|  |  |  |
| --- | --- | --- |
| http://www.hcmus.edu.vn/images/stories/logo-khtn2009_transparency.png | **CS426- Mobile Device Application Development** |  |

Class: APCS 2012

Date: June 28-30, 2014 (48 hours)

**Mini Project 1**

**Personal Report**

**Student ID: 1251040**

**Student’s full name: Võ Minh Thành**

**Email: vmthanh@apcs.vn**

**Tel: 0909179583**

# Project Self Assessment

In this section, each student should personally evaluate his or her own effort to complete the project.

* Please check [x] in ***column C*** if you partially or completely finish this feature. This means your application has this feature.
* If you check [x] in column C, you should self-evaluate the level of completeness of this feature in ***column D***.

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Max pts** | **Have this feature?** | **Level of completeness**  **(100%)** |
| (A) | (B) | (C) | (D) |
| * ***Note management*** | | | |
| * + **Create a new note** |  |  |  |
| * + - Each note includes a short title and a full description. | 50 | X | 100 |
| * + - One or multiple tags (keywords) can be assigned to a note. A note can have no tags. | 25 | X | 50 |
| * + - Each note belongs to one category. Each category has its own representative icon. | 25 | X | 100 |
| * + - Each note can be linked with a specific geographical location. | 50 | X | 100 |
| * + - Each note may contains one or several photos: |  |  |  |
| * + - * Photos can be selected from the media library or | 50 | X | 90 |
| * + - * Photo can be taken from the camera. | 50 | X | 90 |
| * + - Each note may be associated with a voice note. | 50 | X | 100 |
| * + - Reminder: each note may be associated with several reminder schemes. |  |  |  |
| * + - * Time-based reminder | 50 | X | 100 |
| * + - * Location-based reminder | 50 | X | 100 |
| * + **Seach for notes:** A user can search for notes or see the list of all notes. | 50 | X | 100 |
| * + **Update a note:** A user can update a note | 50 | X | 100 |
| * + **Delete a note:** A user can delete a note. | 50 | X | 100 |
| **Feature** | **Max pts** | **Have this feature?** | **Level of completeness**  **(100%)** |
| (A) | (B) | (C) | (D) |
| * ***Note visualization:*** | | | |
| * + **Note visualization on a map** |  |  |  |
| * + - The list of notes can be visualized on the map | 50 | X | 100 |
| * + - The user can select which category of notes for visualization. | 50 | X | 100 |
| * + - Each note is represented by an icon corresponding to its category. | 50 | X | 100 |
| * + - When a user tap on an icon, the detail information of the corresponding note should be display (and the user can then update or delete it). | 50 | X | 100 |
| * + **Find and display the best path to the location of a note:** |  |  |  |
| * + - Find and display the best path from the current position to the location of a note. | 50 | X | 100 |
| * + - Maps can be downloaded for offline display | 50 | X | 100 |

# Advanced Features

Please list in details all ***advanced features*** in your project.

***Screenshots*** are suitable to illustrate these advanced features.

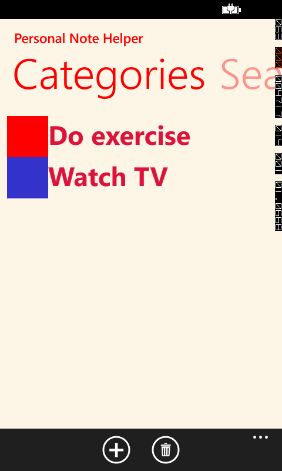
………

# Screenshots

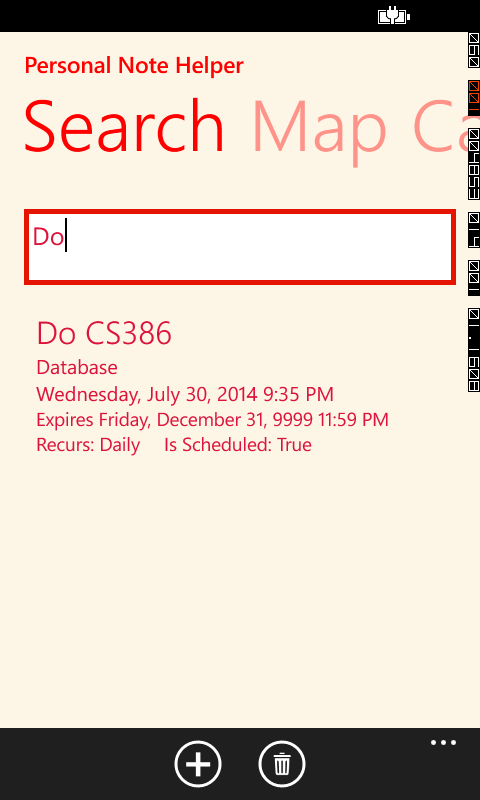
Please insert ***screenshots*** in this section to illustrate ***all main features*** of your project.

