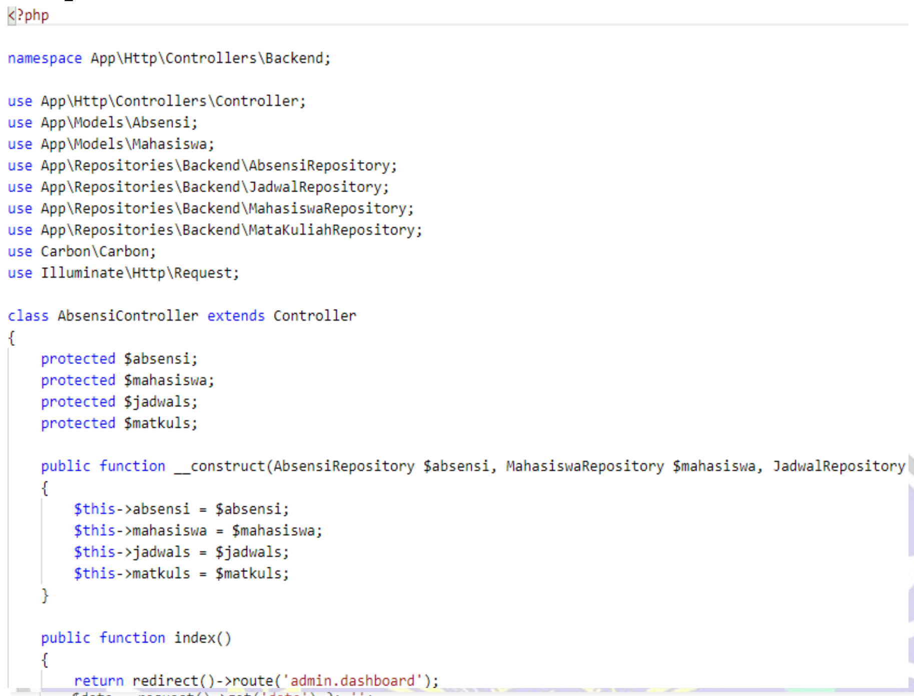
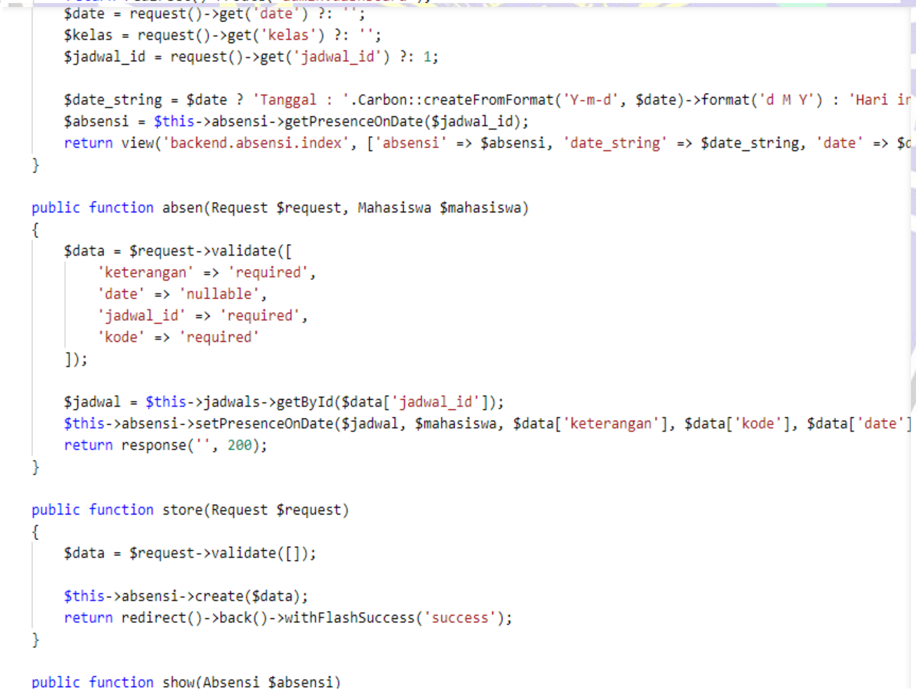
LAMPIRAN

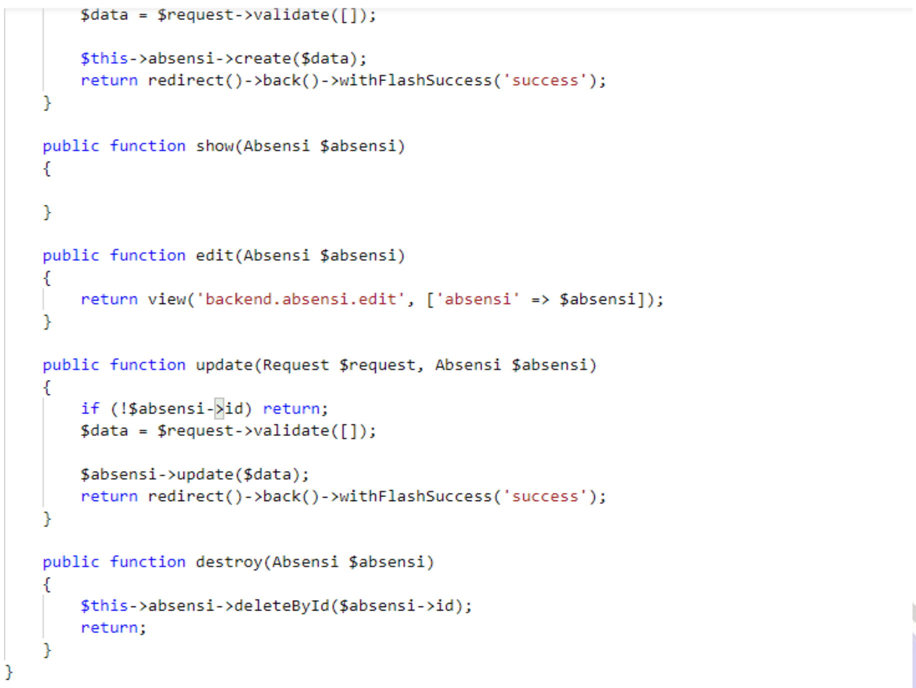
Lampiran 1 : Source Code Absensi Controller



```php
<?php

namespace App\Http\Controllers\Backend;

use App\Http\Controllers\Controller;
use App\Models\Absensi;
use App\Models\Mahasiswa;
use App\Repositories\Backend\AbsensiRepository;
use App\Repositories\Backend\JadwalRepository;
use App\Repositories\Backend\MahasiswaRepository;
use App\Repositories\Backend\MataKuliahRepository;
use Carbon\Carbon;
use Illuminate\Http\Request;

class AbsensiController extends Controller
{
    protected $absensi;
    protected $mahasiswa;
    protected $jadwals;
    protected $matkuls;

    public function __construct(AbsensiRepository $absensi, MahasiswaRepository $mahasiswa, JadwalRepository
    {
        $this->absensi = $absensi;
        $this->mahasiswa = $mahasiswa;
        $this->jadwals = $jadwals;
        $this->matkuls = $matkuls;
    }

    public function index()
    {
        return redirect()->route('admin.dashboard');
```

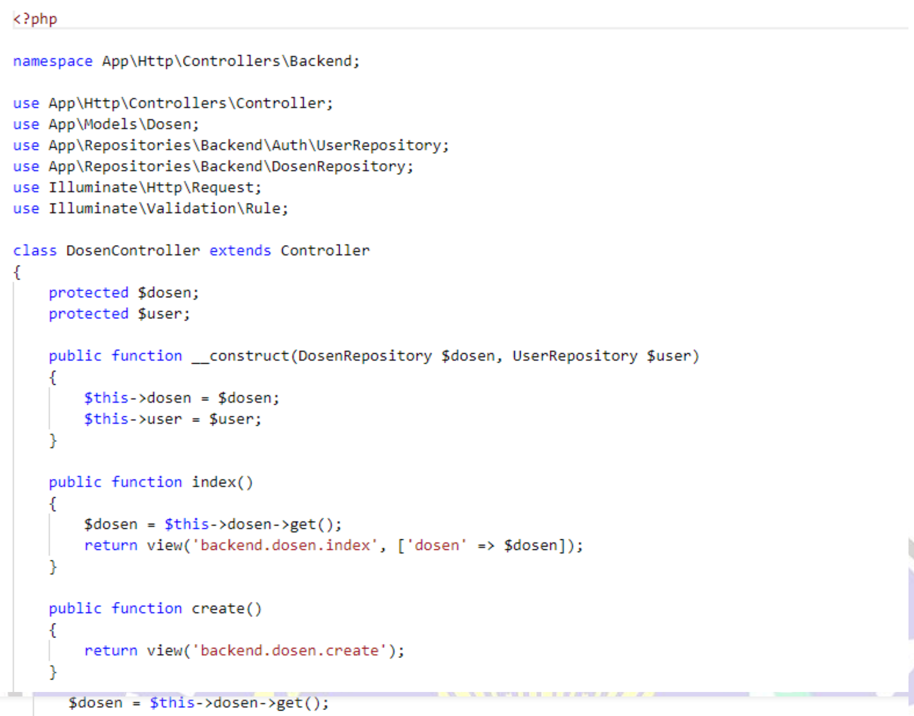


```php
        $date = request()->get('date') ?: '';
        $kelas = request()->get('kelas') ?: '';
        $jadwal_id = request()->get('jadwal_id') ?: 1;

        $date_string = $date ? 'Tanggal : '.Carbon::createFromFormat('Y-m-d', $date)->format('d M Y') : 'Hari ir
        $absensi = $this->absensi->getPresenceOnDate($jadwal_id);
        return view('backend.absensi.index', ['absensi' => $absensi, 'date_string' => $date_string, 'date' => $c
    }

    public function absen(Request $request, Mahasiswa $mahasiswa)
    {
        $data = $request->validate([
            'keterangan' => 'required',
            'date' => 'nullable',
            'jadwal_id' => 'required',
            'kode' => 'required'
        ]);

        $jadwal = $this->jadwals->getById($data['jadwal_id']);
        $this->absensi->setPresenceOnDate($jadwal, $mahasiswa, $data['keterangan'], $data['kode'], $data['date']
        return response('', 200);
    }

    public function store(Request $request)
    {
        $data = $request->validate([]);

        $this->absensi->create($data);
        return redirect()->back()->withFlashSuccess('success');
    }

    public function show(Absensi $absensi)
```

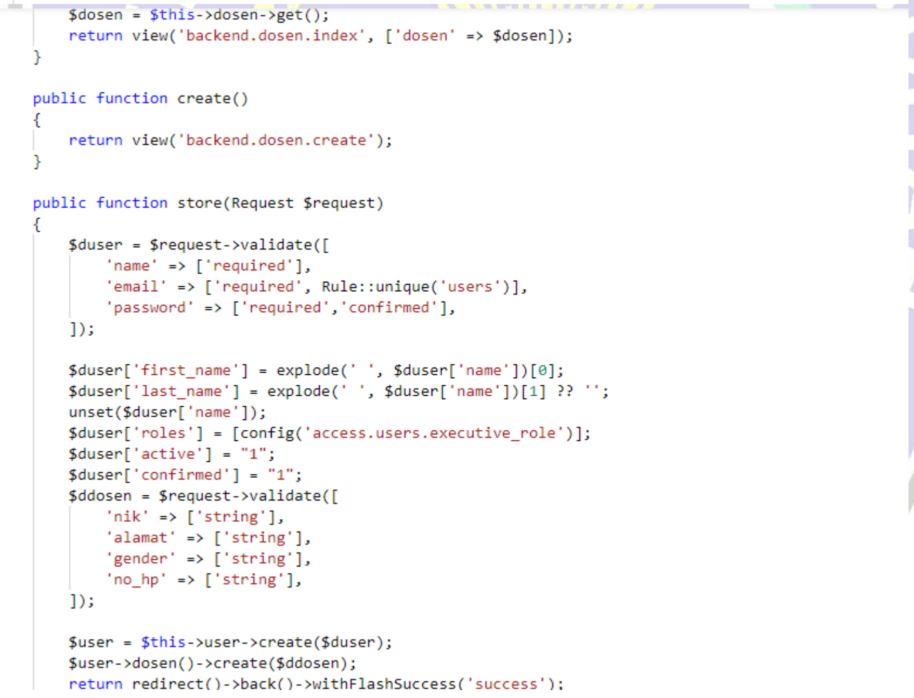


```
        $data = $request->validate([]);

        $this->absensi->create($data);
        return redirect()->back()->withFlashSuccess('success');
    }

    public function show(Absensi $absensi)
    {

    }

    public function edit(Absensi $absensi)
    {
        return view('backend.absensi.edit', ['absensi' => $absensi]);
    }

    public function update(Request $request, Absensi $absensi)
    {
        if (!$absensi->id) return;
        $data = $request->validate([]);

        $absensi->update($data);
        return redirect()->back()->withFlashSuccess('success');
    }

    public function destroy(Absensi $absensi)
    {
        $this->absensi->deleteById($absensi->id);
        return;
    }
}
```

Lampiran 2 : Source Code Dosen Controller

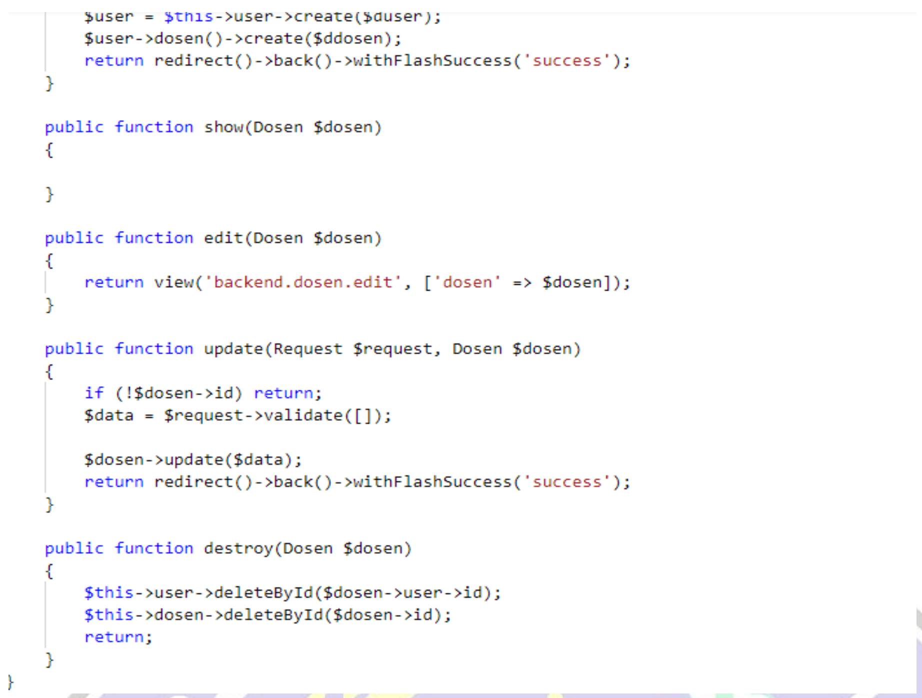


```php
<?php

namespace App\Http\Controllers\Backend;

use App\Http\Controllers\Controller;
use App\Models\Dosen;
use App\Repositories\Backend\Auth\UserRepository;
use App\Repositories\Backend\DosenRepository;
use Illuminate\Http\Request;
use Illuminate\Validation\Rule;

class DosenController extends Controller
{
    protected $dosen;
    protected $user;

    public function __construct(DosenRepository $dosen, UserRepository $user)
    {
        $this->dosen = $dosen;
        $this->user = $user;
    }

    public function index()
    {
        $dosen = $this->dosen->get();
        return view('backend.dosen.index', ['dosen' => $dosen]);
    }

    public function create()
    {
        return view('backend.dosen.create');
    }
```



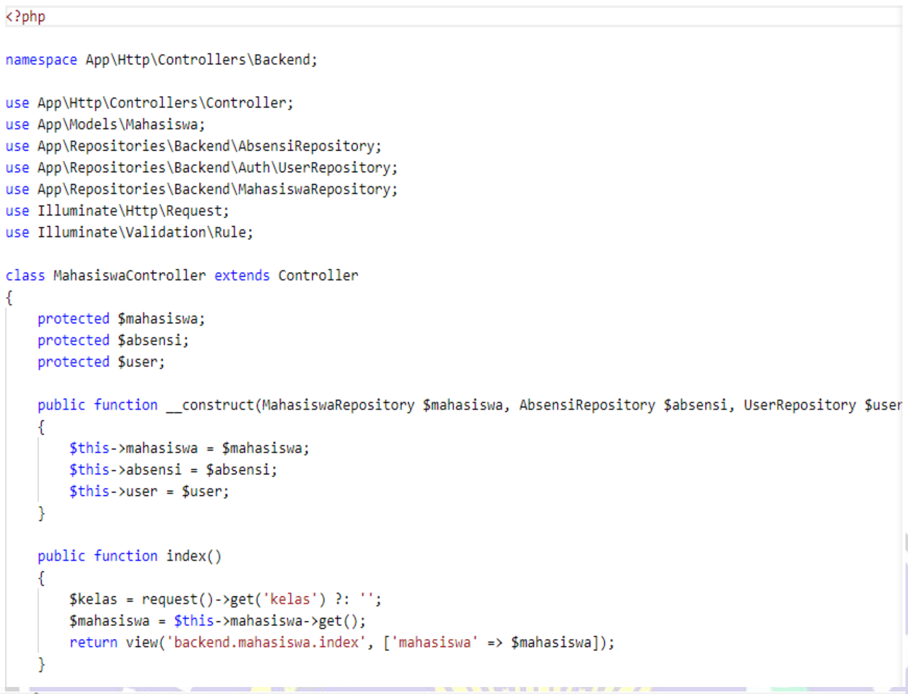
```php
        $dosen = $this->dosen->get();
        return view('backend.dosen.index', ['dosen' => $dosen]);
    }

    public function create()
    {
        return view('backend.dosen.create');
    }
```

```php
    public function store(Request $request)
    {
        $duser = $request->validate([
            'name' => ['required'],
            'email' => ['required', Rule::unique('users')],
            'password' => ['required','confirmed'],
        ]);

        $duser['first_name'] = explode(' ', $duser['name'])[0];
        $duser['last_name'] = explode(' ', $duser['name'])[1] ?? '';
        unset($duser['name']);
        $duser['roles'] = [config('access.users.executive_role')];
        $duser['active'] = "1";
        $duser['confirmed'] = "1";
        $ddosen = $request->validate([
            'nik' => ['string'],
            'alamat' => ['string'],
            'gender' => ['string'],
            'no_hp' => ['string'],
        ]);

        $user = $this->user->create($duser);
        $user->dosen()->create($ddosen);
        return redirect()->back()->withFlashSuccess('success');
```

