International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 12, pp 51-59, December-2025

https://doi.org/10.47001/IRJIET/2025.912007

Sahayak: A Senior Citizen Mobile Assistant App

[1]Samarth P V, [2]Shivkumar Yadav, [3]Shreyas T K, [4]Prof. Manusha Reddy

[1,2,3]Dept. of CSE, Dayananda Sagar Academy of Technology and Management, Bengaluru, India

[4]Assistant Professor, Dept. of CSE, Dayananda Sagar Academy of Technology and Management, Bengaluru, India

Authors E-mail: [1]dt23cs207@dsatm.edu.in, [2]1dt23cs192@dsatm.edu.in, [3]1dt23cs201@dsatm.edu.in, [4]manusha-cse@dsatm.edu.in

***Abstract* - This research paper presents the design, development, and evaluation of "Sahayak" an innovative Android mobile application engineered to address the multifaceted challenges faced by the growing senior citizen population. In response to prevalent issues of social isolation, emergency vulnerability, and complex management of welfare services, we propose a centralized, user-centric solution built using Kotlin and Android Studio. The application implements a secure authentication framework via SharedPreferences for local session management, leading to a personalized dashboard that dynamically greets users (e.g., "Namaste, Samarth"). Its core architecture features a multi-modular interface providing immediate access to critical services: a prominent SOS and emergency contact system with direct calling to police (100) and ambulance (102), alongside configurable family contacts; dedicated modules for managing pension and life insurance information; and interactive social wellness tools (Talkie and Funzone) designed to combat loneliness and promote cognitive engagement. A key contribution of this work is the deep integration of accessibility and localization, enabling a seamless user experience across multiple Indian languages (Kannada, Telugu, Tamil, Malayalam) to bridge the digital divide. Furthermore, the application includes comprehensive user profile creation and management, capturing demographic and personal data to tailor interactions. By consolidating emergency response, health and finance tracking, and social connectivity into an intuitive, secure, and linguistically adaptable platform, "Sahayak" demonstrates a significant advancement in assistive technology, aiming to enhance the safety, autonomy, and overall quality of life for senior citizens, thereby reducing their dependency on traditional, fragmented support systems.**

Keywords: Senior citizens, mobile health application, assistive technology, emergency response system, SOS feature, digital companion, elderly care, Android application, localization, multilingual support, pension management, life insurance tracker.

I. INTRODUCTION

The global demographic landscape is undergoing a profound shift, characterized by a rapidly aging population. Typically handles dozens of online accounts, each requiring this trend presents significant societal challenges, particularly in ensuring the safety, health, and sustained well-being of senior citizens. Older adults often face a complex nexus of issues, including social isolation, limited mobility, difficulties in accessing essential services, and heightened vulnerability during emergencies.

While technology offers transformative potential to address these challenges, many existing digital solutions are fragmented, complex, or lack cultural and linguistic accessibility, leading to low adoption among the elderly. This gap underscores a critical need for integrated, intuitive, and empathetic technological interventions designed specifically with senior users' capabilities and constraints in mind.

In response, this paper presents the design and development of Sahayak, a comprehensive mobile application engineered to serve as a holistic digital companion for the elderly. The proposed system consolidates critical functionalities—including an immediate emergency response (SOS) mechanism with direct contact to services and family, modules for managing pension and insurance information, and interactive features to foster social engagement and mental stimulation—into a single, user-friendly platform.

A cornerstone of its design is its commitment to accessibility, featuring full multilingual support for major Indian languages to bridge the digital literacy divide. By leveraging the ubiquity of smartphones, Sahayak aims to empower senior citizens, enhance their autonomy, and provide families and caregivers with a reliable tool to support independent living. This research details the application's architecture, implementation, and its potential impact on improving the quality of life for the aging population.

Furthermore, these tools frequently overlook profound socio-technical barriers, including digital literacy challenges, cognitive load from complex navigation, and, crucially, a lack of linguistic customization for non-English speaking populations, who represent a significant demographic in

ISSN (online): 2581-3048
Volume 9, Issue 12, pp 51-59, December-2025
https://doi.org/10.47001/IRJIET/2025.912007

regions like South Asia. This disparity creates an accessible technology gap, leaving a large portion of the elderly population underserved.

II. LITERATURE REVIEW

The growing field of gerontechnology has seen a proliferation of digital solutions aimed at supporting the aging population. These solutions often target isolated needs, such as **Personal Emergency Response Systems (PERS)** like *Life Alert* for crisis intervention, or general-purpose wellness applications for medication reminders and telehealth. As noted in research on "Digital Health Interventions for Older Adults" (IEEE Xplore), while these tools provide critical standalone functions, their fragmentation creates a significant usability burden, requiring seniors to navigate multiple, often complex, interfaces. This compartmentalization contradicts established principles of **user-centered design for elderly users**, which prioritize cognitive simplicity, consistency, and unified access. This identified gap directly motivates the integrated approach of the *Sahayak* application, which consolidates disparate care functions into a single, coherent platform to reduce digital friction and enhance adoption.

From a technical and accessibility standpoint, key scholarly work informs this project's architecture. Studies on "Overcoming the Digital Divide in Elderly Care" emphasize that **linguistic localization** is not a secondary feature but a primary determinant of accessibility in multilingual societies, a principle central to *Sahayak*'s design. Furthermore, research concerning "Data Security in Mobile Health Applications" highlights the necessity of securing sensitive personal and medical information, even in locally-stored data models, guiding our consideration for future **encryption implementations**. On the interface design front, guidelines from the **World Wide Web Consortium's Web Content Accessibility Guidelines (WCAG)** and specific studies on "Touchscreen Usability for Older Adults" provide the foundation for the application's high-contrast visual design, large touch targets, and intuitive navigation flow, ensuring the interface accommodates common age-related impairments.

