Requirements:

1. Tracks games played and their outcome (who won).
2. Multiple competitors play in the tournament
3. Creates a tournament plan (who plays in what order).
4. Schedules games
5. A single loss eliminates a player.
6. The last player standing is the winner.

Questions:

1. How many players will the tournament handle? Is it variable?

The application should be able to handle a variable number of players in a tournament.

1. If a tournament has less than the full complement of players, how do we handle it?

A tournament with less than the perfect number (a multiple of 2, so 4, 8, 16, 32, etc.) should add in “byes”. Basically, certain people selected at random get to skip the first round and act as if they won.

1. Should the ordering of who plays each other be random or ordered by input order?

The ordering of the tournament should be random.

1. Should we schedule the game or are they just played whenever?

The games should be played in whatever order and whenever the players want to play them.

1. If the games are scheduled, how does the system know when to schedule games for?

They are not scheduled so we do not care.

1. If the games are played whenever, can a game from the second round be played before the first round is complete?

No. Each round should be fully completed before the next round is displayed.

1. Does the system need to store a score of some kind or just who won?

Storing a simple score would be nice. Just a number for each player. That way, the tracker can be flexible enough to handle a checkers tournament (the winner would have a 1 and the loser a 0) or a basketball tournament.

1. What type of front-end should this system have (form, webpage, app, etc.)?

The system should be a desktop system for now, but down the road we might want to turn it into an app or a website (long-term goal).

1. Where will the data be stored?

Ideally, the data should be stored in a Microsoft SQL database but please put in an option to store a text file instead.

1. Will this system handle entry fees, prizes, or other payouts?

Yes. The tournament should have the option of charging an entry fee. Prizes should also be an option, where the tournament administrator chooses how much money to award a variable number of places/ The total cash amount should not exceed the income from the tournament. A percentage-based system would also be nice to specify.

1. What type of reporting is needed?

A simple report specifying the outcome of the games per round as well as a report that specifies who won and how much they won. These can just be displayed on a form or they can be emailed to tournament competitors and the administrator.

1. Who can fill in the results of a game?

Anyone using the application should be able to fill in the game scores.

1. Are there varying levels of access?

No. The only method of varied access is if the competitors are allowed into the application and instead, they do everything via email.

1. Should this system contact users about upcoming games?

**Yes, the system should email users that they are due to play in a round as well as who they are scheduled to play.**

1. Is each player on their own or can teams use this tournament tracker?

The tournament tracker should be able to handle the addition of other members. All members should be treated as equals in that they all get tournament emails. Teams should also be able to name their team.

**Big Picture Design**

**Structure:** Windows Forms application and Class Library

**Data:** SQL and/or Text file (for storing information)

**Users:** One at a time on one application

(This puts boundaries around you and keeps you on track to stick to this design)

**Key Concepts**

* Email
* SQL
* Custom Events
* Error Handling
* Interfaces
* Random Ordering
* Texting (Alert people of upcoming events)

**Mapping the Data**

**Team (class)**

* **properties**
* TeamMembers (List<Person>)
* TeamName (string) (property)

Person (class)

* properties
* FirstName (string)
* LastName (string)
* EmailAddress (string)
* CellphoneNumber (string)

Tournament (class)

* TournamentName (string)
* EntryFee (decimal)
* EnteredTeams (List<Team>)
* Prizes (List<Prize>)
* Rounds (Lists<List<Matchup>>)

**Prize**

* PlaceNumber (int)
* PlaceNumber (string)
* PrizeAmount (decimal)

**Matchup**

* Entries (List<MatchupEntry>)
* Winner (Team)
* MatchupRound (int)

**MatchupEntry**

* TeamCompeting (Team)
* Score (double)
* ParentMatchup (Matchup)

**Microsoft SQL Server Management Studio Notes:**

Tables needed:

* MatchupEntries
* Matchups
* People
* Prizes
* TeamMembers
* Teams
* TournamentEntries
* TournamentPrizes
* Tournament

**Stored Procedures needed:**

* MatchupEntries\_GetByMatchup
* Matchups\_GetByTournament
* People\_GetAll
* Prizes\_GetByTournament
* Team\_GetByTournament
* TeamMembers\_GetByTeam
* Tournaments\_GetAll

**File Cleanup**

* Created separate files for DataAccess and Models
  + DataAccess contains Models contains

-IDataConnection -MatchupEntryModel

-SqlConnector -MatchupModel

-TextConnector -PersonModel

-PrizeModel

-TeamModel

-TournamentModel

**Connection to SQL**

* **Using Dapper for connection**
* **Add reference to Dapper**
* **TrackerLibrary is just a DLL so it will is the app.config file from the user interface**
* **Added App.config file to TrackerUI**
* **Add connection string to App.config (within TrackerUI)**
* **Added System.Configuration.ConfigurationManager Nuget package to TrackerLibrary.**

// create new IDbconnection, fill connection with System.Data.SqlClient

using (IDbConnection connection = new System.Data.SqlClient.SqlConnection(GlobalConfig.CnnString("Tournaments")))

{

}

* Create stored procedure in SQL database
* Added settings property to TrackUI
  + Added setting for connection to SQL Server.

Graphical user interface, text, application, email

Description automatically generated

**Graphical user interface, text, application, email

Description automatically generated**

- After adding SQL server connection the App.config file was updated with these settings.

* **Graphical user interface, text

  Description automatically generated**
* The method CnnString in GlobalConfig.cs is called in SqlConnector.cs to make the connection to the SQL Server.

**Text

Description automatically generated**

**Features to Add**

Add in text box to all forms that says “Thank you, you’re entry has been received.

**Bugs**

* System.Data.SqlClient.SqlException:
* In SqlConnector.cs

Found when connecting PersonModel to stored procedure dbo.spPeople\_GetAll in SQL Server

A screenshot of a computer

Description automatically generated

* Fixed the issue by altering the stored procedure

Graphical user interface, text, application

Description automatically generated

* Add Member button not working as expected
  + Fix – in properties for Add Member Button changed events -action – click to addMemberButton\_Click