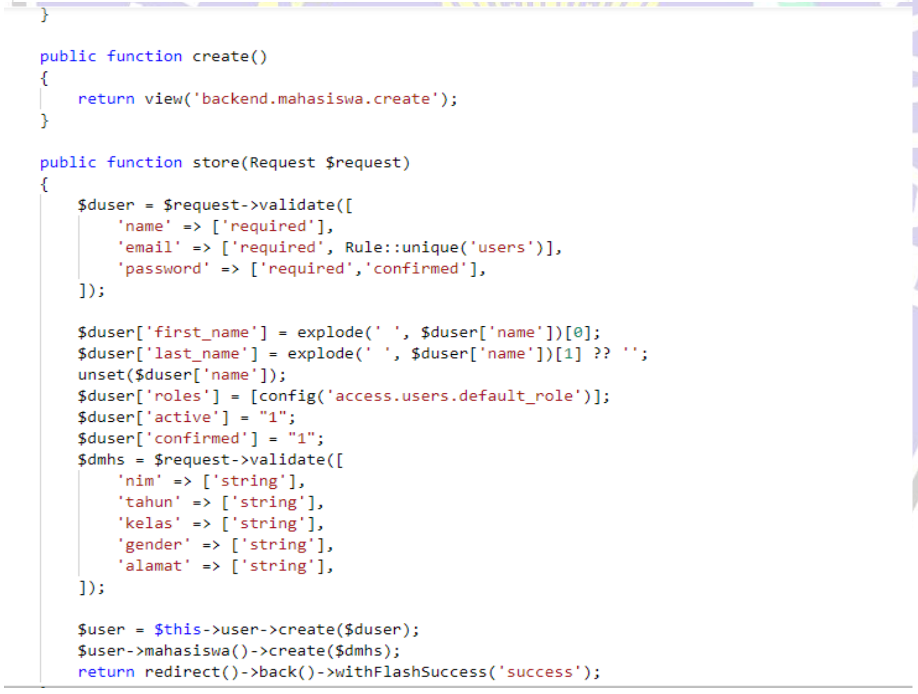


```
        $user = $this->user->create($duser);
        $user->dosen()->create($ddosen);
        return redirect()->back()->withFlashSuccess('success');
    }

    public function show(Dosen $dosen)
    {

    }

    public function edit(Dosen $dosen)
    {
        return view('backend.dosen.edit', ['dosen' => $dosen]);
    }

    public function update(Request $request, Dosen $dosen)
    {
        if (!$dosen->id) return;
        $data = $request->validate([]);

        $dosen->update($data);
        return redirect()->back()->withFlashSuccess('success');
    }

    public function destroy(Dosen $dosen)
    {
        $this->user->deleteById($dosen->user->id);
        $this->dosen->deleteById($dosen->id);
        return;
    }
}
```

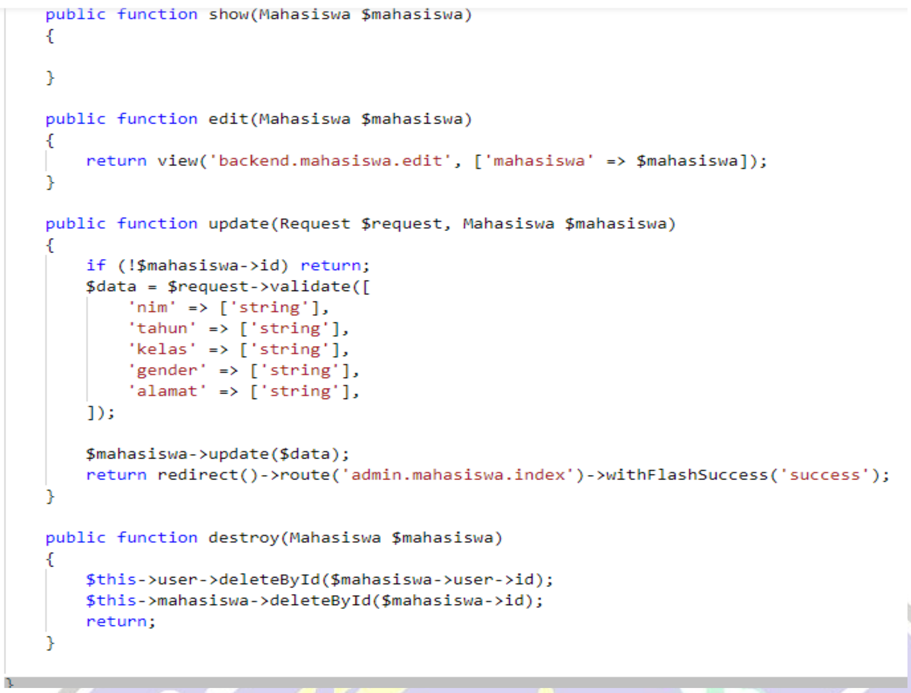
Lampiran 3 : Source Code Mahasiswa Controller



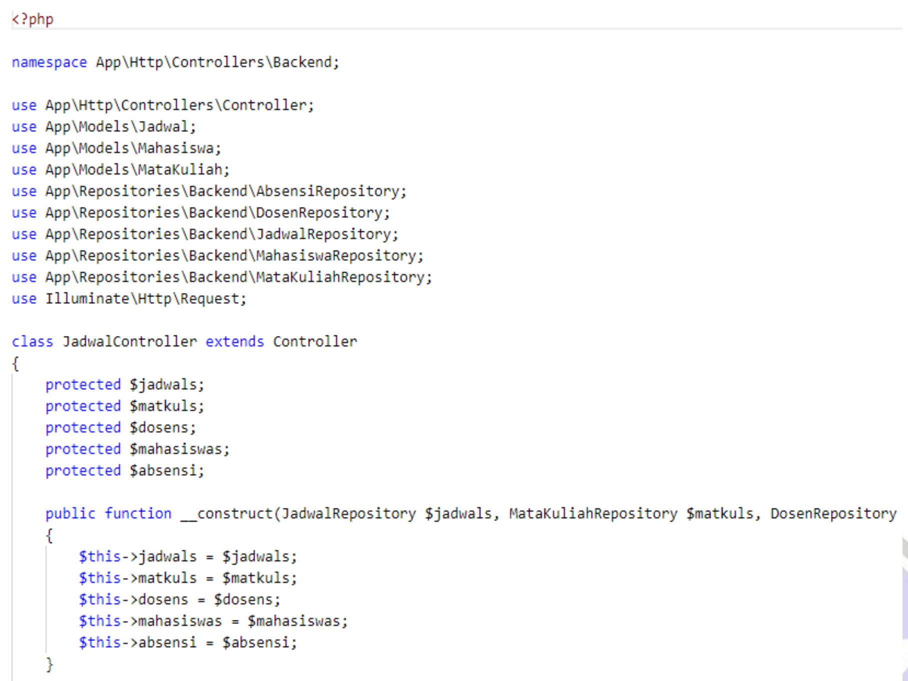
```php
<?php

namespace App\Http\Controllers\Backend;

use App\Http\Controllers\Controller;
use App\Models\Mahasiswa;
use App\Repositories\Backend\AbsensiRepository;
use App\Repositories\Backend\Auth\UserRepository;
use App\Repositories\Backend\MahasiswaRepository;
use Illuminate\Http\Request;
use Illuminate\Validation\Rule;

class MahasiswaController extends Controller
{
    protected $mahasiswa;
    protected $absensi;
    protected $user;

    public function __construct(MahasiswaRepository $mahasiswa, AbsensiRepository $absensi, UserRepository $user
    {
        $this->mahasiswa = $mahasiswa;
        $this->absensi = $absensi;
        $this->user = $user;
    }

    public function index()
    {
        $kelas = request()->get('kelas') ?: '';
        $mahasiswa = $this->mahasiswa->get();
        return view('backend.mahasiswa.index', ['mahasiswa' => $mahasiswa]);
    }
```



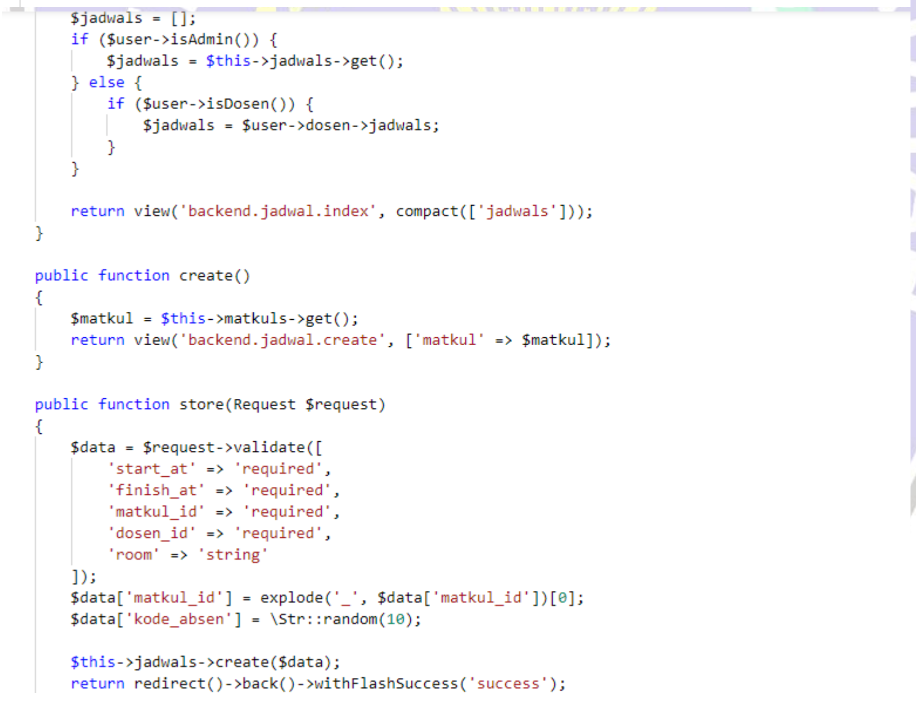
```php
    }

    public function create()
    {
        return view('backend.mahasiswa.create');
    }

    public function store(Request $request)
    {
        $duser = $request->validate([
            'name' => ['required'],
            'email' => ['required', Rule::unique('users')],
            'password' => ['required','confirmed'],
        ]);

        $duser['first_name'] = explode(' ', $duser['name'])[0];
        $duser['last_name'] = explode(' ', $duser['name'])[1] ?? '';
        unset($duser['name']);
        $duser['roles'] = [config('access.users.default_role')];
        $duser['active'] = "1";
        $duser['confirmed'] = "1";
        $dmhs = $request->validate([
            'nim' => ['string'],
            'tahun' => ['string'],
            'kelas' => ['string'],
            'gender' => ['string'],
            'alamat' => ['string'],
        ]);

        $user = $this->user->create($duser);
        $user->mahasiswa()->create($dmhs);
        return redirect()->back()->withFlashSuccess('success');
```


```
    public function show(Mahasiswa $mahasiswa)
    {

    }

    public function edit(Mahasiswa $mahasiswa)
    {
        return view('backend.mahasiswa.edit', ['mahasiswa' => $mahasiswa]);
    }

    public function update(Request $request, Mahasiswa $mahasiswa)
    {
        if (!$mahasiswa->id) return;
        $data = $request->validate([
            'nim' => ['string'],
            'tahun' => ['string'],
            'kelas' => ['string'],
            'gender' => ['string'],
            'alamat' => ['string'],
        ]);

        $mahasiswa->update($data);
        return redirect()->route('admin.mahasiswa.index')->withFlashSuccess('success');
    }

    public function destroy(Mahasiswa $mahasiswa)
    {
        $this->user->deleteById($mahasiswa->user->id);
        $this->mahasiswa->deleteById($mahasiswa->id);
        return;
    }
```

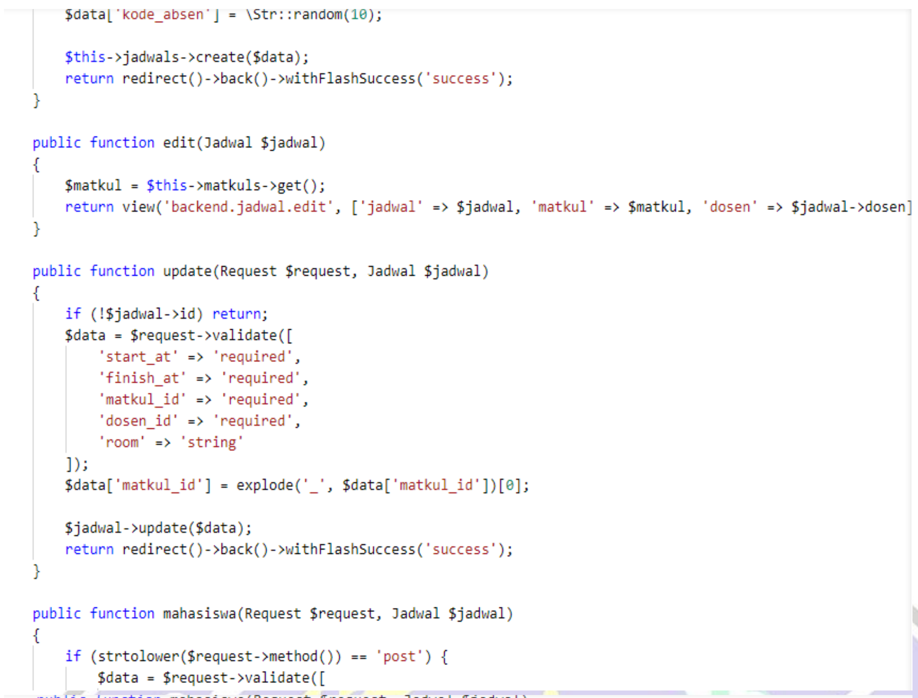
Lampiran 4 : Source Code Jadwal Absensi Controller



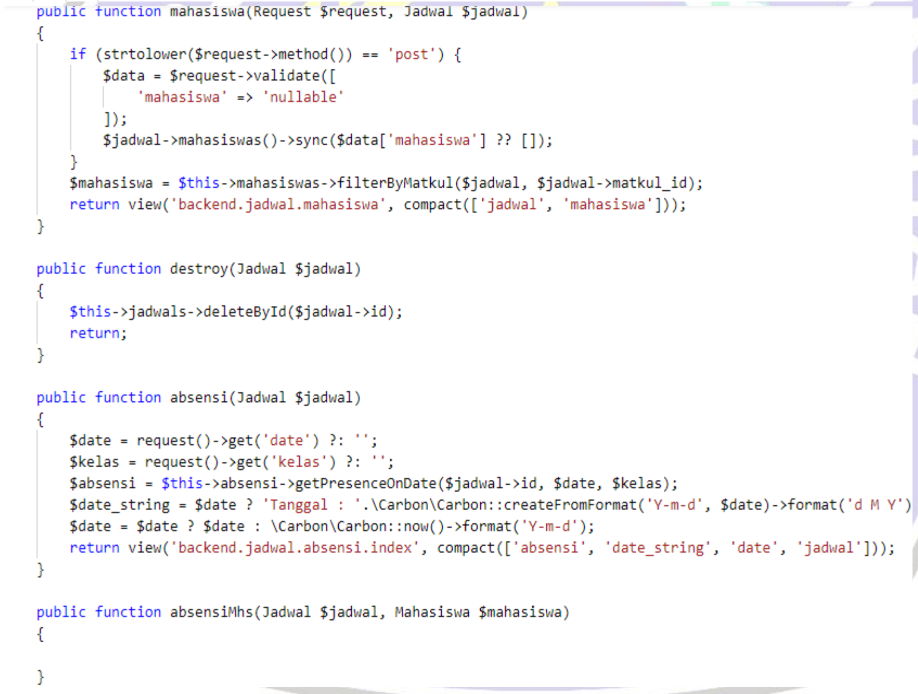
```php
<?php

namespace App\Http\Controllers\Backend;

use App\Http\Controllers\Controller;
use App\Models\Jadwal;
use App\Models\Mahasiswa;
use App\Models\MataKuliah;
use App\Repositories\Backend\AbsensiRepository;
use App\Repositories\Backend\DosenRepository;
use App\Repositories\Backend\JadwalRepository;
use App\Repositories\Backend\MahasiswaRepository;
use App\Repositories\Backend\MataKuliahRepository;
use Illuminate\Http\Request;

class JadwalController extends Controller
{
    protected $jadwals;
    protected $matkuls;
    protected $dosens;
    protected $mahasiswas;
    protected $absensi;

    public function __construct(JadwalRepository $jadwals, MataKuliahRepository $matkuls, DosenRepository
    {
        $this->jadwals = $jadwals;
        $this->matkuls = $matkuls;
        $this->dosens = $dosens;
        $this->mahasiswas = $mahasiswas;
        $this->absensi = $absensi;
    }
```



```php
        $jadwals = [];
        if ($user->isAdmin()) {
            $jadwals = $this->jadwals->get();
        } else {
            if ($user->isDosen()) {
                $jadwals = $user->dosen->jadwals;
            }
        }

        return view('backend.jadwal.index', compact(['jadwals']));
    }

    public function create()
    {
        $matkul = $this->matkuls->get();
        return view('backend.jadwal.create', ['matkul' => $matkul]);
    }

    public function store(Request $request)
    {
        $data = $request->validate([
            'start_at' => 'required',
            'finish_at' => 'required',
            'matkul_id' => 'required',
            'dosen_id' => 'required',
            'room' => 'string'
        ]);
        $data['matkul_id'] = explode('_', $data['matkul_id'])[0];
        $data['kode_absen'] = \Str::random(10);

        $this->jadwals->create($data);
        return redirect()->back()->withFlashSuccess('success');
```

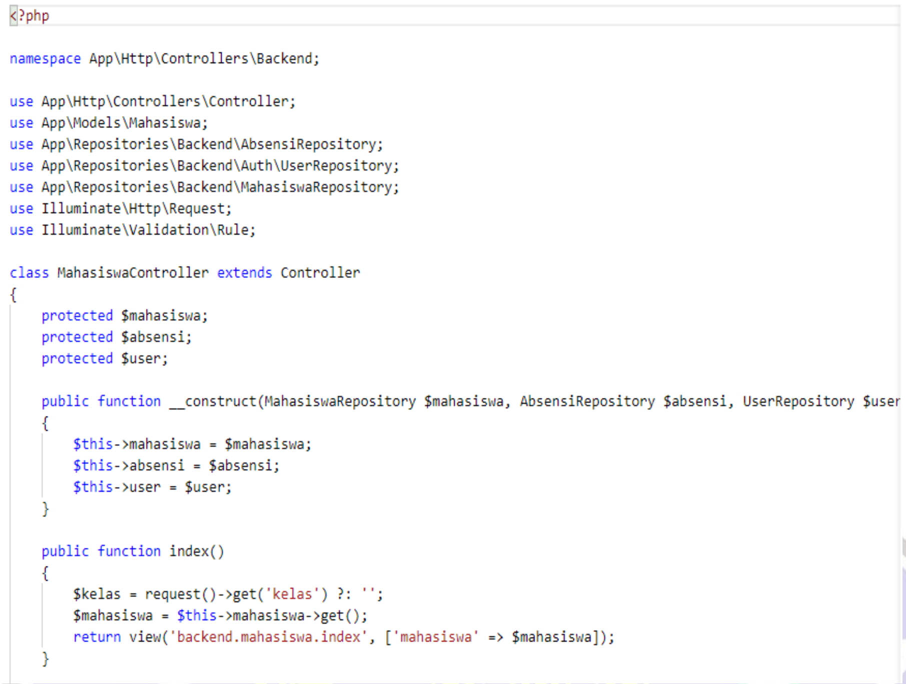


```
    $data['kode_absen'] = \Str::random(10);

    $this->jadwals->create($data);
    return redirect()->back()->withFlashSuccess('success');
}

public function edit(Jadwal $jadwal)
{
    $matkul = $this->matkuls->get();
    return view('backend.jadwal.edit', ['jadwal' => $jadwal, 'matkul' => $matkul, 'dosen' => $jadwal->dosen]
}

public function update(Request $request, Jadwal $jadwal)
{
    if (!$jadwal->id) return;
    $data = $request->validate([
        'start_at' => 'required',
        'finish_at' => 'required',
        'matkul_id' => 'required',
        'dosen_id' => 'required',
        'room' => 'string'
    ]);
    $data['matkul_id'] = explode('_', $data['matkul_id'])[0];

    $jadwal->update($data);
    return redirect()->back()->withFlashSuccess('success');
}

public function mahasiswa(Request $request, Jadwal $jadwal)
{
    if (strtolower($request->method()) == 'post') {
        $data = $request->validate([
```

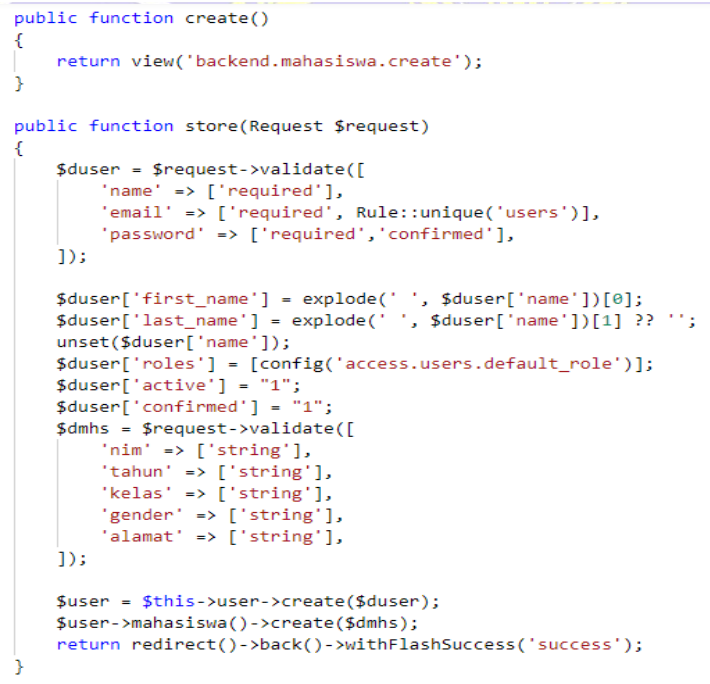


```
public function mahasiswa(Request $request, Jadwal $jadwal)
{
    if (strtolower($request->method()) == 'post') {
        $data = $request->validate([
            'mahasiswa' => 'nullable'
        ]);
        $jadwal->mahasiswas()->sync($data['mahasiswa'] ?? []);
    }
    $mahasiswa = $this->mahasiswas->filterByMatkul($jadwal, $jadwal->matkul_id);
    return view('backend.jadwal.mahasiswa', compact(['jadwal', 'mahasiswa']));
}

public function destroy(Jadwal $jadwal)
{
    $this->jadwals->deleteById($jadwal->id);
    return;
}

public function absensi(Jadwal $jadwal)
{
    $date = request()->get('date') ?: '';
    $kelas = request()->get('kelas') ?: '';
    $absensi = $this->absensi->getPresenceOnDate($jadwal->id, $date, $kelas);
    $date_string = $date ? 'Tanggal : '.\Carbon\Carbon::createFromFormat('Y-m-d', $date)->format('d M Y')
    $date = $date ? $date : \Carbon\Carbon::now()->format('Y-m-d');
    return view('backend.jadwal.absensi.index', compact(['absensi', 'date_string', 'date', 'jadwal']));
}

public function absensiMhs(Jadwal $jadwal, Mahasiswa $mahasiswa)
{

}
```

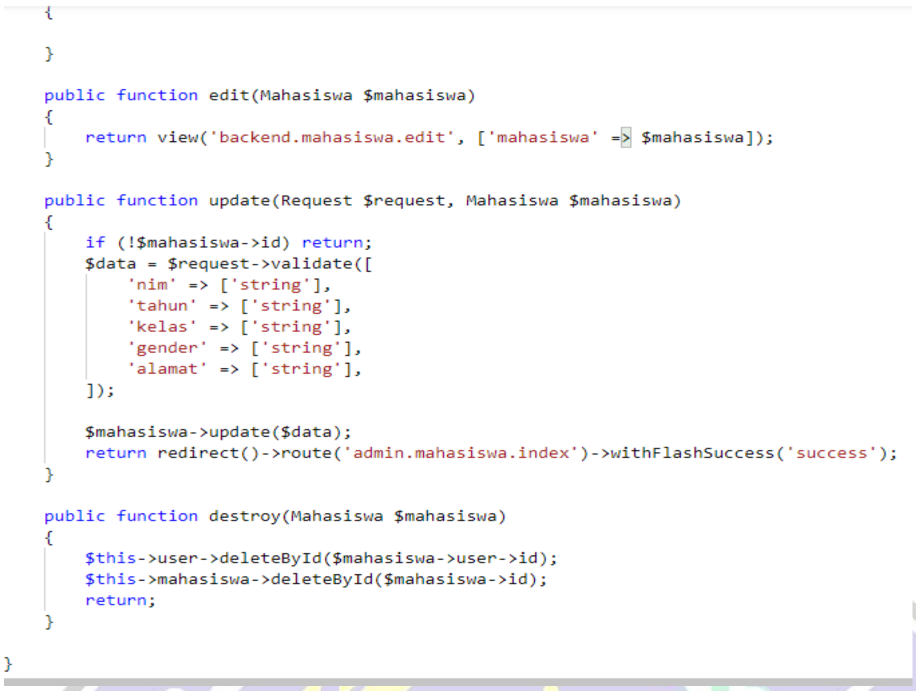
Lampiran 5 : Source Code Mata Kuliah Controller



```php
<?php

namespace App\Http\Controllers\Backend;

use App\Http\Controllers\Controller;
use App\Models\Mahasiswa;
use App\Repositories\Backend\AbsensiRepository;
use App\Repositories\Backend\Auth\UserRepository;
use App\Repositories\Backend\MahasiswaRepository;
use Illuminate\Http\Request;
use Illuminate\Validation\Rule;

class MahasiswaController extends Controller
{
    protected $mahasiswa;
    protected $absensi;
    protected $user;

    public function __construct(MahasiswaRepository $mahasiswa, AbsensiRepository $absensi, UserRepository $user
    {
        $this->mahasiswa = $mahasiswa;
        $this->absensi = $absensi;
        $this->user = $user;
    }

    public function index()
    {
        $kelas = request()->get('kelas') ?: '';
        $mahasiswa = $this->mahasiswa->get();
        return view('backend.mahasiswa.index', ['mahasiswa' => $mahasiswa]);
    }
```

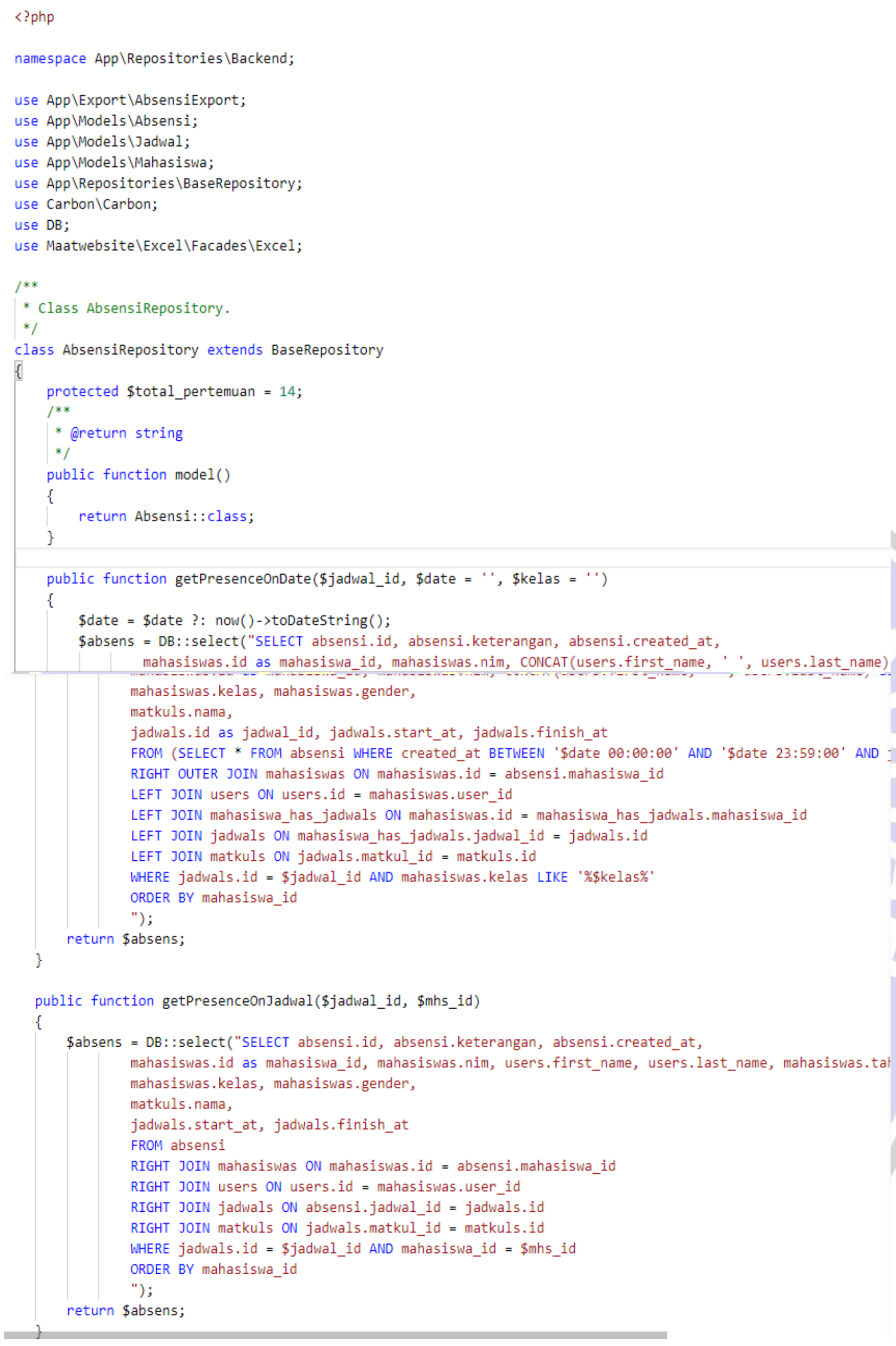


```php
public function create()
{
    return view('backend.mahasiswa.create');
}

public function store(Request $request)
{
    $duser = $request->validate([
        'name' => ['required'],
        'email' => ['required', Rule::unique('users')],
        'password' => ['required','confirmed'],
    ]);

    $duser['first_name'] = explode(' ', $duser['name'])[0];
    $duser['last_name'] = explode(' ', $duser['name'])[1] ?? '';
    unset($duser['name']);
    $duser['roles'] = [config('access.users.default_role')];
    $duser['active'] = "1";
    $duser['confirmed'] = "1";
    $dmhs = $request->validate([
        'nim' => ['string'],
        'tahun' => ['string'],
        'kelas' => ['string'],
        'gender' => ['string'],
        'alamat' => ['string'],
    ]);

    $user = $this->user->create($duser);
    $user->mahasiswa()->create($dmhs);
    return redirect()->back()->withFlashSuccess('success');
}
```

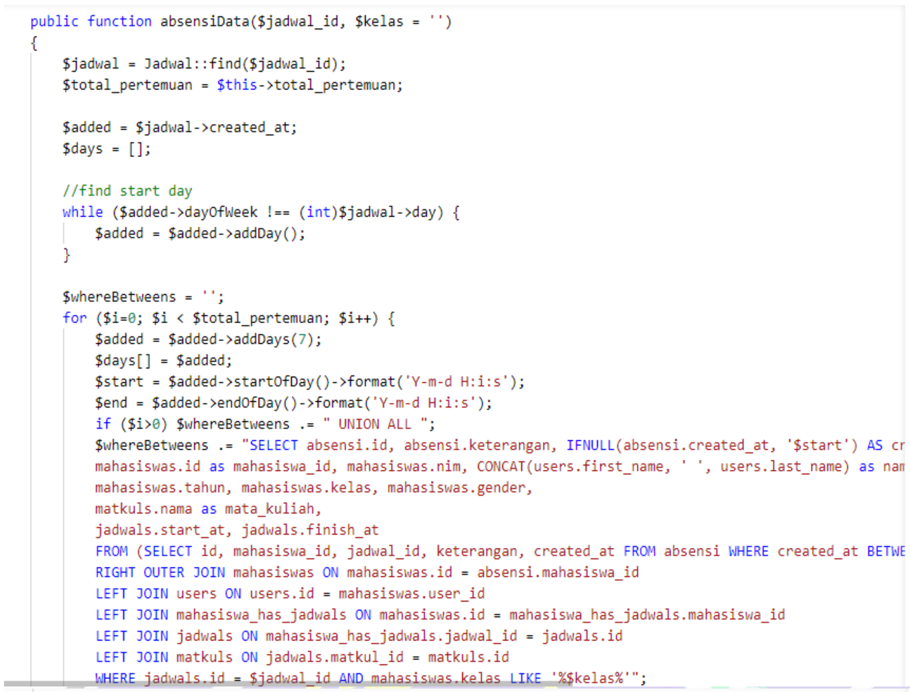


```
    {

    }

    public function edit(Mahasiswa $mahasiswa)
    {
        return view('backend.mahasiswa.edit', ['mahasiswa' => $mahasiswa]);
    }

    public function update(Request $request, Mahasiswa $mahasiswa)
    {
        if (!$mahasiswa->id) return;
        $data = $request->validate([
            'nim' => ['string'],
            'tahun' => ['string'],
            'kelas' => ['string'],
            'gender' => ['string'],
            'alamat' => ['string'],
        ]);

        $mahasiswa->update($data);
        return redirect()->route('admin.mahasiswa.index')->withFlashSuccess('success');
    }

    public function destroy(Mahasiswa $mahasiswa)
    {
        $this->user->deleteById($mahasiswa->user->id);
        $this->mahasiswa->deleteById($mahasiswa->id);
        return;
    }

}
```

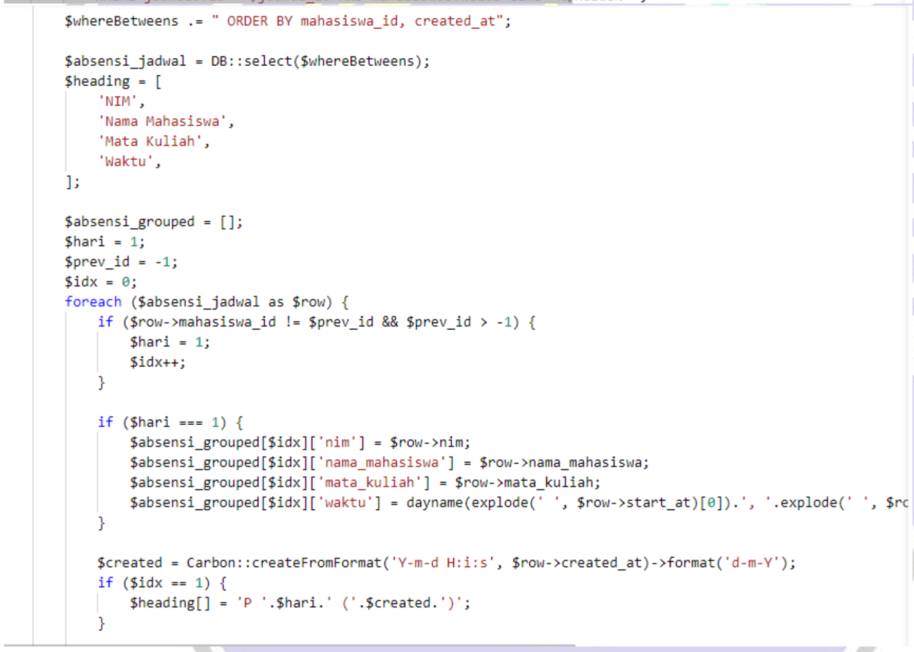
Lampiran 6 :Source Code Absensi Repository



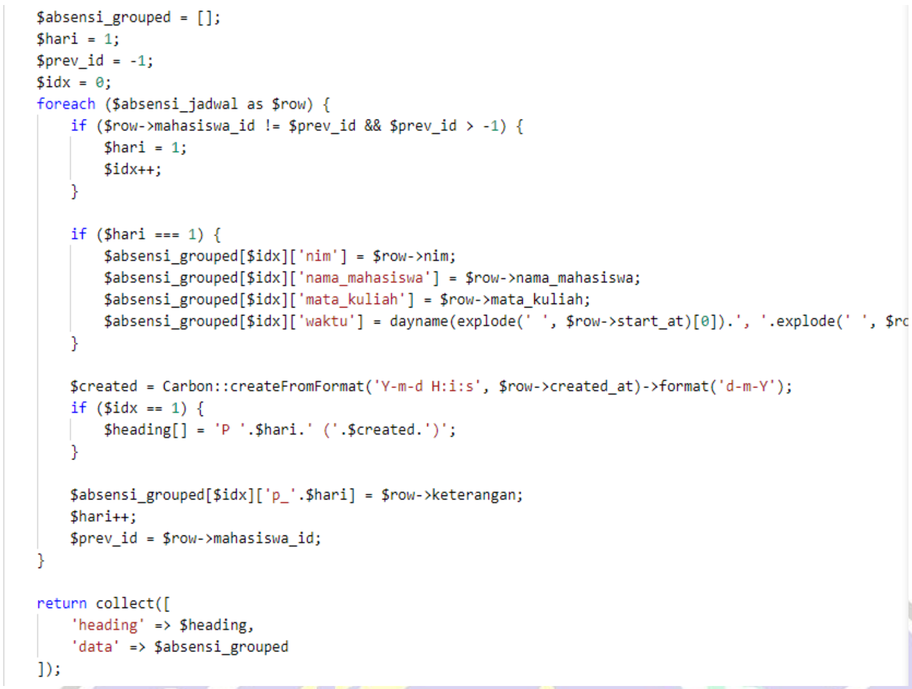
```php
<?php

namespace App\Repositories\Backend;

use App\Export\AbsensiExport;
use App\Models\Absensi;
use App\Models\Jadwal;
use App\Models\Mahasiswa;
use App\Repositories\BaseRepository;
use Carbon\Carbon;
use DB;
use Maatwebsite\Excel\Facades\Excel;

/**
 * Class AbsensiRepository.
 */
class AbsensiRepository extends BaseRepository
{
    protected $total_pertemuan = 14;
    /**
     * @return string
     */
    public function model()
    {
        return Absensi::class;
    }

    public function getPresenceOnDate($jadwal_id, $date = '', $kelas = '')
    {
        $date = $date ?: now()->toDateString();
        $absens = DB::select("SELECT absensi.id, absensi.keterangan, absensi.created_at,
                 mahasiswas.id as mahasiswa_id, mahasiswas.nim, CONCAT(users.first_name, ' ', users.last_name)
                mahasiswas.kelas, mahasiswas.gender,
                matkuls.nama,
                jadwals.id as jadwal_id, jadwals.start_at, jadwals.finish_at
                FROM (SELECT * FROM absensi WHERE created_at BETWEEN '$date 00:00:00' AND '$date 23:59:00' AND
                RIGHT OUTER JOIN mahasiswas ON mahasiswas.id = absensi.mahasiswa_id
                LEFT JOIN users ON users.id = mahasiswas.user_id
                LEFT JOIN mahasiswa_has_jadwals ON mahasiswas.id = mahasiswa_has_jadwals.mahasiswa_id
                LEFT JOIN jadwals ON mahasiswa_has_jadwals.jadwal_id = jadwals.id
                LEFT JOIN matkuls ON jadwals.matkul_id = matkuls.id
                WHERE jadwals.id = $jadwal_id AND mahasiswas.kelas LIKE '%$kelas%'
                ORDER BY mahasiswa_id
                ");
        return $absens;
    }

    public function getPresenceOnJadwal($jadwal_id, $mhs_id)
    {
        $absens = DB::select("SELECT absensi.id, absensi.keterangan, absensi.created_at,
                mahasiswas.id as mahasiswa_id, mahasiswas.nim, users.first_name, users.last_name, mahasiswas.ta
                mahasiswas.kelas, mahasiswas.gender,
                matkuls.nama,
                jadwals.start_at, jadwals.finish_at
                FROM absensi
                RIGHT JOIN mahasiswas ON mahasiswas.id = absensi.mahasiswa_id
                RIGHT JOIN users ON users.id = mahasiswas.user_id
                RIGHT JOIN jadwals ON absensi.jadwal_id = jadwals.id
                RIGHT JOIN matkuls ON jadwals.matkul_id = matkuls.id
                WHERE jadwals.id = $jadwal_id AND mahasiswa_id = $mhs_id
                ORDER BY mahasiswa_id
                ");
        return $absens;
    }
```



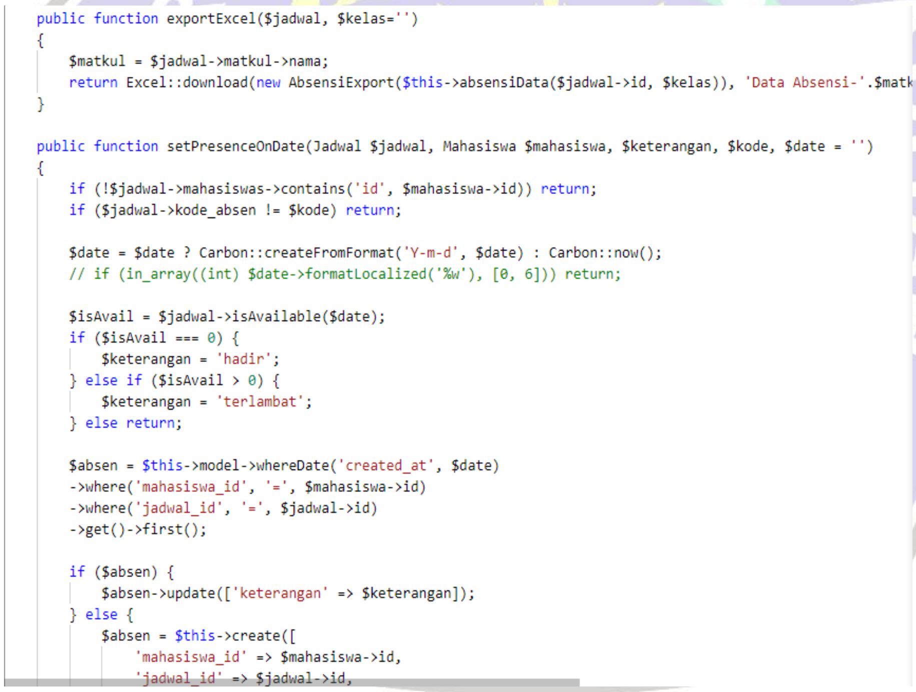
```php
public function absensiData($jadwal_id, $kelas = '')
{
    $jadwal = Jadwal::find($jadwal_id);
    $total_pertemuan = $this->total_pertemuan;

    $added = $jadwal->created_at;
    $days = [];

    //find start day
    while ($added->dayOfWeek !== (int)$jadwal->day) {
        $added = $added->addDay();
    }

    $whereBetweens = '';
    for ($i=0; $i < $total_pertemuan; $i++) {
        $added = $added->addDays(7);
        $days[] = $added;
        $start = $added->startOfDay()->format('Y-m-d H:i:s');
        $end = $added->endOfDay()->format('Y-m-d H:i:s');
        if ($i>0) $whereBetweens .= " UNION ALL ";
        $whereBetweens .= "SELECT absensi.id, absensi.keterangan, IFNULL(absensi.created_at, '$start') AS cr
        mahasiswas.id as mahasiswa_id, mahasiswas.nim, CONCAT(users.first_name, ' ', users.last_name) as nam
        mahasiswas.tahun, mahasiswas.kelas, mahasiswas.gender,
        matkuls.nama as mata_kuliah,
        jadwals.start_at, jadwals.finish_at
        FROM (SELECT id, mahasiswa_id, jadwal_id, keterangan, created_at FROM absensi WHERE created_at BETWE
        RIGHT OUTER JOIN mahasiswas ON mahasiswas.id = absensi.mahasiswa_id
        LEFT JOIN users ON users.id = mahasiswas.user_id
        LEFT JOIN mahasiswa_has_jadwals ON mahasiswas.id = mahasiswa_has_jadwals.mahasiswa_id
        LEFT JOIN jadwals ON mahasiswa_has_jadwals.jadwal_id = jadwals.id
        LEFT JOIN matkuls ON jadwals.matkul_id = matkuls.id
        WHERE jadwals.id = $jadwal_id AND mahasiswas.kelas LIKE '%$kelas%'";
```



```php
    $whereBetweens .= " ORDER BY mahasiswa_id, created_at";

    $absensi_jadwal = DB::select($whereBetweens);
    $heading = [
        'NIM',
        'Nama Mahasiswa',
        'Mata Kuliah',
        'Waktu',
    ];

    $absensi_grouped = [];
    $hari = 1;
    $prev_id = -1;
    $idx = 0;
    foreach ($absensi_jadwal as $row) {
        if ($row->mahasiswa_id != $prev_id && $prev_id > -1) {
            $hari = 1;
            $idx++;
        }

        if ($hari === 1) {
            $absensi_grouped[$idx]['nim'] = $row->nim;
            $absensi_grouped[$idx]['nama_mahasiswa'] = $row->nama_mahasiswa;
            $absensi_grouped[$idx]['mata_kuliah'] = $row->mata_kuliah;
            $absensi_grouped[$idx]['waktu'] = dayname(explode(' ', $row->start_at)[0]).', '.explode(' ', $ro
        }

        $created = Carbon::createFromFormat('Y-m-d H:i:s', $row->created_at)->format('d-m-Y');
        if ($idx == 1) {
            $heading[] = 'P '.$hari.' ('.$created.')';
        }
```

