BAB III

PELAKSANAAN KERJA MAGANG

3.1. Kedudukan dan Koordinasi

Dalam pengembangan dan perancangan sistem project management penulis mengerjakannya bersama Steven Wijaya sebagai satu tim menggunakan *framework iterative waterfall model*. Dimana penulis melakukan *user requirement analysis* terlebih dahulu untuk mengetahui cara kerja dari sistem *project management* yang ada saat ini. Kemudian tim akan melakukan desain seperti *framework backend* dan *database management system* yang akan digunakan. Selanjutnya pada tahap *coding* dan *unit testing* penulis bersama tim akan membagi pekerjaan, dilanjutkan dengan melakukan *deployment* ke *web server* kemudian yang terakhir *testing* ke *user* dan *maintenance* apabila ada perubahan atau penambahan fitur.

Kegiatan kerja magang dilakukan dengan pengawasan dan bimbingan dari Erlina Prasetyowati selaku General Manager MIS (*Management Information System*), dan berperan untuk memberikan informasi mengenai sistem *project management* yang akan dibuat seperti *workflow* dan fitur-fitur yang diinginkan untuk ada pada sistem tersebut. Lalu user juga akan memberikan masukan-masukan atau penambahan fitur mengenai sistem yang sudah dibangun.

3.2. Tugas yang dilakukan

Tugas yang dilakukan dibagi berdasarkan fitur yang dibuat, dimana penulis membuat *login, upload* dokumen csv yang akan digunakan untuk memasukkan *site list* (daftar tower) oleh divisi *sales*, kemudian *form* yang akan digunakan untuk memasukkan data Site Hunting (proses mencari lokasi tower yang akan dibangun) dan BAN Process (proses negosiasi harga sewa lahan) serta penulis juga mengerjakan fitur *approval* beserta *manage approval* (proses persetujuan dari atasan ketika mengajukan dokumen *milestone* tertentu).

Proses kerja menggunakan perangkat GitHub sebagai *version control system* untuk memudahkan proses *development* karena penulis mengerjakan bersama tim. Setiap *pull* / *push* wajib dimasukan terlebih dahulu ke *branch* sebelum di *merge* ke *master*. Berikut tabel realisasi dari proses kerja magang.

3.3. Uraian Pelaksanaan Kerja Magang

Proses pelaksanaan kerja magang dibagi menjadi tiga bagian yaitu proses pelaksanaan, kendala yang ditemukan, dan solusi atas kendala yang ditemukan.

3.3.1. Proses Pelaksanaan

Proses pelaksanaan pengembangan sistem *project management* pada PT Gihon Telekomunikasi Indonesia membutuhkan perangkat lunak dan perangkat keras. Perangkat lunak yang digunakan untuk membangun sistem *project management* adalah sebagai berikut.

1. Visual Studio Code versi 1.4
2. Laragon versi 4.0.16
3. phpMyAdmin versi 5.0
4. PHP versi 7.4.8
5. MySQL versi 5.7.24
6. Github Desktop 2.5.4

7. Google Chrome versi 85.0.4183.83 (64-bit)
8. Operating System Windows 10 Home Single Language 64-bit

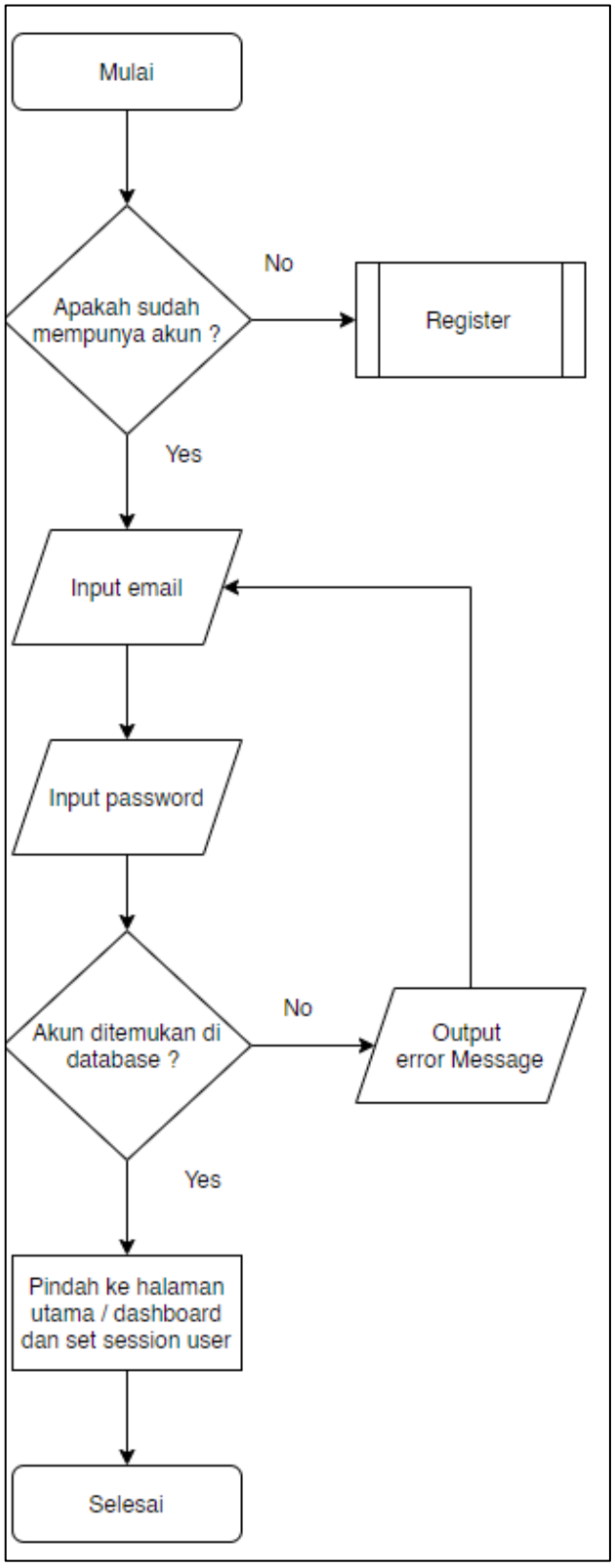
Perangkat keras yang digunakan untuk membangun sistem *project management* adalah laptop Asus TUF FX505DY dengan spesifikasi sebagai berikut.

1. *Processor* AMD Ryzen 5 3550H
2. RAM 16 GB
3. *Integrated Graphic Card* AMD Radeon Vega 8
4. *Dedicated Graphic* Card AMD Radeon RX 560X

Proses pengerjaan dibagi menjadi bagian 4 berdasarkan pengembangan fitur yang dikerjakan oleh penulis yaitu, *login*, *user management - register*, *manage sites – insert new data*, *request approval*, *manage approval*, *form site hunting* dan *form ban process* seperti yang sudah dijabarkan pada tabel realisasi kerja magang.

A. *Login*

Login adalah halaman yang pertama kali user lihat ketika membuka *website* sistem *project management* (pm.indonesiamart.id). Seperti halaman *login* pada umumnya, maka *user* akan diminta untuk memasukkan *email* dan *password*. Untuk mengetahui alur kerja dari fitur login dapat dilih pada *flowchart login* (Gambar 3.1) Berikut adalah tampilan (Gambar 3.2) dan *code* yang di gunakan untuk *page login* (Gambar 3.3).



Gambar 3.1 *Flowchart Login*

Gambar 3.2 Tampilan *Login*

```php
public function login_authenticator(){
    $employeeModel = new \App\Models\EmployeeModel();
    $session = session();
    $email = $_POST['email'];
    $password = md5($_POST['pass']);
    $error = "Email dan/atau password salah. <br> Silahkan coba lagi";
    $currUser = $employeeModel->where("Email",$email)->where("Password",$password)->first();
    if($currUser != null){
        $this->setUserSession($currUser);
        return redirect()->to(base_url('Home/checkLogin'));
    }else{
        $session->setFlashdata('error_login', $error);
        return redirect()->to(base_url());
    }
}
```

Gambar 3.3 *Controller Login*

Ketika *user* memasukkan *username* dan *password* milik mereka dan menekan *button login*, maka *form* akan menjalankan *controller login* Gambar 3.3, dimana pada *controller* tersebut sudah terdapat *variable* $employeeModel yang digunakan untuk menampung *model* dari *EmployeeModel* (Gambar 3.4) untuk *query* ke *database*, kemudian *session* di *generate*, karena password yang disimpan dalam *database* berbentuk *hashing* md5 agar tidak bisa diketahui oleh *admin*, maka di sini *controller* akan mengubah *password* yang di $_POST menjadi bentuk md5 untuk digunakan *query* ke *database*.

```php
class EmployeeModel extends Model
{
    protected $table = 'employee';
    protected $primaryKey = 'EmployeeID';
    protected $allowedFields = ['Name','PhoneNumber','Email','Password','Division','Privilege'];

    public function getAllEmployee()
    {
        return $this->findAll();
    }

    public function insertEmployee($data){
        return $this->insert($data);
    }
    public function updatePrivilege($EmployeeID,$NewPrivilege){
        return $this->update($EmployeeID,$NewPrivilege);
    }
    public function updatePhoneNumber($EmployeeID,$NewPhone){
        return $this->update($EmployeeID,$NewPhone);
    }
    public function updatePassword($EmployeeID,$NewPass){
        return $this->update($EmployeeID,$NewPass);
    }

}
```

Gambar 3.4 *Employee Model*

Apabila *user* berhasil *login*, maka *session user* akan di-*generate* sehingga *user* tidak perlu *login* kembali apabila belum *log out*, tetapi *session user* akan berakhir apabila tidak ada interaksi dari *user* ke *website* selama 15 menit atau 900 detik. (Gambar 3.5 Durasi *Session User*)

```php
public $sessionDriver            = 'CodeIgniter\Session\Handlers\FileHandler';
public $sessionCookieName        = 'ci_session';
public $sessionExpiration        = 900;
public $sessionSavePath          = WRITEPATH . 'session';
public $sessionMatchIP           = false;
public $sessionTimeToUpdate      = 300;
public $sessionRegenerateDestroy = false;
```

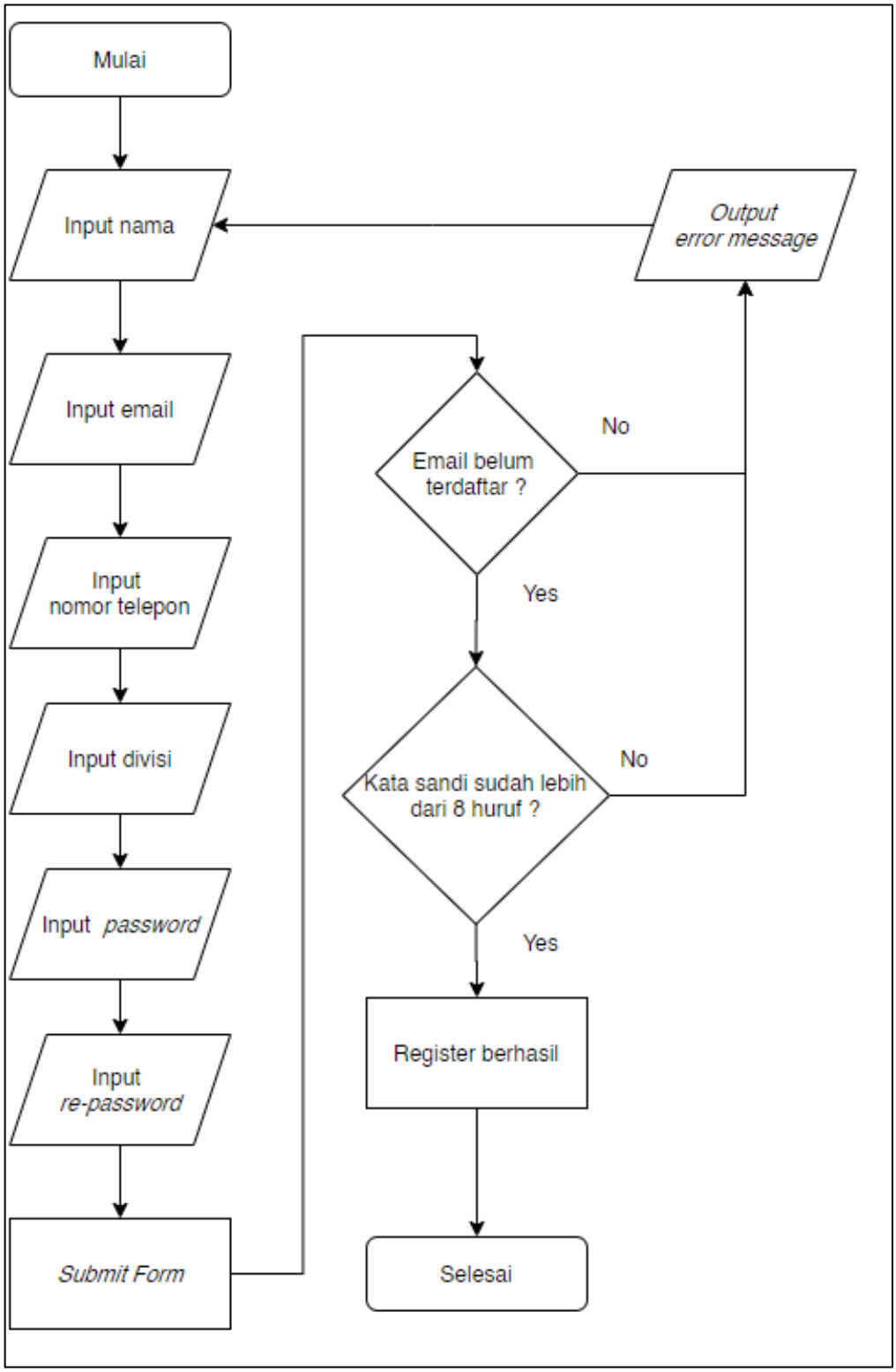
Gambar 3.5 Durasi *Session User*

Apabila *user* gagal untuk *login*, maka *user* akan kembali ke halaman *login* dengan *error message* seperti Gambar 3.6 *Error Login*.

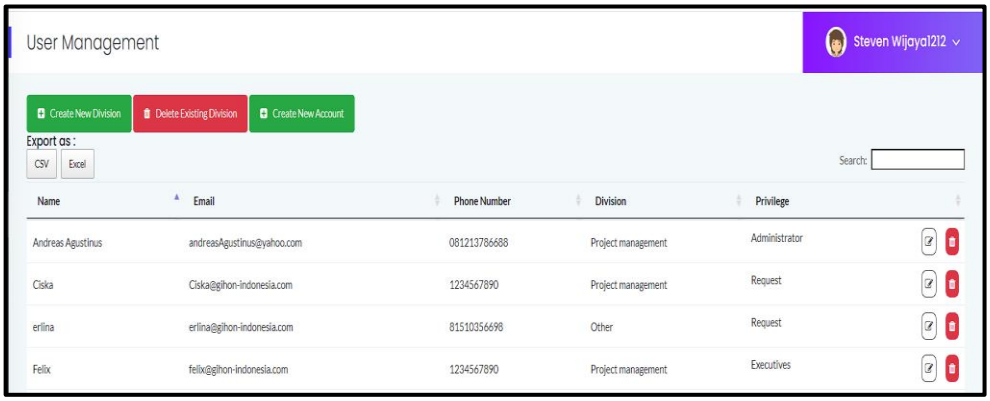
Gambar 3.6 *Error Login*

B. *User Management - Register*

Sebelum *user* memiliki akun untuk *login*, maka *user* perlu meminta *admin* untuk mendaftarkan dirinya terlebih dahulu ke dalam sistem. Untuk alur kerja fitur register dapat dilihat pada *flowchart register* (Gambar 3.7) *Admin* dapat mendaftarkan *user* baru dengan masuk ke *menu User Management* (Gambar 3.8 *User Management*) dimana *menu* tersebut hanya bisa diakses jika memiliki *priviledge* sebagai *admin*.

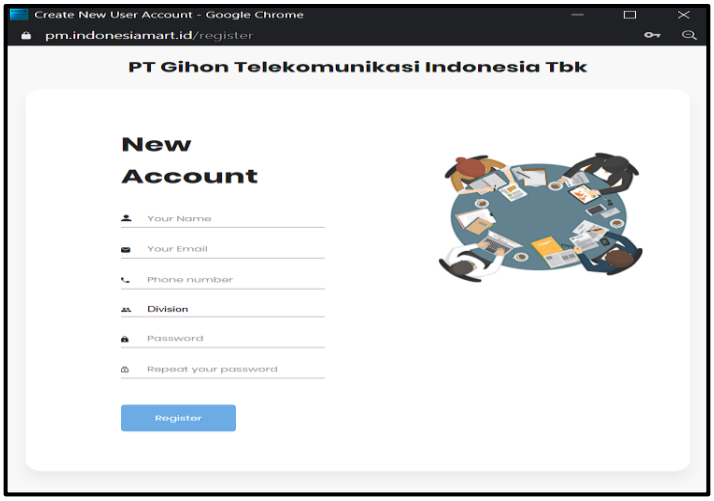


Gambar 3.7 *Flowchart Register*



Gambar 3.8 *User Management*

Lalu *admin* bisa memilih *Create New Account*, maka akan keluar *pop up* seperti Gambar 3.9, kemudian *admin* akan meminta *user* untuk mengisikan data-data pribadi yang diminta oleh sistem. Ketika *form* di-*submit* maka *function Register Authenticator* (Gambar 3.10 *Function Register Authenticator*) dalam *controller login* akan di panggil.

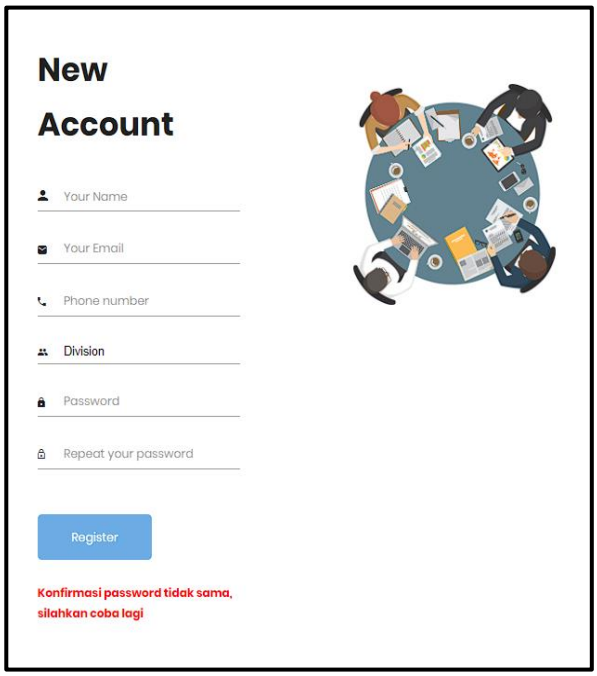


Gambar 3.9 Membuat Akun Baru

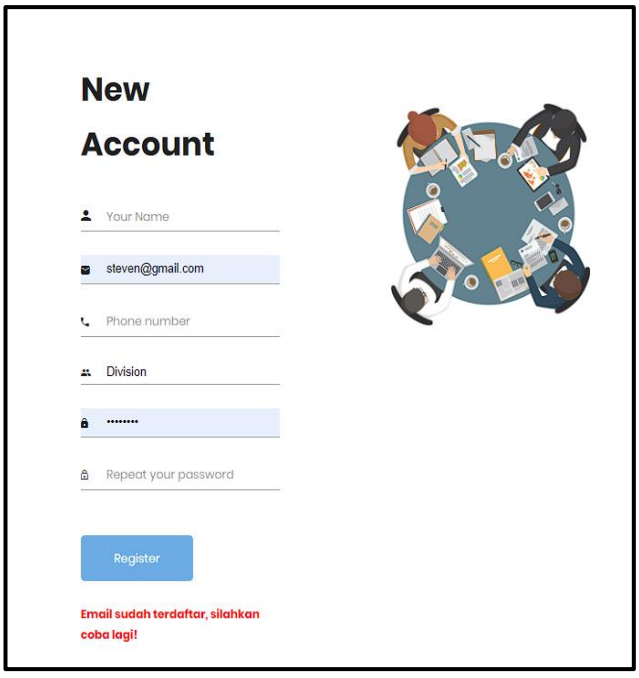
```php
public function register_authenticator(){
    $employeeModel = new \App\Models\EmployeeModel();
    $session = session();
    $name = $_POST['name'];
    $email = $_POST['email'];
    $phoneNumb = $_POST['noTelp'];
    $division = $_POST['divisi'];
    $password = $_POST['pass'];
    $re_pass = $_POST['re_pass'];
    $error_email = "Email sudah terdaftar, silahkan coba lagi!";
    $error_password = "Konfirmasi password tidak sama, silahkan coba lagi";

    if($password != $re_pass ){
        $session->setFlashdata('error_password_confirmation', $error_password);
        return redirect()->to(base_url('Login/register'));
    }else{
        if($employeeModel->select("Email")->where("Email",$email)->find() == null){
            $hashPass = md5($password);
            $data = [
                'Name' => $name,
                'PhoneNumber'  => $phoneNumb,
                'Email'  => $email,
                'Password' => $hashPass,
                'Division' => $division,
                'Privilege' => 'Request'
            ];
            $employeeModel->insertEmployee($data);
            echo'<script>window.close();</script>';


        }else{
            $session->setFlashdata('error_email_duplicate', $error_email);
            return redirect()->to(base_url('Login/register'));
        }
    }
}
```

Gambar 3.10 *Function Register Authenticator*

Untuk melakukan *query* ke *database employee* maka *variable* $employeeModel yang berisi *EmployeeModel* (Gambar 3.4) harus diinisialisasi kembali, setelah itu $_POST akan digunakan untuk mengambil hasil dari *form* yang di-*submit* oleh *user*, jika *password* tidak sama dengan *re-type password* atau ketik ulang password maka akan memunculkan *error* seperti Gambar 3.11, apabila *email* yang dimasukkan oleh *user* sudah terdaftar maka akan memunculkan *error* seperti Gambar 3.12



