COMP3900 Computer Science Project

P1: ServiceUNSW: Centralised University Services App

October 08, 2025
3900-H09B-APPLE

Allan Zhang	Backend Developer	z5311565@ad.unsw.edu.au
Cassandra Tran	Software Testing Engineer	z5257701@ad.unsw.edu.au
David Nie	Scrum Master	z5303303@ad.unsw.edu.au
Ebenzer Ajaero	Project Owner	z5362105@ad.unsw.edu.au
Ferdinand Woo	Backend Developer	z5350438@ad.unsw.edu.au
Terry Zhong	Full-Stack Developer	z5482100@ad.unsw.edu.au

Contents

Project Background

01

Problem Definition & Impact

Domain

To access services, UNSW students must access manifold services (incl. Moodle, myUNSW, Uni-Verse, Nucleus and Arc). This fragmentation reduces the ability of students to efficiently access services. Concerning student services, it increases unnecessary friction and institutional inefficiencies. Additionally, it reduces uptake of essential university amenities, especially among less digitally astute students, thus hampering student well-being.

Scale and severity

Inefficiencies and User Satisfaction

UNSW provides services to 70,000+ students and staff. Compounded frictions and inefficiencies become significant at such a scale. Several studies comment that app/context switching has negative effects on human productivity, citing time lost in adjusting to new app contexts, purpose and interfaces. Some studies report up to 4-20 hours per week lost (Murty et al., 2022; Quickbase, 2023), while others observe up to 32 days per year lost (Faw, 2018). Additionally, it is noted that users show a strong preference for consolidating all their preferences into a single platform (Faw, 2018). Students at UNSW likely face commensurable inefficiencies, albeit on a smaller scale. Nevertheless, this is still a severe problem as it affects everyone who requires UNSW services, including marginalised and vulnerable groups that have a greater dependence on UNSW services. Consolidating UNSW's fragmented platforms to a single platform would help alleviate this problem.

Accessibility and awareness

Accessibility and inclusiveness of digital services have recently been highlighted as one of the top three issues of higher education technology (Council of Australasian University Directors of Information Technology [CAUDIT], 2025). Fragmentation of critical services like those at UNSW reduces awareness and accessibility of essential resources and services, especially for less digitally literate students (Gkrimpizi et al., 2023). This issue has become increasingly more significant over recent years, demonstrating its growing impact (CAUDIT, 2025). Combining UNSW services into an accessible mobile app would assist with this problem.

Lower student engagement

To access services, UNSW students must access manifold services (incl. Moodle, myUNSW, Uni-Verse, Nucleus and Arc). This fragmentation reduces the ability of students to efficiently access services. Concerning student services, it increases unnecessary friction and institutional inefficiencies. Additionally, it reduces uptake of essential university amenities, especially among less digitally astute students, thus hampering student well-being.

The complexity involved in using disconnected platforms lowers student engagement. Alternatively, mobile-first access increases student satisfaction. According to the 2024 Educause Student Technology report, 72% of students surveyed preferred mobile access to university services with mobile notifications (Comunetti, 2025; Handshake UK, 2025).

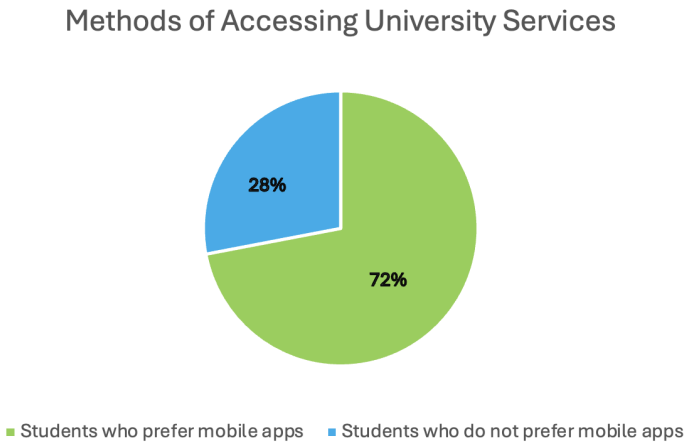


Figure 1: Students' preferred methods of accessing university services

The report shows how critical mobile-first platforms have become for student engagement. Student engagement is a less severe problem than efficiency and accessibility. However, it would still be beneficial to address it.

UNSW-specific issues

The UNSW Uni-verse app integrates core utilities (e.g. timetable, services), but student criticism highlights limited depth of integration, dated user interface (UI) and stability issues/bugs. UNSW critical services are spread across different portals and platforms. This leads to missed/inconsistent information, inefficient workflows and lower engagement.

Potential solutions

Recently, governments have emphasised mobile self-service as a potential solution to similar problems. For example, the Service NSW application serves 3 million+ monthly users, providing them regular access to amenities, digital ID credentials and more. Such solutions can also be implemented concerning university student amenities.

Proposed solution

We propose ServiceUNSW, a mobile app that consolidates core pre-existing student platforms, and emphasises intuitive and accessible workflows (e.g. agentic AI-powered workflows). ServiceUNSW will use a modular architecture to integrate the following:

a. Digital ID
b. Timetables/Calendars
c. Student Life Events and Academic Calendar
d. Service Directory
e. Campus Map
f. Notifications
g. UNSW Scout AI Assistant.

ServiceUNSW will address the pain points in the Uni-Verse app. It will make use of modularity to allow for future flexibility and incremental development. Additionally, this will allow the app to adapt to changing university demands. ServiceUNSW, through incorporating UNSW Scout AI will serve as an example of how AI technology) can make positive impacts in the Ed-Tech space.

Review of Existing Systems

UNSW Uni-Verse (Official App)

Problem Addressed
The UNSW Uni-Verse (Official App) centralises access to the UNSW timetable, maps, events and support links, specifically for UNSW students.

Strengths to Emulate
The Uni-Verse app implements the UNSW Single-Sign-On (SSO) standard. This reduces credential fatigue and normalises authentication. It maintains a catalogue of university services. Furthermore, it has a high degree of trust as it emphasises its official status through consistent UNSW branding.

Weaknesses
The app has been criticised for having shallow integration. Many links open in an external web browser rather than within the app. The map is clunky and not modernised. The UI is dated and needs modernisation. Additionally, it lacks in-app notification history and Digital ID features. It has only 3/5 stars on public reviews on the Apple App Store.

Design Implications
ServiceUNSW will prioritise native workflows for frequent actions, rather than just providing external links to services. It will provide an in-app notification history centre, as well as Digital ID features. ServiceUNSW will emphasise a modern consistent UI.

ServiceNSW

Problem Addressed
The original Service NSW app provides unified access to governmental services and veritable digital credentials. Initially, NSW government services were fragmented similarly to UNSW's current offerings. Citizens had to navigate across disconnected government web portals and backend systems. The Service NSW app instead provides an accessible mobile that centralises these systems. The benefit of the Service NSW app has been recognised by its extremely high user trust, with 3 million+ users per month.

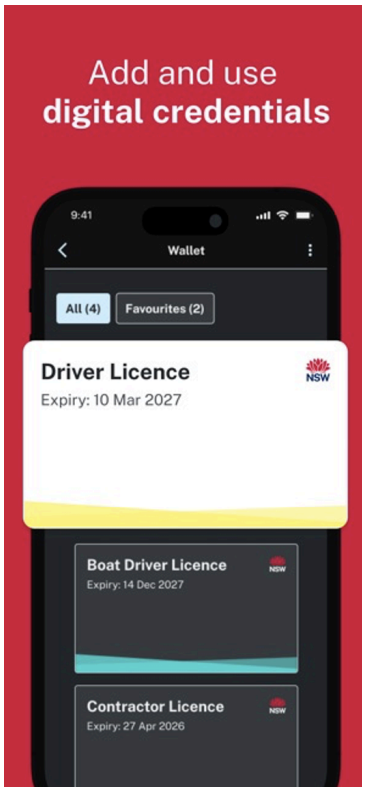


Figure 2: ServiceNSW intuitive digital credential system (ServiceNSW, 2025).

Strengths to Emulate

The Service NSW app has a well-implemented digital ID service with the capability to connect to an on-device wallet (e.g. Apple Wallet) (Figure. 2). Additionally, the mobile app features an accessible and inclusive design, which has boosted adoption rates. The Service NSW app implements a clear and modular architecture with its multi-agency services (Service NSW, 2025)

Limitations

Service NSW has limited applicability for our app. This is because they reside in different domains. For example, ServiceNSW is concerned with citizens and legal verification, as opposed to student identification within the university. Therefore, it implements stringent digital ID verification methods that are excessive for the use case of the university. UNSW student ID, instead, requires status contexts such as enrolment/term status and capability for offline verification (e.g. for exams).

Design Implications

ServiceUNSW will use a similar digital ID feature to ServiceNSW, albeit with a different schema for Student ID (with similar fields on the UNSW physical ID: zID, enrolment, program, etc). ServiceUNSW, although it will not implement as stringent a verification, will allow the digital ID to serve as verification for UNSW exams. ServiceUNSW, like ServiceNSW will maintain accessibility standards, abiding by WCAG 2.2 AA standards.

UNSW “Scout” AI Agent

Problem Addressed

UNSW Scout AI unifies fragmented workflows and routine FAQs into a single conversational entry point. It allows users to intuitively access services and information using natural language enquiries, building on the principle of “recognition not recall”.

Strengths to use

UNSW Scout AI already aligns with UNSW’s long-term Ed-tech roadmap and plan to modernise technology. As a foundation, it has the capacity to be used as an LLM engine. With Scout + an API, ServiceUNSW can achieve the same functionality without having to fine-tune an LLM from scratch (Microsoft, 2025).

Limitations

UNSW Scout AI has unclear API maturity. As a pilot program, it has several future unknowns, including rate limits, privacy guardrails and how it would incorporate user feedback.

Design Implications

ServiceUNSW should rely primarily on Scout AI for answering FAQs. However, it should create fallbacks, e.g. redirect screens. The ServiceUNSW app should provide feedback to the Scout service to iteratively improve the AI.

myUni (University of Newcastle)

Problem Addressed

The UoN app, myUni, centralised all pre-existing systems and information students need for studies and university life (University of Newcastle, 2018). It became a single point of accessing timetables (with directions), maps, email and local services.

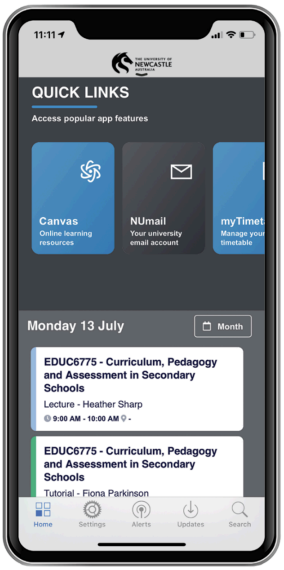


Figure 3: University of Newcastle’s myUni app (University of Newcastle, 2018).

Strengths to Emulate

UoN's myApp has strong breadth of useful services consolidated in the app. In addition to timetables, maps, emails and local services, it also boasts integration with Microsoft OneDrive, Transport NSW and retail services. Another strength lies in myUni's customisability, allowing students to pick and choose which systems they would like to see in their Quick Links and their homepages. These strengths are particularly relevant to the problem domain, since it allows students to access all the services and information they want from a single app, instead of having to actively search and switch platforms themselves, as students currently do at UNSW. Since the myUni app successfully addresses the exact same problem that Service UNSW aims to solve, Service UNSW will also aim to incorporate many of these features in its own design.

Limitations

The University of Newcastle's myUni app does have a few weaknesses. Reviews from students report complaints about having to repeatedly login, issues with crashes, as well as dissatisfaction with its poor interface (University of Newcastle, 2018). Similarly to Uni-Verse from UNSW, these limitations are significant as users will not be satisfied with an app which does not work, is inconvenient, and also unappealing. As such, Service UNSW will be designed to be a robust and reliable app with no login issues and with a visually appealing interface.

Design Implications

ServiceUNSW will consider customisability, hiding and rearranging of different UI tiles, favouring simple personalisation. ServiceUNSW will also incorporate silent refreshes and SSO token renewals, while maintaining a reasonable performance budget.

Sydney Uni (University of Sydney)

Problem Addressed

The University of Sydney also has its own mobile application, with a similar function to previously mentioned university applications: centralising key university systems and features for students (University of Sydney, 2025).

Strengths to Emulate

Likewise, its strength lies in the various features it offers, including a class timetable with maps that can direct students to their classrooms, the ability to book appointments and study spaces, and so on. Service UNSW will also aim to include similar features.

Limitations

The Sydney Uni app also has its own limitations. A recurring weakness with existing mobile apps aiming to centralise key university services and features seems to be frequent crashes and poor performance, drawing many complaints from users despite the convenient features that are included (University of Sydney, 2025).

Design Implications

Therefore, reliability and performance will be one of the key priorities for the development of the Service UNSW app.

User Stories & Sprints

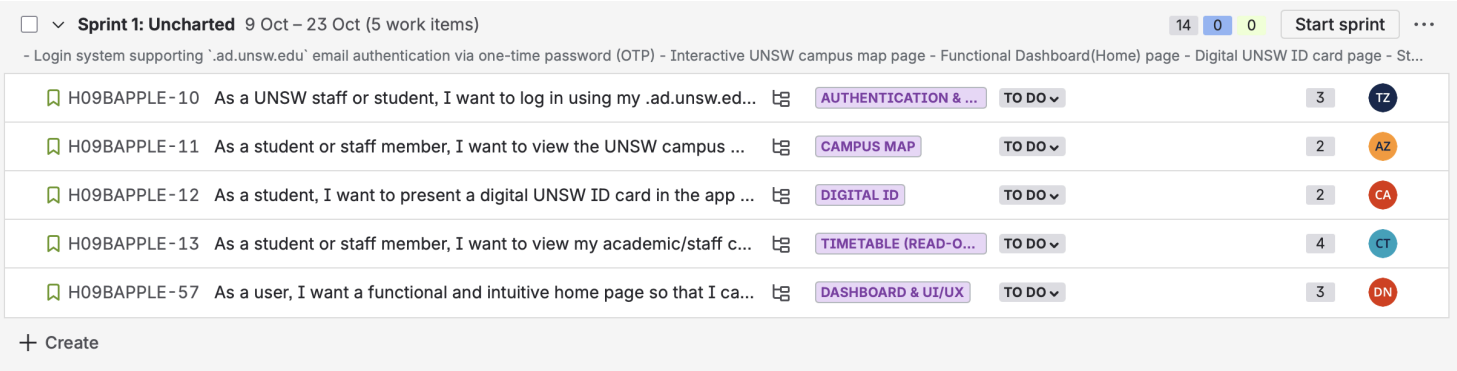
02

Scope Statement

The project will deliver a mobile-first application featuring a login system with email authentication, an embedded interactive UNSW campus map, a digital UNSW ID card, service directory, event directory, and student calendar. Modules will use mock data in the absence of API access. This enables demonstration of end-to-end functionality while maintaining modular interfaces for future system integration. Production system integrations, advanced administrative tooling that integrate with UNSW real systems, and advanced security features are beyond the scope, given the current limitations on access and permissions.

The scope statement was shared with the client on 8 October 2025, with feedback to be discussed at the next meeting on 10 October 2025.

Sprint 1 User Stories



Sprint 1: Uncharted 9 Oct – 23 Oct (5 work items) 14 0 0 Start sprint

- Login system supporting `.ad.unsw.edu` email authentication via one-time password (OTP) - Interactive UNSW campus map page - Functional Dashboard(Home) page - Digital UNSW ID card page - St...

H09BAPPLE-10	As a UNSW staff or student, I want to log in using my .ad.unsw.ed...	AUTHENTICATION & ...	TO DO	3	TZ
H09BAPPLE-11	As a student or staff member, I want to view the UNSW campus ...	CAMPUS MAP	TO DO	2	AZ
H09BAPPLE-12	As a student, I want to present a digital UNSW ID card in the app ...	DIGITAL ID	TO DO	2	CA
H09BAPPLE-13	As a student or staff member, I want to view my academic/staff c...	TIMETABLE (READ-O...	TO DO	4	CT
H09BAPPLE-57	As a user, I want a functional and intuitive home page so that I ca...	DASHBOARD & UI/UX	TO DO	3	DN

+ Create

As a UNSW staff or student, I want to log in using my .ad.unsw.edu email and a one-time password so that I can securely access the application without needing a permanent password.

To Do

Description

Acceptance Criteria

- **Given** I am on the login page, **when** I enter a valid `.ad.unsw.edu` email, **then** the system sends a one-time password (OTP) to that email.
- **Given** I have received an OTP, **when** I enter it correctly within the valid time period, **then** I am successfully logged in and redirected to the home page.
- **Given** I enter an invalid or expired OTP, **when** I attempt to log in, **then** the system displays an error message stating "Invalid or expired OTP."
- **Given** I enter an email that is not under the `.ad.unsw.edu` domain, **when** I attempt to log in, **then** the system displays a message stating "Please use a valid UNSW email address."

Child work items

0% Done

Work	Prio...	Ass...	Status
H09BAPPLE-85 FE: UI Design & backend access points	= M.	U	TO DO
H09BAPPLE-86 FE: API endpoints	= M.	U	TO DO
H09BAPPLE-87 DB: Compatible Schema	= M.	U	TO DO
H09BAPPLE-88 FE: Backend Integration	= M.	U	TO DO
H09BAPPLE-89 Testing & Product Owner Review	= M.	U	TO DO

As a student or staff member, I want to view the UNSW campus map within the app so that I can quickly find locations on campus.

To Do

Description

Acceptance Criteria

Embed Loads

- **Given** I am **online** and on the **Map page**,
  When I open the page,
  Then an **embedded UNSW campus map** should render inside the page.

Offline State

- **Given** I am **offline**,
  When I open the **Map page**,
  Then I should see a **friendly message** explaining the map isn't available, with a **Retry button**.

External Links

- **Given** the **embedded map** includes **outbound links**,
  When I tap one of those links,
  Then it should **open in my device's browser**.

Performance & Privacy

- **Given** the embedded map loads **third-party resources**,
  When it renders,
  Then no **personally identifiable information** should be stored, and **platform privacy settings** should be respected.

As a student, I want to present a digital UNSW ID card in the app so that I can show basic identity for campus use (events, building entry checks) without high-security verification in MVP.

To Do

Description

Acceptance Criteria

Card display
Given I open My digital ID while logged in, **when** the card loads, **then** I should see my name, ZID, faculty/school, and either my profile photo or a placeholder image.

QR (static MVP)
Given I am on the ID card view, **when** I tap "Show QR", **then** a static QR code displays containing minimal information (name and ZID) along with a disclaimer that it's for campus use only.

(Stretch Goal) Apple Wallet entry point
Given, I'm eligible for Apple Wallet, **when** I tap "Add to Apple Wallet", **then** the app should add my digital ID to my Apple Wallet and tell me if it was added successfully or it failed.

Disclaimers

Given I am viewing my ID card, **then** clear text should be displayed stating "For UNSW campus use; not a government ID" or equivalent wording.

No Strong Security Claims (MVP)

Given I am viewing or using my digital ID, **then** the interface should not include any statements implying it is government-grade, tamper-proof, or otherwise official identification.

As a student or staff member, I want to view my academic/staff calendar in the app by supplying an .ics link so that I can see my term schedule in an accessible weekly view.

To Do

Description

Acceptance Criteria

Add calendar with valid URL
Given I am on the calendar page with a valid .ics URL, when I paste the URL and press Add, then the events should load into an agenda or week view.

Invalid URL
Given I am on the calendar page with an invalid or unreachable URL, when I press “Add”, then I should see a helpful error message and have the option to retry or re-enter the URL.

Refresh
Given I have already added a calendar feed, when I pull to refresh, then the app should fetch new events and update the calendar view.

As a user, I want a functional and intuitive home page so that I can easily access key features, view important information at a glance, and navigate through the platform efficiently.

To Do

Description

Layout and Structure

- **Given** I open the dashboard,
  When it loads on any device,
  Then it should display a **clean, responsive layout** with **consistent navigation and widgets**, following **UNSW’s colour palette**.

Usability and Intuitiveness

- **Given** I view dashboard content,
  When metrics or charts are displayed,
  Then they should be **clear, easy to interpret**, and **visually distinct**.

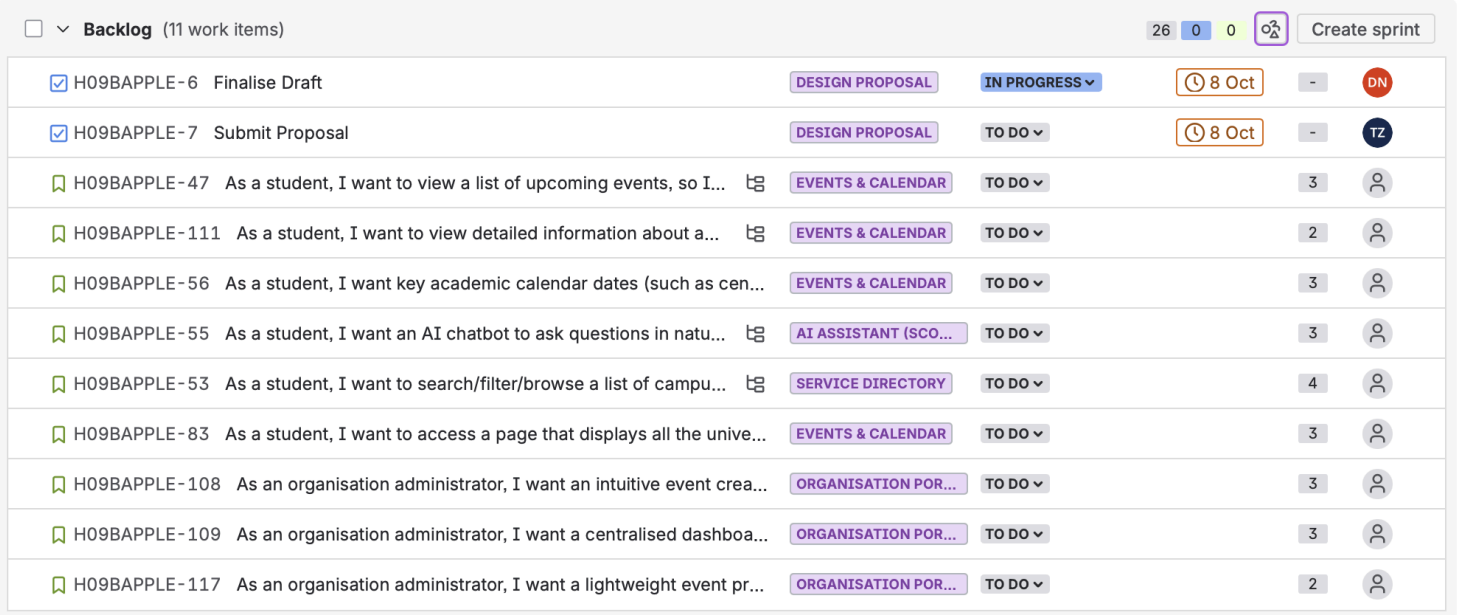
Performance and Scalability

- **Given** I load or refresh the dashboard,
  When data updates occur,
  Then it should **load quickly**, **update asynchronously**, and **support future expansion**.

Consistency and Accessibility

- **Given** I interact with the dashboard,
  Then all components should be **visually consistent**, **accessible (WCAG AA+)**, and **properly aligned**.

Product Backlog



Backlog (11 work items) 26 0 0 Create sprint

H09BAPPLE-6	Finalise Draft	DESIGN PROPOSAL	IN PROGRESS	8 Oct	-
H09BAPPLE-7	Submit Proposal	DESIGN PROPOSAL	TO DO	8 Oct	-
H09BAPPLE-47	As a student, I want to view a list of upcoming events, so I...	EVENTS & CALENDAR	TO DO		3
H09BAPPLE-111	As a student, I want to view detailed information about a...	EVENTS & CALENDAR	TO DO		2
H09BAPPLE-56	As a student, I want key academic calendar dates (such as cen...	EVENTS & CALENDAR	TO DO		3
H09BAPPLE-55	As a student, I want an AI chatbot to ask questions in natu...	AI ASSISTANT (SCO...	TO DO		3
H09BAPPLE-53	As a student, I want to search/filter/browse a list of campu...	SERVICE DIRECTORY	TO DO		4
H09BAPPLE-83	As a student, I want to access a page that displays all the unive...	EVENTS & CALENDAR	TO DO		3
H09BAPPLE-108	As an organisation administrator, I want an intuitive event crea...	ORGANISATION POR...	TO DO		3
H09BAPPLE-109	As an organisation administrator, I want a centralised dashboa...	ORGANISATION POR...	TO DO		3
H09BAPPLE-117	As an organisation administrator, I want a lightweight event pr...	ORGANISATION POR...	TO DO		2

As a student, I want to view a list of upcoming events, so I can stay informed and bookmark those I plan on attending

To Do

Description

Acceptance Criteria:

1. View Event page list

Given I am on the events list page, **when** I scroll through it, **then** I should see a list of upcoming university events with name, date, time and location

2. Events ordered in Chronologically

Given multiple upcoming events exist, **when** i view the list of events, **then** the events should be ordered by date of event

3. Bookmarking event button

Given I am on the event page, **when** I click the bookmark icon, **then** the bookmark icon should visibly change to indicate that the event has been bookmarked.

As a student, I want to view detailed information about an event, so that I can conveniently learn about and join events without switching between multiple platforms

To Do

Description

Acceptance Criteria:

1. **Event Information Display**
   - Given I am on the Event Details page,
     When I open an event,
     Then I should see the event's **thumbnail image**, **title**, **date/time**, **location**, and **description** clearly displayed.
2. **Sign-Up Functionality**
   - Given I am viewing an event that allows sign-ups,
     When I click the **Sign Up** button,
     Then my registration should be confirmed instantly with a **success message** (e.g., "You've successfully signed up for this event") if no other information is required by the organiser.
3. **Unified and Intuitive Design**
   - Given I am using the platform,
     When I access event details,
     Then the interface should be consistent with the rest of the platform (same theme, navigation, and button styles).
4. **Error and Feedback Handling**
   - Given the event is full or sign-up fails,
     When I attempt to sign up,
     Then the sign-up button should be disabled, or I should receive a clear and concise **error message** explaining why (e.g., "This event has reached capacity").
5. **Ease of Navigation**
   - Given I'm browsing events,
     When I return to the Event Details page,
     Then I should be able to **go back** to the event list easily using a visible and intuitive control (e.g., a "Back" button or breadcrumb).

As a student, I want key academic calendar dates (such as census dates) to be clearly highlighted within the app, so that I can conveniently stay aware of important deadlines and plan my schedule effectively.

To Do

Description

Acceptance Criteria:

1. View academic events list page

Given I am logged onto the app, **when** I open academic calendar page, **then** i should see a list of upcoming events including census dates, flex week, and exam periods

2. Events are ordered chronologically by date

Given the list of academic calendar is displayed, **when** I scroll through the list, **then** the events should be shown in chronological order

3. View Empty state

Given no events are currently available in the system, **when** I open the Academic events page, **then** I should see the message "No upcoming academic events available"

As a student, I want to view detailed information about an event, so that I can conveniently learn about and join events without switching between multiple platforms

To Do

Description

Acceptance Criteria:

1. **Event Information Display**
   - Given I am on the Event Details page,
     When I open an event,
     Then I should see the event's **thumbnail image**, **title**, **date/time**, **location**, and **description** clearly displayed.
2. **Sign-Up Functionality**
   - Given I am viewing an event that allows sign-ups,
     When I click the **Sign Up** button,
     Then my registration should be confirmed instantly with a **success message** (e.g., "You've successfully signed up for this event") if no other information is required by the organiser.
3. **Unified and Intuitive Design**
   - Given I am using the platform,
     When I access event details,
     Then the interface should be consistent with the rest of the platform (same theme, navigation, and button styles).
4. **Error and Feedback Handling**
   - Given the event is full or sign-up fails,
     When I attempt to sign up,
     Then the sign-up button should be disabled, or I should receive a clear and concise **error message** explaining why (e.g., "This event has reached capacity").
5. **Ease of Navigation**
   - Given I'm browsing events,
     When I return to the Event Details page,
     Then I should be able to **go back** to the event list easily using a visible and intuitive control (e.g., a "Back" button or breadcrumb).

As a student, I want key academic calendar dates (such as census dates) to be clearly highlighted within the app, so that I can conveniently stay aware of important deadlines and plan my schedule effectively.

To Do

Description

Acceptance Criteria:

1. View academic events list page

Given I am logged onto the app, **when** I open academic calendar page, **then** i should see a list of upcoming events including census dates, flex week, and exam periods

2. Events are ordered chronologically by date

Given the list of academic calendar is displayed, **when** I scroll through the list, **then** the events should be shown in chronological order

3. View Empty state

Given no events are currently available in the system, **when** I open the Academic events page, **then** I should see the message "No upcoming academic events available"

As a student, I want an AI chatbot to ask questions in natural language, so that I can get instant, accurate answers about UNSW services without extensively searching through UNSW websites

To Do

Description

Acceptance criteria

1. Natural language understanding

Given I am on the ai chatbot page, **when** I type a question in plain english(i.e. “What does census date mean?”), **then** the chatbot provides a relevant, accurate answer to the user’s query.

2. Chatbot interface availability

Given I am on the chatbot page in the app, **when** the page loads, **then** I should see a text input field where I can type and a send button to interact with the chatbot

3. Error handling

Given I send a message, **when** the chatbot is unavailable at the moment or fails to return a response, **then** i should see a message saying “Sorry, the chatbot is currently unavailable. Please try again later”

4. session persistence

Given I have an ongoing chat session, **when** I refresh or reopen the chatbot, **the** previous conversation should still remain visible(or session rules should be clearly defined to the user (i.e “the conversation will close if you leave the chatbot”)

As a student, I want to search/filter/browse a list of campus services(i.e. IT help, library, counselling) so that I can find and access the service I need

To Do

Description

Acceptance Criteria:

View campus service list

1. **Given** when I am logged onto the app, **when** I open the service directory, **then** I should see a list of available campus services

Keyword search

2. **Given** I am on the service directory page, **when** i type a keyword into the search bar (i.e. counselling), **then** i should see services whose name or description contains the word

Search by category

3. **Given** I am on the service directory page, **when** I apply a filter by category (i.e. IT services), **then** I should only see services that belong to that category

View Individual service information

4. **Given** I am viewing the list of services, **when** i tap on a specific service (i.e. counselling), **then** I should see its details(i.e. contact info, opening hours, description, and link)

As a student, I want an AI chatbot to ask questions in natural language, so that I can get instant, accurate answers about UNSW services without extensively searching through UNSW websites

To Do

Description

Acceptance criteria

1. Natural language understanding

Given I am on the ai chatbot page, **when** I type a question in plain english(i.e. “What does census date mean?”), **then** the chatbot provides a relevant, accurate answer to the user’s query.

2. Chatbot interface availability

Given I am on the chatbot page in the app, **when** the page loads, **then** I should see a text input field where I can type and a send button to interact with the chatbot

3. Error handling

Given I send a message, **when** the chatbot is unavailable at the moment or fails to return a response, **then** i should see a message saying “Sorry, the chatbot is currently unavailable. Please try again later”

4. session persistence

Given I have an ongoing chat session, **when** I refresh or reopen the chatbot, **the** previous conversation should still remain visible(or session rules should be clearly defined to the user (i.e “the conversation will close if you leave the chatbot”)

As a student, I want to search/filter/browse a list of campus services(i.e. IT help, library, counselling) so that I can find and access the service I need

To Do

Description

Acceptance Criteria:

View campus service list

1. **Given** when I am logged onto the app, **when** I open the service directory, **then** I should see a list of available campus services

Keyword search

2. **Given** I am on the service directory page, **when** i type a keyword into the search bar (i.e. counselling), **then** i should see services whose name or description contains the word

Search by category

3. **Given** I am on the service directory page, **when** I apply a filter by category (i.e. IT services), **then** I should only see services that belong to that category

View Individual service information

4. **Given** I am viewing the list of services, **when** i tap on a specific service (i.e. counselling), **then** I should see its details(i.e. contact info, opening hours, description, and link)

As an organisation administrator, I want an intuitive event creation portal where I can easily set up new events, so that I can quickly publish events without technical difficulty or confusion.

To Do

Description

Acceptance Criteria:

1. View/Edit create event form

Given I am logged into the organisation portal, **when** I select “Create Event,” **then** I should see a clear, guided form to input event details.

2. Adding event details to form

Given I am filling the create event form, **when** I scroll through the form, **then** I should see that the system can support uploading of images, descriptions, and links in a user-friendly interface.

3. Ensure corrupted documents not provided

Given that I am in the process of filling in the details in the event create form, when upload text, picture, or link, then it should notify me if document i added is wrong/corrupted(i.e “link provided does not exists”, or “picture is in wrong file type”)

4. Form validation

Given that I am done filling in the form, **when** I click next/ready to submit button, **then** the event creation flow should include a confirmation of the details entered via a preview before allowing you to publish/submit to ensure the correct details have been entered in.

5. Successful publish of event

Given that I am done filling in the event details in the event create form publishing, **when** i click publish/submit button, **then** the event should appear in the event page list and/or organisation’s event list after a few seconds or after the page is refreshed.

As an organisation administrator, I want a centralised dashboard to manage my events, so that I can track attendance easily

To Do

Description

Acceptance Criteria:

- View all my Organisation’s events

Given I am logged onto the organisation portal as an admin, **when** I open the my events page, **then** my events page should list all current and past events

- Event lists view can be filtered

Given that I am on my Organisation’s events page as an admin, **when** I click the filter button, **then** I am able to filter the events shown to me by date(i.e upcoming, past)

- Dashboard for each event

Given I am on my Organisation’s events page as an admin, **when** I click on an event, **then** i should see a display of key statistics on a simple dashboard(i.e number of registrations)

- Edit/Delete event

Given I am looking at the details of one of my organisation’s event as an admin, **when** i click edit, **then** I should be able to edit or delete the event

As an organisation administrator, I want a lightweight event preview page accessible from the dashboard, so that I can quickly view how the event will appear to students.

To Do

Description

Acceptance Criteria:

1. **Accessing the Preview Page**
   - **Given** I am logged into the organisation dashboard,
   - **When** I click the “Preview” button on an event card,
   - **Then** I should be redirected to or shown a preview page that displays the event’s current details in the same format students will see.
2. **Accurate Event Rendering**
   - **Given** the event has saved details (e.g., title, date, description, image, location),
   - **When** I open the preview page,
   - **Then** the system should render all event elements exactly as they would appear to students (including responsive layout and formatting).
3. **Ease of Navigation**
   - **Given** I am viewing the preview page,
   - **When** I want to return to the dashboard or edit event details,
   - **Then** I should see a clear “Back” or “Edit Event” button that navigates me back without confusion.
4. **Lightweight Performance**
   - **Given** I open the preview page,
   - **When** it loads,
   - **Then** it should display event content within two seconds (assuming stable network conditions), ensuring a seamless experience.
5. **Consistency with Student View**
   - **Given** I open an event preview,
   - **When** comparing it to the student’s event page,
   - **Then** both should be visually consistent, with matching layout, typography, and media placement.

Sprint Timeline & Milestones

Disclaimer

All deliverables will include complete frontend–backend integration. In the event that access to UNSW API endpoints is unavailable, a mock implementation will be developed to allow future teams to seamlessly integrate the official endpoints.

Sprint 1

Start Date: 9th of Oct (Week 4 Thursday)
End Date: 23rd of Oct (Week 6 Thursday)
Duration: 2 weeks

Deliverables:

- Login system supporting @ad.unsw.edu email authentication via one-time password (OTP)
- Interactive UNSW campus map page
- Functional Dashboard(Home) page
- Digital UNSW ID card page
- Staff and student Timetable page

Sprint 2

Start Date: 23rd of Oct (Week 6 Thursday)
End Date: 13th of Nov (Week 9 Thursday)
Duration: 3 weeks excluding flexibility week

Deliverables:

- Service Directory Page
- Event Feature & Calendar
- Organisation Portal for event management & creation

Sprint 3

Start Date: 13th of Nov (Week 9 Thursday)
End Date: 27th of Nov (Week 11 Thursday)
Duration: 2 weeks

Deliverables:

- AI FAQ Chatbot
- Unit/Integration Testing
- CI/CD Pipeline

Demonstration

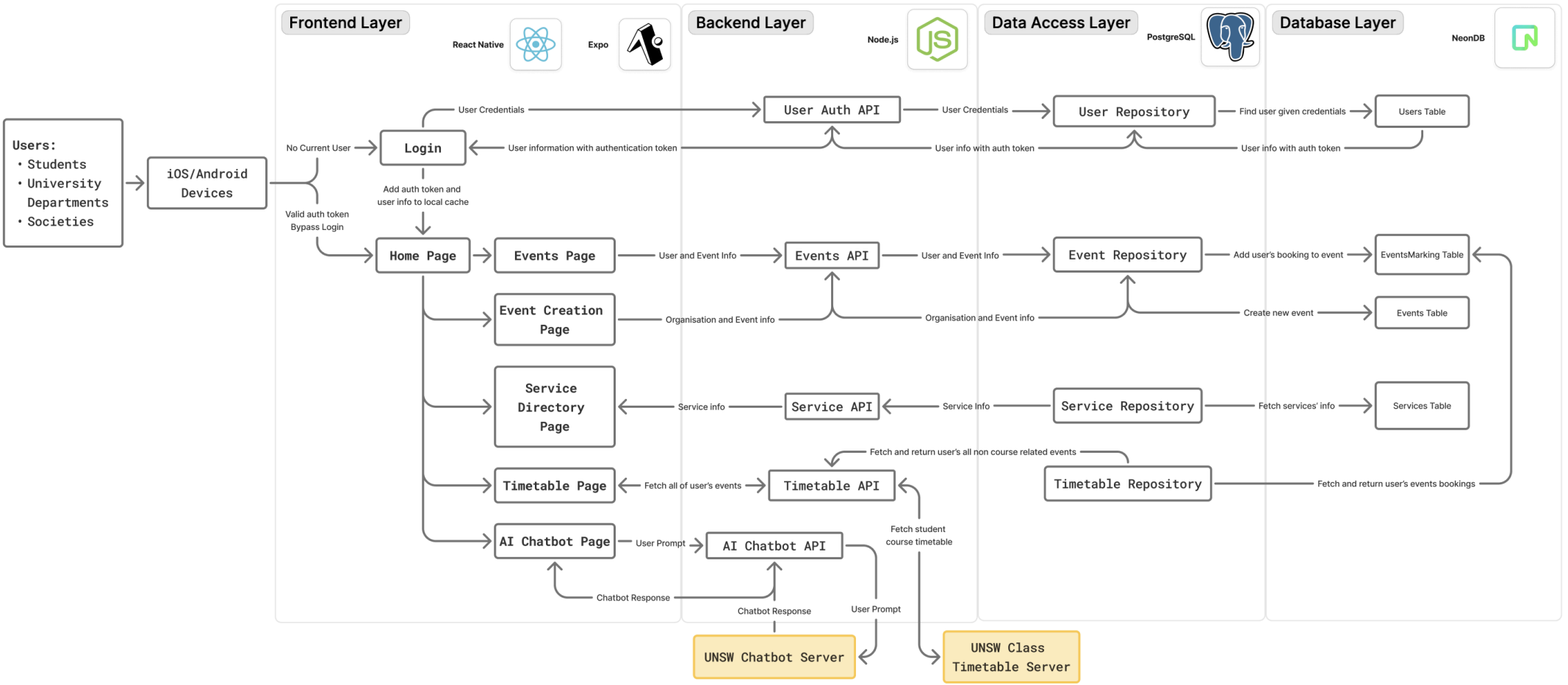
At the conclusion of Sprint 3, a full end-to-end demonstration of the mobile-first application will be conducted. The demo will showcase all implemented features, including authentication, navigation through the interactive campus map, access to the digital UNSW ID, and event and service directories. Mock data and APIs will be used where UNSW endpoints are unavailable, ensuring that each feature demonstrates complete frontend–backend interaction.

The goal of the demo is to validate usability, consistency, and integration across all modules, and to provide a clear foundation for future teams to extend functionality and connect with live UNSW systems.

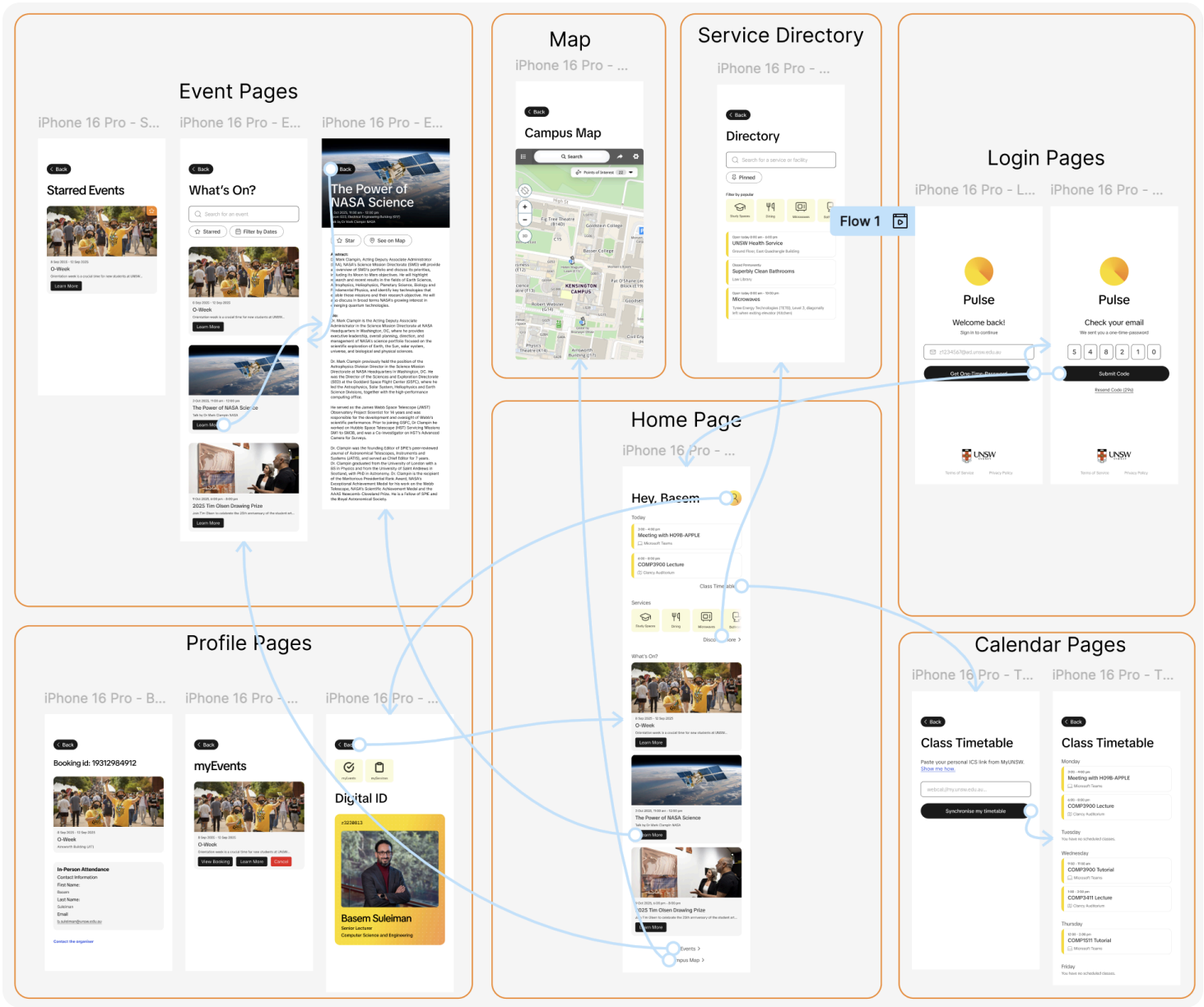
Technical Design

03

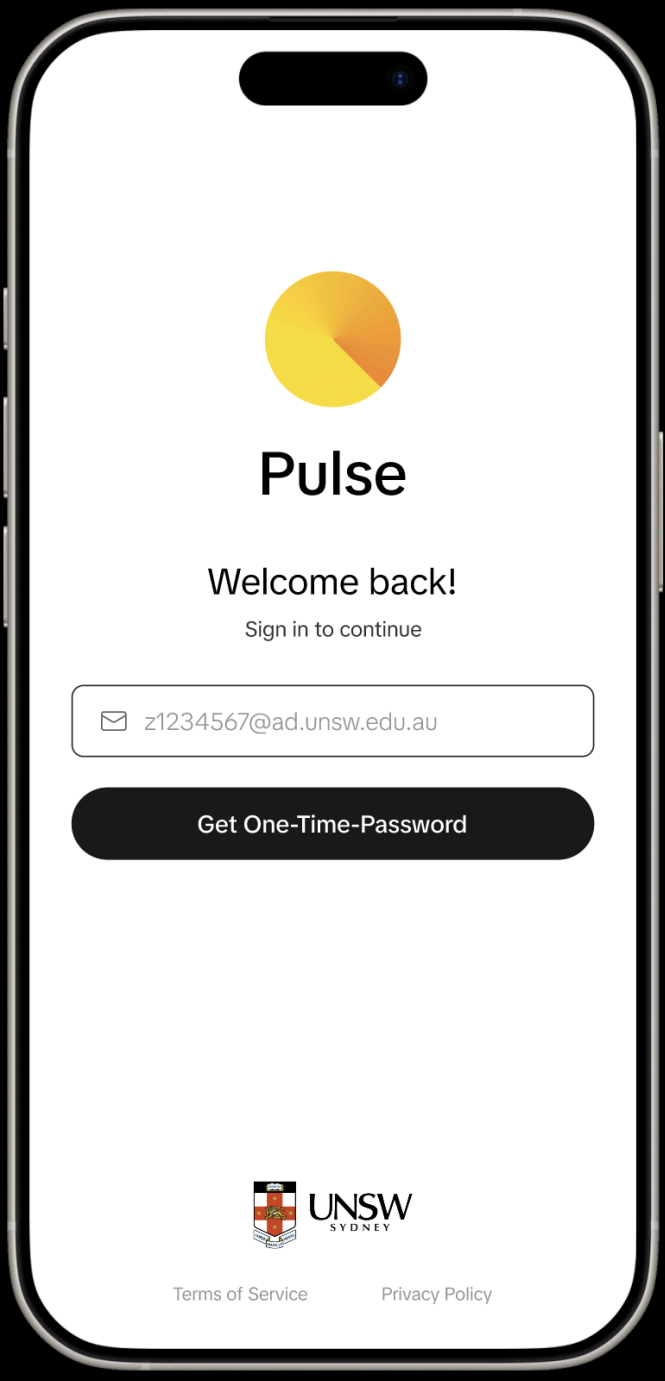
System Architecture Diagram

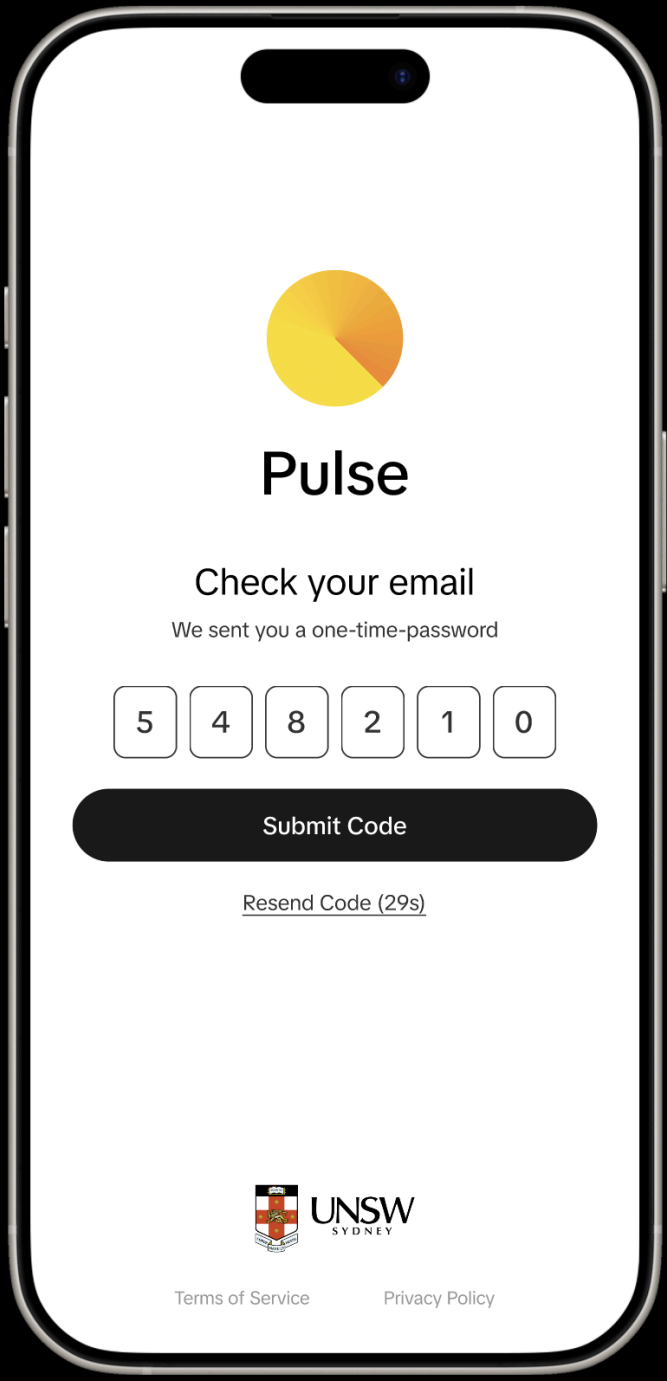


User Navigation Workflow

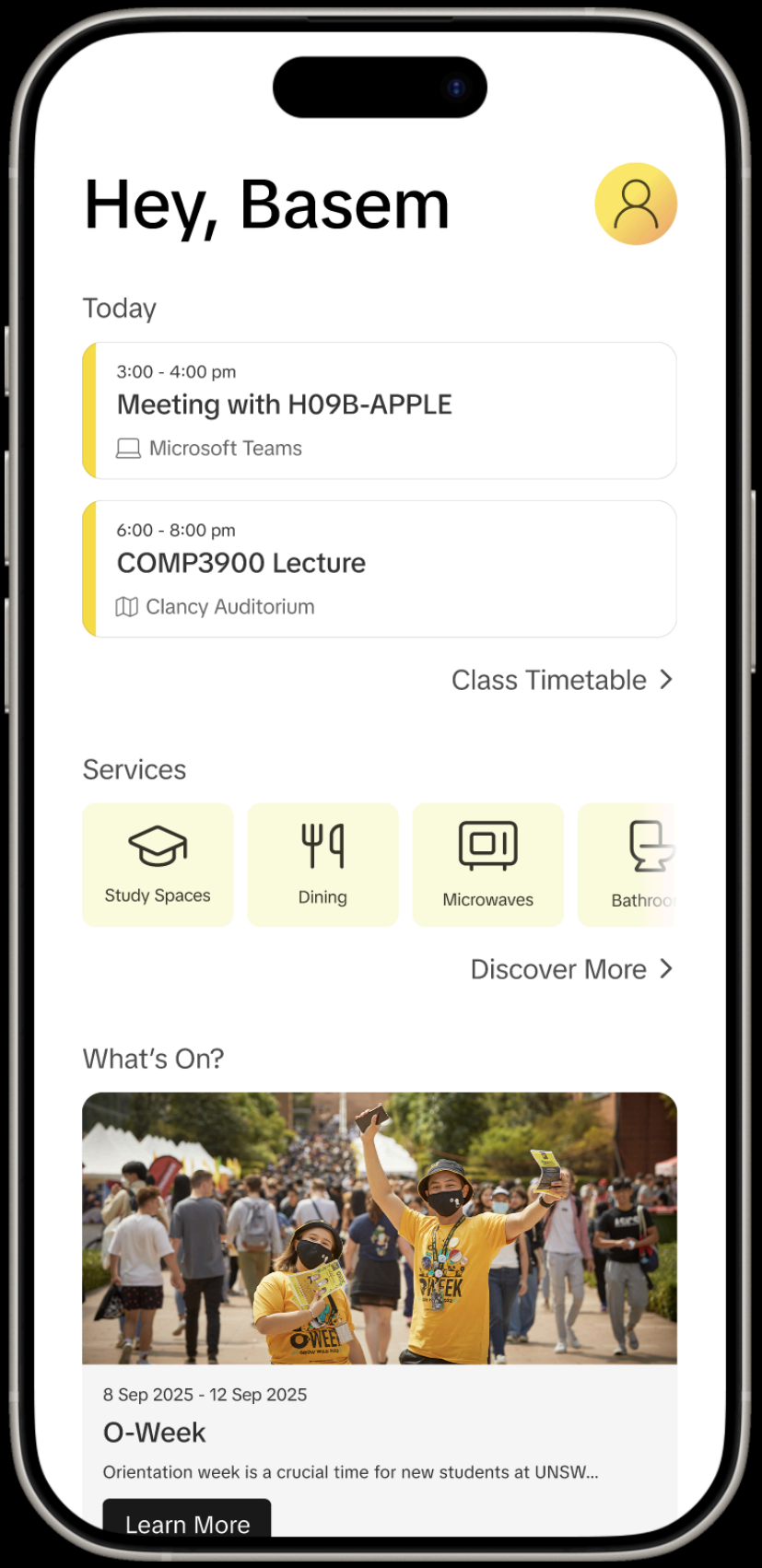


Login Panel

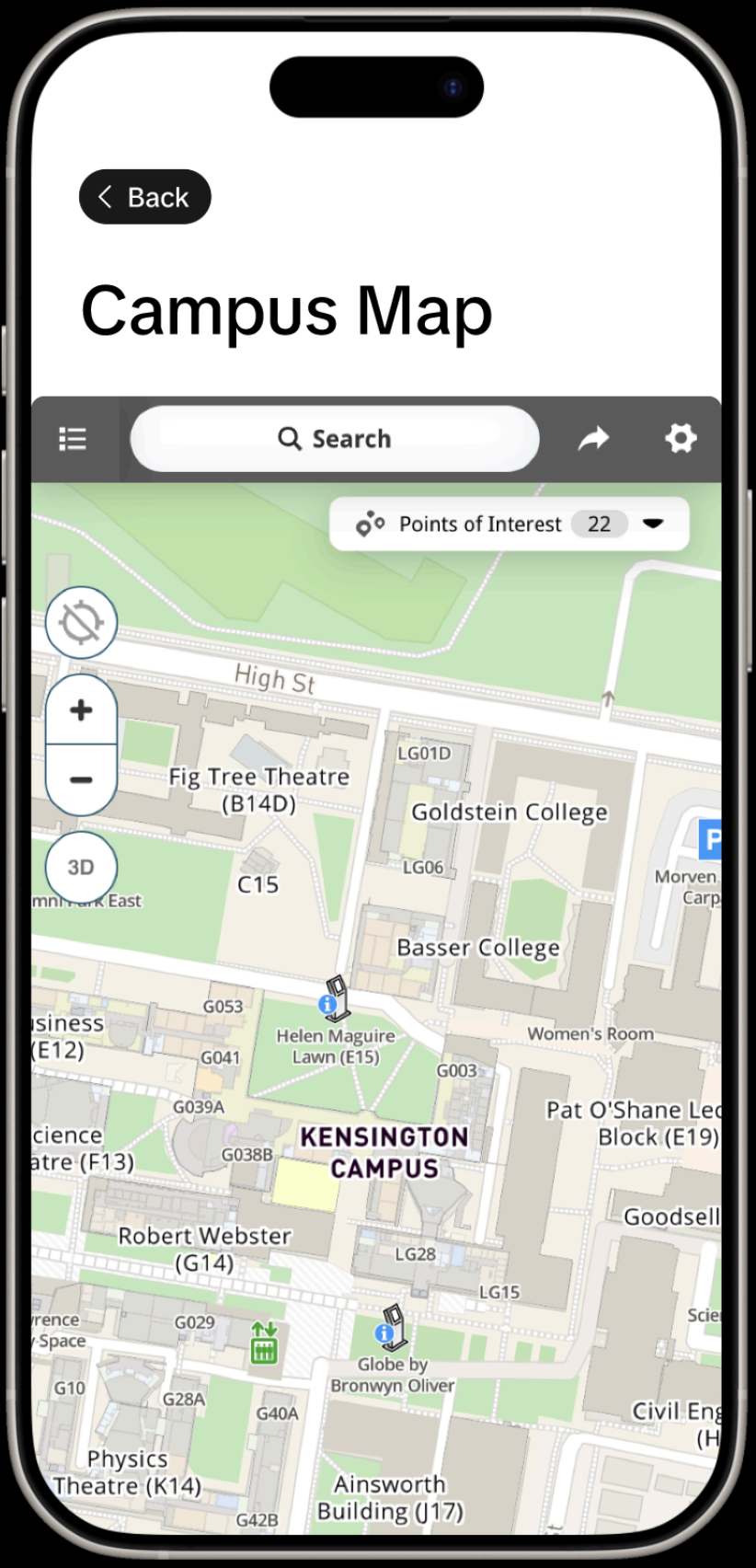

Pulse
Welcome back!
Sign in to continue
z1234567@ad.unsw.edu.au
Get One-Time-Password
UNSW SYDNEY
Terms of Service
Privacy Policy


Pulse
Check your email
We sent you a one-time-password
5 4 8 2 1 0
Submit Code
Resend Code (29s)
UNSW SYDNEY
Terms of Service
Privacy Policy

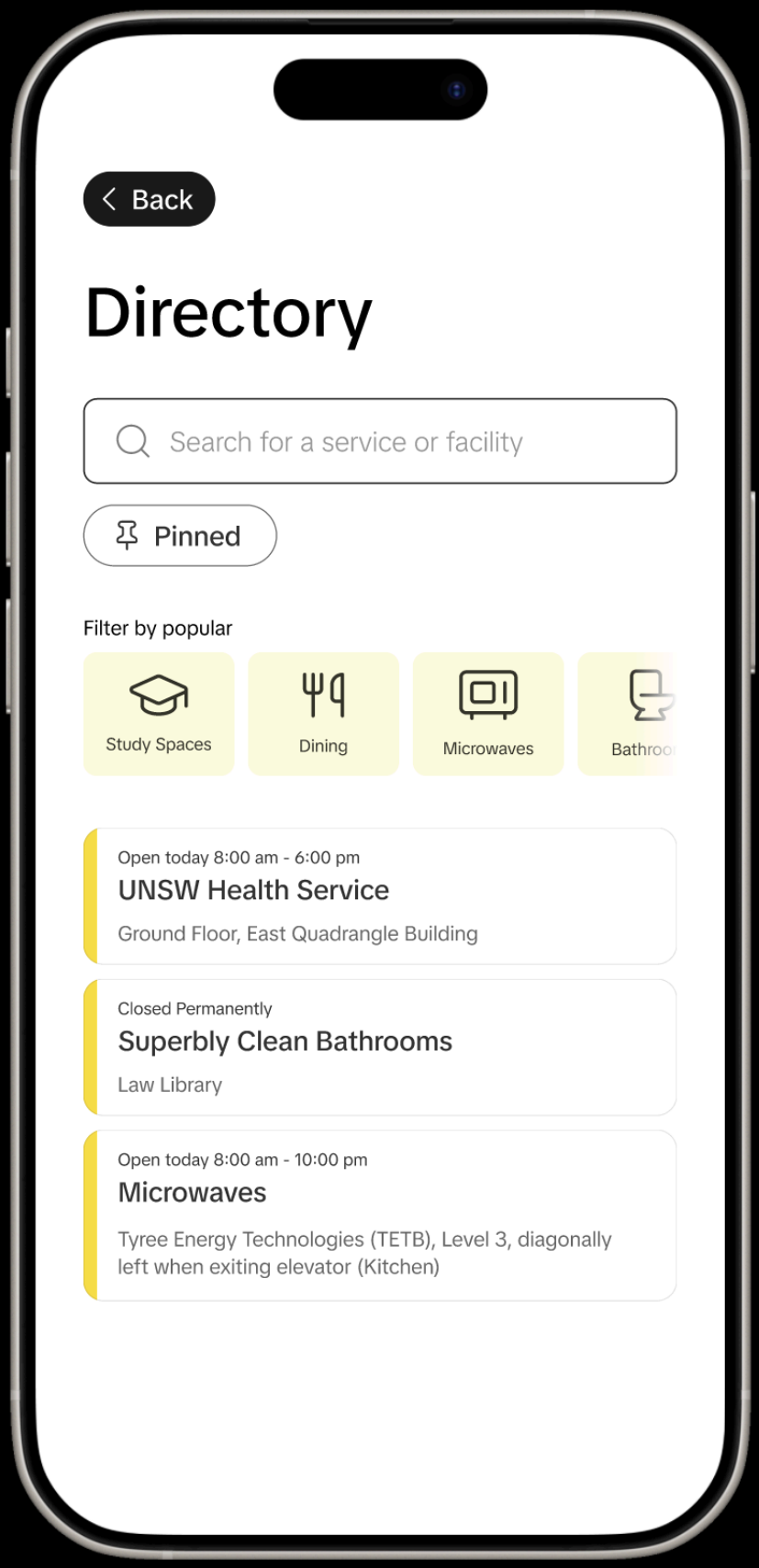
Dashboard Panel



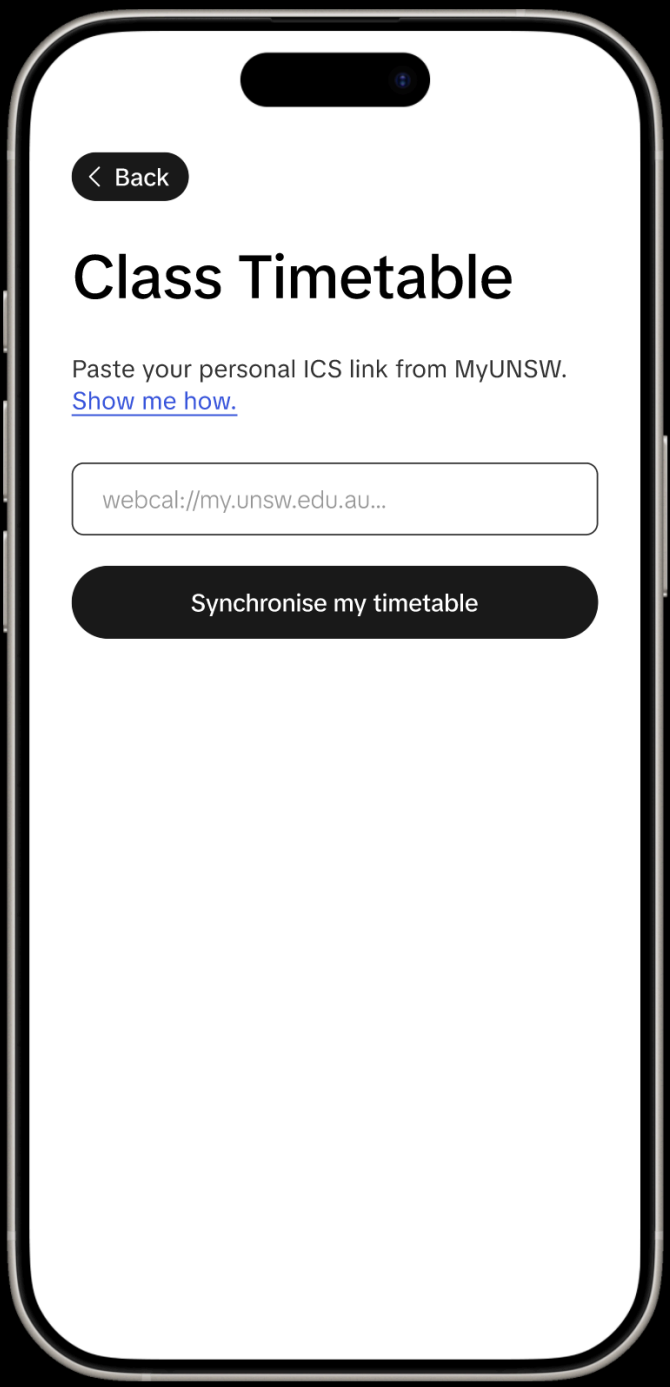
Campus Map Panel

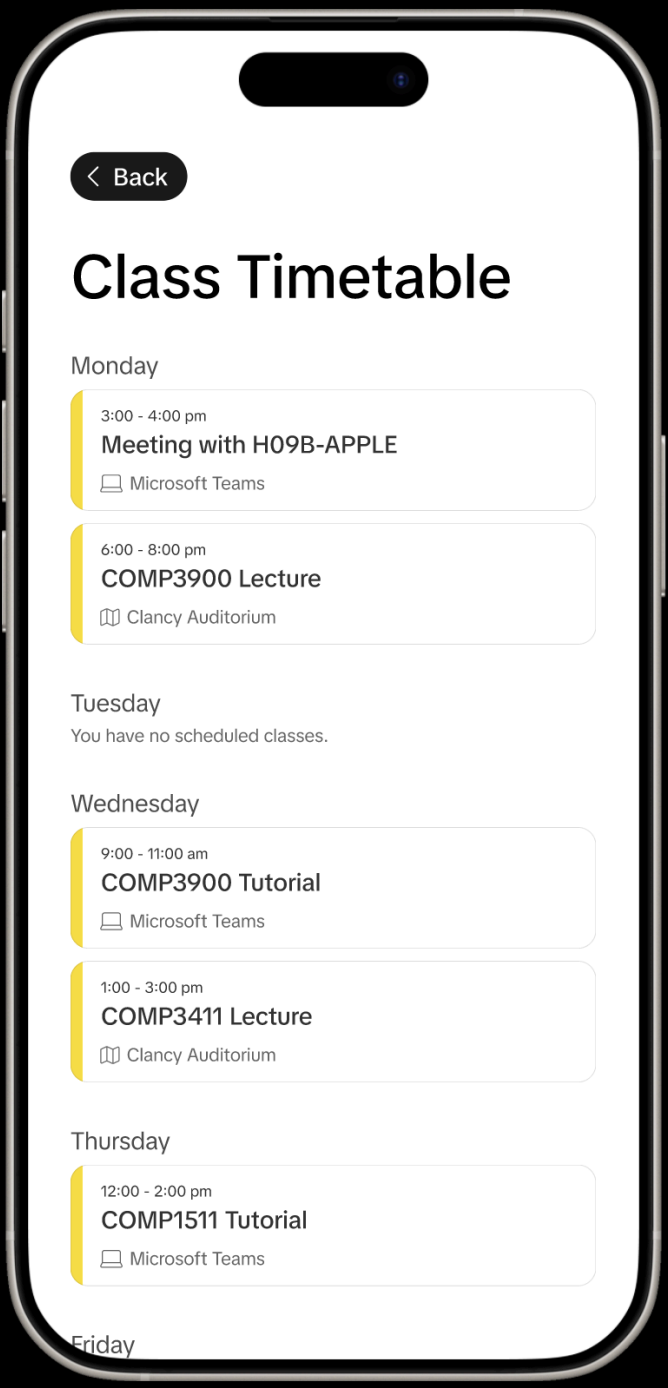


Service Directory Panel

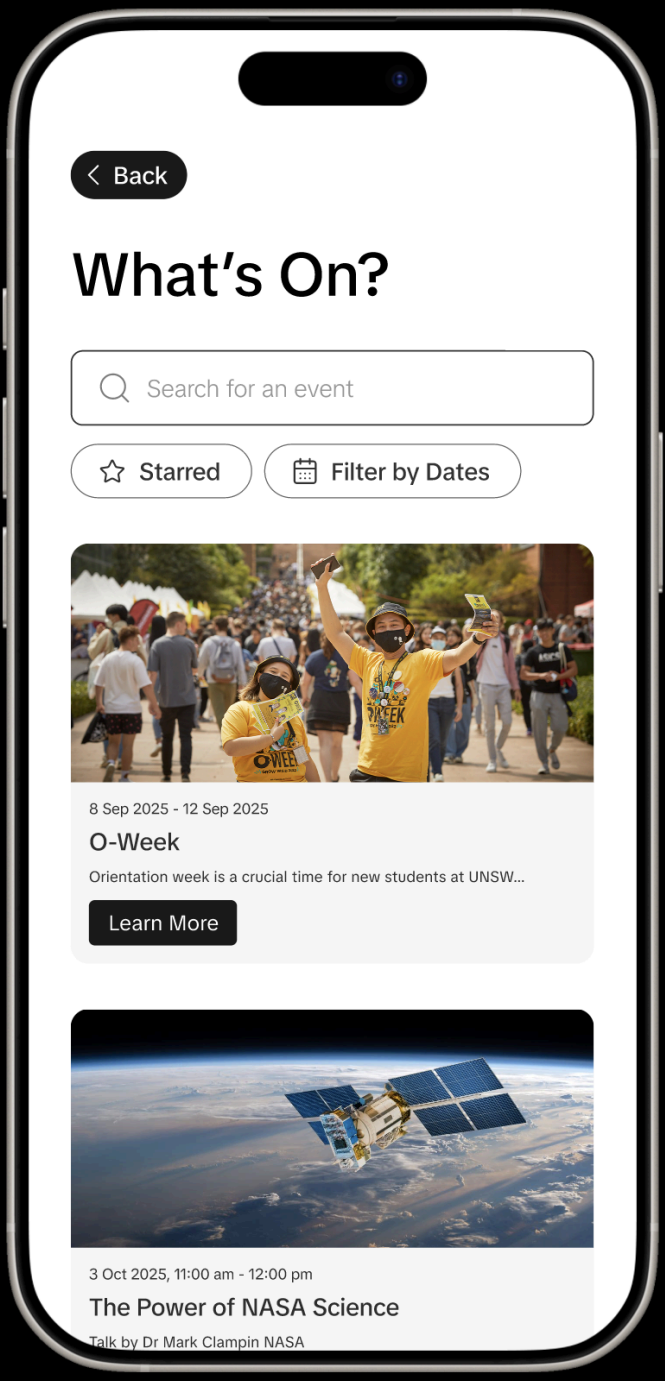


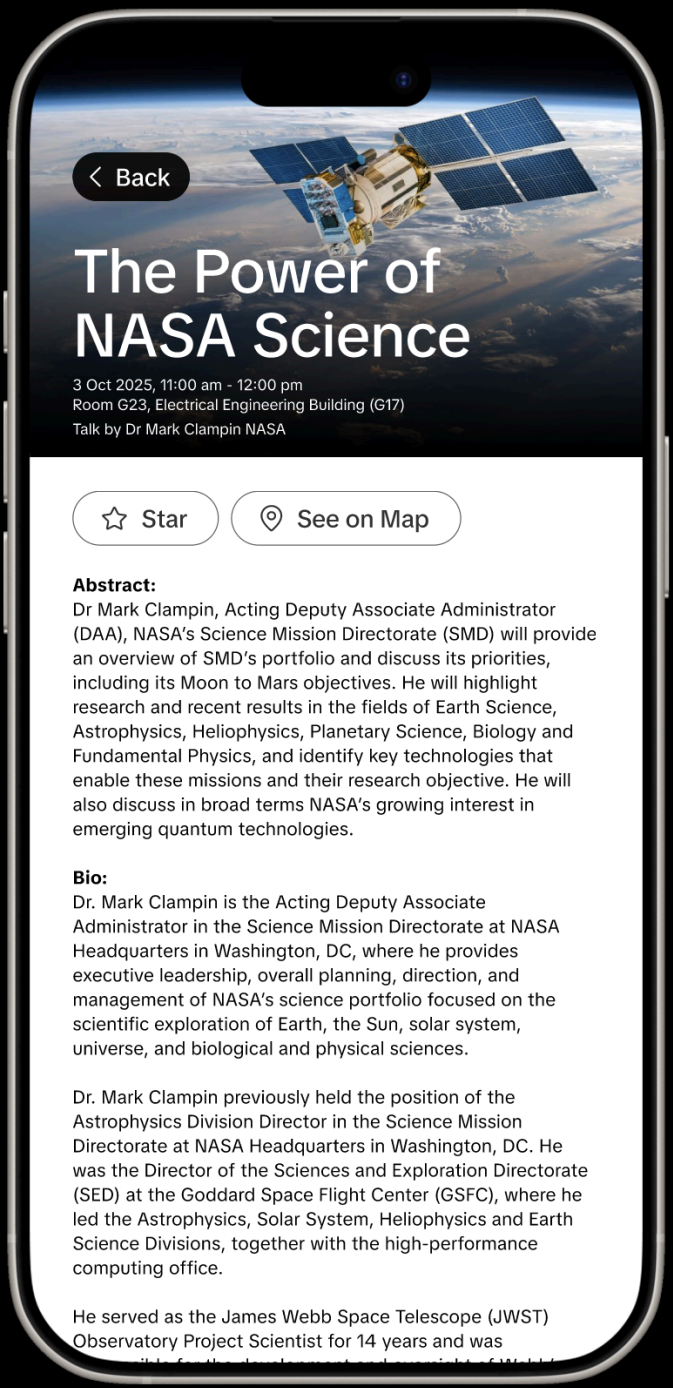
Calendar Panel



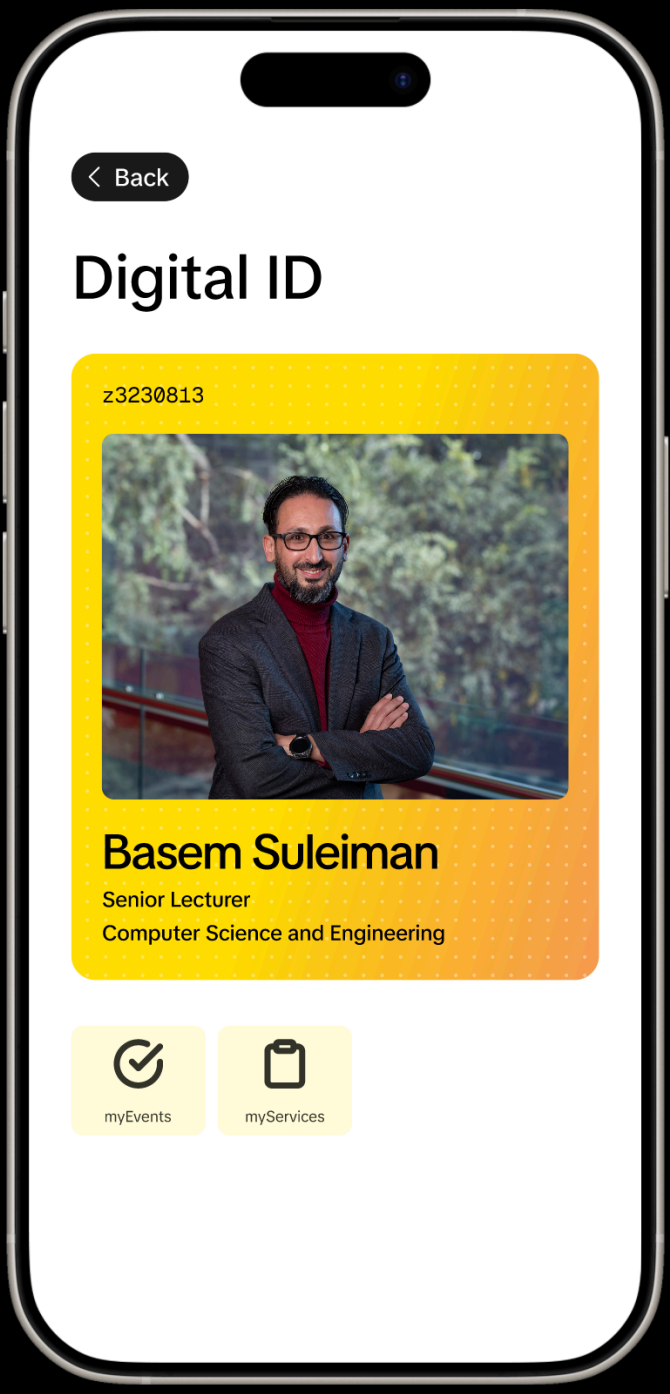


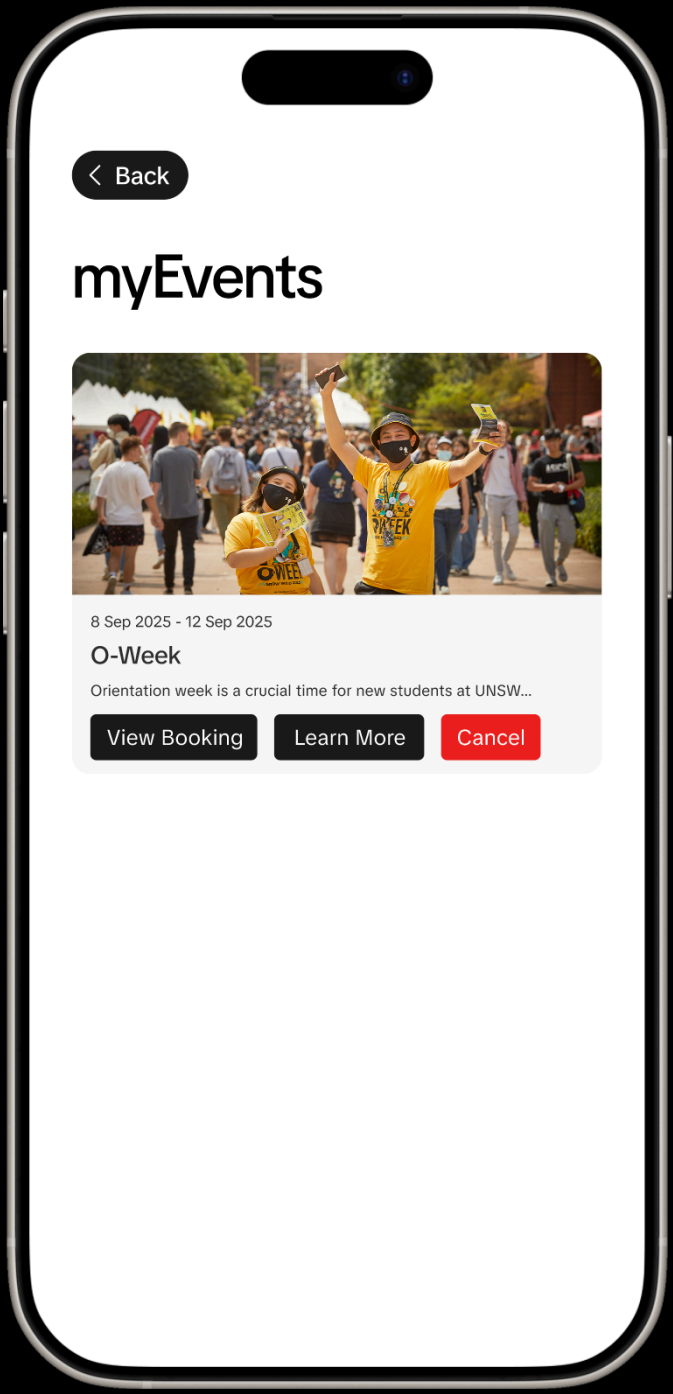
Events Panels





Profile Panels





Design Justification

Breaking down the problem

We broke down the problem by considering user requirements, how users would interact with our services and how our services would need to interact with each other to meet our overall goal. We abstracted the problem into front-end/UI subproblems, subproblems pertaining to architecture (services and business logic) and subproblems around data access and persistence.

Subproblem 1

How can users interact on their phones to access services?

Solution (Frontend)
ServiceUNSW will be implemented using the following stack:

- React Native (0.74+)
- Expo SDK 52+
- TypeScript 5
- React Native Reusables
- TanStack Query
- Lucide Icons
- NativeWind (Tailwind CSS for Mobile)
- WCAG 2.2 AA-compliant styles

Expo will allow ServiceUNSW to deliver consistent and responsive UIs cross-platform across iOS and Android with a single codebase. With Expo, ServiceUNSW will use native and intuitive workflows for high-frequency services (such as Timetable and Digital ID).

Alternatives 1
Alternatively, we considered developing separate native applications (e.g. Swift/Xcode and Kotlin for iOS and Android, respectively). This would have doubled development time and cost. Additionally, it would require a much greater skillset the team for the team to learn. Finally, platform-specific demands would have lead to inconsistent feature parity.

Alternatives 2
Creating a Progressive Web App (PWA) was also considered. This application would have relied on React.JS. However, it would, as a PW lack deep platform integration thus preventing key features such as native notifications and offline capability.

Why we chose the solution
React Native + Expo + TypeScript would enable the development of a visually and behaviourally consistent mobile-first platform, thus successfully reducing context switching (Murty et al., 2022; Quickbase, 2023). It would also promote more student engagement (CAUDIT, 2025). The use of React + Typescript would allow for shared codebase integration with the Next.JS backend. Additionally, it would allow access for platform-specific APIs needed for Notifications, Location Services and Camera which are all features necessary for key services.

Furthermore, React Native has several tools for rapid prototyping and cross-platform testing. DevOps and Build Pipelines are essential to ServiceUNSW's success (as it is a lage project) and Expo provides managed workflows for these.

Novel functionality vs Pre-existing Solutions
Pre-existing UNSW applications, such as the University, offer services (Timetable, Maps, etc) in separate systems, causing fragmentation in user interaction. The new Expo-based mobile interface would unify these sub-features into a single app that would be coherent both in terms of aesthetics and UX. Such an app would take full advantage of the native platform to provide context-aware notifications based on intelligence.

Subproblem 2

How can we design a clean and modular architecture that supports our services?

Solution (Next.js)
We found it beneficial to further divide our backend/database development into controllers (which maintain clean authentication and allow for robust entry point testing), services (which allow the centralisation of all platform clients, mobile and web), and data access layers (for maintaining persistence, schema consistency and supporting future migrations).

Summary (Controllers)
In terms of architecture, we chose to create a layered MVCS structure in which the React Native/Expo frontend is connected to a set of controllers using an API. These controllers then interface with different services and with the data access layers.

Alternative 1 – Monolithic Controller
The first alternative to this architecture would be a monolithic approach where controllers are embedded with business logic. Such an approach is easy and fast to implement, yet harder to test. It is less extensible and goes against the modular philosophy of ServiceUNSW. This is because it involves coupling business logic, communication layers and data access layers.

Alternative 2 – Microservices
Microservices offer high modularity and extensibility; however, they may be considered overkill at a pre-alpha version of ServiceUNSW. The programmer Donald Knuth oncesaid, “Premature optimisation is the root of all evil. Premature optimisations frequently lead to trade-offs, sacrificing implementation time or pressing features.

Additionally, Microservices require more infrastructure (e.g. authentication between microservices and a considered approach to distribution). A single codebase, instead, minimises unnecessary complexity. In the future, ServiceUNSW should consider microservices as it begins to scale.

Why we chose the solution
The division of the architecture into different controllers, and separating controllers from the UI, services and data layers, abides in accordance with the principle of “separation of concerns” while minimising unnecessary complexity. It was noted in our research that “complexity lowered student engagement” (Comunetti, 2025; Handshake UK, 2025), so this alleviates that. The MVCS structure is simple to implement yet maintains modularity and extensibility.

Novel functionality vs Pre-existing solutions
Pre-existing university service systems often have separate APIs for different services. The boundary controllers, in combination with the service adaptors) allow for a single consistent point of contact/API that still integrates the peculiarities of multiple services behind the scenes. Such a system allows for easy development of service application platforms (e.g. VR or other platforms in the future).

Subproblem 3

How do we safely handle data persistence and access?

Solution – Data Access Layers
ServiceUNSW will use PostgreSQL 16 (Neon) and Drizzle ORM to handle data access, storage and management. Collectively, they offer open-source, stable and secure workflows to data persistence, safety and schema management.

Alternative 1 – Firebase Firestore
Firebase Firestore was concerned as an alternative. Although Firebase offers simpler setups, it has weaker approaches to complex relational joins. In addition, as proprietary software, it has the risk of “locking in” the software to the provider. These could affect the extensibility and maintenance of ServiceUNSW in the future.

Alternative 2 – MongoDB Unstructured JSON
MongoDB was considered for its strengths in unstructured data, simple schemas but was ultimately declined for its weaknesses in data structures that emphasise strong relationships and referential consistency. Furthermore, reasonable concerns exist concerning MongoDB's RAM (resource) consumption and its ability to scale.

ServiceUNSW must consider scale as UNSW has 70,000+ students (Microsoft, 2025).

Why we chose the solution
Data access is important to ServiceUNSW as it needs to maintain a consistent and reliable state across the app to maximise user satisfaction and engagement (Comunetti, 2025). To achieve this, Drizzle ORM (an open-source Typescript solution with an emphasis on API and schema safety) and NeonDB (a serverless solution that supports safe scaling) assist in enforcing referential integrity and enabling/managing future migrations. This allows ServiceUNSW to reliably maintain persistent relational data such as across users, events, services and more.

Novel functionality vs Pre-existing
Pre-existing systems use separate databases/spreadsheets to store data concerning events, room bookings and other services. Using a single database unifies the relations between all these data sets. ORMs like Drizzle allow for typed schemas using TypeScript types. This decreases runtime errors and allows for stable, consistent data queries and provides an easy adaptor for API calls to use.

Limitations and solutions

Limitation 1
UNSW SSO may not be ready for implementation in the ServiceUNSW prototype.
Solution: Our current solution is to use one-time passwords and to design our authentication service in a way that mirrors the current standardised SSO sign-in service, and in addition, allows for the one-time passwords to be swapped in the future with SSO while minimising code churn.

Limitation 2
The reliability of the UNSW API cannot be guaranteed. The Uni-verse app has been subject to time-outs.
Solution: ServiceUNSW will display "last updated" timestamps so that users will always be aware of outdated/stale data.

Limitation 3
ServiceUNSW cannot guarantee the accuracy of the UNSW Scout Agent, especially in regards to sensitive topics. For example, students reaching out concerning mental health or to make important decisions regarding UNSW policies. LLMs have been shown to hallucinate, and the mechanism for hallucination is not totally understood.
Solution: ServiceUNSW will reference sources it uses. It will document user feedback to iteratively improve LLM functionality.

Database Schema

Our proposed database schema is detailed below. We understand that this schema will likely evolve throughout the course of the project, particularly as client requirements advance and as we implement the authentication-specific tables required by our authentication library (BetterAuth).

```
Users(
  id: string,
  zid: number,
  email: string,
  name: string,
  enrolment_year?: number,
  created_at: timestamp,
  updated_at?: timestamp,
  deleted_at?: timestamp,
  calendar_link: string,
  access_level: enum {
    USER,
    ORGANISATION_ADMIN,
    ADMIN,
})
```

```
Organisations(
  id: string,
  userId: string,
  name: string,
)
```

```
OrganisationUser(
  id: (organisationId, userId).primaryKey(),
  userId: string,
  organisationId: string,
)
```

```
Services (
  id: int,
  name: string,
  description: string,
  location?: string,
  link: string,
  phone?: number,
  email?: string
  created_at: timestamp,
  updated_at?: timestamp,
  deleted_at?: timestamp,
)
```

```
Events (
  id: string,
  owner: foreign key Users.id,
  organisation: foreign key Organisations.id
  name: string,
  description: string,
  location?: string,
  start: timestamp,
  end: timestamp,
  created_at: timestamp,
  updated_at?: timestamp,
  deleted_at?: timestamp,
  capacity: number constraint capacity >= 0,
  price: int constraint price >= 0,
)
```

```
EventsBookings (
  id: (event_id, booker_id),
  event_id: foreign key events.id
  booker_id: foreign key Users.id,
  created_at: timestamp
)
```

Functionality Mapping

Function / Objective	User Story	Implementation	Justification
Login / Authentication	As a UNSW staff or student, I want to log in using my @ad.unsw.edu email and a one-time password so that I can securely access the application without needing a permanent password.	Authentication in this project will be conducted using local testing data to use for proof of concept.	Due to uncertain access of the UNSW Single-Sign-On API, we will focus on realising a proof of concept and modularity to ensure smooth handover.
Dashboard	As a user, I want a functional and intuitive dashboard page so that I can easily access key features, view important information at a glance, and navigate through the platform efficiently.	Comprehensive dashboard that users that clearly displays core information using React Native's existing libraries and additional custom components.	Creating a dedicated dashboard will minimise user friction related to navigation of the app by allowing the user to reach other pages and features with the fewest number of taps.
Events	As a student, I want to view a list of upcoming events, so I can stay informed and sign-up for those I plan on attending.	An intuitive events page that contains a list of all upcoming events	Collecting all relevant events into a single page (infinite scroll) allows the user to quickly browse upcoming events.
Event Details	As a student, I want to view detailed information about an event, so that I can conveniently learn about and join events without switching between multiple platforms.	A page that contains all relevant details of an event such as an event's name, time, location, and description.	This detailed page allows users to consider all relevant information regarding an event before deciding to sign-up or bookmark it. This allows them to have a full understanding of the event without needing to resort to external platforms.
Events Creation	As an organisation administrator, I want an intuitive event creation portal where I can easily set up and publish new events with minimal technical difficulty or confusion.	A form-based event creation module will be built with React Native , Zod (for form validation), and the React Form hook, connected via REST API to	This setup allows creation of events by admins while maintaining a consistent structure. It also aligns with modular service design for future expandability (e.g. event templates, recurring events).

Functionality Mapping

		backend controllers managing event records in PostgreSQL via Drizzle ORM.	
Bookmarked Events Page	As a student, I want to access a page that displays all the university events I've booked, so that I can easily review, manage, and remove events I'm no longer interested in attending.	An "I'm going" button triggers API calls to register event attendance; confirmation stored in the EventsBookings table with event and user foreign keys.	Integrating bookings directly into the event view streamlines the user experience and eliminates fragmentation between systems. A relational booking model ensures data consistency and provides a strong query foundation for the user-facing interface.
Service Directory	As a student, I want to search/filter/browse a list of campus services(i.e. IT help, library, counselling) so that I can find and access the service I need.	Create a React Native SearchBar component powered by PostgreSQL with indexed search queries using Drizzle ORM. Search and filter implemented using fuzzy text search and tag-based categorisation.	Consolidating all UNSW services into one searchable directory removes the need for multiple websites. Tagging and fuzzy matching improves discoverability.
Timetable	As a student or staff member, I want to view my academic/staff calendar in the app by supplying an .ics link so that I can see key dates and events in an accessible weekly view.	API call to process uploaded calendar URLs. The app periodically fetches the latest version of the calendar which is displayed using a React Native component.	To avoid being blacklisted by calendar API providers, the app will restrict the refresh limit. Integrated calendar makes for easy viewing of a user's class timetable.
Academic Calendar	As a student, I want key academic calendar dates (such as census dates) to be clearly highlighted within the app, so that I can conveniently stay aware of important deadlines and plan my schedule effectively.	Static academic calendar data fetched from UNSW's public site (scraped or imported yearly) and stored in PostgreSQL.	Static integration ensures fast and offline-accessible data while maintaining consistency with UNSW's published dates. Highlighting key events reduces missed deadlines.

Functionality Mapping

Campus Map	As a student or staff member, I want to view the UNSW campus map within the app so that I can quickly find locations on campus.	Embed the existing UNSW campus map as a frame within the app.	Current UNSW map offers additional functionality such as directing users to specific floors and rooms that native map apps do not currently support.
Digital ID	As a student, I want to present a digital UNSW ID card in the app so that I can show basic identity for campus use (events, building entry checks) without high-security verification in MVP.	ID rendered dynamically from user profile data (name, zID, role). QR code generated for potential scanning at events. Future versions may integrate UNSW SSO or NFC when API access is available.	Digital ID provides convenient low-security verification (attendance, access) without full integration into UNSW's secure ID systems. Modular design means the feature can later expand into official validation workflows once APIs are accessible.
AI Chatbot	As a student, I want an AI chatbot to ask questions in natural language, so that I can get instant, accurate answers about UNSW services, course information, administrative processes, and deadlines without extensively searching through UNSW websites	Use UNSW's own generative chatbot API (Scout) to deliver user prompt and fetch response to be displayed in a chat-based user interface.	UNSW's own chatbot offers specific answers to UNSW-related questions, making it the optimal solution tailored for our app's user base.

04

Citations

[1] Bopp, M., Sims, D., Matthews, S. A., Rovniak, L. S., Poole, E., & Colgan, J. (2016). There's an app for that: Development of a smartphone app to promote active travel to a college campus. Journal of Transport & Health, 3(3), 305–314. https://doi.org/10.1016/j.jth.2016.02.007

[2] Comunetti, C. (2025, April 1). Mobile-First Student Engagement in Higher Education – raftr. Raftr – Community Apps, Reimagined.https://www.raftr.com/mobile-first-student-engagement-in-higher-education/

[3] Council of Australasian University Directors of Information Technology. (2025). 2025 Top Ten. https://www.caudit.edu.au/resources/2025-top-ten-report/

[4] e2s Team. (2024, October 3). How can university mobile apps enhance student retention? Engage2Serve. https://www.engage2serve.com/blog/university-mobile-apps/

[5] Faw, L. (2018, March 5). Workers Waste 32 Days A Year Due To "Workplace Efficiency" Apps. Forbes. https://www.forbes.com/sites/larissafaw/2018/03/05/workers-waste-32-days-a-year-due-to-workplace-efficiency-apps/?sh=446ad39a1d51

[6] Gkrimpizi, T., Peristeras, V., & Magnisalis, I. (2023). Classification of barriers to digital transformation in higher education institutions: Systematic literature review. Education Sciences, 13(7), 746. https://doi.org/10.3390/educsci13070746

[7] Handshake UK. (n.d.). Why careers services need mobile apps to connect with Gen Z students. Retrieved 8 October 2025, from https://joinhandshake.co.uk/blog/careers-services/why-careers-services-need-mobile-apps-to-connect-with-gen-z-students

[8] Microsoft. (2025, August 4). UNSW's Scout: Simplifying student life with agentic AI technology. Microsoft Source (Asia). https://news.microsoft.com/source/asia/2025/08/04/unsws-scout-simplifying-student-life-with-agentic-ai-technology/

[9] Murty, R. N., Dadlani, S., & Das, R. B. (2022, August 29). How Much Time and Energy Do We Waste Toggling Between Applications? Harvard Business Review. https://hbr.org/2022/08/how-much-time-and-energy-do-we-waste-toggling-between-applications

[10] Quickbase. (2023). Report: 70 Percent of Workers Lose 20 Hours a Week to Fragmented Systems. Quickbase. https://www.quickbase.com/about-us/media/report-70-percent-of-workers-lose-20-hours-a-week-to-fragmented-systems

[11] Service NSW. (2025, March 21). App-ealing features boost Service NSW accessibility, security and functionality. https://www.service.nsw.gov.au/news/app-ealing-features-boost-service-nsw-accessibility-security-and-functionality

[12] Service NSW. (2025). Service NSW (Version 9.16.0) [Mobile app]. Apple App Store. https://apps.apple.com/au/app/service-nsw/id577283433

[13] University of New South Wales. (2017). UNSW Uni-Verse: Official (Version 11.30.0) [Mobile app]. Google Play. https://play.google.com/store/apps/details?id=com.ombiel.campusm.unsw&hl=en_AU&pli=1

[14] University of Newcastle. (2018). Mobile apps. University of Newcastle. https://www.newcastle.edu.au/current-students/support/it/student-systems/mobile-apps

[15] University of Newcastle. (2018). myUni University of Newcastle (Version 11.30.0) [Mobile app]. Google Play. https://play.google.com/store/apps/details?id=com.ombiel.campusm.myUON&hl=en_AU

[16] University of Sydney. (2025). Apps. University of Sydney. https://www.sydney.edu.au/students/student-it/apps.html

[17] University of Sydney. (2025). Sydney Uni (Version 6.3.4) [Mobile app]. Google Play. https://play.google.com/store/apps/details?id=com.blackboard.android.central.sydney_au&hl=en