Raider Point

Design Document #1

Version 2.1

March 12, 2018

Scrummies

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Draft 1.0	Scrummies Team	Opening draft of design document #1	02/21/2018
Draft 1.1	Scrummies Team	Revisions to draft of design document #1	02/25/18
Draft 2.0	H. Hoyat	Revisions for final design document	02/27/2018
Draft 2.1	Scrummies Team	Final revisions for final design document #1	3/12/2018

Contents

1 Introduction

1.1 Overview

The purpose of this document is to describe the implementation of the Raider Point application as described in the Raider Point SRS document. It will illustrate how the team is organized, the tools the team will use to develop the software, and include a risk analysis for the project. This document is primarily for the use of the current development team, any individuals who will work to develop the system in the future, and project stakeholders.

1.2 Deliverables

The following is a list of deliverables as specified in the CSC 492 syllabus, from the current date to the final presentations.

Major CSC 492 Project Due Dates

- Design Document #1 Draft: Monday, February 26
- Design Document #1 Final: Monday, March 12
- Design Document #2 Draft: Monday, March 21
- Design Document #2 Final: Wednesday, March 28
- Status Meeting/Presentation: Monday, April 2
- Press Release #2 Draft: Monday, April 9
- Press Release #2 Final: Wednesday, April 11
- User's Manual Rough Draft: Monday, April 16
- Project Due: Monday, April 23
- User's Manual Final: Monday, April 23
- Final Presentations: Friday, April 27

1.3 Scheduled Reviews

Scheduled reviews for Raider Point will include weekly team meetings, dates for prototype demonstrations, and any other reviews scheduled in the future if needed. Weekly team meetings will be held every Wednesday and act as the primary source for reviewing team progress. Goals set from the previous week will be reviewed for progress, and will help determine what goals the team should set for the upcoming week.

During the first week of April, the team will have a working prototype of Raider Point to demonstrate current application functionality.

1.4 Glossary of Terms

Android - a mobile operating system developed by Google.

Authorized user - a person using Raider Point through the web portal.

Backend Database - a database that is accessed by users indirectly through an external application rather than by application programming stored within the database itself or by low level manipulation of the data.

Graphical User Interface (GUI) - a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators

Short Message Service (SMS) - a text messaging service component of most telephone, World Wide Web, and mobile device systems.

Agile life cycle model - a process for planning, creating, testing, and deploying an information system through agile methodologies.

Raider Point - name of this project. Used to refer to all software needed to make the application functional.

Scrum - an agile framework for managing work with an emphasis on software development.

Scrummies - team name, quoted from Dr. Cindric when describing how a scrum life cycle model would be an ineffective way to approach a project for this course.

Silver server - the web server maintained by Mount Union's Computer Science department.

Software Requirements Specification (SRS) - a description of a software system to be developed.

Web Portal - specially designed website that connects to a backend database.

Web Server - a computer system that processes requests via HTTP, the basic network protocol used to distribute information on the World Wide Web.

2 Project Organization

2.1 Software Lifecycle Model

The Kanban life cycle model will be the overall approach to develop the Raider Point application. Kanban is a type of agile life cycle model used for managing the creation of products with an emphasis on continuous delivery, while not overburdening the development team. This is achieved by using a Kanban board, a tool used to visualize work and optimize the flow of the work among developers. The board is separated into columns representing different stages of the development process, while cards representing different tasks to be completed are placed on the board depending on which stage of the development process the task is currently in.

Much like a scrum, other than having to meet and deliberate, tasks and objectives are clear and achieved throughout the project life. Kanban would be a beneficial choice for Raider Point since the application includes many features, making it important for all team members to know exactly how far a feature has progressed in the development process. Kanban would also allow the team to release multiple iterations of the application by prioritizing different features to pull from the backlog first, ensuring all necessary functionalities make it into the application.

2.2 Allocation of Responsibilities

Hailey Hoyat, current team leader, is in charge of planning meeting times and maintaining communication between the team and other persons outside of the team (i.e. team advisor, stakeholders). Hailey is also responsible for taking the lead on written documentation and deliverables, and will assist with designing the application GUI. Austin Olbrych, currently in charge of configuration management, is responsible for maintaining version control for team deliverables, and updating the team blog. Austin will also take the lead on developing the application within the Android environment. Evan Olbrych, current architectural designer, will take the lead on software testing, as well as setting up the backend database and web portal for the app.

While these current roles help divide responsibilities among the team, team members will most likely change roles and change their respective responsibilities when needed throughout the project.

3 Managerial Process

3.1 Reporting and Monitoring

Team monitoring and reporting will mostly be done each week at the weekly team meetings. Team members will report to each other on the progress made on various team goals, and decide what steps the team should take next to further the project. During these meetings, the team will also review the team schedule to keep track of how the project has progressed as a whole, and ensure the team can meet all deadlines. The schedule will be updated when needed.

In addition to in-person meetings, the team will communicate with each other throughout the week to ensure all team members are making progress with their responsibilities, help each other where needed, and answer any questions anybody may have. This will be done through email and SMS messaging. Collaborative software, such as GitHub and Google Docs, will allow team members to monitor each other's progress on a daily basis.

The team blog will be updated weekly with the team's progress and include team deliverables, to allow anybody to see how the project has progressed. Announcements will be made on the blog when key features or schedule updates occur.

The team will also update the team advisor, Dr. Kirchmeyer, throughout the project.

4 Technical Process

4.1 Tools

The team will use the following software tools to design Raider Point.

Tool	Purpose
Android Studio	Create application for Android environment
Microsoft SQL Server Management Studio	Create database
Silver Server	Store database and web portal
Microsoft Visual Studio	Create web portal
Github	Share and maintain version control for code
Google Docs	Share and maintain version control for written documentation

4.2 Documentation Strategy

Documentation for application code and written documents will be handled using various version control software. Github will be used to maintain the application code, allowing all team members to see the current state of the code, make changes, and share those changes with the team. Similarly, Google Docs will be used to maintain written documentation for the team.

5 Risk Analysis

5.1 Risk identification and risk management strategies

The following is a risk analysis for Raider Point, and the table below details the criteria used to conduct the risk analysis. The likelihood of occurrence describes how likely it is for a risk to occur, the severity of occurrence describes the impact the risk would have on the project if it were to happen, and the risk mitigation strategy describes what the team will do to reduce the chances of the risk happening.

Likelihood of Occurrence	Severity of Occurrence
Low	Tolerable
Medium	Moderate
High	Serious
	Catastrophic

1. Unfamiliar with software environments
    a. Probability: High
    b. Impact: Serious
    c. Mitigation Strategy: The team will audit their current software skills for the environments the project entails, and change the team schedule and app functionality to ensure the project is completed on time.

2. Feature Creep
    a. Probability: Medium
    b. Impact: Moderate
    c. Mitigation Strategy: The team will stick to the functionalities outlined in the SRS document, and not add any additional functionalities unless all others have been completed.

3. Team members behind on other coursework
    a. Probability: Medium
    b. Impact: Serious
    c. Mitigation Strategy: The team will use weekly goals to help break the project down into manageable steps to make completion of the project easier. Also, team members will help each other out if other team members fall behind on their respective responsibilities.

4. Not familiar with how to use the different software environments in tandem with each other
    a. Probability: High
    b. Impact: Tolerable
    c. Mitigation Strategy: The team will research and experiment how to link the necessary software environments together early in the project to identify any issues (i.e. database to Android app, web portal to database). If an unavoidable problem is found which cannot be resolved within the scope of the team's time and resources, alternative software environments and methods will be explored.

