BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

MANAGEMENT SYSTEM FOR FACEBOOK MARKETING CAMPAIGNS

SYSTÉM PRO SPRÁVU MARKETINGOVÝCH KAMPANÍ NA SOCIÁLNÍ SÍTI FACEBOOK

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR — **ANDREA STEJSKALOVÁ**
AUTOR PRÁCE

SUPERVISOR — **Ing. JIŘÍ HYNEK**
VEDOUCÍ PRÁCE

BRNO 2018

Zadání bakalářské práce/21252/2017/xstejs27

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Stejskalová Andrea**
Obor: Informační technologie
Téma: **Systém pro správu marketingových kampaní na sociální síti Facebook**
Management System for Facebook Marketing Campaigns
Kategorie: Informační systémy

Pokyny:

1. Prostudujte možnosti a principy marketingu na sociální síti Facebook.
2. Prostudujte Facebook API. Proveďte průzkum existujících nástrojů určených pro správu marketingových kampaní na sociální síti Facebook. Odhalte jejich výhody a nedostatky.
3. Navrhněte systém pro tvorbu, správu a vyhodnocování marketingových kampaní na sociální síti Facebook. Zaměřte se na analýzu cílových zákazníků a jejich případné reakce. Využijte tyto informace pro segmentaci trhu.
4. Navržený systém implementujte.
5. Implementovaný systém otestujte a vyhodnoťte jeho přínosy. Navrhněte jeho další možná rozšíření.

Literatura:

- White, C.: *Email Marketing Rules: A Step-by-Step Guide to the Best Practices that Power Email Marketing Success.* CreateSpace Independent Publishing Platform, 2nd edition, 2014, ISBN 978-1500981976.
- Facebook [online]: *Facebook Developer Documentation.* 2017 [cit. 2017-10-04]. Dostupné z: https://developers.facebook.com/docs/

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Hynek Jiří, Ing.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

L.S.

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstract

The main goal of this bachelor thesis is to design and implement a management system for Facebook marketing campaigns. This system is designed to easily manage marketing campaigns of Facebook pages via Messenger. It also allows processing results to determine the following marketing goals. The system is written in PHP framework Nette. It was designed according to the requirements from the Q2 Interactive company because there was not found any suitable existing solution on the internet. The system will be used to manage Facebook campaigns.

Abstrakt

Hlavním cílem této bakalářské práce je navrhnout a implementovat systém pro správu marketingových kampaní na sociální síti Facebook. Tento systém je navržen pro snadnou správu marketingových kampaní stránek služby Facebook přes platformu Messenger a nabízí možnost zpracování výsledků pro stanovení následujících marketingových cílů. Při vývoji tohoto systému byl použit PHP framework Nette. Systém byl navržen na základě požadavků firmy Q2 Interactive, protože na trhu nebylo nalezeno odpovídající existující řešení. Systém bude využíván pro správu kampaní na Facebooku.

Keywords

web application, management system, Facebook, marketing, data collection, PHP, Nette

Klíčová slova

webová aplikace, systém pro správu, Facebook, marketing, sběr dat, PHP, Nette

Reference

STEJSKALOVÁ, Andrea. *Management System for Facebook Marketing Campaigns.* Brno, 2018. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Jiří Hynek

Rozšířený abstrakt

Sociální sítě jsou v dnešní době všude kolem nás. Miliony lidí je každodenně používají, aby mohli komunikovat s vnějším světem. V současnosti je nejpoužívanější sociální síť Facebook. Její potenciál je především v nepřeberném množství způsobů sociální interakce. Facebook nejenže umožňuje komunikaci lidí s jinými lidmi, ale také nabízí možnost spojit se s lidmi a vystupovat přitom jako různé komerční subjekty. Konkrétní subjekt může být reprezentován například stránkou sítě Facebook. Schéma stránky funguje tak, že tvoří mezivrstvu mezi administrátory stránky a lidmi, které daná stránka zajímá. V mnoha případech je administrátorem stránky marketingový specialista, který se snaží co nejvíce zvýšit marketingový vliv stránky na potenciální zákazníky. Může tak činit různými způsoby, například psát příspěvky na zeď stránky nebo pomocí placené reklamy. Může také využít možnost spojit se zákazníky přes platformu Messenger, která se vyvinula z původního chatu na síti Facebook.

Hlavním cílem této bakalářské práce je navrhnout a vytvořit novou aplikaci, která bude umět spravovat online marketingové kampaně, které běží přes platformu Messenger. Tato aplikace umožní marketingovým specialistům vytvářet nové online kampaně a spravovat je. Dále také nabídne možnost prohlížet si a exportovat data o uživatelích nasbíraná přes tyto kampaně. Tato data budou moci být později použita pro zlepšení cílení dalších kampaní.

Obsah této práce je rozdělen do několika kapitol, kde každá kapitola popisuje konkrétní etapu vývoje zadané aplikace. Kapitola 2 je věnovaná principům dnešního marketingu. Více je zde rozveden online marketing, jeho druhy a způsoby využití. S ohledem na téma práce jsou zde rozebrány možnosti marketingu na sociální síti Facebook a jeho platformě Messenger. Kapitola 3 pojednává o požadavcích na aplikaci. Je zde také představeno několik existujících řešení, která jsou poté porovnaná se stanovenými požadavky. V kapitole 4 je podrobněji rozebrána integrace aplikace s aplikačním rozhraním sítě Facebook. Jsou zde popsány mechanismy, podmínky a potřebná nastavení na straně Facebooku. Jejich dodržení je potřeba pro splnění požadavků na aplikaci a zároveň také pro správnou funkci aplikace. Kapitola 5 je zaměřena na návrh aplikace. Obsahuje detailní popis návrhu databáze, ke kterému jsou použity diagram užití a entity-relationship diagram. Na use case diagramu jsou znázorněny všechny role a možné akce, které může uživatel s tou konkrétní rolí v aplikaci provést. Entity relationship diagram pak srozumitelně popisuje navržené databázové entity a druhy vztahů mezi nimi. Druhou částí této kapitoly je popis webových technologií, které byly použity k vývoji aplikace. Kapitola 6 se zabývá implementací navržené aplikace. Zahrnuje oblasti jako jsou: vývojové prostředí, práce s databází, architektura aplikace a její moduly. V sekci moduly jsou pak detailně popsány důležité komponenty, které implementují funkce vyplývající z kapitoly 3 o požadavcích na aplikaci. Pro větší názornost a pochopení textu jsou k těmto popisům přidány snímky z prostředí aplikace. Ty jsou k dispozici pro náhled v příloze A. Kapitola 7 je o testování aplikace a je zde také zahrnuta sekce o možných rozšířeních, která by mohla být implementována v budoucím vývoji aplikace.

Sled postupu řešení všech záležitostí uvedených v zadání kopíruje sled popsaných kapitol výše. Nejdříve jsem se seznámila s principy marketingu, které se v dnešení době používají. Dále jsem nastudovala existující řešení a vyhodnotila jejich výhody a nevýhody vzhledem ke stanoveným požadavkům na aplikaci od firmy Q2 Interactive. To vedlo k návrhu a pozdější implementaci nové aplikace. Ta je postavena na PHP frameworku Nette, který podporuje MVC architekturu, takže jsou oddělena data od logiky a zobrazení. Implementovaná aplikace splňuje požadavky popsané v kapitole 3. Lze díky ní spravovat kampaně běžící na platformě Messenger, konfigurovat jejich zprávy a přiřazovat je ke konkrétní stránce. Stránky se dají jednoduše propojit s těmi Facebookovými a dále se dají seskupovat do

projektů kvůli větší přehlednosti. Z marketingového hlediska je implementovaná aplikace prospěšná díky ukládání odpovědí od lidí na rozeslané zprávy. Tyto odpovědi mohou být pak v rámci aplikace použity pro stanovení dalšího (konkrétnějšího) marketingového cíle, který se díky získaným vstupním informacím může stát úspěšnější. Nasbírané informace mohou být také exportovány z aplikace ve formátu CSV. Co se týče dalšího vývoje, tak aplikace bude dále vyvíjena jako projekt firmy Q2 Interactive. V současnosti běží v testovacím módu na webu ajdinas.cz/bt[1].

[1]Dostupné na webu: https://ajdinas.cz/bt

Management System for Facebook Marketing Campaigns

Declaration

Hereby I declare that this bachelor's thesis was prepared as an original author's work under the supervision of Mr. Ing. Jiří Hynek. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

.....
Andrea Stejskalová
May 16, 2018

Acknowledgements

I would like to thank Mr. Tomáš Antoš from Q2 Interactive for valuable advice in the area of online marketing and also for an idea about information system which is a purpose of this thesis. Also, I would like to thank Ing. Jiří Hynek for the thesis supervision. My thanks also go to my family and especially to my parents for endless support and motivation.

Contents

Chapter 1

Introduction

Social networks are everywhere around us today. Millions of people use them on a daily basis in order to communicate with the outer world. The most used social network now is Facebook. Its potential is in a wide range of ways of social interacting. It does not only allow to connect people with each other but also with various commercial subjects. The subject can be represented by the Facebook page. The Facebook page scheme is designed to be a layer between the page administrator and the people interested in the page. In many cases, the page administrator is a marketer who tries to make the best to increase marketing influence of the page onto potential customers. There are several ways to do it. For example, by publishing posts on the wall of the page or by the sponsored advertisement. There is also an option to get in touch with customers via Messenger which has developed from the former Facebook chat.

The main purpose of this bachelor thesis is to create a new application which will be able to manage online marketing campaigns running via Messenger. The application will allow marketers[1] to create new online campaigns that will include a message, manage these campaigns, list the collected information from the Facebook users and help to improve the next campaigns according to the retrieved data.

The content of the thesis is separated into several chapters, each of them describes a particular topic: Chapter 2 is about marketing and its areas such as online, email or social media marketing. There is also a description of marketing possibilities on Facebook and Messenger. Chapter 3 discusses the application requirements and existing solutions. In the end, there is their confrontation with the requirements. Chapter 4 is dedicated to the Facebook API integration with the designed application. It consists of Facebook Login description and also other necessary Facebook tools that have to be used for the proper function of the application. Chapter 5 is focused on the application design, which means that there can be found database design along with entities. The next part of this chapter is a description of the technologies which were used for application development. In chapter 6 there is included the implementation of the designed application. It describes these areas: development environment, working with the database, application architecture, and modules. Chapter 7 is about the testing of the application, and there is also included a section about possible extensions that can be implemented in the future development.

[1] people who work in the marketing field

Chapter 2

Marketing today

There will be described terms such as marketing, online marketing, email marketing and relationships with each other in this chapter. Along with these terms, there will be introduced features of the Facebook social network that are usable in online marketing, mainly focusing on the Messenger Platform as a new marketing tool.

2.1 Marketing

According to Kotler and Armstrong [23], marketing is defined as the process by which companies create value for customers and build strong customer relationships in order to capture value from customers in return. This process consists of steps mentioned in Figure 2.1:

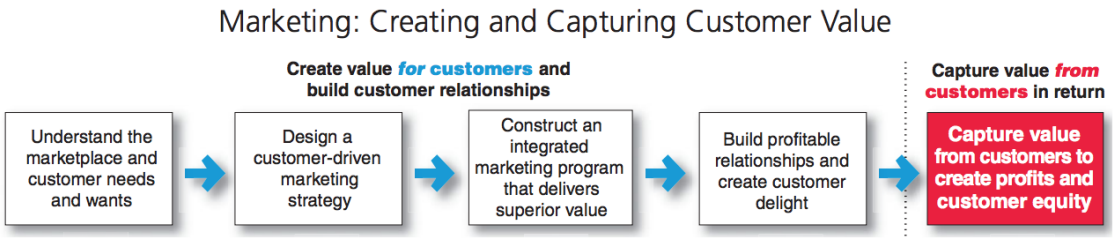

Figure 2.1: Marketing as a process [23]

2.2 Online marketing

Online marketing is a modern branch of marketing itself. As the Internet is still becoming more popular and reachable for a lot of people, it has overtaken some principles and behavior from the real world. A simple example can be a principle of shopping: In the real world there are stores and people buy goods by putting them into the shopping cart and then pay at the checkout. The online shopping took over the principles of the store, shopping cart and checkout entities, they were just modified to the Internet environment. The same thing happened with marketing. Instead of the real billboards, there are places for rent on the internet websites. And instead of the leaflets in the real mailboxes, there are advertisements in the email boxes. As the previous sentences indicate, there are more types

of online marketing such as banners, pop-ups, SEO advertising or emailing. I would like to describe email marketing more and highlight a similarity between it and the Facebook Messenger marketing features in the next sections.

2.3 Email marketing

Despite the fact that email marketing represents the group of older tools for online marketing, it is still being used nowadays. The main advantage of it is that the advertisement in email can reach people all over the world and the delivery itself costs nothing. On the other hand, many people today consider newsletters and advertising emails as spam. This decreases the efficiency of the email marketing. Furthermore, it is important to mention that in some countries it is even forbidden to send the unsolicited newsletters. Because of the need for the agreement with receiving them, many websites use for example a required agreement checkbox as a condition to proceed through the registration or another form. Fortunately, this practice is becoming deprecated. Instead of the required agreement, there is usually an option to choose whether a person wants to receive newsletters or not [14]. The newsletters that are agreed by person's will are actually better-targeted because of the obvious person's interests.

2.4 Social media marketing

In contrast to email marketing, social media marketing is the newer part of online marketing. It is closely associated with the arrival of *Web 2.0* because people became more active participants than ever before. *Web 2.0* has allowed users to interact with other users and businesses. The rise of the social networks caused the expansion of developing new online marketing strategies and tools which are being improved until these days.

2.4.1 Facebook and marketing

Facebook is a good tool for marketing today because millions of people use it on a daily basis. According to web Statista[1], Facebook is the most used social network in these days, followed by the Messenger on the 4th place. The detailed statistics can be seen in Figure 2.2.

Facebook offers its own interface for marketers for making campaigns. A preview of campaign settings can be seen in Figure 2.3. The outputs of these campaigns are sponsored ads which are being displayed to users everywhere on the Facebook website (according to the settings of each campaign).

[1] http://statista.com/

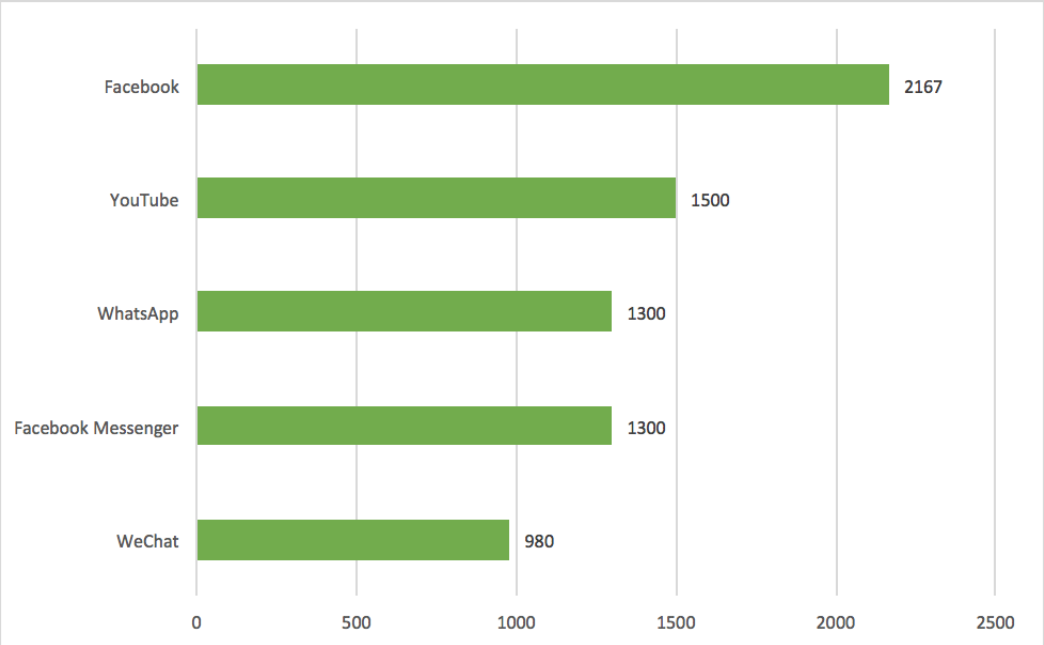

Figure 2.2: Number of active users on social networks (in millions) [13]

2.4.2 Messenger and its marketing potential

As it was mentioned in the previous paragraph, the marketers can manage their sponsored posts by the Facebook interface. There is one specific feature in advertisement settings that allows the marketer to display a button which directly triggers a conversation between the page and the user. These settings are shown in Figure 2.3, and the button setting is in the red-bordered area. This feature is an important part of the marketing on the Messenger because the page can send messages only to people, who have interacted with the page via the Messenger ever before. The Messenger documentation [8] declares it clearly: To send subscription messages, you must apply for the *pages_messaging_subscriptions* permission for your bot. If your bot has *subscription_messaging permission*, users automatically opt-in to receive subscription messages when they start a conversation; however, it is highly recommended that your bot ask permission to send subscription messages to ensure a positive user experience.

With this in mind, we can see that there is an analogy between the page messaging subscription permission and the newsletter subscription permission in email marketing (mentioned in section 2.3). Thanks to the regulations described in the previous paragraph, the Messenger can efficiently protect its users from unsolicited messages.

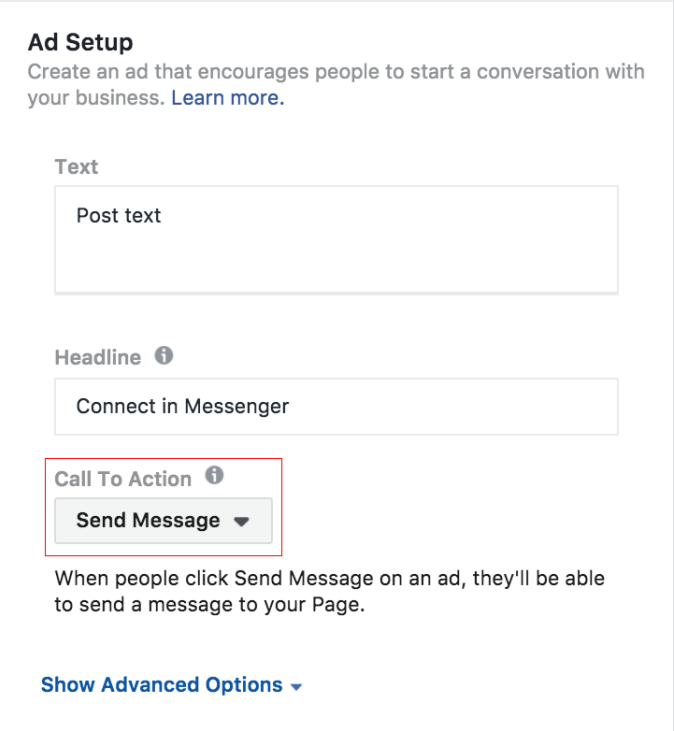

Figure 2.3: Ad Setup from Facebook Ads Manager [2]

Chatbot as a new marketing tool

A *chatbot* is an automated assistant that has permission to communicate with people via messages. It always runs on some platform, for example, the Messenger. It also can just inform people about something by an automated answer to specific keywords as it can be seen in Figure 2.4.

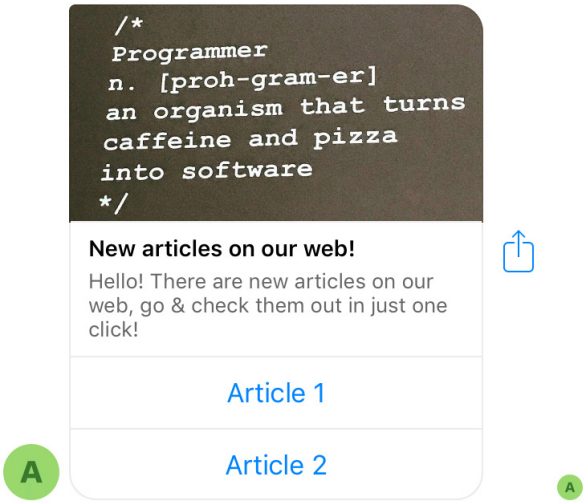

Figure 2.4: Simple newsletter chatbot example

Chapter 3

Requirements analysis

This chapter focuses on the requirements analysis in section 3.1 and contains a brief review of existing solutions in section 3.2. In the end, there is a conclusion with the comparison of the application requirements and features offered by the described solutions.

3.1 Application requirements

The application requirements were created in accordance with company's assignment. Their detailed description follows in the next paragraphs.

The first requirement was *Facebook Login* service integration. It is needed for later interaction with user's Facebook account, in this case with Facebook pages and their conversations with people.

Another requirement was to allow Facebook pages to be sorted into projects. In this case, the project may indicate the client (e.g. company name). One client can have multiple Facebook pages. For example, the client can own a store which is situated in different countries. The store can have a Facebook page for each country in order to the different offered products, sales, etc.

There should be also an option to create campaigns dedicated to particular Facebook page. In this case, the campaign indicates a message that can be sent to people. The message settings have to be intuitive. The Messenger supports not only simple text messages but also several templates. These templates are composed of texts, buttons, images, etc. The message settings have to allow to choose a particular template and show proper settings according to it. There should be also a message preview to check that everything is set properly. The next feature that has to be implemented is recipients settings. It should be possible to search and filter through page's people (along with information about them) and select campaign recipients from them in bulk. The number of recipients can be in thousands, so it is important to implement a solution that can handle and send this amount of messages.

Further, the feedback from recipients has to be caught because of the development and improvement of other campaigns. The Messenger offers a quick reply option to a text message. The user's reply must be kept in the application as user's specific information. Then it can be used for better targeting of another campaign. The collected data about people has to be filterable and exportable from the application.

3.2 Existing solutions

Despite the fact that the usage of Messenger API is not so expanded yet (and mainly here in the Czech Republic), there already exist solutions for configuring and managing automated chatbots and send broadcast messages. In this section, there are described several solutions that are available nowadays on the Internet.

3.2.1 Motion AI

Motion AI[1] describes itself as a platform, that gives anyone the ability to easily build and deploy chatbots on a variety of different mediums [15]. It is running as a web application and supports not only the Messenger but also another APIs such as Slack[2], SMS, e-mail, web chat or even custom API. It also offers several plans which differentiate in price, branding, maximum message count and maximum bot count. There is an option of a free account, but this account is very limited and probably dedicated to being just a tryout version of account and it is not suitable for a longterm usage.

Broadcast messages

Considering the context of this thesis, the most important feature of *Motion AI* is broadcast messaging. It allows the marketer to send single messages to the people that have already started a conversation with the particular page. This is a suitable tool for spreading news, updates etc. to the certain circle of people. But there are also limitations that make using broadcast messages harder.

Firstly, there are no kept reports of the broadcast messages, only for chatbots.

Secondly, there is only option to filter recipients by first and last name and first and last sent message to the page. There is no option to filter them by other attributes, e.g. gender or language.

Thirdly, a chatbot is needed for broadcast messaging, because there is no other way how to get recipients through *Motion AI* than from previously set and used chatbot. With this, there is a problem with user account plans limitations. When you want to manage broadcast messages for more pages at the same time, you have to have more chatbots and this leads to purchasing one of the paid user account plans.

User interface

Motion AI's user interface is clear and simple. By default, it consists of 2 navigation menus and a big content block. The horizontal menu consists of the page title and user info and settings, the vertical one offers quick links to the main features of the application such as bots settings, conversation overview or reports.

[1] http://motion.ai
[2] http://slack.com

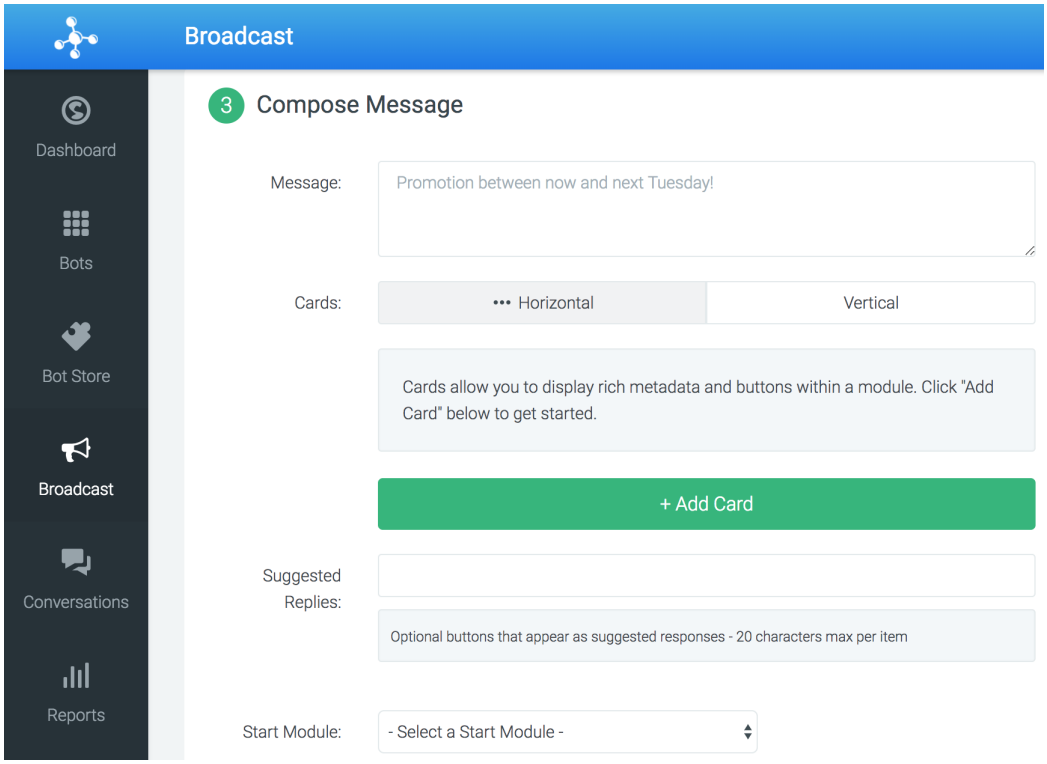

Figure 3.1: *Motion AI* interface preview

3.2.2 It's Alive!

It's Alive![3] is another tool for managing chatbots and broadcast messages. In comparison to *Motion AI*, it offers fewer features and seems more simple. It is a web application built for Messenger chatbots only and does not support any other API.

Broadcast messages

It's Alive! supports sending broadcast messages through the chatbot recipes. The recipe is an analogy to the message settings and its interface preview can be seen in Figure 3.2. When somebody wants to broadcast a message, it has to have *user trigger setting* set as *instant notification*. Unfortunately, there are also limitations and a couple of them are the same or similar to *Motion AI*.

Firstly, the analytics of the recipes are available only for premium users, so when the marketer wants to have some feedback, he is forced to purchase one of the paid user account plans.

Secondly, as in *Motion AI*, a chatbot is needed for every recipe. And if the marketer wants to manage more pages, he is also forced to purchase one of the paid user account plans.

Thirdly, it is inconvenient that there is no option to choose recipients of the particular recipe. The message can be sent only to all people that belong to the particular page.

[3] http://itsalive.io/

User interface

The user interface of *It's Alive!* looks simple, but it did not seem to me as clear as the *Motion AI*'s one. It took me a while to understand how it works and where all of the settings are located. But on the other hand, the message configuration itself is natural and human-oriented.

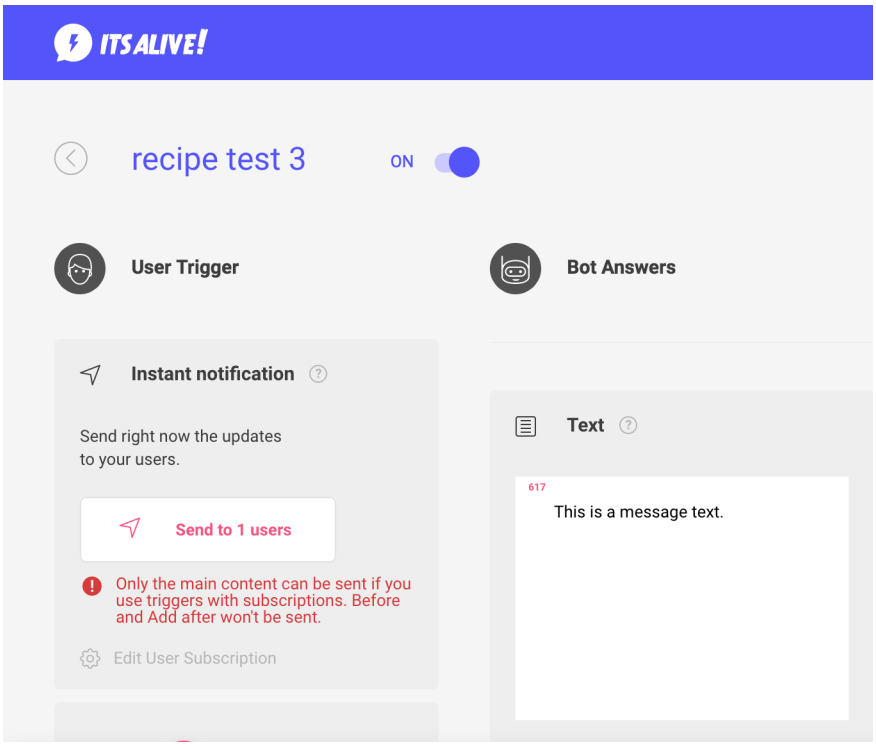

Figure 3.2: *It's Alive!* interface preview

3.2.3 ChatFuel

ChatFuel[4] describes itself as the tool whose goal is to make bot-building easy for anyone [1]. It is a very complex tool for managing Messenger chatbots and broadcast messages and in my opinion the most complex tool from the existing solutions presented in this thesis. It is also running as a web application as the previous existing solutions presented before.

Broadcast messages

ChatFuel has a wide offer of broadcast messaging settings. There is an option to choose recipients by various attributes. These attributes are not only the ones retrieved from Facebook but also the custom ones that can be caught by the chatbot.

For example, the chatbot is set to ask whether the person wants to get red or white wine. The response is set to be caught to the attribute called *wine*. Once the person chooses one of the options, the response will be saved to the *wine* attribute. The marketer can now use this attribute to search and filter all people with it, send them broadcast messages or just export them.

[4] http://chatfuel.com/

The attributes are a smart way how to sort people to the groups. They can be used for better targeting of the next campaigns because of the information that people tell to the page during the conversation. This feature is very helpful in creating new campaigns because the marketer can deduct what is the people's point of interest from the attributes.

However, when it comes to the option of a free user account, it is the same as in *Motion AI* and *It's Alive!*. The free account is limited. For example, the people statistics based on the attributes are only available with *ChatFuel PRO plan* which is paid and the price is increasing according to the count of chatbots/pages. So if the marketer wants to manage for example 10 or more pages, it is really expensive.

User interface

ChatFuel's user interface is very friendly and clear and it consists of 2 main menus and a number of logically divided blocks of content. The horizontal one is for breadcrumb navigation to get person know where he is and the vertical one offers the quick links to the page settings such as people, broadcast messages or general configuration. The message configuration is well designed and easy to use, a preview of it can be seen in Figure 3.3.

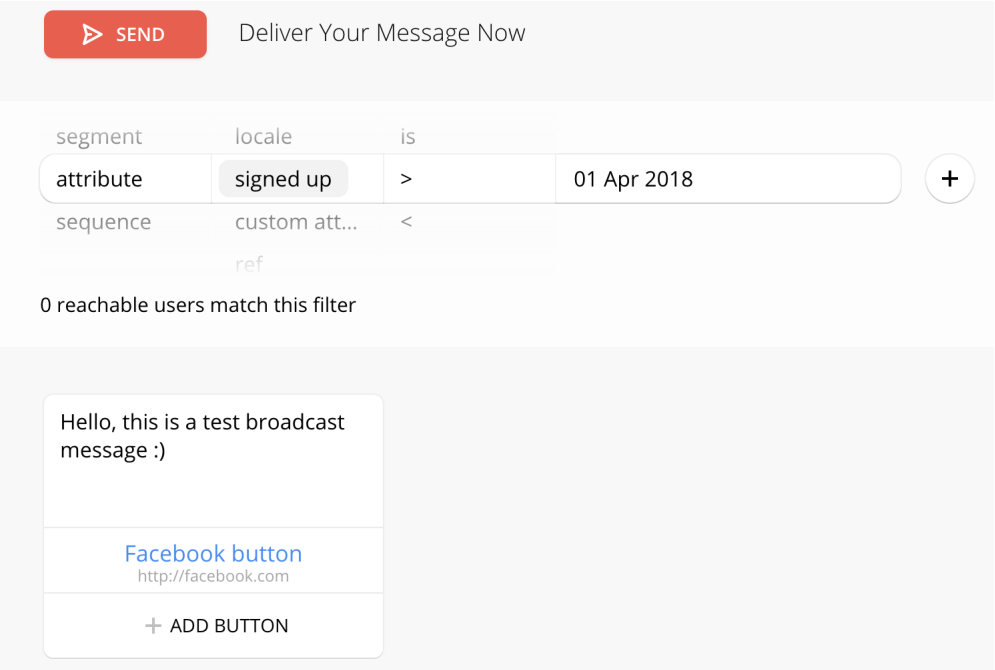

Figure 3.3: *ChatFuel* broadcast message configuration preview

3.3 Conclusion

There were presented three existing solutions with the highlight on their most important features according to application requirements stated in section 3.1. In conclusion, it seems like the *ChatFuel* is the best solution from them when considering the application features. It supports the widest range of people sorting options and allows later work with retrieved information from the conversation with the chatbot/page.

However, all of the presented solutions need to have a chatbot assigned to the page before the broadcast messages can be sent. There is no other way how to put users into

the audience database of these tools than set up a chatbot which saves each person who starts communication with it.

The biggest disadvantage is that the count of the managed pages is directly proportional to the price. The exception is that one page is usually for free, which means that these tools are more suitable for single managing than for managing by a marketer in some agency.

In conclusion, according to the existing solutions analysis, it has been revealed that there is no suitable application for the online marketer to manage more Facebook pages with campaigns, people, responses, and statistics. All of the existing solutions either do not offer features important for people targeting based on the information they give through the conversation or they offer these features only with the paid user account or only for one page for free.

Chapter 4

Facebook API

This chapter discusses the integration with Facebook. In the first section, there is a brief description of *Facebook Login*. On the other hand, the second section contains information about *Graph API* including *Send API*, *Page-scoped ID*, and *Webhook*.

