authorization of an admin/moderator.)

1.1.5.1 User should be able to see relevant heritage items in home page and if he/she is registered he/she should be able to add new items in homepage.

- 2.4.1 The user's passwords shall be at least 8 characters and contain at least one number and one letter.
- 2.4.2 The passwords shall be encrypted using MD5 protocol.

VI. PROJECT PLAN (IN TABULAR FORMAT; YOU MAY OMIT GANTT TIME LINES AND DISPLAY DATES/DURATIONS ONLY AS TEXT FOR READABILITY)

Task Explanation	Start Date	End Date
Reviewing project documents	18/09/2017	18/09/2017
Reviewing requirements	24/09/2017	24/09/2017
Reviewing use case	24/09/2017	24/09/2017
Reviewing scenarios	24/09/2017	24/09/2017
Reviewing class diagram	24/09/2017	24/09/2017
Reviewing sequence diagrams	24/09/2017	24/09/2017
Reviewing test cases	24/09/2017	24/09/2017
Reviewing project plan	24/09/2017	24/09/2017
Setting up the project environment	22/09/2017	30/09/2017
Build Eclipse with Git integration	22/09/2017	23/09/2017
Build a database server	23/09/2017	24/09/2017
Download and build dependent libraries	25/09/2017	26/09/2017
Fixing possible bugs and initial commit	26/09/2017	27/09/2017
Learning the basics of Web/Android	27/09/2017	30/09/2017
Learning basic HTML/CSS	27/09/2017	30/09/2017
Learning web framework and related web language	27/09/2017	30/09/2017
Learning basics of JavaScript (if needed)	27/09/2017	30/09/2017
Learning Android SDK	27/09/2017	30/09/2017
First commit!	27/09/2017	30/09/2017
Database design	30/09/2017	07/10/2017
Designing the DB tables on blueprint	30/09/2017	07/10/2017
Creating DB schema and necessary	30/09/2017	07/10/2017
Sign up and Login System	07/10/2017	14/10/2017
Login page design	07/10/2017	14/10/2017
Front-end, android and back-end integration	07/10/2017	14/10/2017
Integration on serverside, external login API integration	07/10/2017	14/10/2017
Basic tests on login system	07/10/2017	14/10/2017
Bug fixes	07/10/2017	24/10/2017
Item upload	14/10/2017	24/10/2017
Front-end implementation	14/10/2017	24/10/2017
Back-end implementation	14/10/2017	24/10/2017
Image upload integration	14/10/2017	24/10/2017
Android activity implementation	14/10/2017	24/10/2017
Basic tests	14/10/2017	24/10/2017
Bug fixes	14/10/2017	24/10/2017
Milestone 1	24/10/2017	24/10/2017
Commenting, Rating System, Profile Activities	30/10/2017	15/11/2017
Front-end implementation	30/10/2017	13/11/2017
Back-end implementation	30/10/2017	13/11/2017
Android activity implementation	30/10/2017	13/11/2017
Profile page implementation	30/10/2017	13/11/2017
Basic tests	13/11/2017	15/11/2017
Bug fixes	14/11/2017	15/11/2017
Milestone 2	15/11/2017	15/11/2017
Searching System with Semantic Tags	15/11/2017	23/11/2017
Research on the semantic search implementation	15/11/2017	17/11/2017
Front-end implementation	20/11/2017	20/11/2017
Back-end implementation	20/11/2017	22/11/2017
Android activity implementation	20/11/2017	21/11/2017
Basic tests	23/11/2017	23/11/2017
Bug fixes	23/11/2017	23/11/2017
Recommendation System and Feed	24/11/2017	05/12/2017
Research on machine learning and recommendation algorithms	24/11/2017	24/11/2017
Front-end implementation	27/11/2017	27/11/2017
Back-end implementation	28/11/2017	30/11/2017
Android activity implementation	01/12/2017	01/12/2017
Basic tests	04/12/2017	04/12/2017
Bug fixes	05/12/2017	05/12/2017
Subscription	03/12/2017	11/12/2017
Front-end implementation	03/12/2017	03/12/2017
Back-end implementation	04/12/2017	05/12/2017
Android activity implementation	06/12/2017	07/12/2017
Basic tests	08/12/2017	08/12/2017
Bug fixes	11/12/2017	11/12/2017

Last Enhancements	09/12/2017	13/12/2017
JavaScript enhancements, UI improvements	09/12/2017	11/12/2017
QR Code integration (If time remains)	12/12/2017	13/12/2017
Testing	01/12/2017	31/01/2018
General Testing	01/12/2017	22/12/2017
Writing Unit Tests (JUnit or equivalent)	25/12/2017	09/01/2018
Preparing a guide for Android application	10/01/2018	16/01/2018
Preparing a guide for web application	17/01/2018	23/01/2018
Update mockups with real screenshots	24/01/2018	31/01/2018
Presentation	15/12/2017	16/01/2018
Completion of documentation	15/12/2017	29/12/2017
Customer presentation	01/01/2018	05/01/2018
Technical presentation	08/01/2018	12/01/2018
Review of presentations	15/01/2018	16/01/2018
Milestone 3 - Project Delivered	17/12/2018	17/12/2018

VII. CODE STRUCTURE AND GROUP PROCESS

Frontend: We have created a frontend branch to work separately from other development branches. For the sake of not conflicting with our works, we had different folders for different pages. Actually, reactjs helps us to work with components which ease our work. Reactjs renders in a single file so that we didn't need to update that file. In our team, everyone has pushed just modified files so that we didn't have trouble with overwriting or deleting any file. At the end, when milestone 1 is reached, we have merged our branch with master.

