Ilke Söylemez

WAPICE NEWS MOBILE APPLICATION

Information Technology

2017

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Dr. Ghodrat Moghadampour for his continuous support during my studies and this project. I am indebted to him for his endless guidance, encouragement and help in the development of my career generally.

My deep and sincere gratitude to my family for being present, despite the distance. Special thanks to my parents, my first teachers and earliest encouragers for lightening my life. My sincere thanks also goes to my friends. This journey would not have been possible without the support of my family and friends.

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Degree Program in Information Technology

ABSTRACT

Author	Ilke Söylemez
Title	Wapice News Mobile Application
Year	2017
Language	English
Pages	66
Name of Supervisor	Ghodrat Moghadampour

Since the mobile phones started to have an increasingly significant role in daily life, the mobile application development also started to be an important area in the software industry. The problem for mobile application developers is to develop a mobile application which supports all the devices and platforms on the market. This issue created a need for cross platform mobile applications. The cross platform mobile development refers to the development of mobile applications that could be used on different devices and platforms.

The aim of this project was to develop a cross platform mobile application for Wapice's employees that gives opportunity to follow internal news. The application was built with Xamarin which is a C# based solution for cross platform application development. A Portable Class Library was used to target increasing code sharing and simplifying debugging. The MVVM (Model-View-ViewModel) design pattern principles was followed for simplification of presentation separation. The Wapice News mobile application provides access to company's internal news and additionally company's social media pages. It allows the employees to read the company's internal news and shared news on the company's social media pages.

The main tools of the project included C# for backend development, XML for frontend development, SharePoint REST (Representational State Transfer) API (Application Program Interface) for application data persistence, Apache Subversion (SVN) for code management and Visual Studio for coding.

Since the project involves confidential information, only the most important aspects of the process are presented in this document.

Keywords Xamarin, SharePoint REST Service, PCL, MVVM

Table of Contents

LIST OF FIGURES, TABLES AND CODE SNIPPETS

LIST OF ABBREVIATIONS

API	Application Program Interface
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
PCL	Portable Class Library
OData	Open Data Protocol
REST	Representational State Transfer
MVVM	Model - View – View Model
QFD	Quality Function Development
UML	Unified Modeling Language
URL	Uniform Resource Locator
OPC UA	Object Linking and Embedding for Process Control Unified Architecture
CPQ	Configure, Price, Quote
XAML	Extensible Application Markup Language
AJAX	Asynchronous JavaScript and XML
XML	Extensible Markup Language
SVN	Apache Subversion
URL	Uniform Resource Locator
HMTL	Hypertext Markup Language
UWP	Universal Windows Platforms

CLI	Common Language Infrastructure
SDK	Software Development Kit
IOS	IPhone Operating System
CIM	Consumption Information Management

1 INTRODUCTION

In recent years, smartphones started to offer advanced technologies with much more functionality than a personal computer which makes the phone industry one of the fastest growing industries worldwide. The mobile phones have a huge impact on people's life, better networking systems and useful applications make smart phones an essential part of life.

The objective of this project was to develop a cross platform mobile application that helps Wapice employees to follow company's internal news. The application could be considered as an initial implementation of company's mobile application.

Nowadays communication has a key role in every organization. Wapice is aware of the importance of communication among the employees and aims to reach more internal cohesion. The Wapice News mobile application was designed for Wapice to help to reach their goal of internal cohesion.

The Wapice News mobile application is an internal news app for employees at Wapice. It is developed to meet Wapice's needs to keep employees up to date with the latest company news. Additionally, it gives an opportunity to employees to share their knowledge with their colleagues. Moreover, it could be considered as an open platform for announcing and exchanging the ideas.

The application is implemented as a cross platform mobile application with Xamarin. The Wapice News mobile application interacts remotely with the SharePoint object model by using REST (Representational State Transfer) web requests. The REST exposes all the SharePoint entities and operations that are available in the Wapice API (Application Program Interface). The MVVM (Model-View-ViewModel) design pattern is used to facilitate presentation separation and to write more organized and reusable code. A core project was implemented as a PCL (Portable Class Library) for increasing the amount of the shared code between the different mobile phone platforms.

This document provides an overview of developing a cross platform mobile application by using Xamarin. The solution was built for Wapice Oy and the aim

was to have a mobile application for the company which provides quick access to Wapice's internal news.

1.1 Wapice

Wapice Ltd. is a Finnish software company which is located in Vaasa. Established in 1999, Wapice Ltd currently employs 330 software and electronics experts in 9 locations around the Finland. It delivers solutions to globally leading industrial companies mainly in the fields of energy technology moving machinery. /1/

Wapice Ltd is leading technology partner for industrial businesses. Wapice's focus is on increasing the client's performance across all the functions by injecting information technology at its best. /1/

Wapice Ltd has six products which are IoT-Ticket, Summium, EcoReaction, Embedded OPC-UA (Object Linking and Embedding for Process Control Unified Architecture), CANrunner. The IoT-Ticket is a complete remote control and management system which integrates customer's machines and devices with their information systems. It allows the customers to monitor and control all their equipment simultaneously. Summium CPQ (Configure, Price, Quote) is a visual browser based tool which helps companies to sell complex and configurable product and service combinations. It configures the product and service combination based on predetermined sales options and generates a sales option specific quotation with illustrative pictures and additional documents. EcoReaction is a web based CIM (Consumption Information Management) solution that helps to improve the energy efficiency. EcoReaction gives information to the consumers about their own energy consumption whenever it is needed. Embedded OPC-UA Server is a server on a Linux platform which has been developed by Wapice Ltd. Wapice Ltd is an active member of the OPC Foundation since 2001 and has plenty of experience on developing OPC solutions in customer projects. The Canrunner is a powerful tool that provides advanced data monitoring and analysis capabilities for software developers and service engineers. /2/

2 RELEVANT TECHNOLOGIES

In this section, the main technologies used in this project are introduced. The main technologies were Xamarin, C#, JSON (JavaScript Object Notation), MVVM (Model-View-ViewModel), MvvmCross, PCL (Portable Class Library), SharePoint REST (Representational State Transfer) Service and SVN (Apache Subversion).

2.1 Android

Android is an open source Linux based mobile operating system developed by Google. It could be considered as a stack of software components which is divided into four main layers that are Linux Kernel layer, Native Library layer, Application Framework layer and Application layer. Each layer of the stack is tightly integrated to provide the optimal application development and execution environment for mobile devices. /3/

The Linux Kernel layer provides a level of abstraction between the device hardware and the upper layers of the Android component stack and contains the essential hardware drivers like keypad, display, camera etc. The next layer in the Android architecture includes the Android's native libraries. The Application Framework layer provides higher level services for applications like activity manager, content providers, resource manager, notification manager etc. The application layer located to the top of the Android component stack which function both as applications for users and to provide that developers can access from their own applications. /4/

2.2 C#

C# is an object-oriented language, but C# further includes support for component oriented programming. Contemporary software design increasingly relies on software components in the form of self-contained and self-describing packages of functionality. The key to such components is that they present a programming model with properties, methods, and events; they have attributes that provide declarative information about the component; and they incorporate their own

documentation. C# provides language constructs to directly support these concepts, making C# a very natural language in which to create and use software components. /5/

Xamarin supports cross platform development in C#. In this project, C# used as a main programming language which brought many significant features to help to write the code in less time. Properties, lambda expressions, event handling and asynchronous programming features could be considered as a main benefit in Xamarin. C# property system gives developers to access member variables safely and directly without writing setter and getter methods. Lambda expressions help developers to write local functions that can be passed as arguments or returned as the value of function. C# provides language level support for events through delegates which identifies the method that provides the response to the event. Asynchronous programming features keep applications responsive. They help to write code that performs long running tasks without blocking the main thread of application. /5/

2.3 JSON

JSON (JavaScript Object Notation) is an open-standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is one of the common data formats used for asynchronous browser/server communication, largely replacing XML, and is used by AJAX. /6/

The JSON could be considered a way to store information in an organized and minimal format. The data is written as name-value pairs. A key is always a sting and the value could be a boolean expression, number, array, object, null or string. The JSON is one of the supported data format by the SharePoint REST API. /7/

In Wapice News mobile application, the data is transmitted between the SharePoint REST and the client mobile application in JSON format by using REST web requests.

The following code snippet demonstrates the way of using JSON in Wapice News mobile application.

```
{
    “Title” : “Thinking Twice”,
    “Date” : “10.11.2016”,
    “Content” : “<url>”,
    “Author” : “Ilke Söylemez”
    “Address” : {
            “Email” : “ilke@sylmz.com”
            “Phone” : “12345678”
            “Address” : “Wolffintie 30 Vaasa”
            “Office” : “Futura”
    }
}
```

Code Snippet 1. An example of JSON content

2.4 MVVM

The MVVM (Model-View-ViewModel) is a design pattern which could provide possibility to separate presentation logic from business logic. It helps to create applications that are well organized and easy to maintain, test, debug and extend. The MVVM design consists of three primary components which are model, view and view-model. /8/ The interaction between the components could be seen from the following illustration.

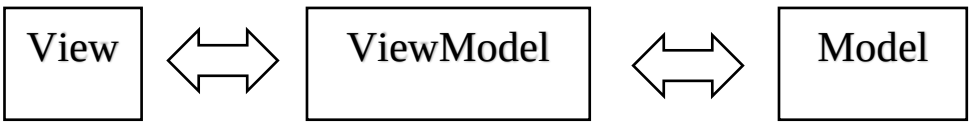

Figure 1. Interaction between MVVM components

- The model represents the data model along with business and validation logic. The model is not aware of the existence of the view model, it only contains

the objects with properties but not behaviors or events that manipulate the information. /9/

- The view model encapsulates the presentation logic and the data. It acts as a link between view and model. In another word, it represents the information being presented in the view and uses data binding to keep the view and view model in sync. Two way bindings allow the view model for being updated by the view. /9/
- The view is responsible for appearance of user interface elements. It contains data binding information, behaviors and events. The view is aware of the existence of the view model, it updates the view model's properties by using data bindings. It represents the data in the appropriate format and collects user interactions. /9/

2.5 MvvmCross

MvvmCross is a cross platform MVVM framework specifically developed for Xamarin which includes a lot of helper functions and extensions. It supports Xamarin. IOS, Xamarin.Android, Xamarin.Mac, Xamarin. Forms, UWP (Universal Windows Platforms) and WPF (Windows Presentation Framework). /10/

The MvvmCross provides a list of elevated level features that consists of MVVM architecture pattern, navigation system, data binding, platform specifics support, inversion of control and dependency injection. Inversion of control, dependency injection and plugins gives a possibility to have interface driven development and makes easy to target all the platforms. The possibility to override any part of MvvmCross could be also considered as a big advantage for developers. /10/

The MvvmCross increases the portability of applications; minimizes the work and makes the code clean and simple to maintain, in this way saves time and cost of the development phase.

2.6 Portable Class Library

The Portable Class Library (PCL) project enables you to write and build managed assemblies that work on more than one .NET Framework platform. You can create

classes that contain code you wish to share across many projects, such as shared business logic, and then reference those classes from different types of projects. /11/

The key benefit of building a cross platform application is being able to share the code across multiple platform specific projects. Thus, it helps to reduce the time and cost of developing, testing and maintaining the code.

Writing a PCL gives a possibility to select platforms which are going to be supported and does not requires inserting compiler directives switch code depending on each platform. It provides making changes on target platforms when it is needed. Visual Studio complies the library with new assemblies without causing any problem. /12/

2.7 SVN

Apache Subversion (SVN) is a software versioning and revision control system distributed under the Apache License. SVN is used as a version control system for the development of the Wapice News mobile application. The Version control system is a software that helps developers to work simultaneously across time and location and keep track of history of their work. It provides easy access to all previous versions of project. /13/

The SVN repositories are located on a server which has a hierarchy of directories and files. The repositories stores information in the form of a file system tree. Any number of clients could connect to the repositories. Connecting to a repository gives clients possibility to read or write to files.

2.8 Xamarin

“Xamarin is a Microsoft-owned San Francisco, California-based software company founded in May 2011 by the engineers that created Mono, Mono for Android and MonoTouch, which are cross-platform implementations of the Common Language Infrastructure (CLI) and Common Language Specifications (often called Microsoft .NET). “/14/

Xamarin presents a set of tools that delivers high performance compiled code with full access to all the native APIs. In addition to the .Net classes, it offers platform specific classes for Xamarin Android and Xamarin IOS, in this way it enables access to platform specific SDKs. This provides an identically matching link between the native APIs and the Xamarin libraries. /14/

Xamarin provides three core products for developing cross platform mobile application which are Xamarin.Forms, Xamarin.Android and Xamarin.IOS.

- Xamarin.Forms is a cross platform user interface toolkit that allows developers to create user interfaces that can be shared across different platforms. Xamarin.Forms applications are architected in traditional cross platform applications, Portable Class Libraries or Shared Projects are used to increase the amount of shared code between the platforms. Xamarin.Forms aims to maximize code sharing. /15/

- Xamarin.Android allows developers to create native Android applications using the user interface control as same as Android native applications. Anything developers can do in Java, could be also done in Xamarin.Android and C#. Xamarin.Android offers several Android API level settings that specify application's compatibility with versions of Android. Setting target framework specifies which framework to use in application, setting minimum Android version specifies the oldest supported Android version and setting target Android version specifies the version of Android that application is intended to run on. /16/

- Xamarin.IOS is used to develop IOS applications using C#. It exposes all the native user interface objects and native libraries provided by Apple. The Xamarin Studio and Visual Studio allows developers configure properties related to SDK. Setting the IOS SDK version allows developers to use different versions of an Apple published SDK. Setting Deployment Target specifies the minimum required version of the operating system that the application will run on. /17/

The following figure represents the main difference between the Xamarin native and Xamarin Forms applications.

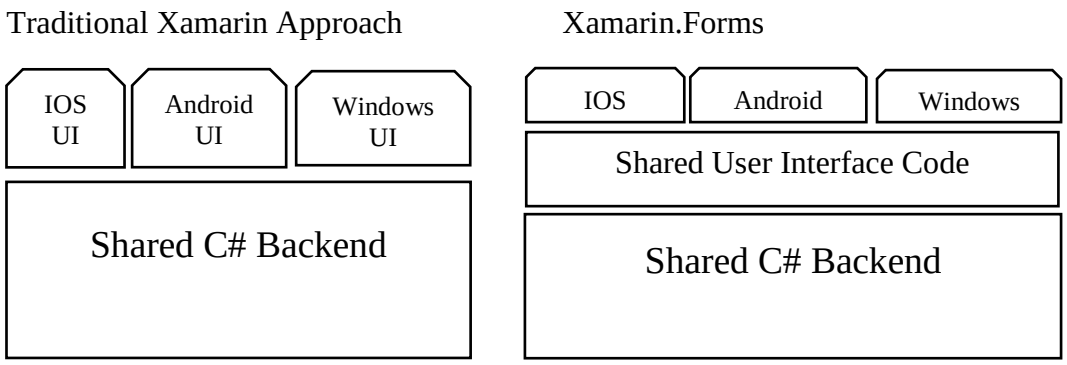

Figure 2. Difference between Xamarin Native and Xamarin.Forms

2.9 SharePoint Rest Service

SharePoint is a platform to support browser based collaboration and content management system which is developed by Microsoft. SharePoint's core functions are to store documents effectively and to bring an organization together so that everyone receives information that is relevant to them. /18/

SharePoint 2013 introduced a Representational State Transfer (REST) service that is comparable to the existing SharePoint client object models. The developers can interact remotely with SharePoint data by using any technology that supports REST web requests. This means that developers can perform Create, Read, Update, and Delete (CRUD) operations from their SharePoint Add-ins, solutions, and client applications, using REST web technologies and standard Open Data Protocol (OData) syntax. /19/

The SharePoint REST service is implemented in a client.svc file in the virtual folder /_vti_bin on the SharePoint. It supports the abbreviation "_api" as a substitute for "_vti_bin/client.svc" therefore the base URL could be constructed like "http://<domain>/<site url>/_api/". The service-relative URLs of specific

endpoints are appended to given base. The following snippet demonstrates retrieving a list from a SharePoint page. /20/

```
http://<domain>/<site url>/_api/web/lists/getByTitle('sampleList')/
```

Code Snippet 2. Retrieving a list by specifying its title

3 APPLICATION DESCRIPTION

The Wapice News app is a mobile application which gives possibility to follow Wapice's internal and social media news. The application was implemented as a cross platform application with Xamarin which is a development environment for cross platform mobile applications that offers performance similar to the native applications.

The application consists of a single core project which is written as a PCL and Xamarin.Android project for Android platform. The core project contains services, view models and models. The Xamarin.Android project contains the views and platform specific codes which are responsible for interreacting with the core project. The MVVM design pattern are followed for the simplification of presentation separation and to increase testability, flexibility and maintainability. The Wapice News mobile application interacts remotely with the SharePoint object model by using REST web requests. The REST exposes all the SharePoint entities and operations that are available in the Wapice API.

The authentication is handled by passing the current user credentials to the HttpWebRequest class. The user credential consists of a username and a password which are registered in the company's system. Currently, the Wapice News mobile application is restricted by Wapice's network. The future plan contains allowing restricted intra access over internet with a mobile phone which has the Wapice client certificate installed.

3.1 Main Functions

In this section, a detailed description of the application and the expected requirements of the application are provided. The required functions for Wapice News mobile application are prioritized in three different categories which are normal requirements, expected requirements and exciting requirements. The requirements based on level of priorities are listed below.

Normal Requirements with priority level 1

- The application should start with a login screen
- The user should be able to login to the system by providing username and password
- The user should be connected to the company's network to be able to login to the system
- The application should have the remember me option on the login page
- After a successful login process, if the remember me option is selected, the application should start at the home page
- The home page should consist of three tabs and their contents which are corporate news headlines, social stream headlines and blog stream headlines.
- Switching the news sections should be done by swiping or selecting from tabs.
- The user should be able to read the news by selecting the headline from list.
- The user should be able to logout.

Expected Requirements with priority level 2

- The user should be able to see a descriptive message, if login fails.
- The application should have a user-friendly user interface design.
- The application should be installed and run on all Android Mobile devices with lowest version 3.0

Exciting Requirements with priority level 3

- The user should be able to view the company's news which is shared on LinkedIn.
- The user should be able to view the company's news which is shared on Twitter.
- The user should be able to view the company's news which is shared on Facebook.
- PCL should be used to reduce the time and cost of implementation.

In this section, it is defined in a step by step listing of what the user will see and do while using this application.

The following figure shows the basic flow of events for Wapice News mobile application. It provides an external view of Wapice News mobile application and its interactions with the user.

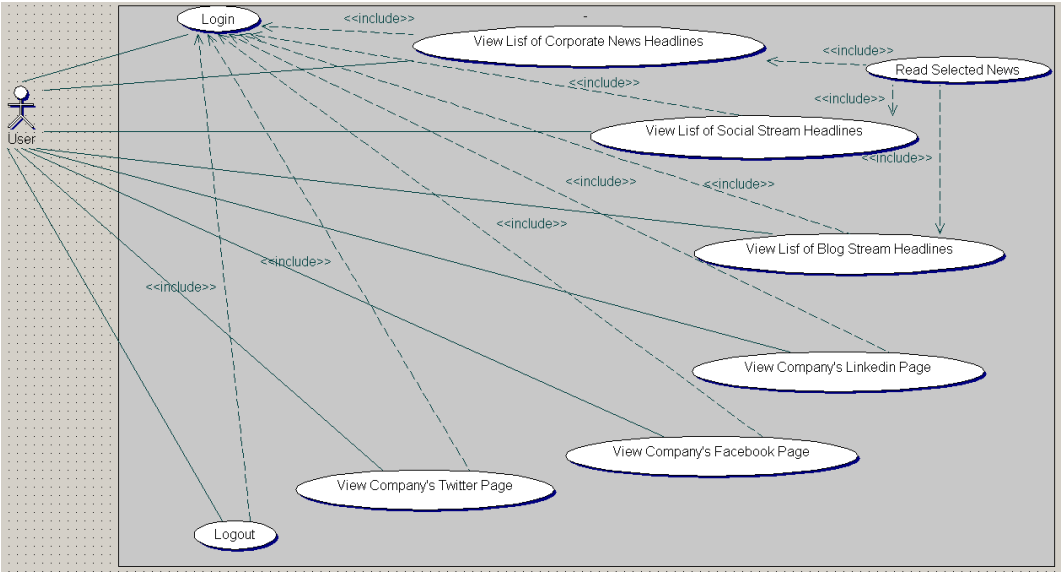

Figure 3. The main functions provided by Wapice News mobile application

3.2 Functional Definition

This section provides functional achievement of each function. Each table is associated with a goal of one of the use cases and each diagram identifies sequences of related events that result in some desired outcome. A detailed interaction that occur over time among the objects associated with each of the use cases will be shown under the relevant functional specification table.

Table 1. Functional Specification for login

Case	Login
Preconditions	The application should be launched
Input	The users log in to Wapice News app by using their username and password
Description	The application starts with a login screen that activates a login routine that can route to the application's home page or to a failure notice.
Exceptions and errors	Network connection
Result and outputs	If the username and password is valid, the application navigates to home page. If the provided information is invalid, the application informs users to check their user names and passwords.

Login page is the first page of the application that is shown when users launch the application for first time use. The user must provide username and password and click the "Login" button. If the user credentials are invalid, an appropriate error message will be displayed, otherwise the user will be logged in. After successfully

activating the application, future access to the application will not require login information.

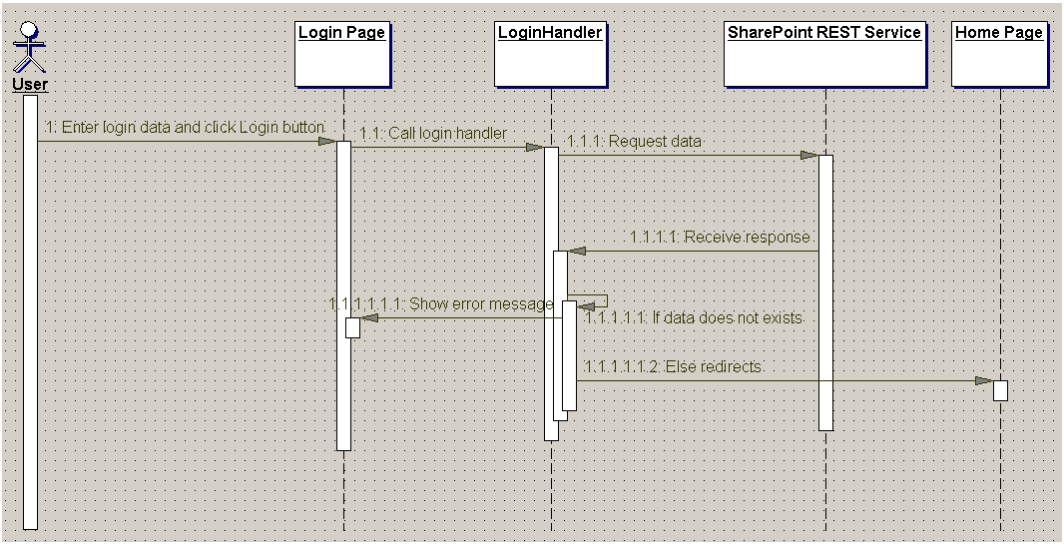

Figure 4. Implementation of login functionality

The functional specification table for a reading list of corporate news headlines consists of descriptions of user tasks, possible user input actions and the application response actions.

Table 2. Functional specification for a reading list of corporate news headlines

Case	Read list of corporate news headlines
Preconditions	Log in to system by providing username and password
Input	No input is required

Description	After a successful login, the application displays the home page which consists of news streams based on categories. Changing the category is done by selecting related category from the tab section. The Corporate News is selected by default. It contains a list of headlines sorted by published date. Swiping up to scroll down the screen automatically hides the header to maximize screen real state. Scrolling to the tab of the screen restores the header.
Exceptions and errors	Network connection
Result and outputs	Users can read the list of corporate news headlines which are sorted by published date.

Figure 5 shows the sequence of steps required for listing corporate news headlines. The user must provide valid user credentials on login page and click on Login button. After a successful login, the user will be redirected to the home page that shows list of corporate news headlines by default. The corporate news page calls the service handler and service handler fetches the data from SharePoint REST service. And finally, the corporate news page displays the requested data.

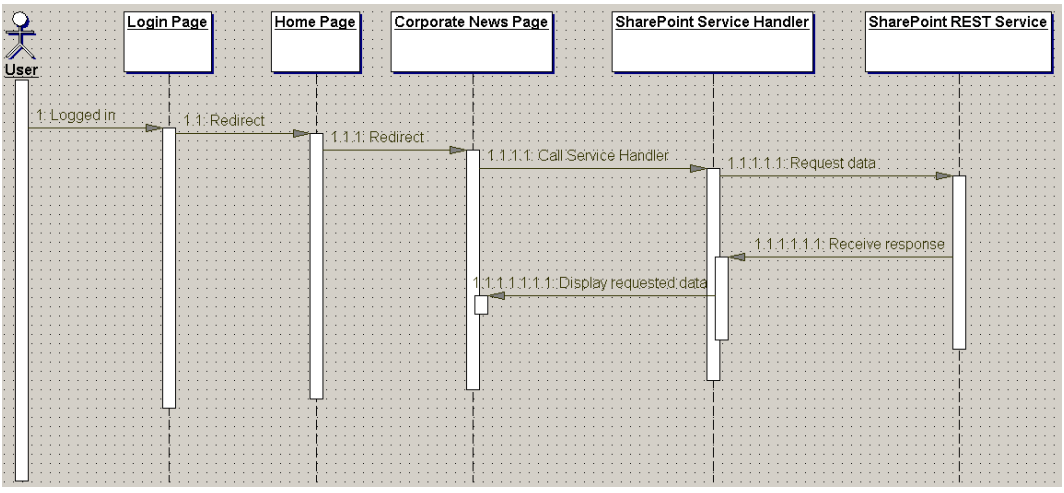

Figure 5. Implementation of read list of corporate news functionality

The functional specification table for the reading list of social stream headlines consists of descriptions of user tasks, possible user input actions and the application response actions. The following table dissects reading list of social stream headlines case and clarifies each statement.

Table 3. Functional specification for reading list of social stream headlines

Case	Reading list of social stream headlines
Preconditions	Log in to the system by providing username and password Selecting the related category from the tab section
Input	No input is required
Description	Social stream news headlines are listed under the Social Stream tab. The home page displays Corporate News stream by default. Users can navigate between the different categories by tapping on their respective tab names or swiping left or right. Swiping up to scroll down the screen automatically hides the header to maximize screen real state. Scrolling to the tab of the screen restores the header.
Exceptions and errors	Network connection
Result and outputs	Users can read the list of social stream news headlines which are sorted by published date.

Figure 6 describes how the user view list of social stream headlines. After the login process, the application shows home page which displays corporate news stream by default. The user must swipe left or tap on Social Stream from tab control to be able to view the social stream headlines.

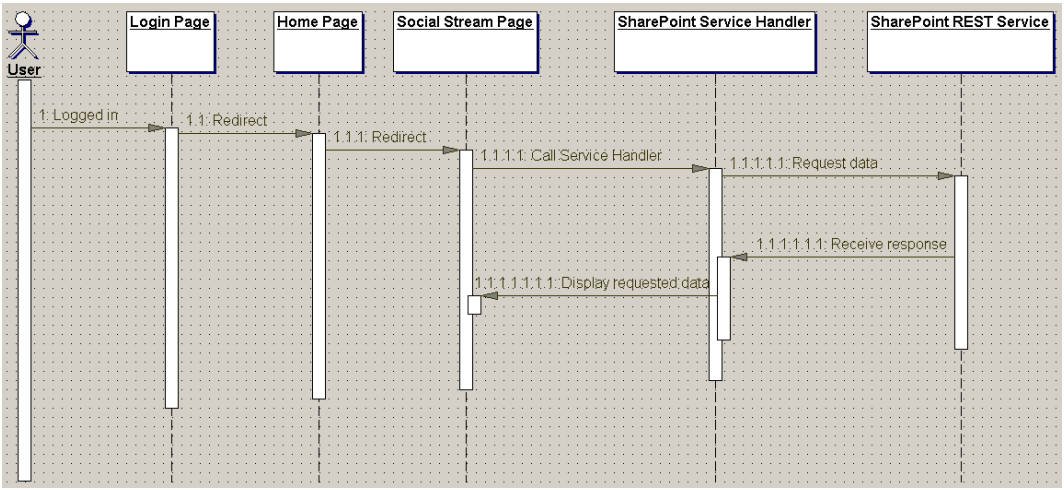

Figure 6. Implementation of read list of social stream functionality

The functional specification table for reading list of blog stream headlines consists of descriptions of user tasks, possible user input actions and the application response actions. The following table dissects a reading list of blog stream headlines case and clarifies each statement.

