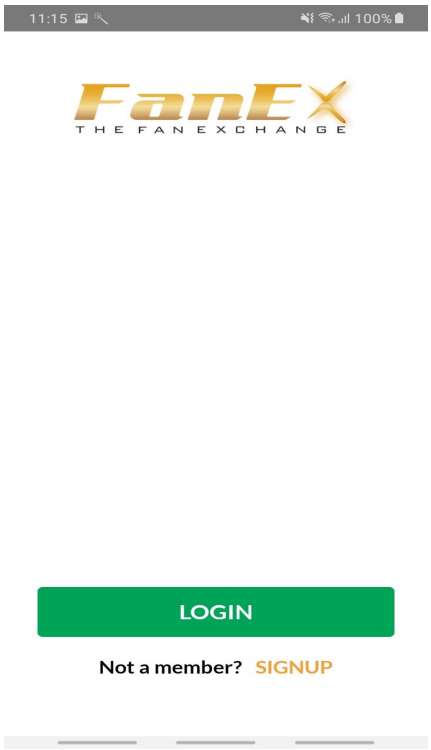
SPLASH SCREEN

Screen Name:-- SplashActivity
Xml page name:-- *activity_splash*
Here I have checked that it is a logged in user or not, if it is not a logged in user page will automatically redirect to ChooseLoginSignupActivity else it will call a method getAccountDetails(accounts/ & account_id).
From this method api return a value hasJoinedContest if it is true means user already joined a contest then page redirect to dashboard page else call another method name is getFirstSignuplobbydata(competition/firstSignUpLobby) and page redirect to SelectMVPActivity
On this page there is another method printHashKey. It is required to generate HashKey for facebook development.

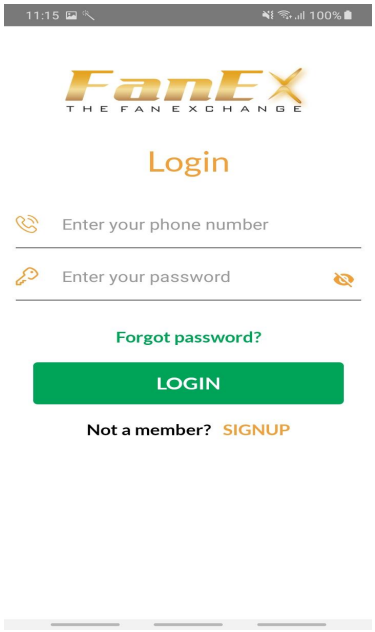
CHOOSE LOGIN SIGNUP



Screen Name:- ChooseLoginSignupActivity
Xml page name:- activity_choose_login_signup
In this page there is two button

1. LOGIN
2. SIGNUP

When user pressed LOGIN button it will redirect to LoginActivity
And when user pressed SIGNUP button it will redirect to SignupActivity



LOGIN SCREEN

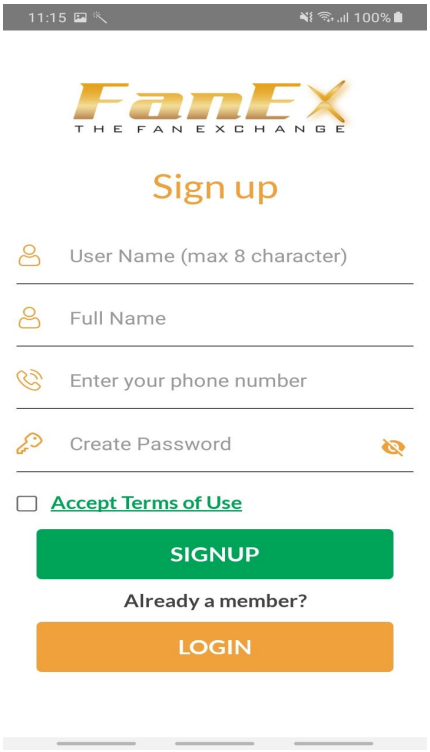
Screen Name:- LoginActivity
Xml Page Name:- activity_login
There is two input field

1. Enter phone number
2. Enter password

One eye option to show and hide the password.
When user pressed the login button first check input fields are blank or not , if both are blank show snack bar with message otherwise call this method userLogin(accounts/login/,params) params is given below

```
HashMap<String, String> params = new HashMap<>();
params.put("email", emailphoneno);
params.put("password", password);
params.put("deviceId", android_id);
params.put("deviceType", "android");
```

If a user logged in successfully, api returns some response and I have stored some data to sharedPreference (i.e token, userid, accout_id, hasJoinedContest) . Now checked hasJoinedContest is true or false if true page will redirect to DashboardActivity else call another method name is getFirstsignuplobbydata(competition/firstSignUpLobby) and page redirect to SelectMVPActivity. When a user pressed the Forgot Password? Option page will redirect to ForgotPasswordActivity. And when a user pressed Not a member ? SIGNUP option page will redirect to SignupActivity.



SIGNUP SCREEN

Screen Name:- SignupActivity
Xml Page Name:- activity_signup
There is four input field

1. Enter user name(max 8 character)
2. Enter Full name
3. Enter phone number
4. Enter Password

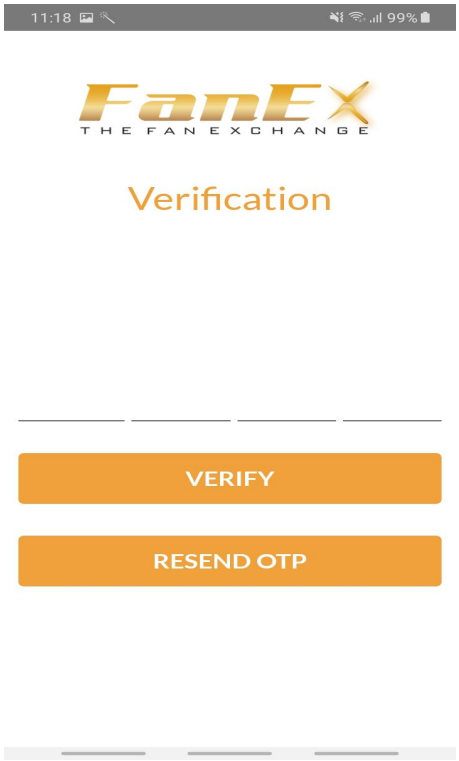
One eye option to show and hide the password.
One check box for accept terms of use
When user pressed the signup button first check all input fields are blank or not , if both are blank show snack bar with message otherwise call this method checkusernamewxistsornot (accounts/checkUsername/,params) params is given below

```
HashMap<String, String> params = new HashMap<>();
params.put("username", username);//username that is given input by the user.
```

If api response returns ack1 then call another method signupDataSubmit() if response return ack0 means username already exists. In signupDataSubmit() method i have checked full name validation, phone no validation and password validation after successfully validation I am calling userSignup(accounts/,params) with params

```
HashMap<String, String> params = new HashMap<>();
params.put("email", emailorphone);
params.put("user.first_name", firstname);
params.put("user.last_name", lastname);
params.put("user.username", username);
params.put("user.password", password);
params.put("about", "");
params.put("phone", "");
params.put("countryCode", countrycodewithplus);
params.put("address", myaddress);
params.put("latitude", latitude.toString());
params.put("longitude", longitude.toString());
params.put("deviceId", android_id);
params.put("deviceType", "android");
userSignup(params, emailorphone);
```

After successfully registering api return user_id, account_id,token,hasJoinedContest. I have stored phone no and hasJoinedContest value to sharedPreferences and page will redirect to OtpVerificationActivity page.



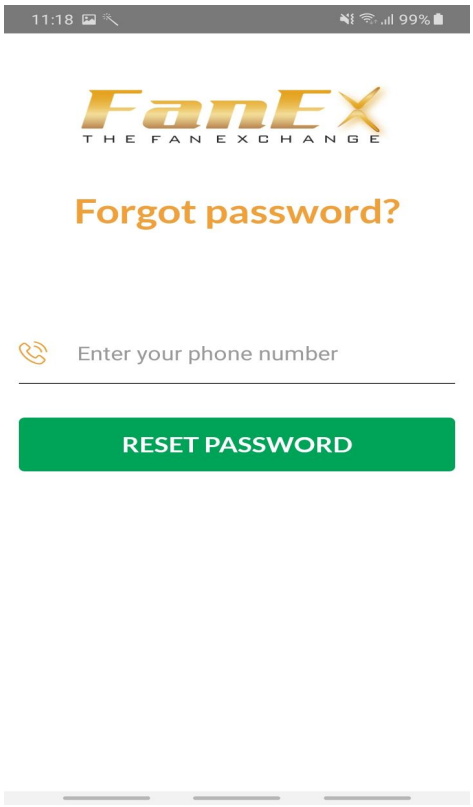
OTP VERIFY SCREEN

Screen Name: OtpVerification Activity
Xml page name: activity_otp_verification
If a user has received the OTP then enter the OTP here and pressed the verify button to verify the the given OTP. If not received any otp user will pressed the Resend OTP button for the new OTP.

```
/**
* OTP verification api params
* @param email (email represent the user phone no)
* @param otp
*/
HashMap<String, String> params = new HashMap<>();
params.put("email",
AppController.getUserPref().string(UserPref.PHONE));
params.put("otp", otp1val + otp2val + otp3val + otp4val);
Api name accounts/verifyOtp/
otpVerification(params);
```

After successful OTP verification users will redirect to WelcomeActivity if hasJoinedontest is false and if hasJoinedCOntest is true page will redirect to DashboardActivity. If user comes from ForgotPassword page then page will redirect to ResetPasswordActivity
If user has not received any OTP then user will pressed the resend OTP button for send the OTP again.

```
/**
* Resend OTP api params
* @param email (email represent the user phone no)
*/
HashMap<String, String> params = new HashMap<>();
params.put("email",
AppController.getUserPref().string(UserPref.PHONE));
resendOtp(params);
```

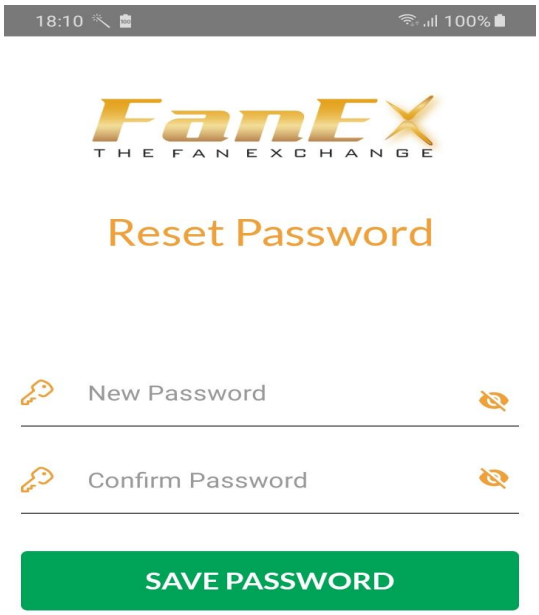


FORGOT PASSWORD SCREEN

Screen Name: ForgotPasswordActivity
Xml page name: activity_forgot_password
When a user presses the ForgotPassword button from the Login page it will redirect to this page. Here users will input the phone no for Reset Password. After valid input one api is calling here for receive the otp

```
/**
* Get otp for reset password api params
* @param email (email represent the user phone no)
*/
HashMap<String, String> params = new HashMap<>();
params.put("email", emailorphone);
forgotPassword(params);
Api name is accounts/forgotPassword/
```

After successful api calling page will redirect to OtpVerificationActivity. After OTP verification page will redirect to ResetPasswordActivity

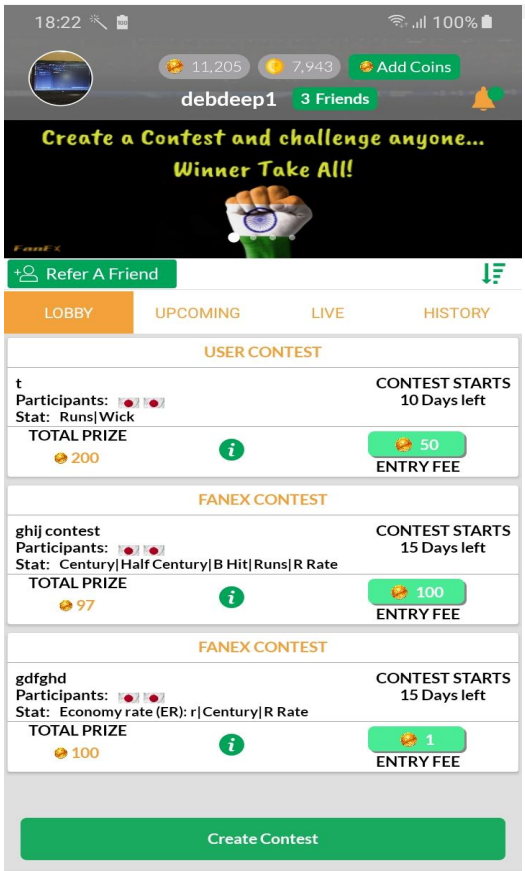


RESET PASSWORD SCREEN

Screen Name: ResetPasswordActivity
Xml Page name : activity_reset_password
Here users input new password and confirm password and there is an option to view password and hide password. After that one api is calling with params that is

```
/**
* new password api params
* @param newPassword
*/
HashMap<String, String> params = new HashMap<>();
params.put("newPassword", newpassword);
resetPassword(params);
```

Api name is accounts/resetPassword/
After successful api calling page will redirect to LoginActivity



DASHBOARD SCREEN

Screen Name: DashboardActivity
Xml page name: activity_dashboard
In this page there is Four Module

1. Header section
2. Banner section
3. Lobby,Upcoming,Live,History section
4. Footer section

For Header section one api is called method name is

```
/**
* Account details api call
* @param account_id
*/
getaccountdetails(FantexUrl.sGetUserDetails +
AppController.getUserPref().string(UserPref.ACCOUNT_ID),"");
```

Api name is accounts/
After successful calling this api another method is calling for load all banners method name is

getBannerImage()
Api name is extras/banners/
Others method are
PriceDropDialog() It is called when the wallet coin value is less than 100.
When a user pulls to refresh the whole page from top then override the onRefresh() method is called. shareVia() is called when users refer the app to other users.
sortByDialogWithRadioGroup() is called when a user clicks on the sort button.
When a user clicks on the AddCoins button page will redirect to RewardActivity page.
When a user clicks on Friends button page will redirect to FriendsActivity page.
When a user clicks on the Notification button page will redirect to NotificationActivity page.
If UserTotalnumberNotifications is 0 then show a message “There is no notification for you !!” Otherwise it will redirect to NotificationActivity page. If numberNotifications is greater than 0 then the green notification bubble is showing otherwise it is not showing.
sortBy() method is called for sorting the data according to the sort parameter.
Footer Section is common for all pages the page name in FooterFragment.
Lobby Fragment is used for loading lobby data.
Upcoming Fragment is used for loading upcoming data.
Live Fragment is used for loading live data.
History Fragment is used for loading history data.
fragment-->Lobby
getLobbyData() is used to Fetch Lobby data from api . Api name is competition/lobby/ .
hideCreateContestViews() is used to hide Create contest view when user scroll down
showCreateContestViews() is used to show Create contest view when user scroll to top
beginSort() is used to sort data accordingly like Name, Date, Prize, Entry Fee
fragment-->Upcoming
getUpcomingData() is used to fetch Upcoming data from api. Api name is competition/upcoming/.
beginSort() is used to sort data accordingly like Name, Date, Prize, Entry Fee
fragment-->Live
getLiveData() is used to fetch Live data from api. Api name is competition/live/.
When the user clicks on the view button from the live contest another api will be called for details of the live match. The name is competition/standings?contestId=”contestID”
To view individual player list details need to call another api the name is competition/playerDetails/

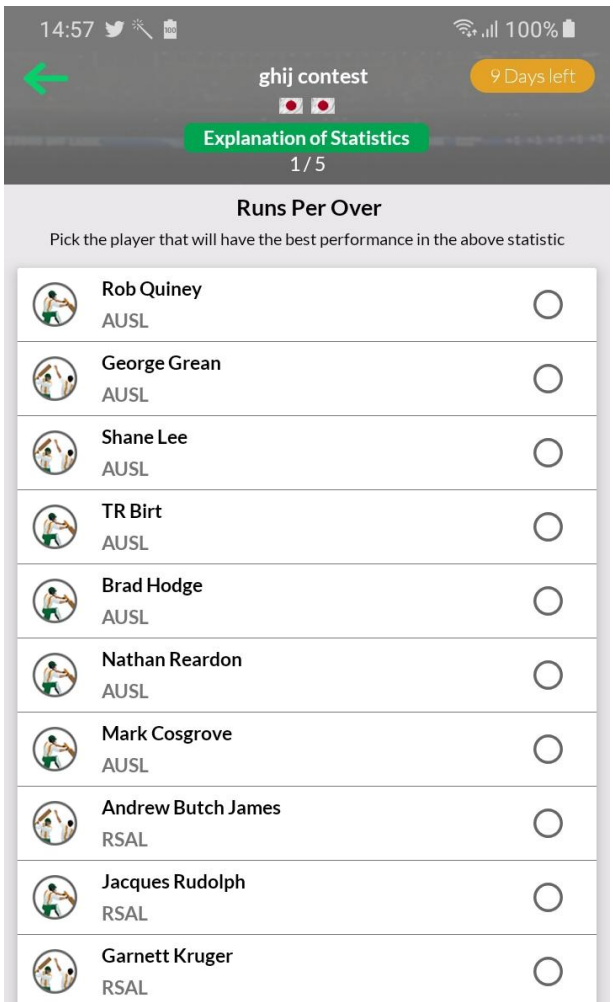
```
HashMap<String, String> params = new HashMap<>();
    params.put("contestId", mContestId);
    params.put("username", mUserName);
```

beginSort() is used to sort data accordingly like Name, Date, Prize, Entry Fee
fragment-->History
getHistoryData() is used to fetch history data from api. Api name is competition/history/.

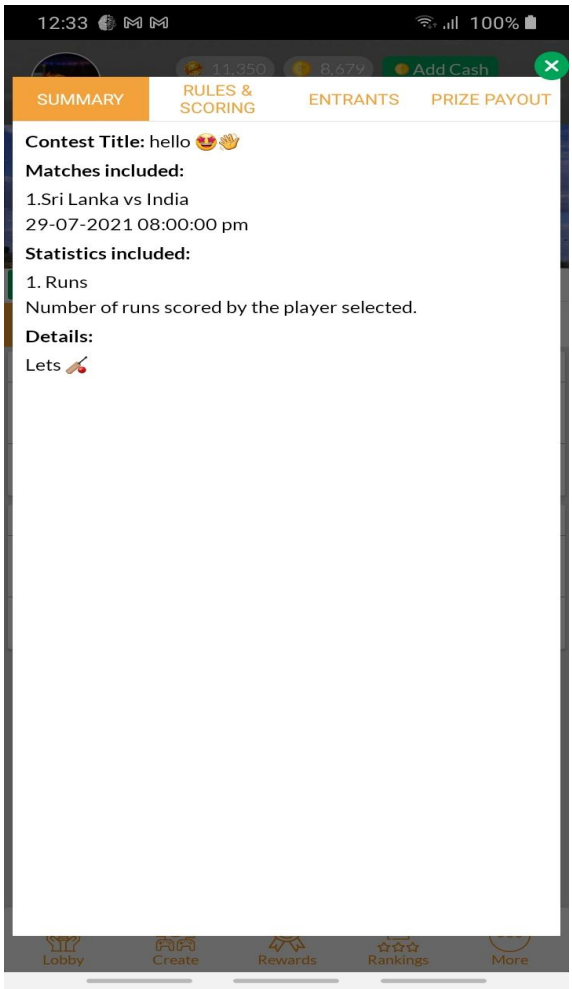
When the user clicks on the view button from the history contest another api will be called for details of the history match. The name is competition/standings?contestId=”contestID”
To view individual player list details need to call another api the name is competition/playerDetails/

```
HashMap<String, String> params = new HashMap<>();
        params.put("contestId", mContestId);
        params.put("username", mUserName);
```

beginSort() is used to sort data accordingly like Name, Date, Prize, Entry Fee
beginSearch() is used to Search data from history by contest name

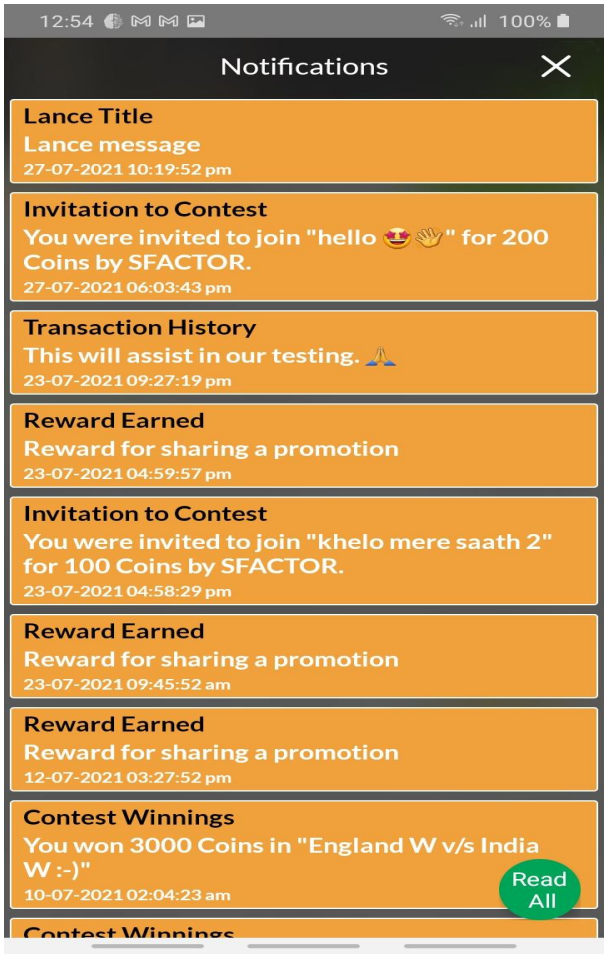


SELECT MVP SCREEN
Screen Name : SelectMVPActivity
Xml Page Name: activity_select_mvp
Here users can add players as per statistics. Users can choose only one player per statistic.
getLobbyMatchDetails() is used to fetch player lists per statistics . Api name is Competition/lobby
showExplanationStatisticsDialog() is used to show details of a statistic.
if user not join any contest then show a welcome popup and after disappearing the popup show the player list of this match. welcomeDialog() is the method name. It will dismiss automatically after 5 sec.



MATCH INFO SCREEN

Screen Name: InfoViewActivity
Xml Page Name: activity_info_view
Here user can view summary of a particular match, Rules and scoring of that match, how many user are participants in this contest via Entrants and prize structure of this match.
competition/summary?contest=”contestId” is the api url for fetch summary details.
https://fanex.in/rules is the url for displaying rules and scoring.
competition/participants?contest=”contestId” is the api url for fetch entrants data.
competition/prizeBreakdown?contest=”contestId” is the api url for prize payout data.



NOTIFICATION VIEW SCREEN

Screen Name: NotificationViewActivity

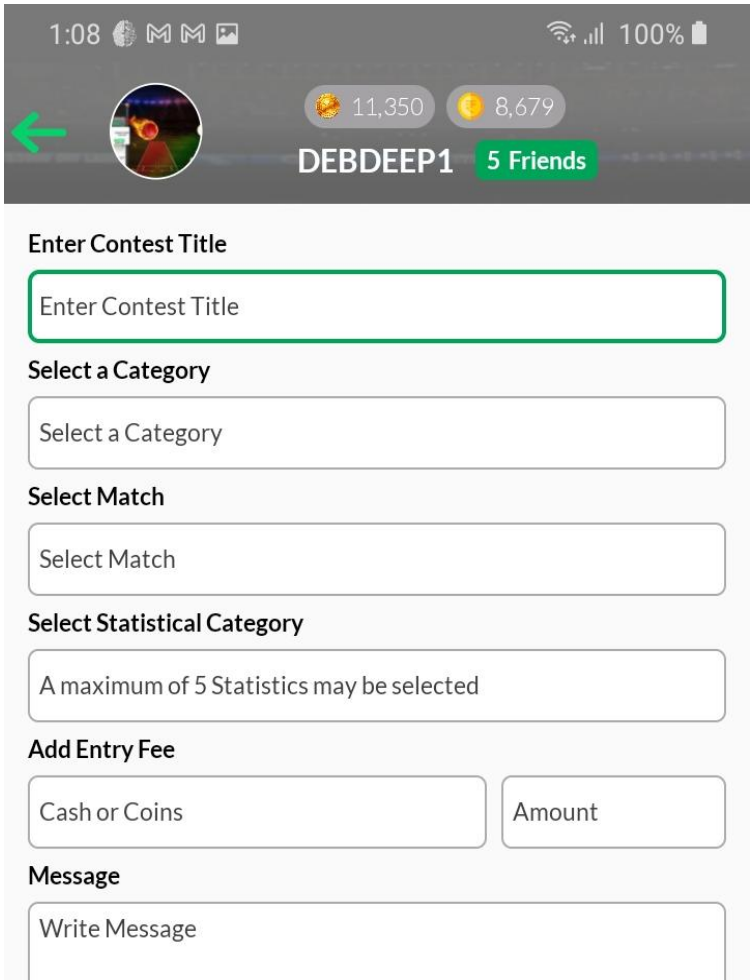
Xml Page Name: activity_notification_view

This is the app notification page where users can view different types of notifications. When a user tap any notification it will redirect to its corresponding page and the notification list color will change from gold to gray.

notification/list is the api name to fetch notification list.

To read all notifications at a time there is a read all button.

notification/clear is the api name to read all notifications.



CREATE CONTEST SCREEN

Screen Name: CreateMVPActivity

Xml page name: activity_create_mvp

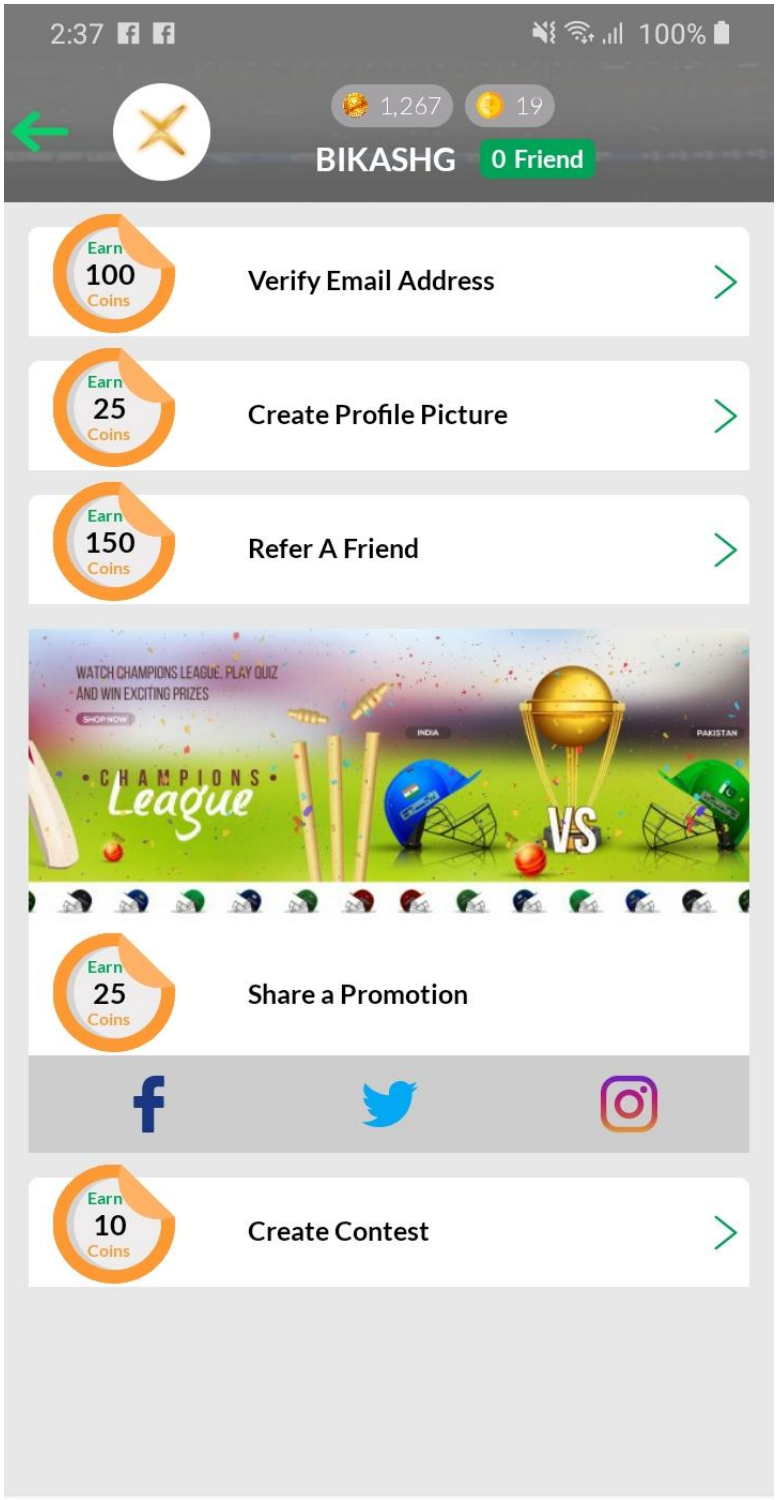
Here user can create different contests. Step by step you need to follow all instructions otherwise you can't create any contest.

