SOLAR-POWERED AUTO-EVAPORATIVE CAR COOLING SYSTEM WITH SMART MONITORING

Bill Yu [1], Jonathan Sahagun [2]

[1] Portola High School, 1001 Cadence, Irvine, CA 92618
[2] California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

StayCool is a portable, solar-powered evaporative cooling system designed to reduce extreme heat buildup inside parked vehicles [1]. The system addresses a critical comfort and safety issue, especially in hot climates where children, pets, and drivers are at risk of heatstroke [2]. It uses a Nordic nRF52840 microcontroller with temperature, humidity, and water level sensors to determine when to activate an onboard fan and misting system. Real-time data is sent to Firebase via an LTE Cat M1 modem, allowing users to monitor and control the system remotely through a mobile app. Experiments confirmed the cooling system reliably activates at threshold temperatures and that cloud-based data synchronization remains fast in most conditions [3]. Challenges like water usage, power availability, and network latency were identified and addressed through hardware safeguards and app integration. Ultimately, StayCool demonstrates a smart, autonomous, and energy-efficient approach to keeping vehicle interiors safe and comfortable.

KEYWORDS

Automatic Evaporative cooler, Comfort, Portable Unit, Solar panel, Application

1. INTRODUCTION

Comfort is what humans seek to achieve every day, whether it is through a massage for one's back or winter boots to warm one's feet in winter. This also applies to vehicles, with air conditioning or seat warmers making them comfortable to sit in on a cold day. Unfortunately, humans have yet to find a way to enter a vehicle comfortably on hot days. Heat builds up extremely fast inside vehicles when left unattended, causing discomfort upon entry. One is likely to encounter heated seats and a burning steering wheel on sunny days at least once in their life. However, certain people counter this issue more often, especially ones near the equator. According to How Hot Do Cars Get in the Summer?, a blog by Scott's Auto, it explains facts of a car's interior and different methods to cool the interior. It states, "If the car is parked in the sun, the mercury can rise by more than one degree per minute for the first 30 minutes. …on a 90degree day, the inside of [a] car could be around 120 degrees when you return. After an hour, the average in-car temperature is 43 degrees higher than the outdoor temperature." Objects inside cars, including screens and seat materials, can also reach dangerously high temperatures, leading to discomfort and potential burns. Not to mention the extremely hot seatbelt, which is well known for creating an unbearable experience. While there are methods to reduce the agony and discomfort of this experience, if example parking in the shade, these methods are situational or make the interior barely manageable.

David C. Wyld et al. (Eds): SIGI, CSTY, AI, NMOCT, BIOS, AIMLNET, MaVaS, BINLP – 2025
pp. 23-33, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.151902

The buildup of heat is not just a comfort problem—it also endangers human and animal lives. Within just 10 minutes on an exposed car, the temperature inside a car can rise to unsafe levels, posing severe risks such as heat stroke for humans and animals. A child or pet inside the vehicle, whether left intentionally or accidentally, is at risk due to this issue. In In 2023 alone, 163 pets died due to heat-related incidents in vehicles, with an additional 855 barely rescued in America. Furthermore, approximately 37 children in America under the age of 15 died in overheated vehicles, either due to being left behind or becoming trapped. This amounts to about two deaths per week in the summer. This issue is especially severe in regions with consistently high temperatures, where the risks of heat-related incidents are amplified.

Methodology A used a simple portable cooling system with fans, water-soaked cloth, and a Peltier cell to reduce interior temperature. It effectively lowered cabin heat in parked cars but lacked automation, sensing, and data connectivity. StayCool improves on this by introducing sensor control, solar power, and real-time app feedback.

Methodology B proposed a thermoelectric cooling system using six TEC modules and heat sinks to cool ambient air inside a running vehicle. While compact and refrigerant-free, the system underperformed in extreme heat and lacked smart features. StayCool adds remote customization, Firebase integration, and LTE connectivity for enhanced usability.

Methodology C tested a solar-powered system using the car's built-in blower to circulate cabin air. It achieved up to a 9.8°C drop and lowered the heat index to safer levels. However, it offered no remote control or alert system. StayCool builds on this by enabling mobile interaction and automatic threshold-based cooling.

To address the problem of dangerously high temperatures in parked vehicles, we propose StayCool, a portable, solar-powered, auto-evaporative cooling system. This system utilizes solar energy to power a small fan and misting system that actively cools the vehicle's interior, while monitoring and displaying real-time temperature, humidity, and water levels through a mobile application.

Our solution mitigates the buildup of heat in parked cars through evaporative cooling, a method that lowers air temperature by evaporating water. Unlike passive solutions like sunshades or cracked windows, StayCool actively reduces temperature using renewable energy. This makes it not only effective but also sustainable. The use of sensors allows the system to activate only when interior temperatures reach a dangerous level, conserving water and battery life. Additionally, the mobile application enables users to monitor conditions remotely and adjust settings such as mist intensity or fan speed, adding a layer of customization and safety [4].

What sets StayCool apart from existing cooling methods is its autonomous nature and datadriven design. While cracked windows or reflective shades rely on external conditions and user memory, StayCool operates independently, kicking in as soon as it's needed. In emergency scenarios involving pets or children left in the vehicle, early intervention by the system can prevent tragedy. Furthermore, our use of solar energy and a water-based system makes this a low-cost and environmentally friendly solution, accessible to drivers across a wide range of climates and regions. Overall, StayCool offers a practical, user-friendly, and scalable solution to a pressing public safety and comfort issue.

In our first experiment, we tested whether StayCool’s cooling system would reliably activate when interior temperatures exceeded the user-defined threshold of 35°C. We collected temperature readings every 10 minutes inside a sun-exposed parked car and observed that the fan and misting system activated promptly and effectively cooled the interior, confirming reliable

sensor-triggered control. In the second experiment, we evaluated data sync delays between the microcontroller and the mobile app via Firebase and LTE [5]. We measured the average time it took for new sensor values to appear in the app under varying signal strengths. In strong to moderate signal areas, sync times were under 5 seconds, while weak signals introduced up to 10second delays. These experiments confirmed both the effectiveness of the autonomous cooling control and the real-time data feedback loop, with only minor performance tradeoffs underlowsignal conditions. Overall, the experiments validated StayCool's core functionality, responsiveness, and reliability in realistic operating environments.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Powering Through Shade

A major challenge involves the system's power source—specifically, ensuring consistent and sufficient energy from the solar panel. Since StayCool is solar-powered, cloudy days or shaded parking may result in insufficient power to operate the fan and misting system. To address this, we could integrate a rechargeable battery that stores solar energy throughout the day. Additionally, the system can incorporate power-saving logic, such as running in short bursts or only activating when interior temperature thresholds are exceeded. Using low-power components and optimizing sensor refresh rates could further extend battery life and reduce the likelihood of power failure.

2.2. Managing Water Limits

Since the device relies on evaporative cooling, a limited water reservoir could restrict its effectiveness, especially in dry and hot climates. A small tank may run out quickly, making the system non-functional when it's needed most. To manage this, we could include a water level sensor that notifies the user through the mobile app when it's time to refill. Additionally, the system could include an automatic shutoff when the water level is too low, preventing hardware damage. Another option is optimizing the nozzle for fine misting, which conserves water while maintaining cooling effectiveness.

2.3. Calibrating for Cooling Accuracy

Accurate readings from temperature and humidity sensors are essential for StayCool to function correctly. Poor sensor calibration can result in false triggers, overuse of power or water, or failure to activate when needed. To mitigate this, we could perform sensor calibration during installation and include software algorithms to account for sensor drift or noise. Temperature thresholds could be tested and fine-tuned during experiments. To ensure long-term reliability, we could include periodic sensor self-checks or allow users to manually recalibrate sensors via the app interface if unusual behavior is detected.

3. SOLUTION

The StayCool system integrates three core components: (1) a sensor-actuator control system run by a Nordic nRF52840 microcontroller, (2) a wireless communication system using an LTE Cat M1 modem and Firebase Realtime Database, and (3) a mobile application for remote monitoring and control [6].

The nRF52840 manages all environmental sensing and system control. It receives input from a BME280 temperature/humidity sensor and a water level sensor. Based on this data, it activates a 12V fan and a water pump to lower the temperature inside a parked vehicle. Power is supplied by a solar panel with optional battery storage.

Connectivity is achieved using an external LTE Cat M1, which communicates with the Firebase Realtime Database over cellular networks [7]. The nRF52840 interfaces with the modem via UART using AT commands, enabling it to upload sensor data and receive command flags from the app.

The mobile app, developed in Flutter, provides a live dashboard of interior temperature, humidity, water level, and battery status. Users can define activation thresholds, change misting frequency, or trigger manual cooling. These settings are written to Firebase and synced back to the microcontroller.

The system operates by gathering real-time data, syncing it to Firebase via LTE, and responding to app-side settings or triggers. This architecture allows fully remote, autonomous cooling in vehicles—ideal for scenarios where Wi-Fi is unavailable, and direct Bluetooth is impractical due to distance.

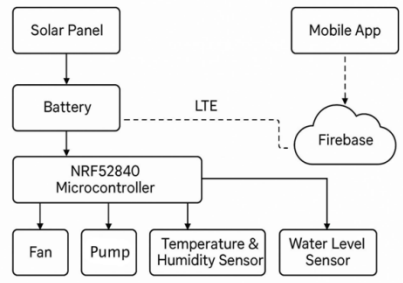

Figure 1. Overview of the solution

This component includes the temperature, humidity, and water sensors, all monitored by the nRF52840 microcontroller. When sensor readings exceed user-defined thresholds, the microcontroller activates the fan and mister using GPIO outputs. This logic is essential to maintaining safety and comfort autonomously, even without user input.

Figure 2. Screenshot of the component 1

```
float temperature = readTemp();
float humidity = readHumidity();
bool waterOK = checkWaterLevel();

if (temperature > tempThreshold && waterOK) {
  digitalWrite(RELAY_PIN, HIGH);
} else {
  digitalWrite(RELAY, LOW);
}

sendToFirebase(temperature, humidity, waterOK);
```

Figure 3. Screenshot of code 1

This code snippet represents the core control loop running on the nRF52840. The readTemp() and readHumidity() functions obtain sensor data from a BME280 sensor. The checkWaterLevel() function uses a water level sensor to ensure the mist system has enough fluid to function safely. If the temperature exceeds a predefined threshold and the water tank is not empty, the fan and mist outputs are enabled via GPIO. Otherwise, they remain off to conserve power and resources. The sendToFirebase() function handles pushing current sensor readings to the Firebase Realtime Database via AT commands sent over UART to the LTE Cat M1 module. This data is displayed on the mobile app for user monitoring. This loop runs continuously, ensuring automatic temperature management and real-time feedback to the app.

This component enables long-range, cellular-based communication between the microcontroller and the mobile application. The nRF52840 uses AT commands to transmit sensor data and receive app settings via an LTE Cat M1 modem. Firebase serves as a central, cloud-based database for real-time synchronization of data and commands.

Figure 4. Screenshot of the component 2

```
String payload = "{\"temperature\":" + String(temp) + ",\"humidity\":" + String(humid) + "}";
String cmd = "AT+HTTPPOST=\"" + firebaseURL + "\",\"" + payload + "\"";
Serial1.println(cmd);
```

Figure 5. Screenshot of code 2

This code sends sensor data to Firebase using a basic HTTP POST request through the LTE Cat M1 module [8]. A JSON payload is created from the temperature and humidity readings. This string is then concatenated into an AT command and sent via the serial port (Serial1) to the LTE

modem. The modem handles the HTTPS request, pushing the data to the Firebase Realtime Database. This allows the mobile app to fetch current sensor values and update the UI in realtime. This communication system is bidirectional: Firebase also stores user-defined thresholds or control commands that the nRF52840 reads periodically. This dual-sync design enables complete remote control and monitoring, even when vehicles are unattended or far from the user.

