Planning Poker Estimation for Maha-Khumb Mobile App Development

Prepared by: P.Likhith

Roll Number: 205224015

Example: Using Planning Poker for Estimating a Maha-Khumb Mobile App Development

A development team of 10 members is tasked with building a mobile app for Maha-Khumb within 3 months.

The app needs to support key features like user registration, event schedules, GPS navigation, push notifications, and vendor listings. The team decides to use Planning Poker for estimation.

Step 1: Introduction to Estimating Approaches

Before starting, the Scrum Master (facilitator) explains the estimation process. The team agrees to use

Planning Poker, based on the modified Fibonacci sequence (1, 2, 3, 5, 8, 13, etc.) for estimating effort in story points.

Step 2: Understanding Planning Poker

- Each team member receives a deck of Planning Poker cards.
- The Product Owner presents a user story from the backlog.
- Example User Story:

  "As a user, I want to register and log in using my phone number and OTP."

- The team discusses the complexity, including authentication, OTP verification, and database storage.

- After the discussion, all members simultaneously reveal their cards.

Step 3: Consensus Building in Estimation

First Round of Estimation

- Team members reveal different estimates: 3, 8, 2, and 5.

- Since estimates vary, the Scrum Master asks outliers (e.g., the one who picked 8 and the one who picked 2) to explain their reasoning.

  - The developer who picked 8 thinks OTP security and backend authentication will take more effort.

  - The developer who picked 2 believes using Firebase authentication will reduce complexity.

Step 4: Iterative Estimation Process

- After discussion, team members re-estimate and reveal their cards again.

- New estimates: 5, 5, 5, 8, 5, 5, 5, 5, 5, 5. Majority agrees on 5 story points, so it is recorded.

Step 5: Estimating Other Features

The same process is repeated for other user stories:

User Story	First Estimates	Final Estimate
GPS Navigation	8, 13, 8, 8, 5	8 Story Points
Live Event Updates	5, 5, 3, 5, 5	5 Story Points
Push Notifications	3, 3, 3, 3, 2	3 Story Points
Vendor Listings	8, 5, 5, 5, 5	5 Story Points
Emergency Contacts	2, 3, 2, 3, 2	2 Story Points

Step 6: Handling Discrepancies in Estimates

Scenario 1: The GPS Navigation feature had a split between 8 and 13.

  - Some developers argued it was complex due to real-time tracking, while others felt using Google

Maps API would simplify it.

- The team decided on 8 story points after agreeing on the best technical approach.

Scenario 2: The Push Notifications feature had 3 and 2 as estimates.

- The developer who picked 2 thought it was simple with Firebase Cloud Messaging (FCM).
- The developer who picked 3 considered additional effort for custom notification templates.
- The team settled on 3 story points after discussion.

Step 7: Establishing Baselines for Estimation

- The team selected the "User Registration" story (5 points) as a baseline.
- They then identified another "Live Event Updates" story as a 5 to validate consistency.
- This helped ensure all future estimates remain relative to these baselines.

Step 8: Using Planning Poker Resources

- The Scrum Master suggested using an online Planning Poker tool for remote team members.
- The team is now aligned with estimates and can move forward with Sprint Planning.

Conclusion

Using Planning Poker, the Maha-Khumb mobile app development team successfully estimated effort, resolved uncertainties,

and built a realistic development plan for completing the app within 3 months.

Analysis of PR Rejection Rate

1. Introduction

Pull Request (PR) Rejection Rate is a key metric used to evaluate the efficiency of software development processes in open-source projects. According to the research paper 'Measuring Software Development Waste in OSS Projects' (https://arxiv.org/pdf/2409.19107), PR rejection rate helps in identifying wasted efforts and improving collaboration.

2. Formula & Calculation

The PR Rejection Rate is calculated using the formula:

PR Rejection Rate = Unmerged Closed PRs / Total Closed PRs

From the given data:

- Total Closed PRs: 100
- Merged PRs: 84
- Unmerged PRs: 16

Thus, PR Rejection Rate = 16 / 100 = 0.16 (16%)

3. Interpretation of Results

The results indicate that:

- 84% of closed PRs were merged, showing a high acceptance rate.
- 16% of PRs were closed without merging, indicating rejection or abandonment.

A low rejection rate (0.16) suggests that project maintainers actively engage with contributors, ensuring that most PRs meet the required standards.

4. Key Takeaways & Recommendations

- The project has an efficient PR process with a high acceptance rate.
- A PR rejection rate of 16% is relatively low, meaning most contributions add value.

Potential next steps:

- Analyze reasons for PR rejections (e.g., outdated contributions, poor quality, conflicts).
- Compare rejection rates across repositories to identify trends and areas for improvement.

PR Rejection Rate Screenshot