Name of Analyst :- Shagun Mahamuni.
Emp id :- 1514

Task B :- How the UI interacts with a service API to correctly view, update, create, and remove schedule entries using the “Test Driven Development” process.

- **Scenario** 1: if A registered and authorized coach wants to create a schedule event for “2020- 2021” season

- **Steps/Validation**:
  1. Coach selects “Add event “and the “Season” example: “2020 –2021” in the schedule. Must enter the “Event Type, Game Type, Location, Select Opponent, Date and Time.
  2. **Post**:

     As coach hits the save button, the data is sent in the response body of the request and then sent to the server in the form of JSON.

     {
     gameid: "1234567" ,sqlId: "1234567", date: "2020-01-01T19h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ]
     }
  3. **RESPONSE**:
     Is game properly stored in database ? Result = true
     HTTP Code – Success ; 200 OK Status Code
     Response –
     {
     Show the added game, error: “NULL”
     }
  4. **Validate that** game is stored into Database.

- **Scenario** 2: if the coach is authorized and wants to add a game in Season and there is Internal Server Error.

- **Steps/Validation**:
  1. Coach selects “Add event “and the “Season” example: “2020 –2021” in the schedule. Must enter the “Event Type, Game Type, Location, Select Opponent, Date and Time.
  2. **Post**:

     As coach hits the save button, the data is sent in the response body of the request and then sent to the server in the form of JSON.

     {
     gameid: "1234567" ,sqlId: "1234567", date: "2020-01-01T19h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ]

}

3. **RESPONSE**:
   Is game properly stored in database ? Result = False
   HTTP Code - Internal Server Error; 500 Status Code
   Response –
   {
   Error: “Internal Server Error”
   }
4. **Validate that** game is stored into Database.

- **Scenario** 3: if the coach is un-authorized and wants to add a game in Season.

- **Steps/Validation**:

1. Coach selects the “Add Event” “Season” example: “2020 –2021” in the schedule. Must enter the “Event Type, Game Type, Location, Select Opponent, Date and Time.
2. **RESPONSE**:
   Is coach authorized in database ? Result = False
   HTTP Code - Unauthorized; 401 Status Code
   Response –
   {
   error: “unauthorized Error”
   }
3. **Validate that** coach is authorized into Database.

- **Scenario** 4: if the coach is authorized and wants to View a game from a Season.

- **Steps/Validation**:

1. Coach clicks on the season “2020-2021” to view the schedule entries.
2. **Get**:

   The GET method Is used to retrieve data from server at the specific resource.

   {

   gameid: "1234567" ,sqlId: "1234567", date: "2020-01-01T19h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ]

   }

3. **RESPONSE**:
   Is game properly stored in database ? Result = True
   HTTP Code -Success; 200 OK Status Code
   Response –
   {

   gameid: "1234567" ,sqlId: "1234567", date: "2020-01-02T20h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ], error: “NULL”

   }

4. **Validate that** game is stored into Database and shown to coach.

- **Scenario** 5: A registered and authorized coach wants to update the schedule entry in the season “2020-2021”.

- **Steps/Validation**:
  1. Coach clicks on the particular schedule event he wants to update in the season “2020-2021.
  2. **PUT :**

     The PUT method is used to send data to API to update or create a resource. We are making changes in the date and time of a particular game.

     {

     gameid: "1234567" ,sqlId: "1234567", date: "2020-01-02T20h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ]

     }

  3. **RESPONSE**:
     Is game updates properly stored in database ? Result = True
     HTTP Code -Success; 200 OK Status Code
     Response –
     {

     Entryid:”1234567”,”updategame”: {

     gameid: "1234567" ,sqlId: "1234567", date: "2020-01-02T20h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ], error: “NULL”}

     }
  4. **Validate that** game updates is stored into Database.

- **Scenario** 6**:** A registered and authorized coach wants to delete the schedule entry in the season “2020-2021”.

- **Steps/Validation**:
  1. The coach clicks on the particular event and delete the event.

  2. DELETE :-

  Delete method delete the resources at the specified URL.

  {

  gameid: "1234567" ,sqlId: "1234567", date: "2020-01-01T19h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ]
  }

3. **RESPONSE**:
   Is game deleted properly from database ? Result = True
   HTTP Code - Success ; 200 Status Code.

4. **Validate that** game is properly Deleted from Database.

- **Scenario** 7: A registered and authorized coach wants to get the deleted schedule entry.

- **Steps/Validation**:

1. **GET :-** GET method is called to get the requested schedule entry

   {

   gameid: "1234567" ,sqlId: "1234567", date: "2020-01-01T19h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ]

   }

2. **RESPONSE**:
   Is game present in database ? Result = False
   HTTP Code - Forbidden; 403 OK Status Code
   Response –
   {
   error: “Forbidden”
   }

- Scenario 8: if A registered and authorized coach wants to create a schedule event for “2020-2021” season and if the date, time are not in proper order.
- **Steps/Validation**:
  1. Coach selects “Add event “and the “Season” example: “2020 –2021” in the schedule. Must enter the “Event Type, Game Type, Location, Select Opponent, Date and Time(not in order).
  2. **Post**:

     As coach hits the save button, the data is sent in the response body of the request and then sent to the server in the form of JSON.

     {
     gameid: "1234567" ,sqlId: "1234567", date: "2020-032-01T32h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ]
     }
  3. **RESPONSE**:
     Is game properly stored in database ? Result = true
     HTTP Code – Forbidden ; 403 OK Status Code

Response –
{
error: “Forbidden”
}

4. **Validate that** game is stored into Database are in proper order.

- Scenario 9: A registered and authorized coach wants to access opponent entry which is not present into database.
- **Steps/Validation**:
  1. **GET :-** The GET method Is used to retrieve data from server at the specific resource.
     {
     gameid: "1234567" ,sqlId: "1234567", date: "2020-01-01T19h00h00", opponent: "TestOpponent", opponentId: "123456", isHome: true, gameType: 0, categories: [ ] }
  2. **RESPONSE**:
     Returns with status code of 404 Resources not found
     {
     Error :- “Not found”
     }
