National College of Information Technology (NACIT)
(Blantyre Campus)
Advanced Diploma in Computing (Level5DC)
Computing Project Proposal Paper

Mobile Application for Visually Impaired Indoor Navigation at LMJ Hospital

Name of Student: Chrispine Sibale
Student ID Number: 00186137
Email Address: raf33arn@gmail.com
Phone Number: 0981167823 / 0887833995

Name of Project Coordinator: Mrs Mwausegha
Name of Project Supervisor: Mrs Singini

Abstract

This project proposes the development of a mobile application designed to assist visually impaired individuals in navigating LMJ Hospital independently. The nature of the project involves integrating smartphone-based technologies—such as computer vision, natural language processing, and machine learning—to support real-time obstacle detection, voice-activated interaction, landmark recognition, and audio-based indoor navigation. The system will use on-device processing, operate offline using SQLite-stored maps, and offer a user-friendly voice interface. The intended outcome is to empower visually impaired patients by enhancing mobility, privacy, and access to healthcare services without needing physical changes to hospital infrastructure. The project is expected to contribute a scalable, affordable, and practical solution to digital accessibility within local medical environments.

Table of Contents

Abstract 2
1. Project Overview: Problem Statement and Aims 4
2. Initial Class Diagram 5
Figure 1: 5
3. Structural Design: Detailed Class Diagram 6
Figure 2: 6
4. Behavioural Design 7
4.1 Use Case Analysis and Modelling 7
Figure 3: 7
4.2 Illustrative Sequence Diagram for a Core Use Case 8
Figure 4: 9
5. Conclusion 9
6. Referencing 10
7. Appendix 11
A.1 Detailed Class Definitions 11

1. Project Overview: Problem Statement and Aims

Navigating the complex indoor environment of LMJ Hospital in Blantyre presents significant challenges for visually impaired individuals seeking medical care. The hospital's layout, with its multiple wings, specialized departments, and busy corridors, creates a difficult setting for autonomous movement. Traditional navigation aids like canes or guide dogs offer basic obstacle avoidance but cannot provide specific directional guidance or contextual information about the hospital's various departments, consultation rooms, or essential facilities. While hospital staff strive to assist, their urgent medical duties make consistent personal guidance impractical.

This reliance on memory, verbal directions, or accompanying family members compromises the independence and dignity of visually impaired patients and can potentially delay their access to timely medical attention. Current accessibility features like handrails and screen readers provide some assistance but fail to offer comprehensive spatial awareness within the hospital. These limitations can generate anxiety, causing patients to postpone necessary care while also increasing the workload for hospital staff.

The aim of this project is to address these challenges by developing a smartphone-based mobile application that provides a real-time, hands-free navigation solution tailored for the visually impaired. The application will leverage computer vision to detect obstacles, provide step-by-step audio guidance using offline hospital maps, and support hands-free interaction through voice commands. The system is designed to enhance accessibility, improve patient safety, and promote user autonomy without requiring modifications to the hospital's physical infrastructure.

2. Initial Class Diagram

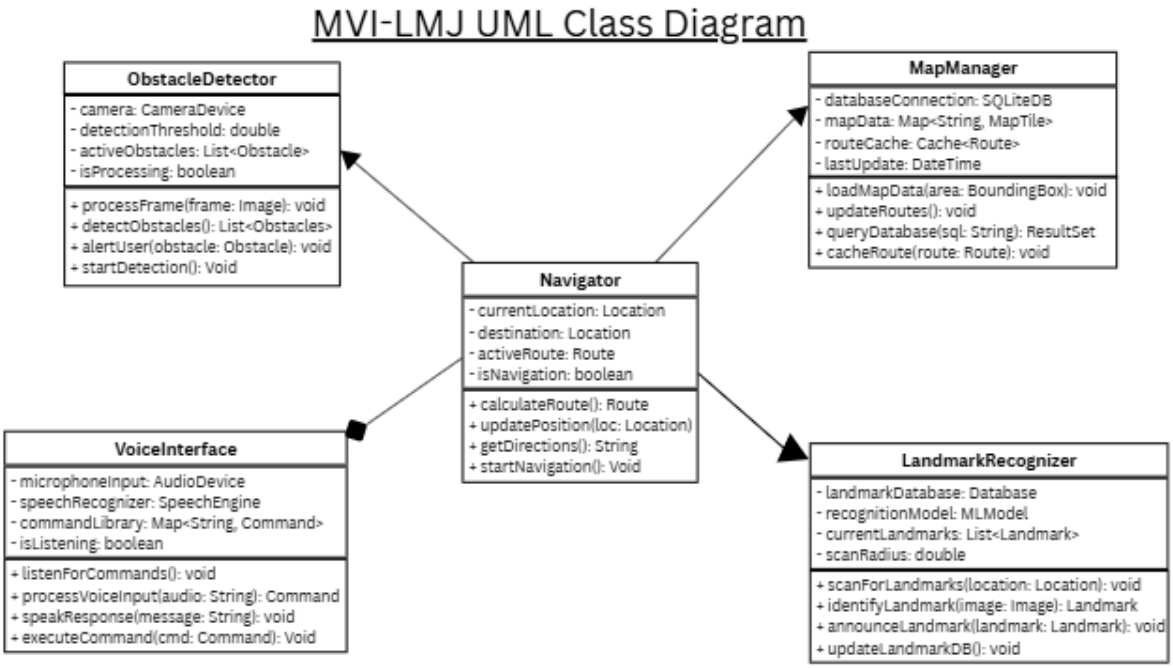

Figure 1: *Initial class diagram for MVI-LMJ*

The MVI-LMJ Hospital Navigation System employs a layered architecture with 20 interconnected classes. The **Navigator** serves as the central orchestrator, coordinating with **ObstacleDetector** for real-time safety, **LandmarkRecognizer** for location identification, **MapManager** for route planning, and **VoiceInterface** for hands-free interaction. **Patient** and **Administrator** classes represent system users, while domain classes (**Location**, **Route**, **Obstacle**, **Landmark**) model navigation entities. Technical infrastructure classes (**MLModel**, **SQLiteDB**, **CameraDevice**, **AudioDevice**, **SpeechEngine**) provide AI processing, offline storage, and hardware abstraction. This modular design ensures offline functionality, on-device processing, and safe navigation for visually impaired users in hospital environments.

3. Structural Design: Detailed Class Diagram

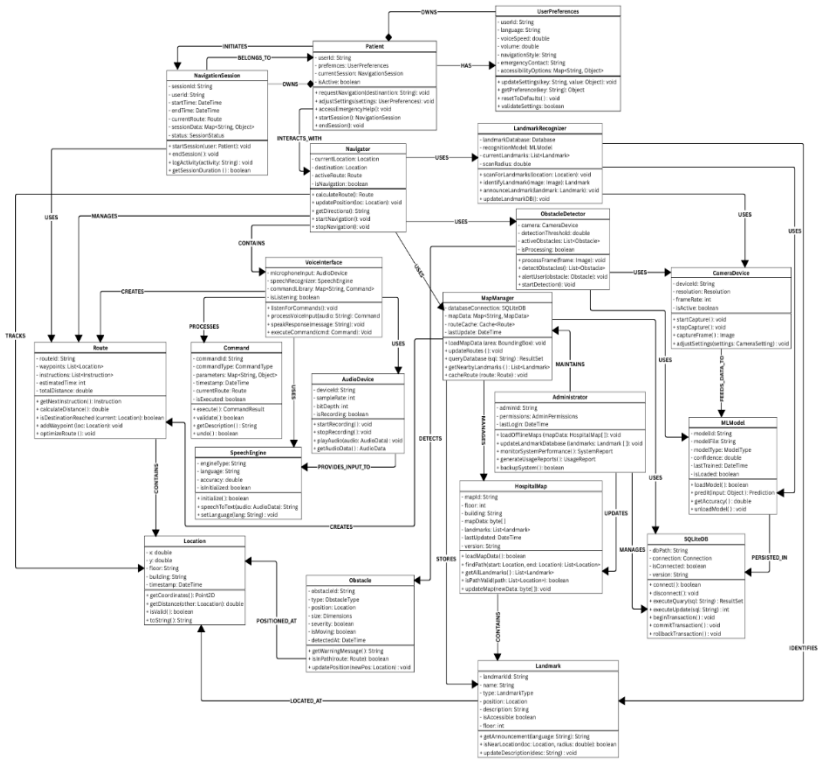

Figure 2: *Detailed class diagram for MVI-LMJ*

The system employs a coordinator pattern with the Navigator class serving as the central orchestrator that manages navigation logic and coordinates interactions between specialized subsystems. The structural design, as modelled in the UML class diagram (Figure 6 in the original document), demonstrates a clear separation of concerns, which is a core principle of Object-Oriented Methodology. This approach ensures that each class has a single responsibility and is minimally coupled, facilitating maintainability and extensibility.

The Navigator class maintains the system's navigation state through private attributes such as currentLocation, destination, and activeRoute. It exposes public methods for calculateRoute() and updatePosition() to manage the navigation flow. The Navigator establishes usage relationships with three key components: ObstacleDetector, LandmarkRecognizer, and MapManager. This design allows the Navigator to delegate complex tasks to dedicated subsystems without being responsible for their internal implementation.

The ObstacleDetector is responsible for real-time hazard detection through camera processing, while the LandmarkRecognizer identifies and announces predefined locations. The MapManager serves as a shared data repository, providing access to map data from the SQLite database. The VoiceInterface class is integrated through an aggregation relationship, indicating that the Navigator contains a voice interface component for processing speech commands and providing audio feedback. This architecture successfully translates the abstract principles of OOM—modularity, reusability, and encapsulation—into a concrete, operational design.

4. Behavioural Design

4.1 Use Case Analysis and Modelling

The behavioural design of the MVI-LMJ system is defined by its use cases, which describe the interactions between the actors and the system's functionalities. The primary actor is the **Patient**, a visually impaired individual who uses the system to navigate the hospital independently. The secondary actor is the **Administrator**, IT personnel responsible for system maintenance and data updates.

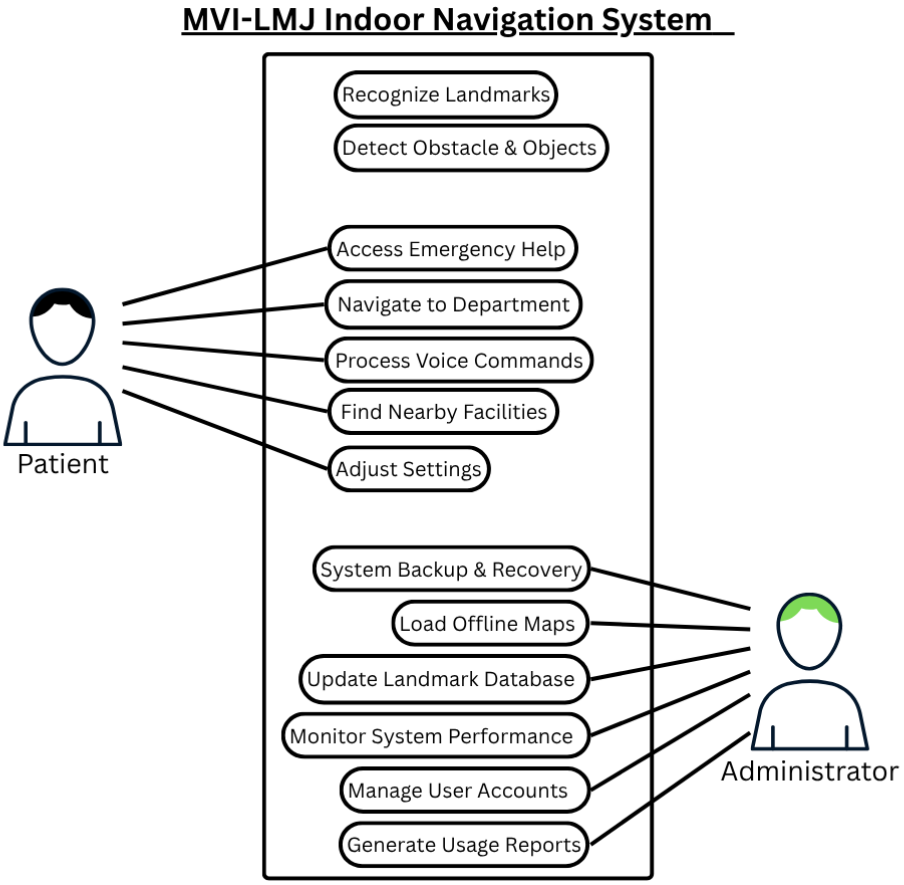

Figure 3: *Use Case Diagram for MVI-LMJ*

The use case diagram (Figure 3) presents a comprehensive model of the system's capabilities. It includes critical system-level use cases such as Recognize Landmarks and Detect Obstacles & Objects, which are fundamental to the system's core purpose of providing real-time spatial awareness and safety. The diagram also illustrates key patient-centric use cases, including Navigate to Department, Process Voice Commands, and Access Emergency Help, which directly address the user's need for autonomy and safety. The inclusion of administrator use cases, such as Load Offline Maps and Monitor System Performance, demonstrates a mature understanding of the system’s complete lifecycle, from end-user functionality to back-end maintenance and data management.

4.2 Illustrative Sequence Diagram for a Core Use Case

The sequence diagram for the use case Navigate to Department illustrates the dynamic interaction between the user and the system's core components. This diagram (Below) provides a dynamic view of how the static class structure collaborates to fulfill a specific user request. The process begins with the **Patient** actor initiating a request via a voice command, for example, "**Navigate to Radiology**."

1. The **Patient** speaks the command.
2. The **VoiceInterface** receives and processes the speech command.
3. The **VoiceInterface** sends the parsed request to the **Navigator**.
4. The **Navigator** receives the request, identifies the destination, and calls the calculateRoute() method within the **MapManager** to determine the optimal path.
5. The **MapManager** accesses the SQLite database, retrieves the necessary map data, and returns the calculated route to the **Navigator**.
6. The **Navigator** then begins the navigation loop, which involves:
   - Sending a request to the **ObstacleDetector** to analyse the real-time camera feed.
   - The **ObstacleDetector** processes the feed and returns any detected obstacles or objects to the **Navigator**.
   - The **Navigator** integrates this real-time information with the pre-calculated route and sends a structured audio instruction (e.g., "Obstacle ahead, turn right in five meters") to the **VoiceInterface**.
7. The **VoiceInterface** converts this instruction into speech and delivers it to the **Patient**.

This sequence of events continues until the LandmarkRecognizer identifies the final destination, at which point the **Navigator** provides a final voice instruction and terminates the navigation process. This flow demonstrates how the system’s modular components work in concert to provide a safe, responsive, and context-aware navigation experience for the user.

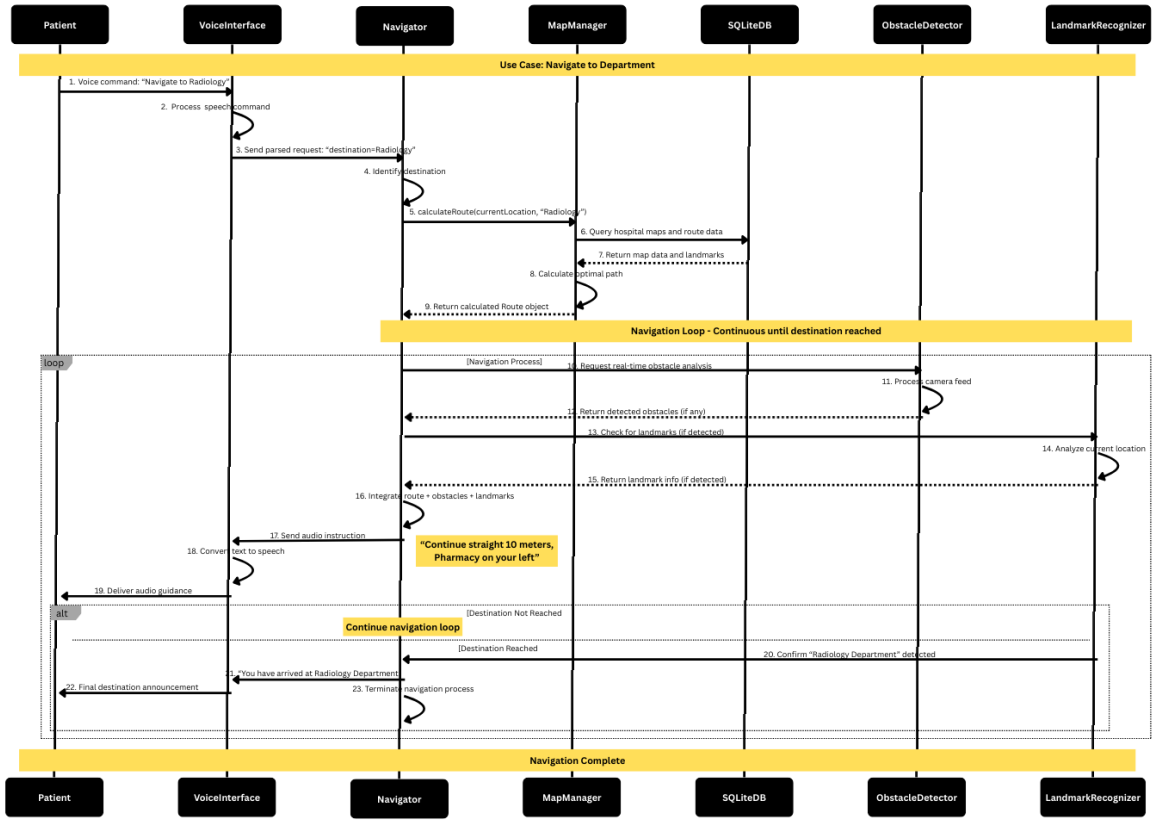

Figure 4: *Sequence Diagram for a Patient navigating to Radiology Department*

5. Conclusion

This design structure paper has provided a comprehensive overview of the MVI-LMJ project, detailing its purpose and design. The analysis has confirmed that the proposed solution effectively addresses the problem of indoor navigation for visually impaired individuals at LMJ Hospital by focusing on a unique, infrastructure-independent approach. The MVI-LMJ project's reliance on on-device, camera-based AI, inspired by solutions like Seeing AI (SeeingAI, 2025), provides a more practical and affordable solution. The architectural design, with its layered and modular structure, directly supports the project's core non-functional requirements for offline access, security, and maintainability. Furthermore, the detailed use case, and behavioural analysis, prove that the design is not only structurally sound but also functionally viable. The project is well-positioned to deliver a transformative solution that enhances the safety, independence, and dignity of visually impaired patients in the hospital environment.

6. Referencing

Apple, 2022. *App Store Preview: Navigine*. [Online] Available at: https://apps.apple.com/jo/app/navigine/id972099798 [Accessed 1 July 2025].

Apple, 2025. *Image showing Seeing AI's capabilities to Read Text, Describe Surroundings, Scan Products, Locate Items, and Recognize People*. [Online] Available at: https://apps.apple.com/us/app/seeing-ai/id972099798 [Accessed 1 July 2025].

Corada, 2025. *LowViz Guide Navigation*. [Online] Available at: https://www.corada.com/products/lowviz-guide-navigation [Accessed 1 July 2025].

Guardian, S. t., 2021. *Blindnes and Visual Impairement*. [Online] Available at: https://www.theguardian.com/society/2021/jul/22/new-device-could-help-visually-impaired-avoid-obstacles-research-suggests [Accessed 19 April 2025].

Guild, L., 2018. *A Hybrid Indoor Positioning System for the Blind and Visually Impaired Using Bluetooth and Google Tango*. [Online] Available at: https://lighthouseguild.org/a-hybrid-indoor-positioning-system-for-the-blind-and-visually-impaired-using-bluetooth-and-google-tango/ [Accessed 1 July 2025].

Kendrick, B., 2015. *LowViz Guide: Indoor Navigation for the Visually Impaired*. [Online] Available at: https://www.corada.com/products/lowviz-guide-navigation [Accessed 1 July 2025].

Liao, Y., 2024. *Seeing AI*. [Online] Available at: https://www.microsoft.com/en-us/garage/wall-of-fame/seeing-ai/ [Accessed 1 July 2025].

Lukman, Angreani, D., Muh & Bakti, R. Y., 2025. *The Use of Artificial Intelligence in The Development of Vision Aids for The Visually Impaired*. [Online] Available at: https://www.researchgate.net/publication/389070867_The_Use_of_Artificial_Intelligence_in_The_Development_of_Vision_Aids_for_The_Visually_Impaired [Accessed 1 July 2025].

Medicine, N. L. o., 2020. *Indoor Navigation Systems for Visually Impaired Persons: Mapping the Features of Existing Technologies to User Needs*. [Online] Available at: https://pubmed.ncbi.nlm.nih.gov/31979246/ [Accessed 1 July 2025].

Microsoft, 2016. *An app for visually impaired people that narrates the world around you*. [Online] Available at: https://www.microsoft.com/en-us/garage/wall-of-fame/seeing-ai/ [Accessed 1 July 2025].

Microsoft, 2022. *Navigine Platform*. [Online] Available at: https://appsource.microsoft.com/en-us/product/web-apps/navigine1585640136850.navigine_platform?tab=overview [Accessed 1 July 2025].

Perkins, D., 2022. *How AI is helping the visually impaired navigate their world*. [Online] Available at: https://www.microsoft.com/en-us/garage/wall-of-fame/seeing-ai/ [Accessed 1 July 2025].

PicoVoice, 2022. *NLP Applications in Voice Recognition*. [Online] Available at: https://picovoice.ai/blog/voice-recognition-NLP/ [Accessed 18 April 2025].

Riehle, T., 2008. *An Indoor Navigation System to Support the Visually Impaired*. [Online] Available at: https://umaine.edu/vemi/wp-content/uploads/sites/220/2016/08/Riehle-etalEMBC-2008-An-Indoor-Navigation-System-to-Support-the-Visually-Impaired.pdf [Accessed 1 July 2025].

SeeingAI, 2025. *Google Play*. [Online] Available at: https://play.google.com/store/apps/details?id=com.microsoft.seeingai [Accessed 1 July 2025].

Tara, 2025. *LowViz Guide*. [Online] Available at: https://www.microsoft.com/en-us/garage/wall-of-fame/seeing-ai/ [Accessed 1 July 2025].

Taylor&Francis, 2020. *Seeing AI*. [Online] Available at: https://taylorandfrancis.com/knowledge/Engineering_and_technology/Computer_science/Seeing_AI/ [Accessed 1 July 2025].

University, C., 2022. *Computer Vision and Pattern Recognition*. [Online] Available at: https://arxiv.org/abs/2212.04745 [Accessed 17 April 2025].

7. Appendix

A.1 Detailed Class Definitions

Class Name	Attributes	Methods
Navigator	currentLocation (Location), destination (Location), activeRoute (Route), isNavigating (boolean)	calculateRoute(): Route, updatePosition(loc: Location): void, getDirections(): String, startNavigation(): void, stopNavigation(): void ()
LandmarkRecognizer	landmarkDatabase (Database), recognitionModel (MLModel), currentLandmarks	scanForLandmarks(location: Location): void, identifyLandmark(image: Image): Landmark, announceLandmark(landmark:

	(List<Landmark>), scanRadius (double)	Landmark): void, updateLandmarkDB(): void
MapManager	databaseConnection (SQLiteDB), mapData (Map<String, MapData>), routeCache (Cache<Route>), lastUpdate (DateTime)	loadMapData(area: BoundingBox): void, updateRoutes(): void, queryDatabase(sql: String): ResultSet, getNearbyLandmarks(): List<Landmark>, cacheRoute(route: Route): void
VoiceInterface	microphoneInput (AudioDevice), speechRecognizer (SpeechEngine), commandLibrary (Map<String, Command>), audioLibrary (boolean)	listenForCommands(): void, processVoiceInput(audio: String): Command, speakResponse(message: String): void, executeCommand(cmd: Command): void
Patient	userId (String), preferences (UserPreferences), currentSession (NavigationSession), isActive (boolean)	requestNavigation(destination: String): void, adjustSettings(settings: UserPreferences): void, accessEmergencyHelp(): void, startSession(): NavigationSession, endSession(): void
Administrator	adminId (String), permissions (AdminPermissions), lastLogin (DateTime)	loadOfflineMaps(mapData: HospitalMap[]): void, updateLandmarkDatabase(landmarks: Landmark[]): void, monitorSystemPerformance(): SystemReport, generateUsageReports(): UsageReport, manageUserAccounts(users: Patient[]): void, backupSystem(): boolean

Location	x (double), y (double), floor (int), building (String), timestamp (DateTime)	getCoordinates(): Point2D, getDistance(other: Location): double, isValid(): boolean, toString(): String
Route	routeId (String), waypoints (List<Location>), instructions (List<Instruction>), estimatedTime (int), totalDistance (double)	getNextInstruction(): Instruction, calculateDistance(): double, isDestinationReached(current: Location): boolean, addWaypoint(loc: Location): void, optimizeRoute(): void
Obstacle	obstacleId (String), type (ObstacleType), position (Location), size (Dimensions), severity (SeverityLevel), isMoving (boolean), detectedAt (DateTime)	getWarningMessage(): String, isInPath(route: Route): boolean, updatePosition(newPos: Location): void
Landmark	landmarkId (String), name (String), type (LandmarkType), position (Location), description (String), isAccessible (boolean), floor (int)	getAnnouncement(language: String): String, isNearLocation(loc: Location, radius: double): boolean, updateDescription(desc: String): void
HospitalMap	mapId (String), floor (int), building (String), mapData (byte[]), landmarks (List<Landmark>), lastUpdated (DateTime), version (String)	loadMapData(): boolean, findPath(start: Location, end: Location): List<Location>, getAllLandmarks(): List<Landmark>, isPathValid(path: List<Location>): boolean, updateMap(newData: byte[]): void

UserPreferences	userId (String), language (String), voiceSpeed (double), volume (double), navigationStyle (String), emergencyContact (String), accessibilityOptions (Map<String, Object>)	updateSettings(key: String, value: Object): void, getPreference(key: String): Object, resetToDefaults(): void, validateSettings(): boolean
NavigationSession	sessionId (String), userId (String), startTime (DateTime), endTime (DateTime), currentRoute (Route), sessionData (Map<String, Object>), status (SessionStatus)	startSession(user: Patient): void, endSession(): void, logActivity(activity: String): void, getSessionDuration(): long, isActive(): boolean
Command	commandId (String), commandType (CommandType), parameters (Map<String, Object>), timestamp (DateTime), isExecuted (boolean)	execute(): CommandResult, validate(): boolean, getDescription(): String, undo(): boolean
MLModel	modelId (String), modelFile (String), modelType (ModelType), confidence (double), lastTrained (DateTime), isLoaded (boolean)	loadModel(): boolean, predict(input: Object): Prediction, updateModel(trainingData: Dataset): void, getAccuracy(): double, unloadModel(): void

SQLiteDB	dbPath (String), connection (Connection), isConnected (boolean), version (String)	connect(): boolean, disconnect(): void, executeQuery(sql: String): ResultSet, executeUpdate(sql: String): int, beginTransaction(): void, commitTransaction(): void, rollbackTransaction(): void
CameraDevice	deviceId (String), resolution (Resolution), frameRate (int), isActive (boolean)	startCapture(): void, stopCapture(): void, captureFrame(): Image, adjustSettings(settings: CameraSettings): void
AudioDevice	deviceId (String), sampleRate (int), bitDepth (int), isRecording (boolean)	startRecording(): void, stopRecording(): void, playAudio(audio: AudioData): void, getAudioData(): AudioData
SpeechEngine	engineType (String), language (String), accuracy (double), isInitialized (boolean)	initialize(): boolean, speechToText(audio: AudioData): String, textToSpeech(text: String): AudioData, setLanguage(lang: String): void