**The communication protocol**

**1.The messages that will communicate Mulligan app with Mulligan server:**

Use case 1: Starting a Parking Event

1. **startParking** vehicleID, parkingSpaceNumber

The line begin with startParking (case sensitive) followed by one space then followed by vehicleID followed by one space then followed by parkingSpaceNumber, finally the line ends with new line character. (client to server)

1. **handleStartParkingEvent** vehicleID, parkingSpaceNumber
   1. The line begin with parkingStarted (case sensitive) followed by one space then followed by vehicleID followed by one space then followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
   2. The line begin with invalidVehicleID (case sensitive) followed by one space then followed by vehicleID followed by one space then followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
   3. The line begin with invalidParkingSpaceNumber (case sensitive) followed by one space then followed by vehicleID followed by one space then followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
   4. The line begin with error (case sensitive). (server to client)

Use case 2: Stopping a Parking Event

1. **stopParking** vehicleID

The line begin with stopParking (case sensitive) followed by one space then followed by vehicleID, finally the line ends with new line character. (client to server)

1. **handleStopParkingEvent** vehicleID
   1. The line begin with parkingStopped (case sensitive) followed by one space then followed by vehicleID, finally the line ends with new line character. (server to client)
   2. The line begin with thereIsNoParkingEvent (case sensitive) followed by one space then followed by vehicleID, finally the line ends with new line character. (server to client)
   3. The line begin with invalidVehicleID (case sensitive) followed by one space then followed by vehicleID, finally the line ends with new line character. (server to client)
   4. The line begin with error (case sensitive). (server to client)

Use case 3: Retrieving list of Parking Events

1. **retrievingListOfParkingEvents** vehicleID

The line begin with getParkingEventsList (case sensitive) followed by one space then followed by vehicleID, finally the line ends with new line character. (client to server)

1. **handleSendParkingEventsList** vehicleID
2. **(Description: display the parking event list with total amount of money of all parking events)**

The line begin with sendParkingEventsList (case sensitive) followed by one space followed by vehicleID followed by one space followed by parking event list in json-string format followed by one space followed by total amount of money, finally the line ends with new line character. (server to client)

1. **(Description: show "There are no parking events" and the total amount of all parking events is 0)**

The line begin with sendParkingEventsListIsEmpty (case sensitive) followed by one space followed by vehicleID, finally the line ends with new line character. (server to client)

1. The line begin with invalidVehicleID (case sensitive) followed by

one space then followed by vehicleID, finally the line ends with new line character. (server to client)

1. The line begin with error (case sensitive). (server to client)

Use case 4: Investigating parked vehicle

1. **investigatingParkedVehicle** vehicleID, parkingSpaceNumber

The line begin with checkVehicle (case sensitive) followed by one space

followed by vehicleID followed by one space followed by parkingSpaceNumber, finally the line ends with new line character. (client to server)

1. **handleCheckVehicleEvent** vehicleID, parkingSpaceNumber
2. The line begin with checkVehicleOk (case sensitive) followed by one space followed by vehicleID followed by one space followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
3. The line begin with checkVehicleNotOk (case sensitive) followed by one space followed by vehicleID followed by one space followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
4. The line begin with invalidVehicleID (case sensitive) followed by one space then followed by vehicleID followed by one space then followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
5. The line begin with invalidParkingSpaceNumber (case sensitive) followed by one space then followed by vehicleID followed by one space then followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
6. The line begin with error (case sensitive). (server to client)

Use case 5: After the fact vehicle check

1. **checkPastParking** vehicleID, parkingSpaceNumber, givenTime

The line begin with checkVehicle (case sensitive) followed by one space followed by vehicleID followed by one space followed by parkingSpaceNumber followed by one space followed by givenTime "yyyy-MM-ddTHH:mm" format, finally the line ends with new line character. (client to server)

1. **handleCheckPastParkingEvent** vehicleID, parkingSpaceNumber, givenTime
2. The line begin with checkPastParkingOk (case sensitive) followed by one space followed by vehicleID followed by one space followed by parkingSpaceNumber followed by one space followed by givenTime "yyyy-MM-ddTHH:mm" format, finally the line ends with new line character. (server to client)
3. The line begin with checkPastParkingNotOk (case sensitive) followed by one space followed by vehicleID followed by one space followed by parkingSpaceNumber followed by one space followed by givenTime "yyyy-MM-ddTHH:mm" format, finally the line ends with new line character. (server to client)
4. The line begin with invalidVehicleID (case sensitive) followed by one space followed by vehicleID followed by one space followed by parkingSpaceNumber followed by one space followed by givenTime "yyyy-MM-ddTHH:mm" format, finally the line ends with new line character. (server to client)
5. The line begin with invalidParkingSpaceNumber (case sensitive) followed by one space followed by vehicleID followed by one space followed by parkingSpaceNumber followed by one space followed by givenTime "yyyy-MM-ddTHH:mm" format, finally the line ends with new line character. (server to client)
6. The line begin with error (case sensitive). (server to client)