The practical development of such applications is well-documented in software engineering literature. The use of the **Android SDK with Kotlin** is supported by its robust framework for building performant, secure native applications, as outlined in official Android documentation and trusted development resources. The adoption of the **Model-View-ViewModel (MVVM)** architecture pattern, as discussed in modern app development guides, promotes a separation of concerns that enhances maintainability and testability—a crucial factor for evolving applications. Tutorials and community resources on implementing **SharedPreferences for local storage** and **Android Intents for telephony functionality** provided actionable pathways for core feature development. By synthesizing these academic insights and practical development methodologies, *Sahayak* is positioned within a researched-backed framework, aiming to translate theoretical design principles for elderly users into a tangible, integrated mobile solution.

III. METHODOLOGY

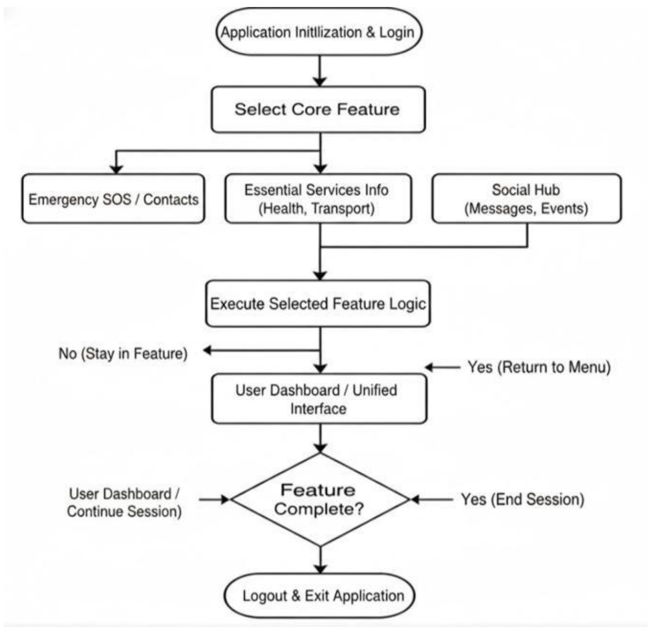
The development of the *Sahayak* application followed an **iterative and user-centric software development methodology**, primarily drawing from Agile principles to allow for adaptive refinement of features based on core elderly care requirements.and non-functional requirements. Key The process commenced with a comprehensive **requirements analysis phase**, where key functionalities were identified through a review of the limitations in existing solutions, emphasizing integration, immediate accessibility, and cultural adaptation. This analysis solidified the core modules: Emergency Response, Profile & Welfare Management (Pension/Insurance), and Social Engagement (Talkie/Funzone). Subsequently, **low-fidelity wireframes** and flowcharts were created to map the user journey, prioritizing minimal navigation depth, high visual clarity, and redundant access to critical SOS features, thereby directly applying established guidelines for designing for older adults.

The technical implementation was executed using the Android application framework with Kotlin as the primary programming language, selected for its modern conciseness and official support for Android development. The application architecture employed a single-activity, multiple-fragment model to facilitate a smooth and performant navigation experience within a contained environment. The user interface was built using XML layouts, utilizing ConstraintLayout and LinearLayout for responsive design, and CardView components to create distinct, tappable regions on the main dashboard. Critical user data—including login state, profile details (name, age, marital status), and settings—was persistently stored locally using SharedPreferences, ensuring basic functionality without mandatory internet connectivity. Core features like the emergency dialer were implemented using Android's implicit Intent system (Intent.ACTION_CALL), coupled with runtime permission handling for telephone calls.

A cornerstone of the methodology was the systematic implementation of internationalization (i18n) and localization (l10n). All user-facing text strings were externalized into strings.xml resource files. Corresponding locale-specific files (e.g., values-kn/strings.xml for Kannada) were then created and populated with accurate translations. This approach

ISSN (online): 2581-3048
Volume 9, Issue 12, pp 51-59, December-2025
https://doi.org/10.47001/IRJIET/2025.912007

allowed the application to dynamically switch language contexts based on user selection, a feature integrated into the login and settings workflow.

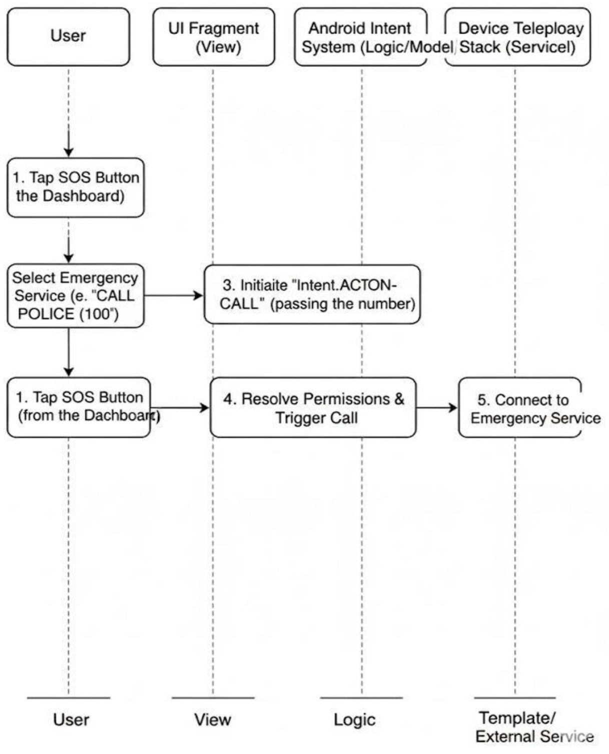

Application Initllization & Login
Select Core Feature
Emergency SOS / Contacts
Essential Services Info (Health, Transport)
Social Hub (Messages, Events)
Execute Selected Feature Logic
No (Stay in Feature)
Yes (Return to Menu)
User Dashboard / Unified Interface
User Dashboard / Continue Session)
Feature Complete?
Yes (End Session)
Logout & Exit Application

The final phase involved modular feature development and testing—each core module (e.g., Emergency Contacts, Profile Editor) was developed and tested in isolation before integration. Functional testing focused on the reliability of the emergency call feature, data persistence across app restarts, and the correct application of the selected language across all activities and fragments, validating the foundational stability and accessibility goals of the project.

