PCF

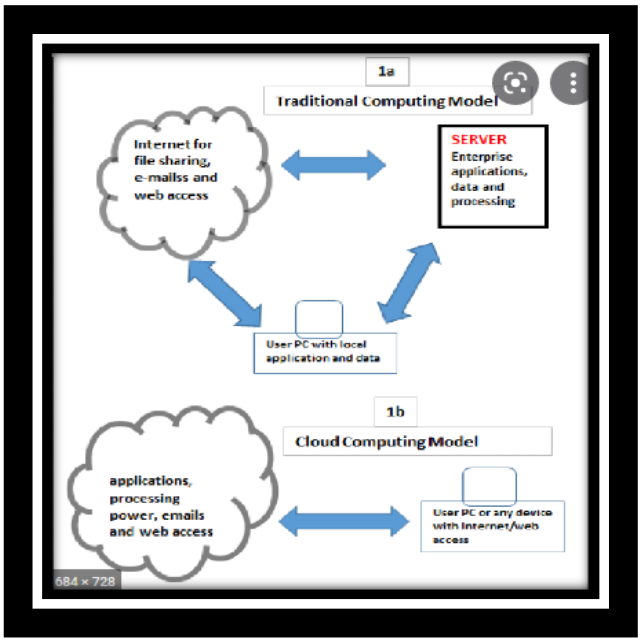
PIVOTAL CLOUD FOUNDRY

Pivotal Cloud Foundry is now VMware Tanzu Application Service.

Developed by **the Social Work Reform Board**

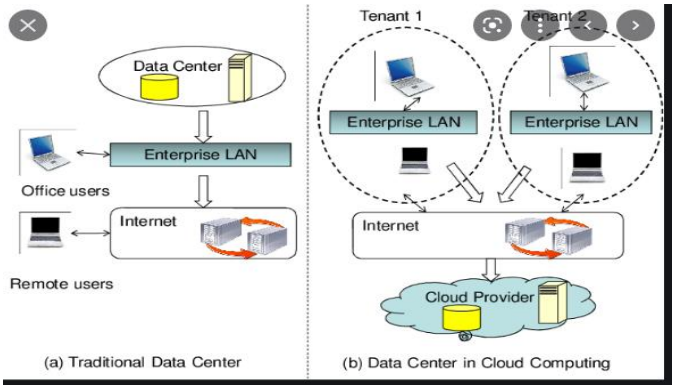
PCF is one example of an PaaS.

PCF is **a cloud native platform for deploying next-generation apps**. open source technology.



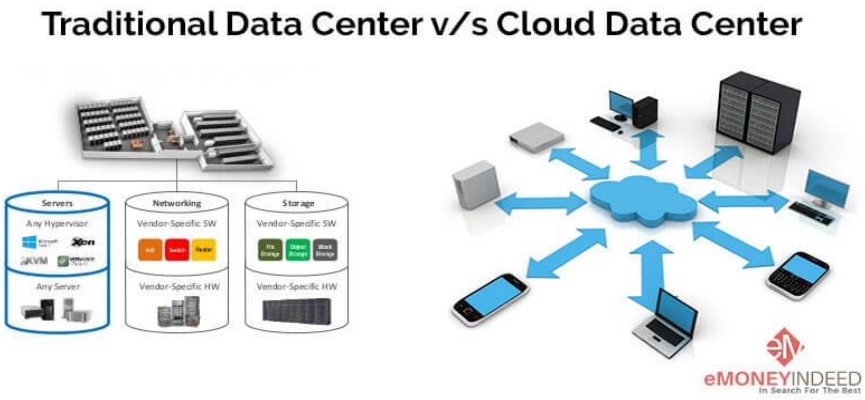
Working:

on –premises physically accessible,remote access.



Management:

Internal business responsibility,outsourced to third party providers.

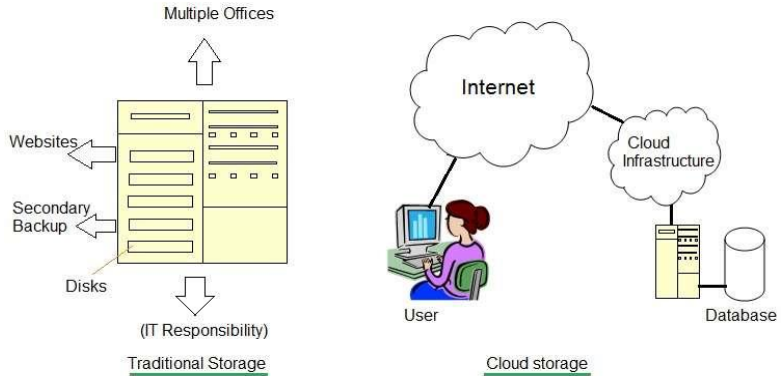


Administration:

In house IT professionals used,cloud can access at any Remote area.

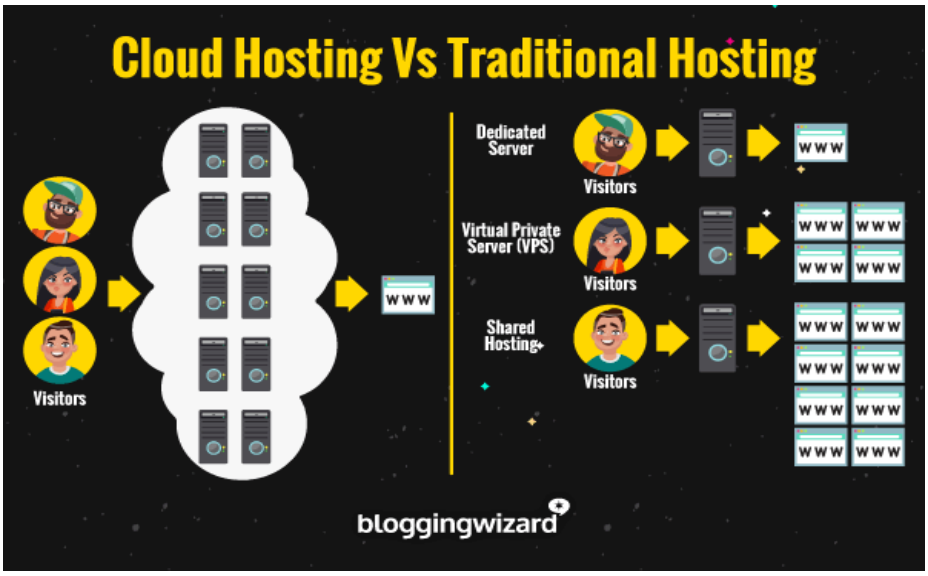
Usage:

Lot of intermediate physical access,is a direct access.



