

# Test cases Document for Backend testing - Iteration 4

Extended System test cases of "Test Cases\_reviewed.pdf"

Date: 7.6.2016

Last modifications: 22.6.2016

Performed by mzeen

Version: 0.2 in development, on commit 449e3b5

Tests marked with *Redone*: Redone after implementation of temporary work-around to comply with interface documentation

Performed by ricarda42

Date 26.6.2016

Version: commit b07e6de in branch developmentIt3

System test cases extended for iteration 4; redone tests from iteration 3 also to be tested with cleaned-up code

Sr.no	Use Case	Function Being Tested	Initial System State	Input	Expected Output	Test execution documentation
1	IC_C1	SF_S3: Send list of stops	Stop table exists in database	./stops	List of stops with stopId, stop name, line passing through the stops, location of the stop, bus schedule of the stop is displaying. Schedule contains line name, lineId, stopId, arriving time and timestamp.	Passed
2	IC_C2	SF_S5: Send properties of bus			<i>Not needed</i>	
3		SF_S6: Send GPS-Data	GPS data was sent to database	./busses/{busId}/	Current GPS Data coordinates with timestamp	Passed
4	IC_C3	SF_S6: Send GPS-Data			See above	
5	IC_B1	SF_S1: Send list of lines		./lines	List of Lines with properties lineId, name, routeId, timestamp and busses running on that specific line	Passed

6		<b>SF_S2: Send list of busses</b>	Bus table exists in database	./busses	List of all busses with their busId, number plate, color, picture and the line they serve	Passed
7		<b>SF_S4: Send list of routes</b>	Route table exists in database	./routes	List of all routes with routeId, route coordinates in geojson format and timestamp	Passed
8		<b>SF_S7: Store bus and line</b>	Bus and line tables exist in database	./updateBusStatus  Method: POST Content type: application/json Content: { "lineId":number, "busId":number }	Result status 200, Bus and line information stored	Passed; tested with lineId = 3, busId = 2, with initially bus 2 on line 1. Output of post is 200, calling ./lines reveals that bus 2 is now on line 3.
9	IC_B2	<b>SF_S13: Store number of available seats (see No. 12 – identical interface)</b>		./realTimeData  Method:POST Content type: application/json Content: { "busId":number, "position": { "type": "Point", "coordinates": [double, double]}, "takenSeats": number "timestamp": timeStamp }	Result status 200; GPS data and taken seats stored	
10	IC_B3	<b>SF_S4: Send list of routes</b>			See above	
11	IC_B4	<b>SF_S3: Send list of stops</b>			See above	

12	-	<b>SF_S8: Store GPS data</b>	Bus table exists in database	./realTimeData  Method: POST Content type: application/json Content: <pre>{   "busId":number,   "position": { "type":     "Point", "coordinates":     [double, double]},   "timestamp":     timeStamp }</pre>	Result status 200, GPS Data of bus is stored, timestamp not changed	Passed; With busId = 1, coordinates [15.0, -3.4] and timeStamp = 1098692636000 the output of POST is 200, calling ./busses/1 reveals that coordinates have been changed to [15.0, -3.4]. The timeStamp remains 1098692634000.
13	-	<b>SF_S9: Send latest timestamp</b>	Line, bus, route and stops tables exist in database	./update	Last timestamps for bus, route, line and stops	Passed
14	IC_B5	<b>SF_S10: Notification of custom stop request</b>  <b>Post custom stop request</b>	CitizenApp sends new Custom Stop Request	./customStops  Method: POST Content type: application/json Content: <pre>{   "lineId":number,   "location": { "type":     "Point",     "coordinates":     [double , double] },   "numberOfPersons":     number,   "deviceId":String,   "info": {     "name":       "string",     "address":       "string",     "assistance":       [number,number...]   } }</pre>	Result status 200; New custom stop with assigned requestId, lineId, pickUpTime, location, number of persons, deviceId, information of passenger included name, address, and assistance, status and timestamp	

				<pre>         }         "pickUpTime":Timestamp       } </pre>		
15.1		<b>Get custom stop requests ref. to line Valid 1</b>	Request from CitizenApp with lineId stored where global status is 1. DriverApp requests customStops for given lineId and busId. The bus has not yet rejected the custom stop (i.e. busId not in rejectingBus list)	./customStops?lineId={lineId}&busId={busId}	Custom stop request included in output list	
15.2		<b>Get custom stop requests ref. to line Invalid 1</b>	Request from CitizenApp with lineId stored where global status is 1. DriverApp requests customStops for given lineId and busId. The bus has already rejected the custom stop (i.e. busId in rejectingBus list)	./customStops?lineId={lineId}&busId={busId}	Custom stop request not included in output list	
15.3		<b>Get custom stop requests ref. to line Valid 2</b>	Request from CitizenApp with lineId stored. DriverApp requests customStops for given lineId and busId. Global status is 2. The bus has accepted this custom stop (i.e. acceptingBus is busId).	./customStops?lineId={lineId}&busId={busId}	Custom stop request included in output list.	
15.4		<b>Get custom stop requests ref. to line Invalid 2</b>	Request from CitizenApp with lineId stored. DriverApp requests customStops for given lineId and busId. Global status is 2. The bus has accepted this custom stop (busId is not acceptingBus)	./customStops?lineId={lineId}&busId={busId}	Custom stop request not included in output list.	

15.5		<b>Get custom stop requests ref. to line Invalid 3</b>	Request from CitizenApp with lineId stored. DriverApp requests customStops for given lineId and busId. All busses on this line rejected this custom stop, i.e. global status is 3.	./customStops?lineId={lineId}&busId={busId}	Custom stop not in output list	
15.6		<b>Get custom stop requests ref. to line Invalid 4</b>	Request from CitizenApp with lineId stored. DriverApp requests customStops for given lineId and busId. A bus accepted and completed this custom stop, i.e. global status is 4.	./customStops?lineId={lineId}&busId={busId}	Custom stop not in output list	
15.7		<b>Get custom stop requests ref. to line Invalid 5</b>	Request from CitizenApp with lineId stored. DriverApp requests customStops for given lineId and busId. A bus accepted this custom stop, but the citizen did not show up, i.e. global status is 5.	./customStops?lineId={lineId}&busId={busId}	Custom stop not in output list	
15.8		<b>Get custom stop requests ref. to line Valid 3</b>	Request from CitizenApp with lineId stored. DriverApp requests customStops for given lineId and busId. This bus accepted this custom stop, but the citizen canceled it, i.e. global status is 6 and busId is acceptingBus.	./customStops?lineId={lineId}&busId={busId}	Custom stop is in output list	
15.9		<b>Get custom stop requests ref. to line Invalid 6</b>	Request from CitizenApp with lineId stored. DriverApp requests customStops for given lineId and busId. A bus different from this bus accepted this custom stop, but the citizen	./customStops?lineId={lineId}&busId={busId}	Custom stop is not in output list	

			canceled it, i.e. global status is 6 and busId is not acceptingBus.			
15.10		<b>Get custom stop requests ref. to line Invalid 7</b>	DriverApp requests pending customStops for given lineId and busId; no requests for this line	./customStops?lineId={lineId}&busId={busId}	Empty list	
15.11		<b>Get custom stop requests ref. to line Invalid 6</b>	DriverApp requests pending customStops for given lineId and busId; requests for that line have pickUpTime before current time	./customStops?lineId={lineId}&busId={busId}	Empty list	
16.1		<b>SF_S11: Response to custom stop request</b>  <b>Update status of custom stop requests Valid 1</b>	DriverApp of a given bus received custom stop requests for the given line. The driver accepts a custom stop with status pending.	./customStops/{requestId}  Method: POST Content type: application/json Content: { "status":2 "busId": number }	Result status 200; Status of given request set to 2, acceptingBus set to busId	
16.2		<b>Update status of custom stop requests Invalid 1</b>	DriverApp of a given bus received custom stop requests for the given line. The driver accepts a custom stop where the status has already been changed.	./customStops/{requestId}  Method: POST Content type: application/json Content: { "status":2 "busId": number	Result status 200; status of request unchanged	

				}		
16.3		<b>Update status of custom stop requests Valid 2</b>	DriverApp of a given bus received custom stop requests for the given line. The driver rejects a custom stop with status pending. Not all busses on this line have already rejected this custom stop.	./customStops/{requestId}  Method: POST Content type: application/json Content: { "status":3 "busId": number }	Result status 200; status of request unchanged, busId added to list rejectingBus	
16.4		<b>Update status of custom stop requests Valid 3</b>	DriverApp of a given bus received custom stop requests for the given line. The driver rejects a custom stop with status pending. All busses on this line have already rejected this custom stop.	./customStops/{requestId}  Method: POST Content type: application/json Content: { "status":3 "busId": number }	Result status 200; status of request set to 3, busId added to list rejectingBus	
16.5		<b>Update status of custom stop requests Valid 4</b>	DriverApp of a given bus received custom stop requests for the given line. The driver rejects a custom stop where the status has already been changed.	./customStops/{requestId}  Method: POST Content type: application/json Content: { "status":3 "busId": number }	Result status 200; status of request unchanged, busId added to rejectingBus list	
16.6		<b>Update status of custom stop</b>	DriverApp of a given bus received custom stop requests for the given line. The driver accepted a	./customStops/{requestId}  Method: POST	Result status 200; status of request set to 4	

		<b>requests Valid 5</b>	custom stop (i.e. status is 2 and acceptingBus equals busId) and now marks it as complete.	Content type: application/json Content: { "status":4 "busId": number }		
16.7		<b>Update status of custom stop requests Invalid 2</b>	DriverApp of a given bus received custom stop requests for the given line. A driver different from this driver accepted a custom stop (i.e. status is 2 and acceptingBus is not equal to busId) This bus driver now marks this custom stop as complete.	./customStops/ {requestId}  Method: POST Content type: application/json Content: { "status":4 "busId": number }	Result status 200; status of request unchanged	
16.8		<b>Update status of custom stop requests Invalid 3</b>	DriverApp of a given bus received custom stop requests for the given line. This driver accepted a custom stop but the status was updated further (i.e. status is not 2 and acceptingBus is equal to busId). This bus driver now marks this custom stop as complete.	./customStops/ {requestId}  Method: POST Content type: application/json Content: { "status":4 "busId": number }	Result status 200; status of request unchanged	
16.9		<b>Update status of custom stop requests Invalid 4</b>	DriverApp of a given bus received custom stop requests for the given line. The bus driver now marks a custom stop as complete he did not accept and where the status is not accepted.	./customStops/ {requestId}  Method: POST Content type: application/json Content: { "status":4	Result status 200; status of request unchanged	



				"busId": number }		
16.1 0		<b>Update status of custom stop requests Valid 6</b>	DriverApp received custom stop requests for the given line. The driver accepted a custom stop (i.e. status is 2 and acceptingBus equals busId) and now marks it as not shown up.	./customStops/ {requestId}  Method: POST Content type: application/json Content: { "status":5 "busId": number }	Result status 200; status of request set to 5	
16.1 1		<b>Update status of custom stop requests Valid 7</b>	DriverApp of a given bus received custom stop requests for the given line. The driver accepted a custom stop (i.e. status is 2 and acceptingBus equals busId) and now marks it as not shown up.	./customStops/ {requestId}  Method: POST Content type: application/json Content: { "status":5 "busId": number }	Result status 200; status of request set to 5	
16.1 2		<b>Update status of custom stop requests Invalid 5</b>	DriverApp of a given bus received custom stop requests for the given line. A driver different from this driver accepted a custom stop (i.e. status is 2 and acceptingBus is not equal to busId) This bus driver now marks this custom stop as not shown up.	./customStops/ {requestId}  Method: POST Content type: application/json Content: { "status":5 "busId": number }	Result status 200; status of request unchanged	
16.1 3		<b>Update status of custom</b>	DriverApp of a given bus received custom stop requests for the given line.	./customStops/ {requestId}	Result status 200; status of request unchanged	

		<b>stop requests Invalid 6</b>	This driver accepted a custom stop but the status was updated further (i.e. status is not 2 and acceptingBus is equal to busId). This bus driver now marks this custom stop as not shown up.	Method: POST Content type: application/json Content: { "status":5 "busId": number }		
16.1 4		<b>Update status of custom stop requests Invalid 7</b>	DriverApp of a given bus received custom stop requests for the given line. The bus driver now marks a custom stop as not shown up he did not accept and where the status is not accepted.	./customStops/{requestId}  Method: POST Content type: application/json Content: { "status":5 "busId": number }	Result status 200; status of request unchanged	
16.1 5		<b>Update status of custom stop requests Valid 8</b>	DriverApp of a given bus received custom stop requests for the given line. The citizen cancels the custom stop.	./customStops/{requestId}  Method: POST Content type: application/json Content: { "status":6 "busId": number }	Result status 200; status of request set to 6	
16.1 7		<b>Update status of custom stop requests Valid 9</b>	DriverApp of a given bus received custom stop requests for the given line. The citizen cancels the custom stop.	./customStops/{requestId}  Method: POST Content type: application/json Content: {	Result status 200; status of request set to 6	

				"status":6 }		
18.1		Get custom stop requests ref. to requestId and deviceId	CitizenApp has the requestId	./customStops?requestId={requestId}&deviceId={deviceId}	Custom stop request for the given requestId with all properties	<i>Redone:</i> Passed
18.2		Get custom stop requests ref. to requestId	CitizenApp has the requestId	./customStops?requestId={requestId}	Custom stop request for the given requestId with all properties	
18.3		Get custom stop requests ref. to deviceId	CitizenApp has the requestId	./customStops?deviceId={deviceId}	All custom stops for the given deviceId with all properties	
19	IC_B6	SF_S11: Response to custom stop request	DriverApp send status as complete	See above with status complete		

**Author:** Ricarda Rosemann

**Status:** Complete

**Reviewer:**

**Review Status:**