Laravel Framework for Web Applications

SESSION 03

WRITING A SIMPLE APPLICATION

OBJECTIVES

- Learn to create simple Web application using Laravel
- Explain how to use routes to serve different Web pages
- Understand how views can enhance presentation of an application
- Use the Blade Templating Engine
- Learn to Add Master Template and Nested Views

PLANNING THE APPLICATION

Configure and run on the domain name `simpleApp.local`.

↓

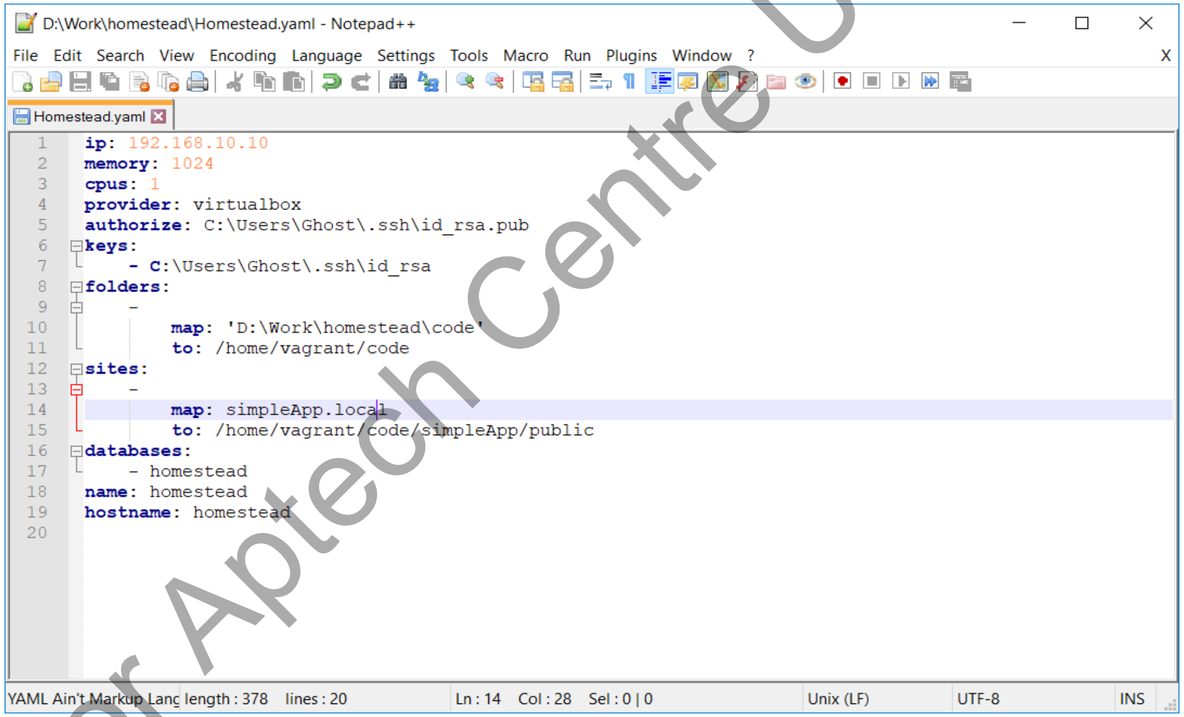
Serve a static home page (`http://simpleApp.local/`)

↓

Serve a dynamic test page (`http://simpleApp.local/test`) - that will fetch input from the MVC URI structure and reflect the value

CONFIGURING THE ENVIRONMENT (1-2)

Configuring homestead:


D:\Work\homestead\Homestead.yaml - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Homestead.yaml

```
ip: 192.168.10.10
memory: 1024
cpus: 1
provider: virtualbox
authorize: C:\Users\Ghost\.ssh\id_rsa.pub
keys:
    - C:\Users\Ghost\.ssh\id_rsa
folders:
    -
        map: 'D:\Work\homestead\code'
        to: /home/vagrant/code
sites:
    -
        map: simpleApp.local
        to: /home/vagrant/code/simpleApp/public
databases:
    - homestead
name: homestead
hostname: homestead
```

