# Declaration of Academic Integrity

Academic Integrity is a central tenet of Singapore Polytechnic. The polytechnic rules state that "Cheating in examinations and other assessed work is a very serious offence. This includes copying and using plagiarised material. Any student who cheats, attempts to cheat or breaches any rules for examinations and tests will face disciplinary action. The student is liable to be expelled."

Check **only one** of the two options below:

I affirm that the work I submit is my own, produced without help from any AI tool(s) and/or other source(s).

I affirm that the work I submit has been produced with the use of AI tool(s) and/or other source(s) which I have acknowledged fully in the [following section](#_Acknowledgement_of_use).

By signing this form, I declare that the above affirmation made is true, and that I have read and understood the rules stated in Students Handbook on “[Plagiarism](https://www.sp.edu.sg/sp/student-services/osc-overview/student-handbook/intellectual-property-copyright-and-plagiarism)” and “[Breach of Examination/Assessment Rules](https://www.sp.edu.sg/sp/student-services/osc-overview/student-handbook/conduct-in-examinations-breach-of-exam-rules)”.

|  |  |  |
| --- | --- | --- |
| Name:  Su Hnin Khine | Student ID:  2340841 | Class:  DCITP/FT/1A/02 |
| Signature: | Date:  5/8/2023 | |

## Acknowledgement of use of AI tool(s) and/or other source(s) (where applicable)

### AI tool(s)

|  |  |
| --- | --- |
| **Name of AI tool** | ChatGPT |
| **Input prompt** | after submission of a form, how to make it so that it will get linked to another html form |
| **Date generated** | 1/08/2023 |
| **Output generated** | To link one HTML form to another after submission, you can use a combination of HTML and JavaScript. Here's a step-by-step guide on how to achieve this:  1. Create the first HTML form (source form):  In this form, you will capture user input and submit it to the server. The form's `action` attribute should point to a server-side script that will process the data.  ```html  <!-- source\_form.html -->  <form id="sourceForm" action="submit\_data.php" method="post">  <!-- Your form fields -->  <input type="text" name="name" placeholder="Name">  <input type="email" name="email" placeholder="Email">  <!-- Add any other form fields as needed -->    <input type="submit" value="Submit">  </form>  ```  2. Create the second HTML form (target form):  In this form, you can display the submitted data or any other information relevant to the user. This form should be placed in a separate HTML file, and we will link to it using JavaScript.  ```html  <!-- target\_form.html -->  <h2>Submitted Data:</h2>  <div id="targetForm"></div>  ```  3. Add JavaScript to handle form submission and linking:  We'll use JavaScript to handle the form submission and dynamically link to the target form. We'll prevent the default form submission behavior and fetch the data from the server asynchronously. Once we have the data, we'll display it in the target form.  ```html  <!-- script.js -->  <script>  document.getElementById("sourceForm").addEventListener("submit", function (event) {  event.preventDefault();  const formData = new FormData(event.target);  const data = {};  formData.forEach((value, key) => {  data[key] = value;  });  // Send the form data to the server using fetch or AJAX  fetch("submit\_data.php", {  method: "POST",  body: JSON.stringify(data),  headers: {  "Content-Type": "application/json",  },  })  .then((response) => response.json())  .then((responseData) => {  // After successful submission, load the target form and display the data  window.location.href = "target\_form.html";  })  .catch((error) => {  console.error("Error submitting form:", error);  });  });  </script>  ```  4. Set up the server-side script:  Create a server-side script (e.g., "submit\_data.php") that will receive the form data and process it as needed. This could involve storing it in a database or performing other operations.  Note: This example uses JavaScript's `fetch` API to send data to the server. If you prefer, you can use AJAX or any other server communication method that suits your application.  Remember to adapt the server-side script to handle the form data appropriately based on your specific requirements.  With this setup, when the user submits the source form, the form data will be sent to the server for processing. Once the submission is successful, the user will be redirected to the target form page, where the submitted data will be displayed or used as needed. |
| **Impact on submission** | I used it in form validation |

|  |  |
| --- | --- |
| **Name of AI tool** | ChatGPT |
| **Input prompt** | what do you mean by responsive image |
| **Date generated** | 28/07/2023 |
| **Output generated** | A responsive image is an image on a website that automatically adjusts its size and dimensions based on the screen or viewport size of the device it is being viewed on. The goal of responsive images is to ensure that the image looks good and fits well within the layout of the webpage, regardless of whether it is viewed on a large desktop monitor, a tablet, or a mobile phone.  The concept of responsive images is closely related to the broader concept of responsive web design. Responsive web design aims to create websites that adapt to different screen sizes and devices, providing an optimal user experience. Images are an essential part of web content, and making them responsive ensures that they don't disrupt the overall design and usability of the website on different devices.  There are several techniques for implementing responsive images. Some common methods include:  1. Using CSS Media Queries: CSS Media Queries allow you to define different styles based on the screen size or device capabilities. By setting the `max-width` property of the image to 100% within a media query, the image will scale down proportionally to fit its container on smaller screens.  ```css  /\* CSS for responsive image \*/  img {  max-width: 100%;  height: auto;  }  ```  2. Using the `srcset` attribute: The `srcset` attribute allows you to provide multiple image sources (different resolutions or sizes) and let the browser choose the most appropriate one based on the device's screen resolution.  ```html  <!-- Responsive image using srcset -->  <img src="image-small.jpg"  srcset="image-small.jpg 480w,  image-medium.jpg 768w,  image-large.jpg 1200w"  alt="Responsive Image">  ```  3. Using the `picture` element: The `picture` element allows you to define multiple sources and specify media conditions for each of them. The browser will then choose the appropriate source based on the device's characteristics.  ```html  <!-- Responsive image using the picture element -->  <picture>  <source media="(max-width: 768px)" srcset="image-small.jpg">  <source media="(min-width: 769px)" srcset="image-large.jpg">  <img src="image-medium.jpg" alt="Responsive Image">  </picture>  ```  By implementing responsive images, you can enhance the user experience and ensure that your website looks visually appealing and functions well across various devices and screen sizes. |
| **Impact on submission** | I used it in creating responsive images |

### Other source(s)

|  |  |
| --- | --- |
| **Source** | **Impact on submission** |
| < *Link to online source* > | W3schools,getbootstrap.com,mdnwebdocs.com  I used these sources for form validation and studying about javascripts.I also used them to learn about bootstrap components.  Images credit to pixabay.com,splash.com,flaticon.com,sp.edu.sg  fonts credit to googlefonts.com |
| < *Author, Book / Article / Journal, Year, Pages* > | < *Briefly explain which part of your submitted work the Book / Article / Journal was applied* > |
| < *Student / Person name* > | < *Briefly explain which part of your submitted work the help of another student / person was applied* > |
| < *Others* > | < *Briefly explain which part of your submitted work the source was applied* > |