Part A

1. Divisible by 3 or 7

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript" src="division.js"></script>
<title>Number Division Check</title>
</head>
<body>
<h1>Division by 3 or 7 </h1>
<br><br>
Enter the number:<br><br>
<input type="text" id="number" placeholder="Enter Number">
<br><br>
<input type="button" value="Check Divisibility" onclick="check()">
<br><br>
Result:<br>
<p id="result"></p>
</body>
</html>
```

division.js

```
function check(){
num=document.getElementById("number").value;
if(isNaN(num))
{
alert("Enter a number");
}
else
{
x=num%3;
y=num%7;
if (!x&&!y)
{
     document.getElementById('result').innerHTML = num + ' is divisible by 3 and
7';
}
else if (!y)
{
      document.getElementById('result').innerHTML = num + ' is divisible by 7
and not divisible by 3';
}
else if (!x)
{
      document.getElementById('result').innerHTML = num + ' is divisible by 3
and not divisible by 7';
}
else
{
     document.getElementById('result').innerHTML = num + ' is neither divisible
by 3 and nor by 7';
}
}
}
```

2. JSON – Author details

```
<!DOCTYPE html>
<html>
<body>
<h1>Author details - JSON</h1>
<table id="authortable" border=10>
<th>Author name</th><th> Book Title</th>
<tr><td id="author1"></td><td id="title1"></td></tr>
<tr><td id="author2"></td><td id="title2"></td></tr>
</table>
<p id="author3"></p>
<p id="author4"></p>
<script type="text/javascript">
    var authors = [
        {"author":"JK Rowling","title":"Harry Potter" },
        {"author":"George Martin","title":"Game of Thrones" },
        {"author":"Leo Tolstoy", "title":"The prisoner"  },
                    {"author":"Enid Blyton","title":"Hardy boys"}
    ];

    var i; var k=0;

    for(i = 0; i < 2; i++)
    {
        var obj = authors[i];
                    k=k+1;
                    n = "author"; n +=k;
        m = "title"; m += k;
        document.getElementById(n).setAttribute("id",n);
        document.getElementById(n).innerHTML = obj.author;
                    document.getElementById(m).setAttribute("id",m);
        document.getElementById(m).innerHTML = obj.title;
    }
                    var t3=authors[2];
                    var t4=authors[3];
                    document.getElementById("author3").innerHTML="Author :
"+t3.author+" Title : "+t3.title;
                    document.getElementById("author4").innerHTML="Author :
"+t4.author+" Title : "+t4.title;
</script>
<body>
</html>
```

3. Find longest word in a sentence

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript" src="longest.js"></script>
<title>Longest Word in Textarea</title>
</head>
<body>
<textarea rows="4" cols="50" id="input1" name="comment"
form="usrform"></textarea>
<br>
<input type="button" id="btn1" value="Get Longest Word" onclick="long()">
<br/>
<p>Longest Word:<br>
<span id='sp1'></span>
</p>
</body>
<html>
```

longest.js

```
function long()
{
        var vals = document.getElementById('input1').value.split(' ');
        var maxi = vals[0];
        vals.forEach(function(v){ if(v.length>maxi.length) maxi = v;});
        document.getElementById('sp1').textContent = maxi + '   length:   ' +
maxi.length;
}
```

4. Multiply a number by 2 and itself

```
<!DOCTYPE html>
<html>
<head><script type="text/javascript" src="a4.js">
</script></head>
<body>
<form method="post">
enter a number:<input type='text' id='num'></br>
<button type="button" onclick="double()">twice</button>
<button type="button" onclick="square()">square</button>
<p result is: id="res"></p>
</form>
</body>
</html>
```

a4.js

```
function double()
{
var x;
x=document.getElementById("num").value;
if(x=="")
{
                    alert("can't be empty")
}
else if(isNaN(x)){
document.getElementById("res").innerHTML='not valid data type';
}
else{
                    document.getElementById("res").innerHTML=2*x;
}}
function square()
{
var x;
x=document.getElementById("num").value;
if(x=="")
{
                    alert("can't be empty")
}
else if(isNaN(x)){
document.getElementById("res").innerHTML='not valid data type';
}
else{
                    document.getElementById("res").innerHTML=x*x;
}
}
```

