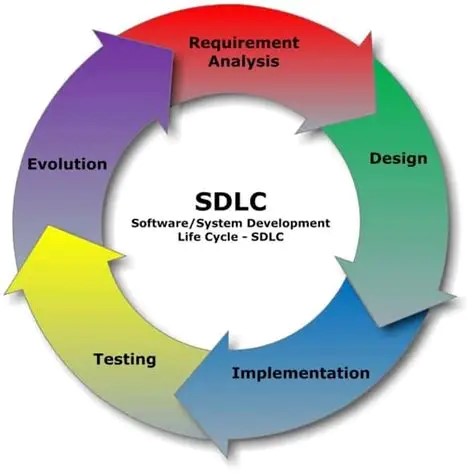
**PRACTICAL 1 :– study the complete system development life cycle (sdlc) and analysis various activity conducted as a part of various phases for each (sdlc) phase identify the objectives and summarizes outcomes choose appropriate software life cycle appropriate.**

Software Development Life Cycle (SDLC) is structured process that is used to design, develop and test good-quality software. SDLC, or software development life cycle, is methodology that defines the entire procedure of software development step-by-step.

**PHASES OF SDLC**



## ****1. Requirement Gathering & Analysis****

### Objectives:

* Understand what the user wants.
* Identify functional and non-functional requirements.
* Assess feasibility.

### Activities:

* Stakeholder interviews
* Brainstorming sessions
* Document analysis
* Feasibility studies (technical, economic, legal, operational)

### Outcomes:

* Software Requirements Specification (SRS) document
* Feasibility report
* Project charter

## ****2. System Design****

### Objectives:

* Translate requirements into a blueprint for development.
* Design system architecture and components.

### Activities:

* High-level design (HLD)
* Low-level design (LLD)
* Define databases, APIs, data flows, UI/UX mockups

### Outcomes:

* Design Documents (HLD & LLD)
* Data Flow Diagrams (DFDs)
* System architecture diagrams
* UI prototypes

## ****3. Implementation (Coding)****

### Objectives:

* Develop the actual software based on design.
* Ensure code quality and adherence to design.

### Activities:

* Writing source code
* Code reviews
* Unit testing
* Integration of modules

### Outcomes:

* Source code repository
* Executable software
* Unit test results

## ****4. Testing****

### Objectives:

* Ensure software meets requirements.
* Identify and fix defects.

### Activities:

* Test planning and design
* Functional, integration, system, and regression testing
* Bug tracking and fixing

### Outcomes:

* Test cases and test reports
* Bug reports
* Validated software ready for deployment

## ****5. Deployment****

### Objectives:

* Release the software to users.
* Ensure installation and setup are correct.

### Activities:

* Deployment planning
* Installation on production systems
* User training and support

### Outcomes:

* Deployed software
* Installation guide
* Training manuals

## ****6. Maintenance****

### Objectives:

* Keep the software running efficiently.
* Fix issues and accommodate future enhancements.

### Activities:

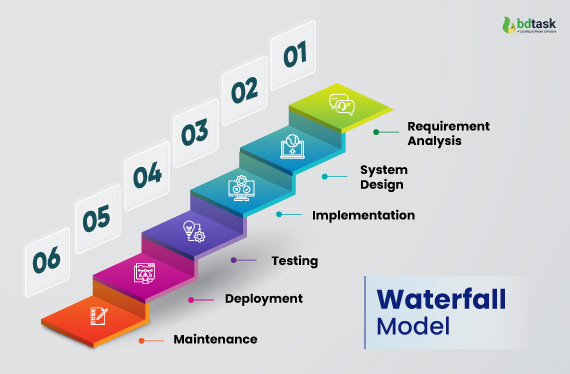
* Issue tracking
* Performance monitoring
* Software updates and patches

### Outcomes:

* Updated software versions
* Maintenance logs
* Enhanced features

**TYPES OF SDLC**

**1.Waterfall Model:**

****

**Definition:**

A linear sequential approach to software development where progress flows through distinct, non-overlapping phases from conception to deployment.

**Phases:**

1. **Requirements Analysis**: Capture and document complete project requirements
2. **System Design**: Create comprehensive system architecture and design specifications
3. **Implementation**: Develop software based on design specifications
4. **Testing**: Verify and validate system functionality
5. **Deployment**: Release the final product to the end-users
6. **Maintenance**: Provide ongoing support and system updates

**Advantages:**

* Clear, structured approach
* Easy to understand and manage

**Disadvantages:**

* Inflexible to changes
* High risk due to late testing

**2. Spiral Model:**

****

**Definition:**

A risk-driven software development process that combines iterative development with systematic aspects of the Waterfall Model, emphasizing risk analysis.

**Phases:**

1. **Planning**: Determine project objectives and constraints
2. **Risk Analysis**: Identify and evaluate potential project risks
3. **Engineering**: Develop product through iterative cycles
4. **Evaluation**: Review and get customer feedback

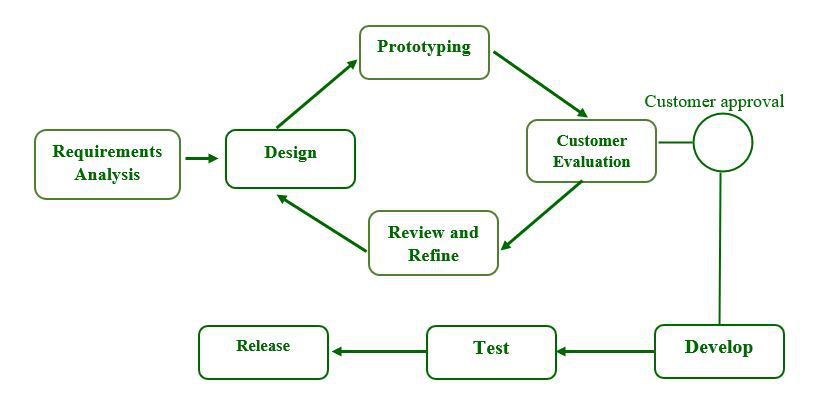
**Advantages:**

* Excellent for large, complex projects
* Continuous risk assessment

**Disadvantages:**

* Complex and expensive
* Requires extensive risk management expertise

**3. Prototype Model:**

****

**Definition:**

A software development approach focused on creating initial working models to understand and refine project requirements through iterative feedback.

**Phases:**

1. **Requirements Gathering**: Collect initial project requirements
2. **Quick Design**: Develop preliminary system prototype
3. **Prototype Development**: Build basic working model
4. **Customer Evaluation**: Gather feedback and refine requirements
5. **Refinement**: Improve prototype based on feedback

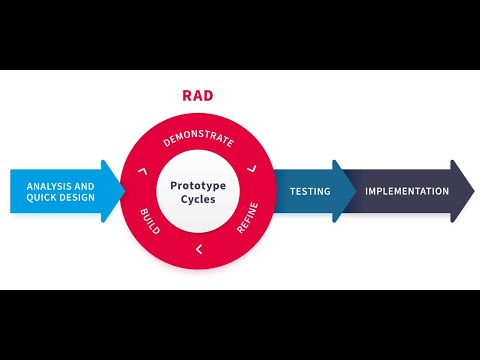
**Advantages:**

* Improved user involvement
* Better requirement understanding

**Disadvantages:**

* Potential for excessive iterations
* High development costs

1. **Rapid Application Development (RAD) Model:**

****

**Definition:**

A high-speed development approach focusing on rapid prototyping, iterative development, and minimal planning.

**Phases:**

1. **Business Modeling**: Define project scope and requirements
2. **Data Modeling**: Develop data structures and processes
3. **Process Modeling**: Create system workflow
4. **Application Generation**: Develop working application
5. **Testing and Turnover**: Validate and deploy system

**Advantages:**

* Fast development cycles
* Highly flexible

**Disadvantages:**

* Requires highly skilled developers
* Not suitable for large, complex projects

1. **Incremental Model:**

****

**Definition:**

A software development approach that divides the project into small, manageable modules developed and delivered incrementally.

**Phases:**

1. **Initial Planning**: Define overall system requirements
2. **Incremental Design**: Create design for initial and subsequent modules
3. **Incremental Implementation**: Develop modules sequentially
4. **Testing**: Validate each module
5. **Integration**: Combine and test complete system

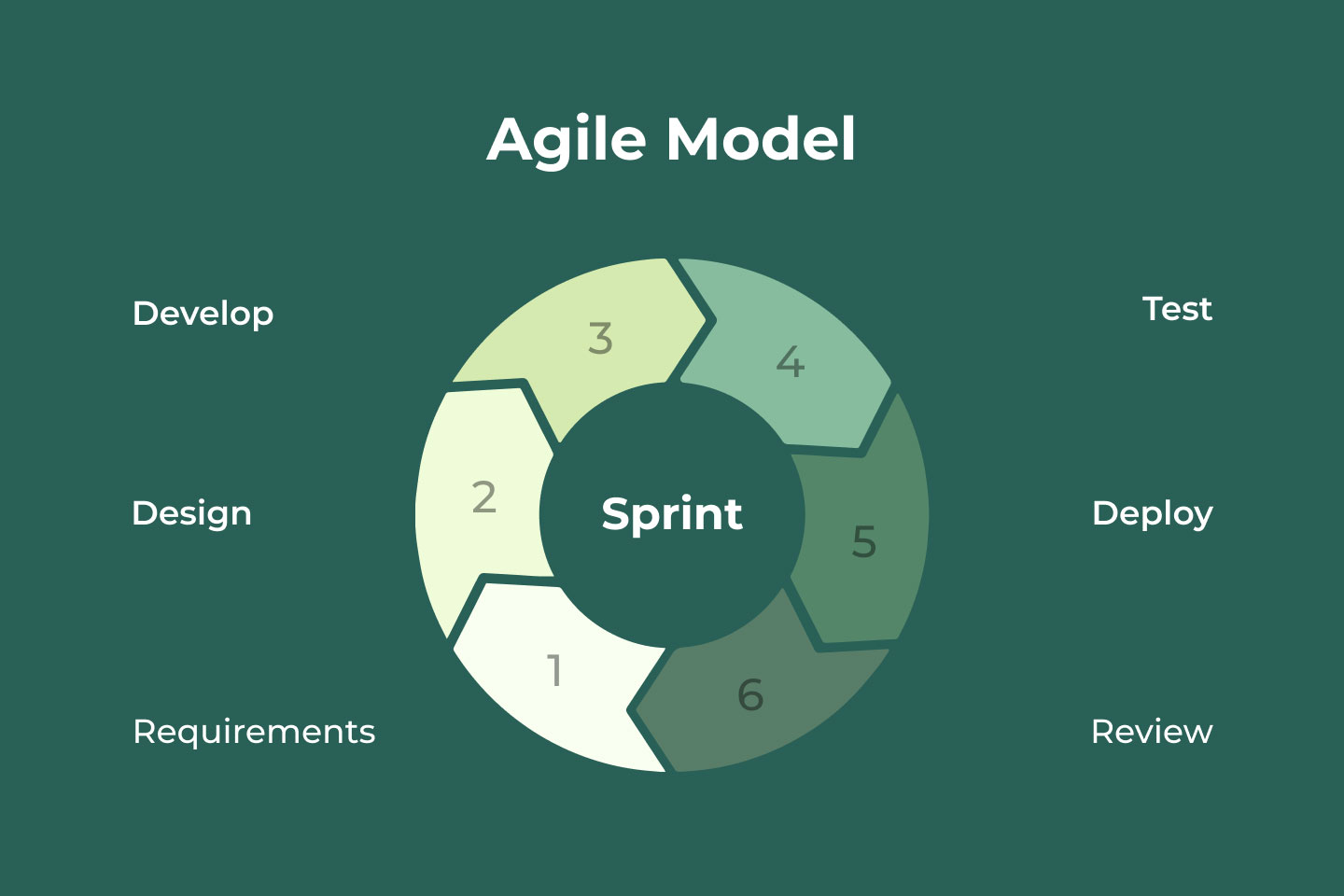
**Advantages:**

* Partial system deployments
* Reduced initial delivery time

**Disadvantages:**

* Requires good initial planning
* Total cost may be higher

1. **Agile Model:**

****

**Definition:**

An iterative, incremental approach emphasizing flexibility, collaboration, and rapid delivery of functional software.

**Phases:**

1. **Planning**: Define project vision and initial requirements
2. **Design**: Create flexible architectural approach
3. **Development**: Iterative coding with continuous integration
4. **Testing**: Ongoing quality assurance
5. **Deployment**: Incremental and frequent releases
6. **Review**: Continuous feedback and improvement

**Advantages:**

* High flexibility
* Continuous improvement

**Disadvantages:**

* Less predictability
* Requires active customer involvement