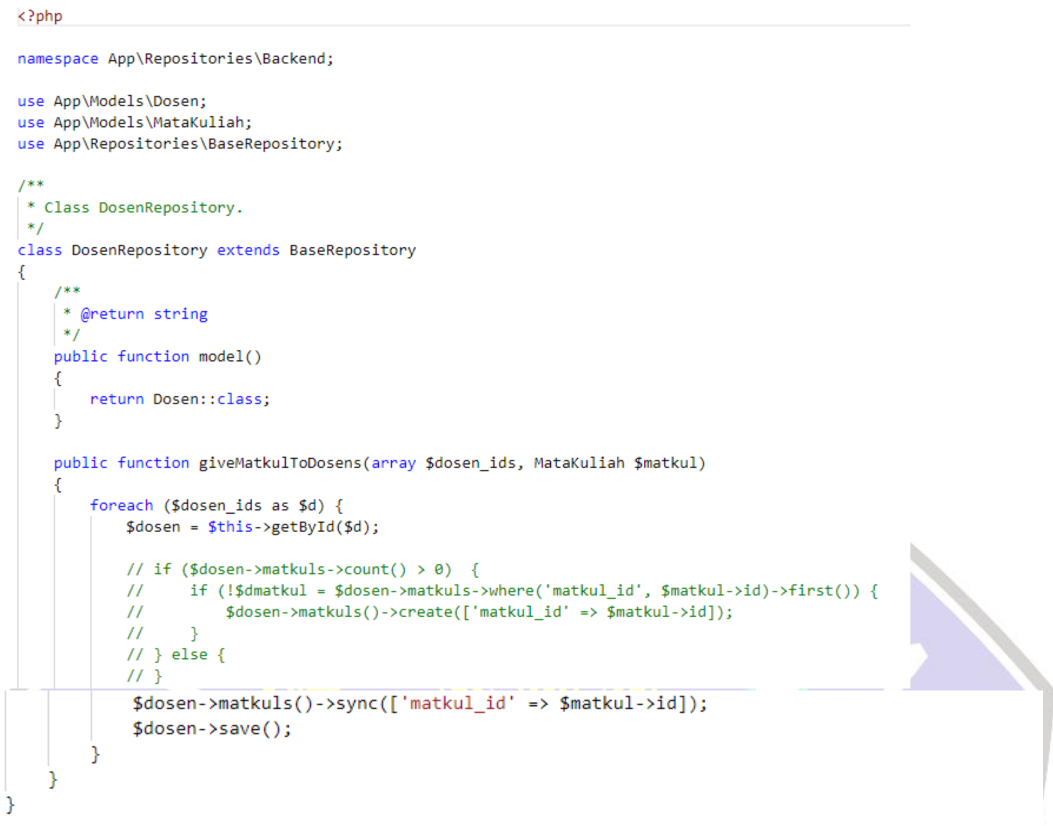


```
$absensi_grouped = [];
$hari = 1;
$prev_id = -1;
$idx = 0;
foreach ($absensi_jadwal as $row) {
    if ($row->mahasiswa_id != $prev_id && $prev_id > -1) {
        $hari = 1;
        $idx++;
    }

    if ($hari === 1) {
        $absensi_grouped[$idx]['nim'] = $row->nim;
        $absensi_grouped[$idx]['nama_mahasiswa'] = $row->nama_mahasiswa;
        $absensi_grouped[$idx]['mata_kuliah'] = $row->mata_kuliah;
        $absensi_grouped[$idx]['waktu'] = dayname(explode(' ', $row->start_at)[0]).', '.explode(' ', $rc
    }

    $created = Carbon::createFromFormat('Y-m-d H:i:s', $row->created_at)->format('d-m-Y');
    if ($idx == 1) {
        $heading[] = 'P '.$hari.' ('.$created.')';
    }

    $absensi_grouped[$idx]['p_'.$hari] = $row->keterangan;
    $hari++;
    $prev_id = $row->mahasiswa_id;
}

return collect([
    'heading' => $heading,
    'data' => $absensi_grouped
]);
```



```
public function exportExcel($jadwal, $kelas='')
{
    $matkul = $jadwal->matkul->nama;
    return Excel::download(new AbsensiExport($this->absensiData($jadwal->id, $kelas)), 'Data Absensi-'.$matk
}

public function setPresenceOnDate(Jadwal $jadwal, Mahasiswa $mahasiswa, $keterangan, $kode, $date = '')
{
    if (!$jadwal->mahasiswas->contains('id', $mahasiswa->id)) return;
    if ($jadwal->kode_absen != $kode) return;

    $date = $date ? Carbon::createFromFormat('Y-m-d', $date) : Carbon::now();
    // if (in_array((int) $date->formatLocalized('%w'), [0, 6])) return;

    $isAvail = $jadwal->isAvailable($date);
    if ($isAvail === 0) {
        $keterangan = 'hadir';
    } else if ($isAvail > 0) {
        $keterangan = 'terlambat';
    } else return;

    $absen = $this->model->whereDate('created_at', $date)
    ->where('mahasiswa_id', '=', $mahasiswa->id)
    ->where('jadwal_id', '=', $jadwal->id)
    ->get()->first();

    if ($absen) {
        $absen->update(['keterangan' => $keterangan]);
    } else {
        $absen = $this->create([
            'mahasiswa_id' => $mahasiswa->id,
            'jadwal_id' => $jadwal->id,
```

Lampiran 7 : Source Code Dosen Repository



```php
<?php

namespace App\Repositories\Backend;

use App\Models\Dosen;
use App\Models\MataKuliah;
use App\Repositories\BaseRepository;

/**
 * Class DosenRepository.
 */
class DosenRepository extends BaseRepository
{
    /**
     * @return string
     */
    public function model()
    {
        return Dosen::class;
    }

    public function giveMatkulToDosens(array $dosen_ids, MataKuliah $matkul)
    {
        foreach ($dosen_ids as $d) {
            $dosen = $this->getById($d);

            // if ($dosen->matkuls->count() > 0)  {
            //     if (!$dmatkul = $dosen->matkuls->where('matkul_id', $matkul->id)->first()) {
            //         $dosen->matkuls()->create(['matkul_id' => $matkul->id]);
            //     }
            // } else {
            // }
            $dosen->matkuls()->sync(['matkul_id' => $matkul->id]);
            $dosen->save();
        }
    }
}
```

Lampiran 8 : Source Code Jadwal Repository

```php
<?php

namespace App\Repositories\Backend;

use App\Models\Jadwal;
use App\Repositories\BaseRepository;

/**
 * Class JadwalRepository.
 */
class JadwalRepository extends BaseRepository
{
    /**
     * @return string
     */
    public function model()
    {
        return Jadwal::class;
    }

    public function giveMatkulToDosens(array $dosen_ids, Jadwal $matkul)
    {
        // $matkul->dosens()->sync($dosen_ids);
    }
}
```

Lampiran 9 : Source Code Mahasiswa Repository

```php
<?php

namespace App\Repositories\Backend;

use App\Models\Jadwal;
use App\Models\Mahasiswa;
use App\Repositories\BaseRepository;

/**
 * Class MahasiswaRepository.
 */
class MahasiswaRepository extends BaseRepository
{
    /**
     * @return string
     */
    public function model()
    {
        return Mahasiswa::class;
    }

    public function filterByMatkul(Jadwal $jadwal, $matkul_id)
    {
        return $this->model->whereHas('jadwals', function($q) use ($jadwal){
            return $q->where('id', $jadwal->id);
        })
        ->orWhereDoesntHave('jadwals', function($q) use($matkul_id) {
            return $q->where('matkul_id', '=', $matkul_id);
        })->orderBy('kelas')->get();
    }
}
```

Lampiran 10 :Source Code Matakuliah Repository

```php
<?php

namespace App\Repositories\Backend;

use App\Models\MataKuliah;
use App\Repositories\BaseRepository;

/**
 * Class MataKuliahRepository.
 */
class MataKuliahRepository extends BaseRepository
{
    /**
     * @return string
     */
    public function model()
    {
        return MataKuliah::class;
    }

    public function giveMatkulToDosens(array $dosen_ids, MataKuliah $matkul)
    {
        $matkul->dosens()->sync($dosen_ids);
    }
}
```