DAFTAR LAMPIRAN

Lampiran - 1 Surat Balasan Penelitian.



PEMERINTAH KOTA PAREPARE
DINAS LINGKUNGAN HIDUP
Jalan Jend. Ahmad Yani Km. 6 Parepare, Telp. (0421) 23949
Kode Pos 91132, Email : dlh@pareparekota.go.id

SURAT KETERANGAN
TELAH MELAKSANAKAN PENELITIAN
Nomor : 600.4/319/ DLH

Yang bertanda tangan dibawah ini :

NAMA	: SUSIANNA S., S.IP
NIP	: 19760423 199412 2 001
PANGKAT/ GOLONGAN	: PEMBINA TK. I, (IV/b)
JABATAN	: KEPALA DINAS LINGKUNGAN HIDUP

Menerangkan bahwa mahasiswa (i) yang tersebut di bawah ini :

N A M A	: ST. KHAERIANI AKHMAR
UNIVERSITAS/LEMBAGA	: UNIVERSITAS MUHAMMADIYAH PAREPARE
JURUSAN	: TEKNIK INFORMATIKA
ALAMAT	: JL. SIRATAL MUSTAKIM PAREPARE

Telah melaksanakan penelitian pada Dinas Lingkungan Hidup Kota Parepare sesuai dengan surat Rekomendasi Penelitian Nomor : 679/IP/DPM-PTSP/8/2024 tanggal 26 Agustus 2024 untuk memperoleh Data/Keterangan dengan judul penelitian: "APLIKASI PENGELOLAAN DOKUMEN PADA DINAS LINGKUNGAN HIDUP KOTA PAREPARE"

Demikian Surat Keterangan ini dibuat dengan sesungguhnya untuk dapat dipergunakan sebagaimana mestinya.

Parepare, 28 Oktober 2024
KEPALA DINAS LINGKUNGAN HIDUP
KOTA PAREPARE

SUSIANNA S., S.IP
Pembina TK. I, (IV/b)
NIP. 19760423 199412 2 001

Lampiran - 2 Listing Program

--config

```
<?php
define("DEFAULT_TIMEZONE", "Asia/Makassar"); // set php date functions timezone
define("DEVELOPMENT_MODE" , true);// set to false when in production
// return full path of application directory
define("ROOT", str_replace("\\", "/", dirname(__FILE__)) . "/");
// return the application directory name.
define("ROOT_DIR_NAME", basename(ROOT));
define("SITE_NAME", "Dinas Lingkungan Hidup");
// Get Site Address Dynamically
$site_addr = (isset($_SERVER["HTTPS"]) && $_SERVER["HTTPS"] !== "off" ? "https" : "http") . "://" . $_SERVER["HTTP_HOST"] . dirname($_SERVER["SCRIPT_NAME"]);
//Must end with /
$site_addr = rtrim($site_addr, "/\\") . "/";
// Can Be Set Manually Like "http://localhost/mysite/".
define("SITE_ADDR", $site_addr);
define("APP_ID", "052c03e1c3f4437b4cf37b048ed7d895");
// Application Default Color (Mostly Used By Mobile)
define("META_THEME_COLOR", "#000000");
//Application resource access status
define("AUTHORIZED", 200);
define("UNAUTHORIZED", 401);
define("NOROLE", 404);
define("FORBIDDEN", 403);
// Application Files and Directories
define("IMG_DIR",  "assets/images/");
define("FONTS_DIR",  "assets/fonts/");
define("SITE_FAVICON", IMG_DIR . "favicon.jpeg");
define("SITE_LOGO", IMG_DIR . "logo.jpeg");
define("CSS_DIR", SITE_ADDR . "assets/css/");
define("JS_DIR", SITE_ADDR . "assets/js/");
define("APP_DIR", "app/");
define("SYSTEM_DIR", "system/");
define("HELPERS_DIR", "helpers/");
define("LIBS_DIR", "libs/");
define("LANGS_DIR", "languages/");
define("MODELS_DIR", APP_DIR . "models/");
define("CONTROLLERS_DIR", APP_DIR . "controllers/");
define("VIEWS_DIR", APP_DIR . "views/");
define("LAYOUTS_DIR", VIEWS_DIR . "layouts/");
define("PAGES_DIR", VIEWS_DIR . "partials/");
define("AUDIT_LOGS_DIR", "logs/");
// File Upload Directories
define("UPLOAD_DIR", "uploads/");
define("UPLOAD_FILE_DIR", UPLOAD_DIR . "files/");
define("UPLOAD_IMG_DIR", UPLOAD_DIR . "photos/");
define("MAX_UPLOAD_FILESIZE", trim(ini_get("upload_max_filesize")));
// First page to see after user login
define("HOME_PAGE", "Home");
define("DEFAULT_PAGE", "index"); //Default Controller Class
define("DEFAULT_PAGE_ACTION", "index"); //Default Controller Action
define("DEFAULT_LAYOUT", LAYOUTS_DIR . "main_layout.php");
define("DEFAULT_LANGUAGE", "english"); //Default Language
// Page Meta Information
define("META_AUTHOR", "");
define("META_DESCRIPTION", "");
define("META_KEYWORDS", "");
define("META_VIEWPORT", "width=device-width, initial-scale=1.0");
define("PAGE_CHARSET", "UTF-8");
// Email Configuration Default Settings
define("USE_SMTP",false);
define("SMTP_USERNAME", "");
define("SMTP_PASSWORD", "");
define("SMTP_HOST", "");
define("SMTP_PORT", "");
//Default Email Sender Details. Please set this even if you are not using SMTP
define("DEFAULT_EMAIL", "");
define("DEFAULT_EMAIL_ACCOUNT_NAME", "");
// Database Configuration Settings
```

```
define("DB_HOST", "localhost");
define("DB_USERNAME", "root");
define("DB_PASSWORD", "");
define("DB_NAME", "lingkungan");
define("DB_TYPE", "mysql");
define("DB_PORT", "");
define("DB_CHARSET", "utf8");
define("MAX_RECORD_COUNT", 20); //Default Max Records to Retrieve  per Page
define("ORDER_TYPE",  "DESC");  //Default  Order Type
// Active User Profile Details
define('USER_ID',(isset($_SESSION[APP_ID.'user_data']) ? $_SESSION[APP_ID.'user_data']['id_user'] : null ));
define('USER_NAME',(isset($_SESSION[APP_ID.'user_data']) ? $_SESSION[APP_ID.'user_data']['username'] : null ));
define('USER_EMAIL',(isset($_SESSION[APP_ID.'user_data']) ? $_SESSION[APP_ID.'user_data']['email'] : null ));
define('USER_PHOTO',(isset($_SESSION[APP_ID.'user_data']) ? $_SESSION[APP_ID.'user_data']['foto'] : null ));
define('USER_ROLE',(isset($_SESSION[APP_ID.'user_data']) ? $_SESSION[APP_ID.'user_data']['roles'] : null ));
```

-arsip

```
<?php
/**
 * Arsip Page Controller
 * @category  Controller
 */
class ArsipController extends SecureController{
        function __construct(){
                parent::__construct();
                $this->tablename = "arsip";
        }
        /**
    * List page records
    * @param $fieldname (filter record by a field)
    * @param $fieldvalue (filter field value)
    * @return BaseView
    */
        function index($fieldname = null , $fieldvalue = null){
                $request = $this->request;
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $fields = array("arsip.id_arsip",
                        "arsip.id_dokumen",

        "dokumen.nomor_dokumen AS dokumen_nomor_dokumen",

        "arsip.klasifikasi_arsip",
                        "arsip.retensi_tahun",
                        "arsip.lokasi_fisik",

        "arsip.kondisi_dokumen",
                        "arsip.tanggal_arsip",

        "arsip.tanggal_musnah",
                        "arsip.status_arsip",
                        "arsip.created_at",
                        "arsip.updated_at");
                $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
                //search table record
                if(!empty($request->search)){
                        $text = trim($request->search);
                        $search_condition = "(

        arsip.id_arsip LIKE ? OR

        arsip.id_dokumen LIKE ? OR

        arsip.klasifikasi_arsip LIKE ? OR

        arsip.retensi_tahun LIKE ? OR

        arsip.lokasi_fisik LIKE ? OR

        arsip.kondisi_dokumen LIKE ? OR

        arsip.tanggal_arsip LIKE ? OR

        arsip.tanggal_musnah LIKE ? OR

        arsip.status_arsip LIKE ? OR
```

```
            arsip.created_at LIKE ? OR

            arsip.updated_at LIKE ?
                                    )";
                                    $search_params =
array(

            "%$text%","%$text%","%$text%","%$text
%","%$text%","%$text%","%$text%","%$text%","%$t
ext%","%$text%","%$text%"
                                    );
                                    //setting search
conditions
                                    $db-
>where($search_condition, $search_params);
                                     //template to use when
ajax search
                                    $this->view-
>search_template = "arsip/search.php";
                            }
                            $db->join("dokumen",
"arsip.id_dokumen = dokumen.id_dokumen",
"INNER");
                            if(!empty($request->orderby)){
                                    $orderby = $request-
>orderby;
                                    $ordertype =
(!empty($request->ordertype) ? $request->ordertype :
ORDER_TYPE);
                                    $db-
>orderBy($orderby, $ordertype);
                            }
                            else{
                                    $db-
>orderBy("arsip.id_arsip", ORDER_TYPE);
                            }
                            if($fieldname){
                                    $db-
>where($fieldname , $fieldvalue); //filter by a single
field name
                            }
                            $tc = $db->withTotalCount();
                            $records = $db->get($tablename,
$pagination, $fields);
                            $records_count =
count($records);
                            $total_records = intval($tc-
>totalCount);
                            $page_limit = $pagination[1];
                            $total_pages = ceil($total_records
/ $page_limit);
                            $data = new stdClass;
                            $data->records = $records;
                            $data->record_count =
$records_count;
                            $data->total_records =
$total_records;
                            $data->total_page = $total_pages;
                            if($db->getLastError()){
                                    $this-
>set_page_error();
                            }
                            $page_title = $this->view-
>page_title = "Arsip";
                            $this->view->report_filename =
date('Y-m-d') . '-' . $page_title;
                            $this->view->report_title =
$page_title;
                            $this->view->report_layout =
"report_layout.php";
                            $this->view->report_paper_size
= "A4";
                            $this->view->report_orientation
= "portrait";
                            $this-
>render_view("arsip/list.php", $data); //render the full
page
                    }
                    /**
     * View record detail
                     * @param $rec_id (select record by table
primary key)
     * @param $value value (select record by value of
field name(rec_id))
     * @return BaseView
     */
                    function view($rec_id = null, $value = null){
                            $request = $this->request;
                            $db = $this->GetModel();
```

```
                $rec_id = $this->rec_id = urldecode($rec_id);
                $tablename = $this->tablename;
                $fields = array("arsip.id_arsip",
                                "arsip.id_dokumen",
                                "dokumen.nomor_dokumen AS dokumen_nomor_dokumen",
                                "arsip.klasifikasi_arsip",
                                "arsip.retensi_tahun",
                                "arsip.lokasi_fisik",
                                "arsip.kondisi_dokumen",
                                "arsip.tanggal_arsip",
                                "arsip.tanggal_musnah",
                                "arsip.status_arsip",
                                "arsip.created_at",
                                "arsip.updated_at");
                if($value){
                        $db->where($rec_id, urldecode($value)); //select record based on field name
                }
                else{
                        $db->where("arsip.id_arsip", $rec_id);; //select record based on primary key
                }
                $db->join("dokumen", "arsip.id_dokumen = dokumen.id_dokumen", "INNER");
                $record = $db->getOne($tablename, $fields );
                if($record){
                        $page_title = $this->view->page_title = "View  Arsip";
                        $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                        $this->view->report_title = $page_title;
                        $this->view->report_layout = "report_layout.php";
                        $this->view->report_paper_size = "A4";
                        $this->view->report_orientation = "portrait";
                }
                else{
                        if($db->getLastError()){
                                $this->set_page_error();
                        }
                        else{
                                $this->set_page_error("No record found");
                        }
                }
                return $this->render_view("arsip/view.php", $record);
        }
        /**
     * Insert new record to the database table
         * @param $formdata array() from $_POST
     * @return BaseView
     */
        function add($formdata = null){
                if($formdata){
                        $db = $this->GetModel();
                        $tablename = $this->tablename;
                        $request = $this->request;
                        //fillable fields
                        $fields = $this->fields = array("id_dokumen","klasifikasi_arsip","retensi_tahun","lokasi_fisik","kondisi_dokumen","tanggal_arsip","tanggal_musnah","status_arsip");
                        $postdata = $this->format_request_data($formdata);
                        $this->rules_array = array(
                'id_dokumen' => 'required',
                'klasifikasi_arsip' => 'required',
                'retensi_tahun' => 'required',
```

```
            'lokasi_fisik' => 'required',

            'kondisi_dokumen' => 'required',

            'tanggal_arsip' => 'required',

            'tanggal_musnah' => 'required',

            'status_arsip' => 'required',
                                    );
                                    $this->sanitize_array
= array(

            'id_dokumen' => 'sanitize_string',

            'klasifikasi_arsip' => 'sanitize_string',

            'retensi_tahun' => 'sanitize_string',

            'lokasi_fisik' => 'sanitize_string',

            'kondisi_dokumen' => 'sanitize_string',

            'tanggal_arsip' => 'sanitize_string',

            'tanggal_musnah' => 'sanitize_string',

            'status_arsip' => 'sanitize_string',
                                    );
                                    $this->filter_vals     =
true; //set whether to remove empty fields
                                    $modeldata = $this-
>modeldata = $this->validate_form($postdata);
                                    if($this->validated()){
                                                $rec_id     =
$this->rec_id = $db->insert($tablename, $modeldata);

            if($rec_id){

            $this->set_flash_msg("Record           added
successfully", "success");

            return       $this->redirect("arsip");
                                                }
                                                else{
            $this->set_page_error();
                                                }
                                    }
                        }
                        $page_title     =     $this->view-
>page_title = "Add New Arsip";
                        $this-
>render_view("arsip/add.php");
            }
            /**
    * Update table record with formdata
             * @param $rec_id (select record by table
primary key)
             * @param $formdata array() from $_POST
    * @return array
    */
            function edit($rec_id = null, $formdata =
null){
                        $request = $this->request;
                        $db = $this->GetModel();
                        $this->rec_id = $rec_id;
                        $tablename = $this->tablename;
                         //editable fields
                        $fields     =     $this->fields     =
array("id_arsip","id_dokumen","klasifikasi_arsip","rete
nsi_tahun","lokasi_fisik","kondisi_dokumen","tanggal_
arsip","tanggal_musnah","status_arsip");
                        if($formdata){
                                    $postdata     =     $this-
>format_request_data($formdata);
                                    $this->rules_array     =
array(

            'id_dokumen' => 'required',

            'klasifikasi_arsip' => 'required',

            'retensi_tahun' => 'required',

            'lokasi_fisik' => 'required',

            'kondisi_dokumen' => 'required',

            'tanggal_arsip' => 'required',
```

```
            'tanggal_musnah' => 'required',
            'status_arsip' => 'required',
                                    );
                                    $this->sanitize_array
= array(
            'id_dokumen' => 'sanitize_string',
            'klasifikasi_arsip' => 'sanitize_string',
            'retensi_tahun' => 'sanitize_string',
            'lokasi_fisik' => 'sanitize_string',
            'kondisi_dokumen' => 'sanitize_string',
            'tanggal_arsip' => 'sanitize_string',
            'tanggal_musnah' => 'sanitize_string',
            'status_arsip' => 'sanitize_string',
                                    );
                                    $modeldata = $this-
>modeldata = $this->validate_form($postdata);
                                    if($this->validated()){
                                            $db-
>where("arsip.id_arsip", $rec_id);;
                                            $bool =
$db->update($tablename, $modeldata);
                                            $numRows
= $db->getRowCount(); //number of affected rows. 0 =
no record field updated
                                            if($bool
&& $numRows){
            $this->set_flash_msg("Record updated
successfully", "success");
            return $this->redirect("arsip");
                                            }
                                            else{
            if($db->getLastError()){
                    $this->set_page_error();
            }
            elseif(!$numRows){
            //not an error, but no record was updated
            $page_error = "No record updated";
            $this->set_page_error($page_error);
            $this->set_flash_msg($page_error,
"warning");
            return $this->redirect("arsip");
            }
                                                    }
                                            }
                                    }
                                    $db->where("arsip.id_arsip",
$rec_id);;
                                    $data = $db->getOne($tablename,
$fields);
                                    $page_title = $this->view-
>page_title = "Edit  Arsip";
                                    if(!$data){
                                            $this-
>set_page_error();
                                    }
                                    return $this-
>render_view("arsip/edit.php", $data);
            }
            /**
     * Update single field
             * @param $rec_id (select record by table
primary key)
             * @param $formdata array() from $_POST
     * @return array
     */
            function editfield($rec_id = null, $formdata =
null){
                    $db = $this->GetModel();
                    $this->rec_id = $rec_id;
```

```
$tablename = $this->tablename;
//editable fields
$fields = $this->fields = array("id_arsip","id_dokumen","klasifikasi_arsip","retensi_tahun","lokasi_fisik","kondisi_dokumen","tanggal_arsip","tanggal_musnah","status_arsip");
$page_error = null;
if($formdata){
$postdata = array();
$fieldname = $formdata['name'];
$fieldvalue = $formdata['value'];
$postdata[$fieldname] = $fieldvalue;
$postdata = $this->format_request_data($postdata);
$this->rules_array = array(
'id_dokumen' => 'required',
'klasifikasi_arsip' => 'required',
'retensi_tahun' => 'required',
'lokasi_fisik' => 'required',
'kondisi_dokumen' => 'required',
'tanggal_arsip' => 'required',
'tanggal_musnah' => 'required',
'status_arsip' => 'required',
);
$this->sanitize_array = array(
'id_dokumen' => 'sanitize_string',
'klasifikasi_arsip' => 'sanitize_string',
'retensi_tahun' => 'sanitize_string',
'lokasi_fisik' => 'sanitize_string',
'kondisi_dokumen' => 'sanitize_string',
'tanggal_arsip' => 'sanitize_string',
'tanggal_musnah' => 'sanitize_string',
'status_arsip' => 'sanitize_string',
);
$this->filter_rules = true; //filter validation rules by excluding fields not in the formdata
$modeldata = $this->modeldata = $this->validate_form($postdata);
if($this->validated()){
$db->where("arsip.id_arsip", $rec_id);;
$bool = $db->update($tablename, $modeldata);
$numRows = $db->getRowCount();
if($bool && $numRows){
return render_json(
array(
'num_rows' =>$numRows,
'rec_id' =>$rec_id,
)
);
}
else{
if($db->getLastError()){
$page_error = $db->getLastError();
}
elseif(!$numRows){
```

```
            $page_error = "No record updated";

            }

            render_error($page_error);
                                    }
                        }
                        else{

            render_error($this->view->page_error);
                        }
                }
                return null;
            }
            /**
     * Delete record from the database
             * Support multi delete by separating record id by comma.
     * @return BaseView
     */
            function delete($rec_id = null){
                Csrf::cross_check();
                $request = $this->request;
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $this->rec_id = $rec_id;
                //form multiple delete, split record id separated by comma into array
                $arr_rec_id = array_map('trim', explode(",", $rec_id));
                $db->where("arsip.id_arsip", $arr_rec_id, "in");
                $bool = $db->delete($tablename);
                if($bool){
                    $this->set_flash_msg("Record deleted successfully", "success");
                }
                elseif($db->getLastError()){
                    $page_error = $db->getLastError();
                    $this->set_flash_msg($page_error, "danger");
                }
                return $this->redirect("arsip");
            }
}
<?php
/**
 * Formulir Page Controller
 * @category  Controller
 */
class FormulirController extends SecureController{
            function __construct(){
                parent::__construct();
                $this->tablename = "formulir";
            }
            /**
     * List page records
     * @param $fieldname (filter record by a field)
     * @param $fieldvalue (filter field value)
     * @return BaseView
     */
            function index($fieldname = null , $fieldvalue = null){
                $request = $this->request;
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $fields = array("id_formulir",
                    "kode_formulir",
                    "tipe_formulir",
                    "file_formulir");
                $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
                //search table record
                if(!empty($request->search)){
                    $text = trim($request->search);
                    $search_condition = "(

            formulir.id_formulir LIKE ? OR

            formulir.kode_formulir LIKE ? OR

            formulir.tipe_formulir LIKE ? OR

            formulir.file_formulir LIKE ?
                    )";
```

```
                $search_params = array(
                    "%$text%","%$text%","%$text%","%$text%"
                );
                //setting search conditions
                $db->where($search_condition, $search_params);
                 //template to use when ajax search
                $this->view->search_template = "formulir/search.php";
            }
            if(!empty($request->orderby)){
                $orderby = $request->orderby;
                $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                $db->orderBy($orderby, $ordertype);
            }
            else{
                $db->orderBy("formulir.id_formulir", ORDER_TYPE);
            }
            if($fieldname){
                $db->where($fieldname , $fieldvalue); //filter by a single field name
            }
            $tc = $db->withTotalCount();
            $records = $db->get($tablename, $pagination, $fields);
            $records_count = count($records);
            $total_records = intval($tc->totalCount);
            $page_limit = $pagination[1];
            $total_pages = ceil($total_records / $page_limit);
            $data = new stdClass;
            $data->records = $records;
            $data->record_count = $records_count;
            $data->total_records = $total_records;
            $data->total_page = $total_pages;
            if($db->getLastError()){
                $this->set_page_error();
            }
            $page_title = $this->view->page_title = "Formulir";
            $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
            $this->view->report_title = $page_title;
            $this->view->report_layout = "report_layout.php";
            $this->view->report_paper_size = "A4";
            $this->view->report_orientation = "portrait";
            $this->render_view("formulir/list.php", $data); //render the full page
        }
        /**
     * View record detail
         * @param $rec_id (select record by table primary key)
     * @param $value value (select record by value of field name(rec_id))
     * @return BaseView
     */
        function view($rec_id = null, $value = null){
            $request = $this->request;
            $db = $this->GetModel();
            $rec_id = $this->rec_id = urldecode($rec_id);
            $tablename = $this->tablename;
            $fields = array("id_formulir",
                "kode_formulir",
                "tipe_formulir",
                "file_formulir");
            if($value){
                $db->where($rec_id, urldecode($value)); //select record based on field name
```

```
                }
                else{
                        $db->where("formulir.id_formulir", $rec_id);; //select record based on primary key
                }
                $record = $db->getOne($tablename, $fields );
                if($record){
                        $page_title = $this->view->page_title = "View  Formulir";
                        $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                        $this->view->report_title = $page_title;
                        $this->view->report_layout = "report_layout.php";
                        $this->view->report_paper_size = "A4";
                        $this->view->report_orientation = "portrait";
                }
                else{
                        if($db->getLastError()){
                                $this->set_page_error();
                        }
                        else{
                                $this->set_page_error("No record found");
                        }
                }
                return $this->render_view("formulir/view.php", $record);
        }
        /**
    * Insert new record to the database table
         * @param $formdata array() from $_POST
    * @return BaseView
    */
        function add($formdata = null){
                if($formdata){
                        $db = $this->GetModel();
                        $tablename = $this->tablename;
                        $request = $this->request;
                        //fillable fields
                        $fields = $this->fields = array("kode_formulir","tipe_formulir","file_formulir");
                        $postdata = $this->format_request_data($formdata);
                        $this->rules_array = array(
        'kode_formulir' => 'required',
        'tipe_formulir' => 'required',
        'file_formulir' => 'required',
                        );
                        $this->sanitize_array = array(
        'kode_formulir' => 'sanitize_string',
        'tipe_formulir' => 'sanitize_string',
        'file_formulir' => 'sanitize_string',
                        );
                        $this->filter_vals = true; //set whether to remove empty fields
                        $modeldata = $this->modeldata = $this->validate_form($postdata);
                        if($this->validated()){
                                $rec_id = $this->rec_id = $db->insert($tablename, $modeldata);
        if($rec_id){
        $this->set_flash_msg("Record added successfully", "success");
        return $this->redirect("formulir");
                                }
                                else{
        $this->set_page_error();
```

```
                                        }
                                }
                        }
                        $page_title = $this->view->page_title = "Add New Formulir";
                        $this->render_view("formulir/add.php");
                }
                /**
    * Update table record with formdata
                 * @param $rec_id (select record by table primary key)
                 * @param $formdata array() from $_POST
    * @return array
    */
                function edit($rec_id = null, $formdata = null){
                        $request = $this->request;
                        $db = $this->GetModel();
                        $this->rec_id = $rec_id;
                        $tablename = $this->tablename;
                         //editable fields
                        $fields = $this->fields = array("id_formulir","kode_formulir","tipe_formulir","file_formulir");
                        if($formdata){
                                $postdata = $this->format_request_data($formdata);
                                $this->rules_array = array(

                'kode_formulir' => 'required',

                'tipe_formulir' => 'required',

                'file_formulir' => 'required',
                                );
                                $this->sanitize_array = array(

                'kode_formulir' => 'sanitize_string',

                'tipe_formulir' => 'sanitize_string',

                'file_formulir' => 'sanitize_string',
                                );
                                $modeldata = $this->modeldata = $this->validate_form($postdata);
                                if($this->validated()){
                                        $db->where("formulir.id_formulir", $rec_id);;
                                        $bool = $db->update($tablename, $modeldata);
                                        $numRows = $db->getRowCount(); //number of affected rows. 0 = no record field updated
                                        if($bool && $numRows){

                $this->set_flash_msg("Record updated successfully", "success");

                return $this->redirect("formulir");
                                        }
                                        else{

                if($db->getLastError()){

                $this->set_page_error();

                }

                elseif(!$numRows){

                //not an error, but no record was updated

                $page_error = "No record updated";

                $this->set_page_error($page_error);

                $this->set_flash_msg($page_error, "warning");

                return $this->redirect("formulir");

                }
                                        }
                                }
                        }
                        $db->where("formulir.id_formulir", $rec_id);;
```

```
                $data = $db->getOne($tablename, $fields);
                $page_title = $this->view->page_title = "Edit  Formulir";
                if(!$data){
                        $this->set_page_error();
                }
                return $this->render_view("formulir/edit.php", $data);
        }
        /**
    * Update single field
         * @param $rec_id (select record by table primary key)
         * @param $formdata array() from $_POST
    * @return array
    */
        function editfield($rec_id = null, $formdata = null){
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                //editable fields
                $fields = $this->fields = array("id_formulir","kode_formulir","tipe_formulir","file_formulir");
                $page_error = null;
                if($formdata){
                        $postdata = array();
                        $fieldname = $formdata['name'];
                        $fieldvalue = $formdata['value'];
                        $postdata[$fieldname] = $fieldvalue;
                        $postdata = $this->format_request_data($postdata);
                        $this->rules_array = array(
        'kode_formulir' => 'required',
        'tipe_formulir' => 'required',
        'file_formulir' => 'required',
                        );
                        $this->sanitize_array = array(
        'kode_formulir' => 'sanitize_string',
        'tipe_formulir' => 'sanitize_string',
        'file_formulir' => 'sanitize_string',
                        );
                        $this->filter_rules = true; //filter validation rules by excluding fields not in the formdata
                        $modeldata = $this->modeldata = $this->validate_form($postdata);
                        if($this->validated()){
                                $db->where("formulir.id_formulir", $rec_id);;
                                $bool = $db->update($tablename, $modeldata);
                                $numRows = $db->getRowCount();
                                if($bool && $numRows){
        return render_json(
        array(
                'num_rows' =>$numRows,
                'rec_id' =>$rec_id,
        )
        );
                                }
                                else{
        if($db->getLastError()){
        $page_error = $db->getLastError();
        }
        elseif(!$numRows){
```

```
            $page_error = "No record updated";

            }

            render_error($page_error);
                                    }
                        }
                        else{

            render_error($this->view->page_error);
                        }
                }
                return null;
            }
            /**
     * Delete record from the database
             * Support multi delete by separating record
id by comma.
     * @return BaseView
     */
            function delete($rec_id = null){
                Csrf::cross_check();
                $request = $this->request;
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $this->rec_id = $rec_id;
                //form multiple delete, split record
id separated by comma into array
                $arr_rec_id = array_map('trim',
explode(",", $rec_id));
                $db-
>where("formulir.id_formulir", $arr_rec_id, "in");
                $bool = $db->delete($tablename);
                if($bool){
                    $this-
>set_flash_msg("Record deleted successfully",
"success");
                }
                elseif($db->getLastError()){
                    $page_error = $db-
>getLastError();
                    $this-
>set_flash_msg($page_error, "danger");
                }
                return $this-
>redirect("formulir");
            }
}
<?php

/**
 * Home Page Controller
 * @category  Controller
 */
class HomeController extends SecureController{
            /**
     * Index Action
     * @return View
     */
            function index(){

                $this-
>render_view("home/index.php" , null ,
"main_layout.php");


            }
}
-index
<?php
/**
 * Index Page Controller
 * @category  Controller
 */
class IndexController extends BaseController{
            function __construct(){
                parent::__construct();
                $this->tablename = "user";
            }
            /**
     * Index Action
     * @return null
     */
            function index(){
                if(user_login_status() == true){
                    $this-
>redirect(HOME_PAGE);
                }
                else{
                    $this-
>render_view("index/index.php");
```

```
                }
        }
        private function login_user($username , $password_text, $rememberme = false){
                $db = $this->GetModel();
                $username = filter_var($username, FILTER_SANITIZE_STRING);
                $db->where("username", $username)->orWhere("email", $username);
                $tablename = $this->tablename;
                $user = $db->getOne($tablename);
                if(!empty($user)){
                        //Verify User Password Text With DB Password Hash Value.
                        //Uses PHP password_verify() function with default options
                        $password_hash = $user['password'];
                        $this->modeldata['password'] = $password_hash; //update the modeldata with the password hash

        if(password_verify($password_text,$password_hash)){
                unset($user['password']); //Remove user password. No need to store it in the session

        set_session("user_data", $user); // Set active user data in a sessions
                                //if Remeber Me, Set Cookie

        if($rememberme == true){

        $sessionkey = time().random_str(20); // Generate a session key for the user

        //Update user session info in database with the session key

        $db->where("id_user", $user['id_user']);

        $res = $db->update($tablename, array("login_session_key" => hash_value($sessionkey)));

        if(!empty($res)){

        set_cookie("login_session_key", $sessionkey); // save user login_session_key in a Cookie

        }
                        }
                        else{

        clear_cookie("login_session_key");// Clear any previous set cookie
                        }

        $redirect_url = get_session("login_redirect_url");// Redirect to user active page

        if(!empty($redirect_url)){

        clear_session("login_redirect_url");

        return $this->redirect($redirect_url);
                        }
                        else{

        return $this->redirect(HOME_PAGE);
                        }
                }
                else{
                        //password is not correct
                        return $this->login_fail("Username or password not correct");
                }
        }
        else{
                //user is not registered
                return $this->login_fail("Username or password not correct");
        }
    }
    /**
```

```
    * Display login page with custom message when login fails
    * @return BaseView
    */
            private function login_fail($page_error = null){
                    $this->set_page_error($page_error);
                    $this->render_view("index/login.php");
            }
            /**
    * Login Action
    * If Not $_POST Request, Display Login Form View
    * @return View
    */
            function login($formdata = null){
                    if($formdata){
                            $modeldata = $this->modeldata = $formdata;
                            $username = trim($modeldata['username']);
                            $password = $modeldata['password'];
                            $rememberme = (!empty($modeldata['rememberme']) ? $modeldata['rememberme'] : false);
                            $this->login_user($username, $password, $rememberme);
                    }
                    else{
                            $this->set_page_error("Invalid request");
                            $this->render_view("index/login.php");
                    }
            }
            /**
    * Insert new record into the user table
             * @param $formdata array from $_POST
    * @return BaseView
    */
            function register($formdata = null){
                    if($formdata){
                            $request = $this->request;
                            $db = $this->GetModel();
                            $tablename = $this->tablename;
                            $fields = $this->fields = array("username","password","nama_pengguna","email","foto","roles"); //registration fields
                            $postdata = $this->format_request_data($formdata);
                            $cpassword = $postdata['confirm_password'];
                            $password = $postdata['password'];
                            if($cpassword != $password){
                                    $this->view->page_error[] = "Your password confirmation is not consistent";
                            }
                            $this->rules_array = array(
                                    'username' => 'required',
                                    'password' => 'required',

                'nama_pengguna' => 'required',
                                    'email' => 'required|valid_email',
                                    'foto' => 'required',
                                    'roles' => 'required',
                            );
                            $this->sanitize_array = array(
                                    'username' => 'sanitize_string',

                'nama_pengguna' => 'sanitize_string',
                                    'email' => 'sanitize_string',
                                    'foto' => 'sanitize_string',
```

```
                                        'roles' => 'sanitize_string',
                                    );
                                    $this->filter_vals = true; //set whether to remove empty fields
                                    $modeldata = $this->modeldata = $this->validate_form($postdata);
                                    $password_text = $modeldata['password'];
                                    //update modeldata with the password hash

                $modeldata['password'] = $this->modeldata['password'] = password_hash($password_text , PASSWORD_DEFAULT);
                                    //Check if Duplicate Record Already Exit In The Database
                                    $db->where("username", $modeldata['username']);
                                    if($db->has($tablename)){
                                        $this->view->page_error[] = $modeldata['username']." Already exist!";
                                    }
                                    //Check if Duplicate Record Already Exit In The Database
                                    $db->where("email", $modeldata['email']);
                                    if($db->has($tablename)){
                                        $this->view->page_error[] = $modeldata['email']." Already exist!";
                                    }
                                    if($this->validated()){
                                        $rec_id = $this->rec_id = $db->insert($tablename, $modeldata);

                    if($rec_id){

                    $this->login_user($modeldata['email'] , $password_text);

                    return;
                                    }
                                    else{
                    $this->set_page_error();
                                    }
                                }
                    }
                    $page_title = $this->view->page_title = "Add New User";
                    return $this->render_view("index/register.php");
                }
                /**
     * Logout Action
     * Destroy All Sessions And Cookies
     * @return View
     */
                function logout($arg=null){
                    Csrf::cross_check();
                    session_destroy();

                clear_cookie("login_session_key");
                    $this->redirect("");
                }
}
<?php
/**
 * Jenis_dokumen Page Controller
 * @category  Controller
 */
class Jenis_dokumenController extends SecureController{
                function __construct(){
                    parent::__construct();
                    $this->tablename = "jenis_dokumen";
                }
                /**
     * List page records
     * @param $fieldname (filter record by a field)
     * @param $fieldvalue (filter field value)
     * @return BaseView
     */
                function index($fieldname = null , $fieldvalue = null){
                    $request = $this->request;
```

```
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $fields = array("id_jenis",
                        "id_kategori",
                        "nama_jenis",
                        "template_dokumen",
                        "persyaratan",
                        "created_at",
                        "updated_at");
                $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
                //search table record
                if(!empty($request->search)){
                        $text = trim($request->search);
                        $search_condition = "(
        jenis_dokumen.id_jenis LIKE ? OR
        jenis_dokumen.id_kategori LIKE ? OR
        jenis_dokumen.nama_jenis LIKE ? OR
        jenis_dokumen.template_dokumen LIKE ? OR
        jenis_dokumen.persyaratan LIKE ? OR
        jenis_dokumen.created_at LIKE ? OR
        jenis_dokumen.updated_at LIKE ?
                        )";
                        $search_params = array(
        "%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%"
                        );
                        //setting search conditions
                        $db->where($search_condition, $search_params);
                         //template to use when ajax search
                        $this->view->search_template = "jenis_dokumen/search.php";
                }
                if(!empty($request->orderby)){
                        $orderby = $request->orderby;
                        $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                        $db->orderBy($orderby, $ordertype);
                }
                else{
                        $db->orderBy("jenis_dokumen.id_jenis", ORDER_TYPE);
                }
                if($fieldname){
                        $db->where($fieldname , $fieldvalue); //filter by a single field name
                }
                $tc = $db->withTotalCount();
                $records = $db->get($tablename, $pagination, $fields);
                $records_count = count($records);
                $total_records = intval($tc->totalCount);
                $page_limit = $pagination[1];
                $total_pages = ceil($total_records / $page_limit);
                $data = new stdClass;
                $data->records = $records;
                $data->record_count = $records_count;
                $data->total_records = $total_records;
                $data->total_page = $total_pages;
                if($db->getLastError()){
                        $this->set_page_error();
                }
                $page_title = $this->view->page_title = "Jenis Dokumen";
                $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
```

```
                $this->view->report_title = $page_title;
                $this->view->report_layout = "report_layout.php";
                $this->view->report_paper_size = "A4";
                $this->view->report_orientation = "portrait";
                $this->render_view("jenis_dokumen/list.php", $data); //render the full page
        }
        /**
    * View record detail
         * @param $rec_id (select record by table primary key)
    * @param $value value (select record by value of field name(rec_id))
    * @return BaseView
    */
        function view($rec_id = null, $value = null){
                $request = $this->request;
                $db = $this->GetModel();
                $rec_id = $this->rec_id = urldecode($rec_id);
                $tablename = $this->tablename;
                $fields = array("id_jenis",
                        "id_kategori",
                        "nama_jenis",
                        "template_dokumen",
                        "persyaratan",
                        "created_at",
                        "updated_at");
                if($value){
                        $db->where($rec_id, urldecode($value)); //select record based on field name
                }
                else{
                        $db->where("jenis_dokumen.id_jenis", $rec_id);; //select record based on primary key
                }
                $record = $db->getOne($tablename, $fields );
                if($record){
                        $page_title = $this->view->page_title = "View  Jenis Dokumen";
                        $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                        $this->view->report_title = $page_title;
                        $this->view->report_layout = "report_layout.php";
                        $this->view->report_paper_size = "A4";
                        $this->view->report_orientation = "portrait";
                }
                else{
                        if($db->getLastError()){
                                $this->set_page_error();
                        }
                        else{
                                $this->set_page_error("No record found");
                        }
                }
                return $this->render_view("jenis_dokumen/view.php", $record);
        }
        /**
    * Insert new record to the database table
         * @param $formdata array() from $_POST
    * @return BaseView
    */
        function add($formdata = null){
                if($formdata){
