



STREAMJAR

Project Description

Confidentiality

This document is not to be shared. All information and intellectual property contained in this document belong to StreamJar.co and its owner. Legal action will be taken if caught attempting to replicate these ideas for your own benefit. By continuing to read this document, you are acknowledging these circumstances and agree not to steal any intellectual property.

StreamJar

Site: <http://streamjar.co>

Demo: <http://streamjar.co/demo.html>

Overview:

Streamers on Twitch and YouTube (eventually other platforms) can sign up and create an account on StreamJar. On Sign Up, they can either sign up with email and password and create a unique username, or they can sign up directly with their Twitch or YouTube account.

Once signed up and logged in, users will be directed to their dashboard where they fill out profile information. This profile information will be displayed instead of the dashboard for users who aren't logged in. The dashboard will keep track of how much money the streamer has made in total since joining the site, and how much money has been earned since they have last been paid. Users who are not signed in can visit a registered user's profile and donate to the streamer by downloading apps, completing surveys, and other activities. These offers are fetched using a 3rd party api. When an offer is successfully completed, a postback is sent from the api containing information on the conversion. The StreamJar server must read this postback request and update the database based on the given information from the postback.

As an admin, I need an admin dashboard which will list out all registered users in the order of their unique account ID. I need their payment email, and unpaid balance listed. I need to be able to reset unpaid balance to 0. I don't need automatic PayPal payments implemented. I will do this manually, but I do need to reset paid balances. There will be other features needed on the Admin dashboard, but will be explained in more detail below.

This description and the fine details below are to be used as a guideline for this project. While I do have technical knowledge, and am familiar with good coding standards, I am no expert. This means that if you as the developer feel you can implement the same functionality in a more efficient way, then let me know and we can discuss a different way.

Data needed to be tracked for Registered User

- Account Details
 - Unique Account ID: ID used to track offer completion, start at 10000 for first registered user, and increment by 1 for each new user who signs up
 - Email
 - Username: Must be unique
 - Password
 - Total Money Earned
 - Unpaid Amount
 - Referral Earnings: Earnings the user has made from referrals
 - NumReferrals: Number of people the user has referred
 - ReferralPercentage: .05 should be the default value
 - Banned: True or false value which determines if the user can log in or not
 - Payment Percentage: .8 should be the default value
 - Referred By: Account ID of who referred the user during signup, Default is 00000/none
- Profile Details
 - Profile Picture: Either photo upload or link to hosted photo (we can discuss which would be best/most cost efficient at the start)
 - Profile Description: Description could be as long as a couple paragraphs
 - If no information is given, then fill in default values as a placeholder.
- Payment Details
 - Full Name
 - Payment Method: Only PayPal will be offered to start
 - PayPal Email

NoSQL User Structure

This is how I expect the following information to be organized based on my little knowledge of MongoDB database structures. If you have a better idea or need to add details to achieve the functionality necessary then I of course am open to a different structure.

User

- AccountID
- Email
- Username
- Password
- TotalEarned
- AmountUnpaid
- ReferralEarnings
- PayPercentage
- ReferralPercentage
- ReferredBy
- NumReferrals
- Banned
- ProfileDetails
 - Picture
 - Description
- PaymentDetails
 - FullName
 - PaymentMethod
 - PayPal Email

Pages Needed

- Home
 - Basic Placeholder for now. I will design/get this designed later.
- Sign Up
 - Options to Sign Up with:
 - Twitch
 - YouTube
 - New username, email, and password
 - If new username, email, and password is chosen, then email must be verified
 - Username must be unique
 - If signed up with Twitch, username = Twitch username
 - If signed up with YouTube, username = YouTube username
 - When Signing up with Twitch or YouTube, make the email, password, and username the same as their YouTube/Twitch email, password, and username
 - On Sign Up, the PaymentPercentage field should default to 80% which is .8
 - If the user is referred by another user, ReferredBy should be set to the AccountID of the user who referred the new user. If there is no referral set it to the default value
 - I was thinking referral links could end with ?r=AccountID to track if a user signs up with a referral. The AccountID would be the value ReferredBy is set to.
 - I am not too familiar with best practices for implementing a referral system, but I think this would work.
 - When a new user is referred by another user, NumReferrals for user with AccountID is incremented by 1
- Login
 - User can login with a Twitch, YouTube, or email and password
 - If the Twitch or YouTube account is not already connected to a user, then they are prompted to signup with that account.
 - If you can automatically create an account when a user tries to log in with a Twitch/ YouTube account, that would be even better.
 - Must have Forgot Password? Implementation to automatically reset their password if they forget it
 - Once a user logs in they should be sent to their User Dashboard
- Dashboard
 - Dashboard should be located at steamjar.co/u/username
 - Username should be the unique username for each user
 - At the top of the Dashboard, display the user profile picture, username, and AccountID
 - This isn't the main focus of the dashboard
 - At the top, under the username and accountID, display the total amount earned and the AmountUnpaid
 - Under this, there should be a profile section where their profile picture and description is displayed
 - There should be an edit button where the user can change their picture and description which can then be saved to the database
 - Under the Profile Section should be a referral section
 - This should display their referral links (<http://streamjar.co/signup?r=AccountID>)
 - It will also display the number of people they referred(NumReferrals), and total earnings from referrals (ReferralEarnings)

