Culturage

- Final Milestone Report -

Group 11

Rıza Özçelik
Barın Özmen
Giray Eryılmaz
M.Enes Çakır
N.Ezgi Yücetürk
Abdullatif Köksal
Kaan Uzdoğan
Hilal Benzer
Halil Kalkan
Melike Ermiş

Bogazici University
Jan 1, 2018

Contents

Chapter 1

Final Project Assessment

Our SWE project was to develop a *Cultural Heritage Crowdsourcing Plaftorm*, enabling a user to view and contribute to various cultural heritage items through a website or Android application. For our project, we had two main stages: Planning and Implementation.

During Planning period we had difficulties that would have impact in Implementation part:

- The project was abstract, we could not think of every possibility and condition.
- We had no idea regarding the complexities of requirements we set or intervals needed to implement them.
- We had never planned a project before.

In implementation part we realized that it is troublesome to work with a fixed project plan. In order to meet with deadlines, certain differences were necessary. Teams were formed and tasks were divided between all members. Aside from our weekly meetings, other means of communication were conducted to complete our deliverables for each milestone. These communication methods included Customer meetings, Team-only meetings, GitHub Issues.. etc. **Communication turned out to be essential for proper outcome**.

Also for desirable outputs in different conditions/inputs are requested. For this tests with various input data is important. **Tests** must be applied and undesirable results and bugs must be fixed.

Chapter 2

Coding Work Done by Each Team Member

Group Member	Team	A summary of coding work
Rıza Özçelik	Android	Added choosing location from map and displaying it on the map again for the heritage items. Added notification support as well. When an item the user has added is liked, s/he is notified.
Barın Özmen	Android	Registered user can flag(report) an item. Write unit test cases with JUnit 4 framework for android
Giray Eryılmaz	Android	Added different date resolutions for heritage items, also changed showing the date and picking the date views accordingly. Made documentation of android java code (javadoc).
M. Enes Çakır	Backend	Added backend support for 3 different date resolutions. Added point image annotation support. Added Onesignal push notification when someone liked item.
N. Ezgi Yücetürk	Backend	Write unit test cases for backend Write doxygen documentation
Abdullatıf Köksal	Backend	Implemented Google Chrome Plugin which shows item detail based on the title of the page. Plugin also adds youtube links to the existing items.
Kaan Uzdoğan	Frontend	Frontend deployment Added icon to the website, minor css fixes Implemented Annotations
Hilal Benzer	Frontend	Added location choice to the upload page using Google api. Edited item details panel so that location and date is displayed.

		Implemented map and marker in item page using coordinates of item.
Halil Kalkan	Frontend	Mobile support in multi image slider. Fixing a bug with video and multi image in item page.
Melike Ermiş	Frontend	Different date resolutions are added to upload page. Date format is changed and location information is added to JSON. Flag button is implemented for reporting issues. Profile page is changed , so user can edit the desired field only.

Chapter 3

Requirements

- 1.Functional Requirements
  - ○ 1.1 User Requirements
    - – 1.1.1 Unregistered Users
      - ○ 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`.
        - – 1.1.1.1.2 Registration form contains non-mandatory fields which are `real name`, `birthday` and `location`.
        - – 1.1.1.1.3 Registration shall be done via facebook accounts authentication as well.
      - ○ 1.1.1.2 Unregistered users shall be able to search and view the heritage items.
      - ○ 1.1.1.3 Unregistered users shall be able to share the heritage items and give reference to them.
    - – 1.1.2 Registered Users
      - ○ 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.
      - ○ 1.1.2.4 Registered users shall be able to reset their password and email.
      - ○ 1.1.2.5 Registered users shall be able to add a new heritage item.
        - – 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.
        - – 1.1.2.5.3 Registered users shall be able to add audio.
        - – 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.
    - ○ 1.1.2.8 Registered users shall be able to share the heritage items and give reference to them.
    - ○ 1.1.2.9 Registered users shall be able to add annotations to the heritage items.
    - ○ 1.1.2.10 Registered users shall be able to enable or disable discussion forum option.
    - ○ 1.1.2.11 Registered users shall be able to add several tags to heritage items.
    - ○ 1.1.2.12 Registered users shall be able to see the last contributor and information of the heritage item.
    - ○ 1.1.2.13 Registered users shall be able to follow other registered users.
- – 1.1.3 Admins and Moderators
    - ○ 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.)
    - ○ 1.1.3.2 Admins/Moderators shall be able to review and manage heritage items and annotations if needed.
        - – 1.1.3.2.1 Admins/Moderators shall be able to rate the items' pages.
        - – 1.1.3.2.2 Admins/Moderators shall be able to delete the heritage items and annotations.
- – 1.1.4 Search Requirements
    - ○ 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.
    - ○ 1.1.4.3 Users shall be able to search heritage items with their creators' usernames.
    - ○ 1.1.4.4 Users shall be able to search heritage items with their real historical date and location.
- – 1.1.5 Homepage Requirements
    - ○ 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.
    - ○ 1.1.5.2 Search Bar: User should be able to search heritage items' by using this toolbar.
    - ○ 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.
- 1.1.6 Timeline Requirements
  - 1.1.6.1 Start-end time should be in the timeline.
  - 1.1.6.2 Location of the heritage item should be in the timeline.
  - 1.1.6.3 Several names for heritage item should be in the timeline.
- 1.2 System Requirements
  - 1.2.1 Heritage Items Requirements
    - 1.2.1.1 Mandatory Requirements
      - 1.2.1.1.1 Title which is the name of the item shall be mandatory requirement.
      - 1.2.1.1.2 Annotation, which can be text or tags, shall be mandatory requirement.
      - 1.2.1.1.3 Discussion forum which can be used for both comments and question/answers shall be mandatory requirement.
    - 1.2.1.2 Non-mandatory Requirements
      - 1.2.1.2.1 Description which gives brief information about the item shall not be mandatory.
      - 1.2.1.2.2 Meta data which has at least 1, at most 10 semantic tags shall be non-mandatory.
      - 1.2.1.2.3 Geographical references which state location information about the item shall be non-mandatory.
      - 1.2.1.2.4 Rating/Ranking information collected from the application analytics shall not be mandatory.
  - 1.2.2 Recommendation
    - 1.2.2.1 Recommendations of other heritage items shall be done based on user's location.
    - 1.2.2.2 Recommendations of other heritage items shall be done based on the heritage items user has added before.
    - 1.2.2.3 Recommendations of other heritage items shall be done based on user's rating on heritage items. *2.Non-Functional Requirements**

- 2.1 Reliability
  - 2.1.1 System shall backup the database every day to another computer in less congested hours.
- 2.2 Accessibility
  - 2.2.1 System shall recover itself in at most 2 hours in case of a crash.
  - 2.2.2 Pages shall be able to be loaded in less than 1 second. Even where there is system update, the loading of the page shall take less than 2 seconds.
- 2.3 Privacy

  - 2.3.1 User's location information shall not be shared with third parties.

