# ID607001: Introductory Application Development Concepts

# Project 1: Node.js REST API Assessment Rubric

10-9	8-7	6-5	4-0
REST API contains comprehensive & robe evidence on the following:  REST API is developed using Node.js & can run locally with modification.  An appropriate number of collections & fields with difference data types.  Separate controller & route field each collection.  Custom validation when creat & updating a field.  Collections are seeded with a file.  REST API version is v1.  Appropriate status code & message returned when performing CRUD operations query does not return any AP data & if an endpoint does not exist.  Filter, sort & paginate REST A data.  POST, PUT & DELETE routes a protected.  Rate limit is 25 requests per minute.  REST API data is stored in a MongoDB Atlas database.	of functionality on the following:  REST API is developed using Node.js & can run locally without modification.  An appropriate number of collections & fields with different data types.  Separate controller & route file for each collection.  Custom validation when creating & updating a field.  Collections are seeded with a JSON file.  REST API version is v1.  Appropriate status code & message returned when performing CRUD operations if a query does not return any API data & if an endpoint does not exist.  Filter, sort & paginate REST API data.  POST, PUT & DELETE routes are protected.  Rate limit is 25 requests per minute.  REST API is deployed to Heroku.	<ul> <li>REST API contains evidence on the following:         <ul> <li>REST API is developed using Node.js &amp; can run locally without modification.</li> <li>An appropriate number of collections &amp; fields with different data types.</li> <li>Separate controller &amp; route file for each collection.</li> <li>Custom validation when creating &amp; updating a field.</li> <li>Collections are seeded with a JSON file.</li> <li>REST API version is v1.</li> <li>Appropriate status code &amp; message returned when performing CRUD operations if a query does not return any API data &amp; if an endpoint does not exist.</li> <li>Filter, sort &amp; paginate REST API data.</li> <li>POST, PUT &amp; DELETE routes are protected.</li> <li>Rate limit is 25 requests per minute.</li> <li>REST API data is stored in a MongoDB Atlas database.</li> </ul> </li> </ul>	REST API does not, or does not fully contain evidence on the following:  REST API is developed using Node.js & can run locally without modification.  An appropriate number of collections & fields with different data types.  Separate controller & route file for each collection.  Custom validation when creating & updating a field.  Collections are seeded with a JSON file.  REST API version is v1.  Appropriate status code & message returned when performing CRUD operations if a query does not return any API data & if an endpoint does not exist.  Filter, sort & paginate REST API data.  POST, PUT & DELETE routes are protected.  Rate limit is 25 requests per minute.  REST API data is stored in a MongoDB Atlas database.

ID607001: Introductory Application Development Concepts

Project 1: Node.js REST API Version 1, Semester One, 2022

## REST API thoroughly demonstrates code REST API clearly demonstrates code elegance REST API demonstrates code elegance on the REST API does not or does not fully elegance on the following: on the following: following: demonstrate code elegance on the following: Intermediate variables, idiomatic Intermediate variables, idiomatic Intermediate variables, idiomatic • Intermediate variables, idiomatic control flow, data structures & incontrol flow, data structures & incontrol flow, data structures & incontrol flow, data structures & inbuilt functions, & sufficient built functions, & sufficient built functions, & sufficient built functions, & sufficient modularity. modularity. modularity. modularity. Elegance Functions & variables are named appropriately. appropriately. appropriately. appropriately. Efficient algorithmic approach. Efficient algorithmic approach. Efficient algorithmic approach. Efficient algorithmic approach. REST API groups are named with a Code plural. Filer header & in-line comments. Formatted code using Prettier. Formatted code using Prettier. Formatted code using Prettier. Formatted code using Prettier. Prettier installed as a dev dependency. dependency. dependency. dependency. No dead or unused code. Database configured for Database configured for Database configured for Database configured for production environment. production environment. production environment. production environment. REST API documented in succinct detail using REST API documented in substantial detail REST API documented in detail using REST API not or not fully documented in Postman. using Postman. Postman. detail using Postman. README file contains thorough evidence of: README file contains clear evidence of: README file contains evidence of: README file does not or does not fully Git Usage contain evidence of: URL to the REST API on Heroku. URL to the REST API URL to the REST API URL to the REST API documentation on Postman. documentation on Postman. documentation on Postman. URL to the REST API How to setup the environment for How to setup the environment for How to setup the environment for documentation on Postman. **Documentation &** development & deploy the REST development & deploy the REST development & deploy the REST How to setup the environment for API. development & deploy the REST API. API. API. Git branches are thoroughly named with Git branches are mostly named with Some git branches are named with convention & contain the correct code convention & contain the correct code convention & contain the correct code Git branches are not or are not fully named relating to the functional requirement. with convention & do not or do not fully relating to the functional requirement. relating to the functional requirement. contain the correct code relating to the Git commit messages are comprehensively Git commit messages are clearly formatted & Git commit messages are formatted & reflect functional requirement. formatted & reflect the functionality changes reflect the functionality changes in the functionality changes in detail. in succinct detail. substantial detail. Git commit messages are not or are not fully formatted & do not or do not reflect the functionality changes.

ID607001: Introductory Application Development Concepts

Project 1: Node.js REST API Version 1, Semester One, 2022

## ID607001: Introductory Application Development Concepts

# Project 1: Node.js REST API Marking Cover Sheet

Name:					
Date:					
Learner ID:					
Assessor's Name:					
Assessor's Signature:					
Criteria	Out Of	Weighting	Final Result		
Functionality	10	40			
Code Elegance	10	45			
Documentation & Git Usage	10	15			
	/100				
This assessment is worth 30% of the final mark for the Introductory Application Development Concepts course.					
Feedback:					
Functionality:					
Code Elegance:					
Documentation & Git Usage:					