Software Requirements Specification

for

AUMC GuideVR

Version 1.0

Prepared by

Abdul Rafay	**233679**	**233679@students.au.edu.pk**
Abdul Rehman	**233711**	**233711@students.au.edu.pk**

Instructor: **Sir Ubaid Bin Zafar**

Course: **Software Requirement Engineering**

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Abdul Rafay and Abdul Rehman	Initial completed version of the Software Requirements Specification (SRS) for AUMC GuideVR. This version includes finalized functional and non-functional requirements, use case models, safety/security considerations, system constraints, domain and compatibility requirements, and complete appendices.	25/05/2025

1 Introduction

Navigating large university campuses like Air University Multan Campus (AUMC) can be overwhelming, especially for new students, visitors, and even some faculty members. **AUMC GuideVR** addresses this challenge through an innovative virtual reality-based navigation solution. The application offers immersive visual guidance, an interactive map, and location-specific assistance to enhance campus accessibility and user experience.

This section provides an overview of the project's purpose, its scope, the intended audience for this document, relevant abbreviations and terminology, and the conventions followed during the documentation process. It serves as a foundation for understanding the system requirements defined in the later sections.

1.1 Document Purpose

This Software Requirements Specification (SRS) document describes the requirements for the AUMC GuideVR system, an interactive, virtual reality-based navigation application for Air University Multan Campus (AUMC). This document refers to the system's initial version (v1.0) and outlines all essential functionalities, system behaviors, and interfaces required to fulfill user needs.

The purpose of this document is to provide a shared understanding among stakeholders, including clients, developers, testers, and evaluators, about what the system is, what it is intended to do, and how it should perform. It serves as a contractual reference point for development, implementation, and future enhancements of the application.

1.2 Product Scope

AUMC GuideVR is an innovative VR-based campus navigation application designed to assist users in easily navigating the Air University Multan Campus (AUMC). Through the use of virtual reality, it provides real-time, 3D visual guidance to users—including students, faculty, and visitors—to locate essential areas such as classrooms, hostels, cafeterias, auditoriums, and administrative offices. Key features include floor-based navigation, search and distance calculation, zoom and rotation support, drone-view layout, and classroom identification based on schedules.

This project aims to reduce confusion, save time, and improve campus accessibility, especially for new students and parents unfamiliar with the campus environment.

However, while every effort will be made to implement all major features, the inclusion of certain high-complexity modules (such as real-time drone view simulation, classroom tracking, or 360-degree VR overlays) may be limited by technical, budgetary, or timeline constraints. In such cases, static alternatives, simplified versions, or placeholder modules will be developed to maintain the application's integrity and usability.

1.3 Intended Audience and Document Overview

This document is intended for the following user categories:

- **Students:** To understand how the system will assist in navigating the campus and accessing location-based information.
- **Faculty Members:** To explore how the system can support efficient movement across campus and share announcements or directions.
- **Visitors:** To benefit from guided navigation and ease of finding specific locations within the campus.

The document is organized in sequential sections, beginning with a general overview (Section 1), followed by detailed descriptions of the system's functionality, interfaces, and features. Readers are advised to start with the overview (Section 1), then proceed to system features and specific requirements in the subsequent sections.

1.4 Definitions, Acronyms and Abbreviations

Acronym/Term	Definition
AUMC	Air University Multan Campus
AR	Augmented Reality
VR	Virtual Reality
GuideVR	Guide using Virtual Reality
UI	User Interface
UX	User Experience
GPS	Global Positioning System
API	Application Programming Interface
2D	Two-Dimensional
3D	Three-Dimensional
B1, B2, F1, F2	Basement and Floor Levels (e.g., Ground, First Floor, etc.)
SRS	Software Requirements Specification
XR	Extended Reality (AR, VR, and MR combined)
Drone View	A top-down aerial perspective, often simulated digitally
IEEE	Global organization for advancing technology and engineering standards
Gyroscope	Sensor that measures orientation and rotation

WebXR	API for creating VR/AR experiences on the web
WebGL	JavaScript API for rendering 3D graphics in browsers
Three.js	JavaScript library for 3D graphics using WebGL
A-Frame	Web framework for building VR experiences
HTML5	Markup language standard for modern web pages
CSS3	Styling language used to design web pages
React.js	JavaScript library for building user interfaces
Vue.js	Progressive JavaScript framework for UI development
MongoDB/ Firebase	Optional cloud database for storing building data, room info
GLTF / GLB	3D model file formats optimized for use in web-based VR
Blender	3D modeling tool used to create/edit/export.glb models
UML	Unified Modeling Language for software design diagrams
HTTPS	Secure version of HTTP using encryption (TLS/SSL)
PDF	Portable Document Format for readable file exchange
SVG	Scalable Vector Graphics format for 2D images
CAD	Computer-Aided Design for engineering and drafting
POI	Point of Interest (maps/geolocation term)
OS	Operating System that manages computer hardware/software
QR	Quick Response code for storing scannable info
FPS	Frames Per Second, a measure of display or video speed
TLS	Transport Layer Security, ensures secure internet communication
COMET	Web technique for server push (long-lived HTTP)
Git + GitHub	Version control and collaborative code management
LocalStorage	Browser storage used to save bookmarked locations in VR
IndexedDB	Optional browser database
PII	Personally identifiable information
IT	Information Technology, managing computers and data systems
OWASP	Organization focusing on web app security best practices

IOS	Apple's mobile operating system
GDPR	EU regulation for data privacy and protection
Vercel	Platform to deploy front-end web apps instantly
AWS	Amazon Web Services, cloud computing platform
PECA	Pakistan Electronic Crimes Act
Mapbox	Useful for 2D map apps

1.5 Document Conventions

The formatting and writing of this document follow IEEE standard guidelines and best practices for software documentation. The following conventions have been applied:

- **Formatting Conventions:**
  - **Font Type**: Arial
  - **Font Size**: 11pt (standard text), 14pt (section headings)
  - **Text Style**: Regular for content, *italics* for placeholder comments (which have been removed in this version)
  - **Spacing**: Single line spacing
  - **Margins**: 1" on all sides
  - **Section Headings**: Bold and numbered according to IEEE structure

- **Naming Conventions:**
  - **Project Name**: Always referred to as AUMC GuideVR in titles and formal mentions.
  - **File Names**: Written in lowercase with underscores, e.g., map_module.py, login_page.html.
  - **Class Names**: Written in PascalCase, e.g., UserLogin, CampusMap.
  - **Function/Method Names**: Written in camelCase, e.g., getUserLocation(), loadVROverlay().
  - **Variables**: Written in camelCase, e.g., currentLocation, userType.
  - **Constants**: Written in ALL_CAPS with underscores, e.g., MAX_DISTANCE, FLOOR_COUNT.

1.6 References and Acknowledgments

This Software Requirements Specification (SRS) document was developed with valuable input and guidance from the following individuals and resources:

- **Abdur Rehman** – for providing suggestions regarding user interface design and feature prioritization.

- **Muhammad Arslan Dilawar** – for contributing insights related to 3D navigation and usability concerns.
- **Muhammad Awais** – for supporting the design of interactive and VR-based functionalities.
- **Sir Ubaid Bin Zafar (Project Supervisor)** – for his professional guidance, technical feedback, and constructive review throughout the requirements engineering process.
- **Various online articles, blogs, and documentation** related to web-based VR development, campus navigation systems, and UI/UX best practices.

2 Overall Description

2.1 Product Overview

AUMC GuideVR is a **new, self-contained** augmented reality-based navigation solution specifically developed for the **Air University Multan Campus (AUMC)**. This product does not extend any previous version or system—it is built from the ground up to enhance campus navigation, particularly for new students, and visitors unfamiliar with the campus layout. The application aims to reduce confusion, save time, and improve the on-campus experience by using VR overlays, interactive floor-based navigation, and smart mapping technologies.

This product operates independently but may connect with certain optional third-party services like map APIs or schedule databases if needed. It interacts with the user's mobile device sensors (camera, GPS, gyroscope) to render live navigation assistance and location tracking.

2.2 Product Functionality

- **VR-Based Navigation:** Guiding users visually through live camera view using VR paths and arrows.
- **Floor-Based Navigation:** Selecting specific floors of buildings to navigate within them.
- **Search Locations:** Searching for places like classrooms, offices, labs, cafeterias, etc.
- **Distance Calculator:** Displaying the distance between the current location and the destination.
- **Zoom Functionality:** Allowing users to zoom in and out of the VR view and the map interface, primarily used during step-by-step VR navigation for better clarity in close spaces.
- **Rotate Functionality:** Allowing users to rotate the VR view or map manually or using device orientation, especially useful for aligning direction in indoor navigation.
- **Drone View Mode:** Optional bird's-eye view of the entire campus with building markers.
- **Faculty/Office Locator:** Helping users find the location of specific classrooms or offices, such as those of professors, administrative staff, HR, or security departments.
- **360 Degree Visualization:** Users can view buildings or locations from all angles within VR or 3D mode (rotate the scene).
- **Favorites/Bookmarks:** Saving commonly visited places for quick access.

2.3 Design and Implementation Constraints

- **Platform Dependency**: The system must be a **browser-based web application** accessible on modern desktop and mobile browsers (e.g., Chrome, Firefox, Edge). Mobile-first design should be prioritized for better usability on smartphones.
- **Web-Based VR**: VR features must be implemented using **WebXR** or other web-compatible VR libraries. Integration with **WebGL**, **Three.js**, or **A-Frame** is recommended for immersive 3D rendering and virtual environment creation.
- **Device Access:** The browser must support access to device orientation and motion sensors through web APIs. (Note: camera and GPS are not required in VR.) Users may be prompted to allow sensor or headset access for proper VR functioning.
- **Tech Stack Constraints**:
    - **Frontend**: HTML5, CSS3, JavaScript, and possibly frameworks like **React.js** or **Vue.js**.
    - **Backend** (if applicable): Node.js with Express or a lightweight backend API for dynamic data (e.g. classroom finder).
    - **VR/3D Frameworks**: A-Frame, WebXR, Three.js.
- **Design Methodology**: The system must be developed using the **COMET method** for software design.
    - *Reference: Gomaa, H. (2000). Designing Software Product Lines with UML.*
- **Modeling Language**: All design models and diagrams must use **UML (Unified Modeling Language)**.
- **Security Considerations:** The application must run securely over HTTPS. VR device access must follow browser security protocols (permission-based access for sensors or headsets if applicable).
- **Performance**: VR and 3D rendering must be optimized to run smoothly, especially on lower-end mobile devices and browsers.
- **Browser Limitations**: Not all browsers or older versions may support full VR functionality. The product may not support legacy browsers like Internet Explorer.

2.4 Assumptions and Dependencies

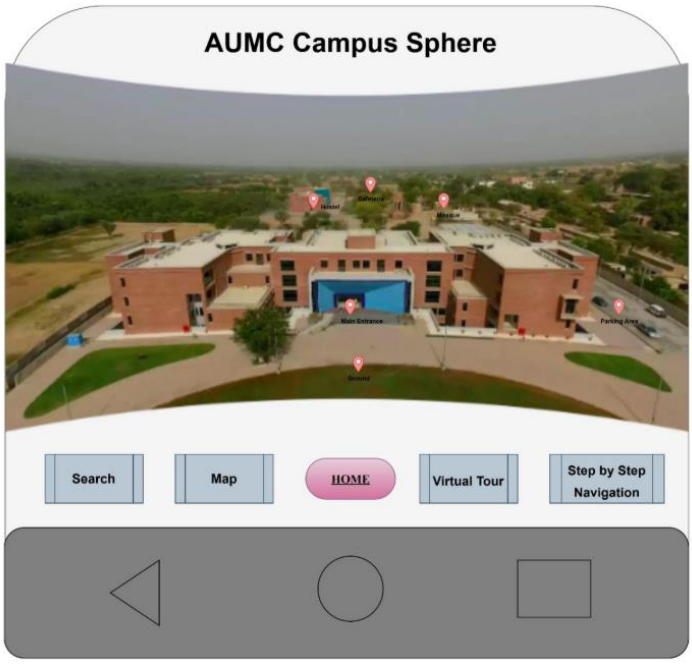
- Users will access the system through a modern web browser on mobile or desktop devices with WebGL and WebXR support for 3D rendering and immersive interaction.
- The system does not require continuous internet access during navigation. Once loaded, all virtual assets (campus model, room labels, navigation system) are accessible offline within the browser.
- The web application assumes that users' devices basic motion/orientation sensors (for look-based navigation in VR), though camera access and GPS are not required for VR-based navigation.
- A complete and up-to-date floor plan of AUMC will be provided by the university in a standard digital format (PDF, SVG, or CAD), suitable for integration into the virtual campus environment.

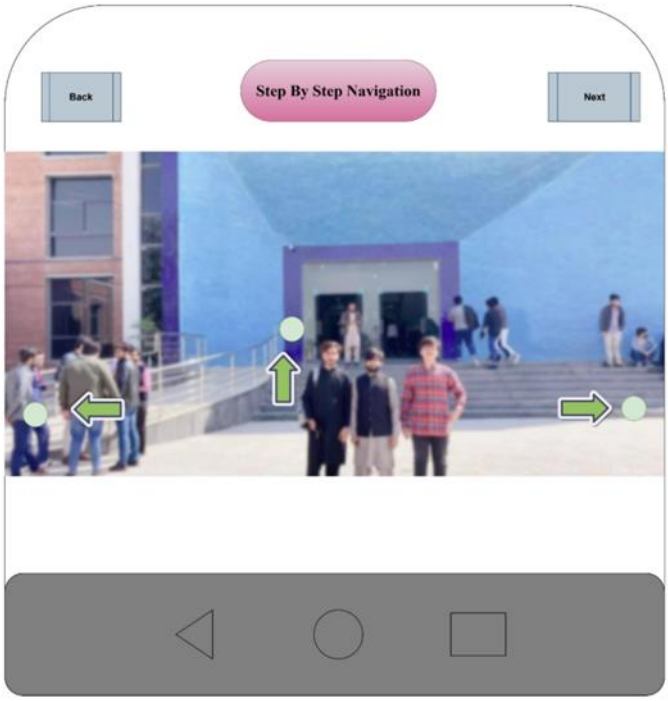
- Faculty, office, and classroom data (e.g., room numbers, labels like “Automation Lab” or “Finance Office”) must be available in a structured format (such as Excel or database) to be visually represented within the VR environment as nameplates or signs on doors.
- All identification (e.g., ‘Faculty Office’, ‘CS Lab’, ‘Professor Room’) will be visually displayed on doors or entry points inside the virtual campus as part of the VR scene.
- The VR experience will be fully client-side, meaning it runs directly in the browser without requiring server interaction during usage. However, an initial load of assets may require a one-time connection.
- The implementation of drone view, real-time schedule integration, or advanced VR features is dependent on data availability. These features may be replaced with static or simulated alternatives if real-time integration is not feasible.
- Users are expected to have basic digital literacy, such as operating a web browser, enabling permissions (camera, location), and navigating through an interactive interface. The system is not responsible for user errors caused by lack of basic device or browser usage knowledge.
- Users are responsible for keeping their browsers updated to modern versions (e.g., Chrome 100+, Firefox 100+). Outdated browser versions may not support necessary APIs such as WebGL or WebXR, which are critical for VR and 3D features.
- Device performance, especially on low-end smartphones or older laptops, may affect VR rendering quality. These limitations are considered hardware-related and not system faults.
- The application is currently not designed for accessibility tools (e.g., screen readers, keyboard-only control), so users relying on such assistive technologies may face limitations in using the VR interface.
- The system assumes users are viewing the VR campus on standard-sized screens (typical smartphones, tablets, laptops). Extremely small, low-resolution, or unusual aspect ratios may reduce visual clarity.
- Pop-up blockers or aggressive browser privacy settings should not interfere, as the system does not require camera, GPS, or permission-based features. Minimal permissions are needed.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces





3.1.2 Hardware Interfaces

The system shall interact with the following hardware interfaces to enable the functionality of the web-based VR navigation system. It also assumes minimum hardware capabilities on the user's device to deliver a smooth and usable experience.

3.1.2.1 Orientation Sensors (Gyroscope & Accelerometer)

- The system shall utilize motion/orientation sensors (if available) to detect device rotation, allowing users to look around naturally in VR mode (especially on mobile devices).
- These sensors enhance immersion in mobile viewports by adjusting the virtual perspective based on user movement.
- **Limitation**: On desktops or devices without such sensors, the user will navigate via mouse/keyboard instead.

3.1.2.2 Touch Interface

- The system shall support touch gestures (tap, pinch to zoom, drag) on mobile and keyboard/mouse interaction on desktops for navigating the virtual campus.
- Users will interact with the VR environment to move between locations, enter rooms, or activate UI components.

3.1.3 Device Capability Requirements

3.1.3.1 RAM & Processor

- The system assumes the user's device has a **minimum of 2 GB RAM** and a **dual-core processor or better** to support smooth loading and rendering of the virtual campus.
- Devices with lower specifications may experience lags or UI stuttering, particularly during VR rendering or large map interactions.

3.1.3.2 GPU / Graphics Processing

- A device with a **GPU or integrated graphics** supporting **WebGL and WebXR** is required for rendering the VR environment.
- Devices lacking WebGL/WebXR support or outdated graphics drivers may not load the virtual experience correctly.

3.1.3.3 Display / Screen Resolution

- The system is optimized for devices with a **minimum screen resolution of 720p (1280x720)**.
- Visual clarity may be reduced on very small screens or non-standard display ratios.

3.1.3.4 Browser Compatibility

- The system requires a **modern web browser** with **WebGL**, **WebXR**, and **JavaScript** support (e.g., latest Chrome, Firefox, Edge).
- **Limitation:** Older browsers or restrictive environments (e.g., institutional networks with script-blocking policies) may block or degrade essential VR functionality.

3.1.4 Software Interfaces

- Web Browser (Frontend Platform)
    - The system shall be accessed through modern web browsers on desktop and mobile devices that support **WebXR**.
    - The frontend shall render the immersive VR campus scene and all interface components (menus, location tags, navigation controls).
    - **Limitation:** Older or unsupported browsers may fail to render the 3D scene or block WebXR functionality.

- VR Library / Framework
    - The system shall use a VR development framework such as A-Frame, WebXR, and optionally Three.js to render and manage the virtual campus environment.
    - These libraries handle scene loading, object placement, spatial interaction, and motion tracking.
    - **Limitation:** Framework compatibility may vary by browser/device; fallback designs should be included for unsupported devices.
- Backend Server (Optional)
    - A backend server may be included to serve structured data such as:
        - Faculty/office/classroom room numbers
        - Building and floor mappings
        - POIs (Points of Interest) such as “Finance Office”, “Automation Lab”, or “Admission Counter”
    - **Note:** The system does not require live schedule data or real-time tracking.
    - The backend may expose APIs for:
        - Searching or retrieving department and room location data
        - Serving static metadata for campus elements (labels, directions)
- Database
    - The system may connect to a local or cloud-based database to store and retrieve:
        - Room numbers, office names, and faculty associations
        - Campus floor layouts and department locations
        - POIs (e.g., cafeteria, auditorium, library)
    - Preferred options: **MongoDB, Firebase, MySQL, or PostgreSQL**
    - **Limitation:** If the database is unavailable, search and dynamic display features may fall back to static content.
- Internet Connectivity
    - The system does not require continuous internet access after the initial load.
    - All 3D scenes, object data, and text labels will be locally available post-load.
    - **Limitation:** Users must load the experience at least once via internet. Updates or dynamic content (if any) will require internet re-connection.
- Operating System Compatibility
    - The system is designed to run on devices with the following minimum OS requirements:
        - Windows 11, macOS 12+, Android 10+, or iOS 13+.
        - Linux-based systems are supported if modern browsers are installed.
    - Older operating systems may face:
        - Incompatibility with WebXR
        - Poor rendering or performance issues
        - Blocked browser APIs
- Optional: Third-party APIs
    - The system may optionally integrate external services for enhanced visuals:
        - Mapbox or Google Maps: to extract 2D layout references or non-interactive maps
        - Drone video embedding (static only): for optional top-view campus previews

- **Limitation:** These integrations are non-critical. If APIs are rate-limited or paid, fallback static content will be used instead.

3.2 Functional Requirements

3.2.1 Module 1: Indoor Navigation

3.2.1.1 F1: VR-Based Indoor Navigation

- F1.1: The system shall allow users to enter immersive VR mode from the main interface.
- F1.2: The system shall allow users to manually select their current position within the virtual campus.
- F1.3: The system shall optionally use device orientation (gyroscope) to adjust the view direction.
- F1.4: The system shall allow users to select a target location (e.g., classroom, admin office).
- F1.5: The system shall display virtual directional arrows within the 3D scene to guide users to their destination.
- F1.6: The system shall provide step-by-step VR guidance as the user navigates through the virtual environment.
- F1.7: The system shall update the navigation path visually as the user moves through the scene.
- F1.8: The system shall re-calculate the navigation path if the user deviates from the original route in the VR scene.
- F1.9: The system shall display text labels (e.g., “CS Lab”, “Auditorium”) on virtual doors and areas.
- F1.10: The system shall display recognizable icons and markers on 3D room locations.

3.2.1.2 F2: Floor-Based Navigation

- F2.1: The system shall allow users to switch between different campus floors (e.g., B2, B1, Ground, F1, F2).
- F2.2: The system shall update visible locations on the interface based on the selected floor.
- F2.3: The system shall update the navigation path according to the selected floor.
- F2.4: The system shall visually indicate the current floor within the VR environment.

3.2.1.3 F3: Search & Explore Campus Locations

- F3.1: The system shall provide a search bar to find locations within the campus.
- F3.2: The system shall provide auto-suggestions as the user types.
- F3.3: The system shall display search results inside the virtual campus view.
- F3.4: The system shall allow users to navigate directly in VR to a selected search result.
- F3.5: The system shall allow users to browse categorized places (labs, offices, classrooms, etc.).

3.2.1.4 F4: Zoom In / Out

- F4.1: The system shall allow users to zoom in during VR navigation using touch or scroll.
- F4.2: The system shall allow users to zoom out during VR navigation using touch or scroll.
- F4.3: The system shall allow users to zoom in on the interactive VR map.
- F4.4: The system shall allow users to zoom out on the interactive VR map.
- F4.5: The system shall preserve the clarity of text labels and icons when the user zooms in.
- F4.6: The system shall preserve the clarity of text labels and icons when the user zooms out.
- F4.7: The system shall restrict maximum zoom-in level to prevent visual distortion.
- F4.8: The system shall restrict maximum zoom-out level to maintain spatial context.

3.2.1.5 F5: Rotate

- F5.1: The system shall allow users to rotate the view manually using touch input.
- F5.2: The system shall allow users to rotate the view manually using mouse input.
- F5.3: The system shall allow rotation using device orientation (if available).
- F5.4: The system shall allow users to rotate the full VR scene using touch input.
- F5.5: The system shall allow users to rotate the map using mouse input.
- F5.6: The system shall update directional arrows based on current view angle.
- F5.7: The system shall update visual icons based on current view angle.
- F5.8: The system shall allow users to reset the scene orientation to default.
- F5.9: F5.8: The system shall allow users to reset the map orientation to default.

3.2.1.6 F6: Distance Calculation

- F6.1: The system shall calculate the distance between the user's current location and the selected destination.
- F6.2: The system shall display the calculated distance to the user on the interface.
- F6.3: The system shall update the displayed distance dynamically as the user navigates.
- F6.4: The system shall re-calculate the distance dynamically if the user changes direction or path.
- F6.5: The system shall notify the user if the destination becomes unreachable in VR (e.g., blocked or inaccessible floor).

3.2.1.7 F7: Faculty / Office Finder

- F7.1: The system shall allow users to select a department from a predefined list.
- F7.2: The system shall allow users to select a staff member or office from a predefined list.
- F7.3: The system shall highlight and label the corresponding room in the VR environment.
- F7.4: The system shall guide the user via VR navigation to that location.

3.2.1.8 F8: Location Access and Permissions

- F8.1: The system shall allow users to manually select their starting point from a list or map.
- F8.2: The system shall provide predefined points for entering the VR environment (e.g., “Start from Admin Block”).

- F8.3: The system functions independently of real-world indoor positioning systems.

3.2.1.9 F9: 360-Degree Visualization

- F9.1: The system shall allow users to view 360-degree panoramic images of rooms or areas (e.g., auditorium, cafeteria).
- F9.2: The system shall allow users to rotate the 360-degree indoor view using touch input.
- F9.3: The system shall allow users to rotate the 360-degree indoor view using mouse input.
- F9.4: The system shall support zooming in and out within the 360-degree indoor view.
- F9.5: The system may display static virtual markers inside the 360 view to indicate key areas.
- F9.6: The system may update static virtual markers inside the 360 view to indicate key areas.
- F9.7: The system shall allow Users to select POIs from the 360 view to start navigation.
- F9.8: The system shall allow switching between panoramic views.
- F9.9: The system shall allow switching between 3D VR environment views.

3.2.2 Module 2: Outdoor Navigation

3.2.2.1 F10: VR-based Outdoor Navigation

- F10.1: The system shall allow users to initiate outdoor navigation mode within the virtual campus scene.
- F10.2: The system shall allow users to manually select their current outdoor location in the VR environment.
- F10.3: The system shall use device orientation (if available) to adjust the user's view direction; no GPS will be used.
- F10.4: The system shall allow users to select an outdoor destination (e.g., parking area, admin block, outdoor seating area).
- F10.5: The system shall display virtual directional arrows or guiding markers within the VR scene to lead users to the selected outdoor location.
- F10.6: The system shall guide users visually through each navigation step using spatial cues inside the virtual environment.
- F10.7: The system shall update the visible path based on the user's movement through the virtual campus.
- F10.8: The system shall re-calculate the navigation path if the user deviates from the designated route within the VR scene.
- F10.9: The system shall display text labels (e.g., “Main Gate”, “Library Block”) on outdoor buildings and structures.
- F10.10: The system shall overlay recognizable icons (e.g., gate, garden, café) within the VR environment to mark outdoor POIs.

3.2.2.2 F11: Drone View Mode

- F11.1: The system shall display a bird’s-eye drone view of the AUMC campus.

- F11.2: The system shall display labeled buildings in drone view mode.
- F11.3: The system shall display Points of Interest (POIs) such as cafeterias, labs, and offices in drone view mode.
- F11.3: The system shall allow users to select a building from the drone view to start navigation.

3.2.2.3 F12: Search and Explore Campus Location

- F12.1: The system shall provide a search bar to find outdoor campus locations.
- F12.2: The system shall provide auto-suggestions while typing in the outdoor search bar.
- F12.3: The system shall display outdoor search results (e.g., admin block, parking, canteen) in the VR interface.
- F12.4: The system shall allow users to initiate VR navigation directly from a selected outdoor search result.
- F12.5: The system shall allow users to explore categorized outdoor places (e.g., parking, gates, lawns, outdoor seating).

3.2.2.4 F13: Zoom In / Out

- F13.1: The system shall allow users to zoom in while using VR-based step-by-step navigation.
- F13.2: The system shall allow users to zoom out while using VR-based step-by-step navigation.
- F13.3: The system shall allow users to zoom in on the general virtual campus map.
- F13.4: The system shall allow users to zoom out on the general virtual campus map.
- F13.5: The system shall preserve the clarity of visual markers when the user zooms in.
- F13.6: The system shall preserve the clarity of visual markers when the user zooms out.
- F13.7: The system shall restrict maximum zoom-in level to prevent visual distortion.
- F13.8: The system shall restrict maximum zoom-out level to maintain spatial context.

3.2.2.5 F14: Rotate

- F14.1: The system shall allow users to rotate the VR view manually using touch input.
- F14.2: The system shall allow users to rotate the VR view manually using mouse input.
- F14.3: The system shall allow users to rotate the VR view using physical device orientation (if sensors are available).
- F14.4: The system shall allow users to rotate the virtual campus map using touch input.
- F14.5: The system shall allow users to rotate the virtual campus map using mouse input.
- F14.6: The system shall update directional arrows during VR view rotation.
- F14.7: The system shall update visual icons during VR view rotation.
- F14.8: The system shall reset the VR view rotation angle to the default orientation when requested by the user.
- F14.9: The system shall reset the general map rotation angle to the default orientation when requested by the user.

3.2.2.6 F15: Distance Calculation

- F15.1: The system shall calculate the distance between the user's current location and the selected destination.
- F15.2: The system shall display the calculated distance to the user on the interface.
- F15.3: The system shall update the displayed distance in real-time as the user navigates the virtual path.
- F15.4: The system shall re-calculate the distance dynamically if the user changes direction or path.
- F15.5: The system shall notify the user if the destination becomes unreachable due to navigation constraints inside the virtual environment.

3.2.2.7 F16: Location Access and Permissions

- F16.1: The system shall allow users to manually select their starting location in the VR environment.
- F16.2: The system shall operate fully without needing real-time location tracking or live camera input.
- F16.3: The system shall function independently of outdoor GPS data or sensor-based positioning.

3.2.2.8 F17: 360-Degree Visualization

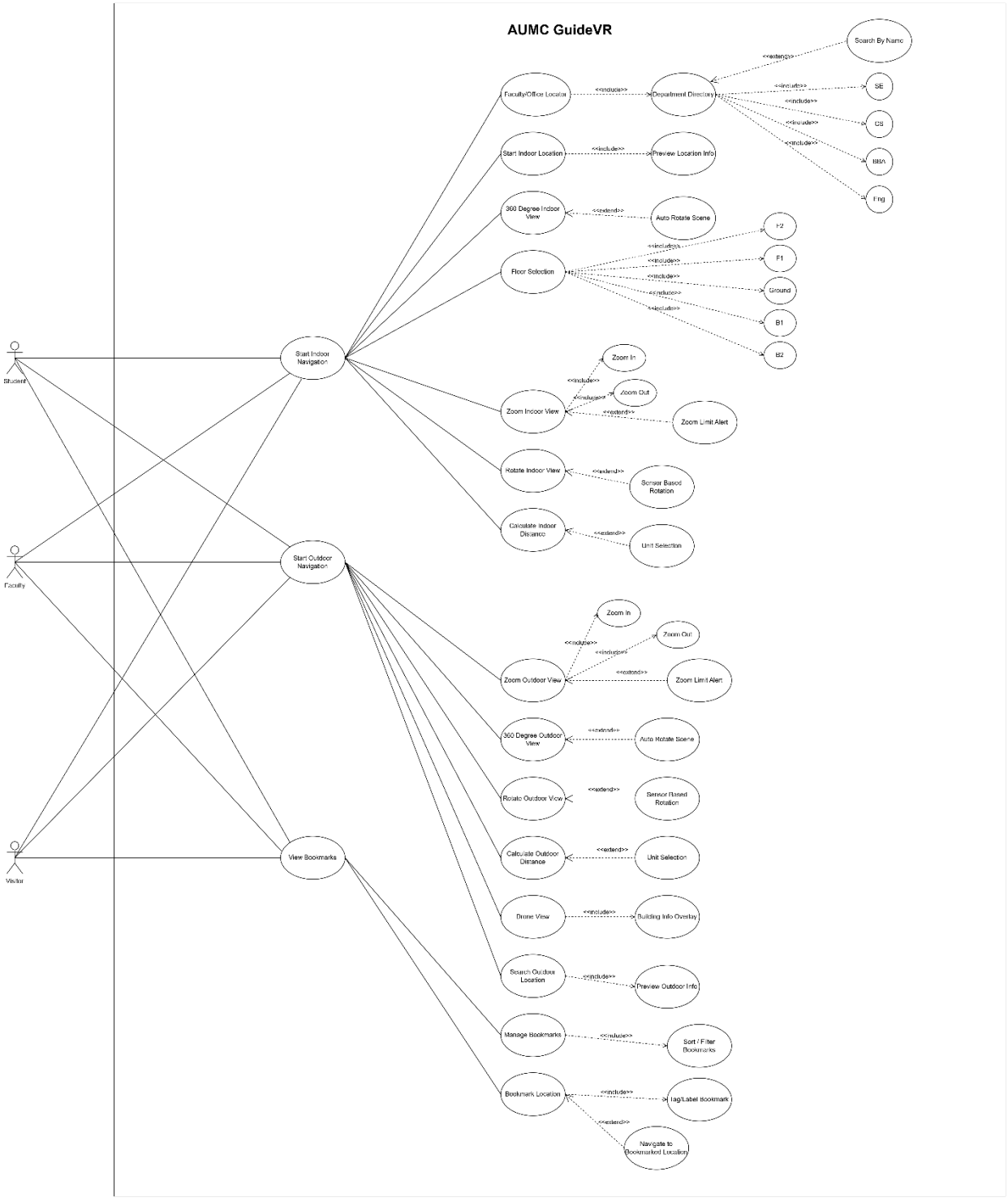
- F17.1: The system shall provide a 360-degree panoramic view of the user’s current outdoor location.
- F17.2: The system shall allow users to rotate the 360-degree outdoor view using touch input.
- F17.3: The system shall allow users to rotate the 360-degree outdoor view using mouse input.
- F17.4: The system shall support zooming in and out within the 360-degree outdoor view.
- F17.5: The system shall overlay virtual markers for outdoor points of interest within the 360-degree outdoor view.
- F17.6: The system shall allow users to explore points of interest within the 360-degree view.
- F17.7: The system shall allow users to select outdoor locations from the 360-degree view to start navigation.
- F17.9: The system shall provide an option to reset the 360-degree outdoor view to the default orientation.
- F17.10: The system shall allow users to toggle between the 360-degree outdoor view and VR-based outdoor navigation mode.

3.2.3 Module 3: Bookmarked Location

3.2.3.1 F18: Favorite / Saved Location

- F18.1: The system shall allow users to manually bookmark any selected location.
- F18.2: The system shall store each bookmarked location in the user's local browser storage.
- F18.3: The system shall provide a list view of all bookmarked locations accessible from the main interface.
- F18.4: The system shall allow users to initiate navigation directly from a bookmarked location.
- F18.5: The system shall allow users to remove a location from their list of bookmarks.
- F18.6: The system shall display the name of each bookmarked location in the list
- F18.7: The system shall display the type (e.g., classroom, office, cafe) of each bookmarked location in the list.
- F18.8: The system shall allow users to sort bookmarked locations alphabetically.
- F18.9: The system shall allow users to search within their list of bookmarked locations.
- F18.10: The system shall highlight bookmarked locations with a distinct icon on the map.
- F18.11: The system shall highlight bookmarked locations with a marker on the map.
- F18.12: The system shall retain all bookmarked locations in browser storage unless manually deleted by the user.

3.3 Use Case Model



3.3.1 Use Case #1 (Start Indoor VR Navigation)

- **Author –** Abdul Rafay
- **Actors –** Student / Faculty / Visitor
- **Purpose** – To guide users indoors via VR navigation with step-by-step directions.
- **Preconditions** – User is on the interface with VR environment loaded.
- **Post conditions** – User is guided to the selected indoor destination through immersive VR movement.
- **Flow of Events**
    - User opens the app and selects "Indoor Navigation".
    - System loads the 3D indoor VR environment.
    - User selects current location manually.
    - User selects a destination.
    - System shows virtual directional arrows inside the VR scene.
    - System updates the navigation path based on user's virtual movement.
    - Navigation ends upon reaching the destination.
- **Alternative Flow**
    - If starting location is not selected, prompt user to choose one.
    - If movement is restricted (e.g., outside scene bounds), prompt reset option.

3.3.2 Use Case #2 (Search and Navigate to a Location)

- **Author** – Abdul Rehman
- **Actors** – Student / Faculty / Visitor
- **Purpose** – To let users search for a place and initiate navigation.
- **Preconditions** – User is on main interface with virtual campus loaded.
- **Post conditions** – Navigation starts from search result.
- **Flow of Events**
    - User types in search bar.
    - Auto-suggestions appear.

- User selects a result (e.g., Lab, Admin Block).
- System offers VR navigation or drone view.
- If chosen, navigation begins in the VR environment.

- **Alternative Flow**
  - If search yields no results, suggest nearby POIs.
  - If VR rendering fails (e.g., device unsupported), display location in static 2D scene.

3.3.3 Use Case #3 (View Drone Mode)

- **Author** – Abdul Rafay
- **Actors** – Student / Faculty / Visitor
- **Purpose** – To let users view a top-down “drone” view of the campus for orientation.
- **Preconditions** – System has loaded the virtual top-down scene.
- **Post conditions** – User can explore the layout and optionally initiate navigation. .
- **Flow of Events**
  - User clicks “Drone View”.
  - System loads campus overview with building labels.
  - User clicks on a building (e.g., Library).
  - System shows details and offers navigation.
- **Alternative Flow**
  - If drone view data unavailable, fallback to 2D layout.
  - If assets are not loaded yet, show loading screen or retry option.

3.3.4 Use Case #4 (Bookmark Location)

- **Author** – Abdul Rehman
- **Actors** – Student / Faculty / Visitor
- **Purpose** – To save favorite places for quick access.
- **Preconditions** – User is navigating or searching a place.
- **Post conditions** – The selected location is stored in bookmarks and accessible later.
- **Flow of Events**

- User selects a location in the VR environment or via search.
- User clicks “Bookmark”.
- System saves it to local browser storage.
- User views it later in “My Bookmarks”.
- Navigation can be started directly from the list.

- **Alternative Flow**
  - If browser storage disabled, prompt error.
  - If location already bookmarked, show warning.

3.3.5 Use Case #5 (Start Outdoor VR Navigation)

- **Author** – Abdul Rafay
- **Actors** – Student / Faculty / Visitor
- **Purpose** – To guide users across outdoor campus spaces using VR.
- **Preconditions** – Outdoor VR environment is loaded.
- **Post conditions** – User is navigated to a selected outdoor POI within the virtual campus.
- **Flow of Events**
  - User selects “Outdoor Navigation”.
  - System loads the 3D outdoor scene.
  - User selects destination (e.g., Parking, Admin Block).
  - System displays virtual directional arrows and highlights the path.
  - As the user moves through the virtual space, navigation updates accordingly.
  - Upon arrival, navigation ends.
- **Alternative Flow**
  - If destination is unreachable (e.g., blocked floor), notify the user and suggest alternate route.
  - If assets for the outdoor scene are unavailable, display fallback layout or notify the user.

4 Other Non-functional Requirements

4.1 Performance Requirements

4.1.1.1 PR1: VR Scene Activation Time

- **Requirement:** The system shall activate the indoor or outdoor VR navigation environment within **3 seconds** after user selection.
- **Rationale:** Fast activation ensures a smooth and responsive experience and reduces waiting time.

4.1.1.2 PR2: 3D Scene Load Time

- **Requirement:** The system shall load essential 3D models and textures within **5 seconds** during initial navigation.
- **Rationale:** Quick loading enhances user engagement and allows faster orientation.

4.1.1.3 PR3: UI Element Rendering Time

- **Requirement:** The system shall render each navigation UI element within **1 second** of entering a new area.
- **Rationale:** Prompt feedback improves usability and reduces confusion.

4.1.1.4 PR4: Navigation Arrow Update Time

- **Requirement:** The system shall update virtual navigation arrows within **0.5 seconds** of user movement through the scene.
- **Rationale:** Frequent updates help maintain path accuracy and immersion.

4.1.1.5 PR5: Label Placement Time

- **Requirement:** The system shall display door labels (e.g., “CS Lab”, “Finance Office”) within **2 seconds** when entering a scene or floor.
- **Rationale:** Timely label display ensures clarity in identifying spaces.

4.1.1.6 PR6: Route Calculation Time

- **Requirement:** The system shall calculate the navigation path within **2 seconds** after a destination is selected.
- **Rationale:** Fast routing supports quick decisions and smooth navigation.

4.1.1.7 PR7: Route Display Time

- **Requirement:** The system shall render and display the navigation route within **1 second** after calculation.
- **Rationale:** Immediate visual response increases user confidence.

4.1.1.8 PR8: Scene Location Update Frequency

- **Requirement:** The system shall update the user's virtual position within the scene every **1 second** while navigating.
- **Rationale:** Frequent updates maintain location accuracy within the virtual space.

4.1.1.9 PR9: VR Frame Rate

- **Requirement:** The system shall maintain a minimum frame rate of **30 FPS** during VR interaction.
- **Rationale:** A stable frame rate ensures smooth and immersive navigation.

4.1.1.10 PR10: Application Launch Time

- **Requirement:** The system shall fully launch and become interactive within **5 seconds** of opening.
- **Rationale:** Fast launch improves usability and perceived quality.

4.1.1.11 PR11: Path Recalculation Time

- **Requirement:** The system shall recalculate a new navigation route within **3 seconds** if the user deviates from the original path.
- **Rationale:** Quick path correction prevents user disorientation.

4.1.1.12 PR12: Touch Response Time

- **Requirement:** The system shall respond to touch or click input within **0.2 seconds**.
- **Rationale:** Fast feedback improves interaction and user satisfaction.

4.1.1.13 PR13: Object Visibility Accuracy

- **Requirement:** The system shall ensure that all location markers and directional icons remain within **±5% of their expected positions** inside the VR scene.
- **Rationale:** Proper placement ensures clarity and accuracy in navigation.

4.1.1.14 PR14: Database Scalability

- **Requirement:** The system shall use a scalable backend to manage faculty/classroom/POI data.

- **Rationale:** Scalability supports future growth without performance loss.

4.1.1.15 PR15: Local Data Loading Optimization

- **Requirement:** The system shall preload all required data (models, labels, paths) for each floor/building during Startup or first-time navigation.
- **Rationale:** Local loading ensures uninterrupted VR experience.

4.1.1.16 PR16: Automatic Data Backup Frequency

- **Requirement:** The system shall perform automatic backups of the backend database every **24 hours**.
- **Rationale:** Frequent backups reduce risk of data loss.

4.1.1.17 PR17: Data Recovery Time

- **Requirement:** The system shall allow complete data recovery within **24 hours** after a failure.
- **Rationale:** Quick recovery minimizes system downtime and disruption.

4.2 Safety and Security Requirements

- **S1:** The system should display a mandatory information dialog before entering VR navigation, informing users about virtual controls and interaction safety.
- **S2:** The system should display a brief on-screen message in the VR environment advising users to stay aware of their device orientation and environment while navigating.
- **S3:** The system should use HTTPS protocol with TLS encryption for all data transmitted between the web browser and backend services.
- **S4:** The system should avoid collecting any personally identifiable information (PII) by using anonymous sessions and avoiding user tracking.
- **S5:** The system should not store any PII in backend databases or client-side storage.
- **S6:** The system should store user bookmarks locally using browser-based storage such as localStorage or IndexedDB.
- **S7:** The system should validate all user inputs using input masks, data type checks, and pattern rules both client-side and server-side.
- **S8:** The system should sanitize all inputs using standard libraries to prevent code injection or cross-site scripting (XSS).
- **S9:** The system should verify the integrity of map data and 3D models by validating digital signatures or hash values before rendering.
- **S10:** The system should comply with Air University's IT security policies for handling files, services, and any device-level operations.
- **S11:** The system should follow relevant data protection regulations (e.g., GDPR, PECA) in terms of how data is accessed, processed, or stored.

- **S12:** The system should request permission to access motion/orientation sensors (e.g., gyroscope) before using device-based rotation features.
- **S13:** The system should include a “Return to Menu” button at all times during VR navigation for safe and immediate scene exit.
- **S14:** The system should detect corrupted 3D assets or map data before loading and alert the user appropriately.
- **S15:** The system should be tested against OWASP Web Security Guidelines, using tools suited to single-page applications and browser-based VR environments.
- **S16:** The system should store user feedback securely in a cloud-based database with backend-controlled access.
- **S17:** The system should implement role-based access control (RBAC) for backend features such as adding new building data, uploading scenes, or modifying labels.
- **S18:** The system should not transmit any user data to third parties without explicit user consent.
- **S19:** The system should protect critical data using regular automatic backups and safe write operations during updates.
- **S20**: The system should store user feedback in a secure cloud-based database with access controlled via backend authentication.

4.3 Software Quality Attributes

- **Q1. Adaptability:**
    - The system should support easy updates to campus 3D models and virtual maps without requiring major software redesign.
    - The system should support adding or modifying points of interest remotely within 24 hours.
- **Q2. Availability:**
    - The system should be available 99.5% of the time during university operating hours (7 AM to 10 PM).
    - The system should be scheduled for maintenance in advance and should not exceed two hours per week.
- **Q3. Correctness:**
    - The system should provide navigation directions with at least 95% accuracy in virtual path placement.
    - The system should provide route guidance with at least 95% accuracy in spatial navigation within the VR environment.

- **Q4. Flexibility:**
    - The system should support integration with additional modules or features (e.g., announcements, schedules) without impacting core VR navigation functionality.
- **Q5. Interoperability:**
    - The system should be compatible with Android 10+ devices using mobile Chrome or Firefox.
    - The system should be compatible with iOS 13 using Safari or supported browsers.
- **Q6. Maintainability:**
    - The system should have a modular codebase to enable faster fixes and updates.
    - The system should allow developers to implement fixes or new features within an average of 3 business days.
- **Q7. Portability:**
    - The system should support deployment on both Android and iOS platforms with minimal changes.
- **Q8. Reliability:**
    - The system should recover from 3D model loading failures (e.g., missing scene data) within 5 seconds without crashing.
    - The system should handle sensor read errors (e.g., orientation input issues) gracefully without disrupting the user session.
- **Q9. Reusability:**
    - The system should have core VR navigation components (e.g., routing, floor switcher, camera control) designed as reusable modules.
- **Q10. Robustness:**
    - The system should handle unexpected inputs (e.g., invalid room ID, null selections) gracefully without crashing.
    - The system should provide meaningful error messages to users when scene data is unavailable or corrupted.
- **Q11. Testability:**
    - The system should include automated tests that cover at least 80% of the core functionality (navigation, interaction, search, bookmarks).
- **Q12. Usability:**
    - The system should allow users to start VR navigation in no more than 3 steps from the main interface.

- The system should achieve at least 85% user satisfaction for ease of use during usability testing sessions.

5 Other Requirements

5.1 Design and Implementation Constraints

- The system shall be designed using modular components for easier updates and maintenance.
- The system shall validate data entries before committing them to the database (e.g., POI names, building data).
- The system shall implement separation of concerns (e.g., VR navigation module, admin interface, user interaction logic).
- The system shall comply with all AUMC policies, including IT usage rules and data ethics standards.
- The system shall store route, POI, and feedback data in a centralized cloud database.
- The system shall run on both desktop and mobile devices.
- The system shall allow administrators to manually trigger data backups through a secure backend interface.
- The system shall be developed primarily in English for its initial release.

5.2 Domain Requirements

- The system should be adaptable for use by other campuses with minimal reconfiguration (e.g., new floor plans or POIs).
- The system should comply with GDPR, PECA, and other applicable data protection regulations.
- The system should support future localization and internationalization features.

5.3 Inverse Requirements

- The system shall not require camera or GPS access, as VR navigation does not depend on real-world inputs.
- The system shall not allow unauthorized users to modify 3D campus scenes, labels, or route data.
- The system shall not store user bookmarks on cloud servers, all bookmarks shall be saved in local browser storage only.
- The system shall not use AR.js, as it is for marker-based AR and not relevant to VR.

5.4 Compatibility Requirements

- The system shall be hosted on a scalable cloud platform (e.g., Firebase, Vercel, AWS).
- The system shall support the Chrome web browser.
- The system shall support the Firefox web browser.
- The system shall support the Edge web browser.
- The system shall support the Safari web browser.
- The system shall be fully responsive to different screen sizes.
- The system shall be fully responsive to different screen orientations.
- The system shall be optimized for mobile browsers.
- The system shall be optimized for desktop browsers.
- The system shall use WebXR, A-Frame, or similar libraries to support browser-based VR functionality.

Appendix A – Data Dictionary

Name	Type	Description	Possible States / Values	Related Operations
currentLocation	State Variable	Stores the user's current location in the VR environment	Manual entry	getUserLocation(), updatePath()
destination	Input / State Variable	Target location selected by the user	Classroom, Office, Parking, etc.	startNavigation(), calculateDistance()
motionSensors	Input Source	Sensor readings for orientation and device movement	Enabled, Disabled	rotateView(), autoAdjustView()
searchQuery	Input	The location keyword typed by the user	Text String	filterSuggestions(), showSearchResults()
searchResults	Output	List of matching campus locations	List of POIs	displaySearchResults()
navigationPath	State Variable	Stores the current path for VR navigation	List of coordinates or waypoints	drawPath(), recalculatePath()
zoomLevel	Input	Zoom level applied to map or VR view	1x to 5x	zoomIn(), zoomOut()
rotationAngle	State Variable	Degree of rotation applied to VR view or map	0°–360°	rotateView(), resetRotation()
bookmarked Locations	State Variable	List of user-saved POIs	List of location IDs	addBookmark(), removeBookmark(), getBookmarks()
deviceType	Constant	Type of device running the system	Mobile, Desktop, Table	detectDevice(), adjustUI()
droneView Enabled	State Variable	Whether drone view is toggled on	True / False	loadDroneMap(), exitDroneMode()
viewMode	State Variable	Current active mode of the	Drone, Map, 360°, Search,	switchMode()

		system	Navigation	
360ViewData	Output	Panoramic scene of a location	Image set or video	render360View(), reset360()
feedbackInput	Input	Feedback message entered by user	Text string	submitFeedback(), storeFeedback()
databaseStatus	State Variable	Indicates backend database connectivity	Connected / Offline	fetchData(), handleDBError()
SceneAssets	Output	Loaded 3D models and POI labels	Object list or scene graph	loadScene(), refreshAssets(
mapLabels	Output	Textual identifiers for buildings and rooms	List of labels	renderLabels(), toggleLabels()

Appendix B - Group Log

Date	Members Involved	Activity	Summary of Outcome
1 Apr – 10 Apr	Abdul Rafay, Abdul Rehman	Initial Discussion	Decided on VR navigation as core feature; divided modules
11 Apr – 21 Apr	Abdul Rafay	Requirement Gathering	Collected basic functional needs through observation and assumption
23 Apr – 04 May	Abdul Rehman	Research	Studied VR libraries (WebXR) for browser-based support
20 Apr – 26 Apr	Abdul Rafay, Abdul Rehman	UI Brainstorming	Sketched UI layout for VR and drone view modes
7 May	Abdul Rafay	SRS Drafting	Completed first draft of SRS document using IEEE template
8 May – 15 May	Abdul Rehman	Functional Expansion	Created detailed breakdown of Indoor, Outdoor, and Bookmark modules
17 May – 22 May	Abdul Rehman	Quality Attributes	Integrated software quality attributes like usability, reliability
23 May	Abdul Rafay	Use Case Model	Added 5 detailed use case models to the document
25 May	Both Members	Final Submission	Final version of SRS submitted to course portal