- 2.4 Security
  - 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.
  - 2.4.3 The system shall be able to detect robots trying to reach the website and prompts a captcha in that case.

Chapter 4

All Design Documents

4.1 Sequence Diagrams

4.1.1 Sequence Diagram 1

Name: Sign Up
Actor: Guests
Preconditions:

- User must have an email address.

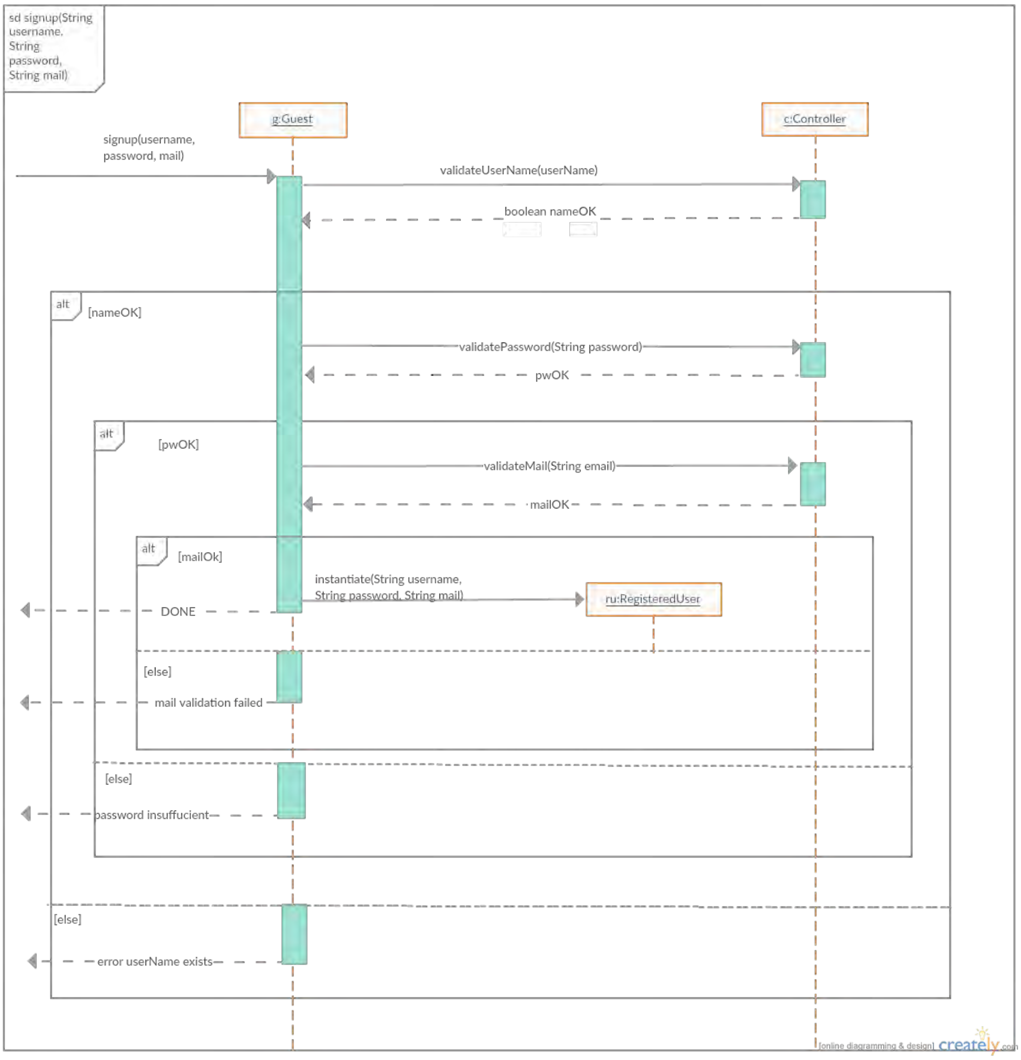
Action Steps:

1. Unregistered user clicks on "Sign Up Now" button, if he/she's on mobile.
2. User enters a valid username, an email address and a password and proceeds.
3. User confirms his/her account via confirmation email sent by the system.

Postconditions:

- User is now a Registered User.

Figure 4.1: Sequence Diagram 1: Sign Up


sd signup(String username, String password, String mail)
g:Guest
c:Controller
signup(username, password, mail)
validateUserName(userName)
boolean nameOK
alt
[nameOK]
validatePassword(String password)
pwOK
alt
[pwOK]
validateMail(String email)
mailOK
alt
[mailOk]
instantiate(String username, String password, String mail)
ru:RegisteredUser
DONE
[else]
mail validation failed
[else]
password insuffucient
[else]
error userName exists
[online diagramming & design] creately.com

4.1.2 Sequence Diagram 3

Name: Search
Actor: Guest, Registered User, Moderator
Preconditions: None
Action Steps:

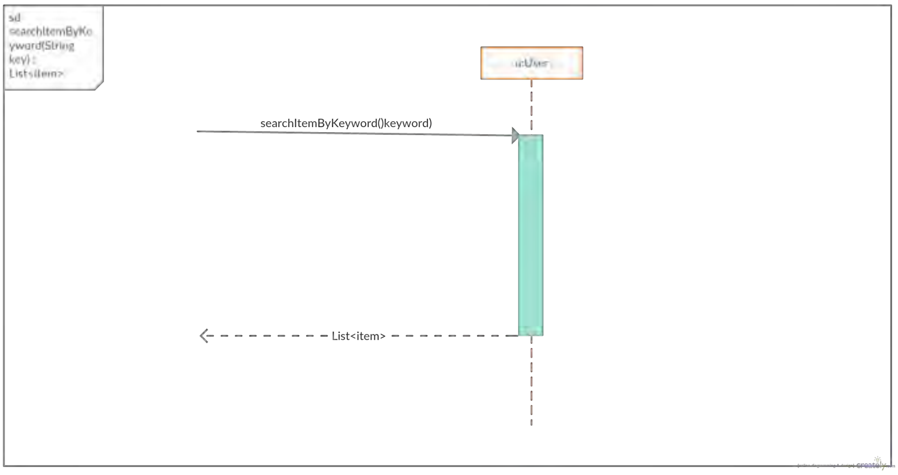
1. User types what he/she wants to search about and presses enter (click search, touches the button).

2. From the listed results user picks what she/he is interested in.

Postconditions:

- User found what he was looking for.

Figure 4.2: Sequence Diagram 3: Search


sd searchItemByKeyword(String key) : List<item>
u:User
searchItemByKeyword()keyword)
List<item>

4.1.3 Sequence Diagram 4

Name: Upload
Actor: Registered User
Preconditions:

- User must have an active account.

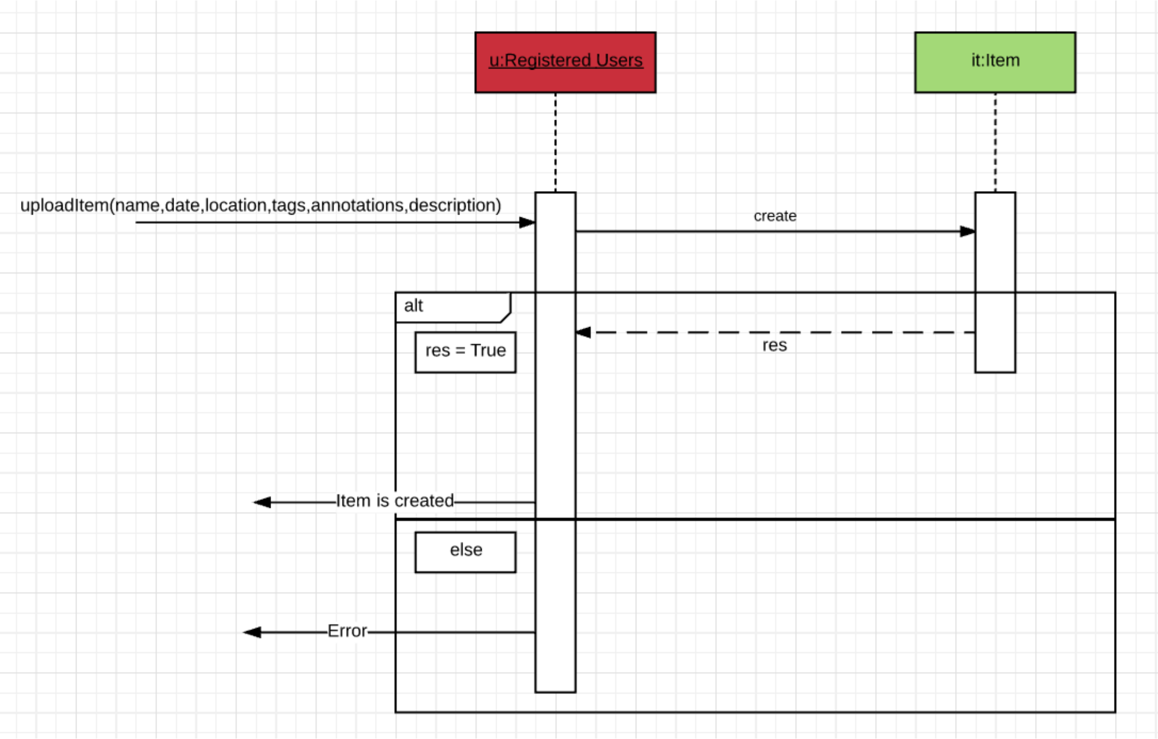
Action Steps:

1. User clicks on "upload" button.
2. From the page that appears, user clicks "upload file from device" button.
3. User chooses a figure and clicks "ok".

Postconditions:

- User's figure is on the system now.

Figure 4.3: Sequence Diagram 4: Upload



4.1.4 Sequence Diagram 8

Name: Report an Item & Check Report
Actor: Registered Users & Moderator
Preconditions:

- User must be logged into the system

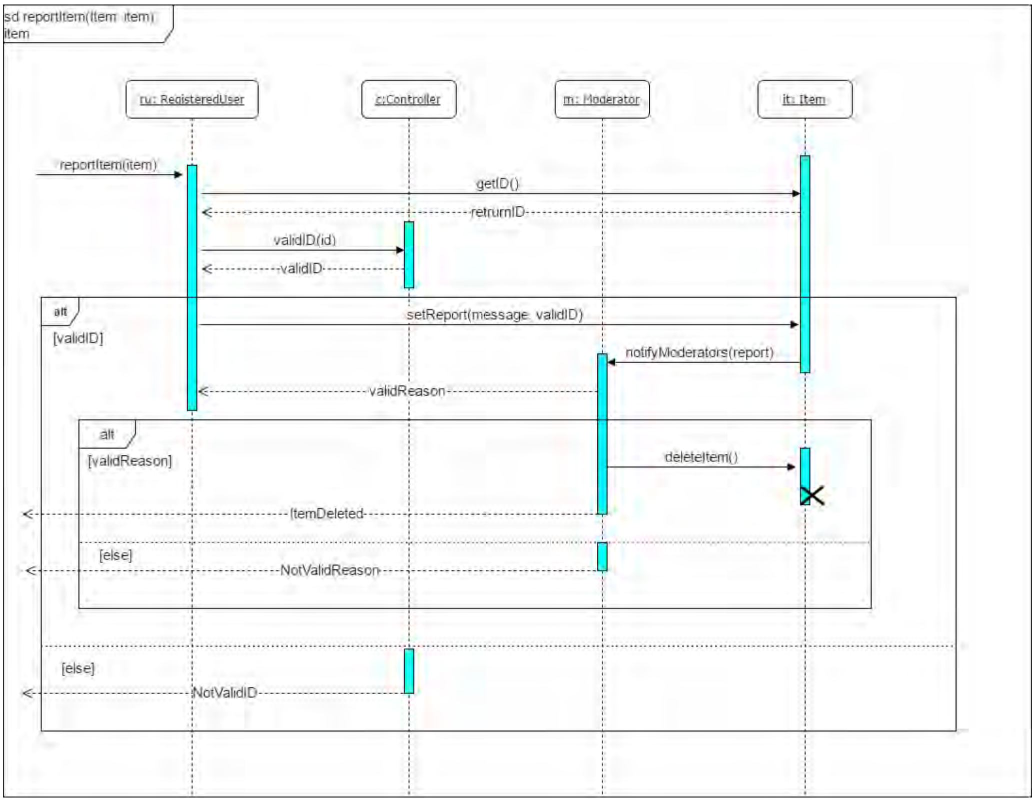
Action Steps:

1. User views the item he/she wants to report.
2. User clicks the "Report the item" button in the page.
3. System moderator gets notified.
4. Moderator deletes the item.

Postconditions:

- System sends an email to the user to figure out the reason of reporting
- Moderator then decides whether or not removing the item.
- Moderator deletes item after decision.