Use case 6: Parking Space Usage Report

1. **getSpaceReport** parkingSpaceNumber

The line begin with getSpaceReport (case sensitive) followed by one space followed by parkingSpaceNumber, finally the line ends with new line character. (client to server)

1. **handleGetSpaceReportEvent** parkingSpaceNumber
2. **[Description: show list of times that the space was parked in(the stop time field left blank if event in start state)]**

The line begin with sendSpaceReport (case sensitive) followed by one space followed by parkingSpaceNumber followed by one space followed by parking space report in json-string format, finally the line ends with new line character. (server to client)

1. **[Description: show empty list]**

The line begin with sendEmptySpaceReport (case sensitive) followed by one space followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)

1. The line begin with invalidParkingSpaceNumber (case sensitive) followed by one space followed by parkingSpaceNumber, finally the line ends with new line character. (server to client)
2. The line begin with error (case sensitive). (server to client)

**2.The messages that will communicate Mulligan server with other Mulligan servers:**

1. **syncDatabasesBetweenServers** sourceDatabaseName sourceLastSyncTime

**(now we in source server)**

The line begin with sync (case sensitive) followed by one space followed by sourceDatabaseName, followed by one space followed by sourceLastSyncTime, finally the line ends with new line character. (source server to target server)

1. **handleSyncEvent** targetDatabaseName targetLastSyncTime

**(now we in target server)**

**[if both server have same lastTimeSync] or**

**[if targetLastSyncTime less than sourceLastSyncTime, target server copy source server database and then send same message]**

1. The line begin with syncStarted (case sensitive) followed by one space followed by targetDatabaseName followed by one space followed by targetLastSyncTime, finally the line ends with new line character. (target server to source server)

**[if targetLastSyncTime greater than sourceLastSyncTime, source server copy target server database and then send same message]**

1. The line begin with syncAfterCopy (case sensitive) followed by one space followed by targetDatabaseName followed by one space followed by targetLastSyncTime, finally the line ends with new line character. (target server to source server)

**[if error happens while trying to sync we send syncFailed message]**

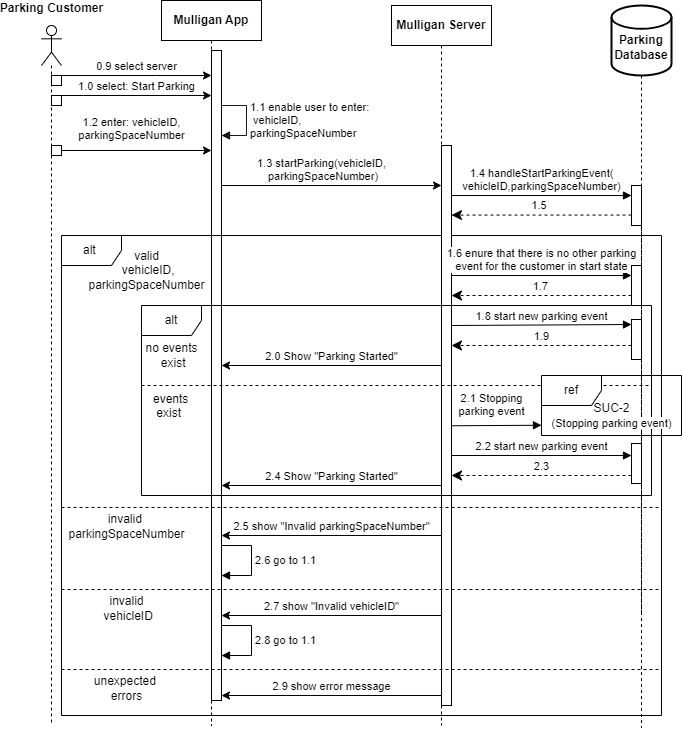
1. The line begin with syncFailed (case sensitive), finally the line ends with new line character. (target server to source server)

