Team Wesley - IEC 2024 Programming Report

Jeremy Mark Tubongbanua, Jerry Shum, Nehmat Farooq, Wesley Kyle De Guzman

Stakeholders and Problem

Stakeholders: The primary stakeholders for this project include business owners, managers, and employees of remote working businesses. These are crucial as they are the users who directly interact with the task management system.

Problem: The problem we must tackle is to effectively assign tasks that balance workloads, respect employees' preferred working hours, and maximize productivity.

Relevance to Stakeholders: This problem is significant to stakeholders because:

- **Managers -** need a streamlined and simple process for task assignment to avoid overloading or under-utilizing employees.
- **Employees -** benefit from balanced workloads and respect for their preferred working hours, which can enhance job satisfaction and productivity.
- **Business Owners -** are concerned with overall productivity and efficiency, which heavily impacts the company's performance.

Chosen Solution

The solution is a React web application named Taskly, a simple task management tool for managers and employees. It facilitates the creation, assignment, and viewing of tasks, which are reflected on a calendar.

How It Addresses the Problem:

Taskly addresses the problem of effectively assigning tasks to employees by introducing an intermediary strategy where we receive the requirements from the manager and translate them to employee artifacts. To put it simply, the manager simply defines what they want and when they want it to be completed, and the employees are automatically notified and told when to work.

Target Users and Software Requirements

Target Users:

- Managers who need to assign and monitor employees' tasks and progress.
- Employees who receive, manage, and work their assigned tasks.

Software Requirements:

- Frontend: React, Node.js, JavaScript, HTML, CSS, TailwindCSS,
- Backend: Python, Flask
- Developer Tools: Github, Git, NPM, Vite.js, VScode

Design, Management, and Development Process

Design process - The initial design was prototyped in Figma, followed by a second design that retained the framework but improved user experience. A design flaw led to the third and final design, resulting in the completed product.

Management Process - Each group member had a responsibility: design, backend development, and front-end development. Collaborating on these responsibilities ensured the project's successful completion.

Development Process - The project started by understanding the challenge. First, we designed the product, then built the front-end for a great user experience. Finally, we implemented the backend algorithm to maximize functionality and efficiency.

Challenges and Future Improvements

General Quality: Due to time constraints, the quality of the application may not be up to industry standards. In the future, we'd like to improve the general functionality, appearance, and quality of Taskly.

Features: With a tight timeline, we struggled to achieve the desired functionality of all our component's and associated logic. We'd like to improve this and achieve the expected functionality for each of our components. There is also a lack of quality of life features present in Taskly. We'd like for employees to have the ability to incorporate different types of hours and days (Sick Days, Injuries, Vacation, Lunch Hours).

Algorithm: Building an algorithm which optimizes tasks scheduled was difficult due to the multiple factors that had to be considered in determining what is important and what is not.

Libraries and Alternatives

Taskly utilizes several open-source libraries, including React, Flask, react-datepicker, react-big-calendar, react-dom, react-hook-form, react-query, and react-router-dom. Which are essential components in ManagerDashboard.jsx and EmployeeDashboard.jsx.

We considered alternatives such as fullcalendar for replacing react-big-calendar. Although fullcalendar is more feature rich, it is very complex to setup. Our time constraints led us to choose react-big-calendar for efficiency. This approach extends to other libraries as well. Using open-source libraries allowed us to streamline development and focus on delivering a working prototype.

Core Functions

A task is defined by the manager, the algorithm receives the task and generates assignments, and the assignments are received by each individual employee. To streamline the manager's job, they simply define and broadcast a task.
Our algorithm will compare employee schedules and assign the task to one or more employees to ensure that the task is completed on time. When the manager is defining the task, the created task object has no knowledge of the employees and their established working hours.
Since the manager's experience (manager dashboard) is very simplified, it simply calls the create_task() API call and the backend handles the rest. The employee dashboard makes API calls like get_users (to get the list of existing users) and get_employee_tasks and get_employee_tasks_by_name, so that the calendar can be generated based on the assignment objects that were created in create_task() and task assignments can be listed out in text for a better user experience.

Method of Installation

To run backend: `cd backend && pip3 install flask flask_cors –break-system-packages && python3 server.py`
To run frontend: `npm i && npm run dev` then go to localhost:5713