

Eldergo+ : Companion Assistance Booking System for Senior Citizens

Dr. Krishna Gudi

Department of Computer
Science Engineering,
K.S. Institute of Technology

Garv B Jain

Department of Computer
Science Engineering,
K.S. Institute of Technology

HL Darshan

Department of Computer
Science Engineering,
K.S. Institute of Technology

Harshavardhan B K

Department of Computer
Science Engineering,
K.S. Institute of Technology

Harsh Mishra

Department of Computer
Science Engineering,
K.S. Institute of Technology

Abstract - Older adults often need help with everyday activities such as reaching a hospital, shopping, attending appointments and taking part in community life while many live far from busy family members. Conventional ride-booking architectures optimize transportation matching. Do not inherently address companion verification, assistance-specific booking workflows or the privacy requirements of elderly users. This paper presents ElderGo+, a proposed web-based platform that coordinates companion-assisted services through a booking, verification, assignment, tracking, messaging, emergency-alert, notification and feedback workflow. The system separates companion registration from approval distributes eligible requests to verified companions and employs a server-authoritative atomic assignment mechanism to prevent several companions from claiming the same booking. Real-time location updates display the assigned companion's position and the remaining straight-line distance to the pickup location to device permissions and connectivity and a notification subsystem communicates booking events through configurable email and SMS integrations. The methodology combines a requirements survey of adults and their families with a provenance-oriented audit of application records schema validation, duplicate detection, exclusion accounting and verification of assignment and authorization invariants. However no measured dataset, completed performance benchmark or verified deployment result is available. Empirical dataset scale, latency, delivery reliability and comparative performance therefore remain unreported than estimated or fabricated. The proposed evaluation framework defines transactional, functional, security and cross-device tests against assignment and synchronization baselines. ElderGo+ thus establishes a system architecture for elderly assistance coordination while its operational reliability and suitability for large-scale deployment remain subject to systematic validation.

Keywords— assistance, companion booking system senior citizen care, real-time tracking, emergency alert system, atomic assignment, geolocation, access control, event-driven notifications, software engineering .

I. INTRODUCTION

Population ageing presents technological and organizational challenges for delivering accessible person-centred support. The World Health Organization identifies age- communities, integrated services and access to appropriate long-term care as important areas of action for healthy ageing . Many older adults find tasks such as going to a hospital, shopping or obtaining essentials increasingly difficult and they often live far from family members who have limited time to help. Digital platforms can help coordinate assistance by connecting individuals who require support with service providers. Nevertheless such coordination requires more than a booking

form: the platform must establish provider eligibility manage competing requests communicate state changes and protect location information. Prior work on companion and volunteerengagement platforms reflects growing interest in this problem while related systems address healthcare management IoTbased monitoring web-based service booking and emergency alerts . Each tends to cover an aspect of the need. Available options may not respond promptly may be difficult to use. May not make users feel safe while a helper is present. Ride-booking systems provide an architectural reference because they coordinate requests, provider availability, assignment and geographic tracking. Elderly assistance differs from transportation in its emphasis, on trust, accessibility,

companionship and potentially sensitive personal circumstances. A companion may assist with an appointment accompany an adult during an outing or provide non-clinical personal support. Consequently provider verification and controlled disclosure of booking information are central system requirements.

I see that ElderGo+ is built as a role-based assistance booking platform that brings together clients, companions and administrators. Clients can log in. Register, fill in details about the help they need look for companions and confirm bookings. Companions can. Share their experience background.

Administrators check registrations before allowing companions to receive requests. When a client submits a booking eligible companions can answer the request. ElderGo+ then assigns the booking to one companion informs all parties. Provides location tracking trip-status updates, messaging, emergency alerts and post-service feedback and ratings.

I notice that the main engineering challenge is keeping data consistent when many users and devices act at the time. A client-side implementation might look fine when one user tests it but it can break if two companions try to accept the same request at once. Likewise a location marker that changes inside one browser cannot provide real cross-device tracking. Therefore ElderGo+ needs backend-enforced assignment rules, state authenticated real-time communication and clear failure handling.

