



Workshop Keprofesian

PEMANFAATAN AUTH LOGIN WITH

GOOGLE PADA FRAMEWORK

LARAVEL /PERTEMUAN I



PEMANFAATAN AUTH LOGIN WITH GOOGLE PADA FRAMEWORK LARAVEL /PERTEMUAN I

Robi Wariyanto Abdullah ,M.Kom
@kursus_komputer_soloraya
robiwariyanto@gmail.com

Persiapan/Instalasi Tool Yang diperlukan



Instalasi Web Server

Laragon atau Xampp atau yang lain



Instal Composer

Composer adalah aplikasi yang diinstall ke perangkat untuk memfasilitasi developer menggunakan *library open source* milik orang lain ke dalam *project* yang sedang dibangun



Instalasi Git

GIT adalah sebuah tools bagi para programmer dan developer yang berfungsi sebagai control system untuk menjalankan proyek pengembangan software



Instalasi Framework Laravel

Laravel adalah satu-satunya framework yang membantu untuk memaksimalkan penggunaan PHP di dalam proses pengembangan website



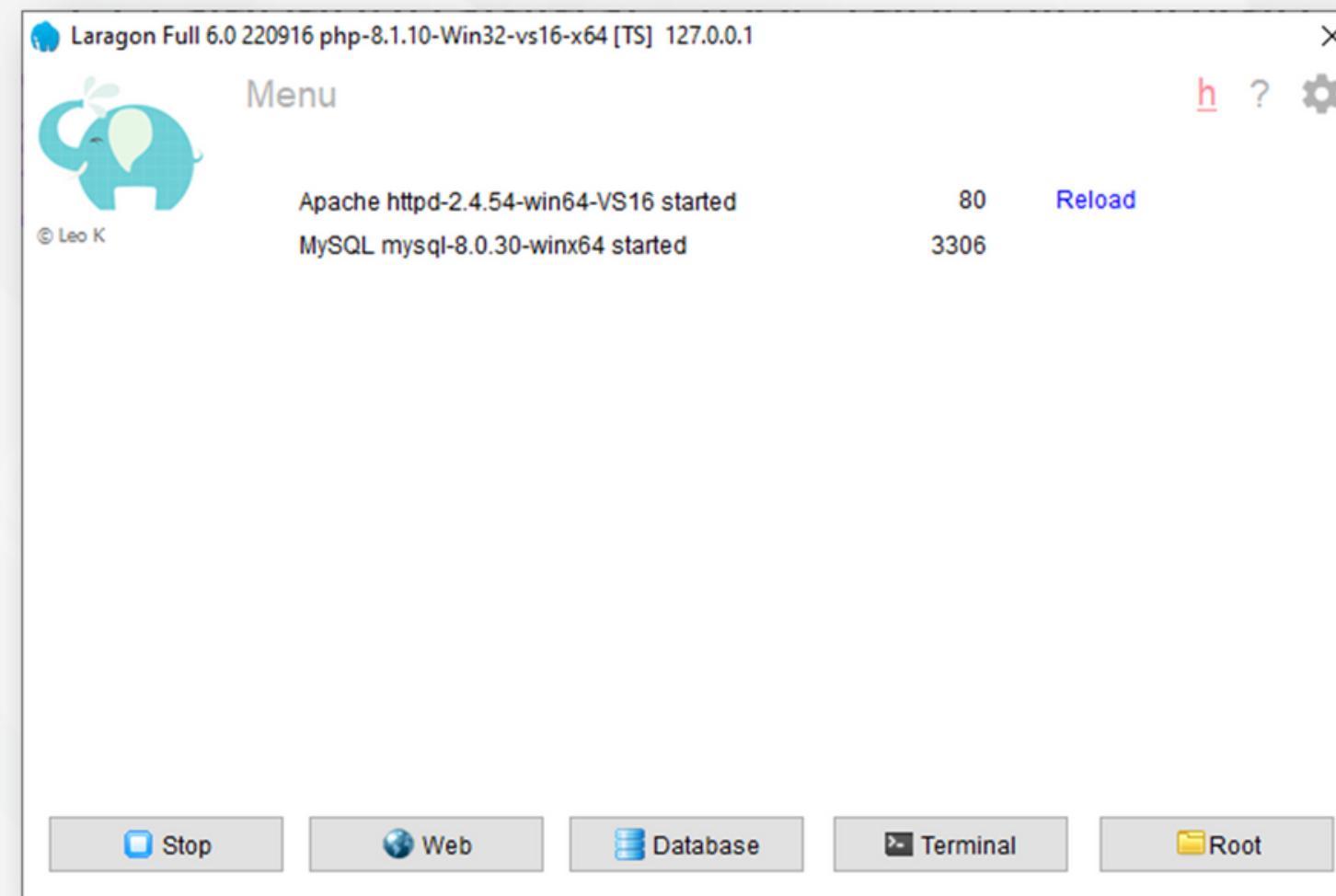
Instalasi Editor

Visual code atau sublime atau editor lainnya

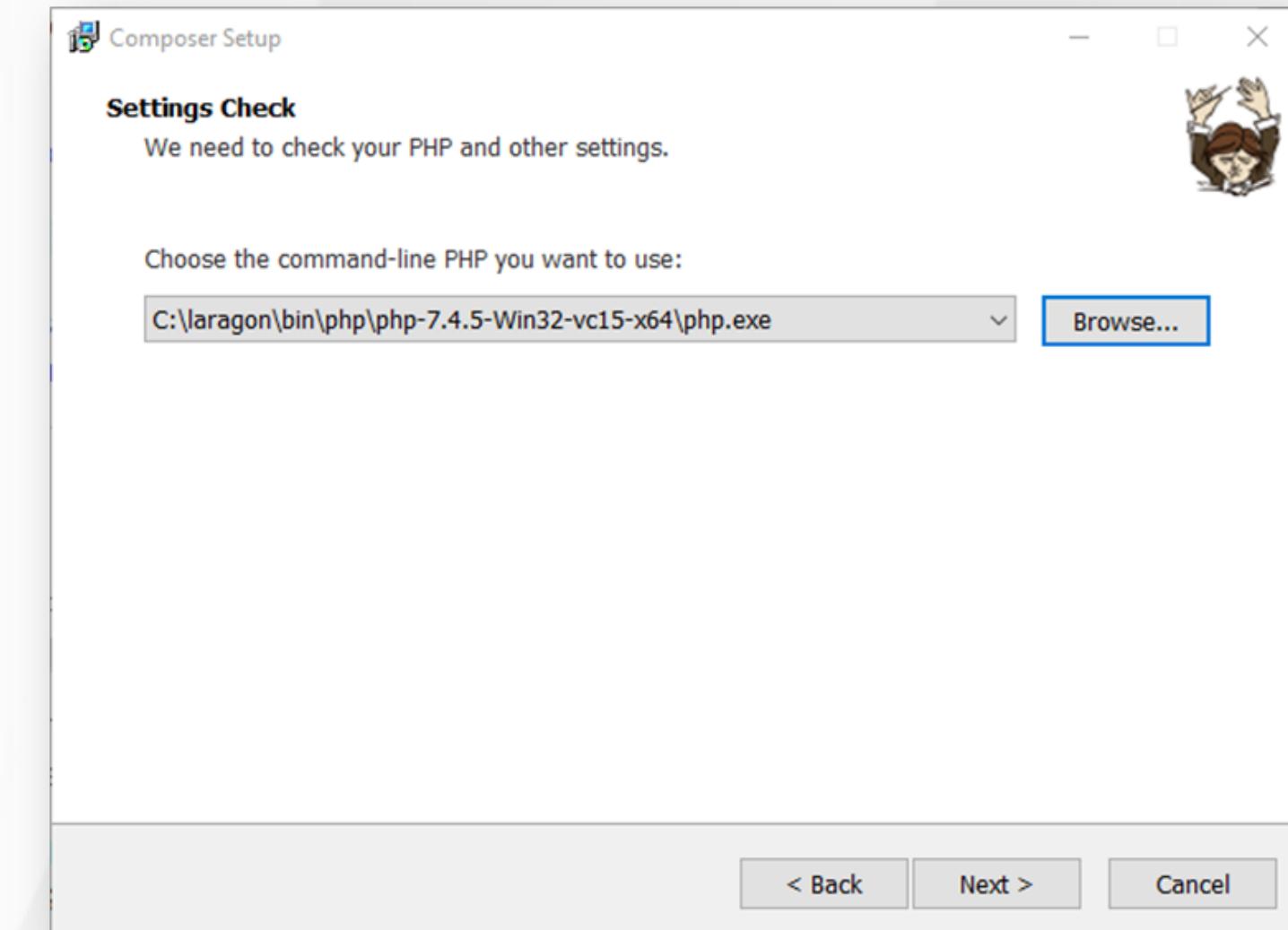


Instalasi Web Server Laragon

<https://laragon.org/download/>



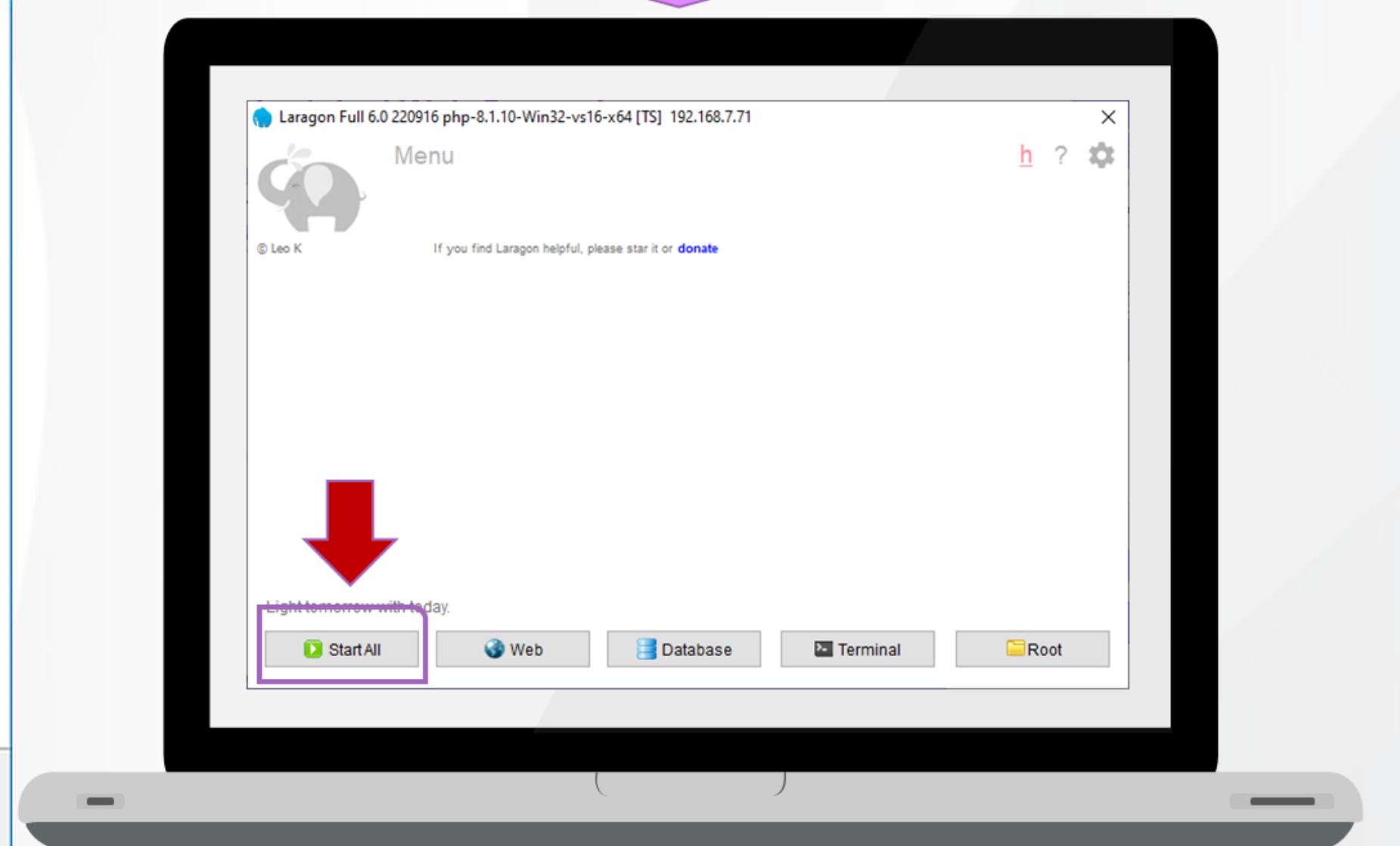
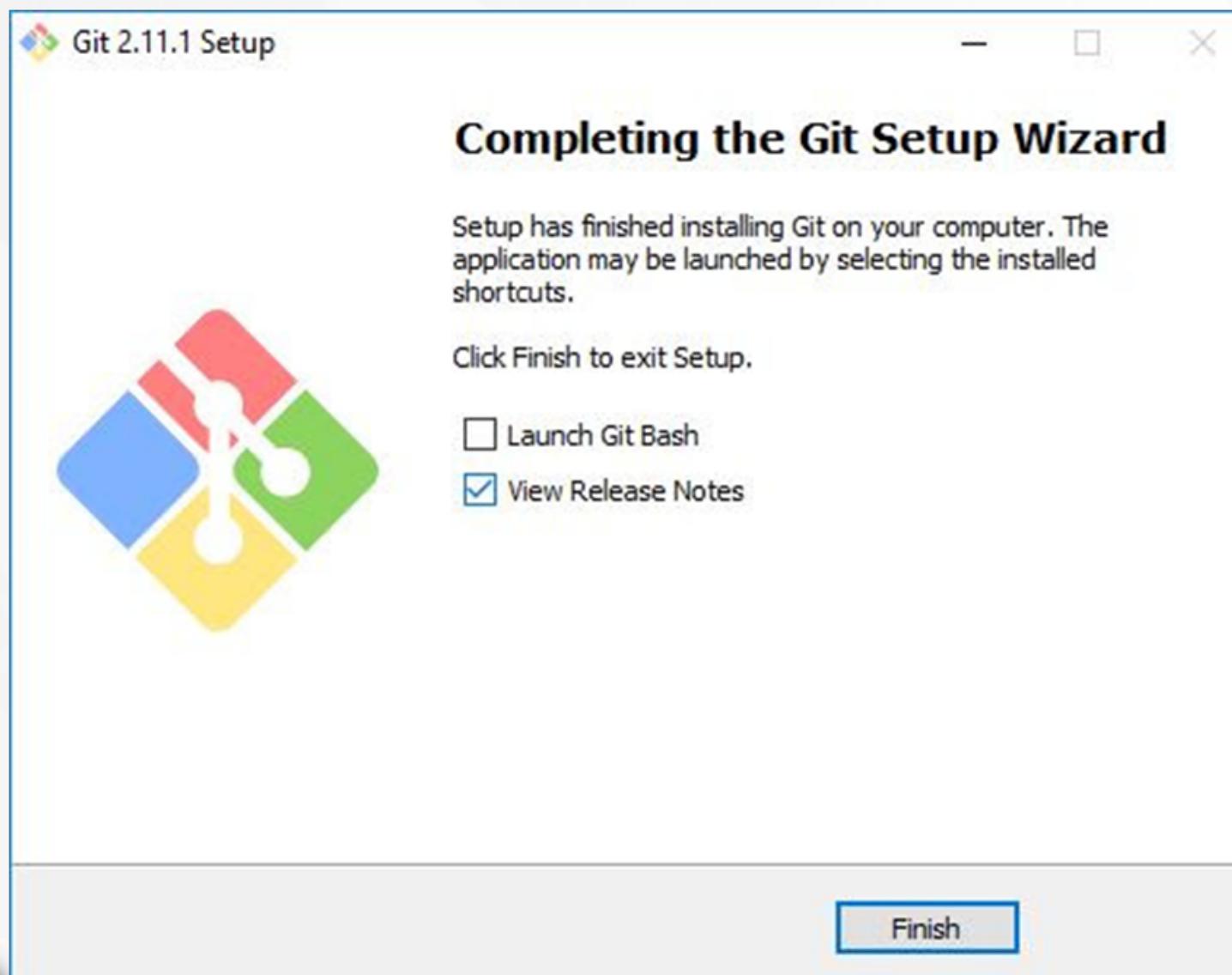
Instalasi Composer



<https://getcomposer.org/download/>

Instalasi git dan laravel

<https://git-scm.com/downloads>



Instalasi Laravel

Instalasi phpmyadmin web server pada laragon

<https://www.phpmyadmin.net/>

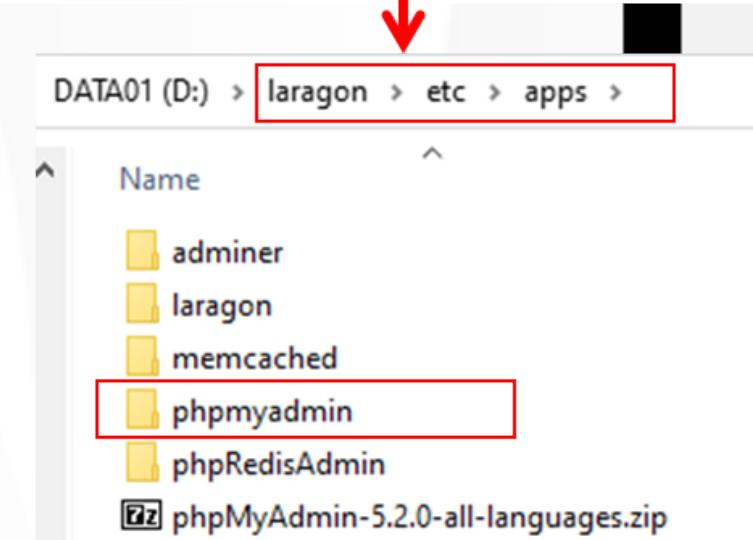
phpmyadmin.net

Tube Maps



Bringing MySQL to the web

about



1. Download phpmyadmin pada <https://www.phpmyadmin.net/>
2. Extract hasil downloadan tadi pada folder instalasi laragon kalian. **Laragon=>etc=>apps**
3. Rename hasil extractkan tadi dengan nama folder menjadi **phpmyadmin**

Download 5.2.1

Try demo

Donate

Sponsors

Instalasi dan Konfigurasi Laravel

Instalasi Laravel pada Webserver Laragon

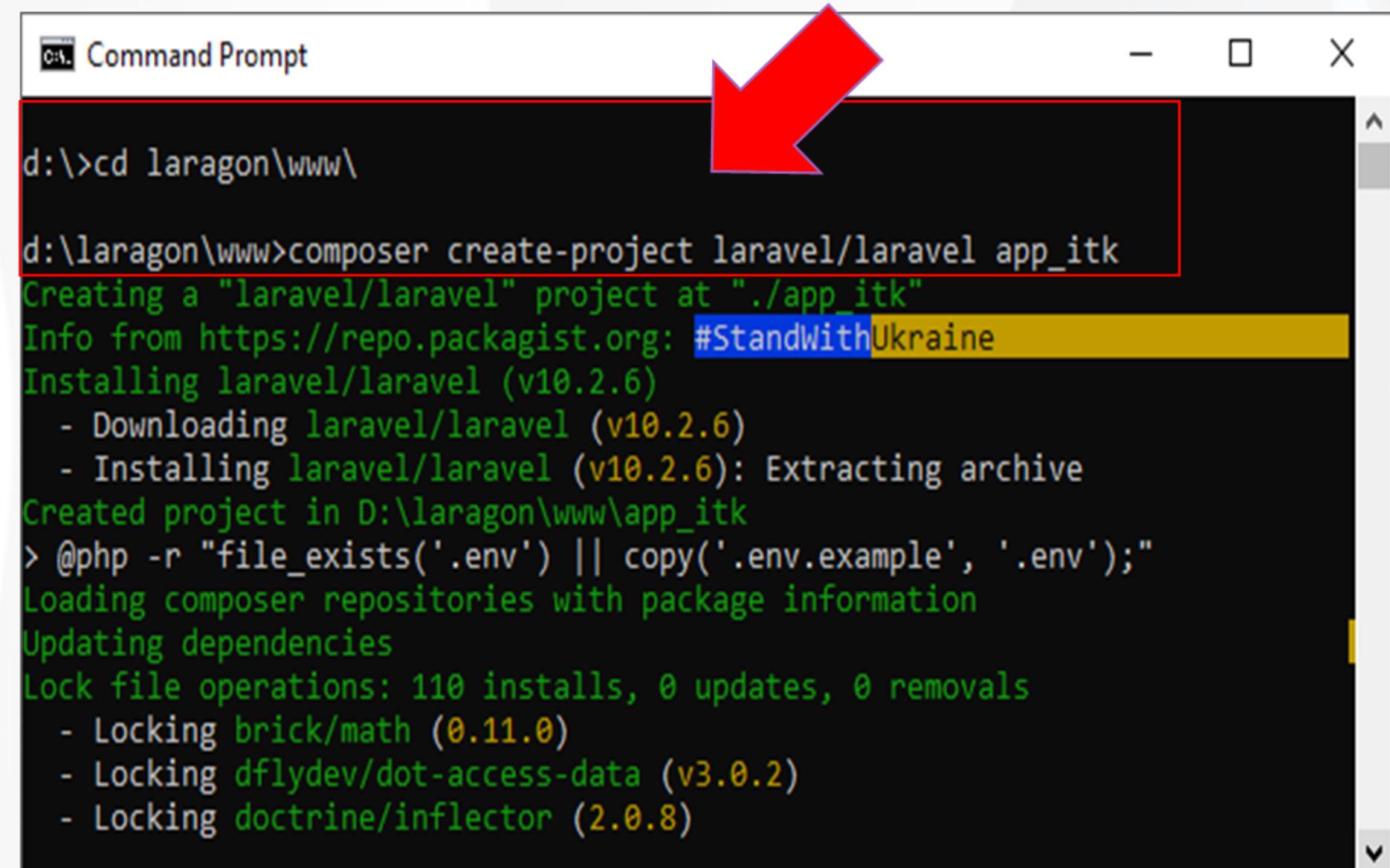
1. Buka Aplikasi command prompt
2. masuk ke drive dimana webserver diinstal.

Contoh pada drive d. kemudian masuk ke folder root web server . Jika menggunakan laragon maka bisa masuk ke drive instalasi laragon dan masuk pada folder laragon\www\ dengan perintah

cd laragon\www.

Jika menggunakan webserver XAMPP maka bisa masuk pada folder xampp/htdocs

3. Untuk instal laravel dapat diketikan perintah **composer create-project laravel/laravel namaproject**
4. Buka editor visual code dan buka project laravel yang sudah terinstal.



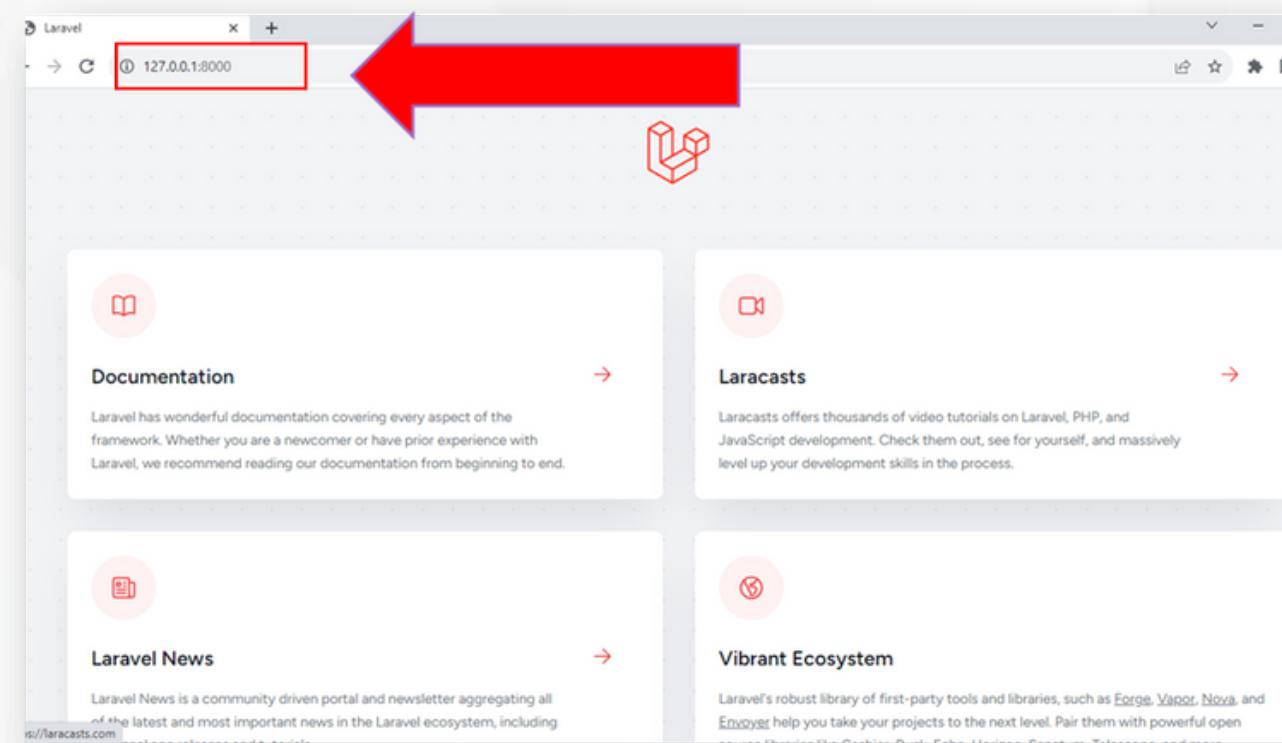
```
d:\>cd laragon\www\

d:\laragon\www>composer create-project laravel/laravel app_itk
Creating a "laravel/laravel" project at "./app_itk"
Info from https://repo.packagist.org: #StandWithUkraine
Installing laravel/laravel (v10.2.6)
- Downloading laravel/laravel (v10.2.6)
- Installing laravel/laravel (v10.2.6): Extracting archive
Created project in D:\laragon\www\app_itk
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 110 installs, 0 updates, 0 removals
- Locking brick/math (0.11.0)
- Locking dflydev/dot-access-data (v3.0.2)
- Locking doctrine/inflector (2.0.8)
```

Instalasi dan konfigurasi laravel

Instalasi Laravel pada Webserver **Laragon**

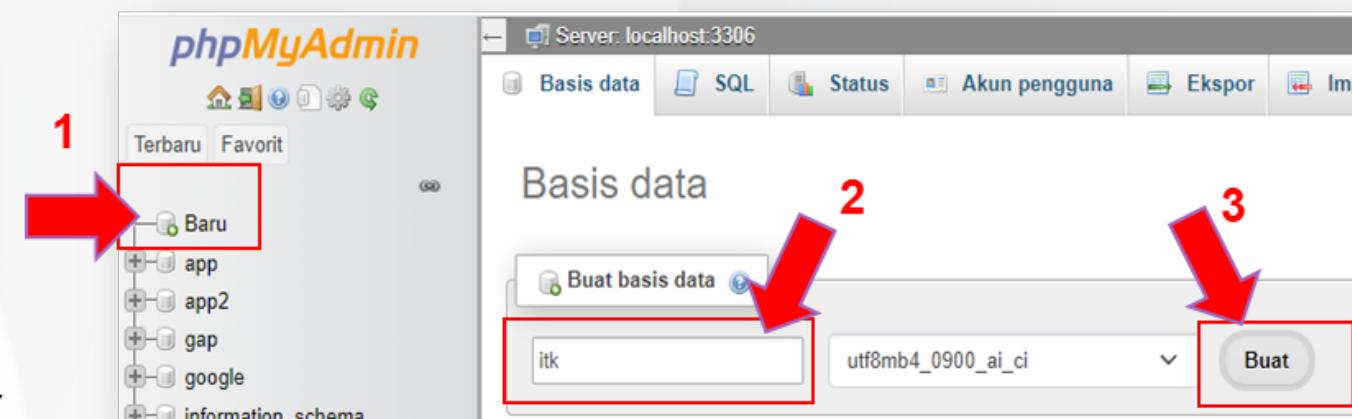
5. Setelah project terbuka pilih pada bagian terminal . Pilih new terminal
6. Pada bagian terminal ketikan perintah sebagai berikut
php artisan serve
setelah mendapatkan url hasil output perintah diatas maka buka browser, kemudian ketikan url dari hasil output yang didapatkan . Jika browser sudah terlihat seperti gambar dibawah maka proses instalasi laravel sudah berhasil

A screenshot of Visual Studio Code. The left sidebar shows the project structure under "APP_ITK". The "Terminal" tab is active, showing the command "PS D:\laragon\www\app_itk> php artisan serve" and the output "INFO Server running on [http://127.0.0.1:8000]. Press Ctrl+C to stop the server". A red arrow points from the browser's address bar in the previous screenshot to the "Terminal" tab here.

Membuat database dan Konfigurasi

1 Membuat Database

1. Buka localhost/phpmyadmin pada web browser atau bisa gunakan tool database lainnya
2. isikan nama database yang akan digunakan kemudian tekan tombol buat seperti gambar disamping
3. Buka editor visual code kembali. Dan buka file **.env** pada project yang telah dibuat
4. Ubah pada nama databse yang sudah dibuat sebelumnya .
5. sesuaikan username , password database pada webserver kalian masing-masing. Pada contoh kasus ini saya gunakan nama database **itk**



```

APP_ITK=true
APP_URL=http://localhost
LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug
DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=itk
DB_USERNAME=root
DB_PASSWORD=
BROADCAST_DRIVER=log
CACHE_DRIVER=file
FILESYSTEM_DISK=local
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120
MEMCACHED_HOST=127.0.0.1
REDIS_HOST=127.0.0.1

```

Membuat database dan Konfigurasi

```
File Edit Selection View Go Run Terminal Help
EXPLORER
  APP_ITK
    app
      Console
      Exceptions
      Http
      Models
      Providers
      bootstrap
      config
    database
      factories
    migrations
      2014_10_12_000000_create_users_table.php
      2014_10_12_100000_create_password_reset_tokens_table.php
      2019_08_19_000000_create_failed_jobs_table.php
      2019_12_14_000001_create_personal_access_tokens_table.php
      seeders
      .gitignore
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS D:\laragon\www\app_itk> **php artisan migrate**

INFO Preparing database.

Creating migration table

INFO Running migrations.

2014_10_12_000000_create_users_table
 2014_10_12_100000_create_password_reset_tokens_table
 2019_08_19_000000_create_failed_jobs_table
 2019_12_14_000001_create_personal_access_tokens_table

1 Membuat Database

1. Buka kembali ke editor visual code dan perhatikan pada folder **app->database->migrations**

Pada saat instal laravel kita sudah diberikan blue print tabel user sample pada bawaan laravel sehingga kita bisa menggunakan atau tidak yang telah disediakan tersebut.

- 2.Nah untuk dapat menggunakan dan bisa tercreate dalam database yang kita buat maka kita bisa masuk pada terminal dan pilih new terminal kemudian ketikan perintah **php artisan serve**

- 3.Jika perintah berhasil dijalankan maka tabel-tabel akan otomatis tercreate pada database yang telah kita pilih. Untuk dapat melihatnya silahkan cek pada localhost/ phpmyadmin pada browser kalian dan pilih databse yang tadi sudah disetting pada **.env**. Dalam database tersebut akan terbentuk tabel-tabel seperti pada gambar dibawah

Tabel	Tindakan	Baris	Jenis	Penyortiran	Ukuran	Beban
failed_jobs	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
migrations	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	4	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
password_reset_tokens	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
personal_access_tokens	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
users	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
5 tabel		Jumlah				
4 InnoDB utf8mb4_0900_ai_ci 80.0 KB 0 8						

Membuat database dan Konfigurasi

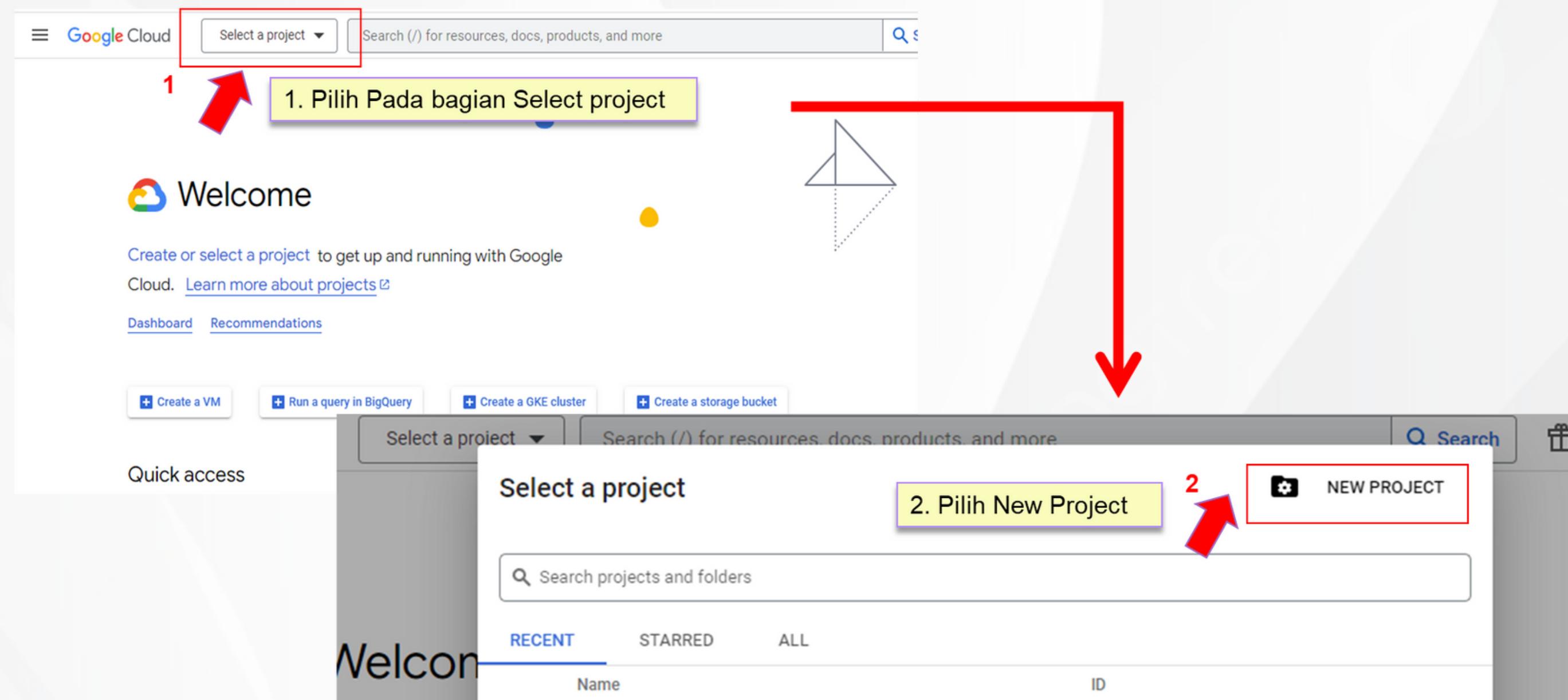
4. Menambahkan 1 field pada tabel users yaitu field **google_id** atau nama lainnya juga bisa tinggal menyesuaikan. Buka web browser Masuk pada localhost/phpmyadmin.

The screenshot shows the PHPMyAdmin interface with the following steps highlighted:

- 1. Pilih database yang telah dibuat**: A red arrow points to the 'itk' database in the left sidebar. A callout box says "1. Pilih database yang telah dibuat".
- 2. Klik pada table user**: A red arrow points to the 'users' table in the main list. A callout box says "2. Klik pada table user".
- Klik pada struktur**: A red arrow points to the 'Struktur' tab in the top menu bar. A callout box says "Klik pada struktur".
- Berikan tanda centang pada kolom tak ternilai.**: A red arrow points to the 'Tak Ternilai' checkbox for the 'google_id' column. A callout box says "Berikan tanda centang pada kolom tak ternilai.". Number 5 is next to the checkbox.
- Pilih kolom penempatan field google_id**: A red arrow points to the 'Kolom' dropdown at the bottom right of the table structure. A callout box says "Pilih kolom penempatan field google_id". Number 4 is next to the dropdown.
- Berikan nilai google_id atau yang lainnya dan pilih jenis yaitu text**: A red arrow points to the 'Jenis' dropdown set to 'TEXT'. A callout box says "Berikan nilai google_id atau yang lainnya dan pilih jenis yaitu text". Number 6 is next to the dropdown.
- Tekan button simpan**: A red arrow points to the 'Simpan' button at the bottom left. A callout box says "Tekan button simpan". Number 7 is next to the button.
- Lakukan hal yang sama seperti langkah 3-7 dengan menambahkan fiel **level** dengan atribut boleh text atau boleh int sesuaikan dengan kebutuhan kalian dalam rancangan system yang akan kalian gunakan**: A red arrow points to the bottom right corner. A callout box says "8 Lakukan hal yang sama seperti langkah 3-7 dengan menambahkan fiel **level** dengan atribut boleh text atau boleh int sesuaikan dengan kebutuhan kalian dalam rancangan system yang akan kalian gunakan".

Konfigurasi Project pada Google Cloud Platform

- Siapkan project Google Cloud Platform (GCP) terlebih dahulu. Silahkan dibuka **google cloud console** pada alamat <https://console.cloud.google.com> dan buat project baru.



Konfigurasi Project pada Google Cloud Platform

≡ Google Cloud Search (/) for resources, docs, prod

New Project

3

Project name * appitk

3. Berikan project Name. pada latihan ini saya berikan dengan nama **appitk**

Project ID appitk It cannot be changed later. EDIT

?

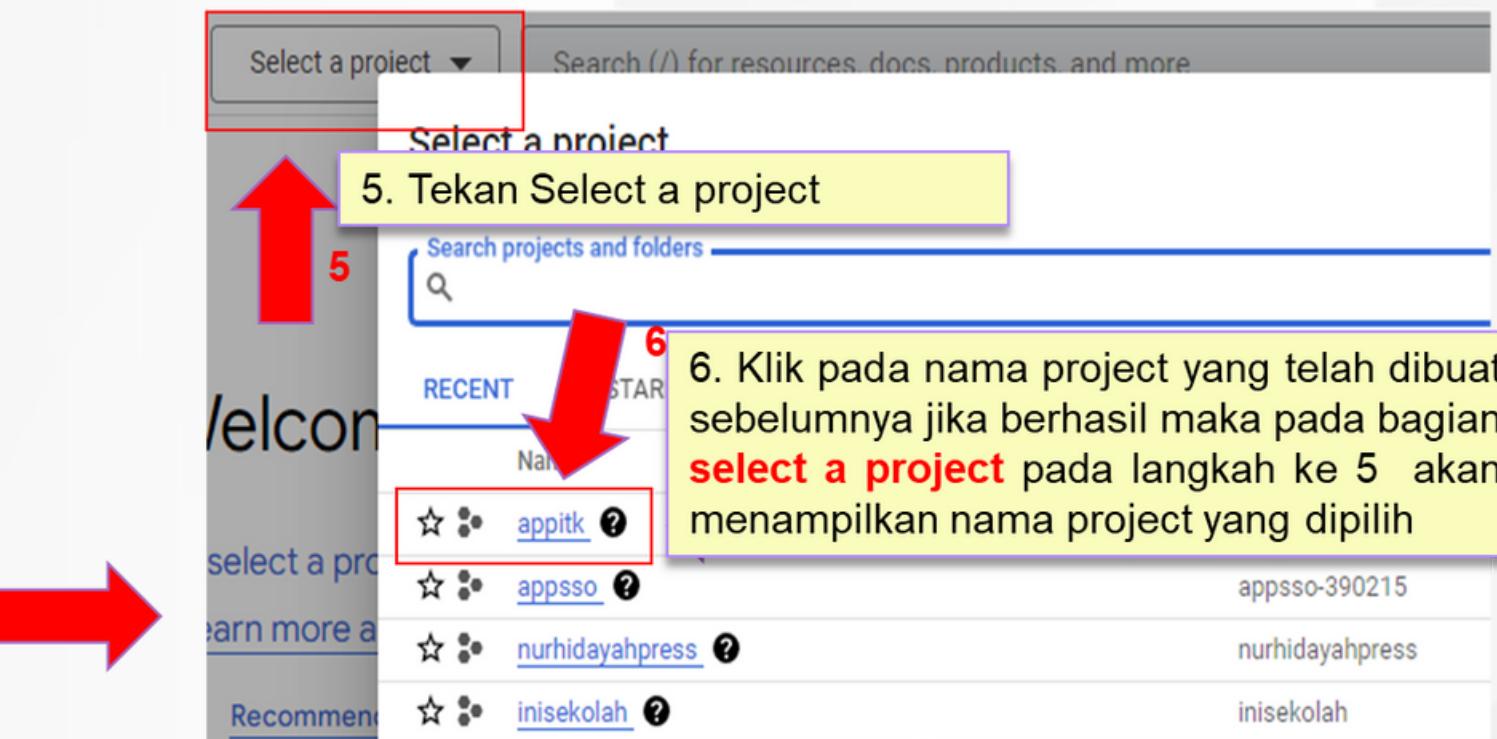
Location * No organization BROWSE

4

Parent organization or folder

4. Tekan Tombol Create

CREATE CANCEL



Welcome

You're working in appitk

Project number: 718385604052

Project ID: appitk

Dashboard Recommendations

Create a VM Run a query in BigQuery Create a GKE

Quick access

API APIs & Services

Konfigurasi Project pada Google Cloud Platform

8. Pilih pada bagian Oauth consent screen

9. Pilih pada external

10. Tekan tombol create

11. Isikan data pada app information yang diperlukan tanda (*) wajib diisi

Konfigurasi Project pada Google Cloud Platform

The screenshot shows the 'Edit app registration' page under 'APIs & Services'. The 'OAuth consent screen' tab is selected. A red box highlights the 'SAVE AND CONTINUE' button at the bottom right of the page.

12.Klik pada save and continue

12

SAVE AND CONTINUE CANCEL

The screenshot shows the 'Test users' configuration page. A red box highlights the '+ ADD USERS' button. A yellow box contains the instruction: '13.Klik pada add user'.

13.Klik pada add user

+ ADD USERS

Filter Enter property name or value

User information
No rows to display

SAVE AND CONTINUE CANCEL

The screenshot shows the 'Add users' dialog box. A red box highlights the input field where 'robiwariyanto@gmail.com' is entered. A yellow box contains the instruction: '14.Isikan email kalian dan tekan tombol add'. An arrow points from the 'Add users' dialog to the input field.

14.Isikan email kalian dan tekan tombol add

robiwariyanto@gmail.com 1 / 100

ADD

Konfigurasi Project pada Google Cloud Platform

OAuth consent screen — Scopes — 3 Test users — 4 Summary

Test users

While publishing status is set to "Testing", only test users are able to access the app. Allowed user cap prior to app verification is 100, and is counted over the entire lifetime of the app. [Learn more](#)

+ ADD USERS

Filter Enter property name or value

User information	robiwariyanto@gmail.com
------------------	-------------------------

SAVE AND CONTINUE CANCEL

15 15.Klik pada save and continue

Edit app registration

OAuth consent screen — Scopes — Test users — Summary

OAuth consent screen

User type External
App name apptk
Support email robiwariyanto@gmail.com
App logo 

Application homepage link Not provided
Application privacy policy link Not provided
Application terms of service link Not provided
Authorized domains Not provided
Contact email addresses robiwariyanto@gmail.com

Scopes

API ↑	Scope	User-facing description
No rows to display		

Test users

1 user (1 test, 0 other) / 100 user cap

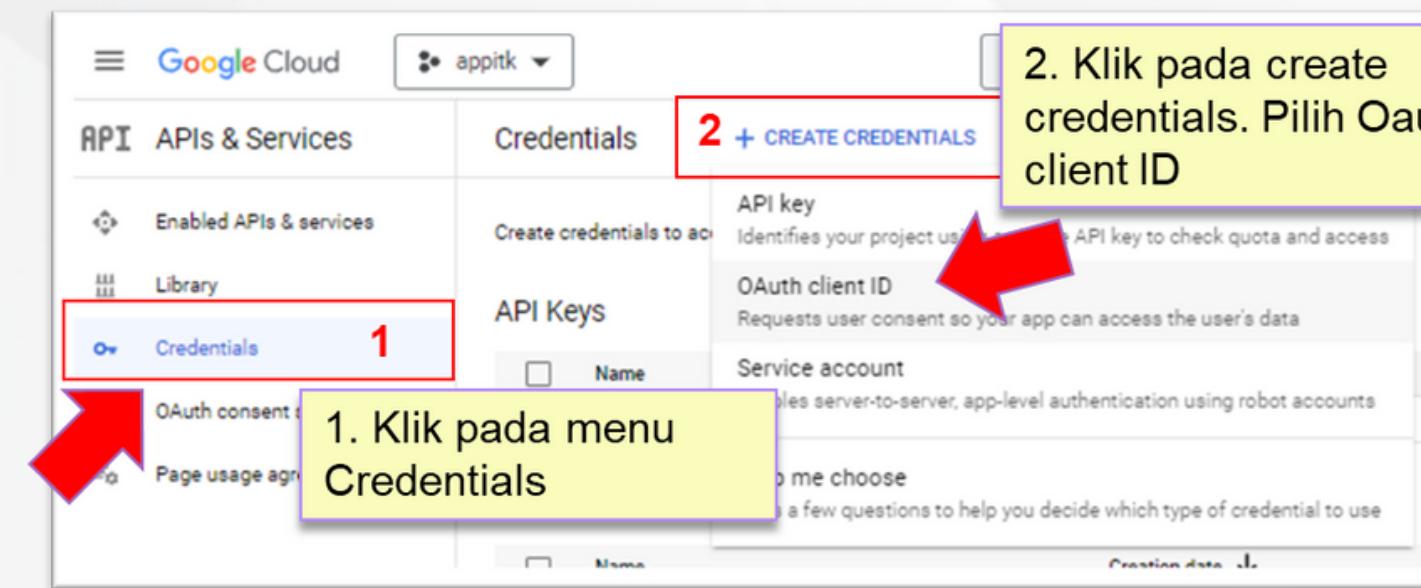
Filter Enter property name or value

User information	robiwariyanto@gmail.com
------------------	-------------------------

BACK TO DASHBOARD

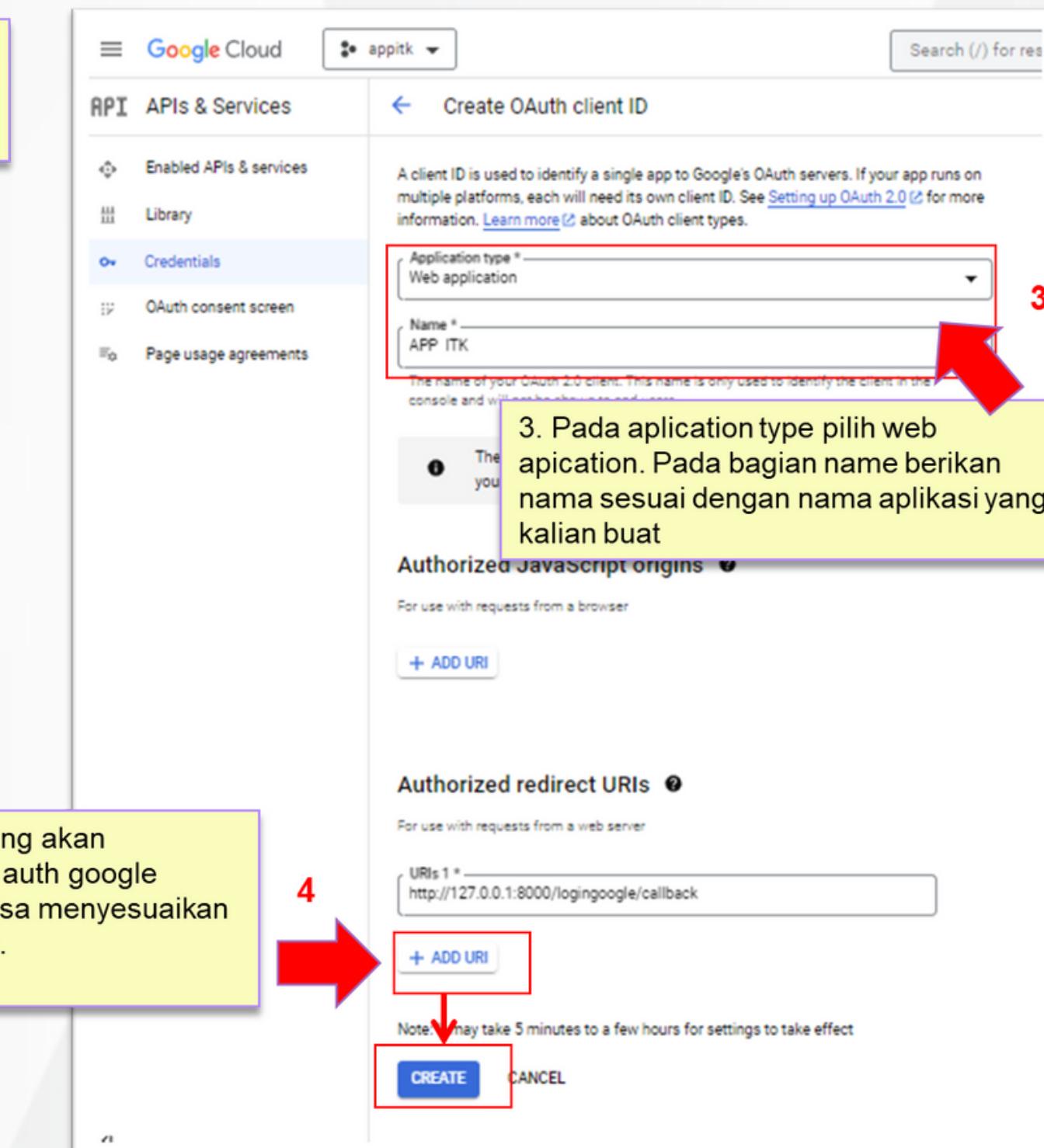
16 16.Klik button back to dashboard jika data sudah benar dan bisa klik edit tombol diatas jika masih ada yang ingin diedit

Konfigurasi Kredential Project pada Google Cloud Platform



1. Klik pada menu **Credentials**

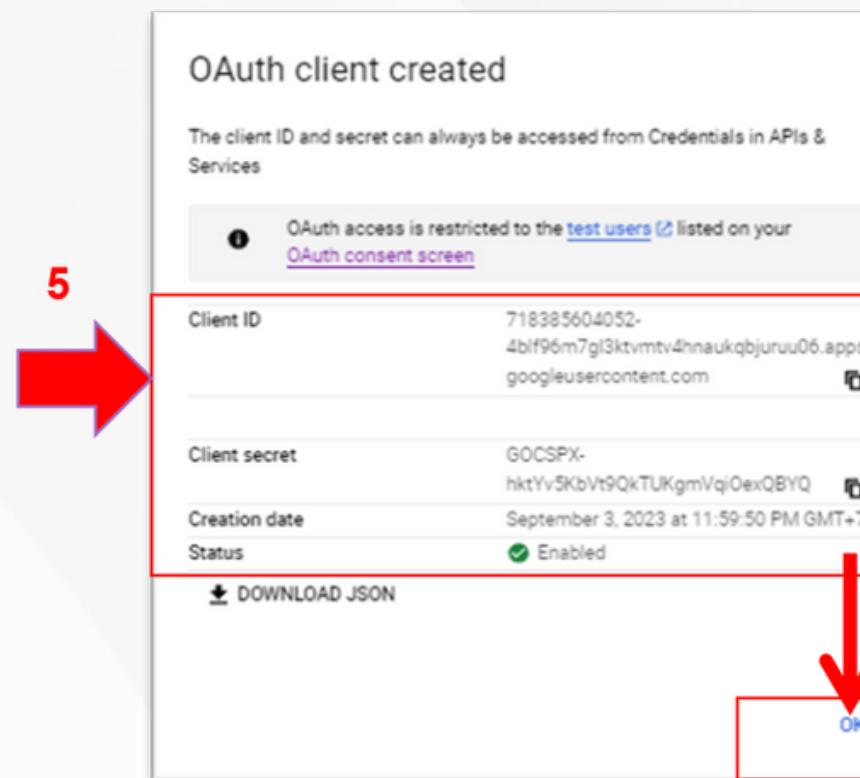
2. Klik pada create credentials. Pilih Oauth client ID



3. Pada application type pilih web application. Pada bagian name berikan nama sesuai dengan nama aplikasi yang kalian buat

4. Ketikan url untuk yang akan menerima respon dari auth google seperti contoh atau bisa menyesuaikan url pada project kalian. Tekan tombol create

Konfigurasi Kredential Project pada Google Cloud Platform



5. Catat client ID dan client secret yang diperoleh setelah tekan tombol create pada langkah sebelumnya. Klik ok

Google Cloud appitk

API APIs & Services Credentials + CREATE CREDENTIALS DELETE RESTORE DELETED CREDENTIALS

Enabled APIs & services Create credentials to access your enabled APIs. [Learn more](#)

Library

Credentials

OAuth consent screen

Page usage agreements

API Keys

No API keys to display

OAuth 2.0 Client IDs

Name	Creation date	Type	Client ID	Action
APP ITK	Sep 3, 2023	Web application	718385604052-4blf...	

6. Data Oauth yang berhasil dibuat akan tampil seperti gambar disamping. Jika ingin melihat id client dan secret atau mengubah data Oauth yang telah dibuat maka dapat diklik pada pensil diisamping

7. Pada tombol ini berfungsi untuk edit dan delete maupun download data Oauth yang telah dibuat

Instal dan konfigurasi Socialite

- Buka Kembali file project laravel yang sudah diinstal dengan visual code . Kemudian buka pada bagian terminal . Ketikan perintah sebagai berikut :

1. **composer require laravel/socialite**

