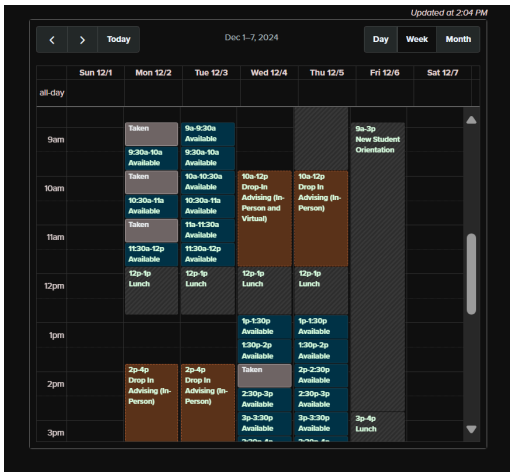
Scheduling + Routine Tracker

Group members: Leah Tesfaye, Amanda Jung, Jesus A Ramirez Quintana,
Bushra Mohammed A Hakami

Overview

The goal of this project is to create an online scheduling/routine tracker which allows users to post their routines, daily schedules, track progress, and share availability with others. Users can interact with each other's schedules by leaving comments or booking time slots marked as available. The primary features of the site may include:

- Schedule Management: Users would be able to create, edit, and delete events on their calendar(jesus)
- Social Interaction: Users can view other people's public schedule, comment on them, and request to book a time slot if it's marked as available for booking
- Messaging: Users have a pop up window where they can send messages to one another(bushra)
- To-do list: Users can organize their tasks in a checklist format. They would be able to add, delete and edit tasks(leah)
- Timer: Users have the ability to set a timer when working on tasks/projects.(leah)



Frontend

Important Pages

- The Homepage
    - A page that is an overview of all the site's features
    - Login and signup buttons
- The Dashboard
    - Displays the user's schedule
    - Allows the user to add, edit, or delete schedules
    - Displays the user's to-do list
    - Allows the user to add, edit, or remove tasks from the to-do list
    - Displays timer the user can set for working on tasks
- The Social
    - Shows the schedules of other users, with options to comment and book time slots
    - Allows the user to message other users directly by clicking on a button in the corner of the page, creating a pop-up window for messaging
- The Help Page
    - FAQ's or contact information in case of bugs
- Settings
    - Font size, theme, etc.

Notes

- CSS Grid/Flexbox
- Dynamic updates to DOM, client side Javascript
- API calls/updates data without reloading the page

Backend

- Uses Node.js, connecting to a MongoDB database
    - Express
    - Mongoose will be used to define the structure of data stored in MongoDB
- Routes
    - User Authentication
        - Signup, Login
    - Task management
        - Schedules, bookings, Status

API Ideas for routes

Method	Route	Function
GET	/api/events	Gets user public schedules for all users?
GET	/api/events/id	Get a specific user's schedule
POST	/api/events/id	Update booking status for a specific user
GET	/api/status/id	Gets booking status for a specific user(in progress, done)
GET	/api/list/id	Get a specific user's to-do list
POST	/api/list/id	Modify to-do list for a specific user
GET	/api/timer	Get status of timer(time remaining)
POST	/api/timer	Set timer duration

Reflection

We maintained the majority of our original project spec, implementing the to-do list, timer, and schedules function. One feature that was left out of our final project was the direct messaging feature. We initially planned to have it so users could message each other directly. Due to the extra time it took us to finish the main features of the application(todo, timer, calendar), we thought it was best to exclude it from the final project.

Setting up the frontend portion of the project was easier for us, as each group member was responsible for a specific feature. Initially, we thought the frontend would be an issue because we had quite a few features. However, assigning everyone individual sections helped to break up the work into manageable portions.

There were a couple issues we had while working on the project, one of which was the backend. We were unsure of whether or not to consolidate the content(routes, models, etc.) into one file or create individual folders and files. Another setback during the project was implementing DigitalOcean to deploy our application. None of us had much prior experience with the platform, so there was a learning curve initially.