SE Assignment -1 Anish

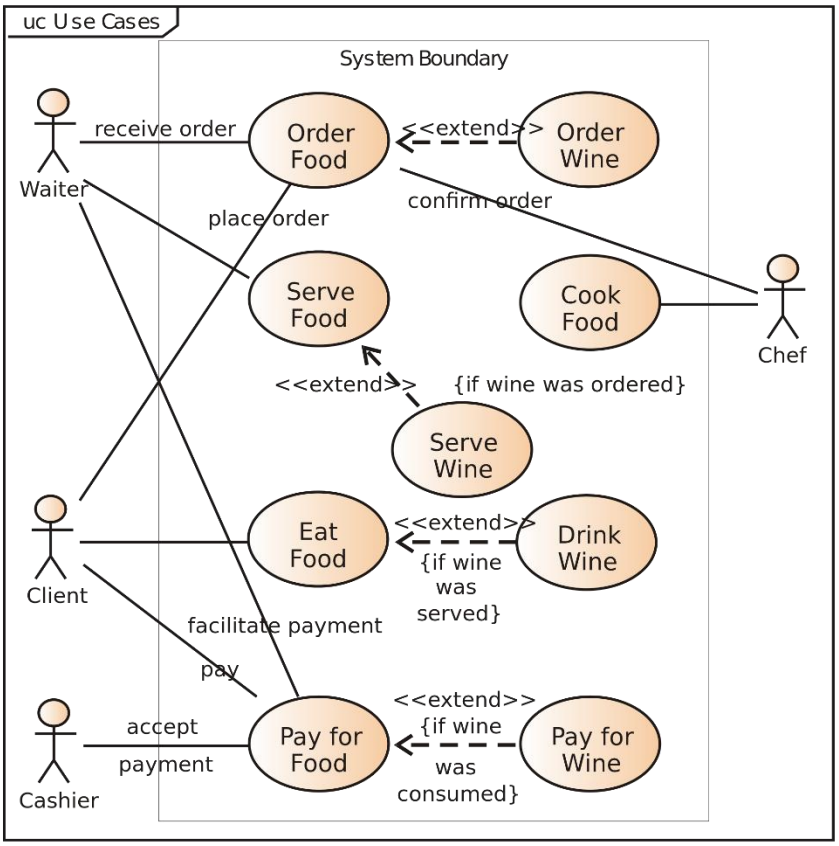
1. What is the difference between cohesion and coupling? How can coupling and cohesion lead to either good or poor software design? What are some examples that outline the difference between the two, and their impact on overall code quality?

A.Cohesion is defined as the functional strength of a module. The functional strength of a module should be high. A cohesive design class has a small focus set of responsibilities and single-mindedly applies attributes and methods to implement those responsibilities.

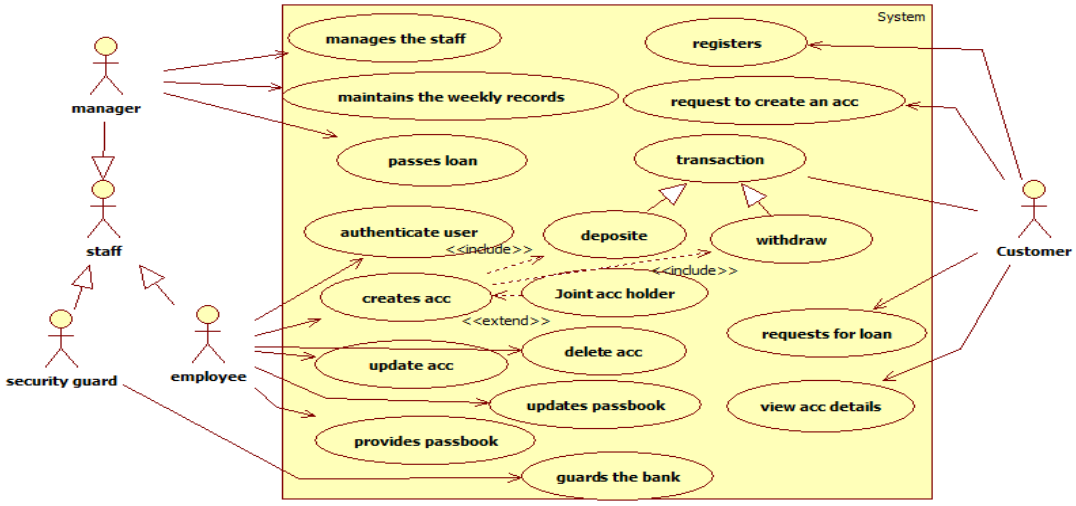
Coupling is defined as the inter-dependency between modules. The coupling of a software should be less. Within a design module, it is necessary for the design classes to collaborate with one-another. BUT, Collaboration should be kept to an acceptable minimum. If a design model is highly- coupled, that is, all design classes collaborate with all other design classes, the system is difficult to implement, to test, and to maintain over time.

Cohesion	Coupling
Cohesion is the indication of the relationship within **module**.	**Coupling** is the indication of the relationships between modules.
Cohesion shows the module's relative **functional** strength.	Coupling shows the relative **independence** among the modules.
Cohesion is a degree (quality) to which a component / module focuses on the **single** thing.	Coupling is a degree to which a component / module is connected to the **other** modules.
While designing you should strive for **high cohesion** i.e. a cohesive component/ module focus on a single task (i.e., **single-mindedness**) with little interaction with other modules of the system.	While designing you should strive for **low coupling** i.e. **dependency** between modules should be less.
Cohesion is the kind of natural extension of data hiding for example, **class** having all members visible with a package having default visibility.	Making private fields, private methods and non public classes provides loose coupling.
Cohesion is **Intra – Module** Concept.	Coupling is **Inter -Module** Concept.

2. Draw a Use case model for the given scenario



2



3.When you know programming, what is the need to learn software engineering concepts?

Which SDLC process model is best suitable Model for a Good GUI in terms of Time, cost and other resources needed to design a S/W product. Justify your answer along with the help of Strength and weakness of that Process model.

A. The development of an application is defined as programming. Software Engineering is defined as the overall development of a high- quality software product. TO make a product efficiently and profitably, a certain set of rules and standards needs to be followed. This can be learned via Software Engineering. To perform any task, an industry-standard procedure is followed.

The best SDLC Model that is suitable to design a software product that has a good GUI, in terms of time, costa and other resources is the Prototype- Process Model. It is the best model as it has a high- amount of customer-involvement from the start and is developed in accordance with the customer's requirements and choices of the GUI.

Advantages:

It increases user-involvement in the product before complete development.

Better understanding of system before development

Defects can be detected earlier so that it'll take less time and cost.

With Feedback from user, it shall be easier to develop and build the project in a better in a better way.

Some missing or confusing functionalities can be identified before-hand.

Disadvantages:

Due to insufficient requirements from the client at the initial stage, we are always dependent on them.

Methodology increases the complexity

It may increase management efforts to manage the project

Sometimes, there will be a confusion between prototype and the original design

Developer's effort will be excess if the management does not efficiently monitor

Developers may try to re-use the existing prototype, but it is not possible.

4.*What is Requirement gathering phase? What is SRS? Explain Functional and Non Functional requirements along with examples. Collect the requirements of Air lines reservation Systems.*

A.Thephase in which the service-provider initiates a request to the client in order to know the requirements and objectives of the high-quality software product to be made, via any means such as e-mail, phone calls, video-conferencing etc. is known as the requirement gathering phase.

SRS is software requirement specification. It is the documentation which provides a better understanding of the software product to be made. It is a description of a software system to be developed. It lays out the functional and non-functional requirements and may include a set of use-cases that describe user-interactions that the software must provide.

Functional Requirements:

Those requirements that determine what the system will do. It defines the function of a system and its components. It is described as a set of inputs, the behavior and the outputs

e.g.- Calculations, Technical Details, Data Manipulation and processing etc..

Non-Functional Requirements:

Those requirements that determine how the system will behave. It specifies criteria that can be used to judge the operation of a system, rather than specific behaviors

E.g. – Reliability, Security, Portability, Inter-Operability etc.

Airline Reservation System:

Airline Reservations Systems contain airline schedules, fare tariffs, passenger reservations and ticket records

The inventory of an airline is generally divided into service classes (e.g. First, Business or Economy class) and up to 26 booking classes, for which different prices and booking conditions apply. Inventory data is imported and maintained through a Schedule Distribution System over standardized interfaces. One of the core functions of the inventory management is the inventory control. Inventory control steers how many seats are available in the different booking classes, by opening and closing individual booking classes for sale. In combination with the fares and booking conditions stored in the Fare Quote System the price for each sold seat is determined.

This display contains flights, which are operated by the airline itself as well as code share flights which are operated in co-operation with another airline. The availability of seats of other airlines is updated through standard industry interfaces. Depending on the type of co-operation it supports access to the last seat (Last Seat Availability) in real-time.

5. ***Which SDLC process model is best suitable Model for the following tasks? Explain with Proper Justification***

a. Object oriented development project.
b. Software products that are prone to several kinds of Risks

A. Iterative-Model.
B. Spiral

6.Briefly describe the phases in the Spiral model. Explain the strengths and weakness of the spiral model. Explain when to use the spiral model.

The Spiral Model is the only model where each step can be analyzed after it is completed. But, we cannot be sure how many times the spiral will spiral.

Strengths:

Changing requirements can be accommodated.
Extensive use of prototypes
Requirements can be gathered, user can see the product being developed very clearly
After each step, there is customer-evaluation
The original problem can be divided into smaller modules

Weakness:

End of project cannot be predicted earlier on.
Suitable for large projects and expensive for smaller ones
Spiral may spiral indefinitely
Management is more complex
Large number of Intermediate stages required.

The Spiral model is generally used when we have changing requirements.