**17/07/2024**

**Our goal: Create my own website and host it.**

**Set up the github properly.**

Things required to build an app:

IAAS-

* applications
* data
* runtime
* middleware
* o/s
* virtualization
* servers
* storage
* networking

(https://www.bing.com/images/search?view=detailV2&ccid=AyFOw%2FFp&id=AF1F625D0EAE0B0C35521EEA6BBD618E882A3A8C&thid=OIP.AyFOw\_Fpq-WNOIUWLyYy4AHaEu&mediaurl=https%3A%2F%2Fwww.redhat.com%2Fcms%2Fmanaged-files%2Fiaas\_focus-paas-saas-diagram-1200x1046.png&exph=1046&expw=1638&q=cloud+iaas+paas+saas+examples&simid=608007013016615119&form=IRPRST&ck=8832A58513E8E30081D564DE1B1CF771&selectedindex=1&itb=1&ajaxhist=0&ajaxserp=0&vt=0&sim=11)

How software is converted into an image?

wsl-(ubuntu,linux)

class Phone{

private String company;

private String modelNo;

private int phoneNo;

}

class Perfume{

private String name;

private int quantity;

private int price;

}

class Diamond{

private int grams;

int price;

}

keywords:

throws

transient

default-switch case, default methods of a functional interface

public class

synchronized-Mutex

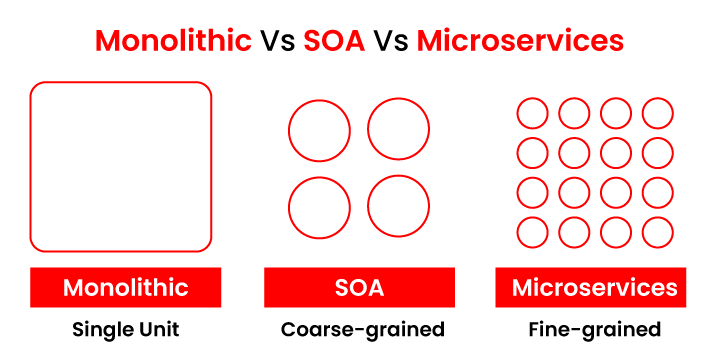
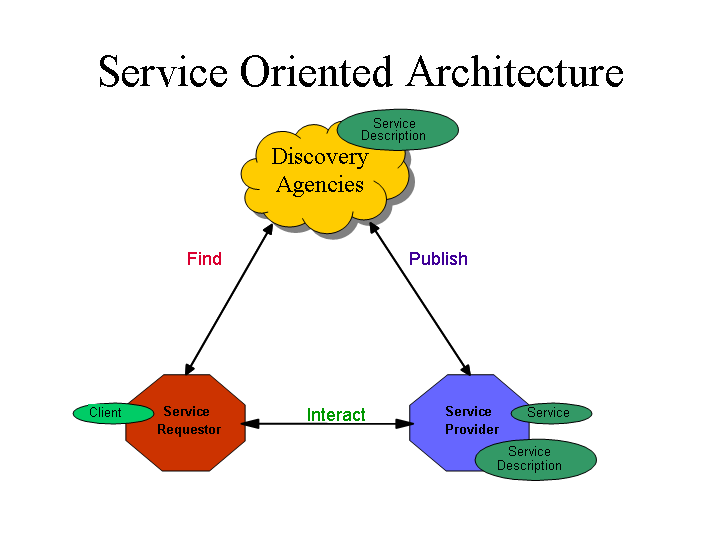
PAAS:

example:AWS Lambda,Azure, etc.

SAAS:

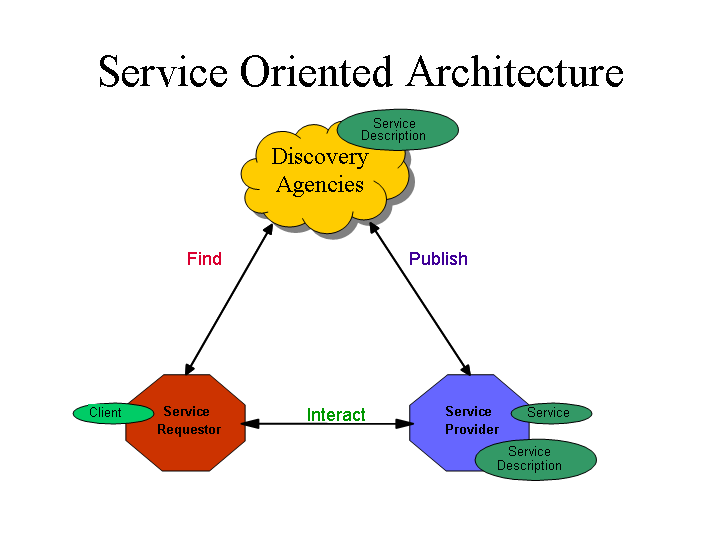
example:Canva, Microsoft office, **Salesforce**, etc.

MICROSERVICES:

* Monolith (opposite of microservices)
  + Difficult to integrate.
  + Eg: all ERP’s (SAP)
  + Not easy to upgrade.
  + Not easy to scale.
    - Horizontal scaling
    - Vertical scaling
* SOA- Service oriented Architecture.( In [software engineering](https://www.bing.com/ck/a?!&&p=e2d58d73b29d6b98JmltdHM9MTcyMTE3NDQwMCZpZ3VpZD0wN2RmYTNmYi1mMWRmLTY3NmMtMDc5Zi1iNzQ2ZjA3MjY2OGImaW5zaWQ9NTAwOQ&ptn=3&ver=2&hsh=3&fclid=07dfa3fb-f1df-676c-079f-b746f072668b&u=a1L3NlYXJjaD9xPVNvZnR3YXJlJTIwZW5naW5lZXJpbmclMjB3aWtpcGVkaWEmZm9ybT1XSUtJUkU&ntb=1), service-oriented architecture (SOA) is an architectural style that focuses on discrete services instead of a [monolithic design](https://www.bing.com/ck/a?!&&p=6734786070f12795JmltdHM9MTcyMTE3NDQwMCZpZ3VpZD0wN2RmYTNmYi1mMWRmLTY3NmMtMDc5Zi1iNzQ2ZjA3MjY2OGImaW5zaWQ9NTAxMA&ptn=3&ver=2&hsh=3&fclid=07dfa3fb-f1df-676c-079f-b746f072668b&u=a1L3NlYXJjaD9xPU1vbm9saXRoaWMlMjBzeXN0ZW0lMjB3aWtpcGVkaWEmZm9ybT1XSUtJUkU&ntb=1). SOA is a good choice for [system integration](https://www.bing.com/ck/a?!&&p=c5c97ea3a6af25d1JmltdHM9MTcyMTE3NDQwMCZpZ3VpZD0wN2RmYTNmYi1mMWRmLTY3NmMtMDc5Zi1iNzQ2ZjA3MjY2OGImaW5zaWQ9NTAxMQ&ptn=3&ver=2&hsh=3&fclid=07dfa3fb-f1df-676c-079f-b746f072668b&u=a1L3NlYXJjaD9xPVN5c3RlbSUyMGludGVncmF0aW9uJTIwd2lraXBlZGlhJmZvcm09V0lLSVJF&ntb=1).)
* Interoperability (made through soa)- Interoperability is the property that facilitates unrestricted sharing and use of data or resources between disparate systems via local area networks (LANs) or wide area networks (WANs).
* SOA principles are realized and implemented using web services.
* 

Install sts

Install JDK 21

Install Maven 3.X

**18/07/24:**

Store:

1. Eggs in arrayList
2. Milk in HashSet
3. Chocolates LinkedList
4. Apples on TreeSet
5. Bread on Queue

Put all the above in HashMap<Integer, Collection>.

::::

Class Eggs{

int quantity;

String Company;

int expiryDate;

public void Eggs(int quantity, String Company, int expiryDate){

this. quantity = quantity;

this.company=company;

this.expiryDate=expiryDate;

}

Public int getQuantity (){

return quantity;

}

}

Class Milk{

int quantity;

String Company;

int expiryDate;

public void Milk(int quantity, String Company, int expiryDate){

this. quantity = quantity;

this.company=company;

this.expiryDate=expiryDate;

}

public void setNumber(int quantity){

this. quantity = quantity;

}

Public int getQuantity (){

return quantity;

}

}

Class chocolates{

int quantity;

String Company;

int expiryDate;

public void chocolates(int quantity, String Company, int expiryDate){

this. quantity = quantity;

this.company=company;

this.expiryDate=expiryDate;

}

Public int getQuantity (){

return quantity;

}

}

Class Apple{

int quantity;

String Company;

int expiryDate;

public void Apple(int quantity, String Company, int expiryDate){

this. quantity = quantity;

this.company=company;

this.expiryDate=expiryDate;

}

Public int getQuantity (){

return quantity;

}

}

Class Bread{

int quantity;

String Company;

int expiryDate;

public void Bread(int quantity, String Company, int expiryDate){

this. quantity = quantity;

this.company=company;

this.expiryDate=expiryDate;

}

Public int getQuantity (){

return quantity;

}

}

Class Store{

public static void main(String[] Args){

ArrayList<Egg> eggArrayList=new ArrayList<>();

HashSet<Milk> milkHashSet=new HashSet<>();

LinkedList<Chocolates> chocoLinkedList=new LinkedList<>();

TreeSet<Apple> appleTree=new TreeSet<>();

Queue<Bread> breadQueue=new Queue<>();

System.out.println(“UST Store”);

System.out.println(“Select an Option:

1.Seller

2.Customer”);

Scanner sc=new Scanner(System.in);

Int opt=sc.nextInt();

If(opt==1){

System.out.println(“What you want to enter? 1.Eggs

2.Milk

3.Chocolates

4.Apples

5.Bread”);

Int opt1=sc.nextInt();

System.out.println(“Enter the number of varieties you want to enter:”);

Int var=sc.nextInt();

Switch(opt1){

Case 1:while(var!=0){

System.out.println(“Enter Company, quantity and expiryDate”);

String company=sc.nextLine();

Int quantity =sc.nextInt()’

Int expiryDate=sc.nextInt();

eggArrayList.Eggs(quantity, company, expiryDate);

var--;

}

Break;

Case 2:while(var!=0){

System.out.println(“Enter Company, quantity and expiryDate”);

String company=sc.nextLine();

Int quantity =sc.nextInt()’

Int expiryDate=sc.nextInt();

milkHashSet.milk(quantity, company, expiryDate);

var--;

}

Break;

Case 3:while(var!=0){

System.out.println(“Enter Company, quantity and expiryDate”);

String company=sc.nextLine();

Int quantity =sc.nextInt()’

Int expiryDate=sc.nextInt();

chocoLinkedList.chocolates(quantity, company, expiryDate);

var--;

}

Break;

Case 4:while(var!=0){

System.out.println(“Enter Company, quantity and expiryDate”);

String company=sc.nextLine();

Int quantity =sc.nextInt()’

Int expiryDate=sc.nextInt();

appleTree.apple(quantity, company, expiryDate);

var--;

}

Break;

Case 5:while(var!=0){

System.out.println(“Enter Company, quantity and expiryDate”);

String company=sc.nextLine();

Int quantity=sc.nextInt();

Int expiryDate=sc.nextInt();

breadQueue.bread(quantity, company, expiryDate);

var--;

}

Break;

Default:

System.out.println(“Invalid option”);

Break;

}

}

Else if(opt==2){

Sys

<bean name=”CountryEgg1” class=”com.ust.products.Egg”>

<

Loggers instead of println