Milestone-2 Report
Cultural Heritage

Group-11

Contents

I. Executive Summary: Summary of project status and any changes that are planned for moving forward.

Frontend: Users can now upload multiple media after initially posting the item. Also Youtube videos can be embedded. Images shown in a slider. Videos and images of the item are seperated and the view can be changed by clicking respective buttons. Profile page is created and user information can be updated. Tag search and standard search is implemented. Recommendations integrated into project: Related items are shown in the item page and recommended items for the user are shown in the news feed page. Users are redirected to news feed if they are already logged in. Users can view the site without logging in but can't interact. Users can like or dislike an item. Users can comment on items. As a future work we will be implementing Facebook login, fixing the item details and comments section design in item page and adding the logo.

Android: We started with some refactoring on working but "dirty" parts of our code then we continued implementing new features. search button was there but was idle,we added its functionality, we added fragments for general news feed and user specific recommendation. added item based recommendation on item view added comments to items,a user can comment and delete his former comments, user now can like an item can view images in full screen multiple images are now supported for each item (both uploading and viewing) youtube media support is added too As for future work, we have very basic location and date support but they will be replaced by more sophisticated ones with multi-time-lines. Also we will support social media login and enable user to make customizations according to his preferences).

Backend: We started our part with fixing some issues that cause bugs in the endpoints enhanced query searching time. We were using the swagger app to generate documentation but we discovered that postman was actually performing the documentation while testing the API endpoints so we created documentation and examples for android and front-end teams to be use them conveniently. We implemented new endpoints one by one as asked in each week by customer. First we implemented comment endpoints then lile and tag enpoints subsequently. We update profile page to support editing. We changed item endpoint to support multiple images as well as media type sources. Timeline and location support endpoints were added although they are not currently used by android and front-end. We already implemented dummy search and dummy recommendation in the first milestone. However, after the tag and like endpoints were implemented, search and recommendation algorithms were updated as well to run based on the tags and user likes. At some point, because our AWS account on Amazon was expired we had to open another account change our back end server. We open another AWS account, re-arrenge AWS and create new project in it. Enes deployed all past and current changes to this new instance.

II. List and status of deliverables

Frontend:

Search: Tags can be search by clicking in the item page. Smart search can be done through the search bar and lists related items as well.
Media and Youtube support: Multiple images and videos may be added. Videos shown in Youtube and images shown in a slider.
Like: Items can be liked or disliked
Comment: Items can be commented
Recommendation: Suggested items are displayed in item page. If user is logged in recommended items show on the left side of the news feed.

Android:

Search: User can search for items
Recommendation: Both user and item based
Multiple media support: Items now have multiple images and you tube links
Like: User can like - and unlike (”take back” his like)
Comment: User can comments and remove his comments
Tags: User can make filtering by tag

Backend:

Comment: All items support comments and comment functionality is fully operational.
Tags: All items can be tagged and tag endpoint supports list of all tags given by an item ID as well as multiple tag addition and deletion.
Like: Users can like or withdraw their like but cannot dislike an item. Like endpoint supports list of all items given by a user ID.
Multiple Media: Back-end supports addition of multiple images as well as audio and video addition to the system. It also supports connection with Youtube.
Profile: Back-end supports profile images and edition of user informations in the system.
Timeline: Back-end supports various kinds of day-time formats including before/after common era and also text based location.
Search: Search functionality performs based on item’s name, description as well as tags in the system.
Recommendation: Recommendation functionality runs on the basis of similarity calculations of user’s and tags of the items that they like.
Test: Back-end started implementing some of the system tests.

III. Evaluation of the status of deliverables and its impact on plan

Frontend:

Search: Search is an important feature of the website enabling the user to use the website easily and efficiently. Username, location and date based search may be implemented later.
Media and Youtube support: Images through url may be implemented in future. Audio upload may also be implemented.
Like: Like/dislike feature is fully implemented and not to be worried about in the next milestone.
Comment: Comment is also fully implemented, changes in design may be done in the next milestone.
Recommendation: Both user based recommendation and item based recommendation are complete and we won’t have to worry about them in the next milestone.

