Ulster
University

COM809

Professional Software Development II

Group Project 2020

Group 11

Donal Doherty, Fearghal O’Boyle and Emma Mason

TABLE OF CONTENTS

1.0. Introduction

For this assignment, our team was tasked with analysing, designing, developing, testing, and delivering a Java Application. The team includes Emma Mason, Fearghal O'Boyle and Donal Doherty. Emma and Fearghal's backgrounds are in education while Donal Doherty's experience is in marketing and photography.

2.0. Development Methodology

An organised approach was essential to completing the group project. To ensure we worked as efficiently as possible and with all members undertaking an equal workload, we decided to utilise the agile methodology.

The agile approach to software development allows a project to evolve and change through its development cycle and each team member must be flexible and willing to adapt to change. Rather than using the waterfall development methodology, which can be compared to "running a long marathon to an imaginary destination" (Bibik, 2018, pg. 16), we opted to use the 'sprints' concept. By utilising this concept, we divided our workload into small chunks or tasks to be completed during sprints or short periods of time. This enabled us to "plan and predict the outcome" of our work after each sprint and then "decide where to go next" (Bibik, 2018, pg. 16).

Before each sprint, our group participated in a planning meeting. These planning sessions took place within Blackboard's Collaborate Ultra groups or Discord. During these meetings as a group, we would decide on the tasks we would commit to completing during our next development sprint. During our first planning session, we identified the key tasks we wanted to complete by the end of our first sprint, which we decided would be weeklong in length. We decided to have a 'User' super-class; a 'FullTimeInfluencer' and a 'CasualInfluencer' sub-class; and a 'Test' program completed. These programs and classes contained stripped down versions of the features we wished to include in our final program. We then decided, as a group, who would complete each task. If someone felt they would be suited to one task over another, they could voice their opinion, and this would be considered during this meeting. This worked effectively as "the most important part (of working as a scrum team) is that the team will be self-governed" (Bibik, 2018, pg. 16).

During our sprints, we also made use of daily meetings, usually on the group Discord server. We discussed our progress so far and any potential difficulties we were having. These meetings were invaluable to our group as they made it easier to detect any mistakes or inefficient code which could cause issues in the future.

These meetings also enabled each group member to embrace change and become more flexible with our programming. If we had not met regularly to discuss our progress until after each sprint was completed, and the group decided to alter a feature one member worked on, they might feel as if they had wasted their time programming it. However, if this feedback was given while they were programming during the sprint, the task of changing the code would be less laborious.

After each sprint, we then had a sprint review meeting. During these, each member uploaded their work into a shared GitHub folder and talked the other members through their code. The review meetings were longer than our usual daily meetings and acted as another opportunity for each group member to voice their opinion. It also allowed each member to suggest how we could improve the work carried out from our previous sprint, thus allowing us to continuously improve our program. Once we had finished discussing the work completed during the sprint, we could then decide what our next steps would be, what tasks needed to be completed during our next sprint and how this workload would be evenly split.

As this cycle of sprints, daily meetings and debriefs repeated every week, we soon found ourselves closer and closer to our final program being completed. This system of organisation allowed us to be efficient in our programming and hence more ambitious in the features we wished to include, as each iteration improved upon its predecessor. The delegation of tasks can be seen in Appendix D.

3.0. Analysis of Requirements

Instagram is a social network with over one billion active users. Some use the social network to connect with and make new friends and others use it to grow their business. Reach is an Instagram Insights tool designed to help users grow their tribe and turn followers into clients. Currently, to our best knowledge, there is no fit for purpose system to offer this.

Using insights, which indicate how well your account is performing, Reach provides the user with a rank and depending on the rank either asks for or provides a tip on how to grow your account. In this way users who are doing well can help the community with tips and enjoy the accountability of the tool. Those trying to grow their following will be motivated to increase their rank and will benefit from those tips.

The tool segments users into those who use Instagram for fun, Casual Users and those who use it for business, Pro Users. Their experience is tailored to their needs with Casual Users receiving tips and unique follower interaction analytics and Pro Users empowered to provide tips should their account be performing well.

While Insights are available from Instagram, the key problem solved by Reach is analysis of those statistics, a rank to motivate users to perform better and guidance on how to improve the performance of your account.

Instagram provides a visual platform to help users to build a more personal connection with friends and customers. Reach aims to be a powerful marketing aid to expand the user's presence and visibility.

4.0. Design

4.1. Program Preparation Sheets

Program preparation sheets were designed and written before each part of code was developed. This helped the group to visualise the piece of code and ensured that each part of code was well thought out, ultimately making writing the program more efficient and all group members could see what each other were aiming for. Outlined below are the preparation sheets created for the super class User, sub class Casual and the read class. See Appendix A for the remaining preparation sheets used for the reach program.

Name: Emma Mason
Date: 18/11/20
Class Name and Description: User. The super class of the reach program which both Casual, Pro and RegistrationTransaction take information from.

Program Design	**Data Requirements**
• Declare instance variables specific for all Users. • Declare a DecimalFormat object. • Create two constructors, one to include all instance variables except the users likes and one to include all instance variables in the parameters. • Getter and setter methods for each instance variable. • Method to prompt for and return an updated follower count. • Method to prompt for and set updated likes. • Method to calculate the average number of likes and return a double • Method which is void to print the likes. • Method to welcome the user. Method to print the user detail by returning a String.	• Static private DecimalFormat *df*=new DecimalFormat("0.00"); • private String userAt, username, password; private int totalFollowers, likes1, likes2, likes3; • public boolean proInfluencer = false, casualUser = false; • public User(String userName, String password, String userAt, int totalFollowers) • public User(String userName, String password, String userAt, int totalFollowers, int likes1, int likes2, int likes3)

Name: Emma Mason
Date: 18/11/20
Class Name and Description: Casual User, a sub class of super-class User. A class to store the variables and methods specific to a casual Instagram user. These methods will include constructors, getters, and setters for all the instance variables, a calculation method for user interaction, a method to calculate their reach grade, a method to return tips and print and update methods for certain variables.

Program Design	Data Requirements
• Declare instance variables specific for a casual user. • Constructors for the casual user objects • Getters and setters for the instance variable • Method to calculate follower's interaction with the user • Method to calculate the user's reach • Method to check a user's reach score. If low, give the user improvement tips. • Print methods. • To string method for all user information	• Super from user class of username, password, Instagram handle and their total followers as well as int variables of the user's latest three likes – int likes1, likes2, likes3. • Users weekly Instagram posts – int postsPerWeek. • Users daily Instagram stories – int storiesPerDay. • Users interaction percentage – double interactPercentage. • A grade calculated based on interaction – char grade. • A String array to store various tips on when to post on Instagram – String [] timeimprovement.

Name: Fearghal O'Boyle
Date: 18/11/20
Class Name and Description:
Pro User, a sub class of the class user. This class stores the variables and methods which are specific to a professional Instagram user. To build the pro user object, the pro constructor will be invoked. These objects will also use accessor, mutators and various other methods which will be outlined below.

Program Design	Data Requirements
• Declare instance variables specific to a pro user object.	• Private int accountsReached, interactions, gainedFollowers; String accountsReachedTip, interactionsTip, gainedFollowersTip.
• Decimal Format with pattern "00".	• Decimal Format used to round numbers.
• String userTips and char grades array with dimensions of 3.	• String array for user tips and Char array to hold their grades.
• Constructors to build pro user objects.	• Uses the super class constructor. Assigns formal parameters to instance variables. • Alternative constructor also included if user does not enter parameters.
• Accessors and Mutators for each instance variable.	• Used within the methods of the Pro class and in the UserInterface program.
• Void welcome method.	• Prints to the screen a welcome message specific to pro accounts.
• String getText method.	• String Prompt passed in as a parameter.
• Void method to rankAccountsReached.	• User enters the number of accounts they reached this week as an integer. Method then assigns a grade which is stored in the char array. If the rank is high, the user is prompted to enter a tip, this is then stored in the String array.
• Void method to rankInteractions.	

• Void method to rankFollowers. • Void method printUserEnteredTips. • String method showTip. • Public String toString method.	• User enters the number of interactions as an integer. The method then assigns a grade which is stored in the char array. If the rank is high, the user is prompted to enter a tip. This is then stored in the String array. • User enters the number of followers as an integer. The method then assigns a grade which is stored in the char array. If the rank is high, the user is prompted to enter a tip. This is then stored in the String array. • Method prints the tips that have been entered by the user from String array userTips. • Method prints a random tip from a String array called tips. • String method which returns the super class toString, and the grades from the char array grades.

Name: Emma Mason
Date: 27/11/20
Class Name and Description: Read class. A class to retrieve user input with exception handling. Should the user enter a null value or an empty string, they will be asked to renter this information again to ensure input.

Program Design	**Data Requirements**
• A String method to read in a value from the user. Can be changed to an int with ParseInt method. • A print statement with a general prompt will be used. • Try and catch method within a while loop to ensure user input (exception handling) • String method will return the value user has entered.	• String value for user input.

4.2. UML Diagrams

UML class diagrams were used as seen in Figure 1. The class diagram shows the different objects in the system, the attributes they possess as well as the operations and relationships amongst them (Sparks, 2011). Figure 2 is a use case diagram, which demonstrates how the user interacts with the system.

