Language	API Method & Post Function	Description
C#	CreateMember	
	FundTransfer	
	CheckFundTransfer	
	GetBetDetail	
	LogIn	
	POST function	Post API
Java	CreateMember	
	FundTransfer	
	CheckFundTransfer	
	GetBetDetail	
	LogIn	
	POST function	Post API
PHP	CreateMember	
	FundTransfer	
	CheckFundTransfer	
	GetBetDetail	
	LogIn	
	POST function	Post API

C#

CreateMember

```
public void doCreateMember()
{
        try
        {
            CreateMemberResult _CreateMemberResult = Newtonsoft.Json.JsonConvert.DeserializeObject<CreateMemberResult>(CreateMember());
            if (_CreateMemberResult.error_code > 0)
            {
                ResponseBody.Text = _CreateMemberResult.error_code.ToString();
                // check error message code
                /*
                Code Text       Description
                ----------------------------
                1       Failed  Failed during executed
                2       Failed  User Name Dupliate
                3       Failed  OperatorId is incorrect
                4       Failed  Odds Type format error
                5       Failed  Currency format error
                6       Failed  Vendor_Member ID Duplicate
                7       Failed  MinTransfer > MaxTransfer
                9       Failed  Invalidate vendor_id
                10      Failed  System is under maintenance
                */
            }
            else
            {
                ResponseBody.Text = "Successfully executed";
                // Code Text    Description
                //    ----------------------------
                //0     OK      Successfully executed
            }
        }
        catch (Exception se)
        {
            ResponseBody.Text = se.Message;
            string sErrorMessage = se.Message;
        }
  }

  class CreateMemberResult
  {
        public int error_code { get; set; }
        public string message { get; set; }
  }

  public string CreateMember()
  {
        string sFuntion = "CreateMember";
        string sVendor_Member_ID = "XXXX";
        string sFirstName = "XXXX";
        string sLastName = "XXX";
        string sOddsType = "X";
        string sCurrency = "20";
        string sOperatorId = "XXX"; // the default value usually is site name
        string sMaxTransfer = "XX";
        string sMinTransfer = "XX";
        return QueryAPI(sFuntion, new Dictionary<string, string>()
            {
                { "vendor_id", sAPI_VendorID },
                { "Vendor_Member_ID", sVendor_Member_ID},
                { "OperatorId", sOperatorId},
                { "FirstName", sFirstName},
                { "LastName",sLastName},
                { "UserName", sVendor_Member_ID},
                { "OddsType", sOddsType },
                { "Currency", sCurrency},
                { "MaxTransfer", sMaxTransfer},
                { "MinTransfer",sMinTransfer}
            }
            );
  }
```

FundTransfer

```
public void doFundTransfer ()
{
        try
        {
            FundTransferResult _FundTransferResult = FundTransferFun();
            /* Scenarios of invoking FundTransfer:    */
            //4. If any exception occurred during the process, please proceed “checkfundtransfer” mechanism. */
            if (_FundTransferResult.error_code > 0)
            {
                CheckFundTransferResult _CheckFundTransferResult;
                _CheckFundTransferResult = CheckFundTransferFun(_FundTransferResult.Data.trans_id.ToString());
                if (_CheckFundTransferResult.error_code > 0)
                {
                    ResponseBody.Text = _CheckFundTransferResult.error_code.ToString();
                    //1  Failed    Failed during executed
                    //2  Failed    Transaction record does not exist
                    //7  Failed    wallet_id input error
                    //9  Failed    Invalidate vendor_id
                    //10           Failed    System is under maintenance
                }
            }
            else
            {
                if (_FundTransferResult.Data.status == 0)
                {
                    ResponseBody.Text = "Successfully executed";
                    // 1. If status code is OK (0), transaction succeeds.    */
                }
                else if (_FundTransferResult.Data.status == 1)
                {
                    //     2. If status code is Failed (1), please check the error code. Fix the error and try again later.
                }
                else if (_FundTransferResult.Data.status == 2)
                {
                    //3. If status code is Pending (2) , please proceed “checkfundtransfer” mechanism.

                    CheckFundTransferResult _CheckFundTransferResult;
                    _CheckFundTransferResult = CheckFundTransferFun(_FundTransferResult.Data.trans_id.ToString());
                    if (_CheckFundTransferResult.error_code > 0)
                    {
                        //1       Failed    Failed during executed
                        //2       Failed    Transaction record does not exist
                        //7       Failed    wallet_id input error
                        //9       Failed    Invalidate vendor_id
                        //10      Failed    System is under maintenance
                    }
                }


            }
        }
        catch (Exception se)
        {
            ResponseBody.Text = se.Message;
            string sErrorMessage = se.Message;
        }
    }

    public class FundTransferData
    {
        public long trans_id { get; set; }
        public decimal before_amount { get; set; }
        public decimal after_amount { get; set; }
        public int status { get; set; }
    }

    public class FundTransferResult
    {
        public int error_code { get; set; }
        public string message { get; set; }
        public FundTransferData Data { get; set; }
    }
    public FundTransferResult FundTransferFun()
    {
        Random r = new Random();
        string vendor_trans_id = GetRandomString(r, 20);
        FundTransferResult _FundTransferResult = Newtonsoft.Json.JsonConvert.DeserializeObject<FundTransferResult>(FundTransfer(vendor_trans_id));
        return _FundTransferResult;
    }
    private string GetRandomString(Random rnd, int length)
    {
        string charPool = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890";
        StringBuilder rs = new StringBuilder();

        while (length > 0)
        {
            rs.Append(charPool[(int)(rnd.NextDouble() * charPool.Length)]);
            length--;
        }
        return rs.ToString();
    }

    public string FundTransfer(string vendor_trans_id)
    {

        string sFuntion = "FundTransfer";
        string sVendor_Member_ID = "XXXXXX";
        string amount = "10";
        string currency = "20";
        string direction = "1"; // 0 = Withdraw, 1= Deposit
        string wallet_id = "";//Wallet ID 1 : Sportsbook 5 : AG 6 : GD
        return QueryAPI(sFuntion, new Dictionary<string, string>()
            {
                { "vendor_id", sAPI_VendorID },
                { "vendor_member_id", sVendor_Member_ID},
                { "vendor_trans_id", vendor_trans_id},
                { "amount", amount},
                { "currency",currency},
                { "direction", direction},
                { "wallet_id", wallet_id },
            }
            );
    }
```

