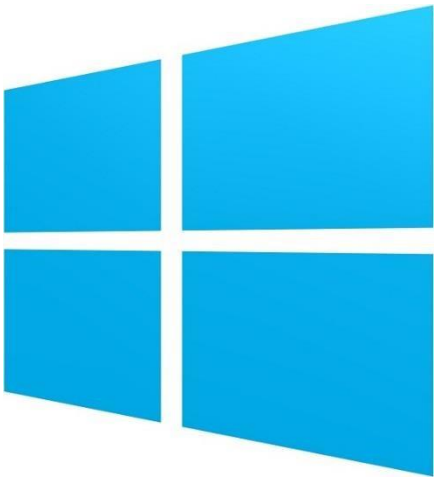
Team Microsoft

School of CIS Windows Store App

It's in the code.



Jason Tierney | Chris Field | Eric Foertsch
4-25-2013

Contents

Abstract:

The School of CIS Windows Store application was set forth as both a way to learn new Microsoft technologies as well as provide a Windows Store app for the GVSU School of CIS. The app provides services for students, alumni, faculty, and prospective students who either want to get specific information or want to learn more about the School of CIS. Features offered include professor information, a job board, news / events feed, Twitter feed, and a video feed from the School of CIS's YouTube channel. This app was developed in coordination as a team of three using SCRUM methodologies. The project also provides a useful and intuitive way for specific users to update the back end data that the app presents to its users. This data is powered by Microsoft's Windows Azure cloud service and the UI components are built upon Microsoft's ASP.NET stack, utilizing the MVC architecture. Ultimately, this project allows for a great way for students to learn Microsoft technologies as well as giving them the opportunity to create an app that is deployable to the Windows Store.

Introduction

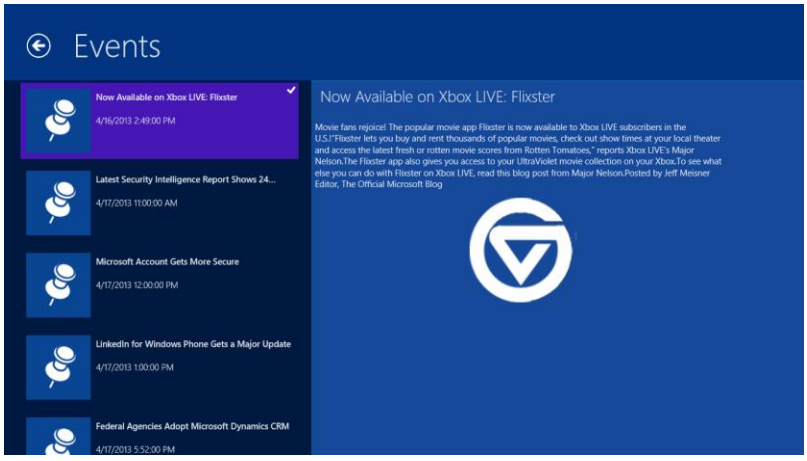
The focus of this app was to both create a Windows Store app that provides a portal to the School of CIS as well as learn new Microsoft technologies. The application provides functionality both from a front-user perspective and also offers back-end functionality. Due to having two separate aspects of the project, it was decided to be split into several different projects to help abstract a lot of the work into smaller components. Throughout this report both the back end and front end will be discussed as separate components that work together to make one big piece.

Front End - An Overview

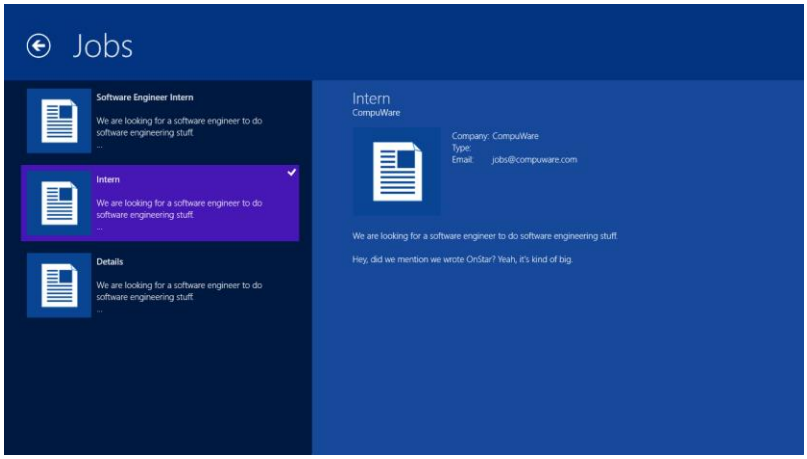
The front end portion of the project includes the Windows Store app. Within the app there are several different forms of coding, including Views, ViewModels, Models, and Services. These different types of coding will be discussed in the body of this report. The main features from an end user perspective include:

Application home page

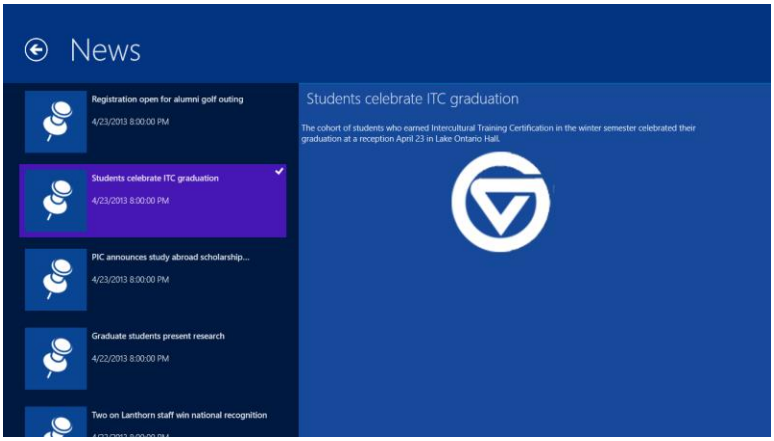
Events Page. The events section displays a list of current events at Grand Valley. If the GVSU feed were to cease to exist or error in any way, the feed will redirect to the Windows Blog RSS feed and display the current events at Microsoft related to Windows 8.



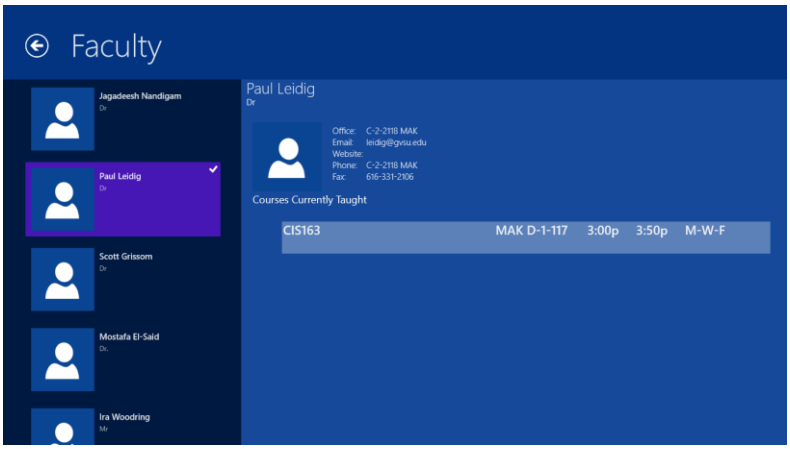
Job board. The job board section displays job listings as found on the physical job board near the CIS department offices in Mackinac Hall. A user can view different information pertaining to each posting, including company name, type of position, contact information, and a description of the job.



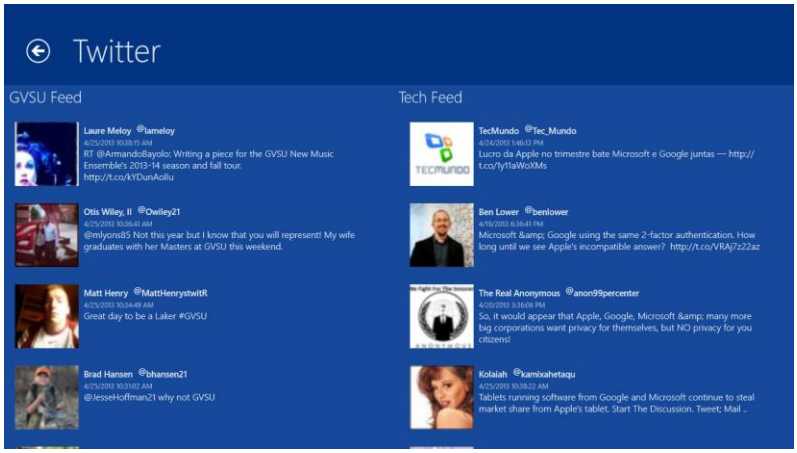
News. The news section displays current news at Grand Valley. If GVSU feed were to cease to exist or error in any way, the feed will redirect to the Visual Studio RSS feed that displays current news related to Visual Studio.



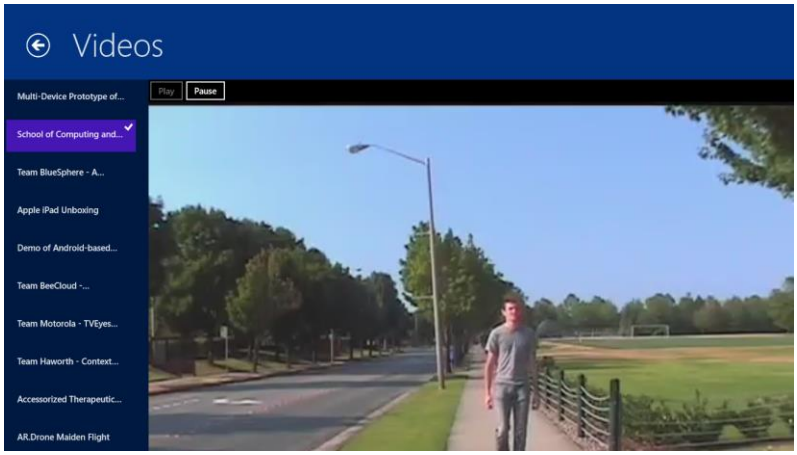
Faculty. The faculty section displays a list of the current faculty and staff employed in Grand Valley's CIS department. By selecting a professor, a user can learn where that professors office is located, what their office phone/fax numbers are, what email address to contact the professor at, and what courses they are currently teaching.



Twitter. The twitter section displays two different set of semi-live tweets. The first set displays a list of tweets that contain the hashtag #GVSU or any mentions Grand Valley. The second set displays a list of tweets that have to do with big technology industry companies, such as Microsoft, Apple, and Google.



YouTube / Video. The video section of the application displays a list of videos pulled directly from the Grand Valley's CIS department's Youtube channel. When a user selects a video it will automatically start to play within the application. Controls above where the video appears allow the user to pause and resume the video.



Back End - An Overview

As well as providing the features to the end user, the app also has several custom backend services it utilizes in order to get the data to the app. The backend services provide both a way to sanitize existing data feeds (Twitter) and also allows for custom data to be used within the app. This backend service is powered by Microsoft's Windows Azure cloud platform and uses JSON formatted strings to send data to the app. The following are some of the features the backend provides for the app. A more detailed discussion of these features will be included later in the report.

Table Storage. Microsoft's Table Storage allows storing textual data. The advantage to table storage is that there isn't the need of creating and maintaining a full database. In addition, entries in table storage can still have unique identifications, similar to a database, but they do not require SQL statements to fetch the data. Instead, standard LINQ statements can be used in lieu of SQL statements.

Blob Storage. Microsoft's Blob Storage allows for storing images on the web service. Blob storage has similar advantages over SQL statements that table storage offers.

CRUD Pages. There are several CRUD (Create, Read, Update, Delete) pages that specific users can use in order to update the data on the back end. These pages interact with the table and blob storage directly to allow users to add, update, and delete data as needed.

Data Interaction. All of the data sent to the app uses JSON formatted strings. This was a design choice chosen based on the popularity of JSON. JSON data is also very easy to parse and to write in C# with the help of the open source library JSON.NET.

Having a combination of blob and table storage allows the app to handle a complex amount of information. Each type of storage also offers an easy way to store, identify, and retrieve unique records without having to manage a database and write SQL statements.

Front-End Features

The School of CIS Windows Store application has a number of features available for its end users. These features allow current and prospective students, alumni students, and faculty to see what is currently going on in Grand Valley's School of CIS department. This application demonstrates the ability to:

- View up to date Grand Valley State University news stories.
- View current events happening around Grand Valley State University.
- Access a comprehensive list of all Grand Valley School of CIS professors and faculty and relevant information about them such as classes taught, office hours and contact information.
- View an aggregation of social media related to GVSU or major technological companies.
- Search for jobs posted by the Grand Valley's CIS department
- View videos posted by Grand Valley's CIS department

Back-End Data Features

This section describes, in more detail, the back-end data services that are used within the application. The application grabs and sends out all of its information via a live web service which is powered by Microsoft's Windows Azure cloud service.

Communication

The application communicates to the Azure Cloud backend using JSON formatted strings. Using an HTTP GET function, JSON string messages are retrieved from the WebService and returned to the application. There is a specific HTTP GET message for each specific page on the app.

NOTE: Each of the following GET functions below uses a standard 127.0.0.1 IP Address. For deployment, this address should be replaced with the website address of the hosted Azure service.

Professor

The following call is made to retrieve the professor:
http://127.0.0.1/Professor/GetProfessors
Response:


```
[
   {
      "RowKey":"434a9adf-a54e-4be1-91d5-23996fd8e285",
      "PartitionKey":"0c859bb7-1004-415a-ac40-aa4a0a448ec6",
      "FirstName":"Jagadeesh",
      "LastName":"Nandigam",
      "ProfessorStatus":"Dr",
      "Position":"Associate Professor",
      "OfficeNum":"C-2-206 MAK",
      "Office":"C-2-206 MAK",
      "FaxNum":"616-331-2106",
      "Image":null,
      "Email":"nadigaj@gvsu.edu",
      "Website":null,
      "Prefix":null,
      "Courses":{

      },
      "OfficeHours":{
         "OfficeHours":null
      },
      "Title":"Jagadeesh Nandigam",
      "Subtitle":"Dr"
   }
]
```

Twitter

Request:

http://127.0.0.1/Twitter/GetTweets

Response:


```
[
   {
      "Time":"2013-04-25T17:00:02+00:00",
      "User":"ThaRealJono",
      "UserName":"Jono",
      "ImageUrl":"http://a0.twimg.com/profile_images/3450646224/ce02723afd1b20867705331c7
a085e4c_normal.jpeg",
      "ImageUrlHttps":"",
      "Source":"",
      "Text":"“@PlatinumLasher: @ThaRealJono I appreciate that. How did you like gvsu?”
}

]
```


YouTube

Returns a list of YouTube video feeds / links from the School of CIS YouTube page.

Request:

http://127.0.0.1/YouTube/GetVideos

Response:


```
[
   {
      "Title":"Multi-Device Prototype of Quick Graph for iPad/iPhone",
      "Url":"Y0NFllUQs6Y"
   },
   {
      "Title":"School of Computing and Information Systems",
      "Url":"y8szCk_oCgU"
   },
   {
      "Title":"Team BlueSphere - A Location aware social networking mobile app",
      "Url":"JYfhQorfaxk"
   }

]
```

Job Board

Returns a JSON formatted string of a list of jobs for the job board.

Request:
http://127.0.0.1/Jobs/GetJobs

Response:


```
[
   {
      "RowKey":"a4435352-90ff-4b1f-a41d-9a18e265f655",
      "PartitionKey":"07293059-f420-4b04-b007-a77c4809f75a",
      "Description":"We are looking for a software engineer to do software engineering st
uff.\r\n\r\nHey, did we mention we wrote OnStar? Yeah, it's kind of big.",
      "Company":"CompuWare",
      "Title":"Software Engineer Intern",
      "Website":null,
      "Phone":"616.651.2189",
      "Email":"jobs@compuware.com",
      "Image":null
   },

]
```


// Place an appropriate response here when we format it in Word.

Updating the Backend

Specific users can update the information stored in the backend. This feature is provided by a simple UI using the Microsoft ASP.NET MVC stack. There are several CRUD (Create, Read, Update, Delete) pages that can be accessed using the web site. These pages allow users to update the back-end information without needing to know anything about coding. Currently, the website supports the following CRUD pages.

Professors. Permitted users can update all professor information using the professor CRUD page. This page first displays all of the current professors in the system. Selecting Edit / Add brings up the Professor Edit page which depending upon the mode either allows a user to add a new professor or edit the selected professor.

Twitter. Users can modify the Twitter feeds (search terms and users) using the Twitter CRUD page.

Job Board. The job board CRUD page offers an area to add, delete, or edit new job postings. These postings can include a picture and utilize both table and blob storage.

As the app continues to grow in the future, adding additional CRUD pages is not difficult for developers. This process is discussed later in the Code Base section.

Code Base

The GVSU CIS application has an extensive code base that is very simple to download to a local development computer running Windows 8 and Visual Studio 2012. To get access to this or any other Team Foundation Server project you must request appropriate access to that project. This particular application's source files are stored in Team Microsoft's Microsoft Team Foundation Server at the URL http://gvsu.visualstudio.com. Follow the steps displayed on the screen to request access to the visual studio server.

After acquiring appropriate access to the project, a developer may check out (download) the relevant files to their local development environment. This is done by "Opening a new instance" of Visual Studio from the browser or by connecting manually from the Visual Studio application itself. Individual files can then be checked out, modified and then checked in. By following proper procedure, edits and updates done to local files can be pushed out to the server from Visual Studio.

Source Control

The entire project is currently under source control using Visual Studio Team Foundation Service. This is a hosted service and can be accessed using either Visual Studio or a standard web browser. It requires a username and password to have either pull or push access. Updates to the GVSU CIS application have to pass through Visual Studio's / Team Foundation Server's source control. This control handles all developer modifications to local source files and handles all merges with the server versions.

Visual Studio 2012 has a very well developed source control, able to handle most conflicts automatically without any interaction from developers. Issues it cannot handle on its own are displayed to the developer, who can choose to keep their version, take the servers or attempt to merge the two.

Setting up the Dev Environment

Following these directions will get the project set up and running properly on a development machine. Some requirements are required in order to get the project set up on your dev machine:

- Windows 8 - required for Windows Store development
- Visual Studio 2012 (Ultimate is recommended)
- StyleCop (optional, but highly recommended)

After the development machine is setup, follow these directions:

1. Open Visual Studio
2. Click View --> Other Windows --> Source Control Explorer
3. Visual Studio should ask for authentication credentials. Enter these credentials in and press login.
4. Source Control Explorer should now connect to the repo. Select GVSU CIS Capstone.
5. Look for "Local Path" and it should say "Not Mapped". Click this link.
6. Select the folder to play the local repo.
7. Open the solution

Once the solution is loaded, the following projects should be available to work with:

Project	Description	Deprecated
Azure Data Storage	Used for setting up the connections to the Azure backend.	No
CapstoneWebsite	The website for the capstone project. This really isn't utilized anymore. We suggest incorporating it into the Computing Club's website instead.	Yes (Sort of.)
CISCapstoneWebApplication	Web application used for updating / pulling information from the Azure Cloud storage.	No
GVSU_CIS_Phone	This is an unfinished phone project that was more used for testing than anything else. This could be used for future projects.	No
GVSU_CIS_SplitApp	This is the main Windows Store application project.	No

Development Style

The project utilized SCRUM development methodology. Our team recommends future teams to continue utilizing the SCRUM methodology; however, any development method would work for future development. Additionally, developers are encourage to use StyleCop and Code Analysis settings within the project. These settings are already set up within the Solution file and doesn't require additional modification to maintain the standard settings.

Working on the Front End

The front end requires both basic design and coding skills. Because the application utilizes a front end GUI, there is a lot of work that has to be done using XAML for the GUI. XAML is Microsoft's front end language for creating GUIs in many of its modern frameworks, including Windows Store apps. Other frameworks that utilize XAML as their language of choice for GUIs are WPF and Silverlight.

The Model-View-ViewModel design pattern is used throughout the front end code. This design pattern allows for separation of the view from other specific code sections, thus allowing for data abstraction at several different layers. There are three specific folders / namespaces in the project that match very closely with the design pattern:

View. Any XAML specific code should be placed in this folder / namespace. There should not be any code behind in these files / namespaces (i.e. any .cs file, only .xaml files). These files contain the UI elements of the application as well as how to handle the varying states of the application (rotated, snapped, etc). Finally there are XAML specific C# files that handle events done to the UI (such as clicking a button).

ViewModel. These are helper classes that interact with the view and the model. These classes generally hold ADTs, as well as offer a place for most event handlers to live. Click event handlers should be abstracted from the code behind in the view. (i.e. there should not be any code in the xaml.cs files.)

Model. Classes found in the Model directory are representations of various components of the application. These classes include but are not limited to a Youtube video, a job, a generic RSS feed item and a professor. Included in the model directory as well are representations of collections of these items such as a professor group, job group, etc. Finally there are data source files for each collection of items found in the directory which serve to pass data to the view models.

Each class should be included in its own file and there should not be any files with more than one class residing in them as per standard design.

Setting up Connections to the Web Service

The following example code can be used to setup a connection to the back end web service.

```
/// <summary>
/// Makes an asynchronous call to the webservice.
/// </summary>
public async Task<IEnumerable<CISCapstoneWebApplication.Model.Job>> GetJobsAsync()
{
    HttpClient client = new HttpClient();
    client.MaxResponseContentBufferSize = int.MaxValue;
    HttpRequestMessage request = new HttpRequestMessage();
    request.Method = new HttpMethod("GET");
    request.RequestUri = new Uri("http://capstoneservice.cloudapp.net/Job/GetJobs",
            UriKind.Absolute);
    var response = await client.SendAsync(request);
    var message = await response.Content.ReadAsStringAsync();
    ObservableCollection<CISCapstoneWebApplication.Model.Job> jobs =
    JsonConvert.DeserializeObject<ObservableCollection<Job>>(message);

    return jobs;
}
```

Most of the data conversion is handled by JSON.NET using data serialization. This makes converting JSON data into usable objects a lot easier for developers.

Working on the Back End

The backend required us to have skills with ASP.net MVC websites, Windows Azure, and JSON messaging. Windows Azure was a decently straightforward system to learn and begin to implement. Azure provides an online portal to work with the different cloud services offered, such as storage tables. To connect our website to the tables we are using for storing data we needed a Cloud Service Project within Visual Studio that had a web role that was linked to the websites project, within this role a connection string needed to be created with the storage table name and primary key associated with that table.

Adding / Modifying Controllers

At the base of the backend service are controllers which offer up most of the content that the application sees. This content is processed through the controller (and often times helped with a service class as well) and then passed onto the application through a JSON string message.

SECEPP Issues

The following section discusses SECEPP issues that arose throughout the project and the semester. These issues were handled via group discussion.

3.01. Strive for high quality, acceptable cost, and a reasonable schedule, ensuring significant tradeoffs are clear to and accepted by the employer and the client, and are available for consideration by the user and the public.

This issue largely addressed during the first few weeks of the project while our team was still getting familiar with the development tools we would be using. During meetings the team would discuss ideas of assorted features we would like to be included in the application, starting with the base description given to us by our sponsor. In the final meetings before development began we clarified with our sponsor what features we were going to implement and which ones we were going to omit from the project. We were given a deadline by Eric Maino to have our MVP by the fifth week of the semester.

3.02. Ensure proper and achievable goals and objectives for any project on which they work or propose.

During the initial planning stages we made a basic plan as to what the status of the application should be when we reached the MVP release. From the MVP release we were able to decide what needed to be done in a general sense before the end of the term. We would create achievable goals and tasks each week during our SCRUM meeting that were meant to be accomplished before the next meeting.

3.04. Ensure they are qualified for any project on which they work or propose to work, by an appropriate combination of education, training, and experience.

The first two weeks of the project were spent getting familiar with the development environments we had available to work with to help us choose what direction we wanted to take the project. During this time we made sure that we understood the environments as best we could. As the project proceeded and we

added other environments, such as Azure, one or two members would spend an appropriate about of time getting familiar with the new environment before we started implementing it into the project.

Conclusion

Throughout the semester we learned a lot about Microsoft technologies, working as a team, creating a commercial product, and working with the SCRUM methodology. Although there was a lot of work and a lot of trouble along the way, the overall application appears to be a success and it will be very interesting to see how well the application takes off when it gets deployed to the Windows Store. Currently, we are just waiting on a solidified Windows Azure account before submitting the application to the Windows Store. Once the School of CIS has their own Windows Azure account we can move on over to this account and submit the application to the Windows Store using the School of CIS's Windows Store account.

Areas of Learning

Throughout the semester each member learned a lot about the Microsoft stack. There was not much of the Microsoft stack that we didn't get some exposure to. From simple C# development to trying to learn new frameworks and exploring the open source development community, there was a lot of new technology that each of us was able to get our hands dirty with and learn about. Although all of us had some knowledge of the C# language and .NET Framework, we had to completely learn new technologies such as ASP.NET, Windows Azure, the Windows Store libraries, and many other libraries and frameworks that we found useful along the way.

Areas of Trouble

As with any project, we did experience a few issues along the way. For the most part, these issues were resolved very quickly; however, we did occasionally run into some issues that took much longer to work out. Some of the major issues we had include the following.

- Time for tasks took much longer than we had planned
- Some tasks were overlooked
- A lot more time was spent sometimes learning how to do something than actually doing

Future Plans

When discussing future plans with Eric M., he mentioned that all of the code we produce is solely ours and we can do with it as we please. As a group, we would all like to see this code persist throughout the academic community at GVSU. To get the most use out of the project, handing the code over to the GVSU Computing Club might be the best way to see this project continue to evolve. Additionally, each of us has said we would commit sometime in the future to get the project started within the Computing Club and help students out if they have any questions with the project. There are still several features that we would like to see implemented by future students, but these are not requirements by any means and we would ultimately prefer the creativity of future students to not be burdened by strict requirements.