**ASSIGNMENT FRONT SHEET**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Qualification** | **BTEC Level 5 HND Diploma in Computing** | | | | |
| **Unit number and title** | **﻿WEBG301 – Project Web** | | | | |
| **Submission date** | 12/07/2022 | **Date Received 1st submission** | |  |
| **Re-submission Date** |  | **Date Received 2nd submission** | |  |
| **Group** | **Student ID & Name** | **Final Score** | | **Student’s signature** |
| 1. Nguyen Van Lap |  | | GCC200110 |
| 2. Tran Ngoc Tang |  | | GCC200247 |
| 3. Ho Kien Vinh |  | | GCC200101 |
| **Student declaration**  I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice. | | | | | |
| **Class** | GCC0902 | | **Assessor name** | Tran Thi Kim Khanh | |

### OBSERVATION RECORD

|  |  |  |  |
| --- | --- | --- | --- |
| **Student name:** | **Nguyen Van Lap** | | |
| **Description of activity undertaken** | | | |
|  | | | |
| **Assessment criteria** | | | |
|  | | | |
| **How the activity meets the requirements of the assessment criteria** | | | |
|  | | | |
| **Student signature:** |  | **Date:** |  |
| **Assessor name:** | **Tran Thi Kim Khanh** | | |
| **Assessor signature:** |  | **Date:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Student name:** | **Tran Ngoc Tang** | | |
| **Description of activity undertaken** | | | |
|  | | | |
| **Assessment criteria** | | | |
|  | | | |
| **How the activity meets the requirements of the assessment criteria** | | | |
|  | | | |
| **Student signature:** |  | **Date:** |  |
| **Assessor name:** | **Tran Thi Kim Khanh** | | |
| **Assessor signature:** |  | **Date:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Student name:** | **Ho Kien Vinh** | | |
| **Description of activity undertaken** | | | |
|  | | | |
| **Assessment criteria** | | | |
|  | | | |
| **How the activity meets the requirements of the assessment criteria** | | | |
|  | | | |
| **Student signature:** |  | **Date:** |  |
| **Assessor name:** | **Tran Thi Kim Khanh** | | |
| **Assessor signature:** |  | **Date:** |  |

|  |  |  |
| --- | --- | --- |
| **❒ Summative Feedback: ❒ Resubmission Feedback:** | | |
| **Grade:** | **Assessor Signature:** | **Date:** |
| **Internal Verifier’s Comments:** | | |
| **IV Signature:** | | |

Table of Contents

[OBSERVATION RECORD 2](#_Toc108340106)

[Group Report Structure 9](#_Toc108340107)

[Chapter 1 – Users’ requirements 9](#_Toc108340108)

[Chapter 2 – System Design 9](#_Toc108340109)

[2.1 About Symfony 9](#_Toc108340110)

[2.2. Git/GitHub 12](#_Toc108340111)

[2.3 Use Case Diagram 18](#_Toc108340112)

[2.4 Entity Relationship Diagram 18](#_Toc108340113)

[2.5 Sitemap 19](#_Toc108340114)

[2.6 System Architecture Diagram 20](#_Toc108340115)

[Chapter 3 – Implementation 20](#_Toc108340116)

[3.1 Sample Source Code 20](#_Toc108340117)

[3.2 Images of final Application 20](#_Toc108340118)

[3.3 GitHub Repository evidences 20](#_Toc108340119)

[Chapter 4 – Conclusion 20](#_Toc108340120)

[4.1 What went well 20](#_Toc108340121)

[4.2 What did not go well 20](#_Toc108340122)

[4.3 Lessons learned and further improvements 21](#_Toc108340123)

[References 22](#_Toc108340124)

Table of figures

[Figure 1 Symfony workflow 10](#_Toc108339332)

[Figure 2 The no branch workflow 15](#_Toc108339333)

[Figure 3 1st PR on another repo with branching 15](#_Toc108339334)

[Figure 4 2nd-nth PR on another repo with branching 15](#_Toc108339335)

[Figure 5 The different between Git and Github 16](#_Toc108339336)

[Figure 6 Use case diagram 16](#_Toc108339337)

[Figure 7 ERD 17](#_Toc108339338)

[Figure 8 Site map before logging in 17](#_Toc108339339)

[Figure 9 Site map after logged in 18](#_Toc108339340)

[Figure 10 System architecture diagram 18](#_Toc108339341)

Table of tables

[Table 1 The branch workflow 15](#_Toc108339372)

### Group Report Structure

# Chapter 1 – Users’ requirements

The homepage must be created using the Section with the appropriate logo, a header can be added that will display images of a variety of clothing and footwear.

This site only sells Apple iPhones, so the interface needs to be designed accordingly.

Phone specs and price should be added with pictures

Customers can register as a member of the store and then can log in to the website

Admin can perform the following functions: Add, delete, (…) products

Admin can perform the following functions: Add, delete, (…) product categories.

# Chapter 2 – System Design

## 2.1 About Symfony

Symfony – Introduction

A PHP web framework is a collection of classes, which helps to develop a web application. Symfony is an open-source MVC framework for rapidly developing modern web applications. Symfony is a full-stack web framework. It contains a set of reusable PHP components. You can use any Symfony components in applications, independently from the framework.

Symfony has a huge amount of functionality and active community. It has a flexible configuration using YAML, XML, or annotations. Symfony integrates with an independent library and PHP Unit. Symfony is mainly inspired by Ruby on Rails, Django, and Spring web application frameworks. Symfony components are being used by a lot of open source projects that include Composer, Drupal, and phpBB.

The Symfony framework consists of several components, such as the HttpFoundation component that understands HTTP and offers a nice request and response object used by the other components. Others are merely helper components, such as the Validator, that helps to validate data. Kernel component is the heart of the system. Kernel is basically the ‘main class’ that manages the environment and has the responsibility of handling a http request.

Symfony’s well-organized structure, clean code, and good programming practices make web development easier. Symfony is very flexible, used to build micro-sites and handle enterprise applications with billions of connections (Tutorialspoint, 2022).

**Symfony Framework - Features**

Symfony is designed to optimize the development of web applications and grows in features with every release.

Some of the salient features of Symfony Framework is as follows (Tutorialspoint, 2022)­­:

* Model-View-Controller based system
* High-performance PHP framework
* Flexible URI routing
* Code reusable and easier to maintain
* Session management
* Error logging
* Full-featured database classes with support for several platforms
* Supports a huge and active community
* Set of decoupled and reusable components
* Standardization and interoperability of applications
* Security against cross-site request forgery and other attacks
* Twig template engine

**Symfony - Architecture**

**Web Framework** (Tutorialspoint, 2022)**:**

Symfony is mainly designed to write high-quality web applications with relative ease. It provides various options to write different types of web applications from simple web site to advanced REST-based web services. Symfony provides web frameworks as separate bundles. The common bundles used in the Symfony web framework are as follows −

* FrameworkBundle
* FrameworkExtraBundle
* DoctrineBundle

Symfony web framework is based on Model-View-Controller (MVC) architecture. Model represents the structure of our business entities. View shows the models to the user in the best possible way depending on the situation. Controller handles all the request from the user, does the actual work by interacting with Model and finally provides the View with the necessary data to show it to the user.

Symfony web framework provides all the high-level features required for an enterprisegrade application. Following is a simple workflow of Symfony web application.

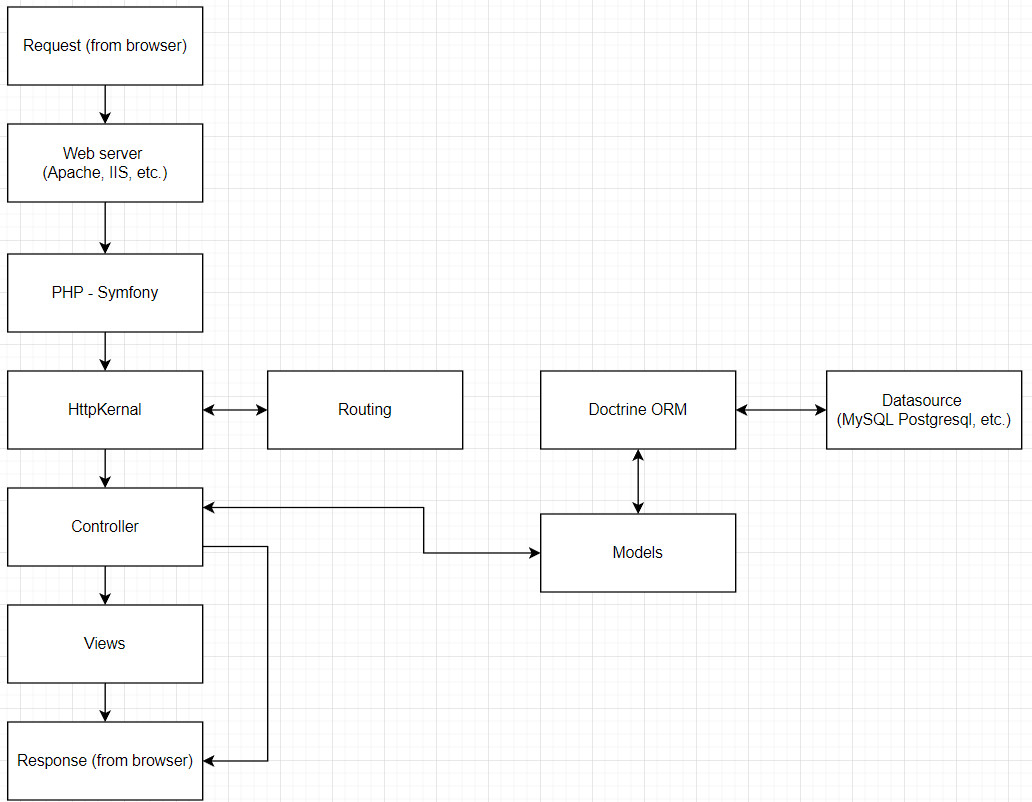


Figure 1 Symfony workflow

The workflow consists of the following steps.

**Step 1 −** The user sends a request to the application through the browser, say http://www.symfonyexample.com/index.

**Step 2 −** The browser will send a request to the web server, say Apache web server.

**Step 3 −** The web server will forward the request to the underlying PHP, which in turn sends it to the Symfony web framework.

**Step 4 −** HttpKernel is the core component of the Symfony web framework. HttpKernel resolves the controller of the given request using Routing component and forward the request to the target controller.

**Step 5 −** All the business logic takes place in the target controller.

**Step 6 −** The controller will interact with Model, which in turn interacts with Datasource through Doctrine ORM.

**Step 7 −** Once the controller completes the process, it either generates the response itself or through View Engine, and sends it back to the web server.

**Step 8 −** Finally, the response will be sent to the requested browser by the web server.

## 2.2. Git/GitHub

**What is Git?**

Git is a version control system which lets you track changes you make to your files over time. With Git, you can revert to various states of your files (like a time traveling machine). You can also make a copy of your file, make changes to that copy, and then merge these changes to the original copy.

(Abba, 2021)

**Basic Git commands**

To use Git, developers use specific commands to copy, create, change, and combine code. These commands can be executed directly from the command line or by using an application like GitHub Desktop. Here are some common commands for using Git:

* git init initializes a brand new Git repository and begins tracking an existing directory. It adds a hidden subfolder within the existing directory that houses the internal data structure required for version control.
* git clone creates a local copy of a project that already exists remotely. The clone includes all the project's files, history, and branches.
* git add stages a change. Git tracks changes to a developer's codebase, but it's necessary to stage and take a snapshot of the changes to include them in the project's history. This command performs staging, the first part of that two-step process. Any changes that are staged will become a part of the next snapshot and a part of the project's history. Staging and committing separately gives developers complete control over the history of their project without changing how they code and work.
* git commit saves the snapshot to the project history and completes the change-tracking process. In short, a commit functions like taking a photo. Anything that's been staged with git add will become a part of the snapshot with git commit.
* git status shows the status of changes as untracked, modified, or staged.
* git branch shows the branches being worked on locally.
* git merge merges lines of development together. This command is typically used to combine changes made on two distinct branches. For example, a developer would merge when they want to combine changes from a feature branch into the main branch for deployment.
* git pull updates the local line of development with updates from its remote counterpart. Developers use this command if a teammate has made commits to a branch on a remote, and they would like to reflect those changes in their local environment.
* git push updates the remote repository with any commits made locally to a branch.
* git rebase command allows you to easily change a series of commits, modifying the history of your repository. You can reorder, edit, or squash commits together.

(github, n.d.)

How Does Git Work?

Git allows users to track code changes and manage their project using simple commands.

The heart of Git is a repository used to contain a project. A repository can be stored locally or on a website, such as GitHub. Git allows users to store several different repositories and track each one independently.

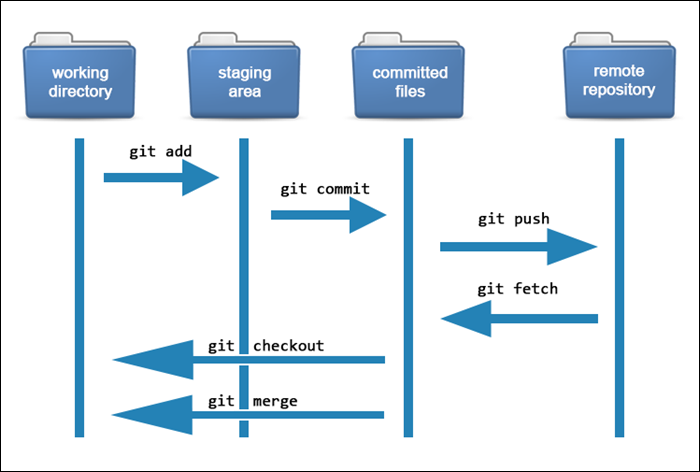
Throughout development, the project has several save points, called **commits**. The commit history contains all the commits, i.e., changes implemented in the project during development. A commit allows you to roll back or fast forward the code to any commit in the commit history.

Git uses **SHA-1 hashes** to refer to the commits. Each unique hash points to a particular commit in the repository. Using hashes, Git creates a tree-like structure to store and retrieve data easily.

The files in each Git project go through several stages:

* **Working directory**. Modified files, but untracked and not yet ready for commit.
* **Staging directory**. Adding modified files to the staging environment means they are ready for commit.
* **Committed**. Snapshots of files from the staging area saved in the commit history.

The following diagram shows the basic Git workflow:



(phoenixnap, n.d.)

**What is the repository?**

A repository, or Git project, encompasses the entire collection of files and folders associated with a project, along with each file's revision history. The file history appears as snapshots in time called commits. The commits can be organized into multiple lines of development called branches. Because Git is a DVCS, repositories are self-contained units and anyone who has a copy of the repository can access the entire codebase and its history. Using the command line or other ease-of-use interfaces, a Git repository also allows for: interaction with the history, cloning the repository, creating branches, committing, merging, comparing changes across versions of code, and more.

Through platforms like GitHub, Git also provides more opportunities for project transparency and collaboration. Public repositories help teams work together to build the best possible final product.

[(github, n.d.)](https://docs.github.com/en/get-started/using-git/about-git)

**What is GitHub?**

GitHub is an online hosting service for Git repositories. Imagine working on a project at home and while you are away, maybe at a friend's place, you suddenly remember the solution to a code error that has kept you restless for days.

You cannot make these changes because your PC is not with you. But if you have your project hosted on GitHub, you can access and download that project with a command on whatever computer you have access to. Then you can make your changes and push the latest version back to GitHub.

In summary, GitHub lets you store your repo on their platform. Another awesome feature that comes with GitHub is the ability to collaborate with other developers from any location.

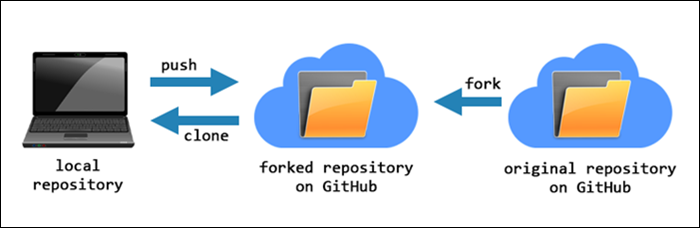
(Abba, 2021)

Forking

A fork is a complete **copy of an existing repository** that allows you to make changes and experiment without affecting the original project. Forking is a way for someone to propose changes to an existing project, or it can be a starting point for a project of your own if the code is open source.

If you want to propose a change or a bug fix for a project, you can fork a repository, make the fix, and make a pull request to the project owner.

The following diagram illustrates how forking works:



(phoenixnap, n.d.)

**How Github works**

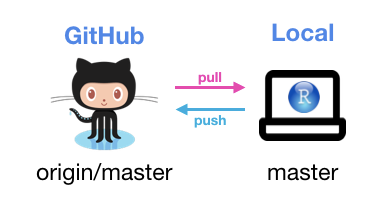


Figure 2 The no branch workflow

|  |  |
| --- | --- |
|  |  |
| Figure 3 1st PR on another repo with branching | Figure 4 2nd-nth PR on another repo with branching |

Table 1 The branch workflow

(Robbins, 2022)

**The differences between Git and Github**

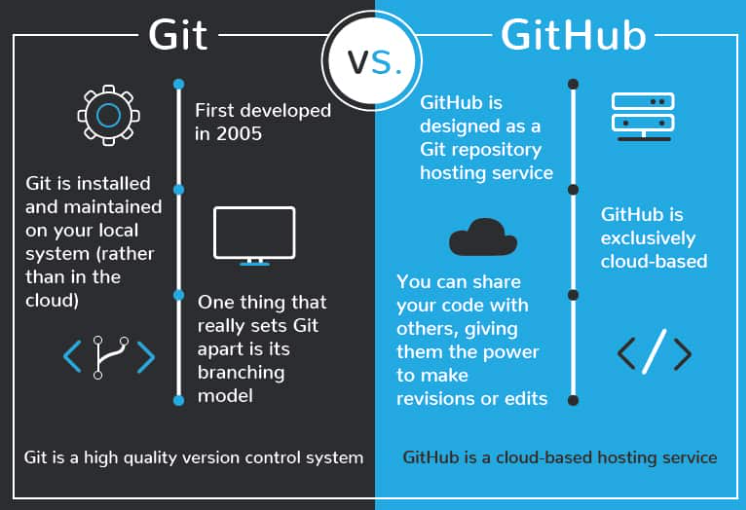


Figure 5 The different between Git and Github

(devmountain, 2022)

## 2.3 Use Case Diagram

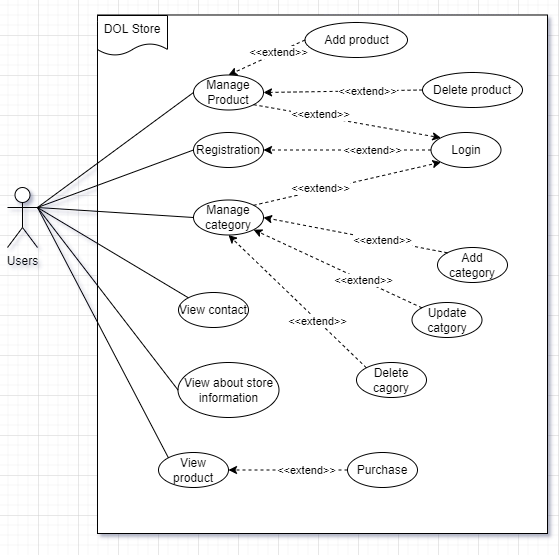


Figure 6 Use case diagram

## 2.4 Entity Relationship Diagram

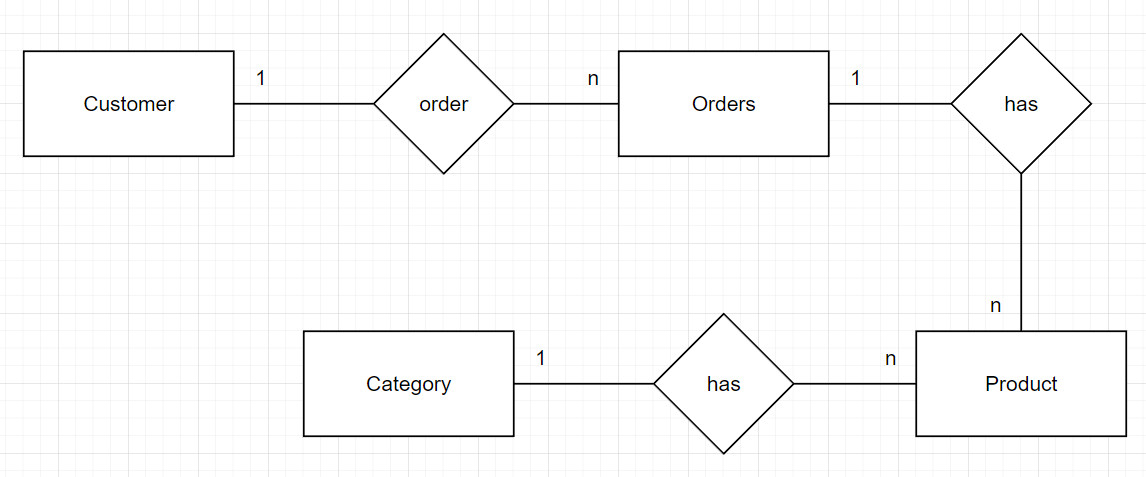


Figure 7 ERD

## 2.5 Sitemap

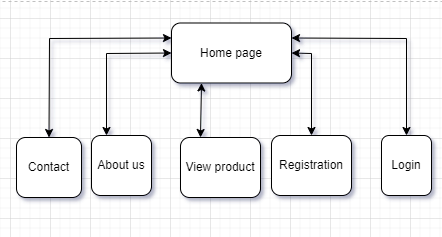


Figure 8 Site map before logging in

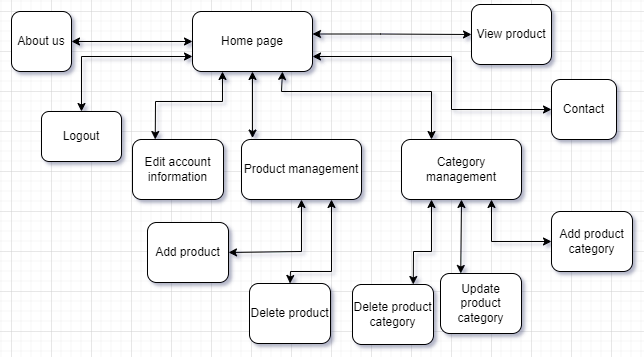


Figure 9 Site map after logged in

## 2.6 System Architecture Diagram

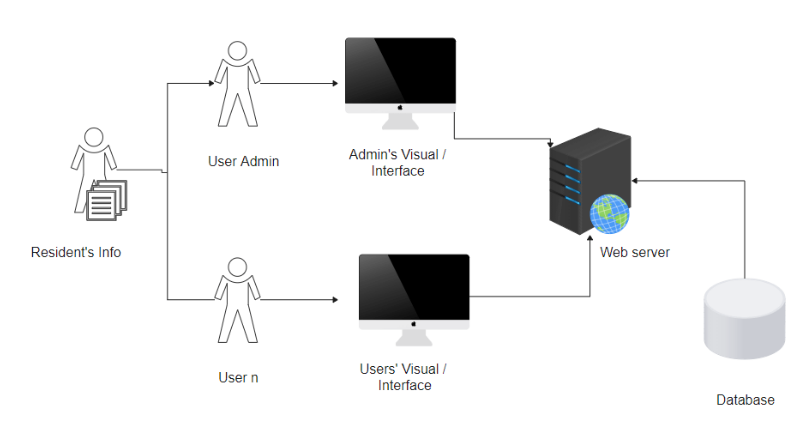
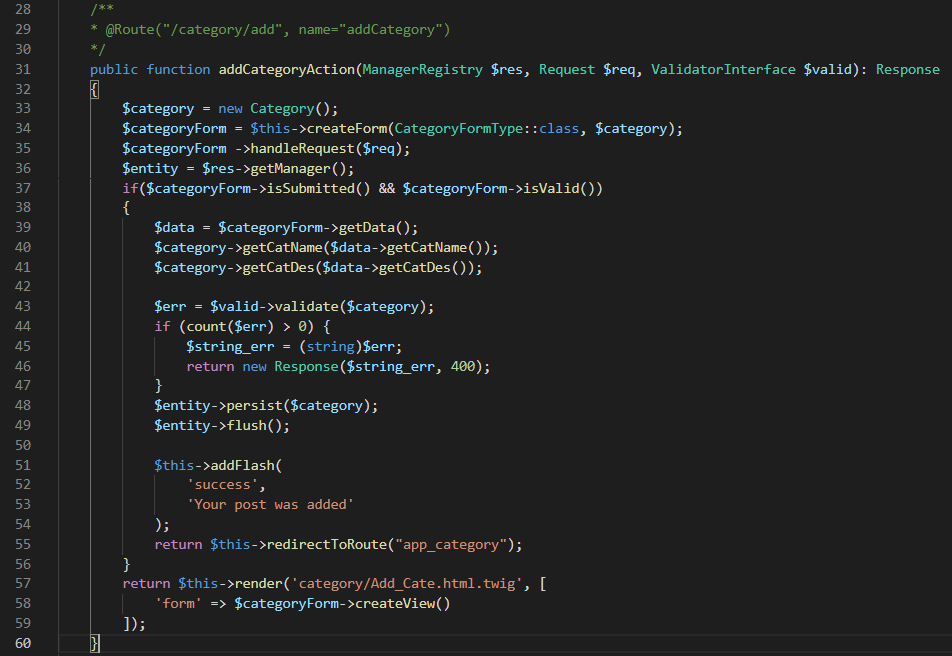


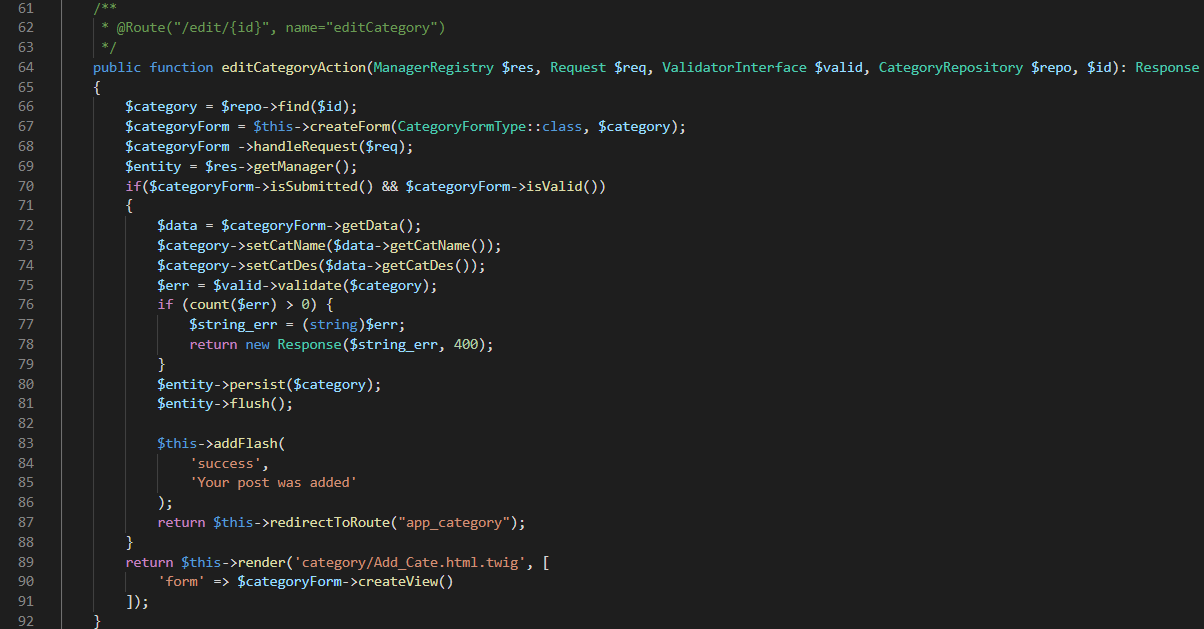
Figure 10 System architecture diagram

(edrawsoft, 2022)

# Chapter 3 – Implementation

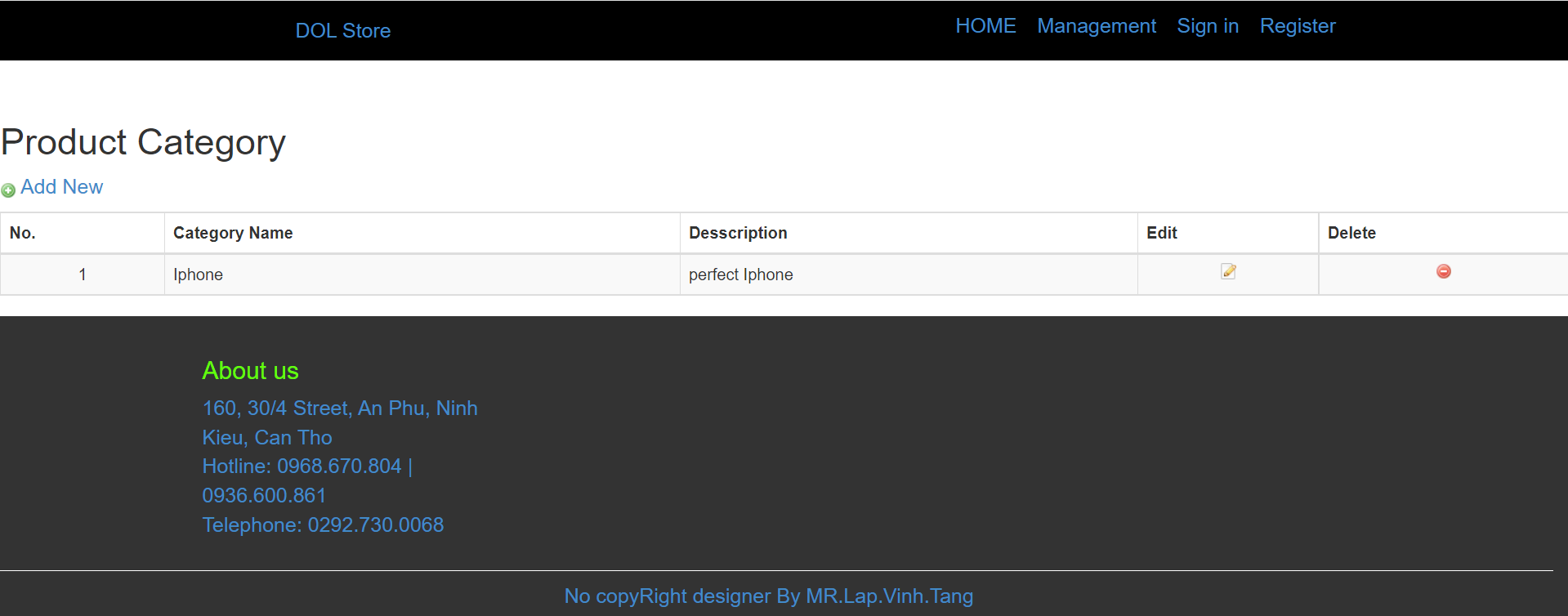
## 3.1 Sample Source Code

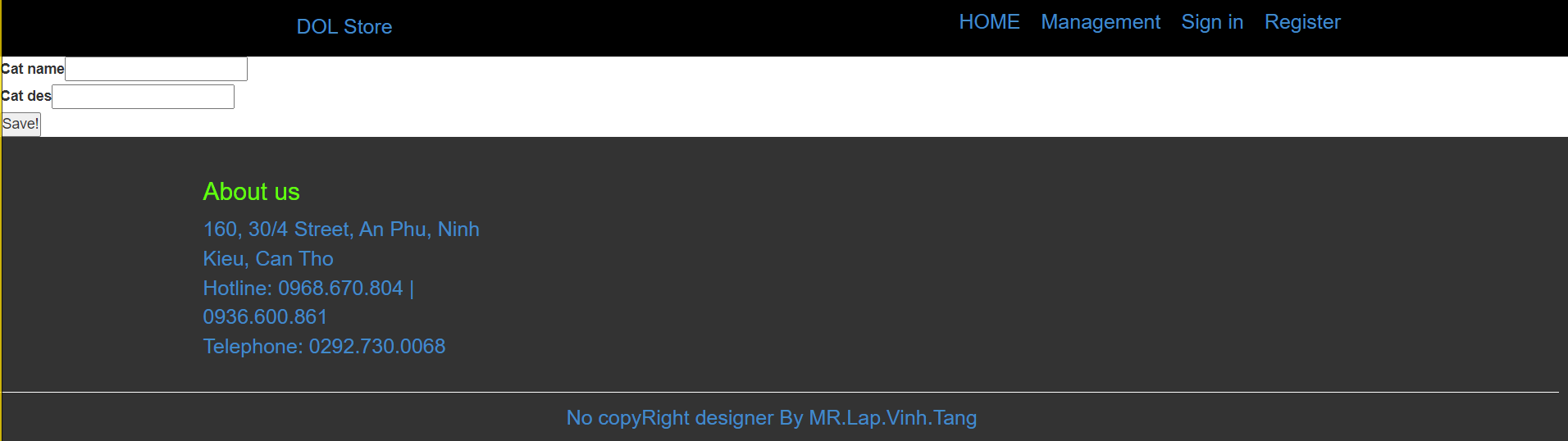




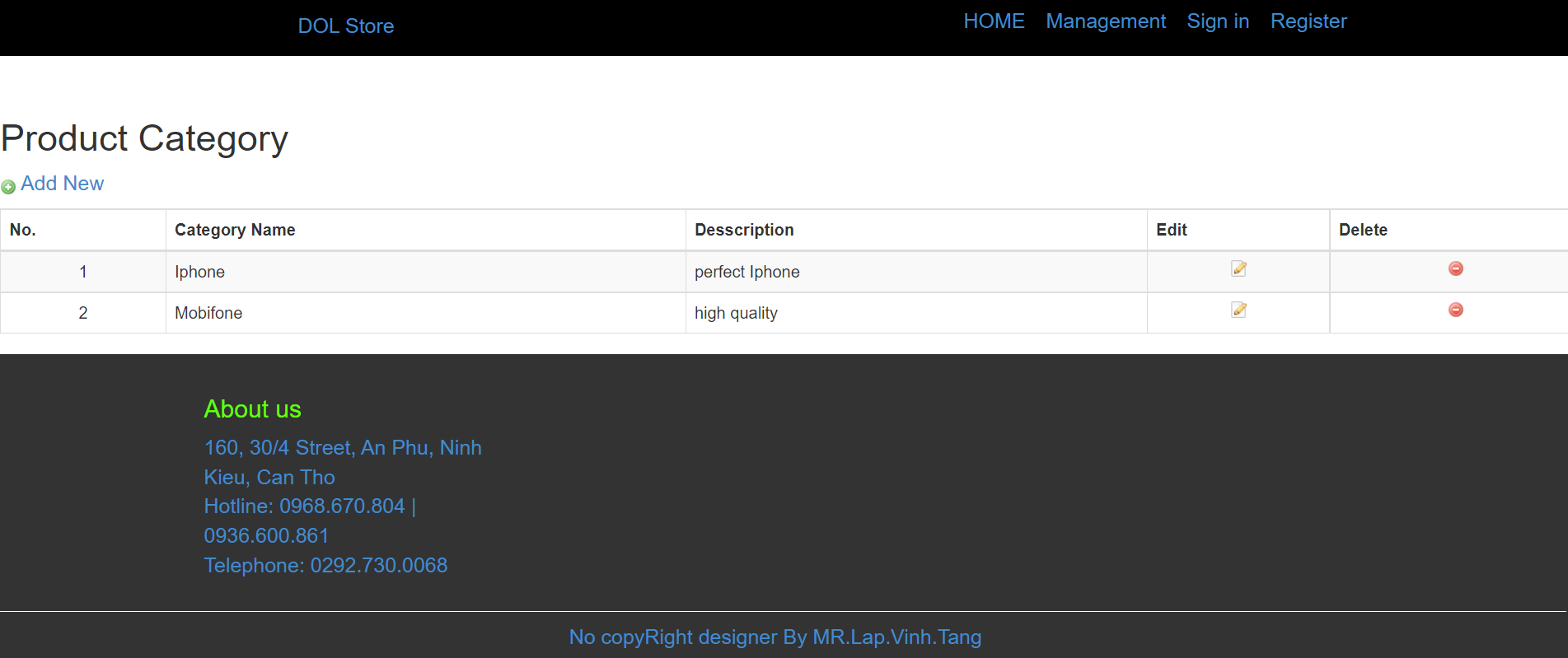


## 3.2 Images of final Application



