Queens College, CUNY, Department of Computer Science

Software Engineering

CSCI 370

Spring 2019

Instructor: Dr. Sateesh Mane

Project 2
May 18, 2019

*** draft project proposal, open to student comments ***

- **Any student who submits Assignment 7 late will receive a grade cap of D for Project 2.**
- Submit your solution in a zip archive with the following name.

  `StudentId_first_last_CS370_project2_Spring2019.zip`

  1. Your project zip archive must contain all your program source code.
  2. Your project zip archive must contain a cover document explaining what you have done.
  3. The cover document must contain instructions how to run your code on my computer.
  4. If you use Java, include a jar file, else I must be able to load and run your package in Eclipse.

- **All students work individually for this project, there are no teams.**
- (Statement about academic policy and cheating, etc.)
- (Other administrative statements.)

1 General comment

- The basic theme of this project is that it is about something you see all around you.
- The purpose of Assignment 7 was to demonstrate that you have all submitted practical working examples of online restaurant menus.
- You have working examples of commercial software to use as a guide/template for what the end product of your work should look like.
- **Bear in mind (see Assignment 7) that the user only sees the web pages of the restaurants and menus.**
  The underlying implementation details do not matter.
  Hence the choice of programming language and database, etc. is up to you.
  As long as I can run your application on my computer.

2 Timetable

- The proposed due date is at the end of the semester.
- The plan is that after Spring Break, all lectures will be used for student demos.
- A demo in class will not count as a final submission.
- I will look at student demos in class and give comments and feedback.
- However, I will not check and debug student source code: *that is your responsibility.*
- **You can present demos more than once.**
  However, a student who is presenting a demo for the first time will receive priority over students who are presenting demos for the second, third, etc. time.
- The available time to view each student demo is limited, hence prepare your questions/presentation ahead of time and do not fumble and waste time during your demo.

3 Programming language/GUI

- My default visualization is Java and GUIs, *but this is not mandatory.*
- **Javascript, HTML, C#, etc. are all acceptable.**
- Your cover document must include instructions how to run your application.
- A GUI is not mandatory: you can create webpages and hyperlinks to those pages.
- **You must submit instructions (in your cover document) how to run your code.**
- It will help a lot if you give a demo in class and show me how to run your application.
- GUI and/or multiple popups:
  1. My default visualization is that I see only one GUI/web page at any one time.
  2. To navigate, there should be a 'back' button (or arrow) to go back to higher-level GUIs/pages.
  3. However, this is not mandatory.
  4. If your application supports multiple popups/web pages, then you must manage state information. For example, if I display a menu of a restaurant and then I go to the administrator page and edit the menu of that restaurant, what will happen?
     Will there be out of date data on the popup menu?
     How will you deal with this scenario?

4 Database/text files

- The use of a database is not mandatory.
- You may employ text files.
- **Whichever design you employ, it must support persistence across invocations.**
- If you use text files, they should be encrypted in some way (hash map/table?), so that I cannot edit the text file directly and change the data in the file.
- **The administrator interface must be the only way to add/delete/modify data.**

5 Address

- The address is an (i, j) integer pair, where $0 \leq i \leq 100$ and $0 \leq j \leq 100$.
- There is no need to invent street names, etc.
- A customer enters his/her location as an (i, j) integer pair in the relevant GUI/web page.
- It is your responsibility to write a sort function to resolve ties and display items in a consistent order.
- Your code should not throw an exception or crash, etc.
- *If you cannot figure this much out for yourself, do not complain if you fail the project.*

6 Restaurants

- The term 'restaurant' is used in a loose sense.
- I have explained that franchises, bars, pastry shops, etc. are all acceptable.
- **Your project should contain 10 restaurants (with the above broad definition).**
    1. **Create fake names for all restaurants, to avoid trademark violations.**
    2. Hence by definition they will be different from the ones you submitted in Assignment 7.
    3. **All names must be written using the English alphabet.**
    4. However the names can contain non-English words, e.g. 'Giovanni' or 'Haji Baba' etc.
- Franchise:
    1. There must be two franchises with the same name but different addresses.
    2. They must have the same menu in the database.
    3. If the menu is updated, the same updates must display for both franchises.
    4. I will add a third franchise location using the administrator page.
    5. You must supply instructions how to connect the third franchise to the menu of the others.
- Opening hours:
    1. There must be at least one 24/7 late-nite diner, open at all hours.
    2. There must be at least one restaurant which is closed on Monday (or another weekday).
    3. Each restaurant must display its opening hours (later hours on Friday & weekend?).
    4. Use a 12 hour clock (i.e. am/pm hours) for the opening hours, not a 24 hour clock.
- There must be at least one restaurant which has multiple menus.
    1. For example breakfast/lunch/dinner and/or kids menu.
    2. For example appetizer/entree/desserts/drinks menu.
    3. The menus could be in different GUIs/web pages, or one long GUI/webpage with a scroll bar.
- **There is no need to display 'special deals' for holidays, etc.**

7 Menus

- The layout of the menu on the screen is up to you.
- The layout could be either single-column or multi-column.
- The menu should be easy to read, but there is no need for fancy formatting.
- **All menu items must be written in English.**
- The choice of text and background colors is up to you, again do not get too fancy.
- It is acceptable if all the menus have the same layout template.
- **There must be a few menu items which display multiple prices.**
    1. For example small/large size for soup cup/bowl or coffee/drinks.
    2. For example 'combo/burger only' with different prices for the combo or only the burger.
- **The prices of all items should be listed to a maximum of 2 decimal places.**
- Optional:
    1. It is optional to include icons for healthy/spicy/lo-cal, etc.
    2. It is optional to include details in small print, e.g. 'includes choice of soup or salad' or 'choice of fried or steamed' etc.
- **Do not insert photographs/images of food and/or the restaurant rooms, etc.**
  It could cause copyright violations.

8 Customer

- A customer can select restaurants sorted by name or else by distance from the customer's location.
- It was explained above that you must design a suitable sort function to display items without throwing an exception, etc.
- The display of restaurants must show each restaurant's name and address and a link to navigate to that restaurant.
- When a customer clicks on a link for a restaurant, the application should navigate to the GUI/web page of that restaurant.
- The restaurant GUI/web page should display the opening hours and the menu, or a link to easily navigate to the menu.
- It was explained above that some restaurants must have multiple menus.
- It should be easy (for me) to navigate to each menu.
- **All GUIs/web pages accessed by a customer should be read-only.**
- **A customer does not have permission to change any data.**
- **A customer does not require a password to use your application.**

9 Administrator

- Only the administrator has permission to add/delete/modify data for the restaurants.
- The administrator GUI/web page must be password protected.
- This is admittedly rather useless, because we are the only people running the application.
- Nevertheless, you must demonstrate you know how to implement password protection.
    1. **Your cover document must include the administrator name and password.**
    2. The admininistrator name and password can be 'admin' and 'admin' (why not?).
- After logging in, there can be more than one admisintrator GUI/web page, if you find it more convenient to implement the administrator functionality in multiple GUIs/web pages.
- Administrator functionality:
    1. Add a new restaurant. The primary key is (name, address). An error message should be displayed if an attempt is made to save a primary key which already exists.
    2. Add/modify the opening hours of a restaurant. An error message should be displayed if an attempt is made to save a new restaurant without setting its opening hours. (However, the opening hours are not part of the primary key: many restaurants have the same opening hours.) The opening hours of a restaurant can be modified but cannot be deleted (a restaurant cannot have zero opening hours). The opening hours should use a 12 hour clock (i.e. am/pm hours, do not use a 24 hour clock).
    3. Add/delete/modify menu items for a restaurant. This includes creating a new menu, when creating a new restaurant. This also includes creating multiple menus for a new restaurant (breakfast/lunch/etc and/or appetizer/entree/drinks etc.). An error message should be displayed if an attempt is made to save a new restaurant without creating a menu for the restaurant (minimum one item in menu). An error message should be displayed if an attempt is made to delete all the items in a restaurant's menu. There must be a minimum of one item in the menu.
    4. The prices of all menu items can be set to a maximum of 2 decimal places.
    5. If a new franchise location is created, connect the menu to that of other franchise locations which already exist.
    6. Delete a restaurant which has closed/gone out of business.