Android:

Search: using the search bar on top of the news feed screen user can search for heritage items.
Recommendation: on the second fragment next to the general recently added items there is a list of recommended items, the list is user based.
Multiple media: of course one heritage item can have lots of images, now we support adding and viewing them, also we support youtube videos too.
Like: Functional features such as upload an item with image, title, description, location and date are working fine.
Tags: Functional features such as upload an item with image, title, description, location and date are working fine.

Backend:

Comment: Ability to comment on an item is a main feature of a crowd-source application. We may consider to implement another comment functionality which enables users to comment on a comment
Tags: Tags are main source for recommendation and search algorithms. It is also the foundation of describing an item as an heritage item. We are going to expand our tag space.
Like: Ability to like/dislike and to report an item is another main feature of a crowd-source application. We are going to implement dislike and report functionalities.
Multiple Media: It would be unrealistic to assume that a heritage item is shown only by a picture.

Profile: Back-end supports profile images and edition of user informations in the system.

Timeline: It was important to fix date-time problem since we are planning to add timeline review of a heritage item which may contain multiple date-time-location imformation of an item. Without this fixation further features cannot be implemented.

Search: Search functionality is a main functionality of a web-page so of course it is a part of our project. Besides, recommendation partly uses search.

Recommendation: Recommendation functionality what makes an applicaiton fancy and smart. It enhances user experience a lot. Therefore, we are going to improve it and make it more sensitive and accurate.

Test: System test is convenient to use and real time saver for all teams. It is helpful especially adding new endpoints because we need to make sure that eberything works correctly. In addition, running just test cases saves time while deploying the project. Otherwise, checking each endpoint one by one would cost so much time and probablly ruin the database.

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 added multiple image uploading and displaying. Also enabled displaying of video content from YouTube. Created a custom dialog for comment uploading and displaying. Connected the display with API back-end. Also designed a horizontal recycler list for item based recommendations Solved the date upload and display issue and led task distribution in Android.
Barın Özmen	Android	Implemented comment and delete comment function. Implemented like function and interface showing likes. On heritage item page, implemented 2 loaders, one is for guest profile picture, the other will be slider for recommended heritage items.
Giray Eryılmaz	Android	Enabled heritage item search, added a neater tags view and filtering on tags, changed the structure of news-feed so that now there are 2 fragments one is for general recently added items and one for user based item recommendation.
M. Enes Çakır	Backend	Made some performance optimizations and fixed N+1 query problem. Migrated API documentation to Postman from Swagger. Wrote like system for items. Added multiple images and Youtube link support to item. Added date serializer for different resolutions. Fixed some issues from Milestone 1
N. Ezgi Yücetürk	Backend	Comment serializer and view was added User-liked-item endpoint was added. Tag post method was added. Userlikes method was added to be used in recommendation. Fully functional item model test-cases added. Some part of test-cases of API-Item endpoint was added.
Abdullatıf Köksal	Backend	Word2Vec model is found. This Word2Vec model helps to find similarity between two words. Also, this is a lite version of word2vec. It only contains 50000 most frequent words because the server must respond fast. Smart search is implemented according to word2vec similarities of tags and query. Smart recommendation system for users is implemented according to word2vec similarities of tags and tags of items which user liked. Smart recommendation system for items is implemented according to word2vec similarities of tags of the item and tags of all other items. Bugs are fixed, generally
Kaan Uzdoğan	Frontend	Redirect to newsfeed if user is logged in. Added random login background. Solved front-end branch merging problem. Deployment to Amazon server. Debugging
Hilal Benzer	Frontend	Added recommended component which is only visible when the user is signed up. Personalized navbar. Adapted some html code to react js, added new react components. Improved CSS in homepage, navbar, search pages
Halil Kalkan	Frontend	Added like and comment feature in item page. Added feature to showing item tags and implementing tag search in frontend. Added suggested items field in item page. Enabled users to upload Youtube video and images to an existing item and in the item page added a feature to show these media properly.
Melike Ermiş	Frontend	Editable profile page is created. Search functionality is implemented into search bar & profile picture is added to navbar. Date information functionality is added to upload page. Tag problem are fixed in upload page, so users can add tags as they wish.

