

Project Description:

Project Purpose:

The tool is a web application designed to help users track their productive hours while using YouTube. This app enables users to set goals and keep track of their progress over time (days, weeks, months). It also allows users to edit their daily goal at any moment. Hoping to encourage people to slowly increase their benchmark goal until they are logging hours they once never believed was possible. One key goal of this project was to seamlessly embed it into the existing youtube UI. After some thought, I decided to include a timer icon into the top nav bar that Youtube already displays on their site. Another aspect of the app worth mentioning is that I want to gamify the process of hitting goals. Similar to how companies like Fitbit do it. As I set up the backend, I will be able to display user data in fun and interactive ways.

The intended audience is users of youtube that want to limit their unproductive video watching. There is a way to use youtube to increase efficiency while working, and this app is meant to help people find that rhythm. This tool will not be useful to someone that does not care what kind of content they are watching. That is why only I put the button to access the tool in a button in the nav bar. I didn't want to hijack the entire UI of Youtube.

Project Files

HTML:

1. Index.Html
2. Home.html
3. Person.html
4. Stopwatch.html
5. newDay.html

6. Calander.html

CSS:

7. Index.css
8. Person.css
9. Stopwatch.css
10. newDay.css
11. Calander.css

Image Files:

1. Cal.png (calendar icon)
2. Person.png (account icon)
3. Plus.png (new day icon)
4. Stopwatch.png (Clock icon)
5. Ytlog.png (youtube logo)
6. Play.png (video thumbnail)
7. Trash.png (delete icon)

Database related files

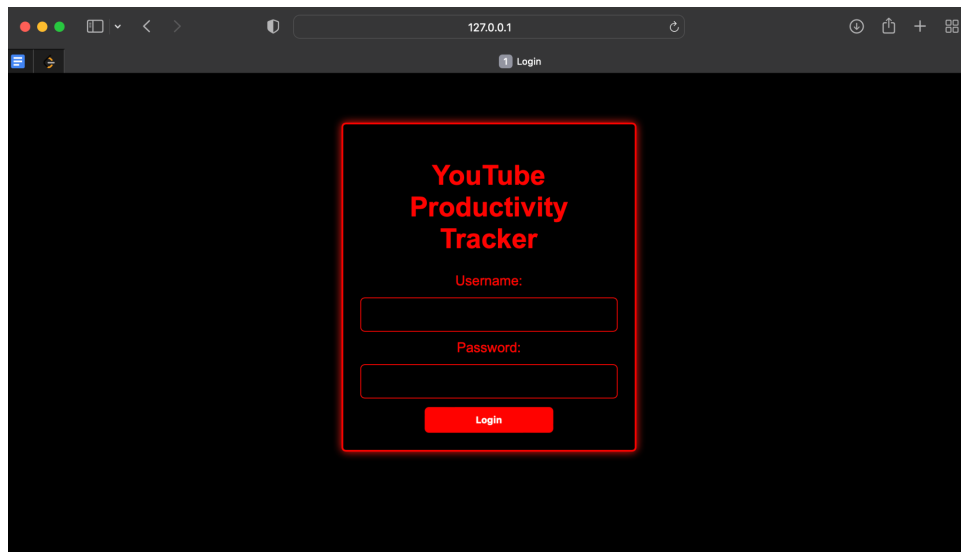
1. populate_calendar_entries.py

Important Django Files

1. Models.py
2. Views.py
3. Urls.py

App Screenshots:

User Log in Page (POST)



List View Page (GET)

STOPWATCH HISTORY

 [Nov. 1, 2023, Total: 7:55:00](#)

 [Nov. 2, 2023, Total: 8:30:00](#)

 [Nov. 1, 2023, Total: 7:55:00](#)

 [Nov. 2, 2023, Total: 8:30:00](#)

 [Nov. 3, 2023, Total: 7:45:00](#)

 [Nov. 4, 2023, Total: 8:15:00](#)

