Integrative Project Assignment

pi-sem2-2024-25 - version 3.1

June 2, 2025

Abstract

This document comprises a detailed description of the Integrative Project Assignment for the practical work to be developed within ESOFT, MATCP, MDISC, PPROG, and LAPR2 courses. The work consists of developing an IT solution that allows a simplified simulation of the operation of railway networks. This document briefly introduces the business domain and sub-areas, the functional and non-functional requirements of the solution to be developed, the Integrative Project technical details, and the operating mode, i.e., the teamwork framework.

Table 1: Version register

Version	Description
1.0	First version
1.1	Requirements Engineering US; Acceptance Criteria added for multiple US; Clarification on Problem Statement
2.0	Sprint 2 US and Acceptance Criteria
3.0	Added new US for Sprint 3; Acceptance Criteria updated for existing US; non-functional requirements added, Graphical UI required for simulator related US; Demand and Offer concept added to the problem.
3.1	Some typos corrected on Acceptance Criteria numbering; Non-functional requirement added concerning current budget and date

Contents

1 Integrative Project

In this project, students should analyze, design, and implement a computer solution that allows a simplified simulation of the operation of railroad networks.

Students must be organized into teams and develop a proof of concept that includes two components: (i) a map editing tool (including industries and cities) and scenarios (including temporal and historical context), and (ii) a simulation tool that will allow the user to define and interact with the operation of a railway network (including lines, stations, and trains) in the context of a map/scenario. Two types of tools will be considered to manage the railway network: statistical tools to assess the network performance and tools to analyze the network connectivity and the quality of the services provided.

Following the good practices learned throughout the degree and in particular in Software Engineering (ESOFT), Programming Paradigms (PPROG), Computational Mathematics (MATCP), Discrete Mathematics (MDISC), and Laboratory-Project II (LAPR2) courses, this project implies applying an iterative and incremental development process. Therefore, the agile SCRUM approach must be used to manage teamwork in each four-week SPRINT.

The software solution to be developed must be composed of a set of Java applications that must meet the requirements. To increase the solution maintainability and respect good software development practices, the implementation must follow a Test-Driven Development (TDD) approach.

1.1 Team operation framework

Each team with LAPR2 students must nominate a Scrum Master for each Sprint (rotative role) and create a Scrum Board in the Organization *Departamento de Engenharia Informática*, following these guidelines:

- Only one team member (the assigned scrum master) creates the Project Board with the title "sem2-pi-24.25-gnnn-board in which nnn must be replaced by the team number;
- The board privacy must be "private";
- Access (writer) must be given to TP and PL teachers assigned to the team's class;
- The project board must include the following columns: Backlog, ToDo, In Process, Testing, Done, Logbook.

All teams involved in the integrative project must create the project repository on GitHub in the Organization *Departamento de Engenharia Informática*, following these guidelines:

- Only one team member creates the repository, with the title sem2-pi-24.25-gnnn-repo based on the provided **template**;
- The repository access must be "private";
- All TP, PL, and OT teachers from the involved class courses supervising the project must be added to the repository with the "Reader" role.

2 Problem Statement: Description

The railway is an efficient, safe, and environmentally friendly way of moving people and goods. Given that the proper installation and maintenance of a railway structure involves very significant investments on a national scale (and transnationally in the European context), it is obvious that there is a need to manage the installed resources effectively. In addition to the structural and economic aspects of railways, trains and railways are a recurring theme in computer games, allowing experimentation from a playful and entertainment perspective, involving some of the complexity inherent to the subject.

In the context of this project, the teams will have to develop a software solution that supports some of the basic functionalities of managing a railway system, inspired by a classic simulation game (RailRoad Tycoon). Two tools will have to be developed: a map and scenario editor, and a simulation tool. In the context of the simulator, functionalities will be created that allow statistical analysis of the railway network functioning, including its multiple components, as well as tools that explore the topological aspects of the network.

2.1 Edition of Maps and Scenarios

The Map Editor allows you to create rectangular maps and place static elements on the map. Elements can be cities or industries.

2.1.1 Cities

A city bears a name, a location (with XY coordinates on the map), and a set of blocks that represent housing. In the simulator's context, cities generate

Figure 1: City of Madrid

and consume passengers, and mail, but they also consume final products (e.g., food, textiles, cars). The figure 1 refers to the city of Madrid.

