HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

COMPUTER SCIENCE

Sieve

Sieve
Data Privacy Made Simple

Term Project

Department of Computer Science
University of Applied Science Rapperswil

Spring Term 2020

Author(s):	Michael Güntensperger Lorenzo Schumacher Raphael Rechsteiner
Advisor:	Prof. Frank Koch
Project Partner:	Sieve Pte. Ltd., Singapore

Abstract

If a mobile device user is interested in an app's privacy policy, he would have to search for it online, just to find out, that it is a very long and pretty much unreadable document. Therefore, a lot of people do not get the information they were looking for, even though it would be very important to know, e.g. how a company handles the user's data. The data privacy company Sieve wants to empower its users by teaching them about privacy policy in a way that is easy to understand. To achieve this, Sieve wants to present the app policies in a condensed form and rate the different apps based on how much data they collect. This should allow users to make an informed decision on what apps to install on their phone.

For the client Sieve, a React Native MobileApp, that shows data privacy information of apps was developed. To help a user choose what to install on his mobile phone, the Sieve App can sort apps into categories and shows a unified rating for each of them. Furthermore, the Sieve App also allows its users to stay up to date on privacy policy changes of their favourite apps and get general news on app privacy. All of the information inside of the Sieve app is entered manually by Sieve using a React web application called AdminWebsite. Between these two components lies a Node.js WebServer that manages the data and is reachable through a REST API. Unstructured files like images are saved in an object storage and the structured data is on a MongoDB Data Store. These two storages as well as the AdminWebsite and the WebServer are hosted in the cloud.

Our prototype contains the three core components MobileApp, AdminWebsite and WebServer, that Sieve needs to inform their clients on app privacy. The three parts of the application each offer basic functionality and can easily be expanded upon, because of the use of well-known web technologies. The modular design and the use of the flexible REST API allows to add different data producers and consumers to the Sieve landscape in the future. A possible expansion that Sieve is considering would be a Natural Language Processing unit, that automatically creates key points from a verbose privacy policy. Such an addition could easily be integrated into the existing system by using the REST API.

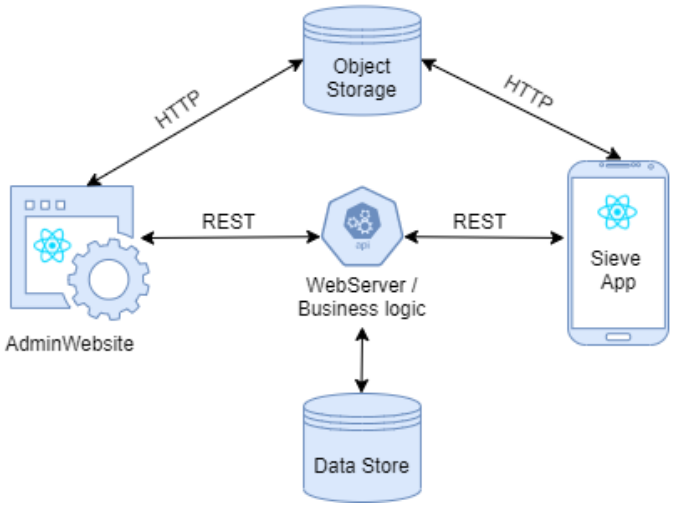


Figure 1 - Architecture diagram - overview

Management Summary

Initial situation

Only few people read the privacy policy, most scroll through it, activate the check mark and accept it. This is understandable, as they are usually written in a very complex way and stretch over several pages. Most people don't know or don't care what data they send to the companies by accepting the privacy policy. This, for example, includes: what the companies do with the data; what happens to it if one deletes the account; how can one have his recorded data deleted. What would be the alternative of just accepting the privacy rules? Not to use the app?
To address this problem, it is important to make the privacy policy as understandable as possible for everyone, to make it easily accessible and to condense it into the most important key points. This way, the user can be sensitized and the privacy policy is brought closer to him step by step.
This is now to be made possible for the user via the Sieve MobileApp.
He can easily, quickly and intuitively find, read and compare the data protection guidelines for the interested applications. With the help of an automatically generated rating, he can choose the right application to use.
Our client Sieve's intention is to automatically process privacy policies using natural language processing in the future. But they have not started with that part of the project yet and want to enter most of the data manually for now.

Procedures, technologies

To support our client's future expansion plans, we wanted to create a flexible product. We decided to split our project into three independent components that communicate with each other.
To have the components as uniform as possible and allowing them to run on different platforms, we decided to use web technologies. For the MobileApp we use React Native with the Expo Toolchain and for the AdminWebsite we use the JavaScript software library React. The WebServer is a Node.js application. This way we can use JavaScript for every application.
The application has a modular structure and can easily be extended with additional functionality. This enables the implementation of optional features in the future and thus to continuously extend and improve the application with new useful features.
For the design, we wanted to realize our client's vision, while also bringing in our own ideas. We tried to achieve this with the extensive use of mockups in the first phase of the project. Our client Sieve gave us a lot of feedback and new ideas during the project. To handle these inputs our development had to be agile.

Figure 2 - Related technologies

Results

A cross-platform MobileApp prototype was developed that meets the customer's desired business cases. A website for the administration to manipulate the data and a WebServer that serves as an interface between the database, the MobileApp and the AdminWebsite manages the data, were also created.
Due to the choice of a platform neutral programming language, the product can be operated with an Android or iOS device. This makes the MobileApp accessible to a wider audience. The app itself was not published to the iOS or Android store.
The WebServer and the AdminWebsite as well as other infrastructure are hosted in the cloud for easy scalability and extensibility.

Outlook

All functions requested by the customer, that we deemed feasible, were implemented. This makes it possible to use the application meaningfully. The customer had many new ideas along the way, which led to many optional features, that can be implemented. These were continuously documented.
The flexible and expandable design of the architecture makes it possible to easily add new features or components later on. An example feature could be a Privacy Tips screen, that would enable the user to receive tips on how to better protect her/his privacy. Or one could add a completely new component like a natural language processing unit to automatically add new privacy policies through the WebServer's REST-API.

Content

1. Introduction and overview 7
1.1 Description of the initial situation 7
1.2 Preliminary work 7
1.3 Vision 8
2. Requirements analysis 9
2.1 User Stories 9
2.2 Functional Requirements 10
2.2.1 Overview 10
2.2.2 Use Cases 12
2.2.3 Optional Features 14
2.3 Non-Functional Requirements 15
2.3.1 Preliminary Remarks 15
2.3.2 Measurable Requirements 15
2.3.3 General Requirements 16
2.4 Domain analysis 18
2.4.1 Domain model 18
2.4.2 System sequence diagrams 20
2.5 Mockups 21
2.5.1 MobileApp Screens 21
2.5.2 MobileApp Screenflow 25
2.5.3 AdminWebsite Screens 26
3. Software Architecture 29
3.1 Design decisions 29
3.2 System overview 30
3.3 Architectural goals & restrictions 31
3.3.1 Goals 31
3.3.2 Restrictions 32
3.4 Tools 32
3.5 API Description 33
3.6 Deployment 34
3.6.1 Automated deployment 35
3.7 Data Storage 36
3.8 Sizes and Performance 37
4. Quality assurance and testing 38
4.1 Performance tests 38
4.2 Metric analysis 39

4.3 Unit Tests 41
4.3.1 Challenges 41
4.3.2 Postman 41
4.4 Integrationtests 42
4.4.1 Mocking 42
4.4.2 Issues 42
4.4.3 Conclusion 42
4.5 Usability tests 43
4.5.1 General conditions 43
4.5.2 Test cases 43
4.5.3 Evaluation 44
5. Results and Outlook 46
5.1 Achieved Goals 46
5.2 Outlook and Extension 48
5.2.1 Information from App Store 48
5.2.2 Natural Language Processing 49

Appendix 50
A. Glossary 50
B. References 52
C. Table of Figures 53
D. Table List 54
E. Project Management ..... -
F. Personal Report ..... -
G. Project Plan 55
H. Final Application 56
I. Meeting protocols ..... -
J. Optional Features 62
K. Developer Documentation 68
L. API Documentation 69
M. Feasibility Study Traffic Analysis 70
N. React Native UI Libraries Comparision 71
O. Test Protocols ..... -
P. Sample Privacy Policy 72
Q. Categories 82
R. Administrative Appendixes ..... -

1. Introduction and overview

1.1 Description of the initial situation

“Companies are abusing the vagueness of data privacy regulations to draft ambiguous, verbose and legally abstract data privacy policies.” [1] They will ask to agree to provide your geolocation details for business purposes but then sell it to third-party vendors.
Sieve’s goal is to empower app users to provide their informed consent, by simplifying user terms into simple key points. This would train users to be comfortable with reading and understanding privacy policies.
Sometimes a user wants to quickly install an app of a certain type and does not want to read through the policies of all the possible candidates, even in a condensed form. For this case, the apps privacy policies should be rated and presented in a form, where similar apps can easily be compared.

Sieve has tasked us with the creation of a prototype mobile application and the supporting infrastructure, that can fulfil their business case, in the context of our term project.
For the successful processing of this task, firstly, a requirements analysis should be carried out and discussed with the customer. The goal is to understand Sieve’s ideas as precisely as possible and guide them towards something that we can develop. The client wants to manage the data and information shown in the Sieve app either manually or create and input it automatically using Natural Language Processing. To support this, a flexible infrastructure has to be developed around the mobile app.

1.2 Preliminary work

There was no preliminary software development work on this project.
The client Sieve has been working in this area for quite some time and saw a need to build an app, so regular people have the possibility to inform themselves on privacy policies in a simple way.

Analysis of existing products
An analysis of already existing products was performed by Sieve.
As shown in the graph, four different existing products were compared. Two of them can be user-tailored, but are expensive. The other two products are cost effective, but very generic.

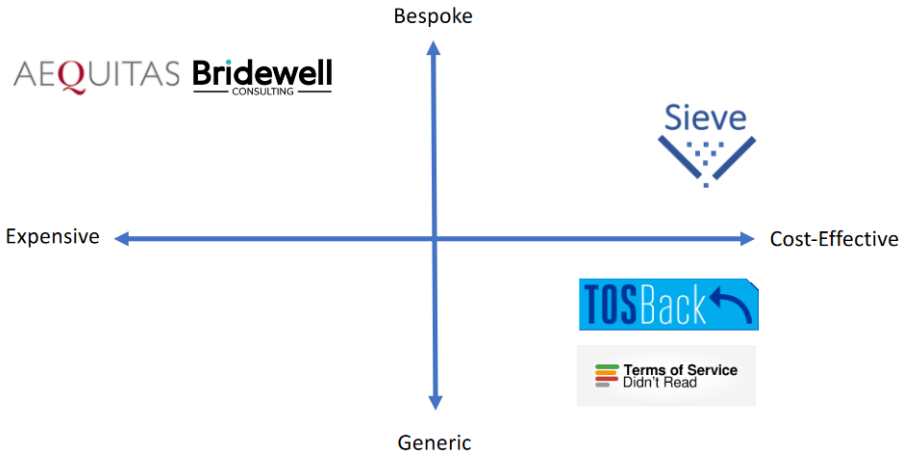


Figure 3 - Analysis of existing products [1]

The competing services are either very expensive or unpopular. This is where Sieve wants to come in with a MobileApp that is free and easy to understand.

App Mockup
The core screens of the app were designed to give us, the developers, a better understanding of what we will be building.
These screens were then adapted by us, to fit the project scope and improve the design.

Figure 4 - First screens of the app [1]

1.3 Vision

The vision is to provide every user of the application with a simple, understandable and intuitive way to familiarize themselves with data privacy policy. This way, it is possible to get a step by step introduction to the topic. The users learn which data the companies store about them, what they do with it and what happens to the data when the app is removed. Thus, the users are sensitized to the topic of data protection. In order to achieve this, the data protection guidelines are broken down into important key points. By rating the apps based on the collected data, the user can easily see, if an application collects a lot of private data or if it is safe from a privacy point of view. To reach the widest possible audience, the mobile application should run on as many devices as possible, which is why it should be available for iOS and Android.

2. Requirements analysis

The requirements analysis is an important step before starting with the software architecture. It is important that the customer's idea and wishes are understood and that all parties involved have the same understanding of the requirements.
Based on the mission statement and the first meetings with the customer, we conducted the requirements analysis. This resulted in the User Stories, with the help of which we were able to derive the functional requirements, such as the Use Cases and the first Optional Features, as well as the non-functional requirements. A great help, also because of the different mother tongues, were the mockups. With the help of these mockups we could visually record and convey the requirements we had developed. This ensured that there was a uniform understanding of the functions. To visualize the interaction of the individual screens and components, the domain model including a sequence diagram was created.

2.1 User Stories

User stories describe features and user interactions in an informal way. The goal is to understand the users and their motivation in using the different features [2]. We imagined two different personas for our application**:** The administrator works for Sieve and manages the application. The user is interested in data privacy and has installed the Sieve mobile app on his phone.

US01 As an administrator I would like to be able to view, create, edit and delete new apps including the associated privacy policy through a website.

US02 As an administrator I would like to be able to view, create, edit and delete new news through a website.

US03 As a user, I would like to know more about the privacy policies of the apps I am interested in.

US04 As a user I would like to see important news about privacy policies and/or data protection any time.

US05 As a user, I would like to be able to add apps I am interested in, to my favourites in order to track the privacy rating and compare it with other apps.

US07 As a user, I would like to be able to remove apps that I am no longer interested in from my Dashboard (favourites).

US08 As a user, I would like to have the apps divided into categories to see similar apps that have a better privacy policy rating.

US09 As a user, I want the privacy policy to be as easy to understand, clearly structured and as intuitive as possible in order to find the information I am looking for as quick as possible.

US10 Optional: As an administrator I would like to be able to view, create, edit or delete categories.

2.2 Functional Requirements

2.2.1 Overview

An overview of all use cases and the optional features that have been developed, including their prioritization. The goal is to implement all the use cases with priority one to three. Priority one is a must goal, priorities two and three are desirable.

Use Case	Description	Prioritization [1 – 3]
UC1	CRUD Apps interested in	1
UC2	CRUD account	1
UC3	CRUD apps	1
UC4	CRUD privacy policy	1
UC5	View privacy policy information of app	1
UC6	View privacy rating of apps	1
UC7	Find similar apps	2
UC8	CRUD privacy news	2
UC9	View privacy news	2
UC10	View privacy rating of installed apps	3
UC11	Open communication with privacy officer	3

Table 1 - Use Case overview

The optional features are categorized by priorities of one to three as well. These are desired goals that will be implemented if time permits.

Optional Features	Description	Prioritization [1 – 3]
OF1	Implement the App GDPR compliant	3
OF2	Login screen – Login with Facebook and Google	2
OF3	Login screen – Sign Up with first name and surname	1
OF4	Category screen – Search	2
OF5	Category screen – Similar apps	1
OF6	Category screen – Navigate with side wiping	3
OF7	Category screen – Images at Apps	2
OF8	Category screen – Apps with rating	2
OF9	Category screen – Different font size	1
OF10	Privacy Tipps screen – Create screen	1
OF11	News screen – Policy updates messages, with app icon	2
OF12	News screen – Push messages for new entries	3
OF13	Settings screen – Account data changeable	3
OF14	Settings screen – Sorting adjustable	3
OF15	Settings screen – Change password	1
OF16	Detail screen – Make chat with Data Privacy Officer look like real chat	3
OF17	Detail screen – Compare apps with side wiping	3
OF18	Detail screen – Privacy Policy key points highlighted	3
OF19	MobileApp - Navigation Order	1
OF20	MobileApp – Landscape Mode	1
OF21	Publish the MobileApp to the Google App Store	3
OF22	Publish the MobileApp in the Apple App Store	3

OF23	AdminWebsite – Import of data with CSV	3
OF24	AdminWebsite – Admin can add or change fields independently	3
OF25	AdminWebsite – Category View	1
OF26	AdminWebsite – Categories CRUD	2
OF27	AdminWebsite – Automatically download Apps Icon from App Store	3
OF28	AdminWebsite – Automatically download Apps description from App Store	3
OF29	AdminWebsite – Data Type Collected points (1 or 5) Changeable	2
OF30	AdminWebsite – Change data structure	3
OF31	AdminWebsite – Filter apps by category	2
OF32	Transfer all through with https	2
OF33	Digital Ocean billing-API to show cost	1
OF34	Ordering of tables by row	1
OF35	Split displayed entries into multiple pages (e.g. apps)	3
OF36	Dashboard optimization	1

Table 2 - Optional Features overview

Restrictions

The following restrictions were agreed upon with the customer:

- No natural language processing interface will be developed. The data shown in the app will all be entered by the administrator.
- The reading of OS-information is device and OS-dependent, either it is not possible at all or very time intensive to implement. Especially the analysis of the data traffic of a MobileApp is a very complex matter and is beyond the scope of this project.
  For more details see in the Appendix under Feasibility study traffic analysis.

Dependencies

The following dependencies exist:

- Quality of the data entered by the admin.
- Version of the device's operating system (see Non-functional Requirements ST3).

2.2.2 Use Cases

Actors & Stakeholder

Actors are the registered user and the administrator. Because it is only possible to use the app with an account, there is not unregistered user.

Use Case Diagram

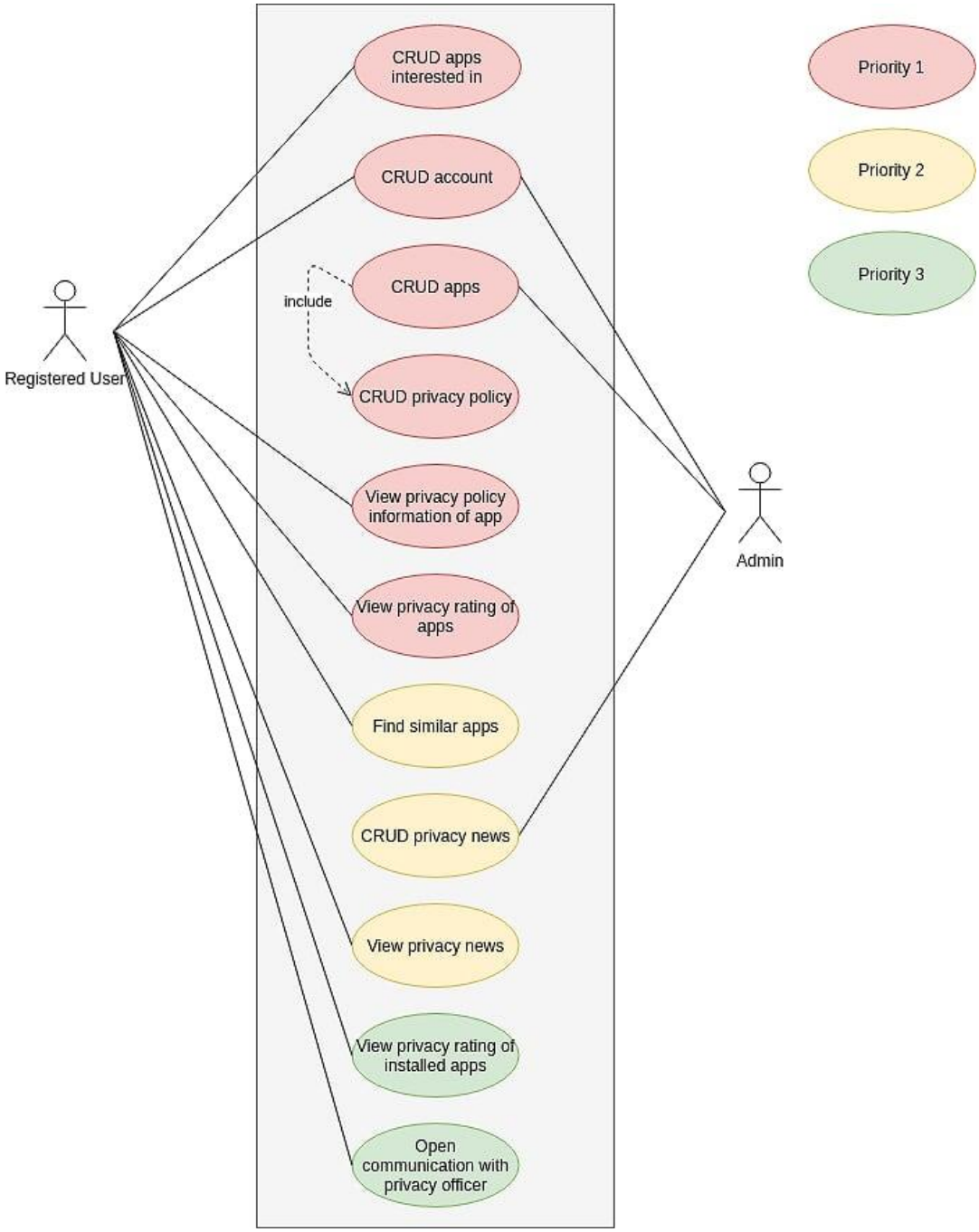


Figure 5 - Use Case Diagram

Use Case Description

Because of the amount of use cases, optional features and the limited business logic they include, they will only be described in brief.

UC1 - CRUD apps interested in
The user installs for example the BBC app on his phone and would like to know more about the privacy policy of the app. Therefore, he opens the Sieve-App and adds BBC to his favourites.

UC2 - CRUD account
The user installs the Sieve-App and registers himself. He then logs in to use the app. He can change the password in the settings.

UC3 - CRUD apps
There is a new communications app on the market. The admin registers the new app with all its basic information to show on the MobileApp.

UC4 - CRUD privacy policy
The admin analysed the privacy policy of an app and registers it on the AdminWebsite.

UC5 - View privacy policy information of app
The user wants to know about the privacy policy of a certain app (e.g. BBC). He navigates to the app-information and opens the privacy-policy-category.

UC6 - View privacy rating of app
The user wants to know about the privacy policy rating of a certain app (e.g. BBC). He adds the app to his favourites and navigates to the dashboard.

UC7 - Find similar apps
The user doesn't like the privacy policy of a certain app (e.g. BBC). He looks for a better alternative, so he opens the Category screen and search in the same Categorie alternative apps, to check the privacy policy of the apps.

UC8 - CRUD privacy news
Sieve finds some privacy news, which might interest its users. The administrator can insert this news piece into the database and link it to the app it might concern. He can also read, update or delete old news pieces on the database.

UC9 – View privacy news
The user can read privacy news on the Sieve-App. The user sees that there is some activity and the app is being maintained.

UC10 - View privacy rating of installed apps
User has recently installed the Sieve-App and wants to see the privacy rating of all the apps installed on his phone.

UC11 - Open communication with privacy officer
After looking at the privacy policy information of an app, the user has some questions and opens a communication channel to the privacy officer of the app.

2.2.3 Optional Features

Throughout the entire project, the customer continuously had ideas on how the applications could be expanded. We documented all of them with a short description. The result is a list of the possible optional features that would be implemented based on the customer's prioritization if there was time left at the end of the project.
Below is a list of all the features we were able to implement. The full documentation of the optional features can be found in the appendix.

MobileApp

OF3 - Login screen - Sign Up with first name and surname
Add the first- and last name as fields to the MobileApp Signup form. This allows the administrator to send out personalized newsletters.
Affected components: MobileApp, WebServer

OF9 - Category screen – Different font size
Change the font size to a smaller one. React Native has default font sizes that are displayed correctly on iOS and Android devices with different sizes and resolutions. To set a different one, the font sizes would have to be set manually.
Affected components: MobileApp

OF19 – MobileApp – Navigation order
Reorganize the order of the Navigation-points. Switch News <-> Dashboard and show the Dashboard screen when a user logs in.
Affected components: MobileApp

OF20 – MobileApp – Landscape mode
Implement the Landscape mode for all screens of the App. Depending on the size of the device and its resolution, the placement of the tiles would have to be changed as well.
Affected components: MobileApp

AdminWebsite

OF25 - AdminWebsite - Category View
Show all the existing categories on the AdminWebsite. This way the administrator has an overview of all the categories including all the icons. It can be extended to include the CRUD functionality.
Affected components: AdminWebsite

OF34 - Ordering of tables by row
Allow the administrator to order all the entries in tables by row. It will not be possible to order every row like e.g. categories because there sometimes is more than one entry.
Affected components: AdminWebsite, WebServer

OF36 – Dashboard optimization
To help the user better understand the rating, it is supplemented with a text below the screen title. Additionally, the score should be displayed as "Score number/ maximum score number" (e.g. 1/5) instead of a fixed value 0...5.
Affected components: MobileApp

2.3 Non-Functional Requirements

2.3.1 Preliminary Remarks

The non-functional requirements (NFR) for this project apply to the ISO standard 9126 [3]. This is only for the MobileApp and AdminWebsite. Only some important aspects are stressed in this section, the others are left empty.

2.3.2 Measurable Requirements

ST1 In the context of our test challenge, all the users attending the test should be able to find apps on their dashboard and by clicking on them find the relevant information *(suitability test)*

ST2 Finding the data policy info of an app should take less than 20 sec. *(accuracy test)*

ST3 The App should run on Android with versions not older than 9.0 and iOS phones with a version not older than 12.4.5. *(interoperability test)*
The AdminWebsite should run on Firefox, Chrome and Safari. Does not have to be responsive.

ST4 SQL Injection tests on input fields should not show vulnerabilities *(security test)*

ST5 Three of four test users have to explain two predefined, specific functionalities without prior knowledge *(understandability test)*:
- Icons on Dashboard
- Attributes of Privacy Policy Category

ST6 Three of four test users should rate their UI experience (criteria: layout, responsiveness, colour, content) of the MobileApp using a smartphone with a grade of at least 7 of 10 where 10 is the best. *(operability test)*

ST7 100% of our test requests should be logged to ensure that the entries are persisted *(analysability test)*

ST8 The application should be able to handle 100 requests a minute *(operability test)*

ST9 The database should be able to handle up to 1’000 privacy policy entries *(reliability test)*

ST10 The app should not take more than 200ms to load *(efficiency test)*

2.3.3 General Requirements

Functionality

Suitability
The functionality is defined by the team itself. We have agreed on the basic use cases and some optional ones. Changes in scope of the software will be in agreement with each party.

Accuracy
The administrator that inputs the data is responsible for its accuracy. We are at this stage not able to check/verify the entered data.

Interoperability
The service is designed as Website/App which can be accessed with every browser/Android or iOS phone. The REST Endpoints are not for public use. The technologies used are React, React Native and Node.js.
Will be checked with: **ST3**

Compliance
All application data will be stored in compliance with the GDPR at a later stage.

Security
No passwords will be stored in plain text in our databases. Simple input validation will be used in every input field.
Will be checked with: **ST4**

Reliability

Maturity
No error should propagate to the user, but a routine should continue with an alternative flow for the user.
Will be checked with: **ST5**

Fault Tolerance
The admin is responsible for his data input. Usernames/password etc. will be validated.

Recoverability
The admin is responsible for backups and recovery of the database. This should be a basic database procedure.

Usability

Understandability
The interface of the app should be self-explanatory. No feature needs an explanation and it isn't necessary to read a manual.
Will be checked with: **ST2, ST5**

Operability
Its main focus is on operations with the smartphone. It should not be necessary to zoom in or out and people between 14 and 60 should be able to read the text without magnifier.
Will be checked with: **ST1**, **ST6**

Efficiency

Time behaviour
Each action should get a result in less than a second for fast internet connections equivalent to a wired ethernet connection from Switzerland with the server anywhere in Europe.
Will be checked with: **ST2**, **ST6**, **ST10**

Maintainability

Analysability
Each error should be logged and can be seen by the administrator of the system.
Will be checked with: **ST7**

Changeability
Because of the basic scope of the software, it should be kept as extensible as possible for further development steps (API's).

Stability
The stability of the application is guaranteed by the SLA's of Digital Ocean.

Testability
Unit Tests are mandatory. UI Tests will be developed if there is enough time/the basic use cases are covered.

Portability

Adaptability
Apps interested in can be added to favourites and will then be shown on the Dashboard. They are being saved on the server.

Installability
The user has to install the app and register / login to use the service.
The admin can use (Firefox, Chrome and Safari) to access the AdminWebsite.

Replaceability
The database can be exchanged by any other database, only the config file on Node.js has to be changed. Because the server is running in a Docker container, it can be moved to another provider easily.

2.4 Domain analysis

Contains the domain model and the most important sequence diagrams. It provides an overview of all concepts and system relationships. Due to the domain analysis the interaction of the individual components and screens could be visualized. It shows which components are needed and how they work together. With the help of the domain model and the sequence diagram, the database model can be created and the interfaces defined in the architecture phase.

2.4.1 Domain model

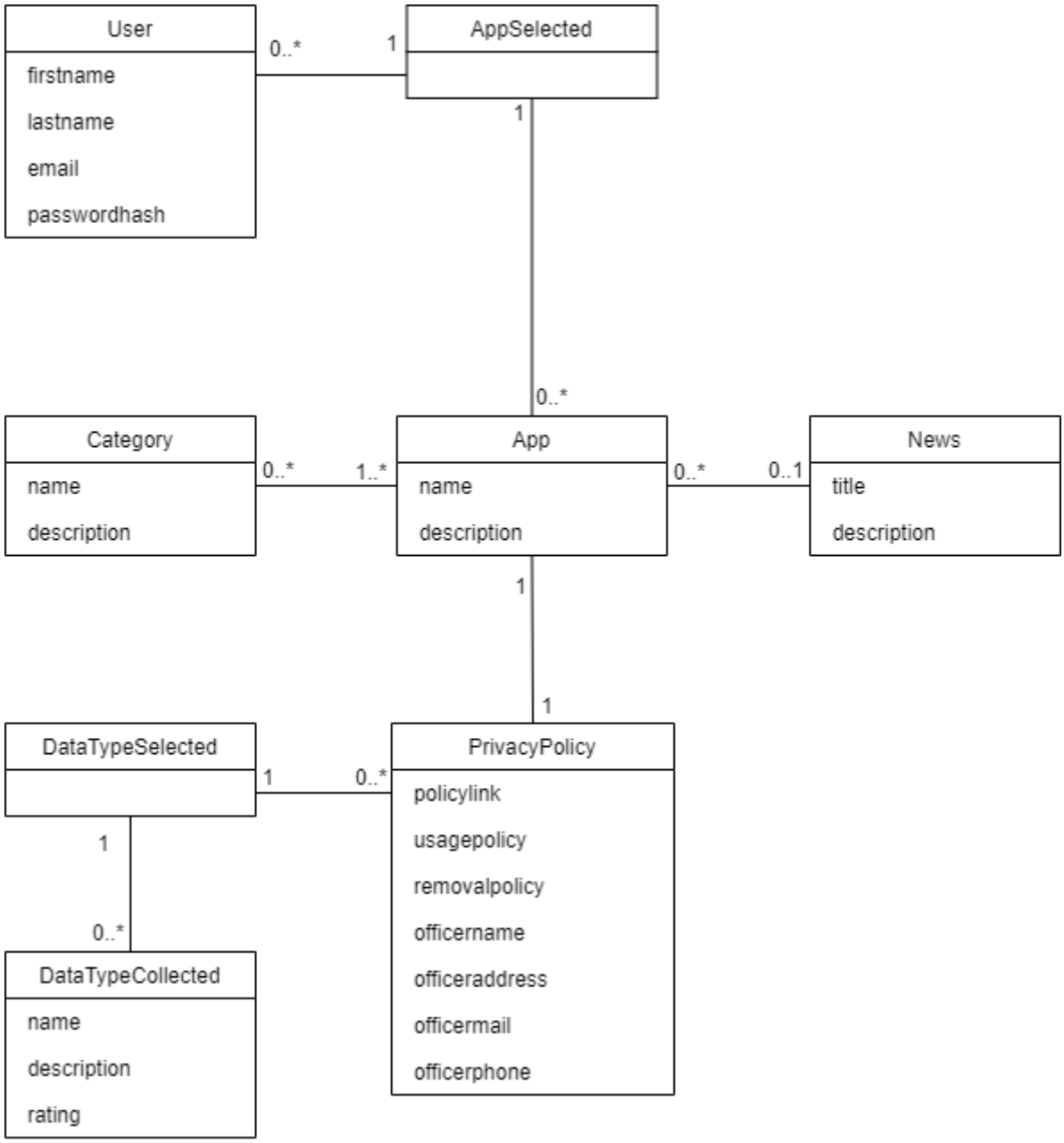


Figure 6 - Domain model

In the following, the most important objects are highlighted and provided with a short description.

Privacy Policy	The Privacy Policy is the heart of the application. A privacy policy is always assigned to exactly one app. All information of the policy is stored here in the form of key points. The administrator can enter and change these on the AdminWebsite.
App	The app is the center of the Domain Model. This is where all data is linked together. A privacy policy is always assigned to exactly one app.
News	The news about privacy policies are presented. There are two types of news. General news, which can be written by the admin himself and static news, if a privacy policy has been modified or added (optional feature).
Category	Categories are used to group similar apps so that the user can find an app faster and find similar apps with a better policy. Multiple apps can be assigned to a category. And an app can appear in more than one category.
Selected Apps	The selected apps are the apps, that a user added to his favorites. These apps are displayed on the dashboard. A selected app can be deselected at any time and disappears from the favorites.
User	In this object the personal data is stored. The attributes mail address and password are used for the login. The user is needed to assign the selected apps to have the possibility to display the dashboard with the favorites.

Table 3 - Domain model description

2.4.2 System sequence diagrams

The following sequence diagram should give a better understanding of how the components are working together to show all the necessary information to the user or perform an action on the MobileApp. This was done by the example for S01.

S01 - CRUD apps interested in (UC1)

The user for example installs the BBC app on his phone and would like to know more about its privacy policy. Therefore, he opens the Sieve-App and adds BBC to his favourites.

Preconditions are that the user already has an account and is logged in. He is currently on the Dashboard screen of the application

The following procedure is illustrated:
1. Change to Categories screen
2. Click on the category News
3. Select BBC
4. Add it to the favourites by clicking on the star icon

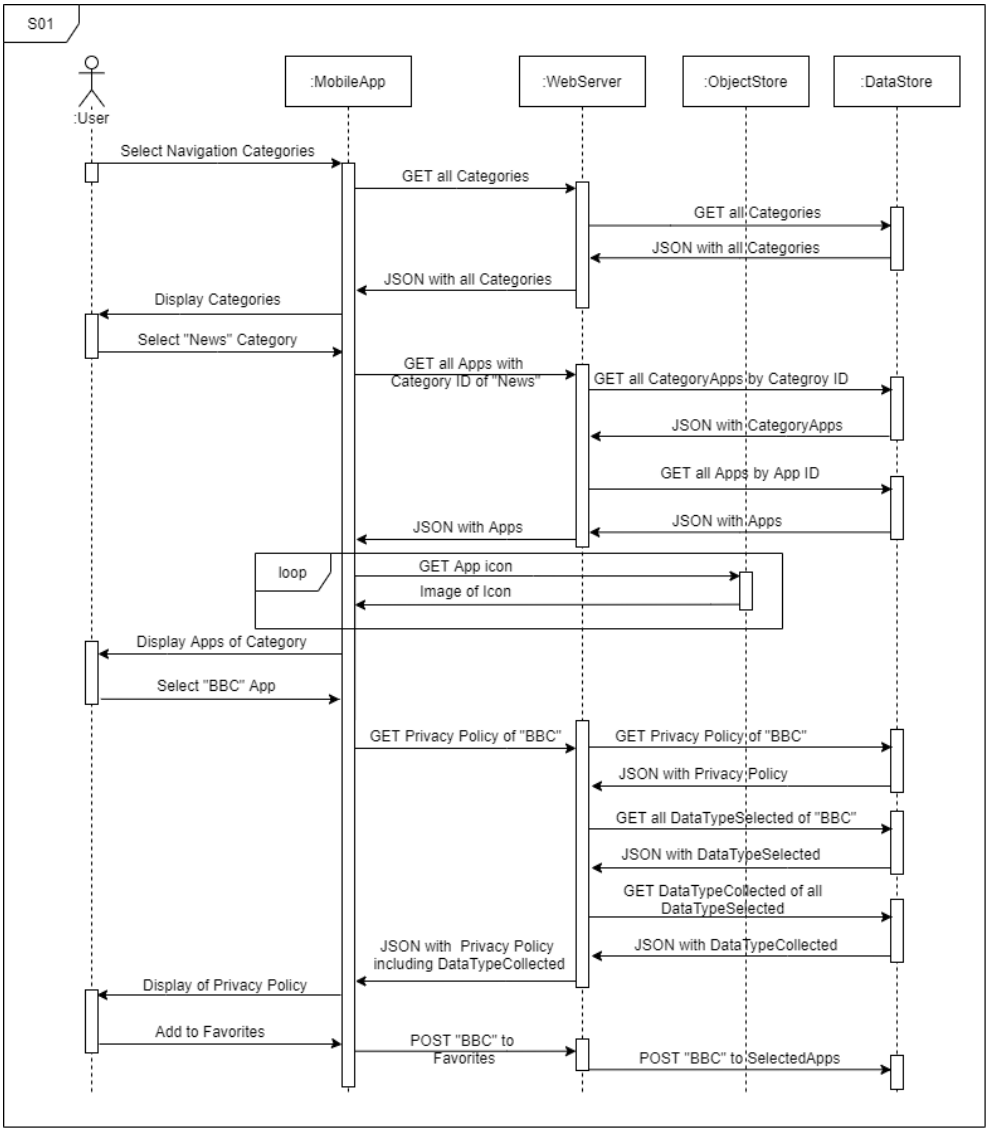


Figure 7 - System seuence diagram S01

2.5 Mockups

The style of the final product may deviate from the mockup shown below. It is used to show the functionality and help us to set clear requirements. All the optional features are removed from the screenflow. The mockup was created with Adobe XD for the MobileApp[1] and the AdminWebsite[2]. Because of the changing requirements and design of the MobileApp, different mockups were created. We show the latest version and compare it to the actual implementation, where it is interesting to explain our design process.

2.5.1 MobileApp Screens

Login / Signup

The user can change between the Login and the Sign Up screen freely. After signing up, the user still has to log in with the credentials he created. The mockup does not show any third party login possibilities (Google, Facebook , …), because we did not see it as a core part of the functionality. It is only possible to use the app after logging in. In the actual app, first- and last name are required as well to sign up.

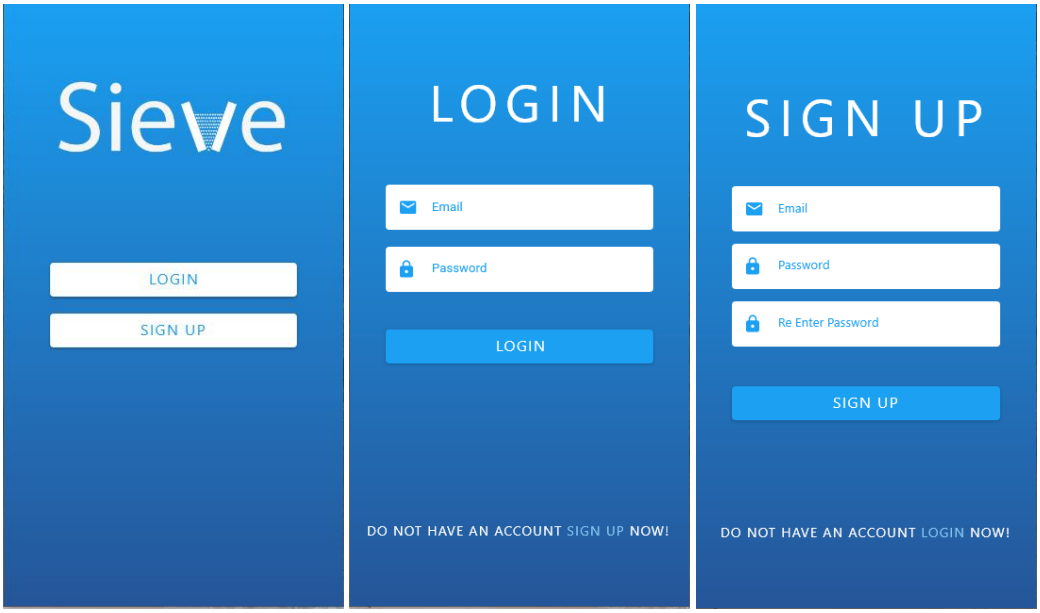


Figure 8 - Mockup Login / Signup

[1] https://xd.adobe.com/view/53098e4c-7617-444c-7861-ebadfcdcf104-2538/
[2] https://xd.adobe.com/view/52b8eb05-ca1b-4e22-5191-612a826aa0c5-2489/

News-Screen

This screen displays news articles regarding privacy policy. The maintainer at Sieve has to write and manage these news stories manually using the AdminWebsite. By having fresh news articles in this section, the user can see that the app is still being maintained and is more likely to trust Sieve's expertise. It is also an additional customer interaction point for the client. From mockup to implementation the appearance changes of this screen where minimal.

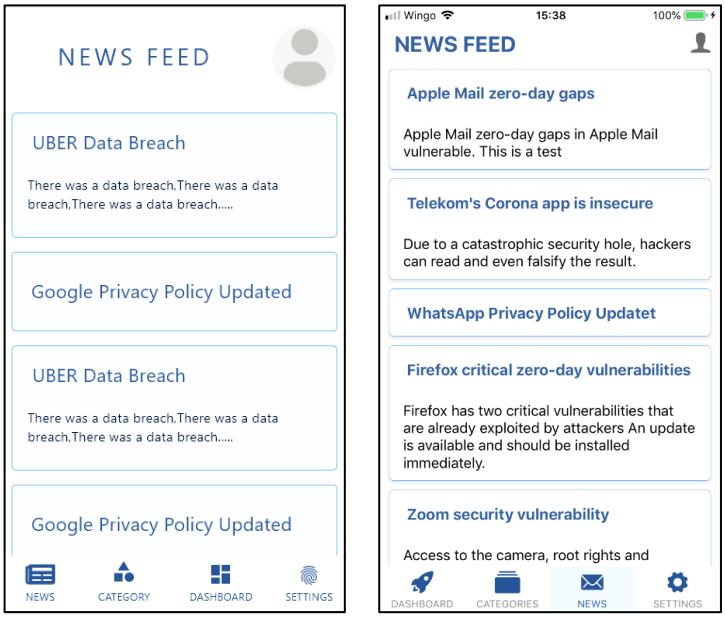


Figure 9 - Mockup / MobileApp News-Screen

Categories Screen

All of the apps registered by Sieve are categorized and this screen shows these categories as tiles. When clicking on a tile, the user can see all the apps belonging to that category. This is a simple tool to facilitate choosing the right app for a certain application. The categories were predefined by the customer and the icons are statically stored on the object storage. It is possible to change these categories or add new ones, but it has to be done manually.

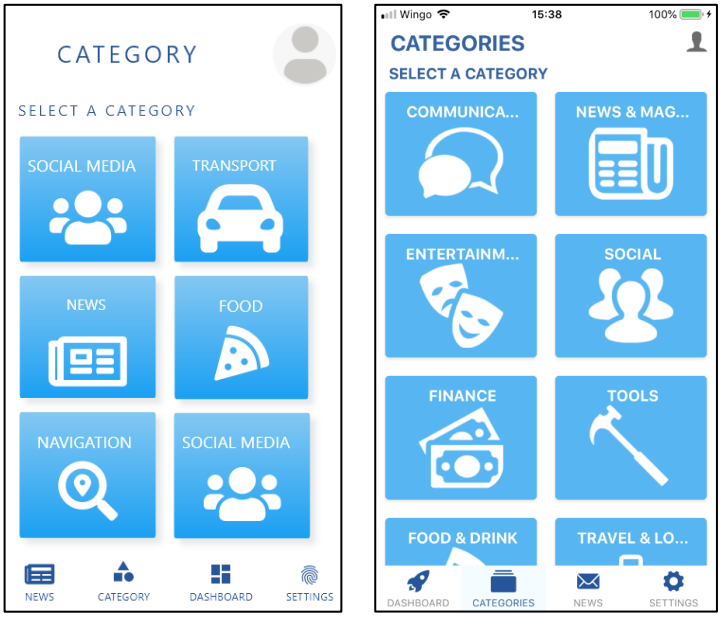


Figure 10 - Mockup / MobileApp Categories Screen

Categories Detail Screen

This is the screen that opens up, when clicking on a category tile. It displays the app icon, name and description. The start on the top right of every app entry indicates whether an app is favourited. Pressing it, changes the state. Clicking on an app tile brings the user to the privacy policy screen. We thought about showing the privacy rating of each app on this screen but decided against it. It would have overloaded the screen and we preferred a tidier design.

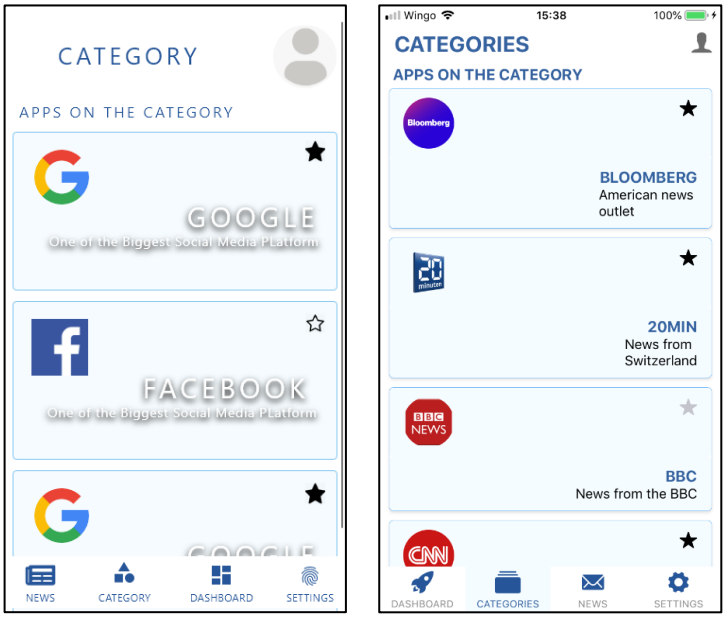


Figure 11 - Mockup / MobileApp Categories Detail Screen

App Privacy Policy Screen

This screen shows all the data privacy information of an app. The different policies are collected into categories to not show too much information on a first look. The categories can be expanded and the individual policies are shown as bullet points. This way, the screen is kept as clean as possible and the scrolling is minimized. Under “Privacy Officer Contact” the user can open communication using third party software to the privacy officer of that company.

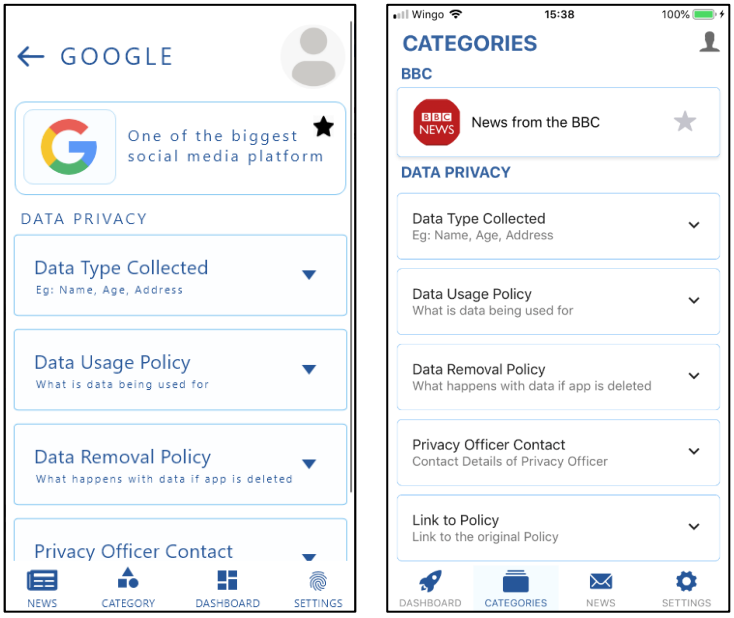


Figure 12 - Mockup / MobileApp Privacy Policy Screen

Dashboard Screen

This screen shows the favourited apps of a user and their rating. The apps are sorted into categories and since an app can have multiple categories associated to it, an app can have multiple entries on the dashboard. The rating of a privacy policy is shown with color coded flames. We did not find this design very intuitive and proposed some alternatives but the customer chose to stick with this design. Since our customer is from South East Asia and we, the developers, are based in Europe, this kind of difference can be explained with cultural differences. Our usability tests with Swiss users confirmed our fear and as a consequence we put in a small description to clarify the rating system better.

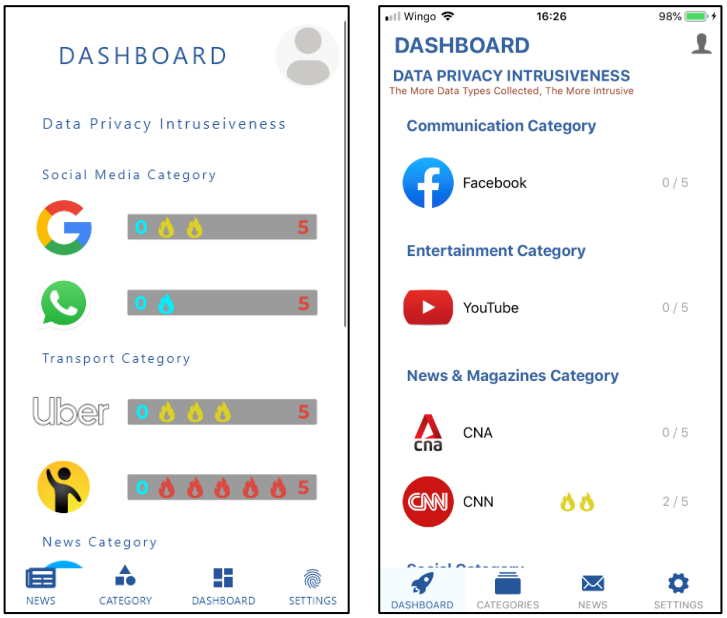


Figure 13 - Mockup / MobileApp Dashboard Screen

2.5.2 MobileApp Screenflow

In order to visualize the interaction of the individual screens, we have created a screenflow. Like the individual mockups, this was very practical to see the structure of the screens and their dependencies during development. It allowed us to create the navigation and routing of the app right from the start and served as the basic structure of the application.

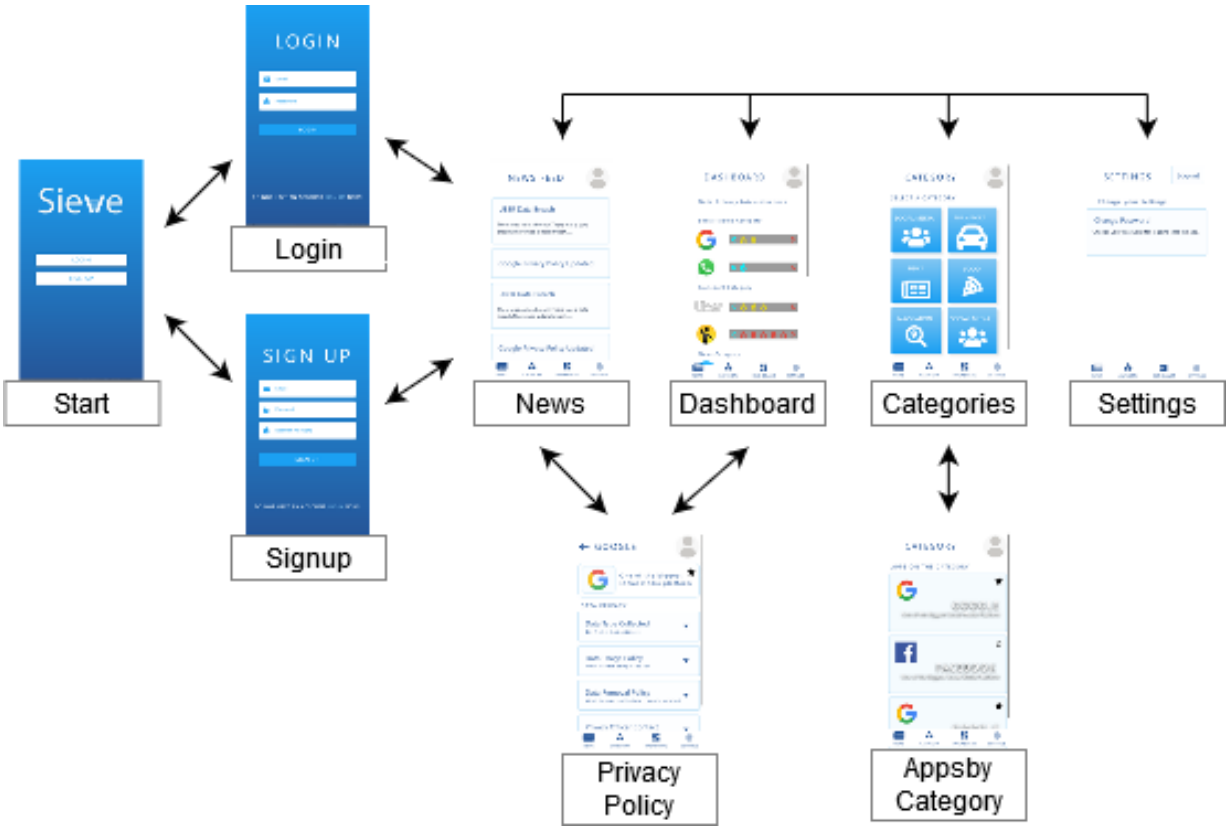


Figure 14 - Mockup Screenflow

As displayed, one cannot use the app without a registration or login. This was a requirement by the customer, in order to bind the users. It is also needed, to be able to save interesting privacy policies to the favourites. After the login, the user lands on the dashboard page in the final product. As the graphic on top shows, we intended to land on the News screen, but the customer preferred starting with the Dashboard. Starting with the Dashboard is slightly confusing for the user, because it is empty on the first startup, before adding any favourites. But other than that, it makes sense as a landing page. Navigation is done via a bottom tab bar, it allows the user to reach all the pages easily. If a user is on a subscreen, he can either click on the corresponding navigation element or use the back button to return to the previous main screen.

2.5.3 AdminWebsite Screens

The mockup serves as a common basis for discussion about the structure and functionalities of the AdminWebsite. Especially the structure of the data type collected was discussed using these mockups for visualization.

Privacy Policy Screen - View
The recorded apps are displayed in a table. By clicking on one, the “Privacy Policy Screen” is shown.

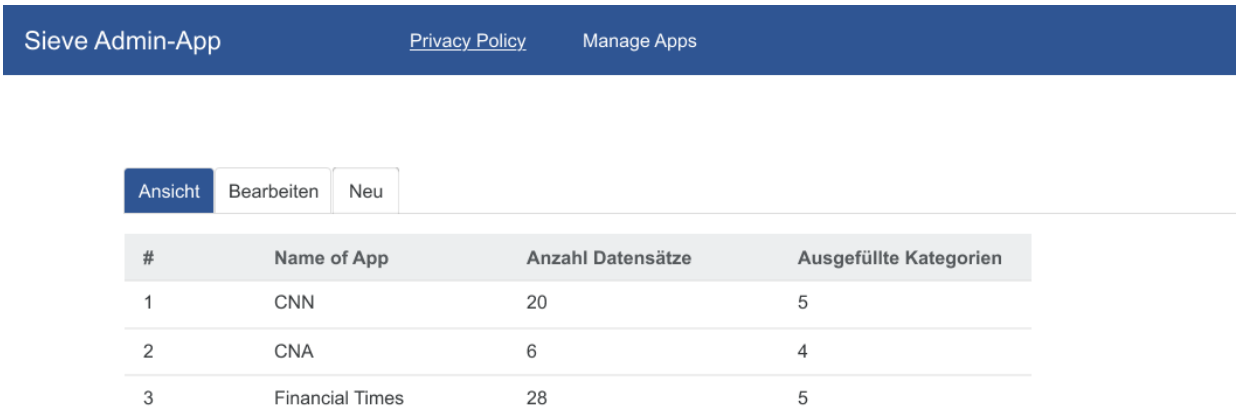


Figure 15 - Mockup - Privacy Policy Screen - View

Comparison to the final version of the AdminWebsite
The navigation was expanded to fit all the functionality. The rows in all the tables are sortable. The information of the privacy policy (e.g. Number of datasets) is not being shown in this screen, because it’s not of interest to the administrator. There are three actions for every app / privacy policy registered:

- View: Look at the app and privacy policy without the possibility of changing anything.
- Edit: The “Privacy Policy Screen - Edit” shows (described below), where all the values can be changed.
- Delete: When clicking it, a small button pops up above the button to confirm the deletion.

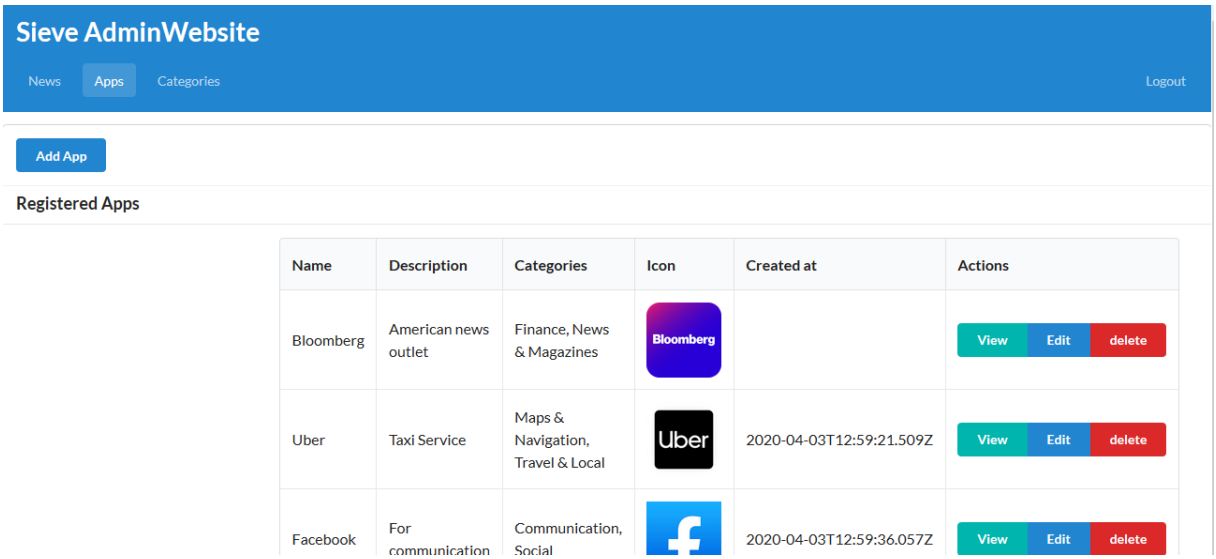


Figure 16: AdminWebsite - App Screen - View

Privacy Policy Screen - Edit

This screen allows the administrator to edit the privacy policy. It is categorized by type. The fields in Data Type Collected are fixed and were predefined by the customer. Adding additional information / fields is an optional feature.

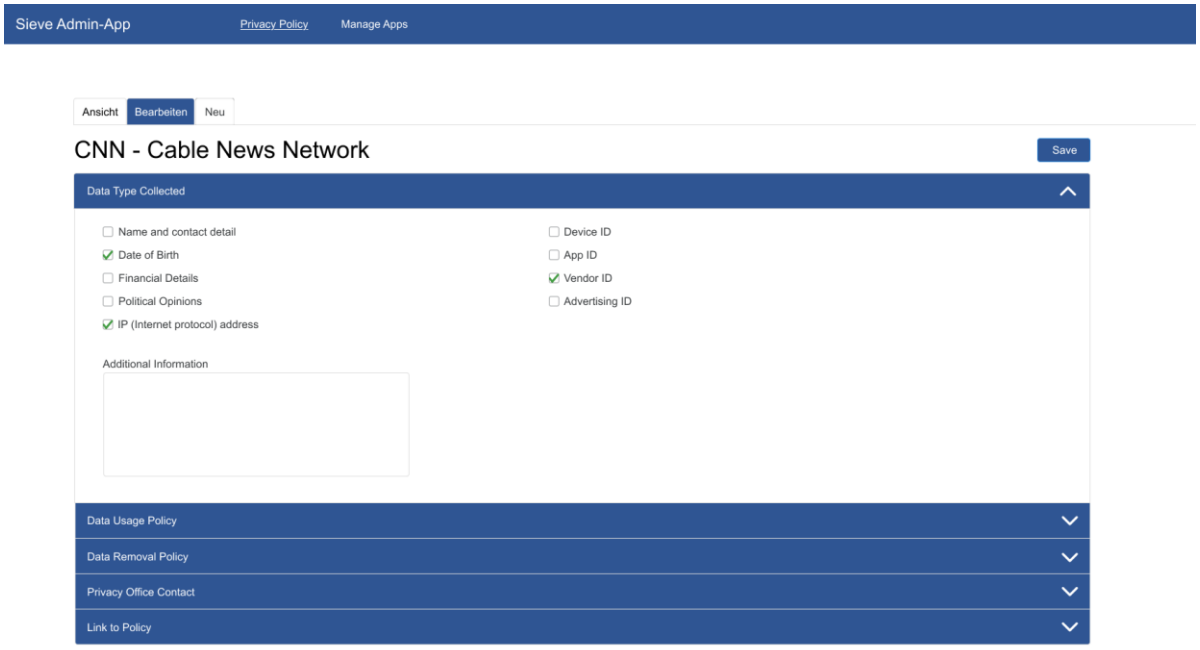


Figure 17: Mockup - Privacy Policy Screen – Edit Data Type Collected

The fields in Data Usage Policy and Data Removal Policy can be extended and removed dynamically.

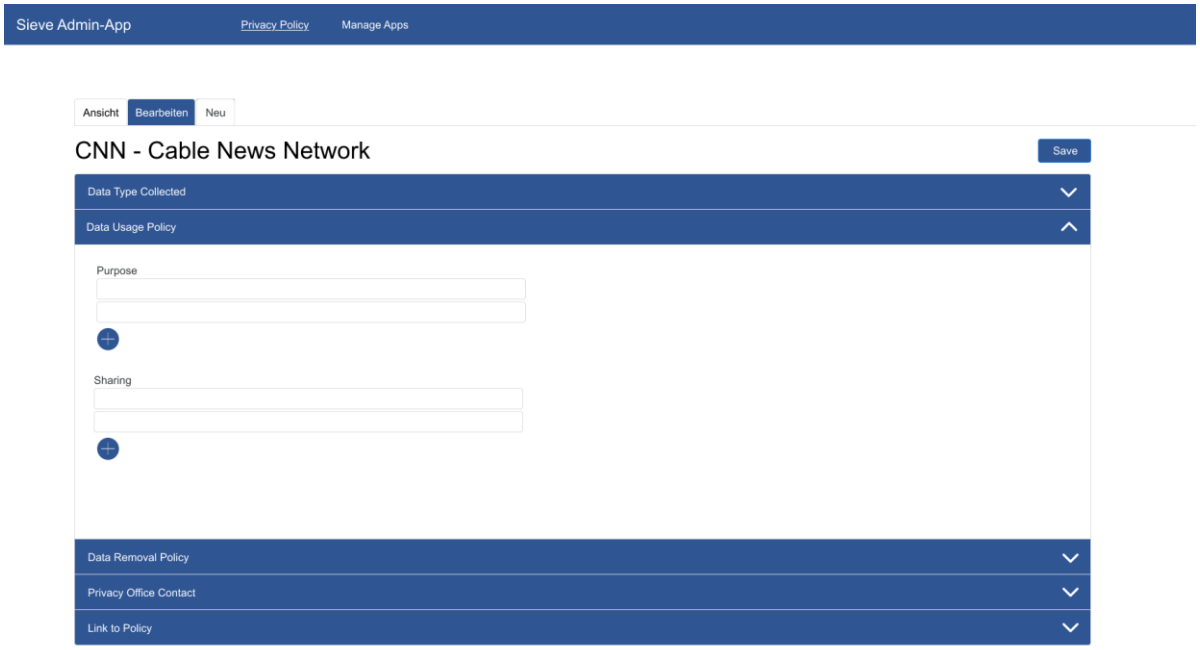


Figure 18: Mockup - Privacy Policy Screen - Edit Data Usage Policy

Comparison to the final version of the AdminWebsite

The edit of the app and privacy policy were added to the same screen because one cannot exist without the other.

Setting the data type collected was solved with a multiple search selection dropdown. Compared to the checkboxes, it needs a lot less space. In the other categories, fields for the other privacy policy category can be added and removed manually.

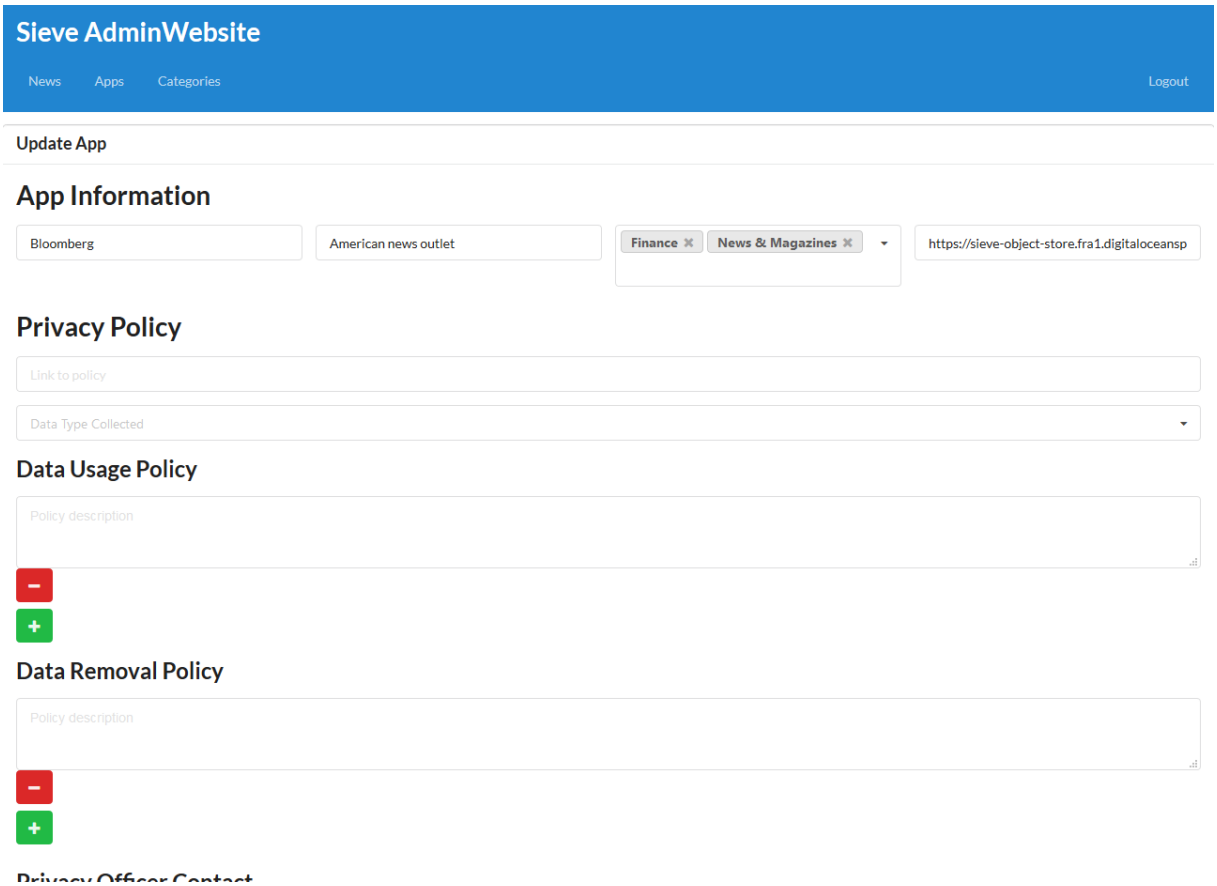


Figure 19: AdminWebsite - App / Privacy Policy Screen - Edit

3. Software Architecture

Contains a description of every component used to build and run the software. All the software except the app is hosted on Digital Ocean. It offers all the functionality that is needed, the deployment is simple and the costs are comparatively low.

