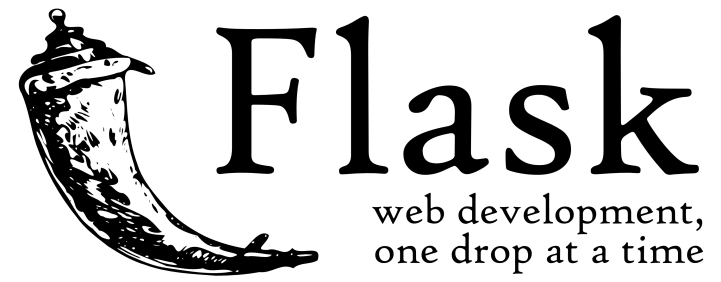
**An Introduction to Web Applications and How to deploy To the Cloud  
- *Based on what I learned so far***

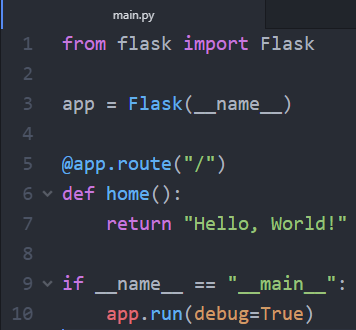
By Salvador Villalon

**Before starting to write code and using Flask. Let’s go to some basics**

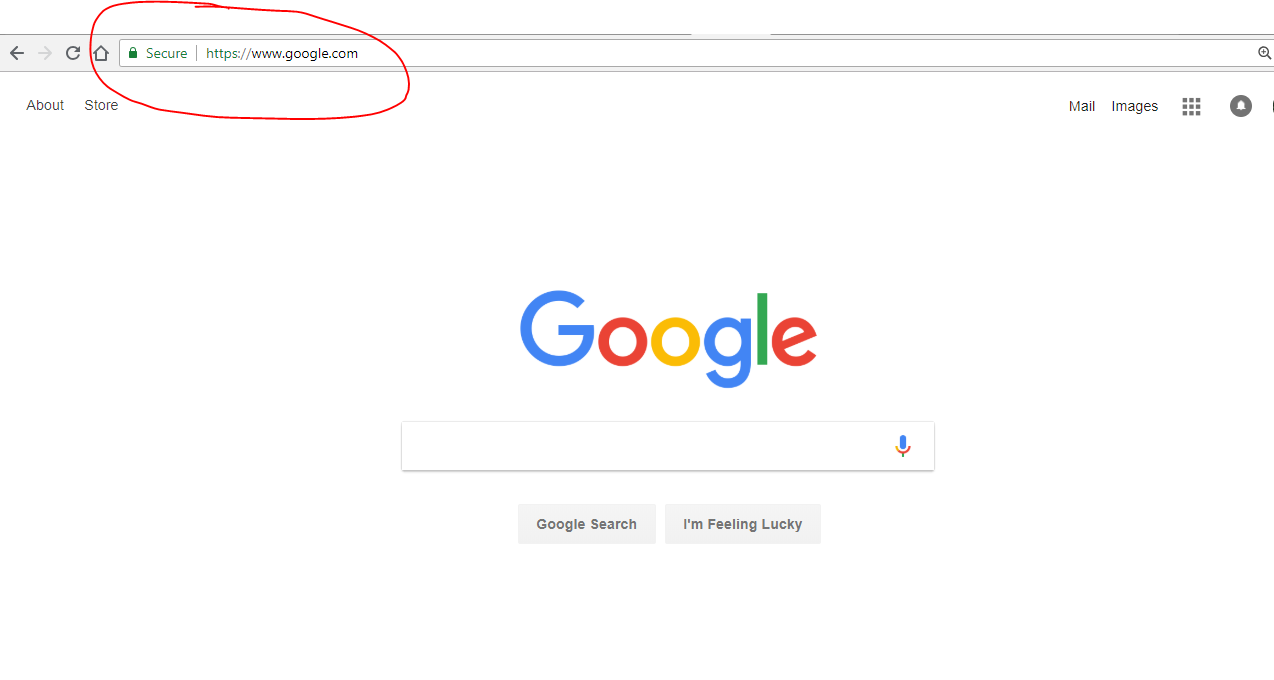
* What is HTTP and What does it have to do with Flask?
  + [HTTP stands for Hyper Text Transfer Protocol](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol)
    - This is the protocol websites and the internet use to interact and communicate with computers and servers
  + Think about it this way
    - When you are typing the name of a website in the address bar of your browser, then you hit enter. What that is really doing it is sending an HTTP Request to some server
    - For example, I go to my address bar and type google.com, then I hit enter
    - What will happen is that:
      * A HTTP Request is send to Google Server
      * Google Server receives the request and needs to figure out some wy to interpret that request
      * Google Server sends back an HTTP Response that contains the information that my web browser then receives.
      * Then it displays it on a page in my browser either Chrome, Edge, or any browser you are using.
* What does this have to do with Flask?
  + We will write code that will take care of the server side processing
    - Our code will receive requests
    - It will figure out what those requests are dealing with and what they are asking for
    - It will figure out what response to send to the user
  + ***To do all of this we will use Flask***
* **What is Flask?**



