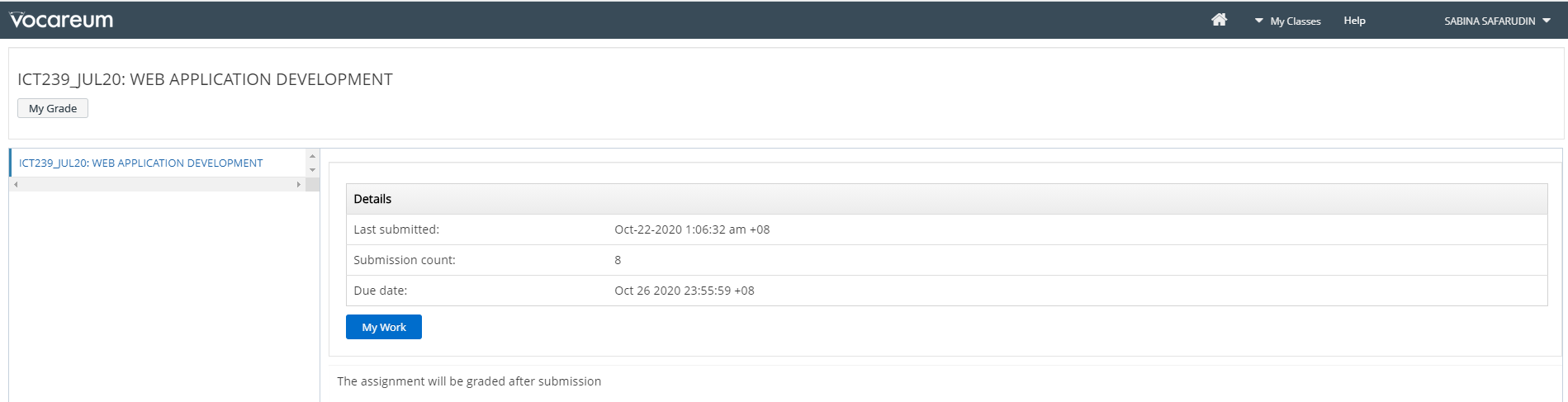
ICT 239 Web Programming TMA 01

Sabina Safarudin

H1710798

Submission date: 22 Oct 2020

Browser used for development and testing: Firefox



**Question 1a)**

home.html

<!DOCTYPE html>

<html>

    <head>

        <script src="https://www.chartjs.org/dist/2.9.3/Chart.min.js"></script>

        <meta charset="UTF-8">

        <link rel="stylesheet" href="css/ramen.css">

    </head>

    <body>

        <div class="header-box">

            <div class="headerlogo">

                <img src="img/DCSlogo.png">

                <div class="header-text">DCS Smart Trolley</div>

                <div class="logoutheader"><a href="#">Logout</a></div>

            </div>

        </div>

        <div class="wrap">

            <div class="sidebar">

                <div class="profile-pic">

                    <img src="img/blankprofile.png">

                    <p>sabina001@suss.edu.sg</p>

                </div>

                <ul><a href="#">Dashboard View</a></ul>

                <ul><a href="#">Account</a></ul>

                <ul><a href="#">Configure Email</a></ul>

                <ul><a href="#">Download</a></ul>

                <ul><a href="#">Logout</a></ul>

            </div>

            <div class="contentbox">

                <div class="active-trolley">

                    <div class="trolley-title">

                        Active Trolleys

                    </div>

                    <div class="number-trolley">

                        0

                    </div>

                    <div class="trolley-footer">

                        Total: 0

                    </div>

                </div>

                <div class="graphbox">

                    <h4>Temperature Recording</h4>

                    <p style="font-size: 12px;">Today's Recording</p>

                    <canvas id="trolleychart"></canvas>

                </div>

            </div>

        </div>

    </body>

</html>

<script>

    var trolleychart = document.getElementById("trolleychart").getContext('2d');

    var tempChart = new Chart(trolleychart, {

        type:'line',

        data: {

            labels: ['Tue 11', '3am', '6am', '9am', '12pm', '3pm', '6pm', '9pm','Wed 12'],

            datasets: [{

                label: 'trolley 1',

                fill: false,

                data: [33,34,21,56],

                borderColor: 'black',

                backgroundColor: 'black'

