***Enterprise*** ***Chat*** ***System***

**DEVELOPER** **DOCUMENT**

**Version:** **1.1**

**Team** **Name: Wireshark**

**Team** **Members:** **Date: 20/05/2017**

**1. Kovvuru Nasirali**

**2. Ginka Anusha**

**3. Hossen Saddam**

**4. Gurram Karthik**

**5. Injeti Ravi Varma**

**6. Ivvala Venkata Sai Krishna Chaitanya**

**7. Janagam Anirudh**

**8. Kadamati Venkata Sai Sidhartha**

**9. Karumanchi Mahesh Chowdary**

**10. Kondepati Divya Naga Krishna**

**11. Sathi Santhosh Reddy**

**Type of Document:** Developer documentation

**Version Number :**1.1

**Publication Date:** June 11th 2017

**1.Preface:**

The enterprise chat system product is developed and implemented to ensure the secure communication between the users. The document describes about the source code organisation, database table format, data format when communicating with the system.

**Release v1.1 on 11-06-2017**

* Changes are made in section 4 : Code used for API functioning is shown
* Changes made in section 6 : Structure of Database is clearly explainedwith the help of uml diagrams.

**Release v1.0 on 19-05-2017**

* initial release

The document is described and organised as follows. In Section 2, Glossary and Abbreviations of terms used in the document are provided. Section 3, explains about the introduction of the document. Sections 4, describes the software description of the product. Section5, deals with the system architecture. Section 6, deals with the structure of the database. Section 7 , deals with extension of the product.

**2.GLOSSARY AND ABBREVIATIONS:**

**GUI:** Graphical User Interface

**API:** Application programming interface

**PDF:** Portable document format

**Project Libre:** Project management software system

**SQL:** Standard Queuing Language

**IP ADDRESS:** Internal Protocol Address

**OFFLINE:** It implies that the client is not connected with the system.

**AVAILABLE:** It shows the state where the client is associated and prepared to start a

discussion.

**BUSY:** It shows the state where the client is connected but unable to start a discussion.

**IDLE:** It shows the state that the client is connected to the system yet there is no response

from the client.

**PHP:** Hypertext Pre-processor

**HTML:** Hyper Text Mark-up Language

**JSON:** JavaScript Object Notation

**REST:** Representational State Transfer

**MYSQL:** An open source relational database management system.

**3. INTRODUCTION**:

This document gives a brief overview of the technical aspects of the product. It helps the developers in assistance for adding new functionalities.

**4.SOFTWARE DESCRIPTION OF THE PRODUCT:**

The backend of the product is written in PHP script, AJAX and JQUERY. The front end of the product is written in HTML, CSS and Javascript which shows the various contents like chat box, login form, address book, status of the users on web based GUI. We used MySQL for designing the Database.

Restful API:

**RESTFUL API** is used between user to sever communication along with JSON data encoding.

Representational State Transfer has become the standard design architecture for developing web APIs. RESTful web services is one way of providing interoperability between computer systems on the Internet. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations.

REST takes advantage of the HTTP request methods to layer itself into the existing HTTP architecture. These operations consist of the following:

GET - Used for basic read requests to the server

PUT- Used to modify an existing object on the server

POST- Used to create a new object on the server

DELETE - Used to remove an object on the server

The product interacts with server and with user is based on RESTful API with JSON data encoding. This can be done through different methods like GET and POST. The information exchange between the database and web page will be in JSON format.

Get and post api are used in the entire system these APIs are tested using https and curl command line.

An Application Programming Interface (API) allows for publicly exposed methods of an application to be accessed and manipulated outside of the program itself.

Restful API is used for the HTTP communication. Restful API’s are designed to access the database from different platforms.

In the project HTML, CSS &JAVASCRIPT is used for the display of the GUI. In the project, PHP version 5 is used and it is embedded with the HTML code. MYSQL database is used to store the users details and a friendly GUI has been included in the XAMPP frame work to access the database. Required databases and tables are configured using URL <http://localhost/phpmyadmin> (NOTE: This project is made for the local users who are connected to switch. So by using internet and by following the link provided below will not let you go to this application , it is exclusively created for the users who are connected to the LAN by means of a switch)

The user must be provided with a web based GUI to perform various operations like editing messages, deleting messages, selecting a user from a contact list etc.

When the user enters his login credentials in the login form, His credentials needed to be authenticated and the user must be redirected to profile page once the credentials match with the credentials from the database.

It can be operated by server run command ‘curl – <http://localhost/example.php>

The testing of API’s working is tested with curl.

Code:

<?php

class REST {

public $\_allow = array();

public $\_content\_type = "application/json";

public $\_request = array();

private $\_method = "";

private $\_code = 200;

public function \_\_construct(){

$this->inputs();

}

public function get\_referer(){

return $\_SERVER['HTTP\_REFERER'];

}

public function response($data,$status){

$this->\_code = ($status)?$status:200;

$this->set\_headers();

echo $data;

exit;

}

private function get\_status\_message(){

$status = array(

100 => 'Continue',

101 => 'Switching Protocols',

200 => 'OK',

201 => 'Created',

202 => 'Accepted',

203 => 'Non-Authoritative Information',

204 => 'No Content',

205 => 'Reset Content',

206 => 'Partial Content',

300 => 'Multiple Choices',

301 => 'Moved Permanently',

302 => 'Found',

303 => 'See Other',

304 => 'Not Modified',

305 => 'Use Proxy',

306 => '(Unused)',

307 => 'Temporary Redirect',

400 => 'Bad Request',

401 => 'Unauthorized',

402 => 'Payment Required',

403 => 'Forbidden',

404 => 'Not Found',

405 => 'Method Not Allowed',

406 => 'Not Acceptable',

407 => 'Proxy Authentication Required',

408 => 'Request Timeout',

409 => 'Conflict',

410 => 'Gone',

411 => 'Length Required',

412 => 'Precondition Failed',

413 => 'Request Entity Too Large',

414 => 'Request-URI Too Long',

415 => 'Unsupported Media Type',

416 => 'Requested Range Not Satisfiable',

417 => 'Expectation Failed',

500 => 'Internal Server Error',

501 => 'Not Implemented',

502 => 'Bad Gateway',

503 => 'Service Unavailable',

504 => 'Gateway Timeout',

505 => 'HTTP Version Not Supported');

return ($status[$this->\_code])?$status[$this->\_code]:$status[500];

}

public function get\_request\_method(){

return $\_SERVER['REMOTE\_ADDR'];

}

private function inputs(){

switch($this->get\_request\_method()){

case "POST":

$this->\_request = $this->cleanInputs($\_POST);

break;

case "GET":

case "DELETE":

$this->\_request = $this->cleanInputs($\_GET);

break;

case "PUT":

parse\_str(file\_get\_contents("php://input"),$this->\_request);

$this->\_request = $this->cleanInputs($this->\_request);

break;

default:

$this->response('',406);

break;

}

}

private function cleanInputs($data){

$clean\_input = array();

if(is\_array($data)){

foreach($data as $k => $v){

$clean\_input[$k] = $this->cleanInputs($v);

}

}else{

if(get\_magic\_quotes\_gpc()){

$data = trim(stripslashes($data));

}

$data = strip\_tags($data);

$clean\_input = trim($data);

}

return $clean\_input;

}

private function set\_headers(){

header("HTTP/1.1 ".$this->\_code." ".$this->get\_status\_message());

header("Content-Type:".$this->\_content\_type);

}

}

?>

Rest\_api.php

Displaying login\_api.php.

**5 SYSTEM ARCHITECTURE:**

**5.1 PROGRAMMING LANGUAGES USED:**

• Html

• PHP

• Java Script

• MySql

• Ajax

• Jquery

* MySQL Database
* JSON

6.SOURCE CODE IS ORGANISED AS FOLLOWS

Front end:

* Index.php
* Connect.php
* Chat.php
* Logout.php
* Time ago.php
* Verify.php
* get\_message\_ajax.php
* Mail.php
* post\_message\_ajax.php

DATABASE:

* Chat.sql

JAVASCRIPT:

* Bootstrap.js
* Jquery.js
* Script.js

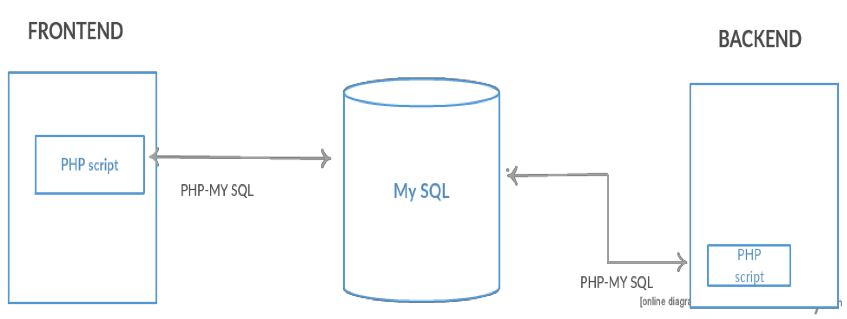
CSS:

* Bootstrap.css
* Style.css

**6.STRUCTURE OF THE DATABASE**

The product uses single database with its name as chat consists of three tables namely:

Admin functionalities are mainly dependent on the database ( delete an user, changing user details, adding users, activity of users etc are monitored using database of our system).



**Users:** In users table contains the users details such as id, username, email, password, ip, time and activation status.  
Id - primary key  
Email-unique key

**TABLE1: Online users:** In online users table contains the users online status this table consists of id, user id, ip, check in, status, checkout.  
Id – primary key  
User id – foreign key reference (users.id).

**TABLE2: Chat:** chat table consists of the chat details such as id, user1, user2, time, ip.  
Id – primary key  
User1, user2 – foreign key reference(users.id).  
**Message:** message table consists of the messages of the users it consists of id, chat\_id, user\_id, message, type, size.

Id -primary key

User\_id - foreign key reference (users.id)

Chat\_id – foreign key reference(chat.id)

**TABLE 3: Broadcast Chat:** chat table consists of the chat details such as id, user1, user2, time, ip.

Id – primary key  
User1, user2 – foreign key reference(users.id).  
**TABLE4: Broadcast Message:** message table consists of the messages of the users it consists of id, chat\_id, user\_id, message, type, size.  
Id -primary key  
User\_id - foreign key reference (users.id)  
Chat\_id – foreign key reference (Broadcast chat.id) .

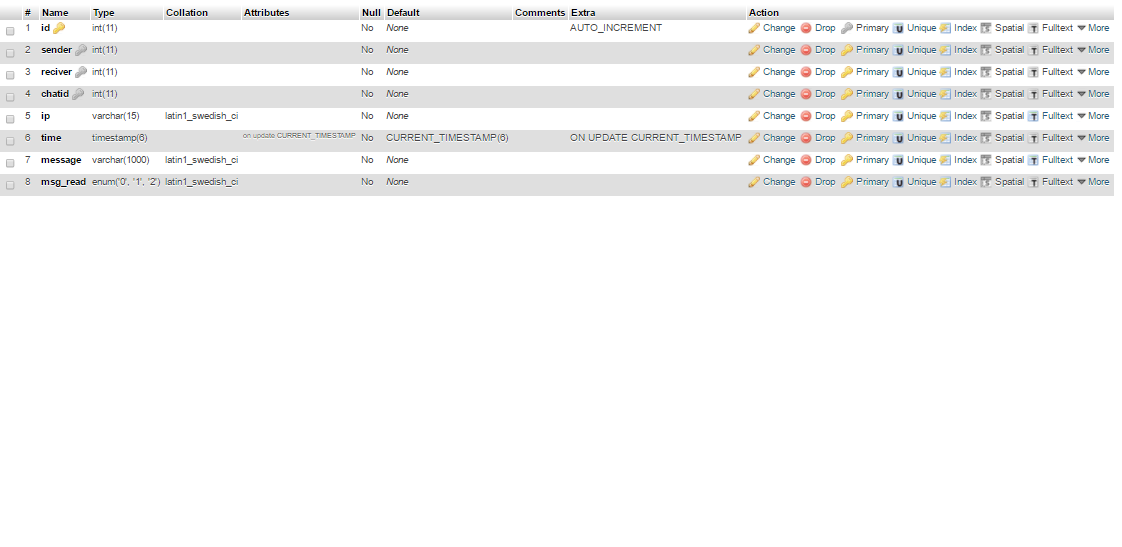
**Credentials table used for storing the details of the users and login/logout status.**

****

**Chat table used for storing the conversation between the users.**



3. Message storage tablets show the messages and to show Read/Unread status.



References:

[1] Software Documentation, https://en.wikipedia.org/wiki/Softwaredocumentation

[2] Ian Sommerville. Software Engineering. 9th Ed.