5. Version control software not working
    a. Probability: Low
    b. Impact: Tolerable
    c. Mitigation Strategy: The team member in charge of configuration management will keep backups of all current team code and documentation on their Mount Union account.

6. End-user input error
    a. Probability: Medium
    b. Impact: Tolerable
    c. Mitigation Strategy: The team will include data validation and error checking within the software that alerts the user of any incorrect data input.

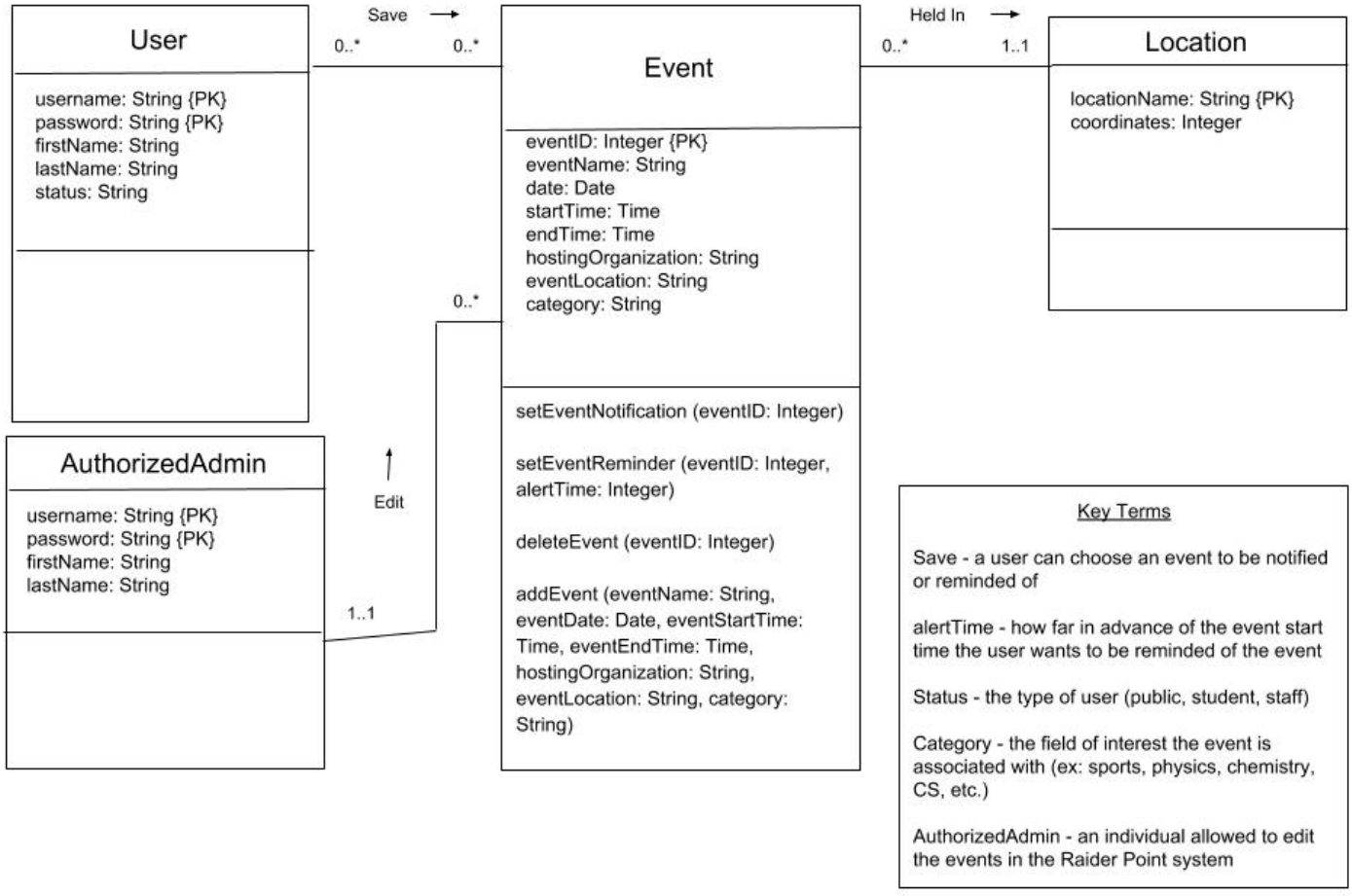
7. Unrealistic schedule
    a. Probability: Low
    b. Impact: Moderate
    c. Mitigation Strategy: The team will only include necessary functionalities and deadlines in the project schedule to ensure all deadlines are met and the team does not get discouraged from an overload of work.

8. User security issues
    a. Probability: Medium
    b. Impact: Moderate
    c. Mitigation Strategy: The team will thoroughly test the application before making it live to ensure using the app does not pose a security threat to users, such as the leaking of sensitive user information.

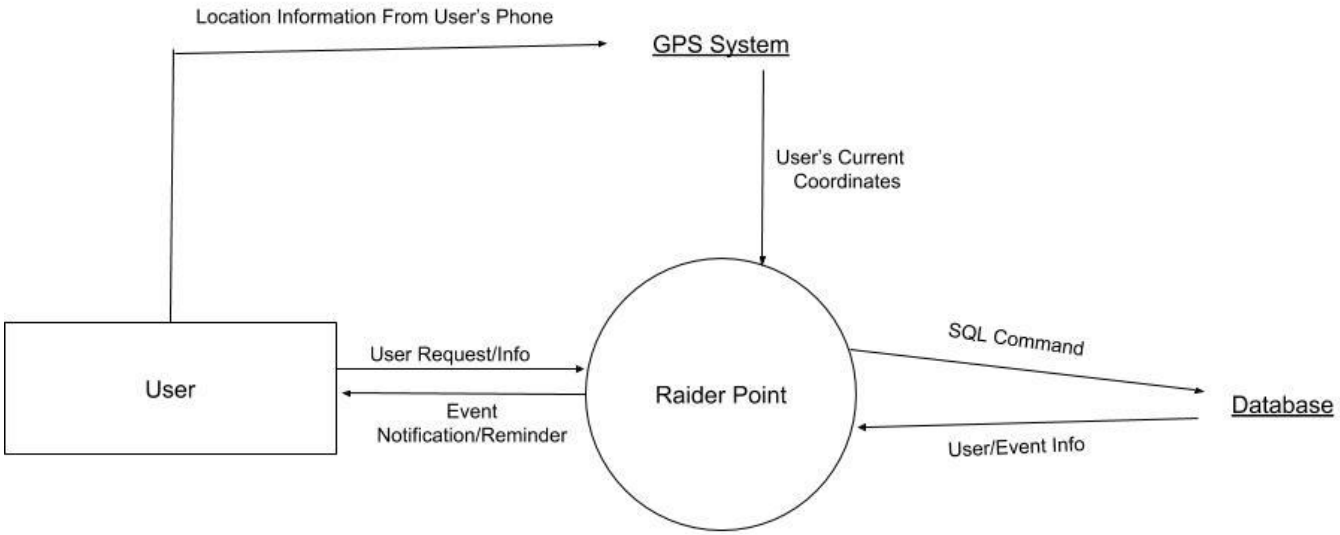
9. Software system becomes out-of-date
    a. Probability: Low
    b. Impact: Serious
    c. Mitigation Strategy: The team will use the most up-to-date software for the Android environment, and alert users that they must have the same software version to successfully run the app.

10. Trouble connecting to the silver server
    a. Probability: Low
    b. Impact: Catastrophic
    c. Mitigation Strategy: The team will be sure to always test and develop the application while on campus. If time allows, the team will also work on moving the database to an off-campus server.

Raider Point UML Diagram



Raider Point Data Flow Diagram
(Context Level-0)



Raider Point Data Flow Diagram
(Level-1)