

ICT239

Web Application Development

Tutor-Marked Assignment

January 2024 Presentation

TUTOR-MARKED ASSIGNMENT (TMA)

This assignment is worth 24% of the final mark for ICT239 - Web Application Development

The cut-off date for this assignment is **Tuesday, 12 March 2024, 2355hrs.**

Note to Students:

You **MUST use the provided solution template** accompanying this TMA.

Answer all questions. (Total 100 marks)

Section A

Question 1 concerns the Staycation case study.

Question 1 (35 marks)

- (a) Follow the steps stated as follows. While you are performing the steps, do a recording of you running through the steps to launch Staycation demo on Vocareum.
1. Login to Vocareum via Canvas, start VS Code and open the folder to ICT239 Past References/staycation/app
 2. Start a new terminal and setup the virtual environment and install packages
 - `python3 -m venv venv`
 - `. venv/bin/activate`
 - `pip install -r requirements`
 3. Start the app
 4. Start the browser to connect to the app
 5. Register an admin user, Admin, admin@abc.com, password:12345
 6. Login using the registered admin user account: admin@abc.com
 7. Click on 'Upload' at the sidebar to get the Upload page.
 8. Select data type Package and upload staycation.csv. Next, select data type Users and upload users.csv. Finally, select data type Booking and upload the file booking.csv. The files can be downloaded from ICT239 Past References/staycation/app/assets/js folder and to initialize the database for charting.
 9. Click on 'Dashboard' to display the chart.
 10. Use Compass on the desktop workspace to demonstrate in your video that the database contains the data of the user accounts, packages and bookings correctly.

(4 marks)

Your answer to parts (b) – (d) must show clear understanding of the various components in the Staycation application. In order to do well in this section, you must provide a thorough answer as well as highlight code by copying ONLY necessary code to be used in your explanation.

You should highlight small chunks of code and explain how each chunk contributes to your answer.

There is penalty if relevant code does not accompany an explanation for each sub-part or if irrelevant code is included in your explanation.

- (b) This question part refers to the code in the frontend component.
- (i) Explain how bootstrap positions the sidebar, the top panel and the centre panel of a web page in the Staycation application.
 - (ii) Explain how the web page can respond to various screen sizes. In particular, explain using the web page you get when you click on the link Packages on the side bar.
 - (iii) Explain 5 styling effects applied to the sidebar.
 - (iv) Explain jinja inheritance as applied in the staycation application for the page displayed for part (b) (ii).
- (15 marks)
- (c) In step 3 of part (a), the application starts, and in step 4 of part (a), the first web page is displayed.
- (i) Identify the html page displayed.
 - (ii) Explain the HTTP request that causes the first page to be displayed. Specifically, explain how, where the HTTP request is made, and the type of HTTP request made.
 - (iii) Explain the interaction between the frontend component and backend component that produces a HTTP response to cause the web page to be displayed. Show the relevant code and use the code to explain the interaction. For this sub-part, you should explain how flask handles URL routing in this situation and the HTTP response.
- (8 mark)
- (d) In step 8 of part (a), the user can upload files to the backend.
- (i) Explain why the view function upload caters to both GET and POST requests. Is it preferable to have two different view functions instead? Give reasons for your answer.
 - (ii) Explain how the backend accesses the additional data that accompanies the GET and POST HTTP requests, if any, for the upload function.
- (8 mark)

Section B

Questions 2 – 4 concern the development of a Web application for meal orders. The application scenario is based on ICT162 July 2023 semester TMA with some modifications.

Eat-Right-Programme (ERP) is a pilot programme of Healthier SG. The key objective of ERP is to encourage Singaporeans to take charge of their diets, leading to healthier lifestyles and see better health outcomes. Singaporeans above 12 years old can enrol in ERP.

To debunk the myth that Healthy food is expensive, bland, and boring. ERP will offer Delis that are deliciously prepared by food providers/agencies, using recipes with strict requirements on calories and fat content.

In ERP, participants can “Order” their meals by selecting Delis to form a DeliSet first. DeliSets are saved and can be repeatedly ordered.

This section will focus on

- displaying information about the ERP program, and the available cold and hot delis (questions 2a).

Refer to the appendix for code for classes that you can use to implement question2 (a). Subsequently, you are to modify the classes in the appendix for question 2(b) so that data is stored in MongoDB instead of class variables.

- creating an admin user account if the account is not yet created, registering non-admin users or participants, logging users in and out (question 2c).
- allowing participants to create deli sets, place order and view their orders (question 3).
- Allowing the admin user to upload participants, delis and orders (question 4).

You are required to **provide explanations** for **ALL** your implementations to show your understanding of the following items where applicable,

- ✓ the model, view and controller components in your implementation and the purpose of the components specifically for the question you are answering
- ✓ the application of HTML and CSS
- ✓ the application of Bootstrap
- ✓ the interactions amongst the frontend and backend components that you added.
- ✓ the application of jinja variables, template filters, statements, inheritance, macros etc
- ✓ any other explanation required for a specific part of a question

Question 2 (26 marks)

Learning objectives:

- Develop a HTML/CSS and Python web framework program
- Apply programming methods to present information in HTML
- Employ web programming framework for developing website

You are to **submit both implementations for part (a) and (b) as two separate applications.**

(a) About and Delis

Create a Flask application with a hyperlink About which leads to the home page shown in Figure Q2(a) (i).

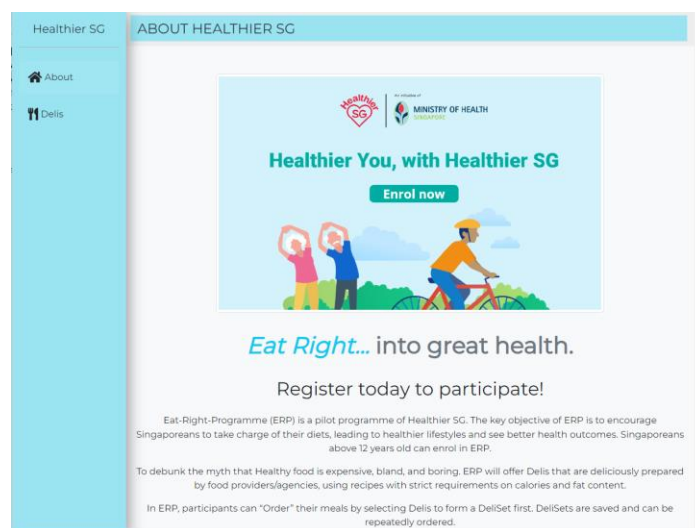


Figure Q2(a) (i): The home page or the About page

The Deli page shows the Cold Delis followed by the Hot Delis as shown in Figure Q2(a) (ii) and (iii).

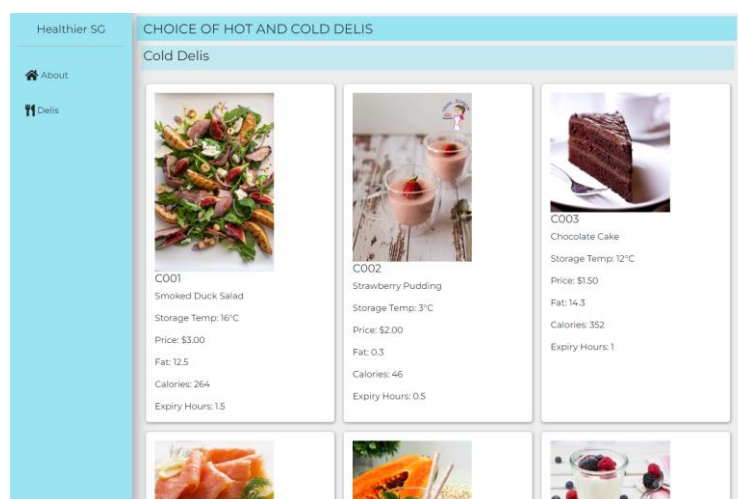


Figure Q2(a) (ii) The Delis page shows cold delis followed by hot delis

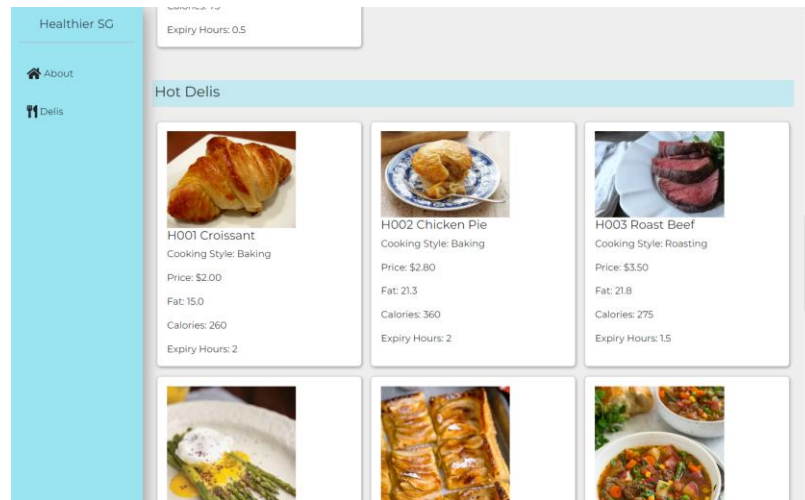


Figure Q2(a) (iii) The Delis page shows hot delis following the cold delis

You must apply responsive web design in your implementation of the pages so that the pages can adapt to various screen sizes. Figure 2(a) (iv) shows the About page on a small screen with side bar and top panel collapsed. When the side bar and top panel are expanded, they are similarly displayed as in Figure 2(a) (viii).



Figure Q2(a) (iv) The About page on small screen size

Figure 2 (a) (v) – (viii) show the Deli page on various screen sizes. Figure 2 (a) (viii) shows the Deli page on a small screen with side bar and top panel expanded. The same hamburger icon as in the About page will show instead when side bar and top panel are collapsed.

