**D1 Submission Report**

**Use Case Diagram**

The use case diagram is an outline of all the functions of the system as well as how they interact with each other and all the actors in the system. When creating the use case diagram, I was planning to separate the BusInfo and @bus subsystems within their own boxes, contained in a bigger box representing the whole Bus Metropolis system as a whole. However, I was advised against this which makes it easier to see how all actors and functions interact with different parts of the system as a whole.

The most prominent actor in the system is the human controller, henceforth referred to as HC. The HC would work at the central facility of the bus metropolis system where they would interact with the BusInfo system. This subsystem gives the HC access to administrative functions of the system which are required for the smooth operation of the overall system.

The BusInfo system provides the HC with the ability to view journeys on a selected route as well as the timetable of that route. This enables them to have an overview of the current state of the system. This is essential as it means they know what actions to take in the event something is not right, i.e. a bus has malfunctioned and they need to know what specific journey that bus was making in order to arrange a replacement bus. They also have the GetBusState function which would show them the state of a given bus such as the driver assigned to the bus. They also have the ability to cancel journeys through the CancelJourney function.

The HC has the ability to add new busses to the system, however they cannot remove buses themselves to prevent any accidental removals. The AddNewBus function includes the ModifyBus function because for a new bus to be fully initialised as part of the system, the bus must be labeled as in service and assigned a route. The ModifyBus function enables the HC to modify a given bus and change its operational status and the bus route it is assigned to. The ModifyBus function also outputs to the Get@BusState function on a bus which sets if that bus is in service or not.

There is a similar function that allows the HC to add new drivers to the system should a new driver be employed. The AddNewDriver function includes the AssignDriverToBus function as every driver on the system must be assigned their own bus.

The BusStopDisplay is shown as an actor in the system to make the use case diagram representative of how the system would actually function. However, this actor will not be in the implementation due to out of the scope of this project. This actor only has one function within the system, that being the DisplayComingBuses function, which will figure out estimated arrival times of buses.

The driver will interact with the @bus subsystem through an interface on the bus, this is possible because @bus will operate locally on each bus. The driver will have the ability to start a journey which requires the driver to input the route they are doing, via the SetRoute function. The StartJourney function also includes SetBusState which sets the state of the bus locally by using the Get@BusState function of the BusInfo system which accesses the ViewTimetableOfRoute and GetBusState functions to initialise the state of the system. The state contains information like the current time, first departure time, the timetable of the set route and if the bus is in service or not as the bus must be in service to start a journey.

The driver is able to end journeys through the EndJourney function which can be used once all the stops in a route have been called at. In addition, the driver is also able to request a journey cancelation in the event of bus malfunction by using the RequestJourneyCancelation function which requires the to contact the HC to get the cancelation approved and filed on the BusInfo system. The driver also has the power to deny passage onto the bus to people who do not have enough money on their card.

The passenger can their card on the entrance reader, this records their entry stop and checks their balance. They also scan the exit reader as they leave the bus, this causes the ProcessPayment function to charge them the fee.

**Foldable Concept Maps (FCMs)**

The FCMs are a series of concept maps which detail all the requirements of the system as well as showing how these requirements will work. The Visio file consists of an overview which shows the entire system and its two subsystems and their requirements. The main system and its subsystems also have overview pages of their own which describe what the system is, why it exists and who will interact with it.

The first requirement of the system is to store a timetable for each route. The timetable database will store many bus routes, each with a clockwise and anticlockwise direction and each of these routes will have their own timetable with set departure times.

The 2nd requirement is for the system to uniquely identify buses, stops and routes. Each bus, stop and route will have a unique numeric ID. In the case of the route, passengers will be able to associate each route with it’s own numeric ID.

The 3rd requirement is for HCs to be able to add new buses and not remove them. They will do this using the add new bus function on the main menu of BusInfo, this will create a blank bus and run the ModifyBus function to initialise the bus as in service and give it a driver.

The 4th requirement is for HCs to be able to label buses as in or out of service. The HC first decides if a bus should be in or out of service by whether it is malfunctioning or not. They then change this parameter via the ModifyBus function.

The 5th requirement is to ensure drivers are assigned a bus and route. The HC will be able to modify a driver or bus by selecting the respective option in ModifyBus. They will select the specific bus or driver to modify and will be able to change information like the buses service status, the driver of a bus or the route a driver makes.

The 6th requirement enables HCs to cancel and reassign journeys by the driver contacting the HC and requesting cancelation. The HC decides if to approve this or not and cancels the journey using the CancelJourney function and proceeds to reassign a bus and driver to fill that gap.

The 7th requirement is for HCs to be able to view a route’s timetable and journeys or a buses stats through the appropriate option in the main menu. The route timetable is printed from the timetable database as well as the journeys on that route. Or if the HC selected to view a bus’

Stats, they select the bus and its stats are displayed before returning to the MainOptions screen.

Requirement 8 belongs to @bus and it is to provide a control interface for the bus. This will allow the driver to start a journey by entering their route number, end a journey after calling at the last stop or request cancelation by contacting HC who will label the bus as out of service, cancel the journey and arrange a replacement. Passengers are alerted of cancellations and are not charged the journey fee.

The 9th requirement is for passengers to pay for journeys only using their RFID card. They swipe the card on entry which records their entry point and checks if they have enough money for the longest journey from that stop. If they have enough, a green light is shown and a happy beep is sounded and they can enter the bus. If they don’t have enough, a red light is shown and angry beep is sounded and the driver doesn’t allow them to board.

Requirement 10 requires passengers to pay for the journey by scanning their card on exit. The price is calculated based on the number of stops called at with them on the bus. If they do not scan their card, the maximum fee is charged. The bus stores each charge locally until the data is uploaded at the next stop.

Requirement 11 is for the bus to locally store and update the charges for each passenger. This is worked out by them scanning their card to enter and is applied on the central server at the next stop. At the end of a journey, all remaining data is uploaded and local storage is wiped.

Requirement 12 is for stops to connect to a bus’ transponder to enable exchange of information. If the transponder malfunctions, the driver is alerted to contact HC so they can arrange a replacement bus, cancel the journey and label the bus as out of service.