3.1 Design decisions

JavaScript

Allows us to use the same programming language on the MobileApp, AdminWebsite and the WebServer. This way everyone can work on every application without having to change the development environment. There is a big community around JavaScript, so a lot of content for problem solving can be found online (not always an advantage) [4].
The client notified us early on that he plans to continue the project after the term project. Because there are a lot of JavaScript developers out there, it should not be too hard to find somebody to continue the development for a reasonable price.

Node.js

Because we do not need a lot of logic (computing server-side) and Node.js is very good at handling huge amounts of simultaneous connections, it's the perfect technology to work with for the WebServer in this project [5]. The entry hurdle is not that high and even simple authentication can be setup in just a few hours.

React

React is one of the most popular front-end frameworks for the web. Compared to its biggest competitor Angular, it has a lower entry hurdle and allows us to start developing more quickly. Because we also use React Native, it allows us to reuse the same concepts [6].

React Native

React Native allows us to develop the iOS and Android app with the same codebase and therefore saves us a lot of time. It's under active development as big players use it as well (Facebook, Skype, Bloomberg, AirBnb,…). Microsoft even created their own fork "React Native for Windows" [7].

Expo

Expo is based on React Native and integrates additional features. It allows us to build the application written with React Native and test it on the devices. This enables the customer to test the application and give feedback during the development [8].

MongoDB

MongoDB is an object oriented, NoSQL database that uses JSON to store documents and works well with Node.js. Because we do not have complex queries, the document-approach seems to be perfect for this project. If the app has a lot of traffic in the future, MongoDB will be easily scalable [9].
Even if this was the wrong decision, changing to a different database should not be a big effort.

3.2 System overview

The diagram provides an overview of the system with a detailed description of all the individual components. All the components are hosted on DigitalOcean.
The Developer Documentation gives a more detailed overview of how to use the different components.

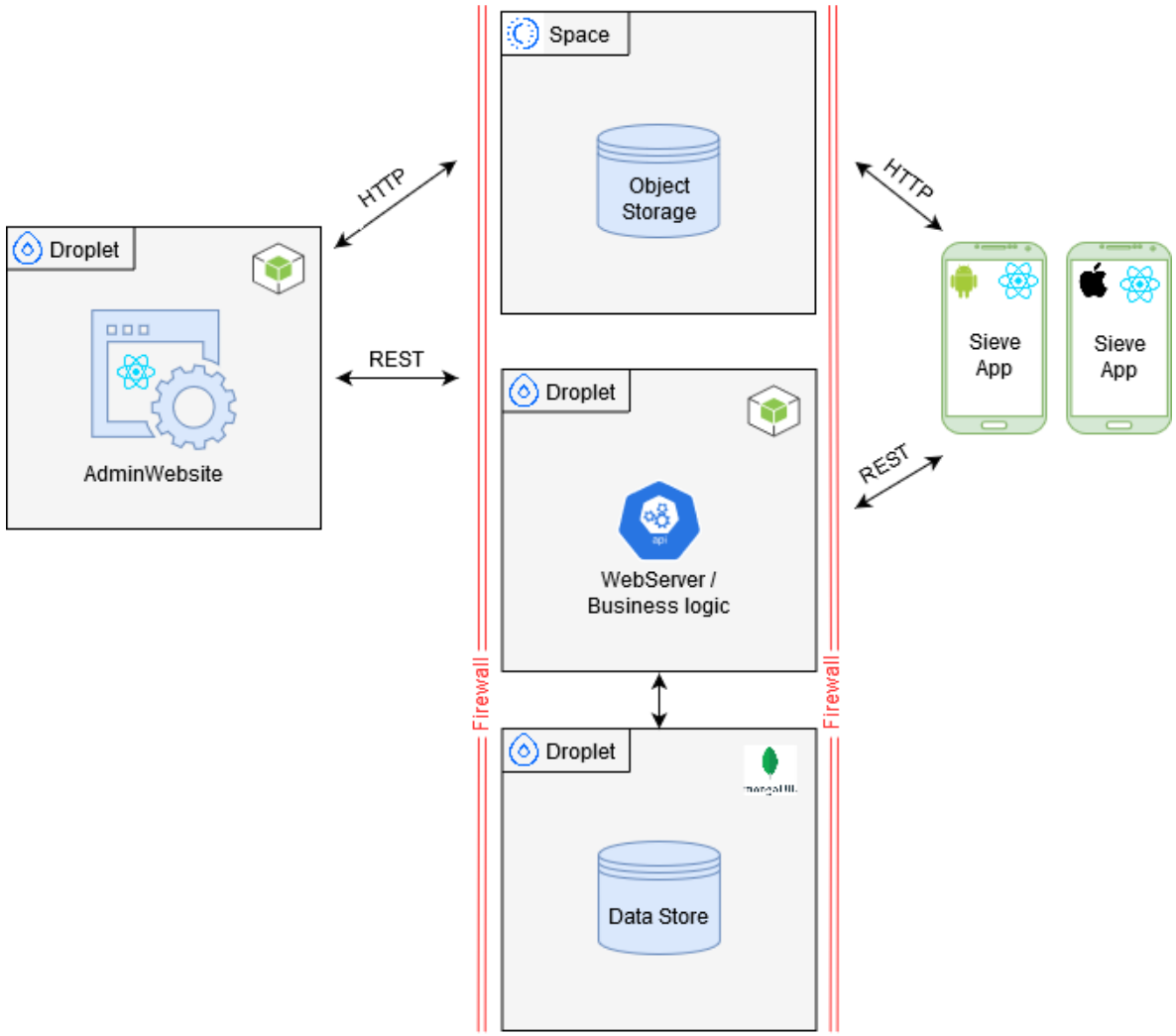


Figure 20 - System overview

Sieve App (MobileApp)

The mobile application is implemented with React Native and the help of Expo. The plan is to publish the app in the app store at some point, this is also where all the code and data will be stored, no extra droplet needed.
It is connected to the WebServer to exchange data and the object storage to get the app icons.

Admin Tool (AdminWebsite)

The Web application for the administrator is implemented in React Native. It provides CRUD operations for the Admin to manipulate the data shown in the MobileApp. It uses the same connections as the MobileApp does, just the authorization is different.

Data Store

The default MongoDB droplet that DigitalOcean offers is running with a few tweaks e.g. authentication and the accessibility restrictions. Backups can be made easily by just creating a snapshot of the droplet. For local development, a simple MongoDB docker container can be run and works exactly the same.
The database can not be accessed directly. If one wants to manage the data, it has to be done through the WebServer either with the AdminWebsite or the API.

Web Server (WebServer)

The WebServer is realized with Node.js, it is the interface between the AdminWebsite, MobileApp and the database. It offers a REST-API which makes it possible to add further functionality / applications later on.

Object Storage

The object storage is a special service provided by DigitalOcean, that allows the simple storage of objects like in our case app icons. Whenever an icon is uploaded, it gets a unique URL that can be used to display the icon on the AdminWebsite and the MobileApp. Therefore, the URL only has to be stored in the database and there is no connection between the WebServer and the object storage necessary.

3.3 Architectural goals & restrictions

3.3.1 Goals

Expandability
Because a REST-API is being used, it will be easy to add new features in the future. The API will be documented, so it is easy to add new functionality to the WebServer or just use the API for new projects.
The AdminWebsite, WebServer and MongoDB will all be deployed to Digital Ocean. Because of the competitive pricing, the easy setup and good documentation, future extensions or new versions can be deployed without problems. Another advantage of the cloud hosting is that if the traffic should spike, the only thing to do is an upgrade of the selected Droplet size.
State of the art tools were used for the development and deployment. Those tools will likely be around for a long time and there are a lot of developers with skills working with them.

Usability
An optimal user interface and usability is of great importance. Through meaningful naming of the individual functions, suitable icons and a very simple structure, the application is made more comprehensible for the end user. It should be clear on which screen the user is located and which functionalities are possible.
By using different, easily distinguishable colours, we ensure that the user can easily distinguish the individual elements from each other and that clickable areas are easily recognised.

Privacy and Security
Since the user discloses personal information about himself, the general data protection regulations must be complied with. This includes the storage of passwords only in hashed form and that no user has access to another users private information.

3.3.2 Restrictions

MobileApp
Because all the data is being stored on the server and none on the phone, the user has to have a stable internet connection to interact with the app.

WebServer and AdminWebsite
Because both is hosted on Digital Ocean, we are dependent on it. If Digital Ocean should be down, the MobileApp is not working either. Digital Ocean guarantees an uptime of 99.99% in its SLA's for all Droplets.
What a downtown of 99.99% means:

- Daily: 8s
- Weekly: 1m 0s
- Monthly: 4m 22s
- Yearly: 52m 35s

With only a small number of users, this is not an issue.

3.4 Tools

These are the main tools, that were used for development:

WebStorm
WebStorm is the web IDE by IntelliJ. It offers excellent support for JavaScript like intelligent code completion. It also allows to debug React Native with Expo, which not many IDE's can. WebStorm is well supported and documented [10].

GitLab
GitLab is our source code management and CI/CD tool of choice. It is a complete DevOps platform, that offers multiple services [11].

Postman
Postman is a collaboration platform for API development. We use it to help us develop and test the REST API of the WebServer [12].

Docker
Docker runs virtualized software packages called containers. We use Docker to run the MongoDB during development. Docker allows to start the container from a predefined state, which enables us to do reproducible tests using a prepared database image [13].

Robo 3T
Robo 3T is a lightweight MongoDB GUI. It allows us to easily check database interaction of our application during development, by just showing the current state of the database [14].

3.5 API Description

The API is the interface between the AdminWebsite, the MobileApp and the database. All the requests to the WebServer will be responded to in JSON format.

Because the requests can only be performed by an authenticated client, one always has to include the access token as Bearer-Token for every request to the WebServer.

The following endpoints are defined in the backend. The detailed API-Documentation can be found in the appendix.

Endpoint	Options
/user	GET, POST
/user/{id}	GET, PUT, DELETE
/user/admin	POST
/user/authenticate	POST
/user/favorites/{id}	GET
/news	GET, POST
/news/{id}	GET, PUT, DELETE
/apps	GET, POST
/apps/{id}	GET, PUT, DELETE
/category	GET, POST
/category/{id}	GET, PUT, DELETE
/category/app/{id}	GET
/privacypolicy	POST
/privacypolicy/{id}	GET, PUT, DELETE
/categoryapp	POST
/categroyapp/{category_id}	GET / DELETE
/appselected	POST
/appselected/{user_id}	GET
/appselected/detail/{user_id}	GET
/appselected/{id}	DELETE
/appselected/{id}/{user_id}	DELETE
/datatypeselected	POST
/datatypeselected/{pp_id}	GET
/datatypeselected/{id}	DELETE
/datatypecollected	POST, GET
/datatypecollected/{id}	PUT, DELETE

Table 4 - API routes

3.6 Deployment

A mobile device is needed to run the MobileApp. At least Android version 9.0 or iOS version 12.4.5 is required. Firefox, Chrome or Safari is required to access the AdminWebsite.

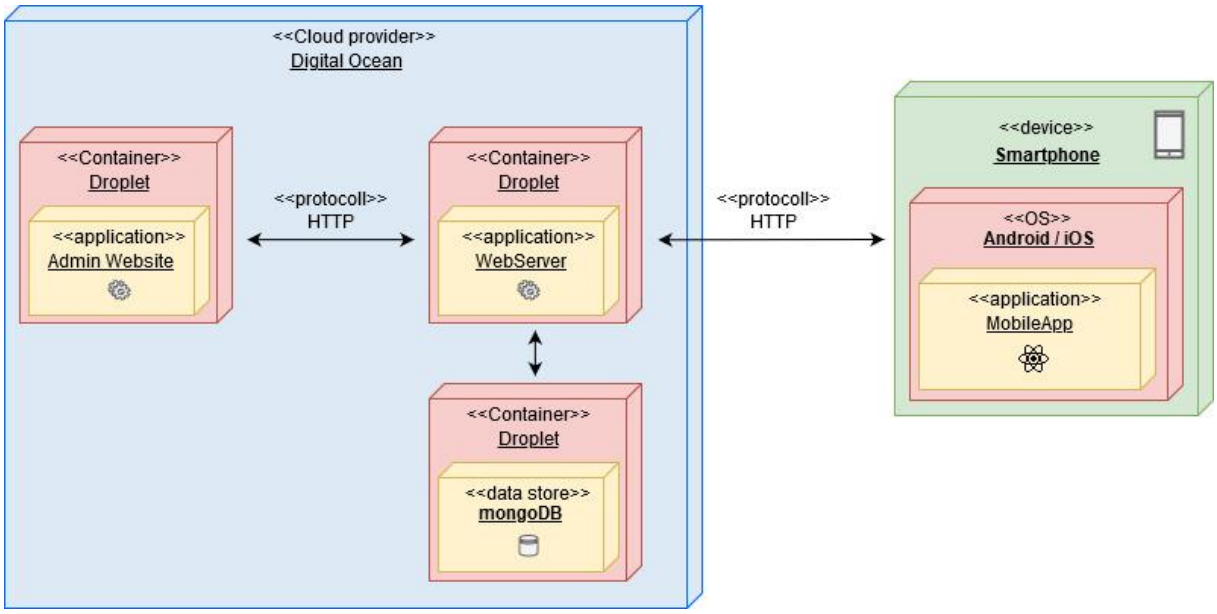


Figure 21 - Deployment diagram

All the components are running on droplets that can be accessed with SSH. Because of the automated deployment, this is only rarely needed. The database is behind an extra firewall and only reachable with its internal IP-address.

3.6.1 Automated deployment

To avoid having to deploy the applications manually every time changes to the code are made, we set up deployment pipelines that are executed every time a commit to the production branch is published. First, the allocation is tested and only if the tests succeed the application is built and then deployed. This helps us to focus on the programming and not the publishing itself.
Whenever a pipeline fails, the user that made the commit is informed by email.

The pipelines run on GitLab CI/CD. The MobileApp is deployed to Expo. This process takes some time, but after being finished the new version of the app can be accessed using Expo. The WebServer and AdminWebsite are deployed to their respective droplets on DigitalOcean. After the code on the Droplets is updated, the processes are automatically restarted and the services are available again almost instantly.

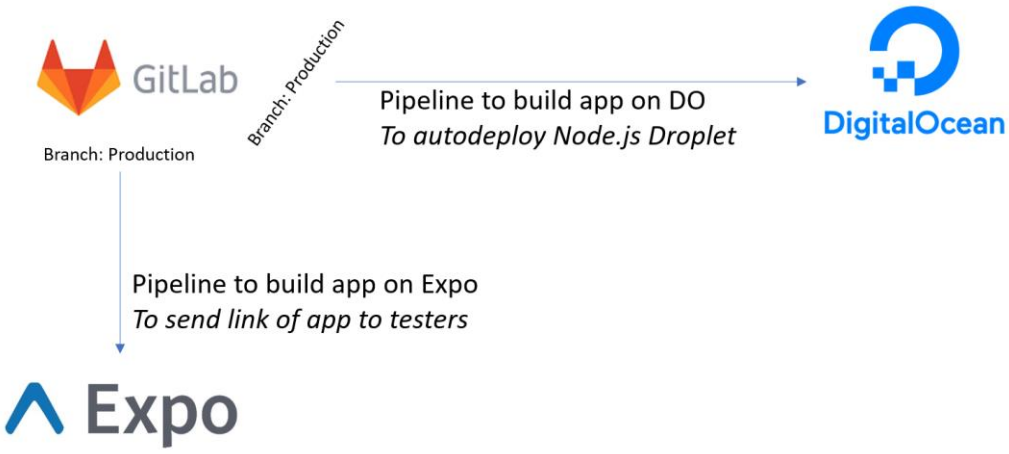


Figure 22 - Automated deployment

3.7 Data Storage

Based on the domain model, a database model was created, which is described by the following diagram:

Figure 23 - Database model

It was used to build the database and to develop the API of the backend.
The blue tables map the privacy policy including the dependent tables. In green, all tables that are needed for app queries. The news table is linked to the app table because in the future it should be possible to relate them to an app, for example when a privacy policy has been updated, there will be a new news entry generated (optional feature).
User related data is greyed out. This data is mainly used to distinguish if a user is an admin to log on to the AdminWebsite or not and to assign the selected apps (favourites) to the user.

3.8 Sizes and Performance

All of our containers are running on Digital Ocean. Compared to other cloud-providers where we would have to run our own VM, Digital Ocean allows us to only have containers and not having to take care of updates, security and more. This also makes it easier for the client to maintain the setup after the project.
At the moment the cheapest offer ($5 a month) has more than enough power. The advantage of deploying in the cloud is that if there should ever be the need for more size or performance, it is "click to scale". If needed, the containers can be hosted in different counties. For now, we have everything running in Amsterdam.
The way we designed our API, only a minimal number of requests are needed to display data on the MobileApp. The icons are even outsourced on a separate object storage (Spaces on Digital Ocean). Therefore, the app is running faster and the WebServer has less incoming requests to handle.

The following graphic shows a possible vertical scaling scenario on DigitalOcean. First of all, a load balancer would be put in front of the WebServers. Multiple WebServer instances would help with the huge number of requests coming in. The MongoDB would be scaled with more memory and CPU power.

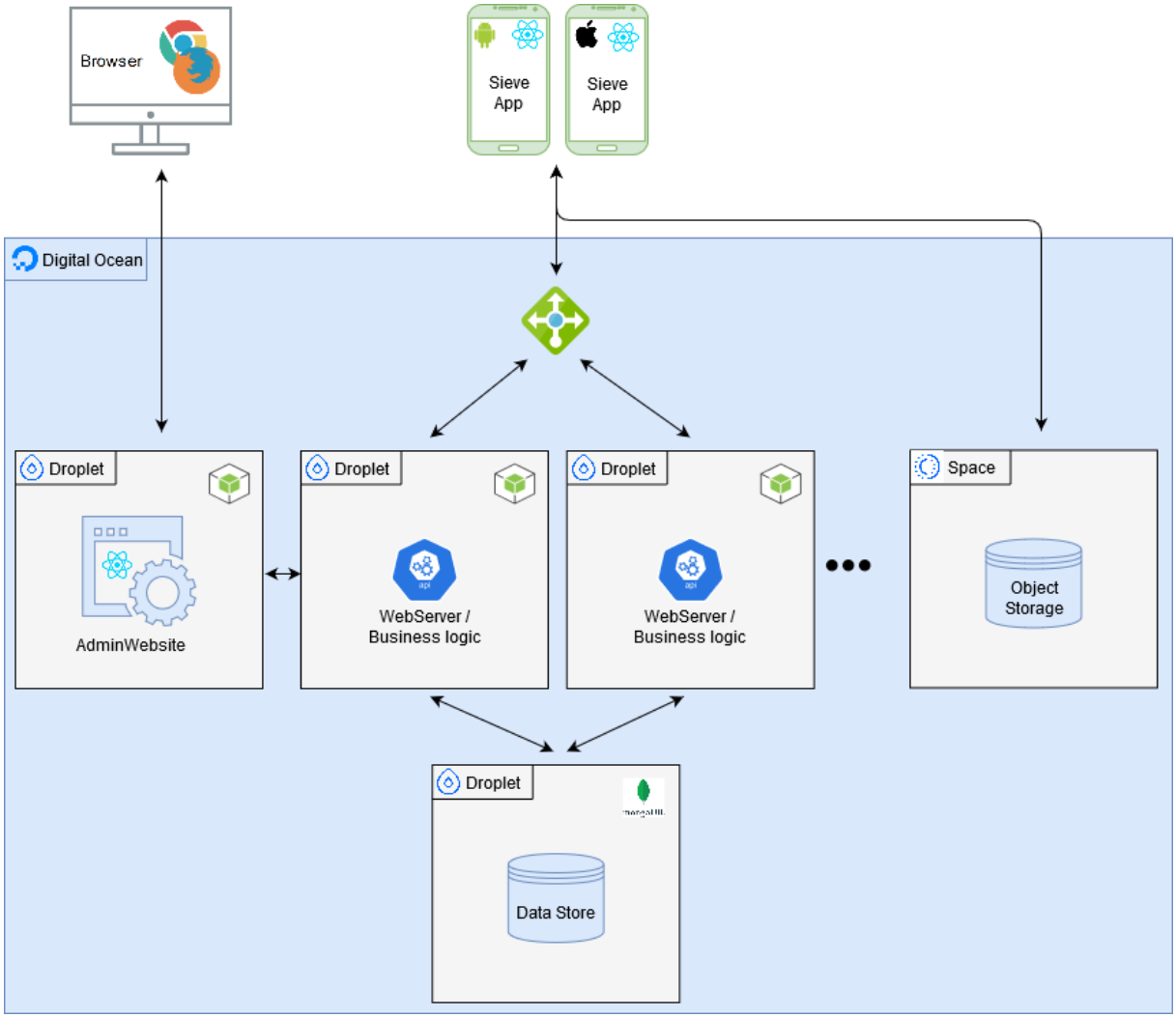


Figure 24 - Size and Performance

4. Quality assurance and testing

To assure a high quality of the end-product, we took several measures. One part was setting up the development processes, to make sure that code was being written properly. The other part was performing a variety of tests on the different aspects of the product.

4.1 Performance tests

During the project, the client asked us how many concurrent requests our application could handle. We knew that Node.js is event-driven and thus can handle parallel requests by design. Generally speaking, Node.js can handle thousands of concurrent requests, but if there is too much computation to be done in the backend, the response times can spike up heavily. In theory, the WebServer's processes are supposed to be computationally simple and response times should stay fast, even under a heavy workload. To make sure that this was the case, we decided to use a simple command line tool called ApacheBench. It is an HTTP server benchmarking tool, which can send many concurrent requests to a server and report on the response times. [15]

```
Benchmarking 167.172.44.35 (be patient)


Server Software:
Server Hostname:        167.172.44.35
Server Port:            3000

Document Path:          /apps/
Document Length:        35 bytes

Concurrency Level:      1000
Time taken for tests:   1.019 seconds
Complete requests:      1000
Failed requests:        0
Non-2xx responses:      1000
Keep-Alive requests:    1000
Total transferred:      331000 bytes
HTML transferred:       35000 bytes
Requests per second:    981.18 [#/sec] (mean)
Time per request:       1019.185 [ms] (mean)
Time per request:       1.019 [ms] (mean, across all concurrent requests)
Transfer rate:          317.16 [Kbytes/sec] received
```

Figure 25: Benchmarking of 1000 concurrent requests of /apps on WebServer

The results were as expected and the WebServer had no problem handling 1000 concurrent requests. It is hard to make a quantitative statement from the response times because they depend on the server hardware and the internet connection of the tester. But we were able to make a qualitative assessment and rule out any major problems with response times.

4.2 Metric analysis

Using metric analysis, developers are able to find problematic parts of the code to revise. In addition, it gives an overview of the code base. To get all the metrics data, we used the Metrics Reloaded plugin for WebStorm. The tool allowed us to determine the number of lines of code and the number of files by file type.
The Lines of code indicate how many characters are physically present in the source code. This can be derived by the number of line breaks in the project. The Non-comment lines of code correspond to the effective lines of code, these are the lines of code minus all comment lines, empty lines and lines with only one opening or closing bracket.

Metrics	Size
Lines of Code	57’046
Non-comment lines of code	56’882
JavaScript lines of code	5’418
Non-comment JavaScript lines of code	5’254
Number of JavaScript files	79

Table 5 - Metric analysis – MobileApp

It is evident that React Native and the used frameworks Expo and Native Base have created a large overhead for the administration of components and modules. This is shown by the large number of lines of code that do not correspond to the JavaScript lines of code.

Metrics	Size
Lines of code	1’866
Non-comment lines of code	1’791
JavaScript lines of code	1’615
Non-comment JavaScript lines of code	1’536
HTML lines of code	46
Non-comment HTML lines of code	26
CSS lines of code	45
Non-comment CSS lines of code	45
Number of JavaScript files	14
Number of HTML files	1
Number of CSS files	2

Table 6 - Metric analysis – AdminWebsite

Based on the lines of code compared to the javaScript lines of code, it is easy to see that the overhead for the AdminWebsite and the WebServer is much lower than for the MobileApp. This is due to the fact that the AdminWebsite is written in React, which was developed for one platform, the web browser, and not for multiple platforms like iOS and Android.

Metrics	Size
Lines of code	1’316
Non-comment lines of code	1’288
JavaScript lines of code	1’255
Non-comment JavaScript lines of code	1’229
Number of JavaScript files	30
Number of tests	32 Unit Tests + 44 Postman

Table 7 - Metric analysis – WebServer

The same as for the AdminWebsite also applies to the WebServer, which has no graphical user interface, but is the interface between the applications and the database.

Metrics	Size
Lines of code	60’228
Non-comment lines of code	59’971
JavaScript lines of code	8’288
Non-comment JavaScript lines of code	8’119
HTML lines of code	46
Non-comment HTML lines of code	26
CSS lines of code	45
Non-comment CSS lines of code	45
Number of JavaScript files	123
Number of HTML files	1
Number of CSS files	2
Number of tests	32 Unit Tests + 44 Postman

Table 8 - Metric analysis – entire project

If we now look at the three projects as a whole, we can see from the table above that we have written an average of 64 lines of code per file, which is a pretty good average. There are many different rules on how many lines of codes should be written per file, but everything under 120 Should be ok. [16]

4.3 Unit Tests

The unit tests were performed on the Webserver on every route. The test framework Mocha was used to perform the tests and Chai as the assertion library.
To test the database queries as simply as possible, an extra docker container with a prepared database-configuration, that can be imported at the beginning, was set up.
Because there is not a lot of business logic, usually the returned value stored in the database was tested.

We made a separate test section for every API route. Inside it the CRUD-terms were used for a better overview. Additionally, the time required to perform the test is also displayed for every test. This is very helpful for troubleshooting if the WebServer is running on a very slow notebook and the response takes more than 200ms, the tests don't work.

```
Datatypecollected
  Get all
    √ Should return success (98ms)
  Create Datatypecollected
    √ Should return success (89ms)
  Update Datatypecollected
    √ Should return success (86ms)


32 passing (3s)
```

Figure 26: Unit Test success

4.3.1 Challenges

The API requests are performed with the chai-http module. The issue here are the asynchronous API request queries and that there is usually a DELETE request following a CREATE request. Usually this is not a problem because the business logic and not the API is being tested. Because the tests are not running in any order, it is often the case that DELETE is called, even before CREATE can be finished.

Solution: Creating custom functions for example for the admin authentication. They can be used before every test if needed. This way, the test has to wait until that function has finished and returned the values.
Because it does not make sense to do a function for every function that cannot run asynchronously, we decided to use Postman to test the DELETE requests.

4.3.2 Postman

Does not do unit testing but because there is no business logic, it has the same effect for testing the API. In the appendix "API Documentation" is a more detailed documentation on how to use it. Additionally, there is also an export of all the routes with demo requests already prepared.
We mostly used Postman because of its easy setup, overview of all the requests and simple starting of requests.

4.4 Integrationtests

We decided to perform Snapshot tests for the AdminWebsite and the MobileApp. During Snapshot tests a chosen component of the application is rendered into an HTML file and then compared to an older version. This way, one can check if the changes to the code had unintended effects on the rendering. If the change is legitimate you can update the snapshot for future tests. Our intention was to have a snapshot test for every screen, to avoid unwanted side effects of code changes. We chose Jest as the test framework and test renderer, because it is the recommended one when working with React or React Native. [17]

```
PASS  __tests__/newsfeedScreen.test.js
  √ renders correctly (133ms)
```

Figure 27: Snapshot test success

4.4.1 Mocking

Testing with REST calls to a real WebServer is not a very good idea. It is complicated to automatically start the WebServer and database for every test run in the correct state to be able to have reproducible tests. Furthermore, changes to the server might make the tests on the AdminWebsite fail, which is an unwanted effect for this kind of test. To avoid this, we mocked the API calls using static responses saved in JSON files. For every test, the correct responses have to be prepared in the right order. This would guarantee isolated and reproducible tests, where the results should be understandable.

4.4.2 Issues

We ran into a lot of problems setting up the Snapshot tests. Most of them were due to the asynchronous elements in our code. The way React and React Native work, the components are first rendered empty and when this succeeds the REST calls are sent out to get the real data to display. After getting the response the component is than rendered. REST calls are asynchronous by nature because they are sent over the network and can take some time to be answered. Even though in theory our mocked responses were not asynchronous, the code structure was done in a way to allow asynchronous responses for the real application.
The core problem is that the test does not know when the rendering is finished and therefore compares and creates the snapshot at the wrong time. There are some possible workarounds like setting a hardcoded timer to wait for the rendering, but they all bring their own problems. Using workarounds, we got some tests to work but inexplicably failed others.

4.4.3 Conclusion

Overall, we were not happy with the snapshot tests in our application. It seems that these are not meant to be run using asynchronous elements in the rendering with React and React Native. We invested a lot of time but got no satisfying results. We still like the idea of Snapshot tests and would like to use them in the future but are now aware of possible problems and would analyse our intended application more in detail before choosing them.

4.5 Usability tests

In order to get feedback as early as possible, the MobileApp and AdminWebsite were shown to the customer and project supervisor at the end of milestones to get feedback as regularly and early on as possible.
To test the usability and user-friendliness of the MobileApp, tests with project-independent people were conducted. No usability tests were carried out on the AdminWebsite, because only the administrator can access it and the customers feedback on it was enough for us.

4.5.1 General conditions

Based on the user stories and scenarios defined in the document Requirements Specifications, concrete test cases were created and played through with every test person.
For the observer to learn what the test person is looking for or what to expect, they were asked to think out loud.
To have the same conditions for every test, a test database with all the necessary data to solve the different scenarios was set up.

Selection of test person

The test subjects should be as diverse as possible. It is important to use people from different professions with different experiences and in different age groups for the tests. This way you get the best possible feedback on the usability.

Test device

The user is using his own phone. The expo-App has to be installed in advance and the app is open at the start of the test.

4.5.2 Test cases

The defined scenarios should simulate the use of the app in everyday life as well as possible. We tried to cover all functions of the MobileApp with the following five test cases that were then given to the testers.

Test Case 1: Which is the last news reported?
As a user interested in privacy, you want to check if there are current news on the privacy policy that concerns you.
To get an update, sign up for an account and log in to the application. Once you are logged in, read the latest news. What is the latest news?

Test Case 2: Check out BBC's privacy policy
You like to know what is going on in the world, so you recently installed the News app from BBC on your smartphone.
Now you would like to know if and which private data BBC collects from you. Find out if the BBC collects data from you. If so, which data?
You do not agree with this and would like to contact the privacy officer. Contact the privacy officer.

Test Case 3: Add an app to the favourites
You have now read the BBC's privacy policy and would like to follow it. To not search for it again, you want to add it to your favourites.
Add the BBC app to your favourites.

Test Case 4: Check out the rating
After adding the BBC app to your favourites, you now have the app on your dashboard. You would like to know the rating of the app compared to the CNA app.
Find out which app has the better rating.

Test Case 5: Logout of the app
Because the smartphone is not only used by you, but also by other users, you decide to log out of the app. Log out from the application.

4.5.3 Evaluation

The usability test was performed between the 01.05.2020 and 03.05.2020.
The following is a list with all the weaknesses that were found and the measures to solve the issues.
The Complete Usability Tests can be found in the appendix.

Test Case	Observations	Measures
TC1	Some users would like to have a start screen that already contains information instead of the Dashboard screen, which is empty the first time they use it. This was confusing for some test users.	At first, we had the news screen as start screen, but this was changed to the dashboard screen at the request of the customer. His test customers had wanted the dashboard as their start screen, which also makes sense once you have apps in your favourites.
TC1	A user does not like the name "News" and suggested to call the navigation element "News about Privacy".	No measures, because a navigation element description must be short and concise and be labelled with only one word.
TC1	A user finds it better to be logged in immediately after registration, instead of having to log in.	No action, because this is a standard, that one has to log in after registration. So, there is also the possibility to send an activation mail later as an optional feature, if this would be desired.
TC2	With the highlighting of clickable links like the ones to the original privacy policy and email of the privacy officer, it would be more intuitive.	Highlight the text in blue colour, standard design of a clickable link.
TC4	The function of the dashboard was not clear to most users at the beginning.	Added a hint field on the dashboard, with a description of what needs to be done to display information.
TC4	Rating is confusing because the app with the least points is the best one. Users understand it the other way around.	Adding a text above the title that explains how to interpret it. In addition, display the numbers from zero to five differently (ex: 0/5), so that it becomes clearer that it is a rating.
TC4	A user would like a search functionality to find apps faster.	Add a search on the category screen. This is an optional feature (OF4).
TC4	A user would like to find the rating of an app not exclusively on the Dashboard.	Show an apps rating on the categories screen as well. This is an optional feature (OF8)

Table 9 - Test Case evaluation

The test subjects also found the navigation to be consistent and clearly understandable. The design is simple but beautiful and tidy. They quickly found their way through the app after checking out the individual screens.

Discovered Bug

At the end of April, the customer received a first build for Android devices. He tested the generated apk-file with different users. This resulted in a supposed resolution bug, one tester had a white space below the navigation, which only appeared on his device.
After we received the exact device data of the test user from the customer, we tried to reconstruct the bug to be able to fix it.
During the tests it was discovered that this did not depend on the resolution, as we could not reconstruct the bug even with different resolutions and devices. We asked the test user for more information about the operating system. We found that he was using a very old Android version. Now we were able to reconstruct these issues and found the same bug, but no matter what screen resolution. We had to conclude, that the issue was caused by the Android version 5.1, which was released on 09 March 2015. The latest version is Android 10 (Released on September 03, 2019).
On the left screen with Android version 7 the bug is not present and on the right screen with Android 5.1 you see the white space below the navigation.

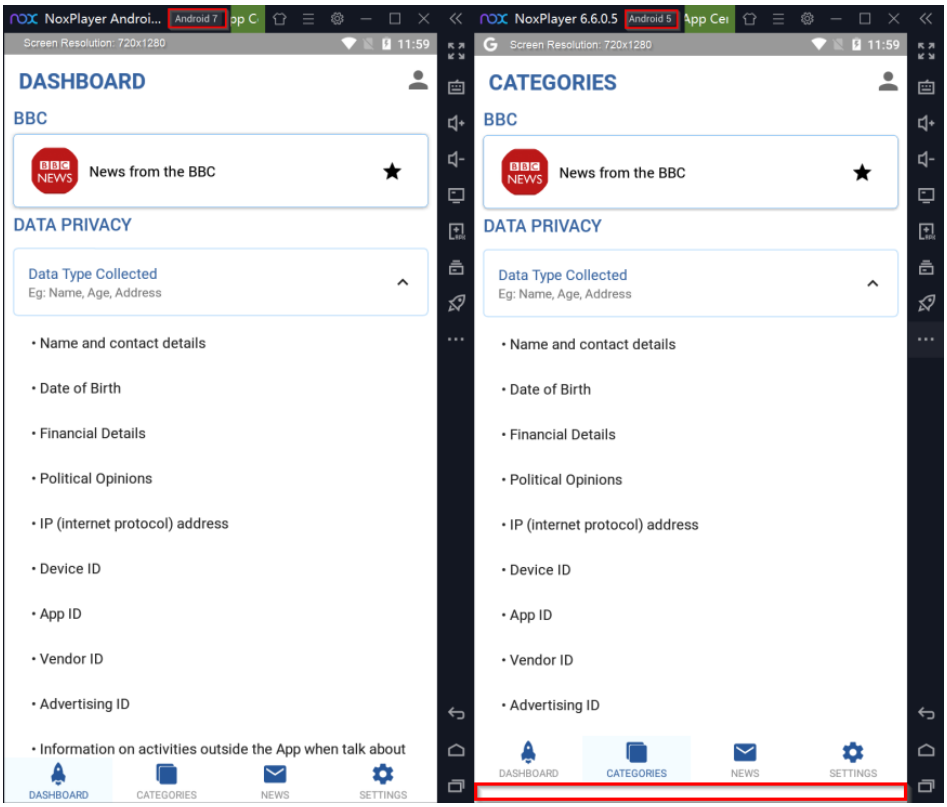


Figure 28 - Resolution bug

We have tried to fix the bug with various workarounds, but without success. Android 5.1 does not seem to be compatible with the architecture. After 4 hours we decided not to pursue the bug any further.
Since we agreed on the minimum Android version 9 for security reasons in the non-functional requirements and the effort to be invested does not bring any noticeable added value for the application.
The complete test protocol can be found in the appendix.

5. Results and Outlook

5.1 Achieved Goals

The achieved objectives can be measured on the basis of the functional and non-functional requirements. The functional requirements were classified as Use Cases and Optional Features with a priority of one to three. During the project, the features were subject to change by the developers and the client. Prioritizations were changed with the goal to satisfy Sieves needs and have a presentable working prototype at the end of the project. To achieve some stability over the short development time, we had to fixate the core Use Cases at the start of the project. All use cases except one could be implemented. In agreement with the client, we decided not to implement use case 10 "View privacy rating of installed apps" with priority 3, because this could only be implemented for Android for technical reasons and the goal was a platform neutral application. The use case 2 could only be partially implemented, because changing the password was defined as an optional feature due to the customer's priorities.

Use Case	Description	Prioritization [1 – 3]	Implemented
UC1	CRUD Apps interested in	1	Yes
UC2	CRUD account	1	Partially
UC3	CRUD apps	1	Yes
UC4	CRUD privacy policy	1	Yes
UC5	View privacy policy information of app	1	Yes
UC6	View privacy rating of apps	1	Yes
UC7	Find similar apps	2	Yes
UC8	CRUD privacy news	2	Yes
UC9	View privacy news	2	Yes
UC10	View privacy rating of installed apps	3	No
UC11	Open communication with privacy officer	3	Yes

Table 10 - Compliance with the functional requirements (use cases)

Additionally, seven optional features could be implemented. The most important ones on the MobileApp were the landscape mode and the optimized dashboard based on the test feedback. On the AdminWebsite an extra column to display an app's categories, as well as the functionality to sort by the columns was added. It is to note, that even though some possible functionalities of the AdminWebsite, like adding or removing app categories, were not developed, they are still implemented on the WebServer and available by using the REST API. The API is well documented and additional functionality can easily be added to the WebServer at a later stage.

Optional Features	Description	Prioritization [1 – 3]	Implemented
OF1	Implement the App GDPR compliant	3	No
OF2	Login screen – login with Facebook and Google	2	No
OF3	Login screen – sign Up with first name and surname	1	Yes
OF4	Category screen – search	2	No
OF5	Category screen - similar apps	1	No
OF6	Category screen – navigate with side wiping	3	No
OF7	Category screen – images at Apps	2	No
OF8	Category screen – apps with rating	2	No
OF9	Category screen - Different font size	1	Yes

OF10	Privacy Tipps screen - create screen	1	No
OF11	News screen – policy Updates messages, with App Icon	2	No
OF12	News screen – push messages for new entries	3	No
OF13	Settings screen – account data changeable	3	No
OF14	Settings screen – sorting adjustable	3	No
OF15	Settings screen – change password	1	No
OF16	Detail screen – Make chat with Data Privacy Officer look like real chat	3	No
OF17	Detail screen – Compare apps with side wiping	3	No
OF18	Detail screen – Privacy Policy key points highlighted	3	No
OF19	MobileApp - Navigation Order	1	Yes
OF20	MobileApp - Landscape Mode	1	Yes
OF21	Publish the MobileApp to the Google App Store	3	No
OF22	Publish the MobileApp in the Apple App Store	3	No
OF23	AdminWebsite – import of data with CSV	3	No
OF24	AdminWebsite – Admin can add or change fields independently	3	No
OF25	AdminWebsite – Category View	1	Yes
OF26	AdminWebsite – Categories CRUD	2	No
OF27	AdminWebsite – Automatically download Apps Icon from App Store	3	No
OF28	AdminWebsite – Automatically download Apps description from App Store	3	No
OF29	AdminWebsite – Data Type Collected points (1 or 5) Changeable	2	No
OF30	AdminWebsite – Change data structure	3	No
OF31	AdminWebsite – Filter apps by category	2	No
OF32	Transfer all data through https	2	No
OF33	Digital Ocean billing-API to show cost	1	No
OF34	Ordering of tables by row	1	Yes
OF35	Split displayed entries into multiple pages (e.g. apps)	3	No
OF36	Dashboard optimization	1	Yes

Table 11 - Compliance with the functional requirements (optional features)

Except of one non-functional requirement, they could all be completed. The requirement “compliance” had to be moved to the optional features, because of the significant time investment it would have needed and the fact that it would not have added anything to the demonstration purposes of the app. We still tried to be conscious of data protection where it did not cost us too much, like making sure to store all passwords in a hashed form or protecting our database from outside access. By design it was ensured that adding security features, like communication via HTTPS, would be possible without having to change the structure of the application.

Non-Functional Requirements	Fulfilled	Comment
Functionality		
Suitability	Yes	
Accuracy	Yes	Not directly measurable, continuous process of data entry by administrator
Interoperability	Yes	Tested with Android, iOS and various browsers
Compliance	No	Was set as an optional feature
Security	Yes	Tested on phone and browser

Reliability		
Maturity	Yes	Measured with usability test
Fault Tolerance	Yes	
Recoverability	Yes	The backup/restore can be done with a single click on digital ocean
Usability		
Understandability	Yes	Measured with usability test
Operability	Yes	Measured with usability test
Efficiency		
Time behaviour	Yes	Was tested with ApacheBench
Maintainability		
Analysability	Yes	Logfiles are stored on the WebServer
Changeability	Yes	The app is modular and therefore easily expandable
Stability	Yes	To this point we did not encounter any problems
Testability	Yes	Measured with Unit, Usability tests and Postman
Portability		
Adaptability	Yes	Measured with usability test
Installability	Yes	APK-file was delivered to the customer
Replaceability	Yes	DB-config document would have to be changed to connect with new database.

Table 12 - Compliance with the Non-functional requirements

5.2 Outlook and Extension

The following section explains how the applications could be further extended, modified and improved with two cases as example. Additionally to the two presented features, there is an entire list of additional features in the appendix. Rough concepts are presented. The feasibility of these extensions was not tested but only theoretically worked out. The first extension “Information from the App Store” was our idea and is something we would find very interesting to investigate further. The second possible addition to Sieve, we chose to describe, is “Natural Language Processing”. This idea comes from the customer and is something he wants to do in the future.

5.2.1 Information from App Store

The idea is to read app information like name, description or icon, automatically from app store, when registering a new App on the AdminWebsite. This would make the life of the administrator a lot easier, by not having to get the data for every app from the app store manually. Especially with the natural language processing that would automate the registration of privacy policies in mind, this would be important to further automate the registration process.

This possible functionality was registered in the form of two optional features:

- AdminWebsite – Automatically download App Icons from App Store (OF27)
- AdminWebsite – Automatically download App descriptions from App Store (OF28)

Neither the Android nor the Apple store provide this information in a usable format. But there are different independent providers, that offer such a service. The most popular is AppMonsta. The service searches the Android and iOS store every few hours for new apps and adds them to it’s list of apps. Currently, they advertise with the tracking of about 9.6 Mio. Android and 4.6 Mio. iOS apps. Their API provides all the needed information including the icon.
With the free tier, it is possible to make 3000 requests every month (limit of 100 per day) but limited to the apps published in the US store. This would be enough to at least test the service. If there are

more requests needed, prices start at $ 2’500 per month, which is a lot for the benefit it provides in this case. [18]
The problem with the service is that because they are not working with Google or Apple, there is no guarantee for the service to be available long-term. Changes to the app stores could mean the end of a service provider like AppMonsta.
In our opinion, the best solution would be to add the AppMonsta API to the AdminWebsite and allow the administrator to fill out the information automatically but still give him the possibility to control and add app-information manually.

5.2.2 Natural Language Processing

At the beginning of the project, it was planned to have a team in Africa to work on the natural language processing. The idea is to scan through the verbose form of the privacy policies using some sort of artificial intelligence, which is then able to extract the key aspects and present them in a standardized form. The generated information would then be automatically fed into the system using the WebServers API.
This would automate the app registration process and massively lower the workload of the administrator. He would not have to read, analyse, and then manually enter the data of every app into the AdminWebsite.
Natural Language Processing is a very complex topic, which involves the use of artificial intelligence. We do not consider ourselves experts in this field and believe that a different development team would be more suited for this task.
We took the future addition of this component into consideration when designing the system. This is one of the reasons why we decided to have our WebServer Backend exposed using the universal REST API. If Sieve were to develop this feature, it could be easily integrated into the existing infrastructure.

Appendix

A. Glossary

Term	Description
Adobe XD	User experience design-tool by Adobe to create mockups and test them on devices e.g. phone.
Angular	Frontend framework to build mobile and desktop web applications
ApacheBench	For http server benchmarking, in our case speed testing the AdminWebsite
API	An **A**pplication **P**rogramming **I**nterface is a program part that is made available by a software system to other programs for connection to the system.
APK	"Is the package file format used by the Android operating system for distribution and installation of mobile apps, mobile games and middleware"
AppMonsta	API for global app store data
Chai	Assertion library for Node.js, can be paired with any test framework.
CRUD	**C**reate data record, **R**ead data set, **U**pdate record, **D**elete data record.
CSS	**C**ascading **S**tyle **S**heets is a stylesheet language for electronic documents and one of the core languages of the World Wide Web.
CSV	**C**omma-**S**eparated **V**alues is a file format and describes the structure of a text file.
Digital Ocean	US cloud infrastructure provider often used for the simple setup of cloud services.
Docker	Free software that offers OS-level virtualization to e.g. run a Node.js on a local machine.
Droplet	Different types of virtual machines by Digital Ocean, hosted in one of their datacentres.
Expo	Toolchain built around React Native that helps develop an app quickly.
GDPR	The **G**eneral **D**ata **P**rotection **R**egulation is a legal framework for persons living in the European Union. It lays down guidelines of personal data of individuals.
GitLab	Project- and source code management software.
HTML	The **H**yper**T**ext **M**arkup **L**anguage is a text-based markup language for structuring electronic documents such as texts with hyperlinks, images and other content.
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol is a stateless protocol for transmitting data on the application layer over a computer network.
IDE	Integrated Development Environment is an application used for developing software. It offers a source code editor, build automation tools, a debugger, source control integration and more.
ISO	The **I**nternational **O**rganization for **S**tandardization is the international association of standardisation organisations.
JavaScript	Programming language mostly used for Web-development.

Jest	Is a JavaScript Testing Framework with focus on simplicity.
Jira	Project management tool by Atlassian.
JSON	**J**ava**S**cript **O**bject **N**otation is a compact data format in an easily readable text form and serves the purpose of data exchange between applications.
Mocha	JavaScript test framework for Node.js programs.
MongoDB	Object oriented database.
Native Base	React Native framework to build React Native apps. Offering cross-platform UI-components.
NFR	**N**on-**F**unctional **R**equirements is a requirement that lays down criteria by which the operation of a system, rather than a specific behaviour, can be evaluated.
Node.js	Server side platform developed by Google, most often used for Webservers.
OS	**O**perating **S**ystem
React	JavaScript library for building user interfaces.
React Native	A framework for building native apps using React.
REST-API	**RE**presentational **S**tate **T**ransfer-**A**pplication **P**rogramming **I**nterface Programming interface that explains the behaviour between the client and server.
SLA	A **S**ervice **L**evel **A**greement is a basic agreement between the client and the service provider for recurring services.
SQL	The **S**tructured **Q**uery **L**anguage is a database language for defining data structures in relational databases and for editing (inserting, changing, deleting) and querying databases based on them.
UI	The **U**ser **I**nterface is a form of user interface of a computer.
URL	A **U**niform **R**esource **L**ocator identifies and locates a resource via the access method to be used and the location of the resource in computer networks.
VM	A **V**irtual **M**achine is based on a computer architecture and offer the functionality of a physical computer.

B. References

[1] C. Wei Jie, “Sieve Slide Deck,” Mail, 2019.

[2] M. Rehkopf, “User Stories with Examples and Template,” [Online]. Available: https://www.atlassian.com/agile/project-management/user-stories. [Accessed 19 05 2020].

[3] ISO/IEC, 9126, Software engineering - Product quality, 2001.

[4] E. Elliott, “How Popular is JavaScript in 2019?,” Medium, 11 05 2019. [Online]. Available: https://medium.com/javascript-scene/how-popular-is-javascript-in-2019-823712f7c4b1. [Accessed 19 05 2020].

[5] Atul, “When should I use Node JS for backend ?,” Medium, 5 11 2016. [Online]. Available: https://medium.com/@atulanand94/when-should-i-use-node-js-for-backend-8379d6b224d2. [Accessed 19 05 2020].

[6] N. Pandit, “What and Why React.js,” 05 03 2020. [Online]. Available: https://www.c-sharpcorner.com/article/what-and-why-reactjs/. [Accessed 19 05 2020].

[7] «https://github.com/microsoft/react-native-windows,» [Online].

[8] Expo, [Online]. Available: https://docs.expo.io/.

[9] MongoDB, Inc., “Why and When to Use MongoDB,” [Online]. Available: https://www.mongodb.com/why-use-mongodb.

[10] JetBrains s.r.o, [Online]. Available: https://www.jetbrains.com/webstorm/.

[11] GitLab Inc., [Online]. Available: https://about.gitlab.com/.

[12] Postman, Inc., [Online]. Available: https://www.postman.com/.

[13] Docker, Inc., [Online]. Available: https://www.docker.com/.

[14] 3T Software Labs, [Online]. Available: https://robomongo.org/.

[15] Apache Software Foundation, “ab - Apache HTTP server benchmarking tool,” [Online]. Available: https://httpd.apache.org/docs/2.4/programs/ab.html. [Accessed 21 05 2020].

[16] D. Keller, Software Engineering 1, Rapperswil: HSR Script, 2019.

[17] Facebook Inc., “Testing,” [Online]. Available: https://reactnative.dev/docs/testing-overview. [Accessed 21 05 2020].

[18] “AppMonsta,” 14 05 2020. [Online]. Available: https://appmonsta.com/?utm_campaign=Is-there-any-official-API-to-get-app-details-from-the-Play-Store&utm_medium=answer&utm_source=quora. [Accessed 19 05 2020].

[19] EU General Data Protection Regulation, «GDPR.EU,» 2020. [Online]. Available: https://gdpr.eu/compliance/.

C. Table of Figures

Figure 1 - Architecture diagram - overview 2
Figure 2 - Related technologies 3
Figure 3 - Analysis of existing products [1] 7
Figure 4 - First screens of the app [1] 8
Figure 5 - Use Case Diagram 12
Figure 6 - Domain model 18
Figure 7 - System seuence diagram S01 20
Figure 8 - Mockup Login / Signup 21
Figure 9 - Mockup / MobileApp News-Screen 22
Figure 10 - Mockup / MobileApp Categories Screen 22
Figure 11 - Mockup / MobileApp Categories Detail Screen 23
Figure 12 - Mockup / MobileApp Privacy Policy Screen 23
Figure 13 - Mockup / MobileApp Dashboard Screen 24
Figure 14 - Mockup Screenflow 25
Figure 15 - Mockup - Privacy Policy Screen - View 26
Figure 16: AdminWebsite - App Screen - View 26
Figure 17: Mockup - Privacy Policy Screen – Edit Data Type Collected 27
Figure 18: Mockup - Privacy Policy Screen - Edit Data Usage Policy 27
Figure 19: AdminWebsite - App / Privacy Policy Screen - Edit 28
Figure 20 - System overview 30
Figure 21 - Deployment diagram 34
Figure 22 - Automated deployment 35
Figure 23 - Database model 36
Figure 24 - Size and Performance 37
Figure 25: Benchmarking of 1000 concurrent requests of /apps on WebServer 38
Figure 26: Unit Test success 41
Figure 27: Snapshot test success 42
Figure 28 - Resolution bug 45
Figure 30: Time evaluation, invested time per project phase ..... **Fehler! Textmarke nicht definiert.**
Figure 31: Time evaluation, invested time per team member ..... **Fehler! Textmarke nicht definiert.**

D. Table List

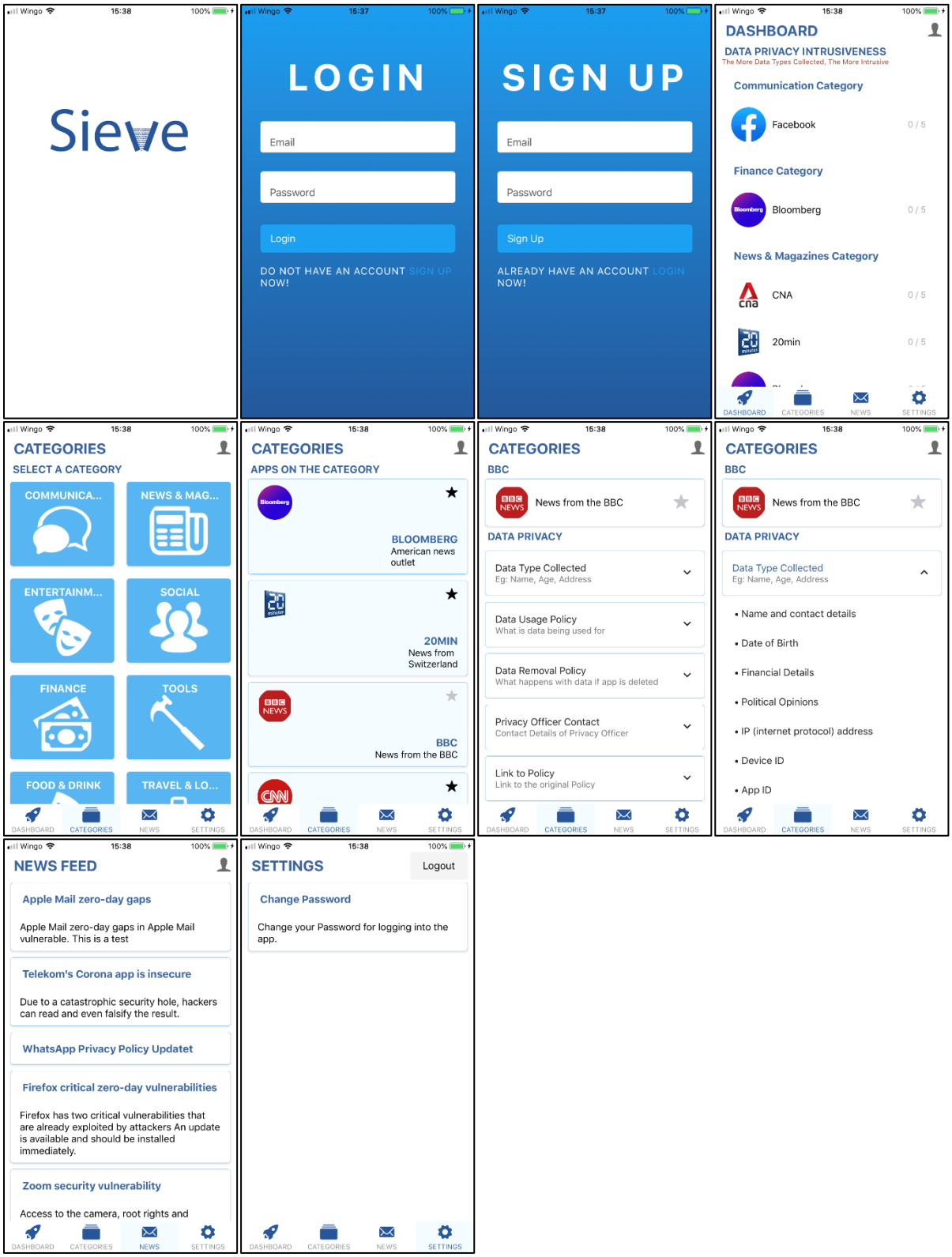
Table 1 - Use Case overview 10
Table 2 - Optional Features overview 11
Table 3 - Domain model description 19
Table 4 - API routes 33
Table 5 - Metric analysis – MobileApp 39
Table 6 - Metric analysis – AdminWebsite 39
Table 7 - Metric analysis – WebServer 39
Table 8 - Metric analysis – entire project 40
Table 9 - Test Case evaluation 44
Table 10 - Compliance with the functional requirements (use cases) 46
Table 11 - Compliance with the functional requirements (optional features) 47
Table 12 - Compliance with the Non-functional requirements 48

E. Project Plan

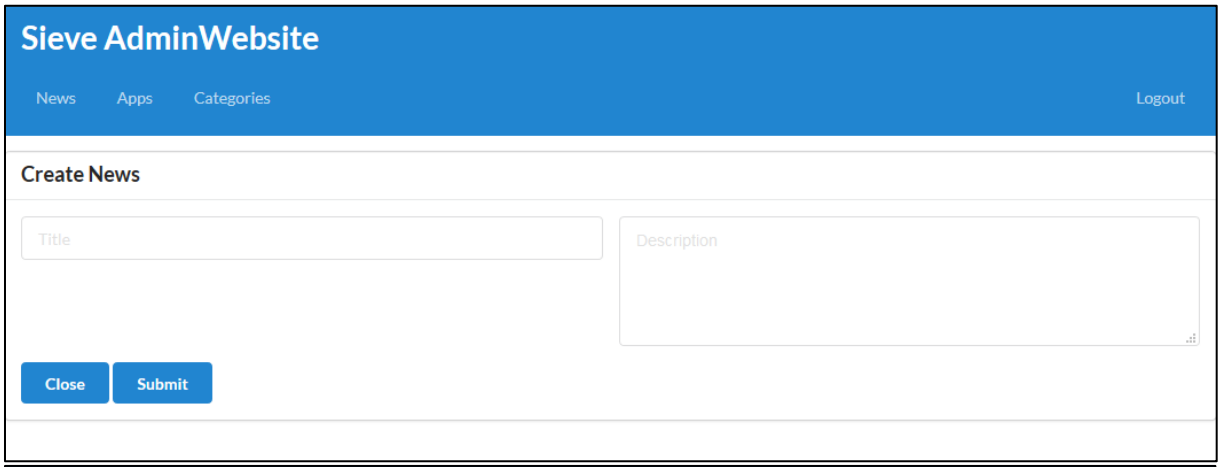
- Projektplan: 01_Projectplan/Projectplan.pdf
- Technical Risks: 01_Projectplan/TechnicalRisks.pdf