CheckFundTransfer

```
public void doCheckFundTransfer ()
{
        try
        {
            string trans_id = "XXXXXXXXXXX";
            CheckFundTransferResult _CheckFundTransferResult = CheckFundTransferFun(trans_id);
            if (_CheckFundTransferResult.error_code > 0)
            {
                ResponseBody.Text = _CheckFundTransferResult.error_code.ToString();
                //1     Failed  Failed during executed
                //2     Failed  Transaction record does not exist
                //7     Failed  wallet_id input error
                //9     Failed  Invalidate vendor_id
                //10    Failed  System is under maintenance
            }
            else
            {
                ResponseBody.Text = "OK";
            }
        }
        catch (Exception se)
        {
                string sErrorMessage = se.Message;
        }
    }
    public class CheckFundTransferData
    {
        public long trans_id { get; set; }
        public DateTime transfer_date { get; set; }
        public decimal amount { get; set; }
        public int currency { get; set; }
        public decimal before_amount { get; set; }
        public decimal after_amount { get; set; }
        public int status { get; set; }
    }

    public class CheckFundTransferResult
    {
        public int error_code { get; set; }
```

```
            public string message { get; set; }
            public CheckFundTransferData Data { get; set; }
        }

        public CheckFundTransferResult CheckFundTransferFun(string vendor_trans_id)
        {
            CheckFundTransferResult _CheckFundTransferResult =
Newtonsoft.Json.JsonConvert.DeserializeObject<CheckFundTransferResult>(CheckFundTransfer(vendor_trans_id));

            /*Scenarios of invoking CheckFundTransfer:
              * a. If status code is OK (0), the queried transaction succeeded.
              * b. If status code is Failed (1), the queried transaction failed for some reason.
              * c. If status code is Pending (2) , please continue with “checkfundtransfer”. I. Repeat every 5 min until the solid response (statuscode 0 or 1) is received.
              * d. If the status code is null, please check error code and fix it before query again.
              * e. If any exception occurred during the process, please continue with “checkfundtransfer” mechanism.*/
            if (_CheckFundTransferResult.error_code > 0)
            {
                //      * e. If any exception occurred during the process, please continue with “checkfundtransfer” mechanism.*/
            }
            else
            {
                if (_CheckFundTransferResult.Data.status == 0)
                {
                    //a. If status code is OK (0), the queried transaction succeeded.
                }
                else
                {
                    if (_CheckFundTransferResult.Data.status == 1)
                    {
                        // * b. If status code is Failed (1), the queried transaction failed for some reason.
                        string log = "b. If status code is Failed (1), the queried transaction failed for some reason. ";
                        // * d. If the status code is null, please check error code and fix it before query again.
                    }
                    else if (_CheckFundTransferResult.Data.status == 2)
                    {
                        // * c. If status code is Pending (2) , please continue with “checkfundtransfer”. I. Repeat every 5 min until the solid response (statuscode 0 or 1) is
received.
                        Thread.Sleep(60000); //1 min
                        attempts++;
                        if (attempts > 3)
                        {
                            //contact OneWorks
                            string log = "contact OneWorks ";
                        }
                        else
                        {
                            CheckFundTransferFun(vendor_trans_id);
                        }
                    }
                }
            }
            return _CheckFundTransferResult;
        }

        public string CheckFundTransfer(string vendor_trans_id)
        {
            string sFuntion = "CheckFundTransfer";
            string wallet_id = "";//Wallet ID 1 : Sportsbook 5 : AG 6 : GD
            return QueryAPI(sFuntion, new Dictionary<string, string>()
                {
                    { "vendor_id", sAPI_VendorID },
                    { "vendor_trans_id", vendor_trans_id},
                    { "wallet_id", wallet_id },
                }
                );
        }
```

