BusPlanner	Version: 1.0
Requirements definition	Date: 2016-11-11

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

Revision History

Date	**Version**	**Description**	**Author**
2016-11-11	1.0	Initial draft	Team

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	Mlardalens Hgskola, Vsters, 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 fleet bus to the route.
  - Add/Remove/Modify route.
  - Add/Remove/Modify schedule time.
  - Map the user requests.
  - View previous user requests.
- **Bus driver**, who performs the following activities:
  - Login
  - Handle user request
  - View user request
  - View schedule
- **Passenger**, who generates the user requests for a 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 or remove a bus from a route.	High	Add bus to route. Remove bus from route.
UserStory3	As fleet manager I want to be able to add or remove a schedule time.	High	Add schedule time. Modify schedule time.
UserStory4	As fleet manager I want to be able to map user requests.	High	Mapping user request.
UserStory5	As fleet manager I want to be able to add or modify a route.	High	Add route. Modify route. Delete route.
UserStory6	As fleet manager I want to be able to get the position of all the buses.	High	Get bus location.
UserStory7	As fleet manager I want to be able to get the utilization of the selected bus.	Average	View bus utilization.
UserStory8	As a bus driver I want to be able to login (or logout) into the system with my account at any time.	High	Login.
UserStory9	As a bus driver I want to be able to see the schedule of the route I have to cover.	High	View schedule.
UserStory10	As a bus driver I want to be able to view the user requests from the passengers.	Average	View user requests.
UserStory11	As a bus driver i want to be able to manage the user request.	High	Manage user requests

2.3 Use cases

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

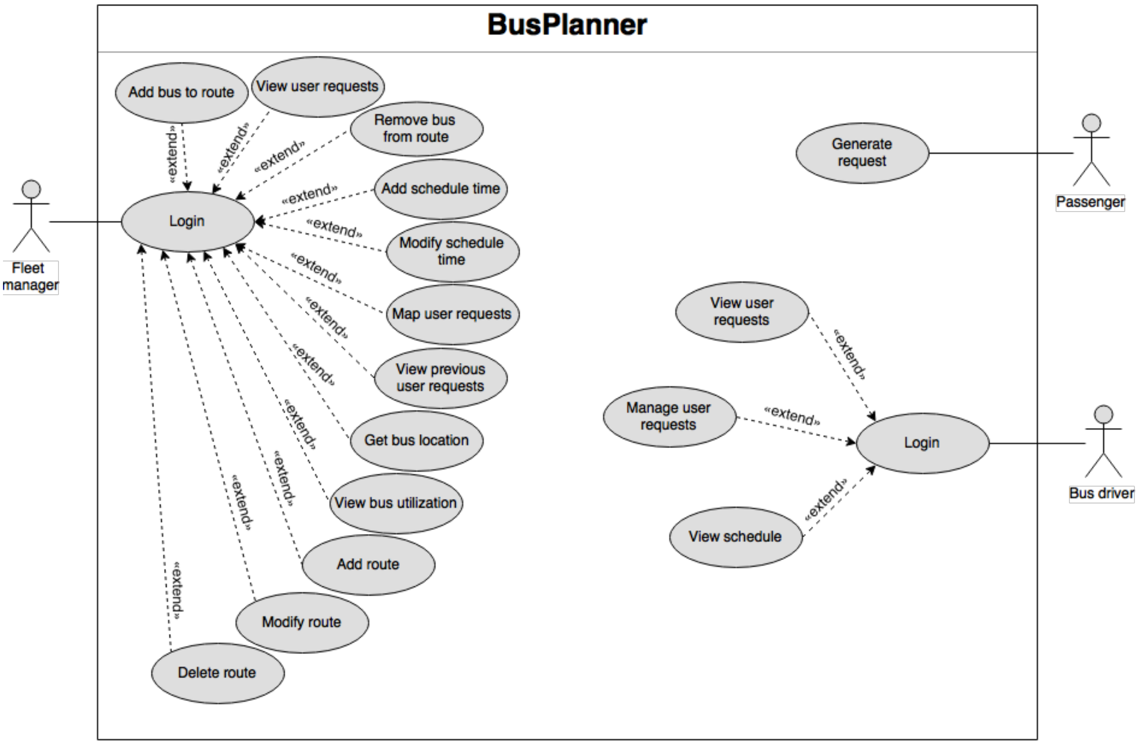


Figure 1: BusPlanner use case

2.4 Use case description

2.4.1 Passenger

Name	Generate request [Sequence diagram]
Actor	Passenger
Entry conditions	Passenger is already known with the available buses on that route and all the necessary information.
Flow of Events	1. Check bus availability. 2. See bus details for location. 3. See bus schedule. 4. Make seat selection. 5. Send location with timestamps.
Exit Conditions	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.

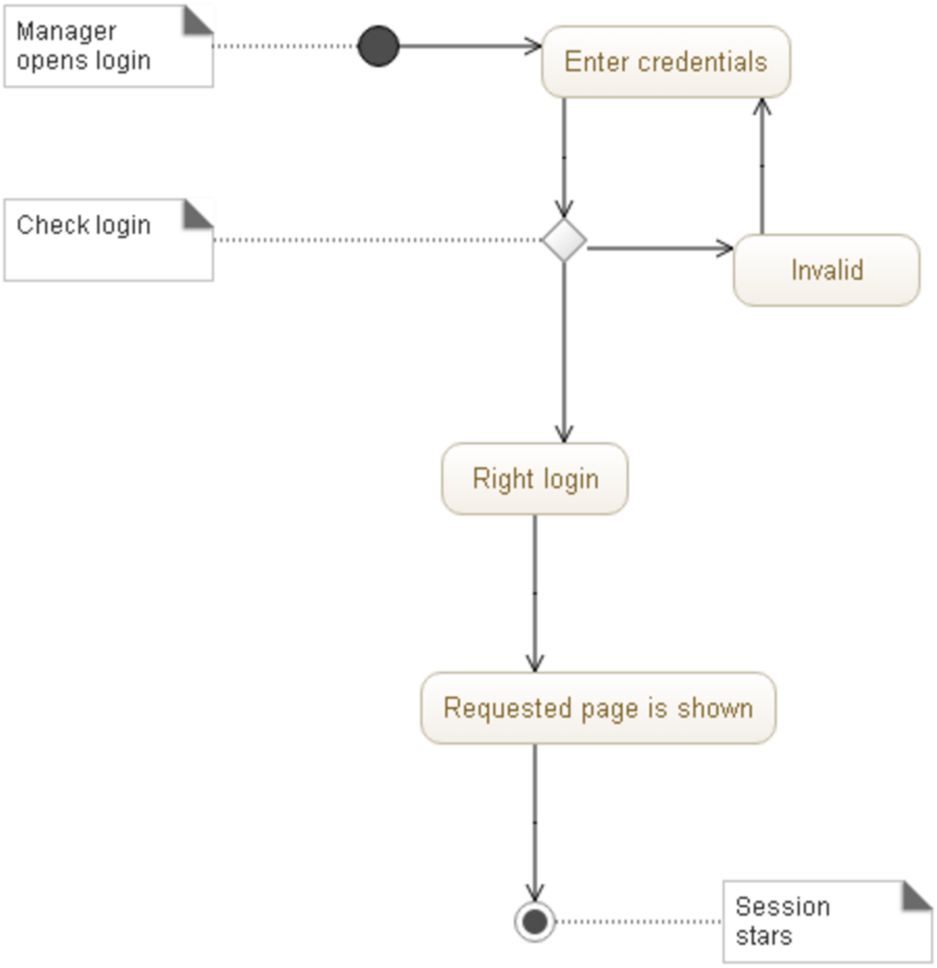
In the real world users can request a bus only if they are at a bus stop (or near one). The bus driver will be able to see the request and realize how many passengers will be at each bus stop. Requests are useful also for fleet managers, whose duty will be to assign a bus (buses are of two different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

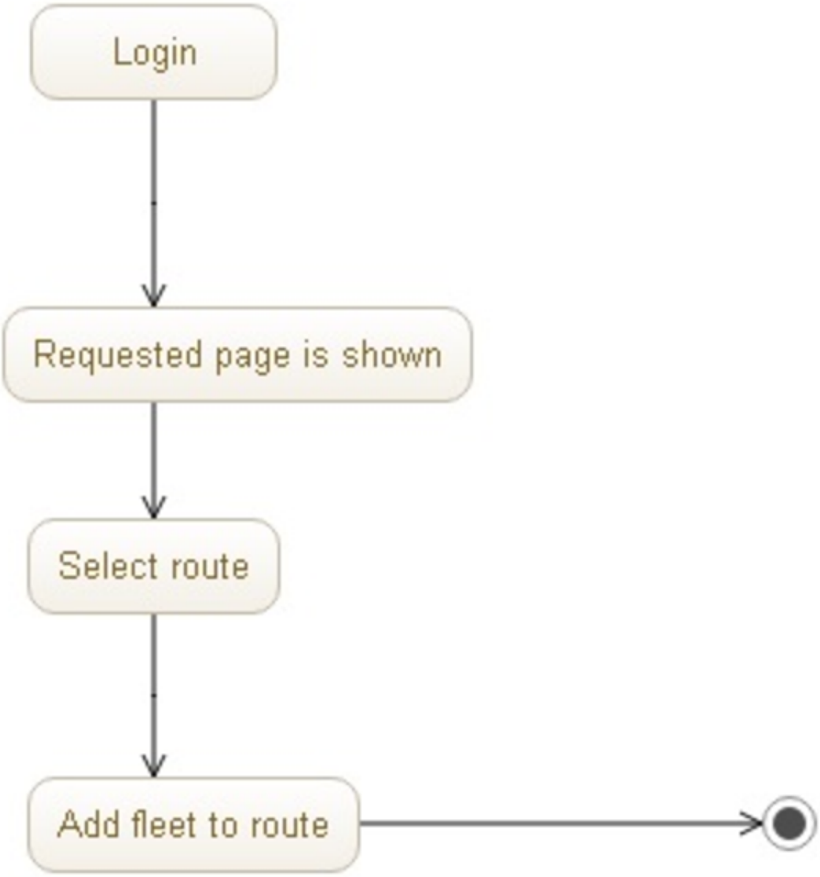
- Login:

Name	Login [Sequence diagram]
Actor	Fleet manager
Entry conditions	The form is filled with credentials.
Flow of Events	1. Web page opened. 2. Enter the credentials. 3. Enter button pressed.
Exit Conditions	Session variables.
Exceptions	Wrong credentials. Step is repeated.



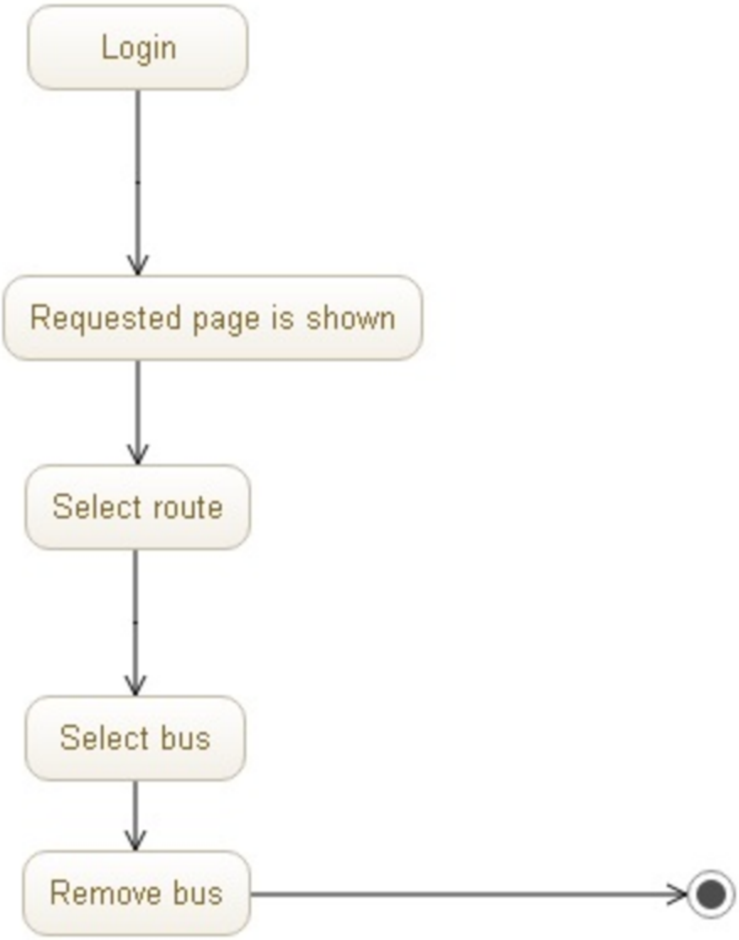
- Add bus to route:

Name	Add bus to route [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page opened. 2. Respective form is filled with bus technical details. 3. Desired route and fleet selected. 4. Submit button pressed.
Exit Conditions	Database confirmation.
Exceptions	Wrong informations are inserted.



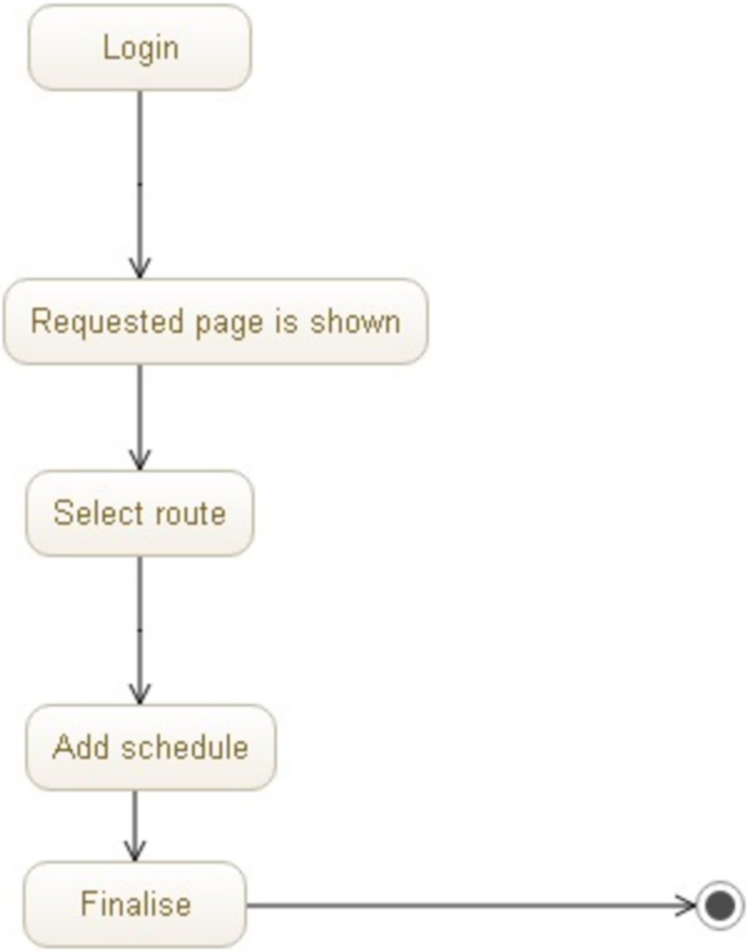
- Remove bus from route:

Name	Remove bus from route [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired route and fleet selected. 3. Information is shown on map. 4. Bus is removed from the route.
Exit Conditions	The schedule is affected.
Exceptions	Bus not found.



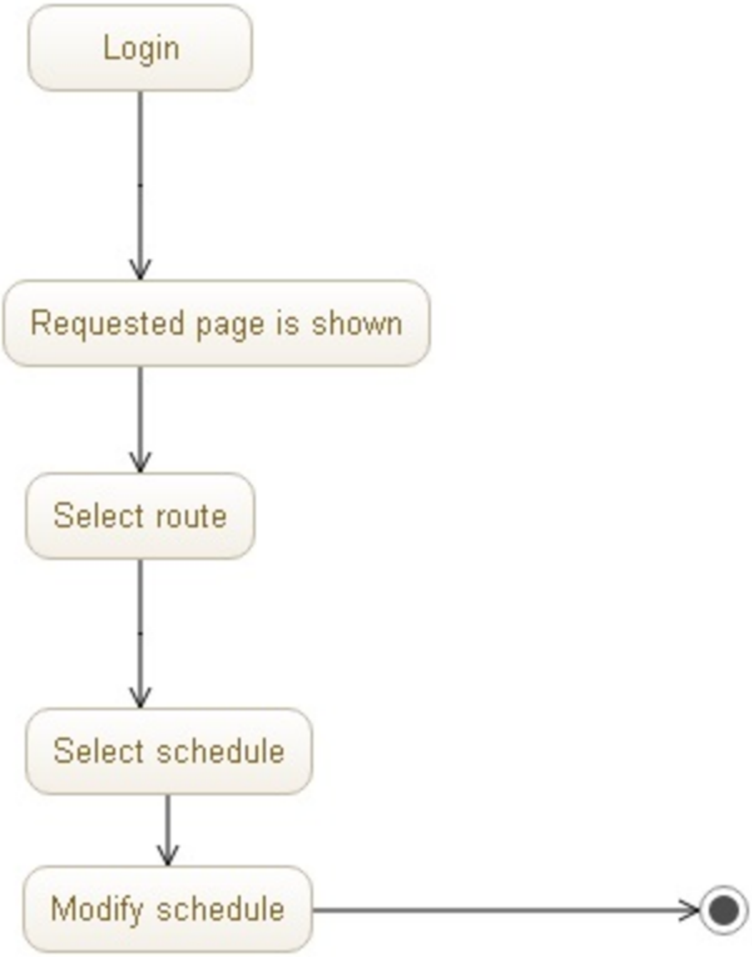
- Add schedule time:

Name	Add schedule time [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired route is selected. 3. Information is shown on the map. 4. Schedule form is filled.
Exit Conditions	The database is updated.
Exceptions	The driver cannot see the schedule.



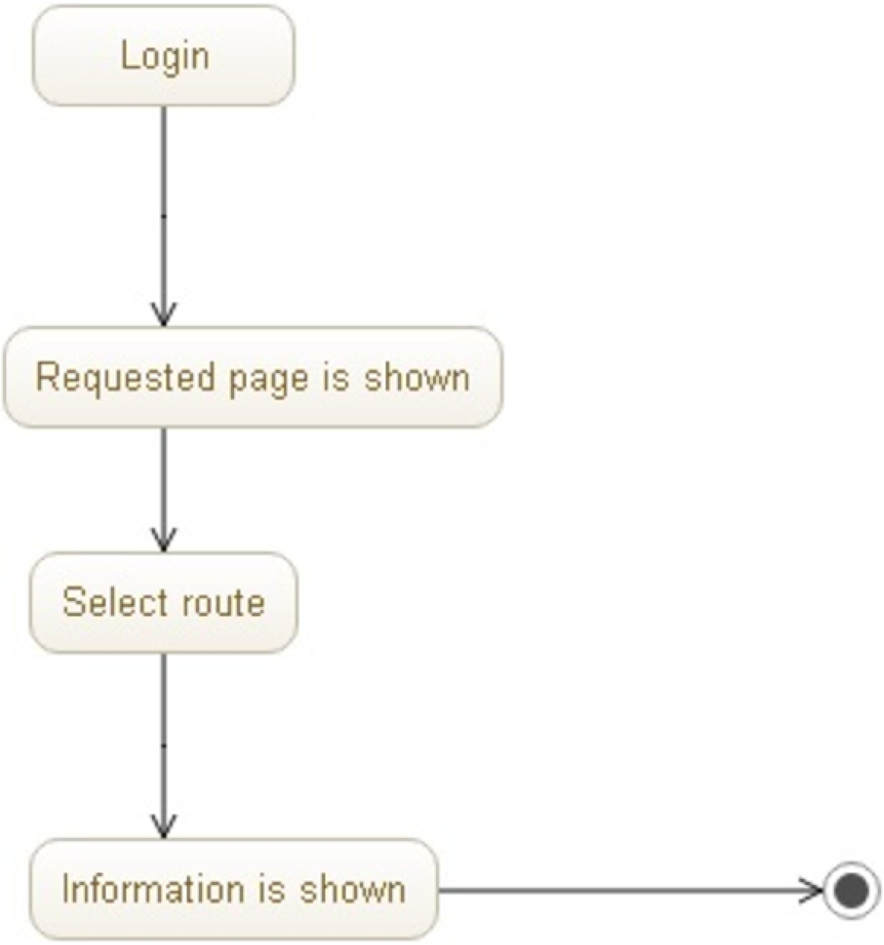
- Modify schedule time:

Name	Modify schedule time [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired schedule is selected. 3. Information is shown. 4. Schedule form is filled.
Exit Conditions	The database is updated.
Exceptions	The driver cannot see the schedule.



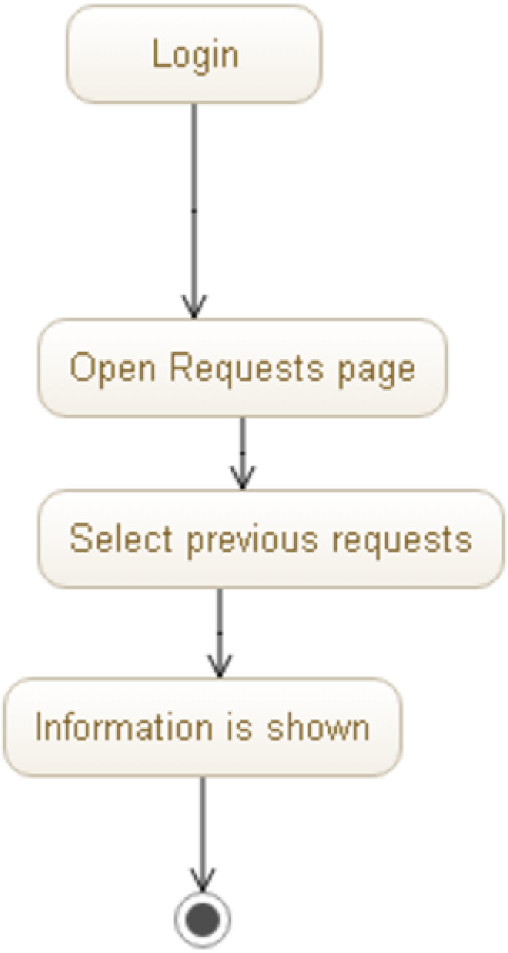
- Mapping user requests:

Name	Mapping user requests [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Read from database. 3. Information is shown.
Exit Conditions	The database is updated.
Exceptions	Data is not available.



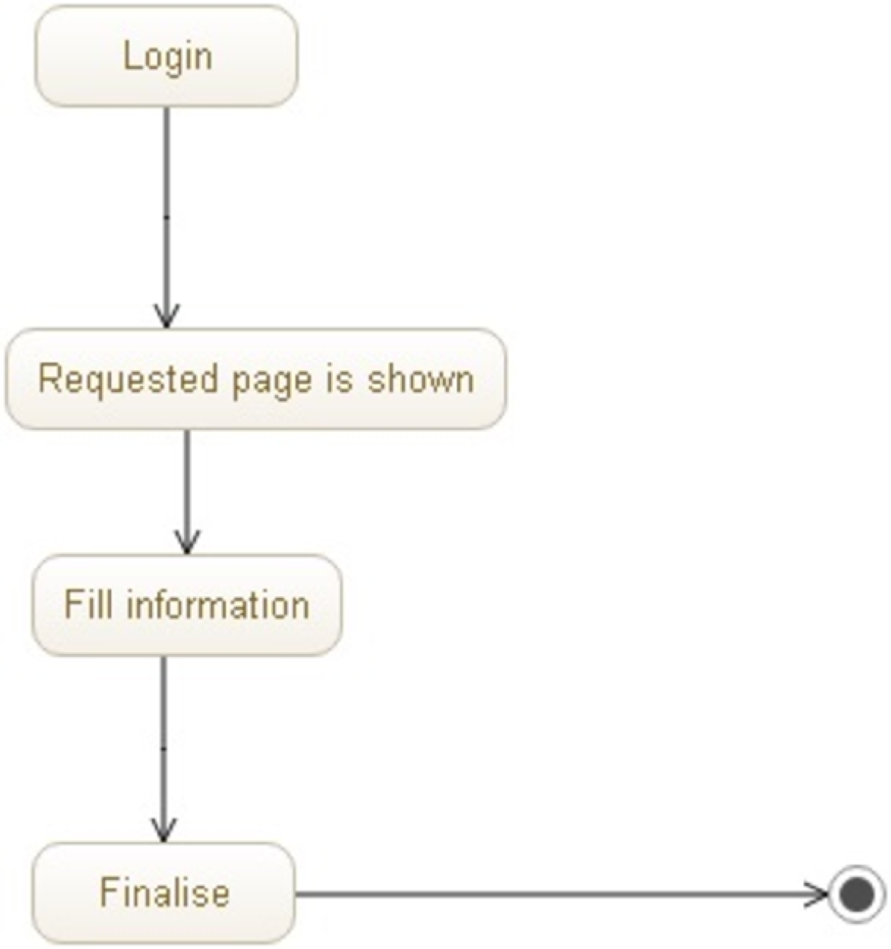
- View previous user requests:

Name	View previous user requests [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Previous user requests are selected. 3. Read data from the database. 4. Information is shown.
Exit Conditions	Getting previous users requests.
Exceptions	Data is not available.



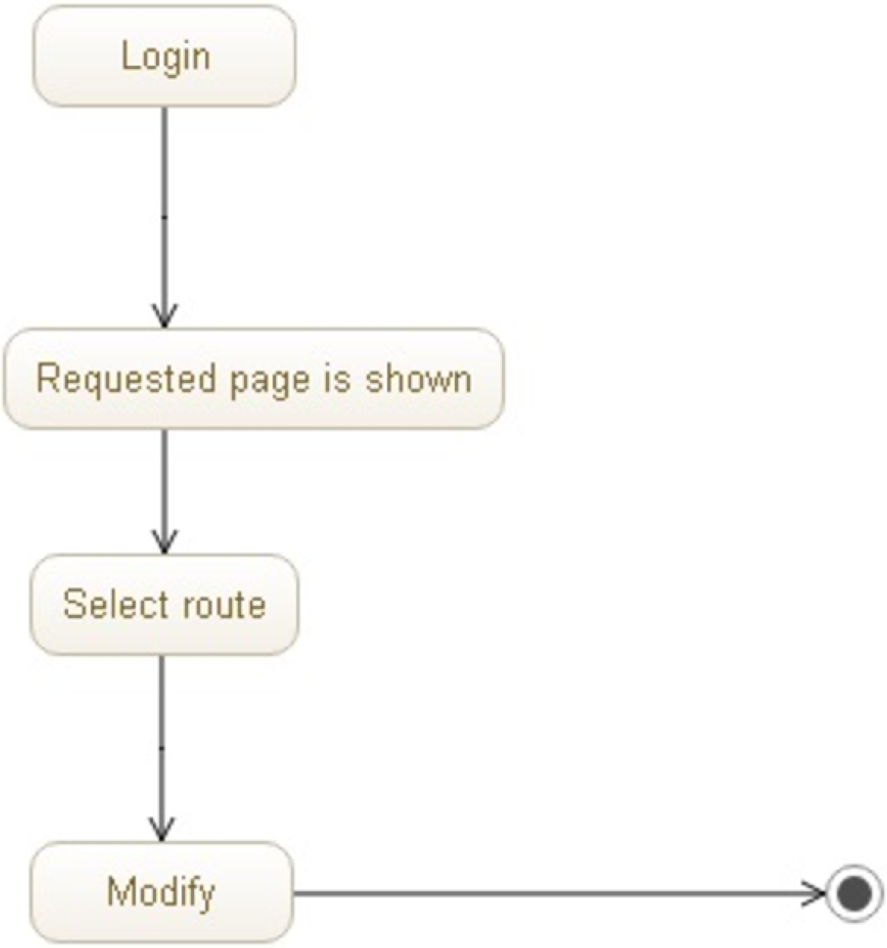
- Add route:

Name	Add route [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired route information is filled. 3. Submit button is pressed.
Exit Conditions	The database is updated.
Exceptions	The database is not updated.



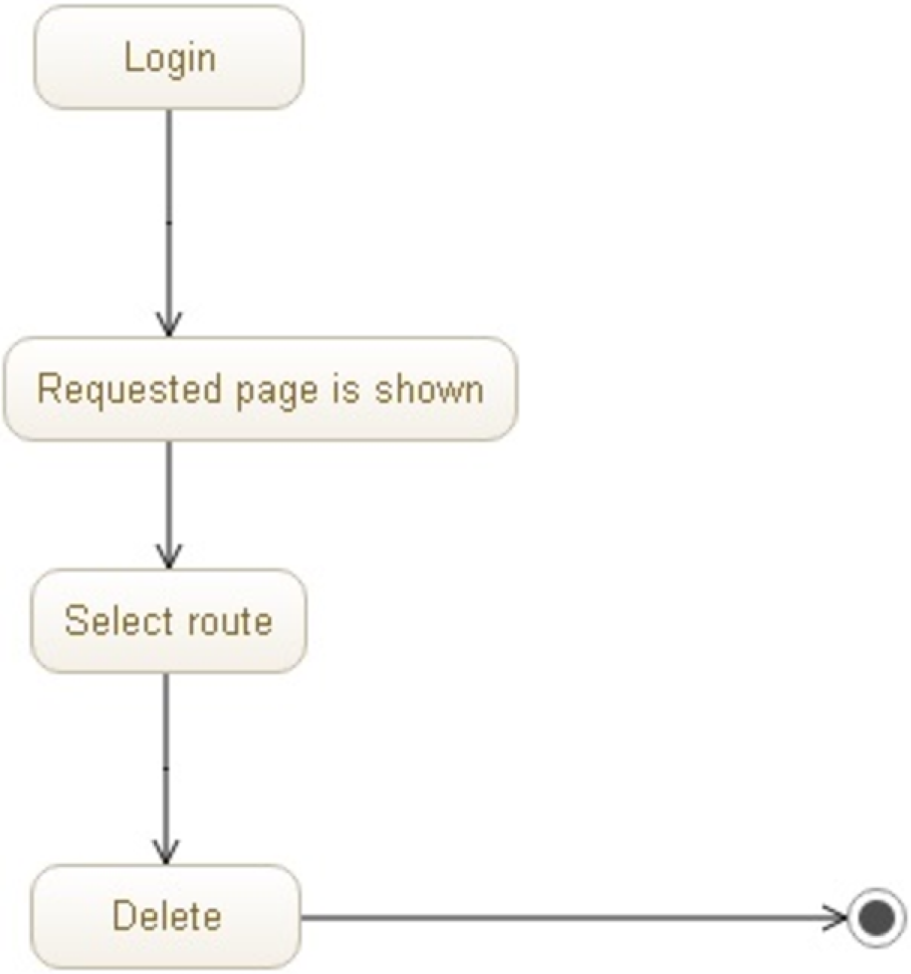
- Modify route:

Name	Modify route [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired route information is modified. 3. Submit button is pressed.
Exit Conditions	The database is updated.
Exceptions	The database is not updated.



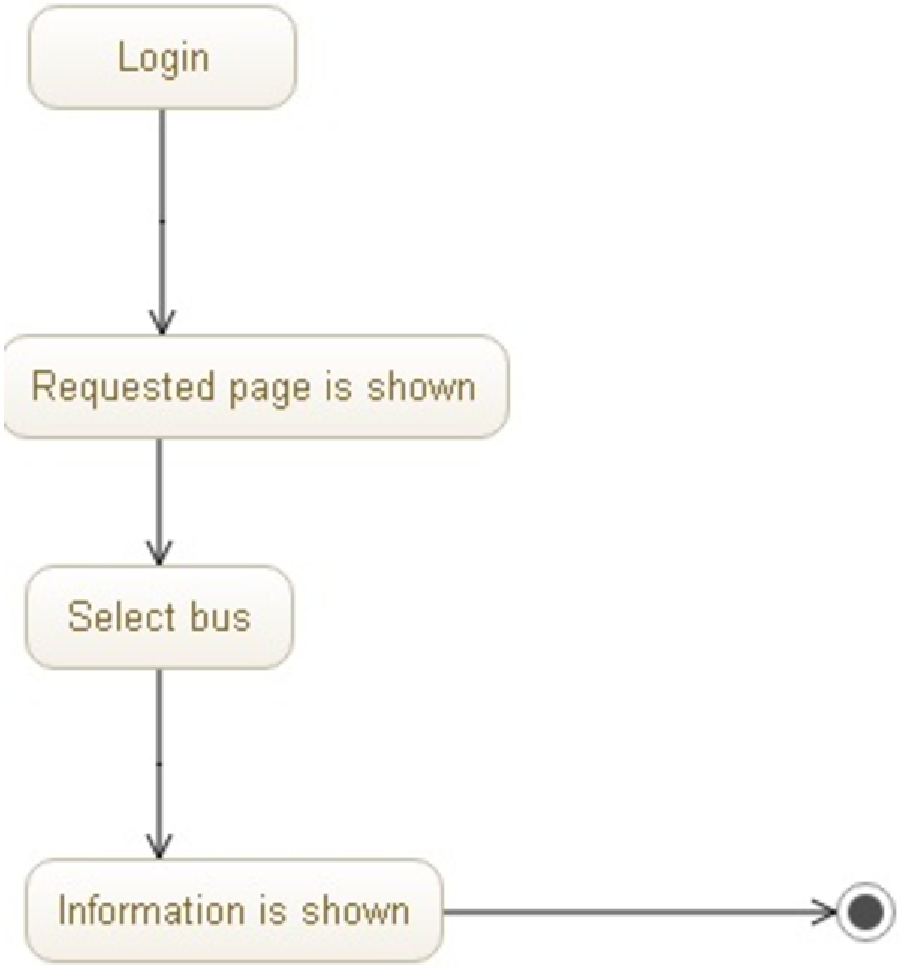
- Delete route:

Name	Delete route [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired route information is deleted. 3. Submit button is pressed.
Exit Conditions	The database is updated.
Exceptions	The database is not updated.



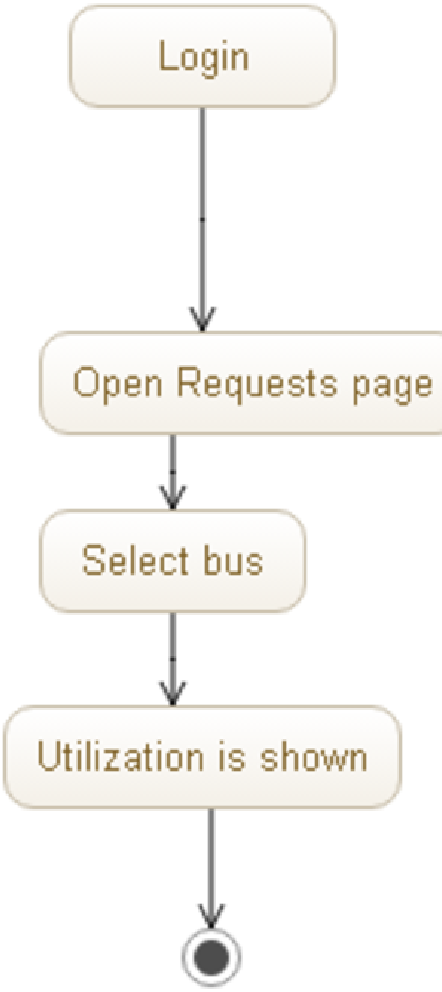
- Get bus location:

Name	Get bus location [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired bus is selected. 3. Information is shown in the map.
Exit Conditions	Getting the bus's current position in map.
Exceptions	Data not available.



- View bus utilization:

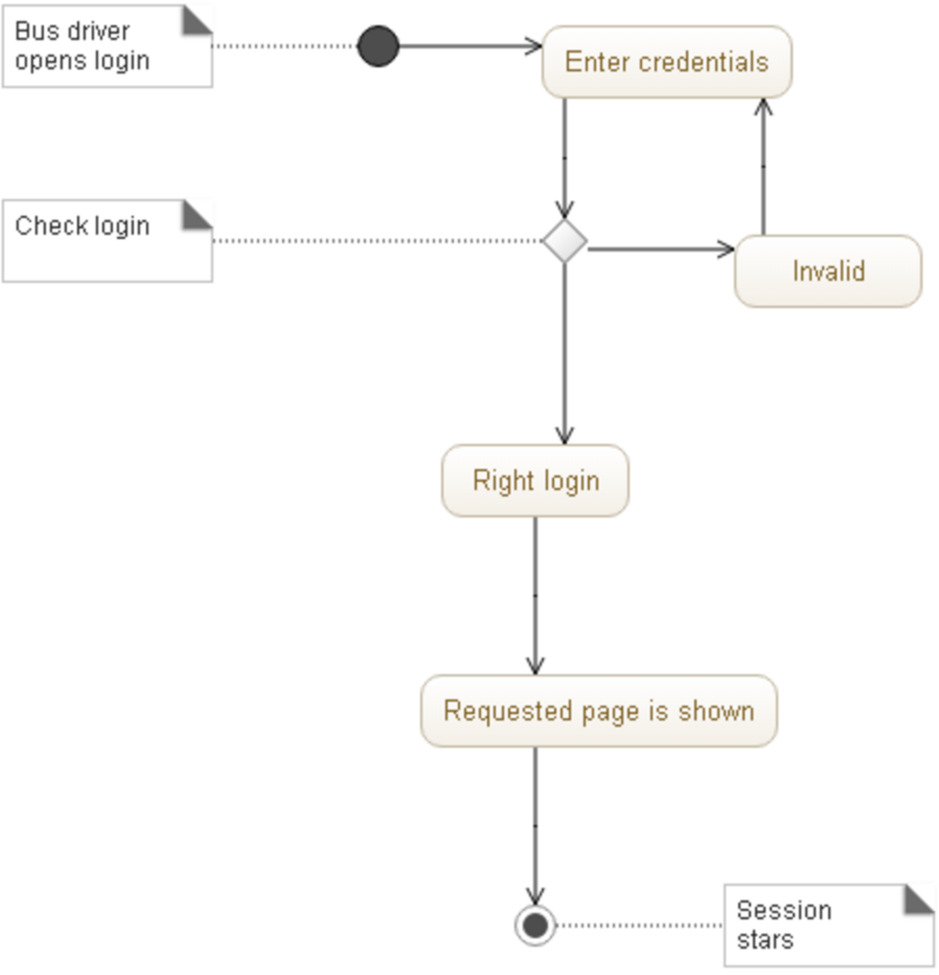
Name	View bus utilization [Sequence diagram]
Actor	Fleet manager
Entry conditions	Fleet manager is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired bus is selected. 3. Information is shown.
Exit Conditions	Getting the utilization of buses.
Exceptions	Data not available.



2.4.3 Bus driver

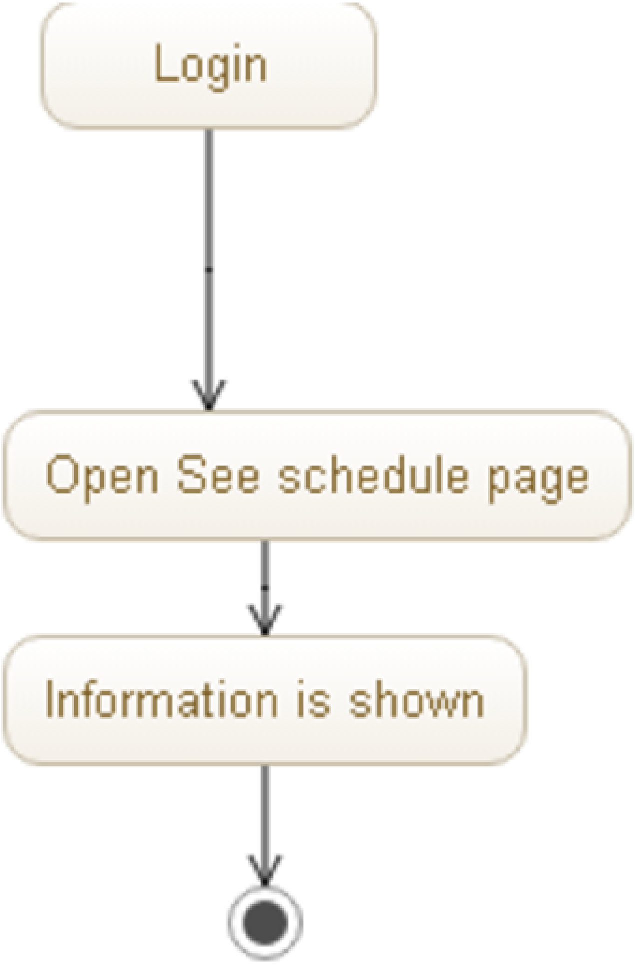
- Login:

Name	Login [Sequence diagram]
Actor	Bus driver
Entry conditions	Form is filled with credentials.
Flow of Events	1. Web page is opened. 2. Form is filled. 3. Enter button pressed.
Exit Conditions	Session variables.
Exceptions	Wrong credentials. Step is repeated.



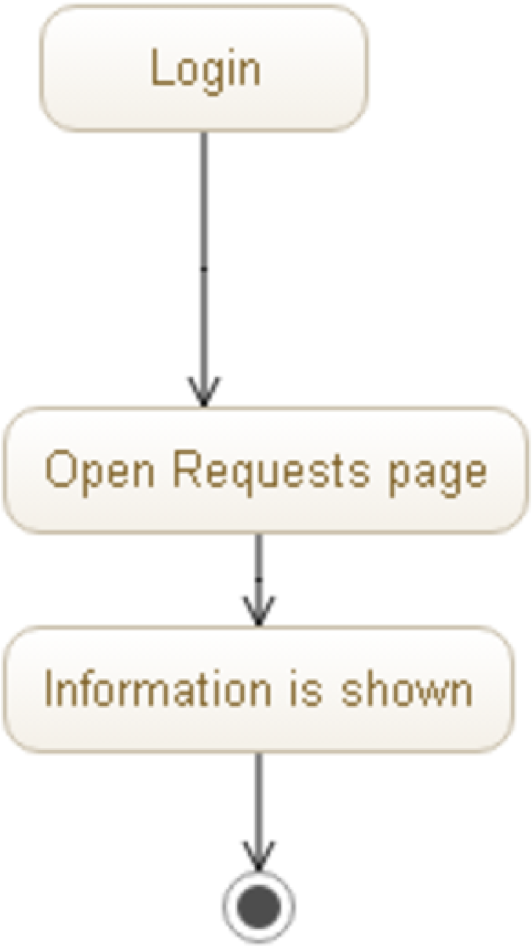
- View schedule:

Name	View schedule [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 in the screen.
Exit Conditions	Getting the schedule.
Exceptions	Data not available.



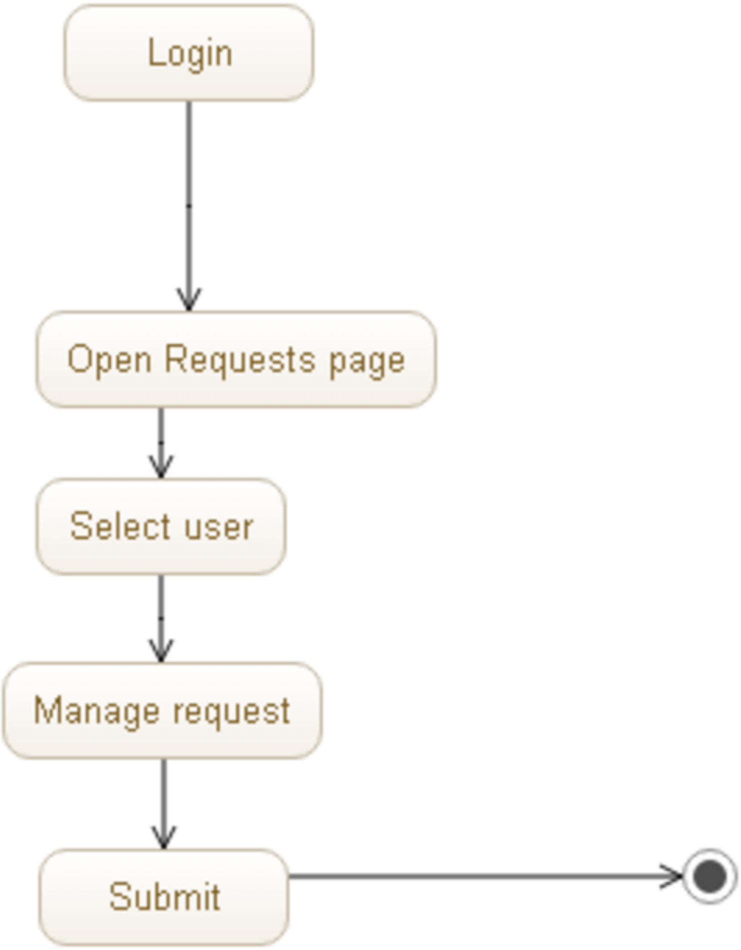
- View user requests:

Name	View 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 the database. 3. Information is shown on the screen.
Exit Conditions	Getting the user's requests.
Exceptions	Data not available.



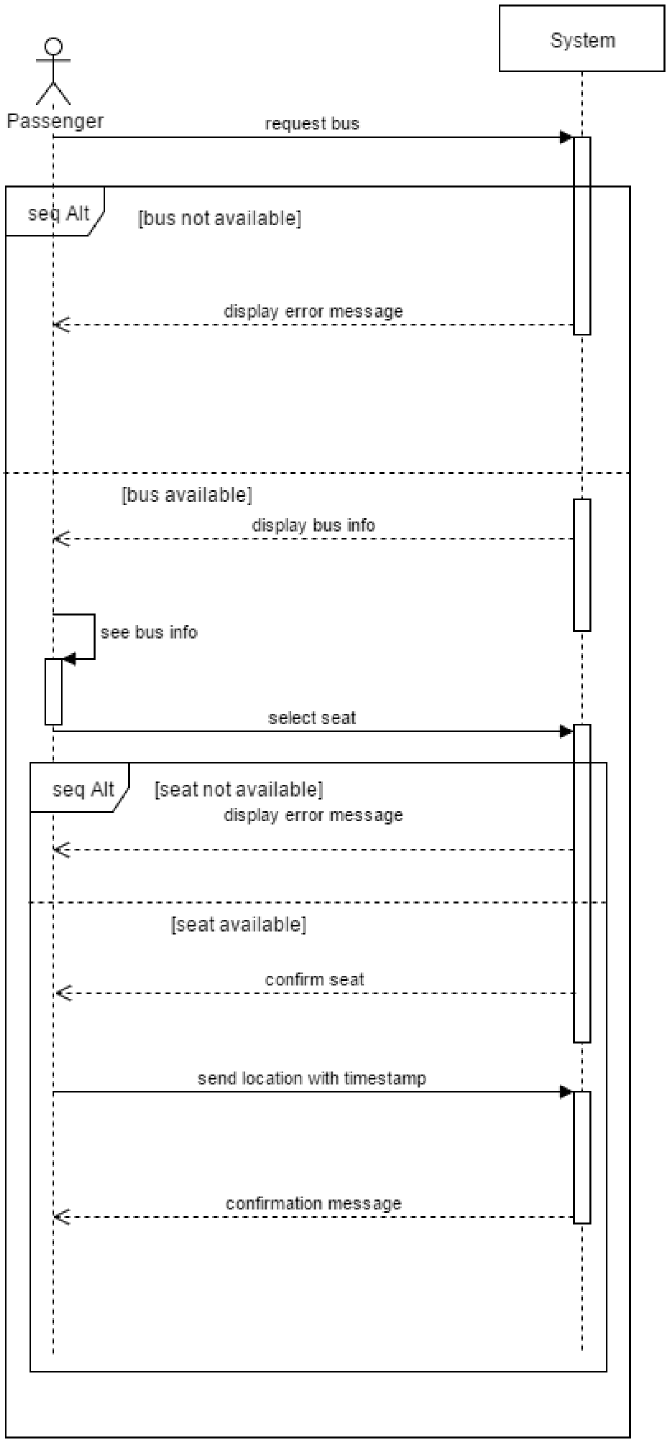
- Manage user requests:

Name	Manage user requests [Sequence diagram]
Actor	Bus driver
Entry conditions	Bus driver is logged in.
Flow of Events	1. Respective web page is opened. 2. Desired user selected. 3. Desired request is managed. 4. Submit button is pressed.
Exit Conditions	Database is updated.
Exceptions	Database is not updated.

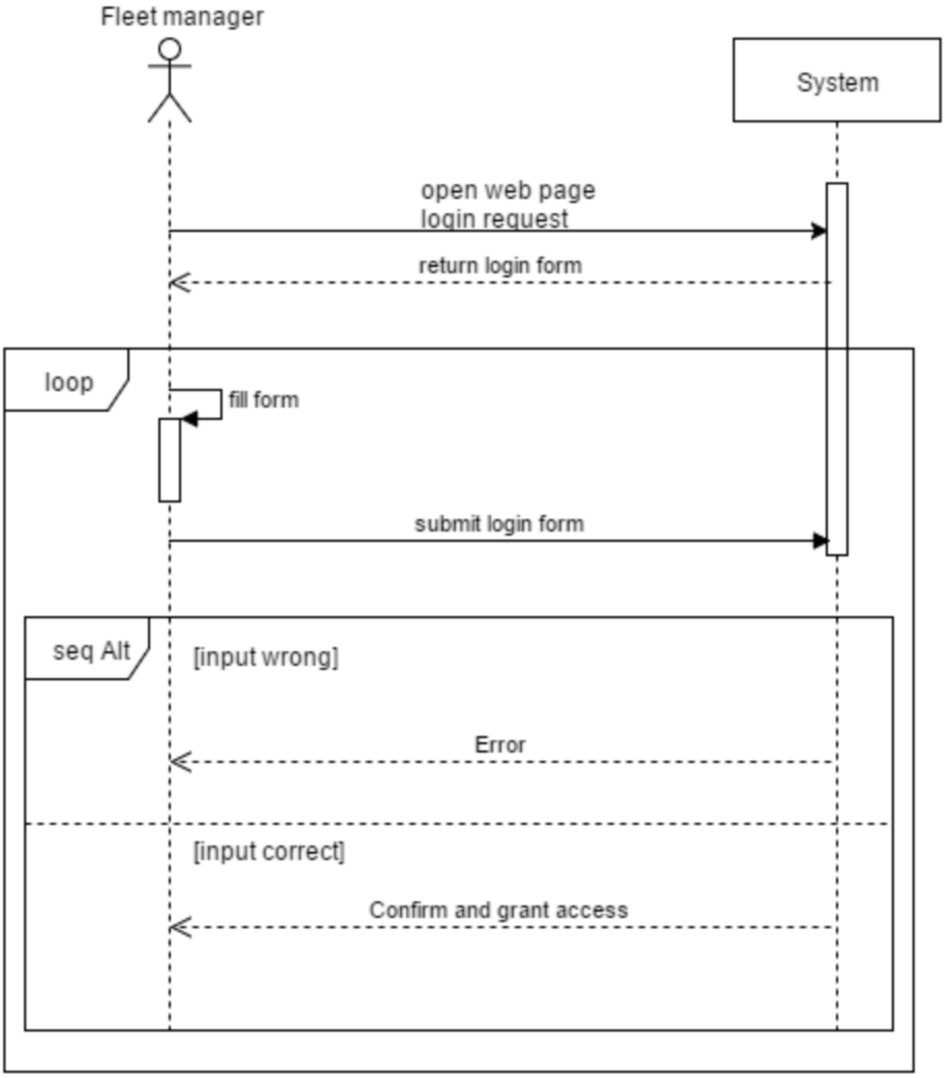


2.4.4 Sequence diagrams

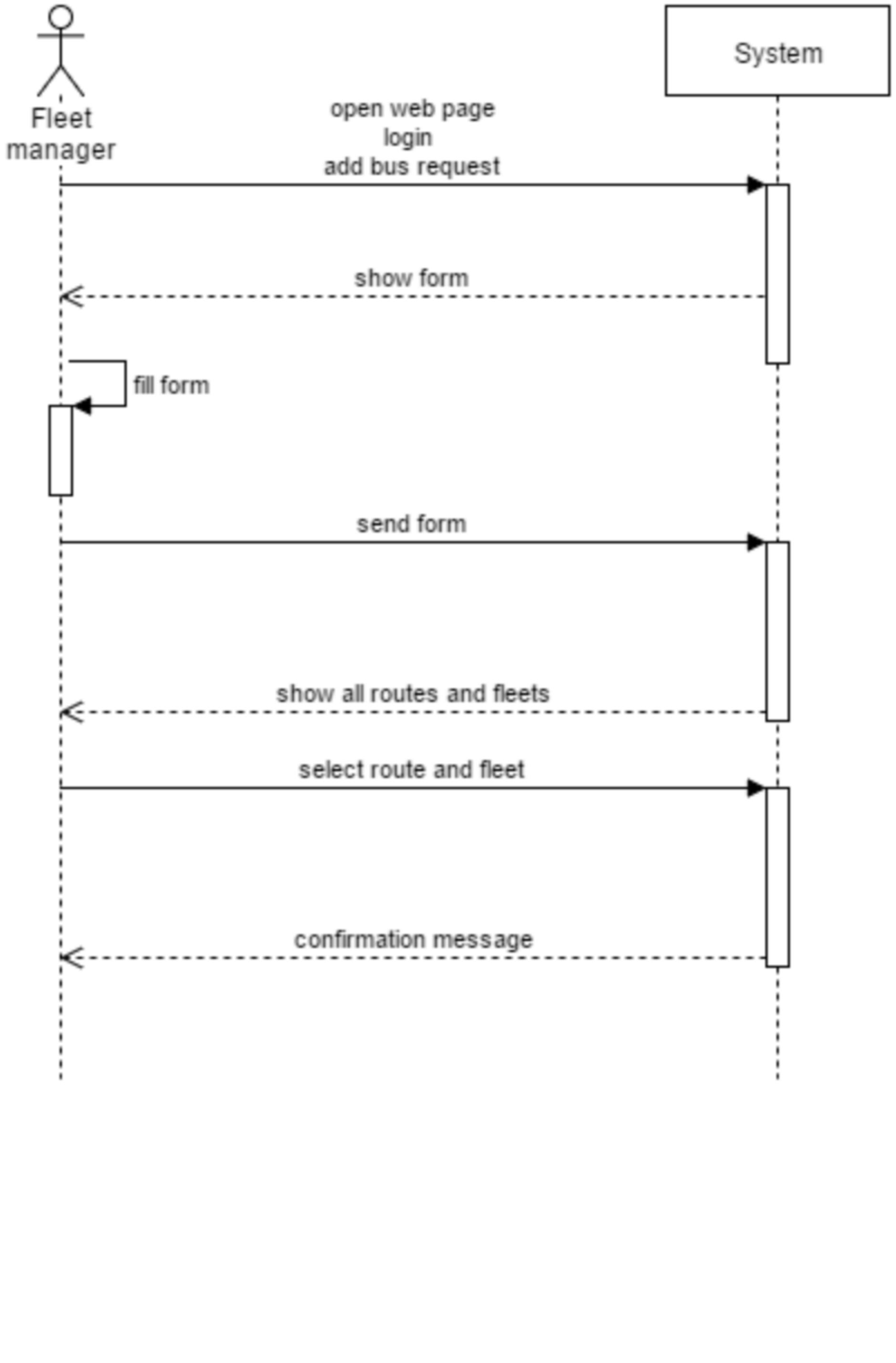
- Generate request:



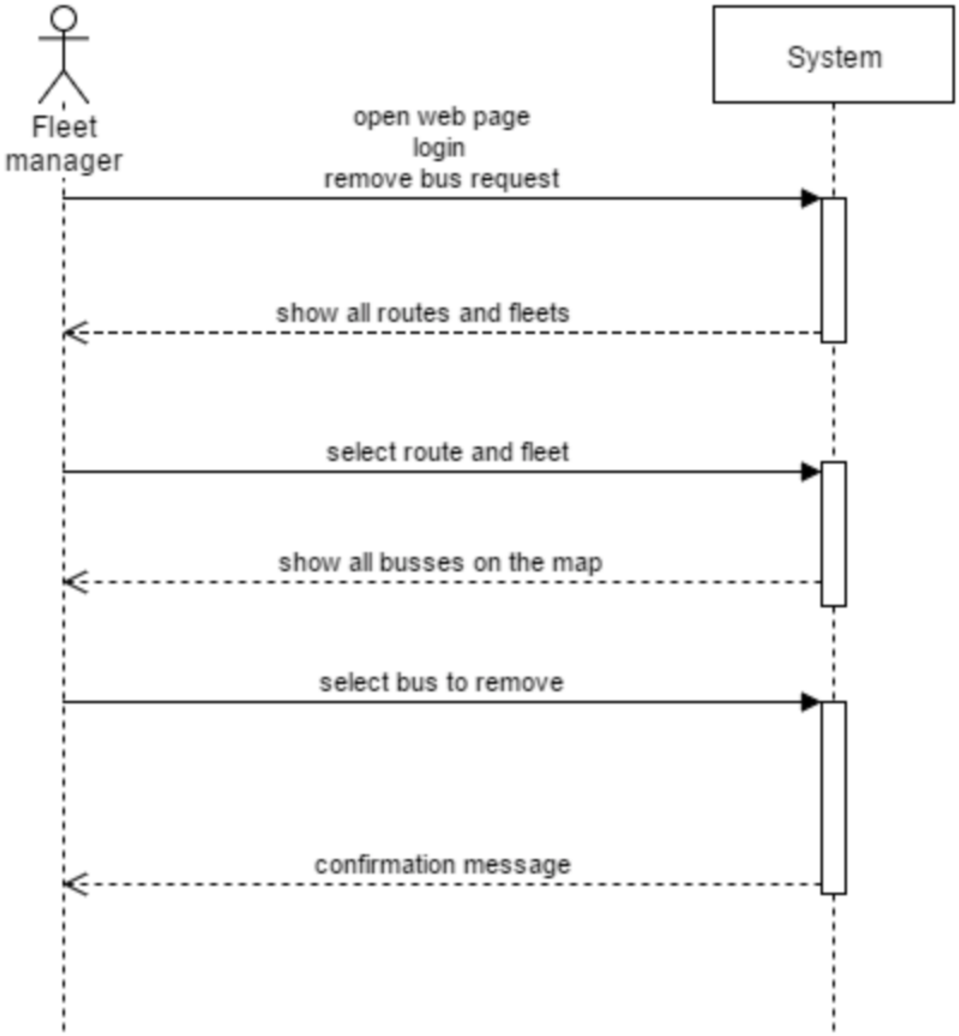
- Fleet manager login:



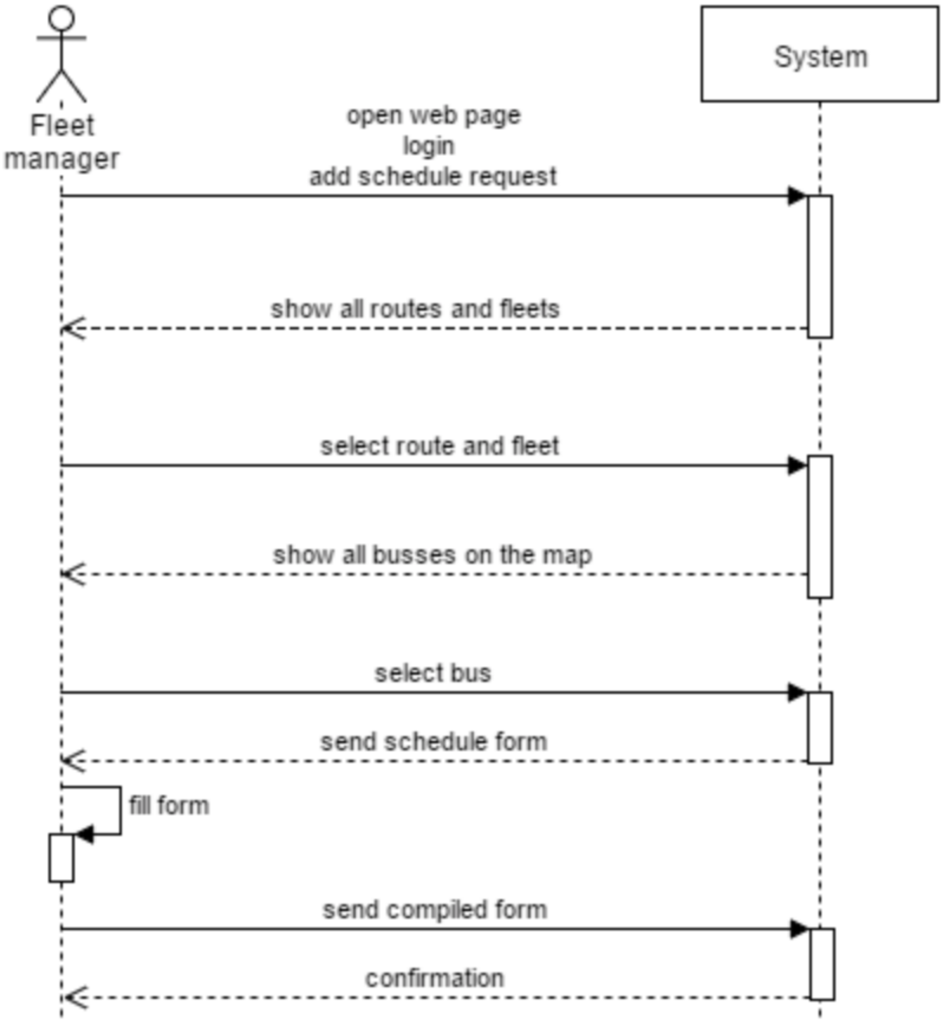
- Add bus to route:



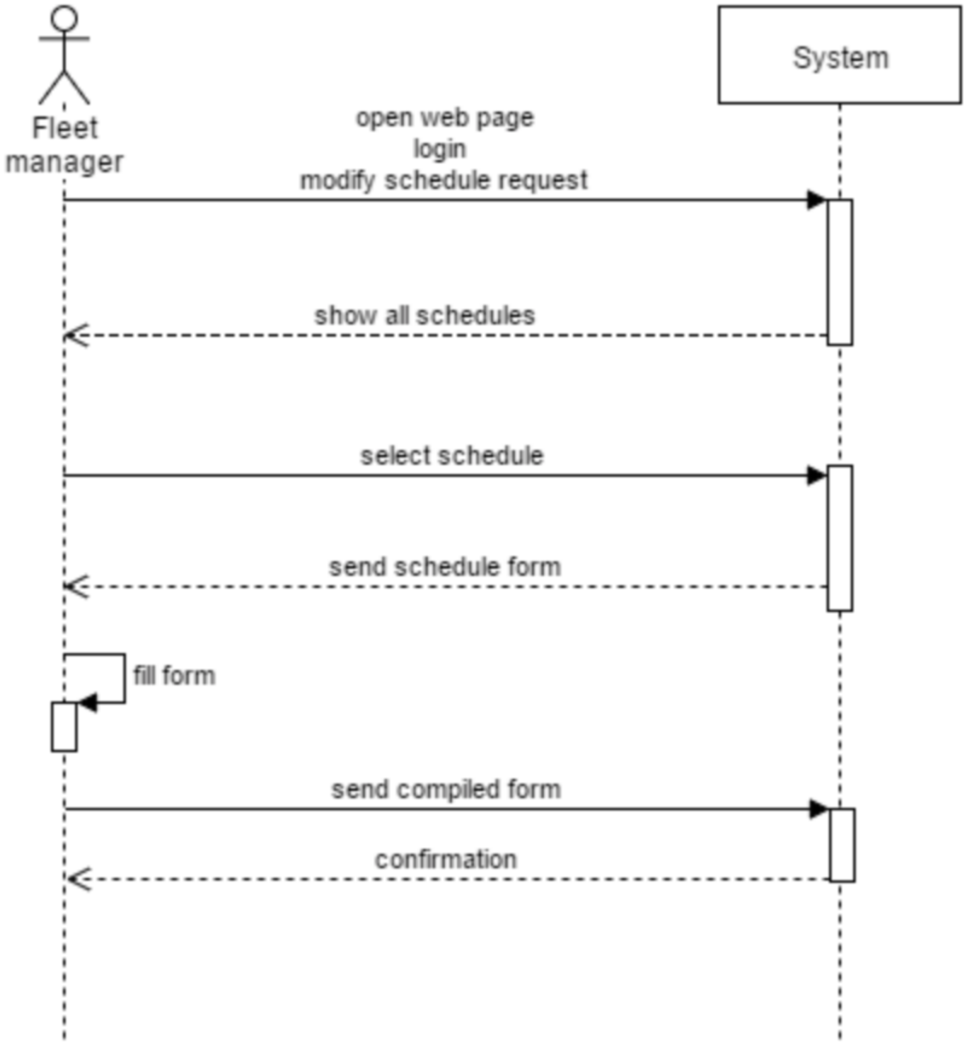
- Remove bus from route:



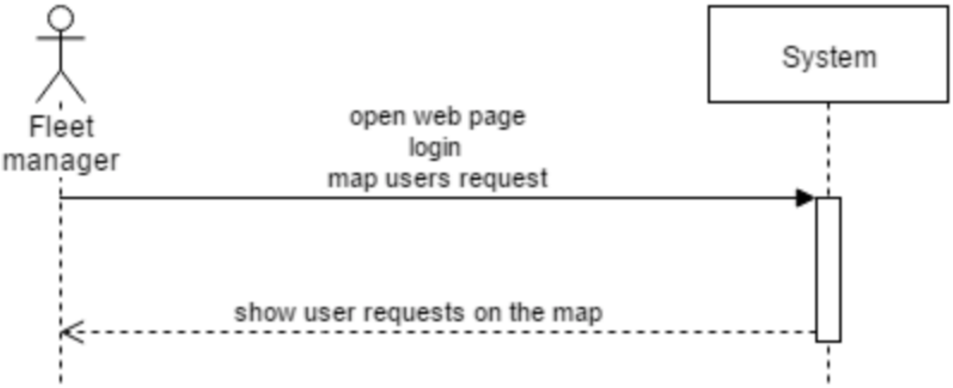
- Add schedule time:



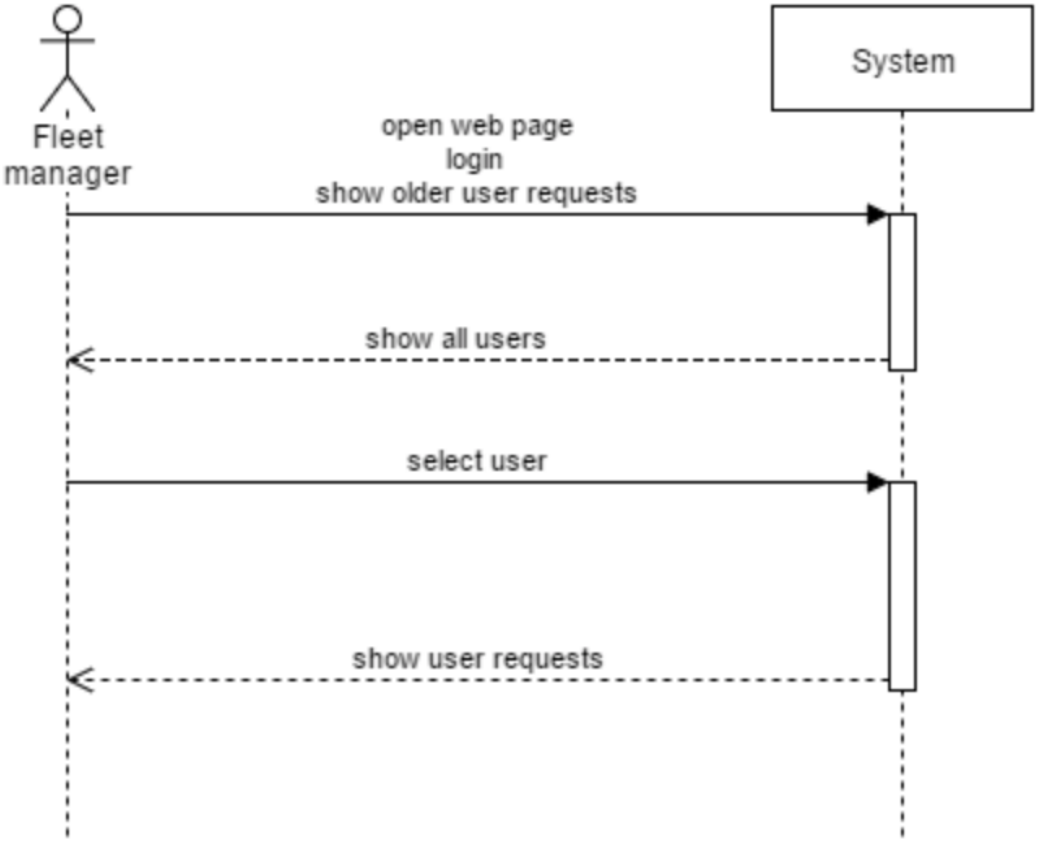
- Modify schedule time:



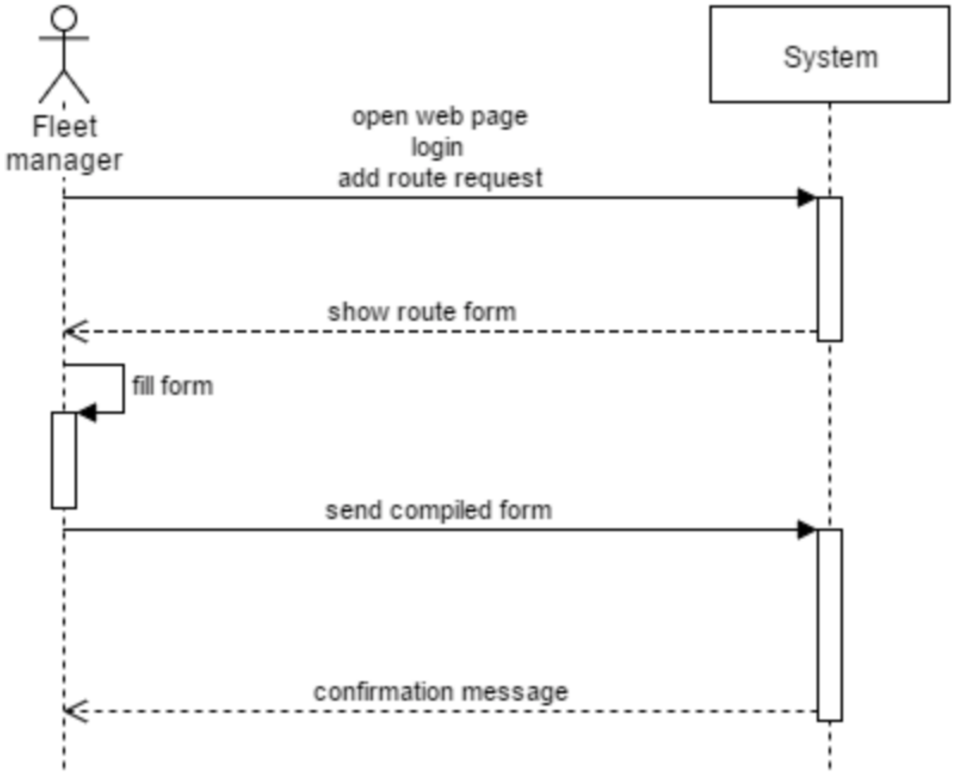
- Mapping user requests:



- View previous user requests:

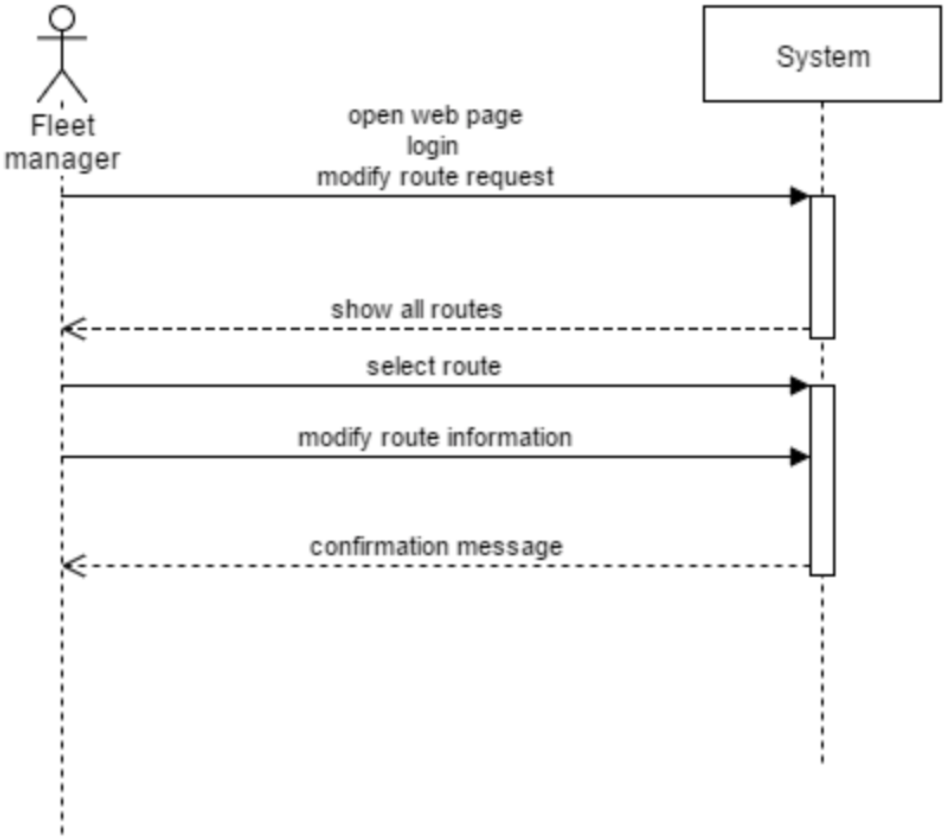


- Add route:

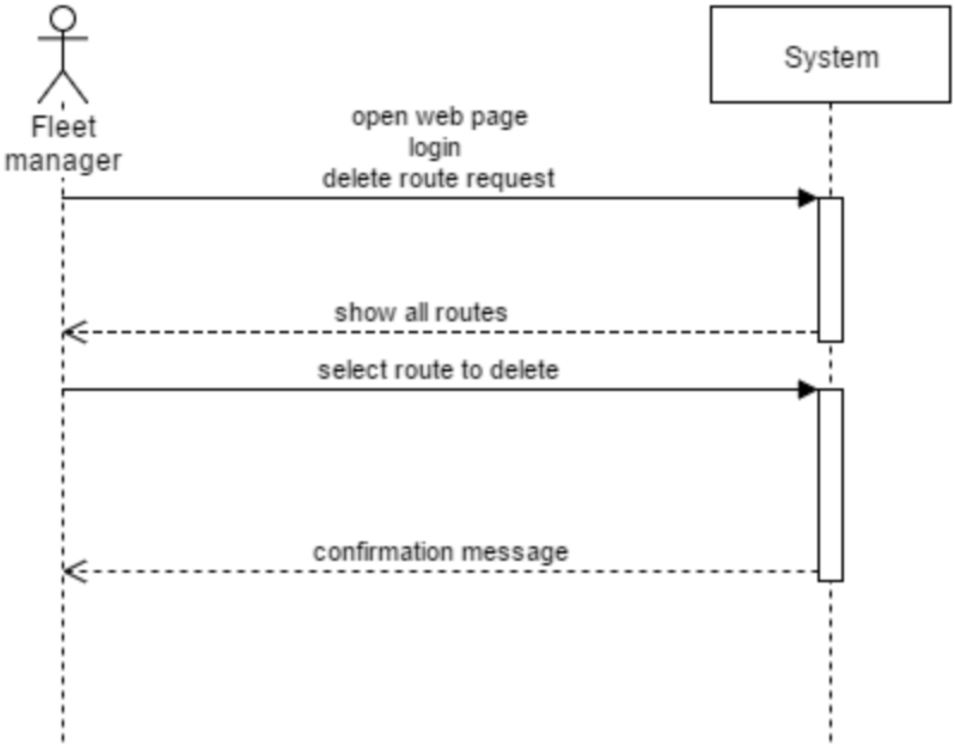


Fleet manager
System
open web page
login
add route request
show route form
fill form
send compiled form
confirmation message

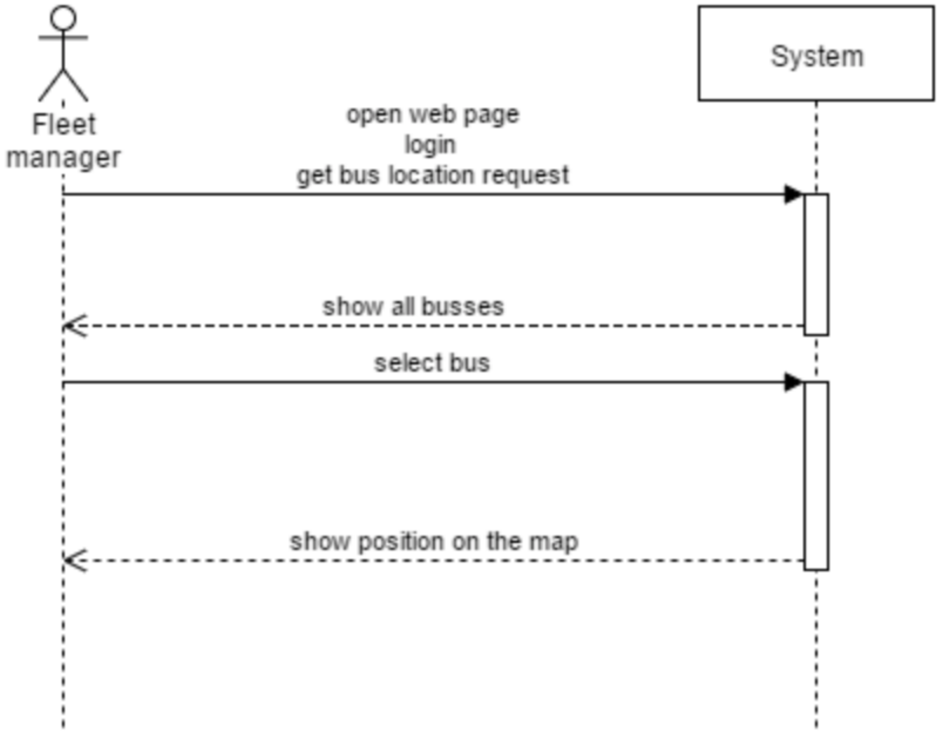
- Modify route:



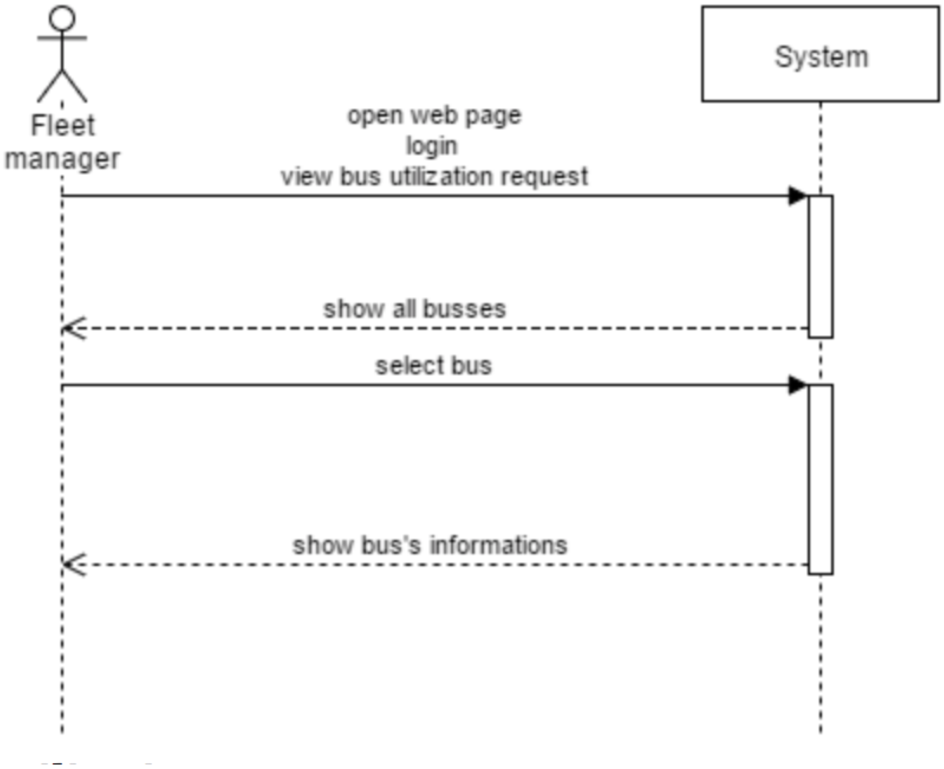
- Delete route:



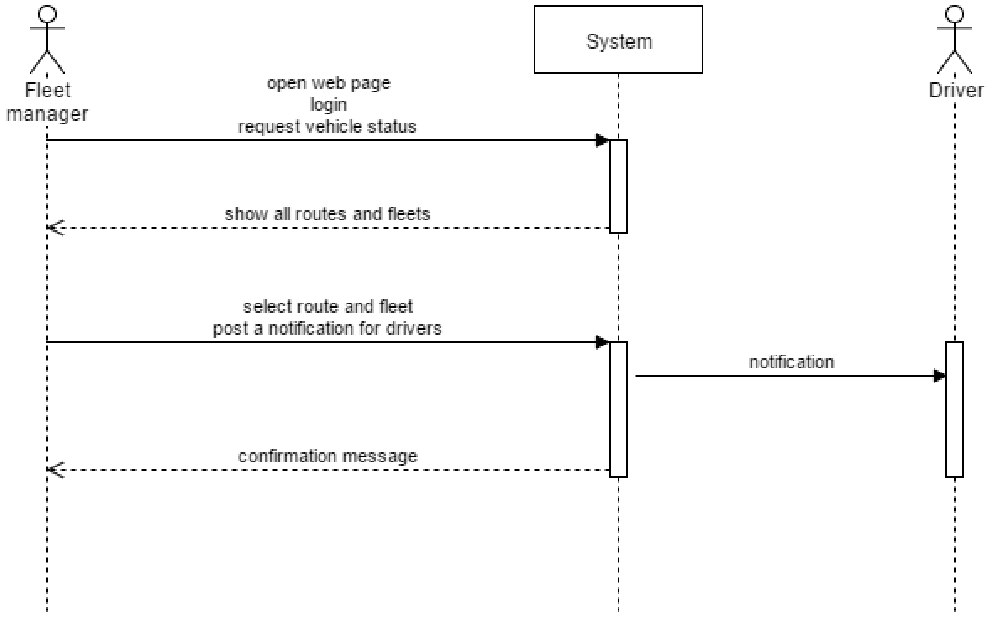
- Get bus location:



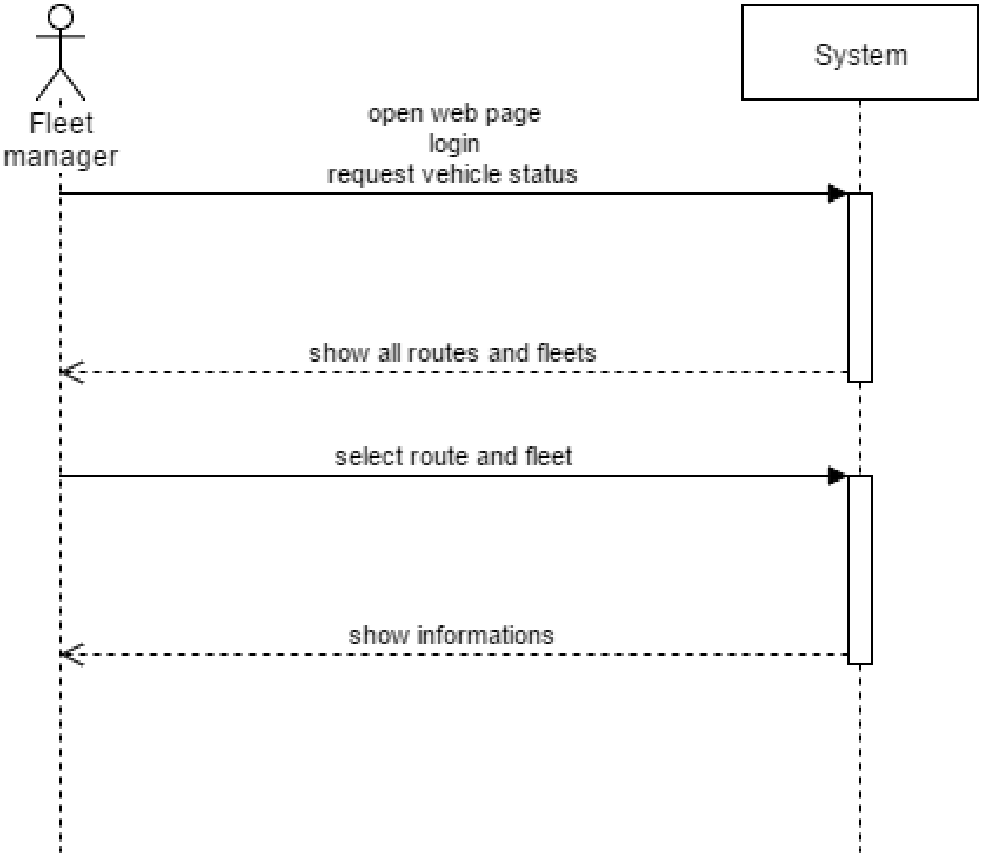
- View bus utilization:



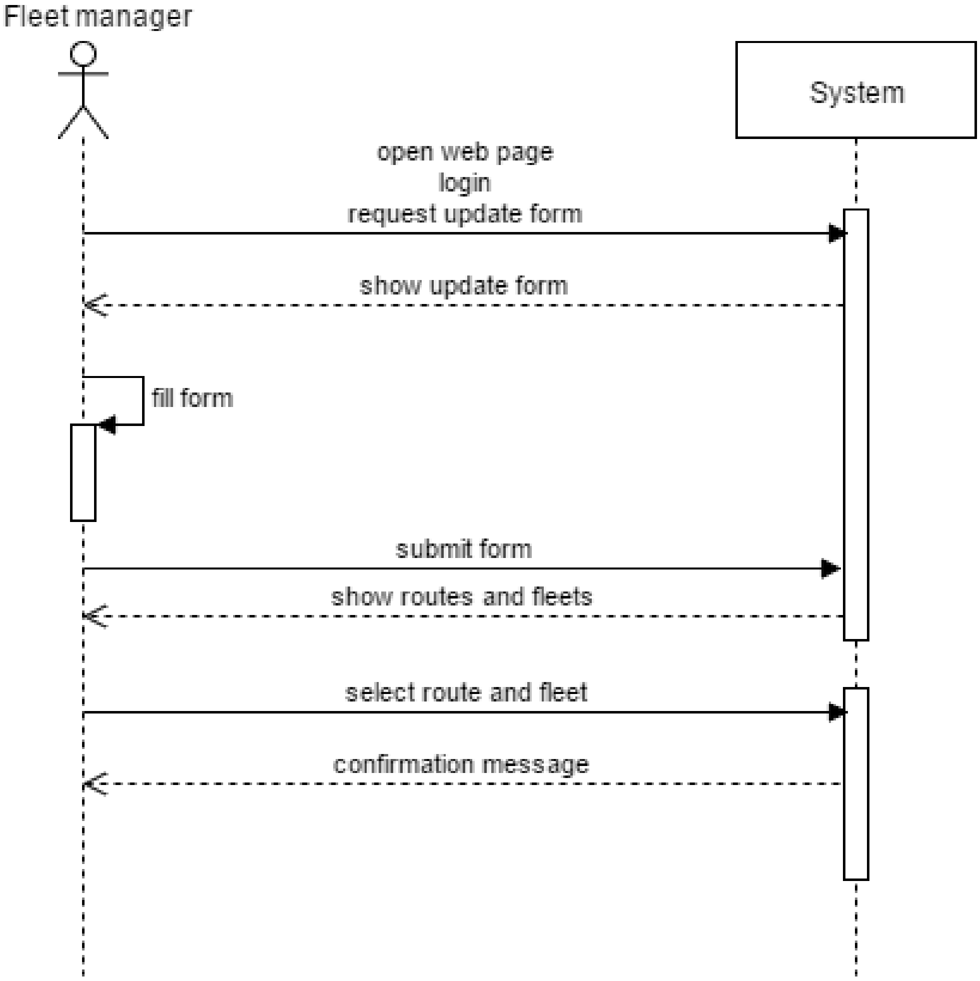
- Add notification for drivers:



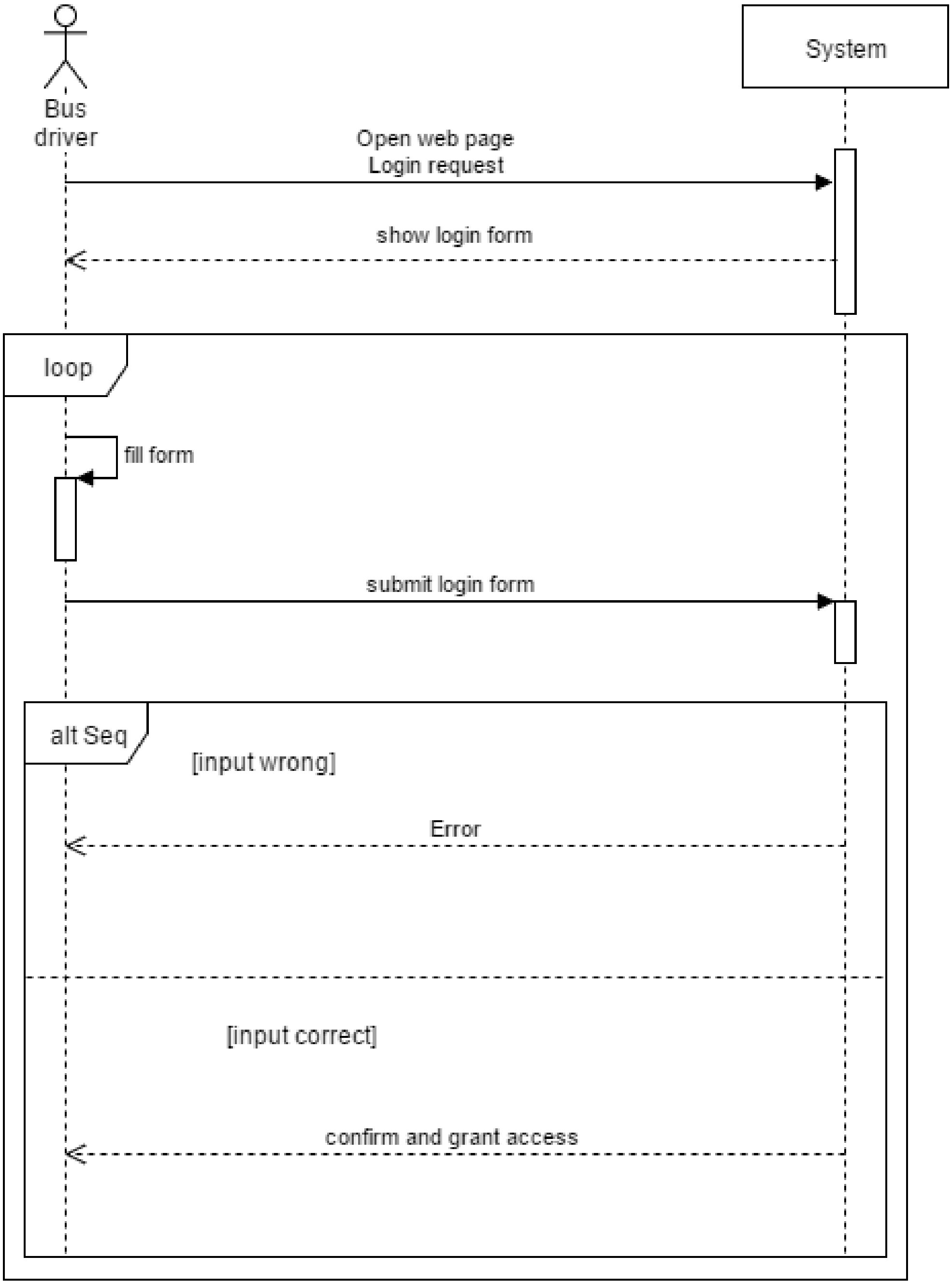
- Control vehicle status:



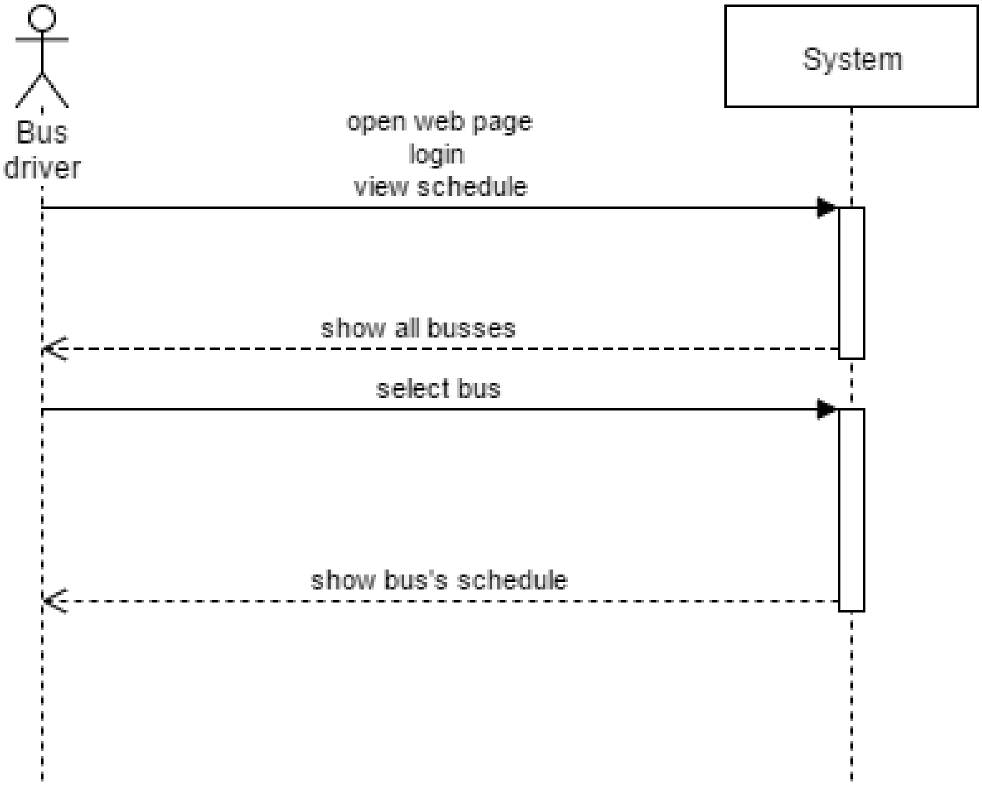
- Update fleet information:



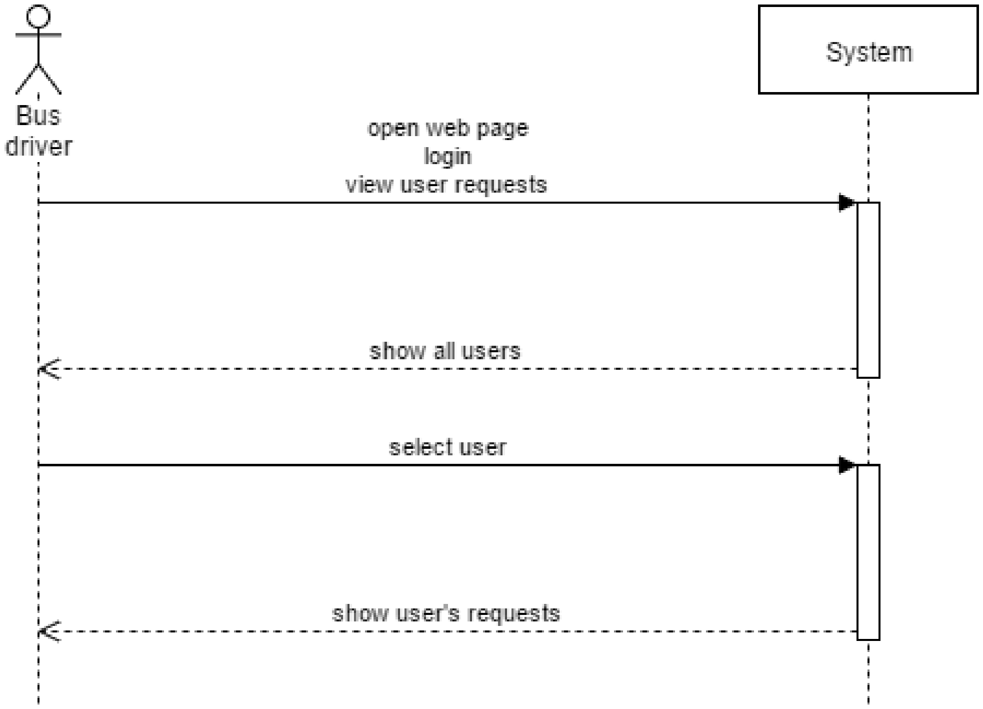
- Bus driver login:



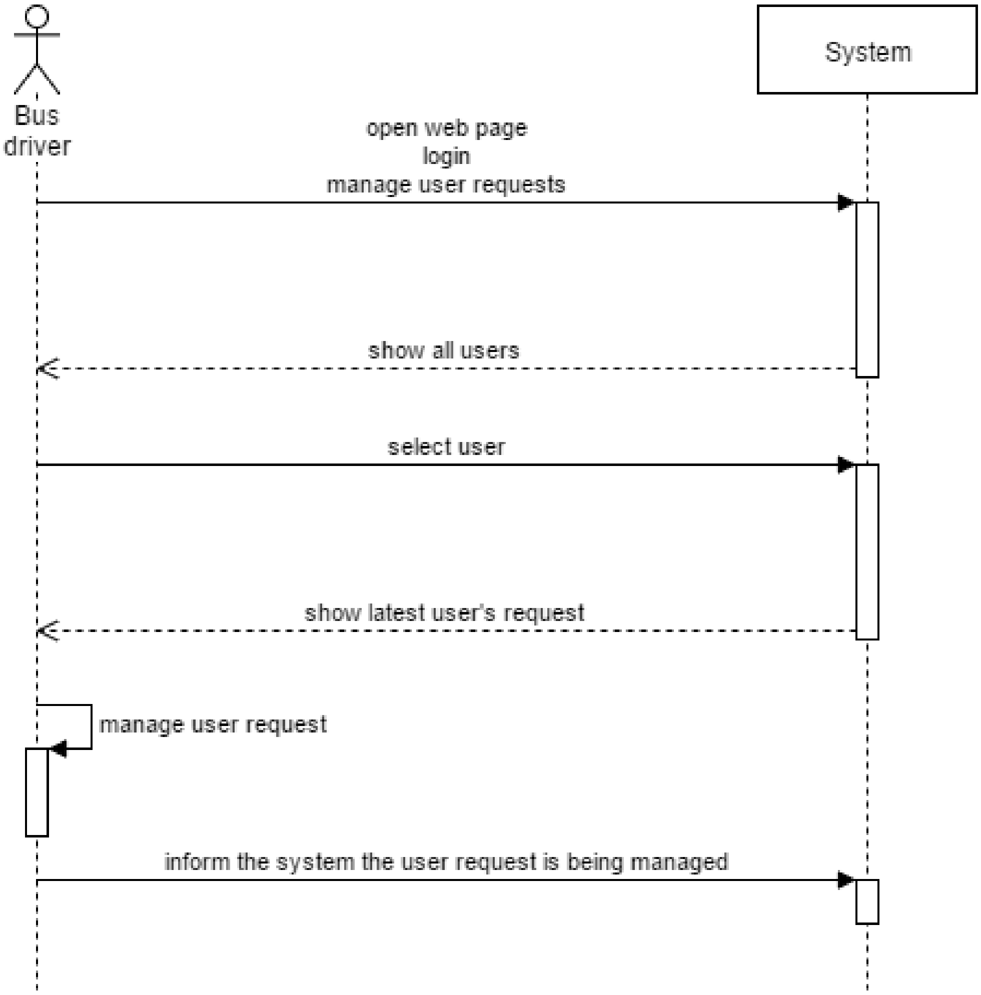
- View schedule:



- View user requests:



- Manage user requests:



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.