Mos and Friends: Mohidul Abedin, William Cao, Brian Moses, Alex Thompson
SoftDev1 pd 1 & 2
P04: Let the Data Speak
2020-05-04

Title: Mos and Friends Economic Analysis

Roles:

- Brian Moses - Project Manager, Analysis
- William Cao - Analysis, Backend, Data Management
- Mohidul Abedin - CSS + D3
- Alex Thompson - Front End, CSS, D3

Objective:

Our project is a site where the user can view graphs of US economic data such as GDP growth, treasury yields, or unemployment data over time. Additionally, the site will feature several interactive 'case studies' in important economic events in US history, such as showing the impact of the spike in oil prices on inflation and unemployment during the 1973 recession.

Finally, we will allow users to create their own case studies by combining graphs of their choice with their own analysis. They will then be able to publish their case studies on the site for others to use.

Background:

We have all taken economics and analyzed graphs in class, but they were not interactive and often difficult to see how all the macroeconomic trends complement or contradict each other. With this website, users can walk through examples or form their own conclusions by choosing two economic data sets. Being able to compare graphs is especially important today since we are having greater focus on data and trends.

Outline:

- **Flask**: backend server framework
- **Mongo**: database storage. This is more flexible than SQL databases for storing variable sized lists (case studies can have different amount of graphs and texts)
- **unittest**: run tests for backend. We want to learn how to use this framework, but we will focus on getting the project done first.
- **Bootstrap**: frontend css framework. We are most familiar with this, and it will allow us to use a bootstrap datepicker library.
    - https://www.eyecon.ro/bootstrap-datepicker/

- **D3.js**: frontend javascript framework for creating graphs, data analysis, DOM manipulation, transitions.
- **Data:**
    - Federal Reserve Economic Data https://fred.stlouisfed.org/

Timeline (Due 5/4)

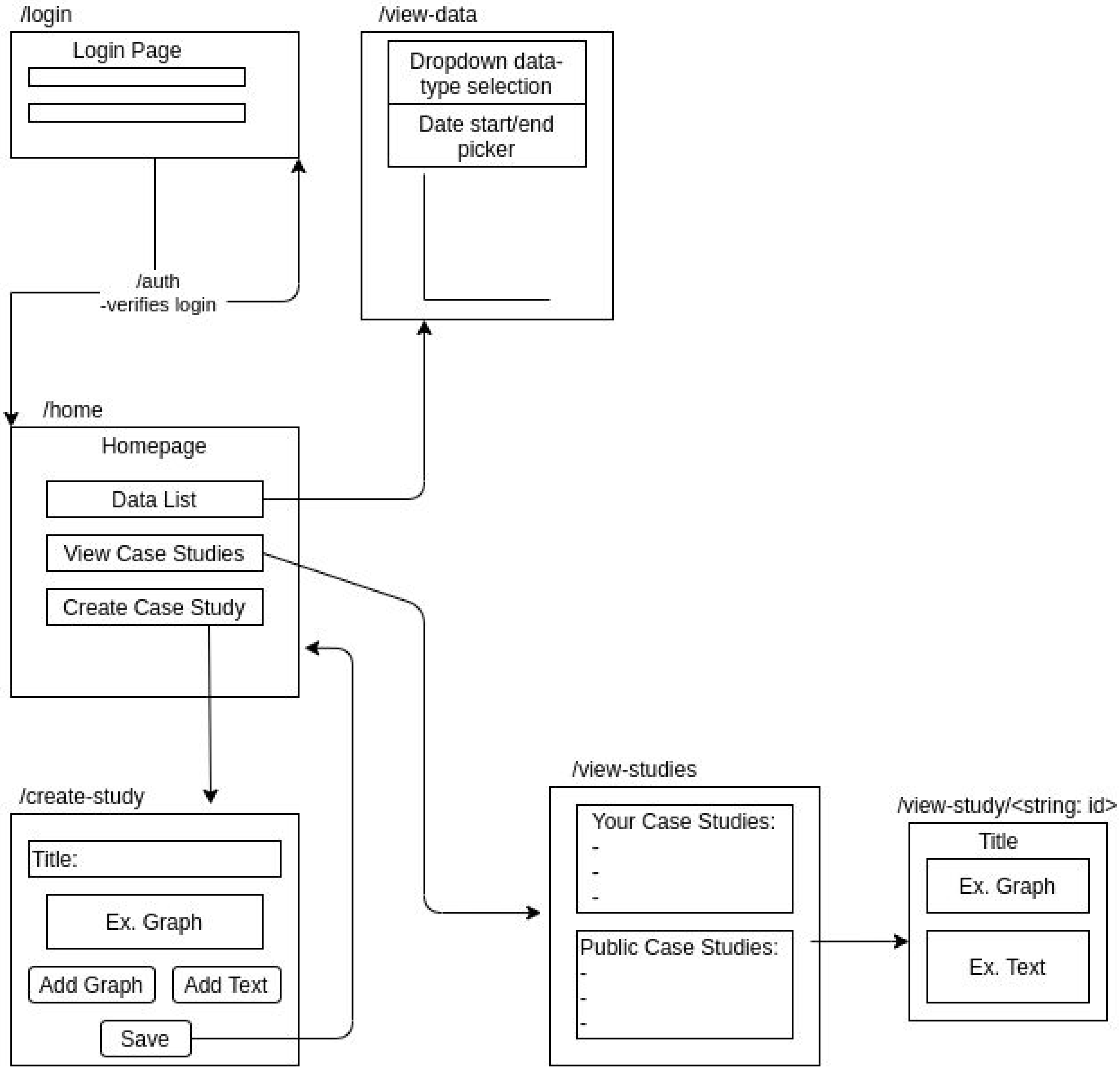
Design Doc - Friday (4/24)
Web routes and line graph - Sunday (4/26)
View studies - Friday (5/1)
Finalize- Saturday & Sunday (5/2-5/3)

Detailed Outline

Program Components/Features:

- Case studies ("adventure")
    - There will be graphs for data involving case studies. For example, for the 1970s recession, there will be oil prices, GDP, unemployment rate.
    - Under each graph will include analysis on the graphs. Answers these questions:
        - What does this show/mean?
        - How does it relate to the other graphs?
    - Conclusion: Should answer these questions:
        - Why should you care about this?
        - How does this relate to today? (Example: Modern oil price dropping)
- Comparison of data
    - User will select two dataset and view graphs side by side

Sitemap



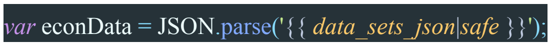
Session

- "username": Used to check if the user is logged in and what the username is

Routing

- YELLOW = app.py only has render_template() nothing else
- GREEN = app.py is finish
- RED = does not exist
- Naming scheme:
  - "title" -- what the user sees
  - "name" -- the name of the csv file without the .csv

Webpage Routes (All GET)

- /login
    - Only guests can use (redirect to previous route if logged in)
    - Variables needed:
        - None
    - HTML file: templates/login.html
- /home
    - Anyone can use
    - Variables needed:
        - None
    - HTML file: templates/home.html
- /view-data
    - No login required
    - Variables needed:
        - data_sets
            - List of dictionaries. Dictionary keys/value: “title”/string, “routing_name”/string, “start_date”/string, “end_date”/string, “units”/string
        - data_sets_json
            - JSON representation of “data_sets”
            - Loaded into javascript variable “econData”
                - 

                  ```
                  var econData = JSON.parse('{{ data_sets_json|safe }}');
                  ```
    - HTML file: templates/view-data.html
- /view-studies
    - Anyone can use
    - Variables needed:
        - case_studies
            - List of dictionaries, each dictionary like this:

```
{
        Title: “title”,
        Description: “random text”,
        *username: username,
        Content: [
                {
                Type: “chart”| “text”
                Chart-start: “YYYY-MM-01”,
                Chart-end: “YYYY-MM-01”,
                Chart-name: string,
                Text: text
                },
            …
            ]
}
```

    - HTML file: templates/
- /view-study/<string:id>
    - Anyone can use
    - Variables needed:
        - case study

```
{
        Title: “title”,
        Description: “random text”,
        username: username,
        Content: [
                {
                Type: “chart”| “text”
                Chart-start: “YYYY-MM-01”,
                Chart-end: “YYYY-MM-01”,
                Chart-name: string,
                Text: text
                },
            …
            ]
}
```

    - HTML file: templates/
- /create-study
    - Logged in user
    - Variables needed:
        - econ_data
            - List of dictionaries. Dictionary keys/value: “title”/string, “routing_name”/string, “start_date”/string, “end_date”/string, “units”/string

- /create-account
    - Only guests can use (redirect to previous route if logged in)

API Routes

- POST: /logout
    - Only logged in user can use
    - Sends to server:
        - Nothing
    - Response:
        - Nothing
- POST: /create-account
    - Only guests can use
    - Sends to server:

```
    {
       username: string,
       password: string
    }
    ```
    - Response:
        - If error:
            - HTTP response code: **400**
            - Flash: error: string
        - If successfully created:
            - HTTP response code: **201**
            - No JSON response
- POST: /login
    - Only guests can use
    - Sends to server:
      ```
      FormData: {
              username: string,
              password: string
      }
      ```
    - Response:
        - If error:
            - Flash error
        - If successful:
            - Redirect to /home
- GET: /date start<data>/<<>to<date end>>
    - Anyone can use
    - <data>: “oil-prices”, “gdp”, “cpi”
    - <<date start>to<date end>>:
        - Inclusive both dates
    - Example: /gdp/1970-01-13to1980-01-13
        - Gets data from Jan 13, 1970 to Jan 13, 1980 (Inclusive both)
    - Response:
        - If successful:
            - HTTP response code: **200**
            - Response:
              ```
              {
                      values: [
                              {
                                      date: "YYYY-MM-DD",
                                      value: <value>
                              }
                      ]
              }
              ```

- POST /create-case-study
    - Logged in user only
    - Sending:

```
{
        Title: “title”,
        Description: “random text”,
        *username: username,
        Content: [
                {
                        Type: “chart”| “text”
                        Chart_start: “YYYY-MM-01”,
                        Chart_end: “YYYY-MM-01”,
                        Chart_name: string,
                        Text: text
                },
        …
        ]
}
```