Software Engineering

Books or notes are **not** allowed.
Write only on these sheets. **Concise** and **readable** answers please.

Surname, name, matricola ____

Gasoline prices
Each gas station in Italy is free to define the price of fuels (regular gasoline, premium gasoline, diesel). Drivers are interested in finding as easily as possible the closest gas station with the cheapest price.

In the following you should analyze and model an application that supports drivers in finding the best price of fuels.
The application should be available both as web site via PC, and as app on smartphones. A driver first has to sign in to access the service. The application is based on a graphical interface that shows on a map gas stations, and the prices they offer.
The smartphone app shows the gas stations in the area of the driver (of course this needs the activation of localization service). The web site shows the gas stations in an area selected by the user.
The user can customize what is shown (for instance show all gas stations and all prices, or only gas stations that have prices below average, show all prices or only gasoline / only diesel).
The user can then select a gas station and ask directions to it (navigation function). The navigation function is external (for instance uses a Google navigation service).
The prices are collected by the drivers themselves. So a function must be available for users to communicate to the application the prices of a certain gas station. Therefore a price has a time tag, and an obsolescence property too. The application has to manage somehow the reliability of prices, in terms both of obsolescence, and trust of users that communicate the prices. Trust of users is evaluated by reports or votes of other users (for instance if a user reports that the price of gas station X is wrong, and user U has communicated that price recently, then the trust of U lowers. Viceversa if the price is correct).

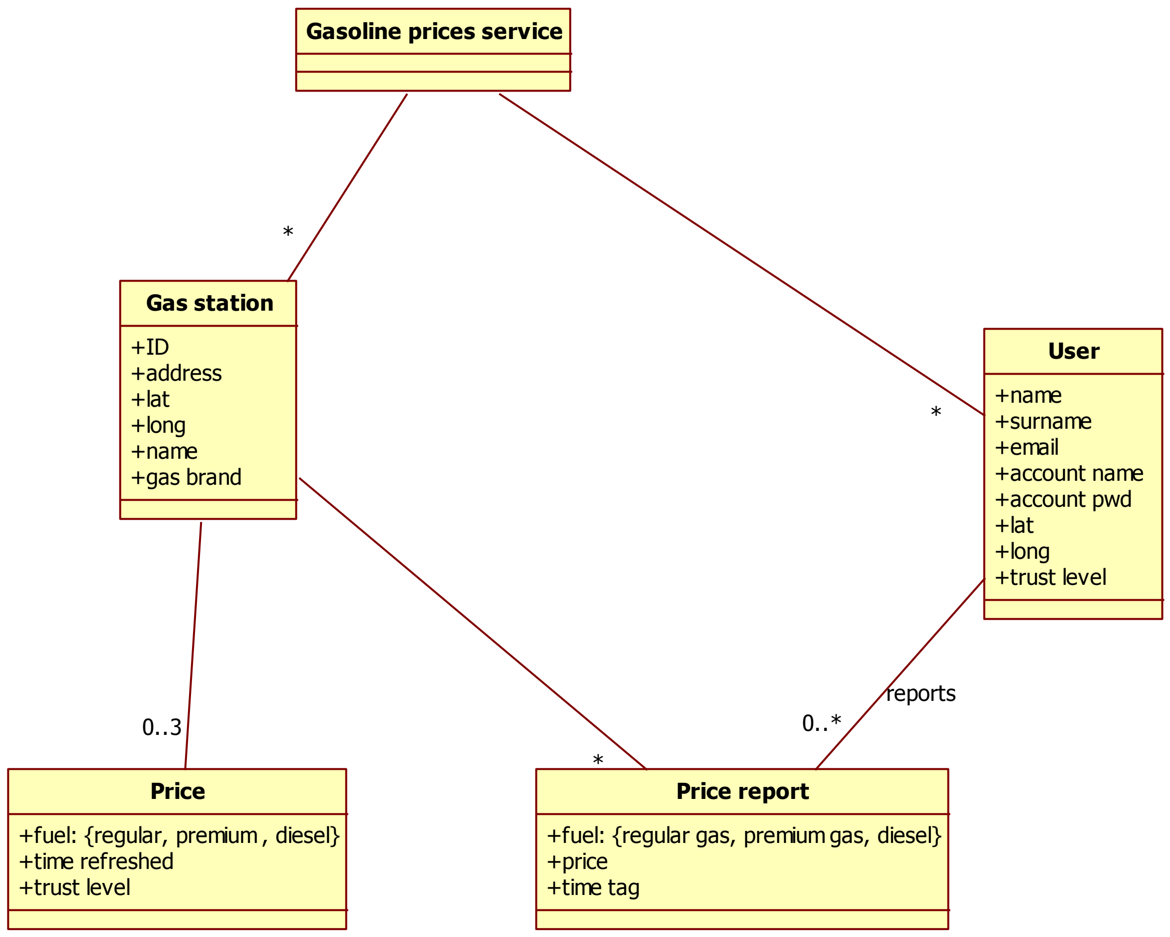
1 (15 points) – a. Define the context diagram (including relevant interfaces)

Actor	Physical interface	Logical interface
User (driver)	PC or smartphone	GUI
Admin	PC	GUI
Localization service	GPS in smartphone	Localization API / OS call
Navigation service	Internet connection	Web service API

Define the glossary (key concepts and their relationships) (UML class diagram) for the application

Many users could report prices over the same gas station in the same time frame (ex early morning). Besides, trust of users depends on the prices they have reported (how many times, how precisely). For these reasons we need two classes, ‘price report’ and ‘price’.

The class ‘price report’ represents a specific report about a price by a user (it is an event, with a time tag). The class ‘price’ instead is the price of a fuel that will be shown by the app for a certain gas station. ‘price’ has to be calculated starting from relevant ‘price report’s, with a certain frequency (ex early morning, when stations open and may change the price, and then each time there is a new price report). Many algorithms can be used to compute ‘price’ (the simplest being ‘price’ = last ‘price report’; or ‘price’ = most frequent ‘price report’ in the last hour; and so on). ‘price.trust level’ needs also to be computed, based on when the price was refreshed (obsolescence) and on trust of users who reported that price. ‘user.trust level’ is computed in function of how many prices the user reports, that are confirmed (or not considered wrong) by other users.



List the requirements in tabular form (do not forget to list important NF requirements)

ID	Type (Functional Non Functional)	Description
1	F	User management: CRUD User, authorize and authenticate
2	F	User trust: compute trust of user using the trust of his/her price reports
3	F	Gas station management: CRUD Gas station
4	F	Price report management. CRUD Price report, attach price report to user, to gas station;
5	F	Price management. CRUD price management, compute price starting from (recent) price reports of a station, compute level of trust of the price report Signal price is correct / incorrect
6	F	Navigation. Compute path from position of user to defined gas station
7	F	Show gas stations and their fuel prices on a map, customizable by area, fuel type
8	NF	Privacy. Data of user should be visible only to user, not to other users. Identity of user who defined a price report should not be visible to other users.
9	NF	Efficiency. All functions should complete in <0,5 sec
10	NF	Portability. Application should be available on PC and smartphones (IoS, Android, ..)

List the user level goal Use cases

(user) Register to service and obtain an account
(user) Report price of fuel for a gas station
(user) Obtain price of fuel for gas stations in a certain geographic area
(admin) Create / modify information about a gas station
(admin) Modify / delete price reports (admin here plays the role of a moderator in case of frauds)
(admin) Create / modify/ delete account of user

Select one use case from the point above and describe it

Name	Obtain price of fuels
Scope	Gasoline prices app
Level	User goal
Intention	Driver wants to find the cheapest/closest gas station to fill tank
Primary actor	Driver
Precondition	Driver is registered and has valid account Driver has defined preferences (fuel = diesel, criterion = lowest price within 5km radius) Driver accesses via smartphone
Main success scenario	1 login 2 system retrieves location of driver, retrieves map, retrieves gas stations and prices in 5km radius, computes average price on those gas stations, selects gas stations with price <= average 3 system shows gas stations on map 4 logout
Extensions	1b driver changes preferences 3b driver selects gas station on map and asks for navigation to it 3c no gas stations found in selected area

2 (7 points) -Define black box tests for the following function, using equivalence classes and boundary conditions.

```
int times[]  computeWaterTiming(int start, int duration, int nTimes)
```

This function programs a timer to water gardens. The timer has an internal clock and opens/closes a valve. The timer considers only a one day range. The first parameter defines the time (in minutes) when the valve opens, the second parameter the duration (in minutes) the valve remains open, and the third parameter the number of times the cycle is repeated. nTimes can be only 1, 2, or 3. If the value is 1 the cycle is repeated every 24 hours, if 2 every 12 hours, if 3 every 8 hours. The function returns an array with the specific start / end times, in minutes, when the valve remains open. Returns an error in case the values passed are not meaningful.

```
Ex.     computeWaterTiming(60, 20, 1) → [60, 80, -1 , -1 , -1,-1]
        // open the valve 1 time per day, starting at 1am, during 20 minutes
        // first open period from minute 60 to 80

        computeWaterTiming(60, 20, 2) → [60, 80, 780 , 800 , -1,-1]
        // open the valve 2 times per day, starting at 1am, during 20 minutes
        // first open period from minute 60 to 80, second period from minute  780 to 800

        computeWaterTiming(60, 20, 3) → [60, 80, 540 , 560 , 1020, 1040]
        // open the valve 3 times per day, starting at 1am, during 20 minutes
 // first open period from minute 60 to 80, second period from 780 to 800, third 1020 to 1040
```

Minutes in one day= 60x24= 1440

start	duration	Ntimes	Start + duration*nTimes <= 1440	
[minint, 0[	-	-	-	T(-10,1,1 ; err)
[0, 1440]	[minint, 0[	-	-	T(10,-101,1 ; err)
	[0,1440]	[minint, 0[	-	T(10,1,-10 ; err)

		1	T	T(60, 20, 1 ; 60, 80,-1-1-1-1)
				Tb(1, 20, 1 ; 1, 21,-1-1-1-1)

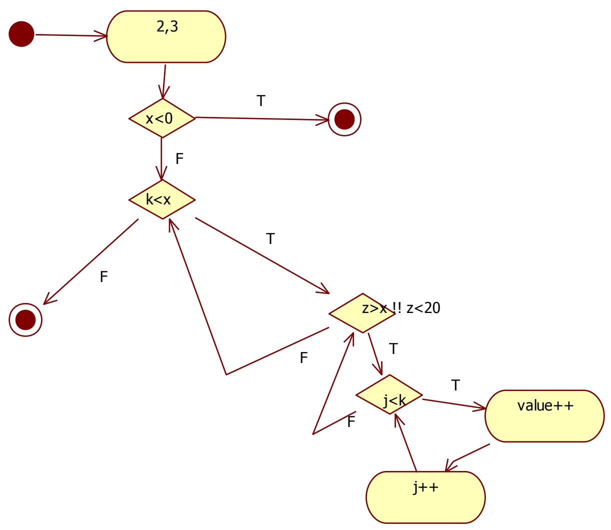
			F	T(60,1400,1 ; err)
				Tb(1,1440,1; err)

		2	T	T(60, 20, 2 ; 60, 80,780, 800, -1-1)
			F	T(60,700,2 ; err)
		3	T	T(60, 20, 3 ; 60, 80,540, 560, 1020,1040)
			F	T(60,800,3 ; err)
		[4, maxint]	-	T(10,1,10 ; err)
	[1441, maxint]	-	-	T(10,1500,1 ; err)
[1441, maxint]	-	-	-	T(1500,1,1 ; err)

3 (7 points) – For the following function define the control flow graph, and define test cases to obtain the highest possible node coverage, edge coverage, multiple condition coverage, loop coverage, path coverage.
For the test cases, **write only the input value**.

WRITE control flow graph here

```
1 int function(int x, int z){
2 int k; int j;
3 int value =0;
4
5 if (x < 0) return -1;
6 for (k=0; k<x; k++){
7    if ( z> x ||  z<20)
8       {
9          for (j =0; j<k; j++ ){
10
11                value ++;
12          }
13       }
14 }
15 return value;
16 }
```



Coverage type	Number of test cases needed to obtain 100% coverage	Coverage obtained with test cases defined (%)	Test cases defined
Node	2	100	T1, T2
Edge	3	100	T1, {T2 or T4 or T5}, T6
Multiple condition line 7	4	100	T2, T3 T T T4 F T T5 T F T6 F F
Loop line 9	3	100	T3, no enter T2, one time T4, many times
Path	O (x^2)	Not feasible	

Write test case ID (t1, T2 ..) in the rightmost column, and test cases here
T1(-1, 10)
T2(2, 3)
T3(1,3)
T4(3,3)
T5(2,21)
T6(22,21)

4 (1 points) – 140 person hours are equivalent to how many person months? Explain your answer

The conversion rate depends on national and company rules. If one person month == 140 person hours then result is of course 1. If one person month == 160 person hours then result is 140/160. And so on.

5 (1 point) – Describe the key concepts of 'mutation testing'.

A technique to evaluate how good a test suite is. Errors are inserted in the program under test (mutations). The more mutations the test suite finds, the better the test suite.

6 (1 point) – Describe the key steps in an inspection process of a requirement document.

See slides

7 (1 point) – Describe the key concepts of the Scrum process

See slides

8 (1 point) – Describe shortly the 'Facade' Design pattern

See slides