Mountain Goats Project: Medical Procedure Price Checker

Nick Mavromatis, Patrick Chesnut, Cooper Ide

Github

- Overall, an excellent tool for version control and collaboration
- Industry Standard
- Easy to use to collaborate and track and update changes
- Slight learning curve-add, commit, pull, issues
- Easy to use both in UI and command line
- Challenges: Merge Conflicts!
- Solution: Followed advice in error message, such as `git merge --abort`

Python and Libraries

- Easy language to use with many useful libraries
- Relatively easy to integrate with Flask and to debug
- Used Jupyterlab notebooks and .py script files
- Libraries:
  - **Pandas:** Conversion from JSON to excel, then from excel to a database file. Easy to read each sheet of the excel file and turn that into SQL tables.
  - **SQLite3:** Lightweight and easy to use SQL library
  - **flask_wtf:** Defining forms in python scripts with useful built in class (I.E. FlaskForm)
  - **wtforms:** For flexible webform rendering, used with flask_wtf (I.E. StringField)
  - **flask_login:** provides user session management
- Challenges: The learning curve for flask integration and form creation was somewhat high.

SQLite

- Easy to install, use, and integrate
- Convenient to create a local database file
- Lightweight without many complicated features
- Databases are easy to use plain files
- Automatically avoids issues of corruption
- Great for low traffic websites, but doesn’t scale well (fine for our limited project)

Flask

- A micro web framework written in Python
  - Micro: the core is simple, but extensible
  - 3rd party extensions available for form validation, upload handling, authentication
- Request handling performed by functions decorated by @app.route(/link)
  - Functions can return strings that are automatically converted to html
  - Return Python dictionaries that are automatically converted JSON, useful for dynamically updating webpages
  - Dynamic paths can be created using angle brackets in the route (e.g. /hello/<name>)
- Built in support for Jinja, a templating engine, primarily used for creating HTML files
  - Evaluate Python expressions using double curly braces
  - Use curly braces with % signs to create blocks, useful for inheriting headers, footers, navbars
- Easily integrate sqlite3 queries and return JSON or dynamic HTML
- Challenges: Slight learning curve, especially for dynamic table creation
- Challenges: Hard to get up and running for students new to web development
- Limitations: A more comprehensive web framework would have built in authentication tools
- Evaluation: Great choice for a first backend and simple projects

Bootstrap

- Bootstrap is a CSS framework that provides responsive and aesthetically pleasing web page design
- Advantages
  - Quickly create web pages that look nice without having to create your own stylesheets
  - Can load from CDN eliminating the need to store it on your own web server
  - If users visit another website that has used Bootstrap, it is likely to already be cached on their web browser, speeding up your website's load time
  - Very extensive documentation with code examples, allowing you to simply copy and paste components that you want on your website
  - Mobile first design philosophy, code is optimized for phones/tablets and scaled up for use on computers
- Disadvantages
  - Not as customizable as creating unique stylesheets, web pages will look nice but generic
  - Your website will likely not use all of Bootstrap's components, but CSS and Javascript files will include code for all components, resulting in slower performance (Tailwind alternative)
  - Mobile design is evident on laptops/desktops
- Evaluation: Excellent for those not artistically inclined

Javascript

- A client-side scripting language supported by web browsers to make web pages dynamic
- Easily traverse the DOM using getElementById or getElementsByClassName
- Add event listeners to HTML elements like buttons and input fields
- Can reduce the number of requests to the web server by shifting compute to the client
- Can update a webpage with information from the web server by using fetch(url, data_to_send).then(function(response) {commands to update HTML})
- Native JS is verbose, jQuery library could be used to simplify
    - Example: document.getElementById(“target”) vs $(“target”)
- Evaluation: Great for making interactive web pages, TypeScript may be newer alternative, but regular JS works just fine

Trello

- Simple to use drag and drop interface
- However, we didn’t check or update this much
- Mostly, we relied on less formal to do lists during meetings
- Could easily be used to create burn down or burn up charts
- Would work well for teams that do not meet at scheduled times
- Evaluation: Not all that useful for our team. We preferred to assign tasks and update the team during our Monday Zoom meetings. Would be a good option for teams that do not meet.

Heroku and Render

- Heroku was harder to implement (needed to make specific procfile)
- Issue: Service is no longer free
- Render was an easy to use alternative
- Render updates automatically from github
- Unlike heroku, which needed command line commands, Render is used entirely in a UI so is simpler

Challenges/Solutions

- Coordinating tasks as a group with different schedules in different time zones
  - Stuck to a standard meeting time, WEEKLY_STATUS.md used to set goals/tasks
- Learning curves-Especially Flask
  - The labs offered good experience to overcome the learning curves and add more complex features
- Getting flask to run locally on different instances of JupyterHub
  - Carefully reviewed lab 10 code, utilize url_for function to complete first part of url
- Positioning elements where needed using Bootstrap
  - Really digging into the documentation, YouTube tutorials
- Heroku discontinuing their free tier
  - render proved to be a great alternative, auto deploys directly from GitHub
- Getting data from the web server and displaying it on a web page without reloading the entire page
  - Discovering the JS fetch function with the help of web tutorials
- The enormity of creating a website using new technologies
  - We used AGILE effectively to break the project up into small sprints and adjust on the fly as challenges arose
- Short time frame
  - We opted to use the simplest libraries to expedite development

How plans changed

- In general, we had to simplify the project as learning new tools and building a website proved more difficult than we first thought.
- Scaled back the number of hospitals and procedures in the database so we could focus on building a website rather than gathering data
- Did not implement a Google Map showing locations of hospitals

Demonstration Video