COS301

WILDLIFE DRONE FLIGHT PLANS SYSTEMS REQUIREMENTS SPECIFICATION

DR BAM

Matthew Evans	16262949
Andreas Louw	15048366
Bryan Janse van Vuuren	16217498
Deane Roos	17057966
Reinhardt Eiselen	14043302

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to present an overview the proposed drone flight plan system. To elicitate its requirements, along with more information on the system parameters and goals. To describe the target audience, user interface, and high-level hardware and software requirements for the proposed system. This document is intended for EPI-USE, the stakeholders, and the developers who will implement the system.

1.2 Vision

Our vision is to create a real world applicable system that in the future will aid in the conservation of rhinos and elephants.

1.3 Objectives

- To create a prototype of this system that creates randomized optimal fight plans for drones.
- To increase the effectiveness of drone patrols by providing them with a more extensive overview of flight plans and map activities.
- Empower rangers with information about poaching areas within reserves and creating routes to patrol based on this information.

1.4 Business needs

To allow for more optimal and accurate route creation so that the drones can cover more key areas to help prevent the poaching of rhinos and elephants. While randomizing routes to prevent the possibility of patterns being recognized by poachers.

1.5 Scope

The proposed system is a flight plan system that will generate random optimal flight plans for drone pilots and patrol routes for rangers based on poaching hotspots. Pilots and rangers will be notified of the routes and changes to them.
Hotspots will be generated taking into account past incidents, geographical information and animal tracking data.
Flight plans will be generated by taking in the drones capabilities and the hotspots at the time as well as historical data and current incidents reported by rangers and pilots. Rangers and pilots will be able to indicate points of interest on the map.

2 Domain Model

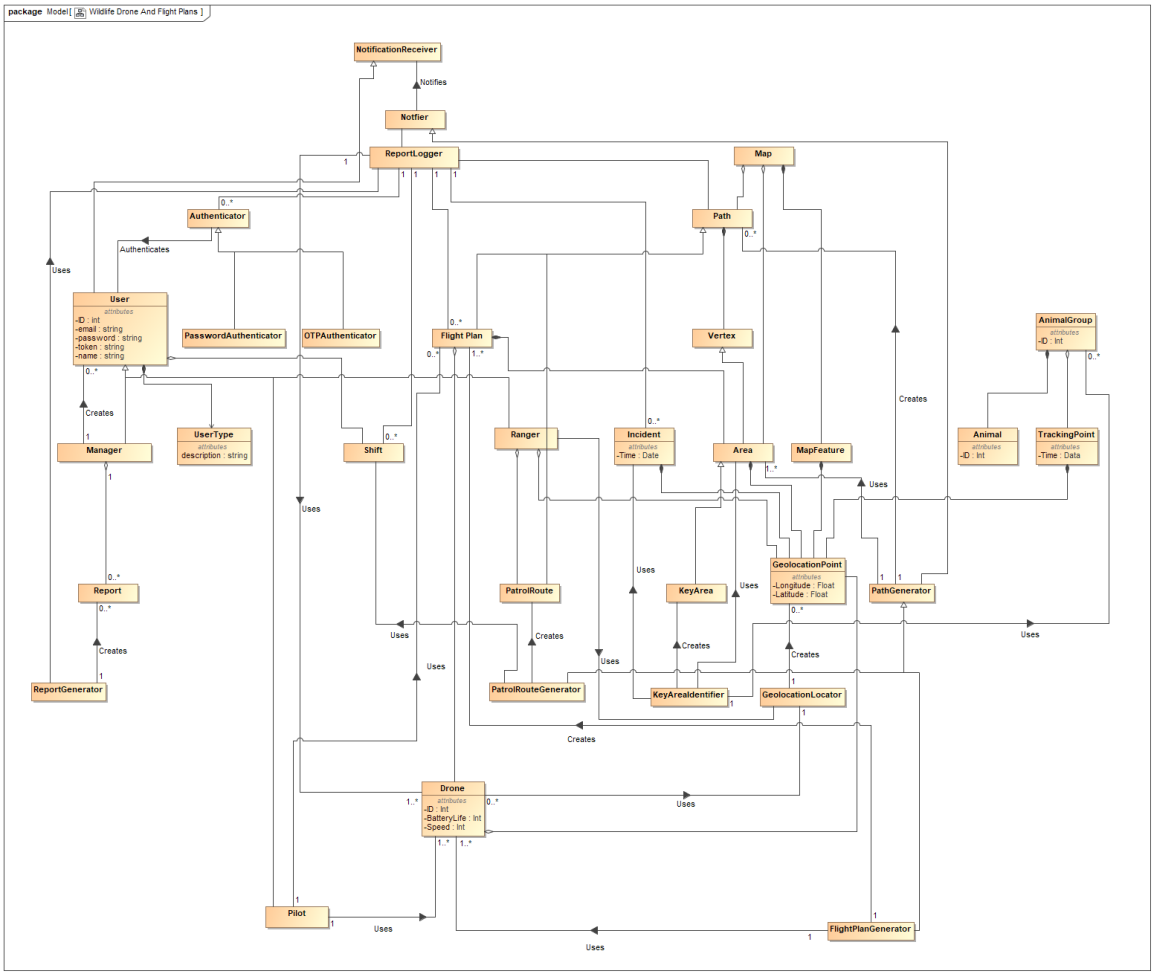

Figure 1: Back-End System.

3 User Characteristics

3.1 Drone Pilot

Will be piloting the physical drone. Will interact with the system to determine flight plan and routes that need to be traveled within a shift. These routes will be restricted to the game reserve's airspace. A drone pilot will also be able to use to use the system to add new hot spots or to report incidents.

3.2 Patrol Rangers

Will interact with the system to determine the patrol route to follow within a shift. These routes will be restricted to reserve roads. A ranger will also be able to use the system to report incidents and to add hot spots.

3.3 Administrators

Will be maintaining and updating the system as necessary. Will also have separate view from ranger and pilots to run diagnostics on data gathered.

4 Functional Requirements

4.1 Use Cases

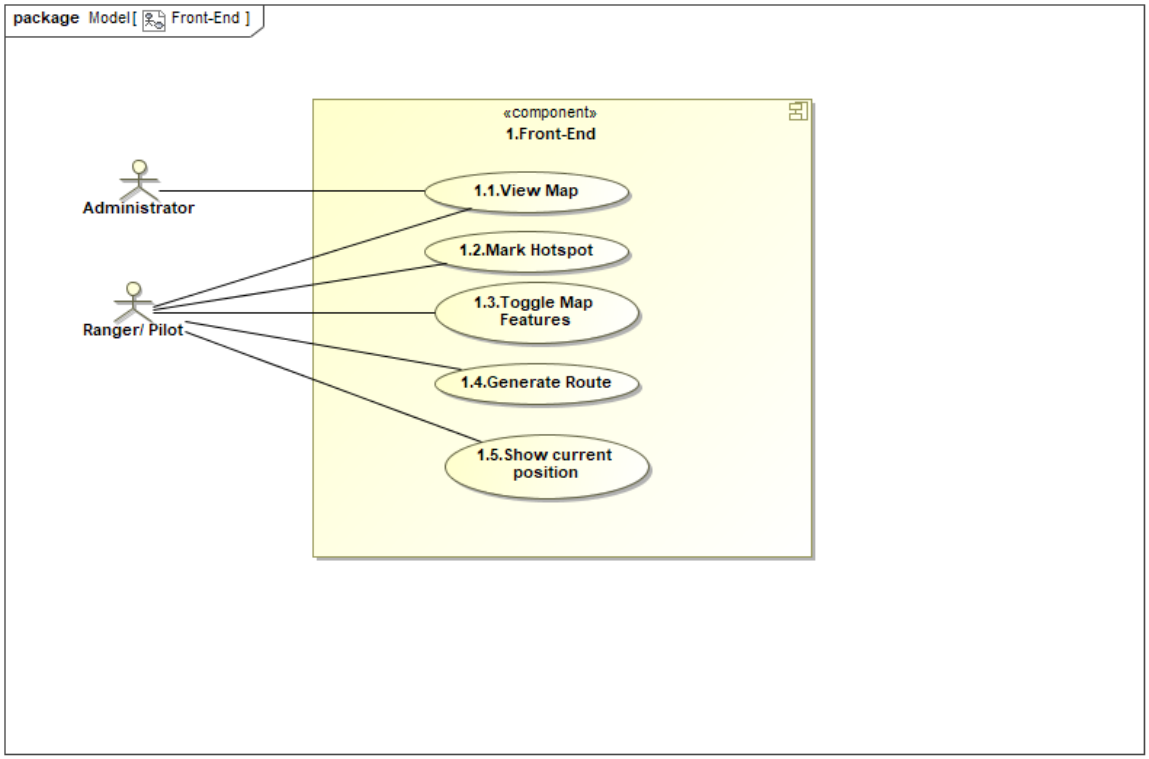

Figure 2: Front-End System.

