

# Flask Tutorial

Rafael Castro G. Silva

Department of Computer Science  
University of Copenhagen

20/05/2025

## Learning Goals

# Learning Goals

- 1 Learn what is Flask, and some basic Flask concepts
- 2 Learn the Model View Control (MVC) pattern
- 3 Learn basic Docker

# Flask Introduction

# What is Flask?

# What is Flask?

- Flask is a micro framework for web applications

# What is Flask?

- Flask is a micro framework for web applications
  - Framework: a set of modules and libraries that provide basic functionalities  
May or may not enforce some standards (e.g. directory structures, MVC pattern, etc)
  - Micro: enforces very little
  - In/for Python

# What is Flask?

- Flask is a micro framework for web applications
  - Framework: a set of modules and libraries that provide basic functionalities  
May or may not enforce some standards (e.g. directory structures, MVC pattern, etc)
  - Micro: enforces very little
  - In/for Python

## Other frameworks out there

- Ruby on Rails
- Django (Python)
- Java Spring
- Symfony (PHP)



# Who uses Flask?

# Who uses Flask?



reddit

Uber

# Prerequisites

# Prerequisites

- Very basic Python
  - Variables, functions, indentation, lists, maps (dictionaries), if-then-else, for loops, class (OO programming), import libraries
- Terminal commands: `ls`, `cd`, `pwd`...
- Basic HTML
- CSS (optional)
- SQL

# Flask Concepts

# The App Variable

- Our app is an instance of the Flask class
- `__name__` is a default configuration telling that the files of the project are in the current directory

```
1 from flask import Flask
2
3 app = Flask(__name__)
```

# Routing

```
1 from flask import Flask
2
3 app = Flask(__name__)
4
5 @app.route("/")
6 def hello_world():
7     return "<p>Hello, World!</p>"
```

- `@app.route("/")` is Python decorator: extends the behavior of a function
  - Tells Flask what URL should trigger our function
  - The `"/"` is the root of our web site domain  
`http://127.0.0.1:5000` takes us to this route
  - We can have other routes: `@app.route("/register")`  
`http://127.0.0.1:5000/register` takes us to this other route

# Routing

```
1 from flask import Flask
2
3 app = Flask(__name__)
4
5 @app.route("/")
6 def hello_world():
7     return "<p>Hello, World!</p>"
```

- `@app.route("/")` is Python decorator: extends the behavior of a function
  - Tells Flask what URL should trigger our function
  - The `"/"` is the root of our web site domain  
`http://127.0.0.1:5000` takes us to this route
  - We can have other routes: `@app.route("/register")`  
`http://127.0.0.1:5000/register` takes us to this other route

## What else to learn:

- Variable rules:  
<https://flask.palletsprojects.com/en/stable/quickstart/#variable-rules>
- HTTP methods:  
<https://flask.palletsprojects.com/en/stable/quickstart/#http-methods>



# Templates

File ./app.py:

```
1 from flask import render_template
2
3 @app.route('/hello/')
4 @app.route('/hello/<name>')
5 def hello(name=None):
6     return render_template('hello.html', person=name)
```

File ./templates/hello.html:

```
1 <!doctype html>
2 <title>Hello from Flask</title>
3 {% if person %}
4     <h1>Hello {{ person }}!</h1>
5 {% else %}
6     <h1>Hello, World!</h1>
7 {% endif %}
```

# Templates

File ./app.py:

```
1 from flask import render_template
2
3 @app.route('/hello/')
4 @app.route('/hello/<name>')
5 def hello(name=None):
6     return render_template('hello.html', person=name)
```

File ./templates/hello.html:

```
1 <!doctype html>
2 <title>Hello from Flask</title>
3 {% if person %}
4     <h1>Hello {{ person }}!</h1>
5 {% else %}
6     <h1>Hello, World!</h1>
7 {% endif %}
```

- Import render\_template (Jinja2 template engine)
- 2 routes for the same function
- The function has a named argument

# Templates

File ./app.py:

```
1 from flask import render_template
2
3 @app.route('/hello/')
4 @app.route('/hello/<name>')
5 def hello(name=None):
6     return render_template('hello.html', person=name)
```

File ./templates/hello.html:

```
1 <!doctype html>
2 <title>Hello from Flask</title>
3 {% if person %}
4     <h1>Hello {{ person }}!</h1>
5 {% else %}
6     <h1>Hello, World!</h1>
7 {% endif %}
```

- Import render\_template (Jinja2 template engine)
- 2 routes for the same function
- The function has a named argument
- render\_template passes the named arguments
- The template has access to the Python variables.  
Inside of {% %} is Jinja2 code  
Inside of {{ }} is to print as HTML

# What else to learn about templates?

- If statements: <https://jinja.palletsprojects.com/en/stable/templates/#if>
- For loops: <https://jinja.palletsprojects.com/en/stable/templates/#for>
- Template Inheritance  
<https://jinja.palletsprojects.com/en/stable/templates/#template-inheritance>

## MVC (extra)

# Model View Controller

# Model View Controller

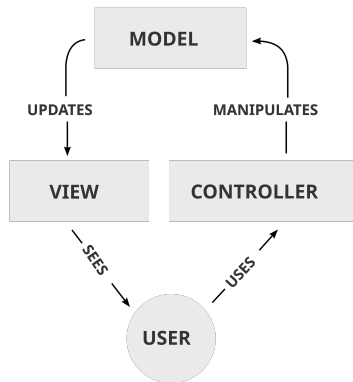
- Model View Controller (MVC) is a software architectural pattern

# Model View Controller

- Model View Controller (MVC) is a software architectural pattern

## We split the applications in:

- Model: The internal representation of the information (e.g., our domain objects)
  - Interact with the database
  - Business logic
  - Do complex computations
- View: The interface for the user
  - Builds the HTML/Javascript/JSON/Text
- Controller: Links model and view





## Docker (extra)

# Docker Introduction

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)
- I use Linux, you may use Windows, your next group colleague may use MacOS

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)
- I use Linux, you may use Windows, your next group colleague may use MacOS
- How do we make sure everybody has the exact same environment?

# Docker Introduction

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)
- I use Linux, you may use Windows, your next group colleague may use MacOS
- How do we make sure everybody has the exact same environment?

## Solution

- We enforce the same environment in declarative manner:  
version of system libraries, databases, the environment variables...
- Docker does exactly that!

# Docker Introduction

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)
- I use Linux, you may use Windows, your next group colleague may use MacOS
- How do we make sure everybody has the exact same environment?

## Solution

- We enforce the same environment in declarative manner:  
version of system libraries, databases, the environment variables...
- Docker does exactly that!

## What is Docker?

# Docker Introduction

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)
- I use Linux, you may use Windows, your next group colleague may use MacOS
- How do we make sure everybody has the exact same environment?

## Solution

- We enforce the same environment in declarative manner:  
version of system libraries, databases, the environment variables...
- Docker does exactly that!

## What is Docker?

- What Docker is not: A virtual machine



# Docker Introduction

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)
- I use Linux, you may use Windows, your next group colleague may use MacOS
- How do we make sure everybody has the exact same environment?

## Solution

- We enforce the same environment in declarative manner:  
version of system libraries, databases, the environment variables...
- Docker does exactly that!

## What is Docker?

- What Docker is not: A virtual machine
- It is a container solution based on a Linux kernel feature called cgroups
- It is also runs on MacOS and Windows

# Docker Introduction

## The problem

- Software depends on specific versions of system libraries, and in the computer environment in general (e.g. directory structures, databases, ambient variables, etc)
- I use Linux, you may use Windows, your next group colleague may use MacOS
- How do we make sure everybody has the exact same environment?

## Solution

- We enforce the same environment in declarative manner:  
version of system libraries, databases, the environment variables...
- Docker does exactly that!

## What is Docker?

- What Docker is not: A virtual machine
- It is a container solution based on a Linux kernel feature called cgroups
- It also runs on MacOS and Windows

## Why use Docker?

- Like git, it is a standard in the industry
- It will make the life of the TAs way easier



## More concepts

- Docker container: isolated process with its own file system
- Docker image: is a package that includes all of the files, binaries, libraries, and configurations to run a Docker container
  - Built on layers: make your own image on top of existing ones
- Docker volume: persistent data stores for containers

## More concepts

- Docker container: isolated process with its own file system
- Docker image: is a package that includes all of the files, binaries, libraries, and configurations to run a Docker container
  - Built on layers: make your own image on top of existing ones
- Docker volume: persistent data stores for containers

## An usual setup:

- Dockerfile: The Docker image in which your main app will run on
- `entrypoint.sh`: The script that the Dockerfile calls when its container is starting up
- `docker-compose.yml`: Organizes all the different Docker containers of your application

## A Minimal MVC Flask app with Docker Project

# A Minimal MVC Flask app with Docker Project

- App

Let's write our web apps!



- Follow my step-by-step guide: <https://github.com/rafaelcgs10/dis2025>
- Follow the official Flask tutorial:  
<https://flask.palletsprojects.com/en/stable/tutorial/>