5. JSON – Author details in HTML Table tag and Plain text

```
<!DOCTYPE html>
<html>
<body>
<h1> authors and their works</h1>
<table id="authortable" border=10>
<th>AUTHORS</th><th>BOOKTITLE</th><th>native</th><th>year</th>
<tr><td id="author1"></td><td id="title1"></td><td id="area1"></td><td
id="year1"></td></tr>
<tr><td id="author2"></td><td id="title2"></td><td id="area2"></td><td
id="year2"></td></tr>
</table>
<p id="author3"></p>
<p id="author4"></p>
<script>
var authors=[
{"author":"J.K. Rowling","title":"Harry Potter","area":"USA","year":"1997 -
2007"},
{"author":"Chethan Bhagat","title":"2 States","area":"India","year":"2009"},
{"author":"Leo Tolstoy","title":"War and Peace","area":"Russia","year":"1869"},
{"author":"William Shakespeare","title":"Hamlet","area":"Denmark","year":"1599"}
];
var i,m,n,p,q,k=0;
for(i=0;i<2;i++)
{
var obj=authors[i];
k=k+1;
n="author";
n+=k;
m="title";
m+=k;
p="area";
p+=k;
q="year";
q+=k;
document.getElementById(n).setAttribute("id",n);
document.getElementById(n).innerHTML=obj.author;
document.getElementById(m).setAttribute("id",m);
document.getElementById(m).innerHTML=obj.title;
document.getElementById(p).setAttribute("id",p);
document.getElementById(p).innerHTML=obj.area;
document.getElementById(q).setAttribute("id",q);
document.getElementById(q).innerHTML=obj.year;
}
var t3=authors[2];
var t4=authors[3];
document.getElementById("author3").innerHTML="Author : "+t3.author+". Title :
"+t3.title+". Area : "+t3.area+". Year : "+t3.year;
document.getElementById("author4").innerHTML="Author : "+t4.author+". Title :
"+t4.title+". Area : "+t4.area+". Year : "+t4.year;
</script>
<body>
</html>
```

6. Arithmetic operations (Calculator)

```
<html>
<head><script type="text/javascript" src="calc.js">
</script></head>
<form>
Operand 1 <br> <input type="number" id="op1"> <br>
Operand 2 <br> <input type="number" id="op2"> <br>
<input type="radio" name="calc" id="add" value="Add" onclick="result()"> Add
<br>
<input type="radio" name="calc" id="sub" value="Sub" onclick="result()"> Sub
<br>
<input type="radio" name="calc" id="mul" value="Multiply" onclick="result()">
Multiply <br>
<input type="radio" name="calc" id="div" value="Divide" onclick="result()">
Divide <br>
</form>
<p id="Res"></p>
</html>
```

calc.js

```
function result(){
var x = document.getElementById("op1").value;
var y = document.getElementById("op2").value;
if ( (x== "") || (y == "") )
{
document.getElementById("Res").innerHTML="Enter values in both fields";
}
else
{
if(document.getElementById("add").checked)
{
                    a = parseInt(x);
                    b = parseInt(y);
document.getElementById("Res").innerHTML=a+b;
}
else if(document.getElementById("sub").checked)
{
document.getElementById("Res").innerHTML=x-y;
}
else if(document.getElementById("mul").checked)
{
document.getElementById("Res").innerHTML=x*y;
}
else if(document.getElementById("div").checked)
{
if(y!=0)
document.getElementById("Res").innerHTML=x/y;
else document.getElementById("Res").innerHTML="Divide by zero";
}
}
}
```

7. Atomic Dictionary

```
def AtomicDictionary():
    d={"fe":"iron","p":"phosphorus","n":"nitrogen"}
    print(d)

    symbol=input("enter the symbol\n")
    element=input("enter the element\n")

    if symbol in d.keys():
            print("key exists and hence value is replaced")
    else:
            print("new key and value added to dictionary")
    d[symbol]=element
    print(d)

    print("length of dictionary: ",str(len(d)))

    key=input("enter the symbol to search\n")
    if key in d.keys():
        print(d[key])
    else:
        print("key does not exists in dictionary")

AtomicDictionary()
```