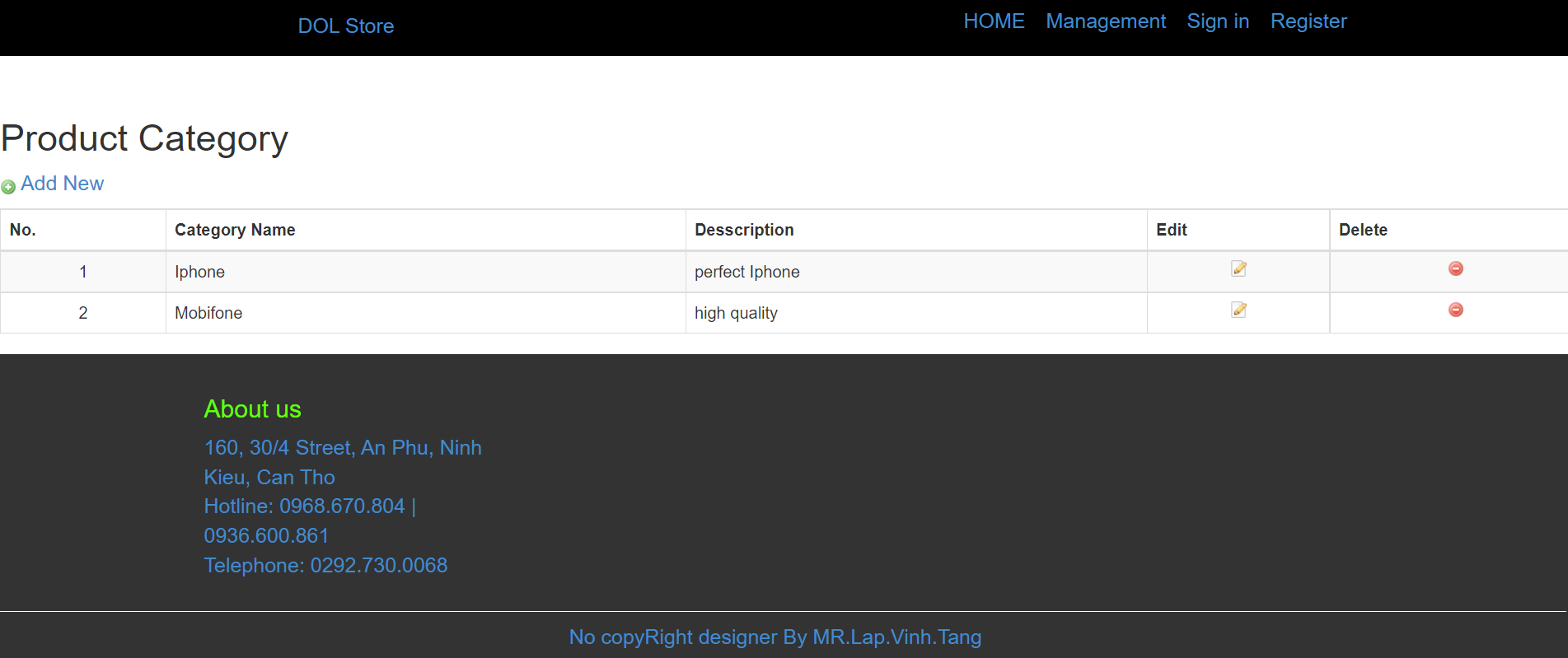


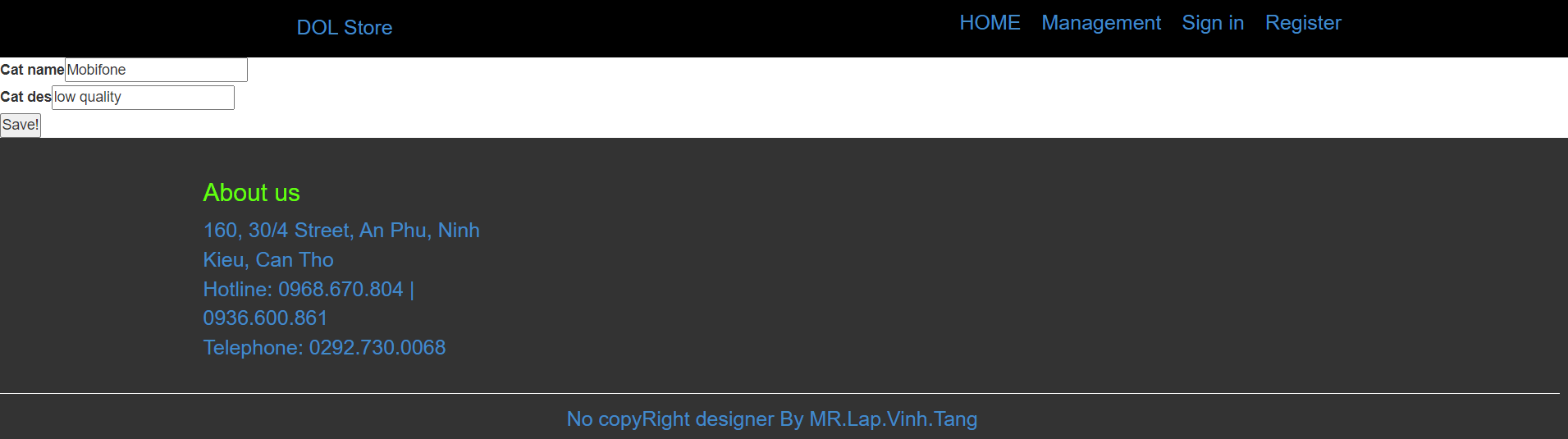


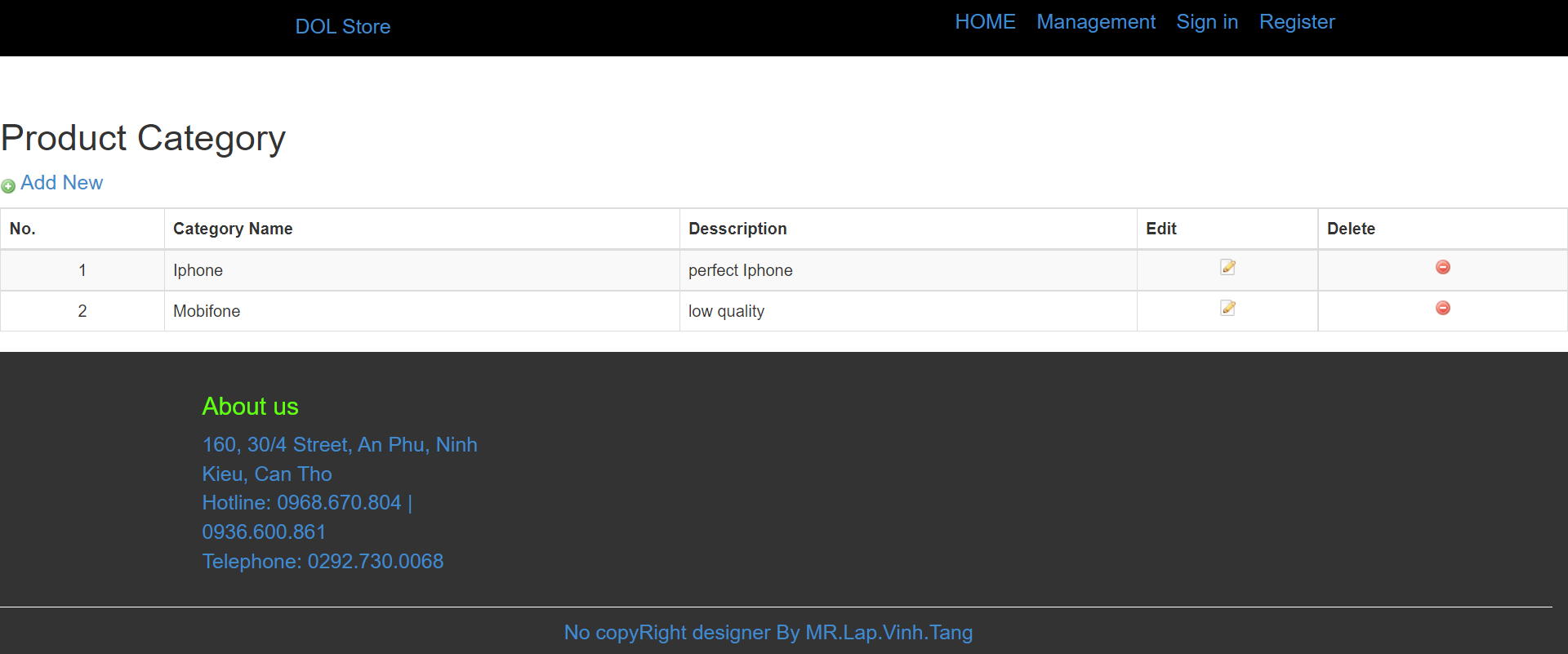


////////////////////////////////////////////////////////////////////////////////////////////////////////////