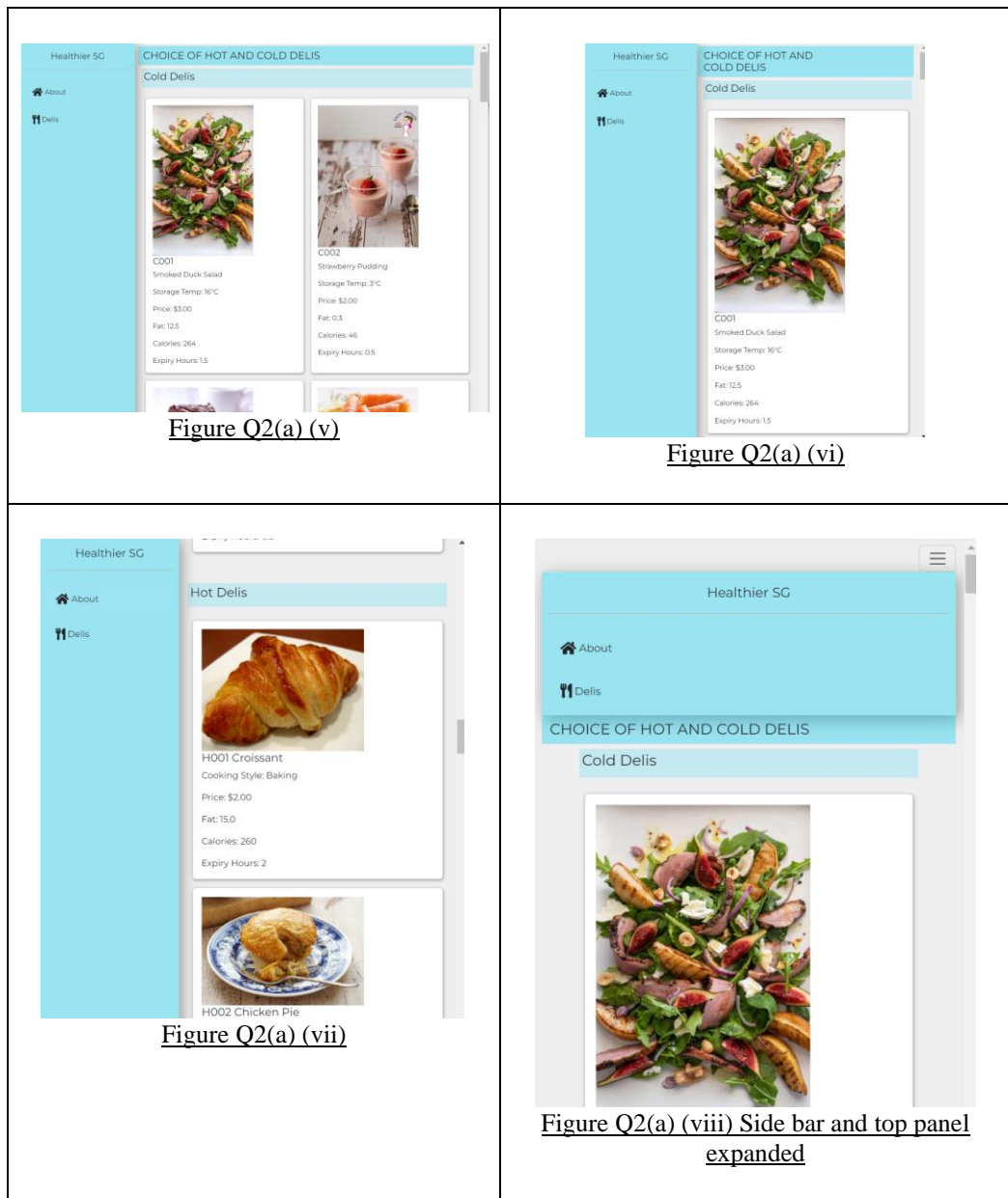


Figure Q2(a) (iv) The Deli page on various screen sizes

Figure Q2a(ix) shows how a link on the sidebar should look like when you hover over it.

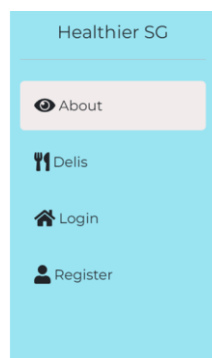


Figure Q2(a) (ix) The sidebar when you hover over a link

You must apply jinja and jinja inheritance in your implementation and to use the data in the class variables. Do NOT hardcode the data in the class variables in any html page.

(10 marks)

- (b) Make a copy of your implementation for part (a) so that you can modify the copy for part (b). Submit part (a) in a separate folder. Part (b) will be used for subsequent development for parts (b) and (c) of this question as well as for questions 3 and 4.

For part (b), you no longer use class variables to store data for the delis. Instead, use MongoDB database to store data about delis. Figure Q2(b) shows the complete class diagram for the application. The relevant classes for part (b) are Deli, ColdDeli and HotDeli.

Implement these three classes such that when a required collection in the database is empty, data is read from the global variable, `all_delis` and stored into MongoDB. You may design and implement any method you consider necessary for the classes to achieve the same effects as described in part (a).

Highlight and provide reasons for the changes you make to part (a) to implement part (b) using MongoDB database. Include in your answer the changes you made to the frontend and/or backend components of your flask application, including code to set up the database.

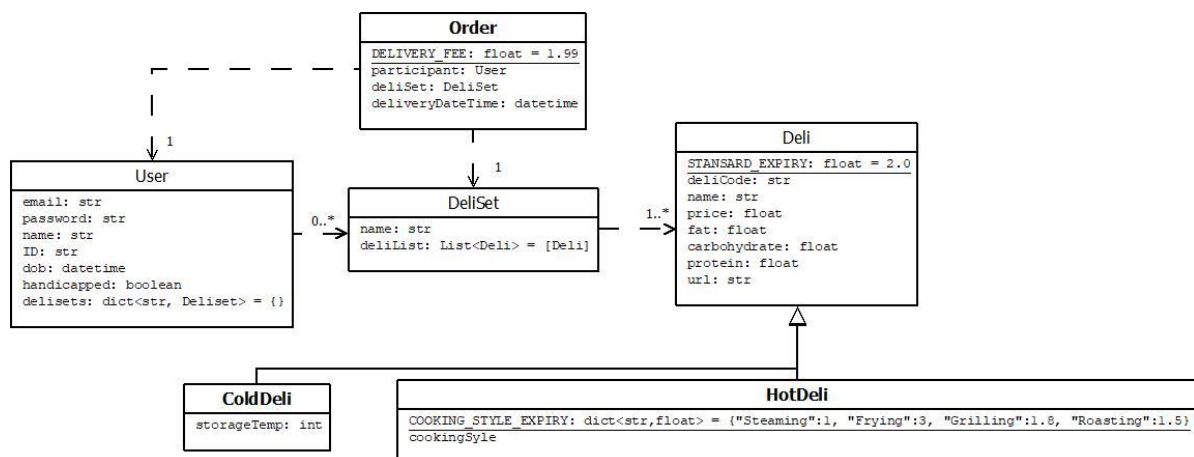


Figure Q2(b) Complete class diagram

(7 marks)

- (c) Register, Login and Logout

Refer to Figure 2(c) (i) and (ii) for the user interfaces for Register and Login respectively. For this question part, you will need to define the User class in Figure Q2(b).

Create the admin user (email: admin@abc.com, password 12345 and name Admin) if the user collection is empty. You can create the admin user either via Compass or through some application code. There is no need to include other details for admin user such as ID dob, handicapped and deli sets.

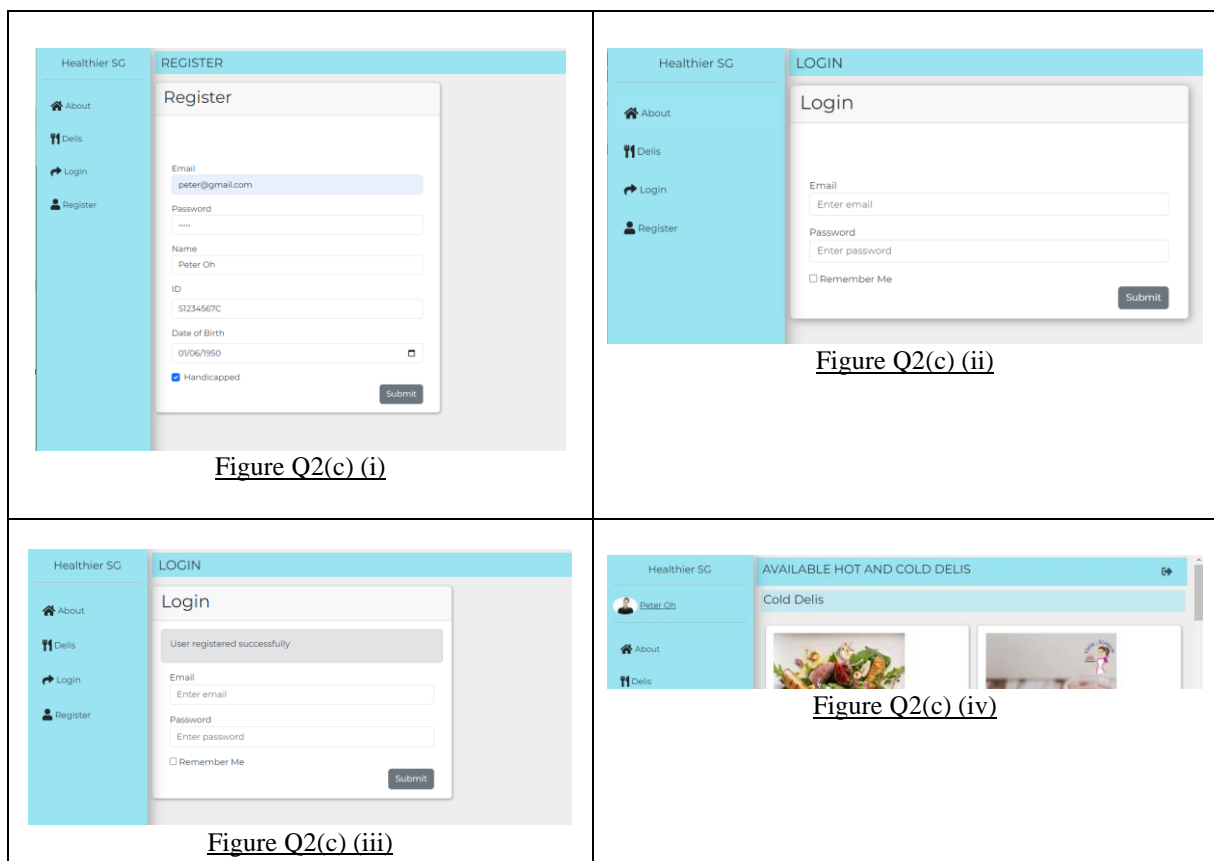
Similar to the Staycation case study, a participant (non-admin user) needs to be authenticated before he can use certain functions of the application. Therefore, he must register for an account. Upon successful registration, he will be directed to a login page with a flashed message displayed using Bootstrap alert as shown in Figure 2(c) (iii).

Handle unsuccessful registration and login in the same manner as the Staycation case study. Any user who successfully logs in will be directed to the Deli page.

To exit from the application, the user selects log out. After a user has logged out, the application should display the About page. The log out functionality is available on the top panel only after a successful login as shown in Figure Q2c (iv).

You may copy the relevant code in the Staycation case study for the register, login and logout functions to your application and make changes to achieve the new effects. Explain your modification.

Register a participant (email peter@gmail.com, password 12345, name Peter Oh, ID S1234567C, dob 1 Jun 1950, handicapped) and test that your functions are working correctly.



(9 marks)

Question 3 (24 marks)

Learning objectives:

- Apply programming methods to present information in HTML
- Employ web programming framework for developing website

This question focuses on the non-admin set functions: create delisets, order deliset and view orders. For this question, you must define the DeliSet and Order classes in Figure Q2(b). Ensure that the User class has been defined according to the class diagram in Figure Q2(b).

(a) Create DeliSet

The participant provides a name for the new deli set to be created. He can select the delis he wishes to add, and then click on the button to add the delis to the set. Figure Q3(a)(i) shows the top of the page for Create DeliSet and Figure Q3(a)(ii) shows the bottom of the page.

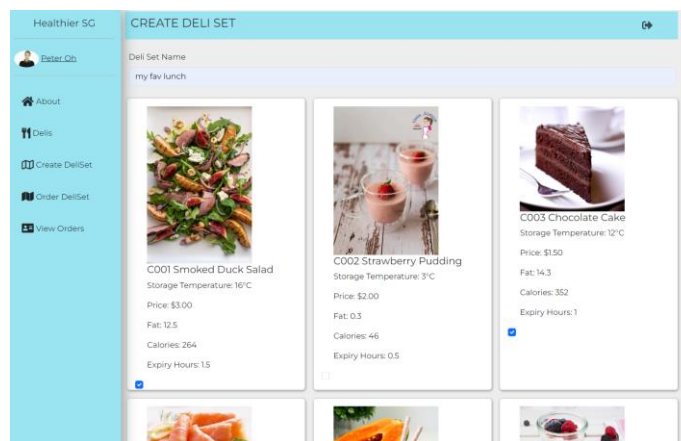


Figure Q3(a) (i) Top of Create DeliSet page with provided input

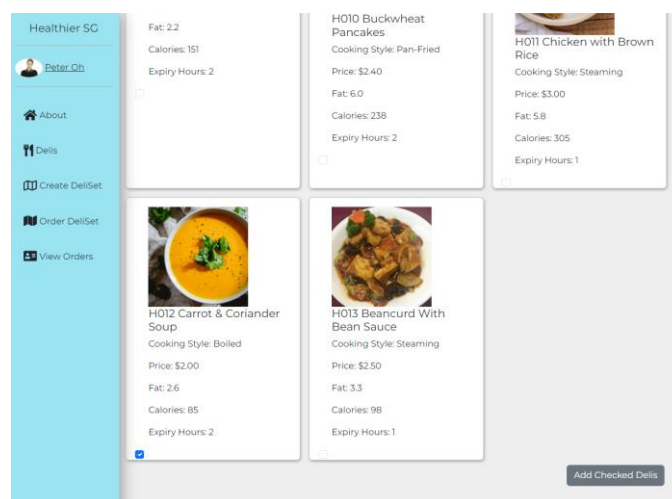


Figure Q3(a) (ii) Bottom of Create DeliSet page with provided input

The price of the deli set is the sum of each selected deli. The total fats and total calories are similarly computed. The consume in hours for a deli set is the minimum of the expiryHours attribute of delis in the deli set. The name of a deli set must have a different name from the names of a participant's existing deli sets. Provide the necessary instruction as shown in Figure Q3(a) (iii).

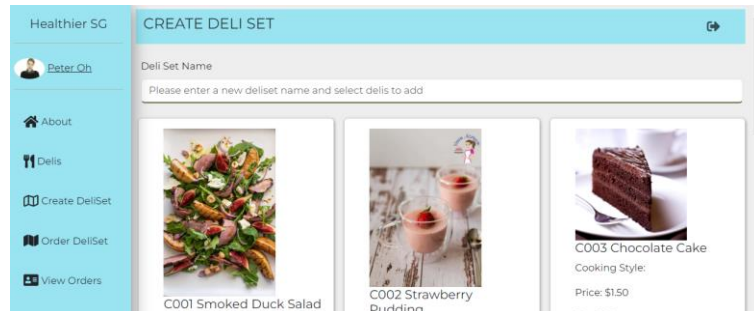


Figure Q3(a) (iii) Create DeliSet page with instruction

Flash appropriate messages such as

- Please use a different deli set name
Deli set name checks are case-insensitive. Therefore, My FAV lunch is considered same as My Fav Lunch.
- Please select at least one deli
A deli set must have at least one selected Deli which is added to a newly created deli set. Subsequent selected delis should be added one at a time to the newly-created deli set.
- Deli set inputName created with the selected delis: deliCode₁, ..., deliCode_n
The message is flashed if the deliset is successfully created and added to the dictionary of deli sets for the participant. Use the deli set name as key and Deliset object as value.

Flash the appropriate message when the button is clicked, and return to the same page. An example is shown in Figure Q3(a) (iv).

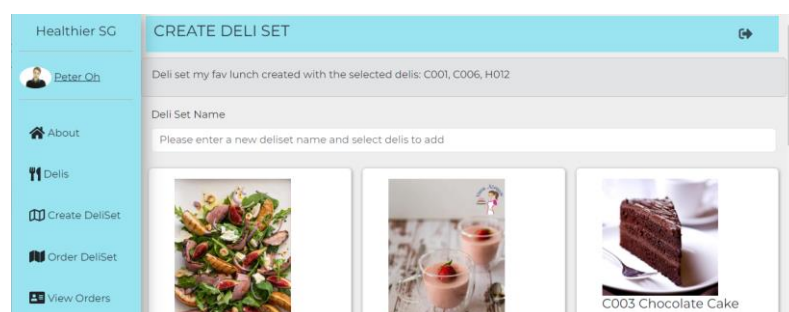


Figure Q3(a) (iv) Create DeliSet page with flashed message

(14 marks)

(b)

(i) Order DeliSet

When a participant has not created a deli set, he cannot place any order. Therefore, clicking on the hyperlink Order DeliSet shows a page as shown in Figure Q3(b) (i).

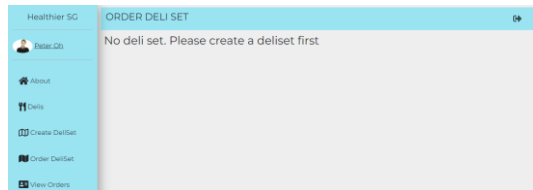


Figure Q3(b) (i) Order DeliSet page when participant has no deli set

If the participant has created deli sets, they are listed in alphabetical order, case insensitive. The participant provides a delivery datetime for a selected deli set and click on the corresponding button to place order.

Check that the same deli set must not have been ordered yet to be delivered at the same time. Flash an appropriate message when the button is clicked, and return to the same page. The two appropriate messages are as follows:

- Order deliset delisetName placed for deliveryDateTime
- Order failed. Deliset delisetName has already been placed for deliveryDateTime

An example is shown in Figure Q3(b) (ii).

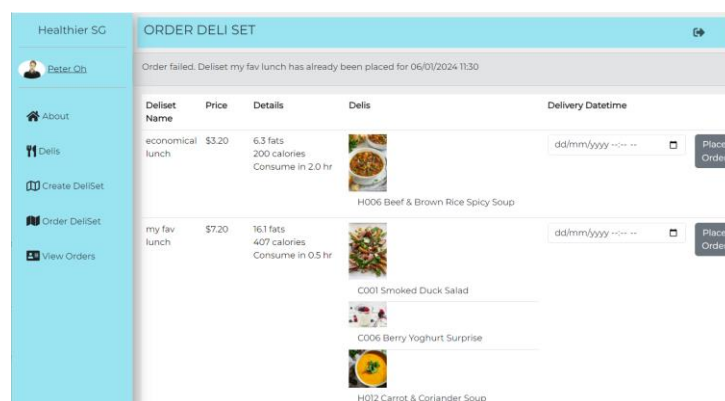


Figure Q3(b) (ii) Order DeliSet page with flashed message

(ii) View Orders

When a participant has not placed any order, clicking on the hyperlink Order DeliSet shows a page as shown in Figure Q3(b) (iii).

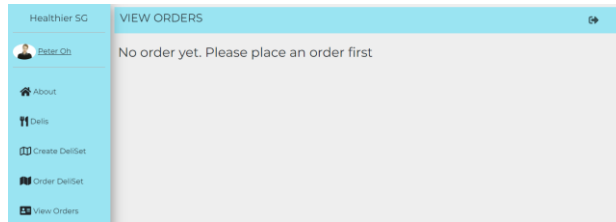


Figure Q3(b) (iii) View Order page when participant has no order

If the participant has placed some orders, they are listed in chronological order, starting from the later datetime, shown in Figure Q3b(iv). Delivery charge is waived for participants who are at least 75 years of age on the year the order is made or are handicapped at the time of order.







Delivery Date	Deliset Name	Price	Total Fats	Total Calories	Consumed In	Delis
07/01/2024 11:45	economical lunch	Deliset: \$3.20 Delivery: \$0.00 Total: \$3.20	6.3	200	2.0 hours	 H006 Beef & Brown Rice Spicy Soup
06/01/2024 11:30	my fav lunch	Deliset: \$7.20 Delivery: \$0.00 Total: \$7.20	16.1	407	0.5 hours	 C001 Smoked Duck Salad  C006 Berry Yoghurt Surprise  H012 Carrot & Coriander Soup
06/01/2024 11:29	my fav lunch	Deliset: \$7.20 Delivery: \$0.00 Total: \$7.20	16.1	407	0.5 hours	 C001 Smoked Duck Salad  C006 Berry Yoghurt Surprise

Figure Q3(b) (iv) View Order page when participant has orders

(10 marks)

Question 4 (15 marks)

Learning objectives:

- Employ web programming framework for developing website

Figure Q4 (i) shows the Upload page. The Upload function is available to only the admin user. To access the function, the admin user must log in. The Deli page will show up, and then, he must click on the Upload link on the sidebar. Note that the admin user need not register himself.

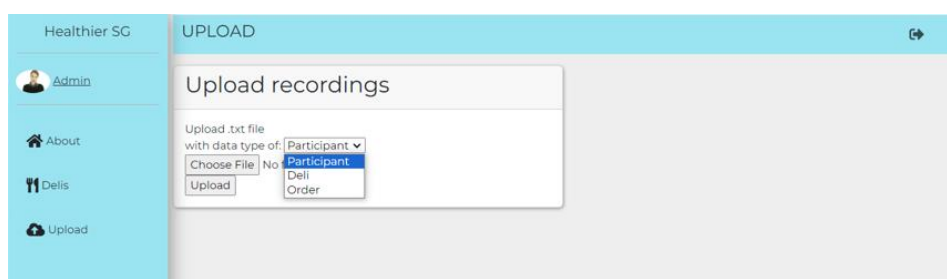


Figure Q4 (i) Upload Page

Once a data type and a file are chosen, the admin user clicks on Upload to activate the upload of data to the backend to be stored in MongoDB.

Data type	Fields, example and messages
Participant	<p>ID,name,dob,handicapped,email S4428006Z,Tan Yan Meng,19480708,False,tanym@gmail.com</p> <p>Flash message when a participant's email is already in database:</p> <ul style="list-style-type: none"> • Participant with email someEmail already exists
Deli	<p>#cold deli deliCode,name,price,fat,carbohydrates,protein,storageTemperature,url C001,Smoked Duck Salad,3.0,12.5,23,15,16,https://picniclifestyle.com/wp-content/uploads/2020/10/Grilled-pear-salad-6.jpg</p> <p>#hot deli deliCode,name,price,fat,carbohydrates,protein,cookingStyle,url H001,Croissant,2.0,15,26.7,4.6,Baking,https://3.bp.blogspot.com/-HMLcbz567qg/TlgWSbpD1uI/AAAAAAACgM/1xnYGFTZemE/s1600/croissant.JPG</p> <p>Flash message when a delicode is already in database:</p> <ul style="list-style-type: none"> • Deli someDeliCode already exists!
Order	<p>#Order email,deliveryDateTime,name,deliList wongyh@gmail.com,2024-01-04 09:00,My Breakfast,H001,C001,H008 peter@gmail.com,2024-01-04 09:00,My fav lunch</p> <p>Flash messages:</p> <ul style="list-style-type: none"> • Participant with email someEmail has not registered! when participant in data line with email not registered in the application.

	<ul style="list-style-type: none"> Order for same deliset someDeliSetName, delivery datetime someDeliveryDateTime by participant with email someEmail already exists when the data line is for an existing order, that is, for the same participant, the same deli set and the same delivery time. deliset someDeliSetName does not exist and cannot be created without delicode when a deli set does not exist yet and there is no deli code in the data line. <p>Note that if the data line contains deli code, then a deli set can be created for a participant, and an order is then placed.</p> <p>However, if a deli set already exists for a participant, it is not necessary for a data line to list the deli codes in the deli set. If there are deli codes in the data line for an existing deli set, the deli codes are ignored. That is, there is no need to update the existing deli set with the deli codes in the data line.</p>
--	---

Flash the number of records created at the end of each file upload. Refer to Figure Q4 (ii) for an example.

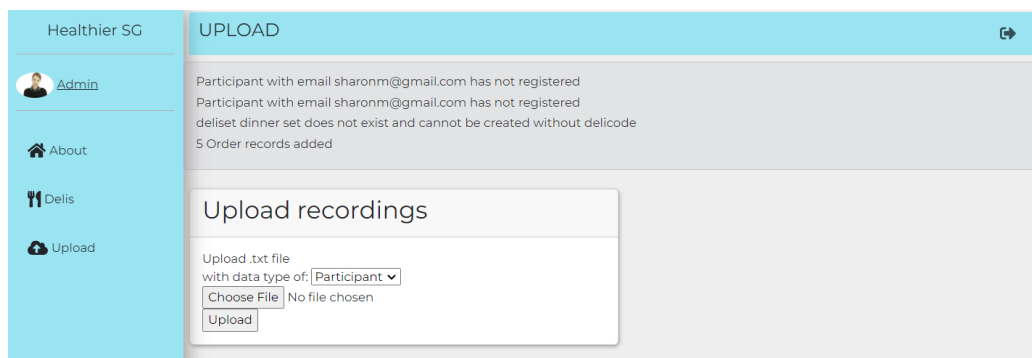


Figure Q4 (ii) Upload Page after an upload

Extend the flask application to allow admin user to upload data as described.

(15 marks)

Appendix

```
all_delis = [{ 'code': 'C001', 'name': 'Smoked Duck Salad', 'price': 3.0, 'fat': 12.5, 'carbohydrate': 23.0, 'protein': 15.0, 'styleOrTemp': '16', 'url': 'https://picniclifestyle.com/wp-content/uploads/2020/10/Grilled-pear-salad-6.jpg' }, { 'code': 'C002', 'name': 'Strawberry Pudding', 'price': 2.0, 'fat': 0.3, 'carbohydrate': 10.0, 'protein': 0.7, 'styleOrTemp': '3', 'url': 'https://veenaazmanov.com/wp-content/uploads/2014/05/Homemade-Strawberry-Pudding-15.jpg' }, { 'code': 'C003', 'name': 'Chocolate Cake', 'price': 1.5, 'fat': 14.3, 'carbohydrate': 50.7, 'protein': 5.0, 'styleOrTemp': '12', 'url': 'https://www.joannasteven.com/wp-content/uploads/2015/08/GFchocolatecake-1024x1024.jpg' }, { 'code': 'C004', 'name': 'Smoked Salmon', 'price': 2.4, 'fat': 3.7, 'carbohydrate': 0.0, 'protein': 16.0, 'styleOrTemp': '3', 'url': 'https://goodtastefoods.co.uk/wp-content/uploads/2019/11/smoked-salmon_143296537-1.jpg' }, { 'code': 'C005', 'name': 'Chilled Papaya Smoothie', 'price': 2.2, 'fat': 2.1, 'carbohydrate': 21.3, 'protein': 7.0, 'styleOrTemp': '8', 'url': 'https://insanelygoodrecipes.com/wp-content/uploads/2022/09/Healthy-Papaya-Smoothie-with-Oat-Flakes-and-Cinnamon-683x1024.jpg' }, { 'code': 'C006', 'name': 'Berry Yoghurt Surprise', 'price': 2.2, 'fat': 1.0, 'carbohydrate': 9.2, 'protein': 3.0, 'styleOrTemp': '3', 'url': 'https://www.pcrm.org/sites/default/files/berries-nondairy-yogurt.jpg' }, { 'code': 'C007', 'name': 'Beetroot Raita', 'price': 2.0, 'fat': 1.0, 'carbohydrate': 12.4, 'protein': 3.5, 'styleOrTemp': '8', 'url': 'https://werecipes.com/app/uploads/2015/04/dahi-raita-beetroot-raita-recipe.jpg' }, { 'code': 'H001', 'name': 'Croissant', 'price': 2.0, 'fat': 15.0, 'carbohydrate': 26.7, 'protein': 4.6, 'styleOrTemp': 'Baking', 'url': 'https://3.bp.blogspot.com/-HMLcbz567qg/TlgWSbpD1uI/AAAAAAAAACgM/1xnYGFTZemE/s1600/croissant.JPG' }, { 'code': 'H002', 'name': 'Chicken Pie', 'price': 2.8, 'fat': 21.3, 'carbohydrate': 32.0, 'protein': 10.0, 'styleOrTemp': 'Baking', 'url': 'https://img.taste.com.au/94eodL40/taste/2016/11/creamy-chicken-and-leek-pies-80359-1.jpeg' }, { 'code': 'H003', 'name': 'Roast Beef', 'price': 3.5, 'fat': 21.8, 'carbohydrate': 0.0, 'protein': 19.6, 'styleOrTemp': 'Roasting', 'url': 'https://i2.wp.com/www.domesticate-me.com/wp-content/uploads/2013/12/slow-roasted-beef-tenderloin-with-rosemary-13.jpg' }, { 'code': 'H004', 'name': 'Asparagus & Poached Egg', 'price': 3.0, 'fat': 4.9, 'carbohydrate': 3.1, 'protein': 7.9, 'styleOrTemp': 'Boiled', 'url': 'https://www.healthbenefitstimes.com/recipe/wp-content/uploads/2018/07/Roasted-Asparagus-with-Poached-Eggs-and-Hollandaise-Recipe.gif' }, { 'code': 'H005', 'name': 'Apple Sesame Tart', 'price': 1.8, 'fat': 7.5, 'carbohydrate': 49.1, 'protein': 4.9, 'styleOrTemp': 'Baking',
```



```

    'url':
    'https://i.pinimg.com/736x/fc/22/c0/fc22c0106471cd2939ca371bfc80c38e.jpg'},
{'code': 'H006', 'name': 'Beef & Brown Rice Spicy Soup', 'price': 3.2, 'fat':
6.35,
 'carbohydrate': 17.7, 'protein': 18.0, 'styleOrTemp': 'Boiled',
 'url':
 'https://i.pinimg.com/originals/f6/78/41/f678414dccf92a0af0a8f09e3572caa1.jpg'},
{'code': 'H007', 'name': 'Fish & Potato Pie', 'price': 3.0, 'fat': 5.5,
 'carbohydrate': 15.0,
 'protein': 16.5, 'styleOrTemp': 'Baking',
 'url': 'https://www.sprinklesandsprouts.com/wp-content/uploads/2019/10/Fish-Pie-
4.jpg'},
{'code': 'H008', 'name': 'Banana Pancakes', 'price': 2.0, 'fat': 7.0,
 'carbohydrate': 20.0,
 'protein': 3.6, 'styleOrTemp': 'Pan-Fried',
 'url': 'https://3.bp.blogspot.com/-
BTNuMoVzv3Y/Vkq1QEgta_I/AAAAAAAAANDY/19_4zLnWfcc/s1600/Buttermilk%2BBanana%2BPancak
es%2Brecipe.jpg'},
{'code': 'H009', 'name': 'Boiled Banana with Grated Coconut', 'price': 2.0, 'fat':
2.2,
 'carbohydrate': 30.7, 'protein': 2.0, 'styleOrTemp': 'Boiled',
 'url': 'https://thumbs.dreamstime.com/b/boiled-slice-banana-eat-coconut-food-
diet-154458927.jpg'},
{'code': 'H010', 'name': 'Buckwheat Pancakes', 'price': 2.4, 'fat': 6.0,
 'carbohydrate': 39.0,
 'protein': 7.0, 'styleOrTemp': 'Pan-Fried',
 'url': 'https://bakeitpaleo.com/wp-content/uploads/2021/05/paleo-buckwheat-
pancake-recipe.jpg'},
{'code': 'H011', 'name': 'Chicken with Brown Rice', 'price': 3.0, 'fat': 5.8,
 'carbohydrate': 48.3,
 'protein': 15.0, 'styleOrTemp': 'Steaming',
 'url': 'https://dinnerthendessert.com/wp-content/uploads/2019/05/Baked-Chicken-
and-Rice.jpg'},
{'code': 'H012', 'name': 'Carrot & Coriander Soup', 'price': 2.0, 'fat': 2.6,
 'carbohydrate': 13.8,
 'protein': 1.6, 'styleOrTemp': 'Boiled',
 'url': 'https://i1.wp.com/cookingwithbry.com/wp-content/uploads/2019/11/Carot-
Coriander-Soup-Recipe-17.jpg'},
{'code': 'H013', 'name': 'Beancurd With Bean Sauce', 'price': 2.5, 'fat': 3.3,
 'carbohydrate': 6.8,
 'protein': 10.2, 'styleOrTemp': 'Steaming',
 'url': 'https://www.pekinghouse.co.nz/wp-content/uploads/2015/10/DeepFriedBean-
CurdinChiliBeanSauce-600x600.jpg'}}]

