## CLASS DIAGRAM



## DATA DICTIONARY

|  |  |  |  |
| --- | --- | --- | --- |
| **MembershipLevel** | | | |
| <<PK>> | MembershipLevelID | :Int | {identity, seed = 1, increment = 1} |
|  | Description | :String | {Unicode, size = 20} |
|  | Active | :Int | =1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **User** | | | |
| <<PK>> | UserID | :Int | {identity, seed = 1, increment = 1} |
|  | UserEmail | :Varchar | {Unicode, size = 50} |
|  | Password | :Varchar | {Unicode, size = 50} |
|  | Salt | :Varchar | {Unicode, size = 50} |
|  | FirstName | :Varchar | {Unicode, size = 50} |
|  | LastName | :Varchar | {Unicode, size = 50} |
|  | Phone | :Varchar | {Unicode, size = 255} |
| <<FK>> | MembershipLevelID | :Int | {reference = MembershipLevel} |
|  | Active | :Int | =1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Course** | | | |
| <<PK>> | CourseID | :Int | {identity, seed = 1, increment = 1} |
|  | CourseName | :Varchar | {Unicode, size = 50} |
|  | ParHole1 | :Int |  |
|  | ParHole2 | :Int |  |
|  | ParHole3 | :Int |  |
|  | ParHole4 | :Int |  |
|  | ParHole5 | :Int |  |
|  | ParHole6 | :Int |  |
|  | ParHole7 | :Int |  |
|  | ParHole8 | :Int |  |
|  | ParHole9 | :Int |  |
|  | ParHole10 | :Int |  |
|  | ParHole11 | :Int |  |
|  | ParHole12 | :Int |  |
|  | ParHole13 | :Int |  |
|  | ParHole14 | :Int |  |
|  | ParHole15 | :Int |  |
|  | ParHole16 | :Int |  |
|  | ParHole17 | :Int |  |
|  | ParHole18 | :Int |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Reservation** | | | |
| <<PK>> | ReservationID | :Int | {identity, seed = 1, increment = 1} |
| <<FK>> | UserID | :Int | {reference = User} |
| <<FK>> | CourseID | :Int | {reference = Course} |
|  | ReservedTime | :DateTime |  |
|  | NumberHoles | :Int |  |
|  | NumberCarts | :Int |  |
|  | Player2 | :Varchar | {Unicode, size = 50} |
|  | Player3 | :Varchar | {Unicode, size = 50} |
|  | Player4 | :Varchar | {Unicode, size = 50} |

|  |  |  |  |
| --- | --- | --- | --- |
| **StandingReservation** | | | |
| <<PK>> | StandingReservationID | :Int | {identity, seed = 1, increment = 1} |
| <<FK>> | UserID | :Int | {reference = User} |
| <<FK>> | CourseID | :Int | {reference = Course} |
|  | ReservedTime | :DateTime |  |
|  | EndTime | :DateTime |  |
|  | NumberHoles | :Int |  |
|  | NumberCarts | :Int |  |
|  | Player2 | :Varchar | {Unicode, size = 255} |
|  | Player3 | :Varchar | {Unicode, size = 255} |
|  | Player4 | :Varchar | {Unicode, size = 255} |
|  | Approved | :Int |  |
|  | Active | :Int |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Score** | | | |
| <<PK>> | ScoreID | :Int | {identity, seed = 1, increment = 1} |
| <<FK>> | ReservationID | :Int | {reference = Course} |
|  | UserEmail | :Varchar | {Unicode, size = 255} |
|  | ScoreHole1 | :Int |  |
|  | ScoreHole2 | :Int |  |
|  | ScoreHole3 | :Int |  |
|  | ScoreHole4 | :Int |  |
|  | ScoreHole5 | :Int |  |
|  | ScoreHole6 | :Int |  |
|  | ScoreHole7 | :Int |  |
|  | ScoreHole8 | :Int |  |
|  | ScoreHole9 | :Int |  |
|  | ScoreHole10 | :Int |  |
|  | ScoreHole11 | :Int |  |
|  | ScoreHole12 | :Int |  |
|  | ScoreHole13 | :Int |  |
|  | ScoreHole14 | :Int |  |
|  | ScoreHole15 | :Int |  |
|  | ScoreHole16 | :Int |  |
|  | ScoreHole17 | :Int |  |
|  | ScoreHole18 | :Int |  |

