Web Systems and Technologies (EIE4432)
Lab 2: Use of Bootstrap, SVG & JavaScript

Expected Outcomes

- Recreate the UI of the Drink Ordering System using Bootstrap, ensuring responsiveness and an appealing design
- Implement JavaScript functionality to capture and validate user inputs, facilitating smooth submission of drink orders
- Develop a Seat Booking System using SVG, providing an interactive interface with clear seat availability indicators
- Enable seamless seat selection and booking, preventing conflicts and ensuring a user-friendly experience
- Deploy the website to the cloud using Vercel, ensuring proper configuration and optimal functionality

Submissions

- Please refer to the teaching schedule for the deadline
- This is an individual work
- Students should host their work on a designated hosting service and provide a public domain for viewing the website live
- Students need to submit a report with screen captures showing they have achieved the outcomes stated above
- Every screen capture should be full screen and show the **Student ID, System Date & Time**, otherwise **1 grade would be deducted** from the laboratory
- Late submission would deduct 20-50% of the mark of this laboratory
- Please declare the use of GenAI tools and how they have been used in the appendix page similar to the following:

    - I declare that Generative AI tools have been used to prepare the submitted work. The Generative AI tools used and the manner in which they were used are as follows: ____.

    - You must reference them in according with the accepted academic conventions (e.g. IEEE or APA styles).

Assessment Criteria

The report will be assessed according to

- The clarity of presentation,
- Technical correctness, and
- Whether the screen captures can demonstrate that the outcomes have been achieved.

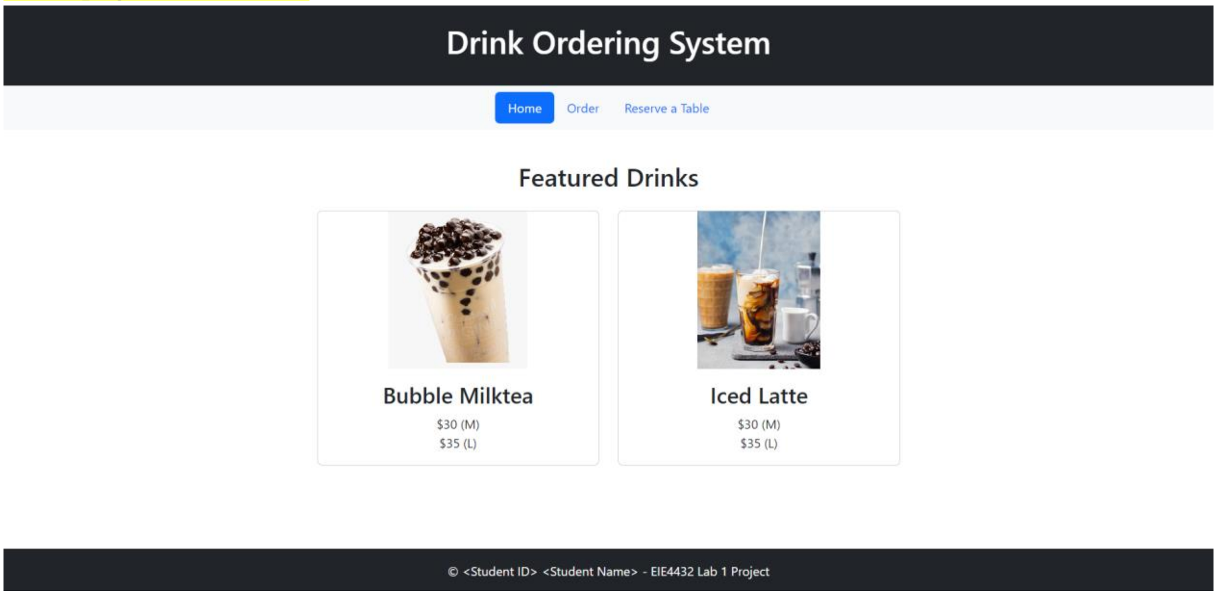
Lab Activities

- Task 1: Recreate UI of Drink Ordering System using Bootstrap (60 mins)
- Task 2: Submit Drink Orders with JavaScript (60 mins)
- Task 3: Table Booking System (45 mins)
- Task 4: Publish your Website (15 mins)

Task 1: Recreate UI of Drink Ordering System using Bootstrap (60 mins)

Expected Outcome

Homepage (index.html)



Drink Ordering System

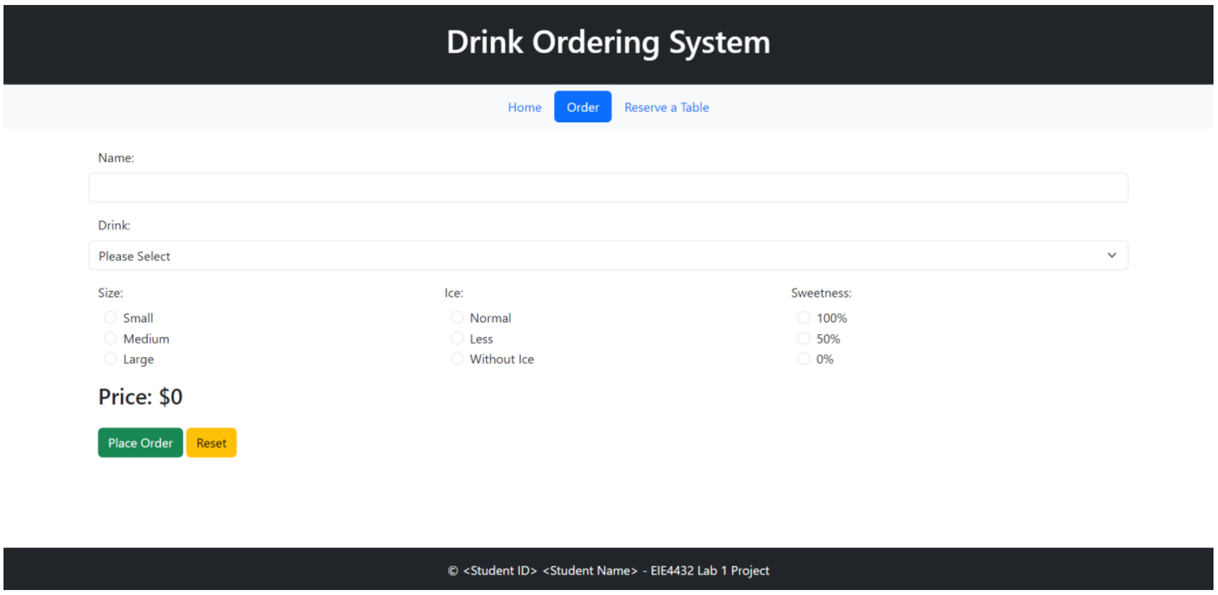
Featured Drinks

Bubble Milktea
$30 (M)
$35 (L)

Iced Latte
$30 (M)
$35 (L)

© <Student ID> <Student Name> - EIE4432 Lab 1 Project

Drink Ordering Form (order.html)



Drink Ordering System

Name:

Drink:
Please Select

Size:	Ice:	Sweetness:
Small	Normal	100%
Medium	Less	50%
Large	Without Ice	0%

Price: $0

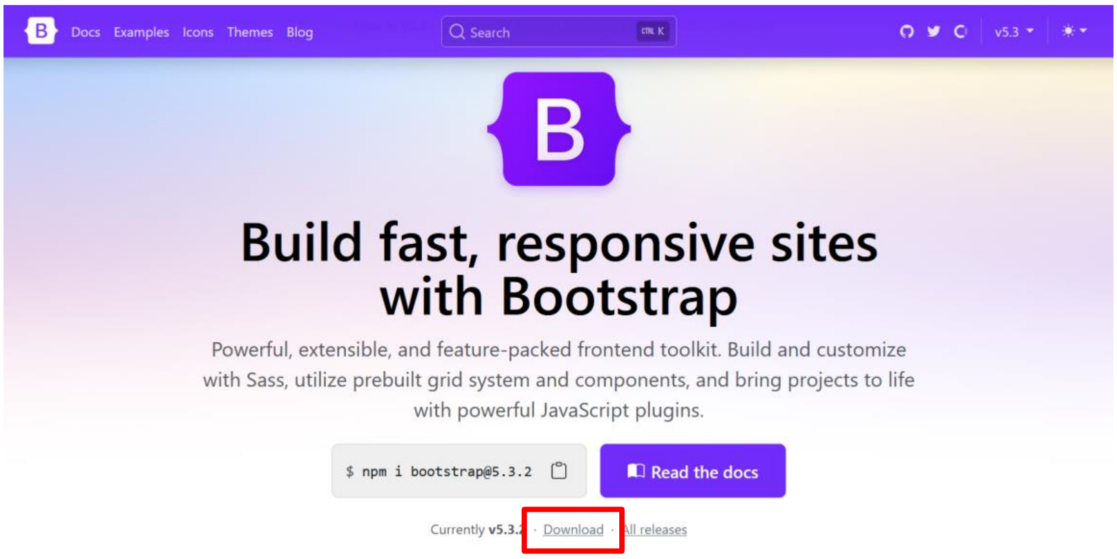
© <Student ID> <Student Name> - EIE4432 Lab 1 Project

Suggested File Structure

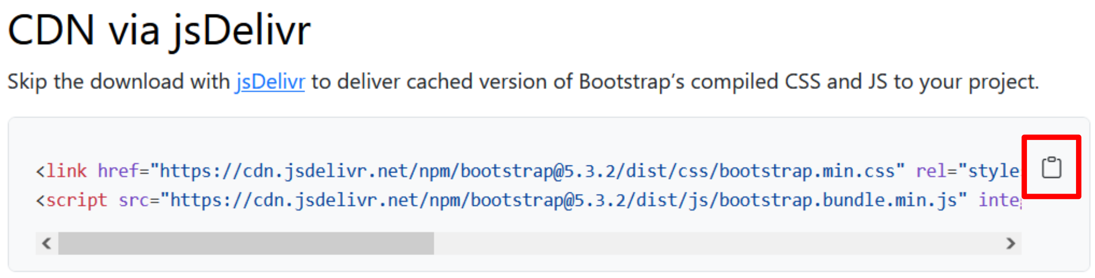
```
└── <Student ID>_EIE4432_LAB2 (Project Root Folder)
    ├── assets
    │   ├── img1.jpg
    │   ├── img2.jpg
    │   └── etc...
    ├── index.html
    ├── order.html
    └── reserve.html
```

- Compare to Lab 1, **styles folder is removed**, as we are integrating Bootstrap into the system UI design

1. Create a folder on designated location, such as desktop, name it as "<Student ID>_EIE4432_LAB2"
2. Rename file as "index.html", save it in the folder created
3. **You are strongly recommended to create a new private repository on GitHub, and keep the project in GitHub, the deployment to cloud service would need to connect with a GitHub repository at the end of the lab exercise**
4. Create the basic HTML document skeleton
5. Go to official website of Bootstrap: https://getbootstrap.com/, click "Download" on home page


Docs
Examples
Icons
Themes
Blog
Search
v5.3
Build fast, responsive sites with Bootstrap
Powerful, extensible, and feature-packed frontend toolkit. Build and customize with Sass, utilize prebuilt grid system and components, and bring projects to life with powerful JavaScript plugins.
$ npm i bootstrap@5.3.2
Read the docs
Currently v5.3.2 · Download · All releases

6. Find the "CDN via jsDelivr" section, click the "board" icon to copy 1st code snippet of <link> & <script>
   - bootstrap.min.css: Managed all CSS styling classes from Bootstrap
   - bootstrap.bundle.min.js: Managed all JS related function from Bootstrap



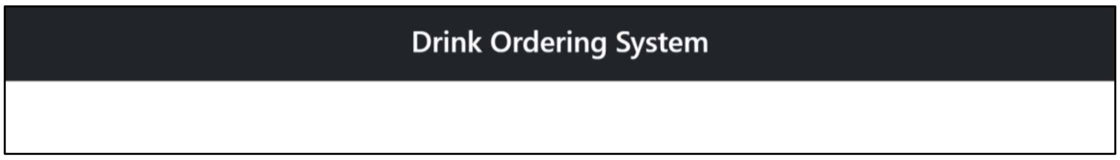
CDN via jsDelivr

Skip the download with jsDelivr to deliver cached version of Bootstrap's compiled CSS and JS to your project.

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="style
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" inte
```

7. Paste the copied code to <head> section of your "index.html"
8. This time, it is not necessary to create any external CSS file. You are required to use plain Bootstrap
9. Create a title for the website: <Student ID>_Drink Ordering System
10. The UI is basically the same structure as Lab 1, you are going to recreate it using Bootstrap for a decent UI and reduce the pain of CSS styling

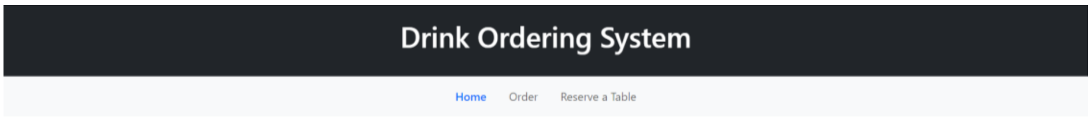
11. Create a <header> element with text "Drink Ordering System", to apply Bootstrap styling, add specific class names to "class" attribute of the element



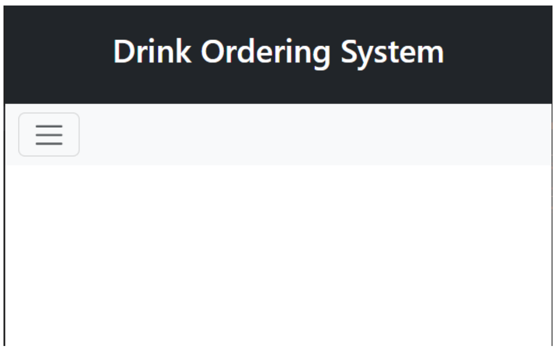
- Select a background color: https://getbootstrap.com/docs/5.3/utilities/background/#background-color
  - e.g. `<header class="bg-primary">Some Text Here</header>`
- Select a text color (default: black text color): https://getbootstrap.com/docs/5.3/utilities/colors/#colors
- Or use both text & background color combination from Bootstrap: https://getbootstrap.com/docs/5.3/helpers/color-background/
- Center text in the middle: https://getbootstrap.com/docs/5.3/utilities/text/#text-alignment
- Create padding vertically for spacious feel: https://getbootstrap.com/docs/5.3/utilities/spacing/#margin-and-padding

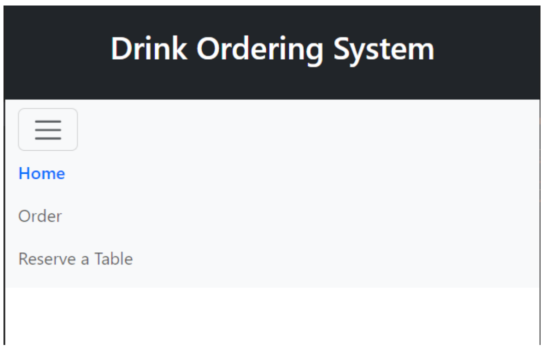
12. Save changes and test with Live Server Extension from VS Code or any other methods
13. Create a navigation bar beneath header, it should include the following 3 links:
    - Home (index.html)
    - Order (order.html)
    - Reserve a Table (reserve.html)
14. Navigation bar should be responsive across different devices
    - When user visits with small screen device, links should be collapsed
    - Otherwise, links should be expanded as usual
15. Choose an appropriate navbar template from Bootstrap, modification from original template code snippet is necessary to achieve desire outcome (This is a skill of using ready-to-use material to create UI)

(Large Screen Device)



(Small Screen Device)





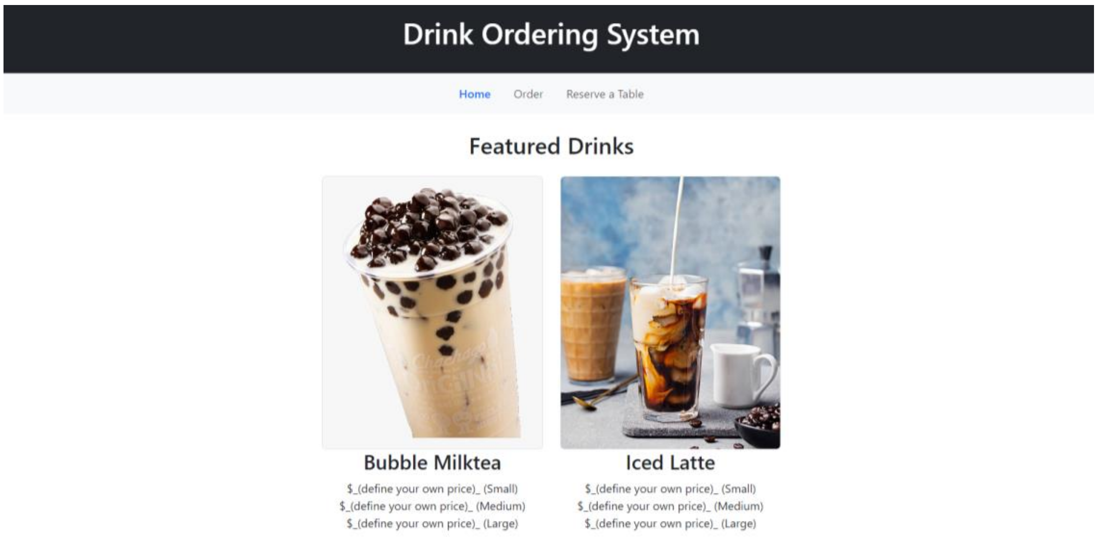
- Navbar Template: https://getbootstrap.com/docs/5.3/components/navbar/

16. For the current page that user is visiting, show a different color & bold the text to indicate it
    - To stylize link color: https://getbootstrap.com/docs/5.3/helpers/colored-links/
    - To bold text: https://getbootstrap.com/docs/5.3/utilities/text/#font-weight-and-italics
17. To test for small screen devices, open context menu with right click / press F 12 to open inspector, then click 🗔 icon to toggle device toolbar
    - Select a dimension, e.g. iPhone SE, to view the UI display on small devices
18. Bonus: Create spacing between each nav links, you can use either margin or gap utilities to achieve so, and the utilities should be activated **under large screen display only**
    - https://getbootstrap.com/docs/5.3/utilities/spacing/

Lab #1.1
Provide the following screenshots with your **Student ID and System Date & Time.** Your code should be indented correctly (each nested tag should be indented exactly once inside of its parent tag)

- Source code of index.html
- Show the header & navigation bar element in **large screen** display
- Show the header & navigation bar element in **small screen** display

19. Create a basic container beneath the navigation bar as a <div> element
    - https://getbootstrap.com/docs/5.3/layout/containers/
20. Here is the expected content & layout in container


Drink Ordering System
Home
Order
Reserve a Table
Featured Drinks
Bubble Milktea
$_(define your own price)_ (Small)
$_(define your own price)_ (Medium)
$_(define your own price)_ (Large)
Iced Latte
$_(define your own price)_ (Small)
$_(define your own price)_ (Medium)
$_(define your own price)_ (Large)

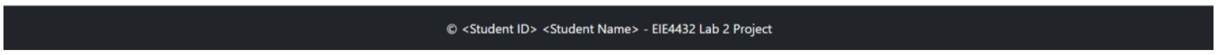
21. In the container, it should include:
    - Heading with text: Featured Drinks
        - Text should be center aligned
        - Add some margin vertically to create a specious look
    - 2 featured drinks information with images & text, define your own price this time, it would be used later in the order form
        - Bubble Milktea $__ (Small) $__ (Medium) $__ (Large)
        - Iced Latte $__ (Small) $__ (Medium) $__ (Large)

22. To create the heading text, add a heading tag such as <h2>
23. Add class to heading tag to make it center aligned
    - https://getbootstrap.com/docs/5.3/utilities/text/#text-alignment
24. Add some margin vertically to create a spacious look on heading text
