Vaishnavi Manthena and Sanjana Pendharkar (Group 19)
Dr. Chang

Project 3: Test Design Document

Test Case #	Purpose of test case	Input data	Expected output
1 - Open Account	Tests that the firstName and lastName methods work properly in the context of opening an account, which means that they only open an account when given valid String fields (cannot be only delimiters or empty strings) ● Case 1: Only entering first name ● Case 2: Only entering last name ● Case 3: Sending nothing into both tabs ● Case 4: Sending a space (single delimiter) into both tabs ● Case 5: Entering valid string values for both first name and last name	● Case 1 First Name = “Richard” Last Name = “” ● Case 2 First Name = “” Last Name = “Scanlan” ● Case 3 First Name = “” Last Name = “” ● Case 4 First Name = “ ” Last Name = “ ” ● Case 5 First Name = “Richard” Last Name = “Scanlan”	● Case 1: Returns catch block, as this is an exception. “Please make sure you enter both a proper first name and last name.” ● Case 2: Returns catch block, as this is an exception. “Please make sure you enter both a proper first name and last name.” ● Case 3: Returns catch block, as this is an exception. “Please make sure you enter both a proper first name and last name.” ● Case 4: Returns catch statement for when length of firstName and lastName is equal to 0. “Make sure you entered a value for both first name and last name.” ● Case 5: Creates a profile with the firstName = Richard, and lastName = Scanlan. Returns the profile, initialized as customer.
2 - Open Account	Tests to make sure that the date in which a particular customer Profile has been opened is valid. This means that only valid inputs (integers) for month, day and year values can be inputted to add a date to the opened account. ● Case 1: Entering alphabetical values for day, month and year for date parameters ● Case 2: Enter nothing into day, month and year	● Case 1: monthValue = “xxx” dayValue = “yyy” yearValue = “wry” ● Case 2: monthValue = “” dayValue = “” yearValue = “” ● Case 3: monthValue = 11 dayValue = 25	● Case 1: Returns the catch statement for NumberFormatException. “Non-numeric data has been entered for date; please enter integers.” ● Case 2: Returns the catch statement for NumberFormatExce

	• Case 3: Year is greater than 2020 • Case 4: Year is a negative integer • Case 5: Value entered for month is greater than 12 • Case 6: Value entered for month is equal to 0 • Case 7: Value entered for day is equal to 0 • Case 8: Normal date entered • Case 9: Month 11 (November) is given 31 days, when it has exactly 30 days • Case 10: Month 3 (March) is given 31 days (correct) • Case 11: Month 2 (February) is given 30 days • Case 12: Month 7 (July) is given 31 days • Case 13: A leap year is inputted for year value, so February is given 29 days • Case 14: A leap year is inputted for year value, and February is given 28 days • Case 15: Not a leap year, and February is given 29 days • Case 16: Not a leap year, and February is given 28 days • Case 17: Not a leap year (divisible by 4 and 100, but not by 400) and February is given 29 days • Case 18: Month 6 (June) is given 30 days, and year is not a leap year • Case 19: Month 5 (May) is given 31 days, and year is a leap year • Case 20: Month 4 (April) is given 30 days, and year is a leap year	yearValue = 2021 • Case 4: monthValue = 11 dayValue = 25 yearValue = -1 • Case 5: monthValue = 13 dayValue = 25 yearValue = 2019 • Case 6: monthValue = 0 dayValue = 25 yearValue = 2019 • Case 7: monthValue = 11 dayValue = 0 yearValue = 2019 • Case 8: monthValue = 7 dayValue = 25 yearValue = 2018 • Case 9: monthValue = 11 dayValue = 31 yearValue = 2019 • Case 10: monthValue = 3 dayValue = 31 yearValue = 2019 • Case 11: monthValue = 2 dayValue = 30 yearValue = 2019 • Case 12: monthValue = 7 dayValue = 31 yearValue = 2018 • Case 13: monthValue = 2 dayValue = 29 yearValue = 1980 • Case 14: monthValue = 2 dayValue = 28 yearValue = 1960 • Case 15: monthValue = 2 dayValue = 29 yearValue = 1981 • Case 16: monthValue = 2 dayValue = 28 yearValue = 1981 • Case 17: monthValue = 2	ption. “Non-numeric data has been entered for date; please enter integers.” • Case 3: This is an invalid date, so we return “11/25/2021 is invalid.” • Case 4: This is an invalid date, so we return “11/25/-1 is invalid.” • Case 5: This is an invalid date, so we return “13/25/2019 is invalid.” • Case 6: This is an invalid date, so we return “0/25/2019 is invalid.” • Case 7: This is an invalid date, so we return “11/0/2019 is invalid.” • Case 8: This is a valid date, so we create an account given 7/25/2018 as our date opened. • Case 9: This is an invalid date, so we return “11/31/2019 is invalid.” • Case 10: This is a valid date, so we create an account given 3/31/2019 as our date opened. • Case 11: This is an invalid date, so we return “2/30/2019 is invalid.” • Case 12: This is a valid date, so we create an account given 7/31/2018 as our date opened. • Case 13: This is a valid date because 1980 is a leap year, so we create an account given 2/29/1980 as our date opened.

		dayValue = 29 yearValue = 1900 ● Case 18: monthValue = 6 dayValue = 30 yearValue = 2011 ● Case 19: monthValue = 5 dayValue = 31 yearValue = 1912 ● Case 20: monthValue = 4 dayValue = 30 yearValue = 1944	● Case 14: This is a valid date, so we create an account given 2/28/1960 as our date opened. ● Case 15: This is not a valid date because 1981 is not a leap year, so we return "2/29/1981 is invalid." ● Case 16: This is a valid date because there are 28 days in February when there is not a leap year, so we create an account given 2/28/1981 as our date opened. ● Case 17: 1900 is not a leap year because it is not divisible by 400 (but it is divisible by 4 and 100). Therefore, there cannot be 29 days of February in 1900. We return "2/29/1900 is invalid." ● Case 18: This is a valid date, so we create an account given 6/30/2011 as our date opened. ● Case 19: This is a valid date, so we create an account given 5/31/1912 as our date opened. ● Case 20: This is a valid date, so we create an account given 4/30/1944 as our date opened.
3 - Open Account	Tests to make sure that a valid integer (greater than 0) is being inputted for opening balance in account. ● Case 1: Entering a negative balance. ● Case 2: Entering 0 as balance. ● Case 3: Entering alphabetical values for balance.	● Case 1 balanceValue = -200 ● Case 2 balanceValue = 0 ● Case 3 balanceValue = wxy ● Case 4	● Case 1: Returns the value in if-statement that checks if balanceValue is less than or equal to 0. "You cannot open an account with a

	• Case 4: Entering nothing for balance. • Case 5: Entering a valid integer that is greater than 0 for balance.	balanceValue = (empty) • Case 5 balanceValue = 550	balance less than or equal to $0." • Case 2: Returns the value in if-statement that checks if balanceValue is less than or equal to 0. "You cannot open an account with a balance less than or equal to $0." • Case 3: Returns the catch block for NumberFormatException. "Non-numeric data has been entered: please enter a decimal for balance." • Case 4: Returns the catch block for NumberFormatException. "Non-numeric data has been entered: please enter a decimal for balance." • Case 5: Parses through and saves the balance value within the current account being created.
4 - Open Account	Tests to make sure that a checking account can be opened that is either a direct deposit account or not a direct deposit account. • Case 1: Select checking for account type and select direct deposit. • Case 2: Select checking for account type and don't select direct deposit.	• Case 1: Selected checking account on GUI Selected direct deposit on GUI • Case 2: Selected checking account on GUI Don't select direct deposit on GUI	• Case 1: Successfully creates a checking account that is also a direct deposit account. • Case 2: Successfully creates a checking account that is not a direct deposit account.
5 - Open Account	Tests to make sure that a savings account can be opened that makes the customer either a loyal customer, or	• Case 1: Selected savings account on GUI Selected loyal customer on GUI	• Case 1: Successfully creates a savings account that is also

	not a loyal customer. ● Case 1: Select savings for account type and select loyal customer. ● Case 2: Select savings for account type and don't select loyal customer.	● Case 2: Selected savings account on GUI Don't select loyal customer on GUI	a loyal customer account. ● Case 2: Successfully creates a savings account that is not a loyal customer account.
6 - Open Account	Tests to make sure that a money market account can be opened. ● Case 1: Select money market for account type.	● Case 1: Selected money market on GUI	● Case 1: Successfully creates a money market account.
7 - Open Account	Tests to make sure that an account cannot be opened with empty values for all fields. ● Case 1: Don't input anything into any field and simply press create account button.	● Case 1: No inputs into any fields on GUI	● Case 1: Does not create an account. Returns the first exception that arises, from scanName class. "Please make sure you enter both a proper first and last name."
8 - Checking Account	Tests to make sure that monthlyInterest has been calculated correctly, with annual interest being 0.05%. ● Case 1: Check that the monthly interest for a checking account with profile = "Jane Doe", date = "5/7/2018", directDeposit = false and balance = 1000 is equivalent to $0.041. ● Case 2: Check that the monthly interest for a checking account with profile = "Jane Doe", date = "5/7/2018", directDeposit = true and balance = 950 is equivalent to 0. ● Case 3: Check that the monthly interest for a checking account with profile = "Jane Doe", date = "5/7/2018", directDeposit = true and balance = 3000 is equivalent to $0.125. ● Case 4: Check that the monthly interest for a checking account with profile = "Jane	● Case 1: Profile = "Jane Doe" Date = "5/27/2018" directDeposit = false Balance = 1000 ● Case 2: Profile = "Jane Doe" Date = "5/27/2018" directDeposit = false Balance = 950 ● Case 3: Profile = "Jane Doe" Date = "5/27/2018" directDeposit = false Balance = 3000 ● Case 4: Profile = "Jane Doe" Date = "5/27/2018" directDeposit = false Balance = 1500	● Case 1: Returns that the monthlyInterest is equivalent to $0.041. ● Case 2: Returns that monthlyInterest is equivalent to 0. ● Case 3: Returns that monthlyInterest is equal to $0.125. ● Case 4: Returns that monthlyInterest is equal to $0.0625. (These changes are reflected when we print the accounts).

	Doe", date = "5/7/2018", directDeposit = false and balance = 1500 is equivalent to $0.0625.		
9 - Checking Account	Tests to make sure that the monthly fee of $25 is only charged with the balance is less than 1500, as well as when account is not a direct deposit account. ● Case 1: Check that monthly fee of $25 is charged when balance is less than 1500 (1000) and directDeposit is false. ● Case 2: Check that monthly fee of $25 is not charged when balance is less than 1500 (950) and directDeposit is true. ● Case 3: Check that monthly fee of $25 is not charged when balance is greater than 1500 (3000) and directDeposit is true. ● Case 4: Check that monthly fee of $25 is not charged when balance is equal to 1500 and directDeposit is false.	● Case 1: Balance = 1500 directDeposit = false ● Case 2: Balance = 950 directDeposit = true ● Case 3: Balance = 3000 directDeposit = true ● Case 4: Balance = 1500 directDeposit = false	● Case 1: Returns monthly fee of $25 ● Case 2: Returns monthly fee of $0 ● Case 3: Returns monthly fee of $0 ● Case 4: Returns monthly fee of $0
10 - Money Market	Tests to make sure that annual interest rate of Money Market accounts, which is at 0.65%, is correctly calculated. ● Case 1: Checks to make sure that a sample account of balance 3000 yields a monthly interest of 1.625. ● Case 2: Checks to make sure that a sample account of balance 500 yields a monthly interest of 13/48.	● Case 1: Balance = 3000 7 withdrawals ● Case 2: Balance = 500 4 withdrawals	● Case 1: Returns a monthly interest of 1.625. ● Case 2: Returns a monthly interest of 13/48.
11 - Money Market	Tests to make sure that a monthly fee of $12 is waived when balance is greater than or equal to 2500, and if the number of withdrawals are less than or equal to 6. ● Case 1: Checks to make sure that even though balance exceeds 2500, if withdrawals exceed 6, monthly fee is not waived. ● Case 2: Checks to make sure that if balance exceeds 2500 and withdrawals are equal to 6, monthly fee is waived.	● Case 1: Balance = 3000 7 withdrawals ● Case 2: Balance = 2800 6 withdrawals ● Case 3: Balance = 4000 5 withdrawals ● Case 4: Balance = 2500 4 withdrawals	● Case 1: Charges a monthly fee of $12 ● Case 2: Does not charge a monthly fee of $12 ● Case 3: Does not charge a monthly fee of $12 ● Case 4: Does not charge a monthly fee of $12 ● Case 5: Does not

	• Case 3: Checks to make sure that if balance exceeds 2500 and withdrawals are less than 6, monthly fee is waived. • Case 4: Checks to make sure that if balance is equal to 2500 and withdrawals do not exceed 6, monthly fee is waived. • Case 5: Checks to make sure if balance is less than 2500 and withdrawals are exactly 6, the monthly fee is not waived. • Case 6: Checks to make sure that if balance is less than 2500 and withdrawals exceed 6, monthly fee is not waived. • Case 7: Checks to make sure that if balance is less than 2500 and withdrawals are less than 6, fee is not waived.	• Case 5: Balance = 2400 6 withdrawals • Case 6: Balance = 1000 8 withdrawals • Case 7: Balance = 500 4 withdrawals	charge a monthly fee of $12 • Case 6: Charges a monthly fee of $12 • Case 7: Charges a monthly fee of $12
12 - Savings	Tests to make sure that monthly fee of $5 is waived when the account balance is greater than or equal to $300. • Case 1: Checks to make sure that monthly fee is not charged when account balance equals 300. • Case 2: Checks to make sure that monthly fee is not charged when account balance is greater than 300. • Case 3: Checks to make sure that the monthly fee is charged when balance is less than 300.	• Case 1: Balance = 300 • Case 2: Balance = 450 • Case 3: Balance = 120	• Case 1: Does not charge a monthly fee. • Case 2: Does not charge a monthly fee. • Case 3: Charges a monthly fee of $5.
13 - Savings	Tests to make sure that annual interest rate of 0.25% is charged when the customer is not a loyal customer and annual interest rate of 0.35% when the customer is a loyal customer. • Case 1: Checks to make sure that annual interest rate of 0.25% is charged with a balance of 1000, when the customer is not a loyal customer. • Case 2: Checks to make sure that annual interest rate of 0.35% is charged with a balance of 1000, when the customer is a loyal customer.	• Case 1: Balance = 1000 isLoyal = false • Case 2: Balance = 1000 isLoyal = true	• Case 1: Charges the annual interest rate of 0.25%, resulting in a monthly interest of $0.208. • Case 2: Charges the annual interest rate of 0.35%, resulting in a monthly interest of $0.291.
14 - Close	Tests to make sure that account is closed only when a first name, last	• Case 1: First name = (empty)	• Case 1: Close account method

