CRISP

for

Cloud Computing (INFS 803)

by

GROUP 8

Juan Herbst (13840146)

Mike Chene (19070279)

Tommaso Cammelli (23215488)

Auckland University of Technology, Auckland, New Zealand

Table of Contents

[Introduction 3](#_Toc166445971)

[Project Planning 3](#_Toc166445972)

[Project Design 4](#_Toc166445973)

[User Experience (UX) & Screen Mock-ups 4](#_Toc166445974)

[Cloud Service Architecture 8](#_Toc166445975)

[Interactions and Sequences 10](#_Toc166445976)

[Schema Design 11](#_Toc166445977)

[REST API Endpoints. 14](#_Toc166445978)

[Recruiter Profile 15](#_Toc166445979)

[User Manual 18](#_Toc166445980)

[Summary 18](#_Toc166445981)

Table of Figures

[**Figure 1** Crisp Kanban Board 3](#_Toc166445982)

[**Figure 2** Sign up for Crisp. 4](#_Toc166445983)

[**Figure 3** Sign in to Crisp. 4](#_Toc166445984)

[**Figure 4** Create, Read and Update the Recruiter Profile. 5](#_Toc166445985)

[**Figure 5** Create, Read and Update the Candidate Profile. 6](#_Toc166445986)

[**Figure 6** The recruiter finds Candidates applied by Job Category. 7](#_Toc166445987)

[**Figure 7** Candidate applying for a Job. 7](#_Toc166445988)

[**Figure 8** Create a Job Posting. 8](#_Toc166445989)

[**Figure 9** Cloud Service used by Crisp. 9](#_Toc166445990)

[**Figure 10** Sequence Diagram depicting interaction between the User, Cognito, API Gateway, Lambda and DynamoDB. 10](#_Toc166445991)

[**Figure 11** Sequence Diagram depicting interaction between the User, Cognito and S3. 11](#_Toc166445992)

[**Figure 12** Candidate Apply for a Job 16](#_Toc166445993)

[**Figure 13** Recruiter Finds and Engages with Candidates 17](#_Toc166445994)

TABLE OF TABLES

[**Table 1** JSON Schema for the Recruiter Profile. 12](#_Toc166446029)

[**Table 2** JSON Schema for the Candidate Profile. 12](#_Toc166446030)

[**Table 3** JSON Schema for Notes. 13](#_Toc166446031)

[**Table 4** JSON Schema for Categories. 13](#_Toc166446032)

[**Table 5** JSON Schema for Jobs. 14](#_Toc166446033)

[**Table 6** Recruiter Profiles REST Endpoints. 14](#_Toc166446034)

[**Table 7** Candidate Profiles REST Endpoints. 15](#_Toc166446035)

[**Table 8** Notes REST Endpoints. 15](#_Toc166446036)

[**Table 9** Categories REST Endpoints. 15](#_Toc166446037)

[**Table 10** Jobs REST Endpoints. 15](#_Toc166446038)

## Introduction

## Project Planning

Scrum, a widely adopted agile framework, was used during the project design, development, and testing phases. Founded on transparency, inspection, and adaptation principles, Scrum empowers cross-functional teams to deliver high-quality products incrementally within short time frames, known as sprints. Through its structured ceremonies, including daily stand-ups, sprint planning, reviews, and retrospectives, Scrum fosters a culture of continuous improvement and responsiveness to changing requirements.

**Figure 1**  
Crisp Kanban Board

A screenshot of a computer

Description automatically generated

Trello was utilised as a platform to effectively track and manage the product's roadmap and maintain thorough oversight of the fortnightly sprint backlog.

## Project Design

### User Experience (UX) & Screen Mock-ups

Screen mock-ups visually represent the user interface, providing stakeholders with a tangible preview of the final product's layout and functionality. These mock-ups facilitate collaborative discussions, enabling teams to iterate on design concepts, gather feedback, and refine user interactions before implementation.

**Figure 2**  
Sign up for Crisp.

A screenshot of a login form

Description automatically generated

To register for Crisp, users must provide their email address and password and confirm their password. Additionally, they must specify their registration type and choose between signing up as recruiters or candidates.

**Figure 3**   
Sign in to Crisp.

A screenshot of a login form

Description automatically generated

Users utilise their registered email address and a valid password to authenticate and sign-in to the Crisp platform.

**Figure 4**  
Create, Read and Update the Recruiter Profile.

A screenshot of a computer

Description automatically generated

**Figure 5**  
Create, Read and Update the Candidate Profile.

A screenshot of a computer

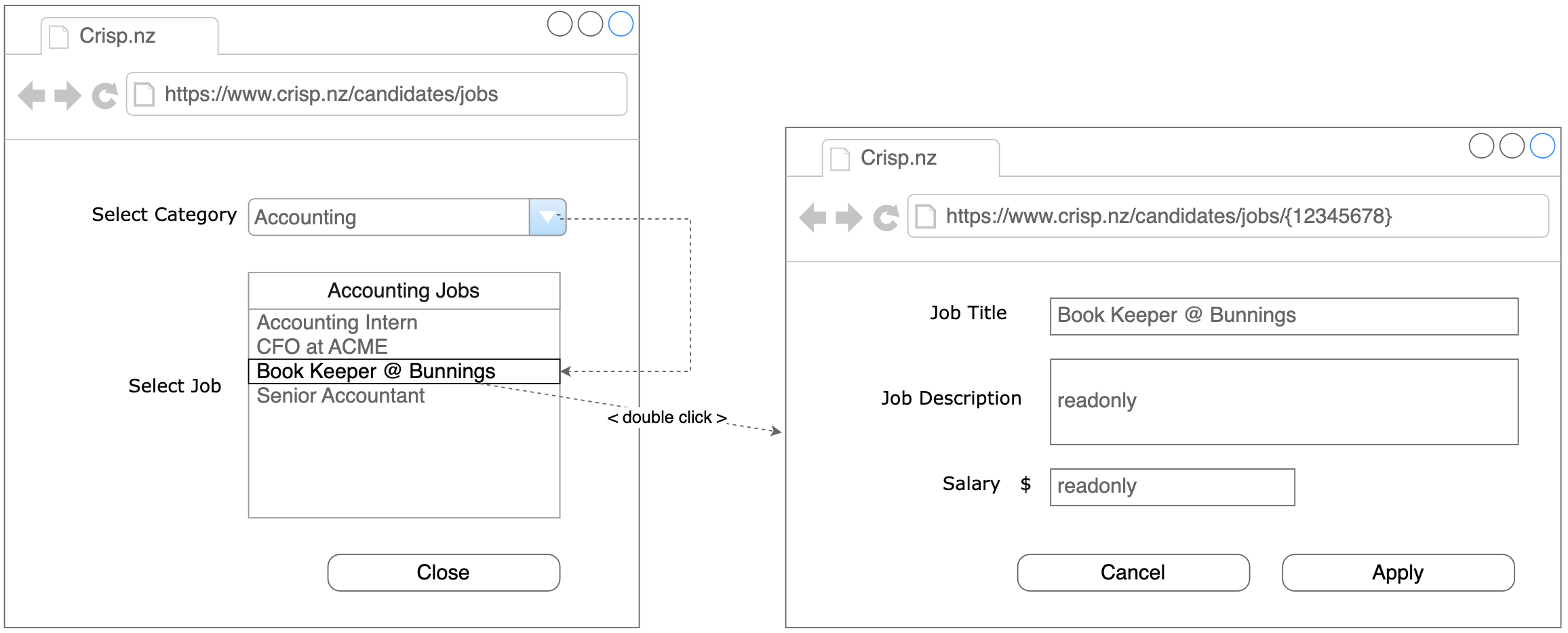
Description automatically generated

**Figure 6**  
The recruiter finds Candidates applied by Job Category.

A screenshot of a computer

Description automatically generated

**Figure 7**  
Candidate applying for a Job.



**Figure 8**  
Create a Job Posting.

A screenshot of a computer

Description automatically generated

### Cloud Service Architecture

*Amazon Cognito* is an identity management service offered by Amazon Web Services (AWS). It facilitates secure user authentication and authorisation for web and mobile applications. Amazon Cognito offers a range of features, including user sign-up, sign-in, and account recovery, along with user profile management and data synchronisation across multiple devices. Furthermore, it supports multi-factor authentication (MFA), social identity provider integration, and fine-grained access control policies.