    - https://getbootstrap.com/docs/5.3/utilities/spacing/#margin-and-padding
25. Create a row to wrap 2 featured drinks information, a row should be a <div> element. Ensure everything in the row are horizontally centered, find appropriate classes from the following links
    - https://getbootstrap.com/docs/5.3/layout/grid/#auto-layout-columns
    - https://getbootstrap.com/docs/5.3/utilities/flex/#justify-content
26. Create 2 columns, each column for 1 drink information
    - For responsive purpose, columns shall be full width on small screens, 1/3 on medium, and 1/4 on large screens
    - Add class “col-12 col-md-4 col-lg-3” to each column <div> element
27. Add image of the drink (download from the assests.zip) to each column, make sure the value of image `src` attribute is **relative link**, but not absolute link that only link to your LOCAL path, it would affect the deployment stage later
    - Relative Link: assets/bubble-milktea.png
    - Absolute Link: C:\Users\User\Desktop\<Student ID>_EIE4432_LAB2\assets\bubble-milktea.png
28. Adjust the size of image using width & height utilities, add class “w-100 h-75”
    - The number (100/75) is in unit of percentage (%), only 25/50/75/100 are allowed
    - https://getbootstrap.com/docs/5.3/utilities/sizing/
29. Ensure the images keep its aspect ratio and fills the given dimension using the “object-fit” utility
    - https://getbootstrap.com/docs/5.3/utilities/object-fit/
    - Add “object-fit-cover” to class, so the aspect ratio of images can be retained
    - Add “border rounded” to class, so there would be a thin border & rounded corner around the image to make it look more decent
    - `alt` attribute is required, so that image description could be available when the image is unable to be displayed due to any reason
30. Add sub-heading to each column for the name of drink
    - 1st drink: Bubble Milktea
    - 2nd drink: Iced Latte
    - Text should be center aligned using Bootstrap class name
    - Give some space between the drink name & image using margin utility

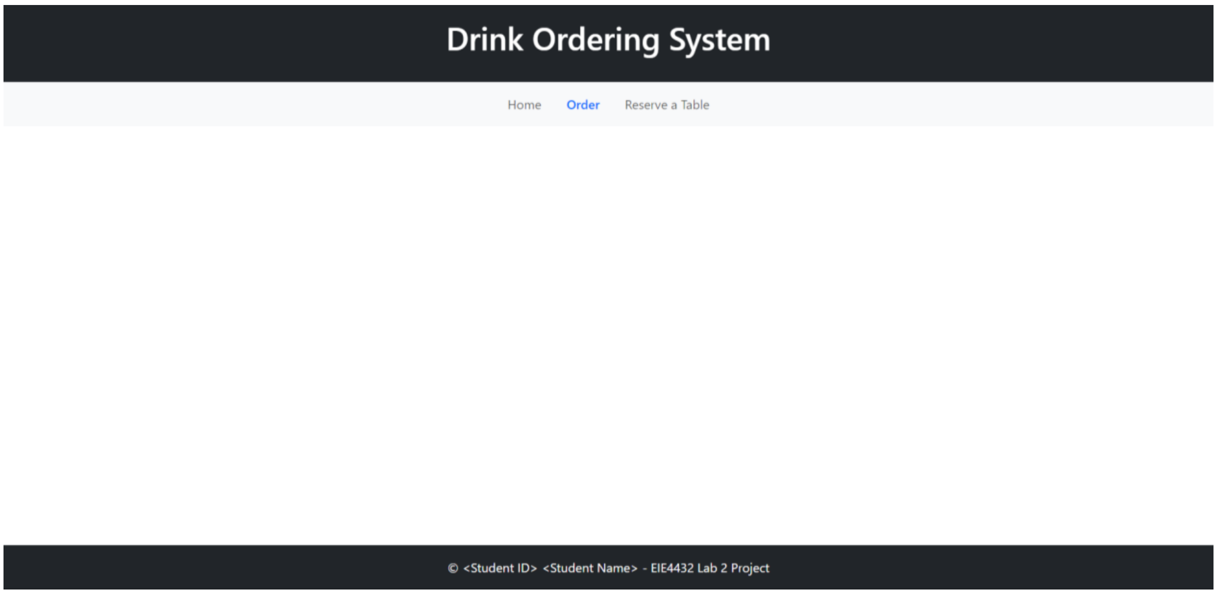
31. Add 3 <div> elements for pricing information of drink. This time, the price of drink is defined by yourself, and it would be used in task 2 for drink order price calculation
    - 3 prices are needed: small, medium & large size
    - The information must be center aligned using Bootstrap class name

> **Lab #1.2**
> Provide the following screenshots with your **Student ID and System Date & Time.** Your code should be indented correctly (each nested tag should be indented exactly once inside of its parent tag)
>
> - Source code of index.html (container part)
> - Show the homepage in **large screen** display
> - Show the homepage in **small screen** display

32. Add <footer> element after the container <div> element

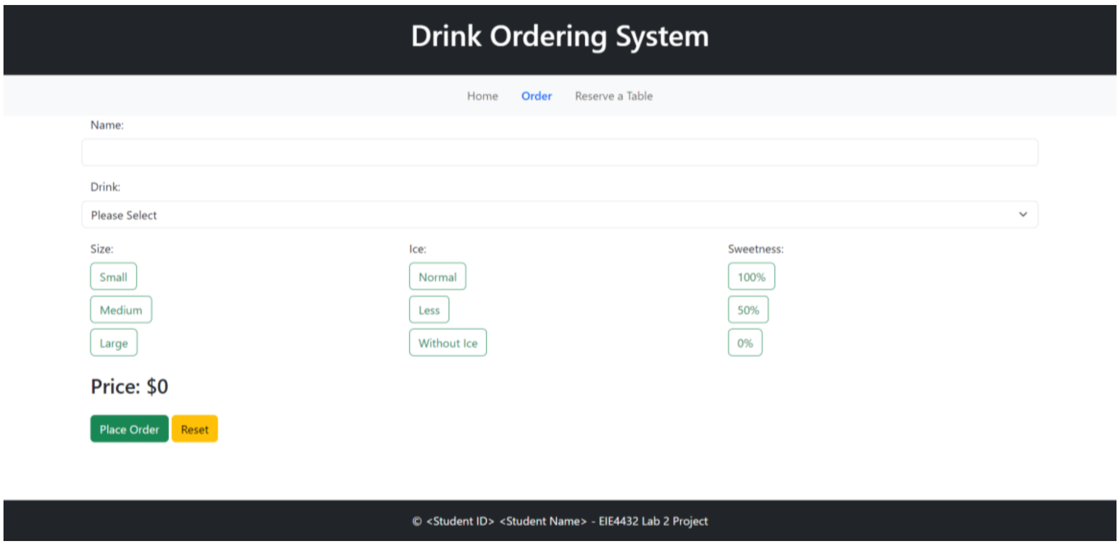


    - Text: © <Student ID> <Student Name> - EIE4432 Lab 2 Project
    - Background color must be the same as <header>
    - Give vertical spacing using the padding utility
    - Text should be center aligned
33. Create a new HTML file and rename as "order.html"
34. Copy all code from "index.html" to "order.html"



    - Clear everything inside container **(Remember to KEEP container)**
    - Add some vertical spacing to container using padding utility, e.g. py-4
    - Update the navigation active styling from "Home" to "Order"
    - Footer should float in the middle of the screen now, make it stick to bottom edge by adding "fixed-bottom" class
    - https://getbootstrap.com/docs/5.3/helpers/position/#fixed-bottom
35. Add a form element in the container
    - Set the value of `action` attribute to "`#`" placeholder
    - Set the value of `method` attribute to "`POST`"

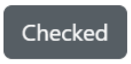
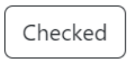
36. Consider the values you need to get from the user, create 4 rows for the form layout



- 1st row: Name (Text Input)
- 2nd row: Drink Selection (Dropdown Menu)
- 3rd row: Size, Ice & Sweetness Selection (Radio Buttons with Bootstrap button style)
- 4th row: Price (Simply <div> element)
- For each row, add some margin to its bottom side for spacious feel, e.g. mb-3
- Beneath the 4th row, add 2 buttons
  - "Place Order" button [type of button: `submit`]
  - "Reset" button [type of button: `reset`]

37. For 1st row, create text input field for customer name, refer to the "Form control" section of Bootstrap
    - https://getbootstrap.com/docs/5.3/forms/form-control/#example
    - Get template code snippet, and change the value of `type` attribute of <input> accordingly
    - It is very important to set `name` attribute properly, for further handling using JavaScript
38. For 2nd row, create the dropdown menu with <select> element, remember to set `name` attribute properly
    - Provide at least 3 options for users
    - The 1st option **MUST** be a default option with text: **Please Select**
    - It is very important to set `value` attribute for each option properly
39. For 3rd row, create 3 columns for: Size, Ice & Sweetness selections
    - Each column should be evenly distributed, simply use "col" class is enough

40. In each column, add a label and 3 <div> elements. Each <div> element consists of a radio button using Bootstrap button style instead of traditional radio button.
    It can enhance the usability for mobile users, so they can select an option more easily.
    - https://getbootstrap.com/docs/5.3/forms/checks-radios/#radio-toggle-buttons
    - Avoid using "secondary" color class, as it would lead user that buttons are disabled
    - Use outline style button instead of solid style, button state would be more clear

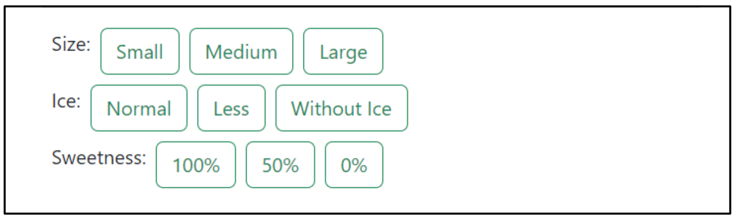
Solid style button	**Outline style button**
 Checked	 Checked

    - It is very important to set `name` & `value` attribute properly for each option
    - To control the style of radio button, modify `class` attribute of <label> element
41. Make 3 options for each selection, suggested options are as follows:
    - Size: Small, Medium, Large (name: size)
    - Ice: Normal, Less, Without Ice (name: ice)
    - Sweetness: 100%, 50%, 0% (name: sweetness)
42. For the 4th row, create a <div> element with text: Price $0
    - Text should be bigger and bolder than default
    - https://getbootstrap.com/docs/5.3/utilities/text/#font-size
43. As the price would update later using JavaScript, use a <span> element to wrap "0" with a unique id
    - Price: $`<span id="price">0</span>`
44. Add 2 <button> elements beneath 4th row: Place Order & Reset
    - https://getbootstrap.com/docs/5.3/components/buttons/
    - Place Order: Set the value of `type` attribute to `submit`
    - Reset: Set the value of `type` attribute to `reset`

Lab #1.3
Provide the following screenshots with your **Student ID and System Date & Time.** Your code should be indented correctly (each nested tag should be indented exactly once inside of its parent tag)

- Source code of order.html (container part)
- Show the order page in **large screen** display
- Show the order page in **small screen** display

45. The selection buttons are not mobile friendly, as the area for users to tap on the buttons is very compact. To make it more user-friendly, transform the buttons in row layout on small device display, and keep the current column layout for large screen display.
    <u>Expected display on small screen device</u>



   - There are plenty ways to achieve the expectation, you can use your own way to complete it or follow the below steps
46. Add “d-inline-block d-lg-flex” to `class` attribute to 3rd row
    - d-inline-block: Makes the element display as an inline-block on all screen sizes
    - d-lg-flex: Changes the display to flex on large screens and above
47. Add “d-flex d-lg-block” to `class` attribute to every column in 3rd row (i.e. every selection section)
    - d-flex: Makes the element display as a flex container
    - d-lg-block: Changes the display to block on large screens and above
48. Add “mb-2 ms-2 ms-lg-0” to `class` attribute to every <div> element of radio buttons
    - Adds a margin-bottom of 2 units
    - Adds a margin-start (left) of 2 units
    - Removes the margin-start (left) on large screens and above
    - Without the margin utility settings, buttons are very close to each other in small screen device display

Lab #1.4
Provide the following screenshots with your **Student ID and System Date & Time.** Your code should be indented correctly (each nested tag should be indented exactly once inside of its parent tag)

- Source code of order.html (container part)
- Show the order page in **small screen** display

Task 2: Submit Drink Orders with JavaScript (60 mins)

Expected Outcome

- Calculate the price based on the selected options (size, ice, sweetness) and updates the price display
- Validate the form to ensure all required fields are filled
- Handles the submission of the order form, performs necessary actions (e.g. remembering data in web storage)
- Resets the form to its initial state, clearing all selected options and input fields
- Updates the price dynamically when any option (size, ice, sweetness) is changed