YAML Ain't Markup Lang length : 378 lines : 20 Ln : 14 Col : 28 Sel : 0 | 0 Unix (LF) UTF-8 INS

CONFIGURING THE ENVIRONMENT (2-2)

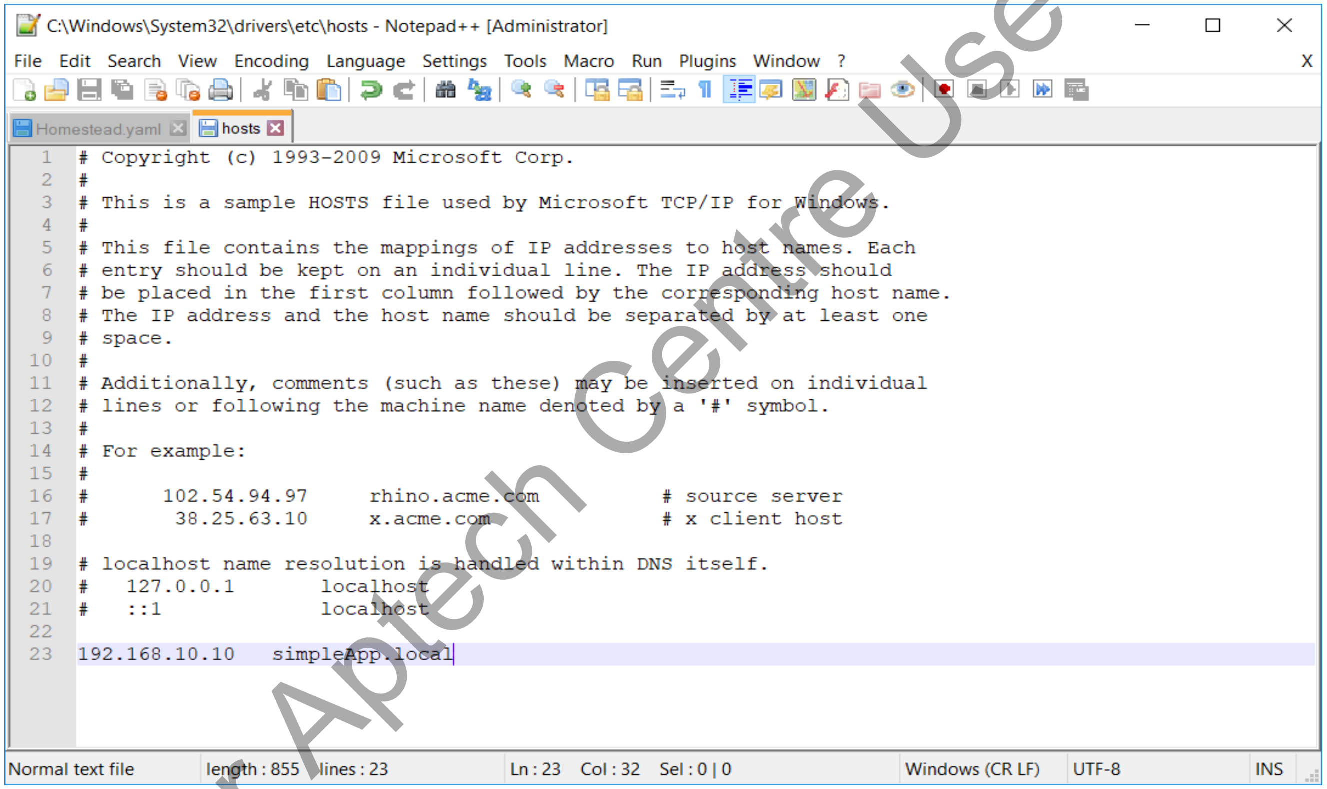
Map the simpleApp.local domain:

```
sites:
    -
        map: simpleApp.local
        to: /home/vagrant/code/simpleApp/public
```

Command:

```
vagrant reload --provision
```

CONFIGURING THE HOSTS FILE

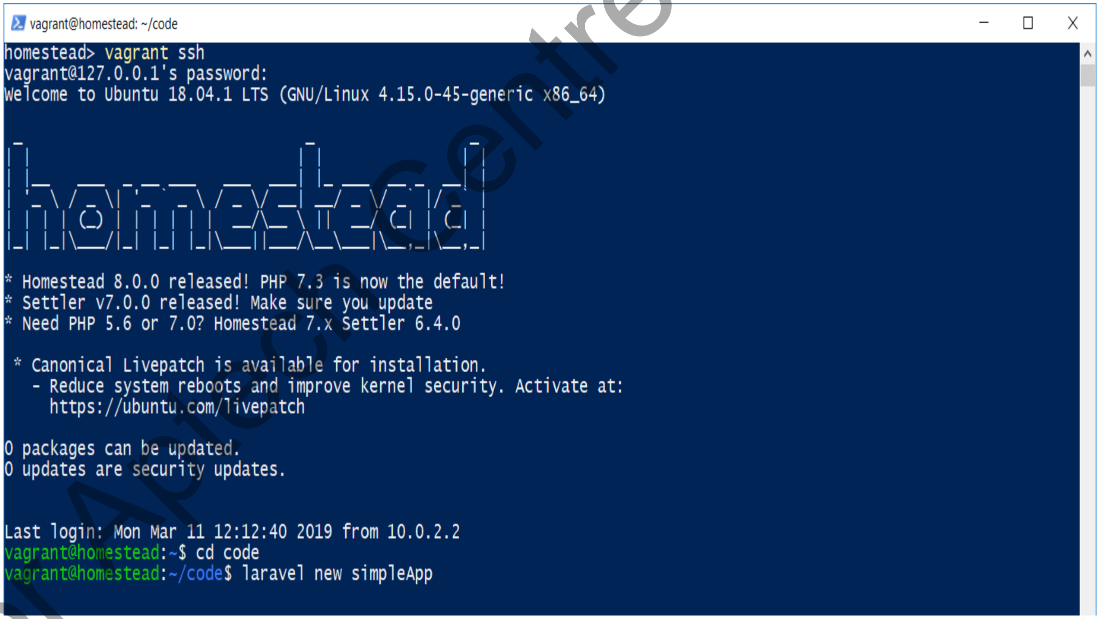

C:\Windows\System32\drivers\etc\hosts - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Homestead.yaml
hosts

```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#    127.0.0.1       localhost
#    ::1             localhost

192.168.10.10   simpleApp.local
```

CREATING LARAVEL APPLICATION (1-3)

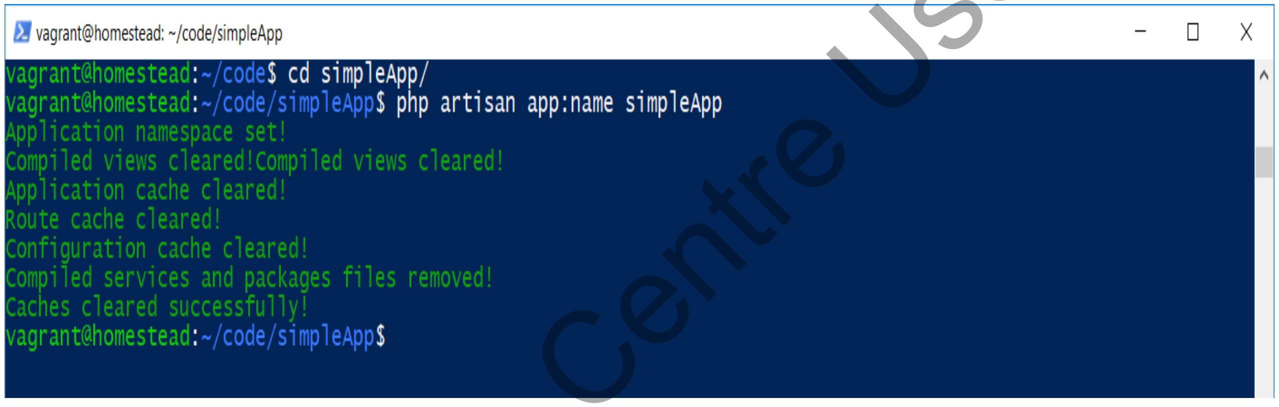
Syntax to create a new Laravel project:

```
laravel new [name]
```



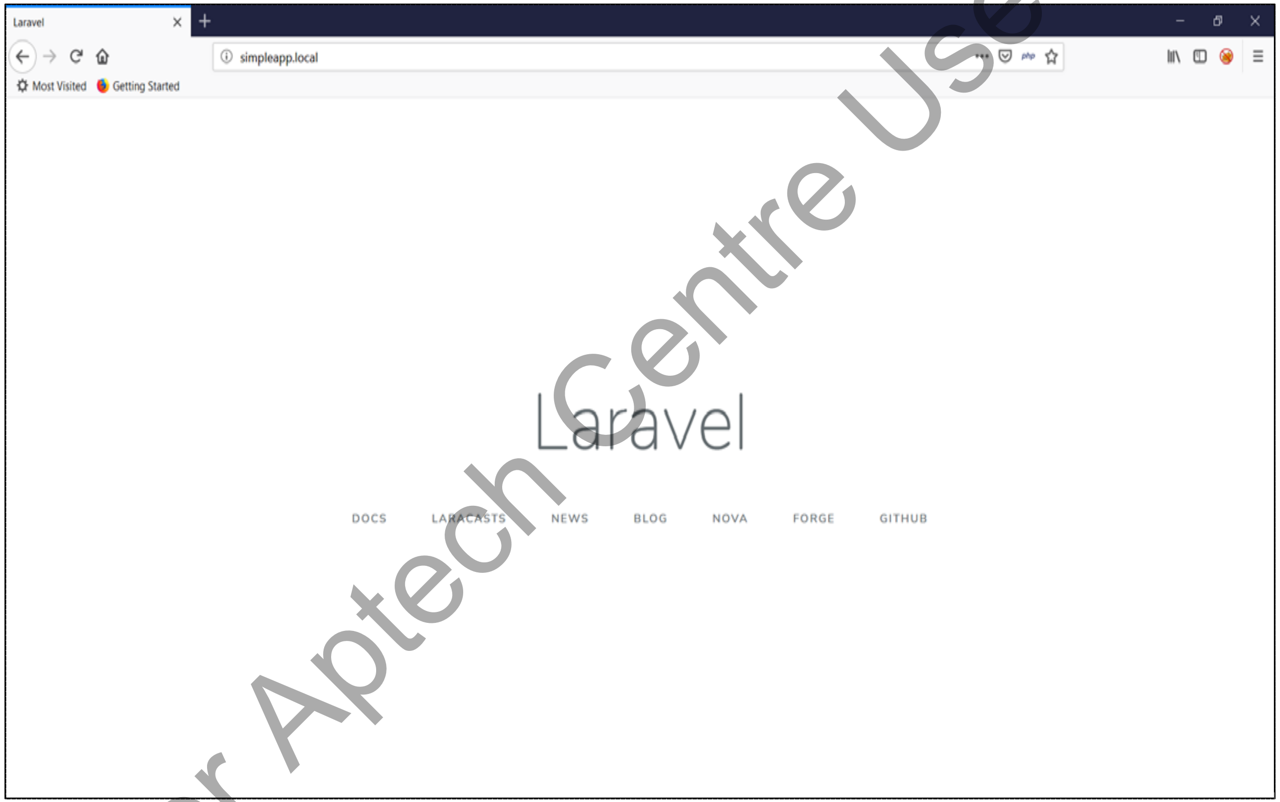
```
homestead> vagrant ssh
vagrant@127.0.0.1's password:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-45-generic x86_64)

* Homestead 8.0.0 released! PHP 7.3 is now the default!
* Settler v7.0.0 released! Make sure you update
* Need PHP 5.6 or 7.0? Homestead 7.x Settler 6.4.0

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.


Last login: Mon Mar 11 12:12:40 2019 from 10.0.2.2
vagrant@homestead:~$ cd code
vagrant@homestead:~/code$ laravel new simpleApp
```

CREATING LARAVEL APPLICATION (2-3)