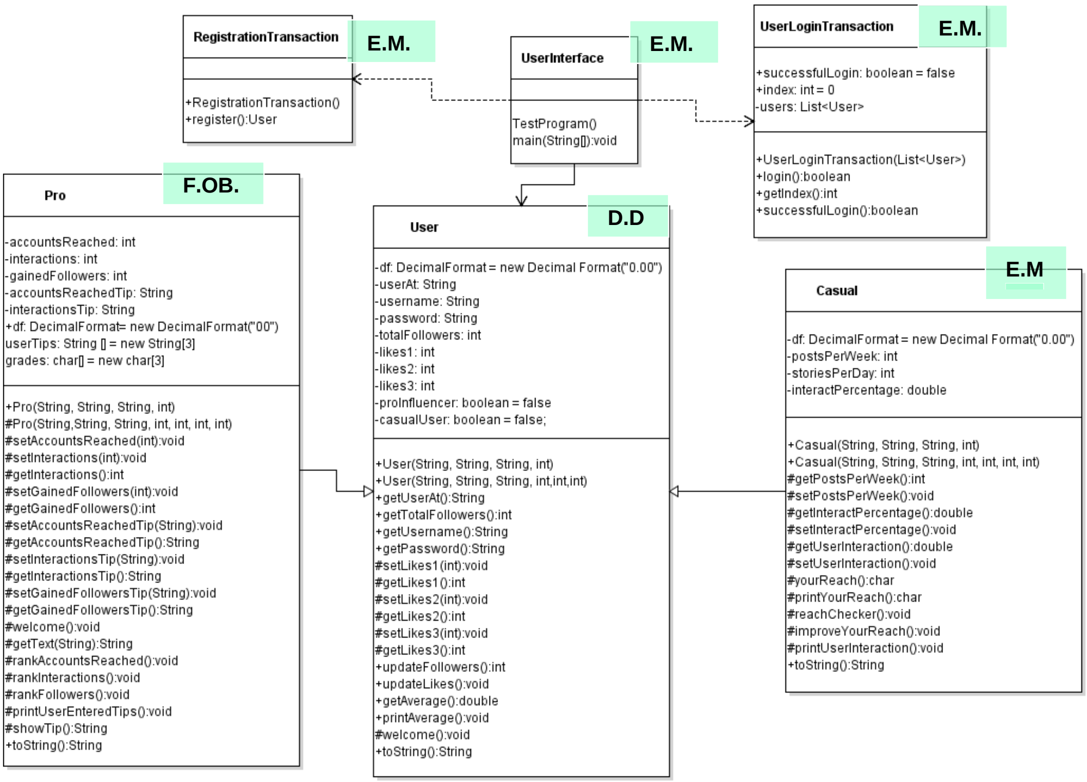
RegistrationTransaction
E.M.
+RegistrationTransaction()
+register():User
UserInterface
E.M.
TestProgram()
main(String[]):void
UserLoginTransaction
E.M.
+successfulLogin: boolean = false
+index: int = 0
-users: List<User>
+UserLoginTransaction(List<User>)
+login():boolean
+getIndex():int
+successfulLogin():boolean
Pro
F.OB.
-accountsReached: int
-interactions: int
-gainedFollowers: int
-accountsReachedTip: String
-interactionsTip: String
+df: DecimalFormat= new DecimalFormat("00")
userTips: String [] = new String[3]
grades: char[] = new char[3]
+Pro(String, String, String, int)
#Pro(String,String, String, int, int, int, int)
#setAccountsReached(int):void
#setInteractions(int):void
#getInteractions():int
#setGainedFollowers(int):void
#getGainedFollowers():int
#setAccountsReachedTip(String):void
#getAccountsReachedTip():String
#setInteractionsTip(String):void
#getInteractionsTip():String
#setGainedFollowersTip(String):void
#getGainedFollowersTip():String
#welcome():void
#getText(String):String
#rankAccountsReached():void
#rankInteractions():void
#rankFollowers():void
#printUserEnteredTips():void
#showTip():String
+toString():String
User
D.D
-df: DecimalFormat = new Decimal Format("0.00")
-userAt: String
-username: String
-password: String
-totalFollowers: int
-likes1: int
-likes2: int
-likes3: int
-proInfluencer: boolean = false
-casualUser: boolean = false;
+User(String, String, String, int)
+User(String, String, String, int,int,int)
+getUserAt():String
+getTotalFollowers():int
+getUsername():String
+getPassword():String
#setLikes1(int):void
#getLikes1():int
#setLikes2(int):void
#getLikes2():int
#setLikes3(int):void
#getLikes3():int
+updateFollowers():int
+updateLikes():void
+getAverage():double
+printAverage():void
#welcome():void
+toString():String
Casual
E.M
-df: DecimalFormat = new Decimal Format("0.00")
-postsPerWeek: int
-storiesPerDay: int
-interactPercentage: double
+Casual(String, String, String, int)
+Casual(String, String, String, int, int, int, int)
#getPostsPerWeek():int
#setPostsPerWeek():void
#getInteractPercentage():double
#setInteractPercentage():void
#getUserInteraction():double
#setUserInteraction():void
#yourReach():char
#printYourReach():char
#reachChecker():void
#improveYourReach():void
#printUserInteraction():void
+toString():String

