**Coding Guidelines**

### **1. Coding Guidelines**

* File Naming: Use PascalCase for component/Hook files and camelCase for utility files.
* Reusable Components: Place common components in a components folder and avoid duplicating logic.
* Avoid any: Use proper interfaces or types and limit the use of any.
* Utility Types: Use built-in TypeScript utility types like Pick, Partial, or Record.
* Global Types: Store shared types in a types/ directory.
* Functional Components: Prefer functional components with hooks over class components.
* Avoid Inline Styles: Define styles outside the render method for performance.
* Safe Imports: Avoid deep imports from React Native libraries
* Add comments for non-obvious business logic or workarounds.
* Add comments to each file/hook explaining its purpose.
* Use Husky for managing Git hooks to enforce linting and type-checking during commits and pushes.
* Use ESLint for consistent code quality with recommended configurations for TypeScript and React Native.
* Use Prettier for formatting and integrate it with ESLint for seamless code styling.
* Ensure type safety using the TypeScript compiler and include type checks in the automation process.
* Use lint-staged to run linters and formatters on staged files to maintain code quality during commits.
* Use static code analysis tools(SonarQube) for maintainability tracking and enforcing standards.
* –no-verify option for code commits should only be limited to POC/SPIKE branches.

### **Team Expectations**

* Code Reviews: Enforce mandatory reviews with a minimum 2 approvals from leads one from each TCS & GIB.
* PR Author to resolve review comments and follow up/pair with the reviewer for comments, discussions and resolution.