Figure 4.4: Sequence Diagram 8: Report an Item & Check Report



4.1.5 Sequence Diagram 9

Name: Adding Comments
Actor: Registered Users
Preconditions:

- User must be logged into the system

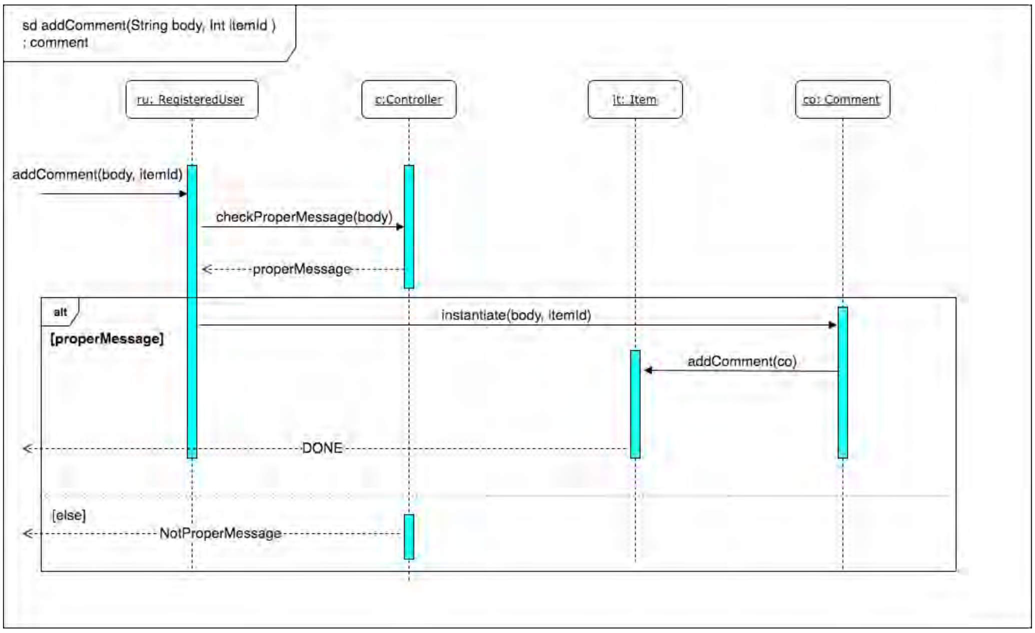
Action Steps:

1. User opens the item page he/she wants to comment.
2. User writes his/her comment on the comment bar.
3. User clicks submit.

Postconditions:

- System updates the item page with users' comments.

Figure 4.5: Sequence Diagram 9: Adding Comments



4.1.6 Sequence Diagram 10

Name: Adding Annotations
Actor: Registered Users
Preconditions:

- User must be logged into the system

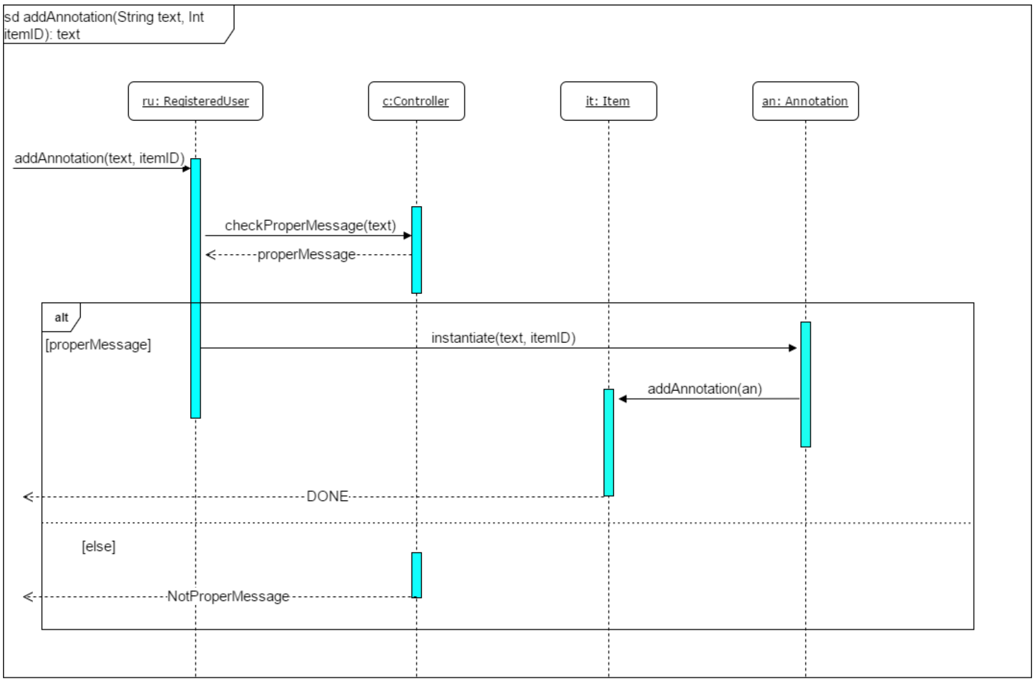
Action Steps:

1. User opens the item page he/she wants to add annotations.
2. User clicks the annotations button to add annotations.
3. User clicks the image to add annotations on it.
4. User add his text/tags to define annotations.
5. User clicks submit.

Postconditions:

- System updates the item page with annotations.
- System updates the item image with blue dots that show the annotation's places.

Figure 4.6: Sequence Diagram 10: Adding Annotations



4.1.7 Sequence Diagram 12

Name: Follow Discussion
Actor: Registered Users
Preconditions:

- User must be logged in to the system, system has already created a discussion

Action Steps:

1. User opens the discussion that he/she wants to follow.

2. User clicks the follow button.

Post-conditions:

- User gets an email when somethings changed on discussion.

- Followed discussion is added to followed discussion list

- Following user is added to followers list of followed discussion

Figure 4.7: Sequence Diagram 12: Follow Discussion

4.1.8 Sequence Diagram 13

Name: Unfollow Discussion
Actor: Registered Users
Preconditions:

- User must be logged in to the system. User must be following the discussion he/she wants to unfollow.

