**LAMPIRAN**

* 1. **Source Code**

*Source Code* merupakan bahasa pemrograman atau barisan kode program yang telah di buat oleh penulis pada sistem yang dijalankan, beberapa kode yang terpenting dapat dilihat dibawah ini, diantaranya:

1. **Login**

<?php

header('Access-Control-Allow-Origin: \*');

defined('BASEPATH') OR exit('No direct script access allowed');

class Login\_m extends CI\_Model

{

private $\_mongoDb;

function \_\_construct() {

parent::\_\_construct();

$this->load->database();

}

public function getUserData($data) {

$password = $data['password'];

$email = $data['email'];

$this->db->where('email', $email);

$user = $this->db->get('tb\_users')->result();

if(!empty($user)){

$salt = $user[0]->salt;

$hash = $this->checkhashSSHA($salt, $password);

$this->db->where('email', $email);

$this->db->where('encrypted\_password', $hash);

$query = $this->db->get('tb\_users')->row();

return $query;

} else {

return "";

}

}

public function update\_password($email, $password){

$hash = $this->hashSSHA($password);

$encrypted\_password\_sent = $hash["encrypted"];

$salt = $hash["salt"];

$data = array(

"encrypted\_password" => $encrypted\_password\_sent,

"salt" => $salt

);

$this->db->where('email', $email);

$update = $this->db->update('tb\_users', $data);

if(!$update) {

$error = $this->db->error();

return ['status' => false, 'message' => $error['message']];

}

return ['status'=> true, 'message' => 'Reset password berhasil'];

}

//fungsi untuk cek encrypted password

private function checkhashSSHA($salt, $password) {

$hash = base64\_encode(sha1($password . $salt, true) . $salt);

return $hash;

}

//fungsi untuk encrypt password

private function hashSSHA($password) {

$salt = sha1(rand());

$salt = substr($salt, 0, 10);

$encrypted = base64\_encode(sha1($password . $salt, true) . $salt);

$hash = array("salt" => $salt, "encrypted" => $encrypted);

return $hash;

}

}

1. **Master Data**

<?php

header('Access-Control-Allow-Origin: \*');

defined('BASEPATH') OR exit('No direct script access allowed');

class Master\_m extends CI\_Model

{

private $\_mongoDb;

function \_\_construct() {

parent::\_\_construct();

$this->load->database();

}

public function get\_data($tb)

{

return $this->db->get($tb);

}

public function get\_data\_fields($tb)

{

$data = $this->db->get($tb)->result();

$fields = $this->db->list\_fields($tb);

return array("data" => $data, "fields" => $fields);

}

public function get\_data\_where($where, $val, $tb)

{

$this->db->where($where, $val );

return $this->db->get($tb)->result();

}

public function hapus\_data($where, $table){

$this->db->where($where);

$this->db->delete($table);

}

public function tambah\_data($table, $data){

if(!$this->db->insert($table, $data)) {

$error = $this->db->error();

return ['status' => false, 'message' => $error['message']];

}

return ['status' => true, 'message' => 'Data berhasil ditambahkan.'];

}

public function ubah\_data($table, $data, $param, $id){

$where[$param] = $id;

if(!$this->db->update($table, $data, $where)) {

$error = $this->db->error();

return ['status' => false, 'message' => $error['message']];

}

return ['status'=> true, 'message' => 'Data berhasil diubah'];

}

public function checkEmail($email, $table){

$this->db->where('email', $email);

$isemailExist = $this->db->get($table)->result();

return $isemailExist;

}

public function checkPass($pass, $table){

$this->db->where('encrypted\_password', $pass);

$result = $this->db->get($table)->result();

if(!$result) {

return false;

}

return true;

}

public function get\_summary(){

$date = date('Y-m');

$this->db->select('nama\_mobil, COUNT(id) as total');

$this->db->group\_by('nama\_mobil');

$this->db->where("DATE\_FORMAT(tgl\_booking,'%Y-%m')", $date);

$this->db->order\_by('total', 'desc');

$query = $this->db->get('tb\_book')->result();

return $query;

}

public function get\_outsp(){

$this->db->select('nama\_sparepart, mobil');

$this->db->where('jumlah', 0);

$this->db->order\_by('mobil', 'asc');

$query = $this->db->get('tb\_sparepart')->result();

return $query;

}

public function get\_jml\_status(){

$this->db->select('status, COUNT(id) as total');

$this->db->group\_by('status');

$query = $this->db->get('tb\_book')->result();

return $query;

}

}

1. **Data Utama**

<?php

if (!defined('BASEPATH'))

exit('No direct script access allowed');

class Data\_m extends CI\_Model {

public $table = 'tb\_checklist';

public $id = 'id\_checklist';

public $order = 'DESC';

function \_\_construct() {

parent::\_\_construct();

}

//ini untuk memasukkan kedalam tabel pegawai

function loaddata($dataarray) {

for ($i = 0; $i < count($dataarray); $i++) {

$data = array(

'nama\_checklist' => $dataarray[$i]['nama\_checklist'],

);

//ini untuk menambahkan apakah dalam tabel sudah ada data yang sama

//apabila data sudah ada maka data di-skip

// saya contohkan kalau ada data nama yang sama maka data tidak dimasukkan

//$cek = $this->db->where('nama\_checklist', $this->input->post('nama\_checklist'));

//if ($cek) {

return $this->db->insert($this->table, $data);

// }

}

}

}

1. **Transaksi**

<?php

header('Access-Control-Allow-Origin: \*');

defined('BASEPATH') OR exit('No direct script access allowed');

class Transaction\_m extends CI\_Model

{

private $\_mongoDb;

function \_\_construct() {

parent::\_\_construct();

$this->load->database();

}

public function get\_data($tb)

{

return $this->db->get($tb);

}

public function get\_data\_id($uid, $table)

{

$this->db->where('uniqueid\_book', $uid);

return $this->db->get($table);

}

public function update\_data($table, $data, $param, $id){

$where[$param] = $id;

return $this->db->update($table, $data, $where);

}

public function get\_report() {

$this->db->where('status', 'declined');

$this->db->or\_where('status', 'finished');

return $this->db->get('tb\_book');

}

public function get\_active\_trans() {

$this->db->where('status', 'new');

$this->db->or\_where('status', 'inprocess');

return $this->db->get('tb\_book');

}

public function tambah\_data($table, $data) {

if(!$this->db->insert($table, $data)) {

$error = $this->db->error();

return ['status' => false, 'message' => $error['message']];

}

return ['status' => true, 'message' => 'Data berhasil ditambahkan.'];

}

public function get\_data\_spc($mobil) {

$this->db->select('\*');

$this->db->from('tb\_sparepart');

$this->db->where('mobil', $mobil );

return $this->db->get();

}

public function hapus\_data($where, $table){

$this->db->where($where);

$this->db->delete($table);

}

}

1. **Report**

<?php

defined('BASEPATH') OR exit('No direct script access allowed');

class Report extends CI\_Controller {

public function \_\_construct() {

parent::\_\_construct();

$this->load->model('Transaction\_m');

if(!$this->session->userdata('userid')){

redirect('login');

}

}

public function index()

{

$data['page']="content/report\_v";

$data['book'] = $this->Transaction\_m->get\_report()->result();

$this->load->view('themes/sidebar\_v', $data);

}

public function detail\_report($uid)

{

$data['page']="content/reportdetail\_v";

$data['booking'] = $this->Transaction\_m->get\_data\_id($uid, 'tb\_book')->result();

$data['checkb'] = $this->Transaction\_m->get\_data\_id($uid, 'tb\_checklist\_booking')->result();

$data['spb'] = $this->Transaction\_m->get\_data\_id($uid, 'tb\_sparepart\_booking')->result();

$this->load->view('themes/sidebar\_v', $data);

}

}

?>

1. **Upload**

<?php

defined('BASEPATH') OR exit('No direct script access allowed');

class Upload extends CI\_Controller

{

function \_\_construct()

{

parent::\_\_construct();

// load helper dan library

$this->load->helper(array('form', 'url'));

$this->load->library('form\_validation');

}

public function index($error = NULL)

{

$data = array(

'action' => site\_url('upload/proses'),

'judul' => set\_value('judul'),

'error' => $error['error'] // ambil parameter error

);

$this->load->view('content/upload\_v', $data);

}

public function proses()

{

// validasi judul

$this->form\_validation->set\_rules('judul', 'judul', 'trim|required');

if ($this->form\_validation->run() == FALSE) {

// jika validasi judul gagal

$this->index();

} else {

// config upload

$config['upload\_path'] = './temp\_upload/';

$config['allowed\_types'] = 'xls';

$config['max\_size'] = '10000';

$this->load->library('upload', $config);

if ( ! $this->upload->do\_upload('userfile')) {

// jika validasi file gagal, kirim parameter error ke index

$error = array('error' => $this->upload->display\_errors());

$this->index($error);

} else {

// jika berhasil upload ambil data dan masukkan ke database

$upload\_data = $this->upload->data();

// load library Excell\_Reader

$this->load->library('Excel\_reader');

//tentukan file

$this->excel\_reader->setOutputEncoding('230787');

$file = $upload\_data['full\_path'];

$this->excel\_reader->read($file);

error\_reporting(E\_ALL ^ E\_NOTICE);

// array data

$data = $this->excel\_reader->sheets[0];

$dataexcel = Array();

for ($i = 1; $i <= $data['numRows']; $i++) {

if ($data['cells'][$i][1] == '')

break;

$dataexcel[$i - 1]['nama\_checklist'] = $data['cells'][$i][1];

}

//load model

$this->load->model('Data\_m');

$query = $this->Data\_m->loaddata($dataexcel);

print\_r($query);

//delete file

$file = $upload\_data['file\_name'];

$path = './temp\_upload/' . $file;

unlink($path);

}

//redirect ke halaman awal

// redirect(site\_url('out'));

}

}

}

1. **Search**

<?php

defined('BASEPATH') OR exit('No direct script access allowed');

require APPPATH . '/libraries/REST\_Controller.php';

use Restserver\Libraries\REST\_Controller;

/\*\*

\* THIS CODE IS REUSEABLE FOR EACH TABLE IN EVERY DB

\*/

class Search extends REST\_Controller {

function \_\_construct($config = 'rest')

{

parent::\_\_construct($config);

$this->load->database();

}

//Menampilkan semua data yang mengandung parameter yang disebutkan

function index\_get($table\_name) {

//initialize variable

$param = array();

$where = '';

//baca nama kolom dalam tabel.

$table\_col = $this->db->list\_fields($table\_name);

//ambil id dari table, id di table ada pada kolom pertama, maka itu $table\_col[0]

$id = $this->get($table\_col[0]);

//kolom id dipisah sendiri agar terfilter dengan spesifik

if($id){

$where .= $table\_col[0].' = '.$this->get($table\_col[0]);

}

//update where berdasar parameter lain yang diberikan di url

$where = $this->create\_where($table\_col, $where);

if (! empty($where)) {

$this->db->where($where);

$data = $this->db->get($table\_name)->result();

} else {

$query = $this->db->get($table\_name);

if(!$query){

$error = $this->db->error();

$data = array('status' => "NG", 'error' => $error['message']);

} else {

$data = array('status' => "ok", 'hasil' => $query->result());

}

}

$this->response($data, 200);

}

//Mengirim atau menambah data baru

function index\_post($table\_name){

$table\_col = $this->db->list\_fields($table\_name);

for($i = 0; $i <= count($table\_col)-1; $i++) {

$col = isset($table\_col[$i]) ? $table\_col[$i] : "";

$value = $this->post($col);

if($value == NULL) {

$value = "";

}

$data1[$col] = $value;

}

$insert = $this->db->insert($table\_name, $data1);

if ($insert) {

$this->response($data1, 200);

} else {

$this->response(array('status' => 'fail', 502));

}

}

//Memperbarui data kontak yang telah ada

function index\_put($table\_name){

$id = $this->put('id');

$table\_col = $this->db->list\_fields($table\_name);

for($i = 0; $i <= count($table\_col)-1; $i++) {

$col = isset($table\_col[$i]) ? $table\_col[$i] : "";

$value = $this->put($col);

$data1[$col] = $value;

}

$this->db->where('id', $id);

$update = $this->db->update($table\_name, $data1);

if ($update) {

$this->response($data1, 200);

} else {

$this->response(array('status' => 'fail', 502));

}

}

//Menghapus salah satu data kontak

function index\_delete($table\_name) {

$id = $this->delete('id');

$this->db->where('id', $id);

$delete = $this->db->delete($table\_name);

if ($delete) {

$this->response(array('status' => 'success'), 201);

} else {

$this->response(array('status' => 'fail', 502));

}

}

//get where clause

function create\_where($table\_col, $where){

$param = array();

//cek & buat parameter dengan isi sesuai parameter pada url (fungsi $this->get)

for($i = 1; $i <= count($table\_col); $i++) {

$col = isset($table\_col[$i]) ? $table\_col[$i] : "";

array\_push($param, $this->get($col));

//create parameter where lain jika param tersedia

if(! empty($param[$i-1])){

$par = isset($param[$i-1]) ? $param[$i-1] : "";

$where .= (empty($where)? '' : ' or ').$col ." like '%".$par."%'";

}

}

return $where;

}

}

?>