BusPlanner	Version: 1.2
Requirements definition	Date: 11-11-2016

Distributed **S**oftware **D**evelopment: **BusPlanner** **Requirements Definition**

Revision History

Date	**Version**	**Description**	**Author**
11-11-2016	1.0	Initial draft	Team
08-12-2016	1.1	Changes in use cases	Team
04-01-2017	1.2	Changes in use cases and sequence diagrams	I. Agosti

Contents

1 INTRODUCTION

1.1 Purpose of this document

The purpose of this document is to specify the functional and nonfunctional requirements of the project.

1.2 Document organization

The document is organized as follows:

- Section 1, Introduction section describes the content of this document.
- Section 2, Functional requirements section describes the functional requirements of the project as Use cases, Use case descriptions with activity diagrams, and sequence diagrams.
- Section 3, Nonfunctional requirements section describes nonfunctional requirements such as availability, security, privacy, data redundancy and performances.

1.3 Intended audience

The intended audience of this document is:

- Development team, as a guidance during the development activities and for the team to ensure they understand the requirements of the project.
- The supervisors who can use this document to understand the future process of the project.
- The customer who can ensure that all the requirements are captured by the team.

1.4 Scope

This document provides the high level requirements description of the project. Both the functional and nonfunctional requirements of the projects are presented using some UML diagrams such as use case diagrams, sequence diagrams and activity diagrams.

1.5 Definitions and acronyms

1.5.1 Definitions

Keyword	**Definitions**
User	A person who requests for bus by being from bus stop.
Fleet Manager	Who owns the buses. He/she wants to know the utilization of buses and scheduling of buses.
User Request	Information generated with timestamps for the scheduling purpose.
Algorithm	A method used to enhance the scheduling process which is static as well as dynamic.
Sprint	A repeatable work cycle which is also known as iteration.

1.5.2 Acronyms and abbreviations

Acronym/abbreviation	**Definitions**
UI	User Interface
GUI	Graphical User Interface
MDH	Mälardalens Högskola, Västerås, Sweden
POLIMI	Politecnico di Milano, Milan, Italy
QA	Quality Assurance
DSD	Distributed Software Development

2 FUNCTIONAL REQUIREMENTS

2.1 Actors

- **Fleet manager**, who performs the following activities:
  - Login.
  - Get bus location.
  - Add/Remove/Modify bus.
  - Assign drivers to buses.
  - Add/Remove route.
  - Add/Remove/Modify driver.
  - View the user requests on a map.
  - View previous user requests.
  - View routes utilization.
- **Bus driver**, who performs the following activities:
  - Login.
  - View schedule with user requests.
- **Passenger**, who generates the user requests for a bus, specifying at which stop he/she wants to get on and off the bus.

2.2 User stories and related requirements

User stories are short and simple sentences that contain the features customers expect to find into the system. The customer's requirements are not equally important; for this reason high, average or low priority is attributed to each of them.

ID	User story	Priority	Use case
UserStory1	As fleet manager I want to be able to login (or logout) into the system with my account at any time.	High	Login.
UserStory2	As fleet manager I want to be able to add, modify or remove a bus.	High	Add bus. Modify bus. Remove bus.
UserStory3	As fleet manager I want to be able to add, modify or remove a driver.	High	Add driver. Modify driver. Remove driver.
UserStory4	As fleet manager I want to be able to view user requests on a map.	High	View user requests.
UserStory5	As fleet manager I want to be able to view previous user requests.	High	Previous user requests.
UserStory6	As fleet manager I want to be able to get the position of all the buses.	High	Get buses location.
UserStory7	As fleet manager I want to be able to get the utilization of all the routes.	Average	View routes utilization.
UserStory8	As fleet manager I want to be able to add or remove a route.	High	Add route. Remove route.
UserStory9	As a bus driver I want to be able to login (or logout) into the system with my account at any time.	High	Login.
UserStory10	As a bus driver I want to be able to see the schedule of the route I have to cover, with the user requests I need to satisfy.	High	View schedule.

2.3 Use cases

The following functional requirements describe the systems behavior with respect to the BusPlanner project and its actors.

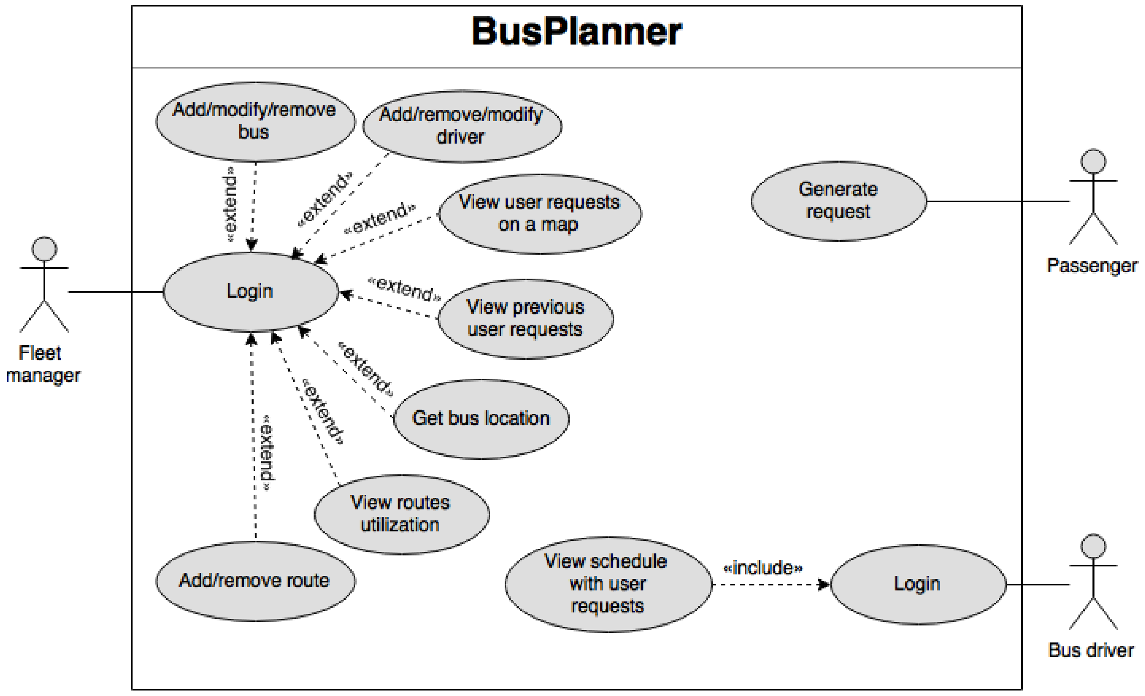


Figure 1: BusPlanner use case

2.4 Use case description

2.4.1 Passenger

Name	Generate request [Sequence diagram]
Actor	Passenger
Entry conditions	No entry conditions.
Flow of Events	1. Open web page. 2. Select a starting bus stop. 3. Select an ending bus stop. 4. Check bus availability.
Exit Conditions	Passenger gets confirmation.
Exceptions	No bus/seat available.

This use case does not correspond to any user story because our user requests will be simulated. So users will not actually be able to generate requests for a bus.

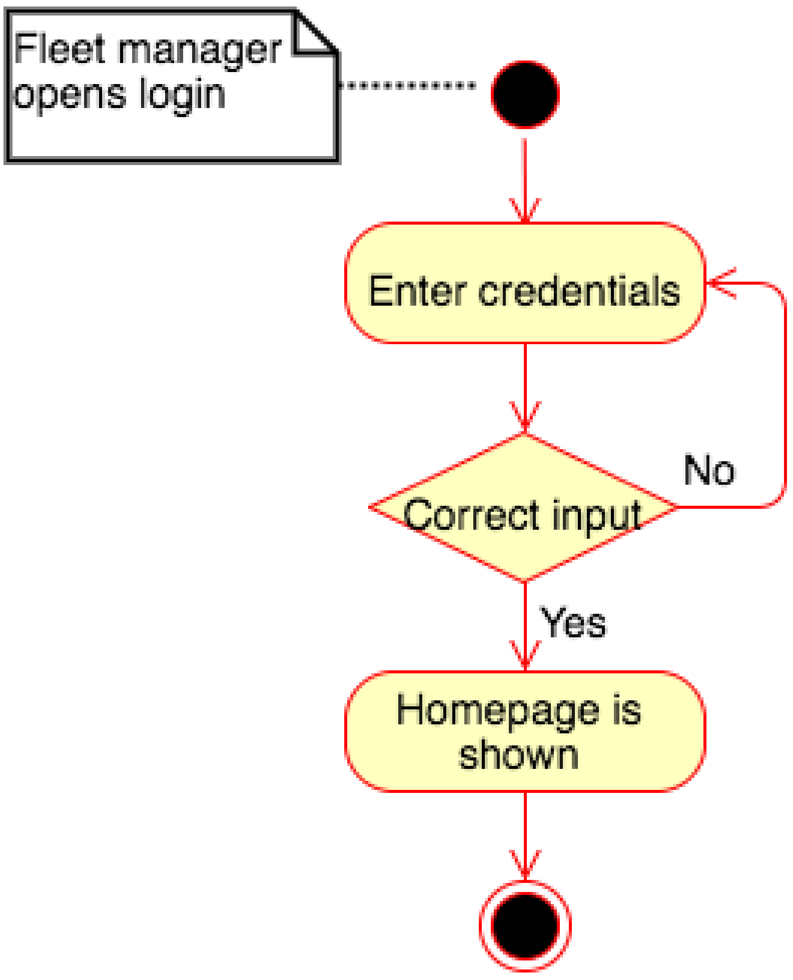
In the real world users can request a bus only if they are at a bus stop (or near one). The bus driver will be able to see the request and realize how many passengers will be at each bus stop. Requests are useful also for fleet managers, whose duty will be to assign a bus (buses are of two different sizes) to a route based on the number of requests on that same route.

The routes are five and we cannot change them. The only thing that can change is the type of bus that is assigned to a route every day.

2.4.2 Fleet manager

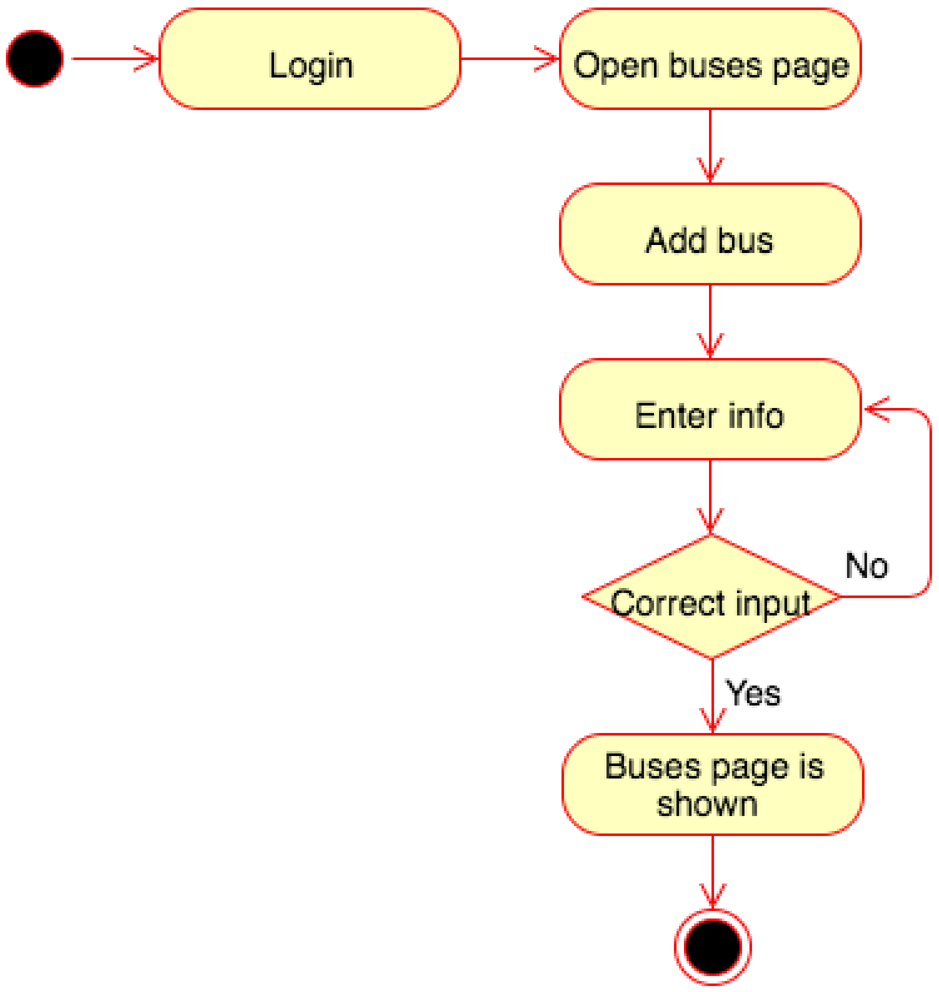
- Login:

Name	Login [Sequence diagram]
Actor	Fleet manager
Entry conditions	No entry condition.
Flow of Events	1. Web page opened. 2. Enter credentials. 3. "Login" button pressed.
Exit Conditions	Homepage is shown.
Exceptions	Wrong credentials.



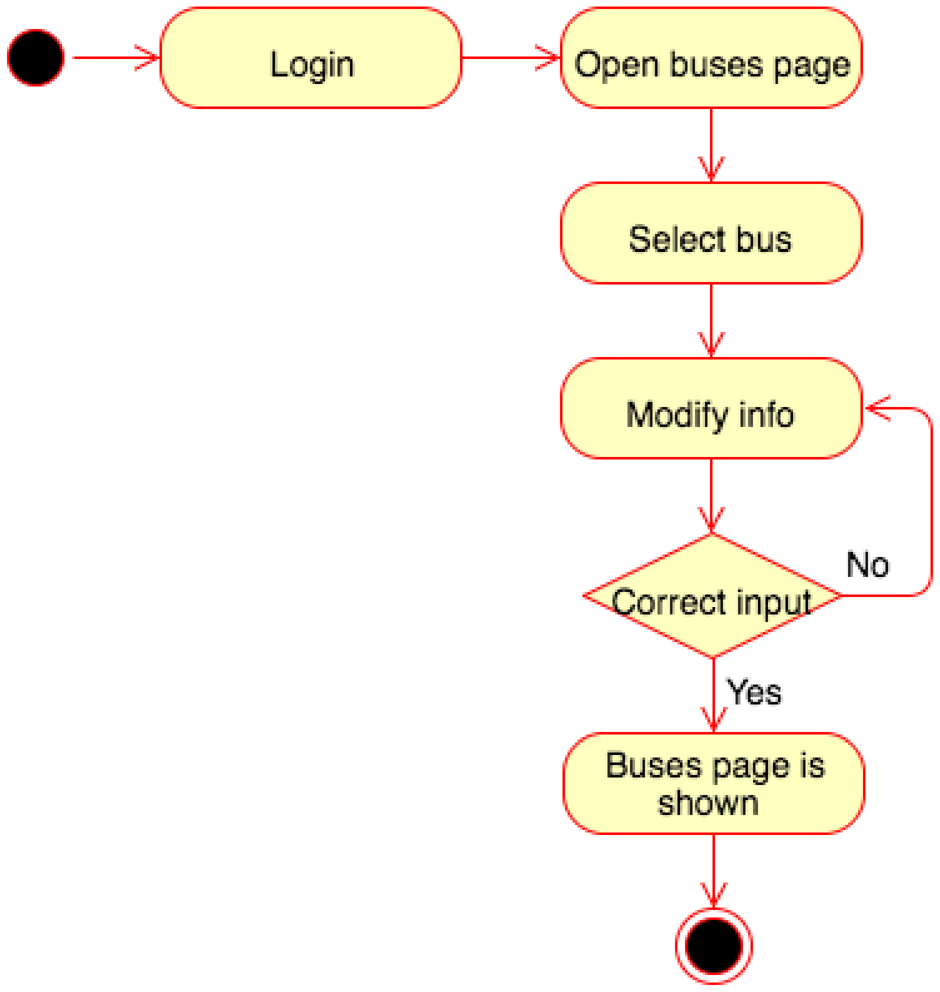
- Add bus:

Name	Add bus [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open buses page. 2. Form is filled with bus technical details. 3. Submit button pressed.
Exit Conditions	Database confirmation and buses page shown.
Exceptions	Wrong information is entered.



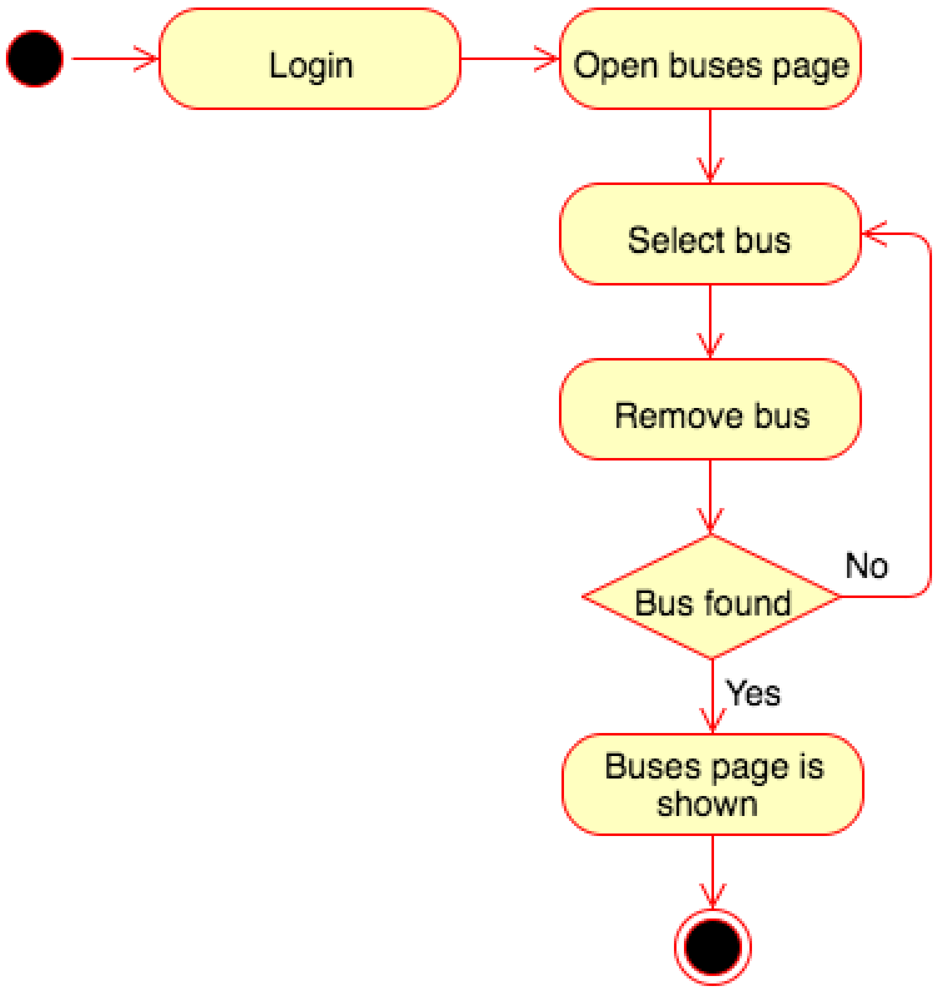
- Modify bus:

Name	Modify bus [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open buses page. 2. Bus is selected. 3. Form with bus technical details is modified. 4. Submit button pressed.
Exit Conditions	Database confirmation and buses page shown.
Exceptions	Wrong information is entered.



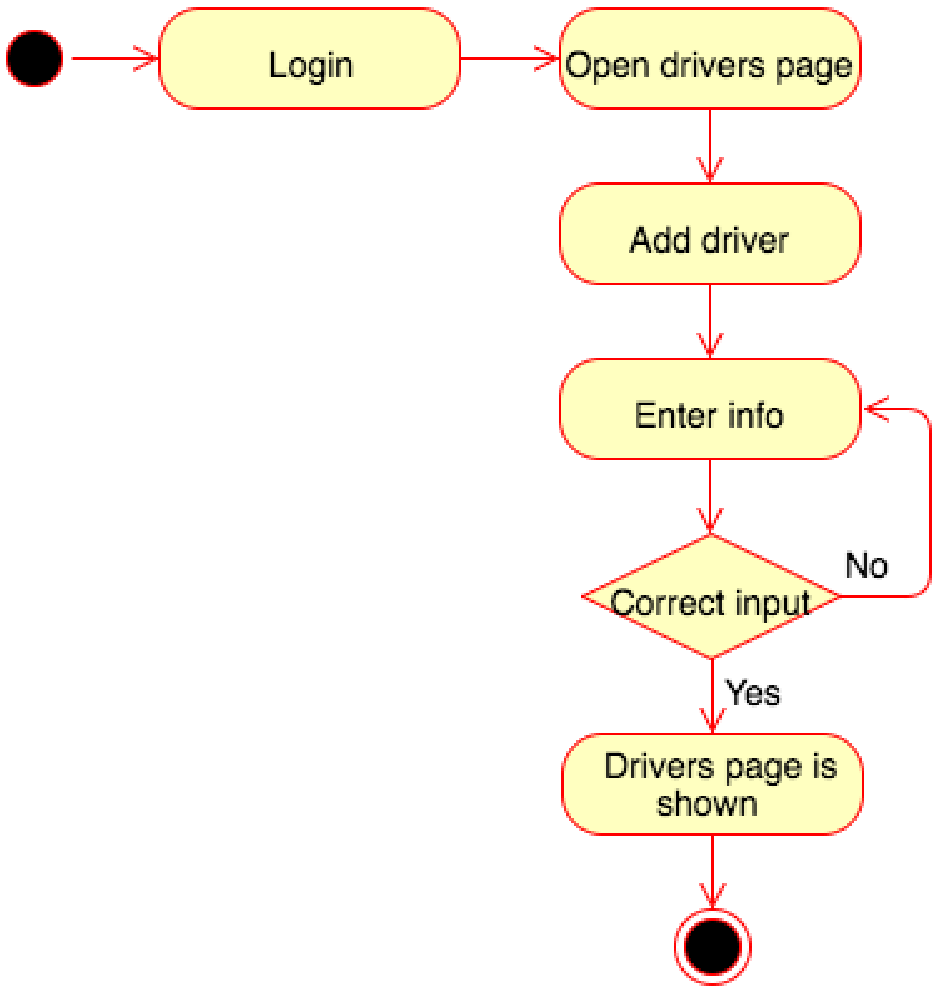
- Remove bus:

Name	Remove bus [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open buses page. 2. Bus is selected. 3. "Delete" button pressed.
Exit Conditions	Bus is removed and buses page is shown.
Exceptions	Bus not found.



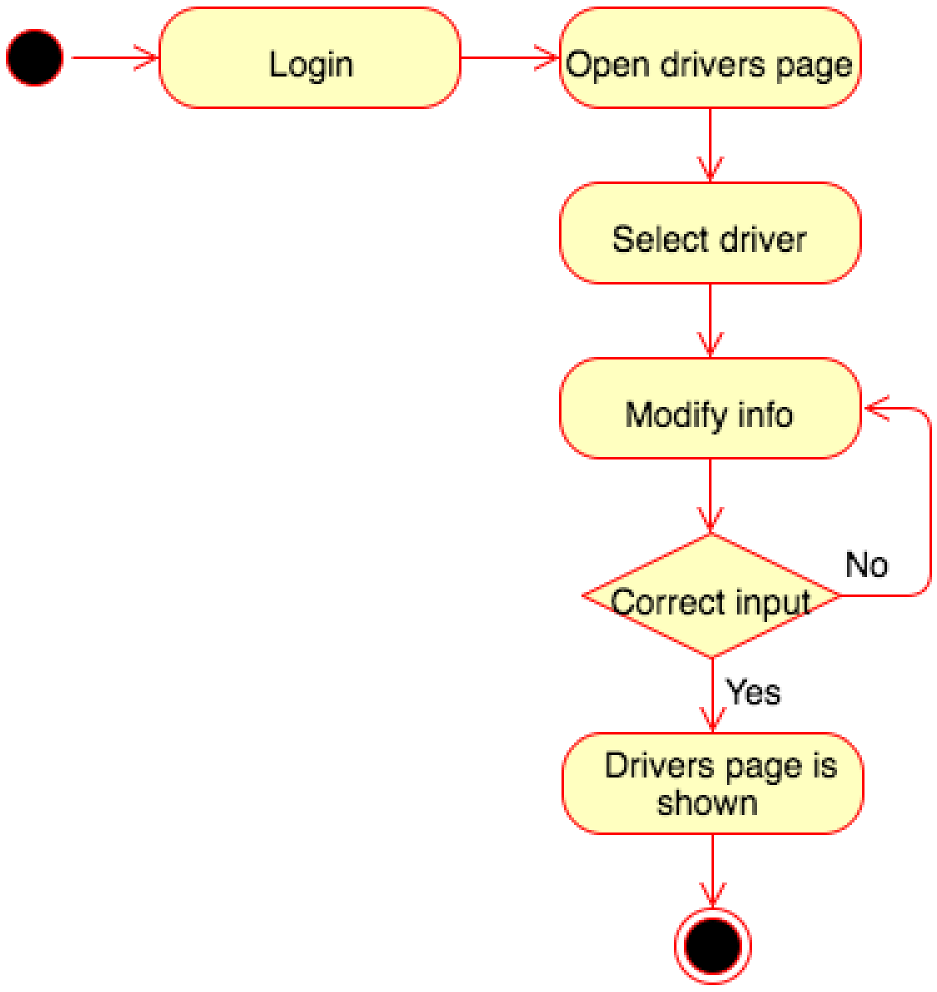
- Add driver:

Name	Add driver [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open drivers page. 2. Form is filled with driver's technical details. 3. Submit button pressed.
Exit Conditions	Database confirmation and drivers page shown.
Exceptions	Wrong information is entered.



