Software Development Process (SDP)

[Principles](#)
[Process](#)
[Roles](#)
[Tooling](#)
[Definition of Done (DoD)](#)
[Release Cycle](#)
[Environments](#)

Principles

Responsive Communication

- Team members commit to responding to asynchronous communication within 24 hours.
- Backlog Management

- Utilize a Github projects board for continuous backlog management.
- Ensure that the backlog is maintained with work items for at least the next two weeks.

Work Item Definition

Each work item in the backlog must include:

- User story or motivation.
- Technical description.
- Dependencies.
- Estimate.
- Acceptance criteria.

Version Control

- All code changes should be developed in separate git branches to ensure isolation.

Code Review Process

- After completing a feature, bug fix, or refactoring, create a Pull Request (PR).
- PRs must be reviewed by at least one team member before merging.

Definition of Done (DoD)

- Ensure that PRs comply with the Definition of Done, which should include quality checks, testing, and documentation.
- Link PRs to the corresponding work items in the backlog.

Process

Software Development Lifecycle/ software development process:

Backlog Management

- The product owner and development team continuously maintain the backlog.
- New work items are created with user stories, technical descriptions, estimates, and acceptance criteria.
- Dependencies are identified and documented.

Development

- Developers create new branches for their work items, following the version control principle.
- They develop and test the code, ensuring that it complies with the Definition of Done.

Code Review

- Developers create Pull Requests for completed work items.
- At least one team member reviews the code, providing feedback and ensuring quality.

Demo/Review with Stakeholders

- Showcase completed work items.
- Collect feedback from stakeholders and make necessary adjustments.
- Validate that work items meet acceptance criteria.

Merge and Deployment

- Once a Pull Request is approved, it is merged into the main branch.
- (Possibility) Continuous integration/continuous deployment (CI/CD) pipelines may automatically build and deploy the changes.

Adaptation

- Adapt and refine the process based on feedback, performance data, and changing project needs.

Roles

Based on our client meeting, I believe we will be building a mobile Android app that interfaces with a backend and involves designing the user interface (UI). We could break up the work as follows:

Systems Developer (2 members):

- Responsible for Android app development.
- Implement the user interface and user experience.

- Integrate with the backend API.
- Ensure the app's performance, responsiveness, and compatibility with various Android devices.
- Collaborate closely with the UI/UX Designer.

UI/UX Designer (2 member):

- Create the visual design and user experience of the app.
- Design app screens, layouts, icons, and other graphical elements.
- Ensure a user-friendly and visually appealing interface.
- Work closely with the Systems developer(s) to implement the design.

Quality Assurance (QA) Tester (Everyone):

- Responsible for testing the Android app to identify and report bugs or issues.
- Conduct functional testing, usability testing, and performance testing.
- Ensure the app meets quality and performance standards.
- Collaborate with developers to resolve issues.

Project Manager or Scrum Master (1 member, Also works with UI or systems development):

- Coordinate and plan project tasks, timelines, and milestones.
- Facilitate communication and collaboration among team members.
- Track progress, manage resources, and handle any project-related issues.
- Ensure the project stays on schedule and within budget..

Tooling

Version Control	GitHub.
Project Management	GitHub Issues and Projects
Documentation	https://github.com/withastro/starlight (recommended), Astro, Sphinx, Read the Docs, Markdoc, Confluence, Notion, GitHub Wiki, README, etc.
Test Framework	Possibly Espresso: Espresso is an Android UI testing framework for creating user interface (UI) tests. Unit Testing with Mock Server: A mock server simulates the backend responses, without relying on the actual backend

Linting and Formatting	Android Lint will analyzes Android projects for potential issues, like compatibility problems
CI/CD	GitHub Actions (maybe)
IDE	Visual Studio Code Android Studio: Official (IDE) for Android app development.
Graphic Design	Adobe XD or Sketch: For UI/UX design and prototyping. Figma: For collaborative design and prototyping. InVision: For creating interactive design prototypes.
Others	Discord and MS teams for Comminucation. Zoom for client meetings

Definition of Done (DoD)

Acceptance Criteria: The acceptance criteria defined for the issue, task, or user story have been met. This includes functional requirements, user stories, and specific criteria outlined by the product owner or stakeholders.

Code Changes: All code changes related to the work item are implemented and thoroughly reviewed. Code is properly structured, and best practices are followed.

Merged to Main Branch: Changes are merged into the main branch of version control (e.g., Git) after review and approval.

Testing:

- Unit Testing: Unit tests have been written and are passing.
- Integration Testing: Integration tests, ensuring that the code works well with other components, are successfully passed. Changes are implemented in all relevant components, such as backend, frontend, libraries, and any other areas where they are applicable
- Regression Testing: Full regression tests are executed to ensure that new changes do not introduce regressions in existing functionality.

Documentation:

- Documentation, including code comments, user manuals, and technical documentation, is updated regularly.
- Deployment Instructions are included to facilitate deployment to various environments.
- Release notes are updated to reflect the changes, bug fixes, and new features included in the release.

Backward Compatibility: Any breaking changes are evaluated and, if necessary, are avoided or well-documented. Efforts are made to maintain backward compatibility.

Staging Deployment: The changes are deployed to a staging environment for final testing and validation before production deployment.

Demo Preparation: A demo or presentation of the changes is prepared for the next stakeholder meeting or sprint review.

Quality Assurance: The work item has passed quality assurance checks and has been reviewed for completeness and correctness.

Performance Testing: If performance-related changes were made, they have been validated against performance criteria.

Accessibility Testing: If applicable, accessibility criteria have been met and verified.

Release Cycle

Considering a 6 month to release time frame:
Week 1: Project Startup

- Conduct Team Startup meetings to discuss project goals, requirements, and scope.
- Define the release schedule and milestones.

Weeks 2-3: Design and Wireframing

- UI/UX designers work on wireframes and initial design concepts.
- Get stakeholder feedback on design iterations.

Week 4: Architecture and Component Design

- Define the app's architecture and component structure.
- Backend and frontend teams collaborate to finalize API specifications.

Weeks 5-18: App Development

- Developers work on implementing the app's features, backend integration, and UI.
- Frequent code reviews, continuous integration, and testing cycles.

Weeks 19-21: Integration Testing

- Combine the app's frontend with the backend to ensure proper data flow.
- Verify that API endpoints function as expected.

Weeks 22-24: Alpha Testing

- Internal testing phase with the development team and selected testers.
- Identify and address issues, bugs, and performance bottlenecks.

Week 25: Beta Testing

- Release a beta version of the app to a limited group of external testers.
- Collect feedback on usability, performance, and any issues.

Weeks 26-27: Optimization

- Based on beta feedback, work on performance optimization, bug fixes, and user interface refinements.
- Ensure frequent releases to the beta group.

Week 28: User Acceptance Testing (UAT)

- Conduct UAT with stakeholders to ensure the app meets their expectations.
- Finalize any pending changes based on UAT feedback.

Week 29: Documentation

- Prepare user documentation, FAQs, and help resources.

Weeks 30-32: Final Testing and Quality Assurance

- Rigorous testing, including regression testing and security assessments.
- Ensure that the app complies with any platform-specific guidelines.

Week 33: Pre-Launch

- Submit the app to Google Play Store for review.
- Configure monitoring and error tracking tools.

Week 34: Official Release

- Launch the app on the Google Play Store.
- Notify stakeholders, customers, and users about the release.
- Monitor initial user feedback and address any immediate issues.

Week 35: Post-Launch

Continuously monitor app performance, user feedback, and bug reports.
Plan for future updates and feature enhancements.

Environments

Environment	Infrastructure	Deployment	What is it for?	Monitoring
Production	Google play store	Final Release	Google Play is the expected distribution platform for Android apps.	Google Play Console for monitoring and managing apps
Staging (Test)	Staging server	A dedicated staging server replicating the production environment.	This server host backend services, databases, and any other components that interact with the Android app	Detailed logging in the staging server environment to capture key events and interactions with the ap
Dev	Local (macOS and Windows)	Commit	Development and unit tests	N/A