# PROJECT – CRIC MANIA!

By Swapnil Yadav, Deepa Ratnam, Yashaswini Rajesh

- **Project Overview**

  Cricket is a popular sport in the Indian subcontinent, England and Australia. Cricket is a bat-and-ball game played between two teams of 11 players on a roughly circular field, at the center of which is a rectangular 22-yard long pitch.

  Each team takes it in turn to bat, in which they attempt to accumulate as many runs as possible, while the other team fields, attempting to prevent the batting team from scoring runs. Teams may bat once or twice each depending upon the format of the game.

  Each turn is known as an innings. The game progresses as one member of the fielding team, known as the bowler, delivers the ball to the batsman down the length of the pitch.

  The batsman then attempts to strike the ball with his bat, in order so that the ball either reaches the boundary or enables him to run to the other end of the pitch and thus accumulate runs.

  The batsman may continue batting until he is dismissed. Once ten batsmen from the batting side have been dismissed, the team is said to be all out and the two teams change roles.

- **Why this project for CS 5200**

  The idea of building a project around cricket was unanimous amongst all the teammates. Cricket is a very gripping game and it has been an integral part of all of our childhood in India. It is our attempt to showcase to everyone how wonderful this game is. Also, for people who are not familiar with the game, we have attempted to help them understand why this game is so riveting!

  Certain features in this project pull dynamic data – latest available scores using APIs, latest news using RSS feed, latest ranking, etc.

  Dynamic data is great, but what is does not do is provide historical view of the data – match data archive, ability to allow users to add comments on match outcome, etc. We have tried to focus on this aspect of the project to create a rich experience for the cricket maniacs.

- **Project functionalities**

  - **Teams Module:-**

    This module can be viewed from the 'Teams' link that appears on the menu bar on all screens. This screen is effectively a combination of 4 screens – one that shows the top ICC teams, one that shows the test rankings of teams, one that shows the odi rankings and lastly, one that shows the t20 rankings. The arrangement of the screen allows the user to skim through data easily. Each "button" – Test, Odi, T20 is actually an image. For example, if the user selects 'Odi', a JavaScript function is called which replaces the blue Odi image with the green Odi image. Green color indicates the current selection. The default selection is the 'Test' screen. Also, on each selection, the corresponding data is rendered.

- **Teams data**
  This screen provides the list of all teams that are recognized by ICC. The data for this screen is pulled from Yahoo API. The API stores the data in an XML format. On click of this link, we parse data from the API and store it in the database. To avoid duplicate inserts in the table, we first make a check for the existence of the record, based on the team name (this field acts as a unique identifier for each record) and if there is no record for that team, we insert it into the DB. The data is then retrieved from the database and rendered on screen. Each name on the screen is displayed as a link to the team's wiki page to help the user learn more about the team. This is done by appending the team name retrieved from the database with the wiki URL. Thus, as each record is pulled from the DB, the wiki URL is dynamically generated. Same is the case for each flag rendered on the screen. The image is stored as a file containing the team name. For each record found, the corresponding flag from the images folder is rendered.

- **Test Ranking, Odi Ranking, T20 Ranking.**
  The data for all 3 screens is parsed from the Yahoo API for rankings. The API provides details such as – ranking, captain first name, captain last name, coach first name, coach last name, etc. The data is parsed from the API and stored in the database. On click of this screen, if there is an update in any field, the data is updated in the database and the updated data is displayed. For a new record, data is inserted and a new row will show up in front end

- **Upcoming Match Module:-**
  This module can be viewed from the 'Upcoming' sub link under 'Matches' link that appears on the menu bar on all screens. This screen contains a list of all future fixtures. The data is retrieved from the Yahoo API for future matches. The data is in an XML format. On click of link, the latest data is pulled from the API. We pull data for match date, match type, series, venue and teams. The data is inserted into database. The screen is then rendered with data from the database. The screen displays data for matches in the order of dates in which the fixtures are going to occur.

- **Archived Match Module:-**
  – Dynamic data provides fleeting data. If the user comes back after a couple of days, he will lose context of what happened in an exciting match which occurred 2 days back. In order to avoid this, we will store archived match data and maintain it in tables.

  This module can be viewed from the 'Archived' sub link under 'Matches' link that appears on the menu bar on all screens. This screen contains data from matches that have already finished and is retrieved from the Cricbuzz API. The rendering of data was most challenging for this screen as API provides at least 27 records for each completed match. The Cricbuzz

API provides data in tags dynamically (some tags are added midway as the match is in progress) and thus it was important to ensure that data is correctly pulled to avoid null pointer exceptions. After parsing and storing the data, we faced the challenge of rendering all data without cluttering the front end. Thus, 2 screens were used. The main screen provides the basic details – match name, match number, team 1, team 2, match type, series, start date, end date and outcome. The match name is a link which leads to another screen providing all the granular details of the match such as – match city, country, stadium, toss, toss decision, both team's innings 1 and 2 runs, overs and wickets (especially important in a test match where each team plays 2 innings). For Odi and T20 matches, the 2nd innings data for both teams would contain – "Not Available". As we are parsing data from tags which dynamically change depending on type of match (test, odi, t20) being played, we chose to display – "Not Available" in cases where the API does not provide data. Also, the API provides data for each completed international match twice – one with entire data and another with only the summary (most fields have 'Not Available"). To avoid duplicate entries, we filtered data to display completed match information. Due to this filtering, we could only store and display international match information.

- **Player Profile Module:-**
  This module is the best way for users to search for their favorite players in respective teams. This module allows users to view player's personal profile. There are two ways in which users can search for their favorite players.

  1. In this method, the user selects his favorite team first from the drop down list followed with the selection of the format of the game i.e. either Test or T20 or ODI. On click of the 'Search Players' button, the module displays all the active players associated with the team from the database. It shows the player's picture, his date of birth, his birth city etc.

  2. If the user does not find his favorite player in the respective team and format, then there is possibility that he is no longer an active member of the team. In this case, the user can just type in the player's first name or last name in the text box provided and then click on search. The module will fetch the data from the API first and then store them in the database. So next time the user searches for his favorite player, it would directly be retrieved from the database rather than the API.

- **Tweet Module:-**
  Twitter has become an important social media to share one's thoughts. So we thought of including the latest cricket tweets from the world of cricket. We could build this feature with the help of twitter API. In this module, we always display the current date's tweets on the home page. All the tweets are first stored into the database and then rendered on to the screen. The unique feature that we have added, apart from displaying tweets is that our users can comment on these tweets as well. But in order to comment on the tweet, the user needs to first login into the application. In this way, the comments associated with the tweets are stored into the database.

- **Admin Module:-**
  This module provides special privileges to the admin user. The admin can load data to the database from the API. The admin can load news, players, teams , update players profile,

add upcoming matches, add archived matches, add world cup history. All the maintenance is done via this module.

- **Comments Module:-**
  This allows users to check for the comments that they had added to the tweets or the news. This will provide all the historical comments provided by the user. This way the user can view their respective past actions.

- **User Info Module:**
  This module stores user's first name, last name, username, password (hashed), date-of-birth and email address when they sign up. Security is provided by password hashing using SHA-256.
  When a user wishes to login, the user's username and password are checked against the list of registered users stored in our database. If they match, only then the user is recognized and allowed to log in.

- **News module:**
  We, as cricket fans, would always want to know as to what's happening in the world of cricket. To facilitate this, we provide the users with day to day news. Our users are also given the benefit of viewing a particular day's news from the past by just entering the date on which the news they are interested in had appeared. By default, only the current day's news is displayed on the screen.
  Data is retrieved using Yahoo's rest API. Admin loads the database everyday so that current day's news is also available in the database.
  We provide the users with a short description of the news and if the user is interested in knowing more about it he/she can get the complete story by clicking "Get detailed news". Also, users can comment on the news by clicking "Add comment". But in order to add a comment, the user must either sign up or login, i.e. he/she should be a registered user. This helps us track which user commented on what news. The comment related to the news, user id of the user who commented on the news, date on which the comment was entered etc. is all stored in the comments table.

- **World cup module:**
  This module helps users get information about the world cup's that were held right from the first one that was held in 1975 till the latest that was held in 2011.
  Since the world cup is held once every four years, beginning from 1975, there are not many rows to store. Hence data for this table is obtained by parsing a ".csv" file.

- **Conclusion**

  As it is evident, we are hard core cricket buffs. We have tried to integrate both dynamic data from API/ RSS feeds and store data in order to provide the ultimate cricketing experience to any cricket lover. Also, we have tried to provide a pleasing UI to the user so that he can enjoy viewing cricket data.

- **Screen Shots**

  Below are the screen shots depicting various modules:-


Fig 1 - Homepage


Fig 2 - Login & Signup Menu

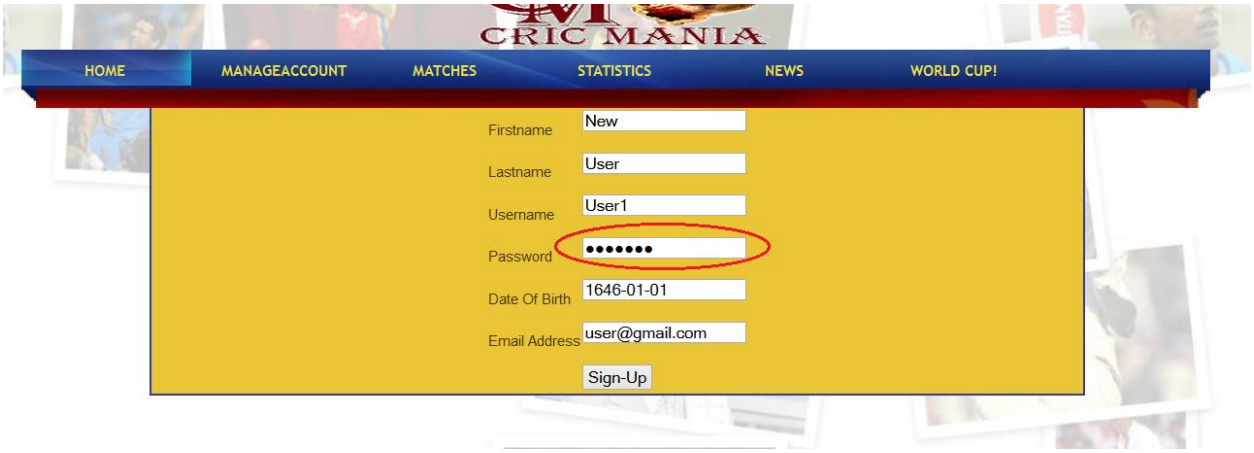



Fig 3 - Signup Page

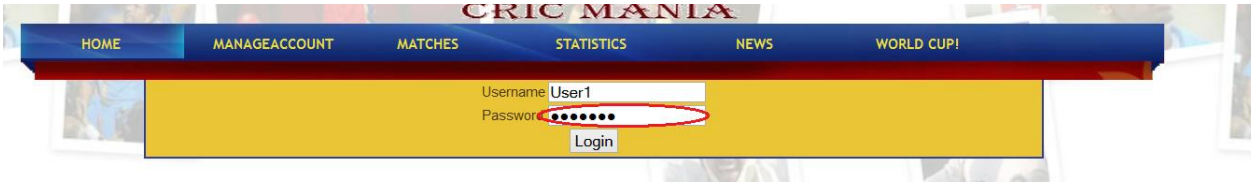



Fig 4 - Login Page


| Host | Date | Status | Team1 | Team2 | Man Of The Match | Man Of The Series |
|---|---|---|---|---|---|---|
| "England" | 1975-06-21 | "West Indies won by 17 runs" | "West Indies" | "Australia" | "Clive Lloyd" | "N/A" |
| "England" | 1979-06-23 | "West Indies won by 92 runs" | "West Indies" | "England" | "Viv Richards" | "N/A" |
| "England" | 1983-06-25 | "India won by 43 runs" | "West Indies" | "India" | "Mohinder Amarnath" | "N/A" |
| "India and Pakistan" | 1987-11-08 | "Australia won by 7 runs" | "Australia" | "England" | "David Boon" | "N/A" |
| "Australia and New Zealand" | 1992-03-25 | "Pakistan won by 22 runs" | "Pakistan" | "England" | "Wasim Akram" | "Martin Crowe" |
| "Pakistan and India and Sri Lanka" | 1996-03-17 | "Sri Lanka won by 7 wickets" | "Sri Lanka" | "Australia" | "Aravinda de Silva" | "Sanath Jayasuriya" |
| "England and Scotland and Ireland and Netherlands" | 1999-06-20 | "Australia won by 8 wickets" | "Australia" | "Pakistan" | "Shane Warne" | "Lance Klusener" |
| "South Africa and Zimbabwe and Kenya" | 2003-03-23 | "Australia won by 125 runs" | "Australia" | "India" | "Ricky Ponting" | "Sachin Tendulkar" |
| "West Indies" | 2007-04-28 | "Australia won by 53 runs" | "Australia" | "Sri Lanka" | "Adam Gilchrist" | "Glenn McGrath" |
| "India and Sri Lanka and Bangladesh" | 2011-04-02 | "India won by 6 wickets" | "India" | "Sri Lanka" | "Mahendra Singh Dhoni" | "Yuvraj Singh" |

Fig 5 - World Cup Page


Fig 6 - News Page


fig 7 - Add Comments Page


Fig 8 - Matches menu


| Match Date | Match Type | Match Series | Match Venue | Team 1 | Team 2 |
|---|---|---|---|---|---|
| 2014-08-15 | test | India in England Test Series, 5 Test Cricket Series, 2014 | The Oval, London | England | India |
| 2014-08-17 | odi | South Africa in Zimbabwe ODI Series, 3 ODI Cricket Series, 2014 | Queens Sports Club, Bulawayo | Zimbabwe | South Africa |
| 2014-08-20 | odi | Bangladesh in West Indies ODI Series, 3 ODI Cricket Series, 2014 | National Cricket Stadium, St George's, Grenada | West Indies | Bangladesh |
| 2014-08-14 | test | Pakistan in Sri Lanka Test Series, 2 Test Cricket Series, 2014 | Sinhalese Sports Club, Colombo | Sri Lanka | Pakistan |
| 2014-08-19 | odi | South Africa in Zimbabwe ODI Series, 3 ODI Cricket Series, 2014 | Queens Sports Club, Bulawayo | Zimbabwe | South Africa |

Fig 9 - Upcoming Matches Page


| Match Name | Match No | Team 1 | Team 2 | Match Type | Series | Start Date | End Date | Outcome |
|---|---|---|---|---|---|---|---|---|
| ENG vs IND | 4th Test | IND | ENG | TEST | India tour of England 2014 | Aug 07 2014 | Aug 11 2014 | Eng won by an innings and 54 runs |
| SL vs PAK | 1st Test | SL | PAK | TEST | Pakistan tour of Sri Lanka, 2014 | Aug 06 2014 | Aug 10 2014 | SL won by 7 wkts |

Fig 10 a - Archived Matches


Fig 10 b - Archived Match Page


Fig 11 - Teams Page


Fig 12 - Players Page


Fig 13 - Admin Page

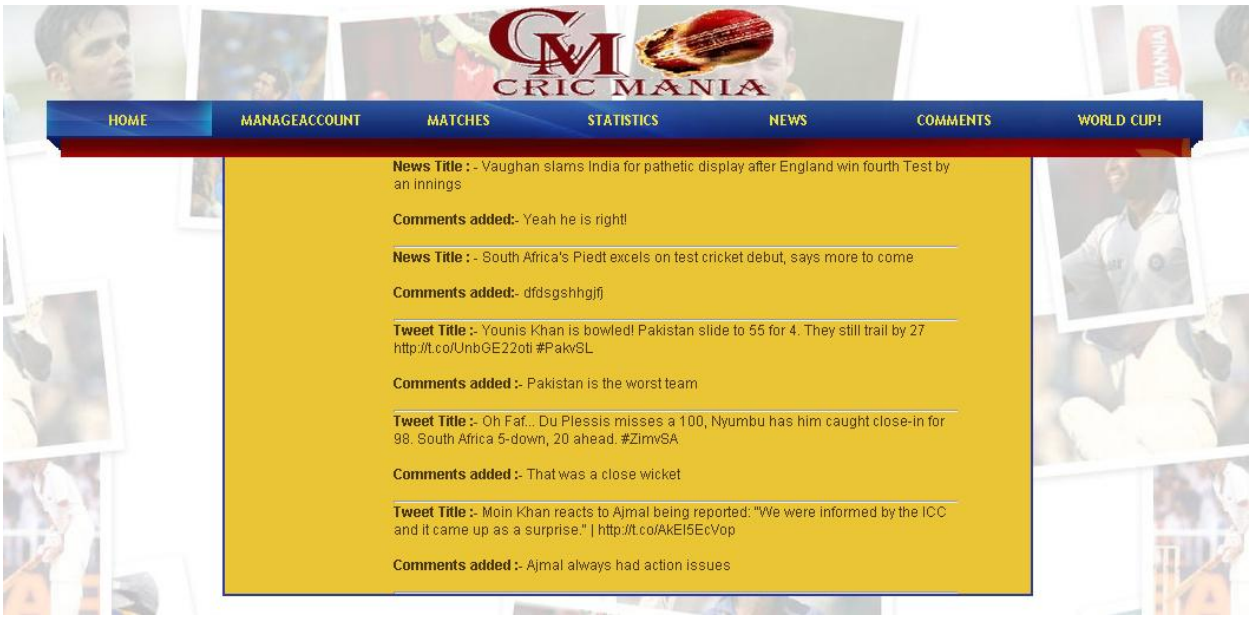
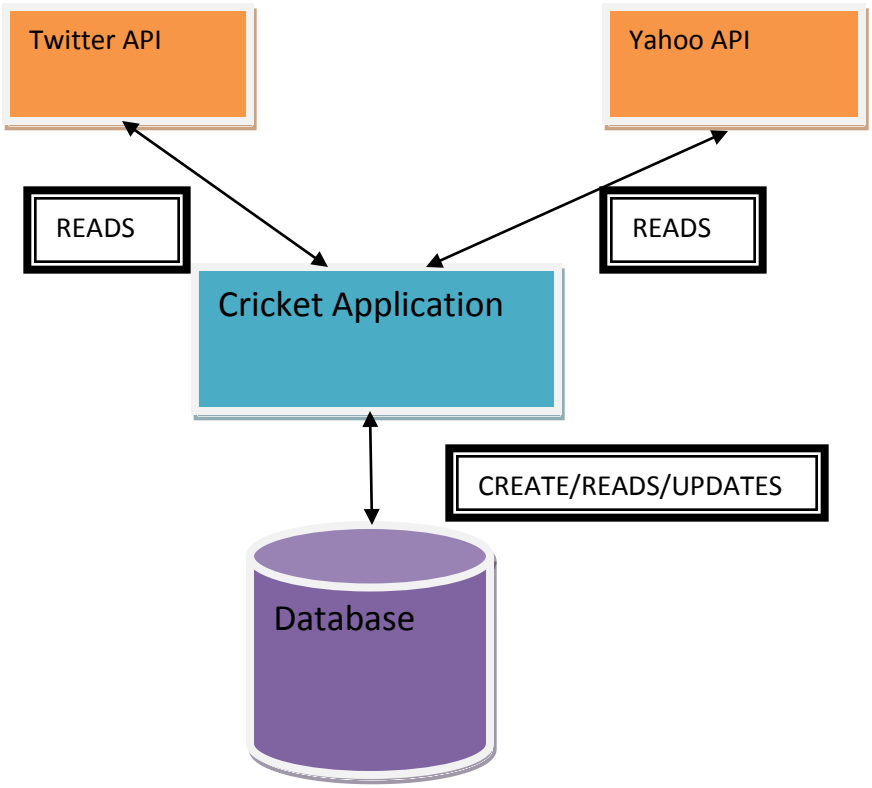



Fig 14 - Comments Page

- **Architecture**

Below is the architecture diagram that shows how are application connects to the database and API.



The above diagram demonstrates how our application interacts to the API and Database. Since, our application is very dynamic and the data keeps on changing daily we are using Database a cache memory. Always the latest data is pulled up from the respective api's and stored into our database. This activity of synchronizing the api data with that present in the database is done only once a day. Thus, when a user accesses any information it is directly pulled from the database rather than the API. The role of keeping the database up to date is given to the admin. This is the way our application works.

**Technology Used**: - We have used the below technologies to accomplish our project.

**Programming Language**: - JAVA

**ORM**: - JPA (We created the classes first and then the tables)

**Database**: - postgressql.

**Server**: - Apache Tomcat 6.0

- **Table Statistics**

**Number of tables : 12**

- **Table**: archivedmatch

1. **Fields**:

archivedmatchid
archivedmatchname
archivedmatchno
archivedmatchseries
archivedmatchtype
archivedmcity
archivedmcountry
archivedmenddate
archivedmstadium
archivedmstartdate
archivedmstatus
archivedmteam1
archivedmteam1innings1overs
archivedmteam1innings1runs
archivedmteam1innings1wickets
archivedmteam1innings2overs

archivedmteam1innings2runs
archivedmteam1innings2wickets
archivedmteam2
archivedmteam2innings1overs
archivedmteam2innings1runs
archivedmteam2innings1wickets
archivedmteam2innings2overs
archivedmteam2innings2runs
archivedmteam2innings2wickets
archivedmtoss
archivedmtossdecision

2. **Foreign Keys:** None
3. **Mapping Table:** None

---

- **Table**: comments

1. **Fields** :

commentid
commentsdate
commentstext
news_newsid
tweetinfo_tweetid
userinfo_userid

2. **Foreign Keys:**

fk_comments_news_newsid having news_newsid referencing archivednewsid column of news table

fk_comments_tweetinfo_tweetid having tweetinfo_tweetid referencing tweeted column of tweetinfo table

fk_comments_userinfo_userid having userinfo_userid referencing userid column of userinfo table

3. **Mapping table:** news, tweetinfo, userinfo

---

- **Table: News**

1. **Fields**:

archivednewsid
archivednewsdateandtime
archivednewslink
archivednewstitle

2. **Foreign Keys:** None
3. **Mapping table**: comments

---

- **Table: oditeams**

1. **Fields:**

   oditeamid
   odicapfname
   odicaplname
   odicoachfname
   odicoachlname
   oditeamranking
   teams_teamid

2. **Foreign Keys:**
   fk_oditeams_teams_teamid having teams_teamid referencing teamid of teams

3. **Mapping tables:** teams, player

---

- **Table: player**

1. **Fields:**

   playerid
   birthcity
   playerbattingstyle
   playerbowlingstyle
   playergender
   playerimage
   playername
   playersdob
   oditeams_oditeamid
   t20teams_t20teamid
   testteams_testteamid

2. **Foreign Keys:**

fk_player_oditeams_oditeamid having oditeams_oditeamid referencing oditeamid column of oditeams table

fk_player_t20teams_t20teamid having t20teams_t20teamid referencing t20teamid column of t20teams table.

fk_player_testteams_testteamid having testteams_testteamid referencing testteamid column of testteams table

3. **Mapping tables:** oditeams, t20teams, testteams

---

- **Table: t20teams**

1. **Fields:**

t20teamid

t20capfname

t20caplname

t20coachfname

t20coachlname

t20teamranking

teams_teamid

2. **Foreign Keys:**

fk_t20teams_teams_teamid having teams_teamid referencing teamid column of teams table

3. **Mapping tables:** teams, player

---

- **Table: teams**

1. **Fields:**

teamid

teamname

yahooteamid

2. **Foreign Keys:** None
3. **Mapping tables:** testteams, oditeams, t20teams

---

- **Table: testteams**

1. **Fields:**

    testteamid

    testcapfname

    testcaplname

    testcoachfname

    testcoachlname

    testteamranking

    teams_teamid

2. Foreign Keys:

    fk_testteams_teams_teamid having teams_teamid referencing teamid column of teams

3. Mapping tables: teams, player

---

- **Table: tweetinfo**

1. **Fields:**

    tweetid

    tweetdate

    tweetname

    tweettext

    username

2. **Foreign Keys**: None
3. **Mapping Tables:** comments

---

- **Table: upcomingmatch**

1. **Fields:**

    matchid

    matchdate

    matchseries

    matchteam1

    matchteam2

    matchtype

matchvenue

2. **Foreign Keys:** None
3. **Mapping Tables:** None

---

- **Table: userinfo**
  1. **Fields:**

     userid

     isuseradmin

     userdob

     useremail

     userfirstname

     userlastname

     userloginname

     userpassword

  2. **Foreign Keys:** None
  3. **Mapping Tables:** comments

---

- **Table: worldcuphistory**

  1. **Fields:**

     worldcuphistoryid

     worlcuphistoryteam1

     worldcuphistorydate

     worldcuphistoryhost

     worldcuphistorymom

     worldcuphistorymos

     worldcuphistorystatus

     worldcuphistoryteam2

  2. **Foreign Keys:** None
  3. **Mapping tables:** None

---

- **Enumeration**: - We do not have any enumerations in our database design.
- **Pages: -** We have given the pages as screen shots of the pages.

- **Interfaces:** - We do not have any interfaces to our application.
- **Fields: -** We have given the list of fields in the table statistics section.

- **CRUD Operations:**

1. **Teams Module:**
    - **Teams Data :**

      Created by the admin:

      When admin selects the "Create Team" button, data is obtained by parsing XML data from the Yahoo API. The retrieved data is then stored in the database.

      Read by all users:

      When a user selects "Statistics->Teams" on the home page the data retrieved from the database is presented to the user.

      Updated by the admin:

      Data is updated by the admin.

      API URL :-

      https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20cricket.teams&diagnostics=true&env=store%3A%2F%2F0TxIGQMQbObzvU4Apia0V0

    - **Test Ranking, ODI Ranking, T20 Ranking:**

      Created by the admin:

      Initially data is obtained by parsing XML data from the Yahoo API. The retrieved data is then stored in the database. (Also created in parallel with the teams data when the admin selects "Create Team".

      Read by all users:

      When a user selects "Statistics->Teams" on the home page the data retrieved from the database is presented to the user. By default the Test ranking is presented to the user. User can select either ODI or T20 to retrieve the respective rankings data from the database.

      Updated by the admin:

      Data is updated by the admin.

      API URL :-

      https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20cricket.teams&diagnostics=true&env=store%3A%2F%2F0TxIGQMQbObzvU4Apia0V0

2. **Upcoming matches data:**

   Created by the admin:

   When admin selects "Add Upcoming Matches" data is obtained by parsing XML data from the Yahoo API. The retrieved data is then stored in the database.

   Read by all users:

When a user selects “Matches->Upcoming” on the home page the data retrieved from the database is presented to the user.

Updated by admin:

Data is updated by the admin.

API URL :-

https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20cricket.upcoming_matches&diagnostics=true&env=store%3A%2F%2F0TxIGQMQbObzvU4Apia0V0

3. **Archived Match Data:**

Created by the admin:

When admin selects “Add Archived Matches” data is obtained by parsing XML data from the Cricbuzz API. The retrieved data is then stored in the database.

