Summary of Lecture 5: User Stories and Agile Estimation

This lecture covers **User Stories** (a key requirement technique in Agile) and **Agile Estimation** methods used for project planning. Agile replaces **heavy documentation** with **collaborative, flexible, and fast-paced development**, and these methods help teams manage uncertainty efficiently.

User Stories: Capturing Requirements in Agile

🔹 **What is a User Story?**
A **short, simple** description of a feature **from a user's perspective**:
➡️ **Format:**

> "As a [type of user], I want [some goal] so that [some reason]."
> Example: *"As an astronaut, I want to align my ship so that I can park at the space station."*

🔹 **Why use User Stories?**

- Users **don't think in formal models** (like Use Cases).
- Stories make requirements **easy to understand and prioritize**.
- Helps **developers focus on user needs** rather than just system behavior.

🔹 **User Stories vs. Use Cases vs. Scenarios**

Feature	User Story	Use Case	Scenario
Definition	Brief statement of user need	Detailed step-by-step interaction	Sequence of user actions
Level of Detail	Minimal	Comprehensive	Sequential flow
Purpose	Defines a goal	Specifies system behavior	Describes a situation

---

Emergent Requirements & Epics

🔹 **Emergent Requirements** – **New requirements** that appear during development:

- **Unforeseen problems** arise.
- **New ideas** get introduced.

- Some features were **overlooked** in initial planning.

🔹 **Solution? Use Epics**

- **Epics = Big User Stories** that need to be **broken down later**.
- Instead of defining **every tiny detail** upfront, define **large goals** first.
- Example **Epic**: "Land on the moon" → Broken into **smaller stories** like "Align spaceship," "Deploy landing gear," etc.

Agile Estimation: Predicting Effort Without Overplanning

🔹 What is Agile Estimation?

- A **flexible**, **team-driven** way to estimate effort in Agile projects.
- Focuses on **relative size**, not absolute time (since software development is unpredictable).
- Encourages **team collaboration** and reduces misunderstandings.

🔹 **Why Use Agile Estimation?**

✅ Helps teams **plan iterations** and **release schedules**.
✅ Avoids **over-detailed planning**, which becomes **obsolete** when changes occur.
✅ Provides a **shared team understanding** of effort required.

Agile Estimation Methods

1️⃣ Planning Poker 🃏 (Most Common)

- **How it works:**
    1. Each team member **privately** selects an estimation card.
    2. Cards are revealed **at the same time** (to prevent bias).
    3. If estimates **differ widely**, team discusses and **re-estimates**.
- **Uses Fibonacci numbers (1, 2, 3, 5, 8, 13, 21...)** because larger tasks have more uncertainty.

2️⃣ T-Shirt Sizing 👕 (Quick & Simple)

- Tasks are categorized into **Small (S), Medium (M), Large (L), Extra Large (XL)** sizes.
- No exact time estimate—just a rough effort comparison.
- Works well for **high-level estimation** before breaking tasks down.

📌 **Example Mapping:**

T-Shirt Size	Estimated Effort
S	1-2 days
M	3-4 days
L	5-6 days
XL	7+ days (needs breaking down)

3️⃣ Wideband Delphi 🏛️

- Each expert **independently** estimates effort.
- Estimates are collected **anonymously**.
- Facilitator **reveals results** → Discussion happens → New estimates are made until consensus is reached.

4️⃣ Affinity Estimating

- **Similar tasks** are **grouped together** and estimated **as a batch**.
- Helps estimate **large backlogs** efficiently.

5️⃣ Dot Voting 🔴🔵

- **Each team member votes** for tasks they think are most important.
- **The most voted items** are prioritized first.

Cognitive Bias: Anchoring Bias

⚠️ **Anchoring Bias** = People tend to **base their estimates on the first number they hear** (even if it's wrong).

Example:
👷 Developer 1: "I think this task takes **30 days**."
👷 Other developers **adjust their estimates around 30 days**, even if **the real effort is much less**.

🔹 **How Agile prevents this?**
✅ **Planning Poker forces silent estimation first** before discussion.
✅ **Using relative estimation** (S, M, L) avoids attachment to exact numbers.

Final Takeaways

🔹 **User Stories = Simple, user-focused requirement descriptions** (No complex models).
🔹 **Emergent requirements exist, so Agile embraces flexibility**.
🔹 **Agile Estimation is not about accuracy—it's about shared team understanding**.
🔹 **Methods like Planning Poker & T-Shirt Sizing help avoid overplanning**.
🔹 **Avoid cognitive biases (e.g., Anchoring Bias) by forcing independent estimation first**.

Keywords

- **User Stories**
- **Epics**
- **Emergent Requirements**
- **Agile Estimation**
- **Planning Poker**
- **T-Shirt Sizing**
- **Wideband Delphi**
- **Affinity Estimating**
- **Dot Voting**
- **Anchoring Bias**