**Tutorial No. 4**

**Problem statement:**

Implement an application for the TPO at VJTI forplacements implementing various activites that place like internship, placement student info. etc.

**Design Assumptions:**

**Microservices Architecture**

Microservices - also known as the microservice architecture - is an architectural style that structures an application as a collection of loosely coupled services, which implement business capabilities. The microservice architecture enables the continuous delivery/deployment of large, complex applications. It also enables an organization to evolve its technology stack.

The microservice architecture is not a silver bullet. It has several drawbacks. Moreover, when using this architecture there are numerous issues that you must address. The microservice architecture pattern language is a collection of patterns for applying the microservice architecture. It has two goals:

1. The pattern language enables you to decide whether microservices are a good fit for your application.
2. The pattern language enables you to use the microservice architecture successfully.

Flask is a micro [web framework](https://en.wikipedia.org/wiki/Web_framework) written in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) and based on the Werkzeug toolkit and [Jinja2](https://en.wikipedia.org/wiki/Jinja_(template_engine)) template engine. It is [BSD licensed](https://en.wikipedia.org/wiki/BSD_licenses).

The latest stable version of Flask is 0.12.2 as of May 2017. Applications that use the Flask framework include [Pinterest](https://en.wikipedia.org/wiki/Pinterest" \o "Pinterest), [LinkedIn](https://en.wikipedia.org/wiki/LinkedIn), and the community web page for Flask itself.

Flask is called a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program.[[7]](https://en.wikipedia.org/wiki/Flask_(web_framework)#cite_note-7)

The route() decorator in Flask is used to bind URL to a function. For example −

@app.route(‘/hello’)

def hello\_world():

return ‘hello world’

Here, URL ‘/hello’ rule is bound to the hello\_world() function. As a result, if a user visits http://localhost:5000/hello URL, the output of the hello\_world() function will be rendered in the browser.

The add\_url\_rule() function of an application object is also available to bind a URL with a function as in the above example, route() is used.

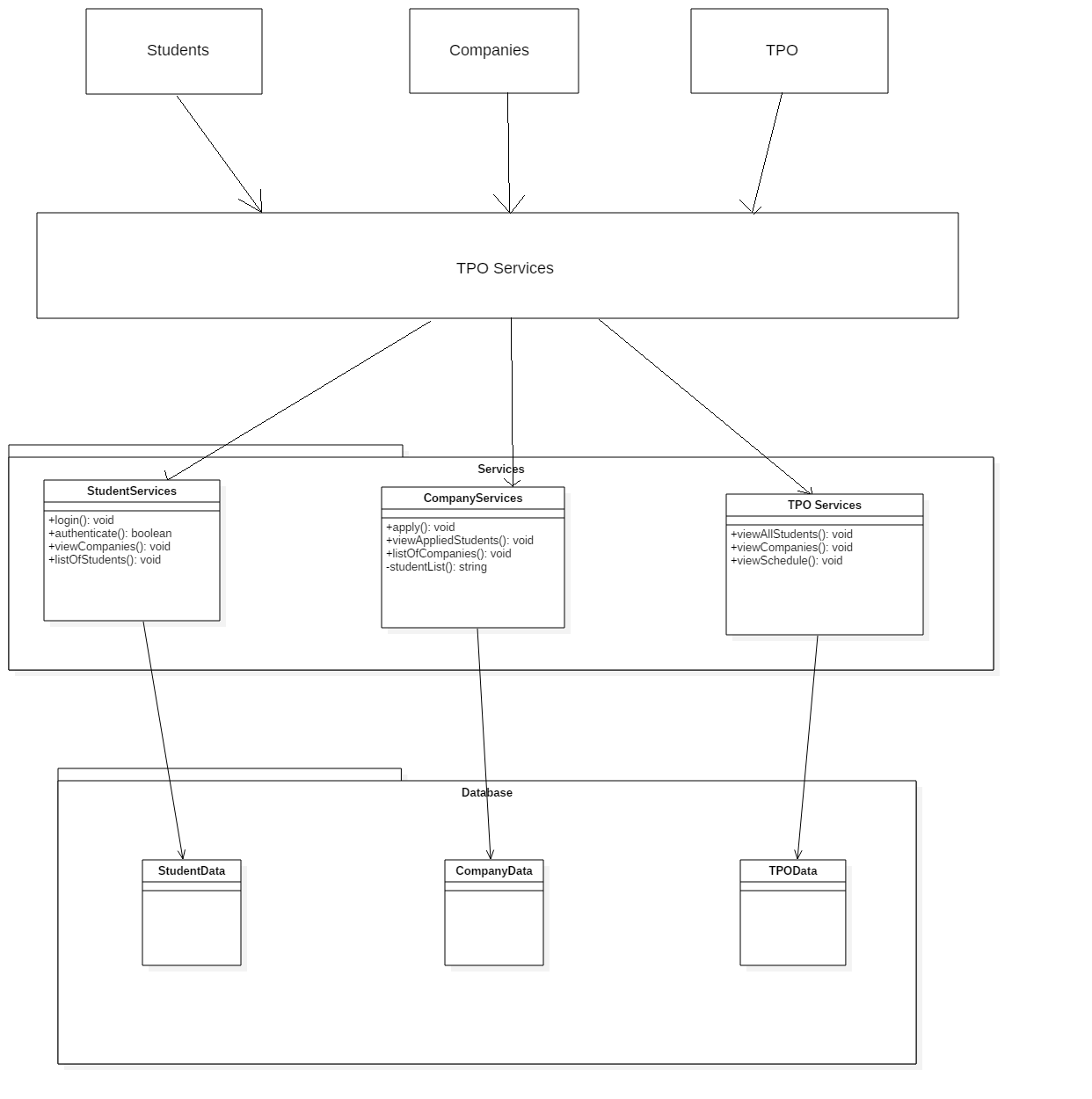
A decorator’s purpose is also served by the following representation −