Read by all users:

When a user selects “Matches->Archived” on the home page the data retrieved from the database is presented to the user.

Updated by admin:

Data is updated by the admin.

API URL :- http://synd.cricbuzz.com/j2me/1.0/livematches.xml

4. **Player Profile Data:**

Created by the admin:

When admin selects “Add Players” data is obtained by parsing XML data from the Yahoo API. The retrieved data is then stored in the database. Only the players with active personId are retrieved from the api.

Read by all users:

When a user selects “Statistics->Players” he is presented with 2 drop down menus and a text box.

User can select any team and any format of game from the drop down menus and the corresponding data is retrieved from the database.

If a user wishes to get a particular player’s info, he/she can do so by typing in the name of the player into the text box and the corresponding player’s data is retrieved from the database.

Updated by admin:

When admin selects “Update Players” data is updated by parsing XML data from the Yahoo API and then the updated data (if any update exists) is stored in the database.

API URL :-

https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20cricket.player.profile%20where%20player_id%3D" + personId + "&diagnostics=true&env=store%3A%2F%2F0TxIGQMQbObzvU4Apia0V0

5. **Tweet Data:**

Created by the admin:

Data is obtained by parsing XML data from the Twitter API. The retrieved data is then stored in the database.

Read by all users:

Tweets are displayed in a rectangular box towards the bottom right of the screen.

Update: No update as current date's latest tweets are always displayed.

API :- Here we use the twitter4j API calls to retrieve all the cricketing tweets from user ESPNCricinfo.

6. **Comments Data:**

Created by the users:

The comment/text entered by the user is stored in the database.

Readable only by the user who has provided the comment, i.e. a user can read his/her comments only.

Update: No update

API URL :- None

7. **User Info Data:**

Created by users:

Data provided by the user during "Sign Up" is stored in the database.

Read: Nothing to read.

Update: No Update

API URL :- None

8. **News Data:**

Created by the admin:

When admin selects "Add News" data is obtained by parsing XML data from the Yahoo API. The retrieved data is then stored in the database.

Read by all users:

When a user selects "News" on the home page the data retrieved from the database is presented to the user. By default the current day's news is displayed on the screen. However user also has a choice of entering a particular date in the text box present on the screen. Depending on the date the corresponding data is retrieved from the database.

Update:

Data is updated by the admin.

API URL:-

https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20cricket.news(0%2C50)%20%20where%20region%3D%22in%22%20%20or%20region%3D%22aus%22%20or%20region%3D%22sa%22%20or%20region%3D%22pak%22%20or%20region%3D%22eng%22%20or%20region%3D%22sl%22%20or%20region%3D%22nz%22%20or%20region%3D%22wi%22%20or%20region%3D%22zim%22%20or%20region%3D%22ban%22&diagnostics=true&env=store%3A%2F%2F0TxIGQMQbObzvU4Apia0V0

9. **World Cup History Data:**
   Created by the admin:
   When admin selects “Add World Cup History” data is obtained by parsing a “.csv” file. The retrieved data is then stored in the database.
   Read by all users:
   When a user selects “World Cup” on the home screen the data retrieved from the database is presented to the user.
   API URL :- None

- **Webservices Api's :-** We have mentioned all the details of the api's and their usage in the CRUD operation sections.

- **Classes :-** Below are the list of all the classes created for our project.

1. ArchivedMatch.java
2. ArchivedMatchDao.java
3. Comments.java
4. CommentsDao.java
5. News.java
6. NewsDao.java
7. OdiTeams.java
8. OdiTeamsDao.java
9. Player.java
10. PlayerDao.java
11. T20Teams.java
12. T20TeamsDao.java
13. Teams.java
14. TeamsDao.java
15. TestTeams.java
16. TestTeamsDao.java
17. TweetInfo.java
18. TweetInfoDao.java
19. UpcomingMatch.java
20. UpcomingMatchDao.java
21. Userinfo.java
22. UserinfoDao.java
23. WorldCupHistoryDao.java
24. WorldCupHistory.java