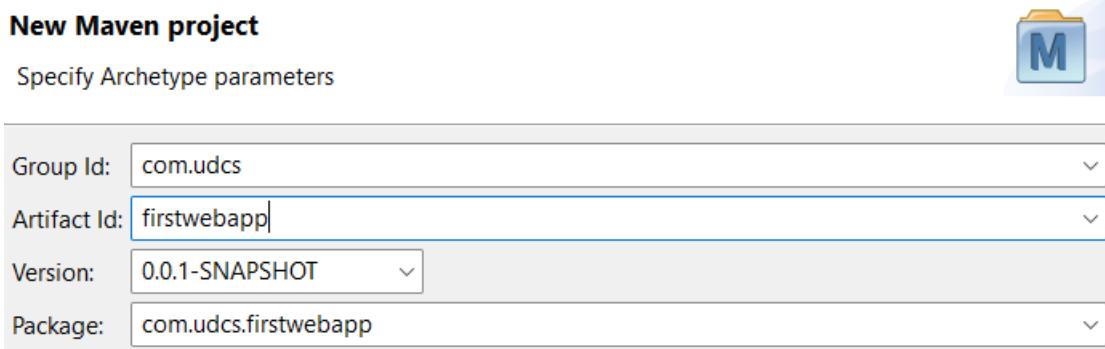


Practical No. 1

Aim: Using the software like JDK 1.8, Eclipse IDE, Apache tomcat server 7.0 Servlets, Spring framework design and develop Web applications using MVC Framework

Implementation:

1. Open Eclipse IDE and create a new Maven project
2. Select the catalog as internal and below that select the last option of the list mentioning ‘web_app’ and click ‘Next’
3. Enter the group_id: com.udcs and artifact_id: firstwebapp



4. Open pom.xml and edit to match the following

```

<project
    xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    http://maven.apache.org/maven-v4_0_0.xsd">

    <modelVersion>4.0.0</modelVersion>
    <groupId>com.gjccs</groupId>
    <artifactId>FirstWebApp</artifactId>
    <packaging>war</packaging>
    <version>0.0.1-SNAPSHOT</version>
    <name>FirstWebApp Maven Webapp</name>
    <properties>
        <jdk.version>1.7</jdk.version>
        <spring.version>3.2.13.RELEASE</spring.version>
        <jstl.version>1.2</jstl.version>
    
```

```

    </properties>

    <dependencies>

        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>${spring.version}</version>
        </dependency>

        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>jstl</artifactId>
            <version>${jstl.version}</version>
        </dependency>

    </dependencies>

    <build>

        <plugins>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.3</version>
                <configuration>
                    <source>${jdk.version}</source>
                    <target>${jdk.version}</target>
                </configuration>
            </plugin>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-war-plugin</artifactId>
                <version>3.3.1</version>
            </plugin>

            <plugin>
                <groupId>org.eclipse.jetty</groupId>

```

```

<artifactId>jetty-maven-plugin</artifactId>
<version>9.2.11.v20150529</version>
<configuration>
    <scanIntervalSeconds>10</scanIntervalSeconds>
    <webApp>
        <contextPath>/spring3</contextPath>
    </webApp>
</configuration>
</plugin>
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-eclipse-plugin</artifactId>
    <version>2.9</version>
    <configuration>
        <downloadSources>true</downloadSources>
        <downloadJavadocs>true</downloadJavadocs>
        <wtpversion>2.0</wtpversion>
        <wtpContextName>spring3</wtpContextName>
    </configuration>
</plugin>
</plugins>
</build>
</project>

```

5. Go to web.xml in src/main/webapp/web-inf and edit it as shown below

```

<web-app
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    version="2.5">
    <display-name>MVC Application</display-name>
    <servlet>
        <servlet-name>spring-web</servlet-name>
        <servletclass>org.springframework.web.servlet.DispatcherServlet
        </servlet-class>
    </servlet>

```

```

<load-on-startup>1</load-on-startup>
<!--
<init-param><param-name>contextConfigLocation</param-name><param-value>/WEB-
INF/spring-mvc-config.xml</param-value></init-param>
-->
</servlet>
<servlet-mapping>
    <servlet-name>spring-web</servlet-name>
    <url-pattern>/</url-pattern>
</servlet-mapping>
</web-app>

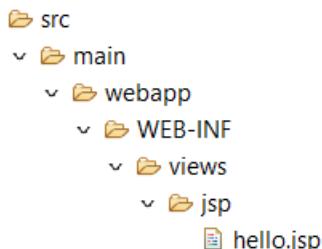
```

6. Right click on web-inf and create XML file named ‘spring-web-servlet.xml’ and edit as follows

```

<beans
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
        http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-3.2.xsd">
    <context:component-scan base-package="com.udcs" />
    <bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix">
            <value>/WEB-INF/views/</value>
        </property>
        <property name="suffix">
            <value>.jsp</value>
        </property>
    </bean>
    <mvc:resources mapping="/resources/**" location="/resources/" />
    <mvc:annotation-driven />
</beans>
```

7. Create new folder in WEB-INF as ‘views and a subfolder as ‘jsp’ and create a ‘hello.jsp’ file in it as follows



8. Right click on src/main/resources in the file explorer and create new folder named ‘java’ but select ‘main’ in the folder creating wizard before that
9. Create a java class in java folder named ‘hellocontroller.java’ as shown below

```
  ✓ src/main/java
    ✓ FirstWebApp
      > HelloController.java
```

10. Edit HelloController.java as follows

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class RegexMatches {
    public static void main(String args[]) {

        // String to be scanned to find the pattern.
        String line = "This order was placed for QT3000! OK?";
        String pattern = "(.*)(\\d+)(.*)";

        // Create a Pattern object
        Pattern r = Pattern.compile(pattern);

        // Now create matcher object.
        Matcher m = r.matcher(line);
        if (m.find()) {
            System.out.println("Found value: " + m.group(0));
            System.out.println("Found value: " + m.group(1));
            System.out.println("Found value: " + m.group(2));
        } else {
            System.out.println("NO MATCH");
        }
    }
}
```

11. Edit the ‘hello.jsp’ file as follows

```
mySelection = app.activeDocument.selection;
myDoc = app.activeDocument;
if (mySelection instanceof Array)
{
    selSwatches = myDoc.swatches.getSelected();

    if(selSwatches.length != 0)
        for (i=0; i<mySelection.length; i++)
    {
        if(mySelection[i].typename == "PathItem" ||
mySelection[i].typename == "CompoundPathItem")
        {
            selItem = mySelection[i];
            selItem.filled = true;
```

```

        swatchIndex = Math.round( Math.random() *
(selSwatches.length - 1 ));

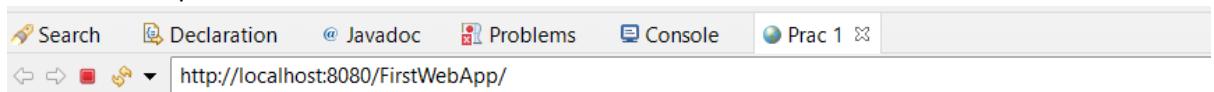
        if(selItem.typename == "PathItem")
            selItem.fillColor = selSwatches[swatchIndex].color;
        else
            selItem.pathItems[0].fillColor =
selSwatches[swatchIndex].color;

    }

}
}

```

12. In IDE, right click on firstwebapp > Maven > update Project > check on ‘force update snapshot’ > Finish
 13. Right click on project > run as > Maven build
 14. In the following pop-up, enter the goals as ‘clean install’
 15. Right click on project > Run as > Run on server > Apache Tomcat v9 > select any file in configured > Finish
- To see the output



Hello World!

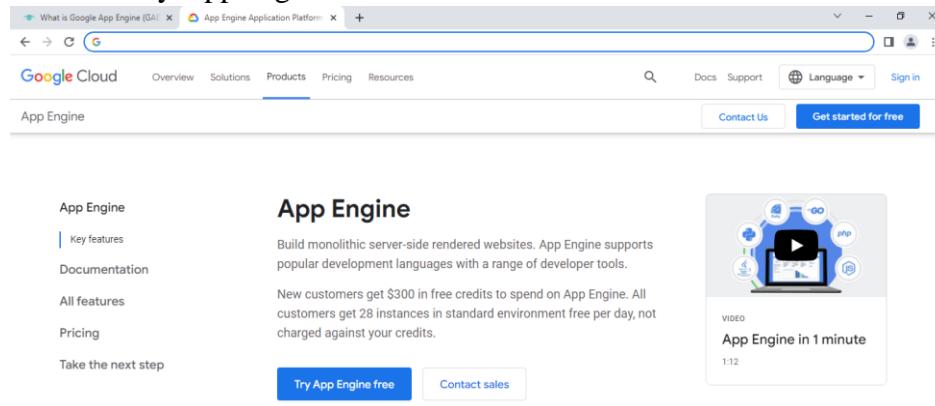
Practical No. 2

Aim: Installing and configuring the required platform for Google App Engine

Implementation:

A. Making Google App Engine account

1. Open your google account and go to the following link
<https://cloud.google.com/appengine>
2. Click on Try App engine free button



3. This page will appear, add country and choose other in describes and click Continue.

Country

What best describes your organization or needs?

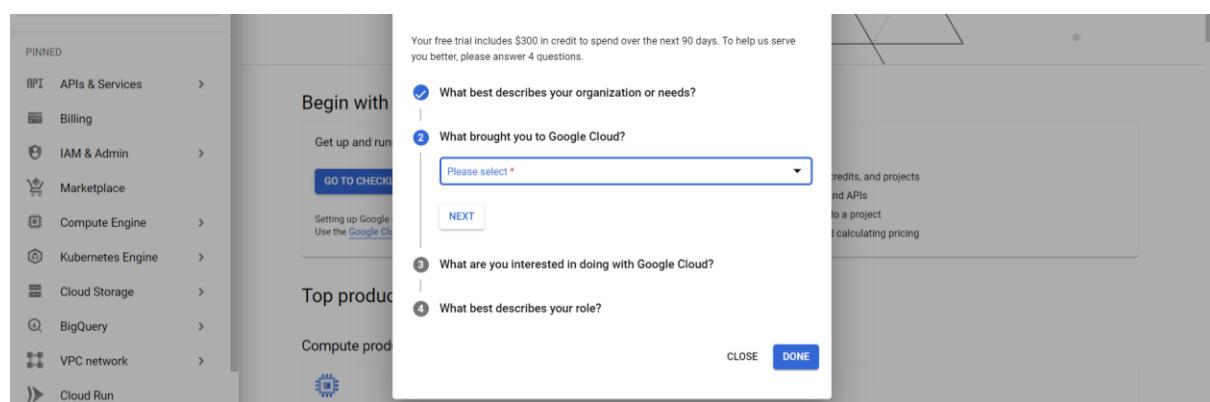
Terms of Service

I have read and agree to the [Google Cloud Platform Terms of Service](#), [Supplemental Free Trial Terms of Service](#), and the terms of service of [any applicable services and APIs](#).

Required to continue

CONTINUE

4. Add you card details (Visa or MasterCard only) and do the payment of Rs. 2
 Fill this according to you purpose



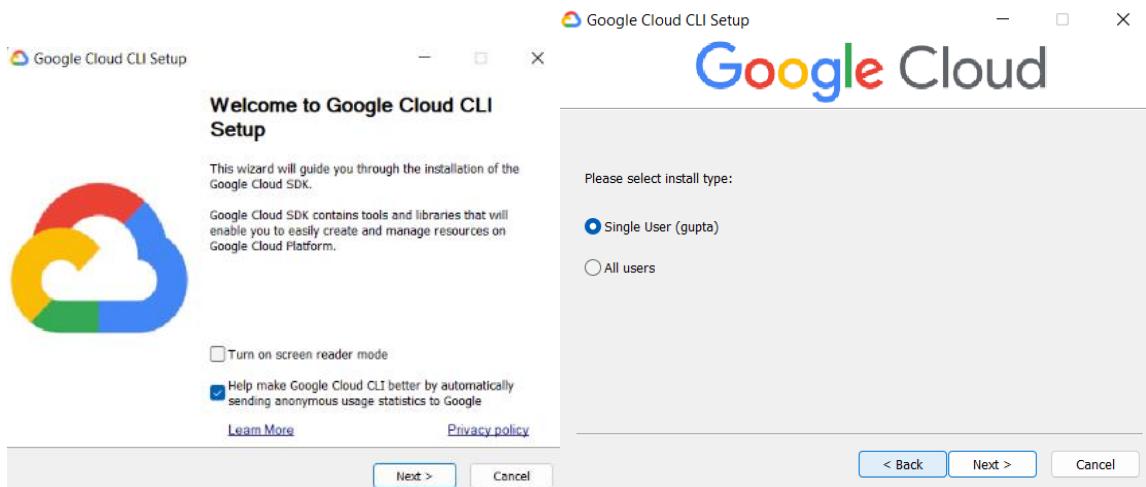
After all procedure we will get,

B. Using GAE account, to download Google Cloud SDK.

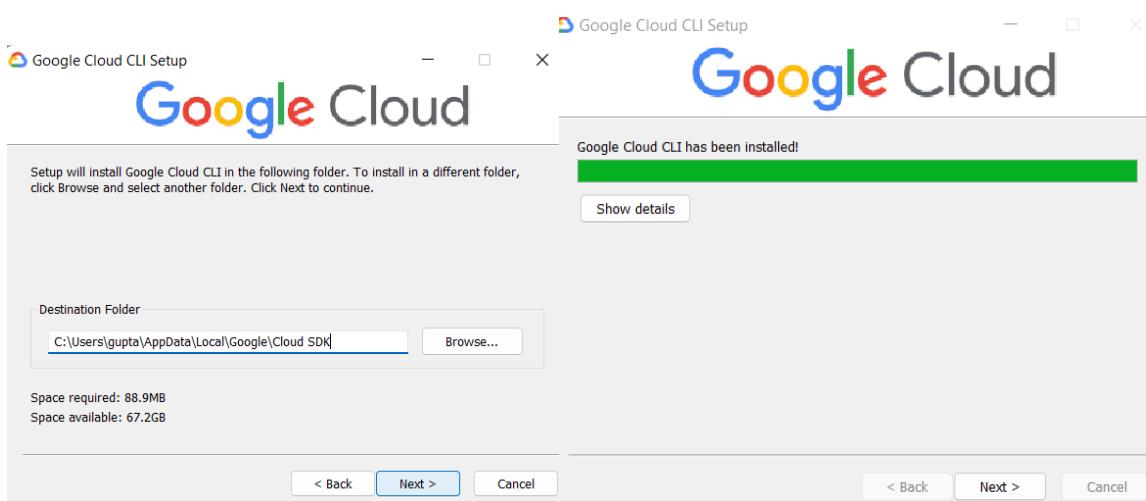
5. Go to google and search **google cloud sdk** go to site <https://cloud.google.com/sdk> then Get Started (you should use this with same account used in google app engine)

6. Install the Google Cloud CLI by clicking on **Google Cloud CLI installer**

7. Launch the installer and follow the prompts. The installer is signed by Google LLC.



8.



After installing it will ask for log in as shown.

Select 5 from it, close this and open Google Cloud SDK Shell it will look like

```
Google Cloud SDK Shell
Welcome to the Google Cloud CLI! Run "gcloud -h" to get the list of available commands.
[...]
C:\Users\gupta\AppData\Local\Google\Cloud SDK>
```

Add the installer path in system also.

Run command ‘ gcloud components install app-engine-java’

Practical No. 3

Aim: Studying the features of the GAE PaaS model.

Platform as a Service (PaaS) - What is it?

PaaS is a category of cloud computing services that provide a computing platform and a solution stack as a service.

Along with software as a service (SaaS) and infrastructure as a service (IaaS), it is a service model of cloud computing. In this model, the consumer creates the software using tools and/or libraries from the provider/vendor. The consumer also controls software deployment and configuration settings.

The provider provides the networks, servers, storage and other services. PaaS offerings facilitate the deployment of applications without the cost and complexity of buying and managing the underlying hardware and software and provisioning hosting capabilities.

PaaS Key Features

1. Services to develop, test, deploy, host and maintain applications in the same integrated development environment
2. Web-based management/administration consoles
 - ✓ Reducing the need for system administration/dev ops
 - ✓ Resource utilization monitoring capabilities
 - ✓ Easily identify bottlenecks
 - ✓ Multi-tenant architecture
 - ✓ Certain PaaS offerings attempt to support use of the application by many concurrent users, by providing concurrency management, scalability, fail-over and security
 - ✓ Support for development team collaboration
 - ✓ Pay for what you use billing model Stop

PaaS - Popular offerings

Heroku

One of the first cloud platforms, has been in development since June 2007, when it supported only the Ruby programming language, but has since added support for Java, Node.js, Scala, Clojure, Python and (undocumented) PHP

Windows Azure

Microsoft's cloud computing platform used to build, deploy and manage applications through a global network of Microsoft-managed datacenters

dotCloud

Founded in 2008 by Solomon Hykes, dotCloud is the first application platform designed from the ground up for modern service-oriented development

Cloud Foundry

- Developed by VMware released under the terms of the Apache License 2.0
- Primarily written in Ruby
- AppCloud runs on Cloud Foundry
- Since it is open sourced, ActiveState has created a commercial distribution of the Cloud Foundry **software for enterprises to host their own private PaaS**

Engine Yard

A San Francisco, California based, privately held platform as a service company focused on Ruby on Rails and PHP, and recently announced support for Node.js deployment and management

Google App Engine (often referred to as GAE or simply App Engine, and also used by the acronym GAE/J)

- A cloud computing platform for developing and hosting web applications in Google-managed data centers
- Applications are sandboxed and run across multiple servers
- Offers automatic scaling for web applications-as the number of requests increases for an application, App Engine automatically allocates more resources for the web application to handle the additional demand
- Is free up to a certain level of consumed resources. Fees are charged for additional storage, bandwidth, or instance hours required by the application
- First released as a preview version in April 2008, and came out of preview in September 2011

What is Google App Engine?

Google App Engine lets you run web applications on Google's infrastructure. App Engine applications are easy to build, easy to maintain, and easy to scale as your traffic and data storage needs grow. With App Engine, there are no servers to maintain: You just upload your application, and it's ready to serve your users.

The Application Environment

Google App Engine makes it easy to build an application that runs reliably, even under heavy load and with large amounts of data. App Engine includes the following features:

- Dynamic web serving, with full support for common web technologies
- Persistent storage with queries, sorting and transactions
- Automatic scaling and load balancing
- APIs for authenticating users and sending email using Google Accounts
- A fully featured local development environment that simulates Google App Engine on your computer

Your application can run in one of three runtime environments: the Go environment, the Java environment, and the Python environment, which gives you a choice of Python 2.5 or Python 2.7.

Why App Engine?

Pros

- Easy to Get Started
- Automatic Scalability
- The Reliability, Performance, and Security of Google's Infrastructure
- Costs less
- There is a generous free usage quota and you only pay for what you use

Cons

- Sandboxed environment limits the scope of your application
- Although we can pay for certain additional resources, there are some that have a hard limit

Traditional Way

1. Write your code . . .
2. Configure & Deploy Web server (Apache/Tomcat)
3. Configure & Deploy SQL database
4. Maintain all of these infrastructure
5. Cost of building and maintaining the infrastructure

App Engine Way

1. Write your code
2. A set of simple configurations to let App Engine know how to serve your application

Tools Bundled with the SDK

Development Server

Uploading and Managing an App

Uploading and Downloading Data

ProtoRPC

webapp Framework

Local Unit Testing

Appstats

Included Libraries (Python 2.5)

- Django, PyCrypto, YAML, zipimport

Included Libraries (Python 2.7)

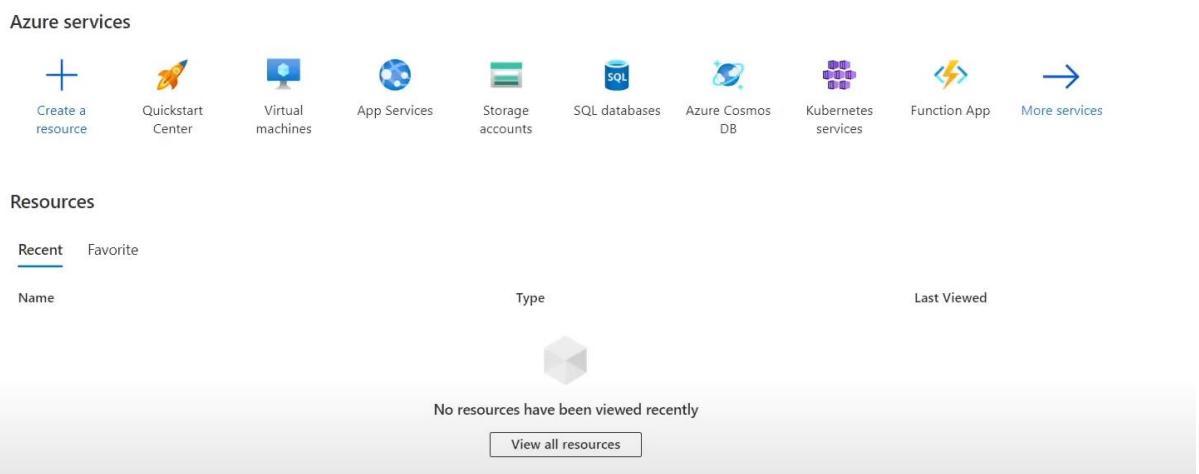
- Jinja2, PIL, webapp2, etc Qburst I meet.google.com is sharing your screen.

Practical No. 4

Aim: Developing an ASP.NET based web application on the Azure platform

Implementation:

1. Go to the Azure portal and click on free account at top right of the home page and go to start page
2. Create a new account and enter debit card details to finalize the account generation process
3. You can view the following image once done



4. Make sure to have installed Visual Studio 2022, and open it
5. Create a new project > ASP.NET Core Web App > Next > Create
6. In your File tree, open pages folder and edit the index.cshtml like so

