

Nama	:	Candra Ahmad Dani
Nim	:	2341720187
Kelas	:	TI-2A

Praktikum 1 - Modal Ajax Tambah Data (Data User)

- 1. Kita buat form tambah data baru dengan menerapkan modal dan proses ajax.
- 2. Pertama yang kita siapkan adalah menambahkan library jQuery Validation dan Sweetalert ke aplikasi web kita. Caranya kita tambahkan link kedua library tersebut ke template.blade.php, library sudah disediakan oleh adminLTE.

3. Selanjutnya Kita modifikasi view user/index.blade.php, kita tambahkan tombol untuk membuat form popup ajax

4. Selanjutnya kita tambahkan kode berikut pada akhir @section('content') pada view user/index.blade.php

```
<div id="myModal" class="modal fade animate shake" tabindex="-1" role="dialog" databackdrop="static"
| data-keyboard="false" data-width="75%" aria-hidden="true"></div>
```

5. Kemudian kita tambahkan kode berikut pada awal @push('js') pada view user/index.blade.php

6. Selanjutkan Kita modifikasi route/web.php untuk mengakomodir operasi ajax

```
function modalAction(url = '') {
    $('#myModal').load(url, function () {
        $('#myModal').modal('show');
    });
}

var dataUser;
$(document).ready(function () {
    dataUser = $('#table_user').DataTable({
```



7. Kemudian Kita tambahkan fungsi create_ajax() pada file UserController.php

```
public function create_ajax(): mixed|View
{
    $level = LevelModel::select('level_id', 'level_nama')->get();
    return view(view: 'user.create_ajax')->with('level', $level);
}
```

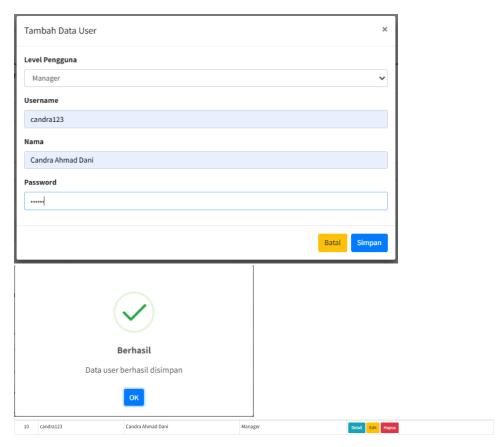
8. Setelah itu, kita buat view baru user/create_ajax.blade.php untuk menampilkan form dengan ajax



9. Kemudian untuk mengakomodir proses simpan data melalui ajax, kita buat fungsi store_ajax() pada UserController.php

10. OK, sekarang kita coba melakukan proses tambah data user menggunakan form ajax. Amati apa yang terjadi dan laporankan pada laporan jobsheet dan commit di github kalian!!!





Praktikum 2 - Modal Ajax Edit Data (Data User)

- 1. Pada Praktikum 2 ini, kita akan melakukan koding untuk proses edit menggunakan ajax.
- 2. Pertama-tama, kita ubah dulu fungsi list() pada UserController.php untuk mengganti tombol edit untuk bisa menggunakan modal



3. Selanjutnya kita modifikasi routes/web.php untuk mengakomodir request edit menggunakan ajax

```
Route::get('/{id}/edit_ajax', [UserController::class, 'edit_ajax']);
Route::put('/{id}/update_ajax', [UserController::class, 'update_ajax']);
```

4. Kemudian, kita buat fungsi edit_ajax() pada UserController.php

```
public function edit_ajax($id): Factory|View
{
    $user = UserModel::find($id);
    $level = LevelModel::select('level_id', 'level_nama')->get();
    return view(view: 'user.edit_ajax', data: ['user' => $user, 'level' => $level]);
}
```

5. Kita buat view baru pada user/edit_ajax.blade.php untuk menampilkan form view ajax

```
asse"modal-body">
v Class="lent alent-danger">
<hb><i class="icon fas fa-ban"></i> Kesalahan!!!</hb>
Data yang anda cari tidak ditemukan
                                               <a href="{{ url(path: '/user') }}" class="btn btn-warning">Kembali</a:
                                                                                               "form-group">
Level Pengguna</label>
name='level_id" id='level_id" class="form-control" required>
tion value=">- Plith Level_-(/option)
makes(\slevel_id = $\slevel_id = \slevel_id) ? 'selected' : '' }} value="{{ $1->level_id }}">
\{ $1->level_name }} \(\slevel_id = \slevel_id) ? 'selected' : '' }} value="\{\slevel_id }}">
\((\slevel_id = \slevel_name )\slevel_id }) \((\slevel_id = \slevel_name ) \slevel_id }) \)
                                                                         cinput value="{{ $user>username }}" type="text" name="username" id="username"
   class="form-control" required>
csmall id="error-username" class="error-text form-text text*mdanger"></small>
                                               //ddv/
ddv class="form-group">
    clabel>Nama:/label>
    cinput value="{{ $user->nama }}" type="text" name="nama" id="nama" class="form-control"
                                                                         \( \text{v} \) \( \text{class="form-group"} \)
\( \text{clabel} \) \( \text{Password} \) \( \text{label} \) \( \text{Password} \) \( \text{label} \) \( \text{Password} \) \( \text{label} \) \( \text{class="form-text text-muted"} \) \( \text{Password} \) \( \text{idal ingin ubah} \)
\( \text{password} \) \( \text{small} \) \( \text{password} \) \( \text{small} \) \( \
ument).ready(function () {
("#form-edit").validate({
```

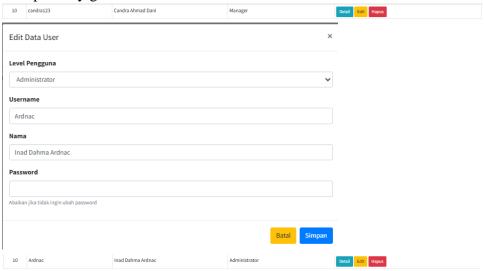


6. Selanjutnya, kita buat fungsi update_ajax() pada UserController.php untuk mengakomodir request update data user melalui ajax

```
function update_ajax(Request $request, $id): JsonResponse|mixed|Redirector|RedirectRes
// cek apakah request dari ajax
if ($request->ajax() || $request->wantsJson()) {
    $rules = [
    'level_id' => 'required|integer',
    'level_id' => 'required|max:20|un
          'username' => 'required|max:20|unique:m_user,username,' . $id . ',user_id',
'nama' => 'required|max:100',
'password' => 'nullable|min:6|max:20'
     $validator = Validator::make($request->all(), $rules);
     if ($validator->fails()) {
         $check = UserModel::find($id);
     if ($check) {
         if (!$request->filled('password')) { // jika password tidak diisi, maka hapus dari request
$request->request->remove('password');
          $check->update($request->all());
          return response()->json([
             'status' => true,
'message' => 'Data berhasil diupdate'
         return response()->json([
              'status' => false,
'message' => 'Data tidak ditemukan'
return redirect(to: '/');
```



7. Sekarang kita coba bagian edit user, amati proses nya. Jangan lupa laporkan dan commit ke repository git kalian!



Praktikum 3 - Modal Ajax Hapus Data (Data User)

- 1. 1. Pada Praktikum 3 ini, kita akan melakukan koding untuk proses hapus menggunakan ajax.
- 2. Pertama-tama, kita ubah dulu routes/web.php untuk mengakomodir request halaman konfirmasi untuk menghapus data

```
Route::delete('/{id}', [UserController::class, 'destroy']);
Route::get('/{id}/delete_ajax', [UserController::class, 'confirm_ajax']);
```

3. Kemudian kita buat fungsi confirm_ajax() pada UserController.php

```
public function confirm_ajax(string $id): Factory|View
{
    $user = UserModel::find($id);
    return view(view: 'user.confirm_ajax', data: ['user' => $user]);
}
```

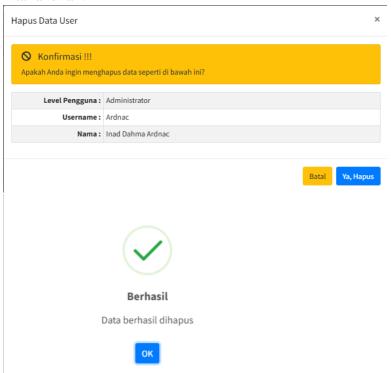
4. Selanjutnya kita view untuk konfirmasi hapus data dengan nama user/confirm_ajax.blade.php





5. Kemudian kita buat fungsi delete_ajax() pada UserController.php untuk mengakomodir request hapus data user

6. Setelah semua selesai, mari kita coba untuk melakukan percobaan dari koding yang telah kita lakukan.



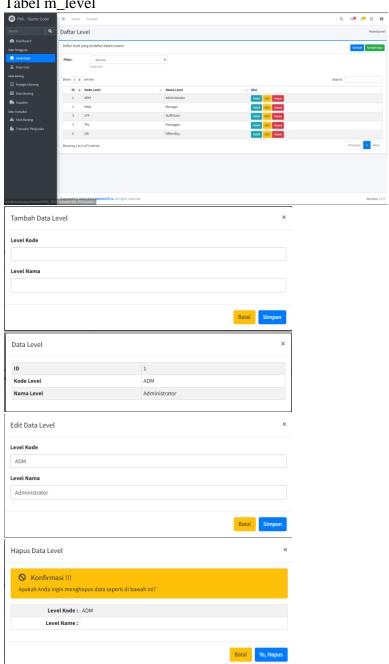
7. Jangan lupa laporkan ke laporan jobsheet dan lakukan commit pada repository git kalian.!!!



Tugas

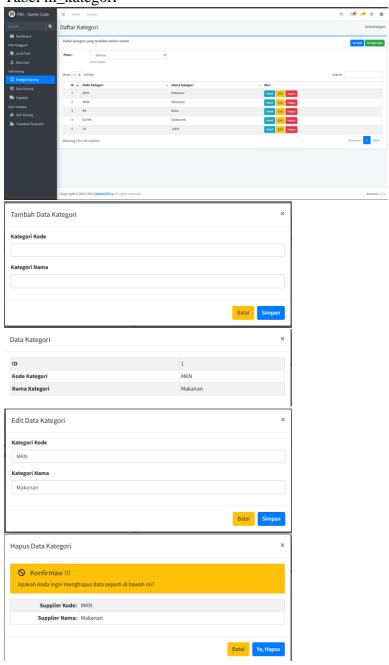
Implementasikan koding untuk Ajax Form dan Client Validation dengan jQuery Validation pada pada menu berikut ini

Tabel m_level



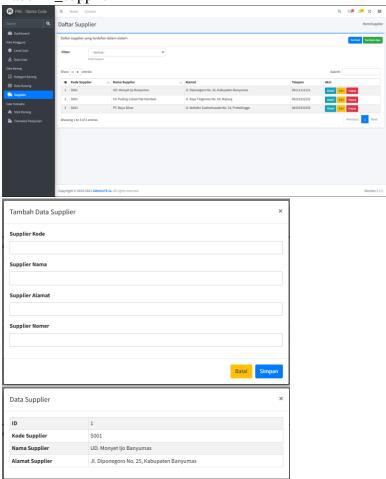


✓ Tabel m_kategori

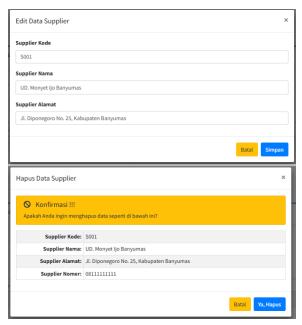




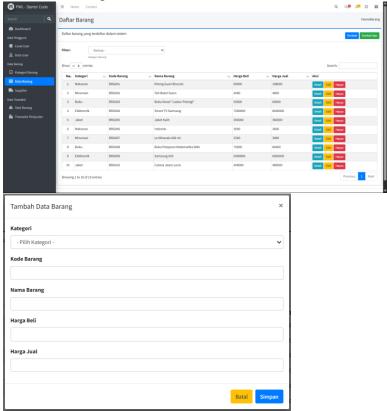
✓ Tabel m_supplier



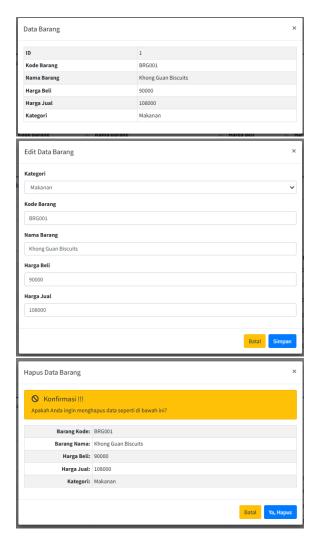




✓ Tabel m_barang







Laporkan pada laporan jobsheet dan Jangan lupa di commit dan push pada repository git kalian.