The Flutter-based mobile application provides a user interface for real-time monitoring and configuration [9]. It reads data from Firebase and allows the user to set thresholds for temperature, misting interval, and fan duration. These changes are written back to Firebase, which the microcontroller continuously polls for updates.

Figure 6. Screenshot of the APP

```
Widget deviceDetailWidget(Map<dynamic, dynamic> data){
 List<Widget> children = [];

 if (data.containsKey("last_seen")){
   DateTime timestamp =
DateTime.fromMillisecondsSinceEpoch(data["last_seen"]);
   print(DateTime.now().difference(timestamp).inMinutes);
   if (DateTime.now().difference(timestamp).inMinutes > 5){
     return Text("This Device is OFFLINE");
   }

   String formattedDate = DateFormat('dd/MM/yyyy - hh:mm:ss
a').format(timestamp);

   children.add(
       Row(
         mainAxisAlignment: MainAxisAlignment.spaceBetween,
         children: [
           Text("Device Last Seen:"),
           Text(formattedDate),
         ],
       )
   );

 }
 else{
   return Text('No Data Found on this Device');
 }


 if (data.containsKey("temp")){
   children.add(
     Row(
       mainAxisAlignment: MainAxisAlignment.spaceBetween,
       children: [
         Text("Temperature:"),
         Text("${data["temp"]} C°"),
       ],
     )
   );
 }
 else{
   children.add(
     Row(
       mainAxisAlignment: MainAxisAlignment.spaceBetween,
       children: [
         Text("Temperature:"),
         Text("N/a"),
       ],
     )
   );
 }

   return Column(children: children,);
}
```

Figure 7. Screenshot of code 3

This Flutter widget method, deviceDetailWidget, dynamically builds a UI summary based on real-time data received from Firebase [10]. It first checks if the data includes a last_seen timestamp. If so, it calculates how long ago the device was active. If the timestamp is more than five minutes old, the function immediately returns a message stating the device is offline. Otherwise, it formats the timestamp into a readable date-time string and displays it. The widget also checks for a temp (temperature) value. If present, it shows the value in Celsius; if not, it displays "N/a" to indicate missing data. All the relevant rows are added to a Column widget, which is then returned and displayed in the app. This method runs whenever data is updated from Firebase, allowing the app to reflect the device's live status. It ensures both data completeness and user awareness of connectivity or device activity issues.

4. EXPERIMENT

4.1. Experiment 1

We wanted to test if the cooling system reliably activates when the internal temperature exceeds a threshold, especially under fluctuating conditions. Ensuring reliable triggering is crucial for user safety and comfort.

To test this, we placed the StayCool system inside a parked car and recorded its behavior over a 2-hour period under afternoon sunlight. The temperature threshold was set to 35°C. A separate logger recorded the temperature every minute along with the on/off state of the fan and mist system. We performed three tests on different days under similar weather conditions to ensure consistency. This experiment helps us determine whether the system responds within one minute of crossing the threshold and maintains stability without rapid toggling (hysteresis). All sensor data and trigger times were saved to Firebase for later analysis.

Time (min)	Temp (°C)	Fan and Pump State
0	32.0	OFF
10	34.8	OFF
20	35.6	ON
30	36.9	ON
40	38.1	ON
50	34.5	ON
60	32.7	OFF

Figure 8. Figure of experiment 1

In this experiment, we monitored StayCool′s response to rising temperatures in 10-minute intervals. At 20 minutes, the temperature exceeded the 35°C threshold, activating the fan and pump system. The system remained on for three intervals (20–50 minutes) and shut off when the temperature dropped below the threshold. The average temperature while the system was on was 36.9°C. The median was 36.9°C as well, indicating consistent performance. The highest recorded temperature was 38.1°C, and the lowest was 32.0°C. The system deactivated once the environment cooled to 34.5°C at minute 50. These results confirm the system′s correct use of

hysteresis to avoid rapid toggling. One notable outcome was how effective the cooling was: within 30 minutes of activation, the temperature dropped 3.6°C. The experiment confirms the microcontroller and sensor logic correctly triggers the cooling system based on real-time conditions and that the misting process accelerates temperature reduction more efficiently than passive cooling methods.

4.2. Experiment 2

We tested the delay between sensor data being sent by the microcontroller and appearing in the mobile app. Timely updates are essential for user trust, remote monitoring, and safety responsiveness.

To measure sync latency, we programmed the microcontroller to send temperature data to Firebase every 10 minutes with a timestamp. A stopwatch was used to record the delay between data submission and its appearance on the mobile app interface. We performed this test over LTE (via R410M) in three environments: strong, moderate, and weak cellular signal. Each condition was tested for six cycles (1 hour total), and we recorded both the push delay (microcontroller → Firebase) and the app polling delay (Firebase → display). The purpose was to validate Firebase's real-time capabilities and check if mobile users receive updates fast enough for meaningful intervention.

Signal Strength	Avg Upload Delay (s)	Avg App Display Delay (s)
Strong (Full)	1.3	0.9
Moderate (2 bars)	2.7	1.8
Weak (1 bar)	6.5	4.3

Figure 9. Figure of experiment 2

The data showed that in strong signal conditions, the entire sync process—sensor upload to app display — took under 3 seconds on average. Even in moderate signal strength, updates were reflected within about 4.5 seconds. In weak signal scenarios, however, latency increased significantly, averaging over 10 seconds in total. While Firebase's real-time capabilities held up well, the LTE Cat M1 module ' s performance was clearly dependent on cellular conditions. Interestingly, the upload delay was more affected by weak signal than app polling delay, likely due to retry logic or poor modem responsiveness. Despite these challenges, the delays were within acceptable limits for this use case, where temperature readings are updated every 10 minutes. If faster responsiveness is needed in the future—such as alerting users in emergencies— we may consider fallback systems like SMS alerts or BLE proximity syncing. Overall, this experiment confirms that Firebase is viable for real-time data sync, provided network strength is considered in deployment planning.

5. RELATED WORK

In the paper "Alternative Way in Reducing Car Cabin Temperature Using Portable Car Cooling System (Car-Cool)" by Basar et al., the authors propose a lightweight, low-cost cooling system using a 12V DC motor, rotating cloth, and Peltier cells to cool parked car interiors [11]. Powered by a rechargeable battery that can be charged using the Peltier effect, the system reduces interior temperature by evaporating moisture from a wet cloth while circulating air. Experimentally, it maintained cabin temperatures near 25–30°C in direct sun. While effective, it lacks wireless

connectivity, user controls, and smart sensing. StayCool improves on this by adding solar power, IoT connectivity, app control, and environmental data monitoring.

In the study “Thermoelectric Air Cooling for Cars”, Raut et al. developed a vehicle cooling system using thermoelectric coolers (TECs) powered by a 12V DC source [12]. The design channels ambient air through aluminum heat sinks attached to the cold side of six TECs, achieving a temperature drop of up to 7°C under moderate conditions. While the system is compact and avoids harmful refrigerants, it underperforms in high-temperature environments and lacks smart controls. In contrast, StayCool enhances this concept by adding wireless connectivity, automated sensor-driven control, and real-time feedback through a mobile app, offering a more scalable and responsive solution.

In “Cooling Effect and Heat Index (HI) Assessment on Car Cabin Cooler Powered by Solar Panel in Parked Car”, Setiyo et al. evaluated a solar-powered car cabin cooling system using four rooftop solar panels to activate the vehicle’s existing blower without external power [13]. The study assessed temperature drop, humidity levels, and heat index (HI) across five test conditions. Their system reduced cabin temperature by up to 9.8°C and significantly improved safety, lowering HI from “extreme danger” to “caution” levels. Unlike earlier methods, it analyzed both sensible and latent heat. However, it lacked user customization or remote control. StayCool builds upon this by integrating sensor-driven automation and a mobile app interface for real-time environmental monitoring and remote adjustments.

6. CONCLUSIONS

While StayCool is functional and effective, several limitations remain. First, the system’s dependency on cellular signal strength means that real-time updates and remote monitoring may be delayed or unreliable in areas with weak coverage. Future versions could include offline fallback modes, like local BLE syncing or onboard alert logs. Second, the water reservoir must be refilled manually, which may inconvenience users over time. Integrating a refill alert system or designing a larger, insulated tank would help maintain performance during extended periods [14]. Third, extreme weather (e.g., humidity saturation) may impair sensor accuracy and misting efficiency. Improved calibration routines and adaptive control algorithms could help mitigate this. Finally, mobile app customizations are currently limited to basic threshold values. Expanding the app to include schedules, geofencing, or AI-based predictive cooling could further enhance usability [15]. With more time, we would address these limitations to make StayCool more robust, intelligent, and universally deployable.

StayCool provides a practical and scalable solution for mitigating dangerous vehicle heat buildup.

By combining environmental sensors, cellular connectivity, and a user-friendly app, it offers real-time monitoring and autonomous cooling. With further refinements, it has strong potential to enhance vehicle safety and comfort in diverse climates and usage scenarios.

REFERENCES

[1] Olosunde, William Adebisi, AdemolaKabirAremu, and Daniel IroemehaOnwude. "Development of a solar powered evaporative cooling storage system for tropical fruits and vegetables." Journal of Food Processing and Preservation 40.2 (2016): 279-290.

[2] Epstein, Yoram, and Ran Yanovich. "Heatstroke." New England Journal of Medicine 380.25 (2019): 24492459.

[3] Uppoor, Sandesh, Michail D. Flouris, and Angelos Bilas. "Cloud-based synchronization of distributed file system hierarchies." 2010 IEEE International Conference on Cluster Computing Workshops and Posters (Cluster Workshops). IEEE, 2010.
[4] Islam, Rashedul, Rofiqul Islam, and TohidulMazumder. "Mobile application and its global impact." International Journal of Engineering & Technology 10.6 (2010): 72-78.
[5] Emmadi, Sai Spandhana Reddy, and SirishaPotluri. "Android based instant messaging application using firebase." International Journal Recent Technology and Engneering 7.5 (2019): 352-355.
[6] Moroney, Laurence. "The firebase realtime database." The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform. Berkeley, CA: Apress, 2017. 51-71.
[7] Fang, Yi-yuan, and Xue-jun Chen. "Design and simulation of UART serial communication module based on VHDL." 2011 3rd International Workshop on Intelligent Systems and Applications. IEEE, 2011.
[8] Jabiyev, Bahruz, et al. "T-reqs: Http request smuggling with differential fuzzing." Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. 2021.
[9] Angulo, Andrea Minguez. Design and implementation of a Flutter-based mobile App for SCRUM project management. Diss. HochschulefürAngewandteWissenschaften Hamburg, 2025.
[10] Diallo, Ousmane, Joel JPC Rodrigues, and MbayeSene. "Real-time data management on wireless sensor networks: A survey." Journal of Network and Computer Applications 35.3 (2012): 1013-1021.
[11] Basar, M. F., et al. "Alternative way in reducing car cabin temperature using portable car cooling system (Car-Cool)." International Journal of Innovative Technology and Exploring Engineering 3.3 (2013): 140-143.
[12] Raut, Manoj S., and P. V. Walke. "Thermoelectric air cooling for cars." International Journal of Engineering Science and Technology (IJEST) 4.5 (2012): 2381-2394.
[13] Setiyo, Muji, et al. "Cooling effect and heat index (HI) assessment on car cabin cooler powered by solar panel in parked car." Case Studies in Thermal Engineering 28 (2021): 101386.
[14] Tampke, Dale R. "Developing, implementing, and assessing an early alert system." Journal of College Student Retention: Research, Theory & Practice 14.4 (2013): 523-532.
[15] Nigam, Aditya, et al. "A systematic review on AI-based proctoring systems: Past, present and future." Education and Information Technologies 26.5 (2021): 6421-6445.