## COMP 354 Test Document for the project myMoney

## Team PA-PK

April 7, 2018

Table 1: Team

Name	ID Number
Anne-Laure Ehresmann	27858906
Marc-Antoine Dube	40029307
Kadeem Caines	26343600
Abdel Rahman Jawhar	27192142
Keith Dion	40036340
Hrachya Hakobyan	40041555
Andrew-Smith	40034936
Dongyu Chen	27241909
Yauheni Karaniuk	40005680
Renny Xu	40005262
Wei Wang	40041116

Table 2: Revision history

Version	Date	Changes
1.0	15 March 2018	Completed test document

## Contents

1	Introduction	5
2	Test Plan	5
3	Functional Testing	6
	Create User Account	6
	Delete User Account	7
4	Structural Testing	8
	4.1 Unit Test cases	8
	AccountService.addAccount(request, user)	8
	$Account Service. delete Account (account) \ . \ . \ . \ . \ . \ . \ . \ . \ . \ $	10
	$Account Service. delete Accounts For User (user) \ldots \ldots \ldots \ldots \ldots \ldots$	11
	AuthenticationService.authenticate(username, password)	13
	$Remote Account Service.get Account (Get Remote Account Request) \ \dots \dots \dots$	14
	SessionManager.login(username, password)	15
	$Session Manager.logout()  \dots  \dots  \dots  \dots  \dots  \dots  \dots  \dots  \dots  $	16
	$Transaction Service.update Transaction Category (transaction ID, \ category)  .  .  .$	17
	$User Service. create User (User) \ldots \ldots \ldots \ldots \ldots \ldots$	18
	$User Service. delete Bank Account (Account) \ldots \ldots \ldots \ldots \ldots \ldots$	19
	$\label{thm:continuous} User Service. delete User (User) \ . \ . \ . \ . \ . \ . \ . \ . \ . \ $	21
	UserService.updateUser(User)	23
5	Performance Testing	24
6	Acceptance Testing	26
7	Installation Testing	26
8	References	26
$\mathbf{A}$	Description of Input Files	26
В	Description of Output Files	26

# List of Figures

# List of Tables

1	Team	
2	Revision history	

#### 1 Introduction

The aim of this document is to ensure that a coherent and accurate testing strategy is used by each member of the testing team. It seeks to test the implementation of the system described in the Design Plan, testing its validity, robustness, and reliableness as a software, as well as ensuring that the requirements in the Requirements Specification are met. It seeks to do this in a rigorous and justified manner. This document contains an overarching test plan, which seeks to outline each test subsystem, its strategy with regards to testing the associated requirements, and its execution strategy. This document then contains, for each subsystem, a detailed explanation of the set of tests included, and a test case for each individual test. Put together, the test subsystems group into a entire system test.

#### 2 Test Plan

The system test plan has been split into five subsystems:

- Functional Testing: This test subsystem seeks to certify the functionality of the software against the use cases in the Requirements Specification. This category will use black-box testing as its strategy, verifying the usability given different inputs and regardless of the implementation of the sfotware. In its execution, a developer running such a test will typically first identify how the software should perform. Then, he or she verifies the functionality and reliability of the software given valid user behaviour, and then checks for robustness given invalid user behaviour.
- Structural Testing: This test subsystem seeks to verify the structure and code logic of the software. We ensure here that each part of the code functions as expected given both valid and invalid input, and test the behaviour of the system in unexpected states. This will let us confirm the valid flow of our code, and ensure logic faults are caught. For the execution of the test, we will use JUnit to create individual tests for each case. Each test will have an initial setup phase, a test phase, and a teardown phase, to ensure independence of state between each test. A test will also use Mockito, a mocking library, to ensure that the failure of some other, unrelated component of the code does not affect the performance of the tested component in each test.
- **Performance Testing:** This test subsystem seeks to measure the behaviour of the software in extreme states, when under particular workloads or dealing with extremely large datasets. It is useful for testing a number of our non-functional requirements, notably reliability, scalability, and, obviously, performance. In its execution, The tests measure performance statistics given a normal or 'control' environment, then compare it to the performance statistic given a particular dataset or workload.
- Acceptance Testing: This test subsystem seeks to meet the requirements of the use cases in the requirements set in the Requirements Specification. This is also a black-box

testing category, as in functional testing, but unlike the aforementioned, we are instead performing a validation of the system: is our system actually what the user needs? In its execution, the system is given to a user, who will assert whether his needs are met and correspond to what how he or she expects the software to function.

• Installation Testing: This test subsystem seeks to verify that the installation process is both successful and easy in the platforms to be supported. This means ensuring that the choices taken by the user are respected (location of installation, installation just for one user or for whole computer...), verify that all dependent files and libraries are successfully linked and loaded, and valid configurations and connectivity to the database. The execution of this category is simply an activity wherein the installation process is attempted in a particular environment, testing all decisions and options available in the installation.

### 3 Functional Testing

#### Create User Account

Test Case	First name, last name, username and password are mandatory
Description	The user cannot sign up without providing a valid first name, last name,
	a username and a password
	1. Go to 'Sign Up'
Input/Steps	2. Leave all the input fields blank
	3. Click 'Sign up'
Output/Results	• Sign up fails, the account is not created
	• An error window displays all the errors

Test Case	The username must be unique	
Description	The user cannot sign up with an already existing username	
Input/Steps	1. Successfully sign	
	2. Log out	
	3. Go to 'Sign up'	
	4. Fill in all the input fields	
	5. Set the username field to be the username of the user created in the fi	rst stej
	6. Click 'Sign Up'	
Output /Basults	• Sign up failed, the account is not created	
Output/Results	• An error window notifies that the username already exists	

Test Case	The password must be valid
Description	The user cannot sign up with a password not matching the required
	format, as specified in the business rules

Input/Steps	<ol> <li>Go to 'Sign up'</li> <li>Fill in all the input fields</li> <li>Set the password to an alpha-numeric sequence of length less than 4</li> <li>Set the repeat password field to match the password field</li> <li>Click 'Sign Up'</li> </ol>
Output/Results	<ul> <li>Sign up failed, the account is not created</li> <li>An error window notifies that the password is not valid</li> </ul>

Test Case	The user account is successfully created
Description	The user must be able to successfully create an account provided that all
	input information is valid
	1. Go to 'Sign up'
	2. Fill in all the input fields with valid data
Input/Steps	3. Click 'Sign Up'
	4. Moved to the login page: input the username and the passowrd
	5. Click 'Login'
Output /Degulta	• Sign up successful, the account is created
Output/Results	• The user is logged-in to the newly created account

#### Delete User Account

Test Case	Password required
Description	The program asks for the user's password before to delete the account
Input/Steps	<ol> <li>Go to 'Update User Account'</li> <li>Click 'Delete user'</li> </ol>
Output/Results	• A an input window appears asking for the user password

Test Case	The password must be valid	
Description	The user cannot delete the account if the password is invalid	
	1. Go to 'Update User Account'	
Input/Steps	2. Click 'Delete user'	
	3. Enter a wrong password	
Output/Results	• The account is not deleted	
	• An error window must appear notifying the user that the password was	invali

Test Case	The account is successfully deleted
Description	The user account is successfully deleted if the password is correct
Input/Steps	<ol> <li>Go to 'Update User Account'</li> <li>Click 'Delete user'</li> <li>Enter the correct password</li> </ol>

Output/Results	• The account is not deleted	
Output/Results	• An error window must appear notifying the user that the password was	s invali

Test Case	The user account is successfully created	
Description	The user must be able to successfully create an account provided that all	
	input information is valid	
	1. Go to 'Sign up'	
	2. Fill in all the input fields with valid data	
Input/Steps	3. Click 'Sign Up'	
	4. Moved to the login page: input the username and the passowrd	
	5. Click 'Login'	
Output /Pogulta	• Sign up successful, the account is created	
Output/Results	• The user is logged-in to the newly created account	

# 4 Structural Testing

### 4.1 Unit Test cases

 ${\bf Account Service.add Account (request, \, user)}$ 

Table 11: addAccount(request, user)

Tester Name	Hrachya		
Test Date	2/7/18		
Class Name	com.github.comp354project.model.account.AccountService		
Method Name	addAccount(request, user)		
Purpose	This test suite tests the functionality of adding a new bank account		
Use Cases	03		
Test Scenarios			
$testAddAccount\_withIr$	nvalidParameters_shouldThrow		
Input Specification	request accountOwner		
input specification	null null		
Expected Output	ValidationException is thrown		
Expected Output	The number of ValidationErrors is equal to 2		
Actual Output	ValidationException is thrown		
Actual Output	The number of ValidationErrors is equal to 2		
Bug Found	false		
Purpose	Adding an account with invalid request or user should fail		
$test Add Account\_with Nonexistent Remote Account\_should Throw$			
	request accountOwner		

		ID: 1	
	ID: 1	firstName: Hrachya	
		lastName: Hakobyan	
Input Specification		username: admin	
		password: admin	
		email: sample@email.com	
		address: address	
		phone: 111111	
Expected Output		DoestNotExistException is thrown	
Actual Output		DoestNotExistException is thrown	
Bug Found	false		
Purpose		st for adding a nonexistent account should fail	
testAddAccount_withIn	validUser		
	request	accountOwner	
		ID: 1	
		firstName: Hrachya	
		lastName: Hakobyan	
Input Specification	ID: 1	username: admin	
	110.1	password: admin	
		email: sample@email.com	
		address: address	
		phone: 111111	
Expected Output	ValidationException is thrown		
Actual Output	ValidationException is thrown		
Bug Found	false		
Purpose	Adding an account with an invalid owner should throw		
testAddAccount_withE	xistingAc		
	request	accountOwner	
		ID: 1	
		firstName: Hrachya	
		lastName: Hakobyan	
Input Specification	ID: 1	username: admin	
	110.1	password: admin	
		email: sample@email.com	
		address: address	
		phone: 111111	
Expected Output	AccountExistsException is thrown		
Actual Output	AccountExistsException is thrown		
Bug Found	false		
Purpose		an already existing account should throw	
testAddAccount_withV		$\operatorname{nt\_shouldReturnValidAccount}$	
	request	accountOwner expectedAccount	

Input Specification	ID: 1	ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111	ID: 1 user: accountOwner bankName: TD type: Checking balance: 15823.12	
Expected Output	The account is fetched and persisted in the database The persisted account is equal to the 'expectedAccount' object The returned account is equal to the 'expectedAccount' object			
Actual Output	The account is fetched and persisted in the database  The persisted account is equal to the 'expectedAccount' object  The returned account is equal to the 'expectedAccount' object			
Bug Found	false			
Purpose	Adding	Adding a valid account with a valid owner must succeed		

### ${\bf Account Service. delete Account (account)}$

Table 12: deleteAccount(account)

Tester Name	Anne-Laure		
Test Date	3/5/18		
Class Name	com.github.comp354project.model.account.AccountService		
Method Name	deleteAccount(account)		
Purpose	This test suite tests the functionality of removing a user's bank account		
Use Cases	04		
Test Scenarios			
testDeleteAccount_with	NullAccount_shouldThrow		
Input Specification	account		
input specification	null		
Expected Output	ValidationException is thrown		
Actual Output	ValidationException is thrown		
Bug Found	false		
Purpose	Delete a null account should fail		
$testDeleteAccount\_with$	AccountWithNullID_shouldThrow		
	account		
	ID: null		
Input Specification	user: null		
	bankName:		
	type:		
	balance: 0		

Expected Output	ValidationException is thrown		
Actual Output	ValidationException is thrown		
Bug Found	false		
Purpose	Deleting an account with null ID should fail		
testDeleteAccount_with	NonExistentAccount_shouldThrow		
	account		
	ID: 1		
Input Specification	user: accountOwner		
input specification	bankName: TD		
	type: Checking		
	balance: 15823.12		
Expected Output	ValidationException is thrown		
Actual Output	ValidationException is thrown		
Bug Found	false		
Purpose	Deleting an nonexistent account should fail		
testDeleteAccount_with	ValidAccount_shouldSucceed		
	account		
	ID: 1		
Input Specification	user: accountOwner		
input specification	bankName: TD		
	type: Checking		
	balance: 15823.12		
Expected Output	The account is deleted from the database		
Actual Output	The account is deleted from the database		
Bug Found	false		
Purpose	Deleting an existing account should succeed		
testDeleteAccount_with	ValidAccount_shouldDeleteAllAssociatedTransactionsAndAccount		
	account		
	ID: 1		
T	user: accountOwner		
Input Specification	bankName: TD		
	type: Checking		
	balance: 15823.12		
	transactions: [object Object]		
Expected Output	The account is deleted from the database		
Actual Output	All the associated transactions are deleted from the database		
	The account is deleted from the database		
_	All the associated transactions are deleted from the database		
Bug Found	false		
Purpose	Deleting an existing account should delete all associated transactions		

 ${\bf Account Service. delete Accounts For User (user)}$ 

 ${\bf Table~13:~delete Accounts For User (user)}$ 

Test Date 4/2/18 Class Name com_github.comp354project.model.account.AccountService Method Name deleteAccountsForUser(user)  Purpose This test suite tests the functionality of removing a user's bank accounts and associated transactions  Use Cases 04  Test Scenarios  testDeleteAccountsForUser_withNullUserID_shouldThrow  Input Specification  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false  Purpose Deleting accounts with null user ID should fail  testDeleteAccountsForUser_withNonexistentUser_shouldSucceed  Input Specification  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user  ID: 1  firstName: Hrachya lastName: Hakobyan username: admin password: admin p				
Class Name	Tester Name	Hrachya		
Method Name   delete-AccountsForUser(user)				
This test suite tests the functionality of removing a user's bank accounts and associated transactions	Class Name	com.github.comp354project.model.account.AccountService		
and associated transactions   194	Method Name	deleteAccountsForUser(user)		
Test Scenarios   User Localization   User ID   Should Throw   User ID   Input Specification   Input Specificatio	Purpose	· · · · · · · · · · · · · · · · · · ·		
Test Scenarios testDeleteAccountsForUser_withNullUserID_shouldThrow  Input Specification Input Specification Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found False Purpose Deleting accounts with null user ID should fail testDeleteAccountsForUser_withNonexistentUser_shouldSucceed  Input Specification Input Specification Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found False Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: a		and associated transactions		
Input Specification  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false  Purpose Deleting accounts with null user ID should fail testDeleteAccountsForUser_withNonexistentUser.shouldSucceed  Input Specification  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user  ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions.shouldDeleteAccountAndTransactions		04		
Input Specification				
mull	testDeleteAccountsForU			
Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false  Purpose Deleting accounts with null user ID should fail testDeleteAccountsForUser_withNonexistentUser_shouldSucceed  Input Specification 1  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user ID: 1 firstName: Hrachya lastName: Hakobyan username: admin passit: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Actual Output Specification false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Input Specification	userID		
Actual Output         ValidationException is thrown           Bug Found         false           Purpose         Deleting accounts with null user ID should fail           testDeleteAccountsForUser_withNonexistentUser_shouldSucceed           Input Specification         userID           Expected Output         No accounts are deleted. The system state is not changed.           Actual Output         No accounts are deleted. The system state is not changed.           Bug Found         false           Purpose         Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.           testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed           user         ID: 1           firstName: Hrachya         lastName: Hakobyan           lastName: Hakobyan         username: admin           password: admin         email: sample@email.com           address: address         phone: 111111           Expected Output         No accounts are deleted. The system state is not changed.           Actual Output         No accounts are deleted. The system state is not changed.           Bug Found         false           Purpose         Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system	Input Specification	null		
Bug Found   false   Deleting accounts with null user ID should fail	Expected Output	ValidationException is thrown		
Deleting accounts with null user ID should fail testDeleteAccountsForUser_withNonexistentUser_shouldSucceed   userID	Actual Output	ValidationException is thrown		
testDeleteAccountsForUser_withNonexistentUser_shouldSucceed  Input Specification  Expected Output  No accounts are deleted. The system state is not changed.  Actual Output  No accounts are deleted. The system state is not changed.  Bug Found  Purpose  Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user  ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output  No accounts are deleted. The system state is not changed.  Actual Output  No accounts are deleted. The system state is not changed.  Bug Found  False  Purpose  Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Bug Found	false		
Input Specification  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found Found Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found Found False  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	_			
Expected Output	testDeleteAccountsForU	User_withNonexistentUser_shouldSucceed		
Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user  ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Input Specification	userID		
No accounts are deleted. The system state is not changed.	input specification	1		
Bug Found false  Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Expected Output	No accounts are deleted. The system state is not changed.		
Purpose Deleting a nonexistent user's accounts should succeed and should not inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed  user ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Actual Output			
inflict any changes to the system.  testDeleteAccountsForUser_withValidUserAndEmptyAccounts_shouldSucceed    user   ID: 1	Bug Found	false		
Input Specification Input	Purpose	Deleting a nonexistent user's accounts should succeed and should not		
Input Specification Input		inflict any changes to the system.		
ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	testDeleteAccountsForU	Jser_withValidUserAndEmptyAccounts_shouldSucceed		
Input Specification  Input Spe		user		
Input Specification  Input Spe		ID: 1		
Input Specification  username: admin password: admin email: sample@email.com address: address phone: 111111  Expected Output  No accounts are deleted. The system state is not changed.  Actual Output  No accounts are deleted. The system state is not changed.  Bug Found  false  Purpose  Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions		firstName: Hrachya		
password: admin email: sample@email.com address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions		lastName: Hakobyan		
email: sample@email.com     address: address     phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Input Specification	username: admin		
address: address phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions		password: admin		
phone: 111111  Expected Output No accounts are deleted. The system state is not changed.  Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions		email: sample@email.com		
Expected OutputNo accounts are deleted. The system state is not changed.Actual OutputNo accounts are deleted. The system state is not changed.Bug FoundfalsePurposeDeleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the systemtestDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions		address: address		
Actual Output No accounts are deleted. The system state is not changed.  Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions		phone: 111111		
Bug Found false  Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system  testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Expected Output	No accounts are deleted. The system state is not changed.		
Purpose Deleting the accounts of a user who does not have any accounts should succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Actual Output	No accounts are deleted. The system state is not changed.		
succeed and inflict no changes to the system testDeleteAccountsForUser_withAssociatedTransactions_shouldDeleteAccountAndTransactions	Bug Found	false		
$test Delete Accounts For User\_with Associated Transactions\_should Delete Account And Transactions$	Purpose	Deleting the accounts of a user who does not have any accounts should		
		· ·		
ugon account	testDeleteAccountsForU	${\tt Jser\_with Associated Transactions\_should Delete Account And Transactions}$		
user account		user account		