```
PS D:\laragon\www\app_itk> composer require laravel/socialite
Info from https://repo.packagist.org: #standwithUkraine
./composer.json has been updated
Running composer update laravel/socialite
Loading composer repositories with package information
Updating dependencies
Lock file operations: 2 installs, 0 updates, 0 removals
- Locking laravel/socialite (v5.9.0)
- Locking league/oauth1-client (v1.10.1)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 2 installs, 0 updates, 0 removals
- Downloading laravel/socialite (v5.9.0)
- Installing league/oauth1-client (v1.10.1): Extracting archive
- Installing laravel/socialite (v5.9.0): Extracting archive
Generating optimized autoload files
```

2. Buka File pada project anda pada folder config->service.php

3. Tambahkan kode pada file service.php seperti berikut

```
'google'=> [ 'client_id'=> env('GOOGLE_CLIENT_ID'),
    'client_secret'=> env('GOOGLE_CLIENT_SECRET'),
    'redirect'=> env('GOOGLE_REDIRECT_URI'), ],
    /*
    'google' =>
        [
            'client_id' => env('GOOGLE_CLIENT_ID'),
            'client_secret' => env('GOOGLE_CLIENT_SECRET'),
            'redirect' => env('GOOGLE_REDIRECT_URI'),
        ],

```

Instal dan konfigurasi Socialite

```
.env
.env
57 VITE_PUSHER_PORT="${PUSHER_PORT}"
58 VITE_PUSHER_SCHEME="${PUSHER_SCHEME}"
59 VITE_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
60
61 GOOGLE_CLIENT_ID=718385604052-4blf96m7gl3ktvmtv4hnauqbjuruu06.apps.googleusercontent.com
62 GOOGLE_CLIENT_SECRET=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
63 GOOGLE_REDIRECT_URL=http://127.0.0.1:8000/logingoogle/callback
64
```

4. Buka file `.env` pada folder root project kalian.tambahkan data
`GOOGLE_CLIENT_ID`= google id kalian ,
`GOOGLE_CLIENT_SECRET`= client screen kalian
`GOOGLE_REDIRECT_URL` = sesuaikan url callback pada settingan konfigurasi console yang sudah disetting pada slide 15

5. Jika kalian lupa data google client id, screet id kalian maka dapat dibuka pada <https://console.cloud.google.com/> . pilih pada bagian api dan service kemudian pilih pada **credential**. Seperti gambar dibawah

API APIs & Services

← Client ID for Web application

Enabled APIs & services

Library

Credentials

OAuth consent screen

Page usage agreements

Client ID for Web application

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

The domains of the URIs you add below will be automatically added to your OAuth consent screen as authorized domains.

Authorized JavaScript origins

For use with requests from a browser

+ ADD URI

Authorized redirect URIs

URIs 1 *

http://127.0.0.1:8000/logingoogle/callback

Google client id

Client ID	718385604052-4blf96m7gl3ktvmtv4hnauqbjuruu06.apps.googleusercontent.com
Creation date	September 3, 2023 at 11:59:50 PM GMT+7

Client secrets

Google screet id

Client secret	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
Creation date	September 3, 2023 at 11:59:50 PM GMT+7
Status	Enabled

Instal dan konfigurasi Socialite

1. Sekarang kita buka file pada project kalian **app/Models/User.php** dan modifikasi attribute **\$fillable** menjadi seperti ini:

```
<?php  
namespace App\Models;  
  
use Illuminate\Contracts\Auth\MustVerifyEmail;  
use Illuminate\Database\Eloquent\Factories\HasFactory;  
use Illuminate\Foundation\Auth\User as Authenticatable;  
use Illuminate\Notifications\Notifiable;  
use Laravel\Sanctum\HasApiTokens;  
  
class User extends Authenticatable  
{  
    use HasApiTokens, HasFactory, Notifiable;  
  
    /**  
     * The attributes that are mass assignable.  
     *  
     * @var array<int, string>  
     */  
    protected $fillable = [  
        'name',  
        'email',  
        'password',  
        'google_id',  
        'level'  
    ];
```

Instal dan konfigurasi Socialite

3. Membuat Controller dengan nama controller yaitu **LoginGoogle**. Buka project kalian pada editor visual studio code . Kemudian masuk pada bagian terminal dan ketikan perintah berikut :

php artisan make:controller LoginGoogle

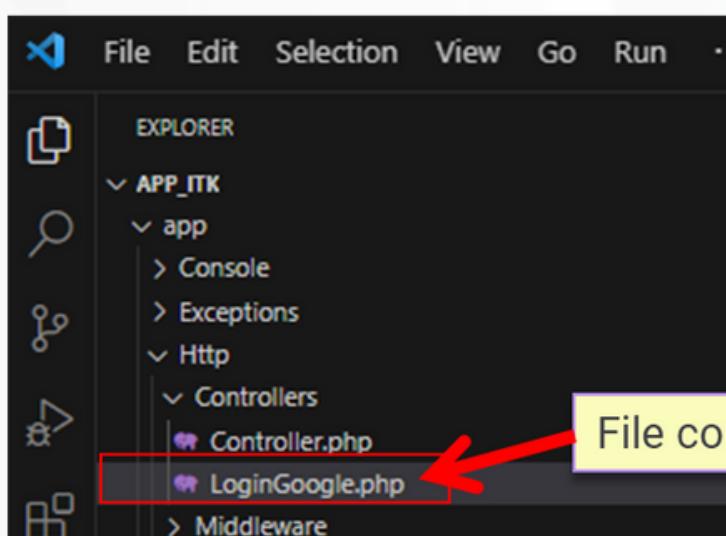
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

INFO No publishable resources for tag [laravel-assets].
No security vulnerability advisories found
Using version ^5.9 for laravel/socialite
PS D:\laragon\www\app_itk> php artisan make:controller LoginGoogle
INFO Controller [D:\laragon\www\app_itk\app\http\Controllers\LoginGoogle.php] created successfully.

PS D:\laragon\www\app_itk>
```

3

LoginGoogle merupakan nama controller yang akan dibuat. Sesuaikan nama controller kalian yang akan digunakan



File controller akan terbentuk pada folder **app->http->controller**

Instal dan konfigurasi Socialite

```
app > Http > Controllers > LoginGoogle.php
1  <?php
2
3  namespace App\Http\Controllers;
4  use App\Models\User;
5  use Laravel\Socialite\Facades\Socialite;
6  use Illuminate\Support\Facades\Auth;
7  use Illuminate\Support\Facades\Hash;
8  use Illuminate\Http\Request;
9
10 class LoginGoogle extends Controller
11 {
12     //
13
14     public function redirectGoogle()
15     {
16         return Socialite::driver('google')->redirect();
17     }
18 }
```

4

5

b. method dengan nama
method/function **callback** untuk
menangkap hasil respon yang diberikan
Apii google saat terhubung dan
mengembalikan respon balik dari google

4. Tambahkan script seperti gambar disamping pada line 4,5,6,7 . Hal tersebut perlu di load dikarenakan akan kita gunakan method/function yang ada pada class User, Socialite, Auth dan Hash

5. Buat 4 method / function pada controller LoginGoogle diantaranya :

a. method dengan nama method/function **redirectGoogle** atau boleh dengan nama lain tinggal disesuaikan sesuai keinginan.

```
19
20     public function callback()
21     {
22         // Google user object dari google
23         try {
24             $user = Socialite::driver('google')->stateless()->user();
25             // $userFromDatabase = User::where('google_id', $user->getId())->first();
26             $finduser = User::where('email', $user->email)->first();
27
28             if($finduser){
29
30                 Auth::login($finduser);
31
32                 $finduser->google_id = $user->getId();
33                 $finduser->save();
34
35                 return redirect()->intended('dashboard');
36             }
37             else{
38
39                 return redirect('/')->withErrors(['error' => 'Email Belum Terdaftar']);
40             }
41         } catch (Exception $e) {
42
43             return redirect('/')->withErrors(['error' => 'Terjadi Kesalahan Saat Terhubung Google']);
44         }
45     }
46 }
47
48
49
50
51 }
```



Instal dan konfigurasi Socialite

```

53
54     public function logout(Request $request)
55     {
56         Auth::logout();
57         $request->session()->invalidate();
58         $request->session()->regenerateToken();
59
60         return redirect('/');
61     }
62

```

c. method dengan nama method/function **logout** menghapus session yang telah dibuat ketika login berhasil. Nama method tidak harus logout bias disesuaikan sesuai keinginan

```

104    public function dashboard(){
105        $form="beranda_admin";
106        return view('dashboard',compact('form'));
107    }
108

```

d. method dengan nama method/function **dashboard**. Boleh gunakan nama method yang berbeda. Method ini digunakan sebagai halaman ketika login berhasil dilakukan

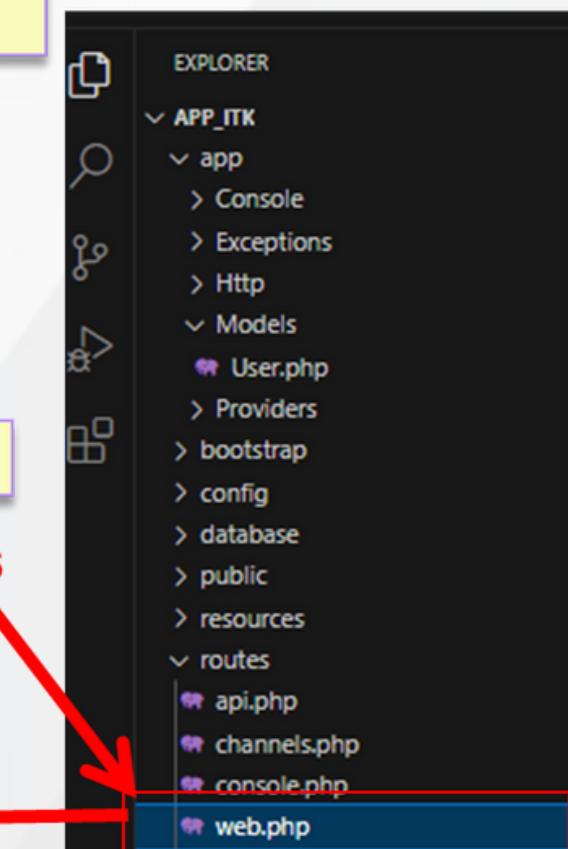
```

routes > web.php
1   <?php
2
3   use Illuminate\Support\Facades\Route;
4   use App\Http\Controllers\LoginGoogle;
5

```

7. Tambahkan kode seperti pada line 4 pada gambar dibawah dan **sesuaikan nama controller** yang kalian gunakan untuk didefinisikan dalam route .

6. Buka pada **app=>route->web**



6

7

Instal dan konfigurasi Socialite

```
Route::get('/', [LoginGoogle::class, 'form_login'])->name('form_login');

Route::controller(LoginGoogle::class)->group(function(){
    Route::get('auth.google', 'redirectGoogle')->name('auth.google');
    Route::get('logingoogle/callback', 'callback');
    Route::get('dashboard', 'dashboard');
    Route::get('logout', 'logout');
});
```

API APIs & Services

Client ID for Web application

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

The domains of the URIs you add below will be automatically added to your [OAuth consent screen](#) as [authorized domains](#).

Authorized JavaScript origins

For use with requests from a browser

Authorized redirect URIs

For use with requests from a web server

URIs 1 * <http://127.0.0.1:8000/logingoogle/callback>

Client secret	GOCSPEX-hktYv5KbVt9QkTUKgmVqjOexQBYQ
Creation date	September 3, 2023 at 11:59:50 PM GMT+7
Status	Enabled

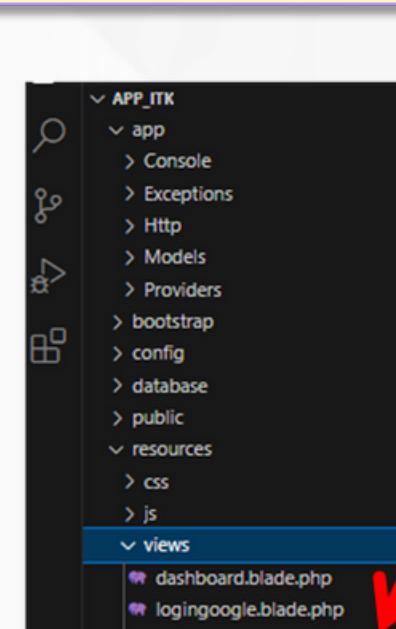
8. Masih pada file yang sama di **route=>web.php** ubahlah return pada line **18** dengan pada function di controller awal yang ingin kalian tampilkan (function pada controller yang akan diakses pertama kali ketika website/system dijalankan.).

9. Definisikan route tiap halaman pada controller logingoogle seperti pada gambar disamping .

- Route **auth.google** merupakan route yang digunakan untuk login mengakses API console google
- Route **logingoogle/callback** merupakan route yang digunakan untuk memberikan respon ketika mendapatkan respon dari API google. Pada route ini **harus sesuai** dengan konfigurasi google **redirect url** pada console google seperti pada contoh slide 18 saya berikan alamat url dengan alamat logingoogle/callback
- Route **Dashboard** merupakan url halaman admin ketika user berhasil login.. Nama route bisa diberikan sesuai keinginan
- Route **logout** digunakan untuk menghapus session yang disimpan

Membuat Tampilan Login

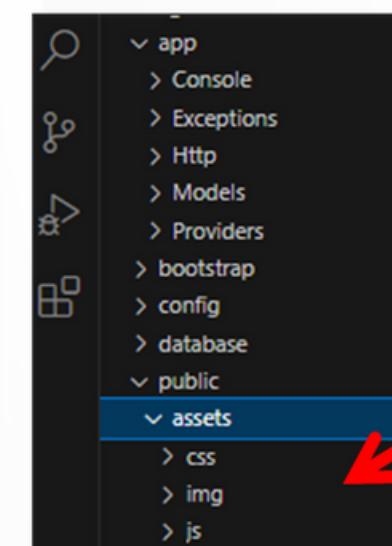
1. Buatlah tampilan sederhana untuk menampilkan tombol login . Boleh cari template login pada internet. Buka pada folder **app->resources=>views**. Buat new file dengan diberikan nama yang disesuaikan pada function controller yang digunakan untuk menampilkan form login.. Pada contoh dalam route (/) saya return pada pada function form_login dimana dalam function tsb ada deklarasi untuk menampilkan content yang saya berikan dengan variable \$form pada file **logingoogle.blade.php sebagai template form login dan sing up . (semua view harus memiliki extensi .blade.php)**



```
1
public function form_login(){
    $form="form_login";
    return view('logingoogle',compact('form'));
}
```

```
2
<div class="row">
    @include($form)
</div>
```

2. Buatlah function pada controller Logingoogle.php seperti kode disamping dengan diberikan **\$form="form_login"**



3. Berikan **@include(\$form)** pada bagian content informasi yang akan ditampilkan. Sehingga setiap function yang menggunakan template **logingoogle.blade.php** harus memiliki variable **\$form** untuk menampilkan content pada function tsb

4. Jika template login yang kalian peroleh terdapat file css dan js atau gambar maka file/folder dapat diletakan pada folder **app=>public =>folder tempat menyimpan css dan js** . Dalam contoh saya berikan dengan folder asset dan didalam folder asset terdapat folder css, dan js

Membuat Tampilan Login

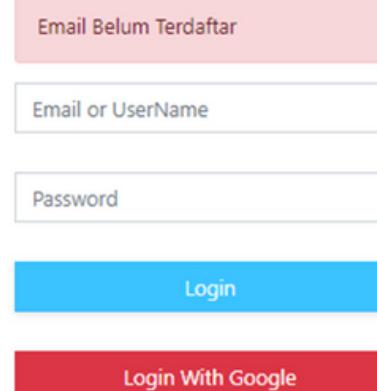
1. Karena pada function form_login di controller LoginGoogle.php , \$form mengarah pada file form_login maka buat satu file baru pada folder **resources=>views** dengan nama file **form_login.blade.php** dan buat desain form login dengan dengan diberikan button login with google
2. Pada file tersebut buatlah button atau sebuah link dan arahkan pada route login ke api console google yang telah dibuat . Lihat kembali pada slide 24. dalam contoh ini name route saya berikan nama auth.google sehingga dalam button diberikan perintah
`onclick="window.location='{{ route("auth.google") }}'"`
3. jika menggunakan atribut link `` maka bisa diberikan perintah
`login with google`

```
<div class="form-group">
    <input type="submit" class="btn btn-primary btn-block" value="Login">
    <input type="button" class="btn btn-danger btn-block" onclick="window.location='{{ route("auth.google") }}'" value="Login With Google">
</div>
```

```
@error('error')
<div class="alert alert-danger">{{ $message }}</div>
@enderror
```

3. Berikan pesan error dengan perintah `@error(namaerror` dalam **controller yang didefinisikan**) dan tutup dengan perintah `@enderror`

LOGIN



Email Belum Terdaftar

4. Contoh pesan kesalahan jika email tidak ditemukan dalam database

Membuat Tampilan Login

3. Buat file baru pada **app=>views** dengan **nama dashboard.blade.php**. Atau bisa disesuaikan nama yang diinginkan . Pada contoh saya gunakan file dashboard sebagai view untuk menampilkan informasi dihalaman admin.
Tambahkan potongan kode program pada layout template dashboar admin kalian. Didalam tag **@auth** maka content akan ditampilkan hanya jika login berhasil dilakukan . Jika tidak login maka content dalam **@auth** tidak akan ditampilkan . Penulisan tag **@auth** harus diberikan/ diakhiri dengan tag **@endauth**. Penulisan tag **@if** harus diakhiri dengan **@endif** . **Auth:user()->name** perintah yang digunakan untuk mengambil data nama user yang login

```
</li>
@auth
@if(Auth::user()->level=='1')
<li class="nav-item">
    <a class="nav-link disabled">Disabled/Super User</a>
</li>
@endif
<li class="nav-item">
    <a href="{{url('logout')}}" class="nav-link disabled">Logout</a>
</li>
@endauth
```

4. Didalam file **dashboard.blade.php** berikan perintah **@include(\$form)** pada bagian yang ingin untuk menampilkan content informasi. Pemberian nama **\$form** tinggal disesuaikan kebutuhan. Karena padatemplate ini saya berikan **\$form** maka setiap function yang menggunakan template **dashboard.blade.php** harus diberikan variable **\$form** yang diisikan nama file yang berisi data informasi yang akan ditampilkan dalam **\$form**

```
59
60 @include($form)
61
```

5. Karena didalam contoh function **dashboard** variable **\$form = "beranda_admin"**. Maka ini meunjukan kita harus memiliki file yang bernama **beranda_admin.blade.php** yang nantinya akan ditampilkan dalam **dashboard.blade.php** . Buat file baru bernama **beranda_admin.blade.php** dan Berikan informasi seperti gambar dibawah.

```
@auth
<h1>Hello, {{ Auth::user()->name }}</h1>
@endauth
```

Menjalankan Project

1. Kita buat dulu **data sample** pada table **users**. Buka browser ketikan **localhost/phpmyadmin**. Pilih database yang digunakan . Dalam contoh saya gunakan databse **itk**. Klik pada table **users**.
2. Pada bagian atas pilih **insert** kemudian isikan data **id, name, email dan level**. Berikan data email **sesuai** dengan email kalian masing-masing untuk pengujian system login with google.

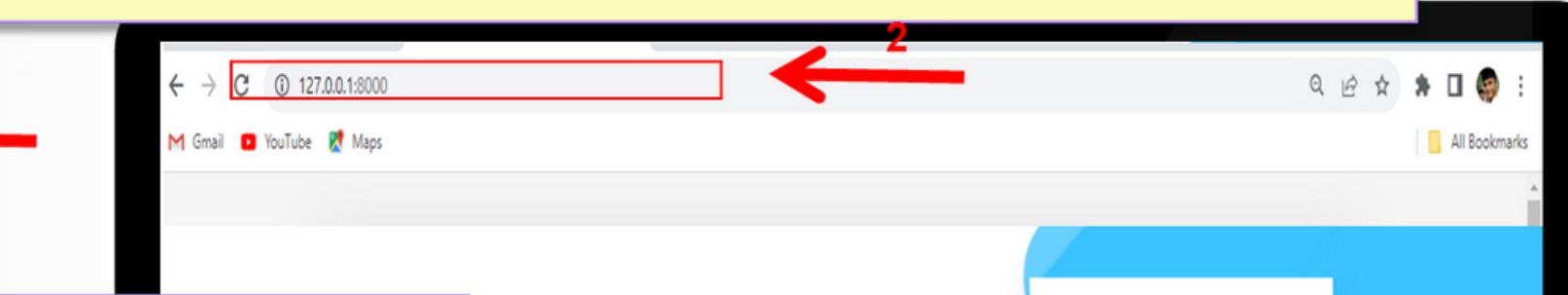
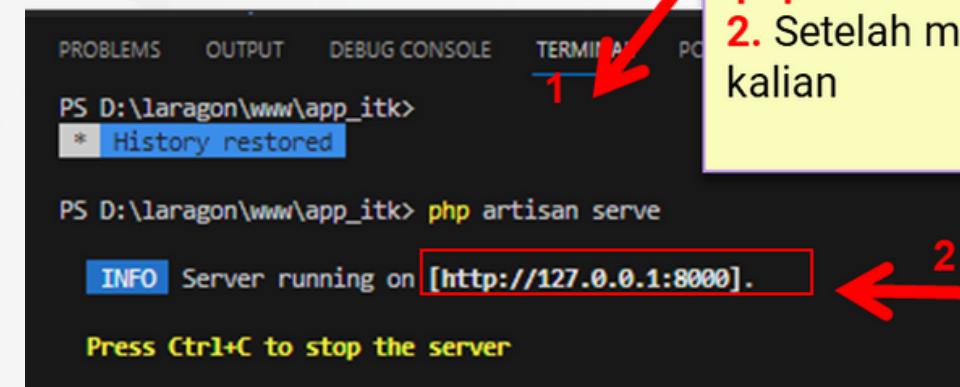
The screenshot shows the phpMyAdmin interface for the 'itk' database. The left sidebar lists databases and tables, with 'itk' selected and 'users' highlighted. The main area shows the 'users' table structure with columns: id, name, email, email_verified_at, password, remember_token, created_at, updated_at, google_id, and level. A new row is being inserted, indicated by the 'Insert' button at the top right of the table view. The 'email' column is filled with 'robiwariyanto@gmail.com'. Red arrows point to the 'itk' database selection, the 'users' table selection, the 'Insert' button, and the 'email' field in the data entry row.

	id	name	email	email_verified_at	password	remember_token	created_at	updated_at	google_id	level
<input type="checkbox"/>	3	robi	robiwariyanto@gmail.com	NULL	NULL	NULL	2023-09-12 15:15:17	2023-09-12 15:15:17		1

Menjalankan Project

1. Buka project pada visual code. Masuk pada bagian terminal . Ketikan perintah berikut :
php artisan serve

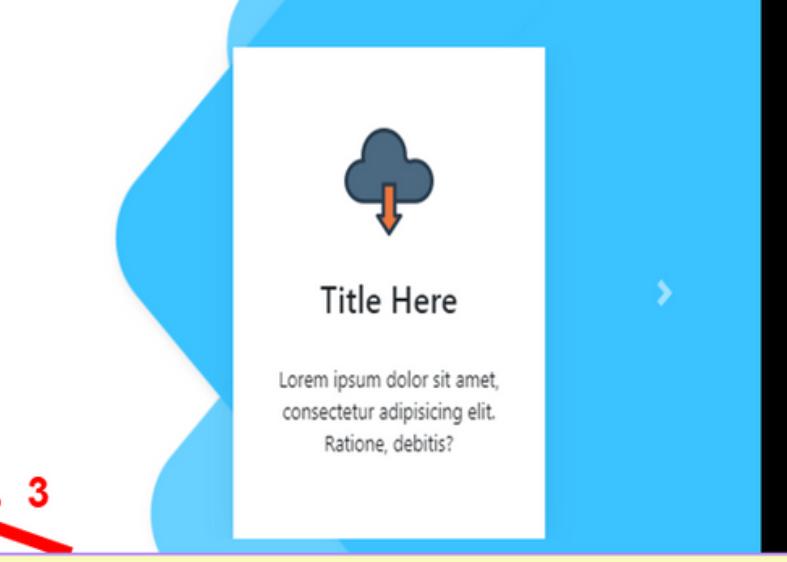
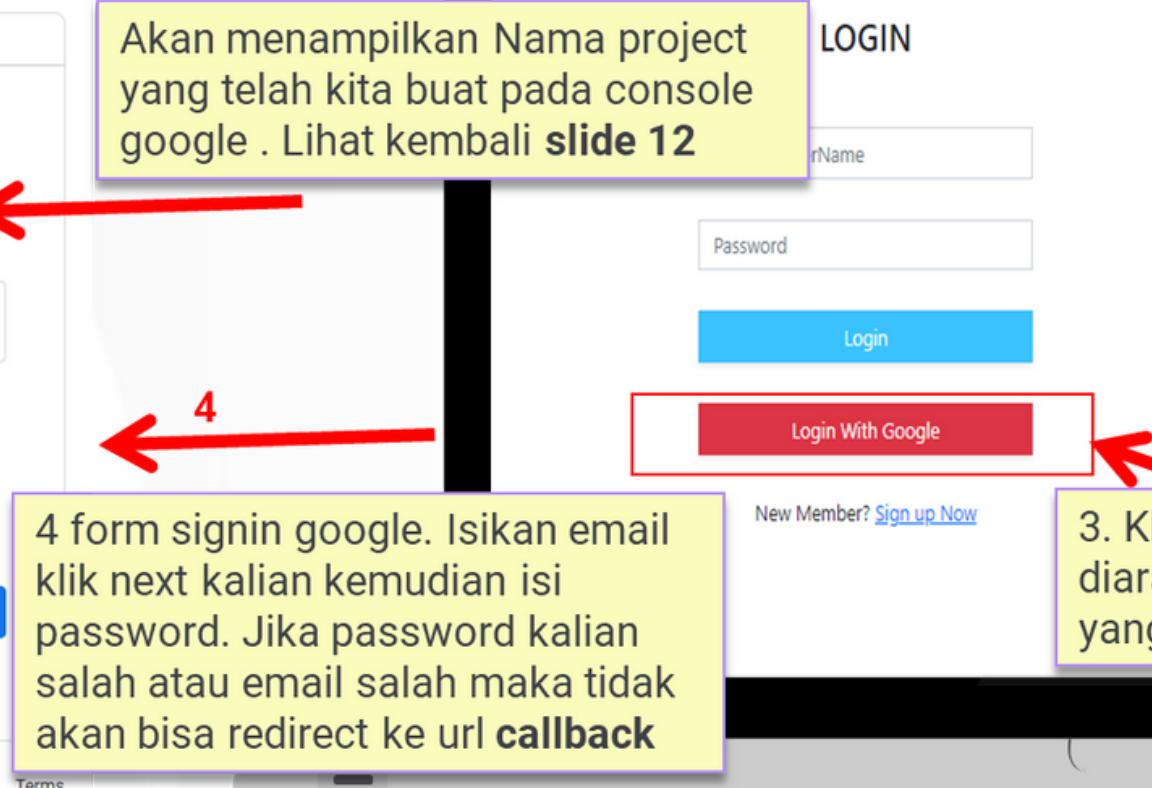
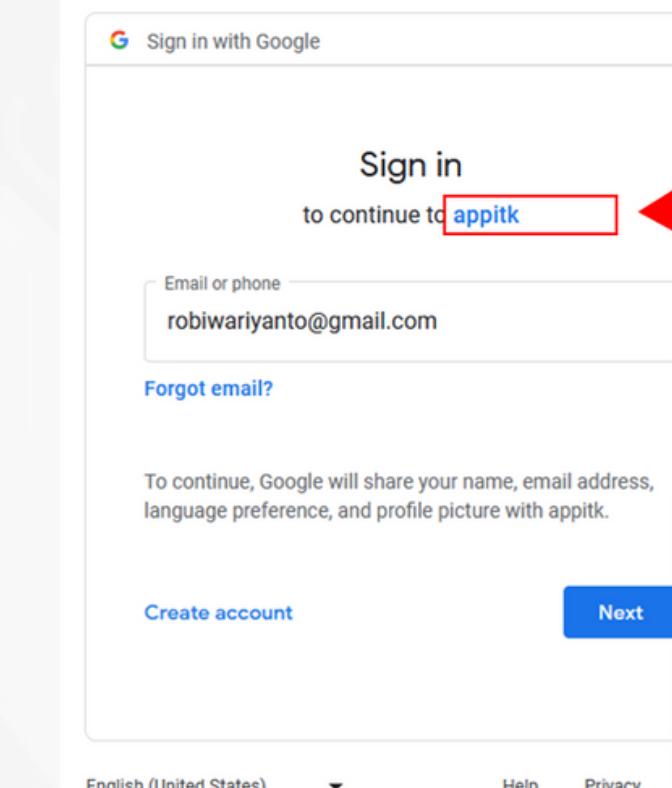
2. Setelah mendapatkan url copy url tersebut . Dan paste / ketikan pada webbrowser kalian



Akan menampilkan Nama project yang telah kita buat pada console google . Lihat kembali **slide 12**

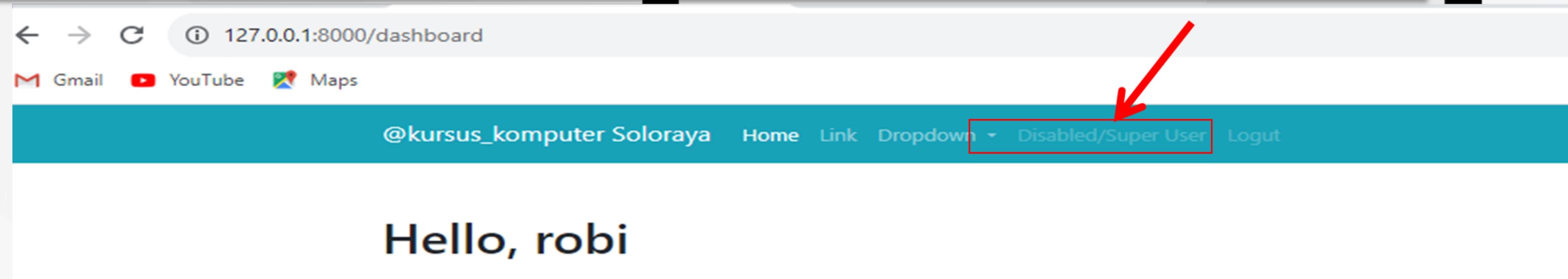
3. Klik pada button klik with google. Maka akan diarahkan form signin with google dari console google yang telah kalian buat sebelumnya

4 form signin google. Isikan email klik next kalian kemudian isi password. Jika password kalian salah atau email salah maka tidak akan bisa redirect ke url **callback**

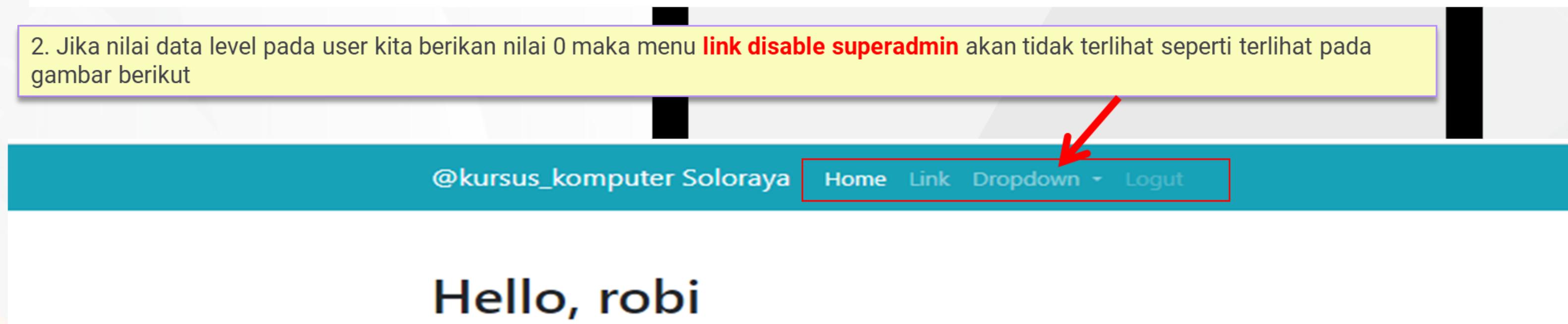


Menjalankan Project

1. Jika email berhasil login maka akan diredirect ke halaman dashboard yang telah kita buat sebelumnya. . Karena dalam contoh data sample pada slide 28 kita berikan data tersebut pada level 1(superadmin) , dan pada contoh latihan ini ketika level data memiliki nilai 1 maka link disable akan keluar