                        $db = $this->GetModel();
                        $tablename = $this->tablename;
                        $request = $this->request;
                        //fillable fields
                        $fields = $this->fields = array("id_kategori","nama_jenis","template_dokumen","persyaratan");
```

```
                $postdata = $this->format_request_data($formdata);
                $this->rules_array = array(

        'id_kategori' => 'required|numeric',

        'nama_jenis' => 'required',

        'template_dokumen' => 'required',

        'persyaratan' => 'required',
                );
                $this->sanitize_array = array(

        'id_kategori' => 'sanitize_string',

        'nama_jenis' => 'sanitize_string',

        'template_dokumen' => 'sanitize_string',

        'persyaratan' => 'sanitize_string',
                );
                $this->filter_vals = true; //set whether to remove empty fields
                $modeldata = $this->modeldata = $this->validate_form($postdata);
                if($this->validated()){
                        $rec_id = $this->rec_id = $db->insert($tablename, $modeldata);

        if($rec_id){

        $this->set_flash_msg("Record added successfully", "success");

        return $this->redirect("jenis_dokumen");
                        }
                        else{

        $this->set_page_error();
                        }
                }
        }
                $page_title = $this->view->page_title = "Add New Jenis Dokumen";
                $this->render_view("jenis_dokumen/add.php");
        }
        /**
    * Update table record with formdata
         * @param $rec_id (select record by table primary key)
         * @param $formdata array() from $_POST
    * @return array
    */
        function edit($rec_id = null, $formdata = null){
                $request = $this->request;
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                 //editable fields
                $fields = $this->fields = array("id_jenis","id_kategori","nama_jenis","template_dokumen","persyaratan");
                if($formdata){
                        $postdata = $this->format_request_data($formdata);
                        $this->rules_array = array(

        'id_kategori' => 'required|numeric',

        'nama_jenis' => 'required',

        'template_dokumen' => 'required',

        'persyaratan' => 'required',
                        );
                        $this->sanitize_array = array(

        'id_kategori' => 'sanitize_string',

        'nama_jenis' => 'sanitize_string',

        'template_dokumen' => 'sanitize_string',

        'persyaratan' => 'sanitize_string',
```

```
                    );
                    $modeldata = $this->modeldata = $this->validate_form($postdata);
                    if($this->validated()){
                        $db->where("jenis_dokumen.id_jenis", $rec_id);;
                        $bool = $db->update($tablename, $modeldata);
                        $numRows = $db->getRowCount(); //number of affected rows. 0 = no record field updated
                        if($bool && $numRows){
            $this->set_flash_msg("Record updated successfully", "success");
            return $this->redirect("jenis_dokumen");
                        }
                        else{
            if($db->getLastError()){
            $this->set_page_error();
            }
            elseif(!$numRows){
            //not an error, but no record was updated
            $page_error = "No record updated";
            $this->set_page_error($page_error);
            $this->set_flash_msg($page_error, "warning");
            return $this->redirect("jenis_dokumen");
            }
                        }
                    }
                }
                    $db->where("jenis_dokumen.id_jenis", $rec_id);;
                    $data = $db->getOne($tablename, $fields);
                    $page_title = $this->view->page_title = "Edit  Jenis Dokumen";
                    if(!$data){
                        $this->set_page_error();
                    }
                    return $this->render_view("jenis_dokumen/edit.php", $data);
                }
                /**
     * Update single field
                 * @param $rec_id (select record by table primary key)
                 * @param $formdata array() from $_POST
     * @return array
     */
                function editfield($rec_id = null, $formdata = null){
                    $db = $this->GetModel();
                    $this->rec_id = $rec_id;
                    $tablename = $this->tablename;
                    //editable fields
                    $fields = $this->fields = array("id_jenis","id_kategori","nama_jenis","template_dokumen","persyaratan");
                    $page_error = null;
                    if($formdata){
                        $postdata = array();
                        $fieldname = $formdata['name'];
                        $fieldvalue = $formdata['value'];
                        $postdata[$fieldname] = $fieldvalue;
                        $postdata = $this->format_request_data($postdata);
                        $this->rules_array = array(
            'id_kategori' => 'required|numeric',
            'nama_jenis' => 'required',
```

```
            'template_dokumen' => 'required',

            'persyaratan' => 'required',
                                    );
                                    $this->sanitize_array
= array(

            'id_kategori' => 'sanitize_string',

            'nama_jenis' => 'sanitize_string',

            'template_dokumen' => 'sanitize_string',

            'persyaratan' => 'sanitize_string',
                                    );
                                    $this->filter_rules =
true; //filter validation rules by excluding fields not in the
formdata
                                    $modeldata = $this-
>modeldata = $this->validate_form($postdata);
                                    if($this->validated()){
                                            $db-
>where("jenis_dokumen.id_jenis", $rec_id);;
                                            $bool =
$db->update($tablename, $modeldata);
                                            $numRows
= $db->getRowCount();
                                            if($bool
&& $numRows){

            return render_json(

            array(

                    'num_rows' =>$numRows,

                    'rec_id' =>$rec_id,

            )

            );
                                            }
                                            else{

            if($db->getLastError()){
                                                    $page_error = $db->getLastError();

                                            }

                                            elseif(!$numRows){

                                            $page_error = "No record updated";

                                            }

                                            render_error($page_error);
                                                    }
                                            }
                                            else{

            render_error($this->view->page_error);
                                            }
                                    }
                                    return null;
                            }
                            /**
     * Delete record from the database
                             * Support multi delete by separating record
id by comma.
     * @return BaseView
     */
                            function delete($rec_id = null){
                                    Csrf::cross_check();
                                    $request = $this->request;
                                    $db = $this->GetModel();
                                    $tablename = $this->tablename;
                                    $this->rec_id = $rec_id;
                                    //form multiple delete, split record
id separated by comma into array
                                    $arr_rec_id = array_map('trim',
explode(",", $rec_id));
                                    $db-
>where("jenis_dokumen.id_jenis", $arr_rec_id, "in");
                                    $bool = $db->delete($tablename);
                                    if($bool){
                                            $this-
>set_flash_msg("Record deleted successfully",
"success");
                                    }
                                    elseif($db->getLastError()){
```

```
                $page_error = $db->getLastError();
                $this->set_flash_msg($page_error, "danger");
            }
            return $this->redirect("jenis_dokumen");
        }
}
<?php
/**
 * Kategori_dokumen Page Controller
 * @category  Controller
 */
class Kategori_dokumenController extends SecureController{
        function __construct(){
            parent::__construct();
            $this->tablename = "kategori_dokumen";
        }
        /**
    * List page records
    * @param $fieldname (filter record by a field)
    * @param $fieldvalue (filter field value)
    * @return BaseView
    */
        function index($fieldname = null , $fieldvalue = null){
            $request = $this->request;
            $db = $this->GetModel();
            $tablename = $this->tablename;
            $fields = array("id_kategori",
                "nama_kategori",
                "deskripsi",
                "created_at",
                "updated_at",
                "kode_kategori");
            $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
            //search table record
            if(!empty($request->search)){
                $text = trim($request->search);
                $search_condition = "(
                    kategori_dokumen.id_kategori LIKE ? OR
                    kategori_dokumen.nama_kategori LIKE ? OR
                    kategori_dokumen.deskripsi LIKE ? OR
                    kategori_dokumen.created_at LIKE ? OR
                    kategori_dokumen.updated_at LIKE ? OR
                    kategori_dokumen.kode_kategori LIKE ?
                )";
                $search_params = array(
                "%$text%","%$text%","%$text%","%$text%","%$text%","%$text%"
                );
                //setting search conditions
                $db->where($search_condition, $search_params);
                 //template to use when ajax search
                $this->view->search_template = "kategori_dokumen/search.php";
            }
            if(!empty($request->orderby)){
                $orderby = $request->orderby;
                $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                $db->orderBy($orderby, $ordertype);
            }
            else{
                $db->orderBy("kategori_dokumen.id_kategori", ORDER_TYPE);
            }
            if($fieldname){
```

```
                $db->where($fieldname , $fieldvalue); //filter by a single field name
                }
                $tc = $db->withTotalCount();
                $records = $db->get($tablename, $pagination, $fields);
                $records_count = count($records);
                $total_records = intval($tc->totalCount);
                $page_limit = $pagination[1];
                $total_pages = ceil($total_records / $page_limit);
                $data = new stdClass;
                $data->records = $records;
                $data->record_count = $records_count;
                $data->total_records = $total_records;
                $data->total_page = $total_pages;
                if($db->getLastError()){
                        $this->set_page_error();
                }
                $page_title = $this->view->page_title = "Kategori Dokumen";
                $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                $this->view->report_title = $page_title;
                $this->view->report_layout = "report_layout.php";
                $this->view->report_paper_size = "A4";
                $this->view->report_orientation = "portrait";
                $this->render_view("kategori_dokumen/list.php", $data); //render the full page
        }
        /**
     * View record detail
         * @param $rec_id (select record by table primary key)
     * @param $value value (select record by value of field name(rec_id))
     * @return BaseView
     */
        function view($rec_id = null, $value = null){
                $request = $this->request;
                $db = $this->GetModel();
                $rec_id = $this->rec_id = urldecode($rec_id);
                $tablename = $this->tablename;
                $fields = array("id_kategori",
                        "nama_kategori",
                        "deskripsi",
                        "created_at",
                        "updated_at",
                        "kode_kategori");
                if($value){
                        $db->where($rec_id, urldecode($value)); //select record based on field name
                }
                else{
                        $db->where("kategori_dokumen.id_kategori", $rec_id);; //select record based on primary key
                }
                $record = $db->getOne($tablename, $fields );
                if($record){
                        $page_title = $this->view->page_title = "View  Kategori Dokumen";
                        $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                        $this->view->report_title = $page_title;
                        $this->view->report_layout = "report_layout.php";
                        $this->view->report_paper_size = "A4";
                        $this->view->report_orientation = "portrait";
                }
                else{
                        if($db->getLastError()){
                                $this->set_page_error();
```

```
                    }
                    else{
                        $this->set_page_error("No record found");
                    }
                }
                return $this->render_view("kategori_dokumen/view.php", $record);
            }
            /**
     * Insert new record to the database table
             * @param $formdata array() from $_POST
     * @return BaseView
     */
            function add($formdata = null){
                if($formdata){
                    $db = $this->GetModel();
                    $tablename = $this->tablename;
                    $request = $this->request;
                    //fillable fields
                    $fields = $this->fields = array("nama_kategori","deskripsi","kode_kategori");
                    $postdata = $this->format_request_data($formdata);
                    $this->rules_array = array(

            'nama_kategori' => 'required',
                        'deskripsi' => 'required',

            'kode_kategori' => 'required',
                    );
                    $this->sanitize_array = array(

            'nama_kategori' => 'sanitize_string',
                        'deskripsi' => 'sanitize_string',

            'kode_kategori' => 'sanitize_string',
                    );
                    $this->filter_vals = true; //set whether to remove empty fields
                    $modeldata = $this->modeldata = $this->validate_form($postdata);
                    if($this->validated()){
                        $rec_id = $this->rec_id = $db->insert($tablename, $modeldata);

            if($rec_id){

            $this->set_flash_msg("Record added successfully", "success");

            return $this->redirect("kategori_dokumen");
                        }
                        else{

            $this->set_page_error();
                        }
                    }
                }
                $page_title = $this->view->page_title = "Add New Kategori Dokumen";
                $this->render_view("kategori_dokumen/add.php");
            }
            /**
     * Update table record with formdata
             * @param $rec_id (select record by table primary key)
             * @param $formdata array() from $_POST
     * @return array
     */
            function edit($rec_id = null, $formdata = null){
                $request = $this->request;
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                 //editable fields
                $fields = $this->fields = array("id_kategori","nama_kategori","deskripsi","kode_kategori");
                if($formdata){
```

```
                $postdata = $this->format_request_data($formdata);
                $this->rules_array = array(
        'nama_kategori' => 'required',
                'deskripsi' => 'required',
        'kode_kategori' => 'required',
                );
                $this->sanitize_array = array(
        'nama_kategori' => 'sanitize_string',
                'deskripsi' => 'sanitize_string',
        'kode_kategori' => 'sanitize_string',
                );
                $modeldata = $this->modeldata = $this->validate_form($postdata);
                if($this->validated()){
                $db->where("kategori_dokumen.id_kategori", $rec_id);;
                $bool = $db->update($tablename, $modeldata);
                $numRows = $db->getRowCount(); //number of affected rows. 0 = no record field updated
                if($bool && $numRows){
        $this->set_flash_msg("Record updated successfully", "success");
        return $this->redirect("kategori_dokumen");
                }
                else{
        if($db->getLastError()){
        $this->set_page_error();
        }
        elseif(!$numRows){
        //not an error, but no record was updated
        $page_error = "No record updated";
        $this->set_page_error($page_error);
        $this->set_flash_msg($page_error, "warning");
        return $this->redirect("kategori_dokumen");
        }
                }
                }
        }
                $db->where("kategori_dokumen.id_kategori", $rec_id);;
                $data = $db->getOne($tablename, $fields);
                $page_title = $this->view->page_title = "Edit  Kategori Dokumen";
                if(!$data){
                $this->set_page_error();
                }
                return $this->render_view("kategori_dokumen/edit.php", $data);
        }
        /**
    * Update single field
         * @param $rec_id (select record by table primary key)
         * @param $formdata array() from $_POST
    * @return array
    */
        function editfield($rec_id = null, $formdata = null){
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                //editable fields
```

```
                $fields = $this->fields = array("id_kategori","nama_kategori","deskripsi","kode_kategori");
                $page_error = null;
                if($formdata){
                    $postdata = array();
                    $fieldname = $formdata['name'];
                    $fieldvalue = $formdata['value'];
                    $postdata[$fieldname] = $fieldvalue;
                    $postdata = $this->format_request_data($postdata);
                    $this->rules_array = array(

        'nama_kategori' => 'required',
                                'deskripsi' => 'required',

        'kode_kategori' => 'required',
                    );
                    $this->sanitize_array = array(

        'nama_kategori' => 'sanitize_string',
                                'deskripsi' => 'sanitize_string',

        'kode_kategori' => 'sanitize_string',
                    );
                    $this->filter_rules = true; //filter validation rules by excluding fields not in the formdata
                    $modeldata = $this->modeldata = $this->validate_form($postdata);
                    if($this->validated()){
                        $db->where("kategori_dokumen.id_kategori", $rec_id);;
                        $bool = $db->update($tablename, $modeldata);
                        $numRows = $db->getRowCount();
                        if($bool && $numRows){
                            return render_json(

                            array(

                                'num_rows' =>$numRows,

                                'rec_id' =>$rec_id,

                            )

                            );
                        }
                        else{

                            if($db->getLastError()){

                            $page_error = $db->getLastError();

                            }

                            elseif(!$numRows){

                            $page_error = "No record updated";

                            }

                            render_error($page_error);
                        }
                    }
                    else{

                        render_error($this->view->page_error);
                    }
                }
                return null;
            }
            /**
             * Delete record from the database
             * Support multi delete by separating record id by comma.
             * @return BaseView
             */
            function delete($rec_id = null){
                Csrf::cross_check();
                $request = $this->request;
```

```
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $this->rec_id = $rec_id;
                //form multiple delete, split record id separated by comma into array
                $arr_rec_id = array_map('trim', explode(",", $rec_id));
                $db->where("kategori_dokumen.id_kategori", $arr_rec_id, "in");
                $bool = $db->delete($tablename);
                if($bool){
                        $this->set_flash_msg("Record deleted successfully", "success");
                }
                elseif($db->getLastError()){
                        $page_error = $db->getLastError();
                        $this->set_flash_msg($page_error, "danger");
                }
                return $this->redirect("kategori_dokumen");
        }
}
```