V. Requirements

Android:

1.1.2.7 Registered users shall be able to search and view the heritage items
1.1.5.3 Hot topics: User should be able to see popular heritage items in this area.
1.1.5.4 Hot topics: User should be able to see newly added heritage items in this area.
1.1.5.5 Recommended items: User should be able to see heritage items' recommendations. It shows the similar items based on search made by the user.

Back-end:

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/user name 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
1.1.2.3 Registered users shall be able to edit their profile page.
1.1.2.5.*Registered users shall be able to add a new heritage item.
1.1.2.6 Registered users shall be able to rate and verify the heritage items.
1.1.2.11 Registered users shall be able to add several tags to heritage items.
1.1.6.* Timeline Requirements
1.1.2.7 Registered users shall be able to search and view the heritage items.
1.1.2.12 Registered users shall be able to see the last contributor and information of the heritage item.
1.2.2.* Recommendation
Full Text Search

Front-end:

1.1.1.2 Unregistered users shall be able to search and view the heritage items.
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.
1.1.2.5.4 Registered users shall be able to add video.
1.1.2.6 Registered users shall be able to rate and verify the heritage items.
1.1.2.7 Registered users shall be able to search and view the heritage items.
1.1.2.11 Registered users shall be able to add several tags to heritage items.
1.1.4.1 Users shall be able to search heritage items with their names, annotations, dates, and tags.
1.1.4.2 Users shall be able to search heritage items with their context and description.
1.1.5.2 Search Bar: User should be able to search heritage items' by using this toolbar.
1.1.5.5 Recommended items: User should be able to see heritage items' recommendations. It shows the similar items based on search made by the user.
1.2.2.3 Recommendations of other heritage items shall be done based on user's rating on heritage items.

VI. Project plan - In tabular format

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 server side, 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've followed the same project structure as the first milestone. We have a separate frontend folder. Under the frontend folder we have the source folder that contains the "components" of the app. React.js let's us build our app in smaller pieces called components. We each have pages assigned that we work on which consist of several component files. This has let us work without conflicts between our codes.

When a feature is done we have built our deployable app with the command "npm run build" and upload it the the Amazon S3 server to have an online running version.

Android: Our style did not change we still work on android branch and use PR's for merging with master. Only we made some more classification in the project for easier implementation, we have five categories in android/Culturage/app/src/main/java/com/culturage/oceans_eleven/culturage/ : namely adapters, baseClasses, network, newsfeed, sign_up login names are self explanatory. Though the classification is not too strict.

Backend: Our code can be inspected on the github. We are still using back-end branch and main contributions are added there. On the other hand we separated the directories belonging to recommendations and search as well as filter. We used filter to reduce number of queries performed by item search, tag search and other endpoints. All endpoints were added into the same file but those mentioned above. Our way of working is the same: an endpoint is assigned someone, then bugs are fixed and other teams starts using it. There was only one pull request which contains all the changes in the back-end.
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
backend/searches/views.py: for search
backend/recommendation/views.py: for recommendation
backend/filter/views.py: for query optimization

VIII. Evaluation of tools and managing the project

Frontend:

git: Used for versioning
Postman: Used for api requests
React js: Used for implementing a dynamic web page with required components and rendering the content
React Bootstrap: Used for basic styling
npm: installing packages and loaders

We have often made use of npm packages for new features such as react-datepicker for user birthdate and react-responsive-carousel for image sliders. Each time a new package is installed every member has to run npm install to have the package locally. This sometimes made us think something wrong is going on with the code. One commit caused conflicts with the backend code on merging. This is because that particular commit deleted some backend files. We've solved the issue by reverting the commit.

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:it is used to test the endpoints locally as well as generate an awesome documentation.
Pycharm : it is one on the IDE we use to develop our code. It is helpful to discover to unknown build-in methods but it is a bit problematic in the pulling and fetching bussiness. It shows lots of conflict.
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.