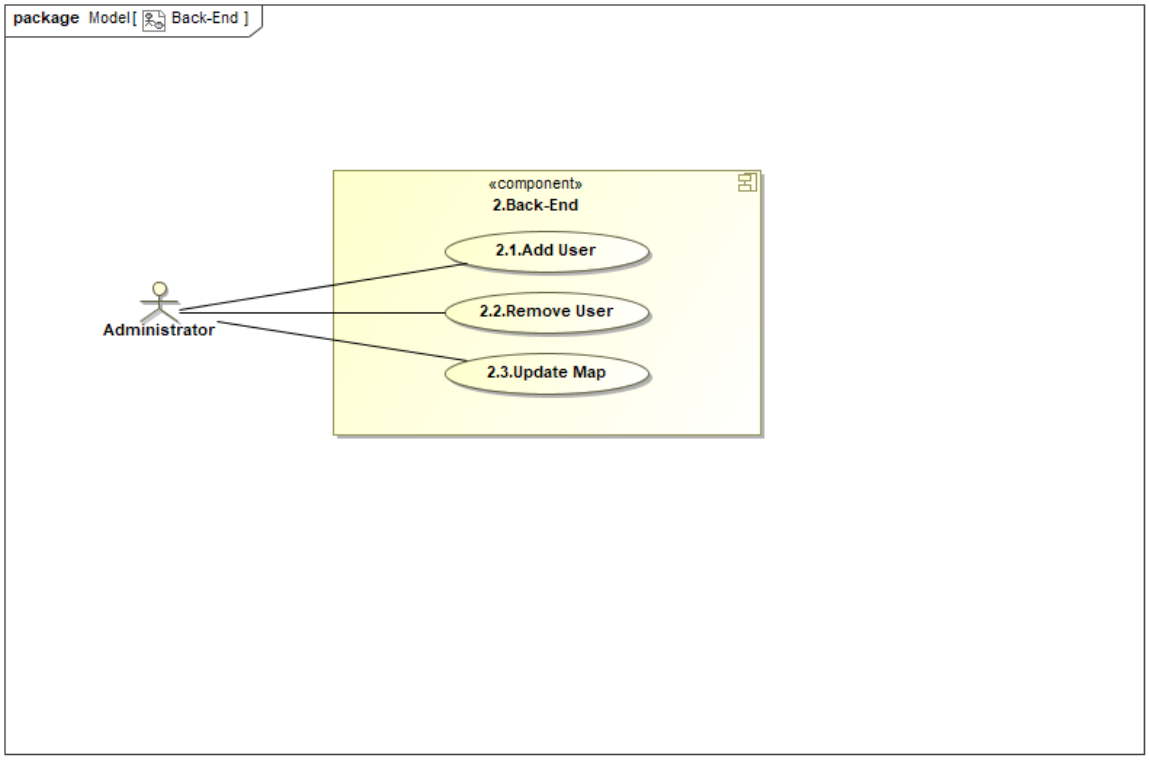

Figure 3: Back-End System.

4.2 Requirements

R1: The system will provide a map for the user.

- **R1.1:** The system will provide an overlay of routes on the map
  - **R1.1.1:** The system will provide an overlay of pilot routes on the map as generated by **R2.1**.
  - **R1.1.2:** The system will provide an overlay of ranger routes on the map as generated by **R2.2**.
- **R1.2:** The system will provide points of interest on the map
  - **R1.2.1:** The system will show user marked points on the map as used in **R4**.
  - **R1.2.2:** The system will allow ranger locations to be shown on the map as gathered from **R3.1**.
  - **R1.2.3:** The system will allow drone locations to be shown on the map as gathered from **R3.2.1**.
- **R1.3:** The system will provide animal hot-spots on the map **R7.1**.

R2: The system will generate random optimal routes.

- **R2.1:** The system will generate a flight plan for drones.
- **R2.2:** The system will generate a possible patrol route for rangers.

R3: The system will track user devices

- **R3.1:** The system will track ranger device location.
- **R3.2:** The system will track drones.
  - **R3.2.1:** The system will track the drone location.
  - **R3.2.2:** The system will track the drone systems, such as battery life.

R4: The system will allow for points of interest to be marked on the map

- **R4.1:** The system will allow pilots to mark points of interest.
- **R4.2:** The system will allow rangers to mark points of interest.

R5: The system will provide notifications for rangers and drone pilots.

- **R5.1:** The system will notify drone pilots when a ranger added a point of interest.
- **R5.2:** The system will notify rangers when a drone pilot added a point of interest.
- **R5.3:** The system will notify pilots when a new flight plan has been generated
- **R5.4:** The system will notify rangers when a new patrol route has been generated

R6: The system will store historic data

- **R6.1:** The system will store previously used flight routes
- **R6.2:** The system will store previously used patrol routes
- **R6.3:** The system will store animal migration data

R7: The system will handle historic data

- **R7.1:** The system will create heat-maps of animal locations based on historical data from **R6.4** and live data(If available).
- **R7.2:** The system will show previous flight plans
- **R7.3:** The system will show previous patrol routes
- **R7.4:** The system will predict migration patterns of animals based on **R6.4**

R8: The system will provide user authentication to use the system

- **R8.1:** The system will allow users to log in
  - **R8.1.1:** The system will authenticate log in details
  - **R8.1.2:** The system will inform a user of whether authentication was successful or not
  - **R8.1.3:** The system will generate a one-time pin
  - **R8.1.4:** The system will send users a one-time pin
  - **R8.1.5:** The system will authenticate a one-time pin

R8.2: The system will allow users to log out

R8.3: The system will allow administrators to manage other users

R8.3.1: The system will allow administrators to register other non-administrator users

R8.3.2: The system will allow administrators to disable other non-administrator users

R8.3.3: The system will allow administrators to change user passwords

R8.3.4: The system will hash passwords

R9: The system will provide reporting of activities.

R9.1: The system will provide logging of user log in attempts.

R9.2: The system will provide logging of points of interest done by rangers and pilots.

R9.3: The system will provide logging of routes taken by users.

R9.4: The system will provide logging of incidents.

R9.5: The system will generate reports from logs for managers.

4.3 Subsystems

4.3.1 Map subsystem

The map subsystem is responsible for displaying map data to the user. It handles loading map data, visually representing it, and handling user interaction such as moving the map and zooming.The map will allow users to toggle between different layers and or features.An example would be a ranger able to view the flight plan of the drone and vice versa. Users will also be allowed to mark hotspots/points of interest onto the map

4.3.2 Hotspot identification subsystem

The hotspot identification subsystem is responsible for identifying hotspots from map data, past poaching incidents, and animal locations.These Hotspots will be used in the generation of routes for Rangers and Drone pilots to help generate more relevant routes to patrol.

4.3.3 Animal predictions subsystem

The animal predictions subsystem is responsible for predicting the future location of animals from past tracking information and map data.The result from processing this information will be used in the generation of routes.The aim of this subsystem it to increase effectiveness of route generation

4.3.4 Route Generation Subsystem

The route generation subsystem is responsible for finding an optimal "travelling salesman" circuit through a set of given hotspots.The system will provide routes for both the Ranger patrols routes and Drone pilot routes.

4.3.5 Data modification subsystem

The data modification subsystem handles the creation, updating and deletion of park data, allowing users to change map features, drone information, add animal tracking data and add hotspots.

4.3.6 Geolocation subsystem

The Geolocation subsystem is responsible for locating the user's device using device by periodically pinging the Geolocation services of the mobile devices. The subsystem relays this information back to the server for processing.

4.3.7 Notification subsystem

The notifications subsystem is responsible for sending notifications to all designated users.

4.3.8 Authentication subsystem

The authentication subsystem is responsible for handling authentication of users, including the creation and managing of users. It is also responsible for verifying two-step authentication.

4.3.9 Reporting subsystem

The reporting subsystem is responsible for handling logging of data generated by and used by users. The subsystem will generate reports for the managers.

5 Constraints

- Users using the system are either a manager, pilot or ranger of the reserve using the system.
- The system will have a stable internet connection (such that timeouts do not occur).
- The mobile application will be run on a device with sufficient battery life.
- The mobile application will be run on a device with sufficient specifications: Android 4.4+, ios 9+, Chrome or Firefox web browser. These devices should support geolocation.
- The system will only work given sufficient reserve data.

6 Quality Requirements

6.1 Performance

QR1.1: The mobile application should start up in under 8 seconds on a mid-range device. This will be measured using a Samsung Galaxy S5.

QR1.2: The mobile application should maintain a frame rate of 30fps.This can be done using device system software or tools.

QR1.3: The system will generate a route within n seconds.This will be measured by logging the time the generate function was called and the time it ended.

6.2 Availability

QR2: We can not specify this at present.

6.3 Reliability

QR3.1: The system will back up the database hourly to a remote server. This will be monitored by viewing postgres backup logs on heroku.

QR3.2: The system will send crash reports to the client if any crashes occur. This will be accomplished by handling 404 and 501 HTTP responses client side and using timeouts

QR3.3: When the server is offline, the device will utilise caching to provide offline functionality. This will be tested using aeroplane mode on-device, or offline mode in browser.

6.4 Usability

QR4.1: The system will be streamlined and will require less than 5 taps to get to the desired view.

QR4.2: The system will be mobile-usable, tested by a score of at least 80 on Google Lighthouse.

QR4.3: The system will use Google Chrome Dev Tools Accessible Colours feature to determine whether colours are accessible. All colour combinations will be either AA or better rating.

6.5 Security

QR5.1: The system will hash and salt user passwords. This will be accomplished using SHA-256 and tested through unit testing.

QR5.2: The system will use HTTPS SSL encryption for all inter-device communication.

6.6 Scalability

QR6.1: The system will support a wide range of database management systems. This will be accomplished using ORM.

QR6.2: The System will support multiple Rangers and pilots at the same time. This will be accomplished using a connection pool to the server.

QR6.3: The system server will be capable of working on any server that can run Node.js and support the system's dependencies.

QR6.4: We do not currently have real-world reserve data, but the database should be chosen bearing the scale of this data in mind.

6.7 Flexibility

QR7.1: The system will be feasible to technological upgrades and updates. This will be accomplished using version control.

QR7.2: The system authentication module will be feasible to authentication protocol upgrades. This will be accomplished through modular system design.

6.8 Testability

QR8: All services offered by the system will be testable through unit tests and integration tests:
Unit tests will test whether the system functions and classes yield correct, predictable results.
Integration tests should test that all needed services from other subsystems are available and work together.
Acceptance testing will be used to determine whether the system meets the use cases and functional requirements described in this document.

The integration tests for each subsystem should test whether all services it needs from other subsystems are available, if the services are not available from the other subsystems, their services should be mocked.

7 Trace-ability Matrix Functional Requirements

	Map	Hotspot identification	Animal predictions	Route generation	Data modification	Geolocation	Notification	Authentication	Reporting
R1.1.1	x			x					
R1.1.2	x			x					
R1.2.1	x				x	x			
R1.2.2	x					x			
R1.2.3	x					x			
R1.3	x	x	x		x				
R2.1		x	x	x		x	x		x
R2.2		x	x	x		x	x		x
R3.1						x			x
R3.2.1						x			x
R3.2.2						x	x		x
R4.1	x				x		x		x
R4.2	x				x		x		x
R5.1							x		
R5.2							x		
R5.3							x		
R5.4							x		
R6.1				x					x
R6.2				x					x
R6.3			x		x				x
R7.1	x	x	x						x
R7.2	x			x					x
R7.3	x			x					x
R7.4			x		x				x
R8.1.1								x	x
R8.1.2							x	x	
R8.1.3								x	
R8.1.4							x	x	x
R8.1.5								x	
R8.2								x	x
R8.3.1					x			x	x
R8.3.2					x			x	x
R8.3.3					x			x	x
R8.3.4					x			x	
R9.1							x	x	x
R9.2	x				x				x
R9.3				x					x
R9.4	x	x			x				x
R9.5							x		x

Figure 4: System trace-ability matrix (functional requirements)

8 Trace-ability Matrix Quality requirements

	Map	Hotspot identification	Animal predictions	Route generation	Data modification	Geolocation	Notification	Authentication	Reporting
QR1.1	x					x		x	
QR1.2	x								
QR1.3	x			x		x	x		x
QR2									
QR3.1					x		x		x
QR3.2							x		x
QR3.3								x	x
QR4.1	x				x				
QR4.2	x				x				
QR4.3	x				x				x
QR5.1					x			x	
QR5.2	x	x	x	x	x	x	x	x	x
QR6.1					x				x
QR6.2		x	x	x	x		x	x	x
QR6.3		x	x	x	x		x	x	x
QR6.4					x				
QR7.1	x	x	x	x	x	x	x	x	x
QR7.2					x			x	
QR8	x	x	x	x	x	x	x	x	x

Figure 5: System trace-ability matrix (quality requirements)