# SUMMER TRAINING REPORT
# ON
# WEB DEVELOPMENT and PYTHON

Submitted in the partial fulfillment of the requirements for the award of degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE ENGINEERING

**Submitted By:**

**Nandeesh Gupta**

**Roll No.:07014802718**

**Semester:V**

**Batch :5-C-4**

**Maharaja Agrasen Institute of technology, PSP area,**

**Sector – 22, Rohini, New Delhi – 110086**

**Guru Gobind Singh Indraprastha University**

# DECLARATION

I hereby declare that the work presented in this report entitled “Data Structure Projects in Real Life” and “Python for Everybody”, in partial fulfillment of the requirement for the award of the degree Bachelor of Technology and submitted in Computer Science & Engineering Department of Maharaja Agrasen Institute of Technology,(affiliated to Guru Gobind Singh Indraprastha University, New Delhi) is an authentic record of my own work carried out during Summer 2020 under the guidance of Mr. Prateek Narang and Mr. Charles Russell Severance

Date: 11-08-2020

NANDEESH GUPTA
07014802718

Certified that the above statement made by the student is correct to the best of our knowledge and belief.

Signatures

Head of Department

(Signature and Seal)

## ACKNOWLEDGMENT

It is my pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behavior and acts during the work.
First and foremost, I express my sentiment of gratitude to Mr Charles Russell Severance, Clinical Professor at the University of Michigan School of Information. Also my gratitude to Mr. Prateek Narang, co-founder of Coding Blocks. They have been a continuous source of inspiration as my training mentor. Without their constant guidance and suggestions, this report would have been nowhere near completion. My gratitude for their trust and generosity goes beyond words. I had a wonderful and an unforgettable experience with these trainings.
I express my sincere gratitude to Computer Science Department, M.A.I.T  for providing me an opportunity to undergo summer industrial training at coursera.
Lastly, I would like to thank the Almighty, and my parents for their moral support and my friends with whom I shared my experience of the training and received lots of suggestions that improved my quality of work.

Nandeesh Gupta
B.Tech (C.S.E)
Enrollment No.
07014802718

# ABSTRACT

This report is prepared to fulfill the requirement of the Bachelor of Technology program of Maharaja Agrasen Institute Of Technology on Front End Web development with JavaScript at Coding Blocks and on Python for Everybody from The University of Michigan (Coursera).

In this duration of training I learned how to work on a web project using vis.js, HTML, CSS, JavaScript in the front end. I also learnt how to use Python and URLLIB, JSON, RegEx, BeautifulSoup libraries to access data from the web.
I learnt various features of Web development like Function hoisting of JavaScript, functional components, libraries of Python, DOM, web crawling, web scraping, Database connection and Sqlite, JSON and XML. These concepts helped in building many mini projects that were submitted as projects during the training.

# Contents

## SECTION 1

## Front End Web Development with Real Life Projects from Coding Blocks

## ABOUT THE COMPANY

Coding Blocks was founded in 2014 with a mission to create skilled Software Engineers for their country and the world. they are here to bridge the gap between the quality of skills demanded by industry and the quality of skills imparted by conventional institutes. Programming is a lot of fun because, unlike other subjects, you get to instantly apply the concepts you are learning. At Coding Blocks, they strive to increase student interest by providing hands on practical training on every concept taught in the classroom. they create confident developers who think beyond industrial jobs and march their ideas into self-created entrepreneurship ventures. Skill and innovative thinking give developers the confidence to transform their ideas into real life products and hopefully go on to build million dollar companies. Along with training students with the latest technologies and programming languages, they also connect them to software companies via their placement assistance program. This program includes practicing a lot of interview problems and mock interviews conducted by company representatives.

## ABOUT THIS COURSE

This course explores Javascript based front-end application development, and in particular the vis.js and phaser library. One of its kind, newly launched course on Data Structures Projects is a must do offering. Designed for college students, this course will help to ignite the interest of students towards solving some real life problems. The course starts with basics of Javascript, diving quickly to problem solving by building a couple of real projects that include games, puzzles & web-apps. The course will gives opportunity to apply our algorithmic skills such as backtracking, graph algorithms, dynamic programming, OOPs concepts to build some interesting projects.

# TOOLS AND TECHNOLOGIES USED

## HTML (Hypertext Mark-up Language)

HTML (Hypertext Mark-up Language) is the most basic building block of the Web. It describes and defines the content of a webpage. "Hypertext" refers to links that connect webpages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.HTML uses "mark-up" to annotate text, images, and other content for display in a Web Browser.

Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. HTML5 is also a candidate for cross-platform mobile applications, because it includes features designed with low-powered devices in mind. Many new syntactic features are included.

## CSS (Cascading Style Sheets)

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a mark-up language. CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colours, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based mark-up language.

This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content. The CSS specifications are maintained by the World Wide Web Consortium (W3C).

CSS3 is the latest version of the CSS specification. The CSS specifications are maintained by the World Wide Web Consortium (W3C). CSS# is a collaboration of CSS2 specifications and new specifications. CSS3 adds several new styling features and improvements to enhance the web presentation capabilities.

- Advanced Selectors
- Box Model
- Transformations
- Animations
- Multi column layout
- Media Queries
- Transitions
- Gradients
- Webfonts

## Bootstrap

Bootstrap is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first web sites. Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSSbased design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Bootstrap components such as navbar, Breadcrumbs, paginations and Bootstrap’s JavaScript plugins such as dropdown, modals, carousel etc. are used in Training Management web application. Bootstrap includes a powerful mobile-first flex box grid system for building layouts of all shapes and sizes. It’s based on a 12 column layout and has multiple tiers, one for each media query range.

The most prominent components of Bootstrap are its layout components, as they affect an entire web page. The basic layout component is called "Container", as every other element in the page is placed in it. Developers can choose between a fixed-width container and a fluid-width container. While the latter always fills the width of the web page, the former uses one of the four predefined fixed widths, depending on the size of the screen showing the page:

- Smaller than 576 pixels
- 576–768 pixels
- 768–992 pixels
- 992–1200 pixels
- Larger than 1200 pixels

Once a container is in place, other Bootstrap layout components implement a CSS grid layout through defining rows and columns.

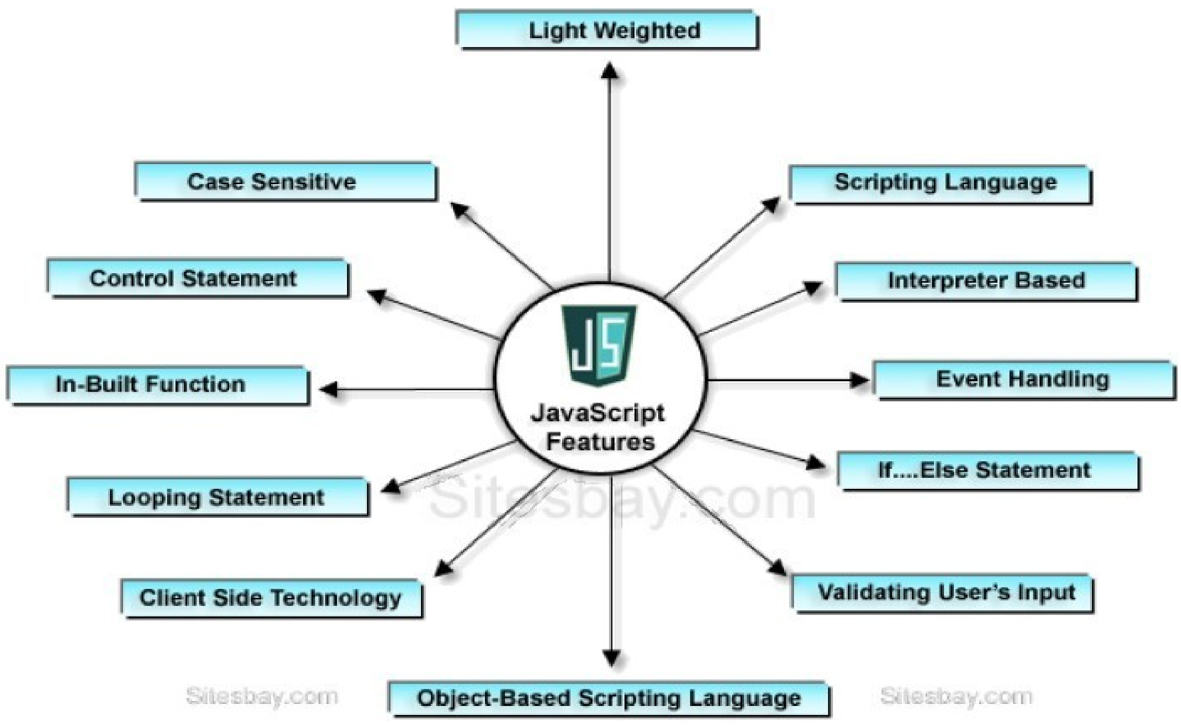
## JavaScript

JavaScript is the client side scripting language of the web. It's one of the most popular and in demand skills in today's job market for good reason. JavaScript not only enables you to add powerful interactions to websites, but is also the foundation of a lot of commonly used libraries like jQuery.

JavaScript often abbreviated as JS,is a high-level,interpreted programming language.It is a language which is also characterized as dynamic,weakly typed,prototype-basedand multi-paradigm.

Alongside HTML and CSS,JavaScript is one of the three core technologies of the World Wide Web.JavaScript enables interactive web pages and thus is an essential part of web applications. The vast majority of websites use it, and all major web browsers have a dedicated JavaScript engineto execute it.

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles.It has an API for working with text, arrays,dates, regular expressions,and basic manipulation of the DOM,but the language itself does not include any I/O,such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.



## GIT and GITHUB

Git is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity and support for distributed, non-linear workflows.
GitHub is an American company that provides hosting for software development version control using Git. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

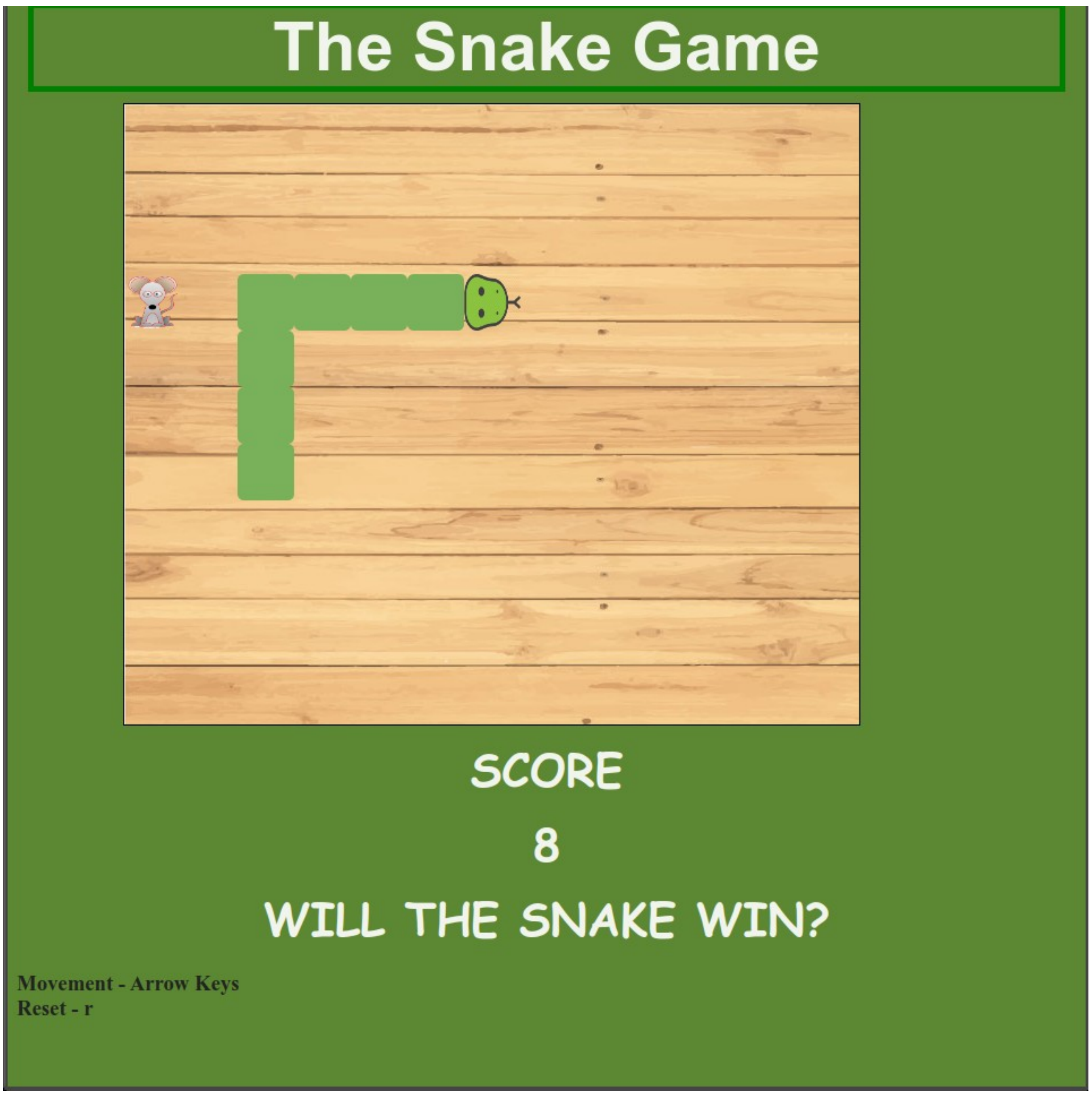
# PROJECT 1

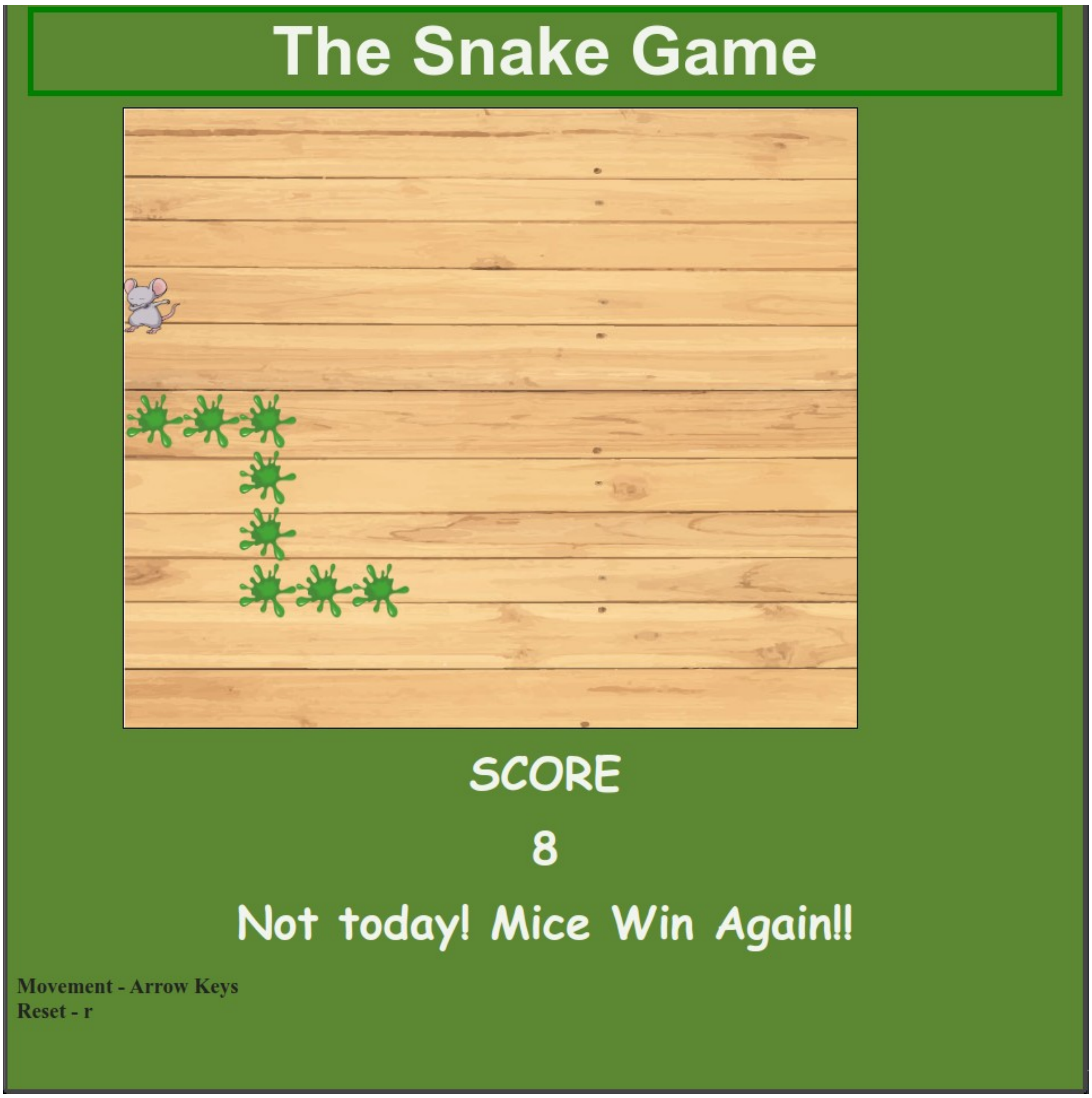
## The Snake Game

Snake is the common name for a video game concept where the player maneuvers a line which grows in length, with the line itself being a primary obstacle. The concept originated in the 1976 arcade game Blockade, and the ease of implementing Snake has led to hundreds of versions (some of which have the word snake or worm in the title) for many platforms. After a variant was preloaded on Nokia mobile phones in 1998, there was a resurgence of interest in the snake concept as it found a larger audience. There are over 300 Snake-like games for iOS alone.

In my project I have used a Canvas element to animate the snake and its food – Mouse.

### Snapshots




The Snake Game

SCORE

8

Not today! Mice Win Again!!

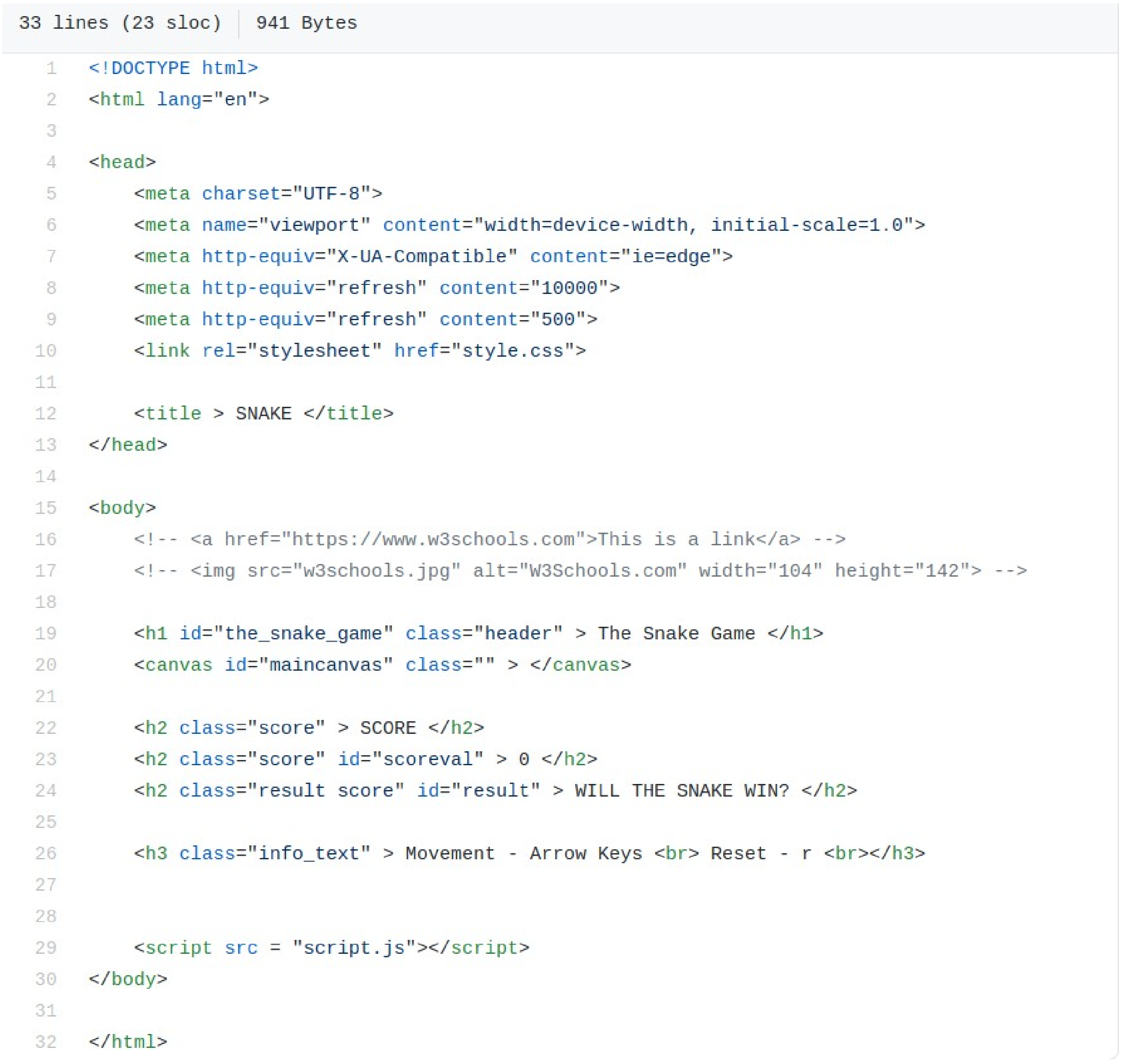
Movement - Arrow Keys
Reset - r

Snake dies and Mouse Celebrates.

## Explanation and Code

Languages used were only HTML,CSS and JavaScript. I used Canvas element for the animations and displayed pictures to simulte movement. There were event listeners which respond to keypresses and reload commands at all time. Mouse was placed randomly, taking care that the random placement should be outside of Snake's body only. The score was noted and displayed below the canvas element. Project is live on https://nandeeshg.github.io/SnakeGame/
Full code is available at https://github.com/NandeeshG/SnakeGame

## Code Snapshots

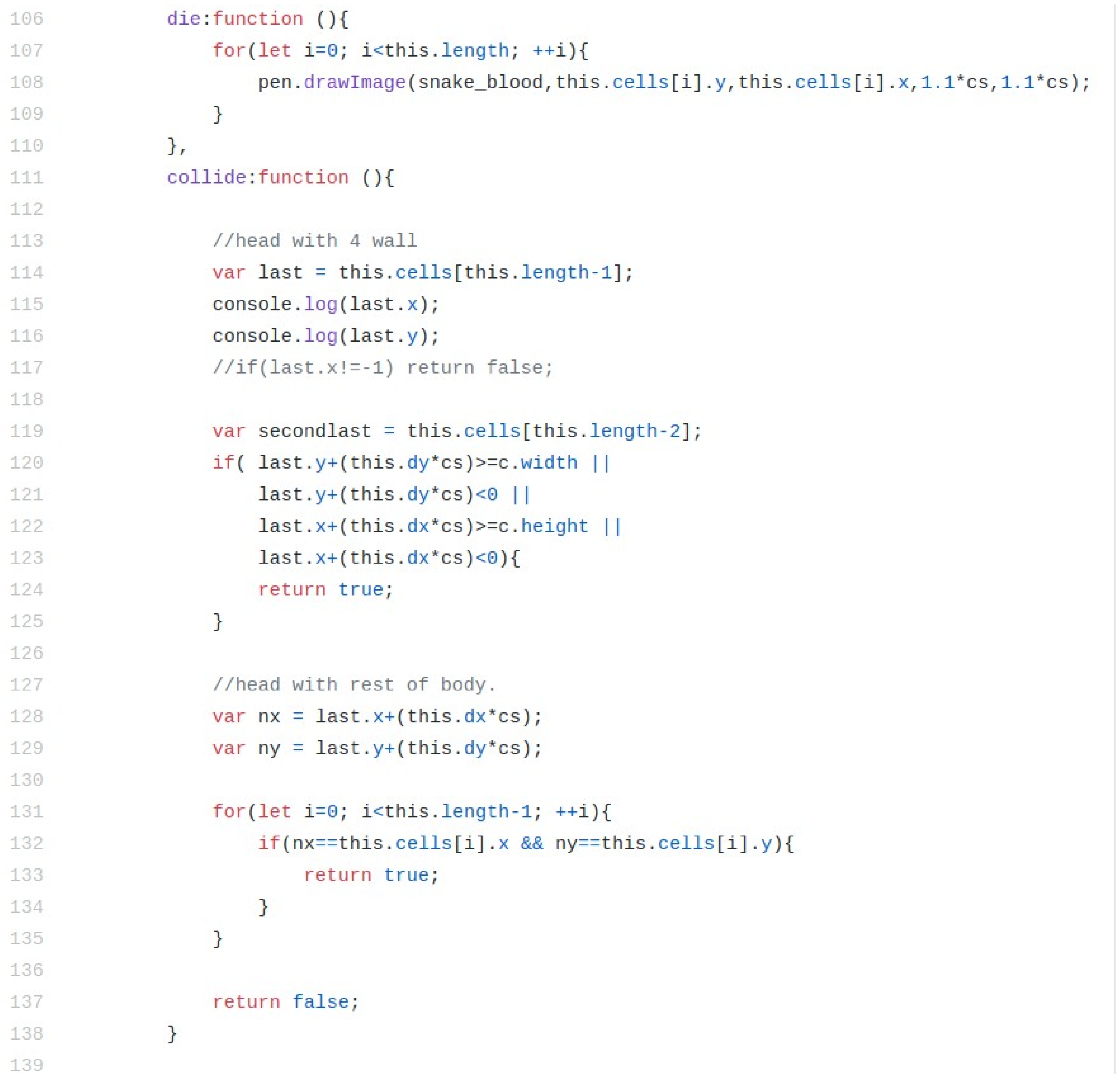


```
33 lines (23 sloc)  |  941 Bytes

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <meta http-equiv="refresh" content="10000">
    <meta http-equiv="refresh" content="500">
    <link rel="stylesheet" href="style.css">

    <title > SNAKE </title>
</head>

<body>
    <!-- <a href="https://www.w3schools.com">This is a link</a> -->
    <!-- <img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142"> -->

    <h1 id="the_snake_game" class="header" > The Snake Game </h1>
    <canvas id="maincanvas" class="" > </canvas>

    <h2 class="score" > SCORE </h2>
    <h2 class="score" id="scoreval" > 0 </h2>
    <h2 class="result score" id="result" > WILL THE SNAKE WIN? </h2>

    <h3 class="info_text" > Movement - Arrow Keys <br> Reset - r <br></h3>


    <script src = "script.js"></script>
</body>

</html>
```

index.html – the layout of screen

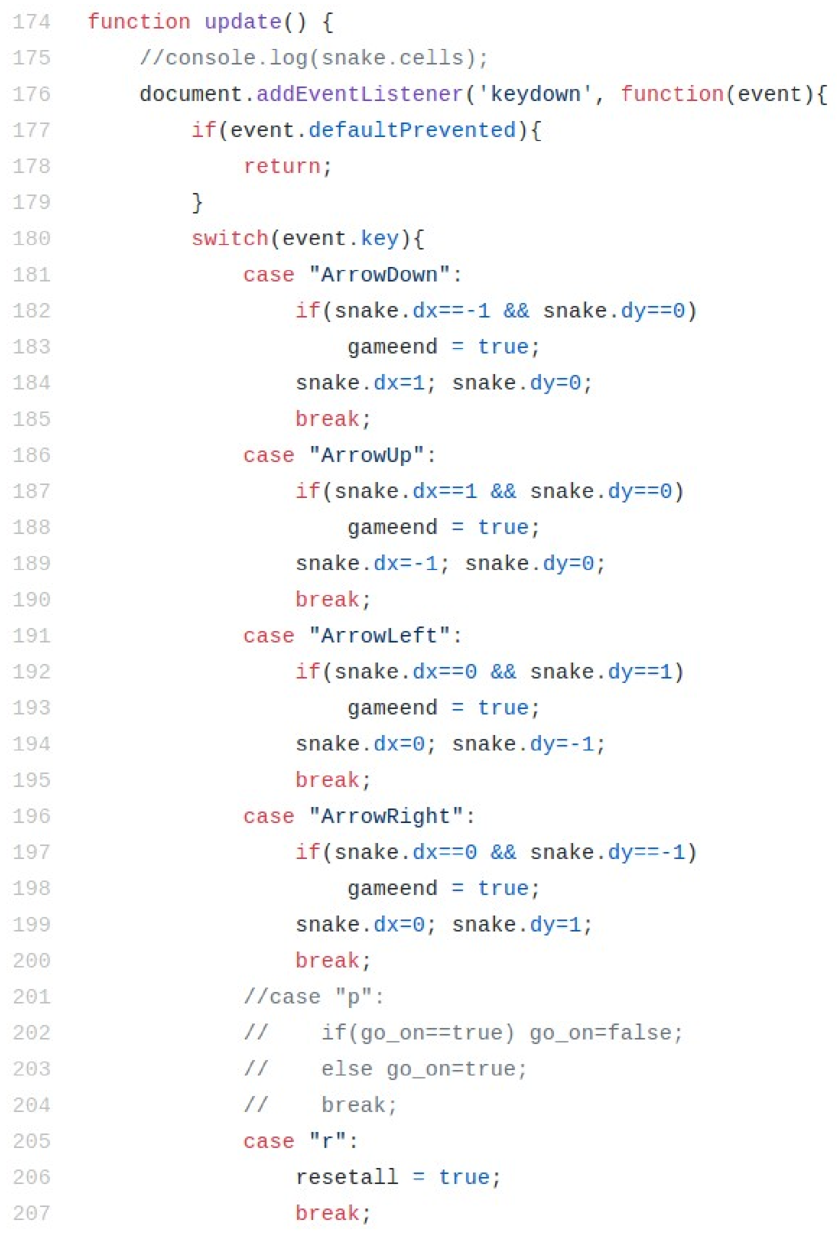
```
        drawSnake:function () {
            for(let i=0; i<this.length-1; ++i){
                pen.drawImage(snake_skin,this.cells[i].y,this.cells[i].x,cs,cs);
            }
            var lastx = this.cells[this.length-1].x;
            var lasty = this.cells[this.length-1].y;
            if(this.dx==1 && this.dy==0){
                pen.drawImage(pat_head_down,lasty,lastx,cs,cs);
            }
            else if(this.dx==0 && this.dy==1){
                pen.drawImage(pat_head_right,lasty,lastx,cs,cs);
            }
            else if(this.dx==0 && this.dy==-1){
                pen.drawImage(pat_head_left,lasty,lastx,cs,cs);
            }
            else if(this.dx==-1 && this.dy==0){
                pen.drawImage(pat_head_up,lasty,lastx,cs,cs);
            }
        },
        shiftSnake:function (){
            if(this.collide()){
                gameend=true;
                return;
            }
            this.eatFood();
            var last = this.cells[this.length-1];
            console.log(last);
            var newlast = {x:last.x+(cs*this.dx), y:last.y+(cs*this.dy)};
            console.log(newlast);
            for(let i=0; i<this.length-1; ++i){
                console.log(this.cells[i]);
                this.cells[i] = this.cells[i+1];
            }
            this.cells[this.length-1] = newlast;
        },
```

script.js – Important function shiftSnake and drawSnake work in tandem to display and shift the snake.



```
        die:function (){
            for(let i=0; i<this.length; ++i){
                pen.drawImage(snake_blood,this.cells[i].y,this.cells[i].x,1.1*cs,1.1*cs);
            }
        },
        collide:function (){

            //head with 4 wall
            var last = this.cells[this.length-1];
            console.log(last.x);
            console.log(last.y);
            //if(last.x!=-1) return false;

            var secondlast = this.cells[this.length-2];
            if( last.y+(this.dy*cs)>=c.width ||
                last.y+(this.dy*cs)<0 ||
                last.x+(this.dx*cs)>=c.height ||
                last.x+(this.dx*cs)<0){
                return true;
            }

            //head with rest of body.
            var nx = last.x+(this.dx*cs);
            var ny = last.y+(this.dy*cs);

            for(let i=0; i<this.length-1; ++i){
                if(nx==this.cells[i].x && ny==this.cells[i].y){
                    return true;
                }
            }

            return false;
        }

```

Script.js – This snapshot shows the function where I check if the game should go on? Basically I am checking for collision with wall or eating of the food.



```javascript
function update() {
    //console.log(snake.cells);
    document.addEventListener('keydown', function(event){
        if(event.defaultPrevented){
            return;
        }
        switch(event.key){
            case "ArrowDown":
                if(snake.dx==-1 && snake.dy==0)
                    gameend = true;
                snake.dx=1; snake.dy=0;
                break;
            case "ArrowUp":
                if(snake.dx==1 && snake.dy==0)
                    gameend = true;
                snake.dx=-1; snake.dy=0;
                break;
            case "ArrowLeft":
                if(snake.dx==0 && snake.dy==1)
                    gameend = true;
                snake.dx=0; snake.dy=-1;
                break;
            case "ArrowRight":
                if(snake.dx==0 && snake.dy==-1)
                    gameend = true;
                snake.dx=0; snake.dy=1;
                break;
            //case "p":
            //    if(go_on==true) go_on=false;
            //    else go_on=true;
            //    break;
            case "r":
                resetall = true;
                break;
```

Script.js – This snapshot shows the eventListeners and how I am using the keypresses to update snake data.

## RESULTS AND DISCUSSIONS

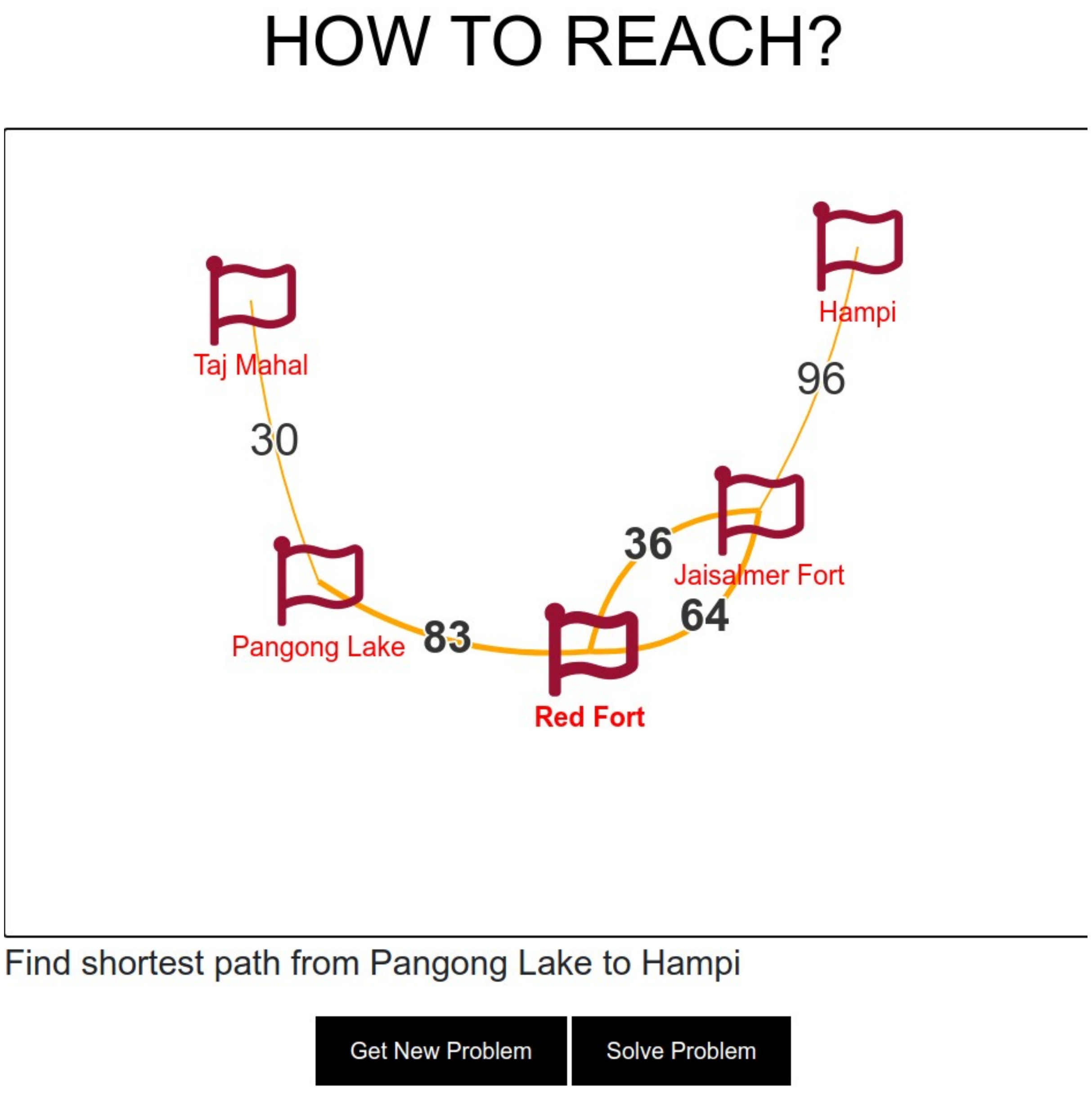
I was able to complete the given task within a deadline and showcase my desire to give back to the community by contributing in open source projects by keeping the game open for everybody to play. Within this duration I was able to grasp the deep understanding of JavaScript, Canvas element and its functions. The display of pictures correctly and without bugs once seemed to be a difficult task but with perseverance and support of instructor helped me to overcome this obstacle.

## PROJECT 2

### The Travel Planner

This project is a demonstration of the capabilities of vis.js and Dijkstra’s Algorithm. I randomly generate and display a graph of various tourist places in India and the paths with distances between them. Now the problem is to find the shortest path from one tourist place to another while at the same time visiting other tourist places on the way. To solve this, I make use of Dijkstra’s algorithm. This algorihtm returns me the shortest path from one tourist place to all other places, and later I reconstruct the path from this data.

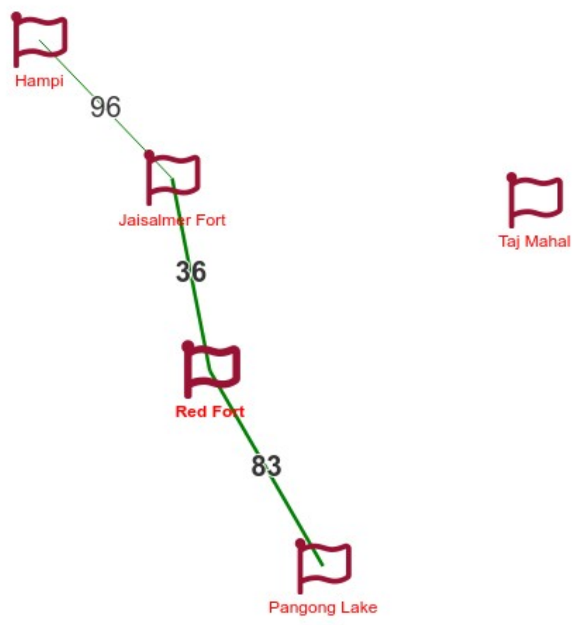
### Snapshots



Find shortest path from Pangong Lake to Hampi

The randomly generated problem and its graph.

# HOW TO REACH?

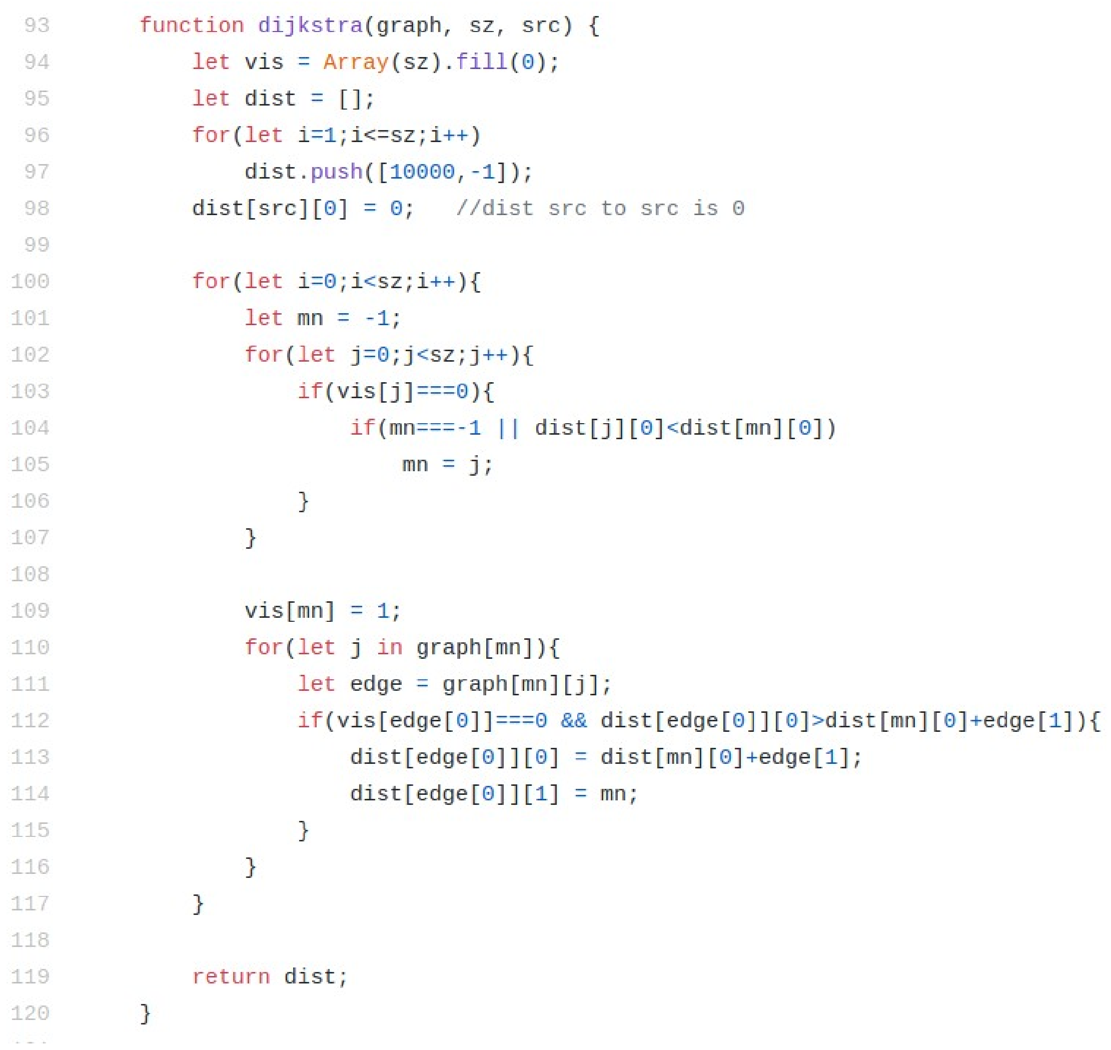


This graph lists the solution path to previous problem.

## Explanation and Code

Languages used were only HTML,CSS and JavaScript with the help of vis.js library.
Project live at https://nandeeshg.github.io/Travel-Planner/
Full code available at https://github.com/NandeeshG/Travel-Planner

**Dijkstra's algorithm**
Dijkstra's algorithm (or Dijkstra's Shortest Path First algorithm, SPF algorithm) is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later. The algorithm exists in many variants. Dijkstra's original algorithm found the shortest path between two given nodes, but a more common variant fixes a single node as the "source" node and finds shortest paths from the source to all other nodes in the graph, producing a shortest-path tree.

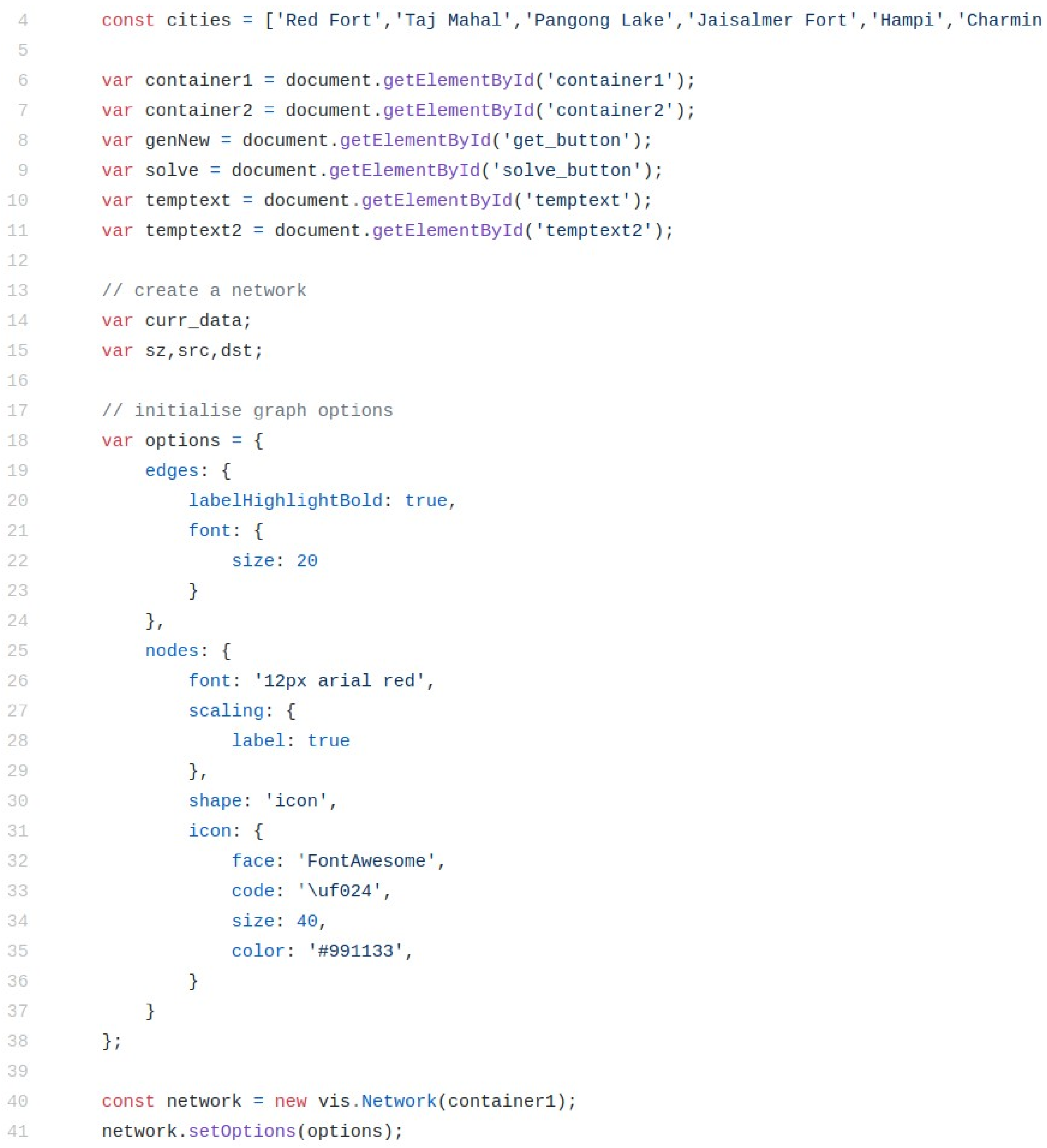


```
function dijkstra(graph, sz, src) {
    let vis = Array(sz).fill(0);
    let dist = [];
    for(let i=1;i<=sz;i++)
        dist.push([10000,-1]);
    dist[src][0] = 0;   //dist src to src is 0

    for(let i=0;i<sz;i++){
        let mn = -1;
        for(let j=0;j<sz;j++){
            if(vis[j]===0){
                if(mn===-1 || dist[j][0]<dist[mn][0])
                    mn = j;
            }
        }

        vis[mn] = 1;
        for(let j in graph[mn]){
            let edge = graph[mn][j];
            if(vis[edge[0]]===0 && dist[edge[0]][0]>dist[mn][0]+edge[1]){
                dist[edge[0]][0] = dist[mn][0]+edge[1];
                dist[edge[0]][1] = mn;
            }
        }
    }

    return dist;
}
```

Dijkstra’s Algorithm code that I implemented.

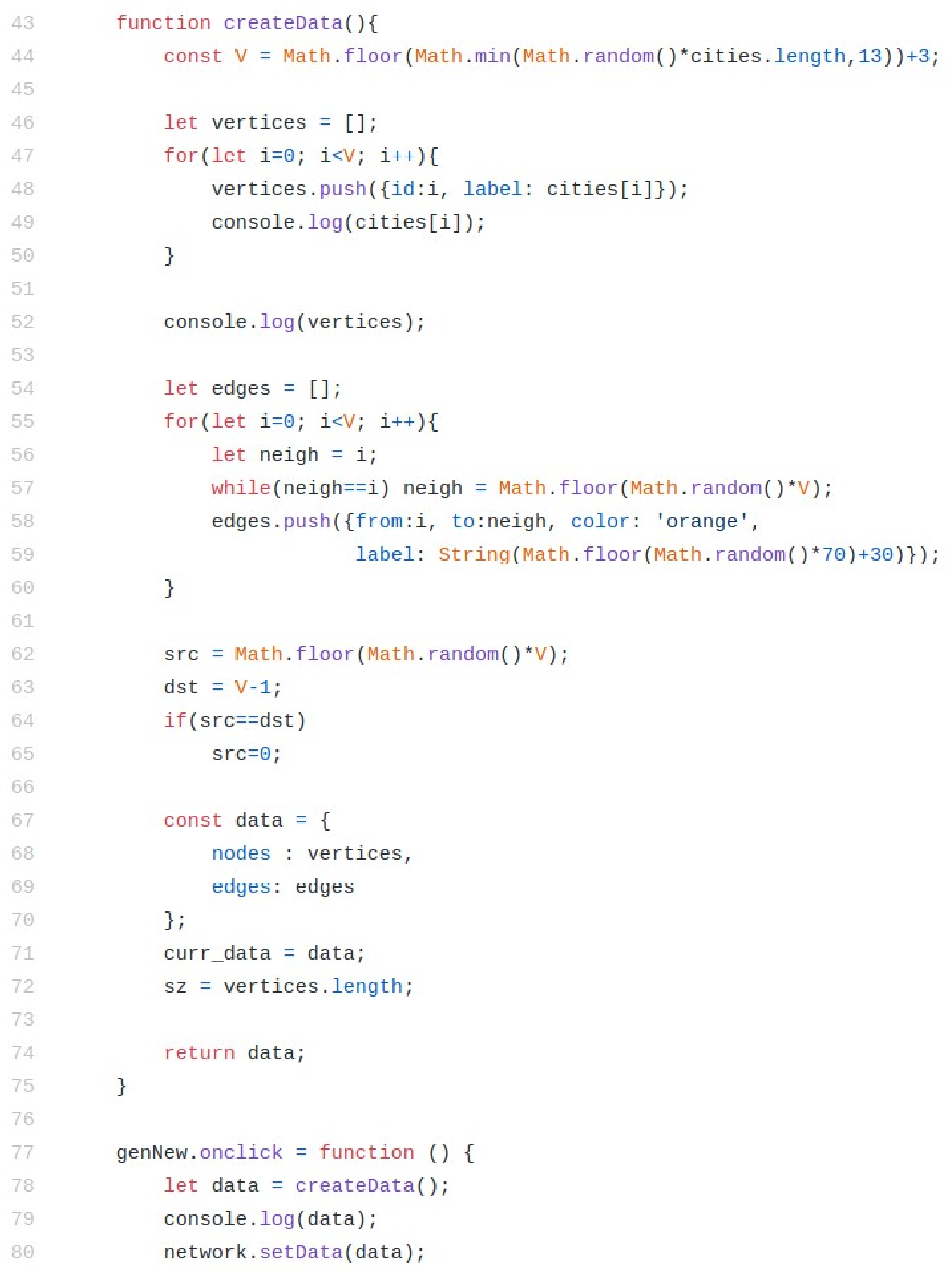
**vis.js**

A dynamic, browser based visualization library. The library is designed to be easy to use, to handle large amounts of dynamic data, and to enable manipulation of and interaction with the data. The library consists of the components DataSet, Timeline, Network, Graph2d and Graph3d.



```javascript
const cities = ['Red Fort','Taj Mahal','Pangong Lake','Jaisalmer Fort','Hampi','Charmin

var container1 = document.getElementById('container1');
var container2 = document.getElementById('container2');
var genNew = document.getElementById('get_button');
var solve = document.getElementById('solve_button');
var temptext = document.getElementById('temptext');
var temptext2 = document.getElementById('temptext2');

// create a network
var curr_data;
var sz,src,dst;

// initialise graph options
var options = {
    edges: {
        labelHighlightBold: true,
        font: {
            size: 20
        }
    },
    nodes: {
        font: '12px arial red',
        scaling: {
            label: true
        },
        shape: 'icon',
        icon: {
            face: 'FontAwesome',
            code: '\uf024',
            size: 40,
            color: '#991133',
        }
    }
};

const network = new vis.Network(container1);
network.setOptions(options);
```

The structure of graph and displaying it in vis.js



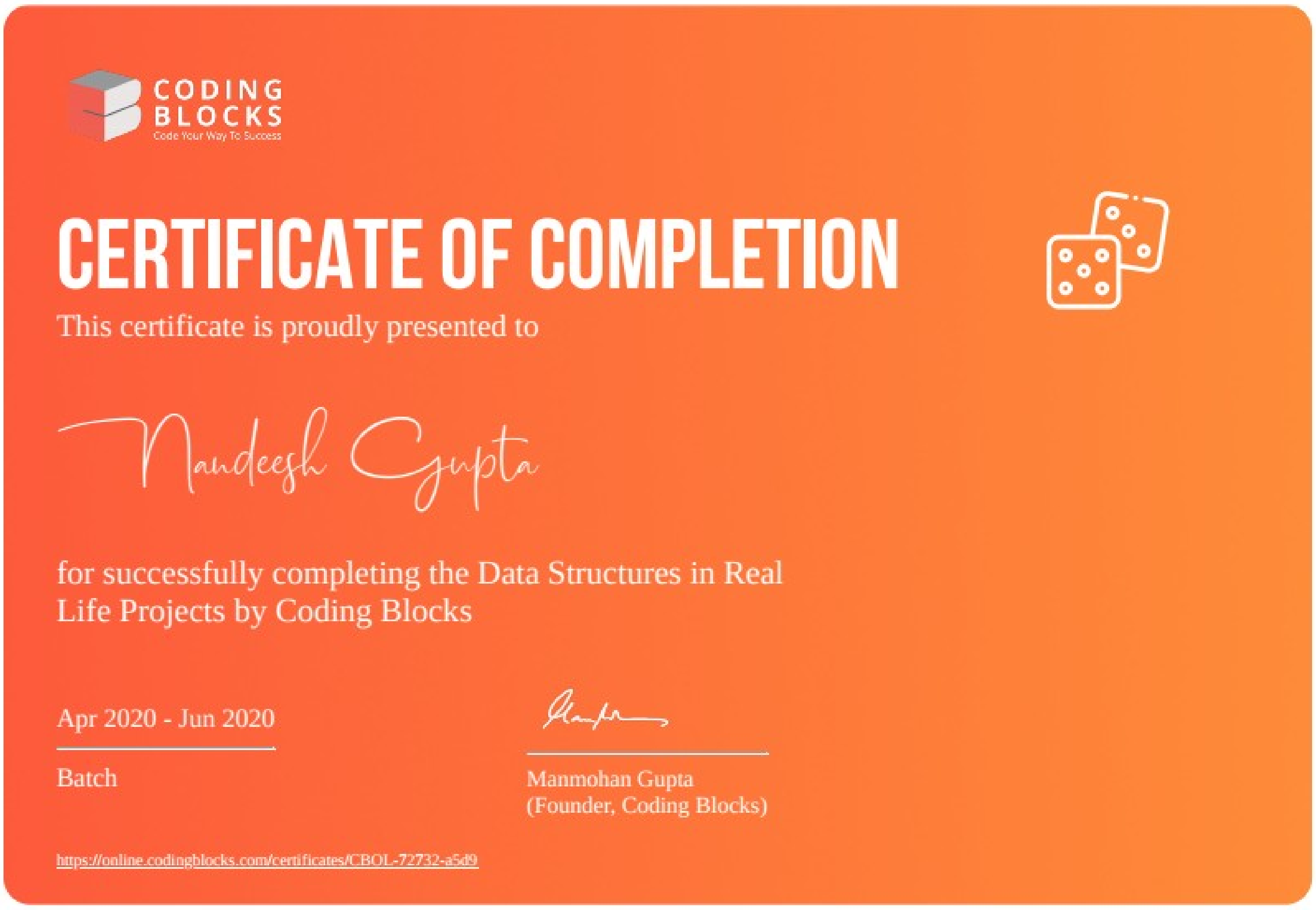
```
function createData(){
    const V = Math.floor(Math.min(Math.random()*cities.length,13))+3;

    let vertices = [];
    for(let i=0; i<V; i++){
        vertices.push({id:i, label: cities[i]});
        console.log(cities[i]);
    }

    console.log(vertices);

    let edges = [];
    for(let i=0; i<V; i++){
        let neigh = i;
        while(neigh==i) neigh = Math.floor(Math.random()*V);
        edges.push({from:i, to:neigh, color: 'orange',
                    label: String(Math.floor(Math.random()*70)+30)});
    }

    src = Math.floor(Math.random()*V);
    dst = V-1;
    if(src==dst)
        src=0;

    const data = {
        nodes : vertices,
        edges: edges
    };
    curr_data = data;
    sz = vertices.length;

    return data;
}

genNew.onclick = function () {
    let data = createData();
    console.log(data);
    network.setData(data);
```

Randomly generating graph data for vis.js

## RESULTS AND DISCUSSIONS

I was able to complete the given task within a deadline and showcase my desire to give back to the community by contributing in open source projects by keeping the app open for everybody to see and learn from. Within this duration I was able to grasp the deep understanding of JavaScript, vis.js library and its functions. The display of the graph correctly and without bugs once seemed to be a difficult task but with perseverance and support of instructor helped me to overcome this obstacle.

