Team Working Agreement

Last Revised: 11/09/2025
Fall Quarter

Grow My Garden

Nuha Fakrudeen - Product Owner
Justin Valdez - Developer
Chris Fang - Developer
Gavin Leach - Developer

Definition of Done

User Stories

- Functionality implemented fully
    - All acceptance criteria for a user story are met and functionality is implemented as needed by requirements.
- Code integrated in correct place, styled, and tested (manual or automated)
    - Code is merged with the main branch and there are no merge conflicts.
    - Tests & Build must pass successfully before being merged.
    - Code must pass Lint Test (kotlin: spotless, swift: SwiftLint)
    - Do not skip the Github Actions
- Additional components removed
    - Anything that isn't necessary to functionality of code must be removed (i.e excess comments, files that aren't used, anything unaccessed)
    - Debug statements removed

Sprints

- Demo sent (if applicable) and approved of by group (i.e. no objections or suggestions)
- All planned user stories in the sprint backlog have been completed and meet DoD for user stories
    - Any incomplete stories are moved to next sprint backlog
- Sprint retrospective is held to discuss what went well, what didn't, and areas for improvement
- Sprint artifacts (burndown chart, backlog) are updated to reflect progress

Style Guide

Our Styles

- Kotlin:
    - Follows the ktLint formatting guidelines
    - Use Koin to implement Dependency injection
        - Constructor Injection is preferred
    - Follows the Android Developer Kotlin Style Guide:
        - https://developer.android.com/kotlin/style-guide
- Apple's Swift API Design Guidelines
    - Using PascalCase for types/enums
        - Ex: AuthManager, PlantsHomeView, PlantTask, etc.
    - Using camelCase for variables/functions
        - Ex: isLoggedIn, isAddingPlant, showReminders, add(_:)
    - Keeping configuration out of the code
        - Use config.xcconfig and Info.plist for API, flags, permissions
    - Encapsulating UI into reusable SwiftUI views
        - Ex: PlantCard, HomeHeader, RoundedBottomBar, etc.
        - This improved reuse and keeps things like PlantsHomeView and AuthRootView readable.
    - Using managers for system features
        - NotificationManager handles all notification logic
        - PhotoPermissionManager handles photo permissions and settings redirection
        - Views call these managers instead of talking directly to system APIs
    - Keeping files structured in a consistent order
        - Imports -> managers -> models -> main screens -> smaller things -> helpers -> previews
        - Using // MARK: headers to separate sections
    - Add TO:DOs when needed to come back (and to make sure things aren't forgotten
    - Reusing shared visual style/colors
        - Using the same green gradient background and color names for a consistent theme.
- Make sure functions/variables are given clear names to ensure readability

Project Structure

- Shared
    - commonMain
        - kotlin.com.gmg.growmygarden
            - data
                - Contains relevant packages for managing/storing data
                - Sub packages should be organized in a sensible manor
            - di
                - Contains Files for managing DI via KOIN
                - Each Module should have it's own file
            - viewmodel
                - Contains all Viewmodels (each gets its own file)
    - commonTest
        - kotlin.com.gmg.growmygarden
            - Contains all common tests (platform irrelevant)
            - Each Unit tested has its own file
    - androidMain/iosMain
        - Contains any actual implementations for commonMain and follows the same file structure
        - Actual implementations must have .ios or .android added after the file name and before the file type in order to work correctly
    - androidTest/iosTest
        - Tests that require platform specific testing implementations
        - Follows the same structure as commonTest
- iosApp
    - Configuration
        - Config.xcconfig
            - Contains configuration information for frontend
            - Any environment flags, constants, API base URLS, etc.
    - iosApp
        - Assets.xcassets
            - Contains colors for the UI interface
        - ContentView.swift
            - Majority of the frontend code
        - Info.plist
            - Contains permissions for photos and notification permissions
        - iOSApp.swift
            - Main app entry
- Vendor – Any needed linking libraries go her