 [Nov. 5, 2023, Total: 9:30:00](#)

 [Nov. 6, 2023, Total: 7:20:00](#)

 [Nov. 7, 2023, Total: 8:50:00](#)

 [Nov. 8, 2023, Total: 6:10:00](#)

 [Nov. 9, 2023, Total: 10:05:00](#)

 [Nov. 10, 2023, Total: 7:40:00](#)

Python Database Page

```
class Command(BaseCommand):
    help = 'Populate CalendarEntry records'

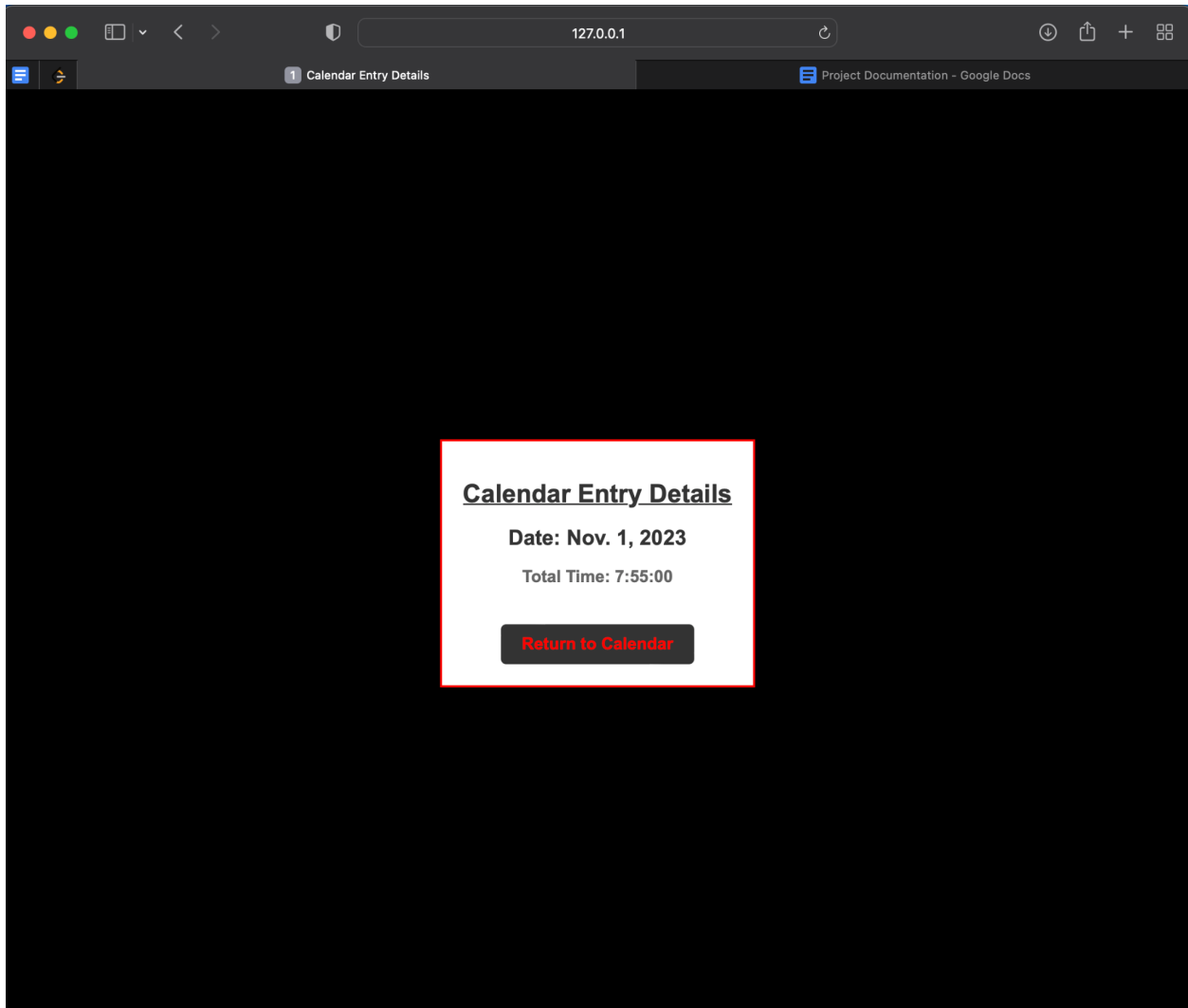
    def handle(self, *args, **kwargs):
        # Create 10 entries with different dates and times