8. Use list, lambda and reduce functions

```
from functools import reduce

l1=[2,4,5,6,8,10]
l2=[x*3 for x in l1]
print("original list :",l1)
print("new list :",l2)
sum = reduce(lambda a,b : a+b,l1)
print ("The sum of the original list elements is :",sum)
sum = reduce(lambda a,b : a+b,l2)
print ("The sum of the new list elements is :",sum)
```

9. Student class

```
class Student:
                    def __init__(self,name,age,marks):
                     self.name=name
                     self.age=age
                     self.marks=marks

                    def  accept():
                     l=[]
                     n=input("Enter name of student : ")
                     a=input("Enter age : ")
                     for i in range(3):
                         x=int(input("Enter marks of sub " + str(i+1) +" : "))
                         l.append(x)
                     s=Student(n,a,l)
                     return s

                    def display(self):
                     print("Name : " + self.name)
                     print("Age : " + str(self.age))
                     print("List of marks : " + str(self.marks))
s1=Student('Pieterson',34,[89,90,91])
s1.display()
s2=Student.accept()
s2.display()
```

10. Student Dictionary

```
s={34:"walter",67:"jesse",78:"gustavo",32:"marie",89:"frank",12:"skyler"}

for i in s:
                    if i%2==0:
          print (str(i)+" : "+s[i])
```

11. String functions

```
def reverse(string):
    string = "".join(reversed(string))
    return string
l=["No 1","I am 2", "not color blind 3","my world 4 ","is black and white
5","try to keep 6"]

print("Even Strings")
for i in range(len(l)):
                    if i%2!=0:
                     print(l[i])

print("\nUpper Case")
for i in range(len(l)):
                    if (i+1)%3==0:
                     print(l[i].upper())

print("\nReverse Strings")
l2=[reverse(i) for i in l]
print(l2)

print("\nDigits")
l3=[item for subitem in l for item in subitem.split() if item.isdigit()]
print(l3)
```

Part B

1. Temperature conversion

```
f=0
kel=0

def c2f(temp):
    global f
    f=(temp*9/5)+32
    print(f)

def c2k(temp):
    global kel
    kel=temp+273
    print(kel)

def tempmenu():
    print("1.Enter new temperature (Celsius)")
    print("2.Celsius to Farenheit")
    print("3.Celsius to Kelvin")
    print("4.Display")
    print("5.Exit")

    while 1:
        print("enter choice")
        ch=int(input())
        if(ch==1):
            print("Enter temperature (Celsius)")
            temp=int(input())
        elif(ch==2):
            c2f(temp)
        elif(ch==3):
            c2k(temp)
        elif(ch==4):
            a=["the temp in C:",temp,"to F is",f,"the temp in K is",kel]
            print(a)
        elif(ch==5):
            print("Program terminated successfully")
            exit()
        else:
            print("Invalid Option")

tempmenu()
```

2. Sentence Reverse

```
class SentenceReverser:
	vowels = ["a","e","i","o","u"]
	sentence = ""
	reverse = ""
	vowelCount = 0
	def __init__(self,sentence):
		self.sentence = sentence
		self.reverseSentence()
	def reverseSentence(self):
		self.reverse = " ".join(reversed(self.sentence.split()))
	def getVowelCount(self):
		self.vowelCount = sum(s in self.vowels for s in
self.sentence.lower())
		return self.vowelCount
	def getReverse(self):
		return self.reverse

items = []

n = int(input("Enter number of sentences required : "))
for i in range(n):
	sentence = input("Enter a phrase : ")
	reverser = SentenceReverser(sentence.strip())
	items.append(reverser)
	print(reverser.reverse)

sortedItems = sorted(items, key=lambda item: item.getVowelCount(), reverse=True)

print ("Sorted on vowel count (descending) : \n")
for i in range(len(sortedItems)):
		print ("Reverse : ", sortedItems[i].getReverse(), ", Vowel
Count : ", sortedItems[i].getVowelCount())
```

