Robin's Nest Parent App

Melissa Pacheco

INFM 412 Capstone Project

Full Stack Android Application

Background & Motivation

- Nature-based preschool with daily parent–school coordination
- Weather & outdoor setting increases time-sensitive communication needs
- Current tools are informal: texts, calls, in-person updates, Facebook group for photos/updates
- Pickup workflow is manual and inconsistent (verbal relay, walkie talkies, recognition-based)
- No centralized, secure system or persistent record for attendance, pickup, and updates
- Need a secure, cloud-connected workflow with a persistent record

Problem Statement

Communication is spread across texts, calls, in-person messages, and social platforms.

This results in:

- No centralized workflow for attendance, pickup, or updates
- Missed or delayed messages
- No persistent record of attendance or pickup plans
- Limited accountability and traceability
- Privacy concerns when sharing child photos and updates

Why This Project Matters

- Improves clarity with one centralized place for attendance and pickup updates
- Creates accountability through persistent, time-stamped cloud records
- Provides a private, secure alternative to social media for photos and daily updates
- Protects student and family privacy
- Demonstrates real-world full-stack mobile development (auth + cloud storage)
- Built on scalable client–cloud architecture for future expansion

Project Objectives

Develop full-stack Android mobile application

- Implement Firebase Authentication
- Integrate Cloud Firestore for persistent storage
- Deliver working cloud-connected prototype
- Enhance functionality if time permits
- Demonstrate secure client–cloud architecture integration

Methodology & Design

Methodology

- Agile, iterative development approach
- Modular feature implementation (Attendance → Pickup → Enhancements)
- Functional, integration, and usability testing

Technologies & Tools

- Kotlin (Android development)
- Firebase Authentication (User identity management)
- Cloud Firestore (Persistent database storage)
- Firebase Storage (Media handling)
- Android Studio

High-Level Design

- Client–cloud architecture
- Android client communicates securely with Firebase backend
- Role-based access control (future enhancement)
- Scalable architecture to support additional features

Phase 1: Core Version

Firebase Authentication (Login / Signup)

- Persistent login session handling
- Home dashboard with structured navigation
- Attendance reporting stored in Firestore
- Pickup plan submission stored in Firestore
- Full cloud data storage and retrieval
- Core system stabilization and integration validation

Phase 2: Enhancements

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

System Architecture

Android Client (UI + Kotlin Logic)

- Firebase Authentication (Identity Management)
- Cloud Firestore (Database)
- Firebase Storage (Media Files)
- External Weather API (Optional Enhancement)

Application Flow

User launches app → Login screen

User selects:

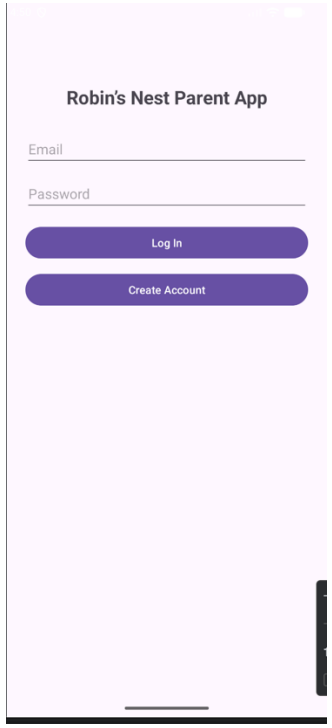
- **Log In** → Firebase Authentication validation → Navigate to Home
- **Sign Up** → Create account in Firebase → Navigate to Home

From Home page, user selects:

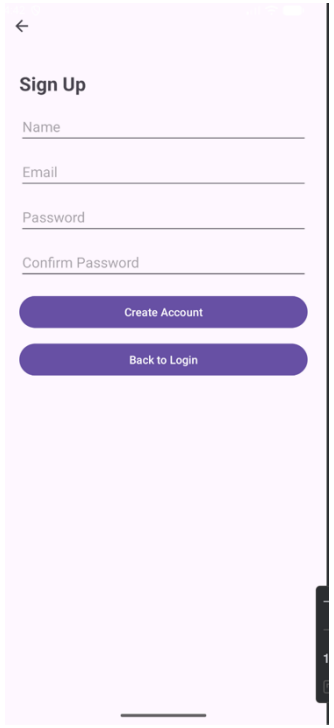
- **Attendance** → Submit status → Data written to Firestore → Confirmation → Return to Home
- **Pickup** → Submit pickup info → Data written to Firestore → Confirmation → Return to Home
- **Weather** → Opens external link in browser

Menu allows navigation between screens

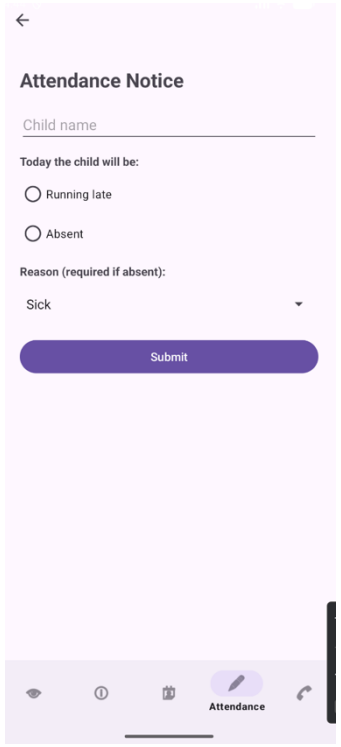
Logout clears session and returns user to Login screen



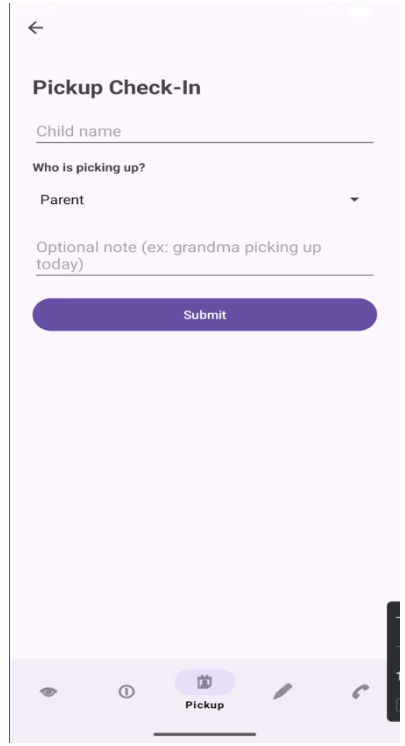
Login



Sign Up



Attendance



Pickup

Testing & Validation

Functional Testing

• Validate account creation, login, logout
• Verify form validation and navigation flow

Integration Testing

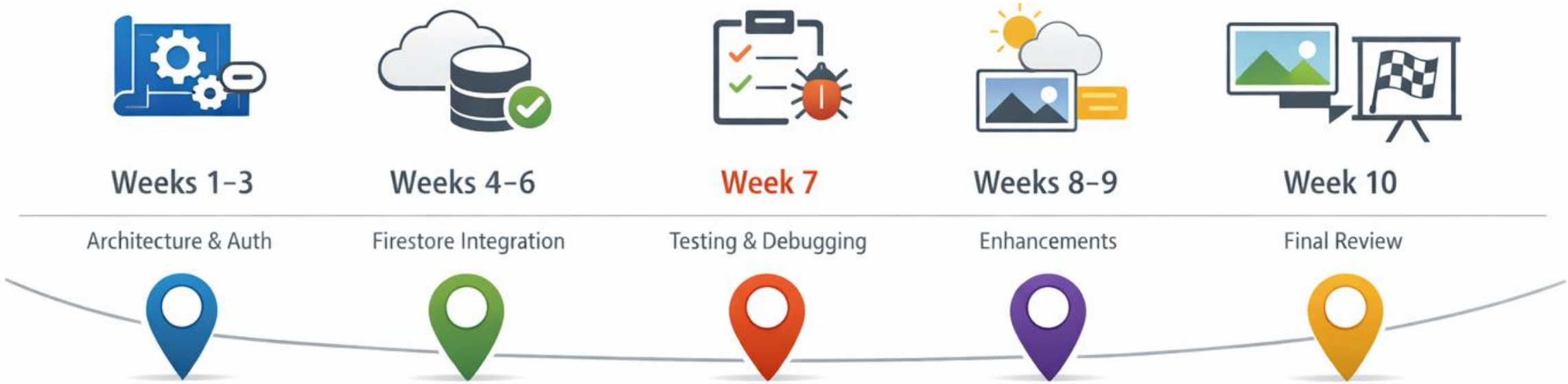
• Confirm successful writes and reads from Firestore
• Validate session persistence across restarts

Usability Testing

• Scenario-based tasks (Report absence, Submit pickup)
• Measure task completion success
• Identify navigation confusion points

Evaluation Metrics

• Persistent cloud data across sessions
• No crashes during full user flow
• Task completion within three interactions



Project Timeline

- **Weeks 1–3**
  Architecture design and Firebase authentication setup
- **Weeks 4–6**
  Cloud Firestore integration for attendance and pickup features
- **Week 7**
  Core version stabilization and integration testing
- **Weeks 8–9**
  Enhancements (Weather API, Firebase Storage) if time permits
- **Week 10**
  Final validation, refinement, and presentation preparation

Success Criteria

- Fully functional cloud-connected Android application
- Authentication success rate ≥ 95% during testing
- Firestore write/read success rate ≥ 95% for attendance and pickup records
- Persistent data across sessions and app restarts
- No crashes during complete user workflow

Questions

Thank you for your time.

I welcome your questions and feedback.