RIPHAH INTERNATIONAL UNIVERSITY

SCD (LAB)

LAB TASK # 01

NAME	**: ASAD ULLAH**
SAP ID	**: 55181**
SECTION	**: SE 5-2**

TASK # 01

Q : Write the uses, pros, and cons of the below software development models:

- **Waterfall**
- **V-Model**
- **Spiral**
- **Incremental**
- **Iterative**
- **Agile**
- **Scrum**

❖ Waterfall Model

Definition:

Waterfall is a step-by-step process where one phase must be finished before the next one starts. It flows like a waterfall – once water falls down, it cannot go back up

Uses (Where it is used):

- Projects where requirements are clear, fixed, and not changing.
- Best for small projects with predictable outcomes.
- Examples:
  - Banking system software (where rules don't change often).
  - Payroll systems.
  - School/college result management system.
  - Government projects.

Pros (Good points):

- Simple and easy to understand.
- Step-by-step method makes it easy to manage.
- Good for projects where everything is well-documented.
- Works well when client knows exactly what they want.

Cons / Limitations (Problems):

- No flexibility – once you finish a phase, it's hard to go back.
- Testing happens late, so mistakes are found too late.
- Not suitable for projects where requirements change.
- Can become time-consuming and costly if errors are found late.

❖ V-Model (Verification & Validation Model)

Definition:

V-Model is just like Waterfall but with more focus on testing at every step.
Each development phase has a matching testing phase. It looks like the letter “V” → on the left side is development, on the right side is testing.

Uses (Where it is used):

- Projects where quality and safety are very important.
- When requirements are clear and stable.
- Examples:
  - Medical devices software (like pacemaker software).
  - Automobile software (airbag control system).
  - Aviation systems (flight control).
  - Banking applications where accuracy is critical.

Pros (Good points):

- Testing is planned early, so mistakes are caught sooner.
- Very organized and structured.
- Ensures high-quality software.
- Good for projects with no requirement changes.

Cons / Limitations (Problems):

- Very rigid (no flexibility).
- If requirements change, it’s very hard to adjust.
- Expensive and time-consuming.
- Not good for small or fast-changing projects.

❖ Spiral Model

Definition:

The Spiral Model combines iterative development (building in small parts) with risk analysis (checking and controlling risks).
It looks like a spiral, where each loop (cycle) includes planning, designing, building, and testing. With every loop, the project grows bigger and better.

Uses (Where it is used):

- For large, complex, and expensive projects.
- When risks (like safety, budget, or failure chances) are high.
- Examples:
  - Military defense software.
  - Aerospace systems (NASA projects, space shuttles).
  - Large banking systems.
  - Huge ERP systems for enterprises.

Pros (Good points):

- Risk management is the main focus.
- Flexible – requirements can change during the project.
- Users can see early versions and give feedback.
- Better suited for long-term projects.

Cons / Limitations (Problems):

- Very expensive (needs more budget).
- Needs highly skilled people for risk analysis.
- Complex to manage compared to other models.
- Not suitable for small projects.

❖ Incremental Model

Definition:

In this model, the project is divided into small parts (increments).
Each increment adds a piece of the software, and step by step the full system is completed.

Example: First increment : login system,
Second increment : product catalog,
Third increment : payment system, and so on.

Uses (Where it is used):

- Projects where requirements can be divided into modules.
- When quick delivery of some features is needed.
- Examples:
  - E-commerce websites (Amazon, Daraz).
  - Mobile apps (adding features in updates).
  - Library management system (first books, then users, then reports).
  - Banking apps (basic banking first, then loans, then credit cards).

Pros (Good points):

- Working software is delivered early.
- Easier to test smaller parts.
- Flexible – changes can be made in later increments.
- Lower risk compared to full Waterfall.

Cons / Limitations (Problems):

- Needs good planning of increments.
- Integration of increments can be difficult.
- May need more resources and time for multiple cycles.
- If not managed well, system may look unfinished for a long time.

❖ Iterative Model

Definition:

In this model, the software is built in repeated cycles (iterations).
Each iteration includes planning → designing → coding → testing.
With every new iteration, the software improves and becomes closer to the final version.

Uses (Where it is used):

- Projects where requirements are not clear at the start.
- When client wants to see versions early and give feedback.
- Examples:
  - Social media apps (Facebook, Instagram keep improving).
  - Game development (basic version, then better graphics, then multiplayer).
  - Learning management systems (first upload feature, then quizzes, then analytics).

Pros (Good points):

- Early versions of software are ready for testing.
- Client can give feedback after each version.
- Easier to manage changing requirements.
- Risks reduced because mistakes are fixed in early cycles.

Cons / Limitations (Problems):

- Needs customer involvement at every stage.
- More time and resources may be required.
- Can lead to scope creep (endless changes if client keeps asking for more).
- Planning all iterations properly can be difficult.

❖ Agile Model

Definition:

Agile is a flexible and fast way of developing software.
Work is done in small pieces (iterations) with continuous feedback from the client.
Instead of one big release, Agile delivers working software quickly and often.

Uses (Where it is used):

- Projects where requirements keep changing.
- When client wants early and continuous delivery.
- Best for dynamic industries like startups, web apps, and mobile apps.
- Examples:
  - E-commerce apps (Daraz, Amazon).
  - Mobile apps (WhatsApp, Foodpanda).
  - SaaS products (Zoom, Slack).
  - Game updates released frequently.

Pros (Good points):

- Very flexible – easy to handle changes.
- Early delivery of useful features.
- Strong customer involvement.
- Encourages teamwork and communication.
- Higher customer satisfaction.

Cons / Limitations (Problems):

- Needs experienced developers.
- Can be hard to manage for very large projects.
- Less focus on documentation (can cause confusion later).
- Requires client to be available and engaged.

❖ Scrum (Agile Framework)

Definition:

Scrum is a framework of Agile used to manage and deliver projects.
Work is divided into short time periods called Sprints (usually 2–4 weeks).
It has defined roles:

- Product Owner – decides what features are needed.
- Scrum Master – makes sure Scrum rules are followed.
- Development Team – builds the product.

Uses (Where it is used):

- Projects that need quick delivery and frequent updates.
- When the client wants to see progress every few weeks.
- Examples:
    - Mobile apps (food delivery apps like Foodpanda, Careem).
    - Websites (adding new features regularly).
    - Startup products where requirements change often.
    - Game development with regular updates.

Pros (Good points):

- Fast delivery of working features.
- Improves teamwork and communication.
- Very flexible to changing requirements.
- Client sees results after every sprint.

Cons / Limitations (Problems):

- Needs a disciplined and experienced team.
- Can be difficult for large projects with many teams.
- Client must be involved regularly.
- If not managed well, project can go off track.