Haiku
Human AI teaming Knowledge and Understanding for aviation safety

Deliverable N.4.4

Low fidelity prototypes

Authors:
Nareg, Minaskan Karabid (DFKI)

Abstract:
This deliverable describes the prototypes developed in HAIKU Use Case 1 “Intelligent Assistant in the cockpit to assist in ‘startle response’ adverse events”, Use Case 3 “Digital Intelligent Assistant for Urban Air Mobility coordinator to assist in traffic management”, Use Case 4 “Intelligent Assistant for tower (and remote tower) controllers to assist in routine and repetitive tasks for aircraft on approach”, and Use Case 6 “Airport Intelligent Assistant to monitor risk factor conditions associated with indoor spread of infectious diseases”.
These prototypes are considered low-fidelity as, amongst the HAIKU use cases, these are envisaged to reach a TRL lower than 6. This document captures the status of these four use cases at month 24 (August 2024).

This project has received funding by the European Union’s Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Information Table

Deliverable Number	D4.4
Deliverable Title	Low fidelity prototypes
Version	2.0
Status	Final version
Responsible Partner	DFKI
Contributors	ENAC, LiU, LFV, Skyway, DBL, Suite5, CERTH
Contractual Date of Delivery	August 31st, 2024
Actual Date of Delivery	August 29th, 2024
Dissemination Level	Public

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Document History

Version	Date	Status	Author	Description
0.1	*21.02.2024*	*First draft*	*N. Minaskan (DFKI)*	*Request for review*
0.2	*22.02.2024*	*In review*	*J. Diaz (THALES)*	*Review by Thales*
0.3	*23.02.2024*	*In review*	*S. Pozzi (DBL)*	*Review by DBL*
0.4	*28.02.2024*	*In review*	*N. Minaskan (DFKI)*	*Review integration*
1.0	*29.02.2024*	*Final Version (1st release)*	*N. Minaskan (DFKI)*	*Final quality check*
1.1	21.06.2024	Draft	N. Minaskan (DFKI)	TOC and document structure
1.2	31.07.2024	Draft	A. Duchevet, J-P Imbert (ENAC) C. Westing (LiU) S. Boonsong (LFV) M. Villegas (Skyway) R. Venditti (DBL) E. Biliri (Suite5) E. Spyrou (CERTH)	UCs inputs received
1.3	01.08.2024	Draft	N. Minaskan (DFKI)	Draft for review
1.4	05.08.2024	Draft	V. Arrigoni (DBL)	Review with comments and suggestions
1.5	16.08.2024	Draft	N. Minaskan (DFKI)	Version for final quality check
1.6	27.08.2024	Draft	V. Arrigoni (DBL) S. Pozzi (DBL)	Final quality check with minor adjustments
2.0	29.08.2024	Final version (2nd release)	N. Minaskan (DFKI)	Final version

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

List of Acronyms

Acronym	Definition
AoI	Area Of Interest
CTOT	Calculated Take- Off Time
ECAM	Electronic Centralized Aircraft Monitor
ECG	Electrocardiography
EFB	Electronic Flight Bag
EMG	Electromyography
EOBT	Estimated off blocks time
FOCUS	Flight Operational Companion for Unexpected Situations
GSR	Galvanic Skin Response
IFR	Instrumental Flight Rules
ISA	Intelligent Sequence Assistant
LSL	Lab Streaming Layer
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MSE	Mean Squared Error
ND	Navigation Display
PFD	Primary Flight Display
PPG	Photoplethysmography
R2	R squared
RR	Respiration Rate

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

SA	Situational Awareness
TRL	Technology Readiness Level
UI	User Interface
VRF	Visual Flight Rules

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Table of contents

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Executive Summary

This document describes the HAIKU low-fidelity prototypes, capturing their status at month 24 (August 2024).
More specifically, the four Intelligent Assistants prototypes considered low-fidelity are the ones designed and developed in Use Case 1, 3, 4 and 6.

Use Case 1 “Intelligent Assistant in the cockpit to assist in ‘startle response’ adverse events” is developing the FOCUS (Flight Operational Companion for Unexpected Situations)Intelligent Assistant and its goal is to support the pilot in single-pilot operation during startle and surprise events.

Use Case 3 “Digital Intelligent Assistant for Urban Air Mobility coordinator to assist in traffic management” is developing the DUC (Digital Assistant for UAM Coordinator (DUC) Intelligent Assistant and its goal is to support the UAM Coordinator in day-to-day operations (both planning and execution), handling standard tasks and providing assistance during emergencies like in-flight medical incidents.

Use Case 4 “Intelligent Assistant for tower (and remote tower) controllers to assist in routine and repetitive tasks for aircraft on approach” is developing ISA (Intelligent Sequence Assistant) and its goal is to support and enhance decision-making for Air Traffic Controllers, optimising runway utilisation in single-runway airports, by providing real-time sequence suggestions for arriving and departing aircraft.

Use Case 6 “Airport Intelligent Assistant to monitor risk factor conditions associated with indoor spread of infectious diseases” is developing COVAID and aims at tackling the critical issue of preventing the spread of airborne diseases in crowded areas, specifically airports.

This project has received funding by the European Union’s Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Introduction

This document describes the HAIKU low-fidelity prototypes, capturing their status at month 24 (August 2024).
More specifically, the four Intelligent Assistants prototypes considered low-fidelity are the ones designed and developed in:

- Use Case 1 "Intelligent Assistant in the cockpit to assist in 'startle response' adverse events",
- Use Case 3 "Digital Intelligent Assistant for Urban Air Mobility coordinator to assist in traffic management",
- Use Case 4 "Intelligent Assistant for tower (and remote tower) controllers to assist in routine and repetitive tasks for aircraft on approach",
- Use Case 6 "Airport Intelligent Assistant to monitor risk factor conditions associated with indoor spread of infectious diseases".

These prototypes are considered low-fidelity as, amongst the HAIKU use cases, are envisaged to reach a TRL lower than 6.
It is noted that, differently from the first release of D4.4, Use Case 5 "Intelligent Assistant to improve airport safety through data analysis" (previously included in this document) has been moved to Deliverable 4.5 "High-fidelity prototypes" as the Airport Safety Watch Intelligent Assistant has significantly progressed over the last year and is now expected to reach a higher TRL than previously planned; it is therefore considered high-fidelity.
However, Use Case 3 has been included in this document since, despite it being advanced from technology perspective, its progression strongly depends on the definition of a variety of contextual aspects associated with the futuristic UAM concept (operations, workforce, etc.) which are still under discussion in the whole aviation community. Therefore, this Intelligent Assistant is unlikely to reach TRL 6 and is therefore considered low-fidelity.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

1. Use Case 1: Flight deck startle response

The assistant in use-case 1 is named FOCUS (Flight Operational Companion for Unexpected Situations) and its goal is to support the pilot in single-pilot operation during startle and surprise events. Starting from TRL 1, the prototype is currently at TRL 3. The following functionalities are currently implemented:

- **Situation awareness (SA) support function**: The purpose of the SA function is to guide the pilot's attention through visual cues in the cockpit when their SA is degraded with respect to parameters concerning flight operation.
- **Stress regulation support function:** This function aims to reduce the pilot's stress through a haptic wristband and a pulsating green light on the main instruments (PFD, ND and ECAM).
- **Startle and surprise detection function**: Detection of startle and surprise in pilots based on physiological inputs.

FOCUS has been implemented in the A320 research simulator at ENAC. The architecture of FOCUS is based on a multi-agent system that communicates through a software bus named IVY. Physiological data is exchanged through a dedicated bus called LSL Lab Streaming Layer. A gateway between the two buses helps to exchange data (simulator data are sent to LSL to have triggers in the physiological time series). FOCUS platform needs several computers with W10 and a NVIDIA Jetson Xavier with Linux Ubuntu for eye tracking data processing. 10 agents that collect or send data on IVY are required for the operation of functionalities and are:

- **Situation awareness**: Application for calculating situation awareness.
- **Stress Regulation Agent**: Application that controls the vibrating bracelet attached to the pilot's wrist and the LED strip under the instruments in the cockpit.
- **ECAM**: User Interface (UI) with a transparent window above the ECAM instrument (electronic centralised aircraft monitoring).
- **PFD_ND**: UI with a transparent window above the PFD (Primary Flight Display and ND (Navigation Display) instruments.
- **ARGaze**: Application managing Tobii eye-tracking glasses. This application sends messages, including the Areas Of Interest (AOI) that the pilot is looking at.
- **Main receiver**: Application that sends raw data produced by the A320 simulator on the Lab Streaming Layer (LSL).
- **Eye2LSL**: Application that sends raw data produced by the Tobii Eye tracking device on the LSL.
- **Speech Warning** : Text to speech app that delivers auditory alarms.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

- **BITalino-PPG**: Application that retrieves the pilot's heart rate information via BITalino (which defaults to sending on LSL) to share this information on the IVY bus.
- **Haiku Monitor**: UI displayed at the top of the pilot's EFB (electronic flight bag) tablet. It allows the pilot to interact with the HAIKU assistant.
- **SimOne:** Aircraft simulator that sends continuous and discrete flight parameter values.
- **IvyProbeWindows:** UI allowing the person in charge of the SimOne aircraft simulator to send additional IVY messages.

In addition to the agents, a log recorder (LSL app) records everything on the LSL bus and a dedicated software records the front scene camera. The overview of the hardware and software architecture is provided in Figure 1.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

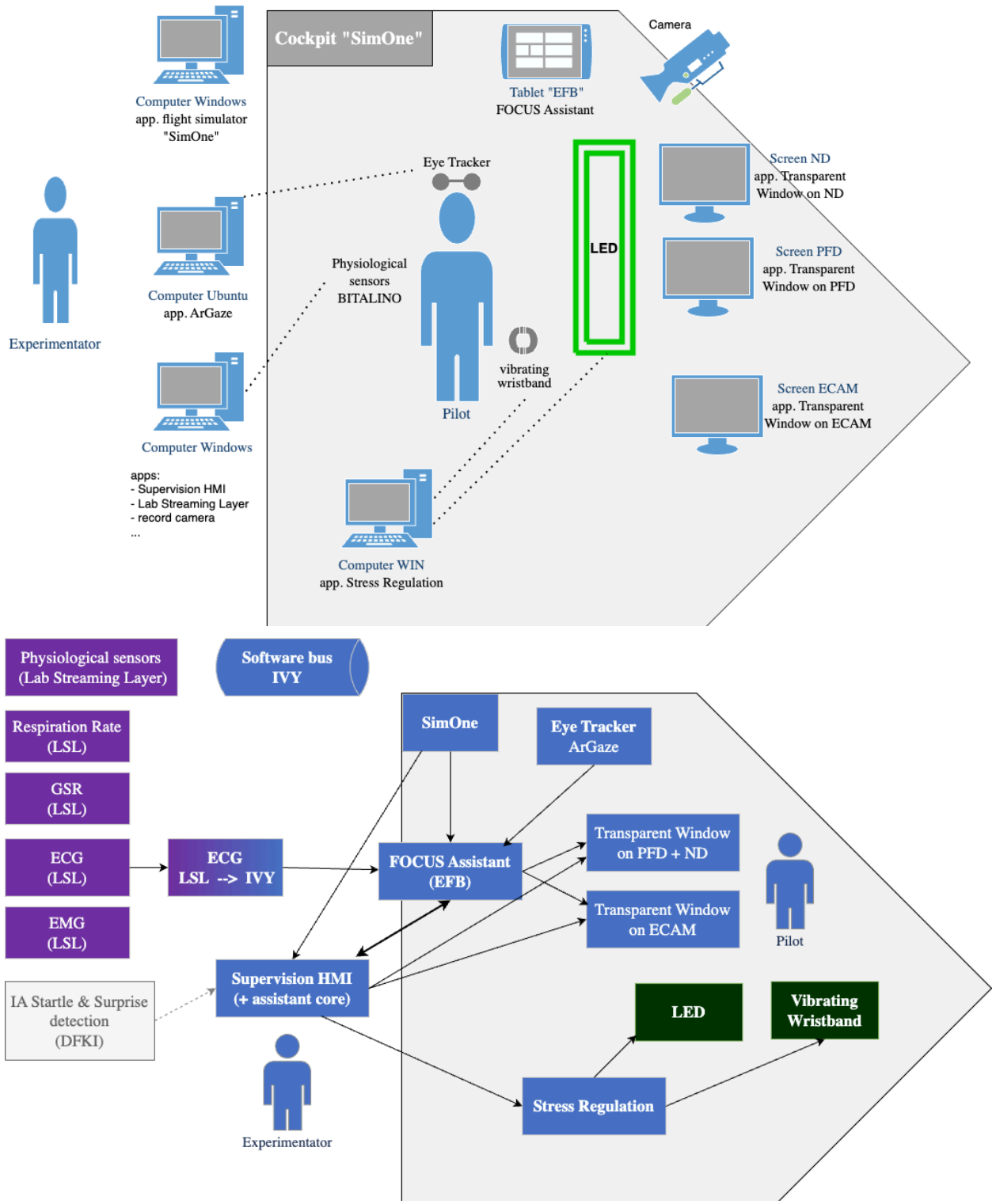

Figure 1: An overview of architecture in FOCUS. Top image: Hardware architecture. Bottom image: Software architecture

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

The operation of FOCUS is as follows: once the startle state is detected, the pilot can see an animated icon appear for 600ms (Figure 2) on their PFD/ND and ECAM; they have 5 seconds to interrupt the launch of the assistant if necessary. An animation displays a buddy coming from the right side of the PFD to join the pilot at the centre of the screen. When FOCUS leaves, the two personages separate. The design rationale is to provide a friendly image of FOCUS to the pilot in an unexpected situation. It also personifies FOCUS as a kind of 'human' pilot monitoring, to increase the trust and acceptance of FOCUS support.

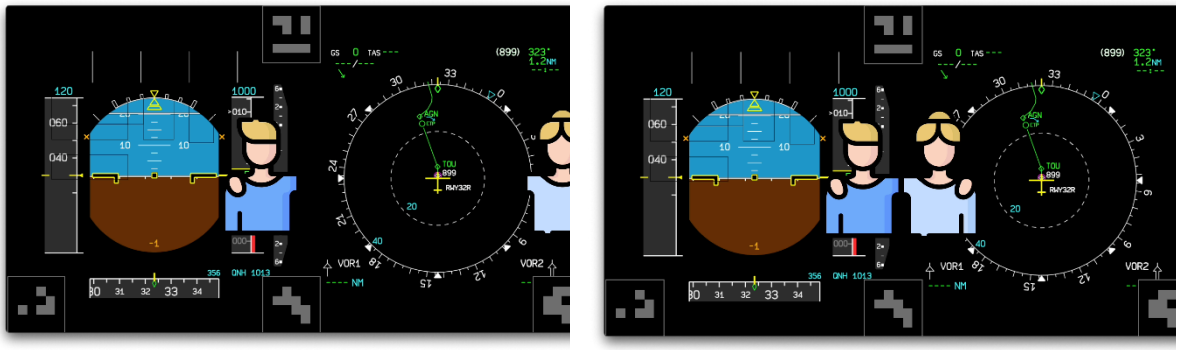

Figure 2: FOCUS personages

Consequently, to VAL1, in order to improve the gaze mapping of the 3D Cockpit by the ArGaze agent, QR codes have been included in the cockpit's HMI (Figure 2).
After these 5 seconds, the stress regulation function starts; after 15 seconds, the situation awareness support function starts. The stress regulation function stops after 30 seconds, and the situation awareness function stops when the pilot's situational awareness is correct for 90 seconds. This sequence is reflected in the figure below (Figure 3).

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

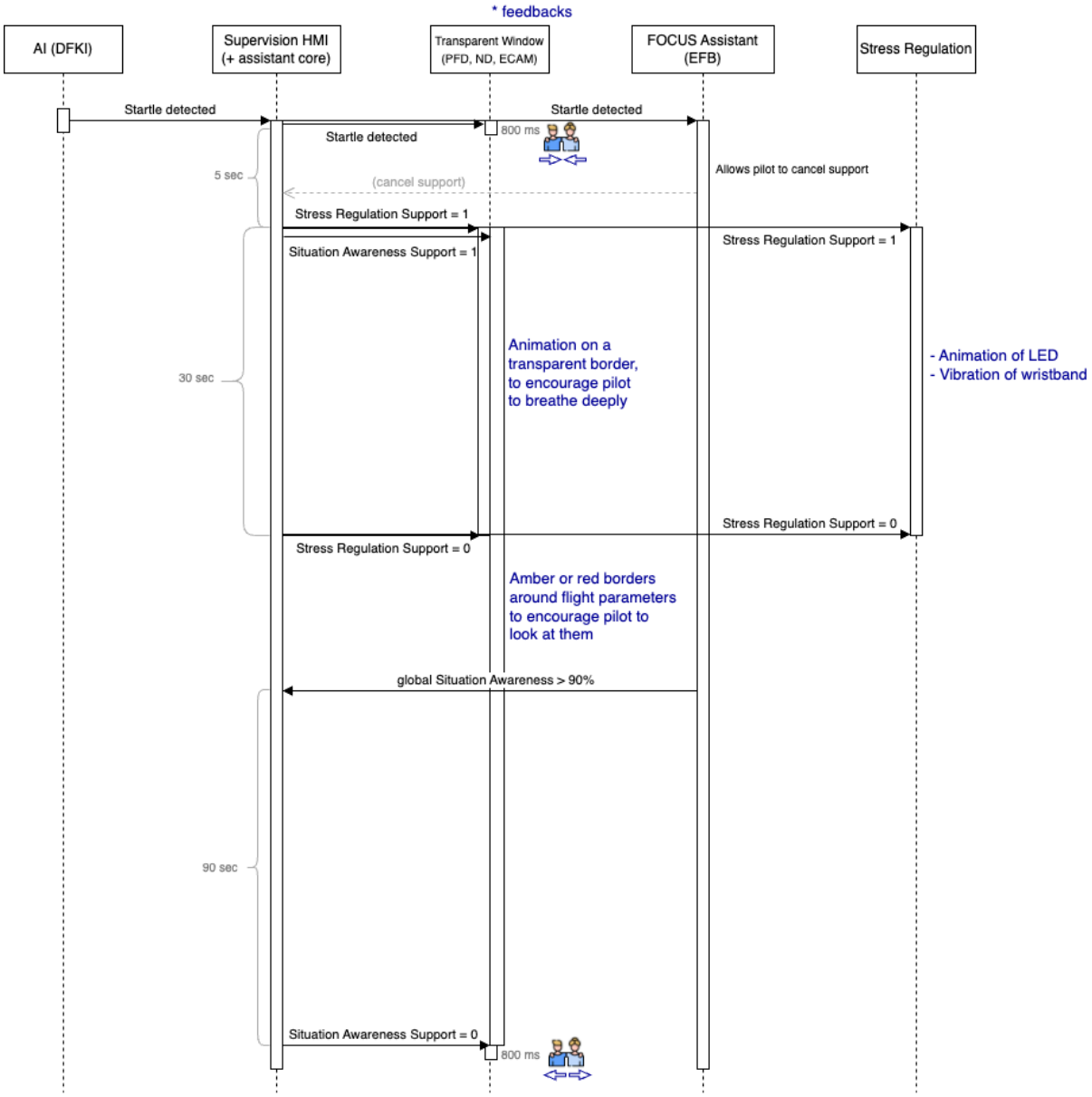

Figure 3: The sequence diagram of FOCUS. The assistant is shown by an icon.

In terms of sensors, the following is used by the agents to ensure the functionalities:

- **Eye tracker:** Eye tracking is provided by Tobii glasses. Moreover, ARGaze app developed for the use-case , keeps track of the fixation of the eye in predefined areas of the cockpit through marker tracking.
- **ECG, GSR, EMG, PPG, RR:** This data is collected by Bitalino Psychobit device and sent to LSL through Bitalino app (electrocardiogram, galvanic skin response, electromyogram, photoplethysmogram, respiration rate).
- **Flight parameters from the simulator:** 9 continuous parameters: altitude, vertical speed, indicated airspeed, heading, pitch, roll, glide slope deviation,

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

localizer deviation, fuel on board. 5 discrete parameters: auto pilot, auto thrust, thrust mode, vertical mode, lateral mode.

The mapping between the functionalities and agents is as follows:

- **Stress regulation support:** Stress regulation agent, BITalino-PPG, Haiku Monitor, ECAM, PFD-ND.
- **Situation awareness support:** situation awareness, ECAM, PFD-ND.

Stress regulation support

To reduce the pilot's stress, the stress regulation triggers vibrations in a haptic wristband at a pace of one vibration per second (equivalent to a 60 heartbeat per minute) to reduce the pilot's stress and a pulsating green light around the main instruments (PFD, ND and ECAM) and pedestal (Figure 4).

Figure 4: Stress regulation function HMI. Green light emanation around displays and haptic wristband.

Situation awareness support

Considering pilot's feedback during VAL1, the behaviour of SA support has been updated and simplified. The estimation of the SA is dependent on both the pilot's gaze fixation on the monitors and the value of the parameters observed (e.g. speed, heading). A global SA score as well as individual SA scores for individual parameters

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

are calculated. The parameters are based on the predefined areas detectable by the eye tracker:

- PFD air speed.
- PFD attitude for pitch and roll.
- PFD heading for yaw and localizer deviation.
- PFD vertical speed
- PFD altitude for altitude glide slope deviation
- PFD FMA mode for auto pilot, auto thrust, thrust mode, vertical mode, lateral mode.
- ECAM engine fuel flaps for fuel on board.
- AP ATR for auto pilot and auto thrust.

The SA score for each parameter is described as a function of time since last viewed by the pilot, it will also be changed if the parameter changed significantly according to the flight phase since last viewed. A change is significant according to a specific flight phase, for example a modification of speed of +10 kts is not considered as a critical change in cruise phase while it is in approach phase. For the discrete parameters the following logic is applied:

- When the value changes (e.g., Auto Thrust changes from "ON" to "OFF"), the pilot is given 5 seconds to look at the parameter (fixate on the corresponding area) and become aware of this value change. Transition from the "Initial" state to the "Change to look at" state.
- If the pilot does not look within 5 seconds, the application sends the warning message "Start WarningOnChange" for that parameter. Transition from the "Change to look at" state to the "Warning" state.
- When the pilot looks at the area corresponding to that parameter, the application sends the end of warning message "Stop WarningOnChange." Transition from the "Warning" state to the "Initial" state.

This logic for continuous and discrete parameters is displayed in Figure 3. SA agent publishes an awareness message every second, containing the situational awareness level for each flight parameter (both discrete and continuous) and the overall SA level, calculated as follows:

Global SA = (SA param.1 x Weight param.1) + (SA param.2 x Weight param.2) + ...

Compared to VAL1, SA behaviour for warning and alerts has been simplified. A warning is issued if the parameter has not been seen until the maximum threshold, an alert is issued if the parameter is above the maximum threshold.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

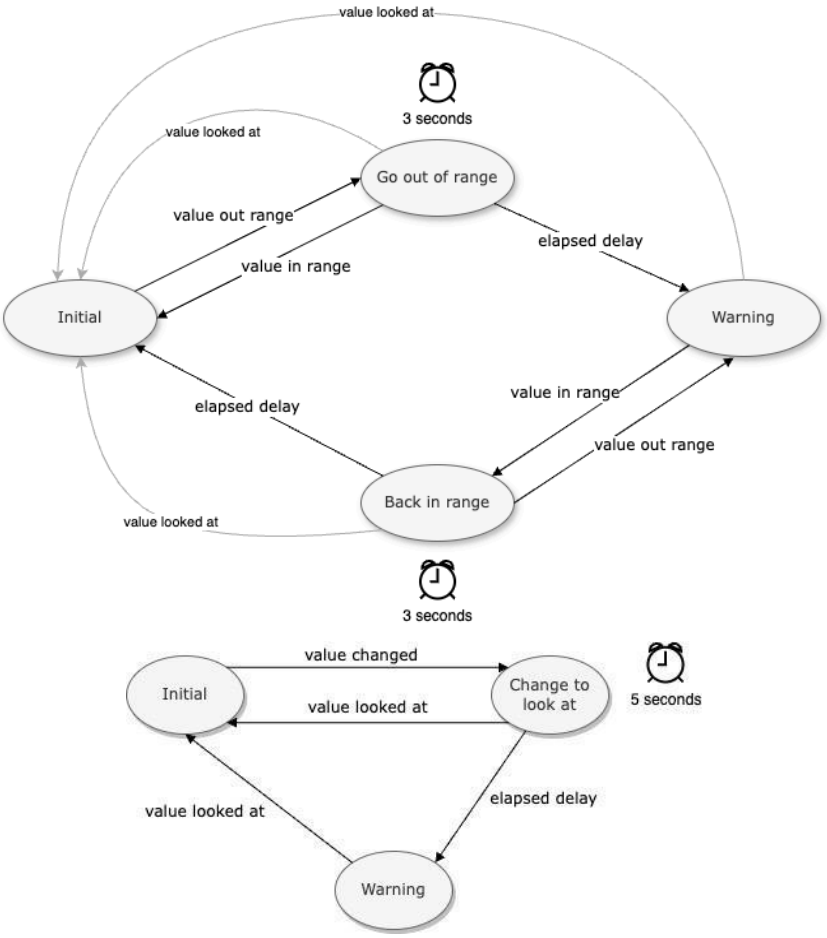

Figure 5: A finite state machine for parameters. Top: continuous parameters. Bottom: discrete parameters.

The warning messages are then visualised on PFD, ND, and ECAM for warning messages with amber and alert messages as red (Figure 6). Additionally, a text to speech agent delivers an auditory message to the pilot in case of alert only (VAL1 pilot's request).

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

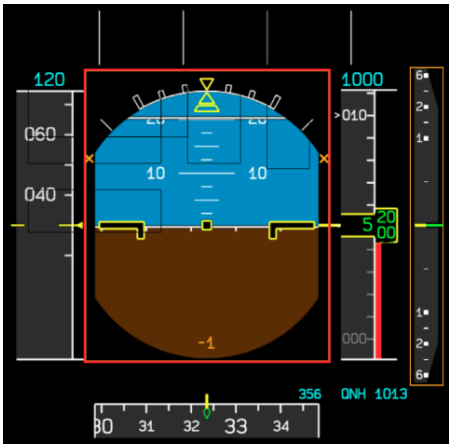

Figure 6: Red frame around the horizon and amber frame around vertical speed

Concerning the HMI, the first iteration of Haiku Monitor has been done for the VAL1. Considering pilot's feedback, we updated the UI to make it nicer, simpler, and more understandable. Haiku Monitor provides additional information about FOCUS internal behaviour. (Figure 7).

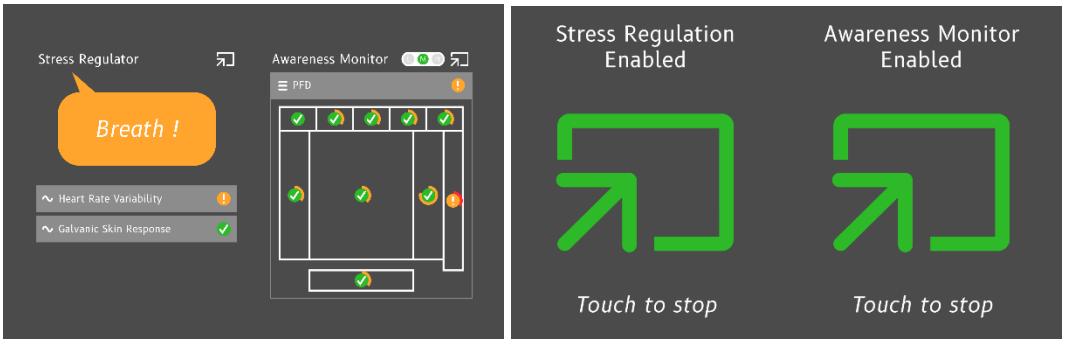

Figure 7: EFB display for SA and stress regulation, left side normal operation, right side startle mode.

In normal operation FOCUS monitoring for stress (HRV and GSR) is displayed on EFB. If the pilot is willing to start stress regulator or SA support, he can start those functions by clicking on the broadcast icon. If a startle is detected, both functions are started, and the startle mode screen is displayed to stop the functions if the pilot needs to. For each function started manually, the touch to stop screen will appear.
The meaning of the displayed icons is the following:

- The green check icon means that a parameter is okay,
- The amber progress bar represents the time since the parameter has not been seen,
- The amber check icon means that the parameter state is a warning (due to eye fixation time out),

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

- The amber exclamation mark (!) with a red progress bar means that the parameter is in alert because of a deviation, the progress bar represents the duration of the deviation,
- TheA red exclamation mark means that a deviation has not been seen since a duration threshold (Figure 8),
- The popup box gives advice to Breath if FOCUS detects an abnormal HRV or GSR,
- The broadcast button can be used by the pilot if s/he wants to request FOCUS support.

Figure 8: status icons

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

2. Use-case 3: Urban Air Mobility

The assistant in UC3 is named DUC (Digital assistant for UAM Coordinator). The UAM Coordinator is the human operator responsible for managing and assuring the safety of urban airspace and its users. This is achieved by means of providing several U-space services to U-space users. The goal of DUC is to support the UAM Coordinator in accomplishing and providing U-space services. To date, UC3 has developed a concept of operations (ConOps) for the UAM Coordinator working station.

The UAM Coordinator working station is developed for overseeing and servicing the operational environment of UAS and UAM traffic in cities. The operational environment comprises the airspace overhead urban areas, known as U-space, and ground infrastructure such as vertiports, cargo hubs, launch sites, and other sites where UAS and UAM aircraft are launched/takeoff and received/land. The operational environment in UC3 is based on the CORUS-XUAM (Eurocontrol, 2023) ConOps for U-space and UAS/UAM.

