**Trackmini API Technical Documentation**

**Description**: The Trackmini API is designed to allow for interaction with a database by any client through use of an HTTP web service written in PHP. The REST API allows any client to retrieve, create, update, or delete rows in a MySQL database.

High Level Overview

A client submits a HTTP request to the URL containing the API files. The parameters of this URL (determined by the content after the API location) are used to inform the service of the action being requested.

Request Overview

TODO

API Overview

The API is installed in a specific API directory on the server. This directory contains a .htaccess file, that instructs the web server to forward all requests made starting with this URL to the **api.php** file in this directory for processing. The original URL is forwarded to the API to be processed.

**Api.php**

All requests to the API will be forwarded to this “landing” script. This class can be used for authorizing the request, but is currently disabled for development. The script will create a generic API object, pass it the request data, and call it’s processAPI method to process the request.

**API.class.php**

Class Variables

* Method – GET, POST, PUT, DELETE
* Endpoint – The name of the Model (database table) requested in the URI.
* Verb – An optional instruction for the endpoint about how to process the data.
* Args – Arguments are found after the endpoint and verb have been parsed from the URI.
* File – Stores the input of a POST, PUT, or DELETE request.

The API class is responsible for receiving the request, validating it, parsing out provided parameters, and forwarding it to be further processed. Upon construction, it will parse the URI, extracting the endpoint, verb, and args. Depending on the type of request (i.e. PUT), it will extract the information from the php://input and store it in the File. After exiting the constructor, the API class has all of the information it needs to route the request and pass along the necessary parameters.

processAPI – based on the endpoint name, this function routes the request for further processing. There are 3 overall options for where a request is routed.

1. API Function – it checks the API itself for a function with the same name as the endpoint, and routes it there if it exists.
2. Class Function – it checks if a class exists with the same name as the endpoint. The class must exist in the “Model” directory of the API, and the file must be named ‘<Class>.class.php’. This is used when creating an endpoint with custom logic. If one is found, it will create an instance of the class and route the request to it.
3. Generic Function – if no API method or class is found, the API will attempt to route the request to a generic parsing function. The generic functions support the basic CRUD needs of the service.