**First phase: Account Manager**

At the starting time, we first want to be sure about the user’s details, testing if all the user’s data (in terms of credentials and preferences) are correctly stored in the database.

|  |  |
| --- | --- |
| TestID | t1 |
| Name | Registration of a new account |
| Components to be tested | AccountManager, DBMS |
| Input | Username, Password, Email |
| Output | Check if the new account has been correctly stored in the database |
| Description | Account Manager contacts the DBMS in order to add the new account to the list of existing ones. It must happens only after clicking on the confirmation link sent by email. |
| Exception | UsernameAlreadyInUseException: the account must not be stored.  InvalidEmailFormatException: the account must not be stored.  PasswordConstraintsViolated: the account must not be stored. |

|  |  |
| --- | --- |
| TestID | t2 |
| Name | Login of an existing account |
| Components to be tested | AccountManager, DBMS |
| Input | Username, Password |
| Output | Check if the login has been performed |
| Description | Account Manager contacts the DBMS in order to check the credentials. |
| Exception | NotExistingAccount: the client must remain in “not registered” status. |

**Second phase: Account Manager, AppointmentManager**

From now on, we enter the most dense part of the project. Here we want to to test all the insertions dynamics.

For instance: insertion in lunch time should fail, it should be impossible to insert an appointment in the past, it should be impossible to insert overlapping appointments.

|  |  |
| --- | --- |
| TestID | t3 |
| Name | Insertion of a new appointment in an empty schedule |
| Components to be tested | AccountManager, AppointmentManager, DBMS |
| Input | Appointment details |
| Output | Check if the appointment has been inserted in the DBMS |
| Description | AppointmentManager stores the appointment details in the DBMS. |
| Exception | BreaksTimeException: if the appointment is held in the user’s breaks time, it must not be stored. |

|  |  |
| --- | --- |
| TestID | t4 |
| Name | Insertion of a new appointment in a non-empty schedule |
| Components to be tested | AccountManager, AppointmentManager, DBMS |
| Input | Appointment details |
| Output | Check if the appointment has been inserted in the DBMS |
| Description | Appointment Manager performs a consistency check and then contacts the DBMS in order to check whether the appointment has been stored. |
| Exception | OverlapsException: if there is an overlap, the appointment must not be stored.  ImpossibleToReachException: if there is not a valid path to move between the previous appointment and the draft appointment (with respect to user’s preferences, locations and travel time), the appointment must not be stored. |

**Third phase: AccountManager, AppointmentManager, ItineraryManager**

In this phase, we plan to test how the computation of itineraries is performed, paying attention to the 5 categories highlighted in RASD (Shortest, Most Ecologic, Cheapest, MinimumChanges, MinimumWalkingDistance).

|  |  |
| --- | --- |
| TestID | t4 |
| Name | Computation of an itinerary between two appointments |
| Components to be tested | AccountManager, AppointmentManager, ItineraryManager DBMS |
| Input | 2 Appointments’ details |
| Output | Check if the itinerary has been inserted in the DBMS and it is suitable to get in time to the appointment |
| Description | AppointmentManager and the ItineraryManager perform a consistency check, then the ItineraryManager should compute the optimal path, according to user’s preferences, retrieved by AccountManager |
| Exception | ImpossibleToReachException: if there is not a valid path to move between the previous appointment and the draft appointment (with respect to user’s preferences, locations and travel time), the appointment must not be stored |

**Fourth phase: AppointmentManager, ItineraryManager, NotificationHandler**

Here we test the notification feature of our system.

|  |  |
| --- | --- |
| TestID | t5 |
| Name | Notification of incoming appointment |
| Components to be tested | ItineraryManager, AppointmentManager, NotificationHandler |
| Input | Appointment details |
| Output | Check whether the application notifies the user with incoming appointment |
| Description | The system must notify the user of an incoming appointment 30 minutes earlier |
| Exception | - |

**Fifth phase: AppointmentManager, ItineraryManager, NotificationHandler, Navigator**

Eventually, we test how all the components interact and cooperate, in order to let the user get in time to the appointment, checking functions like ticket purchase and reserving a vehicle.

|  |  |
| --- | --- |
| TestID | t6 |
| Name | Navigation toward an appointment location |
| Components to be tested | Navigator, ItineraryManager, AppointmentManager, Maps API, WeatherAPI, PublicTransportationServices, NotificationHandler |
| Input | A given appointment |
| Output | Check if the application guides you to the location |
| Description | The navigator must retrieve information about the appointment from the AppointmentManager, the path from the ItineraryManager and the maps must be provided by proper APIs to move into the map. By exploiting this information, it must provide the users with indications toward the appointment location |
| Exception | GenericDelayException/WeatherException/TrafficException/StrikeException: the application must compute other itineraries and ask the user to select a new one (see RASD for further info). |

|  |  |
| --- | --- |
| TestID | t7 |
| Name | Reserving a shared vehicle |
| Components to be tested | Navigator, ItineraryManager, AppointmentManager, Maps API, Vehicle Sharing Service |
| Input | Start a navigation near a vehicle sharing-service |
| Output | Check whether the application redirects to an external system. |
| Description | The navigator must retrieve all the information about vehicle sharing-services nearby and let the user click on the available vehicles. After the click, the application must redirect the user to the external system. |
| Exception | NotSuitableVehicleException: if the vehicle sharing service is not useful to get in time to the appointment, the navigator must not propose it to the user. |

|  |  |
| --- | --- |
| TestID | t8 |
| Name | Buying a ticket |
| Components to be tested | Navigator, ItineraryManager, AppointmentManager, Maps API, PublicTransportationManager, PaymentManager |
| Input | Start a navigation with an itinerary that foresees public transportation. |
| Output | Check whether the application lets the user buy a transportation ticket. |
| Description | The navigator must retrieve the proper information from the scheduled public transportation service and ask the user if he wants to start a PayPal transaction to let the user buy a ticket, then the Payment Manager must handle the purchase. |
| Exception | StrikeException: the application must compute other itineraries and ask the user to select a new one (see RASD for further info).  AlreadySubscribedException: if the user has got a subscription for the given public transportation service, the system must not ask the user to buy a ticket. |