Figure 9 illustrates the UAM Coordinator working stations: a desk with three screens. While DUC is intended to function across all screens, the current prototype has DUC implemented in one screen (Screen 2).

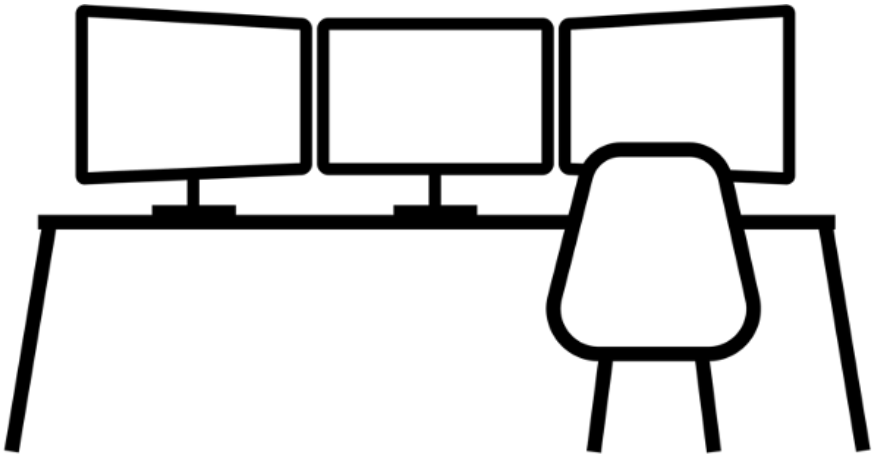

Figure 9: UAM Coordinator working station

Screen 1: Knowledge library with manuals, checklists, procedures to guide work in different situations. DUC should be able to suggest checklists to apply for different situations, execute checklist steps, and keep track of checklist progress. This screen and its HMI components have not yet been developed.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Screen 2: Situation display providing a map view of the city and U-space with information about UAS and UAM aircraft. The human can navigate through the interface using a mouse or touch. Layers and filters can be selected to hide/visualise different information (e.g., aircraft, u-plans, geofences). The DUC should be able to communicate through audio alerts, natural language, and through dialogue windows and overlays embedded in the situation display. The development of this screen and its HMI components was the focus of VAL1.

Screen 3: Digital journal/logbook system, storytelling explainer, and communication log (e.g., recording and storing communication made by DUC, UAM Coordinator, UAS/UAM operators, vertiport operators, other stakeholders). Voice communication should be transcribed and logged. This screen and its HMI components were partly developed for VAL1, focusing on the Storytelling explainer.

Communication system: Phone to contact UAS/UAM operators and vertiport operators. UAM coordinator can request the DUC to initiate calls. Direct connections to key stakeholders such as SOS, JRCC, ATM, Municipality emergency services, Helicopter emergency medical services (HEMS). This system has not yet been developed.

The UAM Coordinator and DUC collaborate on high-level tasks that have been defined according to different U-Space services provided to U-space users. Different screens in the UAM Coordinator's working environment are used to accomplish different high-level tasks. The primary screen for addressing most high-level tasks is screen 2 and the situation display. Figure 10 depicts how high-level tasks have been distributed across the screens in the UAM Coordinator's working position. The high-level tasks are based on the U-space services described in CORUS-XUAM (Eurocontrol). In Figure 10, the green boxes indicate high-level tasks that DUC is directly involved in. The orange boxes indicate high-level tasks where DUC monitors and conveys information provided by other stakeholders (e.g., visualising weather information provided by meteorological institutes or organisations on the situation display). The information is important for DUC to monitor as it can impact other services. This information is also provided to the U-space users.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

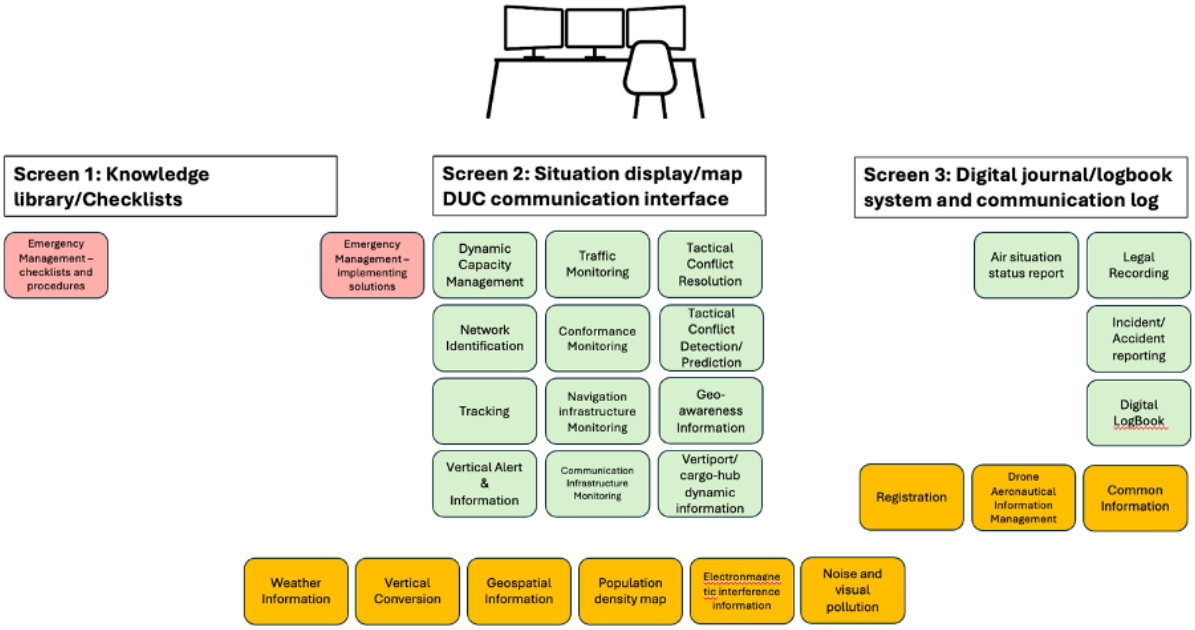

Figure 10: High-level tasks distributed across screens in the UAM Coordinator working position.

From a system architecture perspective, the UAM Coordinator working station prototype can be divided into frontend and backend. The frontend regards the human machine interface (HMI) and the components that the human (i.e., UAM Coordinator) interacts with. The backend focuses on the hardware, software, data, and infrastructure. In the current implementation of the prototype (tested in VAL1) we have focused on realising the situation display and DUC HMI, both part of Screen 2. This comprise the following components:

1. UTM-city HMI and drone sim,
2. a scenario builder,
3. DUC System and HMI

The current implementation of the DUC system is set up as a "wizard of Oz" to allow for testing the HMIs and communication protocols without a functional DUC.

The human mainly interacts with the graphical interface of UTM-city with the IA HMI overlayed on top of it. These three systems are controlled and synchronised by the scenario builder software. Currently, scenarios are scripted in a timeline using the scenario builder software.

The software architecture of Screen 2 is schematically illustrated in Figure 11. Figure 12 depicts the components that the human operator interacts with, and what is controlled by the experimenter. UTM City, DUC system/HMI, storytelling explainer and scenario

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

builder were all hosted on one computer. Two monitors were connected to the computer to present both Situation Display (Screen 2) and Storytelling explainer (Screen 3). All interactions with the system by the operator were made with a computer mouse.

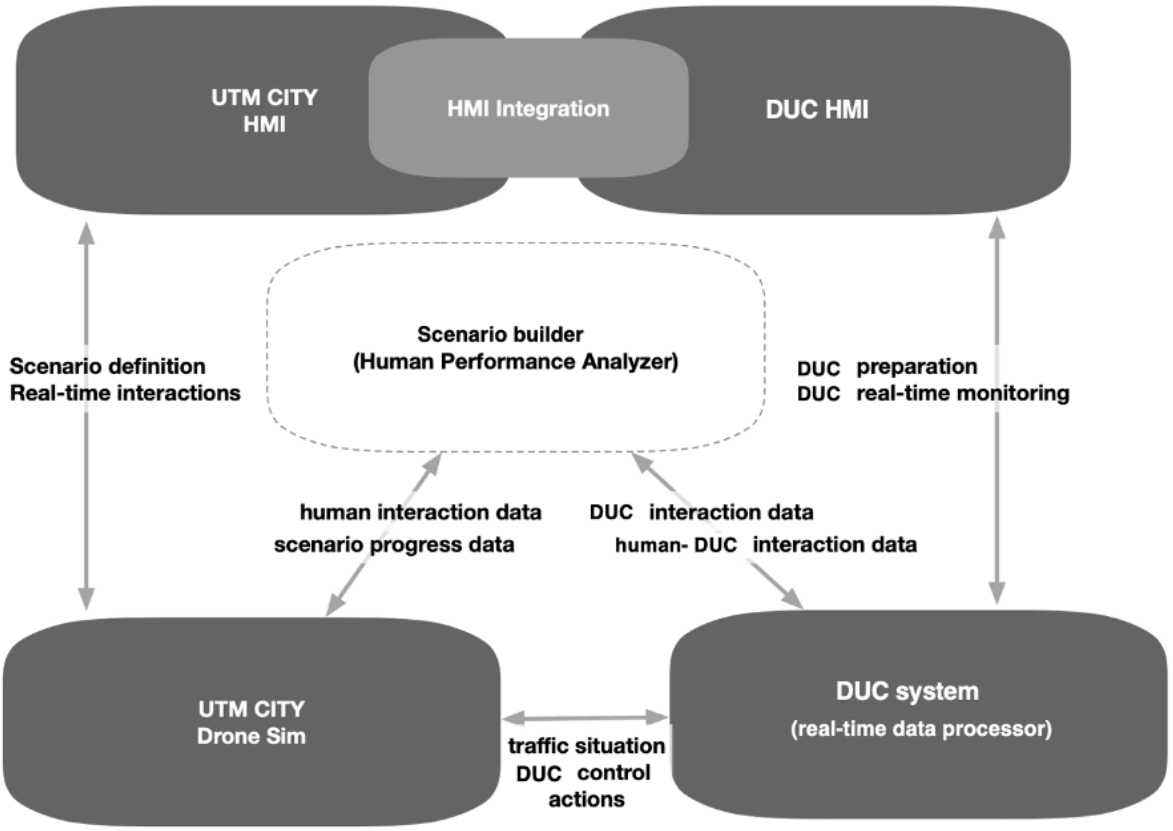

Figure 11: Illustration of Software Architecture of the Situation Display (Screen 2), clarifying the connections between the IA HMI and system (i.e., DUC HMI and system) and the scenario builder.

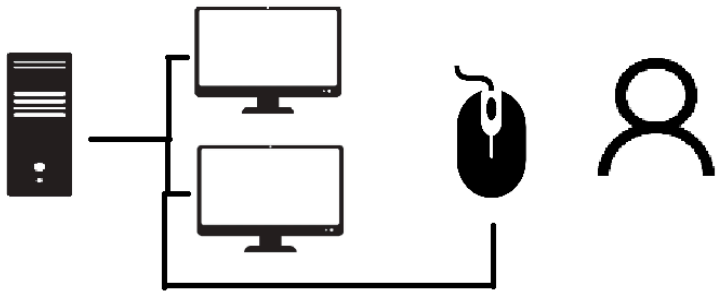

Figure 12: Overview of hardware configuration of the current prototype (used in VAL1).

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

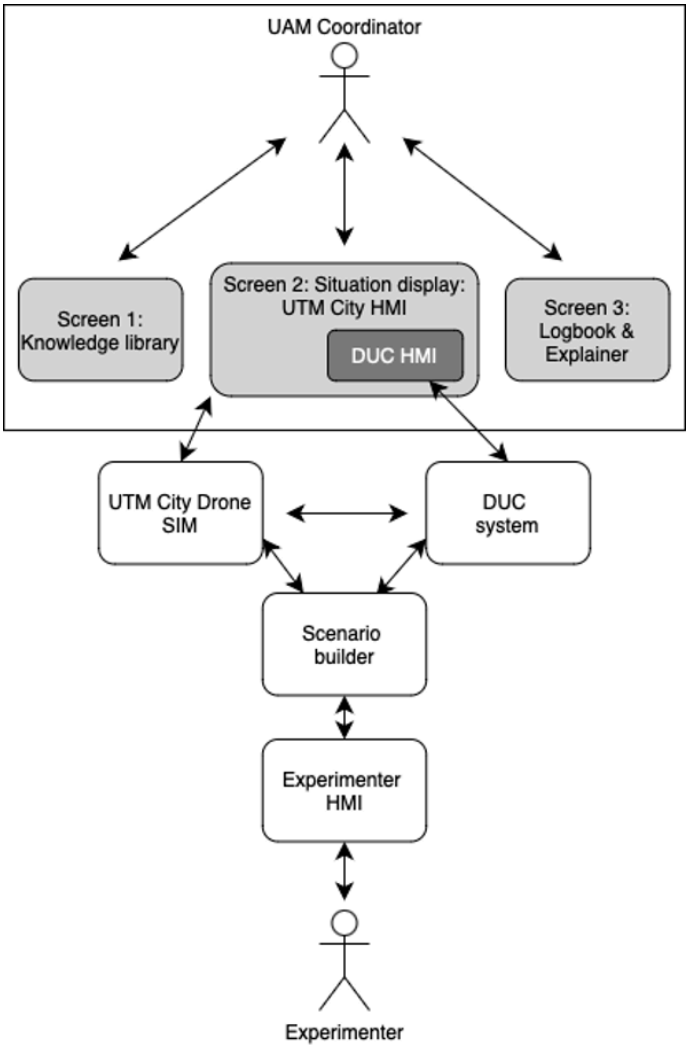

Figure 13: Overview of software components and human actors.

The prototype is made up of two UTM City components, the dronesim and a HMI, and two DUC components, a real-time data processor and a HMI, as well as a scenario builder which communicates with both the dronesim and the DUC data processor.

UTM City dronesim is a simulator of drone (UAS) services, drone traffic and drone control. A service here refers to the operation undertaken by the drone, and can be considered as the U-plan (i.e., flight plan). The dronesim is extended in the use case to also include UAM vehicles. The software sets up services using abstractions and automatically generates an, potentially large, amount of traffic based on the parameters in these services. The two most important examples of such abstraction are 1) point-to-area service, which is typically used to simulate delivery from a logistics hub to a city area, and 2) point-to-point service, which sends one or more

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

drones via a predetermined path, typically simulating a single delivery. The simulator mimics the communication between drones and drone operators (providing the drone services), and an airspace authority. The airspace authority component keeps a list of rule-zones, such as no-fly zones and other types of constraints, and will adjust incoming authorization requests into plans that are valid in the current airspace.

Typically, flights simulated by dronesim will continue their mission until they are done (i.e., have reached their destination or designated endstate). We have added the possibility of setting an alternative endpoint, a diversion from its original route. This is used to simulate the diversion to a close-by hospital in case of an emergency.

UTM City HMI provides a map visualisation of an area of interest, such as a city or part of a city, with components for visualisation and control of aircraft services (e.g., U-plans), rule-zones (i.e., no-fly zones and geofences) and airspace parameters. The view provided is slightly slanted 3D, using drop-lines to emphasise distribution of objects in all dimensions. The HMI provides a menu for editing scenarios, for example by loading pre-defined sets of drone services or adding and drawing new areas for drone delivery, and another menu for controlling the U-space authority and its parameters. However, currently we are not using the scenario menu since scenarios were predetermined for VAL1.

DUC system handles requests from the scenario builder to present pop-up dialogs for the user. Furthermore, the system handles the user inputs from the DUC HMI and forwards them to the scenario builder to further progress the scenario. Currently, the DUC system is controlled by an experimenter through the Experimenter HMI.

DUC HMI provides a user interface for the operator to interact with the DUC system. The HMI consists of pop-up dialogs onto UTM City HMI. The DUC HMI is described in greater detail in the next section.

Scenario builder is used as a scenario planning/building and generation platform that interacts with the UTM City Drone Sim to control scenarios and events in the simulation. The scenario builder allows for scripting events and DUC actions and communication messages, for instance to inform or provide recommendations to the UAM Coordinator.

Experimenter HMI provides the experimenter with an interface to build and control the simulation and DUC System through a Wizard of Oz approach. This includes steering events in the simulation and DUC responses to those events, e.g., by means of providing overlays, aural alerts, or text information in dialogue windows.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

The currently implemented low fidelity prototype of the UAM Coordinator working position comprise two screens:

- Situation display (Screen 2). Building on UTM City, the HMI consists of a situation display comprising an interactive visualisation of the U-space in Stockholm to simulate UAM traffic/services and airspace rules on a map with a dashboard. An overlay system is used to illustrate vertiports in the U-Space airspace. The scenario builder tool is used as a combined scenario planning and generation platform that interacts with the UAM HMI.
- Storytelling explainer (Screen 3): The storytelling explainer is provided on a separate display, next to the situation display. The storytelling explainer consisted of four elements which were scripted video recordings combining visual media and narration to explain a particular construct (e.g., DUC). A storytelling explanation was triggered by specific happening occurring in the simulation as scripted in the scenario builder tool. The Storytelling explainer is described in detail in D5.2.

The DUC HMI is embedded with the situation display. DUC communicates with the UAM Coordinator using voice and aural alerts (e.g, informing opening of geo-zones) and dialogue window The UAM Coordinator uses a mouse to orient and move around the situation display and interact with the DUC HMI.

Dialogue window

The DUC frontend prototype was integrated with the situation display using a pop-up window software package.

Figure 4 shows a dialogue window that DUC presents in relation to a medical passenger emergency (as implemented in VAL1). An initial concept was developed based on glyph design principles from Nylin (2023). The glyph displays the speed and altitude of an ATCO. In VAL1, DUC tasks and actions were scripted beforehand on a timeline using the Scenario builder, making each interaction from DUC appear at a specific time point.

The dialogue window can provide information from, and interactions with, DUC.

- Information-only dialogs only have the option to acknowledge the information with one button (i.e., “OK”).
- Simple decision dialogs have the options to answer “yes” or “no” to an action.
- Multi-choose dialogs have up to three options to choose from, where additional information is provided for each alternative (e.g., priority in red yellow green).

This project has received funding by the European Union’s Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Figure 14 is a multi-choice dialog, regarding the medical passenger emergency, with an extra option to activate medical priority for the specific drone.

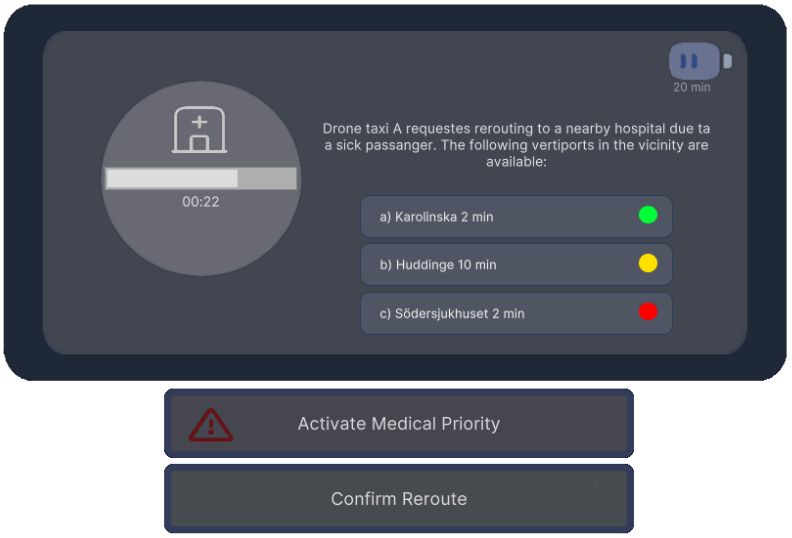

Figure 14: Dialogue window for DUC in medical emergency.

Information overlays on situation display

DUC can, on UAM Coordinator request, provide information symbols relevant to U-space or UAS/UAM components or elements on the situation display. Figure 15 depicts a legend for the implemented symbols, while Figure 16 illustrates how they appear on the situation display. Nine different symbols have been implemented:

- Point-to-point service: UAS/UAM aircraft operating from one point to another, for instance an airtaxi flying from one city area to another or a drone flying from a logistics hub to a city area.
- Point-to-scan area service: one or more aircraft, typically UAS, operating via a predetermined path, typically simulating a single delivery.
- Point-to-loop service: UAS/UAM aircraft operating within a volume of airspace in a loop, such as a surveillance service, photoshoot, or search and rescue.
- No-fly zone/geofence: restricted volume of U-space that may prohibit or restrict access for aircraft.
- Airport gate: restricted u-space surrounding an airport, controlled by air traffic control.
- Vertiport: vertical takeoff and landing site for UAM aircraft
- Point of interest: e.g., destination or departure location for an aircraft.
- Logistic zone: geographical area related to freight distribution
- Area of interest: area that DUC considers is of particular interest, e.g., defining the time window for which a recommendation is valid.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

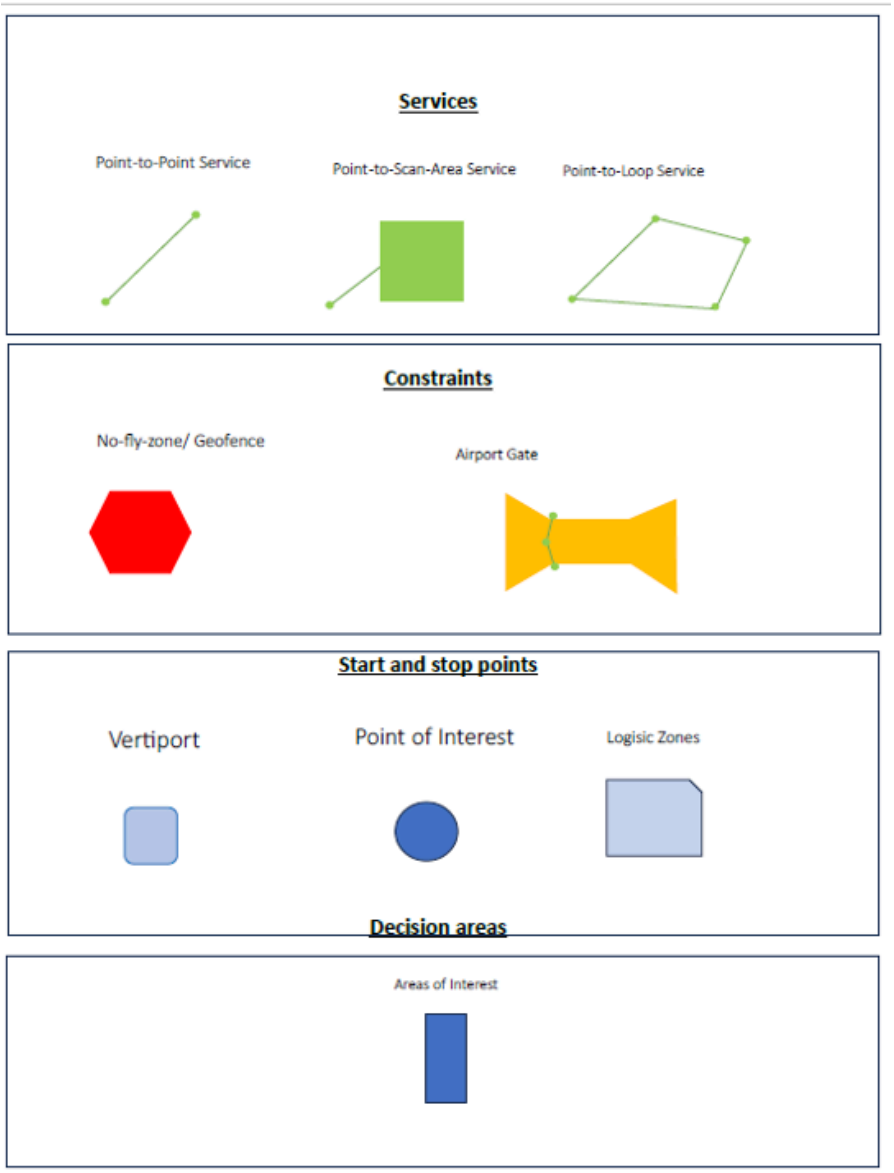

Figure 15. Symbol key with information about UAM related symbols in the UATM HMI

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

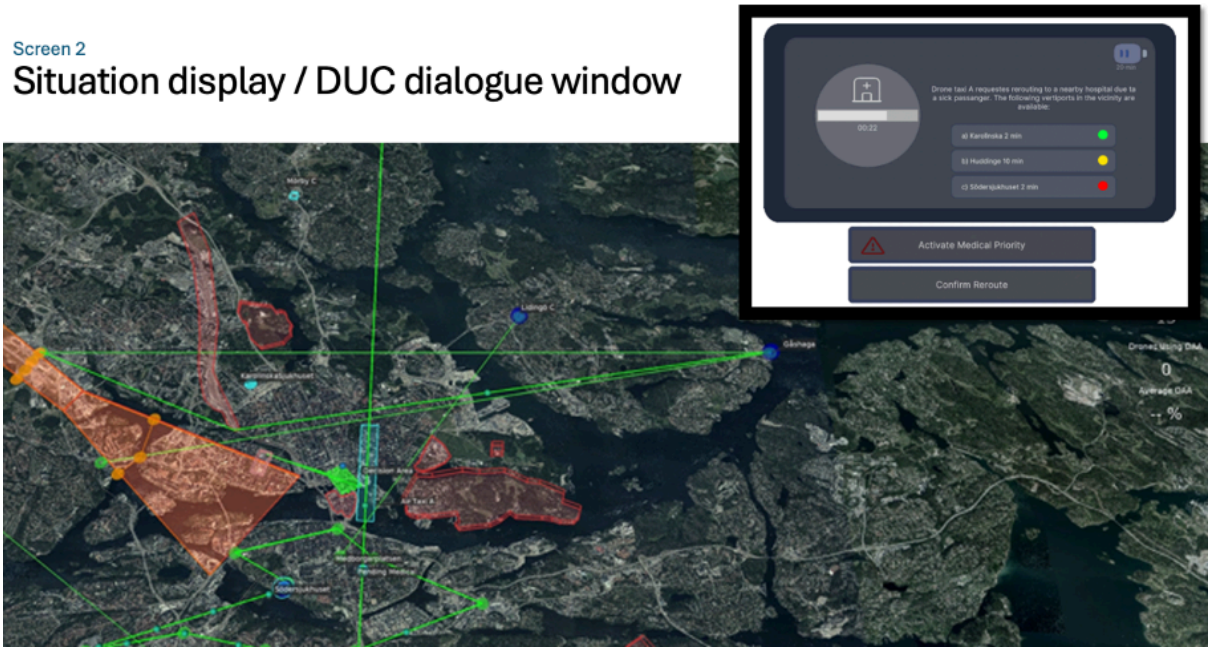

Figure 16. Situation display depicting U-space over Stockholm City. The DUC dialogue window is embedded with the situation display and defaulted to appear in the upper left corner. In this figure it has been enlarged and pasted in the upper right corner for clarity.

Aural alerts and voice communication

DUC can provide aural alerts to provide information to the UAM Coordinator, for instance that something has happened in the U-space that is reflected in the situation display. Examples include the activation of vertiports, geofences, and no-fly zones. An aural alert is provided with information on the map.

The following alerts have been implemented:

- *"Activating Constraints"* - DUC informs the UAM Coordinator that the process of activating geofences/no-fly zones has started. The constraints appear on the situation display with overlay symbols.
- *"Constraints Activated"* - DUC informs UAM Coordinator that the process of activating geofences/no-fly zones is finished. The constraints are shown on the situation display with overlay symbols.
- *"Activating Vertiports"* - DUC informs the UAM Coordinator that the process of activating one or more vertiports. Overlay symbols on the situation display show which vertiports that are affected.
- *"Vertiports activated"* - DUC informs the UAM Coordinator that one or more vertiports are now operational. Overlay symbols on the situation display show which vertiports that are affected.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

- *"Activate Tomteboda no fly zone"* - DUC informs the UAM Coordinator that a particular no-fly zone is being activated. The location and volume of the no-fly zone is shown on the situation display with an overlay symbol.
- *"Pending Medical"* - DUC informs the UAM Coordinator of a potential medical emergency situation onboard an aircraft. The aircraft, its current location, and U-plan is shown on the situation display.
- *"Medical Confirmed"* - DUC informs the UAM Coordinator that the medical emergency situation has been confirmed with the operator of the aircraft. The aircraft, its current location, and U-plan is shown on the situation display.

Interaction example

The following describes DUC's actions in relation to the medical emergency implemented in VAL1.

- DUC receives information about a medical emergency (sick passenger) form the Air Taxi Operator. DUC first notifies the UAM Coordinator in a dialogue window by stating "Pending Medical" together with information about the affected Air Taxi. The UAM Coordinator can acknowledge the message by clicking OK. When clicking OK, the button turns light grey to indicate that it has been clicked. If the UAM Coordinator clicks OK (e.g., using the mouse), the dialogue window closes. The circle to the left in the dialogue window contains a hospital symbol to indicate a medical emergency. The bar below indicates a timeline calculating down (in this example 22 seconds of 1 minute has passed) to indicate time available, according to DUC, for deriving a solution to this problem.

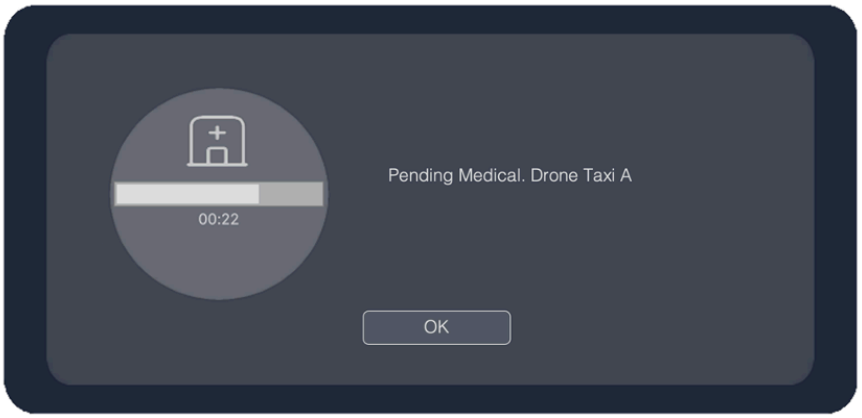

Figure 17. Dialogue window for Pending Medical

- When the medical emergency is confirmed by DUC, the dialogue window is updated with information that DUC is engaging in a search for nearby hospitals to which the Air Taxi could be rerouted. On the situation display, Air Taxi A is

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

indicated by a blue “point of interest” symbol. The U-plan of the Air Taxi is shown as a green point-to-point service.

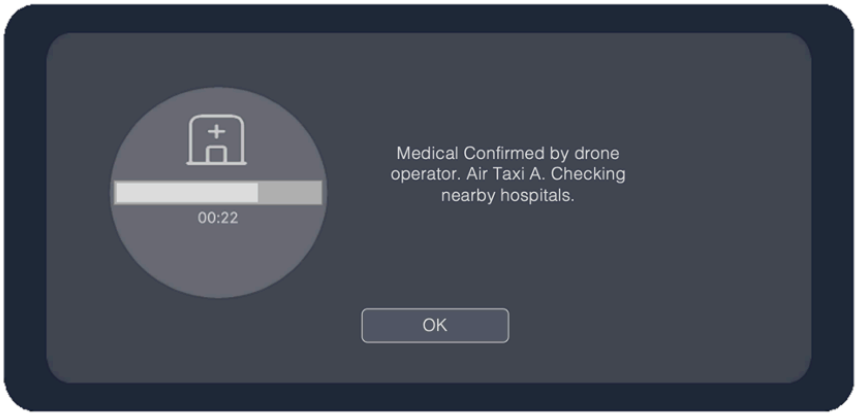

Figure 18. Dialogue window for Medical Confirmed

- DUC then provides the UAM Coordinator with a recommended solution consisting of alternative hospitals for diversion in a priority list. Each alternative has a colour indicating the priority of that selection. To determine the priority, DUC would consider several factors such as a) patient waiting list, b) vertiport availability, and c) cardiologist availability. Note that in the current prototype the alternatives have been scripted beforehand. The green colour was designed to indicate the highest priority, yellow a lower priority, and the red dot indicates the least priority. The green dot next to “Karolinska” indicates the preferred choice suggested by DUC. The UAM Coordinator can decide the hospital to reroute the Air Taxi by clicking on one of the alternatives. There are two buttons below the dialogue window that states “ Activate Medical Priority” and “Confirm reroute”. They are initially shown in dark grey to indicate that medical priority (all other traffic will give way) and a reroute can be implemented, but has not yet been so. On the situation display, the alternative hospitals are shown by means of point of interest symbols. An area of interest symbol is shown along a selected portion of the Air Taxi’s U-plan to indicate the area within which a decision is to be made.

This project has received funding by the European Union’s Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

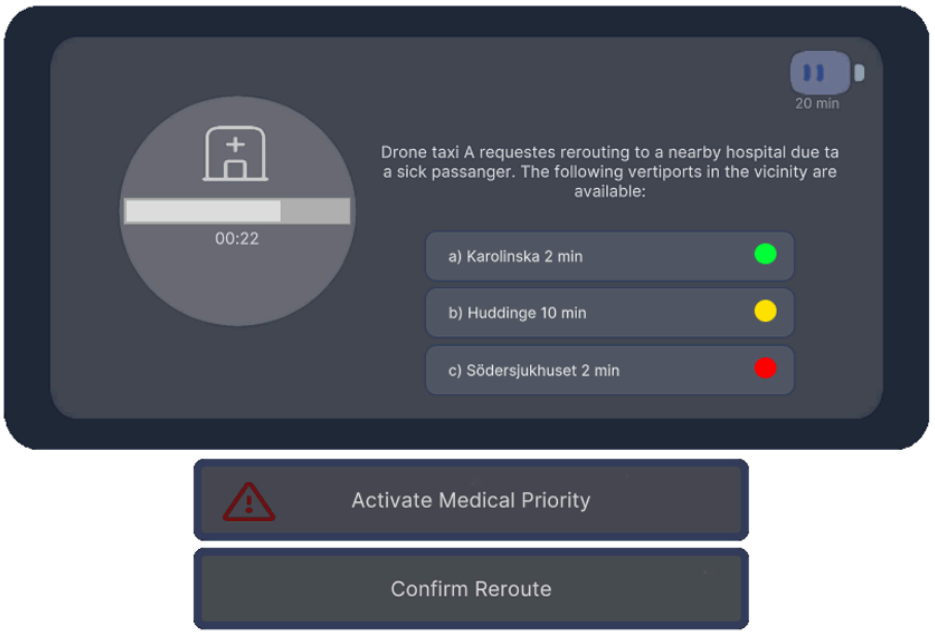

Figure 19. Dialogue window with DUC's recommendations.

- Once the UAM Coordinator has made a decision on which hospital to reroute the Air Taxi (by clicking on it), the selected button turns light grey. The UAM Coordinator can activate the medical priority for the Air Taxi, or it is done automatically by DUC once a selection for reroute has been made. This is also indicated by a green dot and the button turning lighter grey. Once the reroute has been confirmed with the Air Taxi operator and hospital, the "Confirm Reroute" button turns light grey.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

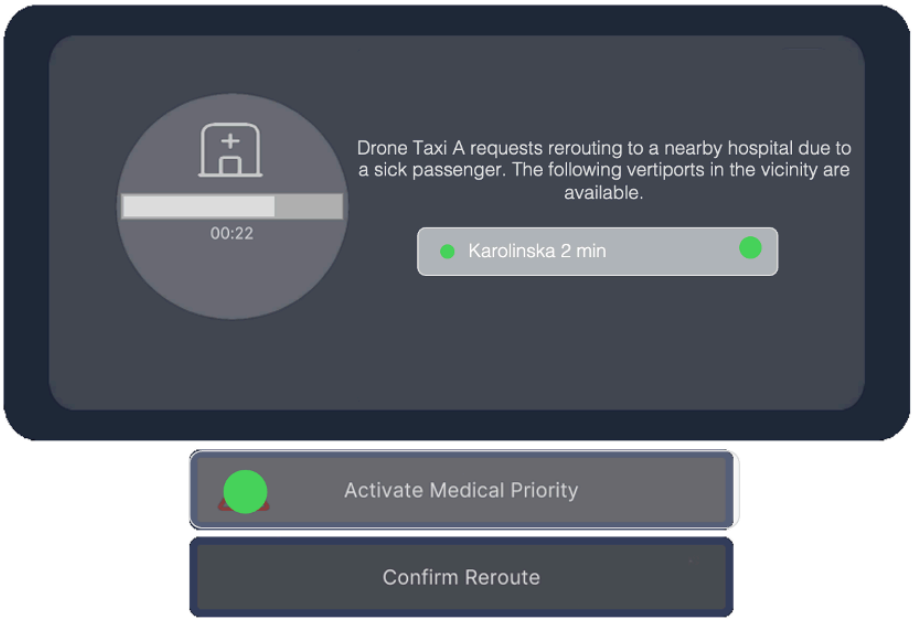

Figure 20. Dialogue window with selected destination for reroute.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

3. Use Case 4: Digital and Remote Tower

The assistant in Use Case 4 is an intelligent sequence assistant (ISA) which will support tower controllers at single runway airports by providing the best sequence for aircraft departures and arrivals. The Use Case started with TRL 2 and is currently at TRL 5. At present three main components are under development:

- **Optimization engine**: Optimization algorithm which calculates the optimal sequence can update the result based on the parameters given by the controller or the incoming flight data.
- **User interface**: The UI will show the controller the optimal sequence calculated by ISA. The interface can display the optimal sequence of the day, explanations on why sequence is suggested and refresh the sequence based on actions performed by the controller.
- **Data sink infrastructure**: An infrastructure which allows data from different sources (e.g. operational, simulation) to be pushed into the prototype.

The optimization engine is consisted of two services:

- A service that computes the estimated arrival time for arrival flights by applying the appropriate trained ML model depending on the active runway (direction) and the initial points of the incoming flight's trajectory. Two types of models have been trained with very similar performance, specifically a linear regression model and a feed forward neural network with one hidden layer. The aircraft's current position, true air speed and track are used as input for the models, whose output is the remaining time in seconds until the aircraft lands.
- A service that configures and applies an optimization algorithm using the current set of active aircraft, i.e. the aircraft which the controller needs to consider for sequencing. The problem is formulated as a MILP problem. Its input includes: the estimated arrival time for arrival flights, the EOBT/CTOT for departure flights together with the tolerance windows, the wake turbulence category, and the flight rules category (instrument or visual) for each aircraft, and the estimated taxi times. The model returns the flight sequence (including arrivals and departures) that minimises the total runway usage time, i.e. the sequence that ensures the last flight of the sequence will depart/land as soon as possible.

The models are configured and trained using 172 simulation exercises, each one containing multiple arrivals and departures. It should be noted that for the first phase, edge cases (e.g. trajectories with irregular movements and training flights) were excluded from the analysis. The following data is used in the simulation:

- Static data from airport such as runway coordinates.
- Flight data in the time frame:
    - Aircraft information.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

  - Flight plan.
  - Timestamped stream of position, track, and true airspeed.
  - Timestamped event information (e.g. landing clearance, engines running).

For departing flights, the flight plan includes the EOBT or CTOT which are given as inputs to the optimization algorithm. For arriving flights, a machine learning model is implemented to provide the estimated arrival times which are required to execute the optimization part to get the flight sequence.

Based on literature review and the nature of the problem, a linear regression and a simple feedforward neural network are initially chosen as prediction models. More complex architectures can be considered depending on results. A different model is trained per runway direction and to train the models the following steps were taken:

- Trajectories of arrival flights are grouped according to the active runway. All subsequent steps are performed per group since the direction of approaching aircrafts requires different handling, hence a different model.
- Data cleaning to remove unusable trajectories e.g. go-around movements (total number after cleaning is 272 trajectories).
- Trajectory data transformation to tabular format. Each row corresponds to a single point within an arrival trajectory. Extra column for seconds-until-aircraft-landed was added.
- Position (x,y,z), track, true air speed, aircraft model and flight rules (IFR, VFR) are used as features.
- Extra features were created to capture the distance of the aircraft from the runway, the distance from a specific point in airspace where arriving aircraft trajectories converge and after which ATCO starts considering an aircraft in their sequence, and the aircraft acceleration.
- Distance computation is done using 3D data and in terms of ground distance.

For evaluation of the models R2, MAE, MSE, and MAPE metrics were used. The linear regression model shows very similar results. Both model types showed satisfactory results and will be further improved and evaluated against new simulations with more complex situations. The optimization engine receives input directly from the simulations and the ETA results to provide the calculated sequence that will optimise runway usage. A potential follow-up for the optimization engine would be to include extra dimensions of airport operations such as taxi routes for arrivals and departures, airport layout, local rules, etc. which affect the sequence of the flight.

In terms of **AI**, there is an optimization engine, which consumes data from different sources will execute the optimization algorithm for the optimal sequence calculation, having the ability to trigger new sequence calculations based on specific stimuli, such

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

as actions performed by the controller, new data flight data available (for example by new incoming/outcoming flights), etc.
There is also a data sink infrastructure, allowing operational and simulation systems to push data into the ISA prototype, which will be stored locally and pushed to the AI optimization engine for the execution of the different calculations. The following ML components are present in the use case:

- A service that computes the estimated arrival time for incoming flights by applying the appropriate trained ML model depending on the active runway (direction) and the initial points of the incoming flight's trajectory. Two types of models have been trained with very similar performance: a linear regression model and a feedforward neural network with one hidden layer. The aircraft's current position, true airspeed, and track are used as inputs for the models, which output the remaining time in seconds until the aircraft lands.
- A service that configures and applies an optimization algorithm using the current set of active aircraft, which the controller needs to consider for sequencing. The problem is formulated as a MILP problem. Inputs include estimated arrival times for incoming flights, EOBT/CTOT for departure flights with tolerance windows, wake turbulence category, flight rules category (instrument or visual), and estimated taxi times. The model returns the flight sequence (including arrivals and departures) that minimises total runway usage time, ensuring the last flight departs/lands as soon as possible.

In terms of **data and sensors**:

- Static airport information: runway & stand coordinates
- The active runway
- Time separation between consecutive flights (there is a predefined value for this, but it is also an adjustable by the user property)
- Per flight information:
    - arrival/departure/overflight indication
    - aircraft type
    - aircraft model
    - wake turbulence
    - assigned parking stand
    - flight rules
    - emergency flight indication
    - flight type (operational or other)
    - EOBT or CTOT for departures
    - callsign
    - etc

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

- Per flight & per timestamp information:
    - Events (e.g., clearances, landing & take-off confirmation, status changes): For each event, the timestamp, description and value are provided.
    - Positions (X, Y, Z), track and true air speed for each timestamp according to the defined frequency

This data is processed in real-time (every second) to obtain:

- Optimal aircraft sequence of arrivals and departures
- The explanations for the sequence and for sequence changes

There is no gender bias at all (no information shared about age, experience, nationality etc.).

The **HMI** is designed based on the requirements outlined in D4.1, with ISA providing sequence suggestions to enhance controllers' decision-making for handling departures and arrivals, aligned with the envisioned HAT level. Initially, the interface was a mock-up created using the prototyping tool Figma and was utilised during VAL 1 to validate the core concept. The interface for Validation 2 will be an interactive prototype developed with Nuxt.js, allowing Air Traffic Control Officers (ATCOs) to interact with ISA in a simulator, closely mimicking real operational scenarios.
The HMI displays the sequence in real-time to controllers, highlighting relevant changes and providing varying levels of information based on the CLT framework outlined in D5.1. The sequence is always shown on the electronic bay on the left-hand side of the ATCOs' display, which includes the electronic strips and connection status (Figure 7). The electronic strips, like those typically used by ATCOs, feature additional magenta numbers on their left side, indicating the order of each aircraft in the current sequence of arrivals and departures. The magenta colour helps distinguish ISA-related features from other colour-coded elements.
The "Connection status" text at the top of the bay indicates whether ISA is connected to the database and calculating a sequence. When there are no calculations, the text just shows when the last sequence was calculated.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

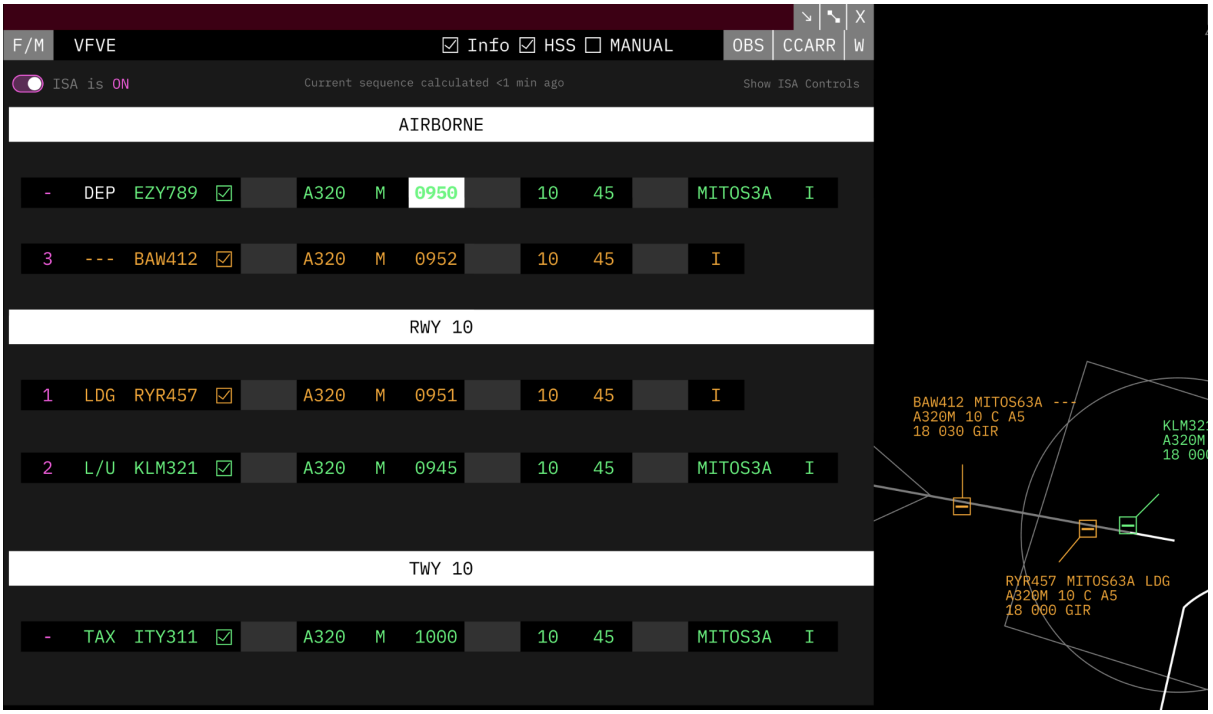

Figure 21: HMI overview.

Initially, ISA displayed sequence numbers for all aircraft on the electronic bay. Following VAL 1, ATCOs indicated they were primarily interested in aircraft approaching the runway soon, so the number of aircraft in the sequence was reduced to three. The critical part of the HMI is when ISA calculates a new sequence and presents it to the ATCO. When a sequence change is triggered by external factors, the numbers on the electronic strips start blinking, alerting the ATCO of the change. ISA then recalculates the sequence. Once the new sequence is calculated, the sequence numbers on the electronic strips are outlined with an arrow next to them (Figure 21), addressing a need identified during VAL 1 where ATCOs sometimes missed the aircraft involved in the sequence change.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Figure 22: 3 sequences and arrows on the electronic strip.

The sequencing procedure is illustrated in the figure below (Figure 22):

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

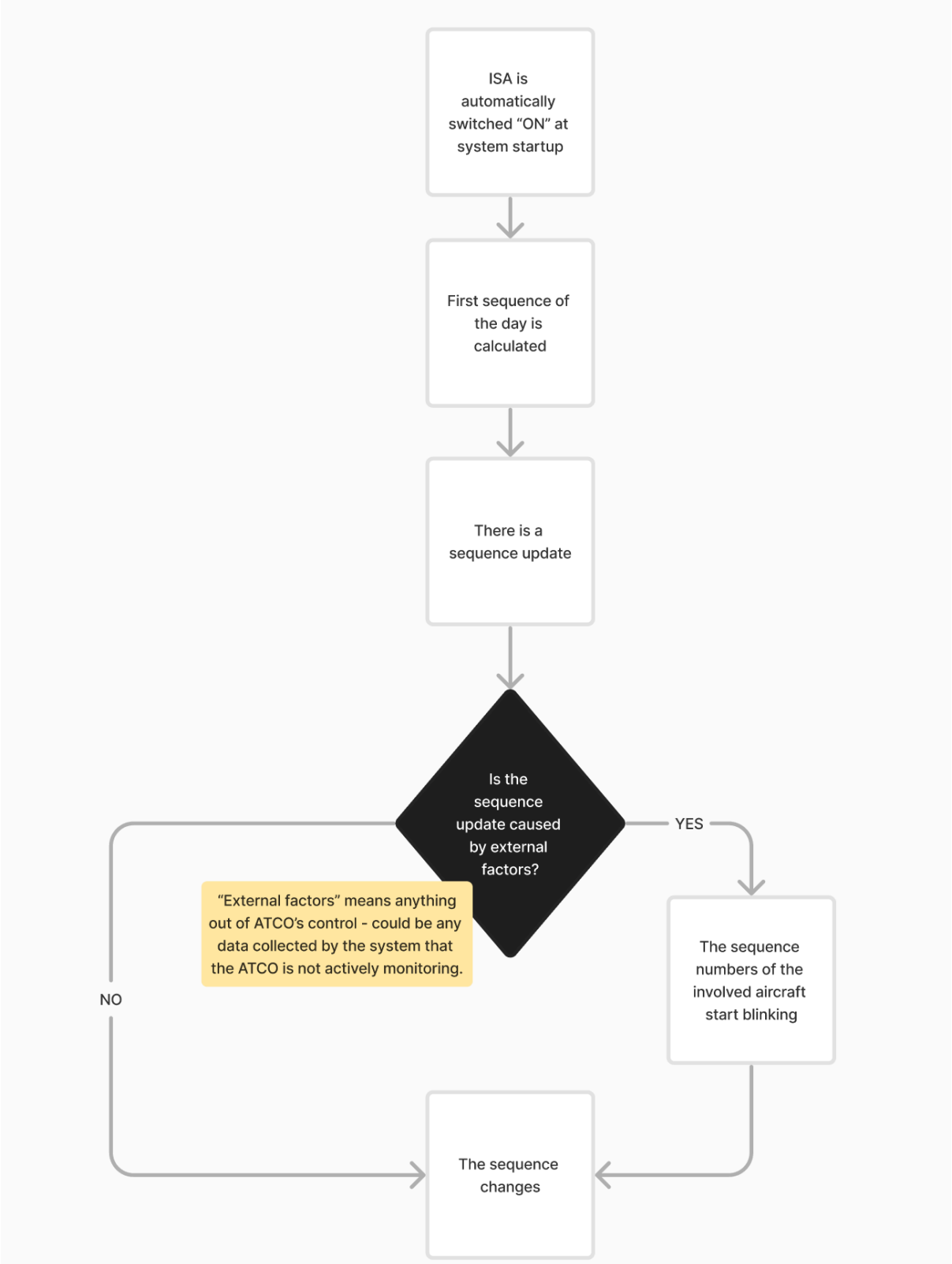

Figure 23: Sequencing procedure of ISA.

This project has received funding by the European Union’s Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

After VAL 1, it was noted that the data displayed in CLT 1 and CLT3 was not clear enough, and both pieces of the interface were redesigned to make it easier to consume the detailed information for the sequence change, per the CLT model. CLT5-6 were removed because all ATCOs during validation said that they are not interested in all the data concerning aircraft, as that is provided by other external organisations (i.e.: EUROCONTROL).
At the end of VAL 1 and the first redesign, the development process started. The overall interface is being developed by using Vue.JS, while the optimisation engine is based on Pyomo open-source libraries, and the data sink infrastructure is based on the MQTT protocol for data publication. The prototype for VAL 2 will include all these features and will be integrated into UFA systems for use in the simulator.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

4. Use Case 6: Airport virus spread prevention

The aim of the intelligent assistant in use-case 6 is to provide the airport passengers with a set of suggestions based on the likelihood of Covid-19 infection. The current TRL level is 2, starting from level 1 and aiming to reach level 6 by the end of the project. The components involved at this time are:

- **Passenger detection and tracking:** keeps track of ongoing passenger traffic in the airport using RGB camera and Lidar data.
- **Air quality application:** provides air-quality information to the user, based on several features such as Co2, temperature, humidity etc. The quality is classified into four classes: low**,** low to medium, medium to high, and high.
- **User interface:** A mobile application which allows users to select their preferred destinations and in turn provides them with the likelihood of covid infection and safest route to travel. Additionally, a chat bot which the user can engage with to get clarification on the suggestions generated by the intelligent assistant.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

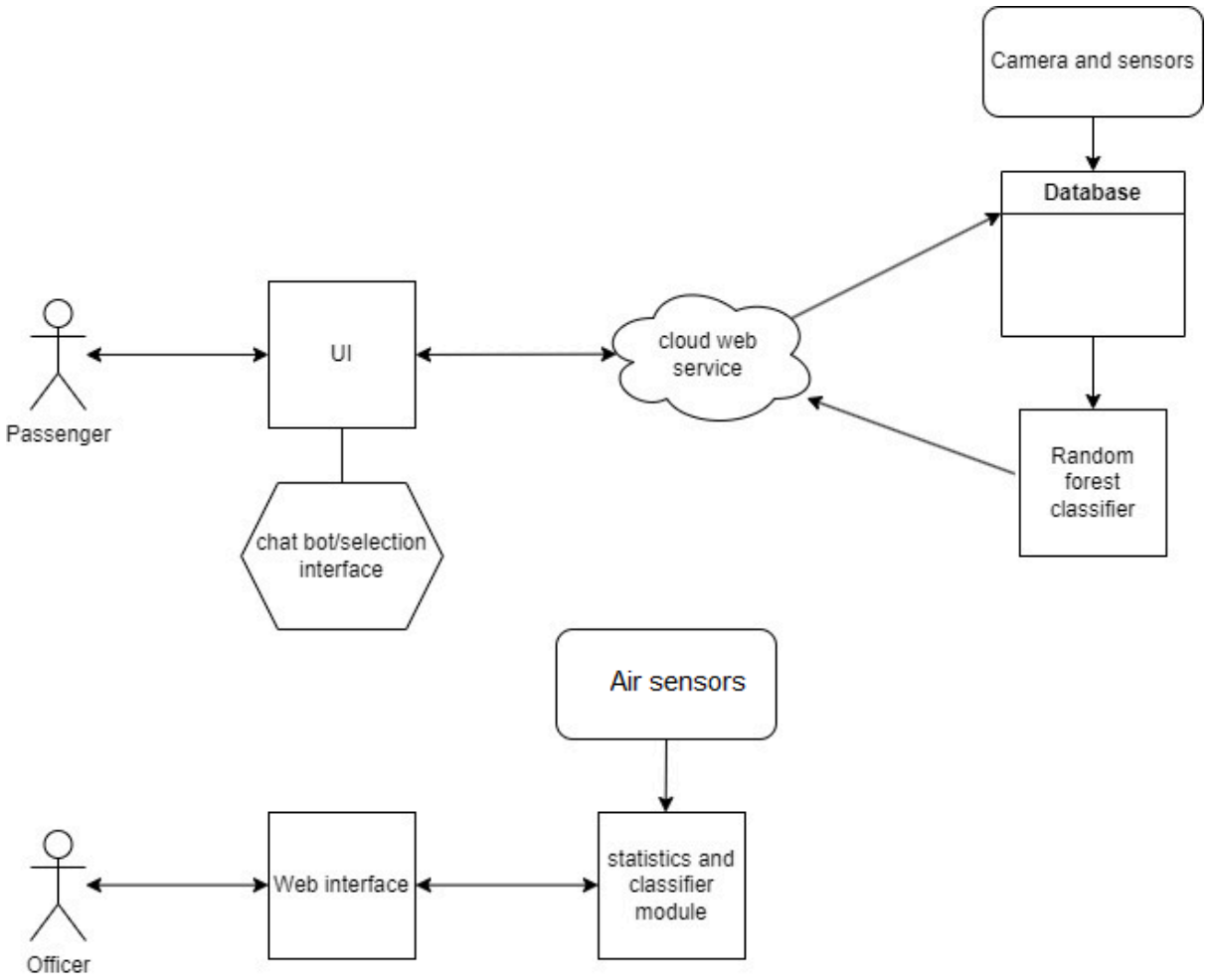

Figure 24:Architecture overview.

The architecture overview is displayed in Figure 24. The application is developed for Android systems with cloud web service support which contains the classification model and operations. Moreover, the airport safety officer has access to a web interface which displays the air quality data. To support the operation the following services are added:

- CSV creator for data formatting for the classifier.
- Weighting factor algorithm for the classifier.

For the routing process of the passenger in the airport common places, the prototype works in different phases. It starts with the passenger opening the Android application and they are shown the available common places to place their preferences. After performing this task, the passenger can move to the notification tab of the Android application where the likelihood of the COVID-19 infection is given as well as the priority of the common places to visit. The details of the operations are:

- The data from the selected common places is sent to a cloud infrastructure.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

- In the cloud application an implemented web service adds the preferences to the last record in the respective database.
- Based on the occupancy, queues, and preferences (how many people move towards each common space), a weighting factor outputs the priority of the common places to be visited. The queue of each common place comes from the developed wireless prototype, which records the visitors queuing outside each common place. The occupancy comes from the commercial cameras installed by person counting and by using the respective API provided. The likelihood of COVID-19 infection is provided by synthetic data to Random Forest classifier.

The chat bot on the passenger application provides information to them regarding their decision. This is based on the first and second levels of CLT for explainability which provides the routing and explanation on the chat tab. The safety officer is provided with basic statistics, air quality classification index, explanations which are based on the SHAP framework.

The data sources for the application are:

- COVID-19 likelihood: synthetic data is being used whereby the data has been devised in terms of common place coming, occupancy and queue, with a sequence of numbers and a pattern has been injected with some faults as well. This was for the model to identify the pattern, which shows whether the likelihood of infection is high or low.
- Air quality classification: online database.

  (https://github.com/twairball/gams-dataset).

- Occupancy: Acquired by the API of the commercial camera vendors, which outputs visitors coming in with respect to visitors coming out. This is the key data populating the database tables and especially the passengers in fields. For the time being a python script that places numbers to the database every two minutes is utilised. We are using the Yolo model[1] to detect the number of persons in the frame of the camera and produce a simple queuing model. A person counting algorithm which can successfully count the visitors going from left to right was developed. The data that we obtain is the number of the detections and we populate the database with the respective number every two minutes. This operation is real-time.
- Social distancing: via lidar, with the calculations currently done offline.
- Air quality: a sensor board was developed comprising the temperature, humidity, Volatile Organic Compounds(tVOC) and eCO2. The data from the sensor is obtained in real-time and populates a csv file as a row.

[1] **Redmon, J. et al., 2016. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 779–788.**

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

- Passenger preferences: the Android application provides the number of the common places to visit in terms of incrementing the database field that gives the visitors going towards this common place every two minutes. In this way, we can have a simple number that can be used for our calculations of the routing process in real-time.

In terms of model used for classification and AI the following were developed:

- Classification and weighting factor for the routing process.
- Chat bot knowledgebase.
- Classification of the air quality index.
- SHAP explainability for the air quality ML model.

The classification of the likelihood of COVID-19 is done by random forest classifier, yielding satisfactory results. It takes into account the occupancy, preferences and queues in the airport. The model is trained with a synthetic dataset with a pattern injected into it. The routing process is based on a weighting factor algorithm (logic based) for distinguishing the priorities of the common places. The chat bot is based on a knowledgebase, currently populated with a small number of Q&A. The bot may also learn to reply if the knowledgebase does not have the answer. Moreover, string similarity is provided for the system to recognize similar questions. The indoor air quality is also based on a random forest classifier, trained on a public dataset. The dataset is processed to identify the number of clusters that exist using the k-means clustering, which is expanded then by the cluster number as a column. The classifier outputs air quality index, having the choice of 4 states. SHAP framework was used to investigate the feature importance in the dataset with a re-defined model.

For the implementation of the HMI for the routing application, Android studio is used to develop a standalone application. Its architecture is quite simple, comprising a set of tabs that are used to navigate to the preference, notification, and chat respectively. For indoor air quality, Streamlit python library was used for plotting and displaying the results of the classification.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

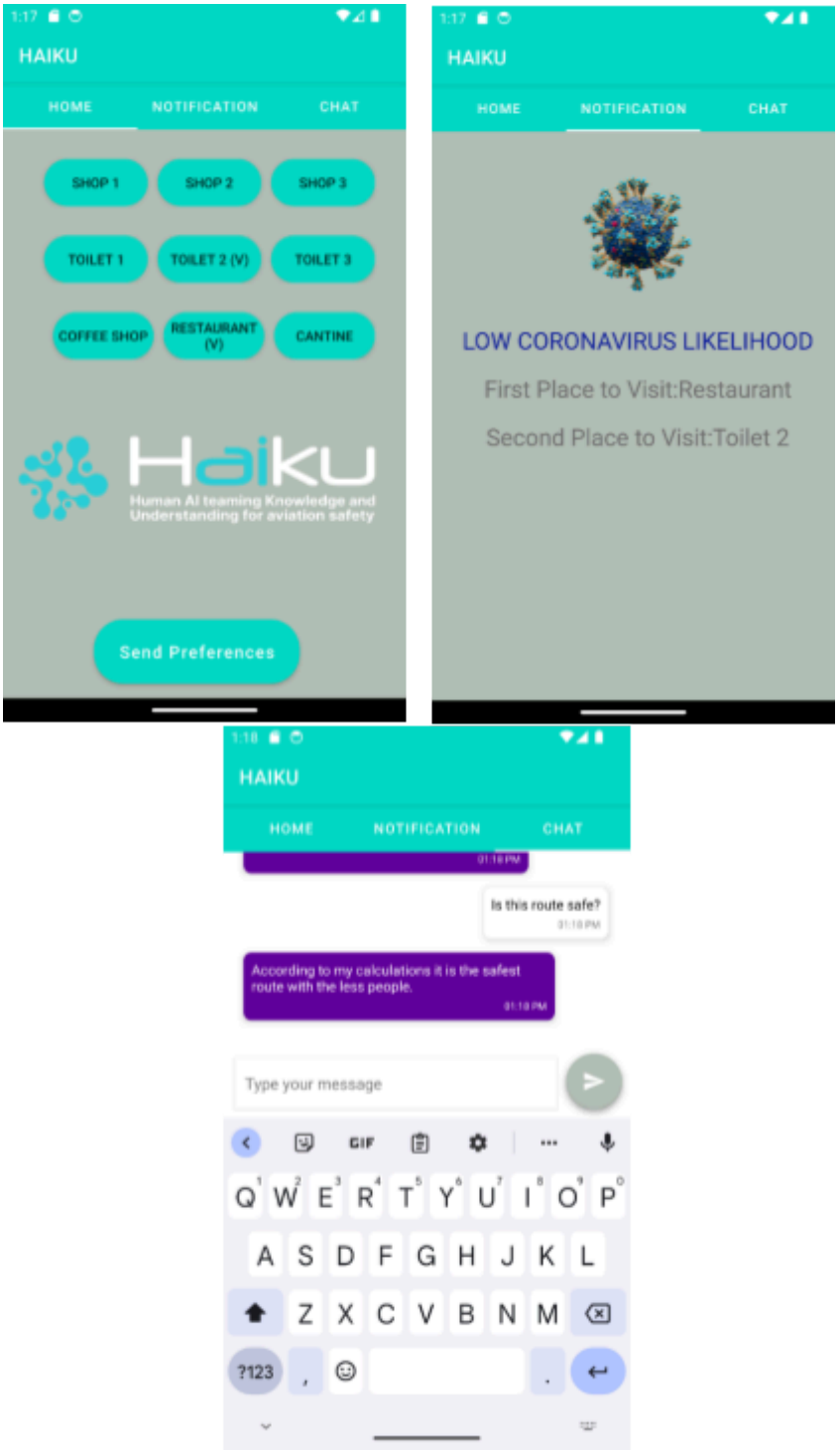

Figure 25: The UI interface for use-case 6. Top-left: the passengers can choose their preferred destinations and get recommendations from the assistant. Top-right: The passengers get notified about likelihood of covid infection based on their selection. Bottom: The passengers can chat with and ask for clarification from the assistant.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

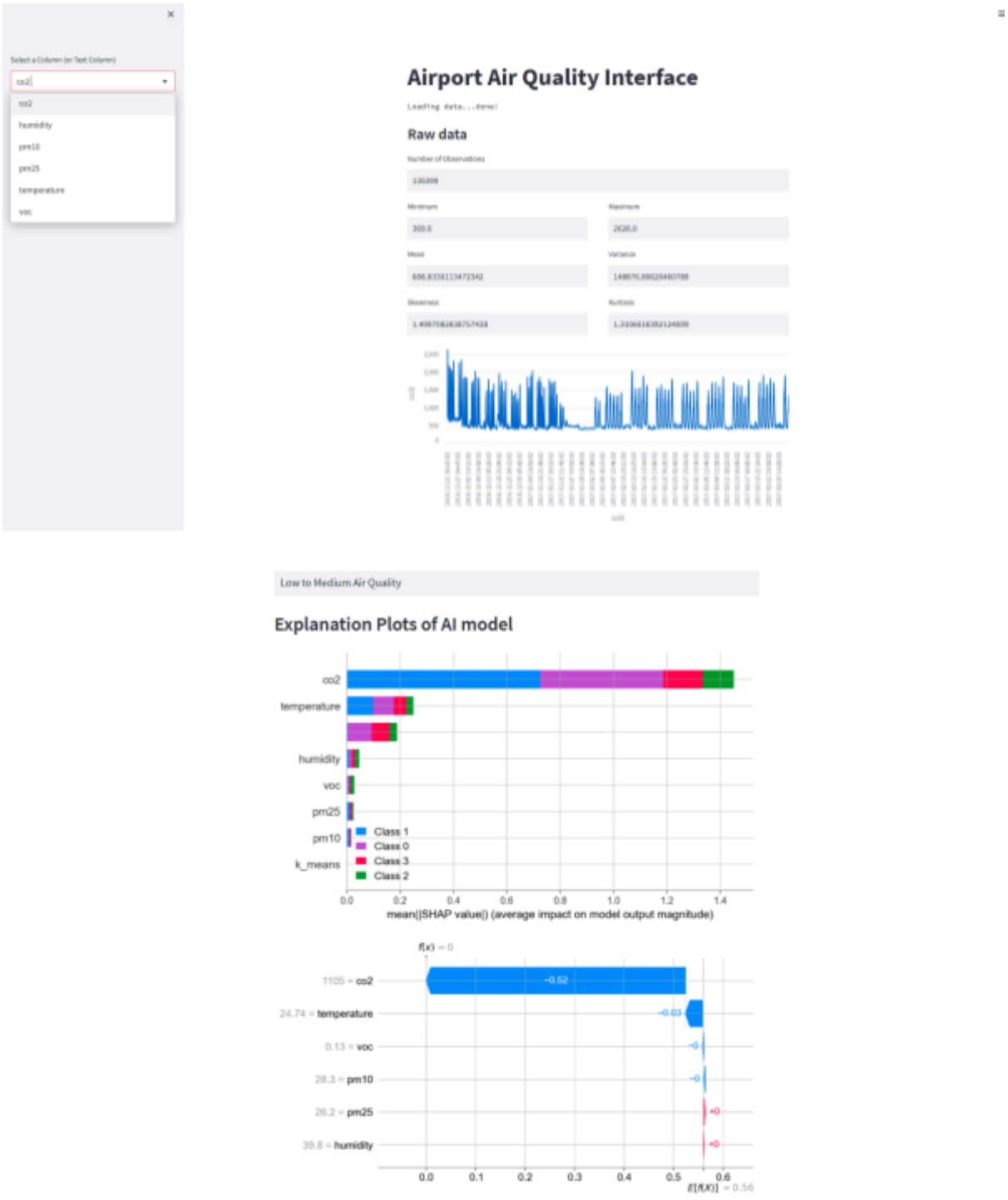

Figure 26: Air quality interface.

The main updates after VAL1 are captured below and are related to the AI components, data and sensors and the HMI.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Regarding the **AI components**, the algorithms developed for the AI infrastructure of the IA encompass four main tasks. The first task involves calculating the likelihood of COVID-19 infection in the airport's common areas. This probability is determined using artificial data processed by a Random Forest machine learning classifier. The classifier operates within the cloud infrastructure and evaluates three different metrics for each common area: occupancy, queue length, and the number of passengers heading towards it. These metrics are obtained from camera sensors and the Android mobile application, respectively. The occupancy data provides real-time insights into how crowded a space is, the queue length indicates waiting times and potential bottlenecks, and the number of incoming passengers helps predict future crowding trends. Together, these metrics allow the Random Forest classifier to generate a dynamic and accurate assessment of COVID-19 infection risk in various airport locations.

The second task focuses on optimising passenger routing through the airport to minimise infection risk. This involves real-time analysis and adjustment of passenger pathways based on the data collected and processed in the first task. The system can update the routing recommendations via the passenger request, directing passengers to less crowded areas, thereby reducing the likelihood of close contact and potential virus transmission. This adaptive routing algorithm, which is based on a weighting factor that resides in the cloud, considers the dynamic nature of the airport environment, ensuring that passengers are always directed towards safer routes. The proposed algorithm determines the weighting factors for each of the passenger's intended stores. If a more optimal sequence of visits is found, the routing is updated to guide the passenger to the next store with the lowest risk coefficient. The formula for calculating the weighting factor for each store is provided below. It is important to note that passengers need to recalculate the weighting factor for each store visit, as the data is dynamic.

$$F_i = \{\left(P_{i,w} + P_{i,g} + \frac{P_{i,in}}{P_{i,max}}\right) * T_i \quad P_{i,max} = P_{i,in}\left(P_{i,g} + \frac{P_{i,in}}{P_{i,max}}\right) * T_i - a \quad P_{i,max} > P_{i,in}$$

Where,

- $P_{i,w}$: is the number of passengers waiting to enter the store i
- $P_{i,g}$: is the number of passengers on route to the store i
- $P_{i,in}$: is the number of passengers inside the store i
- $P_{i,max}$: is the maximum allowed capacity of the store i
- T_i: is the time that one passenger spends in the store i
- a: is a negative constant (e.g., -10000)

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

The third task involves a chatbot that engages the passenger in a Q&A conversation regarding the routing procedure and the respective results. The chatbot is part of the Android application and has been implemented as a tab in the user interface (UI). The chatbot connects to the cloud every time the passenger asks a question or makes a comment in order to access the knowledge base. The knowledge base is essentially a CSV file with the format of Q&As. A Python script processes the string input by the passenger, sends it using an HTTP POST to the web service, and executes it to get the most appropriate response. The top-level schematic of the approach is given in the figure below (Figure 27):

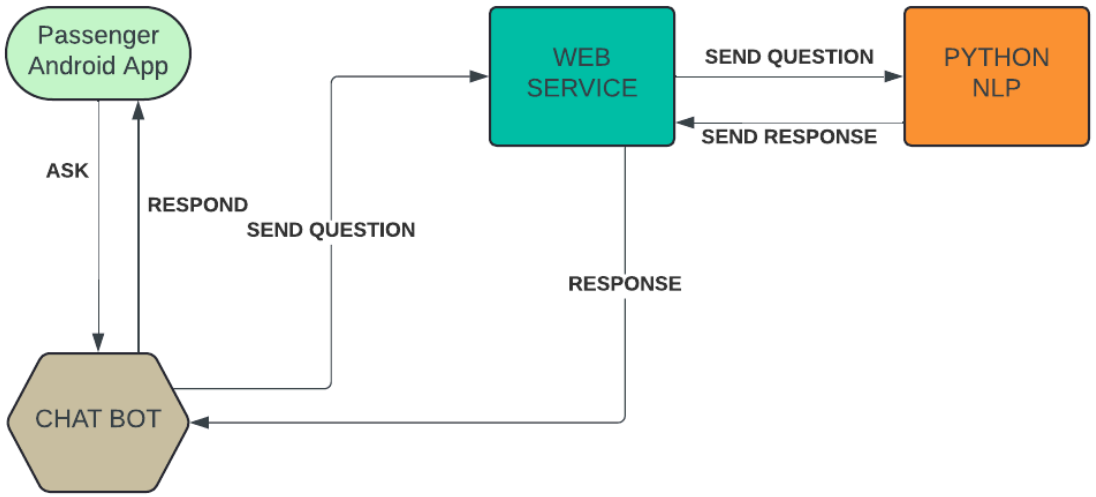

Figure 27: Chatbot component relation overview.

During its execution, the Python script obtains the similarity of the string before the comma using the Jaro-Winkler similarity, and it returns the appropriate response based on the calculated similarity score. This is achieved using the Jellyfish package available as a Python library. However, when integrated with the web service, the shell_exec PHP function did not accept the library, so we had to implement it from scratch. The Jaro-Winkler distance is particularly useful for matching names and titles, where initial characters are often more significant in determining similarity. The Jaro-Winkler similarity equation is given below:

$$sim_w = sim_j + lp(1 - sim_j)$$

The Jaro-Winkler similarity is calculated using the Jaro similarity for two strings s1 and s2, the length l of the common prefix at the start of the string (up to a maximum of 4 characters), and a scaling factor p. The value of p should not exceed 0.25 to prevent the similarity score from exceeding 1, with p=0.1 being the typical value used.

The chatbot response from the knowledge base includes two additional columns in the CSV file. The first column is a ranking value, which indicates how many people have asked a similar question or made a similar comment about the airport visit and routing

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

process. This helps in providing responses that reflect the majority opinion. For instance, if a passenger mentions that a toilet is not overcrowded, the AI will check how many others made similar comments to respond appropriately. Additionally, to handle questions or comments with negations (like "not"), the answer is designed to be generic, reflecting the majority opinion. Sensor data and information from the Android application can also be provided for further clarification.
Another aspect of the algorithm is the last column of the CSV file, which records the time when the response was taken. The script checks the time of the HTTP request and sends the date and time. This allows the AI to determine if the request was made within the last 10 minutes. This time boundary is selected due to the dynamic nature of the application, which needs timely updates rather than just responding to static requests. For example, if a passenger asks if a toilet is overcrowded, the AI should use data from the past ten minutes rather than older information. This column is updated every ten minutes to maintain the application's relevance.
The algorithm for finding the closest match in the chatbot is shown in Figure 28. It is fully executed in the cloud, with responses sent to each passenger's Android phone. The algorithm is kept simple to handle a large number of passengers efficiently, even during peak times.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Algorithm 1 Chatbot Algorithm

Require: $String = q$
Ensure: $x = 0$
Ensure: $ranking = 0$
Ensure: HTTPPOST
Ensure: $highstJaroWink = 0$
0: $pythongetResponse.pyString$
0: **for** i, row in $iterRows$ **do**
0: $currentRow = row['question']$
0: $currentScore = jaroWrinker.similarity(x, String)$
0: **if** $currentScore > highestJaroWink$ **then**
0: $highestJaroWink = currentScore$
0: $ranking = row['rank']$
0: $bestMatch = row['answer']$
0: $answList.append(bestMatch)$
0: $theList.append(highestJaroWink)$
0: $theList1.append(row['question'])$
0: $theList2.append(ranking)$
0: **end if**
0: **end for**
0: **for** k in $range(len(theList))$ **do**
0: **if** $k = 0$ **then**
0: $temp1 = theList1[0]$
0: $rank1 = theList2[0]$
0: **else**
0: $temp1 = theList1[k]$
0: $rank1 = theList2[k]$
0: **if** $temp1 == temp2$ **then**
0: **if** $rank1 < rank2$ **then**
0: $bestMatch = answList[k - 1]$
0: **else**
0: $bestMatch = answList[k]$
0: $rank1 = rank2$
0: **end if**
0: $temp1 = temp2$
0: **end if**
0: **end if**
0: **end for**

Figure 28: Closest match algorithm.

We also developed a separate algorithm to check if the passenger's remark is a comment. If there is no match in the knowledge base, we generate the corresponding question in proper English. If an answer exists, its ranking is incremented. We use the SpaCy library, specifically "en_core_web_sm," to identify verbs and check for passive voice to determine if the question should start with "Is" or "Does." We then split the comment based on the verbs and construct the question from the answer, adding the new Q&A to the knowledge base. Since the PHP shell_exec function did not support the libraries, we implemented them from scratch.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Additionally, we set timers in our Android application to provide CLT levels. This data comes from the HTTP request for the routing process, and we implemented automatic chatbot responses to inform passengers about the routing process during their journey.
The NLP model needs to be more thoroughly tested, and the knowledge base needs to be populated with more Q&As. In the near future, we will connect the Q&As with camera data. For example, if a passenger asks whether Toilet 1 is busy, the NLP will provide the answer using real-time data from the database.
For the fourth task, we built an air quality ML model to assist airport health and safety officers. This model resides on the cloud infrastructure and is integrated into a web application. It provides officers with statistics, air quality classifications, and explanations. We found an indoor air quality dataset from https://github.com/twairball/gams-dataset, which includes seven features: datetime, CO2, humidity, PM10, PM2.5, temperature, and VOC. The dataset contains 135,100 values.
To categorize air quality based on these measurements, we used the k-means algorithm and determined the optimal number of clusters using the elbow method. We then reconstructed the dataset with the respective variables and ran the Random Forest algorithm to detect the air quality index. The model achieved an accuracy of 1 during training. We added explainability using the SHAP method to generate plots showing the significance of each feature. The top-level scheme of the air quality ML model is shown in the figure below (Figure 29).

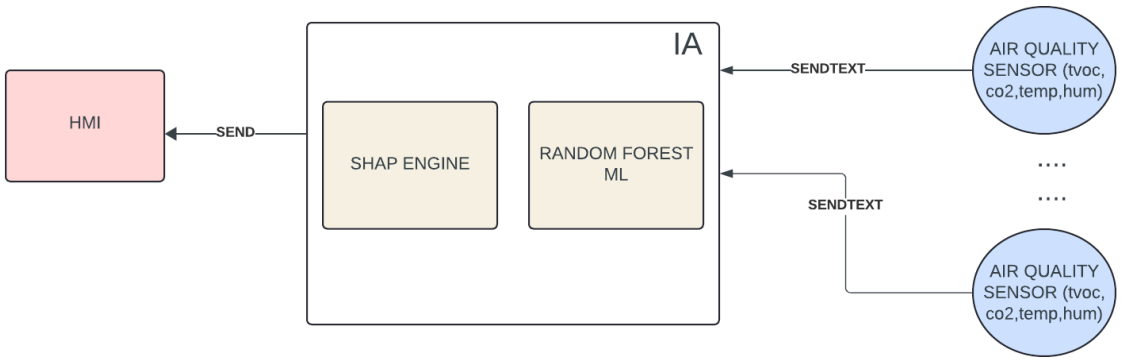

Figure 29: Air quality block diagram.

As for **sensors**, three cameras have been purchased from FootfallCam. These cameras offer good coverage as they detect within a Region of Interest (ROI) that covers a significant part of the door or hallway area. The cameras were selected based on the ceiling height at Amygdaleona Airport in Kavala, Greece. The company provides an API for obtaining occupancy and live data from each device using its

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

unique internal static IP. The devices are connected via Ethernet and can also be managed wirelessly through their unique IPs.
By writing a simple Python script, we can populate the database with the occupancy data of monitored areas. The script performs an HTTP request to obtain a JSON file. The inbound and outbound data is then extracted, and occupancy is calculated using subtraction. The data is stored in the database. The script pauses for two seconds before repeating the process. We decided to retain occupancy data for two consecutive runs to maintain a satisfactory sample size. The cameras have been placed in key locations at Amygdaleona Airport where there is frequent foot traffic, to capture as much movement as possible (Figure 30).

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Figure 30: FootFall Camera Placement..

Additionally, we installed a fourth camera, purchased from V-Count, at the airport premises. This company also provides an API that allows us to obtain occupancy data with the desired time granularity. Using a simple HTTP request, the occupancy data is retrieved and stored in the appropriate field of the respective database table in the cloud. The installation can be seen in Figure 31.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

Figure 31: V-Count camera installation.

Moreover, the queues at each monitored location need to be tracked. To achieve this, a wireless camera prototype was proposed. A mini PC, the Lenovo ThinkCentre M910q Tiny Desktop, was selected, equipped with an Intel Core i3-8100T CPU at 3.10 GHz, 8 GB of RAM, and a capacity of 237 GB. The software used for testing runs in the Windows 10 environment. A Logitech BRIO 500 webcam was used, featuring 1080p/30fps (1920×1080 pixels), 720p/60fps (1280×720 pixels), 4 megapixels, a dFoV of 90°/78°/65°, 4x digital zoom, and USB-C connectivity. The camera was deployed on the mini PC and recognized by setting the appropriate software variable on-the-fly (Figure 32).

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

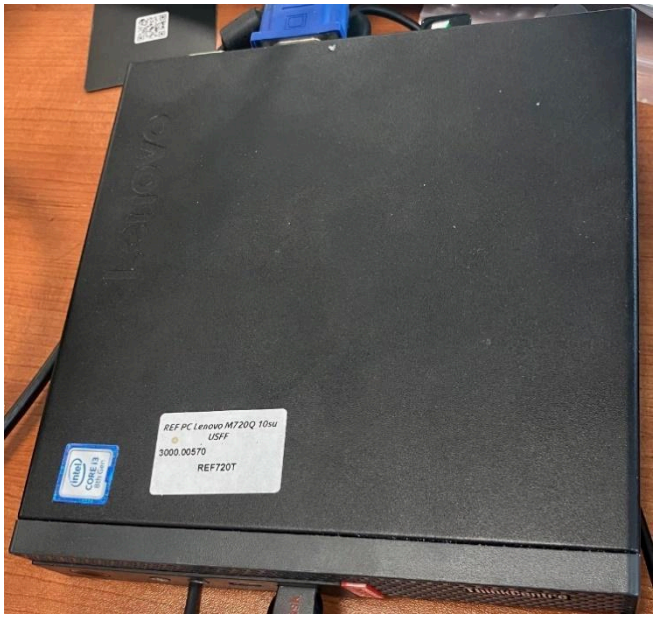

Figure 32: Mini PC and USB camera.

In terms of software libraries, OpenCV of the Python programming language has been used, in conjunction with Torch, SciPy, Torchvision, and TensorBoard. The Torch library, in particular, can only be utilised in 64-bit systems and is an open-source machine learning framework that accelerates the path from research prototyping to production deployment. OpenCV is the computer vision library used to manipulate images and videos. A Yolov8 model is used to detect people, and the number of people is obtained and stored in the database. Since the Yolov8 algorithm is implemented in Python, a simple addition to the code was made to send the data to the database server. In terms of installation, there was an issue with the power required to run the prototype; hence, an electrician was summoned to provide a power supply. The prototypes will be installed near the commercial camera very soon.

To build the indoor air quality sensor, a relatively cheap and scalable solution was needed to form a wireless sensor network. Solutions with microcontrollers (MCUs) were investigated for this purpose. One of the most popular MCUs in the market is the Raspberry Pi 4 Model B 8GB. This device offers hats for I2C and other connections, as well as USB ports for connecting other units. For temperature and humidity sensors, an integrated sensor that would be plug-and-play with the Raspberry Pi was sought. This research led to the TEMPerX232 sensor, a comprehensive USB thermometer with built-in temperature and humidity sensors. Through its expansion interface, it can connect other probes, such as the DS18B20, TX, and HS10 probes. The USB supports both HID mode and USB to COM port mode and includes open-source software to access the sensors and obtain readings.

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

The most challenging task was identifying the best tVOC and eCO2 sensors. Research concluded that the Grove-VOC and eCO2 Gas Sensor (SGP30) would be utilised, serving as an air quality detection sensor. This specific Grove module, built upon SGP30 technology, offers Total Volatile Organic Compounds (TVOC) and CO2eq output. The SGP30 is a digital multi-pixel gas sensor designed for seamless integration into various applications such as air purifiers, demand-controlled ventilation systems, and IoT devices. Leveraging Sensirion's CMOSens technology, the sensor system is consolidated onto a single chip, featuring a digital I2C interface, a temperature-controlled micro hotplate, and two preprocessed signals indicating indoor air quality. Distinguishing itself as the first metal-oxide gas sensor with multiple sensing elements on a single chip, the SGP30 delivers more intricate details about the surrounding air quality.
To connect the SGP30 to the Raspberry Pi, a hat with the appropriate connector is required. Thus, the Grove Base Hat was obtained. The Grove Base Hat for Raspberry Pi provides Digital/Analog/I2C/PWM/UART ports to meet all needs. With the help of a built-in MCU, a 12-bit 8-channel ADC is also available for the Raspberry Pi (Figure 33).

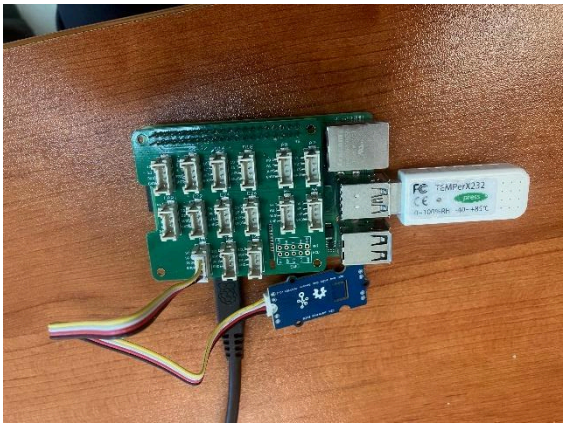

Figure 33: Indoor air quality sensorboard.

Open-source libraries for the sensors were utilised to obtain readings. The data is retrieved in a comma-separated format and then transmitted to the database using simple networking connection Python scripts.
For the lidar sensor, the Velodyne VLP16 will be used for distance measurement, successfully obtaining the distance between two people using the pcap file it generates.
As stated before, data from the Android phone and the cameras are used to produce the likelihood of infection using the ML model and to determine the routing sequence using the weighting factor, both of which reside in the cloud. The database is populated by camera readings and passenger preferences from the Android application. This is a dynamic process, as the numbers change according to the volume of passengers in the airport. Passenger input also makes the system more

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332

dynamic. The results are shown on the Android HMI. Additionally, passengers provide input through Q&As or comments in the chatbot. If the input is a question, it is not kept; however, under specific conditions, a comment may generate a new question that is stored in the knowledge base.
Data from the air quality sensor system is fed to the ML algorithm for air quality classification, and the results are shown on the respective HMI.
The number of users will be the maximum that Amygdaleona Airport in Kavala can accommodate. If some passengers do not have an Android phone, devices will be purchased and provided to them to increase the number of participants. Gender bias is not applicable in this use case since everyone can use the tool.

As for **HMI**, not much has changed since the previous submission (Figure25). CLT 2 and 3 have been added for explanation to the Android Chatbot (Figure 34).

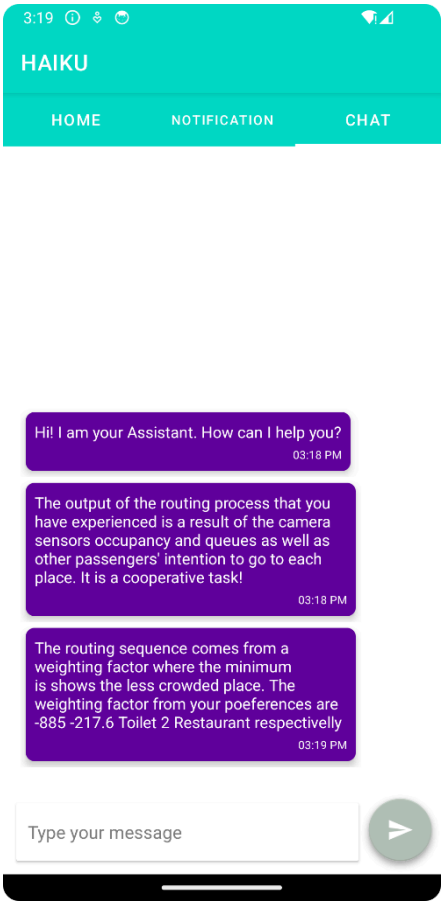

Figure 34: CLT 2 and 3

This project has received funding by the European Union's Horizon Europe research and innovation programme HORIZON-CL5-2021-D6-01-13 under Grant Agreement no 101075332