3. Python for Data Science (titanic dataset)

```
import pandas as pd
from pandas  import Series,DataFrame
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

titanic_df=pd.read_csv("titanic.csv")

print("-----------DATA HEADER---------------")
print(titanic_df.head())
print("----------DATA DESCRIPTION-----------")
titanic_df.info()
titanic_df.describe()
titanic_df=titanic_df.drop(["PassengerId","Name","Ticket"],axis=1)
print("------check if columns are dropped--------")
print(titanic_df.head())
titanic_df["Age"]=titanic_df["Age"].fillna(20)
print(titanic_df["Age"])
print("No of rows in the data set = ",len(titanic_df))
print("No of columns in the data set = ",len(titanic_df.columns))
print("the minimum age = ",titanic_df.Age.min())
print("the maximum age = ",titanic_df.Age.max())
print("the mean age = ",titanic_df.Age.mean())
print("the SD of age = ",titanic_df.Age.std())
titanic_df.Age.hist()
plt.show()
ax1=titanic_df.Age.hist()
ax1.set(xlabel="Age",ylabel="Number of people")
plt.show()
ax2=titanic_df.Age.plot()
ax2.set(xlabel="Age",ylabel="Number of people")
plt.show()
```

4. Python File Handling and List Comprehension

```
import sys
import re
import operator
from functools import reduce

# sys.argv[0] contains the file name
if 1:
    count_dict = {}
    count_list = []
    with open("abc.txt", 'r') as f:
        for line in f:
            for word in line.split():
                word = re.sub(r'[^\w\s]','',word)
                if word not in count_dict:
                    count_dict[word] = 1
                else:
                    count_dict[word] += 1
    count_list = sorted(count_dict.items(), key=operator.itemgetter(1),
reverse=True)
    print (count_list[:10])
    count = []
    print (count_list[0][1])
    for i in range(len(count_list)):
        count.append(count_list[i][1])
    print (reduce(lambda a, b: a + b, count) / len(count))
    print ([x*x for x in count if x%2!=0])
```

5. StudentPerformance dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("studentperformance.csv")

print("<-----Data PreProcessing----->")
print("\nHead and Description of Dataset")
print(df.head(5))
print(df.describe())
print(df.info())
print("\nRemoving unwanted columns")
df1 = df.drop(['lunch','test preparation course'], axis=1)
print(df1.head(5))
print("\nFilling empty values")
df['parental level of education'] = df['parental level of
education'].fillna("bachelor's degree")
print(df.head(5))

print("\nMapping values/attributes in race/ethnicity to types")
df['race/ethnicity'] = df['race/ethnicity'].map({'group A':'Asian
Students','group B':'African Students','group C':'Afro-Asian Students','group
D':'American Students','group E':'European Students'})
print(df.head(5))

print("\n")
print("<-------Data Visualisation------->")
print(pd.crosstab(df.gender,df['test preparation course']))

ax=sns.countplot(data=df,x="test preparation
course",hue="gender",palette="Set1")
ax.set(title="tally",xlabel="courses",ylabel="total")
plt.show()

ax = sns.countplot(data=df,x='gender',hue='race/ethnicity',palette='Set1')
ax.set(title='Male and Female belonging to each
group',xlabel='Gender',ylabel='Count')
plt.show()

interval = (0,40,60,75,100)
category = ['Failed','Second Class','First Class','Distinction']
df['grade_math']  = pd.cut(df['math score'],interval,labels=category)
ax = sns.countplot(data=df,x='grade_math',hue=df['grade_math'],palette='Set1')
ax.set(title='Math Grades',xlabel='Grade Category',ylabel='Count')
plt.show()

df['grade_reading']  = pd.cut(df['reading score'],interval,labels=category)
ax =
sns.countplot(data=df,x='grade_reading',hue=df.grade_reading,palette='Set2')
ax.set(title='Reading Grades',xlabel='Grade Category',ylabel='Count')
plt.show()

df['grade_writing']  = pd.cut(df['writing score'],interval,labels=category)
ax =
sns.countplot(data=df,x='grade_writing',hue=df['grade_writing'],palette='Set3')
ax.set(title='Writing Grades',xlabel='Grade Category',ylabel='Count')
plt.show()
```

6. BlackFriday dataset