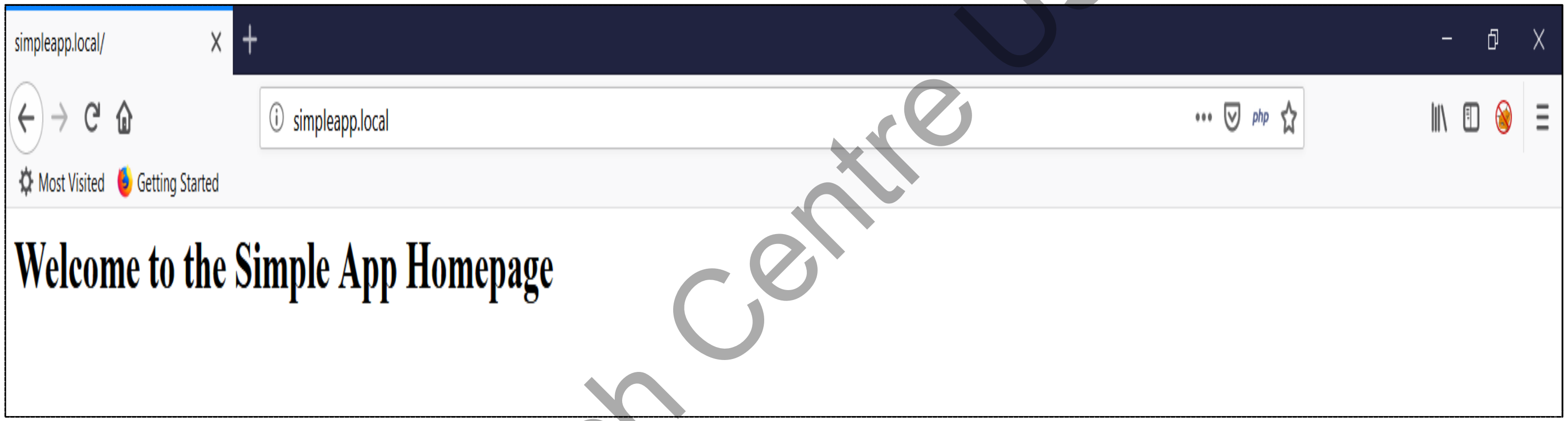


```
vagrant@homestead:~/code$ cd simpleApp/
vagrant@homestead:~/code/simpleApp$ php artisan app:name simpleApp
Application namespace set!
Compiled views cleared!Compiled views cleared!
Application cache cleared!
Route cache cleared!
Configuration cache cleared!
Compiled services and packages files removed!
Caches cleared successfully!
vagrant@homestead:~/code/simpleApp$
```

CREATING LARAVEL APPLICATION (3-3)



WRITING A ROUTE



Welcome to the Simple App Homepage

CREATING A DYNAMIC TEST PAGE

Code Snippet 1:

```
Route::get('test', function () {
    $content = "To access a specific test, visit
http://simpleApp.local/test#id, replace #id with the test
id. For example http://simpleApp.local/test1"
    return $content;
})
```

Code Snippet 2:

```
Route::get('test{id}', function ($id) {
    return "You are trying to access Test #$id";
});
```

USING VIEWS (1-2)

Views

- Provides the modularity required for Web application development.
- Located at `resources\views`
- Separates the controller/application logic from the presentation logic.

USING VIEWS (2-2)

Code Snippet 1:

```
<html>
    <body>
        <h1>Hello. Welcome to your own Simple Web
Application</h1>
    </body>
</html>
```

Code Snippet 2:

```
Route::get('/',function () {
    return view('home');
});
```

USING BLADE TEMPLATING ENGINE

- Lightweight template language
- Provides multiple short codes
- Blade template engine reduces the number of keystrokes
- Increases the readability of templates
- Supports most PHP constructs to create loops and conditions
- Allows templates to be nested and extended
- No master layout in Blade

SUMMARY

- By default, all resources required to run a Web application are installed in the Homestead virtual environment.
- Laravel leverages the MVC design pattern to provide an interactive interface to the user.
- The hosts file in Windows is the first file that is referenced when the operating system tries to resolve a domain name.
- Routes serve different Web pages and Views enhance their presentation.
- Blade is a lightweight template language that provides multiple short codes.
- Master Template and Nested Views can be configured in the master.blade.php file.
- Laravel follows the ‘Convention is better than Configuration’ method.