TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Kirke Krämann 164084IABB

CLIENT IMAGE MANAGEMENT APPLICATION ANALYSIS, REDESIGN, AND DEVELOPMENT FOR INTERNATIONAL E-COMMERCE PRODUCT AFTERPAY

Bachelor's Thesis

Supervisors: Enn Õunapuu, PhD

Aleksandr Kormiltsõn, MSc

Tallinn 2019

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Kirke Krämann 164084IABB

KLIENDI PILDIHALDAMISE RAKENDUSE ANALÜÜS, UUS DISAIN JA ARENDUS RAHVUSVAHELISELE E-KAUBANDUSE TOOTELE AFTERPAY

Bakalaurusetöö

Juhendajad: Enn Õunapuu, PhD

Aleksandr Kormiltsõn, MSc

Tallinn 2019

Author’s Declaration of Originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination or submitted for defence anywhere else.

Author: Kirke Krämann

19.05.2019

Abstract

The main goal of this thesis is to improve client image management for the customer portal MyAfterPay.com, part of the AfterPay payment-after-delivery solution.

Firstly, the analysis of the requirements for the image handling application is presented. Secondly, the image handling and development processes are defined.

Finally, the result of the thesis is the definition of the AfterPay client image handling process, along with requirements for the newly developed AfterPay Merchant Image Branding application. This application allows clients to manage their brand images that are displayed to the customer on the MyAfterPay website.

The thesis is written in English and contains 34 pages of text, 6 chapters, 11 figures, and 1 table.

Annotatsioon

Kliendi pildihaldamise rakenduse analüüs, uus disain ja arendus rahvusvahelisele e-kaubanduse tootele Afterpay

Lõputöö põhiline eesmärk on parandada AfterPay kliendi pildirakenduse haldust, mida kasutab kliendiportaal MyAfterPay, mis on üks AfterPay makselahendus peale toote kättesaamist.

Esiteks analüüsida, millised on kliendi nõuded pildirakenduse haldusele. Teiseks, milline on pildihaldamise protsess. Kolmandaks, kuidas arendada uut pildihaldamise rakendust jagatud meeskonnas.

Lõputöö tulemuseks on arusaam, milline on AfterPay kliendi pildihaldamise protsess, nõuded ja arendatud AfterPay kliendi brändi turunduse rakendus, mis võimaldab klientidel ise hallata, millised brändi pildid on näitatud ostjatele MyAfterPay leheküljel.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 34 leheküljel, 6 peatükki, 11 joonist, 1 tabelit.

Glossary of Terms and Concepts

Merchant, Client	A merchant or a client is a company that operates in the e-commerce business and has integrated with AfterPay [1]
Customer	A customer is a person or a company who has bought from merchants and is now using MyAfterPay [1]
Brand image	An image creating an impression of a brand's personality for consumers' perception. Also called as hero image at the top of the website [2]
Logo	A regognizable, distinctive grapic design element with a name, sybol or trademark, which identifies a product or service [3]
BPMN	Business Process Model and Notation, a method for the illustration of business processes [4]
ASP.NET/.NET	.NET is a developer platform with tools, programming languages, and libraries. ASP.NET extends the platform to web applications [5]
MVC	Model-View-Controller architectural pattern for user interface development [6]
HTML	Hyper Text Markup Language, describes the structure of web pages using markup
CSS	Cascading Style Sheets, style elements for HTML

Table of Contents

List of Figures

List of Tables

1 Introduction

AfterPay is an e-commerce payment-after-delivery provider, which allows customers to pay for their purchases after getting the products and services. AfterPay offers alternative payment methods through the MyAfterPay.com customer portal. For the merchants, AfterPay has an API to integrate the service with web shops to offer another payment method. [7]

On the MyAfterPay website, there are several places where brand images are shown in different sizes. AfterPay would like to have loyal clients and customers. For this purpose, merchants are promoted to customers using logos, brand images, and banners in the consumer communications and MyAfterPay customer portal.

The problem with the present application is that it does not deliver all the client needs for image management. The image handling process is overly complicated, going through multiple actors, and the image tool does not meet the needs of the e-commerce merchants.

The main purpose of the thesis is to analyse the current image tool, investigate the need for a new image tool, and use the resulting requirements to develop an updated and improved version of the image tool.

The main objective of the paper is to concentrate on the following questions:

How to improve client image management for the AfterPay e-commerce platform?

1. What are the client requirements for the image management tool?
2. What is the client image management process?
3. What is the development approach for the client image management tool in a distributed team?

The need for the paper came from client request for an improved image handling requirements and behaviour driven development approach implementation. The paper methodology is based on AfterPay client communication, analysis and tool development, supported by research theory.

The remainder of the paper is structured as follows. Section 2 provides an overview of AfterPay and the importance of image management for the AfterPay system. It also contains a description of the current image management application and image handling process. Section 3 discusses the requirements for the image management process and contains use cases and Behaviour Driven Development (BDD) scenarios for image management. Section 4 defines the development process of the image tool in a distributed team. Section 5 provides an evaluation of our approach, and concludes the paper by providing directions for future research

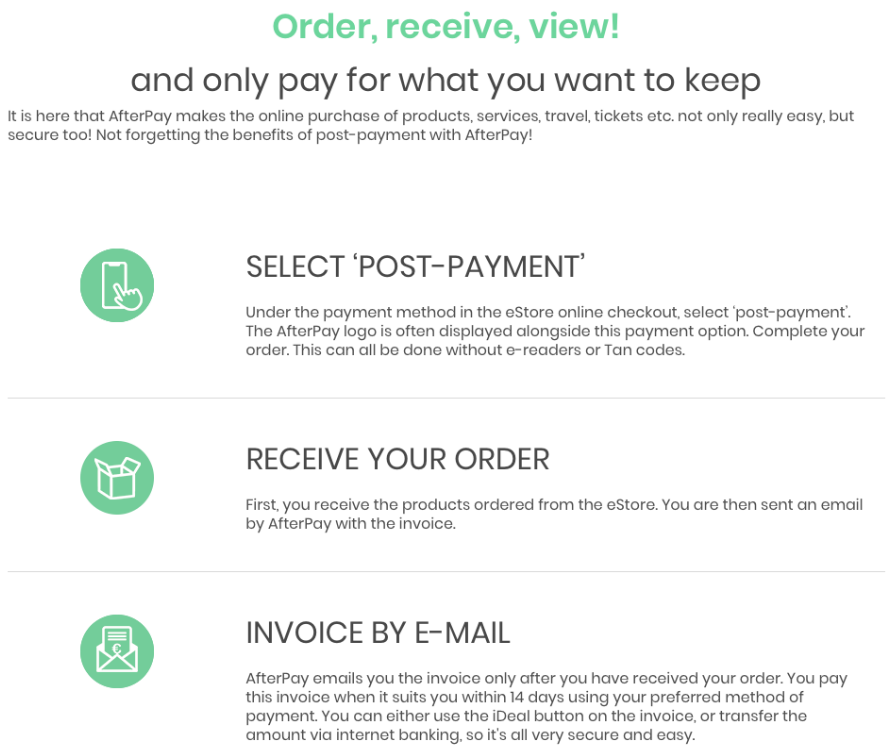
2 Background

This section introduces the AfterPay system and the importance of brand images for clients and customers. An illustration how the present image handling process goes and an introduction of the image tool.

2.1 Overview of AfterPay

AfterPay is a pay-after-delivery solution by Arvato Financial Solutions. It allows omni-channel businesses to separate checkout and payment, and takes over the credit and fraud risk from the merchant. Making payments should be quick and easy: an increasing number of customers make purchases on mobile devices, so it is important to make the checkout process in web shops as simple as possible. For on-the-go consumers, frequent shoppers or the ones who want to experience the goods, it allows them to delay the moment of payment. [8]

In Fig. 1, [9] AfterPay's simplified process is presented. The customer chooses 'post-payment' on the merchant webpage, and receives their purchase and invoice. Then it is the time to start thinking about payment. [9]



Order, receive, view!

and only pay for what you want to keep

It is here that AfterPay makes the online purchase of products, services, travel, tickets etc. not only really easy, but secure too! Not forgetting the benefits of post-payment with AfterPay!

SELECT 'POST-PAYMENT'

Under the payment method in the eStore online checkout, select 'post-payment'. The AfterPay logo is often displayed alongside this payment option. Complete your order. This can all be done without e-readers or Tan codes.

RECEIVE YOUR ORDER

First, you receive the products ordered from the eStore. You are then sent an email by AfterPay with the invoice.

INVOICE BY E-MAIL

AfterPay emails you the invoice only after you have received your order. You pay this invoice when it suits you within 14 days using your preferred method of payment. You can either use the iDeal button on the invoice, or transfer the amount via internet banking, so it's all very secure and easy.

Figure 1. Simplified Overview of Pay-after-delivery

The default option is to pay the invoice within 14 days. There is also a possibility to split payments over time or by amount, pausing the payments for a certain period, or combining smaller purchases into one payment. These are available in the MyAfterPay customer portal. [7]

MyAfterPay.com[1] allows customers to finalize payments, see ongoing and paid invoices, and get an overview of completed purchases and visited merchants. The MyAfterPay customer portal allows merchants to strengthen their brand communication. On every web page, AfterPay displays the merchant's logo and brand image. It is a way to connect the merchant brand perception, brand image, logo and goods that have been bought, in order to drive repeat purchases at the merchant's web shop [10]. AfterPay has an image tool for processing the merchant's brand images.

MyAfterPay is international product operating in nine countries - Sweden, Norway, Finland, Germany, Switzerland, Austria, Denmark, the Netherlands and Belgium. The

[1] https://www.myafterpay.com/

development team is distributed and located in Estonia, Norway, Sweden and the Netherlands.

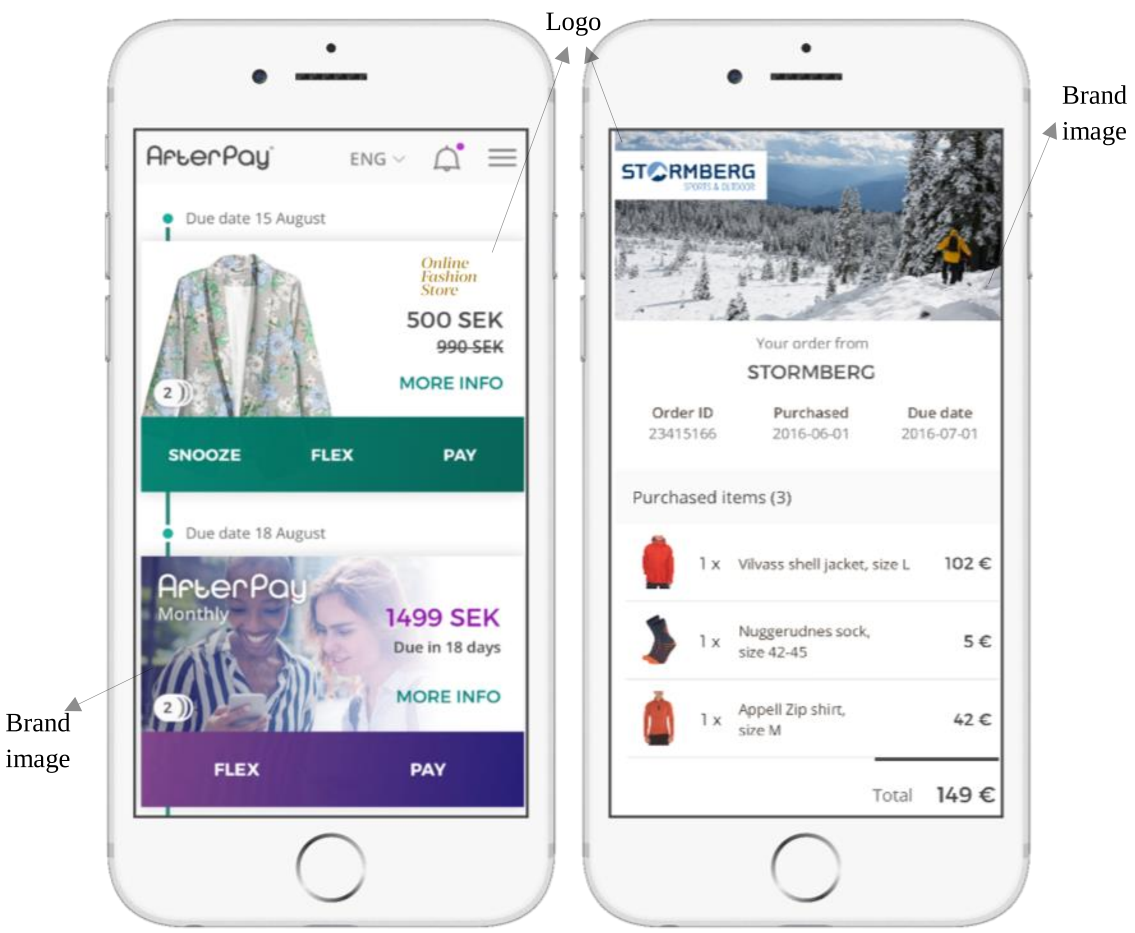
2.2 Importance of Brand Image in AfterPay

The brand image represents the enterprise, creating a vision of the products for the customer. A great brand image paints an idea of a perfect character, while showing the products, which creates a brand personality. The brand image needs to represent the product as well as the overall brand perception. [11]

Enterprises build and shape their brand’s personality. E-commerce gives a broad platform to use the brand images, in order to create better customer recognition, of what it represents, and what should be the customer perception of the merchant. [10]

The change of the brand image might be seasonal, for a holiday or a campaign, but always keeping in mind the brand and their message. And the e-commerce industry should be able to adjust to the changes.

AfterPay gives enterprises the option to show the brand image in two different proportions (5:1 and 4:2), also called the hero image [2]. The brand image is displayed in all invoices and on the MyAfterPay customer portal. Moreover, hero images are shown on a dedicated merchant overview page. The hero image represents the vision, feeling or status the client brand represents. For that reason, it is important to produce brand vision and let the merchant change the images accordingly to connect with customers.



Due date 15 August

500 SEK ~~990 SEK~~

Due date 18 August

1499 SEK
Due in 18 days

Your order from
STORMBERG

Order ID	Purchased	Due date
23415166	2016-06-01	2016-07-01

Purchased items (3)

1 x	Vilvass shell jacket, size L	102 €
1 x	Nuggerudnes sock, size 42-45	5 €
1 x	Appell Zip shirt, size M	42 €
	Total	149 €

Figure 2. Overview of MyAfterpay Client Presentation

The visual presentation in Fig.2 is how the brand images and logos are shown in MyAfterPay. The image on the left is a customer’s overview of which purchases need payment action, accompanied by the product image, a logo and a brand image. The image on the right is an example of an invoice. Every invoice has a hero image with the logo, and the purchased items.

Client association is important for creating loyal customers, who return to the merchant web shop and choose the AfterPay payment method. Customers have an overview of their purchases and visited web shops, to which they can return. Clients have another location where to attract customers.

2.3 Current Image Tool Desktop Application

The AfterPay merchant image handling tool is a desktop application that is built on the .NET framework. Desktop applications need to be downloaded to the user computer, which means different operating systems may need separate development efforts, as well as user interface design and platform limitations. [12]

Developers can only run the application through the internal network, because of company security restrictions and implementation strategy used when development started. The image change request comes from the client and goes to the merchant representative, who contacts the product owner, who must create a ticket for a developer.

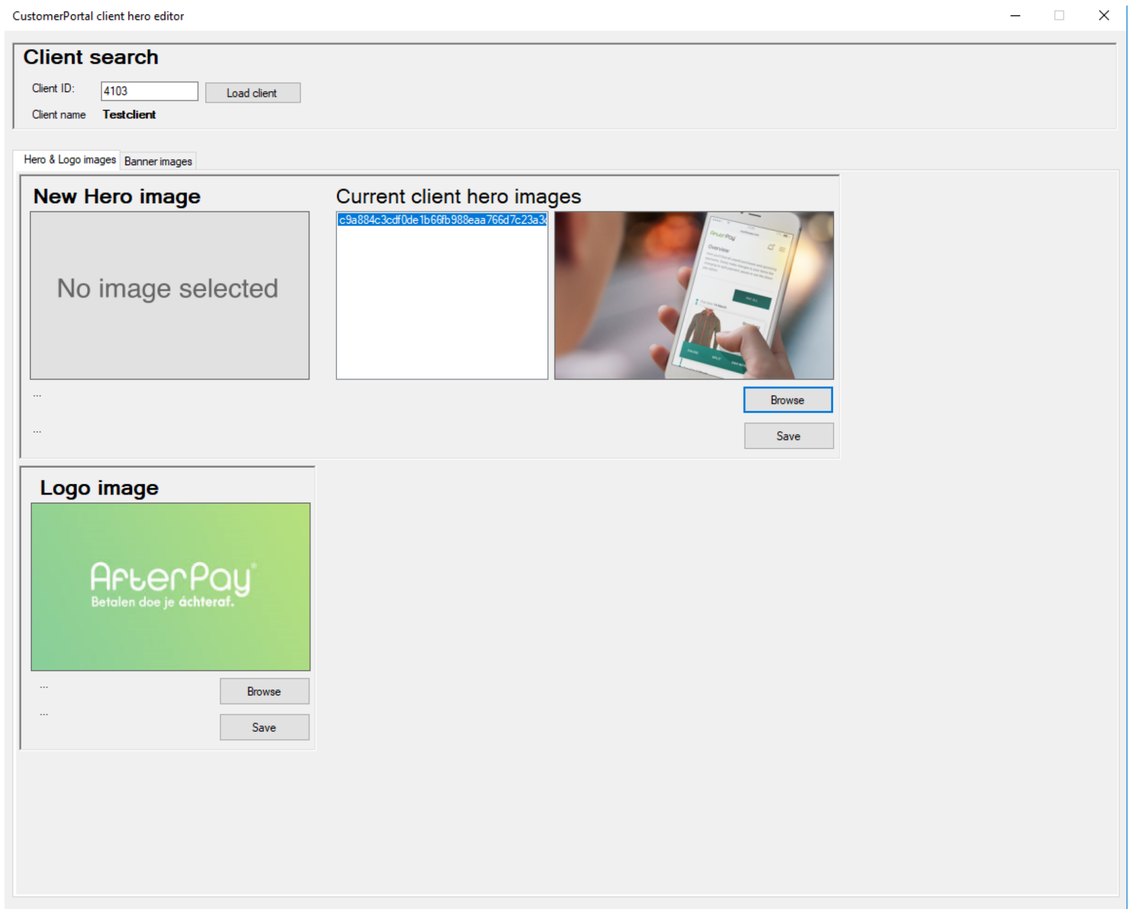


Figure 3. Desktop Image Tool Screenshot

The user interface and user experience for the tool are insufficient, consisting only of an upload button for the brand image and logo, a small preview of the uploaded image, and a hash code for identification. The user must save all the uploaded images separately.

