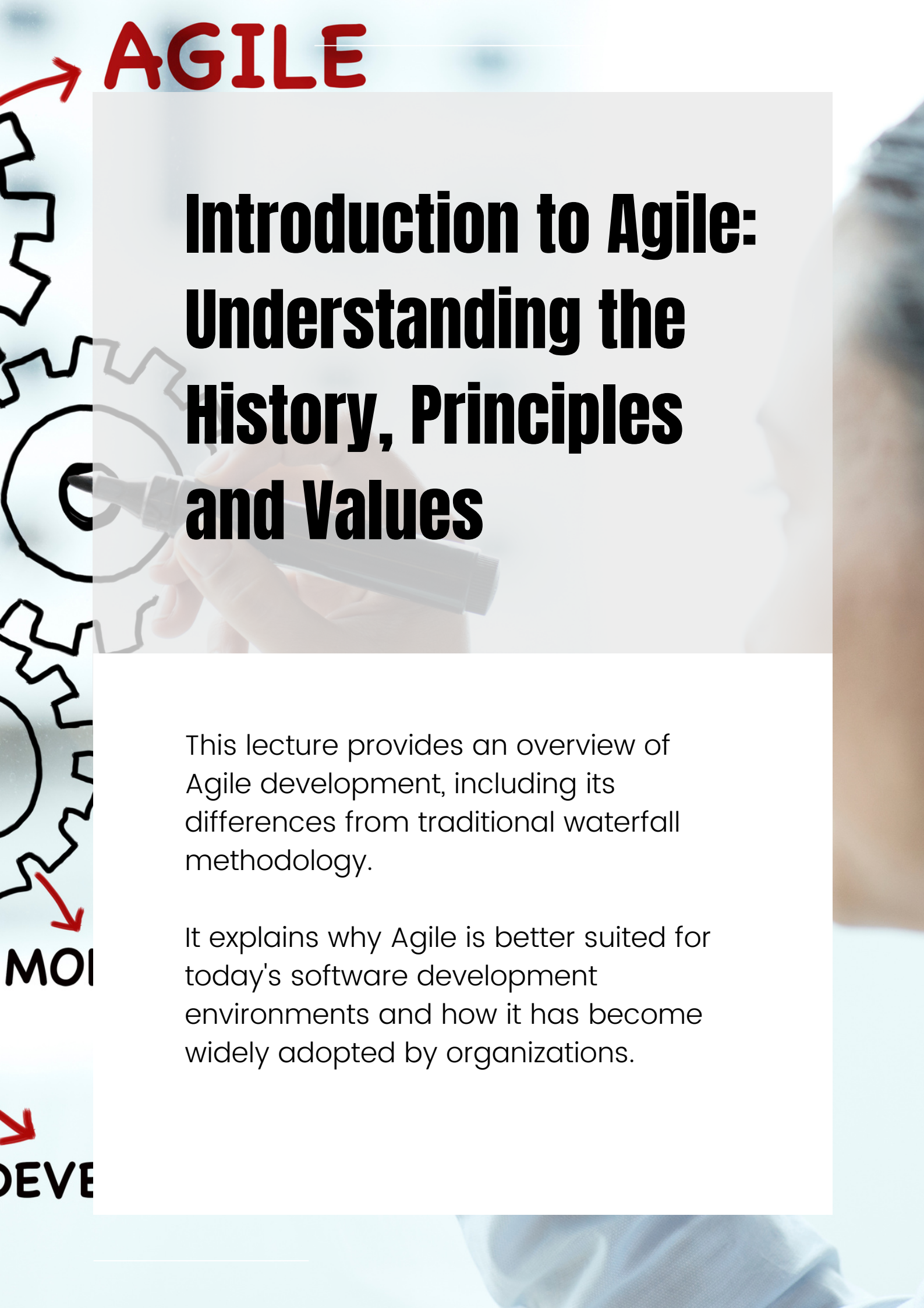


Introduction to Agile: Understanding the History, Principles and Values

This lecture provides an overview of Agile development, including its differences from traditional waterfall methodology.

It explains why Agile is better suited for today's software development environments and how it has become widely adopted by organizations.

FROM WATERFALL TO AGILE:

UNDERSTANDING THE SHIFT IN PROJECT MANAGEMENT

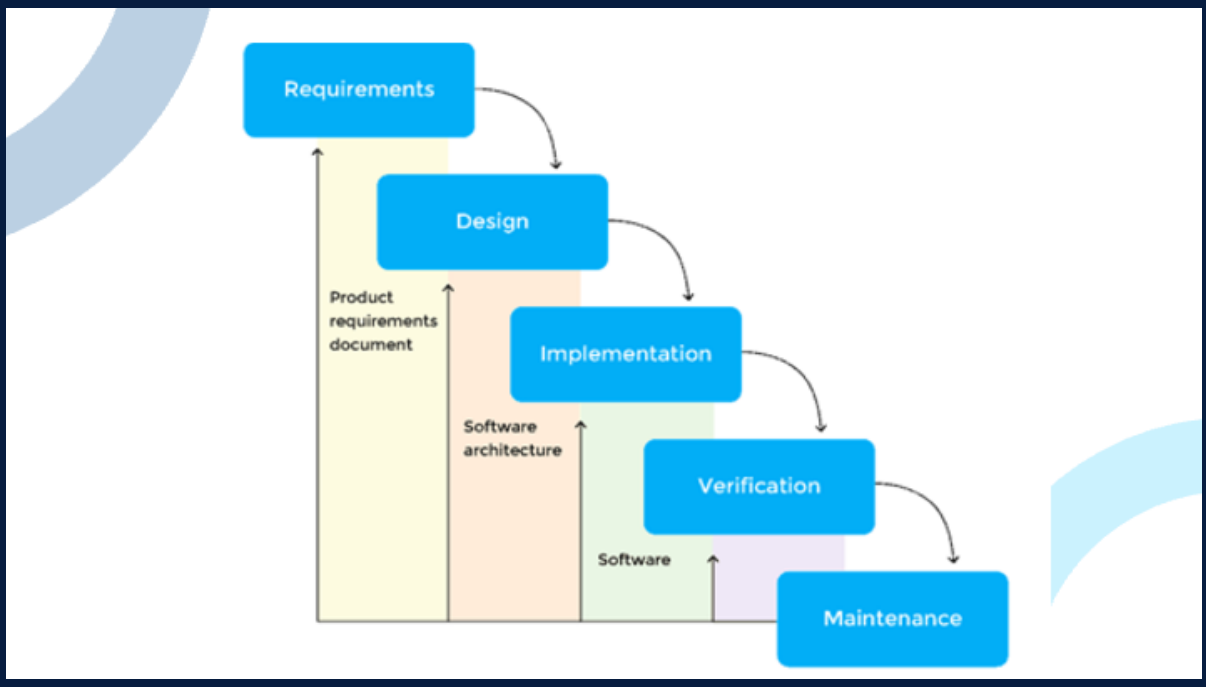
WE MUST FIRST UNDERSTAND THE PROBLEM THAT AGILE IS TRYING TO SOLVE

We look at a problem in traditional project management with the assumption that we can do enough analysis and research to understand everything that needs to be included in the solution.

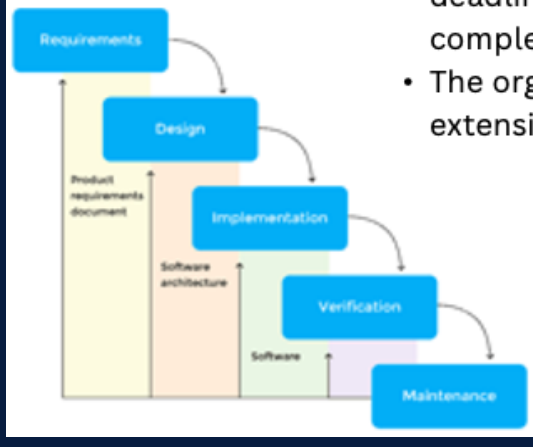
However, we have discovered that most projects planned in this manner have a number of issues.

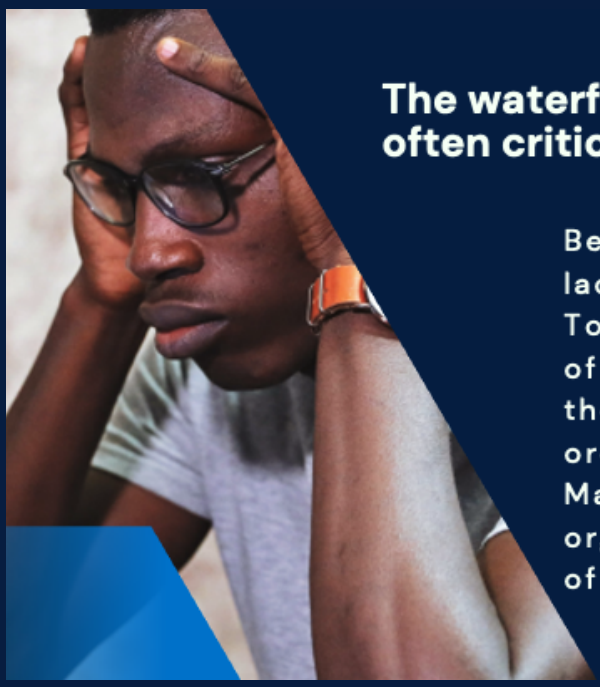
- Projects take longer than we anticipated
- They cost more than we anticipated
- Does not meet the needs of the customers by the time we deliver it

The waterfall model



- The project is made up of sequential, linear phases.
- The organization must plan, define deadlines and budgets, and implement one complete system at a time.
- The organization keeps control of it via extensive documentation





The waterfall model is often criticized for

Being overly bureaucratic and lacking in flexibility.
Too rigid because it is made up of phases that flow from one to the next in a predetermined order
Making it difficult for organizations to react in the face of changing information.

SOME OF THE DISADVANTAGES OF THE WATERFALL MODEL:

- Working software isn't created until late in the project life cycle
- There's a large amount of risk and uncertainty
- Not suited for complex and object-oriented projects
- It is unsuitable for long and ongoing projects
- Measuring the progress within stages is difficult
- Changing requirements cannot be accommodated
- The end-user/client isn't focused on
- Testing is delayed until the project is completed

THE EVOLUTION OF AGILE:

THE VALUES AND PRINCIPLES

THE EVOLUTION OF AGILE THINKING

Dissatisfaction with traditional software development methodologies became more widespread in the late 1990s and early 2000s

In the spring of 2000, a group of influential figures in the software development field gathered in Oregon to address these challenges.

Another significant meeting occurred in Snowbird, Utah, in 2001. The Agile Manifesto was formalized at this point

FOUR KEY AGILE VALUES

- **Individuals** and interactions over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan.

THE GROUP ALSO DEFINED 12 PRINCIPLES THAT SUPPORT THESE VALUES

1.
Our highest priority is to **satisfy the customer** through early and continuous delivery of valuable software.

2.
Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

3.
Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for a shorter timescale.

4.
Businesspeople and developers must **work together** daily throughout the project.

5.
Build projects around **motivated individuals.** Give them the environment and support they need and trust them to get the job done.

6.
The most efficient and effective method of conveying information to and within a development team is **face-to-face conversation**.

7.
Working software is the primary measure of progress.

8.
Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a **constant pace** indefinitely.

9.
Continuous attention to technical excellence and **good design** enhances agility.

10.
Simplicity--the art of maximizing the amount of work not done--is essential.

11.
The best architecture, requirements, and designs emerge from **self-organizing teams.**

12.
At regular intervals, the team **reflects** on how to become more effective, then tunes and **adjusts** its behavior accordingly.

WHAT IS AGILE?

Agile project management is a project philosophy or mindset that takes an iterative approach toward the accomplishment of a task or a project.

GOAL

To create early measurable ROI through defined, iterative delivery of product features meaning delivering a simple version of a project to the client, taking their feedback, and making the improvements per that feedback.

IN SUMMARY

By managing projects and developing software iteratively, Agile enables organizations to deliver value to customers faster and with fewer complications.

Deliver work in small, but consumable, increments rather than betting everything on a "big bang" launch.

Teams are able to respond to change in a timely manner as a result of continuously evaluating requirements, plans, and results.

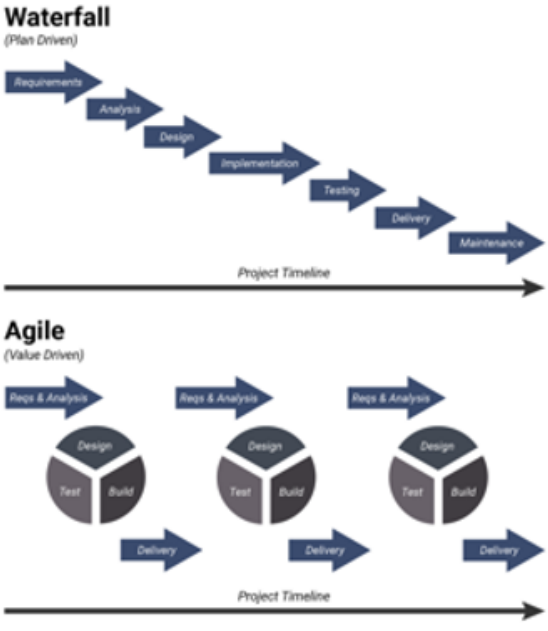
WHY DOES AGILE DEVELOPMENT DELIVER BETTER SOFTWARE THAN TRADITIONAL METHODS?

Agile is built for adaptability and flexibility. Instead, you divide the problem into manageable chunks, which you then develop and test with users.

If something isn't working as well as expected, or if the effort reveals something you hadn't considered, you can quickly adapt the effort and get back on track—or even change tracks if necessary.

Agile allows each team member to make a contribution to the solution while also requiring each member to take personal responsibility for their work.

Agile teams are typically more productive and happier.



OVERVIEW OF COMMON AGILE FRAMEWORKS

There are over fifty different Agile frameworks available today. I'll just mention some of the more common ones here

- **Scrum:** A highly structured framework designed to manage complex projects, emphasizing teamwork, accountability, and iterative progress.
- **Kanban:** Visualizes workflow, allowing teams to optimize flow and reduce waste by focusing on continuous delivery.
- **Lean:** Rooted in manufacturing, this approach aims to enhance efficiency by minimizing waste and streamlining processes.

- **DSDM** (Dynamic Systems Development Method): A robust Agile project delivery framework that ensures all aspects of a project align with business goals.
- **XP** (Extreme Programming): Prioritizes customer satisfaction, emphasizing flexibility to adapt to changing requirements.
- **FDD** (Feature Driven Development): Focuses on building features, with a detailed modeling phase to capture and analyze requirements.
- **Crystal:** A family of Agile methodologies emphasizing collaboration and adaptability. It tailors processes to team size and project needs.

AGILE OVERVIEW:

ESSENTIAL CONCEPTS AND WORKING MODEL

Key Agile Concepts

- **User Stories:** Concise descriptions of a feature from an end-user's perspective. They detail what users need or want, ensuring the development delivers tangible value.
- **Daily Meeting:** A daily sync-up where the team gathers to discuss progress, challenges, and plans for the day to ensure alignment and coordination.
- **Personas:** Fictional, detailed profiles representing potential users of the product, created to guide design and development decisions.
- **Team:** In Agile, a "team" refers to a dedicated group of professionals collaboratively working on the same project or initiative.

- **Incremental Development:** In Agile, teams develop the product in small, tangible increments, with each increment building upon the previous, delivering visible user functionality.
- **Iterative Development:** Agile approaches allow for cyclical development processes, giving teams opportunities to revisit and refine previously completed work based on feedback or changing requirements.
- **Milestone Retrospective:** A dedicated session where the team reflects on significant project events and lessons learned after a substantial period or milestone.

HOW DOES AGILE WORK?

Define the project: The team, along with the customer, defines the project's goals, objectives, and requirements.

Create a backlog: A backlog is a prioritized list of tasks that need to be completed. The customer, product owner, and the team work together to create the backlog.

Plan the sprint: The team plans the sprint by selecting the highest-priority tasks from the backlog and determining how much work can be completed in the upcoming sprint.

Execute the sprint: The team works on completing the tasks planned for the sprint, with daily meetings to check progress and address any issues.

Review and demo: At the end of the sprint, the team demonstrates the completed work to the customer and gets feedback.

Retrospect: The team retrospects on the sprint, discussing what went well, what didn't, and what can be improved for the next sprint.

Repeat: The process is repeated for each sprint until the project is completed. The product is incrementally developed and delivered to the customer in small chunks.

Continuously improve: Agile methodologies focus on continuous improvement. The team reflects on its progress and makes adjustments as necessary to improve processes, tools, and communication for the next sprint.

WHEN SHOULD YOU USE AGILE PROJECT MANAGEMENT?

Agile is a project management approach that is better suited for **ongoing projects** and projects where **specific details are unclear** from the start.

As a result, an Agile approach is a good choice for a **project lacking precise constraints, deadlines, or resources.**

IS AGILE PROJECT MANAGEMENT ONLY FOR SOFTWARE?

While the Agile methodology was originally intended for software development, it has since expanded to include a wide range of projects.

Agile principles are better suited to projects that result in tangible deliverables rather than services.

ANY OF THE FOLLOWING TYPES OF PROJECTS CAN BENEFIT FROM AGILE:

- Projects with **fast-changing deliverables**
- Projects that **evolve or lack clear scope** and **requirements** at the beginning.
- Projects that require **frequent customer interaction** and collaboration with external parties
- Projects focused on **innovation and continual improvements.**

- Projects with many **interdependent tasks** and teams that need to **work closely together**.
- Projects that require **building a prototype** before the final deliverable
- Projects that must be able to **act on feedback during development.**

THE FUTURE OF AGILE

Agile methodologies will **continue to evolve** and will quickly incorporate emerging technologies such as artificial intelligence, machine learning, and the Internet of Things.

This integration will open up new avenues for innovation and efficiency in software development and elsewhere.

Agile will expand its scope beyond software development.

SEVERAL KEY DEVELOPMENTS AND TRENDS

Scaling Agile: The need to scale Agile practices to larger teams and more complex projects is apparent

Agile in Non-IT Domains: Agile is now being used in marketing, sales, human resources, operations, and project management.

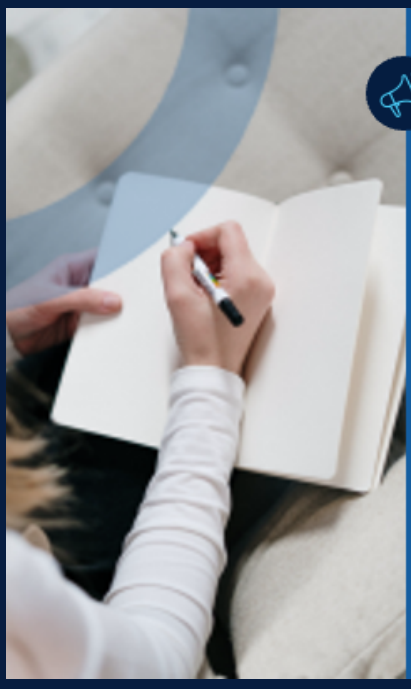
DevOps and Continuous Delivery: Agile is supplemented by DevOps, which shows continuous integration, continuous delivery, and infrastructure automation.

Agile Transformation and Organizational Culture: There is an increasing focus on developing Agile capabilities, creating a learning culture, enabling self-organizing teams, and changing leadership styles.

Agile and Hybrid Approaches: Hybrid models enable organizations to leverage the strengths of various approaches.

EXERCISE:

WATERFALL VS. AGILE: PERSONAL REFLECTION"



INSTRUCTIONS:

1. Reflection :
Imagine you're about to start on a personal project, such as writing a book, starting a blog, or learning a new skill. Which approach (Waterfall or Agile) resonates more with how you'd naturally approach this project? Why?"

2. Follow-Up Questions:

- How might your chosen methodology benefit your personal project?
- What challenges might you face using this method?
- Think about a past personal project or task. Would a different approach (Waterfall or Agile) have resulted in a better outcome? Why or why not?