Android: We have had android branch and Rıza created the Culturage project under that branch and all android team pushed their work in that branch. Most of the changes are occurred in these files:

android/Culturage/app/src/main/res: for design of android.

android/Culturage/app/src/main/java/com/culturage/oceans_eleven/culturage/: for implementing functions in Java language.

Backend: We have had backend branch and all backend team pushed their work in that branch and Enes deployed our changes to the server. We have lots of changes but main changes in the code are done to these files.

backend/api/views.py : Endpoints

backend/base/models.py : Database models and features

backend/base/serializers.py : Serializers

backend/api/urls.py : Corresponding urls for endpoints

VIII. EVALUATION OF TOOLS AND MANAGING THE PROJECT

Frontend:

Postman: used for api requests, checking if the data can be fetched

React js: used for implementing a dynamic web page with required components and rendering the content

Bootstrap: used for basic styling

npm: installing packages and loaders

We understand that versions of the packages are also important point. Some pages were using bootstrap v4.0 components, however some of them were using version 3. For that reason, we have decided to convert them and use version 3 as global. We learned that we need to decide which version of the tool or framework we will be using in our works.

Android:

Android Studio: Used for developing android.

Picasso: Used for uploading images from android to database.

Backend:

Django & django-rest-framework : Our backend framework.

Django Shell & Admin Page: To check our models and create instances of these models manually.

Postman : To check api requests and implementations of endpoints on local.

Swagger: To document our endpoints.

Django-jwt: To authenticate our users because it gives us more than one token(We have both web and mobile applications)

Virtual Env: To use same versions of libraries and applications.

IX. DESIGN

Android Design:

The image displays three Android app design mockups, each with a status bar at the top showing 34% battery and a time of 16:04 or 15:47.

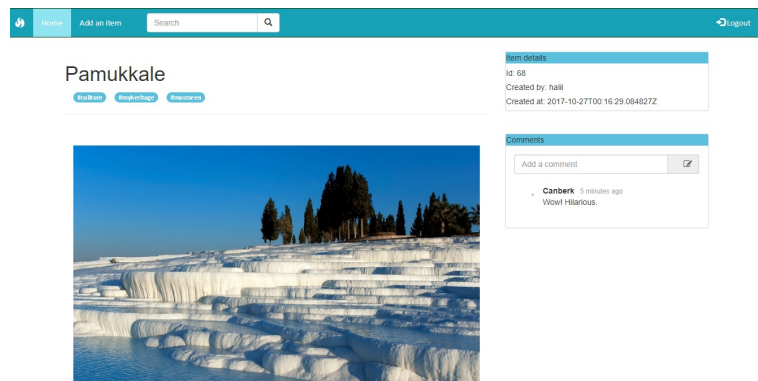
Top Left Mockup (Post Creation): Features a header with a checkmark and close icon. Below is a large image placeholder with a camera icon and a link icon. The form includes fields for Title, Description, Time, Location, and Tags.

Top Middle Mockup (Post Detail): Features a header with a search bar and user icon. The main content shows a post titled "Pamukkale" with a description: "Pamukkale is a natural heritage that everyone should see and experience that atmosphere." Below the text are like and comment icons, both showing 0. At the bottom is a video player with a plus icon for more content.

Top Right Mockup (User Profile): Features a header with a back arrow, checkmark, and close icon. It shows a profile picture placeholder and a camera icon. The form includes fields for Full Name, Username (pre-filled with "tesths"), Email (pre-filled with "test@trst.com"), Location, Birthday, and Password.

Bottom Mockup (Login/Signup): Features a header with a checkmark and close icon. It shows a profile picture placeholder and a camera icon. The form includes fields for Title, Description, Time, Location, and Tags. To the right is a login section with fields for Username and Password, a LOGIN button, and an OR separator. Below the separator is a "Need an account?" section with a SIGN UP button.

Web Design:





Start sharing
knowledge about
cultural heritage

username

password

Login

 Login with Facebook

 Login with Google

Not registered? [Create an account](#)

The screenshot shows the 'Upload New Item' form in the web application. The navigation bar is the same as in the previous screenshot. The form has a title 'Upload New Item'. On the left, there is a file upload section with a 'Choose File' button and a 'No file chosen' status. Below this is a large empty box for the item image. On the right, there are input fields for 'Title', 'Description', and 'Tags'. At the bottom right, there are 'Submit' and 'Clear' buttons.

