React Web Project Assessment – RESTFul API

This document provides a summary of the available RESTful API endpoints .

API Overview

- **URL**: https://comp2140a2.uqcloud.net/api

Authentication

Anonymous access is disabled. Authentication via an JSON Web Token (JWT) is required for all endpoints. The JWT is provided via a column in Blackboard My Grades. Example API calling code is provided to get you started.

Interview Endpoint

The `/interview` endpoint allows management of interview resources.

Interview Fields

Field Name	Type	Size	Required	Description	Example Value
id	Integer	-	Yes (PK)	Auto-generated identifier	1
title	String	255	Yes	Title of the interview	"Front-end Developer Interview"
job_role	String	255	Yes	Role being interviewed for	"Senior Front-end Developer"
description	Text	-	No	Detailed description	"Interview for candidates with React experience"
status	String	50	Yes	Publication status	"Published", "Draft", or "Archived"
username	String	255	Yes	Owner of the interview	"s10000" (This is your student id)

Interview Endpoints

Method	Path	Description	Example
GET	/interview	List all interviews	`GET /interview`
POST	/interview	Create a new interview	`POST /interview` with JSON body
PATCH	/interview	Update interview(s)	`PATCH /interview?id=eq.1` with JSON body
DELETE	/interview	Delete interview(s)	`DELETE /interview?id=eq.1`

Question Endpoint

The `/question` endpoint manages interview questions.

Question Fields

Field Name	Type	Size	Required	Description	Example Value
id	Integer	-	Yes (PK)	Auto-generated identifier	1
interview_id	Integer	-	Yes (FK)	Reference to interview.id	1
question	Text	-	Yes	The question text	"Explain the difference between let and const in JavaScript"
difficulty	String	20	Yes	Question difficulty level	"Easy", "Intermediate", or "Advanced"
username	String	255	Yes	Owner of the question	"s100000" (This is your student id)

Question Endpoints

Method	Path	Description	Example
GET	/question	List all questions	`GET /question?interview_id=eq.1`
POST	/question	Create a new question	`POST /question` with JSON body
PATCH	/question	Update question(s)	`PATCH /question?id=eq.1` with JSON body
DELETE	/question	Delete question(s)	`DELETE /question?id=eq.1`

Applicant Endpoint

The `/applicant` endpoint manages interview applicants.

Applicant Fields

Field Name	Type	Size	Required	Description	Example Value
id	Integer	-	Yes (PK)	Auto-generated identifier	1
interview_id	Integer	-	Yes (FK)	Reference to interview.id	1
title	String	20	Yes	Applicant's title	"Mr", "Ms", "Dr"
firstname	String	255	Yes	Applicant's first name	"John"
surname	String	255	Yes	Applicant's surname	"Smith"
phone_number	String	50	No	Contact phone number	"+61 412 345 678"
email_address	String	255	Yes	Contact email address	"john.smith@example.com"
interview_status	String	50	Yes	Status of interview	"Not Started", "Completed"
username	String	255	Yes	Owner of the applicant record	" s100000"(This is your student id)

Applicant Endpoints

Method	Path	Description	Example
GET	/applicant	List all applicants	`GET /applicant?interview_id=eq.1`
POST	/applicant	Create a new applicant	`POST /applicant` with JSON body
PATCH	/applicant	Update applicant(s)	`PATCH /applicant?id=eq.1` with JSON body
DELETE	/applicant	Delete applicant(s)	`DELETE /applicant?id=eq.1`

Applicant Answer Endpoint

The `/applicant_answer` endpoint manages responses to interview questions.

Applicant Answer Fields

Field Name	Type	Size	Required	Description	Example Value
id	Integer	-	Yes (PK)	Auto-generated identifier	1
interview_id	Integer	-	Yes (FK)	Reference to interview.id	1
question_id	Integer	-	Yes (FK)	Reference to question.id	2
applicant_id	Integer	-	Yes (FK)	Reference to applicant.id	3
answer	Text	-	No	Applicant's response	"I would use const for values that shouldn't change, and let for variables that will be reassigned."
username	String	255	Yes	Owner of the answer record	"s100000" (This is your student id)

Applicant Answer Endpoints

Method	Path	Description	Example
GET	/applicant_answer	List all answers	`GET /applicant_answer?applicant_id=eq.3`
POST	/applicant_answer	Create a new answer	`POST /applicant_answer` with JSON body
PATCH	/applicant_answer	Update answer(s)	`PATCH /applicant_answer?id=eq.1` with JSON body
DELETE	/applicant_answer	Delete answer(s)	`DELETE /applicant_answer?id=eq.1`

Query Parameters

PostgREST supports various query parameters for filtering, ordering, and pagination:

Common Query Parameters

Parameter	Description	Example
select	Select specific columns	`?select=id,title,status`
order	Order results	`?order=id.desc`
limit	Limit number of results	`?limit=10`
offset	Pagination offset	`?offset=20`
eq	Equal to	`?id=eq.1`
gt	Greater than	`?id=gt.5`
lt	Less than	`?id=lt.10`
gte	Greater than or equal	`?id=gte.5`
lte	Less than or equal	`?id=lte.10`
like	LIKE operator	`?title=like.*Dev*`
ilike	Case-insensitive LIKE	`?title=ilike.*dev*`

Header Parameters

Header	Description	Example
Prefer: return=representation	Return the created/modified resource	`Prefer: return=representation`
Prefer: return=minimal	Return minimal response	`Prefer: return=minimal`
Range	Pagination range	`Range: 0-9`