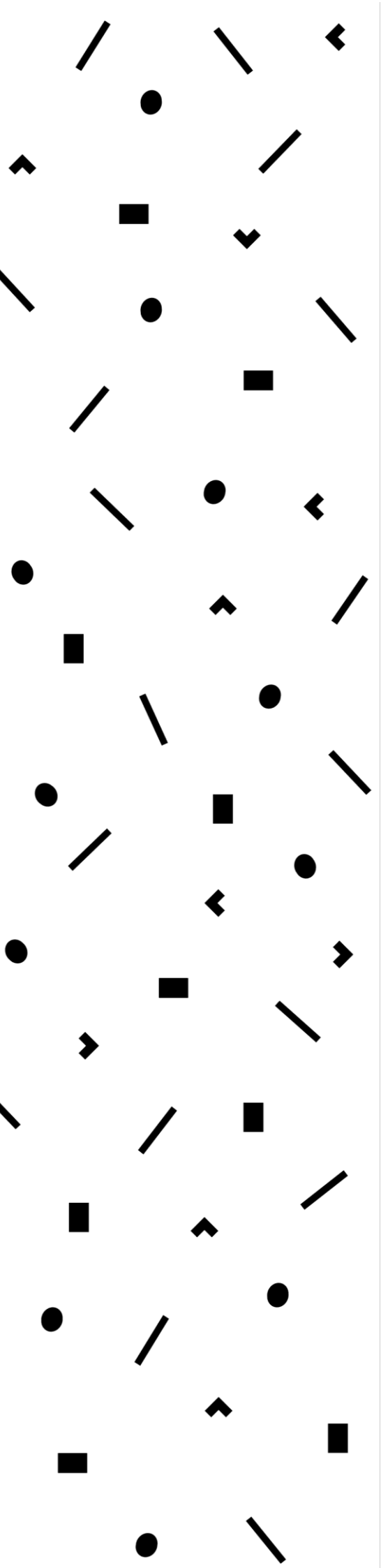


LAMK Lahden ammattikorkeakoulu
Lahti University of Applied Sciences

Developing a hybrid mobile application with Ionic

Case: Kunkku application and SuperApp Oy

LAHTI UNIVERSITY OF APPLIED SCIENCES
Degree programme in Business Information Technology
Bachelor Thesis
Spring 2018
Cong Tung Phan

Lahti University of Applied Sciences
Degree Programmed in Business Information Technology

PHAN, CONG TUNG:

Title: Developing a hybrid mobile application with Ionic

Case: Kunkku application and SuperApp Oy

Bachelor's Thesis in Business Information Technology

48 pages, 2 pages of appendices

Spring 2018

ABSTRACT

The primary goal of the thesis is to find out how Ionic could benefit the case company's projects in general and how the benefits differ based on seniority. Consequently, the thesis focuses on the Ionic framework and its benefits between different developers.

The theoretical part of this study presents Ionic, native and hybrid mobile application development and their pros and cons. The thesis also presents a case study of creating a cross-platform mobile application by using Ionic by two developers. The case study aims to provide data to analyze and conclude the benefit of using the Ionic framework. This study applied the design science research method and an inductive approach to answer the research question.

The results of the thesis are the answer to the research question, the data of the case study of two different developers with analyses. It suggests that by using the Ionic framework, the developer can save time, reduce costs and maximize the use of website programming languages such as HTML, CSS, JavaScript and so on.

Keywords: Ionic, hybrid mobile application development, Android, iOS.

LIST OF ABBREVIATIONS

API	Application Programming Interface
BYOD	Bring Your Own Device
CLI	Command Line Interface
CSS	Cascading Style Sheets
HTML	Hypertext Mark-up Language
Sass	Syntactically Awesome Style Sheets

TABLE OF CONTENT

1 INTRODUCTION

Mobile devices are expanding rapidly, and applications play a prominent role in it (Montag, Błaszkiewicz, Sariyska, Lachmann, Andone, Trendafilov, Eibes & Markowetz 2015). The massive number of applications nowadays need to be available on different operating systems. Developing multiplatform apps is demanding, and therefore some tools are created to support it (Ghatol & Patel 2012, 2-4). Ionic is one of these tools (Ravulavaru 2015, 22). This thesis demonstrates how using Ionic can benefit application development.

1.1 Background

The increasing popularity of smartphone is rooted in its number of advantages and conveniences (USAID 2015). Smartphones help their users to do many things. They can be used as alarm clocks, reminders, pocket calculators, clocks and media devices. Smartphones have such abilities thanks to the power of applications running on them (WebWise 2012). As shown below in Figure 1, in the first quarter of 2017, there was a total of 6,530,000 applications available in app stores. This number indicated that creating and developing applications is undoubtedly an important business.

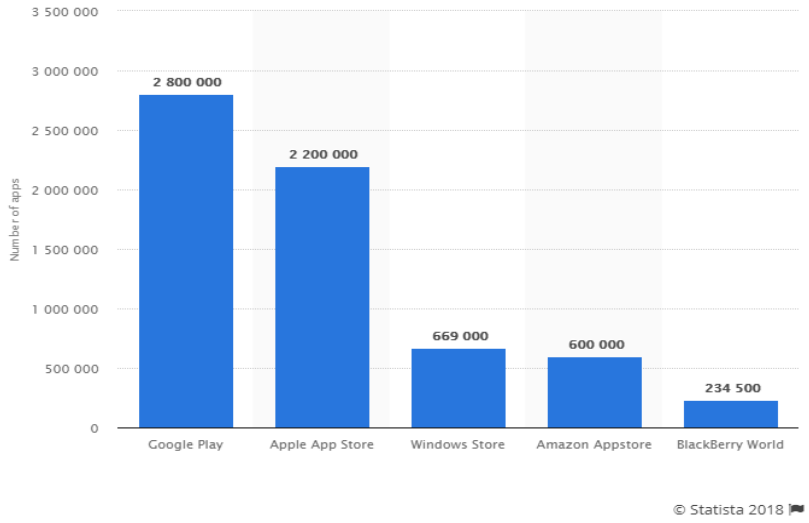


FIGURE 1. Number of apps available in leading app stores as of March 2017 (Statista 2018)

When creating apps for different smartphone operating systems, programming requires time because each system is based on a different programming language (Yusuf 2016, 2). This means that companies must to use different experts to develop applications for each platform.

1.2 Motivation

One efficient solution to overcome the above-mentioned problem is to use hybrid mobile application development. The hybrid mobile application development process supports developers to build applications for multiple mobile platforms at the same time (Panhale 2016, 1). By developing an application for various platforms, the business, as well as the developers, have advantages in time consumption (Pelletire 2013). The hybrid mobile application framework can benefit IT companies. It reduces the time needed for application development. This solution is contemporary because these kinds of businesses own many mobile application projects. If the traditional native application development method is applied, work doubles in every phase of a project. The hybrid mobile application framework saves developers from duplicated work as well as wasting money and human resource (Nikula 2018). In fact, the number of apps and

programmers choosing hybrid apps is increasing. According to a survey conducted by the Ionic survey, in 2015, 20% of programmers responded that they chose the native app, whereas in 2017 only 2.9% of the respondents said the same.

Ionic is the most popular hybrid mobile application framework in the world (IONIC 2018a). There are approximately 5 million programmers who use Ionic, and the number of applications created by this framework is 4 million (IONIC 2018b). The thesis examines how Ionic can brings benefit for the business.

2 RESEARCH DESIGN

2.1 Research Question

A research question is a tool to make the choice of methodology easier, produce a refined set of results, proficiency in the analysis, and identify the critical related literature (Wilson 2010, 44). Thus, a research question plays a significant role in research.

The thesis aims to answer the following research questions:

- Question 1: How the use of Ionic, a hybrid mobile application framework, benefits the company's projects in general?
- Question 2: How the benefits differ based on the level of seniority?

The thesis provides a theoretical background which describes the native and the hybrid mobile application development process, and Ionic. Furthermore, the thesis introduces a case study including the development of a cross-platform mobile application between two different developers by using Ionic. The case study provides data which the author can analyse to generate a theory on how the use of Ionic benefits the case company in general and how the benefit differs based on the level of seniority.

2.2 Research Approach

There are several types of research approaches: an abductive, a deductive and an inductive approach. According to Babbie (2013, 52), the deductive approach tests hypotheses, whereas the inductive approach begins with observation and proceeds towards generalizations (Saunders 2012, 61).

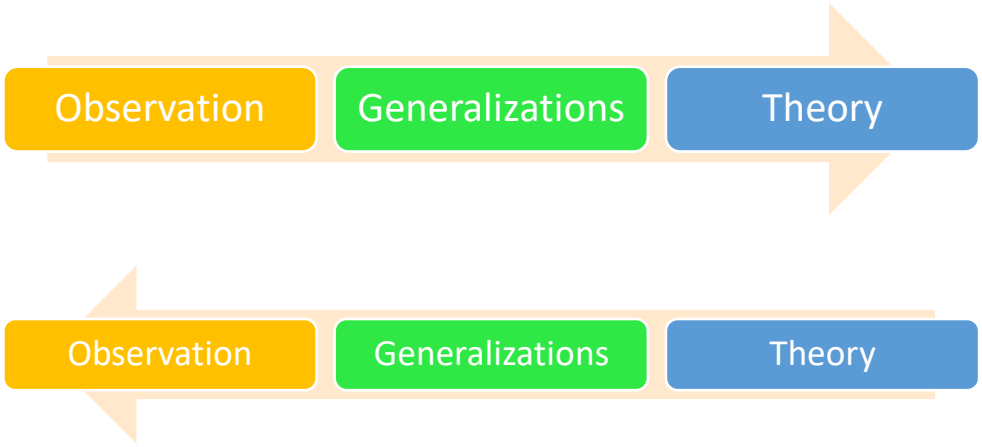


FIGURE 2. Process diagram of inductive (above) and deductive (below) research approach

This paper uses inductive research approach to generate a theory. To be more precise, creating a new theory and concept are the requirements of this thesis. The author gathers data based on the case study, analyses the data and generates a theory to answer the research questions.

2.3 Research Method

The research method of this thesis is design science. According to Hevner (2004, 1), the roots of the design-science model are engineering and sciences of artifact and it is a problem-solving model. The design science method can be used to create, test, develop, evaluate and improve artefacts. The design science method can be used to create, test, develop, evaluate and improve artefacts**.** An artefact is product such as a tool or a work of art. It could be a plan, product or process. By using the design science method, the researcher musts provide the problems and solution's suggestion; then they can choose to develop the artefact or test, evaluate the artefact to finalize conclusion and working solution for the output. The working solution can be, for example, an application, a method or a framework. Figure 3 below shows the reasoning process of design science research method.

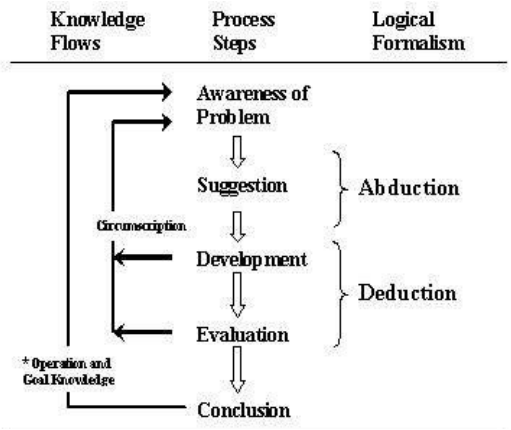


FIGURE 3. Reasoning in the Design Cycle (Takeda, et al. 1990)

Table 1 demonstrates seven guidelines of design-science research. Along with that, detailed descriptions of these principles are also presented.