AfterPay currently has over 5.000 clients, with new ones integrating weekly [9]. AfterPay has been promoting client interactions and associations to the customers, and the merchants want to change images seasonally or for various campaigns. These numbers indicate a growth of resources spent on uploading and managing images. With the increasing client base, image handling must be automated.

The current desktop application shows images in only one size. In the beginning, the brand images were included only in emails. However, the MyAfterPay customer portal has evolved, and client images are now displayed in different places and different aspect ratios (5:1 and 4:2).

The problem with the current image tool is the lack of adjustment possibilities. Various sizes of images may need individual approaches. It may mean cropping, zooming or positioning the images into different aspect ratios shown in MyAfterPay, depending on what the client likes to emphasize and what goes best with the brand image and vision.

2.4 Current Desktop Application Image Handling Process

Business Process Model and Notation (BPMN) is used to describe the current image handling process. BPMN [8] is a graphical representation of the business process, which can be understood by all, including business users and developers. The notation shows different actors, activities, and transactions that are required for achieving results.

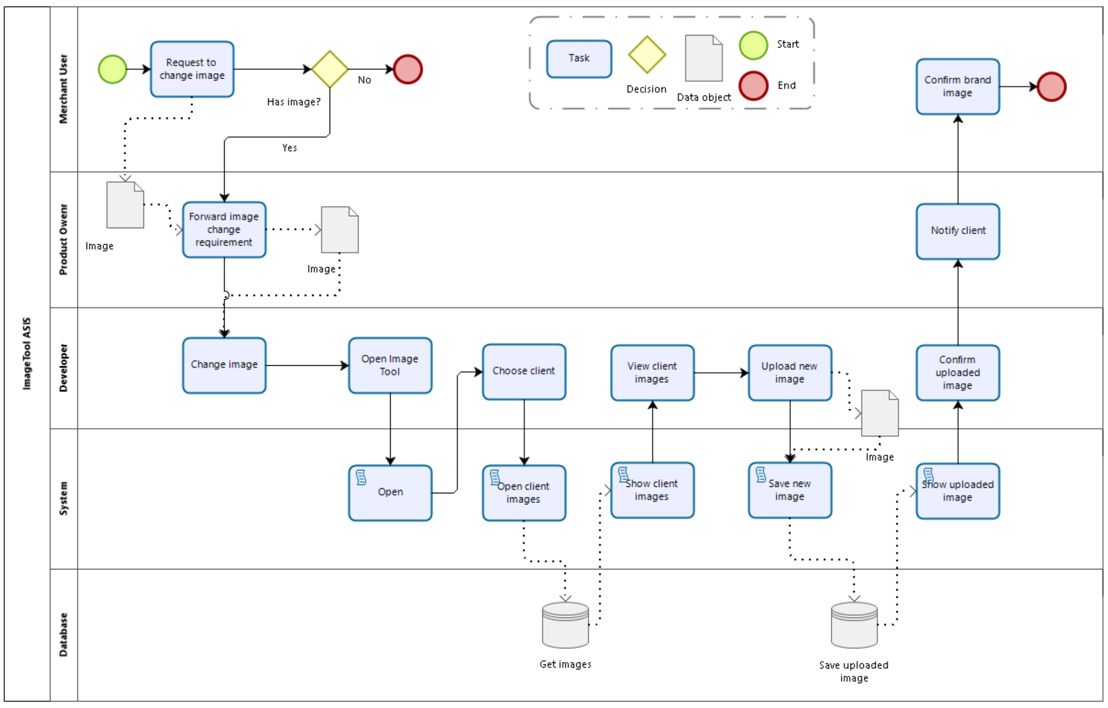


Figure 4. Old Desktop Application Image Handling the BPMN Process

In Fig.4 there are 5 actors – merchant user, product owner, developer, system (image tool desktop application), and database. The diagram presents the procedure for changing an image using the desktop application. In this context, the image can be a logo or brand image; the routine is the same.

1. The process starts with the merchant user's wish to change a brand image.
    a. If the user does not have a new image, the process is finished.
2. The product owner is between the developer and the merchant user, and forwards the request.
3. When the developer receives the task, the image handling desktop application needs to be opened.
4. After the system has opened the tool, the developer must insert the client number.
    a. The system can now get images from the database and present the images.
    b. A developer can see the present images and upload images from their computer.
5. The system saves the images to the database, and then shows the images in the application.
6. The developer must save the images and notify the merchant user, who must do the final confirmation via a product owner.

The process flow described for the current image tool using BPMN diagram.

3 Business Requirements for the New Image Tool

Business requirements for the new image tool were collected from merchants, their representatives, analysts, and AfterPay product owners. The requirements are listed and shown on the goal model. An initial application wireframe was created and described via use cases.

3.1 List of Business Requirements

Lift of requirements gathered from AfterPay business managers, clients and their representatives:

- A merchant representative should be able to handle the image management process themselves
- The merchant user must be able to log in
- Each merchant must see their previously uploaded images
- Each merchant user must be able to upload new images
- The brand images must be adjustable by cropping and zooming
- The tool must show the images in the same sizes and aspect ratios as they appear in MyAfterPay
- The image tool brand images must be integrated with MyAfterPay

The list consists of characterisitcs what the system should provide in the user point of view. The business requirements are an input for the analysts and developers.

3.2 List of System Requirements

List of requirements what the system should provide:

- A merchant user must only see their own client profile and images
- The system must save original images and adjusted brand images
- When an image is deleted in the tool, it must be deleted from the database and not retained there
- The image tool must be integrated with MyAfterPay portal

System requirements provide information about software configuration and how the system works. The system requirements are mostly for developers and software architecture.

3.3 Goal Model

The provided goal model defines the requirements for the new image tool. The main objective of the goal model is to create an understanding between business and technical stakeholders [14]. The model generates understandable domain knowledge, using actors, functional and quality goals to achieve. The goals must be specified precisely to support requirements, explanation, and evolution [15]. The functional goals represent services that the system must provide, while the quality goals show non-functional requirements reflecting the quality of the service.

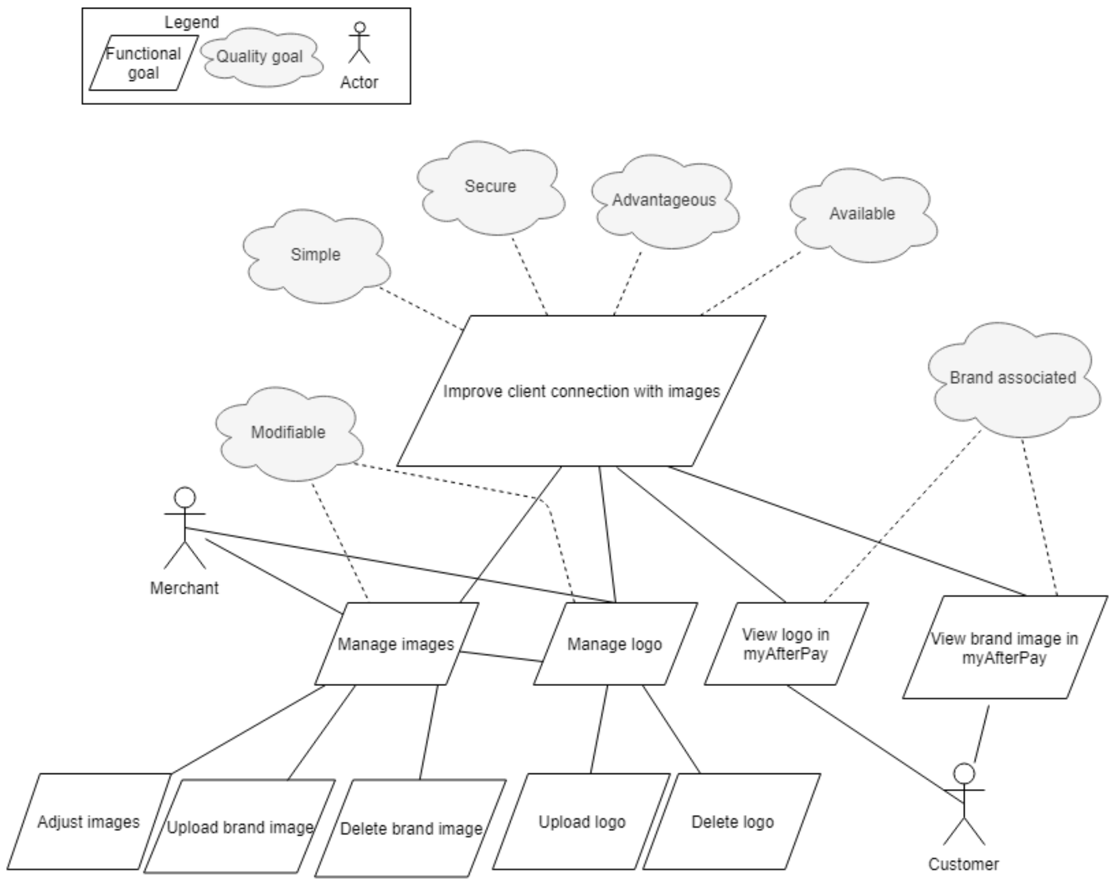


Figure 5. Value Proposition Goal Model for the Web Image Tool.

