Chapter 1

Taxi Management System

(Caller, Called)

1.1 Request Management, Data Access Utilities

insertRequest(request)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A request with an id already existent in the database	An InvalidArgumentValueException is raised.
Formally valid arguments	An entry containing the request data is inserted into the database.

deleteRequest(request)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A request with a inexistent id	An InvalidArgumentValueException is raised.
Formally valid arguments	The entry containing the request data is deleted from the database.

getRequestList()	
Input	*Effect*
Nothing	The list of all pending requests.

insertReservation(reservation)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A reservation with an id already existent in the database	An InvalidArgumentValueException is raised.
Formally valid arguments	An entry containing the reservation data is inserted into the database.

deleteReservation(reservation)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A reservation with a inexistent id	An InvalidArgumentValueException is raised.
Formally valid arguments	The entry containing the reservation data is deleted from the database.

updateReservationList(Reservation[])	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An empty array	An InvalidArgumentValueException is raised.
An array containing some null values	A NullArgumentException is raised.
An array of non-null, but inexistent reservations	An InvalidArgumentValueException is raised.
An array of valid and existing reservations	The corresponding entries in the database are updated to set the reservation as completed.

getReservationList()	
Input	*Effect*
Nothing	The list of all pending reservations.

1.2 Reservation Management, Data Access Utilities

1.3 Taxi Management, Data Access Utilities Management

updateQueues(TaxiQueue[] tq)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An empty array	An InvalidArgumentValueException is raised.
An array containing some null values	A NullArgumentException is raised.
An array of non-null queues, but containing null values	An InvalidArgumentValueException is raised.
A non-empty array of valid queues	The content of the queues is updated in the database.

updateTaxiLocation(taxiId, location)	
Input	*Effect*
A null location	A NullArgumentException is raised.
A set of valid parameters	The new location of the taxi is written in the database.

updateTaxiStatus(taxiId, TaxiStatusAvailable)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A set of valid parameters	The taxi status is set to available in the database.

1.4 Taxi Management, Location Management

1.5 Reservation Management, Location Management

getZone(location)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A location whose coordinates are invalid	An InvalidLocationException is raised.
A location outside the city	An InvalidLocationException is raised.
A location inside the city	The id of the zone to which the location belongs.

checkTaxiDriverLocation(currentLocation, meetingPoint)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A currentLocation whose coordinates are invalid	An InvalidLocationException is raised.
A meetingPoint whose coordinates are invalid	An InvalidLocationException is raised.
A currentLocation outside the city	Returns false.
A meetingPoint outside the city	An InvalidLocationException is raised.
Both currentLocation and meetingPoint inside the city, but not within 50m from one another	Returns false.
Both currentLocation and meetingPoint inside the city, within 50m from one another	Returns false.

1.6 Reservation Management, Request Management

requestTaxi(passengerId, passengerLocation, destination)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A passengerId not correctly formatted	An InvalidArgumentFormatException is raised.
A passengerLocation whose coordinates are invalid	An InvalidLocationException is raised.
A destination whose coordinates are invalid	An InvalidLocationException is raised.
A passengerLocation outside the city	A passengerId not correctly formatted
A valid set of parameters	A new request is created and handled; refer to the RASD for the specific outcomes of this operation.

1.7 Request Management, Taxi Management

existsAvailableTaxiDriver(request, zone)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An inexistent zone	An InvalidArgumentValueException is raised.
A zone with invalid fields	An InvalidArgumentValueException is raised.
A valid set of parameters	true if a taxi driver is available to serve the request, false otherwise

getETA(taxiDriver, passengerLocation)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A passengerLocation whose coordinates are invalid	An InvalidLocationException is raised.
An inexistent taxi driver	An InvalidArgumentValueException is raised.
A valid taxiDriver and passengerLocation	Returns the estimated time of the arrival of the taxi to the location.

taxiDriverDroppedRequest(taxiId)	
Input	*Effect*
An invalid taxiId	An InvalidArgumentValueException is raised.
A valid taxiId	If the taxi driver is allowed to drop the request (check RASD conditions), it will return true, notify the passenger that his request has been dropped and update the database accordingly; otherwise it will return false.

sendCurrentLocation(taxiId, location)	
Input	*Effect*
A null location	A NullArgumentException is raised.
A location whose coordinates are invalid	An InvalidLocationException is raised.
An invalid taxiId	An InvalidArgumentValueException is raised.
taxiId is valid, location is inside city	Location is set as the new position of the taxi, the taxi is moved into the new zone, its status is set as available and the modification is written in the database.
taxiId is valid, location is outside city	Location is set as the new position of the taxi, the taxi is moved into the out-of-city queue, its status is set as outide-city and the modification is written in the database.

<table>
<tr><th colspan="2">isLocationInsideCity(location)</th></tr>
<tr><td><i>Input</i></td><td><i>Effect</i></td></tr>
<tr><td>A null parameter</td><td>A NullArgumentException is raised.</td></tr>
<tr><td>A location whose coordinates are invalid</td><td>An InvalidLocationException is raised.</td></tr>
<tr><td>A location outside the city</td><td>Returns false.</td></tr>
<tr><td>A location inside the city</td><td>Returns true</td></tr>
<tr><th colspan="2">getZone(location)</th></tr>
<tr><td><i>Input</i></td><td><i>Effect</i></td></tr>
<tr><td>A null parameter</td><td>A NullArgumentException is raised.</td></tr>
<tr><td>A location whose coordinates are invalid</td><td>An InvalidLocationException is raised.</td></tr>
<tr><td>A location outside the city</td><td>An InvalidLocationException is raised.</td></tr>
<tr><td>A location inside the city</td><td>The id of the zone to which the location belongs.</td></tr>
</table>

Chapter 2

System Administration

2.1 Zone Management, Data Access Utilities

insertZones(Zone[] zones	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An empty array	An InvalidArgumentValueException is raised.
An array containing some null values	A NullArgumentException is raised.
An array of non-null, but invalid zones	An InvalidArgumentValueException is raised.
An array of valid, but already existing zones	An InvalidArgumentValueException is raised.
A non-empty array of valid and not existing zones	The zones are inserted in the database.

insertZoneZone zone	
Input	*Effect*
A NullArgumentException is raised.	
An invalid zone	An InvalidArgumentValueException is raised.
A valid, but already existent zone	An InvalidArgumentValueException is raised.
A valid and not existing zone	The zone is inserted in the database.

asda

updateZones(Zone[] zones)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An empty array	An InvalidArgumentValueException is raised.
An array containing some null values	A NullArgumentException is raised.
An array of non-null, but invalid or inexistent zones	An InvalidArgumentValueException is raised.
A non-empty array of valid, existing zones	The data associated with the specified zones is updated in the database.

getZoneList()	
Input	*Effect*
Nothing	Returns the zones that are stored in

2.2 API Permission Management, Data Access Utilities

verifyPermission(appId, operation)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A non-existing appId	An InvalidSecurityLevelException is raised.
A non-existing operation	An InvalidSecurityLevelException is raised.
A valid appId and operation, but the app hasn't enough privileges to execute the desired operation	Returns false.
A valid appId and operation, and the app has enough privileges to execute the desired operation	Returns true.

grantPermission(appId, operation)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A non-existing operation	An InvalidSecurityLevelException is raised.
A valid appId and operation	Insert a new ¡appId, operation¿ pair in the permission table in the database.

revokePermission(appId, operation)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A non-existing operation	An InvalidSecurityLevelException is raised.
A valid appId and operation, but the ¡appId, operation¿ is not present in the database	An InvalidArgumentValueException is raised.
A valid appId and operation, and the ¡appId, operation¿ is present in the database	Remove the ¡appId, operation¿ pair from the permission table of the database.

2.3 Taxi Driver Management

2.3.1 Data Access Utilities

insertTaxiDrivers(TaxiDriver[] tds)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An empty array	An InvalidArgumentValueException is raised.
An array containing some null values	A NullArgumentException is raised.
An array of valid, but already existing taxi drivers	An InvalidArgumentValueException is raised.
A non-empty array of valid and not existing taxi drivers	The taxi drivers are inserted in the database.

insertTaxiDriver(TaxiDriver td)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An invalid taxi driver	An InvalidArgumentValueException is raised.
A valid, but already existent taxi driver	An InvalidArgumentValueException is raised.
A valid and not existing taxi driver	The taxi driver is inserted in the database.

updateTaxiDriver(TaxiDriver td)	
Input	*Effect*
A NullArgumentException is raised.	
An inexistent taxi driver	An InvalidArgumentValueException is raised.
A taxi driver with some null or empty fields	An InvalidArgumentValueException is raised.
A valid taxi driver	The data associated with the specified taxi driver is updated in the database.

Chapter 3

Account Management

3.1 Passenger Registration, Data Access Utilities

insertPassenger(Passenger p)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A passenger with one or more null fields	An InvalidArgumentValueException is raised.
A passenger with one or more empty fields	An InvalidArgumentValueException is raised.
A passenger with the same mail address as an existing passenger	An InvalidArgumentValueException is raised.
A valid passenger	The passenger data is inserted in the database and a registration confirmation mail is sent to him. The passenger status is set as ”pending confirmation”.

confirmPassenger(PassengerId)	
Input	*Effect*
A non-existing passengerId	An InvalidArgumentValueException is raised.
A passengerId that has already been confirmed	An InvalidArgumentValueException is raised.
A passengerId that hasn’t already been confirmed	The passenger status in the database is set as ”confirmed”.

3.2 Login, Data Access Utilities

checkCredentials(User u, String password)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An inexistent user	An InvalidArgumentValueException is raised.
An empty password	An InvalidArgumentValueException is raised.
A valid user and password combination, which however is not the correct one	Returns an InvalidCredentialError.
A correct and valid user and password combination	Returns a session cookie.

3.3 Password Retrieval, Data Access Utilities

verifyUserSecretCode(User u, String secretCode)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A valid user and secretCode combination, which however is not the correct one	Returns false.
A correct and valid user and secretCode combination	Returns true.
updateUserPassword(User u, String secretCode, String newPassword)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A valid user and secretCode combination, which however is not the correct one	An InvalidSecurityLevelException is raised.
A correct and valid user and secretCode combination, but an correctly formatted password	An InvalidArgumentFormatException is raised.
A correct and valid user and secretCode combination, and a correctly formatted password	Updates the user password in the database.

3.4 Settings Management, Data Access Utilities

Chapter 4

Component-Component Integration

4.1 Remote Services Interface, Taxi Management System

requestTaxi(passengerId, passengerLocation, destination)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A passengerId not correctly formatted	An InvalidArgumentFormatException is raised.
A passengerLocation whose coordinates are invalid	An InvalidLocationException is raised.
A destination whose coordinates are invalid	An InvalidLocationException is raised.
A passengerLocation outside the city	A passengerId not correctly formatted
A valid set of parameters	A new request is created and handled; refer to the RASD for the specific outcomes of this operation.

endRide(taxiId, currentLocation)	
Input	*Effect*
A null location	A NullArgumentException is raised.
A currentLocation whose coordinates are invalid	An InvalidLocationException is raised.
An invalid taxiId	An InvalidArgumentValueException is raised.
A valid taxiId and currentLocation, but taxi is not on a ride	An InvalidOperationException is raised.
A valid taxiId and currentLocation, the taxi is on a ride and currentLocation is inside city	The ride is considered closed and is finalized in the database, the taxi changes its status to available
A valid taxiId and currentLocation, the taxi is on a ride and currentLocation is outside city	The ride is considered closed and is finalized in the database, the taxi changes its status to outside-city

togglePressed(taxiId)	
Input	*Effect*

4.2 Remote Services Interface, Account Management

insertPassenger(Passenger p)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A passenger with one or more null fields	An InvalidArgumentValueException is raised.
A passenger with one or more empty fields	An InvalidArgumentValueException is raised.
A passenger with the same mail address as an existing passenger	An InvalidArgumentValueException is raised.
A valid passenger	The passenger data is inserted in the database and a registration confirmation mail is sent to him. The passenger status is set as ”pending confirmation”.

confirmPassenger(PassengerId)	
Input	*Effect*
A non-existing passengerId	An InvalidArgumentValueException is raised.
A passengerId that has already been confirmed	An InvalidArgumentValueException is raised.
A passengerId that hasn't already been confirmed	The passenger status in the database is set as ”confirmed”.

checkCredentials(User u, String password)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
An inexistent user	An InvalidArgumentValueException is raised.
An empty password	An InvalidArgumentValueException is raised.
A valid user and password combination, which however is not the correct one	Returns an InvalidCredentialError.
A correct and valid user and password combination	Returns a session cookie.

verifyUserSecretCode(User u, String secretCode)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A valid user and secretCode combination, which however is not the correct one	Returns false.
A correct and valid user and secretCode combination	Returns true.

updateUserPassword(User u, String secretCode, String newPassword)	
Input	*Effect*
A null parameter	A NullArgumentException is raised.
A valid user and secretCode combination, which however is not the correct one	An InvalidSecurityLevelException is raised.

4.3 Remote Services Interface, System Administration

asda

<table>
<tr><td colspan="2">checkPassword(adminId, password)</td></tr>
<tr><td><i>Input</i></td><td><i>Effect</i></td></tr>
<tr><td>A null parameter</td><td>A NullArgumentException is raised.</td></tr>
<tr><td>An empty password</td><td>An InvalidArgumentValueException is raised.</td></tr>
<tr><td>An invalid adminId</td><td>An InvalidSecurityLevelException is raised.</td></tr>
<tr><td>A valid adminId and password, but password does not correspond to an authorized one</td><td>Returns false.</td></tr>
<tr><td>A valid adminId and password, and password does correspond to an authorized one</td><td>Returns true.</td></tr>
<tr><td colspan="2">verifyPermission(appId, operation)</td></tr>
<tr><td><i>Input</i></td><td><i>Effect</i></td></tr>
<tr><td>A null parameter</td><td>A NullArgumentException is raised.</td></tr>
<tr><td>A non-existing appId</td><td>An InvalidSecurityLevelException is raised.</td></tr>
<tr><td>A non-existing operation</td><td>An InvalidSecurityLevelException is raised.</td></tr>
<tr><td>A valid appId and operation, but the app hasn't enough privileges to execute the desired operation</td><td>Returns false.</td></tr>
<tr><td>A valid appId and operation, and the app has enough privileges to execute the desired operation</td><td>Returns true.</td></tr>
<tr><td colspan="2">grantPermission(adminId, appId, operation)</td></tr>
<tr><td><i>Input</i></td><td><i>Effect</i></td></tr>
<tr><td>A null parameter</td><td>A NullArgumentException is raised.</td></tr>
<tr><td>A non-existing operation</td><td>An InvalidSecurityLevelException is raised.</td></tr>
<tr><td>An invalid adminId</td><td>An InvalidSecurityLevelException is raised.</td></tr>
<tr><td>A valid appId and operation</td><td>Insert a new ¡appId, operation¿ pair in the permission table in the database.</td></tr>
<tr><td colspan="2">revokePermission(adminId, appId, operation)</td></tr>
<tr><td><i>Input</i></td><td><i>Effect</i></td></tr>
<tr><td>A null parameter</td><td>A NullArgumentException is raised.</td></tr>
<tr><td>A non-existing operation</td><td>An InvalidSecurityLevelException is raised.</td></tr>
<tr><td>An invalid adminId</td><td>An InvalidSecurityLevelException is raised.</td></tr>
<tr><td>A valid appId and operation, but the ¡appId, operation¿ is not present in the database</td><td>An InvalidArgumentValueException is raised.</td></tr>
<tr><td>A valid appId and operation, and the ¡appId, operation¿ is present in the database</td><td>Remove the ¡appId, operation¿ pair from the permission table of the database.</td></tr>
</table>