Cairo University

Faculty of Computers and Artificial Intelligence

El-Abd on The Go Vending Machine

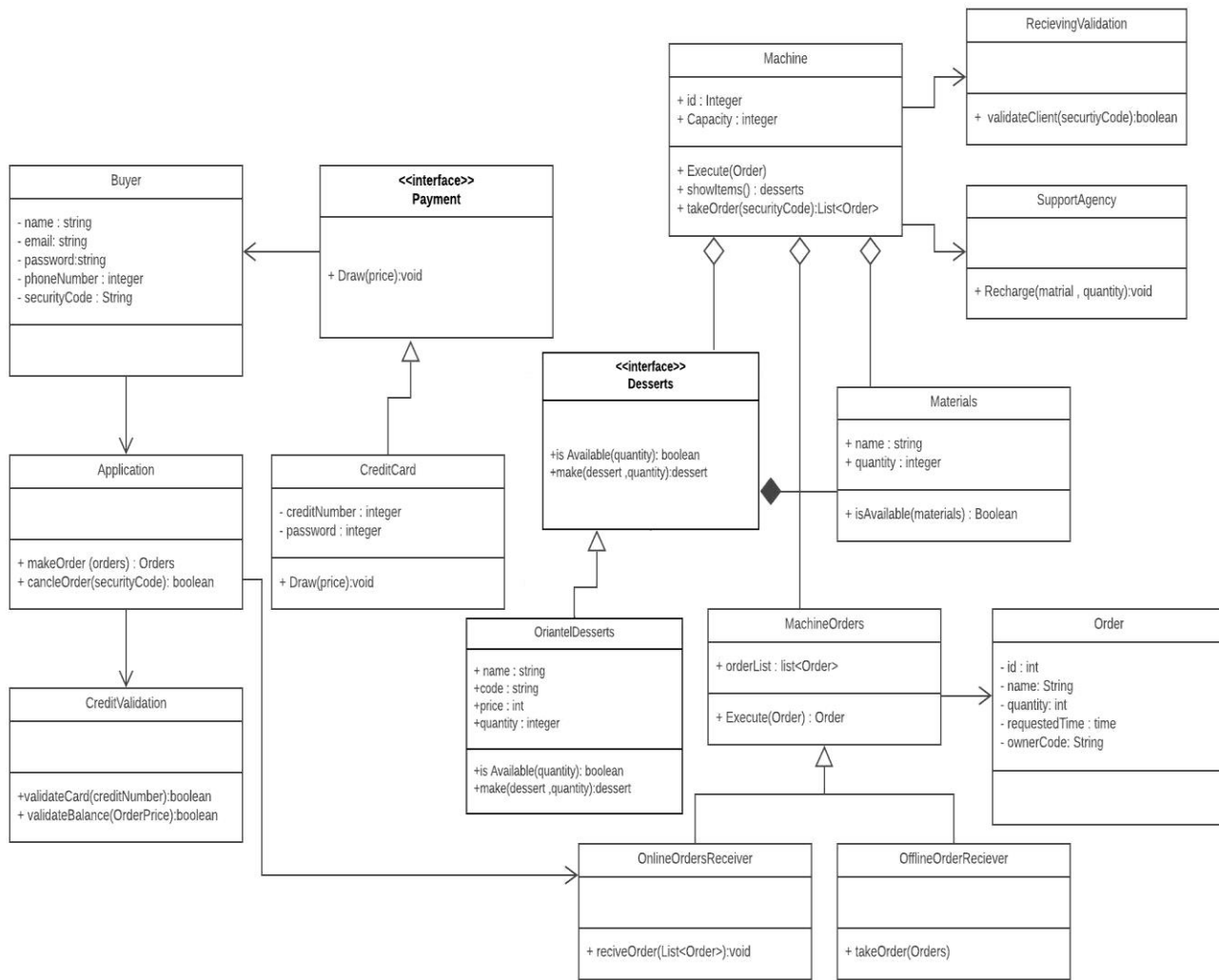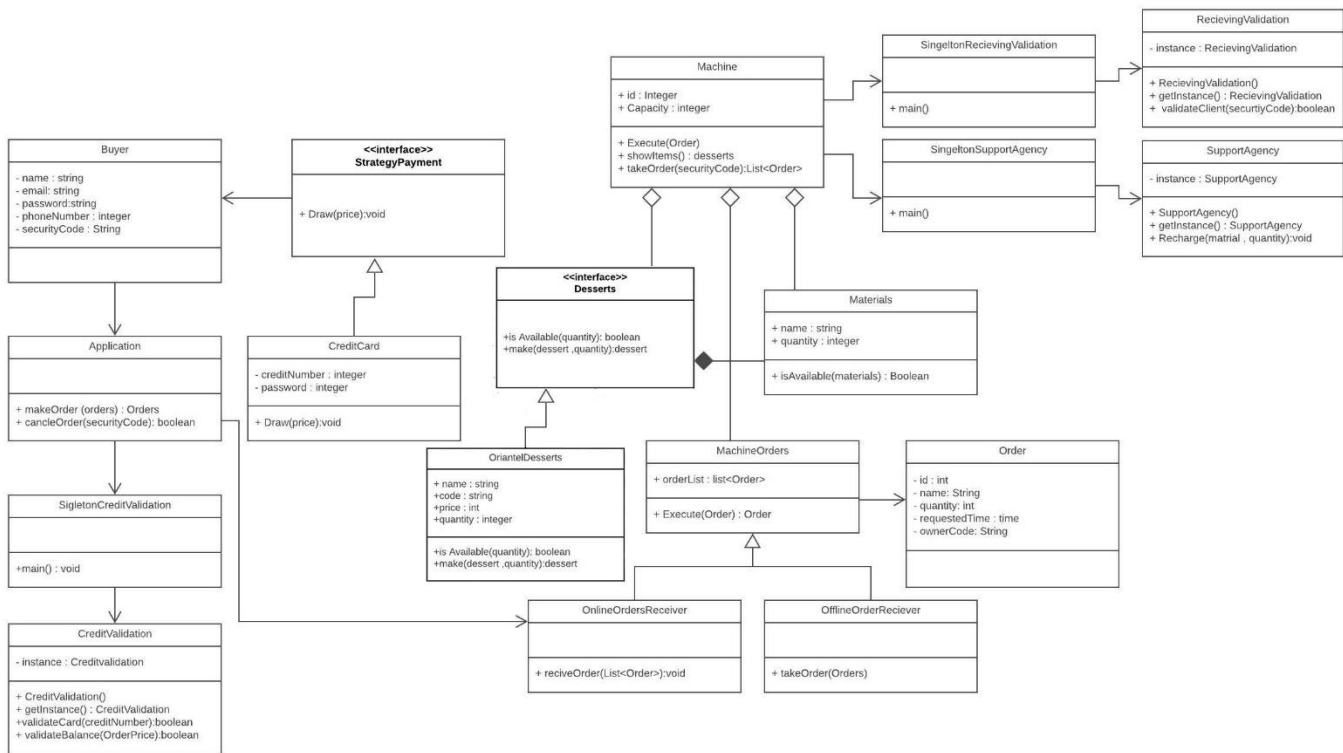| Name	ID
Atef Magdy Mitwally	**20170136**
Abdulrahman Bahig Mahmoud	**20170143**
Ibrahim Ramadan Abdu	**20170002**
Ahmed Sayed Mansour	**20170022**
Hatem Sayed Ali	**20170084**