The main business requirements and quality goals are presented in Fig.5. The main goal for the application is to improve the merchant’s connection with the customer using images. Two main images in MyAfterPay are brand and logo images. The main purpose of these images is to simplify client recognition of web shops and brands. The image tool must be simple, available and advantageous for the client. The application needs to be secure. For the merchant user, image management means the ability to upload new brand images, adjust the uploaded images, and delete the images in the tool. Logo handling needs to be simple; the merchant user must be able to upload and delete logos.

3.4 New Image Tool Wireframe

Previous studies indicate that the testing process should start at the beginning of the development process and continue throughout product development. The design should improve with the product [16]. Users should be able to give feedback and contribute to the design. Wireframes offer the possibility to present a simple outline with basic functionality, which makes testing more interactive, usable and effective while being simple [17].

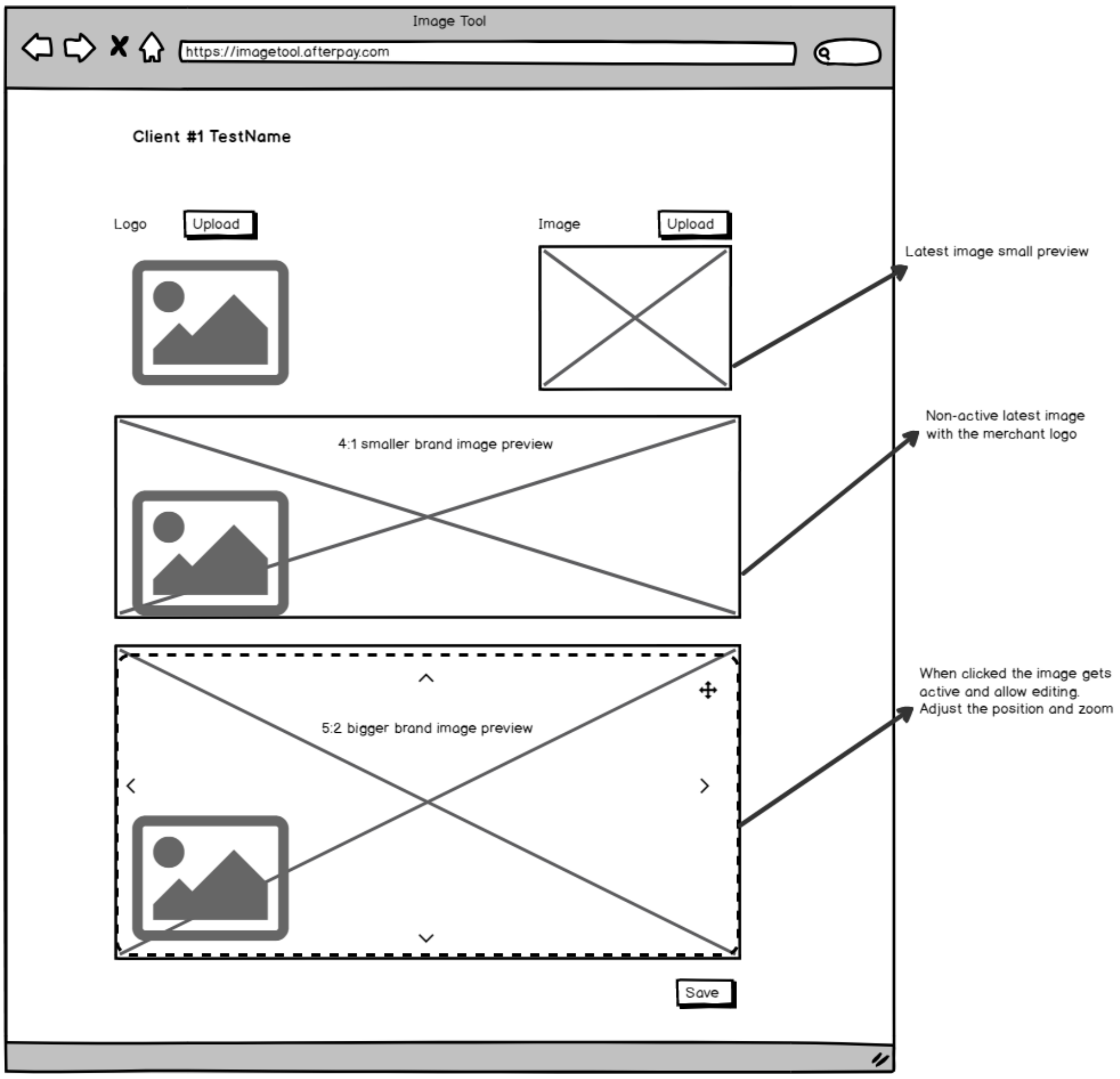


Image Tool
https://imagetool.afterpay.com
Client #1 TestName
Logo
Upload
Image
Upload
Latest image small preview
4:1 smaller brand image preview
Non-active latest image with the merchant logo
5:2 bigger brand image preview
When clicked the image gets active and allow editing. Adjust the position and zoom
Save

Figure 6. New Image Tool Wireframe

The wireframe presented in Fig.6 gives an outline of the merchant branding image tool. The user can see the client name. There is an upload button to add logo and image files from the user’s computer. The tool shows the brand images in two cropped aspect ratios, which are also shown in MyAfterPay. Clicking on one of the brand images activates it, and the user can adjust the images by positioning and zooming in and out. When the user is satisfied with the result, the images must be saved.

3.5 Use Cases for the New Image Tool

A use case diagram is a way to gather system requirements. It gives an overview of the participants and actions that can be done using the framework and system. The diagram is based on requirements and gives input for analysts and development. [18]

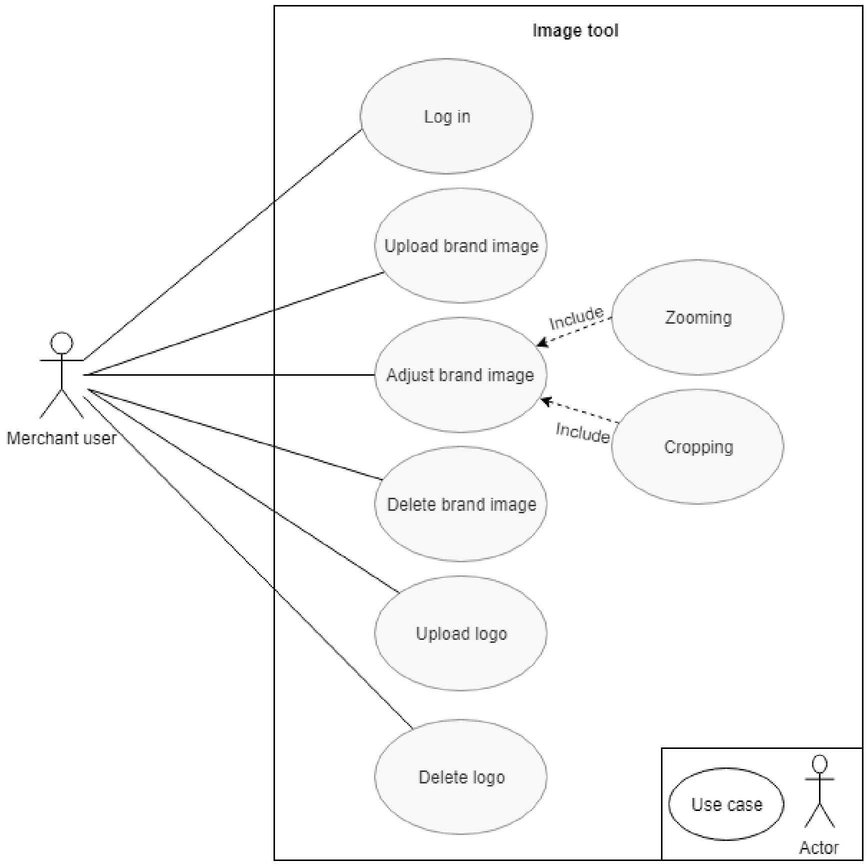


Figure 7. New Web Image Tool Use Case Model

The use case model on Fig.7 presents all the use cases supported by the image tool. The system must allow the merchant user to log in, to upload logo and brand images. The uploaded image must be adjustable by positioning and zooming. The system must allow the user to delete an uploaded image.

There are several ways of describing use cases; these use cases are described as per the Rational Unified Process (RUP) [19]. The use cases consist of the use case name, short description, actor, precondition, success scenario. In same cases, they also feature an alternative flow and postcondition. The use cases describe the actions and flows what the user should be able to do using the image tool.

3.5.1 Use Case “Log In.”

Description: Log In – Merchant user logs in to image tool with username and password. Merchant user client images will be loaded.

Actor: Merchant user

Precondition: the user has a valid username, password

Basic flow:

1. The user enters username and password
2. The user submits username and password
3. Image tool validates username and password
4. Image tool confirms the credentials
5. Image tool loads existing user images from the database
6. The user is logged in, and pictures are shown

Extensive flow:

3.1. Username or password is incorrect

4.1. Image tool shows an error and invalidates log in process

5.1. Image tool loads the page without any images

Postcondition: image tool is loaded and ready for uploading images

3.5.2 Use Case “Upload Brand Image.”

Description: Upload Brand Image - user can upload images from their web browser on image tool website. The uploaded brand image should be visible for the user.