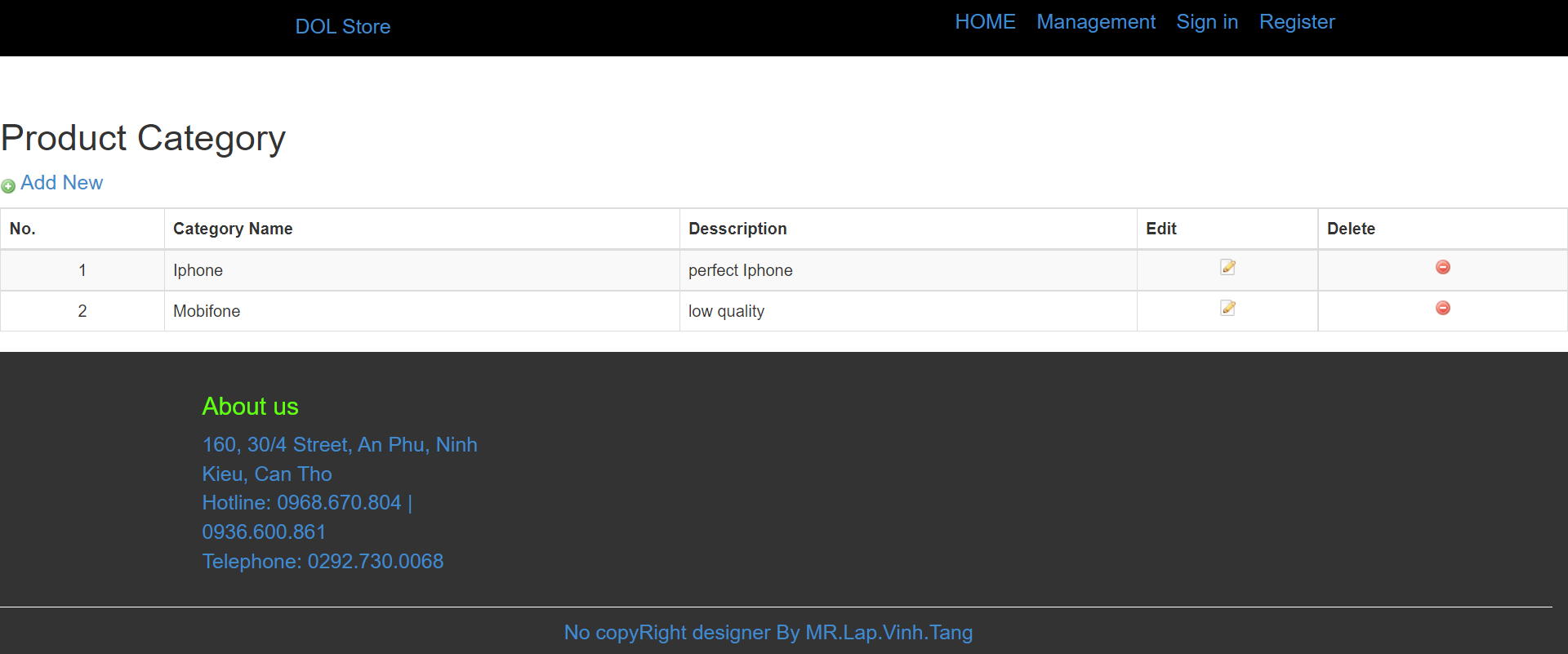
edit

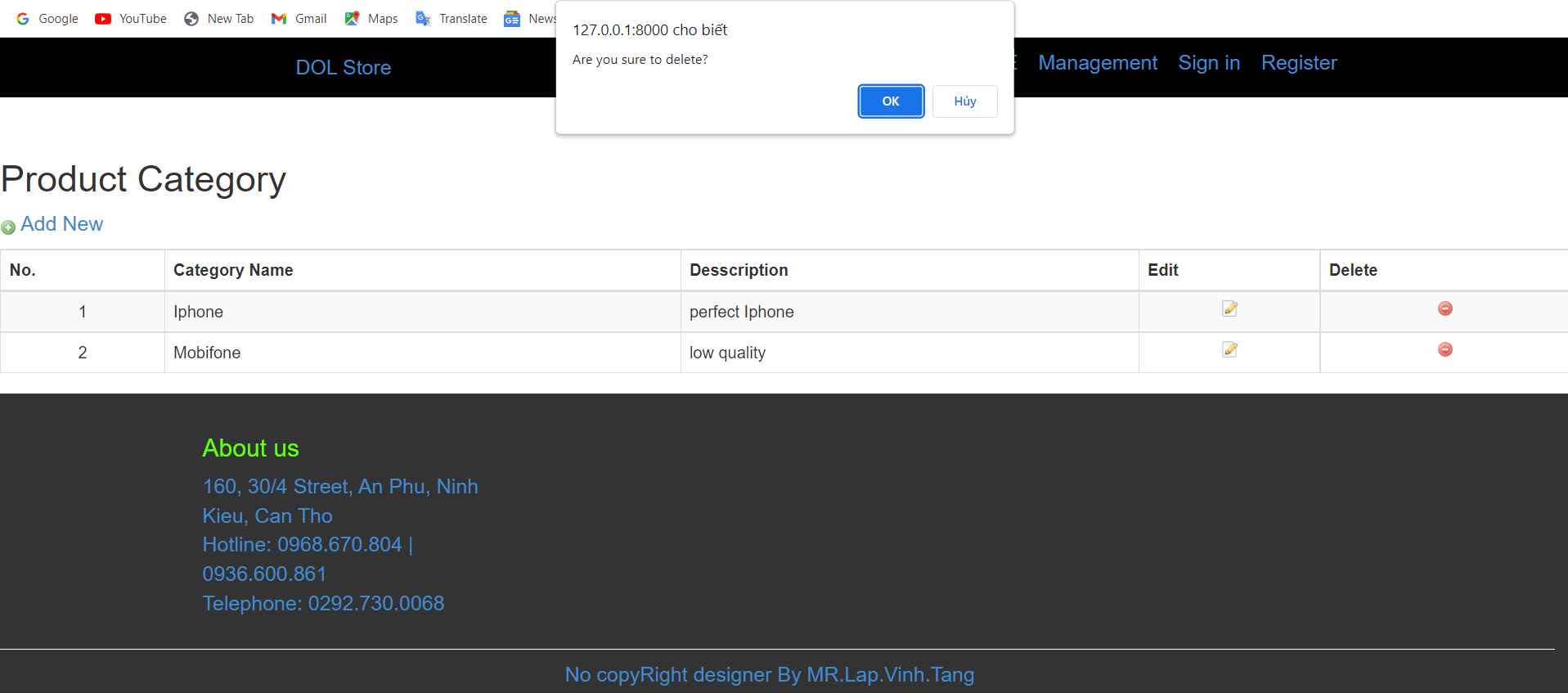


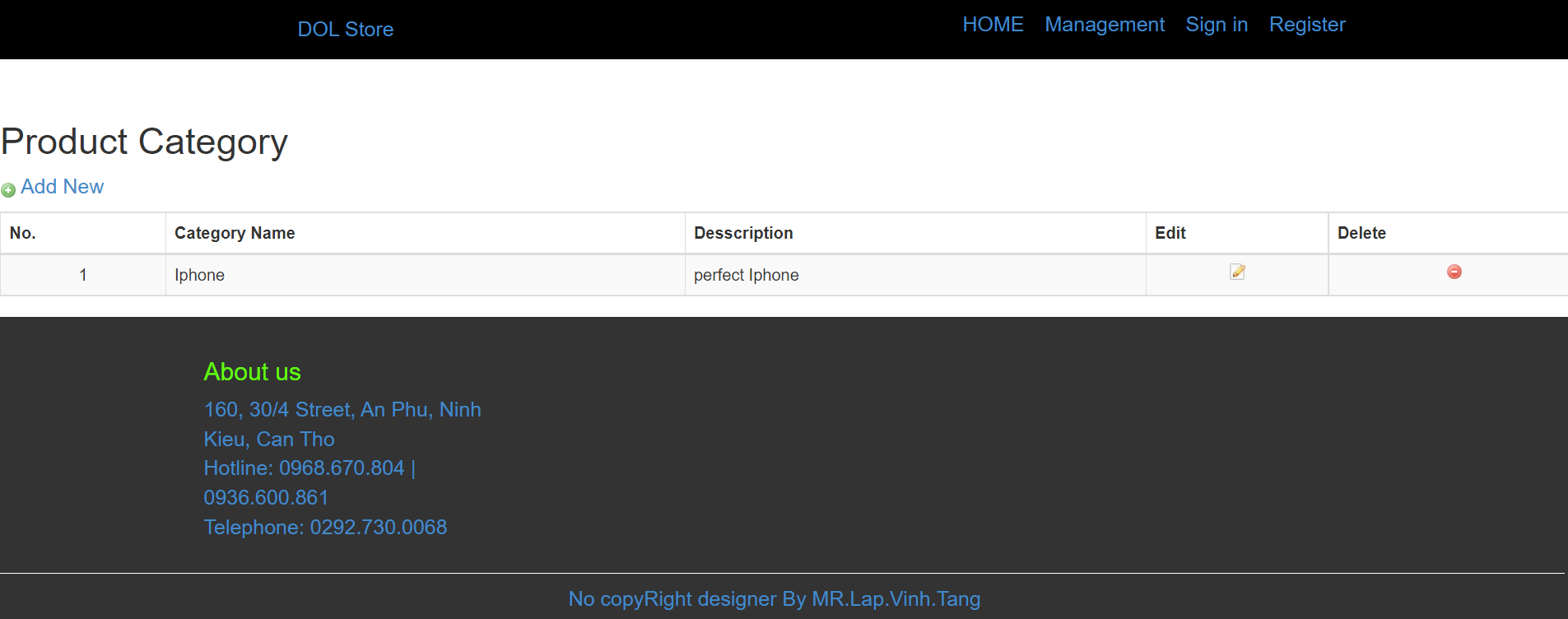




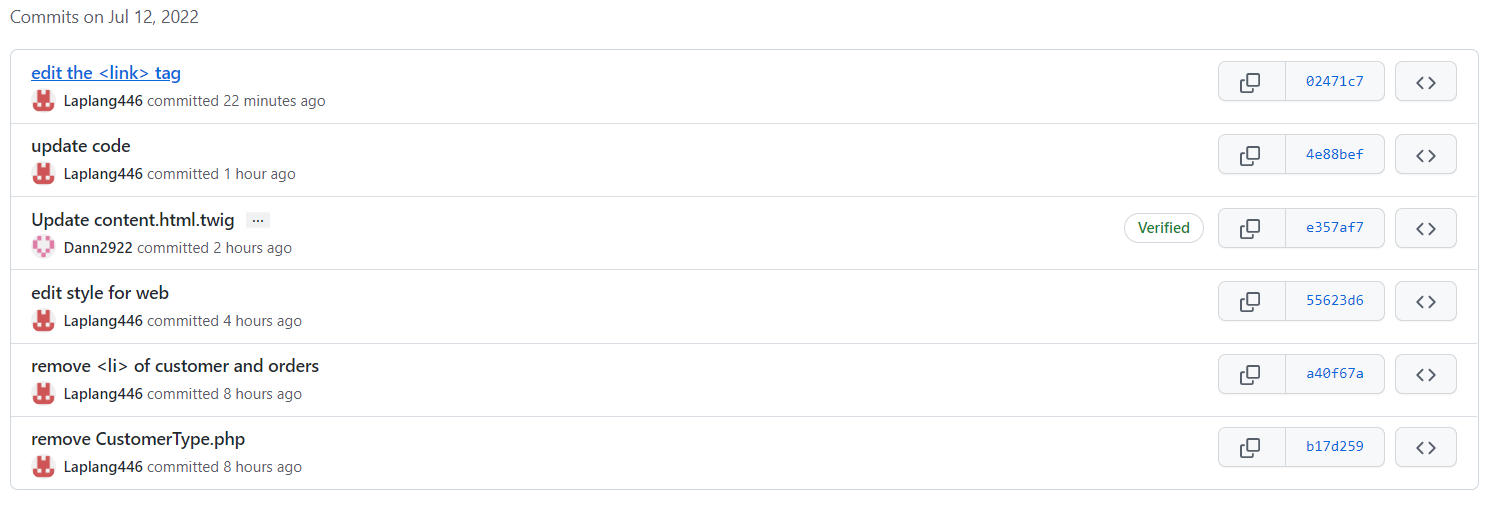
//////////////////////////////////////////////////////////////////////////////////

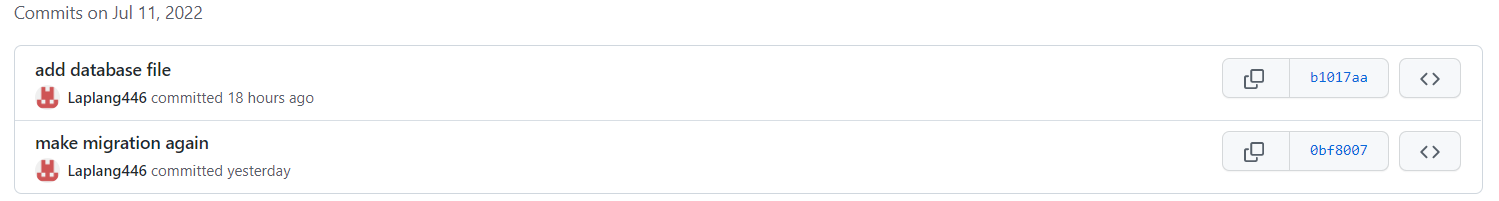


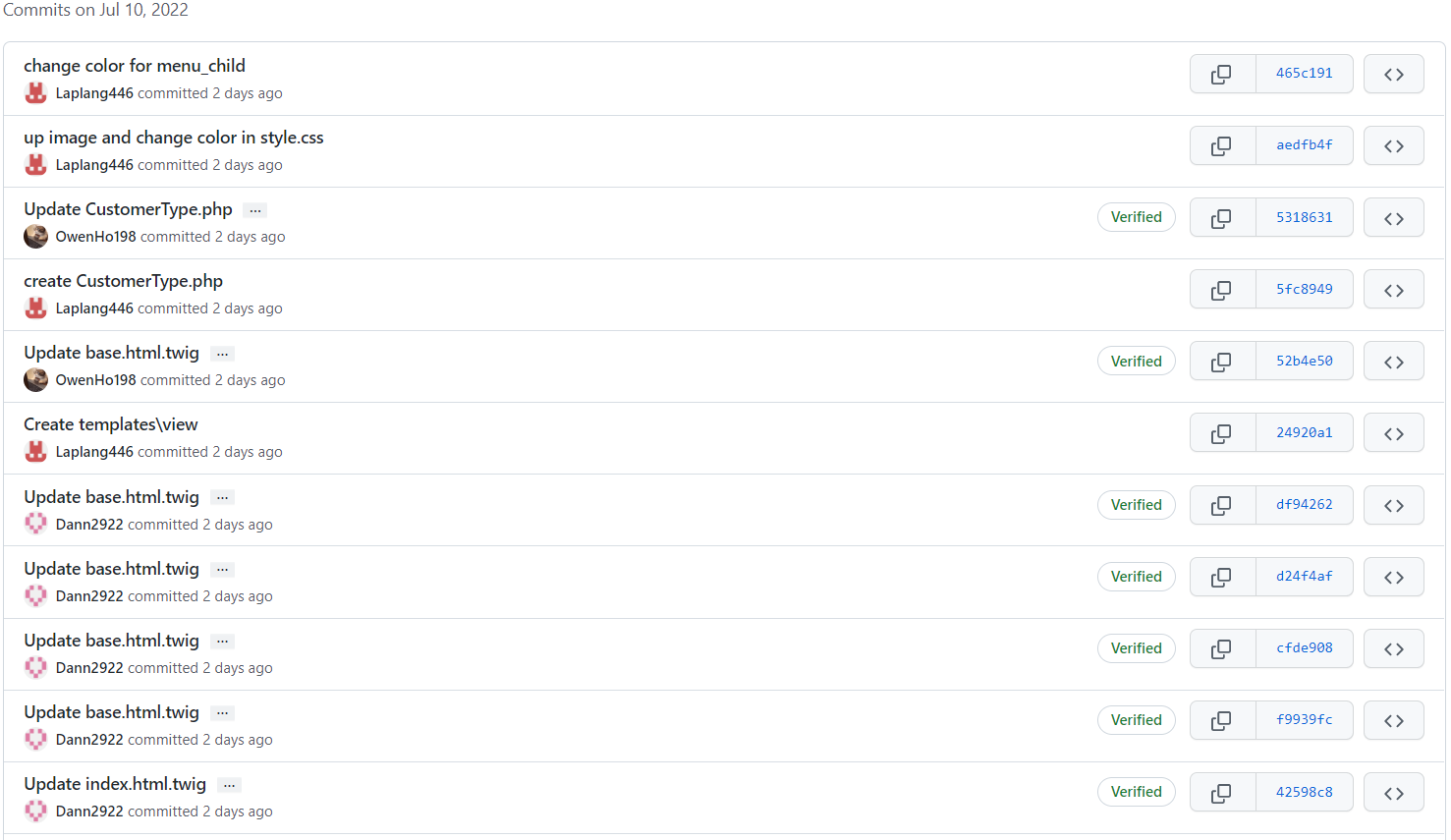


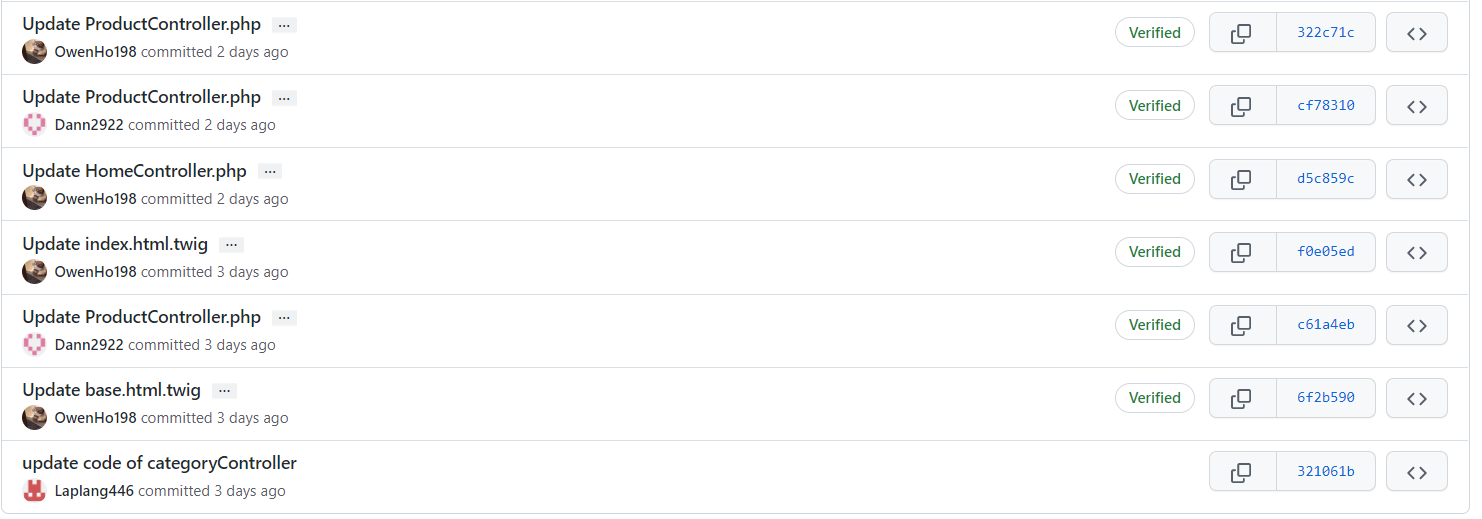


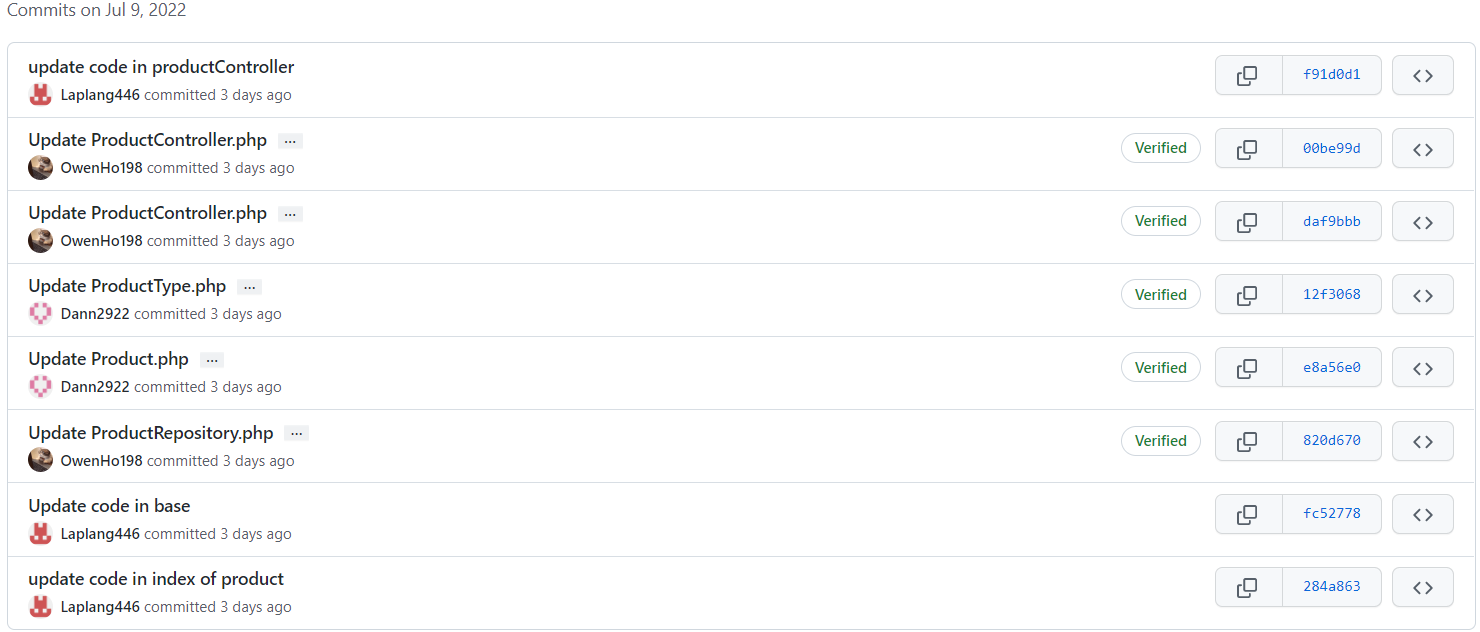
## 3.3 GitHub Repository evidences











# Chapter 4 – Conclusion

## 4.1 What went well

## 4.2 What did not go well

## 4.3 Lessons learned and further improvements

# References

Abba, I. V., 2021. *freecodecamp.* [Online]   
Available at: https://www.freecodecamp.org/news/git-and-github-for-beginners/  
[Accessed 2022].

Abba, I. V., 2021. *freecodecamp.* [Online]   
Available at: https://www.freecodecamp.org/news/git-and-github-for-beginners/  
[Accessed 2022].

devmountain, 2022. *devmountain.* [Online]   
Available at: https://devmountain.com/blog/git-vs-github-whats-the-difference/#:~:text=GitHub%E2%80%A6-,what's%20the%20difference%3F,help%20you%20better%20manage%20them.  
[Accessed 2022].

edrawsoft, 2022. *edrawsoft.* [Online]   
Available at: https://www.edrawsoft.com/article/system-architecture-diagram.html  
[Accessed 2022].

github, n.d. *github.* [Online]   
Available at: https://docs.github.com/en/get-started/using-git/about-git  
[Accessed 2022].

github, n.d. *github.* [Online]   
Available at: https://docs.github.com/en/get-started/using-git/about-git  
[Accessed 2022].

phoenixnap, n.d. *phoenixnap.* [Online]   
Available at: https://phoenixnap.com/kb/how-git-works#:~:text=Git%20allows%20users%20to%20track,and%20track%20each%20one%20independently  
[Accessed 2022].

Robbins, J., 2022. *edav.info.* [Online]   
Available at: https://edav.info/github.html#nd-nth-pr-on-another-repo-with-branching  
[Accessed 2022].

Tutorialspoint, 2022. [Online]   
Available at: https://www.tutorialspoint.com/symfony/symfony\_introduction.htm  
[Accessed 1 7 2022].

Tutorialspoint, 2022. [Online]   
Available at: https://www.tutorialspoint.com/symfony/symfony\_architecture.htm  
[Accessed 10 7 2022].