UNSW
AUSTRALIA

School of Computer Science and Engineering

Faculty of Engineering

The University of New South Wales

mySchool: a prototype school management system

by

Timothy Yue Hay Hor

Thesis submitted as a requirement for the degree of

Bachelor of Engineering (Hons) in Software Engineering

Submitted: June 2019 Student ID: z5019242

Supervisor: Dr. John Shepherd Topic ID: 33932

Abstract

Administrative tasks commonly performed by UNSW staff and students cover a range of areas including people management, seminar listings, course information and job notices. However, the lack of a unified interface to facilitate these activities may make them difficult to carry out. This thesis involved developing a web-based application to integrate various self-service modules and provide easy access to core school functions – a system which may be adopted by any school or faculty within UNSW. Background research indicates that the aforementioned areas are the most common among the existing websites or systems of different schools, so these were offered as the key sections of the solution.

The system architecture features a responsive front-end built using Angular and an API-based, database driven back-end built using Django with PostgreSQL. Data can be kept up to date via a collection of Python scripts that interface with various sources, while releases are automated via a continuous delivery pipeline that tests and deploys new code whenever a build is triggered.

Overall, the end product is a generic yet extensible application which successfully meets expected outcomes regarding usability, correctness, performance and security. Future work may continue building on this foundation with additional functionality such as teaching allocation or results management.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

At some point during their time at UNSW, most staff members or students will have to perform various administrative tasks related to their school or faculty. Examples might include looking up details on other staff members, finding seminars to attend, checking course information or searching for jobs. However, despite being such common activities, different schools may have inconsistent data or user interfaces, making it unnecessarily complicated to achieve these tasks.

School management systems can assist to some extent by providing people with access to various self-service modules. Staff and students would both stand to benefit from such a system; however, not every school or faculty at UNSW has one in place. Even those that did, such as the School of Computer Science and Engineering (CSE), recently saw a substantial number of internal systems and services being decommissioned, meaning there is no longer a single point of entry to many core school functions.

Chapter 2 discusses the background of this problem in more detail, explaining the motivation for a new school management system. Chapter 3 defines the scope and requirements of this project, then provides an overview of the system architecture and design decisions. Chapter 4 analyses and evaluates the implemented system. Finally, Chapter 5 draws up conclusions and examines possible directions for future work.

Chapter 2

Background

2.1 Knowledge Management

At a high level, management is concerned with improving the productivity and efficiency of work to ensure that resources are optimally allocated or used [1]. With the ever-increasing amount of information available today, it is especially crucial for organisations to make relevant knowledge easily accessible and useful if they wish to maintain a competitive advantage. Platforms that identify, capture, organise and disseminate applicable knowledge in a systematic manner can increase shared understanding to help drive processes and innovation, thus producing benefits for both an organisation and its customers [2]. Tools such as knowledge management systems also facilitate progress in these areas by enhancing decision-making, reducing miscommunication and minimising errors [3]. The effectiveness of these technologies is profoundly evident: numerous universities around the world have already integrated them into their strategic plans to boost performance amidst rising competition in the higher education sector [4].

2.2 School Management Systems

To standardise common administrative processes and maximise information handling capacity, some university faculties or schools offer self-service systems hosted online. These may be considered as a subset of knowledge management systems, and will be referred to as *school management systems* throughout this report. In the context of UNSW, such systems also contribute to one of the three key enablers of the 2025 strategy [5], namely, 'Operational Effectiveness and Sustainability'. By removing the administrative burden from schools and helping to make functions run smoothly, they directly fulfil the objective of improving operational excellence, allowing schools to focus on their core activities of research and teaching [6].

2.3 Current Situation

As part of this broader strategy, which aims to overhaul the delivery of professional services at UNSW, there have been attempts to 'centralise' functions within various schools by moving them to existing UNSW platforms and websites. Although not every school had its own custom management system to begin with, those that did were not provided with an adequate replacement. myCSE was one such example, formerly offered by the School of Computer Science and Engineering (CSE). Since it was decommissioned in 2017 along with a number of other internal systems and services (see Figure 2.1), there has no longer been a single point of entry to the various functions it used to deliver.

To provide some examples of the consequences of centralisation: viewing someone's staff profile now requires going to UNSW's Research Gateway; booking meeting rooms requires using Office 365; and thesis management has been relegated to Moodle. The services may have been consolidated but the processes are even more fragmented than before. Furthermore, in the absence of a school jobs board, UNSW Careers and Employment is now the predominant place to browse or post jobs, but opportunities there can easily become buried in the hundreds of new listings added daily [7].


Home › Help & resources › Decommissioning of CSE Systems

Decommissioning of CSE Systems

Objective

The university is undergoing a major change process at the moment, which is resulting in the centralisation of a number of core School functions. As part of this process, CSE has not been able to replace the associated positions dedicated to the upkeep and development of these systems. As as result, internal systems must be decoupled from the current operational activities of the school and be switched off given there are no longer resources dedicated to this function within the School.

Scope

- MyCSE and iCsE have been decommissioned.

Decommissioned Services

Below is the list of services which have now been decommissioned.

Figure 2.1: Decommissioning of CSE systems

2.4 Analysis of Existing Systems

2.4.1 Research Findings

Despite lacking a school management system, many schools at UNSW continue to operate public-facing websites. From these, we can infer what functionality they would want out of such a system if they had one, as well as how they generally want to be portrayed on the web. To determine the kind of information presented online and how it is structured, the websites of 15 schools from a range of faculties were analysed: UNSW Business School; UNSW Law; UNSW Art & Design; School of Aviation; School of Chemistry; School of Physics; School of Computer Science and Engineering; School of Electrical Engineering and Telecommunications; School of Mechanical and Manufacturing Engineering; School of Mineral and Energy Resources Engineering; School of Photovoltaic and Renewable Energy Engineering; School of Chemical Engineering; School of Civil and Environmental Engineering; and Graduate School of Biomedical Engineering. A summary of the main findings is provided below.

Figure 2.2: Default profile page for a UNSW Art & Design staff member

People

Most schools maintain individual profile pages with detailed information about their staff members. If certain data is missing, the fallback approach is usually to display a default page containing just their contact details. The Faculty of Art & Design takes this approach, as depicted in Figure 2.2.

In addition, most schools also provide a list of staff for people to browse through if they are not looking for a specific staff member. Several schools from the Faculty of Engineering currently use the same template to display their staff directories. The title is inconsistent – staff members might be listed under any one of 'Staff directory', 'Staff', 'All staff', 'Our staff', 'Staff contacts', 'Find a staff member' or 'Our people', so it can be confusing to locate this function to begin with – but the interface is largely standardised, at least between these schools. A comparison is drawn in Figure 2.3.

Instead of maintaining a list, some schools, such as CSE, only link to UNSW's generic staff search page which encompasses everyone working at the university. Although it works, this does not add value compared to simply obtaining results from an internet search engine.

Seminars

Most schools, if not all, hold seminars on a regular basis for academics and research students to present and discuss topics of interest. There are several approaches to how these meetings are advertised, varying in the amount of detail given and the ability to drill down using specific search criteria. Some schools simply treat seminars the same

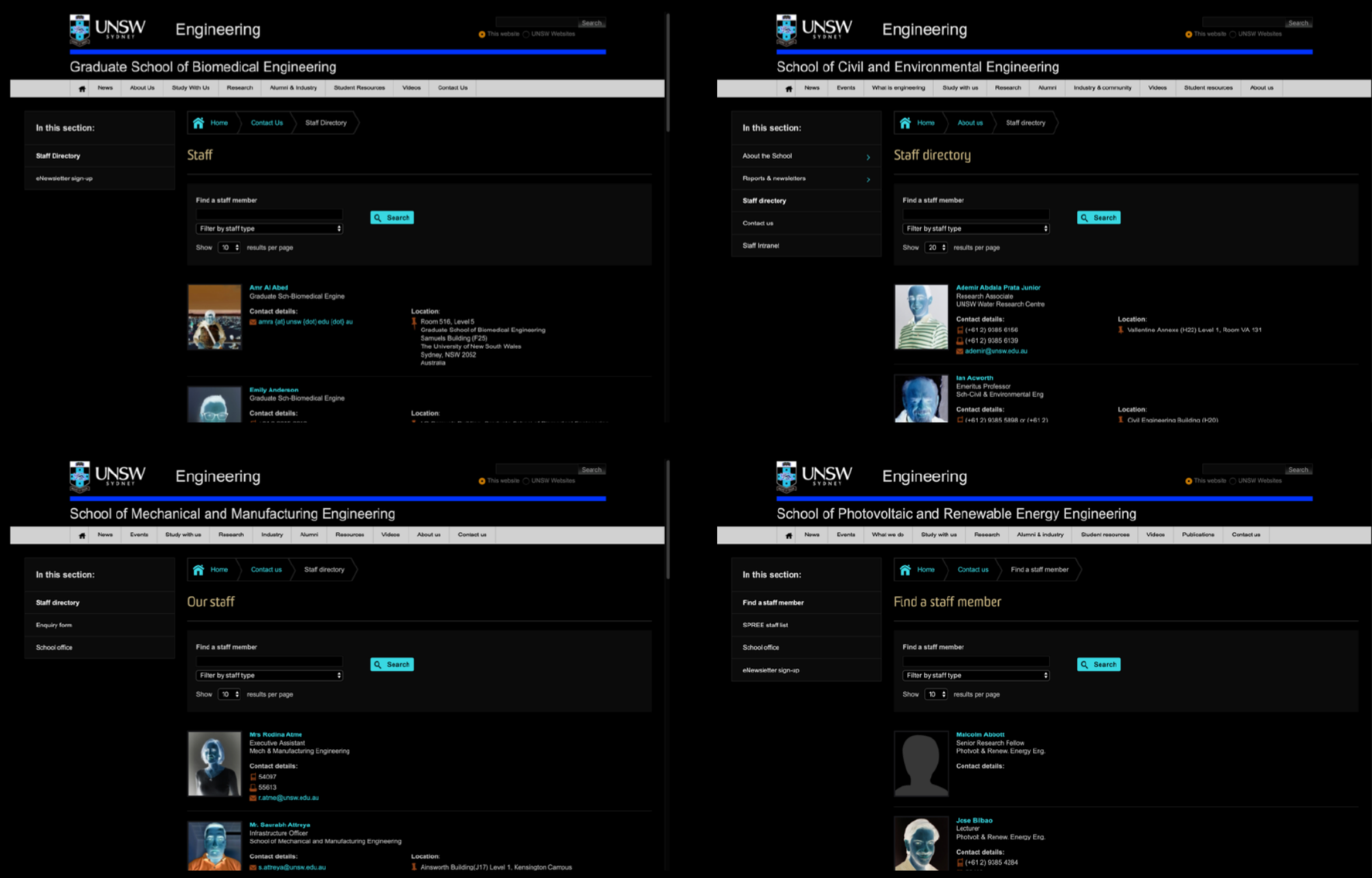

Figure 2.3: Comparison of staff directories

as other events and do not list them under a separate category. Others, such as CSE, proactively inform interested parties of upcoming seminars via email, but only those who have signed up to its mailing list will be notified. Still others, such as the School of Biotechnology and Biomolecular Sciences (BABS), have dedicated sections for both upcoming and past seminars on their website; all event details for the latter section remain archived and available online as shown in Figure 2.4. These pages often provide date filters as well to allow interested parties to further narrow their search.

Courses

As with seminars, there is no single prevailing approach among schools to managing course information, but it is common to see a list of courses and course offerings that are specific to a school. To enhance readability, these may also be separated by year of study, with the example given in Figure 2.5 from the School of Physics. While some provide sample program (degree) outlines, most schools only link to the UNSW Handbook and do not build on that information themselves. Moreover, it is noteworthy that there have been attempts to experiment with more modern ways of showing content.

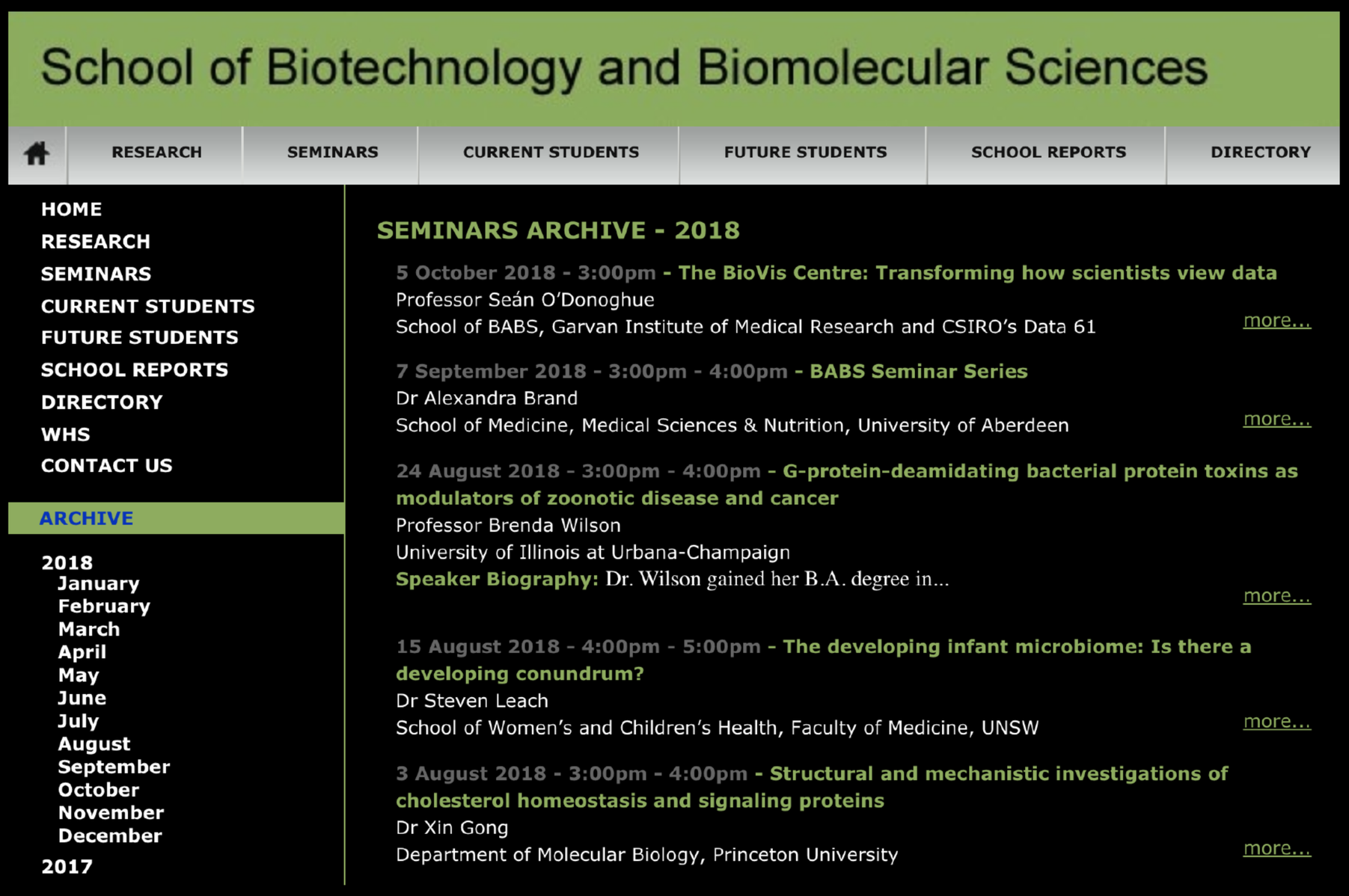

Figure 2.4: Archive of Biotechnology and Biomolecular Sciences seminars

For instance, course information shown on the UNSW Business School website consists of an online version of the course outline, though this can be optionally exported to a PDF file as well to conform to the traditional method of disseminating course outlines.

Now, it is well known that the UNSW Handbook already serves as a comprehensive guide to courses and is widely used by both students and staff members. However, although it is currently possible to browse by area of interest, faculty or subject area, there is no option to filter courses by school. This additional distinction should be made because individual schools usually offer courses in multiple subject areas – for example, CSE manages COMP, SENG and BINF course offerings [8]. Hence it makes sense to have a dedicated grouping of these courses too. The handbook also contains outdated information on occasion, but course staff may be deterred from requesting changes to be made due to the complexity of the process or merely a lack of familiarity with it. Considering that students treat the handbook as a definitive source of knowledge, this can potentially hinder them from making informed decisions on what courses to take.

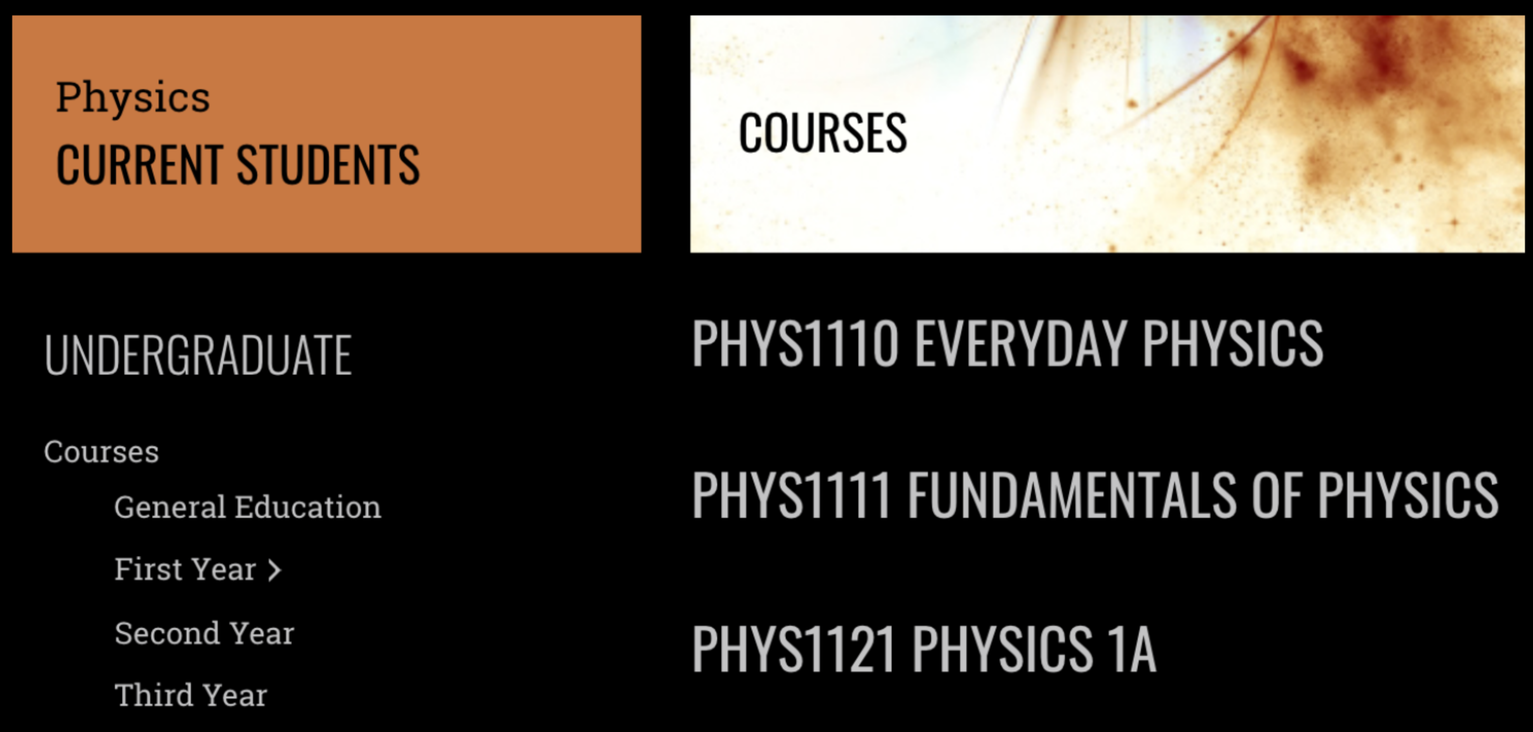

Figure 2.5: List of first year courses offered by the School of Physics

Job Listings

The majority of schools do not offer their own jobs board, preferring to link to UNSW Careers and Employment instead. Unfortunately, this requires students to apply search filters every time to narrow down results to their specific industry, as the sheer volume of jobs available on the site would make it impractical to manually sift through every listing and identify which are relevant. The centralised nature of this service also denies employers the ability to target opportunities at a more relevant audience, ultimately resulting in a lose-lose situation for both parties.

On the other end of the spectrum, there are a few schools that do advertise external employment opportunities. In early 2019, CSE partnered with Data61's *Ribit* to release a custom-made jobs platform for its students [9], shown in Figure 2.6. However, at the time of writing, this service is not well-known, externally operated and only applies to a single school in the whole university. As with UNSW Careers and Employment, users must sign up and log in before they can post or view jobs on this platform. The UNSW Law intranet takes a similar approach, as shown in Figure 2.7, except students can enter using their university-provided zID and zPass instead of creating a new account. Others, such as UNSW Art & Design, display job listings openly on their website and supply online forms for anyone to submit job or volunteer opportunities, presumably having a moderator screen these offers before publishing them.