* + [Flask is a micro framework](http://flask.pocoo.org/) written in Python
  + It make process of designing a Web Application really simple
  + ***Flask lets us focus on what the users are requesting and what sort of response to give back***
* **How Does a Flask App Work?**
  + The Code that lets us run a basic web application that we can serve as if it were a website

******

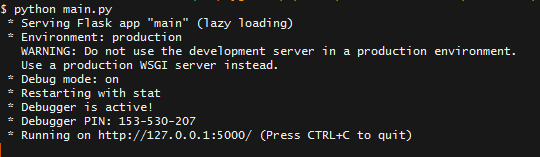
* + **Line 1: Here we are importing the Flask module. We are creating a Flask web server from this Flask module**
    - *Note: Someone else wrote the Flask module*
    - Main idea from this line is that we are saying that in this file we want to use the Flask Module
  + **Line 2:** 
    - \_\_name\_\_ means this current file. In this case, it will be main.py. This current file will represent my Web Application
    - We are creating an instance of the Flask class and calling it app . Here we are creating a new Web Application
  + **Line 5:**
    - Point to Note: Flask is designed in terms of routes where the route is part of the URL you type in order to determine which page you want to request
    - In line 5, it just represents the default page
      * For example, if I go to a website such as “google.com/” with nothing after the slash, then this will be default page of google

****

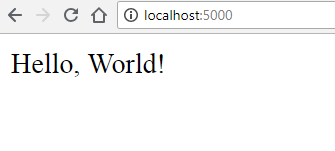
* + **Line 6 – 7:**
    - When the users go to my website and they go to the default page (nothing after the slash), then the function below it will get activated
  + **Line 9:** 
    - When working with Python, it is important to know that when you run your python script Python assigns the name “\_\_main\_\_” to the script when it is executed.
    - If we import another script, this if statement will prevent other scripts from running since when we run main.py, main.py will change its name to \_\_main\_\_ and only then will that if statement be activated
  + **Line 10:** 
    - app.run(debug=True)
    - This will run the application
    - Having debug=True this will allow possible Python error to appear on the web page. This will help us trace the errors

**Let’s Try Running the main.py**

* + **In your Terminal or Command Prompt go to the folder that contains your main.py**
  + Then do **py main.py** or **python main.py**
  + In your terminal or Command Prompt you should see this output

****

* + The important part is where it says “Running on <http://127.0.0.1:5000/>”
    - 127.0.0.1 means this local computer. If you do not know the meaning of this (just like I did when I started [this article is really helpful](https://whatismyipaddress.com/localhost))
  + Go to that address and you should see this



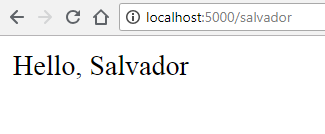
* + Congrats you just made your first Web Application!

**More Fun With Flask**

* + Earlier you just saw what happened when we ran main.py with just one route which was app.route(“/”)
  + Let’s add more routes so you can see the difference



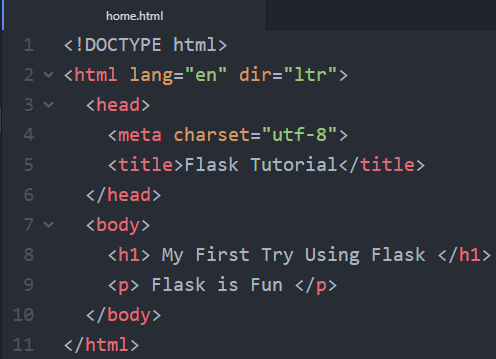
* + In lines 9 – 11. We added a new route, this time to **/salvador**
  + Now run the main.py again and go to[http://127.0.0.1:5000/salvador](http://localhost:5000/salvador)



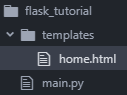
* + **You can add as many routes as you want.**
  + **But this is just returning text, let’s make our website look nicer**

**HTML and Templates in Flask**

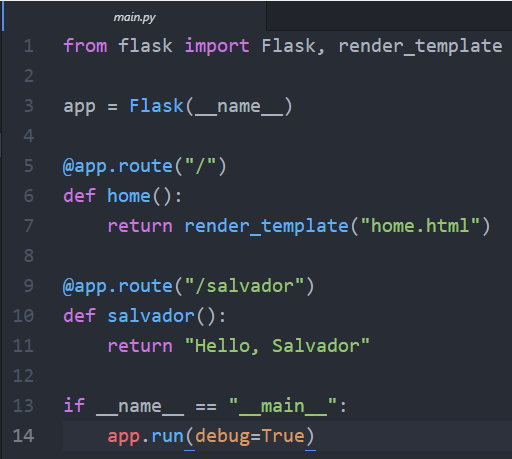
* First create a new HTML file. I called mine home.html
* Here is some code to get you started

****

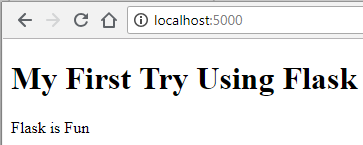
* **Important Point To Remember** 
  + The Flask Framework looks for HTML files in a folder called **templates**
  + You **need to create a templates** folder and put all your html files in there

****

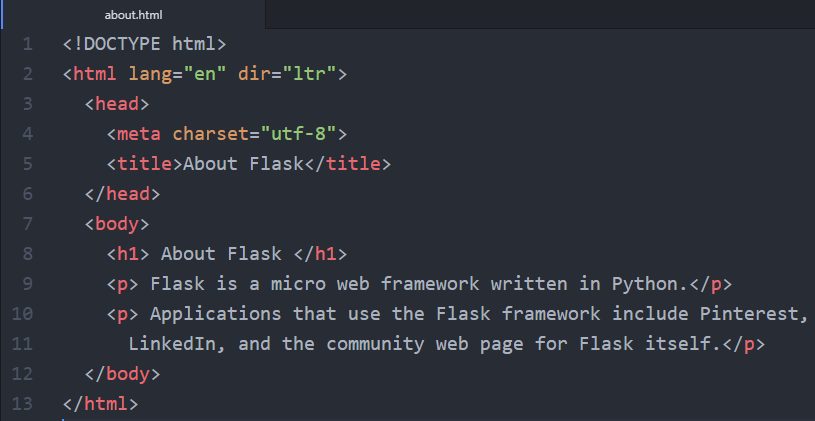
* + Remember to always keep the main.py outside of your templates folder
* **Now Bring it Back to main.py**
  + Now we need to change our main.py so that we can view the html file we created



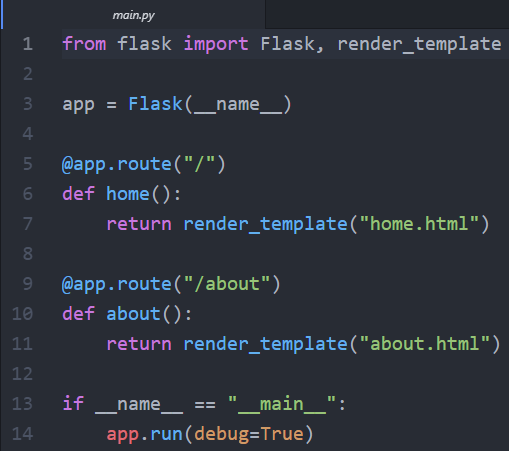
* + We made two new changes
    - **Line 1:** We imported render\_template method from the flask framework. Render\_template looks for a template in the templates folder and it will render the template you ask for
      * **More information here:** [**http://flask.pocoo.org/docs/0.12/quickstart/#rendering-templates**](http://flask.pocoo.org/docs/0.12/quickstart/#rendering-templates)
    - **Line 7:** We change the return so that now it returns render\_template(“home.html”). This will let us view our html file.
  + **Now visit and see the changes: [http://127.0.0.1:5000/](http://localhost:5000/)**