```
import pandas as pd
from pandas import Series,DataFrame
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

df=pd.read_csv("blackfriday.csv");

print("------header------");
print("df.head()");
print("------description-------");
df.info();
df.describe();
df=df.drop(["User_ID","Product_ID","Stay_In_Current_City_Years"],axis=1);
print(df.head());
df["City_Category"]=df["City_Category"].fillna("B");
print(df["City_Category"]);
df["City_Category"]=df["City_Category"].map({
                      "A":"Metro Cities",
                      "B":"Small Towns",
                      "C":"Villages"});
print(df.head());
df=df.rename(columns={"Product_Category_1":"Baseball
Caps","Product_Category_2":"Wine Tumblers","Product_Category_3":"Pet
Raincoats"});
print(df.head());

df["Marital_Status"]=df["Marital_Status"].map({
                        1:"Married",
                        0:"Un-Married"});
print(df.head());
df["Baseball Caps"]=df["Baseball Caps"].fillna(8);
df["City_Category"]=df["Wine Tumblers"].fillna(10);

ax=sns.countplot(x="Baseball Caps",hue="Gender",palette="Set1",data=df);
ax.set(title="tally",xlabel="product_1",ylabel="total");
plt.show();

ax=sns.countplot(x="Wine Tumblers",hue="Gender",palette="Set1",data=df);
ax.set(title="tally",xlabel="product_2",ylabel="total");
plt.show();
ax=sns.countplot(x="City_Category",hue="Gender",palette="Set1",data=df);
ax.set(title="tally",xlabel="cities",ylabel="total");
plt.show();
```

7. JSON Mouse-over (Cars)

```
<!DOCTYPE html>
<html>
<head>
    <title>JSON</title>
    <script src="details.js"></script>
    <style>
        h1 {
            text-align: center;
        }
        table {
            text-align: center;
            width: 100%;
        }
        td {
            font-size: 20px;
        }
        th {
            font-size: x-large;
            background-color: green;
            color: white;
        }
        th:hover {
            background-color: lightblue;
                        color:black;
        }
    </style>
</head>
<body>
    <h1>Hover over a name</h1>
    <table>
        <tr id="menu"></tr>
    </table>

    <table>
        <tr><td><b>Name</b></td><td id="name" height="50px"></td></tr>
        <tr><td><b>Model</b></td><td id="model" height="50px"></td></tr>
        <tr><td><b>Price</b></td><td id="price" height="50px"></td></tr>
        <tr><td><b>Year</b></td><td id="year" height="50px"></td></tr>
    </table>

    <p id="i"></p>
</body>
</html>
```

details.js

```
window.onload = () => {
  const details = [
    {
      model: 'Baleno',
      name: 'Suzuki',
      price: '7 lakhs',
      year: '2016',
    },
    {
      model: 'i20',
      name: 'Hyundai',
      price: '9 lakhs',
      year: '2015',
    },
    {
      model: 'Accord',
      name: 'Honda',
      price: '25 lakhs',
      year: '2013',
    },
    {
      model: 'C220d',
      name: 'Mercedes-Benz',
      price: '45 lakhs',
      year: '2011',
    },
  ];

  details.forEach((item, index) => {
    const listElement = document.createElement('th');
    listElement.onmouseover = () => {
      document.getElementById('name').innerHTML = details[index].name;
      document.getElementById('model').innerHTML = details[index].model;
      document.getElementById('price').innerHTML = details[index].price;
      document.getElementById('year').innerHTML = details[index].year;
    };
    listElement.innerHTML = item.name;
    document.getElementById('menu').appendChild(listElement);
  });
}
```

8. Python and JS – ATM Application

```
<!DOCTYPE html>
<html>
<head>
<title>ATM</title>
</head>
<body>
<h1> ATM </h1>
<h3> Your balance is : {{balance}} </h3> <!-- Account Balance displayed here -->
<h5>{{msg}}</h5> <!-- Error Message displayed here -->

<!-- Form for amount details, with buttons to Deposit or Withdraw -->
<form action="{{url_for('index')}}" method="POST" id="atm">
Amount : <input type="number" name = "amount" required /><br>
<input type="submit" name="action" value="Withdraw" onclick="return
validateForm(this)">
<input type="submit" name="action" value="Deposit" onclick="return
validateForm(this)">
</form>

<!-- Javascript to validate amount -->
<script>
function validateForm(button)
{
	var form = document.getElementById("atm");

	// Checks if amount field is empty
	if (form.amount.value == "" ) {
		alert("Amount required");
		return false;
	}
	// Checks if amount entered is negative
	if (parseInt(form.amount.value) < 0) {
		alert("Cannot enter negative amount");
		return false;
	}

	// Checks if user clicked on Withdraw and amount is greater than 5000
	if(button.value == "Withdraw" && parseInt(form.amount.value) > 5000)
	{
		alert("Cannot withdraw more than 5000");
		return false;
	}

	// Returns true if amount is valid
	return true;
	}
	</script>
</body>
</html>
```

application.py

```
from flask import Flask, redirect, render_template, request, url_for, session
import time
import re
app = Flask(__name__)
app.secret_key = "secret"

@app.route("/", methods=['GET', 'POST'])
def index():
	try:
		balance = session["balance"]
	except KeyError:
		balance = session["balance"] = 8000

	if request.method == "GET":
		return render_template("index.html", balance=balance, msg="")

	if request.method == "POST":

	# Checks if amount field is empty
	if request.form["amount"] == "" :
		msg = "Amount is required"
		return render_template("index.html", balance=balance, msg=msg)

	# Checks if amount entered is negative
	if int(request.form["amount"]) < 0 :
		msg = "Cannot enter negative amount"
		return render_template("index.html", balance=balance, msg=msg)

	# Checks if user clicked on Withdraw
	if request.form["action"] == 'Withdraw':

	# Checks if amount is greater than balance
	if int(request.form["amount"]) > session["balance"] :
		msg = "Cannot withdraw amount greater than balance"
		return render_template("index.html", balance=balance, msg=msg)

	# Checks if amount is greater than 5000
	elif int(request.form["amount"]) > 5000 :
		msg = "Cannot withdraw amount greater than 5000"
		return render_template("index.html", balance=balance, msg=msg)

	# Deducts amount entered from balance and stores in session
	else:
		balance = balance - int(request.form["amount"])
		session["balance"] = balance
		msg = "Money Withdrawn"
		return render_template("index.html", balance=balance, msg=msg)

	# Checks if user clicked on Deposit
	elif request.form["action"] == 'Deposit':

	# Adds amount entered to balance and stores in session
	balance = balance + int(request.form["amount"])
		session["balance"] = balance
		msg = "Money Deposited"
		return render_template("index.html", balance=balance, msg=msg)

if __name__ == '__main__':
		app.run()
```

9. Python and JS – Shopping Cart

store.html

```
<!DOCTYPE html>

<html>
<head>
<title>Add to Cart</title>
</head>
<body>
<h1> Store Items </h1>
<!-- Form containing Items in Store -->
<!-- Add items to cart by entering quantity for item (no negative numbers) -->
<form action = "{{url_for('store')}}" method="POST">
<input name = "eggs" type = "number" min="0" value="0"/>Eggs - Rs.5 <br>
<input name = "milk" type = "number" min="0" value="0"/>Milk - Rs.12 <br>
<input name = "bread" type = "number" min="0" value="0"/>Bread - Rs.22 <br><br>

<input value = "Add to Cart" type = "submit"/>
</form><br>

<!-- Link to view cart -->
<a href="{{url_for('cart')}}"> View your Shopping Cart </a>

</body>
</html>
```

cart.html

```
<!DOCTYPE html>

<html>
	<head>
	<title>Shopping Cart</title>
	</head>

	<body>
	<h1> Items in Cart </h1>

	<!-- View Shopping Cart - item and quantity -->
	{% for item in cart %}
	{{item["name"]}} : {{item["quantity"]}}<br>
	{% endfor %}

	<br>

	<!-- Link to view store -->
	<a href="{{url_for('store')}}"> Continue Shopping </a><br><br>

	<!-- Link to view bill -->
	<a href="{{url_for('buy')}}"> View Bill </a>

	</body>
</html>
```

bill.html

```
<!DOCTYPE html>

<html>
	<head>
	<title>Shopping Cart</title>
	</head>

	<body>
	<h1> Bill </h1>

	<!-- Table displaying item, quantity and price -->
	<table border="1" width="50%">
	<tr>
		<td>Item</td>
		<td>Quantity</td>
		<td>Price</td>
	</tr>

	{% for item in cart %}
	<tr>
		<td>{{item["name"]}}</td>
		<td>{{item["quantity"]}}</td>
		<td>{{item["price"]}}</td>
	</tr>
	{% endfor %}

	</table>

	<!-- Total Bill Amount -->
	<p>Total Bill Amount = Rs. {{amount}} </p>

	<!-- Link to view store -->
	<a href="{{url_for('store')}}"> Continue Shopping </a><br>

	</body>
</html>
```

application.py

```
from flask import Flask, redirect, render_template, request, session, url_for

app = Flask(__name__)

# Secret key for sessions
app.secret_key = "secret"

@app.route("/", methods=["GET", "POST"])
def store():
    if request.method == "GET":
        return render_template("store.html")

    # For each item in store, checks if item is in session
    # If item is in session, increments the count by value entered in the form
    # If item is not in session, initialises the count by value entered in the
form
    # Redirects to the HTML page to view Shopping Cart
    if request.method == "POST":
        for item in ["eggs", "milk", "bread"]:
            if item not in session :
                session[item] = int(request.form[item])
            else:
                session[item] += int(request.form[item])
        return redirect(url_for("cart"))


@app.route("/cart", methods=["GET", "POST"])
def cart():

    # Creates a list of dictionaries containing each item in cart and its
quantity
    # Displays this list in a HTML page
    cart = []
    for item in ["eggs", "milk", "bread"]:
        cart.append({"name":item.capitalize(), "quantity":session[item]})
    return render_template("cart.html", cart=cart)


@app.route("/buy", methods=["GET", "POST"])
def buy():

    # Total amount initialised to 0
    amount = 0

    # Index for list containing prices of items
    index = 0

    # Prices of every item
    prices = [5, 12, 22]

    # Creates a list of dictionaries containing each item in cart, its quantity,
and cost(price*qty)
    # Calculates total bill amount
    # Displays the bill in a HTML page
    cart = []
    for item in ["eggs", "milk", "bread"]:
        row = {}
        row["name"] = item.capitalize()
```

```
        row["quantity"] = session[item]
        row["price"] = prices[index] * session[item]
        amount = amount + row["price"]
        cart.append(row)
        index = index + 1
    return render_template("bill.html", cart=cart, amount=amount)

if __name__ == '__main__':
                    app.run(port=8000)
```

10. Python and JS – Student Registration

index.html

```
<!DOCTYPE html>

<html>
	<head>
	<title>Form Details</title>
	</head>

	<body>

	<h1> Register here </h1>

	<h5> {{msg}} </h5> <!-- Error Message displayed here -->

	<!-- Form for details -->
	<form action="{{url_for('trykare')}}" method="POST" >
	USN (Eg. 1MS02IS001): <input name = "usn" type="text" /><br>
	D.O.B (Eg. dd/mm/yyyy) : <input name = "dob" type="text" /><br>
	Enter Marks for 3 Subjects<br><br>
	<input name="m1" type="number" min=0 max=100 />
	<input name="m2" type="number" min=0 max=100 />
	<input name="m3" type="number" min=0 max=100 /><br>
	<input name = "Register" type = "submit" />
	</form>