Figure 1, Class diagram showing the relationship between classes

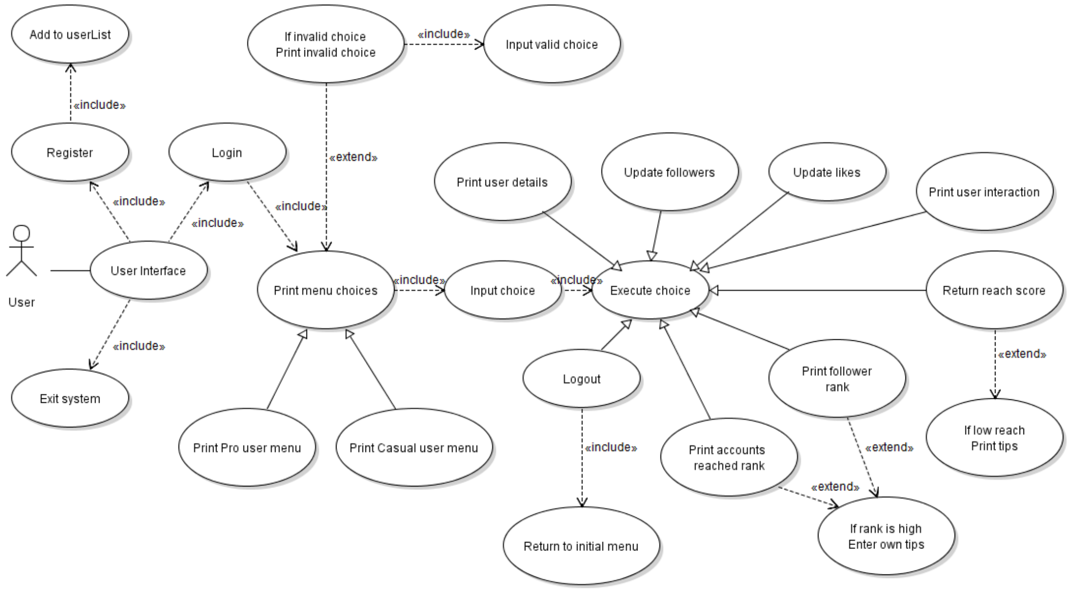

Figure 2, Use case diagram illustrating how the user interacts with the system

5.0. Implementation

We all use Instagram, Emma and Fearghal for fun, and Donal for business. After brainstorming our team came up with the concept for an analytics tool for the platform called 'Reach'. Our initial idea was to help users grow their following. Through the research and development process we decided that this could best be achieved through a ranking system coupled with tips on how to improve account performance. The ranking system gamifies the process of growing your account, giving accountability, and encouraging the user to improve their rank over time. Community is at the heart of social media and we wanted to build that into the DNA of Reach. Users can share what is working for them and these tips are shared with other

users. Reach equips the user with strategies to grow their account, and as more tips are fed into the program it gets better and better with time.
As some people are on Instagram for personal use and others for business, and with different Insights available depending on your account type, we created two distinct user experiences to tailor it to their needs. Although we were constrained with time for the assignment, we took the initiative to add account functionality which enables the user to register an account and login. This added complexity during the development process, however we agreed it would elevate our program and the user experience. A user guide is available for the end user, to improve ease of access (see Appendix C).

6.0. Testing Strategy

Testing was of upmost importance as a blended approach was taken towards developing the program. This may have increased the risk of errors or mistakes due to three different developers working on various aspects of the program. Testing was completed alongside the development of the project for continuous assessment purposes. The aim of the vigorous testing was to ensure no omissions in the requirements and to eliminate as many defects as possible.
The use case diagram outlined in section 4.2, Figure 2, defines how the user interacts with the system and the Test Analysis in Appendix B (see Appendix B) was based on these interactions. Test Cases were generated using each scenario in which the user interacts with the system and providing exact values or actions to these interactions. Testing was completed through the User Interface, testing the initial menu, Casual User menu, the Pro User menu and all the methods used within these. The test cases revealed errors which were overlooked and required modification. Outlined in the appendix, TCase_11 exposed an error with input regarding registration. A method of the String class used failed to account for the difference between lower and uppercase letters. This was causing a breakpoint if the user entered a lowercase ‘c’. The method was subsequently revised and changed to .equalsIgnoreCase(), allowing the user to enter either lower or uppercase input. An editing oversight was discovered in TCase_24. The menu options print statement had been updated, but not the associated switch statement. This meant the system was unresponsive when the user entered the logout option. The same mistake was discovered in TCase_47, which showed this was a common oversight. The defect was easily amended through changing the switch statement case number accordingly. The use of test cases ensured that mistakes such as these did not cause a deficiency in the functionality of the program. Through using test cases, the reliability of the program was confirmed as well as the functionality and ease of access for the user. The program proved to satisfy user requirements through the testing analysis.

8.0. Conclusion

The Reach program, through organised development, preparation, teamwork, and communication, has evolved into a fully functioning application. The feature of providing statistics for not only professional influencers but also casual users, gives Reach a unique selling point and exclusive information for the user. The use of user accounts provides a personalised experience for the user with high functionality.
This collaborative experience has allowed for all group members to improve their skills holistically, in Java, testing, documentation and teamwork. Together the group worked effectively and efficiently to produce a high-quality program which utilises a range of coding techniques.

References

Ilya Bibik (2018). How to Kill the Scrum Monster. New York: Apress.

Sparks, G. (2011) *Database Modelling in UML,* Available at: *http://www.methodsandtools.com/archive/archive.php?id=9* (Accessed: 9th December 2020).

Appendix

Appendix A

Program Preparation Sheets for the Reach Program.

Name: Fearghal O’Boyle

Date: 18/11/20

Class Name and Description:

Pro User, a sub class of the class user. This class stores the variables and methods which are specific to a professional Instagram user. To build the pro user object, the pro constructor will be invoked. These objects will also use accessor, mutators and various other methods which will be outlined below.

Program Design	**Data Requirements**
• Declare instance variables specific to a pro user object.	• Private int accountsReached, interactions, gainedFollowers; String accountsReachedTip, interactionsTip, gainedFollowersTip;
• Decimal Format with pattern “00”.	• Decimal Format used to round number of average likes to a whole number.
• String userTips array with a dimension of 3 and char grades array with a dimension of 3.	• String array to hold the good practice tips entered by users when they achieve a high rank. Char array to hold the grades they have been assigned.
• Constructors to build pro user objects.	• Make use of the super class constructor passing it the variables username, password, userAt, totalFollowers as parameters. Then assigns the passed in formal parameters to instance variables of accountsReached, Interactions and gainedFollowers. • Alternative constructor also included if user does not enter parameters for accountsReached, interactions and gainedFollowers.
• Accessors and Mutators for each instance variable.	• Used within the methods of the Pro class and in the UserInterface program.
• Void welcome method.	• Prints to the screen a welcome message specific to pro accounts, this is displayed in the pro account menu.
• String getText method, utilised in the rank methods for when a user is prompted to enter their good practice.	• String Prompt passed in as a parameter which is then printed. User enters their text at the keyboard which is then returned.
• Void method to rankAccountsReached.	• User enters the number of accounts they reached this week as an integer. Method then assigns a grade to the user based upon this number in relation to their total number of followers. Rank characters are stored in the char array. If the rank is high, the user is prompted to enter what gained them such a

	high rank and this is then stored in the String array.
• Void method to rankInteractions.	• User enters the number of interactions they had this week with their followers as an integer. The method then assigns a grade based upon this number. The rank character is then stored in the char array. If the rank is high, the user is prompted to enter what gained them such a high rank regarding interactions. This is then stored in the String array.
• Void method to rankFollowers.	• User enters the number of followers they gained this week as an integer. The method then assigns a grade based upon this number. The rank character is then stored in the char array. If the rank is high, the user is prompted to enter what gained them such a high rank regarding gained followers. This is then stored in the String array.
• Void method printUserEnteredTips.	• Method prints the tips that have been entered by the user and stored in the String array userTips. If the user has not entered tips for a certain category, they are informed that they have not achieved a high enough rank to enter a tip.
• String method showTip.	• Method contains a String array called tips. Each index position is assigned a String tip and the method uses a Math.random command to return a random number corresponding to an index position in the String array.
• Public String toString method.	• String method which returns the super class toString, and the grades for Accounts Reached, Interactions and Followers from the char array grades.

Name: Emma Mason
Date: 27/11/20
Class Name and Description: Read class. A class to retrieve user input with exception handling. Should the user enter a null value or an empty string, they will be asked to renter this information again to ensure input.

Program Design	**Data Requirements**
• A String method to read in a value from the user. Can be changed to an int with ParseInt method. • A print statement with a general prompt will be used. • Try and catch method within a while loop to ensure user input (exception handling) • String method will return the value user has entered.	• String value for user input.

Name: Emma Mason
Date: 27/11/20
Class Name and Description: RegistrationTransaction class. A class to register a new user to the application and store their password and username. Will distinguish between full time and casual user during registration.