Actor: Merchant user

Precondition: Merchant user is logged in

Basic flow:

1. The user clicks Upload brand image button
2. The user uploads image from their computer
3. Image tool displays uploaded image preview
4. Image tool saves uploaded image to database
5. Image tool shows uploaded images in different sizes

Extensive flow:

2.1 System shows an error when the uploaded file is too big

Postcondition: Merchant image is uploaded and visible to the user on the image tool

3.5.3 Use Case “Upload Logo.”

Description: Upload Logo **-** user can upload logo form their web browser on image tool website. The uploaded logo should be visible for the user.

Actor: Merchant user

Precondition: Merchant user is logged in

Basic flow:

1. The user clicks Upload logo button
2. The user uploads logo file from their computer
3. Image tool displays uploaded logo preview
4. Image tool saves uploaded logo to database
5. Image tool shows the uploaded logo on different size brand images
6. User logo is displayed on MyAfterPay

Extensive flow:

2.1. The system shows an error if the uploaded file is too big

Postcondition: Merchant logo is uploaded and visible to the user on image tool and MyAfterPay

3.5.4 Use Case “Adjust Brand Image.”

Description: Adjust Brand Image - the user can adjust brand images by zooming and positioning. The adjusted brand image must be saved and visible to the user.

Actor: Merchant user

Precondition: Image is uploaded and visible on two sizes

Basic flow:

1. The user selects an image which to adjust
2. The user zooms image
3. The user positions image
4. The user confirms the adjusted brand image
5. Image tool saves the brand adjusted image to database

Postcondition: Adjusted brand image is shown on image tool and MyAfterPay and saved to the database

3.5.5 Use case “Delete Brand Image.”

Description: Delete Image - the user can delete previously uploaded brand images.

Actor: Merchant user

Precondition: the user has uploaded brand images

Basic flow:

1. The user selects a brand image
2. The user clicks Delete button on the image
3. Image tool deletes the brand image from the database

Postcondition: Image is deleted from the image tool and database

3.5.6 Use Case “Delete Logo.”

Description: Delete Logo - the user can delete previously uploaded logo.

Actor: Merchant user

Precondition: the user has uploaded a logo

Basic flow:

1. The user selects a logo
2. The user clicks Delete button on the logo
3. Image tool deletes the logo from the database

Postcondition: The logo is deleted from the image tool and database

4 Client Image Tool Development Process

Based on the collected problems, requirements collection, and process analysis and written use cases, the team decided to start the development with behaviour driven development (BDD) features and scenarios creation.

4.1 Behaviour Driven Development

BDD is an agile development analysis approach for development purposes. BDD approach focuses on business value in addition to testing and testability. It can be implemented in different stages of the process: during requirement collection, in analysis and implementation process. [20]

The user story for the feature definition:

As a (role)
I want (feature)
So that (benefit or value).

The user story should answer the questions [20]:

- What is the role of the user?
- What feature does the user want?
- What benefit or value can the user gain if the system provides the feature?

It is important that the user story's behaviour is acceptance criteria, but it should be fragmented to scenarios. Scenarios should describe specific contexts, workflows, and outcomes of the user story [20]. Scenarios in the BDD are the acceptance criteria. Therefore it is important that every scenario result or "then" is testable. In order to formulate tickets, the scenarios are separated with @, the format for one ticket scenarios are separated with @Scenario-caption.

The scenario format:

@Scenario-caption
Given (initial context)
When (event occurring, action)
Then (result)
And (another result) [21].

Behaviour driven development takes client requirements, business requests, analysts scenarios, and developers knowledge and formats it to a feature on in this case, client image management tool.

AfterPay development teams have just started to implement behaviour driven development, writing the user stories and scenarios. The requirements are discussed with the business using the sequence diagram and scenarios are aligned with the development team, so everyone has an understanding of what needs to be done.

The first step was to define the feature, followed by the Unified Modelling Language (UML) sequence diagram, which gives an overview of actors and the process [22]. After establishing the sequence diagram, follows writing scenarios and creating tickets.

Feature: Image tool

As a merchant

I want to have image tool (What?)

So that I can handle images (Do what?)

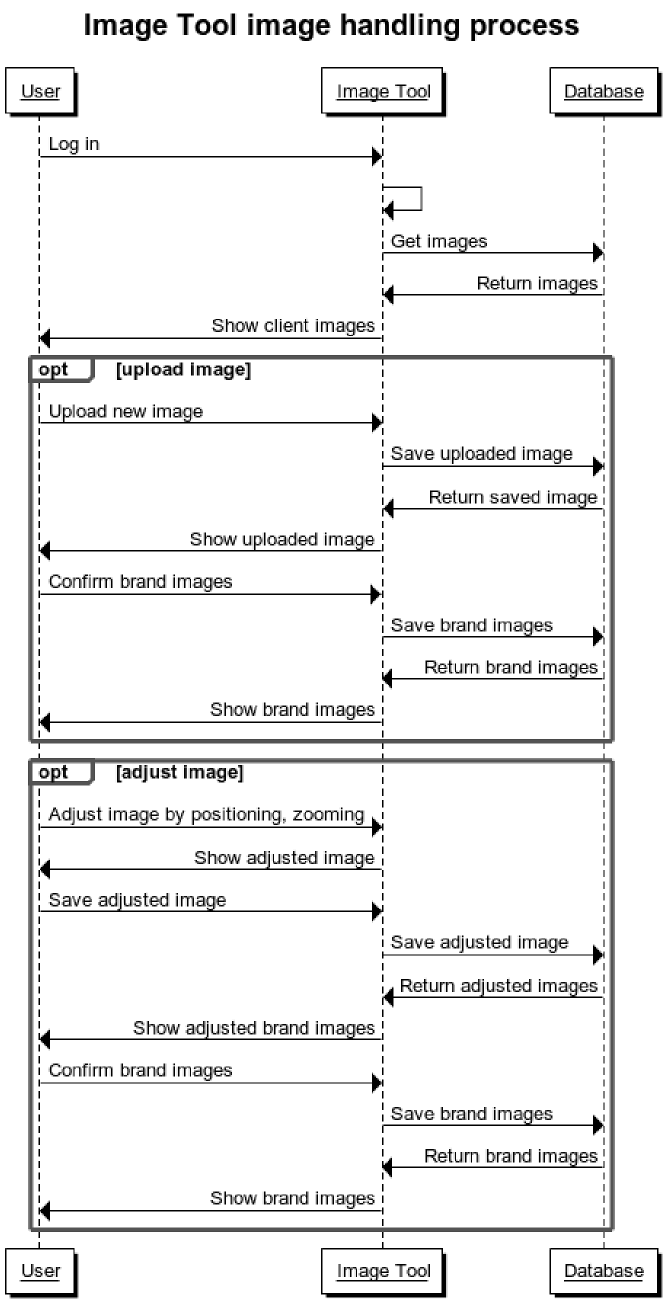


Figure 8. New Web Application Sequence Diagram

The sequence diagram in Fig.8 illustrates, what are the interactions between different actors [22]. The process of drawing the diagram takes input from the business, analysis, and developers, whom all need to have the same understanding of how the process flows and who are the actors on each step. As image and logo loading does not differ from a development point of view, it is simplified to image loading.

@ Image-tool-authentication

Scenario: the merchant is unable to log in

Given merchant is on the image tool login page

When a user logs in with wrong credentials

Then the user is shown an error with message text

@ Image-tool-authentication

Scenario: merchant logs in to image tool

Given merchant is on the image tool login page

When a user logs in with a username, password

Then the user is logged in

@Images

Scenario: Load page IF images do not exist in MyAfterPay DB

Given merchant does not have images in MyAfterPay DB

When Get images request is sent to MyAfterPay DB

Then empty image page is loaded

@Images

Scenario: Show images if it exists in MyAfterPay DB

Given merchant, images are in MyAfterPay DB

When Get images request is sent to MyAfterPay DB

Then client images are shown in the image tool

@Images

Scenario: Upload a new image to image tool

Given merchant is on image uploading page

When a user uploads a new image

Then the new image is showed in the image page

And saved to MyAfterPay DB

@Images

Scenario: Adjust brand images in the image tool

Given the merchant uploaded brand image is shown in the image tool

When the merchant adjusts the brand image by zooming, positioning

Then merchant can see adjusted brand image

@Images

Scenario: Save brand images to MyAfterPay DB

Given merchant has uploaded images to save

When merchant approves brand images

Then brand images are saved to MyAfterPay DB.

The positive and negative scenarios specify most of the flows and fallbacks the system should provide. Well-written scenarios mean less specifications for the developers. The actors in the sequence diagram give simplified overview of the actors in the process and insight to software architecture.

4.2 Architecture

In this section, the architecture of a system and technology stack are considered. The architecture is designed using Domain-Driven Design Layered Architecture pattern [23] and consists of four layers – user interface, application, domain and infrastructure [24].

