Assignment 2 – delivery module:

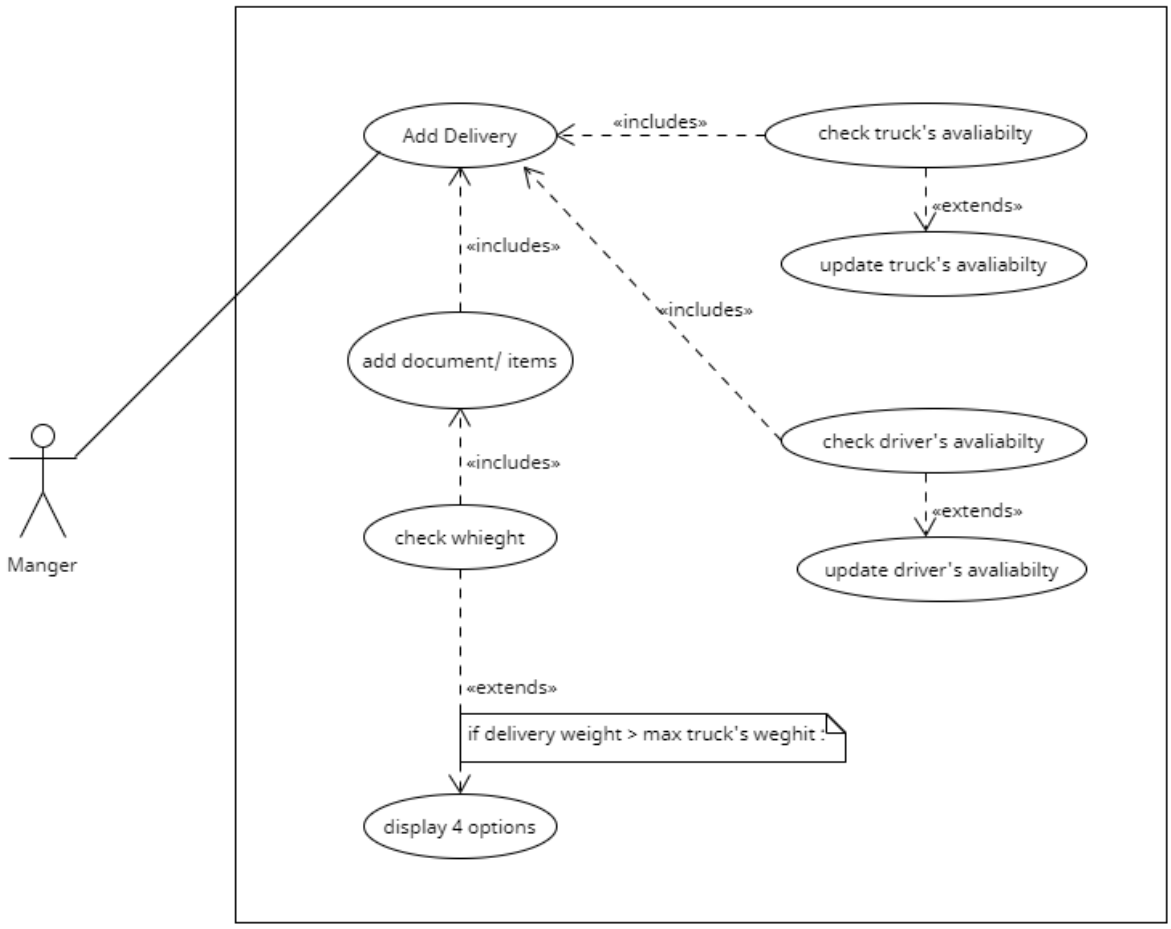
Names:

Rahaf Mrowat, Id: 212683635.

Bashar Qais, Id: 315282038.

<u>The use case that is relevant for the our module :</u>

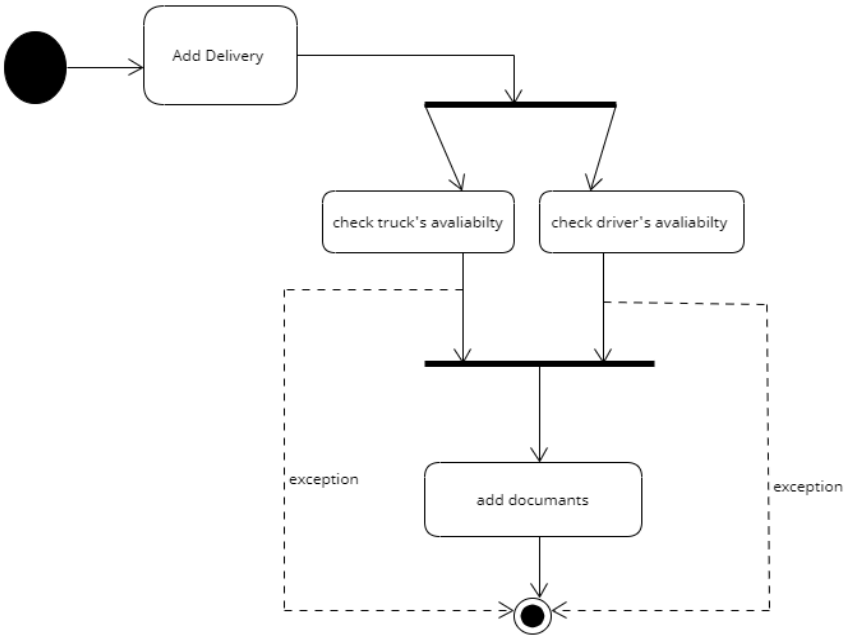
1. prepare delivery (it's h from 1.1)



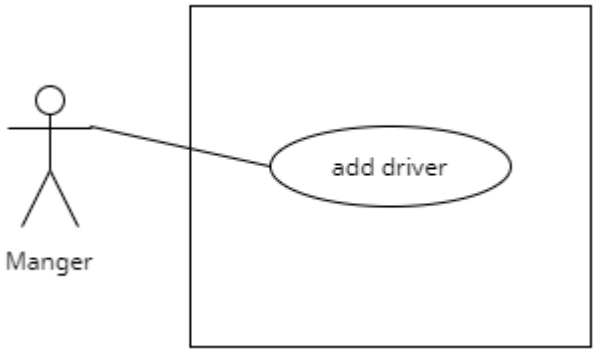
Scenarios description:

Use case name	**Prepare delivery.**
Textual Description	The manger tries to add a delivery to the system, the systems checks if the delivery can be prepared if though makes the delivery and it's documents (for each destination) and add it to the system.
List of actors	Manger.
Pre-conditions	The truck and the driver must exist and be available. The delivery date is valid. The driver is on his shift.
Post-conditions	The delivery has been added successfully.
Main success scenario	The manger tries to add a delivery with: an available truck, an available driver that is on his shift and has a suitable license for the truck and a valid date. The system adds the delivery and makes its documents.
Alternative/Extensions	- if the truck's type does not match the driver's license, returns an error message. - if we try to add items to the delivery that exceed the maximum weight for the truck, then the system displays 4 alternative options for the manger to pick: remove destination from the delivery, change destination, change truck for the delivery, remove items from the delivery.

Activity diagram for this use case:



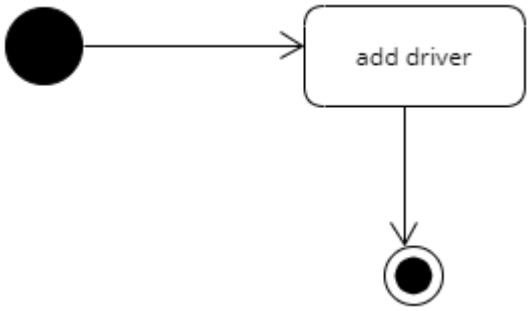
2. Add Driver:



Scenarios description:

Use case name	**Add Driver.**
Textual Description	Adding a new driver to the system.
List of actors	Manger.
Pre-conditions	None.
Post-conditions	Success if the driver doesn't exist.
Main success scenario	the manger tries to add a new driver to the system that doesn't existed already.
Alternative/Extensions	None.

Activity diagram:



Behavioural analysis:

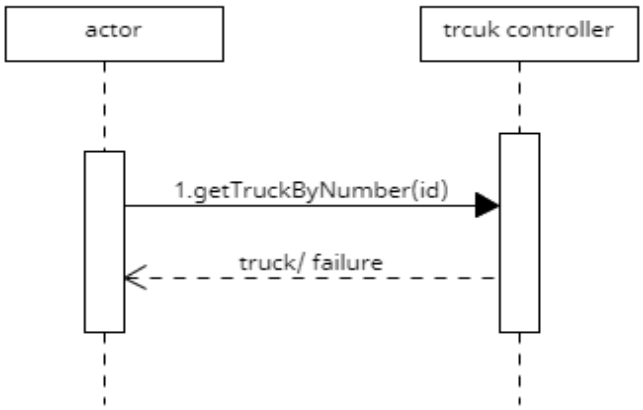
Contracts and sequence diagrams for the relevant system events:

For the first use case – **Prepare Delivery:**

CONTRACT CO1: getTruckByNumber

Operation:	getTruckByNumber(trusk: integer)
References:	Cases: Prepare Delivery.
Pre-conditions:	none
Post-condition:	returns the truck who has this id.

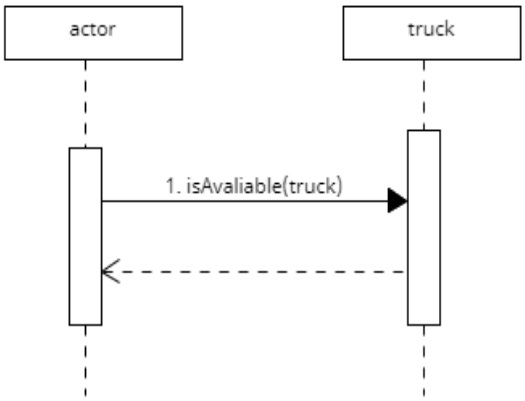
Sequence diagram:



CONTRACT CO2: truckIsAvaliable

Operation:	isAvailable()
References:	Cases: Prepare Delivery.
Pre-conditions:	the truck is already exists.
Post-condition:	returns true if the truck is available else returns fasle.

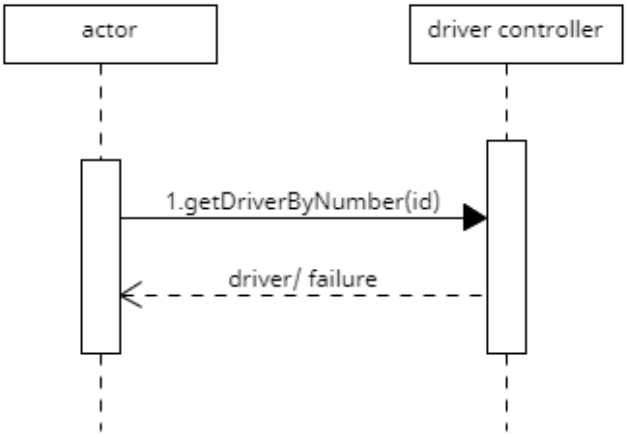
Sequence diagram:



CONTRACT CO3: getDriverByNumber

Operation:	getDriverById (driver: integer)
References:	Cases: Prepare Delivery.
Pre-conditions:	none
Post-condition:	returns the driver who has this id, if does not exit returns an appropriate message.

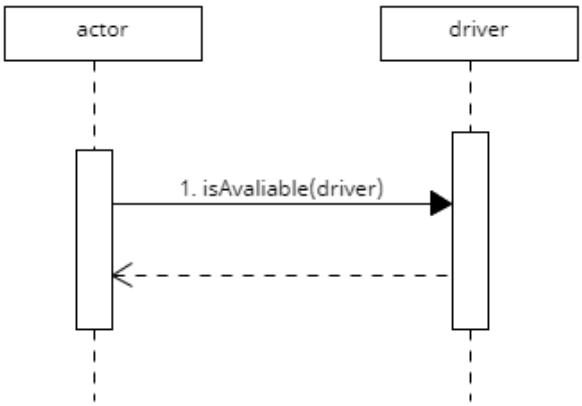
Sequence diagram:



CONTRACT CO4: driverIsAvaliable

Operation:	isAvailable()
References:	Cases: Prepare Delivery.
Pre-conditions:	the driver is already exists.
Post-condition:	returns true if the driver is available else returns fasle.