Table 4. Functional specification for reading list of blog stream headlines

Case	Reading list of blog stream headlines
Preconditions	Log in to the system by providing username and password Selecting the related category from the tab section
Input	No input is required
Description	Blog stream news headlines are listed under the Blog Stream tab. The home page displays Corporate News stream by default. Users could navigate between the different categories by tapping on their respective tab names or swiping left or right. Swiping up to scroll down the screen automatically hides the header to maximize the screen real state. Scrolling to the tab of the screen restores the header.
Exceptions and errors	Network connection
Result and outputs	Users can read the list of blog stream news headlines which are sorted by date of publication.

Figure 7 describes how the user views the list of blog stream headlines. The user must provide a valid username and password and tab on Login button to be able to log in to the system. The home page which contains the corporate news as a default selection is shown. The user must swipe left or tab on Blog Stream from tab control to be able to view the list of blog stream headlines.

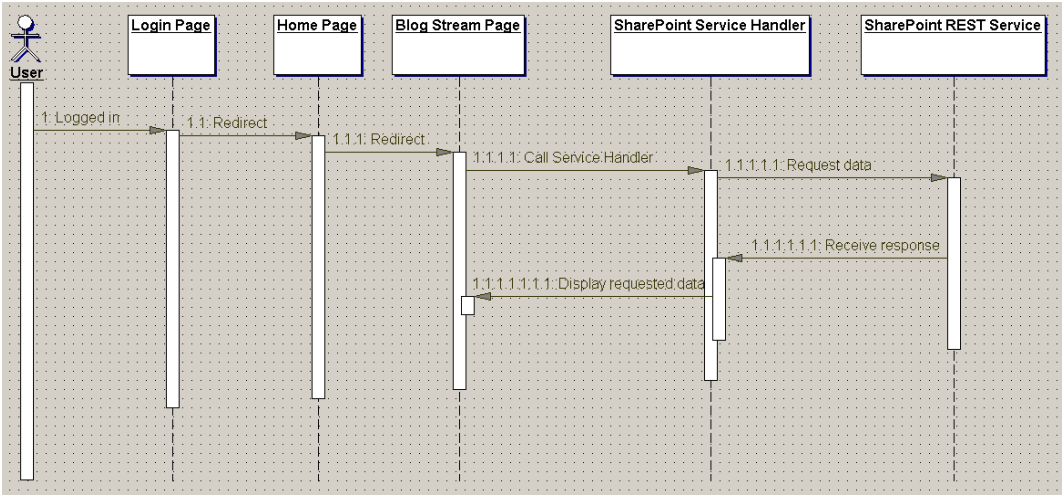

Figure 7. Implementation of read list of blog stream headlines functionality

The functional specification table for reading the selected news consists of descriptions of user tasks, possible user input actions and the application response actions. Table 6 dissects reading the selected news case and clarifies each statement.

Table 5. Functional Specification for Reading Selected News

Case	Read selected news
Preconditions	Selecting a headline
Input	No input is required
Description	Selecting a headline navigates to the news detail page and provides more information to the user about the selected news.
Exceptions and errors	Network connection
Result and outputs	The users could read the detailed information about the news which they have selected

Figure 8 describes the sequence of steps for reading the selected news. The user must provide user credentials to be able log in to the application. When the user performs a selection operation on the listed news stream headlines from the home page, the SharePointService handler will be called which will retrieve news details. The news detail page displays requested data.

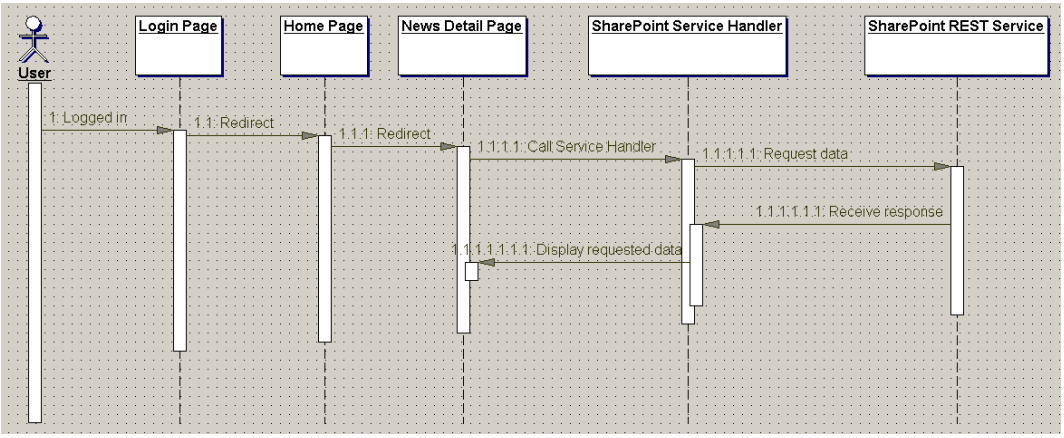

Figure 8. Implementation of selected news functionality

The functional specification table for viewing company's LinkedIn page consists of descriptions of user tasks, possible user input actions and the application response actions. The following table dissects view company's LinkedIn page case and clarifies each statement.

Table 6. Functional specification for viewing company's LinkedIn page

Case	View company's LinkedIn page
Preconditions	Selecting LinkedIn from the side menu
Input	No input is required

Description	Selecting LinkedIn from side menu shows company's LinkedIn page.
Exceptions and errors	Network connection
Result and outputs	The users could read the news which are shared on LinkedIn

Figure 9 describes sequence of steps for viewing the company's LinkedIn page. The user must provide a valid username and password and tab on Login button to be able to log in to the Wapice News mobile application. The user must select LinkedIn from the left menu. The LinkedIn page calls the social media service handler and requests access to resources hosted by the LinkedIn server. And finally, the LinkedIn page displays the data which is received from the social media service.

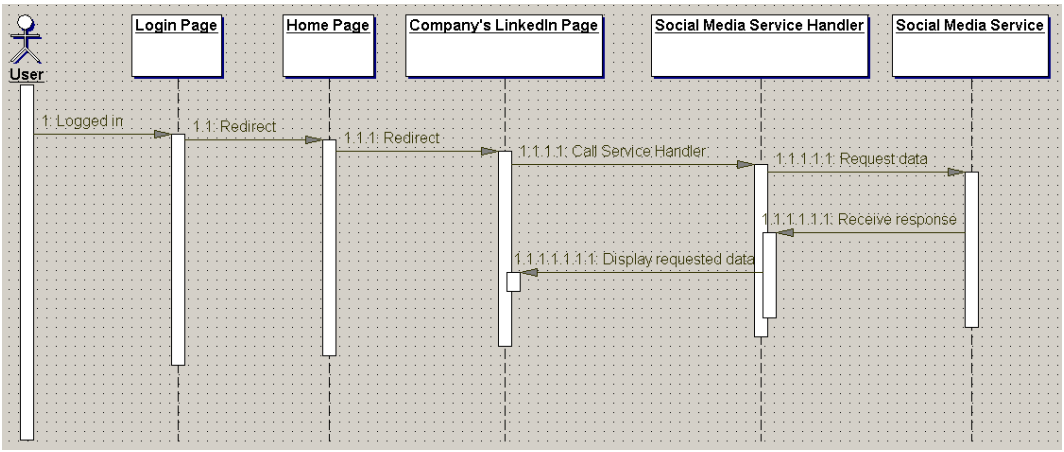

Figure 9. Implementation of view company's LinkedIn page functionality

The functional specification table for viewing the company's Facebook page consists of descriptions of user tasks, possible user input actions and the application response actions. The following table dissects view the company's Facebook page case and clarifies each statement.

Table 7. Functional specification for viewing company's Facebook page

Case	View the company's Facebook page
Preconditions	Selecting Facebook from the side menu
Input	No input is required
Description	Selecting Facebook from side menu views company's Facebook page
Exceptions and errors	Network connection
Result and outputs	The users can read the news which are shared on Facebook

Figure 10 describes the sequence of steps for viewing the company's Facebook page. The user must provide user credentials and tab on Login button to be able to log in to the Wapice News mobile application. The selection for viewing the Facebook page is done from the left menu. The user must tap on Facebook from listed options. The Facebook page calls the social media service handler and requests access to resources hosted by Facebook server. As a final step, Facebook page displays the data which is received from social media service.

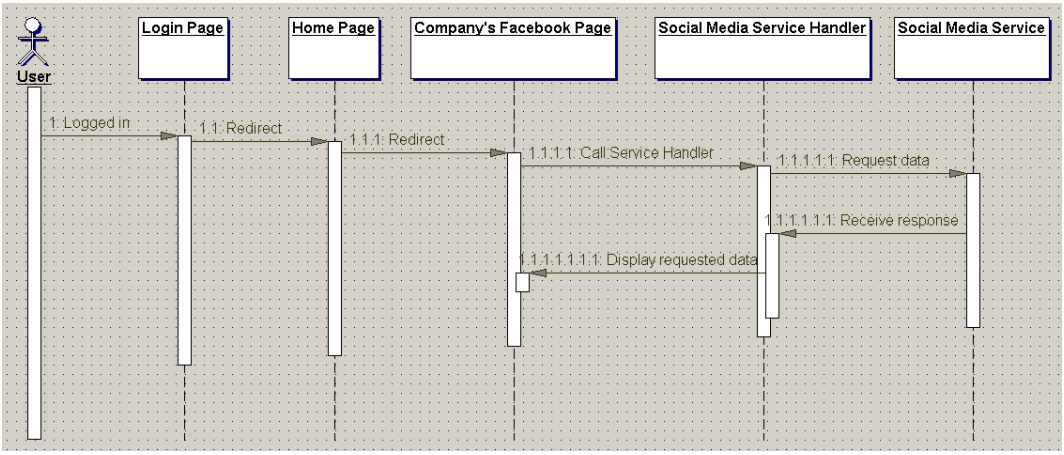

Figure 10. Implementation of view company's Facebook page functionality

The functional specification table for viewing the company's Twitter page consists of descriptions of user tasks, possible user input actions and the application response actions. The following table dissects view company's Twitter page case and clarifies each statement.