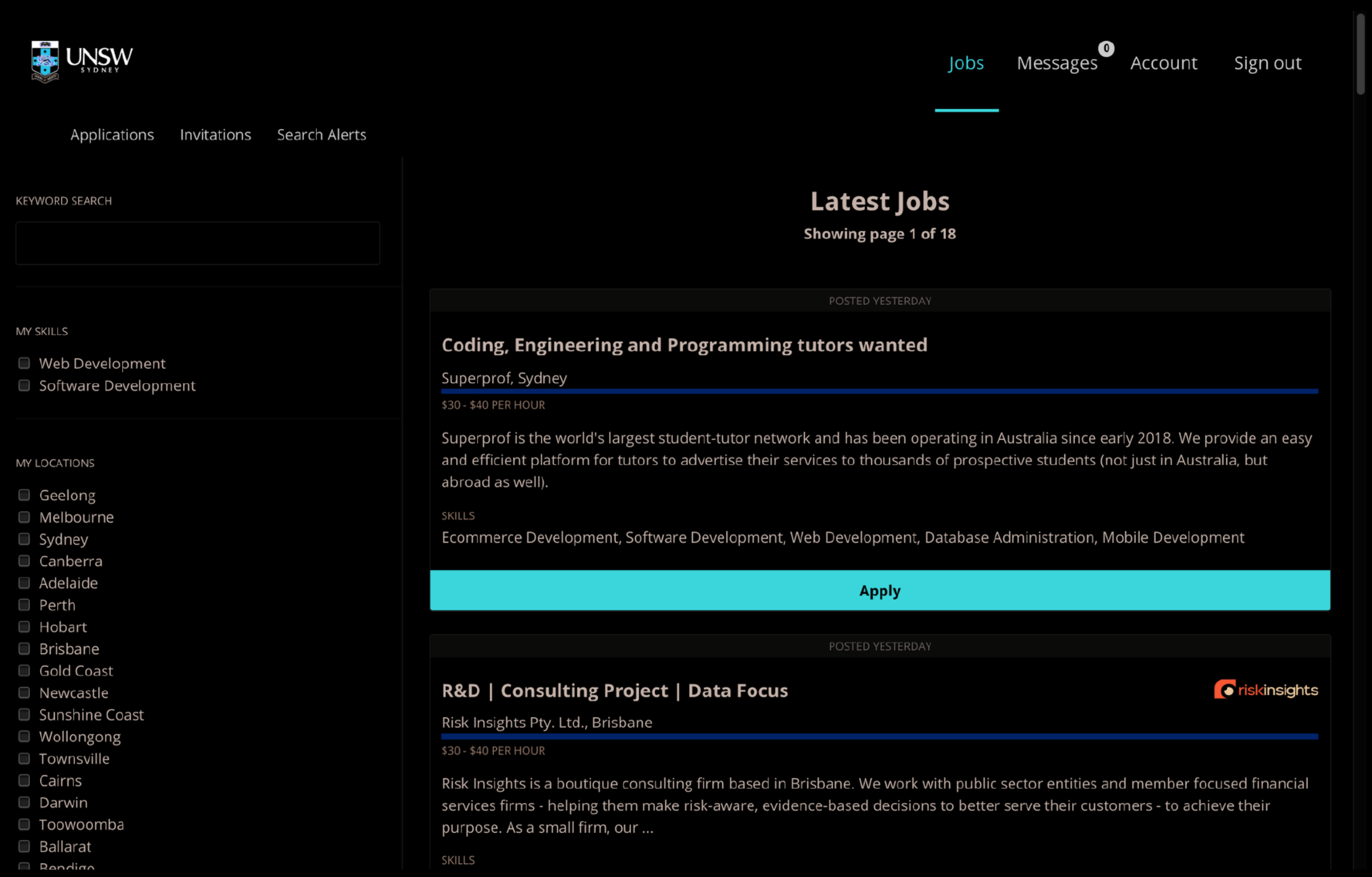

Figure 2.6: Ribit CSE jobs board

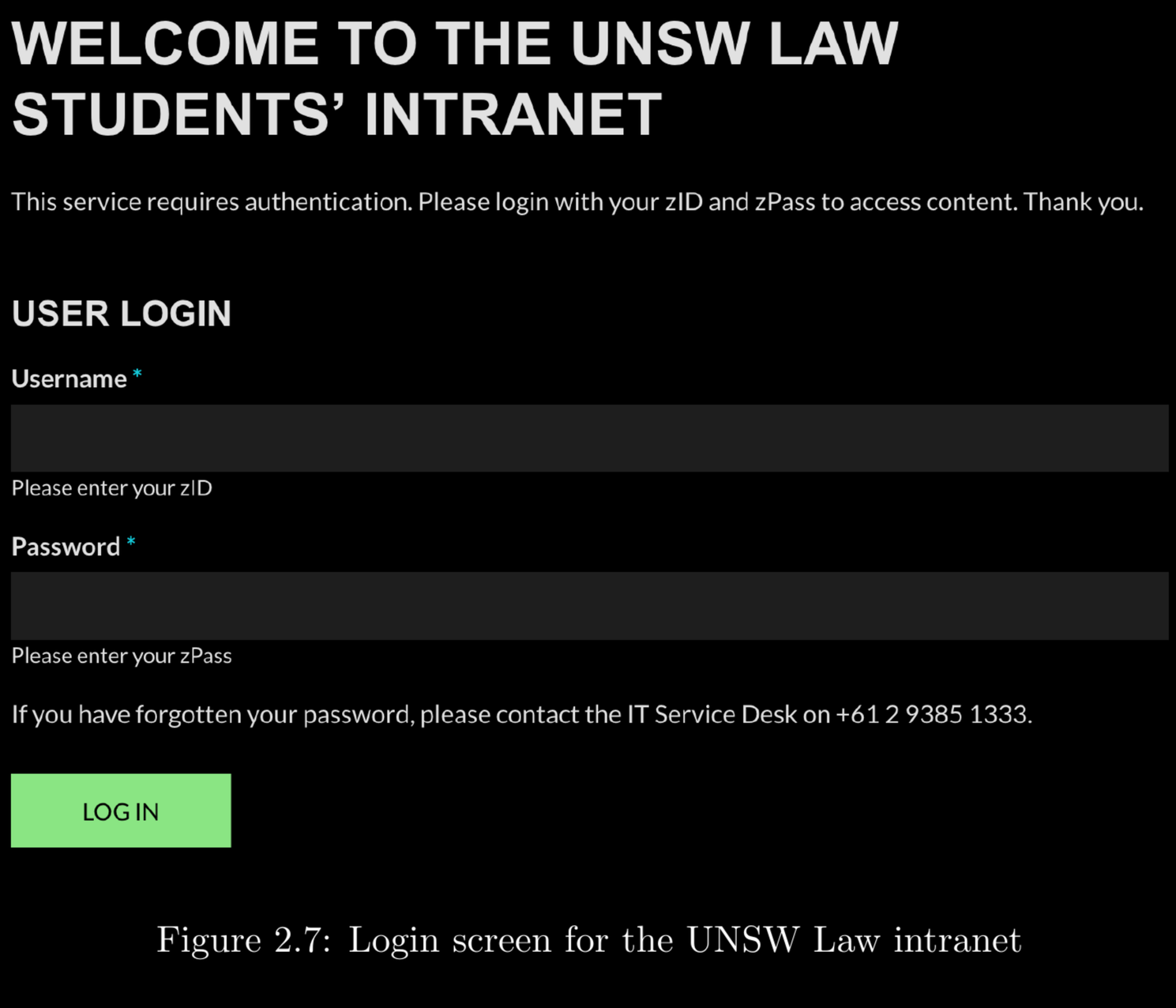

Figure 2.7: Login screen for the UNSW Law intranet

2.4.2 Additional Functions

The categories of information presented above do not comprise an exhaustive list as there are too many different services across all schools to cover comprehensively in this report. Rather, the most common functions have been identified and discussed; other functions to note are listed in Chapter 5.

2.5 Possible Solutions

There are several ways of addressing the aforementioned issues:

1. Use the new set of centralised tools and adjust existing processes to accommodate them. This is not ideal, especially for CSE where staff have already been inconvenienced with the lack of a replacement system for myCSE, and would not be a viable long-term solution without additional supplementary systems to provide missing functionality.

2. Create bespoke school management systems for each school. These would be tailored to individual schools' requirements, but would be difficult and costly to design, build, test and maintain. Such an approach also directly contradicts the university's endeavour to centralise these systems and would be unlikely to have the support of UNSW's senior management.

3. Create a generic, customisable and extensible school management system. As discussed before, there is a significant amount of overlap between the content and structure of the websites of different schools, so it would be logical to integrate similar information and functionality into a single shared platform. Doing so in this way would vastly simplify the development process as well.

It is important to recognise the difference between having centrally *hosted* but disparate systems (as is currently the case) and having a single centralised platform to facilitate

operations. Option (3) leans towards the latter, striking a balance with shared functionality but still maintaining adequate separation of concerns between schools. It is the most effective way of mitigating the issues raised, so it was the method chosen for this project. The developed system shall be termed *mySchool* throughout this report; an overview of the implementation is provided in the following chapter.

2.6 User Research

2.6.1 Survey

Prior to commencing development work, staff and students from various schools and faculties were surveyed to further explore the problem space. Ethics approval for negligible risk human research was formally granted by UNSW for this survey. In total, 88 responses were collected, uncovering insights on how people performed certain tasks and their thoughts on the proposed *mySchool* system.

Question 1 distinguishes between staff and students; the majority of responses were from students.

```
Q1:  Are you a staff member or student?  If you belong to both
     categories, select your primary occupation at UNSW.
```

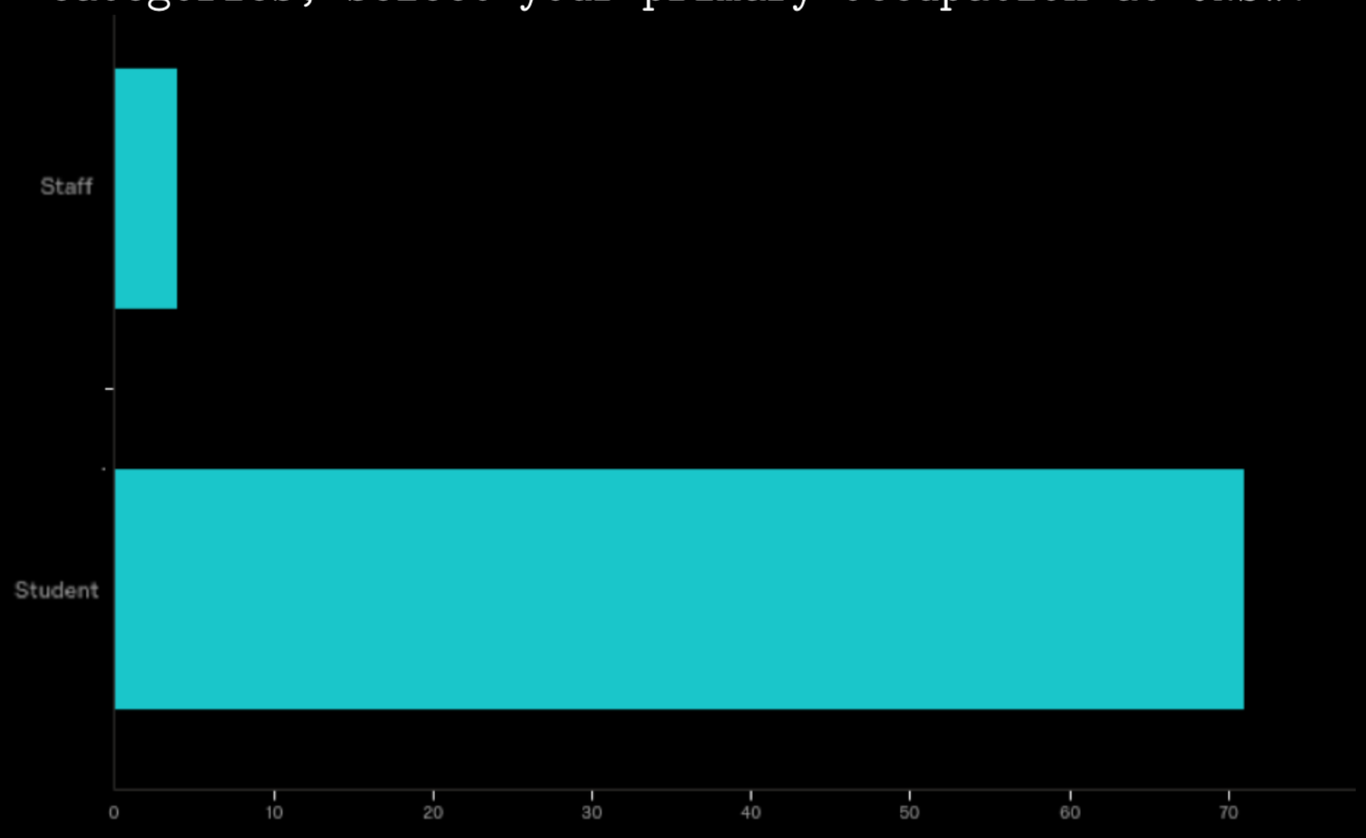

Figure 2.8: Survey question 1

Question 2 gives a breakdown by faculty.

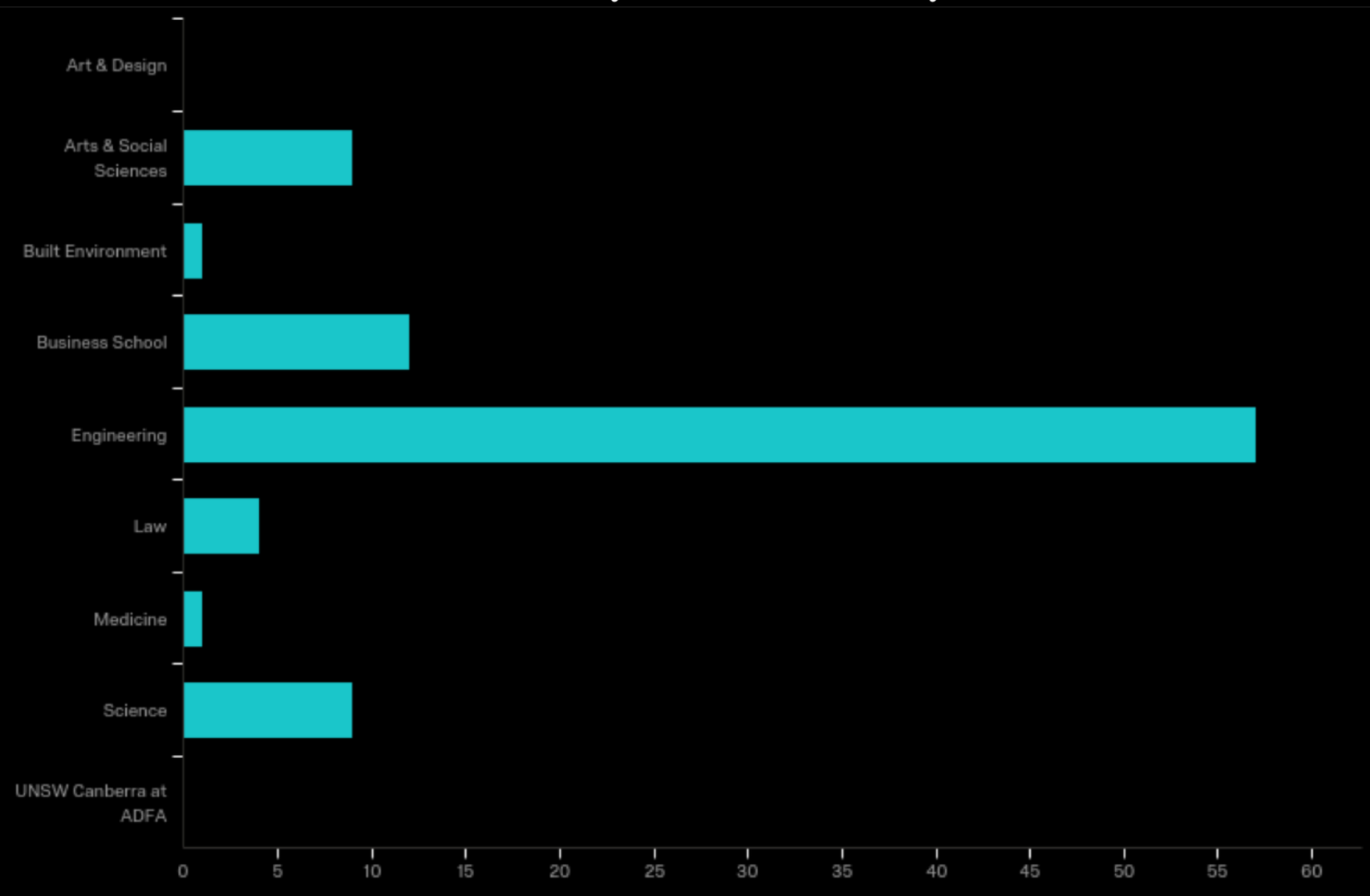

Figure 2.9: Survey question 2

Question 3 gives a further breakdown by school.

