Royal University of Bhutan

འབྲུག་རྒྱལ་འཛིན་གཙུག་ལག་སློབ་སྡེ།།

College of Science and Technology
Rinchending: Bhutan

RUB Wheel of Academic Law: Academic Dishonesty

Section H2 of the Royal University of Bhutan's *Wheel of Academic Law* provides the following definition of academic dishonesty:

Academic dishonesty may be defined as any attempt by a student to gain an unfair advantage in any assessment. It may be demonstrated by one of the following:

1. **Collusion:** the representation of a piece of unauthorized group work as the work of a single candidate.
2. **Commissioning:** submitting an assignment done by another person as the student's own work.
3. **Duplication**: the inclusion in coursework of material identical or substantially similar to material which has already been submitted for any other assessment within the University.
4. **False declaration**: making a false declaration in order to receive special consideration by an Examination Board or to obtain extensions to deadlines or exemption from work.
5. **Falsification of data**: presentation of data in laboratory reports, projects, etc., based on work purported to have been carried out by the student, which has been invented, altered or copied by the student.
6. **Plagiarism**: the unacknowledged use of another's work as if it were one's own.

Examples are:

- verbatim copying of another's work without acknowledgement.
- paraphrasing of another's work by simply changing a few words or altering the order of presentation, without acknowledgement.
- ideas or intellectual data in any form presented as one's own without acknowledging the source(s).
- making significant use of unattributed digital images such as graphs, tables, photographs, etc. taken from test books, articles, films, plays, handouts, the internet, or any other source, whether published or unpublished.
- submission of a piece of work which has previously been assessed for a different award or module or at a different institution as if it were new work.
- use of any material without prior permission of copyright from the appropriate authority or owner of the materials used".

Royal University of Bhutan

འབྲུག་རྒྱལ་འཛིན་གཙུག་ལག་སློབ་སྡེ།།

College of Science and Technology
Rinchending: Bhutan

Practical Assignment 2 - Building a Client-side webpage with React

Introduction

For this assignment, you will be tasked with developing a Pokedex webpage using the React framework. The Pokedex webpage will serve as a comprehensive resource for Pokemon enthusiasts, allowing them to search for and retrieve detailed information about their favorite Pokemon, as well as keep track of the Pokemon they have "caught."

The primary objective of this assignment is to demonstrate your proficiency in building modern web applications using React, a popular and widely-adopted JavaScript library for creating user interfaces. You will be expected to leverage React's component-based architecture, state management techniques, and API integration to create a fully functional and interactive Pokedex.

Requirements

1. Implement a *search feature* that allows users to search for Pokemon by name or other relevant criteria. This will involve making API calls to a Pokemon-related API, such as the official PokeAPI, to fetch data about the searched Pokemon.

You are strictly allowed to only use the fetch API. Failure to comply will result in an immediate failure of assignment.

2. When a user searches for a Pokemon, your webpage should display the relevant details of the searched Pokemon. This may include information such as the Pokemon's name, type, abilities, stats, and an image. On a new page or modal?

You are strictly required to only use the API to request another response AND NOT query a big set of data and filter for the results on the client side. Failure to comply will result in an immediate failure of assignment.

3. Implement a feature that allows users to "catch" Pokemon and keep track of the Pokemon they have caught. This functionality should utilize state management to store and manage the list of caught Pokemon. For managing the state of caught Pokemon, you will be using the Zustand library. Zustand is a lightweight state management library for React that provides a simple and efficient way to manage application state. **Please refer to outcome number 5, 6 & 7.**

You are strictly allowed to only use Zustand for state managment. (https://github.com/pmndrs/zustand) Failure to comply will result in an immediate failure of assignment.

4. Your webpage should be *built using React's component-based architecture*, where different functionalities and UI elements are separated into reusable components. This promotes code modularity, maintainability, and reusability.

You are strictly suppose to implement single responsibility design in your components. Recall the lessons where components that does 1 task are a file and data is passed down to them through attributes and received as props.

5. To fetch Pokemon data, you will need to *integrate with a Pokemon API*, such as the official PokeAPI (https://pokeapi.co/). This will involve making API calls to retrieve data about specific Pokemon or lists of Pokemon.

You are strictly allowed to only use the fetch API. Failure to comply will result in an immediate failure of assignment.

You are strictly allowed to only use the following UI component library.(https://ui.shadcn.com/) Failure to comply will result in an immediate failure of assignment.

Expected Outcomes

1. A fully operational Pokedex webpage built using React, showcasing your mastery of the framework's core concepts and principles.
2. Implementation of a search feature that allows users to search for Pokemon by name or other relevant criteria with pagination.
3. Successful integration with a Pokemon API (e.g., PokeAPI) to fetch data about the searched Pokemon.
4. Proper rendering and display of detailed information about the searched Pokemon, including its name, type, abilities, stats, and an image.
5. Implementation of a feature that allows users to "catch" Pokemon and maintain a list of caught Pokemon.
6. Effective utilization of state management techniques to store and manage the list of caught Pokemon.
7. Successful integration and implementation of the Zustand state management library for managing the state of caught Pokemon.
8. A well-structured and modular codebase that adheres to React's component-based architecture.
9. Separation of concerns and reusability of components throughout the application.
10. Successful integration with a Pokemon API, such as the official PokeAPI, to fetch data about specific Pokemon or lists of Pokemon.
11. Proper handling of asynchronous data fetching and error management.
12. A visually appealing and user-friendly interface that provides a seamless experience for Pokemon enthusiasts.
13. Attention to detail in terms of design, layout, and responsiveness across different devices and screen sizes.
14. Successful deployment of the Pokedex webpage to a hosting platform or service(render/netlify), making it accessible to users.

<u>Submission</u>

Submit the following to the sheet provided here WEB101 - CAP2 SUBMISSION :

1. Your Github Repository containing your working web application including a descriptive README.md. Please note that as the assignment doesn't require a report submission, ensure that the comments successfully suffice in supporting your thought process through this assignment. You are required to define the functionality in your README.md that you application would achieve.
   a. **Github repository name format**: “stdno_WEB101_PA2” (eg: 02190108_WEB101_PA2)
   b. **Submission Portal**: Github
   c. **Deployment to Cloud:** Render- ”xxxxx.onrender.com” OR any other cloud provider.

<u>Date of Submission:</u>

9th June, 2024

Note:

- Plagiarism of work without referencing and understanding the work will not be tolerated.
- Late submission of assignments will cause a 5% deduction in grades every day from the date of submission.
- Any mistake in file format, name of file etc will result in a deduction of 1 mark on every mistaken submission.

Grading Criteria

This continuous assessment will be assessed based on the following criteria:

Effective use of component-based architecture and state management	- 5.0%
Application of Design Patterns	- 5.0%
Code Quality and Efficiency	- 5.0%
User Interface and Experience	- 5.0%
Documentation and Code Organization	- 5.0%

Penalties

Note: You will not be able to score less than zero.

- **[-50% of overall grade of assignment]** - Student is unable to justify and explain code logic of his/her implementation