ID608001: Intermediate Application Development Concepts

Assessment 1: Node.js RESTful API - Open Trivia DB Marking Rubric

	10-9	8-7	6-5	4-0
Functionality	The RESTful API contains comprehensive and robust evidence on the following functionality: User, Quiz, HTTP, Testing, Deployment and NPM Scripts.	The RESTful API contains clear and detailed evidence on the following functionality: User, Quiz, HTTP, Testing, Deployment and NPM Scripts.	The RESTful API contains evidence on the following functionality: User, Quiz, HTTP, Testing, Deployment and NPM Scripts.	The RESTful API does not or does not fully contain evidence on the following functionality: User, Quiz, HTTP, Testing, Deployment and NPM Scripts.
Code Elegance	The RESTful API demonstrates comprehensive evidence on the following: • Environment variables' key is stored in the env.example file. • PostgreSQL databases configured. • Variables, functions and resource groups are named appropriately. • Idiomatic use of control flow, data structures and in-built functions. • File header comment for each controller and route file. • Code is linted and formatted. • Pre-commit hook using Husky. • Mocha, Chai, c8, ESLint, Prettier, Husky and Commitizen are installed as development dependencies.	The RESTful API demonstrates clear evidence on the following: • Environment variables' key is stored in the env.example file. • PostgreSQL databases configured. • Variables, functions and resource groups are named appropriately. • Idiomatic use of control flow, data structures and in-built functions. • File header comment for each controller and route file. • Code is linted and formatted. • Pre-commit hook using Husky. • Mocha, Chai, c8, ESLint, Prettier, Husky and Commitizen are installed as development dependencies.	The RESTful API demonstrates evidence on the following: • Environment variables' key is stored in the env.example file. • PostgreSQL databases configured. • Variables, functions and resource groups are named appropriately. • Idiomatic use of control flow, data structures and in-built functions. • File header comment for each controller and route file. • Code is linted and formatted. • Pre-commit hook using Husky. • Mocha, Chai, c8, ESLint, Prettier, Husky and Commitizen are installed as development dependencies.	The RESTful API does not or does not fully demonstrate evidence on the following: • Environment variables' key is stored in the env.example file. • PostgreSQL databases configured. • Variables, functions and resource groups are named appropriately. • Idiomatic use of control flow, data structures and in-built functions. • File header comment for each controller and route file. • Code is linted and formatted. • Pre-commit hook using Husky. • Mocha, Chai, c8, ESLint, Prettier, Husky and Commitizen are installed as development dependencies.

Documentation & Git Usage	Comprehensive use of project board on GitHub. README file contains comprehensive evidence on the following: • An ERD of your Prisma schema. • Deployment to Heroku. • Setup the development environment. • Open Prisma Studio. • Create a migration. • Lint and fix code. • Format code. • Run API/integration tests. • Code coverage and output the results to HTML. • Use of Markdown. • Spelling and grammar correctness. Git commit messages comprehensively formatted using Commitizen and reflect the changes in concise detail.	Clear use of project board on GitHub. README file contains clear evidence of: • An ERD of your Prisma schema. • Deployment to Heroku. • Setup the development environment. • Open Prisma Studio. • Create a migration. • Lint and fix code. • Format code. • Run API/integration tests. • Code coverage and output the results to HTML. • Use of Markdown. • Spelling and grammar correctness. Git commit messages clearly formatted using Commitizen and reflect the changes in substantial detail.	Use of project board on GitHub. README file contains evidence of: • An ERD of your Prisma schema. • Deployment to Heroku. • Setup the development environment. • Open Prisma Studio. • Create a migration. • Lint and fix code. • Format code. • Run API/integration tests. • Code coverage and output the results to HTML. • Use of Markdown. • Spelling and grammar correctness. Git commit messages formatted using Commitizen and reflect the changes in detail.	Does not or does not full demonstrate use of project board on GitHub. README file does not or does not fully contain evidence of: • An ERD of your Prisma schema. • Deployment to Heroku. • Setup the development environment. • Open Prisma Studio. • Create a migration. • Lint and fix code. • Format code. • Run API/integration tests. • Code coverage and output the results to HTML. • Use of Markdown. • Spelling and grammar correctness. Git commit messages are not or are not fully formatted using Commitizen and do not or do not fully reflect the changes.

ID608001: Intermediate Application Development Concepts

Assessment 1: Node.js RESTful API - Open Trivia Marking Cover Sheet

Name:

Date:

Learner ID:

Assessor's Name:

Assessor's Signature:

Criteria	Out Of	Weighting	Final Result
Functionality	10	50	
Code Elegance	10	35	
Documentation & Git Usage	10	15	
Final Result			/100
This assessment is worth 45% of the final mark for the Intermediate Application Development Concepts course.			

Feedback:

Functionality:

Code Elegance:

Documentation & Git Usage: