# AegisSecure: AI-Powered Threat Detection App

Sprint 2 Test Report

**Group Number: 35**
Mevada Soham Meghalkumar [202301484]
Gohil Suryadeepsinh Hardevsinh [202301463]
Rana Neelabh Vijaykumar [202301476]
Hrithik B Patel [202301441]
Vadsmiya Pransu Pradipkumar [202301445]
Dhruv Jigneshkumar Patel [202301095]
Bhagiya Jenish Rameshbhai [202301480]
Akshat Bhatt [202301460]
Vrajkumar Makwana [202301436]
Chavda Mihirsinh Labhubhai [202301479]


IT314 - Software Engineering
Prof: Saurabh Tiwari
Mentor: Shyam Patel

# Contents

# 1 Test Report Summary

- **Project:** Aegis Secure: AI-Powered Threat Detection App
- **Sprint:** Sprint 2
- **Sprint Objective:** Implement secure user registration and establish reliable email synchronization for spam/ham identification.
- **Test Type:** Manual APK (Black-Box) Testing
- **Test Focus:** Validating User Stories for Secure Registration (US-8), Manual Input (US-3), and Email Integration (US-4).

## 1.1 Test Environment

- **Application:** Aegis Secure
- **Build Version:** Sprint 2 APK (Build Date: 24th October 2025)
- **Test Device:** *e.g., Google Pixel 6, Android 13*
- **Backend API:** `backend.onrender.com`

## 1.2 Scope

- **In Scope:**
  - US-8: Secure User Registration (Sign-up, Sign-in, Email/OTP Verification)
  - US-3: Manual Text Analysis
  - US-4: Email Integration (Google OAuth Flow & Email Fetching)
- **Out of Scope (as per Sprint 2 Report):**
  - US-5: Auto Scanning Service (Planned for Sprint 3)
  - US-4: SMS Fetching (Development paused)

## 2 Detailed Test Case Execution

The following table details the test cases executed based on the Sprint 2 features.

| Test ID | Feature | Test Scenario | Expected Result | Status |
|---|---|---|---|---|
| TC-001 | US-8: Sign Up | User attempts to sign up without checking "I agree to the Terms and Privacy Policy." | The app should prevent sign-up and display an error: "Please agree to the terms & conditions." | **PASS** |
| *Evidence:* | See Appendix, Figure 1a | | | |
| TC-002 | US-8: Sign In | User attempts to sign in with a correct email but an incorrect password. | The app should prevent sign-in and display an error: "Invalid email or password." | **PASS** |
| *Evidence:* | See Appendix, Figure 1b | | | |
| TC-003 | US-8: OTP Verify | User enters an invalid or expired 6-digit OTP code during email verification. | The app should prevent verification and display an error: "Invalid or expired OTP. Please try again." | **PASS** |
| *Evidence:* | See Appendix, Figure 1c | | | |
| TC-004 | US-8: OTP Verify | User successfully enters a valid OTP. | The app should show a success message ("OTP verified successfully!") and redirect to the sign-in page. | **PASS** |
| *Evidence:* | See Appendix, Figure 2a | | | |
| TC-005 | US-3: Manual Scan | User inputs a high-confidence spam message ("You won $1,000,000!...") and submits for analysis. | The app should correctly classify the text as "SPAM" with a high confidence score (e.g., 1.00). | **PASS** |
| *Evidence:* | See Appendix, Figure 2b | | | |
| TC-006 | US-3: Manual Scan | User inputs a message that is likely spam ("Reply to this message to check your CIBIL score.") and submits. | The app should classify the text as "SPAM" with a corresponding confidence score (e.g., 0.67). | **PASS** |
| *Evidence:* | See Appendix, Figure 2c | | | |
| TC-007 | US-4: OAuth | User successfully completes the Google OAuth flow to grant email permission. | The backend should return a success status message (e.g., JSON with `"status":"success"`). | **PASS** |
| *Evidence:* | See Appendix, Figure 3a | | | |

| Test ID | Feature | Test Scenario | Expected Result | Status |
|---|---|---|---|---|
| TC-008 | US-4: OAuth | User cancels the Google OAuth flow or an error occurs (e.g., user hits "Cancel"). | **Expected:** The app should catch the error and show a user-friendly message (e.g., "Email linking cancelled."). **Actual:** The app redirects to a raw JSON error from the backend: `"detail":[...]"msg":"Field required"`. | **FAIL** |
| *Evidence:* | See Appendix, Figure 3b | | | |
| TC-009 | UI/UX | User rotates the screen from portrait to landscape mode after entering data in fields. | The UI should adapt, and any text entered in the "Email" or "Password" fields should be retained. | **PASS** |
| *Evidence:* | See Appendix, Figure 3c | | | |
| TC-010 | US-4: OAuth | User is presented with the Google permission screen during the email-linking flow. | The app correctly identifies the backend (`aegissecurebackend.onrender.com`) as the requester for the user's Google Account. | **PASS** |
| *Evidence:* | See Appendix, Figure 4a | | | |
| TC-011 | US-4: Email Sync | User opens the app's Inbox after a successful OAuth sync. | The app successfully fetches and displays emails in the custom inbox, showing the correct computed spam confidence score for each message. | **PASS** |
| *Evidence:* | See Appendix, Figure 4b | | | |
| TC-012 | US-8: Registration | User attempts to sign up with an email that is already registered. | The app should prevent sign-up and display an error: "Registration failed, please try again." | **PASS** |
| *Evidence:* | See Appendix, Figure 4c | | | |
| TC-013 | US-8: OTP Verify | User clicks the "Resend" button to get a new OTP. | A new, separate OTP email is successfully sent to the user's inbox. | **PASS** |
| *Evidence:* | See Appendix, Figure 4b (Shows multiple OTP emails) | | | |
| TC-014 | US-8: OTP Verify | User waits 5 minutes for the OTP to expire and then enters it. | The app rejects the expired code and displays the "Invalid or expired OTP..." error. | **PASS** |

| Test ID | Feature | Test Scenario | Expected Result | Status |
|---|---|---|---|---|
| *Evidence:* | See Appendix, Figure 1c (Shows "Invalid or expired OTP...") | | | |
| TC-015 | US-3: Manual Input | User opens the manual analysis dialog and does not enter any text. | The "Submit" button does not do anything if its an empty string and hence the user cannot submit an empty string. | **PASS** |
| *Evidence:* | See Appendix, Figure 5a | | | |
| TC-016 | US-3: Manual Input | User submits a normal, non-spam ("HAM") message: "Hello, let's meet tomorrow at 5." | The app should correctly classify the text as "HAM" with a low confidence score. | **PASS** |
| *Evidence:* | See Appendix, Figure 5b | | | |
| TC-017 | US-3: Stress Test | User submits a very long "HAM" message. | The app should not crash and should correctly classify the text as "HAM" with a low confidence score. | **PASS** |
| *Evidence:* | See Appendix, Figure 5c | | | |
| TC-018 | US-3: Stress Test | User submits a very long "SPAM" message. | The app should not crash and should correctly classify the text as "SPAM" with a high confidence score. | **PASS** |
| *Evidence:* | See Appendix, Figure 6a | | | |
| TC-019 | US-3: Model Accuracy | User submits a normal "HAM" message: "congratulations for getting a new job..." | **Expected:** App should classify as "HAM". **Actual:** App incorrectly classifies as "SPAM" with high confidence (0.95). | **FAIL** |
| *Evidence:* | See Appendix, Figure 6b | | | |
| TC-020 | US-3: Model Accuracy | User submits another normal "HAM" message: "I just found out you got a new job..." | **Expected:** App should classify as "HAM". **Actual:** App incorrectly classifies as "SPAM" with high confidence (0.88). | **FAIL** |
| *Evidence:* | See Appendix, Figure 6c | | | |

| Test ID | Feature | Test Scenario | Expected Result | Status |
|---|---|---|---|---|
| TC-021 | US-8: Registration | Password Strength | User attempts to register using weak passwords (examples tested: "12345", "tooshort", "no-number"). **Expected:** The app should reject weak passwords and display an inline validation error describing the password requirements. **Actual:** The app accepts weak passwords and allows registration. | **FAIL** |

# 3 Summary of Findings (Bugs)

This testing identified 2 key bugs that should be addressed in the next sprint.

## 3.1 BUG-001: (TC-008) Unhandled OAuth Cancellation

- **Severity:** Major
- **Description:** When a user cancels the Google OAuth flow, they are redirected to a raw backend JSON error page instead of a user-friendly screen within the app. This is poor error handling and breaks the user experience.
- **Recommendation:** The app's frontend should catch the redirect error from the web view and display an in-app message, then return the user to the previous screen.

## 3.2 BUG-002: (TC-019, TC-020) Model False Positive

- **Severity:** Critical
- **Description:** The spam detection model incorrectly classifies normal, harmless messages ("HAM") as "SPAM". As shown in the evidence, messages about "congratulations" and "getting a new job" were flagged as spam with high confidence.
- **Recommendation:** This is a critical bug. The model needs to be retrained with a more diverse dataset that includes more examples of celebratory or casual "HAM" messages to reduce these false positives.

# 4 Conclusion

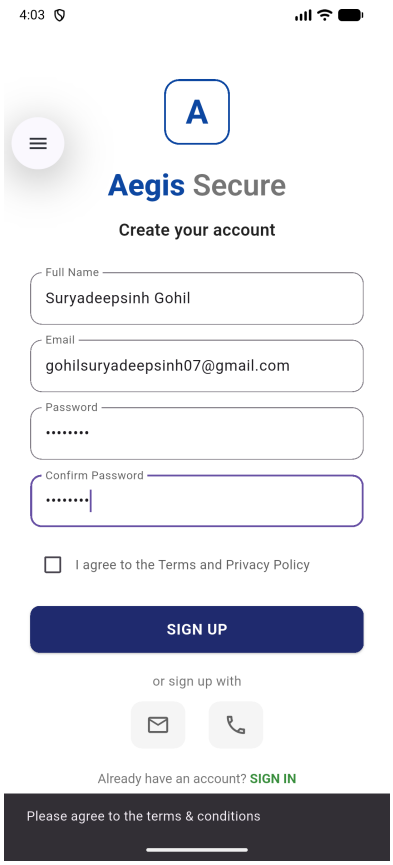
Sprint 2 validated the core functionality for Secure Registration, Manual Scan, and Email Synchronization. The primary sprint objective of establishing reliable email synchronization was achieved (see TC-011), and most happy-path scenarios for the implemented user stories passed during manual testing.

However, testing also uncovered several high-impact issues that affect user experience and security:

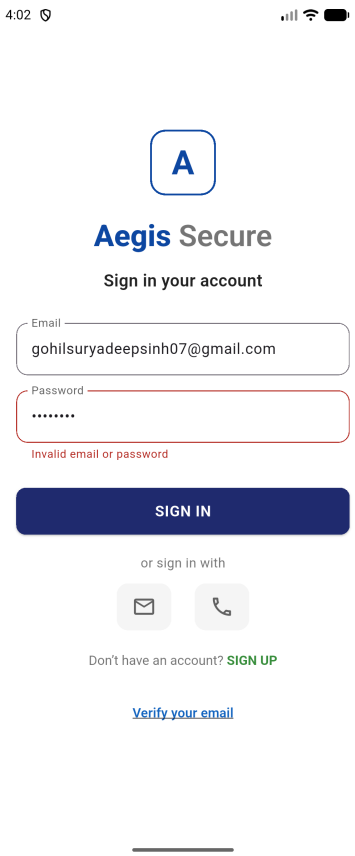
- **BUG-001 (TC-008) — Unhandled OAuth Cancellation:** When a user cancels the Google OAuth flow the app shows raw backend JSON instead of an in-app error message, degrading the UX.
- **BUG-002 (TC-019, TC-020) — Model False Positives:** The spam-detection model produced high-confidence false positives for normal messages, indicating the model needs retraining or tuning.
- **BUG-003 (TC-021) — Weak Password Acceptance:** The registration flow currently accepts weak passwords (e.g., “12345”, “tooshort”, “no-number”), which is a security concern.

In summary, while the sprint met its principal goals and many features work along their expected paths, the issues identified above must be addressed to improve security, accuracy, and the overall user experience before considering a production release.

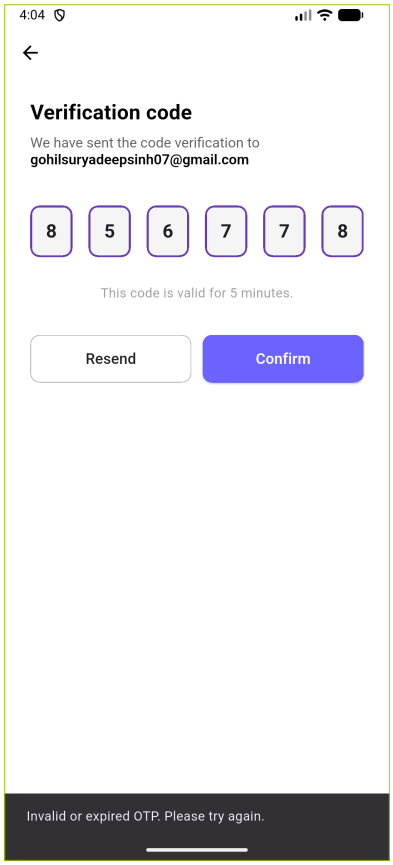
# 5 Appendix: Test Evidence



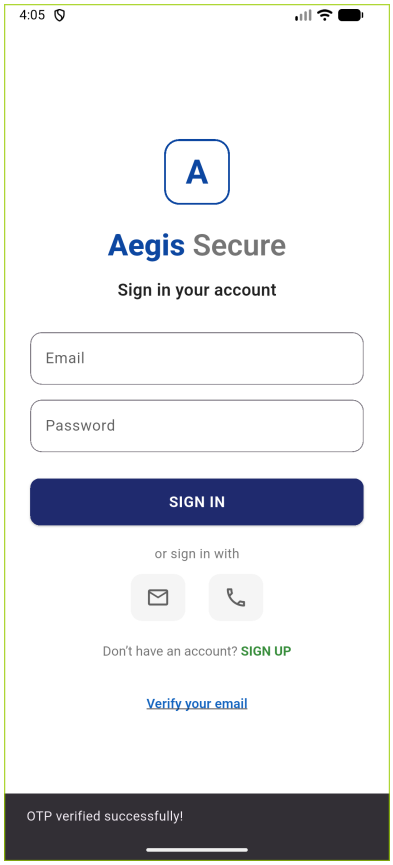

