GREENRIDERS

Windows Phone application

Martin Klátik

Bachelor's Thesis
May 2013

Degree Programme in Software Engineering
School of Technology

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Author(s)
KLÁTIK, Martin

Type of publication
Bachelor´s Thesis

Date
20.04.2013

Pages
36

Language
English

Permission for web publication
(X)

Title
GREENRIDERS- Windows Phone application

Degree Programme
Software Engineering

Tutor(s)
SALMIKANGAS, Esa

Assigned by
Bravioz Oy

Abstract

This thesis was written during the author's practical training in Bravioz Oy, a company which has created GreenRiders solution for carpooling (ride sharing) management. A part of the work in the company was working on the mobile version of this service.

The main objective of this project was to develop a well working basis of the Windows Phone application. The application has all basic functions of the Web version, such as creating, searching and joining of rides as well as many other features.

The thesis starts with theoretical introduction of the used technologies during developing. Information about existing Windows Phone platforms and also tools and technologies important for developing an application on this platform are discussed here. The thesis also contains an overview of GreenRiders service followed by the implementation of the application.

The result of the project is a fully functional application, entirely compliant with the placed basic requirements. The developed solution offers a great basis for other features as well as for implementing a new graphical design according to the specific requirements. During working on this project the author gained some experience of the developing process with co-operation not only with the company colleagues but also with employees of Nokia. After the implementation of the new graphical design the first version of the application will be ready for publishing on the market.

Keywords
Windows phone, Mobile development, Mobile client application

Miscellaneous

CONTENTS

FIGURES

ACRONYMS

API	Application programming interface
IDE	Integrated development environment
OS	Operating system
RAM	Random access memory
REST	Representational State Transfer
SDK	Software Developer Kit
UI	User Interface
WP	Windows phone
XAML	Extensible Application Markup Language

1 INTRODUCTION

Currently, the segment of mobile devices is still growing and it tends to continue growing for at least next few years. Mobile devices replace personal computers and software developers have to adapt to this trend. There are many mobile platforms that developers can choose and develop their products for. The most spread mobile operating systems are Android OS from Google, iOS from Apple and Windows Phone from Microsoft. Developers of mobile OS today also face the important decision: should the application be NATIVE or HTML5? Each way has some advantages and disadvantages. Among the advantages of HTML5 belong cross-platform compatibility and assumed saved costs from native application developed for each platform particularly. On the other hand, HTML5 is still not standardized and native applications have much better performance. Mark Zuckerberg, CEO of Facebook, has declared during an interview for TechCrunch (TechCrunch, 2012):

"The biggest mistake that we made as a company is betting too much on HTML5 as opposed to native because it just wasn't there,"

During the practical training in GreenRiders' company all possibilities of development were considered. The company has already done solutions for iOS, Symbian and for Web. The next platform that was decided to be developed on is Windows Phone. Because of existing mass of Windows Phone 7 users, this thesis project was to create a Windows Phone 7 application, which will be upgraded to Windows Phone 8 version with some extra features of this new generation of the operating system. The main goal of the thesis was to create Windows Phone application and introduce development for this mobile platform. The final version should contain all features of GreenRiders' web version.

2 WINDOWS PHONE OPERATING SYSTEMS

2.1 Windows Phone 7.5 Mango

Windows Phone 7 was released in November, 2010 in the United States. Later in May 2011, was released a new important version of Windows Phone 7, Mango (Windows Phone 7.5). Mango was major update with more than 500 changes. New version included new mobile Internet Explorer 9 with HTML5 support. The update also included multi-tasking, Windows Live SkyDrive access, LinkedIn and Twitter integration and better GPS navigation. (Lein, 2012)

Another minor updates were released in 2012 known as Windows Phone 7.5 Refresh and Tango. The updates brought better media messaging and lowered the hardware requirements to allow for less powerful devices to run Windows phone (new minimum is 800Mhz and 256MB RAM). (Rubino, 2012)

2.2 Windows Phone 7.8

An actual version of windows phone, Windows Phone 7.8 was released in January 2013. In past versions many people complained about WP start screen, so in the newest version is used the same layout of the start screen as in Windows Phone 8 operating system. The update also doubled number of color scheme options and updated lock screen and made it more customizable. This update was released, because Windows Phone 7 devices are not compatible with newer WP8 OS.

Windows Phone 7.8 features (Windows Phone 7.8, 2013)

- **Customizable Start screen**: Resize your Live Tiles—small, medium, or large. Large Tiles provide more info, small ones make it easy to tap and go.
- **20 accent colors:** The color you choose will show up on Start and all around your phone.
- **New lock screen:** New lock screen images from Bing service.

Notes

- The Windows Phone 7.8 update is not available in all markets or for all phones.
- Some features may work differently or not at all if the phone has 256 MB of RAM.
- Some features might not be available in all countries or regions.

FIGURE 1. Comparison WP7(Windows Phone 7, 2013) and WP8 start screen.