	ID: 1			
	firstName: Hrachya	ID: 1		
	lastName: Hakobyan	user: user		
Input Specification	username: admin	bankName: TD		
	password: admin	type: Checking		
	email: sample@email.com	balance: 15823.12		
	address: address	transactions: [object Object]		
	phone: 111111			
Expected Output	The accounts are deleted from the database			
Expected Output	All the associated transactions are deleted from the database			
Actual Output	The accounts are deleted from the database			
Actual Output	All the associated transactions are deleted from the database			
Bug Found	false			
Purpose	Deleting the accounts of the user should also delete all the associated			
	transactions.			

# ${\bf Authentication Service. authenticate (username,\ password)}$

Table 14: authenticate(username, password)

Tester Name	Hrachya		
Test Date	2/3/18		
Class Name	com.github.comp354projec	t.model.auth.AuthenticationService	
Method Name	authenticate (username, par	ssword)	
Purpose	This test suite tests the au	thentication of the user	
Use Cases	01		
Test Scenarios			
testAuthenticate_withIn	nvalidUsernameOrPassword	shouldThrow	
Input Specification	username	password	
input specification	null	null	
Expected Output	ValidationException is thrown		
Expected Output	The number of ValidationErrors is equal to 2		
Actual Output	ValidationException is thrown		
Actual Output	The number of ValidationErrors is equal to 2		
Bug Found	false		
Purpose	A user with invalid credentials should not be able to authenticate		
testAuthenticate_withNonexistentUsername_shouldThrow			
Input Specification	username	password	
input Specification	username	password	
Expected Output	ValidationException is thrown		
Actual Output	ValidationException is thrown		
Bug Found	false		

Purpose	A user with a nonexistent username should not be able to authenticate		
testAuthenticate_withIn	$test Authenticate\_with Incorrect Password\_should Throw$		
	testUser	username	password
	ID: 1		
	firstName: Hrachya		
	lastName: Hakobyan		
Input Specification	username: admin	admin	INCORRECT_PASSWORD
	password: admin	adiiiii	INCORRECT I ASSWORD
	email: sample@email.com		
	address: address		
	phone: 111111		
Expected Output	ValidationException is three		
Actual Output	UserLoggedInException is thrown		
Bug Found	false		
Purpose	Authentication with a valid username but an incorrect password should		
	fail		
testAuthenticate_withC	$correctCredentials\_shouldRet$	urnUser	
	testUser	username	password
	ID: 1		
	firstName: Hrachya		admin
	lastName: Hakobyan		
Input Specification	username: admin	admin	
	password: admin	adiiiii	adiiiii
	email: sample@email.com		
	address: address		
	phone: 111111		
Expected Output	The authentication is successful and the authenticated user is returned		ne authenticated user is returned
Expected Output	The authenticated user is $\epsilon$	_	•
Actual Output	The authentication is successful and the authenticated user is returned		
Actual Output	The authenticated user is equal to the 'testUser' object		
Bug Found	false		
Purpose	Authentication with a valid username but an incorrect password should		
	fail		

#### Remote Account Service. get Account (Get Remote Account Request)

 $Table\ 15:\ getAccount(GetRemoteAccountRequest)$ 

Tester Name	Abed Jawhar
Test Date	3/13/18
Class Name	com. github. comp 354 project. model. account. remote. Remote Account Service and the compact of the compact
Method Name	getAccount(GetRemoteAccountRequest)

Purpose	This test suite tests fetching an account in the 'API' that connects to other systems			
Use Cases	03			
Test Scenarios				
$testGetAccount\_withNullRequest\_shouldThrow$				
Input Specification	request null			
Expected Output	ValidationExceptio	ValidationException is thrown		
Actual Output	ValidationExceptio	ValidationException is thrown		
Bug Found	false			
Purpose	A null account can	't be fetched		
testGetAccount_withIn	validRequest_should	Throw		
Input Specification	request			
Expected Output	ValidationException is thrown			
Actual Output	ValidationException is thrown			
Bug Found	false			
Purpose	An empty account can't be fetched			
$testGetAccount\_withExistingAccount\_shouldReturnValidAccount$				
	expectedAccount	expectedAccountTransactions	request	
Input Specification	ID: 1 bankName: TD type: Checking balance: 15823.12  ID: 1 account: testRem date: 1517091082 amount: 52.2 type: Transfer sourceID: null destinationID: 2			
Expected Output	The fetched account should be the same as the 'expectedAccount' The number of transactions fetched should be 1			
Actual Output	The fetched account should be the same as the 'expectedAccount' The number of transactions fetched should be 1			
Bug Found	false			
Purpose	A valid account should be fetched			

### $Session Manager.login (username,\ password)$

Table 16: login(username, password)

Tester Name	Hrachya
Test Date	2/7/18
Class Name	com.github.comp354project.model.auth.SessionManager
Method Name	login(username, password)

Purpose	This test suite tests the login of a user				
Use Cases	02				
Test Scenarios					
testLogin_withInvalidCr	redentials_shouldThrow				
Input Specification	username password				
Expected Output	ValidationException is three	own			
Actual Output	ValidationException is three	own			
Bug Found	false				
Purpose	A user with invalid credent	ials should	not be able	to login	
testLogin_withValidCree	dentials_shouldReturnUser				
	testUser	username	password	loggedIn	authenticateInvoc
Input Specification  Expected Output  Actual Output	ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111 The method authenticate i The user is logged in The logged in user is equal The user is logged in	to 'testUse s invoked 1	r' object time	true	1
Dug Found	The logged in user is equal	to 'testUse	r' object		
Bug Found	false				
Purpose testLogin_withLoggedIn	A user with valid credentials should be able to login				
testrogiii_witiiroggediii	username	password			
Input Specification		•			
Expected Output	UserLoggedInException is thrown				
Actual Output	UserLoggedInException is thrown				
Bug Found	false				
Purpose	A user that is already logg	ed in should	not be able	e to login a	gain

#### ${\bf Session Manager.logout()}$

Table 17: logout()

Tester Name	Hrachya
Test Date	2/7/18

Class Name	com.github	com.github.comp354project.model.auth.SessionManager			
Method Name	logout()	logout()			
Purpose	This test s	This test suite tests the function to logout			
Use Cases	02				
Test Scenarios					
testLogin_withInvalidCredentials_shouldThrow					
Input Specification	username	password	isLoggedIn	currentUser	
input specification			false	null	
Expected Output	After logou	it, the login	status shoul	d be false	
Expected Output	After logout, the current user should be null				
Actual Output  After logout, the login status should be false		d be false			
Actual Output	After logout, the current user should be null				
Bug Found	false				
Purpose	A user sho	uld be com	pletely logged	out of the application	

#### $Transaction Service.update Transaction Category (transaction ID, \ category)$

Table 18: updateTransactionCategory(transactionID, category)

Tester Name	Hrachya		
Test Date	3/4/18		
Class Name	com.github.comp354project.model.account.TransactionService		
Method Name	updateTransactionCategory(transactionID, category)		
Purpose	This test suite	tests the functionality of updating the category of a trans-	
	action		
Use Cases	08		
Test Scenarios			
testUpdateCategory_wi	thNullTransaction	onID_shouldThrow	
Input Specification	transactionID	category	
input specification	null	Leisure	
Expected Output	ValidationException is thrown		
Actual Output	ValidationException is thrown		
Bug Found	false		
Purpose	Updating a null transaction ID should fail		
testUpdateCategory_wi	$test Update Category\_with Nonexistent Transaction\_should Throw$		
Input Specification	transactionID	category	
input specification	111111	Leisure	
Expected Output	ValidationException is thrown		
Actual Output	ValidationException is thrown		
Bug Found	false		
Purpose	Updating a nonexistent transaction should fail		

testUpdateCategory_wi	testUpdateCategory_withNullCategory_shouldSucceed			
Input Specification	transactionID	category		
input specification	10	null		
Expected Output	The 'category' of the transaction with the specified ID is set to null			
Actual Output	The 'category' of the transaction with the specified ID is set to null			
Bug Found	false			
Purpose	Updating the category of a valid transaction to null must succeed			
testUpdateCategory_wi	- 0	ry_shouldSucceed		
Input Specification	transactionID	category		
input Specification	10			
Expected Output	The 'category'	of the transaction with the specified ID is set to "		
Actual Output	The 'category'	of the transaction with the specified ID is set to "		
Bug Found	false	false		
Purpose	Updating the category of a valid transaction to an empty string must			
	succeed			
$test Up date Category\_with Valid Category\_should Succeed$				
Input Specification	transactionID	category		
	10	Leisure		
Expected Output	The 'category' of the transaction with the specified ID is set to 'Leisure'			
Actual Output	The 'category' of the transaction with the specified ID is set to 'Leisure'			
Bug Found	false			
Purpose	Updating the category of a valid transaction must succeed			
$test Up date Category\_with Invalid Category\_should Throw$				
Input Specification	transactionID	category		
	10	AAAAAAAAAAAAA		
Expected Output	ValidationException is thrown			
Actual Output	ValidationExce	eption is thrown		
Bug Found	false			
Purpose	Updating the	category of a valid transaction to a an invalid value as		
	determined by the business rules must fail			

## ${\bf User Service.create User (User)}$

Table 19: createUser(User)

Tester Name	Hrachya
Test Date	1/31/18
Class Name	com.github.comp354project.model.user.UserService
Method Name	createUser(User)
Purpose	This test suite tests the creation of a user
Use Cases	01
Test Scenarios	

Imput Specification   User   Inull   Imput Specification   Validation   Validatio	createUser_withNullUse	er shouldThrow			
mull					
Actual Output   SudiationException is thrown   Furpose   No null user can be saved in the database	Input Specification				
Actual Output   SudiationException is thrown   Furpose   No null user can be saved in the database	Expected Output	ValidationException is thrown			
Purpose   No null user can be saved in the database	Actual Output				
Input Specification  Expected Output  Actual Output  Actual Output  Input Specification  Expected Output  Actual Output  Actual Output  Input Specification  Actual Output  Actual Output  Input Specification  Input Speci	Bug Found	false			
Input Specification	Purpose	No null user can be saved in the database			
Expected Output  Actual Output  Actual Output  Actual Output  Bug Found Input Specification  Actual Output  Actual Output  Actual Output  Input Specification  Input Specificatio	testCreateUser_withInv	alidUser_shouldThrow			
Expected Output  ValidationException is thrown 4 exceptions are thrown because missing fields: username, password, firstname ValidationException is thrown 4 exceptions are thrown because missing fields: username, password, firstname ValidationException is thrown 4 exceptions are thrown because missing fields: username, password, firstname Insurable ValidationException is thrown Actual Output  Input Specification  ValidationException is thrown	T4 C	user errors			
Actual Output 4 exceptions are thrown because missing fields: username, password, firstname ValidationException is thrown 4 exceptions are thrown because missing fields: username, password, firstname false Purpose No empty value user can be saved in the database testCreateUser_withValidUser_shouldReturnUser  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the input due to the saved user is the same as the input	input Specification	4			
Actual Output  Actual Output  ValidationException is thrown 4 exceptions are thrown because missing fields: username, password, firstname false  Purpose No empty value user can be saved in the database testCreateUser_withValidUser_shouldReturnUser  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME User ID was autogenerated upon save The saved user is the same as the inputted user  Houtput  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save the saved user is the same as the inputted user  User ID was autogenerated upon save the saved user is the same as the inputted user  User ID was autogenerated upon save the saved user is the same as the inputted user  User ID was autogenerated upon save the saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user	E	ValidationException is thrown			
Actual Output 4 exceptions are thrown because missing fields: username, password, firstname false  Purpose No empty value user can be saved in the database testCreateUser_withValidUser_shouldReturnUser  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Last ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  Bug Found false  Purpose A valid user should be inserted in the database testCreateUser_withExistingUsername.shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false	Expected Output	4 exceptions are thrown because missing fields: username, password, first	name,		
Secretions are thrown because missing heids: username, password, histoanee   Purpose	A -t1 Ott	ValidationException is thrown			
Bug Found   false	Actual Output	4 exceptions are thrown because missing fields: username, password, first	name,		
testCreateUser_withValidUser_shouldReturnUser  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Last ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  Bug Found false  Purpose A valid user should be inserted in the database  testCreateUser_withExistingUsername_shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  false	Bug Found				
Input Specification  Input Spe	Purpose	No empty value user can be saved in the database			
Input Specification  Farse and a substraint of first Name: USERNAME  password: PASSWORD  firstName: FIRSTNAME  lastName: LASTNAME  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  Bug Found  false  Purpose  A valid user should be inserted in the database  testCreateUser_withExistingUsername_shouldThrow  user  user username: USERNAME  password: PASSWORD  firstName: FIRSTNAME  lastName: LASTNAME  Expected Output  ValidationException is thrown  Actual Output  ValidationException is thrown  false	$testCreateUser\_withVal$	idUser_shouldReturnUser			
Input Specificationpassword: PASSWORD firstName: FIRSTNAME lastName: LASTNAMEExpected OutputUser ID was autogenerated upon save The saved user is the same as the inputted userActual OutputUser ID was autogenerated upon save The saved user is the same as the inputted userBug FoundfalsePurposeA valid user should be inserted in the databasetestCreateUser_withExistingUsername_shouldThrowInput Specificationuser username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAMEExpected OutputValidationException is thrownActual OutputValidationException is thrownBug Foundfalse		user			
firstName: FIRSTNAME lastName: LASTNAME  User ID was autogenerated upon save The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  Bug Found  False  Purpose A valid user should be inserted in the database  testCreateUser_withExistingUsername_shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found  false		username: USERNAME			
LastName: LASTNAME     User ID was autogenerated upon save     The saved user is the same as the inputted user     Actual Output     User ID was autogenerated upon save     The saved user is the same as the inputted user     Bug Found     False     Purpose   A valid user should be inserted in the database     testCreateUser_withExistingUsername_shouldThrow     User     user     user     username: USERNAME     password: PASSWORD     firstName: FIRSTNAME     lastName: LASTNAME     Expected Output     ValidationException is thrown     Actual Output     ValidationException is thrown     Bug Found     False     Contact     Co	Input Specification	1 *			
Expected OutputUser ID was autogenerated upon save The saved user is the same as the inputted userActual OutputUser ID was autogenerated upon save The saved user is the same as the inputted userBug FoundfalsePurposeA valid user should be inserted in the databasetestCreateUser_withExistingUsername_shouldThrowuseruserusername: USERNAME password: PASSWORD firstName: FIRSTNAMElastName: LASTNAMEExpected OutputValidationException is thrownActual OutputValidationException is thrownBug Foundfalse		firstName: FIRSTNAME			
The saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  Bug Found false  Purpose A valid user should be inserted in the database  testCreateUser_withExistingUsername_shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  LastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found  false		lastName: LASTNAME			
Actual Output  User ID was autogenerated upon save The saved user is the same as the inputted user  Bug Found  False  A valid user should be inserted in the database  testCreateUser_withExistingUsername_shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Expected Output  ValidationException is thrown  Actual Output  ValidationException is thrown  Bug Found  Fine saved user is the same as the inputted user  User ID was autogenerated upon save The saved user is the same as the inputted user  Input Specification  false	Expected Output	User ID was autogenerated upon save			
The saved user is the same as the inputted user  Bug Found  False  Purpose  A valid user should be inserted in the database  testCreateUser_withExistingUsername_shouldThrow  user  user  username: USERNAME  password: PASSWORD  firstName: FIRSTNAME  lastName: LASTNAME  Expected Output  ValidationException is thrown  Actual Output  ValidationException is thrown  Bug Found  false	Expected Output	The saved user is the same as the inputted user			
Bug Found false  Purpose A valid user should be inserted in the database testCreateUser_withExistingUsername_shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false	Actual Output	User ID was autogenerated upon save			
Purpose A valid user should be inserted in the database  testCreateUser_withExistingUsername_shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false	Actual Output	The saved user is the same as the inputted user			
testCreateUser_withExistingUsername_shouldThrow  user username: USERNAME password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  LastName: LASTNAME  ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found  false	Bug Found				
Input Specification  Input Specification  Input Specification  password: PASSWORD  firstName: FIRSTNAME  lastName: LASTNAME  LASTNAME  ValidationException is thrown  Actual Output  ValidationException is thrown  Bug Found  Input Specification  password: PASSWORD  firstName: FIRSTNAME  lastName: LASTNAME  ValidationException is thrown	_	A valid user should be inserted in the database			
Input Specification password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false	testCreateUser_withExi	istingUsername_shouldThrow			
Input Specification password: PASSWORD firstName: FIRSTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false					
firstName: FIRSTNAME lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false		username: USERNAME			
lastName: LASTNAME  Expected Output ValidationException is thrown  Actual Output ValidationException is thrown  Bug Found false	Input Specification	password: PASSWORD			
Expected Output ValidationException is thrown Actual Output ValidationException is thrown Bug Found false					
Actual Output ValidationException is thrown Bug Found false		lastName: LASTNAME			
Bug Found false		_			
-	_	1			
Purpose A user cannot be created if the username is already taken		false			
	Purpose	A user cannot be created if the username is already taken			