4.1 Facebook Login

One of the requirements was to implement *Facebook Login* because it is required to agree with giving permissions when logging in to the third party application. In Figure 4.1 can be seen a dialog which opens up after clicking on the *Login via Facebook* button.

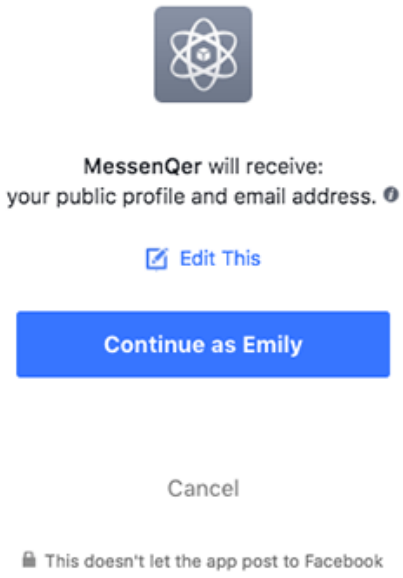

Figure 4.1: Facebook Login permissions dialog

Facebook Login is a secured service which is powered by *OAuth* (Open Authorization). *OAuth* is an open standard for authorization. It provides client applications secure delegated access to server resources on behalf of a resource owner. It also specifies a process for resource owners to **authorize third-party access to their server resources without sharing their credentials** [9].

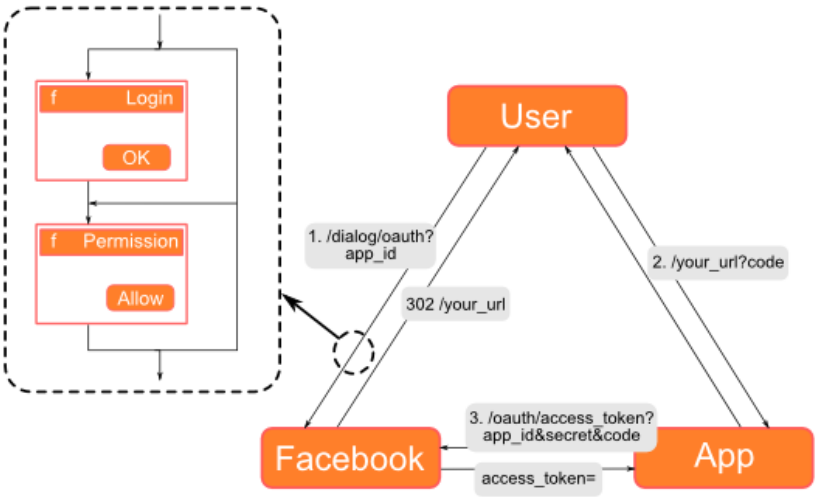

Figure 4.2: Facebook OAuth authentication [3]

As Figure 4.2 shows, the Facebook API returns *user access token* after successful login. This access token is necessary for the further running of the application because information about user's Facebook pages can be retrieved only with it.

4.2 Graph API

Facebook offers ways to send messages to Facebook users as a page. One of these ways is to use the Facebook web itself. But when it comes to chatbots and automated messaging, there is the *Graph API*.

The Facebook documentation [8] says that the Graph API is the primary way to get data into and out of the Facebook platform. It is a low-level HTTP-based API. The applications can use it to programmatically query data, post new stories, manage ads, upload photos, and perform a wide variety of other tasks.

The Graph API is composed of *nodes* (e.g. page), *edges* (e.g. message) and *fields* (e.g. recipient id). These three items serve to get or post information about some Facebook object (e.g. user, page) via HTTP methods GET and POST. When using these methods, the *access token* is always required. Each Facebook page has its own *page access token* which differs from the *user access token* mentioned in section 4.1. The *page access token* serves for communication that is related to the Facebook page. It can be for example getting the list of conversations (GET method) or sending messages (POST method).

4.2.1 Send API

The *Send API* is the main API[1] used to send messages to users, including text, attachments, structured message templates, quick replies and so on. This API uses Graph API. The Send API request example can be seen in Listing 4.1. For sending messages to the user, it is required to know the *page access token* which can be retrieved by sending a proper request via the Graph API and the *page-scoped ID* of the user.

[1] Application Programming Interface

```
curl -X POST -H "Content-Type: application/json" -d '{
  "messaging_type": "<MESSAGING_TYPE>",
  "recipient": {
    "id": "<PSID>"
  },
  "message": {
    "text": "hello, world!"
  }
}' "https://graph.facebook.com/v2.6/me/messages?access_token=
<PAGE_ACCESS_TOKEN>"
```

Listing 4.1: Send API request example

4.2.2 Page-scoped ID

When the user contacts the page by sending a message, Facebook will assign the *page-scoped ID*, shortly PSID, to him. It is a unique ID for this user when communicating with this particular page. By using PSID, it is secured that the page can send messages only to the people who have started a conversation with it. The PSID cannot be retrieved via Graph API and it is not the same ID as user's Facebook ID. The PSID can be retrieved only from the webhook.

4.2.3 Webhook

Webhook is an application endpoint which receives events coming from the Graph API. It has to run via HTTPS protocol. There are several types of events that are sent to the webhook. For example, when the user sends a message or just displays a message from the page. The events are sent in the JSON[2] format which is easy to parse. Webhook event example can be seen in Listing 4.2:

```
{
  "object":"page",
  "entry":[{
      "id":"<PAGE_ID>",
      "time":1458692752478,
      "messaging":[{
          "sender":{
            "id":"<PSID>"
          },
          "recipient":{
            "id":"<PAGE_ID>"
          },
          ...
        }]
    }]
}
```

Listing 4.2: Webhook event example

[2] https://www.json.org/

Chapter 5

Application design

This chapter is divided into two main sections which are database design and used web technologies. In the first section is introduced chosen database management system. There is also the use case diagram and entity relationship diagram along with the description of the database entities. In the second section are introduced web technologies that were used for application development.

5.1 Database design

Databases are the indispensable tool for storing data and working with them. Today there are many types of databases such as relational, object-oriented or noSQL databases. I chose to use a relational database for my application because of two reasons: First, the data that has to be stored in the database can be easily structured to relations. Second, according to the statistics on web DB-Engines[1], the relational database management systems (DBMS) are the most used ones these days. They occupy the first four places in the ranking, see Table 5.1.

5.1.1 MySQL database

MySQL is a DBMS for relational databases. Its main advantages are portability across different platforms and the wide range of implemented application program interfaces [22]. MySQL is currently maintained by Oracle Corporation[2]. According to web DB-Engines [7], it is the second most used DBMS today after Oracle itself:

Position	**DBMS**	**Database model**
1.	Oracle	Relational DBMS
2.	MySQL	Relational DBMS
3.	Microsoft SQL Server	Relational DBMS
4.	PostgreSQL	Relational DBMS
5.	MongoDB	Document store

Table 5.1: Database engines ranking (April 2018) [7]

[1] https://db-engines.com/en/ranking
[2] Official website: https://www.oracle.com/

5.1.2 Use case diagram

Use case diagram, also known as UCD, consists of the application users and their possible interactions. It basically contains scenarios in which the application interacts with people. The use case diagram of my application is based on the information that resulted from the requirements analysis in Chapter 3. It can be seen in Figure 5.1.

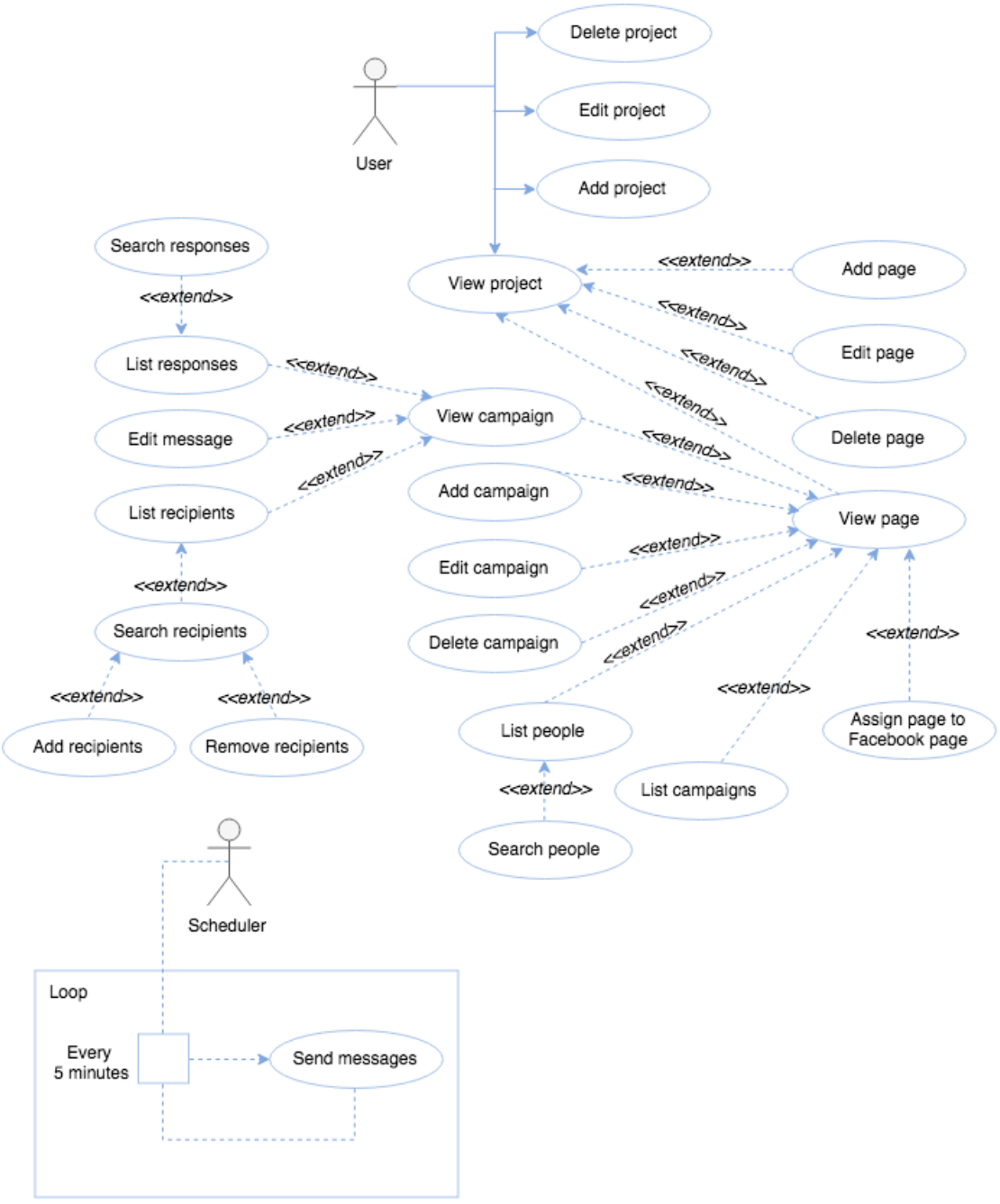

Figure 5.1: Use case diagram with 2 actors: user and scheduled CRON job

5.1.3 Entity Relationship Diagram

Entity Relationship Diagram, also known as ERD or ER diagram, is a type of structural diagram for use in database design. An ERD contains different symbols and connectors that visualize two important information: The major *entities* within the system scope, and the *inter-relationships* among these entities [19]. The reason why I have used ERD for the database schema visualization is that it can be easily transformed into the relational model. The ERD of my application can be seen in Figure 5.2. The detailed description of all entities with their relationships to each other follows in the next paragraphs.

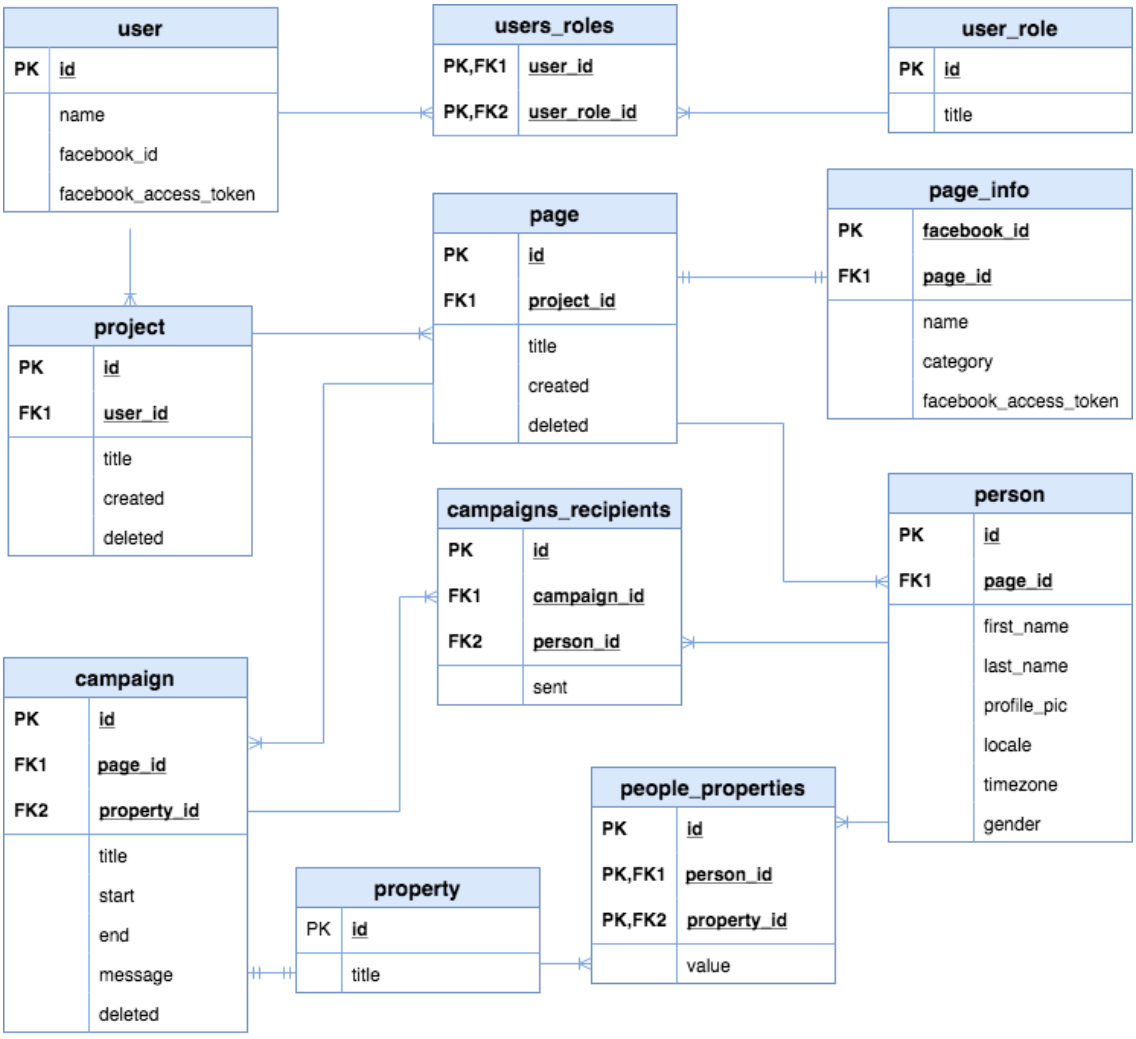

Figure 5.2: Entity Relationship diagram

5.1.4 Database entities

User

The user entity represents a profile of application user, i.e. the person who can log in and further operate in the application. Since the login process is connected with *Facebook login* (see Chapter 4), the `facebook_id` and the `facebook_access_token` attributes have to be saved. Also, the user's `name` is saved to the database.

User role

This entity consists of only one attribute `title`. As the name suggests, it serves to store different user roles in the database. Many users can have many user roles. Due to the many-to-many relationship between these two entities, there is also an associative table *users_roles* in the ER diagram which connects them. The application currently uses only one role called *user*. The user role entity is prepared for the possible addition of other roles, e.g. administrator role.

Project

This entity stores information about projects. The purpose of them is described in Chapter 3. Each project has to be assigned to the user.

Page

The page entity is the key entity of the whole application because almost everything is connected with it. It has to be assigned to the project.

Page info

The page info entity is a representation of the Facebook page in this application. It stores information given by Facebook when the page is being linked to the Facebook page. It is also assigned to the page entity having the one-to-one relationship with it. As in user entity, there have to be saved page's `facebook_id` and `facebook_access_token` attributes for further operation with the Messenger via the application. The next saved attributes are Facebook page's `name` and `category`.

Property

The property entity has only one attribute `title` because it is used for an assignment to campaign and person entities which are described below.

Campaign

This entity is used for storing information about campaign's `title`, `start`, `end`, `message` and if it has been `deleted`. The campaign can be associated with only one page and also with only one property. On the other hand, the campaign can have many recipients. This relationship is represented by the associative table *campaigns_recipients* that connects the campaign with person entity. This associative table contains one more attribute called `sent`, which is used for saving a date when the campaign message was sent to the recipient.

Person

This entity contains information given by Facebook about users that belong to the page. The page is referred by the `page_id` foreign key. Due to the privacy policy, Facebook gives only some of the users attributes including `first_name`, `last_name`, `profile_pic`, `gender`, `locale` and `timezone`. Person entity is also in a many-to-many relationship with property entity. These *people_properties* relations has to have a `value` because they represent responses to the particular campaign message.

5.2 Used technologies

5.2.1 HTML

Hyper Text Markup Language is the standard markup language for creating web pages. This language uses elements in angled brackets with various attributes to describe the structure of web pages [21].

5.2.2 CSS and preprocessors

Cascading style sheets is a language that describes the style of an HTML document and how HTML elements should be displayed.

In last years, the evolution of CSS has started by developing the preprocessors. Today they are *must-have* tools for web application development. Their purpose is to extend CSS with variables, functions, mixins, etc. They also make the code more organized and clear by nesting blocks. They usually have the same or similar syntax as CSS.

The thing is that a compiler is needed for using a preprocessor. This means that the source code has to be recompiled after every change in it. But this is not a problem since there exist automated script tools, for example *Gulp*. It can check source files while running and compile them after every change automatically (more in section 6.1.4).

According to web HTML Mag, the most known and used preprocessors today are *LESS*[3], *SASS*[4], and *Stylus*[5] [12]. For my application, I chose to use SASS preprocessor. The abbreviation SASS stands for *Syntactically Awesome Style Sheets*. The reason for this choice is that SASS is natively supported by the Bootstrap library (see section 5.2.5). And as it was said, the preprocessors extend CSS with handy tools. The SASS preprocessor supports:

- **variables** - starting with `$` sign, expecting one parameter containing value of variable
- **operators** - such as `+`, `-`, `*`, `/`
- **inheritance** - between blocks by extending one by another
- **mixins** - custom functions starting with `=` sign that can be called by the `+` sign with the name of the mixin

SASS also supports two types of the syntax:

1. SCSS - CSS syntax extended by variables, nesting, etc.
2. SASS - CSS syntax with omitted brackets and semicolons, nesting is provided by the indentation

I chose the SASS syntax because of the indentation instead of the classic CSS syntax with brackets and semicolons. It seems to me more organized, clear and better to read.

[3] Official website: http://lesscss.org/
[4] Official website: http://sass-lang.com/
[5] Official website: http://stylus-lang.com/

5.2.3 JavaScript

JavaScript is a programming language which specifies the behavior of web pages. Today it has many libraries and even the whole frameworks are written on its basis. I chose to use the *jQuery* library because in my opinion it has easier syntax than JavaScript itself and the code looks more organized and shorter.

5.2.4 AJAX

AJAX stands for Asynchronous JavaScript and XML and it is based on the following open standards [18]:

- browser-based presentation using HTML and CSS
- data is stored in XML format and fetched from the server
- behind-the-scenes data fetches using *XMLHttpRequest objects* in the browser
- JavaScript to make everything happen

When using AJAX e.g. to submit a form, JavaScript will make a request to the server, interpret the results and update the current screen without need to reloading page. It is much more user-friendly than always redirect page after some request to the server.

5.2.5 Bootstrap library

Bootstrap is an open source toolkit for developing with HTML, CSS, and JavaScript [5]. Building the visual part of the application is much easier with it because it can be used with preprocessors (see section 5.2.2). Bootstrap also offers a responsive grid system, extensive prebuilt components, color schema, useful variables and powerful plugins built on jQuery.

Chapter 6

Implementation

This chapter is about the application implementation. It includes a brief description of the development environment, working with the database, and application architecture along with its modules. There are described terms such as object-relational mapping, Doctrine, Nette, MVC architecture, and their usage.

6.1 Development environment

When implementing a complex application, it is almost necessary to use some integrated development environment, shortly IDE. I have chosen **PhpStorm**[1]. According to the page SitePoint[2], PhpStorm was chosen as the best PHP IDE in 2014 [4]. It is distributed under the student license. It also supports PHP language with frameworks and the MVC architecture. It has a plenty of useful functions, e.g. integrated terminal console, which comes to hand when compiling source code, and integrated version control system. The last reason of this choice is that I use PhpStorm on a daily basis and I am used to working with it. Along with IDE, other useful tools were used during the implementation, such as Git, Composer, NPM, and Gulp. Their brief description follows in the next paragraphs.

6.1.1 Git

Git is a free and open source distributed version control system. It serves for managing changes to source code over time. My application has been stored in a repository on *GitHub*[3] and maintained by the Git client application *SourceTree*[4].

6.1.2 Composer

It is a tool for dependency management in PHP. It allows to declare the libraries that the particular project depends on and it will manage (install/update) them [11]. It is very useful when using libraries that depend on specific versions of each other. All libraries managed by Composer are stored in the `composer.json` file and dependencies information can be found in the `composer.lock` file.

[1] Official website: https://www.jetbrains.com/phpstorm/
[2] SitePoint is a hub for web developers, https://www.sitepoint.com
[3] GitHub website: https://github.com/
[4] SourceTree official website: https://www.atlassian.com/software/sourcetree

6.1.3 NPM

It is the package manager for JavaScript and the world's largest software registry [16]. The packages can be easily downloaded to project by using `npm install` *`package-name`* command in the command line. All of the saved packages are stored in the `package.json` file.

6.1.4 Gulp

Gulp is a JavaScript toolkit for automating tasks in the development workflow [10]. It is a JavaScript file named `gulpfile.js` which helps to automate previously defined tasks. For example, I used it for automated CSS and JavaScript files compilation and minification.

6.2 Database

6.2.1 Object-relational mapping

Object-relational mapping, shortly ORM, is a mechanism that makes it possible to address, access and manipulate objects without having to consider how those objects relate to their data sources. ORM lets programmers maintain a consistent view of objects over time, even as the sources that deliver them, the sinks that receive them and the applications that access them change [20]. In my application is ORM used to connect database entities (tables) with proper model entities (classes). This means that each database entity is mapped to its own model entity, and each database entity attribute (column) is mapped to model entity attribute (variable).

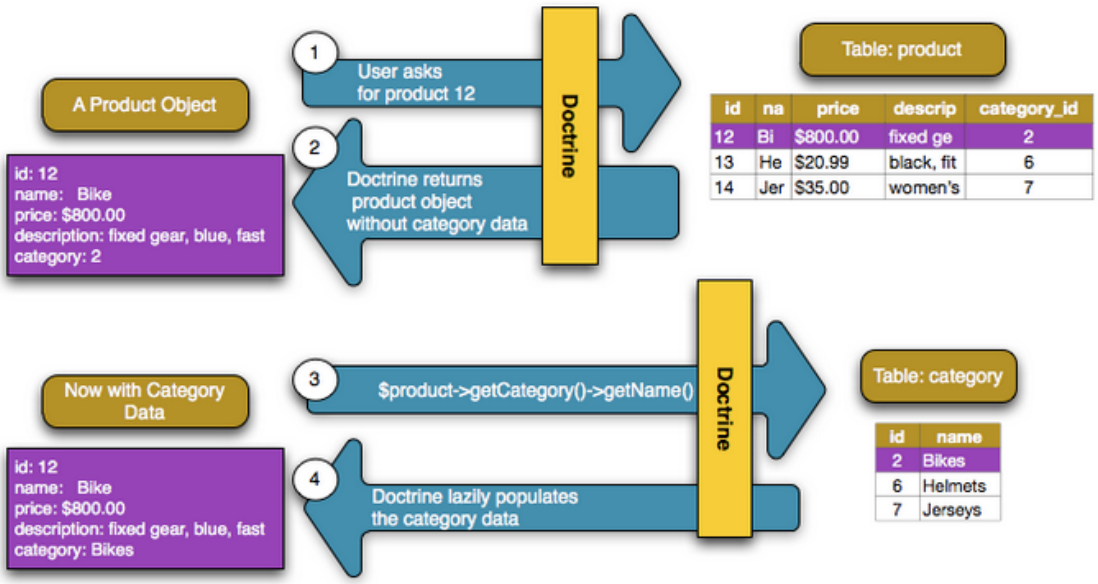

Figure 6.1: Object-relational mapping diagram [6]

Doctrine

I chose *Doctrine* framework[5] to provide the object-relational mapping. It is a framework for PHP that sits on top of a *database abstraction layer*. One of its key features is the option to write database queries in a proprietary object oriented SQL dialect called *Doctrine Query Language*. This provides developers with a powerful alternative to SQL that maintains flexibility without requiring unnecessary code duplication [17]. There is a special *dockblock* annotation[6] for configuration of entities and attributes in PHP using Doctrine in order to map them with the database. Simple entity configuration example can be seen in Figure 6.2.

```
/**
 * @Entity
 * @Table(name="message")
 */
class Message
{
    /**
     * @Id
     * @Column(type="integer")
     * @GeneratedValue
     */
    private $id;
}
```

Figure 6.2: Simple entity configuration example

6.3 Application architecture

I chosen to use *Nette Framework*[7] for my application because it is an open source complex web framework based on PHP and it has a large community of people that provide support to each other. It has been founded and developed by Czech programmer David Grudl, nowadays it is maintained by the Nette Foundation. The current and used version of Nette is 2.4 which runs on PHP 5.6 and above. The framework is aiming at eliminating security risks, and supports the politics of using reusable components and reducing redundant code. Nette supports MVC architecture. *Model-View-Controller* is a software design pattern built around the interconnection of three main component types, in the programming language such as PHP, often with a strong focus on object-oriented programming (OOP) software paradigms [24].

6.4 Modules

The application is divided into several modules: **Default**, **Project**, **Page**, and **Campaign**. Each module represents a separate logical block which has its own templates and presenters (*views* and *controllers* according to the MVC architecture terminology).

Each module has also its own *abstract BasePresenter* which extends the parent abstract BasePresenter. The parent abstract BasePresenter contains the logic of other presenters' accessibility. For example, if the user is not logged in, he can see only log in and privacy policy page. Even if he tries to access the page via address bar, he is redirected back to

[5] Official website: https://www.doctrine-project.org/
[6] Dockblock is a comment which starts with /** and ends with */
[7] Official website: https://nette.org/

the login page. The children BasePresenters in modules are usually used for transmitting persistent parameters (locale, translator, project id, page id, campaign id, etc.) to other presenters that extend them.

6.4.1 Default module

Default module contains mainly presenters that can be accessed without login. These presenters are:

- **LogInPresenter** with Facebook Login component
- **WebhookPresenter** that handles events from Facebook
- **CronPresenter** that periodically sends messages that are set to be sent
- **PrivacyPolicyPresenter** with the privacy policy that is required by Facebook

There are also parts which are accessible only after successful login:

- **PropertiesPresenter**
- **ProjectsPresenter**

These presenters are for properties and projects management and include options like add, edit and delete.

6.4.2 Project module

The Project module consists of only one presenter called **DetailPresenter**. It includes options like add, edit and delete pages which are assigned to the particular project.

6.4.3 Page module

The Page module represents the Page entity, its settings, and information about it. It is composed of five presenters: **DetailPresenter**, **LinkPresenter**, **PeoplePresenter**, **ConversationPresenter**, and **CampaignsPresenter**. A description of the implemented components follows.

Page detail

The page detail preview can be seen in Figure A.1. If the page has already some campaigns, a chart with campaigns' statuses is rendered here. The chart is generated by the JavaScript library *Chart.js*. And when the page has already some people, there is also a list of first 30 of them.

Conversations list

The page's conversations list is handled by using the Graph API. The conversations are categorized and retrieved individually by the person. The change of the person is handled by AJAX.

6.4.4 Campaign module

This module represents campaign entity, its settings and management. It is composed of 4 presenters in total: **DetailPresenter**, **MessagePresenter**, **RecipientsPresenter**, and **ResponsesPresenter**. The next paragraphs describe the most important parts of the module.

Campaign detail

This section of the application shows detailed information about the particular campaign. The displayed information is different in order to the status of the campaign.

As Figure A.2 shows, if the campaign is not set yet, there are help bubbles with advice what to do to set up the campaign. Firstly, the recipients have to be chosen and the message has to be set. Secondly, since every campaign has to have its start and end date, it has to be set too. This can be easily done via JavaScript *DateTimePicker* tool. Once the start and end date are set, the campaign is ready to be sent to the recipients. The mechanism of sending messages is described later in section 6.4.4.

On the other hand, as Figure A.3 shows, if the campaign has already some recipients, there is a list of first 30 of them. And if it is already running or ended and has some collected responses to the sent message from recipients, there is a chart with the responses and their counts. The chart is generated by the JavaScript library *Chart.js*.

Campaign recipients

Same as in campaign detail, the information shown in this part of application depends on the status of the campaign. A preview of the recipients settings is shown in Figure A.4.

If the campaign has not started yet, there is a list of available recipients which is actually the list of all people assigned to the campaign's page. The recipients can be added and removed both individually and in groups. There is an option to filter people by their properties which can be very useful for better targeting of the campaign onto the people that are interested in the campaign topic.

For example, people were asked whether they want to buy a car in the near future. Their responses were stored to the property called *new car 04/18*. The values of this property can be now used in the next campaign where the new cars will be offered. Logically, it is unnecessary to send this offer to people who do not want to buy a car. On the other hand, it is very desirable to target those who selected *yes* in the *new car 04/18* property. There is a bigger chance that those people will buy it and the campaign will be successful.

However, when the campaign is already running (or ended), there is only a list of recipients and status whether the message was sent to them or not. There is no option to change something.

Responses to campaign

If the campaign has the message type *text*, it can also have 0-11 *quick replies* specified in the message settings. These replies can be stored in a property which can be also specified in the message settings. The responses are then stored in the database and shown in the section Responses. They can be filtered and exported to CSV file for later use.

Message configurator

The message configurator is a component which handles message settings. It is composed of message type select box and 7 subcomponents. Each subcomponent represents some Facebook message template type: text, generic, list, button, open graph, media, and JSON. The whole component is handled by AJAX because it seems more user-friendly. When the message type is changed, the subcomponent will be changed too. Since each Facebook message template has its own fields and rules about required and optional ones, it was also necessary to reflect these rules to each subcomponent configuration to avoid misconfiguration from user input.

As it can be seen in Figure 6.3, there are message settings (based on the selected message type) on the left side and the message preview on the right side. The message preview is being refreshed by AJAX after every click on *Save* button.

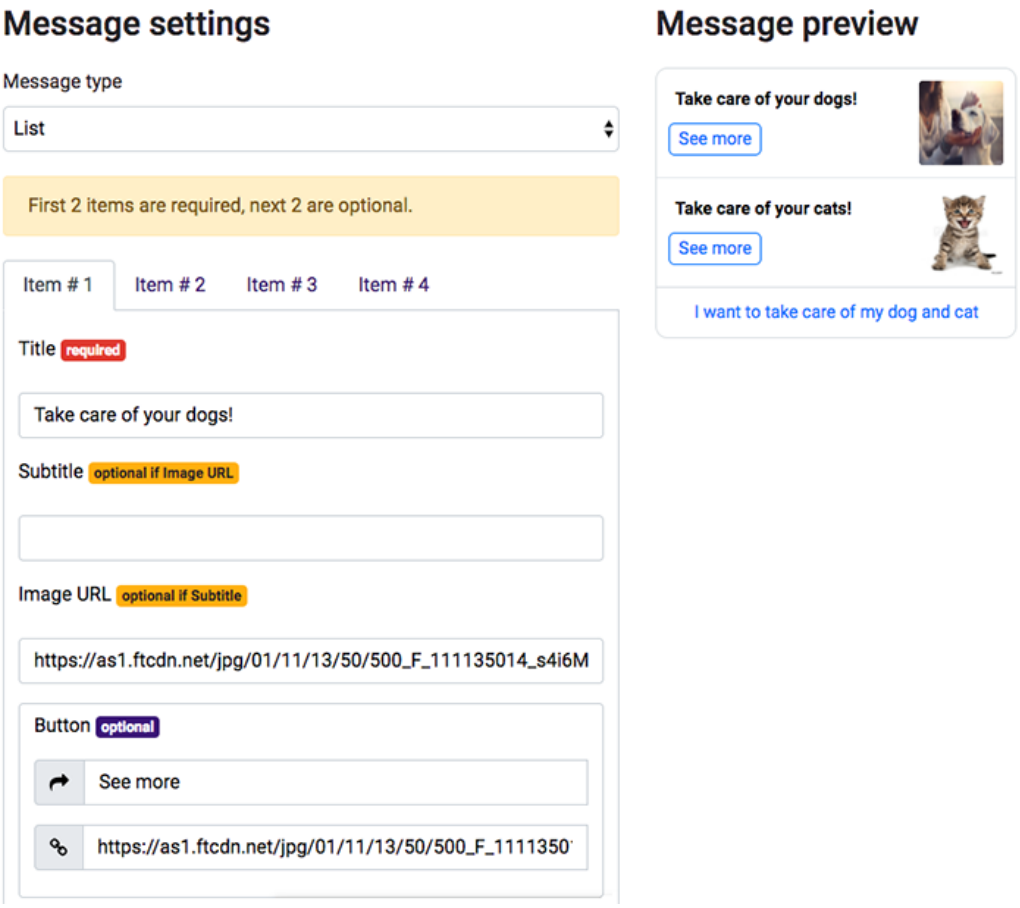

Figure 6.3: Message configurator, list template settings and preview

Cron

Cron is a time scheduler which executes a script at the set time or periodically. The cron job runs from an external server and accesses a page dedicated to cron jobs. Here it is the *CronPresenter* in Default module because it has to be accessible without login and all of these accessible presenters are located there.

The cron was chosen to be used for sending messages. As the application requirements in Chapter 3 say, the campaign can have hundreds or thousands of recipients. This means that the same count of the messages has to be sent. The execution time of such request could be very long, so the messages are sent in batches of 250. The cron is being executed every 5 minutes, so 3000 messages can be safely sent per hour.

Once the start and end dates of the campaign are set, it is ready to be sent to the recipients. The recipients are handled by the FIFO[8] mechanism. It means that the cron searches for recipients who have campaign start date lesser than now and the null message sent time. These recipients are ordered by campaign start date in ascending order, so the ones with earlier start date are served first.

6.5 Table data representation

Because the application has to display large table data (e.g. a list of recipients), it was necessary to choose the appropriate way to do it. The *Ublaboo datagrid*[9] plugin has been used for that purpose. This plugin receives data from database and structures them into table view which can be seen in Figure 6.4. It also supports pagination when there is a lot of records, filtering and sorting by columns, exporting (filtered or all records) and even adding new items and editing the old ones.

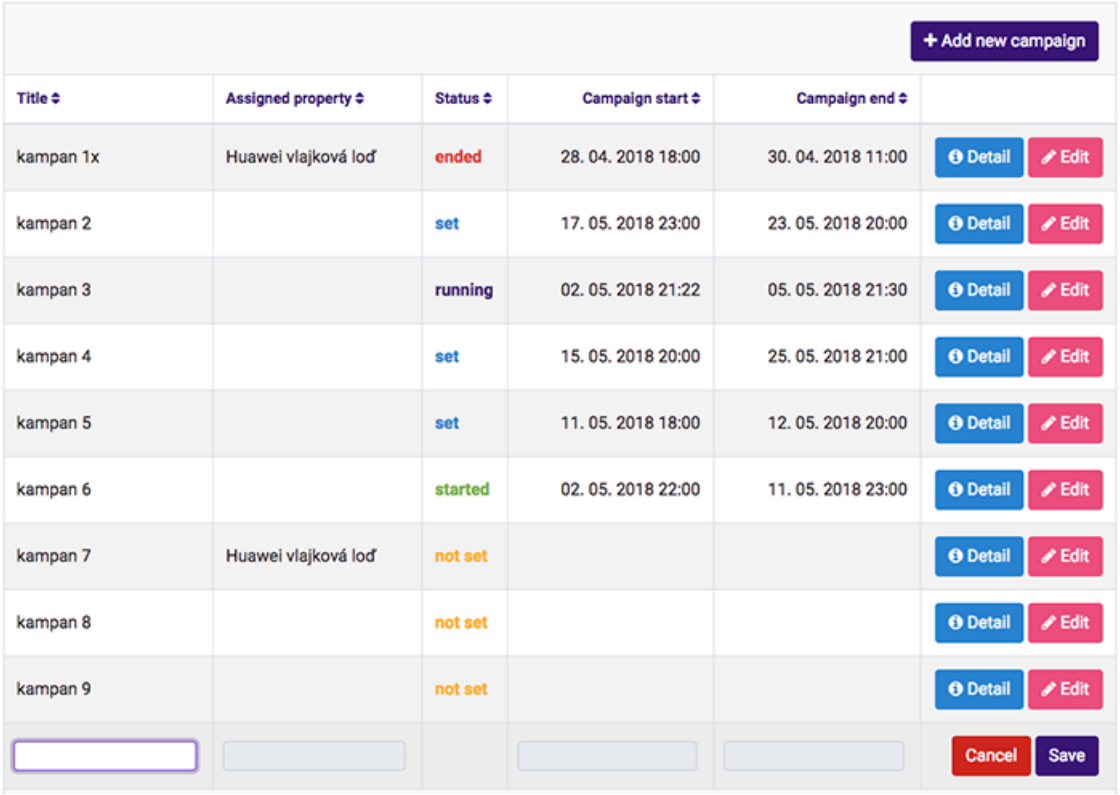

Figure 6.4: Table view of campaigns by Ublaboo datagrid, CSS style by Bootstrap

[8] first in, first out
[9] Official website: https://ublaboo.org/datagrid/

6.6 Localization

The translation system was implemented to the application because of the possibility of international usage. Nette Framework has its own translator interface, only a class which implements the `Nette\Localization\ITranslator` interface is needed. For that, I chose to use *Kdyby/Translation* extension.

6.6.1 Kdyby Translation

The translations are stored in the `lang` directory which is situated in the application directory. There are the translation files using NEON syntax[10], which is also used in Nette configuration files. Each filename is composed of the translation section name and ISO 639-1 and 3166-2 language codes. For example, the Czech file with translations is named `as.cs_CZ.neon` and the English file is named `as.en_US.neon`. The usage of the translator in templates is provided by `{_'as.projects.title'}` macro where `as.projects.title` is the address of translation in the files `as.cs_CZ.neon` and `as.en_US.neon`.

[10] https://ne-on.org/

Chapter 7

Testing and future development

7.1 Testing

The application was being tested all the time of the development. First only on my computer via my personal Facebook account. Then the application was deployed to the test website ajdinas.cz/bt, and the test user with some test pages was created on Facebook.

The manual testing of inputs and scenarios that could happen while using the application was done after every more complex change in the implementation. The proper function and behavior such as project setup, page setup, campaign setup, correct message configuration, and recipients settings were tested. The next subject of testing was sending messages using the cron.

The most of the syntactical errors during implementation was caught by the PhpStorm IDE, the others were caught by Nette debugger called *Tracy*. Tracy was very helpful because it is a high-quality debugger that usually comes up with the error alert which can help to understand where the error is.

According to the Facebook's privacy policy, all new Facebook applications that need `manage_pages`, `read_page_mailboxes` and `pages_messaging` permissions[1], are required for review before deploy. The application review process was paused by Facebook until May 1st, 2018. The application is now ready for submitting for review which has to be done until August 1st, 2018. After the successful review, it will be deployed and used with real data in the test mode for a while. Then it will be used with production data by Q2 Interactive.

7.2 Possible extensions

I got some ideas about several extensions during the application development that could be implemented in the future. Some of them are changes in the application logic, others could simplify or improve the user interface.

7.2.1 Image upload

There is no option to upload own images in the message settings, there is only a field for URL. It means that the image has to be hosted anywhere else. Some user-friendly image uploader is definitely a *nice-to-have* extension that could be implemented in the future.

[1] These permissions are used by my application.

7.2.2 More messages in the campaign

Sometimes it would be nice to have the option to send more messages that are dependent on each other. For example: First, send a question whether the person wants to communicate. If he chooses *yes*, then send another message. But if he chooses *no*, do not send any other message. By this, the good user experience is performed which is the most wanted behavior that the Messenger desires when using automated messages.

7.2.3 Quick replies

The quick replies are now available only with text message because Facebook does not support them natively with other templates. But there is a trick to get around this by adding a text message after the template message. For example, the added text message can say *Choose one of the suggested replies:* to inform the user about the option of quick reply. Considering that collecting responses to messages is a very important function of this application, this extension is certainly very welcome and planned to be implemented. It is also dependent on the previous extension.

Chapter 8

Conclusion

The purpose of this thesis was to design and implement the management system for Facebook marketing campaigns. At first, I became familiar with the principles of online marketing that are being used today. Then I have studied the existing solutions and deducted the advantages and disadvantages of them in comparison to the information that emerged up from the system requirements analysis. This resulted in the design and later implementation of the new application based on PHP framework Nette with MVC architecture.

This application offers a way how to manage campaigns running via Messenger, configure their messages, associate the campaigns to the pages that are linked with Facebook pages and also associate the pages to the projects.

From a marketing point of view, the benefit of this application is that the responses to the sent messages can be saved to the database. Then they can be used to better target recipients in the next campaigns. This could increase the success of the campaign and also the success of the business.

The application will be further developed as a project in Q2 Interactive company. It is running currently in the test mode on ajdinas.cz/bt[1].

[1] Available on: https://ajdinas.cz/bt

Bibliography

[1] About Chatfuel.com. Create a Facebook AI Chatbot Without Coding. [Online; visited 2017/04/12].
Retrieved from: https://chatfuel.com/about-us.html

[2] Ads Manager. [Online; visited 2018/01/22].
Retrieved from: https://www.facebook.com/ads/manager

[3] API - Register & Sign In With Facebook | Dev Costs. How much does it cost to make an app. [Online; visited 2017/05/02].
Retrieved from:
http://www.devcosts.com/features/api-register-sign-in-with-facebook/

[4] Best PHP IDE in 2014 - Survey Results — SitePoint. SitePoint. [Online; visited 2017/04/23].
Retrieved from:
https://www.sitepoint.com/best-php-ide-2014-survey-results/

[5] Bootstrap · The most popular HTML, CSS, and JS library in the world. [Online; visited 2018/01/09].
Retrieved from: https://getbootstrap.com/

[6] Databases and Doctrine (Symfony 2.5 Book). Symfony, High Performance PHP Framework for Web Development. [Online; visited 2017/05/01].
Retrieved from: https://symfony.com/doc/2.5/book/doctrine.html

[7] DB-Engines Ranking - popularity ranking of database management systems. [Online; visited 2018/04/05].
Retrieved from: https://db-engines.com/en/ranking

[8] Facebook Developer Documentation - Facebook for Developers. [Online; visited 2018/01/28].
Retrieved from: https://developers.facebook.com/docs/

[9] Facebook Login with ASP.NET Web Forms – Nick Pinheiro. MSDN Blogs. [Online; visited 2017/04/25].
Retrieved from: https://blogs.msdn.microsoft.com/nickpinheiro/2015/02/28/facebook-login-with-asp-net-web-forms/

[10] gulp.js. gulp.js. [Online; visited 2017/04/23].
Retrieved from: https://gulpjs.com/

[11] Introduction - Composer. Composer. [Online; visited 2017/04/23].
Retrieved from: https://getcomposer.org/doc/00-intro.md

[12] An Introduction to CSS Pre-Processors: SASS, LESS and Stylus | HTML Mag. [Online; visited 2018/04/05].
Retrieved from: https://htmlmag.com/article/an-introduction-to-css-preprocessors-sass-less-stylus

[13] Leading global social networks 2018 | Statistic. [Online; visited 2018/02/18].
Retrieved from: https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/

[14] Legal requirements for email marketing - TermsFeed. [Online; visited 2018/02/18].
Retrieved from:
https://termsfeed.com/blog/legal-requirements-email-marketing/

[15] Motion AI. Motion AI. [Online; visited 2018/01/21].
Retrieved from: https://docs.motion.ai/docs/

[16] npm. npm. [Online; visited 2017/04/23].
Retrieved from: https://www.npmjs.com/

[17] Object Relational Mapper — Doctrine Project. [Online; visited 2018/04/05].
Retrieved from: http://www.doctrine-project.org/projects/orm.html

[18] What is AJAX?. [Online; visited 2017/05/02].
Retrieved from: https://www.tutorialspoint.com/ajax/what_is_ajax.htm

[19] What is Entity Relationship Diagram (ERD)?. Visual Paradigm. [Online; visited 2017/04/23].
Retrieved from: https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/

[20] What is object-relational mapping (ORM)? - Definition from WhatIs.com. .Net development information and Visual Basic / VB resources - SearchWinDevelopment.com. [Online; visited 2017/04/30].
Retrieved from: https://searchwindevelopment.techtarget.com/definition/object-relational-mapping

[21] Duckett, J.: *HTML & CSS: design and build websites*. John Wiley & Sons, Inc. 2011. ISBN 9781118008188.

[22] Maslakowski, M.: *Naučte se MySQL za 21 dní*. Computer Press. 2001. ISBN 8072264486.

[23] Philip, K.; Gary, A.: *Principles of marketing*. Boston: Pearson Prentice Hall. 2012. ISBN 9780132167123.

[24] Pitt, C.: *Pro PHP MVC*. Apress. 2012. ISBN 9781430241652.

Appendix A

Application previews

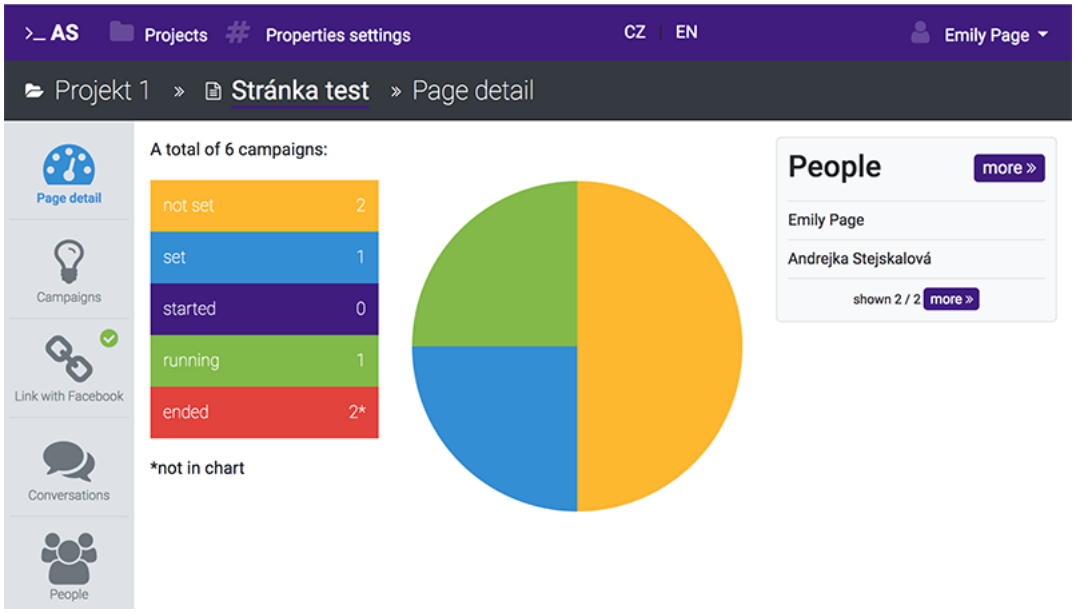

Figure A.1: Page detail

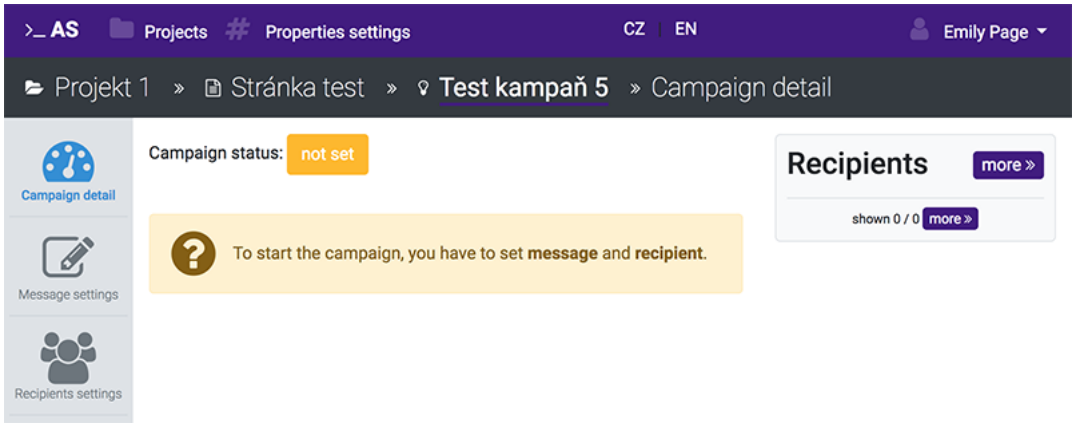

Figure A.2: Detail of the campaign when it is not set

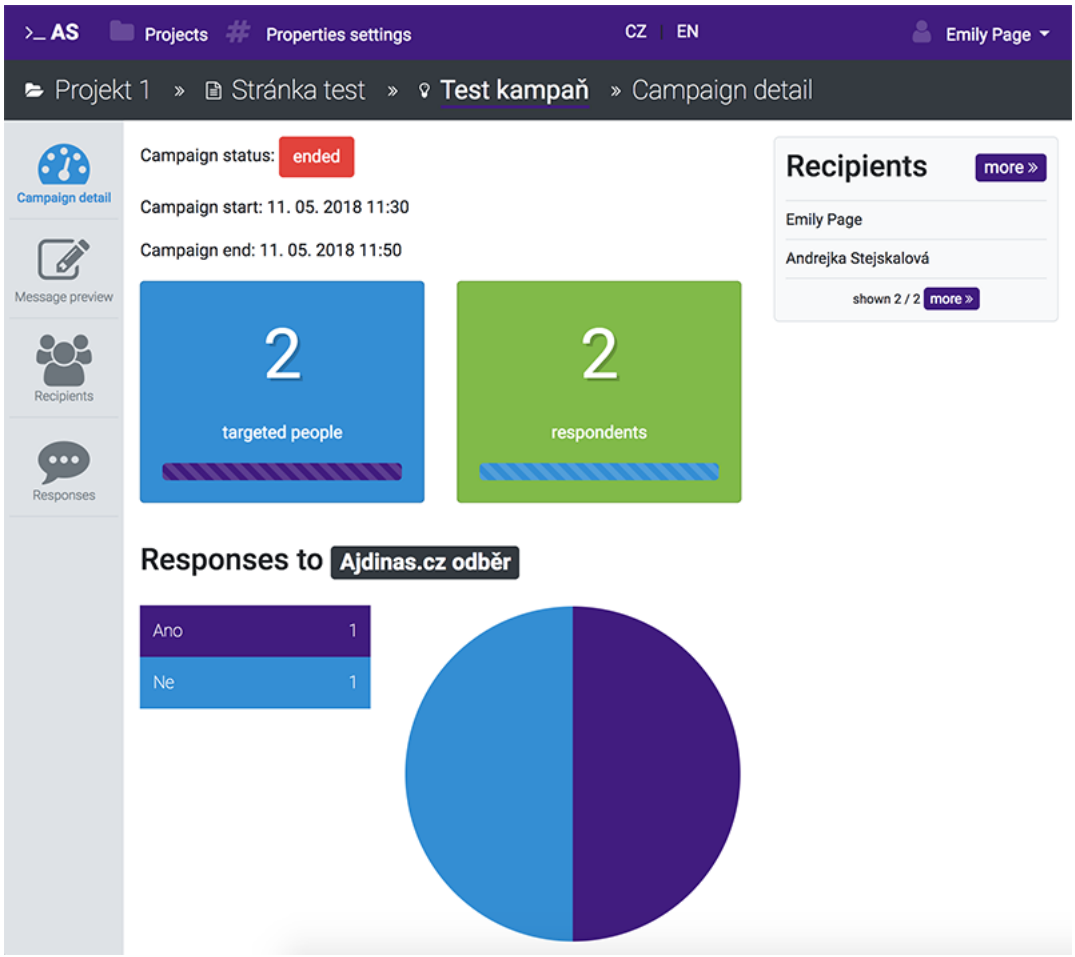

Figure A.3: Detail of the campaign when it is ended

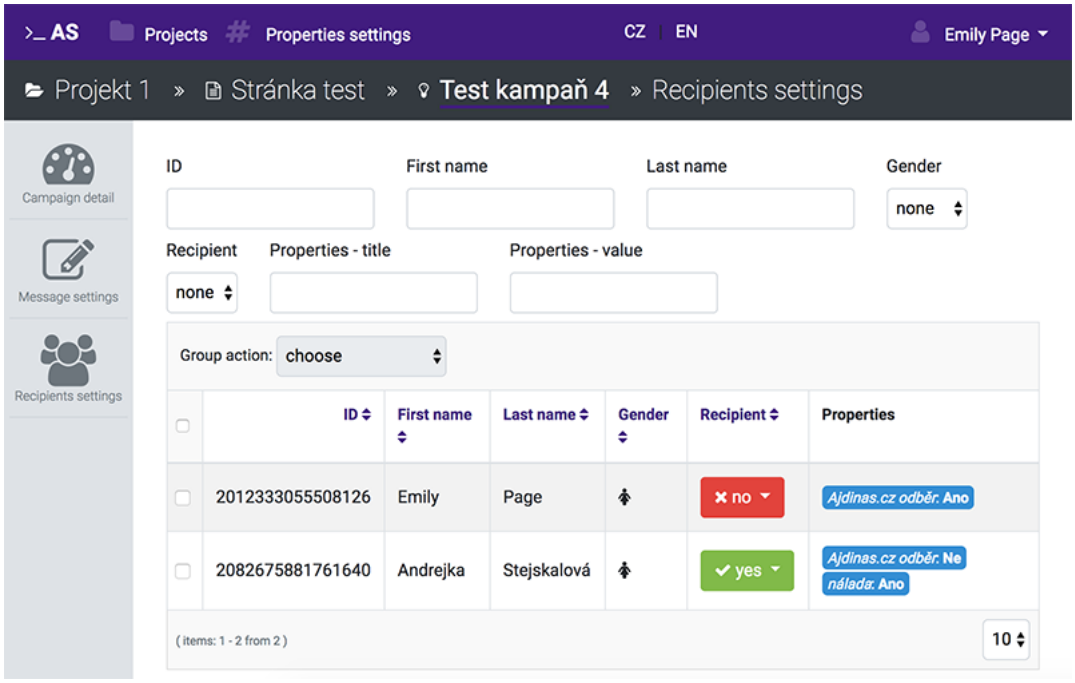

Figure A.4: Recipients settings of the campaign

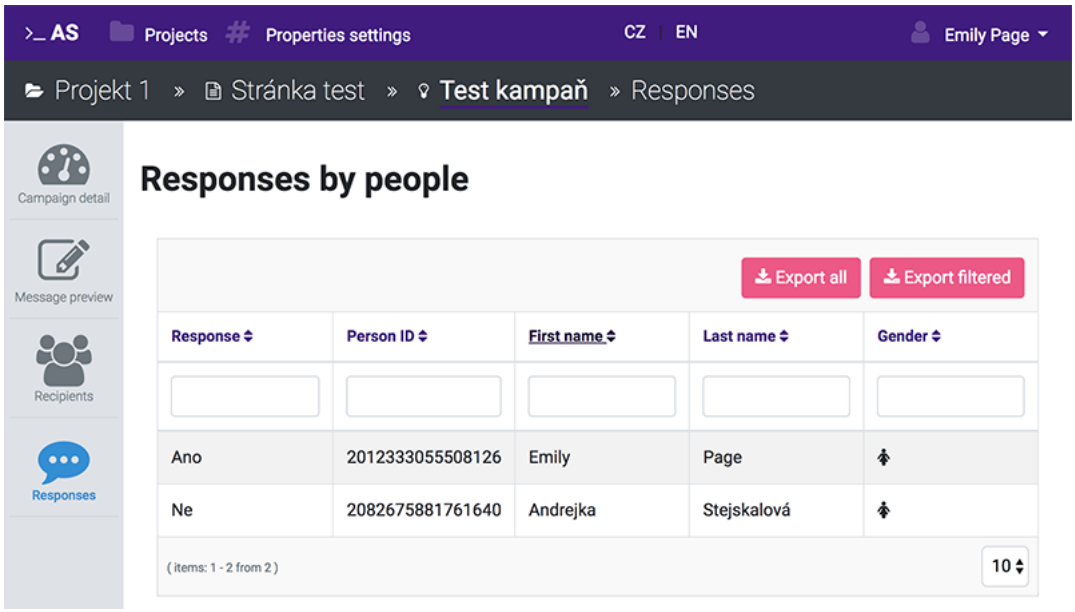

Figure A.5: List of responses to the campaign