Mos and Friends: Mohidul Abedin, William Cao, Brian Moses, Alex Thompson
SoftDev1 pd 1 & 2
P04: Let the Data Speak
2020-05-11

Title: Mos and Friends Economic Analysis

Roles:

- Brian Moses:
    - Project Manager
    - Analysis -> display ‘case studies’ (explained below) for user to see
- William Cao:
    - Backend -> create routing for __init__.py and API routes when necessary
- Mohidul Abedin:
    - CSS and D3 -> allow user to view any dataset they select
- Alex Thompson:
    - CSS and D3 -> create page for allowing user to create ‘case studies’

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 5/11)

One file for drawing graph (Mos) - 4/28
Delete case study frontend (Alex) 4/28
Show flashes (Alex) - 4/29
Navbar (Alex) - 4/28
Line drawing animation (Mos) - 4/29
Multiple datasets on same graph (Mos) - 4/30
Add x, y-axis labels (Mos) - 4/29
More csv's (Brian) - 4/28
Fix negative in graph (Mos) - 4/29
Scroll animation

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:

- Compare data
    - User can select datasets and years during which to view the data
        - datasets are selected from a list that we provide
    - User can choose to select two different datasets and view correlation between data
- View case studies
    - Users can view several **case studies** in economic data. A case study is an explanation of an event using graphs and text. For example, for the 1970s recession, there would be graphs of oil prices, GDP, and unemployment rate, and there woulod be textual analysis explaining how they relate to eachother
- User will also be able to create their own case studies, choosing datasets to display graphically and creating text to analyze graphs

- user can select all or part of the dataset
- other users will be able to see their case studies

MongoDB

- Database name: mo_and_friends_data
  - Collection: login

```
{
    _id: ObjectId
    username: string,
    password: string
}
```

  - Collection: data

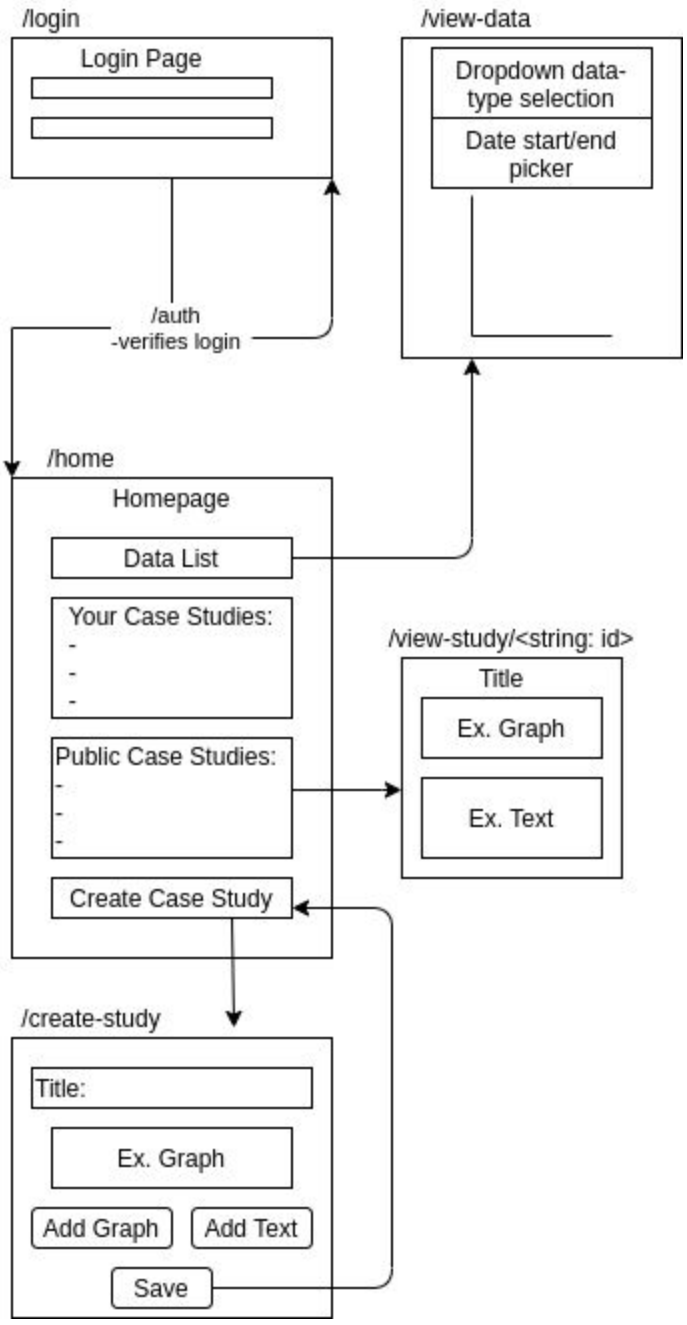
```
{
    name: string,  // name of csv file
    start_date: "YYYY-MM-DD", // earliest date of csv file
    end_date: "YYYY-MM-DD", // latest date of csv file
    // first element will always be date
    column_names: [string, string] // name of columns in csv
    data: [
        // each object is one line of csv file
        {
          date: "YYYY-MM-DD",
          value: number
        },
        {
          date: "YYYY-MM-DD",
          value: number
        },
        ...
      ]
    }
  }
```

  - Collection: case_studies

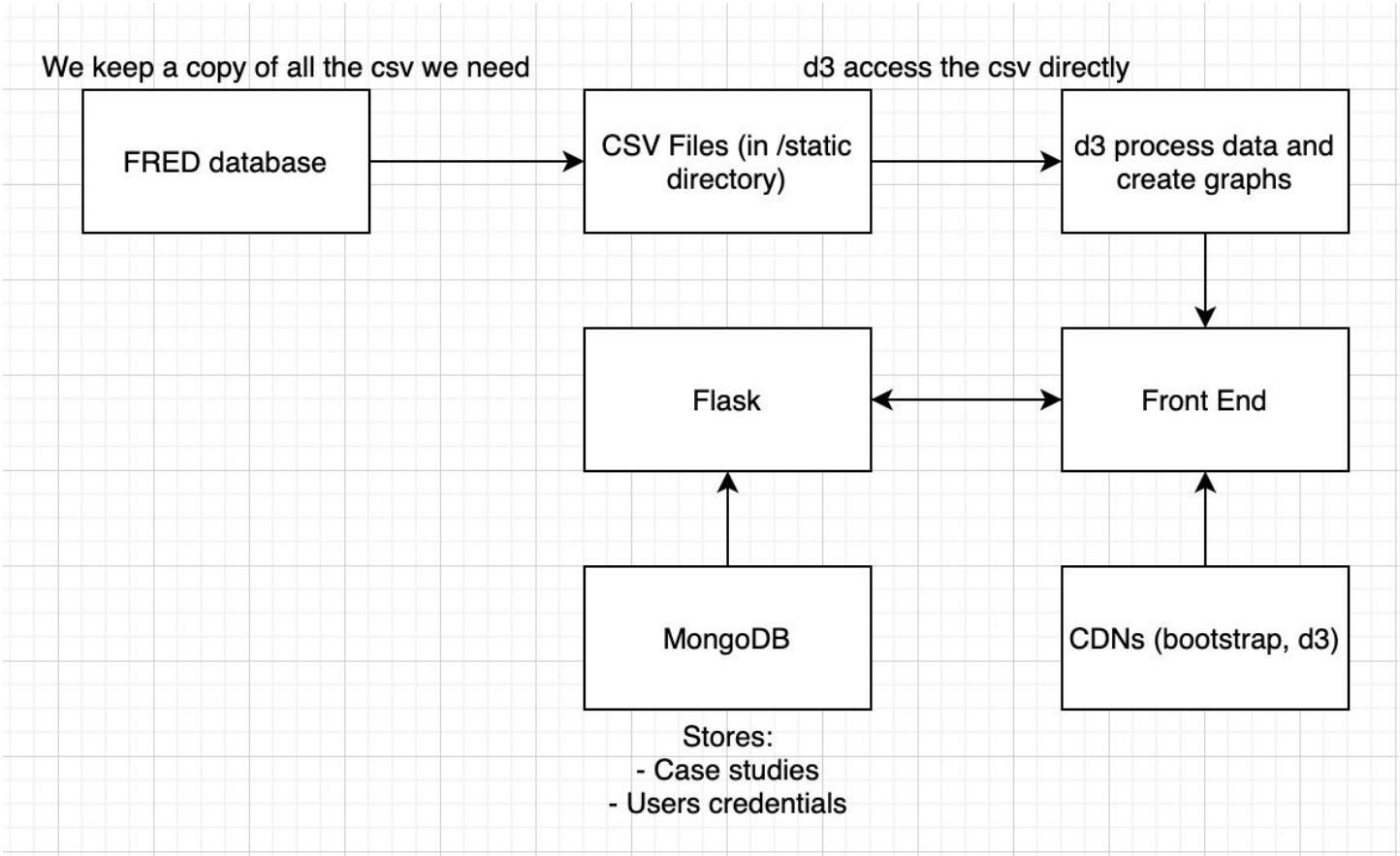
```
{
    _id: ObjectId,
    username: string, // user who created the case study
    title: string,
    description: string,
    content: [
      {
```

```
        type: string, // either "chart" or "text"
        // if type is "chart", these fields exist
        chart_start: string,
        chart_end: string,
        chart_name: string, // routing name
        // if type is "text", these fields exist
        text: string
      },
      ...
    ]
}
```

Sitemap


/login
Login Page
/view-data
Dropdown data-type selection
Date start/end picker
/auth
-verifies login
/home
Homepage
Data List
Your Case Studies:
-
-
-
/view-study/<string: id>
Title
Ex. Graph
Public Case Studies:
-
-
-
Ex. Text
Create Case Study
/create-study
Title:
Ex. Graph
Add Graph
Add Text
Save

Component Map



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
  - HTML file: templates/view-data.html

- /view-studies DROPPED NO LONGER USED
  - Anyone can use
  - Variables needed:
    - List of dictionaries. Dictionary keys/value: “title”/string, “routing_name”/string, “start_date”/string, “end_date”/string, “units”/string
  - HTML file: templates/

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

```
{
    Title: “title”,
    Description: “random text”,
    username: username,
    Content: [
        {
        type: string, // either "chart" | "text"

        // if type is "chart", these fields exist
        chart_start: string,
        chart_end: string,
        chart_name: string, // csv file name without the .csv

        // null values if second dataset is not selected. optional to fill out
        chart_start_2: string,
        chart_end_2: string,
        chart_name_2: string, // csv file name without the .csv

         // if type is "text", these fields exist
        text: string
        }
        …
        ]
}
```

- HTML file: templates/view-study.html

- /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
  - HTML file: templates/create-study.html

- /create-account
  - Only guests can use (redirect to previous route if logged in)
  - HTML file: templates/create-study.html

API Routes

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

    {
    username: string,
    password: string
    }
  - Response:
    - If error:
      - Flash: error: string
    - If successfully created:
      - Redirect to /home
- POST: /login
  - Only guests can use
  - Sends to server:

    FormData: {
    username: string,
    password: string
    }
  - Response:
    - If error:
      - Flash error
    - If successful:

- Redirect to /home

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

```
{
    title: “title”,
    description: “random text”,
    content: [
        {
            type: string, // either "chart" | "text"

            // if type is "chart", these fields exist
            chart_start: string,
            chart_end: string,
            chart_name: string, // csv file name without the .csv

            // null values if second dataset is not selected. optional to fill out
            chart_start_2: string,
            chart_end_2: string,
            chart_name_2: string, // csv file name without the .csv

            // if type is "text", these fields exist
            text: string
        }
        …
        ]
}
```

  - Server responds with:

```
{
    “redirect”: <redirect route>
}
```

- DELETE /delete-study/<string:id>
  - Deleted case study must have username same as one logged in. Make sure to redirect to /view-studies after making this request
  - Send: None
  - Receive: None
- POST /update-study/<string:id>
  - Username logged in must be same as case study username
  - Send: The ENTIRE case study with updated values

```
{
    title: “title”,
    description: “random text”,
    username: username,
```

content: [

{

type: string, // either "chart" | "text"

// if type is "chart", these fields exist

chart_start: string,

chart_end: string,

chart_name: string, // csv file name without the .csv

// null values if second dataset is not selected. optional to fill out

chart_start_2: string,

chart_end_2: string,

chart_name_2: string, // csv file name without the .csv

// if type is "text", these fields exist

text: string

}

…

]

}

- Receive: Should refresh the page