**Software Requirements**

**Specification**

**for**

**<Project>**

**Version 1.0 approved**

**Prepared by <author>**

**<organization>**

**<date created>**

***Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.***

# Table of Contents

[Table of Contents ii](#_TOC_250033)

[Revision History ii](#_TOC_250032)

1. [Introduction 1](#_TOC_250031)
   1. [Purpose 1](#_TOC_250030)
   2. [Document Conventions 1](#_TOC_250029)
   3. [Intended Audience and Reading Suggestions 1](#_TOC_250028)
   4. [Product Scope 1](#_TOC_250027)
   5. [References 1](#_TOC_250026)
2. [Overall Description 2](#_TOC_250025)
   1. [Product Perspective 2](#_TOC_250024)
   2. [Product Functions 2](#_TOC_250023)
   3. [User Classes and Characteristics 2](#_TOC_250022)
   4. [Operating Environment 2](#_TOC_250021)
   5. [Design and Implementation Constraints 2](#_TOC_250020)
   6. [User Documentation 2](#_TOC_250019)
   7. [Assumptions and Dependencies 3](#_TOC_250018)
3. [External Interface Requirements 3](#_TOC_250017)
   1. [User Interfaces 3](#_TOC_250016)
   2. [Hardware Interfaces 3](#_TOC_250015)
   3. [Software Interfaces 3](#_TOC_250014)
   4. [Communications Interfaces 3](#_TOC_250013)
4. [System Features 4](#_TOC_250012)
   1. [System Feature 1 4](#_TOC_250011)
   2. [System Feature 2 (and so on) 4](#_TOC_250010)
5. [Other Nonfunctional Requirements 4](#_TOC_250009)
   1. [Performance Requirements 4](#_TOC_250008)
   2. [Safety Requirements 5](#_TOC_250007)
   3. [Security Requirements 5](#_TOC_250006)
   4. [Software Quality Attributes 5](#_TOC_250005)
   5. [Business Rules 5](#_TOC_250004)
6. [Other Requirements 5](#_TOC_250003)

[Appendix A: Glossary 5](#_TOC_250002)

[Appendix B: Analysis Models 5](#_TOC_250001)

[Appendix C: To Be Determined List 6](#_TOC_250000)

# Introduction

## Purpose

The SkyTrace project is designed to develop an advanced geospatial simulation tool that facilitates interactive path simulation on a map interface, with an emphasis on real-time danger zone detection. The primary purpose of this project is to enhance geospatial analysis capabilities by allowing users to visualize and monitor paths for potential hazards during simulations. This SRS document serves as a formal agreement between the development team and stakeholders, including project sponsors, educators, and end-users. It outlines the detailed software requirements, ensuring that the final product meets the intended functionality, usability, and performance criteria.

## Product Scope

SkyTrace is a standalone desktop application designed to simulate and analyze geospatial paths in real-time. The primary objectives include:

* **Interactive Path Simulation:** Users can dynamically create and simulate paths on a map interface.
* **Real-time Danger Zone Detection:** The application will alert users when a simulated path is near or within predefined danger zones.
* **User Customization:** Users can set simulation speeds, modify paths, and customize alert settings.

**Out of Scope:**

* SkyTrace will not support live data feeds or real-time GPS tracking.
* The application will not include multi-user collaboration features.
* Cloud-based data storage or processing will not be implemented; all operations will be local to the user's machine.

## References

* IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.
* Python 3.8 Documentation, available at <https://docs.python.org/3.8/>
* TkinterMapView Documentation, available at <https://tkintermapview.readthedocs.io/>
* Shapely Documentation, available at <https://shapely.readthedocs.io/en/stable/>
* Pygame Documentation, available at https://www.pygame.org/docs/

# Overall Description

## Product Perspective

SkyTrace is designed as an independent application but fits within the broader context of educational and research tools used for geospatial analysis. It does not rely on external servers or cloud services, ensuring all processing is performed locally on the user’s machine. The application leverages existing Python libraries such as Tkinter for GUI management, TkinterMapView for map visualization, Shapely for geometric calculations, and Pygame for audio notifications. While SkyTrace can operate independently, it could be extended or integrated with other geospatial tools in the future, such as GIS platforms for more advanced data analysis.

## Product Functions

**Function 1: Dynamic Path Creation**

* **Inputs:** User-defined waypoints (latitude and longitude) entered by right-clicking on the map interface.
* **Processing:** The system calculates the path between waypoints and continuously updates the path as new points are added.
* **Outputs:** A dynamically drawn path displayed on the map, updated in real-time as the simulation progresses.
* **Constraints:** The system must handle up to 100 waypoints without significant performance degradation.

**Function 2: Real-time Danger Zone Detection**

* **Inputs:** Predefined danger zones stored in the system's database or loaded from external files.
* **Processing:** The system checks the proximity of the simulated path to the danger zones using geometric calculations (e.g., distance, intersection).
* **Outputs:** Alerts displayed on the screen and played through audio when the path approaches or enters a danger zone. The system specifies whether the danger zone is on the right or left of the path.
* **Constraints:** The system must detect and alert for danger zones within 100 milliseconds of approaching.

## User Classes and Characteristics

* **Aviation Students**

**Description:** Individuals studying aviation, aeronautics, or related fields who require hands-on experience with geospatial simulation tools.

**Characteristics:**

Basic to intermediate knowledge of geospatial concepts.

Familiar with aviation principles, including navigation and flight path planning.

Typically use the application for educational purposes, such as simulating flight paths and understanding the impact of danger zones.

Require an intuitive interface with clear instructions and visual aids.

* **Aviation Instructors**

**Description:** Educators responsible for teaching aviation-related courses and demonstrating the practical applications of geospatial tools in the field.

**Characteristics:**

Advanced understanding of aviation and geospatial principles.

Use the application to create and demonstrate scenarios for teaching purposes.

Require the ability to customize simulations and create specific scenarios for instructional purposes.

Need reliable performance to ensure smooth demonstrations during lessons.

* **Geospatial Analysts**

**Description:** Professionals who analyze geospatial data for various applications, including aviation safety and flight planning.

**Characteristics:**

Expert knowledge of geospatial data, analysis techniques, and tools.

Use the application to test and validate geospatial models, particularly in identifying danger zones.

Require advanced features, such as the ability to import/export data, perform complex calculations, and customize simulation parameters.

Demand high accuracy and reliability in simulations.

* **Software Developers**

**Description:** Developers tasked with extending, maintaining, or integrating the SkyTrace application with other systems.

**Characteristics:**

Proficient in programming, particularly in Python, and familiar with Tkinter and other libraries used in the application.

Interested in the application's architecture, API, and extensibility.

Require access to detailed documentation, code samples, and the ability to debug and modify the source code.

Expect modular design and clean code to facilitate easy integration and enhancement.

* **Aviation Safety Officers**

**Description**: Personnel responsible for ensuring the safety of flight operations by analyzing potential risks and hazards.

**Characteristics:**

Extensive experience in aviation safety protocols and hazard identification.

Utilize the application to simulate flight paths and assess the risk of danger zones.

Require real-time alerts and accurate detection of hazards.

Depend on the application's reliability and performance to make informed decisions about flight safety.

Each user class is expected to interact with the SkyTrace application differently, and the system has been designed to accommodate their specific needs. The application balances usability for novice users while providing advanced features for experts.

## Operating Environment

**Hardware Environment**

**Processor:** Intel Core i5 or equivalent; i7 recommended for complex simulations.

**Memory (RAM):** 8 GB minimum; 16 GB recommended.

**Storage:** 500 MB for installation; additional space for data.

**Graphics:** Integrated graphics or basic GPU recommended.

**Software Environment**

**Operating System:** Windows 10+, macOS 10.15+, Linux (Ubuntu 20.04+).

**Python Version:** Python 3.8 or later.

**Dependencies:** Tkinter, TkinterMapView, Shapely, Pygame, Pillow.

**External Dependencies**

**Internet Connectivity:** Required for installing libraries and accessing online maps.

**GIS Data Sources:** Local or external, depending on user needs.

**Security and Privacy**

**Local Execution:** Runs locally to ensure data security.

**Data Privacy:** No external data transmission, ensuring user privacy.

## Design and Implementation Constraints

**Tkinter Limitations:** The user interface is built using Tkinter, which has limitations in rendering complex graphics and handling high-frequency events, potentially affecting real-time simulation performance.

**TkinterMapView:** The project relies on TkinterMapView for map rendering and path simulation, which may impose constraints on map customization, zoom levels, and coordinate precision.

**Single-Threaded Execution:** Python’s Global Interpreter Lock (GIL) limits the use of multi-threading for real-time processing, which may necessitate careful management of tasks to avoid performance bottlenecks.

**Cross-Platform Compatibility:** Ensuring consistent behavior across different operating systems (Windows, macOS, Linux) requires handling OS-specific issues such as file paths, window management, and library dependencies.

**Dependency Management:** The project depends on several third-party libraries (e.g., Shapely, Pygame, Pillow), requiring careful version control and compatibility testing to avoid conflicts and ensure smooth installation.

**GIS Data Accuracy:** The accuracy of path simulations and danger zone detection depends on the precision of GIS data. Any inaccuracies in the data may lead to incorrect simulation outcomes.

**Resource Constraints:** The application must operate efficiently on systems with limited resources (e.g., lower RAM or CPU performance), requiring optimization of memory usage and processing efficiency.

**Security Considerations:** While the application operates locally, ensuring the security of user data and the integrity of the simulation environment is critical, particularly if sensitive geospatial data is used.

**User Interface Design:** The need to maintain an intuitive and responsive user interface may constrain the complexity of visual elements and interactive features, impacting the overall user experience.

**Data Handling:** Handling large geospatial datasets or real-time data streams could impose constraints on memory and processing, necessitating optimization strategies such as data caching or selective loading.

## User Documentation

* **Introduction**

Overview of the SkyTrace application and its purpose.

Brief description of its key features and functionalities.

* **Getting Started**

Installation instructions, including system requirements.

Initial setup and configuration.

Starting the application and logging in (if applicable).

* **User Interface Overview**

Description of the main components of the interface (e.g., map view, buttons, status bar).

Explanation of each component's function and how to interact with it.

* **Features and Functions**

**Adding Points to the Path:**

How to right-click on the map to add points.

**Simulating Path Movement:**

Using the 'Start Simulation' button.

Adjusting the speed of simulation.

**Managing Paths:**

Using the 'Stop Simulation' button to halt the simulation.

Using 'Clear Path' to remove all points.

Using 'Undo Path' to remove the last added point.

**Loading Coordinates:**

Instructions for using the 'Load Lat Lon' button to import path data.

**Danger Zone Alerts:**

How the application notifies users when paths approach danger zones.

* **Common Tasks**

How to set up a new path simulation.

How to view and interpret danger zone alerts.

How to save and load simulations (if applicable).

* **Troubleshooting**

Common issues and solutions.

How to contact support for further assistance.

Help and Support

* **Online Help**

Accessing online help resources or documentation.

Links to FAQs, tutorials, and user forums (if available).

* **Contact Information**

Support contact details for technical issues or feedback.

Hours of operation and response times.

* **Glossary**

Terms and Definitions

Explanation of specific terms used within the application (e.g., “danger zone,” “simulation speed”).

Examples and Use Cases

## Assumptions and Dependencies

* Users will have basic proficiency in using desktop applications and understanding geographical concepts.
* The user's machine has adequate processing power and memory to handle real-time simulations.
* The necessary Python libraries (Tkinter, TkinterMapView, Shapely, Pygame) are available and compatible with the user's system.

**Dependencies:**

* The project depends on third-party libraries such as Tkinter, TkinterMapView, Shapely, and Pygame. Any changes or updates to these libraries could affect the system’s functionality.
* The availability of map data and danger zone definitions, which could be provided by external files or a pre-loaded database.

# External Interface Requirements

## User Interfaces

* The UI will be designed using Tkinter, featuring a main window with buttons for simulation control (Start, Stop, Clear Path, Undo Path, Load Lat/Lon).
* A map area will be integrated using TkinterMapView, where users can right-click to add waypoints and visualize the path.
* Pop-up windows or dialog boxes will be used to display alerts, such as when a danger zone is detected.

## Hardware Interfaces

* SkyTrace is expected to run on standard desktop hardware with no special requirements.
* The application should function properly with typical input devices, such as a mouse and keyboard.

## Software Interfaces

* SkyTrace will integrate with the TkinterMapView module to provide map visualization capabilities.
* Shapely will be used for geometric operations, such as detecting when a path is near or intersects with a danger zone.
* Pygame will handle the playback of audio alerts.

## Communications Interfaces

* The application will not require any external communication or internet connectivity, ensuring that all data is processed locally.

# System Features

**4.1 Feature 1: Dynamic Path Drawing**

**Description:** The system allows users to create a path by clicking on the map interface. As the simulation progresses, the path is dynamically drawn in green, updating in real-time.

**Functional Requirements:**

* + The map interface must register right-clicks as waypoints.
  + The system should automatically draw lines between waypoints, forming a continuous path.
  + The path color should change to green during simulation to indicate active tracking.

**4.2 Feature 2: Danger Zone Detection**

**Description:** The application monitors the path for proximity to predefined danger zones, alerting the user when a zone is approached.

**Functional Requirements:**

* + The system must load and store danger zones, either from a local database or external files.
  + Geometric calculations should determine if a path is within a defined distance from a danger zone.
  + Visual (on-screen alerts) and auditory (sound alerts using Pygame) warnings should trigger when the path is near a danger zone.

**4.3 Feature 3: Simulation Control**

**Description:** Users can control the simulation with options to start, stop, clear, and undo the path.

**Functional Requirements:**

* + Buttons for 'Start Simulation,' 'Stop Simulation,' 'Clear Path,' and 'Undo Path' must be implemented.
  + The system must handle the starting and stopping of the simulation without errors.
  + Clearing the path should remove all visual traces of the current path, resetting the simulation.

# Other Nonfunctional Requirements

## Performance Requirements

* The application must update the map display in real-time, with no more than a 100-millisecond delay between user input and map updates.
* Danger zone detection calculations should be performed within 50 milliseconds to ensure timely alerts.
* The application must handle up to 100 waypoints and 50 danger zones without significant performance degradation.

## Security Requirements

* SkyTrace will not transmit or store user data externally, ensuring all data remains on the local machine.
* The application must include basic data protection mechanisms, such as encryption for any saved path or danger zone data.
* User access to saved sessions should be restricted by local operating system permissions, ensuring that only authorized users can modify or delete data.

## Safety Requirements

* **Danger Zone Detection:** Accurately detect and alert users when the path is near a danger zone, indicating whether the danger is on the right or left side.
* **User Alerts:** Provide clear visual and auditory alerts for any detected hazards.
* **Error Handling:** Safely manage errors in critical functions like path calculation and danger zone detection, pausing the simulation if necessary.
* **Simulation Safety:** Prevent unauthorized or unintended changes to the path or speed settings during the simulation.
* **Data Accuracy:** Ensure all simulation data (maps, coordinates, danger zones) is accurate and reliable.
* **Emergency Stop:** The 'Stop Simulation' button should immediately halt the simulation in case of emergencies.

## Usability Requirements

* The user interface should be intuitive and easy to navigate, with clear labels and tooltips for all controls.
* A help menu or user guide should be available within the application, providing instructions on how to use each feature.
* The application should support keyboard shortcuts for common actions, such as starting/stopping the simulation and clearing the path.

## Reliability Requirements

* The application must maintain a 99% uptime during normal operation, with minimal crashes or unresponsive behavior.
* Error messages should be clear and provide guidance on how to resolve the issue.
* The system must automatically save the current path and danger zones, allowing users to recover their session after an unexpected shutdown.

# Other Requirements

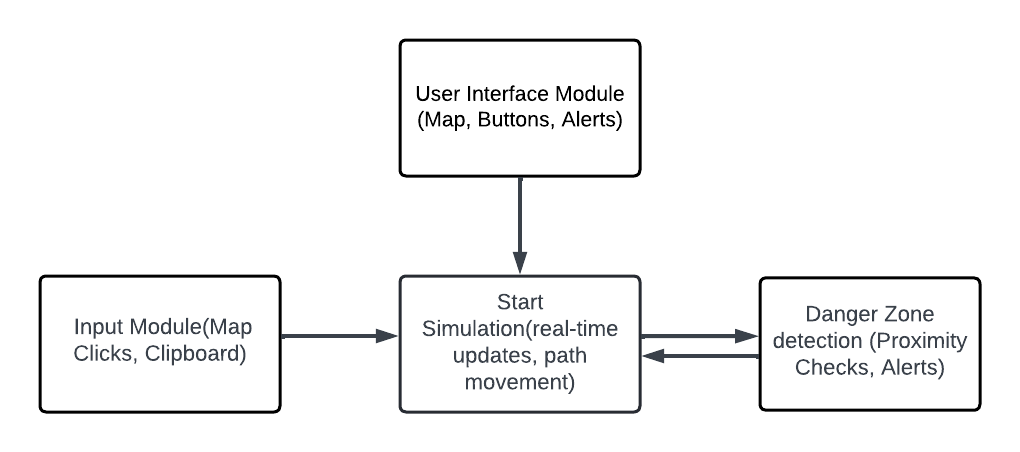
* **Compatibility:** The application should be compatible with common operating systems (e.g., Windows, macOS, Linux) and run smoothly on systems meeting the specified requirements.
* **Scalability:** The system should handle increasing amounts of data, such as more complex paths or larger maps, without performance degradation.
* **Maintainability:** The application’s codebase should be well-documented and organized, making it easy to update and maintain.

# Appendix A: Glossary

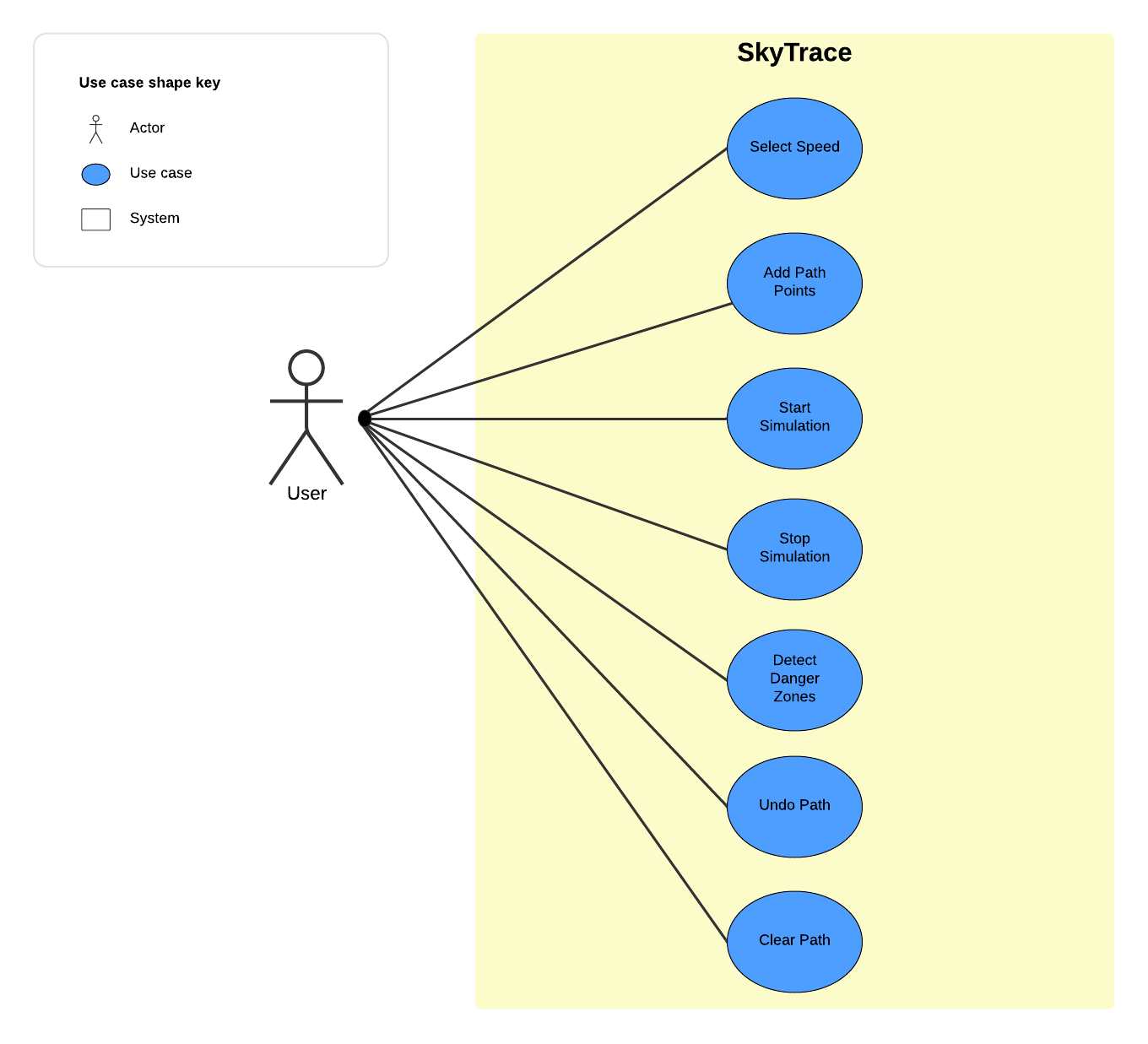
* **Danger Zone:** A specific area on the map identified as hazardous. The application detects when a path approaches these zones and alerts the user.
* **Simulation Speed:** The rate at which the path simulation progresses. Users can adjust this speed to see how the path would unfold in real-time or faster.
* **Path Point:** A coordinate on the map added by the user to define the path. Multiple path points form the complete route that the simulation will follow.
* **SkyTrace:** The name of the application used for geospatial path simulation, which includes features such as path creation, simulation, and danger zone detection.
* **Tkinter:** A standard Python interface to the Tk GUI toolkit. It is used in SkyTrace to create the graphical user interface (GUI).
* **TkinterMapView:** A module within Tkinter that provides interactive map functionality, allowing users to add points, draw paths, and interact with map data.
* **Shapely:** A Python library for manipulation and analysis of planar geometric objects, used in SkyTrace for geometric calculations, such as detecting danger zones.
* **Pygame:** A Python library used for writing video games. In SkyTrace, it is utilized for playing auditory alerts when a danger zone is detected.
* **PIL (Python Imaging Library**): A library used in Python for opening, manipulating, and saving many different image file formats. SkyTrace uses it for processing images related to the map and paths.
* **Simulation:** A process in SkyTrace where the created path is played out on the map, showing the movement from the starting point to the destination in real-time or at a set speed.
* **Lat Lon:** Abbreviation for Latitude and Longitude, which are geographical coordinates used to specify locations on the Earth's surface. In SkyTrace, users can load paths using Lat Lon data.
* **User Interface (UI):** The part of the application through which the user interacts with the system, including components like buttons, map views, and status bars.

# Appendix B: Analysis Models

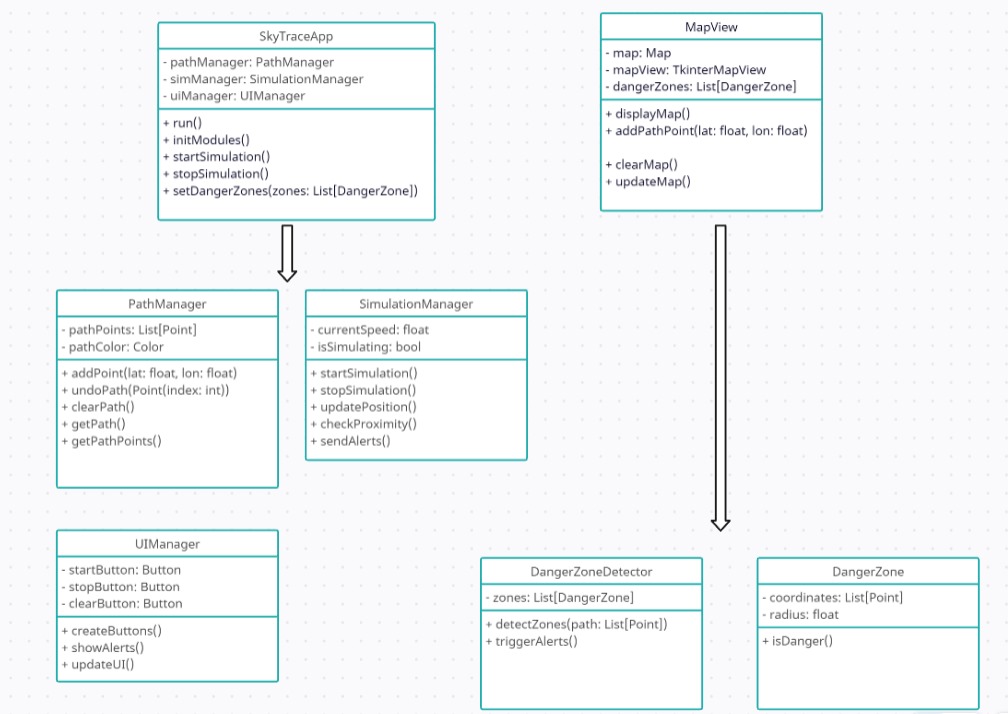
**Block Diagram:**



**Use Case Diagram:**



**Class Diagram:**



# Appendix C: To Be Determined List

* **User Login and Authentication:** Determine whether the SkyTrace application will require a user login system for access control and data security. Define the authentication methods if implemented.
* **Multi-User Collaboration:** Consider the feasibility of enabling multiple users to collaborate on the same path simulation in real-time, with shared access and editing capabilities.
* **Mobile Application:** Assess the demand and potential development of a mobile version of SkyTrace, ensuring compatibility with both Android and iOS platforms.