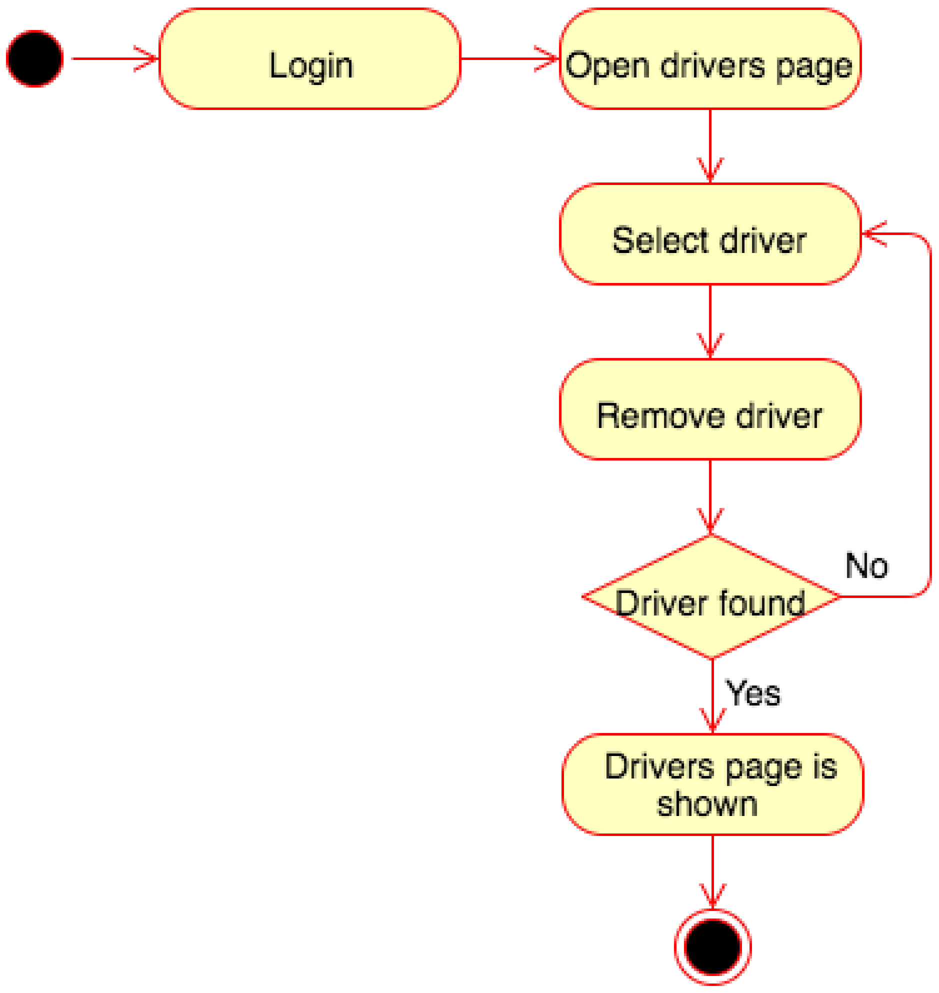
- Modify driver:

Name	Modify driver [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open drivers page. 2. Driver is selected. 3. Form with driver's technical details is modified. 4. Submit button pressed.
Exit Conditions	Database confirmation and drivers page shown.
Exceptions	Wrong information is entered.



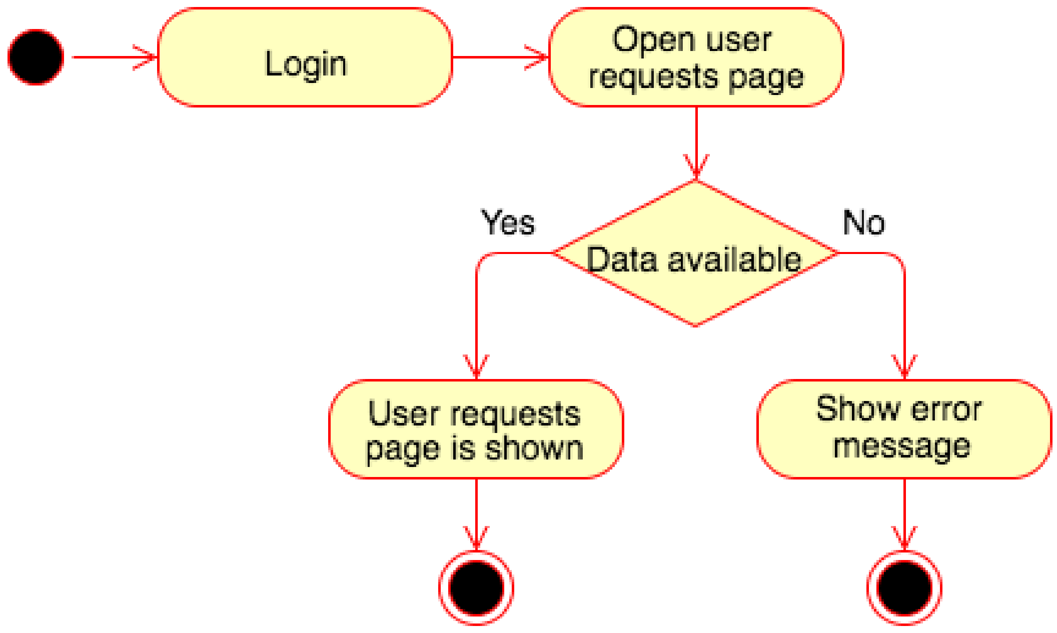
- Remove driver:

Name	Remove driver [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open drivers page. 2. Driver is selected. 3. "Delete" button pressed.
Exit Conditions	Driver is removed and drivers page is shown.
Exceptions	Driver not found.



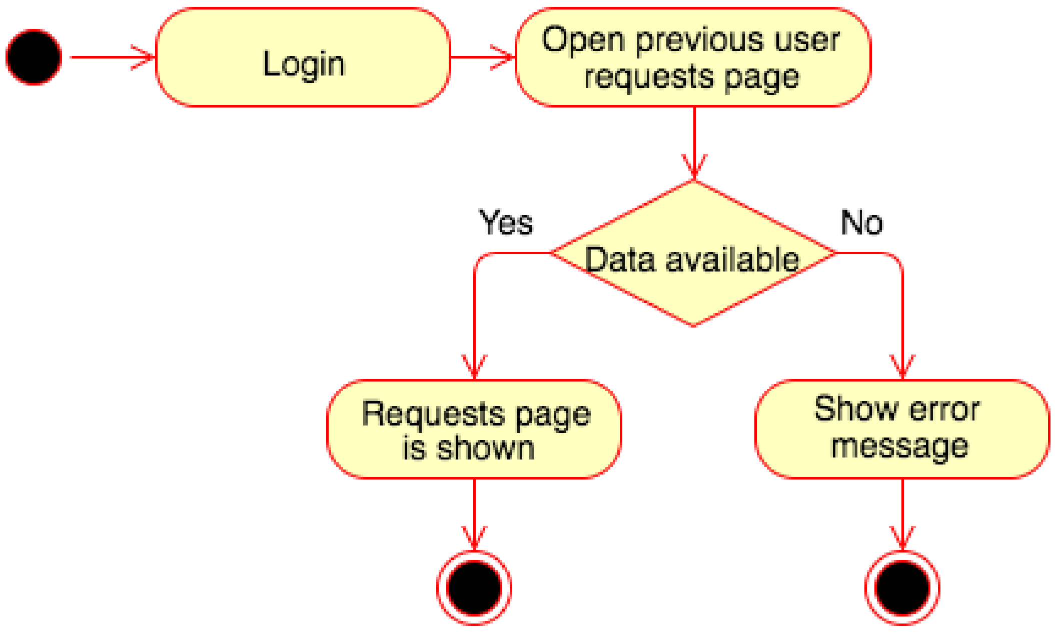
- View user requests on a map:

Name	View user requests on a map [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open user requests page.
Exit Conditions	The fleet manager is able to see the user requests on a map.
Exceptions	Data is not available.



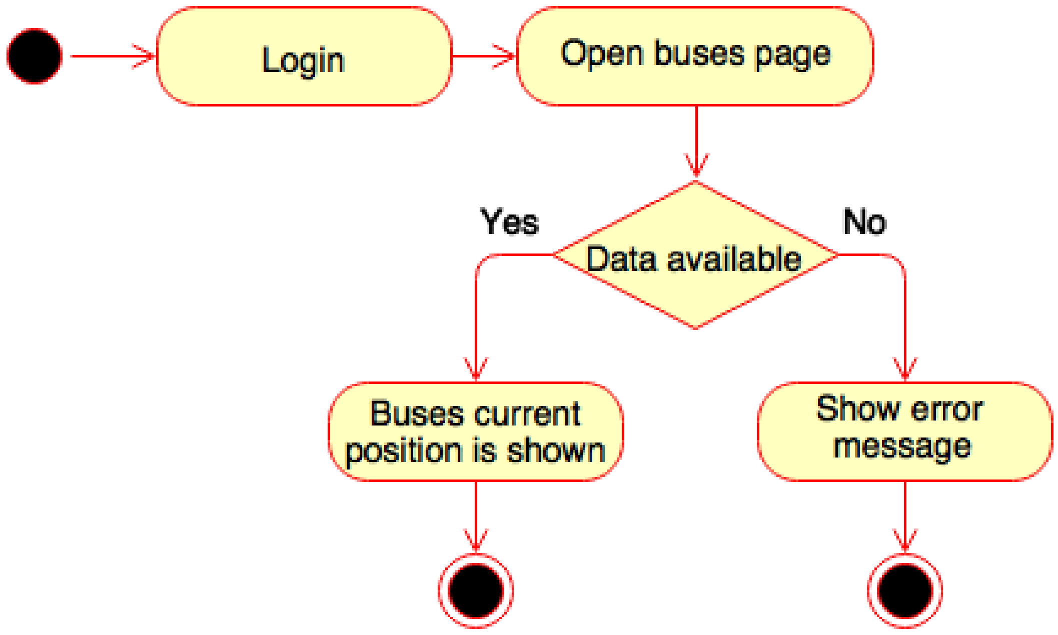
- View previous user requests:

Name	View previous user requests [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open user requests page.
Exit Conditions	Fleet manager is able to see the previous users requests in a table.
Exceptions	Data is not available.



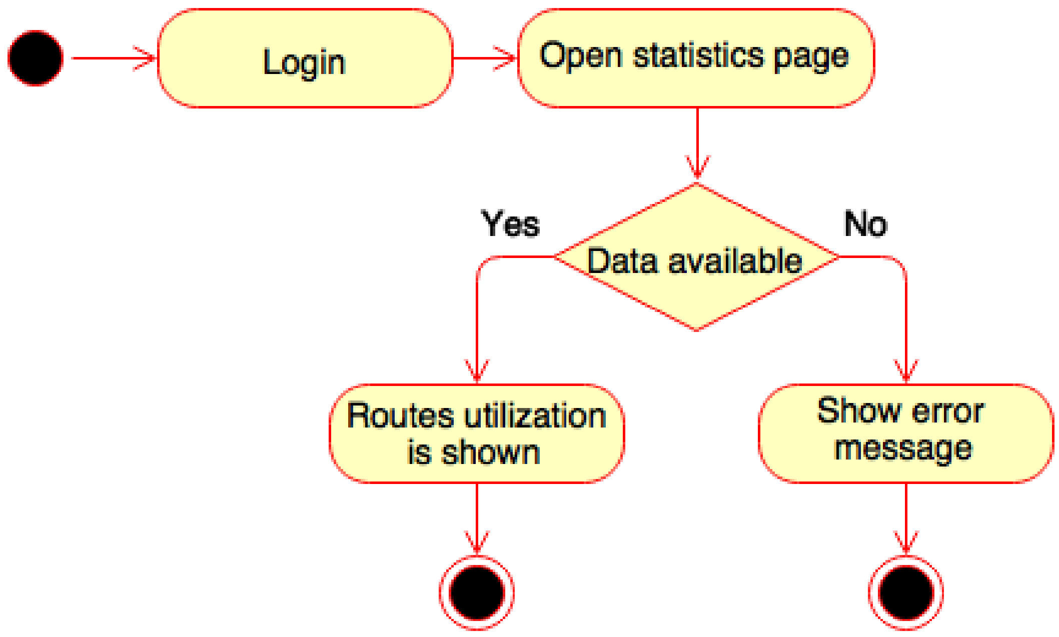
- Get buses location:

Name	Get bus location [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. From the homepage click on *Buses* button.
Exit Conditions	The fleet manager is able to view the buses location on a map.
Exceptions	Data not available.



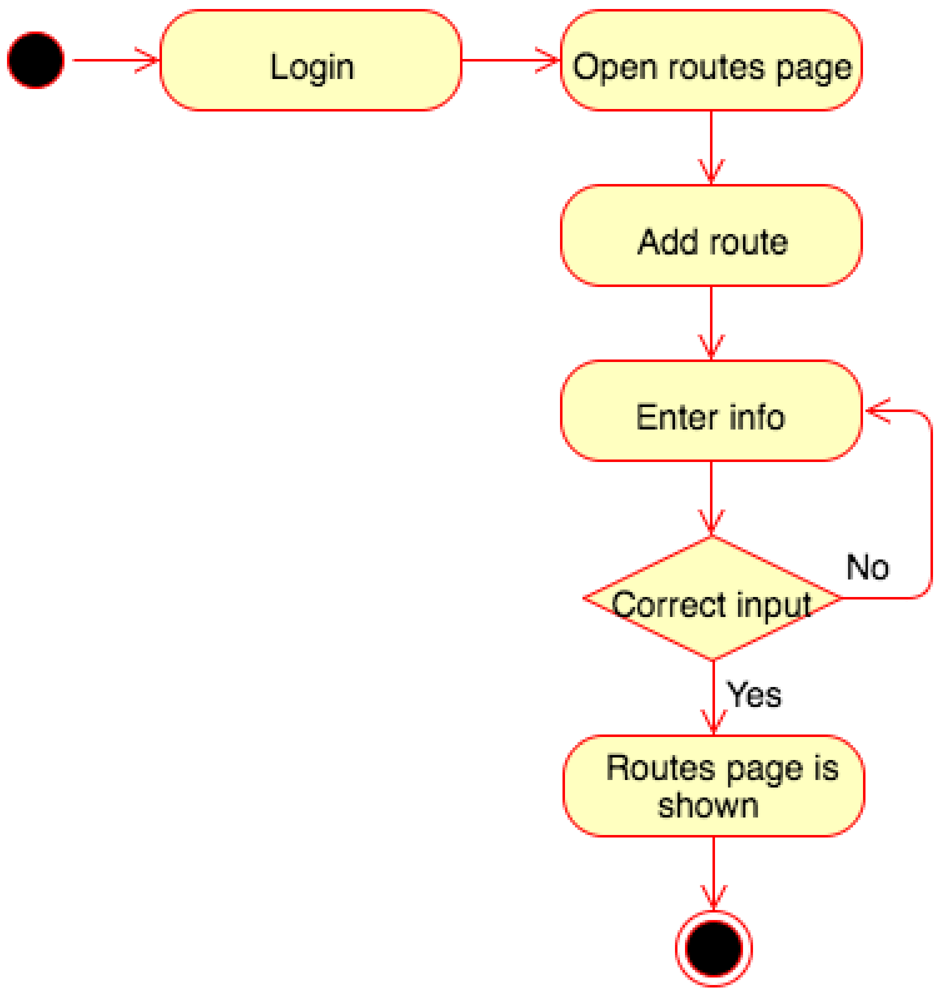
- View routes utilization:

Name	View bus utilization [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. From the homepage click on *Statistics* button.
Exit Conditions	The fleet manager is able to view the utilization of the company's routes.
Exceptions	Data not available.



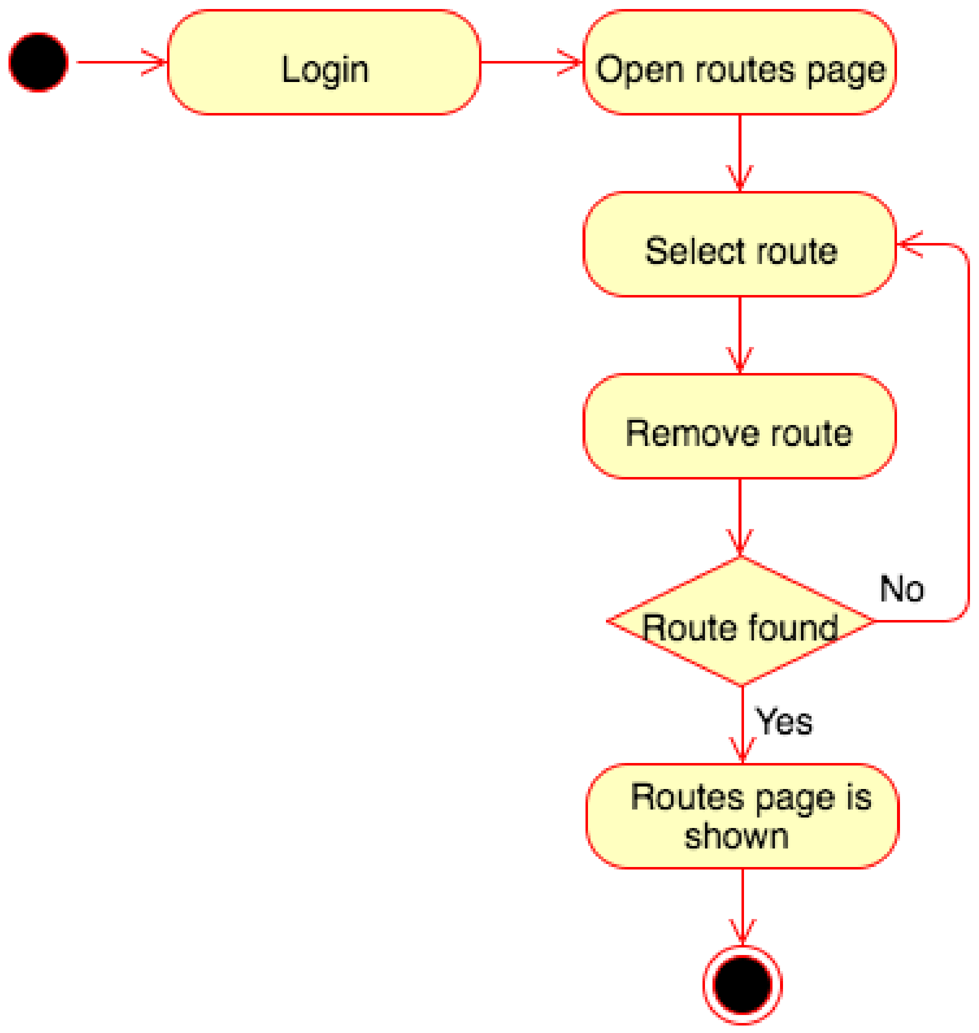
- Add route:

Name	Add route [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open routes page. 2. Form is filled with route's technical details. 3. Submit button pressed.
Exit Conditions	Database confirmation and routes page shown.
Exceptions	Wrong information is entered.



- Remove route:

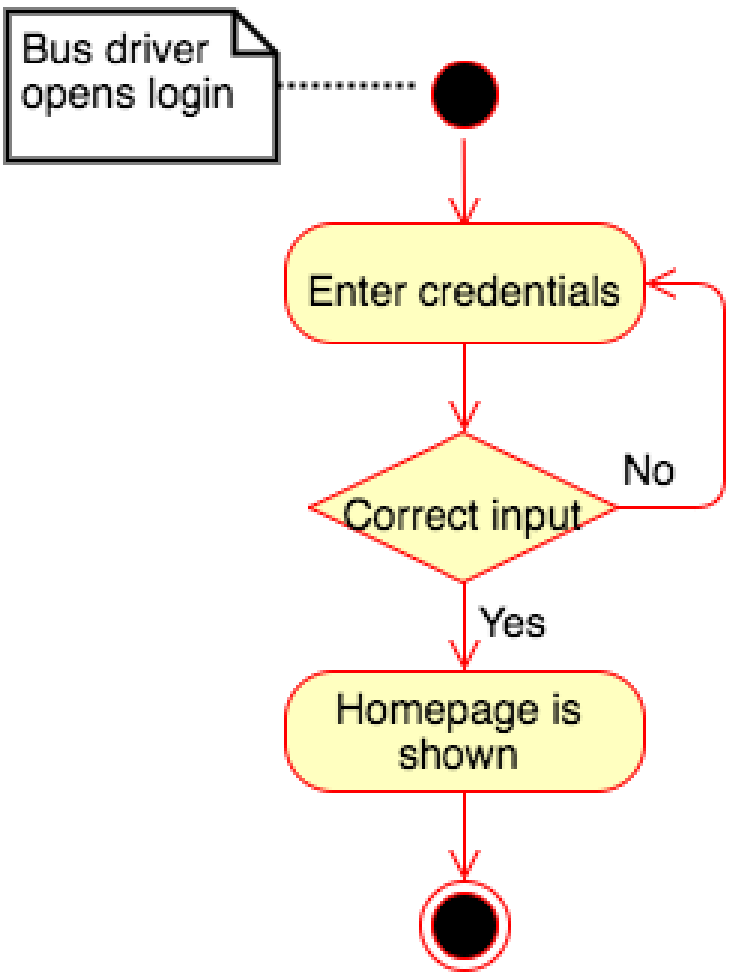
Name	Remove route [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Open routes page. 2. Route is selected. 3. "X" button pressed.
Exit Conditions	Route is removed and routes page is shown.
Exceptions	Route not found.



2.4.3 Bus driver

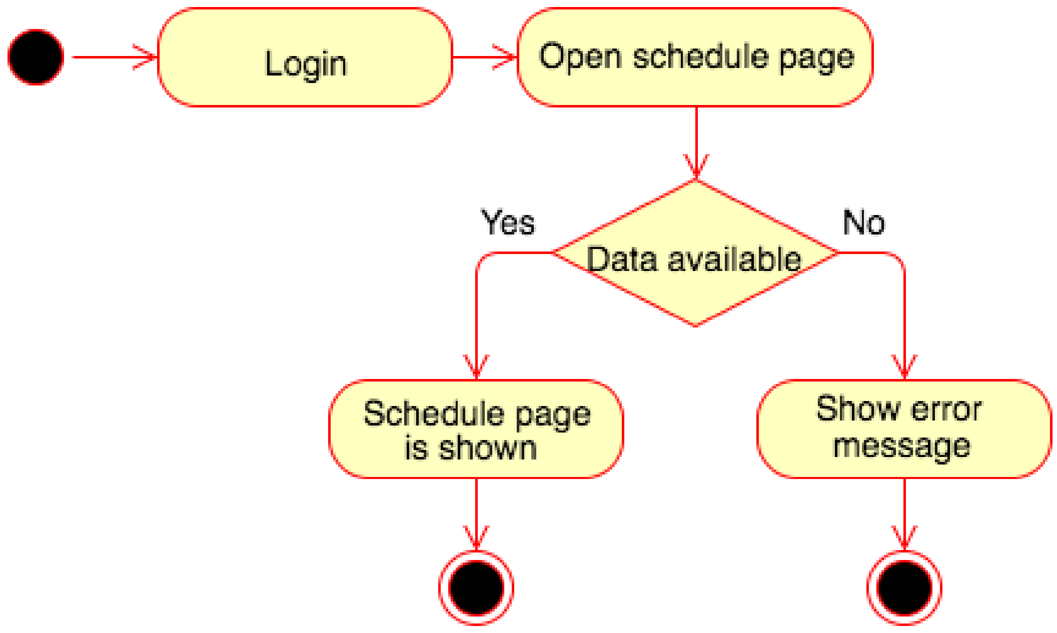
- Login:

Name	Login [Sequence diagram]
Actor	Bus driver
Entry conditions	No entry condition.
Flow of Events	1. Web page opened. 2. Enter credentials. 3. "Login" button pressed.
Exit Conditions	Homepage is shown.
Exceptions	Wrong credentials.



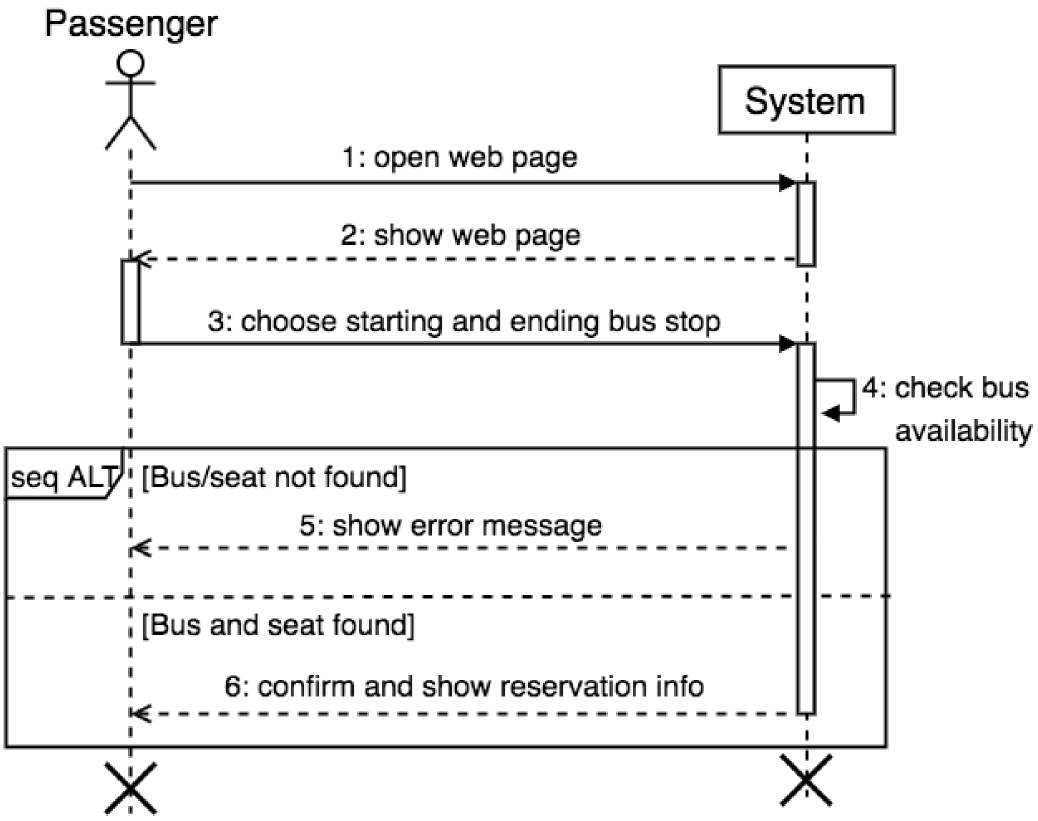
- View schedule:

Name	View schedule with user requests [Sequence diagram]
Actor	Bus driver
Entry conditions	Bus driver is logged in.
Flow of Events	1. Respective web page is opened. 2. Read data from database. 3. Information is shown on the screen.
Exit Conditions	Bus driver is able to view his/her schedule with all the user requests he/she needs to satisfy.
Exceptions	Data not available.

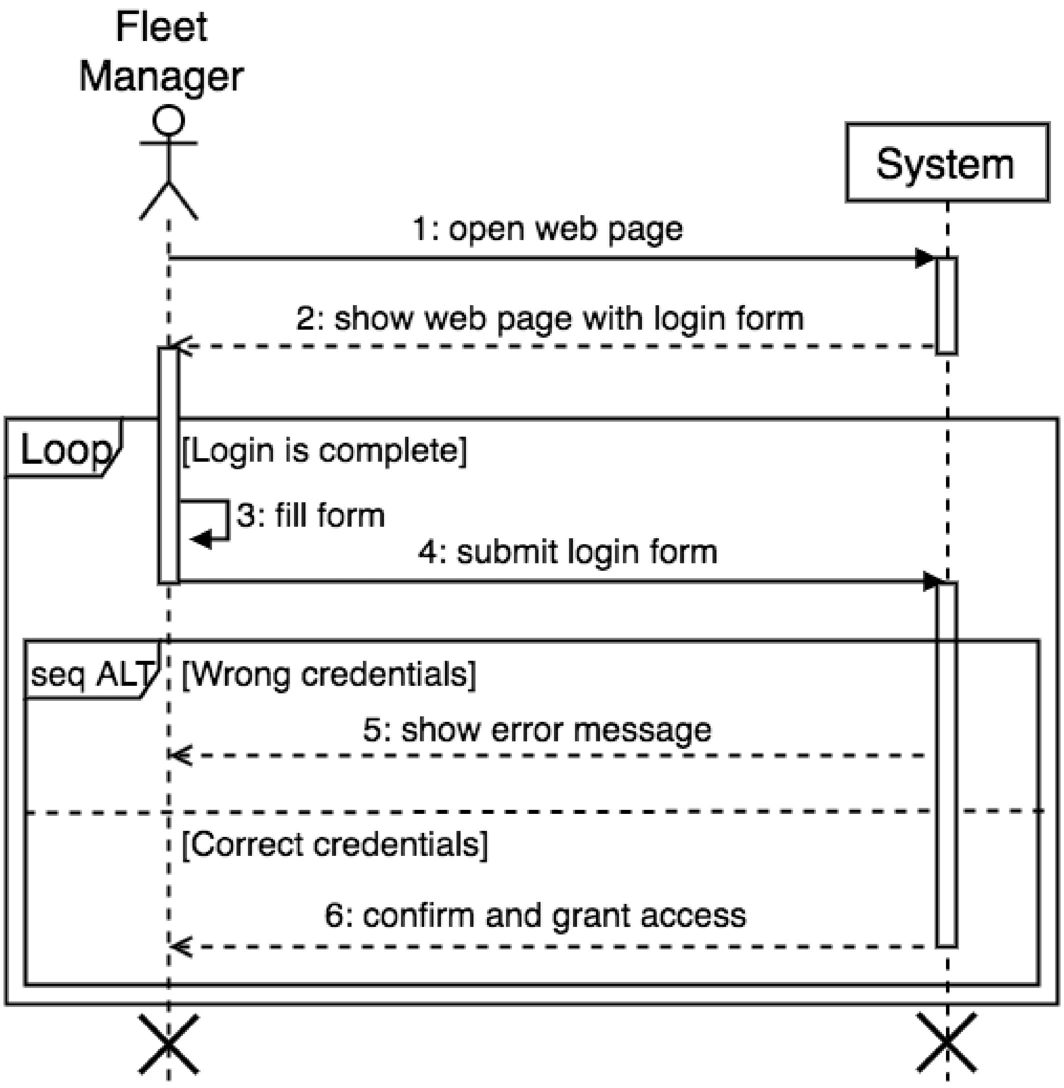


2.4.4 Sequence diagrams

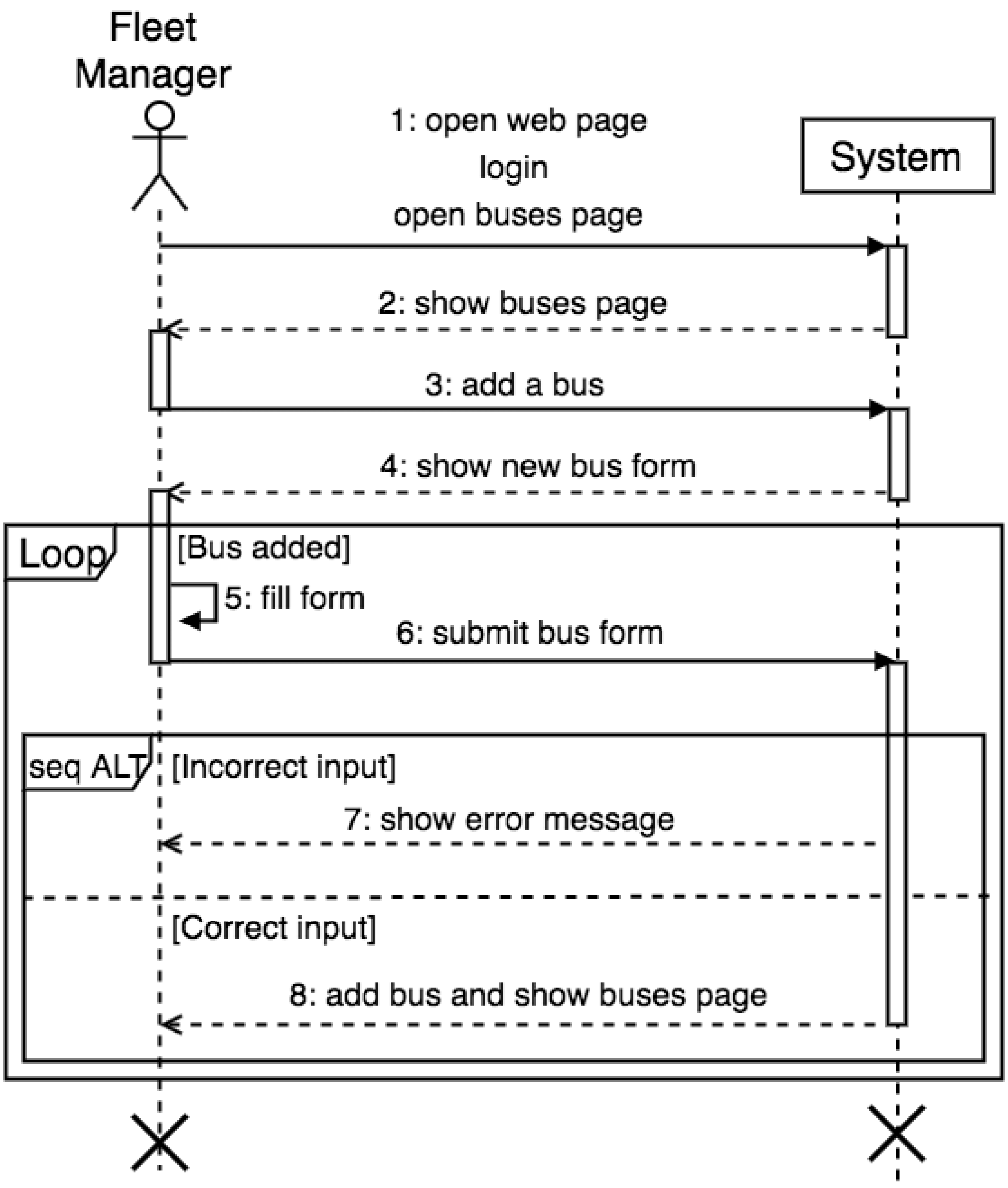
- Passenger generates a request:



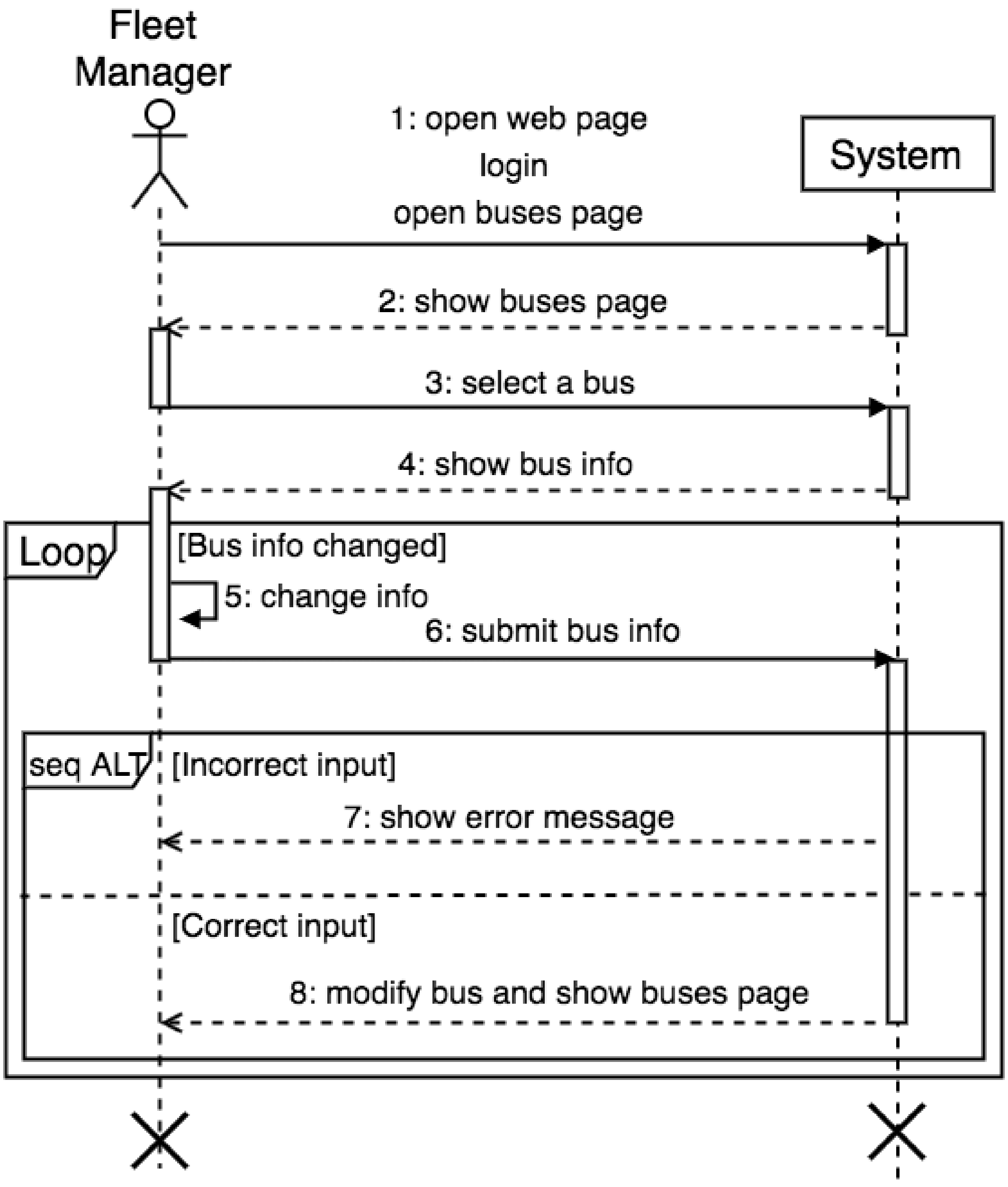
- Fleet manager login:



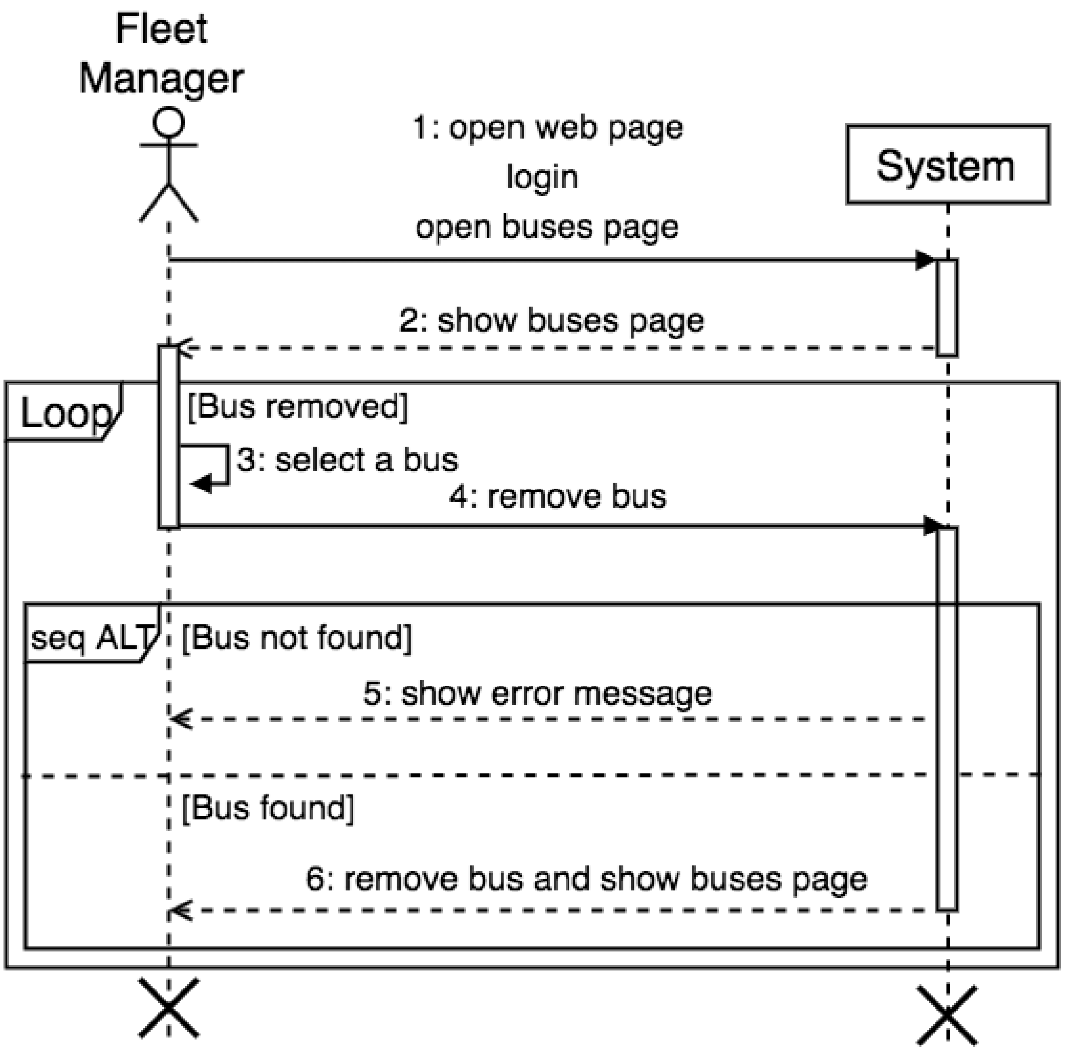
- Add bus:


Fleet Manager
System
1: open web page
login
open buses page
2: show buses page
3: add a bus
4: show new bus form
Loop
[Bus added]
5: fill form
6: submit bus form
seq ALT
[Incorrect input]
7: show error message
[Correct input]
8: add bus and show buses page

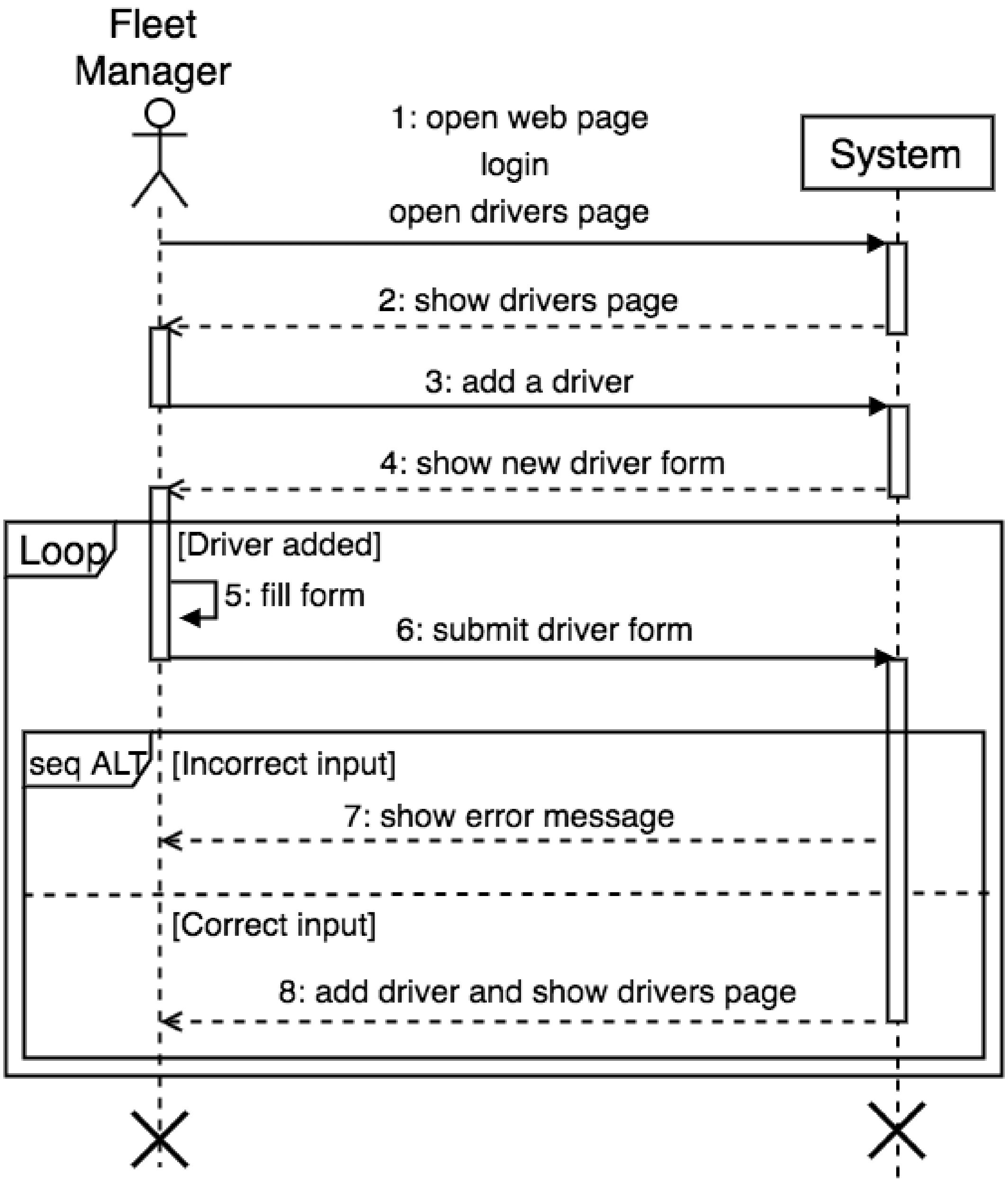
- Modify bus:


Fleet Manager
System
1: open web page
login
open buses page
2: show buses page
3: select a bus
4: show bus info
Loop
[Bus info changed]
5: change info
6: submit bus info
seq ALT
[Incorrect input]
7: show error message
[Correct input]
8: modify bus and show buses page

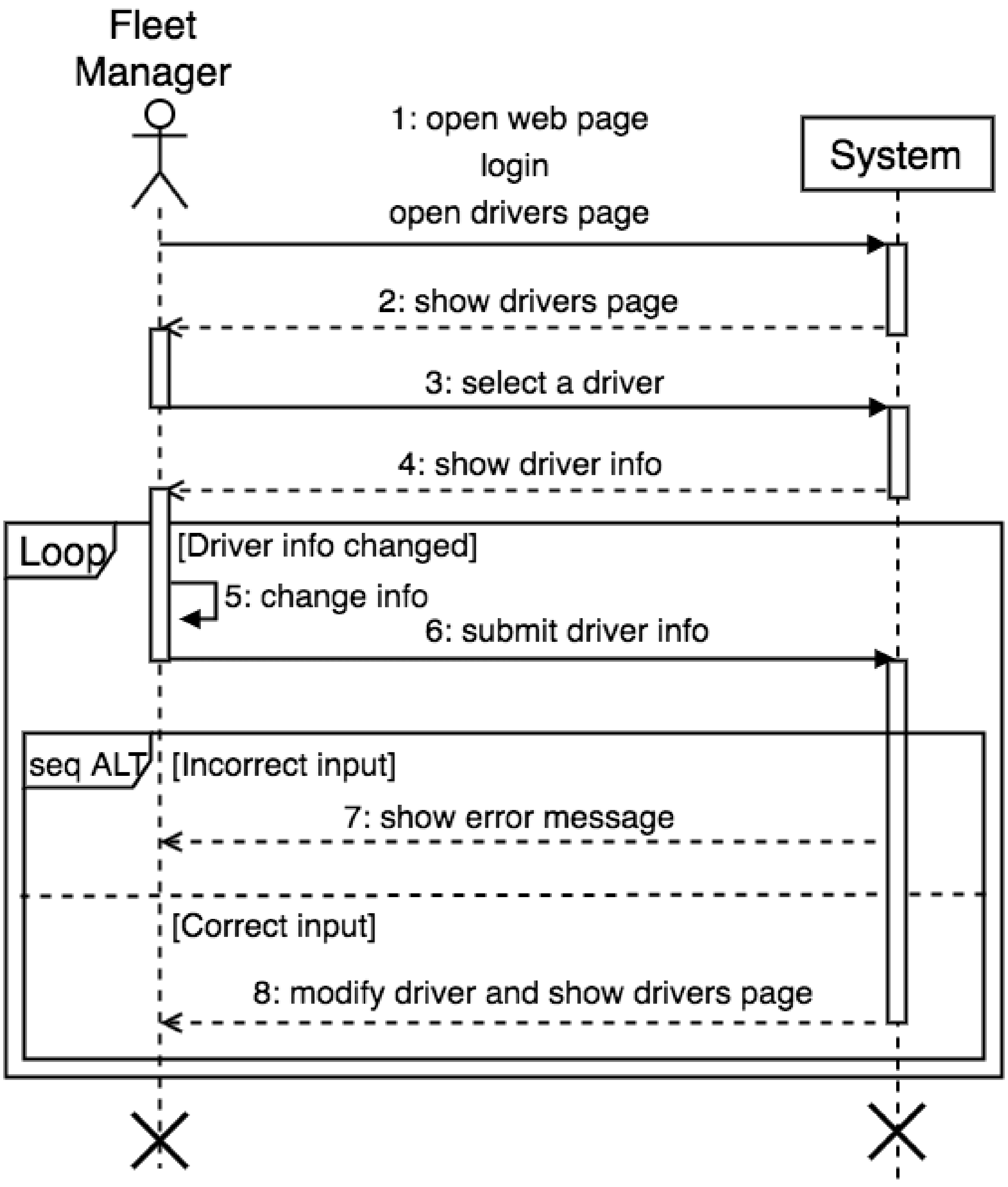
- Remove bus:


Fleet Manager
System
1: open web page
login
open buses page
2: show buses page
Loop
[Bus removed]
3: select a bus
4: remove bus
seq ALT
[Bus not found]
5: show error message
[Bus found]
6: remove bus and show buses page

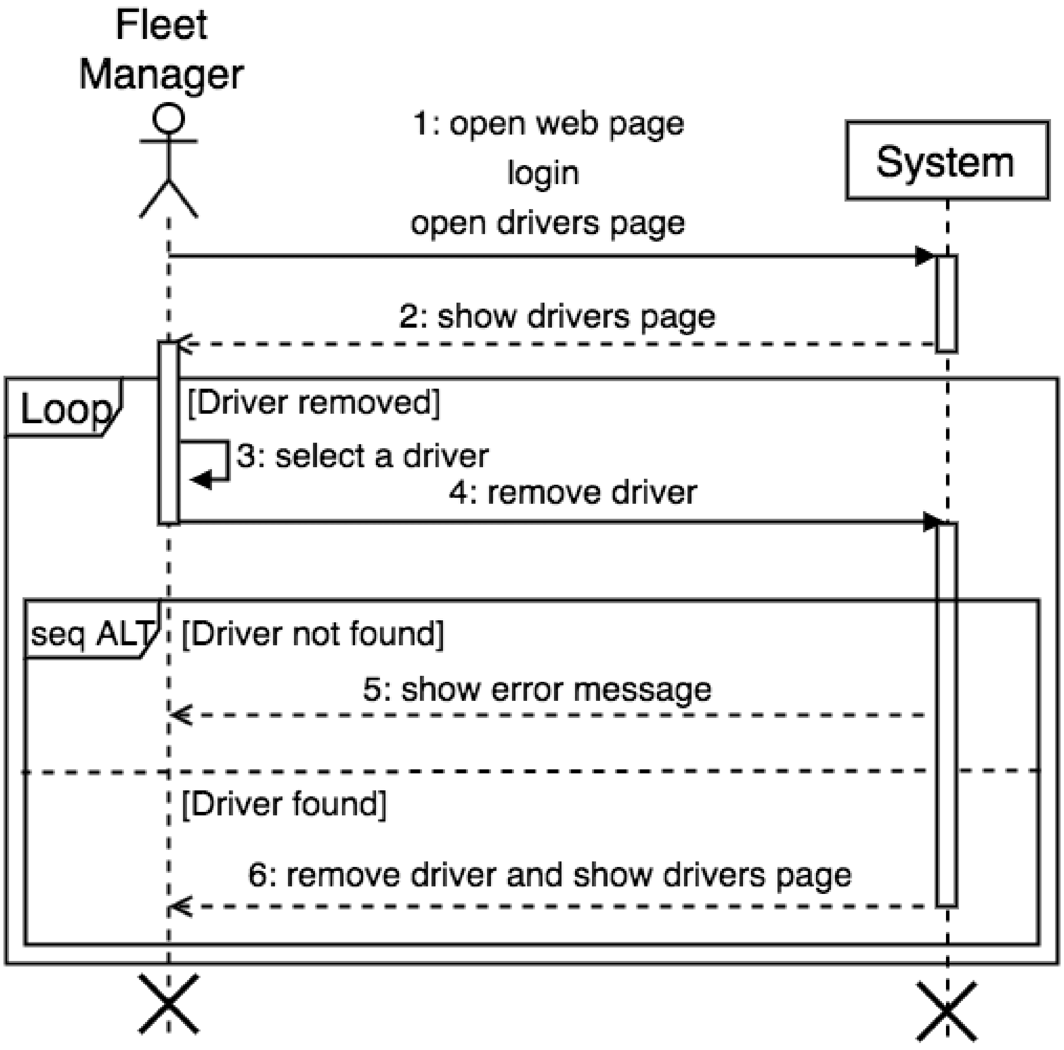
- Add driver:


Fleet Manager
System
1: open web page
login
open drivers page
2: show drivers page
3: add a driver
4: show new driver form
Loop
[Driver added]
5: fill form
6: submit driver form
seq ALT
[Incorrect input]
7: show error message
[Correct input]
8: add driver and show drivers page

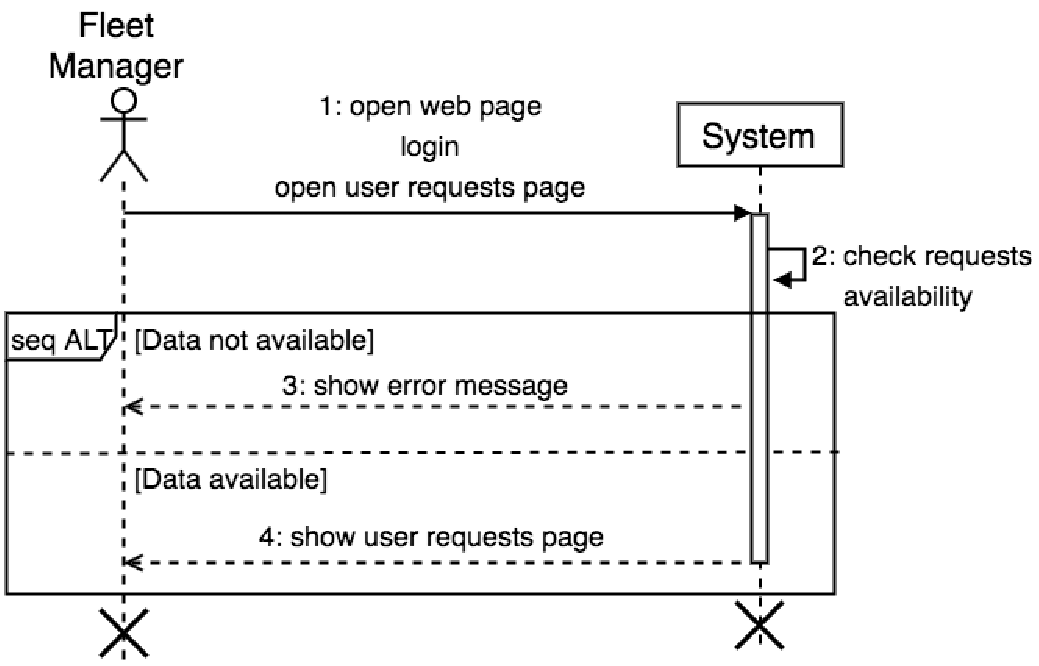
- Modify driver:



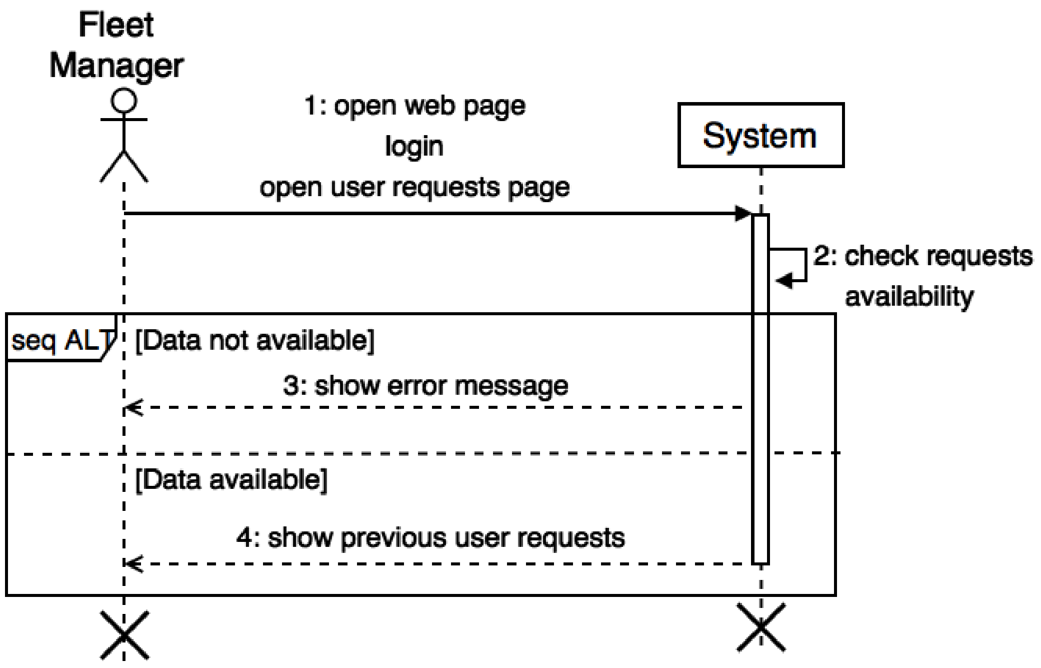
- Remove driver:



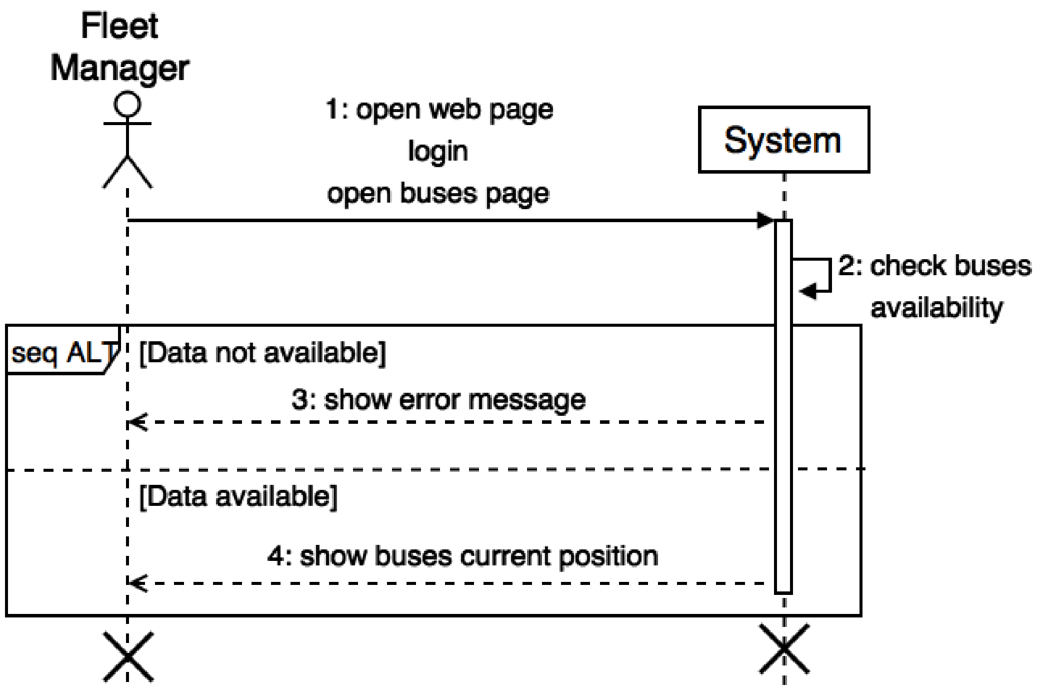
- View user requests on a map:



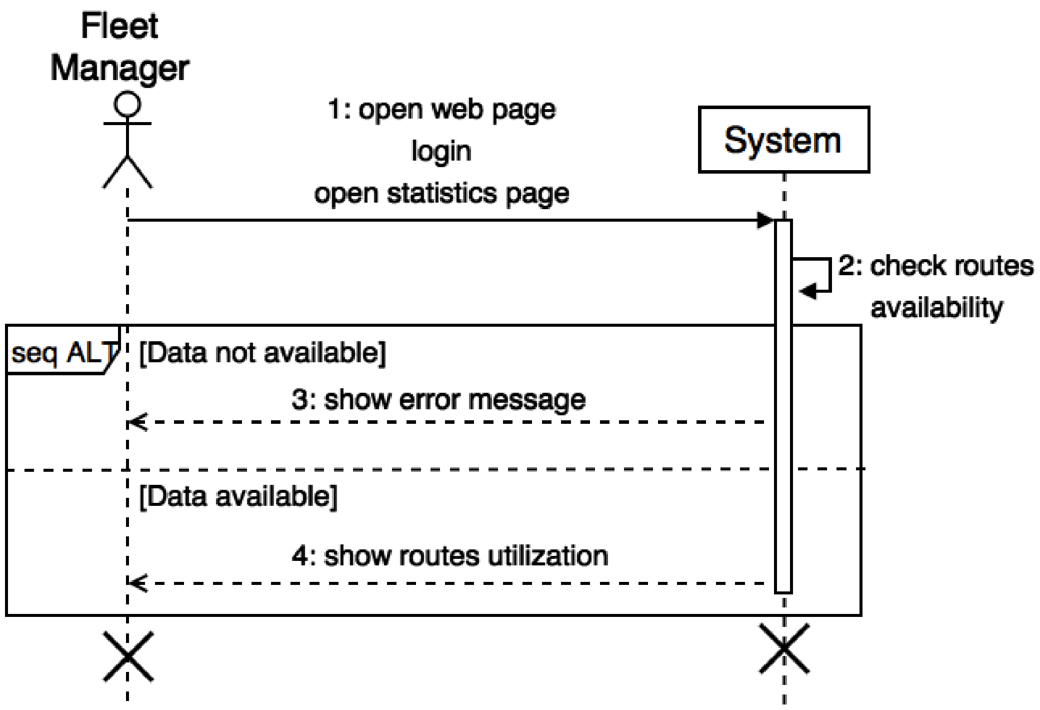
- View previous user requests:



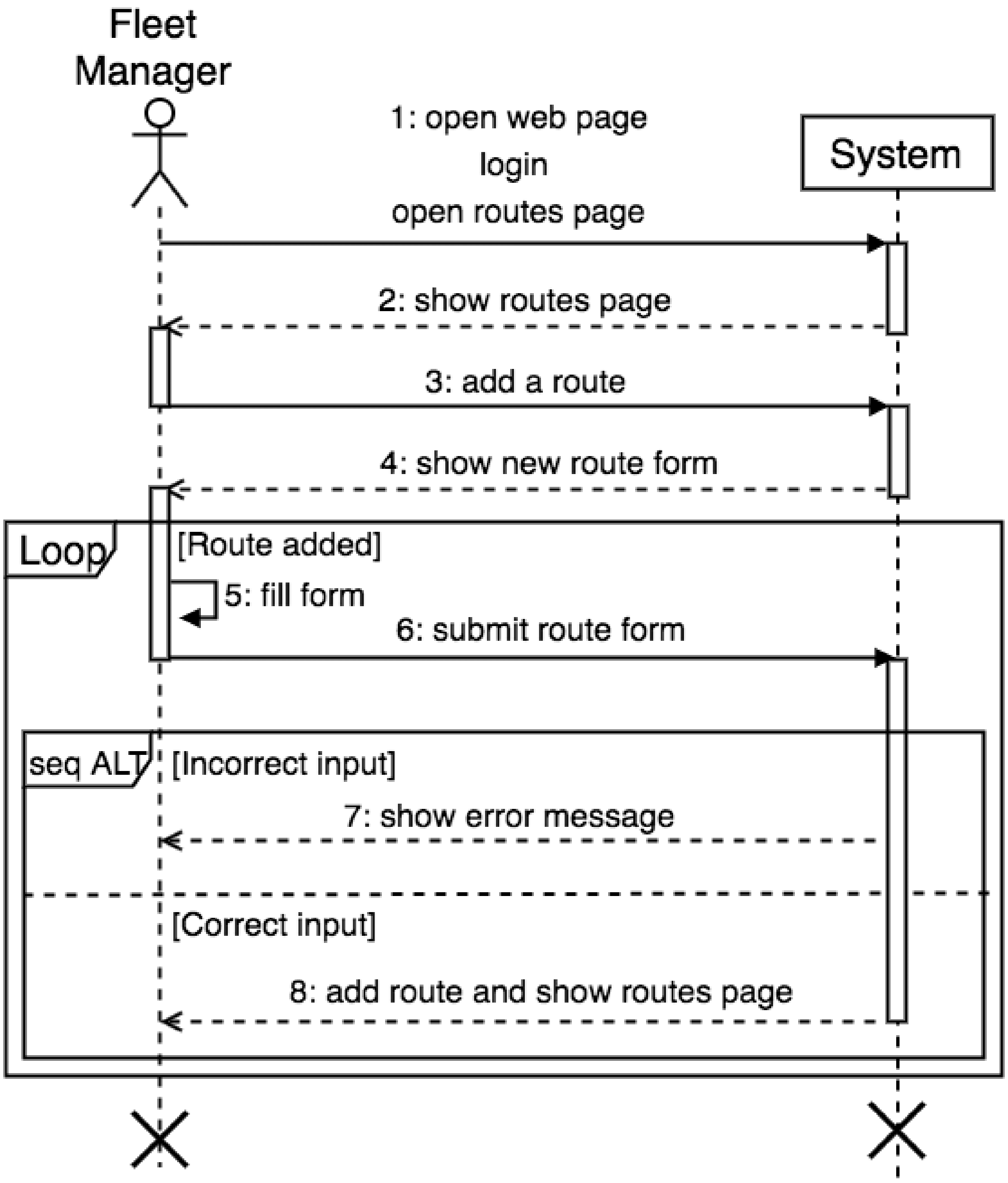
- Get buses location:



