P04: Makers Makin' It, Act II -- The Sequel
Design Document

Roster: Stanley Hoo, Tahmim Hassan, Sasha Murokh
Project Name: Submarine Charts
TNPG: RetiredSubmariners
Target Ship Date: 2025-04-23

Overview:
Our site will allow users to display their data from csv/json files on charts or to create ML models based on their data. The format is as follows:

- User uploads dataset, either csv or json.
- User selects either automatic or manual headings (auto will try to detect headings in csv/json files and use those, manual users will enter name for each field).
- User selects bar graph, line graph, or scatter plot, then an independent variable, then a numerical dependent variable to display a graph.
- User selects non-numerical classifying variable, then the variables to train the model to return a machine learning model graph.

Front-end Framework:
We will be using ApexCharts.js for our interactive js and Bootstrap for the css.

APIs:
We will not be using any APIs for this project.

Components

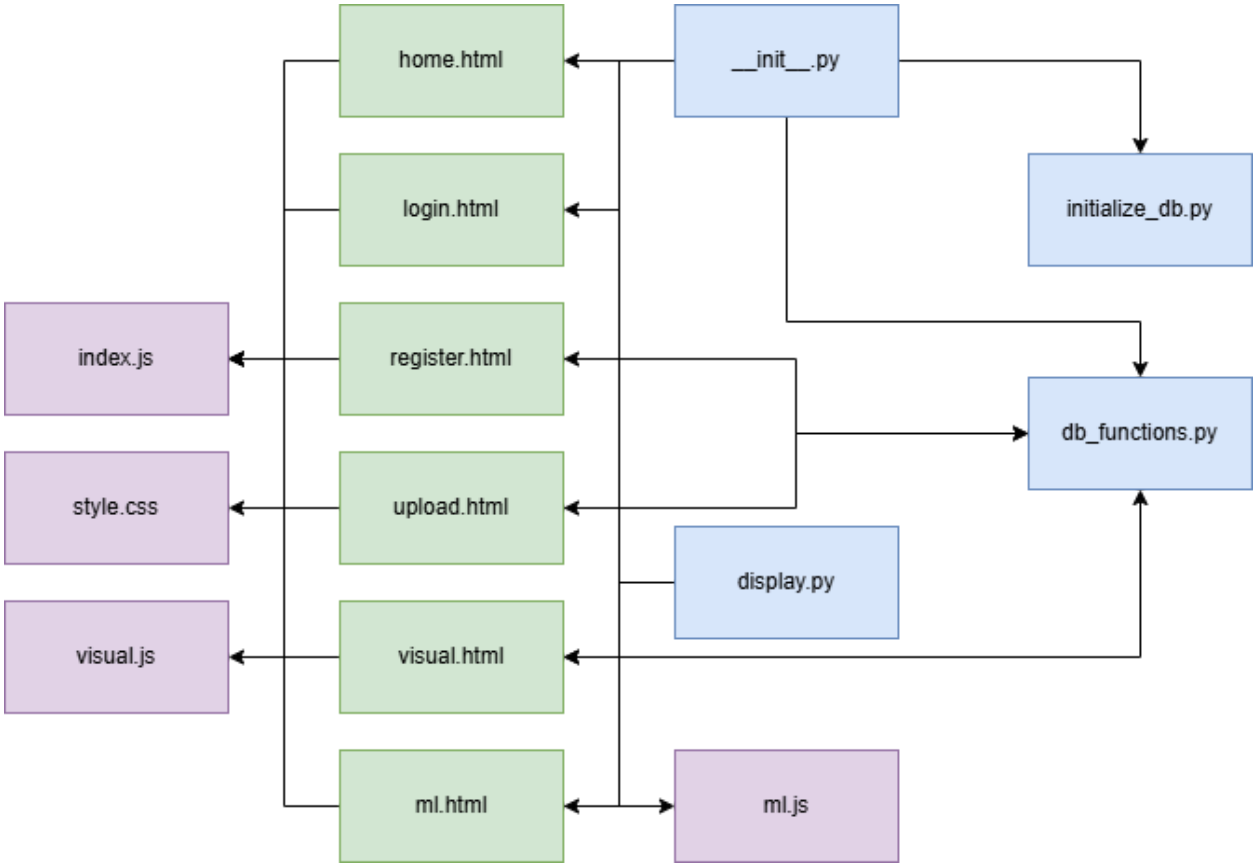
1. **Flask/Python modules:**
    a. __init__.py
        i. All functions for rendering html files
2. **SQLite3 Modules**
    a. db_functions.py
        i. SQL functions
3. **HTML**
    a. home.html
        i. Homepage, displays previously uploaded files and option to upload new file, maybe display some examples

b. login.html
    i. Users are able to log in with username and password, or redirect to create an account if one does not exist. Once user logs in, redirect to dashboard
c. register.html
    i. Users can create an account. Once created, redirect to log in
d. upload.html
    i. Users have an option to upload a file from their computer. Once uploaded, the user selects either manual headings or types in comma separated headings and submit. If there is an issue (number of headings don't match, issue with reading json/csv, etc) flash message and have user retry
e. visual.html
    i. Visual page to view graphs
f. ml.html
    i. ML page to view the ML model

4. Static

a. /css
    i. style.css: Main style for the site
b. /js
    i. index.js: main js for the site
    ii. visual.js: js for the visual site
    iii. ml.js: js for the ml site

Component Map:



Database Structure:
We are using SQLite3, as we only need simple DB functionality

Users: login info and user_id

user_id	**username**	**password**
INTEGER AUTOINCREMENT	TEXT NOT NULL UNIQUE	TEXT NOT NULL

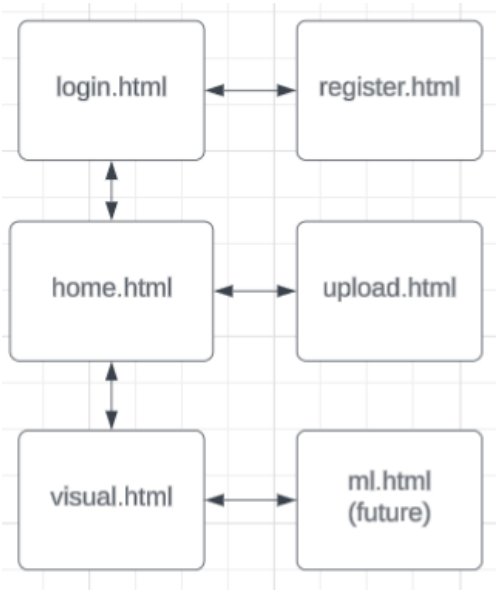
files: file_id, filename, and user_id

file_id	**filename**	**user_id**
INTEGER AUTOINCREMENT	TEXT NOT NULL	INTEGER NOT NULL

file_ {file_id}: stores relevant info for each uploaded file (headers will be TEXT, rows will be INT)

header1	**header2**	**...**
row1	...	...
row2	...	...
...	...	...

Site Map:
Login page
Register page
Dashboard page – shows all previously uploaded files and option to upload file (select from dropdown filetype, choose “auto heading” or manual comma-separated headings) —--select filename to be transferred to visualization page
Visualization page - user can select type of graph, variables to graph (IV and DVs)
ML model, user can select training features, graph is shown



Task Breakdown:

Task	PM Tahmim Hassan	Sasha Murokh	Stanley Hoo
User Auth/Login	✓		✓
Hosting	✓		
File Stuff		✓	✓
DB Lead		✓	
Front End	✓		
Flash error messages			✓
Data Visualization	✓	✓	✓
ML Model		✓	