Question:Read about “Planning Poker” - Agile estimation technique and illustrate an example with a Development Team of 10 who are tasked to develop a mobile app for Maha-Khumb in 3 months.

Illustrating "Planning Poker" with an Agile Development Team for Maha-Khumb Mobile App

Scenario:
A Development Team of 10 members is tasked with building a mobile app for Maha-Khumb within 3 months. They use **Planning Poker**, an Agile estimation technique, to estimate user stories.

Step 1: Preparing the User Stories

The **Product Owner (PO)** has prepared a backlog with user stories. Examples include:

1. **User Registration** – Users should be able to register using mobile/email.
2. **Event Schedule** – Display Maha-Khumb event dates, locations, and details.
3. **Live Streaming** – Stream major Maha-Khumb events.
4. **Push Notifications** – Notify users about event updates.
5. **Navigation & Maps** – Show event locations with directions.

Step 2: Conducting Planning Poker

The team gathers, and each member is given a set of Fibonacci-numbered cards (1, 2, 3, 5, 8, 13, 21, etc.), which represent story points.

For each user story:

Example: Estimating "User Registration" Feature

1. The **Scrum Master** reads the story:
   “As a user, I want to register using my mobile number or email so that I can access event details and updates.”

2. The **Development Team** discusses the complexity, dependencies, and risks.
3. Each member selects a card representing their estimation (e.g., Developer A picks "5", Developer B picks "8").
4. Everyone reveals their cards simultaneously.
5. The highest and lowest estimators explain their reasoning.
6. After discussion, the team re-votes until they reach a consensus (e.g., they agree on "5" story points).

Step 3: Estimating All Stories

The team follows the same process for all user stories. Example estimates:

- **User Registration** → 5 Story Points
- **Event Schedule** → 3 Story Points
- **Live Streaming** → 13 Story Points (complex)
- **Push Notifications** → 5 Story Points
- **Navigation & Maps** → 8 Story Points

Step 4: Sprint Planning

- The team determines their **velocity** (e.g., they can complete ~20 story points per sprint).
- They plan which stories fit into **Sprint 1** (e.g., Registration, Schedule, and Push Notifications).

Q2: Read Paper – Measuring Software Development Waste in OSS Projects - https://arxiv.org/pdf/2409.19107. Pick one measure from this paper and apply it on any open-source repository. Share results.

To assess the **Stale Forks (SFs)** metric for the PyTorch repository, we analyze the activity levels of its forks. A fork is considered **stale** if it has not been updated or has minimal activity over a significant period.

Key Data:

- **Total Forks:** 22,261
- **Active Forks:** 12,406
- **Stale Forks:** 9,855

Stale Forks Percentage:

Stale Forks Percentage=(Stale ForksTotal Forks)×100%=(9,85522,261)×100%≈44.3%\text{Stale Forks Percentage} = \left(\frac{\text{Stale Forks}}{\text{Total Forks}} \right) \times 100\% = \left(\frac{9,855}{22,261} \right) \times 100\% \approx 44.3\%Stale Forks Percentage=(Total ForksStale Forks)×100%=(22,2619,855)×100%≈44.3%

This indicates that approximately 44.3% of the forks are inactive or have minimal activity.

Implications:

A higher percentage of stale forks can suggest:

- **Underutilization of Resources:** Developers may be creating forks without substantial contributions or updates.
- **Potential for Cleanup:** Maintainers might consider strategies to encourage active development or clean up inactive forks.

Considerations:

- **Definition of Activity:** The criteria for what constitutes an "active" fork can vary.
- **Time Frame:** The period considered to determine activity status is crucial.

Monitoring the SFs metric helps maintainers understand the engagement level of the community and the effectiveness of collaboration within the project.