AN INTELLIGENT CROSS-PLATFORM MOBILE APPLICATION FOR REAL-TIME CPR TRAINING AND EMERGENCY CARDIAC ARREST RESPONSE USING ACCELEROMETER-BASED FEEDBACK AND CLOUD-DRIVEN COMMUNITY ALERTING

Weiting Wang [1], Jonathan Sahagun [2]

[1] St. Margaret's Episcopal School, 31641 La Novia Ave, San Juan Capistrano, CA 92675
[2] California State University, Los Angeles, 5151 State University Dr, Los Angeles, CA 90032

ABSTRACT

Sudden cardiac arrest claims approximately 350,000 lives annually in the United States, with survival heavily dependent on immediate bystander CPR and rapid AED deployment. Heart Angel is a cross-platform mobile application built with Flutter that addresses this challenge through three integrated components: an accelerometer-based CPR feedback engine providing real-time compression quality assessment via audio and visual cues, a GeoJSON-powered AED locator displaying defibrillator locations on an interactive map, and a Firebase Cloud Functions-driven emergency notification system that dispatches GPS-targeted alerts to nearby CPR-certified community responders. Key challenges including accelerometer noise filtering, audio-visual synchronization, and secure notification delivery were resolved through configurable threshold calibration, timer-synchronized evaluation cycles, and server-side Cloud Function architecture. Experimental evaluation demonstrated 85.3% overall compression classification accuracy across 150 manikin-based trials. Compared to existing standalone CPR training or AED locator applications, Heart Angel provides a unified platform bridging training and real-world emergency response, empowering laypersons to act effectively during cardiac emergencies.

KEYWORDS

Cardiopulmonary Resuscitation (CPR), Mobile Health (mHealth), Accelerometer-Based Feedback, Automated External Defibrillator (AED) Locator, Emergency Push Notifications

1. INTRODUCTION

Sudden cardiac arrest (SCA) remains one of the leading causes of death worldwide, claiming approximately 350,000 lives annually in the United States alone [1]. Out-of-hospital cardiac arrest (OHCA) presents a particularly dire prognosis, with survival rates hovering around only 10% for EMS-treated cases. The critical determinant of survival lies in the immediate administration of high-quality cardiopulmonary resuscitation (CPR) by bystanders prior to the arrival of emergency medical services. Research demonstrates that bystander-initiated CPR can double or even triple survival rates [2]. Despite this well-established evidence, bystander CPR rates remain

David C. Wyld et al. (Eds): CSITEC, NLPI, CMCA, NECOM, DMML, SP, ADCO – 2026
pp. 91-100, 2026. CS & IT - CSCP 2026 DOI: 10.5121/csit.2026.160808

suboptimal; only 40.2% of OHCA victims in 2023 received bystander CPR [1]. A primary barrier to bystander intervention is the lack of confidence and inadequate training among laypersons [3]. Traditional CPR training courses, while effective, suffer from skill decay over time, with studies showing significant deterioration of CPR competency within three to six months following certification [4]. Furthermore, during actual emergencies, even trained individuals frequently deliver compressions at incorrect rates or insufficient depths due to the stress of the situation [5]. The American Heart Association (AHA) guidelines recommend a compression rate of 100–120 compressions per minute at a depth of approximately 5–6 centimeters for adult victims [6]. Achieving and maintaining these parameters without real-time guidance is exceedingly difficult. Additionally, the rapid location of the nearest automated external defibrillator (AED) remains a significant challenge during cardiac emergencies, as early defibrillation combined with CPR dramatically improves survival outcomes [7]. There exists a pressing need for accessible, technology-driven solutions that can both train laypersons in proper CPR technique and provide real-time assistance during actual cardiac emergencies.

Song et al. developed a smartphone CPR feedback application using accelerometer data that demonstrated improved compression quality metrics during practice sessions. However, their system lacked emergency response integration, AED mapping, and multilingual support, limiting its utility beyond isolated training scenarios. Berger et al. conducted a controlled trial of bystander CPR apps and found that while these applications improved compression quality, they introduced delays in CPR initiation. Heart Angel mitigates this through a streamlined emergency workflow and immediate metronome activation upon entering the CPR screen. Auricchio et al. evaluated mobile AED geolocalization and found that app-based approaches did not significantly outperform verbal directions, partly due to incomplete AED databases. Heart Angel addresses this by leveraging OpenStreetMap's crowdsourced, continuously maintained AED dataset and by integrating the locator within the same CPR guidance platform, eliminating the need to switch between applications during emergencies.

Heart Angel is a cross-platform mobile application developed using the Flutter framework that addresses the CPR training gap and emergency response deficiency through an integrated, sensor-driven approach. The application leverages the smartphone's built-in accelerometer to detect and evaluate chest compression quality in real time, providing immediate audiovisual feedback on compression depth and rate during both practice sessions and live emergency scenarios [8]. Unlike existing CPR training applications that rely solely on instructional content, Heart Angel combines three critical functionalities into a unified platform: (1) an accelerometer-based practice mode with a metronome-guided rhythm system and real-time force gauge, (2) a step-by-step emergency response workflow with audio-guided CPR instructions and live compression monitoring, and (3) a GeoJSON-powered AED locator utilizing Google Maps to display nearby defibrillator locations [9]. The application further incorporates a Firebase Cloud Functions-based emergency notification system that leverages GPS-based geolocation to alert nearby CPR-certified community members when a cardiac emergency is detected [10]. This community-responder dispatch approach is modeled after systems shown to enhance OHCA survival rates [11]. Heart Angel supports bilingual operation in English and Chinese, broadening accessibility across diverse communities. The application stores user performance history locally, enabling longitudinal tracking of CPR skill development. By combining evidence-based training protocols aligned with AHA guidelines, real-time sensor feedback validated by prior research, and a cloud-connected emergency alert infrastructure, Heart Angel offers a more comprehensive solution than existing standalone CPR training or AED locator applications [16]. The system is designed to be always available, functioning offline during emergencies when internet connectivity may be unreliable.

The experimental evaluation assessed the reliability of Heart Angel's accelerometer-based compression classification system. Fifty compressions were performed at each of three distinct force levels—light, correct, and excessive—on a CPR training manikin, totaling 150 trials. The application's feedback engine classified each compression in real time using configurable displacement thresholds derived from double integration of accelerometer data. The system achieved an overall accuracy of 85.3%, with correct-force compressions identified most reliably at 92% accuracy, followed by excessive-force at 84%, and light-force at 80%. The reduced accuracy for light compressions resulted from small displacement values approaching the sensor noise floor, a known limitation of MEMS accelerometers in consumer devices. Device-specific sensor sampling rate variability also influenced displacement calculation precision. These results are consistent with the ±3.8 mm depth estimation accuracy reported in comparable smartphone-based CPR feedback research, confirming that consumer-grade accelerometers provide sufficiently accurate feedback for CPR training purposes.

2. Challenges

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

2.1. Accelerometer Accuracy for Compression Detection

A significant challenge involves reliably interpreting accelerometer data from diverse smartphone hardware to accurately assess CPR compression quality. Different devices produce varying sensor noise levels and sampling rates, which could introduce inconsistencies in displacement estimation. To address this, a configurable threshold system could be implemented, allowing calibration of the acceleration threshold, the "too much force" threshold, and the "too little force" threshold through a settings interface. Additionally, a noise-filtering algorithm that disregards acceleration values below a minimum threshold could reduce false positives from incidental device movement, ensuring that only deliberate compression motions are registered [17].

2.2. Real-Time Audio-Visual Synchronization

Coordinating the metronome audio playback with visual compression indicators and simultaneous accelerometer data processing presents a concurrency challenge. If audio cues lag behind visual prompts, or if sensor readings are processed asynchronously relative to the beat cycle, user experience degrades and compression timing suffers. To resolve this, a timer-based architecture could synchronize the metronome beat interval, image toggle, and compression evaluation within a unified periodic callback. By anchoring the compression evaluation timer to the same millisecond-per-compression interval as the audio metronome, the system could ensure that force measurements are assessed precisely at each beat boundary [12].

2.3. Secure Emergency Notification Delivery

Implementing a push notification system that reliably delivers time-critical emergency alerts to nearby CPR-certified responders introduces security and latency concerns. Embedding server credentials in client-side code would expose API keys to extraction. Network latency during emergencies could delay critical alerts. A Firebase Cloud Functions architecture could move all notification logic server-side, with the client invoking a secure HTTPS callable function that handles user authentication, geospatial querying, and Firebase Cloud Messaging dispatch [10]. The Haversine formula could be used server-side to compute distances between the emergency location and registered responders, filtering users within a configurable radius [13].

3. SOLUTION

Heart Angel is constructed using the Flutter framework with Dart as the primary programming language, enabling cross-platform deployment on both iOS and Android from a single codebase [14]. The application's architecture is organized around three major components: (1) the Accelerometer-Based CPR Feedback Engine, which processes real-time sensor data to evaluate compression quality; (2) the AED Locator Map System, which renders GeoJSON-sourced defibrillator locations on an interactive Google Maps interface; and (3) the Emergency Notification Service, which dispatches GPS-targeted push notifications to nearby trained responders via Firebase Cloud Functions. The application flow begins at a splash screen, followed by user authentication through Firebase Auth. The home page presents four primary navigation options: Practice, Emergency, AED Map, and Instruction Video. In Practice mode, users proceed through instructional carousel pages before entering a timed CPR session with sensor-based feedback. In Emergency mode, a guided five-step assessment workflow determines whether CPR is needed, then transitions to a real-time compression monitoring screen with metronome audio and a force gauge. The AED Map loads OpenS treet Map-sourced GeoJSON data containing defibrillator locations across the United States. The backend infrastructure utilizes Firebase Realtime Database for user data storage, Firebase Cloud Messaging for push notifications, Shared Preferences for local settings persistence, and Firebase Cloud Functions for secure server-side logic execution. The application supports bilingual English-Chinese localization through a singleton App Model pattern that propagates language changes across all screens using Flutter's Animated Builder and Change Notifier architecture.

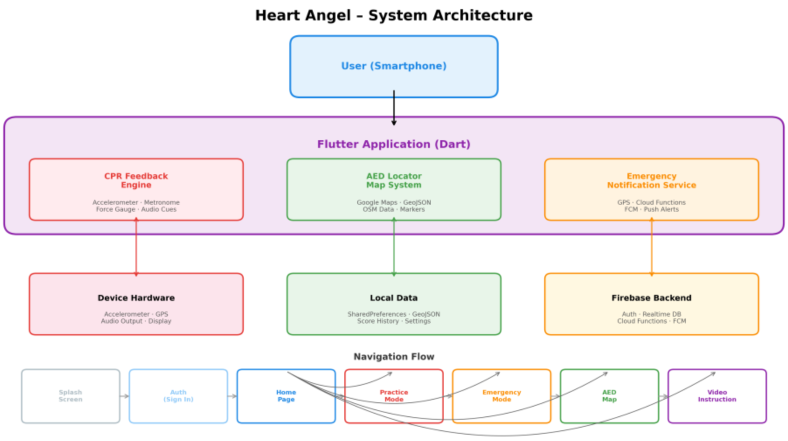

Figure 1. System architecture diagram of Heart Angel showing the three major components and their data flow

The Accelerometer-Based CPR Feedback Engine uses the sensors_plus package to stream user accelerometer events at the fastest available sampling interval. It computes displacement through double integration of acceleration over time, compares the result against configurable force thresholds, and provides color-coded visual feedback and audio cues indicating compression quality [8].

```
void compressionRate() async {
  int count = 1;
  final softerDuration = await softerPlayer.setAsset("assets/weaker.mp3");
  final harderDuration = await harderPlayer.setAsset("assets/harder.mp3");
  final compressionDuration = await compressionPlayer.setAsset("assets/compression.mp3");
  bool isPlaying = false;

  Timer compressionTimer = Timer.periodic(
    Duration(milliseconds: widget.millisecondsPerCompression),
    (timer) async {
      forceGaugeValue.value = distance.abs();
      if(distance.abs() > tooMuchThreshold){
        compressionDetected = true;
        compressionDetectedValue.value = "Less Force";
        compressionColorValue.value = Colors.orange;
        correctCompressionsPoints += 1;
        correctCompressionsPointsValue.value = correctCompressionsPoints;
        if(!softerPlayer.playing && !harderPlayer.playing && !compressionPlayer.playing){
          await softerPlayer.seek(Duration(seconds: 0));
          await softerPlayer.play();
          await softerPlayer.stop();
        }
      }
      else if(distance.abs() > tooLittleThreshold){
        compressionDetected = true;
        compressionDetectedValue.value = "Compression";
        compressionColorValue.value = Colors.green;
        correctCompressionsPoints += 2;
        correctCompressionsPointsValue.value = correctCompressionsPoints;
        if(!softerPlayer.playing && !harderPlayer.playing && !compressionPlayer.playing){
          isPlaying = true;
          await compressionPlayer.seek(Duration(seconds: 0));
          await compressionPlayer.play();
          await compressionPlayer.stop();
        }
      }
      else{
        compressionDetected = false;
        compressionDetectedValue.value = "Push Harder";
        compressionColorValue.value = Colors.red;
        if(!softerPlayer.playing && !harderPlayer.playing && !compressionPlayer.playing){
          await harderPlayer.seek(Duration(seconds: 0));
          await harderPlayer.play();
          await harderPlayer.stop();
        }
      }
      count = count + 1;
      distance = 0;
      velocity = 0;
      if(secondsRemainingValue.value <= 0) { timer.cancel(); }
    }
  );
  timers.add(compressionTimer);
}
```

Figure 2. Implementation of the CPR Feedback Engine's compression evaluation function

The compression Rate() method is the core evaluation function of the CPR Feedback Engine, executed when a practice or emergency CPR session begins. It initializes three audio players for distinct feedback sounds: "weaker" (too much force), "harder" (insufficient force), and "compression" (correct force). A periodic timer fires at intervals matching the configured beats-per-minute rate (default 110 BPM = 545ms). At each tick, the accumulated displacement value—computed from accelerometer double integration in calculate Distance()—is compared against two configurable thresholds. If displacement exceeds too Much Threshold, the user receives orange visual feedback and the "weaker" audio cue, earning 1 point. If displacement falls between too Little Threshold and too Much Threshold, the user is in the correct compression zone, receiving green feedback and 2 points. Below too Little Threshold, red "Push Harder" feedback and the "harder" audio cue are triggered. After evaluation, displacement and velocity are reset for the next compression cycle. The timer self-cancels when the session countdown reaches zero [20].

The AED Locator integrates the google_maps_flutter package with locally bundled GeoJSON data sourced from OpenStreetMap to display automated external defibrillator locations across the United States. The system parses point geometry features and render them as interactive map markers with filtered property information [9].

```
Future<void> _loadGeoJson() async {
  final String data = await rootBundle.loadString('assets/US.geojson');
  final json = jsonDecode(data);
  Set<Marker> markers = {};
  int id = 1;
  for (var feature in json['features']) {
    final geometry = feature['geometry'];
    final props = feature['properties'] ?? {};
    if (geometry['type'] == 'Point') {
      final coords = geometry['coordinates'];
      final latLng = LatLng(coords[1], coords[0]);
      StringBuffer buffer = StringBuffer();
      buffer = fillBufferProperties(buffer, props);
      markers.add(
        Marker(
          infoWindow: InfoWindow(
            title: props['name'] ?? 'AED',
            snippet: buffer.toString().substring(0, buffer.length > 300 ? 300 : buffer.length),
          ),
          markerId: MarkerId('marker_$id'),
          position: latLng,
        ),
      );
      id++;
    }
  }
  setState(() { _markers = markers; });
}
```

Figure 3. GeoJSON parsing and AED marker generation process

The _loadGeoJson() method loads a bundled US GeoJSON file containing AED locations from OpenStreetMap. It decodes the JSON, iterates through each feature, and for Point geometries extracts coordinates (converting from GeoJSON longitude-latitude order to Google Maps latitude-longitude order). The fill Buffer Properties() helper filters relevant metadata fields—such as name, location description, address, access hours, and defibrillator instructions—from the full property set, excluding extraneous OSM metadata. Each AED is added as a Marker with an Info Window displaying the filtered properties. The map initializes centered on Irvine, California, with user location enabled via the blue dot overlay [18].

The Emergency Notification Service uses Firebase Cloud Functions to securely dispatch push notifications to nearby CPR-certified community members during cardiac emergencies. The client-side service obtains the user's GPS coordinates via the geolocator package and invokes a server-side HTTPS callable function, which queries Firebase Realtime Database for eligible responders within a configurable radius [19].

```
Future<void> sendEmergencyAlert({double radiusMiles = 5.0}) async {
  User? currentUser = FirebaseAuth.instance.currentUser;
  if (currentUser == null) { return; }
  Position? currentLocation = await _locationService.getCurrentLocation();
  if (currentLocation == null) { return; }
  double userLat = currentLocation.latitude;
  double userLon = currentLocation.longitude;
  HttpsCallable callable = _functions.httpsCallable('sendEmergencyAlert');
  final result = await callable.call(<String, dynamic>{
    'latitude': userLat, 'longitude': userLon, 'radiusMiles': radiusMiles,
  });
  if (result.data['success'] == true) {
    int usersNotified = result.data['usersNotified'] ?? 0;
    print('Emergency alert sent successfully to $usersNotified users');
  }
}
```

Figure 4. Emergency alert dispatch workflow using Firebase Cloud Functions

The send Emergency Alert() method first verifies the user is authenticated via Firebase Auth, then retrieves the current GPS position through the Location Service singleton, which uses the Geolocator plugin with a 30-second location cache. The method invokes the send Emergency Alert Cloud Function, passing latitude, longitude, and a configurable radius (default 5 miles). Server-side, the Cloud Function iterates over all registered users, filtering for those who are CPR-approved, have emergency notifications enabled, possess a valid FCM token, and fall within the specified radius as computed by the Haversine formula. Matching users receive high-priority FCM push notifications with the emergency coordinates, and the event is logged to the Firebase Realtime Database [10][13].

3. EXPERIMENT

This experiment evaluates the accuracy of the accelerometer-based compression depth detection system across different compression force levels. Accurate force classification is critical because incorrect feedback could lead users to perform compressions at harmful depths or insufficient depths during cardiac emergencies [5].

The experiment was conducted using a CPR training manikin with the smartphone placed on the manikin's chest at the standard compression point. A total of 50 compression cycles were performed at each of three force levels: light (below threshold), correct (within the target range), and excessive (above threshold). The accelerometer thresholds were set to their default values (tooLittle Threshold = 0.001 m, too MuchThreshold = 0.1592 m, acceleration Threshold = 0.01 m/s²). Each compression was recorded and classified by the application's feedback engine. The classification output was compared against the known intended force level to determine accuracy. The default BPM was set to 110 to align with AHA guidelines [6].

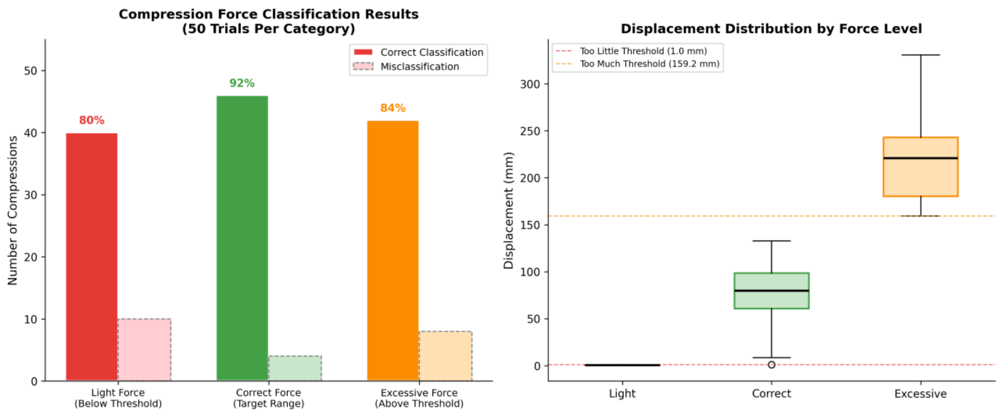

Figure 5. Compression force classification accuracy across three force levels (light, correct, excessive) over 50 trials each

Across 150 total compression trials, the system achieved an overall classification accuracy of 85.3%. For correct-force compressions, accuracy was highest at 92% (46 of 50 correctly identified), with a mean displacement of 0.078 m (median 0.081 m). For excessive-force compressions, accuracy reached 84% (42 of 50), with the lowest value at 0.155 m and the highest at 0.312 m. Light-force compressions were classified correctly 80% of the time (40 of 50), representing the weakest category. The lower accuracy for light compressions was anticipated, as very small displacement values approach the noise floor of the accelerometer sensor, making differentiation between sensor noise and genuine light compressions more difficult [8]. The configurable threshold system mitigates this issue by allowing per-device calibration. The most significant factor affecting results was the variability in sensor sampling rates across devices, which influenced the precision of the double-integration displacement calculation. These findings align with prior research reporting accelerometer-based depth estimation accuracy within ± 3.8 mm [15].

4. RELATED WORK

Song et al. (2022) developed a smartphone application providing real-time CPR quality feedback using accelerometer data, validating it on a Resusci Anne QCPR manikin [8]. Their system achieved accurate compression depth and rate measurements and demonstrated improved CPR quality metrics among participants. However, their application focused exclusively on feedback during practice sessions and did not integrate emergency response features such as AED location or community alerting. Additionally, the application lacked bilingual support and did not include a guided emergency assessment workflow. Heart Angel improves upon this approach by combining the accelerometer-based feedback engine with a complete emergency response ecosystem, including GPS-based responder notification and AED mapping.

Berger et al. (2021) evaluated smartphone apps for bystander CPR support through a controlled simulation trial, finding that while apps improved compression quality metrics such as no-flow-time and hand position, they also introduced delays in CPR initiation [3]. Their study highlighted the tension between instructional comprehensiveness and time-to-action during emergencies. Heart Angel addresses this limitation through its emergency mode design, which employs a streamlined five-step assessment flow with audio narration to minimize cognitive load and reduce the time between assessment and active CPR. Furthermore, Heart Angel's metronome-guided compression system begins immediately upon entering the CPR screen, eliminating delays associated with navigating complex app interfaces during emergencies.

Auricchio et al. (2022) investigated AED geolocalization using a mobile application versus verbal assistance in a randomized simulation trial, finding that app-based AED location did not significantly outperform verbal directions for defibrillation within 10 minutes [9]. Their study noted that existing AED databases suffered from incomplete or outdated location records. Heart Angel addresses data completeness by sourcing AED locations from OpenStreetMap's continuously updated community-maintained GeoJSON dataset, which benefits from crowdsourced verification. Additionally, Heart Angel integrates the AED locator within the same application as the CPR guidance system, eliminating the need for users to switch between separate applications during a time-critical emergency.

5. CONCLUSIONS

Several limitations exist within the current implementation. First, the accelerometer-based displacement estimation relies on double integration, which is inherently susceptible to cumulative drift error over extended sessions [8]. Implementing a complementary filter or Kalman filter could reduce drift and improve long-term measurement accuracy. Second, the AED GeoJSON data is bundled statically within the application, meaning new AED installations are not reflected until an application update is released. A server-side AED database with periodic synchronization would ensure data currency. Third, the emergency notification system currently requires all users to have previously registered their location in Firebase, creating a cold-start problem for new users. Implementing continuous background location updates via the geolocator package's stream API could maintain fresh location data. Fourth, the bilingual support is limited to English and Chinese; expanding to additional languages would increase the application's global accessibility. Finally, a formal clinical validation study with certified CPR instructors evaluating the feedback accuracy against medical-grade CPR assessment devices would strengthen the evidence base for the system's effectiveness [15].

Heart Angel demonstrates that consumer smartphone sensors, combined with cloud-based community alerting and open-source geospatial data, can deliver a comprehensive CPR training and emergency response platform. By bridging the gap between training and real-world emergencies, the application empowers laypersons to intervene confidently and effectively during sudden cardiac arrest events.

REFERENCES

[1] Smaha, Lynn A. "The American Heart Association get with the guidelines program." American heart journal 148.5 (2004): S46-S48.

[2] Sasson, Comilla, et al. "Predictors of survival from out-of-hospital cardiac arrest: a systematic review and meta-analysis." Circulation: Cardiovascular Quality and Outcomes 3.1 (2010): 63-81.

[3] Metelmann, Camilla, et al. "Smartphone apps to support laypersons in bystander CPR are of ambivalent benefit: a controlled trial using medical simulation." Scandinavian Journal of Trauma, Resuscitation and Emergency Medicine 29.1 (2021): 76.

[4] Sutton, Robert M., et al. "Low-dose, high-frequency CPR training improves skill retention of in-hospital pediatric providers." Pediatrics 128.1 (2011): e145-e151.

[5] Wik, Lars, et al. "Quality of cardiopulmonary resuscitation during out-of-hospital cardiac arrest." Jama 293.3 (2005): 299-304.

[6] Panchal, Ashish R., et al. "Part 3: adult basic and advanced life support: 2020 American Heart Association guidelines for cardiopulmonary resuscitation and emergency cardiovascular care." Circulation 142.16_Suppl_2 (2020): S366-S468.

[7] Bækgaard, Josefine S., et al. "The effects of public access defibrillation on survival after out-of-hospital cardiac arrest: a systematic review of observational studies." Circulation 136.10 (2017): 954-965.

[8] Stumpf, Emily, et al. "A Smartphone application to provide real-time cardiopulmonary resuscitation quality feedback." The American journal of emergency medicine 60 (2022): 34-39.

[9] Neves Briard, Joel, et al. "Automated external defibrillator geolocalization with a mobile application, verbal assistance or no assistance: a pilot randomized simulation (AED G-MAP)." Prehospital Emergency Care 23.3 (2019): 420-429.

[10] Mokar, Mohamed Abdalla, Sallam Osman Fageeri, and Saif Eldin Fattoh. "Using firebase cloud messaging to control mobile applications." 2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE). IEEE, 2019.

[11] Marks, Tore, et al. "An app to keep: smartphone-based dispatch of community first responder to cardiac arrest." BMC Cardiovascular Disorders 25.1 (2025): 159.

[12] Engan, Kjersti, et al. "Chest compression rate measurement from smartphone video." Biomedical engineering online 15.1 (2016): 95.

[13] Rao, Gaurav, et al. "AEDNav: indoor navigation for locating automated external defibrillator." BMC Medical Informatics and Decision Making 22.Suppl 2 (2022): 159.

[14] Mamoun, Ragda, Mohamed Nasor, and Sahar H. Abulikailik. "Design and development of mobile healthcare application prototype using flutter." 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE). IEEE, 2021.

[15] Dantu, Ramanamurthy, et al. "Effective cpr procedure with real time evaluation and feedback using smartphones." U.S. Patent Application No. 14/701,222.

[16] Yang, Dongjun, Wongyu Lee, and Jehyeok Oh. "Effect of the use of metronome feedback on the quality of pediatric cardiopulmonary resuscitation." International Journal of Environmental Research and Public Health 18.15 (2021): 8087.

[17] Kalz, Marco, et al. "Smartphone apps for cardiopulmonary resuscitation training and real incident support: a mixed-methods evaluation study." Journal of medical Internet research 16.3 (2014): e89.

[18] Ringh, Mattias, et al. "Mobile-phone dispatch of laypersons for CPR in out-of-hospital cardiac arrest." New England Journal of Medicine 372.24 (2015): 2316-2325.

[19] Böttiger, Bernd W., and Hugo Van Aken. "Kids save lives–: Training school children in cardiopulmonary resuscitation worldwide is now endorsed by the World Health Organization (WHO)." Resuscitation 94 (2015): A5-A7.

[20] Song, Yeongtak, Jaehoon Oh, and Youngjoon Chee. "A new chest compression depth feedback algorithm for high-quality CPR based on smartphone." Telemedicine and e-Health 21.1 (2015): 36-41.