GetBetDetail

```
    int lastVersionKey = XXXXXXXXx;
    public void doGetBetDetail ()
    {
            try
            {
                string sType = "Main";//NOTE: for Main sample only, modify result column if there is needs
                if (sType == "Main")
                {
                    BetDetailResult _BetDetailResult = Newtonsoft.Json.JsonConvert.DeserializeObject<BetDetailResult>(GetBetDetail(lastVersionKey));
```

```
                        if (_BetDetailResult.error_code > 0)
                        {
                                ResponseBody.Text = _BetDetailResult.error_code.ToString();
                                // Code         Text        Description
                                //---------------------------
                                //0    OK       Successfully executed
                                //1    Failed  Failed during executed
                                //9    Failed  Invalidate vendor_id
                                //10 Failed   System is under maintenance

                        }
                        else
                        {
                                ResponseBody.Text = "OK";
                                if (_BetDetailResult.Data.BetDetails.Count > 0)
                                {
                                        // renew version key
                                        lastVersionKey = _BetDetailResult.Data.last_version_key;

                                }

                        }
                }
        }
        catch (Exception se)
        {
                ResponseBody.Text = se.Message;
                string sErrorMessage = se.Message;
        }
}

public class BetDetail
{
        public long trans_id { get; set; }
        public string vendor_member_id { get; set; }
        public string operator_id { get; set; }
        public int league_id { get; set; }
        public int match_id { get; set; }
        public int home_id { get; set; }
        public int away_id { get; set; }
        public DateTime match_datetime { get; set; }
        public int sport_type { get; set; }
        public int bet_type { get; set; }
        public int parlay_ref_no { get; set; }
        public decimal odds { get; set; }
        public decimal stake { get; set; }
        public decimal validbetamount { get; set; }
        public DateTime transaction_time { get; set; }
        public string ticket_status { get; set; }
        public decimal winlost_amount { get; set; }
        public decimal after_amount { get; set; }
        public int currency { get; set; }
        public DateTime winlost_datetime { get; set; }
        public int odds_type { get; set; }
        public string isLucky { get; set; }
        public string bet_team { get; set; }
        public string exculding { get; set; }
        public decimal home_hdp { get; set; }
        public decimal away_hdp { get; set; }
        public object hdp { get; set; }
        public string betfrom { get; set; }
        public string islive { get; set; }
        public int? home_score { get; set; }
        public int? away_score { get; set; }
        public string customInfo1 { get; set; }
        public string customInfo2 { get; set; }
        public string customInfo3 { get; set; }
        public string customInfo4 { get; set; }
        public string customInfo5 { get; set; }
        public string ba_status { get; set; }
        public int version_key { get; set; }
}

public class BetDetailData
{
```

```
            public int last_version_key { get; set; }
            public List<BetDetail> BetDetails { get; set; }
        }

        public class BetDetailResult
        {
            public int error_code { get; set; }
            public string message { get; set; }
            public BetDetailData Data { get; set; }
        }

        public string GetBetDetail(int VersionKey)
        {
            string sFuntion = "GetBetDetail";
            string options = "";
            return QueryAPI(sFuntion, new Dictionary<string, string>()
                {
                    { "vendor_id", sAPI_VendorID },
                    { "version_key", VersionKey.ToString()},
                    { "options", options },
                }
                );
        }
```

LogIn

```
    public void doLogIn ()
    {
            try
            {
                LogInResult _LogInResult = Newtonsoft.Json.JsonConvert.DeserializeObject<LogInResult>(LogIn());
                if (_LogInResult.error_code > 0)
                {
                    ResponseBody.Text = _LogInResult.error_code.ToString();
                    //Code   Text      Description
                    //---------------------------
                    //0          OK        Successfully executed
                    //1          Failed   System Error
                    //2          Failed   member not found
                    //9          Failed   Invalidate vendor_id
                    //10 Failed           System is under maintenance

                }
                else { ResponseBody.Text = "OK"; }
            }
            catch (Exception se)
            {
                ResponseBody.Text = se.Message;
                string sErrorMessage = se.Message;
            }
        }
        public class LogInResult
        {
            public int error_code { get; set; }
            public string message { get; set; }
            public string Data { get; set; }
        }

      public string LogIn()
        {
            string sFuntion = "LogIn";
            string sVendor_Member_ID = "XXXXXXXXXXXXX";
            string domain = "";
            return QueryAPI(sFuntion, new Dictionary<string, string>()
                {
                    { "vendor_id", sAPI_VendorID },
                    { "domain", domain},
                    { "vendor_member_id", sVendor_Member_ID },
                }
                );

        }
```

POST function

```
        static string sAPIUrl = "http://XX.X.XXX.XX:XX/api/";
        static string sAPI_VendorID = "XXXXXX";

        string QueryAPI(string funtion, Dictionary<string, string> args)
        {
```

```
        var dataStr = BuildPostData(args);
        var data = Encoding.ASCII.GetBytes(dataStr);
        var request = WebRequest.Create(new Uri(sAPIUrl + funtion)) as HttpWebRequest;
        if (request == null)
            throw new Exception("Non HTTP WebRequest");
        request.Method = "POST";
        request.Timeout = 15000;
        request.ContentType = "application/x-www-form-urlencoded";
        request.ContentLength = data.Length;
        var reqStream = request.GetRequestStream();
        reqStream.Write(data, 0, data.Length);
        reqStream.Close();
        var response = request.GetResponse();
        var resStream = response.GetResponseStream();
        var resStreamReader = new StreamReader(resStream);
        var resString = resStreamReader.ReadToEnd();
        return resString;
    }

    static string BuildPostData(Dictionary<string, string> d)
    {
        string s = "";
        for (int i = 0; i < d.Count; i++)
        {
            var item = d.ElementAt(i);
            var key = item.Key;
            var val = item.Value;
            s += String.Format("{0}={1}", key, val);
            if (i != d.Count - 1)
                s += "&";
        }
        return s;
    }
```

Java

CreateMember

```
public static void DoCreateMember()
 {
        System.out.println("==== Do CreateMember ====");
        Gson gson = new Gson();
        try
        {
                CreateMemberResult _CreateMemberResult = gson.fromJson(CreateMember(), CreateMemberResult.class);
                System.out.println("_CreateMemberResult error_code: " + _CreateMemberResult.error_code);
                System.out.println("_CreateMemberResult message: " + _CreateMemberResult.message);
                LogInResult _LogInResult = gson.fromJson(LogIn(), LogInResult.class);
                if (_LogInResult.error_code > 0)
                {
                        System.out.println("LogIn error_code: " + _LogInResult.error_code);
                        System.out.println("LogIn message: " + _LogInResult.message);
                        // check error message code
                        /*
                         * Code Text Description ---------------------------- 1 Failed Failed during
                         * executed 2 Failed User Name Dupliate 3 Failed OperatorId is incorrect 4
                         * Failed Odds Type format error 5 Failed Currency format error 6 Failed
                         * Vendor_Member ID Duplicate 7 Failed MinTransfer > MaxTransfer 9 Failed
                         * Invalidate vendor_id 10 Failed System is under maintenance
                         */
                } else
                {
                        // Code Text Description
                        // ----------------------------
                        // 0 OK Successfully executed
                }
        } catch (Exception e)
        {
                e.printStackTrace();
        }
 }

 private class CreateMemberResult
 {
        private int error_code;
        private String message;
 }

private static String CreateMember()
 {
        System.out.println("==== CreateMember ====");
        String sFuntion = "CreateMember";
        String sVendor_Member_ID = "xxxxx";
        String sFirstName = "xxxxx";
        String sLastName = "xxxxxx";
        String sOddsType = "a";
        String sCurrency = "20";
        String sOperatorId = "xxxxx"; // the default value usually is site name
        String sMaxTransfer = "100";
        String sMinTransfer = "1000";
        Map<String, String> params = new LinkedHashMap<String, String>();
        params.put("vendor_id", _APIVendorID);
        params.put("Vendor_Member_ID", sVendor_Member_ID);
        params.put("OperatorId", sOperatorId);
        params.put("FirstName", sFirstName);
        params.put("LastName", sLastName);
        params.put("UserName", sVendor_Member_ID);
        params.put("OddsType", sOddsType);
        params.put("Currency", sCurrency);
        params.put("MaxTransfer", sMaxTransfer);
        params.put("MinTransfer", sMinTransfer);
        return QueryAPI(sFuntion, params);
 }
```

FundTransfer

```
public static void DoFundTransfer()
 {
        System.out.println("==== Do FundTransfer ====");
        try
        {
                FundTransferResult _FundTransferResult = FundTransferFun();
                /* Scenarios of invoking FundTransfer: */
```

```
                    System.out.println("_FundTransferResult.error_code:" + _FundTransferResult.error_code);
                    System.out.println("_FundTransferResult.message:" + _FundTransferResult.message);

        /* Scenarios of invoking FundTransfer:    */
        //4. If any exception occurred during the process, please proceed “checkfundtransfer” mechanism. */
        if (_FundTransferResult.error_code > 0)
        {
            CheckFundTransferResult _CheckFundTransferResult;
            _CheckFundTransferResult = CheckFundTransferFun(""+_FundTransferResult.Data.trans_id);
            if (_CheckFundTransferResult.error_code > 0)
            {
                //1        Failed    Failed during executed
                //2        Failed    Transaction record does not exist
                //7        Failed    wallet_id input error
                //9        Failed    Invalidate vendor_id
                //10       Failed    System is under maintenance
            }
        }
        else
        {
            if (_FundTransferResult.Data.status == 0)
            {
                // 1. If status code is OK (0), transaction succeeds.    */
            }
            else if (_FundTransferResult.Data.status == 1)
            {
                //     2. If status code is Failed (1), please check the error code. Fix the error and try again later.
            }
            else if (_FundTransferResult.Data.status == 2)
            {
                //3. If status code is Pending (2) , please proceed “checkfundtransfer” mechanism.

                       CheckFundTransferResult _CheckFundTransferResult;
                       _CheckFundTransferResult = CheckFundTransferFun("" + _FundTransferResult.Data.trans_id);
                if (_CheckFundTransferResult.error_code > 0)
                {
                    //1  Failed    Failed during executed
                    //2  Failed    Transaction record does not exist
                    //7  Failed    wallet_id input error
                    //9  Failed    Invalidate vendor_id
                    //10           Failed    System is under maintenance
                }
            }
        }
          } catch (Exception e)
          {
                    e.printStackTrace();
          }
}

private class FundTransferResult
{
          private int error_code;
          private String message;
          private FundTransferData Data;
}
private class FundTransferData
{
          private long trans_id;
          private BigDecimal before_amount;
          private BigDecimal after_amount;
          private int status;
}

private static FundTransferResult FundTransferFun()
{
          Gson gson = new Gson();
          Random r = new Random();
          String vendor_trans_id = GetRandomString(r, 20);
          System.out.println("-- vendor_trans_id    " + vendor_trans_id);

          FundTransferResult _FundTransferResult = gson.fromJson(FundTransfer(vendor_trans_id),
                              FundTransferResult.class);
          return _FundTransferResult;
}
```

```
 private static String GetRandomString(Random rnd, int length)
 {
        String sPool = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890";
        char[] charPool = sPool.toCharArray();
        StringBuilder rs = new StringBuilder();

        while (length > 0)
        {
               int i = (int) (rnd.nextDouble() * charPool.length);
               rs.append(charPool[i]);
               length--;
        }
        return rs.toString();
 }

 private static String FundTransfer(String vendor_trans_id)
 {
        System.out.println("==== FundTransfer ====");
        String sFuntion = "FundTransfer";
        String sVendor_Member_ID = "xxxxxxxxx";
         String amount = "10";
         String currency = "20";
        String direction = "1"; // 0 = Withdraw, 1= Deposit
        String wallet_id = "";//Wallet ID 1 : Sportsbook 5 : AG 6 : GD
        Map<String, String> params = new LinkedHashMap<String, String>();
        params.put("vendor_id", _APIVendorID);
        params.put("vendor_member_id", sVendor_Member_ID);
        params.put("vendor_trans_id", vendor_trans_id);
        params.put("amount", amount);
        params.put("currency",currency);
        params.put("direction", direction);
        params.put("wallet_id", wallet_id);
        return QueryAPI(sFuntion, params);
 }
```

