CS100709 - Website Development Project

A Website is one of the most important units of information publishing, sharing in the era of the Internet. Imaging a life without any websites? Therefore, knowing how to design and build a website is a necessity for almost everyone, especially those involve in branches of Informatics. Once you mastered the technique to building a website, you will be able to share your interesting moments on a blog, to manage and share information within an organization using a Wiki, or to help your organization or company broaden its influence, so on and so forth.

(From Wikipedia) Websites have many functions and can be used in various fashions; a website can be a personal website, a commercial website, a government website or a non-profit organization website. Any website can contain a hyperlink to any other website. Websites are written in, or converted to, HTML (Hyper Text Markup Language) and can be viewed or otherwise accessed from a range of computer-based and Internet-enabled devices of various sizes, including desktop computers, laptops, PDAs and cell phones.

Requirements: Working in groups, designing a website for an individual, an organization, a course etc. The website should include both front-end and back-end. You are the BOSS of the theme of your website.

Techniques: The development should be based on FLASK or similar frameworks (FLASK is the simplest one!). The choice of database is flexible, MySQL is highly recommended.

References:

Website background: http://en.wikipedia.org/wiki/Website

HTML-PHP-CSS basics: http://www.w3schools.com/

Bootstrap: http://getbootstrap.com/

FLASK:

- Website: https://palletsprojects.com/p/flask/
- Documentation: https://flask.palletsprojects.com/
- Releases: https://pypi.org/project/Flask/

MySQL: https://www.mysql.com/

Evaluation:

You will demonstrate your website during the Project Evaluation (see the schedule of your class).

You should give presentations about the idea behind your website, the techniques you adopted, the implementation part as well as the experiences during the project.

The evaluation of each group takes 5 minutes and everyone should have a chance to present.

BONUS: The best website will be made online.

Notice: PLAGIARISM IS STRICTLY FORBIDDEN!

抄袭是严格禁止的！（see why）

Project Instructions

Website in a nutshell

A website is a collection of web pages and related content that is identified by a common domain name and published on at least one web server.

web server

the web server is where the files of your website are residing. When you access a website, you are the client, and the host computer that stores the web pages is the Server, who will return you the web pages.

Domain name & URL

A URL (Uniform Resource Locator) is a unique identifier used to locate a resource on the internet. It is also referred to as a web address. A URL is basically an address on the Web, identifying each document uniquely. The simple syntax for a URL is:

http://host/path, e.g. https://dev.mysql.com/downloads/file/

http: Hypertext Transfer Protocol

host: The name of the machine on which the resource lives. aka, *domain name*

path: A path refers to a file or location on the web server. Just like the file path in your computer.

Architecture

Website architecture is the planning and design of the technical, functional and visual components of a website - before it is designed, developed and deployed. It is used by website designers and developers as a means to design and develop a website.

The **Model-View-Controller (MVC)** is an architectural pattern that separates an application into three main logical components: the **model**, the **view**, and the **controller**. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development framework to create scalable and extensible projects.

Model

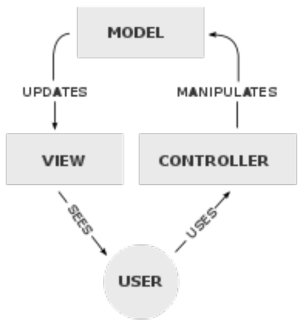
The central component of the pattern. It is the application's dynamic data structure, independent of the user interface. It directly manages the data, logic and rules of the application.

View

Any representation of information such as a chart, diagram or table. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.

Controller

Accepts input and converts it to commands for the model or view.



Advantages

Simultaneous development – Multiple developers can work simultaneously on the model, controller and views.

High cohesion – MVC enables logical grouping of related actions on a controller together. The views for a specific model are also grouped together.

Loose coupling – The very nature of the MVC framework is such that there is low coupling among models, views or controllers

Ease of modification – Because of the separation of responsibilities, future development or modification is easier

Multiple views for a model – Models can have multiple views

Environment

Install MySql

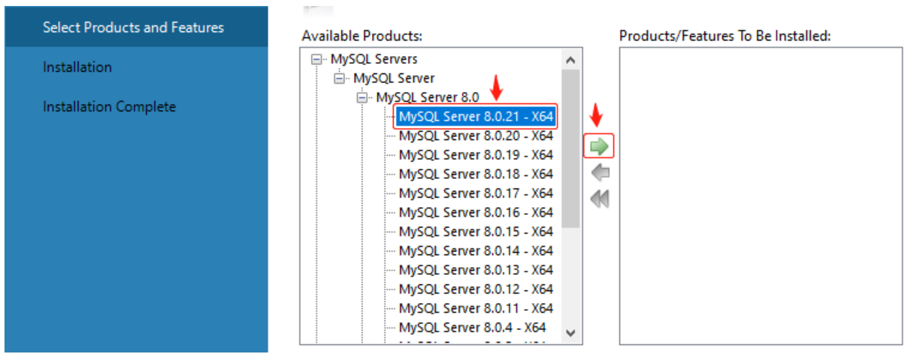
On windows:

1. Enter download page
2. Download Windows (x86, 32-bit) MSI Installer.
3. Execute the installer:

Select “custom” and next.

Select the first one which is x64, click the green arrow in the right, be free to download other products, click next.

Click ‘execute’ to start download, and click ‘next’ until entering passwords is required, remember your password, it will be used soon.


Select Products and Features
Installation
Installation Complete
Available Products:
MySQL Servers
MySQL Server
MySQL Server 8.0
MySQL Server 8.0.21 - X64
MySQL Server 8.0.20 - X64
MySQL Server 8.0.19 - X64
MySQL Server 8.0.18 - X64
MySQL Server 8.0.17 - X64
MySQL Server 8.0.16 - X64
MySQL Server 8.0.15 - X64
MySQL Server 8.0.14 - X64
MySQL Server 8.0.13 - X64
MySQL Server 8.0.12 - X64
MySQL Server 8.0.11 - X64
MySQL Server 8.0.4 - X64
Products/Features To Be Installed:

On Linux:

1. *sudo apt update*
2. *sudo apt install mysql-server*

Install Flask

1. Install python first, from the website, or use Anaconda. We use python3.8 here.
2. windows command line or terminal, enter:

>pip install flask.

Make sure you have included python path in your environment variables and pip installed first.

Example: Hello World!

Quick start

Create a python file xxx.py in anywhere, write down the following code:

```python
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'  # Pay attention to indentation

app.run()
```

So, what did that code do?

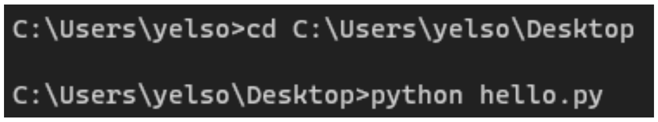
1. First we imported the Flask class. An instance of this class will be our WSGI application.
2. Next we create an instance of this class. The first argument is the name of the application's module or package.
3. We then use the route() decorator to tell Flask what URL should trigger our function.
4. The function is given a name which is also used to generate URLs for that particular function, and returns the message we want to display in the user's browser.

To run the application, open your terminal and change your directory into where the hello.py locates using *cd* command ('cd' command means change directory).

In my case, I create 'hello.py' in my Desktop. So, I run:

> *cd C:\Users\someone\Desktop*

> *python hello.py*



```
C:\Users\yelso>cd C:\Users\yelso\Desktop

C:\Users\yelso\Desktop>python hello.py
```

Or just use one command: > *python C:/Users/yelso/Desktop/hello.py*

If everything goes right, the following information will be displayed:

Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

Now open your browser, and head over to http://127.0.0.1:5000/, and you should see your hello world greeting.

Using html

HTML stands for Hyper Text Markup Language, HTML is the standard markup language for creating Web pages which describes the structure of a Web page.

A Simple HTML tutorial

Example:

```
<!DOCTYPE html>
<html>
  <head>
      <title>Page Title</title>
  </head>
  <body>
      <h1>My First Heading</h1>
      <p>My first paragraph.</p>
  </body>
</html>
```

The `<!DOCTYPE  html>` declaration defines that this document is an HTML5 document

The `<html>` element is the root element of an HTML page

The `<head>` element contains meta information about the HTML page

The `<title>` element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)

The `<body>` element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.

The `<h1>` element defines a large heading

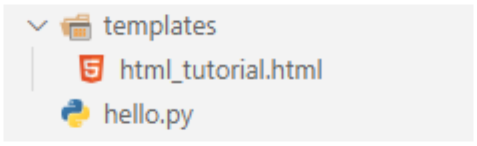
The `<p>` element defines a paragraph

Now add the following code into hello.py:

```
import flask
from flask import request, render_template

app = flask.Flask(__name__)
```

```
# hello world of flask
@app.route('/')
def home():
    return 'Hello World!'

# using  HTML
@app.route('/html')
def html_toturial():
    return render_template('html_tutorial.html')

app.run(debug=True)
```

'render_template()' looks for a template (HTML file) in the templates folder. So you need to create a templates folder and put all your HTML files in there. And import render_template method from the flask framework to use it. Remember to keep the hello.py outside of your templates folder


```
templates
    html_tutorial.html
hello.py
```

run hello.py as before. Then go to [http://127.0.0.1:5000/html.](http://127.0.0.1:5000/html)

>python hello.py

If '*No such file or directory*' occurs, you probably forget to change your directory to where hello.py locates.

Example: A Blog System

This simple blog website is going to have the following components:

A home page in `'/'` displays the latest posts.

A post page in `'/post'`, where we can see the details of the posts.

A page where users can add their posts in `'/add'` page,

A function to add posts to our database.

A function to search posts by index.

app.py

First, create a python file 'app.py', and in that py file we should create five routes:

1. '/', return 'index.html', where newest posts should be displayed ordered by post time;
2. '/about', give a brief intro about yourself and your blog, you can put anything you want on that page.
3. '/post', if we want to see more about a post, click that post, and then we should be redirected to that post page, where all the information displayed.
4. '/add', we can add a post in '/add' page. After filling in all the fields, we can click send button, then a new post will be added to the database, and we will be redirect to the home page, where we can see our new post.
5. '/post/post_id', post_id is the index of a post. For example, we want to see the post 5, we just head to http://127.0.0.1:5000/post/5. we want the post_id to be incremental, which means every time a new post is added, his id increases by one.

Database Setup

First install all the following pkgs,

>pip install flask-sqlalchemy

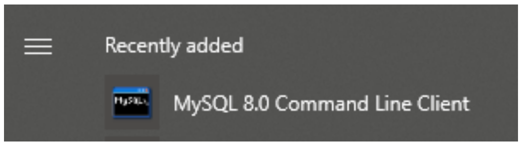
>pip install pymysql

>pip install flask-mysql

>pip install mysqlclient

Create a database for this app:

Open mysql command line:



mysql> CREATE DATABASE blog;

Check what we've done with our db:

mysql> SHOW DATABASES;

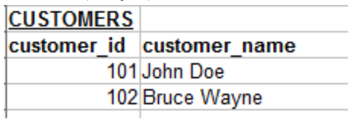
Next, create new file `'init_database.py'`, put it in the same folder as 'app.py'. It helps us to connect to the blog database we just created, and create a table to save all the posts.

This file is available in the attachment, now, open it and take a closer look.

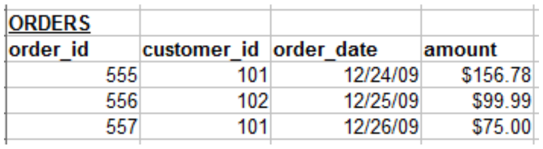
Here are some very important concepts in the relationship database.

A **relational database** is a digital database based on the relational model of data, as proposed by E. F. Codd in 1970.