Here differents api is calling for different field.

competition/createusercontest/ is the api name for submit contest data and param are

```
HashMap<String, String> params = new HashMap<>();
        params.put("category_id", tvselectmatchid.getText().toString());
        params.put("name", title);
        params.put("description", description);
```

```
params.put("entryFees", amount);
params.put("entryFeesType", entryfeestype);
params.put("prizeAmount", amount);
params.put("prizeType", entryfeestype);
params.put("statisticArr", arrstatistic.toString());
params.put("participants", arrselectedfriend.toString());
params.put("matchArr", jsonarray.toString());
```



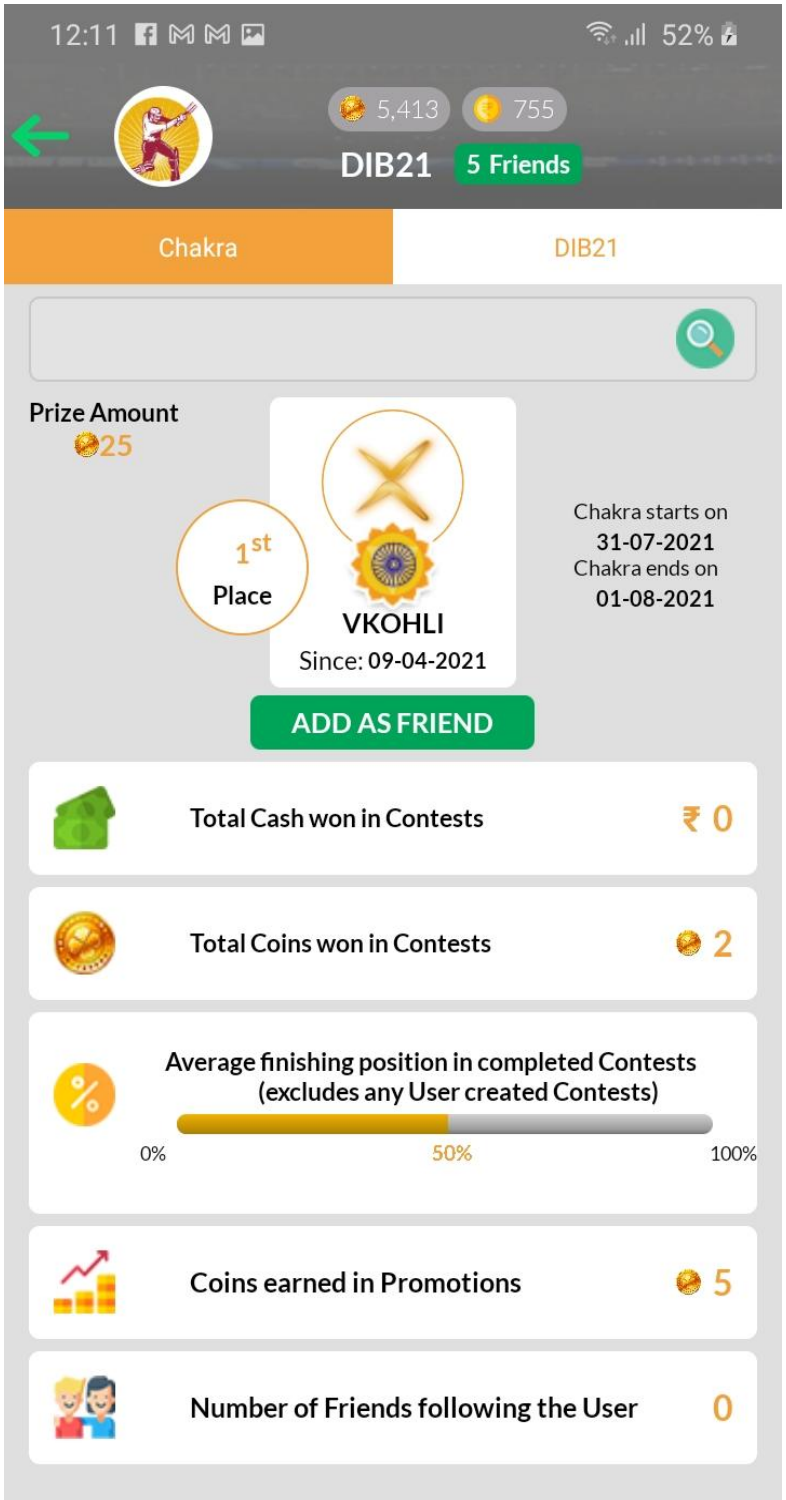
REWARD SCREEN

Screen Name: RewardActivity
Xml page name: activity_reward
In this page user can earn reward coin via performing this list of work. Two types of reward user will get.

1. One time reward:
   A.Verify email address.
   B.Create profile picture.
   C.Share a promotion via twitter, facebook,instagram.
   D. Add favorite player

2. Multiple time reward:
   A. Create Contest
   B. Refer A Friend

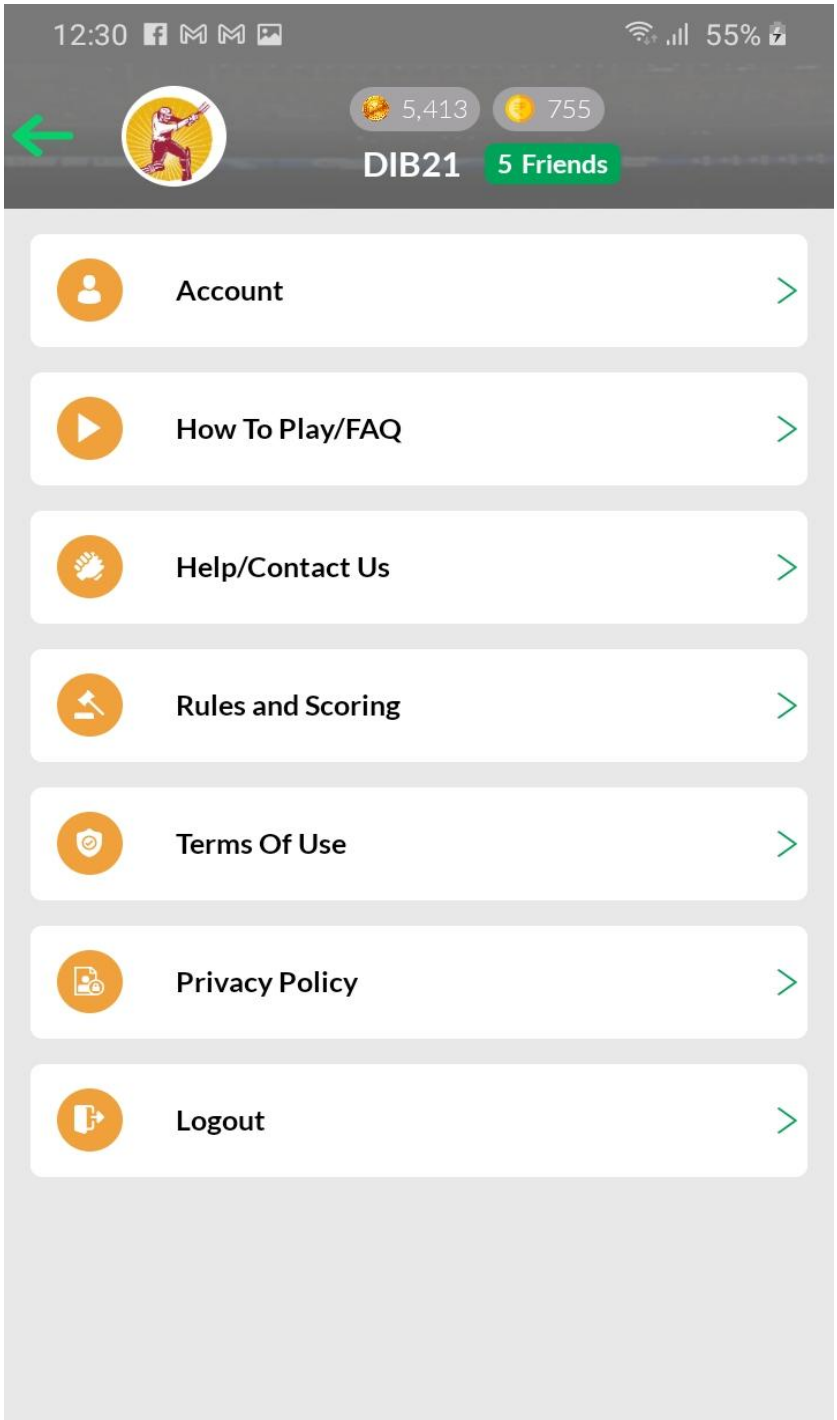
accounts/rewards/ is the api name to get a list of reward lists.



RANKING SCREEN

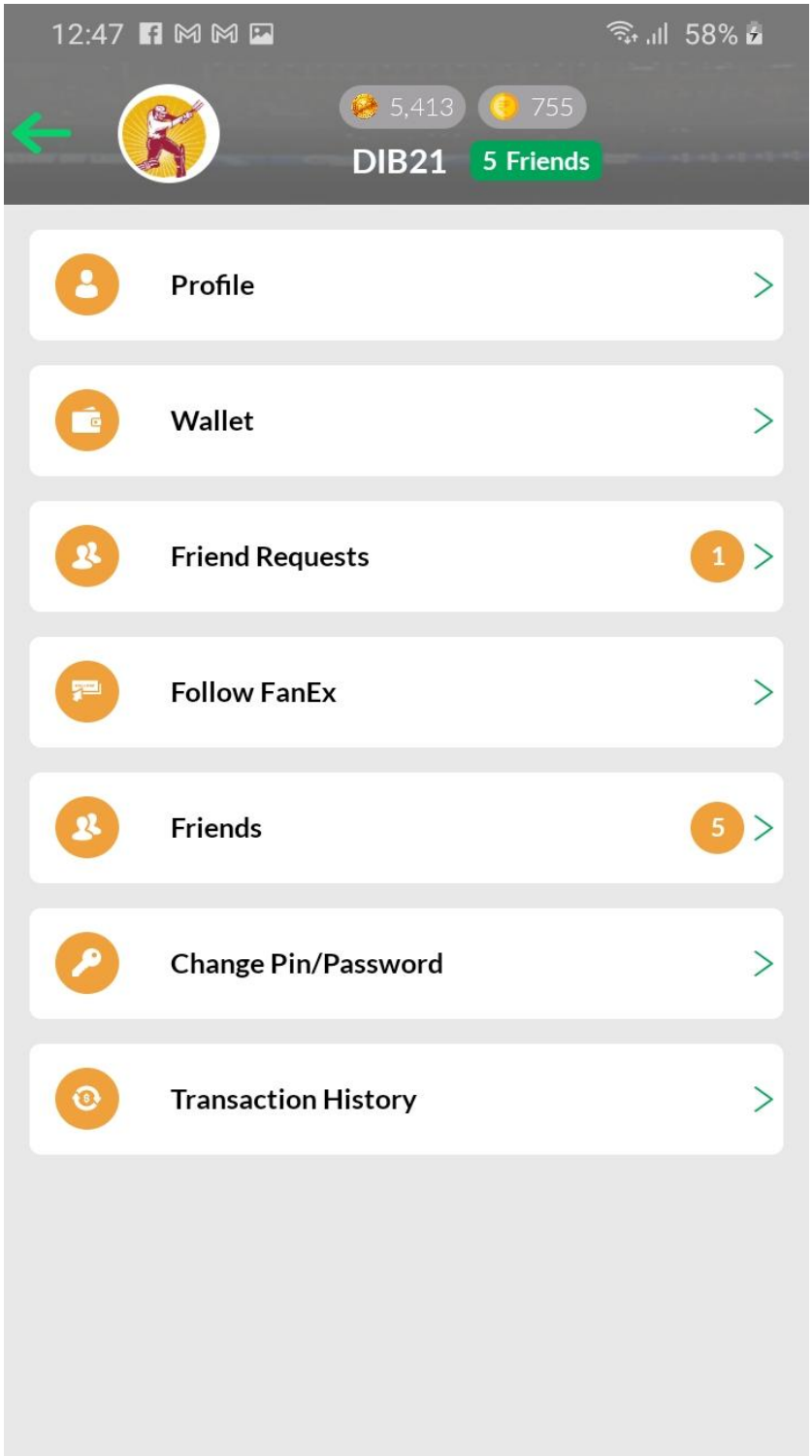
Screen Name: Ranking Activity
Xml Page Name: activity_ranking
In this page user can view his/her own rank as well as who is ranked one. Also user can search any user by his/her username. There is two tab, one for the chakra leaderboard and another for own leaderboard.

chakra/standings/?user= is the api for chakra leaderboard



MORE SCREEN

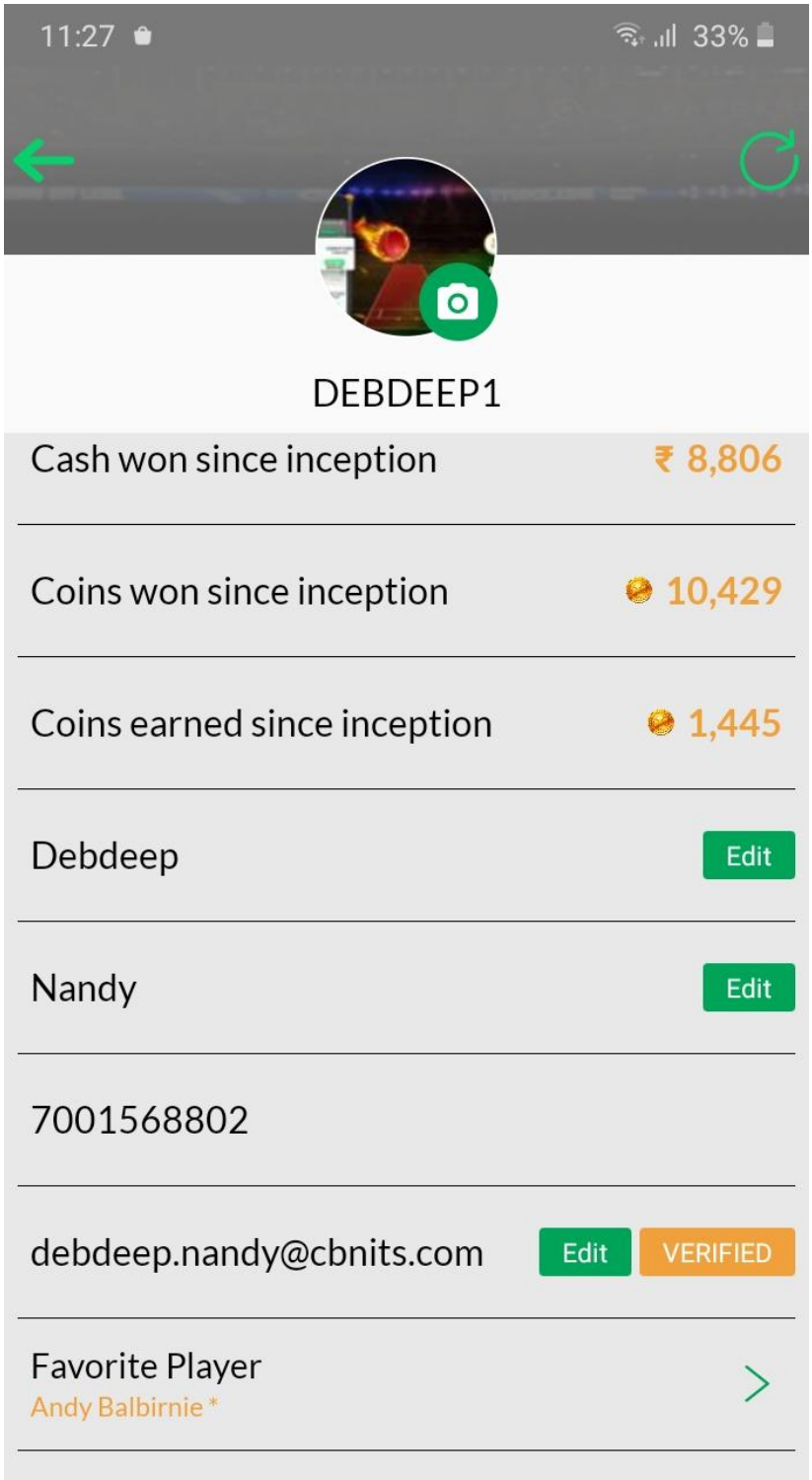
Screen Name: More Activity
Xml Page Name:activity_more
From this screen user can go Account Screen, How To Play/FAQ Screen, Help/Contact Us Screen, Rules and Scoring Screen, Terms Of Use Screen, Privacy Policy Screen, Logout Screen.



ACCOUNT SCREEN

Screen Name:AccountActivity
Xml Page Name:activity_account
From this account screen user can go Profile Screen, Wallet Screen, Friend Requests Screen, Follow FanEx Screen, Friends Screen, Change Pin/Password Screen, Transaction History Screen.

friends/count/ is the api for getting friend requests and friends count.



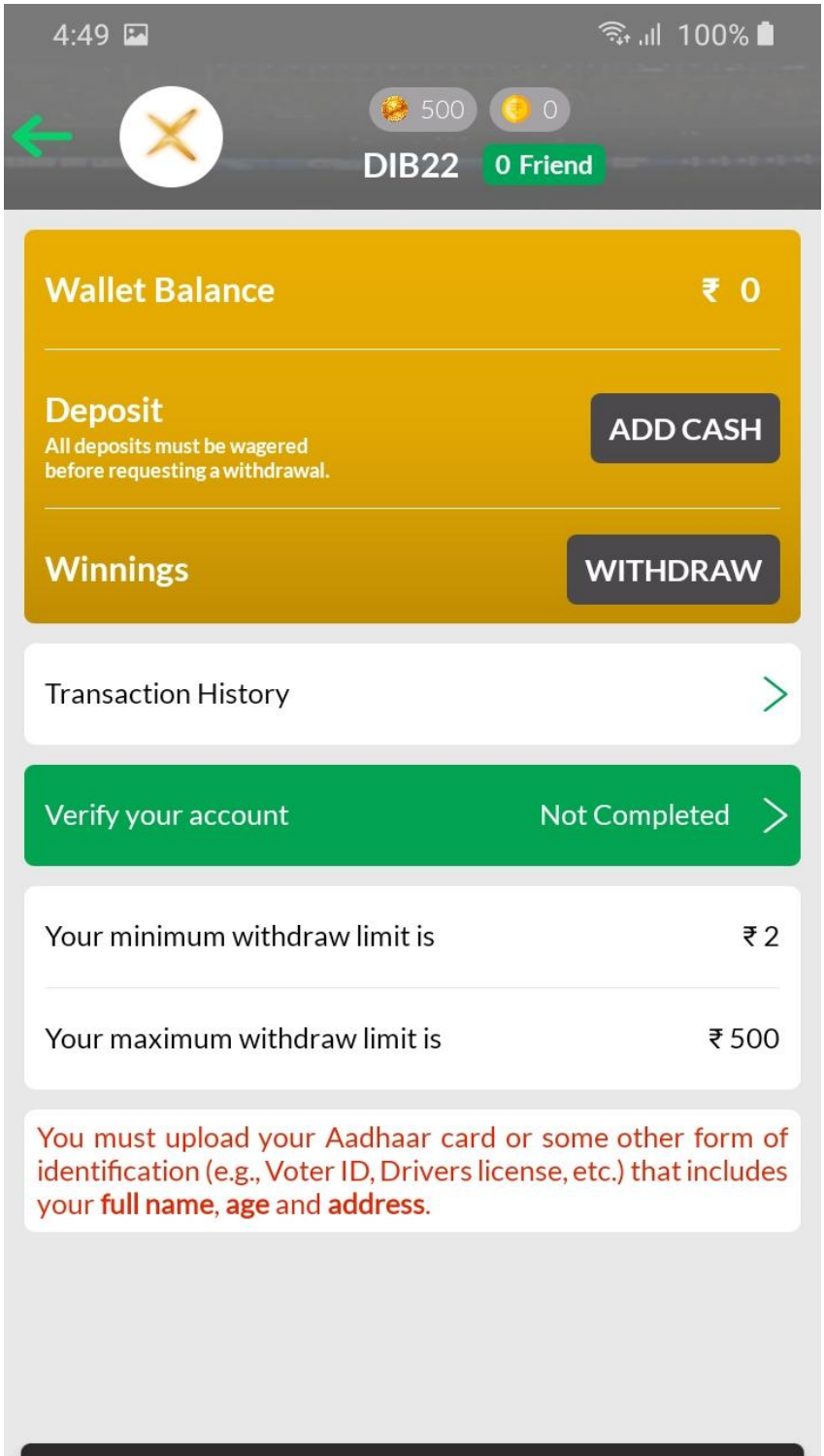
PROFILE VIEW / UPDATE SCREEN

In this screen user can view their own profile details and as well as they can update their profile.

Also user can turn on/off push notifications from that section.

accounts/ is the api name for view profile.

accounts/update/ is the api name for update profile.



WALLET SCREEN / ACCOUNT NOT VERIFIED

In this screen user can view his/her wallet balance. They can deposit cash via add cash option and can withdraw cash from his/her wallet balance. For that he/she needs to verify his/her account via **aadhar card** from the below option **Verify your account.** Otherwise he/she can't do anything. If the account is verified then the **ADD CASH AND WITHDRAW** option will turn into green and **Verify your account** will turn into gray.

To view wallet balance this api is called payment/walletAmount
To view maximum and minimum withdraw limit this api is called payment/withdrawLimit
Upload your document for kyc verification. We are using **onfido** SDK here. The steps is given below.

1. Generate onfido sdk token via

```
/**
        * Get SDK token for onfido params
        *       @param       "device_type":"android",
        *     @param "first_name":"Moi",
        *    @param  "last_name":"bhaumik"
        */
```

Api name is payment/getOnfidoToken

After getting a response from the backend we need to call startFlow(token) method for the uploading process. Here the token we will get from the previous api response.

When the user click submit button in startFlow process then we need call another api via this params

```
/**
params.put("applicant_id", applicant_id);
**/
```

Api name is payment/onfidoCheck

This api response will return checkId for that particular document and the we need to call another api with params.

```
/**
params.put("check_id", checkId);
**/
```

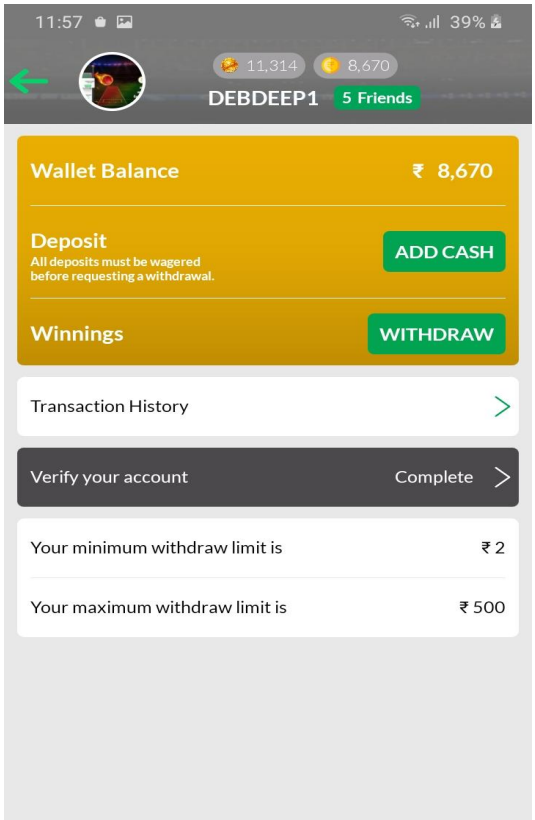
payment/onfidoReport

Using this api user can view their uploaded document status .

The status are (clear means completed, processing, awating, rejected)

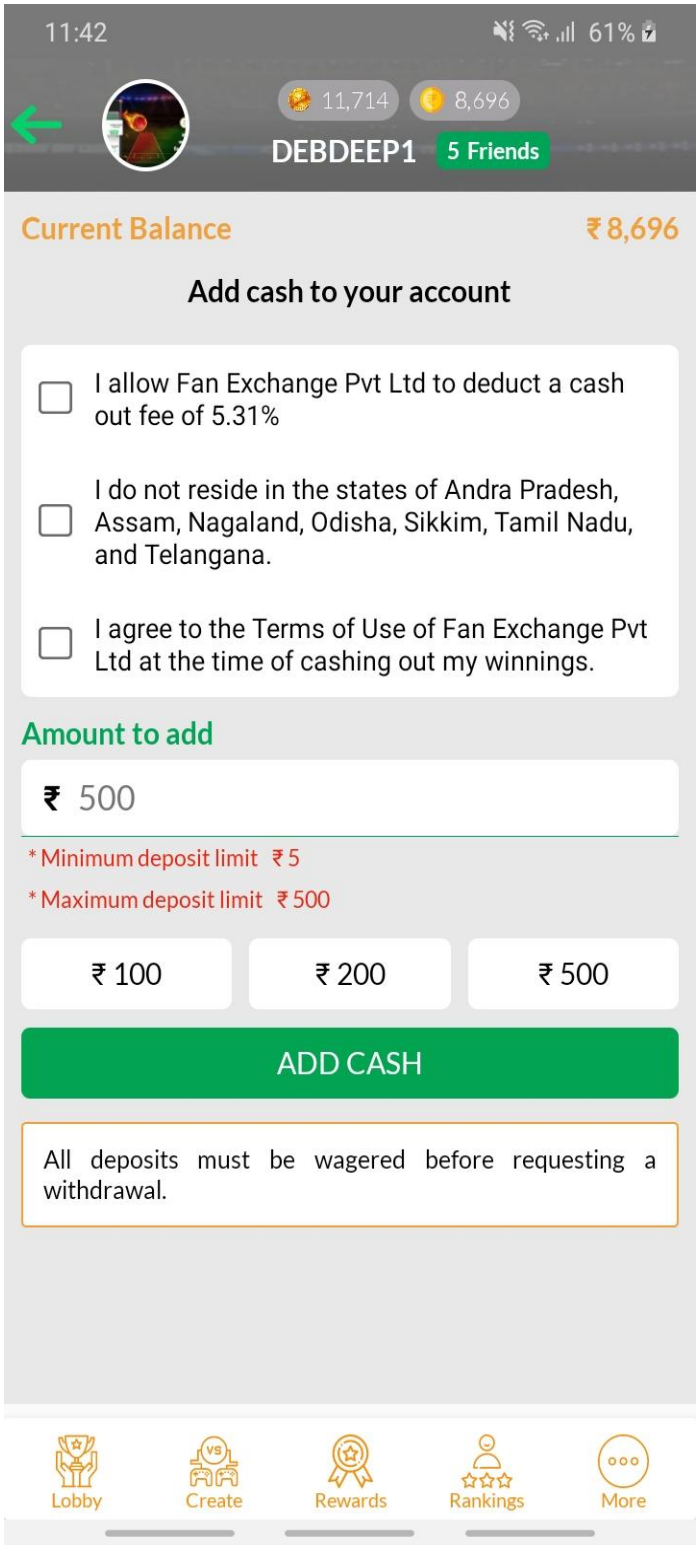
User can try to upload again if the status will be rejected or not completed.

If the document is verified the screen will show



ACCOUNT VERIFIED SCREEN

Document verified successfully



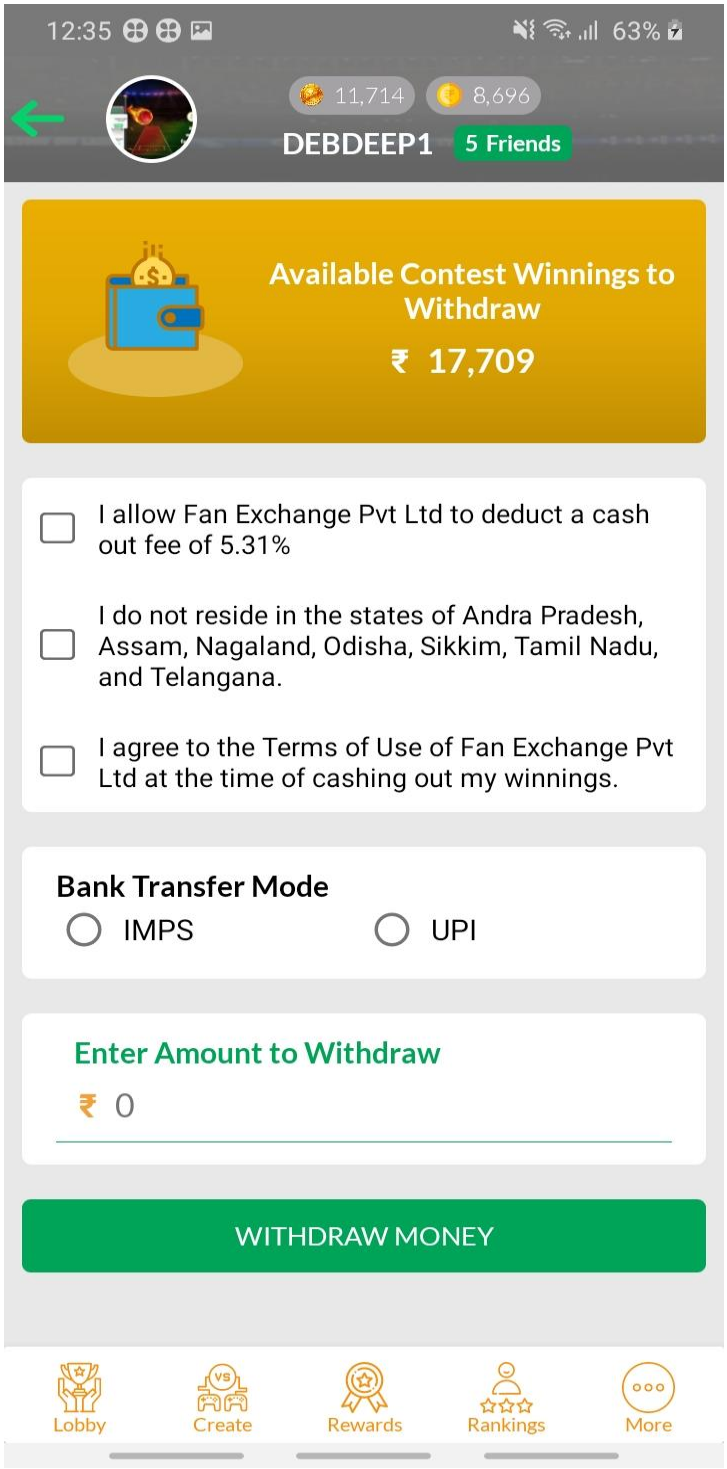
ADD CASH SCREEN

ADD CASH

In this screen user can add cash to his wallet. Before clicking the add cash button you need to check all checkbox first. After that you need to enter the amount that you want to add, amount should be in between min and max amount.
To display the current balance payment/walletAmount/ api will be called here.
To set max and min deposit amount payment/depositeLimit api will be called here.
When user click on Add cash button at first need to call initiate transaction api with params

```
HashMap<String, String> params = new HashMap<>();
  params.put("amount", mEtAddCash.getText().toString());
  params.put("currency", "INR");
```

Api name payment/initiateTransaction/
Here the payment process is done by Paytm payment SDK.



WITHDRAWAL SCREEN

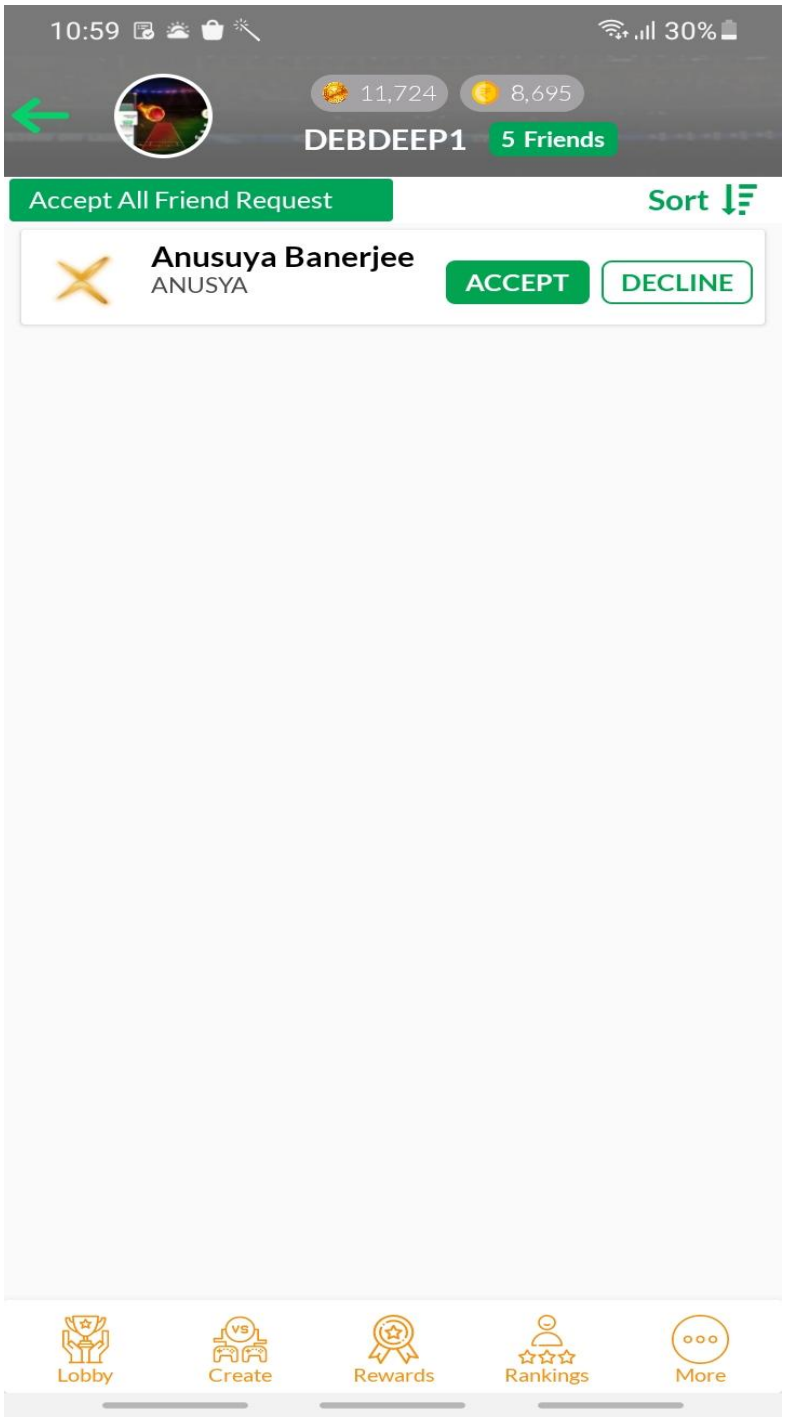
From this screen user can withdraw his/her winning cash amount .
Two way user can withdraw money

1. IMPS
2. UPI

If the user choose UPI type then he needs to input paytm phone number only.
If a user choose IMPS then he need to be submit bank account details like account no and IFS code.
As well as he/she needs to check all terms and conditions.
After that they input the withdrawal amount.
Api Params is given below

```
HashMap<String, String> params = new HashMap<>();
 params.put("amount",sring.valueOf(withdrawBalanceValue));
params.put("accountnumber", accountNo);
params.put("ifsccode", IFSCCode);
params.put("type_withdrawal", type_withdrawal);
params.put("upi_phone", upi_phone);
withdrawAmount(params);
```

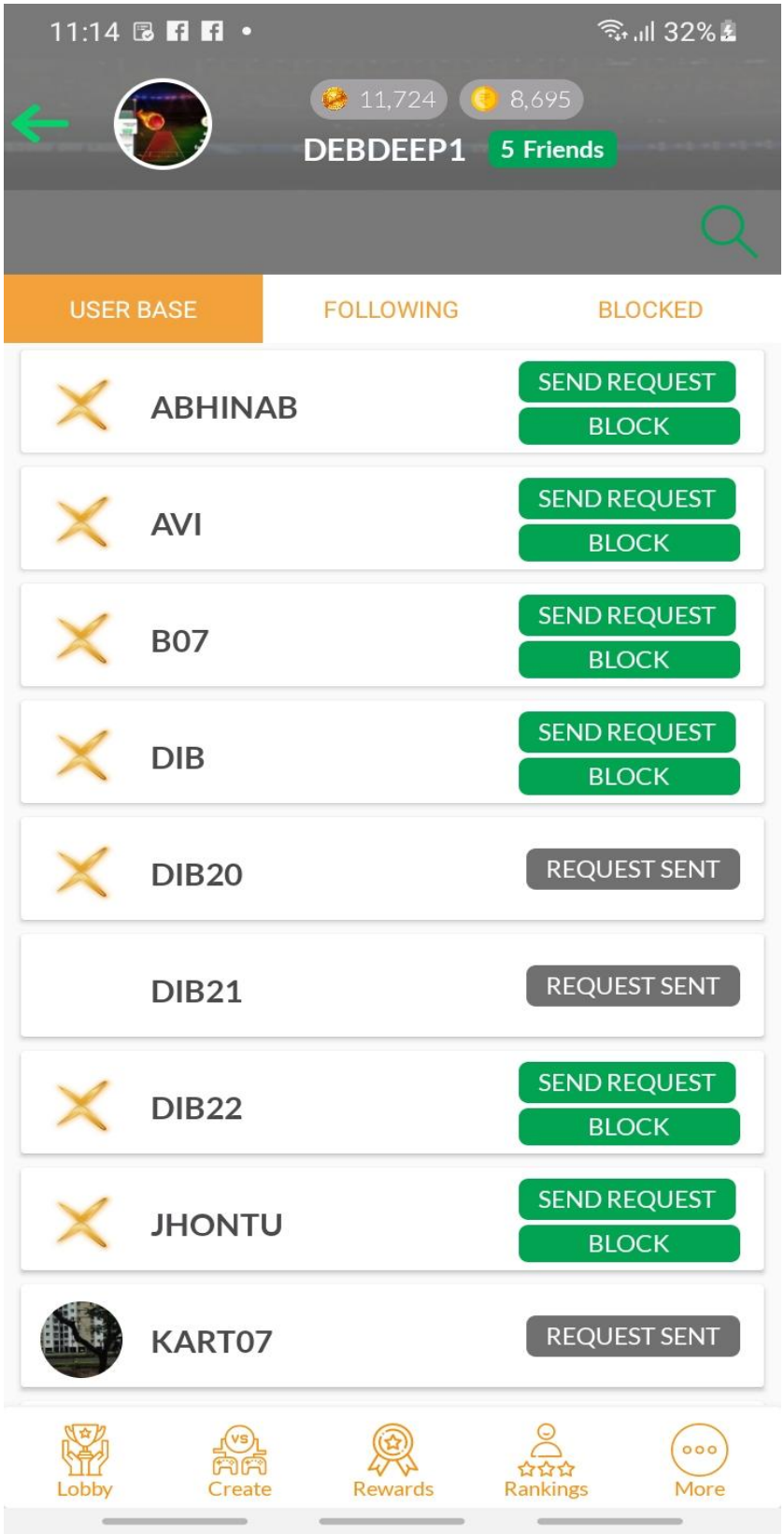
Api Name is payment/payuser/



ACCOUNT/FRIEND REQUEST SCREEN

In this screen the user can view list of friend request coming.
Also user can accept / decline the friend request.
If user wants to accept multiple friend request at a time there is a button “Accept All Friend Request”. Also user can sort the friend request list by name.
To display list of friend request need to call this api the name is given below
friends/invites/
When user click accept or decline button this api will be called
friends/action/
Params is

```
/**
    * Friend request accept or decline method
    * @Param type(1) means accept or
    * @Param type(2) means decline
    * @Param user id
    */
```

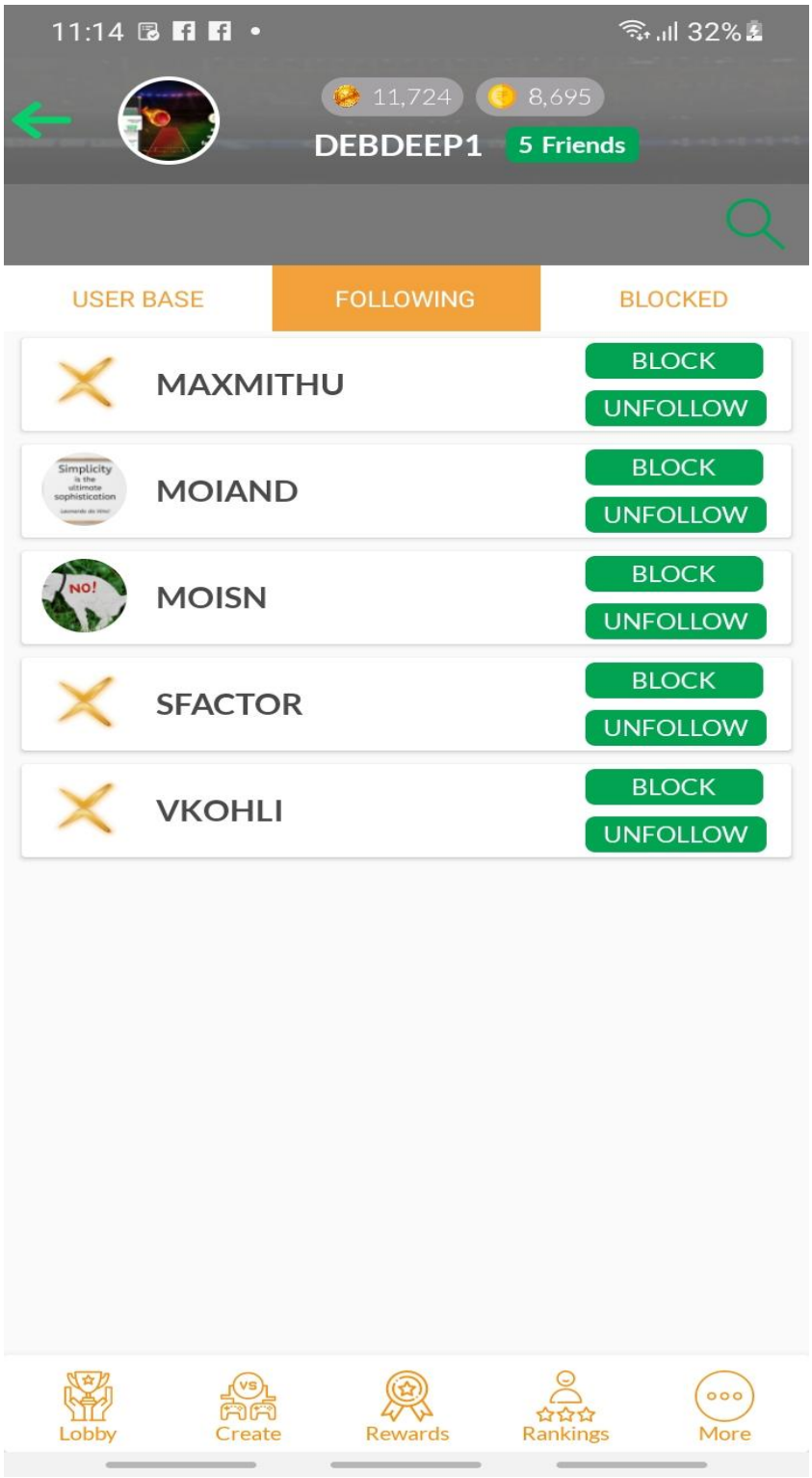


ACCOUNT/FRIEND(USER BASE)

In this screen list of users who have already registered in fanex bot not in logged in user friend. From this section a user can sent a friend request to a particular user and block it to a particular user.

Api name is
friends/users/
For action on send request or block
friends/action/

```
/**
    * Follow or Block Request Send
    * @Param type(4) //5 means Follow
    * @Param type(3) //3 means block
    * @Param user id
    */
```



ACCOUNT/FRIEND(FOLLOWING)

In this screen, a list of the following users is displayed here. From this section a user can block a particular user and also can unblock it to a particular user.

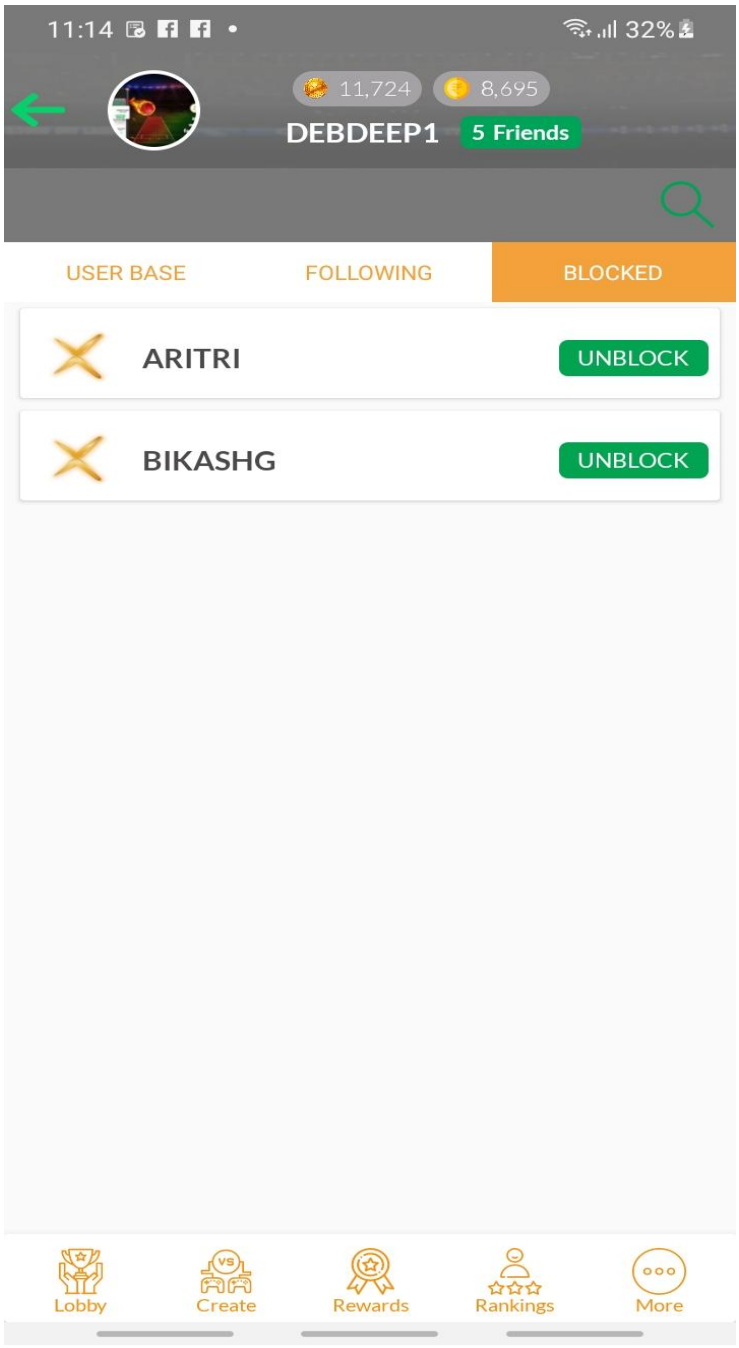
Api name is

friends/

For action on send request or block

friends/action/

```
/**
    * UnFollow or Block Request Send
    * @Param type(5) //5 means unFollow or UnFriend
    * @Param type(3) //3 means block
    * @Param user id
    */
```



ACCOUNT/FRIEND(BLOCKED)

In this screen, a list of the blocked users is displayed here. From this section a user can unblock a particular user.

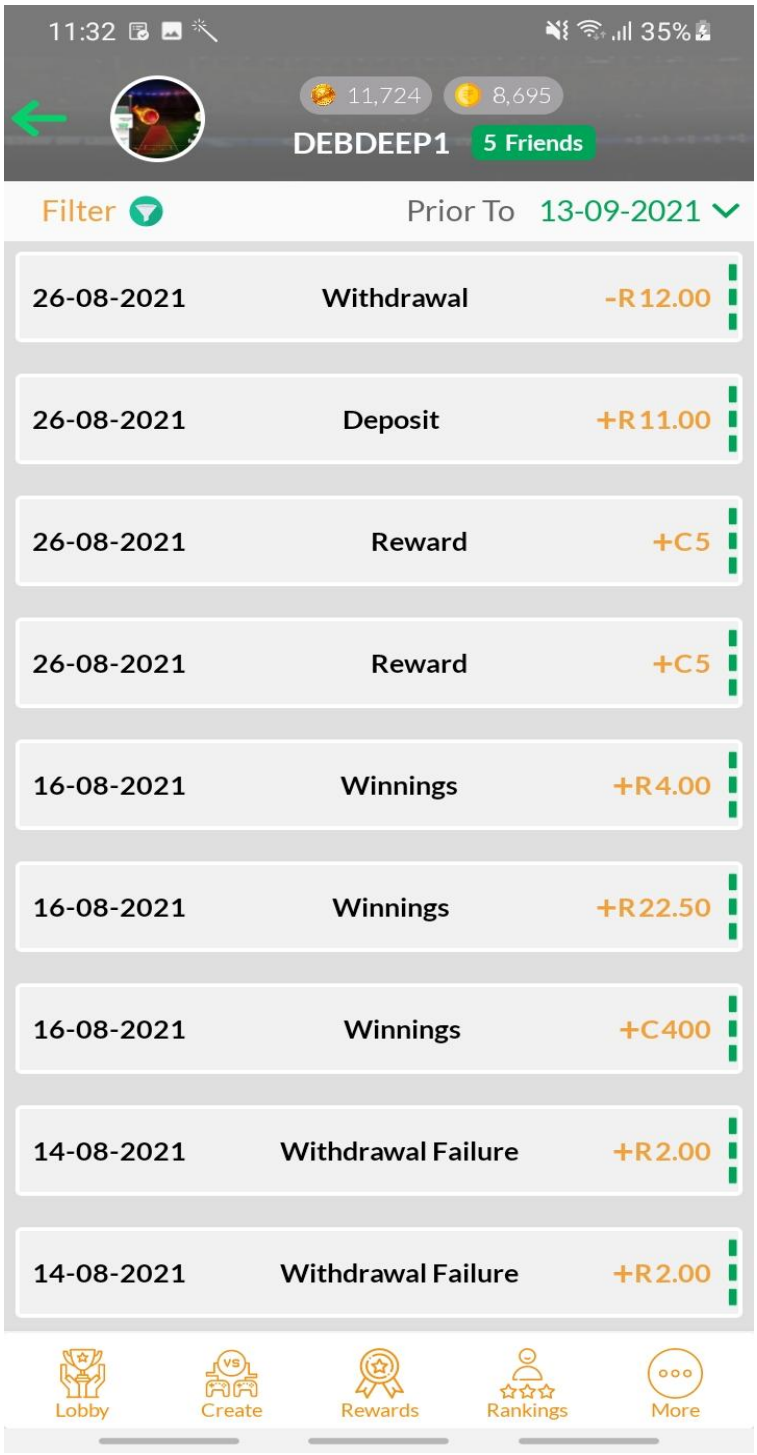
Api name is

friends/blocked

For action on send request or block

friends/action/

```
 /**
     * Block any user
     * @Param type(6) //6 means block
     * @Param user id
     */
```



ACCOUNT/TRANSACTION HISTORY SCREEN

In this screen, User can view his/her data for all transactions. And the name of different types of transactions are given below,

1. Winnings
2. Reward
3. Create Contest
4. Refer
5. Entry Fee
6. Withdrawal
7. Deposit
8. Withdrawal Failure

User can filter the transaction data by using the type of transaction name and date as well.

Api name is accounts/transactionHistory

```
HashMap<String, String> params = new HashMap<>();
        params.put("filters", “filter option data”);
        params.put("date", “selected date”);
```