- Displays a confirmation message to the user after successfully placing the order
- Handles the form submission event and calls appropriate functions for validation and order placement

Instructions

1. Create a new file and rename as "order.**js**"
2. Add a <script> tag in the <head> section of "order.html", the value of src attribute should be "order.js", it ensures the order page can trigger functions in the script
3. Add the `id` attribute to the <form> element in order to attach the event listener
4. Remove the `action="#"` attribute from the <form> element since the form submission is handled by JavaScript
5. Replace the `type="submit"` attribute of the "Place Order" button with `type="button"` to prevent the default form submission behavior
6. In "order.js", create a new function `calculatePrice()`
   - Calculate the price based on the selected size options in the form, update the price dynamically when options change
7. Add `onchange` event handler to each size option in "order.html" and trigger this function
8. Get the value of the selected size from the radio buttons
9. Get the value of the selected drink from the <select> element
10. Initialize the `price` variable to 0
11. If a drink is selected:
    - Check the selected drink using conditional statements
    - Assign the corresponding price to the price variable
12. Adjust the price based on the selected size:
    - Check the selected size using conditional statements
    - Add the appropriate amount to the price variable
    - It **MUST** be matched with the price information on homepage
13. Display the calculated price:
    - Get the element with the id "price"
    - Set its text content to the value of the price variable
14. If no drink is selected:
    - Show an alert message asking the user to select a drink

  - Clear the selected size by unchecking all the radio buttons

15. Create a new function `validateForm()`
    - This function will validate the form inputs before submission
16. Obtain all form values (name, drink, size, ice & sweetness) using appropriate JavaScript methods
17. Check if the name field is empty and show an alert if it is
    - Trim the name value using trim() to remove any leading or trailing whitespace
    - It checks if the trimmed name value is an empty string
    - If the name field is empty, it displays an alert message asking the user to enter their name (e.g. Please enter your name.)
    - The function returns false to prevent form submission
18. Check if a drink is selected:
    - Check if the drink value is "default" or not
    - If no drink is selected, it displays an alert message asking the user to select a drink (e.g. Please select a drink first.)
    - The function returns false to prevent form submission
19. Check if a size is selected:
    - The function checks if the size value is falsy (null, undefined, or false)
    - If no size is selected, it displays an alert message asking the user to select a size (e.g. Please select a size.)
    - The function returns false to prevent form submission
20. Check if an ice level is selected:
    - The function checks if the ice value is falsy (null, undefined, or false)
    - If no ice level is selected, it displays an alert message asking the user to select an ice preference (e.g. Please select an ice preference.)
    - The function returns false to prevent form submission
21. Check if a sweetness level is selected:
    - The function checks if the sweetness value is falsy (null, undefined, or false)
    - If no sweetness level is selected, it displays an alert message asking the user to select a sweetness level (e.g. Please select a sweetness level.)
    - The function returns false to prevent form submission
22. Return true if the form is completely filled:
    - If all the form fields have valid values, the function returns true
    - This indicates that the form is completely filled and ready for submission
23. Create a new function `placeOrder(event)`
    - This function will handle the order placement process by saving the order data to the localStorage
24. Add the `onclick` attribute to the "Place Order" button to trigger `placeOrder(event)`
25. Prevent the default form submission behavior:
    - Receive an event parameter representing the form submission event
    - Use `event.preventDefault()` to prevent the default behavior of form submission
26. Check if the form is validated:

- The function calls the `validateForm()` function to check if the form has been validated