def hello\_world():

return ‘hello world’

app.add\_url\_rule(‘/’, ‘hello’, hello\_world)

**Design Diagrams:**



**Code:**

**Controller\_calc.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Controller;

import Model.Model\_calc;

import View.View\_calc;

import calculator.\*;

import java.util.Scanner;

/\*

\* @author Jatin

\*/

public class Controller\_calc extends Thread {

private Model\_calc m;

private View\_calc v;

private Calculator c;

private volatile String c\_input = "";

private Scanner scn = new Scanner(System.in);

public Controller\_calc(Model\_calc m, View\_calc v){

this.m = m;

this.v = v;

c = new Calculator();

}

public String get\_input(){

while(v.a!=true)

{

if(!"".equals(c\_input)){

String to\_send = c\_input;

c\_input="";

return to\_send;

}

}

if(v.a==true){

v.a = false;

return v.s1;

}

return null; //will never execute

}

public void update\_model(String st, String ans){

m.update\_data(st,ans);

}

public String calculate\_result(){

return c.evaluate(m.export\_data());

}

public void display\_result(){

v.view\_jframe(m.export\_result());

v.view\_cmdline(m.export\_data(),m.export\_result());

}

@Override

public void run() {

c\_input = scn.next();

v.b=true;

}

}

**main.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Main;

import Model.Model\_calc;

import View.View\_calc;

import Controller.Controller\_calc;

/\*\*

\*

\* @author Jatin

\*/

public class Main {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

Controller\_calc c = new Controller\_calc(new Model\_calc(), new View\_calc());

c.start();

String st ="";

st = c.get\_input();

while(true){

String re = "0";

c.update\_model(st, re);

re = c.calculate\_result();

c.update\_model(st, re);

c.display\_result();

st = c.get\_input();

}

}

}

**Model\_calc.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Model;

/\*\*

\*

\* @author Jatin

\*/

public class Model\_calc {

private static String expr;

private static String ans;

public void update\_data(String st, String result){

expr = st;

ans = result;

}

public String export\_data(){

return expr;

}

public String export\_result(){

return ans;

}

}

**View\_calc.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package View;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

/\*\*

\*

\* @author Jatin

\*/

public class View\_calc implements ActionListener{

JTextField t;

JButton dec;

JButton equal;

JButton zero;

JButton one;

JButton two;

JButton three;

JButton four;

JButton five;

JButton six;

JButton seven;

JButton eight;

JButton nine;

JButton plus;

JButton minus;

JButton multiply;

JButton divide;

JButton clear;

private JFrame th;

public boolean a=false;

public boolean b=true;

public String s1;

public View\_calc() {

th = new JFrame("Basic Math calculator");

t = new JTextField();

t.setBounds(45,45,245,40);

one = new JButton("1");

one.setBounds(45,110,50,50);

four = new JButton("4");

four.setBounds(45,175,50,50);

seven = new JButton("7");

seven.setBounds(45,240,50,50);

dec = new JButton(".");

dec.setBounds(45,305,50,50);

two = new JButton("2");

two.setBounds(110,110,50,50);

five = new JButton("5");

five.setBounds(110,175,50,50);

eight = new JButton("8");

eight.setBounds(110,240,50,50);

zero = new JButton("0");

zero.setBounds(110,305,50,50);

three = new JButton("3");

three.setBounds(175,110,50,50);

six = new JButton("6");

six.setBounds(175,175,50,50);

nine = new JButton("9");

nine.setBounds(175,240,50,50);

equal = new JButton("=");

equal.setBounds(175,305,50,50);

plus = new JButton("+");

plus.setBounds(240,110,50,50);

minus = new JButton("-");

minus.setBounds(240,175,50,50);

multiply = new JButton("\*");

multiply.setBounds(240,240,50,50);

divide = new JButton("/");

divide.setBounds(240,305,50,50);

clear = new JButton("clear");

clear.setBounds(110,370,100,50);

th.setSize(350, 470);

th.add(one);

th.add(two);

th.add(three);

th.add(four);

th.add(five);

th.add(six);

th.add(seven);

th.add(eight);

th.add(nine);

th.add(dec);

th.add(zero);

th.add(equal);

th.add(plus);

th.add(multiply);

th.add(divide);

th.add(minus);

th.add(t);

th.add(clear);

th.setLayout(null);

th.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

th.setVisible(true);

one.addActionListener(this);

two.addActionListener(this);

three.addActionListener(this);

four.addActionListener(this);

five.addActionListener(this);

six.addActionListener(this);

seven.addActionListener(this);

eight.addActionListener(this);

nine.addActionListener(this);

zero.addActionListener(this);

equal.addActionListener(this);

plus.addActionListener(this);

multiply.addActionListener(this);

divide.addActionListener(this);

minus.addActionListener(this);

dec.addActionListener(this);

clear.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource()==one){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("1"));

}

if(e.getSource()==two){

{

if(this.b==true){

this.b=false;

t.setText("");

}t.setText(t.getText().concat("2"));

}

}

if(e.getSource()==three){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("3"));

}

if(e.getSource()==four){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("4"));

}

if(e.getSource()==five){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("5"));

}

if(e.getSource()==six){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("6"));

}

if(e.getSource()==seven){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("7"));

}

if(e.getSource()==eight){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("8"));

}

if(e.getSource()==nine){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("9"));

}

if(e.getSource()==zero){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("0"));

}

if(e.getSource()==dec){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("."));

}

if(e.getSource()==plus)

t.setText(t.getText().concat("+"));

if(e.getSource()==multiply)

t.setText(t.getText().concat("\*"));

if(e.getSource()==divide)

t.setText(t.getText().concat("/"));

if(e.getSource()==minus)

t.setText(t.getText().concat("-"));

if(e.getSource()==clear)

{

t.setText("");

a=false;

}

if(e.getSource()==equal)

{

this.s1=t.getText();

if("".equals(s1))

{

JOptionPane.showMessageDialog(th, "Please enter expression");

}

else {

this.a=true;

this.b=true;

}

}

}

public void view\_jframe(String ans){

t.setText(ans);

