|  |  |  |
| --- | --- | --- |
|  |  | **ISM 6225**  **Distributed information systems** |

Assignment 3 – Look and Feel

Primary objective: Introduction to web interface technologies

Secondary objective: Develop familiarity with user interface design

## Introduction

Web user interfaces are a common mechanism by which users interact with distributed information systems. In this assignment, you will create a static version of a data-driven web site, using the common interface elements such as images, styles and html tags as well as simple interactivity using JavaScript. Later in the course, you will gain familiarity with active technologies, which will generate these web pages based on data sources such as APIs and databases.

The goal of this exercise is to draw students’ attention to good user interactions. You are asked to plan the data-driven website you will build in assignment 5, and create a static version of the site. This is also an opportunity for students to become familiar with the common html, css and JavaScript elements. There are numerous resources online to learn html, css and JavaScript. The instructor’s favorite newbie guide to html and css continues to be the NCSA guide to html: <http://main.put.com/HTMLPrimer.html>. For JavaScript, please keep expectations simple. Just introduce some simple JavaScript capabilities, primarily to learn how JavaScript is used on websites.

## Activity

<https://www.data.gov/> is the federal government’s data site. Your task in assignments 3 and 5 is to develop an interactive website that helps users do something useful with this data. In this assignment, you will build the user interface of the website using static html and css and any relevant JavaScript. You will also plan the data model for the website. In the next assignment, you will build the interactivity and persistence of the website and host it on the cloud. As discussed in class, we recommend data.gov because it is simple to use and is appropriate for a first stab at consuming APIs. However in our experience data.gov is not the best source of data. So if you feel comfortable and would like to embellish your project with better data sources, you may use data from other APIs to augment information from data.gov.

Go to <https://api.data.gov/> and signup for the API key. You will need this key for sending HTTP requests to this API. Next, check out the data sources available as web services at the site and **plan** a simple, but useful website that has at least one master-detail relationship and all CRUD functions based on the api.data.gov data. Limit the active component of your website (i.e. pages that will pull data from the api, and/ or interact with the database) to at most 3 pages. A good site structure is a welcome page, a master page, a detail page, and an about us page. The website should serve some useful purpose to a visitor. Plan your site and create a fully functional static version of the site with hard-coded data. Your website for assignment 5 should be identical to this submission, the only difference being that the submission for assignment 5 should be hosted on Azure, pull live data from the API(s) you choose for this assignment, and use a relational database for persistence. You will lose credit for every significant detail in which your submission for assignment 5 differs from your submission for assignment 3.

Use Github to coordinate your work on the assignment. When completed, upload the website to your MyWeb folder. The upload to MyWeb should contain all the elements needs for the website to work, including html, css, images and JavaScript files.

The assignment is straightforward, and intended to help you practice html, css and JavaScript to get good design results. Please feel free to use look and feel frameworks such as Bootstrap.

Submit the URL to the MyWeb home page (typically index.html) as the assignment deliverable for your group.

# Grading scheme

1. (1 point) data model – plan the data model for the site and include the data model on the about us page as an image. The data model should be normalized, should reflect the data used on the site, and should have at least two one -many relationships. The relationships should be based on site data, not individual user information. For example, favorites are not allowed (a user can favorite several links, and a link is favorited by many).
2. (0.5 points) Content – use of api data, dummy CRUD functions reflecting the data in the data model, relevance of the site for end users
3. (0.5 points) Look and feel – professionalism, consistency and general attention to detail
4. (0.5 points) At least one chart based on the API data (the chart can use dummy data, but it should reflect what you expect to show from the API end-point)
5. (0.5 point) Usability – ease of use, consistency and navigation of your interface for end users
6. (0.5 points) Responsiveness – compatibility with small (mobile), medium (tablet) and large (desktop) screen sizes
7. (1 point) About us page – group member details and contributions, API endpoints used, as well as Github link as specified below
8. (0.5 points) Each student should add a self-reflection comment to your submission, reflecting on your learnings, total time required and any recommendations for future assignments. This should go to the comments section of the assignment, so it is easier to access at a later date.

For ease of grading, after completing features implementation, please add an “About us” page to your site. The page should include a picture and brief information about each group member, and describe each team member’s contribution in brief. Also, include the logical data model for your database as an image, and a link to your GitHub repository.

In addition, include the output from the following command, for all team members. This shows the number of changes made by each Github ID, and is an estimate of each team member’s contribution to the project. To run this command, open the command prompt (CMD), navigate to the folder containing the project repository (use the cd command), and type the command:

git log --author="<Github\_ID>" --pretty=tformat: --numstat

## Starting point

The initial folder structure for the project has already created and pushed on GitHub. You can fork it from this link:

<https://github.com/ISM6225/Assignment_LookAndFeel>

The appropriate mechanism to work on the project as a group would be for one member of the group to

fork the project into their own repository from GitHub, add the other group members as collaborators, and for the other group members to clone the forked group repository.

It is recommended that you fork from this repository and start working on this assignment. It will save your time in structuring this assignment as it is already done for you. Remember, this assignment is not a C# MVC project, but just a collection of html, css and javascript files. Assignment 5 will be your MVC project where you can copy paste the html, css and JavaScript code written here for the corresponding views.

## FAQs

1. What are some common errors seen in this assignment
   1. We see a few common issues in student submissions including
      1. Incomplete data models: the data model should be normalized reflecting as much information from the ADMBS class as possible. Primary and foreign keys should be clearly labeled, and all relationships between tables should be shown as 1-1 or 1-many relationships.
      2. Data models not reflecting API data: we have seen data models for hurricanes and storms that include tables for customers and orders.
      3. Missing ids: we have seen data models that use user first name and last name as a composite primary key. In most cases, a key generated by the database is the appropriate primary key.
   2. Website with no master-detail relationship: the great benefit of web pages is that text can link to other information. In our database driven website, the simplest use case is a master-detail relationship, w.g. clicking on a state can take the user to a page showing relevant information from that state. Please make sure that your site has at least one master-detail relationship.
2. What makes a good submission
   1. A good submission displays a set of related data from data.gov and allows the user to perform standard CRUD functions on the site. It is ok to use the simplest methods and minimal data you are comfortable with, but it is important that the site allow all CRUD operations.

## References

<https://developers.google.com/style/link-text>

<https://webmasters.googleblog.com/2019/09/evolving-nofollow-new-ways-to-identify.html>