CheckFundTransfer

```
public static void DoCheckFundTransfer()
 {
        System.out.println("==== Do CheckFundTransfer ====");

        String trans_id = "XXXXXXXXXXXXXXXXX";
        try
        {
               CheckFundTransferResult _CheckFundTransferResult = CheckFundTransferFun(trans_id);
               if (_CheckFundTransferResult.error_code > 0)
               {
                      // 1 Failed Failed during executed
                      // 2 Failed Transaction record does not exist
                      // 7 Failed wallet_id input error
                      // 9 Failed Invalidate vendor_id
                      // 10 Failed System is under maintenance
               }
        } catch (Exception e)
        {
               e.printStackTrace();
        }
 }
public static CheckFundTransferResult CheckFundTransferFun(String vendor_trans_id)
 {
        Gson gson = new Gson();
        CheckFundTransferResult _CheckFundTransferResult = gson.fromJson(
                      CheckFundTransfer(vendor_trans_id),        CheckFundTransferResult.class);
        try
        {
               /*
                 * Scenarios of invoking CheckFundTransfer: a. If status code is OK (0), the
                 * queried transaction succeeded. b. If status code is Failed (1), the queried
                 * transaction failed for some reason. c. If status code is Pending (2) , please
                 * continue with “checkfundtransfer”. I. Repeat every 5 min until the solid
                 * response (statuscode 0 or 1) is received. d. If the status code is null,
                 * please check error code and fix it before query again. e. If any exception
                 * occurred during the process, please continue with “checkfundtransfer”
                 * mechanism.
                 */
            if (_CheckFundTransferResult.error_code > 0)
            {
                //        * e. If any exception occurred during the process, please continue with “checkfundtransfer” mechanism.*/
```

```
                    }
                    else
                    {
                            if (_CheckFundTransferResult.Data.status == 0)
                            {
                                    // a. If status code is OK (0), the queried transaction succeeded.
                            } else
                            {
                                    if (_CheckFundTransferResult.Data.status == 1)
                                    {
                                            // * b. If status code is Failed (1), the queried transaction failed for some
                                            // reason.
                                            String log = "b. If status code is Failed (1), the queried transaction failed for some reason. ";
                                            // * d. If the status code is null, please check error code and fix it before
                                            // query again.
                                    } else if (_CheckFundTransferResult.Data.status == 2)
                                    {
                                            // * c. If status code is Pending (2) , please continue with
                                            // “checkfundtransfer”. I. Repeat every 5 min until the solid response
                                            // (statuscode 0 or 1) is received.
                                            Thread.sleep(60000); // 1 min
                                            attempts++;
                                            if (attempts > 3)
                                            {
                                                    // contact OneWorks
                                                    String log = "contact OneWorks ";
                                            } else
                                            {
                                                    CheckFundTransferFun(vendor_trans_id);
                                            }
                                    }
                            }
                    }
            } catch (Exception e)
            {
                    e.printStackTrace();
            }
            // * e. If any exception occurred during the process, please continue with
            // “checkfundtransfer” mechanism.*/
            return _CheckFundTransferResult;
    }

    private class CheckFundTransferResult
    {
            private int error_code;
            private String message;
            private CheckFundTransferData Data;
    }

    private class CheckFundTransferData
    {
            public long trans_id;
            //public Date transfer_date ;
            public String transfer_date ;
            public String vender_member_id;
            public BigDecimal amount;
            public int currency;
            public BigDecimal before_amount;
            public BigDecimal after_amount;
            public int status;
    }

    private static String CheckFundTransfer(String vendor_trans_id)
    {
            System.out.println("==== CheckFundTransfer ====");
            String sFuntion = "CheckFundTransfer";
            String wallet_id = "";// Wallet ID 1 : Sportsbook 5 : AG 6 : GD
            Map<String, String> params = new LinkedHashMap<String, String>();
            params.put("vendor_id", _APIVendorID);
            params.put("vendor_trans_id", vendor_trans_id);
            params.put("wallet_id", wallet_id);
            return QueryAPI(sFuntion, params);
    }
```

