Church Library App — Phase 1.6 Full Test Cases

Purpose: Complete, traceable test cases covering installation/activation, local flows, admin & librarian day-to-day operations, sync behavior, and edge cases for Phase 1.6. Document prepared for QA execution and sign-off.

Assumptions / Pre-reqs - Device A and Device B (Android) available for two-device cloud tests. - Supabase functions deployed and reachable: `auth-activate`, `sync-push`, `sync-pull`. - App built with `EXPO_PUBLIC_ONLINE_MODE=true` for cloud testing or `false` for offline mocked testing per scenario. - Local DB schema migrated; `shifts` table created and included in snapshots. - Test accounts in server (or mocked response) for `DiguwaSoft` (admin) and `lib1` (librarian). - Tester has access to the project testPlan.md and migration logs.

How to read the cases

- **Step**: Action to execute
- **Input**: Values entered or environment setup
- **Expected output / assertion**: Visible behavior, DB state or server response to validate
- **Notes/Edge cases**: Variations to test and expectations

---

1. Default Admin Flow (Fresh Install / First-time admin activation)

Goal: Validate a brand-new device activation by default admin provided by server snapshot.

Preconditions - App freshly installed (no local DB). - `EXPO_PUBLIC_ONLINE_MODE=true` and `EXPO_PUBLIC_API_BASE_URL` configured to point to Supabase functions. - Server contains admin user `DiguwaSoft` with temporary PIN `1366` and shifts (optional).

Steps 1. Launch app → observe `Initial Setup Required` screen. - **Input:** none - **Expect:** Button `Begin Setup (Online Required)` visible.

1. Tap `Begin Setup` → Activation screen appears.
2. **Input:** Username: `DiguwaSoft`, PIN: `1366`
3. **Expect:** Activating UI; network call to `auth-activate` returns `{ok:true, snapshot, role:'admin', require_pin_change: true}`.
4. Activation applies snapshot locally.

5. **Expect:** `applySnapshot` logs: `Binding device <device_id> to librarian DiguwaSoft` and snapshot entries inserted into local tables: `librarians`, `books`, `users`, `shifts`, `commits`, `meta`.
6. **DB check:** `SELECT * FROM librarians WHERE username='DiguwaSoft'` shows row with `device_id` equal to device id saved in `meta.device_id`.
7. Because `require_pin_change === true`, user is redirected to `/auth/change-pin`.
8. **Input:** Old/temporary PIN: `1366`, New PIN: `2468`, Confirm: `2468`
9. **Expect:** Success message `PIN changed successfully.` and redirected to Admin dashboard `/admin`.
10. **DB check:** `SELECT pin_salt, pin_hash FROM librarians WHERE username='DiguwaSoft'` -> non-empty `pin_salt` and `pin_hash`.
11. On admin dashboard, verify `Admin Panel` shows username and menu.
12. **Expect:** Admin sees `Manage Librarians`, `Device Management`, `Sync & Cloud Control`, etc.
13. Log out.
14. **Expect:** Session cleared and returned to `/auth/login`.

Edge/Negative Cases to run - Wrong PIN: enter `0000` → Activation fails with `Activation Failed: Incorrect PIN`. - Missing network: Activation shows network error and friendly alert.

Pass criteria - Device bound to admin in local DB and meta updated. - PIN change enforced and stored. - Admin dashboard accessible post-change.

---

2. New Admin Flow (Admin creates another admin via Manage Librarians)

Goal: Admin adds a second admin and that admin activates a second device successfully.

Preconditions - Device A has `DiguwaSoft` admin activated. - Device B is fresh install. - Admin on Device A has network access to push commit or uses server-side user creation if running server mode.

Steps 1. On Device A (logged in as `DiguwaSoft`) → Admin → Manage Librarians → `+ Add Librarian`. - **Input:** Username: `AdminTwo`, Full name: `Admin Two`, Role: `admin`. - **Expect:** UI displays `Librarian Created username: AdminTwo Temporary PIN: XXXX`. - **DB check (Device A):** `INSERT` created in

`pending_commits` and `commits` if push is immediate; local `librarians` contains `AdminTwo` with `pin_salt` and `pin_hash`.

1. Sync push on Device A (manual via FAB or Admin Sync button) — ensure `pushPendingCommits()` executes.
2. **Expect:** `sync_log` contains a success entry mentioning pushed commit for `insert(librarians)`.
3. **Server check (Supabase):** `SELECT * FROM librarians WHERE username='AdminTwo'` returns row.
4. Device B fresh install → Begin Setup → Activate with `AdminTwo` temporary PIN shown earlier.
5. **Input:** Username: `AdminTwo`, PIN: `<temp PIN>`, device_id: Device B ID.
6. **Expect:** Activation returns snapshot containing new admin and sets device binding for `AdminTwo` → Device B's local DB now includes `AdminTwo` with `device_id = Device B`.
7. Change PIN enforced if `require_pin_change === true` → change to a secure new PIN.
8. **Expect:** PIN change success and dashboard accessible.

Edge cases - Delay in server push on Device A: Device B activation fails until server receives commit. Test: Do not push from Device A; attempt Device B activation → expected failure `Invalid username`.

Pass criteria - New admin created and visible on server. - Device B activation binds to AdminTwo.

---

3. Day-to-Day Admin Flow

Goal: Verify admin operations: reset PIN, unbind a device, run manual sync, view sync status.

Preconditions - Device A with `DiguwaSoft` admin activated and bound. - Device B with `AdminTwo` activated and bound.

Steps 1. On Device A → Admin → Manage Librarians → select `AdminTwo` → `Reset PIN`. - **Input:** Confirm `Reset`. - **Expect:** Temporary PIN generated e.g. `0000` (or randomized), `updateLibrarianPin()` updates `pin_salt` and `pin_hash` locally, commit added to `pending_commits`.

1. Press `Sync` FAB (or Manual Sync) on Device A to push commits.
2. **Expect:** `pushPendingCommits()` returns success; server `librarians` table updated with new `pin_salt` / `pin_hash`. `pending_commits` and related local commits marked synced.
3. On Device B, attempt login with AdminTwo and temporary PIN.

4. **Input:** Username `AdminTwo`, temp PIN.
5. **Expect:** If activation of Device A's reset propagated, Device B login should succeed and force PIN change if `require_pin_change` is true for reset flow; otherwise allows login and require change next login depending on business rule.
6. `Unbind Device`: On Device A, unbind `AdminTwo` (or any librarian).
7. **Expect:** `device_id` becomes NULL on server after sync; Device B will be prevented for future activations if unbound.
8. View Sync Status indicator on dashboard — shows `pending` count, `lastPush`, `lastPull`.
9. **Input:** none
10. **Expect:** Readable sync status with timestamps.

Edge/Negative Cases - Push failure (server down) → `insertSyncLog` records failure and UI shows alert.

Pass criteria - Admin reset/unbind operations result in server upserts via sync; UI sync indicator accurate.

---

4. New Librarian Added & Installed Flow

Goal: Add a new librarian from Admin UI, then activate a separate device with that librarian and ensure shift enforcement works.

Preconditions - Server has `shifts` table and admin created a shift for the new librarian for today. - Device A (admin) has network and pushed commit creating new librarian + shift. - Device C (fresh) is ready to activate.

Steps 1. Device A (Admin) → Manage Librarians → `+ Add Librarian`. - **Input:** Username: `lib-new`, Full name: `Lib New`, Role: `librarian`. - **Expect:** UI shows temp PIN (e.g., `4638`); local commit created.

1. Admin assigns shift (today) to `lib-new` (Admin → Shifts → Create shift).
2. **Input:** date = today, start_time = now - 5 min, end_time = now + 3 hours.
3. **Expect:** shift created locally and commit present.
4. Push sync on Device A to send commits to server.
5. **Expect:** Server has `librarians` and `shifts` rows for `lib-new` and shift.
6. Device C fresh install → Activate with Username: `lib-new`, PIN: `<temp PIN>`.

7. **Expect:** Activation success, `applySnapshot` writes `shifts` locally, device bound to `lib-new`, and because `require_pin_change === true` redirect to change-pin, after PIN change login continues.

8. Attempt local login on Device C (after PIN change) *during* the scheduled shift window.

9. **Input:** Username `lib-new`, PIN `newPin`

10. **Expect:** `isInsideShift('lib-new')` returns `true` → Login allowed; redirected to librarian dashboard.

11. Sign out → adjust the device clock or admin change shift so the current time is outside shift window. Attempt local login again.

12. **Input:** same credentials
13. **Expect:** Login blocked with alert `❌ You are outside your scheduled shift time.` for librarians; Admin users still allowed.

Edge cases - If shift is not yet propagated or server push failed, activation should still succeed if snapshot provided. If shifts absent, login will be blocked (this case should be tested and admin should be able to override).

Pass criteria - New librarian activation binds device and saves shift locally. - Shift check blocks/permits login correctly.

---

5. Day-to-Day Librarian Flow

Goal: Typical daily tasks: login during shift, borrow a book, return, and ensure commits get queued and pushed.

Preconditions - Librarian device is activated and logged in (inside shift window). - Books and users exist in local DB.

Steps 1. Login as `lib1` during shift window. - **Input:** Username `lib1`, PIN - **Expect:** Login success and landing on librarian dashboard.

1. Borrow flow: navigate to `Borrow` → `Scan user` → `Scan book` → `Confirm`.
2. **Input:** select user `user-1`, book `book-1` (available copies > 0)
3. **Expect:** Local `transactions` row inserted with `type=borrow`, `sync_status='pending'` and a `pending_commits` entry recorded. UI shows success message.

4. Use FAB to `Sync` (or manual Sync) at end of shift.

5. **Expect:** `pushPendingCommits()` pushes the commit(s) to server; `pending_commits` marked synced; `sync_log` entry shows success; server `transactions` table contains new tx.

6. Return flow: navigate to `Return` → scan book → confirm.

7. **Expect:** Local `transactions` updated with `returned_at` and commit queued.

8. Offline behavior: Disable network and borrow another book.

9. **Expect:** Local commit created and persisted; UI allows operation; later when network back and sync triggered, commit pushes successfully.

Edge / Negative Cases - Borrow a book with 0 copies → UI blocks and shows `No available copies`. - Attempt action outside shift window → login prevented and no operations allowed.

Pass criteria - Commits created locally for all operations and pushed successfully when network present. - Librarian cannot login outside shift window.

Cross-cutting Tests & Observability

A. Device Binding Validation - Ensure `meta.device_id` stores device id after activation and `librarians.device_id` for activating user is equal.

B. Sync Log & Recovery - Simulate push failure (turn off server) → `insertSyncLog` shows a failed record. Bring server back → `pushPendingCommits()` reattempts and succeeds; log shows success.

C. Security - PIN hashed & salted stored; raw PINs never stored. - Changing PIN updates `pin_salt` & `pin_hash`.

D. Race Conditions - Concurrent pushes from multiple devices: server side should accept idempotent commits and avoid duplicates. Validate server `commits` table for duplicates.

E. Timezone / Date handling - Ensure shift date/time comparisons are based on local device time (document requirement) and format `YYYY-MM-DD` / `HH:mm`.

Test data examples

- Admin: `DiguwaSoft / 1366` (temp) → change to `2468`
- Demo librarian: `lib1 / 0000` (temp) → change to `4567`
- Book: `book-1` copies = 3
- User: `user-1` (fayda_id)
- Shift: today `2025-12-11`, start `08:00`, end `16:00`

Reporting & Sign-off

- Each test step should be marked `PASS / FAIL` and include timestamp, device id, logs excerpt (local migration logs, sync logs) and snapshots (server & local) when relevant.
- For any failure, include reproduction steps, DB snapshots (server and local), and `adb logcat` or console logs.

End of Test Cases

If you'd like, I can (A) generate an executable checklist CSV, (B) produce a brief test-run script for testers, or (C) convert this into a test-case management import format.