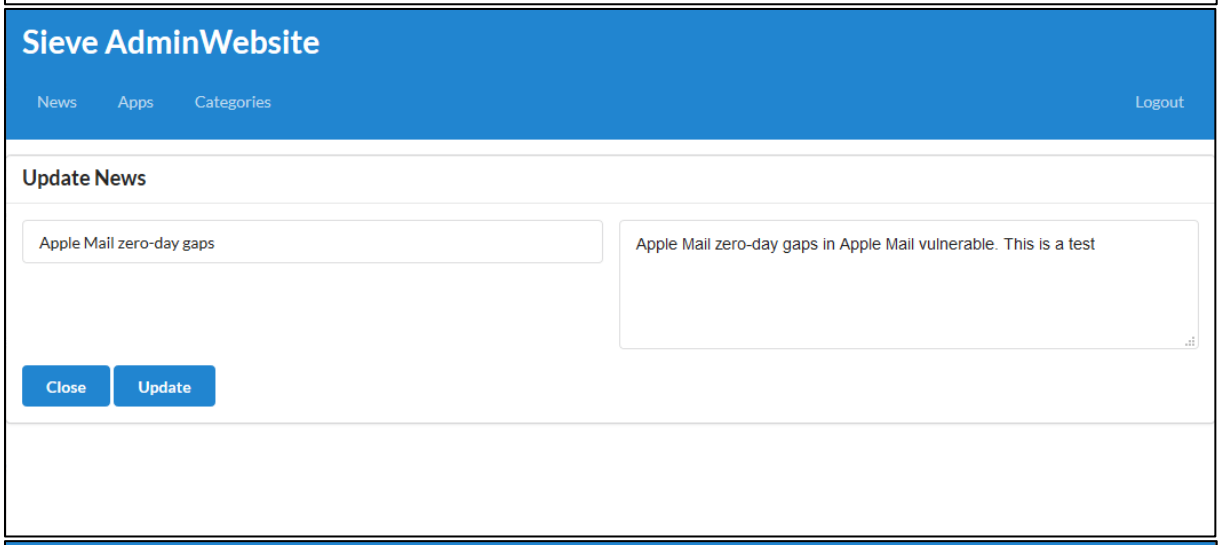
F. Final Application

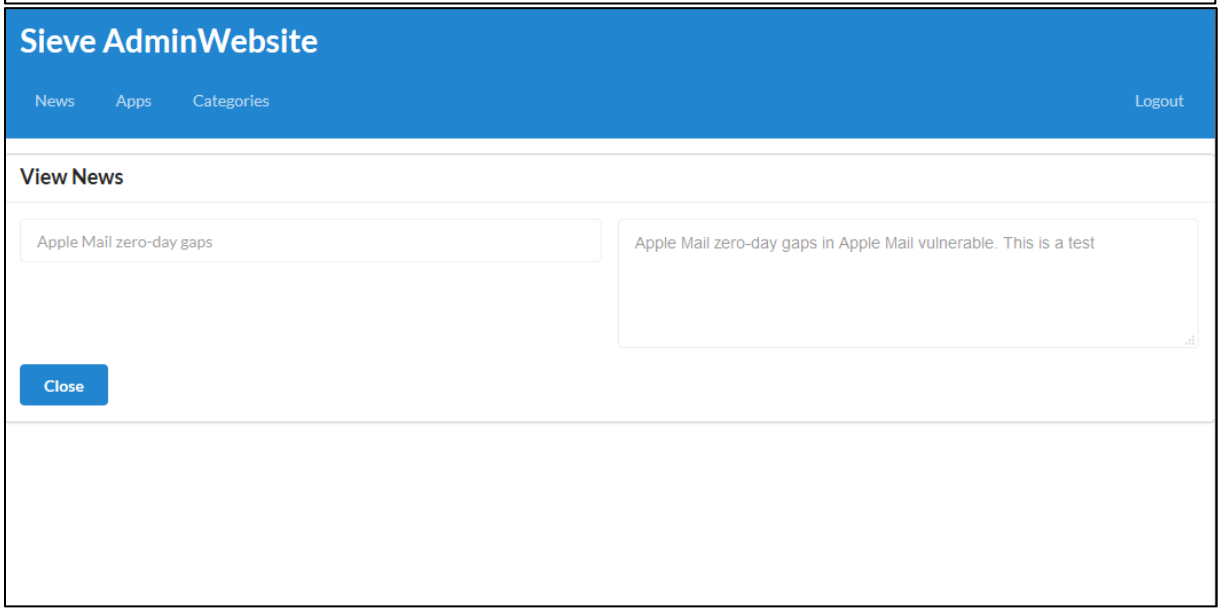
Screenshots of the MobileApp

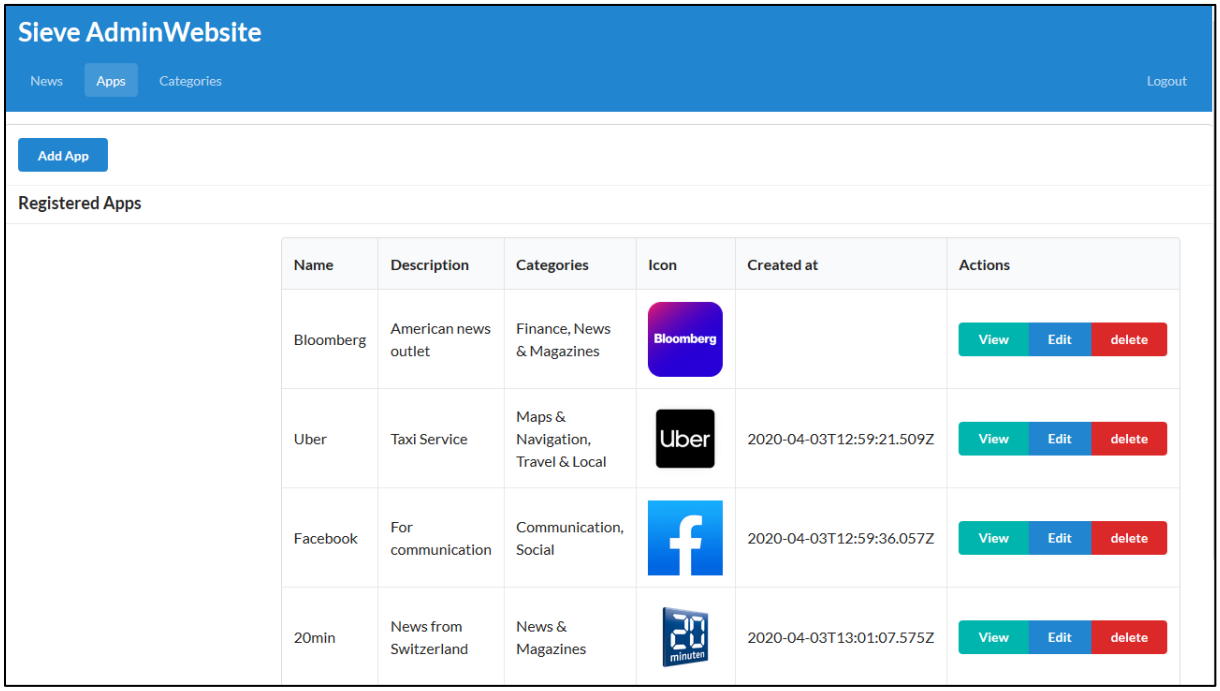


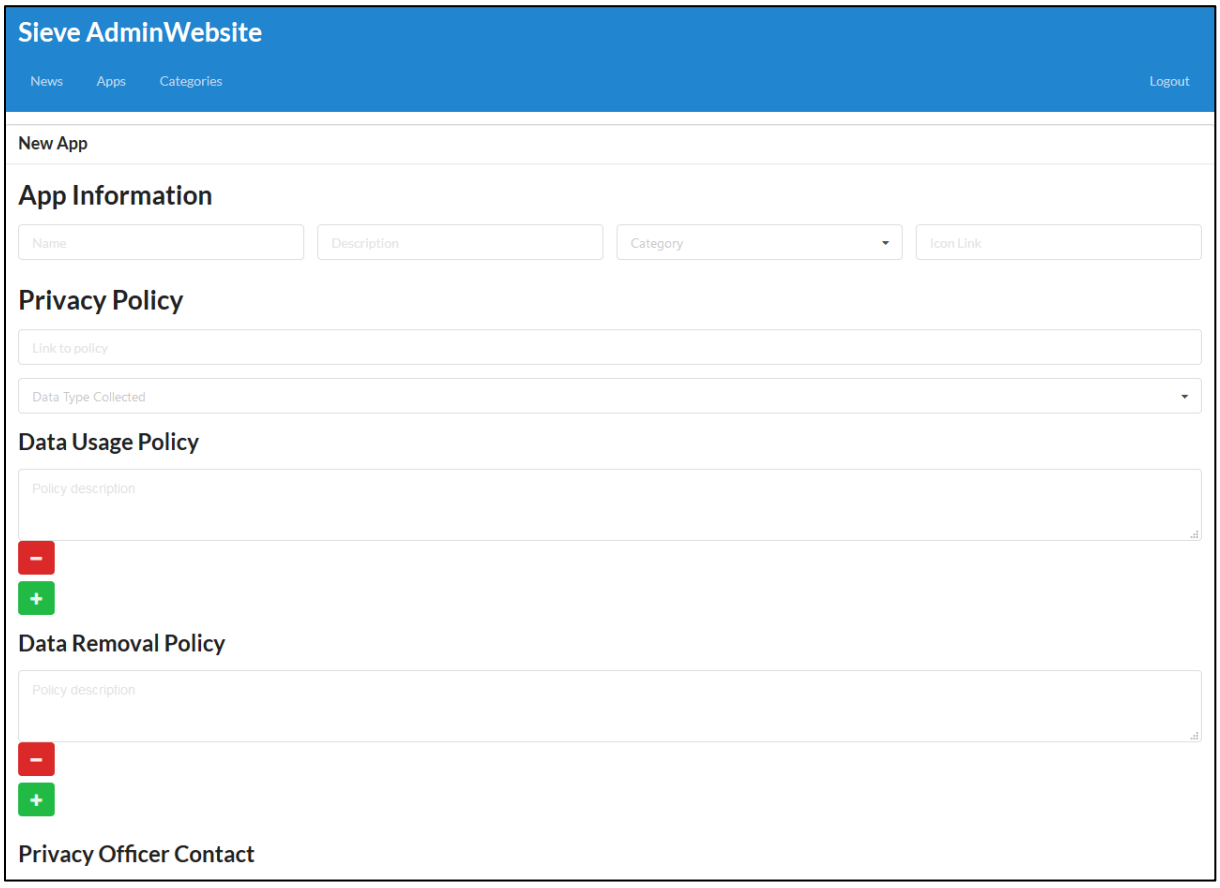
Screenshots of the AdminWebsite

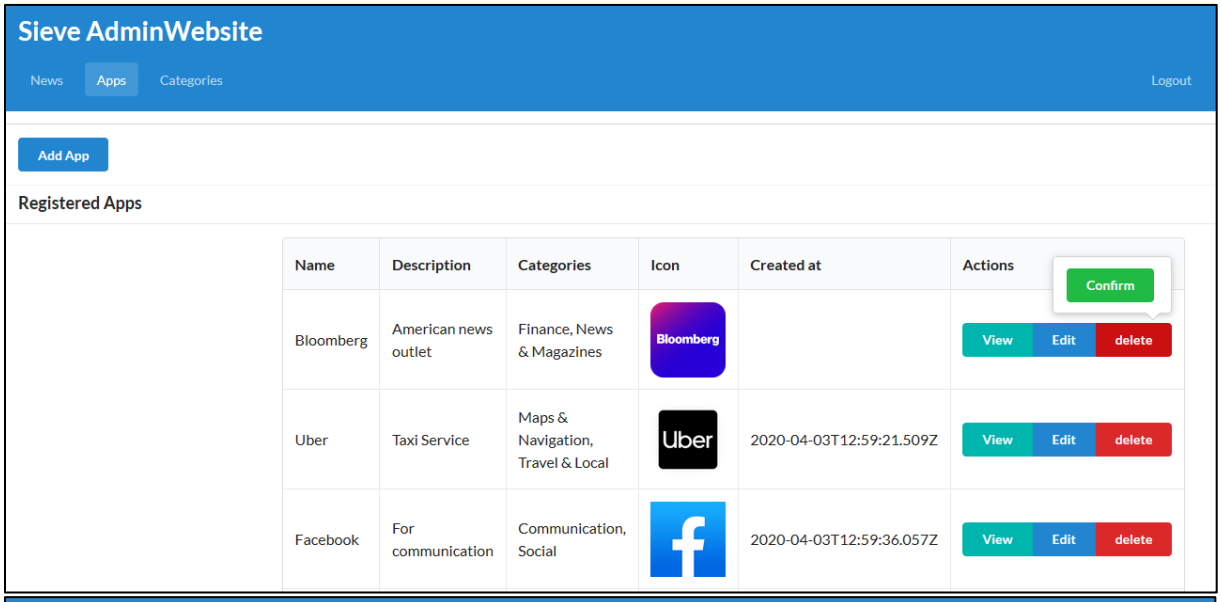


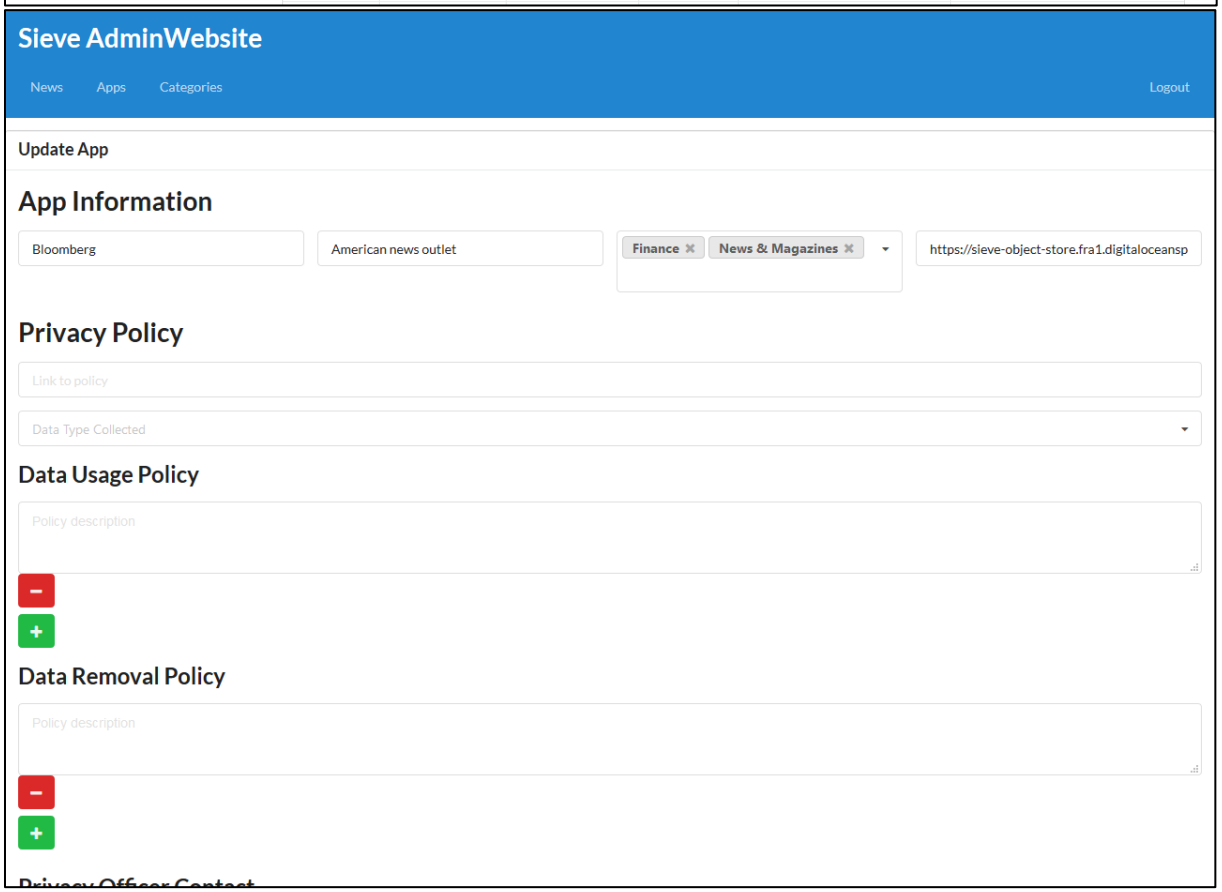


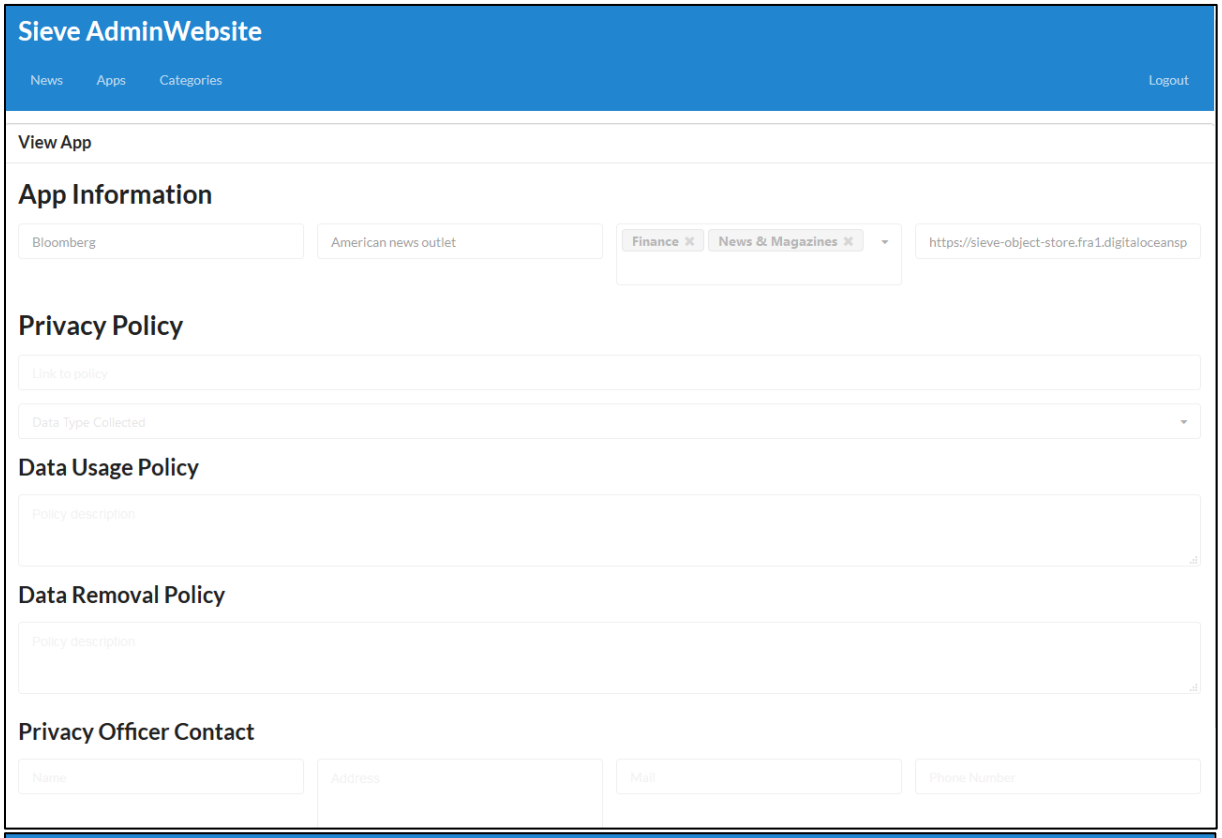


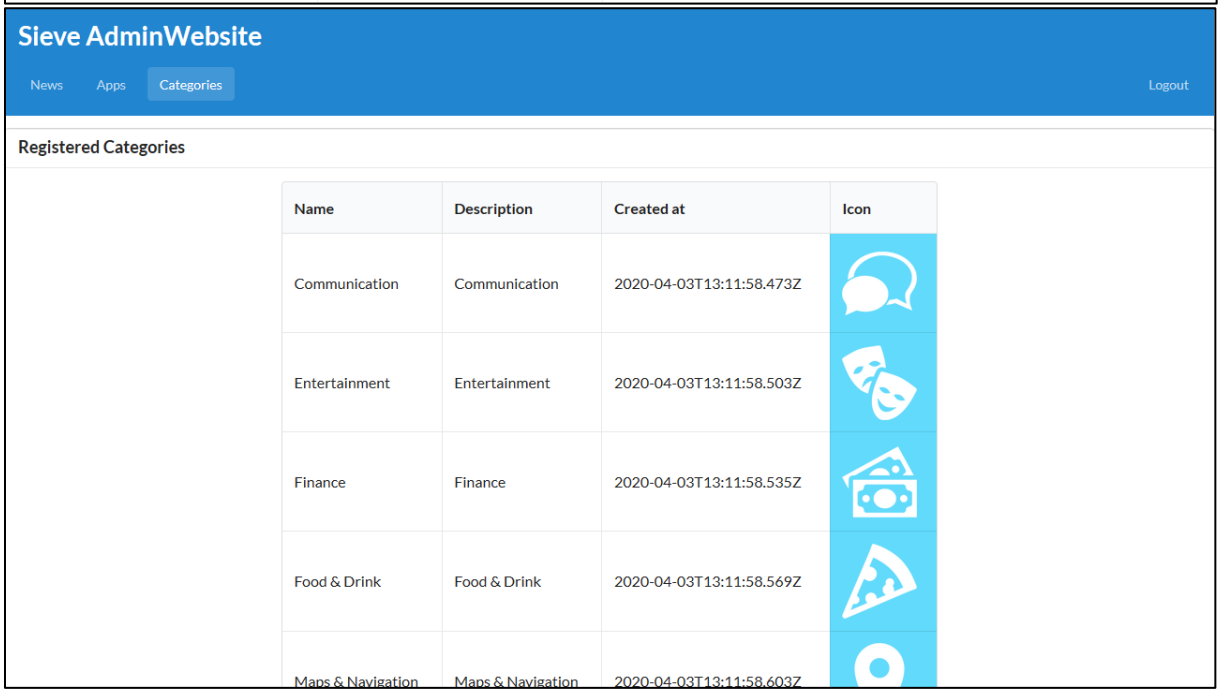

Sieve AdminWebsite
News
Apps
Categories
Logout
Add App
Registered Apps
Name
Description
Categories
Icon
Created at
Actions
Bloomberg
American news outlet
Finance, News & Magazines
View
Edit
delete
Uber
Taxi Service
Maps & Navigation, Travel & Local
2020-04-03T12:59:21.509Z
Facebook
For communication
Communication, Social
2020-04-03T12:59:36.057Z
20min
News from Switzerland
News & Magazines
2020-04-03T13:01:07.575Z

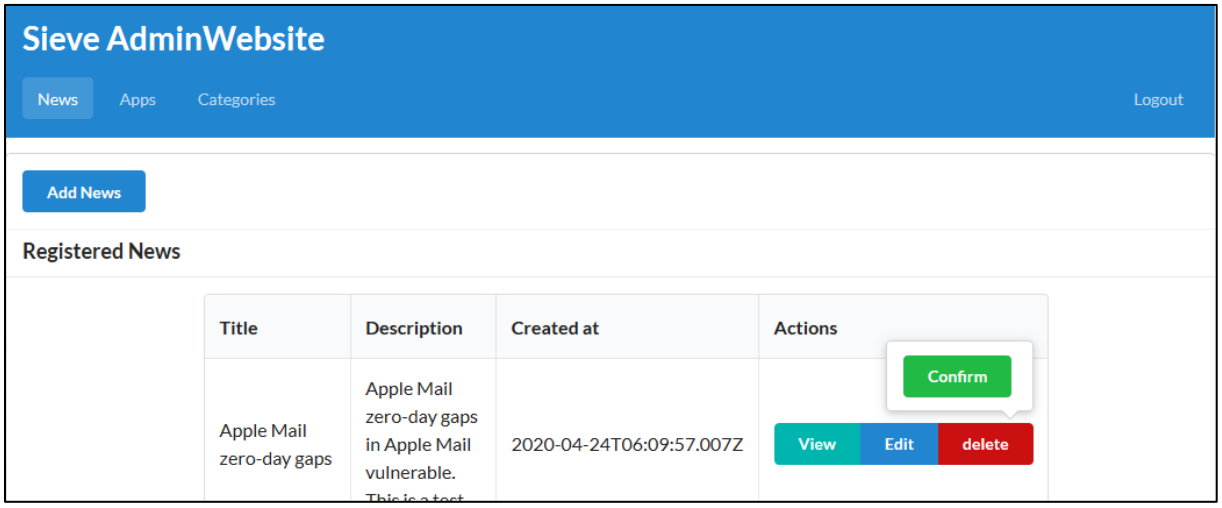

Sieve AdminWebsite
News
Apps
Categories
Logout
New App
App Information
Name
Description
Category
Icon Link
Privacy Policy
Link to policy
Data Type Collected
Data Usage Policy
Policy description
Data Removal Policy
Policy description
Privacy Officer Contact

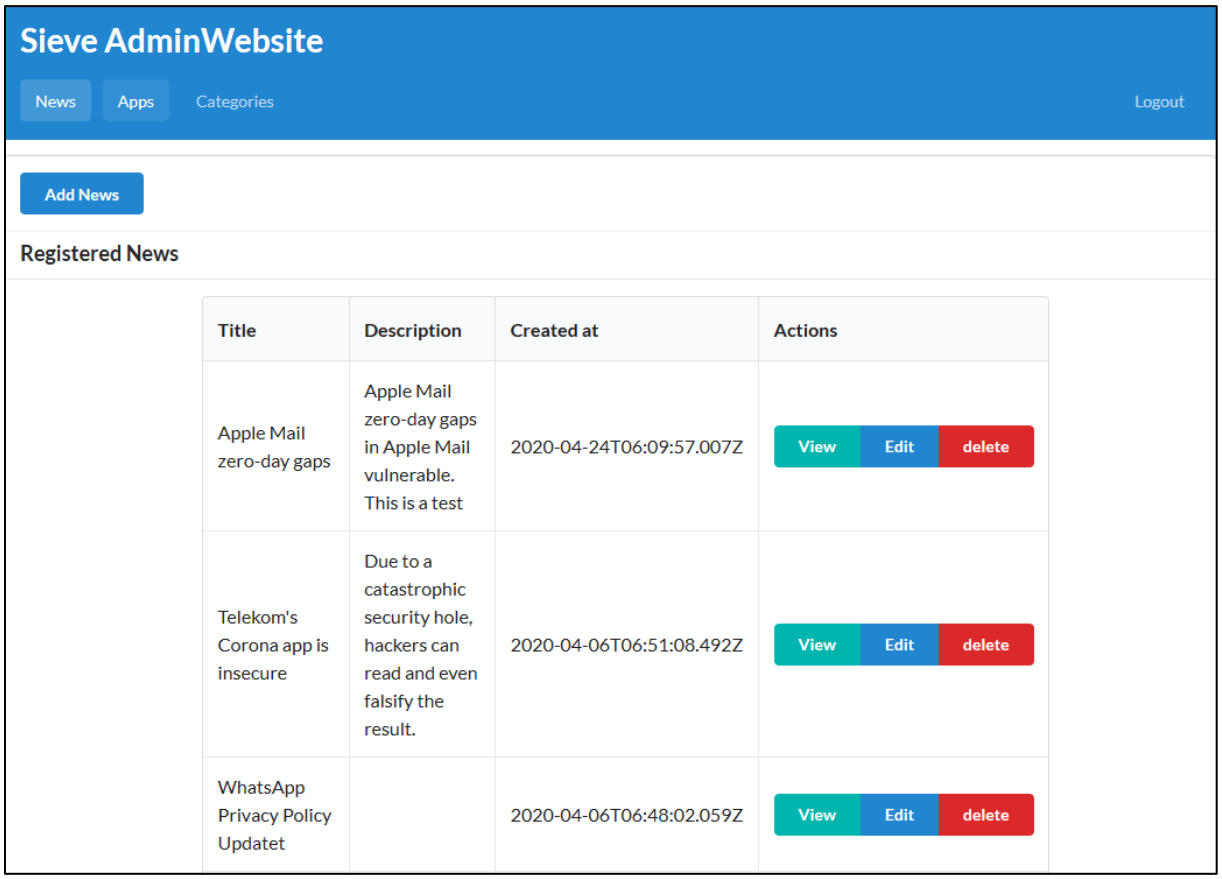

Sieve AdminWebsite
News
Apps
Categories
Logout
Add App
Registered Apps
Name
Description
Categories
Icon
Created at
Actions
Confirm
Bloomberg
American news outlet
Finance, News & Magazines
Bloomberg
View
Edit
delete
Uber
Taxi Service
Maps & Navigation, Travel & Local
Uber
2020-04-03T12:59:21.509Z
View
Edit
delete
Facebook
For communication
Communication, Social
2020-04-03T12:59:36.057Z
View
Edit
delete


Sieve AdminWebsite
News
Apps
Categories
Logout
Update App
App Information
Bloomberg
American news outlet
Finance
News & Magazines
https://sieve-object-store.fra1.digitaloceansp
Privacy Policy
Link to policy
Data Type Collected
Data Usage Policy
Policy description
Data Removal Policy
Policy description


Sieve AdminWebsite
News
Apps
Categories
Logout
View App
App Information
Bloomberg
American news outlet
Finance
News & Magazines
https://sieve-object-store.fra1.digitaloceansp
Privacy Policy
Link to policy
Data Type Collected
Data Usage Policy
Policy description
Data Removal Policy
Policy description
Privacy Officer Contact
Name
Address
Mail
Phone Number


Sieve AdminWebsite
News
Apps
Categories
Logout
Registered Categories
Name
Description
Created at
Icon
Communication
Communication
2020-04-03T13:11:58.473Z
Entertainment
Entertainment
2020-04-03T13:11:58.503Z
Finance
Finance
2020-04-03T13:11:58.535Z
Food & Drink
Food & Drink
2020-04-03T13:11:58.569Z
Maps & Navigation
Maps & Navigation
2020-04-03T13:11:58.603Z


Sieve AdminWebsite
News
Apps
Categories
Logout
Add News
Registered News
Title
Description
Created at
Actions
Apple Mail zero-day gaps
Apple Mail zero-day gaps in Apple Mail vulnerable.
2020-04-24T06:09:57.007Z
Confirm
View
Edit
delete


Sieve AdminWebsite
News
Apps
Categories
Logout
Add News
Registered News
Title
Description
Created at
Actions
Apple Mail zero-day gaps
Apple Mail zero-day gaps in Apple Mail vulnerable. This is a test
2020-04-24T06:09:57.007Z
View
Edit
delete
Telekom's Corona app is insecure
Due to a catastrophic security hole, hackers can read and even falsify the result.
2020-04-06T06:51:08.492Z
View
Edit
delete
WhatsApp Privacy Policy Updatet
2020-04-06T06:48:02.059Z
View
Edit
delete

G. Optional Features

Here is a list of the optional features for each optional feature there is a small description including the affected components.

General

OF 1 - Implement the App GDPR compliant
Handle the data so the applications are GDPR compliant[3]. This could leave a better impression on the interests of privacy policy if an app about privacy policy is GDPR compliant.