TABLE 1. Design-Science Research Guidelines (Hevner 2004)

Guideline	Description
Guideline 1: Design as an Artifact	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
Guideline 2: Problem Relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3: Design Evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4: Research Contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
Guideline 5: Research Rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
Guideline 6: Design as a Search Process	The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
Guideline 7: Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

2.4 Data Collection and Analysis

There are five ways to evaluate the designed artifacts: observational, analytical, experimental, testing, descriptive. The table below shows the detail of each design evaluation methods.

TABLE 2. Design Evaluation Methods (Hevner 2004)

1. Observational	Case Study – Study artifact in depth in business environment
	Field Study – Monitor use of artifact in multiple projects
2. Analytical	Static Analysis – Examine structure of artifact for static qualities (e.g., complexity)
	Architecture Analysis – Study fit of artifact into technical IS architecture
	Optimization – Demonstrate inherent optimal properties of artifact or provide optimality bounds on artifact behavior
	Dynamic Analysis – Study artifact in use for dynamic qualities (e.g., performance)
3. Experimental	Controlled Experiment – Study artifact in controlled environment for qualities (e.g., usability)
	Simulation – Execute artifact with artificial data
4. Testing	Functional (Black Box) Testing – Execute artifact interfaces to discover failures and identify defects
	Structural (White Box) Testing – Perform coverage testing of some metric (e.g., execution paths) in the artifact implementation
5. Descriptive	Informed Argument – Use information from the knowledge base (e.g., relevant research) to build a convincing argument for the artifact's utility
	Scenarios – Construct detailed scenarios around the artifact to demonstrate its utility

The selection of design evaluation methods depends on the research's subject. More specifically, the subject of this thesis is to find out how using Ionic benefits in time consumption. The aim is to compare an Ionic development project called Kunkku between the author (Developer 1) and higher-level developer. The other developer is more experienced in developing hybrid mobile applications with Ionic.

During the developing process of Kunkku application, the author records the work process and keeps a research diary by using Trello and a time recording system called SuperProject. Nguyen (Developer 2), the other developer mentioned above, uses Trello and a development diary to record time consumption. After the development process, the author and Nguyen will evaluate the difficulty of each task.

Finally, the thesis includes the author's time estimation for each task for next time development. Using the Ionic framework for the first time could not maximize its' benefits for the user. Therefore, collecting the time estimation for the same task if the developer has the second time to do it

would show how a developer can utilize Ionic's advantages after having experiences with it.

2.5 Thesis Structure

The thesis has four main sections divided into eight chapters. The diagram below clarifies the layout of the thesis. As can be seen from the design, the four main sections are an introduction, theory, practice, and conclusion. The following summarizes the main content of each section:

Introduction: The introductory section introduces the contents of the thesis and explains key terms and acronyms. This section also presents the research design of the thesis.

Theory: This section indicates the theory of application programming in general as well as demonstrating topics relating to native, hybrid mobile application, and Ionic. More specifically, the primary purpose is to identify these concepts and their pros and cons. Time consumption in mobile application development is also discussed in this section using data collected from previous reports.

Practice: The first part of this section is case description which briefly introduces author's case study. Based on this specific case and theoretical background presented above, working plan is then created in which the project progress is divided into suitable phases. The following chapters show the collection of data derived from case study and analysis of those data.

Conclusion: This section outlines the strengths of using Ionic to program mobile applications. Finally, the disadvantages of using Ionic have also summarized here along with solutions to overcome these difficulties.

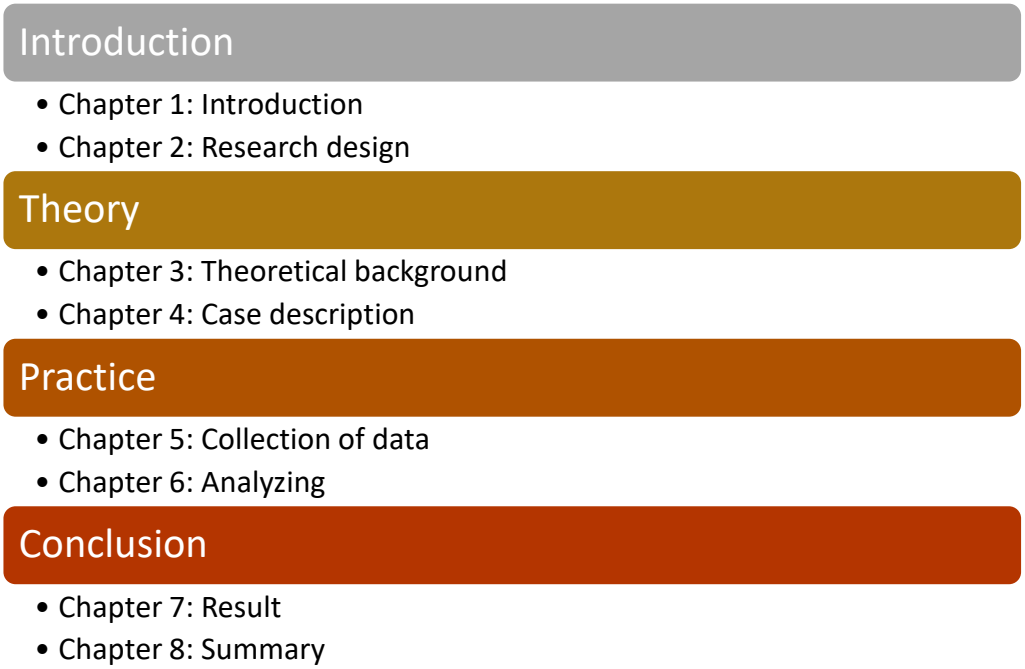


FIGURE 4. Thesis structure

3 THEORETICAL BACKGROUND

3.1 Native Mobile Application Development

This chapter provides information about the native mobile application development. More specifically, there are definition of native mobile application development, relationship between it and BYOD, and its pros and cons.

3.1.1 Definition of Native Mobile Application Development

Techopedia (2018) identifies native mobile application as “a smartphone application that is coded in a specific programming language”. To be specific, developers need to use particular programming language for particular operating systems such as Objective-C or Swift for iOS, Java for Android and C# or XAML for Windows Universal. (Griffith 2017, 3.) Therefore, a native application is only available on its dominant platform.

Native applications can be either installed on the mobile device by default or downloaded on the app store. Data associated with these apps is accessed remotely or stored directly on the mobile devices.

3.1.2 Native Application With Bring Your Own Device Trend

In 2009, Malcolm Harkins, who is Intel’s chief security and privacy officer proposed a Bring Your Own Device (BYOD) strategy. Precisely, it is the recognition of issue which employees increasingly use their mobile device in the office. (Field 2011.)

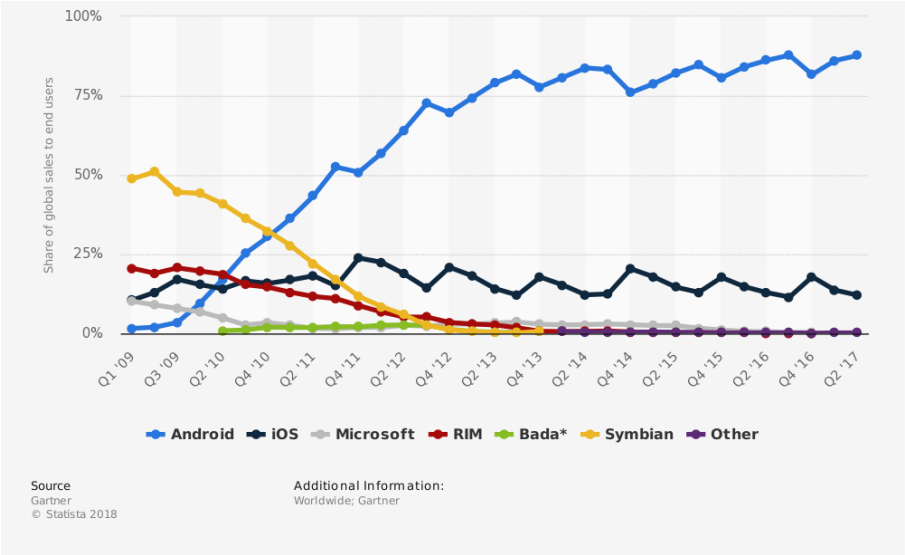


FIGURE 5. Global mobile OS market share in sales to end users from 1st quarter 2009 to 2nd quarter 2017 (Statista 2018).

As can be seen from Figure 5, there are distinct kinds of mobile operating systems. Therefore, developing an application for multiple operating systems, especially for Android and iOS, are required to respond to the BYOD trend.

3.1.3 Pros and Cons of Native Mobile Application Development

There are several reasons why the native application is a traditional standard in the mobile industry. First, this type of application offers a full device integration which makes it easier to access device's built-in features such as GPS, camera, microphone and so on. Also, since the application is designed for one specific framework, developers can fully utilize all the native APIs and features without any add-on solution (Griffith 2017, 3). Another advantage of the native application is its brilliant performance which can enhance user experience.

However, choosing native mobile application may not be an adequate choice for many circumstances because of its disadvantages. The main drawback of this application type is rooted in programming language

issues (Griffith 2017, 4). Native languages are practically difficult to learn which means companies need developers with high expertise. Moreover, since the native mobile app is platform-specific, none of the code can be reused which requires a separate version of the code base for the different operating system. These difficulties result in high development cost and much bigger time consumption than developing cross-platform applications. According to Dabit (2016), developer advocate at AWS mobile, the cost for native mobile application development has been surging recently making it difficult for small companies without substantial funding to build a native app.

3.2 Hybrid Mobile Application Development

This section indicates the definition of hybrid mobile application development. Its advantages and disadvantages compared to native mobile application is also specified in this chapter.

3.2.1 Definition of Hybrid Mobile Application

A mobile application which can be developed for multiple platforms by using single codebase is a hybrid mobile application. However, to harness the native APIs for each platform, developing hybrid mobile application requires some specialized code. (Khanna, Yusuf & Phan 2017, 4.)

As with any other native mobile application, a smartphone’s user can install the hybrid mobile application on their device. Furthermore, hybrid mobile application development can also access smartphone’s hardware, for instance, GPS, flash, camera, mic, speaker and so on. (Khanna, Yusuf & Phan 2017, 4.)

Khanna, Yusuf, and Phan (2017, 5) define two types of hybrid mobile applications: a WebView-based Hybrid App and a Cross-compiled Hybrid App. A native mobile application which runs the web application by using a WebView (or a chromeless browser window that’s typically configured to run full screen) is a WebView-based hybrid mobile application (Griffith

2017, 4). Thanks to the ability to make a responsive Web from HTML5, CSS3, the WebView-based hybrid mobile application can fit for multiple device's screen sizes. There are numerous of JavaScript gesture detection library to handle with touch interaction such as Touch events (by Mozilla), ZingTouch and so on. These extensions help WebView-based hybrid mobile application increases the User Experience and becomes an adequate choice for developing solutions for multiple devices. Cordova, Ionic, Mobile Angular UI are some typical types of WebView-based hybrid mobile application development.