```
<div class="text-center">
    <h1 class="display-4">UDCS Trends in Cloud Computing</h1>
    <p>Learn about <a href="https://docs.microsoft.com/aspnet/core">Practical no. 5, Hosting a ASP.NET App on Azure</a>.</p>
</div>
```

7. Run in localhost to verify its working

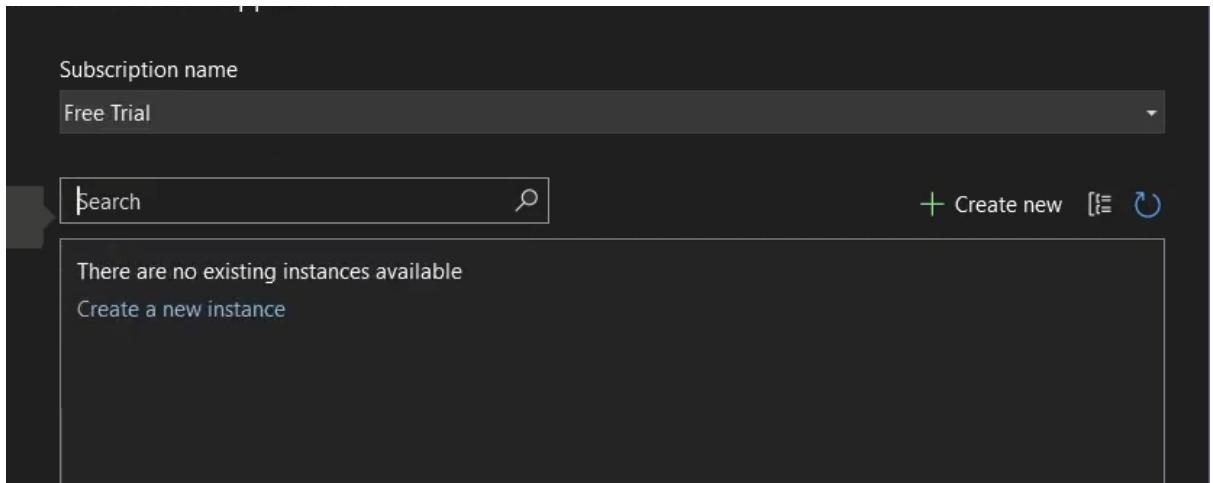
Prac5App Home Privacy

UDCS Trends in Cloud Computing

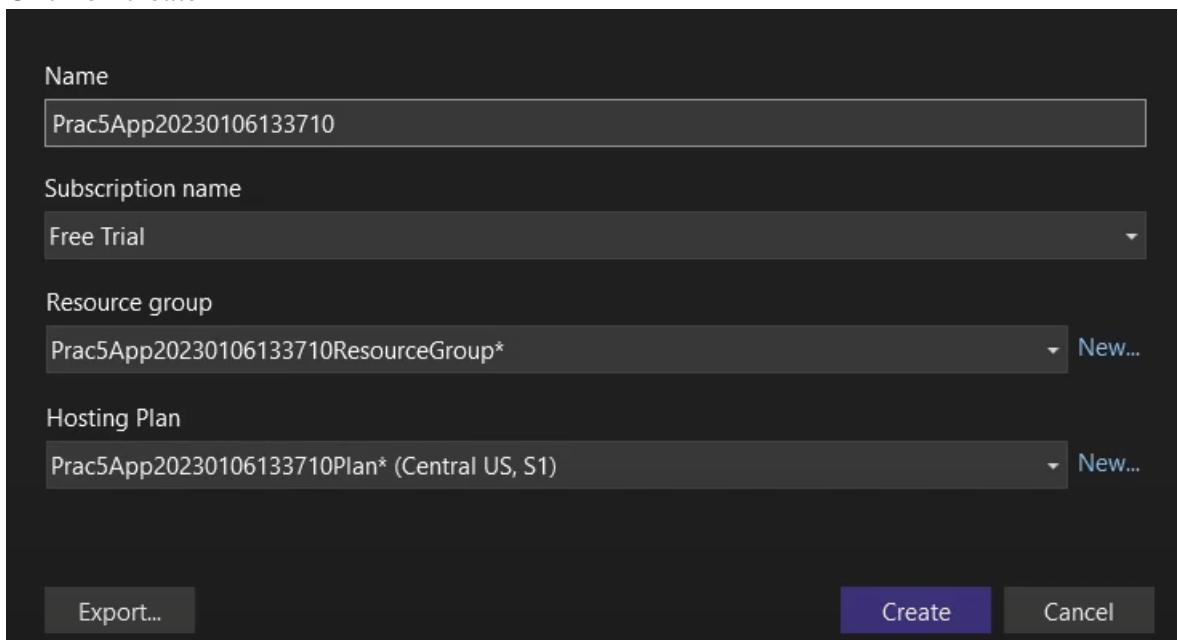
Learn about [Practical no. 5, Hosting a ASP.NET App on Azure](https://docs.microsoft.com/aspnet/core).

8. Now in Build tab, click on public [AppName]

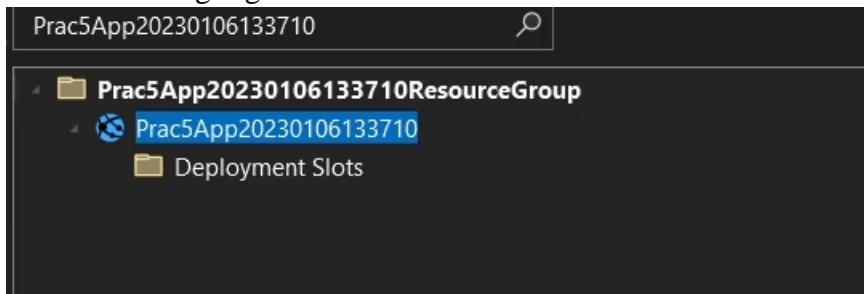
9. Select ‘Azure’ > Next > Azure App Service (windows) > Next
10. Make sure your Microsoft account used for Azure is signed in and currently selected
11. Go for free trial and click on the ‘+ Create new’ to create a new resource



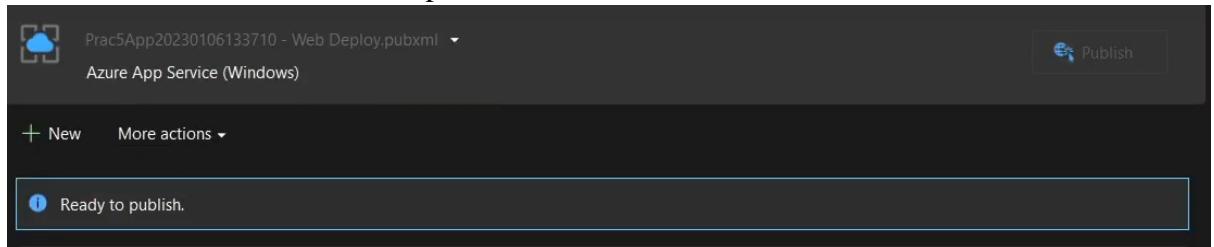
12. Click on create



13. Click on the highlighted resource and click on Finish



14. Click on close and then click on publish



15. Scroll down to find and copy the URL generated and paste it in a browser to view the published and hosted web app

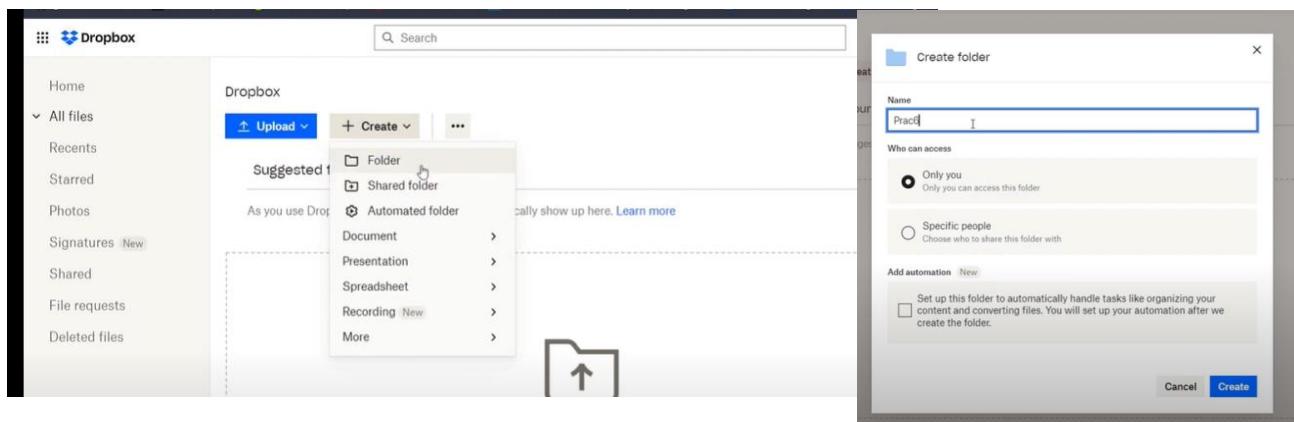
A screenshot of a published Azure web application. At the top, it shows the "Resource name" as "Prac5App20230106133710". Below that, the "Site" URL is listed as "https://prac5app20230106133710.azurewebsites.net" with a copy icon next to it. A "Service Dependencies" section is visible. The main content area displays a sample page titled "UDCS Trends in Cloud Computing". The page includes a navigation bar with "App", "Home", and "Privacy" links. Below the title, there's a link to "Learn about Practical no. 5, Hosting a ASP.NET App on Azure." The page has a clean, modern design with a light gray background and white text.

Practical 5

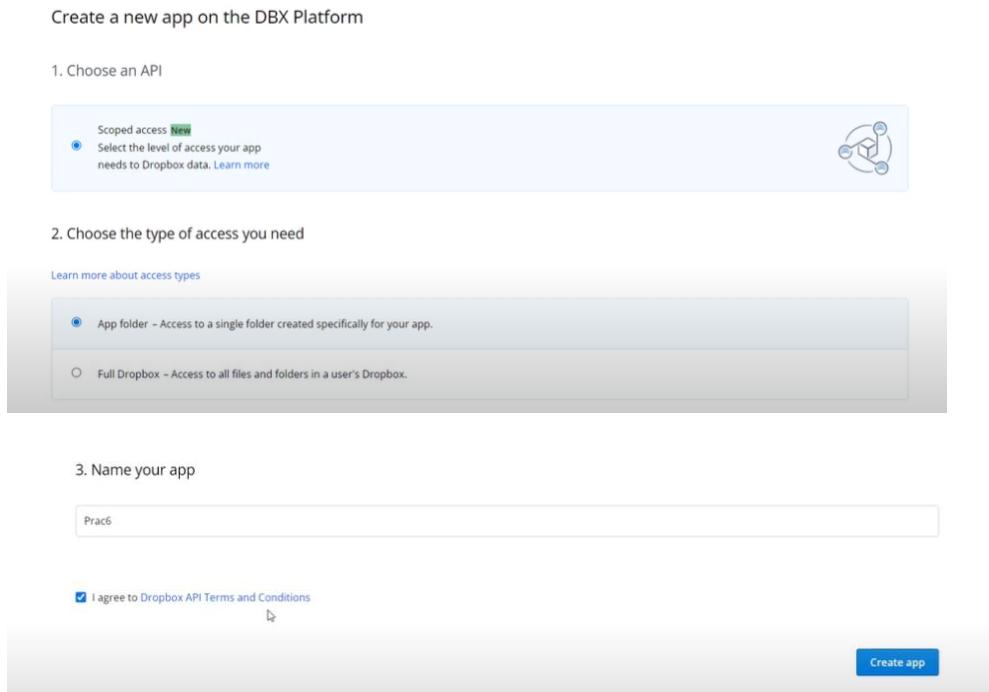
Aim: Creating an application in Dropbox to store data securely. Develop a source code using Dropbox API for updating and retrieving files.

Implementation:

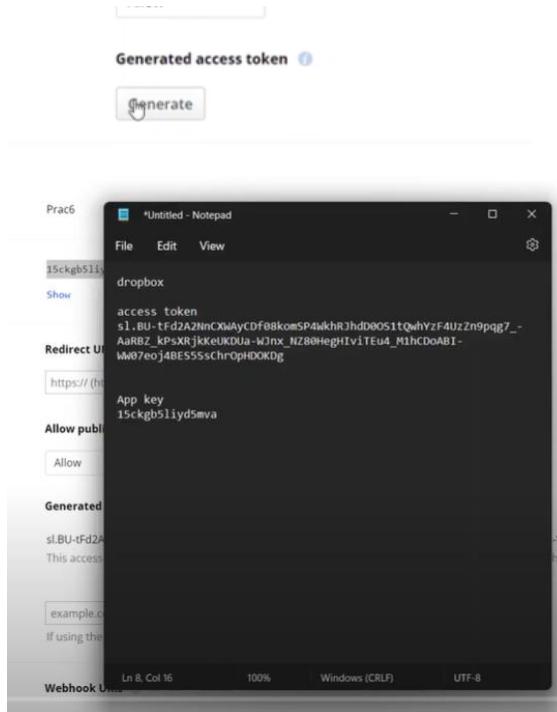
1. Go to <https://www.dropbox.com/> and create an account. You can sign in with your google account.
2. Click on Continue with 2 GB Dropbox Basic plan at the end of the page. Create a ‘Prac6’ folder.



3. Go to <https://www.dropbox.com/lp/developers> and click on **Create apps**.



4. Click on generate, copy the generated access token and App key into a notepad file for later use.



```

Generated access token ⓘ
Generate

Prac6 *Untitled - Notepad
File Edit View
15ckgb5111
Show
dropbox
access token
s1_BU-tFd2A2NnCx4AyCDF08komSP4WkhRJhdD00S1tQwhYZF4UzZn9pqg7_-
AaRBZ_kpSXjkkeUKDUa-Wlnx_NZ80HegHiViTEu4_MlhCDoABI-
Iw07eoj48E555sChrOpHDOKDg
Redirect U https://(h
App key
15ckgb51iyd5mva
Allow publ
Allow

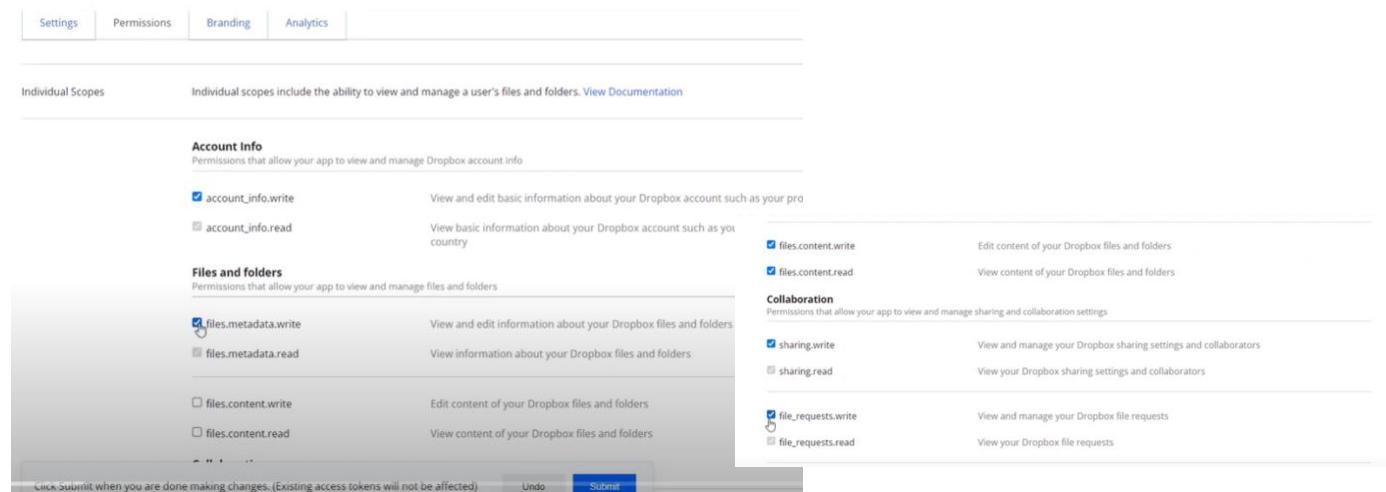
Generated
s1_BU-tFd2A2NnCx4AyCDF08komSP4WkhRJhdD00S1tQwhYZF4UzZn9pqg7_-
This access token can be used to make requests to the Dropbox API on your behalf.

example.c
If using the example code, you will need to replace the access token with this one.

Webhook URL
Ln 8, Col 16 100% Windows (CRLF) UTF-8

```

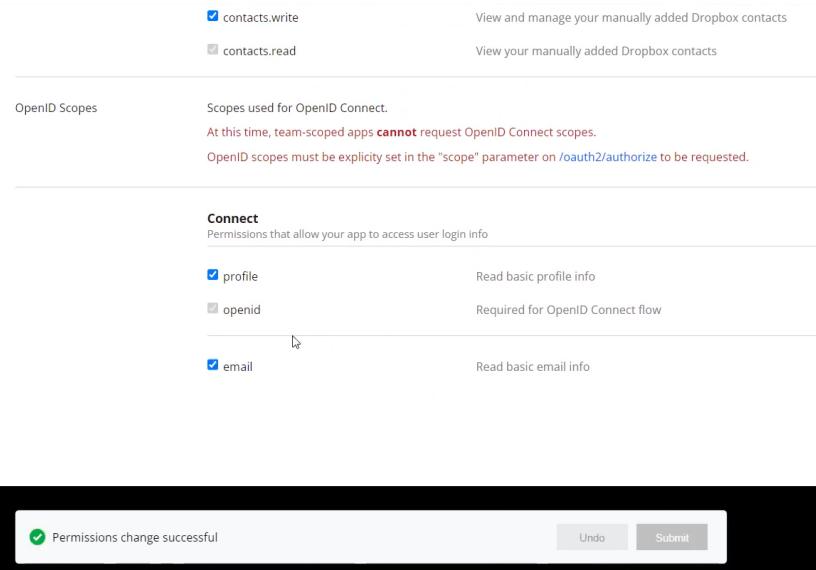
5. Go to permissions and check all the following boxes. After that click on submit.



The screenshot shows the 'Permissions' tab selected in the top navigation bar. The page displays various permission scopes grouped into categories: Account Info, Files and folders, Collaboration, and Requests. Most checkboxes are checked by default. A note at the bottom says 'Click submit when you are done making changes. (Existing access tokens will not be affected)'.

Category	Scope	Description
Account Info	<input checked="" type="checkbox"/> account_info.write	View and edit basic information about your Dropbox account such as your profile picture and bio.
	<input type="checkbox"/> account_info.read	View basic information about your Dropbox account such as your country.
Files and folders	<input checked="" type="checkbox"/> files.metadata.write	View and edit information about your Dropbox files and folders.
	<input type="checkbox"/> files.metadata.read	View information about your Dropbox files and folders.
	<input type="checkbox"/>	Edit content of your Dropbox files and folders.
	<input type="checkbox"/>	View content of your Dropbox files and folders.
Collaboration	<input checked="" type="checkbox"/> sharing.write	View and manage your Dropbox sharing settings and collaborators.
	<input type="checkbox"/> sharing.read	View your Dropbox sharing settings and collaborators.
Requests	<input checked="" type="checkbox"/> file_requests.write	View and manage your Dropbox file requests.
	<input type="checkbox"/> file_requests.read	View your Dropbox file requests.

Click submit when you are done making changes. (Existing access tokens will not be affected) Undo Submit



6. Go to <https://colab.research.google.com/>, create a new notebook and write the following code. Create a notepad file with some texts, upload it to dropbox folder ‘Prac6’.

