

Feature 1

Test 1 Title: Import valid files

Objective: The user is able to upload valid GTFS files into the application

Preconditions:

- Valid GTFS files are available
- The app must be running

Procedure:

1. The user opens the application and selects import files
2. The user is prompted to enter the file location
3. The user selects the individual files

Expected Result: The user gets confirmation that all of the files he selected were imported to the app. The application data structures are valid and filled with data from the files correctly.

Test 2 Title: Import invalid files

Objective: The user attempts to import the files into the app, some of the files are invalid

Preconditions:

- Invalid GTFS file is available
- The app must be running

Procedure:

1. The user selects the option to import GTFS files
2. The user is prompted to enter the file location
3. The user selects the individual files

Expected Result: The user is shown an error message saying some of the files are invalid.

Test 3 Title: Files missing on import

Objective: The user attempts to import the files into the app, the app cannot open some of the files

Preconditions:

- The app must be running

Procedure:

1. The user opens the app and is prompted to import the files
2. The user is prompted to enter the file location
3. The user selects the individual files

Expected Result: The user gets an error message telling him that some of the files cannot be opened or that the user didn't select all required files

Feature 2

Test 1 Title: Valid data

Objectives: User views the distance of each trip

Application is running and valid data is in the data structure

- User knows the trip_id

Procedure:

1. User selects the option to view all the trips.

Expected Result: A list of valid distances is shown for all trips

Test 2 Title: Invalid data

Objectives: User views the distance of each trip that has valid data

Application is running and valid data is in the data structure

- User knows the trip_id

Procedure:

1. User selects the option to view all the trips.

Expected Result: A list of valid distances is shown for all trips with valid data. Trips with invalid data show a 0* indicating that there is an error in the data.

Feature 3

Test 1 Title: Valid data

Objectives: User views the average speed traveled on any trip

Application is running and valid data is in the data structure

- User knows the trip_id

Procedure:

1. User selects the option to view all the trips.

Expected Result: A list of correct average speeds is shown for all trips

Test 2 Title: Invalid data

Objectives: User views the average speeds of each trip that has valid data

Application is running and valid data is in the data structure

- User knows the trip_id

Procedure:

1. User selects the option to view all the trips.

Expected Result: A list of correct average speeds is shown for all trips with valid data. Trips with invalid data show a 0* indicating that there is an error in the data.

Feature 4

Test 1 Title: Valid data

Objectives: User views the number of trips each stop is on

Application is running and valid data is in the data structure

- User knows the stop_id

Procedure:

1. User selects the option to view all the stops.

Expected Result: Each stop should show the valid number of instances that stop is found within all the trips. Including 0.

Feature 5

Test 1 Title: Search by valid stop_id

Objectives: User searches for routes that contains a specific stop by stop_id

Preconditions:

- Application is running and valid data is in the data structure
- User knows the stop_id
- Stop_id is valid

Procedure:

2. User selects search by stop_id option.
3. User prompted for stop_id.
4. User inputs stop_id.

Expected Result: A list of routes are displayed that stop at the specified stop_id.

Test 2 Title: Search by invalid stop_id

Objective: User attempts to search for routes that contain a specific stop but uses an invalid stop_id

Preconditions:

- Application is running and valid data is in the data structure
- Stop_id is invalid

Procedure:

1. User selects search by stop_id option.
2. User prompted for stop_id.
3. User inputs stop_id.

Expected Result: An error message is shown informing the user that the stop_id does not exist.

Test 3 Title: Search without valid data

Objective: User attempts to search for routes by stop_id without the Data Structures being populated with valid data

Preconditions:

- Application is running
- Data Structures are not populated with valid data
- User knows the stop_id

Procedure:

1. User selects search by stop_id option.

Expected Result: An error message is shown informing the user that valid files are not loaded.

Feature 6

Test 1 Title: Search for a route by route_id

Objective: User searches for route by route_id and sees all the stops on the route

Preconditions:

- Application is running and valid data is in the data structure
- Valid route_id is known

Procedure:

1. User chooses a search by route_id option
2. User enters valid route_id

Expected Result: User sees all the stops associated with the route

Test 2 Title: Search for a route by invalid route_id

Objective: User searches for route using an invalid route_id

Preconditions:

- Application is running and valid data is in the data structure
- Invalid route_id is known

Procedure:

1. User chooses a search by route_id option
2. User enters an invalid route_id

Expected Result: User sees an error message indicating that the route_id is invalid

Feature 7

Test 1 Title: Route search showing trips

Objective: Search for a route by route_id and display the trip_id for any trips happening in the future

Preconditions:

- Application is running and valid data is in the data structure
- User must know a valid route_id
- Route_id should have at least one trip happening in the future

Procedure

1. User selects the search for a route to display trips option
2. User enters the route_id

Expected Result: Trip_ids for any trips happening in the future are displayed.

Test 2 Title: Route search showing trips if no trips exist

Objective: Search for a route by route_id and display no trips found if there are no trips

Preconditions:

- Application is running and valid data is in the data structure
- User must know a valid route_id
- Route_id should have no trips happening in the future

Procedure

1. User selects the search for a route to display trips option
2. User enters the route_id

Expected Result: User is informed that the route does not have any trips happening in the future.

Test 3 Title: Route search with an invalid route_id

Objective: Search for a route using an invalid route_id

Preconditions:

- Application is running and valid data is in the data structure
- User must know an invalid route_id

Procedure

1. User selects the search for a route to display trips option
2. User enters the route_id

Expected Result: User is informed that the route_id is not valid

Feature 8

Test 1 Title: Stop search showing next trip

Objective: Search for a stop by stop_id and display the next trip

Preconditions:

- Application is running and valid data is in the data structure
- User must know a valid stop_id
- Stop_id should have at least one trip happening in the future

Procedure

1. User selects the search for a stop to display trips option
2. User enters the stop_id

Expected Result: Trip_id for the next trip is displayed.

Test 2 Title: Stop search showing next trip if no trips exist

Objective: Search for a stop by stop_id and display no trips found if there are no trips

Preconditions:

- Application is running and valid data is in the data structure
- User must know a valid stop_id
- Stop_id should have no trips happening in the future

Procedure

1. User selects the search for a stop to display trips option
2. User enters the stop_id

Expected Result: User is informed that the stop does not have any trips happening in the future.

Test 3 Title: Stop search with an invalid stop_id

Objective: Search for a stop using an invalid stop_id

Preconditions:

- Application is running and valid data is in the data structure
- User must know an invalid stop_id

Procedure

1. User selects the search for a route to display trips option
2. User enters the stop_id

Expected Result: User is informed that the stop_id is not valid

Feature 9

Test 1 Title: Plot stops on a Google Map

Objective: Plot all stops with the correct color on a given route on a Google Map.

Preconditions:

- Application is running and valid data is in the data structure
- Valid route_id must be known
- Google Maps API must be available

Procedure

1. User selects display routes by route_id option
2. User enters the route_id

Expected Result: The user is shown a Google Map with the selected route stops displayed on the map in the appropriate color.

Test 2 Title: Plot based on an invalid route

Objective: After selecting an invalid route_id, the user told to enter a valid id.

Preconditions:

- Application is running and valid data is in the data structure
- Invalid route_id must be known
- Google Maps API must be available

Procedure

1. User selects display routes by route_id option
2. User enters the route_id

Expected Result: The user is told that the route_id is not valid.

Test 3 Title: Plot attempt without access to Google Maps

Objective: The user should be notified if Google Maps is not available.

Preconditions:

- Application is running and valid data is in the data structure
- route_id must be known
- Google Maps API must not be available

Procedure

1. User selects display routes by route_id option
2. User enters the route_id

Expected Result: The user is told that Google Maps is not currently available.

Feature 10

Test 1 Title: Plot based on a valid route

Objective: After selecting a valid route_id, the user is presented with a plot of the current location of all busses scheduled on the route.

Preconditions:

- Application is running and valid data is in the data structure
- Valid route_id must be known
- Google Maps API must be available
- At least one bus is scheduled to be active on the selected route at the current time

Procedure

1. User selects search by route_id option
2. User enters the route_id

Expected Result: The user is shown a Google Map with the scheduled bus locations for the current time shown as pins on the map.

Test 2 Title: Plot based on a valid route without busses

Objective: After selecting a valid route_id, the user is informed that no busses are currently scheduled for the route.

Preconditions:

- Application is running and valid data is in the data structure
- Valid route_id must be known
- Google Maps API must be available
- No busses are scheduled to be active on the selected route at the current time

Procedure

1. User selects search by route_id option
2. User enters the route_id

Expected Result: The user is informed that no busses are currently scheduled for the given route

Test 3 Title: Plot based on an invalid route

Objective: After selecting an invalid route_id, the user told to enter a valid id.

Preconditions:

- Application is running and valid data is in the data structure
- Invalid route_id must be known
- Google Maps API must be available

Procedure

1. User selects search by route_id option
2. User enters the route_id

Expected Result: The user is told that the route_id is not valid.

Test 4 Title: Plot attempt without access to Google Maps

Objective: The user should be informed that access to Google Maps is currently unavailable.

Preconditions:

- Application is running and valid data is in the data structure

- route_id must be known
- Google Maps API must not be available

Procedure

1. User selects search by route_id option
2. User enters the route_id

Expected Result: The user is told that Google Maps is not currently available.

Definition:

Entities	Modifiable Attributes
Stop	stop_id, stop_name, stop_lat, stop_lon
Trip	trip_id, arrival_time, departure_time, stop_sequence
Route	route_id, route_color

Table 1, Attributes

Feature 11

Test 1 Title: Update any attribute (see Table 1)

Objective: Update any attribute of a stop_time, stop, route, or trip (see Table 1)

Preconditions:

- Application is running and valid data is in the data structure
- User knows the attributes they want to modify
- User knows how they want to modify the attribute

Procedure

1. The user selects the entity they want to modify
2. The user selects the attribute they want to modify
3. The user enters the updated information
4. The user confirms the update

Expected Result: The data structure is updated accordingly for the attribute as per the user's input.

Test 2 Title: Update an attribute with invalid data

Objective: Attempt to update an attribute with invalid data

Preconditions:

- Application is running and valid data is in the data structure
- User knows the entity or attribute they want to modify in an invalid way

Procedure

1. The user selects the entity they want to modify
2. The user selects the attribute they want to modify
3. The user enters their modification
4. The user confirms the change

Expected Result: The user is informed what about their entry was invalid

Feature 12

Test 1 Title: Update any attribute (see Table 1) of a trip group

Objective: Update any attribute (see Table 1) of a group of similar trips

Definition of “similar”: Any two trips are similar if their starting and ending stops are the same

Preconditions:

- Application is running and valid data is in the data structure
- The user knows the attribute they want to modify
- The user has a valid trip_id
- The chosen trip contains at least one similar trip

Procedure

1. The user selects the option to edit a group of similar trips
2. The user enters the trip_id
3. The user enters the attribute of the group that they want to update.
4. The user confirms the change

Expected Result: The data structure is updated accordingly for all similar trips according to the user input.

Test 2 Title: Group update with invalid attributes

Objective: Attempt to update a trip group with invalid attribute

Preconditions:

- Application is running and valid data is in the data structure
- The user knows the attribute they want to modify
- The user has a valid trip_id
- Attribute is not valid

Procedure

1. The user selects the option to edit a group of similar trips
2. The user enters the invalid trip_id
3. The user enters the invalid attribute
4. The user confirms the change

Expected Result: The user is presented with a dialog that explains the error with the attempted update

Test 3 Title: Group update with an invalid trip_id

Objective: Attempt to update a trip group with invalid trip_id

Preconditions:

- Application is running and valid data is in the data structure
- The user knows the attribute they want to modify
- The user has a invalid trip_id

Procedure

1. The user selects the option to edit a group of similar trips
2. The user enters the invalid trip_id
3. The user enters an attribute
4. The user confirms the change

Expected Result: The user is presented with a dialog that explains the error with the trip_id

Feature 13

Test 1 Title: Changing stop location on the map

Objective: The user is able to click on a stop and drag it to change its location

Preconditions:

- Application is running and valid data is in the data structure
- Stop to modify is known
- Google Maps API is available and functional

Procedure

1. User selects a stop on the map
2. User drags the stop to the new location
3. A list of trips served by the stop are shown
4. User confirms the change

Expected Result: The changes to the stop location are stored successfully in the data structure.

Feature 14

Test 1 Title: Changing stop times

Objective: The user is able to click on a stop and edit individual trips' arrival and departure times

Preconditions:

- Application is running and valid data is in the data structure
- Stop to modify is known
- Stop to modify is part of a trip
- User has valid times to enter
- Google Maps API is available and functional

Procedure

1. User selects a stop on the map
2. A list of trips served by the stop are shown
3. The user selects the trip to edit
4. User enters the updated arrival and departure times
5. User confirms the change

Expected Result: The changes to the stop arrival and departure times are stored successfully in the data structure.

Test 2 Title: Changing stop times with an invalid arrival or departure time

Objective: The user is given notice if they enter an invalid arrival or departure time

Preconditions:

- Application is running and valid data is in the data structure
- Stop to modify is known
- Stop to modify is part of a trip
- User has valid times to enter
- Google Maps API is available and functional

Procedure

1. User selects a stop on the map
2. A list of trips served by the stop are shown
3. The user selects the trip to edit
4. User enters invalid arrival and departure times
5. User confirms the change

Expected Result: The user is notified that the time is not valid and the data structure is not altered.

Feature 15

Test 1 Title: Exporting GTFS files with valid filepath

Objective: User can export valid GTFS files with a valid filepath

Preconditions:

- Application is running and valid data is in the data structure
- File location is known and has adequate space for storage

Procedure:

1. User selects the export option
2. User selects the desired file location

Expected Result: Valid GTFS files are located in the specified location and the user is shown a dialog indicating the success of the operation.

Test 2 Title: Exporting GTFS files with an invalid filepath

Objective: Inform user that a valid filepath is required

Preconditions:

- Application is running and valid data is in the data structure

Procedure:

1. User selects the export option
2. User enters an invalid filepath

Expected Result: The user is alerted that the filepath is not valid