Eco Clean – Scheduling Solution

Midterm Report

Midterm Video Demonstration (Public Link):

https://collegedouglas-my.sharepoint.com/:v:/g/personal/atapattua_student_douglascollege_ca/IQB95W9axhD3Qq61OILWNagzAe1uYc1YkBY3wmzCLdOMkWE?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJPbmVEcml2ZUZvckJ1c2luZXNzIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOiJNeUZpbGVzTGlua0NvcHkifX0&e=alcGvh

Course: CSIS 4495	Section: 002
Full Name	**Student Number**
Atapattu, Upul M [Team Lead]	300392188
Arachchi Mudiyanselage, Vidarshan	300392818

Table of Contents

Introduction

The Eco Clean Scheduling System is a full-stack web and mobile-based scheduling solution designed for Nettoyage Eco Vert, an eco-friendly cleaning services company operating in Quebec. The system aims to digitize and automate the company's manual scheduling operations.

The current workflow relies heavily on spreadsheets, emails, and manual coordination, leading to inefficiencies, double-bookings, delayed communication, and scalability constraints. This applied research project focuses on designing and implementing a scalable digital platform that integrates scheduling, staff allocation, appointment tracking, and communication in real time.

Research Question that we are trying to answer is, "How a digitally integrated scheduling system contribute to the improvements in operational efficiency and reduce administrative overhead for a service-based cleaning business and how can we design and implement such a platform?"

Summary of the Initially Proposed Research Project

Based on our initial proposal, we are working on the following:

A Web-based Admin Scheduling Platform

- A Cross-platform Mobile App (iOS + Android)
- Real-time notifications
- Calendar integration
- Map-based navigation support
- Automated reminders
- Role-based access (Admin, Staff, Client)

The backend was proposed using:

- Node.js
- PostgreSQL
- Redis
- BullMQ for background scheduling

The frontend was proposed using:

- React
- Next.js
- React Native with Expo

Changes to the Original Proposal

Originally Proposed Work Scope

Web-Based Scheduling Platform

- User-Friendly Interface to accommodate the needs of client, front line worker and administrative staff
- Scheduling management system for booking, rescheduling, and cancelling appointments.
- Admin & Staff Dashboard for monitoring and managing schedules and assigning resources
- Google Maps integration for service area validation and route planning.
- Calendar Integration for both client and staff regarding the appointments
- Notifications and Reminders on bookings and updates

Mobile Application (iOS & Android)

- Client interface to view, book, and manage appointments and receive real-time updates
- Staff interface to access daily schedules, view client addresses and notes, and navigate via integrated maps. Additionally, if the frontline staff is delayed, they can update the client on the delay.

Changes to the Proposed Work Scope

After having a detailed discussion with Stephanie Riddelle, we have the following changes to the original proposal.

- Even though the original proposal initially talked about client-side mobile interface, Stephanie has said the business is not driven by the client-side online booking. Therefore, she did not want to direct our efforts on this.
- The business runs NOT on shift basis and staff travel from their houses to the appointment places.
- Staff Time Tracking is a Must Have, with GPS-based clocked in/out. The interface has should have a count-down timer for the remaining appointment time. The appointments should have 4 different sections for the Location information, WiFi details, Other instructions and Attaching photos.
- Staff interface should have a way to update their availability, apply leave 2 weeks prior. The staff app should be connected to Google Maps for navigation.
- The web application should connect with Google Maps to calculate travel time. Also, when scheduled, the calendar should offer a repeat option.
- Most preferred design idea can be inspired from https://www.getjobber.com/

Furthermore, considering the cost of launching a mobile app for both Android and IOS, we are proposing a Progressive Web Application (PWA) to reduce the cost for platform as it would be an economical solution.

Platform	Recurring Cost	One-Time Cost
iOS App Store	~$99/year	
Google Play Store		$25 (one-time)

PWA is a web-based application that uses browser capabilities in order to deliver app-like experience to users. Compared to traditional mobile applications, a PWA does not require installation through Apple App Store or Google Play Store. These PWAs are built using standard web technologies (HTML, CSS, JavaScript) but also include additional capabilities such as:

- Offline functionality
- Home screen installation
- Push notifications
- Background synchronization
- App-like full-screen experience
- Responsive design for mobile

Since our current web-app is being built using Next.js (App Router), we believe the transition to a PWA model is technically feasible with minimal architectural change.

Originally Proposed Architecture

- Express.js backend
- Separate frontend and backend services
- PostgreSQL database

Currently Implemented Architecture

- Next.js (App Router) for both frontend and backend (API routes)
- Prisma ORM for database abstraction
- PostgreSQL
- Mantine 8 for UI framework
- NextAuth for authentication

Justification

Our decision to adopt a full-stack Next.js architecture provided several advantages:

- Reduced deployment complexity using various components
- Routing and middleware integration was easier to implement

- API routes and UI components were more easily modified for changes
- Improved maintainability

Prisma ORM was selected to:

- Provided schema-driven database modelling
- Reduce runtime SQL errors
- Improve developer productivity

This adjustment strengthened long-term maintainability and reduced configuration overhead.

Prioritizations

For the mid-term evaluation, the priority was given to the following to demonstrate how the solution is going to look and feel from UI perspective. Also, our focus was on developing a stable implementation for the backend. The following are the prioritized components.

Prioritized Components:

- Authentication system (NextAuth with credentials provider)
- Role-based access control
- Client management (CRUD)
- Appointment management (CRUD)
- Staff profile interface
- Leave request UI (partial implementation)
- Database schema design
- API route architecture

Project Planning & Timeline

Phase / Task	Leading Person(s)*		15-Jan-26	22-Jan-26	29-Jan-26	05-Feb-26	12-Feb-26	19-Feb-26	26-Feb-26	05-Mar-26	12-Mar-26	19-Mar-26	26-Mar-26	02-Apr-26	09-Apr-26
1. DISCOVERY & ANALYSIS															
Understanding the Riipen project and its scope	Upul & Vidarshan	Planned													
		Actual													
Market research on existing solutions for scheduling	Upul	Planned													
		Actual													
Current Process Analysis	Upul & Vidarshan	Planned													
		Actual													
Stakeholder Interviews	Upul & Vidarshan	Planned													
		Actual													
2. SYSTEM DESIGN															
User Interface Design	Upul	Planned													
		Actual													
Database Schema Design	Upul	Planned													
		Actual													
3. WEB PLATFORM DEVELOPMENT															
Backend Development	Vidarshan	Planned													
		Changes to the Original Plan													
		Actual													
	Upul	Planned													
		Changes to the Original Plan													
		Actual													
Frontend Development	Vidarshan	Planned													
		Actual													

	Upul	Planned	
		Changes to the Original Plan	
		Actual	
4. MOBILE APP DEVELOPMENT			
~~React Native/Expo Setup~~	Vidarshan	Original	
PWA Development		Changed Scope	
~~Client App Features~~	Vidarshan	Original	
Further Development based on Additional Features such as AI tools		Changed Scope	
Staff App Features	Upul	Planned	
		Started Early	
5. INTEGRATION & TESTING			
API Integration	Vidarshan		
System Testing (Whole system)	Upul & Vidarshan		
User Acceptance Testing	Upul		
6. DOCUMENTATION			
User Manual Creation	Upul		
Final Presentation Prep	Upul & Vidarshan		

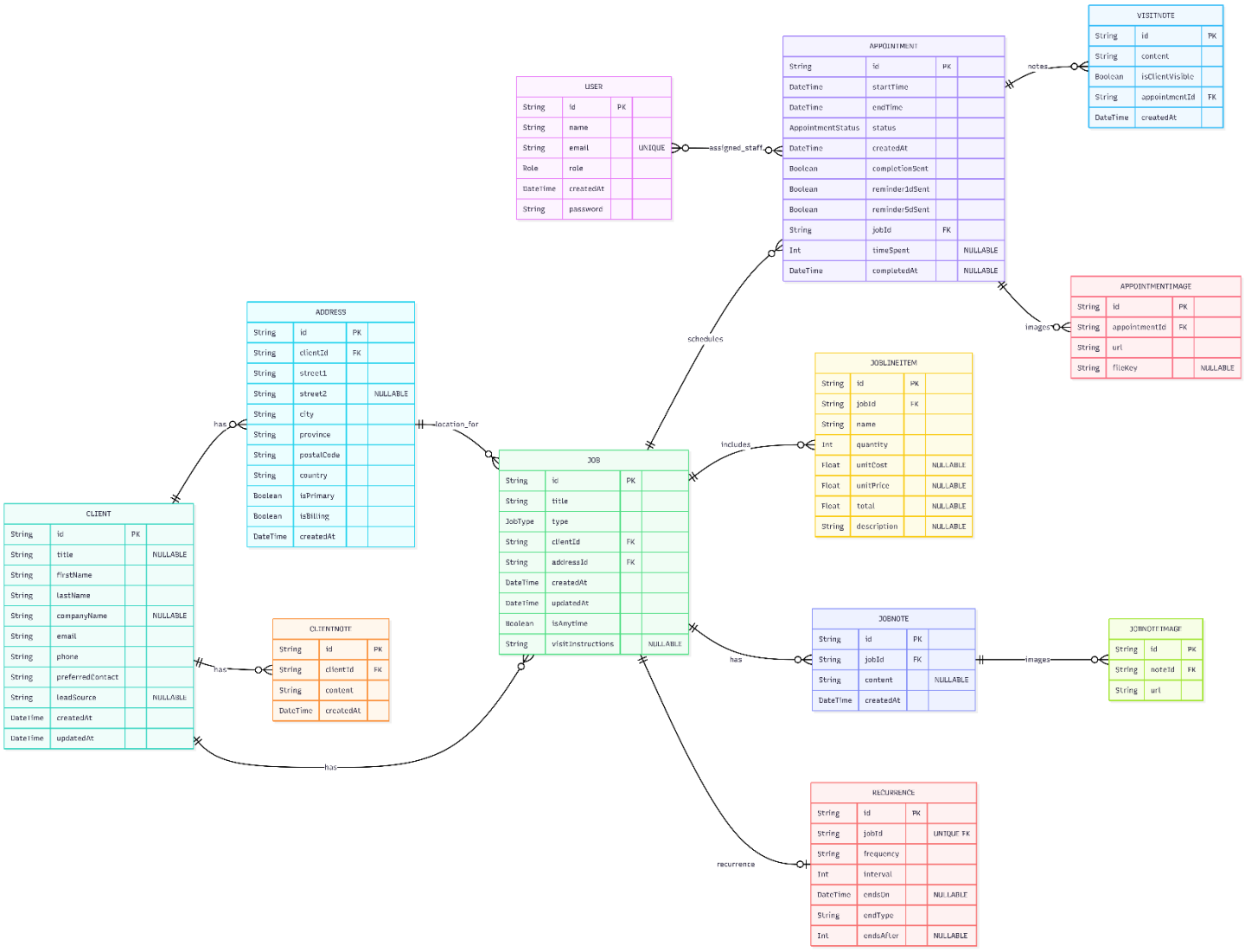
*Please note that both members will collaborate across all phases, with leadership roles assigned according to individual strengths and technical expertise.

Implemented Features (Midterm)

We have implemented the following things in our project up to this point.

4.1 Database Architecture

The following is the currently used DB design (Visualized using Mermaid).

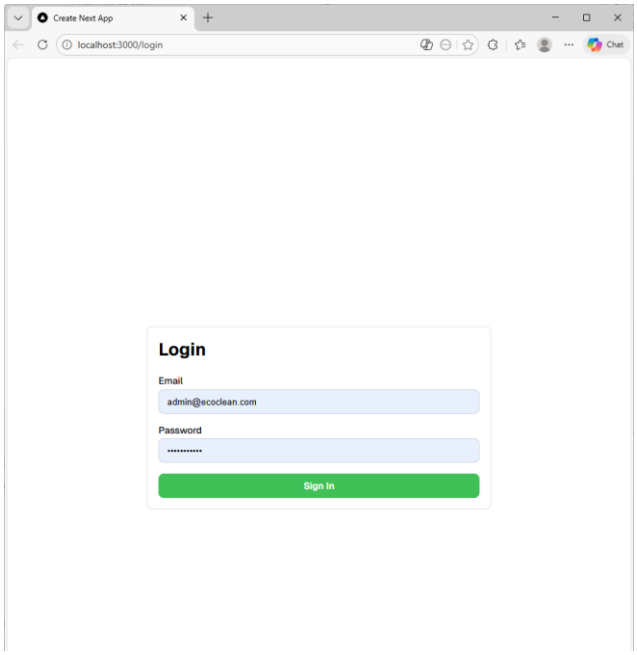


Even though the current roles has been designed for ADMIN, STAFF, CLIENT no interface difference was done based on the role so far.

Frontend Interfaces (Integrated Ones)

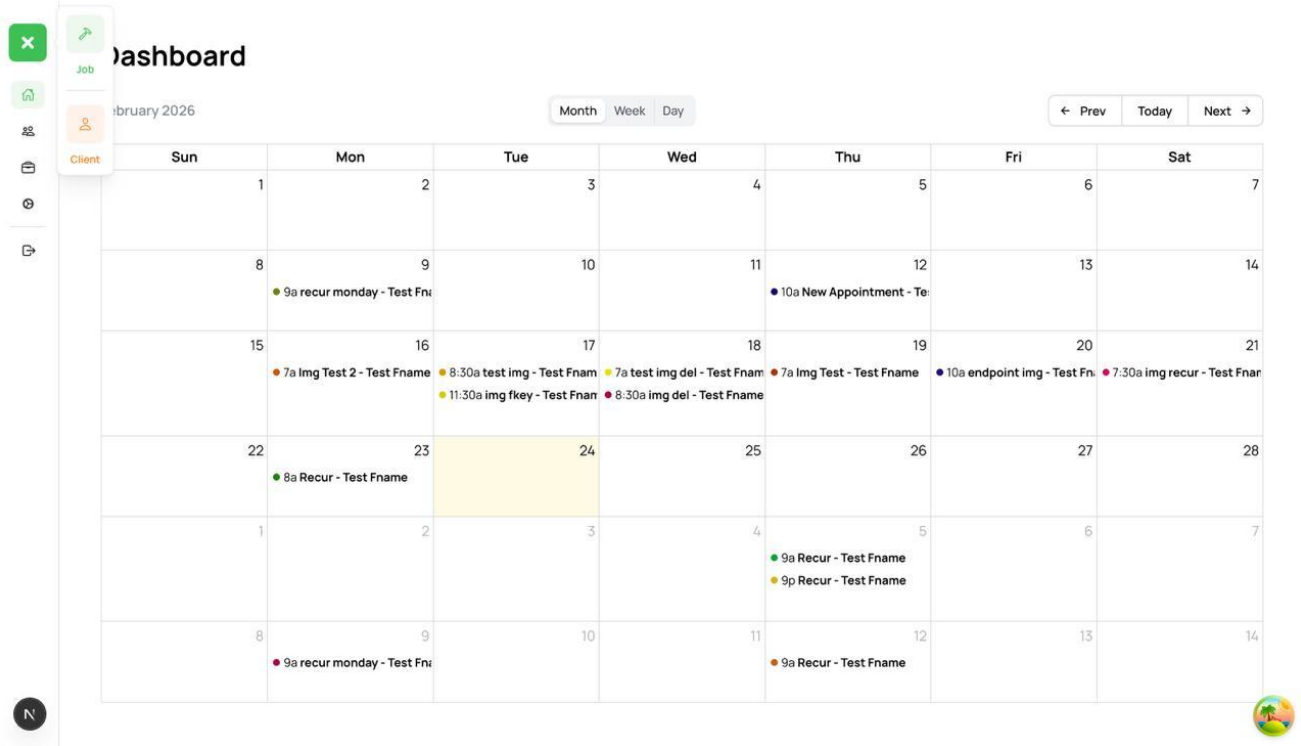
The main Interfaces are as follows:

Login Interface: (Admin or Staff can login)



Dashboard

The dashboard was created with a calendar view to showcase the existing appointments.

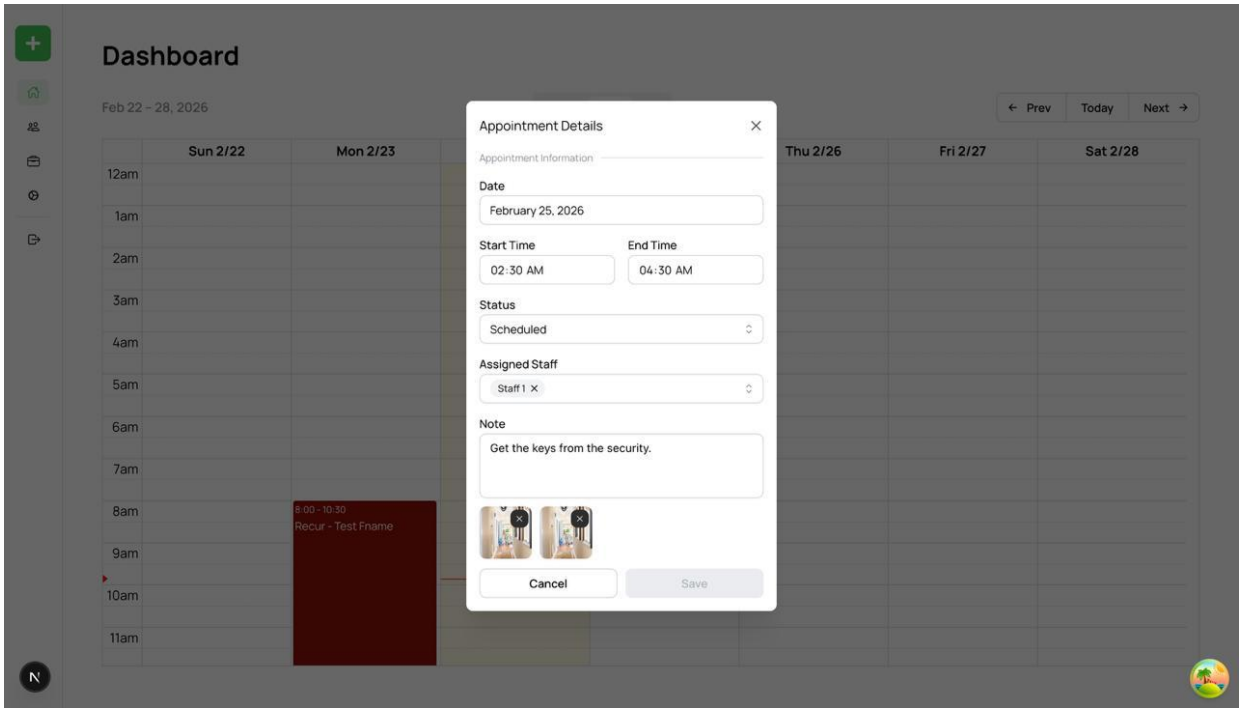


A user can view the exiting appointments and add the appointments.

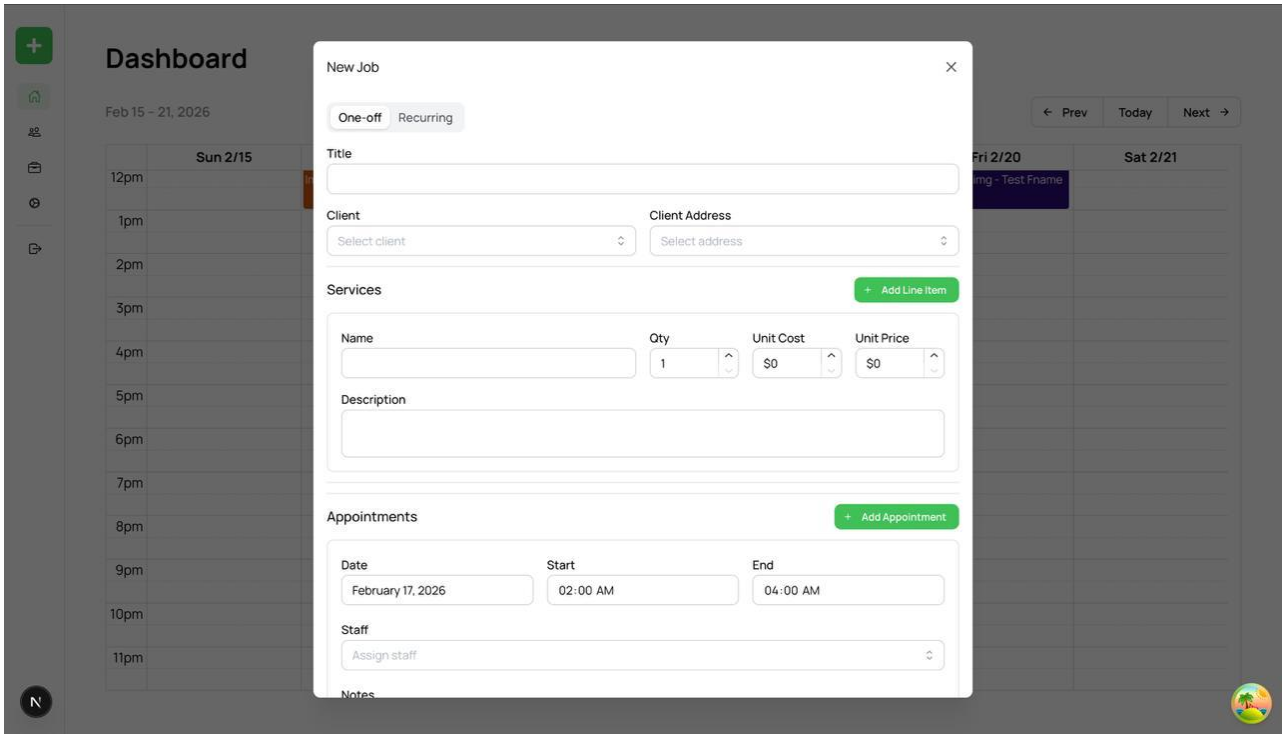
View Appointments, Add One-time Appointment, Add Recurring Appointments

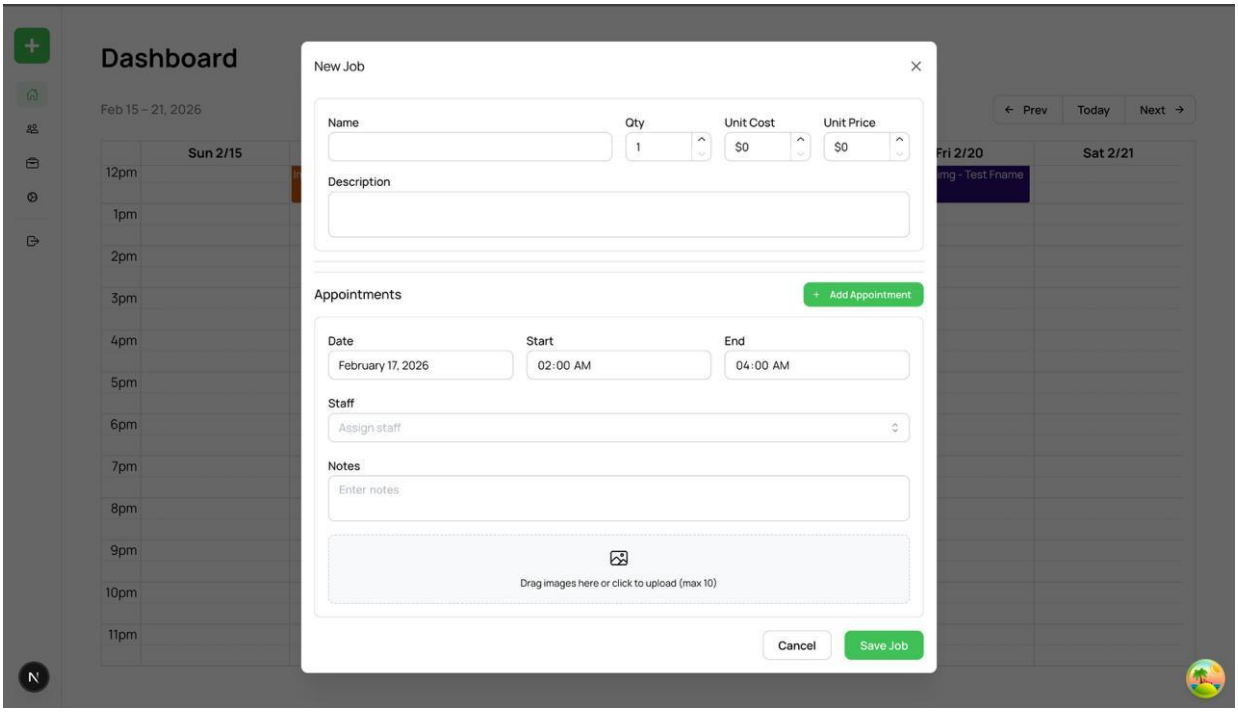
It was a requirement from the client that we are able to create jobs/appointments which facilitate image attachments. Furthermore, the client wanted the one-off and recurring appointment option as well.

