PROJECT PLAN: UNIVERSAL RAFFLE MANAGEMENT SYSTEM

Version 1.0

Team #1

Katie MacEachern

Max Jennings

Logan Brown

Samuel McKinnon

Ethan Rouse

Dylan Coakley

CSCI 485 Software Design

Dr. Man Lin

January 18th, 2018

TABLE OF CONTENTS

INTRODUCTION

The proposed project, characterized in detail in following documentation, describes the development of software that will serve as a Universal Raffle Management System. It is inspired by the Strait Area Chamber of Commerce's Annual 12 Draws of Christmas and the fundraising efforts of other local charities, and will be designed for use by any organization wishing to raise funds through a raffle. The software will be developed over an approximate ten-week period by six individuals and will require no monetary funds. The project will be comprised of several stages as defined below and which follow the practices of proper software development techniques. These techniques will be used to create a web-based application which can be briefly described as a user interface and underlying database which will accommodate the needs of our potential users. These needs may include but are not limited to: the creation and storage of digital tickets, user accounts for ticket sellers which facilitate the sale of said tickets, administrator accounts which provides access to and editing privileges of ticket and seller information, and the generation of statistics relating to the fundraising campaign. Further project constraints pertaining to functional requirements, non-functional requirements et cetera will be outlined in supplementary documentation. The project will be managed throughout the specified time period by careful monitoring by group members of progress in relation to deliverables set by the overseeing professor and milestones predetermined by group members. Adaptations to the project plan will be conducted as necessary, and any major changes will be reported directly to the overseeing professor for approval.

PROJECT ORGANIZATION

The following describes the way in which the development team is organized, the people involved, and their roles in the team. The information within is subject to change:

Name	Role(s)	Responsibilities
Amanda Mombourquette	• Client	• Provide specifications and requirements for software. • Provide feedback during the development process. • Propose new requirements and/or changes to current rendition of software.

Katie MacEachern	• Team Member • Primary Client Communicator • Functional Requirements (Specification Phase) • User Interface Design (Design Phase) • User Interface Implementation (Development Phase) • Documentation (Development Phase)	• Main line of communication between the team and client. Must communicate client's requests to rest of team. • Provide description of the functionality of the software in a concise manner, as specified by the client. • Design and implementation of a user interface that promotes ease of use, and facilitates all functionalities. • Creating documentation that includes pertinent information about the software, such as a user manual.
Max Jennings	• Team Member • Non-functional Requirements (Specification Phase) • Network & Security Design (Design Phase) • Network & Security Implementation (Development Phase) • Testing (Development Phase)	• Provide description of non-functional requirements that are integral to the system, such as performance and efficiency. • Design and implementation of networking in the system to ensure the software is secure over a network, as well as preventing unauthorized user access. • Testing the system to see if it meets all functionality requirements, as well as having good behaviour when making use of these functionalities.
Logan Brown	• Team Member • Functional Requirements (Specification Phase) • Database Design (Design Phase) • Database Implementation (Development Phase) • Research (Development Phase)	• Provide description of the functionality of the software in a concise manner, as specified by the client. • Design of a relational database schema, along with the actual implementation of the database into the system. • Researching the best ways to implement every part of the system, ensuring good performance and proper software structure.

Samuel McKinnon	• Team Member • Non-functional Requirements (Specification Phase) • Database Design (Design Phase) • Database Implementation (Development Phase) • Testing (Development Phase)	• Provide description of non-functional requirements that are integral to the system, such as performance and efficiency. • Design and implementation of a relational database schema • Testing the system to see if it meets all functionality requirements, and has good behaviour when making use of these functionalities.
Ethan Rouse	• Team Member • Functional Requirements (Specification Phase) • Network & Security Design (Design Phase) • General Programmer (Development Phase) • Research (Development Phase)	• Provide description of the functionality of the software in a concise manner, as specified by the client. • Design of networking in the system, ensuring that the software is secure over a network, as well as preventing unauthorized user access. • Integrating all major components (GUI, database, etc.) together using general logic in programming. • Researching the best ways to implement every part of the system, ensuring good performance and proper software structure.
Dylan Coakley	• Team Leader • User Interface Requirements (Specification Phase) • User Interface Design (Design Phase) • General Programmer (Development Phase) • Documentation (Development Phase)	• Establishing meeting times. • Ensuring that each member has a clear task to perform. • Provide description of the user interface using diagrams showing how the user will interact with the system. • Design of a user interface that promotes ease of use, and facilitates all functionalities. • Integrating all major components (GUI, database, etc.) using general logic in programming. • Creating documentation that includes pertinent information about the software, such as a user manual.

RISK ANALYSIS

Risk	Likelihood	Effects	Reduction Strategy
Key staff become seriously ill, and are unavailable to work on the project	low	serious	If staff become ill, prepare other staff to take over so deliverables are on time
Changes to requirements that require major design modifications are proposed	moderate	serious	Be very specific with requirement descriptions
The database is unable to process as many transactions per second as expected	low	serious	Talk with customer to discuss how many sellers may access the DB to ensure parallelism
The time to develop the software is underestimated	moderate	catastrophic	Create a realistic deliverable schedule and determine which functional requirements are not strictly necessary and could be discarded
The size of the software is underestimated	low	tolerable	Be efficient with code, reuse functions
Loss of key staff (ex. conflict within the group)	low	serious	Establish an approach for conflict resolution
The group disbands (ex. major conflicts)	low	catastrophic	Group bonding activities, establish an approach for conflict resolution, establish a clear protocol should the situation arise
The customer cancels the project	low	insignificant	Produce software as planned, stay on time with deliverable schedule, keep customer up-to-date (and meet their expectations), offer finished product to existing and potential clients
Productivity issues (ex. team member misses a deadline)	moderate	severe	Have an earlier due date within the group before the deliverable deadline to allow a limited grace period. Constantly measure progress to ensure no one falls behind.

HARDWARE & SOFTWARE REQUIREMENTS

Client Side

Due to the nature of the project, and because the project will be web-based, the only software requirement is a JavaScript enabled web browser. Therefore, the application will be able to run on all operating systems.

Additionally, for hardware, there are no strict requirements for the client. Since the application is web based it is accessible on any device that is connected and able to browse the internet.

Team Side

Similarly, to the client side software requirements are minimal. A text editor is required to be able to begin the development process. Also, for assistance with versioning control and for file sharing, the web-based application GitHub will be used.

There are no specific hardware requirements for the Team side. Each Team Member will have a personal computer for development and communication with the other Team Members.

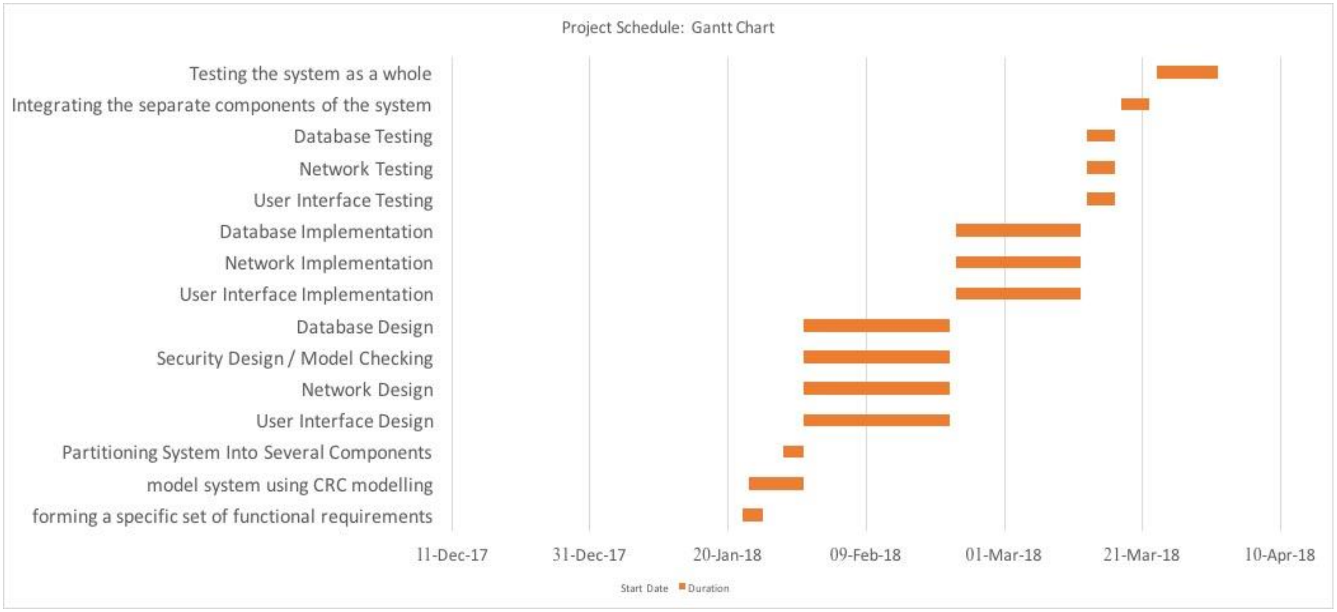
WORK BREAKDOWN

Project Deliverable	**Breakdown of Deliverable**
Specification of Requirements	• Acquire all functional requirements pertaining to the project along with use cases and sub use cases if needed. These use cases include priority level, preconditions, minimal and success guarantees, the trigger, the main success scenario, and the extensions of the use case. • Obtain all non-functional requirements including a unique name, number, priority level, and a clear description for each. All must be quantifiable. • Specify all user interface requirements using diagrams to represent what the interface will look like when it is fully completed. • Create a document containing all the necessary requirements and issues pertaining to each requirement.

Project Design	• Using the requirements obtained for the previous deliverable, make a general design document for the project. • Create an architectural design layout which includes systems, subsystems, and data repositories involved with the project. Alternative architectures and rationale for the decided upon architecture is also required. • A specific design document for the software must be produced. This design includes the objects and classes of the software, the interactions between the objects, and all the underlying methods, attributes, and parameters. • Design of the interface and the general functionality of said interface will be specified. The way the interface looks and how it works must be modeled and described. • Along with all design specifications, the resources required to run the software will be included. • Produce a document that contains the design descriptions and quantifications of required resources for the software.
Development, Testing, and Completed Software	• The software based on the designs and requirements previously mentioned will be fully developed and tested by the team. • Tests on the use cases of the software will be conducted to ensure that all parts of the software are working as intended. The client should be contacted to ensure that they approve of the way the software looks, feels, and acts. • The final product will be a robust version of the designed software that provides all required functionalities with minimal bugs.

PROJECT SCHEDULE

Given the information available, a project schedule was designed. The schedule proposed in this document will be revised regularly as the project takes shape. The project schedule is built around a higher-level waterfall model however, it is expected that other software process models will be incorporated as unit development, testing and integration progresses. The waterfall model works well for this project because the requirements are unlikely to change as the project is developed and are well understood by all team members. The timeline for this project is approximately ten weeks and will be produced by a team of six individuals. The following project schedule proposes how these resources will be organized to produce the web-based application in this time frame. The current project schedule divides the team according to their roles as described in the project organization section of this report; as stated in this section, the roles are subject to change. As many of the system components will be designed and implemented separately, it is expected that teams work together to ensure their components will function correctly when part of the larger system. The following is a Gantt chart of the proposed project schedule:



MONITORING AND REPORTING SCHEDULE

Weekly Member Reports

- Reports that are submitted by each team member every week
- Due before the start of class every Monday for the duration of the project
- Added to the team website

Weekly Team Reports

- Report that is submitted by the Team Leader every week
- Outlines the current status of the project
- Added to the team website

Meeting Minutes

- Written by the Team Leader
- Detailed account of what was discussed during the meetings
- Added to the team website for review by the instructor and for reference as the project progresses

Deliverable Reports

- Used internally to track the individual contributions to each deliverable
- To be submitted prior to the deadline provided by the instructor to allow for feedback between all team members
- Shared within the team, but not posted to the website

Continuing Development Reports

- As development begins, internal documents will be produced by the team members on a weekly basis
- They are to be submitted by the individual(s) who are working on a pre-defined part of development in order to track the efficiency and accuracy of the code that is being produced
- Contains ongoing issues, progress reports, requests for additional labour, and adjustments to internal due dates

Team Leader Reports

- To be submitted bi-weekly
- Explain the current progress of the project
- Discusses upcoming due dates
- Shows the current and updated schedule
- Offers a time for Team Members to volunteer for new jobs within the project