**[if error happens but not related to sync we send error message]**

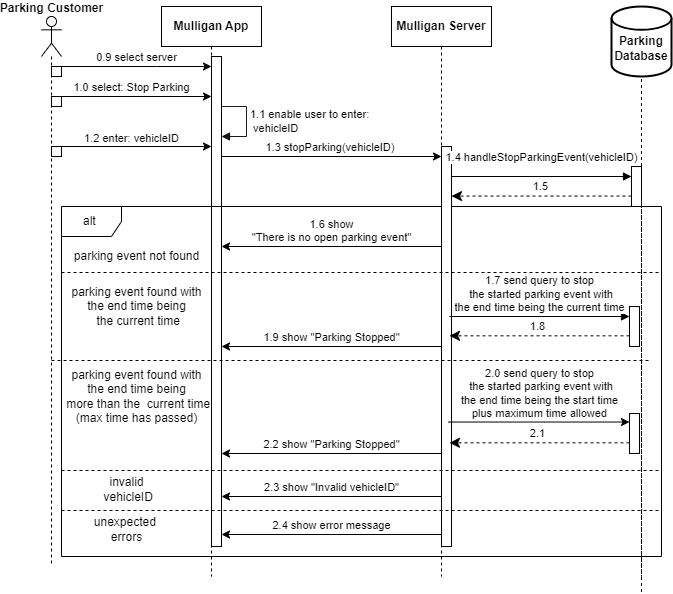
1. The line begin with error (case sensitive). (target server to source server)