2.1.2 Industries

As for industries[1], there are three different types (see figure 2):

- **Primary sector industries** - which generate resources, such as mines (*e.g.,* iron, coal, bauxite, or nitrate mines) and farms (vegetables, cereals, wool, coffee, rubber, cattle). The resources generated by these units can, if transported by train in the context of the simulator, be consumed/processed by other types of industries;

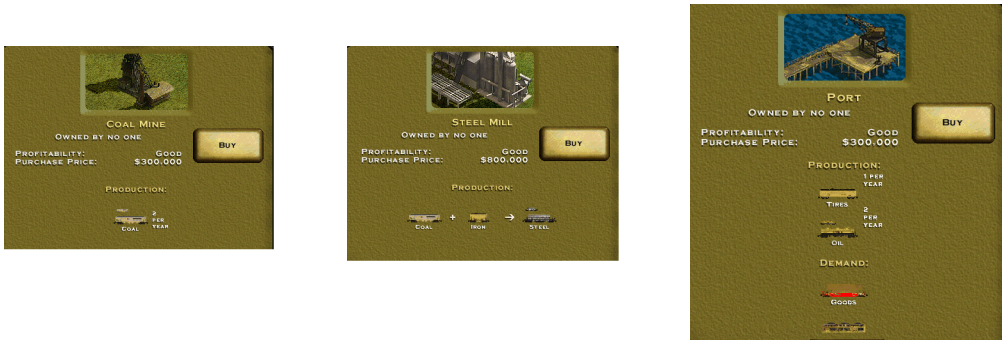

Figure 2: Coal Mine, Steel Mill and Port

- **Transforming industries** - which transform the resources provided by the Primary Sector Industries into new products. This industry includes bakeries, textiles, automobiles, and steel mills. For example, a

[1]For a full list of industries available in the game, check railroad-tycoon.fandom

steel mill transforms coal and iron ore into steel. The steel produced can then be transported to an automobile factory and used in the production of automobiles;

- **Mixed industries** - Ports potentially have the capacity, depending on what is defined in the scenario, to import/export and transform any cargo.

In figure 1, it is possible to see, next to the city of Madrid, the existence of a steel mill, a nitrate plant, and a wool production farm.

The Scenario Editor must allow the user to define the restrictions and conditions for using a given map, namely:

- Time Restrictions - the period in which the simulation will take place;
- Technological Restrictions - for example, which locomotives or industries are available in the scenario;
- Historical Restrictions - historical events that make sense in the scenario and change the operating conditions of the simulator. For instance, a war increases the steel demand, and a vaccination campaign or a schooling program encourages mail and/or passenger traffic. In this context, the goods that the Ports import and export are also defined.

2.2 Simulator

The simulator will enable the manager (or in the playful perspective, the player) to create a railway network composed of stations that serve cities and industries, lines connecting those stations, and trains that transport cargo between stations, according to a route defined by the manager. It is crucial to highlight that cities and industries only generate or transform cargo if they have a station; when the cargo is generated, it is then available to be collected at the station.

2.2.1 Stations

There are three types of stations as represented in figure 3: *Depots* (50k, 3x3 radius), *Stations* (100k, 4x4 radius), and *Terminals* (200k, 5x5 radius). Their cost and economic radius distinguish the stations.

All industrial and population sites within the radius of a station send (supply) and receive (demand) business through the station. By adding the

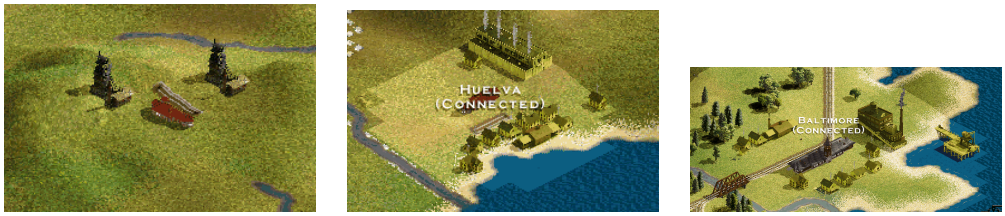

Figure 3: Depot, Station and Terminal

supply and demand for cargoes from the industry and population in a range, the supply and demand for the station are determined. For example, if it is assumed that each coal mine creates an average supply of two coal carloads per year, a station with three coal mines in its economic radius generates a supply of about six carloads of coal per year.

When built, a station consists of just one building, which can be upgraded with other buildings that can increase the station's performance, improve train operation, facilitate train manoeuvres, and increase the lifespan of cargo stored at the station. Some of the buildings that can be used to upgrade stations are: telegraph (later telephone), café (small or large), customs, post office, hotel (small or large), silo, liquid storage.

When viewing a station, in addition to the present buildings, the cargo ready for collection and the cargo that needs to be delivered (as in the example) should also be displayed (figure 4).

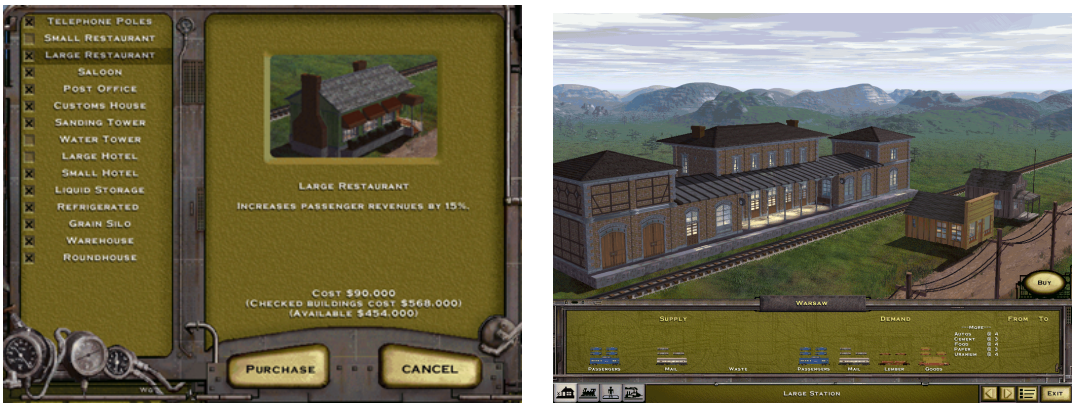

Figure 4: Station upgrades and Warsaw station

2.2.2 Railway lines

Railway Lines allow the connection of two stations. In the context of this project, it is not intended to make a detailed graphical edition of the route,

but only a topological representation of the connection between two stations. A line has a certain length, can be single or double, and can be electrified (or not).

2.2.3 Trains, Locomotives and Carriages

Trains are composed of locomotives and carriages. The locomotives provide traction to the composition (train) and can be of different types, namely, run by steam, diesel, and electricity. Note that electric locomotives require electrified lines. The locomotives are characterized by several technical aspects, such as power, acceleration, top speed, start year of operation, fuel cost and maintenance per year, and acquisition price. The carriages[2] are distinguished by the type of cargo they can transport (*e.g.,* passengers, mail, coal, iron ore, steel, cars) and, in the context of the simulator, they have no acquisition cost.

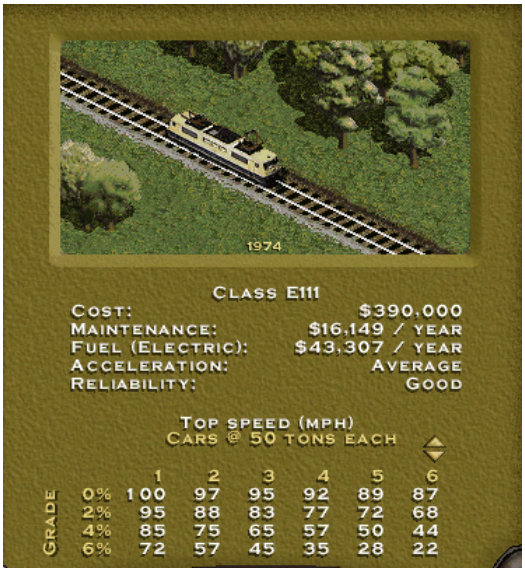

Figure 5: E111 Electric Train

2.2.4 Routes and Cargos

In the simulator, the player/user acquires trains within the available budget and can put the train into service on a specific route. A route is a list of stations the train passes by, along which it loads (carriages) cargo at each station.

[2]For a full list of carriages available in the game check railroad-tycoon.fandom

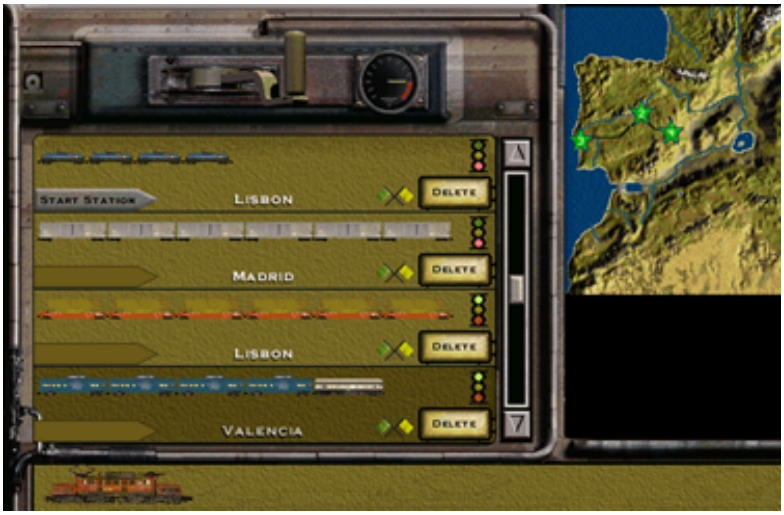

Figure 6: Train Route - Lisbon, Madrid, Lisbon, and Valencia

2.2.5 Simulator Operation

In addition to creating the dynamic elements already described, the player/user can play or pause the simulator. During the simulator's operation, the evolution of events should be displayed, namely the trains' journeys (start and end).

In the simulation, the following values should be considered (further values like locomotives costs and cargoes deliver values should be checked in the game documentation previously referred to):

Item	**Value**
Depot Building Cost	50
Station Building Cost	100
Terminal Building Cost	200
Single Track Cell	6
Double Track Cell	9
Single Track Electrified Cell	11
Double Track Electrified Cell	14

Table 2: Base Values for the Simulator

2.2.6 Demand and Offer

The demand for a specific type of load at a station can vary between 1 and 9. If there is no demand, the value will be 0. For example, if the demand is

5, the delivery of a load is paid for at the base value and will be modified by 20 per cent for each value that goes up or down. For example, for a demand of 8, the load value will be paid at the Base value + 60% and for a demand of 3 the load value will be paid at the Base value - 40%.

If 30 cargoes, or more, of a specific type are delivered in a station per year, the demand will drop 1 value. If 10 cargoes, or less, of a specific type are delivered in a station per year, the demand will raise 1 value.

2.2.7 Project Exclusions

The RailRoad Tycoon (II and later) game/simulator is particularly complete, covering the areas of gameplay, economy and engineering. Keeping in mind that this game is an inspiration for the current project, many topics, however, covered by the game will not be focused on in the project, in particular:

- Collision detection
- Timetable generation/management
- Calculation of train (de)accelerations
- Editing of line details (graphical mode), connections will be topological

2.3 System Users

This system may be used by different users, namely:

- **Editor** - a person who can create maps and scenarios that can be provided in a bundle with the game or sold separately.
- **Player** - a person who wants to play a game; in advanced versions, a scenario can be played by multiple players simultaneously.
- **Product Owner** - the person representing the organization that intends to develop the game.

3 Minimal Viable Product (MVP)

The purpose of this project is to develop a Minimal Viable Product (MVP) iteratively and incrementally; therefore, the work is divided into three Sprints:

- **Sprint 1** – Weeks 3 to 6 – from 10/March to 6/April

- **Sprint 2** – Weeks 7 to 10 – from 7/April to 18/May
- **Sprint 3** – Weeks 11 to 14 – from 19/May to 15/June

A description of the MVP is provided for each sprint. Teams must follow the user stories (US) provided and consider their interconnections and respective dependencies. At the end of each Sprint, each team must meet the specified requirements. Teams must add USs to the backlog, size them appropriately, and distribute them among team members[3].

3.1 Sprint 1

3.1.1 Maps, Scenarios and Simulator

This Sprint aims at developing the following User Stories (US) (Requirement Engineering and Analysis):

- US01 - As an Editor, I want to create a map with a size and a name.

  AC01: The maps dimensions are positive integers.

  AC02: Map name should be a valid file name.

  AC03: The maps should have a scale that states the size of cell map in kms.

- US02 - As an Editor, I want to add an industry (selected from the available industries) in a position XY of the selected (current) map.

- US03 - As an Editor, I want to add a city in a position XY of the selected map, with a name and a positive number of house blocks.

  AC01: A city name cannot have special characters or digits.

  AC02: The house blocks can be assigned manually or automatically (randomly around the city tag position accordingly to normal distribution).

- US04 - As an Editor, I want to create a scenario for a selected map.

  AC01: Definition of the behavior of ports, which cargoes they import/export and/or transform.

  AC02: Definition of the available locomotion types (steam, diesel, and/or electric).

  AC03: (Re)Definition of the factors that alter the generation (frequency) of generating industries.

[3]These will also be supervised in TP LAPR2 classes

- US05 - As a Player, I want to build station (can be a depot, a station or a terminal) with a location in the current map. The system should propose a name for the station based on the closest city and the station (*e.g.,* Porto Terminal, Ovar Station or Silvalde Depot). In case of Depot and Terminal, the center is the geometric one, in the case of station, the center should be defined by the Player (NE,SE,NW,SW).

  AC01: Overbuilding is not possible.

- US06 - As a Player, I want to upgrade a selected station with a building. Each type of station improvement has a date from which it is available. Some pieces of equipment are mutually exclusive (*e.g.,* small and grand hotel) and some equipment/buildings replace others (the telegraph was initially used to facilitate the operation of trains at stations, and was later replaced by the telephone, so after the telephone advent, the telegraph is no more available).

- US07 - As a Player, I want to list all the stations to select one to see its details, including the existing building and the demand/supply cargoes.

- US08 - As a Player, I want to build a railway line between two stations.

  AC01: The player should choose the stations from a list of registered stations.

- US09 - As a Player, I want to buy a locomotive.

  AC01: The player should choose the locomotive from a list of available locomotives for the scenario as well as a current date.

- US10 - As a Player, I want to assign a selected train to a route. A route is a list of Points-of-Route, each Point-of-Route is defined by: a station, a list of cargoes to be picked up.

  AC01: For each Point-of-Route I want to assign a cargo mode, that can be: FULL (the train only departs when full loaded); HALF (the train departs as soon it haves half of carriages loaded); AVAILABLE (the train departs with available cargoes in the station)

- US11 - As a Player, I want to list all trains.

  AC01: The display of trains must include the locomotive and the current cargoes.

  AC02: The trains should be grouped by locomotive type and then listed alphabetically by name.

- US12 - As a Player, I want to play a game using a simulator that generates cargoes at current stations, automatically, considering the cities and industries that the railway network serves.

  AC01: This simulator should provide options for start/pause.

  AC02: The simulator needs to model train departures. At the time of departure, the time of arrival is calculated. The trains travels at the maximum speed of the locomotive on double track sections and at 80% of the maximum speed of the locomotive on single track sections. Each carriage degrades 5% of the locomotive's speed up to a maximum of 30%.

  AC03: The simulator needs to model train arrivals. On arrival, the amount to be received is calculated according to demand and the distance between the departure and arrival stations. The existing building in the station that affect revenues from passengers need to taken into account.

  AC04: The simulator needs to provide a message alerting when a new locomotive becomes available, accordingly to its start year of service. After that it becomes possible acquire that locomotive model.

  AC05: The simulator needs to generate cargoes in the stations accordingly to the House Blocks and Industry facilities served by each station. The number of cargoes depends in each industry type and in the modification rates defined in the scenario.

  AC06: The simulator needs to update the demand in station every year based in the deliveries.

For Sprint 1, regarding the Requirements Engineering activity:

- teams with 5 or more students enrolled in ESOFT must elaborate all user stories;
- teams with 4 students enrolled in ESOFT don't need to elaborate on the US08 and US11;
- teams with 3 or fewer students enrolled in ESOFT don't need to elaborate on the US02, US08, and US11.

3.2 Sprint 2

3.2.1 Maps, Scenarios and Simulator

The user stories elaborated in Sprint 1 should now follow the Software Engineering process and proceed to Design and Implementation phases.

3.2.2 Routes, railway connectivity and maintenance

- US13 - As a Player, given a railway with stations and lines connecting pairs of stations, I want to verify if a specific train (steam, diesel, or electric powered) can travel between two stations belonging to the rail network (or from any type of station to another of the same type).

  AC01: The player should be able to choose the type of train (steam, diesel or electric) and station type (depot, station or terminal) in real time.

  AC02: A visualization of the stations, and the lines connecting stations of this scenario (using, for example, Graphviz or GraphStream packages) should be displayed to the player, where electrified railway lines are drawn with a different color from the others.

  AC03: All implemented procedures (except the used for graphic visualization) must use primitive operations only, and not existing functions in JAVA libraries.

  AC04: The algorithm(s) implemented to solve this problem should be documented/detailed in the repository documentation (using markdown format).

- US14 - As a Player, given a railway with stations and railway lines, I want to see a route that passes once, and only once, by each railway line to carry out maintenance on the lines.

  AC01: The player should be able to choose between the maintenance of all the lines, or only the electrified ones.

  AC02: A warning message should be displayed in case it is not possible to get such route. If possible, the station(s) where the route can start, should be displayed so that the player may select it.

  AC03: A visualization of the rail network (stations, railway lines) should be displayed to the player (using, for example, Graphviz or GraphStream packages), where electrified railway lines are drawn with a different color from the others.

  AC04: All implemented procedures (except the used for graphic visualization) must only use primitive operations, and not existing functions in JAVA libraries.

  AC05: The algorithm(s) implemented to solve this problem should be documented/detailed in the repository documentation (using markdown format).

3.2.3 KPIs and Statistical Analysis

Conducting a statistical analysis of railway station data is crucial for optimizing operations, improving efficiency, and maximizing profitability. By analyzing train traffic, cargo, passengers, and mail volumes, companies can enhance scheduling, resource allocation, customer satisfaction, and logistics. Additionally, reviewing revenue and expenses over time helps identify profitability trends, manage costs, and support strategic decision-making. This data-driven approach ensures long-term financial sustainability, operational improvements, and adaptability to changing demands. The best practices concerning statistical analysis, referring to stations, should be considered to develop the following US. The "Stations_Data.csv" file provides the necessary data to carry out the study.

In the elaboration of this US the following acceptance criteria should be considered:

- Programming Language: Python
- Development environment: Jupyter Notebook
- Work delivery format: A single Jupyter Notebook file, which contains all the work carried out.
- Each US must be composed of:
    1. Detailed description of the objectives and theoretical concepts on which the answers were based (in LaTeX).
    2. Python code and results.
    3. Thorough analysis and interpretation of the obtained results, extrapolated to the PI topic (in LaTeX).
- At the end of the file, you must indicate the contribution (in %) of each team member to the development of the work (the sum of all percentages must be 100%).
- US15 - As a Player, I want to perform a statistical analysis of the average annual profit of a user-specified railway station. In order to achieve this, a table with key descriptive measures: mean, standard deviation, median, and mode will be generated. Additionally, the analysis aims to identify outlier profits as well as to determine the most and least profitable years.

- US16 - As a Player, I want to perform a comparative analysis of passenger arrivals across all stations. To achieve this, a boxplot will be created for each station, displaying the number of arriving passengers using all recorded data. This visualization will facilitate the identification of trends, variations, and anomalies in passenger traffic across different stations.
- US17 - As a Player, I want to perform an analysis of the cargo arriving at each station using all recorded data. Given the station's name and cargo type specified by the user, a histogram will be created to visually represent the distribution of analyzed information, providing insights into cargo patterns.
- US18 - As a Player, I want to perform the analysis of train distribution, passengers, and mail across different stations. Using all recorded data, this distribution will be visually represented through three pie charts, providing a clear and comprehensive overview of how these elements are allocated among the stations.

3.3 Sprint 3

3.3.1 Game Playing and Reports

- US19 - As an Editor, I want to Save a Map in a file for later use.
- US20 - As an Editor, I want to Load a Map from a file previously saved.
- US21 - As an Editor, I want to Save a Scenario in a file for later use.
- US22 - As an Editor, I want to Load a Scenario from a file previously saved.
- US23 - As a Player, I want to Save a Simulation (Game) in a file for later use.
- US24 - As a Player, I want to Load a Simulation (Game) from a file previously saved.
- US25 - As a Player, I want to see the Year Financial Results including Expenses (Track Maintenance, Train Maintenance and Fuel) and Revenues (Cargo Deliveries).

  AC01: Track Maintenance is based on kms of line; the train maintenance is derived from locomotive properties and Fuel is based on traveled kms.

3.3.2 Shortest routes, and algorithms efficiency

- US26 - As a Product Owner, I want to conclude about the efficiency of the algorithms developed in US13 and US14 by analysing their worst-case time complexity.

  AC01: The analysis report of the worst-case time complexity of the algorithms should be documented/detailed in the repository documentation (using markdown format). All the algorithms should be presented in pseudocode, where the complexity analysis should be made.

  AC02: The procedure for the graphic visualization should be excluded from this analysis.

- US27 - As a Player, given a scenario with stations and railway lines, I want to get one of the shortest routes between two stations, which goes through an ordered list of stations I choose.

  AC01: A visualization of the scenario should be displayed to the player, where the shortest route is drawn with a different color.

  AC02: All implemented procedures (except those used for graphic visualization) must only use primitive operations, and not existing functions in JAVA libraries.

- US28 - As a Product Owner, I want to conclude about the efficiency of the above procedure, by analysing the worst-case time complexity.

  AC01: The analysis report of the worst-case time complexity of the algorithms should be documented/detailed in the repository documentation (using markdown format). All the algorithms should be presented in pseudocode, where the complexity analysis should be made.

  AC02: The procedure for the graphic visualization should be excluded from this analysis.

3.3.3 Cargo and Algorithm Statistical Analysis

- US29 - As a Product Owner, I want to run tests for variable-sized inputs to observe graphically the asymptotic behavior of the execution time of the US13 and US14 procedures (excluding graphic visualization).

  AC01: The graphic referring to the asymptotic behavior of the execution running time tests, as a function of the input size, should be presented in a time unit that allows to distinguish the running times of all tested examples (use any package, for example, Gnuplot).

- US30 - As a Product Owner, I want to obtain the best-fitting curve for the previous sample, and see it graphically.

  AC01: The fitted curve should align with the reference function obtained from the theoretical analysis of the procedure's complexity.

- US31 - As a Player, using the same data file as in the previous Sprint (Sprint 2), I want to perform a statistical analysis to determine which type of cargo contributes most significantly to the revenue of a given station, in a specific year, both defined by the user. The goal for the specified station/year is to identify the cargo (independent variable) that shows the highest correlation with revenue (dependent variable). Based on the selected cargo, a linear regression model should be applied, presenting the error associated with the respective fit. Assuming the amount of cargo increased by 10% in January of the following year, estimate the expected revenue for January with a 95% confidence level.

  AC01: Programming Language: Python

  AC02: Development environment: Jupyter Notebook

  AC03: Work delivery format: A single Jupyter Notebook file, containing all the work carried out for all US.

  AC04: Documented in Markdown format, for each US, you must present:

  1. Detailed description of the objectives and theoretical concepts on which the answers were based.
  2. Thorough analysis and interpretation of the obtained results, extrapolated to the PI topic.

3.4 Non-functional requirements

This section describes some non-functional requirements that must be considered when implementing the project.

- During the simulation execution current budget and date should be permanently visible
- The application's graphical interface is to be developed in JavaFX 11 covering the functionalities related with the Player.
- Validation of business rules must be respected when recording and updating data.

- The class structure must be designed to allow easy maintenance and the addition of new features following the best Object-Oriented (OO) practices.
- The application must be developed in Java language.
- All those who wish to use the application must be authenticated with a password of seven alphanumeric characters, including three capital letters and two digits.
- The application documentation must be in English.
- During system development, the team must: (i) adopt best practices for identifying requirements and for OO software analysis and design; (ii) adopt recognized coding standards (e.g., CamelCase); (iii) use Javadoc to generate useful documentation for Java code.
- The development team must implement unit tests for all methods, except for the methods that implement Input/Output operations. Unit tests should be implemented using the JUnit 5 framework. The JaCoCo plugin should be used to generate the coverage report.
- All the images/figures produced during the software development process should be recorded in SVG format.
- The application ought to employ object serialization to guarantee the persistence of the data in two successive runs.