GetBetDetail

```
static int lastVersionKey = XXXXx;
public static void DoGetBetDetail()
```

```
 {
         System.out.println("==== Do GetBetDetail ====");
         Gson gson = new Gson();
         try
         {
                 String sType = "Main";// NOTE: for Main sample only, modify result column if there is needs
                 if (sType == "Main")
                 {
                         BetDetailResult _BetDetailResult = gson.fromJson(GetBetDetail(lastVersionKey), BetDetailResult.class);
                         if (_BetDetailResult.error_code > 0)
                         {
                                 // Code Text Description
                                 // ---------------------------
                                 // 0 OK Successfully executed
                                 // 1 Failed Failed during executed
                                 // 9 Failed Invalidate vendor_id
                                 // 10 Failed System is under maintenance
                         } else
                         {
                                 if (_BetDetailResult.Data.BetDetails.size() > 0)
                                 {
                                         // renew version key
                                         lastVersionKey = _BetDetailResult.Data.last_version_key;
                                 }
                         }
                 }
         } catch (Exception e)
         {
                 e.printStackTrace();
         }
 }

private class BetDetailResult
 {
         private int error_code;
         private String message;
         private BetDetailData Data;
 }

 private class BetDetailData
 {
         private int last_version_key;
         private List<BetDetail> BetDetails;
 }

 private class BetDetail
 {
         public long trans_id;
         public String vendor_member_id;
         public String operator_id;
         public int league_id;
         public int match_id;
         public int home_id;
         public int away_id;
         public Date match_datetime;
         public int sport_type;
         public int bet_type;
         public int parlay_ref_no;
         public BigDecimal odds;
         public BigDecimal stake;
         public BigDecimal validbetamount;
         public Date transaction_time;
         public String ticket_status;
         public BigDecimal winlost_amount;
         public BigDecimal after_amount;
         public int currency;
         public Date winlost_datetime;
         public int odds_type;
         public String isLucky;
         public String bet_team;
         public String exculding;
         public BigDecimal home_hdp;
         public BigDecimal away_hdp;
         public Object hdp;
         public String betfrom;
         public String islive;
```

```
		public int home_score;
		public int away_score;
		public String customInfo1;
		public String customInfo2;
		public String customInfo3;
		public String customInfo4;
		public String customInfo5;
		public String ba_status;
		public int version_key;
	}

	private static String GetBetDetail(int VersionKey)
	{
		System.out.println("==== GetBetDetail ====");
		String sFuntion = "GetBetDetail";
		String options = "";

		Map<String, String> params = new LinkedHashMap<String, String>();

		params.put("vendor_id", _APIVendorID);
		params.put("version_key", Integer.toString(VersionKey));
		params.put("options", options);

		return QueryAPI(sFuntion, params);
	}
```

LogIn

```
	public static void DoLogIn()
	{
		System.out.println("==== Do LogIn ====");
		Gson gson = new Gson();

		try
		{
			LogInResult _LogInResult = gson.fromJson(LogIn(), LogInResult.class);
			if (_LogInResult.error_code > 0)
			{
				System.out.println("LogIn error_code: " + _LogInResult.error_code);
				System.out.println("LogIn message: " + _LogInResult.message);
				// Code Text Description
				// ---------------------------
				// 0 OK Successfully executed
				// 1 Failed System Error
				// 2 Failed member not found
				// 9 Failed Invalidate vendor_id
				// 10 Failed System is under maintenance
			}
		} catch (Exception e)
		{
			e.printStackTrace();
		}
	}

	private class LogInResult
	{
		private int error_code;
		private String message;
		private String Data;
	}

	private static String LogIn()
	{
		System.out.println("==== LogIn ====");
		String sFuntion = "LogIn";
		String vendorMemberID = "XXX";
		String domain = "";
		Map<String, String> params = new LinkedHashMap<String, String>();
		params.put("vendor_id", _APIVendorID);
		params.put("domain", domain);
		params.put("vendor_member_id", vendorMemberID);
		return QueryAPI(sFuntion, params);
	}
```

POST function

```
	private static String _APIUrl = "http://XX.X.XXX.XX:XX/api/";
	private static String _APIVendorID = "XXXXX";
	private static byte[] buildPostData(Map<String, String> params)
	{
		StringBuilder postData = new StringBuilder();
		byte[] postDataBytes = new byte[]
		{};

		try
		{
			for (Map.Entry<String, String> param : params.entrySet())
			{
				if (postData.length() != 0)
				{
					postData.append('&');
				}
				postData.append(URLEncoder.encode(param.getKey(), "UTF-8"));
				postData.append('=');
				postData.append(URLEncoder.encode(String.valueOf(param.getValue()), "UTF-8"));
			}
			System.out.println("buildPostData -- " + postData);
			postDataBytes = postData.toString().getBytes("UTF-8");
		} catch (Exception e)
		{
			e.printStackTrace();
		}
		return postDataBytes;
	}

	private static String QueryAPI(String funtion, Map<String, String> params)
	{
		HttpURLConnection conn = null;
		BufferedReader br = null;
		StringBuilder response = new StringBuilder();
		try
		{
			byte[] postDataBytes = buildPostData(params);
			URL url = new URL(_APIUrl + funtion);
			conn = (HttpURLConnection) url.openConnection();
			conn.setRequestMethod("POST");
			conn.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
			conn.setRequestProperty("Content-Length", String.valueOf(postDataBytes.length));
			conn.setDoOutput(true);
			conn.getOutputStream().write(postDataBytes);
			br = new BufferedReader(new InputStreamReader(conn.getInputStream(), "UTF-8"));
			String line;
			while ((line = br.readLine()) != null)
			{
				response.append(line);
			}
		} catch (Exception ex)
		{
			ex.printStackTrace();
		} finally
		{
			if (conn != null)
			{
				conn.disconnect();
			}
			if (br != null)
			{
				try
				{
					br.close();
				} catch (Exception ex)
				{
					ex.printStackTrace();
				}
			}
		}
		System.out.println("response -- " + response.toString());
		return response.toString();
	}
```

PHP

CreateMember

```
function DoCreateMember()
 {
        echo sprintf('==DoCreateMember==<br>');

        $_CreateMemberResult = json_decode(CreateMember(),false);
        //var_dump($_CreateMemberResult);
        echo sprintf('error_code--->%s<br>', $_CreateMemberResult->error_code);
        echo sprintf('message--->%s<br>', $_CreateMemberResult->message);
        if ($_CreateMemberResult->error_code > 0)
        {
                // check error message code
                /*
                Code Text       Description
                ----------------------------
                1       Failed  Failed during executed
                2       Failed  User Name Dupliate
                3       Failed  OperatorId is incorrect
                4       Failed  Odds Type format error
                5       Failed  Currency format error
                6       Failed  Vendor_Member ID Duplicate
                7       Failed  MinTransfer > MaxTransfer
                9       Failed  Invalidate vendor_id
                10      Failed  System is under maintenance
                */
        }
        else
        {
                // Code Text    Description
                //    ----------------------------
                //0     OK      Successfully executed
        }
 }

function CreateMember()
 {
        global $_url ,$_httpService ,$_APIVendorID;
        echo sprintf('--CreateMember--<br>');
        $Funtion = "CreateMember";
        $Vendor_Member_ID = "XXXX";
        $FirstName = "XXXX";
        $LastName = "XXXX";
        $OddsType = "X";
        $Currency = "XX";
        $OperatorId = "XXX"; // the default value usually is site name
        $MaxTransfer = "XXXX";
        $MinTransfer = "XXXXX";
        $post_data = [
                'vendor_id' => $_APIVendorID,
                'Vendor_Member_ID' => $Vendor_Member_ID,
                'OperatorId' => $OperatorId,
                'FirstName' => $FirstName,
                'LastName' => $LastName,
                'UserName' => $Vendor_Member_ID,
                'OddsType' => $OddsType,
                'Currency' => $Currency,
                'MaxTransfer' => $MaxTransfer,
                'MinTransfer' => $MinTransfer,
        ];
        return $_httpService->sendPost($_url.$Funtion , $post_data);
 }
```

FundTransfer

```
function DoFundTransfer()
 {
        echo sprintf('==DoFundTransfer==<br>');
        $_FundTransferResult = json_decode(FundTransferFun(),false);
        var_dump($_FundTransferResult);
        echo sprintf('error_code--->%s<br>', $_FundTransferResult->error_code);
        echo sprintf('message--->%s<br>', $_FundTransferResult->message);
        echo sprintf('Data->status--->%s<br>', $_FundTransferResult->Data->status);
        if ($_FundTransferResult->error_code > 0)
        {
                $_CheckFundTransferResult = CheckFundTransferFun($_FundTransferResult->Data->trans_id);
                if ($_CheckFundTransferResult->error_code > 0)
                {
                        echo sprintf('error_code--->%s<br>', $_CheckFundTransferResult->error_code);
                        echo sprintf('message--->%s<br>', $_CheckFundTransferResult->message);
                        //1       Failed    Failed during executed
                        //2       Failed    Transaction record does not exist
                        //7       Failed    wallet_id input error
                        //9       Failed    Invalidate vendor_id
                        //10      Failed    System is under maintenance
                }
        }
        else
        {
                if ($_FundTransferResult->Data->status == 0)
                {
                        // 1. If status code is OK (0), transaction succeeds.    */
                }
                else if ($_FundTransferResult->Data->status == 1)
                {
                        //      2. If status code is Failed (1), please check the error code. Fix the error and try again later.
                }
                else if ($_FundTransferResult->Data->status == 2)
                {
                        //3. If status code is Pending (2) , please proceed "checkfundtransfer" mechanism.
                        $_CheckFundTransferResult = CheckFundTransferFun($_FundTransferResult->Data->trans_id);
                        if ($_CheckFundTransferResult->error_code > 0)
                        {
                                //1       Failed    Failed during executed
                                //2       Failed    Transaction record does not exist
                                //7       Failed    wallet_id input error
                                //9       Failed    Invalidate vendor_id
                                //10      Failed    System is under maintenance
                        }
                }
        }
}

function FundTransferFun()
{
        echo sprintf('==FundTransferFun==<br>');
        $vendor_trans_id = GetRandomString(20);
         return FundTransfer($vendor_trans_id);
}
function GetRandomString($len)
{
        $id_len = $len;
        $RandomString = '';
        $word = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890';
        $len = strlen($word);

        for($i = 0; $i < $id_len; $i++)
        {
                $RandomString .= $word[rand() % $len];
        }
        return $RandomString;
}

function FundTransfer($vendor_trans_id)
{
        global $_url ,$_httpService ,$_APIVendorID;
        echo sprintf('--FundTransfer--<br>');

        $Funtion = "FundTransfer";
        $Vendor_Member_ID = "test2018101101";
        $Amount = "10";
        $Currency = "20";
        $Direction = "1"; // 0 = Withdraw, 1= Deposit
        $Wallet_id = "";//Wallet ID 1 : Sportsbook 5 : AG 6 : GD

        $post_data = [
                'vendor_id' => $_APIVendorID,
                'vendor_member_id' => $Vendor_Member_ID,
                'vendor_trans_id' => $vendor_trans_id,
                'amount' => $Amount,
                'currency' => $Currency,
                'direction' => $Direction,
                'wallet_id' => $Wallet_id,
        ];
        return $_httpService->sendPost($_url.$Funtion , $post_data);
    }
```

CheckFundTransfer

```
    function DoCheckFundTransfer()
    {
        global $_url ,$_httpService;
        echo sprintf('==DoCheckFundTransfer==<br>');
        $trans_id = "XXXXXXXXXXXX";
        $_CheckFundTransferResult = json_decode(CheckFundTransferFun($trans_id),false);
        if ($_CheckFundTransferResult->error_code > 0)
        {
            // 1 Failed Failed during executed
            // 2 Failed Transaction record does not exist
            // 7 Failed wallet_id input error
            // 9 Failed Invalidate vendor_id
            // 10 Failed System is under maintenance
        }
    }

   function CheckFundTransferFun($vendor_trans_id)
    {
        echo sprintf('==CheckFundTransferFun==<br>');
        global $attempts;
        $_CheckFundTransferResult = json_decode(CheckFundTransfer($vendor_trans_id),false);
        //var_dump($_CheckFundTransferResult);
        echo sprintf('error_code--->%s<br>', $_CheckFundTransferResult->error_code);
        echo sprintf('message--->%s<br>', $_CheckFundTransferResult->message);
        if ($_CheckFundTransferResult->error_code > 0)
        {
            //      * e. If any exception occurred during the process, please continue with “checkfundtransfer” mechanism.*/
        }
        else
        {
            if ($_CheckFundTransferResult->Data->status == 0)
            {
                //a. If status code is OK (0), the queried transaction succeeded.
            }
            else
            {
                if ($_CheckFundTransferResult->Data->status == 1 || $_CheckFundTransferResult->Data->status == null)
                {
                    // * b. If status code is Failed (1), the queried transaction failed for some reason.
                    // * d. If the status code is null, please check error code and fix it before query again.
                }
                else if ($_CheckFundTransferResult->Data->status == 2)
                {
                    // * c. If status code is Pending (2) , please continue with “checkfundtransfer”. I. Repeat every 5 min until the solid response (statuscode 0
or 1) is received.
                    sleep(60);; //1 min
                    $attempts++;
                    if ($attempts > 3)
                    {
                        //contact OneWorks
                    }
                    else
                    {
                        CheckFundTransferFun($vendor_trans_id);
                    }
                }
            }
        }
        return $_CheckFundTransferResult;
    }

    function CheckFundTransfer($vendor_trans_id)
    {
        global $_url ,$_httpService ,$_APIVendorID;
        echo sprintf('--CheckFundTransfer--<br>');
        $Funtion = "CheckFundTransfer";
        $wallet_id = "";//Wallet ID 1 : Sportsbook 5 : AG 6 : GD
        $post_data = [
            'vendor_id' => $_APIVendorID,
            'vendor_trans_id' => $vendor_trans_id,
            'wallet_id'     => wallet_id,
        ];
```

```
            return $_httpService->sendPost($_url.$Funtion , $post_data);
    }
```

GetBetDetail

```
function DoGetBetDetail()
    {
            echo sprintf('==DoGetBetDetail==<br>');
            $Type = "Main";// NOTE: for Main sample only, modify result column if there is needs
            $lastVersionKey = XXXXXXXXXXXXxx;
            if ($Type == "Main")
            {
                    $_BetDetailResult = json_decode(GetBetDetail($lastVersionKey),false);

                    if ($_BetDetailResult->error_code > 0)
                    {
                            // Code Text Description
                            // ---------------------------
                            // 0 OK Successfully executed
                            // 1 Failed Failed during executed
                            // 9 Failed Invalidate vendor_id
                            // 10 Failed System is under maintenance
                    }
                    else
                    {
                            $lastVersionKey = $_BetDetailResult->Data->last_version_key;
                    }
            }
    }

    function GetBetDetail($VersionKey)
    {
            global $_url ,$_httpService ,$_APIVendorID;
            echo sprintf('--GetBetDetail--<br>');
            $Funtion = "GetBetDetail";
            $options = "";
            $post_data = [
                    'vendor_id' => $_APIVendorID,
                    'version_key' => $VersionKey,
                    'options'      => $options,
            ];

            return $_httpService->sendPost($_url.$Funtion , $post_data);
    }
```

LogIn

```
function DoLogIn()
    {
            global $_url ,$_httpService;
            echo sprintf('==DoLogIn==<br>');
            $_LogInResult = json_decode(LogIn(),false);
            echo sprintf('error_code--->%s<br>', $_LogInResult->error_code);
            echo sprintf('message--->%s<br>', $_LogInResult->message);
            if ($_LogInResult->error_code > 0)
            {
                    // Code Text Description
                    // ---------------------------
                    // 0 OK Successfully executed
                    // 1 Failed System Error
                    // 2 Failed member not found
                    // 9 Failed Invalidate vendor_id
                    // 10 Failed System is under maintenance
            }
    }

    function LogIn()
    {
            global $_url ,$_httpService ,$_APIVendorID;
            echo sprintf('--LogIn--<br>');

            $Funtion = "LogIn";

            $post_data = [
                    'vendor_id' => $_APIVendorID,
                    'domain' => '',
                    'vendor_member_id'      => 'XXX',
            ];
```

```
		return $_httpService->sendPost($_url.$Funtion , $post_data);
	}
```

POST function

```
	class HttpService
	{
		function sendPost($url, $post_data)
		{
			//open    CURL connectionstrings
			$ch=curl_init();
			curl_setopt($ch,CURLOPT_URL,$url);
			curl_setopt($ch,CURLOPT_RETURNTRANSFER,1);
			curl_setopt($ch,CURLOPT_POST,true);
			curl_setopt($ch,CURLOPT_POSTFIELDS, http_build_query($post_data));

			//process
			$result=curl_exec($ch);

			//close CURL connectionstring
			curl_close($ch);

			echo sprintf('result--->%s<br>', $result);
			return    $result;
		}
	}

	$_httpService = new HttpService;
	$_url='http://XX.X.XXX.XXX:XXXX/api/';
	$_APIVendorID='XXXXXXXXX';
```