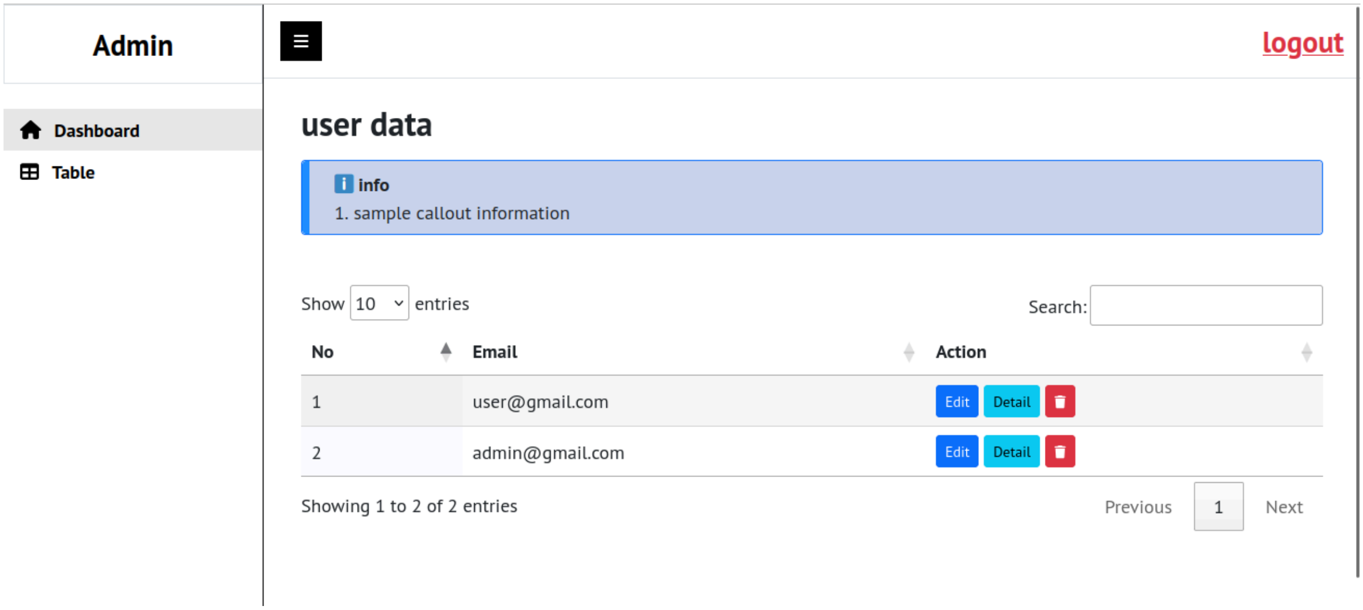
This is a template for building application with codeigniter 4.

feature

1. Simple admin page
2. Login page
3. Authentication and authorization (with user schema sql)
4. Logging (monolog)
5. bootstrap 5
6. Uuid generator (Ramsey UUID)
7. Collection (Ramsey)
8. Mappify (my own lib for mapper data)
9. Jwt cookie (firebase jwt)
10. Datatables
11. Simple file service for moving file locally

Example view

Sample simple dashboard



user data

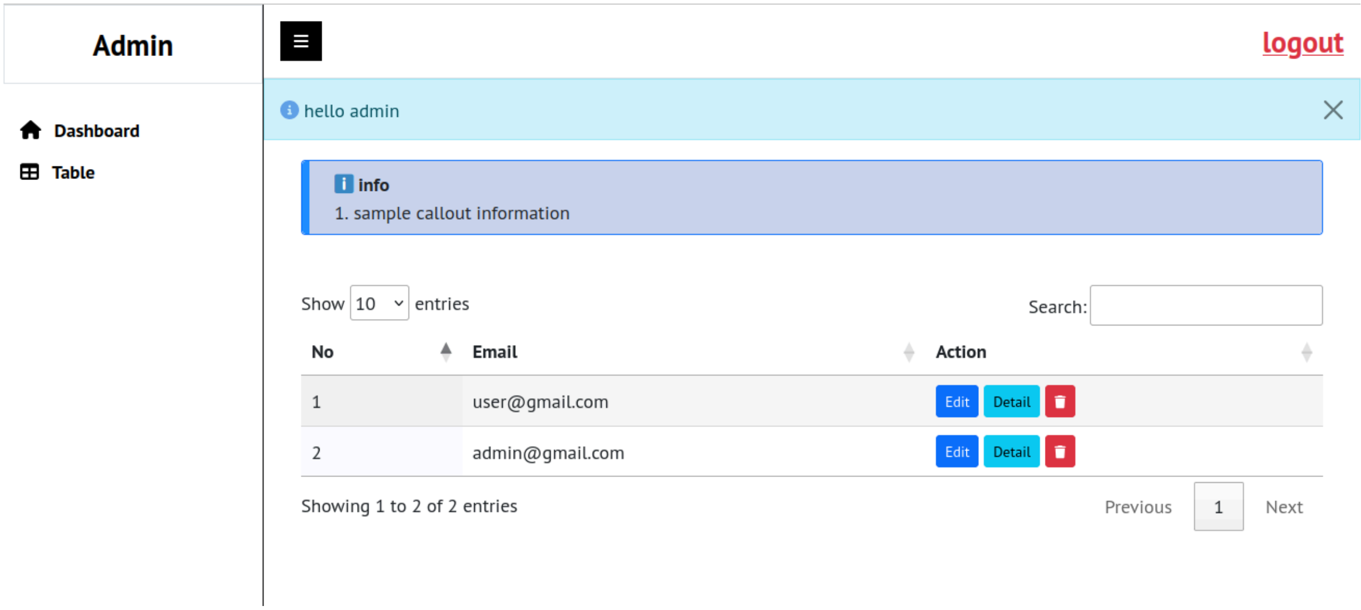
info

1. sample callout information

No	Email	Action
1	user@gmail.com	Edit Detail
2	admin@gmail.com	Edit Detail

Showing 1 to 2 of 2 entries

with message information



hello admin

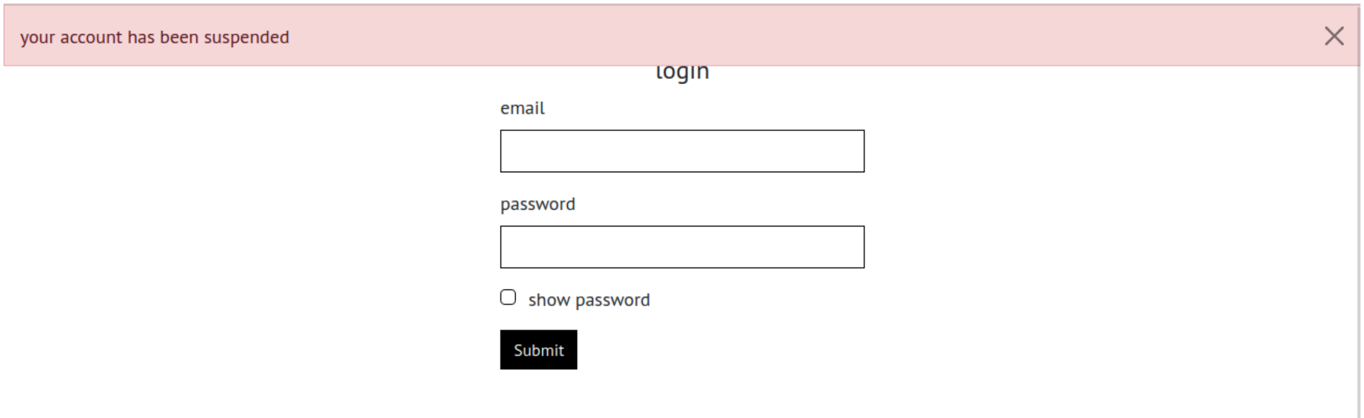
info
1. sample callout information

Show 10 entries Search:

No	Email	Action
1	user@gmail.com	Edit Detail
2	admin@gmail.com	Edit Detail

Showing 1 to 2 of 2 entries

sample login page


your account has been suspended

login

email

password

□ show password

Creating instance (dependency injection ci4)

In codeigniter 4 any method to create an object instance with `app\Config\Service.php`.

```
class Services extends BaseService
{
        // creating userDetailService
    public static function userDetailService($getShared = true)
    {
        if ($getShared) {
            return static::getSharedInstance('userDetailService');
        }

        return new UserDetailServiceImpl(new UserModel());
    }
    }
```

how to used

```
$userService = Services::userDetailService();
```

Routing

im separate the folder for routes, so if you see at `app\Config\RoutingApp` theres any class such as `auth` and `admin`. And for render we include the files at `Routes.php`.

app\Config\Routes.php

```
const ROUTES_PATH = APPPATH . "Config/RoutingApp/";


if(file_exists(ROUTES_PATH."auth.php")){
    require_once ROUTES_PATH."auth.php";
}
```

Creating routes with filter

In this template have filter/middleware mechanism for authenticate and authorize the user can access resource.

sample to use authorization filter

```
$routes->group("/admin",["filter"=>"authorization:ADMIN_CREATE"],function($routes) {

    $routes->group("", ["namespace" => "App\Controllers\Admin"],function($routes) {
        $routes->get("","Home::index");

    });
});
```

So only user have `ADMIN_CREATE` privilege can access the page. You can passing 2 or more with commas `,` like this `["filter"=>"authorization:ADMIN_CREATE,ADMIN_UPADATE"]`

Filter

You can find filter at `app\Filter\AuthFilter.php` the mechanism is validate user request from authentication and authorization.

```
 // validate cookie
            $result = $this->jwtCookie->validateJwtToken($cookie);

            // when the routes have any argument for privileges access
            if($arguments!=null){

                $this->authentication->hasAuthority($arguments,$result->privileges);
            }


```

Filter will validate jwt cookie and if theres any argument `["filter"=>"authorization:ADMIN_CREATE,ADMIN_UPADATE"]` ('ADMIN_CREATE' is argument) will validate privilege the user `$this->authentication->hasAuthority($arguments,$result->privileges`.

Exception

Im already build custom exception `app/Exception` theres two type of exception such as checked and unchecked, the checked exception will extends `Exception` and this class must be handled and the next is unchecked exception, this case for unexpectly the error like failed inserting data or other wheres unhandled.

Checked exception

```
Datanotfoundexception
class DataNotFoundException extends Exception
{
    public function __construct(string $message,... $args)
    {
        $messageFormat = sprintf($message,...$args);
        parent::__construct($messageFormat);
    }
}

example used
throw new
```

```
DataNotFoundException(ExceptionMessageHelper::dataNotFound(class_basename($d
ataClass)));
```

unchecked exception

```
class FailedInsertDataException extends \RuntimeException
{
    private Logger $appLogger;
    public function __construct(string $message,
                                \Throwable $previousException=null,
                                ?array $data=null)
    {
        $this->appLogger  = AppLogger::getMonologConfig();

        parent::__construct($message);

        $c = LoggerContextHelper::contextualMessage
        (
            $previousException->getMessage(),
            $previousException->getTrace(),
            $data,
        );

        $this->appLogger->error($message,$c);

    }
}

Example used
catch (\ReflectionException $e) {
            throw new  FailedInsertDataException(
              ExceptionMessageHelper::failedInsertException(Client::class),
              $e,
              ["email"=>$client->getEmail()]
            );
```

The unchecked exception will generate a logging data and codeigniter 4 mechanism will return to 500 (server error) pages.
You can use `app\Helper\ExceptionMessageHelper` for instantly create string message.

Logger context

This class can found at `app\Helper\LoggerContext.php` help to create logging message, especially for exception.

```
'previous-message'=>$errorMessage,
               'detail-data'=>$data,
               'trace'=>self::simpleTrace($trace)
```

the message created at logging
`simpleTrace` will give wheres the error come from, which line and function from last trace list.

```
 public static function simpleTrace($traceArray)
    {
        $file = $traceArray[0]["file"];
        $line = $traceArray[0]["line"];
        $function = $traceArray[0]["function"];

        $data =
            [
                "file-error"=>$file,
                "line"=>$line,
                "function"=>$function
            ];

        return $data;
    }
}
```

JwtCookie

This class can found at `app\thridparty\JwtCookie.php` for creting and validate jwt cookie. And for encryption key im put at `.env`.

Logging

This class can found at `app\ThirdParty\MonologConfig`, this configuration for output monolog such as file and console. The file output will writed at `Project_name\WRITEABLE\logs`.

format

```
$output     = "%level_name% | %datetime% > %message% | %context% %extra%\n";
        $dateTime   = new DateTime('now', new DateTimeZone('Asia/Jakarta'));
        $currentTime = $dateTime->format('Y-m-d ~ H:i');
```

To use this you initialize with `app\Helper\AppLogger.php`

```
public static function getMonologConfig()
    {
        $logger = new MonologConfig();

        return $logger->config();
    }
```

set true at `$logger->config();` for file output.
example using

```
private JwtCookie $jwtCookie;

    public function __construct()
    {
        $this->appLogger    = AppLogger::getMonologConfig();

    }
```

Result model helper

When searching data the output can be empty so use this class for checking the result and give the exception `DataNotFound`.

example

```
public function getDataById($id)
    {
        $result = $this
            ->where('users.id',$id)
            ->first();
        ResultModelHelper::isResultEmpty($result,Users::class);

        return $result;
    }
```

Authentication and Authorization

This file can found at `app\Service\UserService`, theres 2 core for authentication and authotrization such as `AuthenticationImpl` and `UserDetailService` it just like spring security, user details for getting `UserDetails` object and `Authentication` for authenticate.

authenticationImpl

```
public function authenticate(string $username,string $password): UserDetails
    {
        // get user details
        $user = $this->userDetailServiceImpl->getUserDetails($username);

        // check when user account was suspend
        if($user->isSuspend()){
            throw new BadCredentialException('your account has been suspended');
        }

        // validate password
        if(password_verify($password,$user->getPassword()))
        {
            $this->appLogger->info('user successfully login', ['username'=>$user->getEmail()]);

            return $user;

        }else{
            throw new BadCredentialException('username or password are incorrect');
        }
    }
```

You can change the code for authenticate user, like your app use email and password just change the parameter and user details query search. At `AuthenticationImpl` theres 2 method is `authenticate` (authenticate user login) and `hasAuthority` (authenticate user privilege).

Message view

You can give a message to user for error, information, or validation. This template at `app\views\layout\_message`.
example

```
 return redirect()
                ->to(base_url('login'))
                ->with("error", $e->getMessage());
        return view ("admin",["message"=>"hello world"]);
```

You can use `message`, `error` and `validation`, to use `validation` message just pass the message to exception `ValidationException` to get array message from `validation` codeigniter `$validation->getErrors();` .

Mappify

This is my own library for mapper data from array to object with `psalm`, if your IDE doesnt support it cant used. To use this you can get instance from `app\Helper\RawToObjectFactory`
example

```
public static function getUser(array $raw)
{
    $mappify  = RawToObjectFactory::getInstance();

    $user = $mappify->getObject(
        $raw,
        Users::class,
        [
            "id"=>"user_id"
        ]);

    $role = $mappify->getObject(
        $raw,
        Role::class,
        ["id"=>"role_id","name"=>"role_name"]
    );

    $user->setRole($role);
    return $user;
}
```

reference (here)[https://github.com/Noplyen/RawToObjectMapper-PHP]