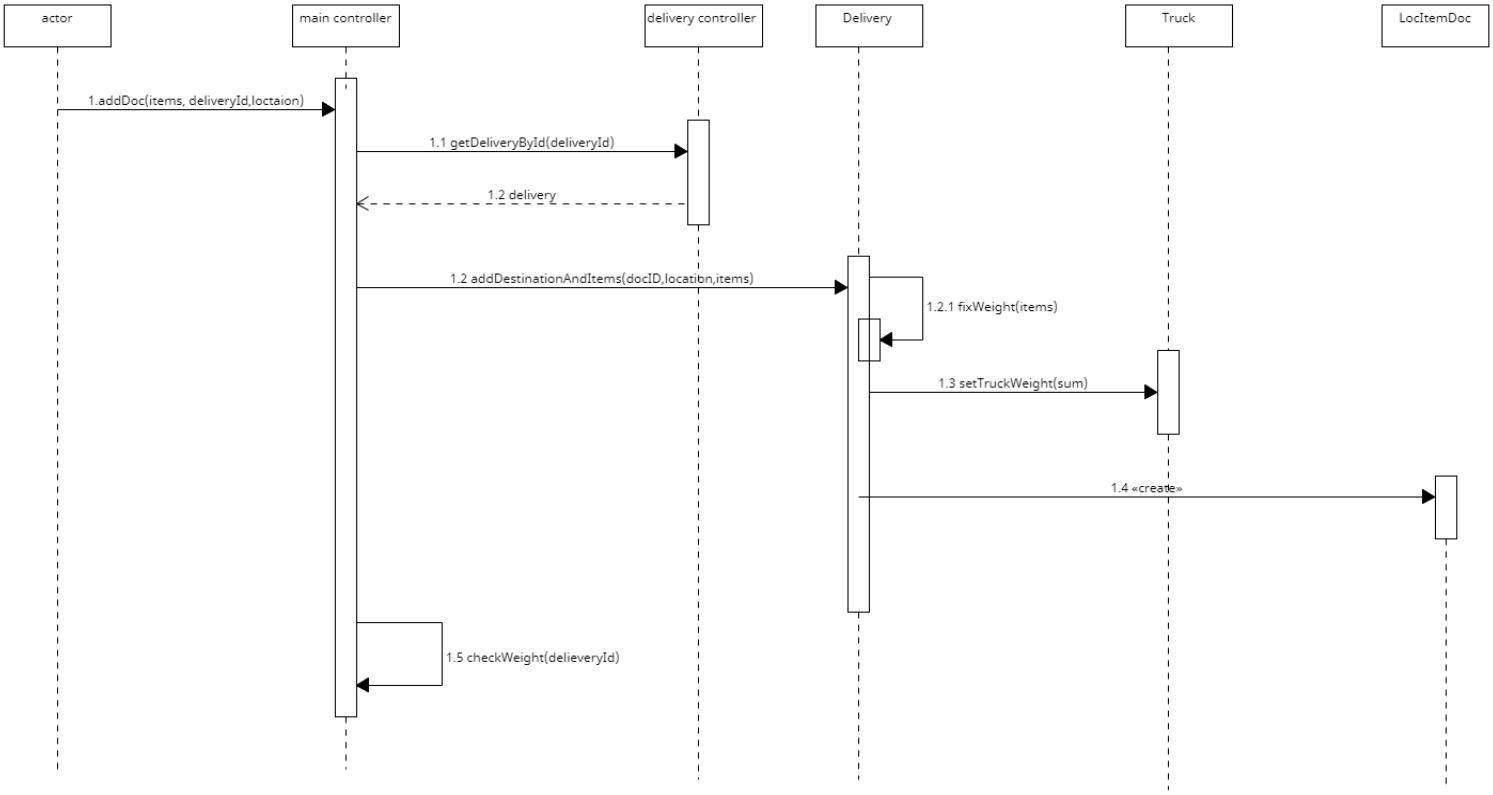
Sequence diagram:



CONTRACT CO5: addDoc

Operation :	addDoc(toadd: HashMap<Item,Integer>, deliveryid: integer , l: Location).
References:	Cases: Prepare Delivery.
Pre-conditions:	the “deliveryid” must be existed for an existed delivery.
Post-conditions:	if the weight is suitable for the truck the items will be added successfully and their document.

sequence diagram:



sequence diagram for the second use case – **Add Driver:**

CONTRACT CO6: addDriver

Operation :	addDriver(int id,String name,String licenseType)
References:	Cases: Add Driver.
Pre-conditions:	none.
Post-conditions:	if the driver does not exits then adding it successfully.

sequence diagram: