Design Document

CSCE 361 - Spring 2018

UniHangouts

1. Introduction

1.1. The purpose of this document is to display the architecture and class relations at a high level for the UniHangouts web application. This document will cover the system's layered architecture model, as well as a class diagram table's content and relations. It will also include the tables and relationships which will be used in the database. The audience for this document is software developers, engineers, and system architects who will be maintaining the system and implementing its components.

2. Definitions

2.1. **Microsoft SQL Enterprise:** Development environment designed for creation and management of Database and Stored Procedures.

2.2. **REST Service:** Representational State Transfer, an architectural style used to approach communications used in web development.

2.3. **JSON:** JavaScript Object Notation, lightweight data interchange format used in serialization.

2.4. **Serialization:** Translating data structure or object state into a format that can be transmitted and reconstructed in a different environment or at a later time.

2.5. **ASP.NET:** An open source web framework for building modern web apps and services with .NET

2.6. **.NET Framework:** Is the full .NET API programmers target to create websites, services, desktop apps, and more on Windows. Libraries and programs targeting this API are not compatible with non-Windows environments.

2.7. **.NET Core:** Is a subset of .NET Framework for targeting Windows and Linux platforms. It primarily focuses on ASP.NET web applications and cross platform libraries.

2.8. **.NET Standard:** Is a subset of .NET Core that provides a common cross platform programming API to target Windows, Linux, Android, and iOS.

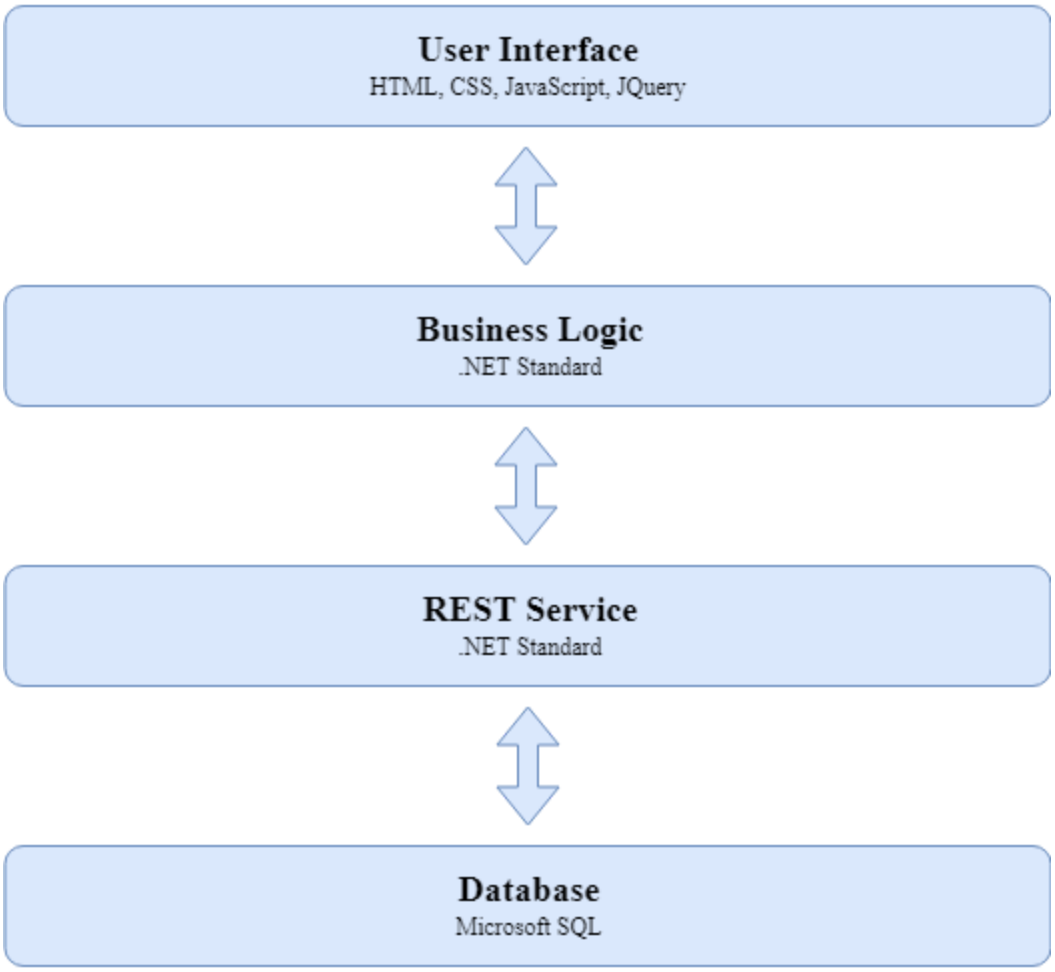
3. Architecture

3.1. Introduction

3.1.1. The system architecture consists of a SQL database, business logic, REST web service, and client layers.

3.1.1.1. The User Interface layer is the consuming application that provides the UI. This may be the ASP.NET website, Android, iOS app, or any other app that correctly interfaces with the REST service.

3.1.2. System Architecture Diagram



3.2. Modules

3.2.1. Database Layer

3.2.1.1. The database layer is responsible for holding all persistent user and event data and for defining the relationships between that information. This component uses a Microsoft SQL database. The relationships between data is described in detail in section 4.1

3.2.2. REST Service, Persistence Layer

3.2.2.1. The REST service layer provides web methods to indirectly interact with the SQL database from consuming apps, validation, and authentication. The REST service will target .NET Core.

3.2.3. Business Logic Layer

3.2.3.1. The Business Logic layer contains data models for serialization and common methods to be referenced and shared by the REST service and client apps. The business logic will target .NET Standard so that code can be shared across all platforms.

3.2.4. User Interface Layer

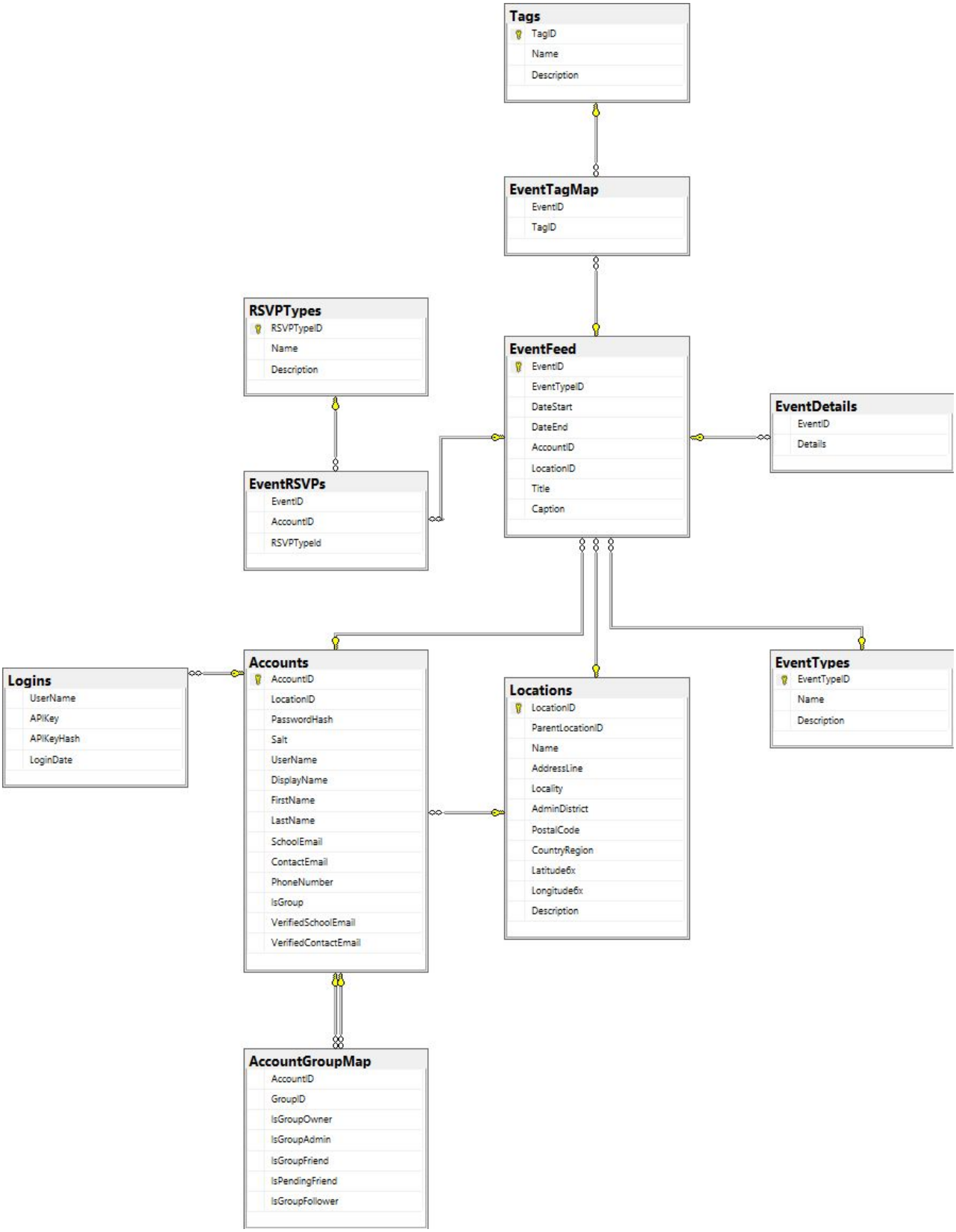
3.2.4.1. This is the application's presentation layer. A website will be made using HTML5, CSS, and a JavaScript framework. The site will receive information to display from the database once it has been altered in the logic layer. The structure is explained in section 4.3.

4. Class Diagrams

4.1. Data Table Classes

4.1.1.1. The system will be using a Microsoft SQL Enterprise database to hold all the necessary data. The database will includes information about users, events, groups, and the relationships between each. The diagram below demonstrates each table for the SQL database as well as the relationship between the different tables.

4.1.2. Schema



4.1.3. Schema Information

4.1.3.1. EventFeed: Holds all data relevant to the event feed, including the type of each event, the starting and ending dates and times, location, title, caption, and account associated with the event. The caption will provide details about the event at a glance and is limited to 160 characters.

4.1.3.2. EventDetails: Holds a full detailed description of the event. The detailed description allows several thousand utf-16 characters which would slow and bloat the EventFeed table.

4.1.3.3. EventTypes: Stores the type of each event and the description for each.

4.1.3.4. EventRSVPs: Keeps track of all users that have joined an event, including their account ID and their RSVP method.

4.1.3.5. RSVPTypes: Stores the name and description of RSVP types.

4.1.3.6. Locations: Holds all information related to the location that an event would take place at. This will include the name of the building, the address, postal code, and locality in terms of information that will be displayed to the user. This table will also keep track of latitude and longitude with the future implementation of an interactive map in mind.

4.1.3.7. EventTagMap: Table that keeps track of the relationship between each event and the tags that describe it.

4.1.3.8. Tags: Keeps track of each tag that can be used to describe an event, stores the name of each tag and a brief description of its meaning.

4.1.3.9. Accounts: Stores user account information, including login credentials, legal name, contact information, and group membership. To enhance security, passwords will be hashed, not stored directly.

4.1.3.10. Logins: Information needed for a user to log in.

4.1.3.11. AccountGroupMap: Stores information about how groups relate to certain users.

4.1.3.12. MapAccountsAndGroups: Keeps track of the relationship between each group and the accounts that are associated with it. This table will also store information about which users are the owners and administrators of the group, in addition to users that are following the group.

4.1.3.13. APIKeys: Table used to store identifying information of programs interacting with the database, including a flag to prevent banned users from interacting.

4.2. Class Information

4.2.1. Classes will be implemented in C# in the REST service and business logic layers of the application in order to represent and format the data coming from or being entered into the database. As such, there will be a class corresponding to each of the tables in the database. The business logic layer passes these classes to the GUI layer in order to be displayed.

4.3. GUI Layer

4.3.1. The top layer of the system will consist of four pages: the login page, the account page, the events feed, and the groups page. Once the user has logged in, there will be tabs at the bottom that the user may select to navigate to one of the three pages after the login page.

4.3.1.1. Login Page: Upon navigating to the application, the user will be prompted to enter their username and password. There will also be an option to register a new account for new users, and an option to sign in as a guest. When a user signs in as a guest, they will only be able to join events. They will not be able to join groups, create events, or view user information as a normal user could. The information entered by the user is verified against the database. If valid, the event feed is loaded by default. When the user logs out of their account via the account page, they will be returned to the login page.

4.3.1.2. Account Page: On the account page, the user will be shown information about their account. This includes groups that they may be a part of, events that they have signed up for, and their username. Users will be able to edit their username from this page if they wish.

4.3.1.3. Event Feed: In this screen, the user is shown a feed of all events that have yet to begin, loaded ten at a time. Information about the location and time of an event, and

what group it is associated with will also be displayed. From this screen, the user can select a specific event in order to join the event or learn more about it. The user will also be able to create a new event from this page if they are logged in. The event feed will also include search functionality. The user can select the search button, which will open a text box at the top of the page to enter queries into. After receiving the relevant information from the database, the application will display the groups or events that are related to the information that is searched for.

4.3.1.4. **Groups Page:** The group page will display all groups that the user is able to join, in addition to information about each group. There will also be an option on this page to create a new group if the user is logged in.