Project 2: React CRUD Assessment Rubric

	10-9	8-7	6-5	4-0
Functionality	Applications demonstrate comprehensive & robust evidence on the following:	Applications demonstrate clear & detailed evidence on the following: API data requested from API resource groups using Axios. Create, update & delete API data via modal. View API data in a table using an id and a variety of query parameters. Incorrectly formatted form fields are handled using validation error messages. Data automatically re-rendered after creating, updating and deleting. API data paginated across several pages. UI styled with Reactstrap. Application deployed to Heroku. Components tested using React Testing Library.	Applications demonstrate evidence on the following: API data requested from API resource groups using Axios. Create, update & delete API data via modal. View API data in a table using an id and a variety of query parameters. Incorrectly formatted form fields are handled using validation error messages. Data automatically re-rendered after creating, updating and deleting. API data paginated across several pages. UI styled with Reactstrap. Application deployed to Heroku. Components tested using React Testing Library.	Applications does not, or does not fully demonstrate evidence on the following: API data requested from API resource groups using Axios. Create, update & delete API data via modal. View API data in a table using an id and a variety of query parameters. Incorrectly formatted form fields are handled using validation error messages. Data automatically re-rendered after creating, updating and deleting. API data paginated across several pages. UI styled with Reactstrap. Application deployed to Heroku. Components tested using React Testing Library.

	I		I	T A 12 12 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
	Applications thoroughly demonstrate code	Applications clearly demonstrate code	Applications demonstrate code elegance	Applications does not or does not fully
	elegance on the following:	elegance on the following:	on the following:	demonstrate code elegance on the
	Appropriate use of control flow,	 Appropriate use of control flow, 	Appropriate use of control flow,	following:
	data structures and in-built	data structures and in-built	data structures and in-built	 Appropriate use of control flow,
a)	functions.	functions.	functions.	data structures and in-built
Elegance	 Sufficient code modularity. 	 Sufficient code modularity. 	 Sufficient code modularity. 	functions.
	 Components written as 	 Components written as 	 Components written as 	 Sufficient code modularity.
	functional, not class.	functional, not class.	functional, not class.	 Components written as
	 Adheres to client-server 	 Adheres to client-server 	 Adheres to client-server 	functional, not class.
Code	architecture.	architecture.	architecture.	 Adheres to client-server
Ö	 Header & in-line comments 	 Header & in-line comments 	 Header & in-line comments 	architecture.
	explain complex logic.	explain complex logic.	explain complex logic.	 Header & in-line comments
	 Formatted code using Prettier & 	 Formatted code using Prettier & 	 Formatted code using Prettier & 	explain complex logic.
	npm script.	npm script.	npm script.	 Formatted code using Prettier &
	No dead or unused code.	No dead or unused code.	No dead or unused code.	npm script.
				No dead or unused code.
	README file contains thorough evidence	README file contains clear evidence of:	README file contains evidence of:	README file does not or does not fully
e e	of:	 URL to application on Heroku. 	 URL to application on Heroku. 	contain evidence of:
sage	 URL to application on Heroku. 	 How to setup the environment 	How to setup the environment	 URL to application on Heroku.
	 How to setup the environment 	for development & deploy the	for development & deploy the	 How to setup the environment
Git	for development & deploy the	application.	application.	for development & deploy the
8	application.			application.
		Git branches are mostly named with	Some git branches are named with	
. <u>ō</u>	Git branches are thoroughly named with	convention & contain the correct code	convention & contain the correct code	Git branches are not or are not fully
Documentation	convention & contain the correct code	relating to the functional requirement.	relating to the functional requirement.	named with convention & do not or do not
Ę	relating to the functional requirement.			fully contain the correct code relating to
J e		Git commit messages are clearly	Git commit messages are formatted &	the functional requirement.
	Git commit messages are comprehensively	formatted & reflect the functionality	reflect the functionality changes in detail.	
0	formatted & reflect the functionality	changes in substantial detail.		Git commit messages are not or are not
	changes in succinct detail.			fully formatted & do not or do not reflect
				the functionality changes.

Project 2: React CRUD App Marking Cover Sheet

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 50% of the final mark for the Introductory Application Development								
	Concepts course.							
Feedback:								
Functionality:								
Code Elegance:								

Documentation & Git Usage:

Name: