Home

RosterApp

The aim of this project is to build a rostering tool that helps casual work employees manage their work schedule and share their work time availability with employers.

We aim to implement a solution that is:

- Employee-centric
- Intuitive and easy to use
- Protect user's privacy, information can only be shared with user's approval
- Maintainable and extendable

Progress Tracking

Trello Board

GitHub

Requirements

Analysis Of Requiremet

Supervisor

Name	Email
Madhavi Mohoni	m.mohoni@unimelb.edu.au

Team

Name	Github	Email	Part	Role
Wenxuan Chen	https://github.com/JeRRyChenwen	wenxchen2@student.unimelb.edu.au	Back End	Developer
Xuechun Chen	https://github.com/chen994	xuecchen@student.unimelb.edu.au	Back End	Product Owner
Sichen Lu	https://github.com/NoiRC256	sichen2@student.unimelb.edu.au	System	Developer
Bo Peng	https://github.com/Bloomberg0512	pengbp@student.unimelb.edu.au	Front End	Scrum Master
Chenxi Zhang	https://github.com/chenxizhang1	czzhan4@student.unimelb.edu.au	Front End	Developer

Client

Name	Photo	Position	LinkedIn	Email

Sharon Tooley		May 2021 - Present Head of Technology (Health Care Management, Finance) at Bupa	https://au.linkedin.com/in/sharon-tooley	sharon.tooley@gmail.com

Recent space activity

Chenxi Zhang
Sprint 4 Review updated 2 minutes ago view change

Wenxuan CHEN
Handover updated 16 minutes ago view change

Bo Peng
Demo updated 22 minutes ago view change

Sichen Lu
Handover updated about 4 hours ago view change
Development Documentation updated about 5 hours ago view change

Space contributors

- Chenxi Zhang (2 minutes ago)
- Wenxuan CHEN (16 minutes ago)
- Bo Peng (22 minutes ago)
- Sichen Lu (4 hours ago)
- Xuechun Chen (10 hours ago)
- ...

Project Overview

Background

A highly casualised workforce often requires workers to seek employment from more than one organisation. This is particularly true of workers in child care, aged care, retail, nursing, fast food, delivery services etc. where individuals will seek multiple employment opportunities in order to earn a meaningful wage.

While many employers utilise tools that assist with workforce management and rostering, employees are left juggling multiple methods of managing availability, shift allocation and their own personal time management.

In particular, smaller organisations often rely on ineffective tools such as SMS, WhatsApp, Phone calls, Facebook, Email, MSExcel, Notepad or other simple tools to create and manage rosters. This is often a time consuming activity that results in multiple and time consuming communications between employer and employee to identify and allocate vacant shifts.

Goals

The goal is to develop an app that helps the casual work employee share and manage their work time availability with employers and the app should fits in IOS system (minimum), android system and web system.

The entire client expectation is mainly divided into 2 phases:

Phase 1

Mandatory requirements, employee can:

- Set standard work day/hour availability
- Set ad-hoc non-working day/hour
- Set ad-hoc availability
- Share availability with employers potentially via email or SMS
- Integrate the shifts from app with personal calendars
- Have a simple calendar view showing allocated shifts including time and workplaces at a glance
- Save tax, superannuation details to allow digital sharing with existing employers.

Specifically:

1. An employee-centric, user-friendly app is required to allow employee save and share availability with employers by SMS and email (expectation is just easily click “share” button, in phase 2, share a link that shows a real-time availability schedule is expected).
2. UI design is up to students but the user experience should be simple, intuitive, easy to use and suitable for old people.
3. Once employee add a shift it allows employee to save this schedule in personal calendar.
4. Once employee add a shift or cancel a shift, the time availability to be shared should be updated accordingly.
5. User's privacy protection is important, private information, including time availability, identity, TFN, superannuation, shouldn’t be sent without permission from the owner (employee can share personal information only when they choose the option for that specific information).
6. Ideally, App should fits in iOS (minimum), android, and web-based system.

Phase 2

optional requirements:

- Employee can share link of a web portal for employers where employers can check real-time availability of the employee.
- Store copy of employee contract
- Login with credential.
- * Credential loading and validation for sharing with employers
- * Identify verification for employers to avoid sharing and employer maintaining identify documents (note: identity documents required for employment such as working with children will share minimum information required such as ID) - assume through 3rd party or Services Australia
- ^ Load award details to ensure timesheet and salary verification
- ^ Ability to calculate pay slips based on rostered/confirmed shifts
- ^ Ability to load pay slips for a consolidated view of total earnings to assist in identifying any upcoming tax implications (where employees work multiple jobs, there is a risk they are assigned a lower tax rate and will end up with a tax bill at EOFY)
- * Ability to load pay slips for all employees direct to tool
- * Employer identity currency check (License and WWCC) where required for employment
- * Real time view of availability between employee and employer
- * Identify available employees based on filtered requirements such as time, certificates etc
- * Push available shifts to available employees who can then accept or reject the shift in real time
- * Employer can manage rosters for all staff
- *^Integration with other common rostering tools from employers to manage allocated shifts (e.g. Deputy and common tools used by hospitals for rostering)

*Items that require a subscription / fee for service for employers

^Items that require a subscription / fee for service for employees

Specifically:

1. Certificate should be checked and validated (for example, truck driver need to present their license and app should be able to verify the validation of license before get a job) .
2. Personal information of both employee and employer must be confidential.
3. Include third party verification to check whether a employee or employer has legitimate ID.
4. Extract information from website (e.g., fare work commission) to calculate the minimum pay standard for employees according to their award. (Award from different job (nurses, teachers), years of experience, level of education and certificate can have different salary. With this function, employer can know how much they should pay for the employee and employee can know the salary they should get. However, the methodology of the minimum salary provided by government could be changed. Therefore, the methodology needs to be updated regularly) .
5. For casual workers, they might get paid by multiple employers, calculate the tax (sum up the total salary and the total tax so that employee know how much they have earned. For example, when your salary reaches the next level, you need to pay more tax)
6. Employers can save employees in a list and filter who is available in the list, filter employee based on credential, from high qualification to low qualification, past working experience and preferences.

Other optional requirements

- Identity security is very important, private information should be well contained unless there are two-way agreement (both employee and employer agree)
- Add security protection for message between employees and employers.
- TFN need to be pushed by employee only (protect employee's confidential information unless employee agree to disclose)

Requirement Analysis

Requirement Details

Phase 1 (mandatory features)

Availability management

- Set standard work day/hour availability
- Set non-working days
- Set ad-hoc availability

Export and sharing

- Employee can share availability with employers potentially via email or SMS. (expectation is just easily click “share” button)
- Employee can save and share their personal information like TFN and SFN with employers
- Integration with personal calendars, once employee accept or cancel a shift the schedule is updated in personal calendar.

Interface information management

- The user interface should be designed simple, intuitive, easy to use.
- Setting time availability function should be stand-alone and easy-to-reach.
- Have a simple calendar view of working week/month showing availability/place and time of shifts at a glance.
- A button for sharing work time availability near the calendar.
- A profile page including identity, tax, superannuation details of employee, this page should be editable and have a share button enabling employee to share selected information with employers.

Compatibility

- Ideally, App should be fits in iOS (minimum), Android, and web system

Phase 2 (non-mandatory features)

Information validation

* Credential loading and validation when share personal information with employers

Certificate verification (for example, truck driver need to present their license to get a job and app should be able to verify the validation of license before assigning a job)

* Identity verification for employers to avoid identity theft and employer maintaining identify documents (note: identity documents required for employment such as working with children certificate will share minimum information required such as ID) - assume through 3rd party organization or Services Australia

* Employer identity currency check (License and WWCC) where required for employment

^ Load award details to ensure timesheet and salary verification

Real-time communication system

^ Ability to calculate pay slips based on rostered/confirmed shifts

^ Ability to load pay slips for a consolidated view of total earnings to assist in identifying any upcoming tax implications (where employees work multiple jobs, there is a risk they are assigned a lower tax rate and will end up with a tax bill at EOFY)

* Ability to load pay slips for all employees direct to tool

* Push available shifts to available employees who can then accept or reject the shift in real time

* Real time view of availability of employee for employers

* Identify employee availability based on filtered requirements such as current allocated shifts, credentials, preference etc

* Manage rosters for all staff

Database system

Store copy of employee contract

Compatibility

* Web based portal for employers

*^Integration with other common rostering tools from employers to manage allocated shifts (e.g. Deputy and common tools used by hospitals for rostering)

Technical Details

Component	Tools	Justification	Alternatives
Frontend Framework (Mobile)	**React Native + TypeScript**	⊕ Cross-platform, supports iOS and Android ⊕ Familiar with JavaScript ⊕ Good ecosystem	**Swift** ⊖ iOS only ⊖ Not very familiar **Kotlin** ⊖ Android only **Flutter** ⊖ Not very familiar
Backend Framework	**Firebase**	⊕ Robust backend services ⊕ Less work for backend ⊕ Free	**Java Spring Boot** ⊕ Familiar with Java **Node.js** ⊕ Lightweight option for AWS Lambda **C# ASP .NET Core** ⊖ Not as familiar as Java
Backend Hosting	**Firebase**	⊕ Robust backend services ⊕ Popular, good community support ⊕ Free	**AWS Lambda, Amazon API Gateway** ⊕ Popular industry solution ⊕ Don't have to worry about server infrastructure **Azure App Services** ⊖ Have to host entire backend ⊖ Potential costs **Web Hosting** ⊖ Have to manage server ourselves
Database	**Firebase**	⊕ Robust backend services ⊕ Popular, good community support ⊕ Free	**MongoDB Atlas** ⊕ Free ⊕ NoSQL solution that fits our needs
DevOps	**Github Actions**	⊕ Directly accessible from GitHub ⊕ Free ⊕ Good ecosystem ⊕ Lots of community support	**Jenkins** ⊖ Steep learning curve **Azure Pipelines** ⊖ Cost

Motivation Model

Do-Be-Feel List

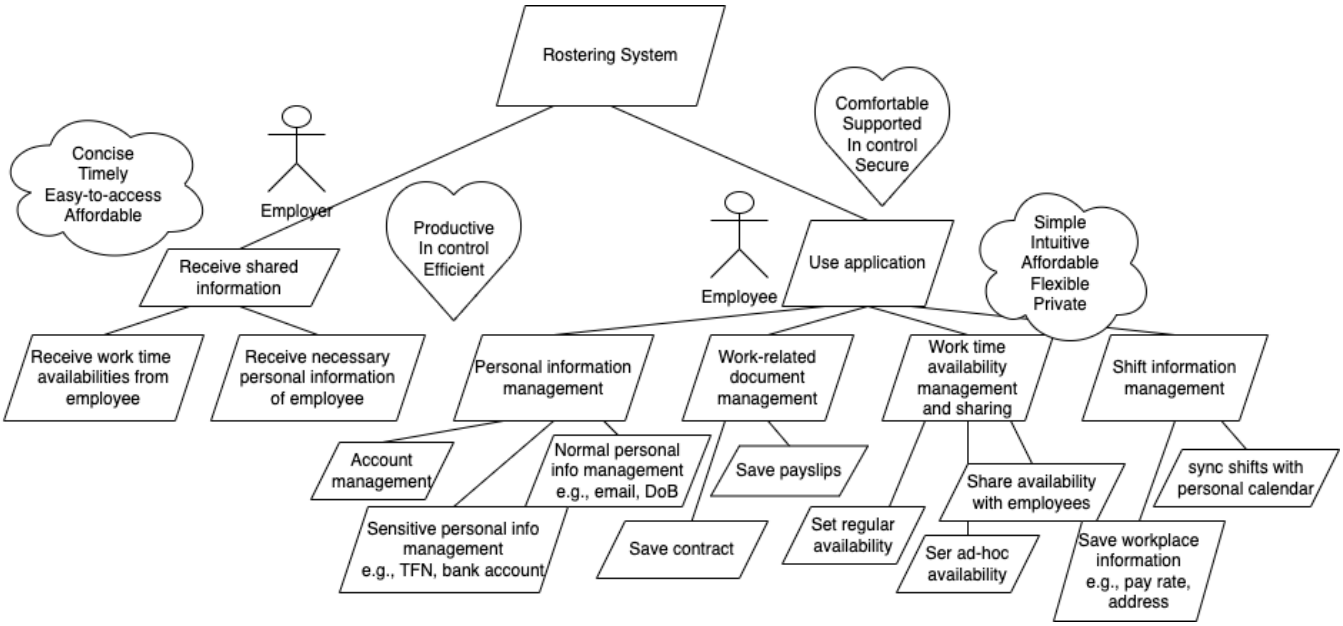
version 1.0

Who	Do	Be	Feel
• Employee with one or more jobs	• Share availability with employer(s) • See upcoming work shifts • Manage work shifts • See shifts information like time and work place • Manage work-related documents • Share personal information	• Simple • Intuitive • Affordable • Flexible • Private	• Comfortable • Supported • In control • Secure
• Employer	• See shared employee availabilities • See shared employee information	• Concise • Timely • Easy-to-access • Affordable	• Productive • In control • Efficient
• Developer	• Work to maintain / extend this project	• Easy to maintain • Easy to extend	• Manageable • Efficient • Confident

version 2.0 (Developer component is removed to stay consistent with goal model and personas. Developer is usually not considered in DBF tables.)

Who	Do	Be	Feel
Employee with one or more jobs	• Share availability with employer(s) • See upcoming work shifts • Manage work shifts • See shifts information like time and work place • Manage work-related documents • Share personal information	• Simple • Intuitive • Affordable • Flexible • Private	• Comfortable • Supported • In control • Secure
Employer	• See shared employee availabilities • See shared employee information	• Concise • Timely • Easy-to-access • Affordable	• Productive • In control • Efficient

Goal Model


Rostering System
Concise
Timely
Easy-to-access
Affordable
Employer
Comfortable
Supported
In control
Secure
Productive
In control
Efficient
Employee
Use application
Simple
Intuitive
Affordable
Flexible
Private
Receive shared information
Receive work time availabilities from employee
Receive necessary personal information of employee
Personal information management
Work-related document management
Work time availability management and sharing
Shift information management
Account management
Normal personal info management e.g., email, DoB
Sensitive personal info management e.g., TFN, bank account
Save payslips
Save contract
Set regular availability
Share availability with employees
Ser ad-hoc availability
sync shifts with personal calendar
Save workplace information e.g., pay rate, address

Personas

Version 2.0: We increase the diversity of our personas in age, ethnicity and disability.

Person	Background	Info	Goals	Frustrations
Eleanor	A proud Aboriginal Australian, Eleanor grew up listening to the stories of her ancestors. At 72, she's visually impaired due to age-related macular degeneration. Having spent years in the administrative department of a university, she took up a part-time role at the local community centre after her retirement.	• **Age**: 72 • **Ethnicity**: Aboriginal Australian • **Native Language**: English • **Disability**: Visual impairment • **Occupation**: Community centre event organiser	• Easily view and confirm her shifts for community events using a digital platform. • Receive timely notifications about any changes in the roster. • Provide her availability for events and view which slots are already filled.	• Difficulty navigating apps that are not designed for visually impaired users. • Confusion arising from last-minute shift changes without clear communication. • Struggling to coordinate with other volunteers due to unclear scheduling.
Noah	A high school student of Filipino-Australian descent, Noah embraces both his cultures. He has a part-time job at a local bookstore.	• **Age**: 16 • **Ethnicity**: Filipino-Australian • **Native Language**: English (from his Australian side) and Tagalog (from his Filipino side) • **Occupation**: High school student and part-time bookstore employee	• Swiftly check, confirm, or swap shifts at the bookstore using the app. • Receive alerts about available extra shifts, especially during weekends or holidays. • Input and view his unavailability (like exam days) in advance.	• Missing out on shifts due to not being informed in time. • Double-booking himself because of using multiple platforms for scheduling. • Not being able to easily communicate his availability to both employers.
Daisy	Born to Italian immigrants, Daisy grew up in Perth. At 45, she's hard of hearing. As a solicitor and a new mother, she occasionally takes on pro bono cases for community members.	• **Age**: 45 • **Ethnicity**: Italian-Australian • **Native Language**: English and Italian • **Disability**: Hard of hearing • **Occupation**: Solicitor	• Efficiently roster her team for different cases and court dates. • View all team members' availability in one place to avoid scheduling conflicts. • Easily update or modify the roster based on sudden court date changes or client meetings.	• Time wasted on back-and-forth communication with team members about their shifts. • Difficulty in finding a replacement when someone can't make it to their shift. • Challenges in integrating other tools or calendars with the rostering system.

User Persona Name

- Age: 72
- Ethnicity: Aboriginal Australian
- Native Language: English
- Disability: Visual impairment
- Occupation: Community centre event organiser

Goals

-
- Easily view and confirm her shifts for community events using a digital platform.
- Receive timely notifications about any changes in the roster.
- Provide her availability for events and view which slots are already filled.

Frustrations

-
- Difficulty navigating apps that are not designed for visually impaired users.
- Confusion arising from last-minute shift changes without clear communication.
- Struggling to coordinate with other volunteers due to unclear scheduling.

A proud Aboriginal Australian, Eleanor grew up listening to the stories of her ancestors. At 72, she's visually impaired due to age-related macular degeneration. Having spent years in the administrative department of a university, she took up a part-time role at the local community centre after her retirement.

Preferred platforms

PC

IOS

Android

""I need an app where I can clearly see my shifts and any changes. It should be simple and considerate of my visual challenges."

DO NEXT **Customer Journey Map**

See related templates

User Persona Name

- Age: 16
- Ethnicity: Filipino-Australian
- Native Language: English (from his Australian side) and Tagalog (from his Filipino side)
- Occupation: High school student and part-time bookstore employee

Goals

-
- Swiftly check, confirm, or swap shifts at the bookstore using the app.
- Receive alerts about available extra shifts, especially during weekends or holidays.
- Input and view his unavailability (like exam days) in advance.

Frustrations

-
- Missing out on shifts due to not being informed in time.
- Double-booking himself because of using multiple platforms for scheduling.
- Not being able to easily communicate his availability to both employers.

A high school student of Filipino-Australian descent, Noah embraces both his cultures. He has a part-time job at a local bookstore.

Preferred platforms

PC

IOS

Android

"My life's hectic enough. I just need an app that clearly tells me when I'm working and lets me flag my busy days."

DO NEXT **Customer Journey Map**

See related templates

User Persona Name

Goals

-
- Efficiently roster her team for different cases and court dates.
- View all team members' availability in one place to avoid scheduling conflicts.
- Easily update or modify the roster based on sudden court date changes or client meetings.

Frustrations

- Time wasted on back-and-forth communication with team members about their shifts.
- Difficulty in finding a replacement when someone can't make it to their shift.
- Challenges in integrating other tools or calendars with the rostering system.

Preferred platforms

PC

IOS

Android

- Age: 45
- Ethnicity: Italian-Australian
- Native Language: English and Italian
- Disability: Hard of hearing
- Occupation: Solicitor

Born to Italian immigrants, Daisy grew up in Perth. At 45, she's hard of hearing. As a solicitor and a new mother, she occasionally takes on pro bono cases for community members.

"I need a rostering tool that's as dynamic and fast-paced as my work. A clear view of everyone's schedule would make things so much smoother."

User Stories

Priority Estimation

Priority is assigned to each user story based on how crucial they are to a minimum viable product, as discussed with our client.

Some user stories with lower priority are excluded from our plan. See details: Out-of-Scope Features

Type	Definition
CORE	Must-have features that we need to prioritize
HIGH	Important features that we need to focus on
MEDIUM	Good-to-have features we will work on where possible
LOW	Optional additional features that we might be able to work on
Out of Scope	Out of Scope

Effort Estimation

The effort we need as a team to complete a user story is approximated by story points (i.e. how many days it would take to complete the user story).

At the initial planning stage, we assume that a higher effort suggests more time will be needed to research / design / implement or conduct related work, potentially due to the following reasons:

- The task is expected to require a lot of code to implement
- At this stage, we are not very familiar with how to implement the task or integrate it with other system of our product

Type	Definition	Story Points
XS	Takes very little effort to implement	1 ~ 2
S	Takes some effort to implement	3 ~ 5
M	Takes moderate effort to implement	7 ~ 11
L	Takes considerable effort to implement	13 ~ 17
XL	Takes huge efforts to implement	19+

V1.0 User Stories

Epic Category	ID	Feature	As a …	I want to …	So that …	Priority	Effort
Account Management	1	Register (employee)	Employee	Register for a new employee account	I can save my information in the system with an uniquely identifiable account	HIGH	M
	2	Log in (employee)	Employee	Log in to my account	I can access the application as a validated user in employee mode	HIGH	M
	3	Normal Password Reset	Employee	Reset my password if I would like to have a new password	I can log in to the account with a new password	HIGH	M
	4	Forgotten Password Reset	Employee	Reset my password if I forgot the password	I can log in to the account with a new password	HIGH	M
	5	Edit Account	Employee	Edit my account information	I can change my account details like username or email address	HIGH	S
	6	Delete my Account	Employee	Delete my app account and get a warning of risks and a 30 days of	I can stop sharing my information with the application	MEDIUM	L
	7	Register (employer)	Employer	Register for a new employer account	I can get started with an employer account	Out of Scope	M
	8	Log in (employer)	Employer	Log in to my account	I can access the application as a validated user in employer mode	Out of Scope	M
Availability Management	9	Set ad-hoc availabilities	Employee	Set when I'm able to work (ad-hoc work availability)	I can let my employer(s) know when I'm available for work	CORE	L
	10	Set standard availabilities	Employee	Set my standard work days/hours availability	I can let my employer(s) know when I'm usually available for work	CORE	M
	11	Share availabilities	Employee	Share my availability times with employer(s) via SMS or Email	I can quickly let my employer(s) know when I'm available for work	HIGH	L

	12	View availabilities	Employer	View the availability of employee in real-time on the web portal	I can check the availability of employees timely	Out of Scope	XL
Information Management	13	Edit personal information	Employee	Edit my personal details on my Profile page	I can change my personal details if I want	CORE	S
	14	Manage identity documents	Employee	Upload my identity documents	I can provide the proof my identify	LOW	S
	15	Manage qualification documents	Employee	Upload my qualification documents	I can provide the proof my qualifications	LOW	S
	16	Manage employment contract	Employee	Upload my employment contract	I can refer to it when needed	HIGH	S
	17	Share information with selected employers	Employee	Share my selected personal details to selected employer (s)	I can let the employer see the details that I want to share, and protect my privacy	MEDIUM	M
Work Shift Management	18	Work shift viewing	Employee	See my allocated work shift information	I can have a visualised view of my planning	CORE	XL
	19	Sync work shifts with personal calendar	Employee	Import or output my upcoming work shifts into or from my personal calendar	I can synchronise my shifts with my personal calendar	LOW	XL
	20	Add, edit work shift information	Employee	Add, edit work shift information	I can record when I need to work	HIGH	S
	21	View all work shift requests	Employee	View all work shift requests from employersView all work shift requests from employers	I can view the available shifts requests sent by the employer	Out of Scope	L
	22	Accept or refuse work shift arrangements	Employee	Accept or refuse work shift arrangements from my employer(s)	I can choose when I wish to work base on my circumstances	Out of Scope	XL
	23	Notify employer when work shift changed	Employee	Notify the employer when I edit or cancel my shift with a note	I can let the employer know when I wish to make changes to my work shifts	Out of Scope	XL
Workplace Management	24	Workplace management	Employee	Add, edit and view different workplaces	I can check the detail of different places for different part-time jobs	HIGH	L
	25	View workplace pay details	Employee	View the pay detail of each work	I can check the pay detail of each part-time job such as the salary, next pay data, and so on	LOW	XL
Pay Management	26	View total earnings	Employee	View my total earnings based on the each salary bill	I can get the a consolidated view of total earnings to assist in identifying any upcoming tax implications	LOW	L
	27	Add pay slips	Employee	Load pay slips	I can add the salary details for each part-time job	Out of Scope	L
	28	Choose different rates based on special dates	Employee	Have choices of different rates (standard, holiday, overtime, casual loading, etc) (daily, hourly) to select pay type	Can have a flexible way to calculate my estimated earnings	Out of Scope	M
App	29	Configure notifications	Employee	Configure app notifications	I can be reminded when some events occur	MEDIUM	M
	37	Home screen	Employee	Have access to a home page	I can easily access other functions of the app from a centralized menu	HIGH	L
Verification	30	2FA	Employer	Receive a verification code via SMS/Email when logging into my account	I can have two-factor authentication to protect the security of my account	HIGH	M
	31	Certificate verification	Employer	Validate certificate (for example, license for truck driver) through the third party	I can make sure the reliability of employee on the app	Out of Scope	L
	32	Identity verification	Employer	Upload my Employer identity currency check (License and WWCC) for validation	I can provide the proof of the employment	Out of Scope	L
Employer Features	33	Employer portal	Employer	Access a web portal of the application (or login the app as an employer, to use the app in employer mode)	I can see and manage rosters on my computer	Out of Scope	XL
	34	Employee filtering	Employer	Identify employee availability based on filtered requirements such as current allocated shifts, credentials, preference etc	I can find specific employees based my needs.	Out of Scope	L
	35	Push available shifts to available employees	Employer	Push available shifts to available employees who can then accept or reject the shift in real time	I can send my request to employees	Out of Scope	L
	36	Manage work shift rosters	Employer	Manage work shift rosters for all staff	I can have a rostering calendar and share it with all staff	Out of Scope	XL

V2.0 User Stories

Epic Category	ID	Feature	As a …	I want to …	So that …	Priority	Effort	Justification
Account Management	1	Register (employee)	Employee	Register for a new employee account	I can save my information in the system with an uniquely identifiable account	HIGH	M	The development of this function is not hard to complete but it is the basic functions of an app.
	2	Log in (employee)	Employee	Log in to my account	I can access the application as a validated user in employee mode	HIGH	M	The development of this function is not hard to complete but it is the basic functions of an app.
	3	Normal Password Reset	Employee	Reset my password if I would like to have a new password	I can log in to the account with a new password	HIGH	M	The development of this function is not hard to complete but it is the basic functions of an app.

	4	Forgotten Password Reset	Employee	Reset my password if I forgot the password	I can log in to the account with a new password	HIGH	M	The development of this function is not hard to complete but it is the basic functions of an app.
	5	Edit Account	Employee	Edit my account information	I can change my account details like username or email address	HIGH	S	The development of this function is not hard to complete but it is the basic functions of an app.
	6	Delete my Account	Employee	Delete my app account and get a warning of risks and a 30 days of	I can stop sharing my information with the application	MEDIUM	L	The development of this function is a bit complicated but it's one of the core requirement at this stage therefore we decided to put it at a latter sprint.
	7	Register (employer)	Employer	Register for a new employer account	I can get started with an employer account	Out of Scope	M	Most of the functions on the employer side is out of scope at this stage but the development is not complicated.
	8	Log in (employer)	Employer	Log in to my account	I can access the application as a validated user in employer mode	Out of Scope	M	Most of the functions on the employer side is out of scope at this stage but the development is not complicated.
Availability Management	9	Set ad-hoc availabilities	Employee	Set when I'm able to work (ad-hoc work availability)	I can let my employer(s) know when I'm available for work	CORE	L	The development of this function is a bit complicated but it is one of the core requirement at this stage.
	10	Set standard availabilities	Employee	Set my standard work days/hours availability	I can let my employer(s) know when I'm usually available for work	CORE	M	The development of this function is not complicated and it is one of the core requirement at this stage.
	11	Share availabilities	Employee	Share my availability times with employer(s) via SMS or Email	I can quickly let my employer(s) know when I'm available for work	HIGH	L	The development of this function is a bit complicated but it is one of the core requirement at this stage.
	12	View availabilities	Employer	View the availability of employee in real-time on the web portal	I can check the availability of employees timely	Out of Scope	XL	Most of the functions on the employer side is out of scope at this stage and this function requires a web portal which is complicated to develop.
Information Management	13	Edit personal information	Employee	Edit my personal details on my Profile page	I can change my personal details if I want	CORE	S	The development of this function is simple but it is one of the core requirement at this stage.
	14	Manage identity documents	Employee	Upload my identity documents	I can provide the proof my identify	LOW	S	The development of this function is simple but it is not the core function of this app.
	15	Manage qualification documents	Employee	Upload my qualification documents	I can provide the proof my qualifications	LOW	S	The development of this function is simple but it is not the core function of this app.
	16	Manage employment contract	Employee	Upload my employment contract	I can refer to it when needed	HIGH	S	The development of this function is not complicated but it is crucial from the client.
	17	Share information with selected employers	Employee	Share my selected personal details to selected employer(s)	I can let the employer see the details that I want to share, and protect my privacy	MEDIUM	M	The development of this function is not complicated but it is required by the client.
Work Shift Management	18	Work shift viewing	Employee	See my allocated work shift information	I can have a visualised view of my planning	CORE	XL	The development of this function is hard but it is one of the core requirement at this stage.
	19	Sync work shifts with personal calendar	Employee	Import or output my upcoming work shifts into or from my personal calendar	I can synchronise my shifts with m y personal calendar	LOW	XL	The feasibility of the development of this function is arguable and this function is not a core requirement.
	20	Add, edit work shift information	Employee	Add, edit work shift information	I can record when I need to work	HIGH	S	The development of this function is not complicated and it is one of the core requirement at this stage.
	21	View all work shift requests	Employee	View all work shift requests from employers	I can view the available shifts requests sent by the employer	Out of Scope	L	This function requires user interface from the employer side and most of the functions on the employer side is out of scope at this stage.
	22	Accept or refuse work shift arrangements	Employee	Accept or refuse work shift arrangements from my employer(s)	I can choose when I wish to work base on my circumstances	Out of Scope	XL	This function requires user interface from the employer side and most of the functions on the employer side is out of scope at this stage.
	23	Notify employer when work shift changed	Employee	Notify the employer when I edit or cancel my shift with a note	I can let the employer know when I wish to make changes to my work shifts	Out of Scope	XL	This function requires user interface from the employer side and most of the functions on the employer side is out of scope at this stage.
Workplace Management	24	Workplace management	Employee	Add, edit and view different workplaces	I can check the detail of different places for different part-time jobs	HIGH	L	The development of this function requires many components and it is one of the core requirement at this stage.
	25	View workplace pay details	Employee	View the pay detail of each work	I can check the pay detail of each part-time job such as the salary, next pay data, and so on	LOW	XL	The logic of this function is complicated and it is not a core requirement at this stage.

Pay Management	26	View total earnings	Employee	View my total earnings based on the each salary bill	I can get the a consolidated view of total earnings to assist in identifying any upcoming tax implications	LOW	L	The development of this function requires user story 25 to be completed and it is not a core requirement at this stage.
	27	Add pay slips	Employee	Load pay slips	I can add the salary details for each part-time job	Out of Scope	L	This function is not a core requirement and its feasibility is arguable.
	28	Choose different rates based on special dates	Employee	Have choices of different rates (standard, holiday, overtime, casual loading, etc) (daily, hourly) to select pay type	Can have a flexible way to calculate my estimated earnings	Out of Scope	M	This function is not a core requirement and it is complicated to complete.
App	29	Configure notifications	Employee	Configure app notifications	I can be reminded when some events occur	MEDIUM	M	This is the basic function of an app and the development is not hard.
	37	Home screen	Employee	Have access to a home page	I can easily access other functions of the app from a centralized menu	HIGH	L	This is the basic function of an app and the development requires multiple components.
Verification	30	2FA	Employer	Receive a verification code via SMS /Email when logging into my account	I can have two-factor authentication to protect the security of my account	HIGH	M	This is a core requirement in terms of security consideration and the development is not hard.
	31	Certificate verification	Employer	Validate certificate (for example, license for truck driver) through the third party	I can make sure the reliability of employee on the app	Out of Scope	L	This is out of scope at this stage and the development requires 3rd party collaboration.
	32	Identity verification	Employer	Upload my Employer identity currency check (License and WWCC) for validation	I can provide the proof of the employment	Out of Scope	L	This is out of scope at this stage and the development requires 3rd party collaboration.
Employer Features	33	Employer portal	Employer	Access a web portal of the application (or login the app as an employer, to use the app in employer mode)	I can see and manage rosters on my computer	Out of Scope	XL	Most of the functions on the employer side is out of scope at this stage and this function is complicated to develop.
	34	Employee filtering	Employer	Identify employee availability based on filtered requirements such as current allocated shifts, credentials, preference etc	I can find specific employees based my needs.	Out of Scope	L	Most of the functions on the employer side is out of scope at this stage and this function is complicated to develop.
	35	Push available shifts to available employees	Employer	Push available shifts to available employees who can then accept or reject the shift in real time	I can send my request to employees	Out of Scope	L	Most of the functions on the employer side is out of scope at this stage and this function is complicated to develop.
	36	Manage work shift rosters	Employer	Manage work shift rosters for all staff	I can have a rostering calendar and share it with all staff	Out of Scope	XL	Most of the functions on the employer side is out of scope at this stage and this function requires a whole new system in the app.

In-Scope Features

According to the 36 user stories under the "user story page", there are many detailed in-scope features that we identified as in-scope features.

1. Account Management

1. Register a new account:
   1. Develop a sign up functionality to allow employee to register an account.
   2. App should provide a application form to collect nessary information for employee to set up an account such as email address, user name, and password
   3. When employee create a new account, system should be able to generate a unique id for each employee (user)
   4. One email should be able to register only one account
   5. The registraion data (email, password ……) should be saved into database
   6. When employee use this app for the first time, the accounr registration page should be the first page show to the employee
   7. Once the user/employee complete the registration process, user/employee should be directed to the home page
2. Edit Account information:
   1. Develop a edit function to allow employee to update their account details which include passwrod, email addresss and user name
   2. When employee update account details, it should also update the account details stored in database.
3. Delete account:
   1. Develop a delete functionality that can allow employee to delete account
   2. Employee's accuount information should be kept in database for 30 days, which can allow employee to recover their data since they delete account.
   5. The registraion data (email, password ……) should be saved into database
4. Exit account:
   1. Develop a exit functionality that can allow employee to exit account

2. Home page:

1. Home page set up:
   1. Develop a home page for the app that could possibly contain navigator button to other page.
   2. User/employee's profile icon or user name should be also reflected in somewhere on the home page.

3. Availability Management

1. Set ad-hoc availabilities:
   1. Develop an ad-hoc availabilities page to allow employer to know employee's schedual.
   2. Ad-hoc availabilities page should be linked to the home page
   3. Develop an functionality to add single availible date on this page.
   4. Develop an functionality to add availible time slut for a date (one date can have multiple available time slut) on this page.
   5. Develop an functionality to add date range on this page.
   6. Develop an functionality or a new page to allow employee to edit availability information on this page.
   7. Availibility information (includes availible dates, availible time slut, and the corresponding notes) should be stored into database and updated in real-time
   8. All availability information should be real-time information, which means that data on this page should be always up to date
2. Set standard availabilities:
   1. Develop a functionality to allow employee to set recurrent available date for example, some employee might want to set every sunday as available date
3. Share availabilities:
   1. Develop a functionality to allow employee to share their availibility information to other people (possibily employer) via sms or email

4. Information Management

1. Create a page that contains user/employee's profile information
   1. Develop a profile page that could allow employee to view their own personal information
   2. This profile page should be linked to the home page
2. Edit personal information
   1. Develop another page for employee to edit personal information
   2. Develop a function to upload identity documents
   3. Personal information should be stored in database and could be updated when user/employee edit it in real-time
3. Manage identity documents:
   1. Develop a function to upload identity documents
4. Manage qualification documents:
   1. Develop a function to upload qualification documents
5. Manage employment contract :
   1. Develop a function to upload contract documents
6. Share information with selected employers:
   1. Develop a function to share any document or information to other people (possibly employer)

5. Work Shift Management

1. Work shift viewing:
   1. Develop an work shift page to allow employee to record their own schedule and display their schedule
   2. This work shift page should be linked to the home page
2. Sync work shifts with personal calendar:
   1. Develop an functionality to integrate work shift information/schedule into user/employee's local calendar

3. Add, edit work shift information:
    1. Develop add function to allow emloyee add new shift on a selected date and this function should enable employee to add multiple shifts on the same date (employee can add time slut, notes, employer name…… to a shift)
    2. Develop edit function to allow emloyee to change information on a shift
    3. Shift information should be stored into database and should be updated when user/employee edit it

6. Workplace management:

1. Add, edit and view different workplaces:
    1. Develop a workplace page that could show different workplace for employee
    2. Develop an add function to add different workplace on this page, possibly develop a new page that could allow employee to enter their name, ABN, address, contact name, phone number, and email address
    3. Develop an edit function to edit existing information of workplace on this page
    4. Workplaces information should be stored into database and should be updated when user/employee edit it
    5. This workplace page should be linked to the home page
2. View the pay detail of each work:
    1. Develop a function to allow employee to enter pay rate for a work
    2. Develop a function to calculate the payroll for employee automedically

7. Pay Management

1. View total earnings:
    1. Develop a functionality to calculate total earnings based on the each salary bill automedically

8. App:

1. Configure notifications:
    1. Develop a functionality to send notification
2. Homepage:
    2. Create a homepage that can navigate to each other page

9. Verification

1. 2FA:
    1. Develop a function to end verification code to employee by sms/email when they try to log in

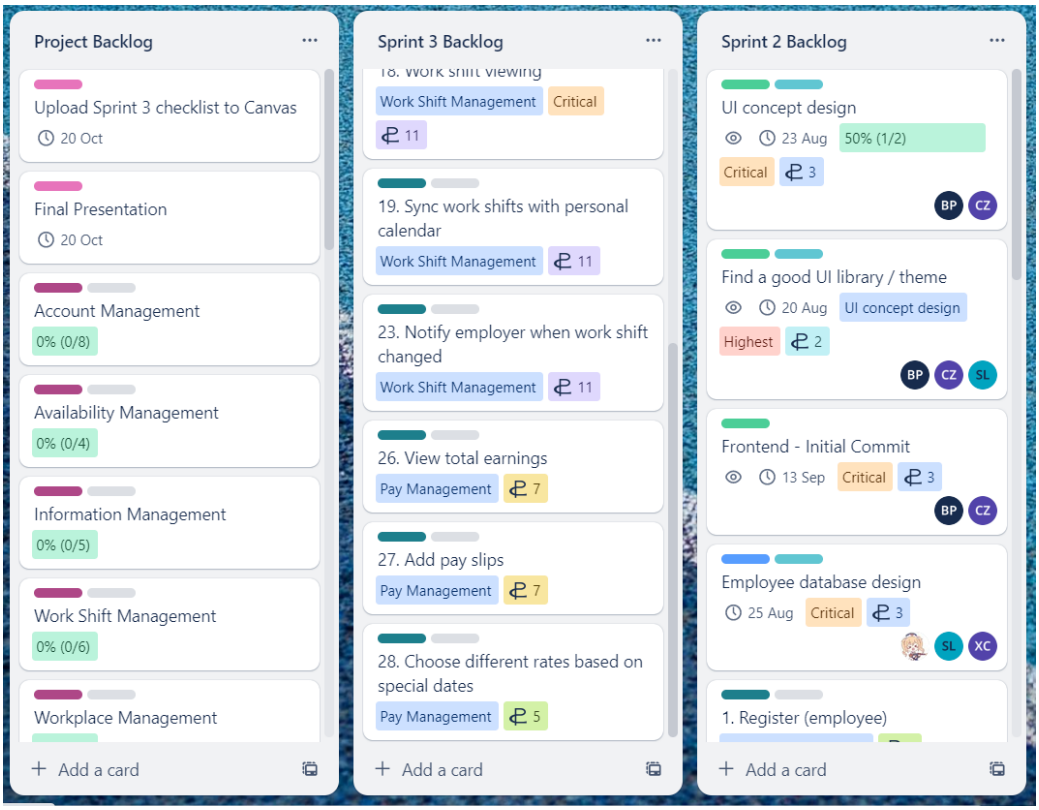
Out-of-Scope Features

According to the 36 user stories under the "user story page" and the discussion between the client, our team identified some out-scope features.

Epic Category	User Story ID	Functionality	Description
Account Management	7	Register (employer)	Create another page or option for an employer to registrate an account
	8	Log in (employer)	Create another portal (employer mode rather than employee mode) for employer to log in
Availability Management	12	View availabilities (employer portal)	Create a page that can dispaly the available time of employee, load data from database to display available time of employee
Work Shift Management	21	View all work shift requests from employer	When employer is interest on a shift from an employee, they can send request to employee, develop a page to display all the request from differnet employer
	22	Accept or refuse work shift arrangements	Create a button to accept or decline the request from employer, implement the backend features to support this
	23	Notify employer when work shift changed	The employer should be notified when they have been declined or accpeted
Pay Management	27	Add pay slips	Create a funcion or a page to allow user/employee to add their pay rate for each shift or job
	28	Choose different rates based on special dates	Create a funcion to allow user/employee to set different rate on different date (for example, regular day, weekend, holiday), create a function that could automatically calculate the earning based on the rate
Verification	31	Certificate verification	Develop a page for employee to upload cerficiate and it can be linked to a thrid party validation (for example, truck driver license)
	32	Identity verification	Develop a page to for employee to upload ID and this page could be linked to the third party authentication (government authentication)
Employer Features	33	Employer portal	For employer, develop another new portal other than employee
	34	Employee filtering	Develop a function for employer portal, which can filter employee availability based on different conditions such as current allocated shifts, credentials, preference etc
	35	Push available shifts to available employees	Develop a page to for employee to push available shifts to available employees who can then accept or reject the shift in real time
	36	Manage work shift rosters	Develop a new page to store work shift rosters for all staff and display it

Project Management

Sprint Planning



https://trello.com/b/DsZtZFFJ/comp90082-ro

Sprint 2 Planning

Vision

Vision	Vision Date	Editor	Description
1.0	18 Aug 2023	Chenxi Zhang	Based on the product backlog, decide the features to be completed and delivered in Sprint 2, assigning priority to each feature.

Sprint Goals

Project: RO-Bluering	Sprint: 2
Goal: Set up the development environment, complete all identified and allocated user stories for Sprint 2, and carry out the development and testing for each feature. Accomplish the delivery to the client.	

Estimated Stories & Level Task Decomposition

Epic Categories	Story ID	Feature	Story Point	Participant	Priority	Status
Account Management	1	Register(Employee)	2	Xuechun Chen	HIGH	Scheduled
	2	Log in (employee)	2	Xuechun Chen	HIGH	Scheduled
	3	Normal Password Reset	2	Xuechun Chen	HIGH	Scheduled
	4	Forgotten Password Reset	2	Xuechun Chen	HIGH	Scheduled
	5	Edit Account	2	Xuechun Chen	HIGH	Scheduled
	6	Delete my Account	2	Xuechun Chen	MEDIUM	Scheduled
Availability Management	9	Set ad-hoc availabilities	8	Bo Peng	CORE	Scheduled
	10	Set standard availabilities	5	Bo Peng	CORE	Scheduled
	11	Share availabilities	8	Bo Peng	HIGH	Scheduled
	13	Edit personal information	3	Chenxi Zhang	CORE	Scheduled
Work Shift Management	20	Add, edit work shift information	7	WenxuanChen	HIGH	Scheduled
Workplace Management	24	Workplace management	6	Chenxi Zhang	HIGH	Scheduled
	25	View workplace pay details	8	Chenxi Zhang	LOW	Scheduled
App	37	Home Screen	4	Sichen Lu	MEDIUM	Scheduled

Sprint 3 Planning

Version

Version	Version date	Editor	Description	End time
1.0	22 Sep 2023	Chenxi Zhang	Confirm the features to be completed in Sprint 3, establish priorities for each feature and assign responsible developers. Set the goals for Sprint 3 to clearly outline the deliverables expected at the conclusion of the sprint. Based on the experience gained during the development process of Sprint 2 and the review of the entire development cycle of Sprint 2, we need to re-plan for Sprint 3. We will update the features that need to be completed in Sprint 3, and re-evaluate each feature's completion time and priority.	20 Oct 2023

Sprint Goal

Project: RO-Bluering	**Sprint: 3**
Goal: By establishing a server connection, utilize cloud services to replace local storage for storing all data within the app. Complete the development of all features required for Phase One and perform testing tasks on all functionalities to ensure product quality. Deliver the product to the client on time. Improve UI design and address some of the issues left over from Sprint 2 in this sprint. Simultaneously create local storage, prioritize reading data from local sources, with the cloud serving merely as a data backup. Every features needs to be considered for cyber security, and ethical considerations.	

Estimated Stories & Level Task Decomposition

The user stories that highlighted by blue colour is the user stories that we haven't complete in this sprint.

Epic	Story ID	Feature	Story Point	Participant	Priority	Status
Information Management	14	Manage identity document	3	Bo Peng	LOW	Scheduled
	15	Manage qualification documents	3	Xuechun Chen	LOW	Scheduled
	16	Manage employment contract	3	Xuechun Chen	HIGH	Scheduled
	17	Share information with selected employers	5	Bo Peng	MEDIUM	Scheduled
Work Shift Management	18	Work shift viewing	10	Wenxuan Chen/Sichen Lu	CORE	Scheduled
	19	Sync work shifts with personal calendar	10	Chenxi Zhang	LOW	Scheduled
Pay Management	26	View total earnings	7	Chenxi Zhang	LOW	Scheduled
	27	Add pay slips	7	Sichen Lu	MEDIUM	Scheduled
Account Management	5	Edit Account	2	Xuechun Chen	HIGH	Scheduled
	6	Delete my Account	2	Xuechun Chen	MEDIUM	Scheduled
Availability Management	11	Share availabilities	8	Bo Peng	HIGH	Scheduled
Work Shift Management	20	Edit shift information	3	Wenxuan Chen	HIGH	Scheduled
Workplace Management	25	View workplace pay details	8	Chenxi Zhang	LOW	Scheduled

Sprint 4 Planning

Version

Version	Version date	Editor	Description	End time
1.0	20 Oct 2023	Chenxi Zhang	Confirm the features to be completed in Sprint 4, establish priorities for each feature, and assign responsible developers. Set the goals for Sprint 4 to clearly outline the deliverables expected at the conclusion of the sprint. Based on the experience gained during the development process of previous Sprints and the review of the entire development cycle, we optimized the planning for Sprint 4. In addition, there are several tasks left from Sprint 3 due to the limited development time, we move these tasks to Sprint 4.	3 Nov 2023

Sprint Goal

Project: RO-Bluering	Sprint: 4

Goal:

Sprint 4 marks the final development phase, where all functionalities must be completed and tested. Comprehensive documentation detailing the entire development process should also be maintained. For the App, the following goals need to be accomplished in Sprint 4:

1. Achieve all tasks including both the tasks from Sprint 4 as well as leftover tasks from Sprint 3
2. Improve UI design and unify the UI style
3. Standardize some validation feedback effects and make the entire App's functional presentation more consistent
4. Reconstruct the code. Strive to separate front-end and back-end code, reduce code redundancy, and ensure code reusability
5. Merge code and fix conflicts

In addition, ensure that two meetings with the client are held before development concludes. One meeting is to showcase the final results of the App and to record the client's feedback on the product. Based on this feedback, the App will undergo a final round of optimization, with the outcomes presented in the last meeting. Finally, conclude all development on the product and deliver all results on time.

Estimated Stories & Level Task Decomposition

The user stories that are highlighted in blue color are the user stories that we haven't completed in the previous sprint.

Epic	Story ID	Feature	Story Point	Participant	Priority	Status
Work Shift Management	19	Sync work shifts with personal calendar	10	Chenxi Zhang	LOW	Scheduled
Availability Management	11	Share availabilities	8	Bo Peng	HIGH	Scheduled

Sprint Review

Sprint 2 Review

In this sprint 2 phase, we have 14 user stories need to be done and the duration of sprint 2 last for around 1 month.

The user stories are as the chart below:

Estimated Stories & Level Task Decomposition

The user stories that highlighted by blue colour are the user stories that we haven't complete in this sprint.

Epic Categories	Story ID	Feature	Story Point	Participant	Priority	Status
Account Management	1	Register(Employee)	2	Xuechun Chen	HIGH	Scheduled
	2	Log in (employee)	2	Xuechun Chen	HIGH	Scheduled
	3	Normal Password Reset	2	Xuechun Chen	HIGH	Scheduled
	4	Forgotten Password Reset	2	Xuechun Chen	HIGH	Scheduled
	5	Edit Account	2	Xuechun Chen	HIGH	Scheduled
	6	Delete my Account	2	Xuechun Chen	MEDIUM	Scheduled
Availability Management	9	Set ad-hoc availabilities	8	Bo Peng	CORE	Scheduled
	10	Set standard availabilities	5	Bo Peng	CORE	Scheduled
	11	Share availabilities	8	Bo Peng	HIGH	Scheduled
	13	Edit personal information	3	Chenxi Zhang	CORE	Scheduled
Work Shift Management	20	Add, edit work shift information	7	WenxuanChen	HIGH	Scheduled
Workplace Management	24	Workplace management	6	Chenxi Zhang	HIGH	Scheduled
	25	View workplace pay details	8	Chenxi Zhang	LOW	Scheduled
App	37	Home Screen	4	Sichen Lu	MEDIUM	Scheduled

These 13 user stories contain 2+2+2+2+2+2+8+5+8+3+7+6+8+4 = 61 story points.

At the very beginning, we assign each epic categories into each team member.

In the very first week of sprint 2, everyone finish UI set up, and our team plan to set up a local App first and then move on to store data into database.

In the second week of sprint 2, some team members have to pay more time on other subjects, which cause the progress of development slow down. When it comes to the end of week 2 (sprint 2), our group start to investigate how to store our data into database (this function is actually belongs to sprint 3 but we think it is better to define the database structure as early as possible because database structure can be difficult to adjust when it comes to the end of development process if we find out the database structure is not very good), what's the database structure, what type of data we are going to store and how to load data from database to the App.

While at the end of sprint 2, we didn't fully finish all user stories we suppose to finish which indlue the following user stories:
1. Edit function in work shift information
2. The function of delete my Account
3. The function of share availabilities
4. The function of display workplace pay details

But we finish the database part in terms of storing data and loading data.

Client Feedback

We showed the prototype to the client at the end of sprint 2. The client acknowledged our effort and gave detailed feedbacks regarding each features we implemented, which can be found in Project Management/Meeting/Client Meeting Notes.

Overall, the development is on schedule and the prototype mostly meets client's requirements with fine-tunings of details.

Sprint 3 Review

In this sprint 3 phase, we have 13 user stories that need to be done and the duration of sprint 3 lasts for around half a month.

The user stories are as the chart below:

Estimated Stories & Level Task Decomposition

The user stories that are highlighted by blue colour are the user stories that we haven't completed in this sprint.

Epic	Story ID	Feature	Story Point	Participant	Priority	Status
Information Management	14	Manage identity document	3	Bo Peng	LOW	Scheduled
	15	Manage qualification documents	3	Xuechun Chen	LOW	Scheduled
	16	Manage employment contract	3	Xuechun Chen	HIGH	Scheduled
	17	Share information with selected employers	5	Bo Peng	MEDIUM	Scheduled
Work Shift Management	18	Work shift viewing	10	Wenxuan Chen/Sichen Lu	CORE	Scheduled
	19	Sync work shifts with personal calendar	10	Chenxi Zhang	LOW	Scheduled
Pay Management	26	View total earnings	7	Chenxi Zhang	LOW	Scheduled
	27	Add pay slips	7	Sichen Lu	MEDIUM	Scheduled
Account Management	5	Edit Account	2	Xuechun Chen	HIGH	Scheduled
	6	Delete my Account	2	Xuechun Chen	MEDIUM	Scheduled
Availability Management	11	Share availabilities	8	Bo Peng	HIGH	Scheduled
Work Shift Management	20	Edit shift information	3	Wenxuan Chen	HIGH	Scheduled
Workplace Management	25	View workplace pay details	8	Chenxi Zhang	LOW	Scheduled

These 13 user stories contain 3+3+3+5+10+10+7+7+2+2+8+3+8 = 71 story points.

All tasks that needed to be completed in Sprint 3 are displayed in the chat above. Each task has been marked with its status. some tasks have been completed, while others have been postponed and moved to Sprint 4. As illustrated in the above figure, there's a noticeable reduction in delayed tasks for this development cycle. The team excelled in time management and in communicating with the client, clearly identifying essential tasks and those that could be postponed or discarded. Some challenging tasks with lower priority have been deferred to Sprint 4, as by then, assignments from other subjects will have been completed, providing the team ample time to collaboratively address issues. In conclusion, we anticipate fewer challenges in Sprint 4.

Client Feedback

We held a meeting with the client about the unfinished features at the end of Sprint 3 and agreed on adjusting priorities and re-organizing the development focus in Sprint 4. The Sprint Planning for Sprint 4 are decided in accordance with our agreements with the client.

Sprint 4 Review

In this sprint 4 phase, There are only two specific features that need to be implemented, the rest of the tasks in Sprint 4 are more about testing and optimization last for around half a month.

The user stories are as the chart below:

Estimated Stories & Level Task Decomposition

The user stories that are highlighted in blue color are the user stories that we haven't completed in the previous sprint.

Epic	Story ID	Feature	Story Point	Participant	Priority	Status
Work Shift Management	19	Sync work shifts with personal calendar	10	Chenxi Zhang	LOW	Scheduled
Availability Management	11	Share availabilities	8	Bo Peng	HIGH	Scheduled

These 13 user stories contain 10 + 8 = 18 story points.

The team has become more coordinated and efficient. The content for optimization as well as the development content have all been fully tested and have completed code review. Overall, Sprint 4 was completed very well.

Client Feedback

The client is pleased with the level of completion of the project. During our meeting, we presented a demo of the final product, which demonstrated the full range of features. The client acknowledged and expressed satisfaction with the functionality we have achieved. well done:)

Sprint Retrospective

Sprint 2 Retrospective

What went well?

We have made very clear plan or each week and everyone has strictly followed our coding conventions and workflow. It make our team has very good progress for each week.

What did not go so well?

At this stage, since each member was developing their own part (basically each member would work individually), each member are keen to use different approach to store data into database because the data they want to store different types of data, and store data in different format. For example, some members want to store nested data while some members want to store easily and load easily from database. Our team notice that there are some disagreements between each member, and we define a format of how to store data. In week 3 from sprint 2, we find out there is a threat of database structure as we dig deeper in database because first of all we might know a little about what other member is doing but we might not know deeply what they exactly doing to store data and unfortunately our database structure defined at week 2 doesn't work well for everyone to store data into database and might be inconvenient to load data from database. At the middle of week 3, we spent some time to discuss about the database structure and learn exactly what other member plan to store and try to figure out a good database structure that could benefit every member or a database structure that could be accepted by every member. When it comes to week 4, everyone already has the product there but not fully done, some member might be stuck in loading data from database, some people might be stuck in making synchronous update on different screen, some member was stuck in reflect data on the App.

What actions need to be taken to improve?

Even though we have make clear plan for each week but our members are actually working alone on their own part. And thus when it comes to storing data into database, we have conflict. We are good but not good enough, we should understand what we should do and also we should be foreseeable to try to figure out what we should do as a team before we start doing our own work. We would like to work collaboratively with each other so that even if we miss out something very important at the very beginning, we could still find it out and make proper adjustment.

Sprint 3 Retrospective

What went well?

Overall, the development process of Sprint 3 was much smoother compared to Sprint 2. With the experience gained from previous development, we've become much more familiar with the use of team development tools and the entire process from development to code merging.

At the beginning of this development phase, based on the issues encountered during the development in Sprint 2, our team made several changes. First and foremost regarding the database, we utilized Firebase for data storage. Given that the data format required for each person's task varies, we revamped the database structure prior to development. This adjustment facilitates easier data retrieval, especially when there's interdependency among functionalities. Additionally, we undertook a code refactoring process. All reusable codes were relocated to the 'component' directory. For instance, the data storage code can be repurposed; for different data, only the storage path needs modification. This approach effectively reduces code redundancy, makes the code clearer, and will aid developers in the next semester to familiarize themselves with the code for continued development.

Each member can work very hard, everyone has good planning and know exactly what they are going to finish in current sprint.

What did not go so well?

Our team under-estimate the difficulty of some user stories such as user story 19 and 11. Our team has to re-negotiate with the client since we might not have enough time to finish it on time.

What actions need to be taken to improve?

Due to some mistakes made previously, we need to take extra care with what we need to do in the next sprint. Since we have already discussed with our client, and our client require us to finish user story 19 and 11, which will be the objective in the next sprint. We will make it by good attitude and good planning.

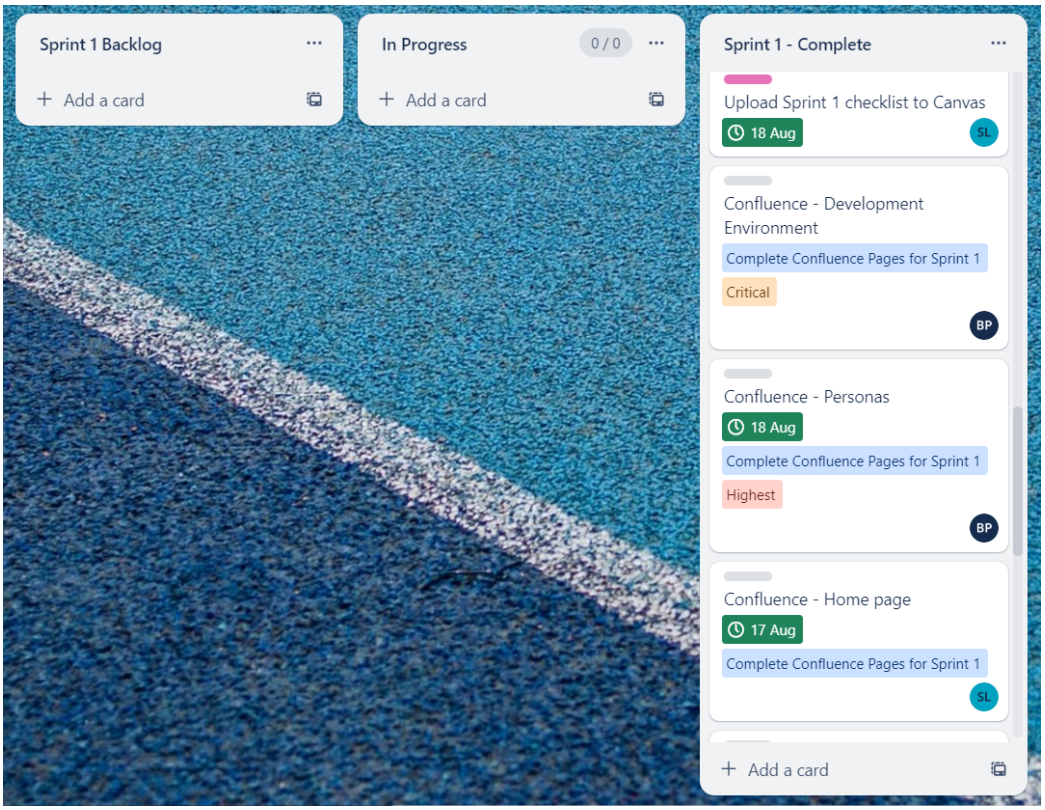
Team Progress

Progress Tracking

Trello Board

GitHub

Sprint 1 Completed



Bloomberg0512 Update README.md — aba9942 12 minutes ago — **15** commits

checklists	Create Sprint-4.md	6 hours ago
docs	Create docs folder	12 hours ago
src	Create src folder	11 hours ago
README.md	Update README.md	12 minutes ago

README.md

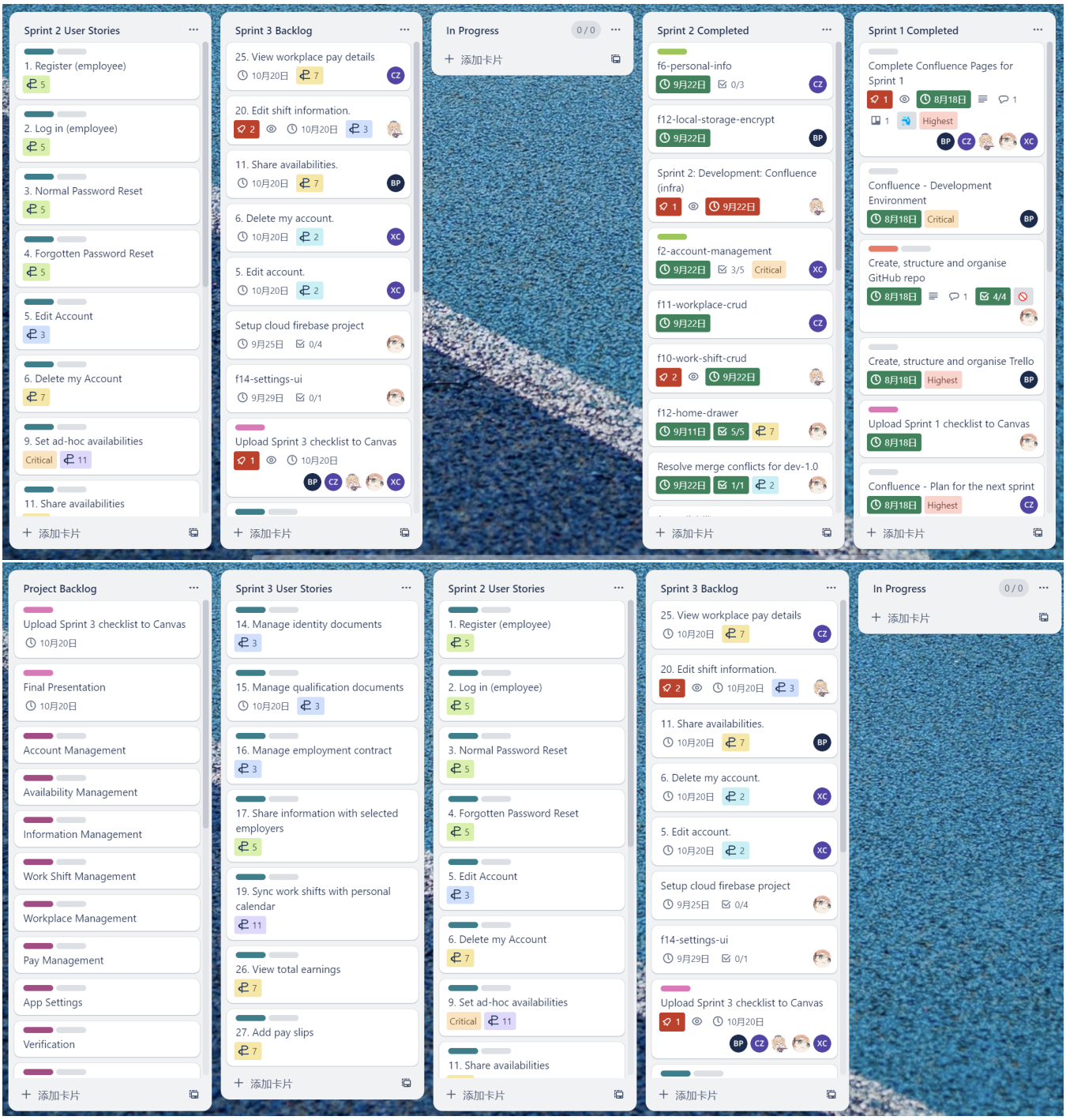
RO-BlueRing

RosterApp The aim of this project is to build a rostering tool that helps casual work employees manage their work schedule and share their work time availability with employers.

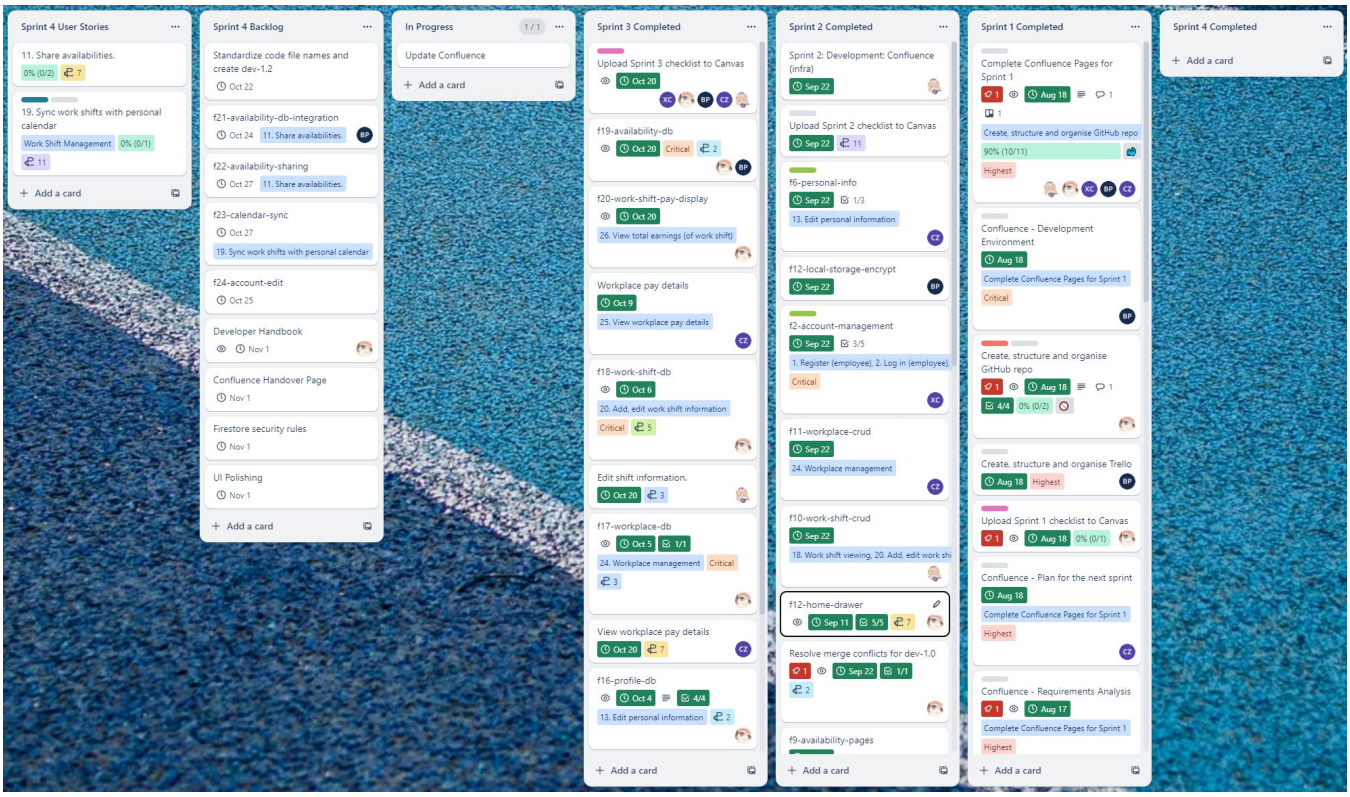
We aim to implement a solution that is:

1. Employee-centric
2. Intuitive and easy to use
3. Protect user's privacy, information can only be shared with user's approval
4. Maintainable and extendable

Sprint 2 Completed



Sprint 3 Completed



Meeting

Client Meeting Notes

The client meeting note is attached below

Since our client might not have too much time and also our team keen to provide some progress to present to our client at each meeting, our team and the client both decide to hold a meeting twice a week.

Client meeting 1 (11/August/2023)

Date:

11/August/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Obtain necessary details of this project and also client's requirements for this project.

Agenda:

Time	Item	Presenter	Notes
5 mins	Tool	Our team	What tech tool should we use on developing this app
5 mins	Database	Our team	Should we use database to store user's data
15 mins	Requirements	Our team	What is the requirments for this project
10 mins	Sample case	Our team	Could we get some example product? (Our team want to know about the UI desgin, and wha exactly is the product should be look like)

Action items
No tasks assigned after client meeting, our team normally assign task to team member during the team meeting

Notes:

The first client meeting mainly discussed the project description and requirements of the project.

Background: some people are working casually with casual contract, sometimes multiple employers want to employ one people at the same time and wasn’t know whether this people is available (some people work in multiple job). This project aim to provide proper solution to both employee and employer (mainly on employee).

Phase 1 (mandatory requirements):

- Set standard work day/hour availability
- Set non-working days
- Set ad-hoc availability
- Share availability with multiple employers potentially via email or SMS
- Integration with personal calendars
- Simple view on a screen of working week/month showing availability/place of work allocated shift times at a glance
- Load tax details to allow digital sharing of Tax File Number
- Load superannuation details to allow digital sharing of superannuation details to prospective employers

Specifically:

An user-friendly app is required to allow both employee and employer save and share availability with multiple employers by sms and email (expectation is just easily click “share” button OR share a live link with your schedule).

Once employee accept the shift it can allow employee to save the schedule in personal calendar.

When you accept the shift in app (employee should not longer available on the interface in real time) and it can be integrated to your personal local calendar. If cancel my shift then my status will return to available

View the availability directly in app from employer’s interface.

Both employer’s Identity and employee’s Identity are important, confidential information shouldn’t be sent without permission from the owner (employee can share personal information only when they accept the invitation from employer). And do anything that you think can improve privacy security.

TFN and superannuation must be locked unless employee allow employer to look at

Ideally, App should be fit in the system of iOS (minimum) and android app, and web based system

Employer should be able to view the availability of employee in real-time

Phase 2 (optional requirements):

- * Credential loading and validation for sharing with employers
- * Identify verification for employers to avoid sharing and employer maintaining identify documents (note: identity documents required for employment such as working with children will share minimum information required such as ID) - assume through 3rd party or Services Australia
- ^ Load award details to ensure timesheet and salary verification
- ^ Ability to calculate pay slips based on rostered/confirmed shifts
- ^ Ability to load pay slips for a consolidated view of total earnings to assist in identifying any upcoming tax implications (where employees work multiple jobs, there is a risk they are assigned a lower tax rate and will end up with a tax bill at EOFY)
- * Ability to load payslips for all employees direct to tool
- * Employer identity currency check (Licence and WWCC) where required for employment
- * Real time view of availability between employee and employer
- Store copy of employee contract
- * Push available shifts to available employees who can then accept or reject the shift in real time
- * Identify employee availability based on filtered requirements such as current allocated shifts, credentials, preference etc
- * Manage rosters for all staff
- * Web based portal for employers
- *^Integration with other common rostering tools from employers to manage allocated shifts (e.g. Deputy and common tools used by hospitals for rostering)

Specifically:

Log in with credential, how to validate the certificate (for example, truck driver need to present their license and app should be able to verify the validation of license befoe get a job) you got is up to student

Employee’s personal information must be confidential.

Use the third party to check whether a employee or employer has legitimate ID.

Extract information (award, what amount of money goanna paid by employer) from website (fare work commission), award from different job (nurses, teachers) can have different salary. Estimate what amount should be paid based on employee’ s experiences (industry, job, education, whether you have certification, age, how many years of working experience, permanent staff or part-time staff). Calculate the salary, employer can know how much theyshould pay for the employee and employee can know the salary they should get (however, the methodology of the minimum salary provided by government could be changed. So, we might need to usually update the methodology) .

For casual workers, they might get paid by multiple employers, calculate the tax (sum up the total salary and the total tax so that they can better plan how much amount of work should be done. For example, when your salary reaches the next level, you need to pay more tax)

Some job require certification, currency check the legality of the certification should be required.
Employers can save employees in a list and filter who is available in the list, filter employee based on credential, from high qualification to low qualification, past working experience and preferences.

Save Contract in the app

Employer can load payslips for all employees directly in this app

Other requirements:

UI design is up to students (simple, something easy to use, for old people)

Identity security is very important, well contained unless there are two way agreement (both employee and employer agree)

Should not store identity information in app

Make sure the message security between employees and employers

TFN need to be pushed by employee only (protect employee's confidential information unless employee aggree to disclose)

Build database by our own (Universty might give some supports)

Client meeting 2 (14/August/2023)

Date:

14/August/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Obtain more details about the requirements.

Agenda:

Time	Item	Presenter	Notes
20 mins	Details of project requirements	Our team	Get more details of this project

Action items

No tasks assigned after client meeting, our team normally assign task to team member during the team meeting

Notes:

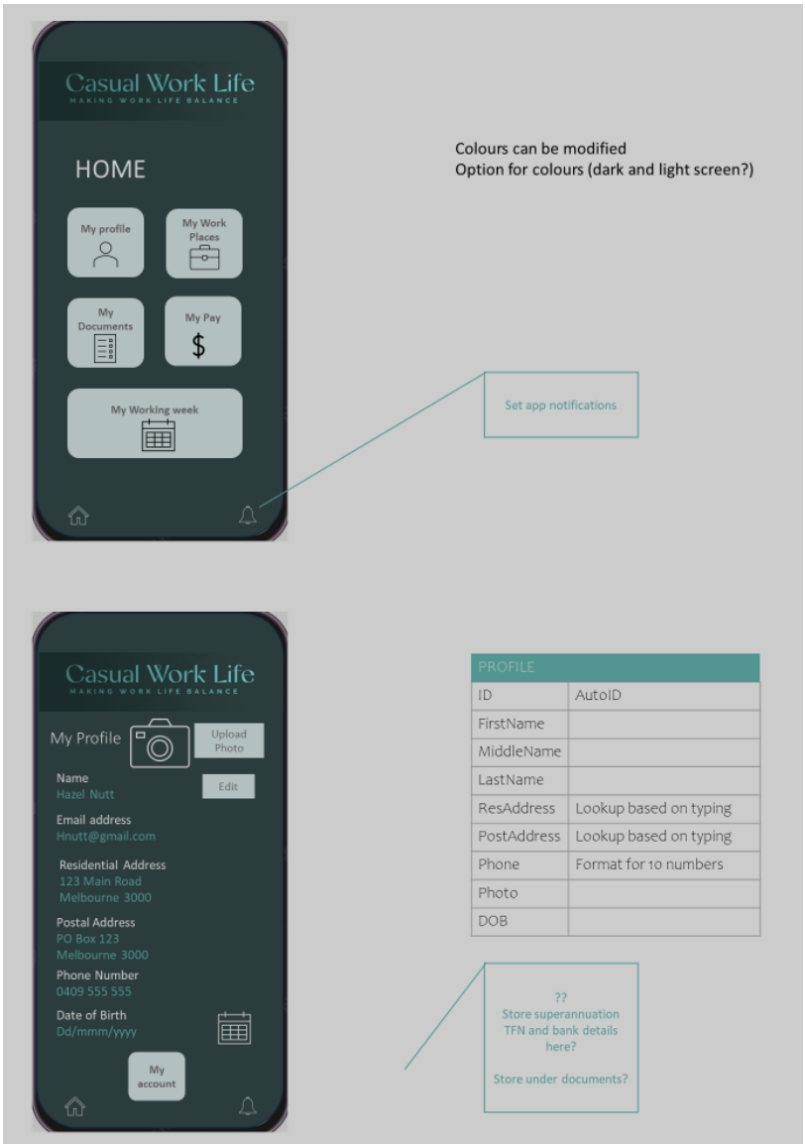
In this meeting we get some more details on the project requirements:

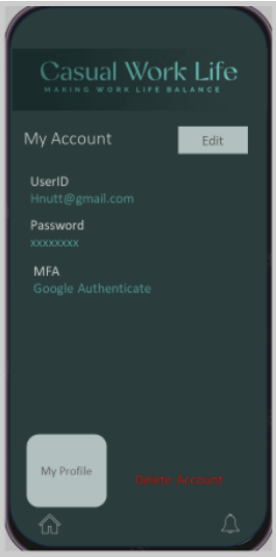
1. Since user may not be willing to share personal information, such as gender and date of birth, there should be check boxes before each item for user to choose to share (we should keep personal information confidential).
2. When deleting account,
    a. A window of Warning should pop up, indicating this operation will delete all of the data.
    b. Do logical delete first (reversible), then physical delete after 30 days(not irreversible).
    c. Provide an option to allow user to recover account information. Account data will be reserved for 30 days after user delete account operation.
3. If employee cancel shift, it should inform the employer that the employee has cancelled the shift. Employee can set another available time and share it with the employer.
4. If we need to verify document, it should let the 3rd party to do this.
5. Employee and employer should both be able to store the contract
6. Pay rate:
    a. Sometimes employee might get extra pay on weekend/holiday, there should be options for them to customize different types of pay rates for a workplace (overtime rate, holiday rate, etc.)
    b. Employee can select the pay rate for each shift.

We should primarily focus on the employee-centric functions.

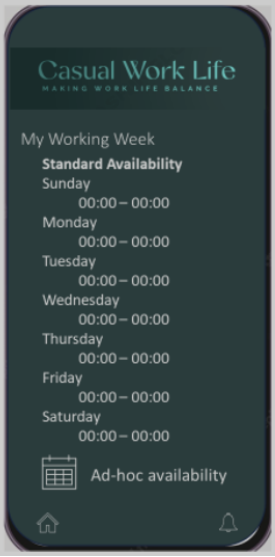
The contents below are provided by the client, which include details about requirements, UI design, and functions that client want to implement:


REQUIREMENTS
MANAGE AVAILABILITY
SET:
Set standard (recurring) availability, unavailability and regular shifts
SET:
Adhoc availability, unavailability (eg: holidays etc)
ACTION:
Quickly assign days/shifts to an employer
VIEW:
Simple view of availability, unavailability and committed shifts. Show place of work, shift times
ACTION:
Simply share availability with individual or multiple employers via SMS or email.
CONSIDER:
Link that provides live availability to employer when clicked
(Possibly subscription from employer phase 2)
INTEGRATE:
Committed shift with personal calendars.
Sync to ensure update when added/modified or deleted
MANAGE EMPLOYMENT
STORE:
Digital copy of employee contract
STORE and SHARE:
Superannuation details (can have many)
TFN details (one only)
Digitally share with employer
Consider if digital signature is required
STORE and SHARE
Load/Scan, store and share certifications or certification details
MANAGE PAY
STORE and SHARE
Bank account details (don't connect to bank).
Bank Name
Account name
BSB
Account Number
CALCULATE:
Pay by shift or selected period
By Employer or multi employers

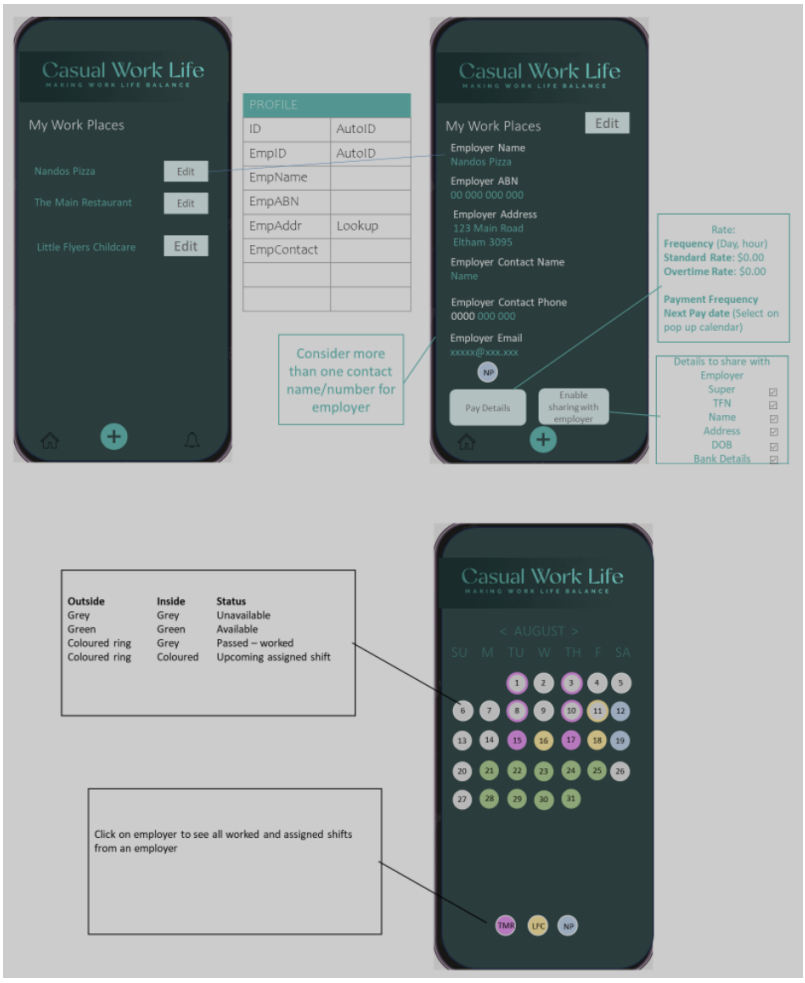

Casual Work Life
MAKING WORK LIFE BALANCE
HOME
My profile
My Work Places
My Documents
My Pay
$
My Working week
Colours can be modified
Option for colours (dark and light screen?)
Set app notifications
Casual Work Life
MAKING WORK LIFE BALANCE
My Profile
Upload Photo
Name
Hazel Nutt
Edit
Email address
Hnutt@gmail.com
Residential Address
123 Main Road
Melbourne 3000
Postal Address
PO Box 123
Melbourne 3000
Phone Number
0409 555 555
Date of Birth
Dd/mmm/yyyy
My account
PROFILE
ID AutoID
FirstName
MiddleName
LastName
ResAddress Lookup based on typing
PostAddress Lookup based on typing
Phone Format for 10 numbers
Photo
DOB
??
Store superannuation TFN and bank details here?
Store under documents?

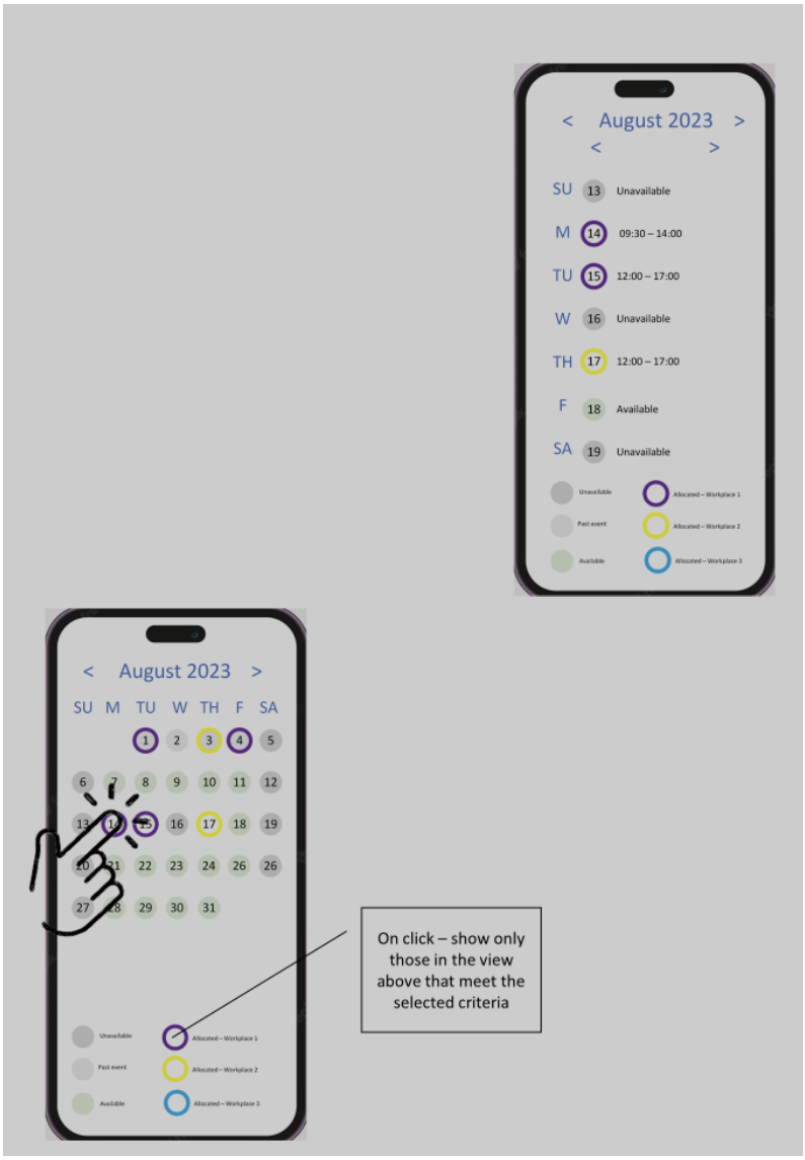


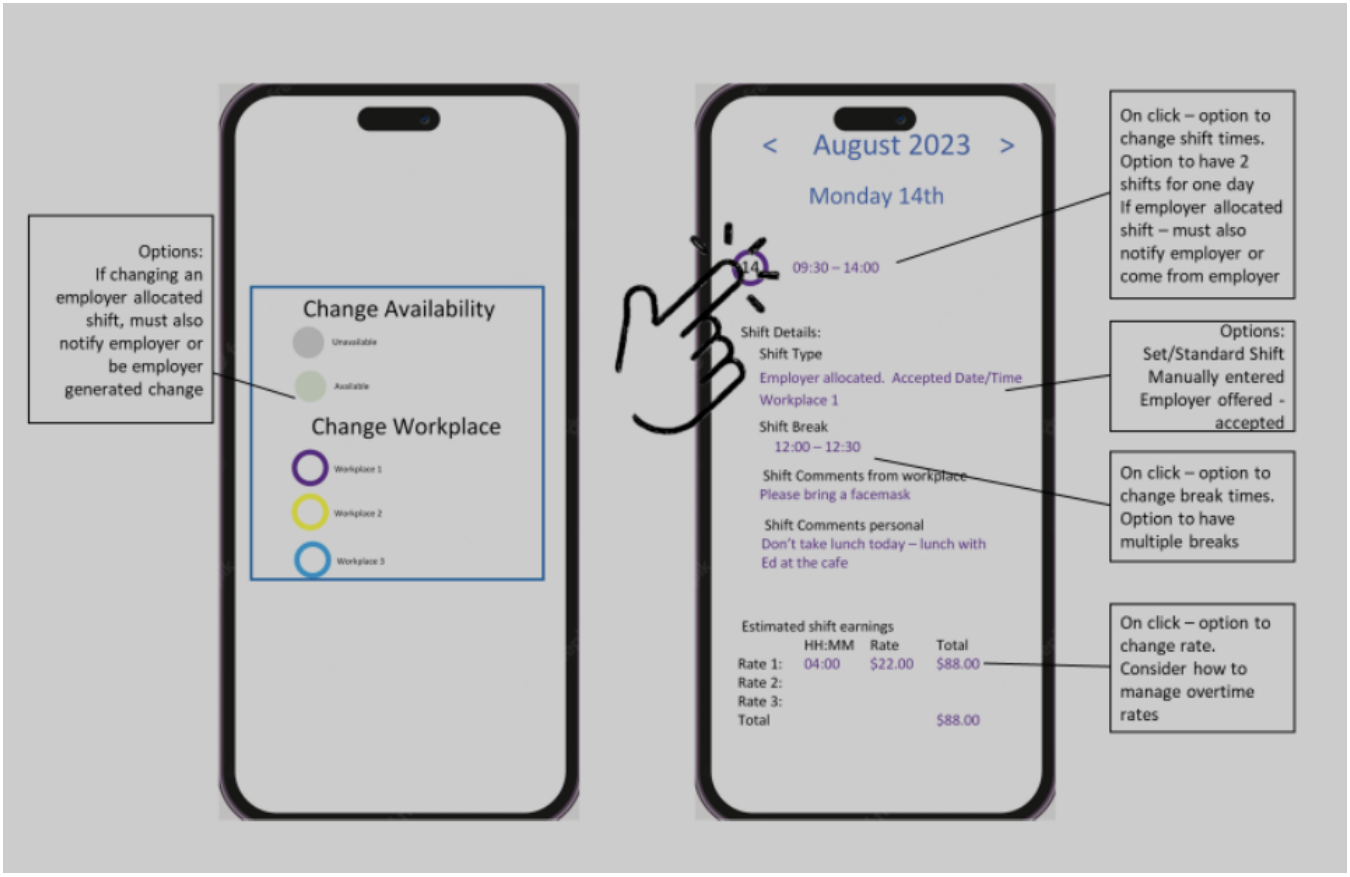
PROFILE	
ID	AutoID
UserID	
Password	Encrypt – allow all characters 8 character min – must include Upper, lower, numeric and special.
MFA	Not sure if this is necessary?
Delete account	Physical or logical delete?



Setup wizard for availability – phase 2


Casual Work Life
MAKING WORK LIFE BALANCE
My Work Places
Nandos Pizza
Edit
The Main Restaurant
Edit
Little Flyers Childcare
Edit
PROFILE
ID
AutoID
EmpID
AutoID
EmpName
EmpABN
EmpAddr
Lookup
EmpContact
Consider more than one contact name/number for employer
Casual Work Life
MAKING WORK LIFE BALANCE
My Work Places
Edit
Employer Name
Nandos Pizza
Employer ABN
00 000 000 000
Employer Address
123 Main Road
Eltham 3095
Employer Contact Name
Name
Employer Contact Phone
0000 000 000
Employer Email
xxxxx@xxx.xxx
NP
Pay Details
Enable sharing with employer
Rate:
Frequency (Day, hour)
Standard Rate: $0.00
Overtime Rate: $0.00
Payment Frequency
Next Pay date (Select on pop up calendar)
Details to share with Employer
Super
TFN
Name
Address
DOB
Bank Details
Outside
Inside
Status
Grey
Grey
Unavailable
Green
Green
Available
Coloured ring
Grey
Passed – worked
Coloured ring
Coloured
Upcoming assigned shift
Click on employer to see all worked and assigned shifts from an employer
Casual Work Life
MAKING WORK LIFE BALANCE
< AUGUST >
SU M TU W TH F SA
TMR
LPC
NP


< August 2023 >
< >
SU 13 Unavailable
M 14 09:30 – 14:00
TU 15 12:00 – 17:00
W 16 Unavailable
TH 17 12:00 – 17:00
F 18 Available
SA 19 Unavailable
Unavailable
Past event
Available
Allocated – Workplace 1
Allocated – Workplace 2
Allocated – Workplace 3
< August 2023 >
SU M TU W TH F SA
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 26 26
27 28 29 30 31
On click – show only those in the view above that meet the selected criteria


Options:
If changing an employer allocated shift, must also notify employer or be employer generated change
Change Availability
Unavailable
Available
Change Workplace
Workplace 1
Workplace 2
Workplace 3
< August 2023 >
Monday 14th
14
09:30 – 14:00
Shift Details:
Shift Type
Employer allocated. Accepted Date/Time
Workplace 1
Shift Break
12:00 – 12:30
Shift Comments from workplace
Please bring a facemask
Shift Comments personal
Don't take lunch today – lunch with Ed at the cafe
Estimated shift earnings
HH:MM Rate Total
Rate 1: 04:00 $22.00 $88.00
Rate 2:
Rate 3:
Total $88.00
On click – option to change shift times. Option to have 2 shifts for one day If employer allocated shift – must also notify employer or come from employer
Options:
Set/Standard Shift
Manually entered
Employer offered - accepted
On click – option to change break times. Option to have multiple breaks
On click – option to change rate. Consider how to manage overtime rates

Client meeting 3 (18/August/2023)

Date:

18/August/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Obtain necessary details of this project in terms of UI design and what features or functionalities we should have on the availability page, shift /calendar page and workplace page.

Agenda:

Time	Item	Presenter	Notes
5 mins	UI design	Our team	We have a UI design sample, but not sure whether it meets the requirements from client. So, we intend to present it and discuss it with client.
10 mins	Requirements	Our team	What features we should have on the availability page, shift/calendar page and workplace page.

Action items

No tasks assigned after client meeting, our team normally assign task to team member during the team meeting

Notes:

We showed the sample UI design to Sharon, and we got some feedbacks:

1. In the availability page:
    a. We should have the function that allow user to set standard availability (such as set recurrent working time for every week).
    b. Ad-hoc availability should override standard availability (should be able to select a range of dates, because user may set ad-hoc availability for next few weeks)
    c. We should have the function that allow user to click calendar and select time slots.
2. In the user profile page:
    a. it shouldn't have bank account, TFN, and SFN on my profile because they are my sensitive personal information.
    b. Profile page should only have name, address mobile, date of birth, email, post address.
    c. Work contracts should be stored in the page of workplace rather than profile page.

For sprint 2 planning, we shouldn't includes the following features, they should be assigned into sprint 3:

1. We should develop a function to allow user to upload payslips.
2. Option for employee to upload payslips (papers that show what amount you are being paid, on what date), employee can be paid weekly. Need to have somewhere to store payslips (png or pdf, select the employer).

In terms of the pay rate, we should include an external link to official guide that indicates the proper pay rates for different kind of employees.

The app should only share personal informations the are explicitly selected by the user.

Furthermore, the client also provide more details of requirements:

Additional items to note

Workplace screen

Next to workplace details include an Icon ⓘ that provides details on what the information is used for.

- Employer Phone number: May be used to send employer availability details.
Only a mobile number can be used to send SMS to employer
- Employer Email: This can be used to send employer availability details

It would be useful to allow more than one phone number / email.

Shift details screen.

Breaks

- Employees can have 2 types of breaks
  - Rest Breaks (Paid – to be included in working hours)
  - Meal Breaks (Unpaid – to be deducted from working hours)
- Include an icon ⓘ that provides the above information when selecting the type of break.

- Include a link for on the workplace page that takes the user to the Fair Work Award page. This allows employees to identify their award and view their pay and conditions.
- On the Workplace page – have an ⓘ called know your Award

This would show the following

- Identify your award and view your Pay and Conditions
- https://calculate.fairwork.gov.au/

Calculating pay

Rates types the employee can select against each workplace would be

- Base rate
- + Casual loading (%)
- + Evening Work Penalty
- + Night Work Penalty
- + Saturday Penalty
- + Sunday Penalty
- + Public Holiday Penalty

Also allow the employee to add an additional rate/penalty option they can name.

The employee can select a rate type and then enter their $ rate against for each of the applicable rate types.

These will then be used for the employee to select against their allocated shift.

Fair Work OMBUDSMAN

Background info team for context only

Award Details Example

Awards (modern awards) are legal documents that outline the minimum pay rates and conditions of employment
There are more than 100 industry or occupation awards that cover most people who work in Australia.

An example of 2 awards are ….
https://library.fairwork.gov.au/award/?krn=ma000003

https://www.fairwork.gov.au/employment-conditions/awards/awards-summary/ma000120-summary

I have decided not to try and use app to calculate the rate of pay – it is significantly easier to direct the employee to the FairWork commission website where they can select the correct award for them and pick up applicable rates.

Client meeting 4 (01/September/2023)

Date:

01/September/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Obtain necessary details of the work-shift page, and workplace page
- Show our progress to the client and get feedback

Agenda:

Time	Item	Presenter	Notes
10 mins	Requirements	Our team	Get more specific requirements from client for work-shift page, and work-place page
10 mins	Present progress	Our team	Present our current progress to client and confirm whether it is okay

Action items

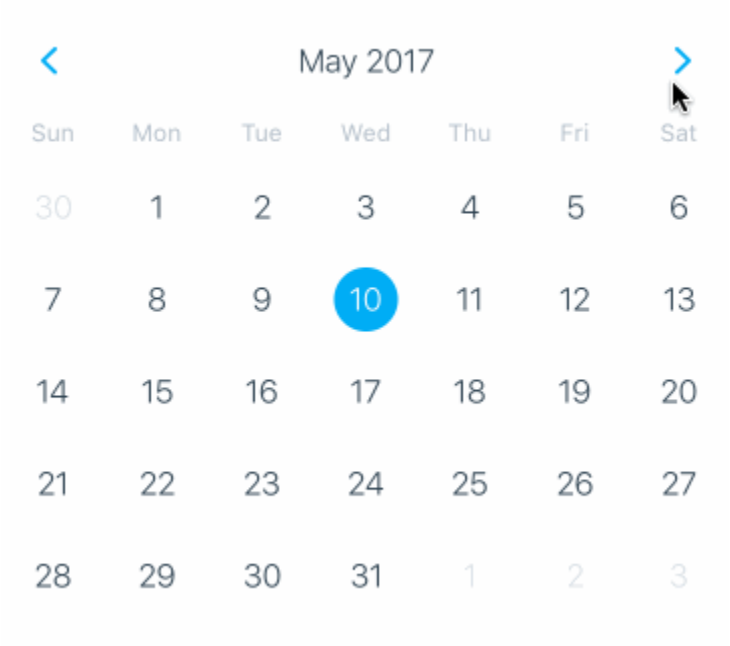
No tasks assigned after client meeting, our team normally assign task to team member during the team meeting

Notes:

In this meeting we presented progress to our client and there are additional requirements and feedbacks provided by the client:

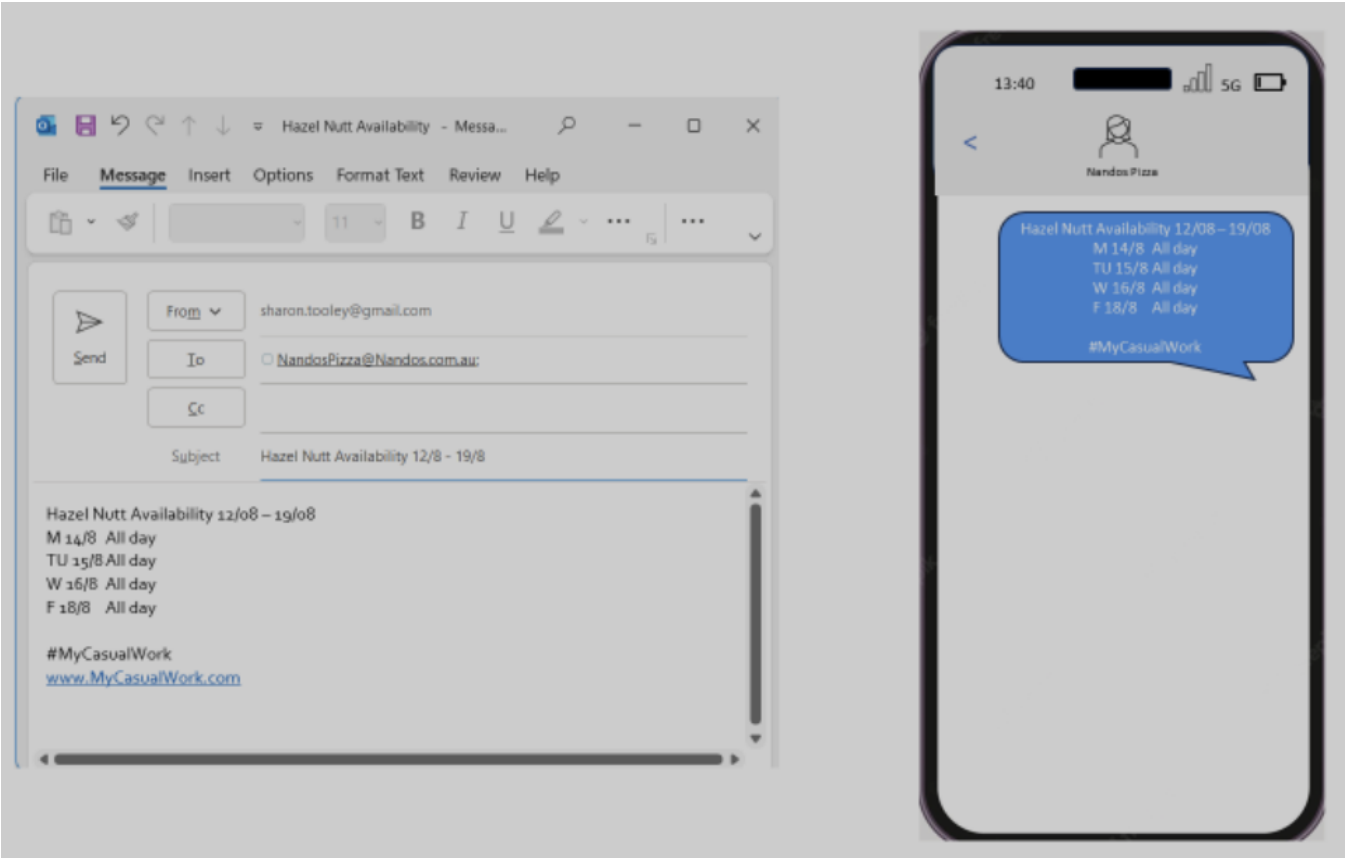
1. In term of ethical consideration, Color blind pattern can be applied on the app
2. The availability page looks busy and user should be allowed to select any time range in a day as one availability time slot.
3. Select availability by using calendar like the gif below (client doesn't require the exact same calendar picker but it should be something similar):

a. This picture is cited from https://github.com/wix/react-native-calendars



4. In Work shift page:
    a. User should be able to add notes in each shift.
    b. User should be able to add the employer name for each shift.
    c. User should be able to select time range for each shift.
    d. Should display the working hours (working hours = start time – end time – break time) for each shift.
5. As mentioned in previous meeting, we should develop a function to calculate the estimated earning for user

Furthermore, our team also discuss about how to deploy the functionality of sync with user's personal calendar and availability sharing functionality (via sms or email), client gave some ideas about how to achieve this functionality and one sample layout of availability sharing message as below:



Client meeting 5 (15/September/2023)

Date:

15/September/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Show our current progress and get some feedback from client

Agenda:

Time	Item	Presenter	Notes
20 mins	Present progress	Our team	Our team make improvement based on the feedbck we got from the last client meeting and we need to present the improvement to the client and also get the feedback as well

Action items

No tasks assigned after client meeting, our team normally assign task to team member during the team meeting

Notes:

In this meeting we presented progress to client and got some feedbacks:

Since the last meeting, we make some improvement on the availability page:

1. Fix errors in availability page
2. User should be able to add label indicating "available" or "unavailable" on ad-hoc work availabilities.
3. We should have the functionality of date range picker on calculating total earnings.
4. We should have the functionality to inform user when they set availability that override another availability.

Also, client want to have the following features:

1. If possible deploy data encryption

Client meeting 6 (28/September/2023)

Date:

28/September/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Show our progress to and get feedback from the client

Agenda:

Time	Item	Presenter	Notes
15 mins	Present progress	Our team	Our team make improvement based on the feedback we got from the last client meeting and we need to present the improvement to the client and get the feedback as well

Action items

No tasks assigned after client meeting, our team normally assign task to team member during the team meeting

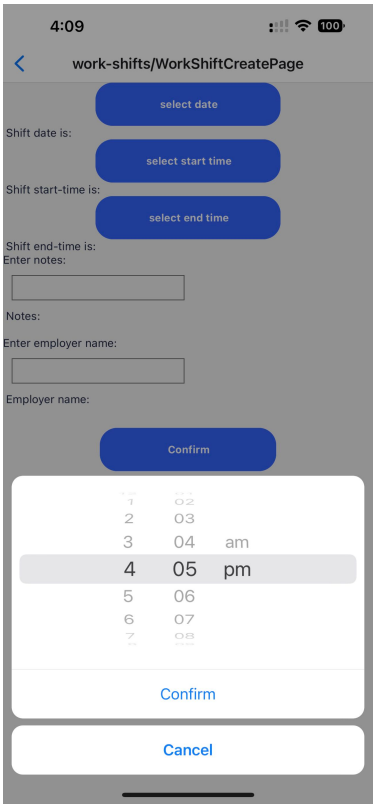
Notes:

In terms of workplace management page, gender field should have a dropdown list.

For the address field in user profile and workplace profile, client want to have a search address function (when user types in key words, app should provide some possible result), which is hard to be implemented.

Phone number field should be in standard format : country code + phone number. For example, +61 444743451

We discussed that the date time picker function performs differently on different system (iOS and android). The client cares more about the IOS platform and we should make the date time picker to be a scrollable picker on iOS.


4:09
work-shifts/WorkShiftCreatePage
select date
Shift date is:
select start time
Shift start-time is:
select end time
Shift end-time is:
Enter notes:
Notes:
Enter employer name:
Employer name:
Confirm
2 03
3 04 am
4 05 pm
5 06
6 07
Confirm
Cancel

Client meeting 7 (13/October/2023)

Date:

13/October/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Due to time limit, we need a re-negotiation of user story, probably drop some user stories from the initial planning.

Agenda:

Time	Item	Presenter	Notes
15 mins	Re-negotiation of user story	Our team	Due to time limit, we want to re-negotiation of user story, probably drop some user stories (our team aim to drop user story 19 and 27 because both user stories are rank in low priority but difficult to achieve)

Action items

No tasks assigned after client meeting, our team normally assign task to team member during the team meeting

Notes:

In this client meeting, our team discussed about whether we should drop some user stories in this project as we may not have enough time to finish everything on time before Nov 3.

Before the meeting, our team think we should drop user story 19 and 27 because both user stories are in low priority and difficult to achieve.

However, the client considered user story 19 to have higher priority than other unfinished user stories. Therefore we don't need to worry about other user stories and focus on finishing user story 19.

The user story 19 is to "Sync work shifts with personal calendar".

On the other hand, we need to add a button on the home screen page to allow user to view their total earnings (user can view earnings in a period).

Moreover, we discussed that the user story 6, "delete account", can only be partially implemented because part of the requirement requires a paid plan from firebase. Instead of paying a membership, we tend to write the procedure of how to accomplish the full functions in the documentation and only implement the basic deleting function in the product. The client agreed with us.

Client meeting 8 (30/October/2023)

Date:

30/October/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present the final product to our client

Agenda:

Time	Item	Presenter	Notes
15 mins	Presentation	Our team	

Action items

Eveyone keep working on our own work

Xuechun Chen will be in charge of the functionality of sign out

Notes:

Next meeting will be the handover meeting which is also the final meeting held on 3 Nov.

Client meeting 9 (3/November/2023)

Date:

3/November/2023

Attendees:

Sharon Tooley

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present the final product to our client

Agenda:

Time	Item	Presenter	Notes
15 mins	Presentation	Our team	

Action items

None

Notes:

Sharon is satisfied with out final product.
This project is finished in this semester and might be imporved further in the future.

Supervisor Meeting Notes

Supervisor meeting 1 (07/Augest/2023)

Date:

07/Augest/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Initial set up of the project

Agenda:

Time	Item	Presenter	Notes
10 mins	Introduction	Our team	Get familiar with each other
20 mins	Set up	Our team	Set up theenvironment for the project

Action items

None

Notes:

Our team has done the following during this meeting:

1. Setting up working environment of Confluence, Trello and github
2. Define the role of scrum master, Product owner and meeting notes recorder

Meeting Recording

https://unimelb.zoom.us/rec/share/KuRVVF58wQe2rqLyLdcSgnt2YVlNrUorMUOQbmEE4-CY8dwO62Weg8kqNnRHoN93.je-B3eS08TA-_yLu
Passcode: e!f5%YZ3

https://unimelb.zoom.us/rec/share/WOEvm1M1pKOQJ0S0WiyUrnsWWr5VX1CYI0IoxgdJLdPs-EW9VnIZevLptGiV45wy.Ay_roRqqYYX8HmNI
Passcode: e!f5%YZ3

Supervisor meeting 2 (14/Augest/2023)

Date:

14/Augest/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Discuss with supervisor about what content we should have in confluence, tello, and github and how to do documentation

Agenda:

Time	Item	Presenter	Notes
10 mins	Documentation	Our team	Documentation for confluence, tello, and github

Action items

None

Notes:

We set up github repository during the meeting

Based on the discussion between supervisor, our project details should contains:

- student details
- supervisor details
- stakeholder photo

Links to Github and Trello should be in confluence

persona : potential user group of this product

About the project:

1. Think what type of employee use this product (how old people or young people use this product, need of different user)
2. Motivation:
    a. who can be your user
    b. how the action of the system should be (expectation of the system)
    c. what your feel about the system

Supervisor meeting 3 (21/Augest/2023)

Date:

21/Augest/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress and get feedbacks from supervisor to see if we are doing correctly

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	

Action items

None

Notes:

Supervisor checked Trello and confluence, and we got the following advice from the supervisor:

1. Merge “manage and share information” with other groups to form a big category.
2. We should assign tasks or user stories for sprint 2 for each person.

Supervisor meeting 4 (28/Augest/2023)

Date:

28/Augest/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress and get feedbacks from supervisor to see if we are doing well

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	

Action items

None

Notes:

We got the following advice from our supervisor:

1. Make sure other people can review other people's work.
2. Break down big epic categories into small pieces (user story level).
3. Make guidelines for team and document for the project (in term of ethical consideration).

Supervisor meeting 5 (4/September/2023)

Date:

04/September/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress and get feedbacks from supervisor to see if we are doing well

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	Present our current progress to client and confirm whether it is okay

Action items

- Xuechun will work on account registration
- Sichen will work on new batch of features on home pag
- Bo will work on re-design UI on availability page
- Chenxi will work on backend side of features.
- Wenxuan will work on add and edit features of work shifts

Notes:

Check progress for each members, and assign new tasks for each member.

Supervisor meeting 6 (11/September/2023)

Date:

11/September/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress and get feedbacks from supervisor to see if we are doing well

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	

Action items

None

Notes:

- In this meeting we checked everyone’s progress and complete our trello.
- Make sure everyone’s work are integrated together
- Deadline should be mentioned in trello
- Backlog should be mentioned in trello
- High level, Low level backlog should be mentioned in trello
- Dependency of other task should be mentioned in trello
- Document client feedback

Supervisor meeting 7 (20/September/2023)

Date:

20/September/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress and get feedback from supervisor to see if we are doing well
- Present our sprint 2 content and see if it is good enough and probably get some feedback from supervisor

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	Get feedback of our content in confluence

Action items

None

Notes:

In this meeting, we discussed about the assignment, *Sprint 2: Development,* with supervisor and got the following feedbacks and suggestions:

1. Spreadsheet for code review (go through lecture, there are template and instructions in lecture slides)
2. Details of cyber security (go through lecture slides)Mark dowm what we have change (goal modal, old version, new version, and what we have change based on the feedback from sprint 1)
3. Markdown what you have done and what you haven't done carefully based on the sprint planning
4. Make sure everything in github, Trello and confluence are consistent.
5. In terms of demo, we should mainly show what we have done, what is our progress on the project and what functionality we have deployed. It would be great if we could show the functionality by checking up the user stories (maybe show functionalities based on the user stories on Trello). Our group decided to make the demo though video.

Supervisor meeting 8 (05/October/2023)

Date:

05/October/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress and get some feedbacks from supervisor to see if we are doing well
- And also discuss what we should improve in our project in terms of source code

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	
5 mins	Possible improvement	Our team	Show our project to supervisor and get some advice to see what we should improve

Notes:

We discuss our progress and show our confluence and Trello progress with supervisor and supervisor think it is good

Supervisor meeting 9 (11/October/2023)

Date:

11/October/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress and get some feedback from supervisor to see if we are doing well
- Since our team might not have enough time to finish all user stories, we want to discuss with the supervisor to see whether it is possible to re-negotiate with the client and drop down some user stories
- Decide the time slot for the presentation

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	
10 mins	Discussion of dropping user stories	Our team	
5 mins	Decide time for the presentation	Our team	

Action items

None

Notes:

- In this supervisor meeting we showed our progress of development with the supervisor.
- We discussed about the details of final presenation and decided to present on Oct 20 afternoon.
- Supervisor suggest us to state the reason why we are not able to complete all the user stories by providing the story map with the client.

Supervisor meeting 10 (18/October/2023)

Date:

18/October/2023

Attendants:

Madhavi Mohoni

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Discuss what we suppose to cover in sprint 3 presentation
- Discuss the feedback we got from sprint 2 development and what we need to improve from sprint 2

Agenda:

Time	Item	Presenter	Notes
10 mins	Presentation content	Our team	
10 mins	Sprint 2 feedback discussion	Our team	Not very clear about image attachment problem, and sprint review structure

Action items

None

Notes:

- Based on the discussion with supervisor, sprint review and sprint retro are 2 different stuff.
- In terms of image attachment problem, supervisor suggest us to not upload any link instead replace link with original files.

Team meeting

Team meeting 1 Agenda (07/Augest/2023)

Date:

07/Augest/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Discuss what we should prepare for the client and what we need to know from the client.

Agenda:

Time	Item	Presenter	Notes
20 mins	Prepare questions	Our team	Prepare questions to know more about this project from client

Action items

No tasks assigned after the initial meeting

Notes:

In the initial meeting we prepared some important questions for the client, which could be helpful for our future development.
Objectives: simple and easy to use.

Base features

1. We can choose our own tech stack
2. Requirements for extensibility, security, etc.
3. Any examples of ideal products?
    a. User interface design: simple and intuitive. Employee centric
    b. Any existing product for reference? Your favorite app- simple - Canva, Uber, Apple
4. Share schedules:
    a. Personal info(TFN, availability) can be shared only after the employee's approval.
5. Local tool or online tool?
    a. If online, we will communicate with Uni for server resources.
6. Integration with personal calendars
    a. What does calendar refer to? E.g. Local calendar apps like Outlook / iOS calendars. Sync with personal calendars on both allocation and cancellation.
7. Are there any specific features that need to be completed by the end of each sprint?
    a. Sprint 1 - Overall design - by Aug 18
    b. Sprint 2 - by Sep 22
    c. Sprint 3 - by Oct 20
    d. Sprint 4 - by Nov 3

Additional Features

1. Credential validation / Identity verification for employers and employees through 3rd party.

Additional features we may be able to do at the moment

Confidence: How confident we are in our ability to implement this feature.

Confidence 1

1. Credential loading and validation (medium - requires 3rd party)

2. Calculate pay slips (medium - high if using employee entered rate)
3. Real time view of availability between employee and employer (high)
4. Store copy of employee contract (high)
5. Identify employee availability based on filtering, e.g. allocated shifts, credentials, preference (medium)
6. Web based portal for employers (medium)

Confidence 2

1. Push available shifts to available employees (low - would require employer portal before this could be implemented)
2. Manage roster for all staff (low - rostering systems will require considerable thought)

Tech Stack

Frontend

- Target Platform: Android, iOS
    - **Device**: supports iOS or Android, just scan a QR code
    - **Simulator**
        - Xcode + iOS simulator, only available for MacOS
        - Android Studio available for Windows and MacOS
    - JavaScript / TypeScript, React Native
    - Must support live preview for debugging
    - Expo provides live preview options: on-device or simulator

Backend

- Java, Spring Boot

1. Hosting
    a. 000WebHost + cheap domain ($2.99/yr) + deploy on server
    b. Azure app services
    a. AWS Lambda + Amazon API Gateway (12 months free)
    b. Pros: less work needed
    a. Option A: Conventional backend (Java Spring Boot)
    b. Option B: No backend, all cloud-based (Node.js or Java Spring Boot)

DevOps

- Github Actions.
    - On pull request to main / dev branch, run tests + build + deploy to server

Database

- MongoDB Atlas

Project Scope

Use Scenario

As an employee with (potentially) multiple casual jobs, I want to be able to edit my schedule using a visual tool and let employers know when I’m available.

- Edit availability schedule using a visual tool (our web app).
- (Local) **Export as text** and share via email or SMS
- (Online) **Share a link**, viewers who click the link could view the schedule on our web app.
    - The employee’s schedule is saved in a remote database, so the viewer can see the schedule via our app on their end or through the web portal.

Minimum Viable Product

1. Android / iOS App

Employee

1. Simple view
    a. Show working week / month
    b. Show availability / allocated work / shift times
2. Set standard work day/ hour availability
    a. Non-working days, ad-hoc availability
3. Integrate with personal calendars
    a. Sync work event allocation and cancellation
4. Share availability via email or SMS
    a. Have a list of employers and contact details to choose from

Team meeting 2 (14/Augest/2023)

Date:

14/Augest/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Discuss our future sprint planning
- Dicuss our personal available time slut and find some common available time
- Assign work to each team members

Agenda:

Time	Item	Presenter	Notes
10 mins	Future sprint planning	Our team	
10 mins	Work allocation	Our team	
5 mins	personal available time slut	Our team	

Action items

Wenxuan Chen will be in charge of work-shift page

Bo Peng will be in charge of work-availability page

Chenxi Zhang will be in charge of work-place page

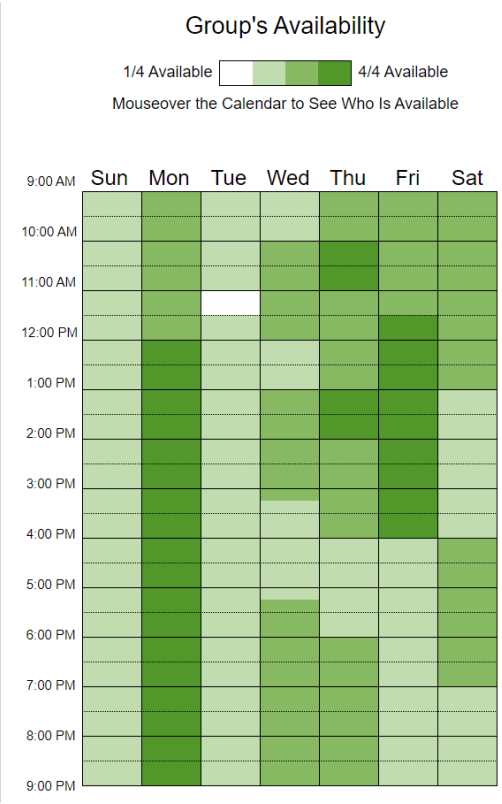
Sichen Lu will be in charge of home page and database set up

Xuechun Chen will be in charge of user log-in and account registration page

Notes:

1. Meeting times
    a. Mon 16:00

b. Fri 14:00



2. Tasks
    a. Aim for Thursday 17 August
    b. How we're doing so far
    c. Assign remaining tasks
    d. Deadlines
3. Client meeting on Friday 16:00 PM
    a. Recurring zoom meeting
    b. Agenda
        i. Show our draft design
        ii. Next sprint focus on local features that don't require networking
        iii. Firebase for networking?
        iv. Mobile app or responsive web app?
    c. Invitation sent?
4. Plan for the next sprint

Team meeting 3 (21/Augest/2023)

Date:

21/Augest/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Disucss what we need to do in this sprint and how to do documentation in confluence

Agenda:

Time	Item	Presenter	Notes
10 mins	Sprint planning	Our team	

Action items

Everyone should complete Confluence Pages for Sprint 1 together, Requirements Analysis
Wenxuan Chen will be in charge of Goals page

Bo Peng will be in charge of Development Environment, Create, structure and organise Trello, Personas

Chenxi Zhang will be in charge of making Plan for the next sprint, Overview page

Sichen Lu will be in charge of Create, structure and organise GitHub repo, Create, structure and organise Trello, DO-BE-FEEL list, Home page, and Technical details page

Xuechun Chen will be in charge of Motivation Model, Background page

Notes:

1. Set up sprint 2 planning
2. Revise confluence pages & go through Sprint 1 checklist
3. Make plan for the next sprint

Team meeting 4 (28/Augest/2023)

Date:

28/Augest/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Report progress of documentaion

Agenda:

Time	Item	Presenter	Notes
10 mins	Report progress	Our team	

Action items

None

Notes:

We have done the following before this meetng

1. Work allocation of sprint 2
2. Assign work for each member
3. Software environment set up
4. Software environment check up

Team meeting 5 (4/September/2023)

Date:

4/September/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Report progress
- Discuss UI desgin
- Discuss ethical consideration problem in our project

Agenda:

Time	Item	Presenter	Notes
5 mins	Report progress	Our team	
5 mins	UI desgin	Our team	UI desgin for each page should be consistent
5 mins	Ethical consideration	Our team	Think about what we can do in terms of ethical considration in thi project

Action items

None

Notes:

1. Decide what UI we should use (need to have consistent UI) (UI kitten style)
2. Decide what color platte we should use on app (color blind mode)

Team meeting 6 (11/September/2023)

Date:

11/September/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Each member present their progress to the team and other team member give feedback

Agenda:

Time	Item	Presenter	Notes
10 mins	Report progress	Our team	Present current progress to other team member

Action items

None

Notes:

1. Interal check each member's work and merge codes together, everyone's work is good and up to standard

Team meeting 7 (18/September/2023)

Date:

18/September/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress
- Dicuss what we haven't finish during this print and document them carefully
- DIsucss what we need to add into the confluence to meet the rubric reqpriment, and also what we need to do to improve our current confluence based on the feedback from last sprint

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	Present our current progress to client and confirm whether it is okay
10 mins	Documentation	Our team	Gather every team members opinions and dicuss how to make the confuence page better

Action items

Wenxuan Chen will be in charge of "in-scope features" and "out-scope features"

Bo Peng will be in charge of updating Trello and ethical consieration and Cyber security page on confluence

Chenxi Zhang will be in charge of sprint planning and sprint review

Sichen Lu will be in charge of making a demo of our current project

Xuechun Chen will be in charge of improve our existing content based on the feedback from the last sprint

And everyone should make a code review

Notes:

Check everyone's progress and discuss the following:
1. Discuss what we have done and what we haven't done, which should be assigned in the next sprint

2.Based on the feedback from previous sprint, make improvement especially need to add "in-scope features" and "out-scope features"

3.Burndwn chart (maybe useful)

4.Schedual diagram

5.Decide how to make a good demo that can demonstrate the functionality we have done?

Possoble approach: from very start, show how user can use this app setp by step and we can ask supervisor for deatils in the next supervisor meeting and we tend to use vedio to demonstrate our progress

6.Update Trello, what we have done and what story points we have done, add description of tasks and also deadline

7.Make sprint review, what challenge we meet, what we need to adjust based on this sprint, do we need to adjust our next sprint planning? based on the performance of this sprint. In this sprint what risk we might gonna experience and provide some protential solution. We can catogorize them into

resolve?

mitigate?

don't do anything?

8.Ethical consieration, develop a color blind mode for special people (might not be able to deploy in this sprint)

9.Team members should ask another meber to do code review when we make pull request from the development branch (make sure everyone's work is okay), we could use chat gpt to make code review and aslo we can add some personal comment on top of this.

10. Writre a database diagram

11. Cyber security, we could use hash key to hide user's password when we store them into database.

12. Test cases, manually do some experiments on the app and check whether it can work out

13. Make proper sprint 3 planning based on the performance on this sprint, and what we learned from this sprint

Team meeting 8 (25/September/2023)

Date:

25/September/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	

Action items

None

Notes:

In this meeting our team members are just brefly report our own progress to each other.

Sichen has finish he re-work of our codes (work place page and work shift page)

Other team members are still develop their own part

Team meeting 9 (02/October/2023)

Date:

02/October/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress
- Disucss our plan for the current week and the next week

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	
10 mins	Planning	Our team	Discuss the planning for this week and probably next week

Action items

None

Notes:

In this team meeting we disucss about what we need to improve in the app. Since we already build up a fundamental app but we still have some functionality left to developed, we need to discuss when to finish which features or user stories on which date.

We finally decide to finish user story 20 and 25 first before Oct 09. And then we will discuss further planning of this sprint in the next meeting.

Team meeting 10(09/October/2023)

Date:

09/October/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress
- Discuss whether we should re-negotiate with client to drop down some user stories (as some user stories are ranked in low priority but difficult to be implemented)

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	
10 mins	Re-negotiation	Our team	See if we could drop some user stories

Action items

None

Notes:

Our team members held this meeting mainly for discussing the schedule of the rest of sprint.

Since team members need to also spend time on other subjects, and based on our members' estimates, we may not have enough time to finish the rest of user stories on time, especially for user story 19 which is to Sync work shifts on user's personal calendar, which could be difficult to fulfill. Our team planning to keep following the planning until the next client meeting, we might need to re-negotiate with the client to see if we could drop some user stories.

Team meeting 11(16/October/2023)

Date:

16/October/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	

Action items

None

Notes:

This is just a small simple team meeting, team members report their own work progress. Since most of team members re vey busy unitl aaround 20 Oct, we decided to postpone our progress on this project and then our team will continue the development progress in sprint 4.

Team meeting 12 (27/October/2023)

Date:

27/October/2023

Attendants:

Wenxuan Chen

Bo Peng

Chenxi Zhang

Sichen Lu

Xuechun Chen

Goals:

- Present progress
- Dicuss what we still need to do in sprint 4
- Assign tasks to each member

Agenda:

Time	Item	Presenter	Notes
10 mins	Present progress	Our team	Present our current progress to other members
10 mins	Discuss our plan in sprint 4	Our team	Gather every team members opinions and dicuss how to make the confuence page better
5 mins	Work allocation	Our team	

Action items

Everyone should polish UI and work together on the user story of "sync work-shift to local calendar"

Wenxuan Chen will be in charge of handover process and relevant documentation

Bo Peng will be in charge of the user story of "sharing avilability" and update cyber security rules

Chenxi Zhang will be in charge of the functionality of "upload profile icon" and "modify profile info type"

Sichen Lu will be in charge of the user story of "sync work-shift to local calendar" and developer handbook.

Xuechun Chen will be in charge of the user story of "deleting account"

And everyone should make a code review

Notes:

None

to be continue

Code Review

Code reviews have been conducted where possible for each feature branch pull request. Participants of each code review posted details on the corresponding pull request pages. Participants used ChatGPT to assist code reviews.

This page provides a brief record of the code reviews.

Code review spreadsheets can be found in the comment section of corresponding pull request pages. Each spreadsheet contains information about who participated in the code review, when did that happen, number of issues identified, etc.

Sprint 2

f4-work-shift-crud

Resolved conflicts, there are some formatting issues, fixed by reviewer.
Lots of code style issues.

- Redundant comments
- Redundant files
- Inconsistent naming

As this is just an initial merge that we've decided to do, these issues can be fixed later on.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/4

f9-availability-pages

Date editing for ad-hoc dates is broken. For example:

- When I set a date "September 15 2023" using the date picker, the date becomes `2023-8-5`.
- When I set a date "December 6 2024" using the date picker, the date becomes `2024-11-5`
- When I set a date "December 25 2024" using the date picker, the date becomes `2024-11-3`

Seems like we're getting 1 month short of the selected month, and the day is a weekday (1 ~ 7) instead of an actual date in that month.

Other issues are minor code style issues.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/11

f12-home-drawer

Mainly code style issues identified by ChatGPT. Some of the recommendations generated by ChatGPT were incorrect, in the end a selected subset of recommendations were addressed.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/9

f2-account-management

Register and login functionality works as expected. Some minor code style issues were identified by ChatGPT and addressed.

Some suggestions include giving meaningful variable names, grouping import statements, etc.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/14

f12-local-storage-encrypt

Note: the feature id of this feature branch should actually be f13 instead of f12, but we're leaving it as-is since no major isssues have arisen.

A minor issue with the timeslot content getting cut off. This was eventually fixed.

Other minor code style issues identified by ChatGPT.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/15

f11-workplace-crud

Some recommendations from ChatGPT include removing unused code (unused imports, log statements, commented-out code).

Separated data storage and validation functionalities into separate files.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/22

f10-work-shift

Mainly code style issues identified by ChatGPT. Removed a lot of unused files and commented-out code.

Future work on the work-shift section of the codebase should focus on organizing the code and keeping the code style consistent.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/25

Sprint 3

f14-settings-ui

Mainly code style issues identified by ChatGPT. Removed a lot of unused files and imports.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/28

f16-profile-db

Refactored backend-related logic for user account data and profile data.

Identified some issues with login and password recovery, which is fixed in the same pull request.

Added `Alert.alert` popups for caught errors.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/30

f17-workplace-db

Refactored backend-related logic for workplace data.

GPT code review workflow was used to conduct automatic code review. Main issues identified include error validation and error handling, which were fixed.

Some issues with code style were fixed.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/32

f18-work-shift-db

Refactored backend-related logic and UI for work shifts.

Some issues with code style was fixed.

There were two issues:

- Work shift edit page data & time fields do not respond to presses on iOS
- Cannot delete work shift from work shift details popup

These issues were fixed in the same pull request.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/33

Newest version of work shift page in sprint 3

Note: This pull request introduces the same changes as f18-work-shift-db, but there was a problem with commits, so it was revoked.

The code review spreadsheet for this pull request also applies to f18-work-shift-db.

Pull request: https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/38

Display pay detail in Workplace

ChatGPT code review identified some minor issues with code style.

https://github.com/COMP90082-2023-SM2/RO-BlueRing/pull/39

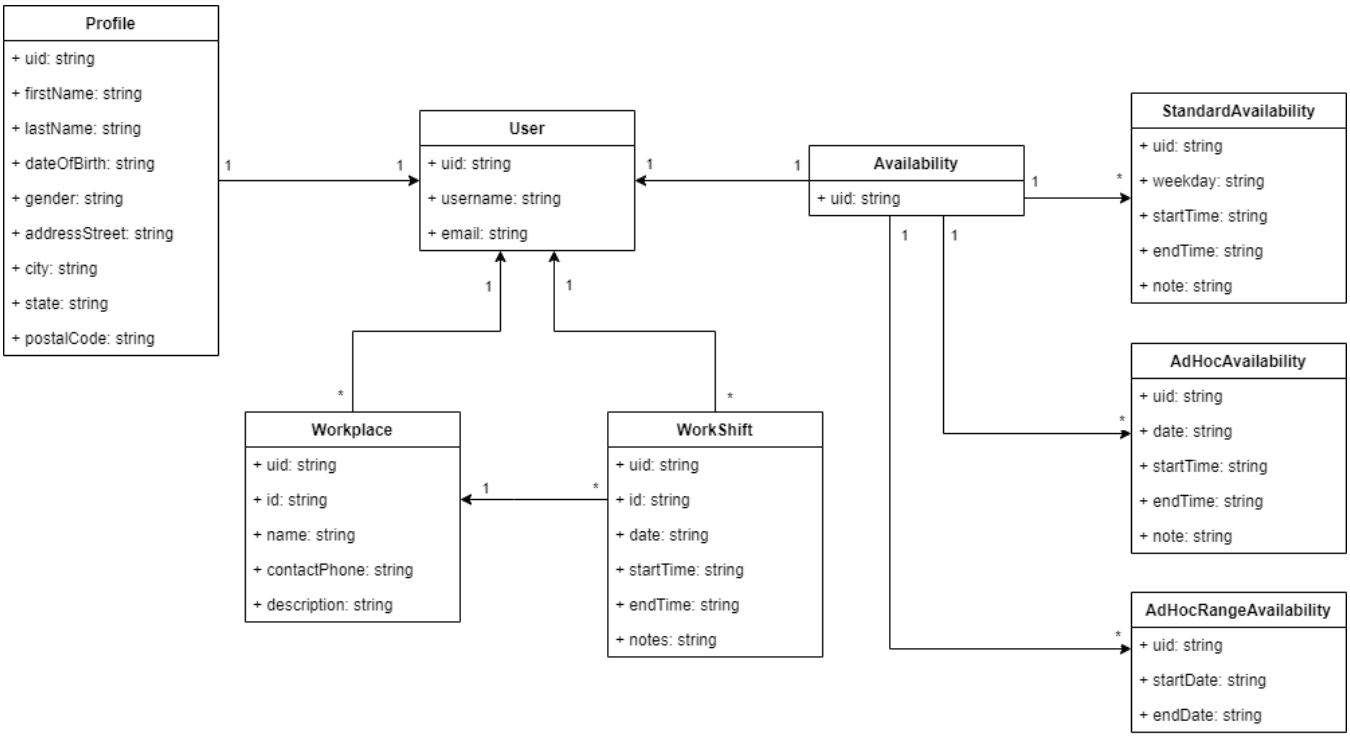
Design

Database Design

Database Conceptual Design

Since we are using a NoSQL database, a UML class diagram is used to represent our database structure (as opposed to other types of diagrams typically used to model SQL databases).

- Each object in this diagram represents a type of data we store in the database.
- The arrow at the tip of a connection line indicates direction of dependency. For example, **WorkShift User** means a **WorkShift** entry is specific to a **User** entry.
- The number or * symbol at each end of a connection line indicates multiplicity. For example, **WorkShift User** is a multiple-to-one relationship.



Data Sample

The following tables show some of the data we might dealing with.

Note that since we are using a NoSQL database, these tables may not accurately represent our database structure. This just serves as an example.

Users

- Each row represents a user's account details
- Usernames (i.e. display name) can be the same
- Emails must be unique
- `userId` is a unique identifier auto-generated by Firebase Authentication on register

userId	username	email
1001	azure	useremail1@gmail.com
1002	azure	useremail2@gmail.com
1003	user3	useremail3@gmail.com

Profiles

- Each row represents a user's personal details (i.e. profile)
- In the furture there may be additional fields as required

userId	firstName	lastName	dateOfBirth	gender	addressStreet	city	state	postalCode
1001	Shemar	Jacobs	01/01/1000	Female	7931 Burke Ave	Melbrourne	VIC	3123
1002	Mervin	Ledner	01/01/1000	Male	1351 Fitzroy St	Melbourne	VIC	3182

Standard Availabilities

- Each row represents a timeslot during which the user is available
- Users are allowed to create multiple timeslots for each weekday

userId	weekday	startTime	endTime	note
1001	Monday	900	1200	content
1001	Monday	1300	1700	content
1002	Wednesday	1300	1600	content

Ad-hoc Availabilities

- Each row represents a timeslot during which the user is available
- Users are allowed to create specific dates and create multiple timeslots for each date

userId	date	startTime	endTime	note
1001	25/12/2023	900	1200	content
1001	26/12/2023	1300	1700	content
1002	12/01/2023	1300	1700	content

Ad-hoc Range Availability will implemented at a later stage.

Workplaces

- Each row represents a workplace that a user creates in their account to record their own employment

userId	id	name	contactPhone	description
1001	1	Macca's	0406-123-456	content
1001	2	KFC	0406-567-890	content
1002	1	UniMelb	0406-123-890	content

Work Shifts

- Each row represents a workshift that a user has planned.

userId	workplaceId	date	startTime	endTime	note
1001	1	01/06/2023	900	1200	content
1001	2	01/06/2023	1330	1700	content
1002	1	01/08/2023	1000	1200	content

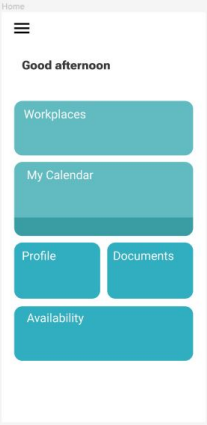
UI Design

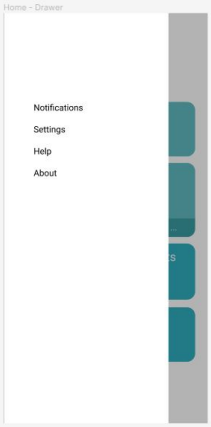
Figma Sketches

Home Page

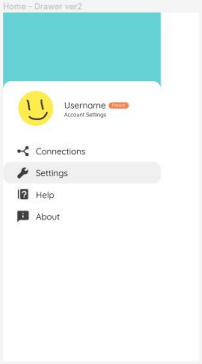
Pros:
- Big display, easy to read
- Using different colours for each feature will make it more intuitive as users get used to the app
  - Or keep things simple and use consistent colour palette (eg. like what Xero.me does)

Bottom navbar?
- Have bottom navbar when in other pages for quick access to important features? (Crucial for good UX)
- Carefully choose which items to include, avoid too much redundancy

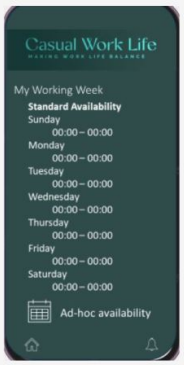


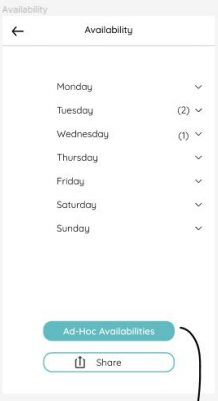


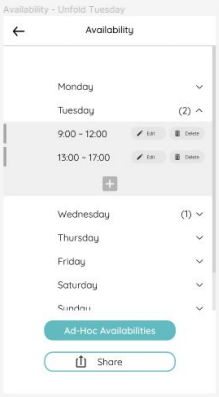
Collapsed Menu
- Account
- Settings
- Help
- About

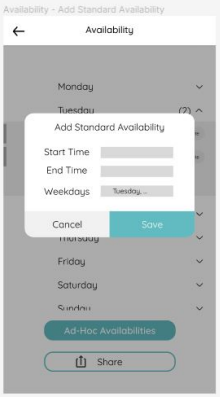


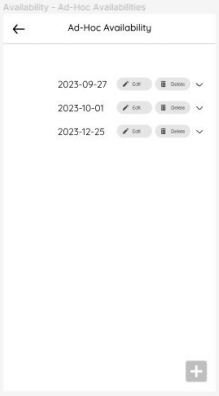
Availability Page





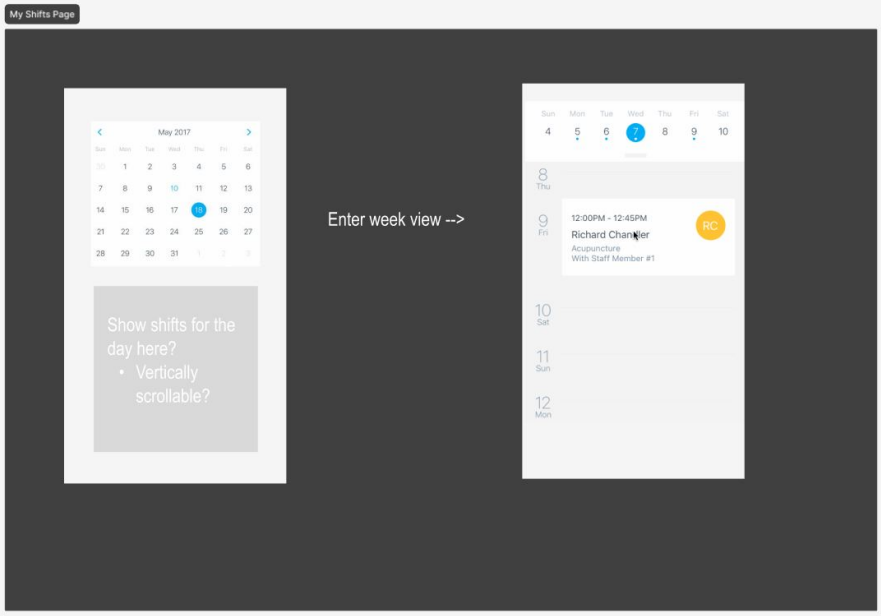






Workplaces
- Manage workplace information
- List out all workplaces
- Each workplace:
  - Name
  - Colour (on “My Shifts” calendar)
  - Other information

My Shifts
- Work shifts calendar



Development

Development Environment

Tools and Online Projects

Component	Tool	Project Link
Frontend IDE	Visual Studio Code	
Version Control	Git, Github Desktop	https://github.com/COMP90082-2023-SM2/RO-BlueRing.git
Backend	Firebase Console	https://console.firebase.google.com/u/1/project/roster-app-2023/overview
Local Backend Emulator	Firebase Local Emulator Suite	
Task board	Trello	https://trello.com/b/DsZtZFFJ/comp90082-ro
Communication	Slack	

Tech Stack

- Development Platform - Expo
- Frontend Framework - React Native
- Language - TypeScript
- Backend - Firebase

Environment Dependencies

Common

- Node.js version 16.13.0 or higher
- npm

Firebase Local Emulator Suite (only if you wish to use the local backend emulator)

- Java JDK version 11 or higher
- Firebase CLI version 8.14.0 or higher

Development Documentation

The development documentation outlines development conventions for our mobile application project RosterApp.

RosterApp File Structure

The project folder for our frontend application is RO-BlueRing/src/Client/roster-app/.

```
- app    // Routing folder. Contains our pages.
- assets    // Any non-code assets, such as images, sounds, etc.
- src
        - api        // Types and functions for interacting with backend.
    - components    // Reusable React components.
    - constants
    - services
        - firebaseConfig.js    // Contains code for communicating with Firebase.
    - styles
```

- The app/ folder should only contain router pages.
- All other components should go into src/components/ where possible.

Import with Path Aliases

Path aliases have been set up for easier importing. This is preferred over relative path imports for more reliability.

- @app corresponds to ./app/
- @assets corresponds to ./assets/
- @components corresponds to ./src/components/
- @constants corresponds to ./src/constants/
- @services corresponds to ./src/services/
- @styles corresponds to ./src/styles/

For example,

❌ DO NOT: import PressableTile from “../../../src/components/PressableTile”

✅ DO: import PressableTile from “@components/PressableTile”

Path aliases can be configured at babel.config.js and tsconfig.json.

To configure path aliases, refer to this example.

Page Navigation

The "app" Folder

We use Expo Router for page navigation. Expo Router is a file-based router built on top of React Navigation.

The app/ folder is a dedicated folder for routing. It contains all of our pages. A page is equivalent to the concept of a "screen" in mobile development. When a file is created in the app folder, it automatically becomes a page in the app, accessible by the router.

- app/index.tsx is the development menu page. It contains buttons that link to other pages of the app.
- app/(drawer)/home/index.tsx is the home page of the app.

Page File Structure

The preferred way to structure a set of page files is to treat each file like a website route. This means:

- Use subdirectories when necessary
- File names and folder names should be all lowercase letters and use dash "-" in place of space.

In the app/ folder, you will probably see two ways of structuring pages:

- ❌ /example-pages/EmptyExample.tsx
- ✅ /example-pages/empty-example/index.tsx

While both ways are valid in Expo Router, the second way is the preferred way in our project.

For the purpose of navigating between pages via page path in the code, the index.tsx at the end of a page path will be ignored, the name of the subfolder will be considered as the name of the route.

- "example-pages/empty-example" is a valid route for /example-pages/empty-example/index.tsx

Page Function Naming

A page file should contain a function with default export. Page function names should have the “Page” postfix to differentiate with other functions.

For example:

example-pages/empty-example/index.tsx

```
const EmptyExamplePage = () => {
  return (
    <View>
      <Text>This is an example page.</Text>
    </View>
  );
};

export default EmptyExamplePage;
```

Page Layout

In the app/ folder, there is a layout file _layout.tsx under some page folders.

Layout files define the navigation layout of all pages under the corresponding folder.

There are many different types of navigation layouts available:

- Stack - Header with a title and a “back” button that allows the user to return to the previous page saved in the navigation stack history.
- Drawer - Collapsed side menu that expands when the button is pressed.
- Tab - Bottom navigation bar.

Layouts can be nested.

app/_layout.tsx is the root stack layout. You can wrap application-level providers in this file.

app/(drawer)/_layout.tsx is the home layout. It adds an expandable drawer for the Home page.

For more information, see https://docs.expo.dev/routing/layouts/

Backend

We use Firestore database provided by our Firebase backend to store data remotely.

Database

Types

The src/api/types/ folder contains type definitions corresponding to each type of document in Firestore. Each file under this folder contains:

- A frontend type definition
- A Firestore Converter definition to convert data between the frontend type and the Firestore document

A frontend type definition usually corresponds to what a document looks like in Firebase.

The Firestore Converter definition is intended to be used in conjunction with firestore api functions, such as getDoc(docRef.withConverter(...)).

To use a type definition elsewhere, simply import it, for example:

```
import { WorkShift } from "@api/types/WorkShift";
```

Database Operations

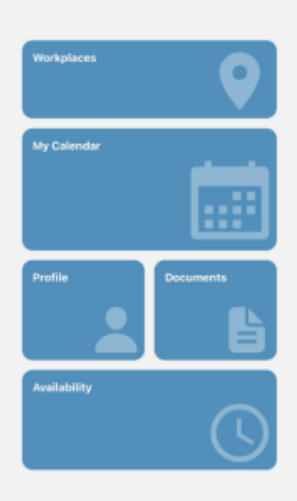
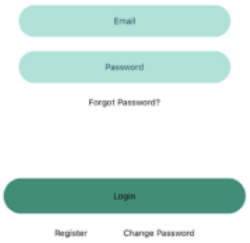
The src/api/db/ folder contains wrapper functions for database CRUD operations that deal with the data types defined in src/api/types.

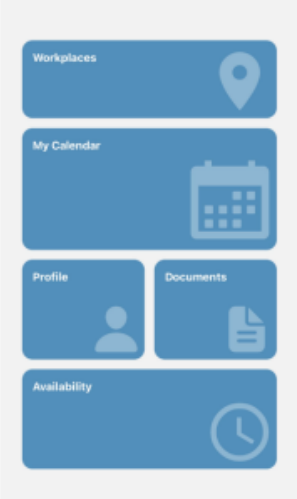
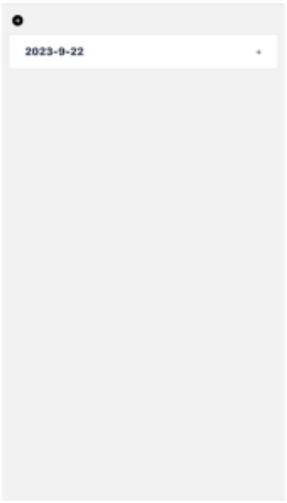
To use specific databse operation functions for some defined data type, simply import them, for example:

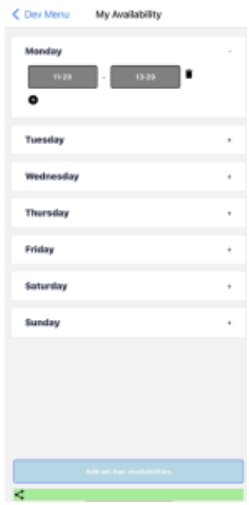
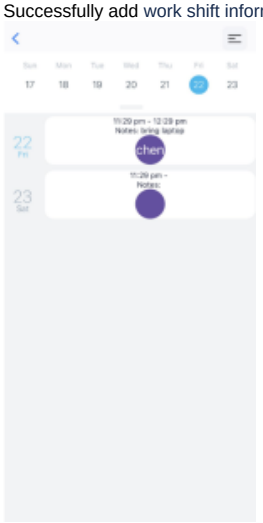
```
import { createWorkShift, getWorkShift, updateWorkShift } from "@api/db/workShiftDatabase";
```

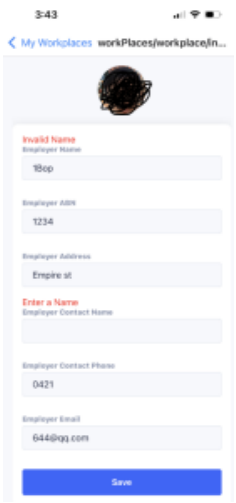
Testing

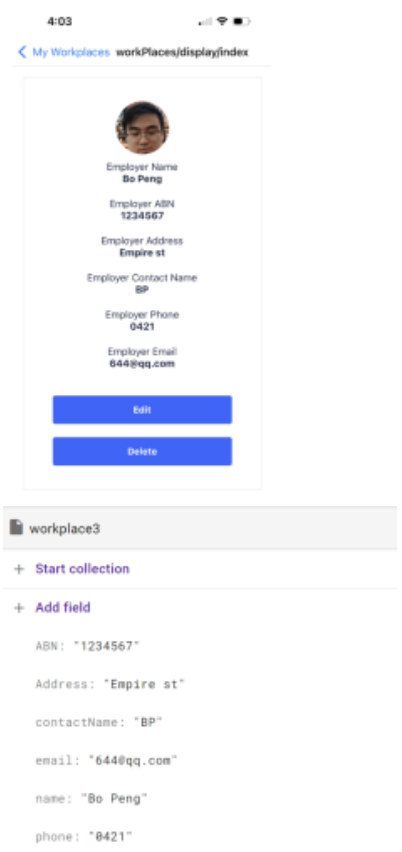
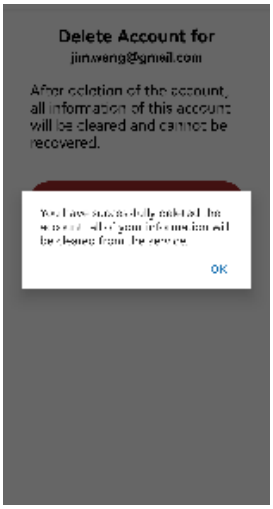
Test Case

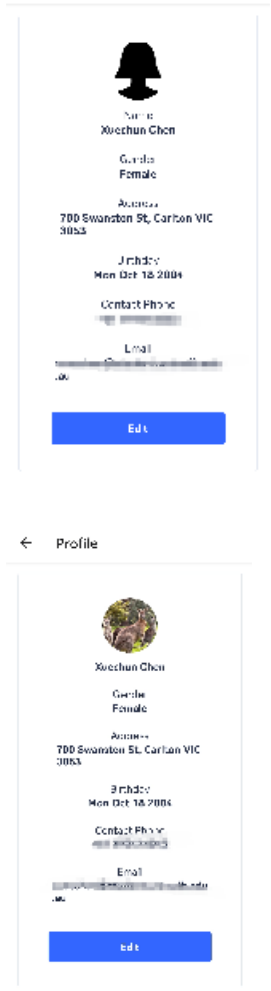
User story ID	Test Case	Test Description	Operation Steps	Result	Actionable Outcomes
1	Account registration	User can successfully regisitor an account with customized username, and password by email authentication	• Go to registration page, click 'register; button • Type in email, username, password and see if user can successfully go to the home page of APP	Successfully directed to home page  	
2	log in	User can successfully log in with the validated username and password	• Go to login page, type in username and password • Click 'log in' button	Successfully log in to home page 	

					
3	Normal password reset	This function is not fully implimented in sprint 2			
4	Forgotten Password Reset	This function is not fully implimented in sprint 2			
5	Edit Account	This function is not fully implimented in sprint 2			
6	Delete my Account	This function is not fully implimented in sprint 2			
9	Set ad-hoc availabilities	User can successfully select time slut and add it into availibility page	• Click the button with plus icon • Click the pen icon to select a date • Click confirm to confirm a date	Successfully add ad-hoc availabilities 	
10	Set standard availabilities	User can successfully select time slut and add it into availibility page	• Select day by clicking the day card • Select time slut by clicking the gray button • Scrow up and down to select start time and end time	Successfully add standard availabilities	

					
11	Share availabilities	This function is not fully implimented in sprint 2			
13	Edit personal information	User input invalid value, the system will display the error information	• Input the workplace details • click the 'save' button	The validation for each input value's format is not accurate enough. Although an error message is returned, the format doesn't meet the requirements	Rewrite the regular expression function used to validate whether the input values conform to the specified format
20	Add, edit work shift information	User can successfully add work-shift information by select date, time, add notes, add employer name and also able to edit those information	• Click meum button • Click the 'select date' button to select a date on calendar • Click the 'select start time' button to select start time • Click the 'select end time' button to select end time • Type in notes in the column below 'enter notes' • Type in notes in the column below 'enter employer name' • Click the 'Confirm' button to submit	Successfully add work shift information 	Edit work shift function is not implimented. Thus, this function should be added in the next sprint
24	Add workplace on the workplace main page	User input invalid value, the system will display the error information	• Input the workplace details • click the 'save' button	The validation for each input value's format is not accurate enough. Although an error message is returned, the format doesn't meet the requirements	Rewrite the regular expression function used to validate whether the input values conform to the specified format

					
24	Display each workplace name on the screen	The screen can display each workplace that the user added into the Firestore	• Click the 'add' icon • Complete the edit workplace information • Complete the storage of workplace details	The name of the workplace stored in the database can be successfully added to the screen	
24	Delete the certain workplace from Firestore	The workplace successfully deleted from Firestore and the workplace name removed from the workplace main page	• Click the workplace name showed on the workplace main page, and the the page will be navigated to the workplace information display page • Click the 'Delete' button	The workplace name is removed from the workplace main page, and the corresponding workplace information is deleted from Firestore (Workplace main page) 	
24	The display page of workplace can load the details of workplace stored in the Firestore	The workplace display page can load the information of workplace from Firestore and display the information on the page	• In the workplace main page, click the workplace name	The information of the workplace correctly corresponds to the workplace I selected on the main screen, and the information is loaded onto the screen	

					
6	Account Deletion	User can choose to delete their account and the account information will be cleared from the firebase console.	• Log in an account. • navigate through Home side navigation settings my account delete account. • press "delete account" button. • check the account records in firebase console.	After account deletion, the account can not be logged in and the account record in Firebase console has been cleared. 	
13	Edit Profile Photo	User can choose to upload a picture from their photo as their profile photo on the app.	• Log in an account • Navigate through Home Profile • Press "Edit" button • Tap Image Icon • Choose photo from the phone and upload • Save	The profile photo changes from the default image to the image chosen from the phone.	

				 Xiaochun Chen Female 700 Swanston St, Carlton VIC 3053 Mon Oct 16 2006 Edit Profile Xiaochun Chen Female 700 Swanston St, Carlton VIC 3053 Mon Oct 16 2006 Edit	

Cybersecurity

In an era of digital reliance, the security and privacy of user data are paramount, which is also the priority concern of our client. Rostering apps, which handle sensitive employee information and company logistics, sit at the intersection of this concern. With regulations like GDPR emphasizing data rights, the stakes are high. As we develop our rostering application, our commitment is clear: to seamlessly integrate top-tier cybersecurity measures, ensuring users can manage schedules confidently, knowing their data is both secure and private.

Security Implementation Checklist

Category	Planning Implementation	Status
Authentication	Registration	Completed
	Login	Completed
	Token-based Authentication	To be decided
	Logout Mechanism	In progress
	Password Recovery	Completed
	Continuous Authentication	To be decided
	Security Enhancements	In progress
	Third-party Authentication	To be decided
Access Control	Role-based Access Control	In progress
Encryption	Data Encryption at Rest	Completed
	Date Encryption in transit	Completed
Integrity	Hashing for Data Verification	To be decided
	Timestamps for Monitoring Data Changes	To be decided
	Firestore Security Rules	To be decided
	Transactional Operations	To be decided
	Regular Audits	To be decided
Non-repudiation	Logging and auditing	In progress
Performance Optimisation	React Native Performance Optimization	In progress
	Firebase Performance Optimization	In progress
Secure Configuration	Application Configuration	In progress
	Firebase Configuration	In progress

Considerations

1. **Authentication and Authorization**:
    - Firebase Authentication: Use Firebase Authentication for user sign-up and login. It offers various methods, such as email/password, phone, and third-party providers (Google, Facebook, etc.).
    - Role-based Access: Implement role-based access control (RBAC) to restrict functionalities based on user roles (e.g., Admin, Manager, Employee).
    - Multi-factor Authentication (MFA): Enable MFA to add an extra layer of security for user accounts.
2. **Data Security**:
    - Firestore Security Rules: Secure the Firestore database by writing appropriate security rules. Ensure that only authenticated users can read/write data, and further refine access based on roles.
    - Data Validation: Always validate data on the server side before saving it to the database.
    - Data Encryption: If storing sensitive information, consider encrypting the data before saving it. Firebase Realtime Database supports client-side encryption.
3. **Network Security**:
    - HTTPS: Ensure all network communications from the React Native app to Firebase or any other service are over HTTPS.
    - CORS: If interacting with other APIs, ensure proper Cross-Origin Resource Sharing (CORS) settings are in place to prevent unauthorized domains from making requests.
4. **Codebase Security**:
    - Dependencies: Regularly check and update the dependencies. Tools like `npm audit` can be used to identify and fix known vulnerabilities in the node packages.
    - Secret Management: Never commit API keys or secrets in the codebase. Use environment variables or Firebase's cloud functions configurations for this purpose.
    - Code Reviews: Regular code reviews can help catch security flaws before they make it to production.
5. **Threat and Vulnerability Awareness**:
    - Keep Updated: Regularly follow React Native and Firebase's official documentation and communities to stay informed about any known security threats or vulnerabilities.
    - Penetration Testing: Consider hiring third-party services to perform penetration testing on the app to identify potential vulnerabilities.
6. **Incident Response**:
    - Monitoring: Set up monitoring and logging to detect unusual activities.
    - Backup: Regularly backup the Firestore data to ensure it can be restored in case of any security incidents.
    - Plan: Have a plan in place detailing the steps to take in case of a security breach.
7. **End-User Security Awareness**:
    - Educate: Educate users about the importance of strong passwords, not sharing credentials, and recognizing phishing attempts.
    - Updates: Regularly push updates to the app, addressing any security issues or vulnerabilities.
8. **Mobile Specific Considerations**:
    - Local Storage: If storing data locally on the device, consider encrypting it. Be cautious about what data is stored, especially if it's sensitive.
    - Permissions: Only request necessary permissions from the user.

Threats and Vulnerabilities

1. Identify Assets:

Asset	Description
User Data	Personal details, email addresses, phone numbers, and passwords.
Roster Data	Shift details, employee roles, schedules, locations.
Application Code	The source code of the app.
Configuration Data	API keys, database connection strings, and other configuration data.

2. Define Threat Actors:

Threat Agent	Description
External Attackers	Individuals or groups trying to exploit vulnerabilities from outside.
Insiders	Employees or users with malicious intent.
Accidental Actors	Users who accidentally cause data breaches or losses.

3. Identify Threats using STRIDE:

Asset	STRIDE Category	Potential Threat	Potential Threat Actor	Likelihood	Justification
User Data	Spoofing	Impersonate a user to gain unauthorized access.	External Attacker	High	User data is often targeted for financial gains, identity theft, and fraud.
	Tampering	Maliciously modify user data.	Insider	Medium	Insiders have access, but strict controls can reduce risk.
	Repudiation	Users deny performing an action on their data, but there's no way to prove otherwise.	Accidental Actor	Low	More of a user error than a malicious act.
	Information Disclosure	Unauthorized disclosure of user data.	External Attacker	High	User data leaks are lucrative for cybercriminals.
	Denial of Service	Make user data unavailable.	External Attacker	Medium	DDoS attacks are prevalent, but mitigation tools exist.
	Elevation of Privilege	Unauthorized users gaining higher privileges and accessing data they shouldn't.	Insider	Medium	Proper controls can minimize, but there's always a risk.
Roster Data	Spoofing	Impersonate an admin/manager to modify roster.	Insider	Medium	Insiders might impersonate others to cause disruption.
	Tampering	Unauthorized modifications to the roster.	External Attacker	High	Altering roster data can disrupt operations.
	Repudiation	Users or admins deny making changes to the roster without evidence of the action.	Accidental Actor	Low	Audit trails and versioning can mitigate this risk.
	Information Disclosure	Unauthorized access to schedules and roles.	External Attacker	Medium	Roster data can reveal company operations.
	Denial of Service	Preventing access to the roster system, disrupting operations.	External Attacker	Medium	Attackers might disrupt the rostering process.
	Elevation of Privilege	An employee gaining manager or admin rights and making unauthorized changes.	Insider	Medium	Insiders might attempt to gain unauthorized privileges.
Application Code	Tampering	Unauthorized changes to the application logic.	External Attacker	High	Vulnerabilities in application code are often exploited.
	Information Disclosure	Exposure of application logic which could reveal vulnerabilities.	External Attacker	Medium	Exposure can reveal system vulnerabilities.
Configuration Data	Spoofing	Using configuration data to impersonate the app or service.	External Attacker	High	Configuration data provides a system blueprint.
	Tampering	Modifying configuration to disrupt the app's operations.	Insider	Medium	Malicious insiders might alter configurations.
	Information Disclosure	Unauthorized disclosure of API keys or database configurations.	External Attacker	Extreme	Configuration data, especially API keys, are prime targets.

4. Proposed Countermeasures:

STRIDE Category	Countermeasure
Spoofing	- Implement strong authentication mechanisms. - Use Multi-Factor Authentication (MFA).
Tampering	- Implement data integrity checks. - Use Role-Based Access Control (RBAC).
Repudiation	- Implement comprehensive logging and auditing mechanisms.
Information Disclosure	- Encrypt sensitive data both in transit and at rest. - Use secure configurations.
Denial of Service	- Use rate-limiting. - Regularly backup data. - Optimize performance.
Elevation of Privilege	- Strictly enforce user roles. - Regularly review access controls.

Planning Implementation

Authentication

1. Registration:

a. User Details Collection:

- **Email**: Collect the user's email. Firebase requires the email to be unique.
- **Password**: Enforce a strong password policy. Firebase allows passwords with a minimum of 6 characters by default.

Strong Password Policy:

1. **Length**: Minimum 12 characters.
2. **Complexity**:
    - At least 1 uppercase letter.
    - At least 1 lowercase letter.
    - At least 1 number.
    - At least 1 special character (e.g., !, @, #).
3. **No Personal Info**: Exclude user's name, username, or company name.
4. **Avoid Common Choices**: No common passwords (e.g., "password123") or sequences (e.g., "12345").
5. **Reuse**: Can't reuse last 5 passwords.
6. **Account Lockout**: After 5 failed attempts, lock for 15 minutes.

b. Data Validation:

- **Firebase SDK**: The Firebase client SDK will handle basic validation, but always validate data again server-side using Firebase Cloud Functions if needed.

https://firebase.google.com/docs/functions

c. Password Handling:

- **Hashing and Salting**: Firebase Authentication automatically handles the secure storage, hashing, and salting of user passwords.

d. Account Verification:

- **Email Verification**: Firebase provides built-in methods to send email verification links to users.

2. Login:

a. Credential Verification:

- Use Firebase's `signInWithEmailAndPassword` function to verify user credentials.

b. Session Management:

- **Firebase Authentication**: Firebase manages user sessions and provides tokens (JWTs) that can be used to secure your backend.

c. Multi-Factor Authentication (MFA):

- **Firebase MFA**: Firebase supports multi-factor authentication using SMS. Set it up in the Firebase Console and implement it in the app using Firebase SDK.

d. Failed Login Attempts:

- Monitor authentication errors returned by Firebase and implement a mechanism to handle repeated failed attempts, like introducing CAPTCHAs or temporarily disabling the account.

3. Token-based Authentication:

- **Firebase ID Tokens (JWTs)**: Firebase provides JWTs for user authentication. These can be used to secure Firebase Cloud Functions or backend services.

4. Logout Mechanism:

- Use Firebase's `signOut` function to log out users and invalidate their session.

5. Password Recovery:

- Firebase provides built-in methods to send password reset emails to users.

6. Continuous Authentication:

- **Firebase Realtime Database or Firestore Rules**: Implement rules to check for user authentication and permissions for database operations.

7. Security Enhancements:

a. HTTPS:

- Firebase services are always accessed over HTTPS. Ensure that any other backend services are also secured with HTTPS.

b. Account Lockout:

- Implement custom logic using Firebase Cloud Functions to lock accounts after multiple consecutive failed login attempts.

c. Logging:

- Use Firebase's logging and auditing capabilities to monitor authentication events.

d. Rate Limiting:

- Firebase Cloud Functions can be integrated with third-party services or custom logic to implement rate limiting.

8. Third-party Authentication:

- **Firebase Authentication Providers**: Firebase supports third-party authentication providers such as Google, Facebook, Twitter, GitHub, etc. Enable these methods in the Firebase Console and use Firebase SDK to integrate them into the app.

a. Google Sign-In:

- Enable Google as a sign-in provider in the Firebase Console.
- Use Firebase SDK to authenticate users with their Google accounts.

b. Facebook Sign-In:

- Set up a Facebook App, and then enable Facebook as a sign-in provider in the Firebase Console.
- Integrate using Firebase SDK.

c. Other Providers:

- Similar steps can be taken for other providers supported by Firebase.

Role-Based Access Control

- **`Admin`**: Can read, write, update, delete all data.
- **`Manager`**: Can read, write, and update certain data but not delete.
- **`Editor`**: Can read and update certain data.
- **`Viewer`**: Can only read certain data.

Configure security rules in Firebase:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {

    match /someCollection/{docId} {
      // Only allow read access for viewers
      allow read: if get(/databases/$(database)/documents/users/$(request.auth.uid)).data.role == 'Viewer';

      // Allow read, write, and update for managers
      allow read, write, update: if get(/databases/$(database)/documents/users/$(request.auth.uid)).data.role ==
'Manager';
    }
  }
}
```

Encryption

1. Data Encryption at Rest:

Firebase automatically encrypts your data at rest. This includes:

- **Firestore and Firebase Realtime Database**: Data is encrypted under 256-bit Advanced Encryption Standard (AES-256).
- **Firebase Storage**: Files are encrypted using the same AES-256 standard.
- **Firebase Authentication**: Passwords are hashed using bcrypt.

2. Data Encryption in Transit:

Firebase ensures that all data in transit between Firebase services and Firebase SDKs is encrypted using HTTPS and Transport Layer Security (TLS).

Data Integrity Check

1. Hashing for Data Verification:

Generate a cryptographic hash of the data before saving it. Later, recompute the hash and compare it to ensure the data has not been tampered with.

2. Timestamps for Monitoring Data Changes:

Add timestamps to data entries. Monitor when data was created or last modified, helping to ensure that data isn't being changed inappropriately.

3. Firestore Security Rules:

Use Firestore's security rules to ensure data integrity by:

- Restricting write access only to authorized users.
- Validating the structure and type of data being written.
- Preventing modifications to certain fields after they've been set.

4. Transactional Operations:

When performing operations that involve reading data, processing it, and then writing it back, use Firestore transactions. This ensures that if the data changes after reading and before writing, the operation is retried, preserving data integrity.

```
const db = admin.firestore();

db.runTransaction(async (t) => {
    const docRef = db.collection('collectionName').doc('docId');
    const doc = await t.get(docRef);

    // ... perform data operations ...

    t.set(docRef, updatedData);
});
```

5. Regular Audits:

Periodically audit the data and its integrity:

- Compare data with backup copies.
- Recompute hashes and compare with stored hashes.
- Check for anomalies, like data entries without corresponding integrity checks.

Logging and Auditing

1. Define Logging Levels:

- **ERROR**: System errors, exceptions, and critical issues.
- **WARNING**: Potentially harmful situations or possible misconfigurations.
- **INFO**: Regular operations, user activities, and general information.
- **DEBUG**: Detailed information, mainly used for debugging.

2. Identify Key Events to Log:

- **User Activities**:
  - User registration, login, logout
  - Password changes or reset requests
  - Role changes
  - Multi-factor authentication events
- **Roster Activities**:
  - Creating, updating, deleting shifts
  - Creating, , updating, deleting availabilities
  - Changes to employee roles or schedules or workplaces
- **System Activities**:
  - System startups or shutdowns
  - Configuration changes
  - Service interruptions

3. Logging Mechanism:

a. Client-Side (React Native):

- Use a logging library (e.g., Pino or `Winston`) to capture client-side logs.
- Store logs temporarily locally, and periodically send them to the backend for permanent storage and analysis.

7 best Node.js logging libraries and aggregators

b. Server-Side (Firebase Cloud Functions):

- Use Firebase's built-in logging for Cloud Functions.
- Capture and store logs in Firebase's Firestore or an external logging system (e.g., ELK Stack, Splunk).

4. Log Format:

Each log entry should contain:

- **Timestamp**: Precise time of the event.
- **User ID**: Identifier for the user performing the action.
- **Log Level**: ERROR, WARNING, INFO, DEBUG.
- **Event Type**: A brief description of the event.
- **Details**: Detailed information about the event.
- **Source**: Component or module where the event occurred.
- **IP Address**: IP address of the user or system initiating the event (if applicable).

5. Auditing Mechanism:

- **Regularly Review Logs**: Periodically review logs to identify any abnormal patterns or suspicious activities.
- **Automate Analysis**: Use tools or scripts to analyse logs and detect anomalies automatically.
- **Access Control for Logs**: Ensure only authorized personnel can access and modify the logs.
- **Integrity**: Ensure logs cannot be tampered with. Consider cryptographic mechanisms to guarantee the integrity of the logs.

6. Data Retention and Backup:

- **Retention Policy**: Define how long logs will be retained. For instance, retain critical logs for at least a year and non-critical logs for a month.
- **Backup**: Regularly back up logs to a secure location to ensure they're available when needed, especially for forensic purposes.

7. Alerting Mechanism:

- **Real-time Alerts**: Implement real-time alerting for critical events (e.g., multiple failed login attempts, unauthorized access attempts).
- **Notifications**: Send notifications through email, SMS, or other communication tools to inform relevant personnel about the alerts.

8. Compliance:

Ensure the logging and auditing mechanism complies with any relevant legal or industry-specific regulations (e.g., GDPR, HIPAA).

https://gdpr-info.eu/

https://www.cdc.gov/phlp/publications/topic/hipaa.html

Performance Optimisation

React Native Performance Optimization:

1. **Use Hermes**: Hermes improves startup time, decreases memory usage, and reduces app size.
2. **Optimize Renders**:
    - Apply `React.memo` for functional components to prevent unnecessary renders.
    - For class components, leverage `shouldComponentUpdate` or extend `PureComponent`.
    - Avoid inline functions and objects in JSX.
3. **Virtualized Lists**: For long lists of data, utilize `FlatList` or `SectionList` which only render items currently on screen.
4. **State Management**: If using Redux or MobX, optimize state updates to prevent unnecessary re-renders.
5. **Navigation**: Optimize navigators. For instance, with React Navigation, prefer `createStackNavigator`.
6. **Images**:
    - Compress images and use appropriate resolutions.
    - Use the `FastImage` library which offers better performance than the standard `Image` component.
7. **Reduce JavaScript Bundle Size**: Remove unused modules and libraries. Consider code splitting.

Firebase Performance Optimization:

1. **Firestore Queries**:
    - Fetch only necessary data using precise queries.
    - Utilize indexes for complex queries.
    - Limit data retrieval with `limit()`.
2. **Realtime Database**:
    - Adopt flatter data structures to reduce nested data.
    - Fetch data precisely to minimize overhead.
3. **Firebase Storage**:
    - Compress images or files before uploading.
    - Retrieve media with appropriate resolutions.
4. **Caching**:
    - Leverage Firebase's offline capabilities for faster data access.
    - Cache images or other media on the device after retrieval.
5. **On-device Data Store**: If real-time sync isn't required for all data, consider storing some data locally using SQLite or Realm and sync periodically.
6. **Firebase Cloud Functions**:
    - Ensure functions are lightweight and reduce the amount of processing.
    - Use appropriate memory and timeout settings.
7. **Monitor with Firebase Performance Monitoring**: Utilize Firebase's Performance Monitoring tool to identify and fix performance bottlenecks.

Secure Configuration

Application Configuration:

- **Session Management**: Ensure secure session management with timeout features and secure cookie flags.

By default, Firebase's authentication state persists even if the app is restarted. This is achieved using a combination of tokens and secure storage appropriate for each platform.

Firebase provides various session persistence levels:

- `LOCAL` (default): Auth state persists even if the app is closed.
- `SESSION`: Auth state persists only until the app or browser tab is closed.
- `NONE`: Auth state persists only in memory and is lost when the app is closed.

```
firebase.auth().setPersistence(firebase.auth.Auth.Persistence.SESSION)
  .then(() => {
    return firebase.auth().signInWithEmailAndPassword(email, password);
  });
```

- **Logging**: Ensure applications log relevant security events, but avoid logging sensitive information.
- **Error Handling**: Properly handle errors to prevent information leakage. Avoid exposing stack traces or internal system details to end users.

Firebase Configuration:

- **Firebase Security Rules**: Ensure Firestore and Firebase Storage security rules are properly configured to restrict access.
- **Firebase Authentication**: Configure authentication methods securely, ensuring only necessary providers are enabled.
- **API Key Restrictions**: Restrict Firebase API keys in the Google Cloud Console to ensure they can only be used for intended purposes.

Ethical Considerations

Designing a rostering app that shares information between employees and employers entails various ethical considerations to ensure fairness, transparency, and respect for all parties involved. Here are some ethical considerations related to our app:

Privacy and Data Security

Likelihood: High

Justification: Data breaches have become increasingly common, and the sensitive nature of employee data makes the rostering app a potential target. Non-compliance can result in severe penalties and loss of trust among users.

- **Employee Data**: In the digital age, the risks of data breaches and misuse have increased. Employees' personal information might include details such as their names, addresses, phone numbers, and bank details. If this data falls into the wrong hands, it could lead to identity theft or other forms of fraud.
- **Confidentiality**: Improper data storage and transmission methods might leave the data vulnerable to interception. Therefore, using encryption techniques to safeguard the data is essential to ensure that even if intercepted, it cannot be improperly interpreted or misused.

Transparency

Likelihood: Medium

Justification: As the app matures, users will demand transparency in how data is used and how decisions are made. Failure to provide this can result in decreased user trust and potential legal issues.

- **Audit Trails**: To avoid any misunderstandings or conflicts, especially when it comes to changes in the roster, maintaining a detailed record is crucial. This also helps trace and resolve any issues that might arise.
- **Clear Policies**: The absence of clear policies and guidelines might lead to confusion and inconsistent practices, which could affect the satisfaction of employees and the efficiency of the organization.

Fairness and Equality

Likelihood: High

Justification: Discrimination, even if unintentional, can result in significant legal and reputational risks. As the workforce becomes more diverse, the demand for fair treatment and equal opportunities will increase.

- **Equal Opportunities**: In a diverse society, ensuring all employees are treated fairly is paramount. Any form of bias or discrimination can result in ethical and legal consequences.

1. **Blind Recruitment**: Remove personal details that might reveal the age, gender, ethnicity, or other personal attributes of an applicant or employee when considering them for shifts, promotions, or other opportunities. This helps reduce unconscious bias in decisions.
2. **Clear Policies**: Establish and communicate clear policies that promote equal opportunities. This includes guidelines on how rosters are created and changed, ensuring all staff are aware of the principles guiding these processes.

- **Avoiding Discrimination**: Discriminatory practices can not only lead to ethical and legal issues but can also tarnish the company's reputation, resulting in employee attrition and legal actions.

1. **Legal Framework**: Stay updated with local and international laws related to employment and discrimination. Ensure that the rostering system adheres to these laws.
2. **Grievance Redressal**: Establish a clear system where employees can raise concerns or grievances related to discrimination or unfair treatment. Ensure that these concerns are addressed promptly and transparently.
3. **Encourage Reporting**: Create an environment where employees feel safe reporting discriminatory actions or biases without fear of retaliation.

User Experience

Likelihood: High

Justification: A poor user experience can result in decreased adoption and usage of the app. As users become more tech-savvy, their expectations for usability and accessibility will also rise.

- **Accessibility**: A significant number of disabled individuals exist globally, and they too require access to various applications. Overlooking their needs might exclude a potential user base.

Designing for Colour Blindness

Designing for colour blindness is a critical aspect of inclusive design, ensuring that digital interfaces are accessible to a broader audience. Colour blindness, or colour vision deficiency, affects a significant portion of the population. Here are key considerations and best practices:

1. **Use High Contrast**: Regardless of colour choices, ensure that there's a high contrast between foreground and background elements.
2. **Avoid Colour-Only Indicators**: Don't use colour as the only visual means of conveying information, indicating an action, prompting for a response, or distinguishing a visual element. For instance, if you're using color to highlight a required field in a form, also use an asterisk (*) or another indicator.
3. **Use Patterns and Textures**: In charts or graphics, use patterns or textures in addition to colour to differentiate between elements.
4. **Test the Designs**: Utilize tools and software that simulate various types of colour blindness, such as:
    - Colour Oracle
    - Sim Daltons
    - Coblis — Colour Blindness Simulator
5. **Be Mindful of Colour Combinations**: Some colour combinations are problematic for those with colour vision deficiencies.
6. **Use Symbols and Icons**: Incorporate icons or symbols alongside colour cues. For example, traffic lights could be represented with the shapes triangle (for green/up), circle (for amber/wait), and square (for red/stop).

https://www.getfeedback.com/resources/ux/how-to-design-for-color-blindness/

https://davidmathlogic.com/colorblind/#%23D81B60-%231E88E5-%23FFC107-%23004D40

- **Intuitive Design**: The system should be intuitive, meaning that users should be able to use it without requiring extensive training or guidance.
- **Consistency**: Consistent design means using the same symbols, terms, and layouts for similar tasks and scenarios. This reduces the learning curve for users.
- **Clear Hierarchies**: Organize information and actions in clear, logical hierarchies. The most important or common actions should be easily accessible.
- **Effective Use of Language**: Use simple and clear language. Avoid jargon unless it's appropriate for the target audience.
- **Minimize Cognitive Load**: Don't overwhelm users with excessive information or choices. Chunk information and use progressive disclosure techniques.
- **Mobile Responsiveness**: Ensure that the system is usable on various devices, especially given the increasing use of mobile devices.
- **Avoid Unnecessary Interruptions**: Pop-ups, modals, and other interruptions can hinder usability if not used judiciously.

Work-Life Balance

Likelihood: Medium

Justification: Burnout and mental health issues are increasingly being recognized in the modern workplace. Organizations not respecting work-life balance may face backlash and reduced employee morale.

- **Respecting Off-Time**: Excessive working and fatigue can have adverse effects on the health and well-being of employees, consequently affecting their work efficiency and satisfaction.
- **Flexibility**: The modern workforce increasingly seeks work flexibility to cater to their personal and family needs. Not providing flexibility can lead to employee dissatisfaction and higher attrition rates.

Legal Compliance

Likelihood: Extreme

Justification: With stringent regulations like GDPR and the Australian Privacy Act, non-compliance is not an option. Organizations can face severe financial penalties, legal actions, and reputational damage.

- It's a legal obligation for every organisation to comply with labour laws and other related regulations. Non-compliance can result in fines, legal actions, and damage to the organization's reputation.

Informed Consent

Legal Framework: Worldwide

GDPR is a European regulation, it has set the standard for many countries worldwide in terms of data protection.

In the wake of GDPR and other data protection regulations, ensuring users are aware of how their data is collected and used is of utmost importance. Collecting or using data without proper consent can lead to legal issues and fines.

Here are some key points and requirements of GDPR:

1. **Data Protection Principles**: Organizations must handle personal data lawfully, fairly, and transparently. The data should be collected for specified and legitimate purposes and not further processed in a way that's incompatible with those purposes.
2. **Rights of Data Subjects**: GDPR gives individuals several rights, including:
    - **Right to be informed**: Individuals have the right to know how their data will be used.
    - **Right of access**: Individuals can request a copy of the data held about them.
    - **Right to rectification**: Individuals can ask for incorrect data to be corrected.
    - **Right to erasure**: Also known as the 'right to be forgotten', individuals can request that their data be deleted.
    - **Right to restrict processing**: Individuals can ask that the use of their data be limited.
    - **Right to data portability**: Individuals can ask for their data in a machine-readable format or have it transferred to another controller.
    - **Right to object**: Individuals can object to their data being used for specific purposes, such as direct marketing.
    - **Rights related to automated decision-making and profiling**: Individuals can request human intervention or challenge decisions made without human involvement.
3. **Consent**: Organizations must obtain clear, affirmative consent to process data. This consent can't be buried in long terms and conditions; it must be separate and easily understandable.
4. **Data Breach Notifications**: Organizations are required to report certain types of data breaches to the relevant supervisory authority within 72 hours of becoming aware of the breach.
5. **Data Protection Impact Assessment (DPIA)**: In situations where data processing may result in high risks to data subjects, organizations must conduct a DPIA to assess and mitigate those risks.
6. **Data Protection Officer (DPO)**: Organizations that engage in large-scale monitoring or processing of special categories of data must appoint a DPO. The DPO ensures the organization's compliance with GDPR.
7. **Data Minimization and Purpose Limitation**: Organizations should only collect data that's necessary for its intended purpose and shouldn't keep it for longer than needed.
8. **International Data Transfers**: Transferring data outside the EU is subject to restrictions unless certain conditions or safeguards are met.
9. **Penalties**: Organizations that don't comply with GDPR can face hefty fines. For serious infringements, fines can go up to €20 million or 4% of the firm's worldwide annual revenue from the preceding financial year, whichever is higher.

Legal Framework: Australia

In Australia, the primary legislation that governs the collection, use, and disclosure of personal information by private sector organizations is the *Privacy Act 1988*.

The act includes the Australian Privacy Principles (APPs) which set out standards for handling personal information.

1. **Consent and Collection**: Under the APPs, organizations should only collect personal information that is necessary for their functions or activities. The collected information must be done lawfully and fairly, without undue intrusion.
2. **Use and Disclosure**: Personal information should not be used or disclosed for purposes other than its original intention without the individual's consent, unless a specific exemption applies.
3. **Data Security**: Organizations must take reasonable steps to protect the personal information they hold from misuse, interference, and loss, as well as from unauthorized access, modification, or disclosure.
4. **Access and Correction**: Individuals have the right to access their personal information and request corrections if necessary.
5. **Direct Marketing**: Organizations are generally prohibited from using personal information for direct marketing purposes unless an exemption applies.
6. **Cross-border Data Transfers**: Before an Australian entity can disclose personal information to an overseas recipient, they must take reasonable steps to ensure that the overseas recipient does not breach the APPs.

Feedback Mechanism

Likelihood: Medium

Justification: As the user base grows, the need for an effective feedback mechanism becomes more pronounced. Organizations that don't actively seek and act on feedback may miss out on essential insights and potential improvements.

- An effective feedback mechanism can assist organizations in identifying and resolving issues, improving practices, and increasing employee engagement and satisfaction.

Demo

Sprint 3 Development Build Demo Video (dev-1.1)

Google Drive: https://drive.google.com/file/d/1awizpCoL98sFeZyIUixLzmtOndiYElIS/view?resourcekey

Direct Download: roster-app-sprint3-demo.mp4

Sprint 4 Final Product Demo Video (dev-1.2)

Google Drive: https://drive.google.com/file/d/1zOxsx5qXhyeZozqbFkixvkeUFC-1Kr7u/view?usp=drive_link

Direct Download: final-product-demo.mp4

Presentation Slides Download: RosteringApp-comp90082.pptx

Handover

To download the handover book: 

Product Overview

We aim to build up a user-friendly rostering app to meet the highly increasing demand of casualised employees. It helps the casual employee to share and manage their work time availability and schedule with employers. In addition to this, employees can register an account and the personal schedule and basic settings (such as your profile) in the account will be preserved in the database for real-time updates on different devices.

Description of user

This rostering app is developed for casual employees to better manage their working schedule. It is an employee-centric rostering tool for mobile devices, which provides an portal for employees to manage their work information, see upcoming shifts, share their availabilities with employers, and more.

Directory Structure

```
|__ docs/
    |__ Project documentation
|__ src/
    |__ Client/
        |__ roster-app/ (Mobile app project folder)
    |__ Server/
        |__ firebase-emulator/ (Local backend emulator for development use)
README.md
```

Environment

The mobile app "roster-app" is an project that uses React Native with TypeScript and Firebase backend.

Common dependencies:

- Node.js version 16.13.0 or higher, npm

Backend emulator dependencies (You don't have to use the emulator, the project has been deployed as an actual Firebase project):

- Java JDK version 11 or higher
- Firebase CLI version 8.14.0 or higher

Remote backend Firebase console can be accessed here: https://console.firebase.google.com/u/1/project/roster-app-2023/overview

You can grant others access to the Firebase console at the Project Settings Users and Permissions page: https://console.firebase.google.com/u/1/project/roster-app-2023/settings/iam

For more information, see Development Environment

Set up

Setting up the local repository for the first time

1. Clone the repository.

2. Open src/Client/roster-app in VSCode. This is the project folder for our mobile app.

3. At src/Client/roster-app, install dependencies using the command: npm install.

The frontend project automatically connects to the remote Firebase backend while running live preview. It is also possible to set up a local backend emulator if you don't want to worry about exceeding quotas.

For more details, refer to the README.md file in the repository.

Live Preview on Mobile Device

1. Install Expo Go app on your phone.

2. At src/Client/roster-app, start the live preview server by running the code below:

```
npx expo start
```

3. A QR code will appear on the Terminal. Scan the QR code with your Expo Go app.

Note: make sure your phone is connected to the same network as your machine.

For details, see https://docs.expo.dev/get-started/expo-go/

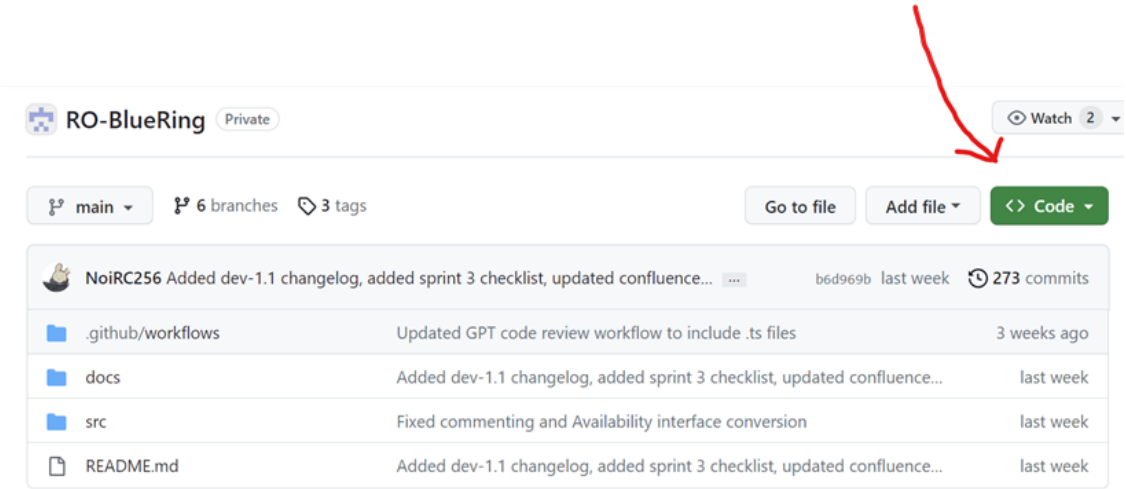
Remote Backend

Remote backend Firebase console can be accessed here: https://console.firebase.google.com/u/1/project/roster-app-2023/overview

Members with the Owner role can grant others access to the Firebase console at the Project Settings Users and Permissions page: https://console.firebase.google.com/u/1/project/roster-app-2023/settings/iam

Step by step instructions

1. On https://github.com/COMP90082-2023-SM2/RO-BlueRing, click <> Code.



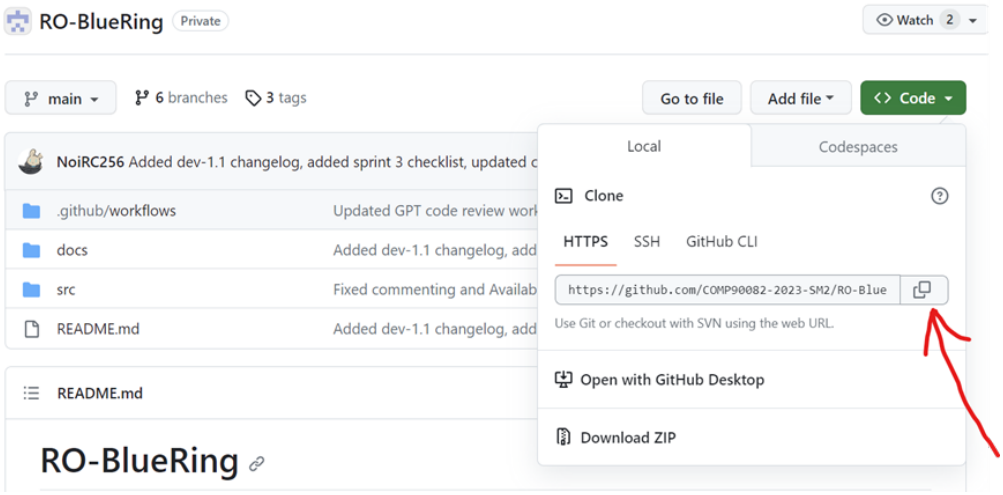
2. We recommend using "Open with Github Desktop" to manage this repository using a graphical interface.

If you are using Github Desktop, skip to step 5.

If you don't wish to use Github Desktop, please refer to the following steps 2 ~ 4.

Copy the URL for the repository.

- To clone the repository using HTTPS, under "HTTPS", click copy.
- To clone the repository using an SSH key, including a certificate issued by your organization's SSH certificate authority, click SSH, then click copy.
- To clone a repository using GitHub CLI, click GitHub CLI, then click copy.



3. Change the current working directory to the location where you want the cloned directory.

4. In your terminal, type git clone, then paste the URL you copied earlier:

```
git clone https://github.com/COMP90082-2023-SM2/RO-BlueRing.git
```

5. For the next steps, you need to change the working directory to the frontend app's project directory, \src\Client\roster-app\

For example: my “roster-app” folder is in the location of “roster-app” D:\UNIOFMEL\S4\project\ProjectAPP\RO-BlueRing\RO-BlueRing\src\Client\roster-app. I just need to change the directory using cd:

```
cd D:\UNIOFMEL\S4\project\ProjectAPP\RO-BlueRing\RO-BlueRing\src\Client\roster-app
```

Note: to make things easier, you can:

- A. Open the project folder \src\Client\roster-app\ in Visual Studio Code and use its terminal for the following steps.
- B. Alternatively, navigate to the directory in your file explorer and directly launch your terminal there.

6. At \src\Client\roster-app\, Install dependencies by running the following code.

Do this every time you switch branch or pull in case there are any changes to project dependencies.

```
npm install
```

7. To view the app on your device, you need to install Expo Go app on your phone. You can install from the website below:

https://expo.dev/client

8. At \src\Client\roster-app\, run the live preview using the following command:

```
npx expo start
```

9. A QR code will appear in the terminal. Scan it using your Expo Go app. You can now use the app!

Note: make sure your phone is connected to the same network as your machine.

For details, see https://docs.expo.dev/get-started/expo-go/

Development Documentation

The development documentation outlines some conventions we use in the project. For details, please refer to Development Documentation

Build & Deploy

For building and deploying, we recommend looking into Expo EAS

User story checklist

Epic Categories	Story ID	Feature	Story Point	Participant	Priority	Status
Account Management	1	Register(Employee)	2	Xuechun Chen	HIGH	Finished
	2	Log in (employee)	2	Xuechun Chen	HIGH	Finished
	3	Normal Password Reset	2	Xuechun Chen	HIGH	Finished
	4	Forgotten Password Reset	2	Xuechun Chen	HIGH	Finished
	5	Edit Account	2	Xuechun Chen	HIGH	Finished
	6	Delete my Account	2	Xuechun Chen	MEDIUM	Finished
Availability Management	9	Set ad-hoc availabilities	8	Bo Peng	CORE	Finished
	10	Set standard availabilities	5	Bo Peng	CORE	Finished
	11	Share availabilities	8	Bo Peng	HIGH	Finished
	13	Edit personal information	3	Chenxi Zhang	CORE	Finished
Information Management	14	Manage identity document	3	Bo Peng	LOW	Droped
	15	Manage qualification documents	3	Xuechun Chen	LOW	Droped
	16	Manage employment contract	3	Xuechun Chen	HIGH	Droped
	17	Share information with selected employers	5	Bo Peng	MEDIUM	Droped
Work Shift Management	18	Work shift page	10	Wenxuan Chen/Sichen Lu	CORE	Finished
	19	Sync work shifts with personal calendar	10	Sichen Lu	CORE	Finished
	20	Add, edit work shift information	7	Wenxuan Chen	HIGH	Finished
Workplace Management	24	Workplace management	6	Chenxi Zhang	HIGH	Finished
	25	View workplace pay details	8	Chenxi Zhang	LOW	Finished
Pay Management	26	View total earnings	7	Chenxi Zhang	LOW	Finished
	27	Add pay slips	7	Sichen Lu	MEDIUM	Finished
App	37	Home Screen	4	Sichen Lu	MEDIUM	Finished

Operating the Product

In terms of operating the product, please view the demo on the confluence link below:

Demo

Final Presentation Slides


RO-BlueRing_Pres...ation_Slides.zip

Additional Resources

Course Resources

Project-Based Course Notes

Tech Stack

Frontend - TypeScript, React Native

Backend - Firebase

Collaboration Files

Project Google Drive

Developer Handbook Google Doc

Figma UI Design Project

VSCode Extensions

React Native Tools

ES7+ React/Redux/React-Native snippets

Prettier

Designing for Colour Blindness

https://www.getfeedback.com/resources/ux/how-to-design-for-color-blindness/

https://davidmathlogic.com/colorblind/#%23D81B60-%231E88E5-%23FFC107-%23004D40

Third-party Documentation

Third-Party Documentation

Common

Development Framework - Expo

Database - Firestore

Authentication - Firebase Authentication

Local Backend Emulator - Firebase Local Emulator Suite

Frontend

UI Library - UI Kitten

Page Navigation - Expo Router

Local Storage for Small Data - Async Storage

Date and Time Picker - Expo DateTimePicker

Tutorials

React Native Expo File Based Router with Firebase Authentication