```
import dropbox
dropbox_access_token=
's1.BUTEvh4dpk6Cma3FXLv2ToJfIGwWDJ6WmruJdbLeDFVLGqoE7g_Jcmy2Yfujqz_eHH
0Rr82G0gWDIASZUQgDo6co4WrbN-
YiuG5JgKiOGBRc7WlanGHglTejJhuqZ8LMLxdaWq9gvRF'      #Enter your own access
token
dropbox_path= "/Prac6/aa.txt"
computer_path="aa.txt"

client = dropbox.Dropbox(dropbox_access_token)
print("[SUCCESS] dropbox account linked")
```

```
client.files_upload(open(computer_path, "rb").read(), dropbox_path)
print("[UPLOADED] {}".format(computer_path))
```

```
import dropbox
dropbox_access_token =
's1.BU8sNjWPUs69LP3K2PH5xJNUd6pcftseLcKHTLMB4jeLVQB0tqS1rnnyFAtDOPZKEL4YeJcx-9zM3WtyR4_zxJMb254nDkhnAjG1N46kBifv1Hynwl
dropbox_path = "/Prac6/aa.txt"
computer_path="aa.txt"

client = dropbox.Dropbox(dropbox_access_token)
print("[SUCCESS] dropbox account linked")

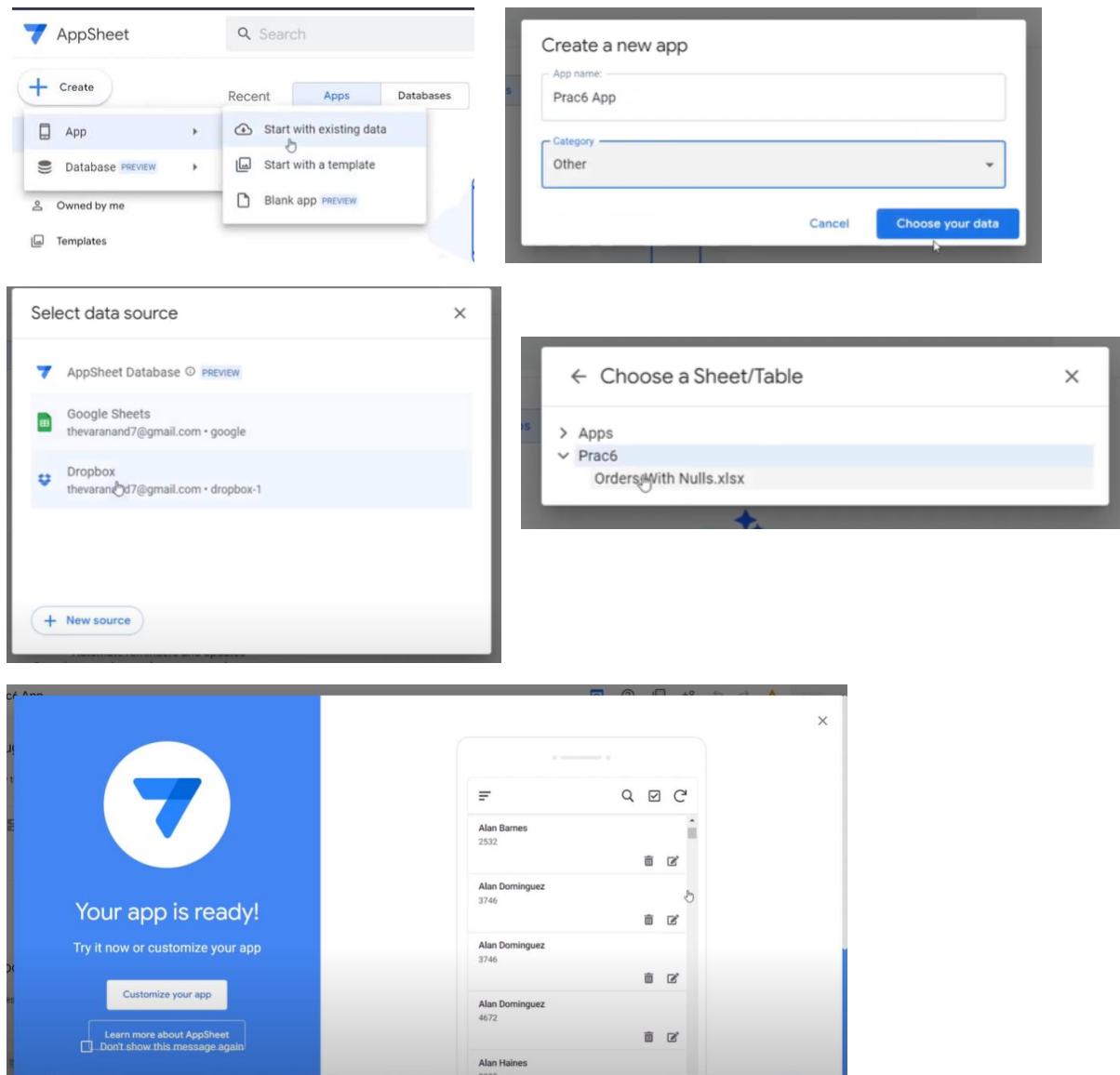
client.files_upload(open(computer_path, "rb").read(), dropbox_path)
print("[UPLOADED] {}".format(computer_path))

[SUCCESS] dropbox account linked
[UPLOADED] aa.txt
```

7. Go to <https://www.wisdomaxis.com/technology/software/data/for-reports/>, get your data sample from here and also upload the excel file in dropbox folder ‘Prac6’.



8. Go to <https://about.appspot.com/>, sign in with google. Create a new app. Choose the sample for the data.



Application created.

Practical No. 6

Aim: Installing Cloud Foundry in localhost and exploring CF commands.

Implementation:

1. **Installing Windows Powershell for CloudFoundry** <https://docs.cloudfoundry.org/cf-cli/install-go-cli.html>

The screenshot shows the Cloud Foundry Documentation page. In the sidebar, under 'Cloud Foundry Command Line Interface (cf CLI)', there is a link to 'Binaries'. The main content area has a heading 'Installers' followed by a bulleted list of operating system packages: Debian 64 bit / 32 bit (deb), Redhat 64 bit / 32 bit (rpm), macOS 64 bit / arm (pkg), and Windows 64 bit / 32 bit (zip). Below this is another heading 'Binaries' with a similar list of packages for Linux, macOS, and Windows.

Download installer or binary file:

This part of the screenshot shows the 'Binaries' section of the documentation. It lists download links for various operating systems and architectures: Linux 64 bit / 32 bit (tgz), macOS 64 bit / arm (tgz), and Windows 64 bit / 32 bit (zip).

Install the file and set a system path. Then open the Windows Powershell. Type >cf to check, and if you're getting the information regarding CF, then you've successfully installed it.

The screenshot shows a Windows PowerShell window. The command 'PS C:\Users\DELL> cf' is entered, and the output displays a detailed list of Cloud Foundry CLI commands categorized into Space management, Org management, CLI plugin management, Commands offered by installed plugins, Global options, and a tip about using 'cf help -a'.

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\DELL> cf

Space management:
spaces      create-space    set-space-role
space-users delete-space   unset-space-role

Org management:
orgs,o     set-org-role
org-users  unset-org-role

CLI plugin management:
plugins      add-plugin-repo  repo-plugins
install-plugin list-plugin-repos

Commands offered by installed plugins:
bg-deploy    mta            purge-mta-config
deploy       mta-ops        undeploy
download-mta-op-logs,dm01  mtas

Global options:
--help, -h          Show help
-v                  Print API request diagnostics to stdout

TIP: Use 'cf help -a' to see all commands.
PS C:\Users\DELL> cf login

```

Exploring Commands

2. Login with the command >cf login

It will have the same endpoint as IBM and we will use the same email and password as the account created in the IBM Cloud practical.

```
PS C:\Users\DELL> cf login
API endpoint: https://api.run.pivotal.io
Email: msccloudprg@gmail.com
Password:
Authenticating...
Credentials were rejected, please try again.

Password:
Authenticating...
OK

API endpoint: https://api.run.pivotal.io (API version: 2.153.0)
User: msccloudprg@gmail.com
No org or space targeted, use 'cf.exe target -o ORG -s SPACE'
ps C:\Users\DELL> cf target -o AA
```

3. Generate source target using the command >cf create-org AA

```
OSNAME:
cf.exe create-org ORG
ALIAS:
  co
OPTIONS:
  -q      Quota to assign to the newly created org (excluding this option results in assignment of default quota)
SEE ALSO:
  create-space, orgs, quotas, set-org-role
PS C:\Users\DELL> cf create-org AA
Creating org AA as msccloudprg@gmail.com...
OK

Org AA already exists.

PS C:\Users\DELL> =
```

Practical No. 7

Aim: Cloud application development using IBM Bluemix Cloud.

Implementation:

1. Log in to IBM Cloud at cloud.ibm.com.

The image consists of five screenshots arranged in a grid, illustrating the steps to create an IBM Cloud account:

- Screenshot 1: Create an IBM Cloud account**
Shows the initial account creation page with fields for Email and Password, and links for Verify email, Personal information, and Account notice.
- Screenshot 2: Verify identity**
Shows the Verify identity step with options for Account information (selected), Billing information, Tax information, and Credit card information. It includes fields for Account type (Company or Personal) and a Next button.
- Screenshot 3: Billing information**
Shows the Billing information step with fields for First name (Sydney), Last name (Jack), Country or region (India), Address line 1 (Mumbai), Address line 2 (optional), City (Mumbai), State (Maharashtra), Postal Index Number (401107), and Phone number (+91 98282 67045). It also has a Search for new address link.
- Screenshot 4: Tax information**
Shows the Tax information step with a Permanent Account Number (PAN) field containing ACVPS3649S, a note about electronic invoices, and a Next button.
- Screenshot 5: Credit card information**
Shows the Credit card information step with a note about usage and spending notifications, fields for Card number, Expiration date (mm/yy), Security code, and a Next button. It also features logos for MasterCard and VISA.

2. Go to the catalog (cloud.ibm.com/catalog).

The screenshot shows the IBM Cloud Catalog interface. At the top, there's a search bar and account information for 'Sydney Jack's Account'. Below the search bar, there are several service icons: a server, a database, a person, a document, and a gear. A large 'Catalog' heading is on the left, and a search bar below it. On the right, there are buttons for 'Sell on IBM Cloud' and 'Catalog settings'. The main area is titled 'Viewing 207 products' and shows three service cards:

- Analytics Engine** by IBM: Submit your Apache Spark applications as needed and customize the Spark runtimes to satisfy the requirements of your application.
- AnonTech VizVault Platform** by Anon Technology, Inc.: Manage personal information as-a-service safely, securely, and in compliance with data privacy regulations using VizVault.
- API Connect** by IBM: An enterprise-grade platform for creating, securing, managing, sharing, monetizing, and analyzing custom APIs located on-premises and on the cloud.

On the left sidebar, there's a 'Category' filter with options like Recommended products (6), Compute (30), Containers (10), Networking (29), Storage (20), AI / Machine Learning (17), Analytics (9), Blockchain (1), and Databases (28).

3. Click the Node-RED boilerplate and fill in the required data to create an instance.

The screenshot shows the 'Node-RED Starter' creation page. It has sections for 'App name:' (with a placeholder 'Enter a unique name'), 'Host name:' (with a placeholder 'Enter a unique name' and a dropdown for 'Domain: mybluemix.net'), 'Selected Plan:' (with a dropdown set to 'Default'), and 'Pricing Plans' (showing a single plan: 'Default' with 'Run one or more apps free for 30 days (215 GB-hours free)' and a price of '90.0905 USD/GB-Hour'). There are also sections for 'SDK for Node.js™' and 'Cloudant NoSQL DB'. At the bottom, there are links for 'Need Help?', 'Contact Bluemix Sales', 'Estimate Monthly Cost', and 'Cost Calculator', and a 'Create' button.