        entry1 = CalendarEntry(date='2023-11-01', total_time=timedelta(hours=7, minutes=55))
        entry1.save()

        entry2 = CalendarEntry(date='2023-11-02', total_time=timedelta(hours=8, minutes=30))
        entry2.save()

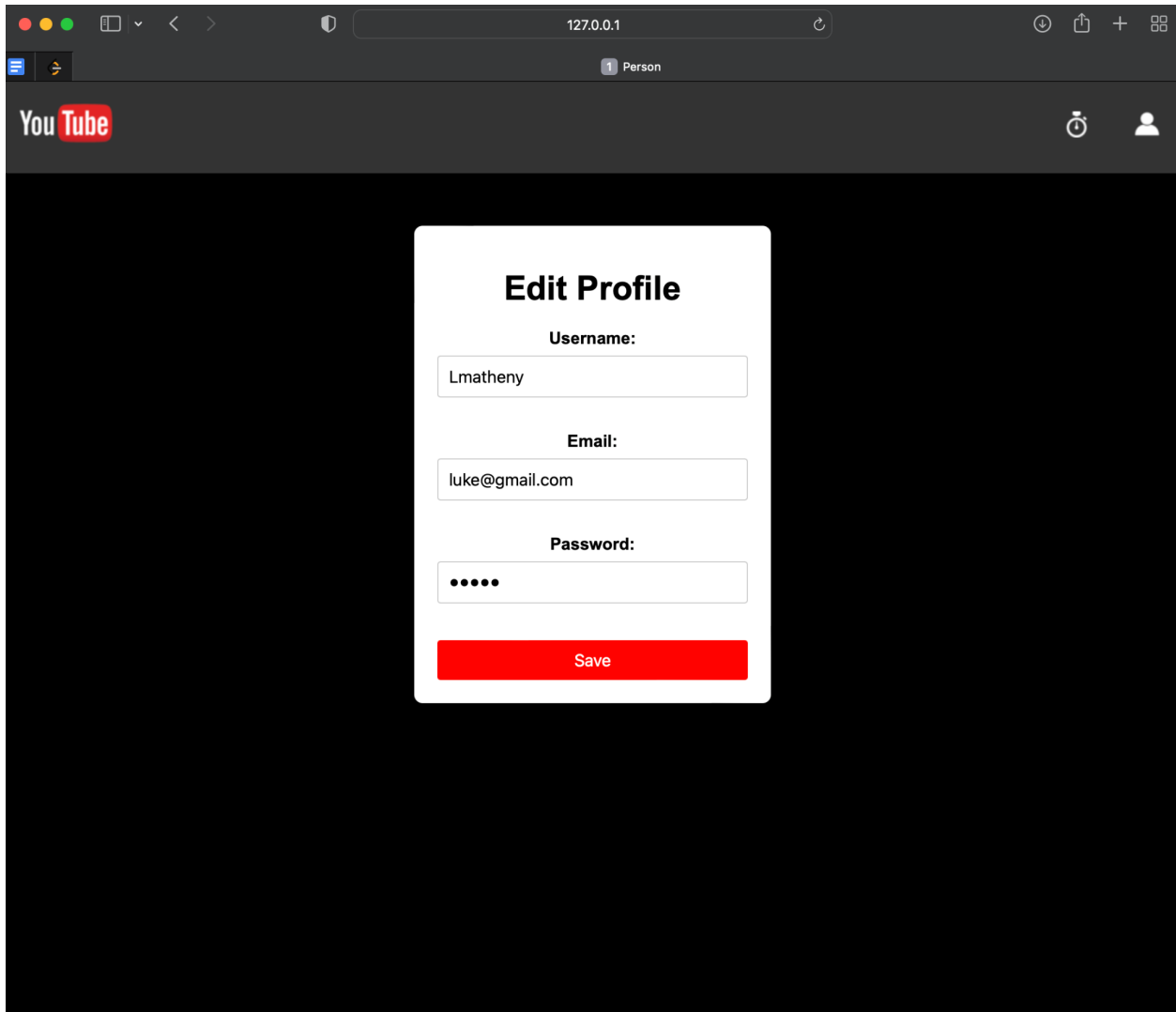
        entry3 = CalendarEntry(date='2023-11-03', total_time=timedelta(hours=7, minutes=45))
        entry3.save()

        entry4 = CalendarEntry(date='2023-11-04', total_time=timedelta(hours=8, minutes=15))
        entry4.save()
```

Single Entry Details Page (GET)



Edit Details Page (GET and POST)



The screenshot shows a web browser window with a dark theme. The address bar displays '127.0.0.1'. The page header features the 'YouTube' logo on the left and a user profile icon on the right. The main content area is a dark gray rectangle. Centered within this area is a white modal box titled 'Edit Profile'. The form inside the modal has three input fields: 'Username' with the value 'Lmatheny', 'Email' with the value 'luke@gmail.com', and 'Password' with five dots. A red 'Save' button is positioned at the bottom of the modal.

Edit Profile

Username:

Lmatheny

Email:

luke@gmail.com

Password:

.....

Save

New Entry Page (POST)

127.0.0.1

1 New Day

You

Tube

START A NEW DAY

Date:

11/10/2023

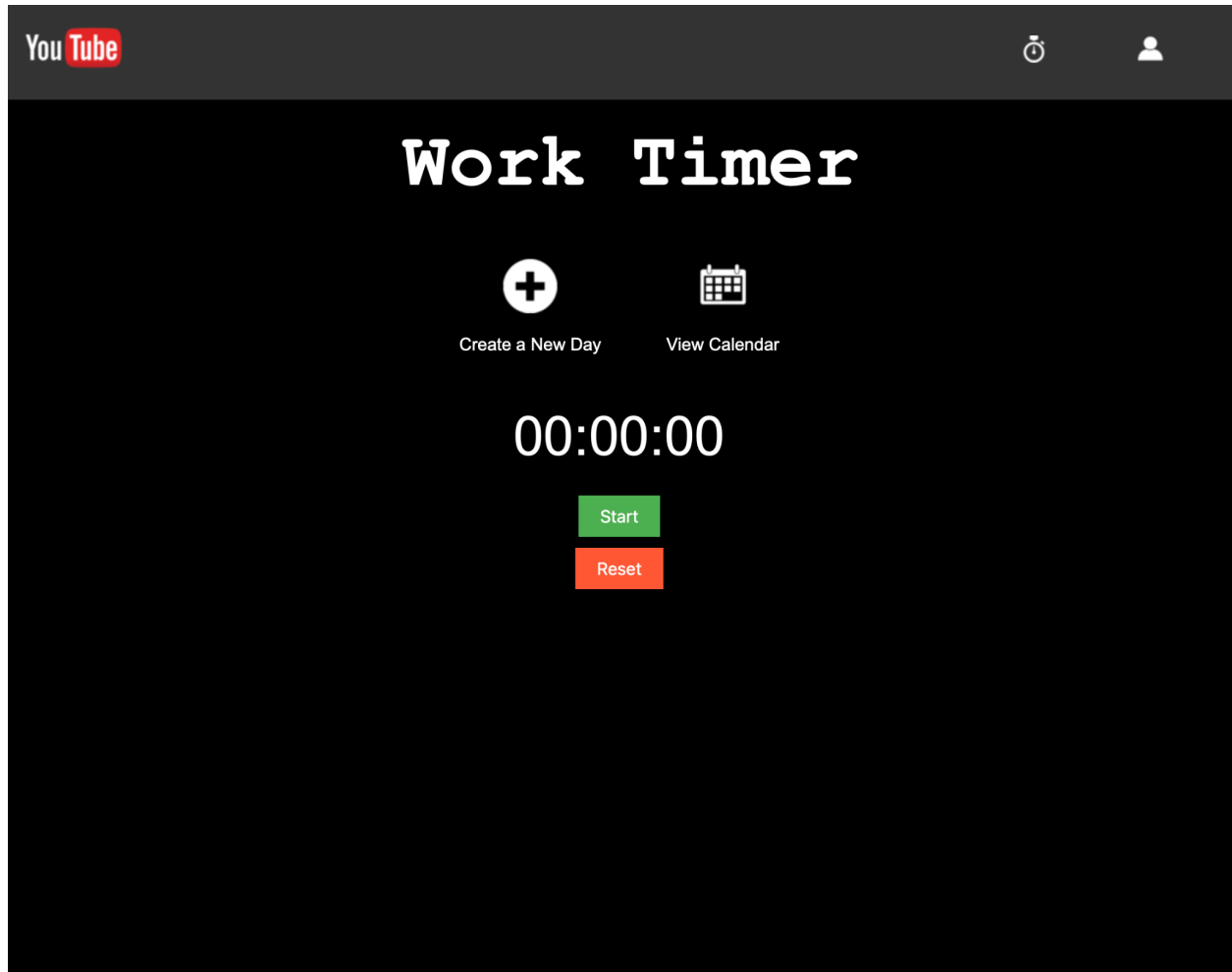
Time Goal for the Day:

13 hours

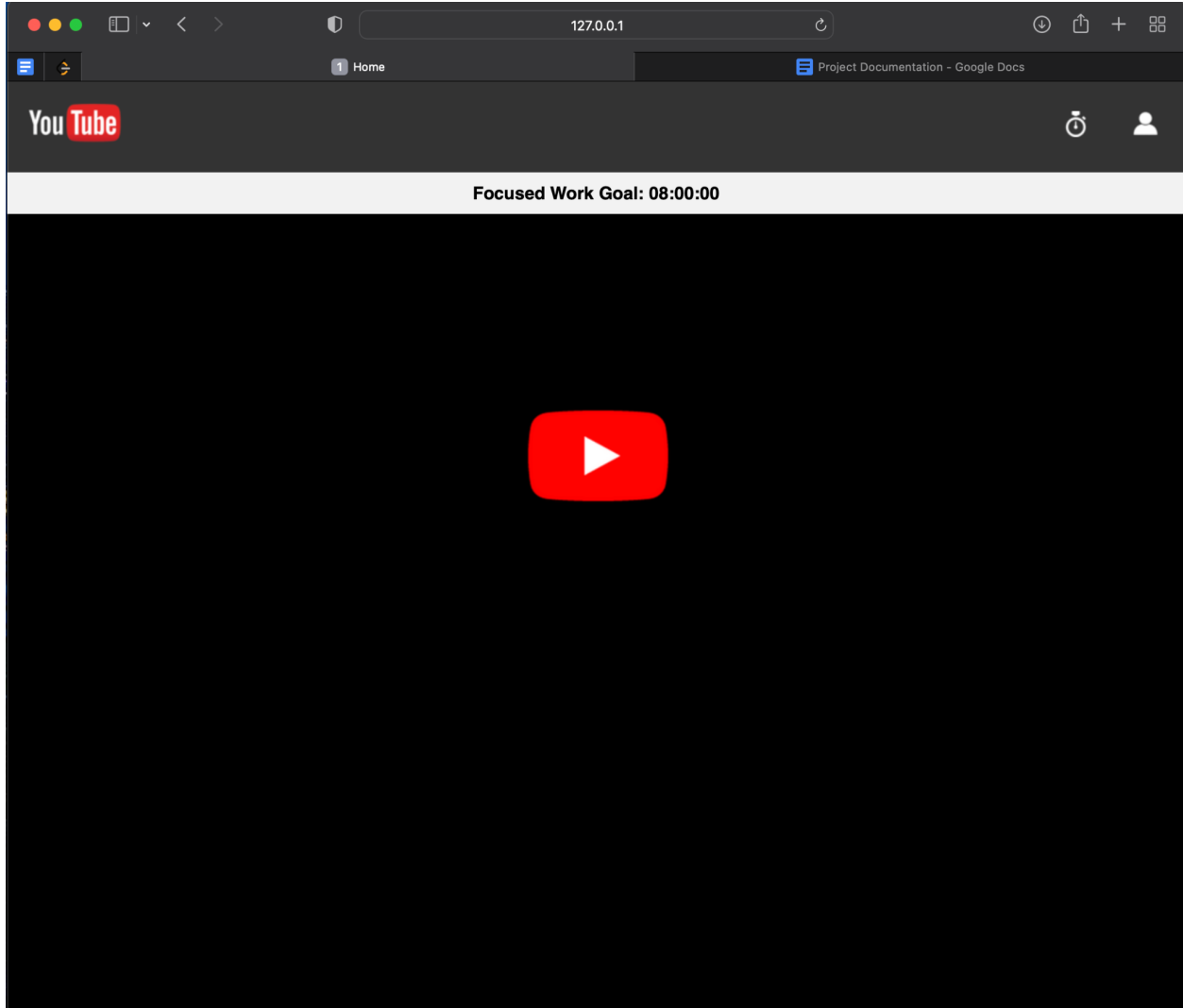
22 minutes

Create

Timer Page



Home Page



Installation steps:

In order to run this project using http protocol, these steps must be followed precisely.

1. Open Visual Studio Code: If you haven't already, download and install Visual Studio Code.

2. Clone the Repository:

- Open Visual Studio Code.
- Click on the "Source Control" icon.
- Click "Clone Repository."
- Enter the repository URL, e.g., <https://github.com/lmatheny/581-M3.git>, and choose a local directory.
- Click "Clone."

3. Create a Virtual Environment (Optional):

- Navigate to the project directory: `cd 581`
- Create a virtual environment (if you don't already have one): `python -m venv venv`

4. Activate the virtual environment:

On Windows: `venv\Scripts\activate`

On macOS and Linux: `source venv/bin/activate`

5. Install the required Python packages: `requirements.txt`

6. Set up the database by running migrations:

```
python manage.py makemigrations
```

```
python manage.py migrate
```

7. Run the Development Server:

- Start the Django development server: `python manage.py runserver`
- The server will run locally at `'http://127.0.0.1:8000/'`.

8. Access the Application:

- Open a web browser and go to `'http://127.0.0.1:8000/'` or the address shown in your terminal.

External Sources:

1. *The all-in-one workspace for your notes, tasks, wikis, and databases.* Notion. (n.d.). <https://blushing-galleon-536.notion.site/Django-Install-adab2ef4a25e438e8ea5845a4ced3812>
2. MozDevNet. (n.d.). *Django tutorial part 3: Using models - learn web development: MDN.* Learn web development | MDN. <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Models>
3. *Chatgpt.* ChatGPT. (n.d.). <https://openai.com/chatgpt>
4. Lawrence Williams, & Williams, L. (2023, September 19). *Get vs. post: Key difference between HTTP methods.* Guru99. <https://www.guru99.com/difference-get-post-http.html>
5. Csaba. (2021, February 9). *Edit Post data before or after save.* Django Forum. <https://forum.djangoproject.com/t/edit-post-data-before-or-after-save/6507>