Affected components: MobileApp, WebServer, AdminWebsite

MobileApp

OF2 - Login screen - login with Facebook and Google
Add the Facebook and Google Authentication service to the MobileApp, so that users can automatically login with Facebook and Google without a signup process.

Affected components: MobileApp

OF3 - Login screen - Sign Up with first name and surname
Add the first- and last name as fields to the MobileApp Signup form. This allows the administrator to send out personalized newsletters.

Affected components: MobileApp, WebServer

OF4 - Category screen – Search
Search field on the Category screen to search all the apps. As the usability tests have shown, users would like to have a search functionality instead of having to search the app by clicking through the categories.

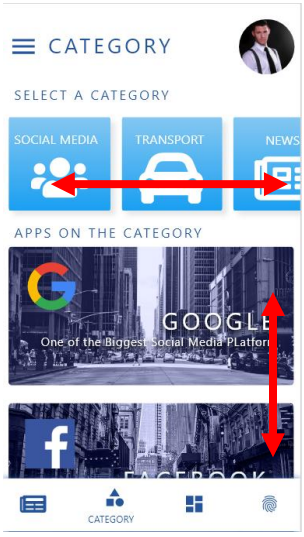
Affected components: MobileApp, WebServer

OF5 - Category screen - similar apps
When searching for apps, similar apps with for example a better privacy policy rating should be displayed. This requires OF4 to be implemented. It would also help to compare privacy policies of different apps, by making it easier to add them to the favourites.

Affected components: MobileApp, WebServer

[3] https://gdpr.eu/compliance/

OF6 - Category screen - navigate with side wiping
Scrolling the categories sidewards and display apps, without changing the screen. This way more information can be displayed on a single screen. When a category is clicked on, the corresponding apps are displayed directly below in the same screen.



Affected components: MobileApp

OF7 - Category screen - Images at Apps
Display background images to an app that is shown on the category screen. This makes it possible for the administrator to set the background image for every app. This could be a picture of the company's headquarters, a similar picture per category or a picture that the customer associates with the company.

Affected components: MobileApp, WebServer, AdminWebsite, ObjectStore

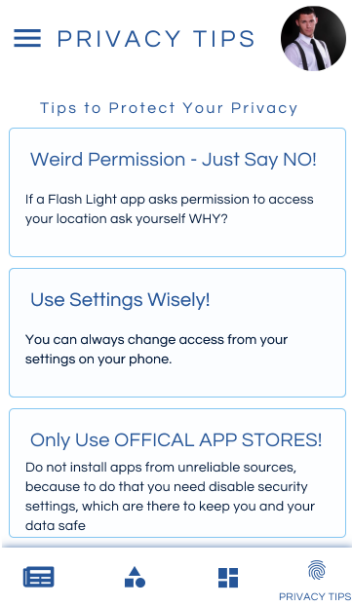
OF8 - Category screen - Apps with rating
Show a calculated rating of all the apps in a category. This allows the customer to see the rating without having to open the app details. This means that he does not have to open the apps that do not comply with his standard.

Affected components: MobileApp, WebServer

OF9 - Category screen – Different font size
Change the font size to a smaller one. React Native has default font sizes that are displayed correctly on iOS and Android devices with different sizes and resolutions. To set a different one, the font sizes would have to be set manually.

Affected components: MobileApp

OF10 - Privacy Tipps screen - create screen
Additional navigation point with separate screen. Like the news screen, just different information. Settings would be accessible by clicking on the icon in the top right corner.



Affected components: MobileApp, WebServer, AdminWebsite

OF11 - News screen - Policy Updates messages, with App Icon
Automatically show newly created or updated privacy policies on the news screen. This would make the app look more dynamic and more regularly maintained.

Affected components: WebServer

OF12 - News screen – Push messages for new entries
Automatically send push notifications to the users MobileApp as soon as a privacy policy has been registered or updated. This way the user can be motivated to open the app again after some time of not using it.

Affected components: MobileApp, WebServer, additional Expo service

OF13 - Settings screen - account data (first name, last name, email address) changeable
Allow a MobileApp-user to edit its first-, last name and email address on the settings screen. If the app should be GDPR compliant (Art. 17, Right to erasure[4]) described in OF1, a functionality to delete an account and all it's data would have to be added

Affected components: MobileApp, WebServer

OF14 - Settings screen - sorting adjustable
Sort the apps on the MobileApp according to the settings set by the user on the settings-screen. This makes it possible to customize the app even more to the user's preferences.

Affected components: MobileApp, WebServer

[4] https://gdpr-info.eu/art-17-gdpr/

OF15 - Settings screen – change password
To allow the user to change the password on the settings screen. Often users only change their password if they forgot it. Because a user has to be logged in to use the app, this would still not be possible. Therefore, it would make sense to implement a forgot password functionality first.

Affected components: MobileApp, WebServer

OF16 - Detail screen - Make chat with Data Privacy Officer look like a real chat
Display a chat-like feature on an apps details screen that behaves like a native chat-app. The communication with the privacy officer will still run through mail in the background because the privacy officer is not responding immediately.

Affected components: MobileApp, WebServer, additional mailserver

OF17 - Detail screen - Compare apps with side swiping
When looking at the privacy policy of an app and swiping left or right, the privacy policy of similar apps should be displayed for an easy comparison. This allows to easily compare sections of the privacy policy instead of just the rating.

Affected components: MobileApp, WebServer

OF18 - Detail screen - Privacy Policy key points highlighted
Show privacy policy entries that the administrator preselected highlighted. This way, the administrator can draw a user's attention to a specific privacy policy entry.

Affected components: MobileApp, WebServer, AdminWebsite

OF19 – MobileApp – Navigation order
Reorganize the order of the Navigation-points. Switch News <-> Dashboard and show the Dashboard screen when a user logs in.

Affected components: MobileApp

OF20 – MobileApp – Landscape mode
Implement the Landscape mode for all screens of the App. Depending on the size of the device and its resolution, the placement of the tiles would have to be changed as well.

Affected components: MobileApp

OF21 - Publish the MobileApp to the Google App Store
Fill out all the administrative forms to publish the app on the Google App store. This includes building the app with all the necessary files like screenshots, keys, etc. for the store. Creating an account and publishing the app there.

OF22 - Publish the MobileApp in the Apple App Store
Fill out all the administrative forms to publish the app on the Apple App Store. Code would most certainly have to be changed in the process to be added to the store because the specifications are very strict.[5]

[5] https://developer.apple.com/app-store/review/guidelines/

AdminWebsite

OF23 - AdminWebsite - Import of data with CSV
Allow the administrator to add datasets by importing CSV-files. This makes it easier to import multiple privacy policies at the time.

Affected components: AdminWebsite

OF24 - AdminWebsite - Admin can add or change fields independently
The administrator could add new data type collected fields to be selected on the creation of privacy policies. At the moment, they are limited to the ones predefined by the customer.

Affected components: AdminWebsite, WebServer, MobileApp

OF25 - AdminWebsite - Category View
Show all the existing categories on the AdminWebsite. This way the administrator has an overview of all the categories including all the icons. It can be extended to include the CRUD functionality.

Affected components: AdminWebsite

OF26 - AdminWebsite - Categories CRUD
Add a CRUD for the categories to the AdminWebsite. The deletion of categories would have to be implemented carefully, so no categories that still have apps assigned can be deleted. Especially with the search functionality (OF 4) not implemented, otherwise it's possible that an app can not be found on the MobileApp because it's in no category.

Affected components: AdminWebsite, WebServer

OF27 - AdminWebsite - Automatically download Apps Icon from App Store
As soon as a new app is being created, the icon will be downloaded from the app store automatically. This way, the admin would not have to upload icons to the Digital Ocean Spaces manually.

Affected components: AdminWebsite, WebServer

OF28 - AdminWebsite - Automatically download Apps description from App Store
As soon as a new app is being created, its information will be downloaded from the app store automatically. The problem is that the tools that offer this service are not working in cooperation with the app stores so the services could unexpectedly stop working.

Affected components: AdminWebsite, WebServer

OF29 - AdminWebsite - Data Type Collected points (1 or 5) Changeable
Allow the administrator to set the points for the data type collected manually. Allows different datatype collected to have different weights for the app rating.

Affected components: AdminWebsite, WebServer

OF30 - AdminWebsite - Change data structure
Change the data structure of the privacy policy to allow different categorization of the entries (e.g. subtitles).

Affected components: AdminWebsite, WebServer, MobileApp

OF31 - AdminWebsite – Filter apps by category
To only see the apps of one category, a dropdown would be added above the table. The question is how apps with multiple categories would be shown or whether it would be possible to filter by multiple categories.

Affected components: AdminWebsite, WebServer

OF32 - Transfer all through with https
Change the transfer protocol from http to https. An SSL certificate is required for this step.

Affected components: AdminWebsite, WebServer, MobileApp

OF33 - Digital Ocean billing-API to show cost
Show the cost for running the services on Digital Ocean on the AdminWebsite. Digital Ocean offers an API for this. This way the administrator would have a simple overview of the monthly costs for running the service every time he logs in to change some content.

Affected components: AdminWebsite

OF34 - Ordering of tables by row
Allow the administrator to order all the entries in tables by row. It will not be possible to order every row like e.g. categories because there sometimes is more than one entry.

Affected components: AdminWebsite, WebServer

OF35 - Split displayed entries into multiple pages (e.g. apps)
If there are a lot of entries in a table, split them into multiple pages. Like the Google-page where one can change to the next page on the bottom. For this to make sense, a lot of entries would have to be in the database and if there are, there would be a lot of pages to go through when looking for an entry. Therefore, a search functionality would have to be implemented as well.

Affected components: AdminWebsite, WebServer

OF36 – Dashboard optimization
To help the user better understand the rating, it is supplemented with a text below the screen title. Additionally, the score should be displayed as "Score number/ maximum score number" (e.g. 1/5) instead of a fixed value 0...5.

Affected components: MobileApp

H. Developer Documentation

- Developer Documentation: 04_Developer/Developer_Documentation.pdf

I. API Documentation

- API Documentation: 04_Developer/API/API_Documentation.pdf

J. Feasibility Study Traffic Analysis

- Feasibility Study Traffic Analysis: 02_Analysis/Feasability_Study_Traffic_Analysis.pdf

K. React Native UI Libraries Comparision

- React Native UI Libraries Comparison: 02_Analysis/React_Native_UI_Libraries_comparison.pdf

L. Sample Privacy Policy

The following data extract of a privacy policy was provided to us by the customer:

Overview of Privacy Policy Analysis

We analysed the privacy policies of 5 major news companies:

1. British Broadcasting Corporation (BBC)
2. Channel NewsAsia (CNA)
3. Cable News Network (CNN)
4. Financial Times (FT)
5. The Straits Times

We break down each privacy policy into 5 segments:

1. Data Type Collected
2. Data Usage Policy
3. Data Removal Policy
4. Privacy Officer Contact
5. Link to Policy

I was thinking whether we can implement a scroll feature for each segment of the privacy policy if we cannot fit everything into the mobile phone's screen.

BBC Privacy Policy

Data Type Collected

- Name and contact details
- Date of Birth
- Financial Details
- Political Opinions
- IP (internet protocol) address
- Device ID
- App ID
- Vendor ID
- Advertising ID
- Information on activities outside BBC when talk about us, articles read, programmes watched

Data Usage Policy

- to deal with your requests, complaints and enquiries
- to personalise services and give you things more tailored to your tastes
- to show you relevant advertising on another company's site
- to research and innovate
- to contact you about various things
- Keep in mind that when you give us personal information it could be being transferred, stored or processed in a location outside the UK and the EEA.

Data Removal Policy

- we only hold your information for as long as we do the activities we told you about or have a valid reason to keep it
- we think about what type of information it is, the amount collected, how sensitive it might be and any legal requirements
- we design our services so that we don't hold your information any longer than we have to

Privacy Officer Contact

- Email: dpa.officer@bbc.co.uk
- Address
  BBC DPO,
  BC2 A4,
  201 Wood Lane,
  London W12 7TP

Link to Policy

- https://www.bbc.co.uk/usingthebbc/

CNA Privacy Policy

Data Type Collected

No information (Give highest penalty)

Data Usage Policy

- To send notices, information, promotions and updates including marketing and advertising materials
- To provide goods and services
- To verify and process payment
- For verification and record of your personal particulars
- To conduct statistical analysis of the users of the Mediacorp Services
- To comply with any request from any third party or any order of court or directive from authorities investigating any alleged offence, misdeed and/or abuse or for the purposes of taking legal action against any users
- We will only share your personal details with a third party for the purposes and situations as described above, or for other purposes where we have your consent to do so.

Data Removal Policy

- In the event that you wish for us to discontinue the collection, use, disclosure and/or processing of your personal data for any of the above purposes, you may also write in to us at dpo@mediacorp.com.sg to inform us of the same.
- Whenever we send you any marketing information, we will include instructions on how to unsubscribe.
- You may update your personal information at any time by accessing your account(s) with us or by writing in to us at dpo@mediacorp.com.sg to inform us of the same.

Privacy Officer Contact

- **Phone**: (65) 6333 3888
- **Email**: dpo@mediacorp.com.sg
- **Address**
  Mediacorp Campus
  1 Stars Avenue
  Singapore 138507
  (Attention: Data Protection Officer)

Link to Policy

https://www.mediacorp.sg/en/privacy-policy-5933440

CNN Privacy Policy

Data Type Collected

- We receive Information that you provide to us directly, such as registration Information, and Information that is collected automatically, such as Information collected from your browser or device.
- From Third Parties. We also receive some Information about you from third parties.

Data Usage Policy

- We share your Information:
- With service providers we have engaged to perform business-related functions on our behalf.
- In response to legal process
- With third parties if this will help us to enforce our policies and terms of use,
- If we, or one of our business units, undergoes a business transition
- With your consent, we share Information with other organisations so they can market their products and services to you.

Data Removal Policy

- You have the right to ask us for a copy of your Information, to correct, delete or restrict (stop any active) processing of your Information
- In addition, you can use our data subject request form or object to the processing of your Information in some circumstances (in particular, where we do not have to process the Information to meet a contractual or other legal requirement).
- Where you have given consent to our using your Information for marketing, you can withdraw your consent at any time.
- You can also ask us not to carry out profiling for direct marketing purposes.
- You can opt out of receiving email newsletters and other marketing communications by following the opt-out instructions provided to you in those emails.
- You also have a right to complain to a supervisory authority for data protection in the country where you live

Privacy Officer Contact

Privacy Policy Coordinator

- privacy.cnn@turner.com
- E-mail Administration
  Turner Broadcasting System, Inc.
  Attention: Privacy Policy Coordinator
  1050 Techwood Drive NW
  Atlanta, GA 30303

Data Protection Officer

- Email: DPO@turner.com
- **Address**
  Data Protection Officer
  Turner Broadcasting System, Inc.
  1050 Techwood Drive NW
  Atlanta, GA 30318

Link to Policy

https://edition.cnn.com/2014/01/17/cnn-info/privacy-policy/index.html

Financial Times Privacy Policy

Data Type Collected

- Basic personal details – such as name, title and date of birth;
- Personal contact details – such as home address, delivery address, home phone number, mobile phone number, and email address;
- Work details – such as job title, department, company name, company address, work email address and office phone number;
- Log in details – such as username and password;
- Payment details – such as your bank account and payment card information.
- Your opinion of our products and services;
- Your preferences, including newsletters or myFT topics;
- Your investment details if you use our portfolio tool to store your holdings;
- Your career details and CV if you use exec-appointments.com or our other recruitment pages;
- Information such as your dietary preferences and accessibility requirements when attending an FT Live event, which may be considered special categories of personal data.
- Your location
- Your usage
- Your device
- Your engagement with advertisements

Data Usage Policy

- **Purpose**
- To provide our service
- To improve and maintain performance
- To bill our customers
- To monitor compliance with our policies and terms
- To personalise our products and services
- To communicate product changes and offers
- For the purpose of advertising
- Recruitment
- To build directories
- To administer prize draws and competitions
- **Sharing**
- We disclose personal information to facilitate the running of our business or to provide specific services you have requested.
- Service providers
- Advertisers
- Institutional customers
- Recruiters
- Social media providers
- Other users
- FT group

- Legal processes and successors in title

Data Removal Policy

- Billing information: For regulatory reasons, we are required to retain accounting records for 7 years
- Business intelligence: To improve and maintain performance, personalise our services and to provide our product
- Comments: To enable our readers to see other readers' comments about our content
- Customer service queries: To provide a case history to assist us to answer your queries
- Communication history: So that we have a record of what communications we have sent to you, to better assist your queries
- Event registration: For event administration and, with your permission, to invite you to future events.
- Market research: To administer market research incentives, avoid duplicating research requests and to understand research responses against usage of our Sites.
- Site features: To improve your experience of our products and services and to enable you to continue accessing site features.
- Subscription information: To enable us to fulfil your contract with us.

Privacy Officer Contact

- Email: Privacy.Officer@ft.com.
- **Address**
  The Financial Times Limited
  Registered Office: Bracken House, 1 Friday Street, London, EC4M 9BT, United Kingdom
  Registered Number: 00227590

Link to Policy

https://help.ft.com/help/legal-privacy/privacy/

The Straits Times Privacy Policy

Data Type Collected

- your name, NRIC, passport or other identification number, telephone number(s), mailing address, email address and any other information relating to you which you have provided us in any forms you may have submitted to us, or in other forms of interaction with you;
- information about your use of SPH's website and services
- your employment history, education background, and income levels;
- your payment related information, such as your bank account or credit card information, and your credit history; and
- information about your usage of and interaction with our website and/or services including computer and connection information, device capability, bandwidth, statistics on page views, and traffic to and from our website.

Data Usage Policy

- responding to your queries and requests and responding to complaints;
- managing the infrastructure and business operations of SPH and complying with internal policies and procedures;
- facilitating business asset transactions (which may extend to any merger, acquisition or asset sale;
- matching any Personal Data held which relates to you for any of the purposes listed herein;
- verifying your identity;
- preventing, detecting and investigating crime, including fraud and money-laundering, and analyzing and managing other commercial risks;
- protecting and enforcing our contractual and legal rights and obligations;
- conducting audits, reviews and analysis of our internal processes, action planning and managing commercial risks;
- preventing, detecting and investigating crime and managing the safety and security of our premises and services (including but not limited to carrying out CCTV surveillance and conducting security clearances); and/or
- compliance with any applicable rules, laws and regulations, codes of practice or guidelines or to assist in law enforcement and investigations by relevant authorities.

Data Removal Policy

- We will only retain your Personal Data for as long as necessary to fulfil the purposes we collected it for, including for the purposes of satisfying any legal, accounting, or reporting requirements.
- Should you wish to withdraw consent to use of your Personal Data (where this is the legal basis for us using your data) or obtain access to or make corrections to your Personal Data records, please log in to the relevant account through which

the Personal Data was provided, if any, failing which please contact our Data Protection Officer via the details listed in section 12 below.

Privacy Officer Contact

- **Name:** Mr Colin Low
- **Phone:** +65 63198555
- **Email**: dataprotection@sph.com.sg
- **Address**
  Singapore Press Holdings Limited
  1000, Toa Payoh North
  News Centre Singapore 319884

Link to Policy

https://sph.com.sg/legal/sph-privacy-gdpr/

Data Type Collected

Specific Data Description (awarded 1 point to compute barometer score)

- Name and contact details
- Date of Birth
- Financial Details
- Political Opinions
- IP (internet protocol) address
- Device ID
- App ID
- Vendor ID
- Advertising ID
- Work details
- Log in details
- Payment details
- Your location
- Your usage
- Your device
- Your engagement with advertisements

Generic Data Description (awarded 5 points to compute barometer score)

- Information on activities outside BBC when talk about us, articles read, programmes watched
- We receive Information that you provide to us directly, such as registration Information, and Information that is collected automatically, such as Information collected from your browser or device.
- From Third Parties. We also receive some Information about you from third parties.
- Basic personal details – such as name, title and date of birth;
- Personal contact details – such as home address, delivery address, home phone number, mobile phone number, and email address;
- Your opinion of our products and services;
- Your preferences, including newsletters or myFT topics;
- Your investment details if you use our portfolio tool to store your holdings;
- Your career details and CV if you use exec-appointments.com or our other recruitment pages;
- Information such as your dietary preferences and accessibility requirements when attending an FT Live event, which may be considered special categories of personal data.

- your name, NRIC, passport or other identification number, telephone number(s), mailing address, email address and any other information relating to you which you have provided us in any forms you may have submitted to us, or in other forms of interaction with you;
- information about your use of SPH's website and services
- your employment history, education background, and income levels;
- your payment related information, such as your bank account or credit card information, and your credit history; and
- information about your usage of and interaction with our website and/or services

M.Categories

List of the customer with the desired categories for the application:

I think the following categories have the most global and universal apps in order:

1. SOCIAL
2. COMMUNICATION
3. ENTERTAINMENT
4. TRAVEL_AND_LOCAL
5. NEWS_AND_MAGAZINES
6. FINANCE
7. TOOLS
8. FOOD_AND_DRINK
9. MAPS_AND_NAVIGATION
10. VIDEO_PLAYERS

Categories were taken from 42matters.com
https://42matters.com/docs/app-market-data/android/apps/google-play-categories
Find the apps in the table on Google Play
https://play.google.com/

ID	Description	Example
SOCIAL	Social	Facebook, Snapchat, Instagram, Tumblr, Meetup, Dribble, Behance
COMMUNICATION	Communication	WhatsApp, Messenger, Gmail, WhatsApp Business, Google Chrome, Firefox, Skype, Google Duo, Hangouts, Signal, Telegram, WeChat, Discord
ENTERTAINMENT	Entertainment	Netflix, IMDb, Vimeo, Eventbrite, Dailymotion
TRAVEL_AND_LOCAL	Travel & Local	Maps, Airbnb, Tripadvisor, Yelp, Expedia, Gojek, Hopper, Couchsurfing
NEWS_AND_MAGAZINES	News & Magazines	BBC News, CNA, CNN, Financial Times, The Straits Times, Google News
FINANCE	Finance	N26, Revolut, Monzo, Monese, Starling Bank, Cashplus, Tandem Bank, Atom Bank, Transferwise, Google Pay
TOOLS	Tools	Google Translate, Google Authenticator, Adobe XD, Adobe Photoshop, Samsung Voice Recorder, Gallery, Samsung Calculator, NordVPN
FOOD_AND_DRINK	Food & Drink	McDonald's, foodpanda, UberEats, Deliveroo, Pizza Hut, Domino's
MAPS_AND_NAVIGATION	Maps & Navigation	Uber, Lyft, Grab, Waze, Citymapper
VIDEO_PLAYERS	Video Players & Editors	YouTube, TikTok, VLC