

**Subject:** Object Oriented Programming

**NRC:** 2858

**Members:**

Cortez Tamayo Jonathan Andrés

Coyago Arce Eunice Pamela

Cují Revelo Milton Roberto

Garcés Mosquera Christian Marce

Guerra López Luciana Noemi

**Subject:** Mechanic Shop

Contenido

[General objective 3](#_Toc64392529)

[Specific Objectives 3](#_Toc64392530)

[Theory 3](#_Toc64392531)

[Glossary 3](#_Toc64392532)

[Definition 3](#_Toc64392533)

[Problem 4](#_Toc64392534)

[Solution 4](#_Toc64392535)

[Scope 4](#_Toc64392536)

[UML Diagram 4](#_Toc64392537)

[UML Diagram Description 5](#_Toc64392538)

# General objective

To develop a management system for a mechanic workshop, so that the program can expand productivity, reduce waiting time for customers, obtain an inventory in order and optimize the results of the enterprise.

# Specific Objectives

* To optimize the administrative services in the enterprise.
* Implement a payment service.
* Register customers in a database for future revisions in their vehicles.
* To organize an inventory by product category.

# Theory

## Glossary

* **Object Oriented Programming:** Manipulates input and output data through specific objects.
* **Library:** Implementation and coding of a programming language.
* **Class:** Set of variables.
* **Variable:** Values or operations to take or process.
* **Programmable Function:** Main algorithm that allows to solve a specific task.
* **Algorithm:** Set of instructions.
* **Object:** Information elements characterized by real data.
* **Method:** Actions that the object can do by itself.
* **Attribute:** Variables related to the state of an object.
* **Event or Behavior:** Realizable actions.
* **Messages:** Data transmission.
* **Instance:** Call of an object.
* **String:** Data type that interprets a text string.

## Definition

The mechanic service workshop "Martinez" is located in the province of pichincha canton quito, Chillogallo neighborhood el Girón. This workshop began in 2011, a small shop that had a master mechanic, over time it grew to have workers and offer a variety of services, becoming a profitable business, acquire several tools due to technological advancement.

## Problem

Based on the growth of mechanics, the need arose to implement a system that allows effectively manage the different processes, such as customer and product registration processes, while disorganization generated a significant loss of time, both for the customer and the worker.

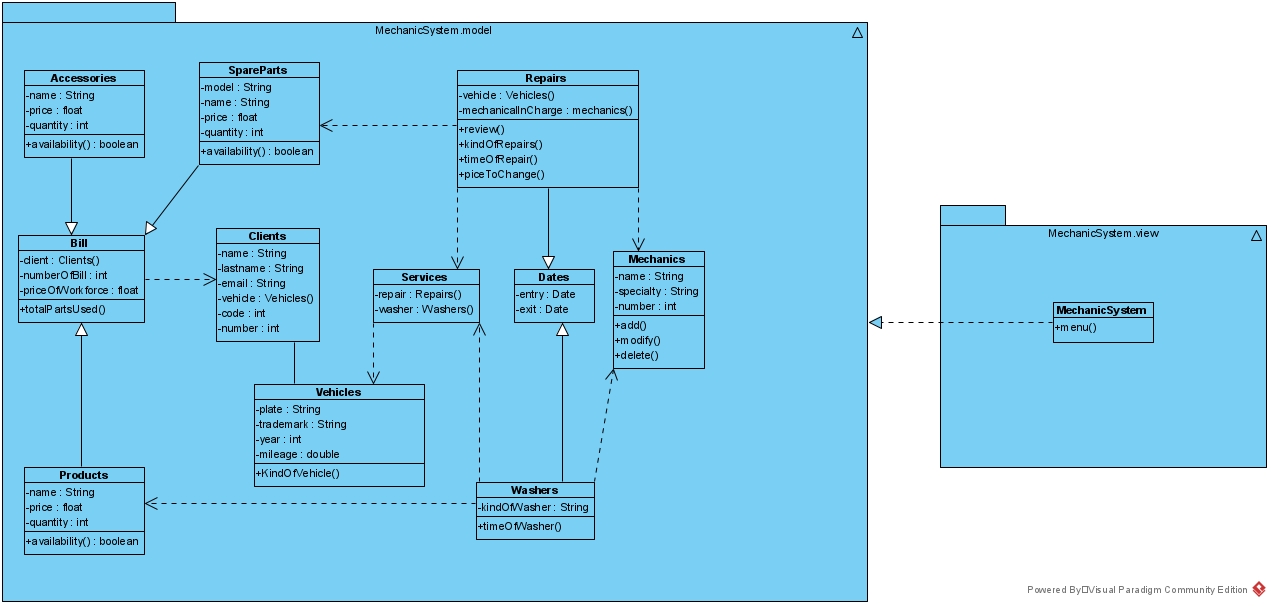
## Solution

Establish a system based on object-oriented programming, which guarantees the optimization of time when providing a service, structure an inventory capable of describing each product and the exact location in the warehouse, develop a payment system, which stores the data of recurring customers securely. Define services and budgets based on what the repair shop offers.

## Scope

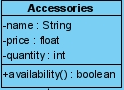
The system will allow the entry of customer and employee data, which will be stored in a database. It will define the services provided by the workshop with their costs and the products used, they will have a description and location of each one, stored in categories by products.

# UML Diagram



# UML Diagram Description

**Accesories**



**Attributes**

**name:** variable used to store the name of an accessory of type string.

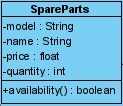
**price:** variable used to store the price of an accessory of type float.

**quantity:** variable to store the number of existing accessories of type int.

**Methods**

**availability():** boolean method determines if an accessory is in stock.

**SpareParts**



**Attributes**

**model:** variable describing the model of a spare part of type string.

**name:** variable used to store the name of a string type spare part.

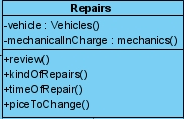
**price:** variable to store the price of a spare part of type float.

**quantity:** variable to store the number of existing spare parts of type int.

**Methods**

**availability():** boolean method determines whether a spare part is in stock.

**Repairs**



**Attributes**

**vehicle:** variable allowing to use the attributes of the vehicle class.

**mechanicalInCharge:** variable allowing to use the attributes of the mechanic class.

**Methods**

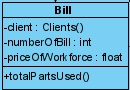
**review():** void method describing an overhaul of a vehicle.

**kindOfRepairs():** void method describing what is performed in a vehicle checkup.

**timeOfRepairs():** void method describing the time taken for a vehicle revision.

**piceToChange():** void method describing the change price.

**Bill**



**Attributes**

**client:** variable allowing to use the attributes of the customer class.

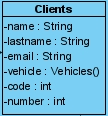
**numberOfBill:** variable that stores the number of delivered invoice of type int.

**priceOfWorkforce:** variable that stores the price of workforce of type float.

**Methods**

**totalPartsUsed():** void method describing the total parts used.

**Clients**



**Attributes**

**name:** variable used to store the name of a string client.

**lastName:** variable that allows saving the last name of a customer of type string.

**email:** variable that saves the customer's email of type string.

**vehicle:** variable that uses the data of the vehicles class.

**code:** variable that allows to store a code for each customer of type integer.

**number:** variable that allows to store the phone number of each customer that enters, of type integer.

**Services**



**Attributes**

**repair:** variable that uses the attributes of the repairs class.

**washer:** variable that uses the attributes of the washers class.

**Dates**

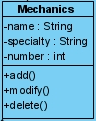


**Attributes**

**entry:** variable that stores the date of entry to the mechanics.

**exit:** variable that saves the date of exit from the mechanics.

**Mechanics**



**Attributes**

**name:** variable used to store the name of a mechanic of type string.

**specialty:** variable that stores the specialty of each mechanic of type string.

**number:** variable that stores the number of each mechanic of type int.

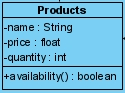
**Methods**

**add():** Void method in which it allows the entry of a plate or data of the person.

**modify():** Void method in which it allows to modify a plate or person's data.

**deleted():** Void method in which it allows the removal of a plate or data of the person.

**Products**



**Attributes**

**name:** variable used to store the name of a product of type string.

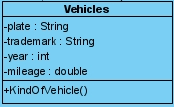
**price:** variable used to store the price of a product of type float.

**quantity:** variable to store the number of existing products of type int.

**Methods**

**availability():** boolean method determines whether a product is in stock.

**Vehicles**



**Attributes**

**plate:** variable that allows to enter the series of a type string.

**trademark:** variable that stores the trademark of a vehicle of type string.

**year:** variable that stores the year of creation of a vehicle of type int.

**mileage:** variable that stores the mileage of a vehicle of type double.

**Methods**

**kindOfVehicle():** void method describing the type of the vehicle.

**Washers**



**Attributes**

**kindOfWasher:** variable that describes the type of vehicle wash.

**Methods**

**timeOfWasher():** void method describing the time it takes to wash a vehicle.

**DEPENDENCE**



Dependency which momentarily uses the attributes of the other class.



Generalization to determine that it requires elements of the other class.