

| Android Test Cases | | | | | | | | |
|--|---|-----------------------------------|---|---|----------------|----------|----------|--|
| App Installation & First Launch | | | | | | | | |
| Test Case ID | Title | Scenario | Steps | Expected Result | Actual Result | Severity | Priority | |
| AND_01 | Verify that the application can be successfully installed and launched from Google Play Store | User installs app from Play Store | 1. Open Google Play Store on Android device. 2. Search for the application name. 3. Tap on Install button. 4. Wait for installation to complete. 5. Tap Open to launch the app. | The application should install without any errors and should launch successfully, displaying the onboarding screen on first launch. | To be executed | Critical | High | |
| AND_02 | Verify that onboarding flow is displayed on first launch after fresh installation | First time app launch | 1. Install the app on a fresh device or after clearing app data. 2. Launch the application. | The application should start with the first onboarding screen and guide the user through the onboarding flow. | To be executed | Major | High | |
| Onboarding Flow | | | | | | | | |
| Test Case ID | Title | Scenario | Steps | Expected Result | Actual Result | Severity | Priority | |
| AND_03 | Verify that user can navigate through onboarding screens using swipe gestures | User navigates onboarding | 1. Launch the application. 2. On onboarding screen 1, swipe left. 3. Repeat swipe for screen 2 and screen 3. | User should be able to smoothly navigate through all onboarding screens using swipe gestures without lag or UI issues. | To be executed | Major | Medium | |
| AND_04 | Verify that user can skip onboarding and proceed directly to login | User skips onboarding | 1. Launch the application. 2. Tap on Skip button on any onboarding screen. | User should be redirected directly to the Login screen, and onboarding should be marked as completed. | To be executed | Major | Medium | |
| AND_05 | Verify that onboarding state is retained when app is sent to background | App lifecycle during onboarding | 1. Launch the app and stay on onboarding screen 2. 2. Press Home button to send app to background. 3. Reopen the app. | The application should resume from the same onboarding screen where the user left off. | To be executed | Major | Medium | |
| Permission Handling | | | | | | | | |
| Test Case ID | Title | Scenario | Steps | Expected Result | Actual Result | Severity | Priority | |
| AND_06 | Verify that push notification permission is granted when user selects Allow | User allows push permission | 1. Launch the application. 2. When push permission prompt appears, tap Allow. | Application should successfully register for push notifications and store the permission status as allowed. | To be executed | Major | High | |
| AND_07 | Verify that application behaves correctly when user denies push notification permission | User denies push permission | 1. Launch the application. 2. When permission prompt appears, tap Deny. | Application should continue functioning normally without crashes, and should not attempt to send notifications. | To be executed | Major | High | |
| AND_08 | Verify that permission can be requested again when user selects Ask Later | User postpones permission | 1. Launch app. 2. Tap Ask Later on permission prompt. 3. Navigate to relevant feature later. | Application should show permission prompt again at an appropriate time. | To be executed | Minor | Medium | |
| Network & App Lifecycle | | | | | | | | |
| Test Case ID | Title | Scenario | Steps to Reproduce | Expected Result | Actual Result | Severity | Priority | |
| AND_09 | Verify that login behaves correctly under poor network conditions | User logs in with weak network | 1. Enable slow network mode or turn on airplane mode. 2. Attempt to login. | Application should show loading indicator or meaningful error message without freezing or crashing. | To be executed | Major | High | |
| AND_10 | Verify that application state is handled correctly after app is killed | App killed and relaunched | 1. Login successfully. 2. Kill the app from recent apps list. 3. Relaunch the app. | Application should either restore user session or redirect to Login screen as per design. | To be executed | Critical | High | |

| | | | | | | | |
|--------|--|---------------------------|---|--|----------------|-------|--------|
| AND_11 | Verify that UI layout remains stable after screen rotation | Device orientation change | 1. Open Profile screen. 2. Rotate device from portrait to landscape. | UI should adjust correctly without losing entered data or breaking layout. | To be executed | Minor | Medium |
|--------|--|---------------------------|---|--|----------------|-------|--------|

iOS Test Cases

| App Installation & First Launch (iOS) | | | | | | | |
|---------------------------------------|--|----------------------------------|--|---|----------------|----------|----------|
| Test Case ID | Title | Scenario | Steps to Reproduce | Expected Result | Actual Result | Severity | Priority |
| IOS_01 | Verify that the application can be installed and launched from Apple App Store | User installs app from App Store | 1. Open App Store on iOS device. 2. Search for the application. 3. Tap Install. 4. Launch the app after installation. | Application should install successfully and open the onboarding screen on first launch. | To be executed | Critical | High |
| IOS_02 | Verify that onboarding flow starts correctly on first launch | First time app launch | 1. Install the app freshly. 2. Launch the app. | User should see the first onboarding screen and be guided through the flow. | To be executed | Major | High |

Permission Handling

| Test Case ID | Title | Scenario | Steps to Reproduce | Expected Result | Actual Result | Severity | Priority |
|--------------|---|---------------------------------------|---|---|----------------|----------|----------|
| IOS_03 | Verify that iOS system permission popup is displayed for push notifications | iOS permission prompt | 1. Launch the app for first time. 2. Observe the permission dialog. | Native iOS system permission popup should be displayed to the user. | To be executed | Major | High |
| IOS_04 | Verify that application continues normally after denying push permission | User denies permission | 1. Tap Don't Allow on permission popup. | App should not crash and should continue functioning without notifications. | To be executed | Major | High |
| IOS_05 | Verify that user can enable push notifications from iOS Settings | User enables permission from settings | 1. Deny permission initially. 2. Go to iOS Settings then App then Notifications. 3. Enable notifications. | Application should start receiving push notifications. | To be executed | Major | Medium |

App Lifecycle & Device Variations

| Test Case ID | Title | Scenario | Steps to Reproduce | Expected Result | Actual Result | Severity | Priority |
|--------------|--|---------------------------|--|--|----------------|----------|----------|
| IOS_06 | Verify that application resumes correctly from background | App sent to background | 1. Login to app. 2. Press Home to send app to background. 3. Reopen the app. | App should resume from last state without data loss. | To be executed | Major | High |
| IOS_07 | Verify that application launches correctly after force close | App force closed | 1. Force close the app. 2. Reopen from home screen. | App should launch normally without crash or blank screen. | To be executed | Critical | High |
| IOS_08 | Verify that UI adapts properly on small screen devices | Small screen device usage | 1. Open app on small device (e.g. iPhone SE). 2. Navigate through Profile screen. | UI elements should fit properly without overlapping or truncation. | To be executed | Minor | Medium |
| IOS_09 | Verify that orientation behavior follows supported rules | Device orientation change | 1. Rotate device orientation. | App should follow defined orientation rules without UI break. | To be executed | Minor | Low |

A short explanation of real device vs emulator testing strategy

| Emulator / Simulator Testing | | | |
|------------------------------|----------|---|--|
| | Purpose | Early-stage and continuous testing | |
| | Used for | Basic functional testing (login, onboarding, profile flows) | |
| | | UI validation and layout checks | |
| | | Regression testing after new builds | |
| | | Testing across multiple OS versions quickly | |

| | | | | | | | |
|--|--|-------------------------------|--|--|--|--|--|
| | | Advantages | Fast execution | | | | |
| | | | Easy setup and configuration | | | | |
| | | | Cost-effective | | | | |
| | | | Supports parallel testing on different OS versions | | | | |
| | | Real Device Testing | | | | | |
| | | Purpose | Pre-release and production validation | | | | |
| | | Used for | Push notification testing | | | | |
| | | | Network interruption scenarios | | | | |
| | | | App lifecycle events (background, kill, resume) | | | | |
| | | | Performance and responsiveness checks | | | | |
| | | | Device-specific and hardware-related issues | | | | |
| | | Advantages | Real hardware behavior | | | | |
| | | | Accurate OS restrictions and permissions | | | | |
| | | | Real network conditions | | | | |
| | | | Closest to real user experience | | | | |
| | | Final Testing Strategy | Emulators are used for early functional coverage and quick feedback, while real devices are mandatory before release to validate critical features and ensure real-world reliability and production readiness. | | | | |
| | | | | | | | |