(a) Evidence for TC-001



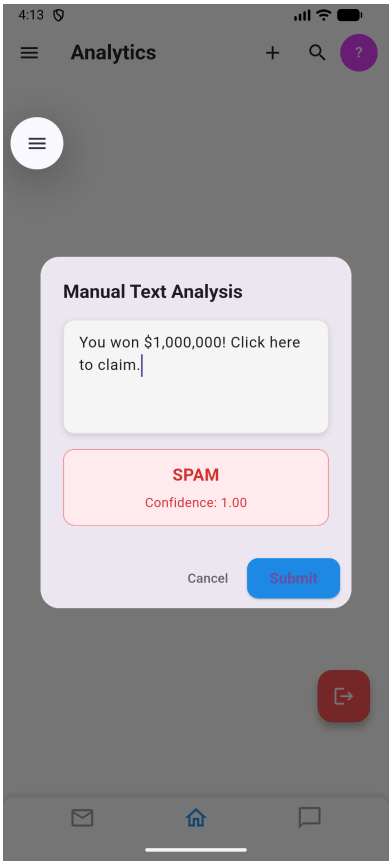

(b) Evidence for TC-002



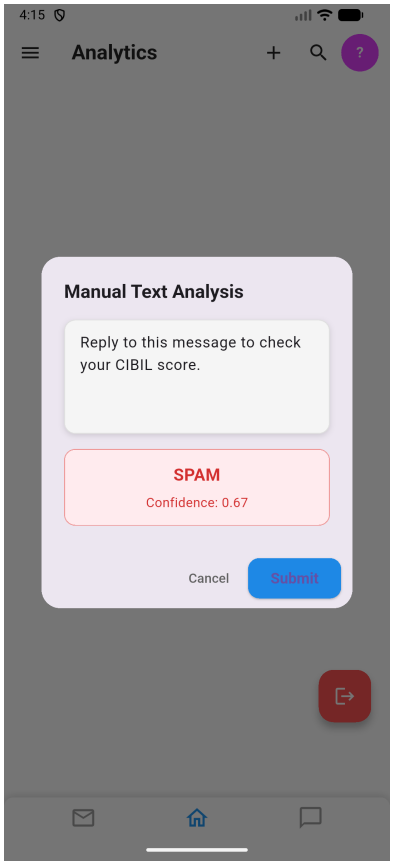

(c) Evidence for TC-003




(a) Evidence for TC-004

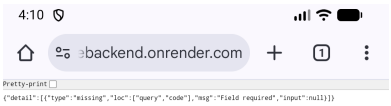



(b) Evidence for TC-005

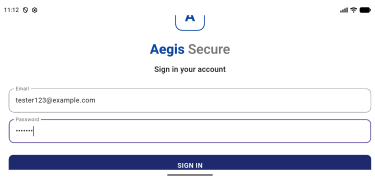



(c) Evidence for TC-006

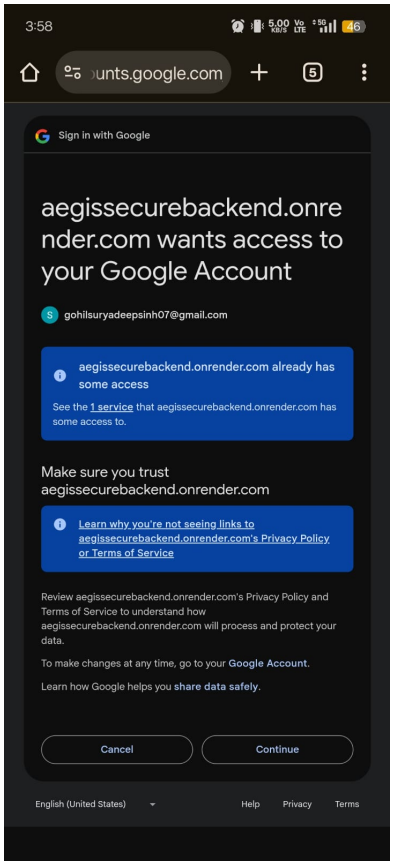

(a) Evidence for TC-007



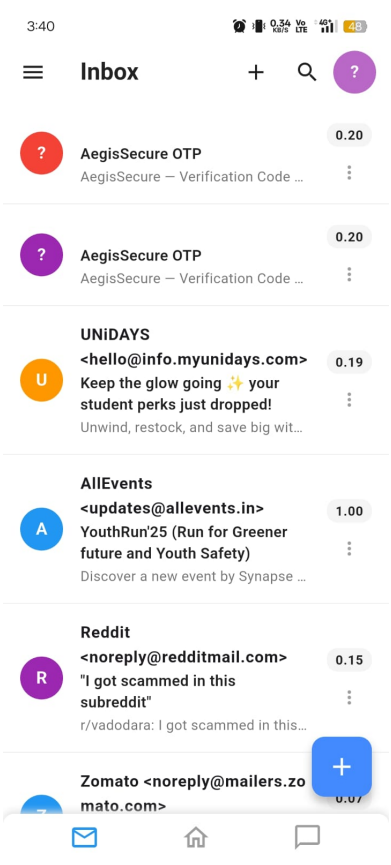

(b) Evidence for TC-008



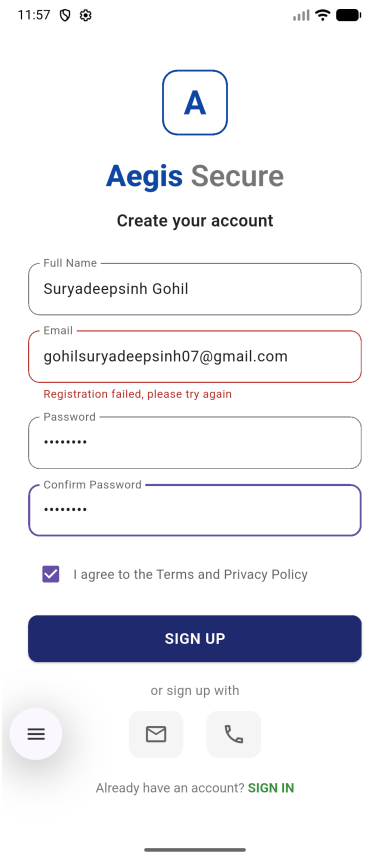

(c) Evidence for TC-009



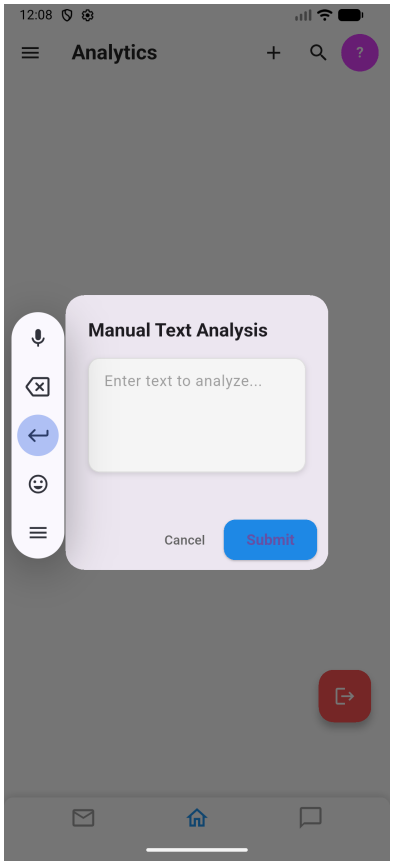

(a) Evidence for TC-010



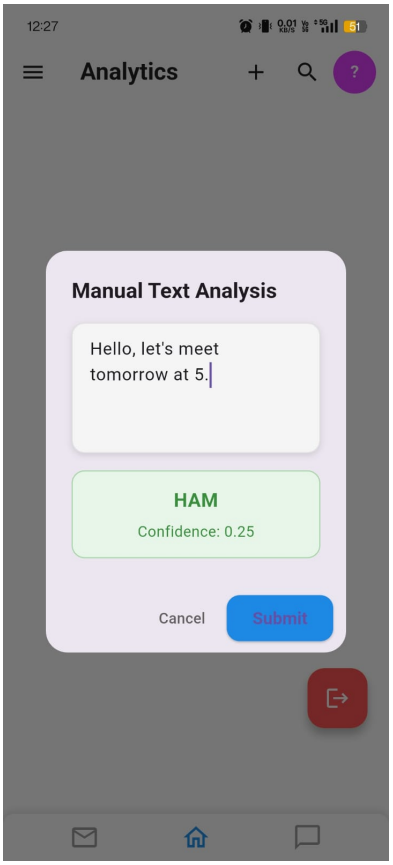

(b) Evidence for TC-011



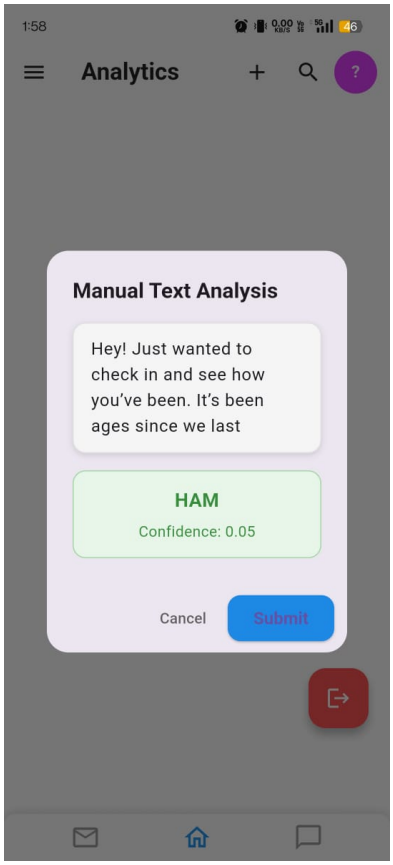

(c) Evidence for TC-012



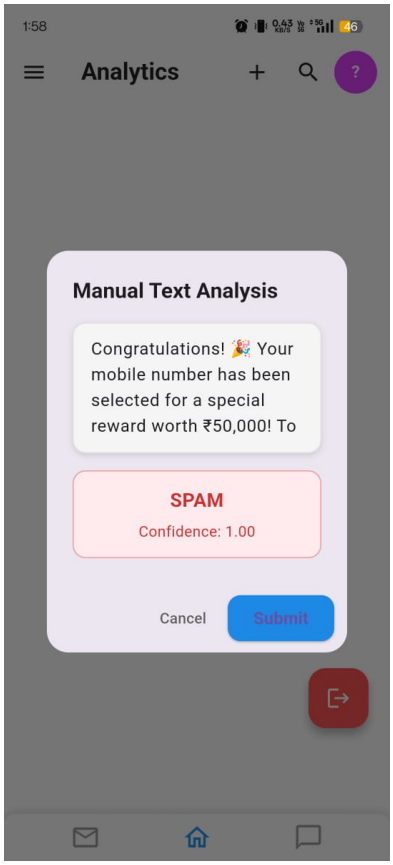

(a) Evidence for TC-015



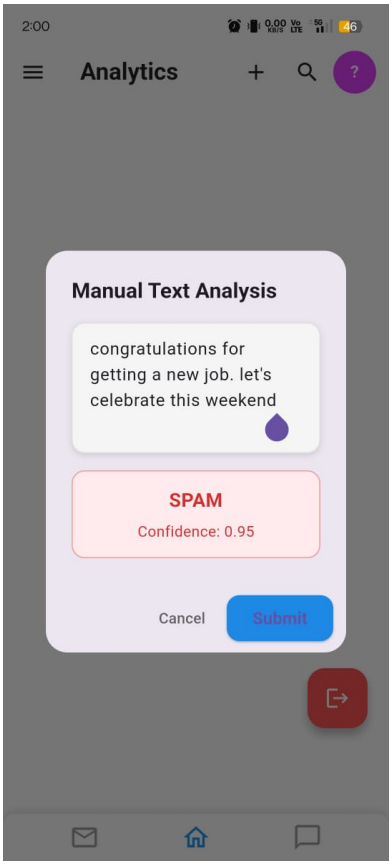

(b) Evidence for TC-016



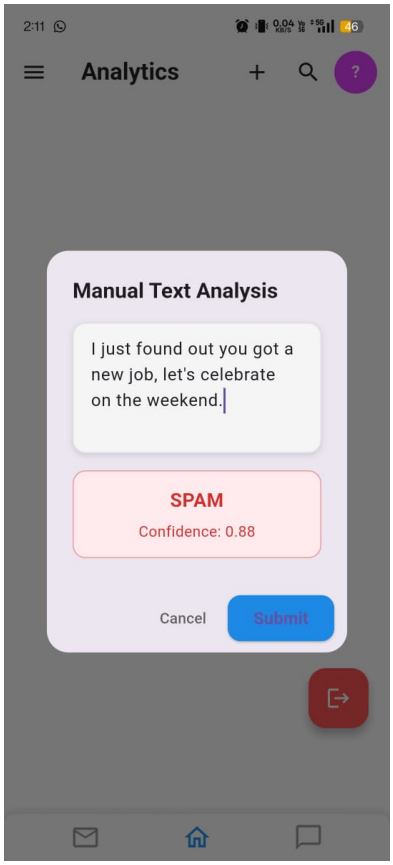

(c) Evidence for TC-017




(a) Evidence for TC-018




(b) Evidence for TC-019




(c) Evidence for TC-020