API Project to get JSON data from server

In this project I've designed an API whit php no framework. All requests are get (there is abilities to increase verbs in future). and response is a sample JSON array;

Architecture

I've used *MVC* atchtecture so there is controller models(while there is no databade to get data from but there is basic models to develope in future) and views.
This project needs to mode_rewrite other vice url needs to change
URL Examle mode_rewrite enable

http://utl.com/shout/steve-jobes?limit=10

URL Example no mode_rewrite

http://utl.com/index.php?req=shout/steve-jobs&limit=10

MVC is a trust able, safe and easy to continue developing architecture so I used it even without a popular PHP *MVC* framework.

Project description

Caching scenario

for beginning I have used a simple method to safe outputs into file. If there is a request younger than 10S it will returns directly without load other parts of site
In bigger project we should have use small databases , sub tables or small SQLite database.
Time is a simple time here and can easily change from helper/cacheing.php in line 11. you can change it easily. But in future this time should be add in a new config file to manage all progect config from a single file. I did not have more configuration so used it as is.
We know that 10 seconds is very short time for caching but because of test random output (read Define request note) we have set it to 10 to show us changes in results.
Caches files also delete automatically after cache time to save space;

Routes

caches requested controller ,name and other query string parameters from URL;
Using this class make it easier to add more controller and methods and routes in this project

Core/api_prg.php
This object loads by index.php and moderate project. Calls controller class and methods using Routes.php and loads view to show result;
in development if some other routes added or routing scenario changes this file will be more usefull. It's even possible to define regex path easily, if we define such a rout separated class.

parent_controller.php and parent_model.php
This tow files are my parent classes all popular and useful methods of controllers and models should add to this classes. So child models and controllers can use them. The parent_controller also initials requested parameters from client. Also parse it to demand any hack ingestion.
The parent_model connects to database (or load data file here) so models do not needs to connect

directly. Also in future it can contains a ORM to CRUD database.

Models/Shout_model.php
This model gets all data from parent model and pars it for requested names, Gets all sentences and returns random sentences. So clients may receive different result.
Note: In case of active caching, clients should stay for cache tile end.
Note: Cache scenario can change to save most requested people all sentences so models can get data faster.

There is only one view file now to echo result as JSON. Result is with JSON header so it's easier for client's to works with.

Testing

I have used phpunit test framework and it's added tho this project. Scenario is in vendor/bin/test/ShoutTest.php and you can test project with calling vendor/phpunit/phpunit/phpunit vendor/bin/test/ShoutTest.php from the project root(in Linux);

I did search and found that phpunit is one of the most families and powerful php test framework. It can install, add to project and moderate easily by composer. So I chose it as my test framework;

How to use API

This API normally needs mode_rewrite enabled in server and activate for this URL in virtualhost and .htaccess file also should be in the root of API project. So URL will rewrite as a beautiful face.

But even without mode_rewrite you can use this API just with query string. Below you can see usage with and without mode_rewrite enabled in server.

In case of mod_rewrite enabled on your server call

url.tls/Shout/name?limit=N

in case of no mode rewrite you should call

url.tls/index.php?Shout/name&limit=N

Url from beginning to question mark is case insensitive. But query string is case sensitive.

Define request

Name is the name of human that you want to receive his/her sentences;

N is an integer value and count of requested sentences.

If name not exists API will returns [“Noting found”]

in N <1 OR N > 10 will returns [“Limit should be between 1 to 10”]

if limit missed return is ["Missed Limit parameter"]

if shout or name missed return is ["missed method"]

Note: As we found in your sample output, the result order was not as same as data in json file, And also it's normal to expect that client not always wants the first sentence if they request limit=1; So we choose a random method for response data.

Project Folders view

index.php This is the first file that runs via apache of any other web service. This file loads main project.

phpunit.xml phpunit configuration. Phpunit test uses this xml as configuration. It is a simple XML file and can change by any advace developer.

caches contains cached results
controllers controller to fetch requests for client. Developers can add new contrntrollers to it. Controllers namespace is controllers;

core All core files should ad din this folder by namespace as core;
All files into this folder is autoload too. Application classes can extends from this classes;

core/api_prg.php main project object who calls controllers via routes. This class loads in index.php This class has a *run* method who run application.

core/parent_controller.php all controllers should extends from this controller. So with add or replace methods in this class you can add behaviors of child methods. In future this method can contains user validation and access controll so I create this class to modrate all new controller that may add to this application.

core/parent_model.php Models uses to load data from database. We do not have database here but I create this to moderate data access from models. So this model can contains or load ORM for database communications and chid models can use its usfull method without creating any new socat to database.

helpers Helper classe are global usable class and can load from all other parts of API. Developers can add their class here so other developers can use their class in all other parts of project.

helpers/caching.php This is a simple class to cache responses. At first checks is any previews same requests exists or not. Cache time is defined to 10 seconds now but it can be change easily from this file in line 11;
This class has 2 method that calls at the beginning ot loading Application and afret return before exit. So it can save any new requests in to files in folder caches.

helpers/Routes.php This class defines and load controller and run requested method by parsing query string. It can redesign, develop and … to get better result in a big project, And why I used it because in my imagination this progect will neet active routing in future.

models/Shoutes_model.php A simple model to get data from data.json file.In future it can load data or CRUD in database with parent_model.

vendor/autoload.php composer autoload. This file loads all important classes at application start. This file is generated by composer and developer shoul not change it.

vendor/bin/phpunit PHPunit main file

vendor/bin/test/ShoutTest.php A phpunit test scenario for current classes and methodes. After any new class or method a new test method should add to this file.

Sincerely yours