## **Landline Mobile Application Design (Android)**

## **Overview**

## The purpose of this document is to outline project goals, development strategy, and technical decisions for the Landline Mobile Application.

## The audience includes both technical and non-technical stakeholders. Developers will use this document to understand the technical requirements and development environment, in order to implement future stretch goals. Business stakeholders can refer to this document to grasp the customer-facing features and expected outcomes. Readers should have a basic understanding of mobile application development and familiarity with modern smartphones.

**Problem**

The Landline Mobile Application is designed to address a common negative effect of smart-phone use: an expectation that the user interact with and respond to multiple sources of notifications, communications, and other alerts that often times results in increased stress, anxiety, a distraction from other activities the user values, and a myriad of other negative effects. Furthermore, it has become common place for people to have expectations that they receive a response within a timeframe that the user cannot, or chooses not to respond within, which can often lead to detrimental effect between those sending and receiving the communication, or an interpretation by the sender that the recipient hasn’t responded due to an accident or other mal-event.

Additionally, while an easy solution to avoid smart-phone distractions would be to turn the device off during periods the user would like to avoid them, this is in direct conflict with one of the benefits users expect from the devices in that turning the phone off prevents loved ones or others from communicating with the user should an emergency arise. As landline use continues to dwindle, this pressure to keep smart-phones operating continuously will increase.

A third proposed solution to diminish notification and alert distractions would be to update each application on the smart-phone to turn off notifications, however this solution doesn’t address the root of the problem, ignores the social ramifications outlined above in doing so, and removes a feature that a user might want, albeit on their own terms instead of the device’s or the senders. The notifications themselves may be useful, turning them off entirely diminishes the value the smart-phone provides, and it would require repetitive, manual time spent to turn them all on/off whenever the user wanted a reprieve.

**Requirements**

The application is designed to provide users with a tool for people to disengage from social media and their phones without feeling obligated to respond immediately, similar to an old-fashioned landline. The app would switch a user's device into "landline mode," diverting notifications and allowing only a few main contacts (like family) to reach them in emergencies. The app would log all notifications, allowing users to review them later by type (i.e. text, email, voice-mail, app). The app would also send targeted "out of office" replies to selected senders (possibly filtered by the user’s contacts, or a selectable subset of contacts), indicating the user is on landline and will respond later. An emergency override feature would allow loved ones and other pre-selected contacts to use a unique contact pin to break through the landline filter and notify the user in urgent situations. The app would also be able to distinguish and allow mass government alerts.

## **Use Cases**

**RED**: NOT possible on iOS

1. Initial Setup
   1. The app should require the user register with the phone number and email addresses to be used with the app, verify themselves via a required response via text and email to each registered address, and accept the terms of use prior to any further setup and operation. Each verification step should include accepting the Terms of Use individually for each phone number or email address they register with the app.
   2. The app should go through a sequence of events that makes it easy to select which phone app’s notifications should be included in “Landline” mode.
   3. The app should present a list of installed apps with the ability to select each individually, or “Select All”
   4. The app should prompt the user to select a set of contacts and email addresses from their contact list that the user would like to provide “I’m on Landline…” out-of-office notifications to, either individually, or “Select All.”
   5. The app should prompt the user to select which contacts and email addresses have “Emergency Access.”
   6. Future versions – The app should offer a variety of style options for the appearance of the log (e.g. rolodex, address book, dark or light modes, different color schemes.)
2. Operation – User Interactions
   1. The user should have a simple sequence to activate “Landline.” A click of an icon on the home screen, answers a question verifying “Would you like to turn on Landline?” and it activates upon the user selecting “Yes”.
   2. While Landline mode is active, an icon should offer the user to select it and open a window to review the current log of notifications.
   3. The user should be able to select a category of notifications to review while the app is still active (i.e. Texts, Emails, Phone Calls, Applications.)
   4. The user should have a simple, one-click button to “Clear” the notifications from the log while the app is still active, resetting the count shown in the log.
   5. The user should have a simple, one-click button within the app to turn off, “Landline.”
3. Operation – Application Active, Automated (not requiring user interaction.)
   1. The app should divert notifications to the log.
   2. The app should turn off notifications on the phone’s screen.
   3. The app should set the phone’s ringer to full silent mode (no vibe.)
   4. When an email is received from a contact selected to receive an out-of-office reply, an email should be sent in response notifying the sender the user is on Landline.
   5. When a text message is received from a contact selected to receive an out-of-office reply, a text should be sent in response notifying the sender the user is on Landline.
   6. When a phone call is received from a contact selected to receive an out-of-office reply, a text should be sent in response notifying the sender the user is on Landline.
   7. …
4. **Terms Agreement:**
   * Prior to first use, the application shall display an agreement to the customer informing them of the terms and conditions of the application's blocking notifications prior to allowing them to use the app. The customer should accept the terms and conditions that they are responsible for any negative repercussions of missing a notification or the delivery of an “out-of-office” message, and hold the app developer free from any liability in perpetuity.
5. **Notification Log UI:**
   * The app shall include a primary UI that displays the number of received notifications, categorized by type.
6. **Notification Log Categories:**
   * The app shall include a category for each: Texts, Emails, Missed Calls, Voicemails, App Notifications
7. **User-Friendly Interface:**
   * The application should be simple, intuitive, and easy to use.

## **Success Criteria**

For the success criteria of the Landline Mobile Application, we want to outline measurable goals that will determine whether the app meets both business and customer needs. These criteria focus on key performance indicators (KPIs) such as user satisfaction, app performance, and operational efficiency. Here’s how you can evaluate the success of the product:

#### **User Adoption and Engagement**

#### 60% of users enable Landline Mode at least once within their first week.

#### Onboarding completion rate above 85% (includes verification and setup steps).

#### **Functionality and Reliability**

* 100% of notifications successfully logged and categorized while Landline Mode is active.
* Emergency override feature successfully bypasses Landline Mode.
* Automated out-of-office replies triggered and sent correctly.
* No critical bugs or crashes reported by more than 2% of users during the first 3 months.

#### **Data Management and Logging**

* 100% of incoming notifications are timestamped, categorized, and stored accurately in the local notification log while Landline Mode is active.
* Users can retrieve and view categorized logs (Texts, Emails, Calls, Voicemails, App Notifications) with zero data loss or corruption.
* All logged data is encrypted at rest and in transit, with appropriate access control to prevent unauthorized viewing or modification.
* Application provides clear and accessible options for users to view, delete, or reset their notification logs in compliance with data privacy standards.

#### **Privacy, Security, and Compliance**

* All personal data (contact lists, notification logs, etc.) stored and handled in compliance with GDPR and CCPA.
* No unauthorized access or data leakage incidents reported during the first 6 months.
* Secure and verifiable terms of use acceptance for all registered phone numbers and emails.

#### **User Satisfaction and Usability**

* Average app store rating of 4.2+ within the first 6 months.
* At least 80% of surveyed users report reduced stress or distraction from using Landline Mode.

# **Technology Stack**

| **Architecture Component** | **Technology Choice** | **Justification** |
| --- | --- | --- |
| Mobile Application Frontend | Kotlin, Jetpack Compose,  Material 3 | Modern, type-safe Android language with a declarative UI framework. Jetpack Compose provides fast UI rendering, easy theming, and reactive state management for a smooth “Landline” mode interface. |
| Notification Logging | NotificationListenerService | Enables background capture and categorization of notifications from other apps (with permission), forming the core of the Landline mode log. |
| Navigation | Navigation-Compose | Simple, type-safe navigation and back-stack handling. |
| Dependency Logging | Hilt | Scalable DI with minimal boilerplate and good tooling support. |
| DND/Silent Control | NotificationManager (ACCESS\_NOTIFICATION\_POLICY), AudioManager | Programmatically enable DND or silence ringer during Landline mode. |
| SMS Auto-reply | BroadcastReceiver (RECEIVE\_SMS), SmsManager, RoleManager (ROLE\_SMS) | Detects inbound SMS and sends rule-based auto-replies; default SMS role improves reliability. |
| Call Detect → Auto Text | TelephonyCallback / PhoneStateListener, CallScreeningService | Observes incoming/missed calls and triggers auto-reply rules. |
| Contacts & Selection UI | ContactsContract, Permission APIs | Lets users pick emergency/OOO contacts with clear runtime permissions. |
| **\*** Installed Apps Inclusion | PackageManager (prefer derive from observed notifications) | Supports include/exclude rules; avoid QUERY\_ALL\_PACKAGES by learning from notifications. |
| Local Data | Room (SQLite) | Durable, queryable on-device store for logs/rules with migrations. Makes it easy to run queries like: *"Show me all missed calls in the last Landline session"* |
| Preferences | DataStore (Proto/Prefs) | Type-safe, async settings (replaces SharedPreferences). |
| Security (On-Device) | Jetpack Security (EncryptedSharedPreferences / EncryptedFile), optional SQLCipher for Room | Encrypts sensitive settings and exported files; option to encrypt DB rows. |
| Background Execution | Foreground Service, WorkManager | Reliable “Landline mode” runtime plus deferrable/retriable jobs (cleanup, exports). |
| Analytics & Stability | Firebase Analytics, Crashlytics | Usage funnels, crash traces, and stability KPIs. |

# **\* PackageManager**: an Android API that lets you query info about installed apps (package name, app label, icon, etc.).

# **Why it’s in Landline**: You want users to include or exclude certain apps from Landline mode. For example, maybe allow notifications from WhatsApp but block Instagram.

# **Privacy Issue:** Android 11+ restricts listing all installed apps unless you declare QUERY\_ALL\_PACKAGES in your manifest. This is **heavily scrutinized by Google Play.**

# **Alternative Approach**: Instead of listing every app, learn which apps to show based on observed notifications from NotificationListenerService. That way you only know about apps the user has actually received notifications from — much easier to justify in Play Store review.

**Backend/Cloud (\* ONLY NECESSARY IF EMAIL REPLIES INCLUDED IN MVP)**

**FastAPI (Python) or NestJS (Node.js)**

* **Purpose:** Serves as the backend API layer, handling user account linking, Landline rules synchronization, and device management.
* **Advantages:** Lightweight, high-performance frameworks with strong typing (NestJS) or concise syntax (FastAPI) that make API development and maintenance faster. Easily integrates with authentication, queues, and cloud hosting services.

**OAuth 2.0 (Gmail API, Microsoft Graph)**

* **Purpose:** Enables secure linking of user email accounts for sending out-of-office auto-replies from the server while Landline mode is active.
* **Advantages:** Uses industry-standard authentication, ensuring security and compliance with provider policies while avoiding the need to store user passwords.

**Worker Queue + Gmail/Graph SDKs**

* **Purpose:** Processes and sends email auto-replies asynchronously while respecting provider rate limits and user-defined rules.
* **Advantages:** Decouples email sending from the main API, improving reliability, scalability, and fault tolerance.

**Firebase Cloud Messaging (FCM)**

* **Purpose:** Delivers push notifications to Android devices for Landline status changes, emergency override alerts, and remote toggles.
* **Advantages:** Scalable, reliable messaging infrastructure maintained by Google; supports priority delivery for urgent alerts.

**PostgreSQL**

* **Purpose:** Stores persistent data, including user profiles, linked accounts, rules, and audit logs.
* **Advantages:** Reliable relational database with strong ACID compliance and broad cloud hosting support.

**Redis + BullMQ (Node) or Celery (Python)**

* **Purpose:** Handles asynchronous jobs, rate-limiting, and retries for outbound auto-replies and scheduled tasks.
* **Advantages:** Improves performance by offloading long-running tasks; ensures reliable delivery with retry logic.

**Cloud KMS or HashiCorp Vault**

* **Purpose:** Secures sensitive credentials such as OAuth client secrets, API keys, and signing material.
* **Advantages:** Centralized secret management with strong encryption, audit logging, and rotation capabilities.

**OpenTelemetry + Prometheus/Grafana (or ELK)**

* **Purpose:** Collects metrics, logs, and traces to monitor backend performance and diagnose issues quickly.
* **Advantages:** Improves system reliability and uptime through real-time observability.

**Firebase Remote Config (or LaunchDarkly)**

* **Purpose:** Manages feature flags for gradual rollouts, cohort testing, and emergency feature disablement.
* **Advantages:** Allows non-disruptive deployment of changes and safer experimentation without releasing new app builds.

**Privacy & Compliance Framework**

* **Purpose:** Documents data flows for GDPR/CCPA, enforces retention policies, and provides user data export/deletion endpoints.
* **Advantages:** Reduces legal risk and increases user trust by ensuring transparent, compliant data handling.

# **Architecture**

# **High-Level Overview (HLD)**

**Activating Landline Mode**

* Toggle on/off from app
* Start/Stop Foreground Service.

**Capturing Events**

* Read posted/removed notifications (NotificationListenerService).
* Listen for SMS and call state (receivers/callbacks).

**Applying Rules**

* Include/exclude apps.
* Mark emergency contacts.
* Enable auto-reply (SMS; email via backend).

**Device Controls**

* Turn on Do Not Disturb (if granted).
* Fallback to ringer/vibrate changes if DND not granted.

**Logging & Review**

* Write events to local DB (Room).
* Show log by category (Texts, Calls, Apps).
* Clear log, filter, search.

**Privacy**

* Local encryption for sensitive data.
* Implement strong encryption, anonymization of data logs, and clear user consent policies. Adopt a “privacy by design” approach.

**Backend (for email replies & sync)**

* OAuth link to Gmail/Outlook.
* Workers send out-of-office emails while Landline is ON.
* Optional push (FCM) for status/emergency alerts.

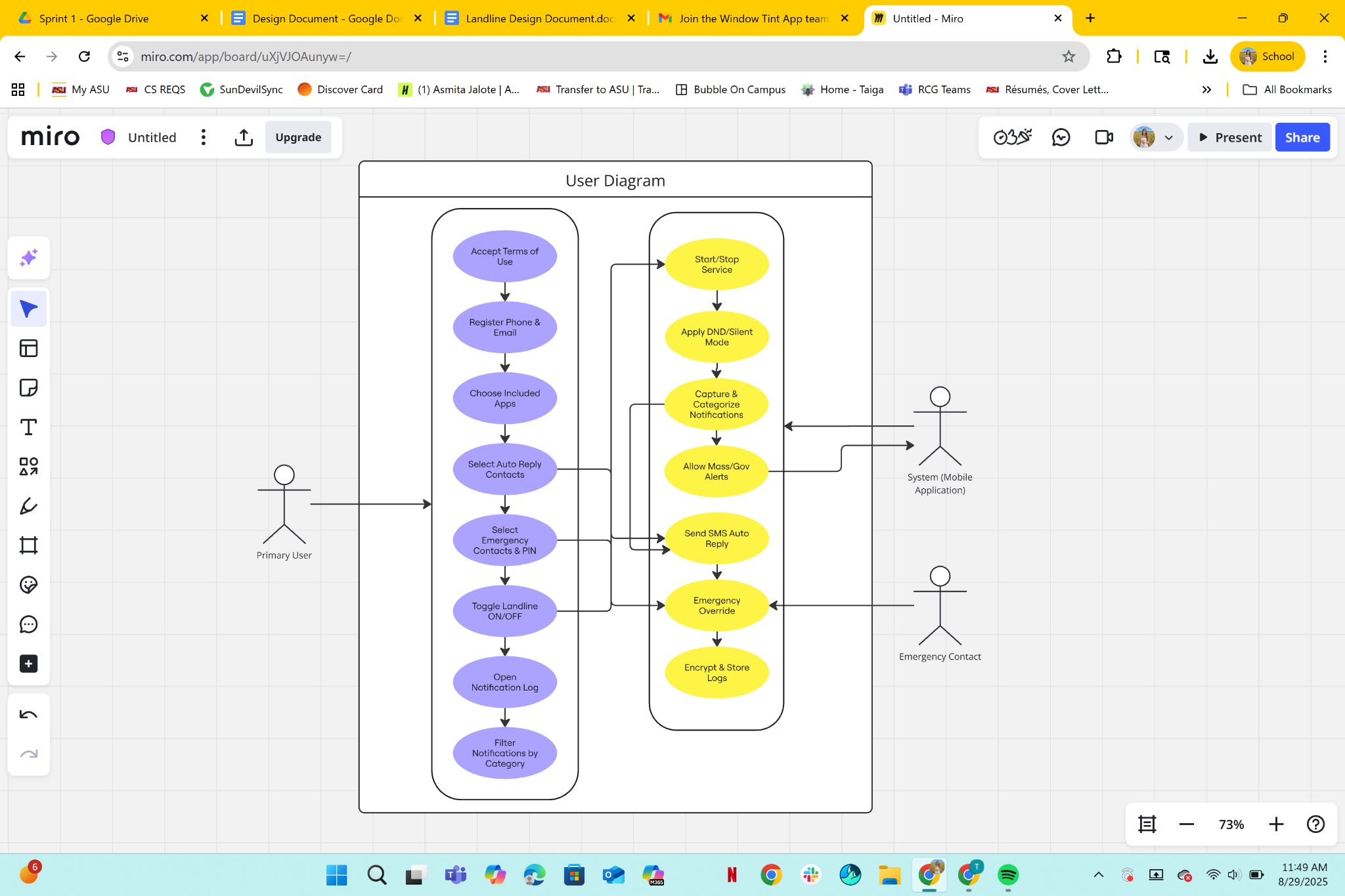
**Settings & Storage**

* Quick preferences (DataStore).
* Contact picker for OOO/emergency.

**Permissions & Safety**

* Guided permission prompts (Notifications, SMS, Calls, Contacts, DND).
* Graceful degradation if a permission is denied.

**User Diagram**

****

**Non-Functional Requirements**

**Performance**

* The app should open in under 5 seconds.
* Turning Landline Mode on/off should feel instant.
* Battery drain should be minimal while running in the background.

**Reliability**

* Landline Mode should keep running unless the user turns it off.
* Logs should not be lost if the app closes unexpectedly.

**Maintainability**

* Code should be split into clear parts (UI, logic, data).
* Use comments and simple documentation so others can understand it.
* Keep dependencies minimal to make updates easier.

**Security & Privacy**

* Only request permissions the app really needs (Notifications, SMS, Calls, Contacts, DND).
* Store data locally on the device.
* Let users clear their logs anytime.

**Usability**

* Easy-to-understand onboarding and setup.
* Landline toggle should be obvious and simple to find.
* Log view should be easy to read and navigate.

**Compatibility**

* Works on Android 8.0 (API 26) and newer.
* Test on at least two different Android phone models.

# **Testing and Observability**

### **Testing Strategy**

To ensure the Landline Mobile Application functions smoothly and reliably, a range of testing strategies will be implemented, from development to production stages.

## **Goals (MVP-level)**

* Catch obvious crashes/regressions before release.
* Prove Landline Mode runs reliably with minimal battery drain.
* Verify the tricky bits: permissions, DND access, SMS/call auto-reply, notification logging.

## **Test Types**

* **Unit tests (logic only):** JUnit + MockK (or Mockito).
  + Target: core use-cases (toggle Landline, rule evaluation).
  + Aim for ~40–50% coverage on domain layer.
* **Instrumentation/UI tests:** Compose UI Test (or Espresso if needed).
  + Verify: toggle flow, log screen pagination, clear log, settings.
* **Manual scenario tests:** A short checklist each sprint on 2 devices.
  + Focus: permissions flows, DND grant/deny, SMS role granted/denied, incoming SMS/Call behavior, notification log accuracy.
* **Device/battery sanity:** Run the app for 30–60 min in Landline Mode and confirm no obvious overheating and acceptable battery usage (informal check via system battery screen).
* **Play-policy checks:** Confirm only required permissions are requested and every permission has an in-app rationale screen.

## **Minimal Device Matrix**

* **OS:** API 26–27 (Oreo) & API 33+ (Android 13).
* **Vendors:** 1 Pixel + 1 Samsung (or any two distinct OEMs).

## **High-Value Test Scenarios (Manual)**

1. **Onboarding Path:** grant Notifications → Contacts → DND → SMS role.
2. **Onboarding With Denials:** deny DND; confirm app falls back / warns politely.
3. **Landline ON/OFF:** toggle on → foreground service visible → toggle off → service stops.
4. **Notifications:** receive a few app notifications; confirm they appear in the log with app label.
5. **Include/Exclude App:** exclude an app and confirm its future notifications don’t appear.
6. **SMS Auto-Reply:** receive SMS from allowed contact; verify single auto-reply is sent; rate-limit further replies.
7. **Incoming Call → auto text:** phone rings; after miss/end, one rule-based SMS is sent.
8. **Emergency Contact:** mark contact as emergency; confirm behavior (e.g., bypass rules/extra alert).
9. **Crash Resilience:** force-stop app; after reopen, Landline state restores correctly.
10. **Reboot Device:** after unlock, if Landline was ON, it resumes within a few seconds.
11. **Accessibility:** large text enabled; TalkBack reads main controls.
12. **Privacy:** clear logs; confirm data is gone.

**Concerns / Risks**

To ensure the successful deployment and operation of the Landline Mobile Application, several potential concerns and risks have been identified. Each risk is accompanied by mitigation strategies to reduce its impact on the application and its users.

**Play Store Policy Rejection (SMS/Call/DND)**

* **Risk:** Google Play tightly restricts RECEIVE\_SMS, SEND\_SMS, READ\_PHONE\_STATE, and DND access. Apps may be rejected if auto-reply isn’t core or insufficiently justified.
  + **Mitigation:** Make auto-reply a core, user-visible feature; add clear in-app rationale screens; consider becoming **default SMS app** for full reliability; document use in Play Console; feature-flag risky capabilities.

**Permission Friction & User Drop-Off**

* **Risk:** Users deny notifications/contacts/DND permissions → app feels broken.
  + **Mitigation:** Progressive onboarding, per-permission “Why” screens, graceful fallbacks (e.g., run without DND, show warning), “Try Again” prompts.

**Auto-reply Abuse / Spam Loops**

* **Risk:** Reply storms, reply-to-bots, or escalating message counts → carrier flags or user complaints.
  + **Mitigation:** Per-contact rate limit (e.g., 1 reply / 12h), global daily caps, “only reply to contacts” option, preview text, opt-out list.

**Emergency Expectations & Liability**

* **Risk:** Users assume Landline will always allow urgent messages through; missed alerts could be blamed on the app.
  + **Mitigation:** Clear disclaimers; explain limits (OS/permissions, carrier, sender behavior); visible “Emergency Contacts” setup; session banner showing current state; exportable logs for support.

**Installed-apps Inclusion (Privacy Policy)**

* **Risk:** Querying all installed apps (QUERY\_ALL\_PACKAGES) can trigger review issues.
  + **Mitigation:** Derive app list from observed notifications by default; avoid broad queries; justify any targeted lookups.

**Backend Complexity (only if email replies included)**

* **Risk:** OAuth flows, token storage, rate limits, provider policy changes.
  + **Mitigation:** Start without email in MVP; if included, keep a minimal API + worker; store tokens with KMS; retries and rate-limits; feature-flag email.

**Testing Coverage**

### **Risk:** Limited device matrix → regressions on certain OEMs.

### **Mitigation:** Test on at least one Samsung + one Pixel; keep a case manual checklist (onboarding, DND, SMS/calls).

# **Future Improvements**

**Smarter Emergency Overrides**

* Add configurable rules (e.g., multiple missed calls from a contact = override).
* Support custom emergency keywords in SMS (e.g., “urgent”, “911”).

**Email Auto-Replies Expansion**

* Extend the backend to support Gmail/Outlook and possibly additional providers.
* Add user-friendly email templates and scheduling.

**Advanced Notification Controls**

* Per-category tuning (social, work, personal).
* Time-based schedules (auto Landline at night or during work hours).

**Improved Data Insights**

* Show usage stats (hours in Landline, notifications avoided).

**UI/UX Enhancements**

* Home screen widget and quick settings tile for faster toggling.
* More customization (themes, notification log layouts).

**Accessibility & Inclusivity**

* Voice controls for toggling Landline.
* Better TalkBack integration and large-text optimization.

**Scalability Features**

* Multi-device support with account login.