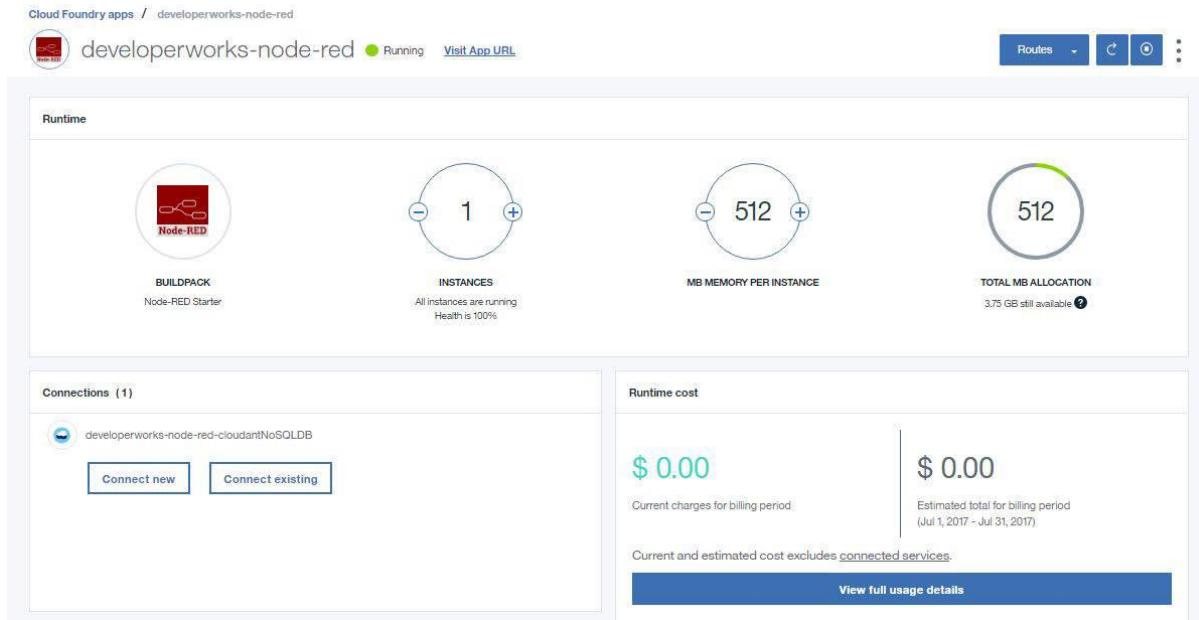
4. Specify an app name and click Create.

A close-up of a form input field labeled 'Name:' containing the text 'MyApp'.

After the process completes, the environment is ready to use. When you access the IBM Cloud dashboard, the Node-RED instance is in Running state.

The screenshot shows the IBM Cloud dashboard under the 'Cloud Foundry apps' section. It lists an app named 'developerworks-node-red'. The status is shown as 'Running' with a green dot. There are buttons to 'Visit App URL' and other navigation links like 'Getting started', 'Overview', and 'Runtime'.

5. Click **Overview** on the left to access the application information. The information instance is displayed.

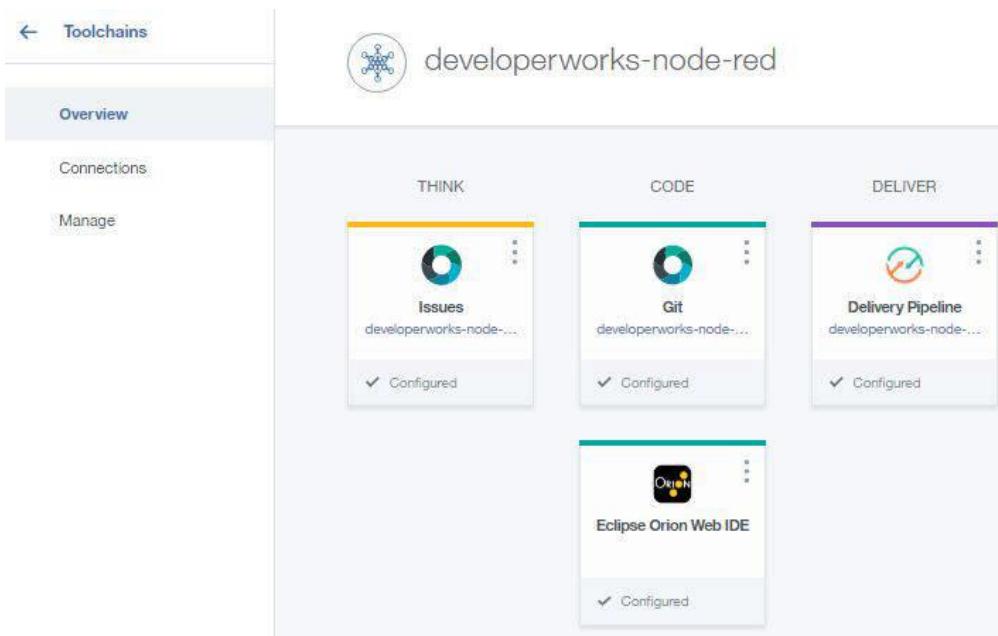


If you named the app app101-node-red, its route would be:

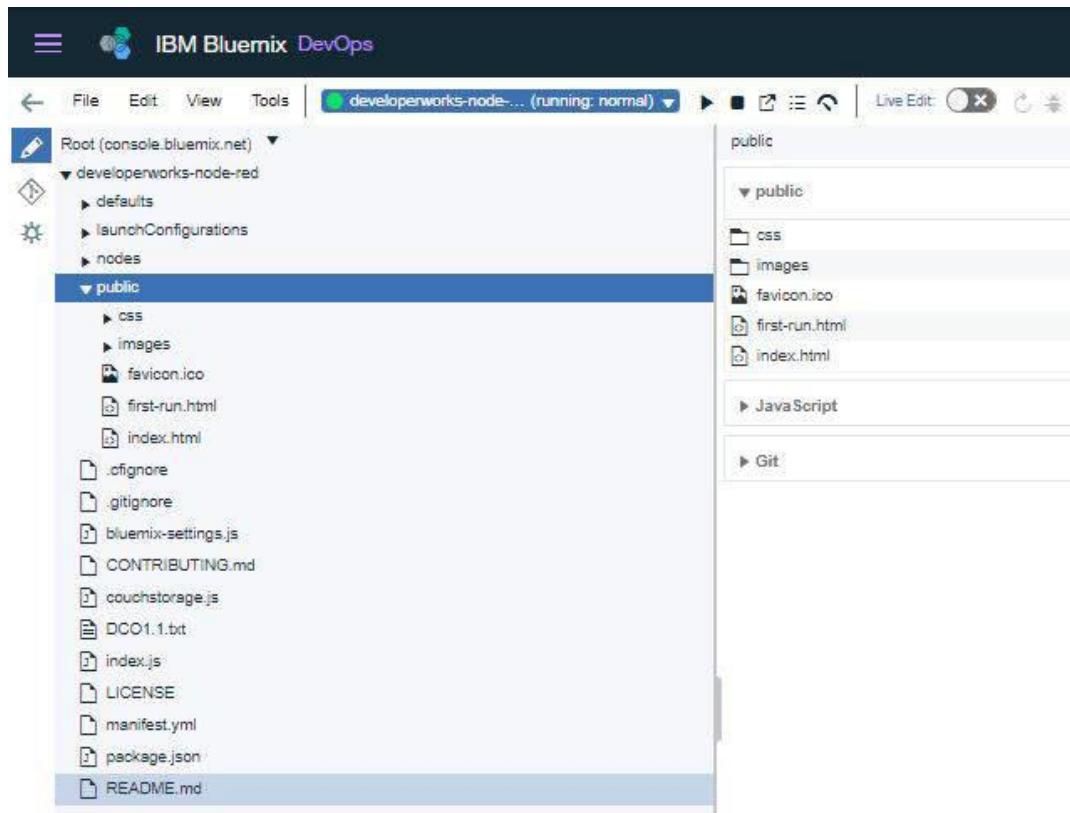
`http://app101-node-red.mybluemix.net.`

6. To modify this app to meet your requirements, you need to have access to its code. IBM Cloud provides a way to allocate space in a GIT repository, where you can access application code and files. Create this space by clicking **Enable**, located in the lower-right corner, and then **Create** in the next panel.

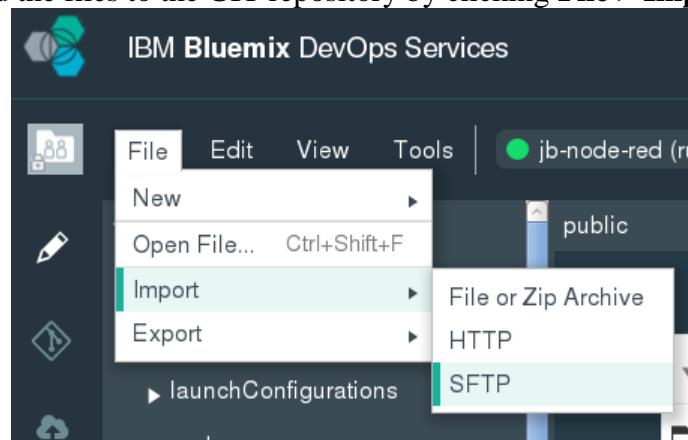
When the process finishes, you see the Toolchain.



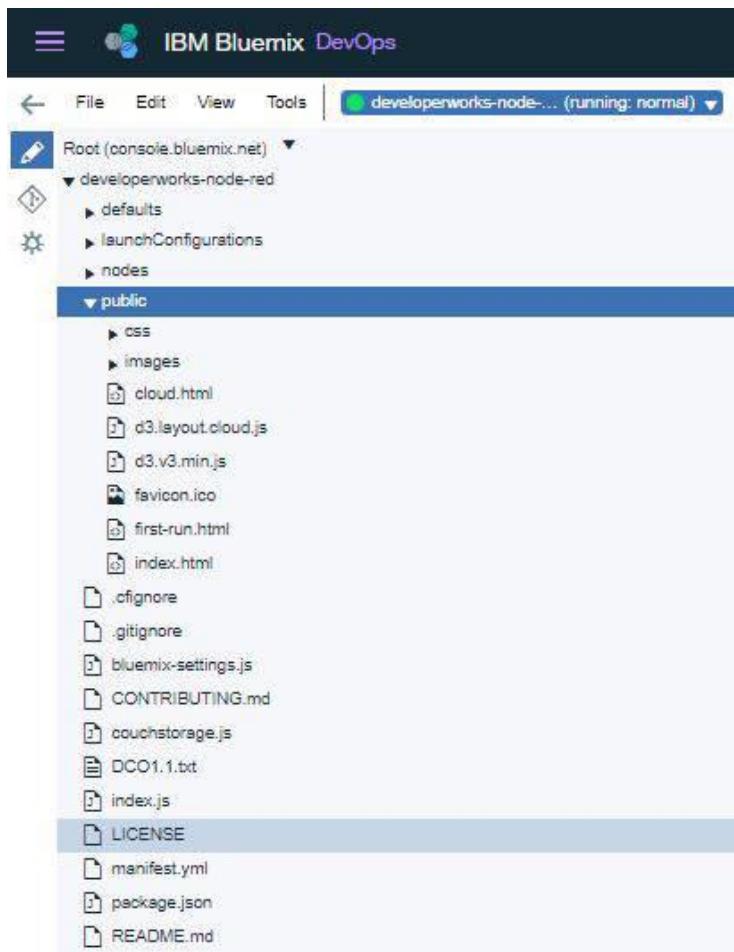
7. To access the code, click **Eclipse Orion Web IDE**, select your application on the left-hand side, and then click the public directory.



8. To set up your app, you need to add and modify these files in the public directory:
- cloud.html
 - d3.layout.cloud.js
 - d3.v3.min.jsDownload these files to your workstation from GitHub at <https://github.com/barabasi/Bluemix-App>.
9. Upload the files to the GIT repository by clicking **File > Import > File or Zip Archive**.



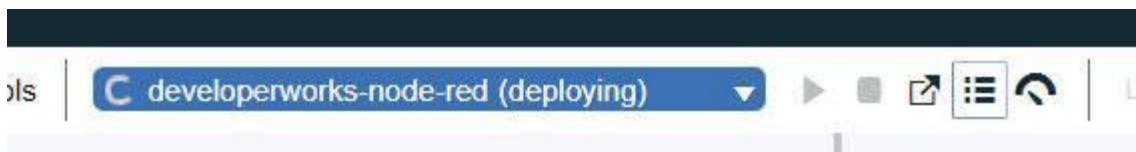
10. After you upload all the files, you need to publish all of the contents of the GIT repository to the running instance on IBM Cloud.



To deploy the changes, press the arrow button. Another option is to enable the **LiveEdit** switch to deploy every change in auto mode.



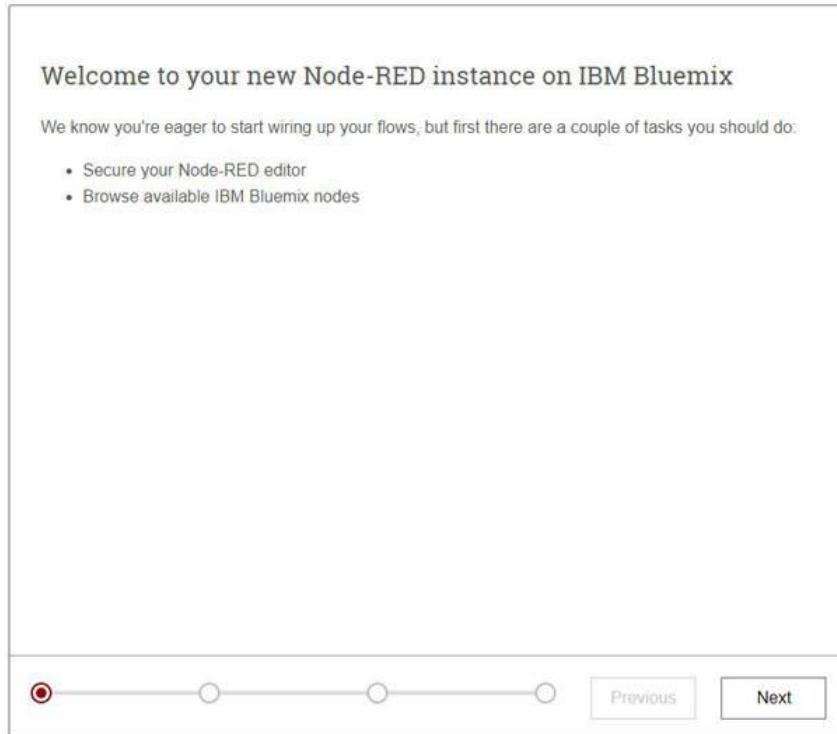
You see a (deploying) state while it is processing.



When the deployment process finishes, the green sign shows (running: normal) again.



11. To create the Node-RED app that will feed the cloud.html file that you just uploaded and deployed, open the Node-RED editor in the browser by clicking the **Link to Application** icon Welcome panel to access the deployed application.
12. The first time you run a Node-Red instance, you need to specify its properties. On the first panel, click **Next**.



13. Enter your username and password, and click **Next**.

Secure your Node-RED editor

Secure your editor so only authorised users can access it

Username

Password good

Allow anyone to view the editor, but not make any changes

Not recommended: Allow anyone to access the editor and make changes

Previous Next

14. Read through the general information panel, and click **Next**.

Browse available IBM Bluemix nodes

There are lots of nodes available from the community that can be used to add more capabilities to your application. The list below is just a small selection.

You can find many more nodes on the [Flow Library](#).

You can use the Palette Manager built into editor to search for and install nodes. Alternatively, you can also edit your application's `package.json` file and adding them to the `dependencies` section.

node-red-dashboard Quickly create dashboards driven by Node-RED	node-red-contrib-ibm-wiotp-device-ops Perform device and gateway operations using the Watson IoT Platform
node-red-contrib-iot-virtual-device Simulate device behavior and use it to run many device instances	node-red-contrib-objectstore Store, delete and restore objects in the ObjectStore service
node-red-contrib-bluemix-hdfs <small>Create and consume files using HDFS for Analytics</small>	node-red-contrib-ibmpush <small>Send push notifications to mobile devices using IBM Push Notifications</small>

Progress bar: [Filled] [Filled] [Filled] [Empty] | Previous | Next

15. Click **Finish** to complete the installation. Your configuration is saved and the Node-Red instance starts.

Finish the install

You have made the following selections:

- Secure your editor so only authorised users can access it

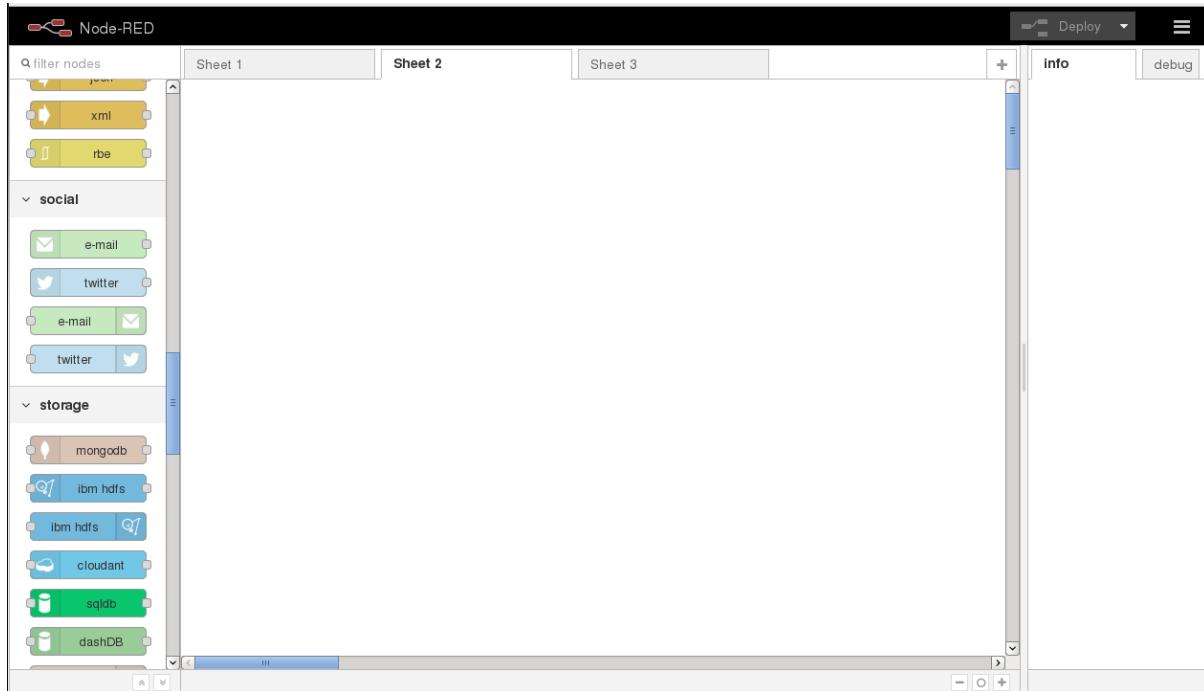
You can change these settings at any time by setting the following environment variables via the Bluemix console:

- `NODE_RED_USERNAME` - the username
- `NODE_RED_PASSWORD` - the password
- `NODE_RED_GUEST_ACCESS` - if set to 'true', allows anyone read-only access to the editor

Progress bar: [Filled] [Filled] [Filled] [Filled] | Previous | Finish

Published in Cloud Computing JOURNAL

16. Select **Go to your Node-RED flow editor** to access the Workflow Editor. Notice your application URL: {your-instance-name}.mybluemix.net.
17. Enter your username and password and click **Login** to open the flow editor.



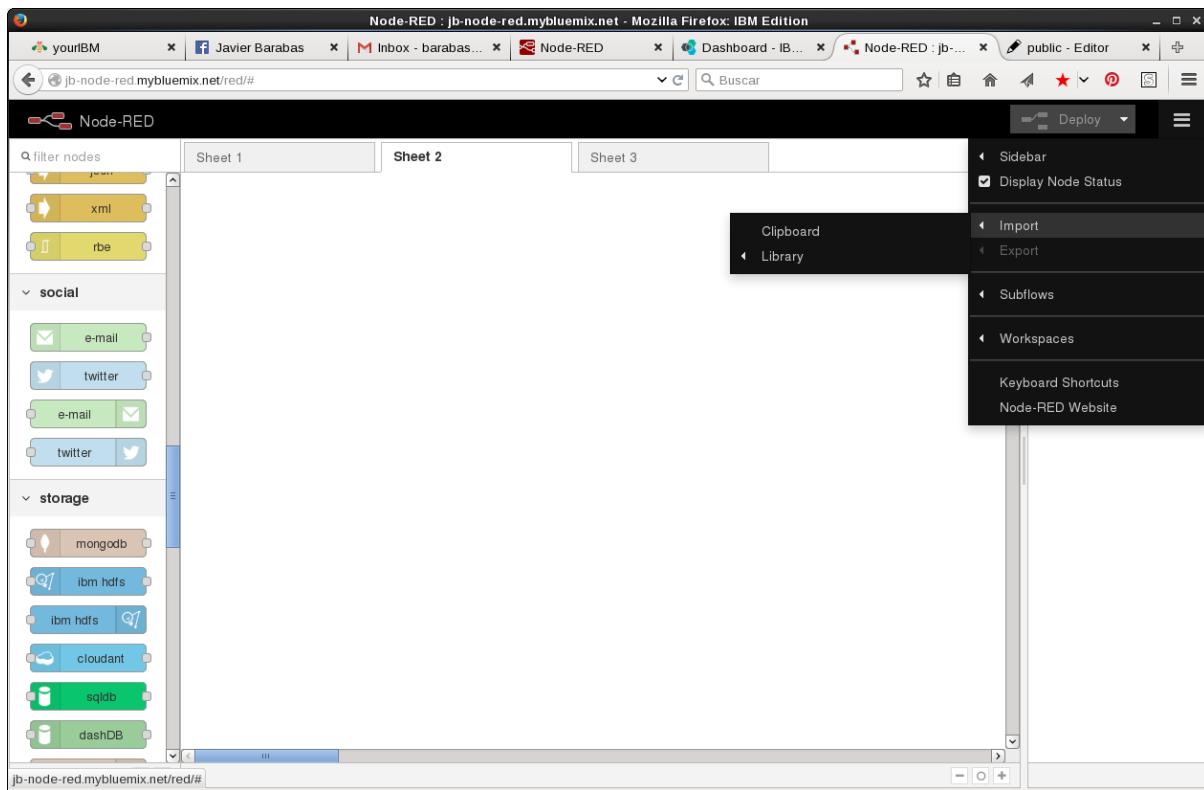
The left-hand nav bar contains all of the tools, services, and functions that you need to compose IBM Cloud apps inside the Node-RED environment. Using the simple drag-and-drop interface, you can build just about any complex app you like. In addition, you can import and export complex code to transfer and reuse. You can use this process to populate your app quickly and easily.

18. One of the files in the [GitHub repo](#) mentioned above is wordcloud.txt. This file contains the text that's exported from the app that you are creating. Select and copy the contents of wordcloud.txt

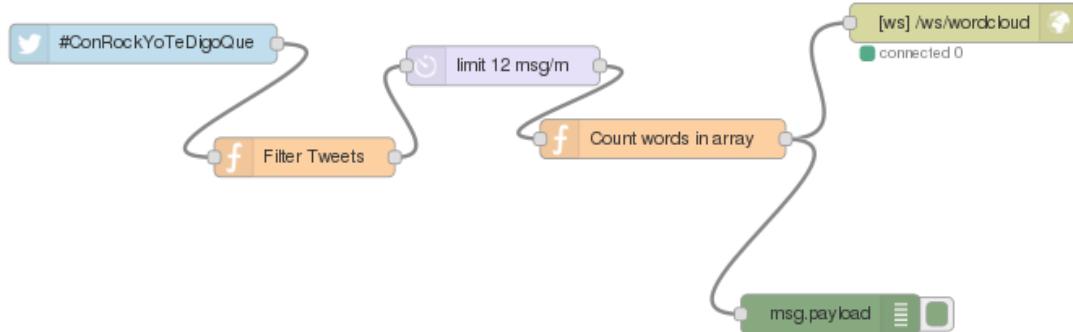
```
[{"id": "1a31b9e0.994c8e", "type": "websocket-listener", "path": "/ws/wordcloud", "whc": "ws://127.0.0.1:8080/ws/wordcloud"}]
```

Show more ▾

19. Next, click **Import > Clipboard** in the Node-RED editor.



The objects that represent the application are shown in the following image:



20. You need to configure some nodes in order to get the application to work. The flow starts reading public tweets accessed by a personal account, filtering the results with a trending topic that ensures that you have matches to be processed by your app. Open the first node (Twitter input):



The help information is displayed on the right:

info **debug**

Node

Type	twitter in
ID	99d296e2.395548

► **Properties**

Twitter input node. Can be used to search either:

- the public or a user's stream for tweets containing the configured search term
- all tweets by specific users
- direct messages received by the authenticated user

Use space for *and* and comma , for *or* when searching for multiple terms.

Sets the **msg.topic** to *tweets/* and then appends the senders screen name.

21. Open the Twitter node by double-clicking on it.

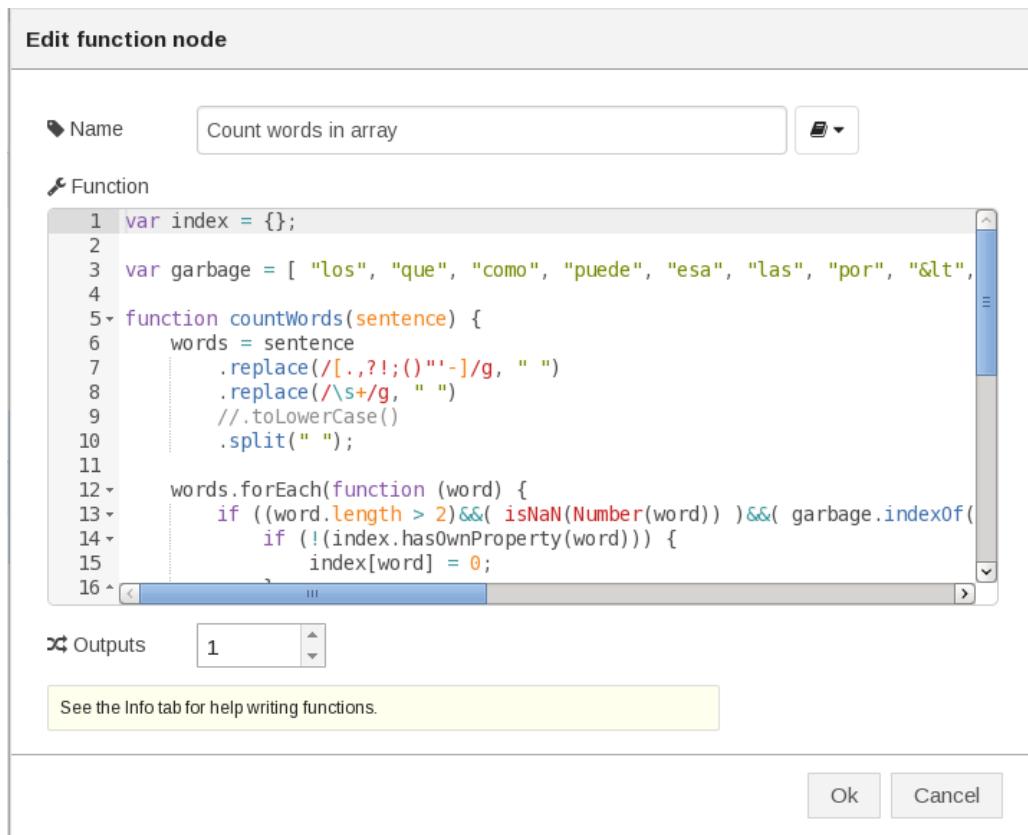
Edit twitter in node

👤 Twitter ID	<input type="text" value="Add new twitter-credentials..."/> 
🔍 Search	<input type="text" value="all public tweets"/>
👉 for	<input type="text" value="#Trending Topic"/>
👉 Name	<input type="text" value="Name"/>

Tip: Use commas without spaces between multiple search terms. Comma = OR, Space = AND.
 The Twitter API WILL NOT deliver 100% of all tweets.
 Tweets of who you follow will include their retweets and favourites.

