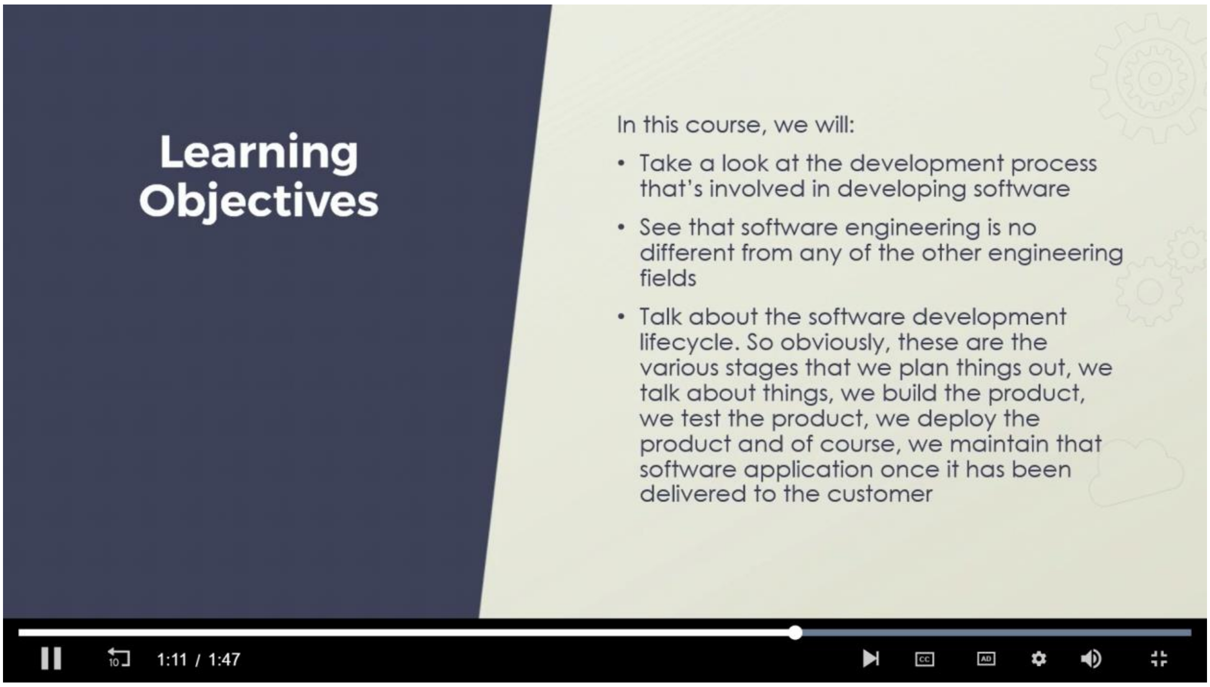
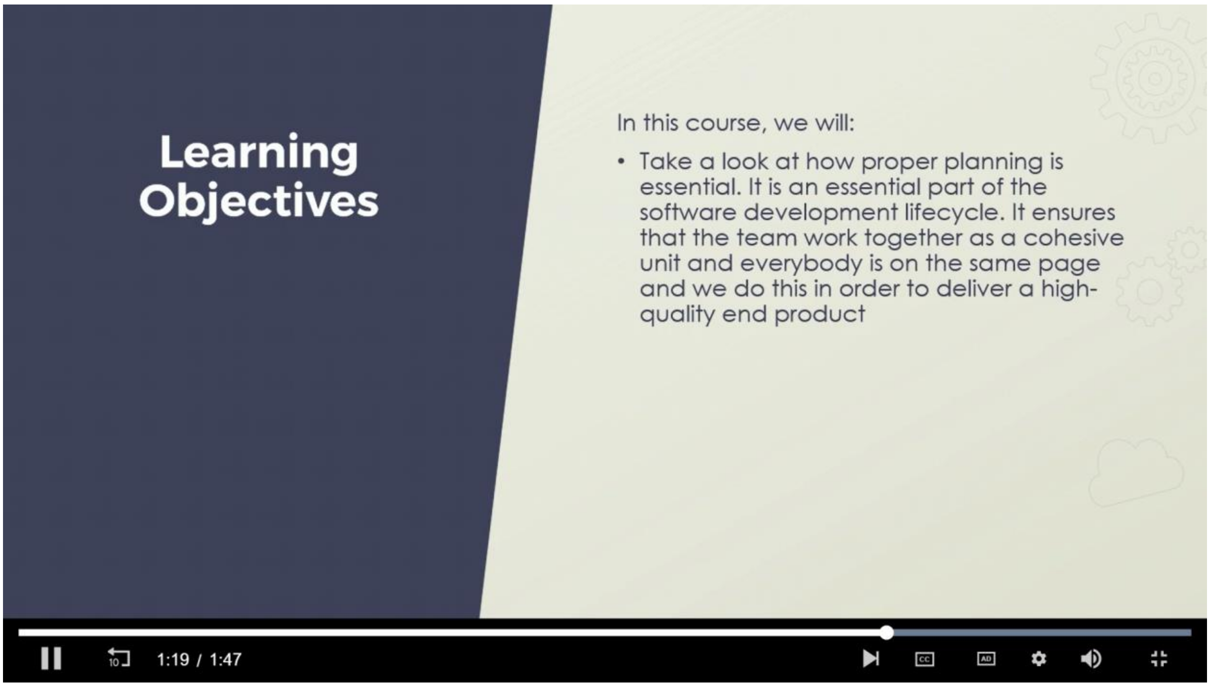


Learning Objectives

In this course, we will:

- Take a look at the development process that's involved in developing software
- See that software engineering is no different from any of the other engineering fields
- Talk about the software development lifecycle. So obviously, these are the various stages that we plan things out, we talk about things, we build the product, we test the product, we deploy the product and of course, we maintain that software application once it has been delivered to the customer



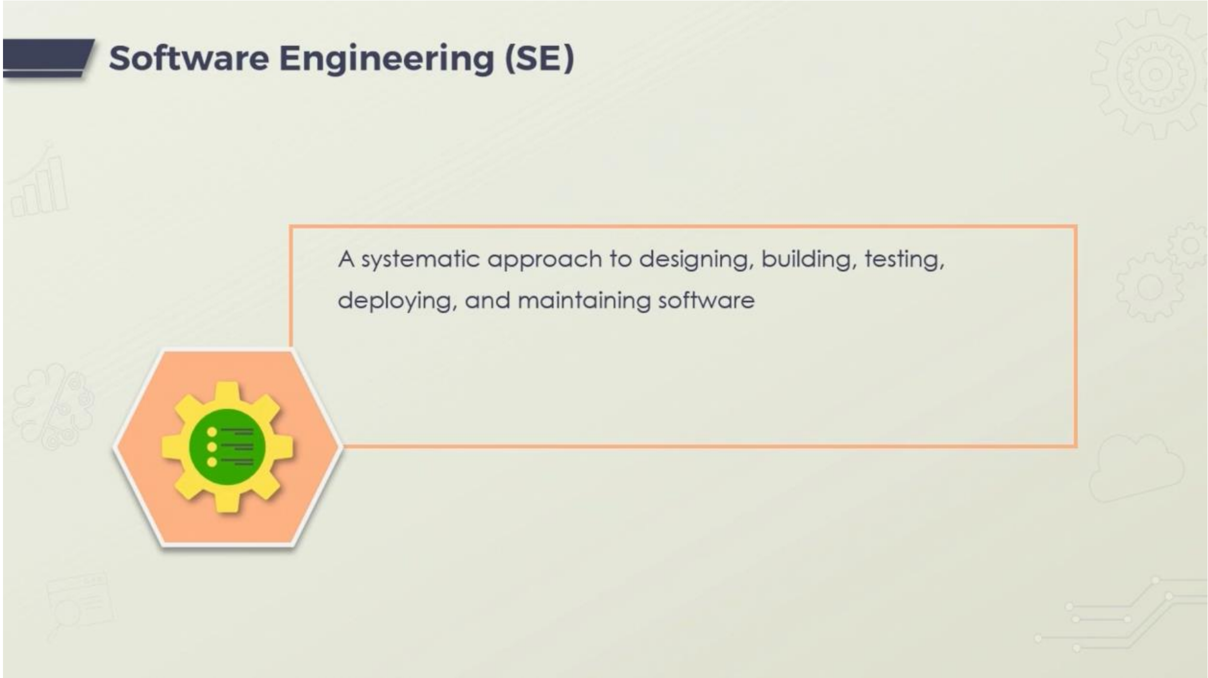
Learning Objectives

In this course, we will:

- Take a look at how proper planning is essential. It is an essential part of the software development lifecycle. It ensures that the team work together as a cohesive unit and everybody is on the same page and we do this in order to deliver a high-quality end product

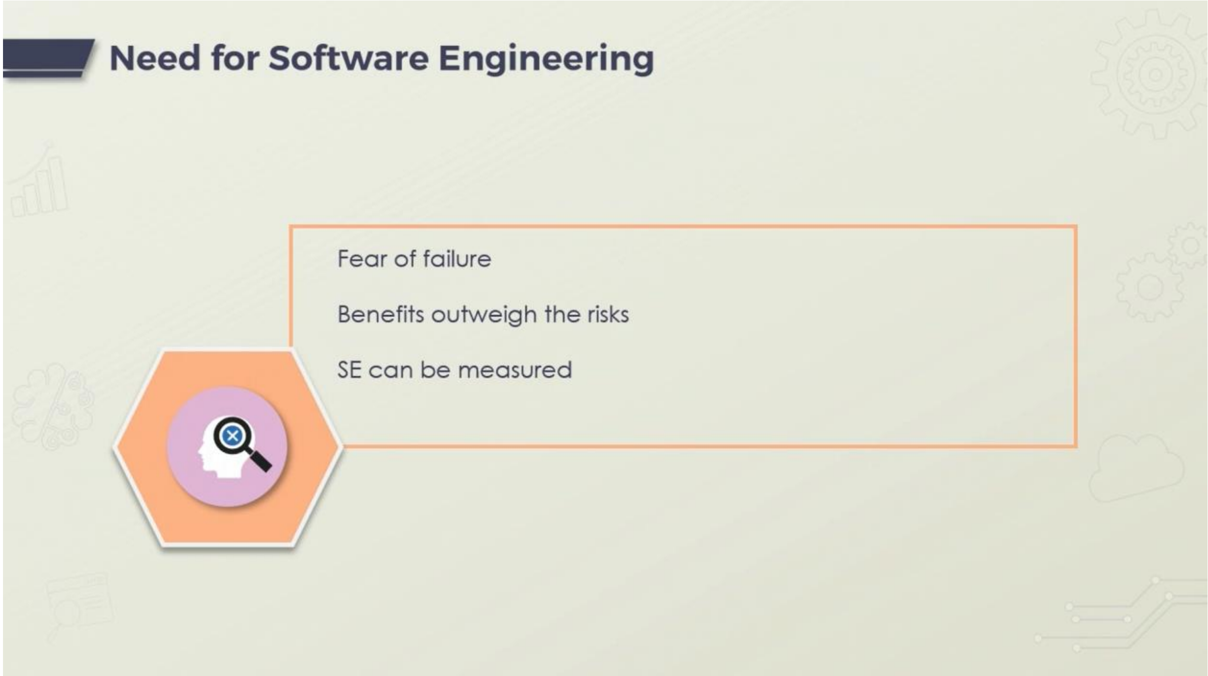
Software Engineering as a concepts

SE



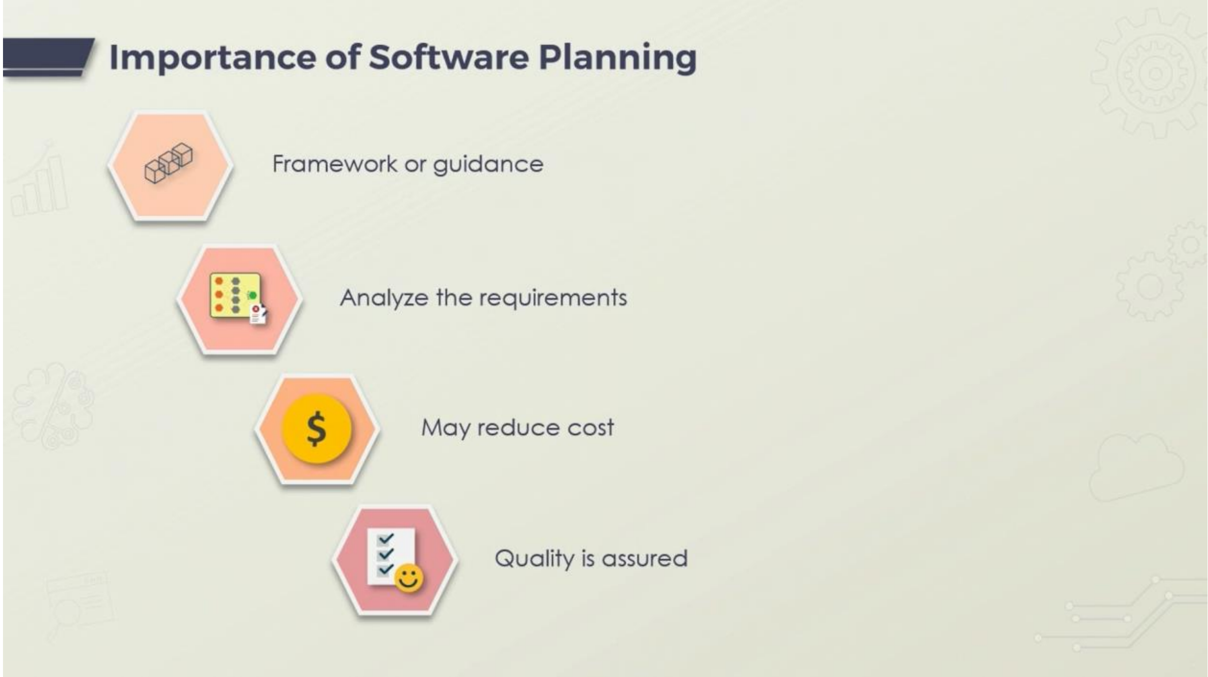
Software Engineering (SE)

A systematic approach to designing, building, testing, deploying, and maintaining software



Need for Software Engineering

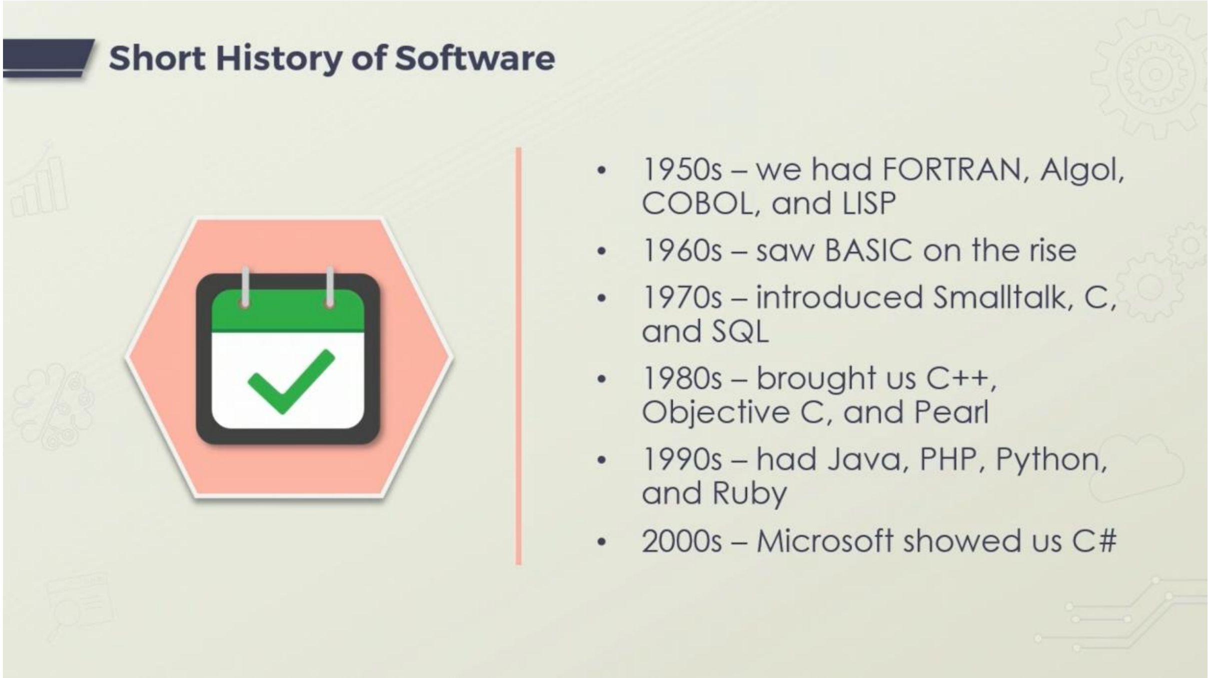
- Fear of failure
- Benefits outweigh the risks
- SE can be measured



Importance of Software Planning

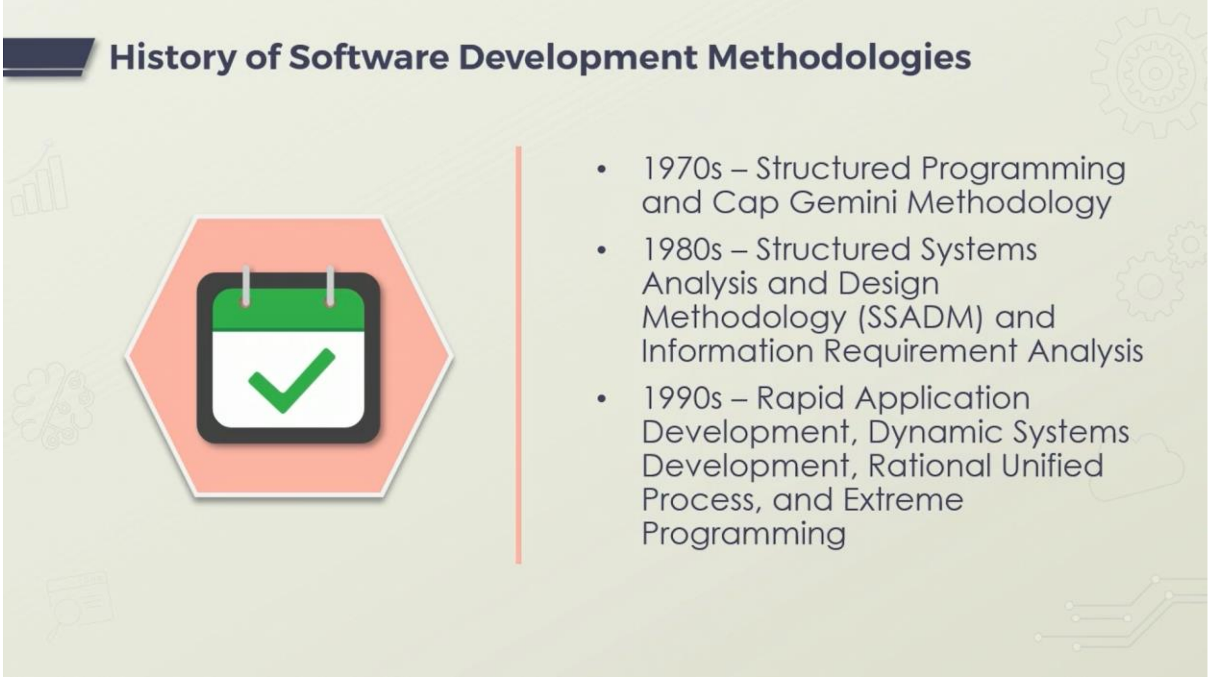
- Framework or guidance
- Analyze the requirements
- May reduce cost
- Quality is assured

Evolution of software and methodologies



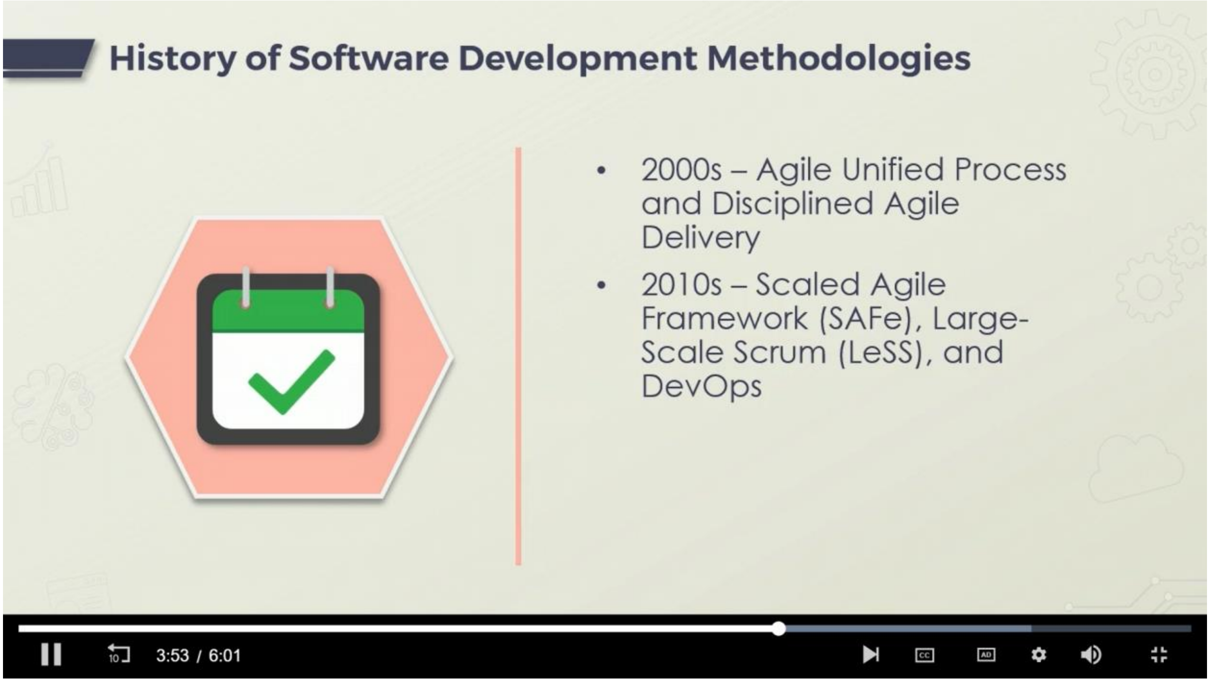
Short History of Software

- 1950s – we had FORTRAN, Algol, COBOL, and LISP
- 1960s – saw BASIC on the rise
- 1970s – introduced Smalltalk, C, and SQL
- 1980s – brought us C++, Objective C, and Pearl
- 1990s – had Java, PHP, Python, and Ruby
- 2000s – Microsoft showed us C#



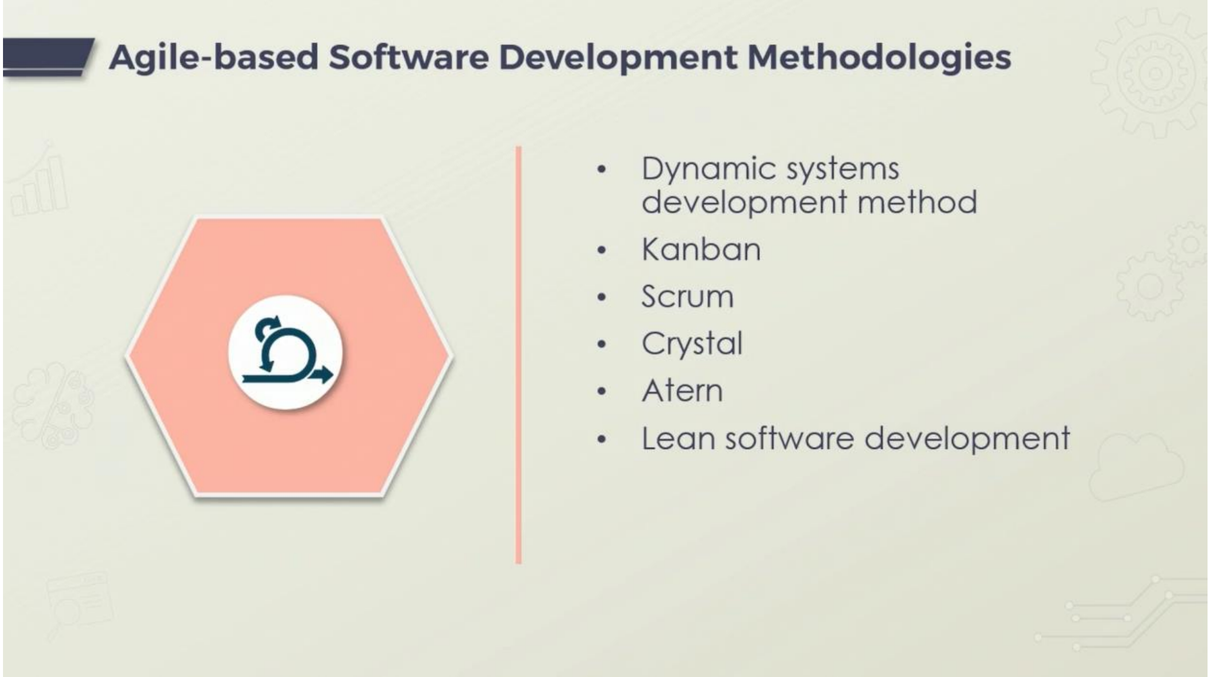
History of Software Development Methodologies

- 1970s – Structured Programming and Cap Gemini Methodology
- 1980s – Structured Systems Analysis and Design Methodology (SSADM) and Information Requirement Analysis
- 1990s – Rapid Application Development, Dynamic Systems Development, Rational Unified Process, and Extreme Programming



History of Software Development Methodologies

- 2000s – Agile Unified Process and Disciplined Agile Delivery
- 2010s – Scaled Agile Framework (SAFe), Large-Scale Scrum (LeSS), and DevOps

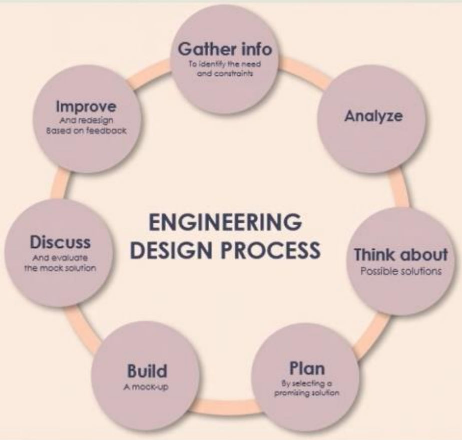


Agile-based Software Development Methodologies

- Dynamic systems development method
- Kanban
- Scrum
- Crystal
- Atern
- Lean software development

Software Development Life cycle sdlc phases

The Design Process – Generally



ENGINEERING DESIGN PROCESS

Gather info
To identify the need and constraints

Analyze

Think about
Possible solutions

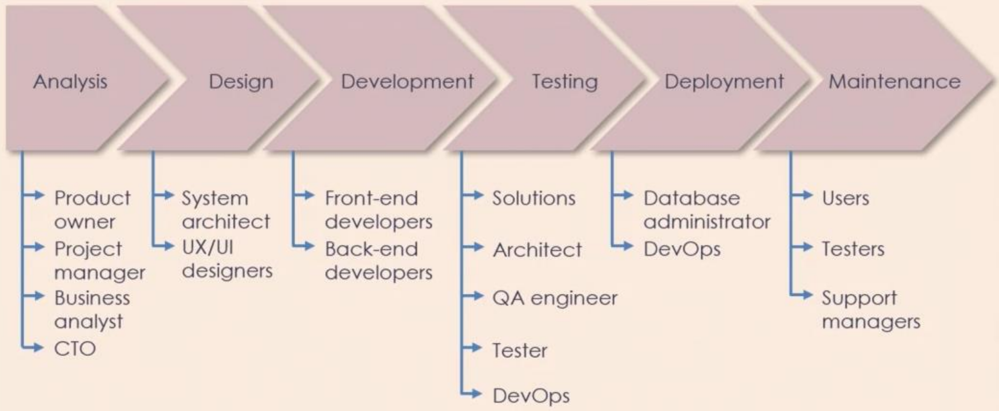
Plan
By selecting a promising solution

Build
A mock-up

Discuss
And evaluate the mock solution

Improve
And redesign Based on feedback

Phases of the Software Development Life Cycle (SDLC)



Analysis
- Product owner
- Project manager
- Business analyst
- CTO

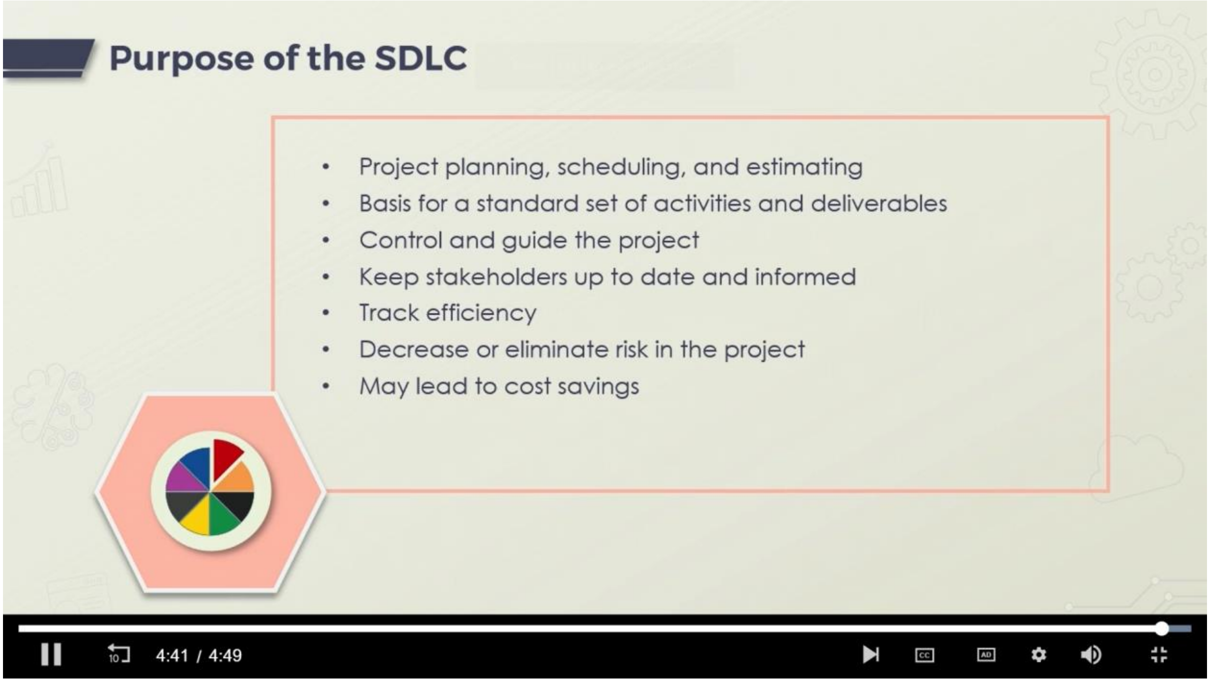
Design
- System architect
- UX/UI designers

Development
- Front-end developers
- Back-end developers

Testing
- Solutions
- Architect
- QA engineer
- Tester
- DevOps

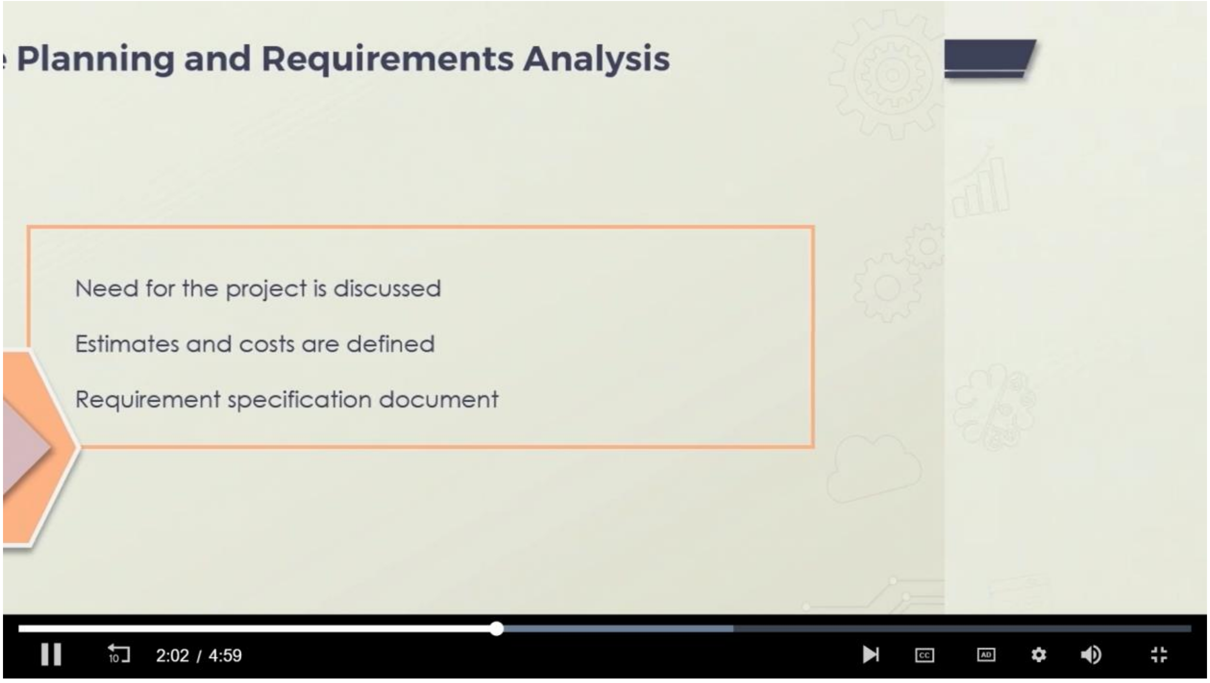
Deployment
- Database administrator
- DevOps

Maintenance
- Users
- Testers
- Support managers



Purpose of the SDLC

- Project planning, scheduling, and estimating
- Basis for a standard set of activities and deliverables
- Control and guide the project
- Keep stakeholders up to date and informed
- Track efficiency
- Decrease or eliminate risk in the project
- May lead to cost savings

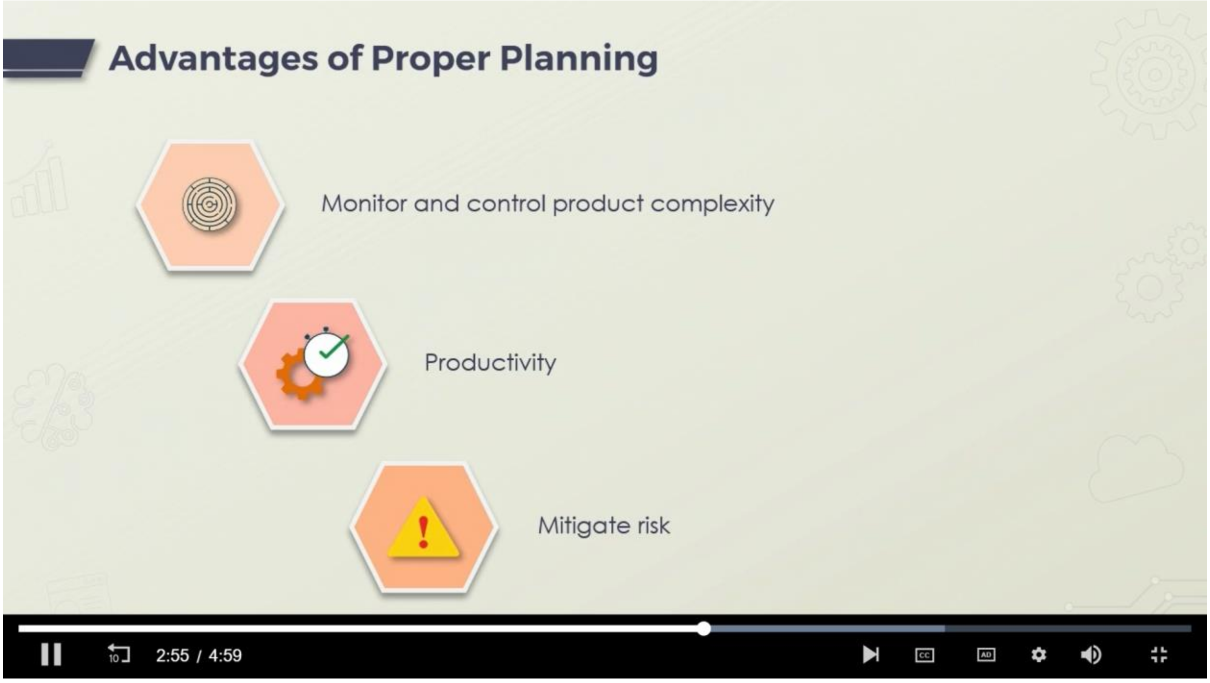


Planning and Requirements Analysis

Need for the project is discussed

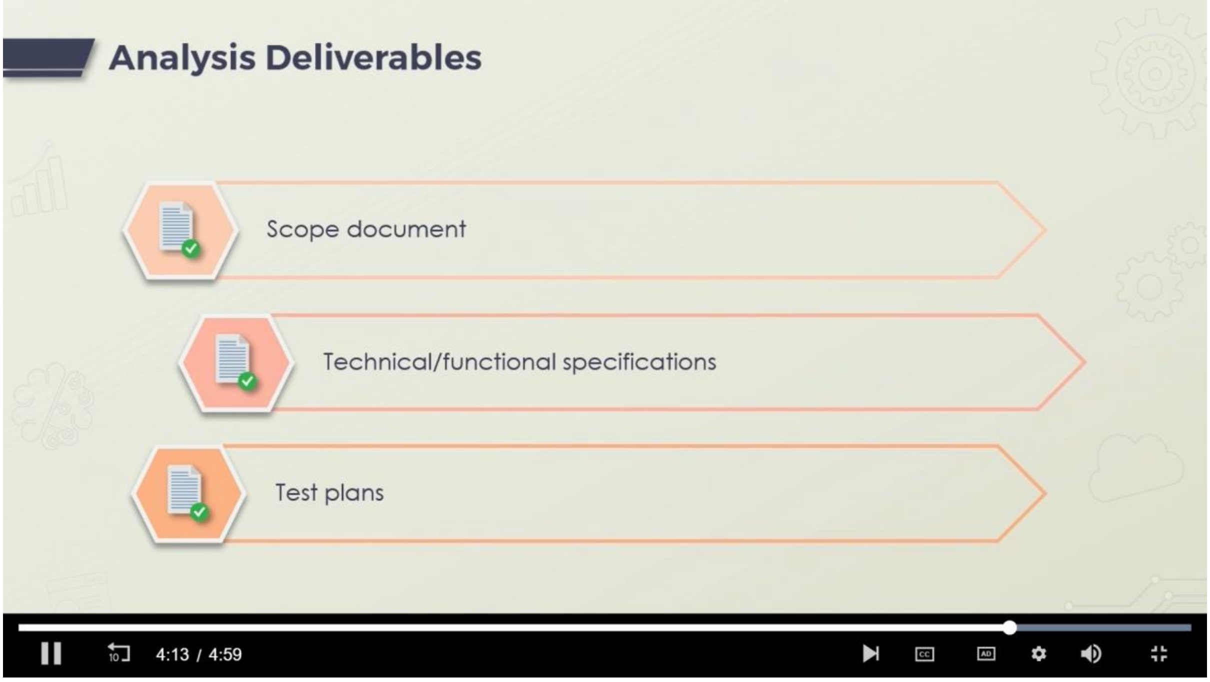
Estimates and costs are defined

Requirement specification document



Advantages of Proper Planning

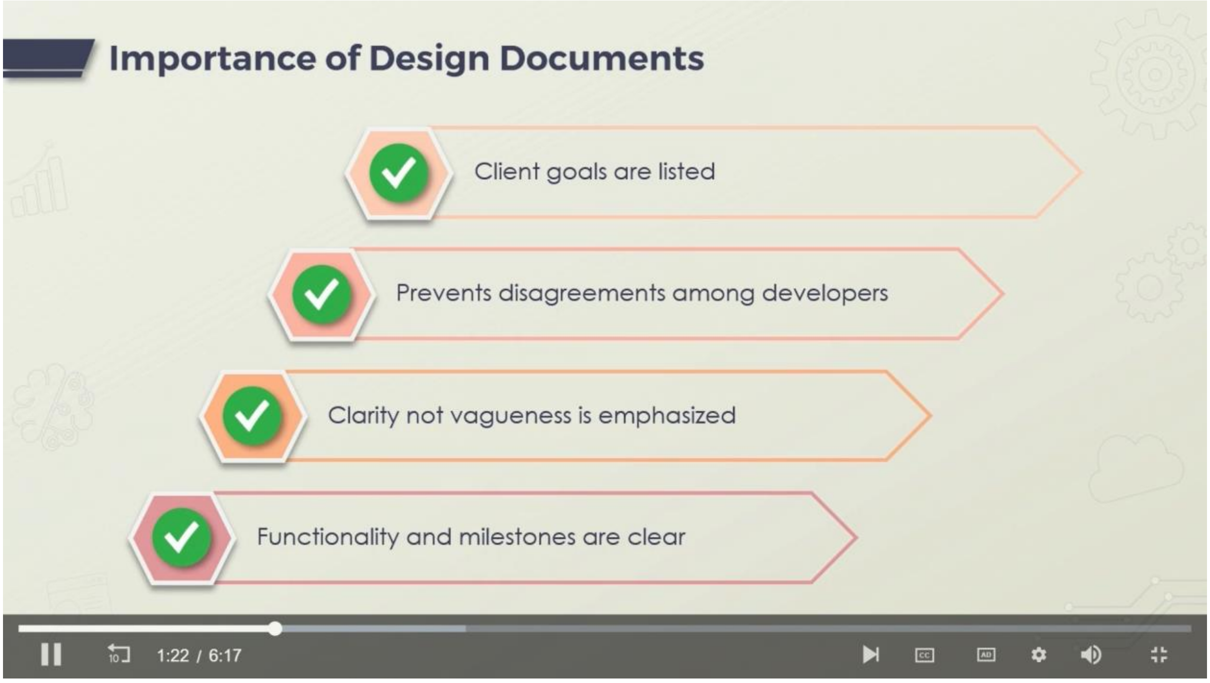
- Monitor and control product complexity
- Productivity
- Mitigate risk



Analysis Deliverables

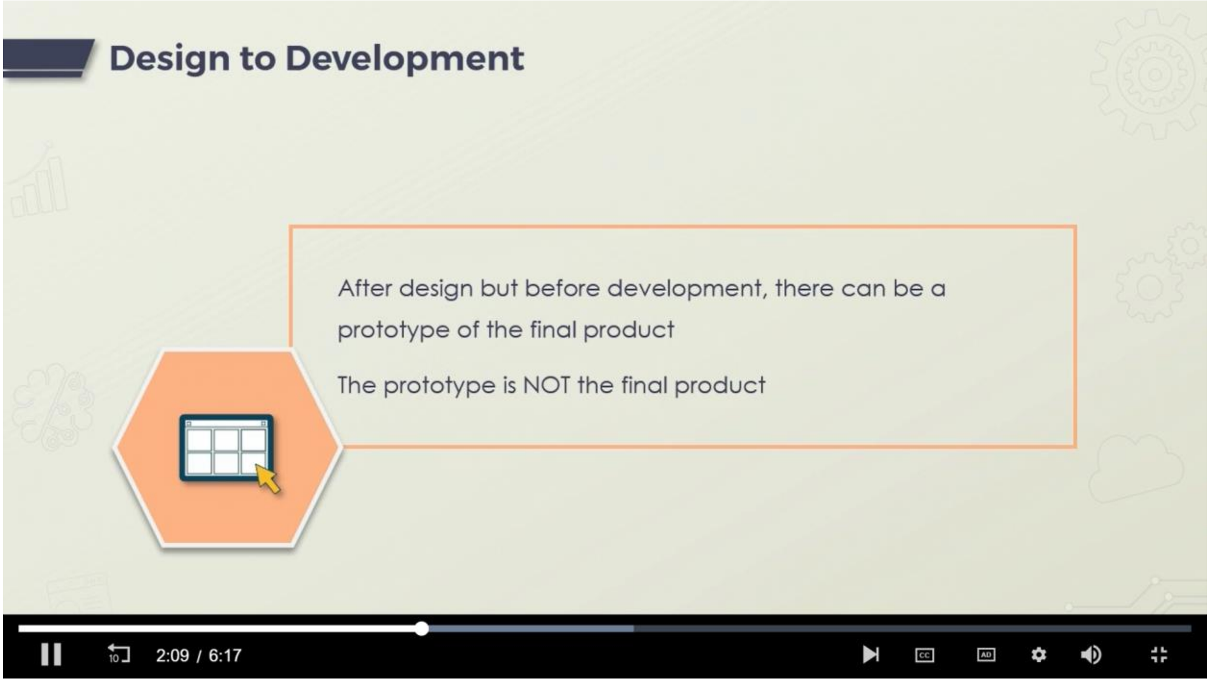
- Scope document
- Technical/functional specifications
- Test plans

Software Design and prototyping



Importance of Design Documents

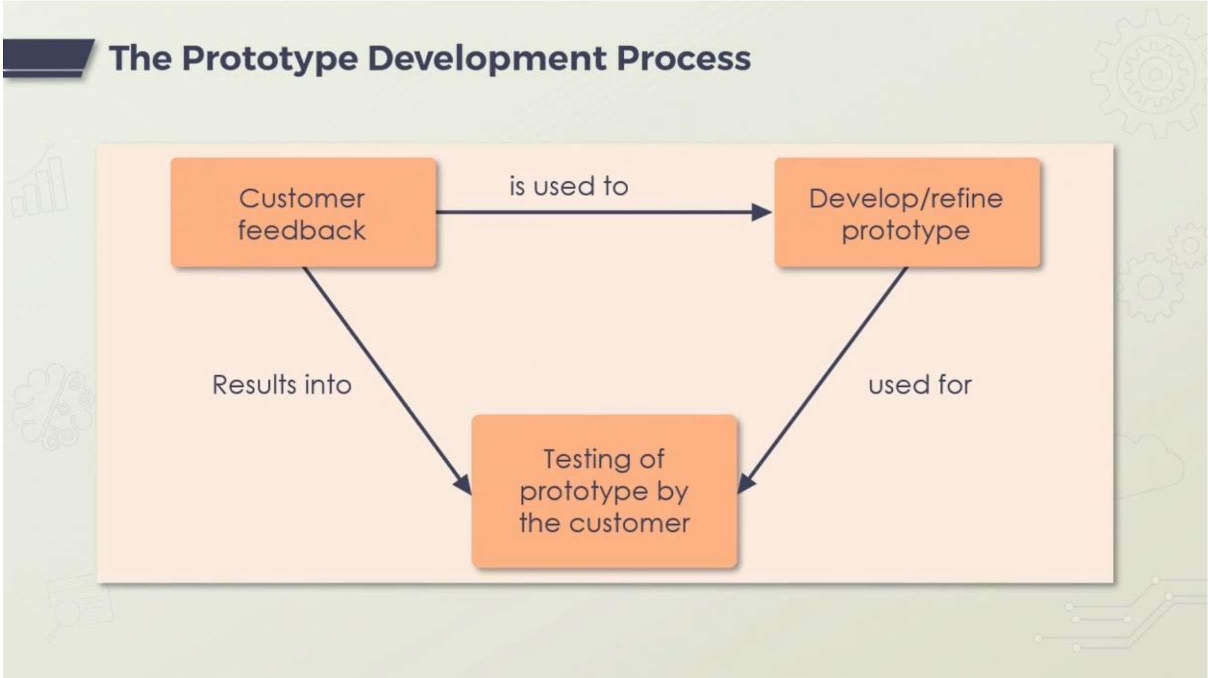
- ✓ Client goals are listed
- ✓ Prevents disagreements among developers
- ✓ Clarity not vagueness is emphasized
- ✓ Functionality and milestones are clear



Design to Development

After design but before development, there can be a prototype of the final product

The prototype is NOT the final product



The Prototype Development Process

Customer feedback

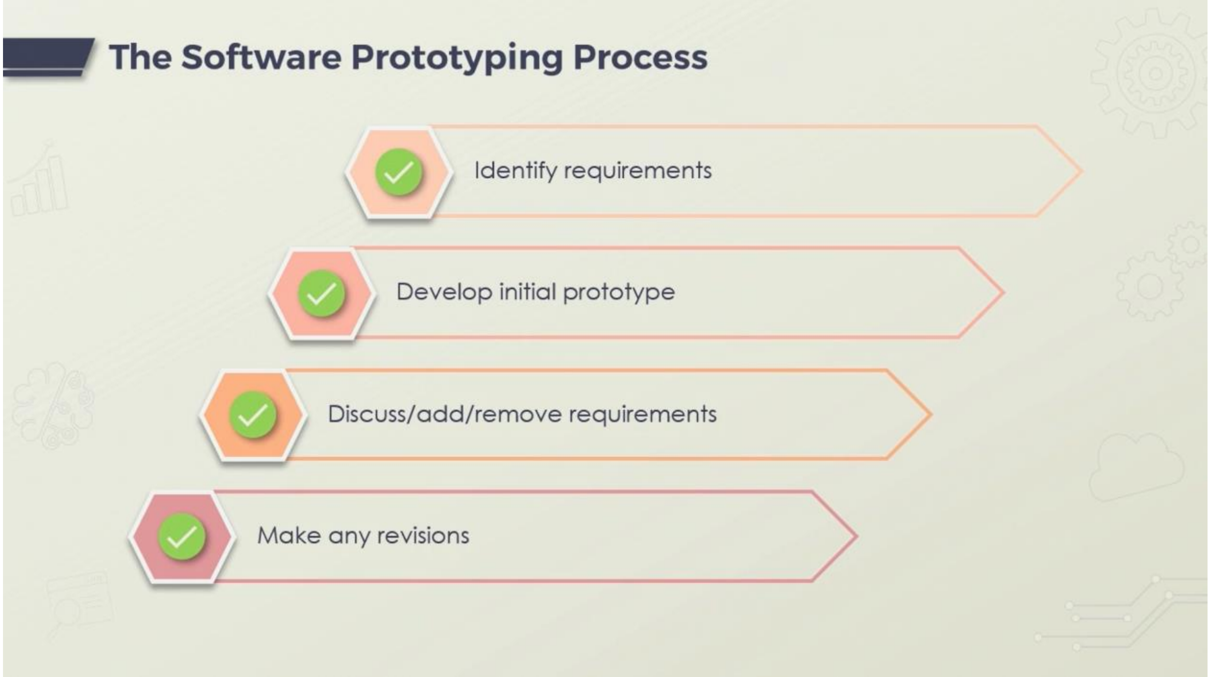
is used to

Develop/refine prototype

Results into

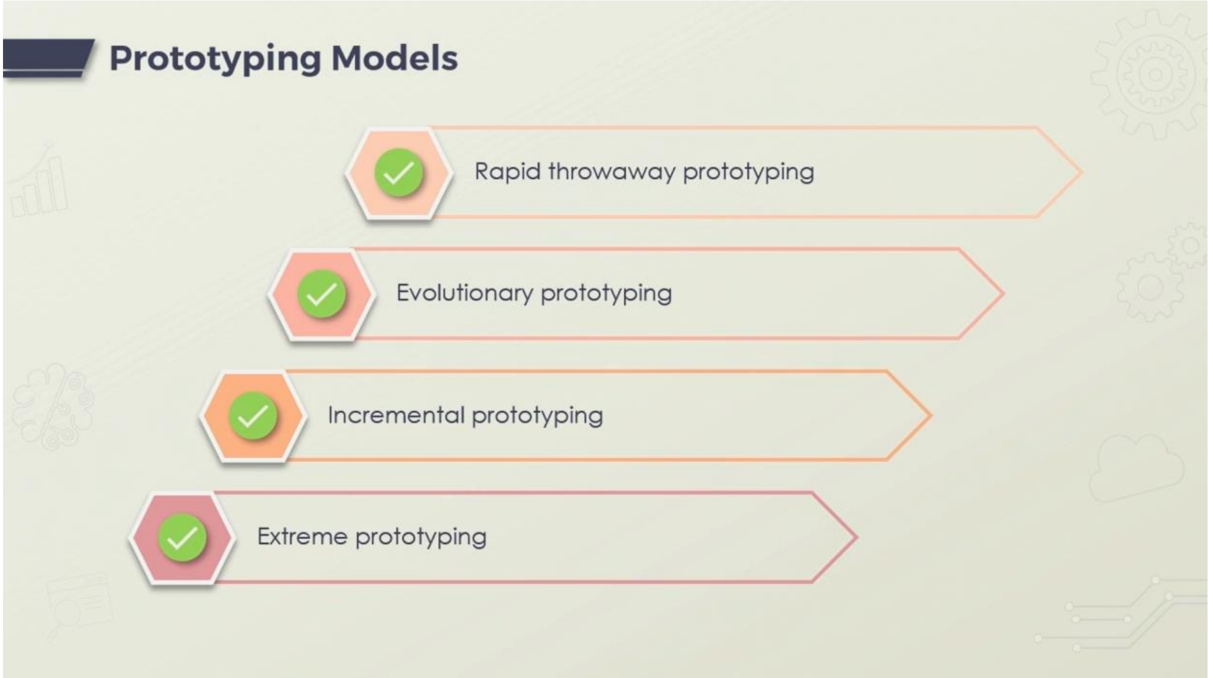
used for

Testing of prototype by the customer



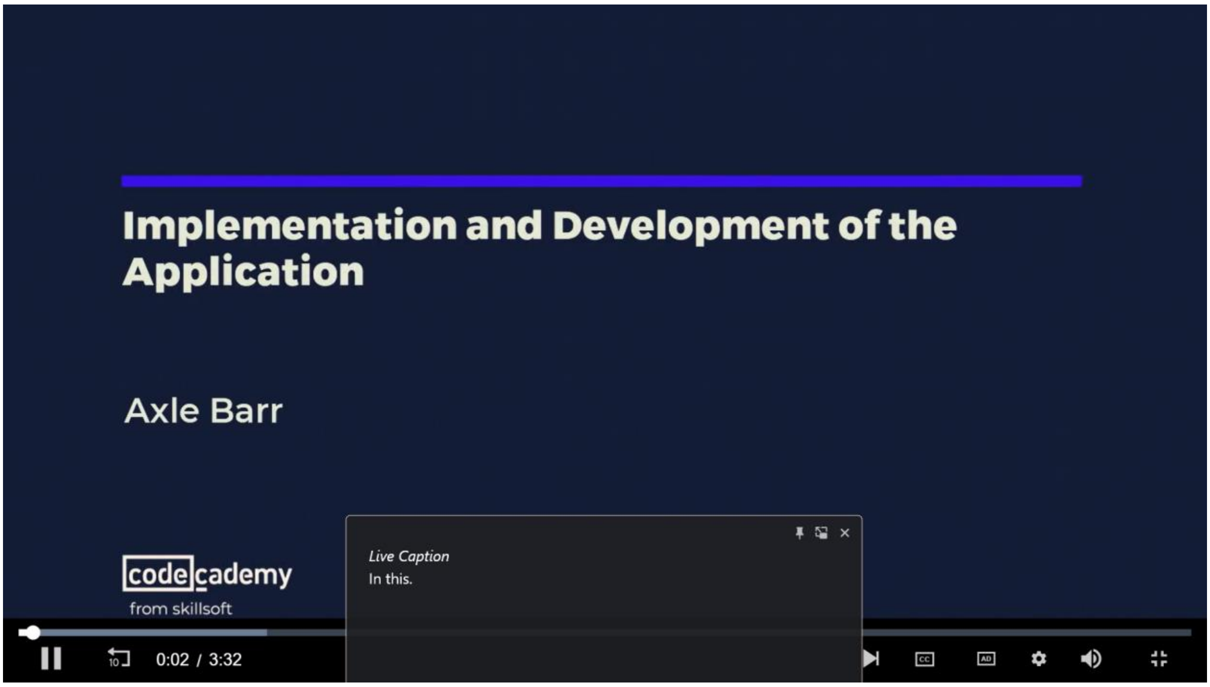
The Software Prototyping Process

- ✓ Identify requirements
- ✓ Develop initial prototype
- ✓ Discuss/add/remove requirements
- ✓ Make any revisions



Prototyping Models

- ✓ Rapid throwaway prototyping
- ✓ Evolutionary prototyping
- ✓ Incremental prototyping
- ✓ Extreme prototyping

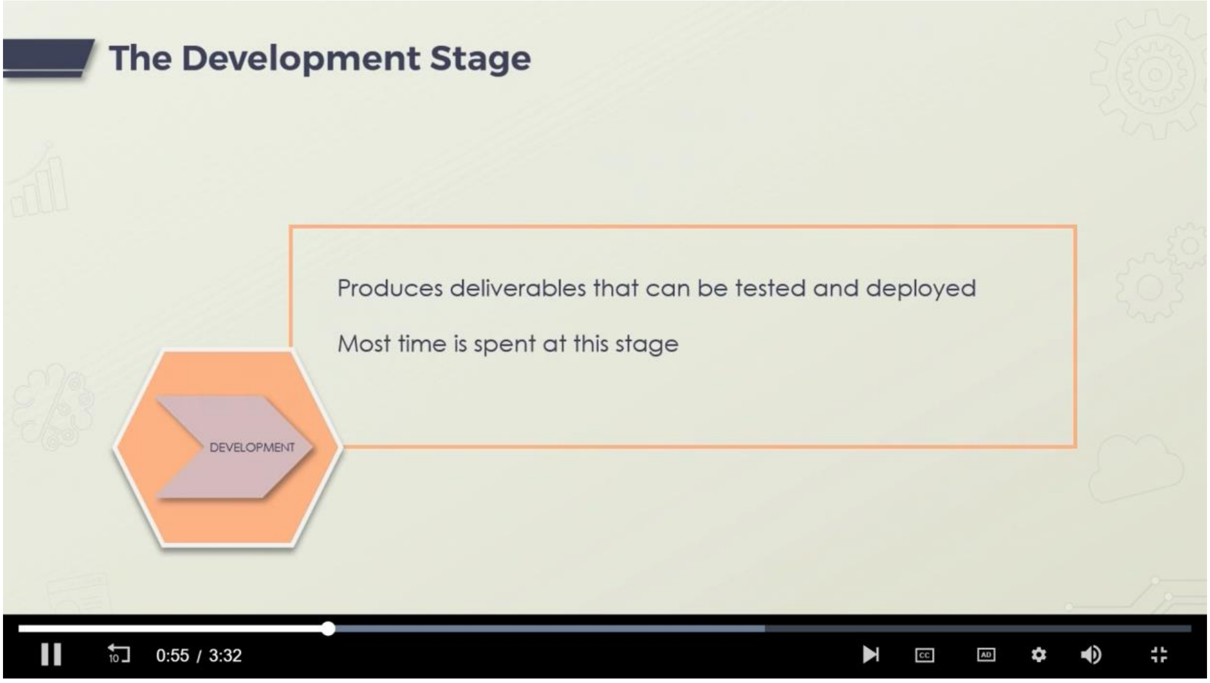


Implementation and Development of the Application

Axle Barr

Live Caption

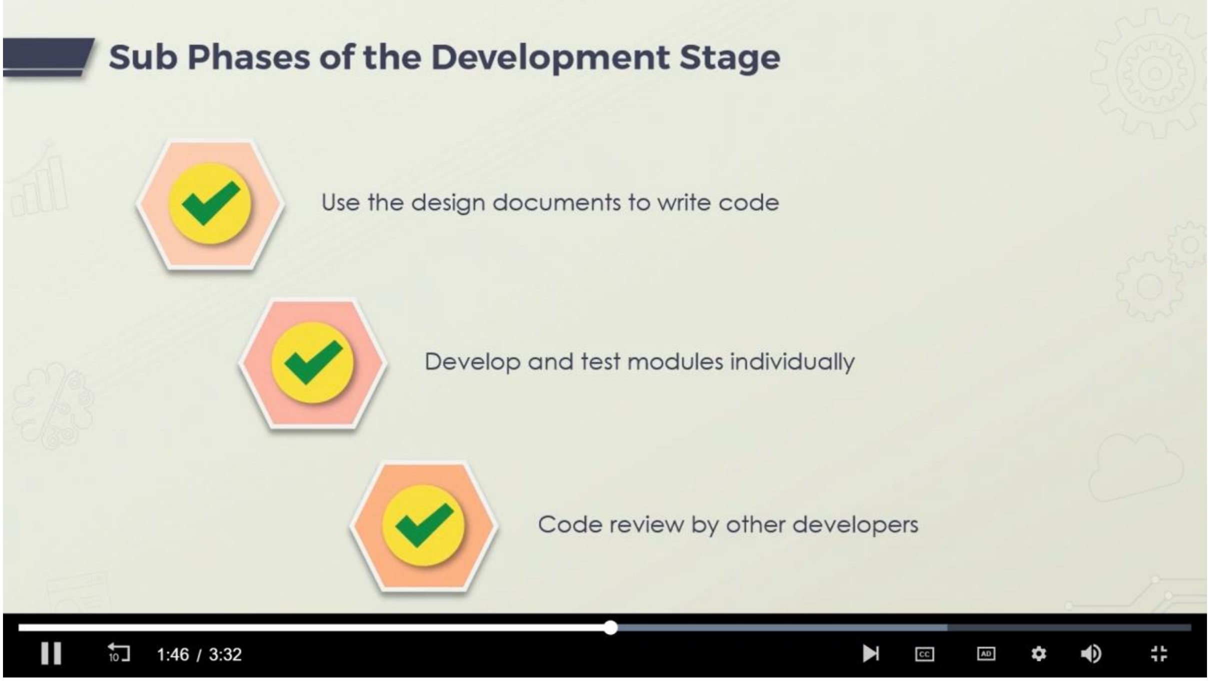
In this.



The Development Stage

Produces deliverables that can be tested and deployed

Most time is spent at this stage

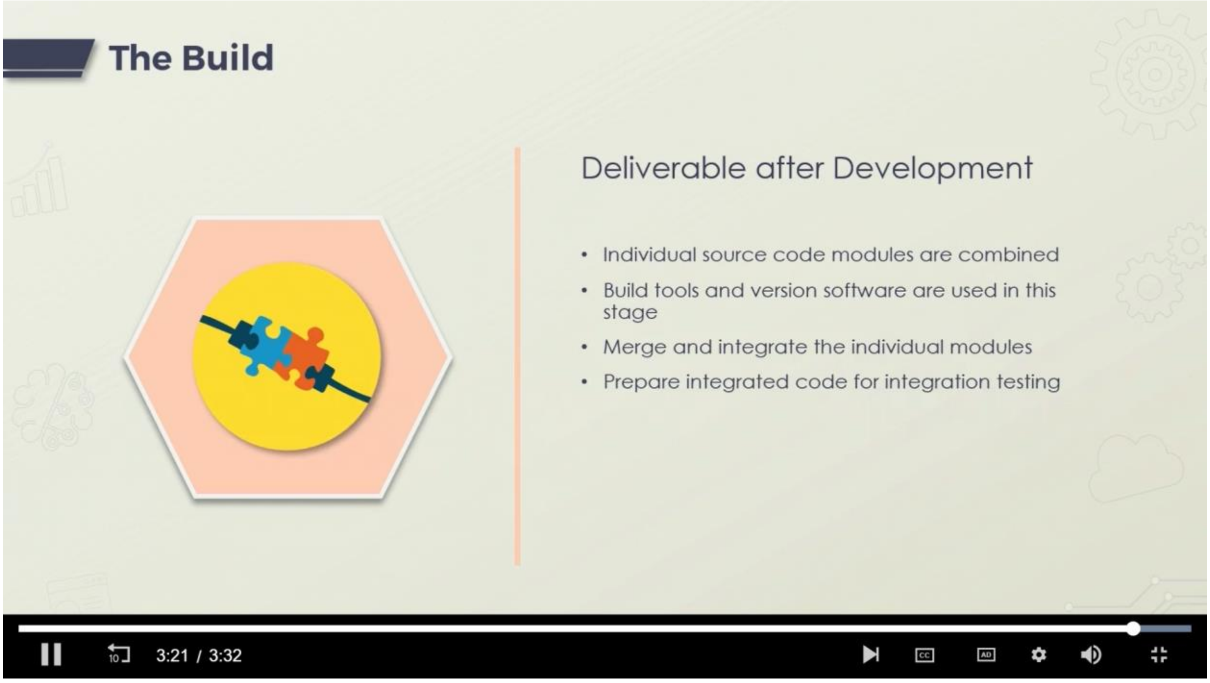


Sub Phases of the Development Stage

✓ Use the design documents to write code

✓ Develop and test modules individually

✓ Code review by other developers

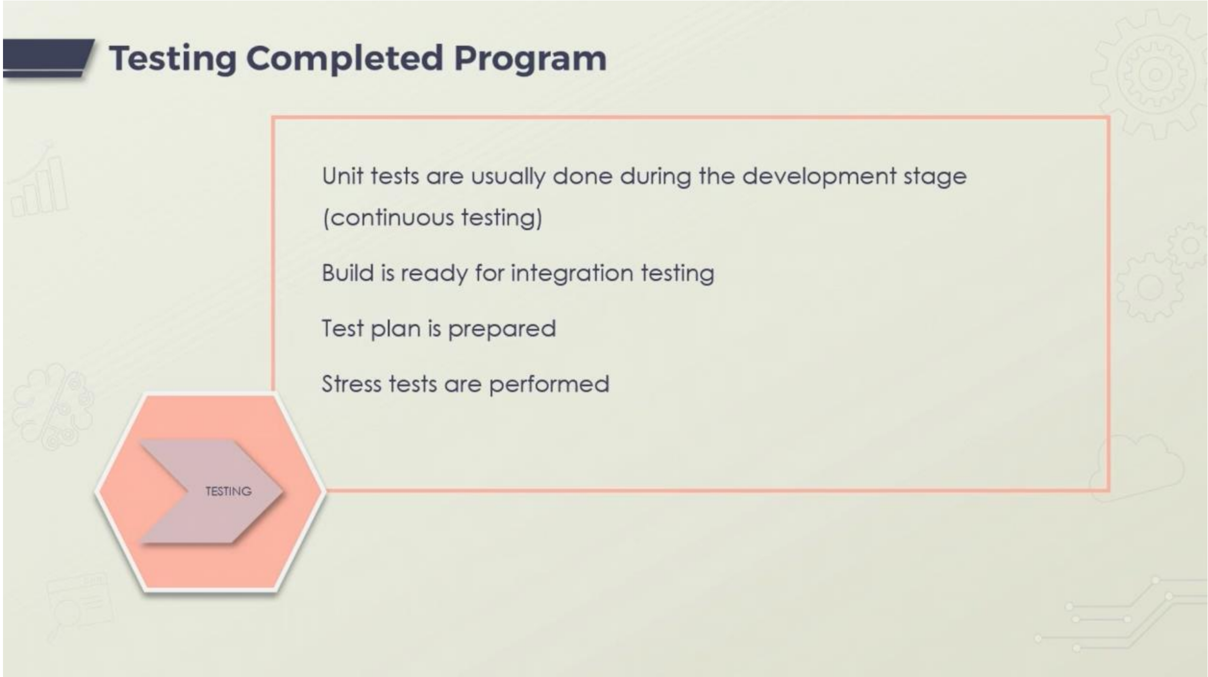


The Build

Deliverable after Development

- Individual source code modules are combined
- Build tools and version software are used in this stage
- Merge and integrate the individual modules
- Prepare integrated code for integration testing

Software testing and deployment



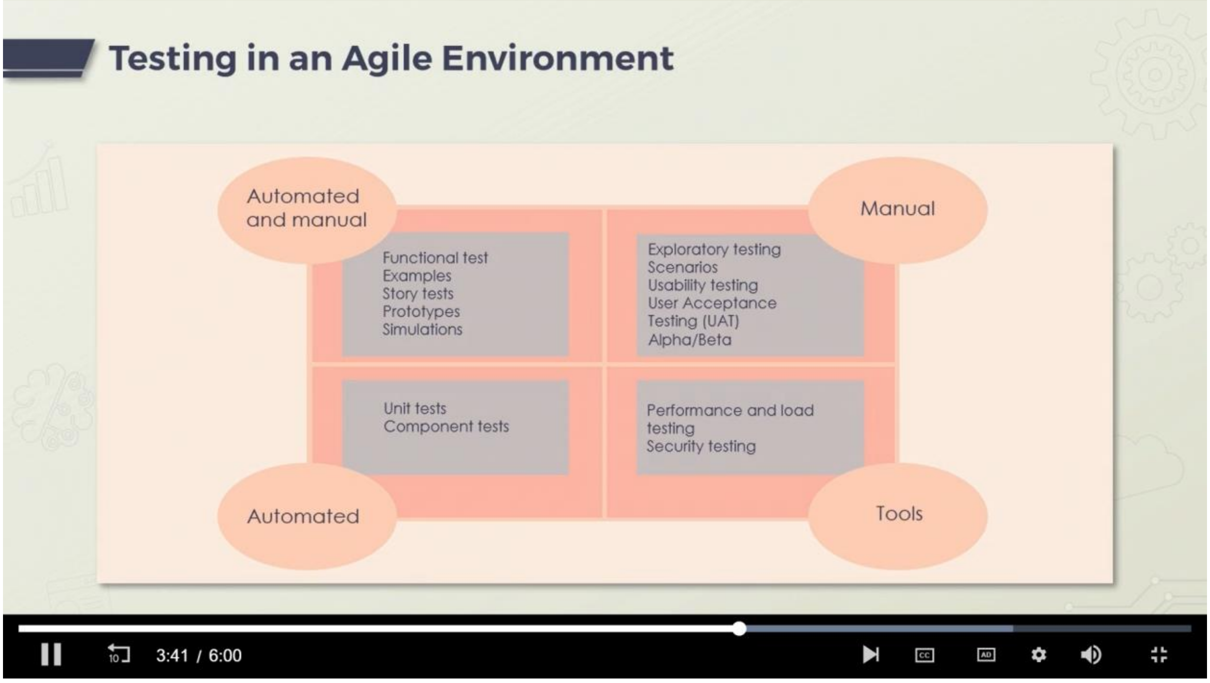
Testing Completed Program

Unit tests are usually done during the development stage (continuous testing)

Build is ready for integration testing

Test plan is prepared

Stress tests are performed



Testing in an Agile Environment

Automated and manual

Functional test
Examples
Story tests
Prototypes
Simulations

Manual

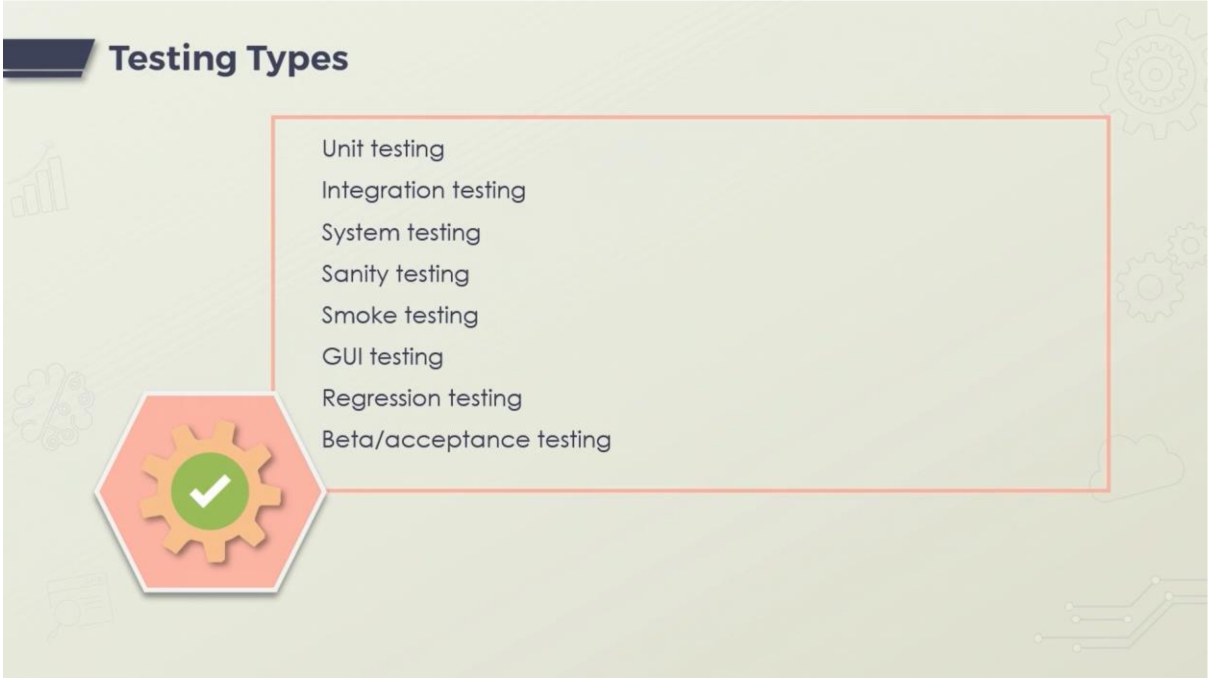
Exploratory testing
Scenarios
Usability testing
User Acceptance Testing (UAT)
Alpha/Beta

Unit tests
Component tests

Automated

Performance and load testing
Security testing

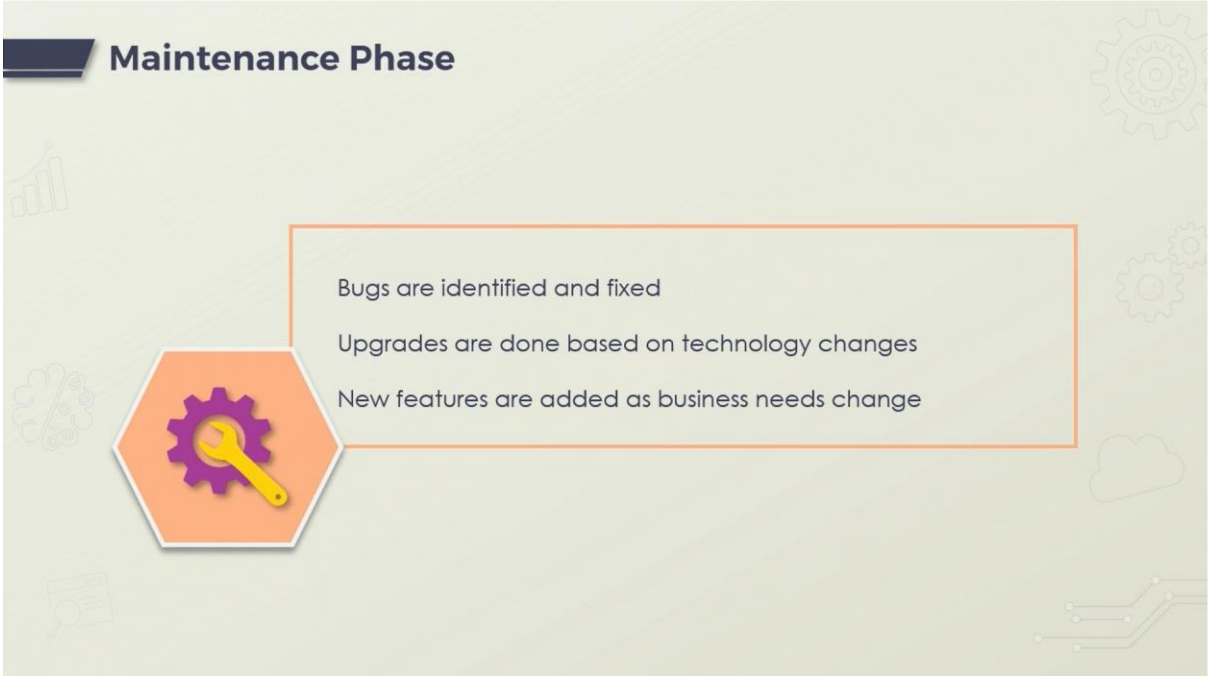
Tools



Testing Types

- Unit testing
- Integration testing
- System testing
- Sanity testing
- Smoke testing
- GUI testing
- Regression testing
- Beta/acceptance testing

Software Maintenance

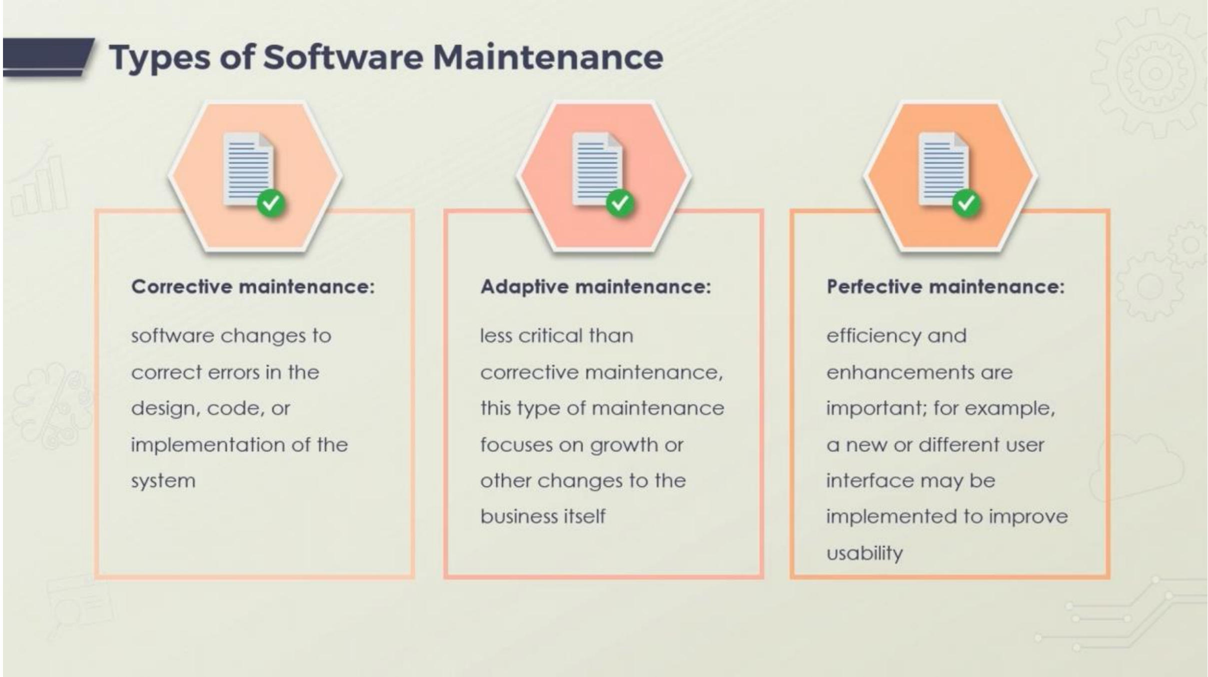


Maintenance Phase

Bugs are identified and fixed

Upgrades are done based on technology changes

New features are added as business needs change



Types of Software Maintenance

Corrective maintenance:

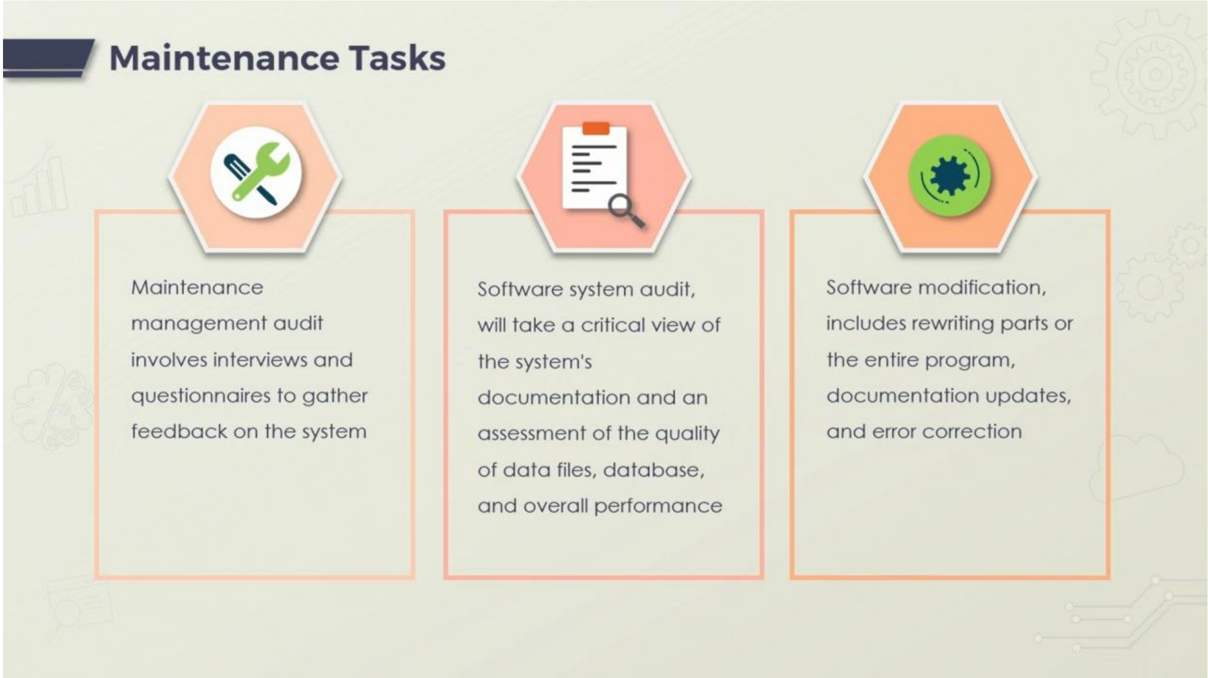
software changes to correct errors in the design, code, or implementation of the system

Adaptive maintenance:

less critical than corrective maintenance, this type of maintenance focuses on growth or other changes to the business itself

Perfective maintenance:

efficiency and enhancements are important; for example, a new or different user interface may be implemented to improve usability



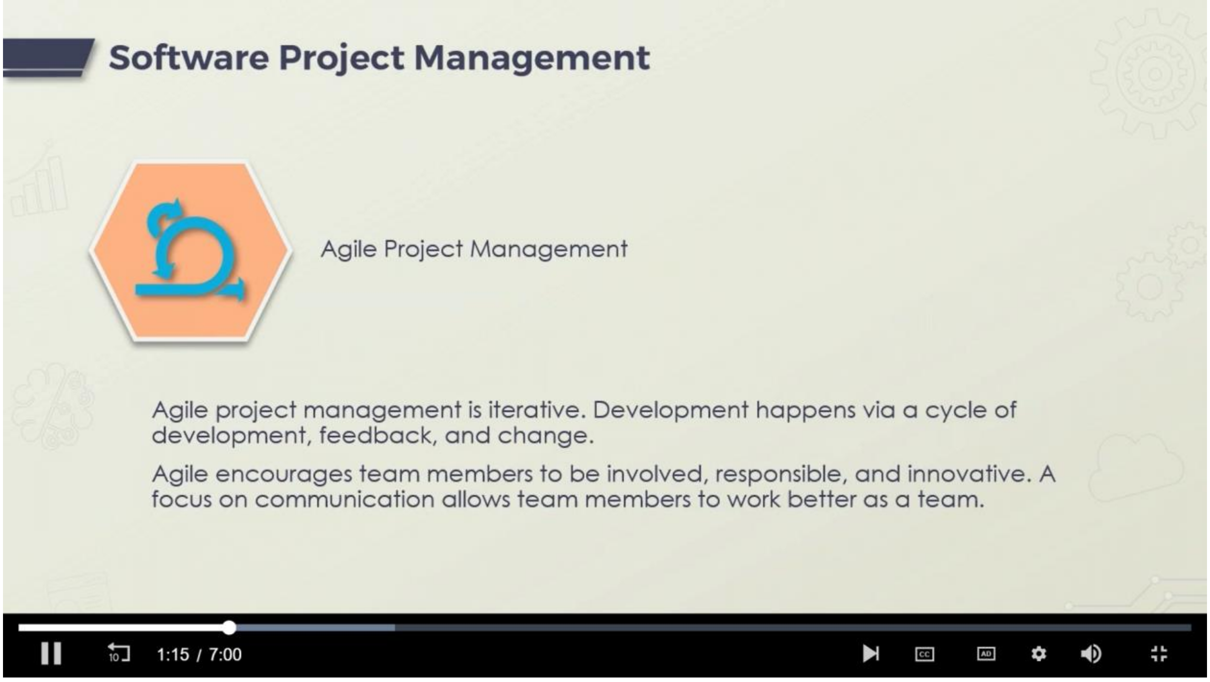
Maintenance Tasks

Maintenance management audit involves interviews and questionnaires to gather feedback on the system

Software system audit, will take a critical view of the system's documentation and an assessment of the quality of data files, database, and overall performance

Software modification, includes rewriting parts or the entire program, documentation updates, and error correction

Softwareq Project management

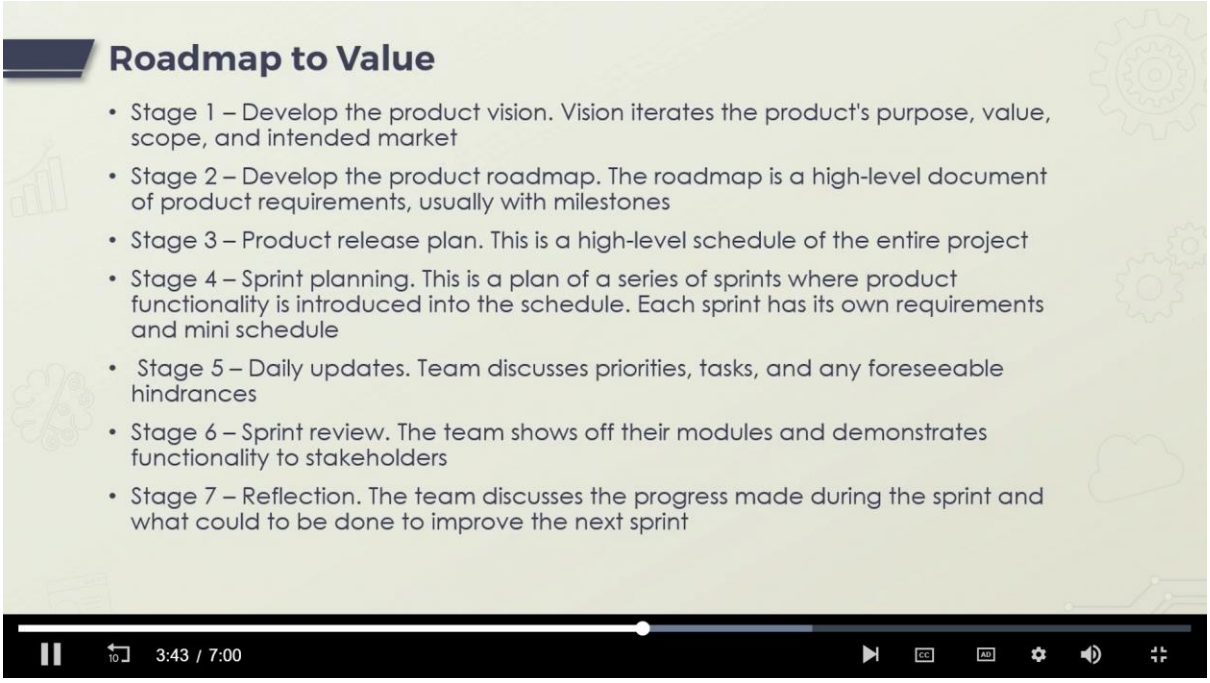


Software Project Management

Agile Project Management

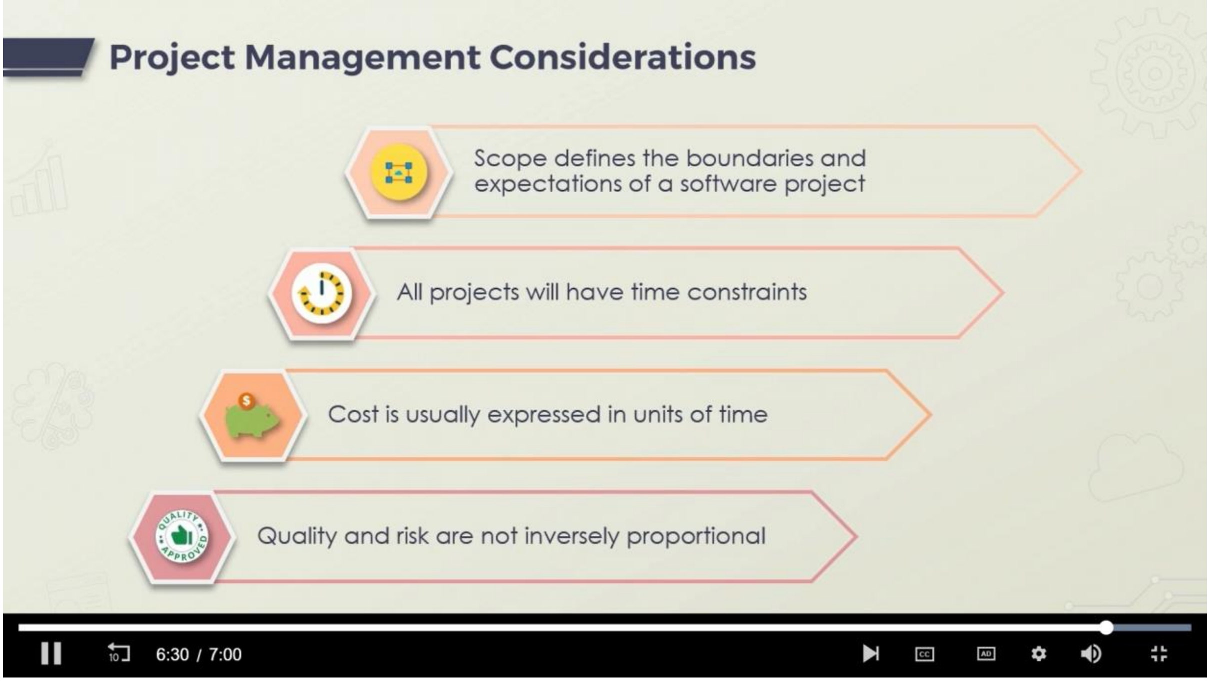
Agile project management is iterative. Development happens via a cycle of development, feedback, and change.

Agile encourages team members to be involved, responsible, and innovative. A focus on communication allows team members to work better as a team.



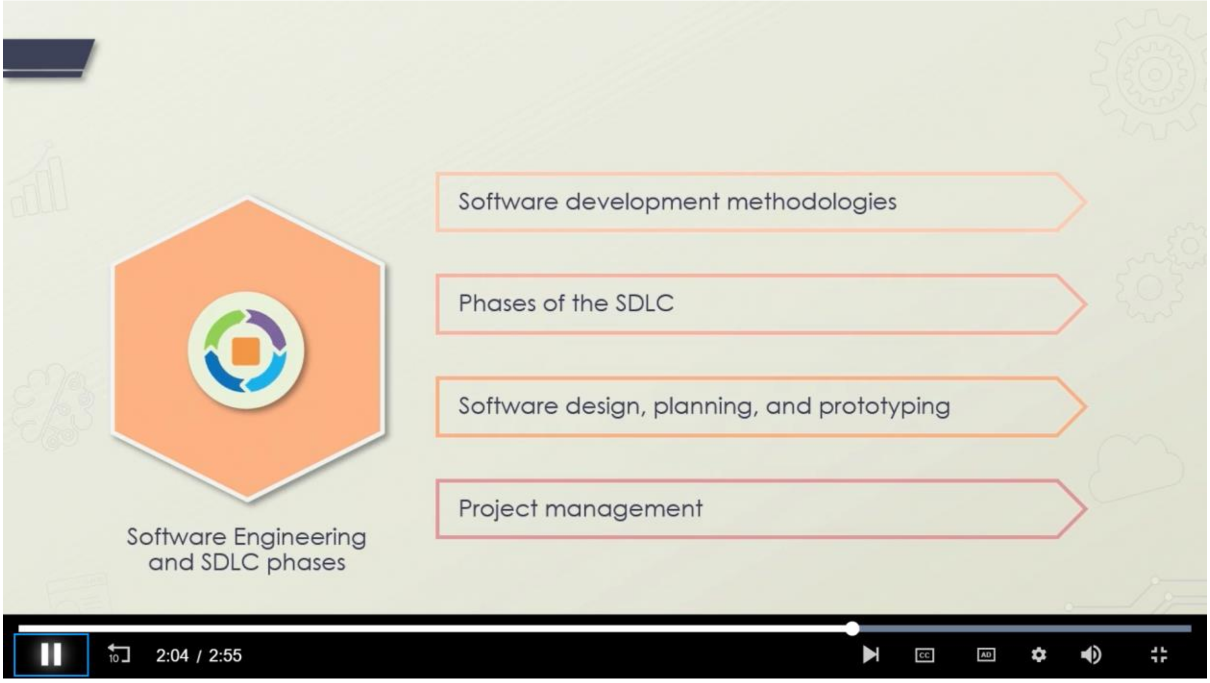
Roadmap to Value

- Stage 1 – Develop the product vision. Vision iterates the product's purpose, value, scope, and intended market
- Stage 2 – Develop the product roadmap. The roadmap is a high-level document of product requirements, usually with milestones
- Stage 3 – Product release plan. This is a high-level schedule of the entire project
- Stage 4 – Sprint planning. This is a plan of a series of sprints where product functionality is introduced into the schedule. Each sprint has its own requirements and mini schedule
- Stage 5 – Daily updates. Team discusses priorities, tasks, and any foreseeable hindrances
- Stage 6 – Sprint review. The team shows off their modules and demonstrates functionality to stakeholders
- Stage 7 – Reflection. The team discusses the progress made during the sprint and what could to be done to improve the next sprint



Project Management Considerations

- Scope defines the boundaries and expectations of a software project
- All projects will have time constraints
- Cost is usually expressed in units of time
- Quality and risk are not inversely proportional



Software Engineering and SDLC phases

- Software development methodologies
- Phases of the SDLC
- Software design, planning, and prototyping
- Project management