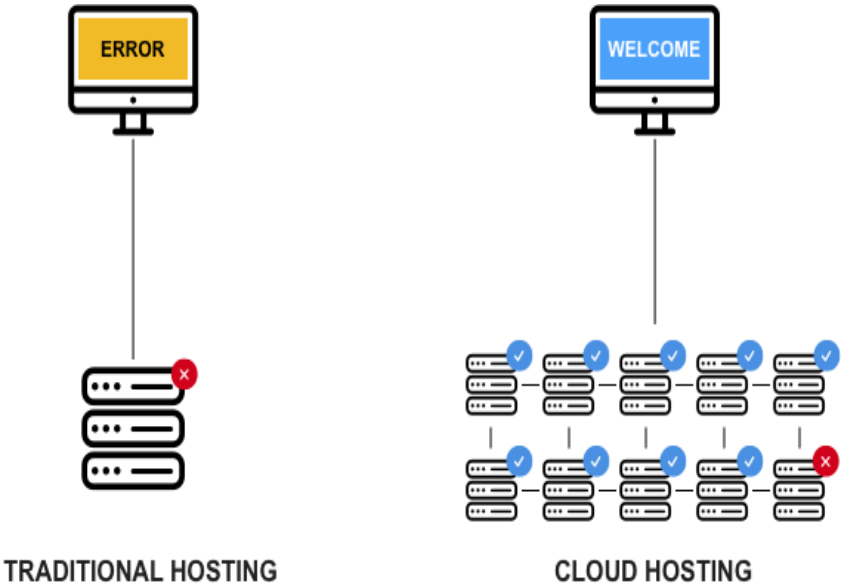
Reliability:

High availability,no down time



Load balancing:

Single point of failure will lead to error,in cf automatically work progress.



PURPOSE OF PCF

The main purpose of PCF is **to provide the underlying infrastructure and environments** that organizations need to facilitate continuous delivery of

software updates, manage the application life cycle and streamline the development, deployment, and scaling of web-based application.

