# GVSU CIS Windows 8 Application

# Design Document



## Team Microsoft

### Members

Jason Tierney

Eric Foertsch

Chris Field

### Introduction

Windows 8 is an evolution of the popular Microsoft Windows operating system. Microsoft released Windows 8 on October 26, 2012. Because Windows 8 is built with a touch interface in mind, a key feature is the ability to download and run applications (“apps”) from Microsoft App Store, much like tablets running the iOS and Android platforms. Because Windows 8 is relatively new to the market, there is currently a huge market for new Windows Store apps.

Team Microsoft has been petitioned to design and create a Windows 8 application for the Grand Valley State University Computer Information Systems (“CIS”) Department. This application will be used by everyone affiliated with the CIS department, both past and present. This includes current and prospective students, alumni, faculty or anyone interested in Grand Valley’s CIS department.

The application itself will contain information about what is going on in the department. Information such as news, event’s, job boards, CIS related Youtube video’s, etc.

### Languages Used

To design Windows 8 applications, there are two main sets of languages that we considered for writing this program HTML/Javascript or C#/XAML. The first, HTML/Javascript is mainly used to design websites but can be used in Windows 8 application development. Microsoft does support HTML/JavaScript entirely and have stated that using these languages, you can accomplish the same amount of work as other languages, such as XAML/C#. The second the set of languages we ultimately decided on is C#/XAML. This set of languages allows us to write a fully functional application rather than a web application. The majority of applications created for Windows 8 are written in C#/XAML.

### Frameworks / Libraries / API’s Used

This section briefly discusses the frameworks, libraries, and APIs that we will use while design our product. The following frameworks, libraries, and APIs we plan to use for this project are listed below with additional detail.

* **Microsoft .NET Framework.** This is required for all Windows Store apps. This framework allows the program to utilize multiple languages.
* **Callisto.** A UI framework developed by Microsoft employees used to extend the UI components in Windows Store apps.

As we progress throughout this project, more frameworks may be utilized. This is potentially part of the nature of the development process we are utilizing. Scrum allows for changes throughout the entire development phase as only bits of the project are known at any given time.

### Code Repository

There are two code repositories used for this project. The first is a cloud hosted repo using Team Foundation Server (gvsu.visualstudio.com). This code repository is used for all of the code we write for this project. The second code repository is hosted on GitHub. This repository’s sole purpose is for any documents relevant for this project, such as this design document.

#### Team Foundation Server

Our default code repository is hosted on Team Foundation Server (gvsu.visualstudio.com). This repository will house all of the code we produce for this class. This is a private repo and does require a username and password for both contributors and viewers.

The source code of this project is stored within a private Microsoft Team Foundation Server repository (gvsu.visualstudio.com). With Team Foundation Server, code is associated with tasks assigned to group members. When a group member completes their assigned task, they are able to associate their code with the task and then check it in. In addition, if a group member wants to work on a file, they must first check it out. This is done so group members do not overwrite each other’s edits.

#### Github

The other repository being used for this project is hosted on Github under the name “fieldc15/Team\_Microsoft”. This repository is used to hold different documentation for the project that is required for the capstone class. Documentation found within this repository include the team members individual journals, the project prospectus, and this project design document.

### Database’s Used

The application will utilize a SQL database hosted on Windows Azure. One of the goals’ of this project is to build a product using Windows technology. The world of computing is moving towards cloud computing and Windows Azure is Microsoft’s cloud computing platform. The SQL database would be created on this platform to feed data into the application.

Since a virtual computer on Windows Azure costs money, this project will utilize a free trial Windows Azure platform. If the Windows Azure/SQL database solution works out, a more permanent Windows Azure cloud computer can be obtained.

Data stored in this SQL database would include information about professors, classes, news articles and other basic information used in this application. If we decide to implement a SQL server database on the backend, we will use a combination of Visual Studio 2012 and SQL Server 2012 to implement the database backend.

### Basic Organization of the System

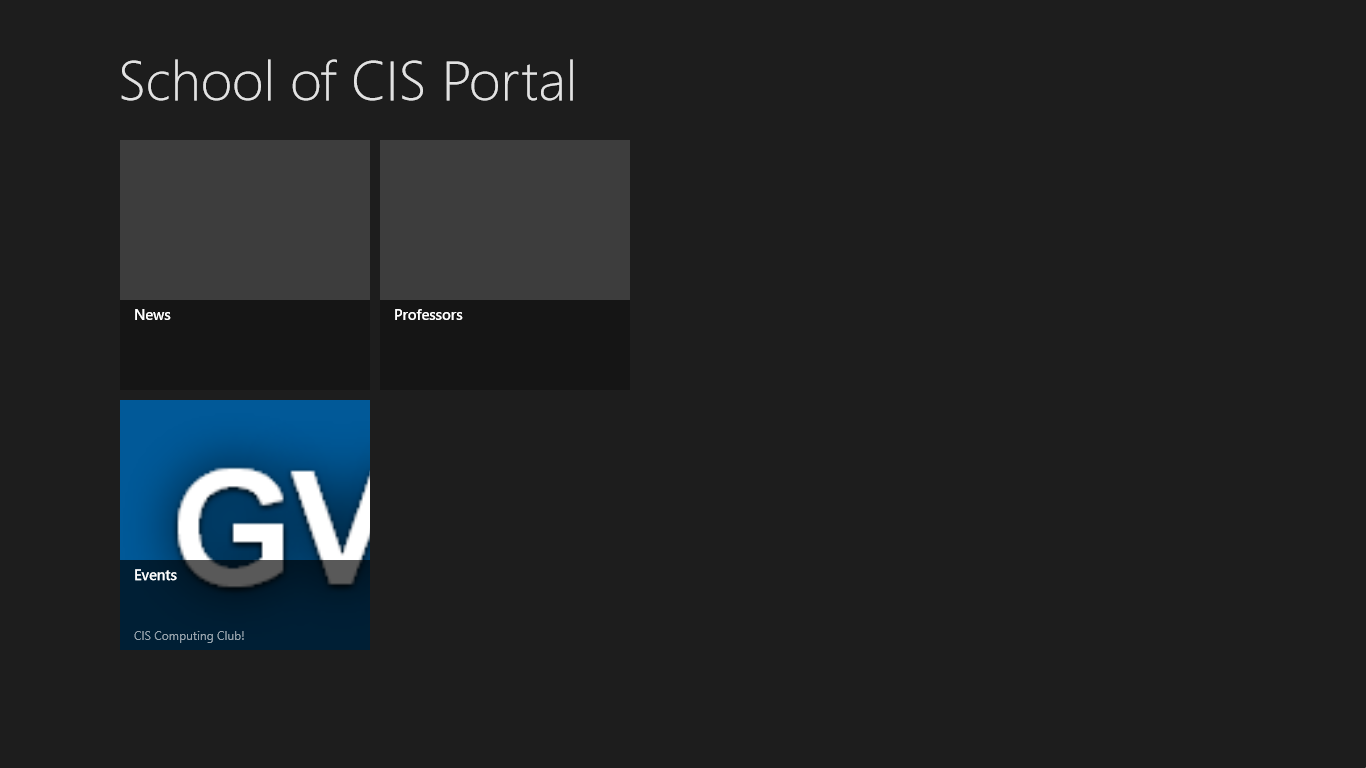
The system will consist of a basic Windows Store app on the front end with several data feeds on the backend. These data feeds will consist of standard RSS/AJAX or custom data backends which work with JSON formatted data strings. Custom data backends may require SQL Server with Windows Azure running on the backend.

For more information about the project scheduling and timeline, please refer to our Microsoft Project file under DesignDocuments which includes our schedule and gantt chart.