Table 8. Functional specification for viewing the company's Twitter page

Case	View company's Twitter page
Preconditions	Selecting Twitter from the side menu
Input	No input is required
Description	Selecting Twitter from the side menu views the company's Twitter page
Exceptions and errors	Network connection
Result and outputs	The users can read the news which are shared on Twitter

Figure 11 describes sequence of steps for viewing the company's Twitter page. The user must provide user credentials and tab on Login button to be able to log in to the Wapice News mobile application. The selection for viewing Wapice's Twitter page is done through the left menu which has listed options. The user must tab on Twitter from menu. The Twitter page calls the social media service handler and requests access to resources hosted by Twitter server. As a final step, Twitter page displays the data which is received from social media service.

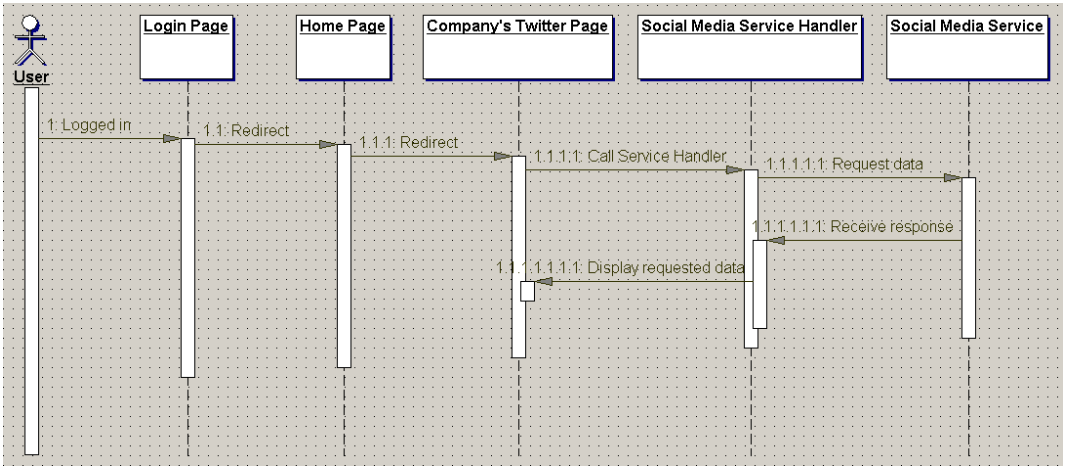

Figure 11. Implementation of view company's Twitter page functionality

The functional specification table for logout consists of descriptions of user tasks, possible user input actions and the application response actions. The following table dissects logout case and clarifies each statement.

Table 9. Functional specification for logout

Case	Logout
Preconditions	Log in to the system by providing username and password
Input	No input is required
Description	Logout is possible by selecting "Logout" from navigation drawer.
Exceptions and errors	Network connection
Result and outputs	Users can log out of the application.

The logout sequence diagram describes the sequential steps for user to logout. The user must be logged in to the Wapice News mobile application by providing username and password. The selection for Logout is done through the left menu. The user must tab on Logout from list of options. By selecting Logout, the Home page redirects the user to the Login page.

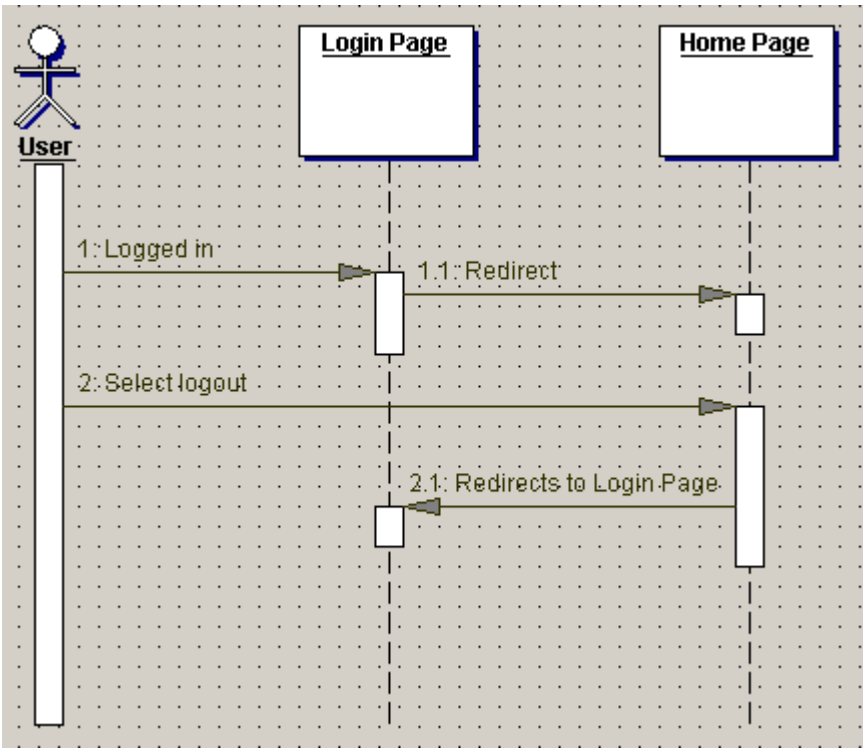

Figure 12. Implementation of logout functionality

4 GUI DESIGN

The user engagement is one of the key indicators for the success and therefore following specific mobile interface design principles has a significant role. The Material design's guideline is followed to increase the user engagement on Wapice News mobile application. This section describes user interfaces involved in Wapice News mobile application. Due to confidentiality reasons, only limited information will be given in this section.

The graphical user interface of Wapice News mobile application for Android is built with XML. The Wapice News mobile application starts with a Login page which consists of two input text fields for user credentials, a checkbox for the remember me option and a button for log in to the application. The figure below represents the login page.

Figure 11. Login page

The home page consists of news streams based on categories. Changing the category is done by selecting related category from tab section. The Corporate News is selected by default. It contains a list of headlines sorted by published date. Swiping up to scroll down the screen automatically hides the header to maximize screen real state. Scrolling to the tab of the screen restores the header. Social stream news headlines are listed under the Social Stream tab control. The users can navigate between the different categories by tapping on their respective tab name or swiping left or right. Blog stream news headlines are listed under the Blog Stream tab control. The users can navigate between the different categories by tapping on their respective tab name or swiping left or right. The figures below represents the home page.

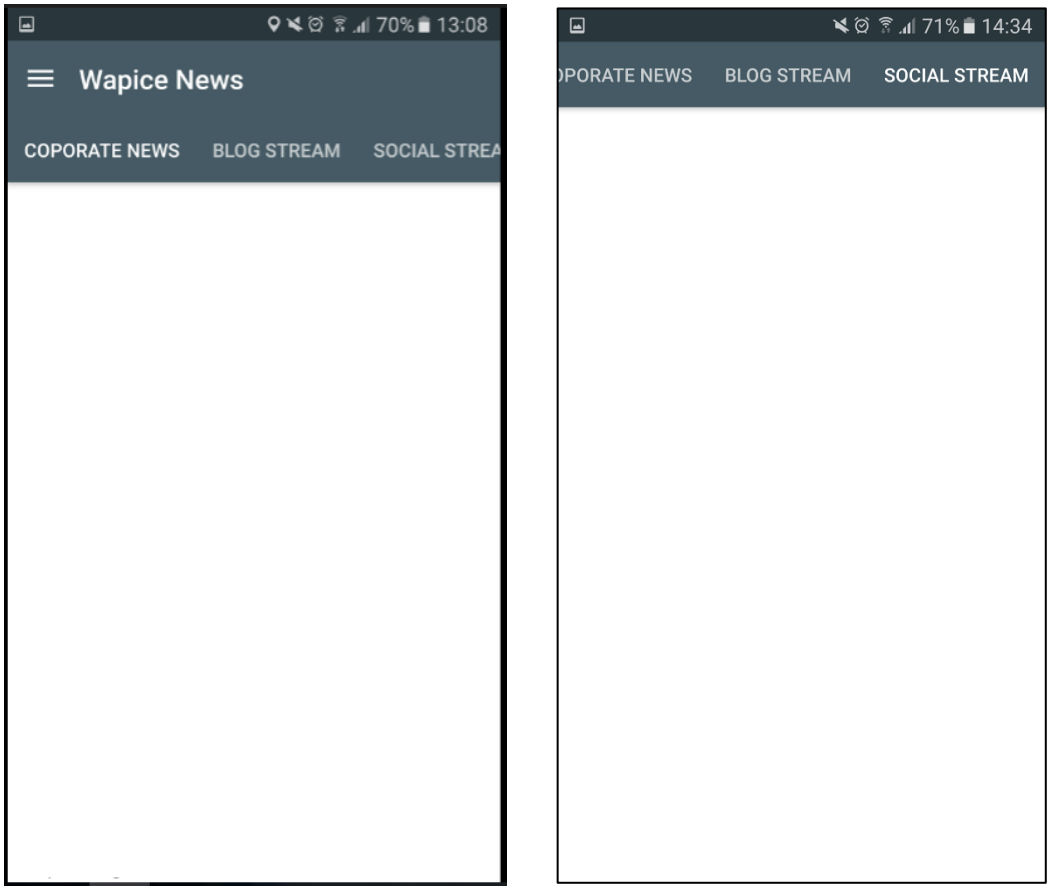

Figure 12. Home page

Selecting a headline navigates to the detail page and provides detailed information about the selected news. The figures below represent the detail page.