Several screenshots and features ***can be the same*** as in Section II. Advanced Features.

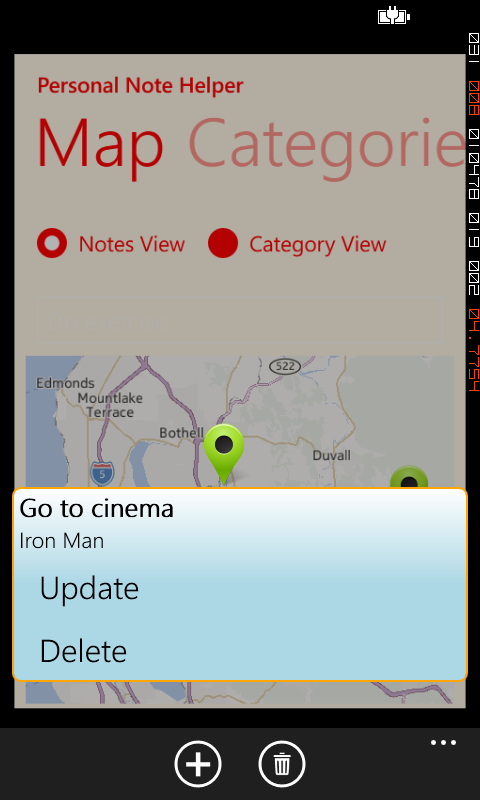
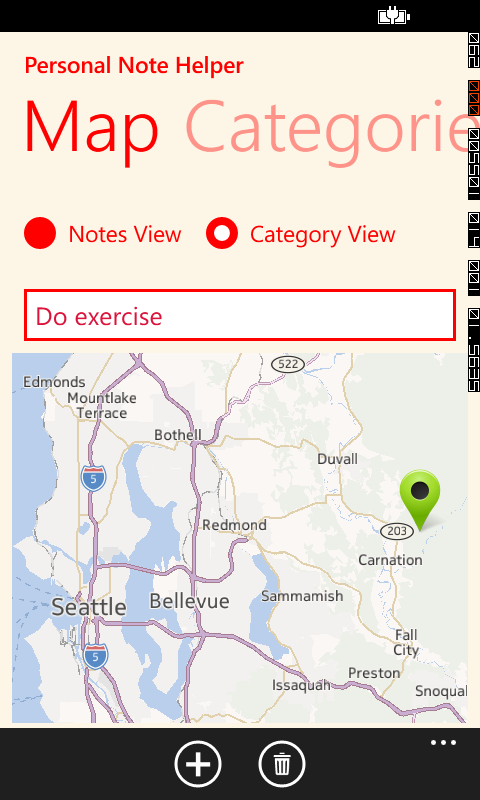
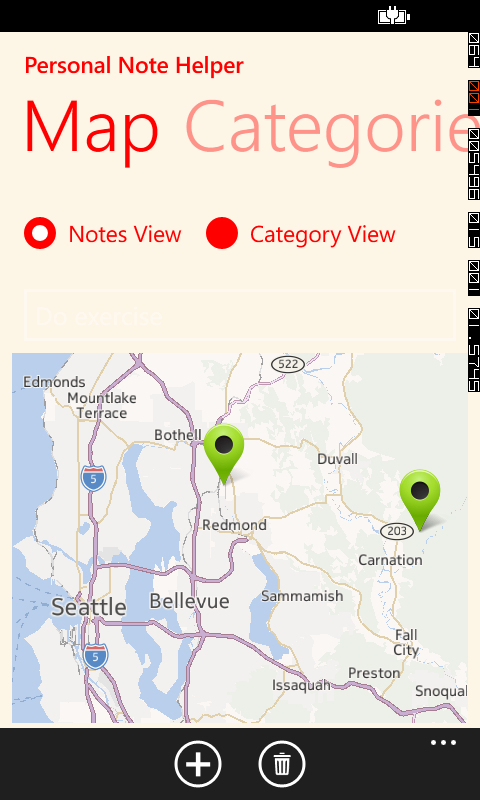
* Categories of Note:



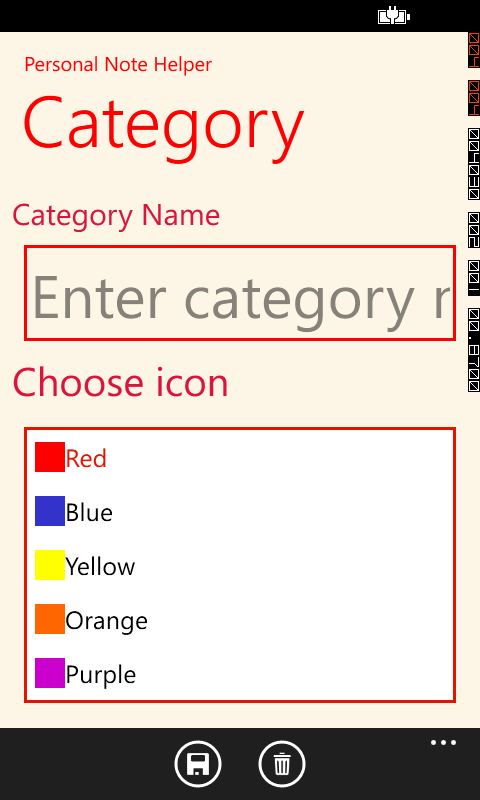
* Search note:



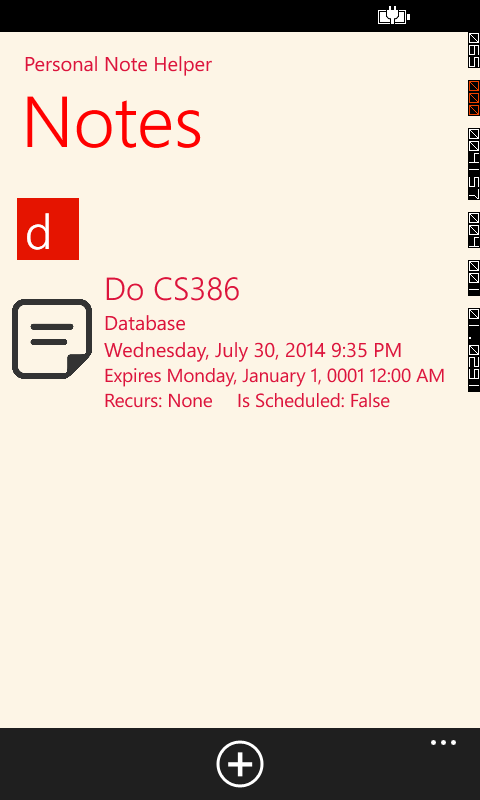
* Note visualization:

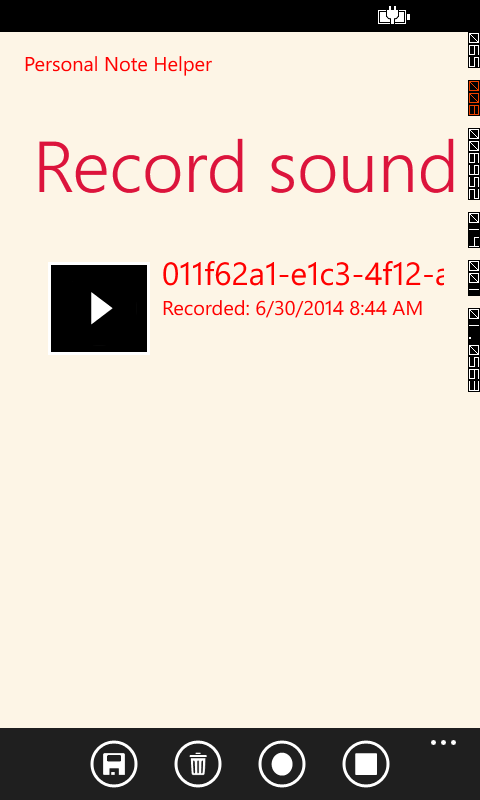
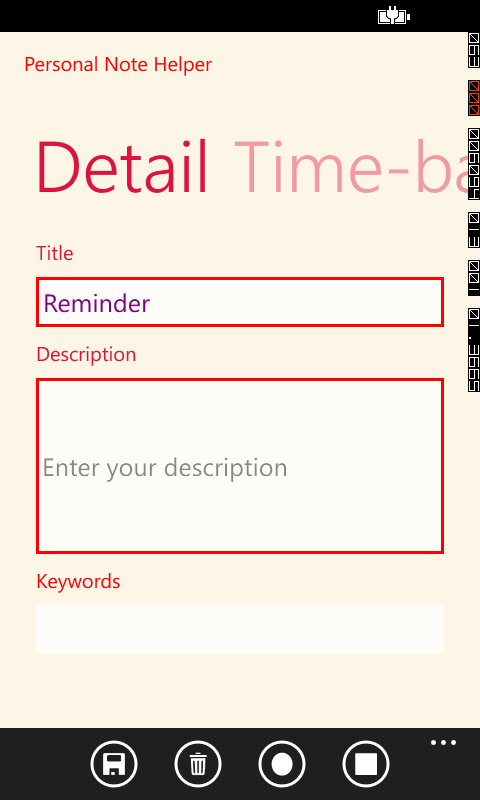
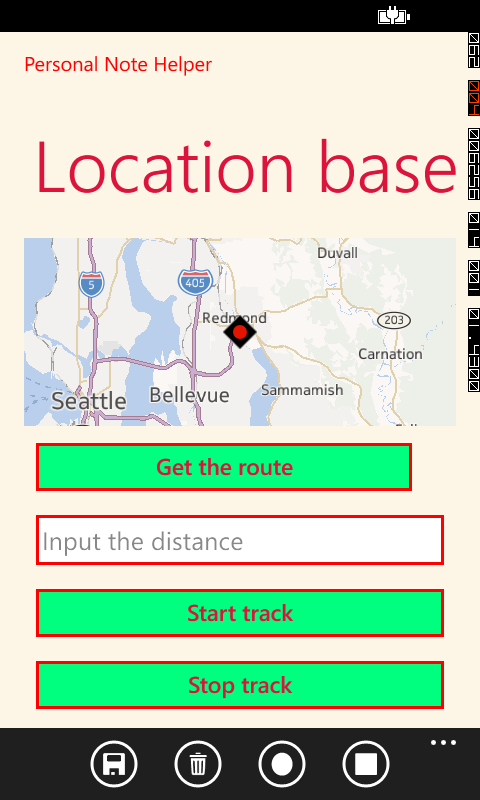


* Create Note:



* Note page:



* **Edit note:** 

# References

You should list all source codes and/or references that you use in your project in this section.

Any code fragments or libraries that are not yours ***MUST be explicitly declared*** in this section.

If you fail/forget to declare those “inherited” resources, you will be considered “cheating”!

1/ Windows Phone Toolkit:

2/ Making Custom PushPin With Custom Tooltip:

<http://code.msdn.microsoft.com/wpapps/Windows-Phone-8-Maps-API-95ef7816>

3/ Record Voice in Windows Phone:

**Reference**: Windows Phone 8 in Action. Timothy Binkley-Jones, Adam Benoit, Massimo Perga, Michael Sync. Chapter 9, section 9.3: Playing and recording with the Music + Video Hubs.

* XML Interface:

<phone:PivotItem Header="Record sound">

<phone:LongListSelector x:Name="recordingList">

<phone:LongListSelector.ItemTemplate>

<DataTemplate>

<StackPanel Orientation="Horizontal"

Margin="12">

<Button Click="play\_Click"

Tag="{Binding Title}"

Background="Black">

<Image Source="/Assets/AppBar/transport.play.png" />

</Button>

<StackPanel Orientation="Vertical">

<TextBlock Text="{Binding Title}"

FontSize="{StaticResource PhoneFontSizeLarge}" Foreground="Red" />

<TextBlock Text="{Binding Date, StringFormat='Recorded: \{0:g\}'}"

FontSize="{StaticResource PhoneFontSizeNormal}" Foreground="Red"/>

</StackPanel>

</StackPanel>

</DataTemplate>

</phone:LongListSelector.ItemTemplate>

</phone:LongListSelector>

</phone:PivotItem>

* Get data from the microphone and write the data to the Memory Stream:

private void microphone\_BufferReady(object sender, EventArgs e)

{

int count = Microphone.Default.GetData(audioBuffer);

audioStream.Write(audioBuffer, 0, count);

}

* Record microphone:

private void ApplicationBarIconButton\_Click\_2(object sender, EventArgs e)

{

if (Microphone.Default.State == MicrophoneState.Stopped)

{

try

{

recordingList.IsEnabled = false;

audioStream = new MemoryStream();

audioBuffer = new byte[Microphone.Default.GetSampleSizeInBytes(TimeSpan.FromSeconds(1))];

Microphone.Default.Start();

}

catch (Exception)

{

MessageBox.Show("Sorry. Sometimes shit happens", "Error", MessageBoxButton.OK);

}

}

}

* Stop the recording:

private async void ApplicationBarIconButton\_Click\_3(object sender, EventArgs e)

{

if (Microphone.Default.State == MicrophoneState.Started)

{

try

{

Microphone.Default.Stop();

string filename = await WriteFile();

audioBuffer = null;

audioStream = null;

recordingList.IsEnabled = true;

recordingList.ItemsSource.Add(new VoiceRecording { Title = filename, Date = DateTime.Now });

}

catch(Exception )

{

MessageBox.Show("Sorry. Sometimes shit happens", "Error", MessageBoxButton.OK);

}

}

}

* Saving audio data to isolated storage:

private async Task<string> WriteFile()

{

StorageFolder localFolder = ApplicationData.Current.LocalFolder;

string FileName = reminder.Name + ".wav";

StorageFile file = await localFolder.CreateFileAsync(

FileName,

CreationCollisionOption.GenerateUniqueName);

using (Stream fileStream = await file.OpenStreamForWriteAsync())

{

using (var writer = new BinaryWriter(fileStream))

{

writer.Write(new char[4] { 'R', 'I', 'F', 'F' });

writer.Write((Int32)(36 + audioStream.Length));

writer.Write(new char[4] { 'W', 'A', 'V', 'E' });

writer.Write(new char[4] { 'f', 'm', 't', ' ' });

writer.Write((Int32)16);

writer.Write((UInt16)1);

writer.Write((UInt16)1);

writer.Write((UInt32)16000);

writer.Write((UInt32)32000);

writer.Write((UInt16)2);

writer.Write((UInt16)16);

writer.Write(new char[4] { 'd', 'a', 't', 'a' });

writer.Write((Int32)audioStream.Length);

writer.Write(audioStream.GetBuffer(), 0,

(int)audioStream.Length);

writer.Flush();

}

}

return file.Name;

}

4/ Scheduled actions:

**Reference**: Windows Phone 8 in Action. Timothy Binkley-Jones, Adam Benoit, Massimo Perga, Michael Sync. Chapter 4: Scheduled Actions:

* Find the reminder in the list of ScheduledActionService:

var notifications = ScheduledActionService.GetActions<ScheduledNotification>()

.OrderBy((item) => item.BeginTime);

foreach (ScheduledNotification notification in notifications)

{

ScheduledAction item =

ScheduledActionService.Find(notification.Name);

items.Add(item);

}

notificationList.ItemsSource = items;

* Adding a reminder to ScheduledActionService

if (ScheduledActionService.Find(reminder.Name) == null)

{

ScheduledActionService.Add(reminder);

isSaveClick = true;

}

else

{

ScheduledActionService.Replace(reminder);

isReplace = true;

}

* Delete a reminder

if (ScheduledActionService.Find(reminder.Name) != null)

ScheduledActionService.Remove(reminder.Name);

5/ Map and Location:

* Formatting longitude and latitude
  + Purpose: Accept angle measurement in degrees, along with the characters to display for positive and negative values, and returns a string displaying direction, degrees, minutes, and seconds

private string FormatCoordinate(double coordinate, char positive, char negative)

{

char direction = coordinate >= 0 ? positive : negative;

coordinate = Math.Abs(coordinate);

double degrees = Math.Floor(coordinate);

double minutes = Math.Floor((coordinate - degrees) \* 60.0D);

double seconds = (((coordinate - degrees) \* 60.0D) - minutes) \* 60.0D;

string result = string.Format("{0}{1:F0}° {2:F0}' {3:F1}\"",

direction, degrees, minutes, seconds);

return result;

}

* FormatAddress:
  + Purpose: accepts a MapAddress parameter and return the address in string

private string FormatAddress(MapAddress address)

{

StringBuilder b = new StringBuilder();

if (!string.IsNullOrWhiteSpace(address.HouseNumber))

b.AppendFormat("{0} ", address.HouseNumber);

if (!string.IsNullOrWhiteSpace(address.Street))

b.AppendFormat("{0}\n", address.Street);

if (!string.IsNullOrWhiteSpace(address.City))

b.AppendFormat("{0}, ", address.City);

b.AppendFormat("{0} {1}", address.State, address.PostalCode);

return b.ToString();

}

* Guide to the Windows Phone 8 Maps API:

Reference: <http://developer.nokia.com/resources/library/Lumia/change-history/archived-content/maps-and-navigation/guide-to-the-wp8-maps-api.html>