-komentar

```
<?php
/**
 * Komentar Page Controller
 * @category  Controller
 */
class KomentarController extends SecureController{
        function __construct(){
                parent::__construct();
                $this->tablename = "komentar";
        }
        /**
    * List page records
    * @param $fieldname (filter record by a field)
    * @param $fieldvalue (filter field value)
    * @return BaseView
    */
        function index($fieldname = null , $fieldvalue = null){
                $request = $this->request;
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $fields = array("id_komentar",
                        "id_user",
                        "isi_komentar",
                        "lampiran_komentar",
                        "tanggal_komentar",
                        "created_at",
                        "updated_at");
                $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
                //search table record
                if(!empty($request->search)){
                        $text = trim($request->search);
                        $search_condition = "(
        komentar.id_komentar LIKE ? OR
        komentar.id_user LIKE ? OR
        komentar.isi_komentar LIKE ? OR
        komentar.lampiran_komentar LIKE ? OR
        komentar.tanggal_komentar LIKE ? OR
        komentar.created_at LIKE ? OR
        komentar.updated_at LIKE ?
                        )";
                        $search_params = array(
        "%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%"
                        );
                        //setting search conditions
                        $db->where($search_condition, $search_params);
                         //template to use when ajax search
                        $this->view->search_template = "komentar/search.php";
```

```
                }
                if(!empty($request->orderby)){
                        $orderby = $request->orderby;
                        $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                        $db->orderBy($orderby, $ordertype);
                }
                else{
                        $db->orderBy("komentar.id_komentar", ORDER_TYPE);
                }
                if($fieldname){
                        $db->where($fieldname , $fieldvalue); //filter by a single field name
                }
                $tc = $db->withTotalCount();
                $records = $db->get($tablename, $pagination, $fields);
                $records_count = count($records);
                $total_records = intval($tc->totalCount);
                $page_limit = $pagination[1];
                $total_pages = ceil($total_records / $page_limit);
                $data = new stdClass;
                $data->records = $records;
                $data->record_count = $records_count;
                $data->total_records = $total_records;
                $data->total_page = $total_pages;
                if($db->getLastError()){
                        $this->set_page_error();
                }
                $page_title = $this->view->page_title = "Komentar";
                $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                $this->view->report_title = $page_title;
                $this->view->report_layout = "report_layout.php";
                $this->view->report_paper_size = "A4";
                $this->view->report_orientation = "portrait";
                $this->render_view("komentar/list.php", $data); //render the full page
        }
        /**
    * View record detail
         * @param $rec_id (select record by table primary key)
    * @param $value value (select record by value of field name(rec_id))
    * @return BaseView
    */
        function view($rec_id = null, $value = null){
                $request = $this->request;
                $db = $this->GetModel();
                $rec_id = $this->rec_id = urldecode($rec_id);
                $tablename = $this->tablename;
                $fields = array("komentar.id_komentar",
                                "komentar.id_user",
                                "user.username AS user_username",
        "komentar.isi_komentar",
        "komentar.lampiran_komentar",
        "komentar.tanggal_komentar",
        "komentar.created_at",
        "komentar.updated_at");
                if($value){
                        $db->where($rec_id, urldecode($value)); //select record based on field name
                }
                else{
```

```
                $db->where("komentar.id_komentar", $rec_id);; //select record based on primary key
                }
                $db->join("user", "komentar.id_user = user.id_user", "INNER");
                $record = $db->getOne($tablename, $fields );
                if($record){
                $page_title = $this->view->page_title = "View  Komentar";
                $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                $this->view->report_title = $page_title;
                $this->view->report_layout = "report_layout.php";
                $this->view->report_paper_size = "A4";
                $this->view->report_orientation = "portrait";
                }
                else{
                if($db->getLastError()){
                $this->set_page_error();
                }
                else{
                $this->set_page_error("No record found");
                }
                }
                return $this->render_view("komentar/view.php", $record);
        }
        /**
    * Insert new record to the database table
         * @param $formdata array() from $_POST
    * @return BaseView
    */
        function add($formdata = null){
                if($formdata){
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $request = $this->request;
                //fillable fields
                $fields = $this->fields = array("id_user","isi_komentar","lampiran_komentar");
                $postdata = $this->format_request_data($formdata);
                $this->rules_array = array(
                'id_user' => 'required',

        'isi_komentar' => 'required',

        'lampiran_komentar' => 'required',
                );
                $this->sanitize_array = array(
                'id_user' => 'sanitize_string',

        'isi_komentar' => 'sanitize_string',

        'lampiran_komentar' => 'sanitize_string',
                );
                $this->filter_vals = true; //set whether to remove empty fields
                $modeldata = $this->modeldata = $this->validate_form($postdata);
                if($this->validated()){
                $rec_id = $this->rec_id = $db->insert($tablename, $modeldata);

        if($rec_id){

        $this->set_flash_msg("Record added successfully", "success");

        return $this->redirect("komentar");
                }
                else{

        $this->set_page_error();
```

```
                                        }
                                }
                        }
                        $page_title = $this->view->page_title = "Add New Komentar";
                        $this->render_view("komentar/add.php");
                }
                /**
    * Update table record with formdata
                 * @param $rec_id (select record by table primary key)
                 * @param $formdata array() from $_POST
    * @return array
    */
                function edit($rec_id = null, $formdata = null){
                        $request = $this->request;
                        $db = $this->GetModel();
                        $this->rec_id = $rec_id;
                        $tablename = $this->tablename;
                         //editable fields
                        $fields = $this->fields = array("id_komentar","id_user","isi_komentar","lampiran_komentar");
                        if($formdata){
                                $postdata = $this->format_request_data($formdata);
                                $this->rules_array = array(
                                                'id_user' => 'required',

                'isi_komentar' => 'required',

                'lampiran_komentar' => 'required',
                                );
                                $this->sanitize_array = array(
                                                'id_user' => 'sanitize_string',

                'isi_komentar' => 'sanitize_string',

                'lampiran_komentar' => 'sanitize_string',
                                );
                                $modeldata = $this->modeldata = $this->validate_form($postdata);
                                if($this->validated()){
                                        $db->where("komentar.id_komentar", $rec_id);;
                                        $bool = $db->update($tablename, $modeldata);
                                        $numRows = $db->getRowCount(); //number of affected rows. 0 = no record field updated
                                        if($bool && $numRows){

                $this->set_flash_msg("Record updated successfully", "success");

                return $this->redirect("komentar");
                                        }
                                        else{

                if($db->getLastError()){

                $this->set_page_error();

                }

                elseif(!$numRows){

                //not an error, but no record was updated

                $page_error = "No record updated";

                $this->set_page_error($page_error);

                $this->set_flash_msg($page_error, "warning");

                return $this->redirect("komentar");

                }
                                        }
                                }
                        }
                        $db->where("komentar.id_komentar", $rec_id);;
```

```
                $data = $db->getOne($tablename, $fields);
                $page_title = $this->view->page_title = "Edit  Komentar";
                if(!$data){
                        $this->set_page_error();
                }
                return $this->render_view("komentar/edit.php", $data);
        }
        /**
    * Update single field
         * @param $rec_id (select record by table primary key)
         * @param $formdata array() from $_POST
    * @return array
    */
        function editfield($rec_id = null, $formdata = null){
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                //editable fields
                $fields = $this->fields = array("id_komentar","id_user","isi_komentar","lampiran_komentar");
                $page_error = null;
                if($formdata){
                        $postdata = array();
                        $fieldname = $formdata['name'];
                        $fieldvalue = $formdata['value'];
                        $postdata[$fieldname] = $fieldvalue;
                        $postdata = $this->format_request_data($postdata);
                        $this->rules_array = array(
                                        'id_user' => 'required',

        'isi_komentar' => 'required',

        'lampiran_komentar' => 'required',
                        );
                        $this->sanitize_array = array(
                                        'id_user' => 'sanitize_string',

        'isi_komentar' => 'sanitize_string',

        'lampiran_komentar' => 'sanitize_string',
                        );
                        $this->filter_rules = true; //filter validation rules by excluding fields not in the formdata
                        $modeldata = $this->modeldata = $this->validate_form($postdata);
                        if($this->validated()){
                                $db->where("komentar.id_komentar", $rec_id);;
                                $bool = $db->update($tablename, $modeldata);
                                $numRows = $db->getRowCount();
                                if($bool && $numRows){

        return render_json(

        array(

                'num_rows' =>$numRows,

                'rec_id' =>$rec_id,

        )

        );
                                }
                                else{

        if($db->getLastError()){

        $page_error = $db->getLastError();

        }

        elseif(!$numRows){
```

```
            $page_error = "No record updated";

            }

            render_error($page_error);
                                    }
                        }
                        else{

            render_error($this->view->page_error);
                        }
                }
                return null;
            }
            /**
     * Delete record from the database
             * Support multi delete by separating record id by comma.
     * @return BaseView
     */
            function delete($rec_id = null){
                        Csrf::cross_check();
                        $request = $this->request;
                        $db = $this->GetModel();
                        $tablename = $this->tablename;
                        $this->rec_id = $rec_id;
                        //form multiple delete, split record id separated by comma into array
                        $arr_rec_id = array_map('trim', explode(",", $rec_id));
                        $db->where("komentar.id_komentar", $arr_rec_id, "in");
                        $bool = $db->delete($tablename);
                        if($bool){
                                    $this->set_flash_msg("Record deleted successfully", "success");
                        }
                        elseif($db->getLastError()){
                                    $page_error = $db->getLastError();
                                    $this->set_flash_msg($page_error, "danger");
                        }
                                    return $this->redirect("komentar");
                        }
}
```