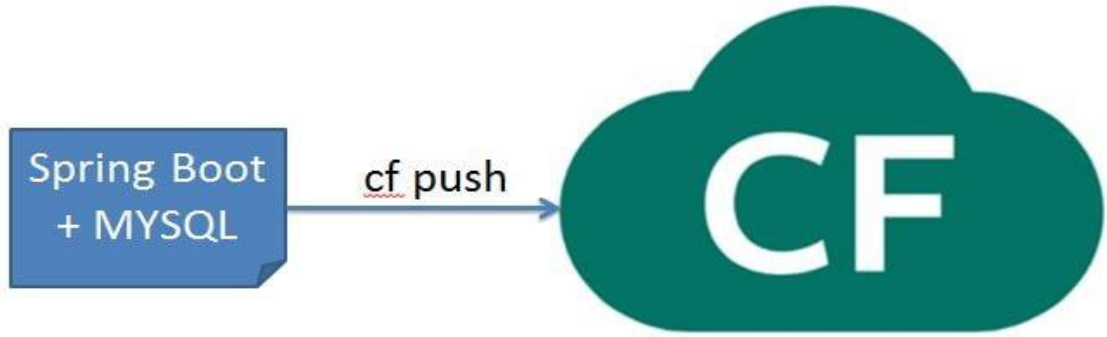
Deploy application in pcf using 2 ways

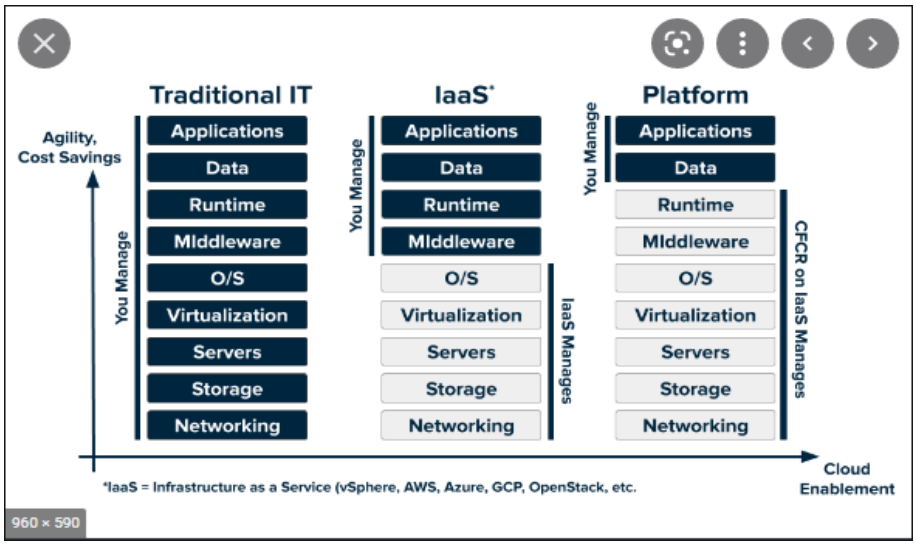
1. Manual

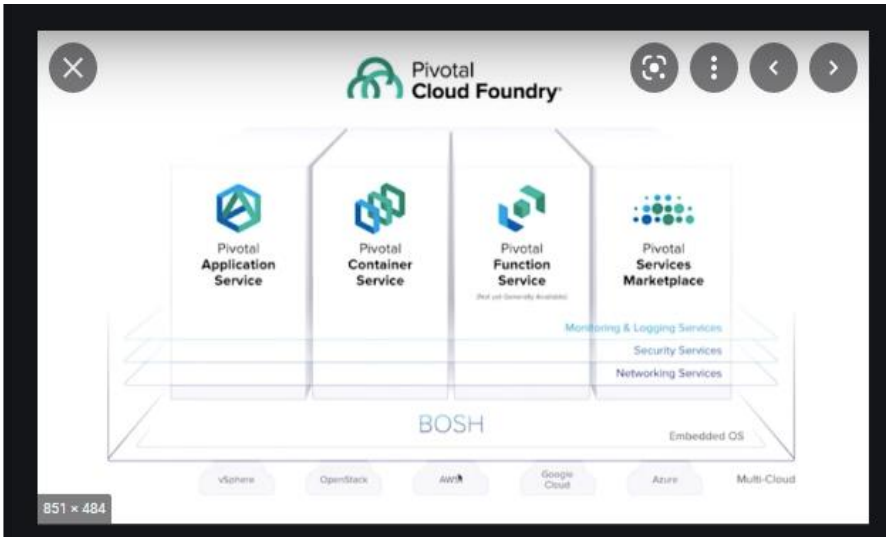
   From local source code-pcf

2. Pipeline

   Using pipeline push our code-pcf







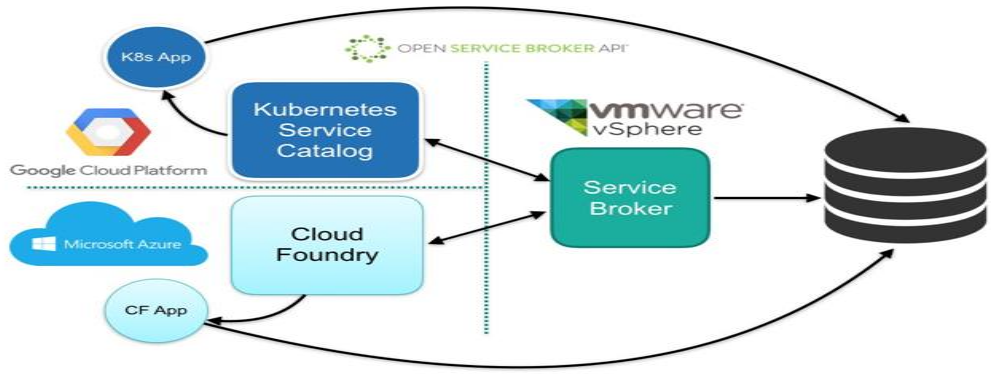
The primary PCF architecture components are:

- BOSH
- Loggregator
- Diego
- Cloud Controller
- Gorouter
- Ops Manager
- User Account and Authentication server (UAA)

BOSH

BOSH is an open source tool that enables deployment and lifecycle management of distributed systems.

.



Loggregator

Loggregator Agent: Loggregator Agents run on both component VMs and Diego Cell VMs. They **receive logs and metrics from** the Forwarder Agents, and then forward the logs and metrics from multiple Dopplers.

Diego cell

Diego is **a self-healing container management system** that attempts to keep the correct number of instances running in Diego Cells to avoid network failures and crashes.

Cloud controller: A cloud controller is **a storage appliance that automatically moves data from on-premises storage to cloud storage**.

Gorouter:

Gorouter Routes **HTTP traffic coming into Cloud Foundry to the appropriate component**. Receives route updates through NATS(Neural **Autonomic Transport System)**.

OPS manager:

Ops Manager is **the dashboard for administering the runtimes and services within your PCF deployment**

User account and authentication server:

User Account and Authentication (UAA) is an open-source identity server project under the Cloud Foundry Foundation. UAA provides enterprise-scale identity management features.

Steps for manually push our application into pcf

1.In pcf a spacial has been created for our project deployment.

2.An url has been generated.

3.Then an temporary url is generated based on the url password has been generated for authentication.

4.Here org Liberty lab

5.Here space Dev

6. API endpoint: https://api.sys.cde.edc1.cf.abc.com (API version: 3.99.0)

User: maha@abc.com

Org: Liberty-Lab

Space: Dev

7.By using cf push java app c:/user/mahalaksghmim/javadev7/

Cf commands

1.cf login-a –login in the cloud foundry with authentication details.

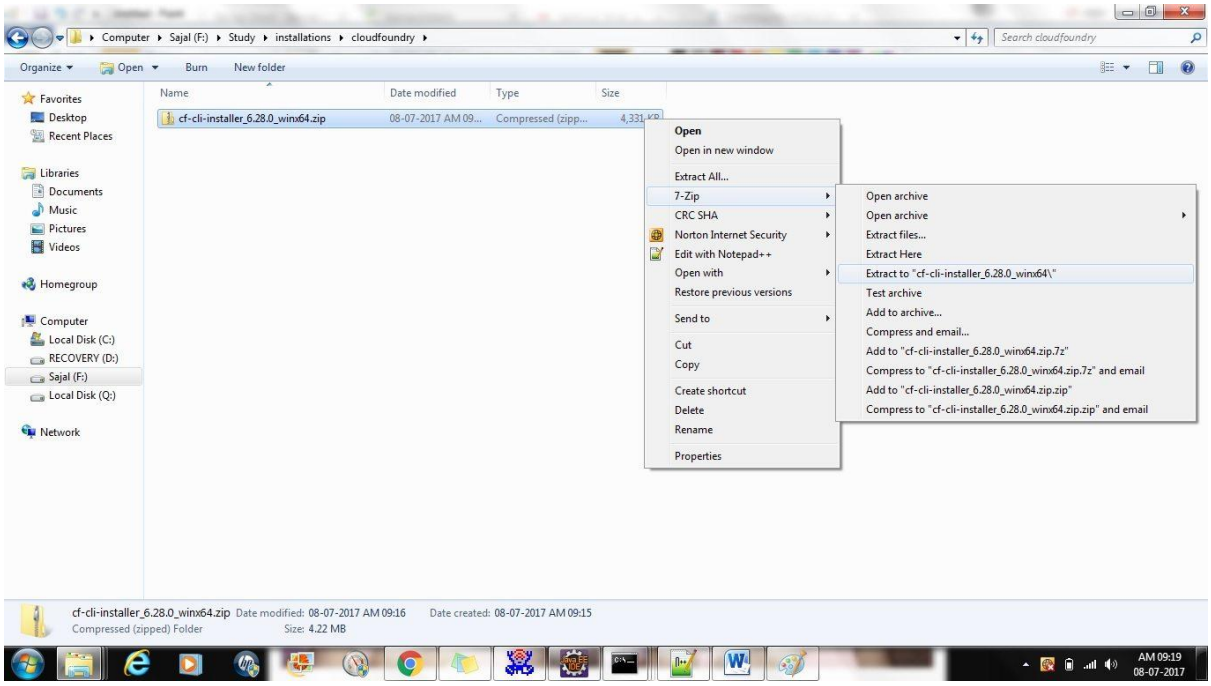
2.cf login –sso(single sign out) enables Cloud Foundry users to authenticate with third-party service dashboards using their Cloud Foundry credentials

3.cf push –p c:\users\mahalakshmim\downloads\java –dev files.

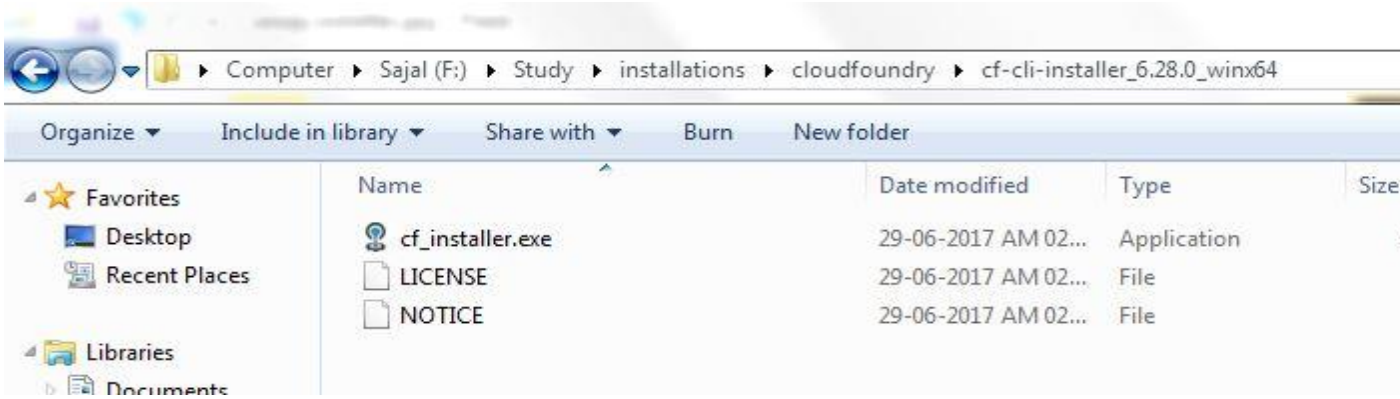
Steps with screenshots how it works

1. Download the CF Windows installer. It will prompt for the download. Save the zip file distribution.

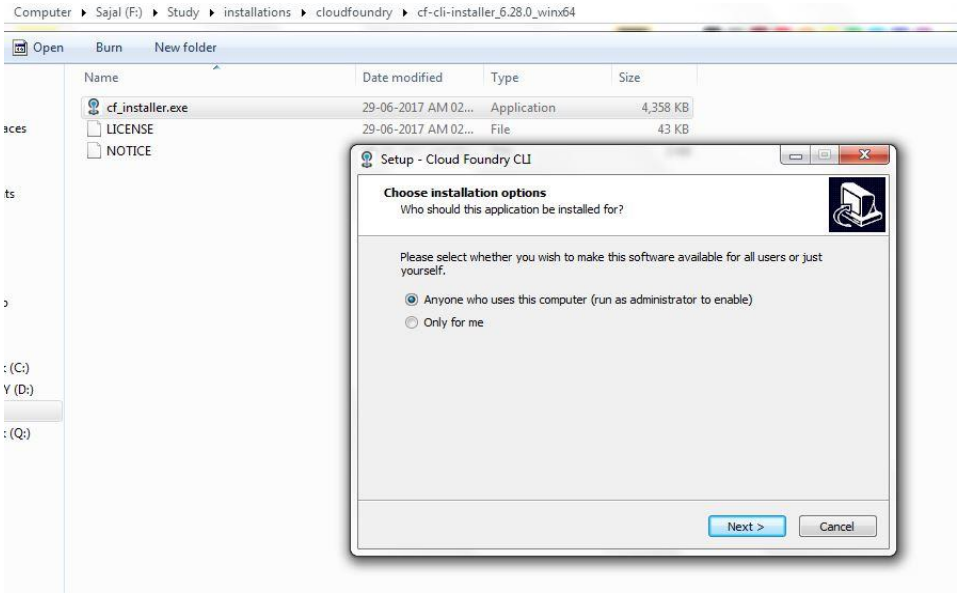
2. Unpack the zip file to a suitable place in your workstation.

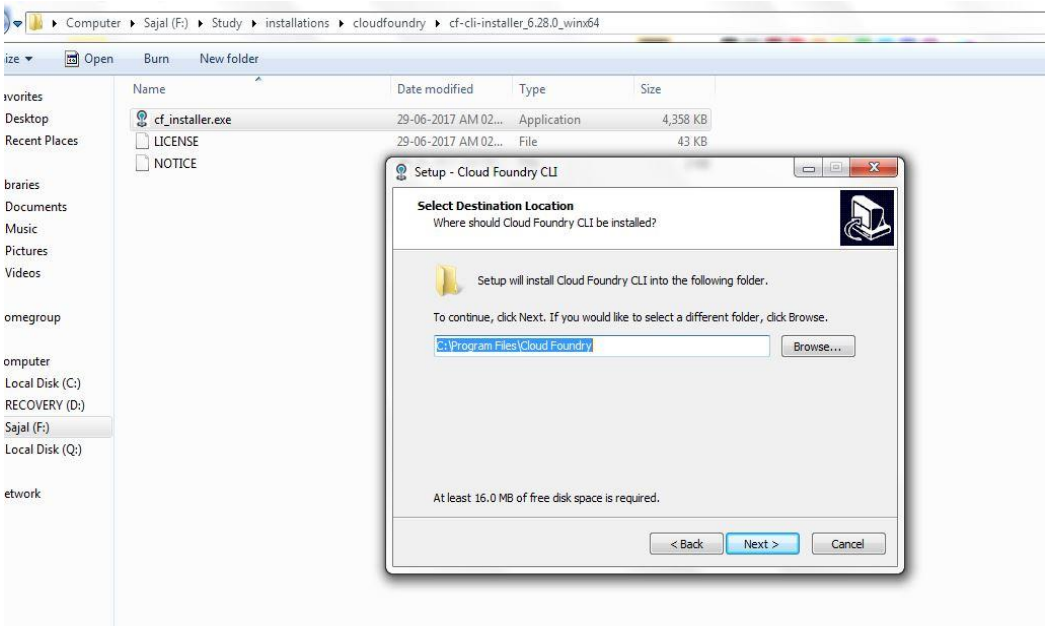


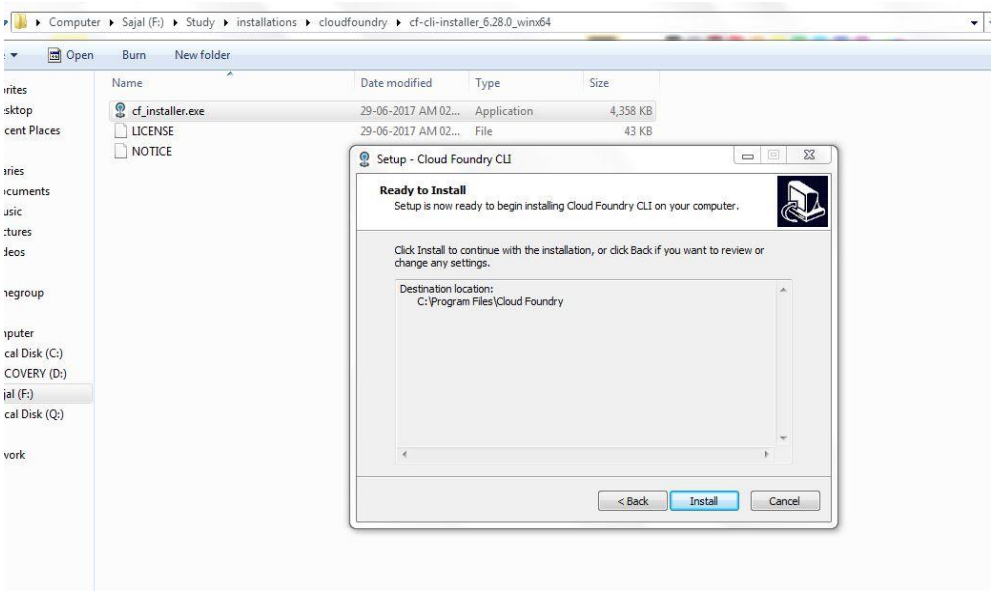
3.After successfully **unzip** operation, double click on the cf CLI executable.

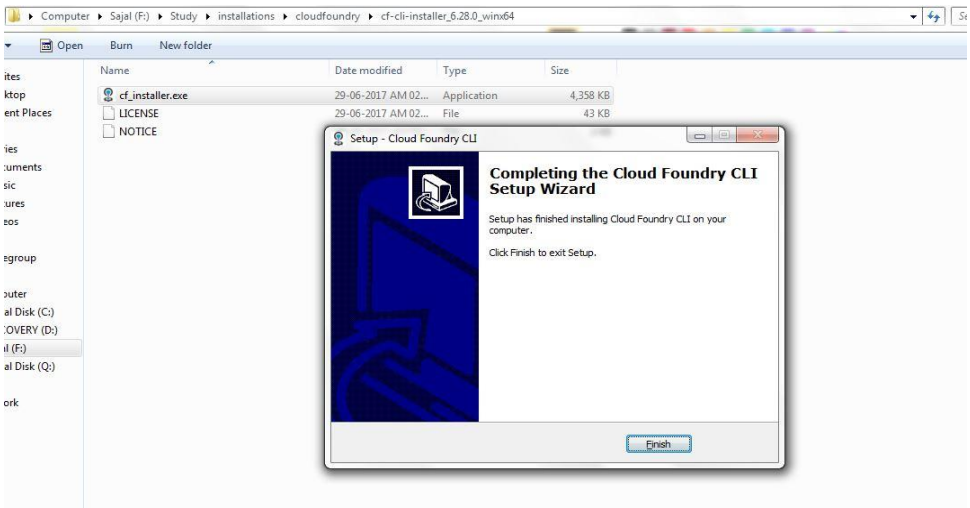


4.When prompted, click **Install**, then Close. Here are the sample steps for the same. This is very straight forward, you can select the default values.