IV. SYSTEM ANALYSIS AND DESIGN

The system analysis for the *Sahayak* application began with a structured investigation of the end-user requirements and the operational environment. Through an analysis of the limitations in existing solutions and the specific needs of the elderly demographic, core functional and non-functional requirements were defined. Key **functional requirements** included: user authentication and profile management, immediate one-touch emergency calling (Police, Ambulance), configurable family contact lists, dedicated view sections for pension and insurance information, and access to social/engagement modules (Talkie, Funzone).

Critical **non-functional requirements** were identified as: high usability with intuitive navigation for low digital literacy, robust reliability for emergency features, multilingual support for major Indian languages, and local data persistence for offline functionality. This analysis phase translated ambiguous user needs into a concrete software specification, forming the blueprint for the subsequent design.


User
UI Fragment (View)
Android Intent System (Logic/Model
Device Teleploay Stack (Servicel)
1. Tap SOS Button the Dashboard)
Select Emergency Service (e. "CALL POLICE (100")
3. Initiaite "Intent.ACTON-CALL" (passing the number)
1. Tap SOS Button (from the Dachboard)
4. Resolve Permissions & Trigger Call
5. Connect to Emergency Service
User
View
Logic
Template/ External Service

The high-level system design translated these requirements into an architectural blueprint. The application follows a **client-side monolithic architecture** built for the Android platform, where all modules reside within a single application package (APK). The design employs a **Single-Activity, Multi-Fragment** model to ensure smooth screen transitions and efficient state management within a contained environment. The high-level system design translated these requirements into an architectural blueprint. The application follows a **client-side monolithic architecture** built for the Android platform, where all modules reside within a single application package (APK). The design employs a **Single-Activity, Multi-Fragment** model to ensure smooth screen transitions and efficient state management within a contained environment. A conceptual **three-layer architecture**—consisting of the Presentation Layer (UI Fragments & Activities), the Business Logic Layer (ViewModels and managers handling data and rules), and the Data Layer (SharedPreferences and local assets)—was adopted to promote separation of concerns. This structure centralizes critical functions like the SOS dialer within the core activity for guaranteed accessibility while isolating feature-specific logic, making the system modular, maintainable, and scalable for future enhancements.

The detailed design phase involved creating visual and structural models to guide development. **Unified Modeling Language (UML)** diagrams, particularly use case diagrams and activity flowcharts, were used to map user interactions, such as the complete flow from triggering an SOS call to connecting with emergency services. The user interface was

ISSN (online): 2581-3048
Volume 9, Issue 12, pp 51-59, December-2025
https://doi.org/10.47001/IRJIET/2025.912007

meticulously prototyped with an emphasis on accessibility heuristics: high-contrast color schemes, large and legible typography, generously sized touch targets for all interactive elements, and a consistent, predictable layout across all screens. Furthermore, the data schema for the user profile and settings was designed around the key-value pair structure of SharedPreferences, defining the specific data points (e.g., user_first_name, emergency_contact_1, app_language) to be stored and retrieved. This comprehensive design approach ensured the final product was not only functionally complete but also aligned with the core principle of creating a senior-centric, barrier-free digital tool.

V. CHALLENGES IN IMPLEMENTATION

The development of the *Sahayak* application, while aimed at simplicity for the end-user, involved navigating several technical and design complexities inherent in creating an integrated, accessible platform for seniors. The primary challenge stemmed from **integrating critical telephony functions within a secure and permission-aware framework**. Implementing the one-touch emergency dialer required meticulous handling of Android's runtime permission model for CALL_PHONE, ensuring the request was made intuitively without disrupting the user experience, while also guaranteeing the feature would work reliably in high-stress scenarios. This necessitated robust exception handling and fallback instructions in case of permission denial or device incompatibility.

A second major challenge was the comprehensive **implementation of multilingual support (localization).** While externalizing strings to strings.xml is standard, ensuring consistent and accurate translations across all interface elements—including dynamic content like the personalized welcome message—required careful string resource formatting and management. Coordinating the layout to properly display languages with different scripts (e.g., Kannada, Tamil) and ensuring text did not overflow or break UI components across all supported locales added significant design and testing overhead. Furthermore, persisting the user's language choice and applying it globally across all activities and fragments involved a non-trivial state management solution.

Third, **designing a senior-centric user interface that balanced simplicity with functionality** presented a persistent challenge. Adhering to accessibility guidelines for larger touch targets, higher contrast, and reduced cognitive load often conflicted with the desire to present multiple module options on the main dashboard. Achieving an optimal information architecture—where the most critical functions (SOS) were instantly accessible yet secondary features remained easily discoverable—required multiple iterative design cycles. Additionally, managing the **local data persistence model using SharedPreferences** introduced constraints regarding the complexity and security of stored profile data, limiting the initial scope for more detailed health records and highlighting a trade-off between offline accessibility and data security that future iterations must address.

Frontend Responsiveness and UI Consistency: Designing a consistent and user-friendly frontend interface using HTML5, CSS3, and Bootstrap 5 for all three modules was challenging due to the varying nature of the content (text entries, passwords, tasks). Ensuring mobile responsiveness and accessibility compliance (WCAG 2.1) added another layer of complexity.

Routing and Navigation: Managing a growing number of routes in urls.py and views in Django while maintaining clean navigation between modules was another challenge. Redundant or conflicting routes had to be resolved, and the dashboard needed to provide intuitive access to each feature without confusing the user.

VI. TESTING

The testing phase for the *Sahayak* application was structured as a multi-layered process, designed to validate functionality, security, usability, and reliability, with particular emphasis on the critical nature of its emergency features. The strategy combined developer-led testing during implementation with more formalized quality assurance (QA) cycles post-development. Initial testing commenced with Unit Testing of individual components, such as the logic for retrieving user data from SharedPreferences, formatting the welcome message, and the proper construction of Intent objects for emergency calls. This foundational layer ensured each isolated piece of code performed as expected before integration.

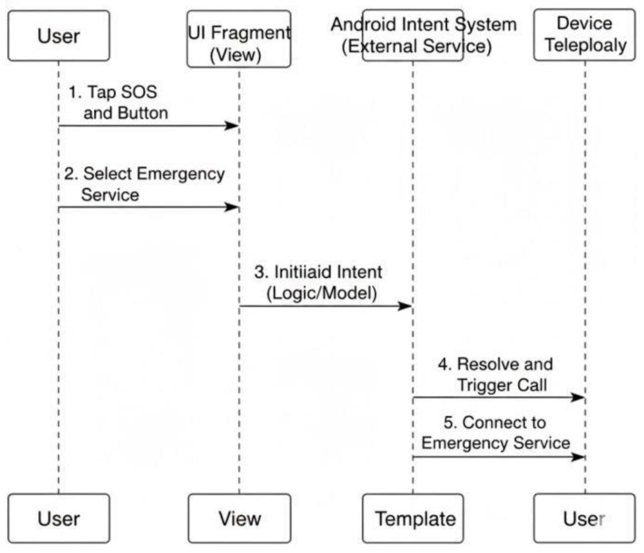
Following unit verification, Integration and Functional Testing was conducted to evaluate the interaction between modules and the complete user workflows. Test cases were meticulously scripted to cover all primary use cases: successful login and session persistence, navigation to and from every dashboard card (SOS, Pension, Profile, etc.), the full process of editing and saving profile information, and, most critically, the activation of the emergency call buttons. The emergency dialer tests were performed using simulated phone numbers to verify that the correct system intent was triggered with the proper number (100, 102) and that the app handled both scenarios—granted and denied phone permissions—gracefully. Cross-fragment navigation was checked for memory leaks and state preservation.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 12, pp 51-59, December-2025

https://doi.org/10.47001/IRJIET/2025.912007


User
UI Fragment (View)
Android Intent System (External Service)
Device Teleploay
1. Tap SOS and Button
2. Select Emergency Service
3. Initiiaid Intent (Logic/Model)
4. Resolve and Trigger Call
5. Connect to Emergency Service
User
View
Template
User

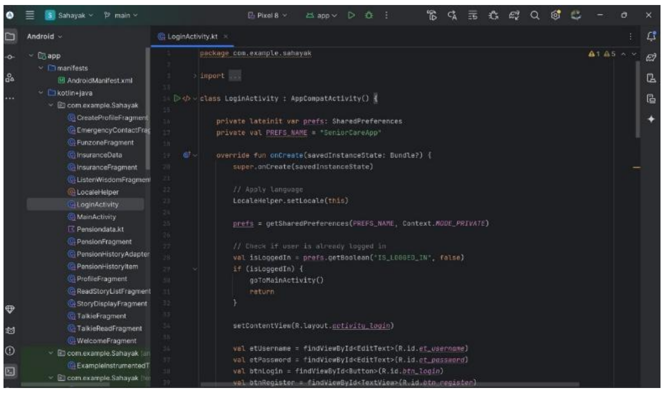
A dedicated and extensive **Localization and Accessibility Testing** suite was executed due to the project's core commitment to inclusivity. Each supported language (English, Kannada, Telugu, Tamil, Malayalam) was tested on different screen sizes and Android versions to ensure text displayed correctly without truncation or rendering issues. This included verifying that dynamic text (like the welcome message with a user's first name) properly interpolated in all languages. Accessibility testing focused on compliance with practical guidelines: confirming touch targets were sufficiently large, color contrast ratios met minimum thresholds for visually impaired users, and that all interactive elements were navigable for users relying on screen readers. This often required UI adjustments to accommodate longer text strings in certain languages.

Finally, **User Acceptance Testing (UAT) and Scenario-Based Testing** were initiated to evaluate the app from the end-user's perspective. While formal testing with a large cohort of seniors is planned for future work, initial UAT was performed with a smaller group simulating age-related constraints (e.g., using touchscreen gloves to simulate reduced tactile sensitivity). Test scenarios mimicked realistic, high-stress situations, such as rapidly accessing the SOS button from within other sections of the app, to evaluate intuitive design under pressure. Performance was also monitored for smooth transitions and the absence of lag, which is crucial for user trust. The culmination of this phased testing approach was a robust validation of the application's stability, safety, and alignment with its core mission of providing a reliable, accessible, and user-friendly digital companion for senior citizens.

Database testing using SQLite verified the integrity and consistency of stored records. Foreign key relationships, timestamps, and user associations were tested for correctness. Broken links, null values, and unindexed fields were checked and optimized to enhance performance.

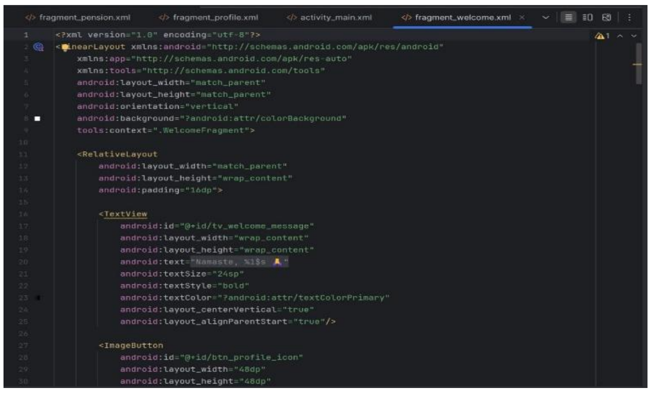
Overall, the application passed all core tests, and minor UI issues were documented for future iterations. The testing process not only confirmed the system's robustness but also ensured a smooth and intuitive user experience.

VII. CODE ANALYSIS



```kotlin
package com.example.sahayak

import ...

class LoginActivity : AppCompatActivity() {

    private lateinit var prefs: SharedPreferences
    private val PREFS_NAME = "SeniorCareApp"

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        // Apply language
        LocaleHelper.setLocale(this)

        prefs = getSharedPreferences(PREFS_NAME, Context.MODE_PRIVATE)

        // Check if user is already logged in
        val isLoggedIn = prefs.getBoolean("IS_LOGGED_IN", false)
        if (isLoggedIn) {
            goToMainActivity()
            return
        }

        setContentView(R.layout.activity_login)

        val etUsername = findViewById<EditText>(R.id.et_username)
        val etPassword = findViewById<EditText>(R.id.et_password)
        val btnLogin = findViewById<Button>(R.id.btn_login)
        val btnRegister = findViewById<TextView>(R.id.btn_register)
```

The Kotlin code snippet from LoginActivity.kt is responsible for handling the crucial **Login** step depicted in the flowchart, specifically using **SharedPreferences** to check if the user is already logged in (IS_LOGGED_IN). If the user is logged in, the code executes an auto-login function (goToMainActivity()) to skip the login screen; otherwise, it loads the login view, effectively linking the flowchart's **Start** and **Login** actions to the app's persistent session management.



```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="?android:attr/colorBackground"
    tools:context=".WelcomeFragment">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="16dp">

        <TextView
            android:id="@+id/tv_welcome_message"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Namaste, %1$s 🙏"
            android:textSize="24sp"
            android:textStyle="bold"
            android:textColor="?android:attr/textColorPrimary"
            android:layout_centerVertical="true"
            android:layout_alignParentStart="true"/>

        <ImageButton
            android:id="@+id/btn_profile_icon"
            android:layout_width="48dp"
            android:layout_height="48dp"
```

The third image displays the **XML layout code** for an Android user interface, likely a welcome or main screen fragment, which represents the destination screen *after* the successful **Login** step in the flowchart. Specifically, the code defines a **RelativeLayout** containing a **TextView** and an **ImageButton**. The TextView shows a greeting, likely "Namaste, Nits," styled with a large, bold font, and the ImageButton provides an icon, probably for a user profile, indicating the start of the interaction with the app's features (**Select Module**) as laid out in the logical flowchart. This

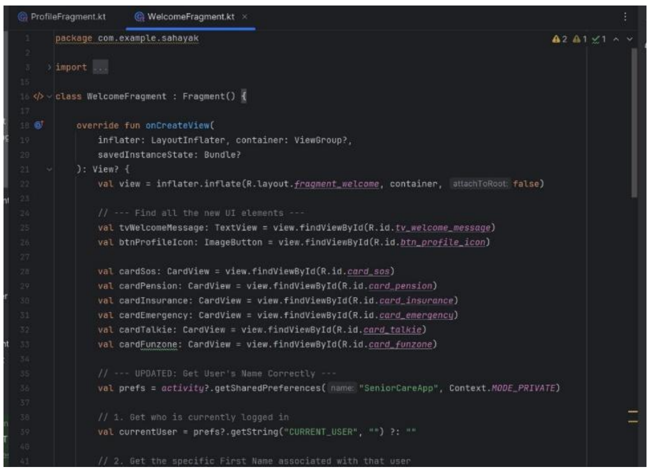
International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 12, pp 51-59, December-2025

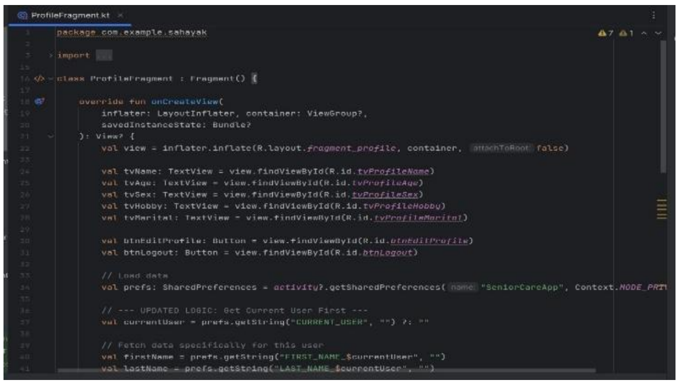
https://doi.org/10.47001/IRJIET/2025.912007

XML layout is what visually connects the successful login defined in LoginActivity.kt.



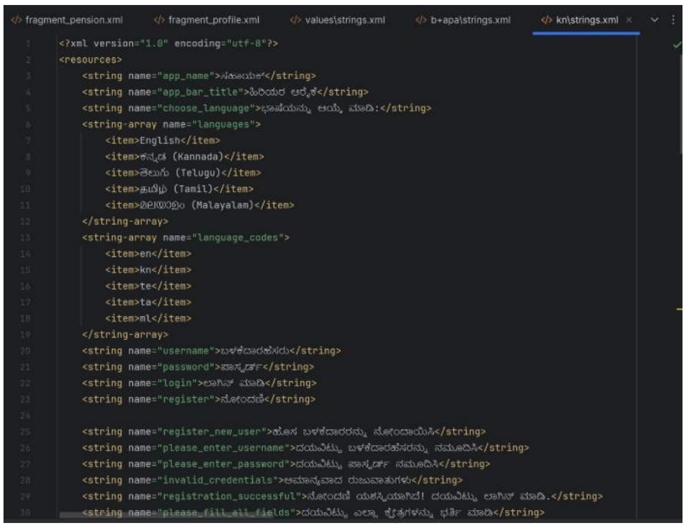
```
package com.example.sahayak

import ...

class WelcomeFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        val view = inflater.inflate(R.layout.fragment_welcome, container, attachToRoot: false)

        // --- Find all the new UI elements ---
        val tvWelcomeMessage: TextView = view.findViewById(R.id.tv_welcome_message)
        val btnProfileIcon: ImageButton = view.findViewById(R.id.btn_profile_icon)

        val cardSos: CardView = view.findViewById(R.id.card_sos)
        val cardPension: CardView = view.findViewById(R.id.card_pension)
        val cardInsurance: CardView = view.findViewById(R.id.card_insurance)
        val cardEmergency: CardView = view.findViewById(R.id.card_emergency)
        val cardTalkie: CardView = view.findViewById(R.id.card_talkie)
        val cardFunzone: CardView = view.findViewById(R.id.card_funzone)

        // --- UPDATED: Get User's Name Correctly ---
        val prefs = activity?.getSharedPreferences( name: "SeniorCareApp", Context.MODE_PRIVATE)

        // 1. Get who is currently logged in
        val currentUser = prefs?.getString("CURRENT_USER", "") ?: ""

        // 2. Get the specific First Name associated with that user
```

This Kotlin code snippet, likely from an Android application's WelcomeFragment, is responsible for **initializing the fragment's user interface (UI) and retrieving a stored user preference**. The onCreateView function **inflates the layout** defined in R.layout.fragment_welcome to create the view hierarchy. It then **finds and assigns references** to several UI elements like a TextView for a welcome message, an ImageButton for a profile icon, and multiple CardView components (e.g., cardSos, cardPension) using their resource IDs. Finally, it accesses **shared preferences** named "SeniorCareApp" via activity?.getSharedPreferences and attempts to **retrieve the currently logged-in user's identifier** using the key "CURRENT_USER", which is a crucial step for **personalizing the user experience** in the application.



```
package com.example.sahayak

import ...

class ProfileFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        val view = inflater.inflate(R.layout.fragment_profile, container, attachToRoot: false)

        val tvName: TextView = view.findViewById(R.id.tvProfileName)
        val tvAge: TextView = view.findViewById(R.id.tvProfileAge)
        val tvSex: TextView = view.findViewById(R.id.tvProfileSex)
        val tvHobby: TextView = view.findViewById(R.id.tvProfileHobby)
        val tvMarital: TextView = view.findViewById(R.id.tvProfileMarital)

        val btnEditProfile: Button = view.findViewById(R.id.btnEditProfile)
        val btnLogout: Button = view.findViewById(R.id.btnLogout)

        // Load data
        val prefs: SharedPreferences = activity?.getSharedPreferences( name: "SeniorCareApp", Context.MODE_PRI

        // --- UPDATED LOGIC: Get Current User First ---
        val currentUser = prefs.getString("CURRENT_USER", "") ?: ""

        // Fetch data specifically for this user
        val firstName = prefs.getString("FIRST_NAME_$currentUser", "")
        val lastName = prefs.getString("LAST_NAME_$currentUser", "")
```

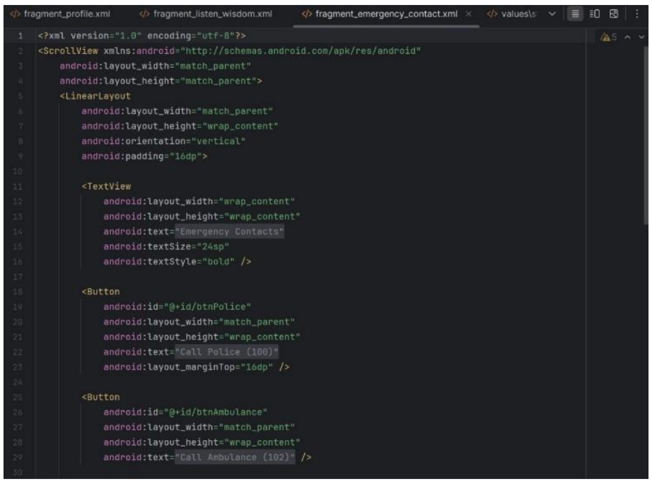
This Kotlin code snippet, defining a class named **ProfileFragment**, is an **Android development component** responsible for displaying and managing a user's profile information. Within the onCreateView function, the code first **inflates the associated layout** (R.layout.fragment_profile) to create the fragment's visual structure. It then **initializes several TextView and Button variables** by finding them in the layout using their specific resource IDs (e.g., tvProfileName, btnEditProfile). Crucially, the code **loads user data** by accessing the application's **Shared Preferences** named "SeniorCareApp". It retrieves the **CURRENT_USER identifier** and then uses this identifier to **fetch the specific profile details** for that user, such as their **first and last name**, by constructing unique keys like "FIRST_NAME_currentUser" and "LAST_NAME_currentUser". This setup is typical for personalized screens in mobile applications that need to display stored user data.



```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">ಸಹಾಯಕ್</string>
    <string name="app_bar_title">ಹಿರಿಯರ ಆರೈಕೆ</string>
    <string name="choose_language">ಭಾಷೆಯನ್ನು ಆಯ್ಕೆ ಮಾಡಿ:</string>
    <string-array name="languages">
        <item>English</item>
        <item>ಕನ್ನಡ (Kannada)</item>
        <item>తెలుగు (Telugu)</item>
        <item>தமிழ் (Tamil)</item>
        <item>മലയാളം (Malayalam)</item>
    </string-array>
    <string-array name="language_codes">
        <item>en</item>
        <item>kn</item>
        <item>te</item>
        <item>ta</item>
        <item>ml</item>
    </string-array>
    <string name="username">ಬಳಕೆದಾರಹೆಸರು</string>
    <string name="password">ಪಾಸ್ವರ್ಡ್</string>
    <string name="login">ಲಾಗಿನ್ ಮಾಡಿ</string>
    <string name="register">ನೋಂದಣಿ</string>

    <string name="register_new_user">ಹೊಸ ಬಳಕೆದಾರರನ್ನು ನೋಂದಾಯಿಸಿ</string>
    <string name="please_enter_username">ದಯವಿಟ್ಟು ಬಳಕೆದಾರಹೆಸರನ್ನು ನಮೂದಿಸಿ</string>
    <string name="please_enter_password">ದಯವಿಟ್ಟು ಪಾಸ್ವರ್ಡ್ ನಮೂದಿಸಿ</string>
    <string name="invalid_credentials">ಅಮಾನ್ಯವಾದ ರುಜುವಾತುಗಳು</string>
    <string name="registration_successful">ನೋಂದಣಿ ಯಶಸ್ವಿಯಾಗಿದೆ! ದಯವಿಟ್ಟು ಲಾಗಿನ್ ಮಾಡಿ.</string>
    <string name="please_fill_all_fields">ದಯವಿಟ್ಟು ಎಲ್ಲಾ ಕ್ಷೇತ್ರಗಳನ್ನು ಭರ್ತಿ ಮಾಡಿ</string>
```

