**Design & Development Section**

**Prototype 1 - Main Page**

**Design**

**Overview:**Overall, I intend to design the button-based navigation workflow necessary for users to navigate in between the different forms (pages). This section will therefore have a flow of what the user can do in the program I’m creating, the creation of the different form instances, and how transition to the next page will work. Where listing is needed, I will try to ensure proper navigation and data management.

**Decomposition to computable sections:**

|  |  |
| --- | --- |
| **Section** | **Justification (suitable for computation because…)** |
| Mapping Buttons to Open New Forms | This will allow my end user to navigate without errors by utilising different event handlers in the program to create and display new form instances. As a result I hope this will make sure user actions (button clicks) are mapped to their respective pages while also maintaining smooth transitions throughout the program as necessary per my analysis for my stakeholders listed above in requirements. |
| Implementing Navigation for Specific Pages | By assigning buttons to target pages (e.g., "Mock Test" opens InstructionsForm), the program can perform the way I want it to do for ease of user navigation in a predictable manner for testing and development throughout. This will also further help me eliminate errors, making sure the correct form opens in response to clicks on the buttons in the program by the end user. |
| Passing Data to Progress Page | Allows the Progress\_Page to show relevant user data - for example test scores. This will allows me to pass a dictionary from a global state in this program, making the content dynamic and real-time, as needed for my program relating to the analysis carried out which states that the user experience is tied directly to the ability to check their progress out. |
| Finalized Navigation Workflow | Allows for a logical workflow in relation to user interaction, navigation and data accessibility life cycle within my program. This will hopefully allow me to add smooth, button-based transitions while keeping the codebase fully functional and prevent any errors that hinder user experience. |

***Step 1: Mapping Buttons to Open New Forms***

To navigate between different pages, I will create event handlers for every button. When the user clicks a button, the program will create an instance of the corresponding form, show it and optionally hide the current form in view.

**Pseudocode:**

WHEN button is clicked:

CREATE a new instance of the target form (e.g., PracticePage)

OUTPUT the target form

HIDE the current form (optional)

**Reason:**  
Mapping buttons to specific actions will allow my end users to navigate through the program seamlessly which is, as emphasised above and in my overall documentation, key to making this program useful – the ability for users to interact with the program and transition between the different forms with no errors or mishaps. By creating a new form instance on a button click, I can therefore make sure that the program responds immediately to the user’s input, improving interactivity and therefore the user experience. Hiding the current form (optional) will also prevent the clutter on the screen for my goal of user experience optimisation.

**Approach:**

|  |
| --- |
| * **For all navigation buttons, I will define click event handlers.** |
| * **Event handlers will be created for each form to instantiate and display it.** |
| * **This overall should guarantee that the actions taken by users are mapped to the respective targets they were intended for within the program.** |

***Step 2: Implementing Navigation for Specific Pages***

In this next step, I need to add functionality to each button to allow users to open the appropriate page. These will consist of buttons such as "Mock Test," "Practice Page," and "Progress Page."

**Pseudocode for Mock Test Page:**

FUNCTION Mock\_test\_Click

CREATE new instance of InstructionsForm called nextForm

OUTPUT nextForm (Show it on the screen)

HIDE the current form

END FUNCTION

**Reason:**  
By clearly linking each button to its associated page (for example, the Mock Test button leads to the InstructionsForm), I can make sure that the program operates as the user expects it to. This is important for my intended user experience since the appropriate pages will load when users click on these buttons in the UI.

**Approach:**

|  |
| --- |
| * **When Mock Test Button is pressed, I will try to make sure InstructionsForm page shows instructions before the user takes the test.** |
| * **Completely check code for button references and their respective matches to prevent any bugs.** |

***Step 3: Passing Data to Progress Page***

To create the Progress\_Page, I will need to provide a dictionary that monitors the user’s test scores. This dictionary will then keep track of the tests that have been taken along with their respective scores for example Matthew scored 50% on Test 1. When I create the Progress\_Page, I will then be able to obtain the test scores from either a global state data class.  
  
**Pseudocode:**

FUNCTION Progress\_Click

RETURN testScores dictionary from GlobalData

CREATE new instance of Progress\_Page called nextForm, passing testScores as an argument

OUTPUT nextForm

CLOSE the current form

END FUNCTION

**Reason:**  
The *Progress\_Page* needs access to user progress data to display attempted tests and scores. By retrieving the testScores dictionary from a global state and passing it to the *Progress\_Page*, I ensure that the page reflects up-to-date and relevant information. This design also promotes separation of concerns, as data management is handled separately from UI navigation.  
  
The Progress\_Page should have access to the data of user progress such as attempted tests and scores. I call the global state for testScores dictionary and pass it to the Progress\_Page, therefore the page returns useful and current info. It also encourages separation of concerns, with data management being independent from UI navigation. I will set up a logic to get test scores from a global data store.

**Approach:**

|  |
| --- |
| * **I will add logic which will aim to retrieve the test scores from a global data store.** |
| * **When I navigate to that form, I will pass this dictionary to Progress\_Page constructor.** |
| * **This should allow *Progress\_Page* to display scores dynamically and provide real-time feedback to the end user improving the user experience.** |

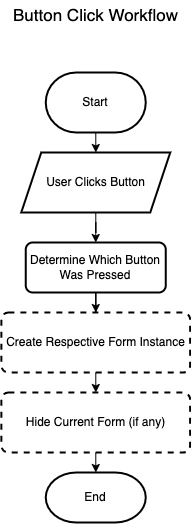
**Final Workflow Summary**

The finalized button navigation workflow will be implemented in three clear steps:

1. Since the state changes as described above, the program listens for individual button clicks.
2. Form Navigation Based on the button click, I will instantiate the relevant form and then show it
3. Data Management: For specific pages (e.g., *Progress\_Page*), I will pass necessary data to support dynamic content.

**Flowchart Representation**

The following flowchart summarizes the Button Click Workflow:

****

By carefully planning these steps, I can ensure smooth navigation between pages and efficient handling of user actions. Each form will serve its intended purpose, and any necessary data will be passed seamlessly. This design should ensures that the workflow is clean, functional, and easy to maintain.

**Development**