Bilkent University

CS 353

Database Systems

Project Design Document

Group 14

Şekip Kaan EKİN 21401795 Section 1

Ege Darçın 21302460 Section 1

Can Demirel 21401521 Section 1

Eren Aslantürk 21401306 Section 2

1. Introduction

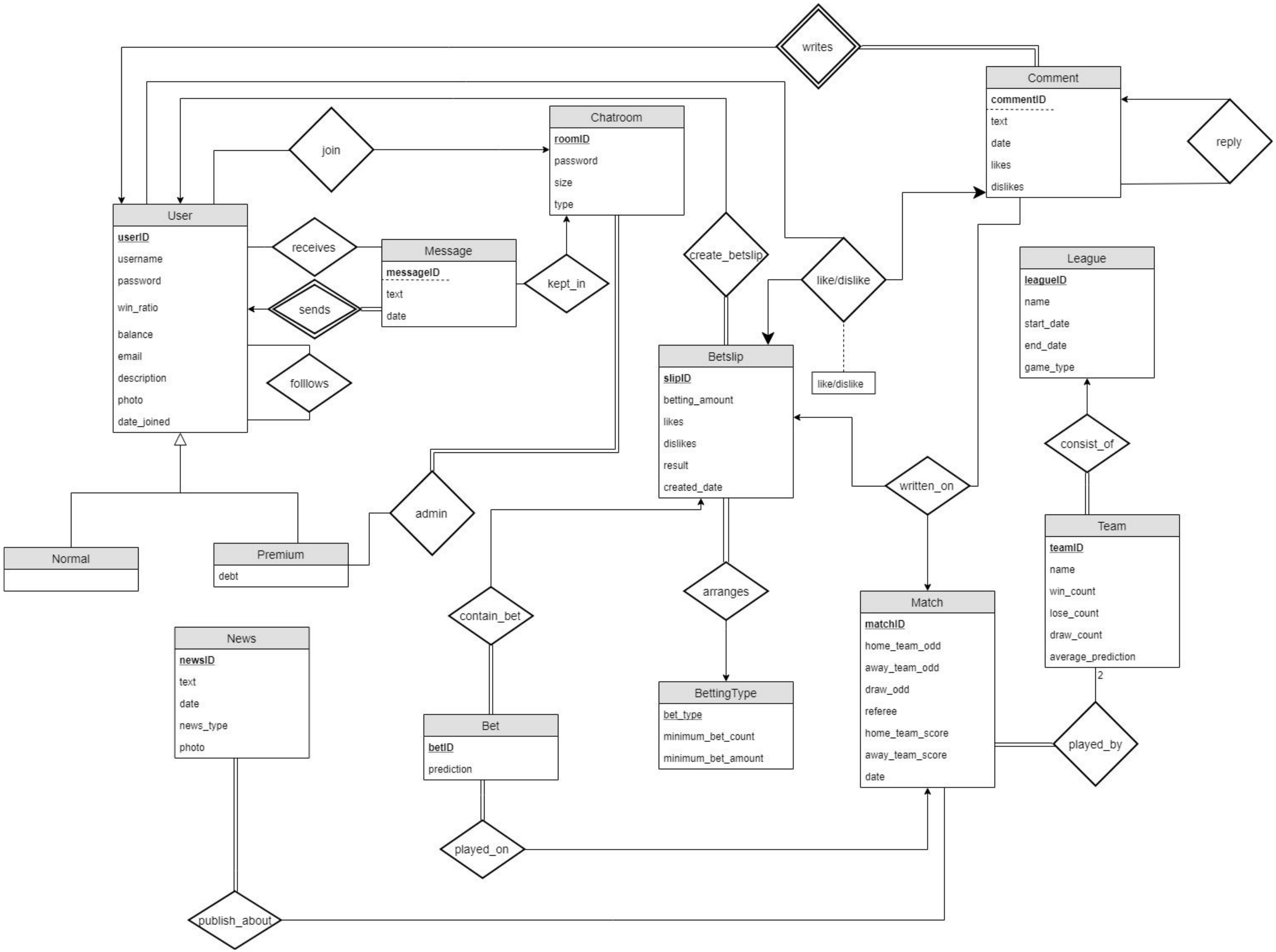
This design report is about our database project called BetBall which is a social betting platform for football and basketball matches with social features. This report includes the revised E/R diagram, relation schemas, functional components, UI designs and corresponding SQL statements, advanced database components and implementation plan of the project.

The report is accessible at:

https://github.com/sekin72/Database-Betting-Project

2. Revised ER Diagram

- Foreign keys are removed from the diagram.
- Message attribute that was kept in the Chatroom is now a seperate entity that has relations with user and chatroom.
- Different betting types holding no special attributes to their own removed from the dagram and added as an attribute to BettingType.
- BetSlip, Bet, News and Team entities are changed to normal entities rather than weak entities.
- Like relation added to the diagram between user, comment and bet slip.
- Incorrect use of ternary relation is corrected.
- Cardinality in the relation reply corrected.


writes
Comment
commentID
text
date
likes
dislikes
reply
join
Chatroom
roomID
password
size
type
User
userID
username
password
win_ratio
balance
email
description
photo
date_joined
receives
Message
messageID
text
date
sends
kept_in
folllows
create_betslip
like/dislike
like/dislike
League
leagueID
name
start_date
end_date
game_type
Betslip
slipID
betting_amount
likes
dislikes
result
created_date
written_on
consist_of
Team
teamID
name
win_count
lose_count
draw_count
average_prediction
2
Normal
Premium
debt
admin
contain_bet
arranges
Match
matchID
home_team_odd
away_team_odd
draw_odd
referee
home_team_score
away_team_score
date
played_by
News
newsID
text
date
news_type
photo
Bet
betID
prediction
BettingType
bet_type
minimum_bet_count
minimum_bet_amount
played_on
publish_about

2.1 Entity Sets:

2.1.1 User

- Relational Model:

User(userID, username, password, win_ratio, balance, email, description, photo, day_joined)

- Functional Dependencies:

userID → username, password, win_ratio, balance, email, description, photo, day_joined

username → userID, password, win_ratio, balance, email, description, photo, day_joined

email → userID, username, password, win_ratio, balance,, description, photo, day_joined

- Candidate Keys:

{ (userID), (username), (email) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE User(
        userID INT PRIMARY_KEY AUTO_INCREMENT,
        username VARCHAR(32) NOT NULL UNIQUE,
        password VARCHAR(32) NOT NULL,
        win_ratio DOUBLE,
        balance DOUBLE,
        email VARCHAR(64) NOT NULL UNIQUE,
        description VARCHAR(64),
        photo IMAGE,
        day_joined CURRENT_TIMESTAMP()
        )
        engine=InnoDB;
```

2.1.2 Normal

- Relational Model:

Normal(<u>userID</u>)

FK: userID references User

- Functional Dependencies:

None

- Candidate Keys:

{ (userID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Normal(
        userID INT PRIMARY_KEY,
        FOREIGN KEY (userID) REFERENCES User(userID))
```

```
engine=InnoDB;
```

2.1.3 Premium

- Relational Model:

Premium(userID, debt)

FK: userID references User

- Functional Dependencies:

None

- Candidate Keys:

{ (userID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Premium(
        userID INT PRIMARY_KEY,
        debt DOUBLE,
        FOREIGN KEY (userID) REFERENCES User(userID))
        engine=InnoDB;
```

2.1.4 Chatroom

- Relational Model:

Chatroom(roomID, size, type)

- Functional Dependencies:

roomID → type

- Candidate Keys:

{ (userID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Chatroom(
        roomID INT PRIMARY_KEY,
        size INT,
        type VARCHAR(32))
        engine=InnoDB;
```

2.1.5 Message

- Relational Model:

Message(messageID, userID, text, date)

FK: userID references User

- Functional Dependencies:

messageID, userID → userID, text,date

- Candidate Keys:

{ (messageID, userID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Message(
        messageID INT AUTO_INCREMENT,
        userID INT,
        text VARCHAR(32),
        date CURRENT_TIMESTAMP(),
        PRIMARY KEY(messageID, userID),
        FOREIGN KEY (userID) REFERENCES User(userID))
        engine=InnoDB;
```

2.1.6 Betslip

- Relational Model:

BetSlip(slipID, betting_amount, likes, dislikes, result, created_date)

- Functional Dependencies:

slipID → betting_amout, likes, dislikes

- Candidate Keys:

{ (slipID)}

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE BetSlip(
        slipID INT PRIMARY_KEY AUTO_INCREMENT,
        betting_amout INT,
        likes INT,
        dislikes INT,
        result INT NULL,
        created_date CURRENT_TIMESTAMP())
        engine=InnoDB;
```

2.1.7 Bet

- Relational Model:

Bet(betID, prediction)

- Functional Dependencies:

betID → prediction

- Candidate Keys:

{ (betID)}

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Bet(
        betID INT PRIMARY_KEY AUTO_INCREMENT,
        prediction INT)
        engine=InnoDB;
```

2.1.8 Betting Type

- Relational Model:

BettingType(bet_type, minimum_bet_count, minimum_bet_amount)

- Functional Dependencies:

bet_type → minimum_bet_count, minimum_bet_amount

- Candidate Keys:

{ (bet_type)}

- Normal Form: 3NF
-
- Table Definition:

```
CREATE TABLE BettingType(
        bet_type INT PRIMARY_KEY AUTO_INCREMENT,
        minimum_bet_count INT,
        minimum_bet_amount INT)
        engine=InnoDB;
```

2.1.9 Comment

- Relational Model:

Comment(userID, commentID, text, date, likes, dislikes)

FK: userID references User

- Functional Dependencies:

commentID, userID → commentID, text, date, likes, dislikes

- Candidate Keys:

{ (commentID, userID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Comment(
        commentID INT AUTO_INCREMENT,
        userID INT,
        text VARCHAR(32),
        date CURRENT_TIMESTAMP(),
        likes INT,
        dislike INT,
        PRIMARY KEY(commentID, userID),
        FOREIGN KEY (userID) REFERENCES User(userID))
        engine=InnoDB;
```

2.1.10 News

- Relational Model:

News(newsID, text, date, news_type, photo)

- Functional Dependencies:

newsID → text, date, news_type, photo

- Candidate Keys:

{ (newsID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE News(
        newsID INT PRIMARY_KEY AUTO_INCREMENT,
        text VARCHAR(32) NOT NULL,
        news_type VARCHAR(32),
        photo IMAGE,
        date CURRENT_TIMESTAMP())
        engine=InnoDB;
```

2.1.11 Match

- Relational Model:

Match(matchID, home_team_odd, away_team_odd, draw_odd, referee, home_team_score, away_team_score, date)

- Functional Dependencies:

matchID → home_team_odd, away_team_odd, draw_odd, referee, home_team_score, away_team_score, date

referee, date → matchID, home_team_odd, away_team_odd, draw_odd, home_team_score, away_team_score

- Candidate Keys:

{ (matchID), (referee, date) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Match(
        matchID INT PRIMARY_KEY AUTO_INCREMENT,
        home_team_odd DOUBLE,
        away_team_odd DOUBLE,
        draw_odd DOUBLE,
        referee VARCHAR(32),
        home_team_score INT,
        away_team_score INT,
        date CURRENT_TIMESTAMP())
        engine=InnoDB;
```

2.1.12 Team

- Relational Model:

Team(teamID, name, win_count, lose_count, draw_count, average_prediction)

- Functional Dependencies:

teamID → name, win_count, lose_count, draw_count, average_prediction

name → teamID, win_count, lose_count, draw_count, average_prediction

- Candidate Keys:

{ (teamID), (name) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Team(
        teamID INT PRIMARY_KEY AUTO_INCREMENT,
        name VARCHAR(32),
        win_count INT,
        lose_count INT,
        draw_count INT,
        average_prediction DOUBLE)
        engine=InnoDB;
```

2.1.13 League

- Relational Model:

League(leagueID, name, start_date, end_date, game_type)

- Functional Dependencies:

leagueID → name, start_date, end_date, game_type

name → leagueID, start_date, end_date, game_type

- Candidate Keys:

{ (leagueID), (name) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE League(
        leagueID INT PRIMARY_KEY AUTO_INCREMENT,
        name VARCHAR(32),
        start_date DATE,
        end_date DATE,
        game_type VARCHAR(32))
        engine=InnoDB;
```

2.2 Relation Sets:

2.2.1 Join

- Relational Model:

join(userID, roomID)

FK: userID references User
roomID references Chatroom

- Functional Dependencies:

None

- Candidate Keys:

{ (userID, roomID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Join(
        userID INT,
        roomID INT,
        PRIMARY_KEY(userID, roomID),
        FOREIGN_KEY (userID) REFERENCES User(userID),
        FOREIGN_KEY (roomID) REFERENCES Chatroom(roomID))
        engine=InnoDB;
```

2.2.2 Sends

- Relational Model:

sends(userID, messageID)

FK: userID references User

messageID references Message

- Functional Dependencies:

None

- Candidate Keys:

{ (userID, messageID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Sends(
        userID INT,
        messageID INT,
        PRIMARY_KEY(userID, messageID),
        FOREIGN_KEY (userID) REFERENCES User(userID),
        FOREIGN_KEY (messageID) REFERENCES Message(messageID))
        engine=InnoDB;
```

2.2.3 Kept In

- Relational Model:

kept_in(roomID, messageID)

FK: roomID references ChatRoom
messageID references Message

- Functional Dependencies:

None

- Candidate Keys:

{ (roomID, messageID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE kept_in(
        roomID INT,
        messageID INT,
        PRIMARY_KEY(roomID, messageID),
        FOREIGN_KEY (roomID) REFERENCES Chatroom(roomID),
        FOREIGN_KEY (messageID) REFERENCES Message(messageID))
        engine=InnoDB;
```

2.2.4 Follows

- Relational Model:

follows(userID, followerID)

FK: userID references User
followerID references User

- Functional Dependencies:

None

- Candidate Keys:

{ (userID, followerID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Follows(
    userID INT,
    followerID INT,
    PRIMARY_KEY(userID, followerID),
    FOREIGN_KEY (userID) REFERENCES User(userID),
    FOREIGN_KEY (followerID) REFERENCES User(followerID))
    engine=InnoDB;
```

2.2.5 Admin

- Relational Model:

admin(userID, roomID)

FK: userID references User
roomID references Chatroom

- Functional Dependencies:

None

- Candidate Keys:

{ (userID, roomID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE Admin(
        userID INT,
        roomID INT,
        PRIMARY_KEY(userID, roomID),
        FOREIGN_KEY (userID) REFERENCES User(userID),
        FOREIGN_KEY (roomID) REFERENCES Chatroom(roomID))
        engine=InnoDB;
```

2.2.6 Contain Bet

- Relational Model:

contain_bet(slipID, betID)

FK: slipID references Betslip
betID references Bet

- Functional Dependencies:

None

- Candidate Keys:

{ (slipID, betID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE contain_bet(
        slipID INT,
        betID INT,
        PRIMARY_KEY(slipID, betID),
        FOREIGN_KEY (slipID) REFERENCES BetSlip(slipID),
        FOREIGN_KEY (betID) REFERENCES Bet(betID))
        engine=InnoDB;
```

2.2.7 Publish About

- Relational Model:

publish_about(newsID, publishedEntityID)

FK: newsID references News
publishedEntityID references Match

- Functional Dependencies:

None

- Candidate Keys:

{ (newsID, publishedEntityID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE publish_about(
        newsID INT,
        publishedEntityID INT,
        PRIMARY_KEY(newsID, publishedEntityID),
        FOREIGN_KEY (newsID) REFERENCES News(newsID),
        FOREIGN_KEY (publishedEntityID) REFERENCES Match(matchID))
        engine=InnoDB;
```

2.2.8 Played On

- Relational Model:

played_on(betID, matchID)

FK: betID references Bet
matchID references Match

- Functional Dependencies:

None

- Candidate Keys:

{ (betID, matchID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE played_on(
        betID INT,
        matchID INT,
        PRIMARY_KEY(betID, matchID),
        FOREIGN_KEY (betID) REFERENCES Bet(betID),
        FOREIGN_KEY (matchID) REFERENCES Match(matchID))
        engine=InnoDB;
```

2.2.9 Writes

- Relational Model:

writes(userID, commentID)

FK: userID references User
commentID references Comment

- Functional Dependencies:

None

- Candidate Keys:

{ (userID, commentID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE writes(
    userID INT,
    commentID INT,
    PRIMARY_KEY(userID, commentID),
    FOREIGN_KEY (userID) REFERENCES User(userID),
    FOREIGN_KEY (commentID) REFERENCES Comment(commentID))
    engine=InnoDB;
```

2.2.10 Create Betslip

- Relational Model:

create_betslip(userID, slipID)

FK: userID references User
slipID references Betslip

- Functional Dependencies:

None

- Candidate Keys:

{ (userID, slipID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE create_betslip(
        userID INT,
        slipID INT,
        PRIMARY_KEY(userID, slipID),
        FOREIGN_KEY (userID) REFERENCES User(userID),
        FOREIGN_KEY (slipID) REFERENCES BetSlip(slipID))
        engine=InnoDB;
```

2.2.11 Arranges

- Relational Model:

arranges(slipID, bet_type)

FK: slipID references Betslip
bet_type references BettingType

- Functional Dependencies:

None

- Candidate Keys:

{ (slipID, bet_type) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE arranges(
    slipID INT,
    bet_type INT,
    PRIMARY_KEY(slipID, bet_type),
    FOREIGN_KEY (slipID) REFERENCES BetSlip(slipID),
    FOREIGN_KEY (bet_type) REFERENCES BettingType(bet_type))
    engine=InnoDB;
```

2.2.12 Written On

- Relational Model:

written_on(commentID, commentedEntityID)

FK: commentID references Comment
commentedEntityID references Betslip
commentedEntityID references Match

- Functional Dependencies:

None

- Candidate Keys:

{ (commentID, commentedEntityID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE written_on(
        commentID INT,
        commentedEntityID INT,
        PRIMARY_KEY(commentID, commentedEntityID),
        FOREIGN_KEY (commentID) REFERENCES Comment(commentID),
        FOREIGN_KEY (commentedEntityID) REFERENCES BetSlip(slipID),
        FOREIGN_KEY (commentedEntityID) REFERENCES Match(matchID))
        engine=InnoDB;
```

2.2.13 Reply

- Relational Model:

reply(commentID, commentedCommentID)

FK: commentID references Comment
commentedCommentID references Comment

- Functional Dependencies:

None

- Candidate Keys:

{ (commentID, commentedCommentID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE arranges(
        commentID INT,
        commentedCommentID INT,
        PRIMARY_KEY(commentID, commentedCommentID),
        FOREIGN_KEY (commentID) REFERENCES Comment(commentID),
        FOREIGN_KEY (commentedCommentID) REFERENCES Comment(commentID))
        engine=InnoDB;
```

2.2.14 Consist of

- Relational Model:

consist_of(leagueID, teamID)

FK: leagueID references League
teamID references Team

- Functional Dependencies:

None

- Candidate Keys:

{ (leagueID, teamID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE consist_of(
        leagueID INT,
        teamID INT,
        PRIMARY_KEY(leagueID, teamID),
        FOREIGN_KEY (leagueID) REFERENCES League(leagueID),
        FOREIGN_KEY (teamID) REFERENCES Team(teamID))
        engine=InnoDB;
```

2.2.15 Played By

- Relational Model:

played_by(matchID, home_teamID, away_teamID)

FK: matchID references Match
home_teamID references Team
away_teamID references Team

- Functional Dependencies:

None

- Candidate Keys:

{ (matchID, home_teamID, away_teamID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE played_by(
	matchID INT,
	home_teamID INT,
	away_teamID INT,
	PRIMARY_KEY(matchID, home_teamID, away_teamID),
	FOREIGN_KEY (matchID) REFERENCES Match(matchID),
	FOREIGN_KEY (home_teamID) REFERENCES Team(home_teamID),
	FOREIGN_KEY (away_teamID) REFERENCES Team(away_teamID))
	engine=InnoDB;
```

2.2.16 Like/Dislike

- Relational Model:

like/dislike(userID, likedEntityID, like/dislike)

FK: userID references User
likedEntityID references Betslip
likedEntityID references Comment

- Functional Dependencies:

None

- Candidate Keys:

{ (userID, likedEntityID) }

- Normal Form: 3NF

- Table Definition:

```
CREATE TABLE like/dislike(
        userID INT,
        likedEntityID INT,
        like/dislike BOOL,
        PRIMARY_KEY(userID, likedEntityID),
        FOREIGN_KEY (userID) REFERENCES User(userID),
        FOREIGN_KEY (likedEntityID) REFERENCES Betslip(slipID),
        FOREIGN_KEY (likedEntityID) REFERENCES Comment(commentID))
        engine=InnoDB;
```

2.2.17 Receives

- Relational Model:

like/dislike(senderID, receiverID, messageID)

FK: userID references User
receiverID references User
messageID references Message

- Functional Dependencies:

None

- Candidate Keys:

{ (senderID, receiverID, messageID) }

- Normal Form: 3NF

- Table Definition:

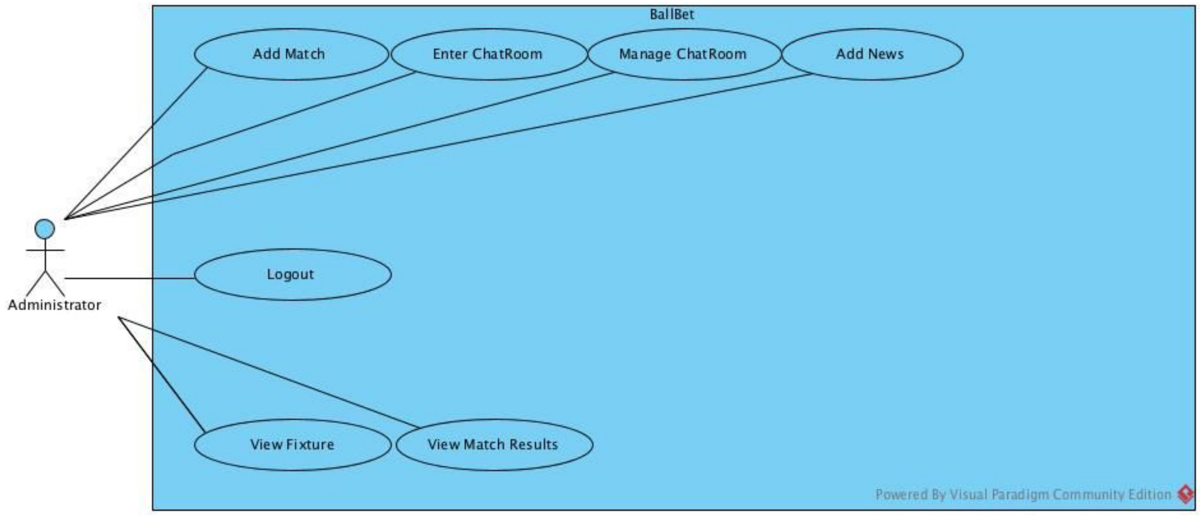
```
CREATE TABLE receives(
        userID INT,
        receiverID INT,
        messageID  INT,
        PRIMARY_KEY(senderID, receiverID, messageID),
        FOREIGN_KEY (userID) REFERENCES User(userID),
        FOREIGN_KEY (receiverID) REFERENCES User(userID),
        FOREIGN_KEY (messageID) REFERENCES Message(messageID))
        engine=InnoDB;
```

3. FUNCTIONAL COMPONENTS

3.1 Use Cases and Scenarios

Online Betting System is providing information to 4 kind of users. There are administrators, regular users, premium users and guests.There are common services provided for both regular and premium users however there are functionalities that can only be accessed by specific type of users in the system.

3.1.1 Administrator



Assumed user already signed in

Add Match

Administrators can add new matches to the Fixture. They will select the teams, betting amount declared by the Iddaa, the match date and referee.

Add News

Administrators can publish news at HomePage about unexpected match scores, user achievements or any other interesting events.

Enter ChatRoom

Administrators can be able to enter any chat room without entering any password to provide technical assistance about system and making observation of the dialogues for inappropriate language.

Manage ChatRoom

Administrators have all permissions in chat rooms. They can ban users that spam messages or using sexist, racist, foul words.

Logout

Logout function will guide user to homepage.

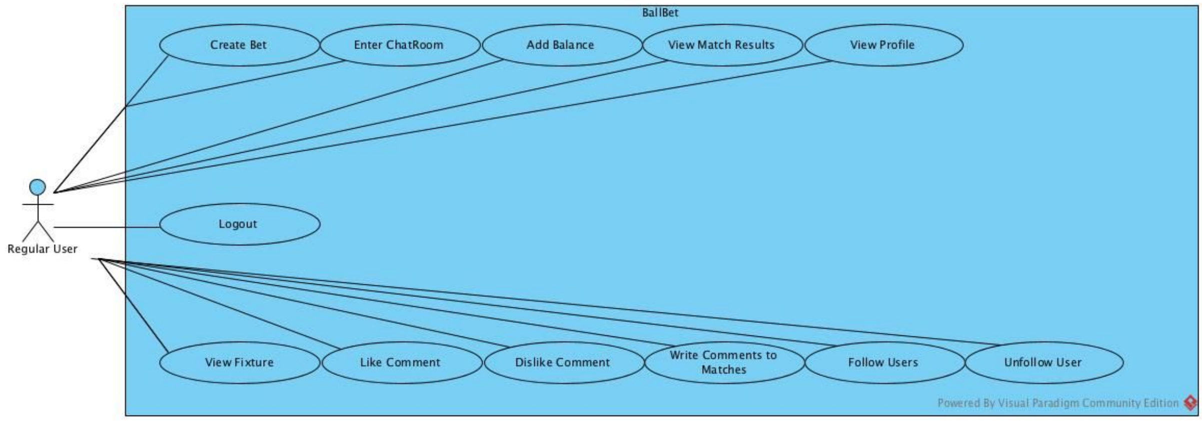
View Fixture

Administrators can view the information about upcoming matches from Programme page.

View Match Results

Administrators can view the results of the matches in Match Results section in Programme page.

3.1.2 Regular User



View Profile

Regular Users can view other users profiles by selecting their name on betslip.

View Match Results

Regular Users can view the results of the matches in Match Results section by visiting Programme page.

Create Bet

Regular Users can create bets by entering Create Bet page. They can also see other popular bets from this page.

Enter ChatRoom

Unlike Premium Users, Regular Users can not create private chat rooms. Therefor they can only enter existing chat rooms.

Add Balance

Regular Users can add balance to their profiles by using their debit cards.

View Fixture

Regular Users can view the upcoming matches by entering to Programme Page.

Like Comment

Regular Users can like other users comments and bets.

Dislike Comment

Regular Users can unlike their existing likes.

Write Comment to Matches

Regular Users may express their thoughts by writing comments to matches.

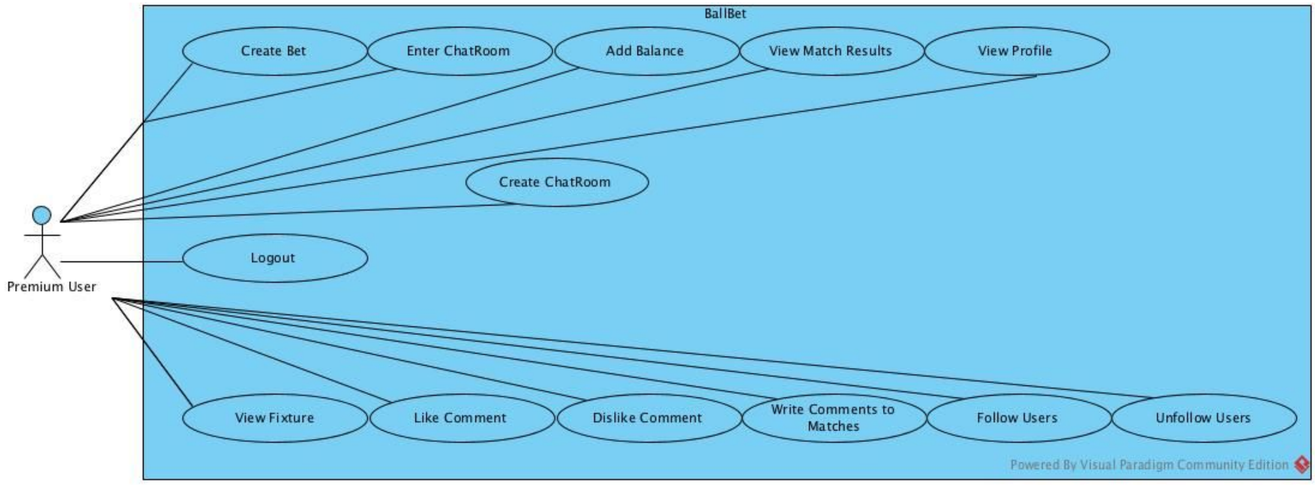
Follow Users

Regular Users may follow other users that they admire.

Unfollow Users

Regular Users may unfollow their existing followed users.

3.1.3 Premium User



View Match Results

Premium Users can view the results of the matches in Match Results section by visiting Programme page.

View Profile

Premium Users can view other users profiles by selecting their name on betslip.

Create Bet

Premium Users can create bets by entering Create Bet page. They can also see other popular bets from this page.

Enter ChatRoom

Premium Users may enter public chat rooms by visiting Chatrooms page.

Add Balance

Premium Users can add balance to their profiles by using their debit cards.

View Fixture

Premium Users can view the upcoming matches by entering to Programme Page.

Like Comment

Premium Users can like other users comments and bets.

Dislike Comment

Premium Users can unlike their existing likes.

Write Comment to Matches

Premium Users may express their thoughts by writing comments to matches.

Follow Users

Premium Users may follow other users that they admire.

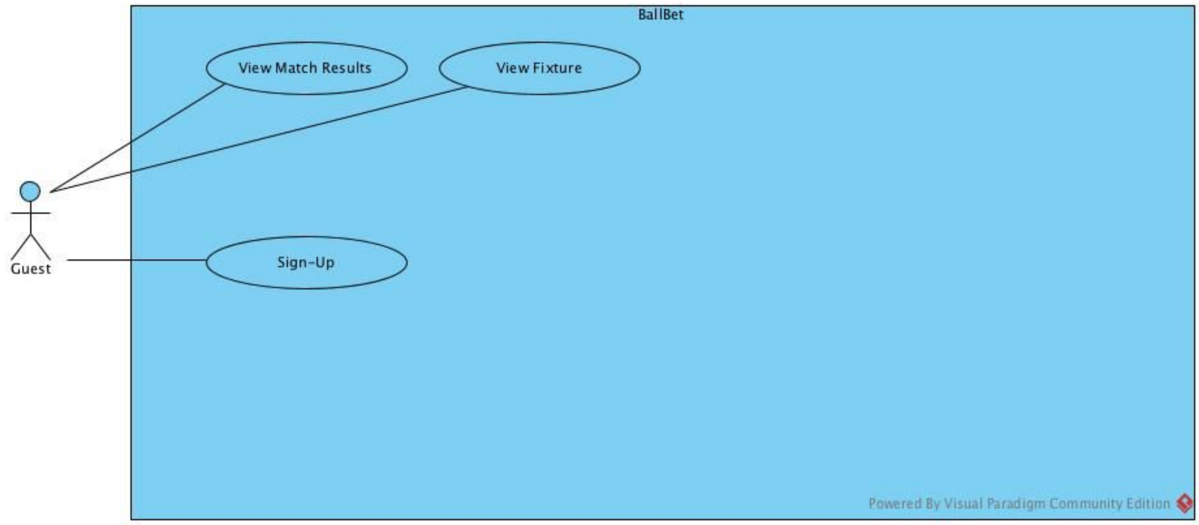
Unfollow Users

Premium Users may unfollow their existing followed users.

Create Room

Premium Users can create their private chat room by selecting create chat room in Chatrooms page.

3.1.4 Guest



View Match Results

Guests can view the results of the matches in Match Results section by visiting Programme page.

Sign Up

Guests can select sign up button in Homepage to register to system and gain access to functions of the website.

View Fixture

Guests can view the upcoming matches by entering to Programme Page.

3.2 Algorithms

3.2.1 Betslip Related Algorithms

As the user selects the matches from betting screen, some algorithms are needed in order to make a proper bet. A user cannot select the same match twice or more in a single betslip. This also eliminates contradicting bets. Moreover, each match has a minimum bet limit, this means that a user cannot select only three matches if a played match has a minimum bet limit of four. Users cannot bet on ongoing and ended matches, the date stamps are checked for each match displayed. The ratios of each bet is multiplied and updated on the generated betslip that user sees as the betslip is being created. After the user confirms the bets, he/she decides on the bet amount and bet type. Possible winnings are calculated on total ratio and bet amount. Users cannot bet more than the balance they have. To increase their balance, they have to add money to the balance via credit card.

3.2.2 Statistics Related Algorithms

As each game ends, updated information about matches will be displayed on the statistics page. Standings will change depending on the matches that are played. Users will be able to search teams and leagues to view their statistics and the search data will filter the displayed elements. Also, ended games will be displayed on the “Previous Games” part of the page, displaying the results and the winning bet.

3.2.3 Chatroom Algorithms

Users can socialize and discuss games at the chatrooms. However, there are some constraints in the chatrooms. If a chatroom’s size is reached, no new users can enter the chatroom. There are some chatrooms which are opened by the system and everybody is free to join to those rooms. Only premium users can open a chatroom but they are limited to 1. Premium Users cannot open a second channel. Premium users can decide the size of the chatroom and can decide to put a password on the channel. By doing so, other users can only join by entering the password to join the chatroom. Users cannot join to multiple chatrooms simultaneously.

3.2.4 Logical Requirements

Logical errors should be absent within our system as people will deposit their money and check here as a result. Dates should be strict especially in terms of boundary dates and they should be checked in a way that program should understand whether the information coming is sensitive or irrelevant.

There are a lot of ways for dates to fail as Match and League has respective dates. For example a match cannot be in th e League if the match's date is after the end_date of the respective League.

Moreover User cannot bet on a match after its date has passed. This is in grave importance as otherwise people can easily make money on the system.

3.3 Data Structures

The relation schemas used in the projects uses Numeric, String and Date data types.

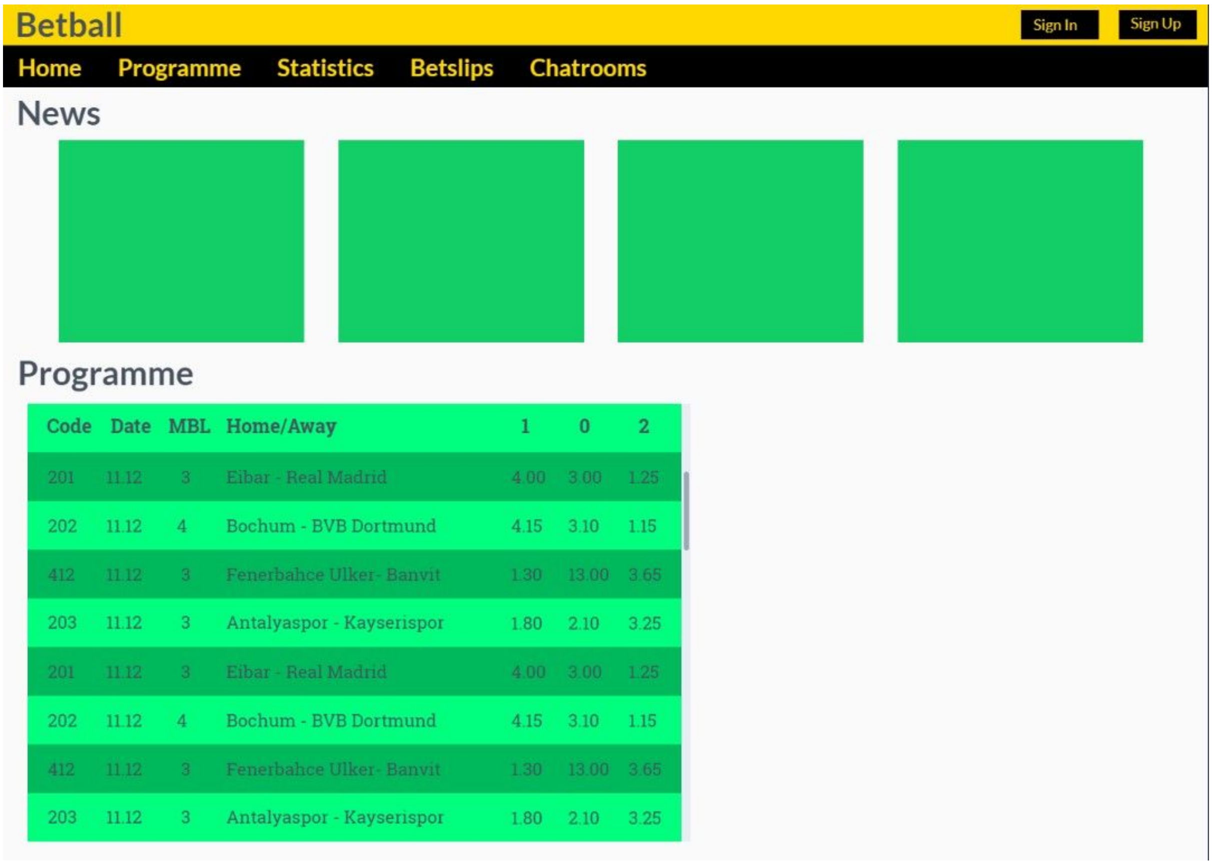
Numeric types are used for storing numeric data. INT data type used for ID's, amounts and counts. DOUBLE data type used for decimal numbers such as betting odds.

String types are used for storing attributes that consist of characters. VARCHAR is used for string attributes. Also, images in the system are stored with IMAGE data type.

For storing date and time values, system uses DATE data types.

4 User Interface Design

Home Page

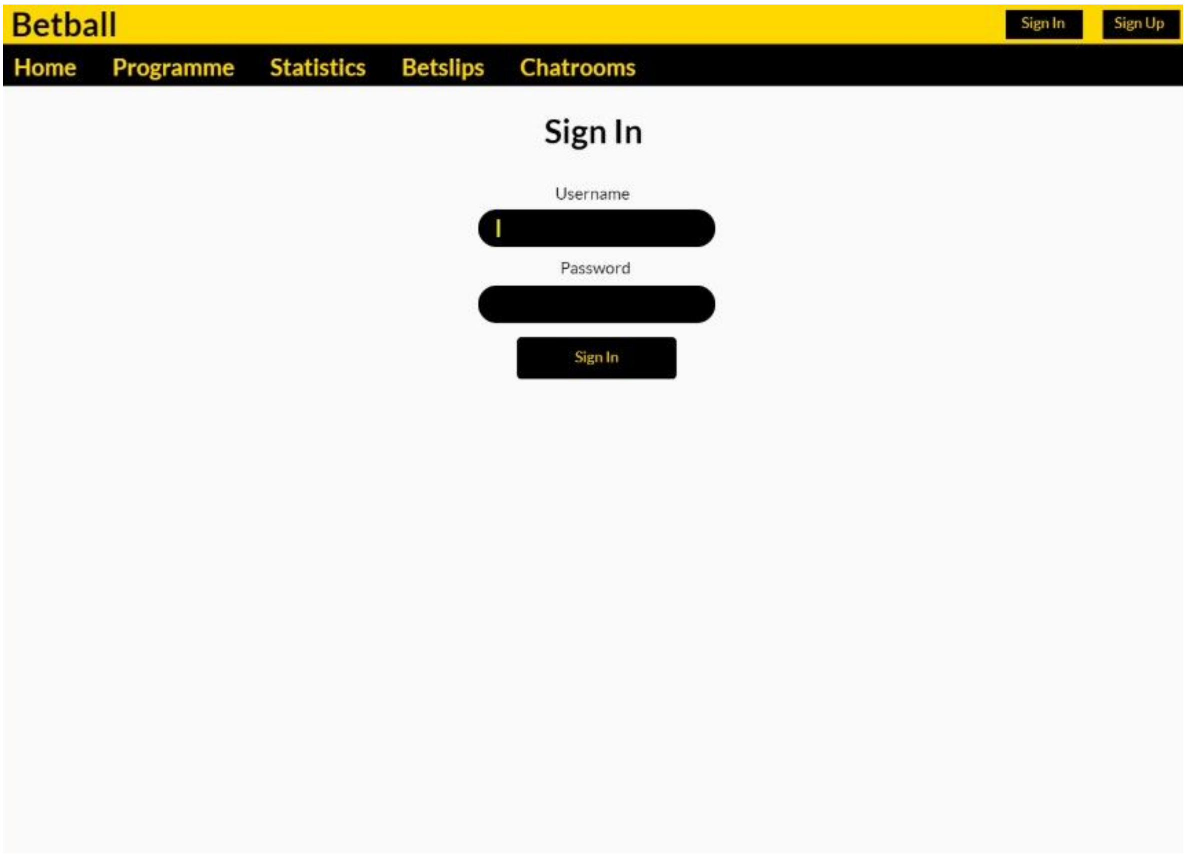


Code	Date	MBL	Home/Away	1	0	2
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25

```
SELECT *
FROM News;
```

```
SELECT *
FROM Match;
```

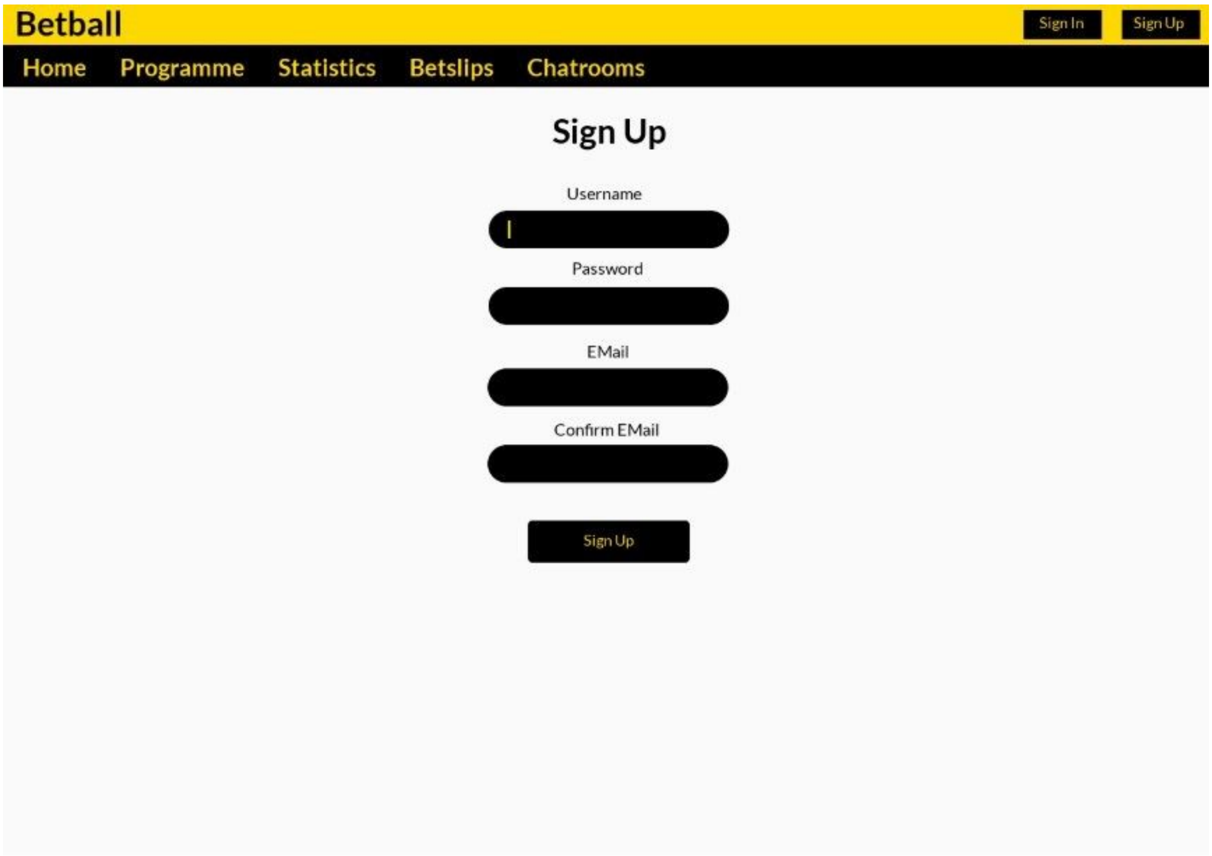
Sign In



Inputs: $username, $password

```
SELECT *
FROM User
WHERE (username =$username AND
                    password = $password)
```

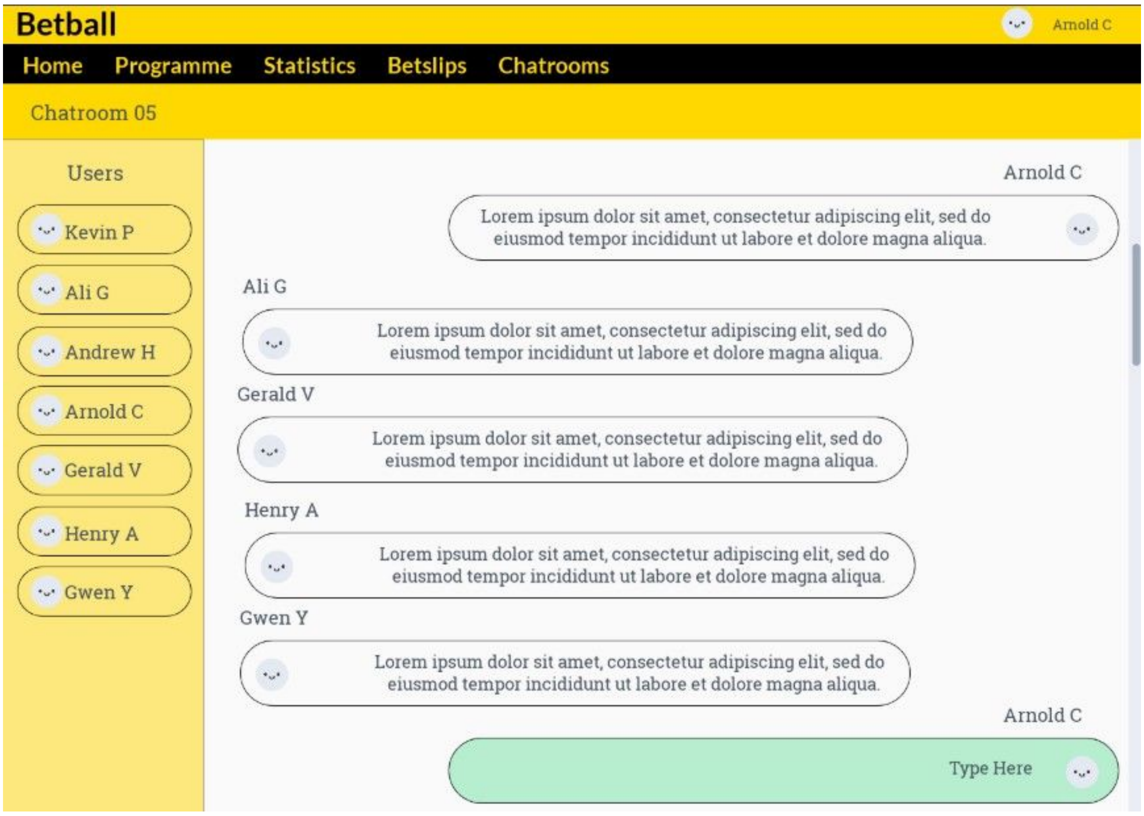
Sign Up



Inputs: $username, $password, $email

```
INSERT INTO User(username, password, email)
        VALUES ($username, $password, $email)
        WHERE (NOT EXISTS( SELECT *
                                      FROM User U
                                      WHERE U.username = $username)) and
                  (NOT EXISTS(        SELECT *
                                      FROM User U
                                      WHERE U.email = $email));
```

Chatroom



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

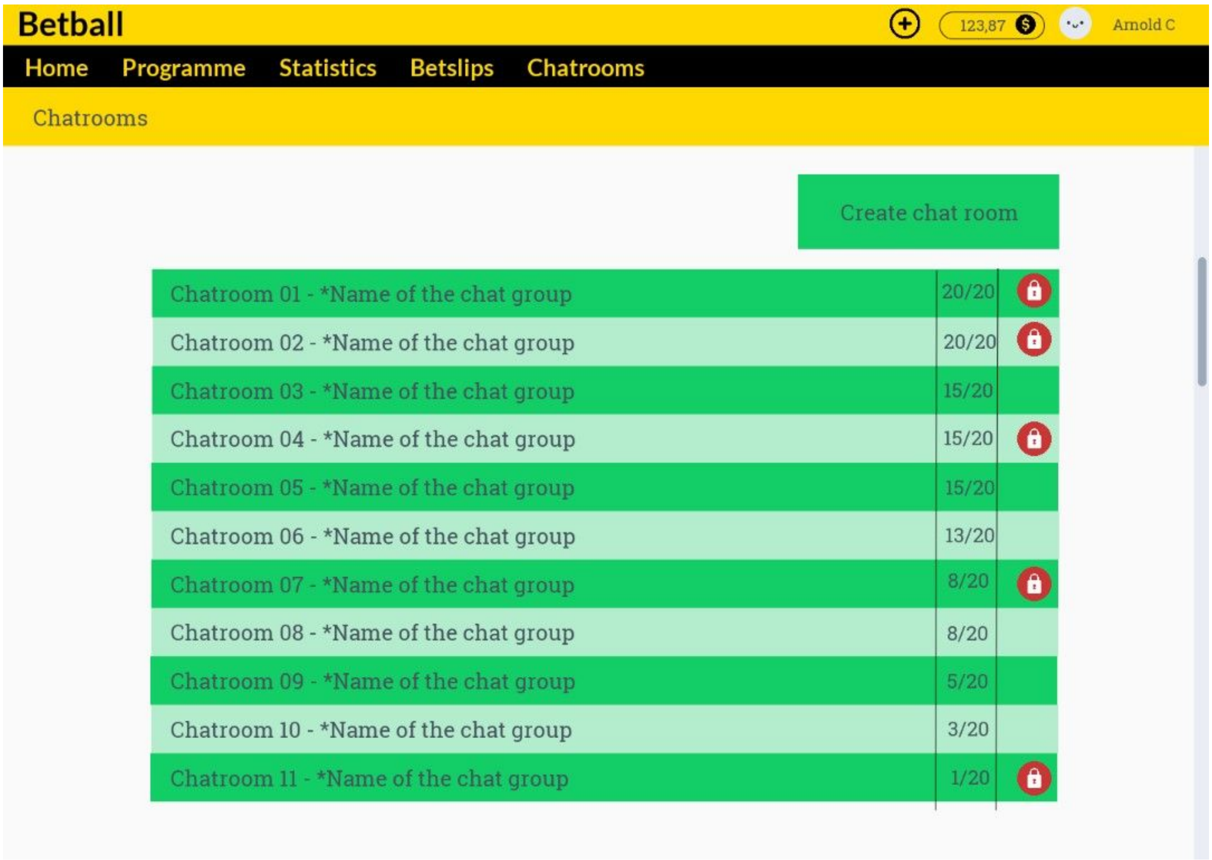
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Inputs: $username, $text, $roomID

```
INSERT INTO Message(userID, text)
        VALUES(SELECT userID
                        FROM user U
                        where U.username = $username, $text)

INSERT INTO sends
        VALUES (SELECT userID
                        FROM user U
                        where U.username = $username,     SELECT messageID
                                                          FROM Message
                                                          ORDER BY messageID DESC
                                                          LIMIT 1)

INSERT INTO kept_in
        VALUES ($roomID,     SELECT messageID
                             FROM Message
                             ORDER BY messageID DESC
                             LIMIT 1)
```

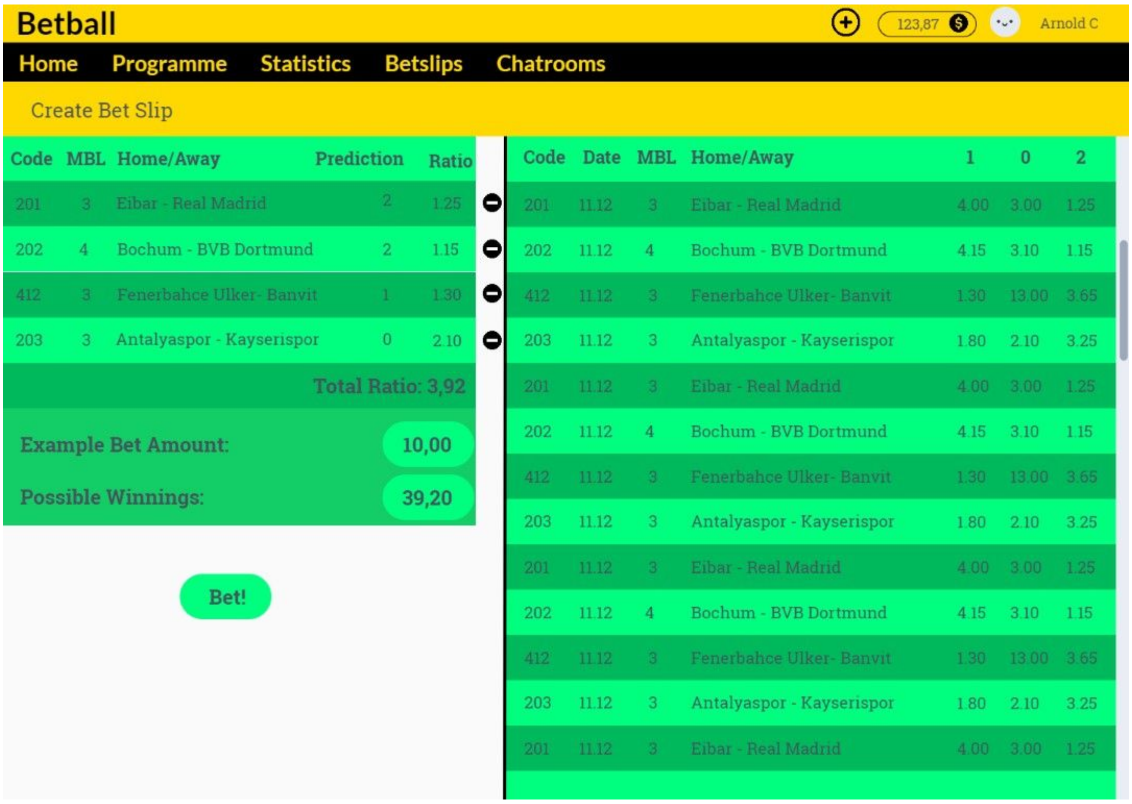
Chatrooms



Chatroom 01 - *Name of the chat group	20/20	🔒
Chatroom 02 - *Name of the chat group	20/20	🔒
Chatroom 03 - *Name of the chat group	15/20	
Chatroom 04 - *Name of the chat group	15/20	🔒
Chatroom 05 - *Name of the chat group	15/20	
Chatroom 06 - *Name of the chat group	13/20	
Chatroom 07 - *Name of the chat group	8/20	🔒
Chatroom 08 - *Name of the chat group	8/20	
Chatroom 09 - *Name of the chat group	5/20	
Chatroom 10 - *Name of the chat group	3/20	
Chatroom 11 - *Name of the chat group	1/20	🔒

```
SELECT *
FROM Chatroom;
```

Create Bet Slip



Code	MBL	Home/Away	Prediction	Ratio
201	3	Eibar - Real Madrid	2	1.25
202	4	Bochum - BVB Dortmund	2	1.15
412	3	Fenerbahce Ulker- Banvit	1	1.30
203	3	Antalyaspor - Kayserispor	0	2.10

Total Ratio: 3,92

Example Bet Amount: 10,00

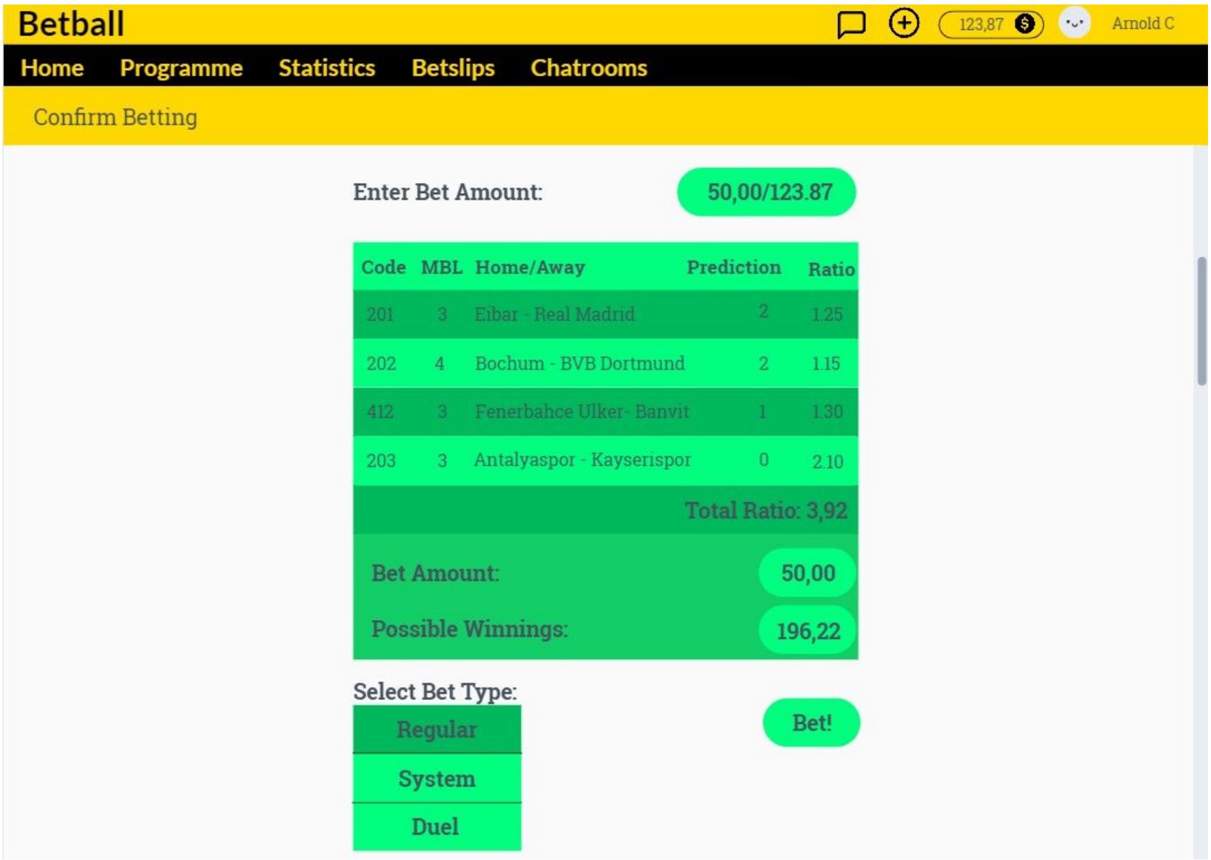
Possible Winnings: 39,20

Code	Date	MBL	Home/Away	1	0	2
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25