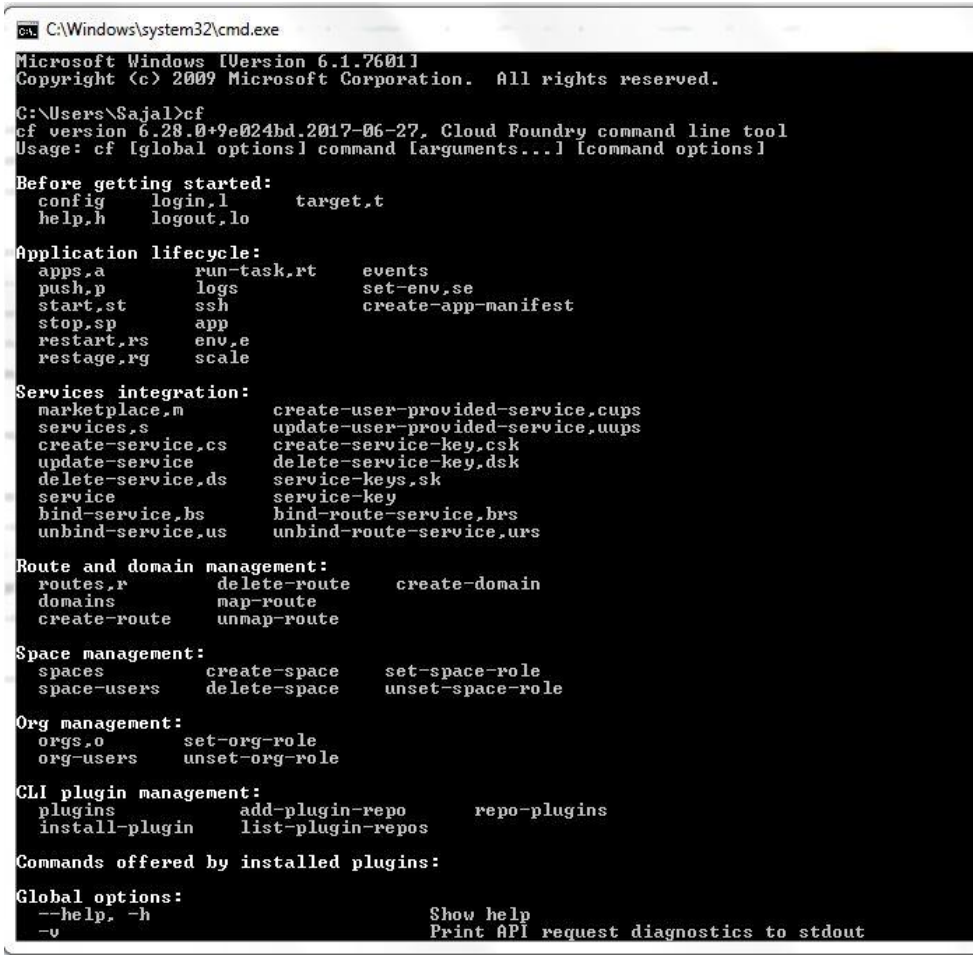




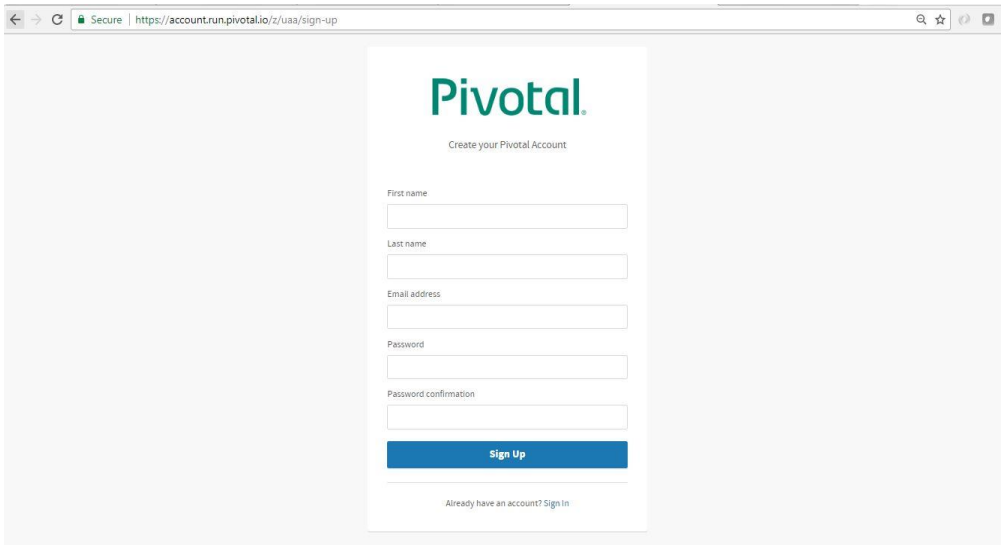




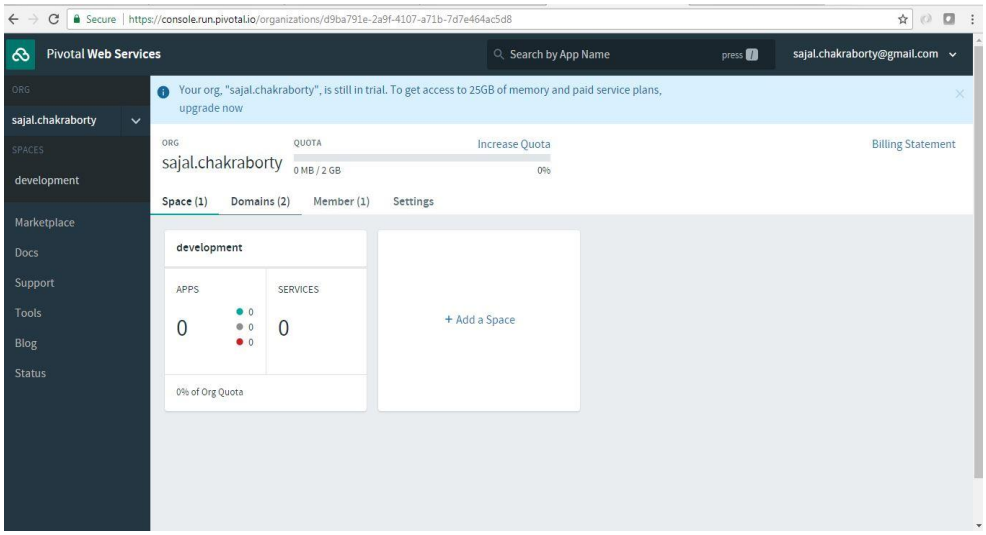
5.Verify the installation by opening a terminal window and type cf. If your installation was successful, the cf CLI help listing appears. This indicates that you are ready to go with any cloud foundry platform from your local workstation..



We will now proceed with *Pivotal Web service account sign up* and development of a sample application and push to cloud foundry.



Here we need to [add org and space] etc.