## CERTIFICATE



CODING BLOCKS
Code Your Way To Success

# CERTIFICATE OF COMPLETION

This certificate is proudly presented to

Nandeesh Gupta

for successfully completing the Data Structures in Real Life Projects by Coding Blocks

Apr 2020 - Jun 2020

Batch

[illegible]

Manmohan Gupta
(Founder, Coding Blocks)

https://online.codingblocks.com/certificates/CBOL-72732-a5d9

## SECTION 2

## Python for Everybody from The University of Michigan

(Accessing Web Data, Using Databases with Python, Using APIs)

# ABOUT THE COMPANY

## Coursera

Coursera was founded by Daphne Koller and Andrew Ng with a vision of providing life-transforming learning experiences to anyone, anywhere. It is now a leading online learning platform for higher education, where 66 million learners from around the world come to learn skills of the future. More than 200 of the world's top universities and industry educators partner with Coursera to offer courses, Specializations, certificates, and degree programs. 2,800 companies trust the company's enterprise platform Coursera for Business to transform their talent. Coursera for Government equips government employees and citizens with in-demand skills to build a competitive workforce. Coursera for Campus empowers any university to offer high-quality, job-relevant online education to students, alumni, faculty, and staff. Coursera is backed by leading investors that include Kleiner Perkins, New Enterprise Associates, Learn Capital, and SEEK Group.

## University of Michigan

The University of Michigan, often simply referred to as Michigan, is a public research university in Ann Arbor, Michigan. The university is Michigan's oldest; it was founded in 1817 in Detroit, as the Catholepistemiad, or the University of Michigania, 20 years before the territory became a state. The school was moved to Ann Arbor in 1837 onto 40 acres (16 ha) of what is now known as Central Campus. Since its establishment in Ann Arbor, the flagship university campus has expanded to include more than 584 major buildings with
a combined area of more than 34 million gross square feet (780 acres; 3.2 km2) spread out over a Central Campus and North Campus, two regional campuses in Flint and Dearborn, and a Center in Detroit. The university is a founding member of the Association of American Universities

## About Instructor

Charles Russell Severance

Clinical Professor

University of Michigan

Charles Severance (a.k.a. Dr. Chuck) is a Clinical Professor at the University Of Michigan School Of Information, where he teaches various technology-oriented courses including programming, database design, and Web development. Chuck has written a number of books including Using Google App Engine, and Python for Everybody. His research field is in the building of learning management systems such as Sakai, Moodle, Blackboard, ANGEL, and others. He was the chief architect for the Sakai Project, a learning management system used at about 300 schools worldwide and wrote the book Sakai: Free as in Freedom, that describes his experiences as one of the leaders of the project. In the mid-1990s he was the host of Internet: TCI, a national television talk show about the Internet that ran for several years on the TCI cable system. He was long-time a columnist for the IEEE Computer Magazine writing a monthly column called "Computing Conversations" that features video interviews with famous technology leaders and innovators.

# ABOUT THIS COURSE

## Python for Everybody - Course 1 (2 weeks)

This course aims to teach everyone the basics of programming computers using Python. We cover the basics of how one constructs a program from a series of simple instructions in Python. The course has no pre-requisites and avoids all but the simplest mathematics. Anyone with moderate computer experience should be able to master the materials in this course. This course will cover Chapters 1-5 of the textbook “Python for Everybody”. Once a student completes this course, they will be ready to take more advanced programming courses. This course covers Python 3.

## Python Data Structures – Course 2 (2 weeks)

This course will introduce the core data structures of the Python programming language. We will move past the basics of procedural programming and explore how we can use the Python built-in data structures such as lists, dictionaries, and tuples to perform increasingly complex data analysis. This course will cover Chapters 6-10 of the textbook “Python for Everybody”. This course covers Python 3.

## Using Python to Access Web Data – Course 3 (2 weeks)

This course will show how one can treat the Internet as a source of data. We will scrape, parse, and read web data as well as access data using web APIs. We will work with HTML, XML, and JSON data formats in Python. This course will cover Chapters 11-13 of the textbook “Python for Everybody”. To succeed in this course, you should be familiar with the material covered in Chapters 1-10 of the textbook and the first two courses in this specialization. These topics include variables and expressions, conditional execution (loops, branching, and try/except), functions, Python data structures (strings, lists, dictionaries, and tuples), and manipulating files. This course covers Python 3.

## Using Databases with Python – Course 4 (2 weeks)

This course will introduce students to the basics of the Structured Query Language (SQL) as well as basic database design for storing data as part of a multi-step data gathering, analysis, and processing effort. The course will use SQLite3 as its database. We will also build web crawlers and multi-step data gathering and visualization processes. We will use the D3.js library to do basic data visualization. This course will cover Chapters 14-15 of the book “Python for Everybody”. To succeed in this course, you should be familiar with the material covered in Chapters 1-13 of the textbook and the first three courses in this specialization. This course covers Python 3.

# TOOLS AND TECHNOLOGIES USED

## Python3

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system with reference counting. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3.

**Python Indentation**

Indentation refers to the spaces at the beginning of a code line. Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important. Python uses indentation to indicate a block of code.

**Python Data Types**

In programming, data type is an important concept. Variables can store data of different types, and different types can do different things. Python has the following data types built-in by default, in these categories:

Text Type: str
Numeric Types: int , float , complex
Sequence Types: list, tuple, range
Mapping Type: dict
Set Types: set, frozenset
Boolean Type: bool

The data type of any object by using the type() function.

**Python Conditions and If statements**

Python supports the usual logical conditions from mathematics:

Equals: a == b

Not Equals: a != b

Less than: a < b

Less than or equal to: a <= b

Greater than: a > b

Greater than or equal to: a >= b

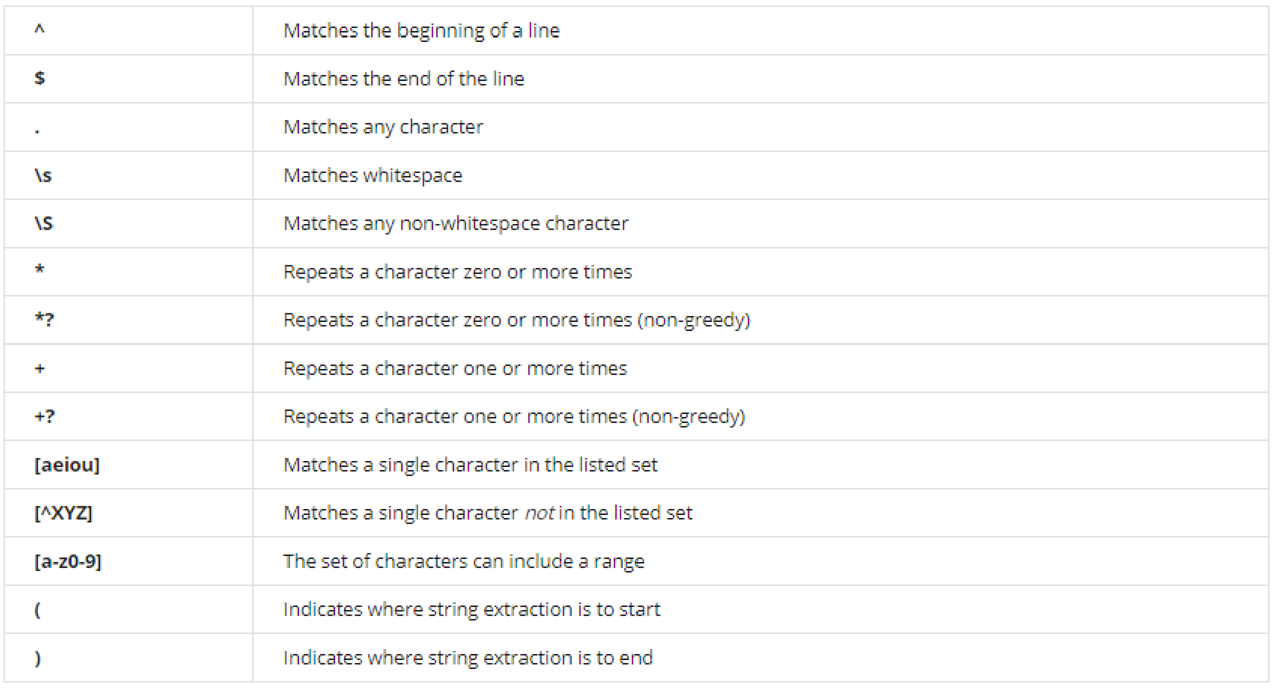
These conditions can be used in several ways, most commonly in "if statements" and loops. An "if statement" is written by using the if keyword. The elif keyword is pythons way of saying "if the previous conditions were not true, then try this condition”. The else keyword catches anything which isn't caught by the preceding conditions.

**Python Try Except**

The try block lets you test a block of code for errors. The except block lets you handle the error. The finally block lets you execute code, regardless of the result of the try- and except blocks.

## Regular Expressions

Regular expressions are a very specialized language that allow us to succinctly search strings and extract data from strings. Regular expressions are a language unto themselves. It is not essential to know how to use regular expressions, but they can be quite useful and powerful.



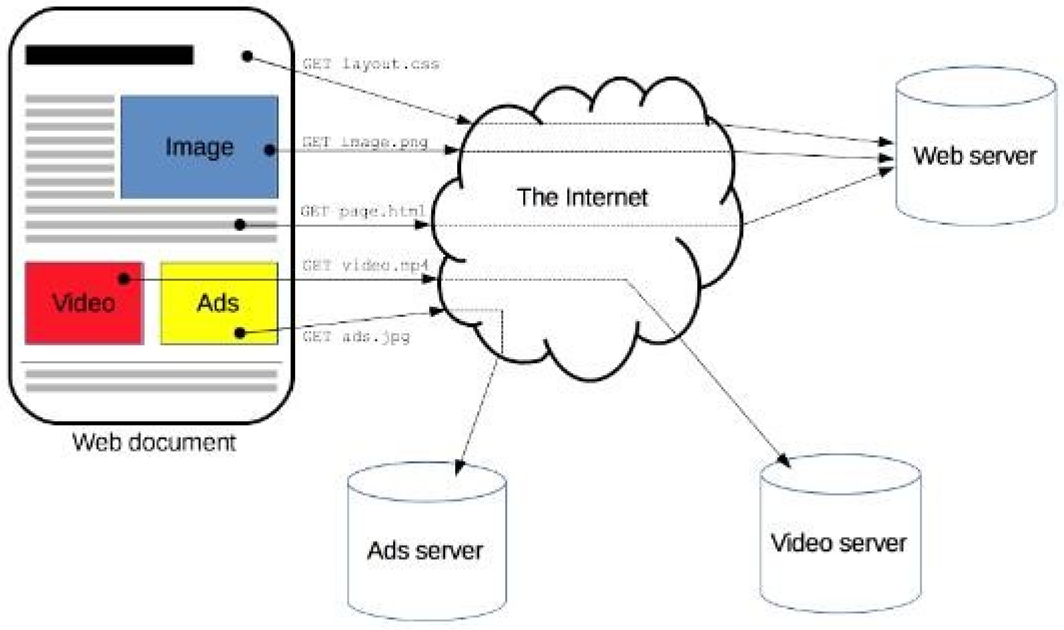
| | |
|---|---|
| **^** | Matches the beginning of a line |
| **$** | Matches the end of the line |
| **.** | Matches any character |
| **\s** | Matches whitespace |
| **\S** | Matches any non-whitespace character |
| ***** | Repeats a character zero or more times |
| ***?** | Repeats a character zero or more times (non-greedy) |
| **+** | Repeats a character one or more times |
| **+?** | Repeats a character one or more times (non-greedy) |
| **[aeiou]** | Matches a single character in the listed set |
| **[^XYZ]** | Matches a single character *not* in the listed set |
| **[a-z0-9]** | The set of characters can include a range |
| **(** | Indicates where string extraction is to start |
| **)** | Indicates where string extraction is to end |

## Networks and Sockets

In this section I learnt about the protocols that web browsers use to retrieve documents and web applications use to interact with **Application Program Interfaces (APIs)**.

**HTTP PROTOCOL**

HTTP is a protocol which allows the fetching of resources, such as HTML documents. It is the foundation of any data exchange on the Web and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser. A complete document is reconstructed from the different sub-documents fetched, for instance text, layout description, images, videos, scripts, and more.



**Sockets**

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server.
They are the real backbones behind web browsing. In simpler terms there is a server and a client.
Socket programming is started by importing the socket library and making a simple socket.
Socket Connection:-

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

## Urllib

Urllib module is the URL handling module for python. It is used to fetch URLs (Uniform Resource Locators). It uses the urlopen function and is able to fetch URLs using a variety of different protocols.

Urllib is a package that collects several modules for working with URLs, such as:

- urllib.request for opening and reading.
- urllib.parse for parsing URLs
- urllib.error for the exceptions raised
- urllib.robotparser for parsing robot.txt files

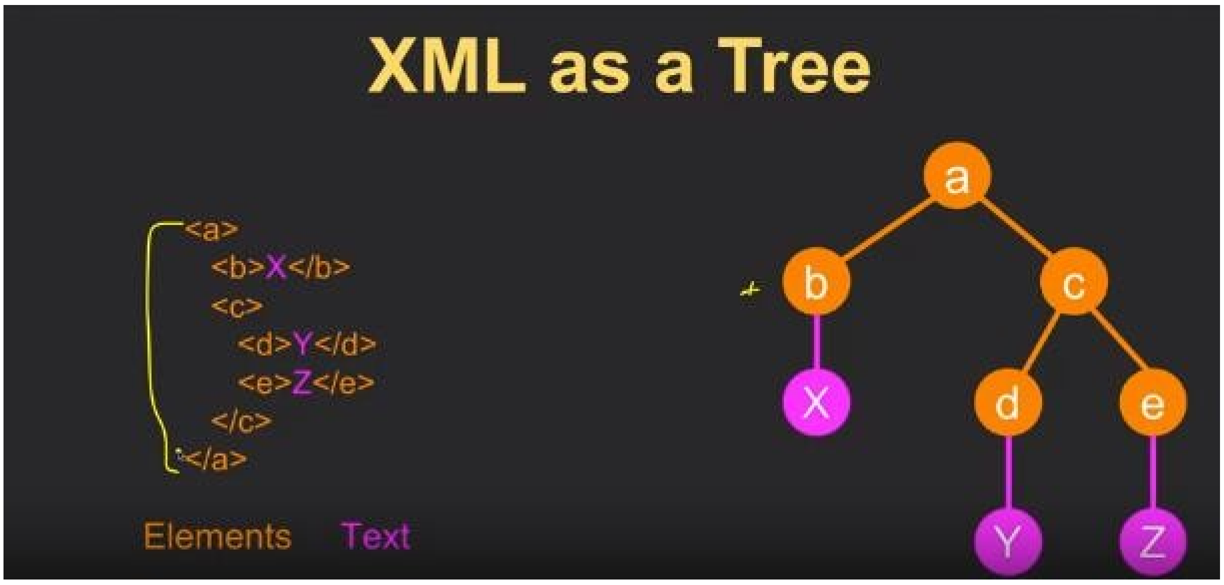
## XML Parsing

XML is a software- and hardware-independent tool for storing and transporting data.

- XML stands for eXtensible Markup Language
- XML is a markup language much like HTML
- XML was designed to store and transport data
- XML was designed to be self-descriptive
- XML is a W3C Recommendation

Most XML applications will work as expected even if new data is added (or

removed).



## Beautiful Soup (HTML parser)

Beautiful Soup is a Python package for parsing HTML and XML documents (including having malformed markup, i.e. non-closed tags, so named after tag soup). It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping.

It is available for Python 2.7 and Python 3.

## SQLite

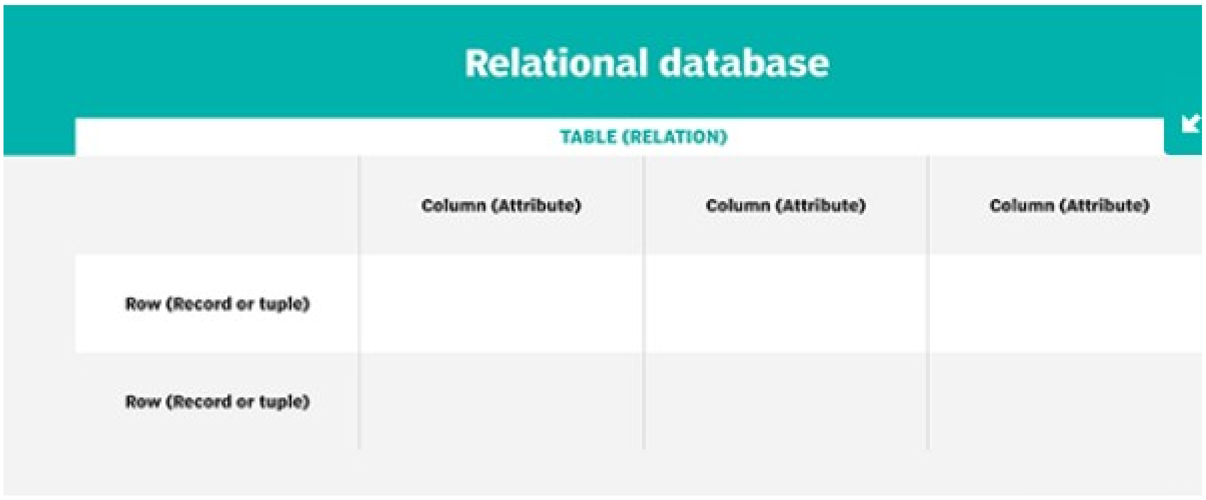
SQLite is a relational database management system (RDBMS) contained in a C library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program.