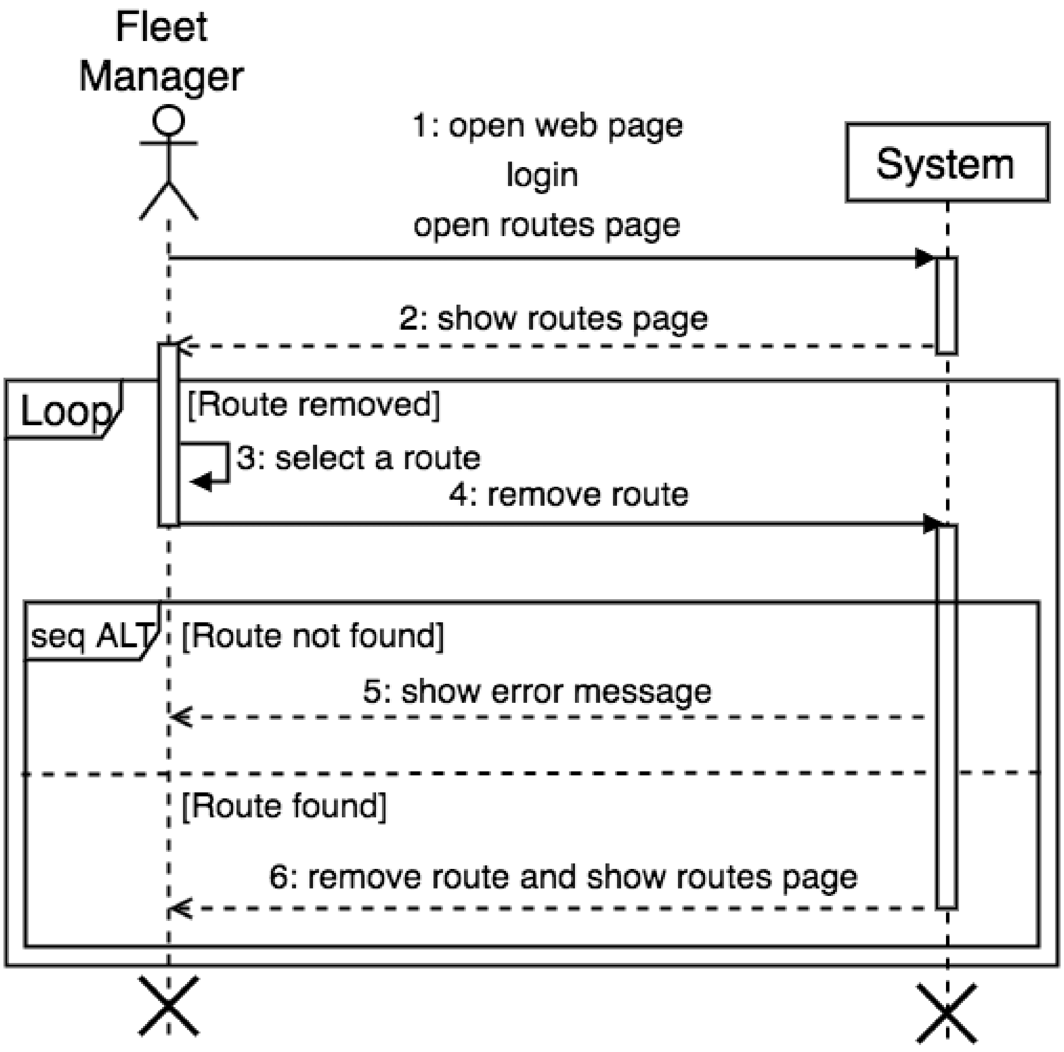
- View routes utilization:



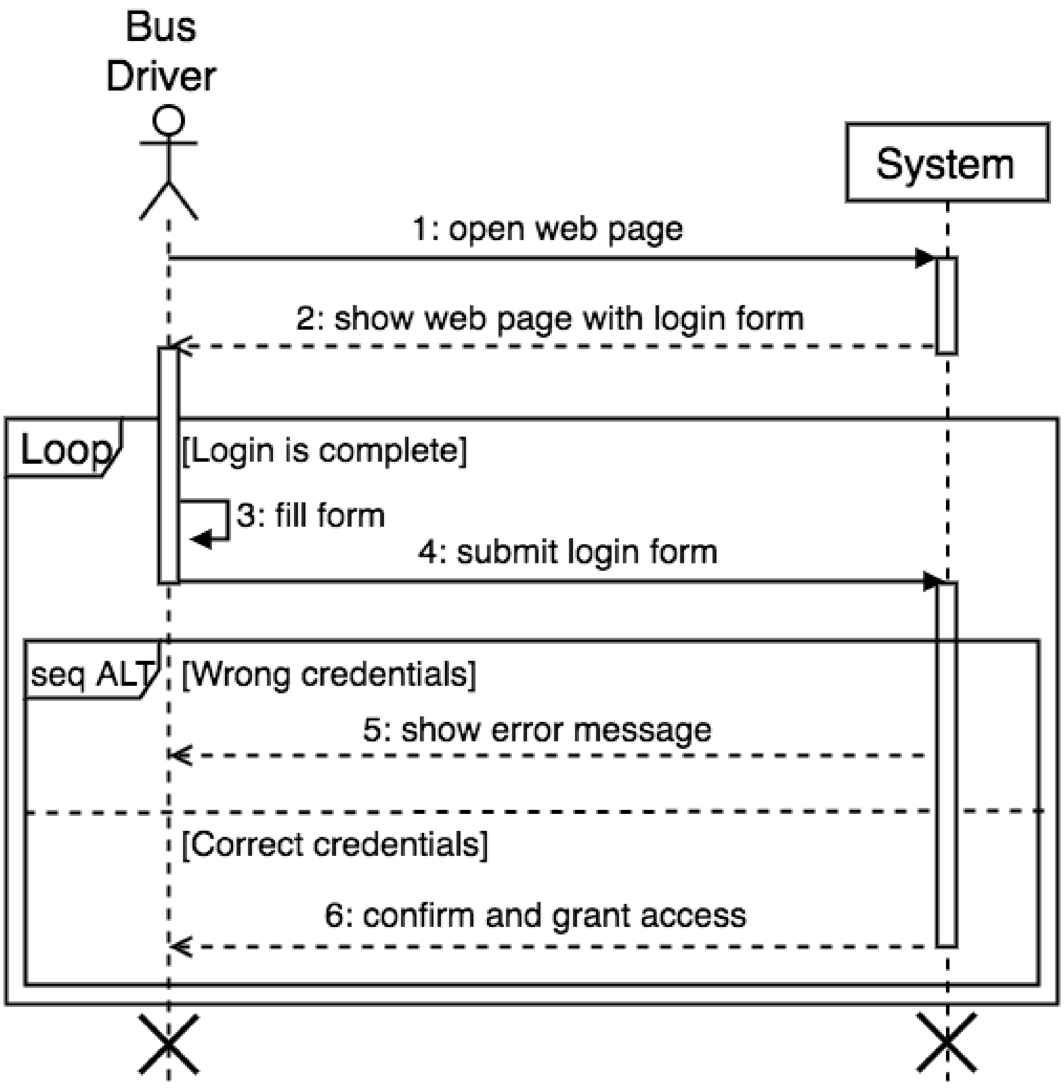
- Add route:


Fleet Manager
System
1: open web page
login
open routes page
2: show routes page
3: add a route
4: show new route form
Loop
[Route added]
5: fill form
6: submit route form
seq ALT
[Incorrect input]
7: show error message
[Correct input]
8: add route and show routes page

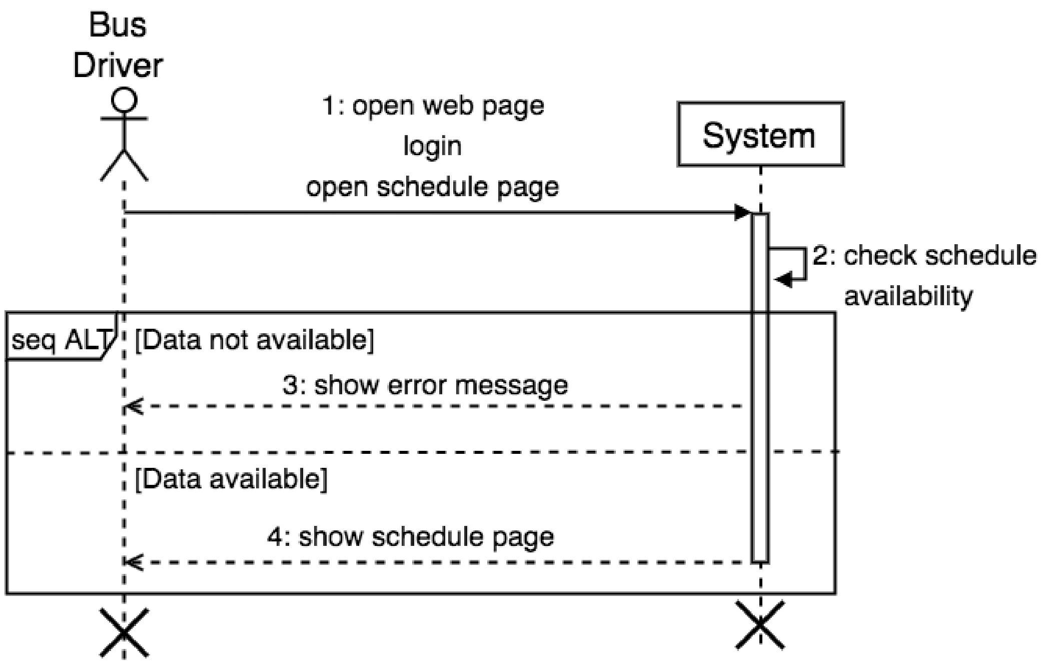
- Delete route:



- Bus driver login:



- View schedule:



3 NONFUNCTIONAL REQUIREMENTS

This section presents the nonfunctional requirements of our BusPlanner project, which describe the behavior of the system. We decided to divide them into 7 main categories.

3.1 Usability

- The application must be mobile responsive.
- The seat reservation should be done in the minimum number of steps possible.
- No fancy GUI.
- Correct and up to date information.
- Presenting data in a visible and understandable way.

3.2 External Libraries

- Our application makes usage of external libraries such as Bootstrap.

3.3 Compatibility issue

- The application is suggested to work only with the deployed version of the used libraries. Updated versions might bring incompatibilities.
- The maps being used should offer the possibility to work with PHP and SQL.

3.4 Security

- Only users of the application are allowed to use the project.
- The application needs to protect C.I.A elements (Confidentiality, Integrity and Availability) of user and nobody can see and change information of others.

3.5 Availability

Considering that the application:

- Can pave the way for users to take a bus as soon as possible with the aim of saving their time.
- Can make it easier for users to take a bus from everywhere in bus timetable.
- Can have friendly interface for users.
- Performance should provide the user a fast experience using the application.
- Has to handle user's request all the time using any device with an Internet connection and an installed web browser.

3.6 Uptime and data redundancy

The BusPlanner application should guarantee high availability and data redundancy. Still, since the application will be created in the context of the DSD course, our team will not build nor require any dedicated infrastructure for it and so estimating and proving exact value for data redundancy and uptime is not possible however, in the case there's the chance to build and test a dedicated infrastructure, an uptime of at least 99.99% is desirable along with at least one database replication.

3.7 Performances

The application has to be able to manage a high volume of requests. Since this application will be created in the context of the DSD course, our team will not build nor require any dedicated infrastructure for it. Furthermore, it is impossible to estimate and prove the exact value for performances. However, it should be easy to update it and improve it if needed.