```

```

from abc import ABC, abstractmethod
class Deli(ABC):
    _STANDARD_EXPIRY = 2
    delis = {}
    def __init__(self, deliCode, name, price, fat, carbohydrates, protein, url):
        self.deliCode = deliCode
        self.name = name
        self.price = price
        self.fat = fat
        self.carbohydrates = carbohydrates
        self.protein = protein
        self.url = url

    @property
    def calories(self):

```

```

        return self.fat * 9 + self.carbohydrates * 4 + self.protein * 4

    @abstractmethod
    def expiryHours(self):
        return type(self)._STANDARD_EXPIRY

    def __str__(self):
        return f"{self.deliCode:4s} - {self.name:35s} Price:
    ${self.price:.2f} Fat: {self.fat:>4.1f} Calories: {self.calories:>5.1f}"

    @classmethod
    def getAllDelis(cls):
        if not cls.delis:
            for dataDict in all_delis:
                if dataDict['code'].startswith('H'):
                    deli = HotDeli(dataDict['code'], dataDict['name'],
dataDict['price'], dataDict['fat'], dataDict['carbohydydrate'],
dataDict['protein'], dataDict['styleOrTemp'], dataDict['url'])
                else:
                    deli = ColdDeli(dataDict['code'], dataDict['name'],
dataDict['price'], dataDict['fat'], dataDict['carbohydydrate'],
dataDict['protein'], int(dataDict['styleOrTemp']), dataDict['url'])

                cls.delis[dataDict['code']] = deli
            return cls.delis.values()

    @classmethod
    def getDeliType(cls, deliType):
        if not deliType: return []
        return [d for d in cls.getAllDelis() if
d.deliCode.startswith(deliType[0].upper())]

    @classmethod
    def getDeli(cls, code):
        return cls.getAllDelis().get(code)

class ColdDeli(Deli):
    def __init__(self, deliCode, name, price, fat, carbohydrates, protein,
storageTemperature, url):
        super().__init__(deliCode, name, price, fat, carbohydrates, protein, url)
        self.storageTemperature = storageTemperature

    def expiryHours(self):
        if self.storageTemperature <= 10:
            return 0.5
        elif self.storageTemperature <= 15:
            return 1
        elif self.storageTemperature <= 20:
            return 1.5
        else:
            return super().expiryHours()

class HotDeli(Deli):
    _COOKING_STYLE_EXPIRY = {"Steaming":1, "Frying":3, "Grilling":1.8,
"Roasting":1.5}

    def __init__(self, deliCode, name, price, fat, carbohydrates, protein,
cookingStyle, url):
        super().__init__(deliCode, name, price, fat, carbohydrates, protein, url)

```

```
        self.cookingStyle = cookingStyle

    def expiryHours(self):
        return type(self)._COOKING_STYLE_EXPIRY.get(self.cookingStyle,
type(self)._STANDARD_EXPIRY)
```

---- END OF ASSIGNMENT ----