Login and logout from PWS Console using CLI

1.Login to PWS – We will use cf login -a api.run.pivotal.io command to login to pivotal web service console from CLI tool that we have installed in our local workstation.

2.Logout from PWS Console – We will use command cf logout to logout from the platform, once we have all the work done for that session.

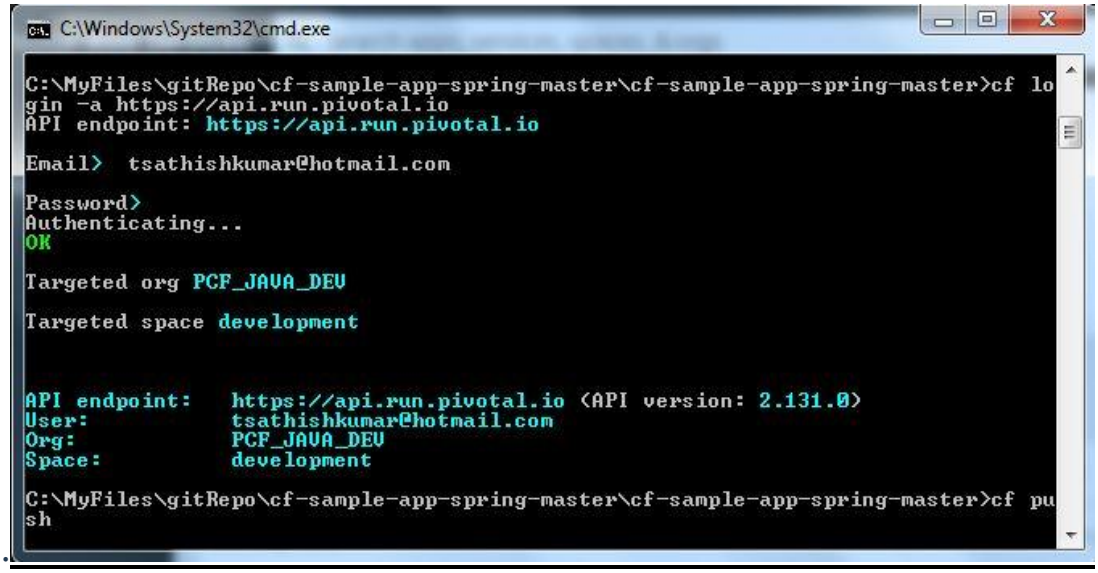
//To login

>> cf login -a api.run.pivotal.io

//To logout

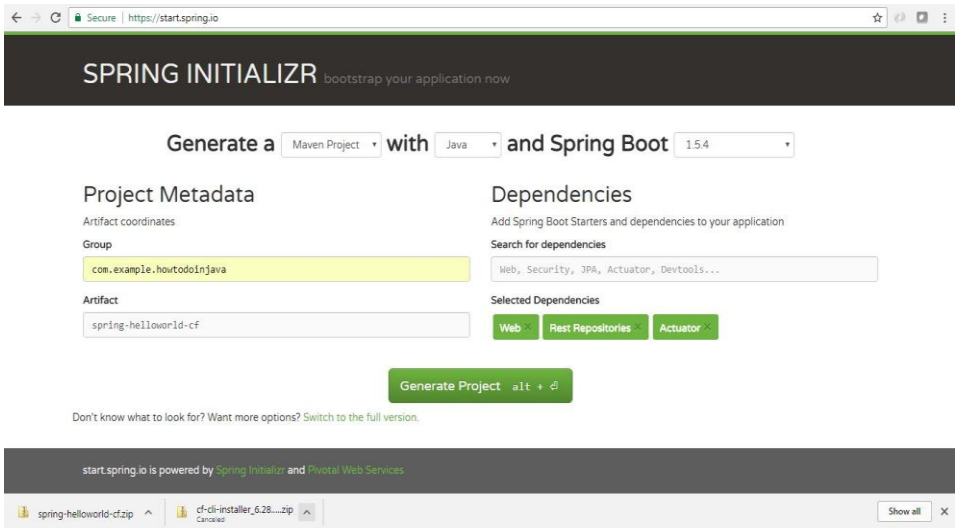
>> cf logout

Here is the login and logout looks like from command prompt.



Generate Spring boot application

Start with spring boot initializer portal



Add REST Controller and Endpoint

```java
package com.example.howtodoinjava.springhelloworldcf;

import java.util.Date;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication
public class SpringHelloworldCfApplication {

    public static void main(String[] args) {
        SpringApplication.run(SpringHelloworldCfApplication.class, args);
    }
}

@RestController
class MessageRestController {

    @RequestMapping("/hello")
    String getMessage(@RequestParam(value = "name") String name) {
        String rsp = "Hi " + name + " : responded on - " + new Date();
        System.out.println(rsp);
        return rsp;
    }
}
```

Project Configuration

Add Context path and required properties in bootstrap.properties file in src\main\resources directory and add two properties there.

server.contextPath = /hello
management.security.enabled = false

Test locally

Deploy Spring Boot Application in Cloud Foundry Platform

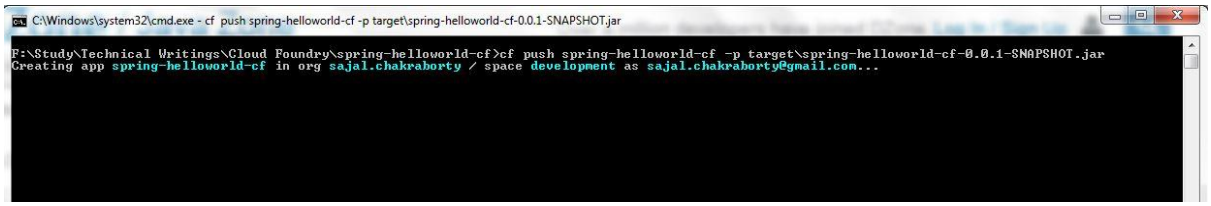
Login to PWS Console

To do that open command prompt and go to maven application's home directory and use cf login -a api.run.pivotal.io command to login to pivotal web service console.

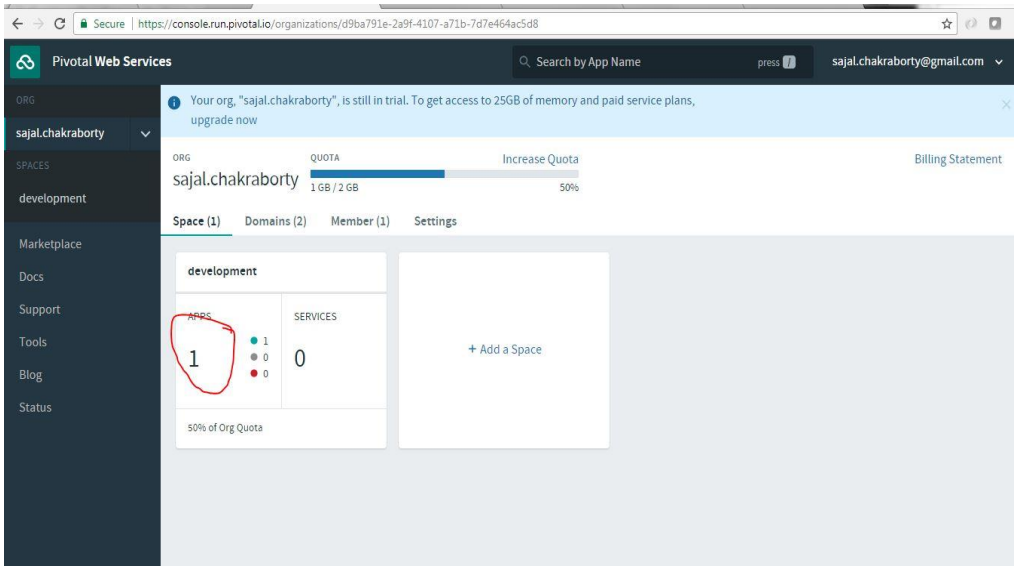
Push Application to Console

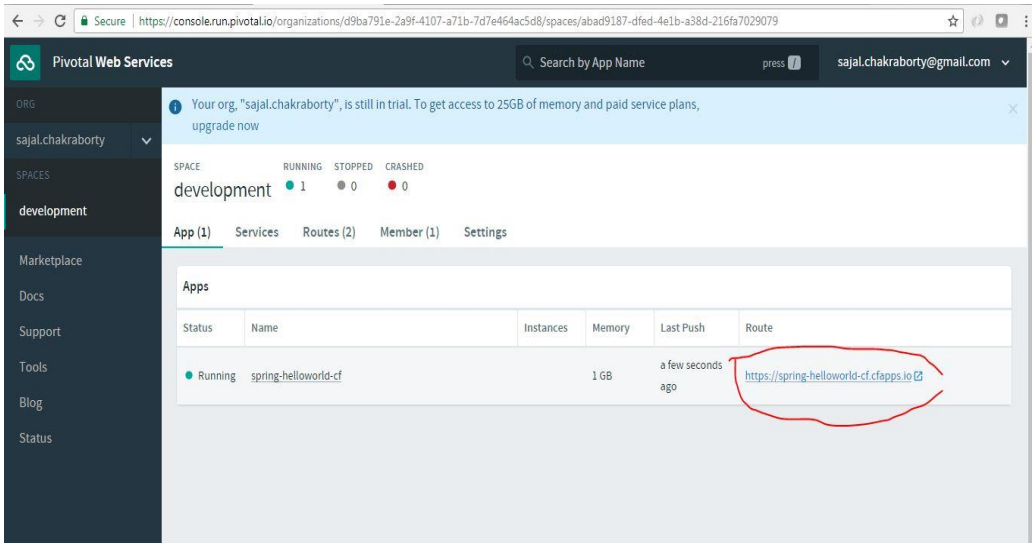
Now we need to push the application with the command cf push.

cf push spring-helloworld-cf -p target\spring-helloworld-cf-0.0.1-SNAPSHOT.jar



Verify Application Deployment





Test REST Endpoint

Now to the browser and access the application with the url host published in the cf console. For this application url is https://spring-helloworld-cf.cfapps.io/hello?name=howtodoinjava.

l

Solace Infotech Pvt.Ltd.

Advantages Of Cloud Computing