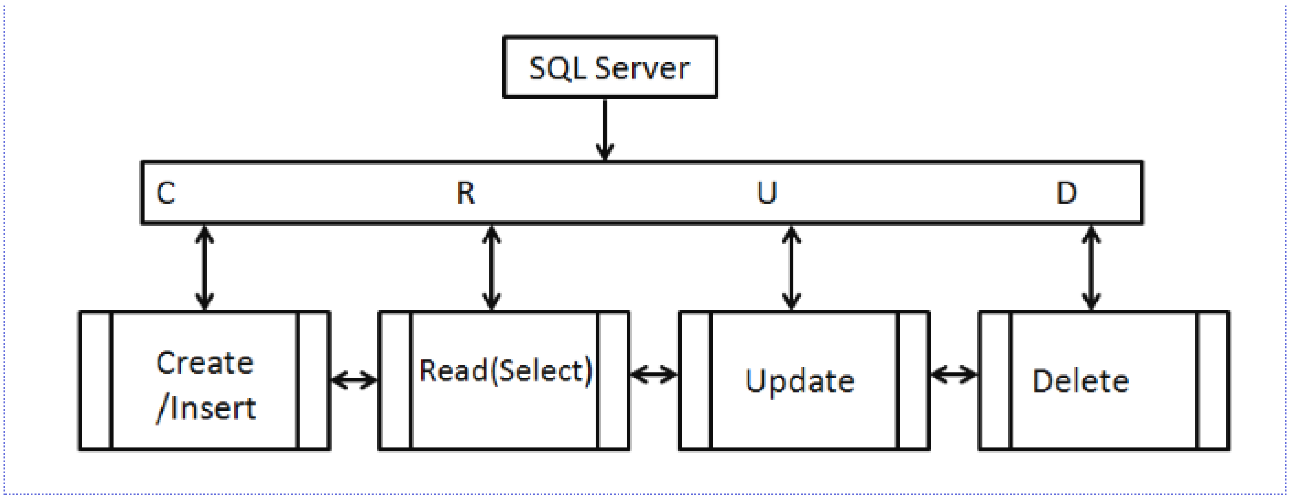
SQLite is ACID-compliant and implements most of the SQL standard, generally following PostgreSQL syntax. However, SQLite uses a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity.This means that one can, for example, insert a string into a column defined as an integer. SQLite will attempt to convert data between formats where appropriate, the string "123" into an integer in this case, but does not guarantee such conversions and will store the data as-is if such a conversion is not possible.

SQLite is a popular choice as embedded database software for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems (such as mobile phones), among others. SQLite has bindings to many programming languages.

## Relational Databases

A relational database is a collection of information that organizes data points with defined relationships for easy access. In the relational database model, the data structures -- including data tables, indexes and views -- remain separate from the physical storage, allowing administrators to edit the physical data storage without affecting the logical data structure. **CRUD** is an acronym that stands for Create, Read, Update, and Delete. These are the four most basic operations that can be performed with most traditional database systems and they are the backbone for interacting with any database.





# PROJECT 1

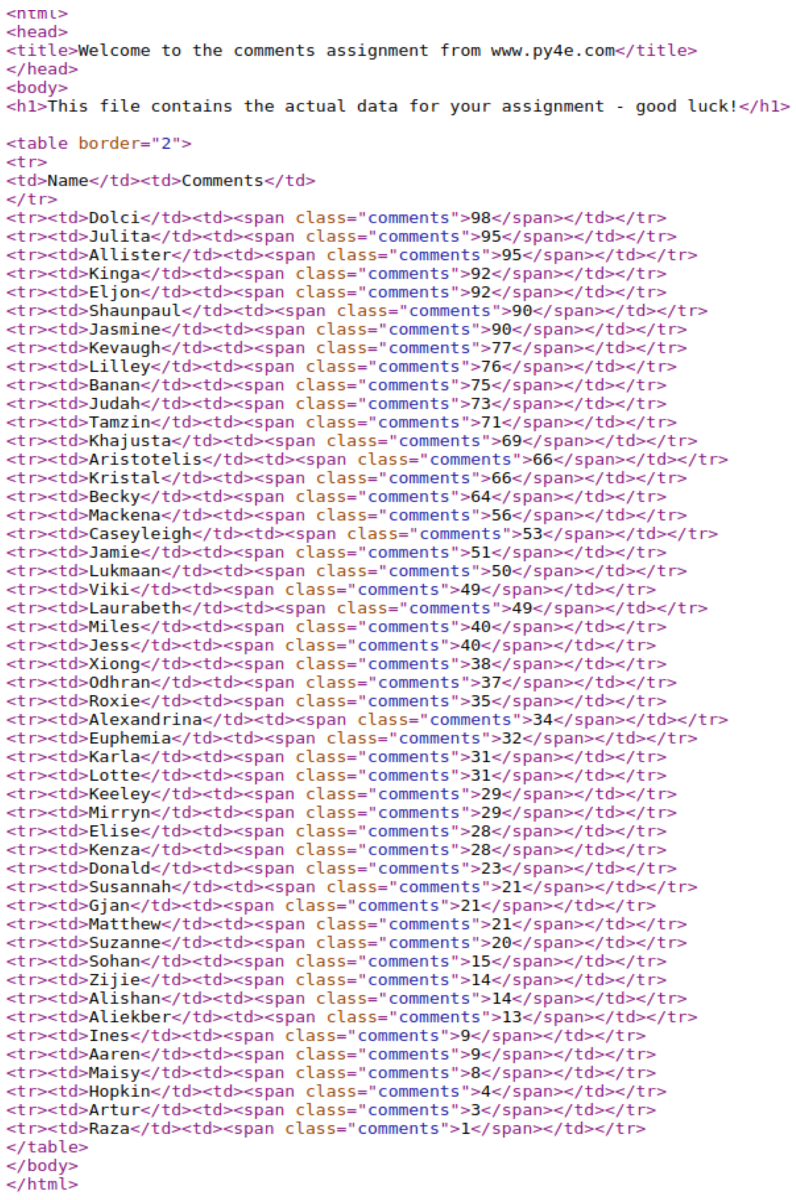
## Web Scraping and Crawling

By using Beautifulsoup library in python, I created an a web crawling and web scraping application. It creates a parse tree for web pages that can be used to extract data from HTML, which is useful for web scraping.

There were two use cases -

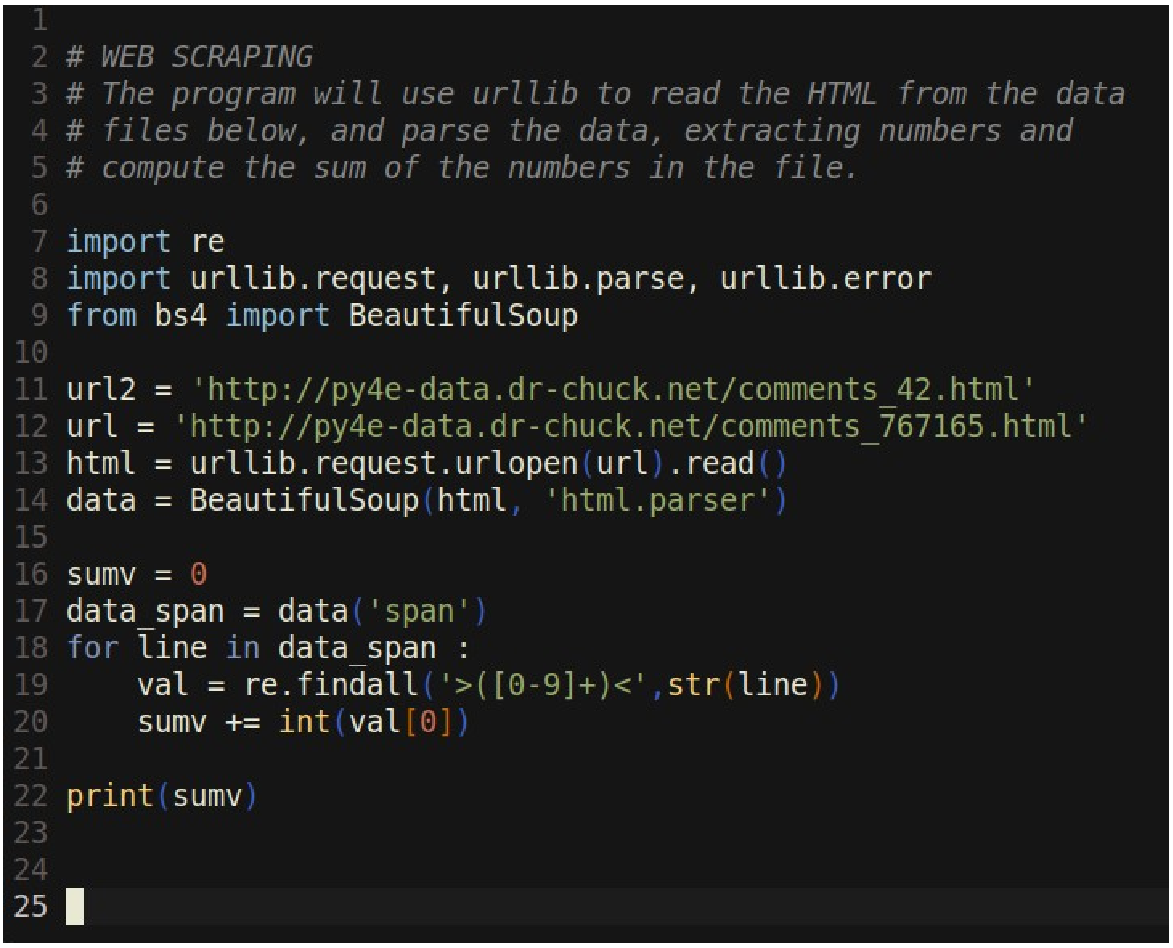
1. Extracting integer values from a webpage's specific HTML element and adding all of those values.
2. Following different links again and again on the web , to simulate a web crawler

### Snapshots – Web Scraping



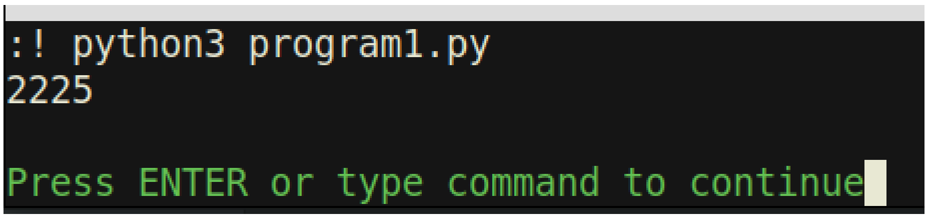
```
<html>
<head>
<title>Welcome to the comments assignment from www.py4e.com</title>
</head>
<body>
<h1>This file contains the actual data for your assignment - good luck!</h1>

<table border="2">
<tr>
<td>Name</td><td>Comments</td>
</tr>
<tr><td>Dolci</td><td><span class="comments">98</span></td></tr>
<tr><td>Julita</td><td><span class="comments">95</span></td></tr>
<tr><td>Allister</td><td><span class="comments">95</span></td></tr>
<tr><td>Kinga</td><td><span class="comments">92</span></td></tr>
<tr><td>Eljon</td><td><span class="comments">92</span></td></tr>
<tr><td>Shaunpaul</td><td><span class="comments">90</span></td></tr>
<tr><td>Jasmine</td><td><span class="comments">90</span></td></tr>
<tr><td>Kevaugh</td><td><span class="comments">77</span></td></tr>
<tr><td>Lilley</td><td><span class="comments">76</span></td></tr>
<tr><td>Banan</td><td><span class="comments">75</span></td></tr>
<tr><td>Judah</td><td><span class="comments">73</span></td></tr>
<tr><td>Tamzin</td><td><span class="comments">71</span></td></tr>
<tr><td>Khajusta</td><td><span class="comments">69</span></td></tr>
<tr><td>Aristotelis</td><td><span class="comments">66</span></td></tr>
<tr><td>Kristal</td><td><span class="comments">66</span></td></tr>
<tr><td>Becky</td><td><span class="comments">64</span></td></tr>
<tr><td>Mackena</td><td><span class="comments">56</span></td></tr>
<tr><td>Caseyleigh</td><td><span class="comments">53</span></td></tr>
<tr><td>Jamie</td><td><span class="comments">51</span></td></tr>
<tr><td>Lukmaan</td><td><span class="comments">50</span></td></tr>
<tr><td>Viki</td><td><span class="comments">49</span></td></tr>
<tr><td>Laurabeth</td><td><span class="comments">49</span></td></tr>
<tr><td>Miles</td><td><span class="comments">40</span></td></tr>
<tr><td>Jess</td><td><span class="comments">40</span></td></tr>
<tr><td>Xiong</td><td><span class="comments">38</span></td></tr>
<tr><td>Odhran</td><td><span class="comments">37</span></td></tr>
<tr><td>Roxie</td><td><span class="comments">35</span></td></tr>
<tr><td>Alexandrina</td><td><span class="comments">34</span></td></tr>
<tr><td>Euphemia</td><td><span class="comments">32</span></td></tr>
<tr><td>Karla</td><td><span class="comments">31</span></td></tr>
<tr><td>Lotte</td><td><span class="comments">31</span></td></tr>
<tr><td>Keeley</td><td><span class="comments">29</span></td></tr>
<tr><td>Mirryn</td><td><span class="comments">29</span></td></tr>
<tr><td>Elise</td><td><span class="comments">28</span></td></tr>
<tr><td>Kenza</td><td><span class="comments">28</span></td></tr>
<tr><td>Donald</td><td><span class="comments">23</span></td></tr>
<tr><td>Susannah</td><td><span class="comments">21</span></td></tr>
<tr><td>Gjan</td><td><span class="comments">21</span></td></tr>
<tr><td>Matthew</td><td><span class="comments">21</span></td></tr>
<tr><td>Suzanne</td><td><span class="comments">20</span></td></tr>
<tr><td>Sohan</td><td><span class="comments">15</span></td></tr>
<tr><td>Zijie</td><td><span class="comments">14</span></td></tr>
<tr><td>Alishan</td><td><span class="comments">14</span></td></tr>
<tr><td>Aliekber</td><td><span class="comments">13</span></td></tr>
<tr><td>Ines</td><td><span class="comments">9</span></td></tr>
<tr><td>Aaren</td><td><span class="comments">9</span></td></tr>
<tr><td>Maisy</td><td><span class="comments">8</span></td></tr>
<tr><td>Hopkin</td><td><span class="comments">4</span></td></tr>
<tr><td>Artur</td><td><span class="comments">3</span></td></tr>
<tr><td>Raza</td><td><span class="comments">1</span></td></tr>
</table>
</body>
</html>
```

HTML of the page I'll be scraping ahead.



```python

# WEB SCRAPING
# The program will use urllib to read the HTML from the data
# files below, and parse the data, extracting numbers and
# compute the sum of the numbers in the file.

import re
import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup

url2 = 'http://py4e-data.dr-chuck.net/comments_42.html'
url = 'http://py4e-data.dr-chuck.net/comments_767165.html'
html = urllib.request.urlopen(url).read()
data = BeautifulSoup(html, 'html.parser')

sumv = 0
data_span = data('span')
for line in data_span :
    val = re.findall('>([0-9]+)<',str(line))
    sumv += int(val[0])

print(sumv)
```

Using URLLIB and beautiful soup, I access the webpage. Then a simple loop through the elements allow me to calculate the desired values.

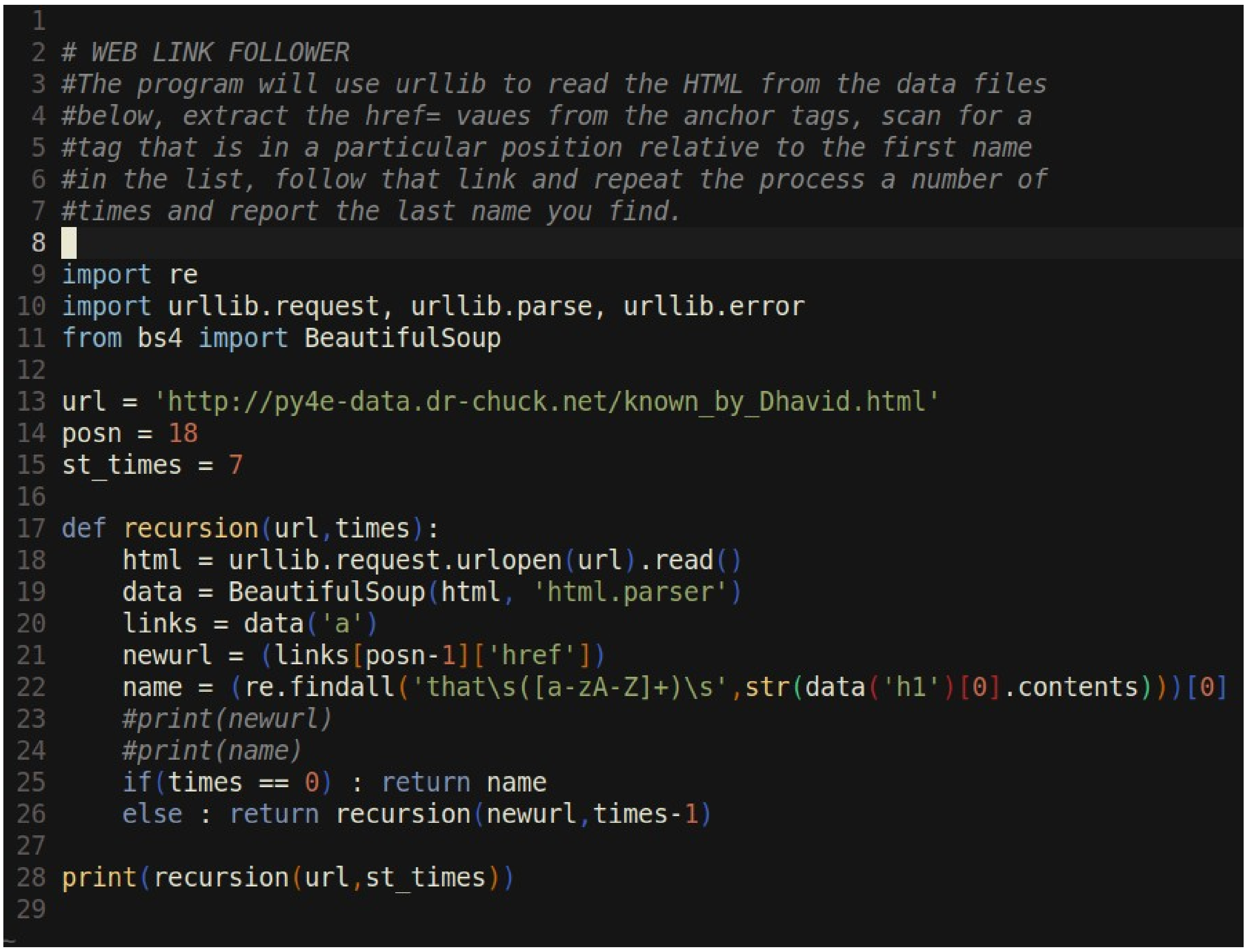


```
:! python3 program1.py
2225

Press ENTER or type command to continue
```

Output.

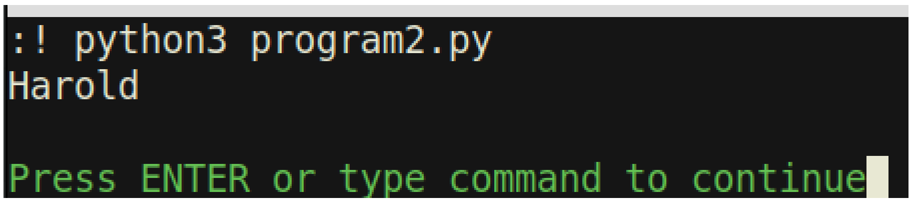
The url I accessed -
[http://py4e-data.dr-chuck.net/comments_767165.html](http://py4e-data.dr-chuck.net/comments_767165.html)

## Snapshots – Web Crawling



```

# WEB LINK FOLLOWER
#The program will use urllib to read the HTML from the data files
#below, extract the href= vaues from the anchor tags, scan for a
#tag that is in a particular position relative to the first name
#in the list, follow that link and repeat the process a number of
#times and report the last name you find.

import re
import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup

url = 'http://py4e-data.dr-chuck.net/known_by_Dhavid.html'
posn = 18
st_times = 7

def recursion(url,times):
    html = urllib.request.urlopen(url).read()
    data = BeautifulSoup(html, 'html.parser')
    links = data('a')
    newurl = (links[posn-1]['href'])
    name = (re.findall('that\s([a-zA-Z]+)\s',str(data('h1')[0].contents)))[0]
    #print(newurl)
    #print(name)
    if(times == 0) : return name
    else : return recursion(newurl,times-1)

print(recursion(url,st_times))

```

Here I made use of URLLIB, RegEx library, and Beautiful Soup to access the web page, extract name information and then recursively follow the set link from the page. In the end the program returns the name for the number of times of repetitions we commanded it to do.



```
:! python3 program2.py
Harold

Press ENTER or type command to continue
```

The output of above web crawler.

Webpage I crawled - http://py4e-data.dr-chuck.net/known_by_Dhavid.html

## RESULTS AND DISCUSSIONS

I was able to complete the given task within a deadline. With this mini-project I was able to grasp the working of RegEx, URLLIB, BeautifulSoup and its functions to use for Web Scraping and Web Crawling.

# PROJECT 2

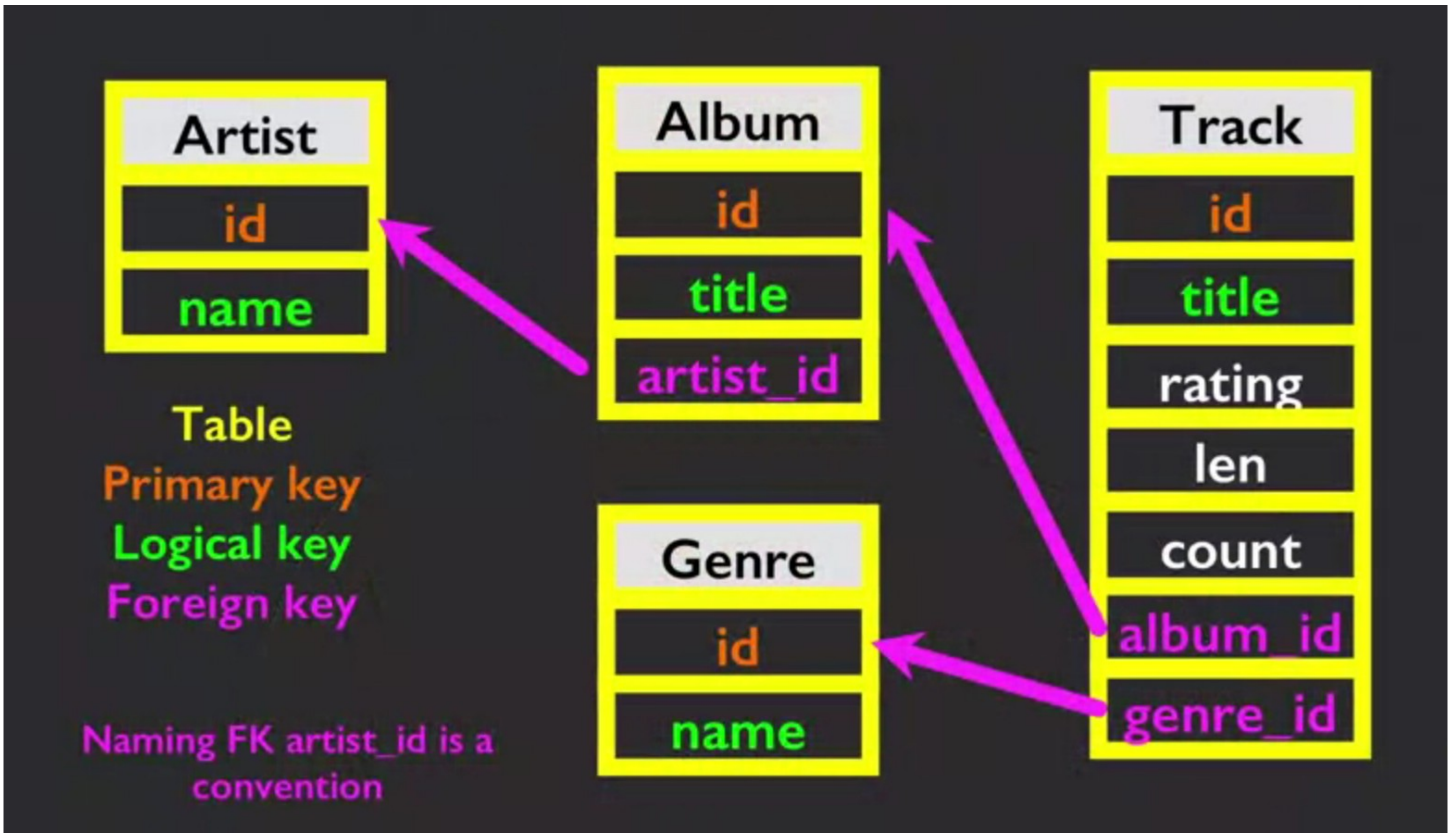
## SQLite Music Database

Using SQLite3 library in python I created a Database of user music library in local databse. SQLite is a RDBMS which is not a client–server database engine. Rather, it is embedded into the end program, here as a python library.
Sqlite3 allows us to directly Create, Remove, Update Data from the Database using python.

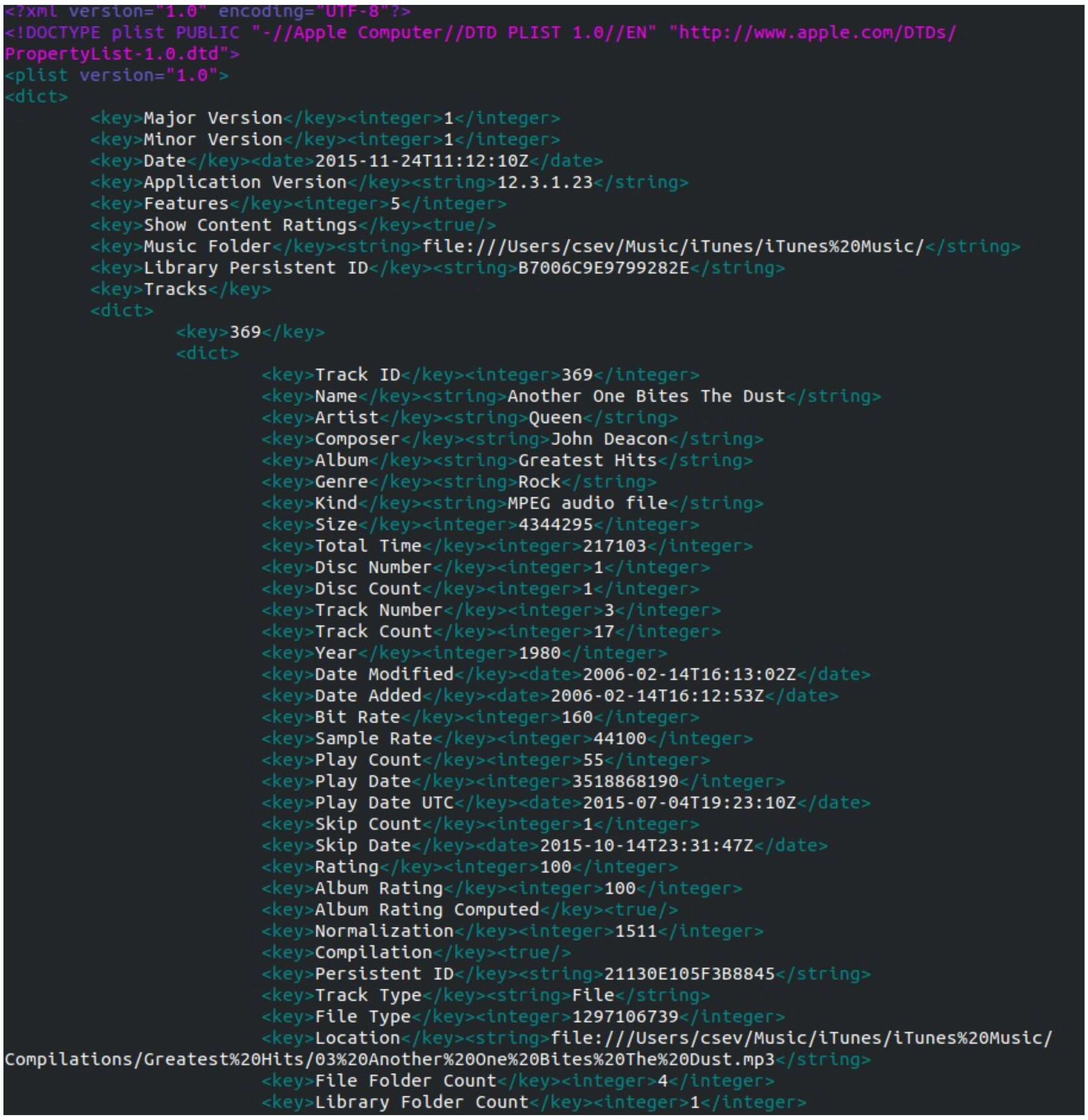
I learnt reading XML and JSON with the usage of SQLite3 to fill the Database

Queries can be done both from python or SQlite browser.

### Snapshots

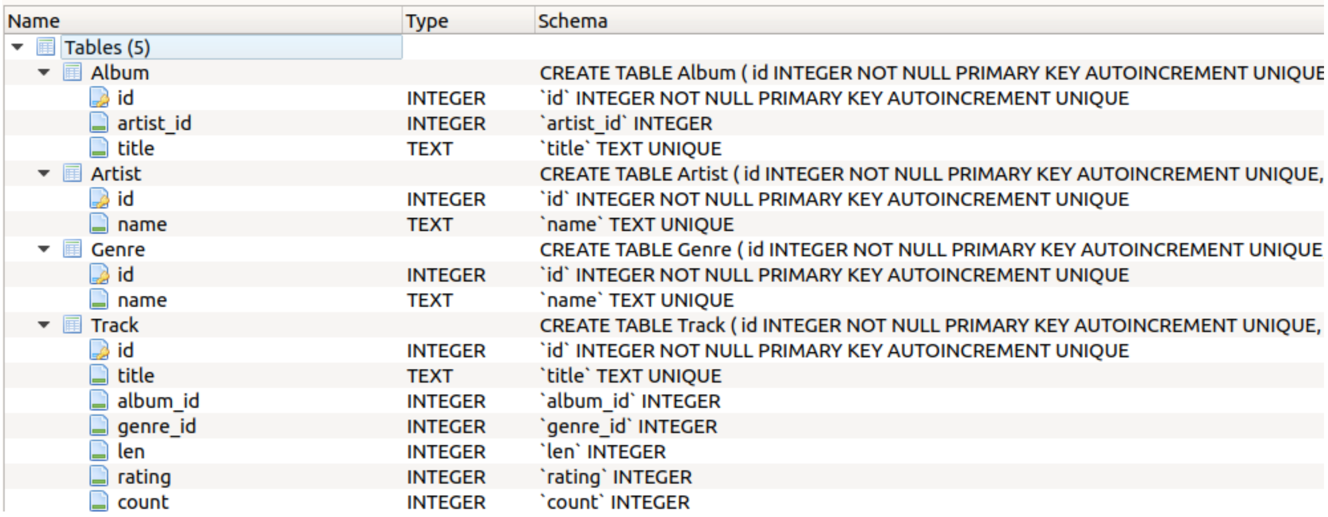


The Database Relational Model



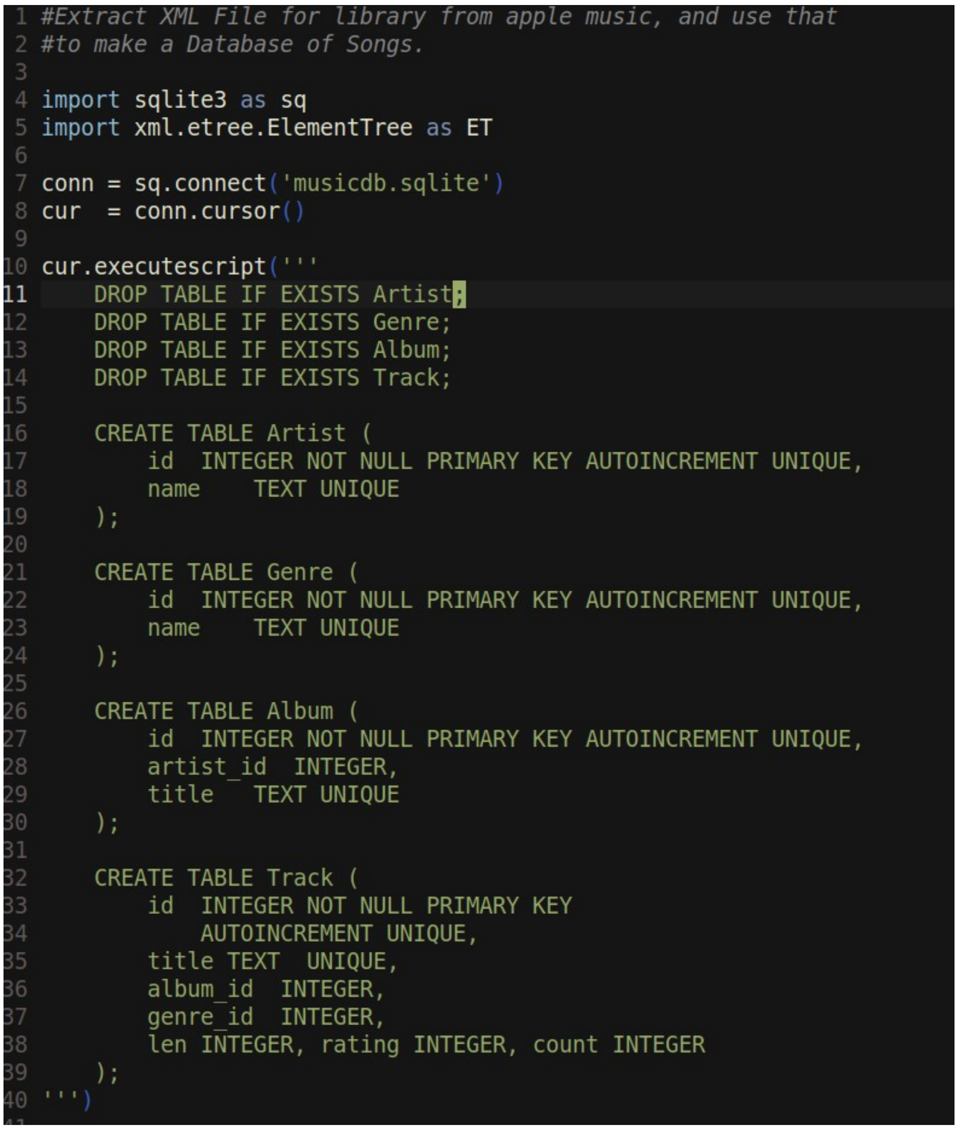
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>Major Version</key><integer>1</integer>
        <key>Minor Version</key><integer>1</integer>
        <key>Date</key><date>2015-11-24T11:12:10Z</date>
        <key>Application Version</key><string>12.3.1.23</string>
        <key>Features</key><integer>5</integer>
        <key>Show Content Ratings</key><true/>
        <key>Music Folder</key><string>file:///Users/csev/Music/iTunes/iTunes%20Music/</string>
        <key>Library Persistent ID</key><string>B7006C9E9799282E</string>
        <key>Tracks</key>
        <dict>
                <key>369</key>
                <dict>
                        <key>Track ID</key><integer>369</integer>
                        <key>Name</key><string>Another One Bites The Dust</string>
                        <key>Artist</key><string>Queen</string>
                        <key>Composer</key><string>John Deacon</string>
                        <key>Album</key><string>Greatest Hits</string>
                        <key>Genre</key><string>Rock</string>
                        <key>Kind</key><string>MPEG audio file</string>
                        <key>Size</key><integer>4344295</integer>
                        <key>Total Time</key><integer>217103</integer>
                        <key>Disc Number</key><integer>1</integer>
                        <key>Disc Count</key><integer>1</integer>
                        <key>Track Number</key><integer>3</integer>
                        <key>Track Count</key><integer>17</integer>
                        <key>Year</key><integer>1980</integer>
                        <key>Date Modified</key><date>2006-02-14T16:13:02Z</date>
                        <key>Date Added</key><date>2006-02-14T16:12:53Z</date>
                        <key>Bit Rate</key><integer>160</integer>
                        <key>Sample Rate</key><integer>44100</integer>
                        <key>Play Count</key><integer>55</integer>
                        <key>Play Date</key><integer>3518868190</integer>
                        <key>Play Date UTC</key><date>2015-07-04T19:23:10Z</date>
                        <key>Skip Count</key><integer>1</integer>
                        <key>Skip Date</key><date>2015-10-14T23:31:47Z</date>
                        <key>Rating</key><integer>100</integer>
                        <key>Album Rating</key><integer>100</integer>
                        <key>Album Rating Computed</key><true/>
                        <key>Normalization</key><integer>1511</integer>
                        <key>Compilation</key><true/>
                        <key>Persistent ID</key><string>21130E105F3B8845</string>
                        <key>Track Type</key><string>File</string>
                        <key>File Type</key><integer>1297106739</integer>
                        <key>Location</key><string>file:///Users/csev/Music/iTunes/iTunes%20Music/
Compilations/Greatest%20Hits/03%20Another%20One%20Bites%20The%20Dust.mp3</string>
                        <key>File Folder Count</key><integer>4</integer>
                        <key>Library Folder Count</key><integer>1</integer>
```

Structure of the XML Document I used to built my Database



| Name | Type | Schema |
|---|---|---|
| Tables (5) | | |
| Album | | CREATE TABLE Album ( id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE |
| id | INTEGER | \`id\` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE |
| artist_id | INTEGER | \`artist_id\` INTEGER |
| title | TEXT | \`title\` TEXT UNIQUE |
| Artist | | CREATE TABLE Artist ( id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, |
| id | INTEGER | \`id\` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE |
| name | TEXT | \`name\` TEXT UNIQUE |
| Genre | | CREATE TABLE Genre ( id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE |
| id | INTEGER | \`id\` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE |
| name | TEXT | \`name\` TEXT UNIQUE |
| Track | | CREATE TABLE Track ( id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, |
| id | INTEGER | \`id\` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE |
| title | TEXT | \`title\` TEXT UNIQUE |
| album_id | INTEGER | \`album_id\` INTEGER |
| genre_id | INTEGER | \`genre_id\` INTEGER |
| len | INTEGER | \`len\` INTEGER |
| rating | INTEGER | \`rating\` INTEGER |
| count | INTEGER | \`count\` INTEGER |

Final Databse that was created



```
#Extract XML File for library from apple music, and use that
#to make a Database of Songs.

import sqlite3 as sq
import xml.etree.ElementTree as ET

conn = sq.connect('musicdb.sqlite')
cur  = conn.cursor()

cur.executescript('''
    DROP TABLE IF EXISTS Artist;
    DROP TABLE IF EXISTS Genre;
    DROP TABLE IF EXISTS Album;
    DROP TABLE IF EXISTS Track;

    CREATE TABLE Artist (
        id  INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
        name    TEXT UNIQUE
    );

    CREATE TABLE Genre (
        id  INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
        name    TEXT UNIQUE
    );

    CREATE TABLE Album (
        id  INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
        artist_id  INTEGER,
        title   TEXT UNIQUE
    );

    CREATE TABLE Track (
        id  INTEGER NOT NULL PRIMARY KEY
            AUTOINCREMENT UNIQUE,
        title TEXT  UNIQUE,
        album_id  INTEGER,
        genre_id  INTEGER,
        len INTEGER, rating INTEGER, count INTEGER
    );
''')
```

Snapshot of some part of the code. Full code available at https://github.com/NandeeshG/training_presentation/blob/master/SQLite_Music_Database/program.py

## RESULTS AND DISCUSSIONS

I was able to complete the given task within a deadline. With this mini-project I was able to grasp the working of RDBMS, SQlite3, XML parsing and its functions to use for Database creation and working.

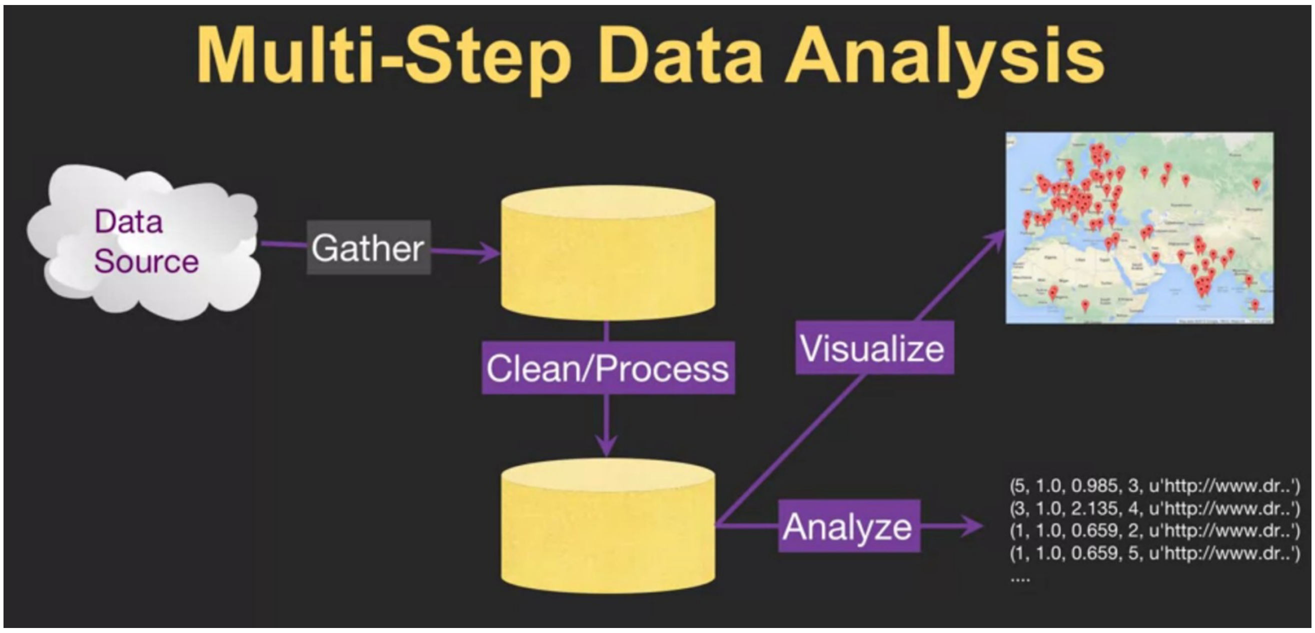
# PROJECT 3

## Visualize Tourist Places

This application works by collecting the location information of some famous tourist places of India  from the Google Maps API. It stores all of that data into a local Database. After the database is filled, it plots the data onto a map and displays the webpage.

Gathering is restartable and happens many times instead of once. This is helpful in case of some crash or even due to API Rate limits. The Database helps us in this manner by keeping the data yet received safe and organised.

To visualize, I have taken help of Google API and the tutorials from there.

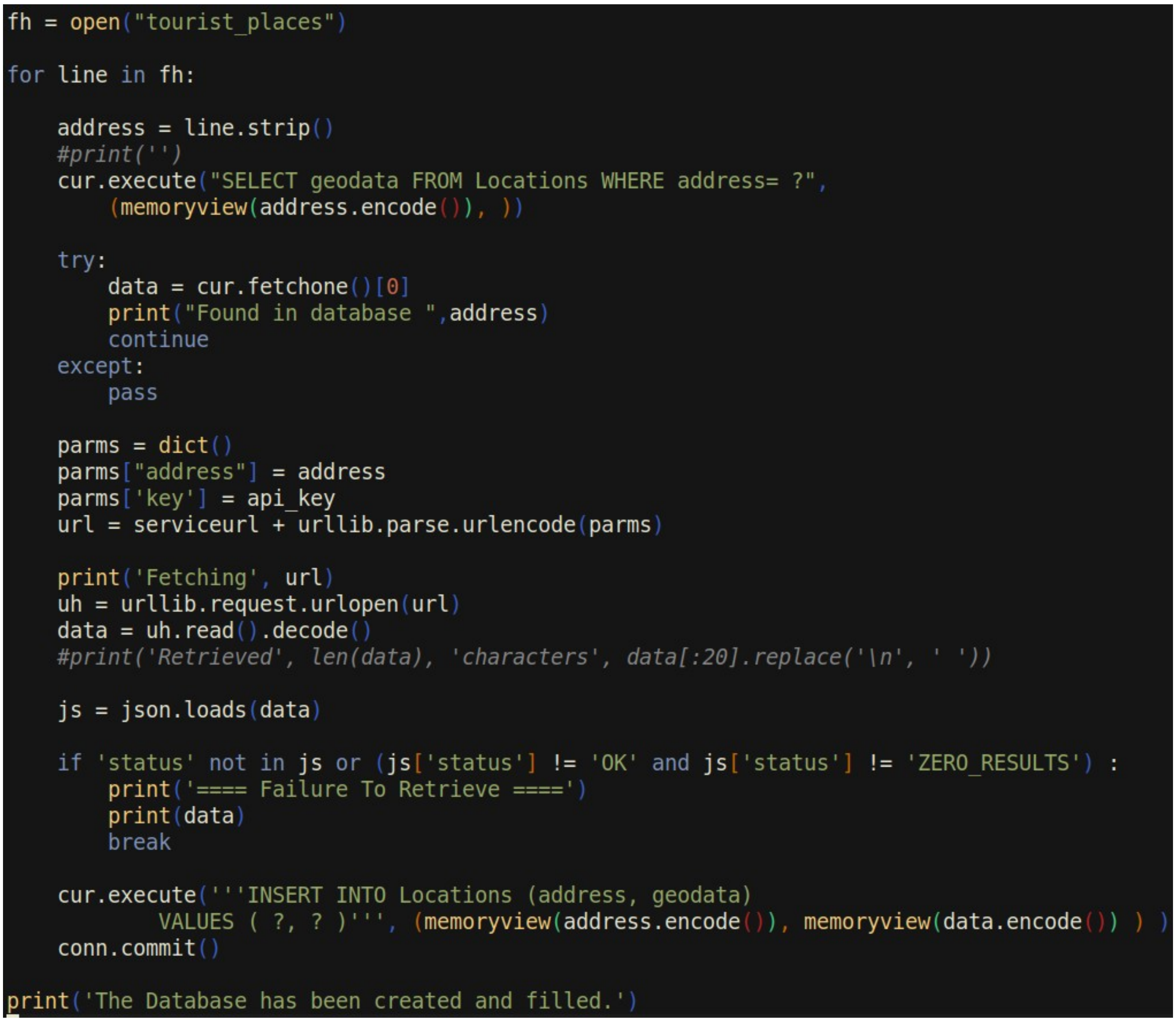

Multi-Step Data Analysis
Data Source
Gather
Clean/Process
Visualize
Analyze
(5, 1.0, 0.985, 3, u'http://www.dr..')
(3, 1.0, 2.135, 4, u'http://www.dr..')
(1, 1.0, 0.659, 2, u'http://www.dr..')
(1, 1.0, 0.659, 5, u'http://www.dr..')
....

This picture shows the flow of application.

I have tourist places saved in a file named ‘tourist_places’. I run gather.py on it, which connects to the API and retrieves the location data, and also saves all of that data into database. Now we have the databse named ‘places_database.sqlite’ running.
Next I run make_coordinates.py which reads this database and pushes the coordinate data into the Javascript file - ‘coordinates.js’
Now when index.html will be opened, all places will be marked on a Google Map.

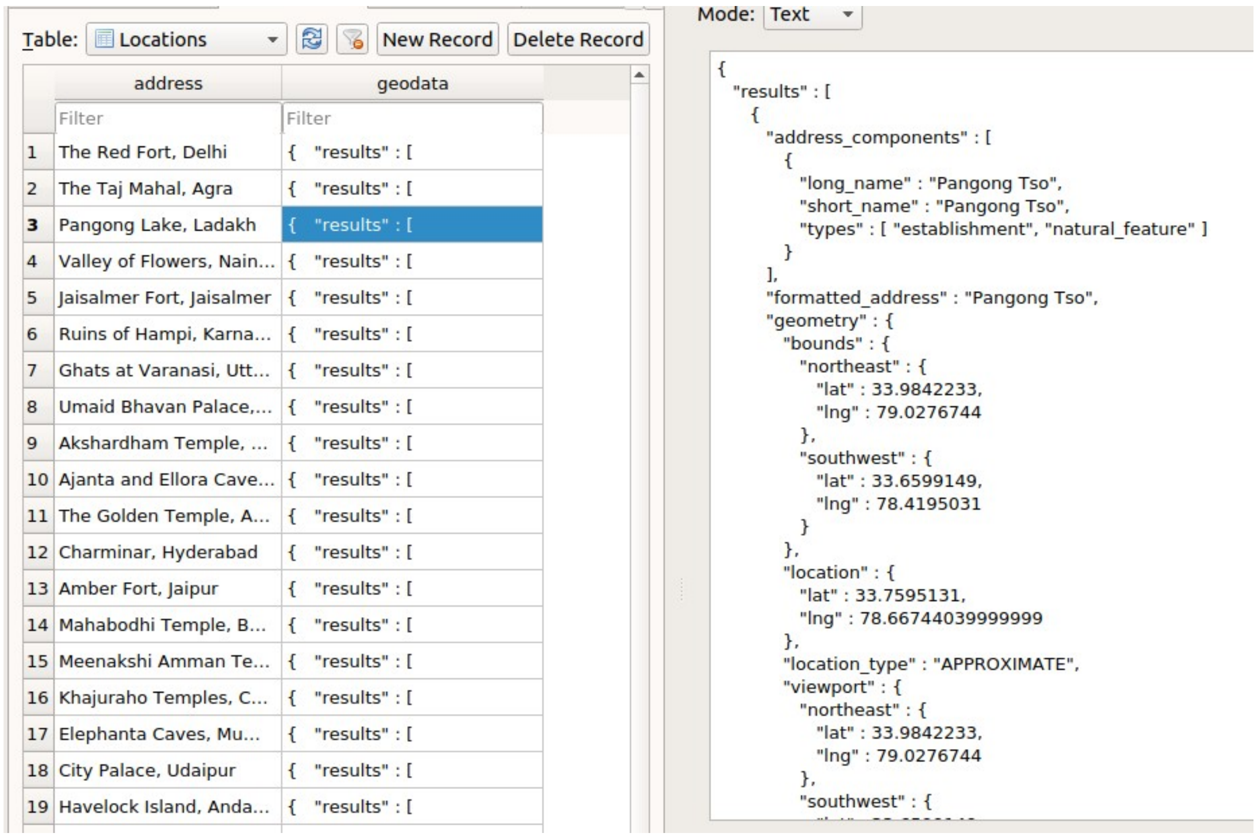
All code available at https://github.com/NandeeshG/Geolocation

## Snapshots



```
fh = open("tourist_places")

for line in fh:

    address = line.strip()
    #print('')
    cur.execute("SELECT geodata FROM Locations WHERE address= ?",
        (memoryview(address.encode()), ))

    try:
        data = cur.fetchone()[0]
        print("Found in database ",address)
        continue
    except:
        pass

    parms = dict()
    parms["address"] = address
    parms['key'] = api_key
    url = serviceurl + urllib.parse.urlencode(parms)

    print('Fetching', url)
    uh = urllib.request.urlopen(url)
    data = uh.read().decode()
    #print('Retrieved', len(data), 'characters', data[:20].replace('\n', ' '))

    js = json.loads(data)

    if 'status' not in js or (js['status'] != 'OK' and js['status'] != 'ZERO_RESULTS') :
        print('==== Failure To Retrieve ====')
        print(data)
        break

    cur.execute('''INSERT INTO Locations (address, geodata)
            VALUES ( ?, ? )''', (memoryview(address.encode()), memoryview(data.encode()) ) )
    conn.commit()

print('The Database has been created and filled.')
```

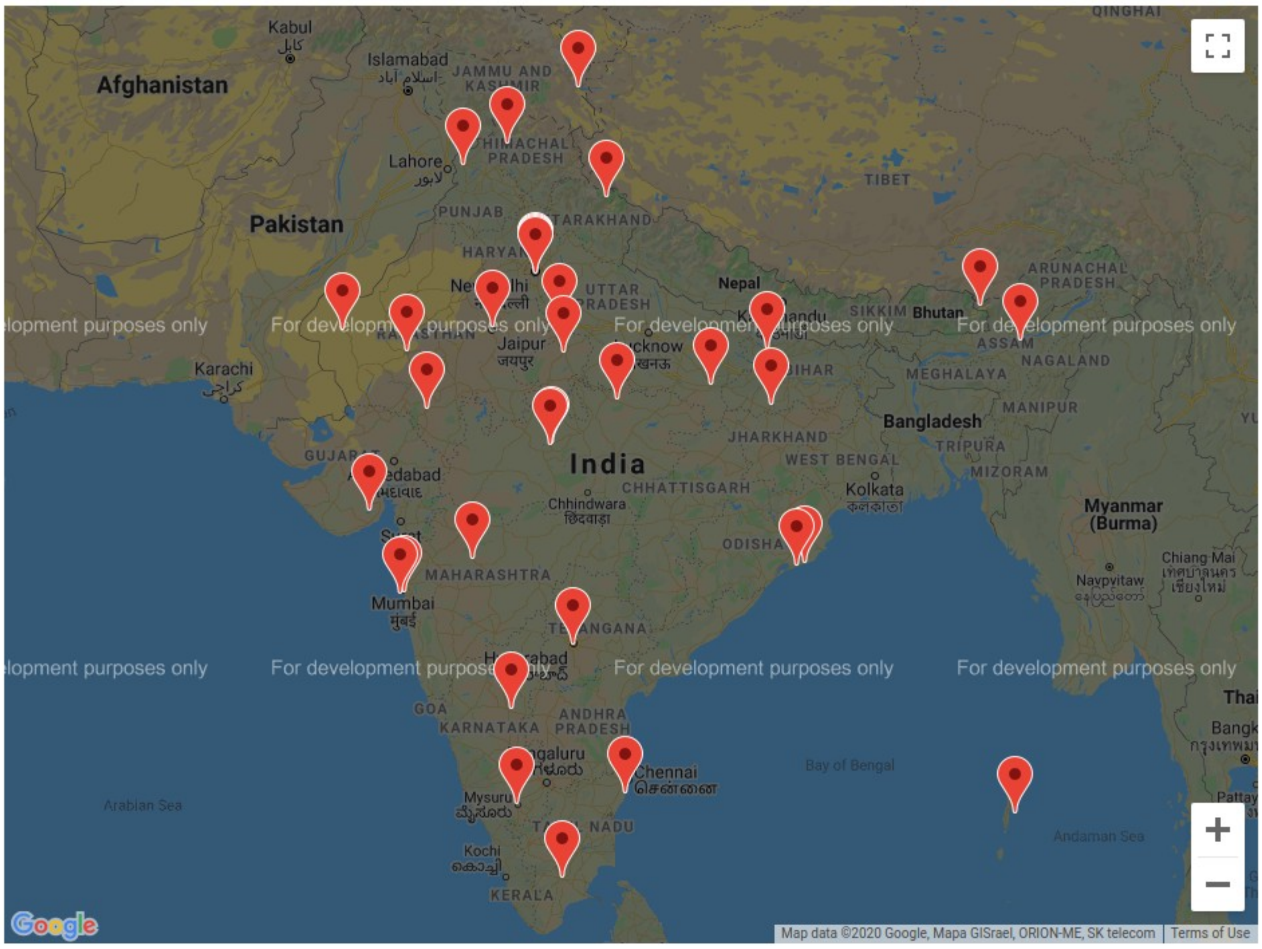
A section of code from gather.py



| | address | geodata |
|---|---|---|
| | Filter | Filter |
| 1 | The Red Fort, Delhi | { "results" : [ |
| 2 | The Taj Mahal, Agra | { "results" : [ |
| 3 | Pangong Lake, Ladakh | { "results" : [ |
| 4 | Valley of Flowers, Nain... | { "results" : [ |
| 5 | Jaisalmer Fort, Jaisalmer | { "results" : [ |
| 6 | Ruins of Hampi, Karna... | { "results" : [ |
| 7 | Ghats at Varanasi, Utt... | { "results" : [ |
| 8 | Umaid Bhavan Palace,... | { "results" : [ |
| 9 | Akshardham Temple, ... | { "results" : [ |
| 10 | Ajanta and Ellora Cave... | { "results" : [ |
| 11 | The Golden Temple, A... | { "results" : [ |
| 12 | Charminar, Hyderabad | { "results" : [ |
| 13 | Amber Fort, Jaipur | { "results" : [ |
| 14 | Mahabodhi Temple, B... | { "results" : [ |
| 15 | Meenakshi Amman Te... | { "results" : [ |
| 16 | Khajuraho Temples, C... | { "results" : [ |
| 17 | Elephanta Caves, Mu... | { "results" : [ |
| 18 | City Palace, Udaipur | { "results" : [ |
| 19 | Havelock Island, Anda... | { "results" : [ |

