# **ID607001: Introductory Application Development Concepts**

# **Project Marking Rubric**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **10-9** | **8-7** | **6-5** | **4-0** |
| **Functionality – Your choice REST API** | The REST API developed in Node.js contains comprehensive and robust evidence on the following functionality: development and production modification, models, data types, relationships, enum, files, messages, filtering, sorting, pagination, 404 endpoint, validation, Swagger documentation, PostgreSQL database and deployment. | The REST API developed in Node.js contains comprehensive and robust evidence on the following functionality: development and production modification, models, data types, relationships, enum, files, messages, filtering, sorting, pagination, 404 endpoint, validation, Swagger documentation, PostgreSQL database and deployment. | The REST API developed in Node.js contains comprehensive and robust evidence on the following functionality: development and production modification, models, data types, relationships, enum, files, messages, filtering, sorting, pagination, 404 endpoint, validation, Swagger documentation, PostgreSQL database and deployment. | The REST API developed in Node.js contains comprehensive and robust evidence on the following functionality: development and production modification, models, data types, relationships, enum, files, messages, filtering, sorting, pagination, 404 endpoint, validation, Swagger documentation, PostgreSQL database and deployment. |
|  | **10-9** | **8-7** | **6-5** | **4-0** |
| **Functionality – OpenTDB API** | The REST API developed in Node.js contains comprehensive and robust evidence on the following functionality: development and production modification, enums, models, admin user, basic user, validation, seeding, Helmet, CORS, rate limiting, compression, Swagger documentation, PostgreSQL database and deployment. | The REST API developed in Node.js contains clear and detailed evidence on the following functionality: development and production modification, enums, models, admin user, basic user, validation, seeding, Helmet, CORS, rate limiting, compression, Swagger documentation, PostgreSQL database and deployment. | The REST API developed in Node.js contains evidence on the following functionality: development and production modification, enums, models, admin user, basic user, validation, seeding, Helmet, CORS, rate limiting, compression, Swagger documentation, PostgreSQL database and deployment. | The REST API developed in Node.js does not or does not fully contain evidence on the following functionality: development and production modification, enums, models, admin user, basic user, validation, seeding, Helmet, CORS, rate limiting, compression, Swagger documentation, PostgreSQL database and deployment. |
| **Documentation and Git Usage** | Comprehensive use of project board on GitHub.  README file contains comprehensive evidence on the following:   * A URL to your REST API as web service on Render. * A URL to your published REST API documentation. * An ERD of your database. * Setup the environment. * Run your REST API locally. * Create and apply a migration. * Reset your database. * Open Prisma Studio. * Format your code. * Use of Markdown. * Spelling and grammar correctness.   Git commit messages comprehensively formatted and reflect the changes in concise detail. | Clear use of project board on GitHub.  README file contains clear evidence of:   * A URL to your REST API as web service on Render. * A URL to your published REST API documentation. * An ERD of your database. * Setup the environment. * Run your REST API locally. * Create and apply a migration. * Reset your database. * Open Prisma Studio. * Format your code. * Use of Markdown. * Spelling and grammar correctness.   Git commit messages clearly formatted and reflect the changes in substantial detail. | Use of project board on GitHub.  README file contains evidence of:   * A URL to your REST API as web service on Render. * A URL to your published REST API documentation. * An ERD of your database. * Setup the environment. * Run your REST API locally. * Create and apply a migration. * Reset your database. * Open Prisma Studio. * Format your code. * Use of Markdown. * Spelling and grammar correctness.   Git commit messages formatted 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:   * A URL to your REST API as web service on Render. * A URL to your published REST API documentation. * An ERD of your database. * Setup the environment. * Run your REST API locally. * Create and apply a migration. * Reset your database. * Open Prisma Studio. * Format your code. * Use of Markdown. * Spelling and grammar correctness.   Git commit messages are not or are not fully formatted and do not or do not fully reflect the changes. |

# **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 | 50 |  |
| Code Elegance | 10 | 40 |  |
| Documentation & Git Usage | 10 | 10 |  |
| **Final Result** | | | /100 |
| **This assessment is worth 40% of the final mark for the Introductory Application Development Concepts course.** | | | |

**Feedback:**

**Functionality:**

**Code Elegance:**

**Documentation & Git Usage:**