User Service. delete Bank Account (Account)

Table 20: deleteBankAccount(Account)

Tester Name	Anne-Laure				
Test Date	3/7/18				
Class Name	com.github.comp354project.model.user.UserService				
Method Name	deleteBankAccount(Account)				
Purpose	This test suite tests the deletion of a bank account				
Use Cases	04				
Test Scenarios					
testDeleteBankAccount	$testDeleteBankAccount\_withNullAccount\_ShouldThrow$				
Input Specification	account null				
Expected Output	ValidationException is three	own			
Actual Output	ValidationException is three ValidationException is three				
Bug Found	false	JW 11			
Purpose	No null account can be pas	esed to the function			
	_withoutBeingLoggedIn_Sho				
0CS0DCIC0CDank/ICCOunt	account	testUser			
Input Specification	ID: 1 user: testUser bankName: TD type: Checking balance: 15823.12	ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email. address: address phone: 111111	com		
Expected Output	AuthenticationException is	sthrown			
Actual Output	AuthenticationException is	s thrown			
Bug Found	false				
Purpose	A user that is not authenti	cated cannot delete his	accounts		
test Delete Bank Account	$\_$ without $P$ roper $A$ uthorisatio	$n\_ShouldThrow$			
	testUser	user2	testAccount		
Input Specification	ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111	username: username password: password firstName: firstname lastName: lastname ID: 999	ID: 1 user: testUser bankName: TD type: Checking balance: 15823.12		
Expected Output	AuthorisationException is thrown				
Actual Output	AuthorisationException is	thrown			

Bug Found	false			
Purpose	A user cannot modify the accounts of another user			
$test Delete Bank Account\_With Proper Authorisation\_Should Succeed \\$				
	testUser	testAccount	invocationCount	
Input Specification	ID: 1 firstName: Hrachya lastName: Hakobyan username: admin password: admin email: sample@email.com address: address phone: 111111	ID: 1 user: testUser bankName: TD type: Checking balance: 15823.12	1	
Expected Output	Execution of the deletion of the account once			
Actual Output	Execution of the deletion of the account once			
Bug Found	false			
Purpose	An authenticated user should succeed in deleting his own bank accounts			

## ${\bf User Service. delete User (User)}$

Table 21: deleteUser(User)

Tester Name	Abed Jawhar	
Test Date	3/13/18	
Class Name	com.github.comp354project.model.user.UserService	
Method Name	deleteUser(User)	
Purpose	This test suite tests the deletion of a user	
Use Cases	02	
Test Scenarios		
testDeleteUser_withNul	llUser_shouldThrow	
Input Specification	user	
Input Specification	null	
Expected Output	ValidationException is thrown	
Actual Output	ValidationException is thrown	
Bug Found	false	
Purpose	A null user can't be deleted	
$test Delete User\_with Nonexistant User\_should Throw$		
	testUser	

	ID: 1 firstName: Hrachya					
T (Consideration	lastName: Hakobyan					
Input Specification	username: admin					
	password: admin					
	email: sample@email.com address: address					
	phone: 111111					
Expected Output	ValidationException is thro	วนฑ				
Actual Output	ValidationException is three					
Bug Found	false					
Purpose	A user that does not exist of	can't be deleted				
	istingtUser_shouldSucceed					
	testUser	returnSize				
	ID: 1					
	firstName: Hrachya	l		ļ		
	lastName: Hakobyan	I		ļ		
Input Specification	username: admin	0				
	password: admin					
	email: sample@email.com					
	address: address					
	phone: 111111	<u> </u>				
Expected Output	The number of users with I					
Actual Output		The number of users with ID 1 is 0				
Bug Found	false					
Purpose	A valid user should be deleted					
testDeleteUser_witiiLxi	stingtUser_shouldDeleteAsso	1	1 Cigo	1 1 d A s s compt I parco		
	testUser ID: 1	testAccount	returnSize	deleteAccountInvo		
Input Specification	firstName: Hrachya lastName: Hakobyan username: admin	ID: 1 user: testUser		1		
	password: admin email: sample@email.com address: address phone: 111111	bankName: TD type: Checking balance: 15823.12				
Expected Output	The number of users with I					
Expected Output	Delete account should be in					
	The number of users with ID 1 is 0 Delete account should be invocated 1 time					
Actual Output	Delete account should be in					
Actual Output  Bug Found  Purpose		nvocated 1 time				

### User Service.update User (User)

Table 22: updateUser(User)

Tester Name	Abed Jawhar						
Test Date	3/13/18						
Class Name	com.github.comp354project.model.user.UserService						
Method Name	updateUser(User)						
Purpose	- ,	This test suite tests the update of a user					
Use Cases	02						
Test Scenarios							
testUpdateUser_withNu	ullUser_shouldThrow						
Innut Chasification	user						
Input Specification	null						
Expected Output	ValidationException is three	own					
Actual Output	ValidationException is three	own					
Bug Found	false						
Purpose	A null user can't be update						
testUpdateUser_withNo	$onexistenttUser\_shouldThrow$	V					
	testUser						
	ID: 1						
	firstName: Hrachya						
	lastName: Hakobyan						
Input Specification	username: admin						
	password: admin						
	email: sample@email.com						
	address: address						
	phone: 111111						
Expected Output	ValidationException is three	own					
Actual Output	ValidationException is three	own					
Bug Found	false						
Purpose	A user that does not exist	can't be upo	dated				
testUpdateUser_withVa							
	testUser	firstName	lastName	password			
	ID: 1						
	firstName: Hrachya						
	lastName: Hakobyan						
Input Specification	username: admin	Abed	iowhor	admin2			
	password: admin	Abca	jawhar	adminz			
	email: sample@email.com						
	address: address						
	phone: 111111						

	The firstName is updated to 'Abed'
Expected Output	The lastName is updated to 'jawhar'
	The password is updated to 'admin2'
	The firstName is updated to 'Abed'
Actual Output	The lastName is updated to 'jawhar'
	The password is updated to 'admin2'
Bug Found	false
Purpose	A valid user should be updated

# 5 Performance Testing

Tester Name	Anne-Laure
Test Date	5/4/18
Purpose	This test suite contains the series of tests performed with yourKit Java
	Profiler.
System specification	
OS	GNU/Linux Fedora 27 x64, version 4.13.9-300
RAM	4GB
Graphics Card	Intel Celeron 3205U @ 1.50GHz x 2
OpenJDK version	1.8.0_144
Profiler	YourKit Java Profiler 2017.02-b75

control stress test: local database with 5 accounts and 5 transactions		
CPU usage chart	CPU usage chart 1	
Thread Count chart	Thread Count chart 1	
	44m 38s: application launched	
	44m 40s: login menu loaded	
	44m 45s: logged in	
Events	44m 47s: sorted accounts by type	
Evenus	44m 53s: removed bank account 2	
	44m 58s: added back account 2	
	45m 02s: viewed all transactions	
	45m 15s: shut down application	

Memory						
	Heap-Memory		Non-Heap Memory			
Used	Allocated	Limit	imit Used Allocated L			
73MB	140MB	910MB	65 MB	65 MB	65 MB	

CPU							
Classes	Threads						
8,415	Currently live	Peak Total crea					
	11	5	31	53			

stress test: local datab	stress test: local database with 11,000 accounts and 5 transactions		
CPU usage chart	CPU usage chart 2		
Thread Count	Thread Count chart 2		
chart	Thread Count chart 2		
	2m 43s: application launched		
	2m 46s: login menu loaded		
	2m 51s: logged in		
	2m 52s: sorted accounts by type		
Events	2m 53s: reversed sort of accounts by type		
Evenus	2m 54s: sorted accounts by ID		
	2m 55s: clicked on an account to view account details		
	2m 57s: returned to account list		
	3m 02s: viewed all transactions		
	3m 15s: shut down application		

Memory							
Heap-Memory				Non-Heap Memory			
Used	Allocated Limit Used Allocate					Limit	
123MB	203MB	910MB	67 M	В	67 MB	67 MB	
	CPU						
Classes			Thr	eads			
8,454	Currently	live Curre	ently live	Peal	k	Total created	
0,494		da	daemons				
	14		6	31		48	

stress test: local database with 11,000 accounts and 10,000 transactions			
CPU usage chart 2			
Thread Count	Thread Count short 2		
chart Thread Count chart 2			

	34m 11s: application launched
	34m 14s: login menu loaded
	34m 18s: logged in
	34m 20s: sorted accounts by type
	34m 25s: reversed sort of accounts by type
Events	34m 27s: sorted accounts by ID
	34m 29s: clicked on an account to view account details
	34m 30s: returned to account list
	34m 34s: viewed all transactions
	34m 37s: returned to account list
	35m 1s: shut down application

Memory						
Heap-Memory				Non-Heap Memory		
Used	Allocated	Allocated Limit Use			Allocated	Limit
158MB	207MB	910MB	70 M	В	70 MB	70 MB
	CPU					
Classes			Thre	eads		
8,456	Currently	live Currer	ntly live	Pe	ak	Total created
daemons						
	14 6		6	3	1	88

# 6 Acceptance Testing

# 7 Installation Testing

#### 8 References

# A Description of Input Files

Describe/include test data from input files.

# **B** Description of Output Files

Describe/include test expected output that are output files.