**3.The Sequence diagrams of each use case (Server-Client):**

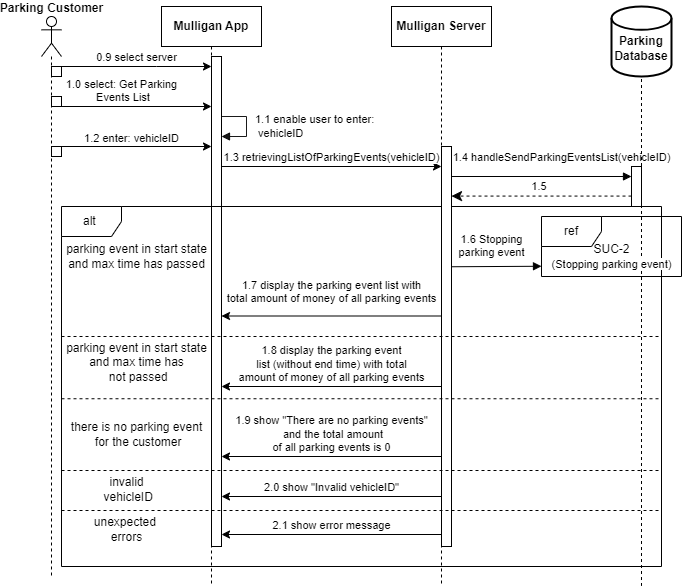
Use case 1: Starting a Parking Event



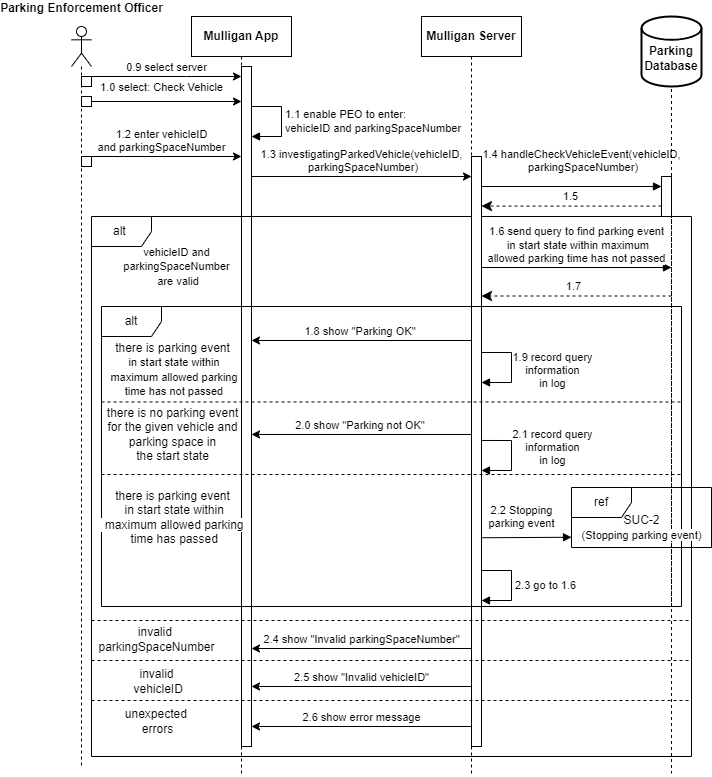
Use case 2: Stopping a Parking Event



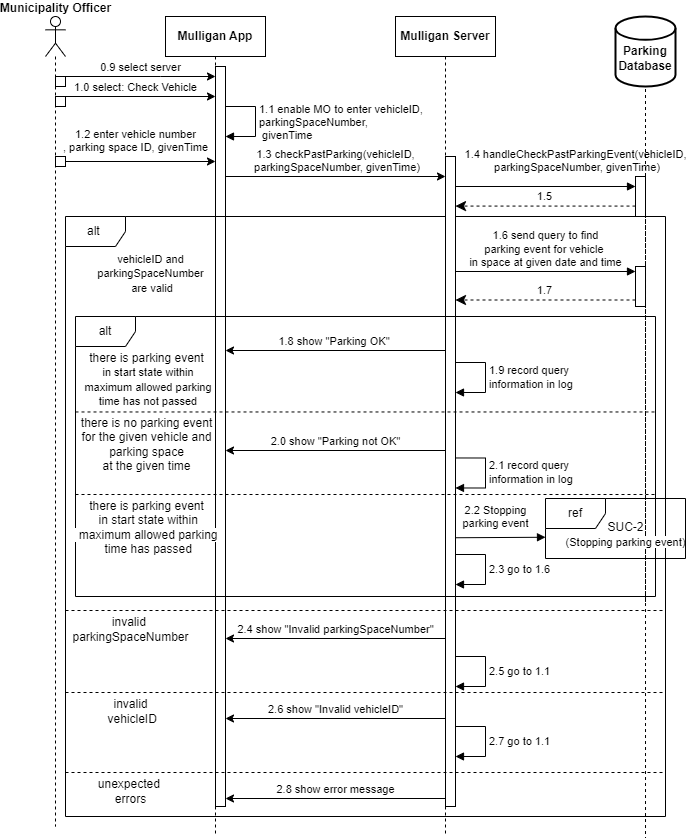
Use case 3: Retrieving list of Parking Events



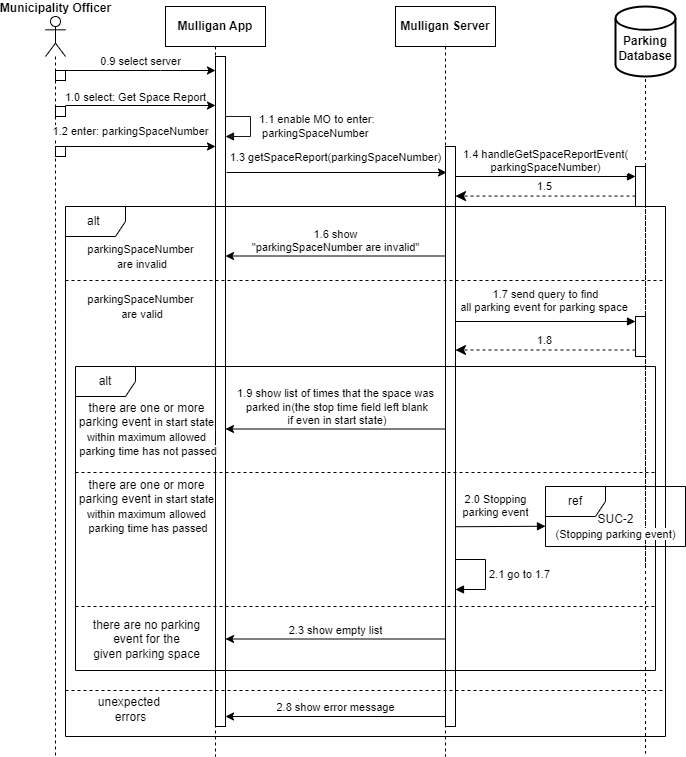
Use case 4: Investigating parked vehicle

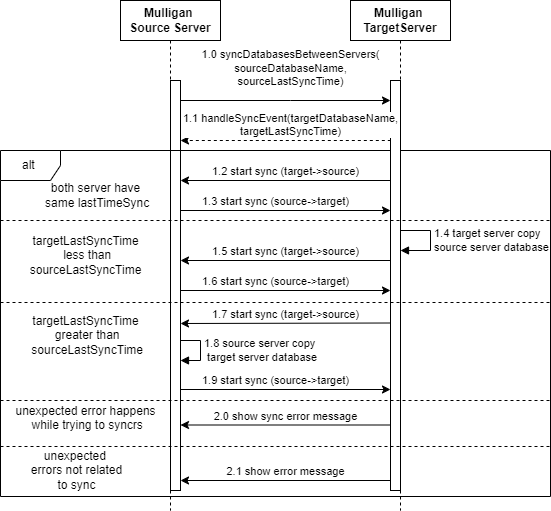


Use case 5: After the fact vehicle check



Use case 6: Parking Space Usage Report



**4.The Sequence diagrams of use case that sync Databases between Servers (Server-Server):**