LAMPIRAN

1. Route

```
<?php

use App\Http\Controllers\DataController;
use App\Http\Controllers\DeviceController;
use App\Http\Controllers\LoginController;
use App\Http\Controllers\RegisterController;
use App\Http\Livewire\Content;
use Illuminate\Support\Facades\Route;
use Livewire\Livewire;
use App\Http\Controllers\OverviewController;
use App\Http\Controllers\StateController;
use App\Http\Controllers\UserController;
use App\Models\Device;
use GuzzleHttp\Middleware;

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "web" middleware group. Make something great!
|
*/

Route::redirect('/', '/dashboard');
Route::get('/dashboard', [DataController::class, 'index'])->middleware('auth')-
>name('dashboard');
Route::get('/overview', [OverviewController::class, 'index'])->middleware('auth');
Route::get('/login', [LoginController::class, 'index'])->name('login')-
>middleware('guest');
Route::get('/', [LoginController::class, 'index'])->name('login')-
>middleware('guest');
Route::post('/login', [LoginController::class, 'authenticate']);
Route::post('/logout', [LoginController::class, 'logout']);

Route::get('/register', [RegisterController::class, 'index'])->middleware('guest');
Route::post('/register', [RegisterController::class, 'store']);

Route::post('/data',[DataController::class,'store']);
Route::resource('/state',StateController::class);
```

```
Route::get('/setting', [UserController::class, 'index'])->middleware('auth')-
>name('setting');
Route::get('/setting/changeprofile', [UserController::class, 'editProfile'])-
>middleware('auth')->name('editProfile');
Route::put('/setting/changeprofile', [UserController::class, 'update'])-
>name('updateProfile');
Route::put('/setting/changepassword', [UserController::class, 'update'])-
>name('updatePassword');
Route::get('/setting/changepassword', [UserController::class, 'editPassword'])-
>middleware('auth')->name('editPassword');
Route::post('/setting/add', [DeviceController::class, 'store'])-
>name('storeDevices');
Route::get('/setting/add', [DeviceController::class, 'addDevices'])-
>middleware('auth')->name('addDevices');
```

2. LoginController

```
<?php

namespace App\Http\Controllers;

use Illuminate\Support\Facades\Auth;
use App\Http\Controllers\Controller;
use Illuminate\Http\Request;

class LoginController extends Controller
{
    public function index()
    {
        return view('login.index', [
            'title' => 'Login',
        ]);
    }

    public function authenticate(Request $request)
    {
        $credential = $request->validate([
            'email' => 'email:rfc,dns|required',
            'password'=> 'required'
        ]);

        if (Auth::attempt($credential)) {
            // regenerate digunakan untuk keamanan dari "session fixation"
            $request->session()->regenerate();

            return redirect()->intended('/dashboard');
        }

        return back()->with('loginError', 'Login Failed!');
    }

    public function logout(Request $request)
    {
        Auth::logout();

        $request->session()->invalidate();
        $request->session()->regenerateToken();
        return redirect('/login');
    }
}
```

