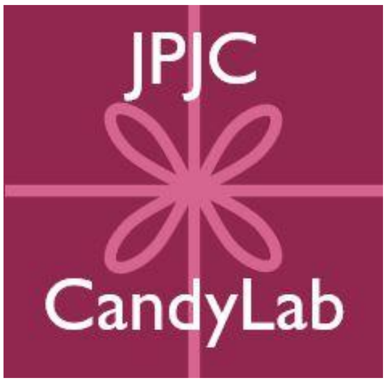
Web Application with Database Project
JPJC Candy Lab

INTRODUCTION AND AIMS TO THE PROJECT
There are two parts to this project.

In **part A,** you will apply your knowledge to **design a normalised relational database** for a web application.

In **part B**, you will proceed with the **development** of **the database and web application**.



LEARNING OUTCOMES FOR PROJECT
- Apply knowledge on relational database to solve a real-world problem.
- Create a detailed proposal for your product. Include the following components in your proposal:
  - Database design
    - table descriptions
    - ER diagram
- Demonstrate skills such as critical and inventive thinking and self-directed learning.
- Apply good time management.

Any form of plagiarism will be considered as serious violation of academic honesty.

SUBMISSION DEADLINE
By: 3 May 2024 (Fri)

Part A folder containing:
1. Table description (for only 3NF)
2. ER diagram (for only 3NF)

Part B folder containing:
1. Database file (JPCandyLab.db)
2. Python file (.py) – create DB/insert
3. Python Flask file (.py) – web app
4. Text files
5. **templates** folder containing html files
6. **static** folder containing image and css files

Submit Part A & Part B folders in a zip file:
<studentName>_WebAppDBProject.zip

PROJECT BRIEF
JP Candy Lab has a web application for customers to order Candies online. Customers may sign up as members and log in to enjoy 10% discount on their orders. Membership is free. Otherwise, customers proceed to order as guests with no discount.

For this Web App with Database Project, the company wants you to migrate the data from the text files (inventory.txt and member.txt) to a relational database and develop a web application for customer usage.

You are given:

i. inventory.txt
`<item_id>, <item_name>, <description>, <price>, <available_quantity>,<image_filename>`

ii. member.txt
`<mem_name>, <email>, <password>`

iii. transactionlog.txt
`<order_id>, <customer_name>, <date>, <item1_name>, <quantity1>, <item2_name>, <quantity2>, <item3_name>, …, <is_member>, <total_payable>`

iv. image files for all 5 items in the inventory, and JPJC Candy Lab Logo.

Part A

The relational database that you are required to **design** and **normalise** should store the following:

1. data about the inventory (refer to `inventory.txt`)

2. data about member (refer to `member.txt`).
   The `email` will be used as the login ID since it is unique.

3. data about each order (refer to `transactionlog.txt`)
   The `is_member` field is a Boolean stored as integer 0 (False) or 1 (True).

Submit your answer for:
1. Table description (for only 3NF)
2. ER diagram (for only 3NF)

PART B

1. Write the Python program to develop the normalised database **JPCandyLab.db** as follows:
   - Create tables
   - Read from the given files **inventory.txt** and **member.txt**, and insert the records into the database

2. The web application can perform the following:
   - Display information of available candy item with pictures (in table) for customer order
   - Member login, or order as Guest
   - Display order confirmation (in table) with 10% discount for members
   - Store transaction records
   - New member account creation
   - Successful/unsuccessful signup message