## USE CASE MODEL



USE CASE : Login

Brief Description: The purpose of this use case is to allow a User to login

Flow of Events:

* Basic Flow:
  1. Actor enters username and password and click, “Submit” button
  2. System fetches user information from database and verifies the input, then sends users to the Default.aspx page
* Primary Scenario:
  1. Johnny Appleseed, a user, fills in his username and password in the username and password fields
  2. System checks Johnny’s input against what is in the database, and redirects him to the Default.aspx page
* Alternative Flow:

2a. If input does not match database information, user will be stopped from going to the default page and will be prompted to verify their username and password

Preconditions : Actor is not logged in

Post conditions: Actor is logged in

Sequence Diagram:

USE CASE : Create Reservation

Brief Description: The purpose of this use case is to allow a User to create a reservation

Flow of Events:

* Basic Flow:

1. User selects a day, picks a time, chooses a course, and selects the number of holes, numbers of carts, and number of players for the reservation
2. System sends the reservation to the database and redirects User to ViewReservations.aspx page

* Primary Scenario:
  1. Johnny Appleseed, a user, picks a time, chooses a course, and selects the number of holes, numbers of carts, and number of players for the reservation
  2. System sends the input to the database and redirects Johnny to the ViewReservations.aspx page
* Alternative Flow:

1a. If User has MembershipLevel “ADMIN”, User will be required to search for a member

Preconditions : Reservation does not exist, User is logged in

Post conditions: Reservation exists, user is on CreateReservation.aspx page

Sequence Diagram:

USE CASE : Update Reservation

Brief Description: The purpose of this use case is to allow a User to update an existing reservation

Flow of Events:

* Basic Flow:

1. User selects the reservation for edit in the ViewReservations.aspx page
2. System fetches the UpdateReservation.aspx, with the chosen reservation info auto-filled into the form
3. User selects a day, picks a time, chooses a course, and selects the number of holes, numbers of carts, and number of players for the reservation
4. System sends the new information to the database and redirects User to ViewReservations.aspx page

* Primary Scenario:
  1. Johnny Appleseed, a user, selects a reservation he wants to edit in the ViewReservations.aspx page
  2. Johnny is sent to the UpdateReservation.aspx page, with the information from the reservation he chose auto-filled into the form
  3. Johnny picks a time, chooses a course, and selects the number of holes, numbers of carts, and number of players for the reservation
  4. System sends the input to the database and redirects Johnny to the ViewReservations.aspx page
* Alternative Flow:

1a. If User has MembershipLevel “ADMIN”, User will be required to search for a member

Preconditions : Reservation exists, User is logged in, user is on ViewReservations.aspx page

Post conditions: Reservation is updated to reflect form input

Sequence Diagram:

USE CASE : Create Standing Reservation

Brief Description: The purpose of this use case is to allow a User to create a standing reservation

Flow of Events:

* Basic Flow:

1. User selects a day, picks a time, chooses and end date, chooses a course, and selects the number of holes, numbers of carts, and fills out the names of the players he will be playing with
2. System sends the standing reservation information to the database and redirects User to ViewReservations.aspx page

* Primary Scenario:
  1. Johnny Appleseed, a user, picks a day, chooses a time, selected an end day, chooses a course, and selects the number of holes, numbers of carts, and fills in the names of the players he will be playing with
  2. System sends the input to the database and redirects Johnny to the ViewReservations.aspx page
* Alternative Flow:

1a. If User has MembershipLevel “ADMIN”, User will be required to search for a member

Preconditions : Standing reservation does not exist, User is logged in, user has MembershipLevel “Gold” or “ADMIN”, user is on CreateStandingReservation.aspx page

Post conditions: Standing reservation exists

Sequence Diagram:

USE CASE : Update Standing Reservation

Brief Description: The purpose of this use case is to allow a User to update an existing standing reservation

Flow of Events:

* Basic Flow:

1. User selects a StandingReservation for edit in the ViewStandingReservations.aspx page
2. System fetches the UpdateStandingReservation.aspx, with the selected standing reservation information auto-filled into the form fields
3. User selects a day, picks a time, chooses an end date, chooses a course, and selects the number of holes, numbers of carts, and fills out the names of the other players that will be playing
4. System sends the standing reservation information to the database and redirects User to ViewStandingReservations.aspx page

* Primary Scenario:
  1. Johnny Appleseed, a user, selects the standing reservation he wishes to edit in the ViewStandingReservations.aspx page
  2. System fetches UpdateStandingReservation.aspx page with chosen standing reservation information auto-filled into form fields
  3. Johnny picks a day, chooses a time, selected an end day, chooses a course, and selects the number of holes, numbers of carts, and fills in the names of the players he will be playing with
  4. System sends the input to the database and redirects Johnny to the ViewReservations.aspx page
* Alternative Flow:

Preconditions : Standing reservation exists, User is logged in, user has MembershipLevel “ADMIN”,

user is on ViewStandingReservations.aspx page

Post conditions: Standing reservation is updated to reflect input

Sequence Diagram:

USE CASE : Approve Standing Reservation

Brief Description: The purpose of this use case is to allow a User to update an existing standing reservation

Flow of Events:

* Basic Flow:

1. User selects a StandingReservation for edit in the ViewStandingReservations.aspx page
2. System fetches the UpdateStandingReservation.aspx, with the selected standing reservation information auto-filled into the form fields
3. User clicks “Approve” button
4. System sends update information to the database and redirects user to ViewStandingReservations.aspx page.

* Primary Scenario:
  1. Johnny Appleseed, a user, selects the standing reservation he wishes to edit in the ViewStandingReservations.aspx page
  2. System fetches UpdateStandingReservation.aspx page with chosen standing reservation information auto-filled into form fields
  3. Johnny clicks “Approve” button
  4. System sends the update to the database and redirects Johnny to the ViewReservations.aspx page
* Alternative Flow:

Preconditions : Standing reservation exists, User is logged in, user has MembershipLevel “ADMIN”, user is on ViewStandingReservations.aspx page

Post conditions: Standing reservation is approved as a reservation

Sequence Diagram:

USE CASE : Enter Score

Brief Description: The purpose of this use case is to allow a User to submit a score

Flow of Events:

* Basic Flow:

1. User selects a course, and fills out his scores for each hole
2. System sends the score to the database and redirects to the ViewScores.aspx page

* Primary Scenario:
  1. Johnny Appleseed, a user, selects the course and fills out the scores for each hole
  2. System sends the score to the database, Johnny is redirected to the ViewScores.aspx page
* Alternative Flow:

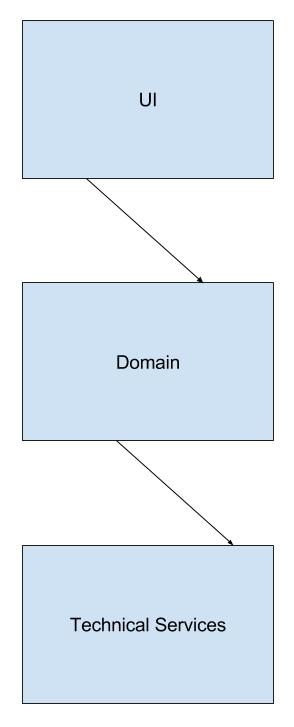
1a. If user has membership level “ADMIN”, user will be required to search for a member first.

Preconditions : Score does not exist, User is logged in, user has MembershipLevel “ADMIN” or “Gold” or “Silver”, user is on SubmitScore.aspx page

Post conditions: Standing reservation is approved as a reservation

Sequence Diagram:

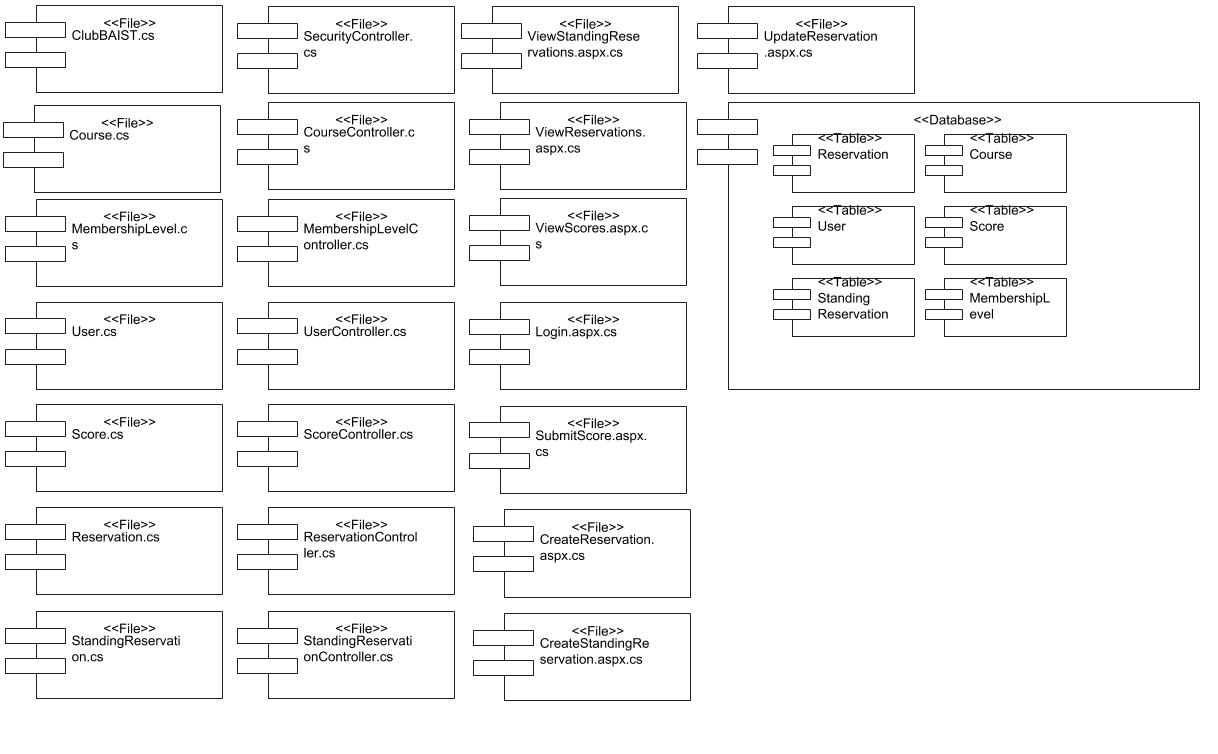
## LOGICAL VIEW



DESIGN CLASSES



## IMPLEMENTATION VIEW



## TEST SCRIPTS

|  |  |
| --- | --- |
| Author | Kyle Pedersen |
| Creation Date | 2018-03-01 |
| Scenario | Create Reservation |
| Test Type | UAT |
| Version | 2.0 |
| Scenario | 001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Result # | Steps | Result | Pass/Failed | Comments |
| TR #001 | 1. Navigate to CreateReservation.aspx 2. Click March 9, 2018 3. Click 07:00:00 4. Click “Front 9” 5. Click “2 Carts” 6. Click “2 players” 7. Click “Submit” button | Reservation is added, page is redirected to ViewReservations.aspx | pass |  |