Under supervisor
Dr. Soha Makady

Requirements:

Functional requirements:

1- The machine will be connected to a server, via the internet. This server must be connected to an application named "El-Abd on The Go" To enable the user to make order online.

2- Clients can create accounts on the "El-Abd On the Go" application by entering their email, name, password and their phone number. The application will validate the input email format and password length. The server will check this email doesn't exist before.

3- Clients can login into their accounts by entering their email and password the and browse their profile. The server validates the login accounts and passwords.

4- "El-Abd on The Go" enables the Client to browse all available items in the machine in a menu list. The menu displays the items' names, photos, prices and estimated quantities available.

5- Each client will have a cart in which he can add his orders from menu list, after picking order he will be asked for the quantity of the item (at least 1). He can then submit orders when he is done choosing.

6- After submitting, the total price will be displayed to the client and he will be asked to choose the pickup time, in which he can come pick his order.

7- Clients will be asked for credit card number and password each time they finish ordering.

8- The application will validate the credit card number, if it is right it will proceed with the server to communicate with credit card companies to draw the required price.

9- A count down for the order pick up time will be displayed to the client in his profile, and a message will be sent to the email registered denoting a successful order, with the pickup time, price charged, order id and order details.

10- Machine will prepare the orders before the time which is selected by the clients. The client will be notified by a reminder several minutes before the order is done.

11- Filling machine is done by conducting a request for the support agencies asking for the supply, If the machine has one of its gradients going short below 15%, and if the gradient is frequently falling, it will ask for larger quantities.

Class Diagram:

Machine

+ id : Integer
+ Capacity : integer

+ Execute(Order)
+ showItems() : desserts
+ takeOrder(securityCode):List<Order>

RecievingValidation

+ validateClient(securtiyCode):boolean

SupportAgency

+ Recharge(matrial , quantity):void

Buyer

- name : string
- email: string
- password:string
- phoneNumber : integer
- securityCode : String

<<interface>>
Payment

+ Draw(price):void

<<interface>>
Desserts

+is Available(quantity): boolean
+make(dessert ,quantity):dessert

Materials

+ name : string
+ quantity : integer

+ isAvailable(materials) : Boolean

Application

+ makeOrder (orders) : Orders
+ cancleOrder(securityCode): boolean

CreditCard

- creditNumber : integer
- password : integer

+ Draw(price):void

CreditValidation

+validateCard(creditNumber):boolean
+ validateBalance(OrderPrice):boolean

OriantelDesserts

+ name : string
+code : string
+price : int
+quantity : integer

+is Available(quantity): boolean
+make(dessert ,quantity):dessert

MachineOrders

+ orderList : list<Order>

+ Execute(Order) : Order

Order

- id : int
- name: String
- quantity: int
- requestedTime : time
- ownerCode: String

OnlineOrdersReceiver

+ reciveOrder(List<Order>):void

OfflineOrderReciever

+ takeOrder(Orders)

The Updated class diagrams


Singleton Design Patterns:

The purpose of choosing the Singleton is to control object creation, it is responsible to create an object while making sure that only a single object gets created. due to there is only one Singleton instance, any given time, a Singleton will occur only once per class. single class provides a way to access its only object which can be accessed directly without the need to instantiate the object of the class.it allowing the flexibility to create more objects if the situation changes. **We use this pattern in (Receiving Validation, Support agency, Credit Validation) Class.**
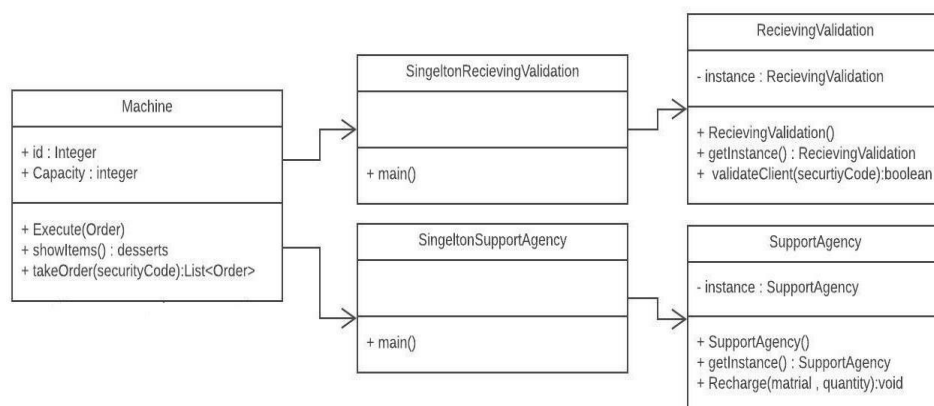
Facade Design Pattern:

Facade pattern hides the complexities of the system and provides an interface to the client using which the client can access the system. This type of design pattern comes under structural pattern as this pattern adds an interface to existing system to hide its complexities.

This pattern involves a single class which provides simplified method (Execute ()) which calls to methods of existing system classes. **We use this pattern in (Machine) Class.**
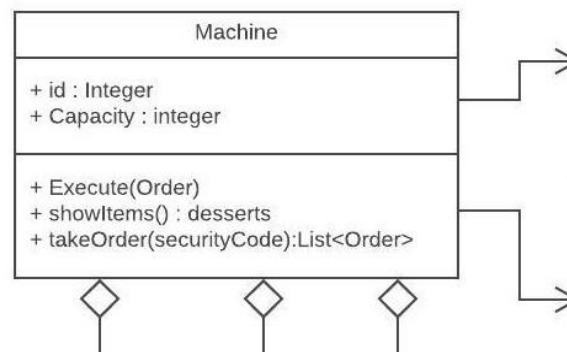
Strategy Design Pattern:

The Strategy Pattern is used to create an interchangeable family of algorithms from which the required process is chosen at run-time". it making the system more flexible and easier to expand.
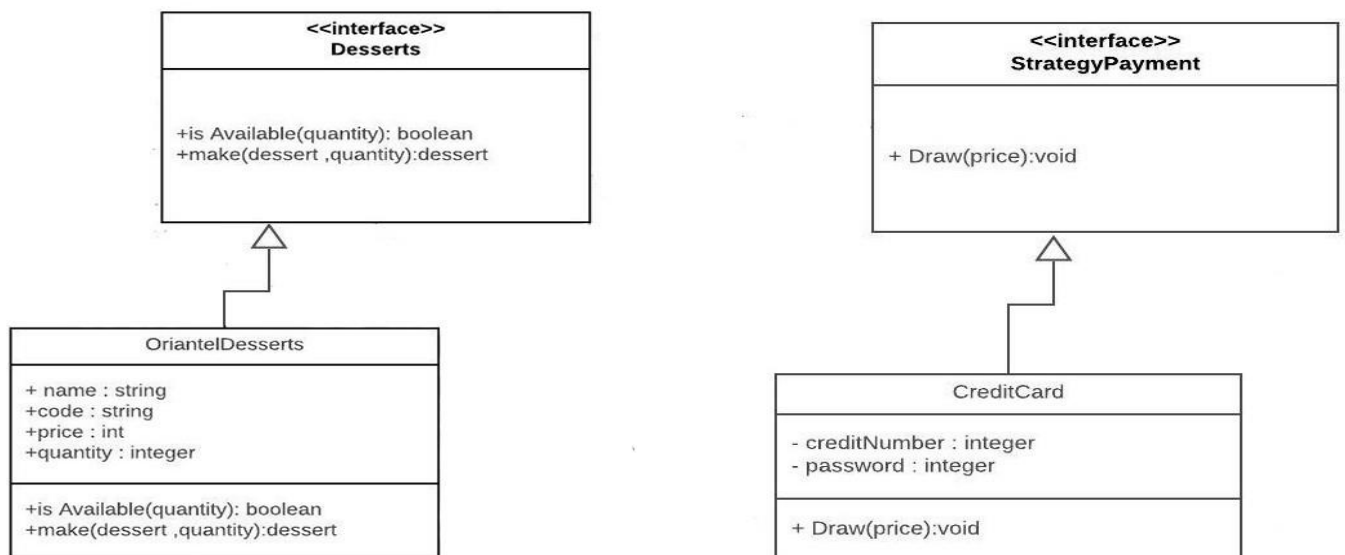
Subsystem Decomposition:

we divide the system in way that every layer or interface use one of subsystems in the below layers or in the same layer.

So, we put the buyer in the top layer because he is user of everything below, and put the application in the next layer with the payment, because the application uses the credit card in its process.

(Top 2 layers are the buyer side)

(Remaining layers are the machine side)

So, we put the machine in the top of these layers with the validation and Support agency interfaces, because they more related with the machine so it's good to put them in the same layer rather than put them in another layer.

and we put in the last layer the orders and items interfaces because application interface uses orders and machine uses Orders and Items.
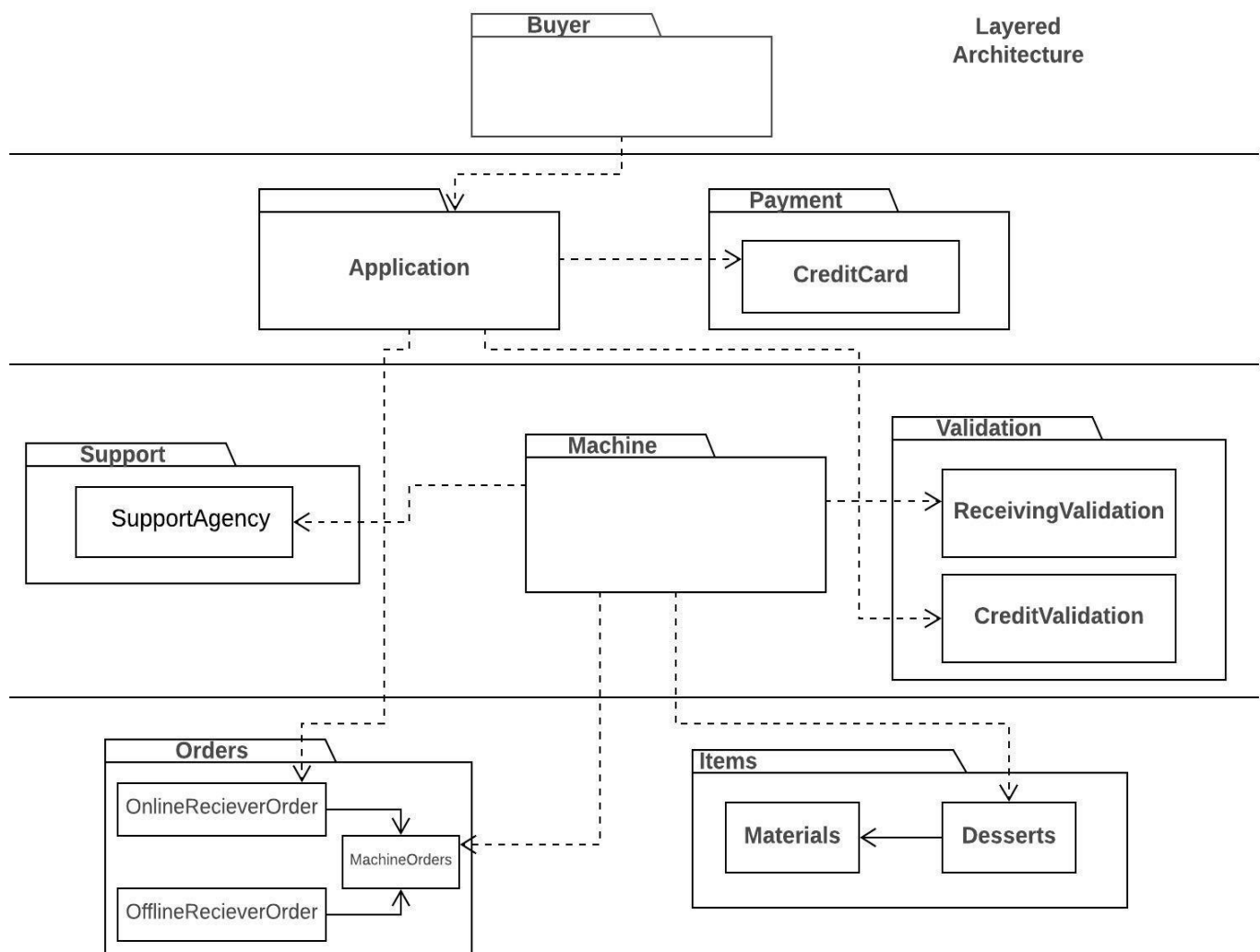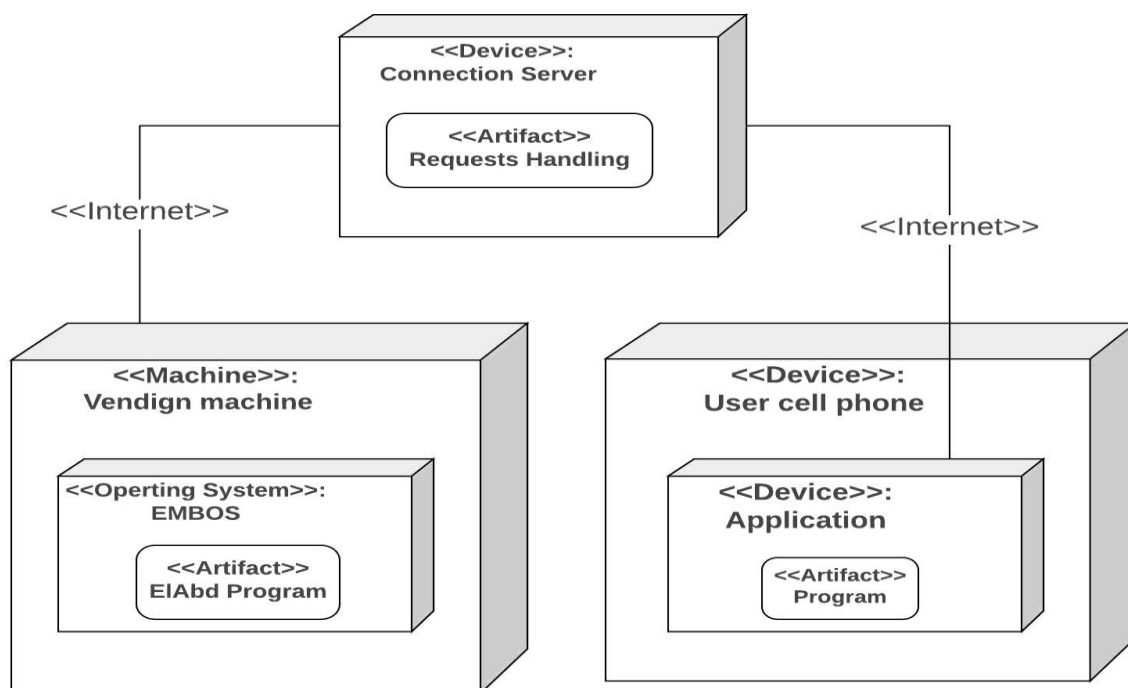
The cohesion in the subsystem decomposition is high, the classes in the subsystem perform similar tasks and are related to each other via associations.

And the coupling is low, a change in one subsystem does not affect any other subsystem.

Component Diagram:
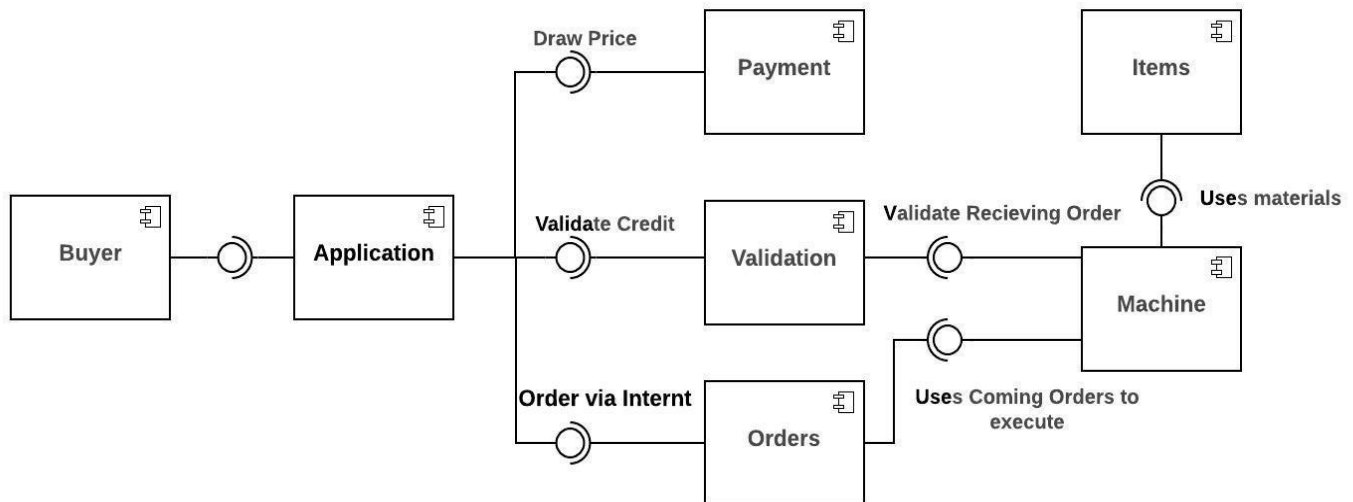
Deployment Diagram:

Design Goals:

Extensibility:

The ability to add new functionality with ease. We can easily add new types of desserts, new payment methods.

Flexibility:

Easily adaptable to reasonable changes. System may be implemented as website in the future. User can order from website instead of mobile application

Usability:

Easy to use the system. User can easily make an order via his cell phone

Response time:

System response to the user request as soon as possible.

Extensibility and Flexibility is maps to Supportability

Response time maps to Performance