FULLSTACK ASSESSMENT

REDUX

PART 1.

1.1 The reducer function is defined for managing the state in a Redux application. The function takes the two parameters state and action. It checks the type of the action and performs different operations based on the action type. If the action type is 'increment', it increments the value property of the state by 1 and returns the updated state. If the action type is anything other than 'increment', it returns the current state.

1.2 To decrease the value by 1 a new “else if”condition can be added to check if the action type is 'decrement'. If it is, the reducer function can reduce the value property of the state by 1 and can return the updated state.

1.3 In order to reset the state an additional “else if” condition can be added to check if the action type is 'reset'. If it is, the reducer function sets the value property of the state to 0 and therefore, resetting the state to its initial value.

PART 2.

2.1 On line 34 the useState hook is used to declare the state variable “studentsCount “and the function “setStudentsCount” to update its value. The initial value of studentsCount is set to 0.
On line 39 the button will be rendered on the UI.The addStudent function is assigned as the event handler, therefore if the button is clicked the “onclick” event will be invoked and the addStudent funktion will be executed

2.2

1. Create onClick event “addstudent” that executes the allocated function when clicked on
2. Add logic to function by filtering the array and only get students were present = true
3. Get length of this array to get the total num of present student
4. Update state var “studentCount” accordingly

2.3

To make sure that the func is triggered when clicking on the btn, it is important to add the addStudent function the the event handler:
<button onClick={addStudent} >Add Students/button>

2.4 In order to update the state with the result, it is important to update the setStudentCount var that was passed into the useState hook:
setStudentCount(presentStudents)

PART 3.

3.1 On line 174, the code is updating the state in the countReducer function. The code takes the current value of state.value and adds the value of action.payload to it

3.2. For that, the useDispatch hook must be imported and inside the component, the useDispatch hook is used to get the dispatch function. Additionally, the handleButtonClick function can be used as the click handler for the button.

3.3. Fig. 5 might be better suited to ensure that the “increment” action updates the state with the correct total number of students who are present. The difference is in Figure 5, when the "increment" condition is invoked, the value of the state is replaced directly with the value of action.payload. This means that the state's value will be set to the exact total number of present students.
Whereas in Fig. 4, the state's value is updated by adding the value of action.payload to the current value of the state. This assumes that action.payload is a value that will be added incrementally to the current state.

In summary, the code in Fig. 5 makes sure that the state's value is replaced with the total number of present students, while the code in Fig. 4 updates the state by adding a value to its current value.

Algorithms 1 (Coding)

```python
def palindromeOrNot(string):
    palindrome = string[::-1]

    print(palindrome)
    if palindrome in string:
        return True
    else:
        return False

string = "radar"
is_palindrome = palindromeOrNot(string)
print(is_palindrome)
```

The time complexity of the shown approach is O(n), where n is the length of the string. Reversing the string using slicing takes O(n) time and comparing the two strings also takes O(n) time.

When it comes to the space complexity of my approach it is O(n) as well. The code stores the reversed string in a separate variable, which requires additional memory proportional to the length of the input string. I chose this approach as it is easy and understandable however, this may give false results so an optimisation would be efficiently.

Algorithms 2 (Coding)

1. Initialise a var “n” with the length of the array +1
2. For loop through each num in the array to check whether it is a positive integer and if it is not or a non-numeric value,return error
3. Calculate the expected sum of the num 1 to 10 and find the missing number (?)
4. Return the missing num