## DEPARTMENT OF INFORMATION TECHNOLOGY

**WEBLAB**

## LABORATORY MANUAL



**SIES GRADUATE SCHOOL OF TECHNOLOGY**

## PLOT 1-C, D & E, SECTOR V, NERUL, NAVI MUMBAI-400 706

**VISION OF THE INSTITUTE**

To be a center of excellence in Engineering Education committed towards Socio-Economic Advancement of the country.

**MISSION OF THE INSTITUTE**

1. To impart advanced knowledge in Engineering and Technology.
2. To transform young minds towards professional competence by inculcating values and developing skills.
3. To promote research and ensure continuous value addition among students and employees.
4. To strengthen association with industry, research organizations and alumni to enhance knowledge on current technologies.
5. To promote next generation technocracy and nurture entrepreneurial culture for social-economic growth.

**VISION OF THE DEPARTMENT**

To develop IT professionals for accomplishment of industrial &societal needs through quality education.

**MISSION OF THE DEPARTMENT**

1. To impart advanced knowledge and develop skills in InformationTechnology and allied fields.
2. To enhance professional competence by inculcating values and ethics.
3. To upgrade technical skills and also encourage research culture.
4. To extend industry and alumni association for knowledge enhancement.
5. To nurture entrepreneurial talent and contribute towards socio-economic growth.

## Lab Outcomes (LOs):

**At the end of the course the student should be able to:**

1. Students will be able to Understand open source tools for web analytics and semantic web apps development and deployment.
2. Students will be able to Understand the basic concepts of TypeScript for designing web applications.
3. Students will be able to Implement Single Page Applications using AngularJS Framework
4. Students will be able to Develop Rich Internet Applications using AJAX.
5. Students will be able to Create REST Web services using MongoDB.
6. Students will be able to Design web applications using Flask.

**Course Articulation Matrix: Mapping of Course Outcomes (COs) with Program Outcomes (POs)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course Title: WEB LAB | | | | | | | | | Semester: 6 | | | | | |
| Course Code: ITL602 | | | | | | | | | Year: 2022 | | | | | |
| **Course Outcomes (COs) / Program Outcomes (POs)** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **1**  **1** | **1**  **2** | **P S O 1** | **P S O 2** |
| 1. Students will be able to Understand open  source tools for web analytics and semantic web apps development and deployment. | - | - | 2 | 3 | 3 | 2 | - | 2 | - | - | - | 2 | 3 | 3 |
| 2. Students will be able to Understand the  basic concepts of TypeScript for designing web applications. | - | - | 2 |  | 3 | 2 | - | 2 | - | - | - | 2 | 3 | 3 |
| 3. Students will be able to Implement Single Page Applications using AngularJS Framework | - | - | 3 | 3 | 3 | 2 | - | 3 | - | - | - | 2 | 3 | 3 |
| 4. Students will be able to Develop Rich Internet Applications using AJAX. | - | - | 3 | 3 | 3 | 2 | - | 3 | - | - | - | 2 | 3 | 3 |
| 5. Students will be able to Create REST  Web services using MongoDB. | - | - | 3 | 3 | 3 | 2 | - | 3 | - | - | - | 2 | 3 | 3 |
| 6. Students will be able to Design web  applications using Flask. | - | - | 3 | 3 | 3 | 2 | - | 3 | - | - | - | 2 | 3 | 3 |

# CODE OF CONDUCT FOR THE LABORATORIES

* All the students must observe the Dress Code while in the laboratory.
* Sandals or open-toed shoes are NOT allowed.
* Food, drinks and smoking are NOT allowed.
* All bags must be left at the indicated place.
* The lab timetable must be strictly followed.
* Be PUNCTUAL for your laboratory sessions.
* Noise must be kept to minimum.
* Workspace must be kept clean and tidy all the time.
* Handle the systems and interfacing kits with care.
* All students are liable for any damage to the accessories due to negligence.
* All interfacing kits connecting cables must be RETURNED to the lab supervisor.
* Students are strictly PROHIBITED from taking out any items from the laboratory.
* Students are NOT allowed to work in the laboratory without the Lab supervisor.
* Report immediately to the Lab Supervisor of any malfunction of the accessories..

## Before leaving the lab

* Place the chairs/stools properly.
* Return the consumables, experiment kits/breadboard etc to the lab supervisor.

At the end of each laboratory session you must obtain the signature of the teacher along with the marks for the session out of 05 on the lab notebook and 05 for journal.

## Lab Reports

* Note that, although students are encouraged to collaborate during lab, each must individually prepare a report and submit.
* They must be organized, neat and legible.
* Your report should be complete, thorough and understandable.
* You should include a well-drawn and labeled engineering schematic for each circuit investigated.
* Your reports should follow the prescribed format, to give your report structure and to make sure that you address all important points.
* Graphics requiring- drawn straight lines should be done with a straight edge. Well drawn free-hand sketches are permissible for schematics.
* Space must be provided in the flow of your discussion for any tables or figures. Do not collect figures and drawings in a single appendix at the end of the report.
* Reports should be submitted within one week after completing a scheduled lab session.

## Presentation

* Experimental facts should always be given in the past tense.
* Discussions or remarks about the presentation of data should mainly be in the present tense.
* Discussion of results can be in both the present and past tenses, shifting back and forth from experimental facts to the presentation.
* Any specific conclusions or deductions should be expressed in the past tense.

# LIST OF EXPERIMENTS

|  |  |  |
| --- | --- | --- |
| **Exp No.** | **Name of the Experiment** | **Lab Outcomes** |
| 1. | To Study web analytics using open-source tool like Matomo, Google Analytics,AWStats | LO1 |
| 2. | 1. To Study Semantic Web Open-Source Tools OWL and ptotege/Sesame. 2. Demo of SPARQL using SPARQLWrapper -is a simple Python wrapper around a SPARQL service to remotely execute your queries. | LO1 |
| 3. | Write Type Script program -Hello World, Calculator | LO2 |
| 4. | Write TypeScript program using Inheritance concept | LO2 |
| 5. | Building a Simple Website using React with TypeScript | LO2 |
| 6. | Create a simple HTML “Hello World” Project using AngularJS Framework and apply ng-controller, ng-model and expressions. | LO3 |
| 7. | Create an application for like Students Record using AngularJS | LO3 |
| 8. | 1. Write a JavaScript program for a AJAX. 2. Write a program to use AJAX for user validation using and to show the result on the same page below the submit button 3. Design and develop small web application using AJAX, HTML and PHP | LO4 |
| 9. | Build a RESTful API using MongoDB. | LO5 |
| 10. | Design Feedback Form using Flask. | LO6 |
| 11. | Design Weather App using Flask. | LO6 |
| 12. | Design Portfolio Website using Flask. | LO6 |

## Experiment 1

**Aim**: To Study web analytics using open-source tool like Matomo/AWStats Theory:

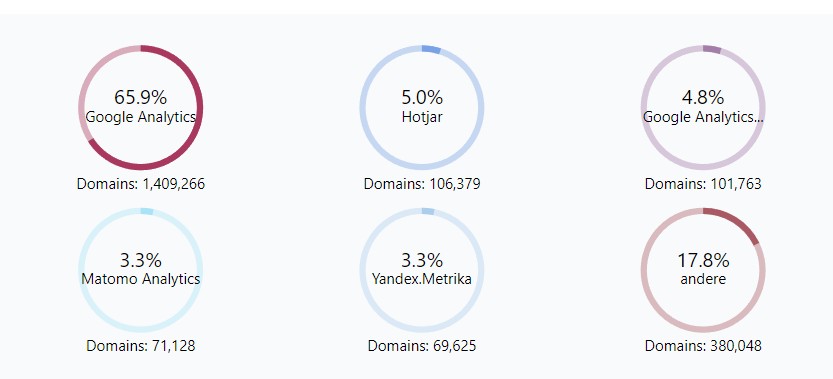
## Web analytics:

Web analytics is the collection, reporting, and analysis of website data. The focus is on identifying measures based on your organizational and user goals and using the website data to determine the success or failure of those goals and to drive strategy and improve the user's experience.

Major types of web analytics tools, some popular examples, and how to know when you need them.

* + Content analytics tools
  + Customer analytics tools
  + Usability (UX) analytics tools
  + A/B and multivariate testing tools
  + Social media analytics tools
  + SEO analytics tools
  + General enterprise analytics tools
  + Open source web analytics tools
  + Product analytics tools

Here are some of the top tools that you can use to gain more understanding about your website traffic.



Comparison of web analytics tools <https://www.business.com/images/content/53e/5260cc9786b9e2da0d11b/0-0-/>

AWStats:

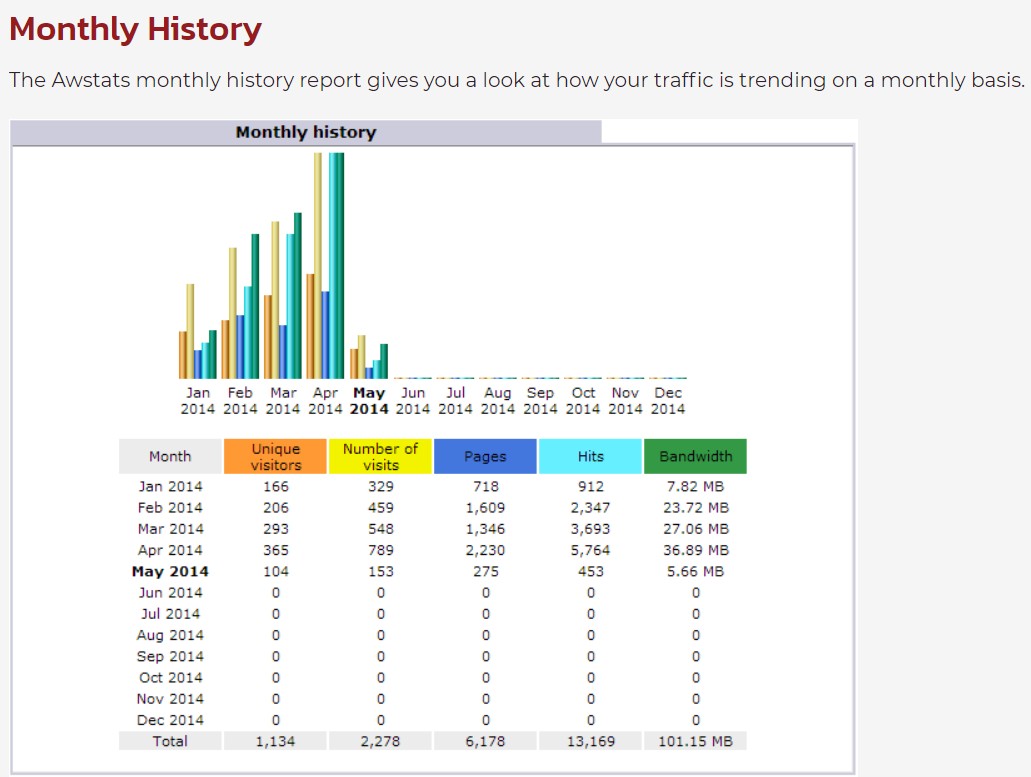
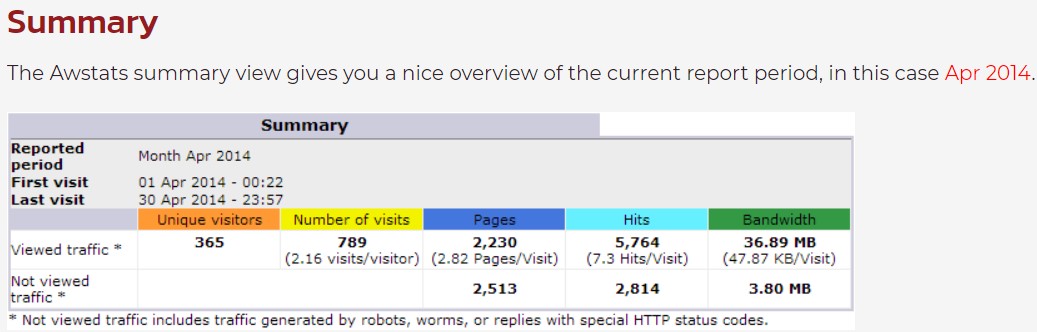
AWStats is a powerful tool that creates graphical reports from your website’s access logs. It’s great for taking a quick look at things such as visits to your site per day and the number of pages viewed on your website.

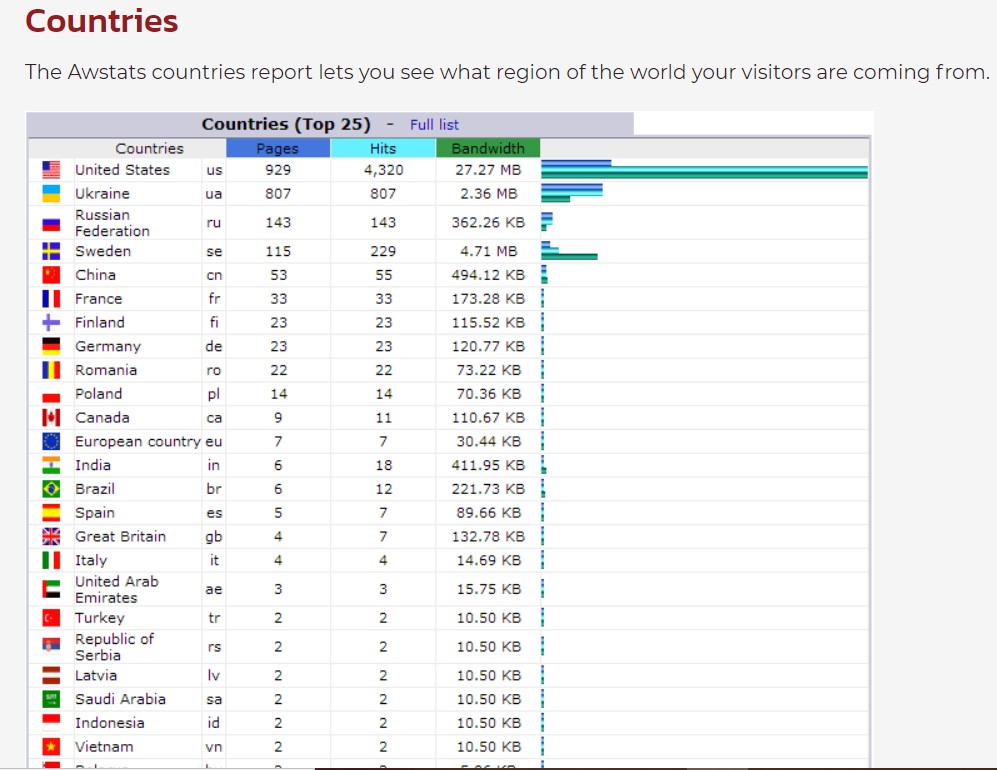
Here are some of the common terms that you’ll run across when viewing your Awstats reports:

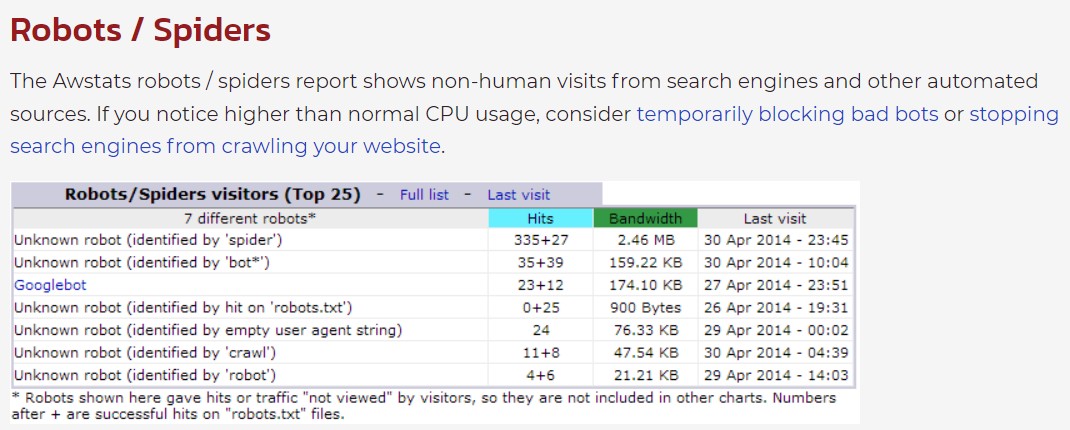
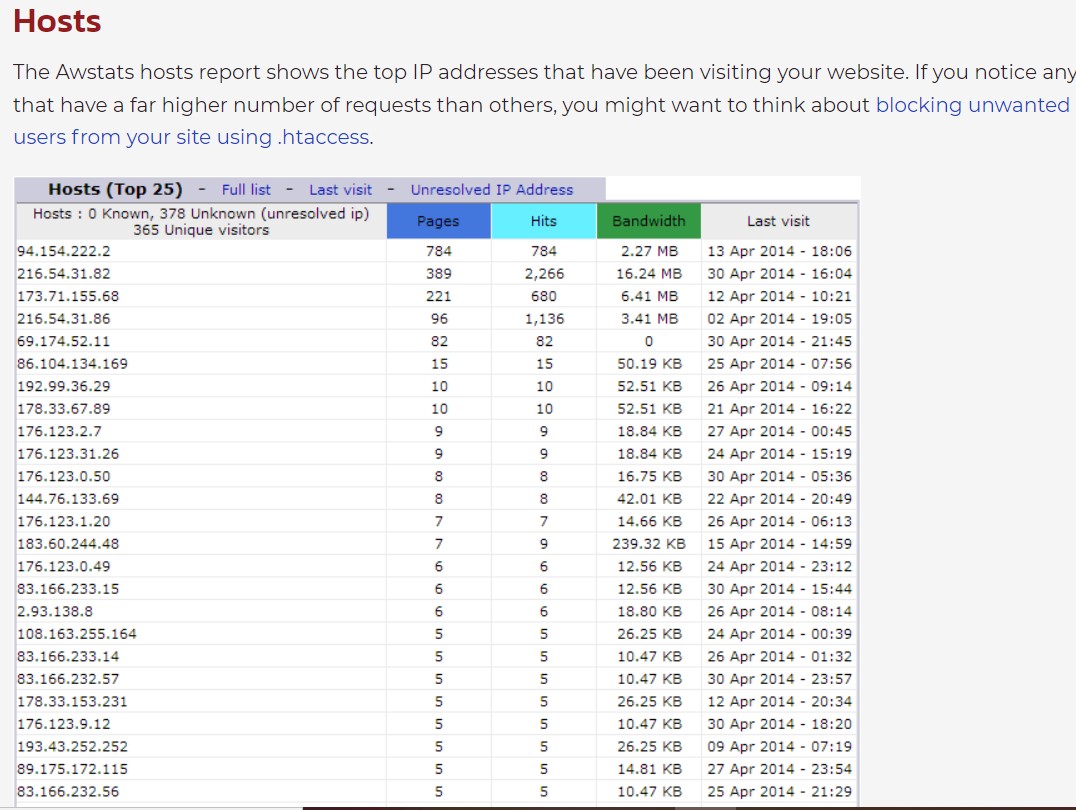
**Unique Visitors:** The unique amount of visitors you’ve had to your website. **Number of Visits:** The number of visits to your website made by all unique visitors. **Pages**: The number of full pages that have been viewed.

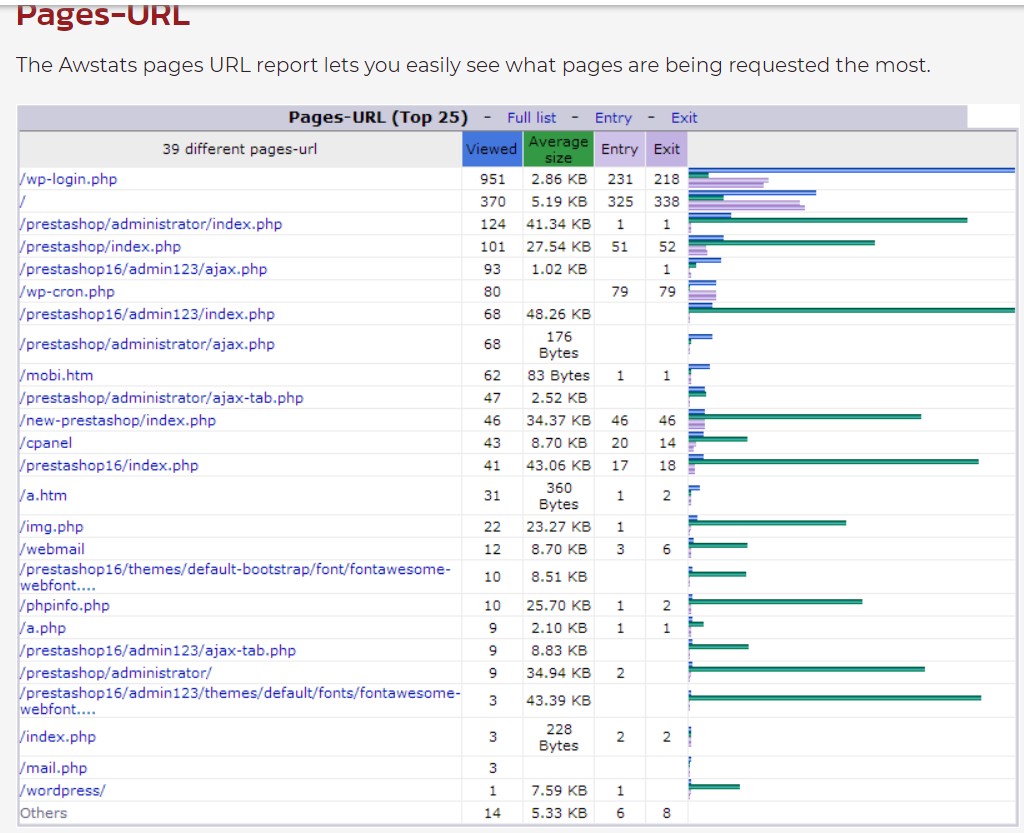
**Hits:** The total number of hits for resources such as images or JavaScript files.

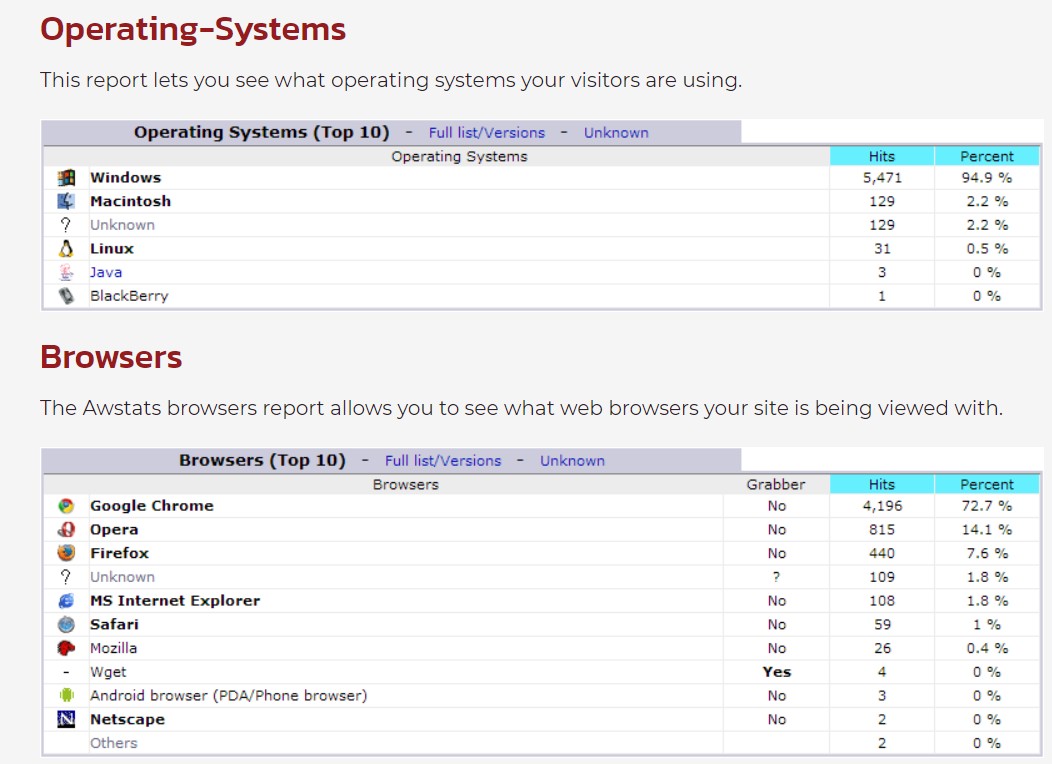
**Bandwidth:** The total amount of bandwidth consumed serving all of the website requests.











Links:

AWStats-<https://awstats.sourceforge.io/#DEMO> Matomo-<https://matomo.org/>

Conclusion: AwStats web analytics tool study experiment is successfully learnt and understood.

## Experiment 2 a

**Aim**: To Study Semantic Web Open-Source Tool protege

The Semantic Web is an evolving extension of the World Wide Web in which the semantics of information and services on the web is defined, making it possible for the web to understand and satisfy the requests of people and machines to use the Web content. It derives from W3C director Tim Berners -Lee vision of the Web as a universal medium for data ,information and knowledge exchange.

The semantic web addresses this shortcoming, using the descriptive technologies Resource Description Framework (RDF) and Web Ontology Language (OWL), and the data-centric, customizable Extensible Mark-up Language (XML). These technologies are combined in order to provide descriptions that supplement or replace the content of Web documents

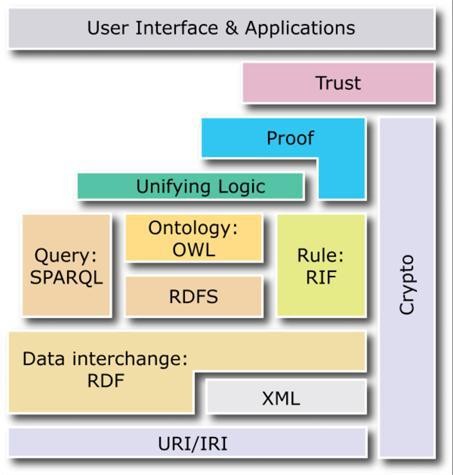
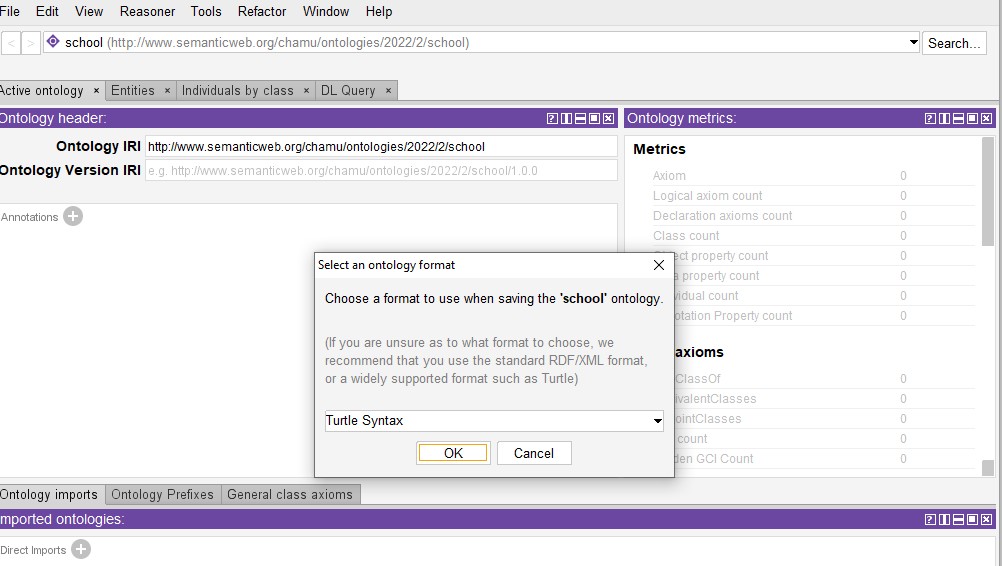


Fig:Semantic web stack

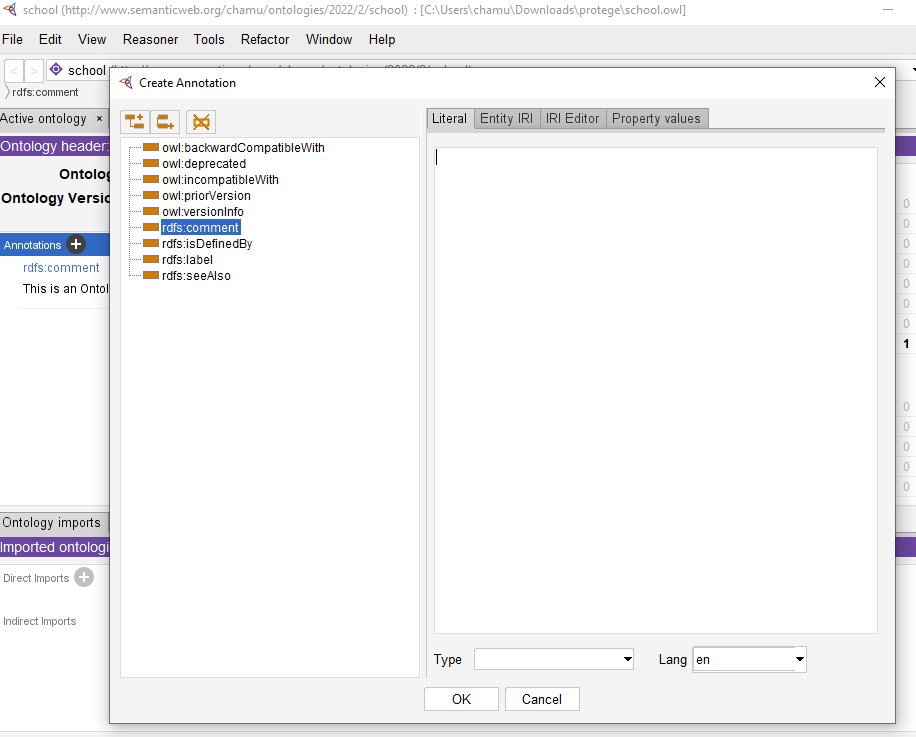
Web Ontology language: The W3C Web Ontology Language (OWL) is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things.