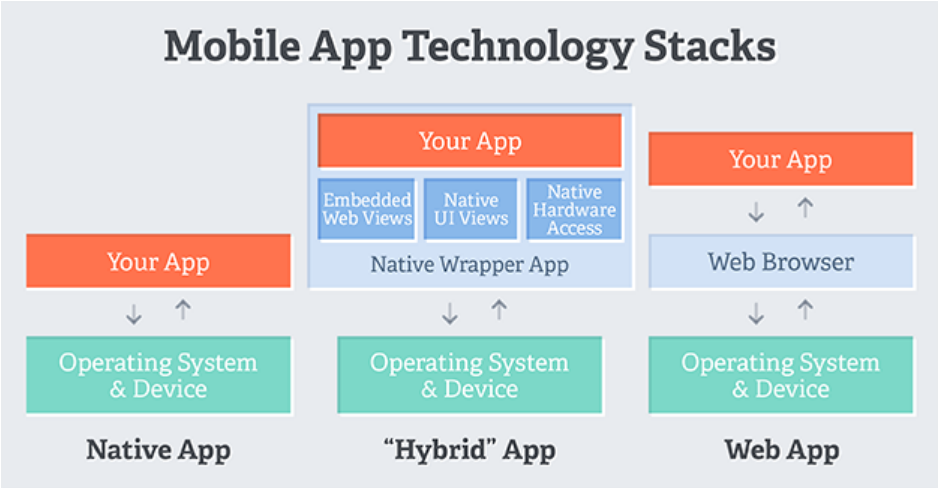


FIGURE 6. Mobile App Technology Stacks (Globetrotter 2018)

Another kind of Hybrid Apps, a Cross-compiled Application, allows developers to convert a specific single language into native language component at compile time (or during the runtime). It creates links between native components and their custom constructs in the programming language. Xamarin with C#, Corona with C, and Kony with JavaScript are three examples of Cross-compiled Hybrid Apps. (Khanna, Yusuf & Phan 2017, 4.)

3.2.2 Pros and Cons of Hybrid Mobile Application Development

The hybrid mobile application has several advantages. Firstly, to deploy multiple platforms, native mobile application development requires a

corresponding number of programming language whereas Hybrid only use one. Thus, the time consumption of developing a cross-platform application with a hybrid mobile application development is much less than a native mobile application. Furthermore, hybrid mobile application development often offers the programmer a familiar and accessible programming language to use. In another word, web developer does not have to learn new languages, the development is also more natural to maintain the code, it still provides full access to device's features, and it is cost-effective. (Griffith 2017, 4.)

On the other hand, according to Panhale (2016, 20), the hybrid mobile application has limited device-specific feature APIs compared to native mobile application. Furthermore, these applications are the web application. Therefore, the performance and capabilities cannot smooth as native mobile application. The poor performing happens more frequently in older devices. Next, APIs and plug-ins are the links between WebView and native feature, that is another dependency for user's project. There is no guarantee for this communication could available all the time. (Griffith 2017, 4.)

3.3 Ionic

This chapter introduces one of the popular types of hybrid mobile application development: Ionic framework along with its functionalities.

3.3.1 Introducing to Ionic Framework

Ionic framework is a popular choice for native mobile application development (Griffith 2017, 1; Khanna, Yusuf & Phan 2017, 1). By using Ionic framework, developers can create a native-looking mobile application with web technologies' languages such as HTML5, CSS3, JavaScript. Ionic's users can build and upload their application to the marketplace without any cost because Ionic is an open source.

For building the application from web content, Ionic chooses Apache Cordova to build on top. It gives Ionic abilities to simplify mobile application development as well as improving User Interface and user experiences. For instance, the ListView, Optimized Touch gesture, Side Menus, Tabs and so on are ready for developers to implement in few steps. (Khanna, Yusuf & Phan 2017, 14.) Especially, Apache Cordova allows the web application access native's capabilities and become a native application. (Griffith 2017, 2.)

The Ionic framework also built with AngularJS. According to Khanna, Yusuf & Phan (2017, 14.), this widely-used and well-tested framework allows Ionic's user to create a Single Page Application, and easier to organize.

Each of mobile platform has their requirements on UI components. To help the user save time with that, Ionic provides a ready-made UI for mobile components, and it automatically applied based on the platform build. The developer can use SASS to generate CSS in Ionic. The image below shows an example of Ionic's template CSS components.

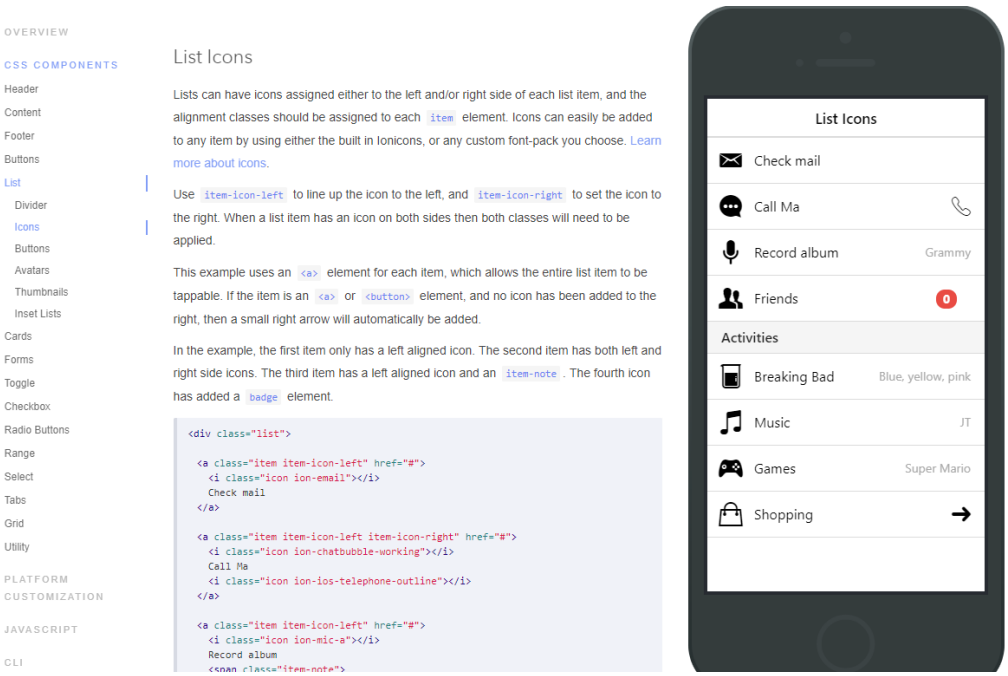


IMAGE 1. Ionic's CSS components documentation (Ionic 2018)

Ionic also provides JavaScript features for developers, which is essential for building an Ionic application. Some of these JavaScript features are Modal, Slide box, Action sheet, Sidebar, tabs, lists and so on. As mentioned above, again, all the JavaScript are built with AngularJS.

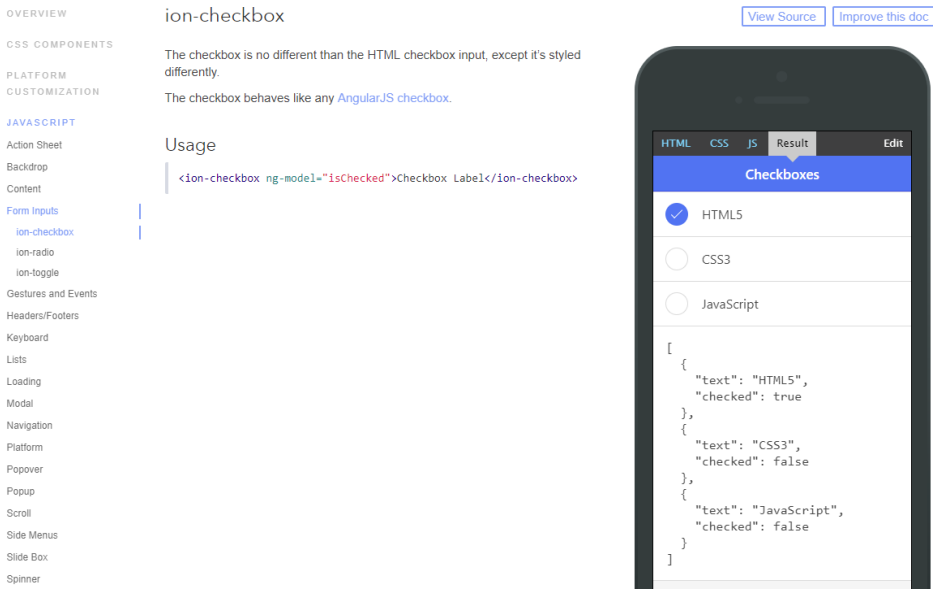


IMAGE 2. Ionic's JavaScript components documentation (Ionic 2018)

The Ionic Command Line Interface (CLI) is another essential part of Ionic. This feature allows Ionic's user to user Terminal (in MacOS) or Command Line (in Windows) to use the Ionic commands. More specific, developers can create an Ionic project, development, and testing, Ionic generator, build, run sass and so on via Ionic CLI feature. (Khanna, Yusuf & Phan 2017, 146-147.)

TABLE 3. Examples of Ionic command line interface

`$ npm install -g ionic@latest`	CLI command to install latest Ionic Version
`$ ionic start myNewProject`	Create a "myNewProject" project by using ionic start

`$ cd ./myNewProject` `$ ionic serve`	Go to “myNewProject” folder and preview app in the browser. Changes made to code will automatically refresh.
`$ ionic install -g cordova`	Install Cordova to Ionic’s Project
`$ ionic cordova --help`	Getting help and commands from ionic Cordova
`$ ionic cordova build ios`	Build current project to native iOS application
`$ ionic cordova build android`	Build current project to the native Android application
`$ ionic cordova build window`	Build current project to native Windows application

Ionic provides Ionic Creator which is a drag and drop tool for the user with limited coding skills. The users can design themselves their application and test it in the browser. The result of Ionic Creator is a folder with codes, and it can build an application. This is reduced the time and workloads between designers and developers.

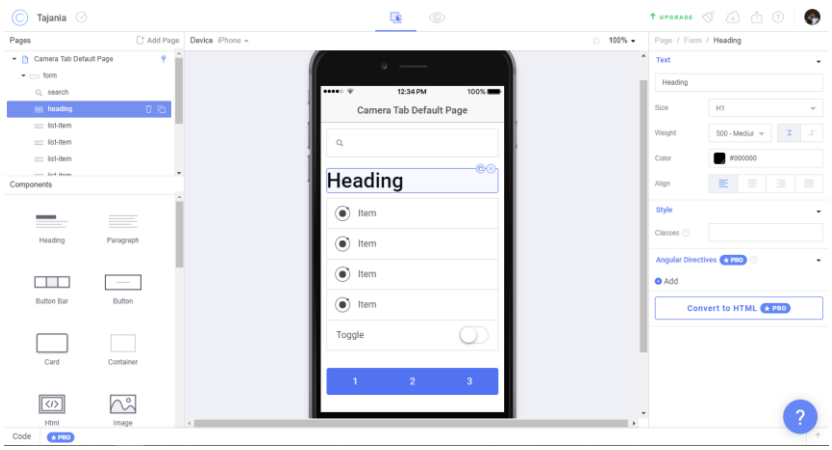


IMAGE 3. Ionic Creator with Drag-and-Drop (Ionic 2018)

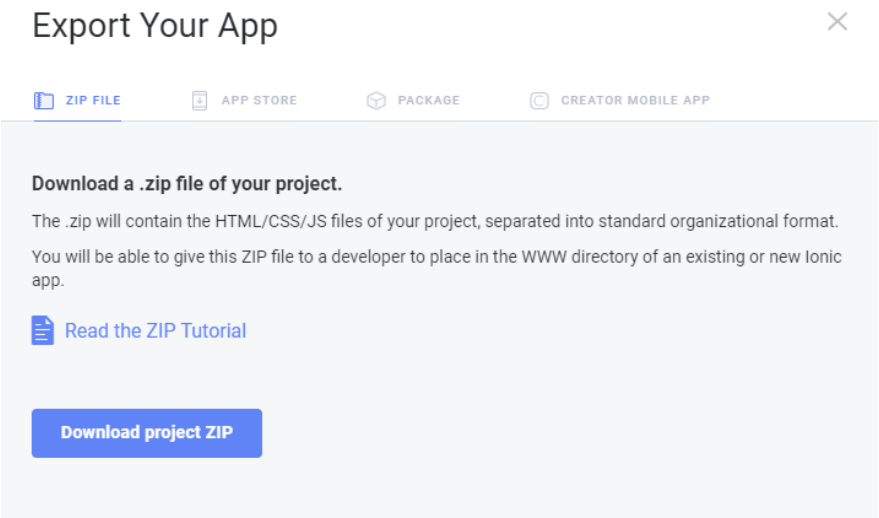


IMAGE 4. Ionic Creator Export (Ionic 2018)

3.3.2 The Use of Ionic

According to Khanna, Yusuf & Phan (2017, 142), the Ionic framework is a suitable developing tool for a developer who has basic of knowledge on web technologies. Furthermore, the simple, great and mighty application is the goals of most people. Besides that, others want to maximize the use of their preexisting skill set to do new things, such as build a cross-platform mobile application. To be more precise, Griffith (2017, 7) includes the understanding of HTML, CSS, JavaScript, and Angular to the required skills to developing an application with Ionic.