	name and account type is entered that matches an account in the accounts database. ● Case 1: Checks to make sure that account is not closed when there are no inputs for first name and last name. ● Case 2: Checks to make sure that account is closed when a name that is in accounts database has been entered. ● Case 3: Checks to make sure that account is not closed when name entered does not match an entry in the database. ● Case 4: Checks to make sure that account is not closed when only spaces are entered for first and last name. ● Case 5: Checks to make sure that account is not closed when valid entries are inputted for first and last name, but no radio button has been selected. ● Case 6: Checks to make sure that account is not closed when nothing has been entered, but close and clear buttons have been pressed.	Last name = (empty) ● Case 2: First name = “Richard” Last name = “Scanlan” ● Case 3: First name = “Abby” Last name = “Augusten” ● Case 4: First name = “ ” Last name = “ ” ● Case 5: First name = “Richard” Last name = “Scanlan” No radio button pressed ● Case 6: First name = (empty) Last name = (empty) Close and clear are pressed simultaneously	calls scanName, which then returns “Make sure you entered a value for both first name and last name.” ● Case 2: Successfully removes account and returns “Account has been removed.” ● Case 3: Searches for account in accounts database, and when it is not found, returns “Account does not exist.” ● Case 4: Close account method calls scanName, which then returns “Make sure you entered a value for both first name and last name.” ● Case 5: Returns “Please select an account type.” ● Case 6: Returns “Make sure you entered a value for both first name and last name.” This is because close() calls scanName() method.
15 - Accounts Database	Tests to make sure that accounts database is working properly, which means that its functionalities (print accounts, print statements by date and print statements by last name) are outputting the correct result. ● Case 1: Start with an empty accounts database and press print accounts, print statements by date and print statements by last name. ● Case 2: Create an accounts database of four entries with diverse last names and date	● Case 1: (Empty database) Press print accounts, print statements by date and print statements by last name ● Case 2: Open accounts ->\ First name: Alicia Last name: Aladdin Date opened: 10/21/2017 First name: Eunice	● Case 1: Returns “Database is empty.” ● Case 2: Print accounts: 1) Alicia Aladdin 2) Eunice Culkins 3) Xander Smith 4) Walter Rhoads Print statements by date opened: 1) Alicia Aladdin 2) Xander Smith

	opens. Then, press print accounts, print statements by date and print statements by last name. ● Case 3: Input the duplicate of an account that is in accounts database, and check to make sure that only one entry of that account is there when you print all of the accounts. ● Case 4: Start with an empty database and click print options quickly and simultaneously, to make sure that the GUI doesn't crash.	Last name: Culkins Date opened: 9/2/2019 First name: Xander Last name: Smith Date opened: 9/1/2018 First name: Walter Last name: Rhoads Date opened: 10/1/2020 Press print accounts, print statements by date and print statements by last name ● Case 3: First name: Xander Last name: Smith Press print accounts, print statements by date and print statements by last name ● Case 4: (Empty database) Press print accounts, print statements by date and print statements by last name	3) Eunice Culkins 4) Walter Rhoads Print statements by last name: 1) Alicia Aladdin 2) Eunice Culkins 3) Walter Rhoads 4) Xander Smith ● Case 3: Print accounts: 1) Alicia Aladdin 2) Eunice Culkins 3) Xander Smith 4) Walter Rhoads Print statements by date opened: 1) Alicia Aladdin 2) Xander Smith 3) Eunice Culkins 4) Walter Rhoads Print statements by last name: 1) Alicia Aladdin 2) Eunice Culkins 3) Walter Rhoads 4) Xander Smith ● Case 4: Program does not crash.
16 - Project 2 Sample Input	Tests to make sure that the appropriate, correctly formatted entries in Project 2 test cases work for Project 3's GUI.	We input the sample input that has been formatted correctly from Project 2's documentation.	We should get the corresponding output within Project 2's documentation.
17 - GUI Functionality Test	Tests to make sure that GUI does not crash when any buttons are pressed. ● Case 1: Open two accounts, and print accounts. ● Case 2: Close one of those two accounts, and print accounts. ● Case 3: In the remaining account, deposit once and print to make sure that the account's balance has been increased by that specific amount. ● Case 4: in the remaining account, withdraw the same amount and make sure that the balance is decreased by	● Case 1: First name: Scarlett Last name: Johansen Balance: 100 First name: Sonia Last name: Brown Balance: 200 Print accounts. ● Case 2: Close Scarlett Johansen. Print accounts. ● Case 3: Deposit: 500 into Sonia Brown's	● Case 1: Print accounts: 1) Scarlett Johansen 2) Sonia Brown ● Case 2: Print accounts: 1) Sonia Brown ● Case 3: Deposit: 500 1) Sonia Brown, Balance = 700 ● Case 4: Withdraw: 500 1) Sonia Brown, Balance = 200

	that amount.	account. ● Case 4: Withdraw: 500 from Sonia Brown's account Make sure balance is decreased by that amount.	
18 - Deposit/Withdraw	We checked deposit and withdraw with the tests that we performed in the balance field of open account as well as close account to make sure the values were printed correctly and no exception occurs.	Similar to previous tests	Similar to previous outputs