            },

            {

                label: 'trolley 2',

                fill: false,

                data: [22, 15, 67, 44],

                borderColor: 'red',

                backgroundColor: 'red'

            }

            ]

        },

        options: {

            legend: {

                display:false,

                pointStyle: 'circle'

            }

        }

    });

</script>

ramen.css

body {

    margin: 0;

    padding: 0;

    list-style: none;

    text-decoration: none;

    overflow: hidden;

    font-family: Arial, Helvetica, sans-serif;

}

.wrap {

    height: 100%;

    margin: 0px;

    display: flex;

    box-sizing: border-box;

}

.header-box {

    width: 100vw;

    height: 30px;

    display: flex;

    background: silver;

    padding: 5px;

    flex-direction: column;

}

.header-box .headerlogo img {

    width: 30px;

    height: 30px;

    float: left;

    border-radius: 50%;

}

.header-box .headerlogo .header-text {

    font-weight: bold;

    display: inline-block;

    padding-left: 5px;

    padding-top: 5px;

}

.header-box .headerlogo .logoutheader {

    float: right;

    color: lightgrey;

    padding-right: 10px;

    padding-top: 5px;

}

.header-box .headerlogo .logoutheader a {

    color: grey;

}

.sidebar {

    float: left;

    width: 185px;

    height: 90vh;

    background: lightgrey;

    position: relative;

    padding: 0px;

}

.sidebar .profile-pic {

    color: black;

    text-align: center;

    font-size: x-small;

    margin-top: 10px;

}

.sidebar .profile-pic img {

    width: 70px;

    height: 70px;

    position: relative;

    border-radius: 50%;

}

.sidebar ul {

    padding: 0px;

    margin-top: 0px;

    margin-bottom: 0px;

}

.sidebar a {

    color: black;

    display: block;

    text-decoration: none;

    padding: 13px;

    font-size: smaller;

}

.sidebar a:hover {

    color: white;

    background: gray;

}

.contentbox {

    width: 75%;

    padding: 10px;

}

.contentbox .active-trolley {

    margin-left: auto;

    margin-right: auto;

    width: 350px;

    height: 100px;

    box-shadow: 1px 1px 1px 1px lightgray;

}

.contentbox .active-trolley .trolley-title {

    font-size: small;

    padding: 8px;

}

.contentbox .active-trolley .number-trolley {

    font-size: xx-large;

    font-weight: bold;

    text-align: center;

}

.contentbox .active-trolley .trolley-footer {

    font-size: small;

    text-align: right;

    padding: 5px;

}

.contentbox .graphbox {

    margin-left: auto;

    margin-right: auto;

    width: 95%;

    height: 80%;

    box-shadow: 1px 1px 1px 1px lightgray;

    margin-top: 30px;

}

**Question 1b)**

home.html

<!DOCTYPE html>

<html>

    <head>

        <script src="https://www.chartjs.org/dist/2.9.3/Chart.min.js"></script>

        <meta charset="UTF-8">

        <link rel="stylesheet" href="css/ramen.css">

    </head>

    <body>

        <div class="header-box">

            <div class="headerlogo">

                <img src="img/DCSlogo.png">

                <div class="header-text">DCS Smart Trolley</div>

                <div class="logoutheader"><a href="#">Logout</a></div>

            </div>

        </div>

        <div class="wrap">

            <div class="sidebar">

                <div class="profile-pic">

                    <img src="img/blankprofile.png">

                    <p>sabina001@suss.edu.sg</p>

                </div>

                <ul><a href="#">Dashboard View</a></ul>

                <ul><a href="#">Account</a></ul>

                <ul><a href="#">Configure Email</a></ul>

                <ul><a href="#">Download</a></ul>

                <ul><a href="#">Logout</a></ul>

            </div>

            <div class="contentbox">

                <div class ="summarybox">

                    <div class="active-trolley">

                        <div class="trolley-title">

                            Active Trolleys

                        </div>

                        <div class="number-trolley">

                            2

                        </div>

                        <div class="trolley-footer">

                            Total: 6

                        </div>

                    </div>

                    <div class="trolleyTemp">

                        <div class="date">

                            Tuesday, August 11 2020

                        </div>

                        <div class="tempheader">

                            Highest Temperature

                            <div class="resultheader">

                                30.0 (trolley1)

                            </div>

                        </div>

                        <br>

                        <div class="tempheader">

                            Lowest Temperature

                            <div class="resultheader">

                                11.0 (trolley1)

                            </div>

                        </div>

                    </div>

                </div>

                <div class="graphbox" style="width: 100%;height: 50vh;">

                    <h4>Temperature Recording</h4>

                    <p style="font-size: 12px;">Today's Recording</p>

                    <canvas id="trolleychart"></canvas>

                </div>

            </div>

        </div>

    </body>

</html>

<script>

    Chart.defaults.global.legend.labels.usePointStyle = true;

    Chart.defaults.global.title.align = 'start';

    var trolleychart = document.getElementById("trolleychart").getContext('2d');

    var tempChart = new Chart(trolleychart, {

        type:'line',

        data: {

            labels: ['Tue 11', '3am', '6am', '9am', '12pm', '3pm', '6pm', '9pm','Wed 12'],

            datasets: [{

                label: 'trolley 1',

                fill: false,

                data: [11, 30, 29, 15, 20, 25, 27, 14, 12],

                borderColor: 'black',

                backgroundColor: 'black'

            },

            {

                label: 'trolley 2',

                fill: false,

                lineTension: 0,

                data: [12, 18, 19, 29, 22, 21, 27, 14, 20],

                borderColor: 'red',

                backgroundColor: 'red'

            }

            ]

        },

        options: {

            title: {

                display:false,

                text: ['Temperature Recording', "Today's Recording"],

            },

            responsive: true,

            maintainAspectRatio: false,

            legend: {

                display:true,

                align: 'start',

            },

            scales: {

                yAxes: [{

                    ticks: {

                        beginAtZero: true,

                        padding: 15,

                    }

                }]

            }

        }

    });

</script>

ramen.css

body {

    margin: 0;

    padding: 0;

    list-style: none;

    text-decoration: none;

    font-family: Verdana, Tahoma, sans-serif;

    overflow: hidden;

}

.wrap {

    height: 100%;

    margin: 0px;

    display: flex;

    box-sizing: border-box;

}

.header-box {

    width: 100vw;

    height: 30px;

    display: flex;

    background: silver;

    padding: 5px;

    flex-direction: column;

}

.header-box .headerlogo img {

    width: 30px;

    height: 30px;

    float: left;

    border-radius: 50%;

}

.header-box .headerlogo .header-text {

    font-weight: bold;

    display: inline-block;

    padding-left: 5px;

    padding-top: 5px;

}

.header-box .headerlogo .logoutheader {

    float: right;

    color: lightgrey;

    padding-right: 10px;

    padding-top: 5px;

}

.header-box .headerlogo .logoutheader a {

    color: grey;

}

.sidebar {

    float: left;

    width: 185px;

    height: 100vh;

    background: lightgrey;

    position: relative;

    padding: 0px;

}

.sidebar .profile-pic {

    color: black;

    text-align: center;

    font-size: x-small;

    margin-top: 10px;

}

.sidebar .profile-pic img {

    width: 70px;

    height: 70px;

    position: relative;

    border-radius: 50%;

}

.sidebar ul {

    padding: 0px;

    margin-top: 0px;

    margin-bottom: 0px;

}

.sidebar a {

    color: black;

    display: block;

    text-decoration: none;

    padding: 13px;

    font-size: smaller;

}

.sidebar a:hover {

    color: white;

    background: gray;

}

.contentbox {

    width: 80%;

    height: 90vh;

    padding: 10px;

    display: flex;

    flex-direction: column;

}

.summarybox {

    width: 80%;

    height: 20%;

    display: flex;

    flex-direction: row;

    padding: 5px;

    justify-content: space-evenly;

}

.summarybox .active-trolley {

    margin-left: auto;

    margin-right: auto;

    width: 300px;

    height: 120px;

    box-shadow: 1px 1px 1px 1px lightgray;

}

.summarybox .active-trolley .trolley-title {

    font-size: small;

    padding: 8px;

}

.summarybox .active-trolley .number-trolley {

    font-size: xx-large;

    font-weight: bold;

    text-align: center;

    padding: 10px;

}

.summarybox .active-trolley .trolley-footer {

    font-size: small;

    text-align: right;

    padding: 5px;

}

.graphbox {

    margin-left: auto;

    margin-right: auto;

    margin-top: 20px;

    position: relative;

}

.summarybox .trolleyTemp {

    margin-left: auto;

    margin-right: auto;

    width: 300px;

    height: 120px;

    box-shadow: 1px 1px 1px 1px lightgray;

}

.summarybox .trolleyTemp .date {

    font-size: x-small;

    color: silver;

    text-align: right;

    padding: 5px;

}

.summarybox .trolleyTemp .tempheader {

    font-size: 12px;

    padding-left: 10px;

}

.summarybox .trolleyTemp .tempheader .resultheader {

    font-size: x-small;

    color: lightgrey;

}

**Question 2a)**

Data model:

**Collection name** : trolley **(name of the collection in mongodb)**

***Attributes saved in collection trolley***

* Name : : str name of the trolley
* Date :: datetime object
* Temp :: numerical (int) of the temperature of the trolley

For q2b, I have manually keyed in the data in mongodb

**Routes**:

* /dashboard with def dashboard\_view() : to view the page and chart
* /upload/ : routes to create.html where can upload csv file and also the same page to show all added data on the database (for q3)
* /upload/create: routes to create.html where can add records into db (for q3)
* /trolley : routes to the trolley.html and shows the data so far in the mongodb
* /trolley/create : routes to the form.html to create and add data into the mongodb
* /trolley/<datehere> : pulls out only specific data from date in the mongodb and represents in the trolley.html

**Templates:**

* Dashboard.html
  + Passed data of active trollies, total trollies, the filtered query, highest and lowest temperature
* Create.html
  + Uploading and creating of documents into mongodb
  + Shows the entire records in trolley collection
* Trolley.html
  + Show the entire data in the database in readable format
* Form.html
  + To add trolley data into the database

**Controllers:**

* Get\_high() : getting the highest temperature for the day
* Get\_low() : getting the lowest temperature for the day
* f\_date() : extracting all the documents in the collection trolley with filtered results of date (mainly for the chartjs)
* Count\_date() : counting all the active trolleys in the collection
* Count\_all() : counts all the trolleys in the collection
* Create\_recording() : create recording (for q3)
* Find\_all() : finds all records and sorts the latest document as first (for q3)

**Question 2b, c, d)**

app.py

#app.py

from flask import Flask, render\_template, request

import json

from datetime import datetime

app = Flask(\_\_name\_\_)

import repo

@app.route("/trolley/<strdate>")

def test\_date(strdate):

    #convert to dt obj

    dd = datetime.strptime(strdate, '%Y-%m-%d')

    print(type(dd))

    print(type(strdate))

    print(dd.year)

    print(dd.month)

    print(dd.day)

    return render\_template("trolley.html", dlist=repo.f\_date(dd))

@app.route("/trolley/delete", methods=['POST','GET'])

def trolley\_delete():

    if (request.method == 'POST'):

        repo.delete\_trolley(request.form.get('name'))

        return "Thank you for submitting"

    return render\_template("form.html")

@app.route("/trolley/update", methods=['POST','GET'])

def update\_trolley():

    if (request.method == 'POST'):

        repo.update\_trolley(request.form.get('name'), request.form.get('date'))

        return "Thank you for updating"

    return render\_template("update.html")

@app.route("/trolley/create", methods=['POST','GET'])

def trolley\_create():

    if (request.method == 'POST'):

        repo.create\_trolley(request.form.get('name'), request.form.get('date'), request.form.get('temp'))

        return "Thank you for submitting"

    return render\_template("form.html")

@app.route("/trolley")

def trolley\_read():

    return render\_template("trolley.html", dlist=repo.f\_date())

@app.route("/dashboard")

def dashboard\_view():

    t1\_list = []

    t2\_list = []

    dd = datetime.strptime('2020-08-11', '%Y-%m-%d')

    for i in repo.f\_date():

        if i['name'] == 'trolley1':

            t1\_list.append(i['temp'])

        if i['name'] == 'trolley2':

            t2\_list.append(i['temp'])

    #print(t1\_list)

    #print(t2\_list)

    return render\_template("dashboard.html", t1=t1\_list, t2=t2\_list, total=repo.count\_all(), cactive = repo.count\_active(), xlabel=repo.f\_date(dd), highestTemp=repo.get\_high(), lowestTemp=repo.get\_low())

#DateObj = datetime.strptime(date\_string, "%Y-%m-%d") to convert into dateobj

#to plug in the data and labels, pull out all the data into a string and plug into the chart

if \_\_name\_\_ == "main":

    app.run()

repo.py

import pymongo

from datetime import datetime, timedelta

# Configure MongoDB

connection = pymongo.MongoClient('mongodb://localhost:27017')

db = connection['TMA']

#collection : trolley

def get\_high(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.month, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

    return db.trolley.find(filter).sort("temp",-1).limit(1) #get only the highest

def get\_low(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.month, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

    return db.trolley.find(filter).sort("temp",+1).limit(1) #get only the lowest

def f\_date(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.month, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

    return db.trolley.find(filter)

def count\_active(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.month, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

        results = db.trolley.distinct("name")

        r\_len = len(results)

        return r\_len

def count\_all(): #gets all unique trolley names in the collection

    results = db.trolley.distinct("name")

    r\_len = len(results)

    return r\_len

def create\_trolley(name, date, temp):

    assert name is not None

    assert date is not None

    assert temp is not None

    DateObj = datetime.strptime(date, "%Y-%m-%d")

    db.trolley.insert\_one({'name':name, 'date': DateObj, 'temp': temp})

def update\_trolley(name, new\_date):

    assert name is not None

    assert new\_date is not None

    db.trolley.update\_one({'name':name}, {'$set':{'date':new\_date}})

def delete\_trolley(name):

    assert name is not None

    db.trolley.delete\_one({'name':name})

dashboard.html

<!DOCTYPE html>

<html>

    <head>

        <script src="https://www.chartjs.org/dist/2.9.3/Chart.min.js"></script>

        <meta charset="UTF-8">

        <link rel="stylesheet" href= {{url\_for('static', filename='ramen.css')}}>

        <title>DSC Smart Trolley</title>

    </head>

    <body>

        <div class="header-box">

            <div class="headerlogo">

                <img src={{url\_for('static', filename="DCSlogo.png")}}>

                <div class="header-text">DCS Smart Trolley</div>

                <div class="logoutheader"><a href="#">Logout</a></div>

            </div>

        </div>

        <div class="wrap">

            <div class="sidebar">

                <div class="profile-pic">

                    <img src={{url\_for('static', filename="blankprofile.png")}}>

                    <p>sabina001@suss.edu.sg</p>

                </div>

                <ul><a href="#">Dashboard View</a></ul>

                <ul><a href="#">Account</a></ul>

                <ul><a href="#">Configure Email</a></ul>

                <ul><a href="#">Download</a></ul>

                <ul><a href="#">Logout</a></ul>

            </div>

            <div class="contentbox">

                <div class ="summarybox">

                    <div class="active-trolley">

                        <div class="trolley-title">

                            Active Trolleys

                        </div>

                        <div class="number-trolley">

                            {{ cactive }}

                        </div>

                        <div class="trolley-footer">

                            Total: {{ total }}

                        </div>

                    </div>

                    <div class="trolleyTemp">

                        <div class="date">

                            <script>document.write(new Date().toDateString())</script>

                        </div>

                        <div class="tempheader">

                            Highest Temperature

                            <div class="resultheader">

                                {% for trolley in highestTemp %}

                                    {{trolley.temp}} ({{trolley.name}})

                                {% endfor %}

                                </div>

                        </div>

                        <br>

                        <div class="tempheader">

                            Lowest Temperature

                            <div class="resultheader">

                                {% for trolley in lowestTemp %}

                                    {{trolley.temp}} ({{trolley.name}})

                                {% endfor %}

                            </div>

                        </div>

                    </div>

                </div>

                <div class="graphbox" style="width: 100%;height: 50vh;">

                    <h4>Temperature Recording</h4>

                    <p style="font-size: 12px;">Today's Recording</p>

                    <canvas id="trolleychart"></canvas>

                </div>

            </div>

        </div>

        <script>

            Chart.defaults.global.legend.labels.usePointStyle = true;

            Chart.defaults.global.title.align = 'start';

            var trolleychart = document.getElementById("trolleychart").getContext('2d');

            var LABELDATA = [

                {% for x in xlabel %}

                    '{{x.date.strftime('%H:%M%p')}}',

                {% endfor %}]

            var chart1 = [

                            {% for i in t1 %}

                                    {{i}},

                                {% endfor %}

            ]

            var chart2 = [

                            {% for i in t2 %}

                                    {{i}},

                                {% endfor %}

            ]

            var tempChart = new Chart(trolleychart, {

                type:'line',

                data: {

                    labels: LABELDATA,

                    datasets: [{

                        label: "Trolley 1",

                        data: chart1

                    },

                    {

                        label: "Trolley 2",

                        data: chart2

                    }

                    ]

                },

                options: {

                    title: {

                        display:false,

                        text: ['Temperature Recording', "Today's Recording"],

                    },

                    responsive: true,

                    maintainAspectRatio: false,

                    legend: {

                        display:true,

                        align: 'start',

                    },

                    scales: {

                        yAxes: [{

                            ticks: {

                                beginAtZero: true,

                                padding: 15,

                            }

                        }]

                    }

                }

            });

        </script>

    </body>

</html>

form.html

<!DOCTYPE html>

<html>

    <head>

    </head>

    <body>

        <form method='POST'>

            Trolley Name: <input name="name"><br>

            Date: <input name='date'>YYY-MM-DD <br>

            Temperature: <input name='temp'><br>

            <button>Submit</button><br>

        </form>

    </body>

</html>

Trolley.html

<!DOCTYPE html>

<html>

    <head>

    </head>

    <body>

        <h1>Trolley List</h1>

        <hr>

        <br>

        <table border ="1">

            <tr>

                <th>Name</th>

                <th>Date Time</th>

                <th>Temperature</th>

        </tr>

        {% for trolley in dlist %}

        <tr>

            <td>{{trolley.name}}</td>

            <td>{{trolley.date}}</td>

            <td>{{trolley.temp}}</td>

        </tr>

        {% endfor %}

        </table>

        </table>

        <br>

    </body>

</html>

Ramen.css

body {

    margin: 0;

    padding: 0;

    list-style: none;

    text-decoration: none;

    font-family: Verdana, Tahoma, sans-serif;

    overflow: hidden;

}

.wrap {

    height: 100%;

    margin: 0px;

    display: flex;

    box-sizing: border-box;

}

.header-box {

    width: 100vw;

    height: 30px;

    display: flex;

    background: silver;

    padding: 5px;

    flex-direction: column;

}

.header-box .headerlogo img {

    width: 30px;

    height: 30px;

    float: left;

    border-radius: 50%;

}

.header-box .headerlogo .header-text {

    font-weight: bold;

    display: inline-block;

    padding-left: 5px;

    padding-top: 5px;

}

.header-box .headerlogo .logoutheader {

    float: right;

    color: lightgrey;

    padding-right: 10px;

    padding-top: 5px;

}

.header-box .headerlogo .logoutheader a {

    color: grey;

}

.sidebar {

    float: left;

    width: 185px;

    height: 100vh;

    background: lightgrey;

    position: relative;

    padding: 0px;

}

.sidebar .profile-pic {

    color: black;

    text-align: center;

    font-size: x-small;

    margin-top: 10px;

}

.sidebar .profile-pic img {

    width: 70px;

    height: 70px;

    position: relative;

    border-radius: 50%;

}

.sidebar ul {

    padding: 0px;

    margin-top: 0px;

    margin-bottom: 0px;

}

.sidebar a {

    color: black;

    display: block;

    text-decoration: none;

    padding: 13px;

    font-size: smaller;

}

.sidebar a:hover {

    color: white;

    background: gray;