I note that this paper explains the architecture and evaluation method for ElderGo+. It separates the features that are proposed from those that have been proven. It does not claim that a user interface alone means the system is ready for production. The principal contributions are:

1. Verification-aware service coordination: a workflow that separates companion registration, administrative approval and booking eligibility.
2. Concurrency-safe assignment: a model that uses invariants to stop duplicate companion allocation.
3. Cross-device tracking: a model for updating location and estimating distance that includes authorization, consent, freshness and connectivity checks.
4. Integrated event notifications: a design that connects booking state changes to email, SMS and alert delivery.
5. Reproducible evaluation methodology: a testing framework that covers correctness, concurrent assignment, security, location synchronization and notification delivery and it requires clear reporting for any unmeasured results.

I will describe the rest of the paper. It reviews word formalizes the booking problem explains the data audit and implementation method describes the architecture and

evaluation framework and discusses deployment needs and limits.

Still the big goal of ageing is not the same as proving a health result. A booking platform can help people get assistance. It does not prove it improves health, independence or reduces loneliness – that needs separate study. ElderGo+ focuses on the coordination side not on diagnosing conditions or replacing care. ElderGo+ covers assistance requests, companion verification, appointment coordination and booking visibility. ElderGo+ addresses the coordination layer rather than attempting to diagnose medical conditions or replace clinical care. Its scope includes assistance requests, companion verification, appointment coordination, and booking visibility.

B. On-demand service allocation

I observe that on-demand platforms normally involve creating a request choosing a provider assigning the task and finishing the service. Web-based booking designs matching algorithms and studies of user experience on service platforms cover parts of this process.

A simple allocation picks one provider for each request. In a distributed setting several providers might try to take the request before their screens update. ElderGo+ uses a shared request model where many eligible companions can see a booking. A backend decides who gets assigned. Unlike solutions that depend on interface updates ElderGo+ makes database consistency and server-side authorization the key correctness rules.

ElderGo+ uses a shared request model in which several eligible companions can view a booking, while a backend operation determines the successful assignment. Unlike an implementation that relies exclusively on interface updates, the proposed design treats database consistency and server-side authorization as correctness requirements.

C. Real-time communication and geolocation

I understand that the WebSocket protocol lets clients and servers talk to each other in time so it can carry live updates . Browser geolocation lets apps read a device's location. Only with user permission and platform limits . Real-time location tracking has been used in safety apps . These tools can share location and sync booking status. They do not, by themselves set proper authorization or keep background tracking running all the time. It formalizes the booking problem, specifies the data audit and implementation methodology, describes the architecture and evaluation framework, and discusses deployment requirements and limitations.

II. RELATED WORK

A. Digital support for healthy ageing

I read that healthy-ageing initiatives stress keeping functional ability and building environments that let older adults join daily life . Digital coordination systems can help by making

assistance easier to request and organize. Users look at well-being and software usability, backed by a sufficiently recent authorized location update.

D. Event notifications and system reliability

Email, SMS and alert mechanisms can inform clients and companions of booking events of whether an application is currently visible. Reliable notification processing requires handling of delivery failures, retries, duplicate events and provider responses.

The proposed system distinguishes the persistence of a booking transition from the delivery of its notification. A notification provider failure must not reverse a committed companion assignment. This separation also allows notification delivery to be evaluated independently of booking correctness.

E. Research gap and positioning

The contribution of ElderGo+ is the integration of companion verification, assistance-specific booking workflows, atomic assignment, cross-device tracking and event notifications in a system specification. This paper does not claim that each individual mechanism is novel. Instead it defines their coordination. Establishes testable invariants appropriate to elderly assistance services. IoT-based monitoring is complementary to than a replacement for the companion-coordination layer addressed here.

A comparative claim of superiority over ride-booking systems would require access to comparable implementations and controlled experiments. No such comparison is asserted here.

III. DATASET CONSTRUCTION AND DATA/LEAKAGE AUDIT

A. Data provenance and inventory

ElderGo+ is mainly a software system, not a predictive model that is trained on a labelled research dataset. Its operational data may contain user profiles, companion registrations, verification decisions, bookings, assignment events, location updates, feedback and ratings and notification records. Requirements were gathered via Google Forms surveys of adults and from conversations with older adults and their families. The instrument, the number of responses and the interview notes have not been. Therefore they are not summarized quantitatively here.

A reproducible audit must begin with an inventory of the implementation and available records. The current project description does not contain database exports, event logs, schema files or a measured record count. Therefore we cannot yet determine the data volume and quality of the deployed application. The audit should record the source table or file the extraction timestamp, the record count, the schema version and the transformation history for each dataset.

B. Data cleaning and validation

The following checks are proposed for the initial audit:

1. Schema validation: confirm that each required field is present that data types are correct that identifiers exist that timestamps are valid. That status values are allowed.
2. Experience validation: reject experience values that're negative or malformed and verify that a value of zero is still considered valid.
3. Identity consistency: detect identifiers, invalid references and duplicate logical profiles.
4. Booking integrity: identify bookings that have no related data (orphaned) detect assignment references that're invalid and find booking states that are incompatible.
5. Duplicate-event detection: identify events that are repeated such as acceptance, arrival, completion or notification events.
6. Consistency: check that assignment timestamps do not come before booking creation and that location timestamps are in the correct order.
7. Geospatial validation: reject coordinates that fall outside the latitude and longitude ranges and flag any implausible jumps for review.
8. Notification integrity: distinguish between queued, submitted, delivered and failed states when the provider exposes these distinctions.

Cleaning must not silently erase anomalous records. Each exclusion should have a documented reason, source identifier, and count.

C. Exclusion ledger and leakage prevention

In a transactional system, leakage is more appropriately understood as information crossing a trust boundary or test boundary, rather than only as target leakage in machine learning. The audit must examine:

- Whether unapproved companion profiles are exposed to booking-distribution queries.
- Whether companions can change their verification status.
- Whether clients can read clients' bookings or location records.
- Whether stale or post-completion locations remain accessible.
- Whether repeated acceptance events create assignments.
- Whether the evaluation dataset contains events generated by the test run used to validate the claimed result.
- Whether synthetic test accounts are incorrectly counted as users.

- Whether notification logs are mistaken for evidence of delivery.

Where historical data is available, training and evaluation should be separated temporally if predictive components are introduced in future work. The present system specification does not establish a predictive task or justify a machine-learning leakage audit based on model labels.

TABLE I. DATA INVENTORY AND AUDIT STATUS.

Data source	Audit requirement and current evidence
Client profiles	Schema, identity, access control; records not supplied
Companion profiles	Experience, status, duplicates; records not supplied
Verification records	Decision provenance, authorization; logs not supplied
Booking records	State and assignment consistency; logs not supplied
Location updates	Coordinates, timestamps, permissions; trace not supplied
Notification records	Event identity, delivery status; provider logs not supplied
Feedback and ratings	Provenance, duplicate submissions; records not supplied
Requirements survey (Google Forms)	Instrument and response count; not supplied
Interviews with older adults and families	Interview notes and consent record; not supplied
Test fixtures	Synthetic-data identification; test inventory not supplied
Exclusion ledger	Reason-coded counts; not yet constructed

No dataset size or exclusion count is stated because the evidence for these numbers is not available.

TABLE II. SYSTEM CONFIGURATION AND REPRODUCIBILITY PARAMETERS.

Parameter	Required record
Frontend framework	Web and mobile application; actual framework and version to be recorded
Backend framework	Flask or Django per the project description; final choice and version to be recorded
Database	Engine and schema revision (user, companion, and booking data)
Authentication	Provider and authorization model
Assignment mechanism	Transaction or atomic-operation specification
Real-time transport	Actual configured implementation
Map provider	Provider and API configuration version
Notification services	Configured email, SMS, and alert providers
Build identifier	Source revision and build metadata
Test environment	Hardware, operating system, browser, network
Random seed	Required only for randomized experiments
ML hyperparameters	Not applicable unless an ML model is introduced

The final paper should replace these configuration requirements, with the real values taken from the project.

IV. SYSTEM PIPELINE AND ARCHITECTURE

A. Layered architecture

The proposed architecture splits user interaction, application logic, data storage and external services . The user interface layer gives the client booking interface, companion request interface, admin verification dashboard, tracking screen, messaging and feedback forms all delivered as a web or mobile frontend. The interface shows states that the backend has confirmed and gives error handling.

B. Booking distribution

For each booking b_i the app calculates its eligible set E_i using formula . The system then shows the booking to companions via stored backend state and authorized real-time subscriptions. A companion's request view alone is not final. The backend has to check eligibility when an acceptance comes in because approval, availability or booking state could have shifted after the request was shown.

C. Atomic assignment

A simple conditional assignment operation looks like:

$$\text{Assign}(b_i, p_j) = \begin{cases} \text{commit,} & \text{if eligible and unassigned} \\ \text{reject,} & \text{otherwise.} \end{cases}$$

The condition check and assignment write must be an atomic operation or transaction. A uniqueness constraint on the booking assignment gives extra integrity protection. If two companions try to accept at the time only one operation can commit the assignment. The companion that does not win receives a conflict response. The interface updates to show the authoritative booking state.

D. Location update pipeline

The companion's device gets coordinates after the necessary location permission is granted. The update is sent over an authenticated channel. Checked by the backend. A location record is accepted when its booking, actor, coordinates, timestamp and authorization are all valid. The client tracking interface receives updates via the real-time mechanism and recalculates the shown distance. Define location freshness as: where the time stamp of the newest accepted location update. When $F(t)$ is larger than a set threshold τ the interface must show that the location might be stale of showing it as current.

E. Notification pipeline

A successful booking transition creates a notification event that has an event identifier. The notification worker or backend integration handles the event. Logs the provider response. The emergency-alert feature mentioned in the project description is treated as a high-priority event class. Its recipients, escalation path and delivery guarantees have to be described and checked separately. It must not be seen as a replacement for emergency services. The logical order is:

1. Validate the requested state transition.
2. Commit the booking update and event record.
3. Send the notification asynchronously when possible.
4. Log the delivery attempt and provider status.
5. Retry failures that are eligible without repeating the booking transition.

If the architecture has an outbox it can help lower the chance of committing a booking change without safely saving its notification event.

F. Serving gates

Before an operation is accepted the backend should check the following gates:

- Authentication and role authorization;
- Companion approval and current eligibility;
- Booking state and assignment uniqueness;
- Otp validity for the transitions that require it.
- Location permission and booking authorization.
- Notification configuration and delivery error handling.

If an authorization or assignment gate fails the operation must be rejected. If an external notification service fails it must not damage a booking. If location delivery fails the interface should show an stale state.

V. EVALUATION METHODOLOGY

A. Evaluation objectives

Evaluation should show whether the platform meets its concurrency, synchronization, security and reliability needs. Because the current project input has no benchmark results this part lists experiments to run of giving unseen outcomes. The evaluation environment should log the application revision, database version, test-device setup, network conditions and service-provider settings. Every test must use booking identifiers and separate synthetic test activity from real operational records.

B. Baselines

B1, a client-side assignment baseline: acceptance is mainly handled through interface state and client-side checks. This is a negative-control setup that shows why client-side checks alone cannot guarantee assignment.

B1 must only be used in a separate test environment. It must not be run with users or sensitive data. Comparisons should look at correctness, conflict handling, response time and consistency. These baselines are engineering comparators, not claims that they are better than a service.

C. Methodology architecture

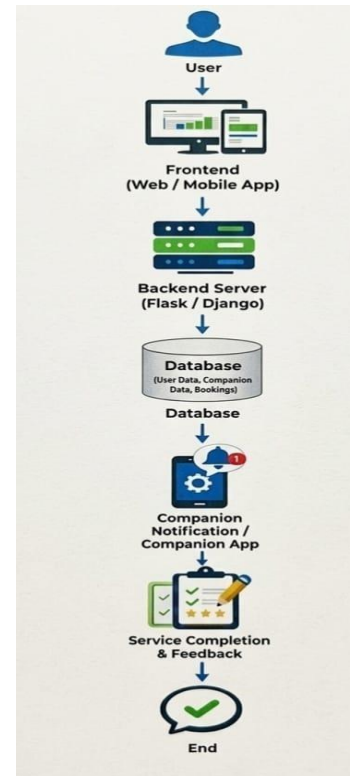


Fig.1. ElderGo+ System Architecture

D. Spatial and temporal validation

Location tests must use authorized devices or controlled test fixtures that have known coordinates. Location tests should measure how long it takes for a location update to appear, how the system detects locations, the error in distance calculation compared to the chosen geographic reference and how the system recovers after network interruptions. Location tests should also cover different devices and networks test both foreground and background behavior when the platform allows it and test scenarios where permissions are denied.

Temporal testing must measure how events travel from booking creation to when a companion becomes visible from acceptance to when a client receives confirmation from location publication to when a map updates and from state transition to when a notification provider responds.

The notification success rate R_{notif} is the number of confirmed deliveries N_{conf} divided by the number of eligible notification events.

The notification metric must be calculated only when the provider sends delivery receipts or when another reliable confirmation is available. A successful API submission does not automatically mean a message was delivered. Notification latency must be reported using the median, the 95th percentile and the 99th percentile when the sample size is large enough to give estimates. Location distance error must be measured

against a documented reference and location freshness must be reported separately from accuracy.

For repeated benchmark trials report how many samples were used and the uncertainty intervals. Bootstrap confidence intervals can be used for latency and error metrics. Paired comparisons or suitable nonparametric tests can also be applied to repeated runs that're comparable. Statistical significance should not be claimed unless actual observations exist and a justified testing procedure has been used.

E. Application Architecture

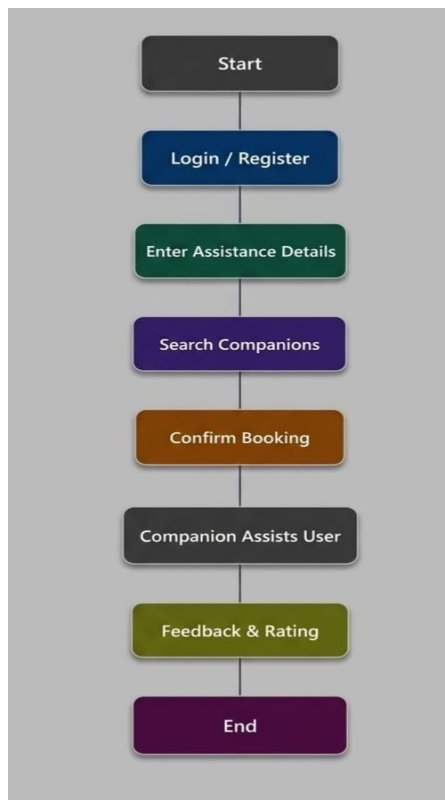


Fig.2. User Assistance Booking Workflow

VI. RESULTS

A. Functional correctness and assignment integrity

The intended assignment mechanism is designed to enforce the single-assignment invariant shown in equation . However the project description does not show any concurrency tests or database traces that have been run. Because of this it is not possible to report a duplicate-assignment rate or to confirm that the implementation already satisfies this invariant. The primary experiment should submit acceptance requests that target the same booking and then inspect the database records that have been committed. The expected acceptance criterion is that no than one active assignment exists for any booking during all tested trials.

Qualitative statements in the project description for example that matching clients with available companions reduces waiting time and that live tracking increases user confidence are

design intentions. These statements are not supported by measurements in this paper. They require a user study and the experiments that were described earlier.

TABLE III. PRIMARY SYSTEM EVALUATION METRICS.

Metric (measurement method)	Result
Companion registration correctness (valid and invalid registration tests)	Not measured
Zero-year experience preservation (registration and database verification)	Not measured
Unauthorized approval attempts (negative authorization tests)	Not measured
Duplicate active assignments (concurrent acceptance stress test)	Not measured
Valid assignment success rate (valid assignments / eligible bookings)	Not measured
Booking-state transition correctness (workflow and transition tests)	Not measured
OTP verification correctness (valid and invalid OTP tests)	Not measured
Cross-device booking synchronization (independentdevice integration test)	Not measured

The expected outcome of the atomic assignment design is that concurrent acceptance attempts cannot create than one active assignment. This expectation remains an acceptance criterion until it is verified against the implementation.

B. Location and notification performance

The location subsystem should be evaluated by using updates that come from the device of the assigned companion. The analysis must separate distance-calculation accuracy, location freshness, transport latency and map-rendering delay. A decrease in distance is expected when the companion's measured position moves nearer to the pickup point. The distance does not always decrease steadily when real GPS noise or movement is present.

Notification performance should be assessed separately for email and for SMS. Provider submission, delivery confirmation, failure handling and duplicate-event prevention must each be measured on their own.

C. Latency, failure cases, and generalization

Booking propagation, assignment confirmation, location update and notification response times must be reported separately as shown in Table IV. I also want the system to be tested across device types network conditions, companion approval states and booking outcomes. Important cases are approved companions rejected companions simultaneous acceptance attempts, location permission denial, stale coordinates, provider timeouts and cancellation during an active workflow.

No subgroup measurements or negative experimental results have been supplied. Because of that I cannot say that every device type, network condition or geographic area has been tested. Browser background restrictions can stop location publishing and email or SMS delivery may depend on external provider availability. These problems must be listed as real

failure modes if they appear during testing not assumed fixed by the architecture.

TABLE IV. DOWNSTREAM AND PIPELINE EVALUATION METRICS.

Metric (required evidence)	Result
Booking propagation latency (timestamped creation and receipt events)	Not measured
Assignment confirmation latency (backend commit and client receipt timestamps)	Not measured
Location update latency (device publication and client receipt timestamps)	Not measured
Location freshness (distribution of update ages)	Not measured
Geographic distance error (reference coordinates and calculated distance)	Not measured
Email delivery success (provider delivery receipts)	Not measured
SMS delivery success (provider delivery receipts)	Not measured
Recovery after disconnection (controlled interruption and reconnection tests)	Not measured
Notification duplicate rate (event identifiers and provider logs)	Not measured

VII. LIMITATIONS AND THREATS TO VALIDITY

A. Lack of measured operational data

The main limitation is that we do not have a dataset execution logs, benchmark records or a deployment report. Because of that we cannot make any conclusion, about how often assignments succeed how long location updates take, how well notifications are delivered, how much traffic the system can handle or how available the service is. Even if a working prototype exists it still needs to be tested in controlled and realistic conditions before we can claim operational performance.

B. Geographic and temporal transferability

Location behaviour depends on device accuracy, environmental conditions, network availability and the chosen mapping service. Results from an area or a short test do not show how the system works everywhere. Future tests should cover places real movement patterns, different devices and different connectivity.

C. Privacy and security

Location information can reveal details about a person's movements. Access must therefore be restricted to authorised users and relevant booking periods. Location retention should be. Access policies should be evaluated independently of the interface. Administrative verification also presents risks if the platform collects documents or personal information. Verification procedures, retention policies and access to those records require design and review.

D. Notification and infrastructure dependencies

Email and SMS delivery depend on provider configuration, network availability, recipient information, rate limits and regional requirements. The application cannot guarantee delivery solely by submitting a request to a provider. Similarly real-time communication may be interrupted by device suspension, network changes or server-side connection limits.

The system must expose unavailable states instead of masking interruptions.

E. User safety and service scope

ElderGo+ is a coordination platform. Should not be represented as a substitute for emergency services, licensed healthcare or clinical monitoring unless separately developed and appropriately validated for those functions. Companion verification and administrator approval also do not by themselves prove competence, reliability or suitability for every assistance task. A real deployment requires a defined verification policy, complaint handling, cancellation procedures and escalation processes.

F. Adoption and accessibility dependencies

Practical use depends on the availability of companions on users having internet access and on users knowing how to use the application. These dependencies were identified in the project description. Must be addressed, for example through accessible interface design and onboarding support if a large number of older adults are to use the platform. The project description also suggests that the platform can create earning opportunities for companions; this socio-economic effect has not been measured.

G. Baseline and experimental limitations

The proposed baselinesre controlled engineering comparators. They do not establish superiority over systems or prove improvements in the well-being of older adults. Future user studies should include informed consent, privacy protections, accessibility evaluation and institutional review where required. Any usability or health-related conclusions must be based on a study design rather than inferred from software functionality alone.

VIII. CONCLUSION

This paper presented ElderGo+, a companion assistance booking system designed to coordinate support for citizens through companion registration, administrative verification, request distribution, exclusive assignment, real-time location tracking, event notifications and feedback. The proposed system model formalises companion eligibility, transitions, atomic assignment, location-distance estimation and the separation of transactional state changes from external notification delivery. It also specifies an architecture and a reproducible evaluation framework covering functional correctness, concurrency, authorisation, location freshness, notification reliability and deployment verification.

The principal technical requirement is that important decisions remain authoritative at the backend. In particular simultaneous acceptance requests must not produce assignments unapproved companions must not receive booking privileges and location information must be disclosed only within an authorised tracking workflow.

The available project description does not establish a dataset scale completed benchmark results or verified production deployment. Accordingly no quantitative performance advantage or broad operational reliability claim is made. The proposed evaluation plan provides a foundation for establishing those properties through testing. Possible extensions, such as artificial-intelligence-based companion matching and IoT-based monitoring would require their design and validation.

ElderGo+ can serve as a practical foundation for digitally coordinated elderly assistance, provided that its implementation is validated for transactional correctness, security, accessibility, and real-world operational constraints before deployment at scale.

IX. REFERENCES

- [1] World Health Organization, Decade of Healthy Ageing: Baseline Report. Geneva, Switzerland: World Health Organization, 2021. [Online]. Available: <https://www.who.int/publications/i/item/9789240017900>
- [2] K. Dahake, P. Gowardhan, N. Paul, P. Kore and R. Gahlod, " companion enhancing quality life of older adults " International Journal of Trend in Scientific Research and Development 2024.
- [3] V. Somani, T. Nandi, P. Gupta and K. K. Sethi "Elderly care: Elderly well-being through volunteer engagement " in Proc. IEEE 4th Int. Conf. ICT in Business Industry and Government (ICTBIG) Indore, India, 2024.
- [4] A. Sharma, "Healthcare management systems, for services " International Journal of Healthcare Management 2021.
- [5] In 2020 M. K. M. Nasution, S. A. M. Noah and U. Harahap presented a study titled "Pharmacy management systems in hospitals." The study "Pharmacy management systems in hospitals" was published in the journal Systematic Reviews in Pharmacy.
- [6] In 2022 S. R. Singh introduced an idea called "Smart healthcare monitoring system using IoT." The article "Smart healthcare monitoring system using IoT" appeared in the International Journal of Engineering Research.
- [7] In 2022 T. Wang, Y. Zhang and L. Chen shared their research on "IoTbased monitoring systems for care." The paper "IoT-based monitoring systems for care" was featured in Sensors Journal.
- [8] In 2021 P. Gupta and A. Verma wrote about "Web-based service booking platform design." The work " service booking platform design" was published in the International Journal of Computer Applications.
- [9] In 2021 S. Reddy and P. Kumar delivered a presentation titled "Emergency alert systems for safety" at the IEEE International Conference on Smart Systems. The paper "Emergency alert systems for safety" was included in the conference proceedings.
- [10] In 2023 A. K. Mishra and D. Singh explored " assistance systems for elderly using web technologies." The study " assistance systems for elderly using web technologies" appeared in the International Journal of Advanced Research in Computer Science.
- [11] In 2020 J. H. Kim focused on "Design of user-centered web applications for users." The article "Design of user-centered web applications for users" was published in the Human-Computer Interaction Journal.
- [12] In 2021 K. Roy and S. Das discussed "Service matching algorithms in platforms." The paper "Service matching algorithms in platforms" was featured in the International Journal of Computer Science and Technology.
- [13] In 2022 N. M. Shah examined " service platforms and user experience." The research " service platforms and user experience" appeared in the Journal of Web Engineering.
- [14] In December 2011 I. Fette and A. Melnikov released the document titled "The WebSocket Protocol." The document "The WebSocket Protocol" is listed as RFC 6455 by the Internet Engineering Task Force. Can be accessed online at <https://www.rfc-editor.org/rfc/rfc6455>.
- [15] On March 26 2026 M. Cáceres and R. Grant published a snapshot titled "Geolocation" for the World Wide Web Consortium. The snapshot "Geolocation" is available online at <https://www.w3.org/TR/geolocation/>.
- [16] In 2020 R. Jain and S. Mehta presented "Real-time location tracking systems for safety applications." The paper "Real-time location tracking systems, for safety applications" appeared in IEEE Access.