```
SELECT *
FROM Match;
WHERE DATEDIFF(match_Date, CURRENT_TIMESTAMP()) < 10
```

Since the betslip is not added to the database yet, we don’t do anything with it yet.

Confirm Bet Slip



Enter Bet Amount: 50,00/123.87

Code	MBL	Home/Away	Prediction	Ratio
201	3	Eibar - Real Madrid	2	1.25
202	4	Bochum - BVB Dortmund	2	1.15
412	3	Fenerbahce Ulker- Banvit	1	1.30
203	3	Antalyaspor - Kayserispor	0	2.10

Total Ratio: 3,92

Bet Amount: 50,00

Possible Winnings: 196,22

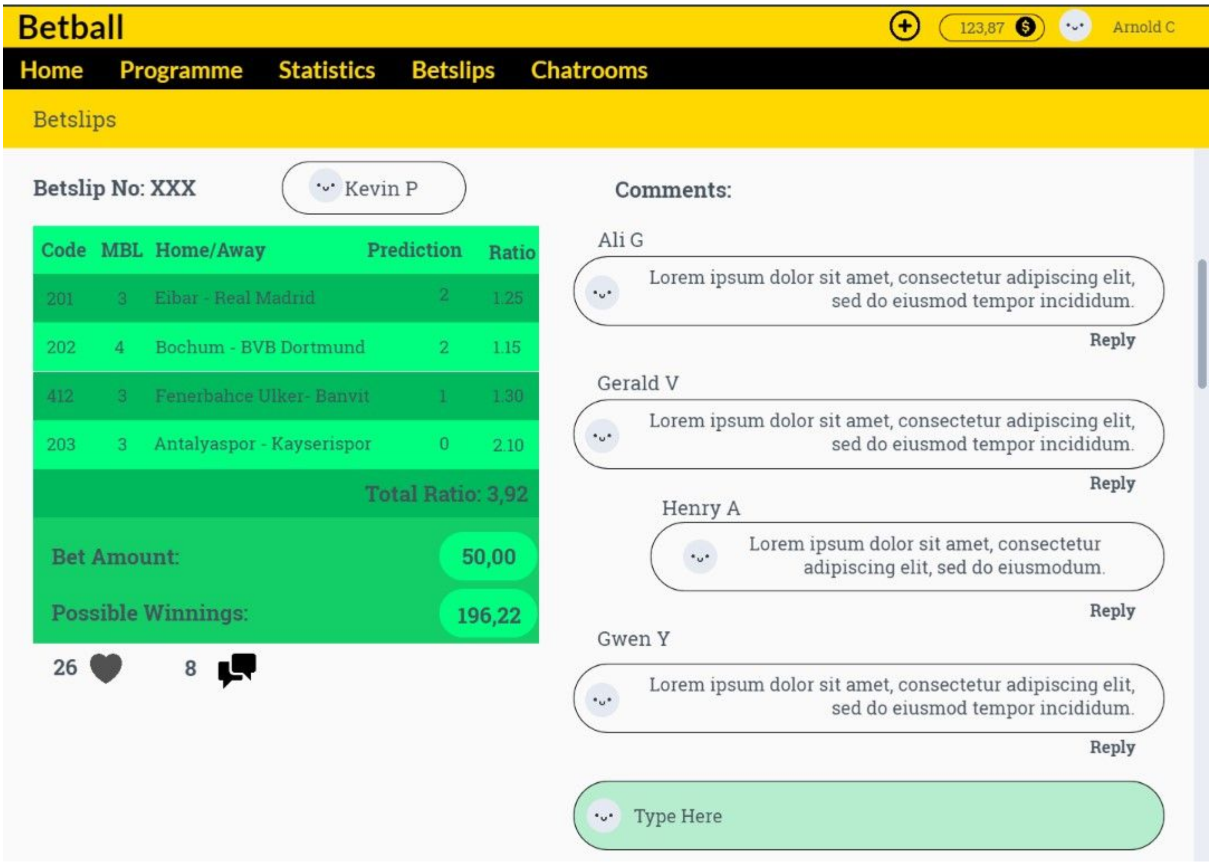
Select Bet Type:

Inputs: $username, $matchID (for each match), $prediction (for each match), $betAmount, $bettingType

```
INSERT INTO BetSlip(betting_amout)
        VALUES ($betAmount);

INSERT INTO BettingType(betting_type)
        VALUES($bettingType);

INSERT INTO arranges
        VALUES(         SELECT slipID
                        FROM BetSlip
                        ORDER BY slipID DESC
                        LIMIT 1,                        SELECT betting_type
                                                        FROM BettingType
                                                        ORDER BY betting_type DESC
                                                        LIMIT 1);

For each match
{
```

```
INSERT INTO Bet
        VALUES ($prediction)

INSERT INTO played_on
        VALUES(         SELECT betID
                        FROM Bet
                        ORDER BY betID DESC
                        LIMIT 1, $matchID)

INSERT INTO contain_bet
        VALUES(         SELECT slipID
                        FROM BetSlip
                        ORDER BY slipID DESC
                        LIMIT 1,                        SELECT betID
                                                        FROM Bet
                                                        ORDER BY betID DESC
                                                        LIMIT 1)
}
```

Betslip Social (Comment and Likes)



Betslip No: XXX

Code	MBL	Home/Away	Prediction	Ratio
201	3	Eibar - Real Madrid	2	1.25
202	4	Bochum - BVB Dortmund	2	1.15
412	3	Fenerbahce Ulker- Banvit	1	1.30
203	3	Antalyaspor - Kayserispor	0	2.10
			Total Ratio: 3,92	

Bet Amount:	50,00
Possible Winnings:	196,22

Comments:

Ali G

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididum.

Gerald V

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididum.

Henry A

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmodum.

Gwen Y

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididum.

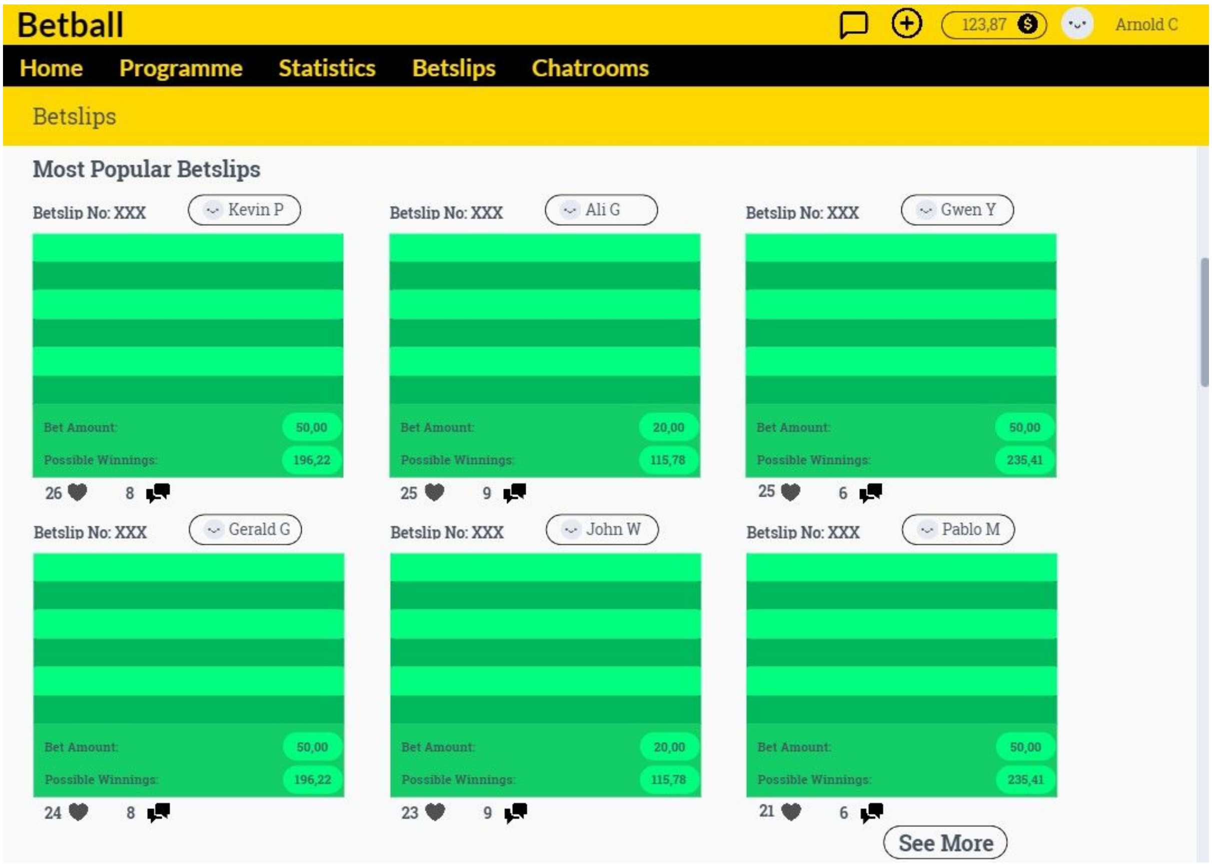
Inputs: $slipID, $username

```
INSERT INTO Comment(userID, text)
        VALUES(SELECT userID
                        FROM user U
                        where U.username = $username, $text)

INSERT INTO writes
        VALUES (SELECT userID
                        FROM user U
                        where U.username = $username,    SELECT commentID
                                                         FROM Comment
                                                         ORDER BY commentID DESC
                                                         LIMIT 1)
                                                         INSERT INTO written_on
        VALUES (        SELECT commentID
                        FROM Comment
                        ORDER BY commentID DESC
                        LIMIT 1, $slipID)
```

Outputs:

```
SELECT C.text
WHERE Betslip S, Match M, Comments C
FROM (S.slipID = $slipID and S.slipID = C.slipID)
```

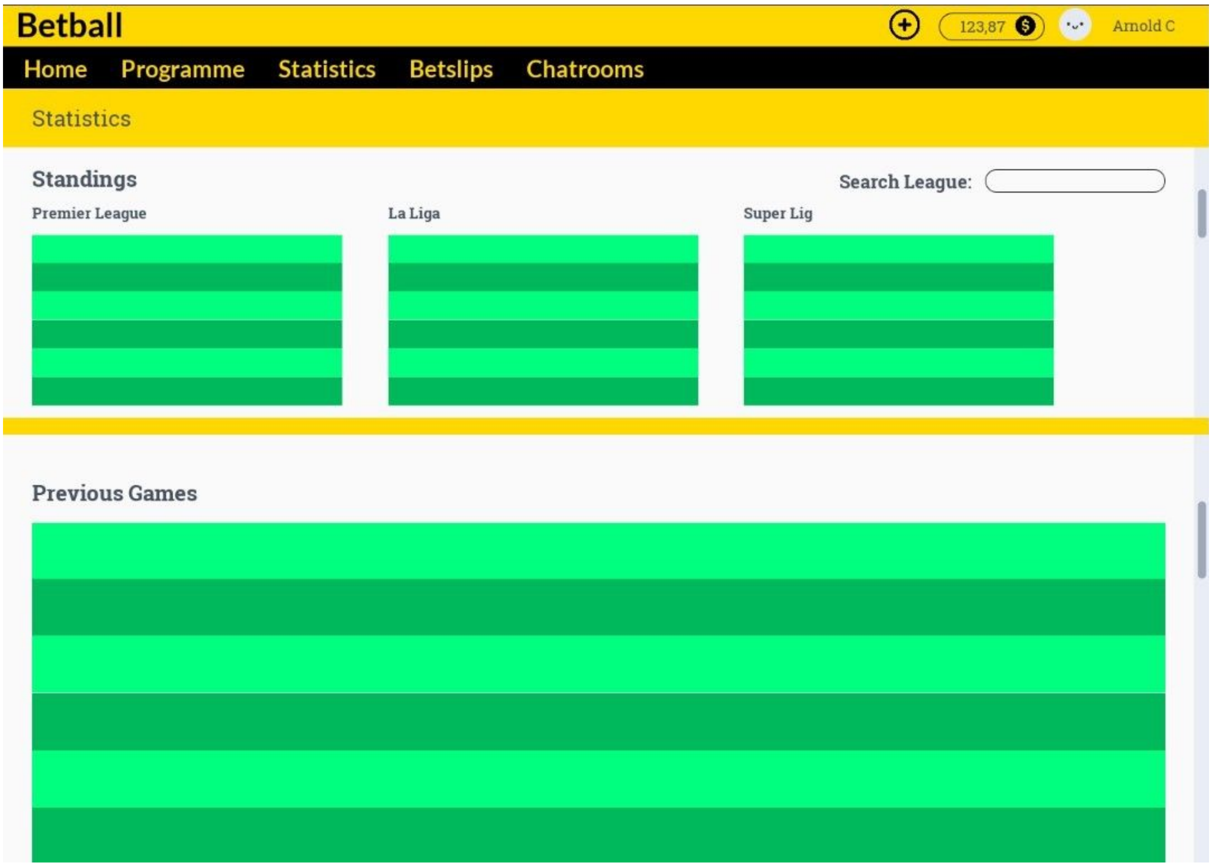
Betslips



```
SELECT *
```

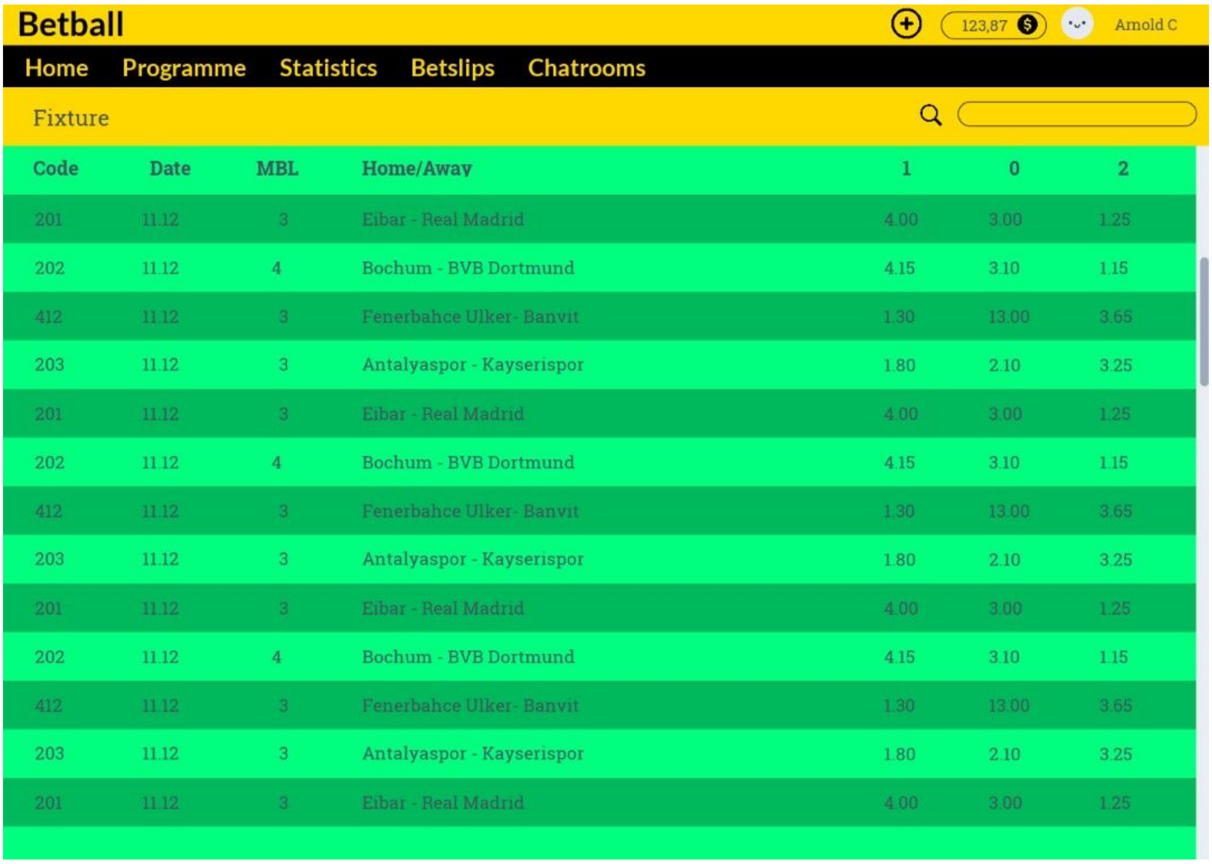
```
FROM Betslip
ORDER BY likes DESC
LIMIT 6;
```

Statistics



Outputs:
Finished Matches View that can be found at 5.1.1
Most Winning Users View that can be found at 5.1.2
League Ranking Viewthat can be found at 5.1.3

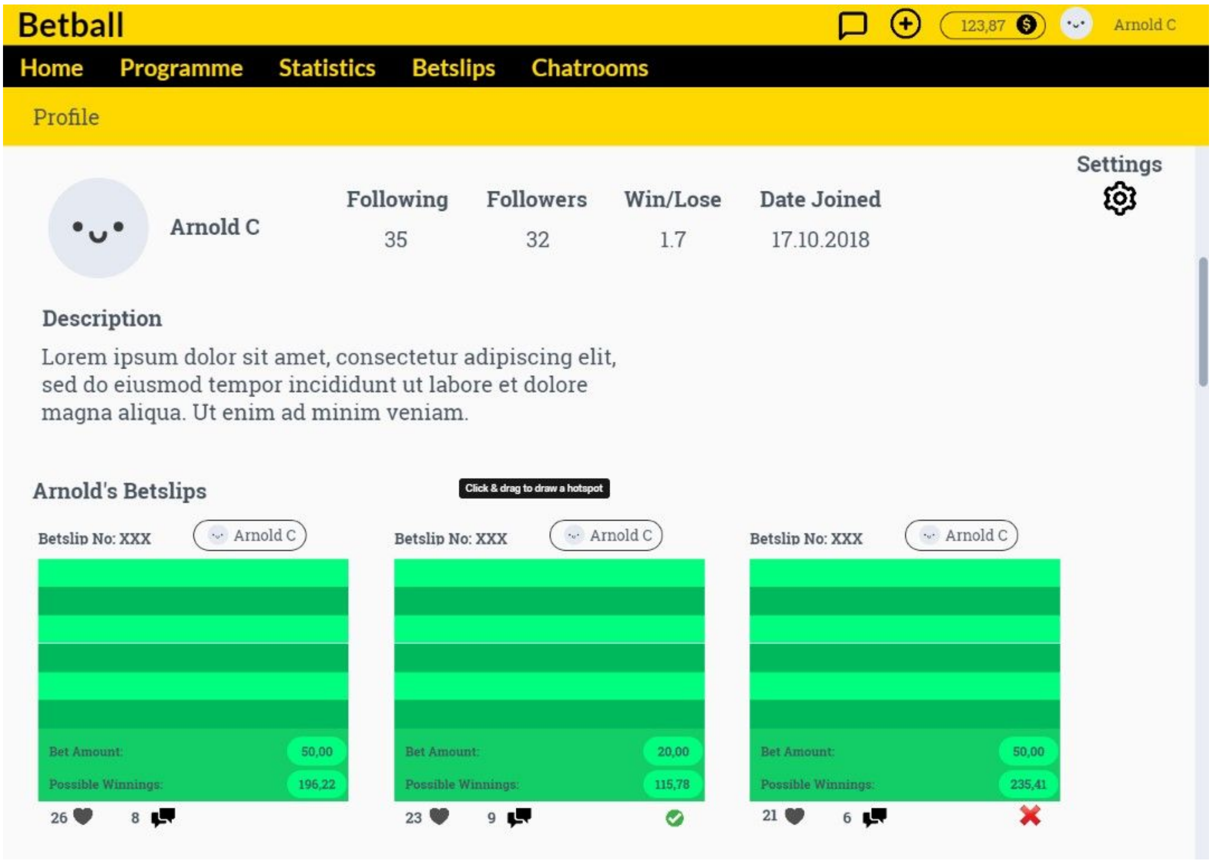
Fixture



Code	Date	MBL	Home/Away	1	0	2
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25
202	11.12	4	Bochum - BVB Dortmund	4.15	3.10	1.15
412	11.12	3	Fenerbahce Ulker- Banvit	1.30	13.00	3.65
203	11.12	3	Antalyaspor - Kayserispor	1.80	2.10	3.25
201	11.12	3	Eibar - Real Madrid	4.00	3.00	1.25

```
SELECT *
FROM Match;
```

User Profile



Arnold C

Following	Followers	Win/Lose	Date Joined
35	32	1.7	17.10.2018

Description

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

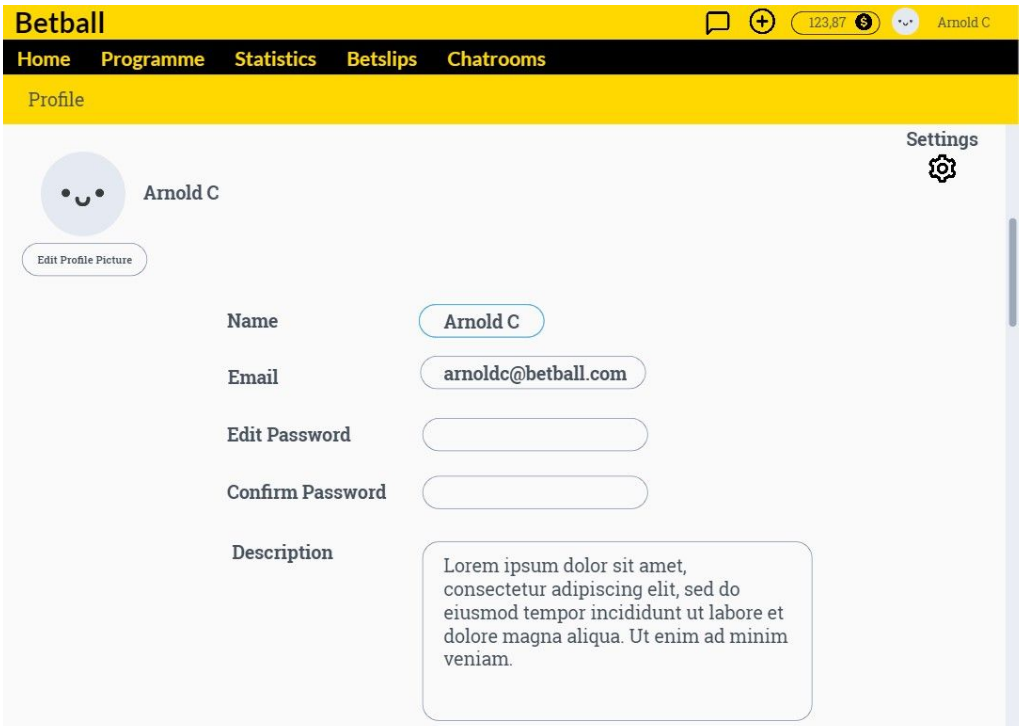
Arnold's Betslips

Betslip No: XXX	Betslip No: XXX	Betslip No: XXX
Arnold C	Arnold C	Arnold C
Bet Amount: 50,00	Bet Amount: 20,00	Bet Amount: 50,00
Possible Winnings: 196,22	Possible Winnings: 115,78	Possible Winnings: 235,41
26 ♥ 8	23 ♥ 9 ✓	21 ♥ 6 ✗

Inputs: $username

```
SELECT U.username, U.description, U.win_ratio, U.balance, U.photo, U.date_joined,
                     B.slipID, B.betting_amount, B.likes, B.comments
FROM User U, Betslip B
WHERE (U.username = $username and B.username = $username)
```

User Settings



Name: Arnold C

Email: arnoldc@betball.com

Edit Password:

Confirm Password:

Description: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Inputs: @newusername, @newpassword, @newemail, @newdescription, @username, @password, @email, @description

Change Username

```
UPDATE User
SET username = @newusername
WHERE username = @username
```

Change Password

```
UPDATE User
SET password= @newpassword
WHERE username = @username
```

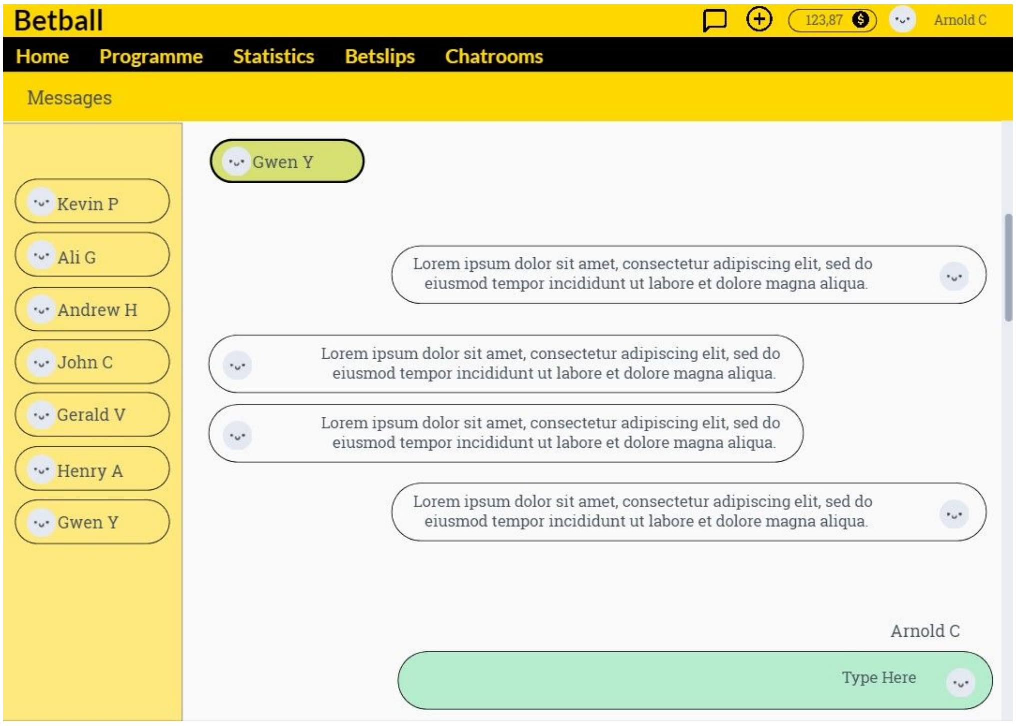
Change E-mail

```
UPDATE User
SET email= @newemail
WHERE username = @username
```

Change Description

```
UPDATE User
SET description = @newdescription
WHERE username = @username
```

Messages



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

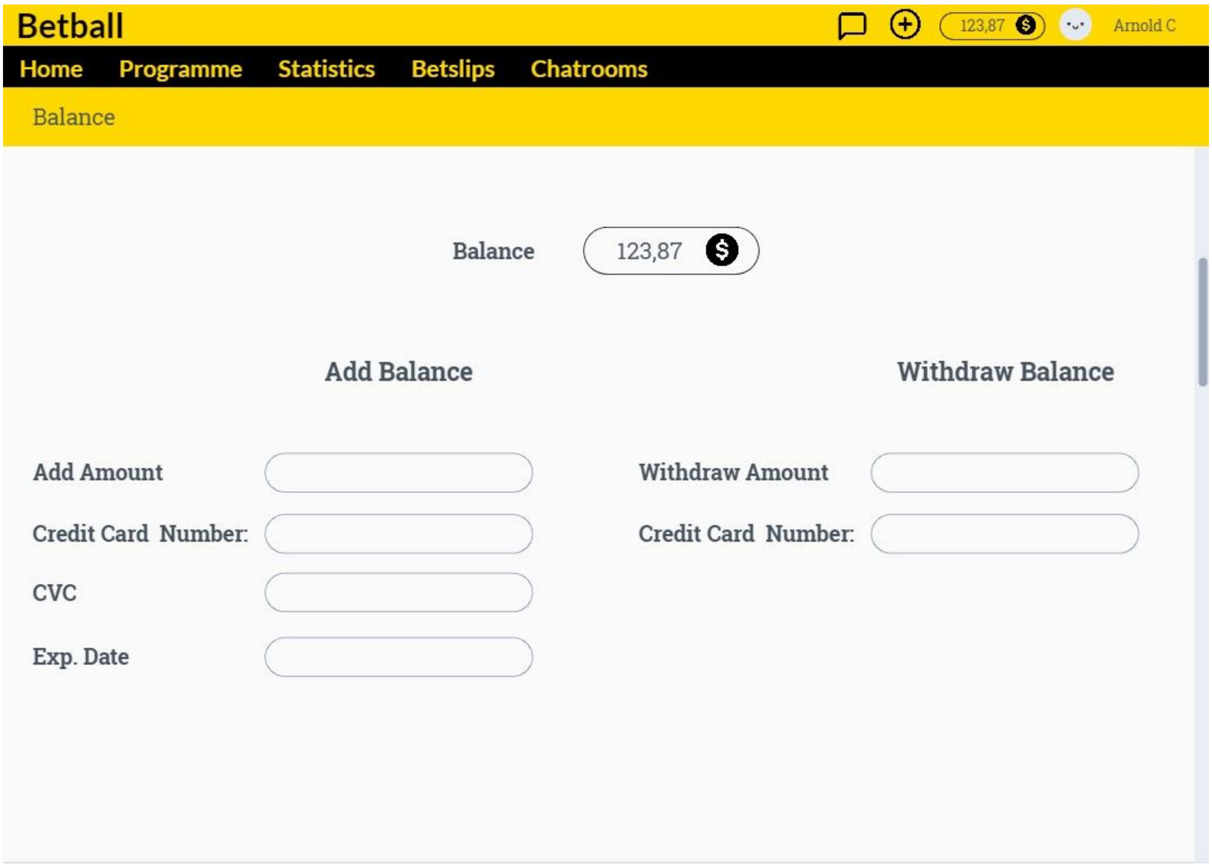
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Inputs: $username, $text, $receiverID

```
INSERT INTO Message(userID, text)
      VALUES(SELECT userID
                   FROM user U
                   where U.username = $username, $text)

INSERT INTO sends
      VALUES (SELECT userID
                   FROM user U
                   where U.username = $username,   SELECT messageID
                                                   FROM Message
                                                   ORDER BY messageID DESC
                                                   LIMIT 1)
INSERT INTO receives
      VALUES (SELECT userID
                   FROM user U
                   where U.username = $username, $receiverID,
                                                   SELECT messageID
                                                   FROM Message
                                                   ORDER BY messageID DESC
                                                   LIMIT 1);
```

Balance



Inputs: $username, $balance, $add_amount, $withdraw_amount

Add Balance

```
UPDATE User
SET balance = $balance + $add_amount
WHERE username = $username
              AND balance = $balance
```

Withdraw Balance

```
UPDATE User
SET balance =$ balance - $withdraw_amount
WHERE username = $username
              AND balance = $balance
```

5. Advanced Database Components

5.1 Views

5.1.1 Finished Matches View

This view will show the matches that finished in the last week and their results.

```
CREATE VIEW finished_matches AS
        SELECT *
        FROM Match
        WHERE DATEDIFF(CURRENT_TIMESTAMP(), created_date) < 7
        GROUP BY slipID;
```

5.1.2 Most Winning Users View

This view will show the users who have the biggest winning ratio.

```
CREATE VIEW user_win AS
        SELECT *
        FROM User
        ORDER BY win_ratio DESC
        LIMIT 10;
```

5.1.3 League Ranking View

This view will show any league's ranking from the most winning team the least.

Asked league's ID will be given beforehand as asked_leagueID.

```
CREATE VIEW league_ranking AS
        SELECT *
        FROM Team JOIN consist_of
        WHERE leagueID=asked_leagueID
        ORDER BY win_count DESC;
```

5.2 Reports

5.2.1 User Monthly Report

Users will be presented with a report of their total losses/wins and how many bets did they lose with how many matches so that they can see how close they were to winning and keep playing.

$userID is kept in the session when the user logs in and keeps the userID.

```
CREATE VIEW monthly_user_report AS
        SELECT result
        FROM (Betslip JOIN create_betslip as T)
        WHERE T.userID=$userID and
                DATEDIFF(created_date, CURRENT_TIMESTAMP()) < 30
        GROUP BY slipID;
```

5.2.2 New Comments Report

This report will be given to admin so that he/she can examine the comments and delete the foul ones.

```
CREATE VIEW new_comments_report AS
        SELECT text
        FROM Comment
        WHERE DATEDIFF(date, CURRENT_TIMESTAMP()) < 30;
```

5.3 Triggers

- When a Betslip is created related User account should be updated.
- When a Match has ended related Team and League tables should be updated. Moreover betslip’s that predicted this match’s result false should be finalized and marked as lose and user’s win_ratio should be updated.
- When a Comment or Betslip got liked/disliked their likes/dislikes should be updated.
- When a User starts following or getting followed

5.4 Constraints

- Betting amount cannot exceed the balance of the user and can't be smaller than the slip's minimum bet amount.
- Users cannot join to a Chatroom if it is full.
- Count of bets can't be smaller than the slip's minimum bet count.

6. Implementation Plan

To implement BetBall, it is planned to use Bootsrap, HTML and CSS for front-end development, JavaScript for back-end development and MySQL for database implementation. Programs planned to be used are FileZilla and Wamp for server operations and Visual Studio for coding.

7. References

- https://github.com/sekin72/Database-Betting-Project
- https://www.nesine.com
- https://www.iddaa.com
- https://www.tuttur.com