22. Enter your Twitter ID and any topic you want to display. You can obtain better results by specifying Trending Topic.

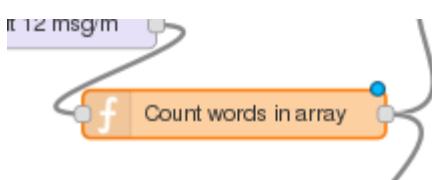
To exclude all non-significant words from the resulting tweets, depending on the language of the matching tweets, you can update the "Count words in array" function node (line 3) to reflect the selected ones. The variable "garbage" must contain all words selected to be ignored by the counters.



23. To publish the changes, click **Deploy** in the upper right.



Every time a change to an object is made, a little dot is displayed on the node. Once the app has been deployed, the dot disappears.



To debug your app and display the processed tweets, a debug node has been added to the flow:



Matching tweets are displayed in the debug window of the GUI:

The screenshot shows the Node-RED interface with the 'info' and 'debug' tabs selected. The 'info' tab is empty. The 'debug' tab displays a log of messages received by the app. The log entries are as follows:

```

15/2/2016 16:35:24 57187a0.0e58808
msg.payload : string [1325]
{@oopsainzmicheli}:1,"Sos":2,"histrica":1,"

...
15/2/2016 16:35:29 57187a0.0e58808
msg.payload : string [1235]
{@El_ejercitoDeMV}:2,"Gracias":1,"ser":2,"

...
15/2/2016 16:35:34 57187a0.0e58808
msg.payload : string [1211]
{#AferradosAEsteAmorVicente}:17,"Sos"

...
15/2/2016 16:35:39 57187a0.0e58808
msg.payload : string [1319]
{@Micaelistapors2}:1,"Mica":4,"hasta":1,"

...

```

24. To display the word cloud created by the app, access the following URL:

http://_<app_name>_.mybluemix.net/cloud.html.



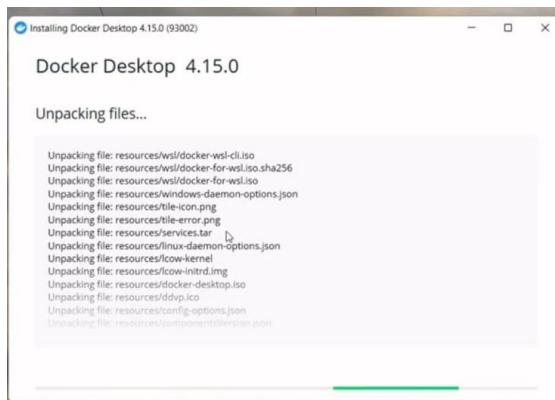
Practical No. 8

Aim: Installing and Configuring Dockers in localhost and running multiple images on a Docker Platform.

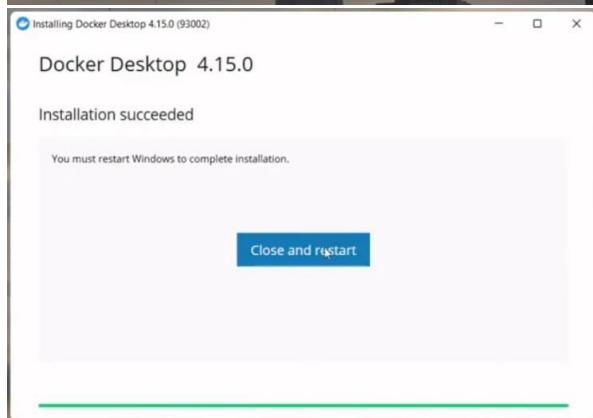
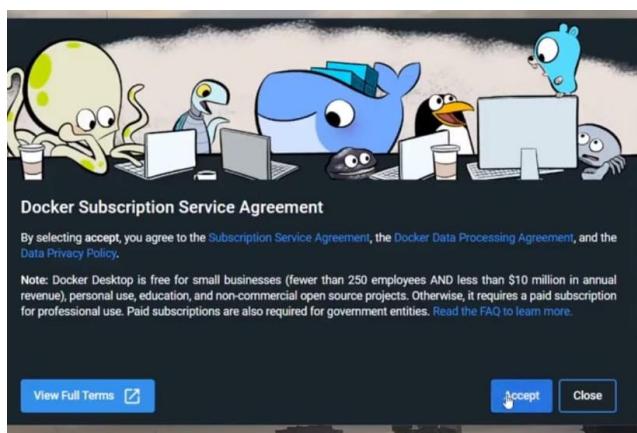
Implementation:

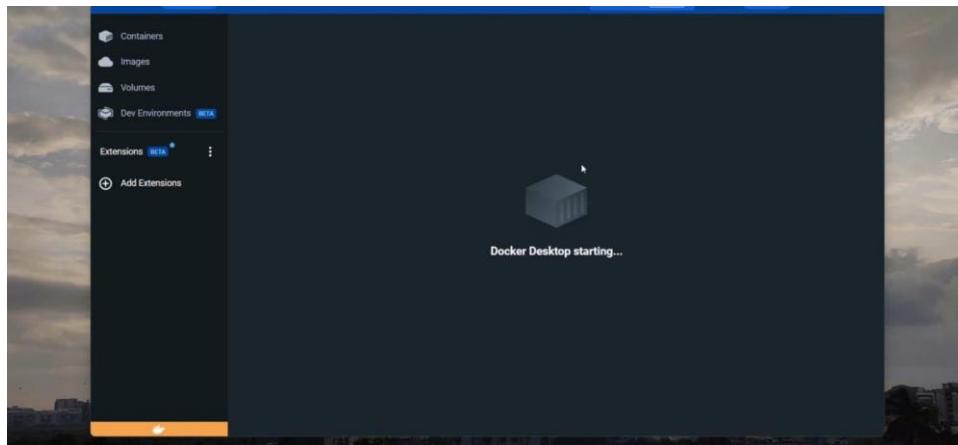
1. Installing Docker Desktop <https://www.docker.com/products/docker-desktop/>

Click on Get Started and download Docker Desktop for Windows

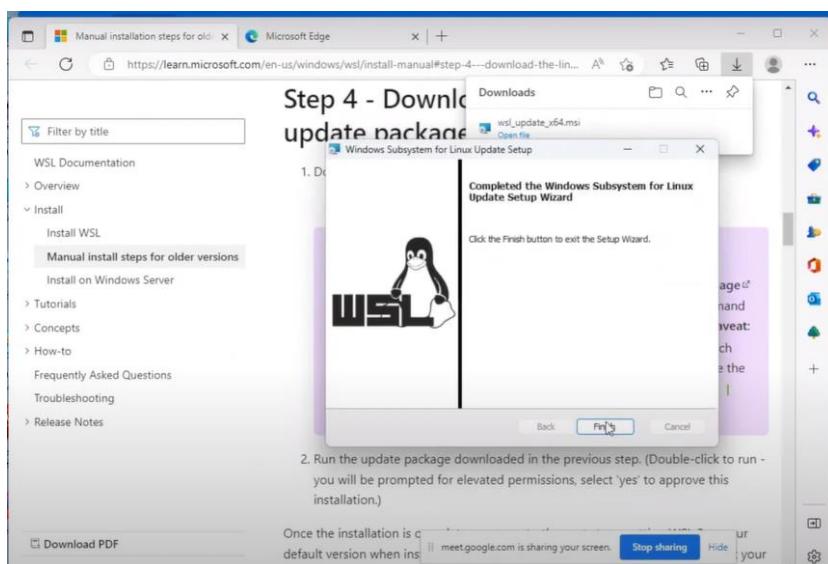
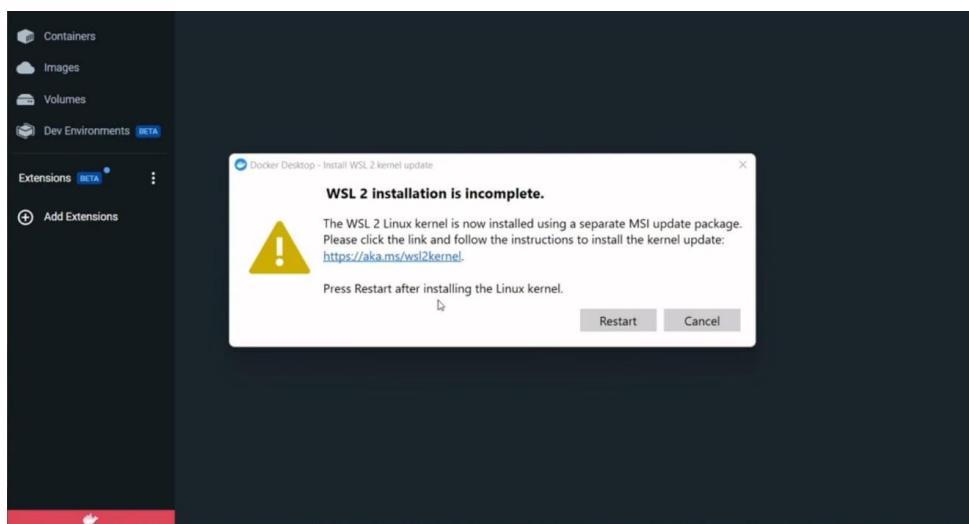


Click on Accept and complete the installation of Docker Desktop





2. Download and Install the updated version of WSL2 Click on the link and download and install the updated version



3. Install minikube <https://minikube.sigs.k8s.io/docs/start/>

Click on the .exe download to download minikube and Install minikube

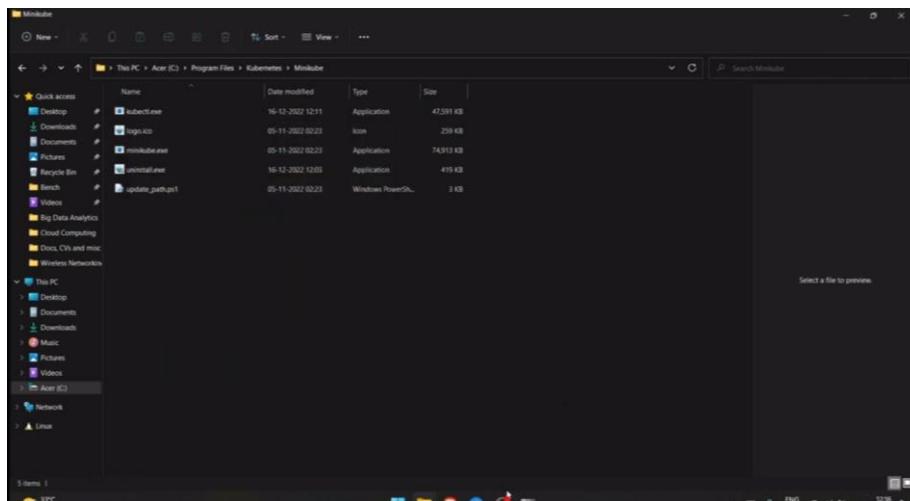
The screenshot shows the minikube documentation page. Under the 'Operating system' section, 'Windows' is selected. Under 'Architecture', 'x86-64' is selected. Under 'Release type', 'Stable' is selected. Under 'Installer type', 'exe download' is selected. A callout box highlights the 'exe download' link, which points to the latest minikube release for Windows.

4. Install kubectl for windows <https://kubernetes.io/docs/tasks/tools/install-kubectl-windows/>

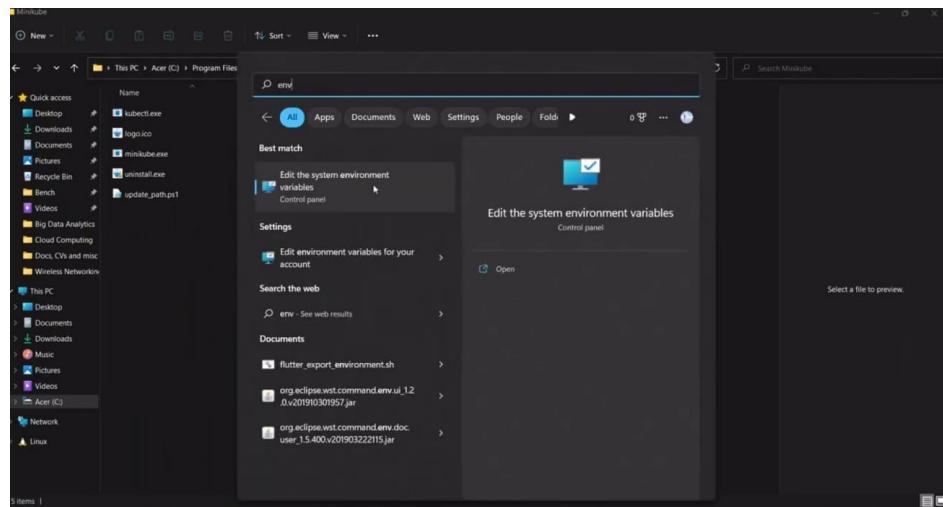
Click on the latest release and download the kubectl

The screenshot shows the Kubernetes documentation page under the 'Tasks' section. It lists two methods for installing kubectl on Windows: 'Install kubectl binary with curl on Windows' and 'Install on Windows using Chocolatey, Scoop, or winget'. The first method is highlighted. Step 1 shows the command 'curl.exe -LO "https://dl.k8s.io/release/v1.26.0/bin/windows/amd64/kubectl.exe"'. Step 2 shows the command 'curl.exe -LO "https://dl.k8s.io/v1.26.0/bin/windows/amd64/kubectl.exe.sha256"'.

Copy the kubectl where minikube is saved and then copy the path

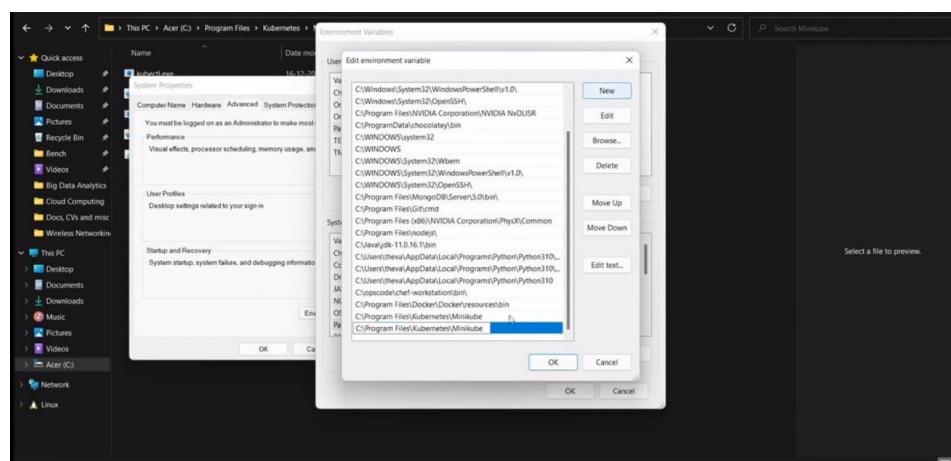
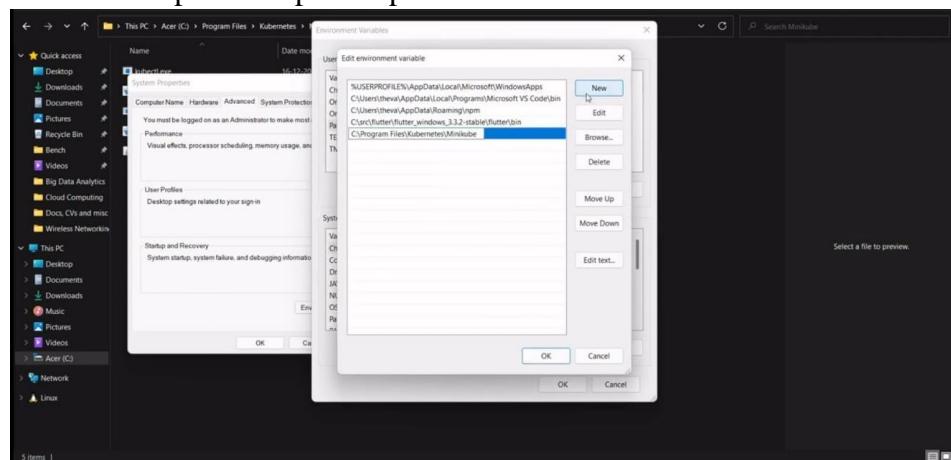


Go to environment variable:



Set the path in environment variable for the user as well as the system

Click new > paste the path copied before



Check whether Kubectl is installed

```

Windows PowerShell

auth           Inspect authorization
debug          Create debugging sessions for troubleshooting workloads and nodes

Advanced Commands:
diff           Diff the live version against a would-be applied version
apply          Apply a configuration to a resource by file name or stdin
patch          Update fields of a resource
replace        Replace a resource by file name or stdin
wait           Experimental: Wait for a specific condition on one or many resources
kustomize      Build a kustomization target from a directory or URL.

Settings Commands:
label          Update the labels on a resource
annotate       Update the annotations on a resource
completion    Output shell completion code for the specified shell (bash, zsh, fish, or powershell)

Other Commands:
alpha          Commands for features in alpha
api-resources Print the supported API resources on the server
api-versions  Print the supported API versions on the server, in the form of "group/version"
config         Modify kubeconfig files
plugin         Provides utilities for interacting with plugins
version        Print the client and server version information

Usage:

```

Open Docker and copy the line below and paste in the command prompt



```

Windows PowerShell

options        Show a list of global command-line options (applies to all commands).
license        Outputs the licenses of dependencies to a directory

Other Commands:
completion     Generate command completion for a shell

Use "minikube <command> --help" for more information about a given command.
PS C:\Users\theva> docker run -d -p 80:80 docker/getting-started
Unable to find image 'docker/getting-started:latest' locally
latest: Pulling from docker/getting-started
c158987b0551: Pull complete
1e35f6679fab: Pull complete
cb9626c74200: Pull complete
b634b6ace34: Pull complete
f1d1c9928c82: Pull complete
9b6f639ec6ea: Pull complete
ee6d3549ec8: Pull complete
def978bed4f4: Pull complete
277dd14911d: Pull complete
Digest: sha256:aab880b9692678146332b3218f73789513ddf9e4273306b491250cb86fc9499
Status: Downloaded newer image for docker/getting-started:latest
f2516de348ala637736al14a185581eaff40b03666a9d17fed3cde5a623e8873

```

Pull the images in minikube

```

ps C:\Users\theva> minikube start --vm-driver=docker
● minikube v1.28.0 on Microsoft Windows 11 Home Single Language 10.0.22000 Build 22000
  Using the docker driver based on user configuration
  Starting control plane node minikube in cluster minikube
  Pulling base image ...
  Downloading Kubernetes v1.25.3 preload ...
  > preloaded-images-k8s-v18-v1...: 798.84 KiB / 385.44 MiB [ ] 0.20% ? p/s ?]

```

Check for the container status below

The image shows two screenshots of the Docker Desktop application interface.

Containers View:

NAME	IMAGE	STATUS	PORT(S)	STARTED	ACTIONS
practical_pascal f2516de348a1	docker/getting-started:latest	Running	80:80 ↗ 50957:22 ↗ 50958:2376 ↗ 50959:32443 ↗ 50960:5000 ↗ 50961:8443 ↗	59 minutes ago	⋮
minikube c3f5b4730f13	gcr.io/k8s-minikube/kicbase:v0	Running	50957:22 ↗ 50958:2376 ↗ 50959:32443 ↗ 50960:5000 ↗ 50961:8443 ↗	46 minutes ago	⋮

Images View:

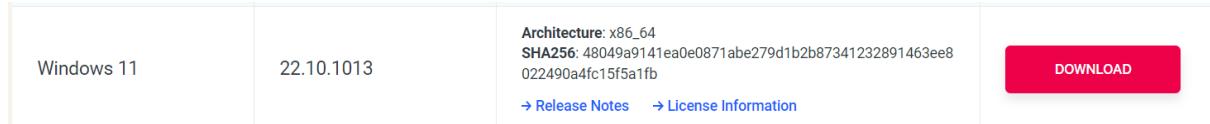
NAME	TAG	STATUS	CREATED	SIZE	ACTIONS
docker/getting-started 992c2ddee3ba	latest	In use	1 day ago	46.96 MB	⋮
gcr.io/k8s-minikube/kicbase 866c1fe4cf2	v0.0.36	In use	about 2 months ago	1.11 GB	⋮

Practical No. 9

Aim: Configuring and deploying VMs/Dockers using Chef/Puppet Automation tool

Implementation:

1. To install Chef workstation, search on google for ‘chef workstation windows setup’ and select the first link
2. Select the windows downloads page and download the most stable version of the installer



3. Run the downloaded MSI installer and once finished, open the cw powershell and type ‘chef’ on your cmd to verify

```
PS C:\Users\UDCS\Desktop> chef
The Chef command line tool for managing your infrastructure from your workstation.
Docs: https://docs.chef.io/workstation/
Patents: https://www.chef.io/patents

Usage:
  chef [command]

Available Commands:
  capture           Capture a node's state into a local chef-repo
  clean-policy-cookbooks Delete unused Policyfile cookbooks on the Chef Infra Server
  clean-policy-revisions Delete unused policy revisions on the Chef Infra Server
  completion        Generate the autocompletion script for the specified shell
  delete-policy     Delete all revisions of POLICY_NAME policy on the Chef Infra Server
  delete-policy-group Delete a policy group on Chef Infra Server
  describe-cookbook Prints cookbook checksum information for the cookbook at COOKBOOK_PATH
  diff              Generate an itemized diff of two policyfile lock documents
  env               Prints environment variables used by Chef Workstation
  exec              Runs COMMAND in the context of Chef Workstation
  export             Export a policy lock as a Chef Infra Client code repository
  gem               Runs the 'gem' command in the context of Chef Workstation's Ruby
  generate           Generate a new repository, cookbook, or other component
  help              Help about any command
  install            Install cookbooks from a policyfile and generate a locked cookbook set
  push               Push a local Policyfile lock to a policy group on the Chef Infra Server
  push-archive       Push a policy archive to a policy group on the Chef Infra Server
  report             Generate reports from a Chef Infra Server
  shell-init         Set shell context to the Chef Workstation environment
  show-policy        Show policyfile objects on the Chef Infra Server
  supermarket       Show cookbook objects on the Chef Infra Server
  c Supermarket     chef supermarket subcommand is used to interact with cookbooks that are located in on
  undelete          Undo a delete command
  update             Updates a Policyfile.lock.json with the latest run_list and cookbooks

Flags:
  --chef-license ACCEPTANCE  Accept product license, where ACCEPTANCE is one of 'accept', 'accept-no-persi
  accept-silent'
  -c   --config CONFIG_FILE_PATH  Read configuration from CONFIG_FILE_PATH
```

4. Enter the following command
> Chef generate cookbook admin

```

    └── pycache
PS C:\Users\UDCS\Desktop> chef generate cookbook admin
Generating cookbook admin
- Ensuring correct cookbook content
- Committing cookbook files to git

=====
Error executing action run on resource 'execute[git-commit-new-files]'

Mixlib::Shellout::ShellCommandFailed
-----
Expected process to exit with [0], but received '128'
---- Begin output of git commit -m "Add generated cookbook content" ----
STDOUT:
STDERR: Author identity unknown
*** Please tell me who you are.

Run
git config --global user.email "you@example.com"
git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'UDCS@DESKTOP-588IHIF.(none)')
---- End output of git commit -m "Add generated cookbook content" ----
Ran git commit -m "Add generated cookbook content" returned 128

Resource Declaration:
# In C:/opscode/chef-workstation/embedded/lib/ruby/gems/3.0.0/gems/chef-cl-5.6.1/lib/chef-cl,

```

5. Type tree once done to see your file structure

```

PS C:\Users\UDCS\Desktop> tree
Folder PATH listing
volume serial number is 1A80-EB15
C:.
├── .ipynb_checkpoints
├── AAkIF
├── crypt
└── demo
    ├── .kitchen
    │   └── logs
    ├── compliance
    │   ├── inputs
    │   └── profiles
    ├── waivers
    ├── recipes
    ├── test
    │   └── integration
    │       └── default
    ├── image processing
    ├── IoT practical required libs
    │   └── Arduino UNO Library for Proteus
    ├── Proteus.Pro.8.13.SP0.31525
    └── saurabh
        ├── .kitchen
        │   └── logs
        ├── compliance
        │   ├── inputs
        │   └── profiles
        ├── waivers
        ├── recipes
        └── test
            └── integration

```

6. Now type the following commands
>chef gem install kitchen-docker
7. Go to the directory of the admin cookbook and edit the kitchen.yml file as follows
Driver:
Name: docker
Transport:
Name: docker

Platforms:

```
-name: exec
Driver:
    Name: exec
    -name:exec
```

8. Now type the following commands

```
>kitchen create
>kitchen list
>kitchen converge
>kitchen list
```

```
PS C:\Users\UDCS\Desktop\admin> kitchen create
----> Starting Test Kitchen (v3.3.2)
----> Creating <default-exec>...
    Finished creating <default-exec> (0m0.00s).
----> Creating <default-exec>...
    Finished creating <default-exec> (0m0.00s).
----> Test Kitchen is finished. (0m3.23s)
PS C:\Users\UDCS\Desktop\admin> kitchen list
Instance   Driver  Provisioner  Verifier  Transport  Last Action  Last Error
default-exec Exec    ChefInfra   Inspec    Exec      Created     <None>
default-exec Exec    ChefInfra   Inspec    Exec      Created     <None>
PS C:\Users\UDCS\Desktop\admin> kitchen converge
----> Starting Test Kitchen (v3.3.2)
----> Converging <default-exec>...
```

```
PS C:\Users\theva\Desktop\prac10> kitchen list
C:/Users/theva/AppData/Local/chef/gem/ruby/3.0.0/gems/kitchen-docker-2.13.0/lib/docker/version.rb:11: warning: initialized constant Docker::VERSION
C:/opscode/chef-workstation/embedded/lib/ruby/gems/3.0.0/gems/docker-api-2.2.0/lib/docker/version.rb:11: warning: previous definition of VERSION was here
Instance   Driver  Provisioner  Verifier  Transport  Last Action  Last Error
default-exec Exec    ChefInfra   Inspec    Exec      Converged   <None>
default-exec Exec    ChefInfra   Inspec    Exec      Converged   <None>
```