

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
UNIVERSITY OF NEBRASKA—LINCOLN

# Campus Map Project

---

## CSCE 361 – Software Engineer Project

**Cody Bodfield, Jiachun Han, Yan Xin Lee, Toan Nguyen**

**11/15/2013**

**Version 1.0**

This document describes a system design that will allow qualified users to upload pictures they took around campus, share them with other members via place markers on Google Maps.

## Revision History

Version	Description of Change(s)	Author(s)	Date
1.0	Initial draft of this design document		2014/06/17

## Table of Contents

Revision History .....	1
1. Introduction .....	3
1.1 Purpose of this Document .....	3
1.2 Scope of the Project.....	3
1.3 Definitions, Acronyms, Abbreviations .....	3
1.3.1 Definitions .....	3
1.3.2 Abbreviations & Acronyms .....	3
2. Overall Design Description.....	4
2.1 Alternative Design Options .....	4
3. Detailed Component Description .....	4
3.1 Database Design.....	4
3.1.1 Component Testing Strategy .....	5
3.2 Class/Entity Model .....	5
3.2.1 Component Testing Strategy .....	5
3.3 Database Interface.....	5
3.3.1 Component Testing Strategy .....	5
4. Agile Development.....	5
4.1 Sprint 1 .....	5
4.2 Changes & Refactoring.....	6
5. Website Implementation .....	6
6. API Implementation .....	6
6.1 Twitter API .....	6
6.2 Google Maps API.....	6
6.3 Imgur API.....	6
7. Bibliography .....	6

## 1. Introduction

This is the Software Design Description of Campus Map Project system for the University of Nebraska – Lincoln. That will allow qualified users to upload pictures on and around main city campus, and share them with other members through place markers on Google Maps. It outlines the technical design of the application that is being developed as a small social network of City Campus photography.

### 1.1 Purpose of this Document

The purpose of this document is to outline each iteration of this project. This includes successes and failures, testing methodology, and implementation strategies. This document also outlines the system's MySQL database, ASP.NET web application, utilized API's, and backend web hosting implementation.

### 1.2 Scope of the Project

This ASP.NET web application is developed for qualified users to upload photos they take around city campus. It is designed as a small social network whose functions allow photo uploading, geo-tag plotting to Google Maps, commenting on photos, and filtering of other users' uploaded photos. This project integrates the Google Maps API for photos to the campus map, the Twitter API as the login manager, and the Imgur API for image uploading and persistence.

### 1.3 Definitions, Acronyms, Abbreviations

#### 1.3.1 Definitions

**BizLogic** – A sub-project that encapsulates our application's primary classes and data access classes

**ASP.NET** – The server-side web application framework utilized in our web application

#### 1.3.2 Abbreviations & Acronyms

**API** – Application Programming Interface

**AWS** – Amazon Web Services

**ER Diagram** – Entity-Relationship Diagram

**IIS** – Internet Information System

**SQL** – Structured Query Language

**UML** – Unified Modeling Language Diagram

**XML** – Extensible Markup Language

## 2. Overall Design Description

In keeping with the OOP paradigm for this application development the use of unique classes is essential. Relevant data, methods, and functionality are built into respective classes based on designing good encapsulation enforcement. The current primary classes include: Comment, Picture, Profile, Results. Its data access classes include: CommentData, PictureData, and ProfileData.

The Result class is a generic class. It is designed to push success or error messages from our data operations down to the user interface. The Results class holds (at minimum) a boolean for success or failure, and a string message.

### 2.1 Alternative Design Options

Alternative design options considered in the development of this application:

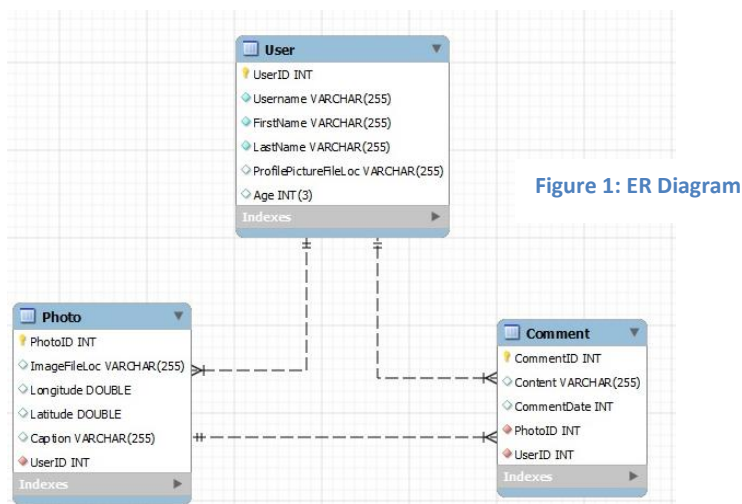
- Implementing a MVC architecture instead of 3-Tier Client/Server
- Building project in Java EE instead of ASP.NET
- Implementing SQL database instead of MySQL database

## 3. Detailed Component Description

Classes are used to represent instances of given objects. Object creation is handled by constructor methods in the respective classes via provided data. The provided data values are encapsulated and belong to the class they were used to create. By default, all member variables of classes are set to private so that variable interfacing is handled by property methods contained with the parent classes.

### 3.1 Database Design

The ER Diagram presented in Figure 1 represents the application's MySQL database schema. This diagram denotes the primary tables with their respective fields and relationships to other tables in the schema. The database schema is loosely based on the VB.NET class entities presented in the application. Careful planning was done to enforce separation of distinct tables and ensure encapsulation of proper information into their respective, distinct tables.



### 3.1.1 Component Testing Strategy

Database testing to be implemented in future sprints.

## 3.2 Class/Entity Model

This part will be illustrated by a UML diagram in future sprints. Following is the first draft description.

There will be four classes for this project, Comment, Picture, Profile, and Result class. The Comment class is used for users to do comment on their photos and on others' photos. It consists of six variables: \_sConnection, \_sComment, \_sPicture, \_dTime, \_sCommentID, and \_sUsersID. The Picture class is used to contain pictures users uploaded. It has variables of \_sConnection, \_sLongitude, \_sLatitude, \_sCaption, \_sPictureID, \_lComment (as list of String), \_sUserID and \_sImagePath. Then we will also have Profile class. It has variables of \_sConnection, \_sUsername, \_sFirstName, \_sLastName, \_sUsersID, \_nAge, \_sProfilePicturePath, \_lCommentList (as list of Comment), \_lPictureList (as list of Picture). At last, we have Results class. Results class allows us to push success or error messages from our operations down to the user.

### 3.2.1 Component Testing Strategy

Class component validation and testing to be implemented in future sprints.

## 3.3 Database Interface

Database interface to be implemented in future sprints. All database interfacing will be conducted through class' respective data access classes.

### 3.3.1 Component Testing Strategy

Database interface testing to be implemented in future sprints.

## 4. Agile Development

Development of this web application is implemented using the Agile software development paradigm. The method of agile development chosen for this project is SCRUM. Our development timeline is divided into four sprints. The application is built incrementally, approximately 25% of project completion per sprint, with any requirements adaption being implemented as needed between sprints. Backlog management is handled via Trello at <https://trello.com/csce361groupproject> . Source code management is handled via GitHub at <https://github.com/CSCE361GroupProject> .

### 4.1 Sprint 1

The initial sprint of this application development focused the design of the project structure, division of tasks between team members, and projected backlog generation for each sprint. This sprint focused on the implementation of the project framework. This includes database design and implementation, project classes design and implementation, web page design, and backend hosting setup.

## 4.2 Changes & Refactoring

Sprint I – Initial design. No changes made.

## 5. Website Implementation

The backend hosting of the web application is handled on a dedicated Windows Server 2012 instance provided by Amazon Web Services. Deployment of website application to the web server is completed via direct file transfer within Remote Desktop Connection. The dedicated Windows server's IIS manager handles server controls used within the web application. The web application can be accessed at <http://54.88.28.177/Login.aspx>.

## 6. API Implementation

### 6.1 Twitter API

Twitter API will be implemented in future sprints to manage application login, profile registration, and profile loading.

### 6.2 Google Maps API

The Google Maps API will be implemented in future sprints to display photos on map via geo-tag data. Each marker on the map will represent a photo. Selecting a given marker will display the photo and comments.

### 6.3 Imgur API

The Imgur API will be implemented in future sprints to manage photo upload and hosting.

## 7. Bibliography

*MySQL Workbench Documentation*. (2013, November 13). Retrieved October 2013, from MySQL: <http://dev.mysql.com/doc/workbench/en/index.html>