Assignment #2: Interactive Visualization Exercise

1. Plan of action

From the dataset, the first intuition that came to mind was to represent all the tournaments at a high-level and design filters and drilldowns to get more granular details. But at the same time, I didn't want the user to lose sight of the high-level context when analyzing finer details. To maximize screen estate and not lose the holistic sense, I decided to place the high-level chart on the left of the screen and any drilldowns would be on the right.

2. Missing data

There was a total of 586 rows with missing metrics spread across various columns. So for each row, I get the player names and average the stats from the matches between the same players (which have data) to populate empty columns. If they've never had a match, I'd do the same operation except it'd be on their matches with other players. This ensured every cell was populated with near-accurate data. The Python code is present in filler.ipynb.

3. Design and filters

The Viz dashboard has 3 components:

- **Filters**
  We use two filters for the assignment:
  - **Year filter:** Select any year and the tournament details for that year will be displayed in the high-level chart.
  - **Stage filter:** This select box filters out all the matches and players who haven't made it to that stage. For instance, if we pick *'Fourth round'* as the stage, it'll only show matches and players from the fourth round till the final round.
- **High level overview (arcs and chords)**
  This is the first chart that's loaded on the page. For this part, I used a chord diagram since a circular layout can convey a lot of information with minimal screen estate. All the players are represented as arcs along the circumference of the circle and assigned a random color. The size of the arc gives us an idea of how well the player has fared through the tournament. There are overlapping chords connecting each player arc inside and they have the following properties.
  - A chord between player arcs implies there was a match between them at some point in the tournament.
  - The chords overlap but we can focus on the ones we want by hovering which fades the other chords.
  - The color of the chord indicates who won and the varying thickness of the chord from one arc to the other signifies the gap in points. For instance, If Federer( ) won against Nadal( ) in just 3 sets [6-0, 6-1, 6-3], the color of the chord would be and it would be thicker near Federer's arc and thinner towards Nadal.
- **Player overview**
  Let's say a user has filtered the tournament to show data from the Fourth round in the AUS open 2012 but still wants to get some insight into how some of the players got to that round. Instead of adjusting the filters back and forth to see previous stages, we can simply hover over the player name or arc and on the right, we get a timeline view of his previous matches and the points spread.
- **Match overview**
  Since the chords represent the matches between the players(arcs), simply hover over the chords inside the circle and you get a spider chart view that shows the stats(firstPointWon, secPointWon, break, return, net) between the two players during the match along with stage, winner and scoreboard.

4. Execution

Since we use d3.csv to ingest the data, it'll run into CORS issues if index.html is accessed directly. We need to run a http server inside the folder directory. [`python -m SimpleHTTPServer` **`or`** `python -m http.server`]

5. Conclusion

From the design section, the transition tree flow of the interactive visualization is as follows:

Tournament (Filtered by year and stage)
|___Players (Hover to get timeline)
|___Matches (Hover to get match stats)

Unlike traditional charts which destroy high level visualization to get to the drilldown, this viz dashboard enables a user to seamlessly transition from the tournament to the players to individual matches without losing sight of the bigger picture.