- If the form is validated (all required fields have valid values), the following actions are executed. Otherwise, no further actions are taken

27. If the form is validated, obtain all form values (name, drink, size, ice & sweetness) using appropriate JavaScript methods
28. Group all data into an array
    - Create an array called `orderData` and assign the form values in the order of name, drink, size, ice, and sweetness
29. Save the order data to localStorage:
    - Use `localStorage.setItem("key", value)` to store the orderData array in the browser's localStorage
    - The key used to store the data is "orders"
30. Display a confirmation message
    - Display an alert message to inform the user that the order has been placed successfully (e.g. Order placed successfully! Thank you for your order.)
31. Reset the form to its initial state
    - Use `document.getElementById("order-form").reset()` to reset the form fields to their initial values
    - This clears the entered or selected values in the form
32. Update the price display
    - Set the text content of the element with the id "price" to 0
    - This updates the price display to indicate that the form has been reset

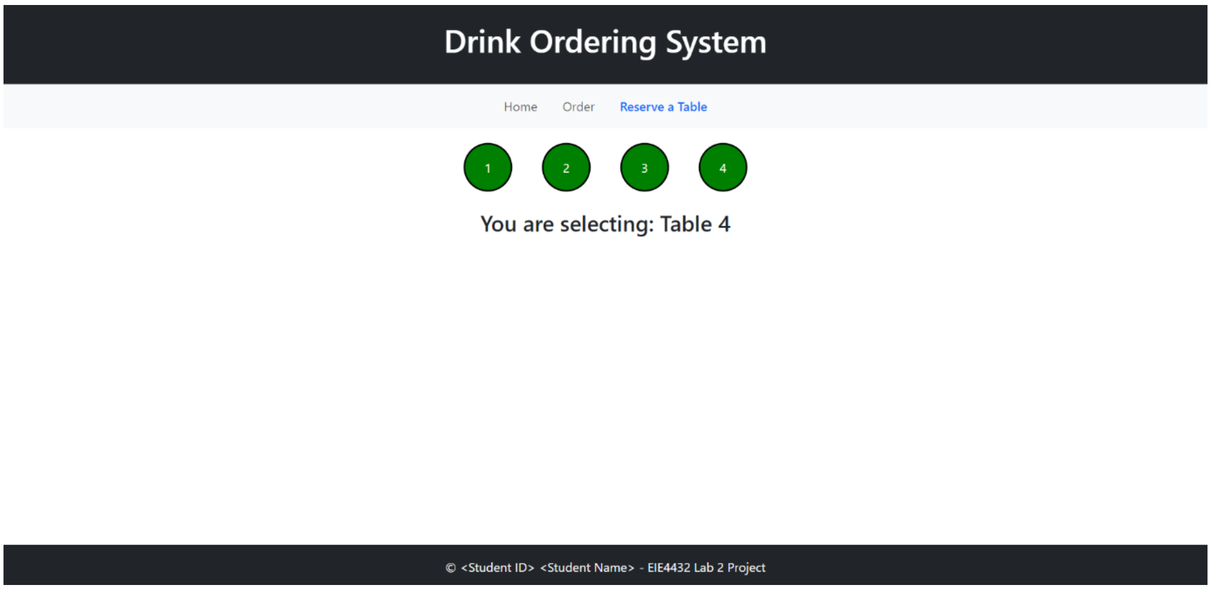
Lab #2.1

Provide the following screenshots with your **Student ID and System Date & Time.** Your code should be indented correctly (each nested tag should be indented exactly once inside of its parent tag)

- Show the screenshot when name is not entered
- Show the screenshot when drink is not selected
- Show the screenshot when size is not selected
- Show the screenshot when ice level is not selected
- Show the screenshot when sweetness is not selected
- Show the screenshot when **medium size of Bubble Milktea** is selected
- Show the screenshot when **large size of Iced Latte** is selected
- Show the screenshot when **small size of the third drink** is selected
- Show the screenshot of placing order successfully
- Show the screenshot of localStorage value from “Inspector”

Task 3: Table Booking System (45 mins)

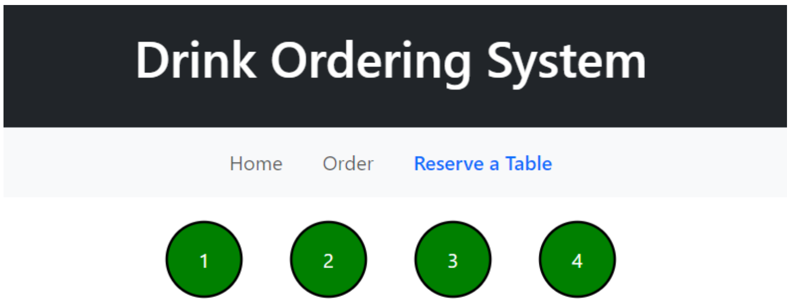
Expected Outcome


Drink Ordering System
Home
Order
Reserve a Table
1
2
3
4
You are selecting: Table 4
© <Student ID> <Student Name> - EIE4432 Lab 2 Project

1. Display a seat map with four tables arranged in a row
2. Each table is represented by a circle with a seat number
3. Seats are initially colored green
4. The selected table number is displayed below the seat map

Instructions

1. Create a new HTML file and rename as "reserve.html"
2. Copy all code from "index.html" to "reserve.html"
   - Clear everything inside container **(Remember to KEEP container)**
   - Add some vertical spacing to container using padding utility, e.g. py-4
   - Update the navigation active styling from "Home" to "Reserve a Table"
3. Footer should float in the middle of the screen now, make it stick to bottom edge by adding "fixed-bottom" class
4. Create a <svg> element in container, the size of <svg> is 400 (W) x 100 (H)



5. Within the <svg> element, create 4 <circle> elements to represent each seat/table
   - Each circle has a class name of "table"
6. Set the `cx` and `cy` attributes to specify the center coordinates of each circle
7. Adjust the `r` attribute to control the size of the circle, e.g. 30
8. Add <style> tag in <head> section

9. Create a rule for "table" class
    - Set the fill color (`fill`) to any color (e.g. green)
    - Set the stroke color (`stroke`) of circle to any color (e.g. black)
    - Set the thickness of stroke (`stroke-width`) of outline to 1 – 3 pixels
    - Set the `cursor` property to pointer, so it can change the mouse cursor to a pointer when hovering over the element
10. Add text element (<text>) inside each circle to display the table number
    - https://www.w3schools.com/graphics/svg_text.asp
11. Set the `x` and `y` attributes to define the position of the text within the circle
12. Center the text horizontally and vertically within the circle
    - Set `text-anchor` attribute to "middle"
    - Set `alignment-baseline` attribute to "central"
13. Stylize text to white text color using `fill` attribute
14. Add a <div> element after container with a unique id (e.g. selected-table), it is for showing the selected table dynamically
    - Text should be center aligned
    - It should have a bigger font size and bolder than default (Use Bootstrap class names)
15. Add a <script> tag beneath <footer> element
    - As we would ensure that the HTML content is loaded before executing the JavaScript code
16. Create a function `selectTable(tableNumber)` in <script> tag
    - Take a `tableNumber` parameter as input