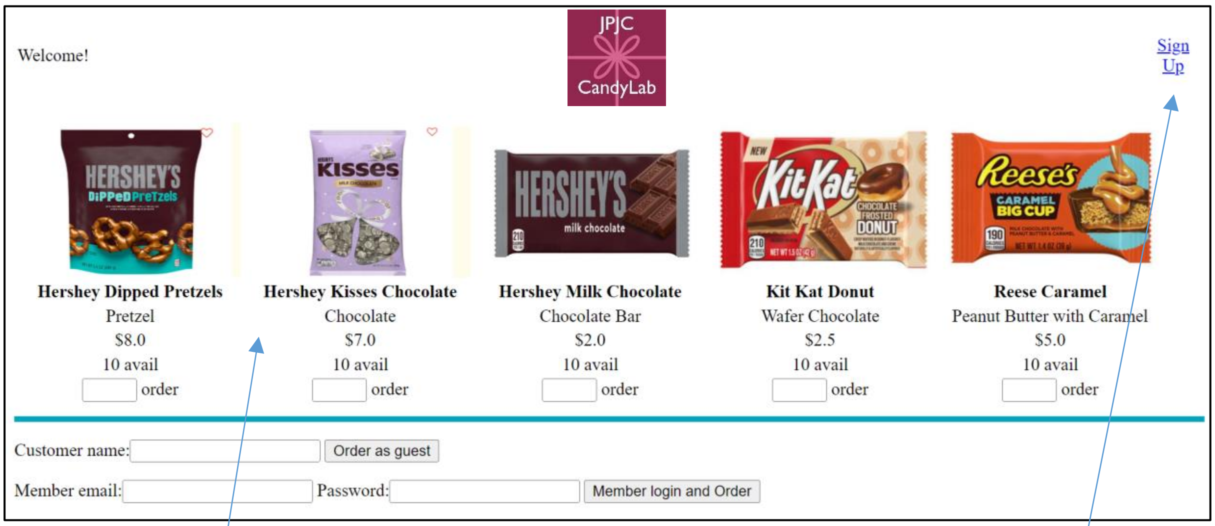
Use the Python programming, Flask and SQLite database to develop the web application.

Refer to Annex A for screenshots of sample pages. You are to implement the web application according to the design given.

Annex A

1. Order Page (Home page)

For ordering of candies, 5 types.



Welcome!

JPJC CandyLab

Sign Up

Hershey Dipped Pretzels	Hershey Kisses Chocolate	Hershey Milk Chocolate	Kit Kat Donut	Reese Caramel
Pretzel	Chocolate	Chocolate Bar	Wafer Chocolate	Peanut Butter with Caramel
$8.0	$7.0	$2.0	$2.5	$5.0
10 avail	10 avail	10 avail	10 avail	10 avail
order	order	order	order	order

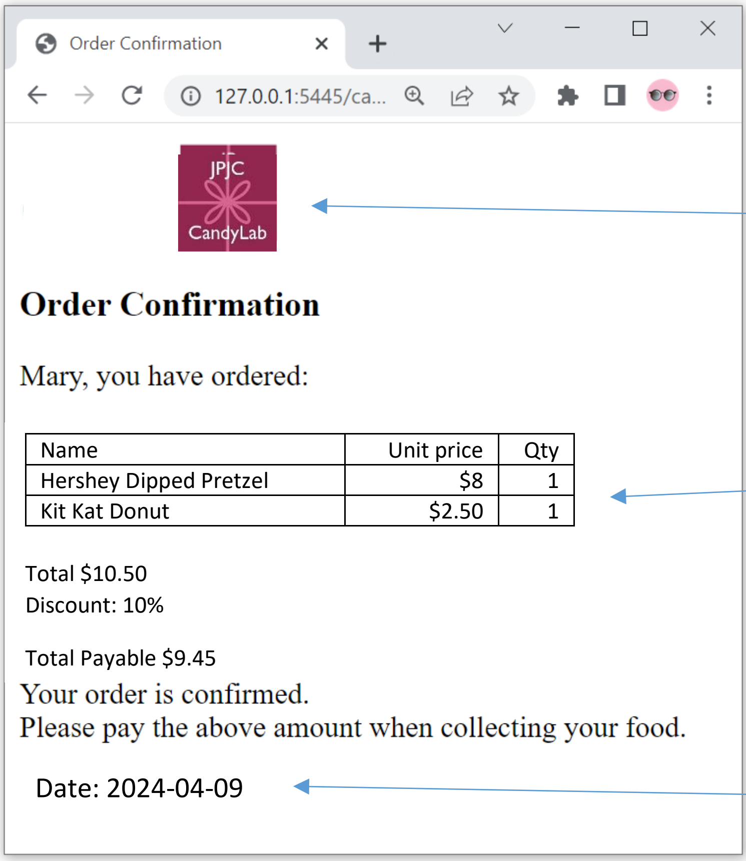
Customer name: [Order as guest]

Member email: Password: [Member login and Order]

Display candy information in a table, with no border.

Link to membership sign up page

2. Order Confirmation page



Order Confirmation

Mary, you have ordered:

Name	Unit price	Qty
Hershey Dipped Pretzel	$8	1
Kit Kat Donut	$2.50	1

Total $10.50
Discount: 10%

Total Payable $9.45

Your order is confirmed.
Please pay the above amount when collecting your food.

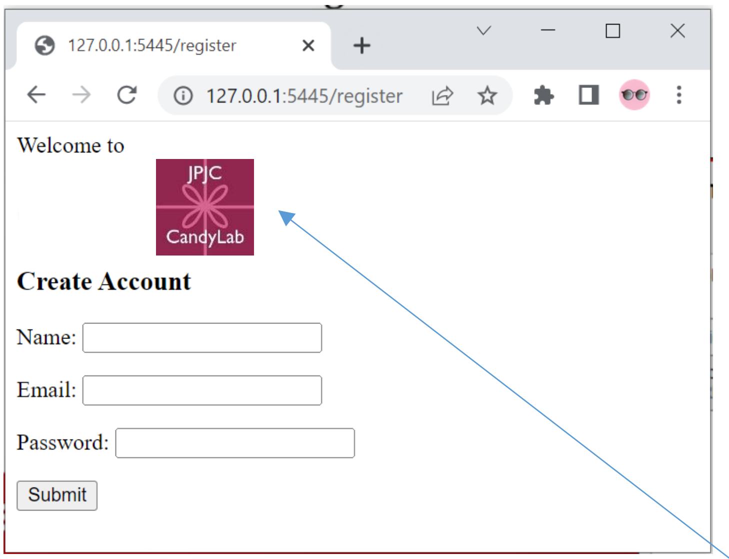
Date: 2024-04-09

Clicking on the JPJC Candy Lab logo will go back to the order page

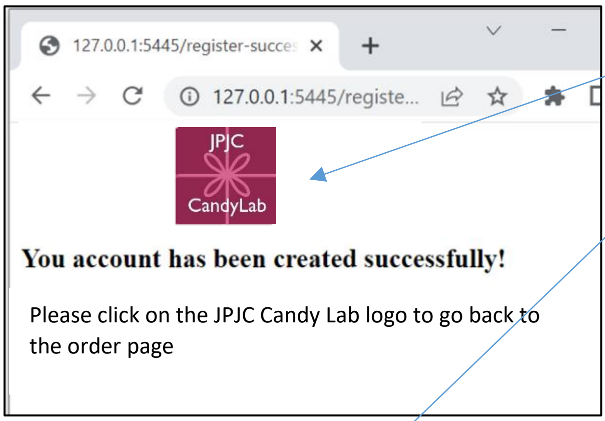
Please display this in a table

Please display date of order

3. Account signup page



Signup successful page



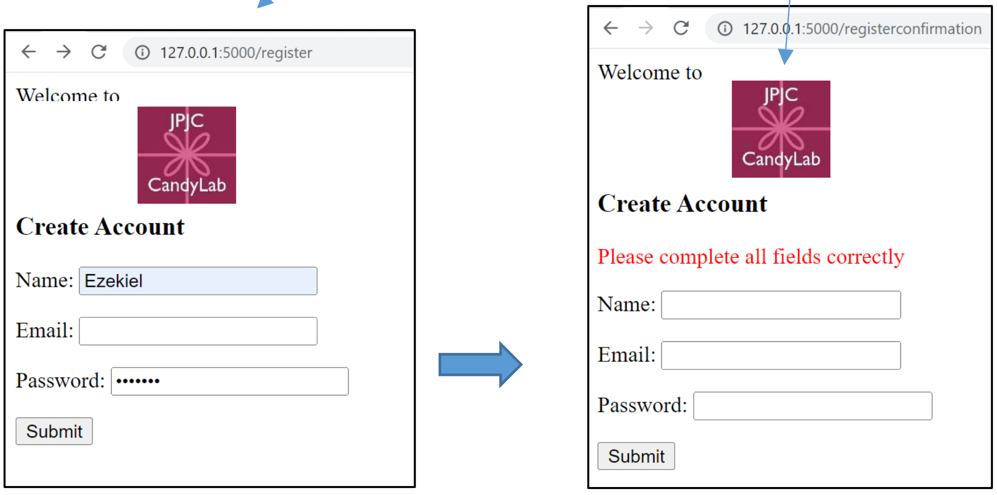
You account has been created successfully!

Please click on the JPJC Candy Lab logo to go back to the order page

Clicking on the JPJC Candy Lab logo will go back to the order page

Signup Unsuccessful page (due to invalid inputs)

Eg. Missing email address



Annex B

1. Display Image

Place image in `static` folder.

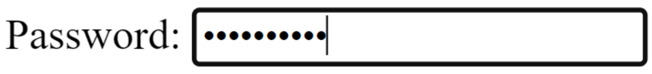
Place html files in `templates` folder.

To display image from the html file, use the html `img` image tag.

Set the `width` and `height` of your image using the attributes. Example,

```
<img src= "static/HersheyDippedPretzel.jpg" width="200"
height="140">
```

2. Password Field



Use the input type `password` to create the password field, where input text is obscured so it cannot be read. Usually each character is replaced by * or dot ("•") depending on the browser and operating system. The above is created using the html codes below.

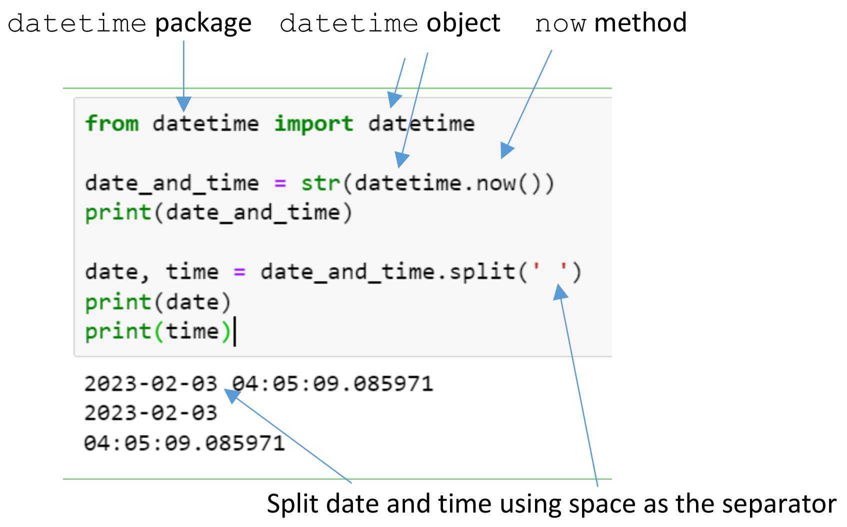
```
Password: <input type="password" name="pswd">
```

3. How to read Auto-increment numbers

```
# retrieve last orderID (last autoincrement number)

cursor = connection.execute("select seq from sqlite_sequence")
```

4. Current Date and Time

From the Python's `datetime` package, its `datetime` object has a `now`() method which will give the current date and time.



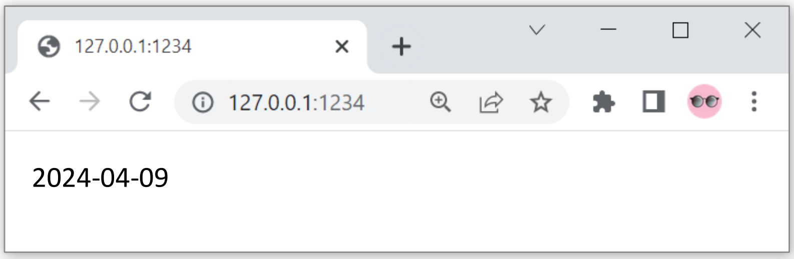
```
from datetime import datetime

date_and_time = str(datetime.now())
print(date_and_time)

date, time = date_and_time.split(' ')
print(date)
print(time)
```

```
2023-02-03 04:05:09.085971
2023-02-03
04:05:09.085971
```

Putting the codes in your flask server program,

```
from datetime import datetime
from flask import *

app = Flask(__name__)

@app.route("/")
def home():

    date_and_time = str(datetime.now())

    date, time = date_and_time.split(' ')
    return date

if __name__ == "__main__":
  app.run(host= "127.0.0.1", port=1234, debug=True)
```

Output:


127.0.0.1:1234

2024-04-09