2. Jika nilai data level pada user kita berikan nilai 0 maka menu **link disable superadmin** akan tidak terlihat seperti terlihat pada gambar berikut





Login dengan Password

Login Dengan Password

1. Tambahkan 1 function pada controller **LoginGoogle** seperti berikut

1

```
public function login_action(Request $request)
{
    $validator = Validator::make($request->all(), [
        'email' => 'required',
        'password' => 'required',
    ], [
        'email.required' => 'Username Wajib Diisi.',
        'password.required' => 'Password Wajib Diisi.',
    ]);
    if ($validator->fails()) {
        return back()->withErrors($validator);
    }
    if (Auth::attempt(['email' => $request->email, 'password' => $request->password])) {
        $request->session()->regenerate();
        return redirect()->intended('/dashboard');
    }
    return back()->withErrors("User/Password Tidak tepat");
}
```

2. Masuk ke file **routes=>web.php**. Tambahkan route **login_action** seperti pada potongan kode berikut

```
Route::controller(LoginGoogle::class)->group(function(){
    Route::get('auth.google', 'redirectGoogle')->name('auth.google');
    Route::get('logingoogle/callback', 'callback');
    Route::get('dashboard', 'dashboard');
    Route::get('logout', 'logout');
    Route::post('login_action', 'login_action');
});
```

2

Login Dengan Password

3. Pada view form_login.blade.php ubahlah pada bagian `<form action` dan tambahkan kode `>{!! csrf_field() !!}` dibawah element `<form>`

```
3 <form class="login-form" action="{{url('login_action')}}" method="POST" >
    {{csrf_field()}}
```

4. Tambahkan satu button type submit pada diatas button login dengan goggle yang telah dibuat sebelumnya.

```
4 <input type="submit" class="btn btn-primary btn-block" value="Login">
<input type="button" class="btn btn-danger btn-block" onclick="window.location('{{ route("auth.google") }}'" value="Login With Google">
```

```
@if (Session::has('msg'))
    <div class="alert alert-success">
        <ul>
            <li>{{ Session::get('msg') }}</li>
        </ul>
    </div>
@endif 5
```

5. Tambahkan juga untuk kode seperti diatas untuk menampilkan jika data berhasil diubah. Sesuaikan nama variable nya pada function dicontroler

6. Update data sample pada field password pada table user yang digunakan untuk menguji login melalui form dengan memberikan inputan password . Buatlah function untuk mengubah data sample yang telah dibuat sebelumnya seperti kode disamping kanan. Sesuaikan email dan password sesuai data yang kalian buat .

```
public function generate_password(){
    $email="robiwariyanto@gmail.com";
    $finduser = User::where('email', $email)->first();
    $finduser->password = Hash::make('robi');
    $finduser->save();
    return redirect('/')->with('msg', 'Password Berhasil diupdate');
}
```

7. Tambahkan route generate_password pada group controller LoginGoogle melalui file `routes=>web.php`

```
Route::controller(LoginGoogle::class)->group(function(){
    Route::get('auth.google', 'redirectGoogle')->name('auth.google');
    Route::get('loggingoogle/callback', 'callback');
    Route::get('dashboard', 'dashboard');
    Route::get('logout', 'logout');
    Route::post('login_action', 'login_action');
    Route::get('generate_password', 'generate_password');
});
```

Login Dengan Password

7. Pastikan server project masih berjalan. Buka kembali pada web browser kemudian akses pada url http://127.0.0.1:8000/generate_password. Jika password berhasil diupdate maka akan menampilkan pesan **Password berhasil Diupdate**

LOGIN

- Password Berhasil diupdate

Email or UserName

Password

Login

Login With Google

7. Menguji login dengan password. Akses kembali pada <http://127.0.0.1:8000> isikan email dan password kemudian tekan tombol login.

Pesan error ketika email atau password yang diinputkan user tidak tepat

LOGIN

- User/Password Tidak tepat

Pesan error ketika email dan password tidak diberikan nilai

LOGIN

- Username Wajib Diisi.
- Passsword Wajib Diisi.

Ketika user dan password yang diinputkan terdapat ditemukan maka akan redirect ke halaman **dashboard** dan akan menampilkan data nama user yang login

@kursus_komputer Soloraya Home Link Dropdown - Disabled/Super User Logout

Hello, robi

Input Data User dengan AJAX

1. Buka file view **form_login.blade.php** berikan kode seperti gambar dibawah

```
<script type="text/javascript" language="javascript">
    var url="{{url('register')}}";
    var csrf="{{csrf_token()}}";
</script>

<div class="form-group">
    <div class="text-center">New Member? <a href="javascript:void(0)" onclick='add_data(url,csrf)'>Sign up Now</a></div>
</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"
integrity="sha384-geWF76RCwLtnZ8qwlowPQNgul3RmwHVBC9FhGd1KrxdiJJigb/j/68SIy3Te48kz" crossorigin="anonymous"></script>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.4/jquery.min.js"></script>
<script src="{{asset('/assets/js/app/add_register.js')}}"></script>
```

```
function add_data($urladd,token){
    var method= 'get';
    $.ajax({
        headers: {'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')},
        url: $urladd,
        type: 'get',
        data: {
            "_token": token,
            "_method": method,
            // "key": id
        },
        success: function (data) {
            $('#adddata').modal('toggle');
            $('#form').html(data);
            $('#judul_form').html("Register");
        }
    });
}
```

2.. Buatlah file baru dengan nama **add_register.js** atau bisa kalian sesuaikan dengan nama lain. Simpan file tersebut pada folder **public/assets/js**. Sesuaikan path yang kalian buat sebelumnya

3. pada file add_register.js buat function add_data(seperti kode disamping. Function ini digunakan untuk menampilkan form add user ke dalam modal form . **\$form** digunakan untuk menampilkan form yang ingin ditampilkan.

Input Data User dengan AJAX

4. Pada route web berikan route seperti gambar dibawah

```
Route::get('register', [LoginGoogle::class, 'form_register'])->name('form_register');
```

5. Buat function form_register pada controller LoginGoogle seperti kode disamping

```
public function form_register(){  
    return view('form_register');  
}
```

6. Buat file baru bernama **form_register.blade.php** dan berikan id **form=formregister** atau bias disesuaikan dengan nama lain

```
2  <!-- <div class="modal-body" -->  
3  <form action="{{url('api/register_action')}}" method="POST" id="formregister">  
4  <div class="modal-body">
```

7. Load **add_register.js** pada **form_register.blade.id** seperti pada kode gambar disamping

```
<script src="{{asset('/assets/js/app/add_register.js')}}"></script>
```

```
<div class="modal fade bd-example-modal-lg" id="adddata" role="dialog" style="display:none" aria-labelledby="exampleModalLabel"  
aria-hidden="true">  
    <div class="modal-dialog modal-lg">  
        <div class="modal-content">  
            <div class="modal-header btn-info">  
                <h5 class="modal-title" id="exampleModalLabel"><span id="judul_form"></span></h5>  
                <button type="button" class="close" data-bs-dismiss="modal" aria-label="Close">  
                    <span aria-hidden="true">&times;</span>  
                </button>  
            </div>  
            <div id="form"></div>  
        </div>  
    </div>  
</div>
```

8. id="form" bisa disesuaikan pada script function add_data() pada slide 35

Input Data User dengan AJAX

8. Buat function **register_action** pada controller LoginGoogle.php seperti kode gambar dibawah

```
public function register_action(Request $request)
{
    $validator = Validator::make($request->all(), [
        'email' => 'required|unique:users,email,$request->email',
        'password' => 'required',
        'level' => 'required',
        'name' => 'required',
    ], [
        'email.required' => 'email Wajib Diisi.',
        'email.unique' => 'email Sudah Digunakan.',
        'password.required' => 'Password Wajib Diisi.',
        'level.required' => 'Level Wajib Diisi.',
        'name.required' => 'Nama Wajib Diisi.',
    ]);

    if ($validator->fails()) {
        return response()->json(['status' => false, 'errors' => $validator->errors()]);
    }
    $user=new User;
    $user->email=$request->email;
    $user->name=$request->name;
    $user->password=Hash::make($request->password);
    $user->level=$request->level;

    $user->save();
    return response()->json(['status' => true]);
}
```

Pada file view **logingoogle.blade.php** (letak file pada **resources=>views**) tambahkan kode untuk load plugin sweetalert untuk menampilkan pesan kesalahan/ pesan sukses saat proses input data seperti berikut

```
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11.7.28/dist/sweetalert2.all.min.js"></script>
<link href="https://cdn.jsdelivr.net/npm/sweetalert2@11.7.28/dist/sweetalert2.min.css" rel="stylesheet">
<script type='text/javascript' src='https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js'></script>
<script type='text/javascript' src='https://stackpath.bootstrapcdn.com/bootstrap/5.0.0-alpha1/js/bootstrap.min.js'></script>
```

Input Data User dengan AJAX

9. Pada **add_registrasi.js** tambahkan kode untuk menyimpan data secara ajax. Sesuaikan nama id form pada slide 36 pada view yang digunakan untuk menyimpan data .

```
$(document).ready(function () {
    $("#formregister").submit(function(e) {
        e.preventDefault();
        e.stopImmediatePropagation();

        var formData = new FormData(this);

        $.ajax({
            headers: {'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')},
            url: $(this).attr('action'),
            type: 'POST',
            data: formData,
        });
    });
});
```

10. Tambahkan routes pada **routes=>web.php** seperti kode dibawah untuk memanggil form register.

```
Route::get('register', [LoginGoogle::class, 'form_register'])->name('form_register');
```

10. Tambahkan routes pada **routes=>api.php** seperti kode dibawah untuk menyimpan data register.

```
Route::post('register_action', [LoginGoogle::class, 'register_action'])->name('register_action');

1 <?php
2
3 use Illuminate\Http\Request;
4 use Illuminate\Support\Facades\Route;
5 use App\Http\Controllers\LoginGoogle;
6 use App\Http\Controllers\UserController;
```

Pada api.php kode paling atas jangan lupa untuk memanggil controller yang akan digunakan seperti kode disamping untuk load controller UserController.php dan LoginGoogle.php

Input Data User dengan AJAX

11. Menguji Form register. Pastikan web server project masih berjalan. Akses kembali url pada browser <http://127.0.0.1:8000/> kemudian klik pada signup . Sehingga akan menampilkan form register seperti gambar disamping.

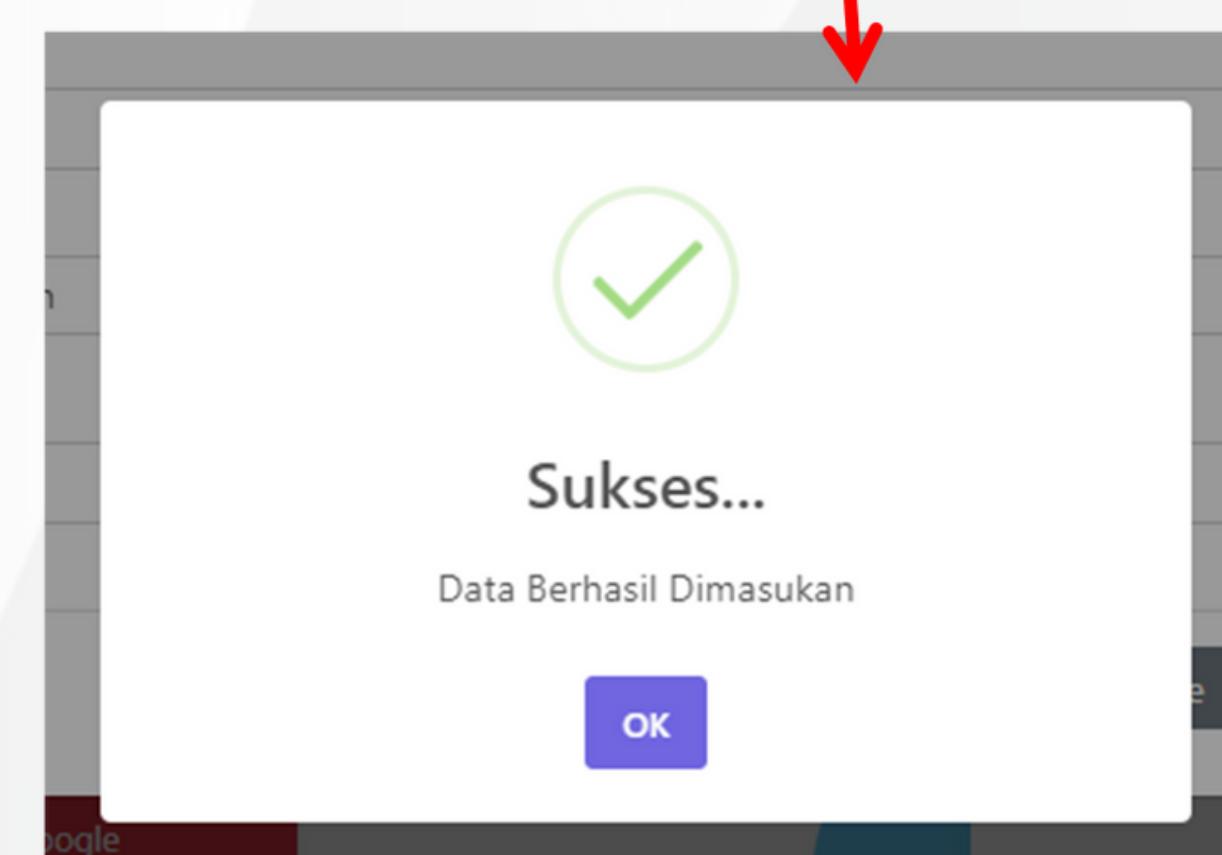
The image shows a comparison between a login interface and a registration form. On the left, a 'LOGIN' screen has fields for 'Email or UserName' and 'Password', and buttons for 'Login' and 'Login With Google'. Below these is a link 'New Member? [Sign up Now](#)'. A red arrow points from this link to the right side. On the right, a 'Register' modal window titled 'FORM REGISTER' contains fields for 'Email or UserName', 'Nama', 'Pilih salah satu Level Admin', and 'Password', along with 'Close' and 'Register' buttons.

Input Data User dengan AJAX

12. Dalam studi case kali ini semua field wajib diisi . Ketika salah satu ada yang terlewat tidak diisi maka akan menampilkan pesan kesalahan pada form yan tidak diisi seperti pada gambar dibawah

The screenshot shows a registration form titled 'Register'. The form has four input fields: Email (robiwa@gmail.com), Password (123456), Role (Super Admin), and Confirm Password. Below the 'Role' field, there is an error message: 'Password Wajib Diisi.' A red arrow points to the close button in the top right corner of the modal window.