- User Profile Page

- This is the page non-logged in users see
- Streamjar.co/u/username
 - This shows the profile of username
- Basic example of what this page will look like: <http://streamjar.co/demo.html>
- To get the app and survey offers, you need to make an API call to the Ogads API which is given in a section below
- Required parameters for API
 - affiliateid: Always equals 21509
 - ip: IP of the person visiting the webpage
 - device: Type of device the person visiting the webpage is viewing on
 - ctype: Always equals 3
 - aff_sub3: AccountID of the registered user with the **username** from streamjar.co/u/username
 - aff_sub4: ReferredBy value of the registered user with the **username** from streamjar.co/u/username
 - aff_sub5: open for any use
- Important JSON info returned from API
 - offers: contains data for many offer elements
 - picture: link to image
 - name_short: name of offer
 - adcopy: description of offer
 - payout: amount offer pays
- Once the API returns a response, we must loop through each returned offer and display each offer in the following form
- Whole box can be clicked on and will redirect to *link* in a new tab




<i>picture</i>	<i>name_short</i> <i>adcopy</i>	<i>payout</i> * PaymentPercentage of the registered user with the username from streamjar.co/u/username
----------------	------------------------------------	---

- Postback

- Not an actual page a user will visit, but there must be a route that can update the Database based on POST requests sent from 3rd party APIs
- /postback/ogads will respond to the OGads API
 - Ogads API will be shown below
- From the postback, we should grab *aff_sub3*, *aff_sub4*, and *payout*
 - *aff_sub3* is the AccountID
 - *aff_sub4* is the ReferralID (ReferredBy)
 - *payout* is the Amount to be paid to the user
- Since you have the AccountID, you can update the users TotalEarned and AmountUnpaid
 - TotalEarned and AmountUnpaid += (payout * PaymentPercentage)
- Since you have the AccountID of the person who referred the user, you can also update the amount of money they earned. If *aff_sub4* is the default value(nobody referred this user) then don't do anything. If *aff_sub4* matches the AccountID of another user, then update the following.
 - Since this person's account is earning a percentage of the income from a user they referred, we must update TotalEarned, AmountUnpaid, and ReferralEarnings
 - TotalEarned, AmountUnpaid, ReferralEarnings += (payout * ReferralPercentage)

- Admin Dashboard
 - At the top Display the Total Unpaid Balance (sum of all users unpaid balance)
 - Will this be inefficient when many many users sign up?
 - Under the total, show a list of users in order by AccountID
 - Above the list I need a button that when clicked reloads the page and shows all users in the list
 - I need a second button that prompts me for an amount and will then reload the page and show only the users who have an unpaid balance above the amount I entered
 - In the list display an Edit Button, Account ID, Username, Payment Email, Amount Unpaid, Button to Pay
 - When the Pay Button is clicked, AmountUnpaid for that user goes to zero.
 - When setting AmountUnpaid to zero, DO NOT just set the database entry to 0. You must subtract the value of AmountUnpaid shown in the list from the value currently in the database
 - This should eliminate the error where I pull up the admin dashboard and AmountUnpaid is changed in the database because a new offer is completed but my screen wouldn't reflect the new database value until the page is reloaded
 - Clicking the edit button should bring up all details about a user's account and allow me to change the following fields if necessary
 - Email: I should be able to set this to a new email if needed
 - Password: Change to any password in case manual reset is needed
 - TotalEarned: Should be able to add/subtract any amount if needed
 - Adding/Subtracting from TotalEarned should also Add/Subtract from AmountUnpaid
 - PayPercentage: Should be able to change to any value between 0 and 1
 - ReferralPercentage: Should be able to change to any value between 0 and 1
 - ReferredBy: Should be able to change to any AccountID (no need to check if account exists, I will do this myself)
 - Banned: I should be able to set banned to true or false
 - ProfileDetails: I should be able to set the ProfilePicture and Description to whatever I want if needed
 - PaymentDetails: Option to clear all payment details (Full Name, PaymentType, PaymentEmail)

Example of List

Edit	Account ID	Username	PayPal Email	Amount Unpaid	Pay
	10000	User1	User1@gmail.com	\$150.00	PAY BUTTON
	10001	User2	User2@gmail.com	\$0.00	PAY BUTTON
	10002	User3	User3@gmail.com	\$1,000.00	PAY BUTTON

Ogads API

API Endpoint: https://mobverify.com/api/v1/		
Parameter	Type	Description
<i>affiliateid</i>	Integer	Ogads account affiliate ID: <u>21509</u>
<i>ip</i>	String	API autodetects country from user's IP
<i>device</i>	String	The mobile device name you want the api to return offers for. Can be <u>android</u> , <u>desktop</u> , <u>ipad</u> , or <u>iphone</u> .
<i>ctype</i>	Integer	*Does not have any effect if device is desktop. Bitwise flag, you should add the following numbers together: <ul style="list-style-type: none"> • 1 CPI • 2 CPA
<i>aff_sub3</i>	String	Additional data to include for advanced users
<i>aff_sub4</i>	String	Additional data to include for advanced users
<p>Example Call: https://mobverify.com/api/v1/?affiliateid=21509&ip=173.78.255.255&ctype=3&aff_sub3=AccountID&aff_sub4=ReferralID</p>		
<p>Example Response: Bold Responses are important for StreamJar</p> <pre>{ "success": true, "error": null, "offers": [{ "offerid": "20696", "name": "Portal Quest - Android, US, CPI ", "name_short": "Portal Quest", "description": "Enter the world of Portal Quest, the next generation of RPG mobile games! Gather and lead your team of battle-ready Heroes to protect against a corrupt force! The Hollow is poised to take over the World as you know it. It\u2019s up to your team to wage through battles to hold the Hollow back. Your enemies will only get tougher so assembling the best team of battle-ready Heroes is essential. Summon your team to take down the enemy while upgrading them to the finest Heroes Portal Quest has seen. It\u2019s the only way we will be able to hold the Hollow back and defeat them for good!
\nConversion: Install & Open.
\nTraffic Restrictions: Custom Creatives; Adult Traffic; Push Notification Traffic; Ads Icon Traffic; SMS Traffic; Discovery App Traffic; Instagram/Twitter Traffic
\nCaps: This campaign is capped, so if you plan on doing more than 150 leads a day check with your Account Manager for additional budget.
\nCreative URL: https://www.dropbox.com/sh/ivkizpguwhg5jyg/AAArTlrm7Z3lrG2LFz8SPFnTa?dl=0
\n
\nag0001=\"true\",campaign_type=\"offerwall\",offerwall_description=\"Enter the world of Portal Quest, the next generation of RPG mobile games!\",offerwall_instructions=\"Download and Open the App,Redeem your points!\",offerwall_category1=\"app\",offerwall_category2=\"free\",platform=\"android\",\"adcopy\":\"Install and complete level 10 to unlock this content.\",\"picture\":\"https://media.go2speed.org/brand/files/ogmobi/20696/thumbnails_100/thumbnail-18563227195b86dcc2047c32.84347395.png\",\"payout\":\"0.36\", \"country\":\"US\", \"device\":\"Android\", \"link\":\"http://jump.ogtrk.net/aff_c?aff_id=21509&offer_id=20696\", \"epc\":\"0.17634\", {\"offerid\":\"20086\", \"name\":\"Hotspot Shield Free VPN Proxy - all, instl. 1473 (Android, INCENT, Free, US, 42M)\", \"name_short\":\"Hotspot Shield Free VPN Proxy\", ...}}"</pre>		

Ogads Postback

Macro	Macro Description
{offer_id}	ID of offer
{offer_name}	Name of offer
{affiliate_id}	ID of affiliate
{source}	Source value specified in the tracking link
{aff_sub3}	Affiliate sub 3 specified in the tracking link
{aff_sub4}	Affiliate sub 4 specified in the tracking link
{aff_sub5}	Affiliate sub 5 specified in the tracking link
{session_ip}	IP address that started the tracking session
{ip}	IP address that made the conversion request
{date}	Current date of conversion formatted as YYYY-MM-DD
{time}	Current time of conversion formatted as HH:MM:SS
{datetime}	Current date and time of conversion formatted as YYYY-MM-DD HH:MM:SS
{ran}	Randomly generated number
{payout}	Amount paid to affiliate for conversion