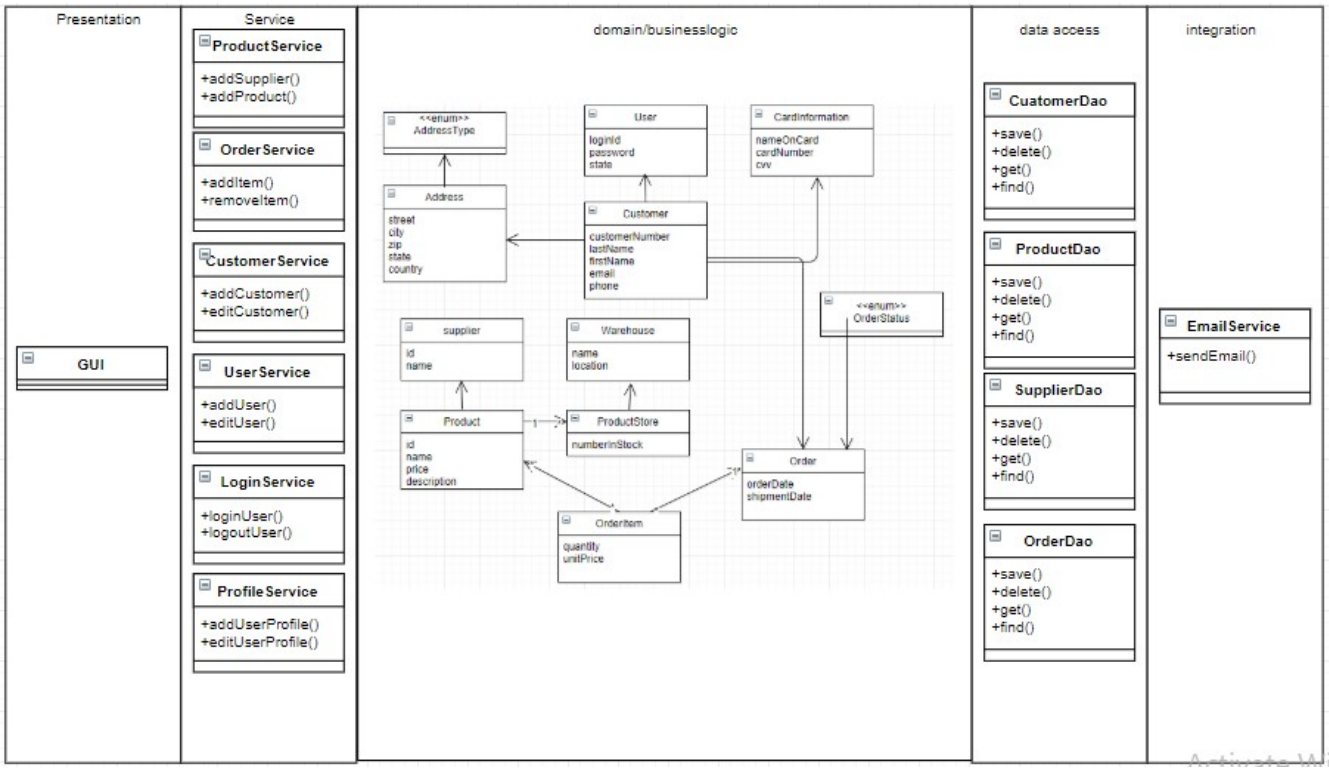
Excercise-1



Exercise 2

Describe the different places where we can store state. Give the advantages and disadvantages of these different options.

Ans: **It depends on senario**

For shopping cart data we can store in client side for example in html5 storage until order been placed or save cart temporarily to server, its ideal for Anonymous user who hasn't been login in the system.
Benefit: quick and fast since no server side call requires to generate
Problem: data might lose in-case he/she didn't save

For logged in user we can save shopping cart data in local storage in client side at the same time cart data need to be saved in server side (behind the scene, so that shopping experience can be smother) either in an RDBMS or in document database

And all the other information like product, customer, inventory, orders etc need to be saved in server side either RDBMS or document database.

Both have advantages and disadvantages

RDBMS:

Advantage:

i) Support ACID by default

Disadvantage:

NoSQL databases do not support reliability features that are natively supported by relational databases. These reliability features can be atomicity, consistency, isolation, and durability. This also means that

NoSQL databases, which don’t support those features, trade consistency for performance and scalability

NoSql:
Advantages:
I) Highly and easily scalable
ii) Maintaining NoSQL Servers is Less Expensive
iii) Lesser Server Cost and open-Source
iv) Support Integrated Caching
v) Schema less you can store arbitrary formatted data

Disadvantages
NoSQL databases do not support reliability features that are natively supported by relational databases. These reliability features can be atomicity, consistency, isolation, and durability. This also means that NoSQL databases, which don’t support those features, trade consistency for performance and scalability

Here is a matrix:

Feature	NoSQL Databases	Relational Databases
Performance	High	Low
Reliability	Poor	Good
Availability	Good	Good
Consistency	Poor	Good
Data Storage	Optimized for huge data	Medium sized to large
Scalability	High	High (but more expensive)

Exercise 3 For each of the following integration possibilities, describe its advantages, its disadvantages and when you would apply it:

1. RMI

Advantages:

-Handles threads for you
-Handles Sockets for you
-Nice GC of lost clients. ie Unreferenced
-Marshalls objects for you
-Dynamic loading of classes are available.
-Can also make changes on the server end, that might not mean you need to change anything on the client side

Disadvantages

-Strictly Java. Cannot use with other code outside Java
-Cannot guarantee that a client will always use the same thread in consecutive calls. Meaning you need to write a mechanism for identifying the client yourself
-I think it is more Hackable, security needs to be monitored more closely.

2. Messaging (JMS)

The advantages of JMS are:

- Asynchronous messaging: Due to asynchronous messaging, all the pieces don't need to be up for the application to function as a whole.
- Storage: MOM stores the messages on behalf of the receiver when it is down and then sends them once it is up.
- Application clients, Enterprise JavaBeans (EJB) components, and web components can send or synchronously receive a JMS message.
- Application clients can in addition receive JMS messages asynchronously. (Applets, however, are not required to support the JMS API.)
- Message-driven beans, which are a kind of enterprise bean, enable the asynchronous consumption of messages.
- A JMS provider can optionally implement concurrent processing of messages by message-driven beans.
– Message send and receive operations can participate in distributed transactions, which allow JMS operations and database accesses to take place within a single transaction.
–

The main disadvantage of JMS is interoperability.

If two apps are using JMS to communicate then they must be on the same implementation.

3. SOAP

Advantages of SOAP

The following are some of the many advantages that SOAP provides.

-Language neutrality :SOAP can be developed using any language.
-Interoperability and Platform Independence : SOAP can be implemented in any language and can be executed in any platform.
-Simplicity : SOAP messages are in very simple XML format.
-Scalability : SOAP uses HTTP protocol for transport due to which it becomes scalable.

Disadvantages of SOAP.
The following are the disadvantages of SOAP.

-Slow : SOAP uses the XML format which needs to be parsed and is lengthier too which makes SOAP slower than CORBA, RMI or IIOP.
-WSDL Dependence : It depends on WSDL and does not have any standardized mechanism for dynamic discovery of the services.

4. REST

-RESTful web services provide a level of simplicity which speeds things up, making integrations cost-effective.
-It requires lower learning curve to consumer.
-It supports caching of URI which improves performance.
-As compared to SOAP web services, it is light weighted which in turn improves performance.

Disadvantage of REST Web Service
It does not comprise of metadata about the implementation. Hence either the implementation should be provided as documentation/manual.

5 Serialized objects over HTTP

The advantages of serialization are:
-It is easy to use and can be customized.
-The serialized stream can be encrypted, authenticated and compressed, supporting the needs of secure Java computing.
-Serialized classes can support coherent versioning and are flexible enough to allow gradual evolution of your application's object schema.
-Serialization can also be used as a mechanism for exchanging objects between Java and C++ libraries, using third party vendor libraries (like RogueWave's Tools.h++) within C++.
-There are simply too many critical technologies that rely upon serialization, including RMI, JavaBeans and EJB.

However, serialization has some disadvantages too:
-It should ideally not be used with large-sized objects, as it offers significant overhead. Large objects also significantly increase the memory requirements of your application since the object input/output streams cache live references to all objects written to or read from the stream until the stream is closed or reset. Consequently, the garbage collection of these objects can be inordinately delayed.

-The Serializable interface does not offer fine-grained control over object access - although you can somewhat circumvent this issue by implementing the complex Externalizable interface, instead.

Since serialization does not offer any transaction control mechanisms per se, it is not suitable for use within applications needing concurrent access without making use of additional APIs.