ID607001: Introductory Application Development Concepts

Project 2: React CRUD Assessment Rubric

	10-9	8-7	6-5	4-0
Functionality	Application demonstrates comprehensive & robust evidence on the following: • Register, login & logout. • Requesting REST API data using Axios. • View REST API data. • Update & delete REST API data. • Incorrectly formatted form fields are handled. • Paginate & search REST API data. • UI designed with Reactstrap. • Deployed to Heroku. • End-to-end that ensures authentication is working as expected.	Application demonstrates clear & detailed evidence on the following: • Register, login & logout. • Requesting REST API data using Axios. • View REST API data. • Update & delete REST API data. • Incorrectly formatted form fields are handled. • Paginate & search REST API data. • UI designed with Reactstrap. • Deployed to Heroku. • End-to-end that ensures authentication is working as expected.	Application demonstrates evidence on the following: • Register, login & logout. • Requesting REST API data using Axios. • View REST API data. • Update & delete REST API data. • Incorrectly formatted form fields are handled. • Paginate & search REST API data. • UI designed with Reactstrap. • Deployed to Heroku. • End-to-end that ensures authentication is working as expected.	Application demonstrates does not, or does not fully demonstrate evidence on the following: • Register, login & logout. • Requesting REST API data using Axios. • View REST API data. • Update & delete REST API data. • Incorrectly formatted form fields are handled. • Paginate & search REST API data. • UI designed with Reactstrap. • Deployed to Heroku. • End-to-end that ensures authentication is working as expected.

Code Elegance	Application thoroughly demonstrate code elegance on the following: • Intermediate variables, idiomatic control flow, data structures & in-built functions, & sufficient modularity. • Functions & variables are named appropriately. • Components are written as functional. • Adheres to a client-server architecture. • Filer header & in-line comments. • Formatted code using Prettier. • Prettier & Cypress are installed as dev dependencies. • No dead or unused code.	Application clearly demonstrate code elegance on the following: • Intermediate variables, idiomatic control flow, data structures & in-built functions, & sufficient modularity. • Functions & variables are named appropriately. • Components are written as functional. • Adheres to a client-server architecture. • Filer header & in-line comments. • Formatted code using Prettier. • Prettier & Cypress are installed as dev dependencies. • No dead or unused code.	Application demonstrate code elegance on the following: • Intermediate variables, idiomatic control flow, data structures & in-built functions, & sufficient modularity. • Functions & variables are named appropriately. • Components are written as functional. • Adheres to a client-server architecture. • Filer header & in-line comments. • Formatted code using Prettier. • Prettier & Cypress are installed as dev dependencies. • No dead or unused code.	Application does not or does not fully demonstrate code elegance on the following: • Intermediate variables, idiomatic control flow, data structures & in-built functions, & sufficient modularity. • Functions & variables are named appropriately. • Components are written as functional. • Adheres to a client-server architecture. • Filer header & in-line comments. • Formatted code using Prettier. • Prettier & Cypress are installed as dev dependencies. • No dead or unused code.
Documentation & Git Usage	README file contains thorough evidence of: • URL to the application on Heroku. • How to setup the environment for development, run end-to-end Cypress tests & deploy the application. Comprehensive use of Markdown syntax, i.e., headings, bold text & code blocks. Thorough spelling & grammar correctness. Git commit messages are comprehensively formatted & reflect the functionality changes in succinct detail.	README file contains clear evidence of: • URL to the application on Heroku. • How to setup the environment for development, run end-to-end Cypress tests & deploy the application. Substantial use of Markdown syntax, i.e., headings, bold text & code blocks. Clear spelling & grammar correctness. Git commit messages are clearly formatted & reflect the functionality changes in substantial detail.	README file contains evidence of: • URL to the application on Heroku. • How to setup the environment for development, run end-to-end Cypress tests & deploy the application. Use of Markdown syntax, i.e., headings, bold text & code blocks. Spelling & grammar correctness. Git commit messages are formatted & reflect the functionality changes in detail.	README file does not or does not fully contain evidence of: • URL to the application on Heroku. • How to setup the environment for development, run end-to-end Cypress tests & deploy the application. Does not or does not fully demonstrate use of Markdown syntax, i.e., headings, bold text & code blocks. Does not or does fully demonstrate spelling & grammar correctness. 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 2: React CRUD 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 50% of the final mark for the Introductory Application Development Concepts course.			

Feedback:

Functionality:

Code Elegance:

Documentation & Git Usage: