IT2901 Sintef Storytelling

Tor Barstad torob@stud.ntnu.no
Jon-André Brurberg jonandbr@stud.ntnu.no
Hallvard Jore Christensen hallvarc@stud.ntnu.no
Øyvind Hellenes oyvihell@stud.ntnu.no
Vegard Storm vegs@stud.ntnu.no
Jørgen Rugelsjøen Wikdahl jorgenrw@stud.ntnu.no

April 25, 2014

Contents

1 Introduction

This report is written as a bachelor thesis by computer science students at NTNU. The project revolves around upgrading and expanding features of a multi platform app called "Stedr" which is currently in beta. Stedr's purpose is to enable people to share their stories about places around the world. This can be anything from a famous attractions to just an ordinary building Trondheim. The contributors will be able to share stories and media through external services like Digitalt Fortalt, Flickr, Instagram, Soundcloud etc. With the application, users can view other peoples stories and images to help them explore a certain place.

1.1 The subject: IT2901

lorem lipsum

1.2 Stedr

lorem lipsum dorem sit amet.

1.3 Stakeholders

1.3.1 The team

The team consists of six students all taking a Bachelor degree in Informatics at The Norwegian University of Science and Technology (NTNU). In our team we have a great variation in areas of expertise and knowledge which hpelped us greatly during the course of the project. Having expertise in many different areas we could help each other and share knowledge across the group to make everybody suited to different tasks. The importance of this project made the whole group very motivated to succeed and make for a good result. We are:

Hallvard Jore Christensen	hallvarc@stud.ntnu.no
Jon-André Brurberg	jonandbr@stud.ntnu.no
Jørgen Rugelsjøen Wikdahl	jorgenrw@stud.ntnu.no
Tor Barstad	torob@stud.ntnu.no
Vegard Storm	vegs@stud.ntnu.no
Øyvind Hellenes	oyvihell@stud.ntnu.no

1.3.2 Customer

Our customer is SINTEF (The Foundation for Scientific and Industrial Research). They are the largest independent research organisation in Scandinavia. The organiszation was established at the Norwegian Institute of Technology (NTH) in Trondheim in 1950 and expanded rapidly in the following years. Our contact person from SINTEF is Jaqueline Floch and she is the primary driver for the app that we are developing.

Jaqueline Floch	*Project Manager* Email: example@example.com **beskrivelse**
Babak	*Intermidiate Manager* Email: example@example.com **beskrivelse**

1.3.3 Course Staff

Moshen	*Supervisorr* Email: example@example.com **beskrivelse**
Moniica Divitini	*Course co-ordinator* Email: example@example.com **beskrivelse**

1.4 Report Structure

Chapter 1
TODO The introduction chapter. Presenting the course, project and the different people involveld in the project.

2 Pre-study

2.1 The stedr application

2.1.1 Existing functionality

Since the application already is considered a working prototype, we will provide a list which gives a description for the functionality. Working functionality is in this report defined as the functionality that is implemented in the frontend or backend. If something is implemented backend it has to be used frontend. A more detailed techincal description is found in the architecture-section.

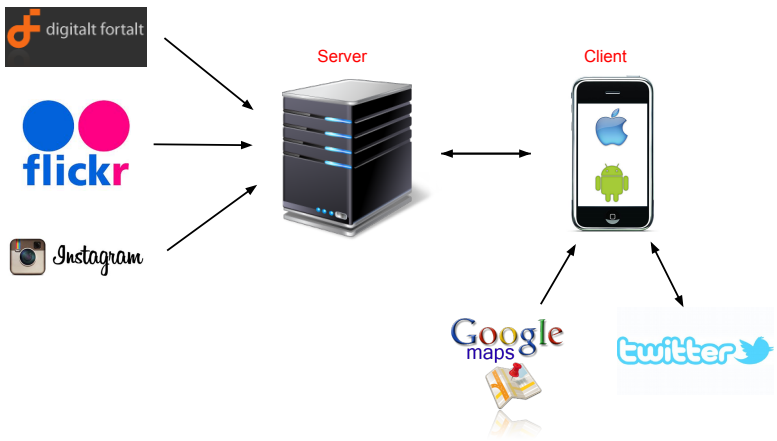

Figure 1 – A simple overview of the architecture

- Browse a map and zoom in and out.
- Load **places**.
- Click on a place in the map and access **stories** from Digitalt Fortalt.
- Get social media related to a place from the content providers Instagram and Twitter.
- Go to a users exact position on a map.
- Search for a location in the map.

2.1.2 Limitations

There are some limitations to the system that needs to be further developed, and some that probably would require total architectural review of the project to be fixed. Our task is to continue the development of the applicaton. An overview of the features we are going to improve are discussed in the requirements-section. Flaws that arised during the development which requires a new architecture will be discussed in the conclusions-section under Recommendations.

3 Project organization

3.1 Responsibility Areas

Good delegation of responsibilities helps so that someone at all times have an overview over what tasks needs to be done in specific areas. This also makes it easier to estimate workloads and delegate tasks during the group meetings. It is important to note that even though there are specific responsibility areas, all the group members will be able to get practical experience in all of the project areas, even though the time spent in different areas will be distributed individually according to the responsobility areas.

Øyvind Hellenes - Scrummaster Øyvind was selected as the scrum master because of his leadership qualities and because he early on took an interest in the organizatorial part of the project.

Jon-André Brurberg - Project leader Jon-André is the driving force in this team and hence, he is also the project leader. Additionally, Jon Andre showed interest in the documentation so he is also responsible for this.

Tor Økland Barstad - Technical coordinator Tor, with his compentance and knowledge of programming, is our technical coordinator and will thus oversee the code and functions as a technical supervisor for the group.

Jørgen Rugelsjøen Wikdahl - Testing manager Jørgen is responsible for testing the application to make sure the application has as few bugs as possible.

Hallvard Jore Christensen - Report coordinator Hallvard have the responsibility of managing the report. This is because of his previous expiriance with report documentation.

Vegard Storm - Usability manager Vegard showed interest in making sure the application is as user friendly as possible and will manage that aspect of the project.

3.2 Process model

For the process model we chose to use the Scrum framework. This was the most natural choice for us amongst the agile methods since it's a system we all have experience with through previous projects. There are many advantages working with Scrum. It gives clear priority for features and deadlines, which will allow us to focus more of our energy on other vital tasks. This approach promotes communication and transparency. All the team members as well as the client always knows what's going on and the current tasks' development through the product backlog. With the backlog cards, the whole production team is also involved with the overall time estimate, which makes it fairly accurate and controllable.

We considered a few other methods as well, like kanban and XP, but came to the conclusion that Scrum was the system for us. This was due to Scrums many structured rules which brings order, but still allows us the freedom we might need during the projects development.

With Scrum we'll work in iterations called "Sprints" which are typically a week or two, we also stibe towards making these sprints incremental. Doing this, the model is designed, implemented and tested incrementally, feature by feature, until the project is finished. The advantage here is that for every sprint we have a working product to show for, which is a good referance to have, both for ourselves and the customer.

Since we already have a working project from the very beginning for us to further develop, there is some obvious phase partitions. The first consists mainly on assessing the current version of the product and define the path ahead before we start the actual programming. This will be done through thorough dialog and discussion with SINTEF, to give us a unison idea of where the product are heading. User evaluation is also important in this phase, both internally and externally within the target user group. And of course technology and framework selection. After this comprehensive planning, the actual coding phase can begin. The sprints will be a big part of this, and since we are working incrementally; So we will do with the testing. Following this: the evaluating phase. In which user tests hopefully will force as many problems and bugs with the early version to surface, for us to correct.

Prototypes through a digital mock-up will be important in the planning phase. We have chosen to use Balsamiq for this, which will mean we'll have an interactive prototype mock-up to show the customer, and should also make sure we're all on the same page. This makes it easier to have something concrete/"physical" as a reference.

3.3 Development Environment

Since our project is based on further developing on an existing product, there's an advantage in using the same main framework as the previous developers. We decided to use Titanium to easily develop a multi-platform app, but we also took a close look at other options (like phonegap) and compared them meticulously in their most critical aspects. With the Titanium framework we use the Titanium SDK which is based on eclipse but tailored for it. For sharing code, Git was our system of choice, mainly because we were already familiar with it, and know it has all the functionality we could need throughout the project. Other documents and files, like notes, summaries, etc. we decided to share through a dedicated Google Drive and Dropbox folder, because each has its own advantages in different aspects. As SCRUM service we first choose Agilefant, but later decided to just use spreadsheets instead.

This project obviously involves working with a big set of APIs, like social media, dictionary and other media services. These'll play a great part of the development and introduce other frameworks we'll have to account for. For communication we often use mail and

chat-services, but we prefer more "personal" forms of communication like a video chat through skype, phone calls and/or ideally, meetings in person.

3.4 Project Planning

3.5 Example of Status report and Activity plan

3.6 Deviations and lessons learned

In terms of management, we have learned some important lessons and made some adjustments to our organisatory model during the course of this project.

With our process modell, Scrum, we found that following it by the book, became very troublesome. Therfore we decided to make some modification to the original model. The main problem with using Scrum by the letter is that we are all students, and this project only counts for half of the semesters study points. This means we all have different schedules, and thus making it difficult to have daily meetings. End meetings, or retrospective meetings is also something we haven't prioritized much.

In Scrum it is common to use something called planning poker when deciding how long tasks should take. This essentially means that everyone "votes" on how time consuming they think a given task will be. We found this to be a little unnecessary because its usually so imprecise, and have therefore chosen to just let the persons responsible give their judgements to save time.

As for lessons, we have especially learned about the importance of clear milestones. In the beginning we had very unclear goals and this affected the group. It was later solved when we got a priority list from our cutomer. Working without a clear focus can be challenging for the team members.

We have also learned how crucial it is to have good communication. Because of a misunderstanding between two of our customers we spent a week working on the user interface of Stedr when we should have prioritised integrating API's instead.

4 System Requirements Specification

4.1 Purpose

This is the software requirement specification for the new version of Stedr, both the backend system that provides content and also the frontend that shows the content and the context of the content to the user. Here the traditional architectural terms backend - and frontend are used, but there are some subtleties to this term, as the frontend itself is managing a content service of its own.

4.2 Intended audience and reading suggestions

Intended readers for this document are current and future developers, and the customer. The reader should also be noted that the SRS both can be read as a stand-alone document to get an overview of the rationalization behind the development process, but that it also is a part of the project report as a whole

4.3 References

The software requirements are based on the standard as provided by ISO/IEC:25010 **25010** and also the models that can be found in this report's section for architecture and modelling. References to the ISO-standard and other literature are found at the end of the project report under references.

4.4 Product perspective

Originally Stedr is a product developed by students at NTNU as a part of the subject TDT4290, and this application will form a basis for out continued development. The state of the exisiting application is considered to be a working prototype, and to some degrees it is an application that is built up with a traditonal server-client architecture. A simple technical overview of the system is provided below.

4.5 User classes and charateristics

The users of the program mainly divide into two categories. One is the primary user group which are using the frontend to see content. A typical user of this sort is a highschool student which is introduced to the program in the context of cultural heritage awareness. Most of these student will likely not use the program outside of the school context, but

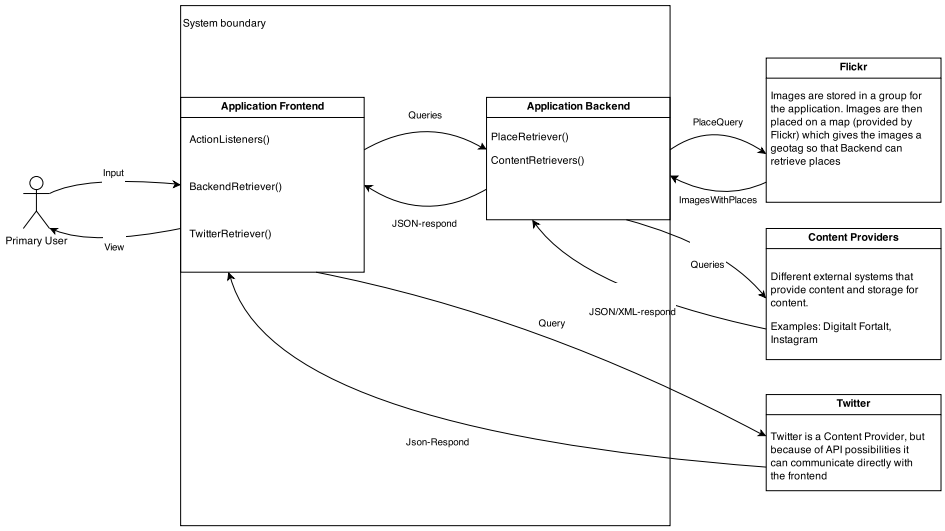

Figure 2 – A simple technical overview of the architecture

there is a possiblity that some of these students transits to the second category. Under the second category there are two user types, one is the content providers and the other is the maintainer. (See another section)

4.6 Operating environment

The frontend application of the system is a mobile application which aims to run on both the Android and iOS platform. Because of difficulties with developing towards the iOS platform without equipment from Apple, it's still discussed how the system is going to be tested on iOS. The backend of the server running as a server application, provided by Heroku.

4.7 Product functions

4.8 Design and Implementation Constraints

4.9 User Documentation

4.10 Assumptions and Dependencies

4.11 External Interface Requirements

4.12 User Interfaces

4.13 Hardware Interfaces

4.14 Software Interfaces

4.15 System Features

SF-1	
Name	Find place on map
Priority	H
Goal	To browse the map to find a given place
Actors	Primary User
Preconditions	1. The home screen is displayes 2. The internal system and external systems are running 3. The device has a internet connection
Stimulus-Response	1. The home screen is displayes 2. The internal system and external systems are running 3. The device has a internet connection
Alternate Flow	2a The place does not exist and is not shown on the map
Functional Requirement	A user should be able to access and browse a map, with places as pinpoints at their respective geographical location. The pinpoints should contain the picture and information found on Flickr. Group places close to eachother in one icon on map.
Related Use Cases	1,3
Dependencies	none

SF-2	
Name	Open menu
Priority	H
Goal	Open the drawer menu
Actors	Primary User
Preconditions	1. 2,3 4 A screen with the menu button
Stimulus-Response	1. The user clicks the menu button 2. The menu opens
Alternate Flow	1a The user clicks the menu button, and the menu is already open 2a The menu closes
Functional Requirement	A button with the possibility to open the menu should always be presented to the user, so that the user easily can navigate the application.
Related Use Cases	1,2
Dependencies	none

SF-3	
Name	Search for a location
Priority	M
Goal	Go to a location on the map
Actors	Primary User
Preconditions	1. 1,2,3
Stimulus-Response	1. The user searches for a location with the search bar in the map view. 2. The map navigates to the location
Alternate Flow	2a Location is not found and is not navigated to.
Functional Requirement	A search bar related to the map should be presented to the user, so the user can search for locations (independent of places) to see if there are any stories at that place.
Related Use Cases	1
Dependencies	none

SF-4	
Name	Refresh map
Priority	H
Goal	Update the map with content.
Actors	Primary User
Preconditions	1. 1,2,3
Stimulus-Response	1. The user clicks the update button. 2. The map refreshes and show new places
Alternate Flow	2a No new places are found, so no places are added to the map.
Functional Requirement	The user should be presented with a button that makes requests for new places with content when pushed. This function should also be done automatically so that new content is sent to the user within 5 minutes after it's added.
Related Use Cases	1
Dependencies	none

SF-5	
Name	Go to location
Priority	H
Goal	Go to users location.
Actors	Primary User
Preconditions	1. 1,2,3
Stimulus-Response	1. The user clicks the gps button. 2. The map zooms to the users location.
Alternate Flow	2a GPS not available so it can't go to the users location.
Functional Requirement	Since the user has the possibility to navigate the map freely, it should also be possible to quickly navigate to places relevant (in context of location) to him/her.
Related Use Cases	1
Dependencies	none

SF-6	
Name	Open views
Priority	H
Goal	Open views and see the content related to that specific view
Actors	Primary User
Preconditions	1. 1,2,3
Stimulus-Response	1. The user clicks on a view 2. The user changes views at will 3. Content
Alternate Flow	1a If the user clicks a button for the already chosen view, nothing should happen.
Functional Requirement	For navigation in the place view, the user should be presented with different buttons (or tabs) so that the user easily can navigate between content and still have an overview of what types of content the application provides. Preview picture gallery when places are grouped together. Add description about place, own vire for sound. Be able to show place location on map from story. Be able to filter stories by tag, author, institution video/no video. preview stories by sound from SoundCloud
Related Use Cases	3
Dependencies	none

SF-7	
Name	Load content
Priority	H
Goal	Content is loaded from the external systems
Actors	Internal System
Preconditions	1. 1,2,3
Stimulus-Response	1. Access the server as done in the previous version of the system 2. Provide input to the server "placeId=" 3. Content is loaded and a JSON-object is replied by the server
Alternate Flow	1a If the user clicks a button for the already chosen view, nothing should happen.
Functional Requirement	The API for DF has to be changed, without changing the behaviour of the response from the server. In additon to this the server will respond with a new container for the audio content. Other content should be handled as normal. Retrieve collectionfrom DF based on hashtag and location. Retrieve stories in a collection from DF based on tags. Open info retrived from SoundCloud based on hashtags or location. Retrieve information from Instagram based on Hashtags. Be able to get tinyUrls to different content.
Related Use Cases	Null
Dependencies	none

SF-8	
Name	Collection
Priority	H
Goal	Get all places related to a theme.
Actors	Primary User
Preconditions	1. 1,2,3
Stimulus-Response	1. Access the menu bar. 2. Click on the Collections-button 3. Choose a collection 4. Collections view is opened 5. Change to map view 6. Places related to the collections is shown on map
Alternate Flow	3a No Collections are available
Functional Requirement	A container called Collections are to be implemented. Collections. Allow switching between map-related and collection related funtionallity. Display picture, title and description about a collection. Have a storyListView. Preview stories in collection story list. Open story in collection list. Places on map view with icon for each story in colelction. Preview a place for story on map.
Related Use Cases	3
Dependencies	none

SF-9	
Name	Upload content
Priority	M
Goal	Upload content
Actors	Primary User
Preconditions	1. 1,2,3 5 The user has an account at the content provider he or she is trying to upload to. 6 Places related to the collections is shown on map
Stimulus-Response	1. Access the tabs for different views 2. Click the add-button in the views.
Alternate Flow	3a No Collections are available
Functional Requirement	The user should have the possibility to add content so that. Add picture directly from stedr. ask the user for login-credentials the first time, then store locally for continued access. A similar approach for SoundCloud. Have relevant hashtags copied to clipboard. Be able to comment and like pictures on instagram.
Related Use Cases	Null
Dependencies	none

SF-10	
Name	Get help and info
Priority	H
Goal	Be informed
Actors	Primary User
Preconditions	1. 1,2,3
Stimulus-Response	1. Access the drawer menu 2. Click the help button. 3. Select the option for what help you need
Alternate Flow	
Functional Requirement	Introduction for first users. Help available at any time.
Related Use Cases	Null
Dependencies	none

4.16 Product Quality

Guided by ISO:25010, meetings with our supervisor and the feature list given to us by the customer the product qualities that are important for the project is functional suitability, portabilty and maintainability.

4.16.1 Compatibility

4.16.2 Performance Efficiency

Even though the system isn't a part of a critical operation, the new and improved system will have performance efficiency as an important model of quality. The reasoning behind this is that decreased response time between components in the system is specfically asked for at multiple places in the feature list provided by the customer.

As of now the time to load new content from the content providers to the application is slow and random. Because of this there are no exact estimation on the time used to pull content from Digitalt Fortalt and Instagram, but the application should use no more than *300 seconds* to pull new content. Unrelated to the goal of performance issue; the user should be informed that the application isn't a real-time application.

Requirements related to resources utilized by the application when performing it's tasks,

are already met by the prototype. The new version of the application are bound also bound by these goals. Specifically the backend is bound by the resources provided by the 1x Heroku Cloud Platform. Because of the utilization of the Google Maps API, the resources frontend is limited to the bound given to the application from Google Maps.

