Assignment 2

- Due Wednesday by 11:59p.m.
- Points 0
- Available after Feb 26 at 12p.m.

Assignment 2: Building a Dynamic Course Website & Flask-Based Web Application

Topics Covered: HTML, CSS, Python Flask, Web Development

Due Date: March 12

Points: 105

Group Work

- **Group Size**: You must work in a group of **two or three.**
- **Group Declaration**: Ensure all group members are declared in MarkUs before submission.

Question 1: Flask Web Application (30 points)

In this question, you will create a **Flask web application** that dynamically processes user input through different routes. You will install Flask on your local machine and write a Python script (`Question1.py`) that generates dynamic HTML responses.

1.1: String Manipulation & Web Response (20 points)

- Create a **Flask route** that processes a given name according to the following rules:
  - If the name is **all uppercase**, convert it to **lowercase**.
  - If the name is **all lowercase**, convert it to **uppercase**.
  - If the name contains **both uppercase and lowercase letters**, standardize it to **title case**.
  - If the name contains **numbers**, remove all numbers and return the remaining letters in **uppercase**.
  - Remove any **leading or trailing spaces** before processing.

Your Flask application should return an **HTML response** in the following format:

```
"Welcome, <Processed Name>, to my CSCB20 website!"
```

Example Test Cases:

URL Request	Response
`http://localhost:5000/alice`	`"Welcome, ALICE, to my CSCB20 website!"`
`http://localhost:5000/ALICE`	`"Welcome, alice, to my CSCB20 website!"`
`http://localhost:5000/aLiCe`	`"Welcome, Alice, to my CSCB20 website!"`
`http://localhost:5000/al1c3`	`"Welcome, ALC, to my CSCB20 website!"`
`http://localhost:5000/Alice`	`"Welcome, Alice, to my CSCB20 website!"`

You can assume that there will be no space between names, and that we will not deviate from the above syntax that is provided to you for testing your code.

1.2: Palindrome Detector (5 points)

Extend your Flask application to check if the **processed name** is a **palindrome** (i.e., it reads the same forward and backward).

- If the processed name is a **palindrome**, return the response: `"Welcome, <Processed Name>. Your name is a palindrome!"`
- Otherwise, return the standard greeting format from Section **1.1**.

Example Test Case:

URL Request	Response
`http://localhost:5000/anna`	`"Welcome, anna. Your name is a palindrome!"`

1.3: Unicode Emoji Name Generator (5 points)

Create an additional Flask route that **replaces vowels in the name with corresponding emojis** based on the following mappings:

Vowel	Emoji
A, a	🔺

E, e 🎗️
I, i 👁️
O, o 🔵
U, u 🆙

Your Flask application should return an HTML response in the format:

`"Welcome, <Processed Name>, to my CSCB20 website!"`

Example Test Cases:

URL Request	Response
`http://localhost:5000/emoji/emma`	`"Welcome, 🎗️mm🔺, to my CSCB20 website!"`
`http://localhost:5000/emoji/alice`	`"Welcome, 🔺l👁️c🎗️, to my CSCB20 website!"`
`http://localhost:5000/emoji/xyz` (No vowels)	`"Welcome, xyz, to my CSCB20 website!"`

How to Run and Test Your Flask Application

Make sure you have Flask installed on your computer.

Run Your Flask App Using the Terminal

You can run your code in multiple ways, however, if you wish to run your code using the terminal on your computer

For **Mac/Linux Users**, run:

```
env FLASK_APP=Question1.py flask run --host 0.0.0.0
```

For **Windows Users**, run:

```
set FLASK_APP=Question1.py flask run --host 0.0.0.0
```

Once running, **test your application** by entering URLs like:

`http://localhost:5000/alice`
`http://localhost:5000/anna`
`http://localhost:5000/emoji/emma`

Question 2: Course Website Development (70 points)

2.1: Course Website Redesign (40 points)

You will design and build a **new course website for CSCB20** using **only HTML and CSS**. The goal is to create a **modern, responsive, and user-friendly website** that effectively presents course content.

Website Requirements:

- **Usability:** The layout should be **clear, easy to navigate, and well-structured**.
- **Responsiveness:** The website must adapt to **different screen sizes** (desktop, tablet, and mobile).
- **Aesthetic Design:** The website should **look modern and polished**, following a **consistent design theme**.

Development Constraints:

You must use:

- **CSS Grid** or **Flexbox** to structure the page layout.
- **Consistent font styles, color themes, and navigation menus.**
- **A sticky header** that remains visible at the top when scrolling.
- **A footer** containing:
  - A link to the **Faculty of Computer Science at UofT**.
  - A short blurb with the website creator's name (e.g., `"Site design by Jane Doe"`).
  - Footer must stick to the bottom of the webpage, should not be visible until the user reaches the bottom of the page

You CANNOT use:

- **Bootstrap, jQuery, or any pre-built CSS frameworks.**

Required Content:

The new website must include **all the content from the original CSCB20 website**, including:
Navigation Menu: Functional links to:

- **Home**

- **Syllabus**
- **Assignments**
- **Labs**
- **Lecture Notes** *(Clicking should open a PDF file with lecture content)*
- **Piazza**
- **Markus**
- **Anon Feedback**
- **Course Team**

Mobile Responsiveness:

- Use **CSS media queries** to ensure proper display on **phones and tablets**.
- The layout should **adjust dynamically** on smaller screens.

2.2: Advanced Features (30 points)

Enhance your course website by adding the following **CSS-based interactive features:**

Required Enhancements:

1. **Dropdown Menus (10 points):**
   - Use **CSS-only hover-based dropdown menus** for navigation.
2. **Hover Effects (5 points):**
   - Links should **change colour or style** when hovered over.
3. **CSS Animations or Transitions (5 points):**
   - Implement smooth **animations or transitions** to make the website feel **more interactive**.
4. **Dark Mode Styling (10 points) [CSS ONLY]:**
   - Create an **alternative dark mode** using **CSS variables or a separate dark mode stylesheet**.
   - Use **CSS pseudo-classes (:root, :hover, :focus, etc.)** to implement the styling.

2.3: Mockups & User Stories (10 points)

Before writing **any HTML or CSS code**, you must first **generate mockups** and **write user stories**

that outline your website's design and functionality.

Each **mockup** will correspond to a **single user story**, demonstrating how a user will **interact with the website**.

Deliverables:

1. User Stories (text files) that describe **specific interactions**.

- Write **at least 10 user stories**, each describing a **specific feature or interaction**.
- **Each user story must match a mockup file.**
  - Example: `mockup1.png` should correspond to `userstory1.txt`.
- **File naming convention:** `userstory1.txt`, `userstory2.txt`, etc.
- User stories must be written in the following format:
  `As a <user role>, I want to <goal> so that <reason>.`
- **Example User Stories:**
  - `"As a student, I should be able to click on the 'Assignments' link and view all assignments."`
  - `"As an instructor, I should be able to update lecture notes easily."`

2. Mock-ups (PNG files) Visual Sketches of Webpages

- **Create at least 10 mockups**, each corresponding to a user story.
- A **mockup** is a **full-size model of a design** that shows how a webpage will look and function.
- Each **mockup file must be saved as a PNG** (`mockup1.png`, `mockup2.png`, etc.).
- Your **mockups should be labeled clearly** to reflect the features of your website.
- **Use any tool** of your choice, such as:
  - **Gomockingbird (https://gomockingbird.com/home)**
  - **Figma (https://www.figma.com/)**
  - **Wireframe.cc (https://wireframe.cc/)**
  - **Balsamiq (https://balsamiq.com/)**
  - **Hand-drawn sketches** (scan or take a clear picture and save as PNG)

Additional Website Requirements

Table-less Design (No `<table>` elements allowed)

- You **must not use tables for layout purposes**.
- Use **CSS Grid or Flexbox** to structure the website.

No Predefined or Third-Party CSS Frameworks

- **You cannot use Bootstrap, Tailwind, jQuery, or any pre-built HTML templates.**
- All **HTML and CSS must be written from scratch**.

Separate CSS File

- Your **HTML structure must be separate** from CSS styles.
- Example:
  - The main entry page must be called `index.html`.
  - The styles for `index.html` should be in `index.css`.

Mobile-Friendly Design

- Your course website must **work well on mobile devices**.
- Use **CSS media queries** to ensure **responsive design**.
- Consider the following adjustments for mobile:
  - **Stacking elements vertically** for small screens.
  - **Hiding non-essential elements** in compact layouts.
  - **Larger touch-friendly buttons**.

2.4: Submission & Deliverables

You must submit a **ZIP file (`A2.zip`)** on Markus.

Folder Structure:

```
A2/
├── MockUpAndUserStories/
│   ├── mockup1.png
│   ├── userstory1.txt
├── src/
│   ├── index.html  # Main entry page
│   ├── index.css   # CSS for index.html
├── HonorCode.txt
├── Report.pdf
├── Question1.py
```

Honor Code File (`HonorCode.txt`)

- Add the **following statement**, signed with your and your partner's names:

```
Honour Code: We (your names) pledge that this assignment represents our own work. We did not receive unauthorized help and understand the consequences of academic dishonesty. We have also read the plagiarism section in the course info sheet of CSC B20 and understand the consequences.
```

- **If this file is absent, we will not mark your assignment, and you will receive zero on the entire assignment.**

Report (`Report.pdf`) (5 points)

Write a **1-2 page report** covering:

- Issues with previous course websites.
- Improvements in your redesigned site.
- Challenges you faced and how you overcame them.

Resources that may help:

- CSS flex: **https://internetingishard.netlify.app/html-and-css/flexbox/ (https://internetingishard.netlify.app/html-and-css/flexbox/)**
- CSS grid layout: **https://learncssgrid.com (https://learncssgrid.com/)**
- Online sandbox for trying out basic ideas: **https://codepen.io/LandonSchropp/pen/KpzzGo (https://codepen.io/LandonSchropp/pen/KpzzGo)**
- HTML wireframe mockup:
  - **https://gomockingbird.com/home (https://gomockingbird.com/home)**
  -
- Story mapping:
  - **https://www.eecg.utoronto.ca/~shuruiz/teaching/ECE444-2020F/slides/Lec7.1-StoryMapping&Risk&Prototype.pdf (https://www.eecg.utoronto.ca/~shuruiz/teaching/ECE444-2020F/slides/Lec7.1-StoryMapping&Risk&Prototype.pdf)**
  - **https://www.atlassian.com/agile/project-management/user-stories (https://www.atlassian.com/agile/project-management/user-stories)**