}

.contentbox {

    width: 80%;

    height: 90vh;

    padding: 10px;

    display: flex;

    flex-direction: column;

}

.summarybox {

    width: 80%;

    height: 20%;

    display: flex;

    flex-direction: row;

    padding: 5px;

    justify-content: space-evenly;

}

.summarybox .active-trolley {

    margin-left: auto;

    margin-right: auto;

    width: 300px;

    height: 120px;

    box-shadow: 1px 1px 1px 1px lightgray;

}

.summarybox .active-trolley .trolley-title {

    font-size: small;

    padding: 8px;

}

.summarybox .active-trolley .number-trolley {

    font-size: xx-large;

    font-weight: bold;

    text-align: center;

    padding: 10px;

}

.summarybox .active-trolley .trolley-footer {

    font-size: small;

    text-align: right;

    padding: 5px;

}

.graphbox {

    margin-left: auto;

    margin-right: auto;

    margin-top: 20px;

    position: relative;

}

.summarybox .trolleyTemp {

    margin-left: auto;

    margin-right: auto;

    width: 300px;

    height: 120px;

    box-shadow: 1px 1px 1px 1px lightgray;

}

.summarybox .trolleyTemp .date {

    font-size: x-small;

    color: silver;

    text-align: right;

    padding: 5px;

}

.summarybox .trolleyTemp .tempheader {

    font-size: 12px;

    padding-left: 10px;

}

.summarybox .trolleyTemp .tempheader .resultheader {

    font-size: x-small;

    color: lightgrey;

}

**Question 3a, b, c)**

app.py

#app.py

from flask import Flask, render\_template, request, flash, redirect, url\_for

from datetime import datetime

from werkzeug.utils import secure\_filename

import repo

import csv

import io

import os

secretkey = os.urandom(12).hex()

app = Flask(\_\_name\_\_)

app.secret\_key = secretkey

trolleyList = list(repo.find\_all())

#upload csv file

@app.route("/")

def index():

    return render\_template("dashboard.html")

@app.route("/upload/", methods=['POST', 'GET'])

def upload\_file():

    if (request.method=='POST'):

        #works to parse csv

        upfile = request.files['csvfile']

securef = secure\_filename(upfile.filename)

        stream = io.StringIO(securef.stream.read().decode("UTF-8-sig"), newline=None)

        fdata = csv.reader(stream, delimiter=',')

        x = list(fdata)

        for item in x:

            #print(item[0])

            #print(item[1])

            #print(item[2])

            fname = item[0]

            fdate = datetime.strptime(item[1], '%Y-%m-%dT%H:%M:%S')

            ftemp = int(item[2])

            #add to db

            repo.create\_recording(fname, fdate, ftemp)

flash(f’Successfully uploaded filename {securef}’)

        return redirect(url\_for('create\_recording'))

    return render\_template("create.html", recordings=trolleyList)

@app.route("/upload/create/", methods=['POST','GET'])

def create\_recording():

    if (request.method=='POST'):

        strName = request.form.get('tname')

        strDate = request.form.get('tdate')

        strTime = request.form.get('ttime')

        fTemp = request.form.get('ttemp')

        intTemp = int(fTemp)

        tdt = strDate + 'T' + strTime + ':00' #combine the date and time together because firefx is a weak browser

        dateObj = datetime.strptime(tdt, '%Y-%m-%dT%H:%M:%S')

        repo.create\_recording(strName, dateObj, intTemp)

        return redirect(url\_for('create\_recording'))

    return render\_template("create.html", recordings=trolleyList)

@app.route("/trolley/delete", methods=['POST','GET'])

def trolley\_delete():

    if (request.method == 'POST'):

        repo.delete\_trolley(request.form.get('name'))

        return "Thank you for submitting"

    return render\_template("form.html")

@app.route("/trolley/update", methods=['POST','GET'])

def update\_trolley():

    if (request.method == 'POST'):

        repo.update\_trolley(request.form.get('name'), request.form.get('temp'))

        return "Thank you for submitting"

    return render\_template("form.html")