|  |  |
| --- | --- |
| Author | Kyle Pedersen |
| Creation Date | 2018-03-01 |
| Scenario | Update Reservation |
| Test Type | UAT |
| Version | 2.0 |
| Scenario | 001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Result # | Steps | Result | Pass/Failed | Comments |
| TR #002 | 1. Navigate to ViewReservations.aspx 2. Click ”Edit” button next to March 9, 2018 reservation 3. Click March 10, 2018 4. Click 07:30:00 5. Click 18 Holes 6. Click 1 Cart 7. Click 4 Players | page is redirected to ViewReservations.aspx, reservation that was clicked for edit is changed | pass |  |

|  |  |
| --- | --- |
| Author | Kyle Pedersen |
| Creation Date | 2018-03-01 |
| Scenario | Create Standing Reservation |
| Test Type | UAT |
| Version | 2.0 |
| Scenario | 001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Result # | Steps | Result | Pass/Failed | Comments |
| TR #003 | 1. Navigate to CreateStandingReservaion.aspx 2. Click March 13, 2018 3. Click 13:00:00 4. Click 18 Holes 5. Click 1 Cart 6. Type in “Second” 7. Type in “Third” 8. Type in “Fourth ” 9. Click “Submit” | page is redirected to ViewStandingReservations.aspx, reservation that was entered is there | pass |  |

|  |  |
| --- | --- |
| Author | Kyle Pedersen |
| Creation Date | 2018-03-01 |
| Scenario | Update Standing Reservation |
| Test Type | UAT |
| Version | 2.0 |
| Scenario | 001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Result # | Steps | Result | Pass/Failed | Comments |
| TR #004 | 1. Navigate to ViewStandingReservations.aspx 2. Click “Edit button” next to March 13 reservation 3. Click March 14, 2018 4. Click 14:00:00 5. Click Front 9 6. Click 2 Carts 7. Type in “Second Man” 8. Type in “Third Guy” 9. Type in “Fourth Lady” 10. Click “Submit” | page is redirected to ViewStandingReservations.aspx, reservation that was updated is changed | pass |  |

|  |  |
| --- | --- |
| Author | Kyle Pedersen |
| Creation Date | 2018-03-01 |
| Scenario | Update Standing Reservation |
| Test Type | UAT |
| Version | 2.0 |
| Scenario | 001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Result # | Steps | Result | Pass/Failed | Comments |
| TR #005 | 1. Navigate to ViewStandingReservations.aspx 2. Click “Edit button” next to March 13 reservation 3. Click March 14, 2018 4. Click 14:00:00 5. Click Front 9 6. Click 2 Carts 7. Type in “Second Man” 8. Type in “Third Guy” 9. Type in “Fourth Lady” 10. Click “Submit” | page is redirected to ViewStandingReservations.aspx, reservation that was updated is changed | pass |  |

|  |  |
| --- | --- |
| Author | Kyle Pedersen |
| Creation Date | 2018-03-01 |
| Scenario | Approve Standing Reservation |
| Test Type | UAT |
| Version | 2.0 |
| Scenario | 001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Result # | Steps | Result | Pass/Failed | Comments |
| TR #006 | 1. Navigate to ViewStandingReservations.aspx 2. Click “Edit button” next to March 14 reservation 3. Click “Approve” | page is redirected to ViewStandingReservations.aspx, reservation that was approved is changed to approved | pass |  |

|  |  |
| --- | --- |
| Author | Kyle Pedersen |
| Creation Date | 2018-03-01 |
| Scenario | Enter Scores |
| Test Type | UAT |
| Version | 2.0 |
| Scenario | 001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Result # | Steps | Result | Pass/Failed | Comments |
| TR #007 | 1. Navigate to SubmitScore.aspx 2. Select “Easy Course” 3. Select 1,2,3,4,5,1,1,1,1,1,1,1,1,1,1,1,1,1 for hole scores 4. Click “Submit” | page is redirected to Default.aspx | pass |  |