

Muqtasid Mohammad Hyder

Portfolio



Software Engineering Student, RMIT University · Melbourne

muqtasidmohammadhyder@gmail.com / s4058801@student.rmit.edu.au · muqtasid.site · github.com/muqtasid21

Dishcovery

iOS app · Apple Foundation Program · Team of 4 · Jun – Jul 2026

WHERE IT STARTED

We knew from day one that we wanted to help our community. What we didn't know was which community, or how. We spiralled on that for days. Every idea felt either too broad to be useful or too narrow to matter, and our sticky note habit was starting to worry the tutors.

THE REFRAME

We stopped looking outward and looked at ourselves. All four of us were second-generation immigrants. We weren't designing for a stranger, we were designing for us. What we actually wanted was a way back to our roots, and food was the door in. Everyone has a dish that means home, and everyone is quietly certain theirs is the best one.

THE MESSY MIDDLE

The design spiralled as hard as the concept did. We sketched, threw it out, then sketched again, round after round on paper and in Figma before anything felt right. Most of what we made never shipped, which sounds like a waste until you see the early drafts.

WHAT I BUILT

The SwiftUI front end: the world map screen, the lesson and story flow, the quiz system, the leaderboard, the passport and stamp collection, and a custom animated tab bar that I may have spent an unreasonable amount of time perfecting.

WHAT I LEARNED

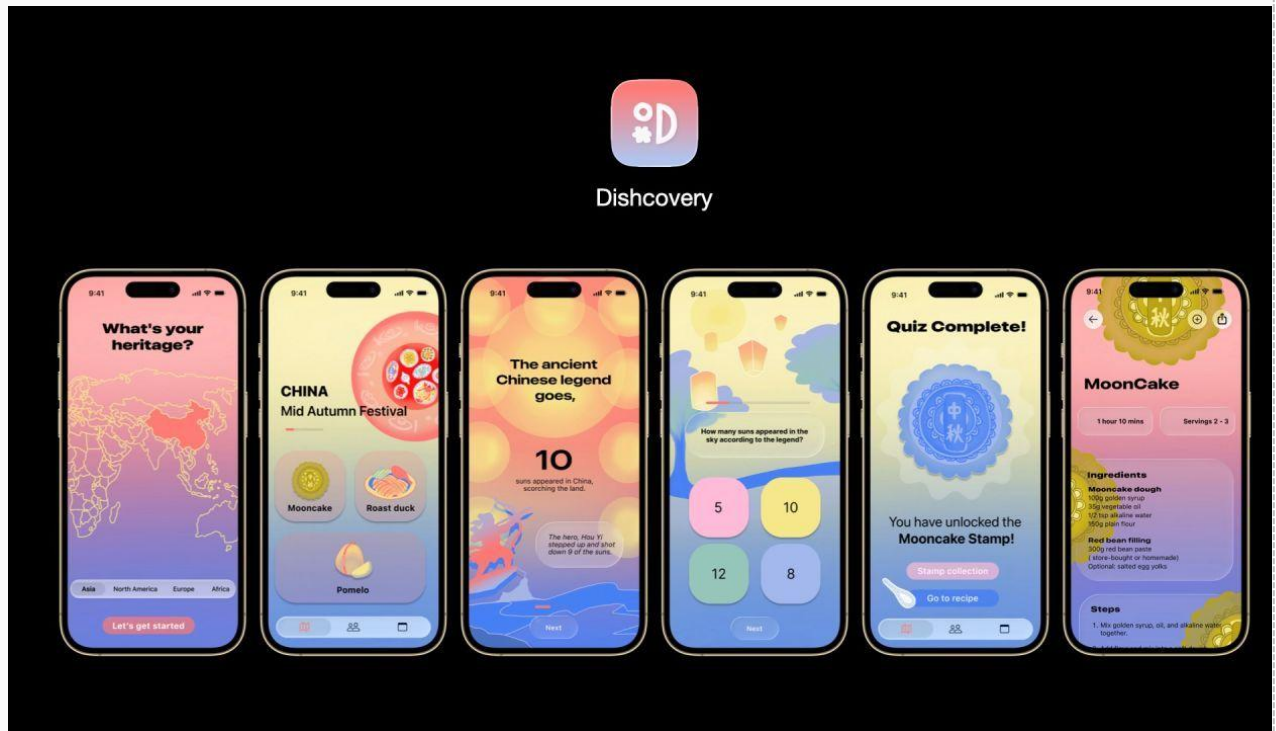
That the fastest way out of a spiral is to stop guessing at strangers and interrogate your own experience. And that a design isn't finished when it looks good, it's finished when it stops getting in the way of the story. Also that four people can agree on everything except colour schemes.

Swift · SwiftUI · Figma · UI/UX · Teamwork

PROCESS & OUTCOME



Early paper sketches, mostly ideas that didn't survive



Final app screens



Our team with the AFP mentors

Venue Vendors

Full-stack venue booking platform · Team · Mar — Jun 2026

THE STARTING PROBLEM

The brief's documentation was ambiguous. In several places it wasn't clear what the user actually wanted, and reading it a second time somehow helped less. Rather than guess and build the wrong thing, we had to work out the intent behind the requirements before we wrote any code.

THE REFRAME I'M PROUDEST OF

We realised we were the users. As students we'd rented venues ourselves, and we knew something the brief never mentioned: when you book venues regularly, you want your favourites kept close. So we added venue ranking, letting hirers order their preferred venues. That feature came from lived experience, and from scrolling past the same hall one too many times.

THE MESSY MIDDLE

Our ER diagram needed real rework. Some data wasn't being extracted properly and the relationships had to be redrawn. The GraphQL layer behind the admin dashboard also developed opinions of its own. Both meant going backwards to fix foundations instead of pushing forward on features, which is the software equivalent of eating your vegetables.

WHAT I BUILT

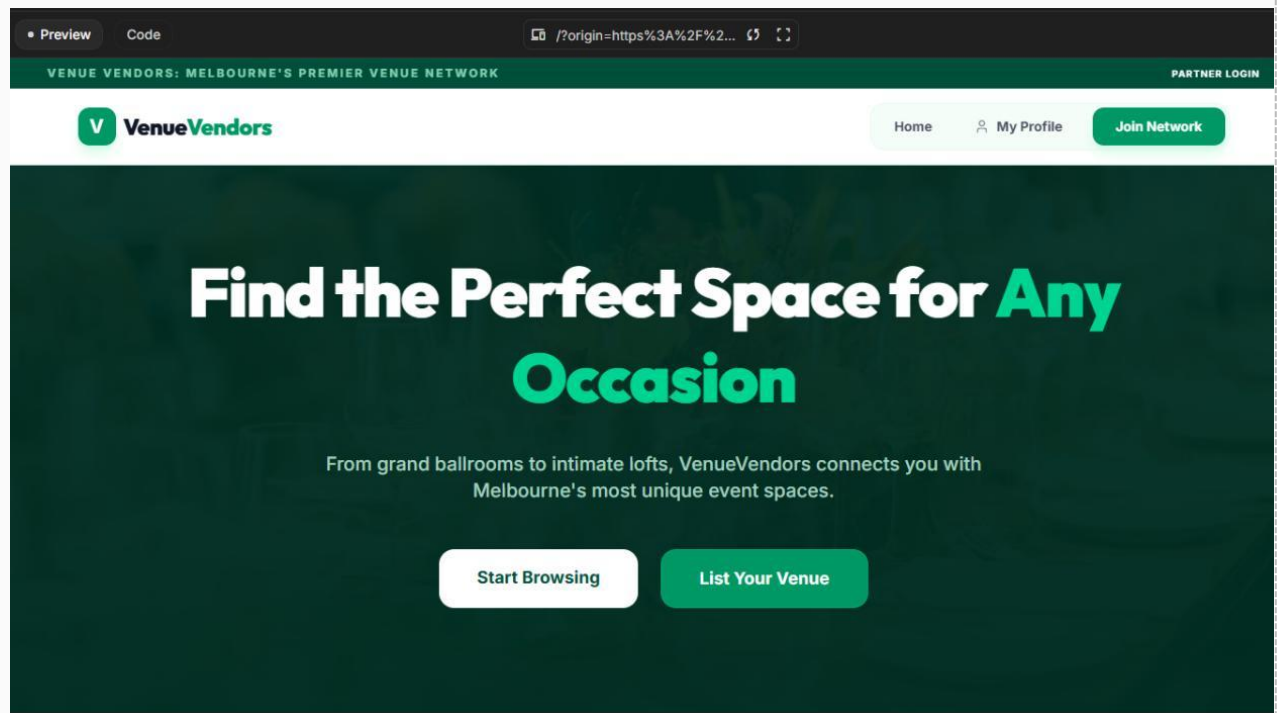
Role-based dashboards for hirers, vendors and admins, JWT authentication with bcrypt hashing and role-based access across the REST endpoints, and the separate GraphQL admin dashboard with Apollo that I am still slightly proud of.

WHAT I LEARNED

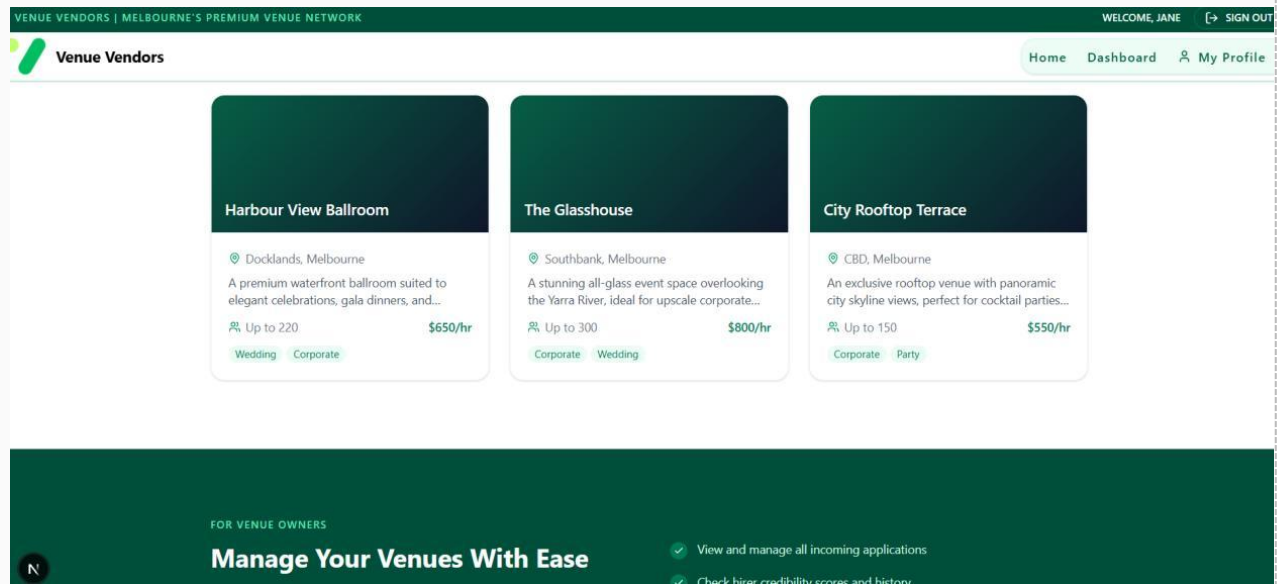
That ambiguous requirements are a prompt to go find the user, not a blocker. And that fixing a data model properly early is cheaper than every workaround you'd otherwise write later, a fact our commit history will cheerfully confirm.

React · TypeScript · Next.js · Node.js · Express · TypeORM · MS SQL (AWS RDS) · GraphQL · Apollo

THE PRODUCT



Homepage



Venue detail page

Live: venue-vendors-frontend-orkr.onrender.com

Code: github.com/muqtasid21/venue-vendors

Sky Shows for Sport

Formula 1 · Sela · Saudi Pro League · drone & pyrotechnic production · 2023

WHY I WAS THERE

I'm a sports fan first. I know exactly what it feels like to be in a crowd when something lands, and I wanted to give that feeling to other people. That's the only reason I took this work, and it's why the long nights never really felt like work. Most of them, anyway.

WHAT WE MADE

Drone shows and pyrotechnics for major sporting moments: the Saudi Pro League club unveils for Ronaldo, Neymar and Benzema, and Formula 1 events. Every show meant deciding which drones, how many of them, and what shapes would actually read against a night sky. It turns out 'it looked great on the laptop' guarantees nothing at all.

REFRAMING THE PROBLEM

For the F1 shows, a drone display on its own felt too calm for a sport that's all noise and intensity. So we pushed to add pyrotechnics to it, because the fire was the point. I handled the pyro work and fed ideas into the aerial designs, and I was entirely too happy about being handed the fireworks.

EMBRACING THE FAILURES

Designs that looked perfect on paper didn't read in the sky during rehearsal. Shapes blurred, timing fell apart, and we went back and redrew them. Some drones couldn't handle the power draw and dropped out mid sequence, which is the polite way of saying they fell out of the sky. That meant long nights repairing broken drones so the show could fly the next evening.

WHAT I LEARNED

That a creative idea isn't real until it survives a rehearsal, and the gap between 'this will look amazing' and 'this works at 2am with half the fleet down' is where the actual work lives. Prototyping isn't only a software thing. We were iterating in the sky, at a considerably higher cost per mistake.

Pyrotechnics · Drone operations · Creative direction · Live production

THE SHOWS



The unglamorous half: fixing drones so the show could fly



drone-video.mp4

Filmed on my phone at the show

Full show video: https://youtu.be/2e-CaeFV0_4

Procedural Village Generation

C++ algorithms in Minecraft · Team of 3 · Sep – Nov 2025

STARTING FROM BEHIND

This was a first-year subject and I genuinely didn't have the algorithms knowledge going in. I learned BFS and DFS while building with them: pathfinding roads between houses, terraforming, plot finding and validation. Learning to swim by being thrown in, essentially.

WHY IT CLICKED

Applying it inside Minecraft made it fun in a way a problem sheet never would have. Watching an algorithm I'd only just learned carve roads between buildings in a world I could walk through made the theory concrete. It is genuinely difficult to be bored by a graph traversal you can go and stand inside.

THE GOLD MINE MOMENT

Late in the project we discovered the mcpp Minecraft C++ library. Finding it felt like unlocking something, and it opened up what was possible right when we thought we'd hit our ceiling. Brilliant discovery, catastrophic timing.

WHAT I CONTRIBUTED

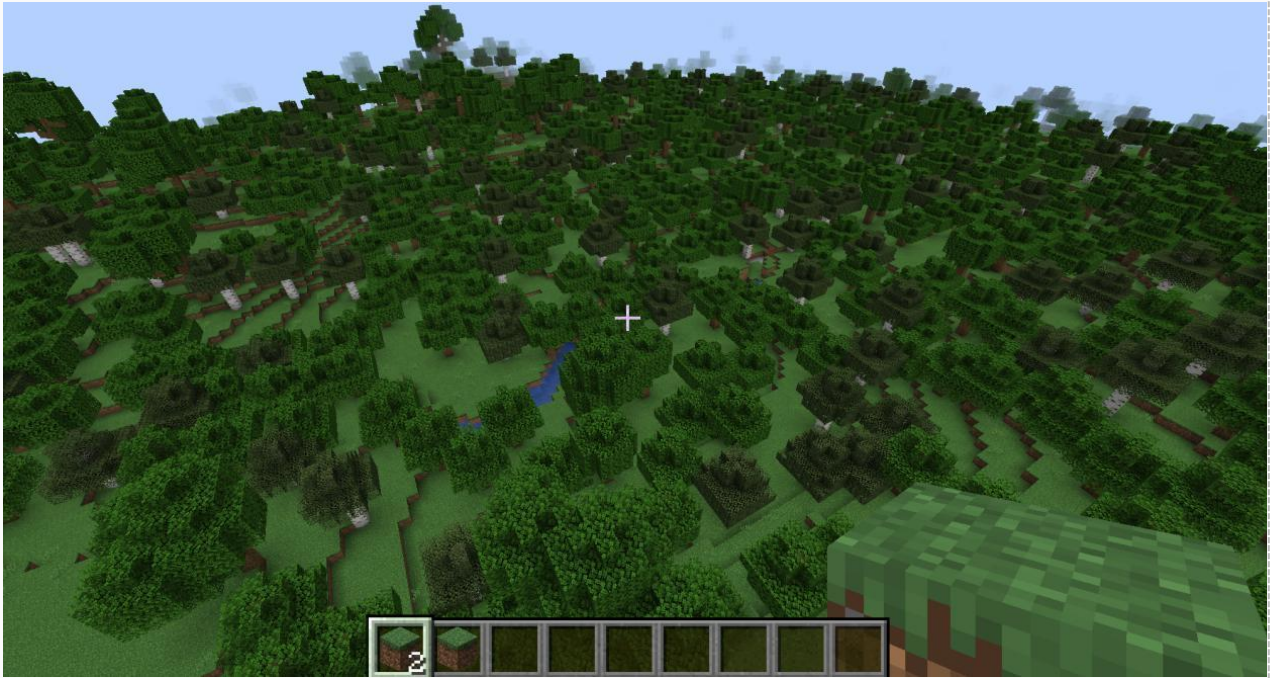
Core plot-finding and validation algorithms, terraforming logic to flatten terrain for foundations, and deterministic test suites so generation stayed consistent across seeds instead of producing a new surprise every run.

WHAT I LEARNED

That not knowing something yet isn't a reason to stay out of the room. I've been recommending this project to peers ever since, possibly more often than they would like. It's the one that made algorithms feel like play instead of homework.

C++ · mcpp · BFS / DFS · Procedural generation · Testing

BEFORE & AFTER



Before: raw, uneven terrain



After: terraformed, with generated boundary walls

Code: github.com/muqtasid21/minecraft-village-generation

Longform

Attention-span trainer & task organiser · Solo build · In progress, 2026

WHY I'M BUILDING IT

Short-form video has quietly wrecked a lot of people's attention spans, mine included. I watched people in my own family struggle to hold focus on things they genuinely wanted to do. Most apps just block distractions, which works beautifully right up until you uninstall them. I wanted something that rebuilds the muscle instead.

THE IDEA

Focus sessions that start at whatever length you can actually manage, even five minutes, and grow as you build the habit, paired with a task organiser that only ever puts one thing in front of you. Calm by design: no red badges, no guilt, no clutter and absolutely no passive aggressive streak reminders.

WHERE I'M UP TO

Backend is running: a Node/Express API over SQLite with tasks and focus sessions, plus the logic that grows your suggested session length as you complete more. The React front end is next. I'm keeping notes on everything that breaks along the way, and that document is growing faster than the app.

WHAT I'M LEARNING

[Filling this in as I go. The honest version, not the tidy one.]

React · TypeScript · Node.js · Express · SQL