Functions are essential for the application. The number of function and complexity of each application are different. For example, some applications get texts from the website to show in the application, some others take a photo with filters, animations. Furthermore, the time to developing the function for the application depends on the developer's experience and skill. Therefore, this phase is not able to estimate.

IMAGE 5. “Cool Reader” – An example of a simple application with only text and image. (Google Play 2018)

IMAGE 6. “Instagram” – An example of a compound application with streaming video, display comments and likes in real-time (Google Play 2018).

Developers must build the application in simulator mode in the computer or to physical devices to test the code. However, it is the requirement for the development of the native application. With Ionic, the developer can view the app by using the web browser. More specific, by typing “ionic serve” in CLI, it launches HTTP server, the user can open their application

in the desktop browser. As mentioned above in Table 3, changes made to code will automatically refresh the web browser immediately.

To build the application to physical devices, CLI command “ionic cordova build android” and “ionic cordova build ios” export a native application file project which can open in Android Studio (with Android) and Xcode (with iOS). After exporting and opening these files, the developer can build the application in the same way with the native mobile application.

By using Ionic, the developer can use installs plugins from Ionic, Cordova or PhoneGap. On March 18, 2018, there are 226 plugins available to download or purchase for Ionic projects in Ionic’s homepage (IONIC 2018). Besides, 1534 is the total number of plugins of PhoneGap and Cordova which user can also install to Ionic’s project. By using the plugins, the developer can make the application more functional and visualize. Developers can download the plugins then move them to Ionic’s project folder, or they can use CLI to install the plugin. After installing the plugins, the developer implements JavaScript code to enable the plugins for the application. Some plugins are functional, which gives the user multiple options during the implementation. Thus, this step requires the developer to understand the plugins by reading the documentation and examples.

3.4 Time Consumption of Mobile Application Development

Oberg (2015) defined the chart which shows the difference of the development speed between Xamarin, Ionic and Native Android in his thesis. As can be seen from the figure below, the speed of developing an application for a single operating system with Ionic is approximately 15% faster than with Native Android. Furthermore, the Ionic framework produces the cross-platform at the end of the development, whereas the Native Android only provides the application for Android platform. Based on this theory, by using Ionic to develop a cross-platform mobile application for iOS and Android, the developer will consume less than a half amount of time (approximately 44%) needed for the native Android and iOS mobile application development.

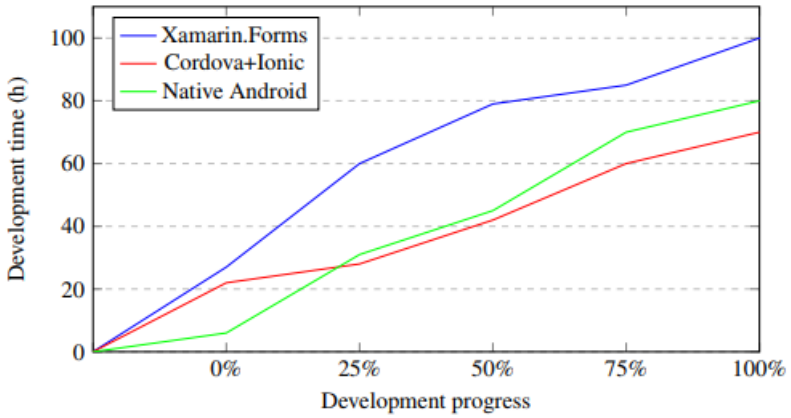


FIGURE 7. Development speed of one example application (Oberg 2015)

4 CASE DESCRIPTION

This section introduces the backgrounds of the hybrid mobile application which developed by the author. The framework which used this artifact is Ionic Framework.

4.1 Client Company

Kunkku Oy is a hostel chain which provides accommodations, restaurants, events, saunas and hot tubs for the customer. They have two branches which are Syvärin Kunkku in Tahko, Kanavan Kunkku in Vääksy. They already have a website and expect to launch a cross-platform mobile application.

SuperApp Oy is an IT company which is located in Helsinki and Lahti. SuperApp Oy has focused on website and mobile application development but also develops other technologies (e.g. IoT, VR, AR, AI). Kunkku Oy is one of SuperApp's customers. After having successfully developed Kunkku's website, SuperApp Oy is responsible for developing a mobile application for Kunkku. The author has been working in this company since the end of 2017 and developing the Kunkku application was one of his tasks.

4.2 The Goals of the Kunkku Application

First, Kunkku should be an application for multiple operating systems: Android and iOS (cross-platform). By accessing the application, the customer can find information about three restaurants: Ravintola Tahko, Ravintola Vääksy, and Majoitus Tahko/Pärnu. The figure below shows the structure of the Kunkku application.

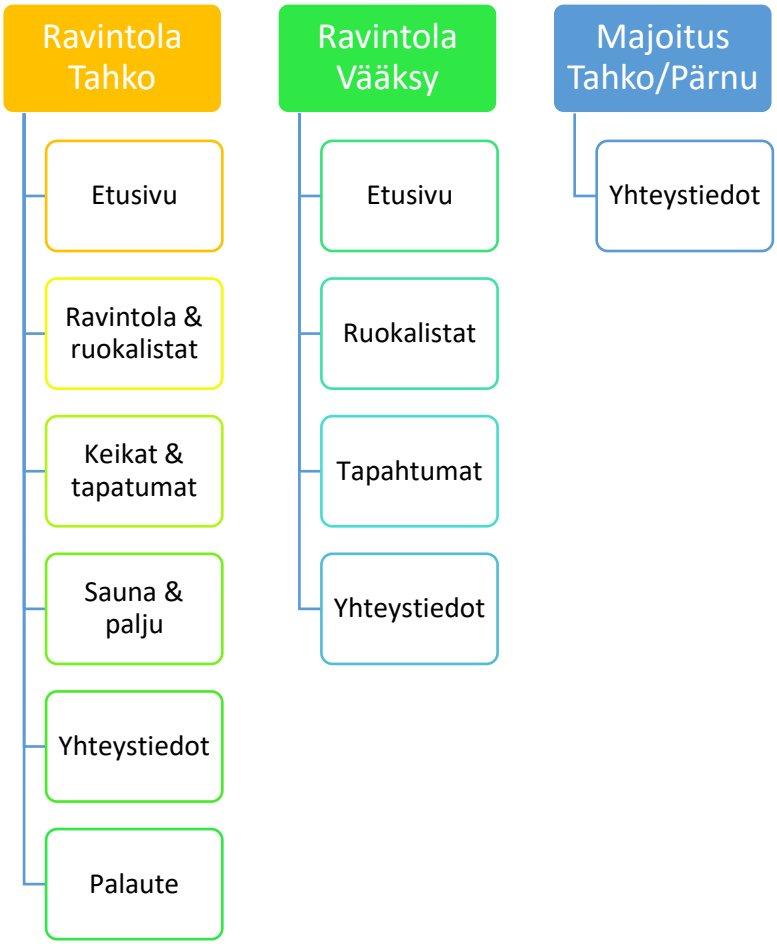


FIGURE 8. Kunkku Application's Structure

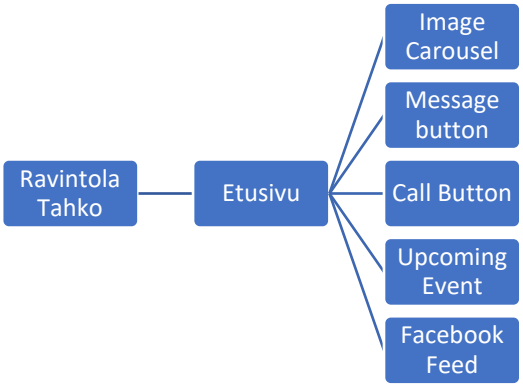


FIGURE 9. Elements in Ravintola Tahko's Etusivu

In Ravintola Tahko section, there should be six different pages which are: Etusivu, Ravintola & ruokalistat, Keikat & tapatumat, Sauna & palju,

Yhteystiedot, and Palaute. The image carousel, address, and two buttons to initiate the mobile call and send a message to the restaurant function are the requirement in the top of the Etusivu. Two other requirements of this section are Upcoming Event and Kunkku's Facebook feed. The Upcoming Event must auto-update the content which can get from Kunkku's Ravintola Tahko website.

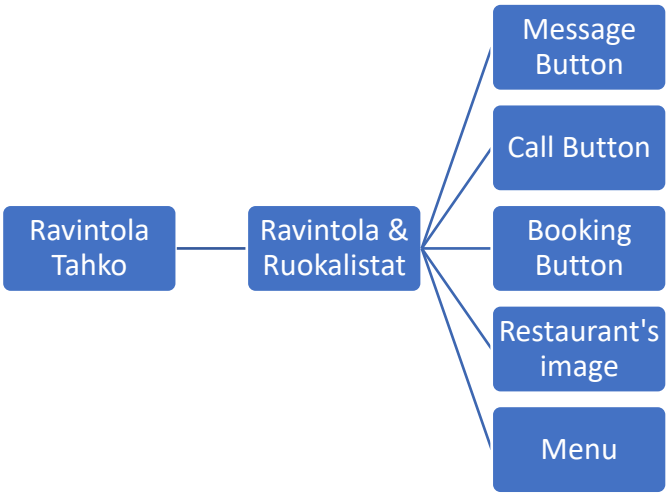


FIGURE 10. Elements in Ravintola Tahko's Ravintola & Ruokalistat

The Ravintola & ruokalistat section also requires the same buttons. However, this page also needs another button to access the Reservation page. Moreover, when the customer reaches this page, they can see the restaurant's picture and restaurant's menus.

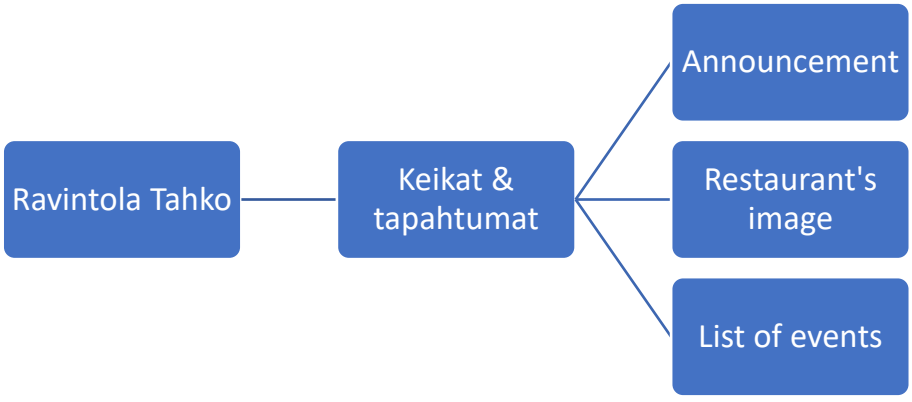


FIGURE 11. Elements in Ravintola Tahko's Keikat & tapahtumat

The announcement is expected to appear on the top Keikat & tapahtumat page. Below this part is the restaurant's image and list of events. The list of events must auto-update the content from the restaurant's website like in the Etusivu.

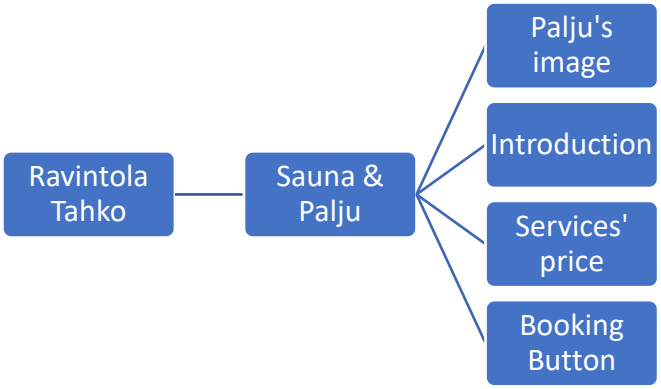


FIGURE 12. Elements in Ravintola Tahko's Sauna & Palju

Palju's image, the introduction of Sauna and Palju services with prices, and booking button is the requirement of Sauna & Palju page.

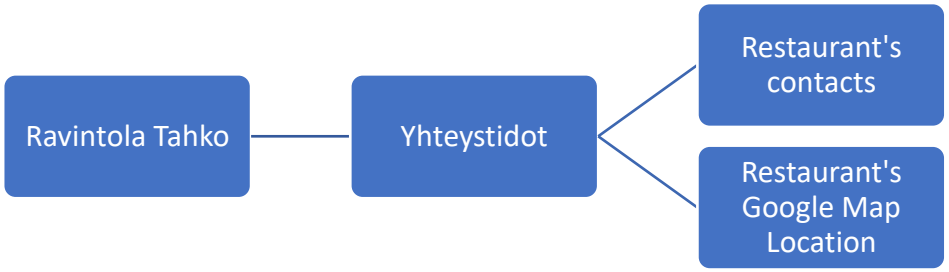


FIGURE 13. Elements in Ravintola Tahko's Yhteystiedot

By accessing the Yhteystiedot page, the user can see the restaurant's contacts and the location of the restaurant in Google Map.

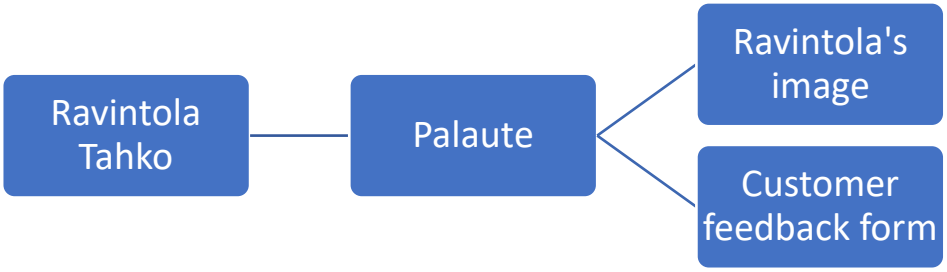


FIGURE 14.Elements in Ravintola Tahko's Palaute

The last section of Ravintola in Tahko is Palaute, where the customer can send feedback to the restaurant by a form.

The Ravintola, Vääksy only has four sections which are Etusivu, Ruokalistat, Tapahtumat, and Yhteystiedot. In general, the structure of these pages is similar to these pages in Ravintola Tahko. The only difference between them is they getting the information from another website.

The Majoitus, Tahko/Pärnu is different from the others. It has only one section which is Yhteysiedot and its structure also like two other Yhteysiedot pages.

There should be a page that shows a booking form when the user presses a button from on the restaurant page or sauna/palju page. The page collects the user's basic information such as name, phone number, and email, and also booking information (number of guest, reservation date, and choosing service sauna or palju).

4.3 Case Study Plan

The case study of this thesis is applying the Ionic framework to develop the Kunkku cross-platform application. The development of Kunkku executed Ionic Framework. There are five sprints in the project which are design, develop User Interface with code, create functions, apply plugins, build to devices and test.

Project name: Kunkku Mobile Application

Tasks	Person	Start date	Ready	Status	Spring 1 (28/11/2017-29/11/2017)	Spring 1 (28/11/2017-29/11/2017)	Spring 2 (29/11/2017-07/12/2017)	Spring 2 (29/11/2017-07/12/2017)	Spring 3 (27/12/2017-29/12/2017)	Spring 3 (27/12/2017-29/12/2017)	Spring 4 (10/01/2018-12/01/2018)	Spring 4 (10/01/2018-12/01/2018)	Spring 4 (20/01/2018-22/01/2018)	Spring 4 (20/01/2018-22/01/2018)
Project Planning and Preparation					Start	End								
Making the project plan	Tung Phan			Done	x	x								
Making the project schedule	Tung Phan			Done	x	x								
Setting up NodeJS, Ionic	Tung Phan			Done	x	x								
Installing Android Studio, Xcode	Tung Phan			Done	x	x								
Installing Sublime Text	Tung Phan			Done	x	x								
Design Mobile App with Creator				Done										
Design Splash Screen	Tung Phan			Done	x	x								
Design Ravintola Tahko Section	Tung Phan			Done	x	x								
Design Ravintola Vaaksy Section	Tung Phan			Done	x	x								
Design Ravintola Majoitus Tahko/Parnu Secti	Tung Phan			Done	x	x								
Project Execution														
Export file from Creator and checking code	Tung Phan			Done			x	x						
Create Booking Form + Feedback	Tung Phan			Done			x	x						
Handling Form and Feedback with PHP	Tung Phan			Done			x	x						
Impliment WordPress Rest API	Tung Phan			Done			x	x						
Adding Facebook Feed	Tung Phan			Done					x	x				
Adding Google Map	Tung Phan			Done					x	x				
Push Notification	Tung Phan			Done							x	x		
Project Finalization														
Testing the Application with iOS	Tung Phan			Done									x	x
Testing the Application with Android	Tung Phan			Done									x	x

FIGURE 15. Development phases of Kunkku Mobile Application

5 COLLECTION OF DATA

5.1 Data Collecting Process

During the development process of Kunkku application, the Developer 1 collected the time consumption's data of developing hybrid mobile application with the Ionic framework. As mentioned in chapter 2.4, the Developer 1 collected data by using Trello and SuperProject, SuperApps' time-management system. Trello is a website provides a collaboration tool which helps user's project. The user can track a project's status and who is working on what. (Trello 2018.) Trello shows the date, time and description of each task during the development. SuperProject not only provides the user a stopwatch, but it also records the task's name and description.

The other developer uses Trello and stopwatch with a development diary, which is similar to the Developer 1 's method. Developer 2 does not use SuperProject because this is not a SuperApp project.

Every time start working the project, developers use SuperProject or stopwatch with development diary and Trello to record task's name and time.

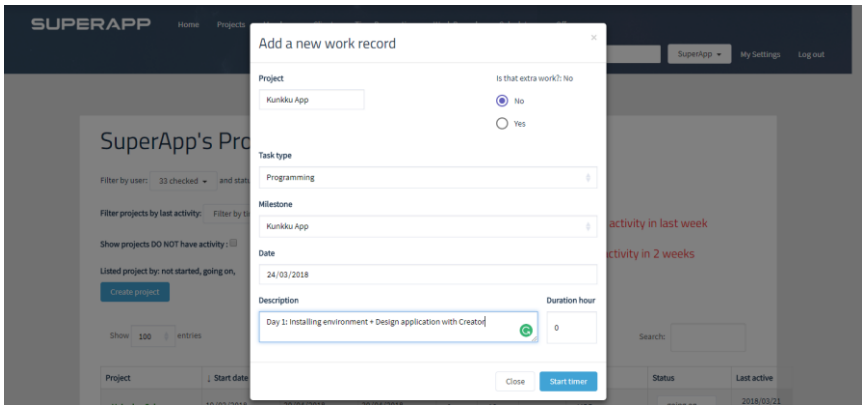
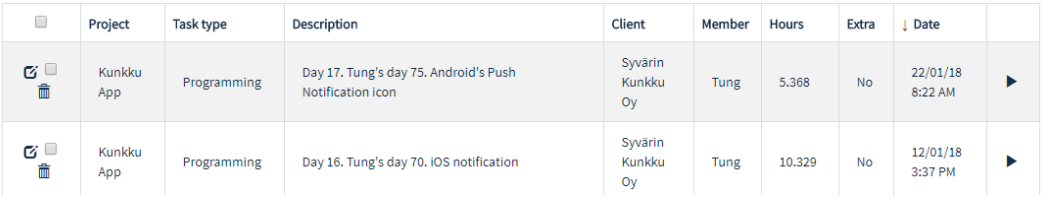


IMAGE 7. Example of adding a new record by using SuperProject

Figure 16 below shows how the Developer 1 can provide project's information to SuperProject. By clicking the "start timer" button, the system begins to count the time. The user uses "stop button" when they want to stop counting the time of the project. Then, the table of time record appears in "Work Records" tab as the figure below.



	Project	Task type	Description	Client	Member	Hours	Extra	↓ Date	
	Kunkku App	Programming	Day 17. Tung's day 75. Android's Push Notification icon	Syvärin Kunkku Oy	Tung	5.368	No	22/01/18 8:22 AM	▶
	Kunkku App	Programming	Day 16. Tung's day 70. iOS notification	Syvärin Kunkku Oy	Tung	10.329	No	12/01/18 3:37 PM	▶

FIGURE 16. SuperProject working report table

On the other hand, Developer 2 uses the online stopwatch with export commentary on the following website: http://stopwatch.online-timers.com/online-stopwatch.

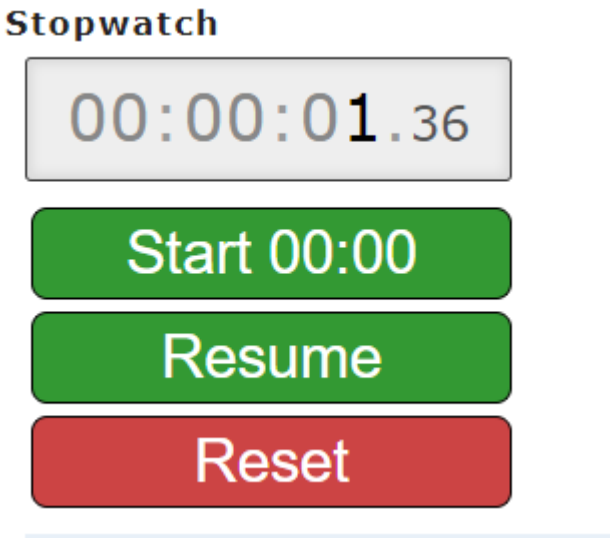
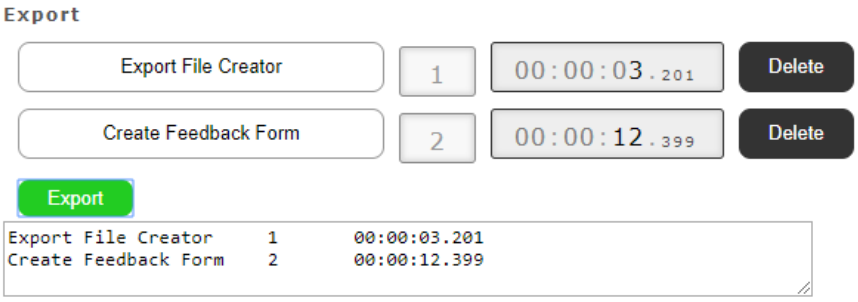


FIGURE 17. Online stopwatch UI on http://stopwatch.online-timers.com/online-stopwatch.



FIU 18 Export with commentaries UI in http://stopwatch.online-timers.com/online-stopwatch.

After each time Developer 2 finishes using this stopwatch, he exports the stopwatch's info and puts all the information in the development diary. Next, the Developer 1 and Developer 2 both create a Trello card for every new task.

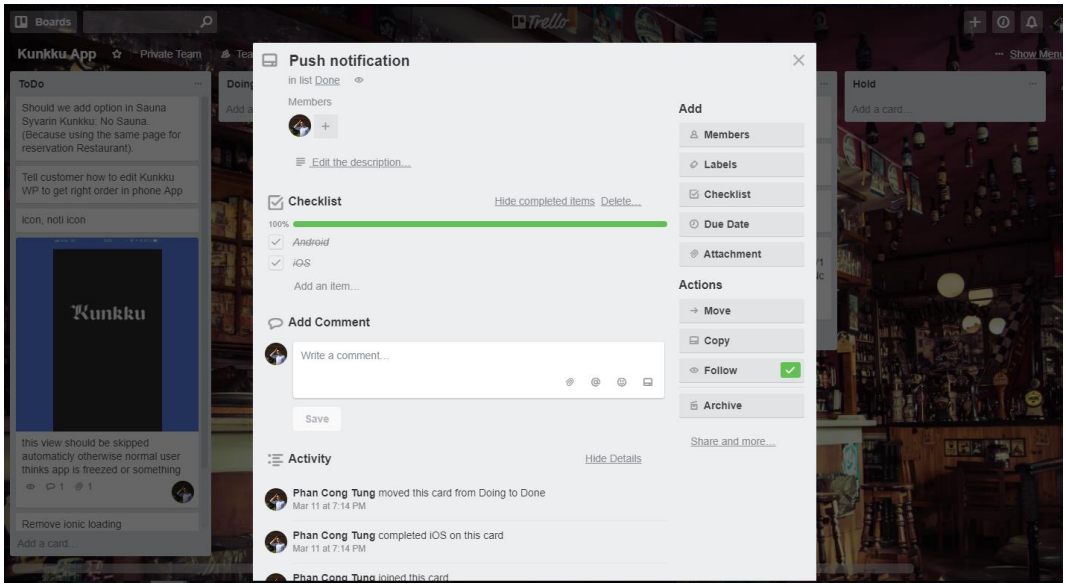


IMAGE 8. An example of Trello card.

Although the developing phases have different characteristics, it still uses the same method of data collection.

5.2 Collected Data Description

This chapter of the thesis mentions all the data which is collected during the development of Kunkku by two different developers.

5.2.1 Data On Developing Time of Two Developers

After finishing developing the Kunkku application, the Developer 1 gathers all the data from SuperProject, Trello and Developer 2's development diary. Then, these data are presented in the tables below which show the developing phases with the duration of developing Kunkku by both developers. There are 11 steps regarding four phases which mentioned in the previous chapter: develop User Interface with code, create functions, apply plugins, build to devices and test. Because both developers must use the same layout, design of the application, therefore, Developer 2 used Developer 1's design and the data added 8 hours to both developer's time consumption.

TABLE 4. Kunkku developing tasks and time by two developers.

No.	Tasks	Duration (Developer 1)	Duration (Developer 2)
1	Export file from Creator and create the Kunkku application	30 mins	20 mins
2	Create Feedback Form	30 mins	10 mins
3	Create Booking Form	6.5 hours	2.5 hours
4	PHP handle Feedback Form	2 hours	1 hour
5	PHP handle Booking Form	4 hours	2.5 hours
6	WordPress Rest API	18 hours	8 hours
7	Adding Facebook Feed	10 hours	5.5 hours
8	Adding Google Maps	7 hours	3 hours
9	Push Notification	20 hours	7.5 hours
10	Testing Android and iOS devices	10 hours	10 hours
11	Fixing Bug	22 hours	3 hours

The table above shows the working time of the Developer 1 in developing the Kunkku application project. The total time for the Developer 1 to successfully develop the application is 109.3 hours.

The total amount of time that Developer 2 used for developing the Kunkku application is 43.5 hours.

5.2.2 Data On Tasks' Difficulty of Two Developers

This section of the thesis provides two table of data which show the evaluation of the difficulty level of each task between the Developer 1 and Developer 2. With these tables, the thesis has more information to find out how the benefit differs based on the level of seniority.

TABLE 5. Evaluation of the difficulty level of each task on both Developer's point of view

No.	Tasks	Difficult (Developer 1)	Difficult (Developer 2)
1	Export file from Creator and create the Kunkku application	1	1
2	Create Feedback Form	1	1
3	Create Booking Form	2	2
4	PHP handle Feedback Form	1	1
5	PHP handle Booking Form	1	2
6	WordPress Rest API	1	4
7	Adding Facebook Feed	2	3
8	Adding Google Maps	1	1
9	Push Notification	3	5

10	Testing Android and iOS devices	2	2
11	Fixing Bug	2	5

After the developing process, Developer 1 and Developer 2 evaluate the difficulty of each task according to the following scale:

1. Very easy, the task should be quickly solvable.
2. Simple task but not immediately obvious.
3. Getting a bit hard, require more time and focus on doing the task.
4. A tough challenge and probably might need to collaborate with other people to solve.
5. Very difficult and time-consuming. There could be multiple answers, but only a few of them work and take much time.

5.2.3 Data On Time Estimation For Next Time Development of Developer 1

The last section of collected data description delivers the table with Developer 1's time estimation for each task for the second time of development. As mentioned in the previous chapter, this data would show how a developer can utilize Ionic's advantages after having experiences with it.

TABLE 6. Developer 1's time estimation for each task for next time development

No.	Tasks	Estimating time to do next time
1	Export file from Creator and create the Kunkku application	20 mins

2	Create Feedback Form	20 mins
3	Create Booking Form	4 hours
4	PHP handle Feedback Form	1 hours
5	PHP handle Booking Form	3 hours
6	WordPress Rest API	10 hours
7	Adding Facebook Feed	6 hours
8	Adding Google Maps	3 hours
9	Push Notification	10 hours

6 ANALYSIS

This section of the thesis analyzes and compares the data which is in the previous chapter.

6.1 Comparing the Developing Time of Different Developers

As mentioned in the previous chapter, the time consumption of developing the Kunkku application of the Developer 1 are 109.3 hours and for Developer 2 is 51.5 hours (plus eight hours because the Developer 1 provided Creator file for Developer 2).

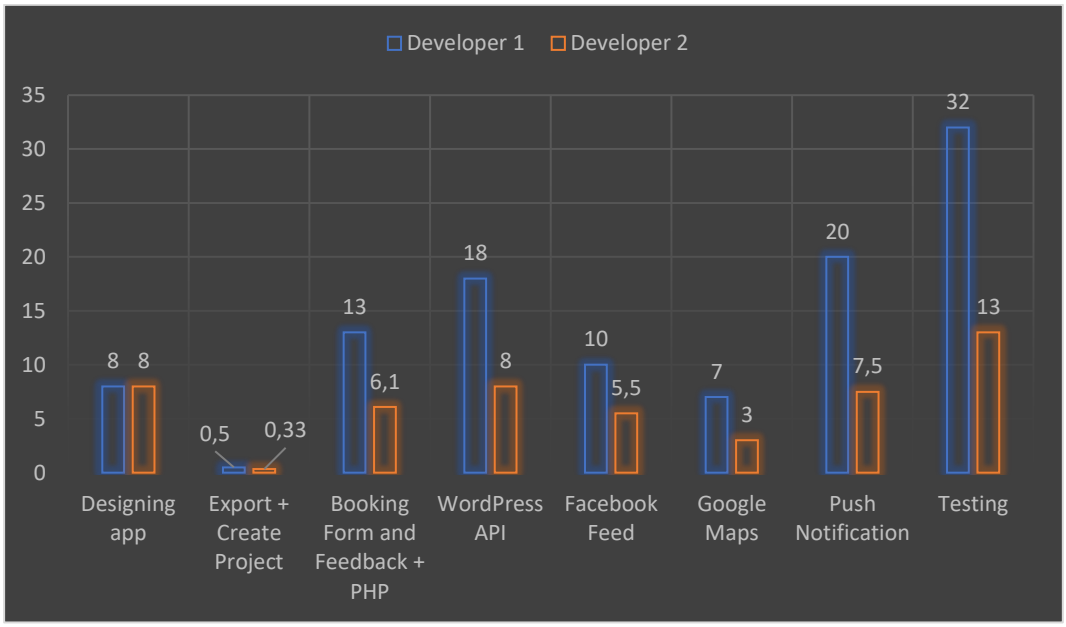


FIGURE 19. The Kunkku application's time consumption between two developers in details

The figure above shows the difference in time consumption of developing the Kunkku application between the Developer 1 and his colleague. In general, the Developer 1 spent double the time of almost all the tasks comparing with Developer 2.

The first column content is the time to design the application with Ionic Creator because both developers used the same layout, so Developer 1 added eight hours for both developers' time consumption. The next two

columns are the time to export Creator and create Ionic Project. The Developer 1 experienced that the time to do this task mostly spend for Ionic CLI, which requires developer understand the CLI's commands and functions. Developer 2 has done three Ionic's projects before Kunkku. Thus, the time consumption of Developer 2 is less than the Developer 1 (ten minutes).

Starting the programming phase, the developers need to work with the booking form and feedback as well as handling with PHP file on the server. The necessities of this step combine Ionic elements in HTML and PHP skills from the developer. The Ionic elements is a part of design the application with code phases. Although this was the new experiences for the Developer 1, however, it did not take much time. The 6.9 hours different between two developers is the consequence of the difference between developers' PHP skill. In the next columns which mention about WordPress API, the dissimilar in time consumption once again does not come from the use of Ionic Framework. The reason is that this task did not require any pieces of knowledge of Ionic from developers. The Developer 1 spent 18 hours, more than 10 hours in contrast to Developer 2 to handle with WordPress API and JavaScript.

On the other hand, the Facebook Feed in Kunkku mobile application did not take much time for both developers to implement. However, the diversity of mobile's screen size made the Facebook Feed not responsive. Thus, 4.5 hours which the Developer 1 took more than Developer 2 was mostly finding the solution for the Facebook Feed available on any phones' screen. The task of implementing Google Maps is similar to the WordPress API, which only requires the website technologies' skill from developers. So, with the advantages in coding experience, Developer 2 only took three hours to develop, four hours faster in contrast to the Developer 1.

The last two tasks took most time from both developers as can be seen from the figure. The Developer 1 took 20 hours to implement Push Notification for Android and iOS operating systems successfully. The part

which takes most of the of this task was iOS configuration. To enable the Push Notification, the developer needs different kind of system, hardware certificate from Apple. Furthermore, the parsePushPlugin, which enables devices to receive notification is provided by PhoneGap. Adding this plugin to the Ionic's project causes few errors for the Developer 1 during the build process. The reason is there are conflicts between the version of PhoneGap, Ionic, Android Studio Bundle and Xcode. Thus, the Developer 1 spent much time to find the solution from the internet. This part is more straightforward for Developer 2 because he has done twice with the notification in the past. Finally, testing is the last task of the developing process. To make sure the application is runnable for all devices, both developer tested the application by building the application to real different devices and after finishing every single function.

In conclusion, the reason of Developer 2 was faster is Developer 2 had done three complexes Ionic mobile application before, whereas Kunkku was the first Ionic project of the Developer 1. The Developer 1 spent more time to find tutorial and solution to do tasks because the Developer 1 only had three months experiences during this time. On the other hand, Developer 2 had three years experiences in web and mobile developer, so he knew the fastest way to do the task.

6.2 Comparing the Difficulty Between Different Developers

When the projects have been accomplished, two developers made a report regarding the difficulty of each Kunkku's task. This report accesses the hard level by the scale from 1 to 6.

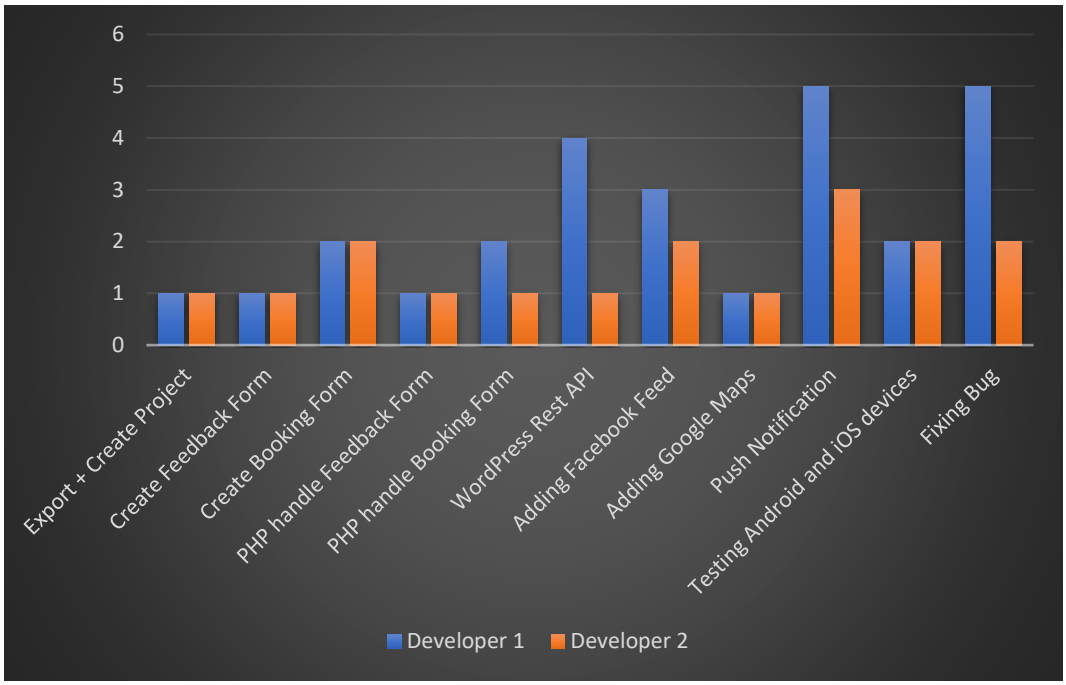


FIGURE 20. Evaluate the difficulty of Kunkku's tasks between two developers in details

The data of the figure above shows that there are six tasks that both developers marked with the same level of hardness. These tasks are export and create an Ionic project, create feedback and form, create a PHP file to handle the feedback form, adding Google Maps, and Testing Android and iOS devices. There are four tasks which marked one point, which is the lowest level of difficulty evaluation are. The Developer 1 and Developer 2 have experiences as a web developer. Thus, they evaluate this tasks at one point. The last equal result is testing task which is simple however not immediately obvious (two points).

In terms of different evaluation, the Developer 1 accessed five tasks harder than Developer 2. The testing task did not require a high level of the developer; however, the developer must be careful with every content, function, User Interface and User Experience of the application. The other two points of both developers are the creating the booking form's task. This was the first time for both developers to implement that, but it was not a big deal. Developer 2 marked the tasks of adding Facebook Feed as two points whereas the Developer 1 chose three points for that. Because

Developer 2 has more experiences in JavaScript and PHP in server side, so he marked the tasks of creating a PHP file to handle booking form and implementing WordPress Rest API at one point. With only three months of experiences, the Developer 1 got many troubles during these step, thus, the Developer 1 set the point of PHP tasks, and WordPress are two and four respectively.

The most challenging tasks for the Developer 1 based on the figure is implementing push notification and fixing bugs. Both tasks require the developer to understand and study many documentations. By having previous knowledge, Developer 2 just set these tasks as three (for the push notification) and two for the fixing bugs.

6.3 Comparing the Differences Between the First Time and Estimated Second Time Development

In general, the estimated time of Developer 1‘s second time development for the same tasks is less than the time consumption in the first time.

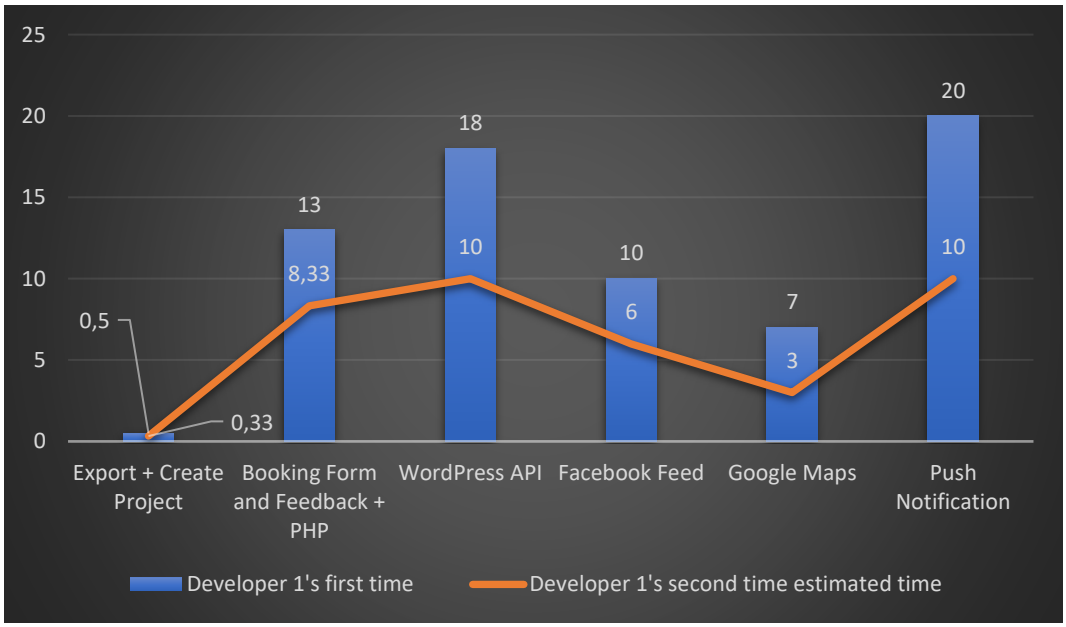


FIGURE 21. The Kunkku application’s time consumption between the first time and estimated second time development.

As can be seen from the graph, even the time required to export and create an Ionic project with CLI can reduce by 30 minutes to 20 minutes. The total of saving time is up to almost 31 hours in total. The task which Developer 1 can save the most time is the push notification (ten hours). After implementing for the first time, the Developer 1 got knowledge of push notification configuration, and certificates. Thus, the Developer 1 will not have to read and find the documentation during the implementation. The first time doing with the Kunkku project also gave the Developer 1 many experiences on handling with WordPress API and JavaScript. Therefore, the Developer 1 could save up to eight hours for the next time developing the WordPress API task. All the other tasks, in general, save four hours for the second time of develop (only the booking form and feedback with PHP file handle could take 4.67 hours).

In conclusion, the more developer uses develop an application with the Ionic framework, the more they can get use to it and maximize its benefit. This data also suggests that the more user develops a mobile application with Ionic, the more time they can save during the development. Thus, it is promising for the Developer 1 to develop the Kunkku application with the same amount of time with Developer 2 in the next two times.

7 RESULTS

This chapter answers the research questions presented in chapter 2. The questions were as follows:

- Question 1: How the use of Ionic, a hybrid mobile application framework, benefits the company's projects in general?
- Question 2: How the benefits differ based on the level of seniority?

7.1 Answer the Question 1

After the research process, it can be concluded that Ionic provides the company with some advantages in mobile application development. These benefits include saving time-consumption, human resources, and reducing the cost.

First, adopting the Ionic framework not only allows the company to build a cross-platform mobile application but also stimulates the development process with its' available built-in plugins and templates. These conveniences result in time-efficiency. Furthermore, instead of using separate languages for different platforms, with the use of the Ionic framework, the developer can utilize web technologies' programming languages to create an application for multiple platforms. This, in other words, means that the company does not need various developers with different expertise which reduces the cost of the employment and competent training.

7.2 Answer the Question 2

Based on the data collection and analyzed, it can be concluded that the developer with higher experiences can exploit the prevalence of the Ionic better than the junior developer.

The collection of time records demonstrates that for the same task and workload, the experienced coder spent approximately a half of the time consumed by the coder who used the Ionic framework for the first time.

This is because the difficulty level varies based on how well does the developer familiar with Ionic. However, thoroughly applying the Ionic framework is not challenging progress because the estimated time consumption for the same project (after having developed once) diminishes significantly.

8 SUMMARY

In conclusion, the thesis has resolved the research question about the benefits of Ionic to business through theoretical research and analysis of case study's result. The key findings include the data of different developers in the same Ionic project. The limitations, reliability and validity of the study as well as suggestions for further study will be discussed in this chapter to complete the research's result.

8.1 Limitations

The thesis has two main limitations. The first limitation is the lack of a cross-platform mobile application with native languages development. For example, in the case study of the project, if the author had developed the Kunkku application with Java in Android Studio for Android operating system and Objective C in Xcode for the iOS operating system, the thesis would have collected more useful information to compare in the data analysis. With that, the thesis could have pointed out the benefits of using the Ionic framework in time consumption in contrast with Native language in practice. The other limitation of the thesis is the restrict in the number of application in the case study. By providing the 2nd case study with the Ionic framework, the data of the thesis could be more accurate and comparable.

8.2 Reliability and Validity

Establishing hypothesis which accepted the scientific truth is the prerequisite. The example for this is the use of stopwatch if researcher performs a time-critical experiment. (Shuttleworth 2008.) In fact, the data of both developers are collected by using the stopwatch from SuperProject and stopwatch website application.