### 

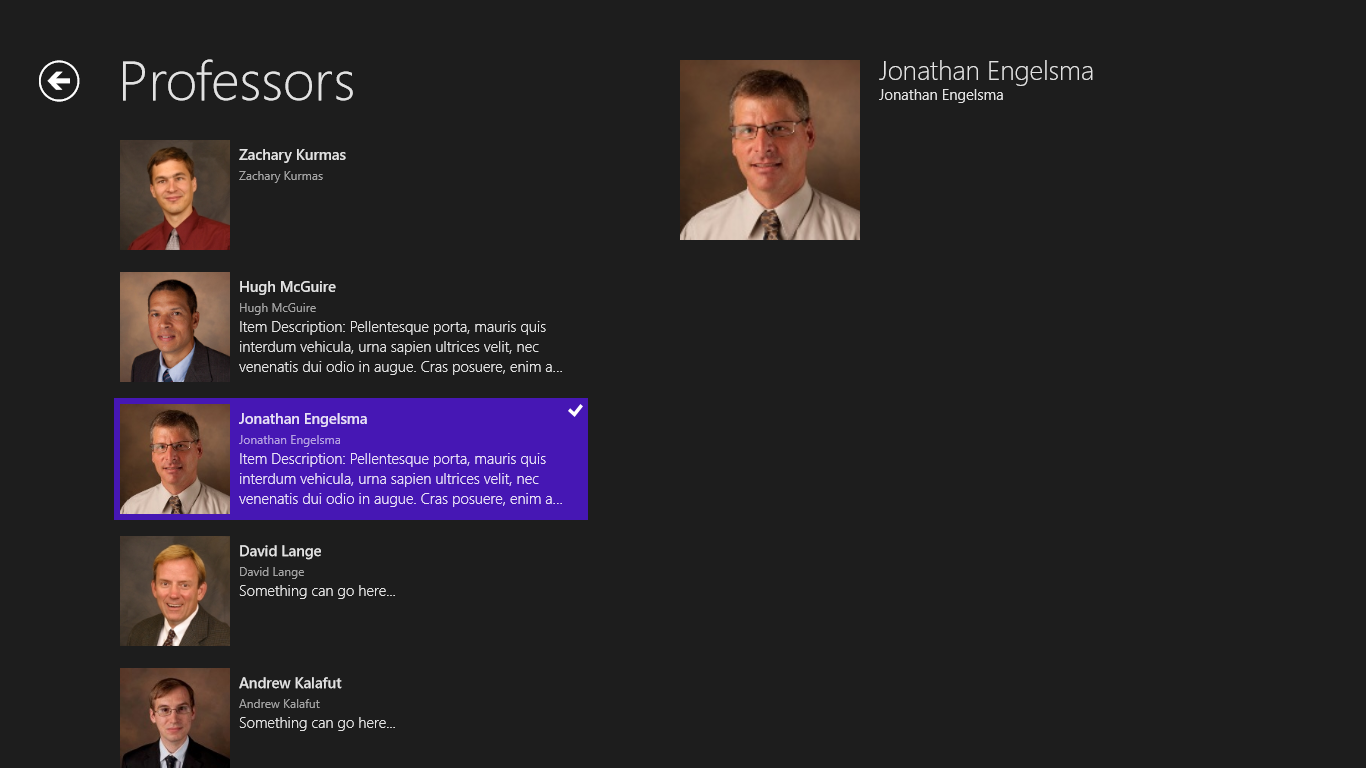
### User Interface

The user interface is built like a standard Windows 8 application. The GUI is based on the template Split Screen Application with modifications. The front page will divide the application into different sections that separates relevant data into each section (Figure 1.1).



*Figure 1.1 GVSU CIS Windows 8 Application Home Page*

In each section, the user will be presented with list on the left hand side. The user is then able to select each individual item in the list to see more information. For example, in the “Professor” section the user is able to select a professor. The application will then present the user with information about the professor; class schedule, office hours, etc (Figure 1.2).



*Figure 1.2 Professor Section of the Application*

Since this user interface is built with Windows 8, the application is designed for touch based input. For this application, the focus is on mouse and keyboard based input. This application will support both touch and mouse and keyboard input.

### Reasonings

The reasonings behind many of our methodologies is based primarily on the type of product we are creating. In this case, the application is a Windows Store app and requires specific frameworks, languages, and development environments in order to create a fully functional app for Windows 8. Because we are already using the Microsoft stack for the majority of our development, the team employs further Microsoft technologies to leverage their existing knowledge and resources of current products.

### Protocols used

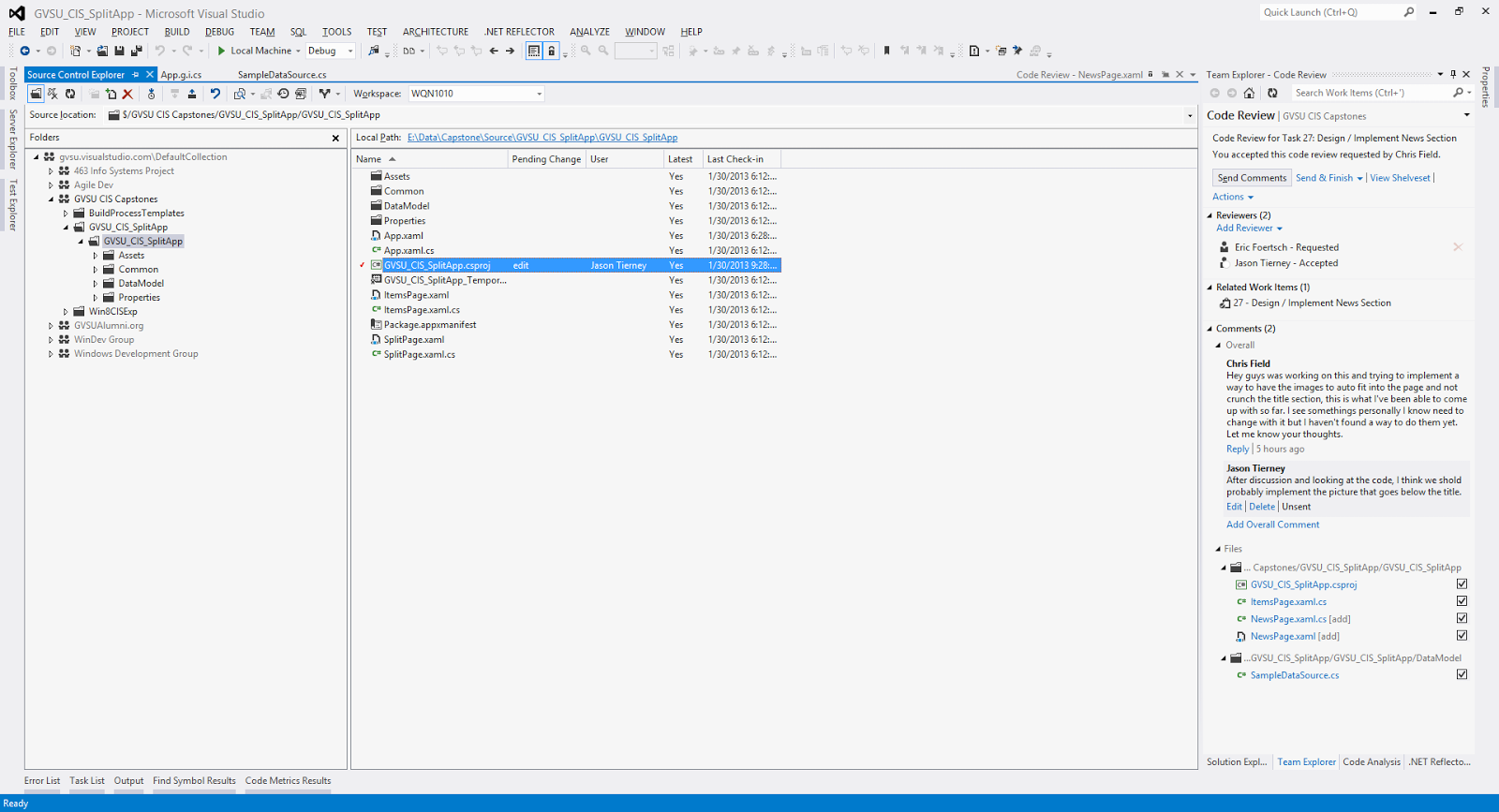
All components of this project will be using standard protocols used in Windows and the .NET Framework.

### Testing

To test this application, the development team will employ several methods of testing. The main source of testing will be the built in Microsoft Visual Studio debugger to test the XAML portion of the application. The C# portion of the application will be tested with both unit tests and with the debugger. Finally this program will both be tested on a computer running Windows 8 and a Microsoft Surface simulator built into Visual Studio. Utilizing these methods, any and all bugs will be removed and a polished product will be created.

### Development

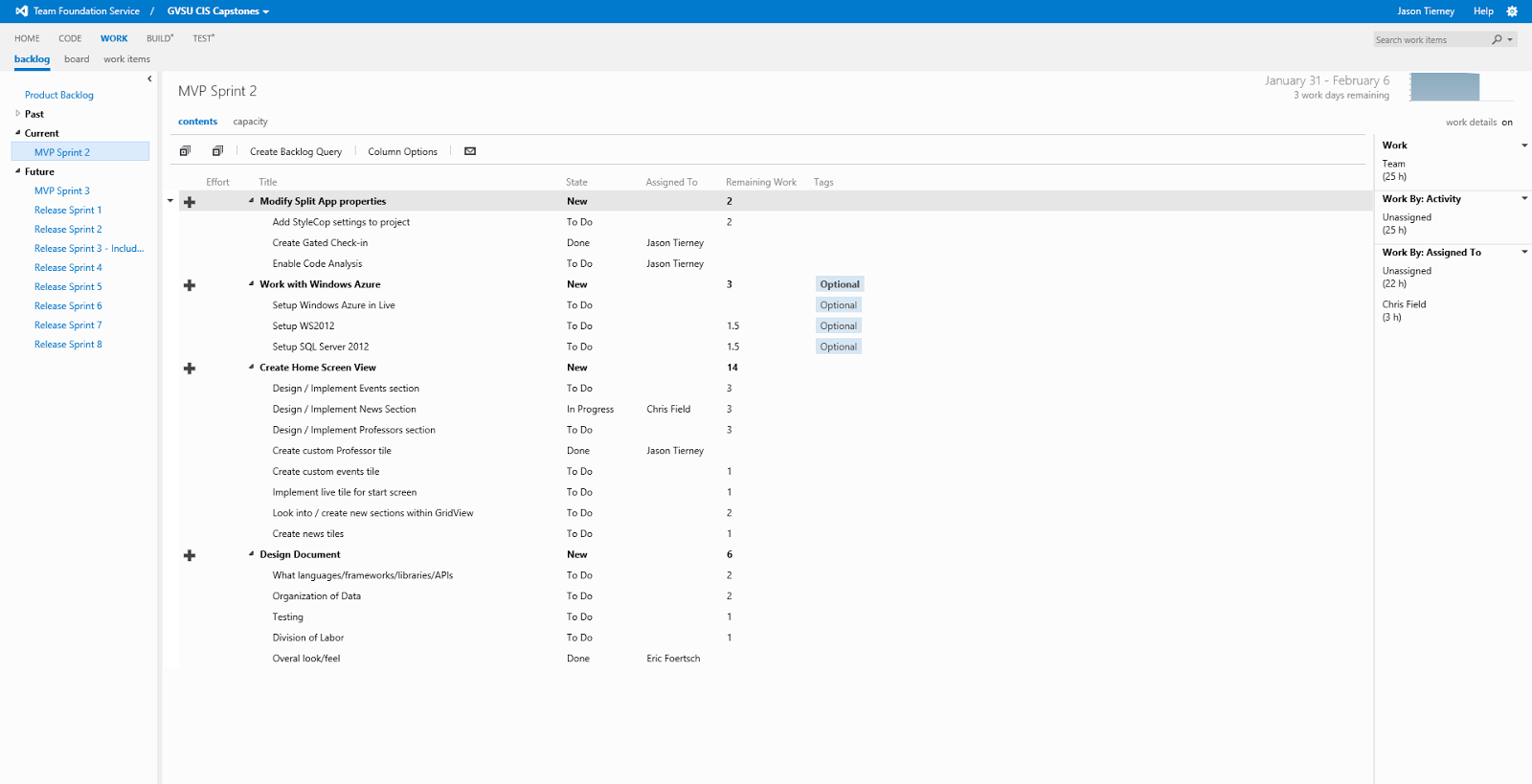
Due to the nature of Windows Store app development, it is require our dev machines have both Windows 8 and Visual Studio 2012 installed. The version of Visual Studio we will use is Ultimate. The Ultimate version of Visual Studio 2012 offers us the features we require to integrate with Team Foundation Server (Figure 2.1).



*Figure 2.1 Visual Studio Development Environment*

### Division of Labor

Development of this application will be divided into many tasks that encapsulate one specific goal, such as “Create a Button” or “Connect program to a SQL database”. Major tasks or tasks that must be completed will be assigned to group members during the weekly sprint meeting (Figure 3.1). Other, less important tasks are left unassigned and to be picked up and assigned to themselves by group members who feel they can accomplish the task.



*Figure 3.1 Team Foundation Server Task Backlog Page*

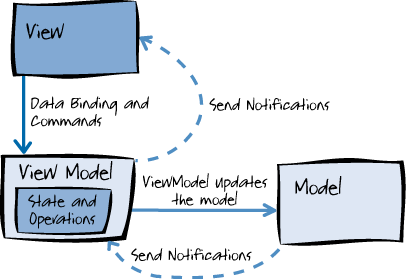
### 

### 

### Design Methodologies

Throughout the course of this project we will use the design methodologies presented and used in the Scrum programming paradigm. We hold weekly sprint planning meetings where we create tasks for the next week. These meetings are scheduled every Wednesday at 4:15 and go anywhere from one hour to two hours.

The overall design of the project will follow the Model-View-ViewModel (MVVM) design paradigm that Microsoft suggests for XAML type applications. This design model is very similar to Model-View-Presenter (MVP) or Model-View-Controller (MVC). However, one of the key differences with this design methodology is that it allows for data binding and thus complete removal of any logic/data from the view. The following diagram illustrates the idea behind MVVM in a much more succinct way.



*Figure 4 Illustration of MVVM design model. (MSDN)*