INFM 412 CAPSTONE PROJECT PROPOSAL

By: Melissa Pacheco

Dr. Long

February 28, 2026

BACKGROUND & MOTIVATION

Robin's Nest Nature School is a nature-based preschool where daily communication between the school and families is especially important. Because the program operates outdoors in varying weather conditions, parents need regular updates about attendance, pickup plans, and daily logistics. Currently, communication is handled through informal methods such as social media posts, text messages, and in-person conversations.

Pickup coordination is one of the clearest examples of how informal communication can create confusion. Right now, parents arrive at the entrance and notify one teacher, who then relays the message by walkie talkie to the teacher supervising the children. A second teacher may be monitoring the gate without a walkie talkie, and the process relies heavily on memory, verbal relay, and staff recognition. During busy pickup times, children may approach the gate simply because they see their parent, even if they were not officially called, which can slow the process down and create avoidable safety concerns.

While these methods work to a degree, they are not centralized or structured. Important updates can be missed, privacy concerns may arise when sharing information online, and there is no dedicated system for organizing attendance or pickup changes. As someone closely familiar with the school environment, I have observed how a structured, secure mobile application could significantly improve clarity, consistency, and privacy.

This project is motivated by the goal of designing and implementing a full-stack Android application that centralizes communication using cloud-based authentication and persistent data storage.

With daily coordination required between multiple families and educators, the lack of a centralized system increases the risk of missed updates and inconsistent record keeping.

PROBLEM STATEMENT

Robin's Nest Nature School does not have a secure, centralized digital system to manage parent communication, attendance reporting, and pickup coordination. The absence of a structured mobile platform leads to inconsistent communication, absence of persistent record tracking and structured workflow, and privacy concerns when information is shared informally.

OBJECTIVES

The primary objectives of this project are to:

- Design and implement a mobile application that centralizes parent–school communication.
- Develop a fully functional Android app using Kotlin.
- Implement secure user authentication using Firebase Authentication.
- Store and retrieve attendance and pickup data using a cloud-based database.
- Demonstrate full-stack development by integrating front-end UI with back-end cloud services.
- Deliver a working cloud-connected prototype.

The project will be developed in two phases:

Phase 1: Core Version (Planned Deliverables)

- Login and Signup using Firebase Authentication
- Persistent login state
- Welcome/Home dashboard
- Attendance reporting stored in Firestore
- Pickup plan submission stored in Firestore
- Navigation menu
- External hyperlink (school website or weather resource)
- Full cloud data storage and retrieval

This version ensures that the application is fully functional and cloud connected.

Phase 2: Enhancements (If Time Permits)

- Weather API integration
- Clothing/gear recommendation logic
- Private photo feed
- Firebase Storage for image uploads
- Role-based access (Parent vs Admin view)
- Improved UI/UX design and layout refinement

The Phase 1: Core Version ensures all required components are fully functional. The Phase 2: Enhancements will be added only after the base system is stable. The project will prioritize completing and testing the core cloud-connected version first, and enhancements will only be added after the base system is stable.

METHODOLOGY & DESIGN

Platform & Tools

- Android Studio
- Kotlin
- XML Layouts (Activities and Fragments)
- Firebase Authentication
    - Firebase Authentication enforces secure identity management using token-based authentication, ensuring only authorized users access protected resources.
- Cloud Firestore
- Firebase Storage
- Weather API
- GitHub for version control

The application will follow a client–cloud architecture in which the Android application serves as the front-end interface, and Firebase services provide authentication and persistent cloud data storage.

FRONT END DESIGN

The front end consists of the visible user interface components:

Login Page

- Email and password fields
- Login button
- Link to Signup page

Signup Page

- Email, password, confirm password
- Create account button

Welcome/Home Page

- Welcome message
- Buttons/icons linking to Attendance, Pickup, Updates, and Resources
- Menu for navigation

Attendance Page

- “Sick Today” button
- “Absent Today” button
- Optional note field
- Confirmation message

Pickup Page

- Pickup person name field
- Relationship field
- Save button

Resources Page

- Button that opens an external website using an Intent

The layout will be clean and structured using ConstraintLayout or LinearLayout.

BACK-END LOGIC

In this project, the back end consists of cloud-based services and application logic that manage authentication and data persistence.

This includes:

- Firebase Authentication for account creation and secure login
- Cloud Firestore for storing attendance and pickup data
- Session persistence across app launches, so users stay signed in securely and do not need to log in every time they open the app, unless they intentionally log out.
- Firebase Storage for image hosting (A-Level)
- API integration for weather data (A-Level)

This architecture ensures that user data is securely stored in the cloud and retrievable across sessions, fulfilling full-stack development requirements.

App Flow

---

1. User launches app → Login screen
2. User selects:
    - Log In → Firebase Authentication validation → Navigate to Home
    - Sign Up → Create account in Firebase → Navigate to Home
3. From Home page, user selects:
    - Attendance → Submit status → Data written to Firestore → Confirmation → Return to Home
    - Pickup → Submit pickup info → Data written to Firestore → Confirmation → Return to Home
    - Daily updates → photo feed showcasing children's activities and highlights
4. Menu allows navigation between screens
5. Logout clears session and returns user to Login screen

Experimentation, Testing, Evaluation, & Validation

Evaluation of the application will focus on functionality, cloud integration, and usability.

Functional Testing

- Validate Firebase Authentication (account creation, login, logout)
- Confirm navigation operates without crashes
- Verify correct form validation and error handling

Integration Testing

- Confirm successful writes to Firestore
- Confirm data retrieval from Firestore
- Verify session persistence across app restarts
- Test image upload and retrieval (A-Level)

Usability Testing

- Conduct scenario-based testing:
    - Report a sick day
    - Submit pickup authorization
    - View updates or weather
- Measure task completion time
- Identify navigation confusion

Evaluation Metrics

- Authentication success rate ≥ 95% during testing
- Firestore write/read success rate ≥ 95%
- Primary task completion within 60 seconds
- Cloud data persists across sessions and app restarts
- No crashes during complete user workflow

The project will be considered successful if the application is fully functional and cloud-connected, meaning authentication works reliably, attendance and pickup data are consistently written to and retrieved from Firestore, and user data persists across sessions without crashes during the core workflow.

Project Planning & Scheduling

Week	Plan
Week 1	Finalize proposal and system architecture
Week 2	Configure Firebase project and authentication
Week 3	Implement login and signup with Firebase
Week 4	Develop dashboard and navigation
Week 5	Implement attendance feature with Firestore integration
Week 6	Implement pickup feature with Firestore integration
Week 7	Complete core integration testing (core version stable)
Week 8	Implement enhancements (Weather API, Firebase Storage)
Week 9	Final integration testing and debugging
Week 10	Final validation and presentation preparation

REFERENCES

Android Developers. (2026). *Android developer guides*. https://developer.android.com/guide

Codes Easy. (2022, December 7). *Login and registration using Firebase in Android* [Video]. YouTube. https://www.youtube.com/watch?v=QAKq8UBv4GI

Firebase. (2026). *Firebase Authentication documentation*. Google. https://firebase.google.com/docs/auth

Firebase. (2026). *Cloud Firestore documentation*. Google. https://firebase.google.com/docs/firestore

Firebase. (2026). *Firebase Storage documentation*. Google. https://firebase.google.com/docs/storage

OpenWeather. (2026). *OpenWeather API documentation*. https://openweathermap.org/api

Phillips, D., Stewart, D., & Marsicano, K. (2021). *Head First Android Development* (3rd ed.). O'Reilly Media.