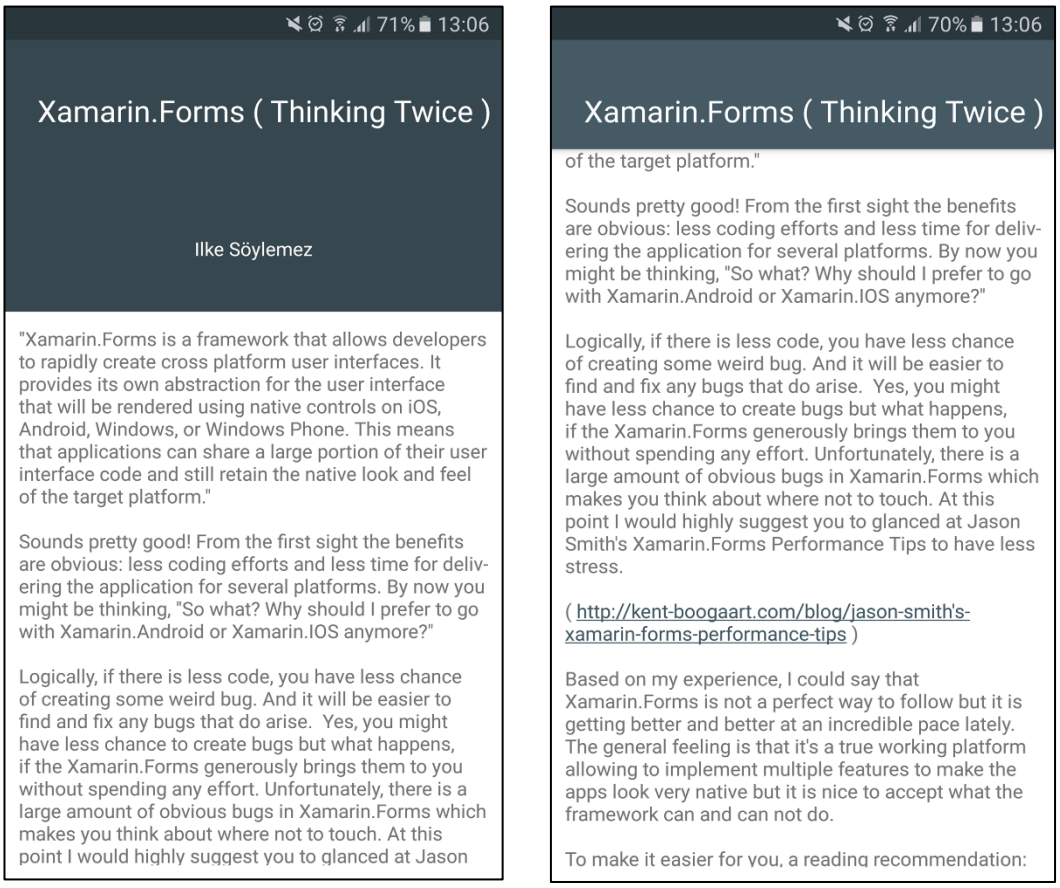

Figure 13. Detail page

A navigation drawer is used to display the list of options. It enables an edge swipe to open the drawer, but it is also accessible by tapping the menu icon from home page. The list of options are home, LinkedIn, Facebook, Twitter and Logout. Selecting name of the social media platform from the menu, views the company's page on the selected platform. Selecting home from the menu, navigates to the home page of the application. Selecting logout, gives a possibility to log out of the application. The figures below represent the company's social media pages.

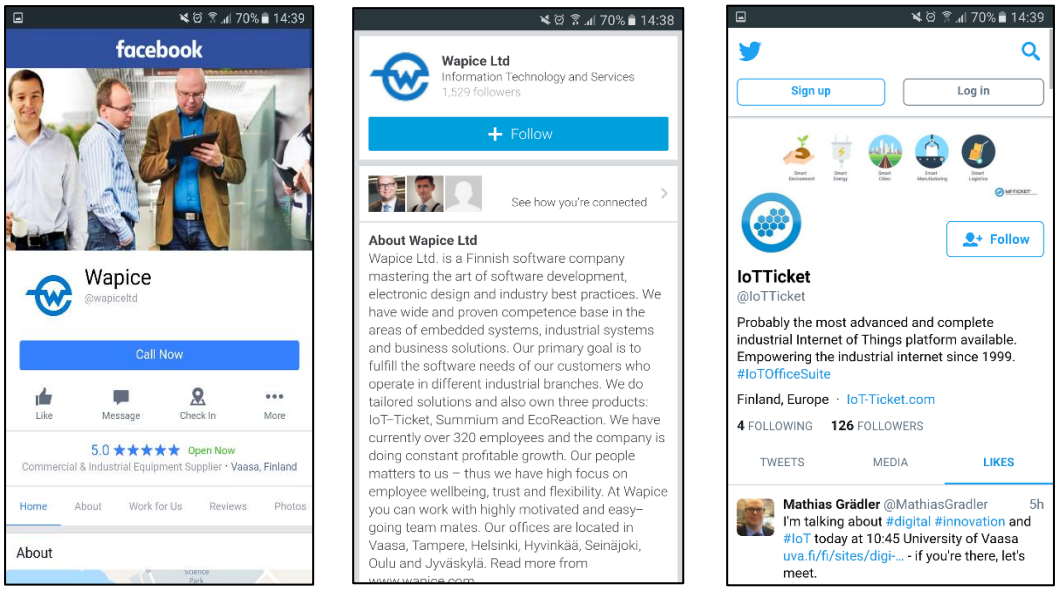

Figure 14. Social media pages

5 IMPLEMENTATION

The Wapice News mobile application is implemented as a cross platform application. It consists of a single core project which is written as a PCL and Xamarin.Android project for the Android platform. The core project contains all the services, models and view models. Xamarin Android project contains the views and platform specific codes for interacting with core project.

The MVVM design pattern principles are used for the simplification of presentation separation. The responsibility for the appearance and the layout of the user interfaces is separated from the responsibility of the business logic. Since the application is structured in MVVM design pattern, the implementation will be explained according to the core components which are View, View Model and Model. Due to confidentiality reasons, the given information will be limited.

5.1 Model Implementation

The Model of Wapice News mobile application encapsulates the application's business logic which consists of interfaces, services and data objects.

5.1.1 Login

The LoginRequest method is responsible for creating a login request. The following code snippet demonstrates the implementation of LoginRequest method.

```
public async Task<bool> LoginRequest(string username, string password)
{
   //Initializes a new instance of the HttpWebRequest class for current url
   HttpWebRequest request =  (HttpWebRequest)WebRequest.Create(url);
   CookieContainer container = new CookieContainer();
   //Sets the network credentials used for authenticating the request
   request.Credentials = new NetworkCredential(username, password);
   //Specifies the collection of CookieCollection objects
```

```
    request.CookieContainer = container;
    //Gets response
   var response = (HttpWebResponse)(await
                Task<WebResponse>.Factory.FromAsync(request.BeginGetResponse,
                request.EndGetResponse, null));
     //Represents the status of response, default value for completed successfully is 200
    if (!response.StatusCode.Equals(200))
       return false;
    return true;
}
```

Code Snippet 3. Function to create login request

5.1.2 View Corporate News Headlines

GetCorporateNewsStreamFeed method is responsible for retrieving the corporate news stream feed. It calls two helper methods which are CreateWebRequest and GetCorporateNewsList. CreateWebRequest method initializes a new instance of HttpWebRequest class for the current URL and sets the user credentials for authenticating the request. The GetCorporateNewsList method is responsible for creating a list of corporate news from the XML tree.

```
public async Task<IEnumerable<News>> GetCorporateNewsStreamFeed(string url)
    {
        //Calls CreateWebRequest method
        var request = CreateWebRequest(url);
        using (var response = (HttpWebResponse)(await
                Task<WebResponse>.Factory.FromAsync(request.BeginGetResponse,
                request.EndGetResponse, null)))
        {
          //Gets the data stream that is associated with the specified url
          using (var responseStream = response.GetResponseStream())
          {
            //Reads the bytes in responseStream
```

```
                using (var sr = new StreamReader(responseStream))
                {
                    // Read the stream to a string
                    string content = await sr.ReadToEndAsync().ConfigureAwait(false);
                    //Calls GetCorporateNewsList
                    News = GetCorporateNewsList(content);
                }
            }
            return News;
        }
    }
```

Code Snippet 4. Function to get corporate news stream feed

5.1.3 View Social Stream Headlines

The GetSocialStreamFeed method is responsible for retrieving the social news stream feed. It calls two helper methods which are CreateWebRequest and GetSocialStreamList. The CreateWebRequest method initializes a new instance of the HttpWebRequest class for the given URL and sets the user credentials for authenticating the request and all the other necessary settings. The GetSocialStreamList method is responsible for creating a list of social streams from the XML tree.

```
public async Task<IEnumerable<News>> GetSocialStreamFeed(string url)
    {
        //Calls CreateWebRequest method
        var request = CreateWebRequest(url);
        using (var response = (HttpWebResponse)(await
                    Task<WebResponse>.Factory.FromAsync(request.BeginGetResponse,
                    request.EndGetResponse, null)))
        {
            using (var responseStream = response.GetResponseStream())
```

```
            {
                //Gets the data stream that is associated with the specified url
                using (var sr = new StreamReader(responseStream))
                {
                    string content = await sr.ReadToEndAsync().ConfigureAwait(false);
                    //Calls GetSocialStreamList method
                    News = GetSocialStreamList(content);
                }
            }
            return News;
        }
    }
```

Code Snippet 5. Function to get social stream feed

5.1.4 View Blog Stream Headlines

The GetBlogStreamFeed method is responsible for retrieving the blog stream feed. It calls two helper methods which are CreateWebRequest and GetBlogStreamList. The CreateWebRequest method initializes a new instance of the HttpWebRequest class for given URL and sets the user credentials for authenticating the request and all the other necessary settings. The GetBlogStreamList method is responsible for creating a list of social streams from the XML tree.

```
public async Task<IEnumerable<News>> GetBlogStreamFeed(string url)
    {
        //Calls CreateWebRequest method
        var request = CreateWebRequest(url);
        using (var response = (HttpWebResponse)(await
                    Task<WebResponse>.Factory.FromAsync(request.BeginGetResponse,
                    request.EndGetResponse, null)))
        {
            using (var responseStream = response.GetResponseStream())
```

```
            {
                //Gets the data stream that is associated with the specified url
                using (var sr = new StreamReader(responseStream))
                {
                    string content = await sr.ReadToEndAsync().ConfigureAwait(false);
                    //Calls GetBlogStreamList method
                    News = GetBlogStreamList(content);
                }
            }
            return News;
        }
    }
```

Code Snippet 6. Function to get blog stream feed

5.2 View Model Implementation

The view model of Wapice News mobile application encapsulates the presentation logic and the data.

The view model implements properties and commands to which the view can data bind and notifies the view of any state changes through change notification events. The properties and commands that the view model provides define the functionality to be offered by the UI, but the view determines how that functionality is to be rendered. /21/

5.2.1 Login View Model

The following code snippet shows used properties for Login View Model. The properties are used for data binding on View and View Model. The public properties update the fields and fire the RaisePropertyChanged event when a property is set.

```
public string WapiceNewsTitle
        {
            get { return _wapiceNewsTitle; }
            set { _wapiceNewsTitle = value; RaisePropertyChanged(() => WapiceNewsTitle); }
        }
        private string _username;
        public string Username
        {
            get { return _username; }
            set { _username = value; RaisePropertyChanged(() => Username); }
        }

        private string _password;
        public string Password
        {
            get { return _password; }
            set { _password = value; RaisePropertyChanged(() => Password); }
        }

        private bool _isChecked;
        public bool IsChecked
        {
            get { return _isChecked; }
            set { _isChecked = value; RaisePropertyChanged(() => IsChecked); }
        }
```

Code Snippet 7. Getter and setter of login

The HandleLogin method is responsible for calling the LoginRequest from the LoginService. The Settings plugin is used to save the user preferences. The code snippet 6 demonstrates HandleLogin method.

```
private void HandleLogin()
```

```
{
    //Shows loading icon during the login process
    UserDialogs.Instance.ShowLoading();
    //Calls the login service to make and runs LoginRequest method
     loggedIn = Task.Run(() => _loginService.LoginRequest(Username.Trim(),
                        Password.Trim())).Result;
    if (loggedIn)
    {
        try
        {
            //Checks remember me selection
            if (IsChecked)
            {
                //Sets  user preferences
                Settings.UserNameSettings = Username.Trim();
                Settings.PasswordSettings = Password.Trim();
                Settings.RememberUserSettings = true;
            }
            else
            {
                //Resets user preferences
                Settings.UserNameSettings = string.Empty;
                Settings.PasswordSettings = string.Empty;
                Settings.RememberUserSettings = false;
            }
            //Navigates to Home page
            ShowViewModel<NewsViewPagerViewModel>();
        }
        catch (Exception e)
        {
            //Alerts user about the possible problems
            UserDialogs.Instance.Alert(ErrorMessagesService.LoginError);
        }
```

```
    }
    //Hides loading icon
    UserDialogs.Instance.HideLoading();
}
```

Code Snippet 8. Function to handle login

This section of code snippet shows binding login command to the UI Control.

```
public IMvxCommand LoginCommand
{
    get
    {
        return new MvxCommand(() =>
        {
            //Checks null or empty conditions on username and password
            if (string.IsNullOrEmpty(Username) || string.IsNullOrEmpty(Password))
            {
                //Alerts user about the required fields
                UserDialogs.Instance.Alert(ErrorMessagesService.RequiredFields);
                return;
            }
            //Calls HandleLogin method
            HandleLogin();
        });
    }
}
```

Code Snippet 9. Login command

5.2.2 Corporate News Recycler View Model

The Corporate News Recycler View Model manages the links between Corporate News Recycler View and Corporate News Recycler View Model. It is responsible for converting model data in a format that can be easily consumed by the Corporate News Recycler View. The Corporate News View Model defines properties to support the Corporate News Recycler View. The public properties are used for updating the fields and firing the RaisePropertyChanged event.

The following code snippet demonstrates the link between Corporate News Recycler View Model and Corporate News Recycler View.

```
public CorporateNewsRecyclerViewModel(IWapiceNewsService wapiceNewsService)
{
   _wapiceNewsService = wapiceNewsService;
   News = new List<News>(Task.Run(() =>
                        _wapiceNewsService.GetCorporateNewsStreamFeed(url)).Result);
}
```

Code Snippet 10. Corporate news recycler view model constructor

5.2.3 Social Stream Recycler View Model

The Social Stream Recycler View Model handles the links between Social Stream Recycler View and Social Stream Recycler View Model. It is responsible for converting the model data into a format that can be easily consumed by the Social Stream Recycler View. The Social Stream Recycler View Model defines properties to support the Social Stream Recycler View. The public properties are used for updating the fields and firing the RaisePropertyChanged event.

The following code snippet demonstrates the link between Social Stream Recycler View Model and Social Stream Recycler View.

```
public SocialStreamRecyclerViewModel(IWapiceNewsService wapiceNewsService)
{
   _wapiceNewsService = wapiceNewsService;
   News = new List<News>(Task.Run(() =>
            _wapiceNewsService.GetSocialStreamFeed(url)).Result);
}
```

Code Snippet 11. Social stream recycler view model constructor

5.2.4 Blog Stream Recycler View Model

The Blog Stream Recycler View Model handles the links between Blog Stream Recycler View Model and Blog Stream Recycler View. It is responsible for converting the model data into a format that can be easily consumed by the Blog Stream Recycler View. The Blog Stream Recycler View Model defines properties to support the Blog Stream Recycler View. The public properties are used for updating the fields and firing the RaisePropertyChanged event.

The following code snippet demonstrates the link between Blog Stream Recycler View and Blog Stream Recycler View Model.

```
public BlogStreamRecyclerViewModel(IWapiceNewsService wapiceNewsService)
{
   _wapiceNewsService = wapiceNewsService;
   News = new List<News>(Task.Run(() =>
            _wapiceNewsService.GetBlogStreamFeed(url)).Result);
}
```

Code Snippet 12. Blog stream recycler view model constructor

5.3 View Implementation

The View of Wapice News mobile application clarifies the structure and appearance of what the user sees on the screen. It references the Wapice News View Models

through their DataContext property. Wapice News Views get the data from their View Models through property bindings, or invoking methods on the View Models.

5.3.1 Login

The implementation of the login view is demonstrated in the following code snippets.

Code snippet 11 represents the implementation of binding for the Wapice News mobile application title.

```
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:gravity="center"
local:MvxBind="Text WapiceNewsTitle; Typeface StringToFont('PrincessSofia')
android:textSize="56dp"
android:textColor="@color/blue_gray" />
```

Code Snippet 13. Binding for the Wapice News mobile application title

The following code snippets describe the implementation of the username, password and remember me sections. The MvxBind property is used for binding and the Android Design Support library is used to implement Material Design principles. It offers a wide range of implementations of several portions of the Material Design specification in a fully backwards compatible fashion when combined with Support Library v7 AppCompat. /22/

```
<android.support.design.widget.TextInputLayout
        android:id="@+id/usernameWrapper"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="@dimen/element_margin_left"
        android:layout_marginRight="@dimen/element_margin_right"
        android:layout_marginBottom="@dimen/element_margin_medium"
        android:focusable="true"
        android:focusableInTouchMode="true">
        <EditText
           android:id="@+id/username"
           android:layout_width="match_parent"
           android:layout_height="wrap_content"
           android:inputType="text"
           android:textColor="@color/blue_gray"
           android:hint="Username"
           local:MvxBind="'Text Username" />
      </android.support.design.widget.TextInputLayout>
```

Code Snippet 14. Implementation of username for login

```
<android.support.design.widget.TextInputLayout
        android:id="@+id/passwordWrapper"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/usernameWrapper"
        android:layout_marginLeft="@dimen/element_margin_left"
        android:layout_marginRight="@dimen/element_margin_right"
        android:layout_marginBottom="@dimen/element_margin_medium"
        android:layout_marginTop="4dp"
        android:focusable="true"
        android:focusableInTouchMode="true">
        <EditText
           android:id="@+id/password"
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:inputType="textPassword"
            android:textColor="@color/blue_gray"
            android:hint="Password"
            local:MvxBind="'Text Password" />
        </android.support.design.widget.TextInputLayout>
```

Code Snippet 15. Implementation of password for login

```
<CheckBox
        android:id="@+id/checkboxRememberMe"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/blue_gray"
        android:layout_marginLeft="@dimen/element_margin_left"
        android:layout_marginTop="@dimen/element_margin_medium"
        android:layout_marginBottom="@dimen/element_margin_medium"
        local:MvxBind="Checked IsChecked"
        android:text="Remember me" />
```

Code Snippet 16. Implementation of remember user preferences

```
<Button
        android:id="@+id/btnLogin"
        android:layout_marginTop="@dimen/element_margin_medium"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="@dimen/element_margin_left"
        android:layout_marginRight="@dimen/element_margin_right"
        android:layout_marginBottom="@dimen/element_margin_medium"
        android:textColor="@color/blue_gray"
        android:text="Login"
        local:MvxBind="Click LoginCommand" />
```

Code Snippet 17. Implementation of login command

5.3.2 Recycler View

The Recycler View is used for presenting a list of Wapice News headlines. It could be considered as a more advanced and flexible version of the ListView with improved performance.

The implementation of the Recycler View is demonstrated in the following code snippet.

```
<MvvmCross.Droid.Support.V4.MvxSwipeRefreshLayout
    android:id="@+id/refresher"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    local:layout_behavior="@string/appbar_scrolling_view_behavior"
    local:MvxBind="Refreshing IsRefreshing; RefreshCommand ReloadCommand">
    <MvxRecyclerView
      android:id="@+id/recycler_view"
      android:scrollbars="vertical"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      local:MvxItemTemplate="@layout/news_recyclerview"
      local:MvxBind="ItemsSource News; ItemClick ItemSelected" />
  </MvvmCross.Droid.Support.V4.MvxSwipeRefreshLayout>
```

Code Snippet 18. Implementation of the recycler view

5.3.3 News Detailed Page

The following code snippet represents the implementation of the News Detailed page. The CollapsingToolBarLayout is used to create animated screen for the News Detailed page and the MvxBind property is used for binding.

```
<android.support.design.widget.AppBarLayout
        android:id="@+id/main.appbar"
        android:layout_width="match_parent"
        android:layout_height="200dp"
        android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
        android:fitsSystemWindows="true">
        <android.support.design.widget.CollapsingToolbarLayout
           android:id="@+id/main.collapsing"
           android:layout_width="match_parent"
           android:layout_height="match_parent"
           app:layout_scrollFlags="scroll|exitUntilCollapsed"
           android:fitsSystemWindows="true"
           app:contentScrim="?attr/colorPrimary"
           app:expandedTitleMarginStart="48dp"
           app:expandedTitleMarginEnd="64dp">
           <ImageView
              android:id="@+id/main.backdrop"
              android:layout_width="match_parent"
              android:layout_height="200dp"
              android:scaleType="centerCrop"
              android:fitsSystemWindows="true"
              android:src="@drawable/blueGray"
              app:layout_collapseMode="parallax" />
           <TextView
              android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:textSize="14sp"
              app:MvxBind="Text Author"
              android:paddingTop="120dp"
              android:gravity="center"
              android:paddingBottom="@dimen/newsDetail_margin_bottom"
              android:textColor="@color/white"
              android:id="@+id/toolbar_right_subtitle" />
```

```
        <android.support.v7.widget.Toolbar
            android:id="@+id/main.toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            app:popupTheme="@style/ThemeOverlay.AppCompat.Light"
            app:layout_collapseMode="pin">
            <TextView
                android:id="@+id/toolbar_title"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                app:MvxBind="Text Title"
                android:textAppearance="?android:attr/textAppearanceLarge"
                android:gravity="center"
                android:textColor="@color/white" />
        </android.support.v7.widget.Toolbar>
    </android.support.design.widget.CollapsingToolbarLayout>
</android.support.design.widget.AppBarLayout>
<android.support.v4.widget.NestedScrollView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:MvxBind="TextFormatted Content, Converter=TextToSpannable"
        android:id="@+id/textViewContent"
        android:padding="@dimen/newsDetail_margin_left" />
</android.support.v4.widget.NestedScrollView>
```

Code Snippet 19. Implementation of the News Detailed page

5.3.4 Navigation Drawer

The Navigation drawer displays the Wapice News mobile application's main navigation options on the left edge of the screen. The following code snippet demonstrates the Navigation Drawer implementation.

```
<item>
   <menu>
    <item
     android:id="@+id/nav_viewpager"
     android:icon="@drawable/homeGray"
     android:title="Home" />
    <item
     android:icon="@drawable/twitter"
     android:id="@+id/nav_twitter"
     android:title="Twitter" />
    <item
    android:icon="@drawable/linkedin"
    android:id="@+id/nav_linkedin"
    android:title="Linkedin" />
    <item
    android:id="@+id/nav_facebook"
    android:icon="@drawable/facebook"
    android:title="Facebook" />
   </menu>
  </item>
  <group android:id="@+id/nav_footer"  android:paddingBottom="5dp">
   <item
      android:id="@+id/nav_logout"
      android:icon="@drawable/ic_logout_variant_black_18dp"
      android:title="Logout"
      local:MvxBind="Click LogoutCommand" />
  </group>
```

Code Snippet 20. Implementation of the navigation drawer

5.3.5 Company's Social Media Pages

The implementation of the social media pages is demonstrated in the following code snippet. The WebViewFragment is used to load the page contents in WebView instead of browser.

```
<WebView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:id="@+id/LocalWebView" />
```

Code Snippet 21. Implementation of WebView

The following code snippet demonstrates, how to load the company's Twitter page in WebView.

```
WebView localWebView = view.FindViewById<WebView>(Resource.Id.LocalWebView);
    localWebView.Settings.JavaScriptEnabled = true;
    localWebView.LoadUrl("https://mobile.twitter.com/wapice");
    return view;
```

Code Snippet 22. Implementation of loading the Twitter page

The following code snippet demonstrates, how to load company's Facebook page in WebView.

```
WebView localWebView = view.FindViewById<WebView>(Resource.Id.LocalWebView);
    localWebView.Settings.JavaScriptEnabled = true;
```

```
localWebView.LoadUrl(”https://touch.facebook.com/wapiceltd/");
return view;
```

Code Snippet 23. Implementation of loading the Facebook page

The following code snippet demonstrates, how to load the company’s Linkedin page in WebView.

```
WebView localWebView = view.FindViewById<WebView>(Resource.Id.LocalWebView);
localWebView.Settings.JavaScriptEnabled = true;
localWebView.LoadUrl(”https://www.linkedin.com/company/wapice-ltd");
return view;
```

Code Snippet 24. Implementation of loading the Linkedin page

6 TESTING

Testing is a phase of software development that could be considered as a crucial point to success. It ensures the software quality and proves that the software is functioning as expected and as needed. In this section, an overall description of testing process will be provided.

Tests were carried out at the end of each main feature implementation and a final testing was done after the implementation of all features for Wapice News mobile application. Additionally, a final user acceptance testing was done before release and ensured that Wapice News mobile application met with the requirements.

The Android Emulator with different API levels, Samsung S6 and Sony Compact Z1 are used to test the Wapice News mobile application.

7 CONCLUSION

The main objective of this work was to develop a cross platform mobile application for Wapice employees which could help them to quickly access the company's internal news and social media pages. The Wapice News mobile application allows employees to read internal news and to view the company's social media pages on their mobile phones by providing their user credentials.

The main challenges in the development of this project were to develop a mobile application with Xamarin.Android, to learn how to use SharePoint REST API, to get familiar with MvvmCross. The development process has presented opportunities for gaining a lot of knowledge such as Xamarin.Android, MvvmCross and SharePoint REST Service. The needed technologies for Wapice News mobile application were studied before the implementation and continually done so throughout the implementation.

The application meets with all the requirement and the goals and the objectives were achieved. The Wapice News mobile application can be extended by adding new features based on company's needs.

8 REFERENCES

/1/ Wapice Ltd – Last access 12.05.2017

https://www.wapice.com

/2/ Products of Wapice Ltd – Last access 12.05.2017

https://www.wapice.com/en/products

/3/ Android Architecture – Last access 12.05.2017

https://www.tutorialspoint.com/android/android_architecture.htm

/4/ The Beginner's Guide to Android: Android Architecture – Last access 12.05.2017

https://www.edureka.co/blog/beginners-guide-android-architecture/

/5/ A tour of the C# Language – Last access 12.05.2017

https://docs.microsoft.com/en-us/dotnet/articles/csharp/tour-of-csharp/index

/6/ JSON – Last access 12.05.2017

https://en.wikipedia.org/wiki/JSON

/7/ JSON Syntax – Last access 12.05.2017

https://www.w3schools.com/js/js_json_syntax.asp

/8/ The MVVM Pattern – Last access 12.05.2017

https://msdn.microsoft.com/en-us/library/hh848246.aspx

/9/ MVVM Responsibilities – Last access 12.05.2017

https://www.tutorialspoint.com/mvvm/mvvm_responsibilities.htm

/10/ Getting Started with MvvmCross – Last access 12.05.2017

https://www.mvvmcross.com/documentation/getting-started/getting-started

/11/ Portable Class Libraries – Last access 12.05.2017

https://msdn.microsoft.com/en-us/library/gg597391(v=vs.100).aspx

/12/ Introduction to Portable Class Libraries – Last access 12.05.2017

https://developer.xamarin.com/guides/cross-platform/application_fundamentals/pcl/introduction_to_portable_class_libraries/

/13/ Apache Subversion – Last access 12.05.2017

https://en.wikipedia.org/wiki/Apache_Subversion

/14/ Xamarin – Last access 12.05.2017

https://en.wikipedia.org/wiki/Xamarin

/15/ Introduction to Xamarin Forms – Last access 12.05.2017

https://developer.xamarin.com/guides/xamarin-forms/getting-started/introduction-to-xamarin-forms/

/16/ Xamarin.Android Guides – Last access 12.05.2017

https://developer.xamarin.com/guides/android/

/17/ Getting Started with IOS – Last access 12.05.2017

https://developer.xamarin.com/guides/ios/getting_started/

/18/ SharePoint Overview – Last access 12.05.2017

https://www.tutorialspoint.com/sharepoint/sharepoint_overview.htm

/19/ Get to Know the SharePoint Rest Service – Last access 12.05.2017

https://dev.office.com/sharepoint/docs/apis/rest/get-to-know-the-sharepoint-rest-service

/20/ SharePoint 2013 – Understanding and Using the SharePoint 2013 REST Interface – Last access 12.05.2017

https://msdn.microsoft.com/en-us/magazine/dn198245.aspx

/21/ Implementing the MVVM Pattern – Last access 12.05.2017

https://msdn.microsoft.com/en-us/library/gg405484(v=pandp.40).aspx

/22/ Material Design with the Android Support Design Library – Last access 12.05.2017

https://blog.xamarin.com/add-beautiful-material-design-with-the-android-support-design-library/