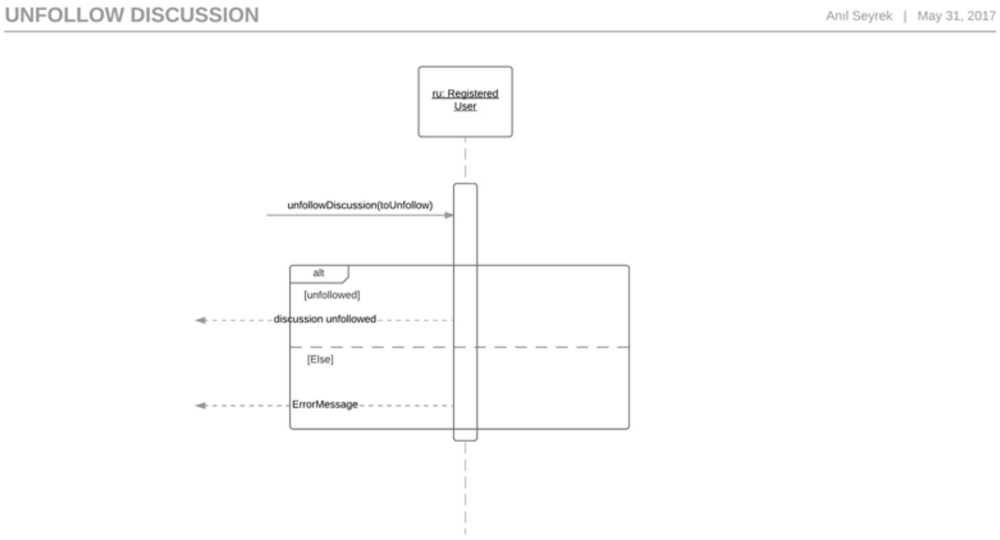
Action Steps:

1. User opens the discussion that he/she wants to unfollow.
2. User clicks the unfollow button.

Post-conditions:

- Unfollowed discussion is removed from followed discussions list.
- Unfollowing user is removed from followers list of unfollowed discussion.

Figure 4.8: Sequence Diagram 13: Unfollow Discussion



4.1.9 Sequence Diagram 15

Name: Take action on reported posts
Actor: Moderators
Preconditions:

1. The post must have been checked by the moderator to determine if it is inappropriate.
2. The user must be logged in as Moderator.

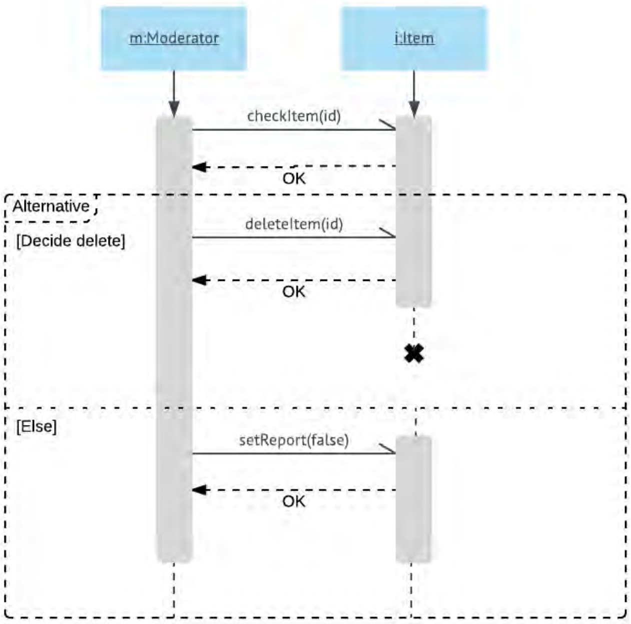
Action Steps:

1. Moderator views the post
2. Deletes the post if it is necessary, leaves it as it is otherwise.

Postconditions:

- Inappropriate posts are cleaned from the system.
- Reports are checked by the moderator.

Figure 4.9: Sequence Diagram 15: Take action on reported posts



4.1.10 Sequence Diagram 16

Name: Share an Item
Actor: Registered Users
Preconditions:

- User must be logged into the system.

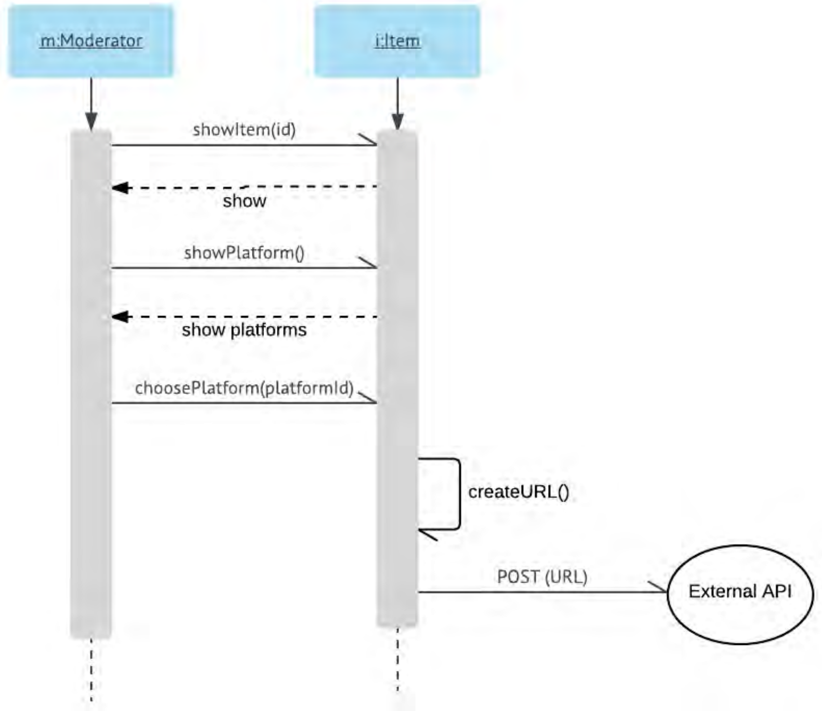
Action Steps:

1. User opens the item he/she wants to share.
2. User clicks the "Share" button.
3. Pop-up windows opens and share-supported platforms are shown.
4. User chooses one of the platforms that she/he wants to share the item on.
5. System creates an Url for the item and connects the API of the platform chosen.

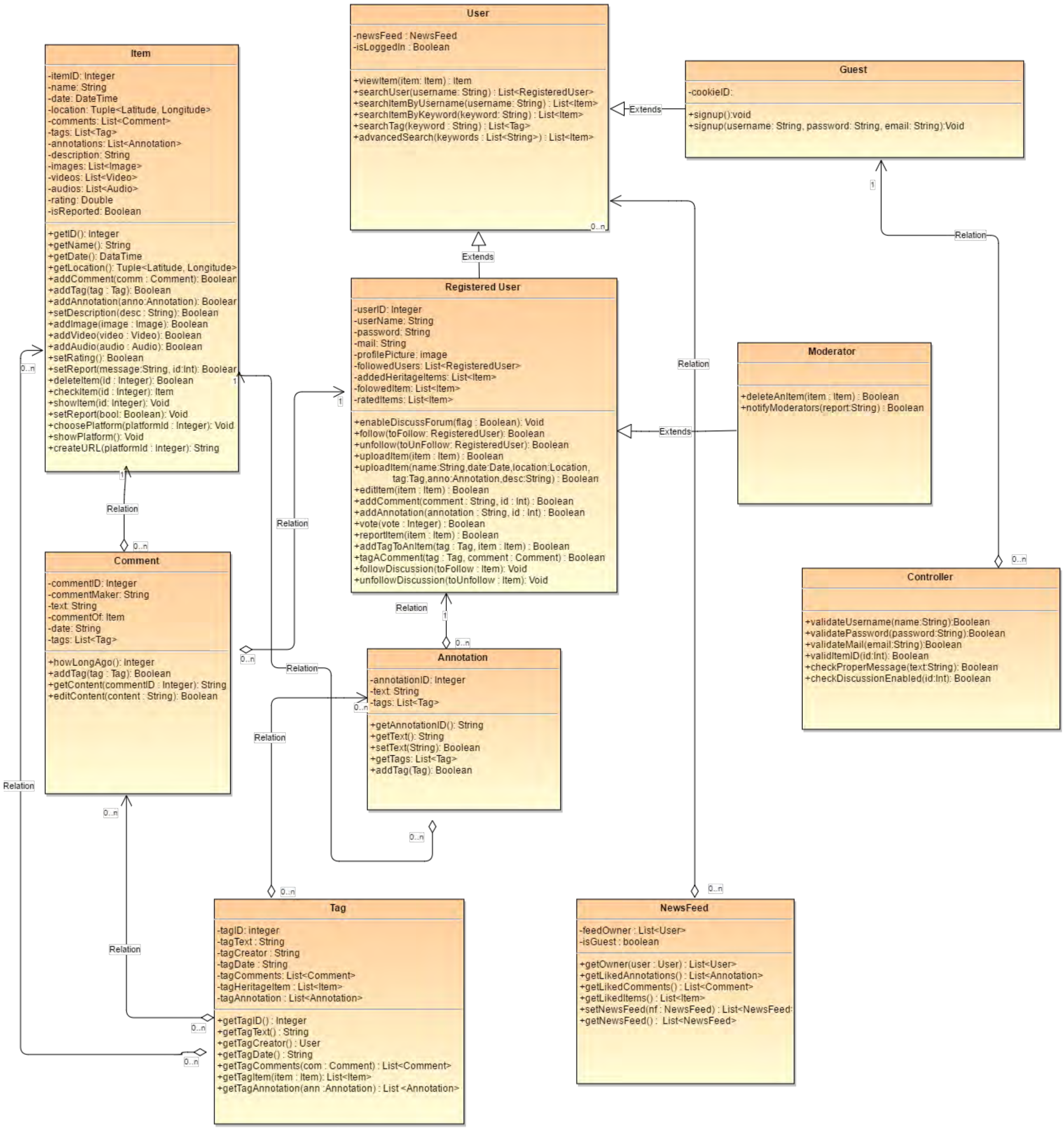
Postconditions:

- System is capable of creating read-only Url for supported platforms.

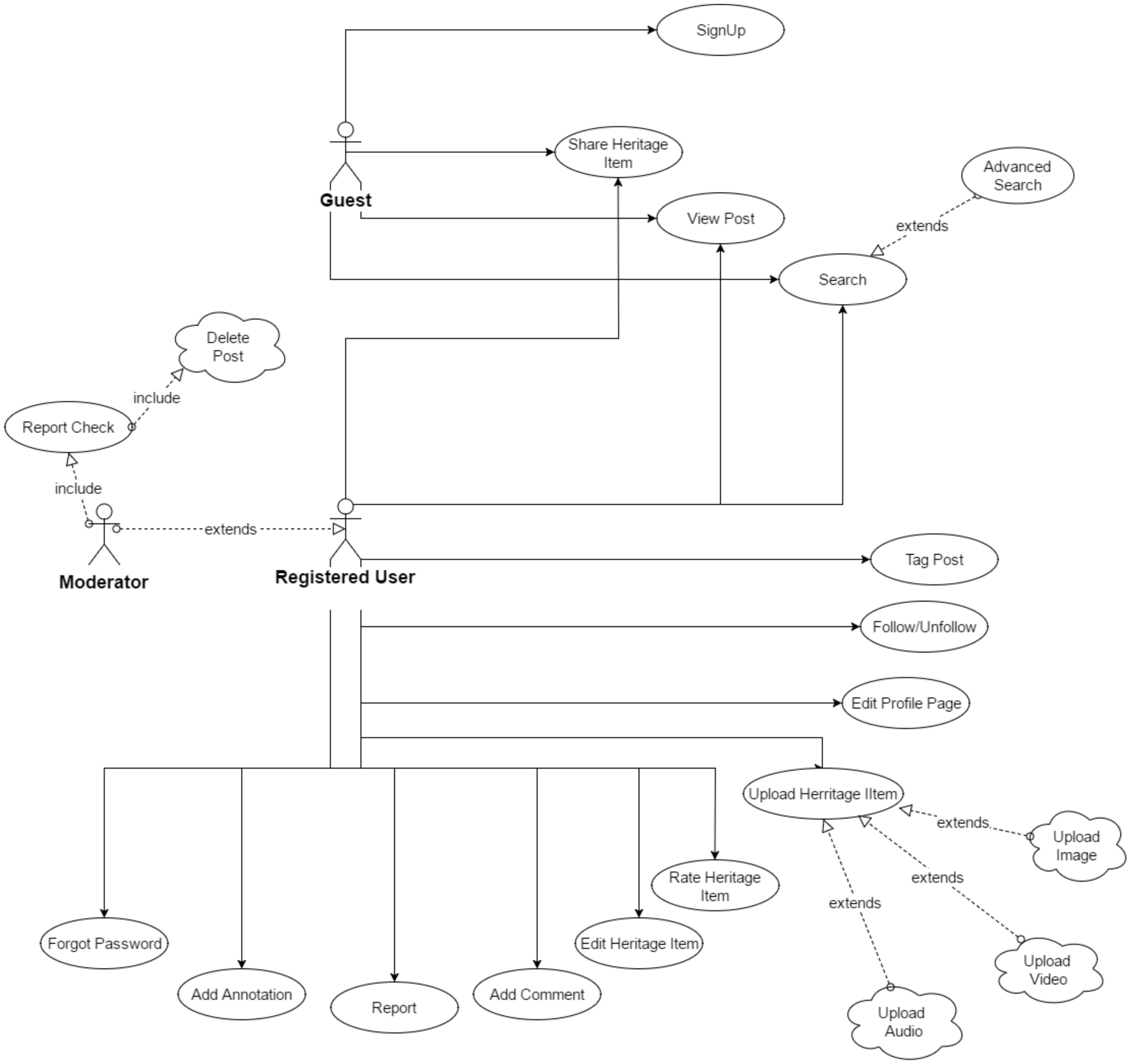
Figure 4.10: Sequence Diagram 16: Share an Item



4.2 Class Diagram


User
-newsFeed : NewsFeed
-isLoggedIn : Boolean
+viewItem(item: Item) : Item
+searchUser(username: String) : List<RegisteredUser>
+searchItemByUsername(username: String) : List<Item>
+searchItemByKeyword(keyword: String) : List<Item>
+searchTag(keyword : String) : List<Tag>
+advancedSearch(keywords : List<String>) : List<Item>
0..n
Extends
Guest
-cookieID:
+signup():void
+signup(username: String, password: String, email: String):Void
1
Relation
Item
-itemID: Integer
-name: String
-date: DateTime
-location: Tuple<Latitude, Longitude>
-comments: List<Comment>
-tags: List<Tag>
-annotations: List<Annotation>
-description: String
-images: List<Image>
-videos: List<Video>
-audios: List<Audio>
-rating: Double
-isReported: Boolean
+getID(): Integer
+getName(): String
+getDate(): DataTime
+getLocation(): Tuple<Latitude, Longitude>
+addComment(comm : Comment): Boolear
+addTag(tag : Tag): Boolean
+addAnnotation(anno:Annotation): Boolear
+setDescription(desc : String): Boolean
+addImage(image : Image): Boolean
+addVideo(video : Video): Boolean
+addAudio(audio : Audio): Boolean
+setRating(): Boolean
+setReport(message:String, id:Int): Boolear
+deleteItem(id : Integer): Boolean
+checkItem(id : Integer): Item
+showItem(id: Integer): Void
+setReport(bool: Boolean): Void
+choosePlatform(platformId : Integer): Void
+showPlatform(): Void
+createURL(platformId : Integer): String
0..n
1
1
Extends
Registered User
-userID: Integer
-userName: String
-password: String
-mail: String
-profilePicture: image
-followedUsers: List<RegisteredUser>
-addedHeritageItems: List<Item>
-folowedItem: List<Item>
-ratedItems: List<Item>
+enableDiscussForum(flag : Boolean): Void
+follow(toFollow: RegisteredUser): Boolean
+unfollow(toUnFollow: RegisteredUser): Boolean
+uploadItem(item : Item) : Boolean
+uploadItem(name:String,date:Date,location:Location,
tag:Tag,anno:Annotation,desc:String) : Boolean
+editItem(item : Item) : Boolean
+addComment(comment : String, id : Int) : Boolean
+addAnnotation(annotation : String, id : Int) : Boolean
+vote(vote : Integer) : Boolean
+reportItem(item : Item) : Boolean
+addTagToAnItem(tag : Tag, item : Item) : Boolean
+tagAComment(tag : Tag, comment : Comment) : Boolean
+followDiscussion(toFollow : Item): Void
+unfollowDiscussion(toUnfollow : Item): Void
1
Relation
Moderator
+deleteAnItem(item : Item) : Boolean
+notifyModerators(report:String) : Boolean
Extends
Relation
Relation
1
0..n
Comment
-commentID: Integer
-commentMaker: String
-text: String
-commentOf: Item
-date: String
-tags: List<Tag>
+howLongAgo(): Integer
+addTag(tag : Tag): Boolean
+getContent(commentID : Integer): String
+editContent(content : String): Boolean
Relation
0..n
Relation
Relation
1
0..n
Controller
+validateUsername(name:String):Boolean
+validatePassword(password:String):Boolean
+validateMail(email:String):Boolean
+validItemID(id:Int): Boolean
+checkProperMessage(text:String): Boolean
+checkDiscussionEnabled(id:Int): Boolean
0..n
Annotation
-annotationID: Integer
-text: String
-tags: List<Tag>
+getAnnotationID(): String
+getText(): String
+setText(String): Boolean
+getTags: List<Tag>
+addTag(Tag): Boolean
0..n
Relation
Relation
0..n
0..n
0..n
0..n
Relation
Tag
-tagID: integer
-tagText : String
-tagCreator : String
-tagDate : String
-tagComments: List<Comment>
-tagHeritageItem : List<Item>
-tagAnnotation : List<Annotation>
+getTagID() : Integer
+getTagText() : String
+getTagCreator() : User
+getTagDate() : String
+getTagComments(com : Comment) : List<Comment>
+getTagItem(item : Item): List<Item>
+getTagAnnotation(ann :Annotation) : List <Annotation>
0..n
0..n
NewsFeed
-feedOwner : List<User>
-isGuest : boolean
+getOwner(user : User) : List<User>
+getLikedAnnotations() : List<Annotation>
+getLikedComments() : List<Comment>
+getLikedItems() : List<Item>
+setNewsFeed(nf : NewsFeed) : List<NewsFeed:
+getNewsFeed() : List<NewsFeed>

4.3 Use Case Diagram



Test Case 1

Title: Edit Profile Page.

Description: To modify personal information.

Related Requirement: 1.1.2.3

Preconditions:

- User must be logged in.
- User must be viewing his/her own profile page.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks the field wants to be edited	Edit buttons in profile page	Editing dialog is opened			
2	User changes his name to 'cRaZYY_34'	Save button in the editing dialog	Name is changed and user is redirected to his/her profile page.			

Postcondition: Personal information is modified.

Test Case 2

Title: Upload Audio Heritage Item

Description: To upload existing audio file as heritage item to the system.

Related Requirement: 1.1.2.5.3

Preconditions:

- User must be logged in.
- User must be viewing uploading screen.
- An audio file must exist in the system.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks upload button	Upload button	Upload screen is viewed			
2	User uploads audio file 'foolish_casanova.mp3'	Audio uploading button	File is uploaded and user is redirected to item's page.			

Postcondition: 'foolish_casanova.mp3' is uploaded to the system.

Test Case 3

Title: Reset Password & Email.

Description: To reset password & email.

Related Requirement: 1.1.2.4

Preconditions:

- User must be logged in.
- User must be viewing his/her account page.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks reset password button	Reset password button	Reset password screen is viewed			
2	User types new password and submits	Submit new Password button	Password is updated and redirected to user account page.			

Postcondition: User's password is changed.

Test Case 4

Title: Search heritage items with their names.

Description: To search items from search bar with giving name.

Related Requirement: 1.1.4.1

Preconditions:

- User must be logged in.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks search bar and types name of heritage item to it.	Search bar in top of the page	Heritage item name is written			
2	User clicks search button	Search button near search bar	Heritage items related to the name written is shown to the user.			

