

## ID607001: Introductory Application Development Concepts

### Project 1: Node.js REST API Assessment Rubric

	10-9	8-7	6-5	4-0
Functionality	<p>REST API contains comprehensive &amp; robust evidence on the following:</p> <ul style="list-style-type: none"> <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>Relationships between collections.</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>HTTP headers secured.</li> <li>REST API is deployed to Heroku.</li> <li>REST API data is stored in a MongoDB Atlas database.</li> </ul>	<p>REST API contains clear &amp; detailed evidence of functionality on the following:</p> <ul style="list-style-type: none"> <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>Relationships between collections.</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>HTTP headers secured.</li> <li>REST API is deployed to Heroku.</li> <li>REST API data is stored in a MongoDB Atlas database.</li> </ul>	<p>REST API contains evidence on the following:</p> <ul style="list-style-type: none"> <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>Relationships between collections.</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>HTTP headers secured.</li> <li>REST API is deployed to Heroku.</li> <li>REST API data is stored in a MongoDB Atlas database.</li> </ul>	<p>REST API does not, or does not fully contain evidence on the following:</p> <ul style="list-style-type: none"> <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>Relationships between collections.</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>HTTP headers secured.</li> <li>REST API is deployed to Heroku.</li> <li>REST API data is stored in a MongoDB Atlas database.</li> </ul>

<b>Code Elegance</b>	<p>REST API thoroughly demonstrates code elegance on the following:</p> <ul style="list-style-type: none"> <li>• Intermediate variables, idiomatic control flow, data structures &amp; in-built functions, &amp; sufficient modularity.</li> <li>• Functions &amp; variables are named appropriately.</li> <li>• Efficient algorithmic approach.</li> <li>• REST API groups are named with a plural.</li> <li>• Filer header &amp; in-line comments.</li> <li>• Formatted code using Prettier.</li> <li>• Prettier installed as a dev dependency.</li> <li>• No dead or unused code.</li> <li>• Database configured for production environment.</li> <li>• Environment variables stored.</li> </ul>	<p>REST API clearly demonstrates code elegance on the following:</p> <ul style="list-style-type: none"> <li>• Intermediate variables, idiomatic control flow, data structures &amp; in-built functions, &amp; sufficient modularity.</li> <li>• Functions &amp; variables are named appropriately.</li> <li>• Efficient algorithmic approach.</li> <li>• REST API groups are named with a plural.</li> <li>• Filer header &amp; in-line comments.</li> <li>• Formatted code using Prettier.</li> <li>• Prettier installed as a dev dependency.</li> <li>• No dead or unused code.</li> <li>• Database configured for production environment.</li> <li>• Environment variables stored.</li> </ul>	<p>REST API demonstrates code elegance on the following:</p> <ul style="list-style-type: none"> <li>• Intermediate variables, idiomatic control flow, data structures &amp; in-built functions, &amp; sufficient modularity.</li> <li>• Functions &amp; variables are named appropriately.</li> <li>• Efficient algorithmic approach.</li> <li>• REST API groups are named with a plural.</li> <li>• Filer header &amp; in-line comments.</li> <li>• Formatted code using Prettier.</li> <li>• Prettier installed as a dev dependency.</li> <li>• No dead or unused code.</li> <li>• Database configured for production environment.</li> <li>• Environment variables stored.</li> </ul>	<p>REST API does not or does not fully demonstrate code elegance on the following:</p> <ul style="list-style-type: none"> <li>• Intermediate variables, idiomatic control flow, data structures &amp; in-built functions, &amp; sufficient modularity.</li> <li>• Functions &amp; variables are named appropriately.</li> <li>• Efficient algorithmic approach.</li> <li>• REST API groups are named with a plural.</li> <li>• Filer header &amp; in-line comments.</li> <li>• Formatted code using Prettier.</li> <li>• Prettier installed as a dev dependency.</li> <li>• No dead or unused code.</li> <li>• Database configured for production environment.</li> <li>• Environment variables stored.</li> </ul>
<b>Documentation &amp; Git Usage</b>	<p>REST API documented in succinct detail using Postman.</p> <p>README file contains thorough evidence of:</p> <ul style="list-style-type: none"> <li>• URL to the REST API on Heroku.</li> <li>• URL to the REST API documentation on Postman.</li> <li>• How to setup the environment for development &amp; deploy the REST API.</li> </ul> <p>Git commit messages are comprehensively formatted &amp; reflect the functionality changes in succinct detail.</p>	<p>REST API documented in substantial detail using Postman.</p> <p>README file contains clear evidence of:</p> <ul style="list-style-type: none"> <li>• URL to the REST API on Heroku.</li> <li>• URL to the REST API documentation on Postman.</li> <li>• How to setup the environment for development &amp; deploy the REST API.</li> </ul> <p>Git commit messages are clearly formatted &amp; reflect the functionality changes in substantial detail.</p>	<p>REST API documented in detail using Postman.</p> <p>README file contains evidence of:</p> <ul style="list-style-type: none"> <li>• URL to the REST API on Heroku.</li> <li>• URL to the REST API documentation on Postman.</li> <li>• How to setup the environment for development &amp; deploy the REST API.</li> </ul> <p>Git commit messages are formatted &amp; reflect the functionality changes in detail.</p>	<p>REST API not or not fully documented in detail using Postman.</p> <p>README file does not or does not fully contain evidence of:</p> <ul style="list-style-type: none"> <li>• URL to the REST API on Heroku.</li> <li>• URL to the REST API documentation on Postman.</li> <li>• How to setup the environment for development &amp; deploy the REST API.</li> </ul> <p>Git commit messages are not or are not fully formatted &amp; do not or do not reflect the functionality changes.</p>

## 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	
Final Result			/100
This assessment is worth 30% of the final mark for the Introductory Application Development Concepts course.			

**Feedback:**

**Functionality:**

**Code Elegance:**

**Documentation & Git Usage:**