	</body>
</html>
```

success.html

```
<!DOCTYPE html>

<html>
	<head>
	<title>Form Details</title>
	</head>

	<body>
	<h1> Registration Successful</h1>
	<h3> Average is {{avg}} </h3>
	</body>
</html>
```

application.py

```
from flask import Flask, redirect, render_template, request, url_for
import time
import re

app = Flask(__name__)

@app.route("/", methods=['GET', 'POST'])
def trykare():
	if request.method == "GET":
		return render_template("indexJS.html")

	if request.method == "POST":

	#Check if form fields are empty
	if request.form["usn"] == "" or request.form["dob"] == "" or
request.form["m1"] == "" or request.form["m2"] == "" or request.form["m3"] == ""
:
		msg = "All form fields are required"
		return render_template("indexJS.html", msg=msg)

	#Check if date entered in dd/mm/yyyy format and is not an invalid date
Eg. 31/11/2016
	#Use strptime() function which raises an exception if date is invalid
	try:
		time.strptime(request.form["dob"],"%d/%m/%Y")
	except ValueError:
		msg = "Date is invalid"
		return render_template("indexJS.html", msg=msg)

	#Regex for USN
	usn_pattern = "^[1][A-Z][A-Z][0-9][0-9][A-Z][A-Z][0-9][0-9][0-9]$"

	#Check if entered USN matches Regex
	if not re.match(usn_pattern, request.form["usn"]) :
		msg = "USN format invalid"
		return render_template("indexJS.html", msg=msg)

	#If form fields are valid return success HTML page
	avg=(int(request.form["m1"])+int(request.form["m2"])
+int(request.form["m3"]))/3
		return render_template("success.html",avg=avg)

if __name__ == '__main__':
		app.run()
```

11. JSON and JS – Patient details

```
<!DOCTYPE html>

<html>

<body>

Hospital name:<p id="h_name" ></p><br><br>

Hospital location:<p id="h_loc" ></p>

<h2 align = "center" id="HoverText" onmouseover="myHoverFunction()"> Patient
Details </h2>

<br>

<table id="table1" border=1 hidden>

<th> Name</th><th>Aadhaar</th><th>Lab_Tests</th>

       <tr><td id="name1"></td><td id = "id1"></td><td id = "test1"></tr>

       <tr><td id="name2"></td><td id = "id2"></td><td id = "test2"></tr>

       <tr><td id="name3"></td><td id = "id3"></td><td id = "test3"></tr>

</table>

<script>

var obj={

       "name":"Fortis hospital",

       "location":"Bangalore"};

var x,y;

x=obj.name;

y=obj.location;

document.getElementById("h_name").innerHTML=x;

document.getElementById("h_loc").innerHTML=y;
```

```
function myHoverFunction()
{
  document.getElementById("HoverText").style.color = "red";
  document.getElementById("table1").removeAttribute('hidden');
    var p = [
        {"name":"Patient 1", "id": "1226", "tests":["CT", "MRI"]},
        {"name":"Patient 2", "id": "1228", "tests":["PET", "CT", "X-Ray"]},
        {"name":"Patient 3", "id": "1248", "tests":["Blood", "MRI"] }
    ];

    var i1, j1, x1=" ",x2=" "; z1=" ";
    var n = "name"; var m="test"; var im = "id"; var k=0;


    for (i1 in p) {
        x1 = " "; x2 = " "; z1 = " ";
        k = k+1;
        n = "name"; n +=k;
        m = "test"; m += k;
        im = "id"; im += k;

        x1 += p[i1].name;
        document.getElementById(n).setAttribute("id",n);
        document.getElementById(n).innerHTML = x1;

        for (j1 in p[i1].tests) {

            x2 +=p[i1].tests[j1]+ "   ";

            document.getElementById(m).setAttribute("id",m);
            document.getElementById(m).innerHTML = x2;
        }
        z1 += p[i1].id;
        document.getElementById(im).setAttribute("id",im);
        document.getElementById(im).innerHTML = z1;
    }
}

</script>
</body>
</html>
```