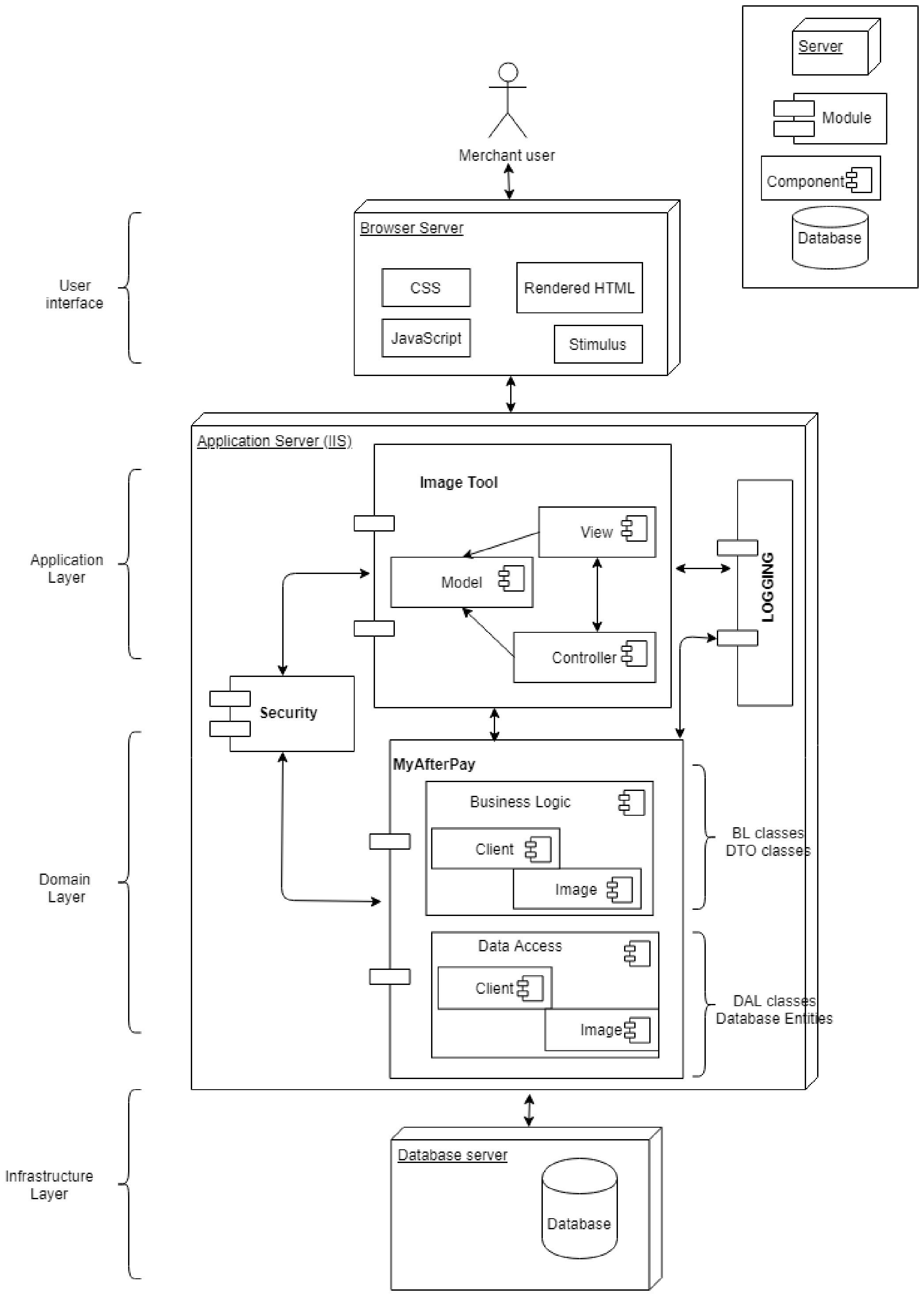


Figure 9. Image Tool Layered Architecture

As shown in Fig. 9, the four layers introduced before are presented. The user interface layer is what and how the merchant user sees the application from their web browser. The front-end frameworks consist of rendered HTML, JavaScript, Cascading Style Sheets (CSS) and Stimulus.

The application server and Internet Information Services (IIS) are facilitating web applications and help to run the software [25]. The application server for both MyAfterPay and image tool is ASP.NET. The application layer describes how the application runs, using the domain layer. Image tool application layer consists of model, view, and controller (MVC), which support one-another to bring user interface to the merchant user. MyAfterPay application and image tool are using the same security component, which allows authentication and authorization as well as guarding overall security of the software.

The shared component with domain layers is a logging component, which keeps track of the changes, warnings, errors, logs messages and events. The domain layer consists of business logic with Data Transfer Objects (DTOs) and the data access layer (DAL) with database entities. The domain layer is the heart of the software, handling most of the systems' complexity. The image tool uses Business Logic and Data Access components from the shared MyAfterPay module. The Business Logic component contains client and image services and appropriate DTOs while Data Access component handles the Data Repositories for client and image entities in the database.

Infrastructure layer supports all the layers above, and most importantly, contains a database that is accessed from DAL classes and database entities. The database server is a Microsoft SQL (Sequence Query Language) server.

4.3 Technology Stack

One of the main client requirements for the image tool is to be accessible from the Internet. The architecture presented in Fig. 9 is derived from the current MyAfterPay architecture and enables the reuse of essential existing components such as business logic services, data access layer, security and logging components.

The application is ASP.NET Core Model-View-Controller (MVC) web application [6]. The ASP.NET extends .NET and C# with frameworks, dependency injections, templates, security and patterns, such as the MVC [5]. In the MVC pattern, the model mediates controller and view with each other. From the view, user actions are performed by the controller with the model [6]. The view displays the model data which is received by the controller. The image tool business logic and data access components are part of MyAfterPay application components. Both of the applications are using Microsoft SQL server database.

It was decided by the designers to keep the front-end design minimalistic and simple, using quite a lot ASP.NET initial design. The rendered web application is using HTML (Hyper Text Markup Language), JavaScript for the web page behaviour, Cascading Style Sheets (CSS) for style. Also, Stimulus is added to JavaScript for making the website shine and simplify the code.

The Stimulus[1] is a JavaScript framework for HTML which helps to connect HTML and JavaScript automatically. Stimulus allows connecting and separating controllers, as well as attributes. It is a way how page events should be triggered by controller methods and how the attributes should be added to the controller's scope. As Stimulus conventions groups related code by name, it makes HTML readable and more understandable. [26]

The front-end and back-end, as well as database, need to work together and use the defined architecture.

4.4 Agile Development Approach

The Agile Manifesto is declaring that individuals, interactions and collaboration should be put before tools, working software and adaption to change should overcome plans and documentation. The measurement of the process is working software [27]. The Manifesto sets general principles, meaning every organization could adopt the rules and practices accordingly. Two most popular approaches are Scrum and Kanban.

The key principles for Kanban are to visualize the workflow, limit the work in process, control the flow and the team divides the work [28]. In the middle stands Kanban board.

[1] https://stimulusjs.org/

For MyAfterPay product development, pure Kanban has not been used, but the connected central system is developed in Kanban. It means the team is working on a few specific tasks and releases after the process is done. It may mean not all the features are released on the right time necessary for AfterPay.

Scrum software development process is for small teams with defined roles, and the delivery process is split into a short period with specific goals to achieve [29]. MyAfterPay has two weeks for each sprint. The sprint goals are set with around 60% of focus for developing new features and the rest 40% left for maintenance, bug fixing and refactoring. During sprint planning, the tasks are chosen and estimated, so the whole team is informed about the chosen features and maintenance goals for the sprint.

The definition of done for MyAfterPay development team is when the task or feature is released to production. From the sprint perspective, the task is finished when development is finished, and the task is successfully tested by quality assurance. After passing through development and quality assurance phases, the task requires business acceptance and approval.

The image tool was developed next to main product MyAfterPay customer portal, with the same resources. The team follows for AfterPay development SCRUM methodology. For the image tool was chosen a hybrid of Scrum and Kanban, it may be called as Scrumban [30]. The team approached the Scrumban in a way – once the product owner prioritized the task, then it was taken to development with the full development and testing cycle, keeping an eye on the process and by the end of the task deployed to release.

AfterPay front-end development and design are done by an external company located in Sweden, but the teams are working together. The front-end team is taking part in MyAfterPay Scrum daily stand-up meetings, and planning and refinement sessions, to align the work process within teams. Sometimes distributed teams may need extra refinements over time.

4.5 Development Process

AfterPay is an international product with the distributed team – business is located in Sweden, Norway and Germany, the developers and quality assurance is based in Tallinn, Estonia, the design and front-end team, are in Malmö, Sweden, the product owner is in

Stockholm, Sweden. Such decentralization would mean merchants may be divided into different countries, just like the team. Sometimes it increases the complexity of working processes as it requires time to create understanding about the requirements and share and get knowledge for everyone.