3. Livewire-Content

```
<?php

namespace App\Http\Livewire;

use App\Models\data;
use App\Models\Device;
use App\Models\State;
use Livewire\Component;
use Illuminate\Queue\Listener;
use Illuminate\Support\Facades\DB;

class Content extends Component
{
    public $data_terakhir, $chartsoh = [], $chartsoc = [], $chartarus = [],
$chartvoltage = [], $chartsuhu = [], $chartdaya = [];
    protected $listeners = ['update' => 'change'];
    public $product, $waktuAll;
    public $selectDevice;


    public function mount()
    {
        if ($this->selectDevice == null) {
            $this->selectDevice = $this->device();
        }

        $data = data::where('device_token', $this->selectDevice)->latest()->take(20)-
>get();

        $i = 0;
        if ($data->count() > 0) {
            foreach ($data as $datas) {
                $waktu[$i] = date_create($datas->updated_at);
                $waktu[$i] = date_format($waktu[$i], 'd-m-Y H:i ');
                $voltage[$i] = $datas->voltage;
                $daya[$i] = $datas->daya;
                $suhu[$i] = $datas->suhu;
                $arus[$i] = $datas->current;
                $soh[$i] = $datas->soh;
                $soc[$i] = $datas->soc;
                $i++;
            }
        }
        else {
            $waktu = [];
            $voltage = [];
```

```
            $daya = [];
            $suhu = [];
            $arus = [];
            $soh = [];
            $soc = [];
        }

        $this->chartsoc = json_encode([
            'labels' => ($waktu),
            'data' => ($soc)
        ]);
        $this->chartsoh = json_encode([
            'labels' => ($waktu),
            'data' => ($soh)
        ]);
        $this->chartarus = json_encode([
            'labels' => ($waktu),
            'data' => ($arus)
        ]);
        $this->chartvoltage = json_encode([
            'labels' => ($waktu),
            'data' => ($voltage)
        ]);
        $this->chartsuhu = json_encode([
            'labels' => ($waktu),
            'data' => ($suhu)
        ]);
        $this->chartdaya = json_encode([
            'labels' => ($waktu),
            'data' => ($daya)
        ]);

        $this->data_terakhir = data::where('device_token', $this->selectDevice)-
>get()->last();
        if (optional($this->data_terakhir)->exists) {
            // Data ditemukan, gunakan nilai aktual
            $waktu = date_create($this->data_terakhir->updated_at);
            $waktu = date_format($waktu, 'd-m-Y H:i');
            $voltage = $this->data_terakhir->voltage;
            $daya = $this->data_terakhir->daya;
            $suhu = $this->data_terakhir->suhu;
            $arus = $this->data_terakhir->current;
            $soh = $this->data_terakhir->soh;
            $soc = $this->data_terakhir->soc;
        } else {
            // Data tidak ditemukan, set nilai default sebagai tanda bahwa data tidak
ada
            $waktu = 'Data tidak ditemukan';
```

```
            $voltage = null;
            $daya = null;
            $suhu = null;
            $arus = null;
            $soh = null;
            $soc = null;
        }

        // dd( $this->chartsuhu);
        $this->waktuAll = json_encode($waktu);
    }

    public function device()
    {
        $data = Device::where('user_id', auth()->user()->id)->get();
        $i = 0;
        foreach ($data as $datum) {
            $token[$i] = $datum->device_token;
            $i++;
        }
        if ($data->count() == 0) {
            return null;
        }
        return $token[0];
    }

    public function render()
    {
        return view('livewire.content', [
            'state'   => State::where('id', 1)->get(),
            'devices' => Device::where('user_id', auth()->user()->id)->get()
        ]);
    }

    public function change()
    {
        $this->mount();
        $this->emit('chartDataUpdateded', [
            'dataSuhu' => $this->chartsuhu,
            'dataSOC' => $this->chartsoc,
            'dataSOH' => $this->chartsoh,
            'dataArus' => $this->chartarus,
            'dataVoltage' => $this->chartvoltage,
            'dataDaya' => $this->chartdaya,

            'waktuAll' => $this->waktuAll

        ]);
```

```
    }

    public function saveCheckboxValue($name)
    {
        $state = State::find(1);
        if ($name == 'discharge') {
            if ($state->discharge == 1) {
                $state->discharge = 0;
            } else {
                $state->discharge = 1;
            };
        } else {
            if ($state->charge == 1) {
                $state->charge = 0;
            } else {
                $state->charge = 1;
            };
        }
        $state->save();
    }
}
```

4. Livewire-overview

```
<?php

namespace App\Http\Livewire;

use App\Models\data;
use App\Models\Device;
use Illuminate\http\Request;
use Livewire\Component;
use App\Models\Item;
use Carbon\Carbon;

class Overview extends Component
{
    public $data_terakhir;
    // public $chartsoh = [];
    // public $chartsoc = [];
    // public $chartarus = [];
    // public $chartdaya = [];
    // public $chartvoltage = [];
    // public $chartsuhu = [];

    public $selectDevice;
    public $chartdata = [];
```

```
    public $sumvoltage = [];
    public $waktu;
    public $pilihan = "voltage";
    public $pilihan_waktu = "hari";
    protected $listeners = ['ubah' => 'change'];

    public function mount()
    {
        if ($this->selectDevice == null) {
            $this->selectDevice = $this->device();
        }

        if ($this->pilihan_waktu == "hari") {
            $data = Data::where('device_token', $this->selectDevice)-
>whereDate('updated_at', today())->get();
        } elseif ($this->pilihan_waktu == "minggu") {
            $data = Data::where('device_token', $this->selectDevice)-
>whereBetween('updated_at', [Carbon::now()->subWeek(),Carbon::now()])-
>get();
        } elseif ($this->pilihan_waktu == "bulan") {
            $data = Data::where('device_token', $this->selectDevice)-
>whereBetween('updated_at', [Carbon::now()->subMonth(),Carbon::now()])-
>get();
        }

        $i = 0;
        if ($data->count() != 0) {
            foreach ($data as $datas) {
                $waktu[$i] = date_create($datas->updated_at);
                if ($this->pilihan_waktu == "hari") {
                    $waktu[$i] = date_format($waktu[$i], 'H:i:s');
                } elseif ($this->pilihan_waktu == "minggu") {
                    $waktu[$i] = date_format($waktu[$i], 'd/m');
                } elseif ($this->pilihan_waktu == "bulan") {
                    $waktu[$i] = date_format($waktu[$i], 'd/m');
                }
                $voltage[$i] = $datas->voltage;
                $daya[$i] = $datas->daya;
                $suhu[$i] = $datas->suhu;
                $arus[$i] = $datas->current;
                $soh[$i] = $datas->soh;
                $soc[$i] = $datas->soc;
                $i++;
            }
        } else {
            $waktu[$i] = 0;
            $voltage[$i] = 0;
            $daya[$i] = 0;
```

```
            $suhu[$i] = 0;
            $arus[$i] = 0;
            $soh[$i] = 0;
            $soc[$i] = 0;
        }
        $this->waktu = $waktu;
        if ($this->pilihan == "voltage") {
            $this->chartdata = json_encode([
                'labels' => $waktu,
                'data'   => $voltage
            ]);
        } elseif ($this->pilihan == "daya") {
            $this->chartdata = json_encode([
                'labels' => $waktu,
                'data'   => $daya
            ]);
        } elseif ($this->pilihan == "suhu") {
            $this->chartdata = json_encode([
                'labels' => $waktu,
                'data'   => $suhu
            ]);
        } elseif ($this->pilihan == "soc") {
            $this->chartdata = json_encode([
                'labels' => $waktu,
                'data'   => $soc
            ]);
        } elseif ($this->pilihan == "soh") {
            $this->chartdata = json_encode([
                'labels' => $waktu,
                'data'   => $soh
            ]);
        } elseif ($this->pilihan == "current") {
            $this->chartdata = json_encode([
                'labels' => $waktu,
                'data'   => $arus
            ]);
        }

        $this->sumvoltage = array_sum($voltage);
        $this->data_terakhir = Data::where('device_token', $this->selectDevice)-
>get()->last();
    }

    public function render()
    {
        return view('livewire.overview',[
            'devices' => Device::where('user_id', auth()->user()->id)->get()
        ]);
```

```
    }

    public function device()
    {
        $data = Device::where('user_id', auth()->user()->id)->get();
        $i = 0;
        foreach ($data as $datum) {
            $token[$i] = $datum->device_token;
            $i++;
        }
        if ($data->count() == 0) {
            return null;
        }
        return $token[0];
    }

    public function change()
    {
        $this->mount();
        $this->emit('chartDataUpdated', [
            'data' => $this->chartdata,
            'waktu' => $this->waktu
        ]);
    }
}
```

5. Pengujian komunikasi BMS dengan HCU dan *device lain* di Universitas Brawijaya