Java Assignment - 5

P.Vikrama dhanvi

AP22110010574

1. Difference between Sending SMS Programmatically vs. Using an Intent

Programmatically: Sending SMS programmatically uses APIs like SmsManager. It gives developers more control, enabling automation and background processing.

Using Intent: This launches the default SMS app with pre-filled content. It requires user interaction to send the message.

Security: Using intents is generally more secure as the user explicitly sends the message, preventing abuse. Programmatic SMS sending can be exploited if not properly managed.

2. Emergency Notification App via SMS

Design:

Maintain a contact list in SQLite or Firebase.

Use SmsManager to send messages.

Use Broadcast Receivers to track SMS_SENT and SMS_DELIVERED.

Ensure Delivery:

Use Delivery reports.

Retry logic for failed messages.

Log failed attempts and notify the admin.

3. Email via Intent vs SMTP

Intent: Opens an email app with content. Easy and user-friendly, but depends on third-party email apps.

SMTP (Programmatically): Direct email sending using libraries like JavaMail. More control, works in the background.

Drawbacks: SMTP needs secure server setup, more prone to spam filters, and requires storing credentials.

4. Email-Scheduling App

Use WorkManager or AlarmManager to schedule emails.

Use a local database for email drafts.

Use SMTP or an email API.

Recurring emails can be scheduled using CRON-like patterns in WorkManager.

5. Geocoding vs Reverse Geocoding

Geocoding: Converts address to coordinates.

Reverse Geocoding: Converts coordinates to address.

Use Cases:

Geocoding: Navigate to a known address.

Reverse: Show address from user's location.

6. Location-Based App for Hikers

Features:

Use FusedLocationProviderClient for tracking.

Set geofences for trails.

Save checkpoints in local storage.

Challenges:

Battery consumption.

GPS signal loss in remote areas.

Offline maps support.

Solutions:

Optimize location update frequency.

Use cached maps.

7. Risks of Poorly Managed Permissions

Risks:

Data leaks (e.g., location, contacts).

Background camera/microphone use.

Real-World Example:

TikTok accessing clipboard data.

Facebook's background location access issue.

8. Dynamic Permissions System

Use context-aware services:

Enable location permission only in outdoor mode.

Revoke camera access on insecure networks.

Implementation:

Use PermissionsManager and runtime checks.

ML to learn user behavior and adjust permissions.

9. Signing Android Applications

Steps:

Generate a keystore.

Sign APK using jarsigner or Android Studio.

Align using zipalign.

Importance:

Verifies publisher.

Ensures integrity.

Risks of Unsigned App:

Rejected by Play Store.

Susceptible to tampering.

10. Deployment in Rural Areas

Strategies:

Use APK instead of Play Store.

Minimize dependencies.

Offline-first approach (Room DB, caching).

Compress assets to reduce size.

11. All-in-One Alert App (SMS, Email, Location)

Security Concerns:

Encryption of location and messages.

Secure APIs for email/SMS.

Permission checks.

Prevent background abuse of location/SMS.

12. Healthcare App with Compliance

Data Privacy Measures:

Store data in encrypted form.

Use secure APIs.

Anonymize user identity.

Comply with HIPAA/GDPR.

Take consent before data collection.

This concludes the Java Assignment - 5.