2.3 Windows phone 8

Windows Phone 8 was released on October 29, 2012. The version comes with whole new architecture. The older WP7 OS CE-based architecture is replaced with the Windows NT kernel found on many Windows 8 components. This change caused incompatibility between older and newer generation of Windows Phone devices. The current WP 7 devices cannot run or update to WP8. For comparison, there are start screens of WP7 and WP8 versions in Figure 1. On the other hand Windows Phone 8

comes with new Internet Explorer 10, multitasking, NFC and many other new features. The system has also higher hardware requirements. The minimum is 512MB RAM for WVGA phones, 1GB RAM for 720p/WXGA phones and 4GB of flash memory. Improvements got also camera and new *Room* concept of social space was added. (Rubino, 2012)

3 OVERVIEW OF TOOLS AND TECHNOLOGIES

3.1 Windows Phone Software Development Toolkit (SDK 8.0)

"The Windows Phone SDK 8.0 is a development environment to use for building apps and games for Windows Phone 8.0 and Windows Phone 7.5. The Windows Phone SDK provides a stand-alone Visual Studio Express 2012 edition for Windows Phone or works as an add-in to Visual Studio 2012 Professional, Premium or Ultimate editions. With the SDK, it is possible for developers to use their existing programming skills and code to build managed or native code apps. In addition, the SDK includes multiple emulators and additional tools for profiling and testing Windows Phone app under real-world conditions." (Windows Phone SDK 8.0, 2013)

3.2 C# and .NET Framework

C# is an object-oriented programming language developed by Microsoft as a part of .NET platform. Programs in C# run on the .NET Framework. For managing the executions of these programs is responsible Common Language Runtime - CLR. The CLR is the component created by Microsoft which provides type safety and memory management. C# code is compiled into an intermediate language- IL, so the code is stored on the disk in a file with extensions typically *.exe* or *.dll*. (msdn, 2013)

3.3 Silverlight

Silverlight brings advanced functions for multimedia and better interaction possibilities. It is a free software plugin based on the .NET framework. The plugin is compatible with many internet browsers and operating systems. Windows Phone 7 OS is compatible with Silverlight 3, but it also supports some of Silverlight 4's features and some other performance improvements in storyboard animations. (Microsoft Silverlight, 2013)

FIGURE 1. Silverlight scheme. (Microsoft Silverlight, 2013)

3.4 The Windows Phone Toolkit

Windows Phone Toolkit is set of components, controls and features for WP application. It brings new functionality for developers and better user experience for users. Windows phone toolkit is open source. The latest release date of the toolkit is October 30th 2012.

Components in WPToolkit (the underlined ones are used in this project) are listed as follows below: (WPToolkit, 2013)

- AutoCompleteBox
- ContextMenu
- CustomMessageBox
- DateTimeConverters
- DateTimePickers
- Effects – SlideInEffect, TiltEffect and TurnstileFeatherEffect
- ExpanderView
- HubTile
- ListPicker
- LongListMultiSelector
- Map extensions
- PhoneTextBox

- RatingControl
- ToggleSwitch
- Navigation transitions
- WrapPanel
- LongListSelector for 7.x
- MultiSelect for 7.x.

3.5 REST

“REST defines a set of architectural principles by which Web services can be designed that focus on a system's resources, including how resource states are addressed and transferred over HTTP by a wide range of clients written in different languages. If measured by the number of Web services that use it, REST has emerged in the last few years alone as a predominant Web service design model. In fact, REST has had such a large impact on the Web that it has mostly displaced SOAP (Simple Object Access Protocol) and WSDL-based (Web Services Description Language) interface design because it is a considerably simpler style to use.” (Rodriguez, 2008)

RESTful web API is a web API with principles of REST using HTML language. It has following four aspects: (Fielding, 2008)

- URI form for API (*http://server.com/services/)*
- API supports Internet media type of data. Usually is used JSON, but it can used be also other Internet media type.
- Operations use HTTP methods: GET, PUT, POST, or DELETE.

The rest of the advantages are listed below: (Cox, Harvey, Ramsbrock, 2011)

- Simplicity
- Infrastructure friendly
- Cacheable
- Scalable
- Stateless or Stateful

- Efficient

3.6 Json .NET

Json.NET project started in 2006. It is a very popular and useful JSON framework for .NET platform.

Benefits and Features

- Flexible JSON serializer for converting between .NET objects and JSON
- LINQ to JSON for manually reading and writing JSON
- High performance, faster than .NET's built-in JSON serializers
- Write indented, easy to read JSON
- Convert JSON to and from XML
- Supports .NET 2, .NET 3.5, .NET 4, .NET 4.5, Silverlight, Windows Phone and Windows 8 Store

The JSON serializer in Json.NET is a good choice when the JSON you are reading or writing maps closely to a .NET class. (Json .NET, 2013)

4 GREENRIDERS

4.1 Overview

*"*GreenRiders is an online and mobile ride sharing service free of charge. Service makes it easy to share any car rides and track carbon emissions - saving nature and money. GreenRiders is an open ecosystem that provides an effective, fun, and socially rewarding service that helps users decrease the amount of carbon emissions created through personal transportation. It is a unique online and mobile solution that makes it simple and easy to share car rides, such as private vehicle, taxi and rental car rides. GreenRiders can be seen as addition to the public transportation system. Solution offers ride management, creating rides and searching of existing rides in the system. "(GreenRiders, 2013)

GreenRiders' mission is to create effective solution for car sharing management. The company has two versions of GreenRiders. One is for public and it is for free to use. People get access to all features which are making car sharing easier and well organized. Paid version is for companies. The service is active in Finland and Slovakia. The company is also working on other international pilots.

GreenRiders for Business

GreenRiders offers for companies an easy way for sharing rides among company employees, and this way helps reduce emission and costs of alternative transportation.

- Clear business case
- Measurable carbon emission reduction
- Corporate-Grade security and robustness
- Safe and simple to use
- Online and mobile access
- Brand value

As can be seen in Figure 3, GreenRiders Corporate Edition comes with online support. In addition to it, training for help-desk personnel can be organized as per need bases. Tier II support to help-desk is also an option. (GreenRiders, 2013)

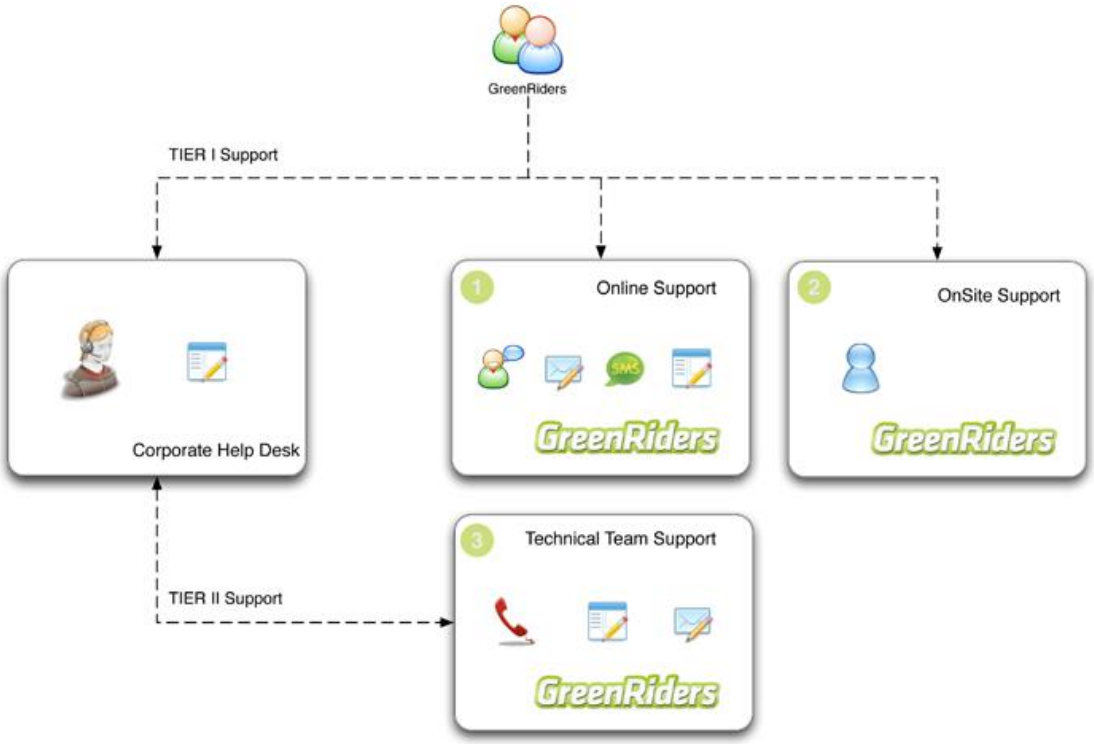

FIGURE 2. GreenRiders Corporate Edition online support. (GreenRiders, 2013)

4.2 GreenRiders API

GreenRiders is open eco-system designed to embrace innovation from developer community around the world. Developers are able to create independent apps utilizing large GreenRiders APIs. Good news is that there is no need for proprietary knowledge because the apps can be created utilizing standard web technologies.

Applications that already have been created utilizing GreenRiders APIs include a mobile application, Facebook apps, and further. GreenRiders Developer Program is currently tested and utilized by selected developer partners. (GreenRiders, 2013)

4.3 Existing solutions

Web version

The web version of GreenRiders is available on the GreenRiders website (http://www.greenriders.fi/en/login), so users can easily use this version via any web - browser. Users get access to all functions of the service after login. In Figure 4 is the Main page, where also basic layout of the application can be seen. As will be sees later the first version of Windows phone application has many common graphics elements with this version. It is like that because many users use both version and it is much easier to get oriented towards the app.

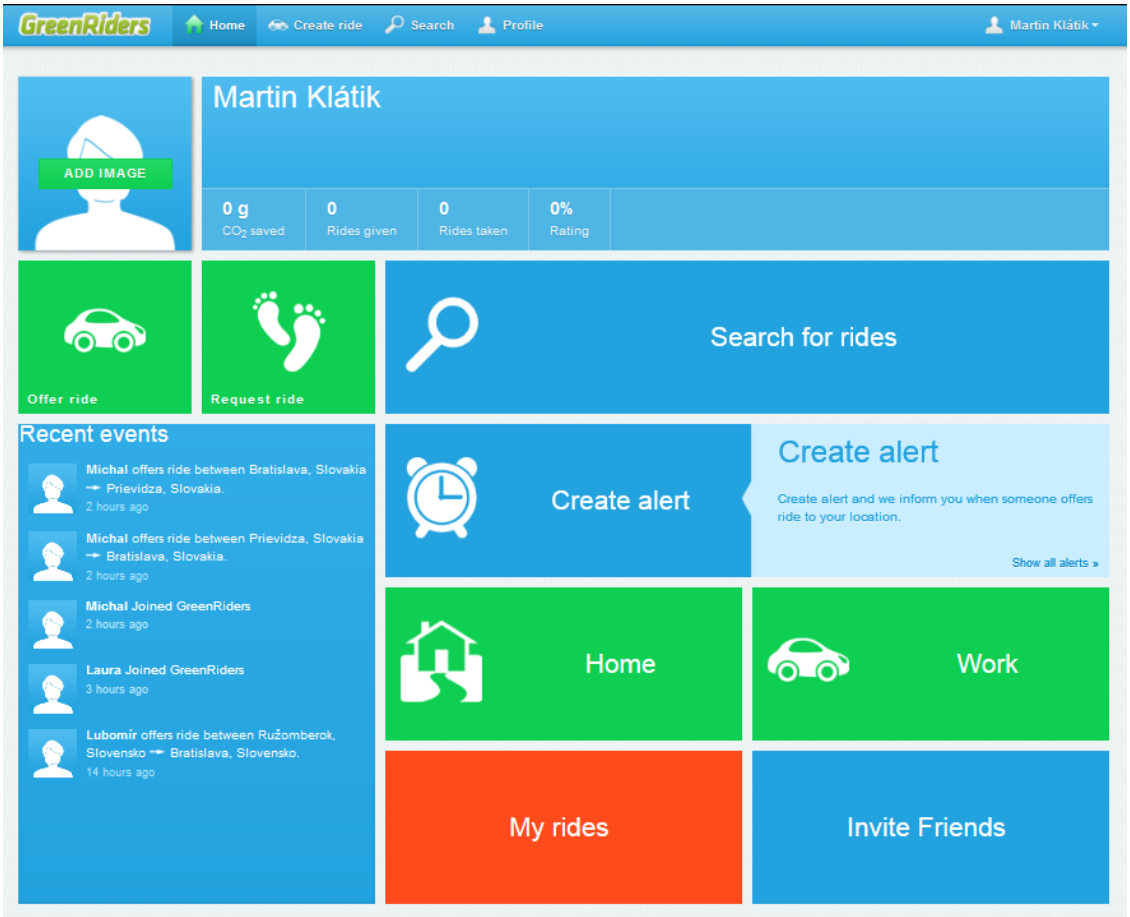

FIGURE 3. Web version screenshot. (Home page)

S60 5th edition (Symbian) version

The oldest mobile version is for Nokia mobiles with Symbian OS. It was developed in 2009. The application has only following basic functions: Create ride, Search ride, List of ride mates, List of my rides and Settings. There are screenshots of Home and Search page in Figure 5.

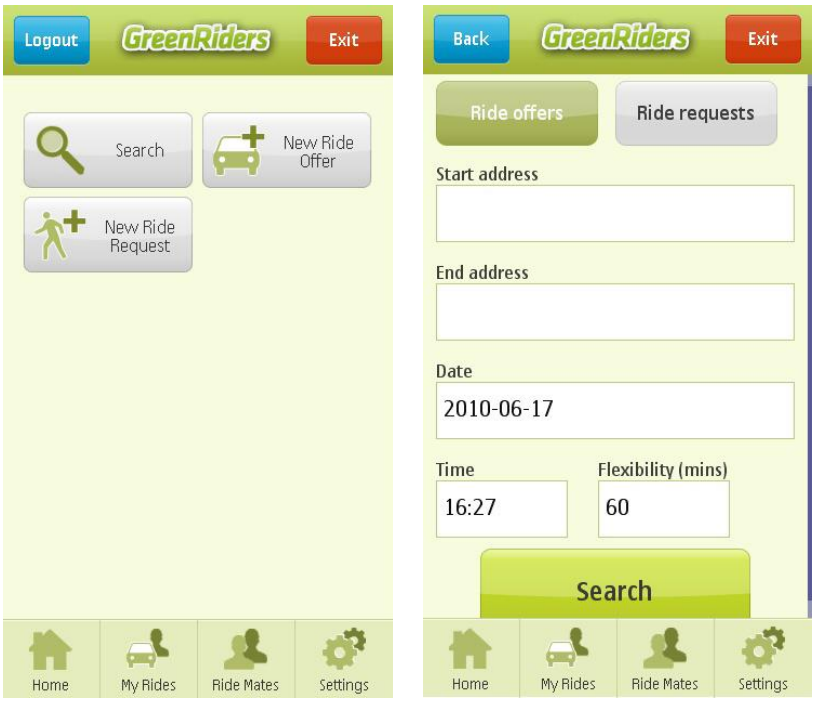

FIGURE 4. S60 app screenshot.(Home page and Search page)

IPhone version

The iPhone version was developed in 2009. Version is still on the market place. It has similar functions as previous Symbian version. Because of many new features in Web version and increased performance of new Apple devices a new update for this platform is planned. In Figure 6 are app screenshots of Home and Ride details page.

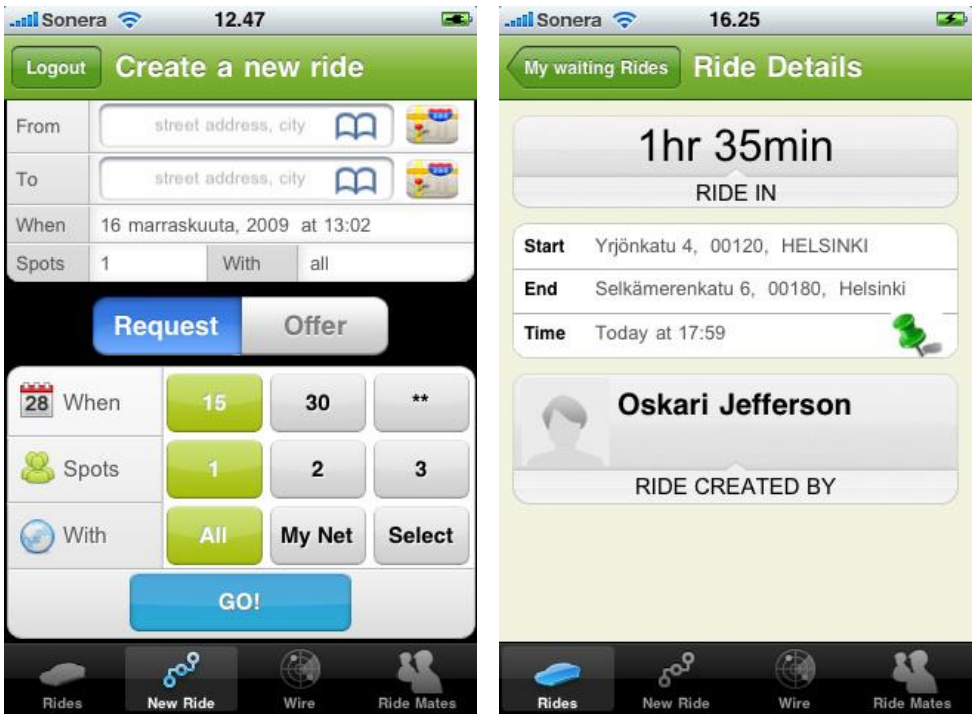

FIGURE 5. iPhone app screenshot. (Home page, Ride Details page)

5 REQUIREMENTS FOR WP APPLICATION

5.1 Functional Requirements

WP7 app should fulfill all basic functions of the existing web version. For the very first version of the app it is very important to create a functional basis prepared for addition of another special feature which offers a mobile platform or for addition of another features from the server side.

Basic functions

Here are listed the basic requirements for the application:

- **Show user profile:** profile page contains all public data, upcoming rides and comments of user. For private view of user also group rides (if the user belongs to the group).
    - **Public data of user:** profile picture, amount of taken and given rides, rating ratio
    - **Upcoming and group rides:** from and to address, seats left/asked, date of ride, creator name
    - **Comments:** creator of comment, date, rating (+/-), note
- **Edit settings:** users should be able to edit all his personal data, localization data, and also notification settings.
    - **Personal settings:** first and last name, date of birth, gender, email, phone number
    - **Localization settings:** street address, postal code, city, country, language
    - **Notification settings:** info about safety recalls, CO_2 saving tips, service info
- **Searching of rides:** searching of rides is one of the most important functions of the application. Searching includes a form for entry data, and results list. Searching is possible to filter according type of ride (offer/request/all). Searching function should be enabled also for not logged users.

    - **Entry data:** from and to address, from and to radius of input address, from and to date interval of searching rides
    - **Results list:** list of found rides (searched ride contains the same data like upcoming or group ride)
- **Show ride details:** ride details are useful when users want to check more information about a ride like additional ride notes, members and messages of crew
- **Join or cancel join of ride:** another very important functions. Users are able to join or cancel join of selected ride. They can also choose the number of seats they are taking
- **Create ride (repeat ride, group rides):** creating of ride contains form for the following entry data
    - **Entry data:** type of ride (offer/request), starting and ending destination, date and time of ride, number of seats, broadcasting group (all, users group), notes, repeat settings
    - **Repeat settings:** repeat (daily, weekly, weekdays), repeat every (number of days/weeks), repeat from and repeat to
- **Alerts (listing, creating, editing. deleting):** alert reminds when the ride in demand is created in the system. Users can create, edit, and remove alerts.
    - **Alerts data:** ride type, start and end address, start and end the date of the demanded ride
- **Locations (listing, creating, editing, deleting):** to save location is useful when a ride is searched for the same location very often. Users can create, edit, and remove locations.
    - **Locations data:** location name, address, place (home, work, other)
- **Rating ride mates:** after participation on some ride a user should rate other ride mates. After every ride, he/she can give positive or negative feedback with note, so another user can easily check ratings on the profile page in comments.

- **Recent events:** recent events show the last activity of GreenRiders users (new user or new ride notifications).

Extra features

- Map implementation: maps could be used for displaying entered locations and routes.
- Position locating: this feature is used in quick searching, where user's location is starting destination and entered location is ending one.
- Change profile picture: part of user setting should be possibility to change profile picture and upload it on the server.

5.2 Non-Functional Requirements

The main UI requirement for this application is sticking to the Windows Phone UI Guidelines. In the first version of the application this requirement is not so strict. The second version will have to pass guideline control from a professional Windows phone graphical designer. In the process of creating the application are also involved Jussi Paanajarvi and Juha-Lasse Latikka (employees of Nokia), who help with many ideas about the realization, and give advice how to make the application more attractive for users.

The second important non-functional requirement concerns the app performance. Every web service in the app has to run asynchronously to not block the UI thread. Also, pre-loading of data is needed for better user experience with navigation through the app.

6 IMPLEMENTATION

6.1 Used technologies

The GreenRiders application is made for Windows phone 7.5 (Mango) in Visual studio Express 2012. The source code is written in C# language and XAML. This client application uses GreenRiders' REST architecture and JSON API. For deserializing JSON objects used is Json.NET framework. For better user experience, there are also control elements from Windows phone toolkit in use.

6.2 UI design (layout, pages, controls)

The main page

The first screen after the first launch of the application shows the page with pivot control with 2 pivot items. (See Figure 5.)

- **Login pivot:** on the first one is standard login form. Login data (email and password field) are saved to the isolated storage, so application can remember login data also on the next launch. There are also *save password check box* and *register link* which opens internet browser for registration on the GreenRiders website.
- **Search pivot:** the second pivot item "search" includes form for searching rides. The reason to place searching feature here is that many users want to search ride without registration or login and in case they will find an interesting one, they need to login to proceed to ride details.
- **Results pivot (dynamic)**: the Main page also includes a page with a results list which shows with relevant results after searching.

FIGURE 7. Main page.

Apollo page

This page uses panorama control and consists of three panorama items (see Figure 17).

- **Home**: There are hubTiles on the first item of this pivot page. HubTile is a Windows phone toolkit control which enables to add animated and informative tiles to the application. A HubTile can have image, title, message and notification. These tiles are basic navigation controls of the app. There is also a rectangle which shows user data directly on the page and navigates to user profile page.
- **My location rides**: this panorama item contains a quick search of rides from/to my locations. All locations are loaded onto the listpicker, and after change the selection it will automatically start to search with new locations data. The parameters are set up to following values: distanceTo: 20, distanceFrom: 20. After finished loading, there is a scrollable list of found rides.
- **Recent events:** - The third panorama item contains a recent event list. This scrollable list shows the last activity of user.

Profile page

- **Profile pivot**: Every user in GreenRiders has their own profile. To show this data a profile page is created. The profile page shows user information. (See Figure 8.)
- **Upcoming pivot**: More important information gains upcoming ride pivot item, and group pivot item, which shows all user future rides. The group pivot item is dynamic. That means that it is shown only when the user belongs to some group.
- **Group pivot:** Group rides shows which rides are offered/requested in the specific group
- **Comments pivot:** Here are shown all user ratings and comments from past rides

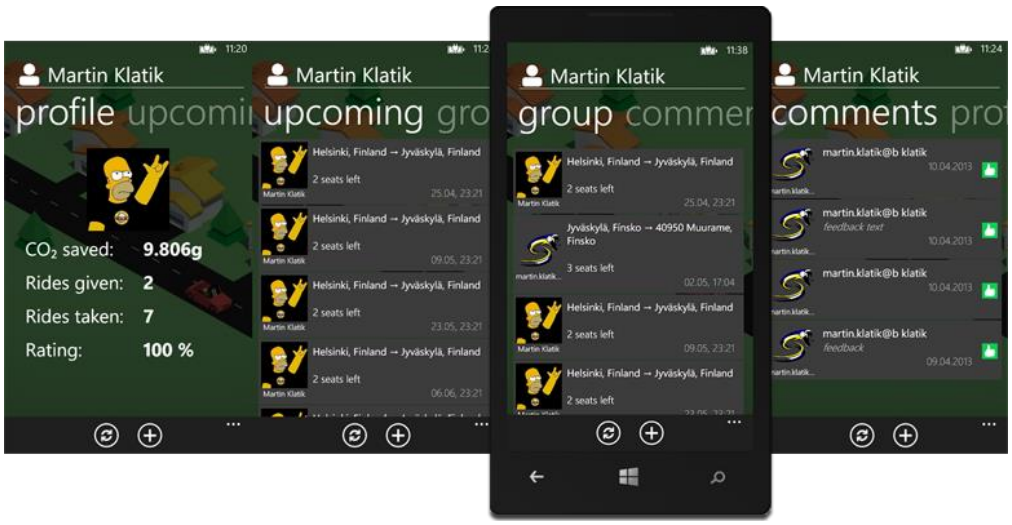

FIGURE 6. Profile page.

Ride detail page

When a user wants to view more information about a ride, s/he taps on a ride, and s/he will be navigated to a detail page of the selected ride. (See Figure 9.)

- **Details pivot**: This pivot page shows all necessary information about the selected ride (from to destinations, time, note…), and it also allows users to join/cancel ride.
- **Messages pivot**: The second pivot item of this page contains a list of messages within the form for adding new ones.
- **Crew pivot:** The third pivot item shows the actual crew of a selected ride.

FIGURE 7. Ride detail page.

Create ride page

New ride page consists of following three pivot items. (See Figure 10.)

- **Create ride pivot:** form for creating a new ride. The form contains all input elements for data mentioned in functional requirements of creating ride function. There is also a button for enabling repeat pivot page.
- **Repeat pivot:** repeat form for setting a repeat ride.
- **Map pivot:** consists of a map view, which shows the physical location of the chosen start and end destination in create ride form.

Figure 8. Create ride page.

Search ride page

- **Search pivot:** Pivot item “search” includes a form for searching rides. This pivot is analogy to search pivot in Main page.
- **Results pivot (dynamic)**: Search ride page also includes page with a results list, which is shown when *download event* is completed with some relevant results.

Alerts pages

- **Alerts list page** contains a list of alerts and an application bar for basic operations (adding, deleting, editing, refresh)
- **New alert page** is a form for creating alerts. The form contains all input elements for data mentioned in functional requirements of alerts function.
- **Edit alert page** is a form for editing alerts. The form contains all input elements for data mentioned in functional requirements of alerts function.

Locations pages

- **Locations list page** contains a list of alerts and an application bar for basic operations (adding, deleting, editing, refresh)
- **New location page** form for creating locations. The form contains all input elements for data mentioned in functional requirements of location function.
- **Edit location page** form for editing locations. The form contains all input elements for data mentioned in functional requirements of locations function.

Home page

This pivot page has 2 pivot items: *fromHome* and *toHome*. Both pivot items have the same content- shortcuts for offering, requesting and searching rides from or to selected home location. It also contains a link for navigation to the location page.

Work page

This page is analogy of Home page, except for using selected work location.

Rating page

Rating page is shown immediately after login to a user who has had some ride recently.

Settings Page

Settings page has 3 pivot items, which contain forms for changing all user data. (See Figure 11.)

Pivot items in Settings page:

- Profile
- Localization
- Notification

FIGURE 9. Settings page.

6.3 Functions of application (data handling)

Create ride

As can be seen in Figure 12, the creation process of ride begins with checking all necessary fields in the form. When all fields are filled in a correct way a *WebClient* instance is created and login credentials are set to the instance. After that all input data are transformed to the output string, which is in JSON format. For calling a method after completed creating event, *UploadStringCompletedEventHandler* is set to the web instance. The last step is calling WebClient's *UploadStringAsych* method with address

and output parameters. This method sends ride data to the server to process the web service.

```
//create new ride button pressed
private void AddRideButton_Click(object sender, EventArgs e)
{
    //set focus back to the page
    PivotItem pivot = (PivotItem)createRidePivot.SelectedItem;
    pivot.Focus();

    //check input fields
    if (checkFields()) return;
    ShowProgress = true;
    // create a WebClient instance
    WebClient webClient = new WebClient();
    // binds a method to the completed event
    web.UploadStringCompleted += new UploadStringCompletedEventHandler(webClient_AddRideCompleted);
    web.Credentials = app.credentials;

    try
    {
        //processing input fields
        //...
        output = "json format of entered data";
        web.UploadStringAsync(new Uri("...webservice address..." + output + "&junk=" + DateTime.Now), "POST");
    }
    catch
    {
        MessageBox.Show("something is wrong cannot create new ride:(");
    }
}
```

FIGURE 10. Create ride method.

When the string is successfully uploaded to the address, *NewRideCompleted* method is called. This method handles refreshing of lists of rides and navigation back to the upcoming ride list.

Search ride

Sending search ride data is similar to creating of ride. The difference is in handling of data which come from server within *SearchRideCompleted* event. These data are deserialized with Newtonsoft *JsonConvert* methods to .Net *Ride* object. After that object's data are bound with *listsBox* in Result page which shows the results of searching. (see Figure 13.) The results of searching are shown as items of *listBox*. It is possible to view detail page of a ride on single tapping on an item.

```
private void webClient_SearchRideCompleted(object sender, UploadStringCompletedEventArgs e)
{
   //clearing last search results
    //...

    //if we got some results, deserialize them to Rides object
    if (e.Error == null && e.Result != null)
    {
        var deserialized = JsonConvert.DeserializeObject<Rides>(e.Result);
        app.searchedRides = deserialized;

        //putting app.searched ride results to different listboxes according
        //entered searching parameters(searh FromTo, search From, search To)
        //...

        ShowProgress = false;

    }
    //show header of Results pivot item and navigate to the pivot
    resultsPage.Header = "results";
    searchRidePivot.SelectedIndex = 1;

}
```

FIGURE 11. Deserializing search result.

Ratings

Each user of a taken ride is rated by other ride mates. The rating page is open after every login, when a user has some waiting rating. *User* can be rated positively or negatively. Users can also send a comment with their rating. When the user was a passenger, from her/his side can be rated only driver of the ride. For statistics purposes users also send alternative transport to their ride. There are six different alternatives. After confirmation of rating, all data are sent via request to the server.

Obtain location data

Very useful and important part of creation, editing and searching feature is gaining location data from an input address. Because right coordinates and full address name are needed for these features, Bing geocoding service is used for gaining the data. For better user experience input address *textBoxes* are autocompleted with suggestions of a written address. These suggestions come from geocoding results.

```
// start destination autoCompleteBox textChanged event method
private void acBox_start_TextChanged(object sender, RoutedEventArgs e)
{
    try
    {
        //clear previous results
        addresses.Clear();
        acBox_start.ItemsSource = null;
        //new geocode request
        GeocodeRequest geocodeRequest = new GeocodeRequest();
        // Set the credentials
        geocodeRequest.Credentials = new Credentials();
        geocodeRequest.Credentials.ApplicationId = ApplicationId;
        // Set the full address query
        if (acBox_start.Text.Equals("")) geocodeRequest.Query = "a";
        else geocodeRequest.Query = acBox_start.Text;

        FilterBase[] filters = new FilterBase[1];
        filters[0] = new ConfidenceFilter() { MinimumConfidence = Confidence.High };
        //geocoding options
        GeocodeOptions geocodeOptions = new GeocodeOptions();
        geocodeOptions.Filters = new ObservableCollection<FilterBase>(filters);
        geocodeRequest.Options = geocodeOptions;
        //asynchrounous request
        geocodeService.GeocodeAsync(geocodeRequest);

    }
    catch { return; }
}
```

FIGURE 12. Start destination autoCompleteBox text changed event method.

The principal is based on listening of *textChanged* event of *autoCompleteBox* element. Each change of the element text field sends new geocode request with updated query (see Figure 14). When results of this request come, *autoCompleteBox* is updated and populated with new suggestions of locations (see Figure 15).

```
//geocode asynch request completed
private void geocodeService_GeocodeCompleted(object sender, GeocodeCompletedEventArgs e)
{

    // The result is a GeocodeResponse object
    GeocodeResponse geocodeResponse = e.Result;
    if (geocodeResponse.Results.Count > 0)
    {
        //adding all address suggestions to List<GeocodeResult> addresses
        for (int i = 0; i < geocodeResponse.Results.Count; i++)
        {
            if (geocodeResponse.Results[i].Locations.Count > 0)
            {
                addresses.Add(geocodeResponse.Results[i]);
            }
        }
        //put addresses to autoCompleteBox item source
        this.acBox_start.ItemsSource = addresses;
        this.acBox_end.ItemsSource = addresses;

        //updating and populating autoCompleteBox with new suggestions
        this.acBox_start.UpdateLayout();
        this.acBox_end.UpdateLayout();
        this.acBox_start.PopulateComplete();
        this.acBox_end.PopulateComplete();
    }
}
```

FIGURE 13. Geocode request completed.

The final step for choosing of a location from suggestions is setting the location to *startDestination* variable. (see Figure 16.) After that the location object is ready to give all necessary data to the request output string. The start destination and end destination variables are also used in a distance calculation. When users want to create a new ride, the application also sends *distance* parameter, which is used for statistics purposes. The distance is calculated with Bing route service.

```
//selecting destinations
private void acBox_start_SelectionChanged(object sender, RoutedEventArgs e)
{
    startDestination = (GeocodeResult)acBox_start.SelectedItem;
}
```

FIGURE 14. Setting selected suggestion to startDestination variable.

6.4 Bing maps implementation

Maps are not a necessary, however, a very useful feature in the application. In the first version of the application a map view in create ride page is implemented. It shows the actual chosen location in *autoCompleteBox*, so a user is able to check if the chosen location corresponds with the location s/he wanted to enter. In the Windows phone 7 version of the app Bing maps are used, but in later Windows phone 8 version of the app they will be implemented new HERE maps API with more features.

6.5 Performance optimization

The performance of the application mainly depends on the server response. For better user experience with application are web services ordered by priority or skipped for faster loading of more important information. All web requests are asynchronous, so they don't block UI thread. Also for faster viewing of user personal data, there are created different variables for personal and for other user's data.

Here is an example how are requests sent for faster navigation to *Apollo page*. After pressing login button and following authentication, the application sends web request to *get user data*. When user data are downloaded, another 3 requests are sent: get *location, alerts* and *ratings*. In this state the app listening for when are ratings downloaded. After that last requests are sent (*get user's upcoming rides, group rides, profile image*) and *Apollo page* is shown. Web request threads of locations, alerts, upcoming rides, group rides, and profile image may still run in the background. However user is able to navigate through the application UI, and doesn't have to wait for loading of all data.

7 VERSION COMPARISONS

As was mentioned before, this thesis describes process of creation of the first version of application. This version has got basic design, and it does not stick Windows Phone guidelines, what is important requirement for the application. There wasn't available professional design during working on this version, so it might be interesting to compare what differences there will be in second version. Figure 17 shows the actual version of the application, in the figure 18 illustrates the new design.

FIGURE 15. Actual design of Main page.

FIGURE 16. New design of Main page.

As can be seen, the changes in versions are not only graphical, but there are also some changes in layout and in the structure of the application. In the new design is also counted with new features such as quick search, which use position locating, or with new leaf rewarding system and GR points.

8 CONCLUSION

This bachelor's thesis was created to provide information about developing GreenRiders Windows Phone application. The thesis informs about needed tools to start development on WP platform, describes GreenRiders service and shows overview of existing GreenRiders solutions on different platforms. There is also described which implementation way of the application I chose and which requirements were possible and which were not possible to realize.

The work on the application still continues, and with cooperation with professional designer the application will be ready for final designer review. In the new version of the app many new features are planned, but first of all the stable basic version is planned to be published immediately after implementing the new design.

During my practical training in Bravioz company, I gained knowledge about mobile client developing and teamwork skills. I got experience in development process such as planning, communication and schedule management. My Windows Phone developing skills which I gained on Windows Phone programming course were emphasized by my self-study and cooperation with my colleagues.

References

Adam Z Lein, 2012. The Ultimate Windows Phone 7.5 Mango Preview. Accessed on 7 May 2013. http://pocketnow.com/windows-phone/the-ultimate-windows-phone-7-mango-preview

Daniel Rubino, 2012. Some new undocumented features of Windows Phone Tango. http://www.wpcentral.com/some-new-undocumented-features-windows-phone-tango

File-Extensions, 2013. Windows Phone 7. Accessed on 7 May 2013. http://www.file-extensions.org/windows-phone-7-file-extensions

TechCrunch, 2012. Mark Zuckerberg: Our Biggest Mistake Was Betting Too Much On HTML5. Accessed on 3 May 2013. http://techcrunch.com/2012/09/11/mark-zuckerberg-our-biggest-mistake-with-mobile-was-betting-too-much-on-html5/

Windows Phone, 2013. Windows Phone 7.8. Accessed on 3 May 2013. http://www.windowsphone.com/en-US/how-to/wp7/start/whats-new-in-windows-phone

Daniel Rubino, 2012. Overview and review of Windows Phone 8. Accessed on 7 May 2013. http://www.wpcentral.com/overview-and-review-windows-phone-8

Windows Phone Silverlight. Accessed on 3 May 2013. http://www.microsoft.com/silverlight/windows-phone/

MSDN, 2013. Introduction to the C# Language and the .NET Framework. Accessed on 3 May 2013. http://msdn.microsoft.com/en-us/library/z1zx9t92.aspx

The Windows Phone Toolkit, 2013. Accessed on 3 May 2013. http://phone.codeplex.com/

Alex Rodriguez, 2008. RESTful Web services: The basics. Accessed on 3 May 2013. http://www.ibm.com/developerworks/webservices/library/ws-restful/

Jack Cox, Doug Harvey and Daniel Ramsbrock, 2011. SOAP vs. REST for Mobile Services. Accessed on 3 May 2013. http://blogs.captechconsulting.com/blog/jack-cox/soap-vs-rest-mobile-services

Roy T. Fielding, 2008. REST APIs must be hypertext-driven. Accessed on 3 May 2013. http://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven

Json .NET, 2013. Accessed on 3 May 2013. http://json.codeplex.com/

GreenRiders, 2013. Accessed on 3 May 2013. http://www.greenriders.fi/en