Relationship database organizes data into one or more tables (or "relations"). A table is defined by some attributes(keys).



CUSTOMERS	
customer_id	customer_name
101	John Doe
102	Bruce Wayne



ORDERS			
order_id	customer_id	order_date	amount
555	101	12/24/09	$156.78
556	102	12/25/09	$99.99
557	101	12/26/09	$75.00

For example, *CUSTOMERS* is a table, *customer_id* and *customer_name* are keys, each row of the table represents a customer.

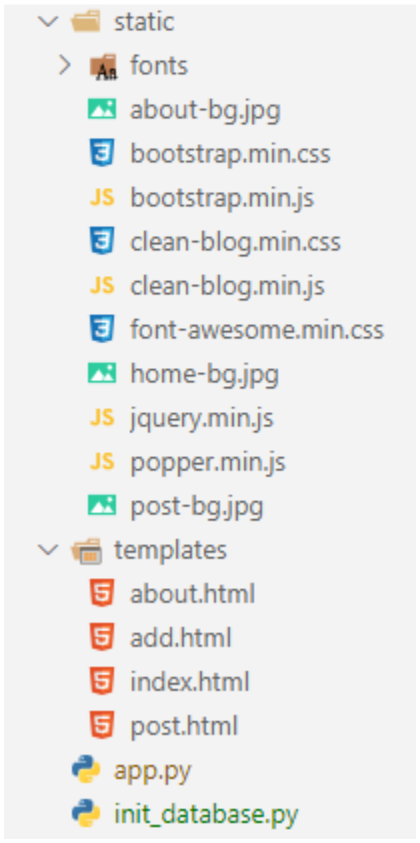
The **Primary Key** of a table is the key to separate a record in the table from other records, it means that a primary key must be unique. customer_id is a primary key, you can find a customer with his customer_id, and no two customers have a same id.

The **Foreign Key** is the key that corresponds to the Primary Key field in the main table. For example, ORDERS is another table, we want to represent the relationship between customers and orders, we can achieve this by adding a foreign key in orders which is *customer_id*: the primary key of CUSTOMERS.

A customer can have several orders, but an order only belongs to one customer. So, this is a one to many relationship.

RUN

The overall structure of this project is shown below.



```
static
  fonts
  about-bg.jpg
  bootstrap.min.css
  bootstrap.min.js
  clean-blog.min.css
  clean-blog.min.js
  font-awesome.min.css
  home-bg.jpg
  jquery.min.js
  popper.min.js
  post-bg.jpg
templates
  about.html
  add.html
  index.html
  post.html
app.py
init_database.py
```

To start the webpages, open command line or terminal, change directory to where the app.py locates. And run:

>Python init_database.py

>Python app.py

You only need to initialize the database once, so run init_database.py only once.

Now go to http://127.0.0.1:5000/, add new posts in your blog, then go to mysql command line:

mysql>use blog;

*mysql>select * from blogpost;*

We selected the blog database first, and then select all posts in that database. All new posts you added will be displayed

We won't go through every line of code, check the codes yourself in the attachment. You can neglect all the files in the 'static' folder, their job is only to make the website look better. If you want to learn more about flask, database, html, css, etc. you can click the links in the references.

References

[1] Website Architecture, < https://www.techopedia.com/definition/30409/website-architecture>

[2] The Building Blocks of Website Architecture, < https://www.keycdn.com/blog/website-architecture>

[3] Flask Installation, < https://flask.palletsprojects.com/en/1.1.x/installation/>

[4] MySQL Installation Guide, < https://dev.mysql.com/doc/mysql-installation-excerpt/5.7/en/>

[5] W3school, https://www.w3schools.com/

[6] Introduction to html, <https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML>

[7] What a Relational Database Is, <https://www.oracle.com/database/what-is-a-relational-database/>

[8] Codecademy, https://www.codecademy.com/learn/learn-sql

[9] PrettyPrinted, flask_blog, <https://github.com/PrettyPrinted/flask_blog>