*Amazon Route 53* is a highly scalable and reliable Domain Name System (DNS) web service provided by AWS. Route 53 effectively translates human-readable domain names into corresponding numeric IP addresses, thereby routing user requests across the Internet. Route 53 offers a range of functions, including domain registration, DNS health monitoring, traffic routing policies, and domain name system security extensions (DNSSEC).

*Amazon Simple Storage Service (S3)* provides a secure and reliable platform for storing and retrieving vast data. With its distributed architecture spanning multiple geographic regions, Amazon S3 ensures high durability and availability, making it suitable for various applications, from data backup and archival to content delivery and big data analytics.

**Figure 9**  
Cloud Service used by Crisp.

A diagram of a computer server

Description automatically generated

*Amazon API Gateway* is a fully managed service provided by AWS that allows developers to create, publish, maintain, monitor, and secure APIs at any scale. As a front door for applications to access data, business logic, or functionality from backend services, API Gateway simplifies the process of building and managing APIs. Leveraging API Gateway, developers can define RESTful APIs or WebSocket APIs, configure endpoints, handle authentication and authorisation, and enforce usage policies.

*Amazon Lambda* is a serverless computing service that enables developers to run code without provisioning or managing servers. Lambda automatically scales and manages the compute resources needed to execute the code in response to incoming requests. Lambda supports various programming languages, including Node.js, Python, Java, and Go, allowing developers to write functions that respond to events triggered by AWS services or custom events.

*Amazon DynamoDB* is a fully managed NoSQL database service designed to provide high performance, scalability, and reliability for applications requiring low-latency data access.

*Amazon* *DynamoDB* offers seamless scalability with automatic partitioning and replication, ensuring consistent performance as data volumes and throughput requirements change. It also offers features such as built-in security, backup and restore capabilities, and flexible querying options.

### Interactions and Sequences

**Figure 10**   
Sequence Diagram depicting interaction between the User, Cognito, API Gateway, Lambda and DynamoDB.

A diagram of a software company

Description automatically generated

The user initiates the authentication process by providing their email address and password, which are validated against AWS Cognito. Upon successful authentication, a security token is issued to the user, allowing them to perform authenticated actions within AWS. Requests are made by providing the security token and URL to the desired REST resource, facilitated by AWS API Gateway. This service routes requests to a serverless backend implemented using AWS Lambda, where business logic is executed. Lambda functions interact with AWS DynamoDB, a NoSQL database service, to retrieve and process data. Retrieved data is transformed and subjected to further business logic within Lambda functions before being delivered back to the user's browser through AWS API Gateway, completing the transaction cycle.

**Figure 11**   
Sequence Diagram depicting interaction between the User, Cognito and S3.

A diagram of a diagram

Description automatically generated

The authentication process begins as the user submits their email address and password for validation against AWS Cognito. Upon validation, an authentication token is provided to the user, granting access to authenticated actions within AWS. Additionally, various objects, such as files, videos, and images, are uploaded to AWS Simple Storage Service (S3), an object storage solution. Conversely, files, videos, and images are retrieved from the S3 storage repository through a download process.

### Schema Design

**A screenshot of a computer

Description automatically generated**

**Table 1**  
JSON Schema for the Recruiter Profile.

|  |
| --- |
| 1. {  2. "$schema": "http://json-schema.org/draft-04/schema#",  3. "type": "object",  4. "properties": {  5. "id": {  6. "type": "string"  7. },  8. "name": {  9. "type": "string"  10. },  11. "lastName": {  12. "type": "string"  13. },  14. "organisation": {  15. "type": "string"  16. },  17. "email": {  18. "type": "string"  19. }  20. },  21. "required": [  22. "id",  23. "name",  24. "lastName",  25. "organisation",  26. "email"  27. ]  28. } |

**Table 2**  
JSON Schema for the Candidate Profile.

|  |
| --- |
| 1. {  2. "$schema": "http://json-schema.org/draft-04/schema#",  3. "type": "object",  4. "properties": {  5. "id": {  6. "type": "string"  7. },  8. "name": {  9. "type": "string"  10. },  11. "lastName": {  12. "type": "string"  13. },  14. "email": {  15. "type": "string"  16. },  17. "dob": {  18. "type": "string"  19. },  20. "videoUri": {  21. "type": "string"  22. },  23. "cvUri": {  24. "type": "string"  25. },  26. "jobId": {  27. "type": "array",  28. "items": [  29. {  30. "type": "string"  31. }  32. ]  33. }  34. },  35. "required": [  36. "id",  37. "name",  38. "lastName",  39. "email",  40. "dob",  41. "videoUri",  42. "cvUri",  43. "jobId"  44. ]  45. }  46. |

**Table 3**  
JSON Schema for Notes.

|  |
| --- |
| 1. {  2. "$schema": "http://json-schema.org/draft-04/schema#",  3. "type": "object",  4. "properties": {  5. "id": {  6. "type": "string"  7. },  8. "recruiterProfileId": {  9. "type": "string"  10. },  11. "candidateProfileId": {  12. "type": "string"  13. },  14. "note": {  15. "type": "string"  16. }  17. },  18. "required": [  19. "id",  20. "recruiterProfileId",  21. "candidateProfileId",  22. "note”  23. ]  24. }  25. |

**Table 4**  
JSON Schema for Categories.

|  |
| --- |
| 1. {  2. "$schema": "http://json-schema.org/draft-04/schema#",  3. "type": "object",  4. "properties": {  5. "id": {  6. "type": "string"  7. },  8. "description": {  9. "type": "string"  10. }  11. },  12. "required": [  13. "id",  14. "description"  15. ]  16. }  17. |

**Table 5**  
JSON Schema for Jobs.

|  |
| --- |
| 1. {  2. "$schema": "http://json-schema.org/draft-04/schema#",  3. "type": "object",  4. "properties": {  5. "id": {  6. "type": "string"  7. },  8. "title": {  9. "type": "string"  10. },  11. "description": {  12. "type": "string"  13. },  14. "salary": {  15. "type": "string"  16. },  17. "categoryId": {  18. "type": "string"  19. }  20. },  21. "required": [  22. "id",  23. "title",  24. "description",  25. "salary",  26. "categoryId"  27. ]  28. }  29. |

### REST API Endpoints.

**Table 6**   
Recruiter Profiles REST Endpoints.

|  |  |  |
| --- | --- | --- |
| Endpoint | Verb | Action |
| https://api.crisp.co.nz/recruiters/profiles | POST | Create a Profile. |
| https://api.crisp.co.nz/recruiters/profiles | GET | Get all Profiles. |
| https://api.crisp.co.nz/recruiters/profiles/{profileId} | GET | Get a Profile by ID. |
| https://api.crisp.co.nz/recruiters/profiles/ | PUT | Update a Profile. |
| https://api.crisp.co.nz/recruiters/profiles/{profileId} | DELETE | Delete a Profile by ID. |

**Table 7**   
Candidate Profiles REST Endpoints.

|  |  |  |
| --- | --- | --- |
| Endpoint | Verb | Action |
| <https://api.crisp.co.nz/jobseekers/profiles> | POST | Create a Profile. |
| <https://api.crisp.co.nz/jobseekers/profiles> | GET | Get all Profiles. |
| [https://api.crisp.co.nz/jobseekers/profiles/{profileId}](https://api.crisp.co.nz/jobseekers/profiles/%7bprofileId%7d) | GET | Get a Profile by ID. |
| <https://api.crisp.co.nz/jobseekers/profiles/> | PUT | Update a Profile. |
| [https://api.crisp.co.nz/jobseekers/profiles/{profileId}](https://api.crisp.co.nz/jobseekers/profiles/%7bprofileId%7d) | DELETE | Delete a Profile by ID. |

**Table 8**  
Notes REST Endpoints.

|  |  |  |
| --- | --- | --- |
| Endpoint | Verb | Action |
| <https://api.crisp.co.nz/notes/> | POST | Create a Note. |
| [https://api.crisp.co.nz/notes/](https://api.crisp.co.nz/notess) | GET | Get all Notes. |
| [https://api.crisp.co.nz/notes/](https://api.crisp.co.nz/notess){noteId} | GET | Get a Note by ID. |
| [https://api.crisp.co.nz/notes/](https://api.crisp.co.nz/notess) | PUT | Update a Note. |
| [https://api.crisp.co.nz/notes/](https://api.crisp.co.nz/notess){noteId} | DELETE | Delete a Note by ID. |

**Table 9**  
Categories REST Endpoints.

|  |  |  |
| --- | --- | --- |
| Endpoint | Verb | Action |
| <https://api.crisp.co.nz/categories/> | POST | Create a Category. |
| <https://api.crisp.co.nz/categories/> | GET | Get all Categories. |
| <https://api.crisp.co.nz/categories/>{categoryId} | GET | Get a Category by ID. |
| <https://api.crisp.co.nz/categories/> | PUT | Update a Category. |
| <https://api.crisp.co.nz/categories/>{categoryId} | DELETE | Delete a Category by ID. |

**Table 10**   
Jobs REST Endpoints.

|  |  |  |
| --- | --- | --- |
| Endpoint | Verb | Action |
| <https://api.crisp.co.nz/jobs/> | POST | Create a Job. |
| <https://api.crisp.co.nz/jobs/> | GET | Get all Jobs. |
| <https://api.crisp.co.nz/jobs/>{jobId} | GET | Get a Job by ID. |
| <https://api.crisp.co.nz/jobs/> | PUT | Update a Job. |
| <https://api.crisp.co.nz/jobs/>{jobId} | DELETE | Delete a Job by ID. |

### Recruiter Profile

In recruitment activities, recruiters navigate to the job posting screen, where they encounter the functionality to perform various actions, including creating, updating, or retrieving job postings. It is pertinent to acknowledge that the category associated with each job posting is selected from a predetermined drop-down list, pre-populated with available options. This mechanism ensures consistency and accuracy in categorising job postings, streamlining the process for recruiters.

In the recruitment process, a candidate accesses the profile screen, where they are presented with the functionality to perform actions such as creating a profile, updating, or retrieving an existing profile. It is imperative to emphasise that the candidate's email address is fetched from the authentication and authorisation service. Consequently, the email address remains immutable and cannot be modified, as it is a foundational identifier utilised during the sign-up process.

**Figure 12**   
Candidate Apply for a Job

A screenshot of a computer

Description automatically generated

The candidate initiates navigation to the available jobs page, where they are prompted to select a category of interest. Upon selecting a specific category, the system retrieves and displays a list of jobs corresponding to the chosen category. To view the details of a particular job, the candidate double-clicks on the respective job listing, thereby accessing comprehensive job information. Within the job detail view, the candidate is presented with the option to apply for the job with a single click, streamlining the application process and enhancing user convenience.

**Figure 13**   
Recruiter Finds and Engages with Candidates

A screenshot of a computer screen

Description automatically generated

To facilitate interaction with candidates who have applied for advertised roles, the recruiter navigates to the jobs page. Here, the recruiter selects the desired category, followed by a double-click action on the list of jobs corresponding to the specified category. Subsequently, a list of candidates associated with the selected job listings emerges. Upon double-clicking on a candidate's name, a detailed screen surfaces, presenting the recruiter with comprehensive information, including the candidate's contact details, curriculum vitae (CV), and a video profile where applicable. Moreover, the recruiter is afforded the option to annotate pertinent observations or remarks against the candidate, facilitating efficient management of candidate evaluations and feedback.

## User Manual

## Summary