}

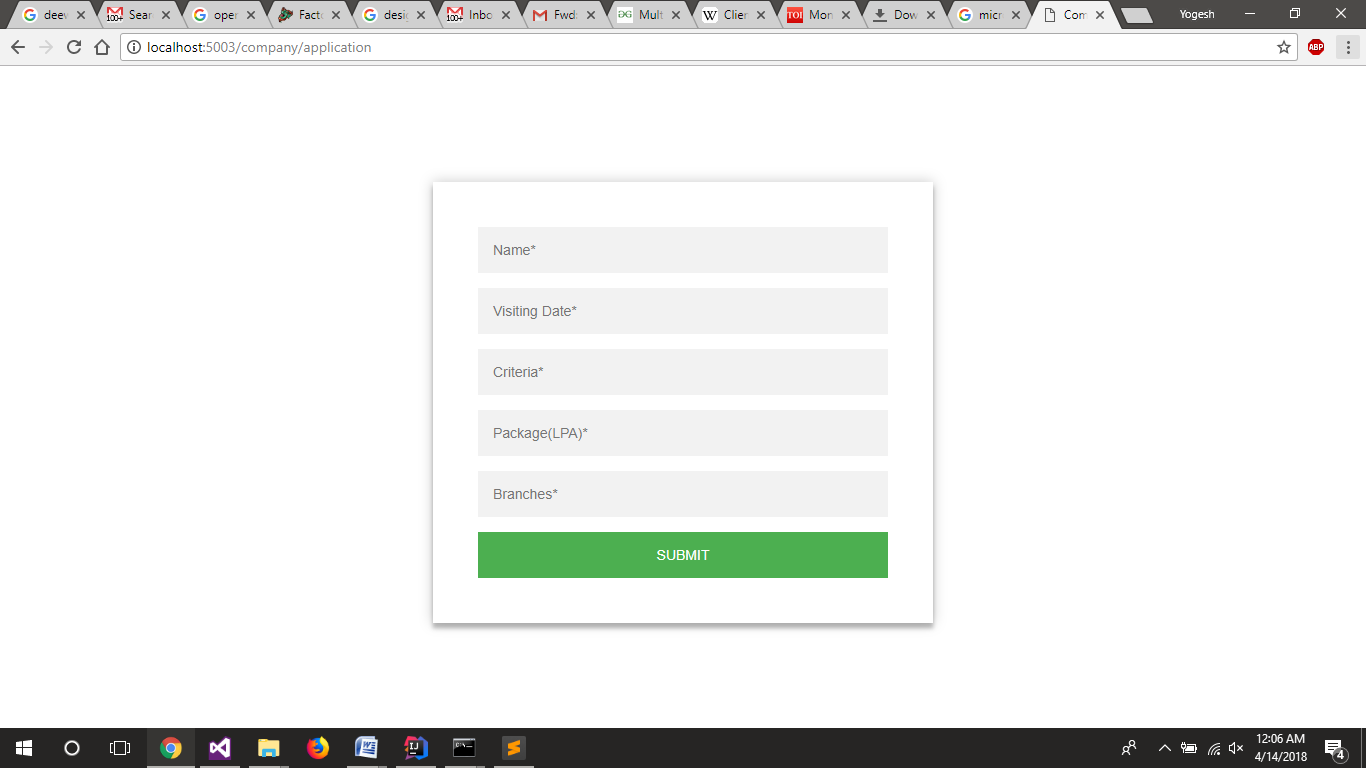
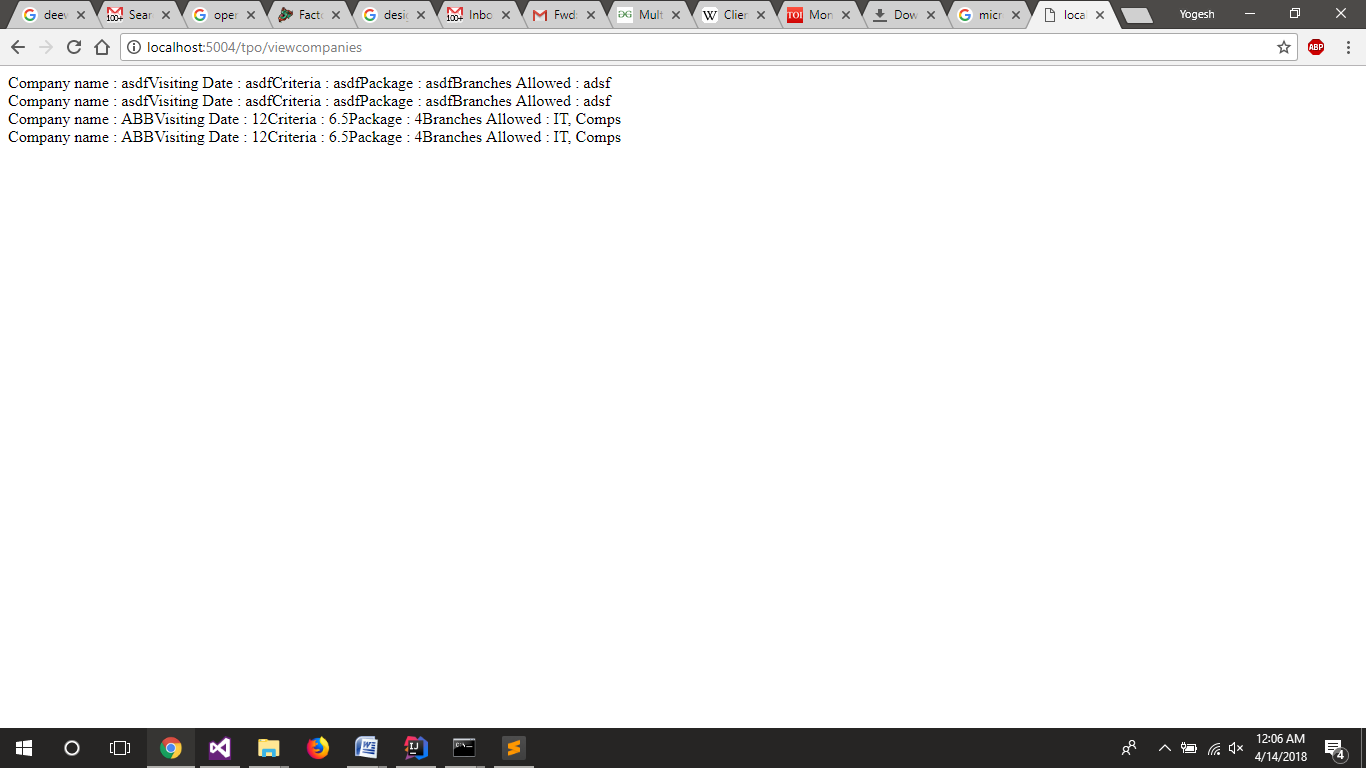
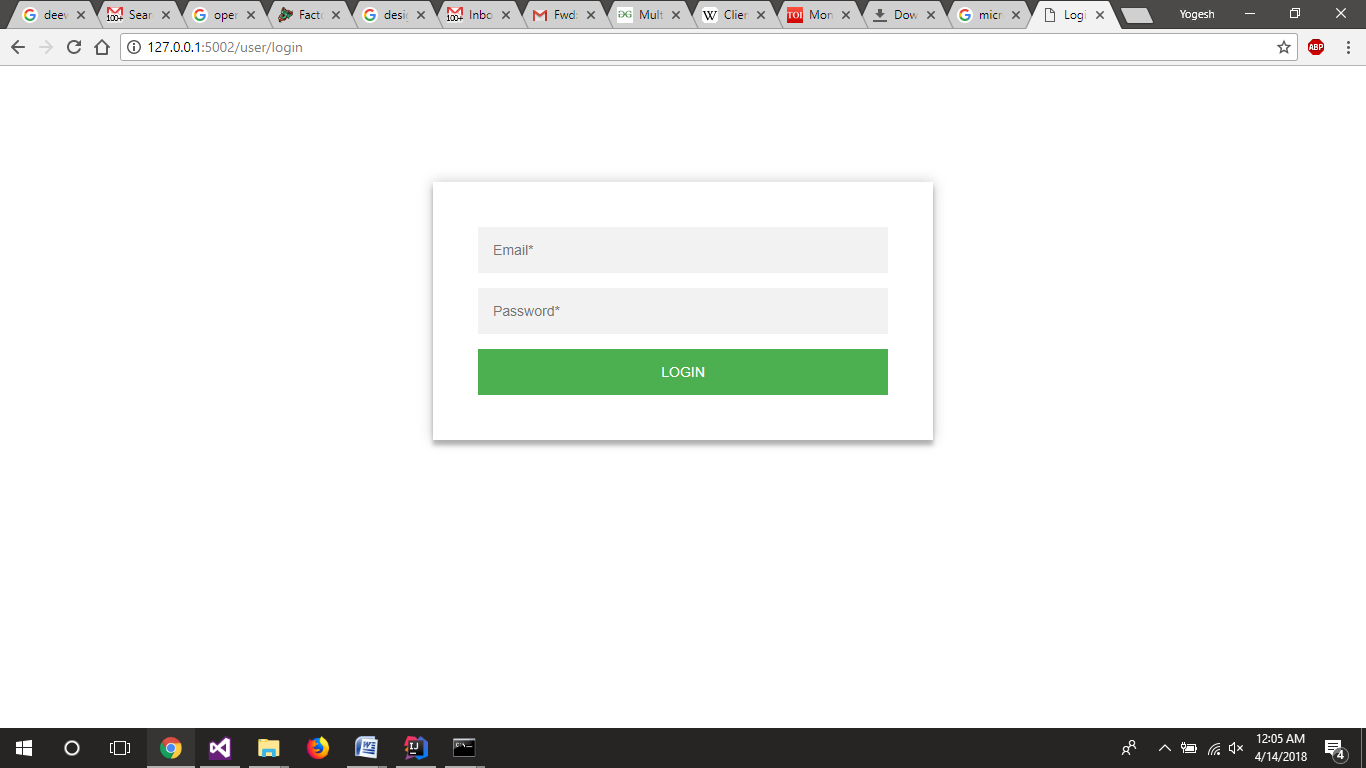
public void view\_cmdline(String exp, String ans){

System.out.println(exp+" = "+ans);

}

}

**Output**



**Observation:**

Thus the microservice architecture was implements using flask in python. While implementing the routing in flask had to be learnt in order to navigate the user to the correct web page and to navigate the request from users as well as other services to the correct service. Overall it was very easy once the routing was understood.