International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Gram Aarogya App: Design and Prototype Implementation of an Offline First Android Telemedicine Platform for Rural Indian Healthcare

Authors: Dr. Vishal Khatri, Asst. Professor, CSE Department
Tanishq Bhadula, Vanshika Tanwar, Sarvagya Bajpai
CSE Department, Bhagwan Parshuram Institute of Technology, Delhi, India

ABSTRACT: Access to proper healthcare is still difficult for many rural communities because of travel costs, limited doctor availability, network problems and difficulty in accessing reliable health information. This paper presents Gram Aarogya App, an Android telemedicine prototype designed to solve the healthcare access problem described for Nabha region of Punjab, in Smart India Hackathon problem statement SIH-25018. The app uses Kotlin and Jetpack Compose interface, Room database with SQLite, Firebase cloud services, Google Maps based clinic locating and Google Gemini AI powered assistance for symptom checking, basic diet guidance and layman language medical report analysis. Various prompt restrictions and safety checks are used to keep AI outputs within an informational role, identify certain emergency red flags and direct the users to contact qualified healthcare professionals in case of urgency. The prototype has undergone initial developer led functional checks of core app flows, AI responses and Firebase/Room connectivity. These checks have established basic operation in the test environment but do not establish clinical accuracy, on ground uses or production level readiness. This paper documents the architecture, system design, implementation, initial functional observations, limitations and an expandable plan for future technical and safety evaluation. The main aim is to make an integrated, rural friendly Android design that works for local data access and has selectively trained AI support as complementary service to medical professionals and doctors.

I. INTRODUCTION

A. Background

The telemedicine apps help patients seek medical advice without visiting a doctor every time. Their use is still dependent on proper access to medical services, the ability to use it and the availability of follow up medical care. India's eSanjeevani service has seen telemedicine sector's operation at a large scale. A published analysis showed 163 million consultations between November of 2019 and September of 2023 [1]. Studies in sector of telemedicine in India show the current problems include poor network, low digital literacy and integration with existing healthcare services [10]–[14].

The problem is inspired from the SIH statement 25018 that describes rural healthcare access related problems around Nabha region of Punjab. According to the given problem statement, the Civil Hospital of Nabha has 11 doctors for the 23 sanctioned posts and serves patients from 173 villages from the region. Travelling to a hospital costs a patient their time and wages, without any proper knowledge if a specialist or a required medicine is available or not. Earlier research on public health facilities in Punjab and Haryana have reported problems in medicine availability [9].

B. Objective and Research Questions

The aim of this project is to build an Android application that gives users present in rural areas access to health information and AI assistance while keeping selected data available offline. It majorly tries to solve three questions: 1) the functions that work in the current prototype 2) making the AI features be guided for medical advice or prescribing and 3) the further testing needed before the application can be deployed in a real healthcare setting?

D. Working Hypothesis

The current working hypothesis is combining Room Database/SQLite local storage with Firebase cloud services and Gemini workflows can produce a functional Android Application prototype that supports local data and connected guidance features. The hypothesis is evaluated only at the level of initial software operations. On ground clinical accuracy, complete offline reliability and real user impact currently remain untested.

C. Contributions

The paper describes how Room database/SQLite storage, Firebase Cloud integration, nearby hospital locator and three Google Gemini AI based workflows are implemented in our Android prototype. It explains the prompt regulations and safety measures added to the AI features. Lastly, it records the team's initial functional checks and outlines further on ground testing needed to ensure smooth operation of our offline sync system, safety and usability.

II. RELATED WORK

A. Telemedicine systems and Rural Healthcare

The Indian telemedicine and mobile health systems have opportunities for growth with issues such as proper infrastructure, integrated workflow, simplified usage and offline availability [10]-[14]. A rural outpatient telemedicine implementation in Pune provides a relevant precedent for remote consultation in an Indian context [15]. Research on community health-worker applications also suggests that dedicated workflows for local health workers can support service delivery, although such benefits require evaluation in the target setting [8].

B. Offline-First Data Design

Local-first software design places a usable local copy of data at the centre of the user experience, rather than making every interaction dependent on a live cloud connection [2]. An offline-first electronic health record feasibility study reported practical benefits in resource-limited clinics while noting workflow friction associated with synchronization timing [3]. These findings motivate the use of Room over SQLite for local persistence and Firebase for cloud services. They do not, by themselves, prove that Gram Aarogya's complete offline synchronization behaviour has been validated.

C. Risks of AI-Generated Health Information

Studies of large language models and online symptom checkers have reported limitations in clinical correctness, triage and contextual reasoning [4], [5]. Evaluations of generated explanations for laboratory results have also

identified incorrect or insufficiently contextual answers [16]. Various diet guidance studies show that legible advice can still be somewhat impractical, not personalized or easily misused and misinterpreted [17], [18]. Accordingly, the Gram Aarogya App treats AI output as assisted support and not a replacement for a medical professional. It applies various prompt restraints and safety/security checks, and directs users to urgent professional care in case of emergencies or complex cases.

D. Gaps in Research

Studies examine the telemedicine, offline first records, AI based medical information, low digital literacy compatible interfaces and medical access as related yet different concerns. Gram Aarogya App explores an integrated solution in one Android prototype. The novelty claimed in the system is context integrated architecture.

TABLE I. SELECTED TOPICS AND DESIGN IMPLICATIONS

Topic	Relevant evidence	Implication for this work
Indian telemedicine	National usage and implementation reviews [1], [10]-[15]	Design around real problems, evaluate usage and integrated services.
Offline first systems	Local first principles and an offline EHR study [2], [3]	Use RoomDB/SQLite for local storage and Firebase cloud sync.
AI based symptom analyzer	Accuracy limitations in AI models and LLMs [4], [5]	Bound output, check red flags and avoid diagnosis or prescriptions.
Report and diet analysis	Personalization and practicality limitations [16]-[18]	Provide personalized support.
Community health workers	Rural mHealth implementation and digital literacy research [7], [8], [14]	Keep navigation simple and evaluate the workflow for intended rural users.

III. SYSTEM REQUIREMENTS

A. Target Users

Gram Aarogya App is designed specifically for rural users with low digital litracy such as farmers and daily wage workers who face travel costs, for doctors who may benefit from structured patient systems and for ASHA or other community health workers who may assist patients with digital tasks.

TABLE II. FUNCTIONAL REQUIREMENTS

ID	Requirement	Current interpretation
FR1	Patient-facing Android interface	Implemented core screens; complete Punjabi/Hindi localization requires verification.
FR2	Local persistence	Room database backed by SQLite stores supported local data.
FR3	Cloud service integration	Firebase integration is connected and has passed basic connectivity checks; synchronization behaviour needs formal measurement.
FR4	AI based symptom analysis chatbot	Gemini prompt restraints and safety checks constrain responses to general medical advice and emergency doctor consultation if needed.
FR5	AI based dietician	Gemini AI provides general diet suggestions, it will not suggest therapeutic diets.
FR6	Medical report analysis	Provides layman language explanation and flags values against the report's stated reference range where supported.
FR7	Clinic discovery	Google Maps/Places-based clinic discovery is included in the prototype.
FR8	Consultation and medicines	Live video consultation, live pharmacy inventory and complete ASHA workflows remain future/incomplete items unless separately verified.

B. Non Functional Requirements

The objectives for designing Gram Aarogya system are offline architecture for certain tasks, easy to use interface, security of patient's health information, scalable logic, and safe AI behaviour.

C. Technology Stack

TABLE III. TECHNOLOGY STACK

Technology	Role in Gram Aarogya
Kotlin (Jetpack Compose)	Android UI
Room over SQLite	Structured local persistence and access to locally stored application data.
Firebase	Cloud services and application integration; the exact Firebase products enabled in the deployed build should be listed in the final version.
Google Gemini API	AI-generated symptom-screening support, general dietary guidance and medical-report explanation.
Google Maps/Places APIs	Clinic discovery.
Git/GitHub	Version control and collaboration.

Room is an abstraction layer over SQLite and not a separate server or cloud backend. Firebase is the cloud based service, while SQLite provides local storage.

IV. SYSTEM ARCHITECTURE

A. Overall System Architecture

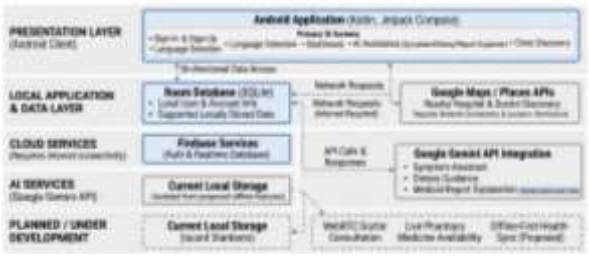


Fig. 1. Overall system architecture of Gram Aarogya, showing the structure, local data storage, Firebase Cloud services, Google Gemini API and Google Maps integration.

B. Data Flow Diagram

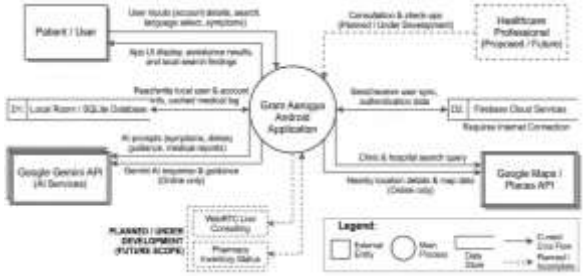


Fig. 2. Data flow diagram showing user input, local data access, cloud operations and connected AI & clinic discovery requests.

C. Data Storage and Connectivity Flow

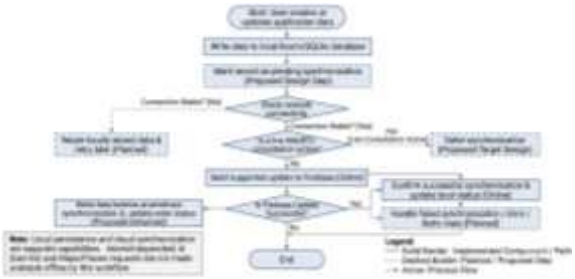


Fig. 3. Local and cloud data flow, distinguishing data stored locally from features that require an internet connection.

V. APPLICATION SCREENSHOTS

A. Home Screen and Dashboard



Fig. 4. Gram Aarogya home screen and dashboard.

B. AI Symptom Assistant Chatbot



Fig. 5. Symptom assistant interface

C. Medical Report Analysis



Fig. 6. Medical report explanation interface.

D. Nearby Clinic Discovery



Fig. 7. Nearby clinic discovery screen

E. User Profile



Fig. 8. Profile screen

VI. MATERIALS AND ENVIRONMENT

A. Materials and Development Environment

Gram Aarogya is an Android based app made using Kotlin and Jetpack Compose. Room provides the local storage layer over SQLite and Firebase provides the cloud storage. Google Gemini API supports the AI components. Google Maps APIs are used for nearby clinics/hospitals. The prototype was developed and checked by the project team. No physical clinical instruments or patient samples were used in the current software checks.

B. System Design

The Android UI is coded with Kotlin and Jetpack Compose. Room, which uses SQLite underneath, is used for local data storage on the user's device. Firebase provides cloud storage. Google Gemini AI is used for AI assisted features. Google Maps API is used for locating nearby clinics/hospitals. All of these features and functions have different network requirements, the data stored on device can be accessed offline, but maps API, AI requests and firebase cloud operations need an active network connection.

C. Data Handling According to Connectivity

The data stored locally on user's device can be access without internet. Cloud based features, Google Maps API calls and Gemini based AI functions require internet. The synchronization design is to flag local changes as pending and sync/update them when connectivity returns, while avoiding data conflicts that overwrite newer information. Because of limited current testing, the paper does not claim guaranteed conflict free or fully validated offline synchronization.

D. AI Safety Restraints

The AI functions use restraint and specific prompts to define their scope and response format. The constraints used are 1) the assistant does not diagnose, prescribe medicines or provide dosage instructions on its own 2) it guides urgent or potentially dangerous symptoms towards immediate medical checkup 3) the model displays a notice that generated information is not a substitute for a doctor in any way 4) it discourages personalized therapeutic diet advice and directs patients with chronic symptoms or concerning reports to a doctor, and 5) report analysis is tied to the values and references ranges visible in the supplied report only. These controls reduce foreseeable misuse and misinterpretation.

E. Security

Health records, symptoms and report content have sensitive user information. The application restricts access to the authenticated user records, applies least privilege rules to the Firebase resources, avoids exposing privileged API credentials in the Android source code, and uses secure protocols for network requests. Local credential storage use an authentication design.

VII. AI BASED FEATURES

A. Symptom Analyzer Chatbot Assistant

The Google Gemini AI based symptom analysis chatbot is there for users to get basic advice and guidance. The hard coded restraints make sure the assistant only follows the intended model, i.e. self-care information for mild symptoms, arranging a clinical consultation if needed, or seeking urgent care in case of concerning systems. Selected red flag patterns trigger an emergency message bypassing the model responses altogether.

B. AI Dietary Guidance

The Google Gemini AI based dietician assistant generates general food and lifestyle guidance using context of the patient for personalized response. Advice refers to easily available local food items and general prescribed balanced diet principles. It does not recommend any special diets for a health problem or try to cure that problem through diet alone. If the patient asks about a severe problem, pregnancy, or concerning test results, the assistant advices urgent medical consultation.

C. AI Report Analyzer

The Google Gemini AI based report analyzer model simplifies medical terminology and values in local layman language. All values are put into categories using the ranges mentioned in the uploaded report. The model advices users to discuss the report with a doctor/ dietician if needed.

TABLE IV. AI SAFETY MEASURES

Measure	Purpose
Task specific prompts	Ensure response is limited to analysis and explanation only.
Emergency red flag checks	Advice urgent medical advice for emergencies.
No diagnosis/prescription instruction	Ensure a response is not treated as final doctor's advice.
Dietary advice limitations	Make sure dietician model isn't misused or misinterpreted.
Report range explanation	Categorize and explain printed values in the report without inferring disease.
Privacy and access controls	Prevent misuse of sensitive user data.

VIII. RESULTS

A. Prototype Components Evaluated

The prototype currently includes the main Android application flow, local persistence through Room/SQLite, Firebase integration, Gemini-based AI workflows and clinic discovery. The AI prompts and safety/security measures have been added, and the dietician module is integrated. Basic functional tests have been reported as working by the development team. The

checks are preliminary and were not designed as a clinical validation study.

TABLE V. IMPLEMENTATION STATUS BASED ON THE CURRENT PROJECT UPDATE

Component	Status	Qualification
Kotlin/Jetpack Compose core screens	Implemented	Basic application flows tested by the development team.
Room/SQLite local database	Implemented	Local persistence; broader offline coverage requires a task-by-task test.
Firebase integration	Integrated	Basic connectivity/function checks reported; security-rule and synchronization behaviour should be separately documented.
Gemini symptom assistant and prompt restrictions	Implemented	Basic tests reported; no clinical accuracy claim.
AI dietary guidance	Integrated	Basic tests reported; requires expert safety review.
Medical-report explanation	Prototype workflow	Retain as implemented only for the exact input formats verified in the current build.
Clinic discovery	Implemented in prototype	Depends on user location and internet connection.
Complete language localization	In development	Local language support needs more extensive testing
Live doctor video consultation	In development	WebRTC in poor network needs more testing
Live pharmacy inventory	Not implemented/verified	Mock data must be labelled as mock if still present.
Dedicated ASHA/doctor roles	Not implemented/verified	Include only after role-specific flows and access controls work.

B. Functional Test Observations

Basic tests have worked for major application functions, AI assistants, local storage on user's device using Room/SQLite and Firebase cloud services integration. The checks done are preliminary observations only. There is not enough numerical data to currently to calculate a pass rate, average response time, confidence interval or p-value.

TABLE VI. PRELIMINARY FUNCTIONAL CHECKS AND REPORTING LIMITS

Check area	Current observation
Core application flow	All major workflows and checks passed.
Local data operations	Room/SQLite stores user's information locally.
Firebase connectivity	Integration is working and major tests passed.
AI prompts' safety/security restraints	Prompt restrictions and safety/security measures added.
Dietary guidance	Module integrated and basic tests reported as working.
Report explanation	Prototype workflow included.

IX. FOLLOW UP EVALUATION PROTOCOL

Certain set of tests are reserved for further evaluation. They specify on ground user interactions and measures that can be used to further collect more data before finalizing our claims about AI security, user data protection, and offline reliability/synchronization for deployment.

TABLE VII. PROPOSED EVALUATION PLAN

Evaluation	Method	Measures
Offline functionality	Run supported local tasks on multiple Android devices in airplane mode, with predefined test cases.	Task success rate, failures, completion time.
Firebase sync	Create and update records under stable and intermittent connectivity; inspect retries and conflicts.	Sync success rate, delay, duplicate/lost records, conflicts.
Symptom safety	Use medically backed symptom sets in local language, including hard coded emergency red flags.	Red flag symptom check, under & over triage, unsafe output count.
Report explanation	Compare output from synthetic or approved anonymized reports against a verified answer key.	Value extraction, range-flag accuracy, unsupported diagnostic statements.
Dietary guidance	The outputs and prompts are reviewed by professionals.	Correctness, simplicity, safety, and practicality ratings.
Security Measures	Test new user onboarding, user authorization, data leak and credential handling.	Pass/fail per test case and nature of results.
Usability	Conduct an approved study with intended users after ethics/consent requirements are met.	Task completion, errors, completion time and a standardized usability measure.

X. DISCUSSION

The current version of Gram Aarogya App is a working prototype, with its core features implemented and a few modules still under development. Our initial checks confirmed that the main application features and functions work as intended. Further testing is now needed to assess how reliably the app handles edge cases and unintended inputs and other situations that may arise after deployment. The system has a working initial architecture and data flow. Its impact and applicability on rural health care sector can be truly verified only after deployment.

A major design USP is balancing local data access with features that depend on an active/reliable internet connection. Room/SQLite have been specifically used to keep locally stored without network. The components such as Google Gemini AI API requests, Google Maps API requests and cloud Firebase functions require an active internet connection. The goal was to build a system that can manage both, offline access for certain essential features, which remain available even without a network connection and functions that can work only with an active internet connection.

The app uses a Google Gemini based generative AI model to support initial symptom checking, basic dietary guidance and medical report analysis. Gemini is chosen as a common model across all assistants to simplify integration and maintenance. The prompt restrictions and certain hard coded checks help keep the AI responses within a safe limit. Gram Aarogya has reached a completely functional stage with the next set of actions being about improving reliability. Further testing on various screens, different mobiles devices, varying network

connections and user input edge cases will help find the refinements needed. Consultation and active feedback of actual users and medical professionals would also help determine the practical use of the app. These measures are needed to refine our system and ensure it is complete for final deployment.

XI. CONCLUSION

This paper presented Gram Aarogya, an Android telemedicine prototype designed for the rural healthcare-access context described for Nabha, Punjab. The system combines Kotlin and Jetpack Compose, Room over SQLite, Firebase integration, clinic discovery and Gemini-based symptom screening, dietary guidance and report explanation. The functional checks are working and AI prompts, safety/security measures, Firebase integration and dietary guidance have been added. These results demonstrate our current working prototype condition, not on-field deployment.

The future scope includes documentation of more intensive test cases and outcomes, validation of security measures and synchronization, checking of AI outputs by professionals, completion of any remaining localization and role based workflows, and evaluation of practical use with real users in rural regions. Gram Aarogya's guiding principle remains that digital tools can support access to information and care coordination but must not replace qualified healthcare professionals.

REFERENCES

[1] S. Sood et al., “Adoption and utilization of India’s eSanjeevani national telemedicine service,” Oxford Open Digital Health, vol. 3, art. oqaf025, 2025, doi: 10.1093/oodh/oqaf025.

[2] M. Kleppmann, A. Wiggins, P. van Hardenberg, and M. McGranaghan, “Local-first software: you own your data, in spite of the cloud,” in Proc. ACM SIGPLAN Onward! 2019, Athens, Greece, Oct. 2019, pp. 154–178, doi: 10.1145/3359591.3359737.

[3] H. Ashista et al., “An offline-first electronic health record for vulnerable populations: A mixed-methods feasibility study,” PLOS Digital Health, vol. 5, no. 2, art. e0001204, 2026, doi: 10.1371/journal.pdig.0001204.

[4] B. Mingole, A. Majumdar, F. A. Choudhury, J. L. Kraschnewski, S. S. Sundar, and A. Yadav, “Dr. GPT Will See You Now, but Should It? Exploring the Benefits and Harms of Large Language Models in Medical Diagnosis using Crowdsourced Clinical Cases,” arXiv:2506.13805, 2025.

[5] E. Riboli-Sasco et al., “Triage and Diagnostic Accuracy of Online Symptom Checkers: Systematic Review,” J. Med. Internet Res., vol. 25, art. e43803, 2023, doi: 10.2196/43803.

[6] E. A. Tarim and H. C. Tekin, “Performance evaluation of WebRTC-based online consultation platform,” Turkish Journal of Electrical Engineering and Computer Sciences, vol. 27, no. 6, pp. 4314–4327, 2019, doi: 10.3906/elk-1903-44.

[7] I. Medhi, S. Patnaik, E. Brunskill, S. N. N. Gautama, W. Thies, and K. Toyama, “Designing Mobile Interfaces for Novice and Low-Literacy Users,” ACM Transactions on Computer-Human Interaction, vol. 18, no. 1, art. 2, 2011, doi: 10.1145/1959022.1959024.

[8] P. Shah, N. Madhiwala, S. Shah, G. Desai, K. Dave, N. Dholakia, S. Patel, S. Desai, and D. Modi, “High uptake of an innovative mobile phone application among community health workers in rural India: An implementation study,” National Medical Journal of India, vol. 32, no. 5, pp. 262–269, 2019, doi: 10.4103/0970-258X.295956.

[9] S. Prinja, P. Bahuguna, J. P. Tripathy, and R. Kumar, “Availability of medicines in public sector health facilities of two North Indian States,” BMC Pharmacol. Toxicol., vol. 16, art. 43, 2015, doi: 10.1186/s40360-015-0043-8.

[10] E. Rajkumar et al., “Applications, benefits and challenges of telehealth in India during COVID-19 pandemic and beyond: a systematic review,” BMC Health Serv. Res., vol. 23, art. 7, 2023, doi: 10.1186/s12913-022-08970-8.

[11] S. Arora, R. K. Huda, S. Verma, M. Khetan, and R. K. Sangwan, “Challenges, Barriers, and Facilitators in Telemedicine Implementation in India: A Scoping Review,” Cureus, vol. 16, no. 8, art. e67388, 2024, doi: 10.7759/cureus.67388.

[12] A. Parthasarathi et al., “Exploring the potential of telemedicine for improved primary healthcare in India: a comprehensive review,” Lancet Reg. Health – Southeast Asia, vol. 27, art. 100431, 2024, doi: 10.1016/j.lansea.2024.100431.

[13] A. Bassi et al., “Current Status and Future Directions of mHealth Interventions for Health System Strengthening in India: Systematic Review,” JMIR mHealth uHealth, vol. 6, no. 10, art. e11440, 2018, doi: 10.2196/11440.

[14] T. M. Rasekaba, P. Pereira, V. Rani G., R. Johnson, R. McKechnie, and I. Blackberry, “Exploring Telehealth Readiness in a Resource Limited Setting: Digital and Health Literacy among Older People in Rural India (DAHLIA),” Geriatrics, vol. 7, no. 2, art. 28, 2022, doi: 10.3390/geriatrics7020028.

[15] V. Vaidya et al., “Healthcare in the Modern Era: Launching a Telemedicine-Based OPD Consultation in Rural Pune (Process, Results, and Challenges),” Cureus, vol. 16, no. 5, art. e60310, 2024, doi: 10.7759/cureus.60310.

[16] Z. He, B. Bhasuran, Q. Jin, S. Tian, K. Hanna, C. Shavor, L. G. Arguello, P. Murray, and Z. Lu, “Quality of Answers of Generative Large Language Models Versus Peer Users for Interpreting Laboratory Test Results for Lay Patients: Evaluation Study,” J. Med. Internet Res., vol. 26, art. e56655, 2024, doi: 10.2196/56655.

[17] L.-L. Liao, L.-C. Chang, and I.-J. Lai, “Assessing the Quality of ChatGPT’s Dietary Advice for College Students from Dietitians’ Perspectives,” Nutrients, vol. 16, no. 12, art. 1939, 2024, doi: 10.3390/nu16121939.

[18] V. Ponzo, I. Goitre, E. Favaro, F. D. Merlo, M. V. Mancino, S. Riso, and S. Bo, “Is ChatGPT an Effective Tool for Providing Dietary Advice?,” Nutrients, vol. 16, no. 4, art. 469, 2024, doi: 10.3390/nu16040469.