

# Team DUS - GUI Report

The game is by nature turn based. In each player's turn, they will be expected to be able to send three trains down custom routes, apply items to the game or trains and be able to have sufficient time for thinking. The GUI must not put unnecessary limits upon the players and that was our main goal when it came to its design.

A player's actions can be broken down into several key sections:

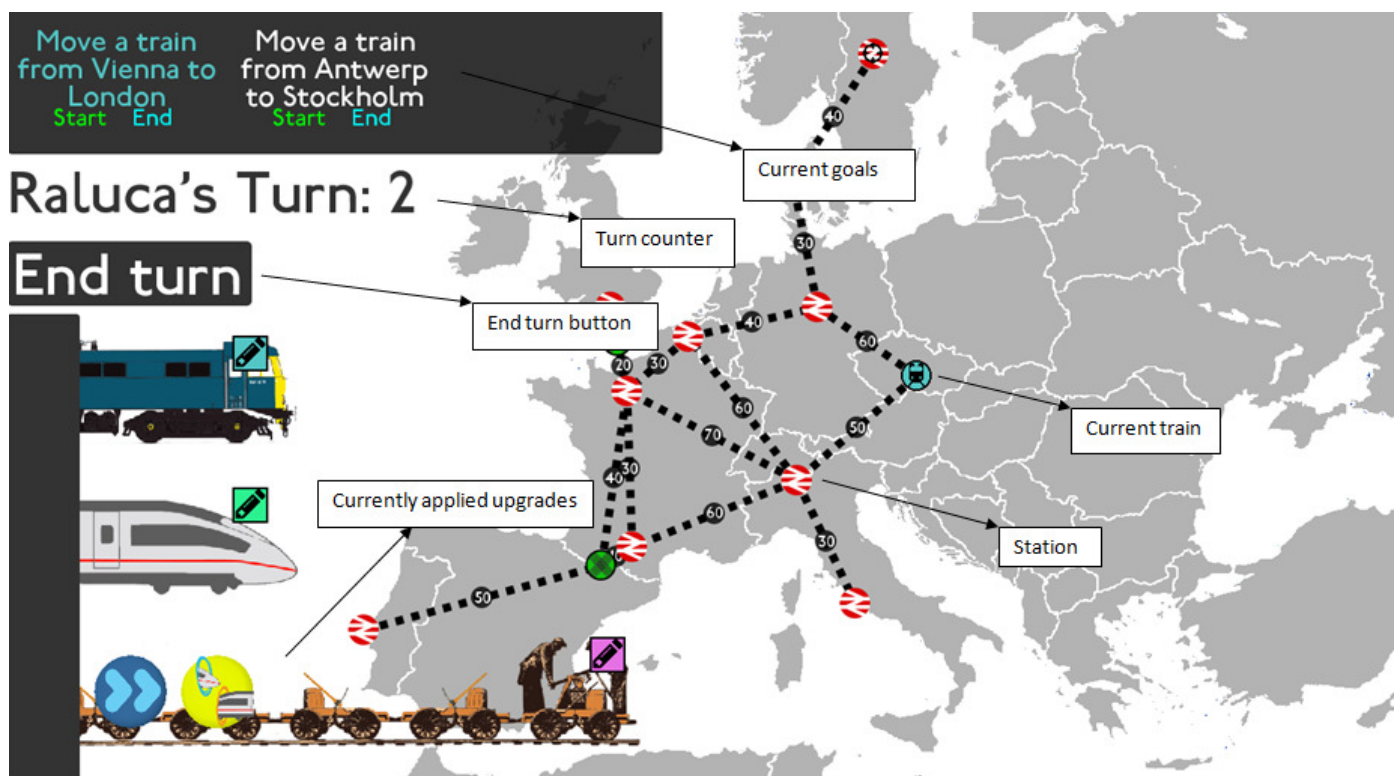
1. Information gathering and processing  
The player must gather information regarding the current state of the game before processing it. The GUI must provide all of this information.
2. Decision making  
The player then uses the processed information that they have gathered to make a decision regarding what to do in the game.
3. Execution of decision  
The player carries out the decided action by navigating through the interfaces which manipulate information behind the scenes. The GUI must allow the user to input all of their decisions to manipulate the state of the game.