****

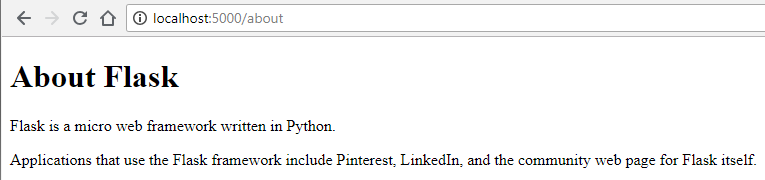
* **Let’s add more pages**
  + Let’s create an **about.html** inside the **templates folder.**
  + **This is the code**

****

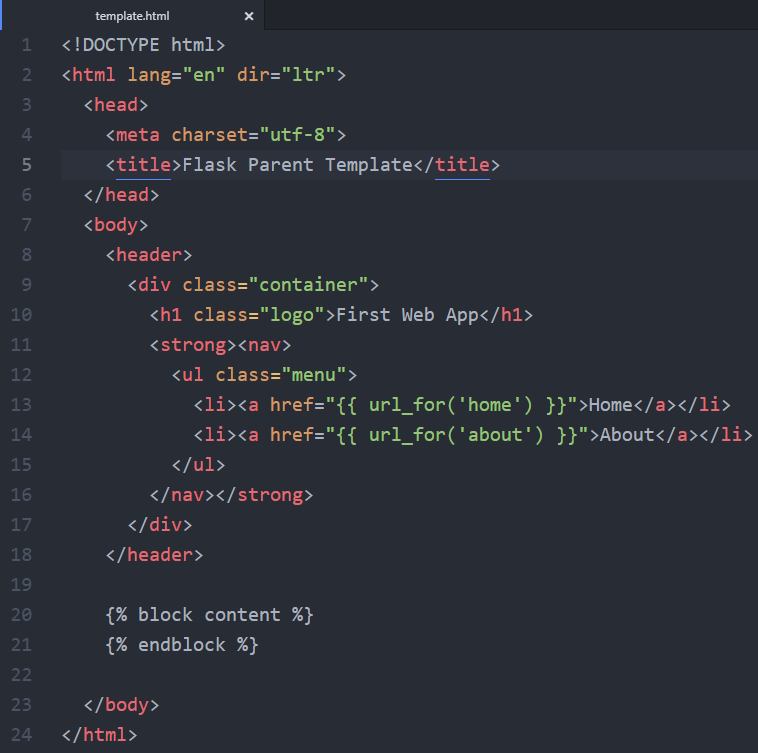
* **Now Bring it Back to main.py**
  + Let’s make a similar change from before to our main.py



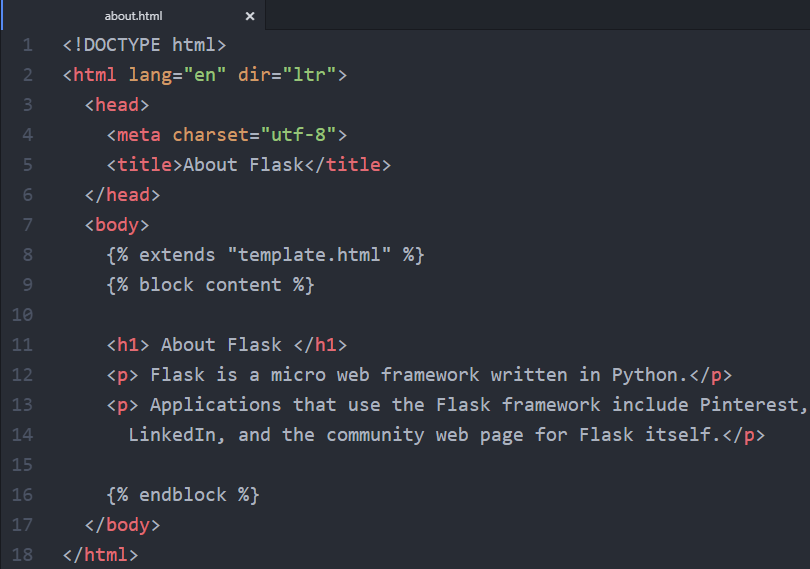
* + We made two new changes
    - **Line 9:** Change the route to “**/about”**
    - **Line 11:** We change the return so that now it returns render\_template(“about.html”)
  + **Now visit and see the changes:** [**http://127.0.0.1:5000/about**](http://127.0.0.1:5000/about)

****

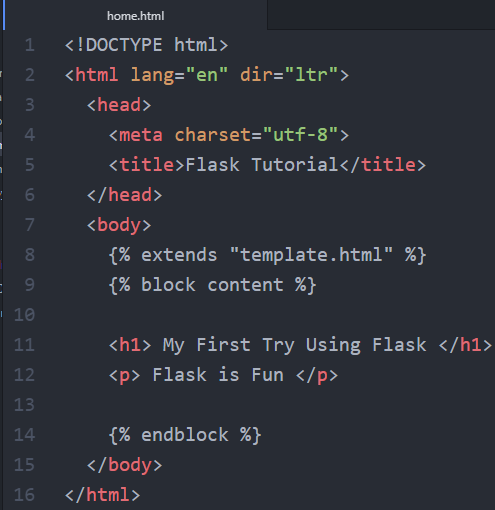
* **Let’s Connect Both Pages with a Navigation**
  + To connect both pages we can have a navigation menu on the top. We can use Flask to make the process of creating a navigation menu easier
  + First, let’s create a **layout.html.** This **layout.html** will server as a parent template that our two child templates can inherit code from.



* + **What is going on here?**
    - **Line 13 – 14:** we use the function called **url\_for().** It accepts the name of the **function as first argument** and a number of keyword arguments, each corresponding to the variable part of the URL rule. **Right now we just gave it the name of the function.**
      * **More info:** [**http://flask.pocoo.org/docs/0.12/quickstart/#url-building**](http://flask.pocoo.org/docs/0.12/quickstart/%23url-building)
    - The two lines with the curly brackets will be **replaced by the content of home.html and about.html** depending on the URL the user is browsing in.
  + We these changes it allows the child pages (home.html and about.html) to connect to the parent (template.html)
  + **about.html**



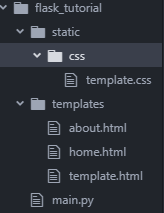
* + - **home.html**

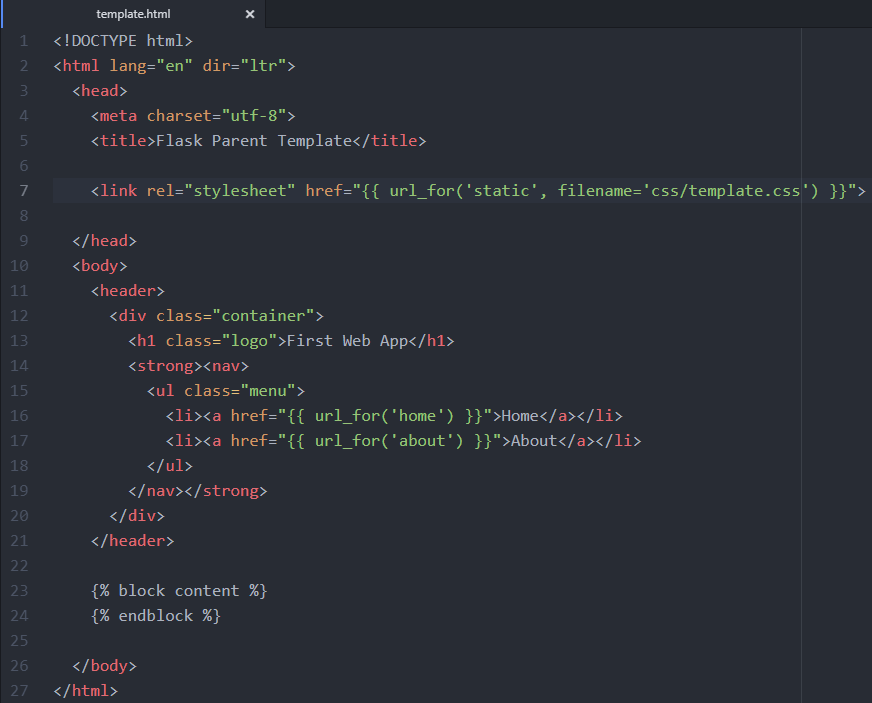
****

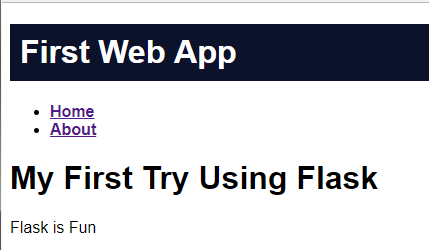
* Our website is great, but now let’s try adding some CSS

**Adding CSS to Our Website**

* + **An important note to remember**
    - Just like we did by creating a folder called **templates** to store all of our HTML templates. We need a folder called **static** to store our stylesheets
    - In **static**, we will store our CSS, JavaScript, images, and other necessary files.
    - That is why it is important that you should create **css folder to store your CSS stylesheets**
  + After you do this your project folder should look like this

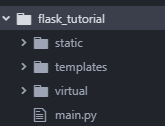


* + **Linking our CSS with our HTML file**
    - Since our template.html is the one that links all pages. We can insert the code here and it will be applied on the child pages. 
    - **Line 7:** Here we are giving the path to where the template.css is located
    - **Now visit and see the changes:** [**http://127.0.0.1:5000/**](http://127.0.0.1:5000/)

****

**Moving Forward with Flask and virtualenv**

* + Now that you are more familiar with how to use Flask and you may start using it in your future projects you must always use virtualenv.
  + Why use virtualenv?
    - If you think about it, you may use Python for others projects besides web-development
    - The more projects you work on it might mean that you will have different version of Python installed, different dependencies or different versions of libraries installed
    - We use virtualenv to create an isolated environment for your Python project. This means that each project can have its own dependencies regardless of what dependencies every other project has.
  + **Start with virtualenv**
    - First, do **pip install virtualenv** on your Command prompt or Terminal
    - Second, do virtualenv “name of virtual environment”
      * Here you can give a name the environment
      * I usually give it a name of virtual: virtualenv virtual
    - When you finished setting up the virtual environment, check your project folder and it should look like this. **The virtual environment should be created in the same directory where your app files are located**



* + **Activate the Virtual Environment**
    - For OS X and Linux Environment
      * **$ . venv/bin/activate**
    - For Windows Environment
      * **$ venv\Scripts\activate**
    - Since I am using Windows Machine, when I activate the environment it will look like this

****

* + - Next step will be to install flask on your virtual environment so that we can run the application inside our environment. Do **pip install flask**
  + **For more information about flask and virtualenv:** 
    - <http://flask.pocoo.org/docs/0.12/installation/#installation>
    - <https://realpython.com/python-virtual-environments-a-primer/>

**Deploy Your Web Application to the Cloud**

* To deploy our Web Application to the Cloud we will use [Google App Engine](https://cloud.google.com/appengine/) (Standard Environment)
* What do we mean by deploying to the cloud?
  + Here we will be **Google App Engine** which is an example of **Platform as a Service (PaaS)**
  + **PaaS** refers to the **delivery of operating systems and associated services over the internet without downloads or installation**. The approach lets customers create and deploy applications without having to invest in the underlying infrastructure. (TechTarget)
  + **More info:** [**https://searchitchannel.techtarget.com/definition/cloud-services**](https://searchitchannel.techtarget.com/definition/cloud-services)
* Google App Engine is platform as a service offering that allows developers and businesses to build and run applications using Google's advanced infrastructure (TechOpedia)

***Resources***

* <https://pythonhow.com/building-a-website-with-python-flask/>
* [https://cloud.google.com/appengine/docs/standard/python/getting-started/python-standard-env](https://cloud.google.com/appengine/docs/standard/python/getting-started/python-standard-env#test_the_application)
* <https://youtu.be/j5wysXqaIV8>
* <https://realpython.com/python-virtual-environments-a-primer/>
* <http://flask.pocoo.org/docs/0.12/installation/#installation>
* [***https://www.techopedia.com/definition/31267/google-app-engine-gae***](https://www.techopedia.com/definition/31267/google-app-engine-gae)
* [***https://cloud.google.com/appengine/***](https://cloud.google.com/appengine/)
* [***https://searchitchannel.techtarget.com/definition/cloud-services***](https://searchitchannel.techtarget.com/definition/cloud-services)
* ***https://www.techopedia.com/definition/31267/google-app-engine-gae***