View Appointments

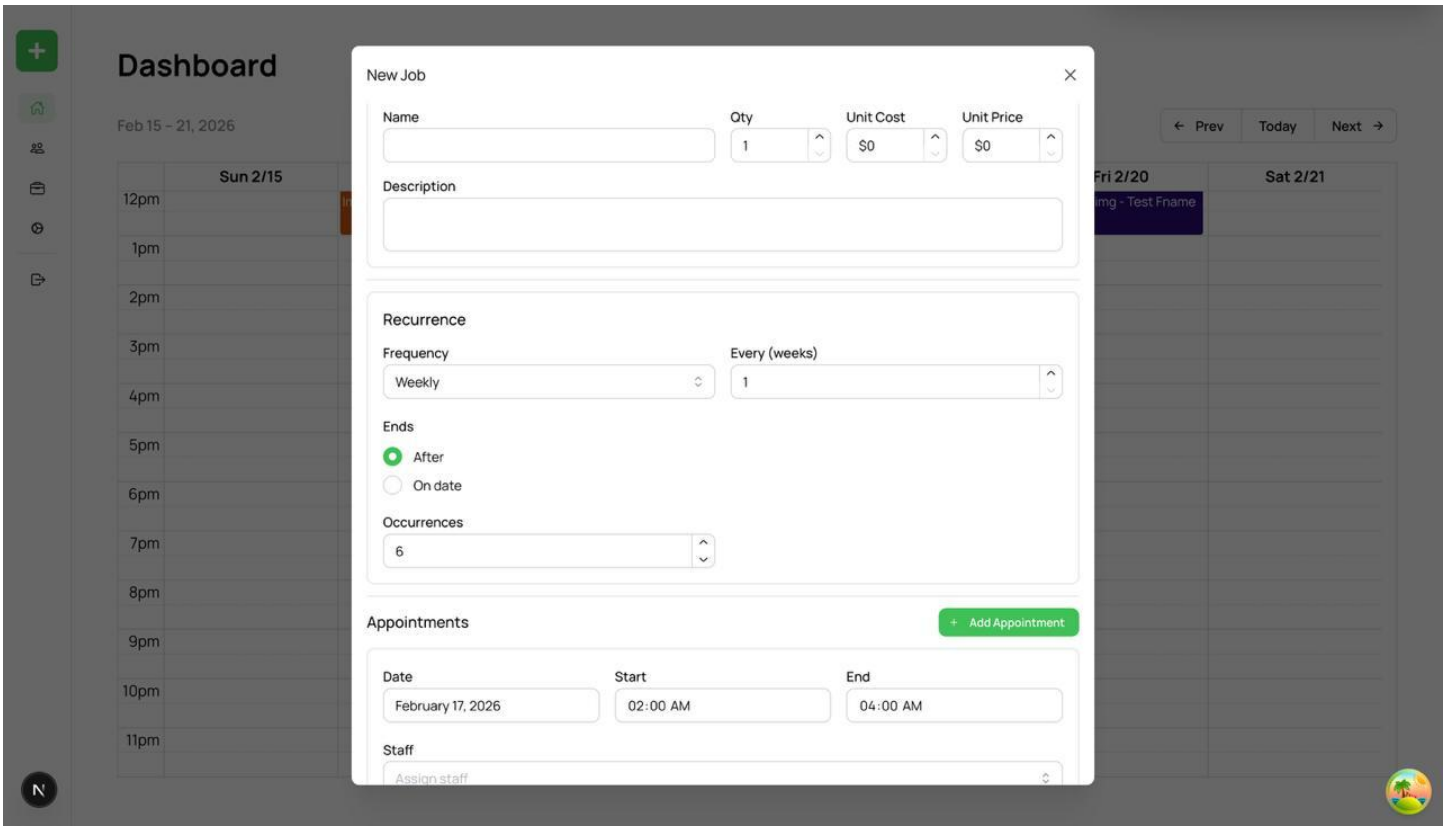


One-time Appointments

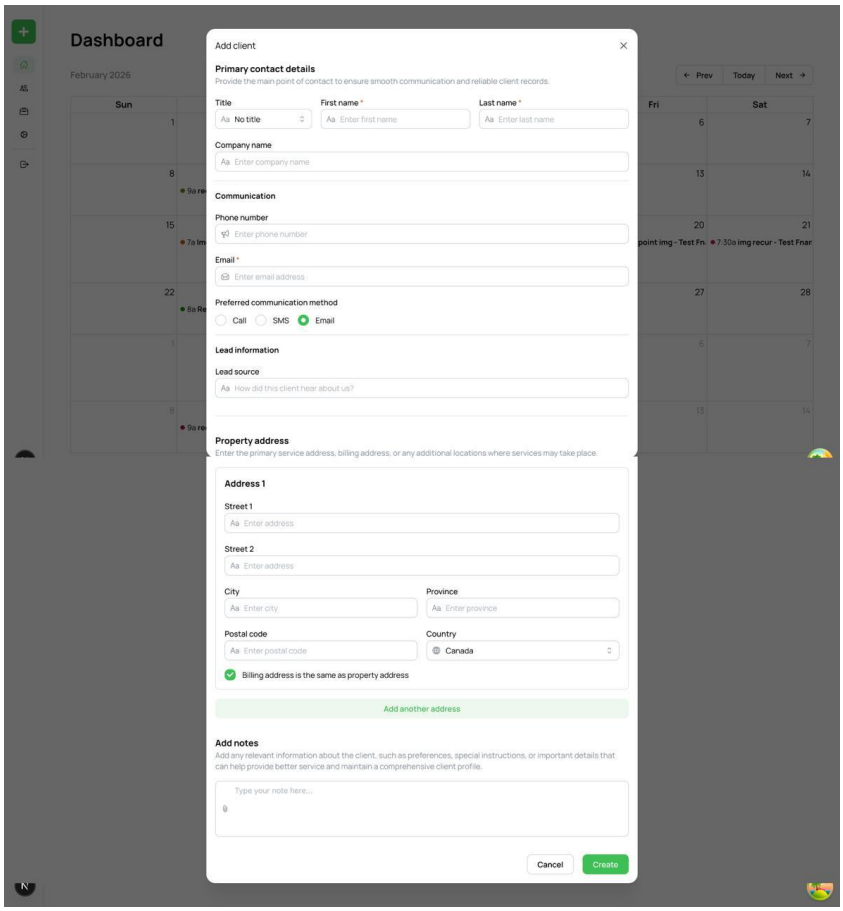


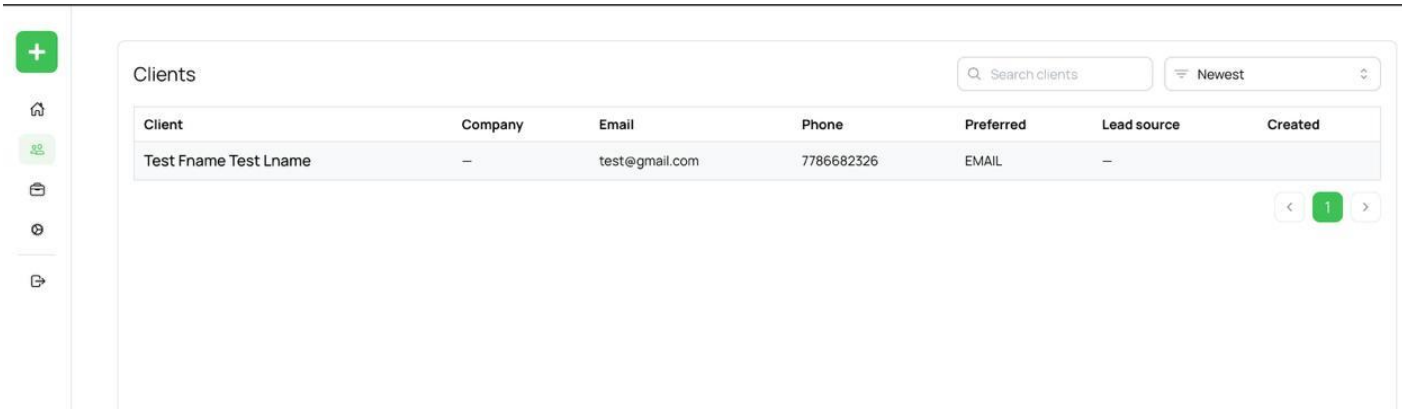


Recurring Appointments



Add Clients and View Clients





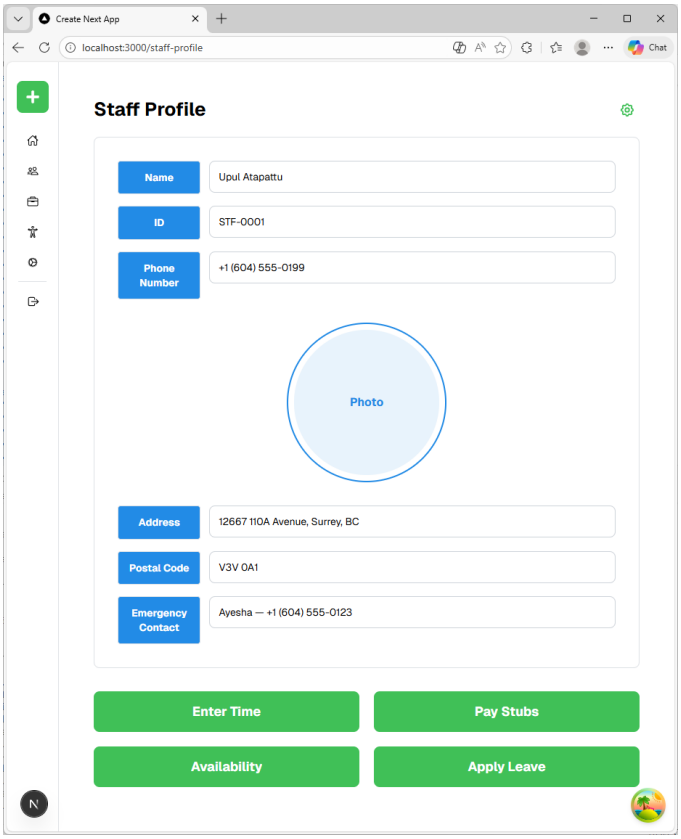
Leave Request Interface and Staff Time Sheet Interface (Only the UI part)

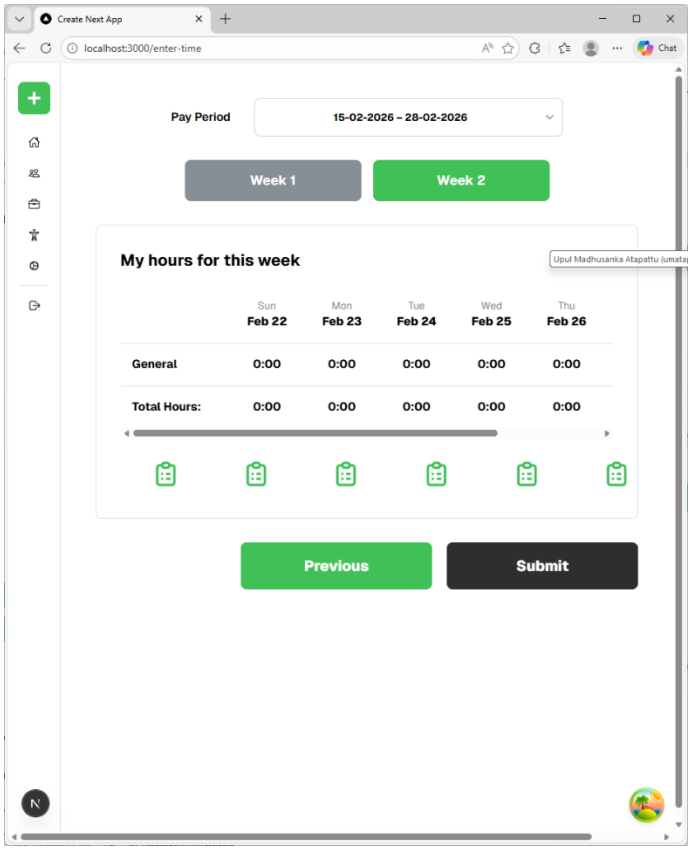
For the Leave and Leave request ge leave request interface was developed:

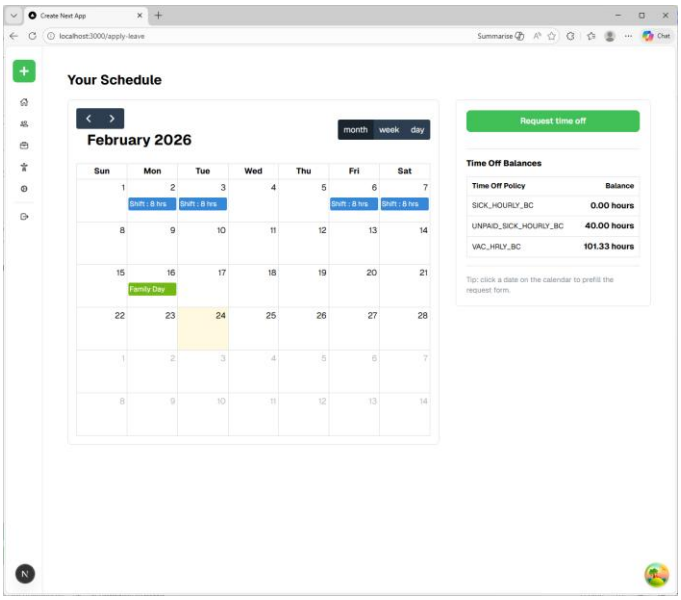
1. Calendar view with leave balance
2. Leave request form with date selection

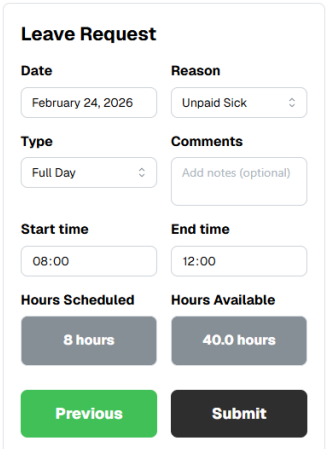
Dynamic UI rendering is managed using React state management.

Backend validation integration is currently in progress.









Challenges Encountered

When dealing with the Implementation, we had run into the following issues.

Prisma 7 migration complexity

We have changed the DB design as the interfaces were developed. So, it took several migrations, which made the process tricky. Furthermore, it made the database structure in one environment becomes different from another environment. (Between development machines). Sometimes we had to reset the development database after making major changes to the database structure.

Environment variable configuration issues

With local db credential mismatch, the ENV file had to be adjusted several times, when we clone the repo in a machine other than one we initially used. Clean install of the Postgrey was need. Sometimes, the database URL string had to manually update after Postgres credential issues.

TimeZone Issue in Calendar

Since the development is done in PST time, when we book an appointment it was recorded in the same time zone. But, since this product is used in Quebec, we had to figure out a way to save the appointment in TimeZone matching way. To resolve this, the appointment data was saved in a timezone-consistent manner (UTC) and converted dynamically based on user location.

Vercel Deployment Errors

Eventhough "npm run dev" is lenient, when we try to implement the development in Vercel, it's stricter build required many strict type errors handling. Even resetting the database was necessary.

Authentication Bypass

Authentication bypassed during development as it made it difficult to collaborative development.

Work Logs

Member: Vidarshan

Date	Hours	Description
Progress Report 1 + Project Proposal		
27th January 2026 4:30pm – 6:30pm	2.0 hours	Setup authentication with Next.js
27th January 2026 8:30pm – 10:30pm	2.0 hours	Implemented user get, edit, delete, create functions on the API
3rd February 2026 10:30pm – 1:30am	3.0 hours	Connected mobile app to the web API
3rd February 2026 1:30pm – 2:30pm	1.0 hours	Meeting with client to discuss the requirements
3rd February 2026 4:00pm – 6:45pm	2.75 hours	Persisted logged-in status on React Native
3rd February 2026 8:30pm – 11:30pm	3.0 hours	Implemented authenticated navigation flow and rectified navigation control issues
4th February 2026 4:30pm – 6:30pm	2.0 hours	Implemented appointments endpoint
4th February 2026 8:30pm – 10:30pm	2.0 hours	Sent emails through cron jobs
5th February 2026 4:30pm – 7:00pm	2.5 hours	Implemented dashboard UI
Progress Report 2 + Midterm Submission		
8th February 2026 6:00pm – 8:00pm	2.0 hours	Implemented client listing with Prisma pagination and sorting
8th February 2026 8:30pm – 10:00pm	1.5 hours	Integrated dynamic orderBy logic (oldest/newest toggle)
9th February 2026 4:30pm – 6:30pm	2.0 hours	Connected Mantine Select to backend client data
9th February 2026 7:00pm – 9:00pm	2.0 hours	Implemented searchable Select with onSearchChange filtering

10th February 2026 3:30pm – 5:30pm	2.0 hours	Structured dashboard layout using Mantine Grid and Stack
10th February 2026 6:00pm – 8:30pm	2.5 hours	Implemented sidebar collapse behavior (Mantine v6 responsive layout)
11th February 2026 2:00pm – 4:00pm	2.0 hours	Designed structured form layout (Primary Contact & section grouping)
11th February 2026 5:00pm – 7:00pm	2.0 hours	Connected form submission to backend API endpoints
11th February 2026 8:00pm – 9:30pm	1.5 hours	Improved spacing, hierarchy, and alignment for form UI
12th February 2026 1:00pm – 3:00pm	2.0 hours	Integrated FullCalendar with Mantine dashboard layout
12th February 2026 3:30pm – 5:00pm	1.5 hours	Configured calendar plugins (timeGrid, dayGrid, interaction)
12th February 2026 6:00pm – 7:30pm	1.5 hours	Debugged inconsistent page sizing and standardized container widths
12th February 2026 8:00pm – 9:00pm	1.0 hours	Generated database dump process for team collaboration
12th February 2026 9:00pm – 10:00pm	1.0 hours	Refactored reusable layout structure for scalability
13th February 2026 10:00am – 12:00pm	2.0 hours	Refactored Job–Appointment relationship logic (Prisma schema updates)
13th February 2026 2:00pm – 3:30pm	1.5 hours	Resolved Prisma migration constraint errors for required datetime fields
13th February 2026 7:00pm – 9:00pm	2.0 hours	Improved job creation transaction flow (Anytime vs Scheduled logic)
16th February 2026 3:00pm – 5:00pm	2.0 hours	Standardized date utilities (ISO ↔ date-only ↔ time conversions)
17th February 2026 11:00am – 1:30pm	2.5 hours	Reworked Anytime scheduling logic in Job POST route
17th February 2026 3:30pm – 6:00pm	2.5 hours	Fixed Prisma staff relation errors (staff vs staffId correction)

18th February 2026 9:00pm – 11:30pm	2.5 hours	Debugged appointment update errors (Invalid Date handling)
18th February 2026 11:30pm – 12:30am	1.0 hours	Normalized API date parsing to prevent runtime failures
19th February 2026 2:30pm – 4:30pm	2.0 hours	Fixed appointment PATCH route (async params resolution)
19th February 2026 5:00pm – 6:30pm	1.5 hours	Resolved DELETE route debugging for uploadthing integration
19th February 2026 7:00pm – 9:00pm	2.0 hours	Refactored line item pricing logic (unitCost vs quantity modeling)
19th February 2026 9:00pm – 10:30pm	1.5 hours	Streamlined API validation and frontend data normalization
20th February 2026 3:00pm – 5:00pm	2.0 hours	Improved appointment image upload handling (Mantine Dropzone integration)
22nd February 2026 8:00pm – 10:30pm	2.5 hours	Planned PWA transition strategy (unified web/mobile architecture)
23rd February 2026 9:30am – 11:30am	2.0 hours	Reviewed cloud scalability & CDN strategy for production readiness
23rd February 2026 1:00pm – 3:00pm	2.0 hours	Evaluated infrastructure design for fault tolerance and automation
Total	**77.75 hours**	

Member: Upul

Date	Hours	Description
15th January 2026 2:30am-3:30am	1.5 hour	Prepare the project briefly after studying the company website and the project scope.
20th January 2026 1:30pm-2:00pm	0.5 hours	Had the initial meeting with Stephanie Riddel to understand the scope of the Riipen project.
24th January 2026 4:00pm - 5:00pm	1 hour	Researched on existing scheduling solutions (ADP) and what it currently serves, what are their capabilities to understand the current market.
25th January 2026 3:00pm – 4.30pm	1.5 hour	I studied and reviewed the section that Vidarshan has done on the technologies and the development tools.

25th January 2026 8:00pm – 8.30pm	0.5 hours	Prepared the project timelines and deliverables
26th January 2026 5:00pm – 8:30pm	3.5 hours	Worked on the project proposal to match the project briefly, giving background, project scope, benefits and justification and finalized.
29th January 2026 10:00pm – 11:00pm	1.5 hours	Cloned the Github repo and reviewed the code that Vidarshan had initiated. However, no contribution to the code was done.
30th January 2026 9:45pm – 11.30pm	1.75 hours	Discussed with Vidarshan on the wayforward in terms of the implementation. Discussed on few brainstorming ideas for initial design for the mobile app (Based on MiHCM) and web portal (based on ADP, timesheets used by Douglas Employees portal)
31st January 2026 3:00pm – 4:00pm 5:30pm – 6:00pm	1.5 hour	Started a Udemy course on React as my fullstack understanding is not enough to do a proper vibe coding job. Covered the state management implementation and building form components with state.
1st February 2026 10:30am – 11:00am 12:00pm – 1:30pm 3:30pm – 4:30 pm 6:30pm – 7:00 pm	3.5 hours	Continued my learning on Udemy platform on component library development such as creating button, input and modal components.
2nd February 2026 9:00 pm – 10:00 pm 11:30pm – 12:30 am	2.0 hours	Covered styling system implementation includes setting up CSS modules and creating responsive layouts.
3rd February 2026 1:30pm – 2:30pm	1 hour	Had a google meet meeting with Stephanie Riddelle to talk about the business model, her requirement and the current process of scheduling.
4th February 2026 12:00am – 1:00am 2:00 am – 3:00am	2 hours	Continued my learning on Udemy platform on routing implementation topics such as setting up route structures, implementing protected routes for different user roles, and creating reusable navigation components.
5th February 2026 12:00 am – 1:30pm	1.5 hours	Reviewed the video call recording for better understanding of the scheduling process and the requirement. Based on Stephanie's recommendation, registered for

		www.getjobber.com as she preferred the user interface design and the workflow is better suited for her business.
7th February 2026 8:00 pm – 9:00pm	1.5 hours	Set up the project in my local machine with Vidarshan's help with the necessary env file for postgre integration.
8th February 2026 8:00 am – 9:00 am 2:00 pm – 3:00 pm 4:00 pm – 4:30 pm	2.5 hours	Continued my learning on Udemy platform on React useEffect topics including API integration patterns, loading and error state management, implementing theme selectors for light/dark mode functionality.
9th February 2026 7:00pm – 8:00pm	1 hour	Learnt about NextJS from YouTube, covering, why it is different from react, server-side rendering and API routes for backend integration for NextJS 15+.
11th February 2026 10:00pm – 11:00pm 12:00am – 1:00am	2 hours	Drafted improved UI/UX wireframes for admin dashboard based on ADP and Jobber systems.
13th February 2026 8:30 am – 9:30 pm 4:30pm – 5:30 pm 6:00pm – 7:30pm	3.5 hours	Ran into and issue with the development set-up due to postgre credentials mismatch. Had to remove the postgre and reinstall it in order to fix it. This has resulted in the previous data dump not integration mismatches. Therefore, had tried various options 1. Tried creating another admin account using postman [which did not go as planned]. 2. Therefore, disabled the authentication logic to move forward with the development for a separate workbranch.
14th February 2026 8:30 am – 9:30 pm 10:30 pm – 11.00 pm	1.5 hours	Since, my development branch got a warning, "The "middleware" file convention is deprecated. Please use "proxy" instead.", worked on moving to the proxy file convention.
15th February 2026 8:30 am – 9:00 am 3:00 pm – 3:45 pm 5:00 pm – 5:30 pm	1.75 hour	I have started on the design of the staff related interfaces, specifically leave management and timesheets related interfaces.
16th February 2026	2 hours	I started implemented the frontend UI related elements of staff related interfaces, with the help of ChatGPT.

18th February 2026 4:00 pm – 5:00 pm	1 hour	Continued working on implementing the UI related elements that I designed and finished the two interfaces.
21st February 2026 9:00 pm – 10:00 pm	1 hour	Configured Supabase PostgreSQL hosting and environment variables get ready for Vercel deployment
22nd February 2026 8:30 am – 9:45 am 3:00 pm – 3:45 pm 5:00 pm – 6:30 pm 8:00pm – 9:00 pm	3.5 hour	After setting up the prerequisites for Vercel, attempted Vercel deployment; encountered various build-time strict TypeScript errors from my end. Eventhough significant time was spend for debugging this, could not get the deployment to build.
23rd February 2026 4:00 pm – 5:30 pm 6:30 pm -7.00 pm	2 hours	Started on improving the DB design, so, we can improve on facilitate the staff leave, tracking work logs which is needed to do timesheets and payroll calculations.
24th February 2026 4:30 pm – 5:30 pm 6:00 pm -8.00 pm 2:00 pm – 3:00 pm	4 hours	Prepared the installation guidelines in the repo. Prepared the Mid Term Report. Edited the Video
Total	**51 hours**	

AI Usage

Member: Upul

AI	Prompt	Value Addition
ChatGPT, Free	I'm trying to create a scheduling web-app. I want create a 24*7 calender in HTML, so I can later use with react, how would you guide me to do it. Let me show it part by part	Created for learning purposes and to create JSX components if necessary to implement in react.
ChatGPT, Free	how to use https://fullcalendar.io/ for do a schedule calendar	For brainstorming purposes
ChatGPT, Free	when I'm trying to do scheduling web app with next js. What are the things that I should learn? I have a grasp of react. What is different in NextJs and what are the areas I should expand?	Learning purposes
ChatGPT, Free	how to setup my postgre env file	To troubleshoot in the development environment
Gemini, Free	how to deal with ⚠ The "middleware" file convention is deprecated. Please use "proxy" instead. Learn more: https://nextjs.org/docs/messages/middleware-to-proxy × The file "./middleware.ts" must export a function, either as a default export or as a named "middleware" export. ?	To troubleshoot in the development environment

ChatGPT, Free	Would you help me to resolve the following error in the terminal? npm error code ERESOLVE npm error ERESOLVE could not resolve npm error npm error While resolving: eco-clean-web@0.1.0 npm error Found: @mantine/core@8.3.14 npm error node_modules/@mantine/core npm error peer @mantine/core@"8.3.14" from @mantine/dates@8.3.14 npm error node_modules/@mantine/dates npm error @mantine/dates@"^8.3.14" from the root project npm error peer @mantine/core@"8.3.14" from @mantine/dropzone@8.3.14 npm error node_modules/@mantine/dropzone npm error @mantine/dropzone@"^8.3.14" from the root project npm error 1 more (the root project) npm error npm error Could not resolve dependency: npm error @mantine/core@"^8.3.13" from the root project npm error npm error Conflicting peer dependency: @mantine/hooks@8.3.15 npm error node_modules/@mantine/hooks npm error peer @mantine/hooks@"8.3.15" from @mantine/core@8.3.15 npm error node_modules/@mantine/core npm error @mantine/core@"^8.3.13" from the root project npm error npm error Fix the upstream dependency conflict, or retry npm error this command with --force or --legacy-peer-deps npm error to accept an incorrect (and potentially broken) dependency resolution. npm error	Fixed it by installing the Mantine version 8.3.14 based on the options provided by ChatGPT
ChatGPT, Free	What are the easier way to visualise the DB design from Prisma scheme?	To create DB design diagram
ChatGPT, Free	How can I generate a similar interface in NextJS (as shown in the image)? I'm hoping to use mantine	The same prompt was used to do vibecode and get the staff related interfaces. I had to manually the stylings, to match the profile I wanted.
ChatGPT, Free	What are the benefits of a PWA and limitations of it?	Learning purposes
ChatGPT, Free	Can a PWA send notifications?	Learning purposes
ChatGPT, Free	How can I create an installation instructions for my GitHub Repo. About the usage of the repository files. My project is a NextJS based web-app with Prisma integration. Do we have to ask them to setup postgre in their own machine?	For setting up the installation guidelines

Member: Vidarshan

AI	Prompt	Value Addition
ChatGPT 5.2 Plus	Prisma client.findMany with filtering, sorting, pagination	Built scalable backend query logic with dynamic orderBy and efficient skip/take pagination
ChatGPT 5.2 Plus	Dynamic Prisma orderBy toggle (oldest/newest) with TypeScript SortOrder fix	Implemented type-safe sorting logic resolving Prisma SortOrder typing errors
ChatGPT 5.2 Plus	Connecting Mantine Select component to backend data	Enabled dynamic dropdown population from API data
ChatGPT 5.2 Plus	Implement searchable Mantine Select with onSearchChange filtering	Added client-side search refinement for improved UX
ChatGPT 5.2 Plus	Mantine Grid and Stack dashboard layout structure	Designed scalable dashboard layout foundation
ChatGPT 5.2 Plus	Sidebar collapse behavior in Mantine responsive layout	Improved dashboard responsiveness and adaptive UI behavior
ChatGPT 5.2 Plus	Structured form layout with grouped sections (Primary Contact)	Enhanced form hierarchy and usability through structured grouping
ChatGPT 5.2 Plus	Connecting Mantine form submission to Next.js API routes	Enabled full-stack data flow between frontend forms and backend endpoints
ChatGPT 5.2 Plus	UI spacing and alignment optimization in Mantine forms	Improved visual hierarchy and professional polish of UI
ChatGPT 5.2 Plus	Integrating FullCalendar in Next.js App Router	Embedded interactive scheduling interface within dashboard
ChatGPT 5.2 Plus	Configuring FullCalendar plugins (timeGrid, dayGrid, interaction)	Enabled multi-view scheduling and interactive event management
ChatGPT 5.2 Plus	Fix inconsistent page sizing and layout width issues	Standardized container widths for consistent cross-page layout

ChatGPT 5.2 Plus	PostgreSQL database dump process for team collaboration	Established reproducible DB export workflow for shared development
ChatGPT 5.2 Plus	Reusable layout architecture in Next.js dashboard	Refactored layout for scalability and future feature expansion
ChatGPT 5.2 Plus	Prisma Job–Appointment relationship modeling	Improved relational integrity and transactional job creation flow
ChatGPT 5.2 Plus	Fix Prisma migration error (required column without default)	Resolved schema deployment blocking issue safely
ChatGPT 5.2 Plus	Anytime vs Scheduled appointment logic in transactional API route	Implemented conditional scheduling architecture
ChatGPT 5.2 Plus	ISO date handling between frontend and Prisma backend	Standardized date conversion utilities preventing runtime Date errors
ChatGPT 5.2 Plus	Fix Prisma staff vs staffId relation error	Corrected relational mapping and eliminated API runtime failure
ChatGPT 5.2 Plus	Debug Invalid Date object error in Prisma update	Prevented data corruption by normalizing API input parsing
ChatGPT 5.2 Plus	Next.js App Router async params handling in PATCH route	Fixed dynamic route parameter resolution preventing 500 errors
ChatGPT 5.2 Plus	Debug DELETE route not being triggered (upload integration)	Resolved routing mismatch and ensured image deletion reliability
ChatGPT 5.2 Plus	Unit cost vs quantity financial calculation modeling	Corrected pricing logic ensuring accurate invoice totals
ChatGPT 5.2 Plus	Frontend-to-backend data normalization strategy	Reduced validation failures and improved API resilience

ChatGPT 5.2 Plus	Mantine Dropzone image upload handling with backend integration	Enabled structured appointment image management
ChatGPT 5.2 Plus	Converting React Native mobile app to PWA architecture	Defined unified deployment strategy reducing maintenance overhead
ChatGPT 5.2 Plus	CDN and scalability considerations for production deployment	Strengthened architectural readiness for scalable cloud hosting
ChatGPT 5.2 Plus	Infrastructure design for fault tolerance and automation	Established production-oriented infrastructure planning framework

References

BullMQ - Background Jobs processing and message queue for NodeJS. (2018). Bullmq.io. https://bullmq.io/

Expo. (n.d.). Expo. Expo. https://expo.dev/

Home - Nettoyage Eco Vert. (2026, January 3). Nettoyage Eco Vert. https://nettoyageecovert.ca/

Mantine. (2025, May 5). *Version 8.0.0* (Changelog). Mantine.dev. https://mantine.dev/changelog/8-0-0/

Prisma. (n.d.). *Introduction to Prisma*. Prisma Documentation. https://www.prisma.io/docs

Redis. (n.d.). Redis. Redis.io. https://redis.io/

Vercel. (2026, February 20). *Getting started (App Router)*. Next.js Documentation. https://nextjs.org/docs/app/getting-started