    - Retrieve the HTML element with the ID "selected-table"
    - Set the text content of the element to "You are selecting: Table " + tableNumber
17. Add `onclick` event handler to each <circle>, it triggers selectTable() function with providing table number as parameter

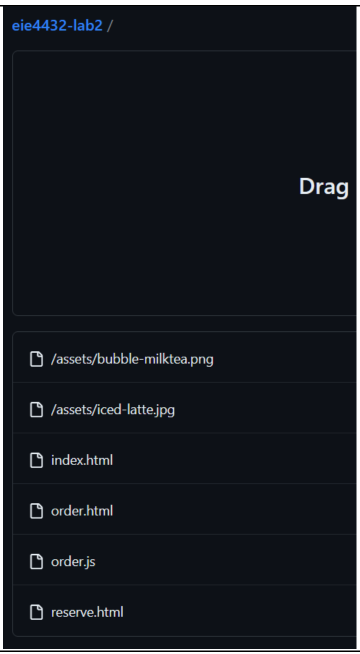
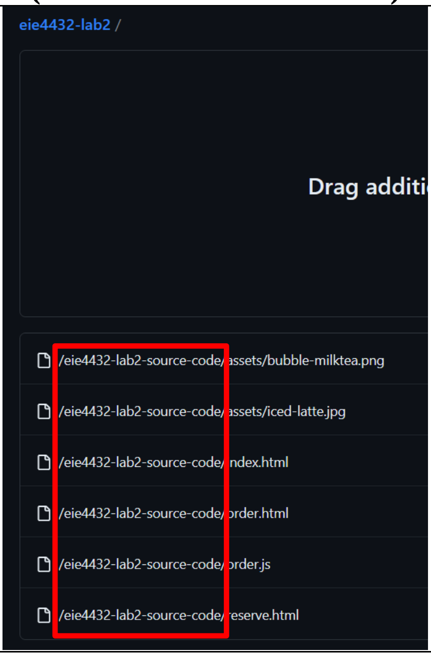
Lab #3.1

Provide the following screenshots with your **Student ID and System Date & Time.** Your code should be indented correctly (each nested tag should be indented exactly once inside of its parent tag)

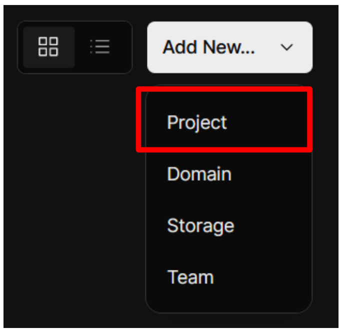
- Show the source code of "reserve.html"
- Show the screenshot of selecting 2 different tables

Task 4: Publish your Website (15 mins)

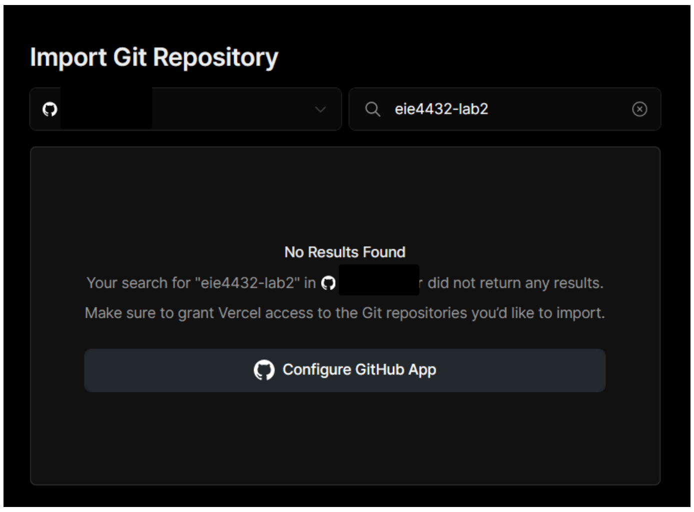
1. Make sure the source code is uploaded to GitHub as a new **PRIVATE** repository
   - Do not upload a whole folder to repository, it would affect how the web server access the files
   - Make sure there is a "index.html" file in your repository

Correct Practice (Without further folder name)	Wrong Practice (With further folder name)
 eie4432-lab2 / Drag /assets/bubble-milktea.png /assets/iced-latte.jpg index.html order.html order.js reserve.html	 eie4432-lab2 / Drag additi /eie4432-lab2-source-code/assets/bubble-milktea.png /eie4432-lab2-source-code/assets/iced-latte.jpg /eie4432-lab2-source-code/index.html /eie4432-lab2-source-code/order.html /eie4432-lab2-source-code/order.js /eie4432-lab2-source-code/reserve.html

2. Go to Vercel (https://vercel.com/) to sign up an account
   - This is a cloud platform for deploying and hosting websites and web applications
   - It specializes in static site generation and serverless functions
   - Offers seamless integration with popular frontend frameworks like Next.js and Nuxt.js
   - Provides features like automatic deployments, domain management, and scalability
   - Supports collaborative development and offers various deployment options, including Git-based workflows
3. You can sign up with **GitHub account** to allow the best integration with the platform
4. In the dashboard of Vercel, click “Add New…” dropdown menu, and select “Project”



5. Search the repository name in the search field, as it is a **PRIVATE** repository, it can’t be found directly, click “Configure GitHub App” to grant access for Vercel



Import Git Repository

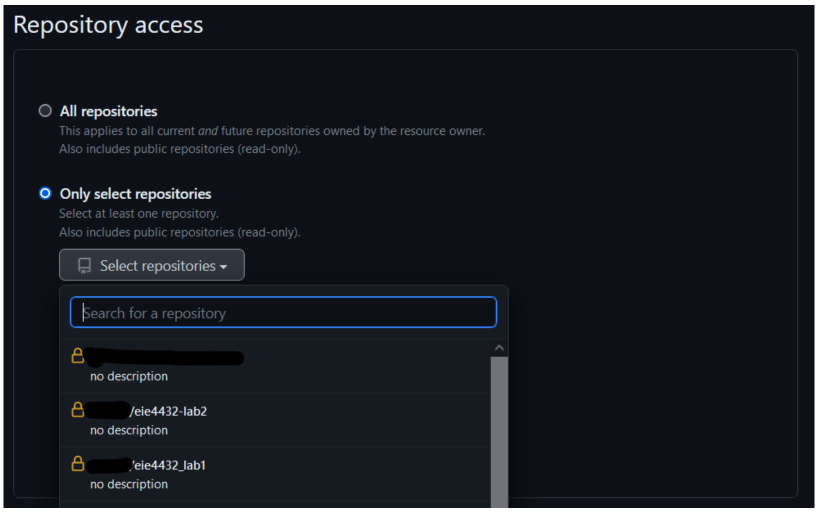
[redacted] eie4432-lab2

No Results Found

Your search for "eie4432-lab2" in [redacted] did not return any results.

Make sure to grant Vercel access to the Git repositories you'd like to import.

6. A pop-up window asks to enter password, after that, it will redirect to a setting page
7. Scroll to find "Repository access" section, select "All repositories" is the most convenient method
8. If you decided to select "Only select repositories", click "Select repositories" dropdown menu, and select the repository of your lab 2 project



Repository access

All repositories
This applies to all current *and* future repositories owned by the resource owner.
Also includes public repositories (read-only).

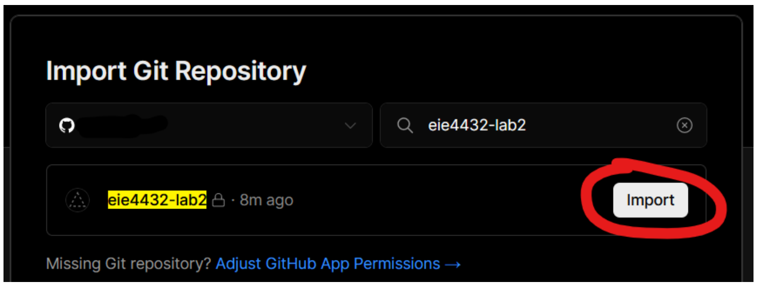
Only select repositories
Select at least one repository.
Also includes public repositories (read-only).

[redacted]
no description

[redacted]/eie4432-lab2
no description

[redacted]/eie4432_lab1
no description

9. After setting, click "Save" and return back to Vercel
10. Find the repository and click "Import"



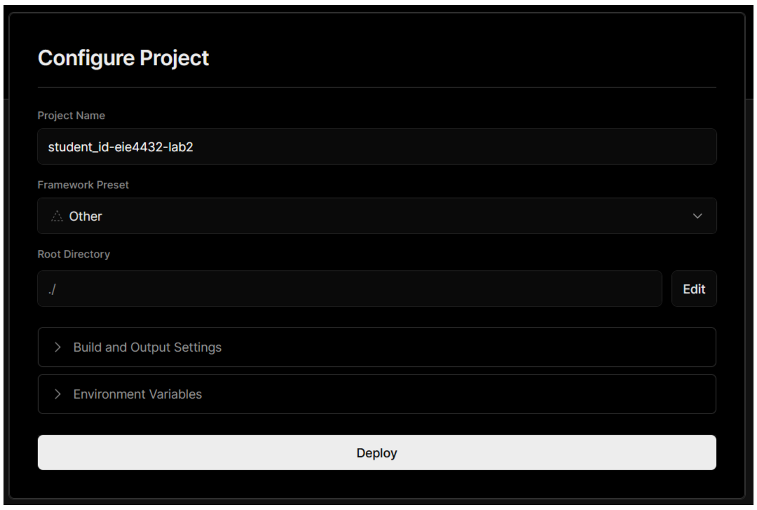
Import Git Repository

[redacted] eie4432-lab2

eie4432-lab2 · 8m ago

Missing Git repository? Adjust GitHub App Permissions →

11. Deploy the website as the following settings, click "Deploy" when everything is ready
    - Note: Uppercase letters & underscore are not allowed in project name



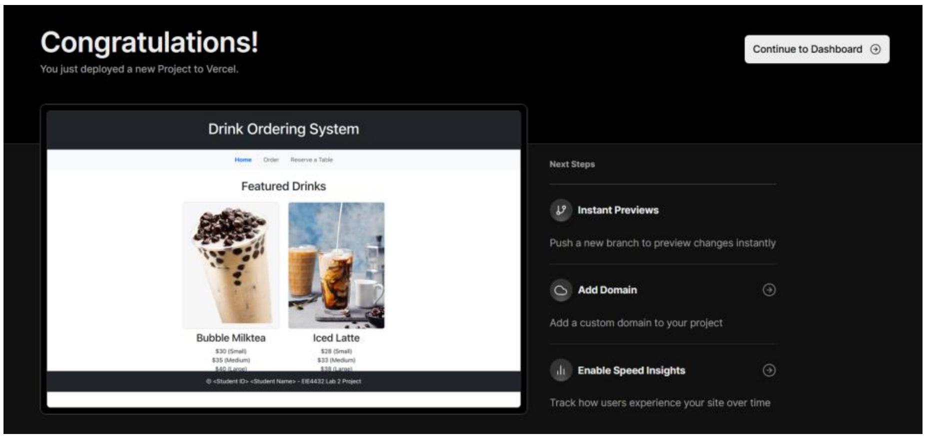
Configure Project

Project Name: student_id-eie4432-lab2

Framework Preset: Other

Root Directory: ./

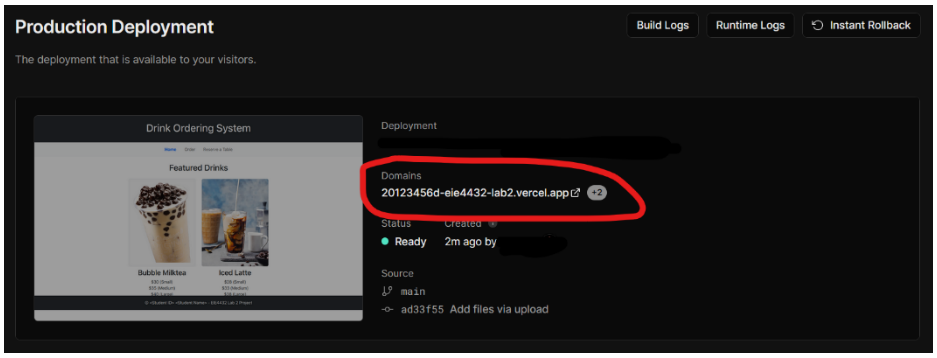
12. You will see following screen when it is successfully deployed, click "Continue to Dashboard" to find the public URL



Congratulations!

You just deployed a new Project to Vercel.

13. Public URL can be found under "Domains"



Production Deployment

The deployment that is available to your visitors.

Deployment: [redacted]

Domains: 20123456d-eie4432-lab2.vercel.app +2

Status: Ready

Created: 2m ago by [redacted]

Source: main

ad33f55 Add files via upload

Lab #4.1
Provide the public URL of the lab project, eg. https://xxx.vercel.app.

END OF LAB 2