**Tournament Manager**

**Description**

Tournament Manager is a multifaceted software application designed to assist in managing any kind of tournament, from bracket style to double-elimination to round robin and anything in between. It can be used for any sport, including eSports - the team names and players are completely customizable - and the software includes features to manage multiple venues to maximize time efficiency.

There are three primary components to the software:

1. Online registration
2. Bracket Manager
3. Venue Manager

Online Registration is the web-based portion of the software. It contains a form for registration, including fields for team name, spaces for team members, contact information, and email address of the team leader. Once registered, the team information will be added to the database and an email will be sent out with confirmation. Tournament organizers can use the contact information to send emails to team leaders containing important information and changes to the tournament.

Bracket Manager is a local feature which allows the tournament organizer to choose between bracket style and round robin style, with the option for double elimination mode. During the actual tournament, bracket manager will allow the tournament organizer to enter wins and losses, with an optional field for the score. When a match is over, the organizer will enter wins and losses, and the software will move the teams to their appropriate place (e.g. the next level in bracket mode or the next matchup in round robin mode). At any point, the organizer will be able to manually override any decision of the software to resolve any issues.

Venue Manager keeps track of multiple-venue tournaments - when there is more than one playing area or room. This software will work with Bracket Manager to assign rooms to teams depending on their location on the bracket. It will be used to match teams to their opponent with the least amount of venue switching possible. It can be used in Traditional Mode, where teams will face each other in the same venue, or in eSport Mode, where teams facing each other must be in different rooms. Like Bracket Manager, Venue Manager can be operated in manual mode to override decisions of the program.

The project will be primarily written in C# for Windows PC. The registration module will be made using ASP.NET, and the project will use SQL for the database backend.

**Timetable, milestones, high-level planning**

Tournament management consists of three interdependent modules, all of which can be developed separately up to a point. The first component developed will be the database, which will be the common medium for data between the three components. Once that is completed, the registration, bracket, and venue managers can be developed separately. The registration component only needs to be able to interact with the database, so it requires the least amount of integration testing. The bracket manager and venue manager, while listed here separately, will actually be part of the same executable, so will require extensive integration testing and must be developed together as much as possible. The project can be divided as follows:

* Phase 1: Requirements
* Phase 2: Design
* Phase 3: Database
  + Code SQL database
* Phase 4: Primary coding
  + Registration
  + Local program
    - Registration
    - Bracket and Venue
* Phase 5: Integration
  + Test registration with database
  + Test bracket and venue with database
* Phase 6: Final Testing
* Phase 7: Deployment

Division of work:

Since the registration module and the database are so interconnected, it would be recommended for the same team member to design and code both systems. The bracket and venue require extensive GUI design and non-trivial engine coding, as well as being closely integrated to each other, so the remaining team members should work on both in tandem. In the event that one module is finished before the others, the team member(s) working on that module will be reassigned to the other module(s).

Manager Report (progress, team structure, member names and roles, SVN policies)

**Risk Management**

**Identify:**

1. Time Management: Time management is going to be key in any project and presents the risk of letting the project getting behind schedule if time is mismanaged.

2. Feature Planning: One large part about any project is planning, obviously. However, one part worth noting in this section is the planning of features. We, as a group, and especially the coders, need to agree on a slate of features in both how they’re made and what they do. This allows us to get to work on the project/features without having to argue about what parts we do/don’t want to include.

3. Identification of Strengths: One key part of the project, that can lead to problems if not accurately completed, is the self-assessment of one’s strengths and weaknesses, and the distribution of duties based upon those assessments.

4. Project Management: An imperative part of software engineering is to be clear and concise about what parts of the projects are whose responsibilities. Risk is presented if people try to do too much or parts don’t get done/get done twice due to mismanagement of duties.

5. Collaboration: An idea stemming from project management, it is necessary as a team to collaborate effectively and efficiently on our project in order to maximize the efficiency of time spent individually. If we are able to work on goals that mesh together, rather than working on parts at random, we can cut down on time spent debugging and reworking code.

**Monitor:**

1. Keeping commits logged (through GIT) will help to prevent time mismanagement, in the vein that we will be able to make accurate predictions of how much time we’ll need to complete milestones, and we’ll be able to stay ahead of the curve.

2. Preventing feature planning from becoming a problem breaks down into making sure that the project is planned out ahead of time, or that each member comes to a consensus over the general trajectory of the project with only fine details needing to be worked out later. Eliminate the question marks early and it eliminates them down the road.

3. A lot of risk management in this regard breaks down into everybody being honest in their strengths. Assuming that they are, there isn’t a huge risk, but it does place faith in everybody’s self-evaluation.

4. A lot of this falls on the shoulders of whoever is managing the distribution of duties, and the risk is twofold:

* · #3 must be completed with truth, else the duties are being distributed based upon false strengths/weaknesses.
* · The person managing the responsibilities must distribute them in a way that caters to strengths but doesn’t place too much work upon any one person (efficiency is key).

5. Collaboration really falls on the shoulders of our team members being willing and able to meet together, either in person or (at the very least) online in order to come to common understandings of where the project is, what is left to be done and how far we want to go/what we want to accomplish before meeting again. The risk involved is that we might not be as good at evaluating where we’re at if we don’t setup meetings. It’s possible, but meetings will go far in eliminating that risk.

**Evaluation:**

One important step in risk management will be our ability to evaluate ourselves and our project as the semester goes on, making sure that we have been accurate in our assessments, stay on schedule and effectively and efficiently continue to progress on the project. Personal evaluation will help us keep up on our individual responsibilities so that we don’t drag down the project, and team evaluation will help us stay on schedule as a whole. Plus, evaluation of our team and project (by ourselves) may help us improve, and at least sustain efficiency.

**Contingency Plan:**

In a project like this, thankfully a client isn’t paying us for the project. That would make a contingency plan wholly more important and necessarily more extensive. For this project, a contingency plan really simply boils down to being stringent on having meetings and being self-disciplining so that we never get into a situation where we feel the need to break out a plan that insinuates panic. If we take our time, plan accordingly and make sure to not stray from said plan, the best contingency plan would be proper preparation.