OWL tool Protégé:- Protégé is an open-source ontology editor and framework developed by the Stanford Center for Biomedical Informatics Research for building knowledge-based solutions. It provides a plug-and-play environment that makes it a flexible base for rapid prototyping. One can either download the desktop application or continue to use web-protégé

Download protégé from below link <https://protege.stanford.edu/products.php#desktop-protege> After install, provide name to file and save a as.owl extension

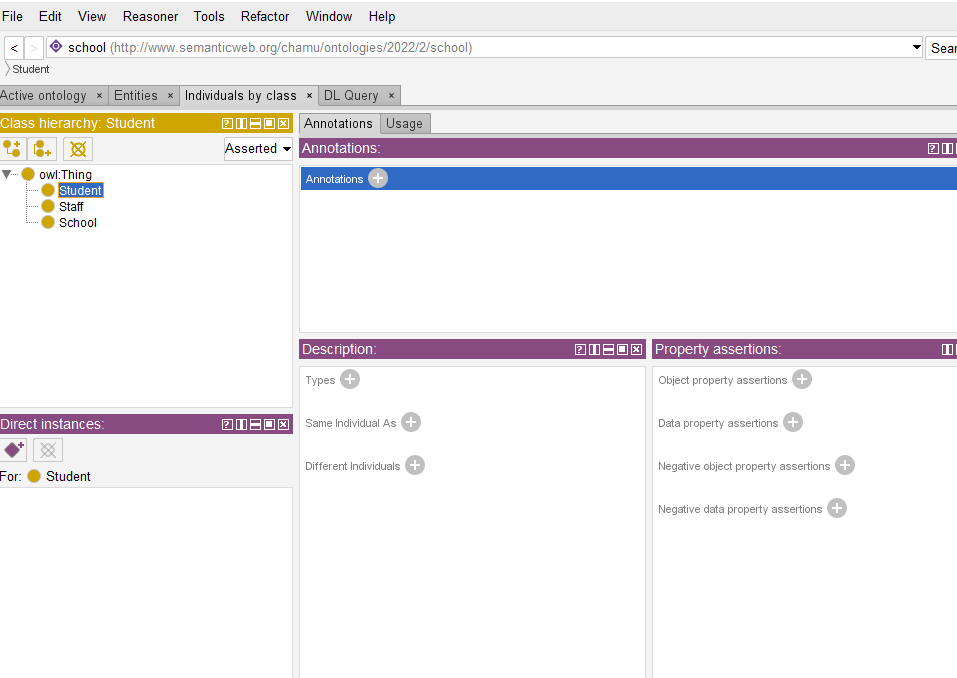


Add annotation

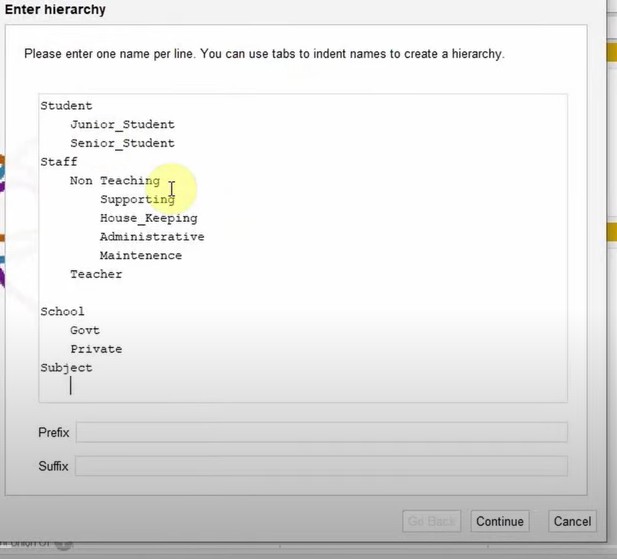


Class properties:

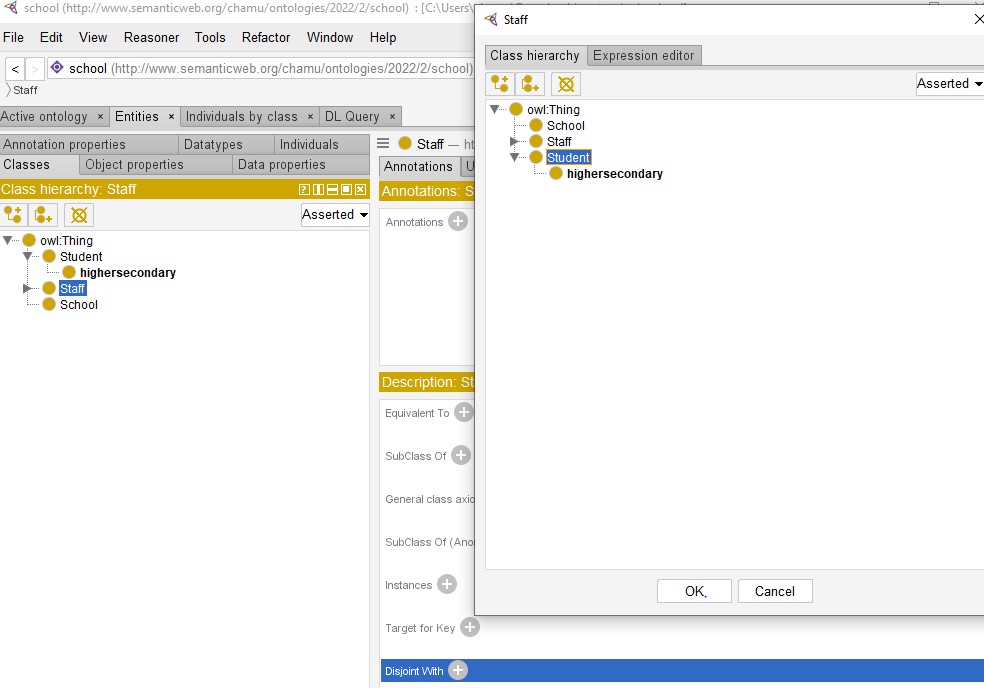
Create a class School, student and staff as subclass of thing



Create class and subclass by class hierarchy as below,Go to tools,select class hierarchy and enter hierarchy details as below.



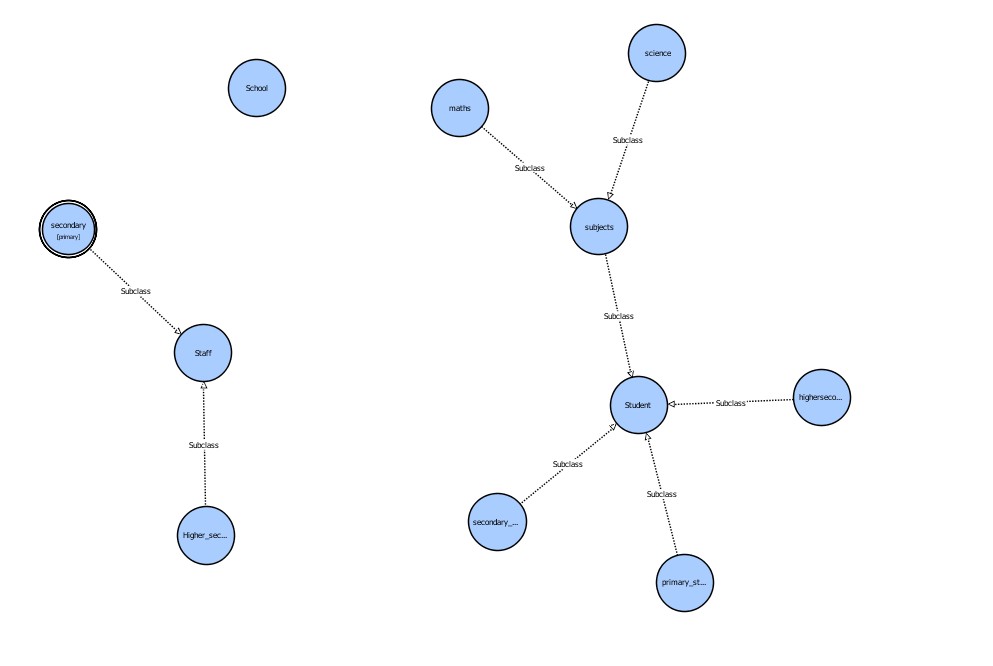
To do disjoint class select class and select option disjoint with .provide which class to be disjoint



To see visualization

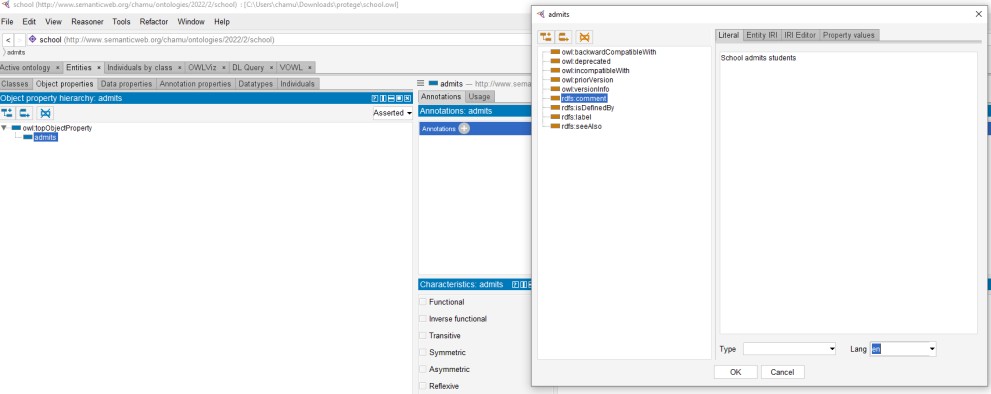
Goto windows->tabs->OWLviz

If visualization is not proper.download vowl plugin From <http://vowl.visualdataweb.org/protegevowl.html>

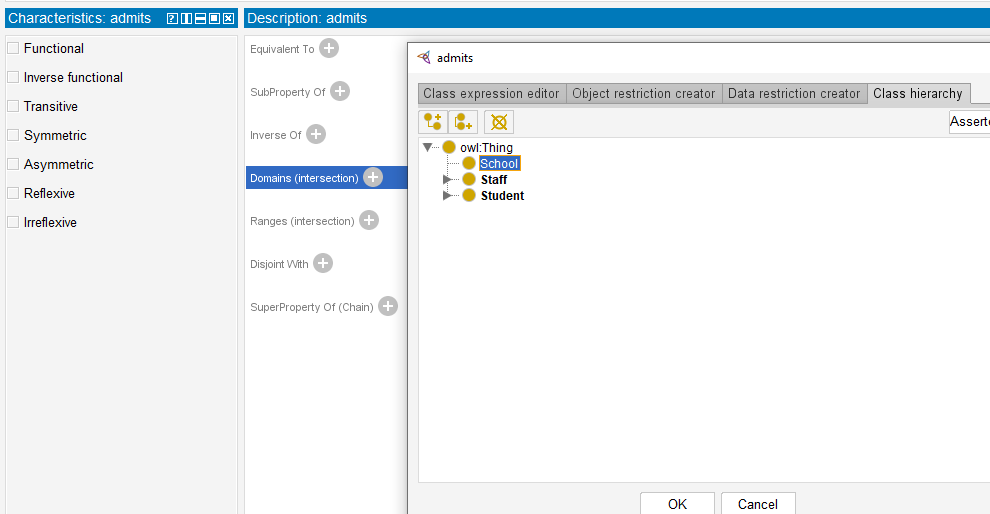


Object properties:

Create new subproperty as admits as below



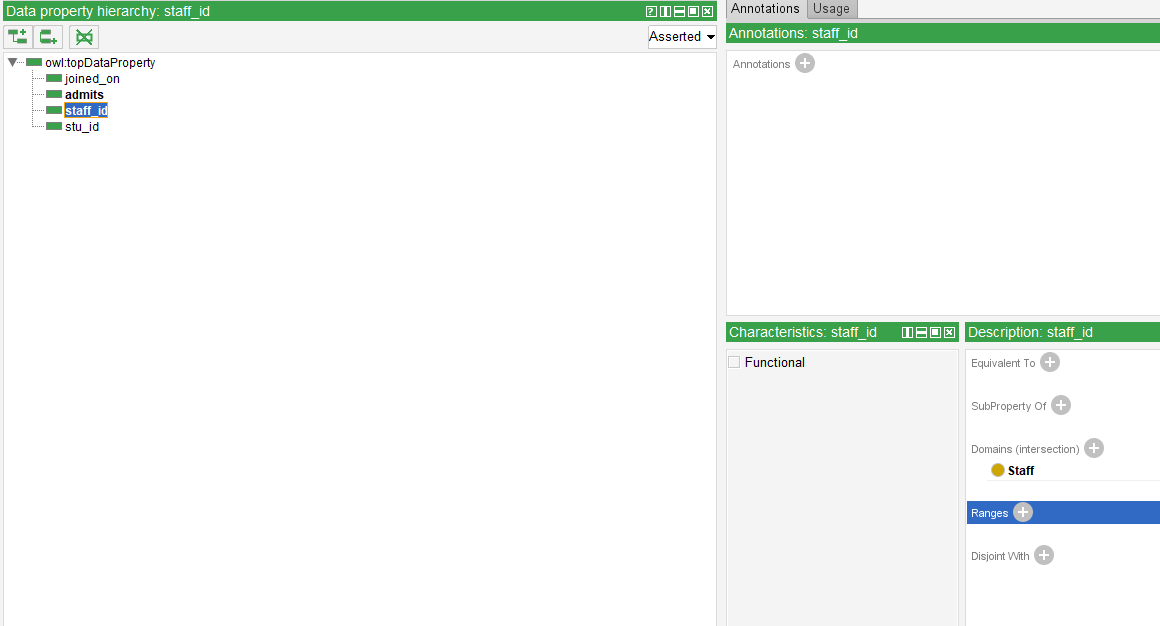
Add domain as school and range as student



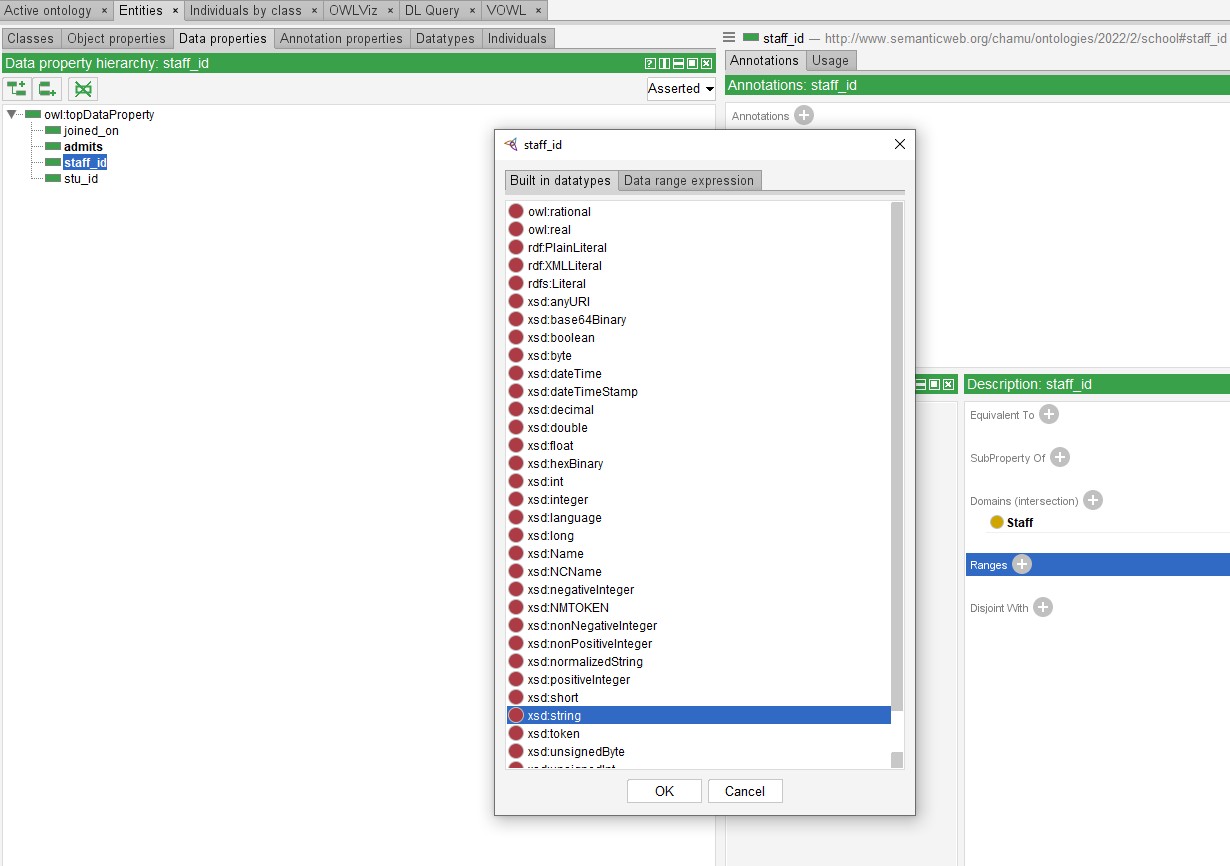
## Data Properties:

Click dataproperties tab

Add subproperty staff id, stuid, joinedon, admits as below

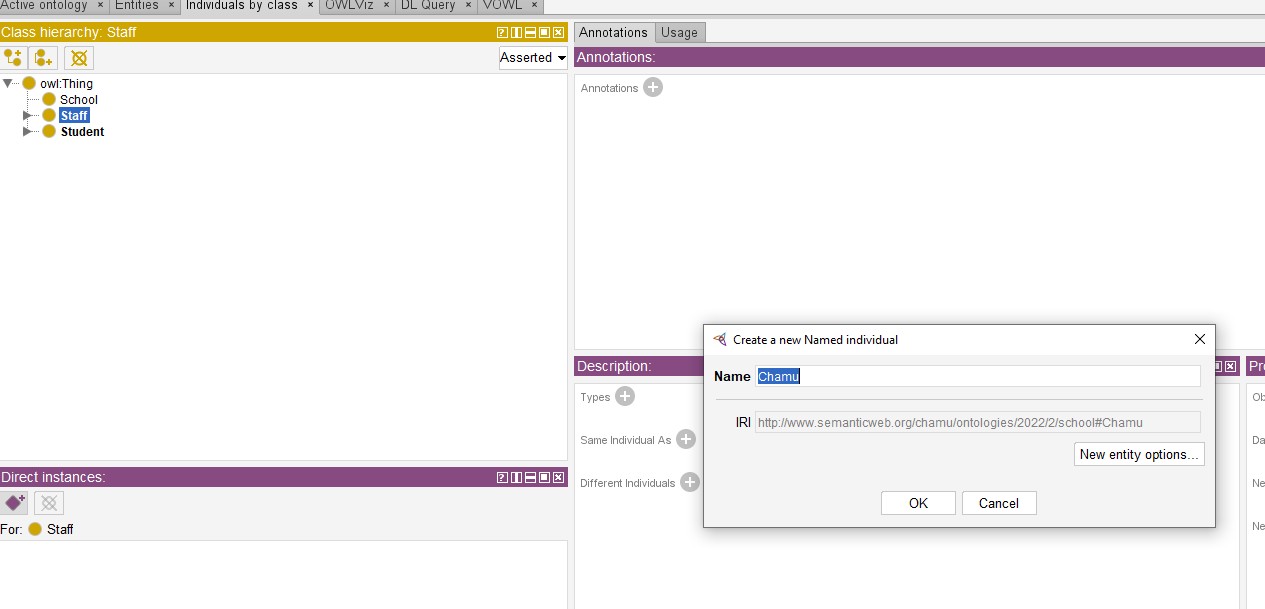


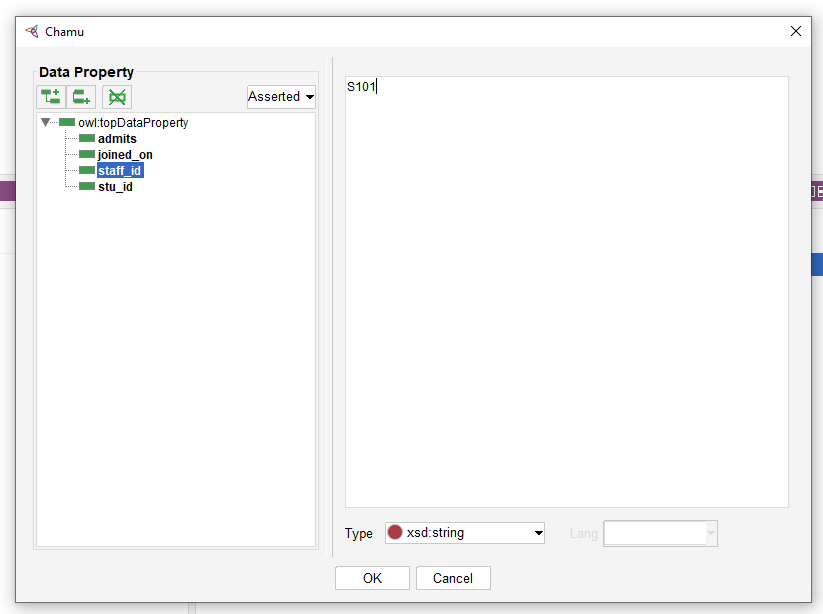
Provide domain of each data property and range as below

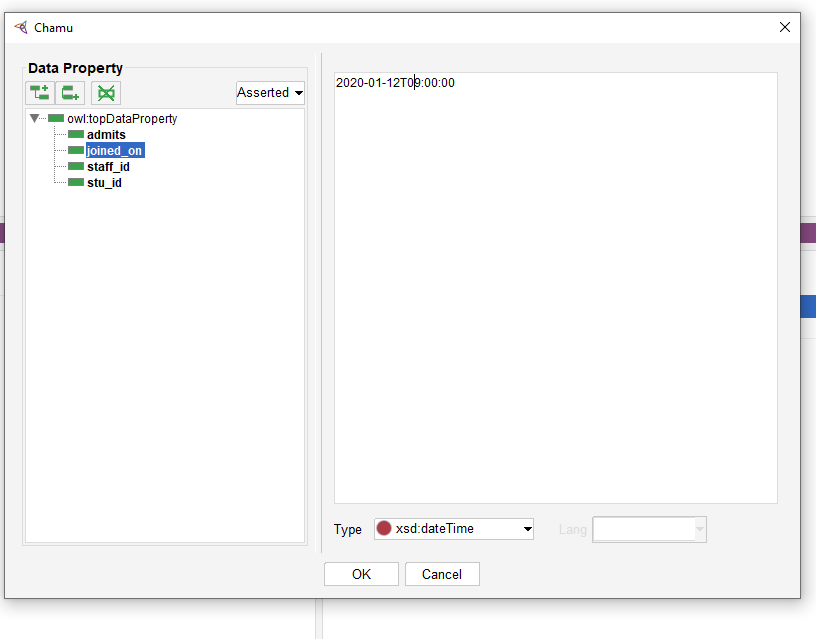


Individual by class: Select class first

Click individual by class Add name of individual







References:

https://medium.com/analytics-vidhya/prot%C3%A9g%C3%A9-d533f024087b <https://www.youtube.com/watch?v=LQ4iW3PO36E>

OWL-https:/[/www.youtube.com/watch](http://www.youtube.com/watch?v=JiGRVIQ9rks)?[v=JiGRVIQ9rks](http://www.youtube.com/watch?v=JiGRVIQ9rks)

Conclusion: Web ontology language visualization is implemented using protégé framework successfully.

## Experiment 2 b

**Aim**: Demo of SPARQL using SPARQLWrapper -is a simple Python wrapper around a SPARQL service to remotely execute your queries.

Theory:

SPARQL is the standard query language and protocol for Linked Open Data and RDF databases. Having been designed to query a great variety of data, it can efficiently extract information hidden in non-uniform data and stored in various formats and sources.

SPARQL has four types of queries. It can be used to:

ASK whether there is at least one match of the query pattern in the RDF graph data; SELECT all or some of those matches in a tabular form (including aggregation, sampling and pagination through OFFSET and LIMIT);

CONSTRUCT an RDF graph by substituting the variables in these matches in a set of triple templates;

DESCRIBE the matches found by constructing a relevant RDF graph.

## Querying DBpedia:

The DBpedia data set enables some astonishing queries against Wikipedia data to be answered. You can query the DBpedia data set online via a SPARQL endpoint (described on this page) and as Linked Data.

## Public SPARQL Endpoint

A public SPARQL endpoint for querying the DBpedia data set is at [http://dbpedia.org/sparql.](http://dbpedia.org/sparql) OpenLink Virtuoso serves as both the back-end database SPARQL query engine and the front- end HTTP/SPARQL server with an nginx overlay primarily to cache results for each submitted query string.The public endpoint does NOT include all available DBpedia data sets. The Loaded Datasets subsection below provides a list of all DBpedia data sets currently loaded into the public SPARQL endpoint.

* + You can run queries against DBpedia using:
  + the OpenLink Interactive SPARQL Query Builder (iSPARQL) at <http://dbpedia.org/isparql>
  + the SNORQL query explorer at <http://dbpedia.org/snorql>(does not work with Internet Explorer)
  + any other SPARQL-aware client tool

## SPARQLWrapper:

SPARQLWrapper is a simple Python wrapper around a SPARQL service to remotelly execute your queries. It helps in creating the query invokation and, possibly, convert the result into a more manageable format.

## How to use

You can use SPARQLWrapper either as a Python command line script or as a Python package. Command Line Script

To use as a command line script, you will need to install SPARQLWrapper and then a command line script called rqw (spaRQl Wrapper) will be available within the Python environment into which it is installed. run $ rql -h to see all the script's options.

Python package

ASK,SELECT,DESCRIBE like queries executed via SPARQLWrapper as a python package.

## Steps:

Use jupyter notebook to implement SPARQL using RDF library in python . Install RDF library and SPARQL Wrapper by below command

!pip install rdflib

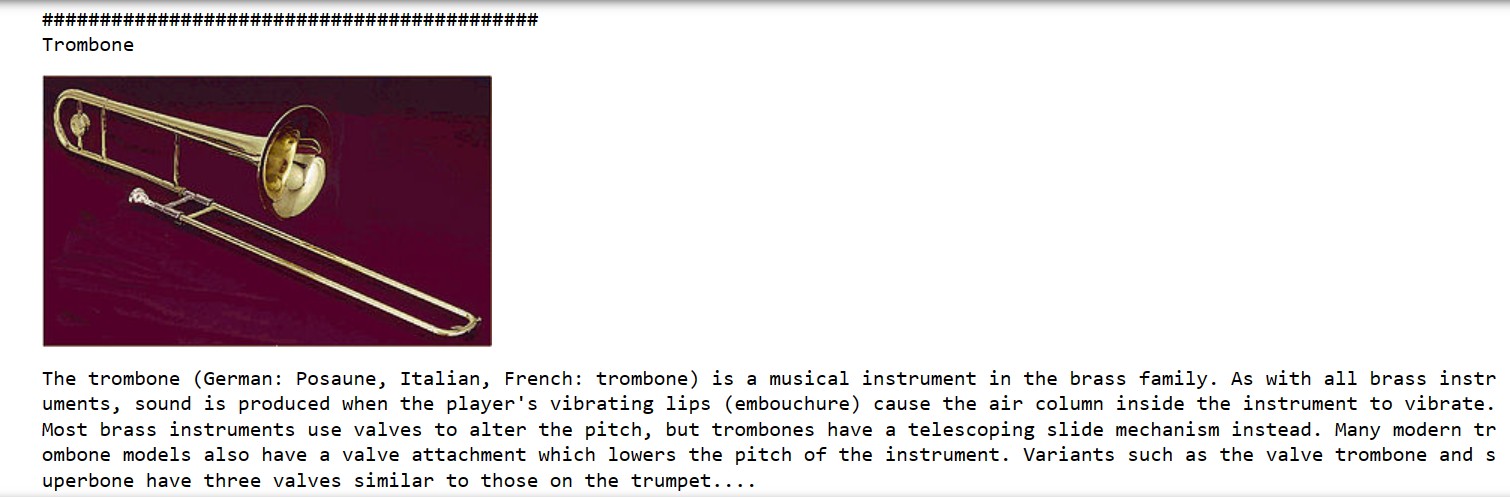
!pip install SPARQLWrapper from rdflib import Graph

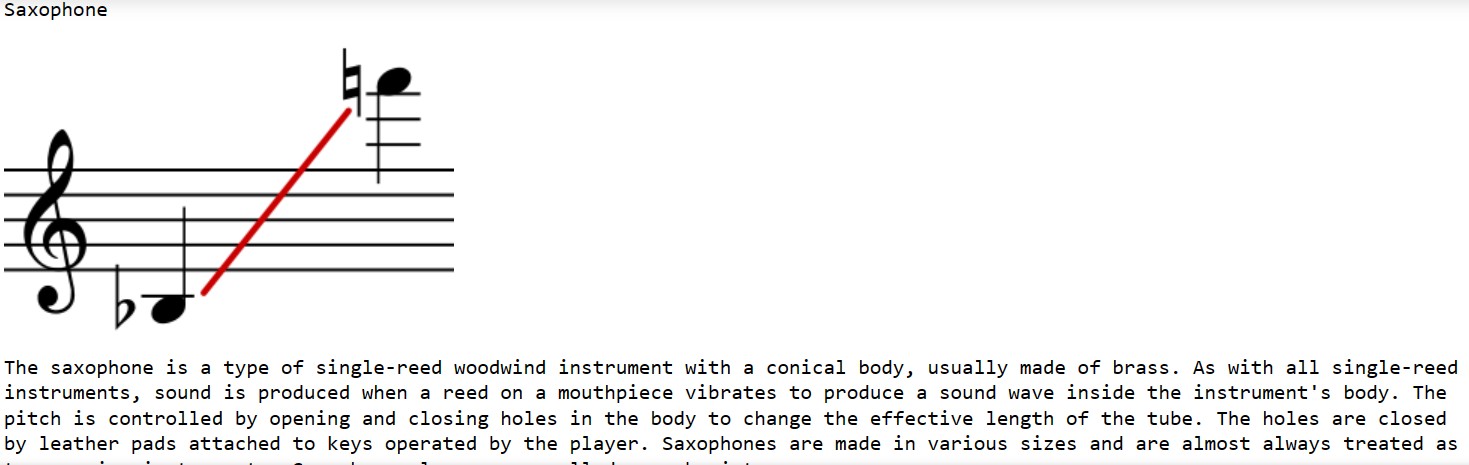
from SPARQLWrapper import SPARQLWrapper,JSON,N3 from pprint import pprint

## Sample execution of Musical Instruments Knowledge base of SPARQL in RDF using python language:



Output:





References: <https://www.dbpedia.org/resources/sparql/> <https://github.com/RDFLib/sparqlwrapper>

<https://www.ontotext.com/knowledgehub/fundamentals/what-is-sparql/>

## Experiment 3

**Aim**: Write Type Script program -Hello World and Calculator

Theory:

The TypeScript source file is in ".ts" extension. We can use any valid ".js" file by renaming it to ".ts" file.

## Text Editor

The text editor helps you to write your source code. Examples of a few editors include Windows Notepad, Notepad++, Emacs, vim or vi, etc. Editors used may vary with Operating Systems.

## The TypeScript Compiler

The TypeScript compiler is itself a .ts file compiled down to JavaScript (.js) file. The TSC (TypeScript Compiler) is a source-to-source compiler (transcompiler / transpiler).



Typescript environment setup:

Installation on Windows

Step 1 − Download and run the .msi installer for Node.

Step 2 − To verify if the installation was successful, enter the command node –v in the terminal window.

Step 3 − Type the following command in the terminal window to install TypeScript.

## npm install -g typescript

The development environment used here is Visual Studio Code (Windows platform). VScode is available at − <https://code.visualstudio.com/>

## Type Script program -Hello World: Step1:

Open editor and type the following code let message: string = 'Hello World'; console.log(message);

## Step2:

Save the above file as .ts extension

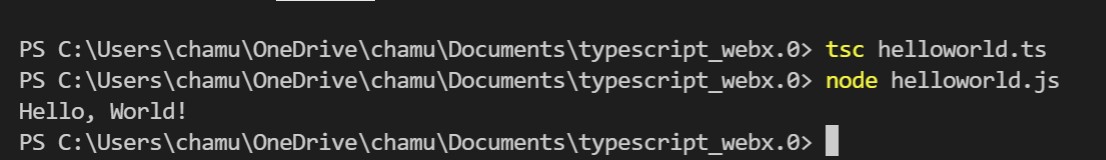
## Step3:

Compile the ts code

## Step4:

Run the js file





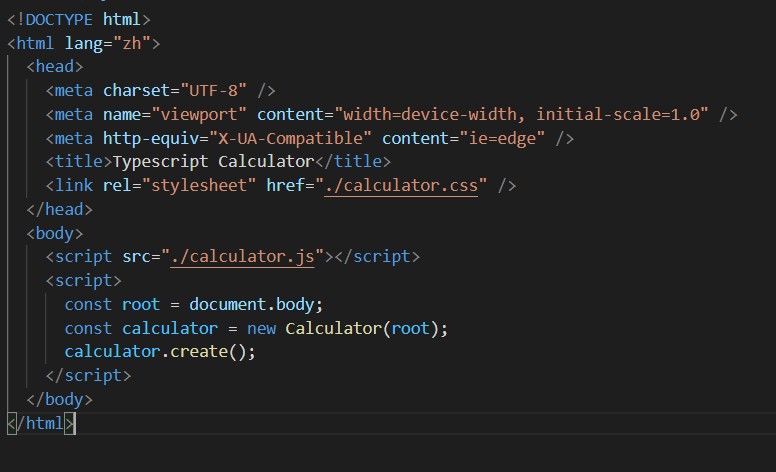
## Type Script program -Calculator

Websites are made up of HTML and/or XML documents. These documents are static, they do not change. The Document Object Model (DOM) is a programming interface implemented by browsers in order to make static websites functional. The DOM API can be used to change the document structure, style, and content. The API is so powerful that countless frontend frameworks (jQuery, React, Angular, etc.) have been developed around it in order to make dynamic websites even easier to develop.

TypeScript is a typed superset of JavaScript, and it ships type definitions for the DOM API. These definitions are readily available in any default TypeScript project.. This type is the backbone for DOM manipulation with TypeScript.

## Step1:

Design a HTML page(calculator.html) for calculator as below



## Step2:

Create a type script file calculator.ts Create a class

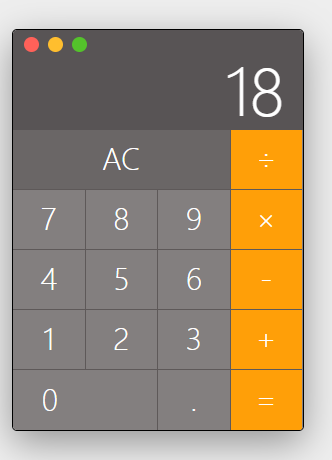
Include all methods like

* + createCalculatorContainer();
  + createDecorator();
  + createResultDisplayContainer();
  + createResultDisplayElement();
  + createButtons();
  + addEventListener()

Use DOM methods like createElement, appendChild to create HTML elements and sub elements for displaying the calculator

Compile the .ts file and .js is passed in to HTML page as normal .js script file

Run the HTML file in browser. below output is visible in browser.



Conclusion: Typescript hello world program and calculator program is implemented successfully.

**Experiment-4 Aim:** Write TypeScript program using Inheritance concept **Theory:**

Inheritance is an aspect of OOPs languages, which provides the ability of a program to create a new class from an existing class. It is a mechanism which acquires the properties and behaviors of a class from another class. The class whose members are inherited is called the base class, and the class that inherits those members is called the derived/child/subclass. In child class, we can override or modify the behaviors of its parent class.Before ES6, JavaScript uses functions and prototype-based inheritance, but TypeScript supports the class-based inheritance which comes from ES6 version.

The TypeScript uses class inheritance through the extends keyword. TypeScript supports only single inheritance and multilevel inheritance. It doesn't support multiple and hybrid inheritance. **Syntax:**

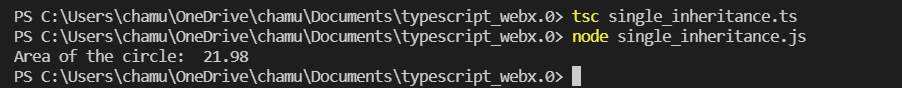
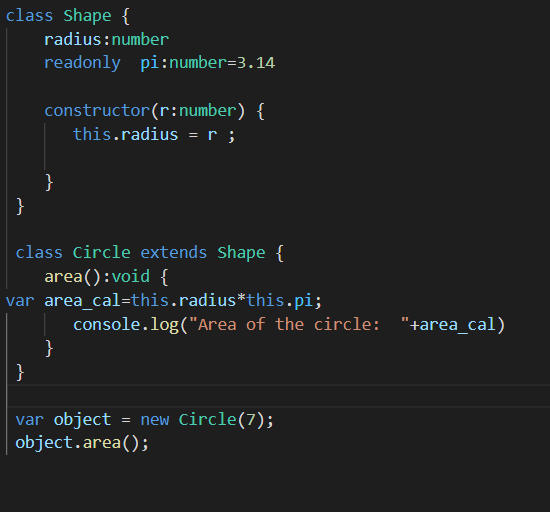
class sub\_class\_name extends super\_class\_name

{

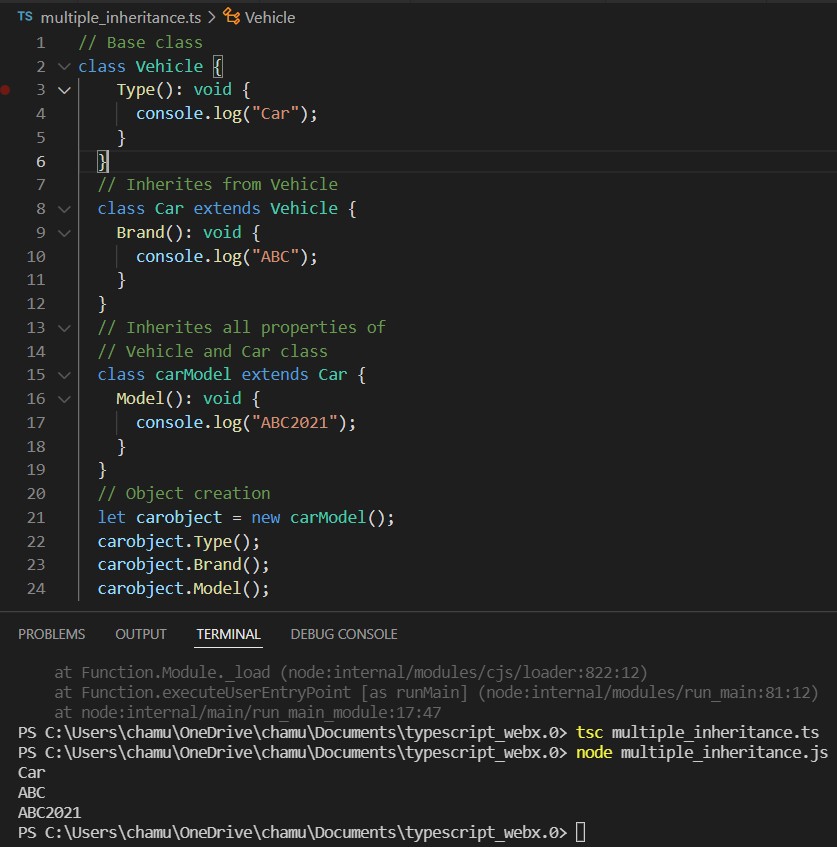
// methods and fields

}

Single Inheritance: Every class can at the most extend from one parent class



Multilevel Inheritance: When a derived class is derived from another derived class, then this type of inheritance is known as multilevel inheritance. Thus, a multilevel inheritance has more than one parent class. It is similar to the relation between Grandfather, Father, and Child.



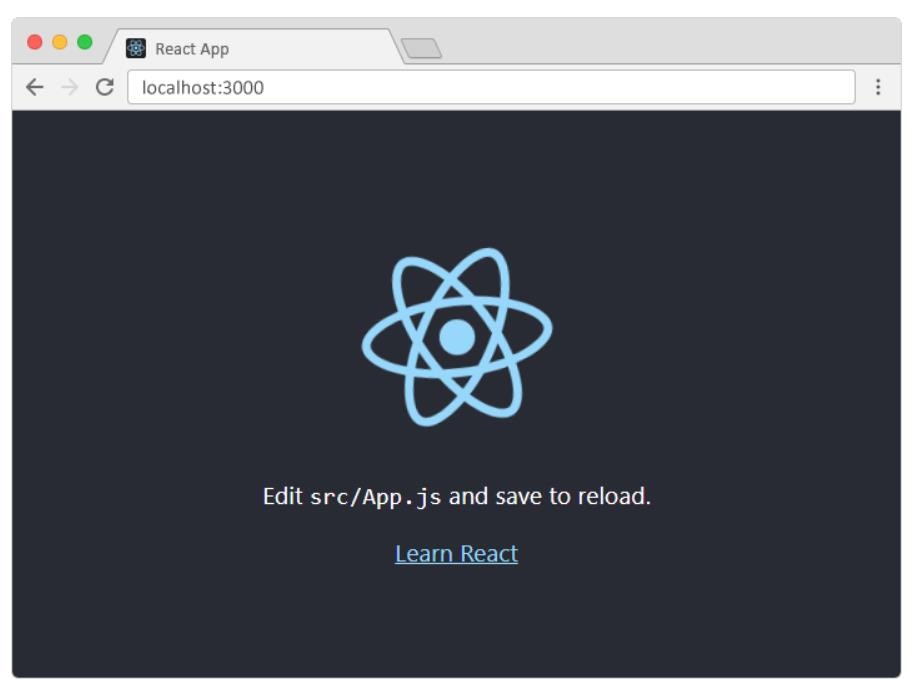
Conclusion:

TypeScript program using Inheritance concept is implemented and executed successfully

**Experiment -5 Aim:** Build a Simple Website using React with TypeScript **Theory:**

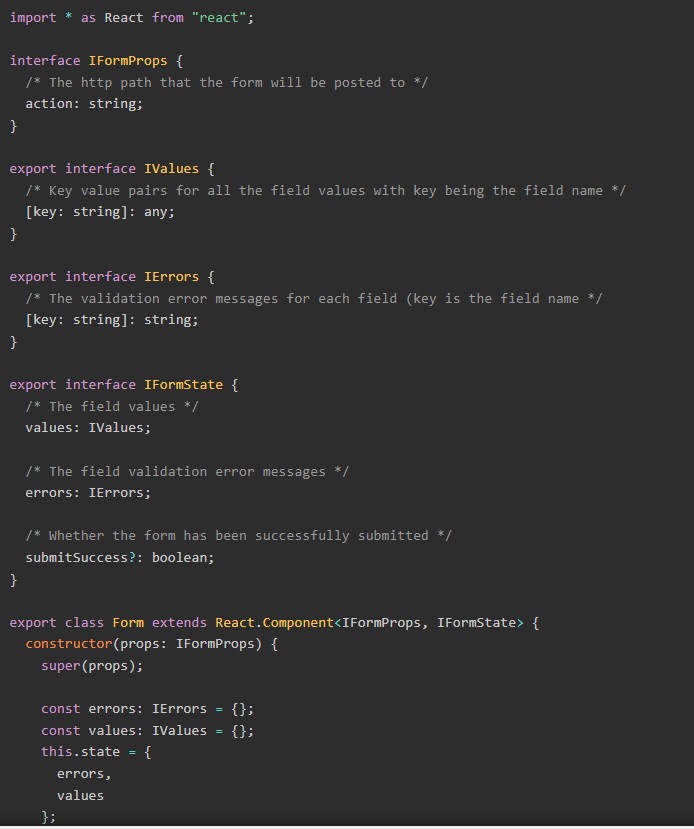
Using Typescript to build our React applications will make our react applications more predictable as we will be able to catch a lot of errors at runtime (during compilation). 1.Steps to create a React app with Typescript

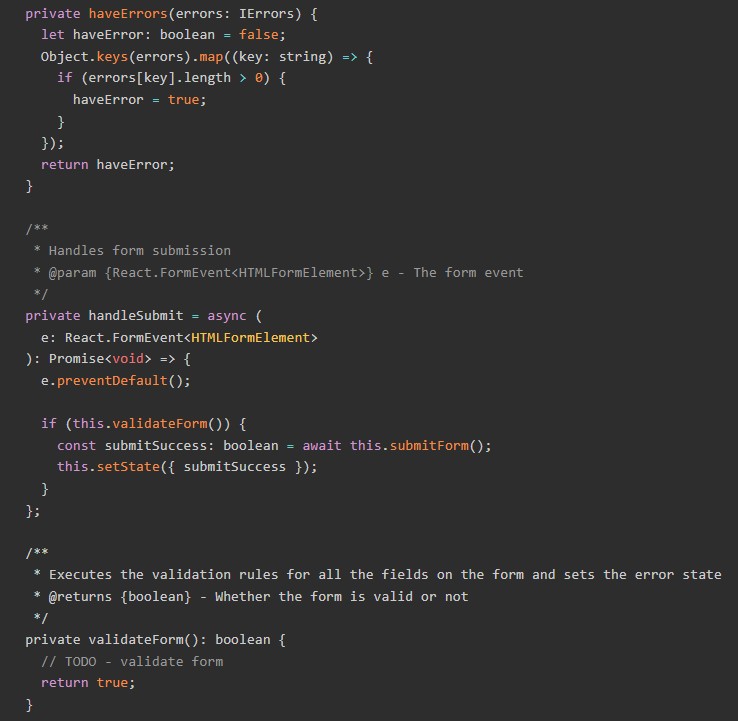
* + npx create-react-app first-app --template typescript
  + cd first-app
  + npm start
  + terminate job by press CTRL+C
  + code .---opens created folder
  + edit code in app.tsx
  + npm start



1. Steps to create a simple Contactus Form using React with Typescript
   * Start by creating a file called **Form.tsx** for our Form component in the src folder and add the code below which gives us a starting point for our form.

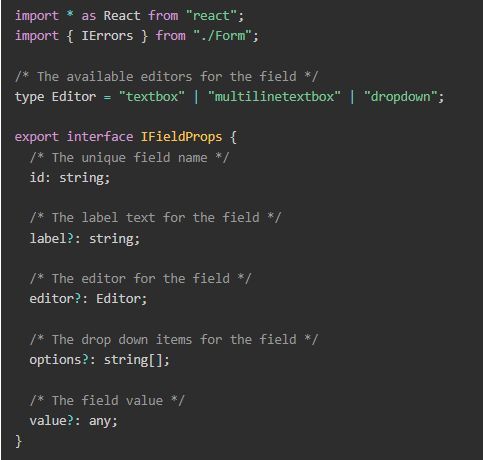
Form.tsx



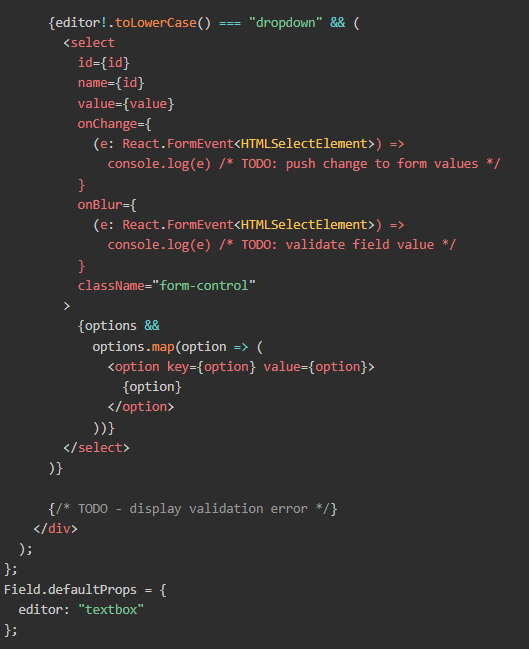




Let’s make a start on a Field component now. Let’s create a file called Field.tsx in the src folder and paste in the code below.



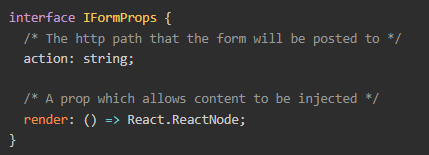




## Rendering fields using render props:

Okay, now let’s start to make Form and Field work together. We’ll start by rendering fields in the appropriate place in the Form component using the render props pattern

So, first we’ll create the render prop:

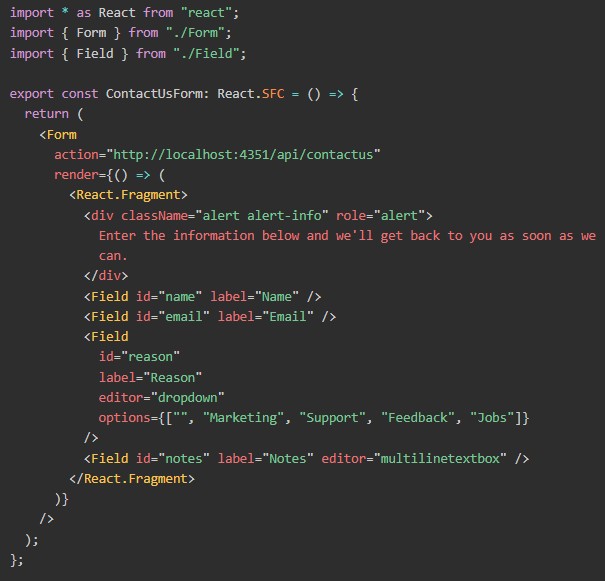


We’ll then make use of this in render():

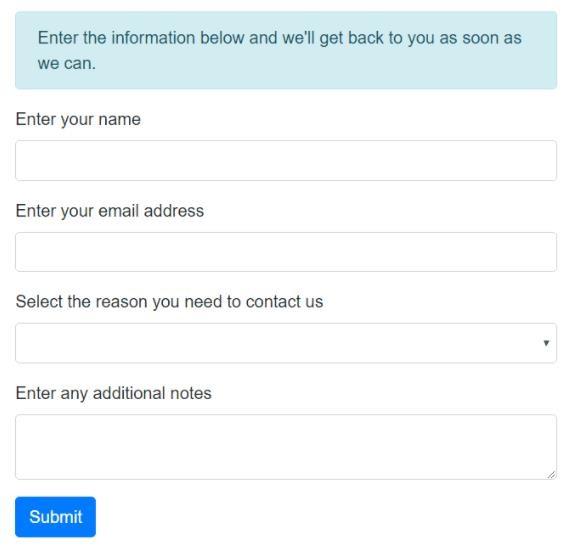


## Creating ContactUsForm

Let’s build the first version of the “contact us” form by creating ContactUsForm.tsx and pasting in the following code:



If we **npm start** the app, it should look like the following:



Conclusion: Simple contact us form have been designed using React with Typescript and implemented successfully.

## Experiment 6

**Aim:** Create a simple HTML Project using AngularJS Framework and apply ng-controller, ng- model and expressions.

## Theory:

Angular JS is an open source JavaScript framework by Google to build web applications. It can be freely used, changed and shared by anyone.

AngularJS is distributed as a JavaScript file, and can be added to a web page with a script tag:

## <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script> AngularJS Extends HTML

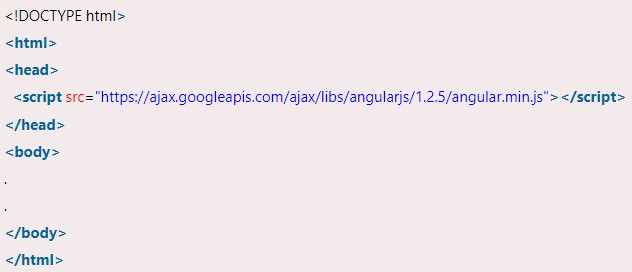
* + AngularJS extends HTML with ng-directives.
  + The ng-app directive defines an AngularJS application.
  + The ng-model directive binds the value of HTML controls (input, select, textarea) to application data.
  + The ng-bind directive binds application data to the HTML view
  + The ng-bind directive binds the content of the <p> element to the application variable name.
  + AngularJS applications are controlled by controllers.
  + The ng-controller directive defines the application controller.
  + A controller is a JavaScript Object, created by a standard JavaScript object constructor.
  + In AngularJS, expressions are used to bind application data to HTML. AngularJS resolves the expression, and return the result exactly where the expression is written.
  + Expressions are written inside double braces {{expression}}.They can also be written inside a directive:

AngularJS HelloWorld:

1. Create a basic HTML page

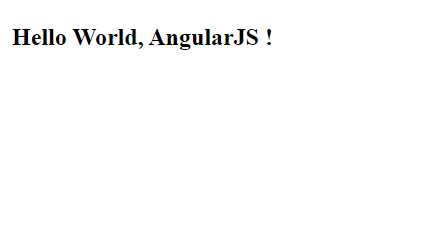


1. Need to include the AngularJS JavaScript file in the HTML page so we can use AngularJS:



1. Add ngapp directive
2. Define module and register defined controller in module





Conclusion: simple Angular JS “Hello World” Project by applying directives ng-controller, ng- model and expressions is designed successfully.

## Experiment 7

**Aim:** Create an application for like Students Record using AngularJS(use angularjs service and form validation )

## Theory:

AngularJS Forms:

Forms in AngularJS provides data-binding Input Controls Input controls are the HTML input elements:

* input elements
* select elements
* button elements
* textarea elements

Data-Binding

Input controls provides data-binding by using the ng-model directive. and validation of input controls

AngularJS Services:

In AngularJS you can make your own service, or use one of the many built-in services. In AngularJS, a service is a function, or object, that is available for, and limited to, your

AngularJS application. Services are created by using service() function on a module and then injected into controllers.

AngularJS Form Validation:

AngularJS provides client-side form validation. It checks the state of the form and input fields (input, textarea, select), and lets you notify the user about the current state.

It also holds the information about whether the input fields have been touched, or modified, or not.

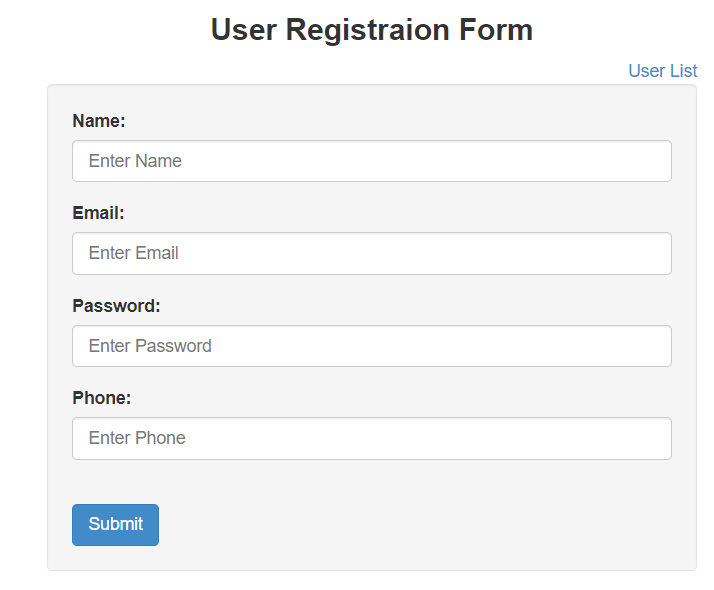
Following directives are generally used to track errors in an AngularJS form:

$dirty - states that value has been changed.

$invalid - states that value entered is invalid.

$error - states the exact error.

**Step1**:design registration form as below



Step2:Define service and use for registration as below myApp.service("RegisterService" , function(){

var uid = 1; var users = [{ 'id' : 0,

'name' : 'John Doe',

'email' : 'johndoe@gmail.com', 'password': 'johndoe',

'phone' : '123-45-678-901'}];

Step3:

Define controller to see list of registered students and to add new student details myApp.controller("RegisterController" , function($scope , RegisterService){

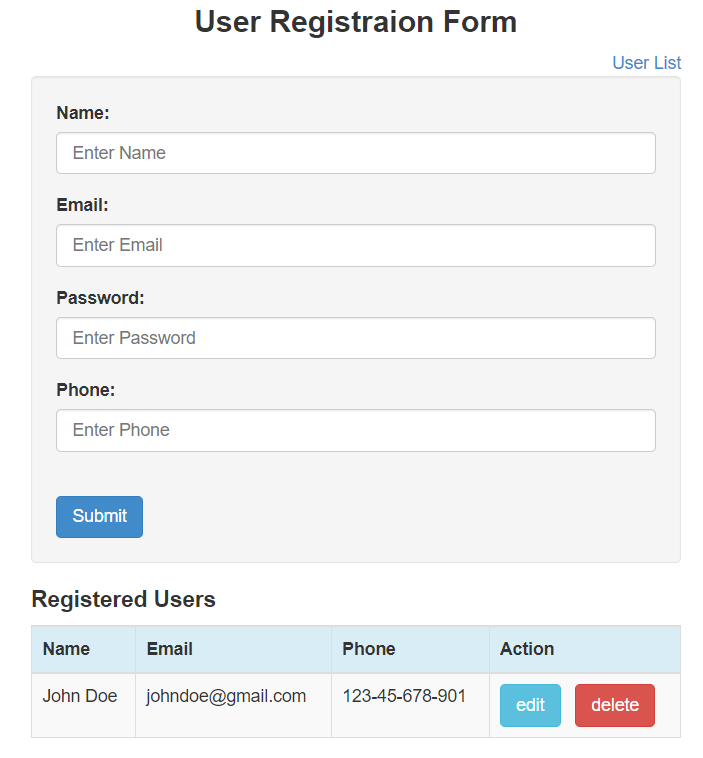
console.clear();

$scope.ifSearchUser = false;

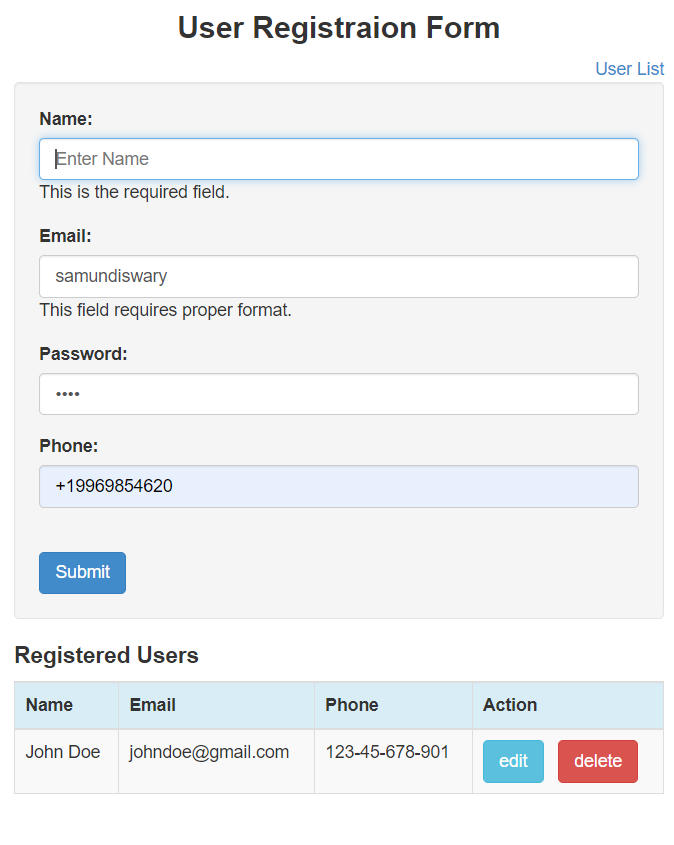
$scope.title ="User List";

$scope.users = RegisterService.list();

$scope.saveUser = function()



Apply validation for required fields and proper format of input as below



Conclusion: Simple Student Registration form have been designed using angularJS with services and form validation concept.

**Experiment 8 Aim:** a.Write a JavaScript program for a AJAX.

b. Write a program to use AJAX for user validation using and to show the result on the same page below the submit button.

c. Design and develop small web application using AJAX, HTML and PHP

## Theory:

AJAX is an acronym for Asynchronous JavaScript and XML. It is a group of inter-related technologies like JavaScript, DOM, XML, HTML/XHTML, CSS, XMLHttpRequest etc.

AJAX allows you to send and receive data asynchronously without reloading the web page. So it is fast.

AJAX allows you to send only important information to the server not the entire page. So only valuable data from the client side is routed to the server side. It makes your application interactive and faster.

Ajax is not a technology but group of inter-related technologies. AJAX technologies includes:

* HTML
* /XHTML
* and CSS
* DOM
* XML
* or JSON
* XMLHttpRequest
* JavaScript

An object of XMLHttpRequest is used for asynchronous communication between client and server.

It performs following operations:

* + Sends data from the client in the background
  + Receives the data from the server
  + Updates the webpage without reloading it.
  + Using simple application in php, will understand the concept of XMLHttpRequest

Create a file **info.php** as shown below

<!DOCTYPE html>

<html>

<body>

<div id="demo">

<h1>The XMLHttpRequest Object</h1>

<button type="button" onclick="loadDoc()">Change Content</button>

</div>

<script>

function loadDoc() {

var xhttp = new XMLHttpRequest(); xhttp.onreadystatechange = function() {

if (this.readyState == 4 && this.status == 200) { document.getElementById("demo").innerHTML = this.responseText;

}

};

xhttp.open("GET", "ajax\_info.txt", true); xhttp.send();

}

</script>

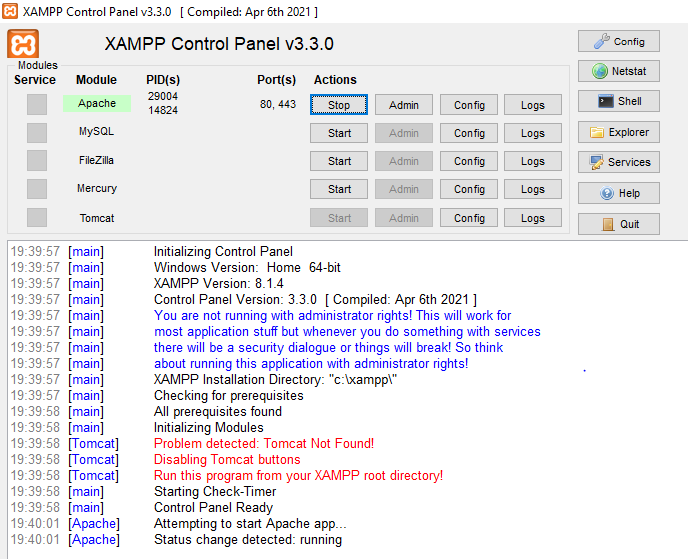
</body>

</html>

Create a file **ajax\_info.txt** which includes as below content shown below AJAX is not a programming language.

AJAX is a technique for accessing web servers from a web page. AJAX stands for Asynchronous JavaScript And XML

Click start Apache server in Xampp



Run info.php file as below



By clicking Change content button ,output display as below.



**Experiment 9 Aim:**Build a RESTful API using MongoDB.

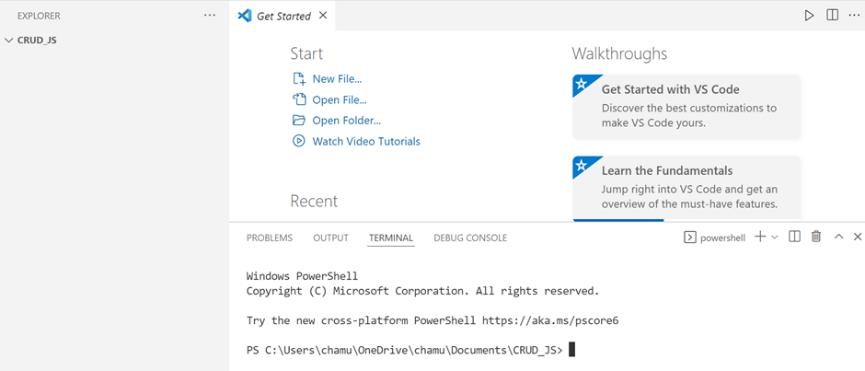
## Theory:

Rest API Overview:

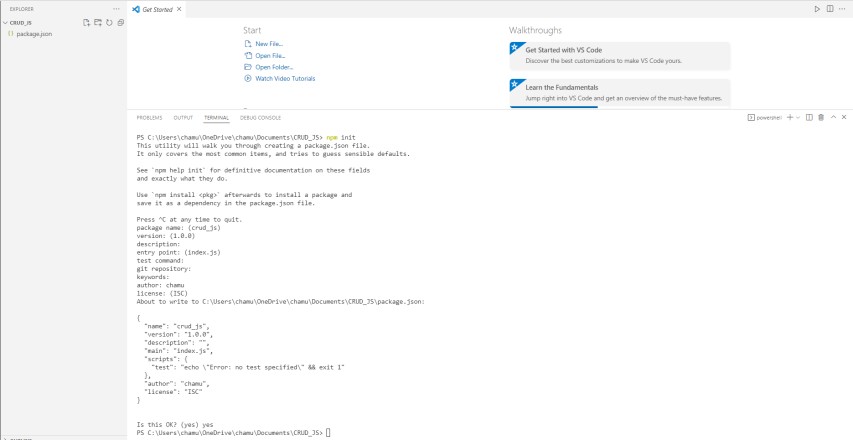
REST stands for REpresentational State Transfer. When a RESTful API is called, the server will transfer to the client a representation of the state of the requested resource. For example, when a developer calls OpenWeather API to fetch weather for a specific city (the resource), the API will return the state of that city, including the temperature, humidity, wind speed, current forecast, extended forecast, and more. The representation of the state can be in a JSON format, and for most web APIs, this is indeed the case. Other possible data formats include XML or HTML. What does the server does when you call it depends on two things that you need to provide to the server:

1. An identifier for the resource. – This is the URL for the resource, also known as the endpoint. In fact, URL stands for Uniform Resource Locator.
2. The operation you want the server to perform on that resource, in the form of an HTTP method. The common HTTP methods are GET, POST, PUT, and DELETE.