Regarding capacity used by the the application, there should be an improvement. Because the application is to be used on the go where there may not be any WiFi-hotspots, the application should restrain itself to dowload content that is unrelated to where the user is. Because of the varied content types, it is hard to set a defined limit in how much contents (in terms of megabytes) the application should download. The limitations given to the application will therefore be set by the equation:

$$\text{Bound} = \text{Content from Digitalt Fortalt} + 5 \times \text{Content from Twitter} + 10 \times \text{Picture from Flickr} + 5 \times \text{Picture from Instagram}$$

4.16.3 Reliability

Since the application is going to be online without a team responsible for the technical maintenance, the server should be operative as long as the external content providers are feeding it with content.

Because of the early versioning of the application, the aspect of maturity is not important for this application. Users of the application are few, and they know what the capabilities of the application is. This means that a user follows a rigid pattern and within that pattern, the probability to execute faults is almost non-existing. Functionality outside that pattern is not supported and thereby it's impossible to execute mistakes.

An important charecteristic of the application is that it has to be available just as often as a professional service. This means that under normal circumstances, the uptime of the backend and front should be 99 %

Whenever faults are occuring, it is crucial that the backend has implemented services so that it can recover without the need of a maintainer. Because of the relative simplicity of the backend, the server should restart itself within *180 seconds*

4.16.4 Portability

It is important to the customer that the application is made available on multiple platform as this is a demand by Tag Cloud. The minimal number of platforms which the product should run on is iOS and Android.

Following this, the frontend of the application should be written once and compiled down to both the iOS and Android platform. The backend should provide agnostic responses,

so that the responses can be handled the same by on Android and iOS devices.

Because of the early development phase of the application, there is not a requirement to install the application from the normal application providers Google Play and Apple Store. It is enough that it is possible to install the applications on development devices. This also leads to that the application doesn't need to consider replaceability at this point.

5 Architecture

The current architecture of the application is as shown in the figure 1 from the requirements-section. We have a backend written in Java that retrieves information from services like Digitalt Fortalt, Flickr and Instagram. Digitalt Fortalt is where all the stories are obtained from, Flicker holds all the locations, and the pictures are taken from Instagram based on tags. The information is stored on the server and can now be used by the client, which holds the frontend of the application that is being developed on Appcelerator Titanium, using mainly JavaScript and XML. Twitter is integrated directly into the frontend and does not have to go through the server. This is what we eventually would like to do for all the external services, and completely get rid of the backend, but given the time available for the project and the features the customer wants us to implement, this is not a task that will be developed. We would also like the user to be able to publish to more of the external services via the application. Publish a picture to instagram, add a new location to flickr, or share a story on facebook are all features we would like to add, but are not top priority given our time restrictions.

5.1 Backend

The Backend is written in Java and mainly retrieves data from external APIs and save it on the server so that it can be used by the application.

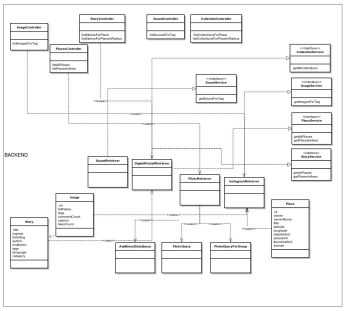

Figure 3 – Backend (Look at end of report for full scale)

5.2 Frontend

The Frontend of the application is an interface to let the user enter, manipulate and view data. It's the part of the application that is being interpreted on the users own device, and is based on XML, TSS and JavaScript for design and functionality.

Every window in the application has a javascript-, tss- and xml-file associated with it. A window can contain various views that can each have different event listeners. What the

user sees depends on the window currently open and its associated xml, tss and javascript files and what happens when interacting with a view depends on the event-listeners attached to that particular view. Interactions can be purely visual or it can trigger core functionalities. For example the refresh button on the map window has an event-listener attached to it so that when the user clicks it, it will attempt to fetch the locations from the server and plot them on the map. It will also animate the refresh icon to spin, giving the user feedback that the click was registered.

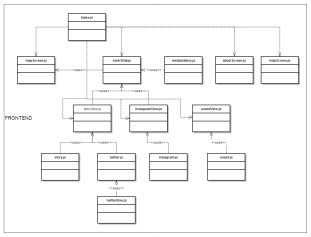

Figure 4 – Frontend (Look at end of report for full scale)

5.3 Use Case

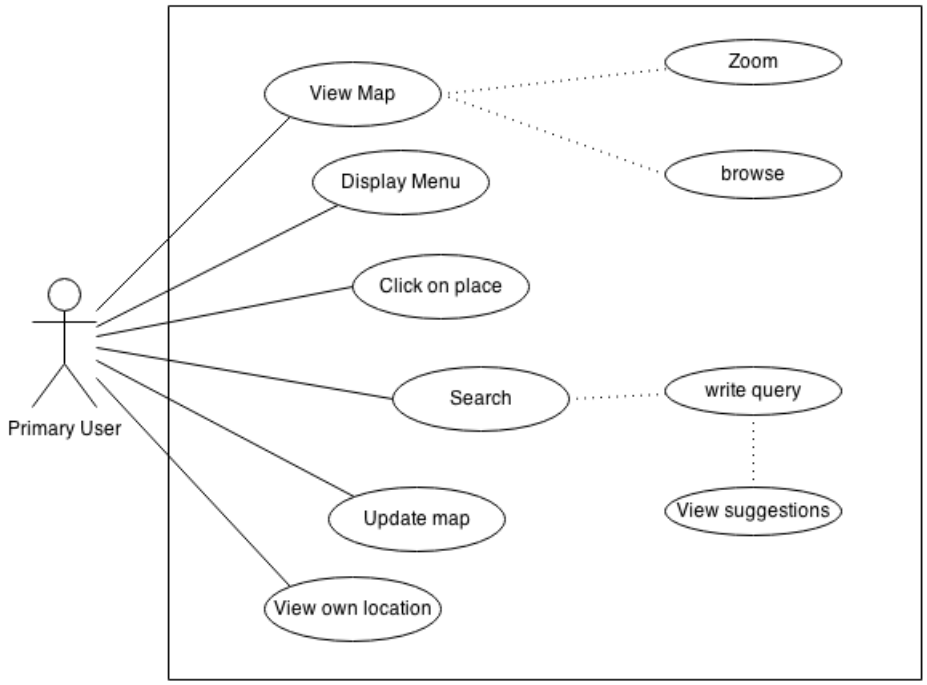

Figure 5 – Map View (Home)

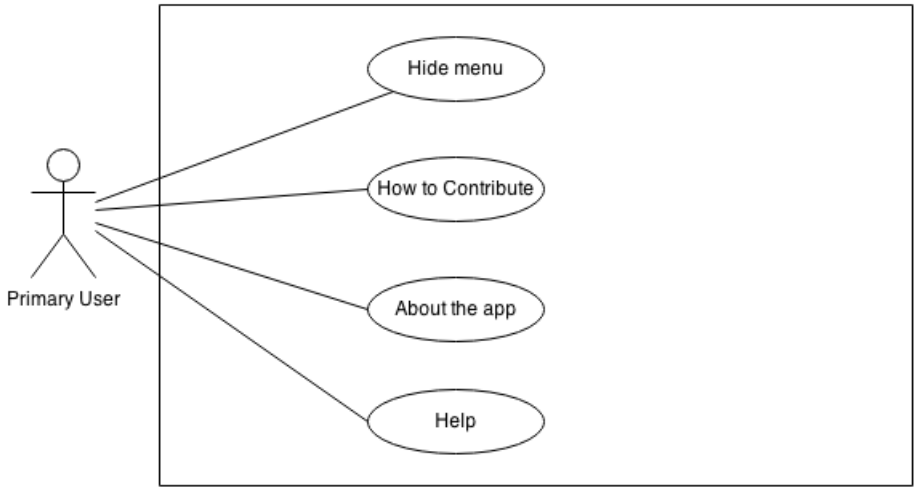

Figure 6 – Menu View

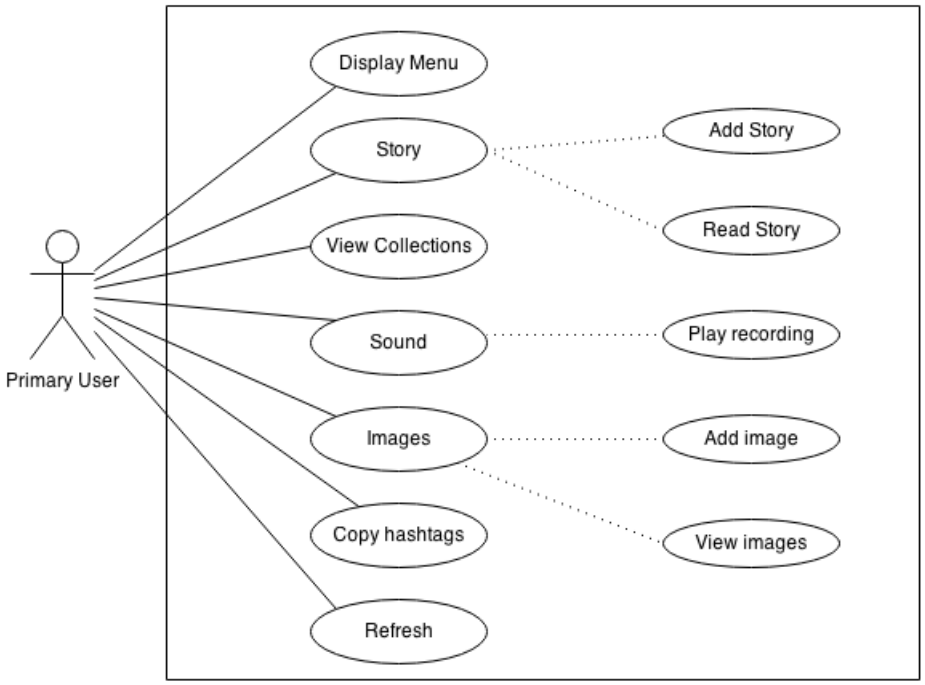

Figure 7 – Place Screen

5.4 Sequence

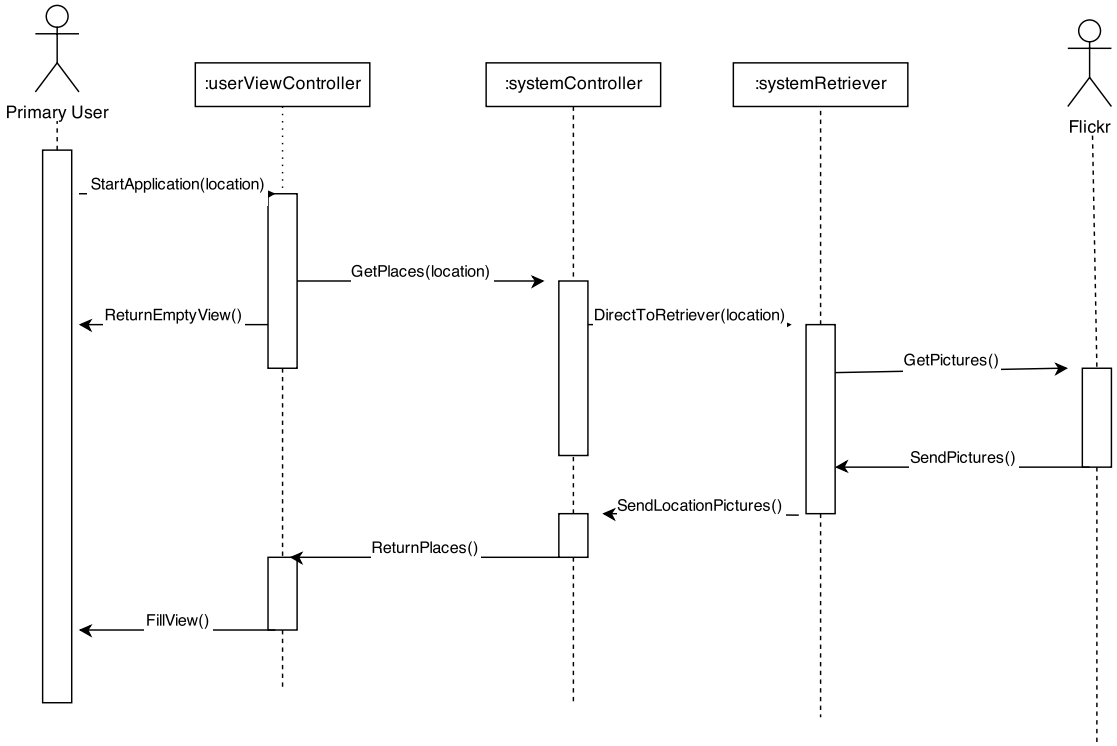

Figure 8 – Get Stories

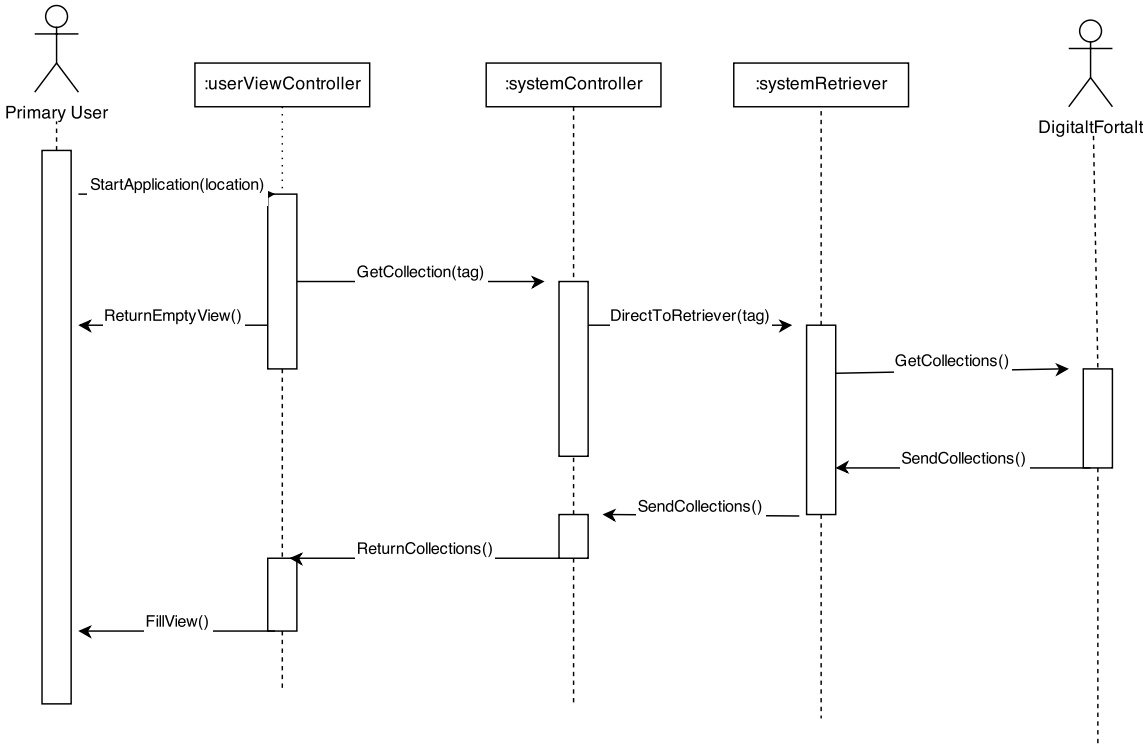

Figure 9 – Get Collections

6 Testing

System testing or software testing, falls into something that is called “Black-box testing”. This is a method of software testing, that investigates the functionality of the application. Eg. what it does, it is simply described as this: It will not require to know how to code, or need any sufficient level of skill to programming when an system test is about to go down. It will neither interfere with it’s internal structure or workings.

6.1 Testing Procedure

When you are about to conduct a test, you find a test-person. Then you tell them what the software is supposed to do. And give them the Test cases[7.2], and explain to them it is very important to think loud so we get the most out of the testing.

6.2 Test Cases

Test cases are built around the specifications and requirements of the application. What the application is supposed to do.

Get Places	
ID	T-F1
Requirements	SF-1
Feature	Places are shown on the map
Preconditions	1. Flickr is up 2. The Flickr group contains photos with locations 3. Application is installed on device 4. Device is connected to the internet
Test Description	1. Open the application 2. Wait for 30 seconds 3. Click on a pinpoint 4. Zoom out to a world view
Expected result	The map should show some clickable pinpoints. When clicked the pinpoints should open a little box containing a thumbnail picture and small text provided by the Flickr Stedr group. When zoomed out new places should be loaded according to what the user see.
Pass/Fail criteria	The test is considered a pass if the expected result happens. The last step that need to be passed is that the place at Grenada is shown. If there are any incosistencies with the expected result, the test should be considered a fail.
Severity	High

Open menu	
ID	T-F2
Requirements	SF-2
Feature	Drawer menu with options is opened.
Preconditions	1,2,3,4
Test Description	1. Click on the menu button 2. Click on all of the icons in the menu 3. Click on the menu again
Expected result	When the menu button is pressed, a drawer menu should open. All of the icons in the drawer menu is also buttons and when clicked again, the menu button should close the drawer menu.
Pass/Fail criteria	The test is considered a pass if the menu button opens and closes a drawer menu. Also, all of the icons should
Severity	High

Views	
ID	T-F3
Requirements	SF-6
Feature	All the views are accessible
Preconditions	1,2,3,4 T-F1 Get places
Test Description	1. Click on a pinpoint 2. Click on the small window that appears 3. Click on one of the buttons *Images, Sound, Story* 4. Dependent on the prevoius step, click on the buttons not yet pushed 5. Click the menu button 6. Click home
Expected result	Which view that is selected is shown to the user by being in a different color than the two other buttons. If the button for the selected view is touched, nothing should happen. For every button representing a non-selected view, the user should be taken to the view as indicated by the button text.
Pass/Fail criteria	The test is passed if the button: Image - Takes you to the image view Story - Takes you to the story view Sound - Takes you to the sound view. The selected view has a unclickable button in a different color representing the selected view. Considered a fail if there are any inconsistencies with the criterias above.
Severity	High

Load Content	
ID	T-F4
Requirements	SF-7
Feature	Content is loaded for the places
Preconditions	T-F3 Views
Test Description	1. Click on a pinpoint(not Camera Obscura) 2. Click on the description 3. Go through the views as in T-F3 3a Click on all of the titles on the story 3b Click on two random images 3c Click on a sound
Expected result	The places should be loaded with relevant and accessible content from all of the content providers.. If some content-types aren't provided for the specific place, the content type should be loaded but indicate that it is empty.
Pass/Fail criteria	The test is considered a pass if the expected result happens. If there are any inconsistencies with the expected result, the test should be considered a fail.
Severity	High

Collection view	
ID	T-F5
Requirements	SF-8
Feature	Show a view with the stories related to a collection
Preconditions	T-F2 Open menu T-F4 Load Content 5 It exist a collection
Test Description	1. Press the menu button 2. Press the Collection button 3. Press a collection
Expected result	When the collection button is pressed a new view should open with the list of stories related to the collection.
Pass/Fail criteria	The test is considered a pass if it is possible to open the menu and access a collection with a list of stories.
Severity	Medium

Collection map view	
ID	T-F6
Requirements	SF-8
Feature	Show places related to a collection as pinpoints in a map
Preconditions	T-F2 Collection View
Test Description	1. Press the menu button 2. Press the Collection button 3. Press a collection 4. Press the *show on map*-button
Expected result	When the "show on map"-button is clicked, a map view should open with related places showed as pinpoints. Pinpoints not related to the collection should not be placed on the map.
Pass/Fail criteria	The test is considered a pass if all places related to a collection is exclusively shown in a map view.
Severity	Medium

Gallery	
ID	T-F7
Requirements	
Feature	Gallery function
Preconditions	T-F4 Load Content The application is in aplcae with a story where there are multiple images to the story.
Test Description	1. Press the story title 2. If there are more pictures related to a story, press the arrows
Expected result	When accessing stories with multiple pictures as content, arrows indicating the possibility to go through picture files should appear. When pressed new images should replace the old picture.
Pass/Fail criteria	The test is considered a pass if the expected result happens. If there are any inconsistencies with the expected result, the test should be considered a fail.
Severity	Low

Upload Content	
ID	T-F8
Requirements	SF-9
Feature	Content can be uploaded to Instagram, Twitter and SoundCloud
Preconditions	T-F4 Load Content 6 Successfully connected to the content (not story provider) providers
Test Description	1. Click on a pinpoint 2. Click on the description 3. Go through the views as in T-F3 4. Upload textual content to Twitter 5. Upload picture to Instagram 6. Upload sound to SoundCloud
Expected result	The places should be loaded with relevant and accessible content from all of the content providers.. If some content-types aren't provided for the specific place, the content type should be loaded but indicate that it is empty.
Pass/Fail criteria	The test is considered a pass if the expected result happens. If there are any inconsistencies with the expected result, the test should be considered a fail.
Severity	Medium

6.3 Test Execution

References

7 Documents

Problem	Description	Likelihood (1-9)	Impact (1-9)	Importance (Likelihood * Impact)	Preventive action	Remedial Action
Communication Loss	Group members doesn't communicate with each other. Group don't establish good communication with the customer and supervisor	3	7	21	Actively establish communication and reach out to the parties.	Talk with the group about the communication, and try to get a good grip of what is failing. Establish communication media, so the group can talk with eachother.
Change requests	Change requests that does not meet the requirements of the product	3	7	21	Well defined requirements spesification, implementing it iterative.	Talk with the customer and ask what he thinks about the request changes.
Technical difficulties	Some problems may turn up to be very hard to solve. This can in turn lead to delays and frustration. And may sometimes be very time consuming.	5	4	20	Regulary have technical discussions with the group, that way the hard problems can be handled by the group as a whole.	If the problem is to hard, try to get help from other groups. Also evaluate if the problem can be handled differentely.

Work-station are noisy	The workstation is filled with people who make alot of sound, so the developers team can't concentrate to the fullest.	5	4	20	Can preorder room, so we get our own workstation to work on.	Preorder room, and move the whole developers team there. If the noise is that bad.
Fail-ing to do planned work	Members of the group fails to do schedueld work due to falling behind in subjects not related to the project or other things.	9	2	18	Good scheduling habits. Sit down every week and see what's planned to do in the project the following week. Coordinate against what you have to do in other subject.	Make up for lost work during weekends or other available time slots
In-suf-fi-cient prod-uct	Devolping a product that does not meet the requirements of the costumer	2	9	18	Good communication with the costumer, in sort of agile devolpment such as Scrum	
API change	The general API has to be changed, because it lack functionality.	2	9	18	Sufficient research about API before implementing it into the project.	Either drop the functionality that is missing, or start developing with the new API.

Different app views	Customer and developers have different views of the apps purpose and funtions	3	6	18	Have regular meetings, inform and discuss all changes to project scope, goals and features.	Discuss with customer and find middle ground.
Scope	The amount of features requested are beyond what the development team can deliver in time	6	3	18	Be specific with the customer how much time we have, and explain deeply how much time it takes to develop a single feature	Discuss what are the nessasery features that must be in the product, and flush out what is the least nessasery.
Lack of competence	Don't have enough competence about the given software we are suppose to use during the project.	8	2	16	Meet every day, do workgroups together and learn by failing.	Talk with other members of the group, and hear if they have the competence. This will prevent hours of searching, when you can listen what the other members have to say. And direct you on the right path for the competence you need.

Install of stedr	Not all group members can install the app on their own device. The purpose is to evaluate stedr.	5	3	15	Install it while you have a meeting, so all the group members can atleast watch the app on another device.	Go together 2 and 2, and watch the app on a phone whose able to install the app.
Missing deadlines	Some work may take longer time than expected, this this may cause delays later on in the project.	3	5	15	Have a steady and diciplined workflow and plan ahead. Overestimate work rather than underestimate.	All members meet and plan what is to be done, and do it at once. So we can deliver as soon as possible.
Customer turnover disruption	A key contact in SINTEF leaves the company, putting the project in a unclear state	2	7	14	Good communication. Multiple contacts with knowledge of the project	Quickly contact the customer and discuss how to proceed and how it's affected
Sickness	Group members or other crucial personell gets sick	4	3	12	Have regular updates about the progress of the work being done, and don't make important task rely completely on one person without a backup plan. Don't freeze and drink a lot of tea.	Talk to the person about the individual tasks, how much he can handle, and distribute the work the member can't.

Group members falling out.	Members doesn't show for meetings, or goes of the grid without notice.	2	6	12	Good communication and agree on a schedule that suits everyone.	Take action at once, and make inquires to why the member didn't show.
Uneven workload	Uneven distribution of workload	6	2	12	Keep updated on the tasks given and work put in, and distribute work	Make the member or members direct task. So it's easy for the member or members to do so.
Conflict over changes	Group members not in agreement over supposed changes in group management, work, responibility etc.	3	4	12	Have an open dialog.	Discuss in group and decide as a democracy.
Late for meeting	Members of the group are late for meetings with group/customer and supervisor	6	2	12	Good communication and agree on a schedule that suits everyone.	Take action at once, and make inquires to why the member came late.
Documents customer/-supervisor meeting	You lack the sufficeint documents for the meeting with the customer. For presentation on how you want the app to be, mockups and reports about fieldwork etc.	2	6	12	Have the documents stored in the cloud so you can acces it where ever you go. With your respective smartphone/tablet and pc's,	Discuss what you remember and try to make the best out of the meeting, as possible.
Equipment failure	Computers and other dependable devices malfunctions.	4	2	12	Keep documents and code in the cloud so you can work from another device if your primary device malfunction.	Get replacement as soon as possible.
Document sharing failed	Authorization of documents sharing is not complete, people don't have access to the groups documents.	2	4	8	Give all the authorization they need for the documents to be shared. So all can view, edit and share documents.	Find out where the problem lies, so everyone can get authorization for the given documents and folders.

Lack of soft-ware	nessasery for the develo-ment progress	1	3	3	Talk about what software is required for the development of the product. Ask the customer for this software.	Ask the customer immediately for the required software, so the development progress don't have any major delays.

8 Attachments

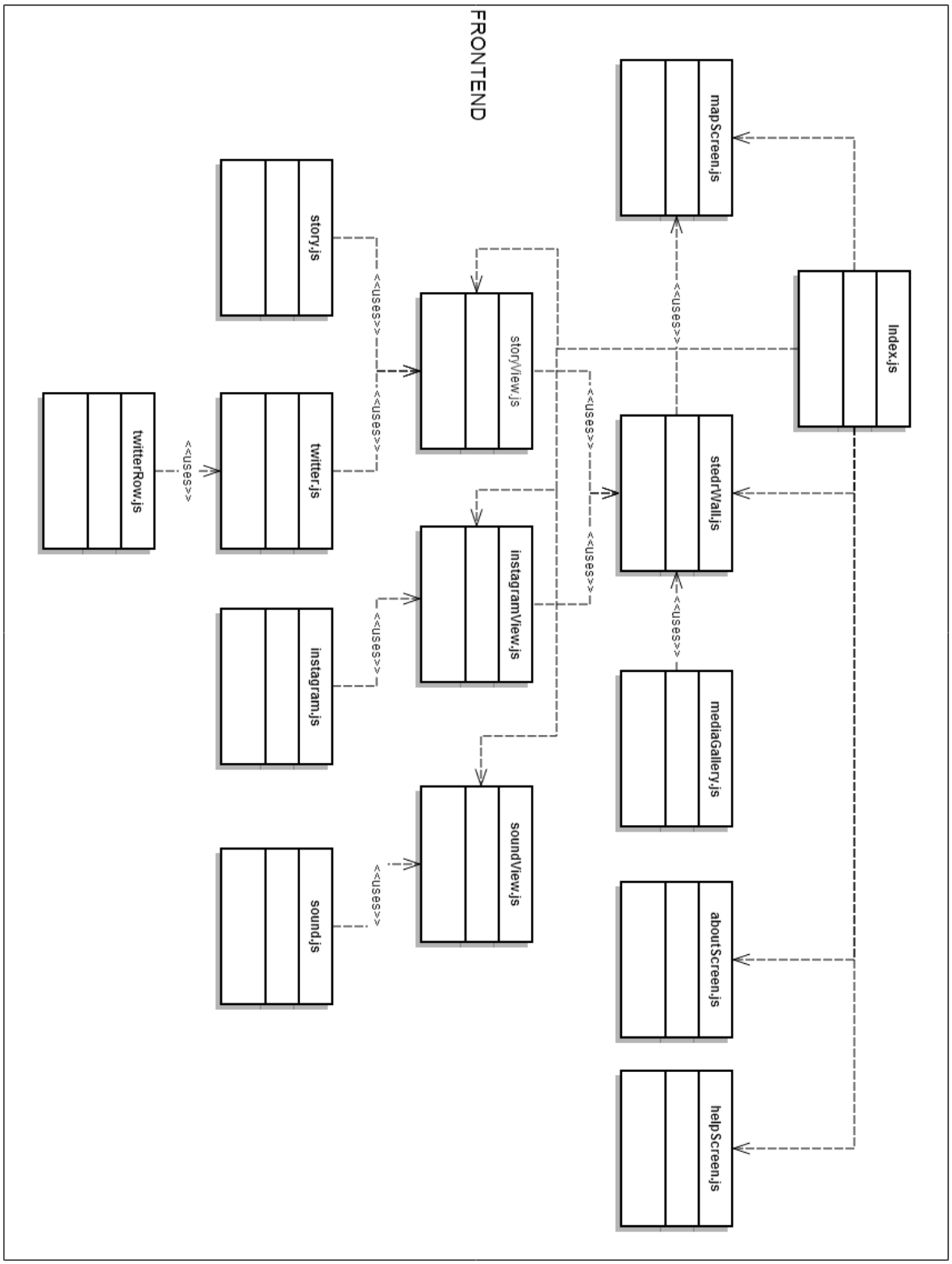

Figure 10 – Class diagram for frontend

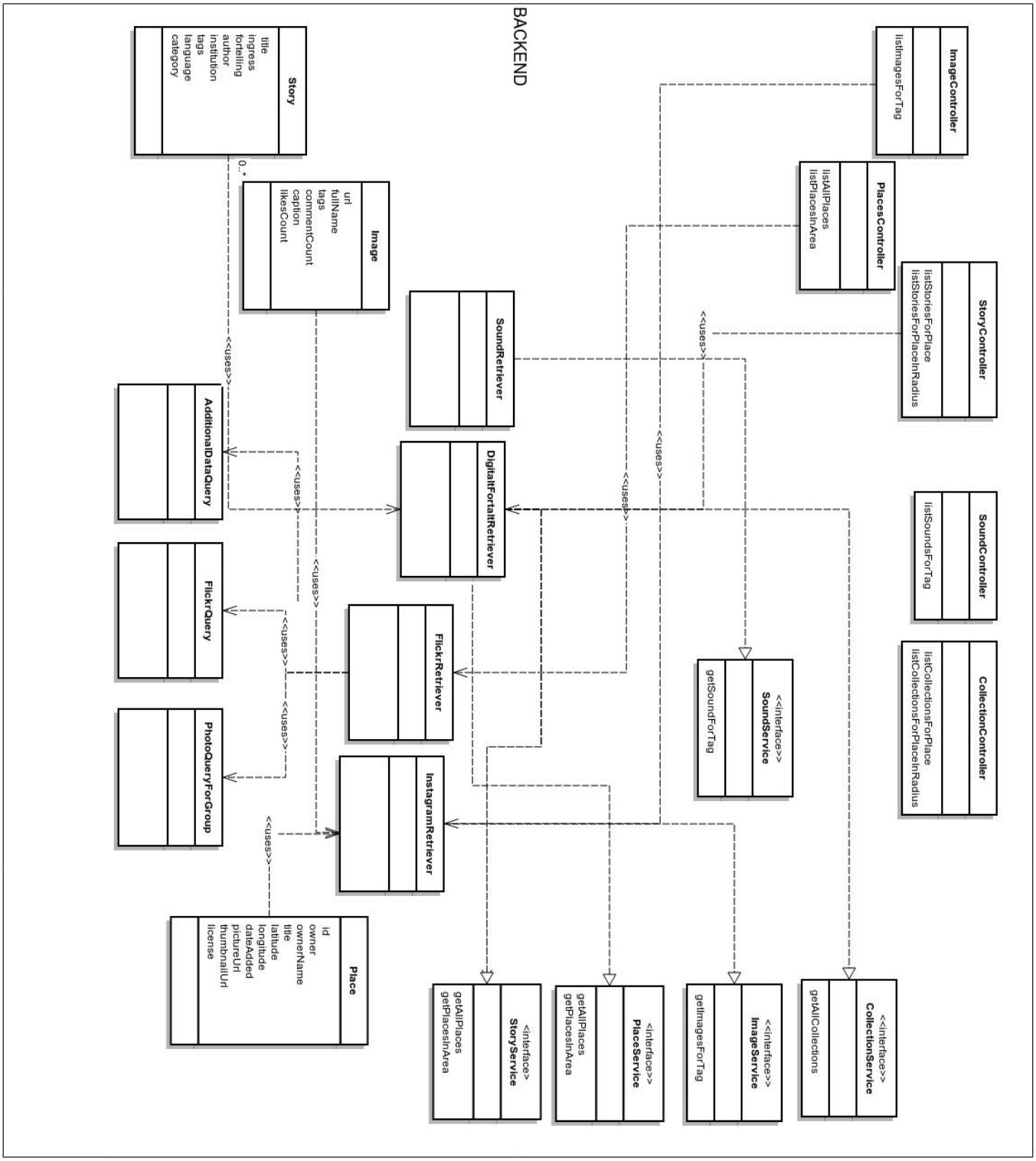

Figure 11 – Class diagram for frontend