The reliability refers to the repeatability of the research and finding (PSC 2018). Furthermore, according to Shuttleworth (2008), the researchers must find the same experiment under the same condition. During the development of the Kunkku application, both developers used the same

methods, development's phases, developing environment, operating system to develop the Kunkku application. As a web developer, both programmers gave numerous numbers of the same result in the assessment of task's difficulty. Furthermore, the data in chapter 7.3 also suggests that the time for developers to develop the app if they are having the same experiences is promising to be equal.

Chapter 2.2 of the thesis shows the information and requirements of the design sciences research method. Besides that, the author provided the structure and step to encompasses all the steps from the method. After generating the suitable plan, the whole processes of research followed it step by step. Therefore, the result of this study is the conclusion of applying design science method, and it is valid research.

8.3 Suggestions for Further Study

The Ionic framework is one of the most popular hybrid mobile application development tools as was mentioned in the introduction. However, the variation of mobile application types leads to many topics for further study. Some of those could be:

- The benefits of Ionic to a company's project in compare to native mobile application development.
- The benefits of Ionic to a company's project in compare to other hybrid mobile application development.
- The benefits of the latest version of Ionic (Ionic 2).

9 LIST OF REFERENCES

Written References

Babbie, E. 2013. The Practice of Social Research. Belmont: Cengage.

Ghatol, R. & Patel, Y. 2012. Beginning PhoneGap: Mobile Web Framework for JavaScript and HTML5. New York: Apress.

Grifith, C. 2017. Mobile App Development with Ionic: Cross-platform Apps with Ionic, Angular & Cordova. California: O'Reilly Media.

Hevner, A., 2004. Design Science in Information Systems Research. Minneapolis: MIS Quarterly.

Khanna, R., Yusuf S., & Phan, H. 2017. Ionic: Hybrid Mobile App Development. Birmingham: Packt.

Panhale, M. 2016. Beginning Hybrid Mobile Application Development. California: Apress.

Pelletier, J., 2013. Mobile App Manual: The Blueprint. Withinsight.

Ravulavaru, A. 2012. Learning Ionic: Build real-time and hybrid mobile applications with Ionic. Birmingham: Packt.

Saunders, M., Lewis, P. & Thornhill, A. 2012. Research Methods for Business Students. 5th edition. Italy: Rotolito Lombarda.

Wilson, J. 2010. Essentials of Business Research: A Guide to Doing Your Research Project. London: SAGE.

Yusuf, S., 2016. Ionic Framework by Example. Birmingham: Packt.

Oral References

Nikula, J. 2018. CTO. SuperApp Oy. Interview 25 January 2018.

Electronic References

Dabit, N. 2016. BYOD: The Cost of Native Mobile App Development is Too Damn High! [accessed 22 March 2018]. Available at: https://www.bankinfosecurity.in/interviews/byod-manage-risks-i-1327

Hockly, N. 2012. Tech-savvy teaching: BYOD [accessed 05 March 2018]. Available at: https://s3.amazonaws.com/academia.edu.documents/29698916/Hockly_MET-21.4.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1520287802&Signature=j%2BwlzW7BgsJkmbaanp%2FcpKz%2FryI%3D&response-content-disposition=inline%3B%20filename%3DTech-savvy_teaching_BYOD.pdf

Ionic 2018. All about Ionic [accessed 05 March 2018]. Available at: https://ionicframework.com/about

Ionic 2018. Plugins [accessed 18 March 2018]. Available at: https://market.ionicframework.com/plugins

Ionic 2018. The top open source framework for building amazing mobile apps [accessed 05 March 2018]. Available at: https://ionicframework.com/framework

Oberg, L. 2015 Evaluation of Cross-Platform Mobile Development Tools: Development of an Evaluation Framework [accessed 25 March 2018]. Available at: https://pdfs.semanticscholar.org/0bb5/9e0e7c29d2ea587f3d992c53acf67e5d8098.pdf

PhoneGap-Plugins 2018. PhoneGap & Cordova Plugins list [accessed 18 March 2018]. Available at: http://phonegap-plugins.com/

PSC 2018. Reliability and Validity [accessed 4 April 2018]. Available at: http://psc.dss.ucdavis.edu/sommerb/sommerdemo/intro/validity.htm

Shuttleworth, M. 2018. Validity and Reliability [accessed 4 April 2018]. Available at: https://explorable.com/validity-and-reliability

Statista 2018. Number of smartphone users worldwide from 2014 to 2020 (in billions) [accessed 10 March 2018]. Available at: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

Statista 2018. Number of smartphone users worldwide from 2014 to 2020 (in billions) [accessed 10 March 2018]. Available at: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

Techopedia 2018. Native Mobile App [accessed 15 March 2018]. Available at: https://www.techopedia.com/definition/27568/native-mobile-app

Tom, F. 2011. BYOD: Manage the Risks [accessed 20 March 2018]. Available at: https://www.bankinfosecurity.in/interviews/byod-manage-risks-i-1327

Trello 2018. What is Trello? [accessed 24 March 2018]. Available at: https://help.trello.com/article/708-what-is-trello

USAID 2016. MOBILE PHONES TACKLING POVERTY [accessed 10 March 2018]. Available at: https://www.usaid.gov/infographics/50th/mobile-phones-tacking-poverty

WebWise Team 2012. What are mobile apps? [accessed 2 April 2018]. Available at: http://www.bbc.co.uk/webwise/guides/mobile-applications

APPENDICES

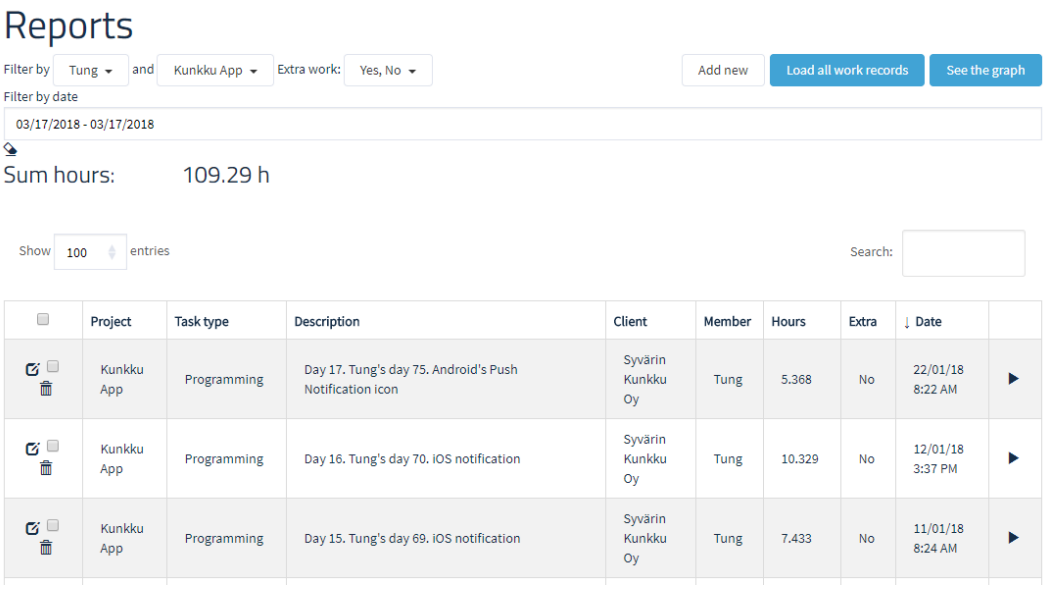
APPENDIX 1. The report of developer 1 in SuperProject (Page 1)

Kunkku App	Programming	Day 6 (Tung's day 45). Change caption + Adding the Facebook feed to pages	Syvärin Kunkku Oy	Tung	7	No	05/12/17 9:49 AM	▶
Kunkku App	Programming	Day 5. (Tung's day 44). Continue with implement WordPress rest API	Syvärin Kunkku Oy	Tung	8.521	No	04/12/17 9:52 AM	▶
Kunkku App	Programming	Sunday's work. Studying WordPress rest API	Syvärin Kunkku Oy	Tung	2.761	No	03/12/17 6:30 PM	▶
Kunkku App	Programming	Day 4. (Tung's day 43). Implement WordPress rest API	Syvärin Kunkku Oy	Tung	7.986	No	01/12/17 9:24 AM	▶
Kunkku App	Programming	Day 3. (Tung's day 42) + night. Create a real Feedback Form + Handling with PHP + Test build for iOS and Android	Syvärin Kunkku Oy	Tung	10	No	30/11/17 9:12 AM	▶
Kunkku App	Programming	Day 2. (Tung's day 41). Update environment + checking manual	Syvärin Kunkku Oy	Tung	7.771	No	29/11/17 9:06 AM	▶
Kunkku App	Programming	Day 2 (night). Check Creator's codes	Syvärin Kunkku Oy	Tung	2	No	29/11/17 8:11 PM	▶
Kunkku App	Programming	Day 1. (Tung's day 40). Ionic Creator + Export	Syvärin Kunkku Oy	Tung	6.847	No	28/11/17 8:15 AM	▶

APPENDIX 2. The report of developer 1 in SuperProject (Page 2)

Kunkku App	Programming	Day 14. Tung's day 68. Working with Push notification.	Syvärin Kunkku Oy	Tung	7.640	No	10/01/18 8:24 AM	▶
Kunkku App	Programming	Day 13. Tung's day 64 (Change hamburger icon + fix fb feed)	Syvärin Kunkku Oy	Tung	0.971	No	06/01/18 4:56 PM	▶
Kunkku App	Programming	Day 12. Tung's day 60 (Fixing bugs)	Syvärin Kunkku Oy	Tung	0.143	No	28/12/17 9:59 AM	▶
Kunkku App	Programming	Day 11. Tung's day 59 (Fixing PDF link, fix CORS -> iOS notification now enable, fixing iframe Facebook in app)	Syvärin Kunkku Oy	Tung	3.102	No	27/12/17 12:45 PM	▶
Kunkku App	Programming	Day 10. (Tung's day 49) Fixing Bugs.	Syvärin Kunkku Oy	Tung	3.200	No	12/12/17 8:32 AM	▶
Kunkku App	Programming	Day 9 (Tung's day 48). Adding map	Syvärin Kunkku Oy	Tung	7.680	No	11/12/17 10:04 AM	▶
Kunkku App	Programming	Day 8 (Tung's day 47). Booking Form	Syvärin Kunkku Oy	Tung	2.902	No	08/12/17 9:29 AM	▶
Kunkku App	Programming	Day 7 (Tung's day 46). Add more content from WordPress website and Fixing bugs.	Syvärin Kunkku Oy	Tung	7.633	No	07/12/17 9:34 AM	▶

APPENDIX 3. The report of developer 1 in SuperProject (Page 3)



Reports

Filter by Tung and Kunkku App Extra work: Yes, No

Add new | Load all work records | See the graph

Filter by date

03/17/2018 - 03/17/2018

Sum hours: 109.29 h

Show 100 entries

Search:

Project	Task type	Description	Client	Member	Hours	Extra	Date
Kunkku App	Programming	Day 17. Tung's day 75. Android's Push Notification icon	Syvärin Kunkku Oy	Tung	5.368	No	22/01/18 8:22 AM
Kunkku App	Programming	Day 16. Tung's day 70. iOS notification	Syvärin Kunkku Oy	Tung	10.329	No	12/01/18 3:37 PM
Kunkku App	Programming	Day 15. Tung's day 69. iOS notification	Syvärin Kunkku Oy	Tung	7.433	No	11/01/18 8:24 AM