CRUD Laundry

- Pelanggan
  - Create

```
public function register(Request $req){
    $val = Validator::make($req->all(),[
        'nama_pelanggan' => 'required',
        'alamat' => 'required',
        'telp' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $buat = ModelPelanggan::create([
        'nama_pelanggan'=>$req->nama_pelanggan,
        'alamat'=>$req->alamat,
        'telp'=>$req->telp,
    ]);

    if($buat){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

  - Update

```
public function edit($id, Request $req){
    $val = Validator::make($req->all(),[
        'nama_pelanggan' => 'required',
        'alamat' => 'required',
        'telp' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $edit = ModelPelanggan::where('id_jeniscuci', $id)->update([
        'nama_pelanggan'=>$req->nama_pelanggan,
        'alamat'=>$req->alamat,
        'telp'=>$req->telp,
    ]);
}
```

- Delete

```php
public function hapus($id){
    $hapus = ModelPelanggan::where('id_pelanggan', $id)->delete();

    if($hapus){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

- Model

```php
class ModelPelanggan extends Model
{
    protected $table = "pelanggan";
    public $timestamps = false;
    protected $fillable = [
        'nama_pelanggan', 'alamat', 'telp'
    ];
}
```

- Petugas
  - Create

```php
public function register(Request $request)
{
    $validator = Validator::make($request->all(), [
        'nama_petugas' => 'required|string|max:255',
        'telp' => 'required|string|max:255',
        'username' => 'required|string|email|max:255',
        'password' => 'required|string|min:6|confirmed',
    ]);

    if($validator->fails()){
        return response()->json($validator->errors()->toJson(), 400);
    }

    $user = User::create([
        'nama_petugas' => $request->get('nama_petugas'),
        'telp' => $request->get('telp'),
        'username' => $request->get('username'),
        'password' => Hash::make($request->get('password')),
    ]);

    $token = JWTAuth::fromUser($user);

    return response()->json(compact('user','token'),201);
}
```

- Login

```php
public function login(Request $request)
{
    $credentials = $request->only('username', 'password');

    try {
        if (! $token = JWTAuth::attempt($credentials)) {
            return response()->json(['error' => 'invalid_credentials'], 400);
        }
    } catch (JWTException $e) {
        return response()->json(['error' => 'could_not_create_token'], 500);
    }

    return response()->json(compact('token'));
}
```

- Update

```php
public function edit($id, Request $req){
    $val = Validator::make($req->all(),[
        'nama_petugas' => 'required',
        'telp' => 'required',
        'username' => 'required',
        'password' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $edit = User::where('id', $id)->update([
        'nama_petugas'=>$req->nama_petugas,
        'telp'=>$req->telp,
        'username'=>$req->username,
        'password'=>$req->password,
    ]);

    if($edit){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

- Delete

```php
public function delete($id){
    $del = User::where('id', $id)->delete();

    if($del){
        return Response()->json(['Berhhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

- Model

```php
class User extends Authenticatable implements JWTSubject
{
    use Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $table = "petugas";
    public $timestamps = false;
    protected $fillable = [
        'nama_petugas', 'telp', 'username', 'password',
    ];

    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];

    public function getJWTIdentifier()
    {
        return $this->getKey();
    }
    public function getJWTCustomClaims()
    {
        return [];
    }
}
```

- Jenis Cuci
  - Create

```
public function register(Request $req){
    $val = Validator::make($req->all(),[
        'nama_jeniscuci' => 'required',
        'harga' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $buat = ModelJenisCuci::create([
        'nama_jeniscuci'=>$req->nama_jeniscuci,
        'harga'=>$req->harga,
    ]);

    if($buat){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

  - Update

```
public function edit($id, Request $req){
    $val = Validator::make($req->all(),[
        'nama_jenis' => 'required',
        'harga' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $edit = ModelJenisCuci::where('id_jeniscuci', $id)->update([
        'nama_jenis'=>$req->nama_jenis,
        'harga'=>$req->harga,
    ]);

    if($edit){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

- Model

```
class ModelJenisCuci extends Model
{
    protected $table = 'jenis_cuci';
    public $timestamps = false;
    protected $fillable = [
        'nama_jeniscuci', 'harga'
    ];
}
```

- Detail Transaksi
  - Create

```
public function register(Request $req){
    $val = Validator::make($req->all(),[
        'id_transaksi' => 'required',
        'id_jeniscuci' => 'required',
        'qty' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $sub = ModelJenisCuci::where('id_jeniscuci', $req->id_jeniscuci)->first();
    $subtotal = $req->qty * $sub->harga;

    $buat = ModelDetailTransaksi::create([
        'id_transaksi'=>$req->id_transaksi,
        'qty'=>$req->qty,
        'id_jeniscuci'=>$req->id_jeniscuci,
        'subtotal'=>$subtotal,
    ]);

    if($buat){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

  - Update

```
public function edit($id, Request $req){
    $val = Validator::make($req->all(),[
        'id_transaksi' => 'required',
        'qty' => 'required',
        'id_jeniscuci' => 'required',
        'subtotal' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $edit = ModelDetailTransaksi::where('id_detailtransaksi', $id)->update([
        'id_transaksi'=>$req->id_transaksi,
        'qty'=>$req->qty,
        'id_jeniscuci'=>$req->id_jeniscuci,
        'subtotal'=>$req->subtotal,
    ]);
}
```

- Delete

```php
public function hapus($id){
    $hapus = ModelDetailTransaksi::where('id_detailtransaksi', $id)->delete();

    if($hapus){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

- Model

```php
class ModelDetailTransaksi extends Model
{
    protected $table = "detail_transaksi";
    public $timestamps = false;
    protected $fillable = [
        'id_transaksi', 'qty', 'subtotal', 'id_jeniscuci'
    ];
}
```

- Transaksi
  - Create

```php
public function register(Request $req){
    $val = Validator::make($req->all(),[
        'id_pelanggan' => 'required',
        'id' => 'required',
        'tanggal_transaksi' => 'required',
        'tanggal_selesai' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $buat = ModelTransaksi::create([
        'id_pelanggan'=>$req->id_pelanggan,
        'id'=>$req->id,
        'tanggal_transaksi'=>$req->tanggal_transaksi,
        'tanggal_selesai'=>$req->tanggal_selesai,
    ]);

    if($buat){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

- Read

```php
public function tampil(Request $req){
    $transaksi = DB::table('transaksi')->join('pelanggan', 'pelanggan.id_pelanggan','=', 'transaksi.id_pelanggan')
    ->where('transaksi.tanggal_transaksi','>=',$req->tanggal_transaksi)
    ->where('transaksi.tanggal_transaksi','<=',$req->tanggal_selesai)
    ->select('nama_pelanggan','telp','alamat','transaksi.id_transaksi','tanggal_transaksi','tanggal_selesai')
    ->get();

    if($transaksi->count() > 0){
        $data_transaksi = array();
        foreach ($transaksi as $t){
            $grand = DB::table('detail_transaksi')->where('id_transaksi','=',$t->id_transaksi)
            ->groupBy('id_transaksi')
            ->select(DB::raw('sum(subtotal) as grandtotal'))
            ->first();

            $detail = DB::table('detail_transaksi')->join('jenis_cuci','detail_transaksi.id_jeniscuci','=','jenis_cuci.id_jeniscuci')
                ->where('id_transaksi','=',$t->id_transaksi)
                ->get();

                $data_transaksi[] = array(
                    'Tanggal' => $t->tanggal_transaksi,
                    'Nama Pelanggan' => $t->nama_pelanggan,
                    'Alamat' => $t->alamat,
                    'No Telp' => $t->telp,
                    'Deadline' => $t->tanggal_selesai,
                    'Grand Total' => $grand,
                    'Detail' => $detail,
                );
            }

            return response()->json(compact('data_transaksi'));
            } else{
            $status = 'tidak ada transaksi antara tanggal '.$req->tanggal_transaksi.' sampai dengan tanggal '.$req->tanggal_selesai;
            return response()->json(compact('status'));
```

- Update

```php
public function edit($id, Request $req){
    $val = Validator::make($req->all(),[
        'id_pelanggan' => 'required',
        'id' => 'required',
        'tanggal_transaksi' => 'required',
        'tanggal_selesai' => 'required',
    ]);

    if($val->fails()){
        return Response()->json($val->errors());
    }

    $edit = ModelTransaksi::where('id_transaksi', $id)->update([
        'id_pelanggan'=>$req->id_pelanggan,
        'id'=>$req->id,
        'tanggal_transaksi'=>$req->tanggal_transaksi,
        'tanggal_selesai'=>$req->tanggal_selesai,
    ]);
}
```

- Delete

```php
public function hapus($id){
    $hapus = ModelTransaksi::where('id_transaksi', $id)->delete();

    if($hapus){
        return Response()->json(['Berhasil']);
    } else {
        return Response()->json(['Gagal']);
    }
}
```

- Model

```php
class ModelTransaksi extends Model
{
    protected $table = "transaksi";
    public $timestamps = false;
    protected $fillable = [
        'id_pelanggan', 'id', 'tanggal_transaksi', 'tanggal_selesai',
    ];
}
```