Gambar 3.11 Error Wrong Password

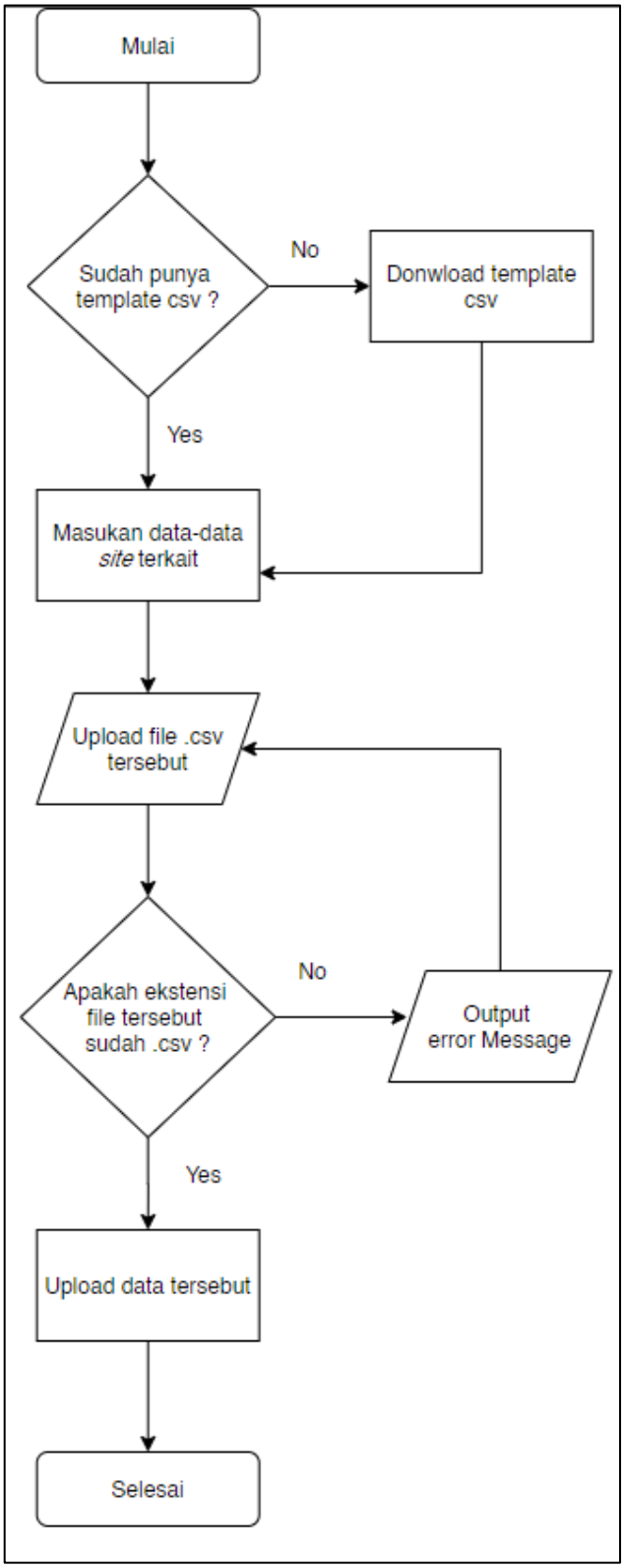


Gambar 3.12 Error Same Email

Jika berhasil *register* maka akan kembali ke halaman *User Management* (Gambar 3.8), lalu akun tersebut sudah terdaftar dengan *privilege default* yaitu *request*.

C. *Manage Sites – Insert New Data*

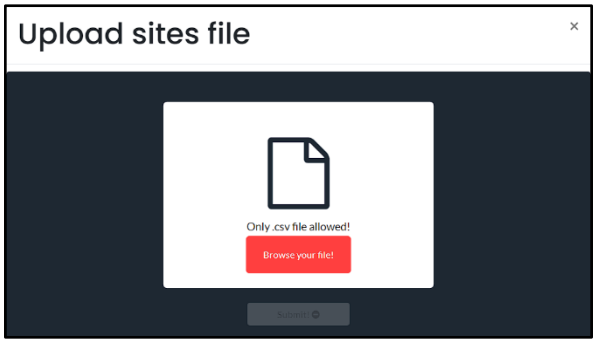
Pada bagian *manage sites* digunakan untuk *user* mengunggah data berubah *list site* atau *list tower* yang akan dimasukkan ke sistem *project management*. Untuk alur kerjanya dapat dilihat pada *flowchart upload sites* (Gambar 3.13) Pertama-tama *user* akan mengunduh template.csv yang sudah disediakan dengan menekan *button Download csv template* (Gambar 3.14), dimana *file* .csv tersebut akan digunakan sebagai *template* untuk memasukkan data *site* agar sesuai dengan yang sudah diatur pada sistem.



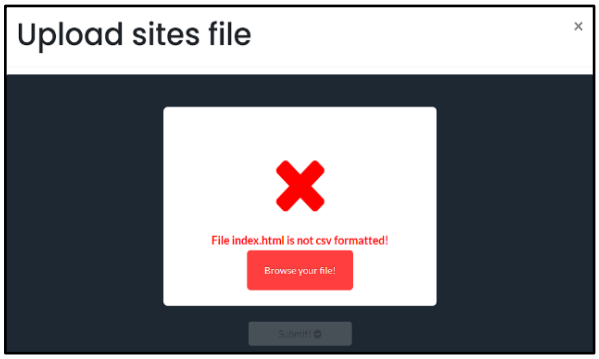
Gambar 3.13 *Flowchart Upload Sites*

Gambar 3.14 *Donwload CSV Template* dan *Upload New Site*

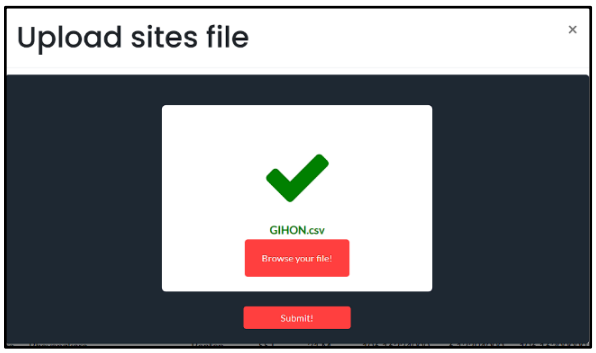
Setelah *user* memasukkan data-data *tower* sesuai dengan *template* tersebut, maka bisa langsung di-*upload* ke sistem dengan menekan *button Upload new site*, maka akan keluar tampilan seperti Gambar 3.15, lalu user dapat mengunggah *file* .csv yang sudah disiapkan. Jika *file* tersebut bukan merupakan *file* dengan ekstensi .csv maka akan ditolak oleh sistem (Gambar 3.16) sehingga tidak bisa diunggah, sedangkan jika *file* tersebut sesuai maka akan bisa diunggah (Gambar 3.17).



Gambar 3.15 Unggah *File* CSV

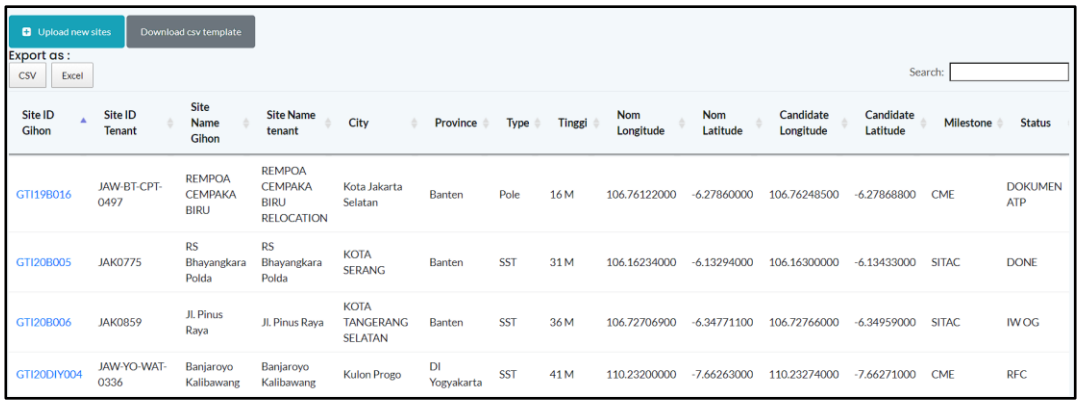


Gambar 3.16 Ekstensi *File* Salah



Gambar 3.17 Ekstensi *File* Benar

Jika berhasil *submit*, maka data-data *site* yang berada di *file* .csv tersebut berhasil diunggah ke sistem (Gambar 3.18).



Site ID Gihon	Site ID Tenant	Site Name Gihon	Site Name tenant	City	Province	Type	Tinggi	Nom Longitude	Nom Latitude	Candidate Longitude	Candidate Latitude	Milestone	Status
GTI19B016	JAW-BT-CPT-0497	REMPOA CEMPAKA BIRU	REMPOA CEMPAKA BIRU RELOCATION	Kota Jakarta Selatan	Banten	Pole	16 M	106.76122000	-6.27860000	106.76248500	-6.27868800	CME	DOKUMEN ATP
GTI20B005	JAK0775	RS Bhayangkara Polda	RS Bhayangkara Polda	KOTA SERANG	Banten	SST	31 M	106.16234000	-6.13294000	106.16300000	-6.13433000	SITAC	DONE
GTI20B006	JAK0859	Jl. Pinus Raya	Jl. Pinus Raya	KOTA TANGERANG SELATAN	Banten	SST	36 M	106.72706900	-6.34771100	106.72766000	-6.34959000	SITAC	IW OG
GTI20DIY004	JAW-YO-WAT-0336	Banjaroyo Kalibawang	Banjaroyo Kalibawang	Kulon Progo	DI Yogyakarta	SST	41 M	110.23200000	-7.66263000	110.23274000	-7.66271000	CME	RFC

Gambar 3.18 Contoh *File Site* Berhasil Diunggah

Proses dimana sistem memasukkan data-data dari *file* .csv tersebut ke dalam sistem ada pada *controller Sales.php*. Dimana pada *controller* tersebut terdapat *function Upload Site Data* (Gambar 3.19).

```php
public function upload_site_data(){
    $session = session();
    if($session->has('Nama') == Null){
        return redirect()->to(base_url(''));
    }
    $sales = new \App\Models\SalesModel();

    $filename = $_FILES["fileup"]["tmp_name"];
    $f_pointer=fopen($filename,"r"); // file pointer
    $i = 1; // remove column title //
    $Message = "Failed to insert ";
    $Duplicated = false;
    while(!feof($f_pointer)){
        $fileData = fgetcsv($f_pointer,'',';');
        if($fileData != null && $i != 1){

            $duplicatedRow = $sales->checkDuplicate($fileData[0]);

            if($duplicatedRow > 0){
                $Message .= $fileData[0].", ";
                $Duplicated = True;
            }
            else{
                $data = [
                    'Site_id_gihon' => $fileData[0],
                    'Site_id_tenant' => $fileData[1],
                    'Site_name_gihon' => $fileData[2],
                    'Site_name_tenant' => $fileData[3],
                    'City' => $fileData[4],
                    'Province' => $fileData[5],
                    'Tower_Type' => $fileData[6],
                    'Tinggi_tower' => $fileData[7],
                    'Nom_long' => str_replace("," , "." , $fileData[8]),
```

```php
                    'Nom_long' => str_replace("," , "." , $fileData[8]),
                    'Nom_lat' => str_replace("," , "." , $fileData[9]),
                    'Candidate_long' => str_replace("," , "." , $fileData[10]),
                    'Candidate_lat' => str_replace("," , "." , $fileData[11]),
                    'Milestone' => $fileData[12],
                    'Status' => $fileData[13],
                    'Partner' => $fileData[14],
                    'Operator' => $fileData[15],
                    'Harga' => $fileData[16],
                    'SPK_Type' => $fileData[17],
                    'SPK_Date' => date('Y-m-d',strtotime($fileData[18])),
                    'Deadline_rfi' =>$fileData[19],
                    'Last_update' => $_POST['last_update']
                ];
                $sales->insertNewSites($data);
            }
        }
        $i++;
    }
    if($Duplicated){
        $Message .= "Site Already Exist";
        session()->setFlashdata('Existing Data', $Message);
    }
    return redirect()->to(base_url('/sales'));
}
```

Gambar 3.19 *Function Upload Site Data*

Pertama dilakukan *check* apakah *session user* masih ada atau tidak, jika tidak ada maka akan dialihkan ke halaman *login*, lalu dengan menggunakan $_FILES, [“fileup”] yang merupakan *parameter* dari *view* yaitu *file*

upload_site.php (Gambar 3.20) yang akan diterima ketika *user* menunggah *file* tersebut Lalu [“tmp_name”] adalah nama sementara yang diberikan kepada *file* tersebut di *backend*. $f_pointer adalah *variable pointer* yang akan berisi *file* mana yang akan dibuka. Lalu karena yang ingin dimasukkan kedalam sistem hanya data-data sites nya atau tanpa *header* maka, header akan di-*remove* dengan cara inisialisasi *variable* i dari 1, sehingga tidak akan membaca header, karena posisi *header* adalah i = 0. Lalu *set variable* $Duplicated untuk melihat apakah ada *site* yang sama. Jika ada *site* yang sama maka sistem akan menolak untuk memasukkan data dan akan mengeluarkan *error message site* mana saja yang sama. Jika tidak ada yang sama maka data akan dimasukkan ke dalam sistem, dimana sebelumnya sudah ada *variable* $sales yang berisi *SalesModel* (Gambar 3.21) yang akan digunakan untuk *query* ke *database*.

```
<form name="upload" method="post" action="<?= base_url('Sales/upload_site_data')?>" enctype="multipart/form-data" accept-charset="utf-8">
    <input type="hidden" name="last_update" value="<?= session()->get('Nama')?>"/>
    <div class="row">
        <div class="col-md-6 col-md-offset-3 center" style="margin:auto;">
            <div class="btn-container">
                <!--the three icons: default, ok file (img), error file (not an img)-->
                <h1 class="imgupload"><i class="fa fa-file-o"></i></h1>
                <h1 class="imgupload ok"><i class="fa fa-check"></i></h1>
                <h1 class="imgupload stop"><i class="fa fa-times"></i></h1>
                <!--this field changes dinamically displaying the filename we are trying to upload-->
                <p id="namefile">Only .csv file allowed!</p>
                <!--our custom btn which which stays under the actual one-->
                <button type="button" id="btnup" class="btn btn-primary btn-lg">Browse your file!</button>
                <!--this is the actual file input, is set with opacity=0 beacause we wanna see our custom one-->
                <input type="file" value="" name="fileup" id="fileup">
            </div>
        </div>
    </div>
        <!--additional fields-->
    <div class="row">
        <div class="col-md-12">
            <!--the defauld disabled btn and the actual one shown only if the three fields are valid-->
            <input type="submit" value="Submit!" class="btn btn-primary" id="submitbtn">
            <button type="button" class="btn btn-default" disabled="disabled" id="fakebtn">Submit! <i class="fa fa-minus-circle"></i></button>
        </div>
    </div>
</form>
```

Gambar 3.20 Form Upload Site Data

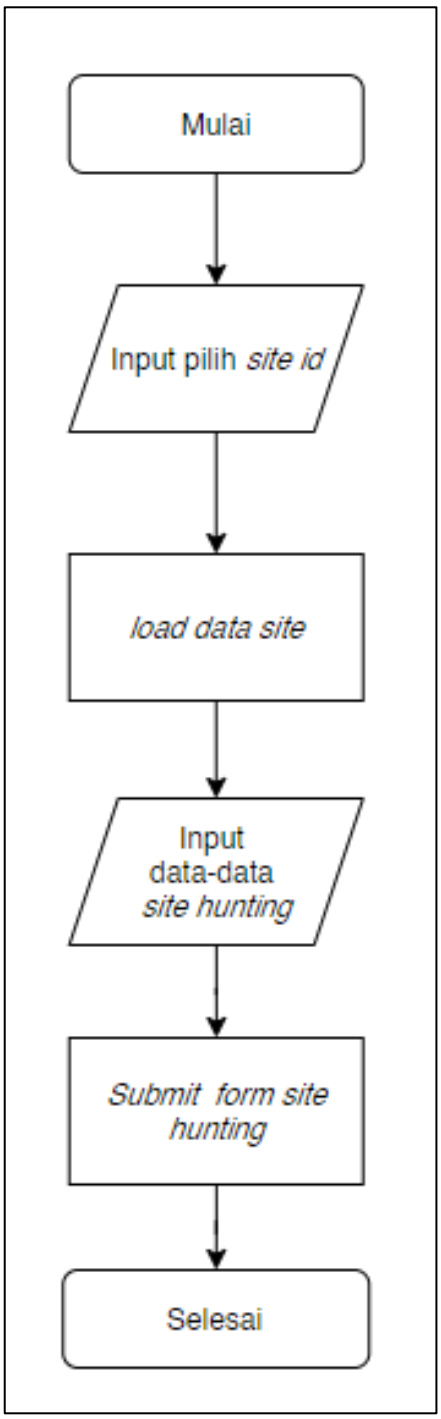
```php
<?php namespace App\Models;

use CodeIgniter\Model;

class SalesModel extends Model
{
    protected $table = 'tower_sites';
    protected $primaryKey = 'Site_id_gihon';
    protected $allowedFields = ['Site_id_gihon', ···
    public function getAllSites()
    {
        return $this->findAll();
    }

    public function insertNewSites($data){
        return $this->insert($data);
    }

    public function updateSites($siteId,$data){
        return $this->update($siteId,$data);
    }
    public function checkDuplicate($siteID){ ···
    }

    public function getOperator(){ ···
    }

    public function getMilestone(){ ···
    }
    public function getCandidate($siteId)
    { ···
    }
}
```

Gambar 3.21 *SalesModel*

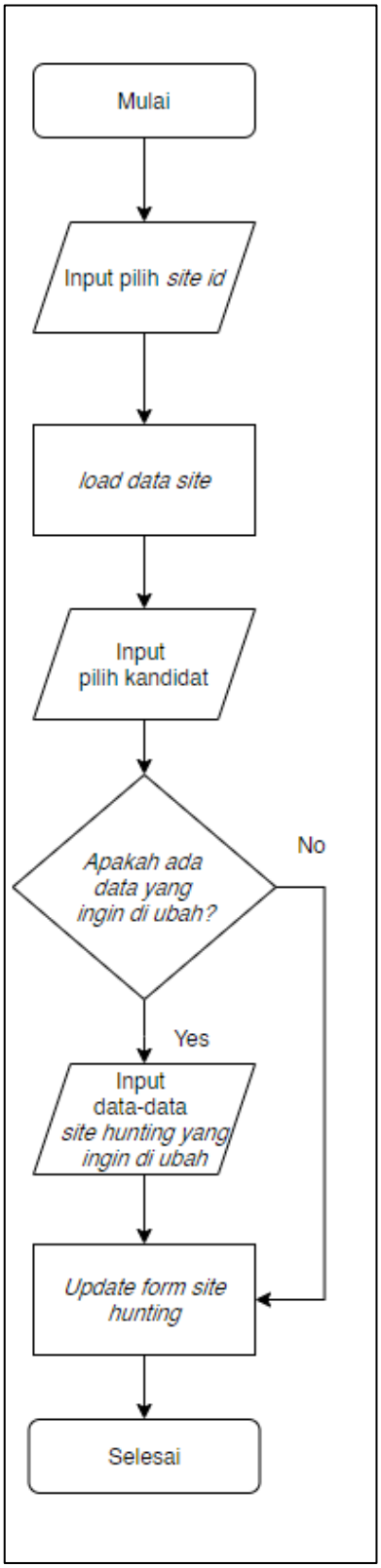
D. *Form Site Hunting*

Sub milestone site hunting adalah proses dimana subkontraktor akan mencari lokasi untuk membangun *site* atau *tower* yang cocok dengan lokasi *tower* yang diminta oleh *provider* atau masih dalam radius lokasi yang di minta *provider*. Untuk alur kerjan ketika membuat *form* baru dapat dilihat di *flowchart new form site* hunting (Gambar 3.22), sementara untuk melihat alur kerja ketika melihat atau melakukan *edit* dari form site hunting ada di *flowchart view site hunting* (Gambar 3.23). Ketika melakukan pencarian lokasi suatu *site*, bisa terdapat banyak kandidat, *Form site hunting* akan digunakan untuk menyimpan data-data kandidat untuk *tower* tersebut. *Form site hunting* dapat diakses melalui *menu Project Management*

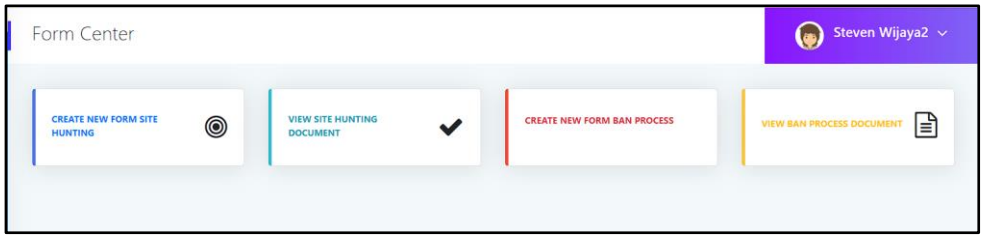
lalu ke *Form Center* (Gambar 3.14) dan *Create new form site hunting* (Gambar 3.25).



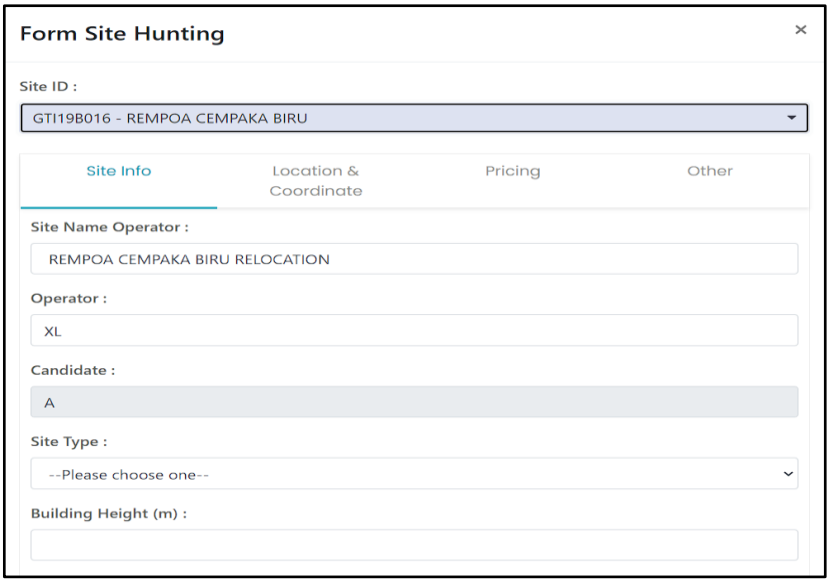
Gambar 3.22 *Flowchart new form site hunting*



Gambar 3.23 *Flowchart view form site hunting*



Gambar 3.24 *Form Center*



Gambar 3.25 *Form Site Hunting*

Kandidat yang ada dalam *form* tersebut akan di-*generate* otomatis dari sistem, apabila sudah ada data kandidat A maka jika ingin membuat *form site hunting* untuk *site* tersebut maka akan berubah secara otomatis menjadi kandidat B. Lalu beberapa data *form* tersebut akan terisi otomatis, dimana data tersebut diambil dari *table manage sites*. Untuk dapat mengambil data-data tersebut *digunakan fetch API* untuk melakukan *query* ke *database* tanpa harus *refresh page* (Gambar 3.26).

```
$("#site_id_hunting").change(function(){
    let currSite = $("#site_id_hunting option:selected").val()
    let today = new Date();
    // fetch data site //
    fetch('<?=base_url("/form/get_form_data")?>',{
            method: "POST",
            headers: {
                'Content-Type': 'application/json',
                "X-Requested-With": "XMLHttpRequest"
            },
            body: JSON.stringify({
                site_id: currSite,
                form_type: "form_hunting"
            })
        })
        .then(response => response.json())
        .then(response => {

            $('input[name=site_name_operator]').val(response["Site_name_tenant"]) // site name operator //
            $('input[name=operator]').val(response["Operator"])                   // operator name //
            $('input[name=rf_height]').val(response["Tinggi_tower"])              // Tinggi Tower //
            $('input[name=city]').val(response["City"])                           // Kota //
            $('input[name=nom_long]').val(response["Nom_long"])                   // Nom Long //
            $('input[name=nom_lat]').val(response["Nom_lat"])                     // Nom Lat //
            $('input[name=tssr_date]').val(today.getFullYear()+'-'+(today.getMonth()+1)+'-'+today.getDate()) // today date //
        })
        .catch(function(err) {
            console.log('Fetch Error :-S', err);
        });
```

Gambar 3.26 *Fetch Site Data*

Fetch API akan memanggil *controller Form* kemudian *function get_form_data* (Gambar 3.27) bagian *Site Hunting* dimana *function* tersebut akan melakukan *query* ke *database* untuk mengambil data dari *tower sites* yang sesuai dengan *site id* yang di kirim oleh *fetch API*, data yang dikembalikan dari *function get_form_data* dalam bentuk JSON. Lalu data tersebut akan dimasukkan menjadi *value* dari beberapa *input* yang ada di *Create new form site hunting*. Ketika user memasukkan semua data *form*, maka *controller form* akan memasukkan data-data tersebut ke *database* lewat *function new_form* (Gambar 3.28) yang berada di *controller Form*.

```php
public function get_form_data(){
    $sales = new \App\Models\SalesModel();
    $candidate = new \App\Models\CandidateModel();
    $ban = new \App\Models\FormBanModel();

    $contentType = isset($_SERVER["CONTENT_TYPE"]) ? trim($_SERVER["CONTENT_TYPE"]) : '';

    if ($contentType === "application/json") {
        //Receive the RAW post data.
        $content = trim(file_get_contents("php://input"));

        $decoded = json_decode($content, true); // decode data //
        $site_id = $decoded['site_id']; // data site id //
        $form_type = $decoded['form_type']; // data form type //

        if($form_type === "form_hunting" ){
            $site_data = $sales->where("Site_id_gihon",$site_id)->first();
            echo json_encode($site_data);
        }else if ($form_type === "form_BAN" ){

            if(isset($decoded['candidate'])){
                $site_data = $candidate->where("Site_id_gihon",$site_id)->where("Candidate",$decoded['candidate'])->first();
            }else{
                // query kalo blm pilih kandidat //
                $site_data = $candidate->where("Site_id_gihon",$site_id)->where("Candidate","A")->first();
            }

            if($site_data == null){
                $form_error = " Can't Proceed to Form BAN for site - " . $site_id . ", please finish site hunting process." ;
                session()->setFlashdata('form_error',  $form_error);
                echo json_encode("ERROR");
            }else{
                echo json_encode($site_data);
            }
```

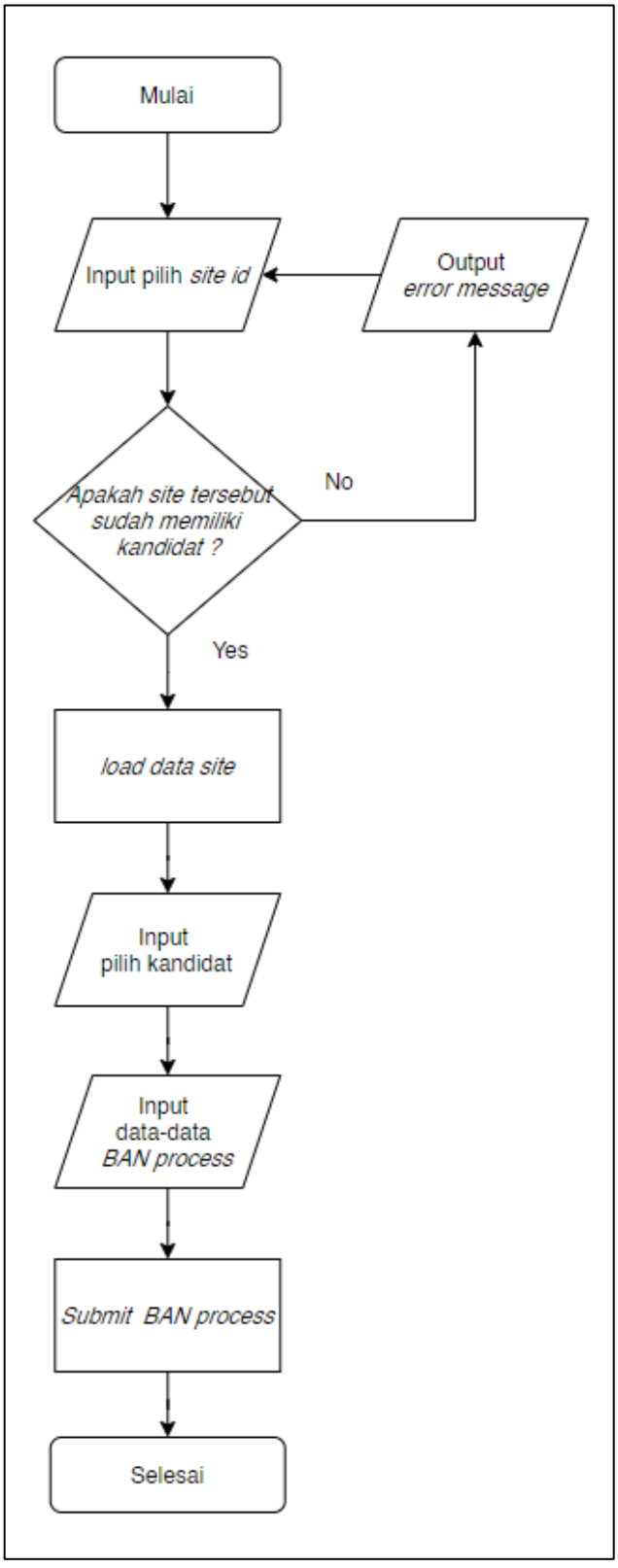
Gambar 3.27 Contoh *Function Get Form Data*

```php
public function new_form(){
    // form hunting //
    if($_POST["form_type"] === "form_hunting"){
        $candidate = new \App\Models\CandidateModel();
        $data = [
            'Site_id_gihon' => $_POST['site_id_hunting'],
            'Site_name_operator' => $_POST['site_name_operator'],
            'Operator' => $_POST['operator'],
            'Candidate' => $_POST['Candidate'],
            'Site_type' => $_POST['site_type'],
            'Building_Height' => $_POST['building_height'],
            'RF_Height' => $_POST['rf_height'],
            'Proposed_Tower_Height' => $_POST['proposed_tower_height'],
            'Land_Ownership_Status' => $_POST['land_ownership_status'],
            'Site_Contour' => $_POST['site_contour'],
            'City' => $_POST['city'],
            'Candidate_Address' => $_POST['candidate_address'],
            'Nom_long' => $_POST['nom_long'],
            'Nom_lat' => $_POST['nom_lat'],
            'Can_long' => $_POST['can_long'],
            'Can_lat' => $_POST['can_lat'],
            'Distance_from_NOM' => $_POST['distance_nom'],
            'Distance_From_Pole_PLN' => $_POST['distance_pole'],
            'BAN_Price' => $_POST['ban_price'],
            'TSSR_Date' => $_POST['tssr_date'],
            'Add_Work_Remark' => $_POST['add_work_remark'],
            'Existing_tp' => $_POST['existing_tp'],
            'Tenant' => $_POST['tenant'],
            'Nearest_existing_tower_height' => $_POST['nearest_existing_tower_height'],
            'Nearest_existing_tower_long' => $_POST['nearest_existing_tower_long'],
            'Nearest_existing_tower_lat' => $_POST['nearest_existing_tower_lat'],
            'Distance_to_nom' => $_POST['distance_to_nom']
        ];
        $form_success = " Form Succesfully submitted " ;
        session()->setFlashdata('form_success',  $form_success);
        $candidate->insertNewCandidate($data);
        return redirect()->to(base_url('form'));
    }
```

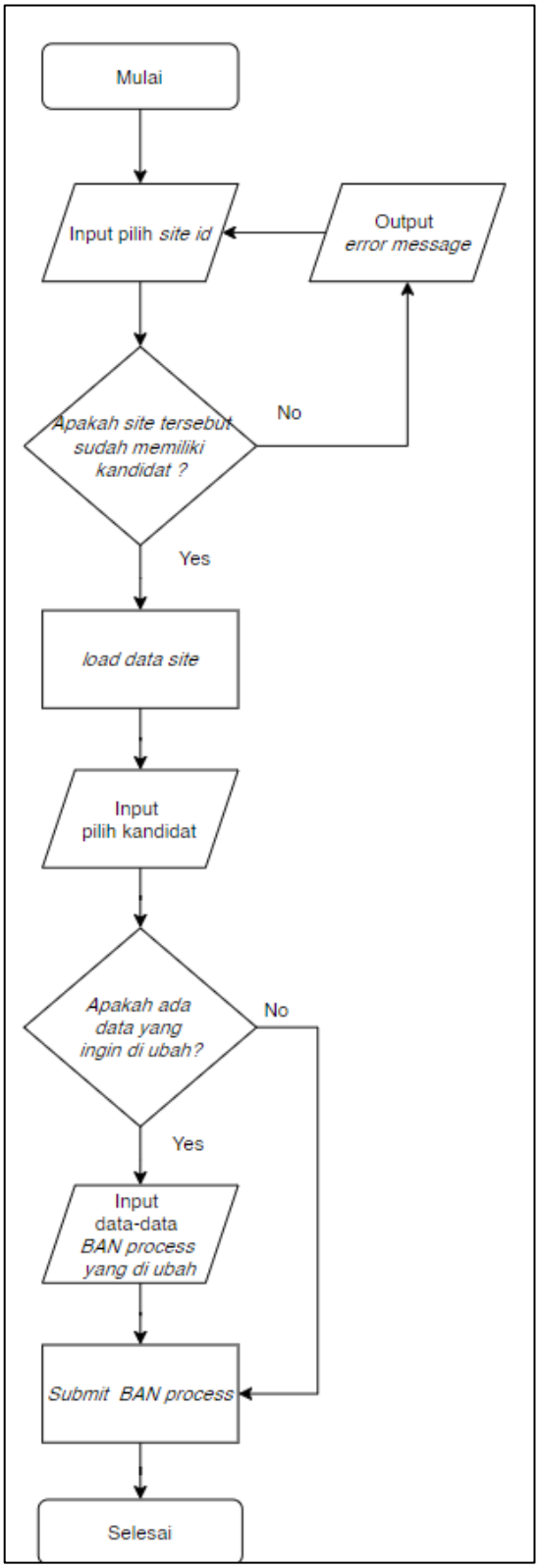
Gambar 3.28 Contoh *Function New Form*

E. *Form BAN Process*

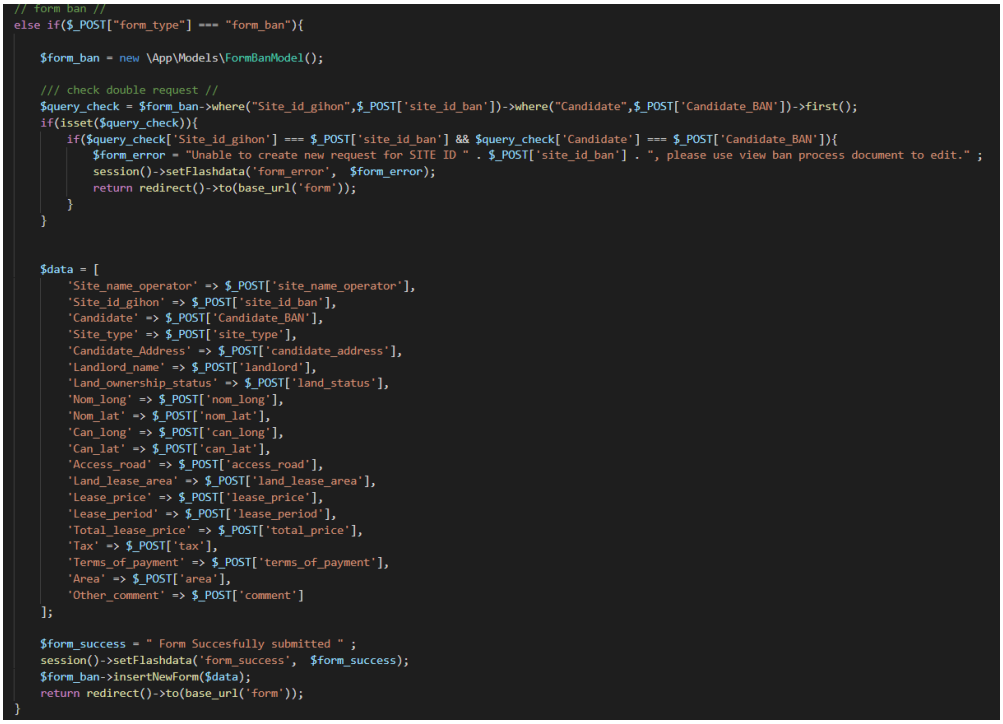
Pada *sub milestone BAN Process*, sama seperti *form* yang digunakan pada *site hunting* hanya data *input* yang berbeda. *Ban Process* adalah tahap dimana subkontraktor akan negosiasi harga sewa lahan dengan pemilik tanah sampai mencapai kesepakatan.Untuk alur kerja ketika ingin membuat *form* baru maka dapat dilihat di *flowchart create new* BAN *process* (Gambar 3.29), sementara ketika ingin melakukan *edit* form tersebut dapat di lihat di *flowchart view form BAN process* (Gambar 3.30). Pada *form Create new form ban process* (Gambar 3.31) akan mengambil data-data kandidat yang sudah ada di *database form site hunting* menggunakan *fetch API* dengan nilai standar kandidat A, lalu ketika *user* memilih kandidat lain, maka *fetch API* akan memanggil *controller Form* lalu *function get_form_data bagian BAN Process* (Gambar 3.27).



Gambar 3.29 *Flowchart Create New Form BAN Process*



Gambar 3.30 *Flowchart View Form BAN Process*

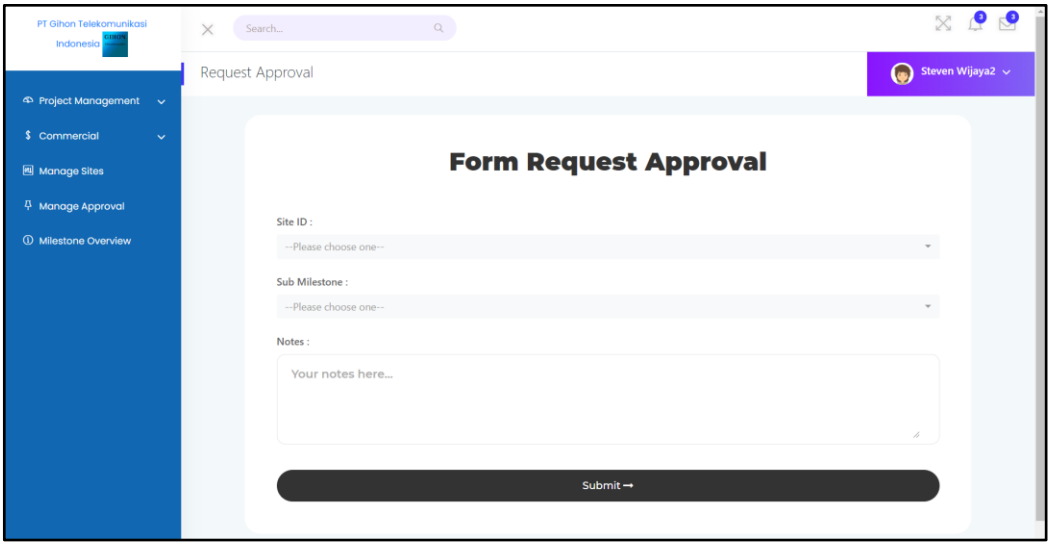


```
// form ban //
else if($_POST["form_type"] === "form_ban"){

    $form_ban = new \App\Models\FormBanModel();

    /// check double request //
    $query_check = $form_ban->where("Site_id_gihon",$_POST['site_id_ban'])->where("Candidate",$_POST['Candidate_BAN'])->first();
    if(isset($query_check)){
        if($query_check['Site_id_gihon'] === $_POST['site_id_ban'] && $query_check['Candidate'] === $_POST['Candidate_BAN']){
            $form_error = "Unable to create new request for SITE ID " . $_POST['site_id_ban'] . ", please use view ban process document to edit." ;
            session()->setFlashdata('form_error',  $form_error);
            return redirect()->to(base_url('form'));
        }
    }


    $data = [
        'Site_name_operator' => $_POST['site_name_operator'],
        'Site_id_gihon' => $_POST['site_id_ban'],
        'Candidate' => $_POST['Candidate_BAN'],
        'Site_type' => $_POST['site_type'],
        'Candidate_Address' => $_POST['candidate_address'],
        'Landlord_name' => $_POST['landlord'],
        'Land_ownership_status' => $_POST['land_status'],
        'Nom_long' => $_POST['nom_long'],
        'Nom_lat' => $_POST['nom_lat'],
        'Can_long' => $_POST['can_long'],
        'Can_lat' => $_POST['can_lat'],
        'Access_road' => $_POST['access_road'],
        'Land_lease_area' => $_POST['land_lease_area'],
        'Lease_price' => $_POST['lease_price'],
        'Lease_period' => $_POST['lease_period'],
        'Total_lease_price' => $_POST['total_price'],
        'Tax' => $_POST['tax'],
        'Terms_of_payment' => $_POST['terms_of_payment'],
        'Area' => $_POST['area'],
        'Other_comment' => $_POST['comment']
    ];

    $form_success = " Form Succesfully submitted " ;
    session()->setFlashdata('form_success',  $form_success);
    $form_ban->insertNewForm($data);
    return redirect()->to(base_url('form'));
}
```

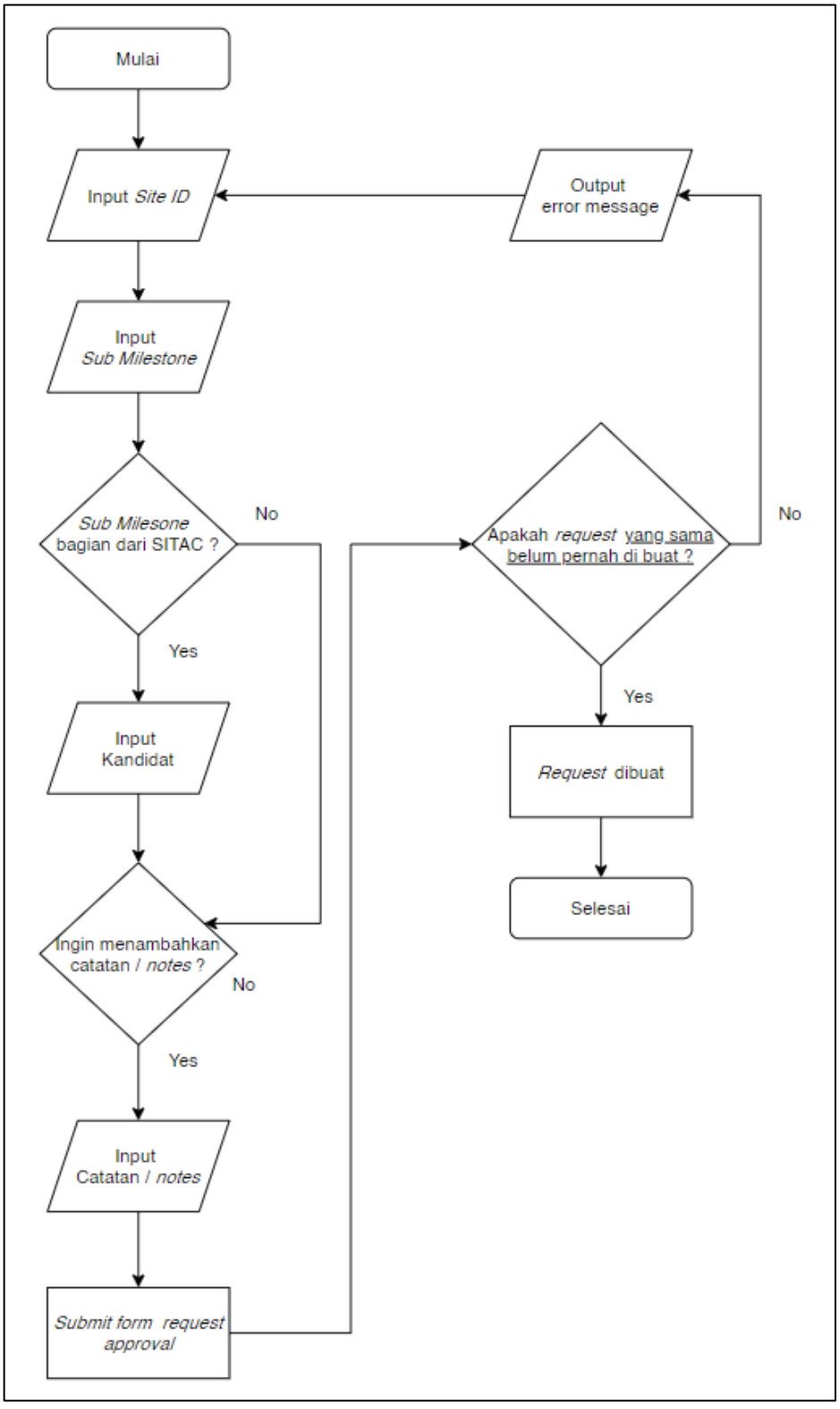
Gambar 3.31 *Create New Form BAN Process*

F. *Request Approval*

Request Approval dapat diakses dari *side menu* dengan memilih *Request Center* dan akan menampilkan seperti berikut (Gambar 3.32). Pada bagian *Request Approval*, akan digunakan *user* untuk melakukan permintaan persetujuan dari atasan untuk *side id* dengan *sub milestone* tertentu. Untuk alur kerjanya dapat dilihat di *flowchart request approval* (Gambar 3.33). *User* dapat melakukan permintaan *approval* setelah *site* atau *tower* dengan tertentu telah memiliki kandidat yang sudah di-*input* melalui *form Site Hunting* (Gambar 3.20).



Gambar 3.32 Halaman *Request Approval*



Gambar 3.33 *Flowchart Request Approval*

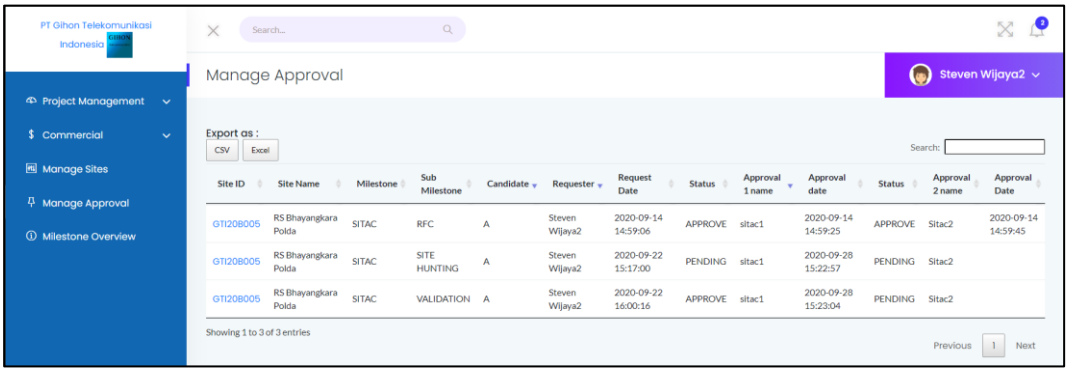
Setelah user memasukkan *site id* dan *sub milestone* tertentu, maka *form* tersebut akan dikirim ke *controller RequestApproval.php* (Gambar 3.34) dimana *controller* tersebut akan menerima *data* dari *user*, lalu akan dibandingkan terlebih dahulu apakah *request* serupa sudah pernah dibuat atau belum, jika belum maka *request* akan berhasil dibuat lalu user tinggal menunggu respon dari atasan atau PIC terkait.

```php
// REQ DATA //
$data = [
    'Site_id_gihon' => $site_id,
    'Site_name_gihon'  => $currSite['Site_name_gihon'],
    'Approval_type' => $milestone,
    'Sub_milestone' =>$sub_milestone,
    'Candidate' => $candidate,
    'Requester' => $requestor,
    'Request_date_time' => date('Y-m-d H:i:s'),
    'Approval_1' => $first_request_condition_approval_1,
    'Approval1_name' =>  $approval_1_name,
    'Approval1_date_time' => null,
    'Approval_2' => $first_request_condition_approval_2,
    'Approval2_name'=>  $approval_2_name,
    'Approval2_date_time' => null,
    'Notes' => $notes
];



$req_success = " Request Success, please wait for response " ;
session()->setFlashdata('req_success',  $req_success);
$approval->insertNewRequest($data);
```

Gambar 3.34 Potongan *Code Controller Request Approval*

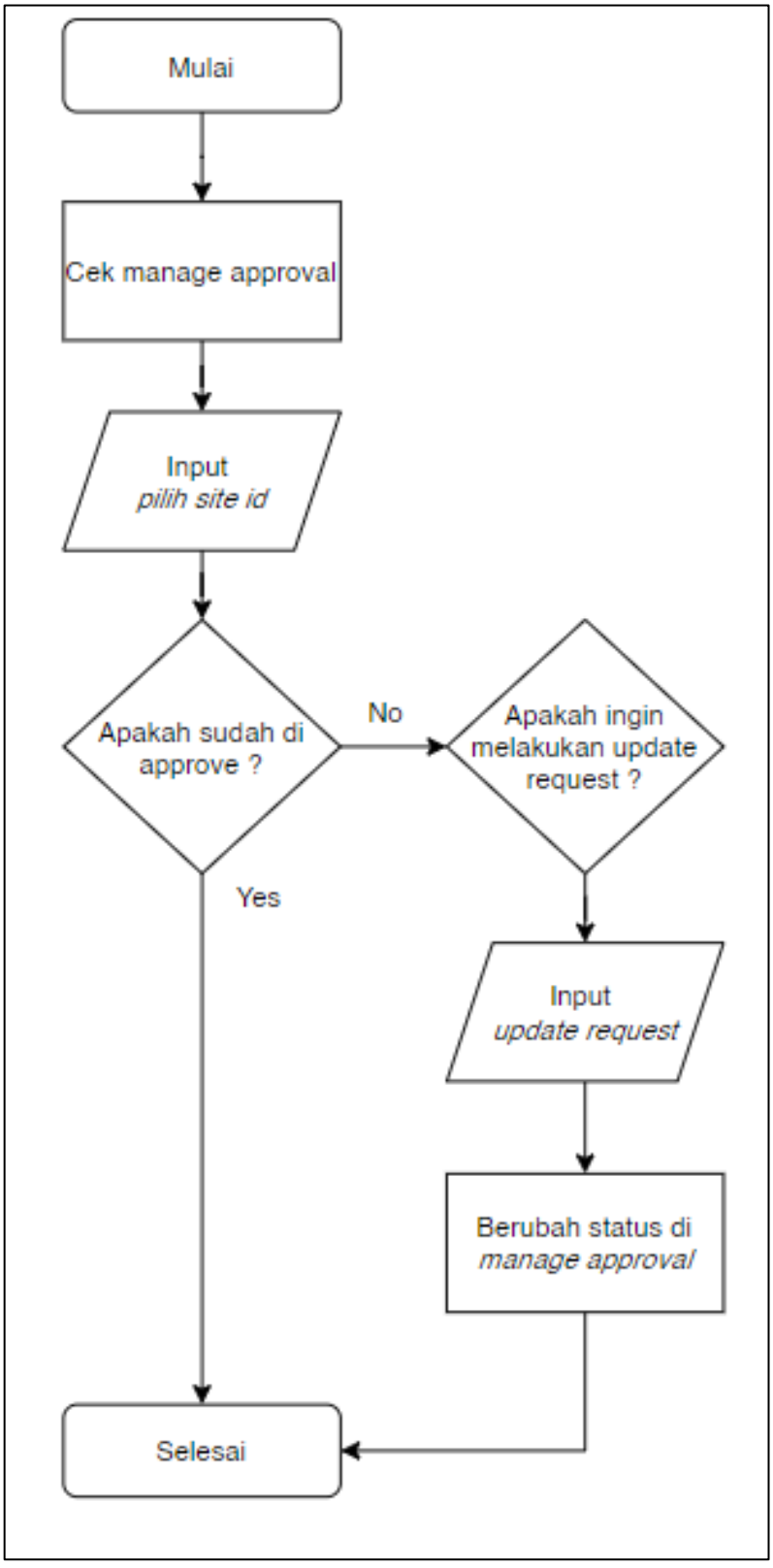
G. *Manage Approval*

Setelah *user* melakukan *request approval*, maka untuk me-*monitoring request* tersebut dapat dilihat di *Manage Approval* (Gambar 3.35). Pada bagian *manage approval* akan menunjukkan apakah *request* tersebut akan di terima atau tolak oleh PIC terkait. Untuk alur kerja proses *manage approval* sisi *requestor* dapat dilihat di *flowchart manage approval* sisi *requestor* (Gambar 3.36) dan untuk kerja proses *manage approval* sisi *approver* dapat dilihat di *flowchart manage approval* sisi *approver* (Gambar 3.37).

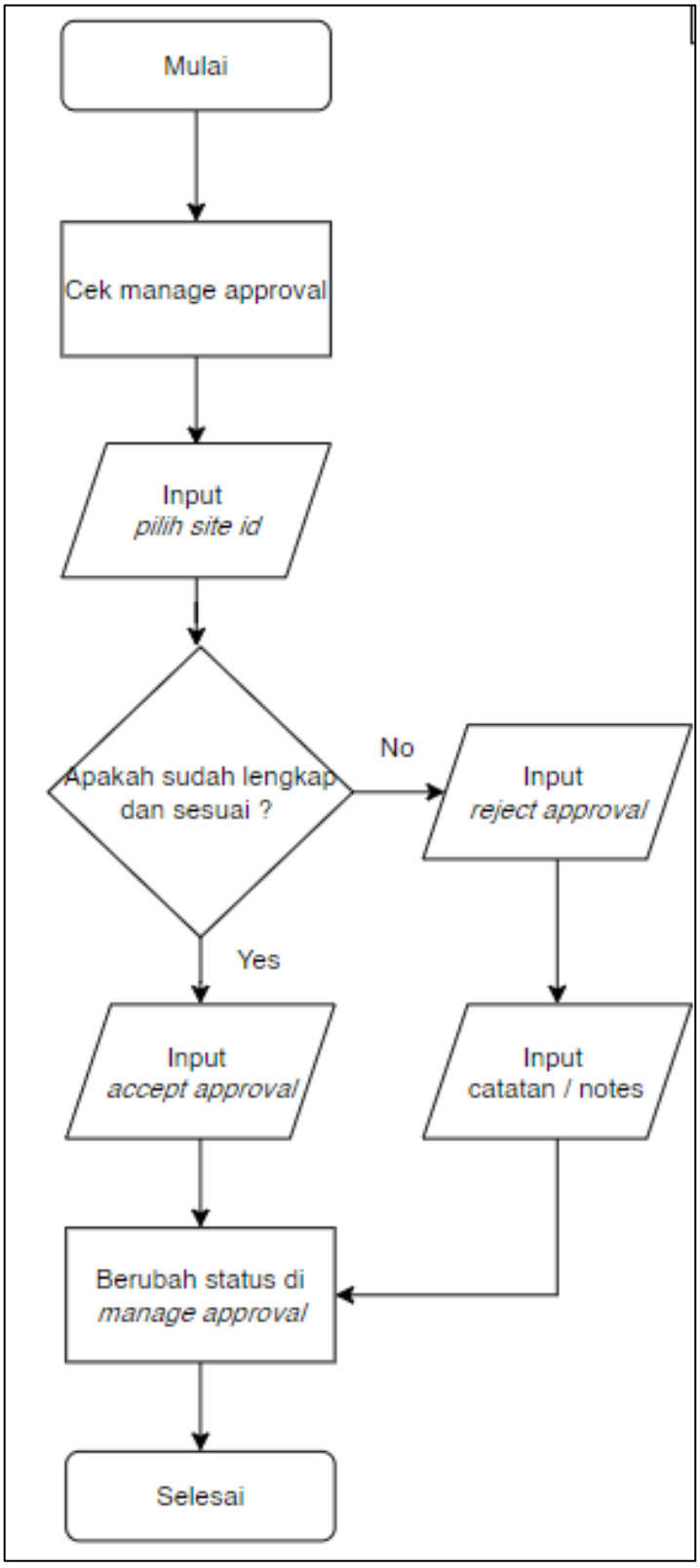


Gambar 3.35 Halaman *Manage Approval*

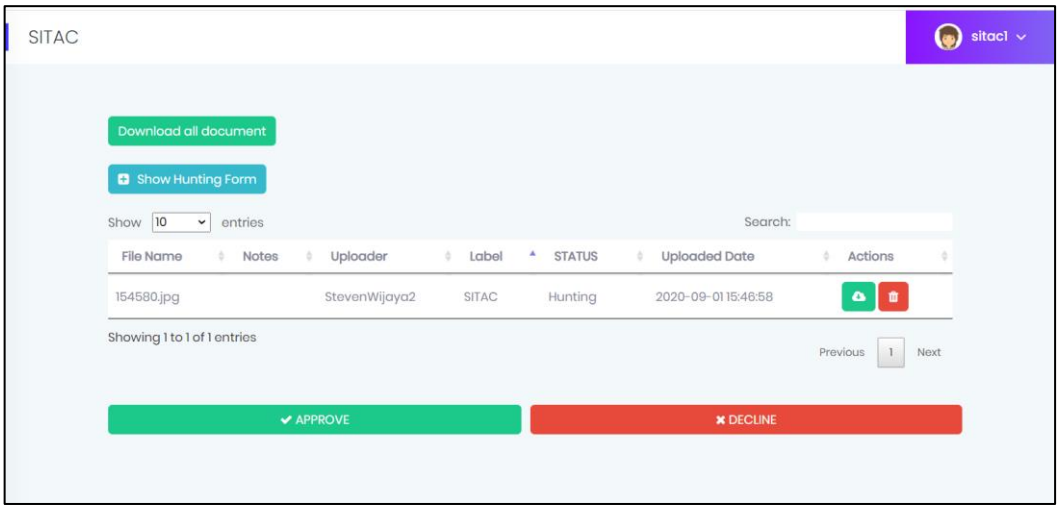
Jika *login* menggunakan akun *approver* atau memiliki *privilege* untuk *approval sub milestone* tersebut, maka ketika *requestor* atau *approver* memilih *site id* di *manage approval*, halaman akan berubah ke data dokumen yang ingin di-*approve* dan *approver* bisa langsung menerima atau menolak *request* tersebut (Gambar 3.38). Apabila diterima maka status akan berubah menjadi "*APPROVE*" sedangkan apabila ditolak maka status akan berubah *menjadi* "*DECLINE*" dan *approver* bisa memberikan *notes* yang akan berisikan alasan *request* tersebut ditolak (Gambar 3.39).



Gambar 3.36 *Flowchart Manage Approval* Sisi *Requestor*



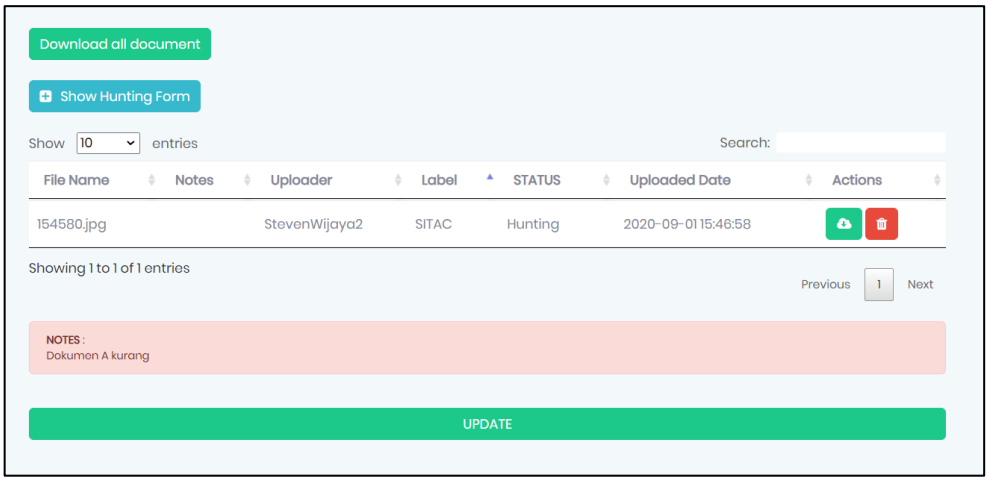
Gambar 3.37 *Flowchart Manage Approval* Sisi *Approver*



Gambar 3.38 Sites Document

Gambar 3.39 *Add Notes Decline Request*

Jika login sebagai requestor, maka *notes* (Gambar 3.40) akan ditampilkan ketika *requestor* membuka halaman *upload* dokumen. Lalu *requestor* dapat melakukan *update request* kembali apabila diperlukan.



Gambar 3.40 Update Request

Ketika *requestor* melakukan *update request* atau *approval* menerima atau menolak *request* maka akan memanggil *controller Manage Approval* (Gambar 3.41). Pada *controller Manage Approval* akan digunakan untuk melakukan *query* ke *database* apabila ada perubahan status di *site* tersebut seperti di-*accept*, *decline*, atau *update*.

```php
<?php
namespace App\Controllers;

class ManageApproval extends BaseController
{
    public function index(){...
    }

    // CHECK APPROVE ATAU DECLINE //
    public function check_approval(){...
    }


   // UPDATE REQUEST KALO MISALNYA DI DECLINE  //
   public function update_request_approval(){...
   }
}
```

Gambar 3.41 *Controller Manage Approval*

Terdapat 2 *function* pada *controller Manage Approval*, *function check_approval* (Gambar 3.42) dan *function update_request_approval* (Gambar

3.43). *Function check_approval* akan digunakan ketika *approval* menerima atau menolak suatu *request* sedangkan *function update_request_approval* akan digunakan ketika *requestor* melakukan *update* atau pengajuan kembali untuk *approval site* tertentu yang sudah pernah diajukan sebelumnya.

```php
// CHECK APPROVE ATAU DECLINE //
public function check_approval(){
    $db      = \Config\Database::connect();
    helper(['form', 'url']);
    $approval = new \App\Models\ManageApproval();
    $sites_list = new \App\Models\SalesModel();
    $candidate = new \App\Models\CandidateModel();
    $email = new \App\Controllers\Email();
    date_default_timezone_set("Asia/Bangkok");
    $approval_id = $db->escapeString($_POST['approval_id']);
    $status = $db->escapeString($_POST['request']);
    $currApproval = $db->escapeString($_POST['currApproval']);

    //read sites data on table manage_approval
    $query = " select Site_id_gihon,Approval_type,Candidate,Sub_milestone from manage_approval where Approval_id = ?";
    $execute = $db->query($query,$approval_id);

    //iterate the result. cause CI4 result is array object
    foreach ($execute->getResultArray() as $result)
    {
        $site_id_gihon = $result['Site_id_gihon'];
        $approval_type = $result['Approval_type'];
        $choosenCandidate = $result['Candidate'];
        $sub_milestone = $result['Sub_milestone'];
    }


    //cek approval 1 apa approval 2 //
    if( $currApproval == "Approval_1" ){
        $approval_date_time = "Approval1_date_time";

    }else{
        $approval_date_time = "Approval2_date_time";
    }

    // DECLINE NOTES //
    if(isset($_POST['approval_note'])){
        $approval_note = $db->escapeString($_POST['approval_note']);
    }else{
        $approval_note = "";
    }


    // data skrng //
    $data =[
        $approval_date_time => date('Y-m-d H:i:s'),
        $_POST['currApproval'] => $status,
        'Notes' => $approval_note
    ];
```

Gambar 3.42 Potongan *Code Function Check_approval*

```php
public function update_request_approval(){
    $db      = \Config\Database::connect();
    helper(['form', 'url']);
    $approval = new \App\Models\ManageApproval();
    $req_approval = $db->escapeString($_POST['req_approval']);
    date_default_timezone_set("Asia/Bangkok");

    if( $_POST['req_approval'] == "Approval_1" ){
        $approval_date_time = "Approval1_date_time";
    }else{
        $approval_date_time = "Approval2_date_time";
    }

    $approval_id = $db->escapeString($_POST['approval_id']);
    $data = [
        $approval_date_time => null,
        'Request_date_time' => date('Y-m-d H:i:s'),
        $req_approval => 'PENDING',
        'Notes' => ''
    ];

    $approval->updateRequest($approval_id, $data);
    return redirect()->to(base_url('ManageApproval'));
}
```

Gambar 3.43 Potongan *Code Function update_request_approval*

3.3.2. Kendala yang Ditemukan

Dalam proses pelaksanaan kerja magang, terdapat beberapa kendala yang dihadapi. Kendala-kendala yang ditemukan adalah antara lain seperti berikut:

1. *User Requirement* yang berubah-berubah, sehingga perlu mengganti alur dari sistem kembali.
2. Beberapa konsep *programming* seperti *Fetch API* dan penggunaan *Composer* untuk mengunduh beberapa *library* tertentu.

3.3.3 Solusi Atas Kendala yang Ditemukan

Dari kendala-kendala yang ditemukan selama proses kerja magang, terdapat beberapa solusi yang ditemukan untuk menyelesaikan kendala tersebut agar tidak menghambat pengerjaan tugas kerja magang. Solusi dari kendala-kendala yang ditemukan adalah antara lain sebagai berikut.

1 Melakukan *requirement gathering*, sehingga tidak ada perubahan secara besar pada sistem yang akan dibuat (hanya penambahan fitur baru atau *fixing bugs*).

2 Mempelajari cara penggunaan *fetch API* dan *Composer* dari berbagai *website* seperti Stackoverflow dan Youtube.