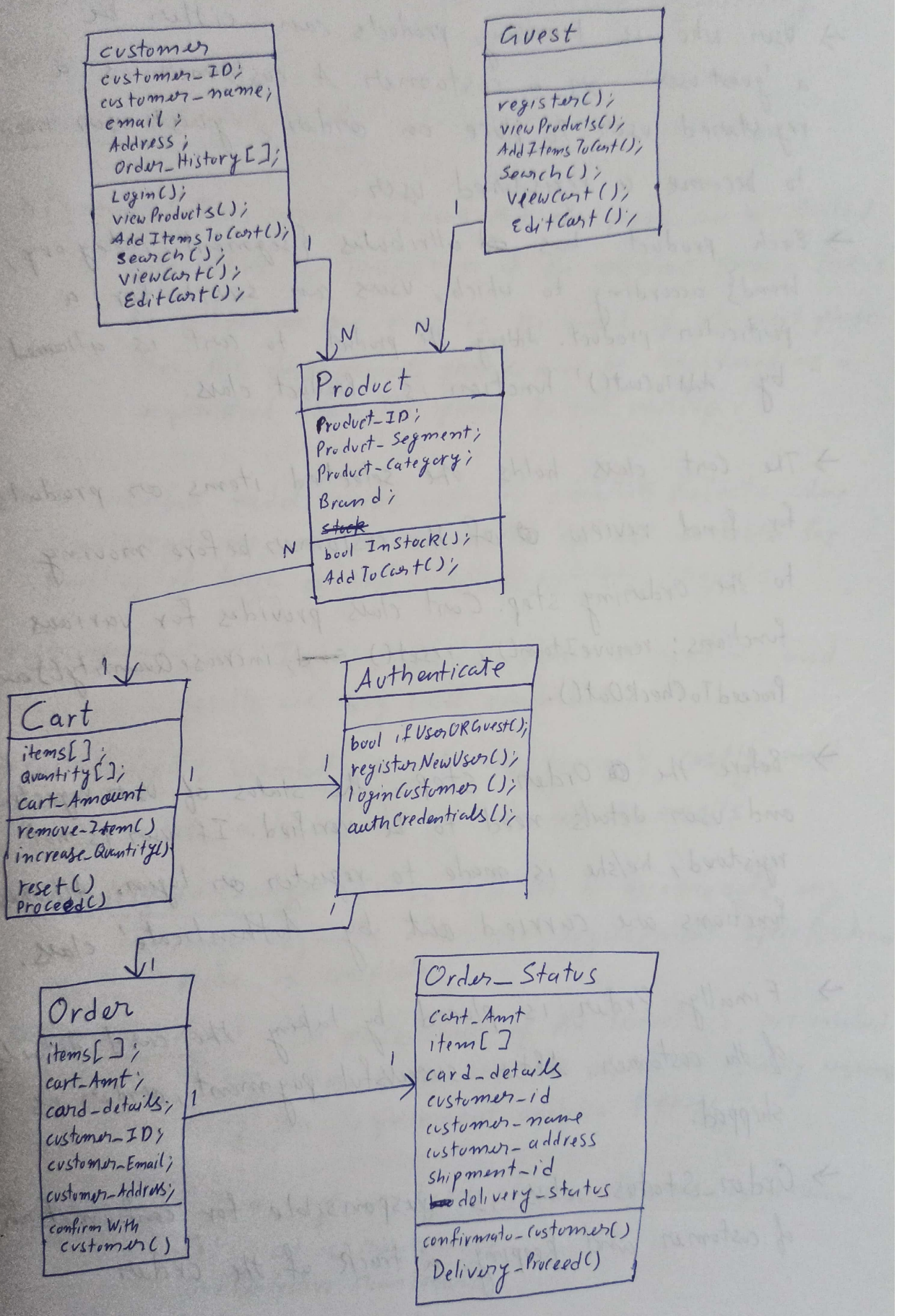
Question 1) Class Diagram for online Shopping



Details of the class Diagram:

→ User who is browsing products can either be a 'guest user' or a customer. A customer is a registered user. To place an order, guest user need to become a registered user.

→ Each product has attributes {segment, category, brand} according to which, users can search for a particular product. Adding the product to cart is allowed by 'AddToCart()' function is Product class.

→ The Cart class holds the selected items or products for final review of the customer before moving to the Ordering step. Cart class provides for various functions: removeItem(), reset(), increaseQuantity() and ProceedToCheckOut().

→ Before the Order step, the status of user registration, and user details need to be verified. If user is not registered, he/she is made to register or login. All these functions are carried out by 'Authenticate' class.

→ Finally Order is placed by taking the card details of the customer. After successful payment, order is shipped.

→ Order_Status class is responsible for confirmation of customer and keeping a track of the order.

Question 2)

Comparative study of Different Software Engineering Models.

(i) Classical Waterfall Model

→ Planning: This model considers that one phase can be started only after the completion of the previous phase. That is the output on one phase is the input of the next phase. Thus, this development process can be considered as a sequential flow. The phases do not overlap.

→ The Waterfall model works well for smaller projects where requirements are well understood. It is not suitable for handling of large and complex problems/projects.

→ Documentation: In this model, processes, actions and results are very well documented.

→ Cost/Budget: Waterfall projects are typically costly and take a lot of time to deliver the results.

→ Flexibility to Change: It is difficult to accomodate any change requests after the requirement specification phase is complete.

→ User Involvement: User or Customer of the Product is not involved in any stage of the Waterfall model. He is only required during the Requirement analysis phase.

→ Maintenance: Three ways or types of maintenance is allowed,
(i) Corrective Maintenance
(ii) Perfective Maintenance
(iii) Adaptive Maintenance

Duration: Time taken to deliver the product is comparatively long than other models.

Risk Analysis: In Waterfall model, ~~plan~~ during planning phase, we tend to plan risks well ahead of time, so our estimation about the likelihood or severity of risks might not be accurate.

Testing: In waterfall model, Testing is a part of two different phases, these phases lie consequtive and are: Coding and Unit testing phase and Integration and system testing.

The the Unit testing phase, each module that has been coded is tested individually.

In Integration testing, integration of different modules is carried out incrementally until all the modules are added and one full product is assembled. After this three different kinds of system testing is done: Alpha testing, Beta testing and Acceptance testing.

Team size: Team size is above moderate to large because of the heirarchical nature of the team required to work in a project using Waterfall model.

Reusability: Not ~~Reusable~~ Reusable because every waterfall model is strictly designed during to planning phase to perfectly fit the requirements of the project in hand.

(ii) Iterative Waterfall Model

→ Planning: The iterative waterfall model provides feedback paths from every phase to its preceding phases, which is the main difference from the classical waterfall model.

→ It is somewhat difficult to implement for handling large projects, it is best suited for small projects with varying levels of complexities.

→ Documentation: In this model, processes, actions and steps are documented consistently.

→ Cost/Budget: The Iterative Waterfall Model is cost-effective as compared to ~~other~~ most other models.

→ Flexibility to Changes: It is difficult to incorporate change requests in this model.

→ User involvement: User or Customer involvement is limited

→ Maintenance: Maintenance is same as classical waterfall model, but feedback is sent to the previous steps depending on the situation.

→ Duration: same as classical waterfall model.

→ Risk Analysis: Risk handling and analysis is not supported in this model.

→ Testing: Testing is same as traditional waterfall model

→ Team size: same as classical waterfall model

→ Reusability: Not reusable

(iii) Incremental Model

→ Planning: In this model, requirements are divided into multiple standalone modules, each module goes through its own design, implementation and testing phases.

→ When requirements are superior and the project is large in terms of duration, Incremental Model is best suited.

→ Documentation: As the project is divided into modules, each module has its own documentation process, thus making it difficult to have one single documentation for the entire project.

→ Cost/Budget: Total cost is high.

~~→ Flexibity~~

→ Flexibility to changes: More Flexible than other models.

→ User Involvement: Due to presence of different modules, user involvement is present but is very tricky.

→ Maintenance: Maintenance for each module is done separately so it is simple and easy.

→ Duration: Although it is best suited for long term projects, it can deliver results of short projects quicker than other models. Duration is small.

→ Risk Analysis: In this model, it is simple to manage risks because it is handled during its iteration.

→ Testing: Testing and debugging is simple and easy.

→ Teamsize: Team size is big since there are different modules being developed at the same time.

→ Revsability: Revsability is allowed.

(iv) Spiral Model

→ Planning: This model works like a spiral with many loops. The exact number of loops in the spiral is unknown and can vary from project to project. Each loop is known as phase.

→ This model is very good for handling large projects.

→ Documentation: Documentation is allowed and available.

→ Cost/Budget: It is not cost effective, it is expensive.

→ Flexibility in Requirements: Change requests in the Requirements at later phase can be incorporated accurately by using this model.

→ User Involvement: Customer can see the development of the product at the early phase of the software development and thus, they habituated with the system by using it before completion of the total product.

→ Maintenance: The spiral model enables gradual releases and refinement of a product through each phase of the spiral as well as the ability to build prototypes at each phase.

→ Duration: As the number of phases is unknown at the start of the project, so time estimation is very difficult.

→ Risk Analysis: The projects with many unknown risks that occur as the development proceeds, in that case, Spiral model is the best development model to follow due to the risk analysis and risk handling at every phase.

→ Testing: In spiral model testing is done in two phases, Engineering phase where development and testing is done. ~~simultaneously~~ Evaluation phase, in this phase consumer is allowed to evaluate the project.

→ Teamsize: Large team size is required as this model mostly caters to large projects.

→ Reusability: Reuse might be allowed for some projects.

(V) Rapid Application Development Model (RAD)

→ Planning: It involves the use of various techniques used in requirement elicitation like brainstorming, task analysis, form analysis, user scenarios, FAST (Falicitated Application Development Technique), etc. It also consists of the entire structured plan describing the critical data, methods to obtain it and the processing it to form the final refined model.

→ This model is efficient only for handling large projects.

→ Documentation: Documentation is available.

→ Cost/Budget: In small-scale projects, the cost of using automated tools and techniques may exceed the entire budget of the project. On the other hand, man power is less, so cost from that front is less.

→ Flexibility in requirements: changes in requirements can be handled ~~smot~~ smoothly.

→ User involvement: Consumer/User involvement is required through out the life cycle.

→ Maintenence: Once testing is done, the product could be deployed on the server. A deployed project generally requires maintenance and maybe an additional a few extra features.

→ Duration: The time frame for delivery is generally 60–90 days.

→ Risk Analysis: Risk analysis is done during the requirements planning phase.

→ Testing: The testing phase involves testing of the product developed. There is a team that is involved in the proper testing of the developed product.

→ Team size: A comparatively smaller development team is required.

→ Reusability: Reusability is allowed and extensively used. The use of resusable components helps to reduce cycle time of the project.

(vi) Agile Model

→ Planning: The requirements are decomposed into smaller parts that can be incrementally developed. The agile model adopts Iterative development. Each increment part is developed over an iteration.

→ This model is suitable for any size of project as it adapts to the requirements of each project.

→ Documentation: There is no documentation.

→ Cost/Budget: Cost highly depends on the project on hand.

→ Flexibility in requirements: Requirements change requests from customers are encouraged and efficiently incorporated.

→ User Involvement: To establish close contact with the consumers during development and gain a clear understanding of various requirements, each Agile project usually includes a customer representative on the team. At the end of each iteration stakeholders and the customer representative review, the progress made and re-evaluate the requirements.

→ Maintenance: Due to the absence of proper documentation, when the project completes and the developers are assigned to another project, maintenence of the developed project can become a problem.

→ Duration: Total development time is very short.

→ Risk analysis: Risk analysis is done using tools like: Exhaustive risk checklists, review of documents, analysis of assumptions and constraints and so on.

→ Testing: same as most other models, Unit testing is the phase after completion of Coding phase and then acceptence testing is carried out.

→ Teamsize: It is recommended that the development team size should be kept small (5 to 9 people) to help the team members meaningfully engage in face to face communication and have collaburative work environment

→ Reusability: Reusability is allowed in this model

(vii) Prototype Model

→ Planning: A working model or a prototype is created for the consumer to test and consumer feedback is taken. Then the prototype is edited or reformed based on feedback and again shown to consumer. This keeps on going until consumer is satisfied. The final prototype is used as basis for development.

→ This model is good for ~~any project~~ projects ~~also~~ that are large in duration, size and complexity.

→ Cost/Budget: costly with respect to money

→ Flexibility in requirements: New requirements can easily be accomodated as there is scope for refinement.

→ User Involvement: User is involved in the entirety of the prototype phase and gives a lot of input in this phase.

→ Maintenance: Need for maintenance is there if the requirements change after the completion of project.

→ Duration: This model takes a lot of time.

→ Risk Analysis: There is a risk of faulty engineering because of the haste of customers after they have seen the prototype. Consumer might lose interest in the project if prototyping phase was not successful.

→ Testing: Testing is mostly same as other models.

→ Team-size: Highly depends on the project in hand.

→ Reusability: The developed prototype can be used by the developer for more complicated projects in the future.