Lumen making ApiResponser.php for succesResponse() , errorResponse() and setting Exceptions from Handler.php

Step 1 : First e app er under e Traits→ApiResponser.php name e ekta php file banate hobe, structure → [app->Traits->ApiResponser.php]

Step 2: ApiResponser.php er coding structure,

```php
<?php

namespace App\Traits;

use Illuminate\Http\Response;

trait ApiResponser
{
   public function successResponse($data, $code = Response::HTTP_OK)
   {
       return \response()->json([
           'data' => $data,
           'status' => $code
       ],$code);
   }

   public function errorResponse($message, $code)
   {
       return \response()->json([
           'error' => $message,
           'code' => $code
       ],$code);
   }
}
```

Step 3 : Controller er moddhe jevabe ApiResponser.php ke use korte hobe,
[A code snippet of a controller with ApiResponser.php]

Note 1 :

```php
<?php

namespace App\Http\Controllers;

use App\Author;
use Illuminate\Http\Request;
use App\Traits\ApiResponser;
use Illuminate\Http\Response;

class AuthorController extends Controller
{
   use ApiResponser;

   /**
    * Display a listing of the resource.
    *
    * @return \Illuminate\Http\Response
    */
   public function index()
   {
       $authors = Author::all();
       return $this->successResponse($authors);

   }
}
```

Step 4: Error codes and messages jevabe Exceptions->Handler.php te jevabe set kore,

Firstly, import korte hobe,

```
use Exception;
use Illuminate\Validation\ValidationException;
use Illuminate\Auth\Access\AuthorizationException;
use Illuminate\Database\Eloquent\ModelNotFoundException;
use Illuminate\Http\Response;
use Laravel\Lumen\Exceptions\Handler as ExceptionHandler;
use Symfony\Component\HttpKernel\Exception\HttpException;
use App\Traits\ApiResponser;
use Illuminate\Auth\AuthenticationException;
```

Secondly, **ApiResponser.php** ke use korte hobe,

```
use ApiResponser;
```

Thirdly, **public function render** er moddhe code jevabe likhte hobe, [function er moddher code muche replace kore dileo hobe],

```
   public function render($request, Exception $exception)
   {
       if($exception instanceof HttpException){
           $code = $exception->getStatusCode();
           $message = Response::$statusTexts[$code];
           return $this->errorResponse($message,$code);
       }

       if($exception instanceof ModelNotFoundException){
           $model = strtolower(class_basename($exception->getModel()));
           return $this->errorResponse("Does not exist any instance of
{$model} with the given id", Response::HTTP_NOT_FOUND);
       }

       if($exception instanceof AuthorizationException){
           return $this->errorResponse($exception->getMessage(),
Response::HTTP_FORBIDDEN);
```

```
        }

        if($exception instanceof AuthenticationException){
            return $this->errorResponse($exception->getMessage(),
Response::HTTP_UNAUTHORIZED);
        }

        if($exception instanceof ValidationException){
            $errors = $exception->validator->errors()->getMessages();

$this->errorResponse($errors,Response::HTTP_UNPROCESSABLE_ENTITY);
        }

        if(env('APP_DEBUG',false)){
            return parent::render($request, $exception);
        }

        $this->errorResponse('Unexpected error. Try
later',Response::HTTP_INTERNAL_SERVER_ERROR);

    }
```

Full **Handler.php** er code,

```
<?php

namespace App\Exceptions;

use Exception;
use Illuminate\Validation\ValidationException;
use Illuminate\Auth\Access\AuthorizationException;
use Illuminate\Database\Eloquent\ModelNotFoundException;
use Illuminate\Http\Response;
use Laravel\Lumen\Exceptions\Handler as ExceptionHandler;
use Symfony\Component\HttpKernel\Exception\HttpException;
use App\Traits\ApiResponser;
use Illuminate\Auth\AuthenticationException;

class Handler extends ExceptionHandler
```

```
{

    use ApiResponser;


    /**
     * A list of the exception types that should not be reported.
     *
     * @var array
     */
    protected $dontReport = [
        AuthorizationException::class,
        HttpException::class,
        ModelNotFoundException::class,
        ValidationException::class,
    ];

    /**
     * Report or log an exception.
     *
     * This is a great spot to send exceptions to Sentry, Bugsnag, etc.
     *
     * @param  \Exception  $exception
     * @return void
     */
    public function report(Exception $exception)
    {
        parent::report($exception);
    }

    /**
     * Render an exception into an HTTP response.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Exception  $exception
     * @return \Illuminate\Http\Response|\Illuminate\Http\JsonResponse
     */


    public function render($request, Exception $exception)
```

```
    {
        if($exception instanceof HttpException){
            $code = $exception->getStatusCode();
            $message = Response::$statusTexts[$code];
            return $this->errorResponse($message,$code);
        }

        if($exception instanceof ModelNotFoundException){
            $model = strtolower(class_basename($exception->getModel()));
            return $this->errorResponse("Does not exist any instance of
{$model} with the given id", Response::HTTP_NOT_FOUND);
        }

        if($exception instanceof AuthorizationException){
            return $this->errorResponse($exception->getMessage(),
Response::HTTP_FORBIDDEN);
        }

        if($exception instanceof AuthenticationException){
            return $this->errorResponse($exception->getMessage(),
Response::HTTP_UNAUTHORIZED);
        }

        if($exception instanceof ValidationException){
            $errors = $exception->validator->errors()->getMessages();

$this->errorResponse($errors,Response::HTTP_UNPROCESSABLE_ENTITY);
        }

        if(env('APP_DEBUG',false)){
            return parent::render($request, $exception);
        }

        $this->errorResponse('Unexpected error. Try
later',Response::HTTP_INTERNAL_SERVER_ERROR);

    }
}
```