13. Jika data berhasil dimasukan maka akan menampilkan pesan alert seperti berikut.





Pertemuan II

Memasang Datatable untuk menampilkan data user

1. Membuat Controller dengan nama controller yaitu **UserController**. buka project kalian pada editor visual studio code . Kemudian masuk pada bagian terminal dan ketikan perintah berikut :
php artisan make:controller UserController

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS D:\laragon\www\app_itk> php artisan make:controller UserController  
INFO Controller [D:\laragon\www\app_itk\app\Http\Controllers\UserController.php] created successfully.  
PS D:\laragon\www\app_itk>
```

```
<?php  
  
namespace App\Http\Controllers;  
use App\Models\User;  
use Laravel\Socialite\Facades\Socialite;  
use Illuminate\Support\Facades\Auth;  
use Illuminate\Support\Facades\Hash;  
use Illuminate\Support\Facades\Validator;  
use Illuminate\Http\Request;
```

2. Tambahkan bagian diatas class UserController seperti kode disamping

```
public function data_user(){  
    $form="data";  
    return view('dashboard',compact('form'));  
}
```

3. Buat function data_user() untuk menampilkan data user

```
Route::get('data_user', [UserController::class, 'data_user'])->name('data_user');
```

4. Tambahkan route data_user pada routes=>web.php

Memasang Datatable untuk menampilkan data user

5. Buat function data_user() pada Usercontroller. Seperti berikut

```
99  public function data_user(){
100    $form="data";
101    return view('dashboard',compact('form'));
102 }
```

6. Buat file baru pada resources =>views . Dikrenakan pada function data_user \$form saya berikan nilai \$form="data" maka view yang akan ditampilkan yaitu harus memiliki file yang bernama **data.blade.php** bisa kalian sesuaikan sesuai keinginan. Berikan kode seperti berikut . Pada table berikan id="user" yang digunakan untuk menampilkan data user

```
<div class="container my-5">
<table class="display" id="user" style="width:100%">

  <thead>
    <tr>
      <th>Nama</th>
      <th>email</th>
      <th>level</th>
      <th>Aksi</th>
    </tr>
  </thead>
</table>
</div>
```

7. Pada file data.blade.php load plugin datatable seperti kode berikut

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.4/jquery.min.js"></script>
<link rel="stylesheet" type="text/css" href="https://cdn.datatables.net/1.13.6/css/jquery.dataTables.min.css"/>
<script type="text/javascript" src="https://cdn.datatables.net/v/dt/dt-1.10.12/datatables.min.js"></script>
```

Memasang Datatable untuk menampilkan data user

8. Pada controller UserController.php buat function untuk mengambil data dari database yang akan ditampilkan di file data.php. Kode selengkapnya akan saya berikan secara langsung.

```
public function dataajax(Request $request)
{
    $columns = array(
        0 =>'email',
        1 =>'name',
        2=>'level',
    );
    $totalData = User::count();
    $totalFiltered = $totalData;
    $limit = $request->input('length');
    $start = $request->input('start');
    $order = $columns[$request->input('order.0.column')];
    $dir = $request->input('order.0.dir');

    if(empty($request->input('search.value')))
    {
        $posts = User::offset($start)
                    ->limit($limit)
                    ->orderBy($order,$dir)
                    ->get();
    }
    else {
        $search = $request->input('search.value');

        $posts = User::where('email','LIKE',"%{$search}%")
                    ->orWhere('name', 'LIKE', "%{$search}%")
                    ->offset($start)
                    ->limit($limit)
                    ->orderBy($order,$dir)
                    ->get();
    }
}
```

Memasang Datatable untuk menampilkan data user

9. Buat route pada **routes/api.php** untuk menjalankan/memanggil function dataajax pada controller UserController .

```
Route::post('dataajax', [UserController::class, 'dataajax'])->name('dataajax');
```

```
1 <?php
2
3 use Illuminate\Http\Request;
4 use Illuminate\Support\Facades\Route;
5 use App\Http\Controllers\LoginGoogle;
6 use App\Http\Controllers\UserController;
```

Patikan pada routes=>api.php sudah diload controller yang akan digunakan

```
4 $(document).ready(function () {
5     $('#user').DataTable({
6         processing: true,
7         serverSide: true,
8         ajax: {
9             url: '_url_data_user',
10            dataType: "json",
11            type: "POST",
12            data:{ _token: token}
13        },
14        columns: [
15            { "data": "email" },
16            { "data": "nama" },
17            { "data": "level" },
18            { "data": "options" }
19        ],
20        columnDefs: [
21            { targets: [3], orderable: false,
22            },
23            { targets: [1], orderable: false,
24            }
25        ],
26    });
27 });
28 });
29 );
```

10. Buat file baru dengan nama **data.js** simpan file tersebut pada **public/assets/js/app/** atau sesuaikan dengan path kalian .

```
<script>
var url_data_user="{{url('api/dataajax')}}";
var token="{{ csrf_token() }}";
</script>
<script src="{{asset('/assets/js/app/data.js')}}"></script>
<script>
function reload_ajax(id) {
    var table = $('#'+id).DataTable();
    table.ajax.reload();
}
</script>
```

11. Pada file **data.blade.php** tambahkan kode seperti kode disamping..**url data_user** untuk mendefinisikan url route yang akan digunakan pada **data.js**

Memasang Datatable untuk menampilkan data user

12.Pada file **dashboard.blade.php** tambahkan satu menu untuk mengakses data user seperti berikut:

```
<li class="nav-item">
| <a class="nav-link" href="{{url('data_user')}}">data user</a>
</li>
```

Pastikan server project masih berjalan dan akses url berikut pada web browser
http://127.0.0.1:8000/data_user

Data User				
Nama	email	level	Aksi	
ddd@gkk.cc	ii	1	<input checked="" type="checkbox"/> 	
ppp@gmail.com	ppp	1	<input checked="" type="checkbox"/> 	
robiwa12@gmail.com	123456	1	<input checked="" type="checkbox"/> 	
robiwa19@gmail.com	robiwa	1	<input checked="" type="checkbox"/> 	
robiwa86@gmail.com	robiwa12	1	<input checked="" type="checkbox"/> 	
robiwa@gmail.com	robiwa	1	<input checked="" type="checkbox"/> 	
robiwariyanto@gmail.com	robi	0	<input checked="" type="checkbox"/> 	
Showing 1 to 7 of 7 entries				
Previous 1 Next				

Menghapus data dengan AJAX

1. Pada potongan kode function dataajax di controller UserController terdapat button delete dimana pada potongan kode delete terdapat function hapus (\$url,\$token) yang mengarah pada route **api/delete_user/idpost**

```
$urledit=json_encode(url("/api/detail_data/{$post->id}"));
$token=json_encode(csrf_token());
$url=json_encode(url("/api/delete_user/{$post->id}"));

$nestedData['email']      = $post->email;
$nestedData['nama']       = $post->name;
$nestedData['level']      = $post->level;
$nestedData['options']   = "&nbsp;<a href='javascript:void(0)' title='EDIT' onclick='return update_data($urledit,$token)'><span class='glyphicon glyphicon-edit'></span></a>
&nbsp;<a href='javascript:void(0)' title='DELETE' class='btn btn-danger' onclick='return hapus($url,$token)'><span class='glyphicon glyphicon-trash'></span></a>";
$data[] = $nestedData;
```

```
function hapus($urlhapus,token){
    var method= 'delete';
    Swal.fire({
        title: "Anda yakin?",
        text: "Data akan dihapus.",
        type: "question",
        icon:'warning',
        showCancelButton: true,
        confirmButtonColor: "#3085d6",
        cancelButtonColor: "#d33",
        confirmButtonText: "Hapus!"
    }).then(result => {
        if (result.value) {
            $.ajax({
                headers: {'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')},
                url: $urlhapus,
                type: delete ,
                data: {
                    "_token": token,
                    "_method": method,
                    // "id":$id
                },
                success: function (data) {
                    Swal.fire({
                        icon: 'success',
                        text: 'Data Berhasil Dimasukan',
                        title: data.status ? "Berhasil" : "Gagal",
                        text: data.status
                            ? "Data berhasil dihapus"
                            : "Data gagal dihapus",
                        type: data ? "success" : "error"
                    });
                    reload_ajax('user');
                }
            });
        }
    });
}
```

2. Buat /tambahkan function hapus pada **data.js** seperti pada kode disamping

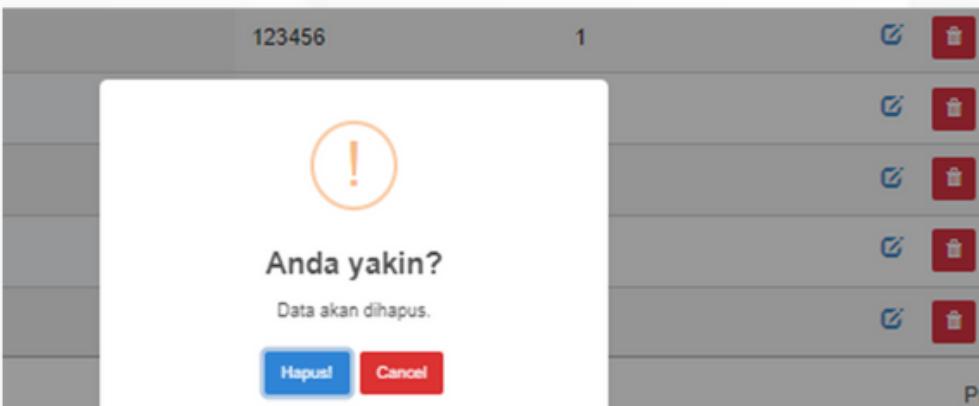
Menghapus data dengan AJAX

```
88 public function delete_data($key){  
89     $detail_data= User::where('id', $key)->get()->first();  
90     $user_id=$detail_data->id;  
91     $hasil= User::find($user_id)->delete();  
92     $data['status']=false;  
93     if($hasil){  
94         $data['status']=true;  
95     }  
96     return response()->json($data);  
97 }  
98 }
```

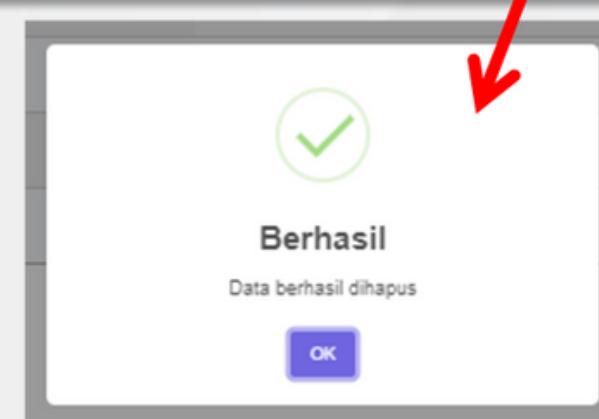
3. Buat function `delete_data($key)` pada controller **UserController** dimana `$key` merupakan id data yang akan dihapus

4. Buat route delete user dengan 1 parameter sebagai id data yang akan dihapus dimana route ini akan digunakan untuk mengakses function `delete_data` pada controller **UserController**

```
Route::delete('delete_user/{id}', [UserController::class, 'delete_data']);
```



5. Pastikan server project masih berjalan. Dan akses url http://127.0.0.1:8000/data_user dan pilih salah satu data dan klik pada button delete yang akan dihapus hingga mengeluarkan konfirmasi hapus data seperti gambar disamping . Jika diklik **cancel** maka data tidak jadi kehapus jika diklik **hapus** maka data akan kehapus dan akan menampilkan pesan data berhasil dihapus



LATIHAN /TUGAS MEMBUAT UPDATE DATA

Terapkan update data jika diklik button edit maka akan mengeluarkan popup form update data sesuai dengan data yang dipilih . Adapun output yang dihasilkan akan menghasilkan seperti pada slide 10

```
$urledit=json_encode(url("/api/detail_data/{$post->id}"));
$token=json_encode(csrf_token());
$url=json_encode(url("/api/update_user/{$post->id}"));

$nestedData['email']      = $post->email;
$nestedData['nama']       = $post->name;
$nestedData['level']      = $post->level;
$nestedData['options'] = "&nbsp;<a href='javascript:void(0)' title='EDIT' onclick='return update_data($urledit,$token)'><span class='glyphicon glyphicon-edit'></span></a>
&nbsp;<a href='javascript:void(0)' title='DELETE' class='btn btn-danger' onclick='return hapus($url,$token)'><span class='glyphicon glyphicon-trash'></span></a>";
$data[] = $nestedData;
```

```
77
78     function update_data($urladd,token){
79         var method= 'get';
80         $.ajax({
81             headers: {'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')},
82             url: $urladd,
83             type: 'get',
84             data: {
85                 '_token': token,
86                 '_method': method,
87                 // "key":id
88             },
89             success: function (data) {
90
91                 $('#adddata').modal('toggle');
92                 $('#form').html(data);
93                 $('#judul_form').html("Update Data");
94
95             }
96         });
97     });
98
99
100 }
```

Berikut contoh function update_data() untuk menampilkan form edit data

Berikut merupakan route yang dibentuk pada **routes=>api.php** untuk menampilkan data yang akan diupdate

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

Berikut merupakan route yang dibentuk pada **routes=>api.php** untuk menyimpan data yang akan diupdate

```
Route::post('update_register_action', [UserController::class, 'update_register_action'])->name('update_register_action');
```



LATIHAN UPDATE DATA

Output yang dihasilkan ketika button edit diklik.

The screenshot shows a user management interface. On the left, there is a table with columns 'Nama' and 'Email'. The table lists several users with their names and email addresses. A modal window titled 'Update Data' is open over the table, centered on the screen. The modal has a title 'UPDATE USER' and contains fields for 'Email' (robiwa19@gmail.com), 'Nama' (robiwa), and 'Role' (Super Admin). Below these fields is a 'Password' input field. At the bottom of the modal are two buttons: 'Close' and 'Update'. To the right of the modal, there is a vertical list of user entries with checkboxes and delete icons. A red arrow points from the 'Update' button in the modal to a success message box at the bottom. Another red arrow points from the word 'Aksi' (Actions) in the header of the list to the same success message box. The success message box itself contains a green checkmark icon, the word 'Sukses...', and the text 'Data Berhasil Dimasukan' (Data successfully inserted). It also has an 'OK' button at the bottom.

Dan akan menampilkan output pesan seperti gambar disamping ketika button update diklik

THANK YOU!!