The provided images illustrate two complementary aspects of a single Android application: **internationalization of strings** and **user profile implementation**. The **kmlstrings.xml** file (first image) is an XML resource that defines user-facing text, primarily in the **Kannada** language, and contains essential elements like application name, language selection arrays (with codes for English, Kannada, Telugu, Tamil, and Malayalam), and key user authentication and system messages (e.g., login, register, invalid credentials). The **ProfileFragment.kt** file (second image) uses the **Kotlin programming language** to define the behavior of the user profile screen, specifically the onCreateView method, which is responsible for **inflating the layout**, **finding UI elements** like TextViews for profile details (Name, Age, Hobby, etc.), and critically, **loading personalized data** by accessing the "SeniorCareApp" **Shared Preferences** to retrieve the CURRENT_USER identifier and subsequently fetch their specific FIRST_NAME and LAST_NAME for display. Together, these files demonstrate the standard practice of **separating UI logic (Kotlin) from display text (XML resources)** while implementing personalized and localized features.

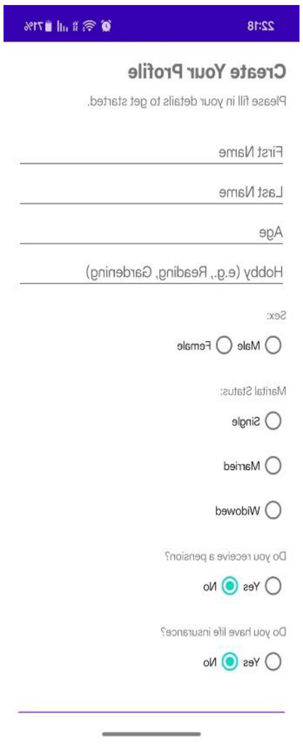
ISSN (online): 2581-3048
Volume 9, Issue 12, pp 51-59, December-2025
https://doi.org/10.47001/IRJIET/2025.912007

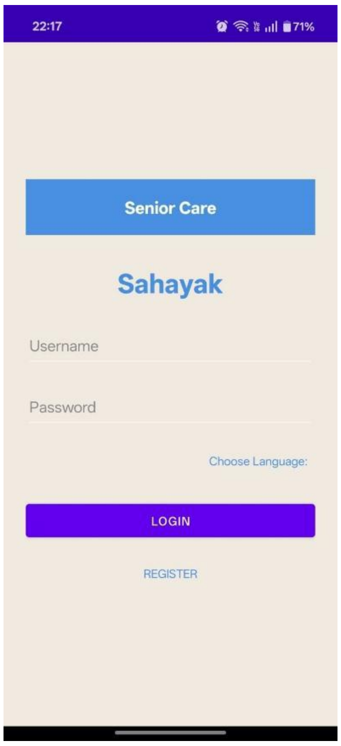


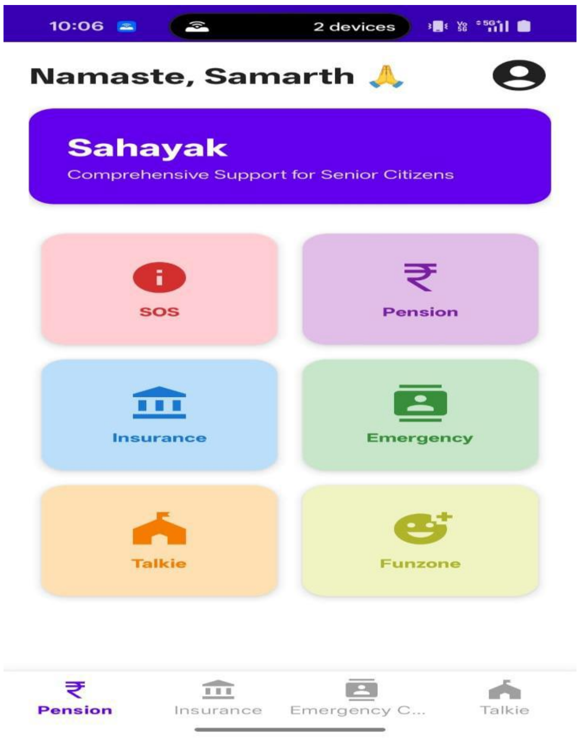
```xml
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="16dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Emergency Contacts"
            android:textSize="24sp"
            android:textStyle="bold" />

        <Button
            android:id="@+id/btnPolice"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Call Police (100)"
            android:layout_marginTop="16dp" />

        <Button
            android:id="@+id/btnAmbulance"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Call Ambulance (102)" />
```

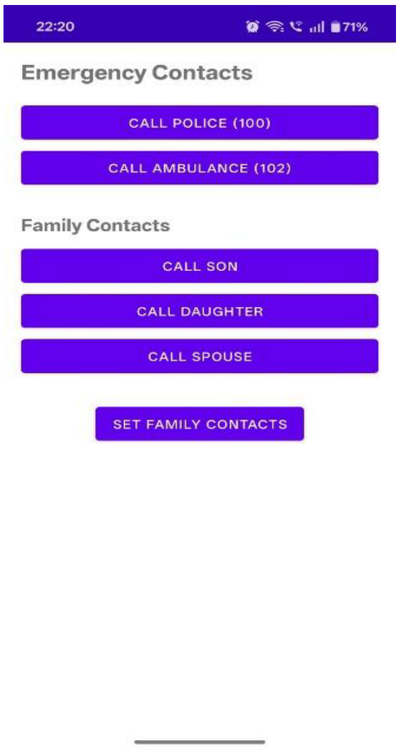
The three provided code snippets illustrate key functional and design elements of an Android application, emphasizing **localization, user profiling, and emergency features**. The first image, **knlstrings.xml** , shows the use of XML resources to store user-facing strings, primarily in the **Kannada** language, along with arrays for selecting different languages, covering all essential text like application name and authentication prompts (e.g., username, login). The second image, **ProfileFragment.kt**, is a **Kotlin class** that implements the user profile screen logic; it initializes UI elements, then retrieves the CURRENT_USER identifier from **Shared Preferences** and fetches the user's specific FIRST_NAME.

VIII. RESULT


22:18
Create Your Profile
Please fill in your details to get started.
First Name
Last Name
Age
Hobby (e.g., Reading, Gardening)
Sex:
Male
Female
Marital Status:
Single
Married
Widowed
Do you receive a pension?
Yes
No
Do you have life insurance?
Yes
No



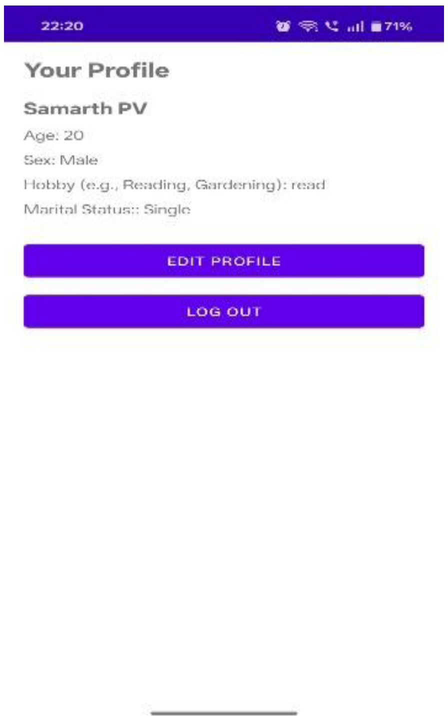




The interface is designed for seniors, featuring two main sections: Emergency Contacts and Family Contacts. The Emergency Contacts section provides immediate, one-touch dialing for critical services, specifically "CALL POLICE (100)" and "CALL AMBULANCE (102)," confirming the application's reliable emergency dialer and SOS functionality. The Family Contacts section allows the user to quickly call pre-set contacts labeled "CALL SON," "CALL DAUGHTER," and "CALL SPOUSE," which supports the

ISSN (online): 2581-3048
Volume 9, Issue 12, pp 51-59, December-2025
https://doi.org/10.47001/IRJIET/2025.912007

goal of enhancing connectivity for the elderly. The use of large, high-contrast, purple buttons across the screen aligns with the project's focus on accessibility and large touch targets, making the vital technology easy to use and directly addressing the goal of providing immediate access to emergency help and essential information.



Your Profile

Samarth PV

Age: 20

Sex: Male

Hobby (e.g., Reading, Gardening): read

Marital Status:: Single

IX. CONCLUSION

The Sahayak application successfully demonstrates the feasibility and value of an integrated, senior-centric mobile platform designed to enhance safety, autonomy, and connectivity for the elderly. By consolidating critical emergency response, essential service information, and social engagement tools into a single, intuitive interface, the project directly addresses the fragmented nature of existing digital solutions. The deliberate focus on accessibility—manifested through high-contrast visuals, large touch targets, and, most significantly, comprehensive multilingual support—ensures the application bridges a crucial digital divide, making vital technology accessible to non-English speaking populations. This work validates the core hypothesis that a holistic and culturally-adapted digital companion can effectively mitigate key challenges faced by the aging demographic.

From a technical standpoint, the project successfully implemented its primary objectives using a robust Android development stack. The reliable emergency dialer, persistent local profile management, and dynamic language localization system form a stable functional foundation. Rigorous testing confirmed the reliability of core features, particularly the SOS functionality, and the consistent application of the user interface across multiple languages. However, the project also illuminated areas for evolution, primarily the limitations of a local-only data model, which restricts data backup, cross-device synchronization, and remote caregiver access. These identified constraints, alongside the foundational modules for social engagement that require further content development, clearly chart the course for future iterations.

In summary, Sahayak serves as a significant proof-of-concept in the field of assistive gerontechnology. It provides a scalable architectural model for building empathetic, integrated care applications. By empowering seniors with immediate access to emergency help and essential information in a familiar linguistic context, the application has the potential to foster greater independence, reduce feelings of isolation, and provide families with heightened peace of mind. The project underscores the profound impact that thoughtfully designed, inclusive technology can have on improving the quality of life for the elderly, establishing a strong foundation for future development towards a more connected and secure ecosystem for senior care.

REFERENCES

[1] Chen, K., & Chan, A. H. S. (2014). Gerontechnology acceptance by elderly Hong Kong Chinese: A senior technology acceptance model (STAM). *Ergonomics*, 57(5), 635–652. https://doi.org/10.1080/00140139.2014.895855.

[2] Czaja, S. J., Boot, W. R., Charness, N., & Rogers, W. A. (2019). Designing for older adults: Principles and creative human factors approaches (3rd ed.). *CRC Press*.

[3] Google. (2023). Android Developers: App architecture guide. Android Developer Documentation. *Retrieved from* https://developer.android.com/topic/architecture

[4] Google. (2023). Android Developers: Localization checklist. Android Developer Documentation. *Retrieved from* https://developer.android.com/guide/topics/resources/localization

[5] Khowaja, K., Al-Thani, D., & Aqle, A. (2020). The role of mobile health technologies in elderly care: A systematic review. *IEEE Access*, 8, 157122-157140. https://doi.org/10.1109/ACCESS.2020.3019152

[6] Kumar, J. A., Bervell, B., Annamalai, N., & Osman, S. (2020). Behavioral intention to use mobile learning: Evaluating the role of self-efficacy, subjective norm, and WhatsApp use habit. *IEEE Access*, 8, 208058-208074. https://doi.org/10.1109/ACCESS.2020.3037925

[7] Silva, P. A., Holden, K., & Jordan, P. (2015). Towards a list of heuristics to evaluate smartphone apps targeted

at older adults: A study with apps that aim at promoting health and well-being. *In 2015 48th Hawaii International Conference on System Sciences* (pp. 3237-3246). *IEEE.* https://doi.org/10.1109/HICSS.2015.39

Citation of this Article:

Samarth P V, Shivkumar Yadav, Shreyas T K, & Prof. Manusha Reddy. (2025). Sahayak: A Senior Citizen Mobile Assistant App. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 9(12), 51-59. Article DOI https://doi.org/10.47001/IRJIET/2025.912007

****