@app.route("/trolley/record", methods=['POST','GET'])

def trolley\_create():

    if (request.method == 'POST'):

        repo.create\_trolley(request.form.get('name'), request.form.get('date'), request.form.get('temp'))

        return "Thank you for submitting"

    return render\_template("form.html")

@app.route("/dashboard")

def dashboard\_view():

    data = repo.f\_date()

    return render\_template("dashboard.html", total=repo.count\_all(), cactive = repo.count\_active(), xlabel=data, highestTemp=repo.get\_high(), lowestTemp=repo.get\_low())

if \_\_name\_\_ == "main":

    app.run()

repo.py

import pymongo

from datetime import datetime, timedelta

# Configure MongoDB

connection = pymongo.MongoClient('mongodb://localhost:27017')

db = connection['TMA']

#collection: trolley

def create\_recording(name, date, temp): #str, dateobj, int

    assert name is not None

    assert date is not None

    assert temp is not None

    db.trolley.insert\_one({"name":name, "date":date, "temp":temp})

def find\_all(): #for the recordings creation page

    return db.trolley.find({}).sort("date", -1) #find latest inserted date

def get\_high(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.nmonth, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

    return db.trolley.find(filter).sort("temp",-1).limit(1) #get only the highest

def get\_low(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.nmonth, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

    return db.trolley.find(filter).sort("temp",+1).limit(1) #get only the lowest

def f\_date(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.nmonth, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

    return list(db.trolley.find(filter))

def count\_active(objDate=None):

    filter = {}

    if objDate is not None:

        start=datetime(objDate.year, objDate.nmonth, objDate.day) #get the start of the date

        end=start+timedelta(days=1) # this get the ending

        filter['date'] = { #this filters only the time between start and end date will appear

            '$lt':end,

            '$gte':start

        }

        results = db.trolley.distinct("name")

        r\_len = len(results)

        return r\_len

def count\_all(): #gets all unique trolley names in the collection

    results = db.trolley.distinct("name")

    r\_len = len(results)

    return r\_len

def create\_trolley(name, date, temp):

    assert name is not None

    assert date is not None

    assert temp is not None

    db.trolley.insert\_one({'name':name, 'date': date, 'temp': temp})

def update\_trolley(name, new\_temp):

    assert name is not None

    assert new\_temp is not None

    db.trolley.update\_one({'name':name}, {'$set':{'temp':new\_temp}})

def delete\_trolley(name):

    assert name is not None

    db.trolley.delete\_one({'name':name})

def all\_trolley():

    return list(db.trolley.find({}))

create.html

<!DOCTYPE html>

<html>

    <head>

        <title>Recordings</title>

        <meta charset="UTF-8">

    </head>

    <body>

        <h1>Create a Recording</h1>

        <form class="creation" action="/upload/create/" method='POST'>

            <label for ="trolley\_name">Trolley:</label>

            <input type="text" id="trolley\_name" name="tname"><br>

            <label for="trolley\_date">Date & Time:</label>

            <!--<input type="datetime-local" id="trolley\_date" name="tdate"><br> #nosupported in ff browser-->

            <input type="date" id="trolley\_date" name="tdate"><input type="time" name="ttime"> (HH:MM AM/PM)<br>

            <label for="trolley\_temp">Temperature:</label>

            <input type="text" id ="trolley\_temp" name ="ttemp"><br>

            <button type="submit" value="Record" id="submit">Record</button>

        </form>

{% with messages = get\_flashed\_messages() %}

{% if messages %}

{% for msg in messages %}

<li>{{ msg }}</li>

{% endfor %}

{% endwith %}

        <hr>

        <h1>Upload a Recording</h1>

        <form action="/upload/" enctype="multipart/form-data" accept=".csv" method="POST">

            <input type ="file" name="csvfile">

            <button type="submit" value="Upload" id="submit">Upload</button>

        </form>

        <hr>

        <h1>Recordings</h1>

        <table border='1'>

            <th>Trolley</th>

            <th>Date & Time</th>

            <th>Temperature</th>

            {% for item in recordings %}

            <tr>

                <td>{{item.name}}</td>

                <td>{{item.date}}</td>

                <td>{{item.temp}}</td>

                </tr>

            {% endfor %}

        </table>

    </body>

</html>