Between the start of the turn and the end of the turn, the player may iterate through these three actions many times as they make decisions and then re-evaluate changes that they have made. A player operating with a longer term game plan may require more time to make a decision once they have gathered information from the board, as we encourage tactical moves through a few means:

1. The player sees some of the opponent's information, which gives them a rough idea of their current state of the game and can guess their intentions. They can see opponent trains on the map, although they cannot see the goal each of train is associated to, nor any upgrades that have been applied to it or any upgrades available in the opponent's inventory.
2. The player can choose to wait to collect upgrades and apply more than one at one time, for a massive boost of train capabilities, or they can choose to apply each upgrade as soon as it is received and as sensibly as possible, knowing there is an inventory limitation and that any resource that is not in use might actually be keeping them behind.
3. The player can focus most of the upgrades on one train and try to achieve as many goals as possible with it or they can spread evenly the upgrades among the three trains. It may not always be clear which way is better without a definite strategy that takes into account the upgrades available and the opponent's movement on the map.
4. The player can guess when it is likely that the opponent has completed a goal, since their train disappears from the map for good, meaning it has either been teleported or reached the destination by itself. However, this may be a tricky guess, as all opponent's trains are colour coded in the same manner, meaning the player can get tangled between the trains and miscalculate when completion of goals occur.

By design, the game contains a low amount of information to be displayed. The player can gather the following information from the GUI:

- Their own current goals; of which there are three.
- Their own active trains on the map; of which there is a maximum of three
- Their opponent's active trains on the map; of which there is a maximum of three
- Their upgrades in their inventory, with a maximum of three train engines and four upgrade items.
- The current state of the game map.
- The completion of a goal
- The possibility of discarding random upgrades and getting new ones once the inventory maximum has been reached.



We wanted to keep the design minimalist, for ease of use and to not create information overload for the players. Additional information can be retrieved through other means, discussed further in this document.

The map contains approximately 10 to 15 nodes. Very little information needs to be given to the player about these nodes. A simple visual graph structure is used to show the map of the stations and junctions. Nodes are connected by edges to show where a train can travel. Whether a node is a junction or a station is indicated by the symbol shown on the node.

The player's current trains are shown in the train container on the bottom left of the screen. A goal specifies two parameters that the associated route must meet to complete the goal. In order for a journey to be planned and initialised, a goal must be attached to the route; therefore the goal information will also display which train it is attached to via colour coding.

## GUI Aesthetics

### Font

The railway industry already uses two very well established fonts in the UK; the British Rail font and the London underground and airport font. The British Rail font is used in outdoor railway stations with open air platforms and is featured on official publications in these stations, whereas the London underground font is found in London train stations and underground stations as well as in airports. The underground font is perceived to be much more recognisable due to the popular culture artistic usage of the font. The underground font has expanded into a font used in any piece of railway art and has become a part of many people's lives in the form of popular culture publications.





Due to the fact that the London Underground font is more recognisable, it is used throughout the game. The players will experience a feeling of familiarity when they first play the game and will be more likely to like the game because of these familiarities. You can see samples of the two fonts below:

**This is the London Underground font. The Quick Brown Fox Jumped Over The Lazy Dog.**

**This is the British Rail Font. The Quick Brown Fox Jumped Over The Lazy Dog.**

### Item Rarity

The player can hold a maximum of four upgrade items and three train engine upgrades. Because upgrades have varying levels of utility and also various levels of usefulness, they can be split up into separate categories of power. More powerful upgrades - which are more useful - are to be received more rarely, while less powerful upgrades are to be received more commonly. In order to display to the player this rarity level, each upgrade is displayed with a different coloured background, which corresponds to the rarity band of the item. The colour bands are as follows:

Common	Uncommon	Rare	Legendary
			
Green	Blue	Purple	Yellow/Gold

The advantages of these colour bands is that they are already well known among people that play some video games already. These colour bands are also used by many RPGs such as “World of Warcraft” or “Destiny”. While our colour bands do not share names with the most-popular ones, our four bands are simply a subset of such colour banding schemes. This means that many of our players will be familiar with this system of categorisation and will know which items in their inventory are perceived to be more desirable and more rare. If a player is not familiar with the colour rarity system, then they can very quickly get used to the four colour rarity rate, this is specified for them in the user manual.

## Overall Colour Scheme

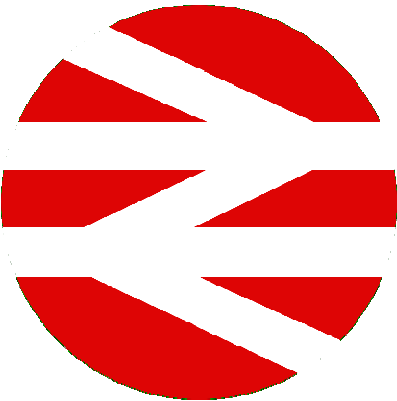
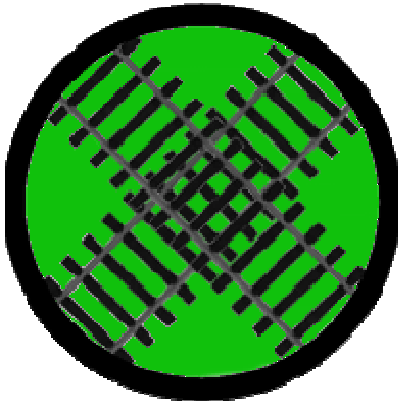
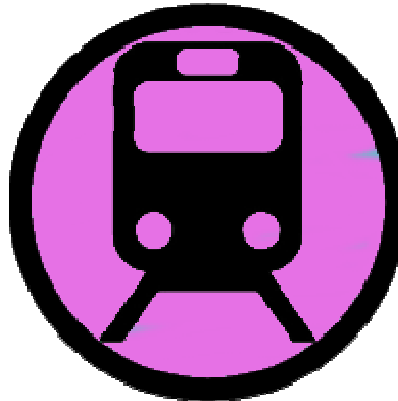
The game utilises a very neutral colour scheme which does not boast a particular colour of focus. The game background is white which provides contrast to the grey map. Interfaces are of a much darker grey than the map is; however interfaces also implement transparency in order to show some detail in areas behind them.

The neutral colour scheme works well to emphasise the more colourful icons used by the game. Nodes on the map are either red or green and show up really well on the map. The neutral map colour also highlights the edges of the graph which are black; contrasting black on a light grey makes the edges of the graph clear to see for the players.

## Sectional Breakdown

### Map

The map of the railway network is represented graphically by using a weighted, undirected graph. Nodes on the graph represent stations or junctions that trains can travel to and from. Stations are represented by the station icon and junctions via the junction icon. Nodes on the graph are connected by edges. Visually, the edges on the graph are drawn using dashed lines to follow the train based theme of the game, this aims to mimic a railway track. Trains are shown on the map with small coloured icons which are placed on the last visited node by the train. Red icons represent trains that belong to the opponent.

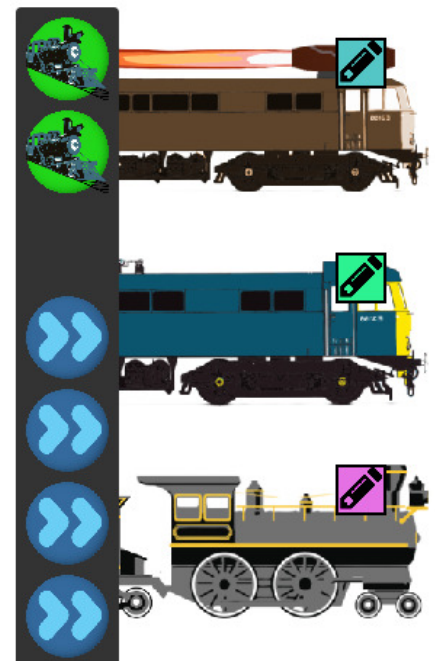
Station Icon	Junction Icon	Train Icon
		

The map is a highly abstracted map of Europe which displays land in grey, borders and areas of sea in white. The player does not necessarily need to know any information about Europe other than its general shape. Since the underlying structure of the game runs upon a graph, the game can be run with no map, utilising fictional locations. However, the premise of Europe is used as an immersion technique, in order to give the game a theme and make it more familiar to the user. It can also be assumed that players will be familiar with the rough geography of Europe. So when a player is given a goal involving specific nodes, they will already know the general location the nodes involved. Since the game utilising major cities in Europe as node locations, this cuts down on the time a player will spend performing information gathering regarding the map; they can instead navigate directly to a node. However, if a player is unsure about the exact location of a city of the map, they can easily hover over their best guess and check if they were right (More in the *Hover Over* section below).

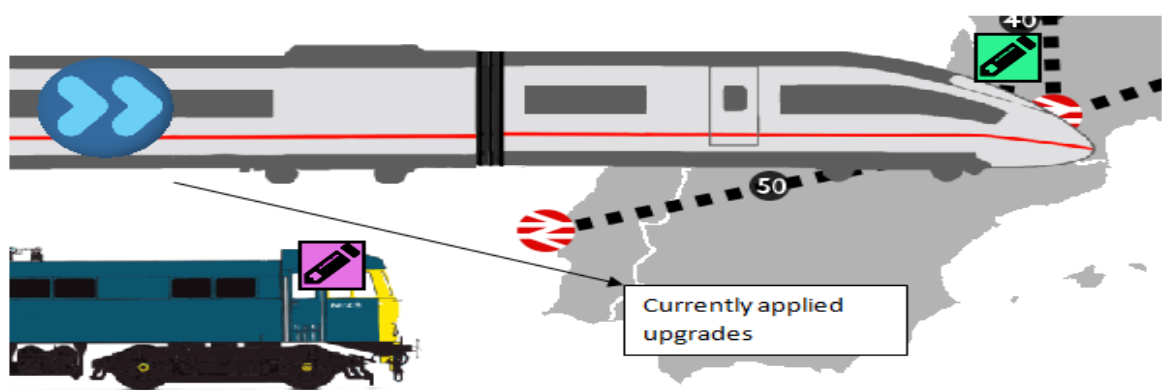
## Train Information

Train information is viewed from the left hand side of the screen directly next to the player's inventory. Three basic trains are hidden off screen with the front of the trains being shown in such a way that they are recognisable. As the trains are clicked upon, a longer version of the train pulls out of the left side of the screen. We implemented this approach as to minimise the space they can take on the screen, so the player can have at all times a clear view of the map, but at the same time be able to retrieve more specific information, if they want to. The upgrades currently applied to the train are shown upon the image of the train to allow the user to see the state of the train and quickly decide if more upgrades are desirable or not. By hovering over an active upgrade, the player is shown the mouse over tooltip explaining what the upgrade is.

The location of trains is a high level abstraction to avoid as much confusion as possible for the players. While the underlying system stores the exact location of the train on the graph, it is represented graphically to the user as being located at the most recent node that it has passed.



Primarily, this design decision was taken to reduce complexity introduced by having two trains travelling on the same track, as well as making it an ideal mechanic with which to implement the blocking of tracks.



In order to establish the link between the three trains shown in the train menu and the trains on the board each train has been colour coded. The colour of the edit button on each of the train in the train menu matches up to a train of the same colour on the game board. The three colours are unique in the game to this system and are as follows: pink, turquoise and pale green. Trains on the board that belong to the opponent are all coloured in red.

## Item Inventory

The item inventory is placed on the leftmost edge of the screen. The train information menus are slightly hidden behind the inventory. In order to aid the player's ability to differentiate between the two different resources, the inventory is split into two sections. The top three slots of the inventory container are reserved for engines; while the bottom four slots are reserved for train upgrades. This leaves the player with a maximum inventory size of seven.

It is important that the inventory is situated next to the train information menu. In order to apply an engine upgrade or train upgrade to one of the player's trains, the player drags the upgrade onto the desired train in the trains menu. The small drag distance means that the player saves time by quickly moving the upgrade onto the train, while the large height of the trains means that the player is less likely to make a mistake which would result in the upgrade being applied to the wrong train.

Each item in the inventory is represented by a small unique icon that represents what bonus the item applies and performs. If the player requires more information then, by mousing over an icon they can view a mouse over tooltip which explains the upgrade in accordance to the instruction manual for the game. The quality and rarity of the item is represented by the colour of the background of the icon, as previously mentioned in the colour coding section.

Once the inventory is full and no resource is consumed, the player is asked if they want to randomly discard and receive new upgrades, which can be a good tactic if the player is stuck with basic, common resources that they would rather not use.





## Goal Information

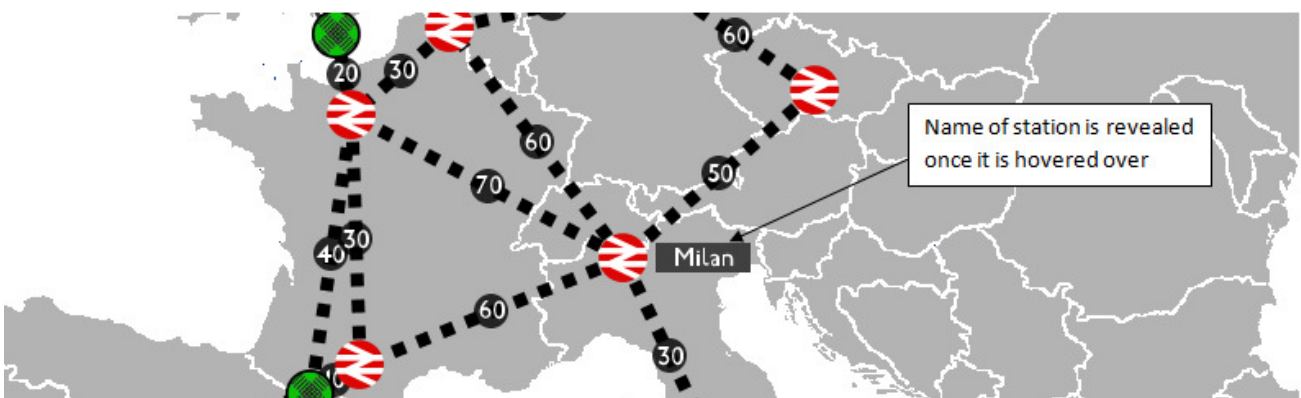
Goals are shown in the upper left hand corner of the user interface. Each goal has its own section, in order to increase readability. Each goal is colour coded based on its associated train and information regarding the start and end nodes of the goal is shown without navigation being required. In order to aid the player in finding the nodes in each goal, the player can mouse over either the start or end node in the goal and the node on the map will become highlighted with a reticule (more in the *Hover Over* section below).



## Hover Over Feature

Our game supports a hover-over feature for almost all GUI elements. This is primarily to reduce the amount of information shown on screen at any one time. If a player needs to find out more about a particular aspect of the game (e.g. the name of a particular node or the name of an upgrade), they can simply hover over it and the information shall be displayed in a tooltip. This technique has been implemented in three places in the game:

- Nodes: As the map is familiar to the user, whom is likely to know the rough location of the cities on a map of Europe, the names of the cities have been omitted. This is in order to make the map less cluttered and to make the connections more visible. However, if the user does not know the location of certain nodes, the name can still be quickly shown by hovering over the node, an effective solution for information retrieval while maintaining a minimalistic GUI.



- Goals: Although the goal clearly states the beginning and end nodes in the description, again, the location of these nodes may not be familiar to the user. So a hover over feature has been implemented in order to link the goal cities with their actual position. By mousing over the start or end node of the goal, a targeting reticule is displayed on the relevant node, even during route creation when it is most useful. By seeing the start and end goal highlighted on the map, the user is more likely to observe an optimal sequence of adjacent nodes that fulfils the goal.



- Upgrades: In order to reduce clutter on the interface, the items in the inventory are also represented only by icons, the name being retrievable through hovering over that specific item. This avoids unnecessary information once the user has become familiar with what each icon represents. This type of interface design is focussed mostly on creating an extremely learnable experience for our game.



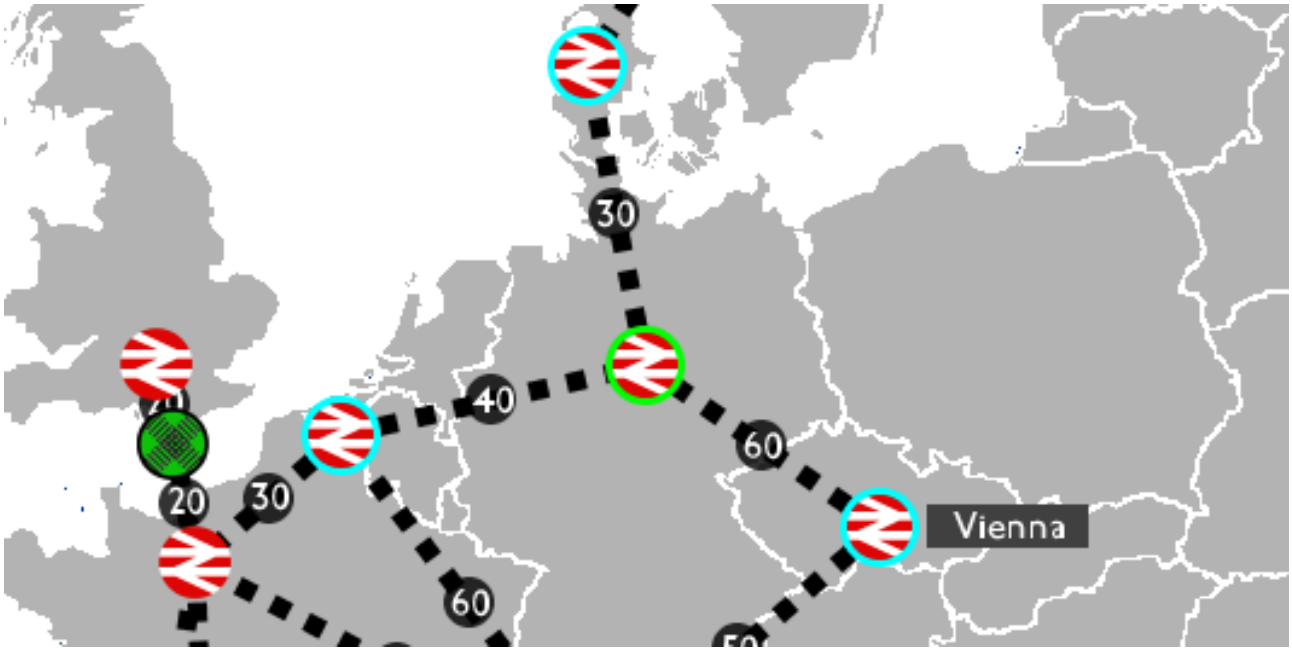
## GUI Interactions

There are a number of interactions that our GUI supports. This section shall outline the possible interactions that the user can have with the GUI and how they are associated with their ability to play the game.

### Create Route

The GUI supports the game's ability to construct a route by clicking the *Create Route* button associated with each train, this will then enter the route creation state. We chose to create a route this way so that a user will not be able to create or edit a route of a train accidentally, which could cause problems later on in the game if it is not behaving as they would expect. It is not mandatory that the player makes use of all of their available trains, as the creation of the route is associated with each train, although this is desirable if they aim to win the game.





The route creation state is very intuitively designed, in the sense that once the user selects the start node of one of the goals, then adjacent nodes will be highlighted. This continues as the user selects adjacent nodes until the route reaches the node specified as the goal's end node. This approach works hand-in-hand with the highlighting of start and end nodes of the goal; it helps the user to easily select a suitable route for the given goal. This is also a form of validation for our route. We can guarantee that the route passed to the route backend is a possible route, as the user can only select adjacent nodes. Hence no unconnected nodes will be in the route in sequence, this removes the need to validate the user's input in the backend and prevents the user from crashing the program through erroneous input. This greatly increases our game's reliability. The player is able to see what train they are editing the route for, as the tracks between the selected nodes become the colour assigned to the train.

## End Turn

The only way to end the current turn is to press the 'End Turn' button which is featured in a prominent location separate from the rest of the GUI. Once the player has finished making their decisions, they can commit them by ending the turn, they then affect the state of the game. This prevents the player from accidentally pressing end turn when they actually mean to apply an upgrade or select a node on the map. This is important as if the player often clicked mistakenly on the button or there was a keyboard hotkey that may be easily pressed, then this would create a frustrating experience which we want to avoid.

## Replace Goal/Upgrade/Engine

The player can easily upgrade from one type of engine to another, by dragging the upgraded onto the desired train. This brings about changes in the speed of the train from that moment onwards. Moreover, if a player tries to redundantly replace an engine/upgrade with the same type of engine/upgrade, they are warned it is not possible and the engine remains in the inventory for further use.