Create folder CRUD\_JS and launch in VS code

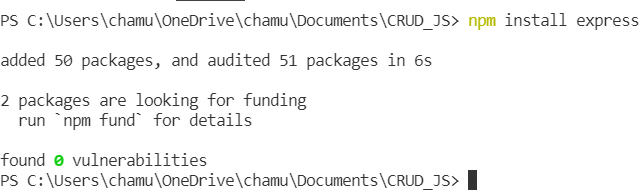


Enter command npm init to initialize packages .json



Install Express by

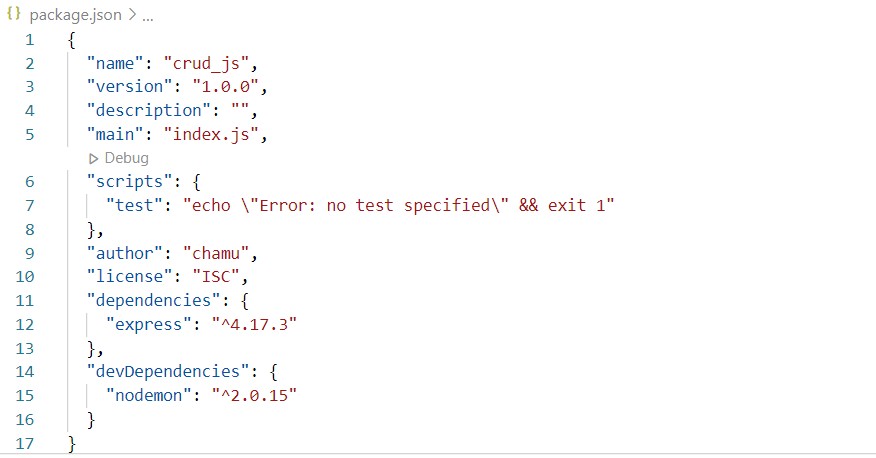
npm install express command



Install MongoDB and mongoose if not having in machine npm install mongodb

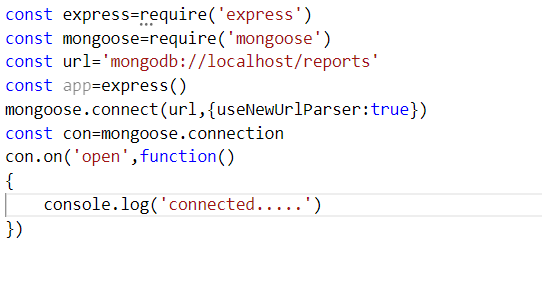
npm install mongoose

if any change is made in server.to reflect automatically without restarting the server npm install -g nodemon --save-dev

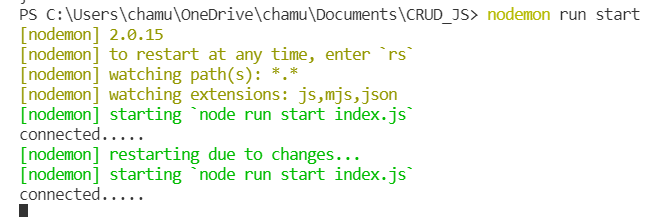


Remove test script from package.json and add "start": "nodemon index.js"

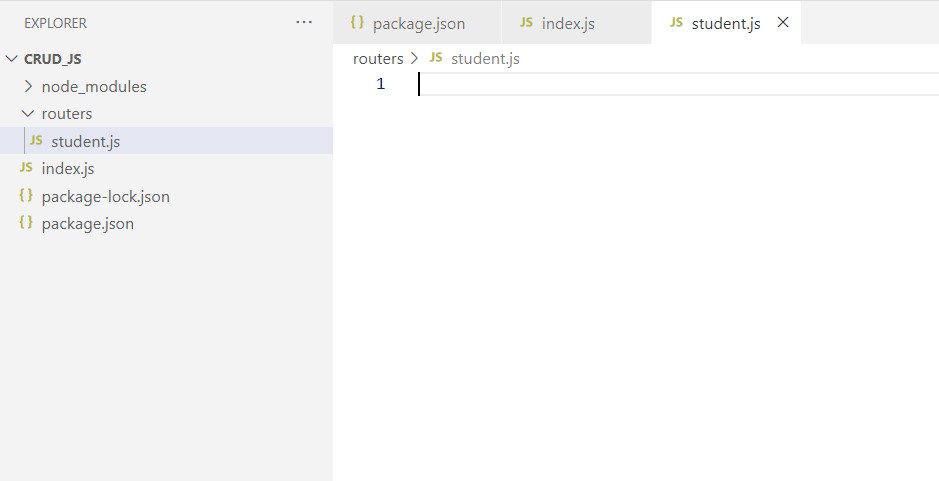
Create a file index.js under root folder and modify code like below to connect nodejs with mongodb



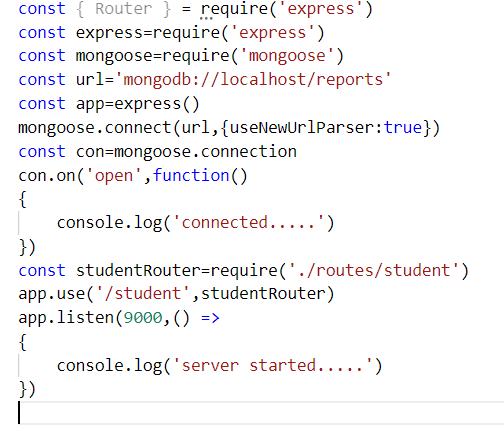
Run index.js by command nodemon run start



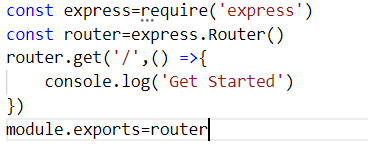
Create folder “routes” and file student.js inside routes folder



Modify index.js and student.js file as below Index.js



Student.js



After the output will be below



Here, we are using Router from Express, and we are exporting it too using module.exports. And now, our app will work fine.

How to Write our Endpoints

Now, let's write our endpoints here in this routes file. We will have five routes for the following actions:

Posting data to Database.

Getting all the data from the Database. Getting data based on the ID. Updating data based on the ID. Deleting data based on the ID.

So, let's create the routes for these actions:

//Post Method

router.post('/post', (req, res) => { res.send('Post API')

})

//Get all Method router.get('/getAll', (req, res) => {

res.send('Get All API')

})

//Get by ID Method router.get('/getOne/:id', (req, res) => {

res.send('Get by ID API')

})

//Update by ID Method router.patch('/update/:id', (req, res) => {

res.send('Update by ID API')

})

//Delete by ID Method router.delete('/delete/:id', (req, res) => {

res.send('Delete by ID API')

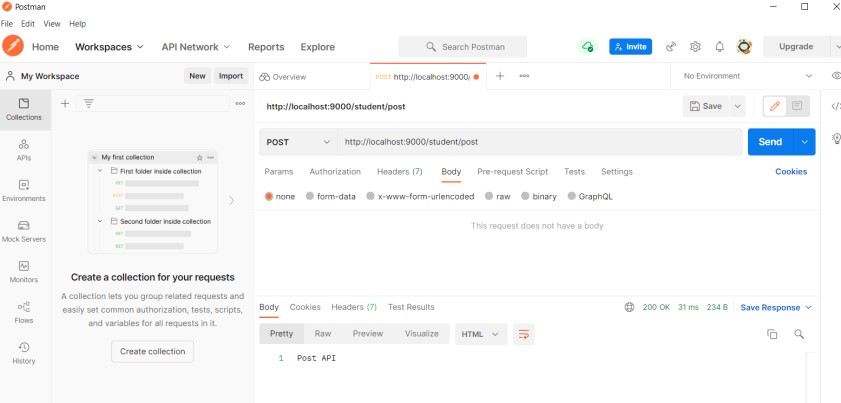
})

Student.js

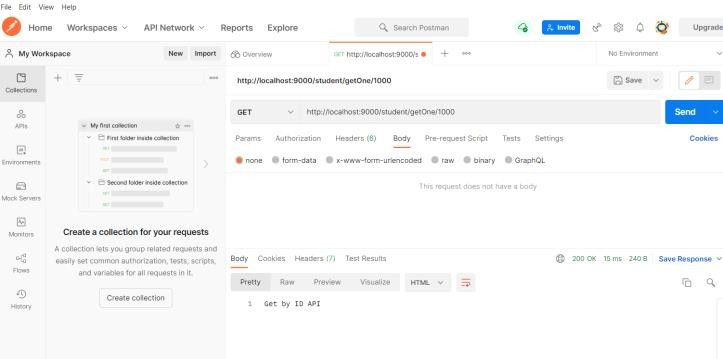


Save this, and open Postman to check the endpoints.

Download Postman if you don't have it using link- https:/[/www.postman.](http://www.postman.com/downloads/)c[om/downloads/](http://www.postman.com/downloads/) http://localhost:9000/student/post



http://localhost:9000/student/getOne/1000



How to Create the Model

Now, let's create a Model that will define our database structure.

Create a folder called model and inside, a file called model.js.Add below code

const mongoose = require('mongoose');

const dataSchema = new mongoose.Schema({ name: {

required: true, type: String

},

age: {

required: true, type: Number

}

})

module.exports = mongoose.model('Data', dataSchema)

Modify student.js file as below const express=require('express') const router=express.Router()

const Model = require('../model/model'); module.exports=router router.post('/post', async (req, res) => {

const data = new Model({ name: req.body.name, age: req.body.age

})

try {

const dataToSave = await data.save(); res.status(200).json(dataToSave)

}

catch (error) {

res.status(400).json({message: error.message})

}

})

Modify index.js file as highlited below const { Router } = require('express') const express=require('express')

const mongoose=require('mongoose') const url='mongodb://localhost/reports' mongoose.connect(url);

const database = mongoose.connection; database.on('error', (error) => {

console.log(error)

})

database.once('connected', () => { console.log('Database Connected');

})const app=express()

mongoose.connect(url,{useNewUrlParser:true})

/\*const con=mongoose.connection con.on('open',function()

{

console.log('connected ')

})\*/ app.use(express.json())

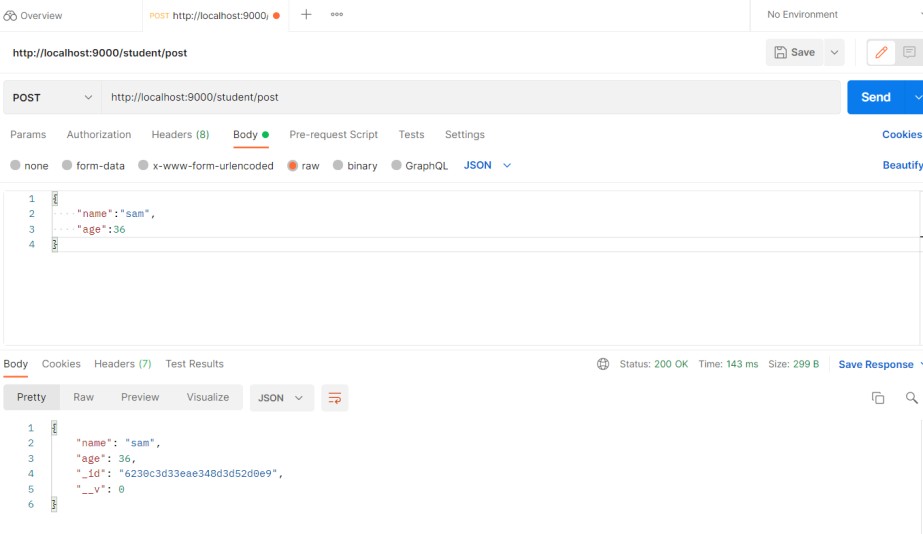
const studentRouter=require('./routes/student') app.use('/student',studentRouter) app.listen(9000,() =>

{

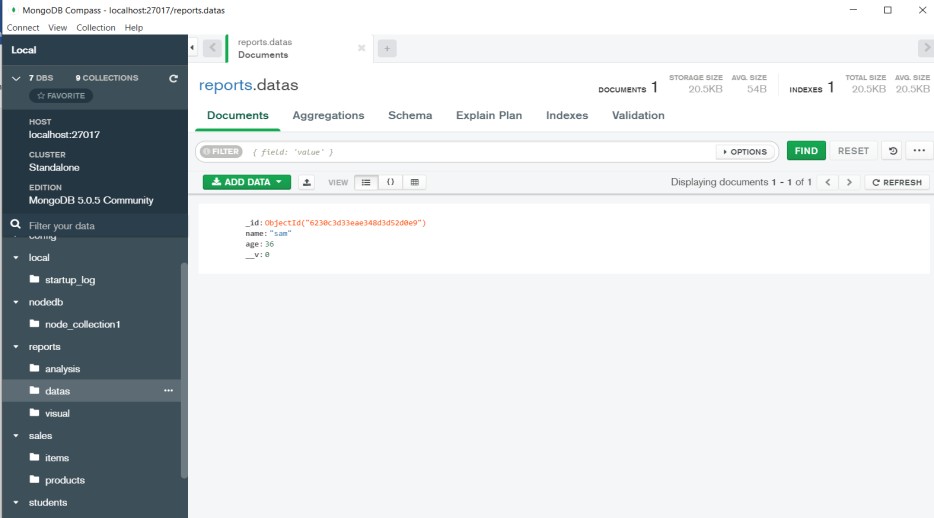
console.log('server started. ')

})

If we add the data in the body and click Send, we will get the following:



It's also generating a unique ID. Open the MongoDB Compass app, and you will see the database and this record you just created:



## References:

https:/[/www.fr](http://www.freecodecamp.org/news/build-a-restful-api-using-node-express-and-mongodb/)e[ecodecamp.org/news/build-a-restful-api-using-node-express-and-mongodb/](http://www.freecodecamp.org/news/build-a-restful-api-using-node-express-and-mongodb/) https:/[/www.youtube.com/watch?v=](http://www.youtube.com/watch?v=eYVGoXPq2RA)e[YVGoXPq2RA](http://www.youtube.com/watch?v=eYVGoXPq2RA)

**Experiment 10 Aim**: Design Feedback Form using Flask

**Theory:**

Flask is a web application framework written in Python. It is developed by Armin Ronacher, who leads an international group of Python enthusiasts named Pocco. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

WSGI

Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

Werkzeug

It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

Jinja2

Jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

The route() function of the Flask class is a decorator, which tells the application which URL should call the associated function.

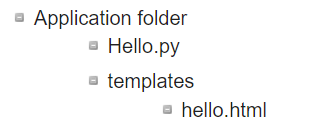
app.route(rule, options)

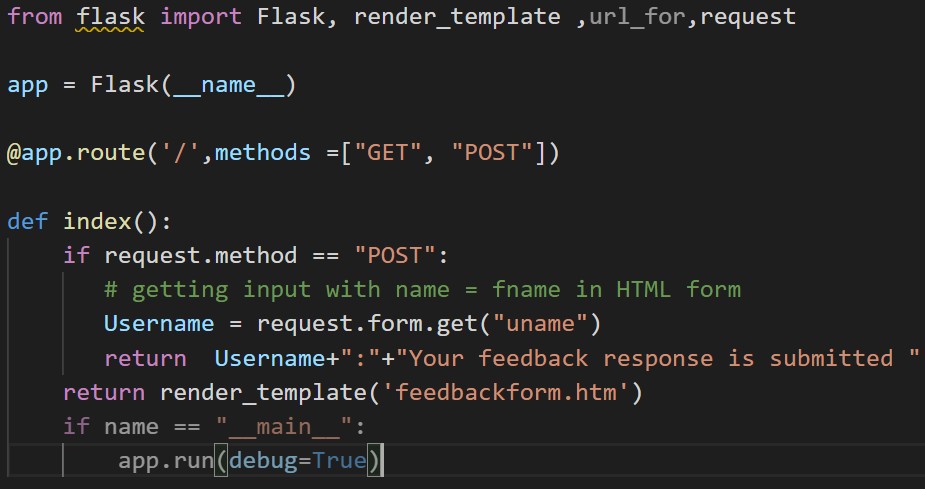
The add\_url\_rule() function of an application object is also available to bind a URL with a function as in the above example, route() is used.

The url\_for() function is very useful for dynamically building a URL for a specific function. The function accepts the name of a function as first argument, and one or more keyword arguments, each corresponding to the variable part of URL.

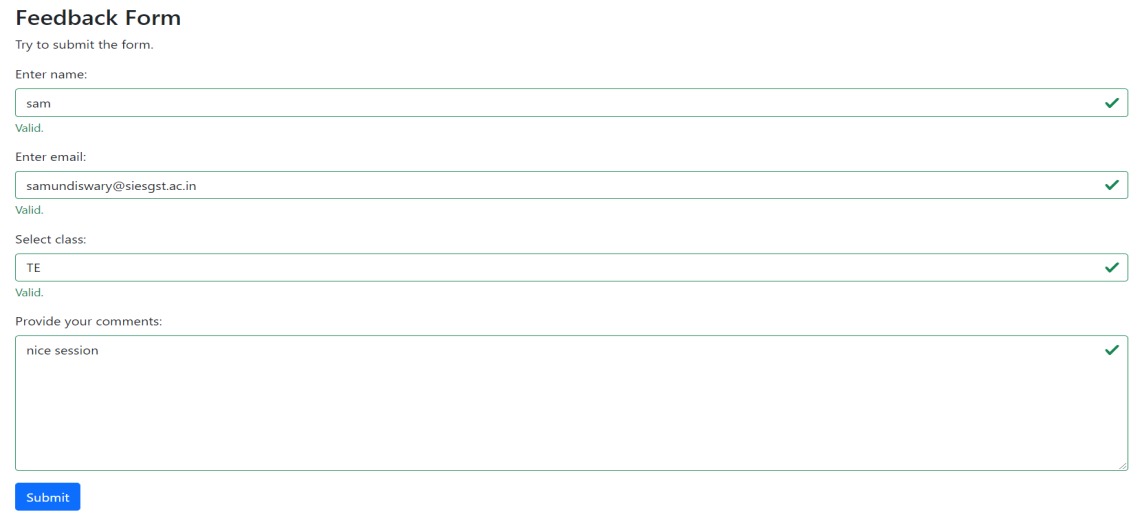
Jinja2 template engine, on which Flask is based. Instead of returning hardcode HTML from the function, a HTML file can be rendered by the render\_template() function.

Flask will try to find the HTML file in the templates folder, in the same folder in which this script is present.





Add feedbackform.html Output:



Username: Your feedback response is submitted

**Conclusion:** Feedback Form is designed using Flask and run successfully.

## Experiment 11

**Aim**: Design Weather app using Flask

**Theory:**

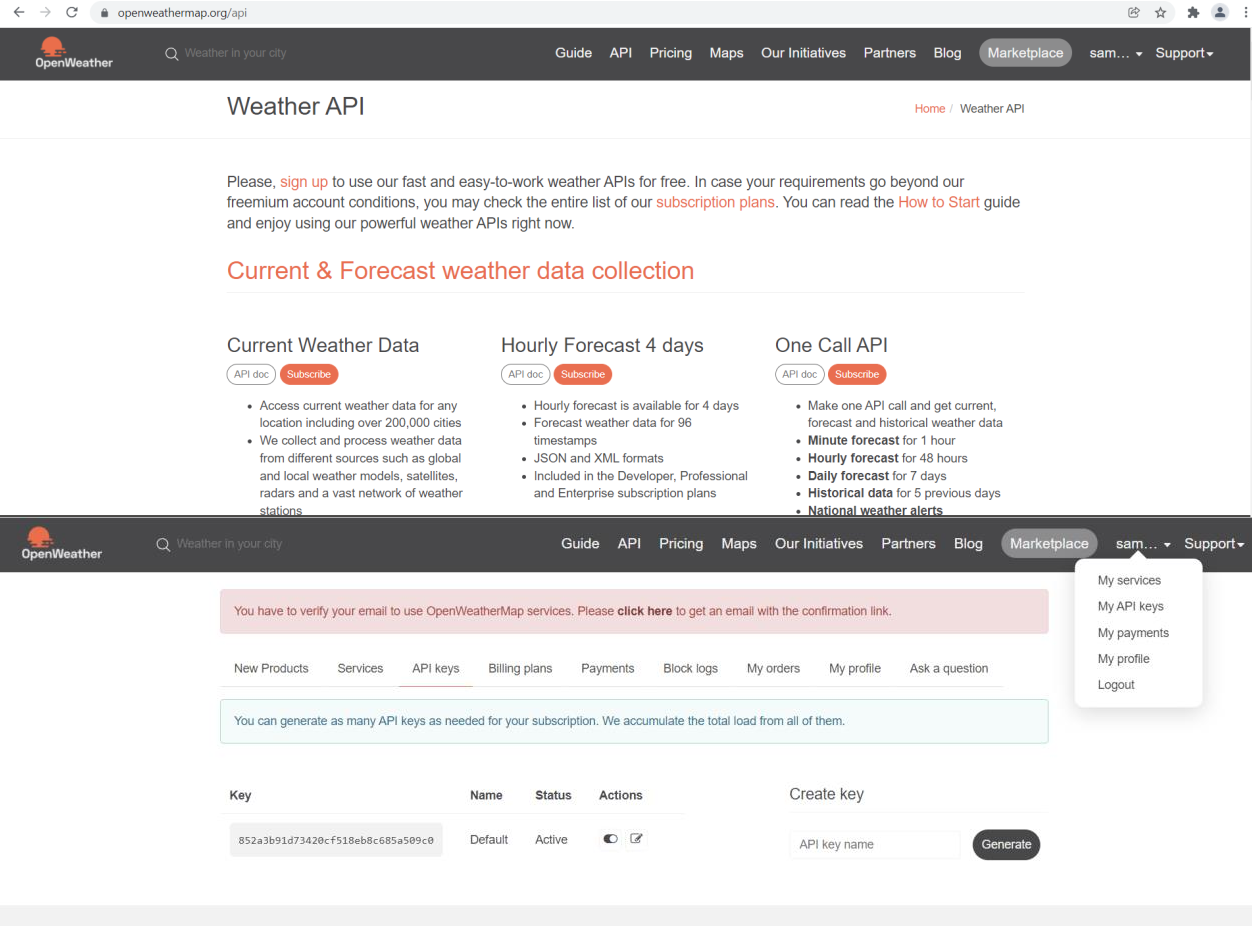
Flask is a lightweight framework written in Python. It is lightweight because it does not require particular tools or libraries and allow rapid web development. we will create a weather app using flask as a web framework. this weather web app will provide current weather updates of cities searched.

Basic setup :

* Create a file and name it as weather.py
* Create html file as weather.html

## Get API Key

Goto https://home.openweathermap.org/users/sign\_up and sign up for a free plan if you haven’t already. Then visit the API key section https://home.openweathermap.org/api\_keys to get your key. You can use the default key or generate a new one as you wish. You may have to wait for a while to get your key verified.



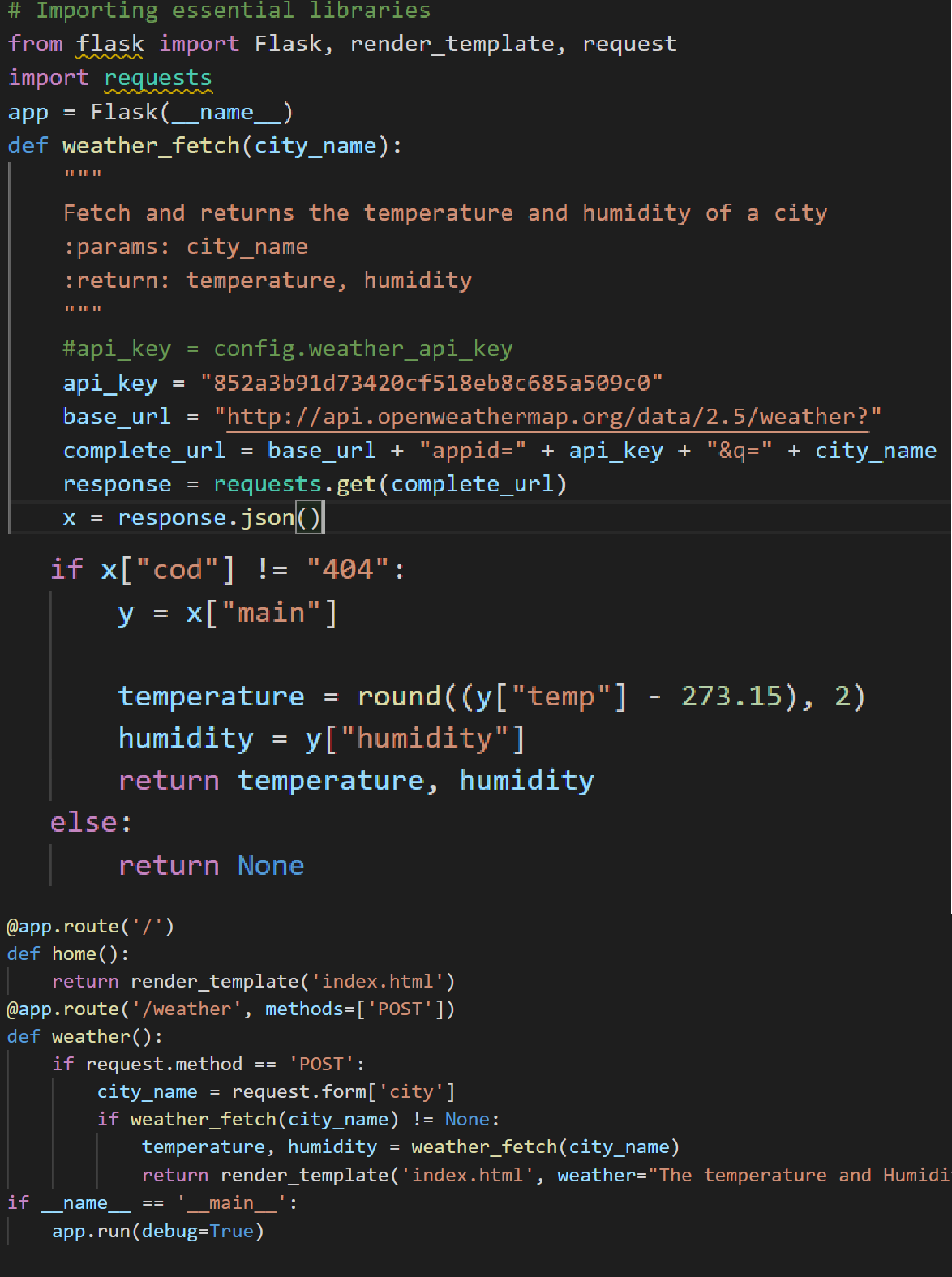
## Making API Calls

Going through the documentation, the simplest way to make requests is using the city name. The url to make request will look like this <http://api.openweathermap.org/data/2.5/weather?q=city_name&APPID=your_api_key>

You can copy paste the url and replace the city\_name and your\_api\_key with appropriate value to see the result.



Creating Flask Application as weather.py



Create weather.html



Run Flask Application using command python weather.py



Output:



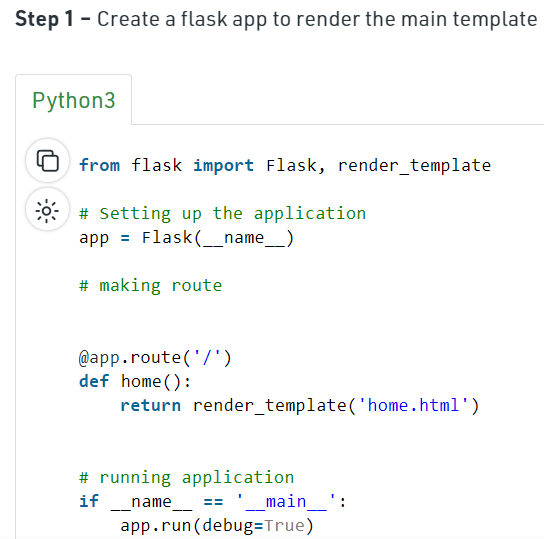
**Experiment 12 Aim**: Design Portfolio Website using Flask.

Theory:

we will create portfolio website using flask framework. this will provide detailed information about a person . At its most basic form, a portfolio website provides professional information about an individual or a company and presents a showcase of their work.

Template Inheritance:

Template inheritance is a very good feature of Jinja templating . Jinja is a web template engine for the Python programming language . We have seen that webpages of a website contains same footer , navigation bar etc. So instead of making same footer and navigation bar in all webpages separately , we make use of template inheritance , which allows us to create the part which is same in all webpages (eg. footer,navigation bar) only once and we also don’t need to write the html , head , title tag again and again . Lets define the common structure of web pages in base.html file. First of all we will render template using flask from main.py file .



Step 2 – Create HTML Files

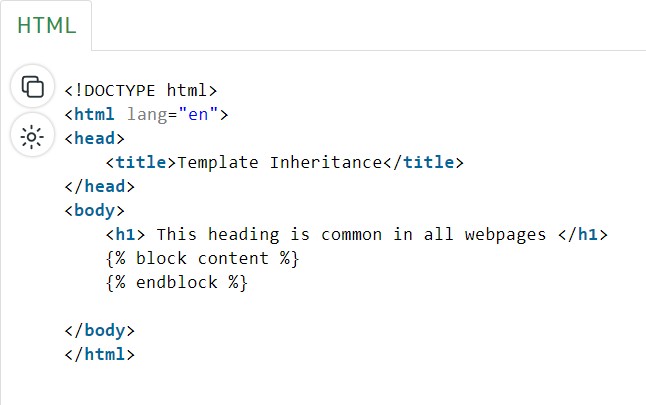
Now we will set up our base.html file in which we have one heading which will be common in all webpages.

Syntax :

{% block content %}

{% endblock %}

The code above and below these lines will be the same for every web pages and the code between them will be for a specific web page .



Now we will set up our home.html file in which we will inherit template from “base.html” file and will write some code for home page also .

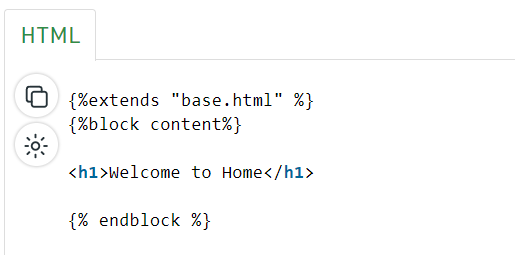
Syntax :

{% extends "base.html" %}

{% block content %}

write code here for home page only

{% endblock %}



Basic setup :

* Create a file and name it as portfolio.py
* Create html file as home.html, about me.html, skills.html, contact.html (apply template inheritance)



Conclusion:Portfolio website has been developed and successfully deployed