Functional requirements.

The software must be able to:

REQ 1: Receive data either through database or using the graphical interface entering the required information.

- Req1.1: Allow user to create a player using the graphical interface.
- Req1.2: Read the database.

REQ 2: Create players, using the input data.

- Req2.1: Give to the player a name.
- Req2.2: Give to the player an age.
- Req2.3: Give to the player a team.
- Req2.4: Give to the player a points per game field.
- Req2.5: Give to the player a rebounds per game field.
- Req2.6: Give to the player a assists per game field.
- Req2.7: Give to the player a steals per game field.
- Req2.8: Give to the player a blocks per game field.

REQ 3: Manage the big amount data.

- Req3.1: Allow user to add a player.
- Req3.2: Allow user to delete a player.
- Req3.3: Allow user to update the players statistics information.

REQ 4: Allow user to ask for specific type of statistic players information giving to four of five types of information an index which means that type of information must be returned faster.

- Give to four of five types of player information an index.
- Collect all the players specific data and return it.
- Return the types with index specific information faster, using a complexity of O(log n).
- Allow user to ask about points per game.
- Allow user to ask about assists per game.
- Allow user to ask about steals per game.
- Allow user to ask about blocks per game.
- Allow user to ask about rebounds per game.

REQ5: Create balanced binary tree saving into it the specific statistic information with index, then justify the BBT usage by comparing the time taken by the search of a specific type of information with index and specific type of information without index.

- Req5.1: Save the players data.
- Req5.2: Show user how long it takes to ask for an information without index.
- Req5.3: Show user how long it takes asking for information with index