```
{
  "results" : [
    {
      "address_components" : [
        {
          "long_name" : "Pangong Tso",
          "short_name" : "Pangong Tso",
          "types" : [ "establishment", "natural_feature" ]
        }
      ],
      "formatted_address" : "Pangong Tso",
      "geometry" : {
        "bounds" : {
          "northeast" : {
            "lat" : 33.9842233,
            "lng" : 79.0276744
          },
          "southwest" : {
            "lat" : 33.6599149,
            "lng" : 78.4195031
          }
        },
        "location" : {
          "lat" : 33.7595131,
          "lng" : 78.66744039999999
        },
        "location_type" : "APPROXIMATE",
        "viewport" : {
          "northeast" : {
            "lat" : 33.9842233,
            "lng" : 79.0276744
          },
          "southwest" : {
```

Snapshot of a section of created Database.

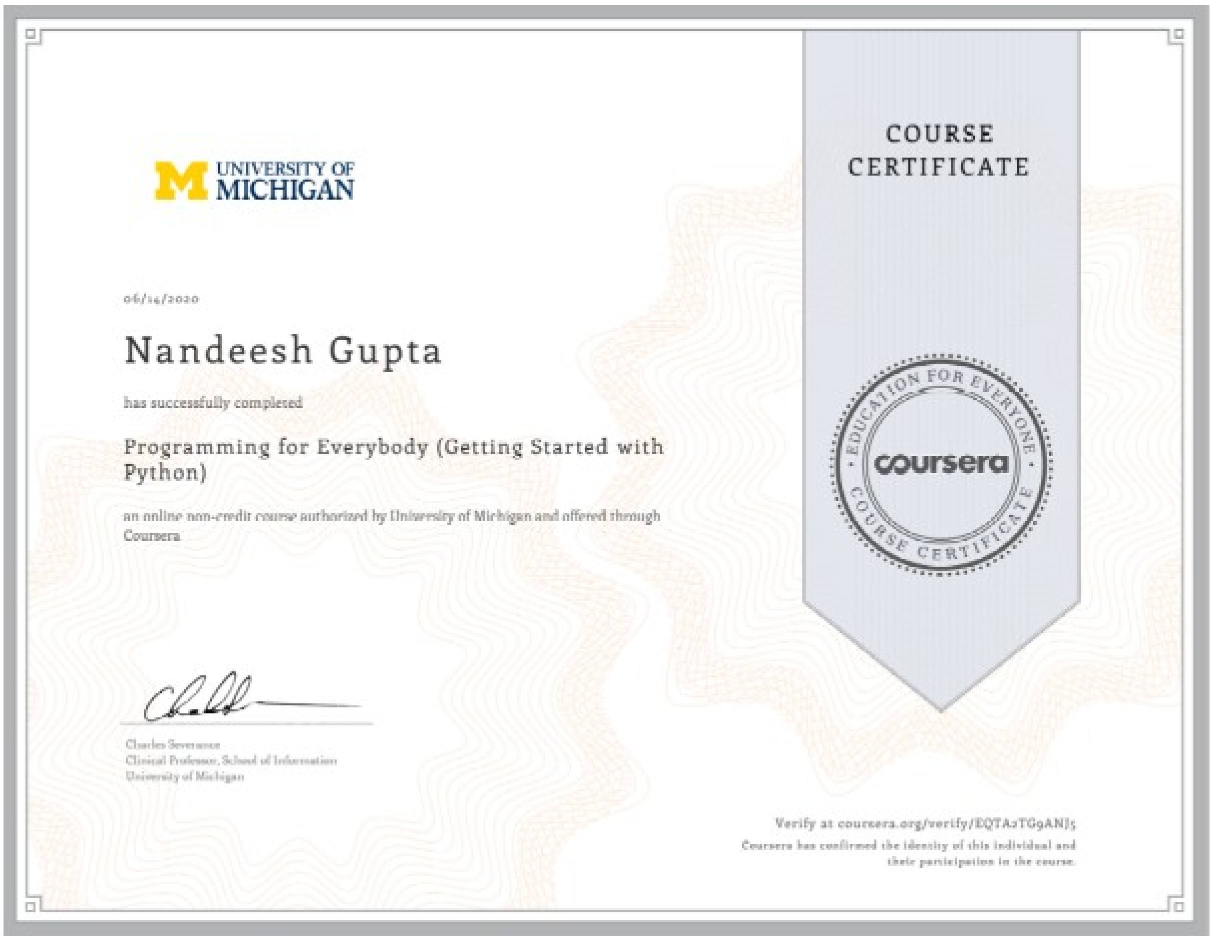


The plotted tourist places on a google map.

## RESULTS AND DISCUSSIONS

I was able to complete the given task within a deadline. With this project I was able to grasp the deep understanding of Python3, Sqlite3, JavaScript and APIS and how to make them work together. The display of the locations correctly on Map once seemed to be a difficult task but with tutorial on Google Maps API page, perseverance and support of instructor helped me to overcome this obstacle.

## CERTIFICATES



UNIVERSITY OF MICHIGAN

COURSE CERTIFICATE

06/14/2020

Nandeesh Gupta

has successfully completed

Programming for Everybody (Getting Started with Python)

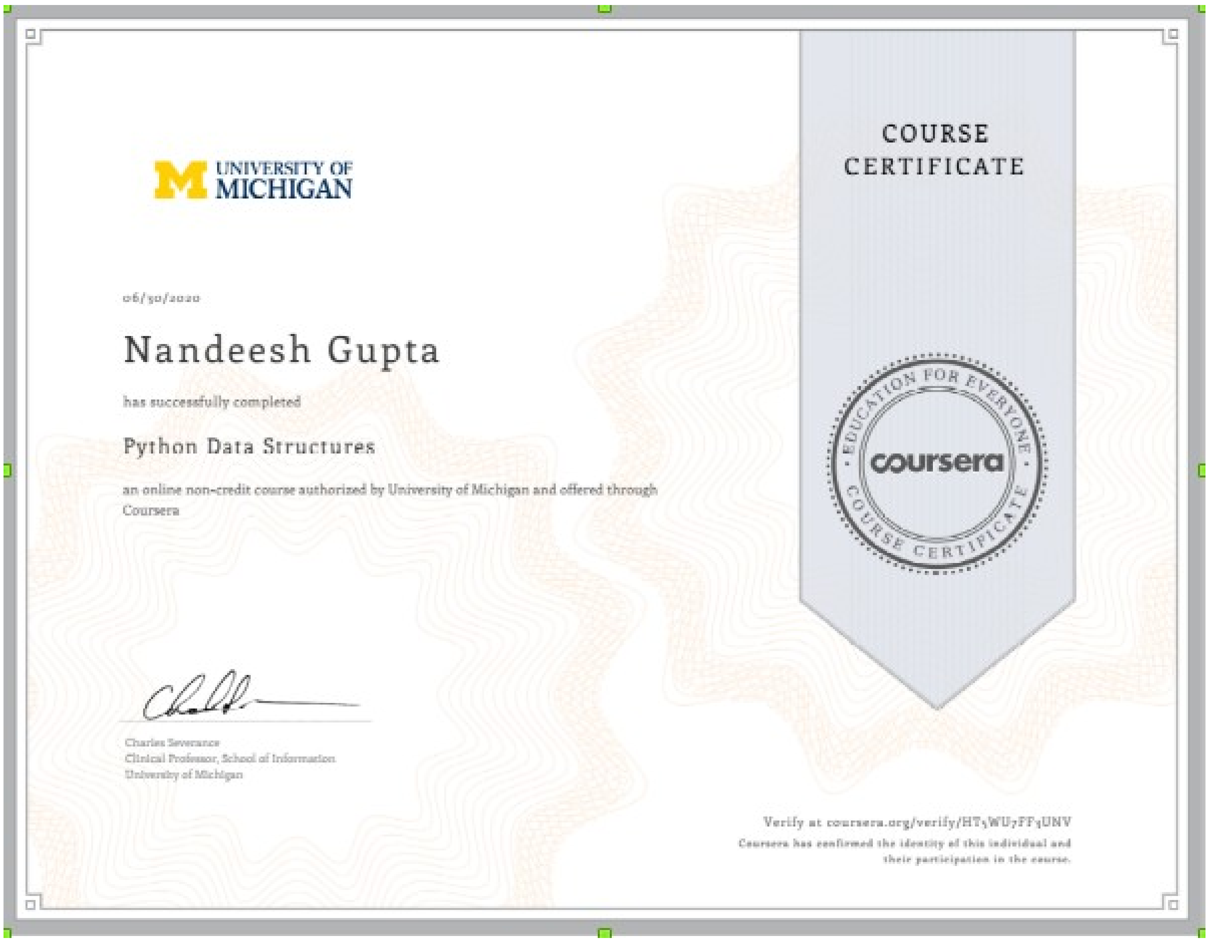
an online non-credit course authorized by University of Michigan and offered through Coursera

[illegible]

Charles Severance
Clinical Professor, School of Information
University of Michigan

EDUCATION FOR EVERYONE · coursera · COURSE CERTIFICATE

Verify at coursera.org/verify/EQTA2TG9ANJ5
Coursera has confirmed the identity of this individual and their participation in the course.



UNIVERSITY OF MICHIGAN

COURSE CERTIFICATE

06/30/2020

Nandeesh Gupta

has successfully completed

Python Data Structures

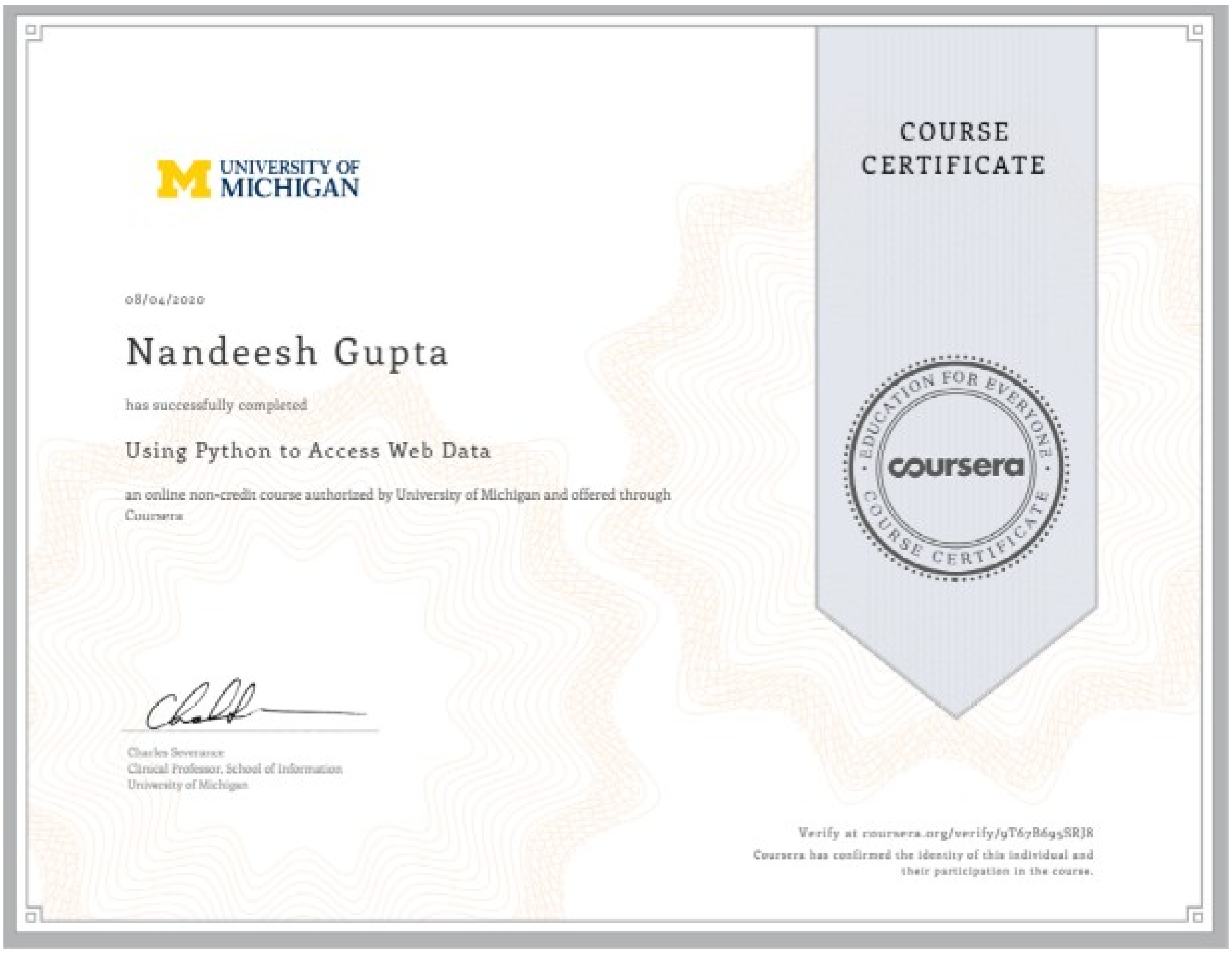
an online non-credit course authorized by University of Michigan and offered through Coursera

[illegible]

Charles Severance
Clinical Professor, School of Information
University of Michigan

EDUCATION FOR EVERYONE · coursera · COURSE CERTIFICATE

Verify at coursera.org/verify/HT5WU7FF4UNV
Coursera has confirmed the identity of this individual and their participation in the course.



UNIVERSITY OF MICHIGAN

COURSE CERTIFICATE

08/04/2020

Nandeesh Gupta

has successfully completed

Using Python to Access Web Data

an online non-credit course authorized by University of Michigan and offered through Coursera

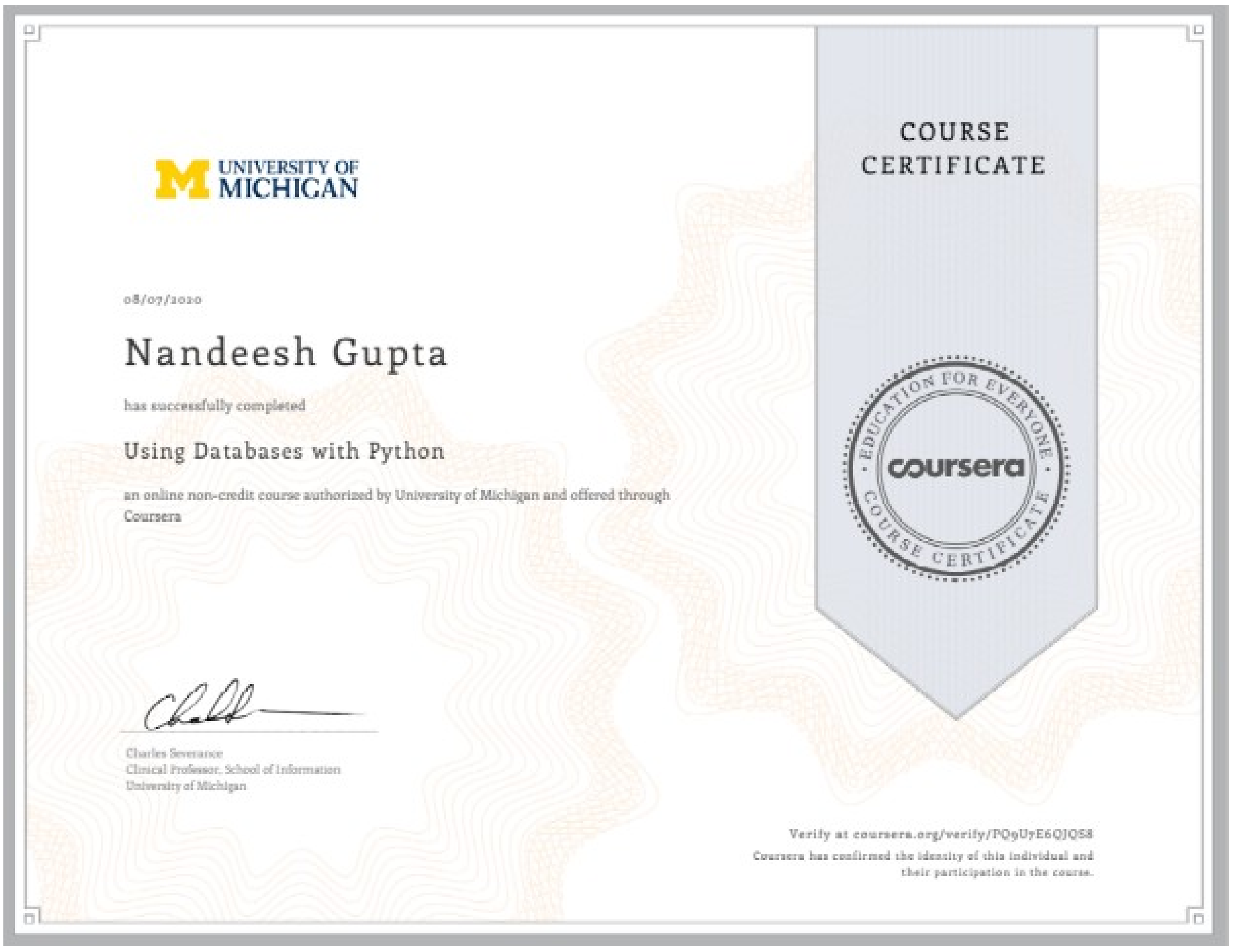
[illegible]

Charles Severance
Clinical Professor, School of Information
University of Michigan

EDUCATION FOR EVERYONE
coursera
COURSE CERTIFICATE

Verify at coursera.org/verify/9T67R695SRJR

Coursera has confirmed the identity of this individual and their participation in the course.



UNIVERSITY OF MICHIGAN

COURSE CERTIFICATE

08/07/2020

Nandeesh Gupta

has successfully completed

Using Databases with Python

an online non-credit course authorized by University of Michigan and offered through Coursera

[illegible]

Charles Severance
Clinical Professor, School of Information
University of Michigan

EDUCATION FOR EVERYONE
coursera
COURSE CERTIFICATE

Verify at coursera.org/verify/PQ9U7E6QJQS8

Coursera has confirmed the identity of this individual and their participation in the course.

# CONCLUSION

At the end it was a great learning experience for me, instructors were very helpful due to which I was able to complete my training and make cool projects along the way. These projects seemed to be a very difficult task at the start but at the end I was able to complete them in required time .There are many takeaways from my summer trainings and I would like to list some. During this period, I learned:

- To work on a open source version control software like git
- Building Static Websites using HTML CSS and JS
- To work on responsive design and UI/UX.
- To add graphical elements, graphs and animations using canvas element and vis.js library
- To import various python libraries and learn to read documentation.
- Get acquainted with DBMS, XML, JSON and APIs
- To maintain codebase on GitHub regularly.
- Creating multi file applications and working with them across various languages.

## REFERENCES

- https://www.coursera.org/specializations/python
- https://visjs.org/
- https://docs.python.org/2/library/urllib.html
- https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- https://docs.python.org/2/library/sqlite3.html
- https://docs.python.org/3/library/json.html
- https://docs.python.org/2/library/xml.etree.elementtree.html
- https://developers.google.com/maps/documentation
- https://github.com/NandeeshG/training_presentation

END

NANDEESH GUPTA

MAIT, Delhi
Roll No – 07014802718
B.Tech CSE