USER SETUP/ SET UP STEPS	Notes	completion
First, examine the UML diagram to see which classes will be needed, along with what properties and methods each class will have.	Reviewed the UML, copied onto word file.	done
Create a new folder for your project and open the folder in VS Code.	C:\Users\mpacr\OneDrive\Desktop\devCodeCamp\GitHub Entries\restaurantEntrepreneur repository is active in github	done
Spend 5-10 minutes creating skeletons of all required classes and methods, remember to use the pass keyword to close off empty methods for now.	created first pass of classes and basic information according to UML	done
Start working on user stories starting from the top.	in progress	

TASKS/ USER STORIES	Notes	completion
(5 points): As a developer, I want to create my classes and methods according to the UML.	reviewed the UML, created shells according to the UML	done
(5 points): As a developer, I want to create an Order parent class and 3 child classes to represent menu items of my choosing	created using init and super init for order over pizza, pasta, and salad	done
(2.5 points): As a developer, I want to create an Order Factory class with a static create_order method.	static create_order method: not called on by an object: does not use self. So this would be pulled up through the front: a static class does not need an instatiater	done

(10 points): As a developer, I want to utilize a Factory Pattern in the create_order() method to instantiate instances of the three different Order child classes. This method should accept a string as a parameter (ex "Pizza") and return the corresponding type of Order child class instantiation (ex Pizza())	def create_order(self,type):
	if type == 'Pizza':
	return Pizza()
	elif type == 'Pasta':
	return Pasta()
	elif type == 'Salad':
	return Salad()
		done

(2.5 points): As a developer, I want to create a log.txt file to keep track of my business.	log.txt. This has been set up and is functional and basic- will clean up	done
(10 points): As a developer, I want to create a Logger class with a log_transaction() method that will accept an Order object and store number and: Increase the Logger's transaction_count by one Add the price of the Order object to the Logger's daily_sales Open the log.txt file Write a well-formatted message to the log.txt file containing the current transaction count, the name of the dish ordered, the store it was ordered from, the price of the item, and the combined daily income. Close the log.txt file.	watched the video on creating the logger, entered in a def write to file that will be subbed for log_transaction. This has been set up and will work on aesthetics	done
(5 points): As a developer, I want to use the Singleton pattern (as shown in the Design Patterns Demo repo) to create a single instance of a Logger object inside the logger.py file and import this instance into the Franchise class to be shared by all instantiations.	reviewed the chat history example. Will use this to build off of.	done
(10 points): As a developer, I want to create a Franchise class with a place_order() method that will: ask a user what food they would like to order call the static OrderFactory.create_order() method to instantiate an order object. call the logger.log_transaction() method to log the order to the log.txt file	input will come from the user --so user input function--to call the order call the order… going from Franchise to OrderFactory to input and get order information to pass on	done

(5 points): As a developer, I want to create a Simulation class with a run_simulation() method to act as a facade pattern. The run_simulation() method should: **Instantiate 3 separate Franchise objects.** **Call place_order() on each franchise object multiple times.**	**location_one = Franchise(1)** **location_two = Franchise(2)** **location_three = Franchise(3)** **location_three.place_order()** **location_one.place_order()** **location_two.place_order()** **location_one.place_order()** **location_three.place_order()** **location_two.place_order()**	**done**

CHECKLIST	**Notes**	**completion**
Run through the Setup Setups and get your project ready to begin work. **Review the Resources outlined below - be sure to have relevant documentation and references open while you develop!**	used all resources possible.. Room for improvement	done

END RESULT	**Notes**	**completion**
This project is meant to simulate tracking sales data from multiple restaurant locations. The end result will be a console application that writes information to a .txt file. When the user runs the console app, it will display several messages prompting the user to order an item from a restaurant location, one at a time, to simulate a day's business. After the orders are concluded, the results will be logged to the log.txt file, tracking the total number and dollar value of the combined sales. **Note that the log.txt can be cleared out routinely during the testing/development process, since it will retain information from all previous times the program ran.**	Meets Requirements… left off where I wanted to make sure there was a loop of sorts to avoid the error out.	