**AGILE**

* Software methodology.
* Deliver small, incremental changes to product in a collaborative, flexible, iterative manner
* Continuous delivery
* Changing requirements, flexibility

Key Principles

* Customer Collaboration Over Contract Negotiation
  + Continuous feedback from customers instead of being fixated on requirements
* Individuals and Interactions Over Processes and Tools
  + Skilled people working together instead of depending on process and tools
* Working Software Over Comprehensive Documentation
  + Quickly deliver, instead of documenting and wasting time
* Responding to Change Over Following a Plan
  + Embrace change even if it is last minute

Product Backlog

* Prioritized list of features, bug fixes, tech tasks
* Owned by product owner
* Constantly changes
* Contains user stories
* Refined by team regularly

Sprint

* Time bound work.
* Shippable after each sprint

Sprint planning

* Meeting to establish outcome of a sprint
* Takes place at the beginning
* Owner presents the product backlog to team and they choose their tasks

Sprint review

* Meeting to discuss outcome of sprint and gather feedback
* Demo at end
* Update backlog

Backlog refinement

* Update backlog

User stories

* Features written in simple words from user’s perspectives
* "As a [role], I want [action] so that [benefit]"
* Ex: "As a customer, I want to reset my password so that I can access my account if I forget it."
* User can receive a password reset email.
* The reset link expires after 24 hours.

**Extreme programming**

* Agile methodology that I used in SMTS
* Pair Programming:
  + Two developers work together at one workstation.
* Test-Driven Development (TDD):
  + Writing tests before writing the actual code.
* Continuous Integration:
  + Frequent code integration and automated testing.
* Small Releases:
  + Frequent releases to get feedback as early as possible.
* Use Case: Suitable for projects with rapidly changing requirements that demand high-quality code.

In XP Sprint -> Iteration (sprint review -> iteration review.)

**Interview questions**

1. What is Agile, and why did your team choose to use it?
   1. Why? -> improve responsiveness, increase collaboration, deliver faster, handle changing requirements
2. Which Agile framework did your team use, and why?
   1. Extreme programming. Why? -> daily standups, sprint planning, retrospectives, User stories
3. How did you handle changing requirements in Agile projects?
   1. Why? -> emphasize flexibility, continuous feedback, regular client meeting, sprint reviews, prioritize tasks
4. How did you ensure the quality of the software during Agile development?
   1. Test-Driven Development (TDD), continuous integration, and automated testing
5. How did your team prioritize tasks or user stories?
   1. MoSCoW (Must have, Should have, Could have, Won’t have) or story points for estimation
6. Can you share a challenging situation you faced in an Agile project and how you resolved it?
   1. Describe a specific instance, such as handling a sudden change in requirements mid-sprint or managing team conflicts. Explain the actions taken to address the issue, such as adjusting the sprint backlog or improving communication.