-notifikasi

```
<?php
/**
 * Notifikasi Page Controller
 * @category  Controller
 */
class NotifikasiController extends SecureController{
            function __construct(){
                        parent::__construct();
                        $this->tablename = "notifikasi";
            }
            /**
     * List page records
     * @param $fieldname (filter record by a field)
     * @param $fieldvalue (filter field value)
     * @return BaseView
     */
            function index($fieldname = null , $fieldvalue = null){
                        $request = $this->request;
                        $db = $this->GetModel();
                        $tablename = $this->tablename;
                        $fields = array("id_notifikasi",
                                    "id_user",
                                    "id_dokumen",
                                    "jenis_notifikasi",
                                    "judul_notifikasi",
                                    "pesan",
                                    "status_baca",
                                    "tanggal_notifikasi",
                                    "updated_at");
                        $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
                        //search table record
                        if(!empty($request->search)){
                                    $text = trim($request->search);
                                    $search_condition = "(

            notifikasi.id_notifikasi LIKE ? OR
```

```
            notifikasi.id_user LIKE ? OR
            notifikasi.id_dokumen LIKE ? OR
            notifikasi.jenis_notifikasi LIKE ? OR
            notifikasi.judul_notifikasi LIKE ? OR
            notifikasi.pesan LIKE ? OR
            notifikasi.status_baca LIKE ? OR
            notifikasi.tanggal_notifikasi LIKE ? OR
            notifikasi.updated_at LIKE ?
                        )";
                        $search_params = array(
            "%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%"
                        );
                        //setting search conditions
                        $db->where($search_condition, $search_params);
                         //template to use when ajax search
                        $this->view->search_template = "notifikasi/search.php";
                    }
                    if(!empty($request->orderby)){
                        $orderby = $request->orderby;
                        $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                        $db->orderBy($orderby, $ordertype);
                    }
                    else{
                        $db->orderBy("notifikasi.id_notifikasi", ORDER_TYPE);
                    }
                    $allowed_roles = array ('administrator');

                if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
                    $db->where("notifikasi.id_user", get_active_user('id_user') );
                    }
                    if($fieldname){
                        $db->where($fieldname , $fieldvalue); //filter by a single field name
                    }
                    $tc = $db->withTotalCount();
                    $records = $db->get($tablename, $pagination, $fields);
                    $records_count = count($records);
                    $total_records = intval($tc->totalCount);
                    $page_limit = $pagination[1];
                    $total_pages = ceil($total_records / $page_limit);
                    $data = new stdClass;
                    $data->records = $records;
                    $data->record_count = $records_count;
                    $data->total_records = $total_records;
                    $data->total_page = $total_pages;
                    if($db->getLastError()){
                        $this->set_page_error();
                    }
                    $page_title = $this->view->page_title = "Notifikasi";
                    $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                    $this->view->report_title = $page_title;
                    $this->view->report_layout = "report_layout.php";
                    $this->view->report_paper_size = "A4";
                    $this->view->report_orientation = "portrait";
```

```
                $this->render_view("notifikasi/list.php", $data); //render the full page
            }
            /**
    * View record detail
             * @param $rec_id (select record by table primary key)
    * @param $value value (select record by value of field name(rec_id))
    * @return BaseView
    */
            function view($rec_id = null, $value = null){
                $request = $this->request;
                $db = $this->GetModel();
                $rec_id = $this->rec_id = urldecode($rec_id);
                $tablename = $this->tablename;
                $fields = array("id_notifikasi",
                        "id_user",
                        "id_dokumen",
                        "jenis_notifikasi",
                        "judul_notifikasi",
                        "pesan",
                        "status_baca",
                        "tanggal_notifikasi",
                        "updated_at");
                $allowed_roles = array ('administrator');

            if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
                $db->where("notifikasi.id_user", get_active_user('id_user') );
                }
                if($value){
                    $db->where($rec_id, urldecode($value)); //select record based on field name
                }
                else{
                    $db->where("notifikasi.id_notifikasi", $rec_id);; //select record based on primary key
                }
                $record = $db->getOne($tablename, $fields );
                if($record){
                    $page_title = $this->view->page_title = "View  Notifikasi";
                    $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                    $this->view->report_title = $page_title;
                    $this->view->report_layout = "report_layout.php";
                    $this->view->report_paper_size = "A4";
                    $this->view->report_orientation = "portrait";
                }
                else{
                    if($db->getLastError()){
                        $this->set_page_error();
                    }
                    else{
                        $this->set_page_error("No record found");
                    }
                }
                return $this->render_view("notifikasi/view.php", $record);
            }
            /**
    * Insert new record to the database table
             * @param $formdata array() from $_POST
    * @return BaseView
    */
            function add($formdata = null){
                if($formdata){
                    $db = $this->GetModel();
                    $tablename = $this->tablename;
                    $request = $this->request;
                    //fillable fields
                    $fields = $this->fields = array("id_user","id_dokumen","jenis_notifikasi","judul
```

```
_notifikasi","pesan","status_baca","tanggal_notifikasi")
;
                $postdata = $this->format_request_data($formdata);
                $this->rules_array = array(
        'id_dokumen' => 'required',
        'jenis_notifikasi' => 'required',
        'judul_notifikasi' => 'required',
                'pesan' => 'required',
        'status_baca' => 'required',
        'tanggal_notifikasi' => 'required',
                );
                $this->sanitize_array = array(
        'id_dokumen' => 'sanitize_string',
        'jenis_notifikasi' => 'sanitize_string',
        'judul_notifikasi' => 'sanitize_string',
                'pesan' => 'sanitize_string',
        'status_baca' => 'sanitize_string',
        'tanggal_notifikasi' => 'sanitize_string',
                );
                $this->filter_vals = true; //set whether to remove empty fields
                $modeldata = $this->modeldata = $this->validate_form($postdata);
                $modeldata['id_user'] = USER_ID;
                if($this->validated()){
                        $rec_id = $this->rec_id = $db->insert($tablename, $modeldata);
        if($rec_id){
                $this->set_flash_msg("Record added successfully", "success");
        return $this->redirect("notifikasi");
                }
                else{
        $this->set_page_error();
                }
                }
                $page_title = $this->view->page_title = "Add New Notifikasi";
                $this->render_view("notifikasi/add.php");
        }
        /**
     * Update table record with formdata
         * @param $rec_id (select record by table primary key)
         * @param $formdata array() from $_POST
     * @return array
     */
        function edit($rec_id = null, $formdata = null){
                $request = $this->request;
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                 //editable fields
                $fields = $this->fields = array("id_notifikasi","id_user","id_dokumen","jenis_notifikasi","judul_notifikasi","pesan","status_baca","tanggal_notifikasi");
                if($formdata){
                        $postdata = $this->format_request_data($formdata);
                        $this->rules_array = array(
        'id_dokumen' => 'required',
        'jenis_notifikasi' => 'required',
        'judul_notifikasi' => 'required',
```

```
                                        'pesan'   =>
'required',
                'status_baca' => 'required',
                'tanggal_notifikasi' => 'required',
                                        );
                                        $this->sanitize_array
= array(
                'id_dokumen' => 'sanitize_string',
                'jenis_notifikasi' => 'sanitize_string',
                'judul_notifikasi' => 'sanitize_string',
                                        'pesan'   =>
'sanitize_string',
                'status_baca' => 'sanitize_string',
                'tanggal_notifikasi' => 'sanitize_string',
                                        );
                                        $modeldata = $this-
>modeldata = $this->validate_form($postdata);
                                        $modeldata['id_user']
= USER_ID;
                                        if($this->validated()){
                        $allowed_roles = array
('administrator');
                if(!in_array(strtolower(USER_ROLE),
$allowed_roles)){
                        $db->where("notifikasi.id_user",
get_active_user('id_user') );
                        }
                                        $db-
>where("notifikasi.id_notifikasi", $rec_id);;
                                        $bool =
$db->update($tablename, $modeldata);
                                        $numRows
= $db->getRowCount(); //number of affected rows. 0 =
no record field updated
                                        if($bool
&& $numRows){
                $this->set_flash_msg("Record updated
successfully", "success");
                return $this->redirect("notifikasi");
                                        }
                                        else{
                if($db->getLastError()){
                $this->set_page_error();
                }
                elseif(!$numRows){
                //not an error, but no record was updated
                $page_error = "No record updated";
                $this->set_page_error($page_error);
                $this->set_flash_msg($page_error,
"warning");
                return $this->redirect("notifikasi");
                }
                                        }
                                }
                        }
                        $allowed_roles = array
('administrator');
                if(!in_array(strtolower(USER_ROLE),
$allowed_roles)){
                        $db->where("notifikasi.id_user",
get_active_user('id_user') );
                        }
                        $db-
>where("notifikasi.id_notifikasi", $rec_id);;
                        $data = $db->getOne($tablename,
$fields);
                        $page_title = $this->view-
>page_title = "Edit  Notifikasi";
                        if(!$data){
```

```
                $this->set_page_error();
            }
            return $this->render_view("notifikasi/edit.php", $data);
        }
        /**
     * Update single field
         * @param $rec_id (select record by table primary key)
         * @param $formdata array() from $_POST
     * @return array
     */
        function editfield($rec_id = null, $formdata = null){
            $db = $this->GetModel();
            $this->rec_id = $rec_id;
            $tablename = $this->tablename;
            //editable fields
            $fields = $this->fields = array("id_notifikasi","id_user","id_dokumen","jenis_notifikasi","judul_notifikasi","pesan","status_baca","tanggal_notifikasi");
            $page_error = null;
            if($formdata){
                $postdata = array();
                $fieldname = $formdata['name'];
                $fieldvalue = $formdata['value'];
                $postdata[$fieldname] = $fieldvalue;
                $postdata = $this->format_request_data($postdata);
                $this->rules_array = array(
        'id_dokumen' => 'required',
        'jenis_notifikasi' => 'required',
        'judul_notifikasi' => 'required',
                    'pesan' => 'required',
        'status_baca' => 'required',
        'tanggal_notifikasi' => 'required',
                    );
                $this->sanitize_array = array(
        'id_dokumen' => 'sanitize_string',
        'jenis_notifikasi' => 'sanitize_string',
        'judul_notifikasi' => 'sanitize_string',
                    'pesan' => 'sanitize_string',
        'status_baca' => 'sanitize_string',
        'tanggal_notifikasi' => 'sanitize_string',
                    );
                $this->filter_rules = true; //filter validation rules by excluding fields not in the formdata
                $modeldata = $this->modeldata = $this->validate_form($postdata);
                if($this->validated()){
                    $allowed_roles = array('administrator');
        if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
                    $db->where("notifikasi.id_user", get_active_user('id_user') );
                    }
                    $db->where("notifikasi.id_notifikasi", $rec_id);;
                    $bool = $db->update($tablename, $modeldata);
                    $numRows = $db->getRowCount();
                    if($bool && $numRows){
        return render_json(
        array(
            'num_rows' =>$numRows,
```

```
                'rec_id' =>$rec_id,

            )

            );
                                    }
                                    else{

            if($db->getLastError()){

            $page_error = $db->getLastError();

            }

            elseif(!$numRows){

            $page_error = "No record updated";

            }

            render_error($page_error);
                                    }
                            }
                            else{

            render_error($this->view->page_error);
                            }
                    }
                    return null;
            }
            /**
    * Delete record from the database
             * Support multi delete by separating record id by comma.
    * @return BaseView
    */
            function delete($rec_id = null){
                    Csrf::cross_check();
                    $request = $this->request;
                    $db = $this->GetModel();
                    $tablename = $this->tablename;
                    $this->rec_id = $rec_id;
                    //form multiple delete, split record id separated by comma into array
                    $arr_rec_id = array_map('trim', explode(",", $rec_id));
                    $db->where("notifikasi.id_notifikasi", $arr_rec_id, "in");
                    $allowed_roles = array ('administrator');

                if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
                    $db->where("notifikasi.id_user", get_active_user('id_user') );
                    }
                    $bool = $db->delete($tablename);
                    if($bool){
                            $this->set_flash_msg("Record deleted successfully", "success");
                    }
                    elseif($db->getLastError()){
                            $page_error = $db->getLastError();
                            $this->set_flash_msg($page_error, "danger");
                    }
                    return $this->redirect("notifikasi");
                }
}
-pengajuan
<?php
/**
 * Pengajuan Page Controller
 * @category  Controller
 */
class PengajuanController extends SecureController{
            function __construct(){
                    parent::__construct();
                    $this->tablename = "pengajuan";
            }
            /**
    * List page records
    * @param $fieldname (filter record by a field)
    * @param $fieldvalue (filter field value)
    * @return BaseView
    */
```

```
function index($fieldname = null , $fieldvalue = null){
		$request = $this->request;
		$db = $this->GetModel();
		$tablename = $this->tablename;
		$fields = array("id_pengajuan",
				"kategori",
				"tipe_pengajuan",
				"file_upload",
				"status",
				"pemohon",
				"tgl_pengajuan",
				"isi_revisi",
				"id_user",
				"nama_instansi",
				"penanggungjawab",
				"jenis_usaha",
				"lokasi",
				"luas_lahan",
				"dampak_signifikan",
				"potensi",
				"upaya_pengelolaan",
				"tgl_pernyataan",
				"ktp",
				"npwp",
				"akte_perusahaan",
				"amdal",
				"surat_izin_usaha",
				"sertfikat_keahlian",
				"rencana_pengelolaan",
				"surat_pernyataan");
		$pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
		//search table record
		if(!empty($request->search)){
			$text = trim($request->search);
			$search_condition = "(
	pengajuan.id_pengajuan LIKE ? OR
	pengajuan.kategori LIKE ? OR
	pengajuan.tipe_pengajuan LIKE ? OR
	pengajuan.persyaratan LIKE ? OR
	pengajuan.file_upload LIKE ? OR
	pengajuan.status LIKE ? OR
	pengajuan.pemohon LIKE ? OR
	pengajuan.tgl_pengajuan LIKE ? OR
	pengajuan.isi_revisi LIKE ? OR
	pengajuan.id_user LIKE ? OR
	pengajuan.nama_instansi LIKE ? OR
	pengajuan.penanggungjawab LIKE ? OR
	pengajuan.jenis_usaha LIKE ? OR
	pengajuan.lokasi LIKE ? OR
	pengajuan.luas_lahan LIKE ? OR
	pengajuan.dampak_signifikan LIKE ? OR
	pengajuan.potensi LIKE ? OR
	pengajuan.upaya_pengelolaan LIKE ? OR
	pengajuan.pernyataan_kesanggupan LIKE ? OR
	pengajuan.tgl_pernyataan LIKE ? OR
	pengajuan.ktp LIKE ? OR
	pengajuan.npwp LIKE ? OR
	pengajuan.akte_perusahaan LIKE ? OR
	pengajuan.amdal LIKE ? OR
	pengajuan.surat_izin_usaha LIKE ? OR
```

```
            pengajuan.sertfikat_keahlian LIKE ? OR

            pengajuan.rencana_pengelolaan LIKE ? OR

            pengajuan.surat_pernyataan LIKE ?
                        )";
                        $search_params = array(

            "%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%","%$text%"
                        );
                        //setting search conditions
                        $db->where($search_condition, $search_params);
                        //template to use when ajax search
                        $this->view->search_template = "pengajuan/search.php";
                    }
                    if(!empty($request->orderby)){
                        $orderby = $request->orderby;
                        $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                        $db->orderBy($orderby, $ordertype);
                    }
                    else{
                        $db->orderBy("pengajuan.id_pengajuan", ORDER_TYPE);
                    }
                    $allowed_roles = array ('administrator');

            if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
                    $db->where("pengajuan.id_user", get_active_user('id_user') );
                    }
                    if($fieldname){
                        $db->where($fieldname , $fieldvalue); //filter by a single field name
                    }
                    $tc = $db->withTotalCount();
                    $records = $db->get($tablename, $pagination, $fields);
                    $records_count = count($records);
                    $total_records = intval($tc->totalCount);
                    $page_limit = $pagination[1];
                    $total_pages = ceil($total_records / $page_limit);
                    $data = new stdClass;
                    $data->records = $records;
                    $data->record_count = $records_count;
                    $data->total_records = $total_records;
                    $data->total_page = $total_pages;
                    if($db->getLastError()){
                        $this->set_page_error();
                    }
                    $page_title = $this->view->page_title = "Pengajuan";
                    $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                    $this->view->report_title = $page_title;
                    $this->view->report_layout = "report_layout.php";
                    $this->view->report_paper_size = "A4";
                    $this->view->report_orientation = "portrait";
                    $this->render_view("pengajuan/list.php", $data); //render the full page
                }
                /**
```

```
    * View record detail
     * @param $rec_id (select record by table primary key)
    * @param $value value (select record by value of field name(rec_id))
    * @return BaseView
    */
        function view($rec_id = null, $value = null){
            $request = $this->request;
            $db = $this->GetModel();
            $rec_id = $this->rec_id = urldecode($rec_id);
            $tablename = $this->tablename;
            $fields = array("id_pengajuan",
                    "kategori",
                    "tipe_pengajuan",
                    "file_upload",
                    "status",
                    "pemohon",
                    "tgl_pengajuan",
                    "isi_revisi",
                    "id_user",
                    "nama_instansi",
                    "penanggungjawab",
                    "jenis_usaha",
                    "lokasi",
                    "luas_lahan",
                    "dampak_signifikan",
                    "potensi",
                    "upaya_pengelolaan",
                    "tgl_pernyataan",
                    "ktp",
                    "npwp",
                    "akte_perusahaan",
                    "amdal",
                    "surat_izin_usaha",
                    "sertfikat_keahlian",

        "rencana_pengelolaan",
                    "surat_pernyataan");
            $allowed_roles = array ('administrator');

        if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
            $db->where("pengajuan.id_user", get_active_user('id_user') );
            }
            if($value){
                $db->where($rec_id, urldecode($value)); //select record based on field name
            }
            else{
                $db->where("pengajuan.id_pengajuan", $rec_id);; //select record based on primary key
            }
            $record = $db->getOne($tablename, $fields );
            if($record){
                $page_title = $this->view->page_title = "View  Pengajuan";
                $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                $this->view->report_title = $page_title;
                $this->view->report_layout = "report_layout.php";
                $this->view->report_paper_size = "A4";
                $this->view->report_orientation = "portrait";
            }
            else{
                if($db->getLastError()){
                    $this->set_page_error();
                }
                else{
                    $this->set_page_error("No record found");
                }
            }
            return $this->render_view("pengajuan/view.php", $record);
        }
        /**
    * Insert new record to the database table
     * @param $formdata array() from $_POST
    * @return BaseView
```

```
    */
        function add($formdata = null){
            if($formdata){
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $request = $this->request;
                //fillable fields
                $fields = $this->fields = array("kategori","tipe_pengajuan","file_upload","tgl_pengajuan","pemohon","id_user","nama_instansi","penanggungjawab","jenis_usaha","lokasi","luas_lahan","dampak_signifikan","potensi","upaya_pengelolaan","tgl_pernyataan","ktp","npwp","akte_perusahaan","amdal","surat_izin_usaha","sertfikat_keahlian","rencana_pengelolaan","surat_pernyataan");
                $postdata = $this->format_request_data($formdata);
                $this->rules_array = array(
                    'kategori' => 'required',
                    'tipe_pengajuan' => 'required',
                    'file_upload' => 'required',
                    'tgl_pengajuan' => 'required',
                    'nama_instansi' => 'required',
                    'penanggungjawab' => 'required',
                    'jenis_usaha' => 'required',
                    'lokasi' => 'required',
                    'luas_lahan' => 'required|numeric',
                    'dampak_signifikan' => 'required',
                    'potensi' => 'required',
                    'upaya_pengelolaan' => 'required',
                    'tgl_pernyataan' => 'required',
                    'ktp' => 'required',
                    'npwp' => 'required',
                    'akte_perusahaan' => 'required',
                    'amdal' => 'required',
                    'surat_izin_usaha' => 'required',
                    'sertfikat_keahlian' => 'required',
                    'rencana_pengelolaan' => 'required',
                    'surat_pernyataan' => 'required',
                );
                $this->sanitize_array = array(
                    'kategori' => 'sanitize_string',
                    'tipe_pengajuan' => 'sanitize_string',
                    'file_upload' => 'sanitize_string',
                    'tgl_pengajuan' => 'sanitize_string',
                    'pemohon' => 'sanitize_string',
                    'nama_instansi' => 'sanitize_string',
                    'penanggungjawab' => 'sanitize_string',
                    'jenis_usaha' => 'sanitize_string',
                    'lokasi' => 'sanitize_string',
                    'luas_lahan' => 'sanitize_string',
                    'dampak_signifikan' => 'sanitize_string',
```

```
                                        'potensi' =>
'sanitize_string',

            'upaya_pengelolaan' => 'sanitize_string',

            'tgl_pernyataan' => 'sanitize_string',
                                        'ktp'     =>
'sanitize_string',
                                        'npwp'    =>
'sanitize_string',

            'akte_perusahaan' => 'sanitize_string',
                                        'amdal'   =>
'sanitize_string',

            'surat_izin_usaha' => 'sanitize_string',

            'sertfikat_keahlian' => 'sanitize_string',

            'rencana_pengelolaan' => 'sanitize_string',

            'surat_pernyataan' => 'sanitize_string',
                                    );
                                    $this->filter_vals    =
true; //set whether to remove empty fields
                                    $modeldata    =    $this-
>modeldata = $this->validate_form($postdata);
                                    $modeldata['id_user']
= USER_ID;
                                    if($this->validated()){
                                        $rec_id    =
$this->rec_id = $db->insert($tablename, $modeldata);

            if($rec_id){

            $this->set_flash_msg("Record          added
successfully", "success");

            return      $this->redirect("pengajuan");
                                        }
                                        else{

            $this->set_page_error();
                                        }
                                    }
                        }
                        $page_title    =    $this->view-
>page_title = "Add New Pengajuan";
                        $this-
>render_view("pengajuan/add.php");
            }
            /**
    * Update table record with formdata
             * @param $rec_id (select record by table
primary key)
             * @param $formdata array() from $_POST
    * @return array
    */
            function edit($rec_id = null, $formdata =
null){
                        $request = $this->request;
                        $db = $this->GetModel();
                        $this->rec_id = $rec_id;
                        $tablename = $this->tablename;
                         //editable fields
                        $fields    =    $this->fields    =
array("id_pengajuan","kategori","tipe_pengajuan","file
_upload","status","isi_revisi","tgl_pengajuan","nama_in
stansi","penanggungjawab","jenis_usaha","lokasi","luas
_lahan","dampak_signifikan","potensi","upaya_pengelo
laan","tgl_pernyataan","ktp","npwp","akte_perusahaan"
,"amdal","surat_izin_usaha","sertfikat_keahlian","renca
na_pengelolaan","surat_pernyataan");
                        if($formdata){
                            $postdata    =    $this-
>format_request_data($formdata);
                            $this->rules_array    =
array(
                                        'kategori'
=> 'required',

            'tipe_pengajuan' => 'required',

            'file_upload' => 'required',
                                        'status'  =>
'required',
                                        'isi_revisi'
=> 'required',

            'tgl_pengajuan' => 'required',

            'nama_instansi' => 'required',
```

```
            'penanggungjawab' => 'required',
            'jenis_usaha' => 'required',
                                    'lokasi'   =>
'required',
            'luas_lahan' => 'required|numeric',
            'dampak_signifikan' => 'required',
                                    'potensi' =>
'required',
            'upaya_pengelolaan' => 'required',
            'tgl_pernyataan' => 'required',
                                    'ktp'      =>
'required',
                                    'npwp'     =>
'required',
            'akte_perusahaan' => 'required',
                                    'amdal'    =>
'required',
            'surat_izin_usaha' => 'required',
            'sertfikat_keahlian' => 'required',
            'rencana_pengelolaan' => 'required',
            'surat_pernyataan' => 'required',
                                );
                                $this->sanitize_array
= array(
                                    'kategori'
=> 'sanitize_string',
            'tipe_pengajuan' => 'sanitize_string',
            'file_upload' => 'sanitize_string',
                                    'status'   =>
'sanitize_string',
                                    'isi_revisi'
=> 'sanitize_string',
            'tgl_pengajuan' => 'sanitize_string',
            'nama_instansi' => 'sanitize_string',
            'penanggungjawab' => 'sanitize_string',
            'jenis_usaha' => 'sanitize_string',
                                                'lokasi'   =>
'sanitize_string',
            'luas_lahan' => 'sanitize_string',
            'dampak_signifikan' => 'sanitize_string',
                                                'potensi' =>
'sanitize_string',
            'upaya_pengelolaan' => 'sanitize_string',
            'tgl_pernyataan' => 'sanitize_string',
                                                'ktp'      =>
'sanitize_string',
                                                'npwp'     =>
'sanitize_string',
            'akte_perusahaan' => 'sanitize_string',
                                                'amdal'    =>
'sanitize_string',
            'surat_izin_usaha' => 'sanitize_string',
            'sertfikat_keahlian' => 'sanitize_string',
            'rencana_pengelolaan' => 'sanitize_string',
            'surat_pernyataan' => 'sanitize_string',
                                );
                                $modeldata = $this-
>modeldata = $this->validate_form($postdata);
                                if($this->validated()){
                        $allowed_roles      =      array
('administrator');
            if(!in_array(strtolower(USER_ROLE),
$allowed_roles)){
```

```
                $db->where("pengajuan.id_user", get_active_user('id_user') );
                }
                $db->where("pengajuan.id_pengajuan", $rec_id);;
                $bool = $db->update($tablename, $modeldata);
                $numRows = $db->getRowCount(); //number of affected rows. 0 = no record field updated
                if($bool && $numRows){

        $this->set_flash_msg("Record updated successfully", "success");

        return $this->redirect("pengajuan");
                }
                else{

        if($db->getLastError()){

        $this->set_page_error();

        }

        elseif(!$numRows){

        //not an error, but no record was updated

        $page_error = "No record updated";

        $this->set_page_error($page_error);

        $this->set_flash_msg($page_error, "warning");

        return $this->redirect("pengajuan");

        }
                }
            }
        }
        $allowed_roles = array ('administrator');
        if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
                $db->where("pengajuan.id_user", get_active_user('id_user') );
                }
                $db->where("pengajuan.id_pengajuan", $rec_id);;
                $data = $db->getOne($tablename, $fields);
                $page_title = $this->view->page_title = "Edit  Pengajuan";
                if(!$data){
                        $this->set_page_error();
                }
                return $this->render_view("pengajuan/edit.php", $data);
        }
        /**
    * Update single field
         * @param $rec_id (select record by table primary key)
         * @param $formdata array() from $_POST
    * @return array
    */
        function editfield($rec_id = null, $formdata = null){
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                //editable fields
                $fields = $this->fields = array("id_pengajuan","kategori","tipe_pengajuan","file_upload","status","isi_revisi","tgl_pengajuan","nama_instansi","penanggungjawab","jenis_usaha","lokasi","luas_lahan","dampak_signifikan","potensi","upaya_pengelolaan","tgl_pernyataan","ktp","npwp","akte_perusahaan","amdal","surat_izin_usaha","sertfikat_keahlian","rencana_pengelolaan","surat_pernyataan");
                $page_error = null;
                if($formdata){
                        $postdata = array();
                        $fieldname = $formdata['name'];
```

```
                $fieldvalue = $formdata['value'];
                $postdata[$fieldname] = $fieldvalue;
            $postdata = $this->format_request_data($postdata);
            $this->rules_array = array(
                    'kategori' => 'required',
        'tipe_pengajuan' => 'required',
        'file_upload' => 'required',
                    'status' => 'required',
                    'isi_revisi' => 'required',
        'tgl_pengajuan' => 'required',
        'nama_instansi' => 'required',
        'penanggungjawab' => 'required',
        'jenis_usaha' => 'required',
                    'lokasi' => 'required',
        'luas_lahan' => 'required|numeric',
        'dampak_signifikan' => 'required',
                    'potensi' => 'required',
        'upaya_pengelolaan' => 'required',
        'tgl_pernyataan' => 'required',
                    'ktp' => 'required',
                    'npwp' => 'required',
        'akte_perusahaan' => 'required',
                    'amdal' => 'required',
        'surat_izin_usaha' => 'required',
        'sertfikat_keahlian' => 'required',
        'rencana_pengelolaan' => 'required',
        'surat_pernyataan' => 'required',
                    );
                    $this->sanitize_array = array(
                            'kategori' => 'sanitize_string',
        'tipe_pengajuan' => 'sanitize_string',
        'file_upload' => 'sanitize_string',
                            'status' => 'sanitize_string',
                            'isi_revisi' => 'sanitize_string',
        'tgl_pengajuan' => 'sanitize_string',
        'nama_instansi' => 'sanitize_string',
        'penanggungjawab' => 'sanitize_string',
        'jenis_usaha' => 'sanitize_string',
                            'lokasi' => 'sanitize_string',
        'luas_lahan' => 'sanitize_string',
        'dampak_signifikan' => 'sanitize_string',
                            'potensi' => 'sanitize_string',
        'upaya_pengelolaan' => 'sanitize_string',
        'tgl_pernyataan' => 'sanitize_string',
                            'ktp' => 'sanitize_string',
                            'npwp' => 'sanitize_string',
```

```
'akte_perusahaan' => 'sanitize_string',
'amdal' => 'sanitize_string',
'surat_izin_usaha' => 'sanitize_string',
'sertfikat_keahlian' => 'sanitize_string',
'rencana_pengelolaan' => 'sanitize_string',
'surat_pernyataan' => 'sanitize_string',
);
$this->filter_rules = true; //filter validation rules by excluding fields not in the formdata
$modeldata = $this->modeldata = $this->validate_form($postdata);
if($this->validated()){
$allowed_roles = array('administrator');
if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
$db->where("pengajuan.id_user", get_active_user('id_user') );
}
$db->where("pengajuan.id_pengajuan", $rec_id);;
$bool = $db->update($tablename, $modeldata);
$numRows = $db->getRowCount();
if($bool && $numRows){
return render_json(
array(
'num_rows' =>$numRows,
'rec_id' =>$rec_id,
)
);
}
else{
if($db->getLastError()){
$page_error = $db->getLastError();
}
elseif(!$numRows){
$page_error = "No record updated";
}
render_error($page_error);
}
}
else{
render_error($this->view->page_error);
}
}
return null;
}
/**
* Delete record from the database
* Support multi delete by separating record id by comma.
* @return BaseView
*/
function delete($rec_id = null){
Csrf::cross_check();
$request = $this->request;
$db = $this->GetModel();
$tablename = $this->tablename;
$this->rec_id = $rec_id;
//form multiple delete, split record id separated by comma into array
$arr_rec_id = array_map('trim', explode(",", $rec_id));
$db->where("pengajuan.id_pengajuan", $arr_rec_id, "in");
```

```
                $allowed_roles = array ('administrator');

        if(!in_array(strtolower(USER_ROLE), $allowed_roles)){
                $db->where("pengajuan.id_user", get_active_user('id_user') );
                }
                $bool = $db->delete($tablename);
                if($bool){
                        $this->set_flash_msg("Record deleted successfully", "success");
                }
                elseif($db->getLastError()){
                        $page_error = $db->getLastError();
                        $this->set_flash_msg($page_error, "danger");
                }
                return $this->redirect("pengajuan");
        }
}
```

-status dokumen

```
<?php
/**
 * Statu_dokumen Page Controller
 * @category  Controller
 */
class Statu_dokumenController extends SecureController{
        function __construct(){
                parent::__construct();
                $this->tablename = "statu_dokumen";
        }
        /**
    * List page records
    * @param $fieldname (filter record by a field)
    * @param $fieldvalue (filter field value)
    * @return BaseView
    */
        function index($fieldname = null , $fieldvalue = null){
                $request = $this->request;
                $db = $this->GetModel();
                $tablename = $this->tablename;
                $fields = array("id_status",
                        "nama_status",
                        "deskripsi_status",
                        "warna_status",
                        "created_at",
                        "updated_at");
                $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
                //search table record
                if(!empty($request->search)){
                        $text = trim($request->search);
                        $search_condition = "(
        statu_dokumen.id_status LIKE ? OR
        statu_dokumen.nama_status LIKE ? OR
        statu_dokumen.deskripsi_status LIKE ? OR
        statu_dokumen.warna_status LIKE ? OR
        statu_dokumen.created_at LIKE ? OR
        statu_dokumen.updated_at LIKE ?
                        )";
                        $search_params = array(
        "%$text%","%$text%","%$text%","%$text%","%$text%","%$text%"
                        );
                        //setting search conditions
                        $db->where($search_condition, $search_params);
                         //template to use when ajax search
                        $this->view->search_template = "statu_dokumen/search.php";
                }
                if(!empty($request->orderby)){
```

```
                $orderby = $request->orderby;
                $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                $db->orderBy($orderby, $ordertype);
            }
            else{
                $db->orderBy("statu_dokumen.id_status", ORDER_TYPE);
            }
            if($fieldname){
                $db->where($fieldname , $fieldvalue); //filter by a single field name
            }
            $tc = $db->withTotalCount();
            $records = $db->get($tablename, $pagination, $fields);
            $records_count = count($records);
            $total_records = intval($tc->totalCount);
            $page_limit = $pagination[1];
            $total_pages = ceil($total_records / $page_limit);
            $data = new stdClass;
            $data->records = $records;
            $data->record_count = $records_count;
            $data->total_records = $total_records;
            $data->total_page = $total_pages;
            if($db->getLastError()){
                $this->set_page_error();
            }
            $page_title = $this->view->page_title = "Statu Dokumen";
            $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
            $this->view->report_title = $page_title;
            $this->view->report_layout = "report_layout.php";
            $this->view->report_paper_size = "A4";
            $this->view->report_orientation = "portrait";
            $this->render_view("statu_dokumen/list.php", $data); //render the full page
        }
        /**
     * View record detail
         * @param $rec_id (select record by table primary key)
     * @param $value value (select record by value of field name(rec_id))
     * @return BaseView
     */
        function view($rec_id = null, $value = null){
            $request = $this->request;
            $db = $this->GetModel();
            $rec_id = $this->rec_id = urldecode($rec_id);
            $tablename = $this->tablename;
            $fields = array("id_status",
                "nama_status",
                "deskripsi_status",
                "warna_status",
                "created_at",
                "updated_at");
            if($value){
                $db->where($rec_id, urldecode($value)); //select record based on field name
            }
            else{
                $db->where("statu_dokumen.id_status", $rec_id);; //select record based on primary key
            }
            $record = $db->getOne($tablename, $fields );
            if($record){
                $page_title = $this->view->page_title = "View  Statu Dokumen";
                $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                $this->view->report_title = $page_title;
```

```
                $this->view->report_layout = "report_layout.php";
                $this->view->report_paper_size = "A4";
                $this->view->report_orientation = "portrait";
                }
                else{
                    if($db->getLastError()){
                        $this->set_page_error();
                    }
                    else{
                        $this->set_page_error("No record found");
                    }
                }
                return $this->render_view("statu_dokumen/view.php", $record);
            }
            /**
     * Insert new record to the database table
             * @param $formdata array() from $_POST
     * @return BaseView
     */
            function add($formdata = null){
                if($formdata){
                    $db = $this->GetModel();
                    $tablename = $this->tablename;
                    $request = $this->request;
                    //fillable fields
                    $fields = $this->fields = array("nama_status","deskripsi_status","warna_status");
                    $postdata = $this->format_request_data($formdata);
                    $this->rules_array = array(
            'nama_status' => 'required',
            'deskripsi_status' => 'required',
            'warna_status' => 'required',
                    );
                    $this->sanitize_array = array(
            'nama_status' => 'sanitize_string',
            'deskripsi_status' => 'sanitize_string',
            'warna_status' => 'sanitize_string',
                    );
                    $this->filter_vals = true; //set whether to remove empty fields
                    $modeldata = $this->modeldata = $this->validate_form($postdata);
                    if($this->validated()){
                        $rec_id = $this->rec_id = $db->insert($tablename, $modeldata);
            if($rec_id){
            $this->set_flash_msg("Record added successfully", "success");
            return $this->redirect("statu_dokumen");
                        }
                        else{
            $this->set_page_error();
                        }
                    }
                    $page_title = $this->view->page_title = "Add New Statu Dokumen";
                    $this->render_view("statu_dokumen/add.php");
                }
                /**
     * Update table record with formdata
                 * @param $rec_id (select record by table primary key)
                 * @param $formdata array() from $_POST
     * @return array
     */
```

```
	function edit($rec_id = null, $formdata = null){
		$request = $this->request;
		$db = $this->GetModel();
		$this->rec_id = $rec_id;
		$tablename = $this->tablename;
		 //editable fields
		$fields = $this->fields = array("id_status","nama_status","deskripsi_status","warna_status");
		if($formdata){
			$postdata = $this->format_request_data($formdata);
			$this->rules_array = array(
	'nama_status' => 'required',
	'deskripsi_status' => 'required',
	'warna_status' => 'required',
			);
			$this->sanitize_array = array(
	'nama_status' => 'sanitize_string',
	'deskripsi_status' => 'sanitize_string',
	'warna_status' => 'sanitize_string',
			);
			$modeldata = $this->modeldata = $this->validate_form($postdata);
			if($this->validated()){
				$db->where("statu_dokumen.id_status", $rec_id);;
				$bool = $db->update($tablename, $modeldata);
				$numRows = $db->getRowCount(); //number of affected rows. 0 = no record field updated
				if($bool && $numRows){
	$this->set_flash_msg("Record updated successfully", "success");
	return $this->redirect("statu_dokumen");
				}
				else{
	if($db->getLastError()){
	$this->set_page_error();
	}
	elseif(!$numRows){
	//not an error, but no record was updated
	$page_error = "No record updated";
	$this->set_page_error($page_error);
	$this->set_flash_msg($page_error, "warning");
	return $this->redirect("statu_dokumen");
	}
				}
			}
		}
		$db->where("statu_dokumen.id_status", $rec_id);;
		$data = $db->getOne($tablename, $fields);
		$page_title = $this->view->page_title = "Edit  Statu Dokumen";
		if(!$data){
			$this->set_page_error();
		}
		return $this->render_view("statu_dokumen/edit.php", $data);
	}
	/**
    * Update single field
	 * @param $rec_id (select record by table primary key)
```

```
     * @param $formdata array() from $_POST
    * @return array
    */
        function editfield($rec_id = null, $formdata = null){
                $db = $this->GetModel();
                $this->rec_id = $rec_id;
                $tablename = $this->tablename;
                //editable fields
                $fields = $this->fields = array("id_status","nama_status","deskripsi_status","warna_status");
                $page_error = null;
                if($formdata){
                        $postdata = array();
                        $fieldname = $formdata['name'];
                        $fieldvalue = $formdata['value'];
                        $postdata[$fieldname] = $fieldvalue;
                        $postdata = $this->format_request_data($postdata);
                        $this->rules_array = array(

        'nama_status' => 'required',

        'deskripsi_status' => 'required',

        'warna_status' => 'required',
                        );
                        $this->sanitize_array = array(

        'nama_status' => 'sanitize_string',

        'deskripsi_status' => 'sanitize_string',

        'warna_status' => 'sanitize_string',
                        );
                        $this->filter_rules = true; //filter validation rules by excluding fields not in the formdata
                        $modeldata = $this->modeldata = $this->validate_form($postdata);
                        if($this->validated()){
                                $db->where("statu_dokumen.id_status", $rec_id);;
                                $bool = $db->update($tablename, $modeldata);
                                $numRows = $db->getRowCount();
                                if($bool && $numRows){

        return render_json(

        array(

                'num_rows' =>$numRows,

                'rec_id' =>$rec_id,

        )

        );
                                }
                                else{

        if($db->getLastError()){

        $page_error = $db->getLastError();

        }

        elseif(!$numRows){

        $page_error = "No record updated";

        }

        render_error($page_error);
                                }
                        }
                        else{

        render_error($this->view->page_error);
                        }
                }
                return null;
        }
```

```
        /**
    * Delete record from the database
            * Support multi delete by separating record id by comma.
    * @return BaseView
    */
        function delete($rec_id = null){
            Csrf::cross_check();
            $request = $this->request;
            $db = $this->GetModel();
            $tablename = $this->tablename;
            $this->rec_id = $rec_id;
            //form multiple delete, split record id separated by comma into array
            $arr_rec_id = array_map('trim', explode(",", $rec_id));
            $db->where("statu_dokumen.id_status", $arr_rec_id, "in");
            $bool = $db->delete($tablename);
            if($bool){
                $this->set_flash_msg("Record deleted successfully", "success");
            }
            elseif($db->getLastError()){
                $page_error = $db->getLastError();
                $this->set_flash_msg($page_error, "danger");
            }
            return $this->redirect("statu_dokumen");
        }
}
-view formulir
<?php
/**
 * View_formulir Page Controller
 * @category  Controller
 */
class View_formulirController extends SecureController{
        function __construct(){
            parent::__construct();
            $this->tablename = "view_formulir";
        }
        /**
    * List page records
    * @param $fieldname (filter record by a field)
    * @param $fieldvalue (filter field value)
    * @return BaseView
    */
        function index($fieldname = null , $fieldvalue = null){
            $request = $this->request;
            $db = $this->GetModel();
            $tablename = $this->tablename;
            $fields = array("kode_formulir",
                "tipe_formulir",
                "file_formulir");
            $pagination = $this->get_pagination(MAX_RECORD_COUNT); // get current pagination e.g array(page_number, page_limit)
            //search table record
            if(!empty($request->search)){
                $text = trim($request->search);
                $search_condition = "(
        view_formulir.id_formulir LIKE ? OR
        view_formulir.kode_formulir LIKE ? OR
        view_formulir.tipe_formulir LIKE ? OR
        view_formulir.file_formulir LIKE ?
                )";
                $search_params = array(
        "%$text%","%$text%","%$text%","%$text%"
                );
                //setting search conditions
                $db->where($search_condition, $search_params);
                 //template to use when ajax search
                $this->view->search_template = "view_formulir/search.php";
```

```
                }
                if(!empty($request->orderby)){
                        $orderby = $request->orderby;
                        $ordertype = (!empty($request->ordertype) ? $request->ordertype : ORDER_TYPE);
                        $db->orderBy($orderby, $ordertype);
                }
                else{
                        $db->orderBy("view_formulir.id_formulir", ORDER_TYPE);
                }
                if($fieldname){
                        $db->where($fieldname , $fieldvalue); //filter by a single field name
                }
                $tc = $db->withTotalCount();
                $records = $db->get($tablename, $pagination, $fields);
                $records_count = count($records);
                $total_records = intval($tc->totalCount);
                $page_limit = $pagination[1];
                $total_pages = ceil($total_records / $page_limit);
                $data = new stdClass;
                $data->records = $records;
                $data->record_count = $records_count;
                $data->total_records = $total_records;
                $data->total_page = $total_pages;
                if($db->getLastError()){
                        $this->set_page_error();
                }
                $page_title = $this->view->page_title = "View Formulir";
                $this->view->report_filename = date('Y-m-d') . '-' . $page_title;
                $this->view->report_title = $page_title;
                $this->view->report_layout = "report_layout.php";
                $this->view->report_paper_size = "A4";
                $this->view->report_orientation = "portrait";
                $this->render_view("view_formulir/list.php", $data); //render the full page
        }
// No View Function Generated Because No Field is Defined as the Primary Key on the Database Table
}
```

