Hacettepe University
Computer Science and Engineering Department

Name and Surname	**:** Murat Celik
Identity Number	**:** 21827263
Course	**:** BBM104
Experiment	**:** Quiz 2
Subject	**:** Inheritence and Polimorphism
Data Due	**:** 30.04.2020 (23:59)

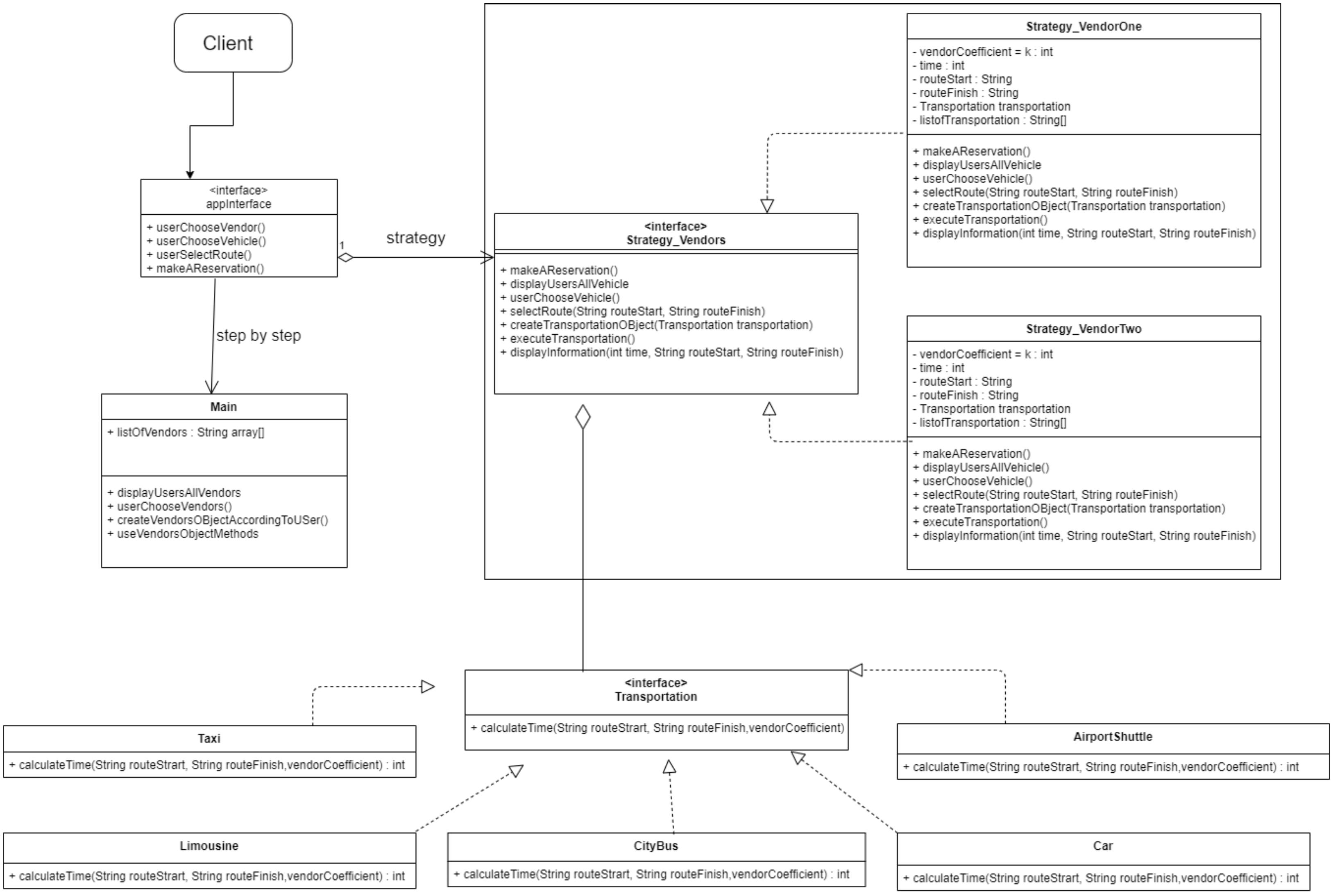
Problem

Developing an online reservation system that can provide airport transportation service to the user

Additional information: Necessary information is given in 104_Quiz3.pdf file.

What I Learnt?

Firstly I have learnt *Open/Closed Principle.* Sometimes we want to improve our code. But we can't do this improvement every time by changing the code. We encounter problems and it is not practical. If we design our code with a certain standard, we can make additions without changing our code. These standards are called *Open Closed Principle*. Secondly I have learnt *The Strategy Pattern*. Sometimes there may be more than one strategy to perform a process, and we customize this strategy according to the user. This is a pattern which called *The Strategy Pattern*.


Client
<interface>
appInterface
+ userChooseVendor()
+ userChooseVehicle()
+ userSelectRoute()
+ makeAReservation()
1
strategy
step by step
Main
+ listOfVendors : String array[]
+ displayUsersAllVendors
+ userChooseVendors()
+ createVendorsOBjectAccordingToUSer()
+ useVendorsObjectMethods
<interface>
Strategy_Vendors
+ makeAReservation()
+ displayUsersAllVehicle
+ userChooseVehicle()
+ selectRoute(String routeStart, String routeFinish)
+ createTransportationOBject(Transportation transportation)
+ executeTransportation()
+ displayInformation(int time, String routeStart, String routeFinish)
Strategy_VendorOne
- vendorCoefficient = k : int
- time : int
- routeStart : String
- routeFinish : String
- Transportation transportation
- listofTransportation : String[]
+ makeAReservation()
+ displayUsersAllVehicle
+ userChooseVehicle()
+ selectRoute(String routeStart, String routeFinish)
+ createTransportationOBject(Transportation transportation)
+ executeTransportation()
+ displayInformation(int time, String routeStart, String routeFinish)
Strategy_VendorTwo
- vendorCoefficient = k : int
- time : int
- routeStart : String
- routeFinish : String
- Transportation transportation
- listofTransportation : String[]
+ makeAReservation()
+ displayUsersAllVehicle()
+ userChooseVehicle()
+ selectRoute(String routeStart, String routeFinish)
+ createTransportationOBject(Transportation transportation)
+ executeTransportation()
+ displayInformation(int time, String routeStart, String routeFinish)
<interface>
Transportation
+ calculateTime(String routeStrart, String routeFinish,vendorCoefficient)
Taxi
+ calculateTime(String routeStrart, String routeFinish,vendorCoefficient) : int
AirportShuttle
+ calculateTime(String routeStrart, String routeFinish,vendorCoefficient) : int
Limousine
+ calculateTime(String routeStrart, String routeFinish,vendorCoefficient) : int
CityBus
+ calculateTime(String routeStrart, String routeFinish,vendorCoefficient) : int
Car
+ calculateTime(String routeStrart, String routeFinish,vendorCoefficient) : int

User Algorithm

1. Open the app (obilet, skyscanner)
2. See all of vendors.
3. Choose one of them.
4. See all of vehicles
5. Choose one of them.
6. Select route where from route start point to route finish point
7. make a reservation
8. Show users reservation details.
9. close the app

App Algorithm

1. Show Users all Vendors
2. Get the choice from the user.
3. Declare strategy according to user choice.
4. Create a object from Vendors according to user choice.
5. Show users all Vehicle which **vendors have**.
6. Get the choice from the user.
7. Create a object from Transportation according to user vehicle choice.
8. Get the route where from route start point to route finish point from user.
9. Call the method for calculate time in vehicle class which users choose
10. Show reservation details.
11. If users want close it go to 12 or if users continue it go to 5.
12. close the app

Solution

We have an interface called a Strategy_Vendors. We can create a Vendor from here. After the user enters the application, he selects the vendors. This means that we have 2 strategies now . If we want more, we do this by creating a vendor class. The user selects the vendor and tells us which strategy to use, and accordingly we start the strategy. After this step, we tell the user vehicles he can choose. This is different for every vendors. The user selects one of the vehicles at the selected vendors class/strategy. We create an object for the selected transportation and use the method. After the user says the start and end points, we calculate the time. After all, reservations are made and information is displayed on the screen.