Technical Report - **Product specification**

A.T.L.A.S.

Advanced Tracking Legendary Automated Spaceship

Course:	IES - Introdução à Engenharia de Software
Date:	Aveiro, 17/12/2024
Students:	114137: Diogo Fernandes 113585: Henrique Oliveira 122949: Mateus Rocha 113736: Raquel Vinagre
Project abstract:	A.T.L.A.S. serves as a real-time monitoring system for a spaceship. It gathers and displays essential spaceship metrics and provides clear data visualization, with an alert system for any anomalies.

Table of contents:

1 Introduction

A.T.L.A.S. is a software application developed for the efficient monitoring, management, and analysis of spaceship operations. The app leverages real-time tracking of data, emitting alerts that are useful for the users. With a Greek mythology-based name, A.T.L.A.S. revolutionizes space exploration.

2 Product concept

Vision statement

Functional Description

ATLAS is an innovative software application designed to monitor and manage spaceship operations in real-time. The system will be used for:

- **Real-Time Tracking**: Monitoring the spaceship's position, speed, and trajectory.
- **Resource Management**: Overseeing critical onboard resources such as water, energy, and oxygen, ensuring optimal usage.
- **Performance Analysis**: Power consumption, and other vital metrics to enhance efficiency and identify potential issues early.
- **Crew vitals monitoring**: Checking crew members' vitals, such as heart rate, body temperature, oxygen levels (...), ensuring tripulation well-being.
- **Environmental Monitoring**: Displaying data on external conditions, such as spaceship position (altitude, apogee, range...)
- **Hazard Warning**: Relaying all pertinent information to the crew, so they can take action according to changes in the environment.
- **Statistics Report:** Full report on data gathered sent back to HQ engineering and scientist team in real time.

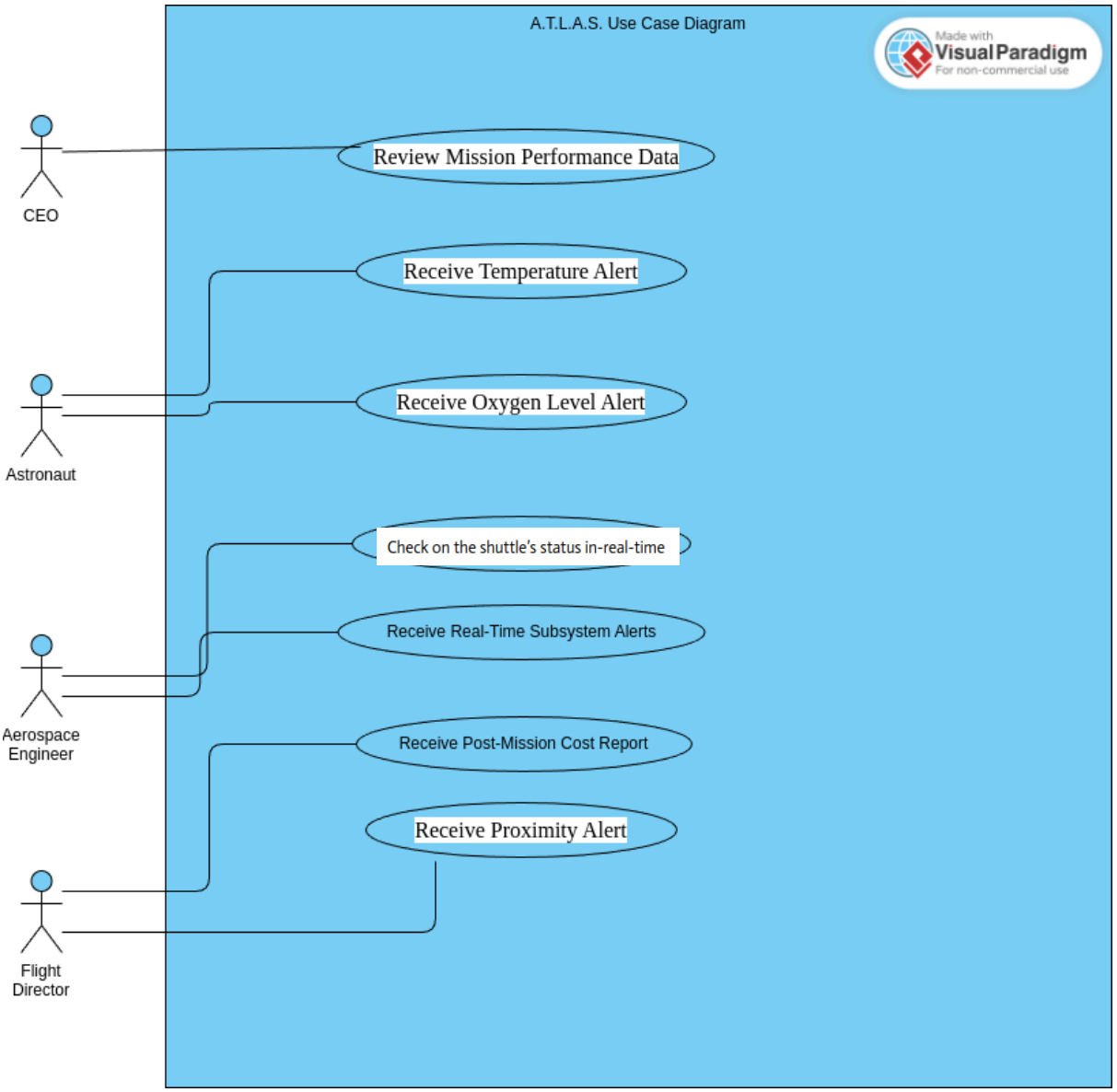
Planned Adjustments

Initially, the concept included a broader focus on the system where the users could control the ship from the software. However, the team decided to concentrate on more of a monitoring system as these are simpler to implement, test and refine, before applying them to broader systems. This pivot allows for targeted development and faster implementation.

Comparison to Other Well-Known Products

While there are existing spacecraft monitoring systems, such as NASA's Mission Control and the European Space Agency's operations systems, ATLAS distinguishes itself by offering a **User-Friendly Interface**. Unlike traditional systems, ATLAS provides an intuitive dashboard that simplifies complex data presentation.

UML Use Case Diagram



Requirements Gathering and Selection

Expert interviews: Discussions with aerospace engineering students about application features.

Personas and Scenarios

Persona 1.

Melon Usk is a 56 year old man, and is the **CEO** of a multimillionaire space company that focuses on exploration missions. The company's main goal is finding life-based planets, so that humans may become a multi-planetary species. With an early interest in space, Usk started a degree in Mechanical Engineering, but ended up dropping out in his third year, as he realized he did not have to be an engineer to be able to put people in space. Instead, he got a degree in Business and Finance, and founded YSpace. Melon is married to Walter Usk, who also works at the company. Walter, Melon, and everyone else at YSpace are focused on their mission to explore the universe and its mysteries.

Motivation: Usk would like to increase profit for his company, as well as accelerating the process of getting civilians to live in space. ATLAS's smart system leverages efficiency and cuts back on human and financial resources, both of which are important for a CEO.

Persona 2.

Juan Direction is a 33-year-old specialized field doctor who transitioned into becoming an **Astronaut** due to his passion for space exploration. He has extensive knowledge of spacecraft systems, engineering principles, and the mechanics of space travel. As a key member of YSpace's astronaut team, Juan is responsible for monitoring crew members' vitals and well-being.

Motivation: Juan is focused on ensuring the crew stays healthy throughout the whole mission. For that, Juan needs to be able to see the crew's vital statistics so he can respond quickly to any anomaly and intervene if

necessary.

Persona 3.

Lua Dipa is a 27 year old bright **Aerospace Engineer** that currently works at YSpace. With an astonishing mind, covered with brilliant ideas, Dipa is one of the most renowned engineers in her sector. She's been making her name ever since she was still in college, receiving job offers from multiple companies, including NASA. However, this young mind's dream has always been to work at YSpace. Why? In a recent interview, Dua answered this famous question: "Well, the answer is in the question (...) everytime you ask me 'Why YSpace?', it just sounds like you're stuttering and I think that's very funny." Known for her sense of humor, Lipa jokes about the topic, avoiding a real answer. At least, that's what most think.

Motivation: Lua would like to receive real-time informative information about the spaceship. She is also responsible for maintaining the spacecraft during missions, conducting repairs, and ensuring that all systems function optimally. With the help of a smart system, she can find solutions to recurring problems in a more efficient way.

Persona 4.

Mona Luísa is a 42 year old **Flight Director**. She grew up near Houston, Texas, inspired by NASA launches. She studied at MIT and became one of the few female Flight Directors in the industry. Known for her calm leadership and technical expertise, she's a key figure in space exploration. Mona oversees space missions, coordinating between engineers, scientists, and astronauts. Her responsibilities include mission planning, real-time decision-making, managing flight controllers, and ensuring crew safety. She leads with authority but encourages team input, especially during high-pressure situations. Mona balances her demanding career with raising 10-year-old twins. She enjoys outdoor activities, especially stargazing with her family.

Motivation: Mona's main drive is to streamline information flow during missions. She relies on software that condenses vast amounts of data into actionable insights, enabling her to make swift, informed decisions and ensure mission success.

Product requirements (User stories)

Story 1.

Title: Juan Direction should be able to receive an alert on low or high temperatures

Description:
As Juan Direction, I want to receive an alert when the spaceship endures dangerously high temperatures, in order to implement countermeasures.

GIVEN the existence of a temperature sensor outside the ship
WHEN the temperature measured reaches critical levels
THEN I should be able to see an alert pop up

Scientist Notes: Thresholds to be set for min and max tolerable temperature

Design Notes: The warning should pop up on the ship's HUD panel, in a visible way (given it's an important warning)

Story 2.

Title: Mona Luísa should be able to, based on a specific time period, get the summarized data of the spaceship.

Description:

As Mona Luísa, I want to be able to see a report of the summarized data of the spaceship, so I can better plan future expeditions and propose pertinent expenses to CEO

GIVEN the tracking system monitors mission status, like sensor data.

WHEN a time period is given

THEN I should receive a detailed "report" summarizing the data that was received from the sensors

Story 3.

Title: Mona Luísa can view the ship's route and trajectory warnings

Description:
As Mona Luisa, I want to receive alerts about dangerous proximity to other bodies in advance by the system, in order to react faster to these threats and reroute, to ensure the safety of the crew.

GIVEN the system being programmed to notify the Flight Director on dangerous proximity to alien bodies
WHEN the system picks up said dangerous proximity
THEN a warning will pop up on the Flight Director's screen

Dev Notes: Make sure to reroute information gathered by the sensors to the Flight Director's machine

Design Notes: Provide an informative interface for the Flight Director to see these warnings.

Story 4.

Title: **Juan Direction should be notified on crew members' low oxygen levels**

Description:

As Juan Direction, I want to receive an alert when the crew members' oxygen levels drop to a critical point, so I can take immediate action to avoid potential harm.

GIVEN the presence of an oxygen monitoring system in the crew spacesuits
WHEN the oxygen levels fall below a safe threshold
THEN I should be notified with an alert on the HUD and a distinct audible alarm

Scientist Notes: Set threshold values for minimum oxygen levels to trigger the alert, ensuring sufficient time for countermeasures

Design Notes: The alert should be highly visible on the HUD and distinct from other alerts, possibly with a flashing icon and sound to ensure quick reaction.

Story 5.

Title: **Mona Luísa should be notified of potential communication system failures**

Description:

As Mona Luísa, I want to receive early warnings of potential communication system failures so that I can notify the crew and implement necessary precautions to maintain contact with the spacecraft.