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : St.Khaeriani Akhmar	**Pembimbing I : Ade Hastuty, ST., S.Kom., MT.**
NIM : 219280087	**Pembimbing II : Ir. Untung Suwardoyo, S.Kom., MT.,IPP**
Judul Skripsi : Aplikasi Pengelolaan Dokumen Pada Dinas Lingkungan Hidup Dengan Metode Waterfall	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 Review Artikel	18 Juli 2023	Konsultasi 1 - Penomoran halaman - Penulisan judul - Perumusan masalah - Desain form	16 November 2023
Konsultasi 2 Buat Proposal	15 Agustus 2023	Konsultasi 2 - Perumusan masalah - Penomoran sub-bab - Margin - Kerangka berpikir.	
Konsultasi 3 Use case Desain Logika Acc Seminar Proposal	05 Oktober 2023	Konsultasi 3 - Kajian terdahulu - Contoh tag - Contoh white box, black box - Flowchart sistem yang diusulkan.	24/Januari/2024
Konsultasi 4		Konsultasi 4 Desain fisik dan Flowchart APP	5 Maret 2024
Konsultasi 5		Konsultasi 5 Acc proposal.	20/Mei/2024

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : St.Khaeriani Akhmar	Pembimbing I : Ade Hastuty, ST., S.Kom., MT.
NIM : 219280087	Pembimbing II : Ir. Untung Suwardoyo, S.Kom., MT.,IPP
Judul Skripsi : Aplikasi Pengelolaan Dokumen Pada Dinas Lingkungan Hidup Kota Parepare	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 -teori flowchart, uml, mulai uji white box, black -database yg dipakai -flowchart, diperbaiki -prosedur.	16/05/25	Konsultasi 1	25-06-2025
Konsultasi 2 -flowchart edit, tambah, hapus disesuaikan -coding berlatar putih/di word. -kumpulan berdasar rumusan masalah	20/07/25	Konsultasi 2	23-08-2025
Konsultasi 3 Acc Ujian Hasil	25/08/25	Konsultasi 3	
Konsultasi 4		Konsultasi 4	
Konsultasi 5		Konsultasi 5	

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

SKRIPSI

Mahasiswa : ST.KHAERIANI AKHMAR	**Pembimbing I : Ade Hastuty, ST., S.Kom., MT.**
NIM : 219280087	**Pembimbing II : Ir.Untung Suwardoyo, S.Kom., MT., IPP**
Judul Skripsi : APLIKASI PENGELOLAAN DOKUMEN PADA DINAS LINGKUNGAN HIDUP KOTA PAREPARE	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 ACC Tutup	09/09/2025	Konsultasi 1 ACC	09/09/2025
Konsultasi 2		Konsultasi 2	
Konsultasi 3		Konsultasi 3	
Konsultasi 4		Konsultasi 4	
Konsultasi 5		Konsultasi 5	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing