TECHNISCHE HOCHSCHULE DEGGENDORF

DriveGuard

Overparking Detection & Safety Assistant

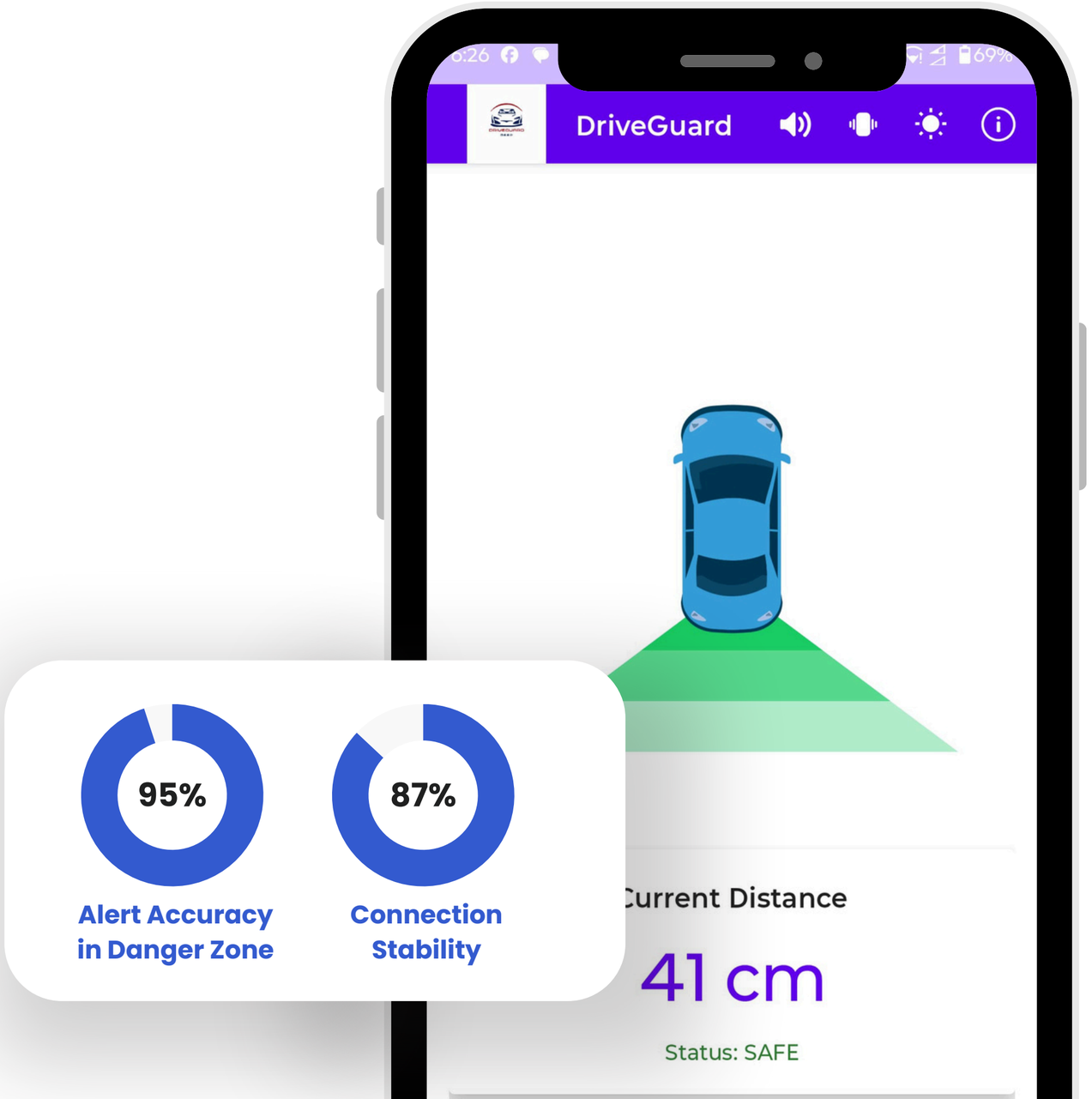
Name: Deep Bharatbhai Savaliya

Matriculation Number: 12501180

Supervisor: Prof. Dr. Goetz Winterfeldt

Course: Mobile Applications & Interaction Design in Vehicles

Submission Date: 09.07.2025



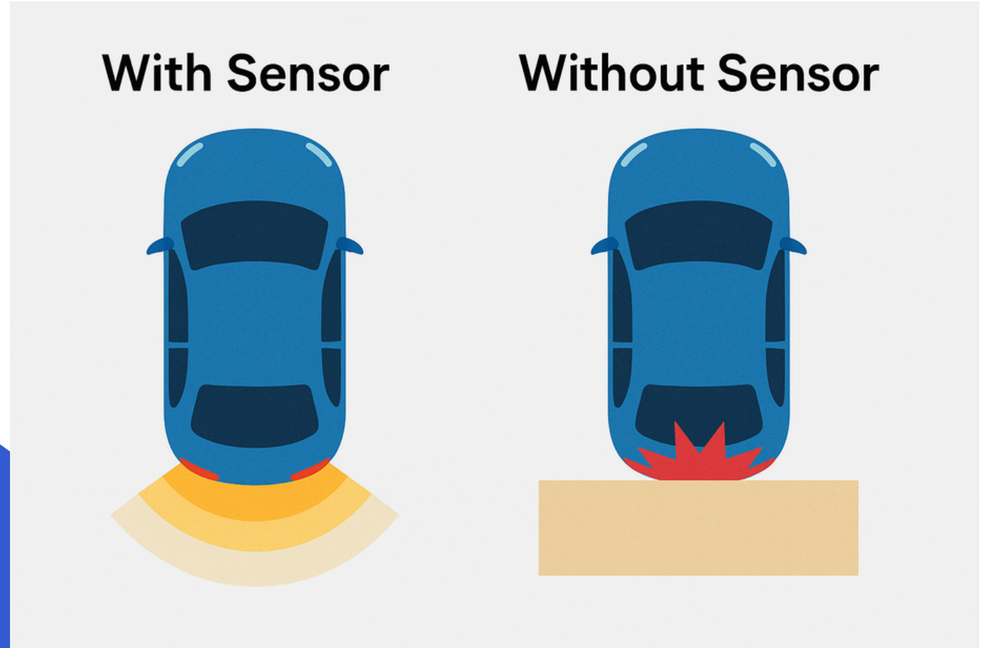
Motivation & Objective

Motivation:

- Rear-end collisions often occur due to poor visibility while parking.
- Many vehicles (especially older or budget models) lack parking assist systems.
- There's a need for a low-cost retrofit solution that is independent of internet access.

Objective:

- Build a real-time proximity alert system using CC3200 and an Android app.
- Communicate via WiFi (no mobile data).
- Provide visual, audio, and vibration alerts in the car while parking.



Target Users

- Private vehicle owners – who want an affordable parking assistant
- Commercial fleet drivers – delivery vans, taxis, who park in tight spots
- Academic/prototype developers – useful in student automotive or IoT projects

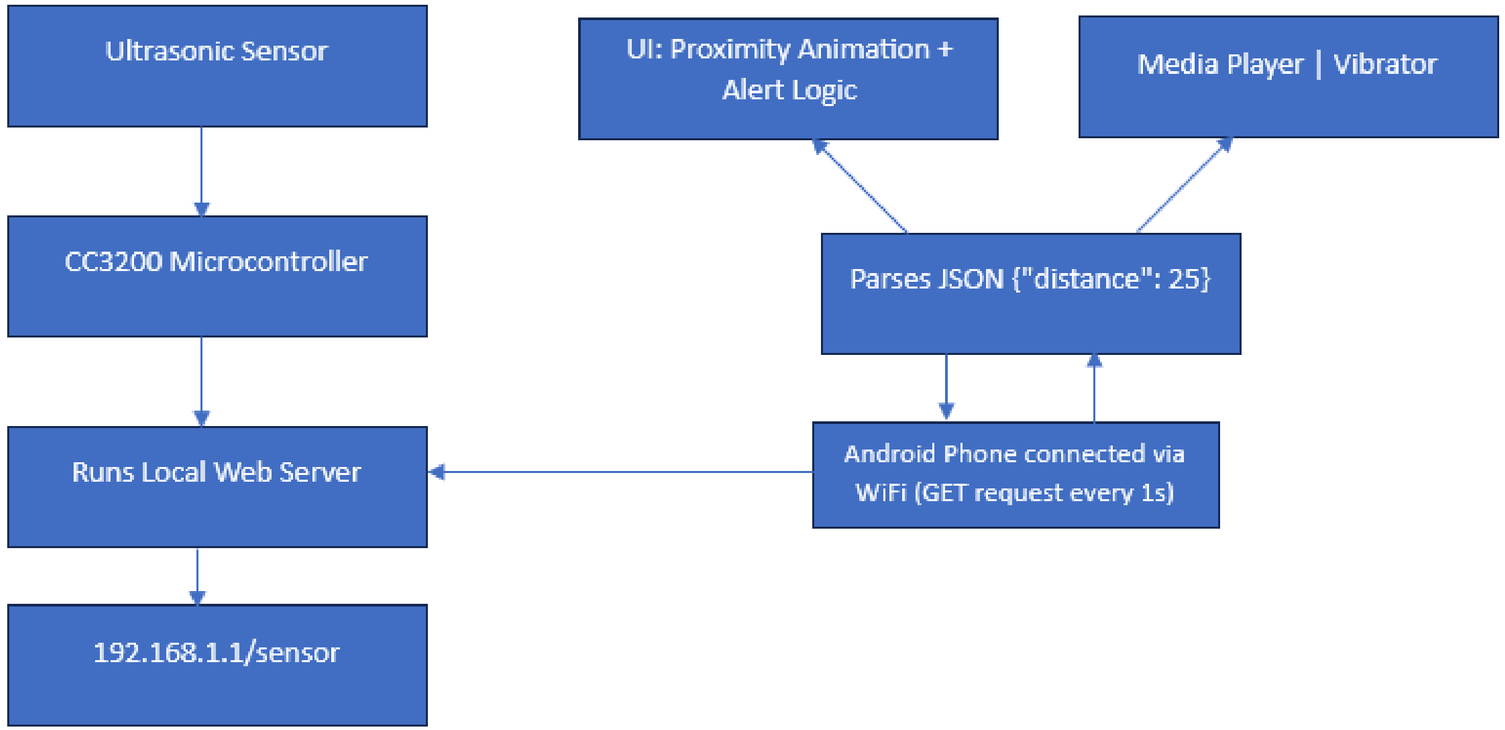
Why it matters:

- Can be integrated into smart vehicle prototypes
- Offers real-time offline operation
- Encourages hands-on learning in embedded and mobile systems

DRIVEGUARD
DEEP

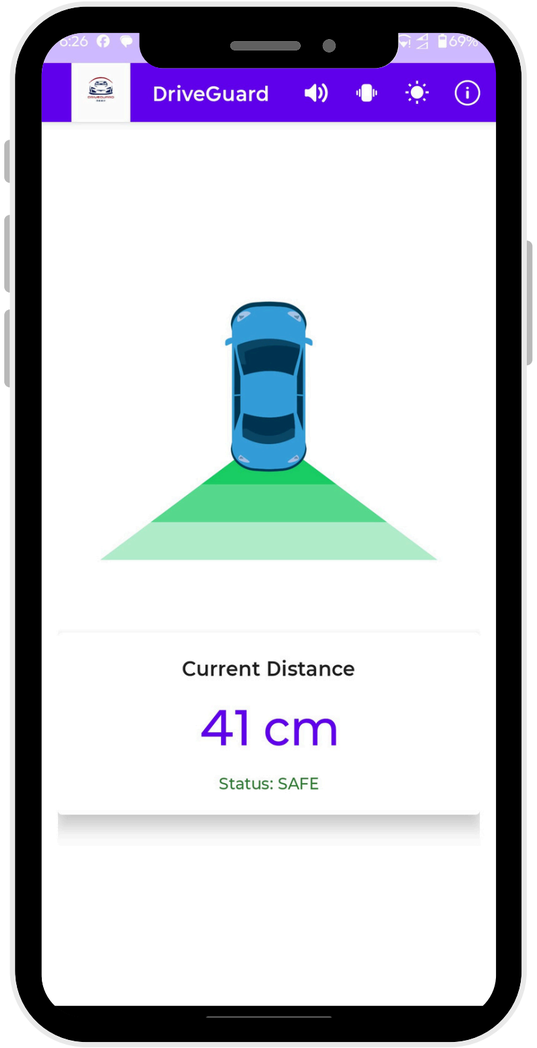
System Overview

- **Hardware:** CC3200 microcontroller + ultrasonic sensor (e.g., HC-SR04)
- **Network:** CC3200 runs as WiFi AP, Android connects to 192.168.1.1
- **Software:** Android app (Kotlin, Jetpack Compose)
  - **Alerts:**
    - Red zone (<10 cm): vibration + buzzer
    - Yellow (10–19 cm): short beep
    - Green (>30 cm): safe
- **CC3200 :**
  - Measures distance
  - Hosts HTTP server
  - Responds with {"distance": value}
- **Android App:**
  - Connects to WiFi
  - Uses Ktor Client to poll /sensor
  - Handles JSON response
  - Displays UI + triggers alerts



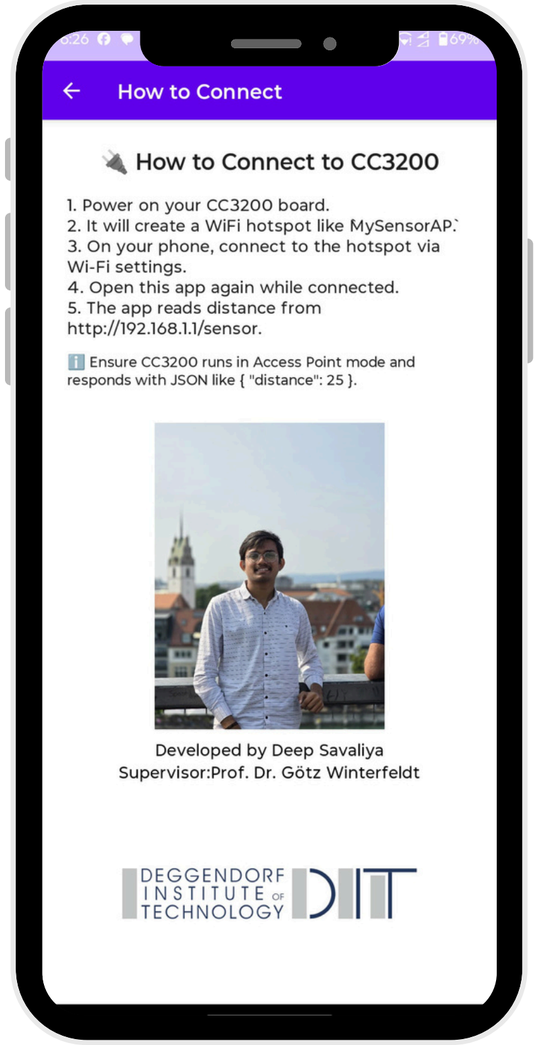
Use Case 1 – Live Rear Proximity Monitoring

- **Trigger**: Driver launches app when connected to sensor
  - **Process:**
    - App sends HTTP GET request every 1 second to /sensor
    - Receives JSON like { "distance": 25 }
    - Updates UI and triggers alert
- **Alert Actions:**
  - UI triangle changes color
  - MediaPlayer plays alert sound
  - Vibration motor triggered if needed



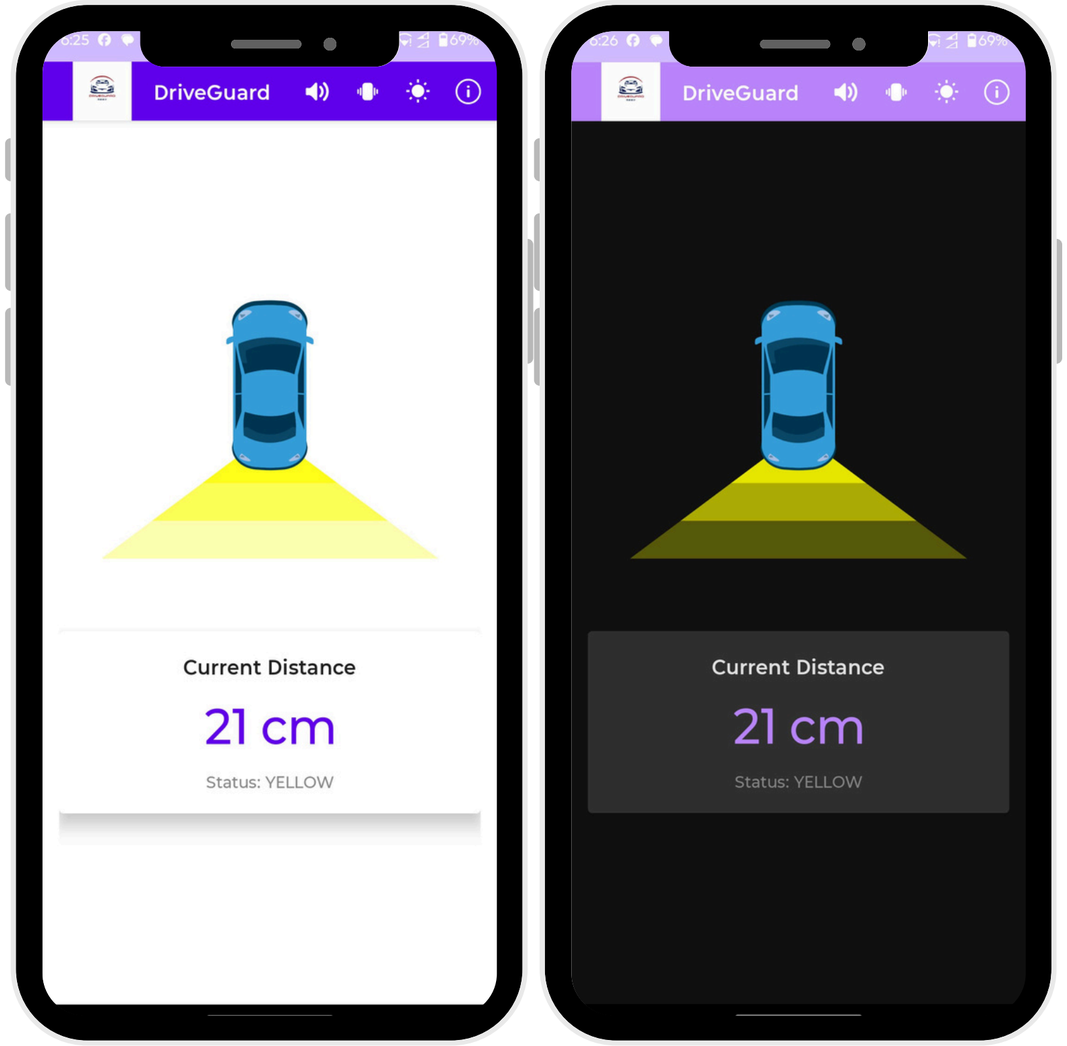
Use Case 2 – WiFi Connection and Instructions

- **New users get guided setup:**
  - Turn on CC3200
  - Connect to WiFi: DistanceMeter
  - Launch app to see live readings
- **StepsActivity displays:**
  - Scrollable list of setup steps
  - Developer name, photo
  - Supervisor and college logos



Use Case 3 – Yellow Zone Alert Logic

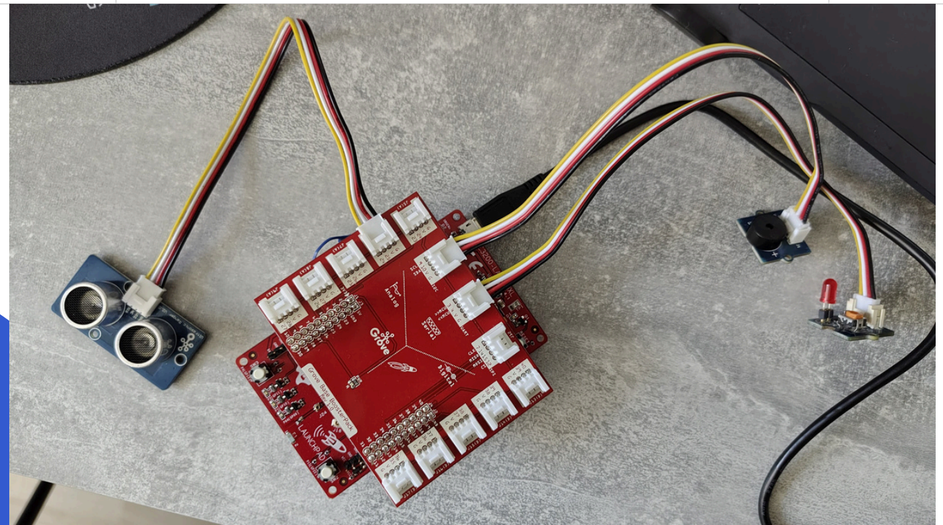
- **Logic**:
  - When distance enters 10–19 cm: one short beep
  - No repeated beeps unless you exit and re-enter
  - Prevents alert spamming and improves user experience
- **Implementation:**
  - Kotlin MediaPlayer instance
  - Zone tracking flag for state change



Circuit Connections for DriveGuard Sensor System

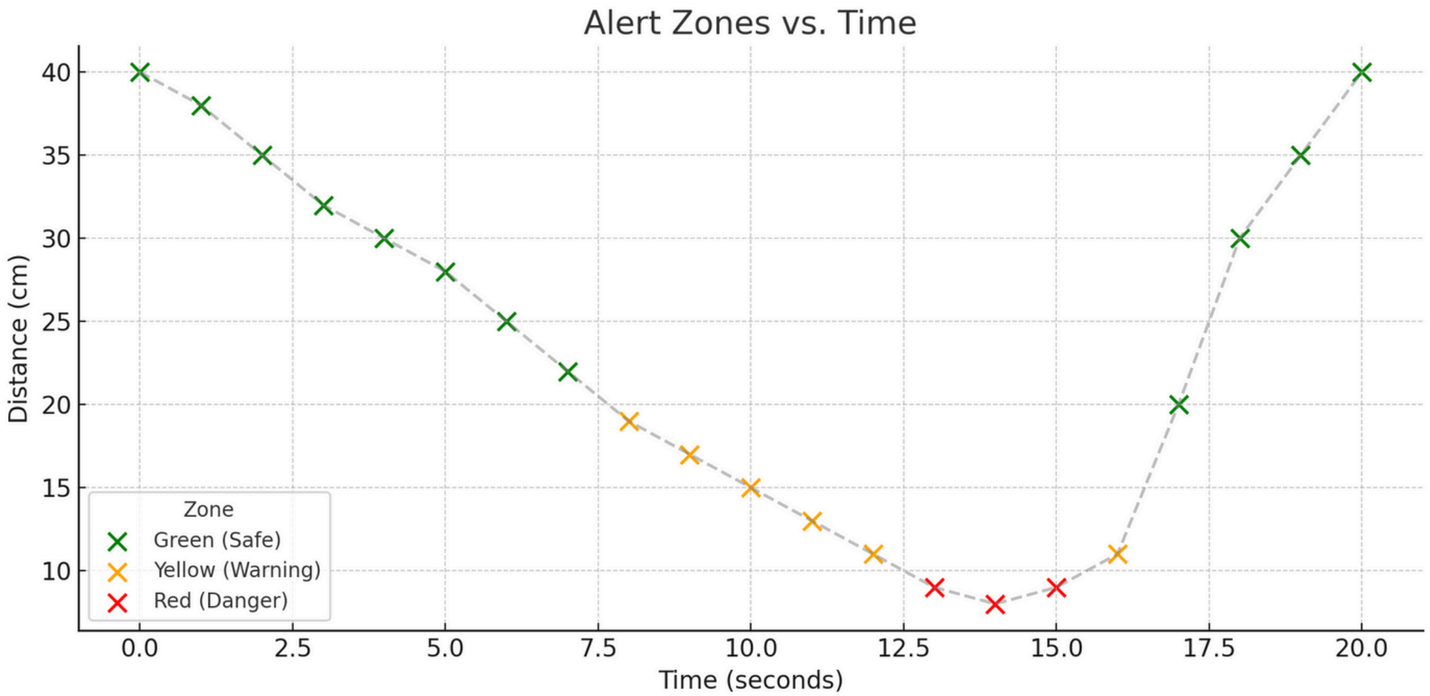
This is the real-world hardware prototype used in the DriveGuard project. It shows the CC3200 LaunchPad microcontroller connected with Grove-based sensors and output devices via a Grove Base BoosterPack.

Module	Port on Grove Base	Function
Ultrasonic Sensor	J7	Measures distance to the rear obstacle (HC-SR04 equivalent)
Buzzer Module	J10	Sounds an alert when distance < 10 cm
LED Indicator	J11	Lights up for visual feedback (e.g., red zone alert)
Grove Base BoosterPack	On top of CC3200	Expands I/O and simplifies sensor connection



Evaluation & Testing

- **Functional Testing:**
  - Sensor ↔ App works in real-time
  - Zone logic accurate
  - Vibration + alert tested
- **Usability:**
  - Tested by peers
  - All instructions easy to follow
  - No crashes on tested devices



Advantages

- Real-time parking feedback using only WiFi – no internet needed
- Retrofit-friendly: works on any vehicle with phone access
- Simple UI with vibration, sound, and animation alerts
- Fully offline-capable via CC3200 WiFi Access Point
- Easily extendable with new features (graphs, logs, etc.)

Limitations & Challenges

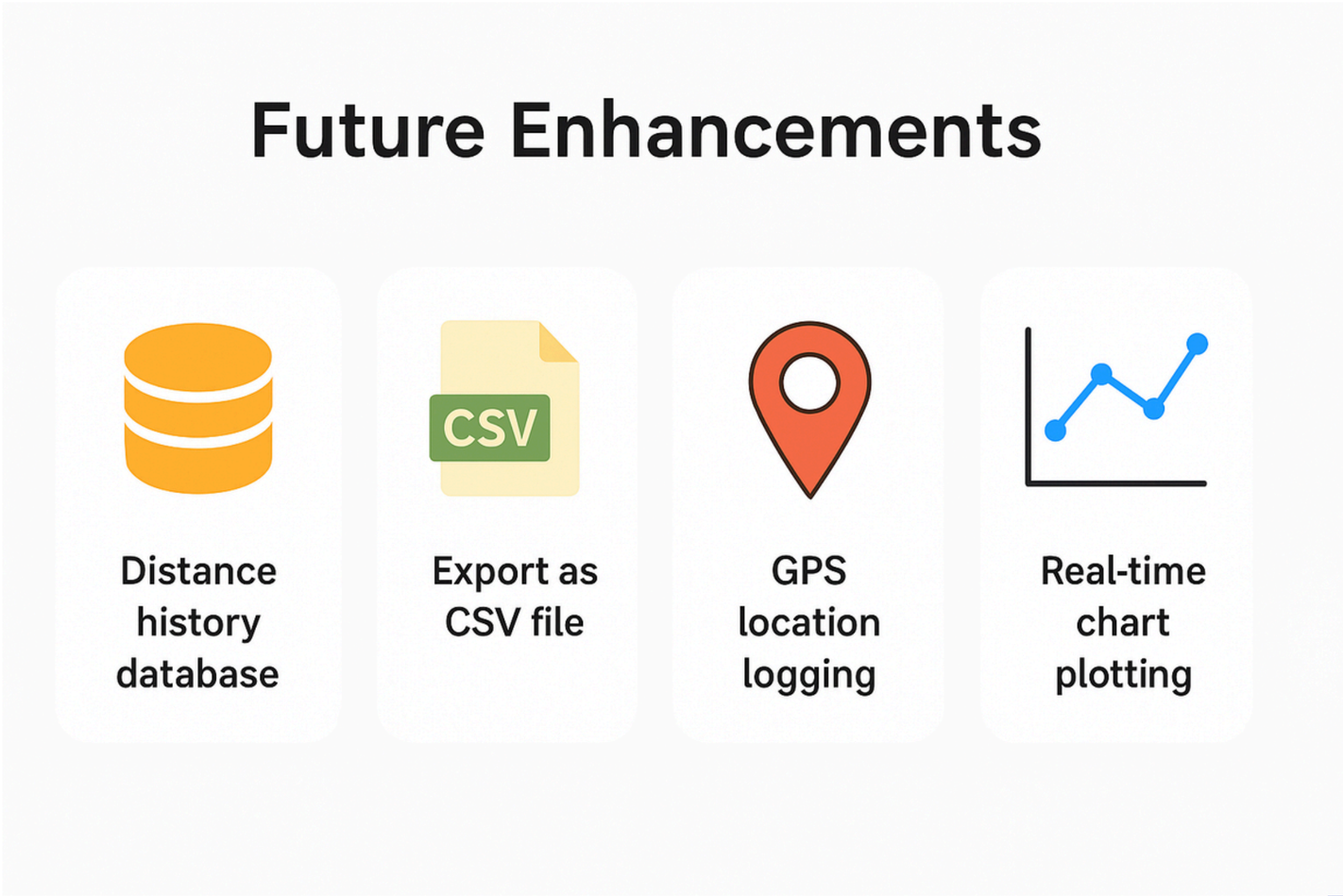
- **Limitations:**
  - WiFi AP disables phone's internet temporarily
  - Single-zone logic may miss dynamic movement nuance
  - Alert delay depends on polling frequency (~1s)
- **Challenges Faced:**
  - Occasional drop in CC3200-Android connection
  - Handling JSON errors during fast polling
  - Synchronizing audio + vibration + UI transitions

Future Improvements

- Add Room DB for distance history storage
- Export alert logs as CSV or PDF
- Add GPS tagging for parking location
- Real-time graphing with MPAndroidChart
- Option to switch to Bluetooth in future versions

Reflection & Future Work

- **What I Learned:**
  - IoT-device to Android communication
  - WiFi Access Point Mode
  - Combining embedded + UI logic
- **Future Plans:**
  - Add Room DB for distance history
  - Export data (CSV/PDF)
  - GPS logging
  - Real-time graph with MPAndroidChart

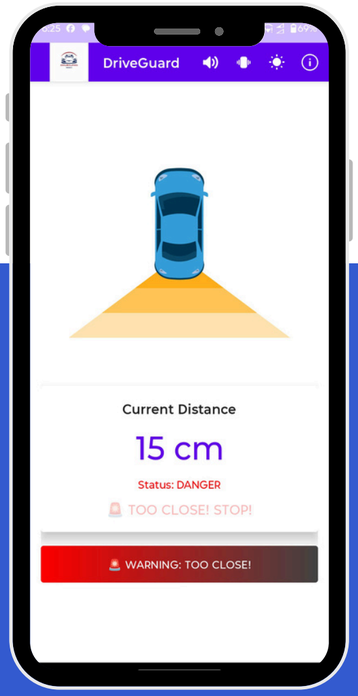


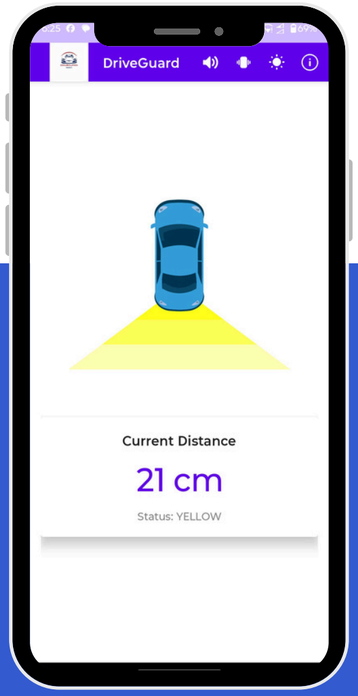
Conclusion

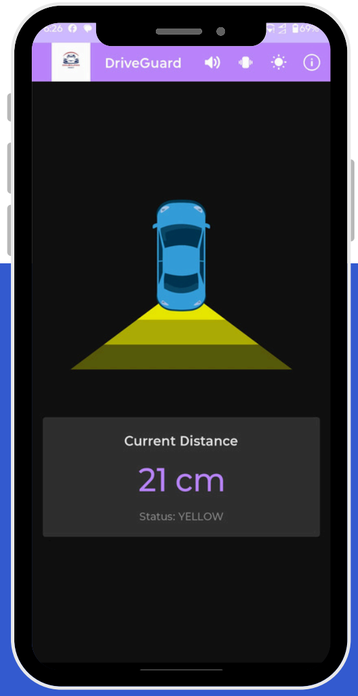
- **Content:**
  - Project succeeded in building a working overparking alert system
  - Fully offline operation via WiFi AP
  - Valuable for smart car retrofitting, education, and IoT demos

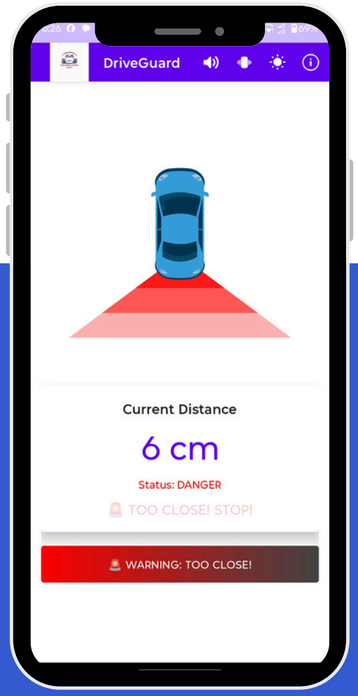
DRIVEGUARD
DEEP

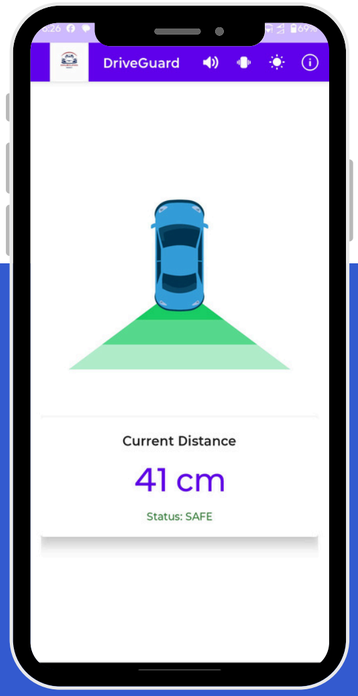
Application Screenshot

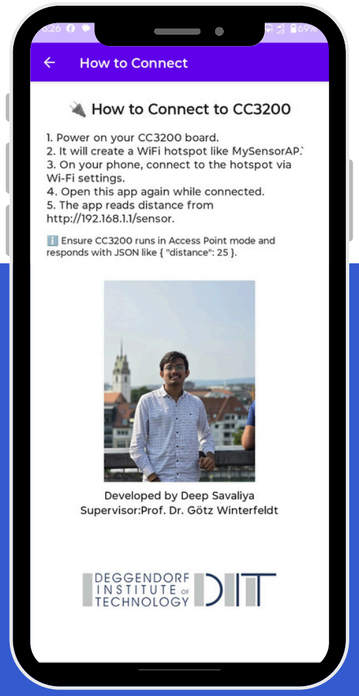






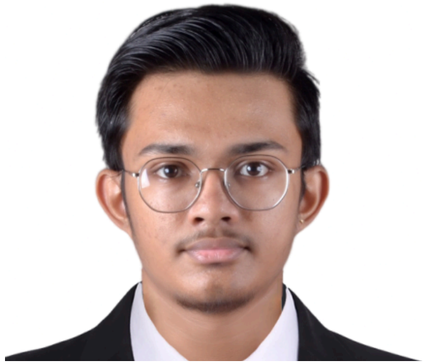






Evaluation Link

Thank You



Name: Deep Bharatbhai Savaliya

Matriculation No. : 12501180

Course: Master in Automotive Software Engineering

Semester: 1