GIVEN the spaceship's communication systems being continuously monitored
WHEN the sensors detect anomalies or signs of degradation in the communication systems (e.g., signal interference, power issues)
THEN I should receive an alert with detailed information about the issue, including its severity, on the mission control console

Dev Notes: Establish thresholds for signal degradation and system performance to trigger the warning

Design Notes: The alert should be visible on the mission control interface.

Story 6.

Title: **Lua Dipa should receive real-time alerts on subsystem anomalies during the mission**

Description:

As Lua Dipa, I want to be alerted when any subsystem of the spaceship experiences anomalies or malfunctions during the mission, so I can collaborate with the crew to address potential issues and ensure mission success.

GIVEN a network of sensors monitoring the health of critical subsystems (e.g., propulsion, navigation, life support)
WHEN a sensor detects an anomaly, such as performance degradation or malfunction
THEN I should receive a real-time alert with detailed information on the affected subsystem, allowing me to advise the crew on potential corrective actions

Dev Notes: Sensor data should be continuously transmitted and recorded, with alerts triggered when readings go beyond predefined thresholds. Historical data should be accessible for post-mission analysis.

Design Notes: The alert system should be highly visible in mission control, with clear,

actionable information on the type and severity of the issue.

Story 7.

Title: **Mona Luísa wants to know the evolution of the time a message from HQ takes to reach the spaceship.**

Description:

As Mona Luísa, I want to be able to see the response time of periodic test messages to the spaceship, so that I know if there is trouble in reaching the spaceship or not.

GIVEN periodic messages sent to the spaceship
WHEN the ship receives said messages
THEN I should be able to see clearly how the response time changes over time

Design Notes: Create a visually clear dashboard with a timeline of response times and ensure the interface is user-friendly for technical and non-technical stakeholders.

Story 8.

Title: **Mona Luísa should be able to filter the Message Log by sender, to find something a specific astronaut said during the mission**

Description:

As Mona Luísa, I want to filter the message log by sender, so that I can quickly find specific communications made by a particular astronaut during the mission.

GIVEN a log of mission messages stored in the system
WHEN Mona Luísa selects a specific sender from the filter options
THEN only the messages from the chosen sender are displayed in the message log.

Story 9.

Title: **Mona Luísa should be able to see in-real-time information about everyone on the ship**

Description:

AS the Flight Director, I want to monitor the real-time vital signs of everyone on the ship, so I can ensure the crew's health and safety during missions.

GIVEN that the ship is equipped with biosensors monitoring crew health
WHEN I access the system's health dashboard
THEN I should be able to view live updates on each crew member's heart rate, oxygen levels, stress indicators, and other critical vitals, with alerts for any anomalies.

Story 10.

Title: **Lua Dipa should be able to check on the shuttle's status in-real-time**

Description:

AS Lua Dipa, I want to be able to check on the spaceship's status, so I can make sure that everything is going as planned.

GIVEN information gathered by the sensors, transmitted in real time
WHEN there are weird influx of data / dangerous data
THEN decide on what upgrades are necessary based on said data

Story 11.

Title: Melon Usk can see a spaceship report in form of a PDF

Description:
As Melon Usk, I want to see a report with information regarding the spaceship's system and alerts received throughout the mission in form of a PDF or other readable format

GIVEN a spaceship and information being generated over time
WHEN information is gathered and a report is created
THEN I should be able to see that report, in PDF format

Design Notes: Output the ship's information in a readable way, easily understandable by minimal technology-savviness workers

Story 12.

Title: Melon Usk can find specific data regarding aspects of the system

Description:
As Melon Usk, I want to be able to find specific information regarding one of the system's sensors (i.e. check the evolution of the cabin temperature over time)

GIVEN a spaceship and information being generated over time
WHEN a specific sensor data is selected
THEN there should be a specific feedback of information regarding the selected sensor

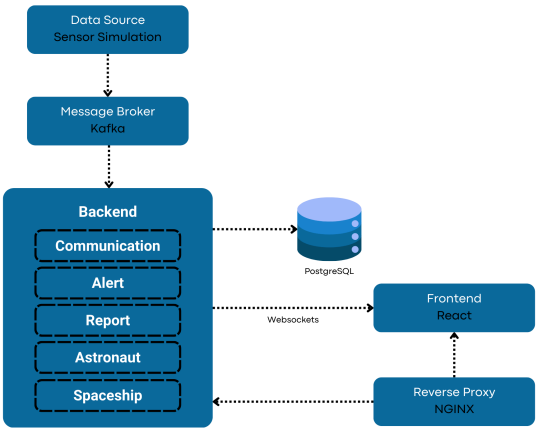
Dev Notes: Possibly implemented via the usage of dynamic graphs, portraying the evolution of the selected sensor info over time.

3. Architecture Notebook (Updated)

Key Requirements and Constraints

- Since A.T.L.A.S. is meant to work under extreme conditions, for long periods of time, it must be reliable and robust for long-term maintenance.
- In case of communication failure, the system will lose contact with HQ but must guarantee 100% functionality inside the spaceship, allowing the crew to continue acting based on sensor information.
- In case of database failure, the system would lose all stored information but must guarantee 100% functionality inside the spaceship. For this reason, data should first be processed and posted, then saved to the database.
- The system must handle large amounts of real-time telemetry data from the spaceship, requiring a scalable architecture.

Architectural View



The A.T.L.A.S. project architecture focuses on scalability, resilience, and efficiency. By using Kafka as a message broker, the system handles real-time, large-scale telemetry data streams asynchronously. PostgreSQL is utilized for data persistence, ensuring reliable storage and retrieval for analysis. NGINX ensures secure, optimized communication between the frontend and backend through request routing. The modular backend architecture organizes the system into functional domains, enhancing maintainability and simplifying deployment.

Main components:

Data Source

Simulates spaceship sensors, generating continuous data such as temperature, fuel levels, and system health metrics. The data is sent to Kafka.

Message Broker

Kafka is used for real-time data ingestion, ensuring efficient and secure transportation to the backend.

Database

PostgreSQL manages persistent data storage, including user information, spaceship data, telemetry, and system messages.

Backend

Built with Spring Boot, the backend is organized in modules by functionality:

- **Astronaut**
- **Spaceship**
- **Communication**
- **Alerts**
- **Report**

Frontend

Built with React, it displays real-time data and provides an intuitive interface, including dynamic graphs and control panels.

Reverse Proxy

NGINX is configured for routing and reverse proxy, improving communication between the frontend and backend.

Service Interactions:

Backend Service

Purpose: Handles the core business logic, data processing, and communication with other services like Kafka, database, and frontend.

Interaction: Manages REST endpoints for frontend communication, processes data, and interacts with the database.

What it should do:

- Expose RESTful APIs for frontend communication.
- Communicate asynchronously with Kafka to process telemetry data.
- Interact with the database to store and retrieve data.
- Push real-time updates to the frontend via WebSockets.

Frontend Service

Purpose: Provides the user interface for interacting with the system.

Interaction: Consumes REST APIs from the backend and receives real-time updates via WebSockets.

What it should do:

- Display data and status updates from the backend.
- Provide user interfaces for astronaut and spaceship status, and telemetry monitoring.
- Handle real-time data streaming via WebSockets to provide live updates to users.

Kafka Service

Purpose: Manages the messaging and event streaming between services, especially for telemetry data.

Interaction: Sends and receives messages from the data generator for real-time telemetry data processing.

What it should do:

- Listen for telemetry data from the data generator and publish it to backend services.
- Ensure that telemetry data is processed in a timely manner.
- Act as a message broker for asynchronous communication between services.

Spaceship Service

Purpose: Maintains and monitors spaceship status, such as location, fuel levels, and hull integrity.

Interaction: Retrieves telemetry data and other metrics from the database to present the current spaceship status.

What it should do:

Provide real-time spaceship status updates.

Maintain a historical log of spaceship data for analysis.

Data Generator

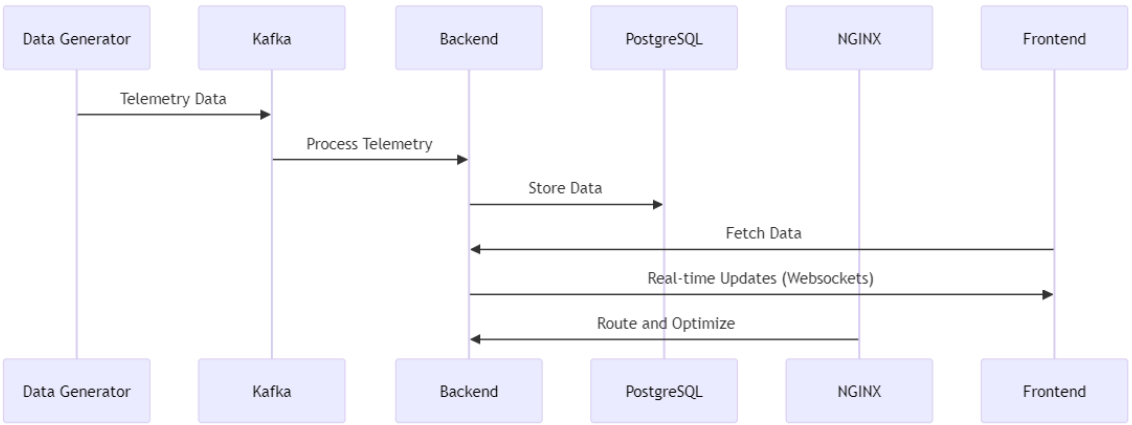
Purpose: Simulates telemetry data, such as sensor readings, to provide input to the system.

Interaction: Pushes simulated telemetry data to Kafka for processing by the backend.

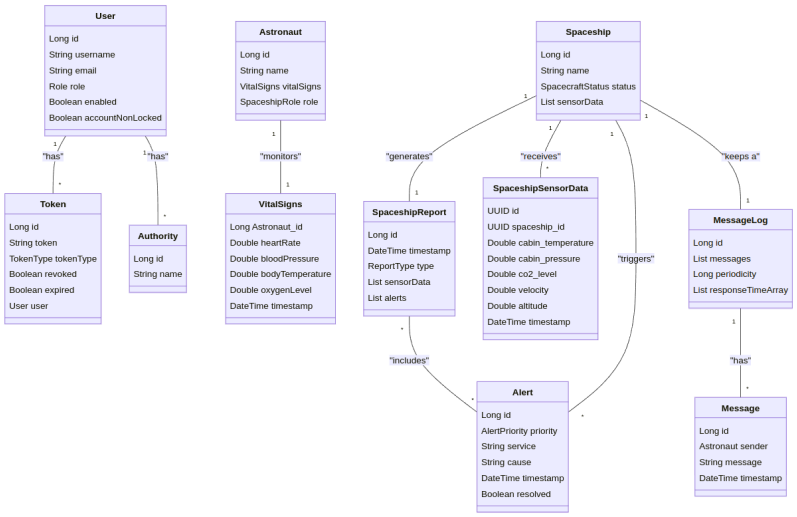
What it should do:

- Generate simulated sensor data (e.g., spaceship metrics, astronaut health).
- Publish telemetry data to Kafka for consumption by the backend.
- Ensure data consistency and simulate real-world conditions for testing and monitoring purposes.

Sequence diagram:



4 Information perspective



5 References and resources

Personas and User Stories:
https://www.pivotaltracker.com/blog/principles-of-effective-story-writing-the-pivotal-labs-way

Spring-Boot:

https://start.spring.io/

Swagger:

https://swagger.io/