Postcondition: User views the heritage item that looking for.

Test Case 5

Title: Registration to the system.

Description: To check if an unregistered user can register properly.

Related Requirement: 1.1.1.1.1

Preconditions:

- User is on the registration page.
- Mandatory fields of registration are visible.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User fills mandatory fields; ezgiy - 1234 - ezgiy@***.com	Mandatory fields	New user created and added into database. Toast message is shown. Back to Login page link appears			
2	User fills mandatory fields except email.	Save button. System interruption.	A toast message is shown saying to fill missing field.			

Postcondition: New user is added into database.

Test Case 6

Title: Adding tag.

Description: Tagging of heritage items via plugin by registered user.

Related Requirement: 1.1.2.12.3

Preconditions:

- User is already login.
- Mentioned heritage item exists on the page.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User start tagging by typing '@' character (on website).	Most related tags	A drop down menu is displayed which shows most related tags			
2	User start tagging by typing '@' character.	Most related tags	A suggestion of most related tag appears on the keyboard of the phone			

Postcondition: New tag is added into the heritage item.

Test Case 7

Title: View Contributer.

Description: To view past contributers of a heritage item.

Related Requirement: 1.1.2.13

Preconditions:

- User must be logged in.
- User must be viewing a heritage item.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks show contributes button.	item history page	List of contributers appear			

Postcondition: User founds list of past contributers.

Test Case 8

Title: Follow another user.

Description: To be notified when a particular user makes a contribution.

Related Requirement: 1.1.2.6

Preconditions:

- User must be logged in.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks the link to the profile page of the user she/he wants to follow.	Profile page link	Profile page of the user is viewed			
2	User clicks on follow user button.	Follow user button	A message is shown saying that user is now following them			

Postcondition: User is able to follow user's contribution to the website.

Test Case 9

Title: Sharing an item.

Description: To share an item through an external website.

Related Requirement: 1.1.1.3

Preconditions: None

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks the share with Twitter button	Share with Twitter button	User is redirected to "Share a Link" page in Twitter			

Postcondition: User is able to share an item she/he finds interesting.

Test Case 10

Title: Upload Image Heritage Item

Description: To upload existing image file as heritage item to the system.

Related Requirement: 1.1.2.5.2

Preconditions:

- User must be logged in.
- User must be viewing uploading screen.
- An image file must exist in the system.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks upload button	Upload button	Upload screen is viewed			
2	User uploads image file 'tower_of_pisa.jpg'	Image uploading button	File is uploaded and user is redirected to item's page.			

Postcondition: 'tower_of_pisa.jpg' is uploaded to the system.

Test Case 11

Title: Upload Video Heritage Item

Description: To upload existing video file as heritage item to the system.

Related Requirement: 1.1.2.5.4

Preconditions:

- User must be logged in.
- User must be viewing uploading screen.
- A video file must exist in the system.

Steps:

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	User clicks upload button	Upload button	Upload screen is viewed			
2	User uploads video file 'at_pisa_tower.mp4'	Image uploading button	File is uploaded and user is redirected to item's page.			

Postcondition: 'at_pisa_tower.mp4' is uploaded to the system.

Chapter 5

User & System Manuals

5.1 Frontend

5.1.1 System Manual

Requirements

- Node.js
- npm
- Google Chrome

Setup

To setup this project, follow these steps:

1. Clone the project to local.
2. Switch to *frontend* directory. (`$ cd bounswe2017group11/frontend`)
3. Install required packages. (`$ npm install`)

Run locally

1. Run scripts. (`$ npm start`)
2. Launch Google Chrome
3. Visit `localhost:3000`

Build & Deploy

1. Run scripts. (`$ npm run build`)
2. Upload files in build folder to S3 Bucket
3. Set S3 Bucket for static website hosting. Give public access

5.1.2 User Manual

Requirements

- Google Chrome

Usage

1. Launch Google Chrome
2. Visit `http://culturage.s3-website-us-east-1.amazonaws.com/`

5.2 Android

5.2.1 System Manual

Requirements

- Android 5+
- User Permissions

Setup

To setup this project, follow these steps:

1. Download the (`$ APK` to your mobile.
2. Touch the file, and it will be installed

5.2.2 User Manual

Requirements

- Android 5+
- User Permissions
- Internet Connection

Usage

1. Open the app
2. Enjoy with our app

5.3 Plugin

5.3.1 Requirements

- Google Chrome

5.3.2 Install

1. Launch Google Chrome
2. Open *Extensions* window
3. Enable *Developer mode*
4. Click *Load unpacked extensions*
5. Select plugin directory

5.3.3 Usage

1. Launch Google Chrome
2. Visit corresponding website
3. To enable plugin click *Culturage* logo

5.4 Backend

5.4.1 System Manual

Requirements

- Python 3

How to Setup

To setup this project, follow these steps:

1. Clone project to local.
2. Install Python 3
3. Install MySQL
4. Install Virtualenv (`$ pip3 install virtualenv`)
5. Create virtual environment for this project (`$ virtualenv venv`)
6. Switch to venv
   - If you are using Windows (`$ venv\Scripts\activate`)
   - If you are using UNIX (`$ source venv/bin/activate`)
7. Install dependencies locally (`$ pip3 install -r requirements.txt`)
8. Copy .env.example (`$ cp .env.example .env`)
9. Set your local environment variables via .env (such as database credentials)

10. Run database migrations (`$ python3 manage.py migrate`)

11. Run server locally (`$ python3 manage.py runserver`)

12. If you get success message, server's running, visit `http://127.0.0.1:8000/`

Apache Deployment

1. Follow Setup Instructions to step 10

2. Configure your server based on `apache.conf` file

Sample Data

It has more than 100 cultural heritage items.

1. Import sample data from `sample-data/sample-data.sql` file to your database.

2. Put `sample-data/media` folder to your root directory for sample item images.

5.4.2 User Manual

Our web service is using `JWT` tokens for authorization. You can get your token from `api/auth/login` or `api/auth/register` endpoints. You have to send token in request header via `Authorization` key with `JWT [YOUR_TOKEN]` value. You can study our API from `Postman Documentation`

Testing API

1. Install Postman app

2. Import `culturage.postman_collection.json` or use "*Run in Postman*" in documentation

3. Create a new environment from "`Manage Enviroments`" option

4. Add two key:value pair `url:http://52.90.34.144:85` and `token:YOUR_TOKEN`

5. You can get token from *login* or *register* endpoint

6. Don't forget to select current environment as environment that you created newly

Chapter 6

Project plan

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