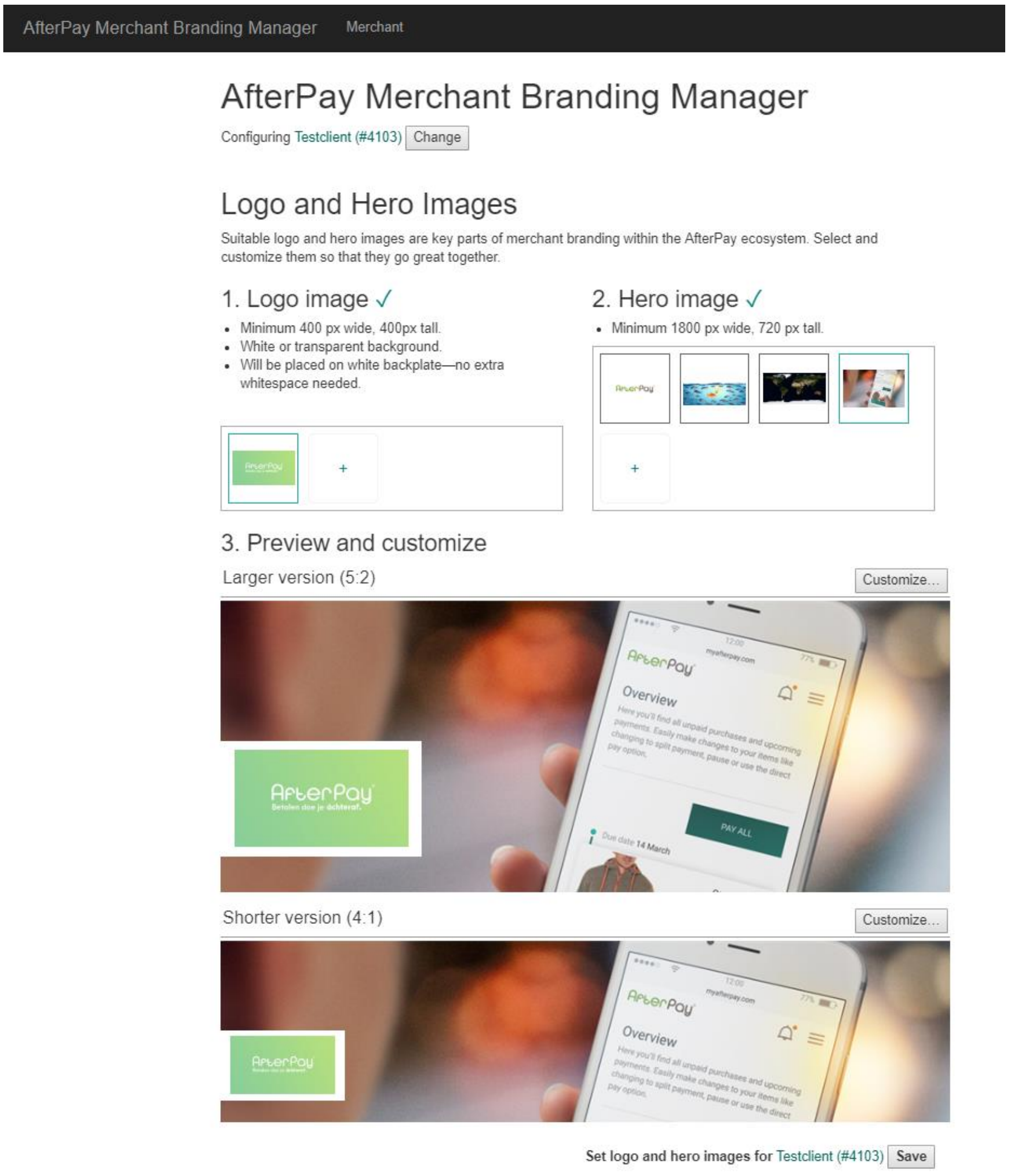
The business provides the requirements to the analysts who formulate the requirements to the front-end team in Sweden. Such communication results in knowledge sharing and common understanding alignment issues between the front-end and back-end teams.

The development process steps:

1. Architecture design
2. Front-end design
3. Front-end development
4. Front-end and back-end integration
5. Setting up the environments
6. Back-end development
    1. Getting images from database
    2. Uploading images to image tool
    3. Saving uploaded images to the database
    4. Adapting adjustments to the brand images
    5. Implementing change of logos
    6. Deleting uploaded images
    7. Adding log in

From the analysis until the realization of these steps, the timeframe lasted for four months, with the breaks when development was paused and resources used for MyAfterPay development. As the log in process with authentication and authorization is another big step, it was not prioritized by the product owner, and it was left out of the first development scope. It will be the next phase of image tool development.

The result of the development process is shown in Fig.10 that is now deployed to production and available for client representatives through an internal network because the login has not been realized for the application.


AfterPay Merchant Branding Manager Merchant

AfterPay Merchant Branding Manager

Configuring Testclient (#4103) Change

Logo and Hero Images

Suitable logo and hero images are key parts of merchant branding within the AfterPay ecosystem. Select and customize them so that they go great together.

1. Logo image ✓
- Minimum 400 px wide, 400px tall.
- White or transparent background.
- Will be placed on white backplate—no extra whitespace needed.

2. Hero image ✓
- Minimum 1800 px wide, 720 px tall.

3. Preview and customize

Larger version (5:2) Customize...

Shorter version (4:1) Customize...

Set logo and hero images for Testclient (#4103) Save

Figure 10. Merchant Branding Manager Image Tool Screenshot

In Fig. 10, is presented a screenshot of the image tool with the name AfterPay Merchant Branding Manager, which allows the user to choose a client, to see previous logos and images, as well as upload new brand images and logos. The tool provides an overview of the brand images in the same sizes as in MyAfterPay with the client logo. If the user clicks on the Customize button, it is possible to adjust the image by zooming or positioning in the size frame. The user must save the brand images in order to save the version to the database and show in MyAfterPay.

4.6 New Web Application Image Handling Process

BPMN is used to illustrate to business and developers how the image handling process is done using a new web application. In this context, the image could be a logo or a brand image. The final result is a brand image with the logo.

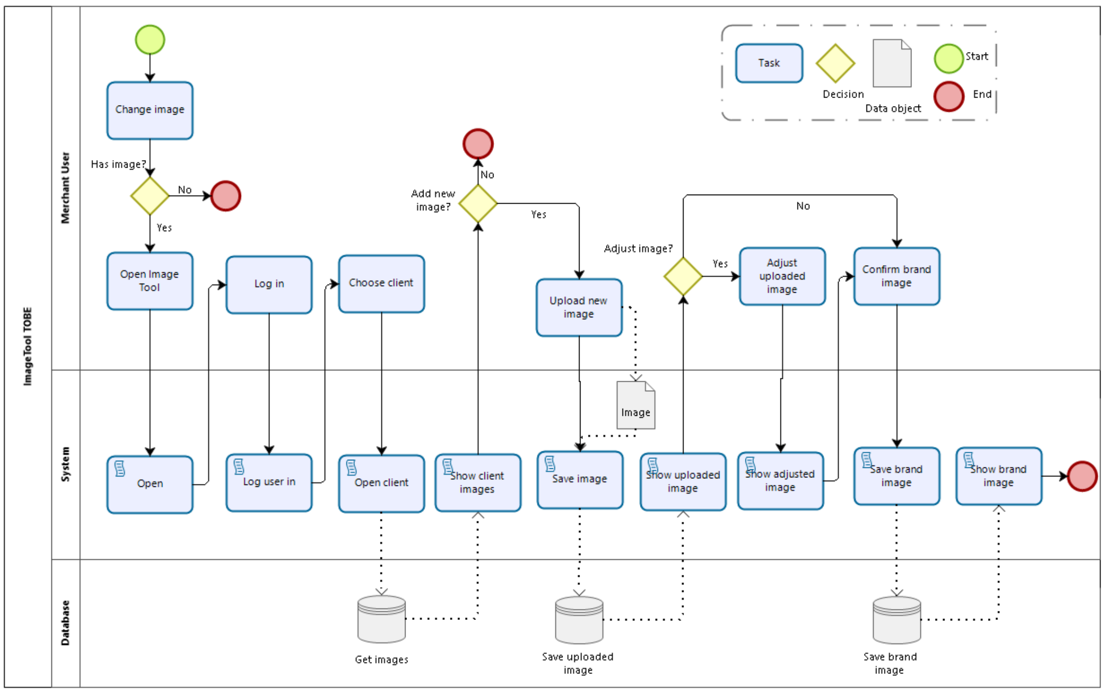


Figure 11. New Web Tool Image Handling the BPMN Process.

The diagram in Fig. 11 shows the image handling process for new image tool web application. The new image handling process has 3 actors – merchant user, system, which is the image tool, and database.

The overview how image handling process work in new web application:

1. The process starts with a merchant user need to change image.
    a. If the user does not have a new image, the process ends.
2. If a user has an image, the user can open the image tool web application.
3. After opening the web application, the merchant user can log in.
4. After authentication, the user must choose the client and system gets images to show from the database.
    a. If the client does not choose to add a new image, the process is ended.

5. When the client uploads a new image file. The system saves the image to database and shows the uploaded image in two sizes.
    a. If the user decides not to add a new image, the process ends.
6. The user can decide wheter to adjust the image or not. If the user decides to adjust, it can be done by cropping and zooming.
    a. If the user decides not to adjust the image, the images must be confirmed.
7. The user needs to confirm the brand image and system saves the brand image to the database and shows the image.
8. The adjustments must be confirmed, and then the brand image is saved by the system to database and process can be finished.

The procedure does not include unnecessary actors who do not have many purposes in the image handling process. During the process, the merchant user has more options to decide how the images should look and if the adjustments are needed. The changes and results can be seen immediately.

4.7 Clients' Representatives Feedback on Web Image Tool

Client testing has started for eleven clients. As the authorization method is not yet implemented the application is hosted in the internal server the accessibility for Merchant image tool is still limited. The image handling is done by two clients' representatives on AfterPay side. The client feedback is gathered from the two client representatives during interviews.

The clients are working together with their AfterPay representatives, so there are some trust and mutual understanding, and it is easier to give feedback to the representative. For one client it took some convincing to agree with being the test client, others allowed to be test subjects. Three of the volunteered merchants elaborated their image adjustment requirements, and the rest of the volunteers trusted their representatives. Two of the volunteered clients asked to make further adjustments.

The client representatives were glad for the possibility to handle the image loading themselves. The user interface was easy and understandable; the explanations for the image requirements were helpful for the images they had received.

One of the representatives noticed that when the uploaded image is exceeding the certain size of the images they are trying to upload, the process is stopped, but the user does not get any feedback. Such a bug came out early in the testing and resulted in a development ticket.

It was also noticed that when canceling the brand image adjustment process, nothing happens. However, when clicking on the image customize button again, the image will be reset to the initial position. The other representative was missing the possibility to adjust the logo image. The sent logo had an extra frame, which the tool did not let to adjust.

Both of the representatives agreed that image handling tool is good addition to AfterPay and MyAfterPay. The user interface is simple, and the merchant user should be able to handle the brand image and logo loading and adjusting process themselves. The representatives are capable of supporting clients with image tool usage.

5 Evaluation

The overview of the new image tool functionality is compared to the current desktop image application. As software development is an ongoing process where improvements can be done on different scales, the web image tool has input from AfterPay business and client representative testing, which could be improved already.

5.1 Comparison of the Current Desktop Image Tool and New Image Tool

Table 1 compares the desktop image tool and web image tool possibilities.

	Desktop image Tool	**Web image Tool**
Application type	Desktop application	Web application
Access rights	Developers	Merchant users Merchant representatives Developers Product owners
Image handling process participants	Merchant user Product owners Developers	Merchant user
Tool possibilities	Load image Load logo Create banners	Load image Load logo Adjust images Show different sizes of images Delete images
Future possibilities	Provide user access	Log in Available for other products Create banners Logo adjustments

Table 1. New and Old Image Tool Comparison

To illustrate and compare the possibilities of the old and the new application, Table 1 presents an overview. The image handling process for old tool starts from the merchant user to the product owner, who priorities the image handling to the developer. The new application allows the users to handle the images themselves, or by AfterPay client representative. The desktop image tool had access rights only for developers, compared to the new web application where access can be granted for merchant users, AfterPay merchant representatives, developers, and product owners, who can manage the images. The desktop application allows to upload a brand and logo images and create banners, which are not much used in MyAfterPay. The new image tool supports image adjustment, to show the brand images in the right sizes for MyAfterPay website, and delete image functionality.

The future possibilities for the desktop application would be to provide user access. The web application needs to have a login function. It would be possible to add logo adjustments and banner creation for the website.

5.2 Future Development Possibilities

The next step for image tool development is to add authorization and authentication. The access should be granted to specific people in the organization. It is important that the merchant representative has access only to certain merchant’s profiles. The goal is to let merchant users handle their images.

AfterPay product roadmap has defined adding banners to MyAfterPay image tool. Banners would be used to present campaigns, discounts, and offers on MyAfterPay. The banner would be an image with some text. Such a change would mean a new feature for adding banner images and country and language-specific text to banners. Adding banners would tighten customer and merchant relationship, it would be another method to bring the merchant web shop closer to the customers, with offers, advertisements, and news.

As AfterPay is an ecosystem with services provided to the clients, for example merchant portal and merchant product image management application, the AfterPay Merchant Branding Manager could be integrated with the other services, making the image tool accessible for the other tools in AfterPay system and creating more opportunities to show the brand images.

6 Summary

The thesis analysed AfterPay client image management for MyAfterPay and how it could be improved. The main image presentation purpose is to create customer assosciation with the client.

The image management process for the current image tool is limited and includes various actors making it a non-scalable application, increasing the time it takes to change the brand images. Also, it is not possible to make any adjustments to the brand images.

The client requirements for the image handling were to simplify the image handling process by giving access to the images handling process with the abiliy to upload and adjust the images themselves.

The new image tool gives AfterPay clients more freedom and possibilities on how to handle their images in order to stand out for the customers. The image handling process is now easier and more flexible; the user can upload and adjust the images by themselves whenever they want.

The development approach for MyAfterPay distributed team means different approach than the traditional agile development. The whole team took part of process to define the Behaviour Driven Development scenarios and from there the development process started.

In the scope of the thesis, the development was done and tested with the clients' representatives. As software development is ongoing process, the business is always requiring improvements and new features, for example adding banners and integrating the tool for other AfterPay services.

References

[1] AfterPay, "AfterPay Terminology," [Online]. Available: https://developer.afterpay.io/terminology. [Accessed 24. 02. 2019].

[2] Optimizely, "Hero Image," [Online]. Available: https://www.optimizely.com/optimization-glossary/hero-image/. [Accessed 10. 05. 2019].

[3] Business Dictionary, "Logo definition," [Online]. Available: http://www.businessdictionary.com/definition/logo.html. [Accessed 13. 05. 2019.].

[4] Object Management Group OMG, "Notation (BPMN) version 2.0," 2011. [Online]. Available: https://www.omg.org/spec/BPMN/2.0/. [Accessed 03. 05. 2019].

[5] Microsoft, "What is ASP.NET," 2015, 2015. [Online]. Available: https://dotnet.microsoft.com/learn/web/what-is-aspnet. [Accessed 30. 04. 2019].

[6] Microsoft, "Overview of ASP.NET Core MVC," 2018. [Online]. Available: https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-2.2. [Accessed 18. 02. 2019].

[7] AfterPay, "AfterPay Basics," [Online]. Available: https://developer.afterpay.io/basics/. [Accessed 03. 03. 2019].

[8] AfterPay, "Technical White Paper version 2.2," 2017. [Online]. Available: http://documents.afterpay.io/guidelines/Technical_White_Paper_ver2.2.pdf. [Accessed 24. 02. 2019.].

[9] AfterPay NL, "How does it work?," [Online]. Available: https://www.afterpay.nl/en/customers/how-does-it-work . [Accessed 29. 04. 2019.].

[10] H. Zheng and M. Wang, "Analysis on the relation between enterprise brand image and product image," 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5374921. [Accessed 14. 02. 2019].

[11] F. Chen, X. Yue, X. Yang and T. Ge, "Study on Classification of Personality-based Brand Archetype from the Perspective of Internet," 2014. [Online]. Available: https://ieeexplore.ieee.org/document/6895394/. [Accessed 11. 02. 2019].

[12] S. Shen, “Differences Between Designing a Web App and a Desktop App,” Muzli, 2019. [Online]. Available: https://medium.muz.li/differences-between-designing-a-web-app-and-a-desktop-app-e7d65a7545eb. [Accessed 22. 03. 2019].

[13] AfterPay NL, “Where can I pay with AfterPay?,” [Online]. Available: https://www.afterpay.nl/en/customers/where-can-i-pay-with-afterpay. [Accessed 22. 04. 2019.].

[14] A. Norta, M. Mahunnah, T. Tenso, K. Taveter and n. C. Narendra, “An Agent-Oriented Method for Designing Large Socio-technical Service-Ecosystems,” 2014. [Online]. Available: https://ieeexplore.ieee.org/document/6903272. [Accessed 07. 05. 2019].

[15] A. Van Lamsweerde, “Goal-Oriented Requirements Engineering: A Guided Tour,” 2001. [Online]. Available: https://ieeexplore.ieee.org/document/948567. [Accessed 07. 05. 2019].

[16] L. Becker, “90% of All Usability Testing is Useless,” 2004. [Online]. Available: https://immagic.com/eLibrary/ARCHIVES/GENERAL//ADTVPATH/A040616B.pdf. [Accessed 07. 05. 2019].

[17] J. Morris and B. Still, “The Blank-Page Technique: Reinvigorating Paper Prototyping in Usability Testing,” 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5467312. [Accessed 07. 05. 2019].

[18] I. Jacobson, I. Spence and K. Bittner, “Use Case 2.0: The guide to succeeding with use cases,” 2011. [Online]. Available: https://www.ivarjacobson.com/sites/default/files/field_iji_file/article/use-case_2_0_jan11.pdf. [Accessed 05. 05. 2019].

[19] K. Bittner, “Introduction To Writing Good Use Cases,” 2006. [Online]. Available: http://www-07.ibm.com/shared_downloads2/software/rsdc2006/ra_day_1/WritingGoodUseCases.pdf. [Accessed 05. 05. 2019].

[20] C. Solis and X. Wang, “A Study of the Characteristics of Behaviour Driven Development,” 2011. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6068372. [Accessed 14. 03. 2019].

[21] D. North, “Introducing BDD,” 2006. [Online]. Available: https://dannorth.net/introducing-bdd/. [Accessed 22. 03. 2019].

[22] Sparx Systems, “UML Tutorial - Sequence Diagram,” [Online]. Available: https://sparxsystems.com/resources/tutorials/uml2/sequence-diagram.html. [Accessed 22. 04. 2019].

[23] E. Evans, “Domain Language: Tacking Complexity in the Heart of Software,” 2004. [Online]. Available: http://domainlanguage.com/ddd/reference/. [Accessed 08. 05. 2019.].

[24] H. Graca, “Medium: Layered Architecture,” [Online]. Available: https://medium.com/@herberto.graca. [Accessed 07. 05. 2019].

[25] J. Ottinger, “TheServerSide: What is an App Server?,” 2008. [Online]. Available: https://www.theserverside.com/news/1363671/What-is-an-App-Server. [Accessed 10. 05. 2019.].

[26] Basecamp, “Stimulus Handbook,” [Online]. Available: https://stimulusjs.org/handbook/introduction. [Accessed 20. 02. 2019].

[27] “Manifesto for Agile Software Development,” 2001. [Online]. Available: http://www.agilemanifesto.org/. [Accessed 20. 03. 2019].

[28] M. Oivo, J. Markulla and M. O. Ahmad, “Kanban in software development: A systematic literature review,” 2013. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6619482. [Accessed 01. 05. 2019].

[29] K. Schwaber and J. Sutherland, “The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game,” 2017. [Online]. Available: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100. [Accessed 27. 04. 2019].

[30] S. Pahuja, “What is Scrumban,” [Online]. Available: https://www.agilealliance.org/what-is-scrumban/. [Accessed 01. 05. 2019].