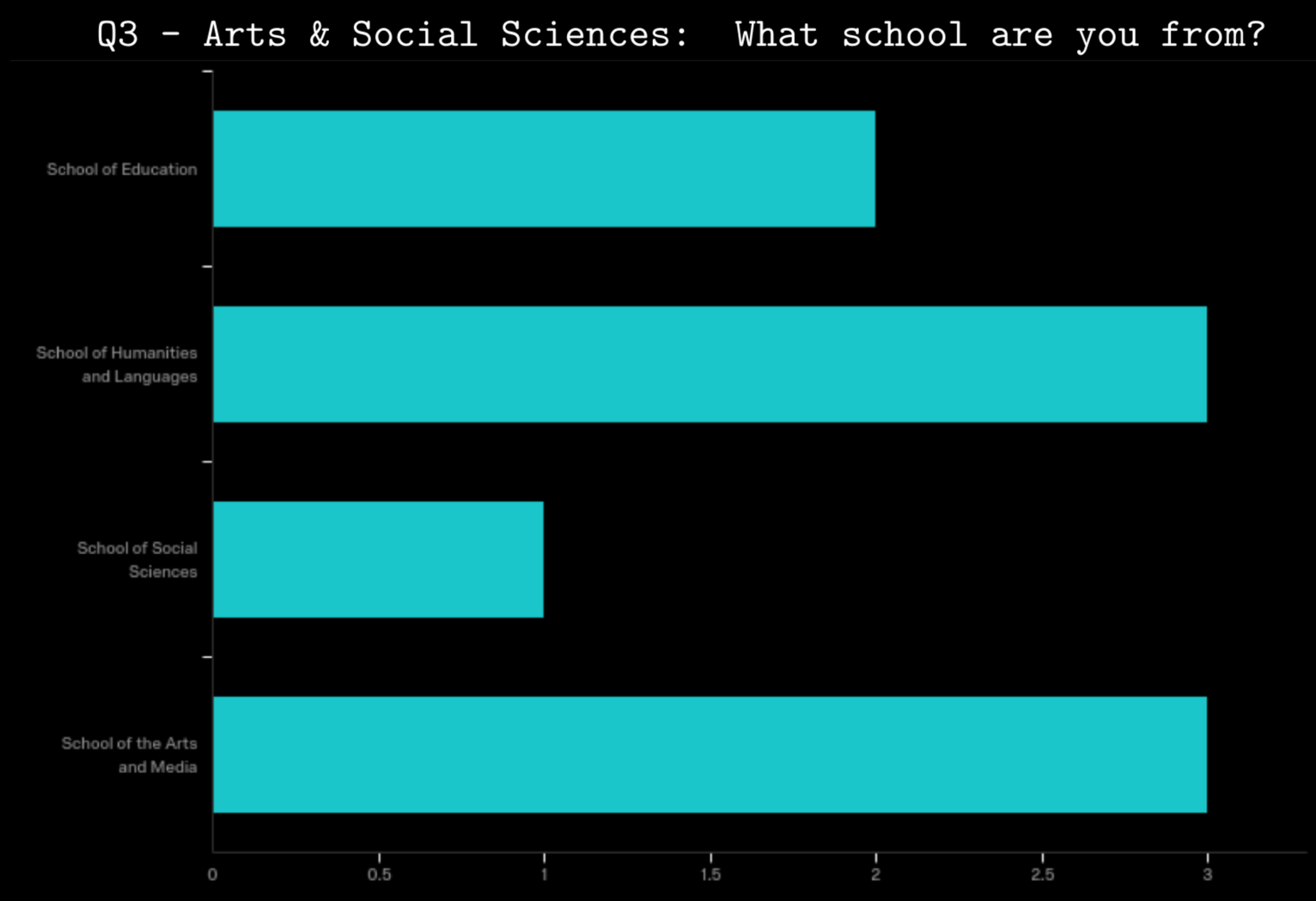

Figure 2.10: Survey question 3 - Arts & Social Sciences

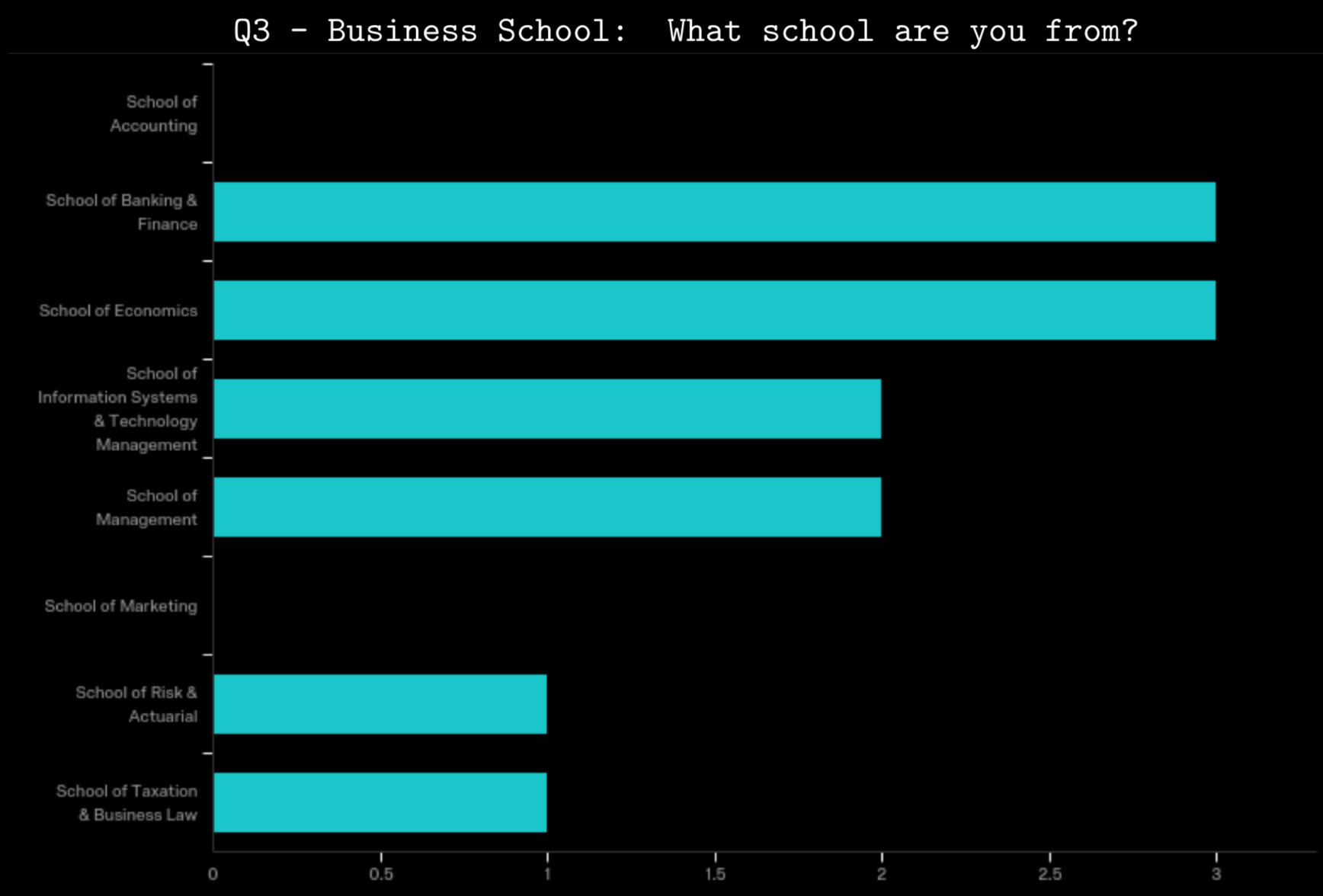

Figure 2.11: Survey question 3 - Business School

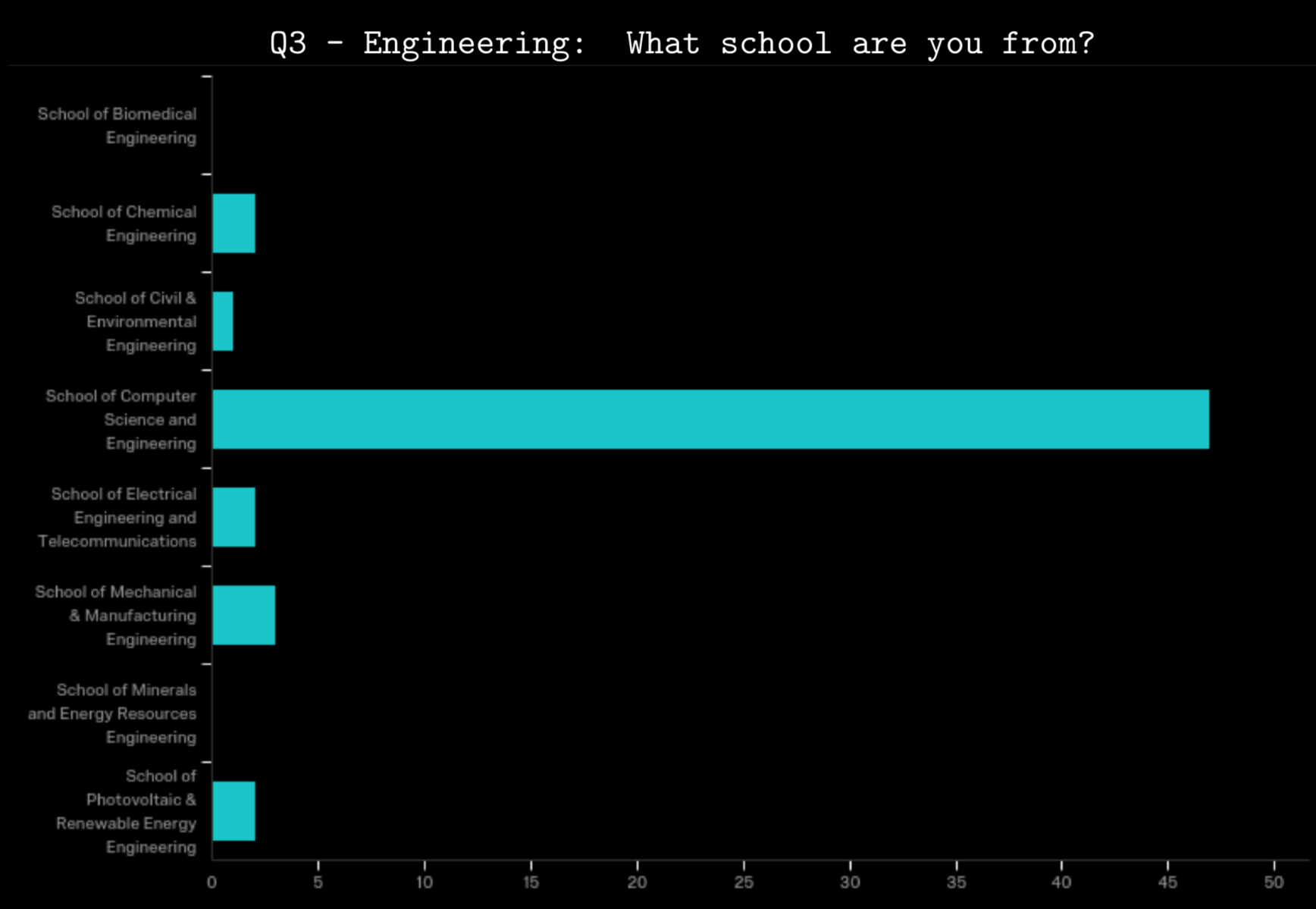

Figure 2.12: Survey question 3 - Engineering

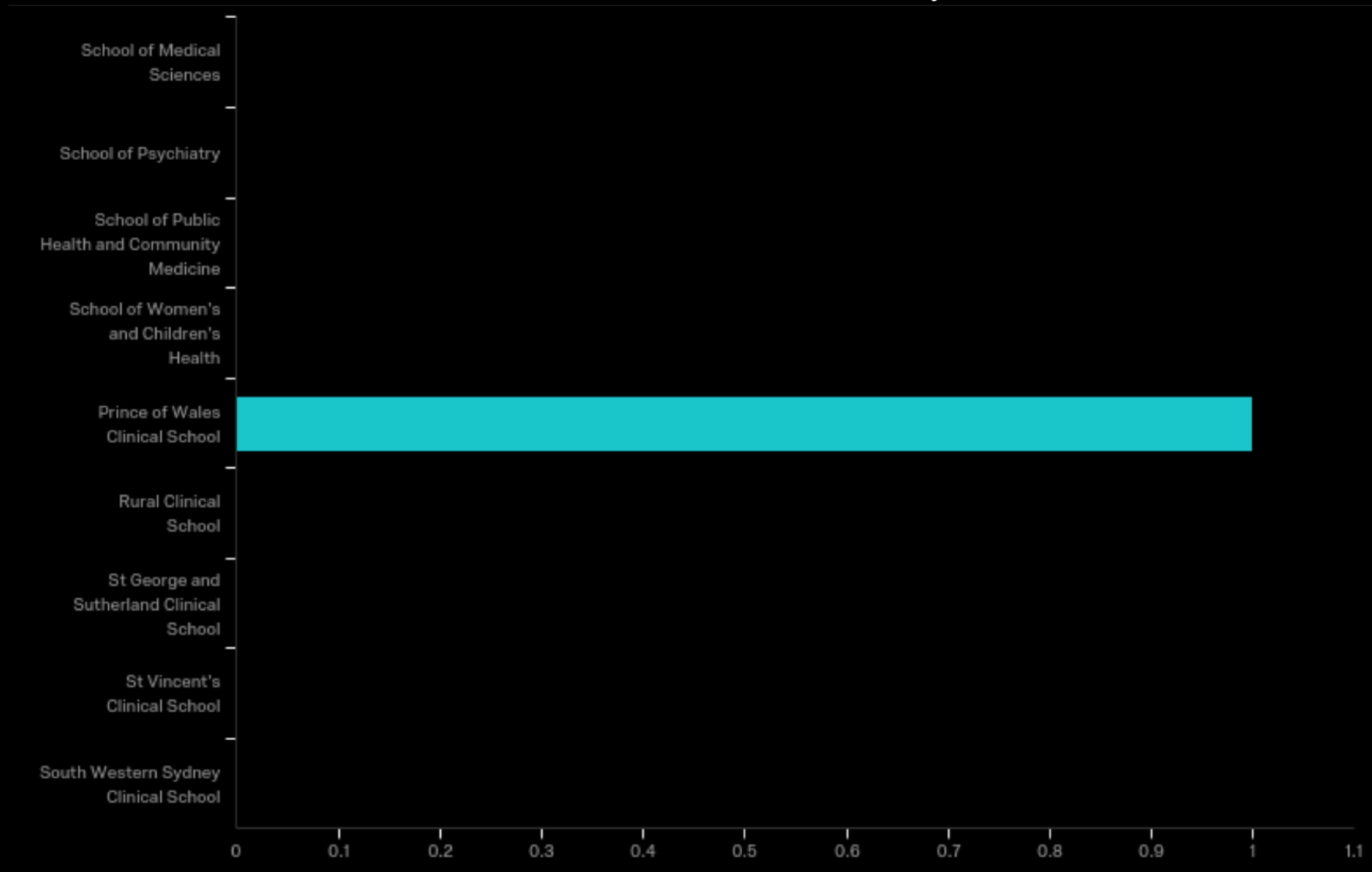

Figure 2.13: Survey question 3 - Medicine

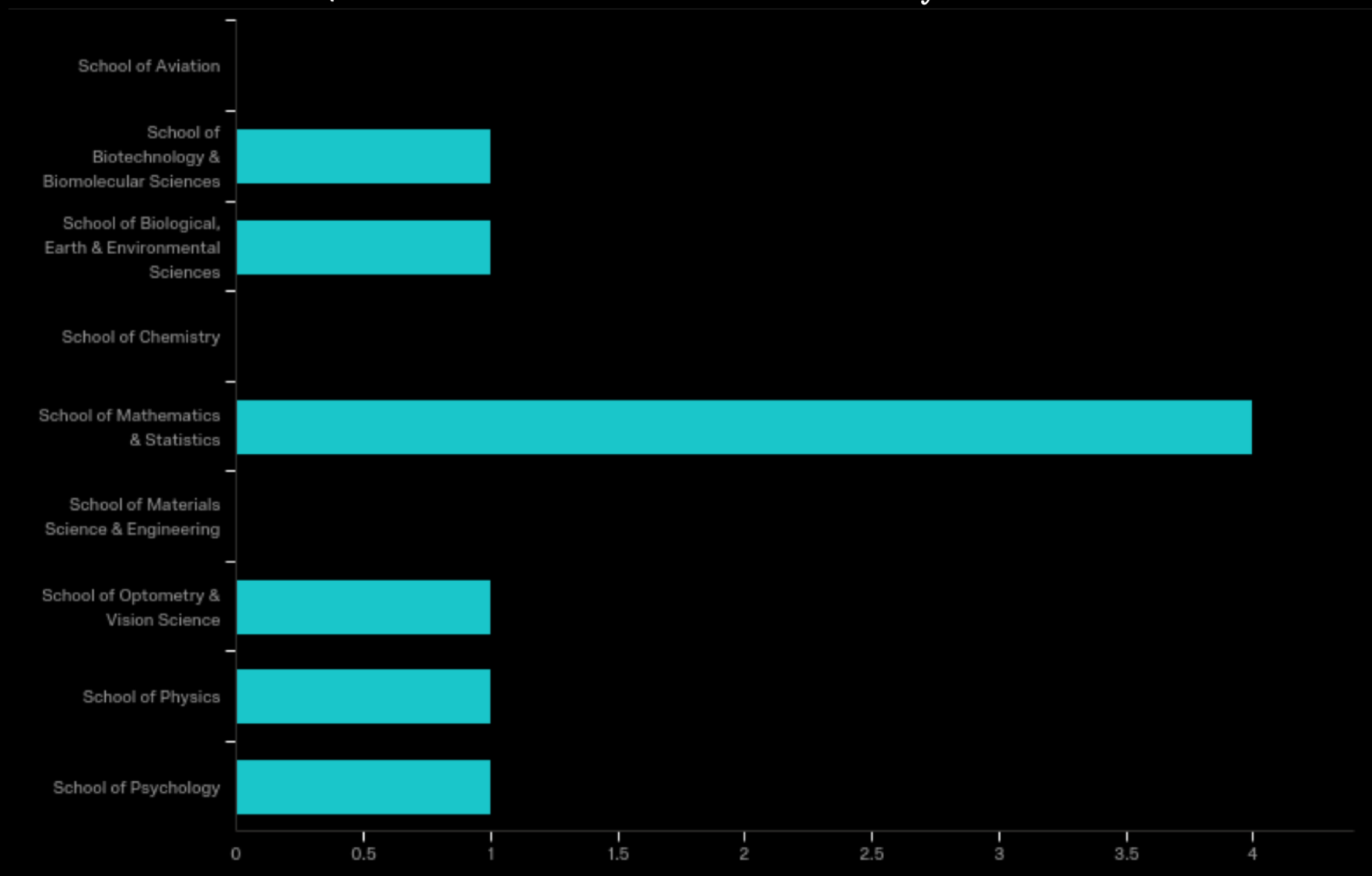

Figure 2.14: Survey question 3 - Science

Questions 4-7 examine how people currently perform some of the common tasks that would be covered by *mySchool*. As expected, the majority of respondents preferred to do them online, which supports the case for a web-based school management system.

Q4: How do you currently look up information on staff members? Select one or more.

Figure 2.15: Survey question 4

Q5: How do you currently find seminars to attend? (If you don't do this, how would you find seminars to attend?) Select one or more.

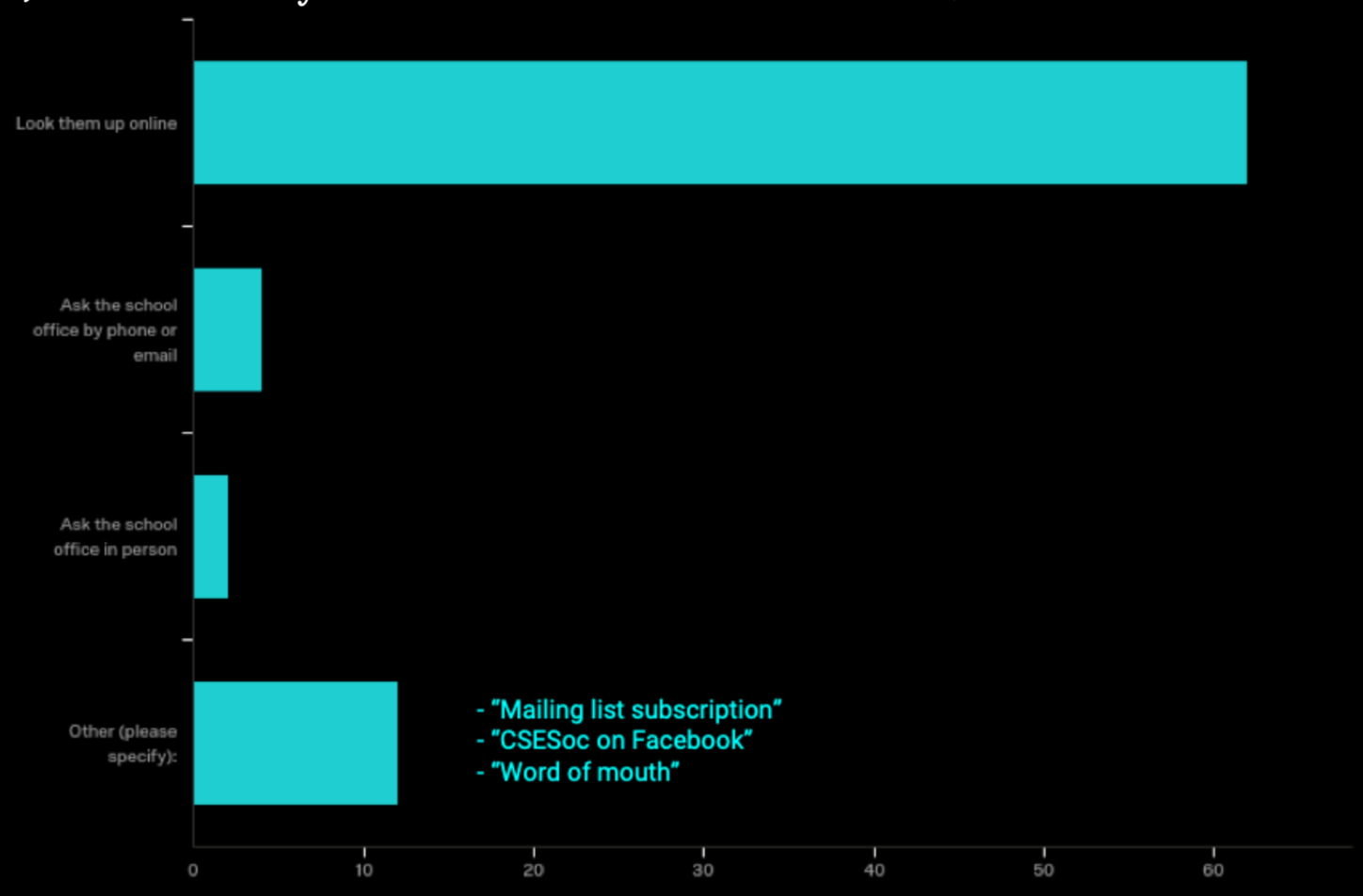

Figure 2.16: Survey question 5

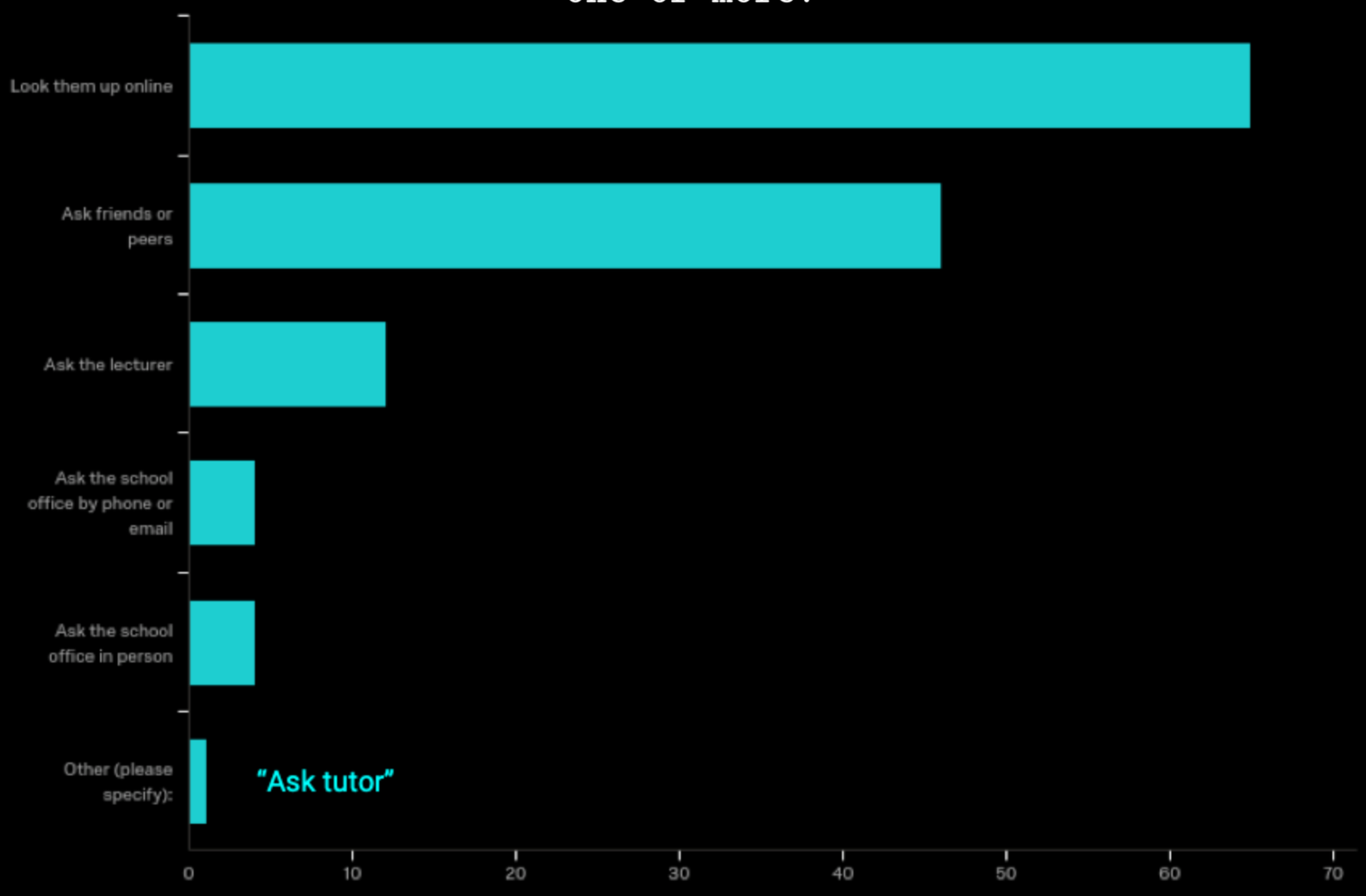

Figure 2.17: Survey question 6

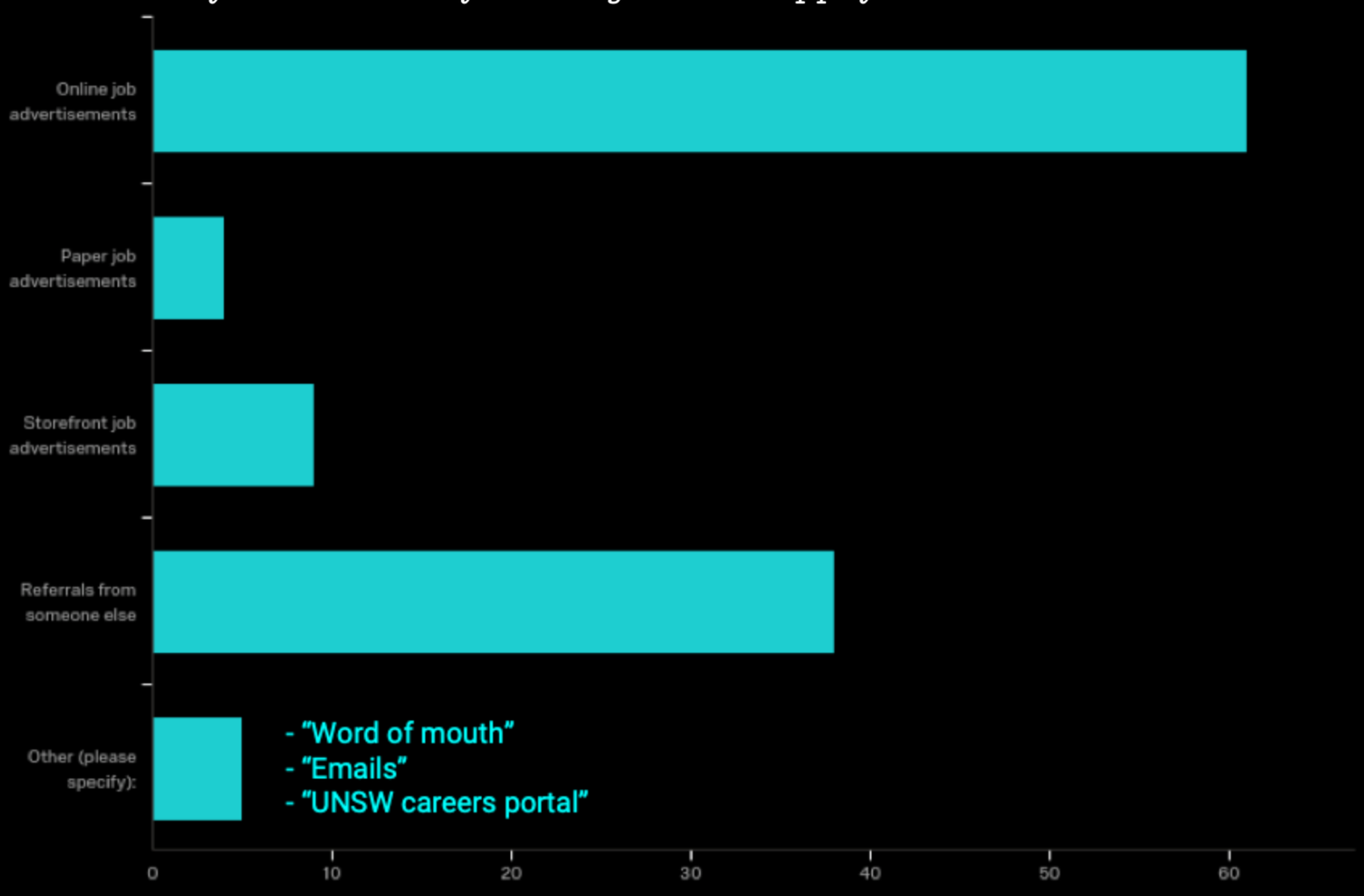

Figure 2.18: Survey question 7

Question 8 and its follow-up questions 8.1 and 8.1.1 look at existing systems that facilitate these tasks as well as people's usage of them.

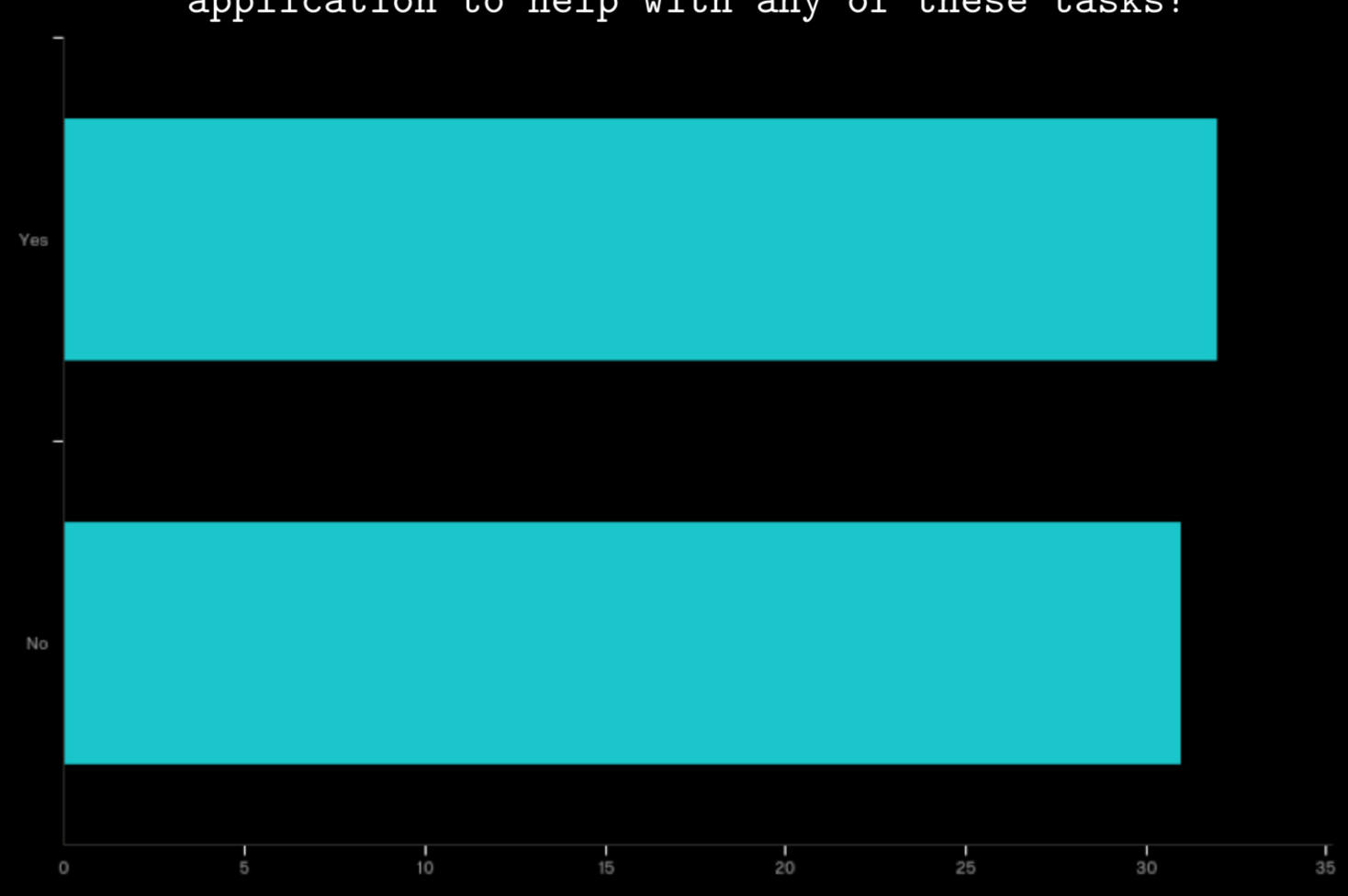

Figure 2.19: Survey question 8

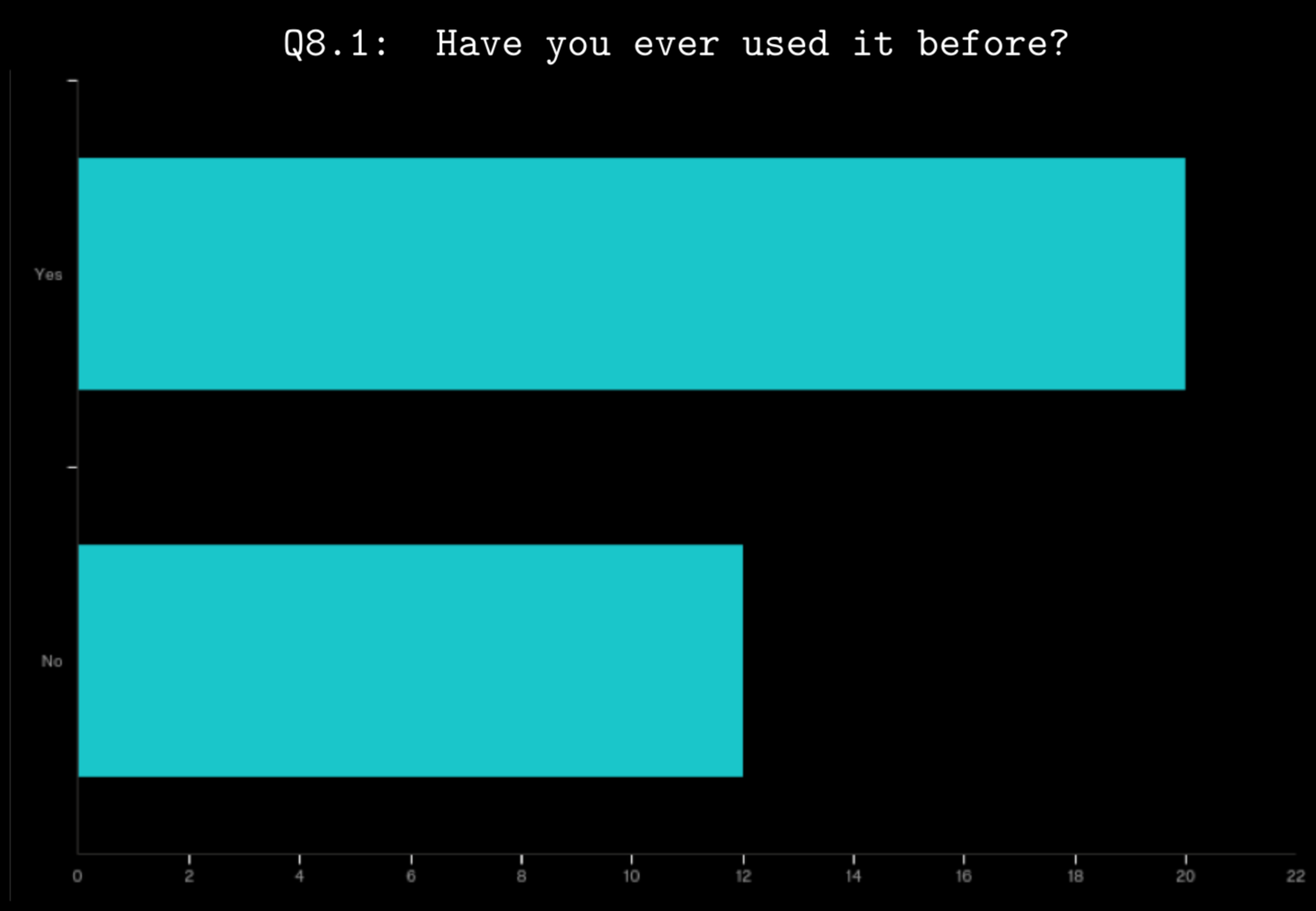

Figure 2.20: Survey question 8.1

Q8.1.1: Which feature is most important to you, and why?

The handbook, because it has all the information I need to structure my degree
Room numbers of staff location. Course outlines.
Info on subjects/courses because it's why I'm here
Staff member info
UX, usability from a visual standpoint to know where to find info
Completed and easy filtration for wanted information
Job search to secure a job after graduation
Relevance of information. I don't want to see jobs or info for arts students... those are irrelevant to me.
The ability to see past course outlines
As a first year, the information about courses and subjects were the most important feature since they greatly helped me with choosing what I want to actually do and study
Information for courses. Most info about the courses are listed online with details, so it's easy to look into multiple courses at the same time.

Table 2.1: Survey question 8.1.1

Part of the survey also included a usability testing component, the results of which are illustrated in the following subsection (2.6.2). Questions 9 and 9.1 of the survey asked about potential issues that users may have encountered with those tasks. Only one user found it difficult to navigate around the site; the other textual responses relating to unexpected behaviour may be attributed to inadequate communication regarding the limitations of the mockup.

Q9: Did you experience any difficulties when completing the series of tasks?

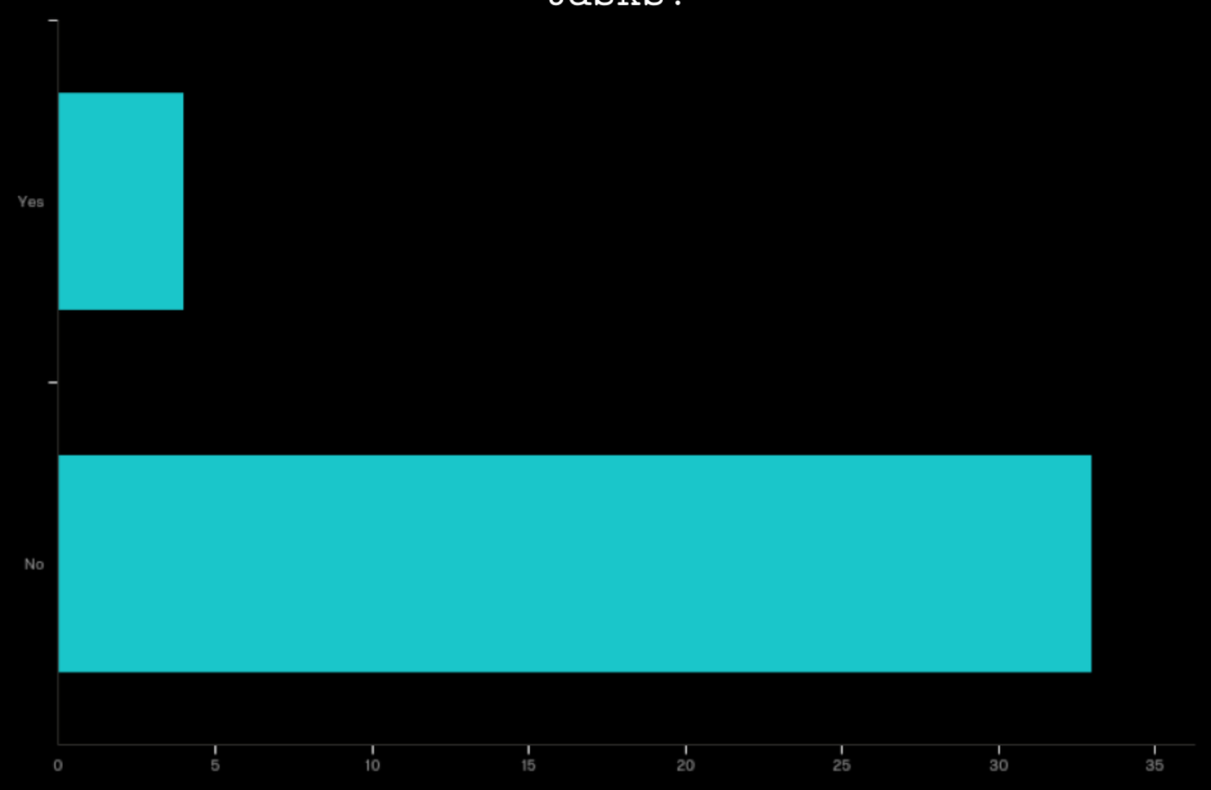

Figure 2.21: Survey question 9

Q9.1: If yes, what did you find difficult? Select zero or more.

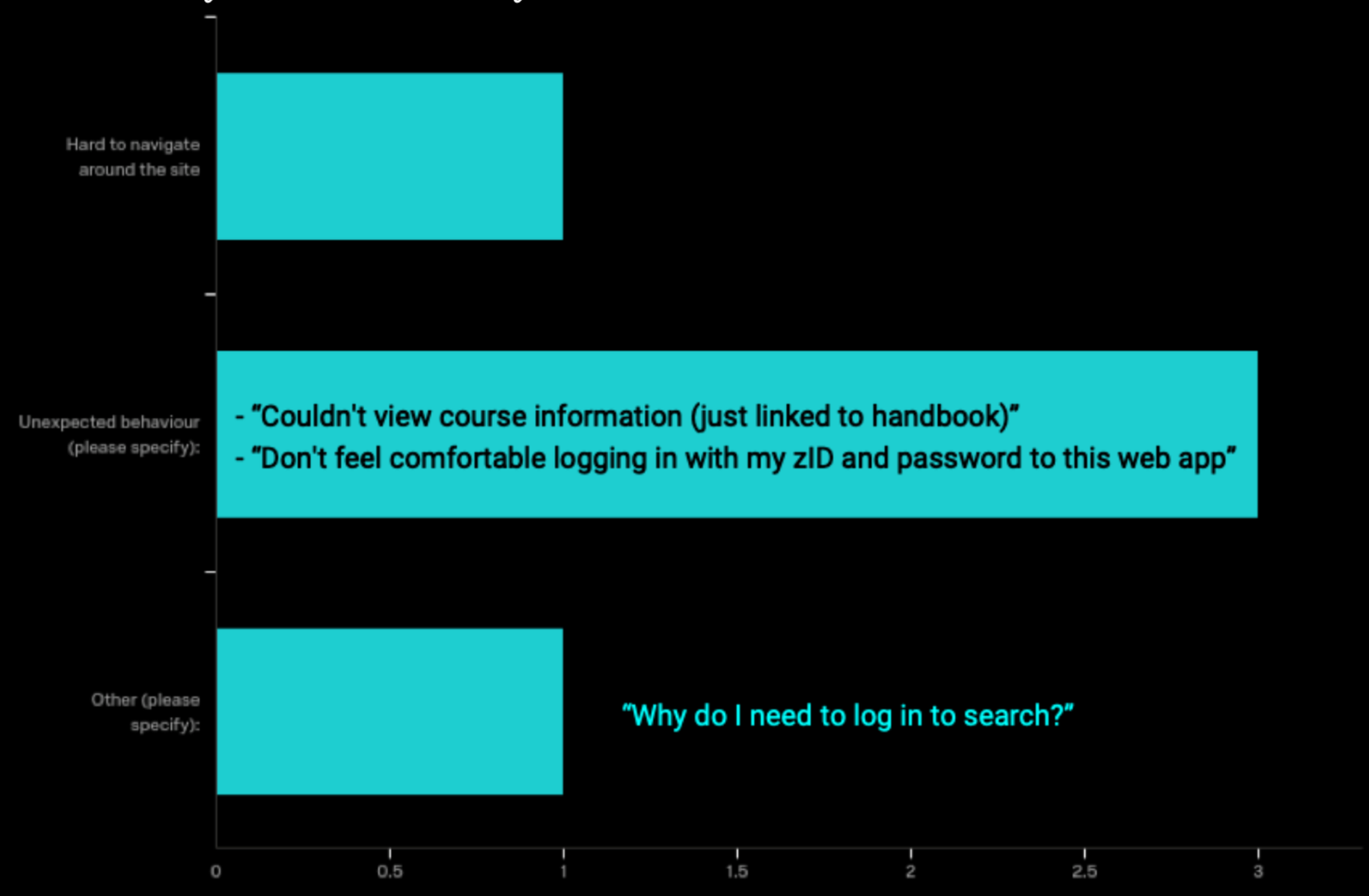

Figure 2.22: Survey question 9.1

Question 10 aimed to gauge the importance of features in *mySchool*. Surprisingly, looking up subjects/courses was the most popular.

Q10: Out of the existing features in the mySchool system, which one would you use the most?

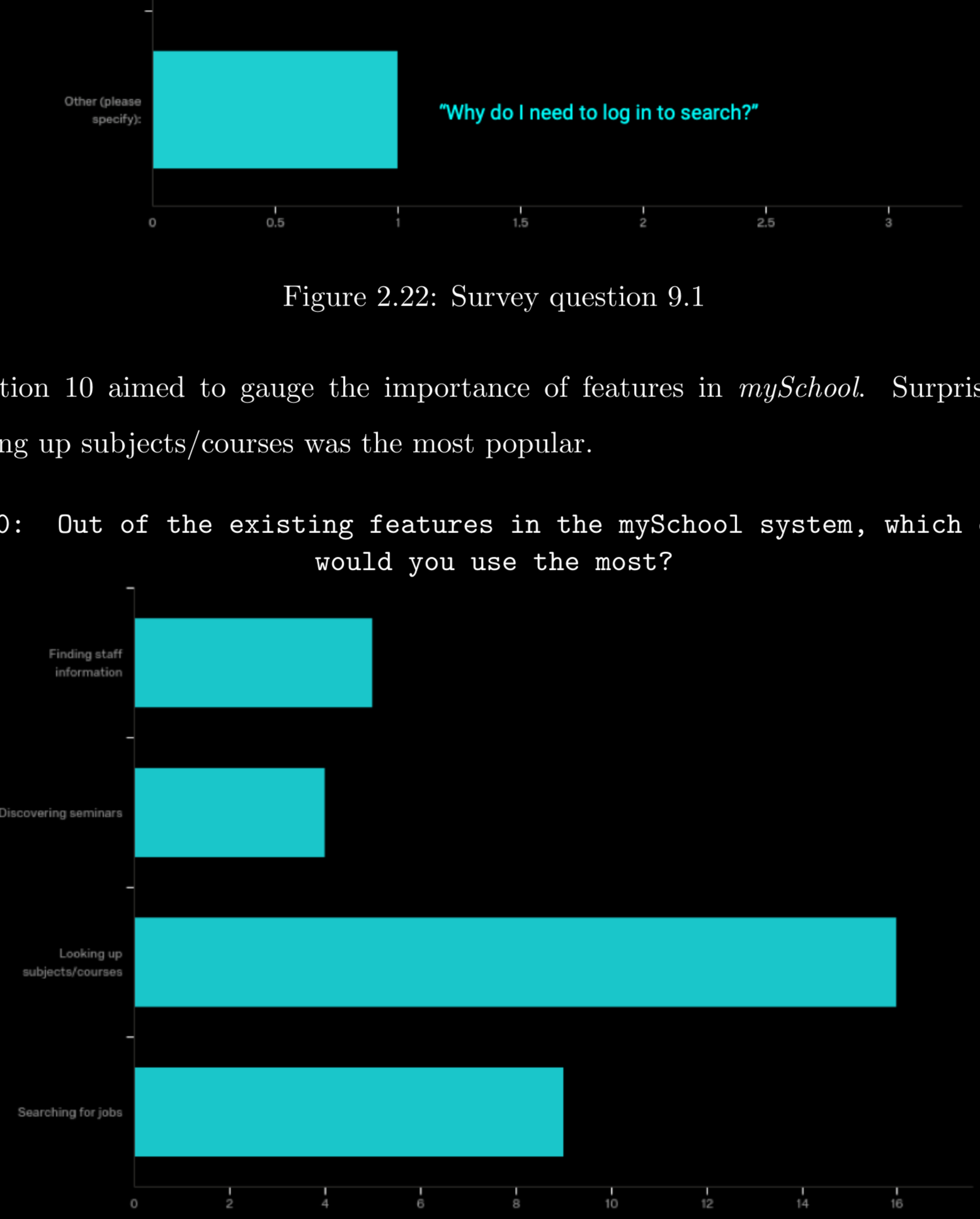

Figure 2.23: Survey question 10

Questions 11-13 requested suggestions for improvements to modules, features and the overall user experience. Several of these were already accounted for in the requirements for *mySchool*; the remainder may be considered as future work (refer to Chapter 5).

`Q11: Do you have any suggestions for how the current modules (People/Seminars/Courses/Jobs) could be improved?`

Ease of access
Probably have filters for people, i.e. related to *course*, works with *other person*, under school of *school*, research in field of *field*, etc., same for courses and jobs
Great, would love to see it go live
Could be more option before going straight to list (e.g. in job section, may have another pages that shows if you want part-time, full-time etc.
Advanced job/seminar/course search with categories – especially jobs
Enhanced search engines could eventually be added, which would be an improvement upon the current version. It would be particularly useful to organise search results via date – to discover jobs or seminars that are upcoming. You should probably consult with some product developers as to whether it would be worthwhile adding a 'recently added' flag to seminars/jobs that have recently popped up, or even to involve businesses and add sponsored items (perhaps an area that could be monetized). Also for all the modules it would also be useful to sort them by category, such as the stream of engineering, etc.
Didn't really pay attention but sorting/filtering jobs would be nice

Table 2.2: Survey question 11

`Q12: What is a feature that is currently missing in the mySchool system but would be important to have?`

Events the school is holding, maybe more social ones
Maybe a link with Arc / societies associated with the faculties and extracurriculars
Maybe something to take the users to different common websites in their daily usage. Like myUNSW, Moodle, Piazza, etc.
Profile
Thesis management stuff was good on myCSE, as was having a marketplace, and a private tutor database. Of those, the private tutor database and thesis management are the most important/useful.
Peers?
Maybe a popular links page? With Moodle, myUNSW, Library etc.
Volunteering page both in UNSW and outside
Search engine filters, item dates (recently added, upcoming, etc.)

Table 2.3: Survey question 12

Q13: Do you have any suggestions for how the overall user experience could be improved?
Possibly move search bar below the modules tabs when in a module page
None, seems easy enough
Make it more personalised, i.e. keep track of courses, previous interests, etc.
No it was very clean
A manual/quick page to help other users see how to get what they want.
Consult more UX designers on making the page as pretty as can be?
No I like the design

Table 2.4: Survey question 13

2.6.2 Usability Testing

As mentioned earlier, part of the survey involved usability testing using an external tool called Maze [10]. Only a subset of respondents completed this section, but those that did were asked to perform 6 short tasks (deemed *missions*) on a series of mockup screens linked together with clickable 'hotspots', similar to the one in Figure 2.24.

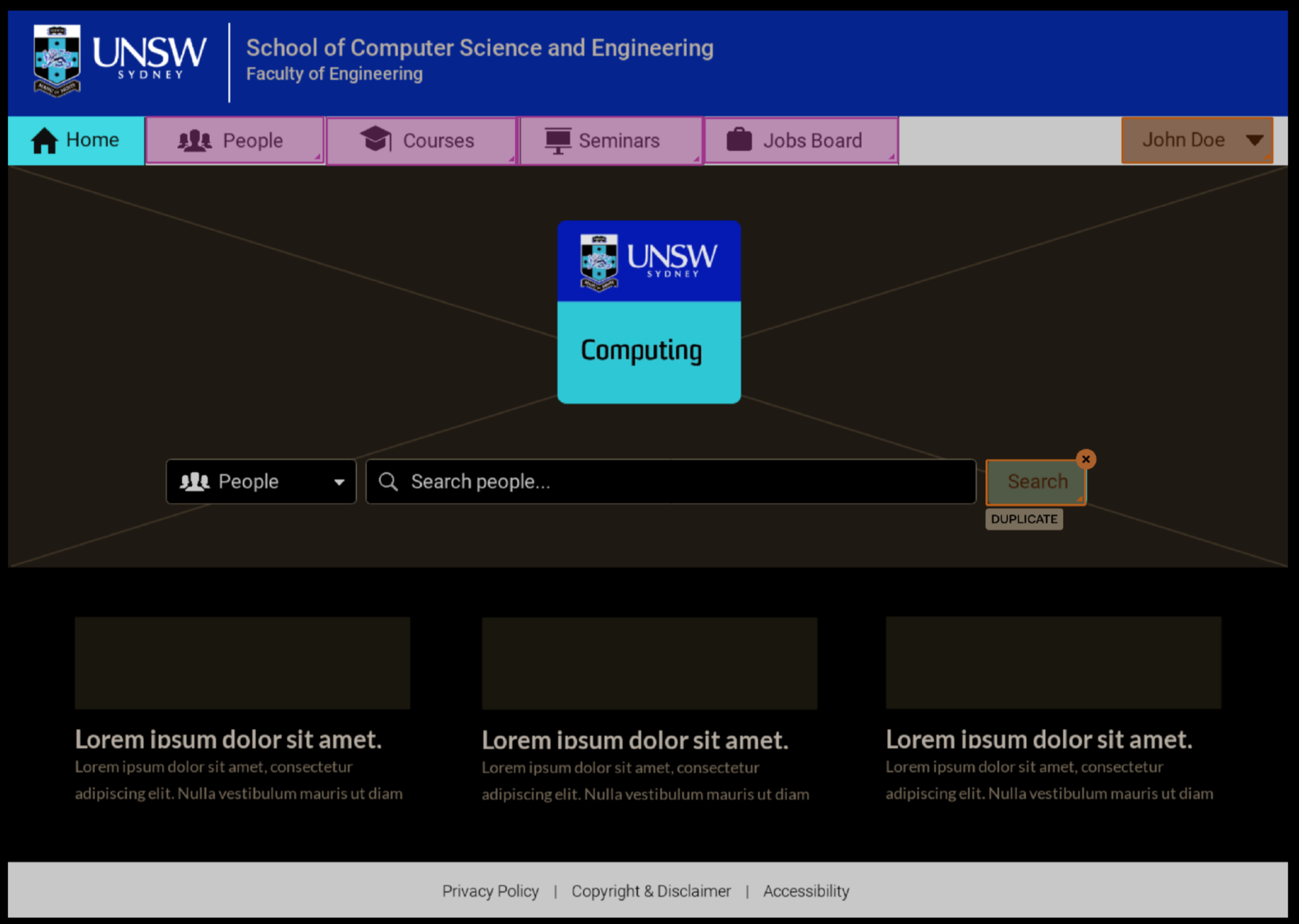

Figure 2.24: Mockup with hotspots

There were 21 testers in total, generating the following results in Figure 2.25.

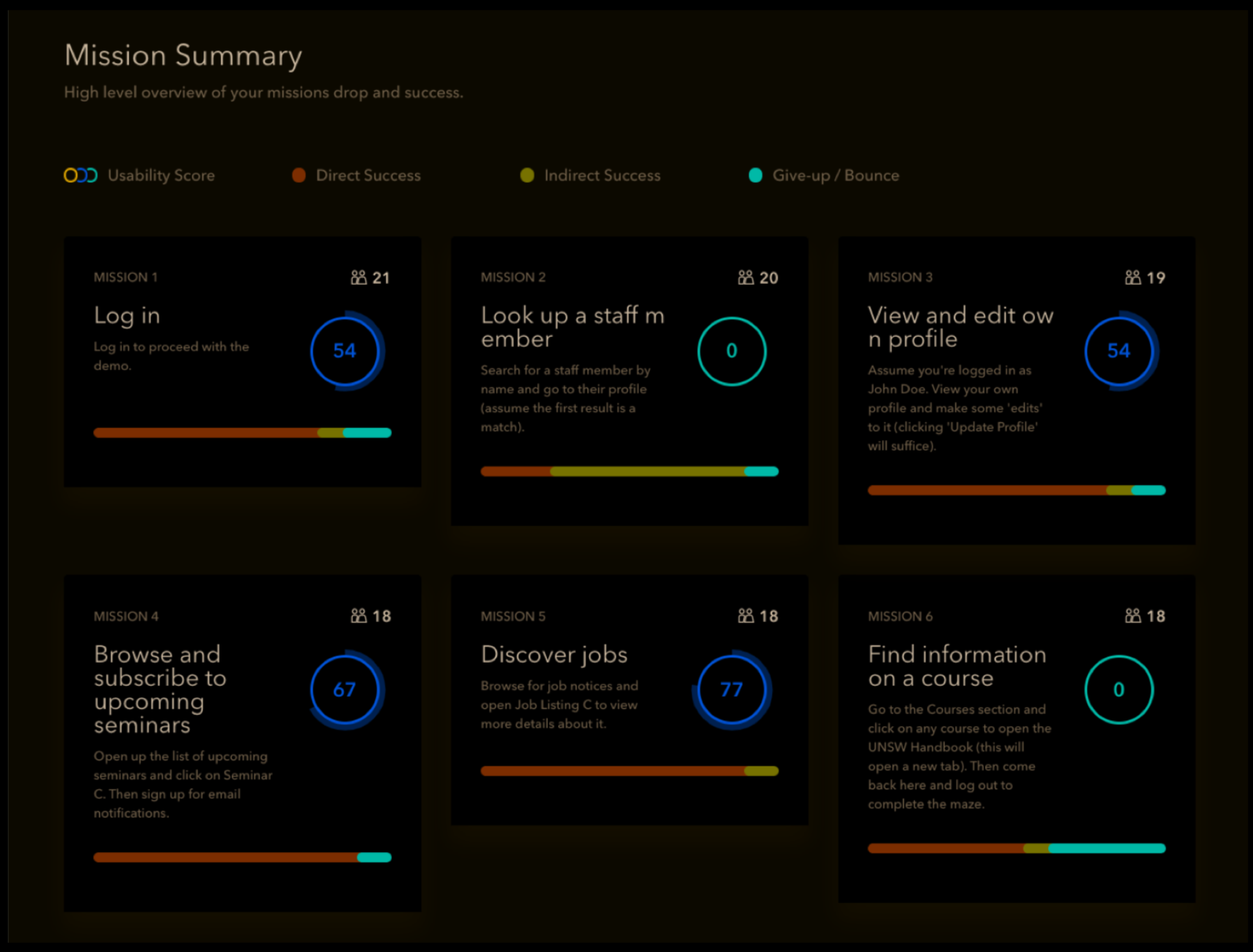

Figure 2.25: Usability testing results

In the legend, *direct success* is when users completed the mission via the expected path; *indirect success* is when users reached the final screen via an alternative path; and *give-up / bounce* is when users left before completing the mission [11].

Overall, 4 of the 6 missions had a majority *direct success* rate, indicating that users found these straightforward and intuitive to complete. Mission 2, which required looking up a staff member, had a high *indirect success* rate since there was more than one way to accomplish this task. Mission 6, which asked users to find information on a course, had a very high *give-up / bounce* rate, most likely due to confusion about the expected behaviour of the mockup, as suggested by the responses to questions 9 and 9.1 of the survey. Heatmaps illustrating the actual areas that users clicked on were also generated for each mission; Figure 2.26 shows the one for mission 2, while Figure 2.27 shows the one for mission 6. The remaining heatmaps can be found in Appendix B.

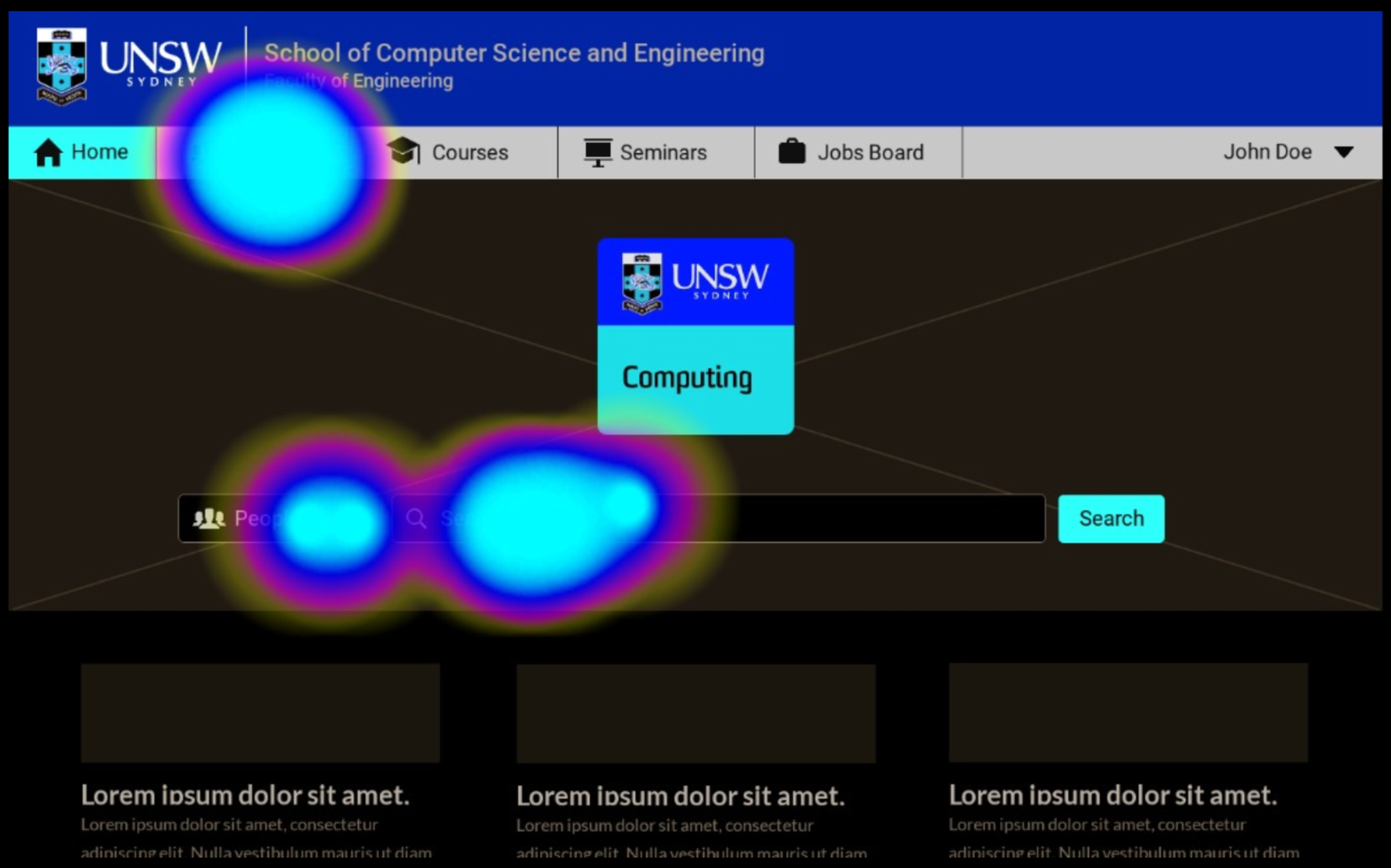

Figure 2.26: Heatmap for mission 2

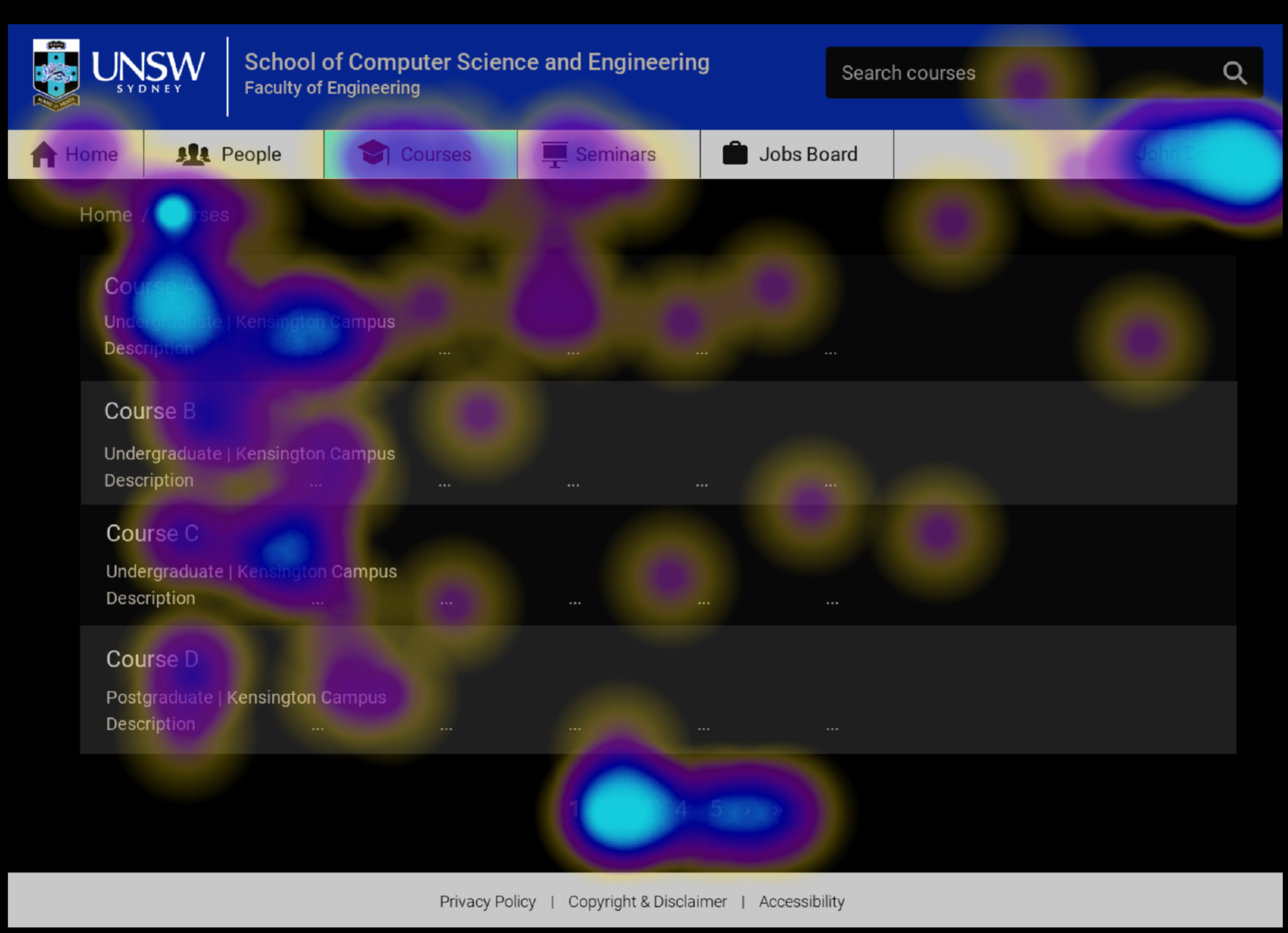

Figure 2.27: Heatmap for mission 6

Finally, some users left additional constructive comments at the end of the testing session, listed in Table 2.5.

`What did you think of this prototype?`
Very intuitive and easy to use, all the information was clearly laid out and quick to access, usually only a few clicks and I was able to find the information I needed
Looks good
It was easy to navigate
The prototype is simple and easy to navigate. Consider introducing a System Usability Score (SUS) survey at the end for useful feedback.
It was really good, loved the UX
Good!
Was pretty simple to use - the tasks were relatively clear (though I did do some useless clicking on features not yet added)
This is really cool. The prototype is actually easy to use and navigate (unlike the new handbook...)
Easy to use, however Edit Profile changes the UI to a different format. It wasn't a smooth change from viewing profile.
Overall good. Only issue I had was the search bar on the navbar is a bit confusing as changing tabs would change the outcome of the results. You would normally assume it would be used to search everything and not a specific category. Minor nitpick, edit profile to profile page layout should be consistent :P
Awesome! Well designed :)

Table 2.5: Maze prototype feedback

2.6.3 Summary of Findings

Preliminary findings from this user research were in generally favour of a web-based school management system to facilitate common administrative tasks. Feedback collected from users played a key role in shaping the requirements discussed in Chapter 3. In particular, the ease of navigation and intuitive UI/UX were all highly rated in the mockup.

Chapter 3

Solution and Implementation

3.1 Solution

mySchool is a greenfield project that integrates a subset of core school functions into a single web application that can be adopted and used by any school at UNSW. The main idea behind it is to enable staff and students to easily access these functions via a single point of entry without having to use tools scattered across different online platforms such as Office 365 or Moodle.

In terms of the main modules for this new system, it primarily handles the key areas covered by most schools' websites, namely: people management, seminar details, course information and job listings.

People Management

This is largely a directory of academic staff, postdoctoral researchers (postdocs) and PhD students. Each person has their own profile page with essential information such as contact details, research areas, publications, role(s) within school, etc. A rudimentary role management subsystem is also included, initially for the purpose of access control (determining who can view or change certain content), but with the potential to be used for workload calculation as well in future versions.

Seminar Details

This is a dedicated space to showcase past and upcoming seminars – in particular, information such as the topic, abstract, speaker, date/time and location. Interested parties, including those external to UNSW, may subscribe to receive email notifications.

Course Information

To avoid confusion surrounding course offerings, each school should have a 'single source of truth', which is what this module aims to provide. For the Head of School and Deputy Head of School, it also enables them to easily see course evaluation results.

Job Listings

Both students and employers would benefit from having a school-specific jobs board. Instead of competing with the entire UNSW job market, they can stand out more easily as jobs would be tailored to a limited set of degrees and industries.

3.2 Scope and Requirements

Naturally, an ideal school management system would not only encompass these 'common' functions, but also be capable of catering to all the different requirements of individual schools in terms of the services they provide to staff and students. However, given the time and resource constraints for this thesis, the initial version of *mySchool* is restricted to the four modules described above. This was to account for the effort expected to be spent designing, implementing, integrating, documenting and testing.

3.2.1 Functional Requirements

Even with a limited scope, there are too many functional requirements to list here. Instead, they were recorded as acceptance criteria for user stories on a project management tool called Trello [12], which will be further explained with examples provided under section 3.5.

3.2.2 Non-Functional Requirements

- Consistent user interface and user experience to make it easy for anyone to navigate regardless of their faculty or school.
- Good performance. Usability may be hindered if any component of the software stack is inefficient, which could cause frustration and possibly drive users away from using *mySchool* altogether.
- Secure handling of data. Although certain knowledge might be publicly available, sensitive information will also be contained in the *mySchool* database and must be properly secured via authentication and authorisation mechanisms [13].
- Well-documented and extensible. Since other developers may have to maintain and/or build on top of existing modules in the initial offering of *mySchool*, a comprehensive set of documentation would help with their understanding of the system architecture and design.
- Well-tested. As code is inevitably maintained, refactored or changed, a combination of manual and automated testing would ensure that the system behaves as expected and that no previous functionality has been broken or software bugs inadvertently introduced [14].

3.3 System Architecture

The following frameworks and systems were selected as part of the technology stack for *mySchool*. Figure 3.1 depicts how they relate to each other.

Client-side

Angular [15] is a web application platform developed by Google. It is based on reusable components (meaning code is more modular and efficient [16]), protects against common web application vulnerabilities out of the box [17], and has excellent performance as well, courtesy of a highly optimised code generation mechanism [18]. That last point

is also one of the reasons for writing *mySchool* as a single page app (SPA), since it asynchronously renders content that has been changed within the same page [19], whereas a traditional website re-renders the entire page and all elements on it whenever clicking on a link or navigating elsewhere. A direct benefit of this client-side rendering approach is that the app feels more responsive (as there is no flash between page loads) and is more performant too (as the lack of a full page refresh means fewer HTTP requests are required) [20].

Server-side

Django [21] is a popular Python web framework developed by the Django Software Foundation, well-suited to working with relational databases due to its powerful Object-Relational Mapper (ORM) [22]. It also has built-in security against common exploits such as Cross-Site Scripting (XSS), Cross- Site Request Forgery (CSRF), SQL injection and clickjacking [23], and comes with a graphical site administration interface [24] as well. There are a number of large companies currently using Django including Instagram, Mozilla and Eventbrite, so it has reliable support from industry [25].

Database

PostgreSQL [26] is a free and open-source object-relational database management system (ORDBMS) with powerful, enterprise-class performance. It is ACID-compliant and is known for its advanced feature set which covers data integrity, concurrency, reliability, security and extensibility [27].

Authentication

Authentication is handled via JSON Web Tokens (JWTs). Users can log in with their zID and zPass, and the back-end will contact UNSW's Lightweight Directory Access Protocol (LDAP) server to validate them. Alternatively, external users such as employers can log in with credentials that have been previously registered with *mySchool*. In both cases, the back-end issues a JWT upon successful authentication. This approach is most appropriate for a single page app that interfaces with a REST API as tokens are portable and stateless [28], and more importantly, the alternative of cookie-based authentication does not work for cross-domain requests. The tradeoff, however, is that

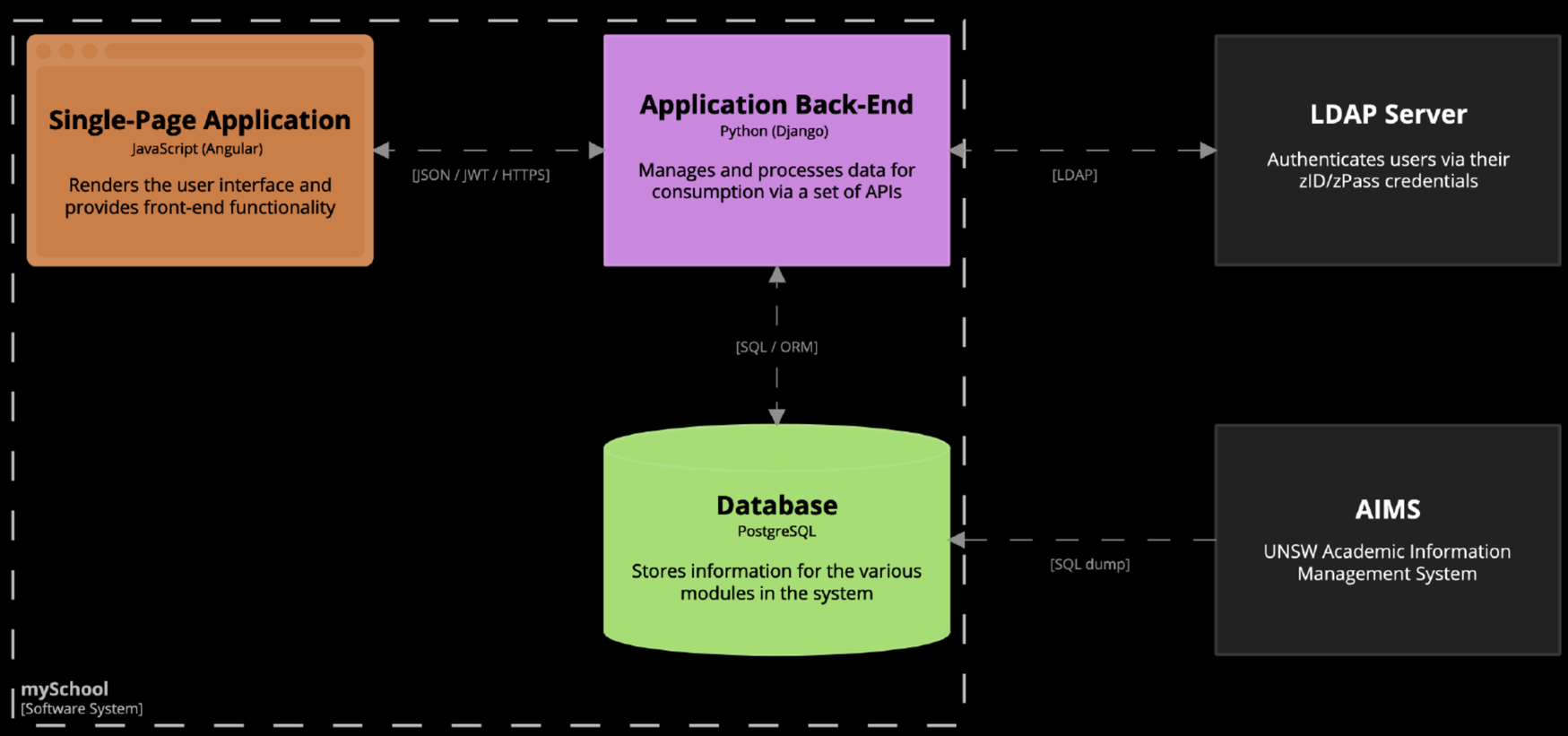

Figure 3.1: System architecture of *mySchool*

anyone who has the token can make privileged requests as the authenticated user until the end of its validity period. The JWTs issued by *mySchool* have a 2-hour expiry, but this limit can be increased or decreased depending on business requirements.

Hosting and Security

The system is currently hosted on a DigitalOcean [29] virtual private server, with DNS resolution handled by Amazon Route 53 [30]. All pages are served through HTTPS to ensure communications are encrypted, making browsing more secure for users [31]. The SSL/TLS certificate is provisioned by Let's Encrypt [32]; each one is valid for 90 days, and the server is configured to automatically renew it before expiry. Furthermore, the Django application restricts which client applications can access the API via a cross-origin resource sharing (CORS) whitelist, and it also restricts the domains that it is allowed to serve in order to mitigate HTTP host header attacks [33].

3.4 Data Import

To begin with, a total of 20 custom-made scripts written in Python 3 were used to either generate sample data or import existing data from a range of sources including,

```
Inserting (127/150): C#/C++ Developer [Autumn Compass]
Inserting (128/150): Graduate Program - Inside Sales Representative - Sydney  [VMware Australia Pty Ltd]
Inserting (129/150): Analyst [Investment Trends Pty Ltd]
Inserting (130/150): Pipeline Technical Director - Junior level [Flying Bark Productions Pty Ltd - Business Affairs]
Inserting (131/150): Pipeline Technical Director - Junior level [Flying Bark Productions Pty Ltd - Business Affairs]
Inserting (132/150): Mandarin Speaking Full-time Administration Assistant Wanted [EASI Sydney ]
Inserting (133/150): Graduate/Junior Software Developer [InfiniGold]
Inserting (134/150): Junior accounts clerk- part time 2-3 days per week [A K Unicargo International Pty Ltd]
Inserting (135/150): Senior Full Stack Developer for AI Music Startup [Muru Music]
Inserting (136/150): Summer Intern - Application Security [Symantec Australia Pty Ltd (Symantec)]
Inserting (137/150): Summer Intern - Security Experience [Symantec Australia Pty Ltd (Symantec)]
Inserting (138/150): Community Engagement Officer  [Oaktree (Oaktree) - NSW]
Inserting (139/150): Nanny Live In - BRONTE  [Mrs SHARMA, AMY]
Inserting (140/150): ONLINE ENGLISH TUTOR - URGENT HIRING NOW [SayABC]
Inserting (141/150): Industrial Officer [Australian Paramedics Association (NSW) (APA (NSW))]
Inserting (142/150): Union Organiser [Australian Paramedics Association (NSW) (APA (NSW))]
Inserting (143/150): Drone Operator [Australian UAV (AUAV)]
Inserting (144/150): Customer Service Champion Needed at Shine Music School Parramatta and Chatswood  [Shine Music School]
Inserting (145/150): Full Stack Python/Django Developer [Stitch - Stitch]
Inserting (146/150): Intern/Research Assistant [Nexus Initiative Pty Ltd]
Inserting (147/150): Language Camp Counsellor in Germany - Summer 2019 [LEOlingo Süddeutschland]
Inserting (148/150): Client Solutions Associate [FactSet Pacific Inc (FactSet)]
Inserting (149/150): Carrington Associates - Software Developer Intern - Sydney, NSW [ACS Foundation Limited (ACSF)]
Inserting (150/150): Tutors for Before & After School Touch Typing Lessons. Earn while you study!  [Typing 4 KIDS (T4K)]
Successfully inserted 150 rows
```

Figure 3.2: Sample output from job listing import script

but not limited to, UNSW Research Gateway, the official websites of individual schools and a database dump of the university's Academic Information Management System (AIMS) from 9 March 2018. Sample output from a script used to scrape job listings from the UNSW Careers and Employment website is displayed in Figure 3.2.

These scripts may be reused to update the database on a regular basis and ensure that information presented to users remains current. Possible mechanisms for automating these updates could consist of setting up scheduled cron jobs which trawl through the appropriate files, or adding server-side checks that scan for any content that is out of date by more than 12 months. Backups should be made before these tasks are run in case any issues arise with the scripts.

3.5 Implementation Approach and Tools

mySchool was implemented using an agile approach, mainly skewed towards a Kanban workflow. To facilitate this, Trello [12] was used to keep track of all development-related tasks. Each task contains a user story, acceptance criteria, priority and story point estimate representing the approximate effort needed to complete it. Figure 3.3 contains one such example of a task.

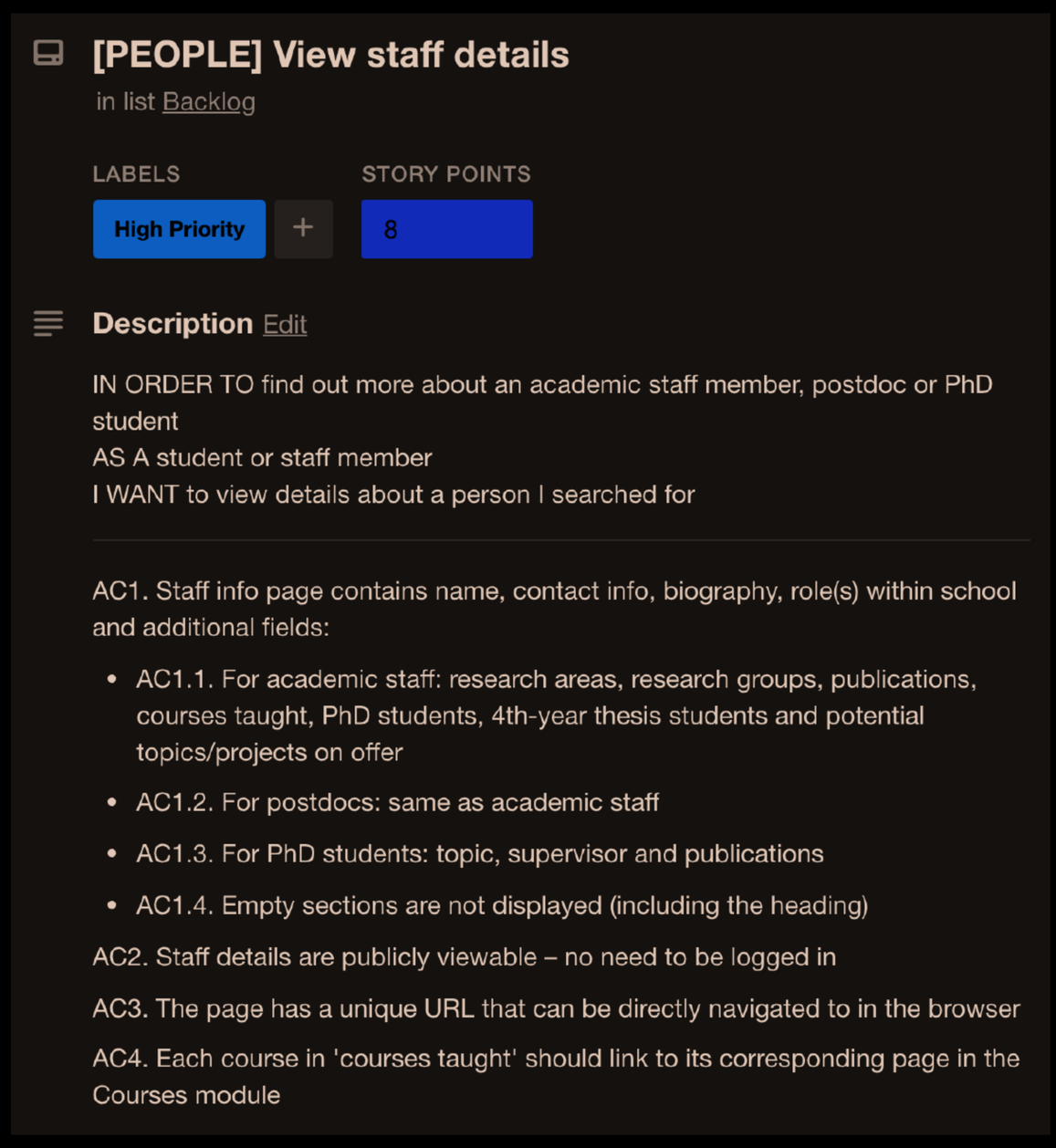

Figure 3.3: Example of a task on Trello

For source code management and version control, GitLab [34] was chosen for two reasons. Firstly, projects can be grouped under a single collection [35], making it easier to manage related repositories. Secondly, it provides a free 'Auto DevOps' feature which has been configured to automatically build, test and deploy a new release of *mySchool* after every commit to the `master` branch [36], allowing for more time and effort to develop the product rather than shipping it out to users. The architecture of this pipeline is detailed in Figure 3.4, and an example given in Figure 3.5 – if any unit test or end-to-end test fails, the entire build process is halted and no deployment is made. Bitbucket, a competitor, offers similar functionality to this but with only 50 free build minutes per month, compared to 2,000 with GitLab [37].

Under this continuous delivery process, prospective users were able to provide feedback as they tested each new version, and this feedback was subsequently used to evaluate how well *mySchool* was meeting requirements on an ongoing basis.

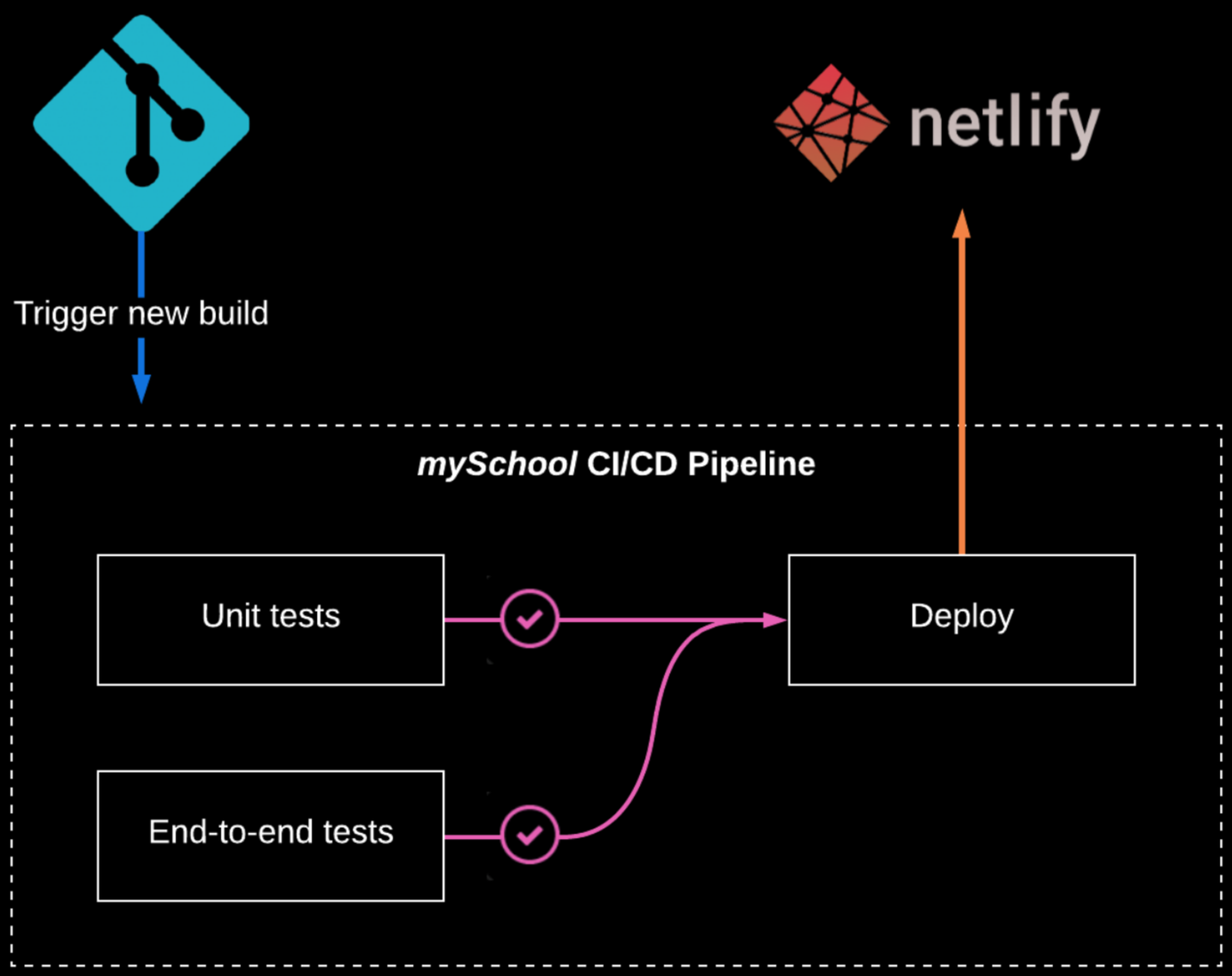

Figure 3.4: Pipeline architecture

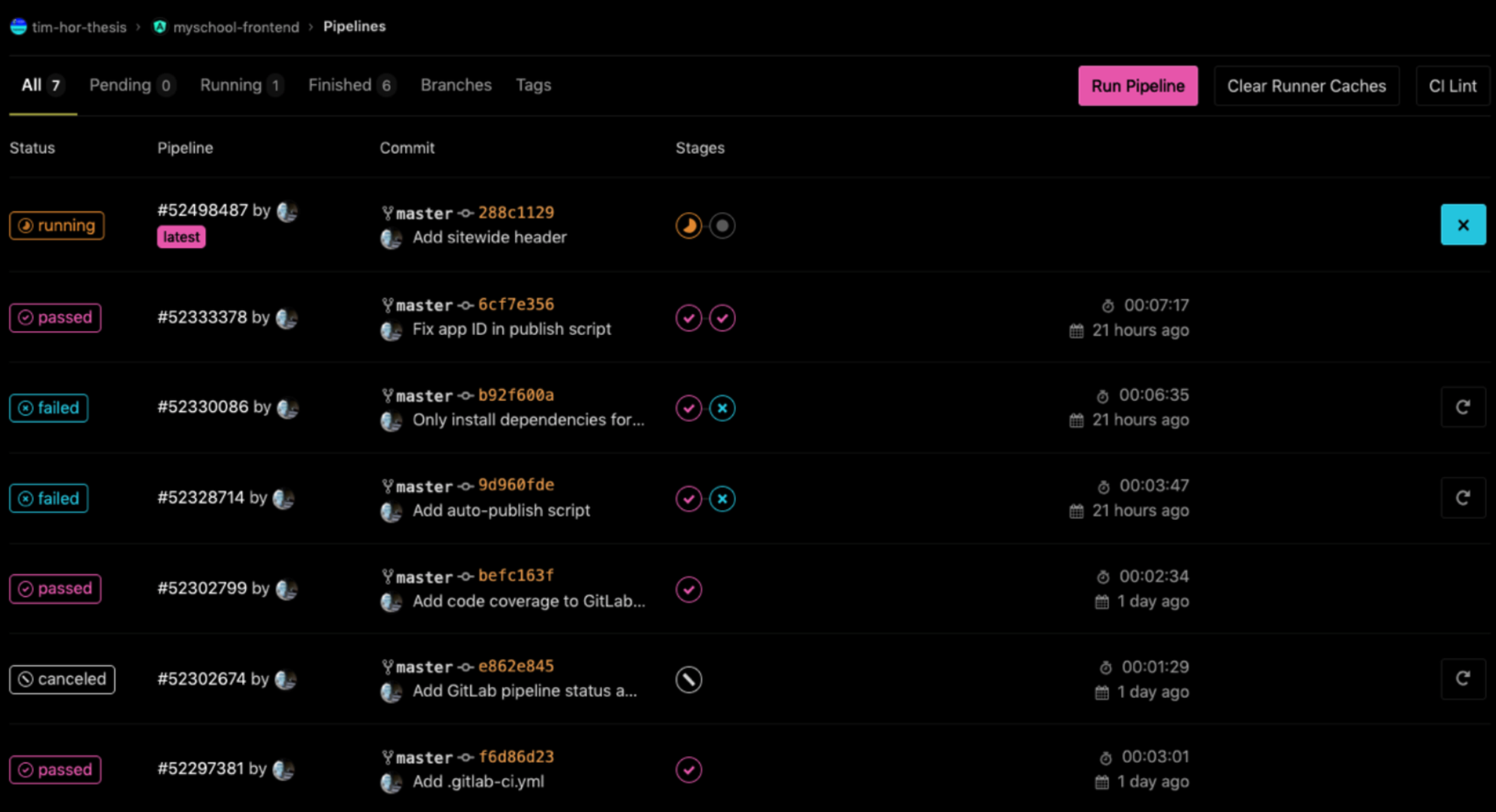

Figure 3.5: Pipeline example

3.6 Implementation Trade-offs

The main trade-off regarding the implementation was finding a compromise between how much data to store directly in the *mySchool* database and how much to request from other sources as needed. To minimise redundancy, *mySchool* could pull data from application programming interfaces (APIs) connected to existing UNSW systems and databases. Unfortunately, no such APIs were found, and any data sources currently being maintained by the university have highly restricted access. To overcome this problem in *mySchool*, the back-end implementation included the development of a new database driven API, bringing with it several advantages:

- The front-end and back-end are loosely coupled with a clear separation of concerns. If Angular needs to be replaced with another framework, the data processing code and database calls can still be utilised without much change, and vice versa for Django.
- It reduces the dependency *mySchool* has on systems maintained by other people as the availability of bespoke API endpoints eliminates the need to query multiple external sources every time. The database simultaneously acts as a 'cache' for data that is imported from other sources.
- Others can build on top of this API for their own projects which may not be necessarily related to *mySchool*.

3.7 Database Design

An entity-relationship (ER) diagram for the *mySchool* database is illustrated in Figure 3.6. Due to its large size, this only shows the relationships between entities. Refer to Appendix A for the full set of attributes for each entity.

Compared to the initial plan, attributes for the `User` entity had to be updated partway through development to maintain compatibility with Django's user model.

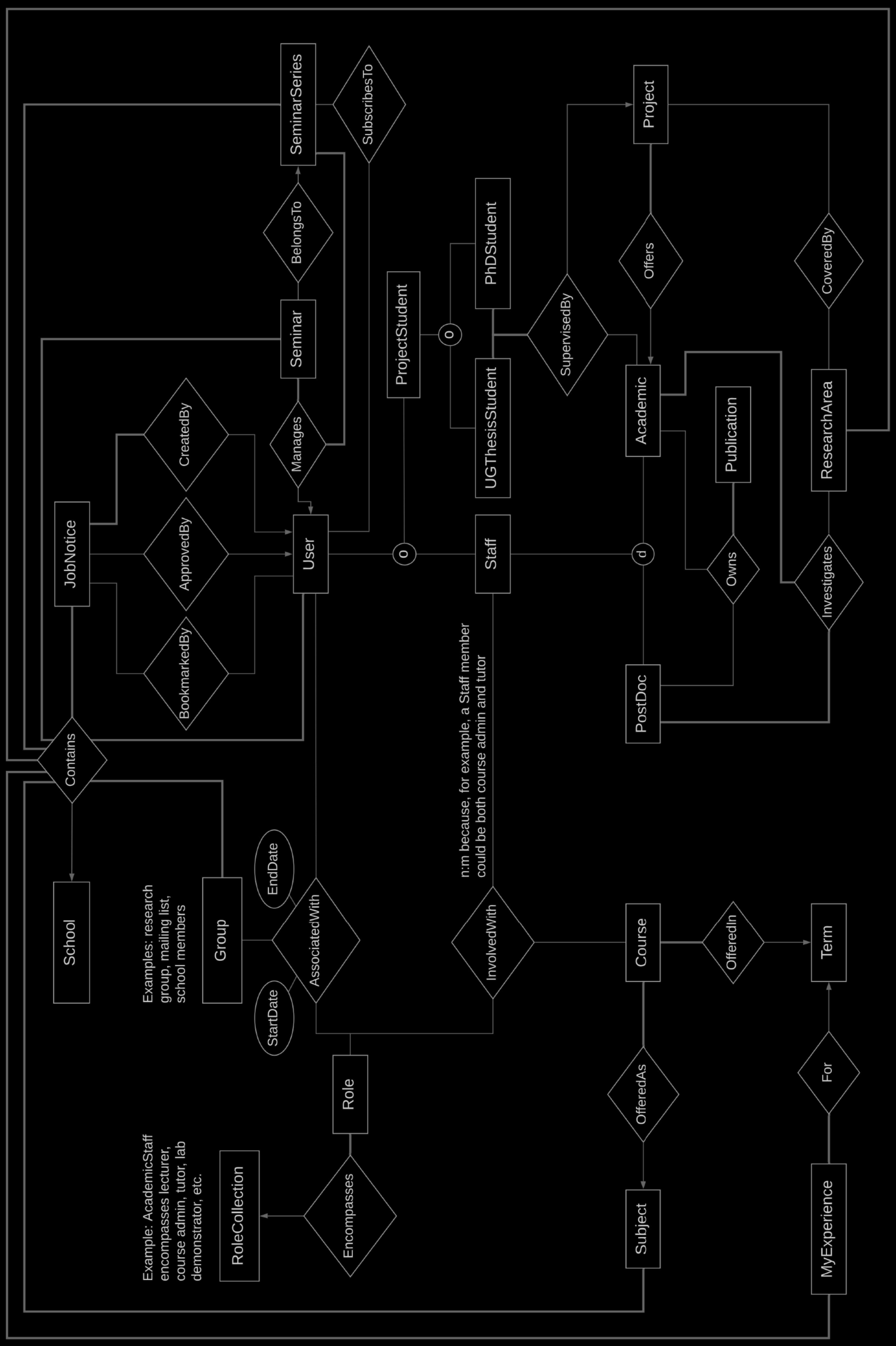

Figure 3.6: ER diagram for *mySchool* database showing relationships between entities

Chapter 4

Evaluation

4.1 Usability

By replicating the UI and UX of the prototype shown to users during the initial research phase, the ease of navigation they appreciated in the mockup could be matched in the live version of *mySchool*. This was supported by empirical feedback collected with each release. The short feedback loop made possible by releasing at least once every fortnight also afforded greater flexibility in validating requirements and responding to change [38]. In addition, the main areas of the application are both keyboard and screen reader accessible, catering to every user including those with disabilities requiring the use of assistive technology [39].

4.2 Correctness

Every front-end component has its own set of unit tests ensuring that all functions work in a predictable manner by comparing expected inputs and outputs. An end-to-end testing framework was set up as well to simulate example usage scenarios and check that dependencies are integrated properly. As described in the previous chapter, the

continuous delivery pipeline runs both types of tests whenever a new build is triggered to minimise the possibility of any bugs making it into the production version of *mySchool*.

4.3 Performance and Security

The entire application was designed to be modular from the very beginning and components have reused wherever possible, resulting in less code to maintain and debug. For the API, raw SQL was written to perform more complex queries involving several database tables, thus bypassing the more expensive ORM completely. Furthermore, appropriate security measures and access control mechanisms have been put in place to safeguard sensitive information.

4.4 Areas for Improvement

Regardless of how well-built a system may be, there is always room for improvement, and *mySchool* is no exception. It is recommended that the following areas be addressed before continuing development.

- The system is hosted offsite and therefore requires a VPN connection to facilitate zID/zPass logins. This can be fixed by migrating it to an on-premise server connected to the university network.
- The suite of unit and end-to-end tests are not fully comprehensive – just over 70% of statements are covered, short of the recommended minimum of 80% [40]. Also, the back-end API does not have any such tests to be validated against.
- Due to time constraints, several features remain unimplemented. For example, additional interactivity with the system such as bookmarking job notices and editing profile information are currently not supported.
- Some data is outdated – for example, the staff list contains people who no longer work for UNSW.

- Some data may be cached locally to reduce the number of API calls being made and hence make the system more robust against potential connectivity issues.
- The system relies on the browser's local storage for state management but a dedicated library such as NgRx would be more effective.

4.5 Summary

The current implementation of *mySchool* meets the vast majority of both functional and non-functional requirements. It provides an intuitive UI and UX, contains checks to maintain correctness, and is performant and secure. In summary, the overarching goal of providing an extensible platform for staff and students to carry out core school functions has been successfully achieved.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This report has explained the motivation and requirements for a web-based school management system, *mySchool*. Common functionality shared among existing systems has been identified and alternative solutions evaluated, with the ultimate decision made to deliver a generic yet extensible application. The high-level details surrounding its implementation have also been discussed, including design decisions as well as ongoing maintenance. Finally, the end product was analysed against an evaluation framework and found to satisfy the expected outcomes.

5.2 Future Work

Following on from this project, there are many possibilities and directions in which to extend *mySchool*.

- **Teaching allocation**. This can be further broken down into specific areas including tutor management, term planning for staff and the ability to track course ratings over time (building on the existing *myExperience* interface).

- **Thesis management**. This could be provided as an alternative to Moodle with functionality for topic nomination, submission of deliverables and marking.
- **Results management**. In addition to displaying students' results, there could be functionality to track their academic standing and automatically set up meetings if it is anything other than 'good'.
- **Internal exam timetable**. It would be valuable for schools holding internal exams to have a standardised interface to communicate details to students as this is another common administrative function.
- **News & events**. The *mySchool* homepage currently contains a placeholder section which can be adapted to show news items and upcoming events. Social media feeds could be incorporated here as well.

Responses to questions 11-13 of the survey from Chapter 2 also contain valuable suggestions for improvements that can be implemented.

Bibliography

[1] L. Lifang and Q. Ziling, "On knowledge management in higher education," in *2011 International Conference on Product Innovation Management (ICPIM 2011)*, pp. 177–179, July 2011.

[2] X. Wang and F. Xu, "Study on knowledge management system based on MAS," in *2010 International Conference on Networking and Digital Society*, vol. 2, pp. 395–398, May 2010.

[3] H. Ramayani, G. Wang, H. Prabowo, T. Sriwidadi, R. Kodirun, and A. Gunawan, "Improving organizational knowledge management (KM) through cloud based platform in higher education," in *2017 International Conference on Information Management and Technology (ICIMTech)*, pp. 10–13, Nov 2017.

[4] S. Numprasertchai and Y. Poovarawan, "Enhancing university competitiveness through ICT based knowledge management system," in *2006 IEEE International Conference on Management of Innovation and Technology*, vol. 1, pp. 417–421, June 2006.

[5] UNSW Marketing Services, "UNSW 2025 strategy." https://www.2025.unsw.edu.au/sites/default/files/uploads/unsw_2025strategy_201015.pdf, accessed 6 September 2018, 2015. UNSW Sydney.

[6] R. Pearce, "IT to be hit hardest by proposed UNSW job cuts." https://www.computerworld.com.au/article/611142/it-hit-hardest-by-proposed-unswjob-cuts/, accessed 6 September 2018, 2016. Computerworld.

[7] UNSW Current Students, "Careers portal." https://student.unsw.edu.au/careers-portal, accessed 6 September 2018, 2018. UNSW Sydney.

[8] School of Computer Science and Engineering, "Course offerings." https://www.engineering.unsw.edu.au/computer-science-engineering/courses-programs/course-offerings, accessed 14 October 2018, 2018. Faculty of Engineering, UNSW Sydney.

[9] Data61, "Flexible student jobs." https://unswcse.ribit.net/, accessed 16 June 2019, 2019. Data61.

[10] Maze home page. https://maze.design/, accessed 17 June 2019, 2019. Maze.

[11] Maze, "Understanding your Results." https://help.maze.design/en/article/understanding-your-results-1fwx3d8/, accessed 17 June 2019, 2019. Maze.

[12] Trello home page. https://trello.com/, accessed 18 October 2018, 2018. Atlassian.

[13] I. Indu and P. M. R. Anand, "Hybrid authentication and authorization model for web based applications," in *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pp. 1187–1191, March 2016.

[14] T. Xie, N. Tillmann, and P. Lakshman, "Advances in unit testing: Theory and practice," in *2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)*, pp. 904–905, May 2016.

[15] Angular home page. https://angular.io/, accessed 18 October 2018, 2018. Google.

[16] Google, "Angular - architecture overview." https://angular.io/guide/architecture, accessed 16 October 2018, 2018. Google.

[17] Google, "Angular - security." https://angular.io/guide/security/, accessed 19 June 2019, 2019. Google.

[18] Google, "Angular - features & benefits." https://angular.io/features/, accessed 16 October 2018, 2018. Google.

[19] L. Caniglia, "Building accessible single page apps." https://codeburst.io/building-accessible-single-page-apps-2ea3e4fbbc01, accessed 16 October 2018, 2017. codeburst.io.

[20] Grab Web Team, "Grab front end guide." https://github.com/grab/front-end-guide#single-page-apps-spas, accessed 19 June 2019, 2018. GitHub.

[21] Django home page. https://www.djangoproject.com/, accessed 18 October 2018, 2018. Django Software Foundation.

[22] Django Software Foundation, "Django at a glance." https://docs.djangoproject.com/en/2.1/intro/overview/, accessed 16 October 2018, 2018. Django Software Foundation.

[23] Django Software Foundation, "Security in Django." https://docs.djangoproject.com/en/2.1/topics/security/, accessed 16 October 2018, 2018. Django Software Foundation.

[24] Jeremy Morris, "How to enable and connect the Django admin interface." https://www.digitalocean.com/community/tutorials/how-to-enable-and-connect-the-django-admin-interface, accessed 16 October 2018, 2018. DigitalOcean.

[25] A. Pietrasik, "Top 10 Django apps and why companies are betting on this framework." https://www.netguru.co/blog/top-10-django-apps-and-why-companies-are-betting-on-this-framework, accessed 6 September 2018, 2018. Netguru.

[26] PostgreSQL home page. https://www.postgresql.org/, accessed 18 October 2018, 2018. PostgreSQL Global Development Group.

[27] PostgreSQL Global Development Group, "PostgreSQL: About." https://www.postgresql.org/about/, accessed 16 October 2018, 2018. PostgreSQL Global Development Group.

[28] R. Chenkie, "5 steps to add modern authentication to legacy apps using JWTs." https://auth0.com/blog/5-steps-to-add-modern-authentication-to-legacy-apps-using-jwts/ , accessed 19 June 2019, 2019. Auth0.

[29] DigitalOcean home page. https://www.digitalocean.com, accessed 19 June 2019, 2019. DigitalOcean.

[30] Amazon Web Services, "Route 53." https://aws.amazon.com/route53/, accessed 19 June 2019, 2019. Amazon.

[31] Electronic Frontier Foundation, "HTTPS Everywhere." https://www.eff.org/https-everywhere, accessed 19 June 2019, 2019. Electronic Frontier Foundation.

[32] Let's Encrypt home page. https://letsencrypt.org, accessed 19 June 2019, 2019. Internet Security Research Group (ISRG).

[33] Django Software Foundation, "Settings - ALLOWED_HOSTS." https://docs.djangoproject.com/en/2.2/ref/settings/#allowed-hosts/, accessed 19 June 2019, 2019. Django Software Foundation.

[34] GitLab home page. https://about.gitlab.com/, accessed 18 October 2018, 2018. GitLab.

[35] J. van der Voort and J. Vosmaer, "GitLab feature highlight: Groups." https://about.gitlab.com/2014/06/30/feature-highlight-groups/, accessed 16 October 2018, 2014. GitLab.

[36] GitLab, "Auto DevOps." https://about.gitlab.com/auto-devops/, accessed 16 October 2018, 2018. GitLab.

[37] GitLab, "GitLab compared to other DevOps tools." https://about.gitlab.com/devops-tools/bitbucket-vs-gitlab.html, accessed 15 October 2018, 2018. GitLab.

[38] L. Ekas, "Short feedback loops everywhere!." https://www.ibm.com/developerworks/community/blogs/beingagile/entry/short_feedback_loops_everywhere, accessed 20 June 2019, 2019. IBM Community.

[39] UNSW Sydney, "Accessibility." https://www.unsw.edu.au/accessibility/, accessed 20 June 2019, 2019. UNSW Sydney.

[40] S. Pittet, "An introduction to code coverage." https://www.atlassian.com/continuous-delivery/software-testing/code-coverage, accessed 20 June 2019, 2019. Atlassian.

Appendix A

A.1 ER Diagram Attributes

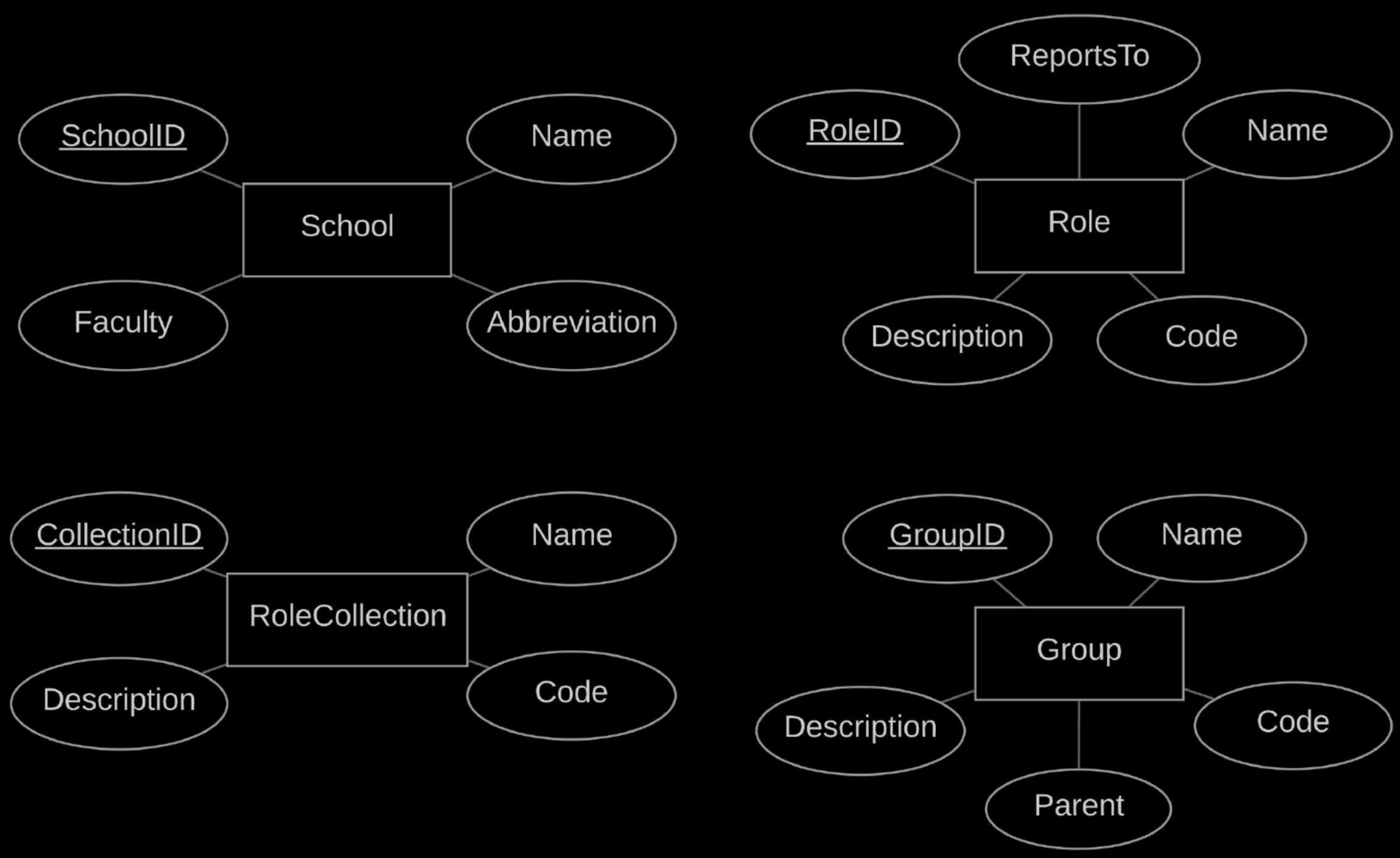

Figure A.1: ER diagram attributes for `School`, `RoleCollection`, `Role`, `Group` and `MyExperience`

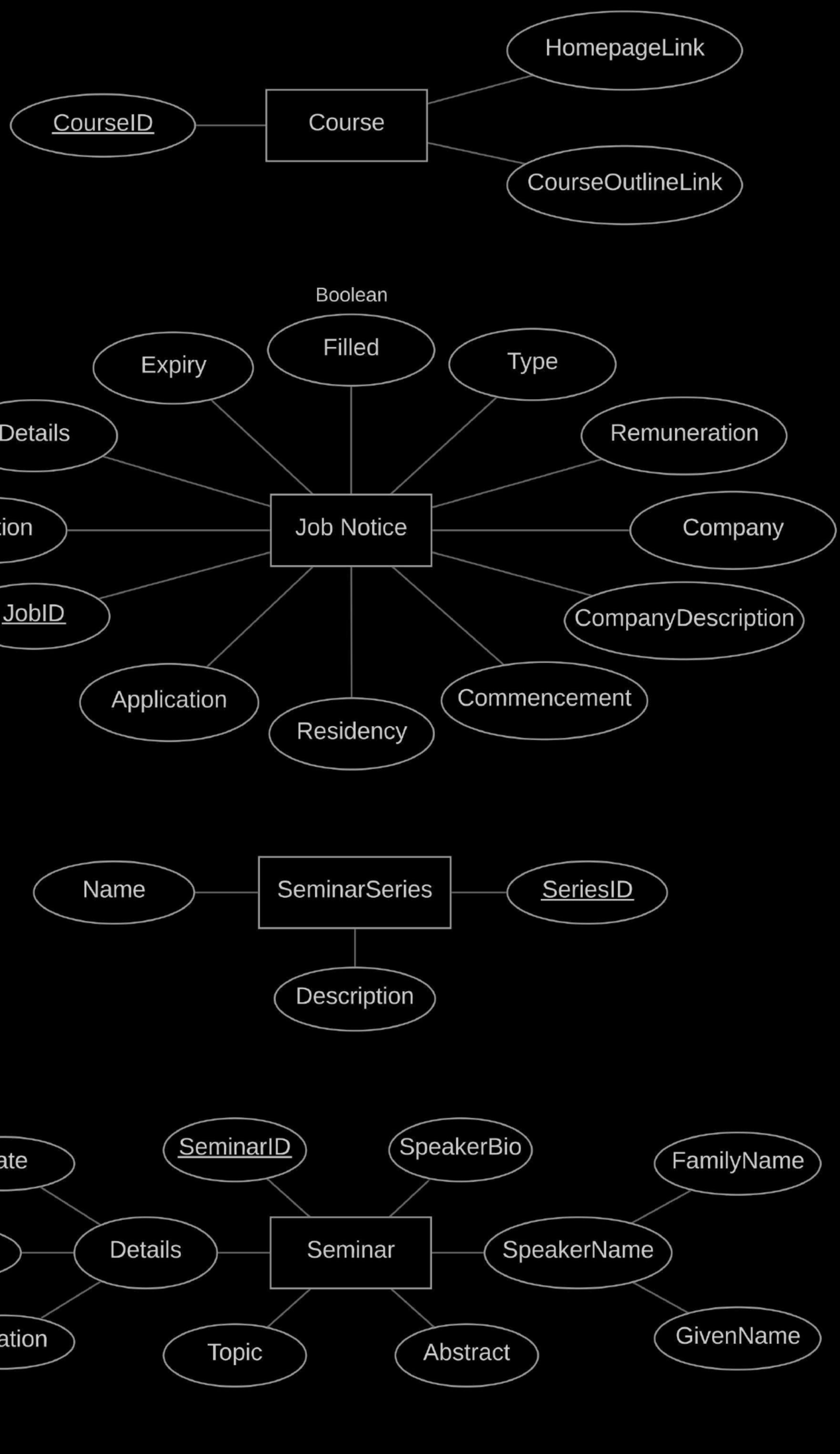

Figure A.2: ER diagram attributes for `Course`, `JobNotice`, `SeminarSeries` and `Seminar`

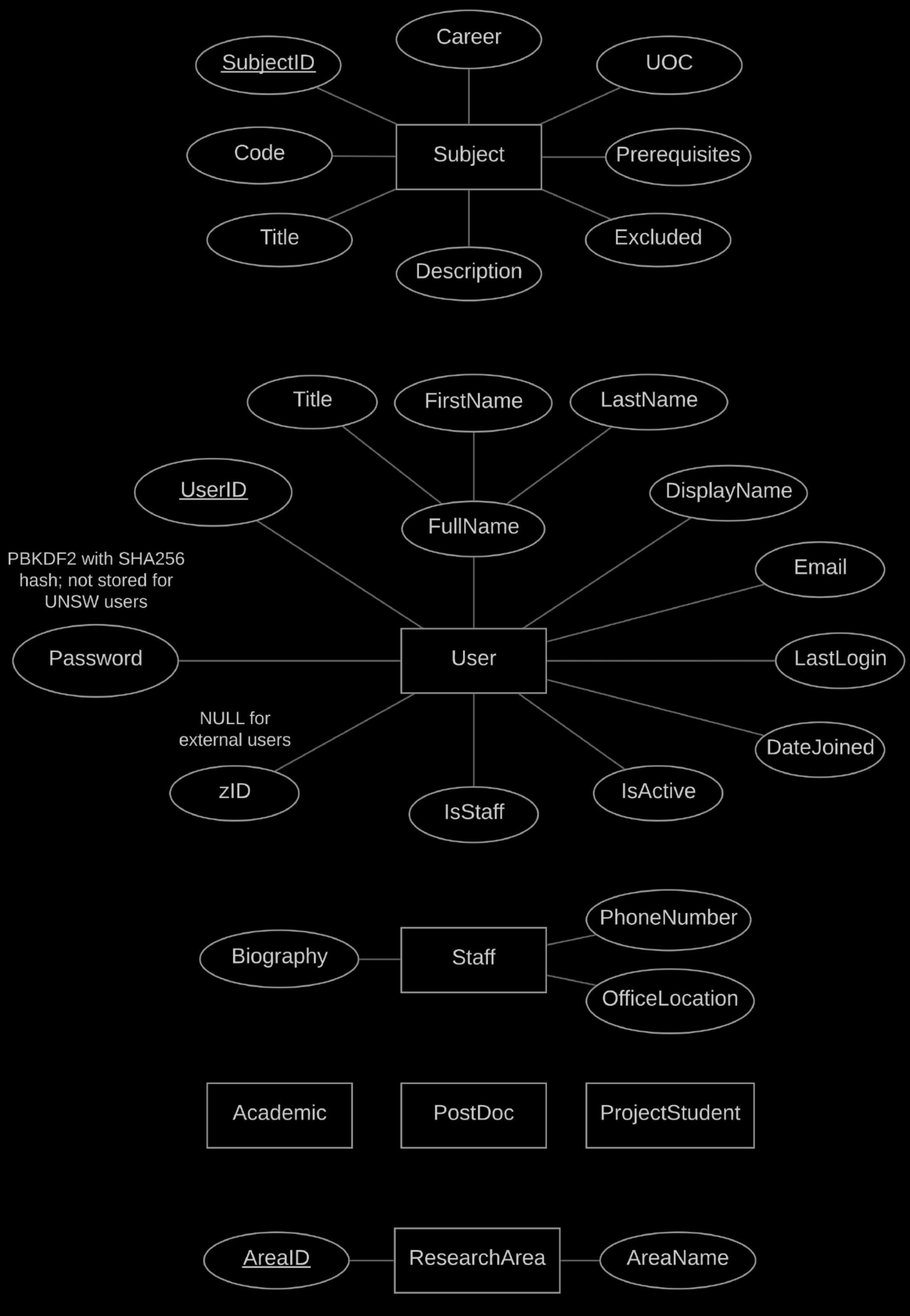

Figure A.3: ER diagram attributes for `Subject`, `User`, `Staff`, `Academic`, `PostDoc`, `ProjectStudent` and `ResearchArea`

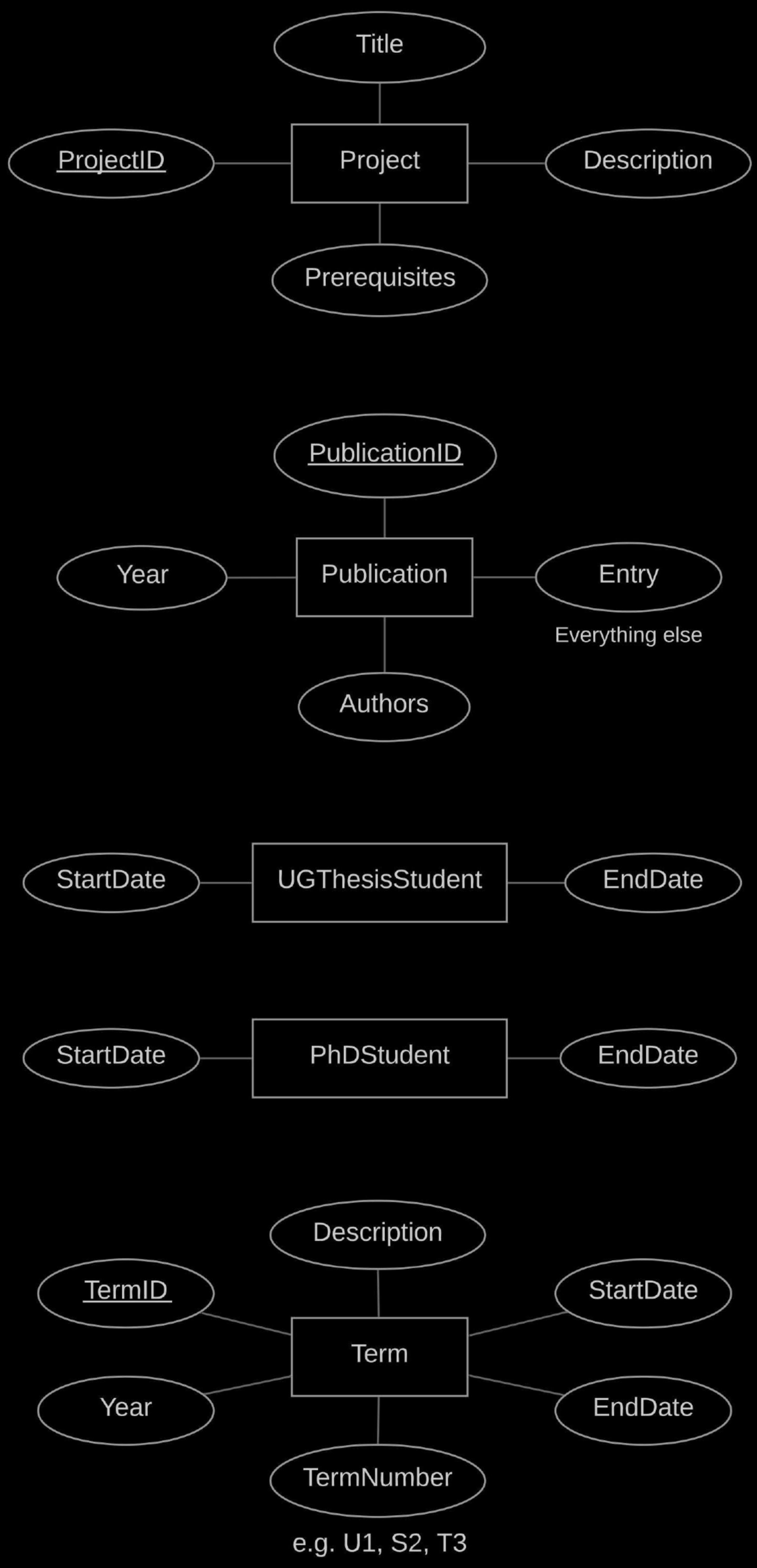

Figure A.4: ER diagram attributes for `Project`, `Publication`, `UGThesisStudent`, `PhDStudent` and `Term`

Appendix B

B.1 Usability Testing Heatmaps

Mission 1: Log in

Log in to proceed with the demo.

Mission 2: Look up a staff member

Search for a staff member by name and go to their profile (assume the first result is a match).

Mission 3: View and edit own profile

Assume you're logged in as John Doe. View your own profile and make some 'edits' to it (clicking 'Update Profile' will suffice).

Mission 4: Browse and subscribe to upcoming seminars

Open up the list of upcoming seminars and click on Seminar C. Then sign up for email notifications.

Mission 5: Discover jobs

Browse for job notices and open Job Listing C to view more details about it.

Mission 6: Find information on a course

Go to the Courses section and click on any course to open the UNSW Handbook (this will open a new tab). Then come back here and log out to complete the maze.

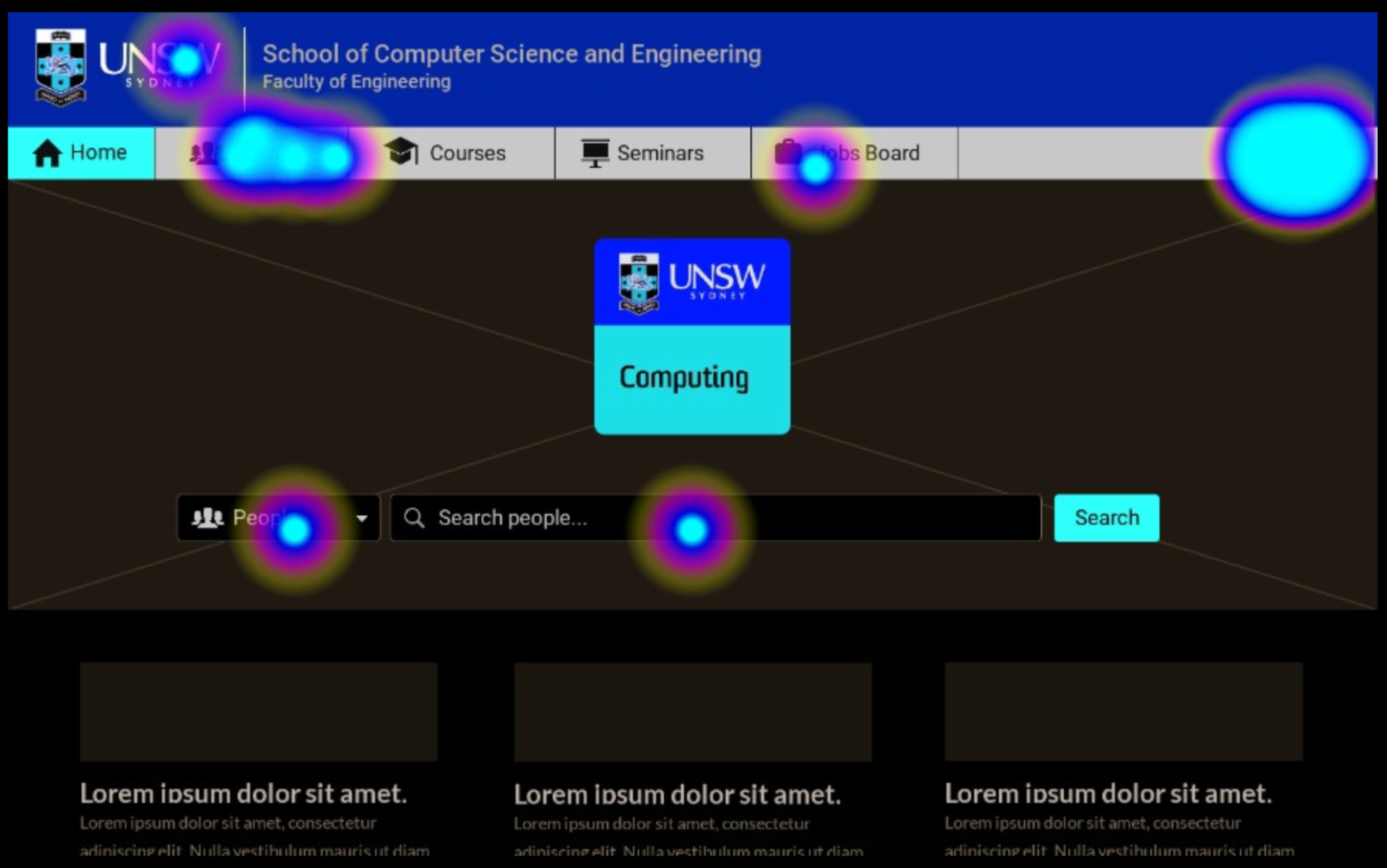

Figure B.1: Heatmap for mission 1 - log in

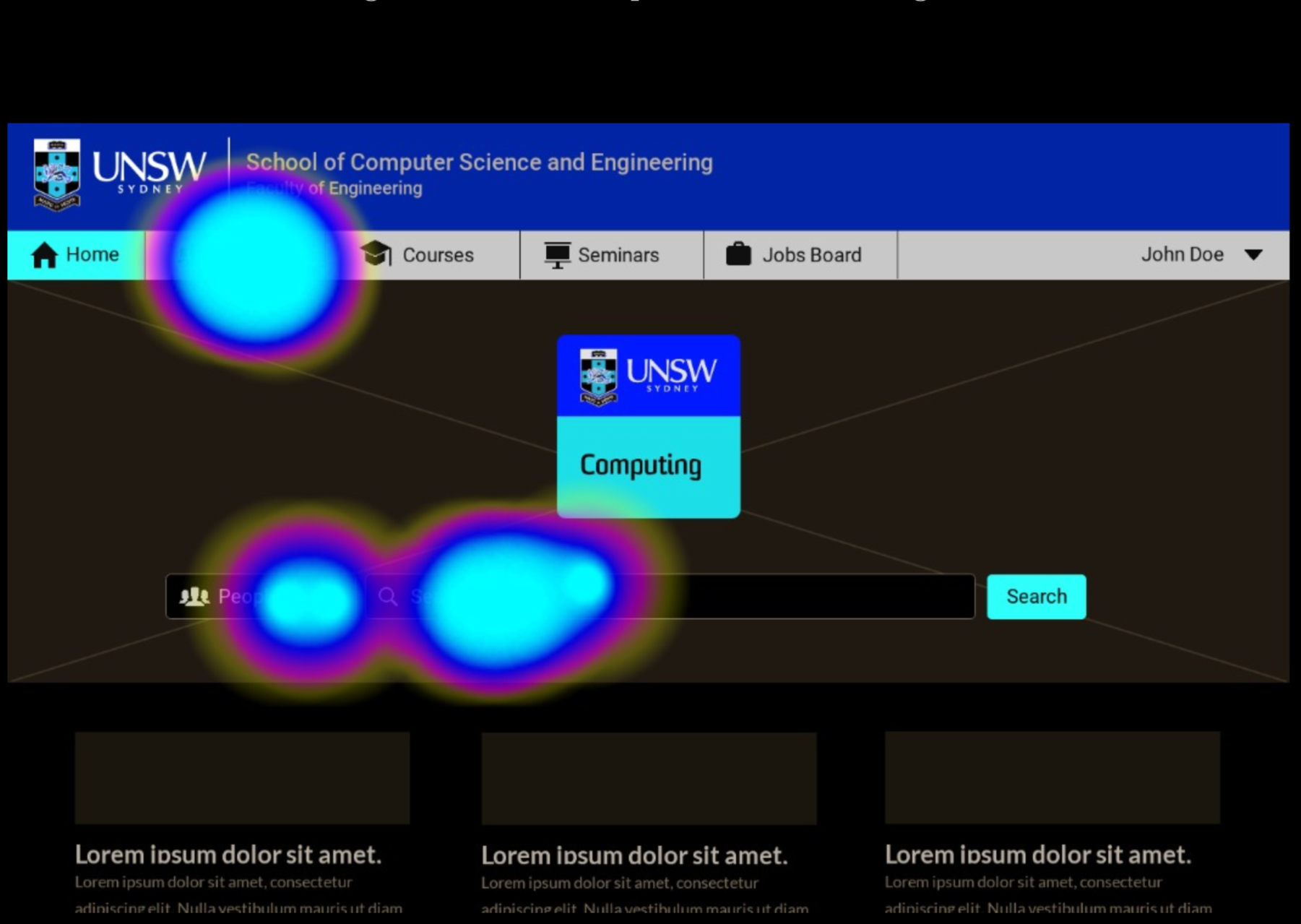

Figure B.2: Heatmap for mission 2 - look up a staff member

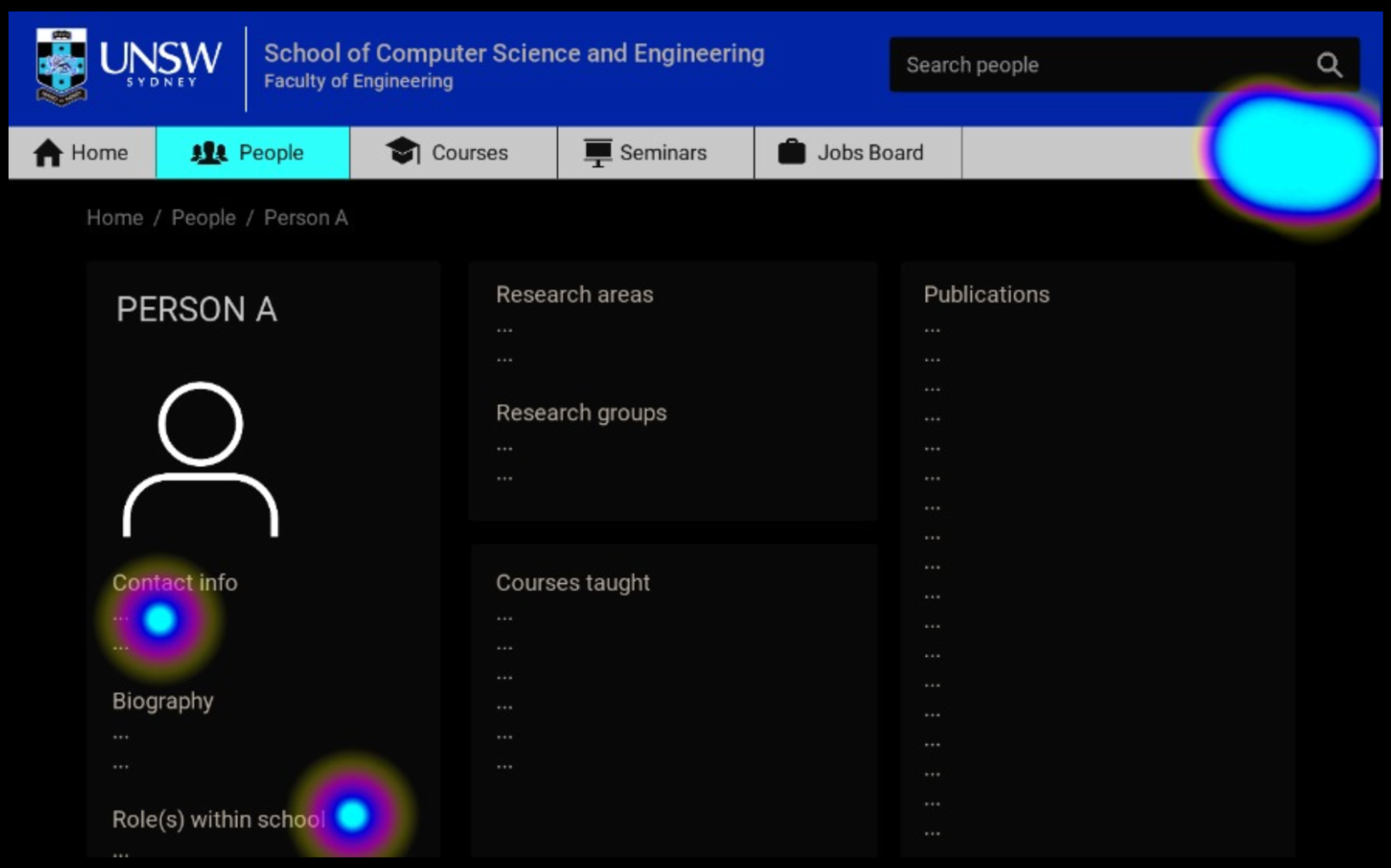

Figure B.3: Heatmap for mission 3 - view and edit own profile

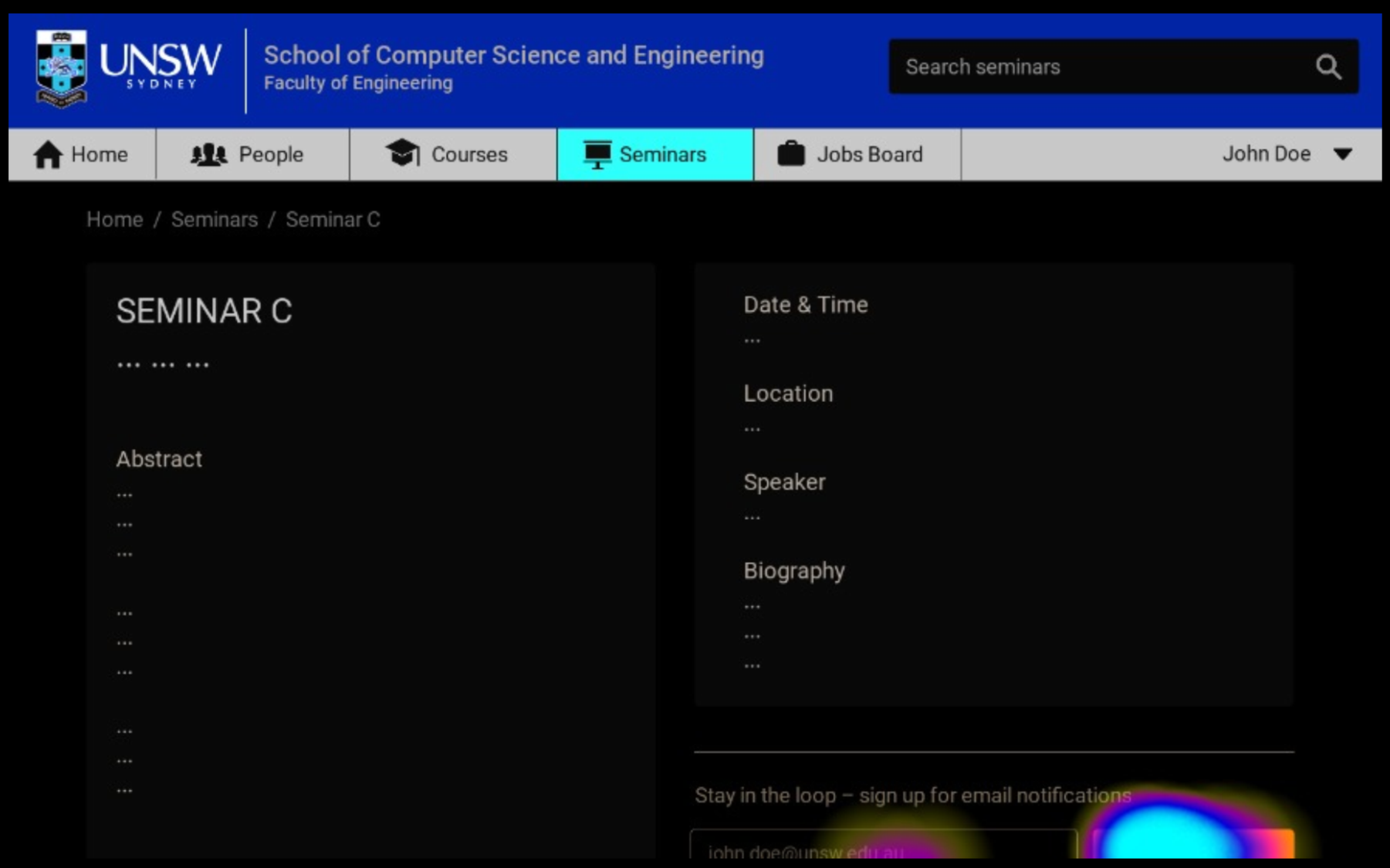

Figure B.4: Heatmap for mission 4 - browse and subscribe to upcoming seminars

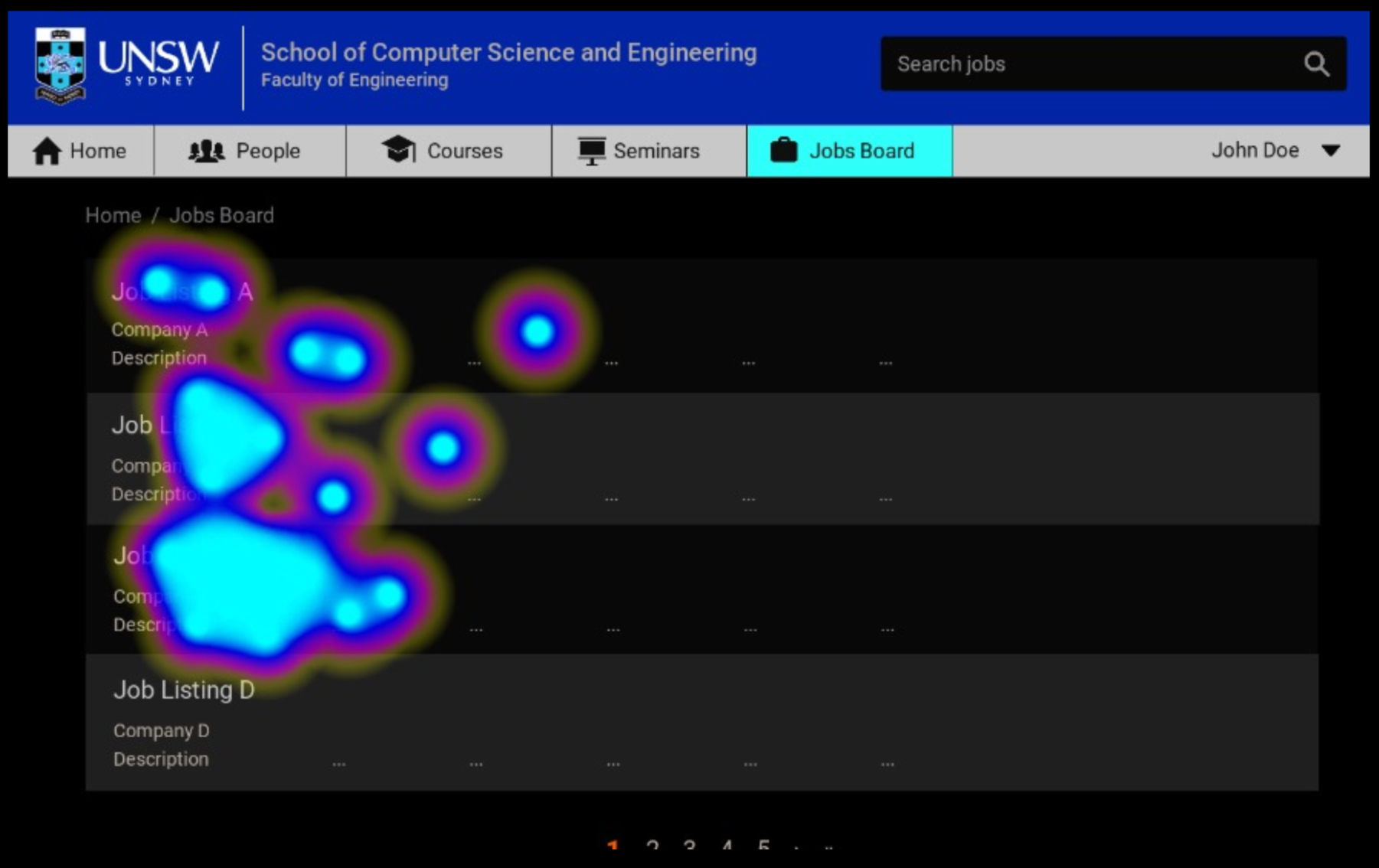

Figure B.5: Heatmap for mission 5 - discover jobs

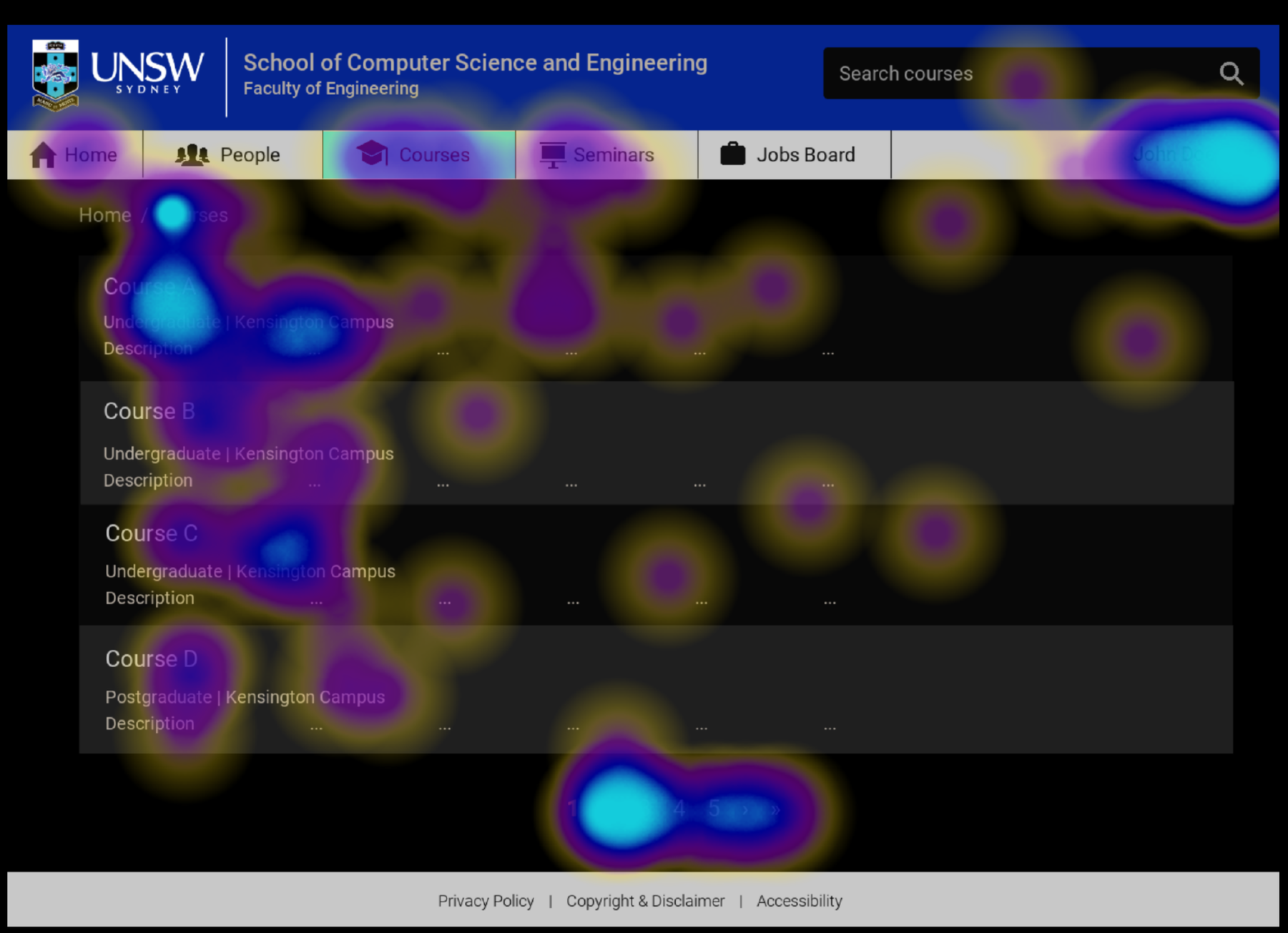

Figure B.6: Heatmap for mission 6 - find information on a course