Program Design	Data Requirements
• Prompt user for information included in the User class – username, password, Instagram handle and their total followers. • Will then ask user for their type – full time or casual. • If statement to filter the users to their appropriate objects – full time or user. • User will be returned.	• User object for default. • String username, password, userAt. • Int totalFollowers. • String userType.

Name: Emma Mason
Date: 27/11/20
Class Name and Description: UserLoginTransaction class. A class to allow for login of hard coded users and newly registered users.

Program Design	Data Requirements
• Read in username and password using read method. • Use a for loop o count through the entry. • Return user not found if it does not match. • Return true if user is found.	• Boolean successfulLogin. • Int index. • An array list – List<User> users. • String username, password. • Int pos.

Appendix B

Table 1. Testing Strategy.

Test case ID	Objective	Preconditions	Test data	Expected Result	Test completion date	Status	Tester	Comments
TCase_1	Check if entering 3 into the console will end the code	The user is on the initial menu on the UserInterface class	Entering option ‘3’.	Code will print a thank you message and will exit with exit code 0.	05/12/20	Passed	Emma Mason	
TCase_2	Check if the login prompts come up when user enters 1 into the console	The user is on the initial menu on the UserInterface and presses 1.	Entering the option ‘1’.	Code will prompt with a message “Provide your username”	05/12/20	Passed	Emma Mason	
TCase_3	Check when the user enters a username, a prompt for password will print	The user has pressed 1 on the initial user interface menu and has entered a username.	Entering hardcoded username ‘u2’.	Code will display a prompt with message “Provide your password”	05/12/20	Passed	Emma Mason	
TCase_4	Check if when a casual user enters a hardcoded password, they will be logged into an account	The user has entered 1 on the initial UserInterface menu and the entered u2 at the username prompt and p2 at the password prompt.	Entering hardcoded username ‘u2’ and hardcoded password ‘p2’	Code will display “Casual User Account” and a specific casual user menu, along with a enter choice prompt.	05/12/20	Passed	Emma Mason	
TCase_5	Check if when a Pro user enters a hardcoded password, they will be logged into an account	The user has entered 1 on the initial UserInterface menu and the entered u1 at the username prompt and p1 at the password prompt.	Entering hardcoded username ‘u1’ and hardcoded password ‘p1’	Code will display “Pro Influencer Account” and a specific Pro user menu, along with an	05/12/20	Passed	Emma Mason	

				enter choice prompt.				
TCase_6	Check if when entering a username and password that has not been created an error message prints.	Enter 1 in the initial UserInterface menu and enter a new username and password.	Enter “emma” in the username prompt and “mason” in the password prompt.	A message will display “User not found” and the initial menu will display again.	05/12/20	Passed	Emma Mason	
TCase_7	Check if when a user enters 2 into the console, a prompt for registering a new user and entering a username will come up.	Run the program and enter 2 in the initial UserInterface Menu.	Enter ‘2’ into the console.	A message will display “Provide your Username”.	05/12/20	Passed	Emma Mason	
TCase_8	Following TCase_7, check that following entering a username, a prompt for a password appears.	Have entered 2 into the console on the initial UserInterface menu and entered a username into the console.	Enter ‘e1’ into the console and enter.	A message prompt will display “Provide your password”	05/12/20	Passed	Emma Mason	
TCase_9	Following TCase_8, the user will be prompted to enter their Instagram handle after entering a password.	Following entering a username into the console, a enter a password prompt will be displayed.	Enter “p1” into the console and enter.	A message will print prompting user for their Instagram handle.	05/12/20	Passed	Emma Mason	
TCase_10	Check that when registering, the program allows input of an Instagram	User has registered with a username and password into the console. A prompt for Instagram handle has printed.	Enter “emmamasonnn” into the console.	A message will print, prompting the user for their	05/12/20	Passed	Emma Mason	

	handle and then displays a prompt for total followers.			total followers.				
	Check that when registering as a new user, the program allows the input of an integer for total follower prompt.	User has registered with a username, password and Instagram handle into the console. A prompt for total followers has printed.	Enter '300' into the console.	A message will print, asking the user whether they a Pro or casual user.	05/12/20	Passed	Emma Mason	
TCase_11	Check that when the user enters that they are a casual user, an account is created.	User has registered with a username, password, Instagram handle and total followers into the console. A prompt for whether the user is a casual or pro user has printed.	Enter 'c' into the console.	A message will print "An account has been created. Please login."	05/12/20	Failed	Emma Mason	I had used .contains the registration class, which did not account for entering a lowercase version of 'c'. Changed both pro and casual to equalsIgnoreCase.
TCase_12	Check that when the user enters that they are a pro user, an account is created.	Follow the steps to register a new user by entering a username, password, Instagram handle and total followers.	Enter 'P' into the console at the prompt asking what type of user they are.	A message will print "An account has been created. Please login."	05/12/20	Passed	Emma Mason	
TCase_13	Check that following registration of a new user, they can login.	User has registered a new user and is back on the main menu in UserInterface.	Enter '1' into the console and enter 'e1' into the username prompt and 'p1' into the password prompt.	Code will display "Casual User Menu" along with a list of options.	05/12/20	Passed	Emma Mason	
TCase_14	Check that when option 1 on the casual user menu is entered,	User has logged into the system with the credential 'e1' and 'p1'. User has the	Enter '1' into the console at the 'Provide your choice:' prompt.	Code will display all user details using the	05/12/20	Passed	Emma Mason	

	the user who is logged in details are printed.	Casual User Account menu on the screen.		toString method				
TCase_15	Check that when the user enters 2 on the Casual User Account Menu, a prompt to update followers is printed on the screen and allows input.	User has logged into the system with the credential 'e1' and 'p1'. User has the Casual User Account menu on the screen.	Enter '2' into the console at the 'Provide your choice:' prompt. Enter '500' into the console at the prompt.	Code will display a prompt 'Provide your follower count today:' and user will be able to enter a number. User will be brough back to Casual User Account Menu	05/12/20	Passed	Emma Mason	
TCase_16	Check that following TCase_15, the user account follower count has updated by printing the user details.	User is on the Casual User Account menu logged in with credentials 'e1' and 'p1'. User has updated follower count to 500.	Enter '1' into the console at the 'Provide your choice:' prompt.	Code will display all user details using the toString method with an updated follower count from 300 to 500,	05/12/20	Passed	Emma Mason	
TCase_17	Check if when user enters option 3 on the Casual User Account menu, they will be given the option to enter three new integers to represent their three latest likes.	User is on the Casual User Account menu logged in with credentials 'e1' and 'p1'.	Enter '3' into the console on the Casual User Account menu. Enter '200' at the first prompt. '300' at the second and '250' at the third.	Code will print three prompts which will allow for entry of three integers and then print the average of these three integers – '250.00'.	05/12/20	Passed	Emma Mason	

TCase_18	Check that following TCase_17, the user account average likes has updated by printing the user details.	User is on the Casual User Account menu logged in with credentials ‘e1’ and ‘p1’ and complete steps in TCase_14.	Enter ‘1’ into the console at the ‘Provide your choice:’ prompt.	Code will display all user details using the toString method with an updated average likes to 250.00.	05/12/20	Passed	Emma Mason	
TCase_19	Check that when the user enters ‘4’ into the console on the Casual User Menu, that their user interaction calculation prints.	User is on the Casual User Account menu logged in with credentials ‘e1’ and ‘p1’ and complete steps in TCase_15 and TCase_17.	Enter ‘4’ into the console at the ‘Provide your choice:’ prompt.	Code will print ‘The percentage of followers that interact with your content is: 50.00%’	05/12/20	Passed	Emma Mason.	
TCase_20	Check that when the user enters ‘5’ into the console on the Casual User Menu, that a reach score prints and the program prompts the user for their number of weekly posts.	User is on the Casual User Account menu logged in with credentials ‘e1’ and ‘p1’.	Enter ‘5’ into the console at the ‘Provide your choice:’ prompt.	Code will print “Your reach score is:” and prompt user for their number of weekly posts.	05/12/20	Passed	Emma Mason	
TCase_21	Check that following the steps in TCase_20 the user can input an integer in the prompt for weekly posts.	User has pressed ‘4’ on the Casual User Account menu.	Enter ‘2’ into the console.	Code will print a statement asking the user to increase their posts to three a week and prompt for their number	06/12/20	Passed	Emma Mason	

				of daily Instagram stories.				
TCase_22	Check that following the steps in TCase_21 the user can input an integer in the prompt for daily Instagram stories.	User has pressed ‘4’ on the Casual User Account menu and entered ‘2’ for the number of weekly posts.	Enter ‘1’ into the console.	Code will print a statement asking user to post at least one story daily and what to use in these stories.	06/12/20	Passed	Emma Mason	
TCase_23	Check that when the user enters a number greater than 3 for posts per week, the program prints a tip for a time when the user should post.	User has pressed option 5 on the Casual User Menu and has entered an integer. The prompt for how many times they post per week should be uploaded.	Enter ‘6’ into the console.	Code will print a randomised tip on when to post.	06/12/20	Passed	Emma Mason	
TCase_24	Check that when the user enters 6 into the prompt on the Casual User Menu, they will be taken back to the initial menu.	User is on the Casual User Account menu logged in with credentials ‘e1’ and ‘p1’.	Enter ‘6’ into the console.	Code will bring user back to the initial main menu.	06/12/20	Failed	Emma Mason	The user was kept on the Casual User Account menu as when rearranging my menu options in the print statement, I changed “0.Logout” to “6.Logout” and did not change the switch statement accordingly.
TCase_25	Check that when the user enters 6 into the prompt on the	User is on the Casual User Account menu logged in with credentials ‘e1’ and ‘p1’.	Enter ‘6’ into the console.	Code will bring user back to the	06/12/20	Passed	Emma Mason	

	Casual User Menu, they will be taken back to the initial menu.			initial main menu.				
TCase_26	Check that when the user enters nothing to the console, an error message will print, and the prompt will be reprinted.	User runs the program and is on the initial menu.	Press the enter key with no input.	Code will print that it was an invalid input and to try again.	06/12/20	Passed	Emma Mason	
TCase_27	Check that when the user enters “null” to the console, the prompt will be reprinted.	User runs the program and is on the initial menu.	Enter “null” to the console and press enter.	Code will print the menu again.	06/12/20	Passed	Emma Mason	
TCase_28	Check that when the user enters nothing to the console on Casual User Account, an error message will print, and the prompt will be reprinted.	User logins into the system under the username “u2” and “p2”.	Press the enter key with no input.	Code will print that it was an invalid input and to try again.	06/12/20	Passed	Emma Mason	
TCase_29	Check that when the user enters “null” to the console, the prompt will be reprinted.	User logins into the system under the username “u2” and “p2”.	Enter “null” to the console and press enter.	Code will print the menu again.	06/12/20	Passed	Emma Mason	
TCase_30	Testing option 1 of Pro Influencer Account – Enter Last 3 Posts Likes	User is in the Pro Influencer Account Menu.	User enters 1 at the keyboard.	Prompted to enter likes of their latest 3 posts.	8/12/20	Passed	Fearghal O’Boyle	

TCase_31	Testing option 2 of Pro Influencer Account – Get Average Likes	User is in the Pro Influencer Account Menu and has completed option 1.	User enters 2 at the keyboard.	The average of the three previously entered values is printed to the terminal with contextual information.	8/12/20	Passed	Fearghal O'Boyle	
TCase_32	Testing option 3 of Pro Influencer Account – Rank Accounts Reached with Reach Tip.	User is in the Pro Influencer Account Menu.	User enters 3 at the keyboard, followed by a low value for accounts reached.	User prompted to enter number of accounts they reached this week. Rank is then displayed along with a reach tip.	8/12/20	Passed	Fearghal O'Boyle	
TCase_33	Testing option 3 of Pro Influencer Account – Rank Accounts Reached.	User is in the Pro Influencer Account Menu.	User enters 3 at the keyboard, followed by an average value for accounts reached.	User prompted to enter number of accounts they reached this week. Rank is then displayed.	8/12/20	Passed	Fearghal O'Boyle	
TCase_34	Testing option 3 of Pro Influencer Account – Rank Accounts Reached with prompt to enter good practice.	User is in the Pro Influencer Account Menu.	User enters 3 at the keyboard, followed by a high value for accounts reached.	User prompted to enter number of accounts they reached this week. Rank is then displayed, and user is prompted to enter what led them to such a high grade.	8/12/20	Passed	Fearghal O'Boyle	

TCase_35	Testing option 4 of Pro Influencer Account – Rank Interactions with Reach Tip.	User is in the Pro Influencer Account Menu.	User enters 4 at the keyboard, followed by a low value for interactions.	User prompted to enter number of interactions they had this week. Rank is then displayed along with a reach tip.	8/12/20	Passed	Fearghal O’Boyle	
TCase_36	Testing option 4 of Pro Influencer Account – Rank Interactions.	User is in the Pro Influencer Account Menu.	User enters 4 at the keyboard, followed by an average value for interactions.	User prompted to enter number of interactions they had this week. Rank is then displayed.	8/12/20	Passed	Fearghal O’Boyle	
TCase_37	Testing option 4 of Pro Influencer Account – Rank Interactions with prompt to enter good practice.	User is in the Pro Influencer Account Menu.	User enters 4 at the keyboard, followed by a high value for interactions.	User prompted to enter number of interactions they had this week. Rank is then displayed, and user is prompted to enter what led them to such a high grade.	8/12/20	Passed	Fearghal O’Boyle	
TCase_38	Testing option 5 of Pro Influencer Account – Rank Followers with Reach Tip.	User is in the Pro Influencer Account Menu.	User enters 5 at the keyboard, followed by a low value for followers.	User prompted to enter number of followers they had this week. Rank is then displayed	8/12/20	Passed	Fearghal O’Boyle	

				along with a reach tip.				
TCase_39	Testing option 5 of Pro Influencer Account – Rank Followers.	User is in the Pro Influencer Account Menu.	User enters 5 at the keyboard, followed by an average value for followers.	User prompted to enter number of followers they had this week. Rank is then displayed.	8/12/20	Passed	Fearghal O’Boyle	
TCase_40	Testing option 5 of Pro Influencer Account – Rank Followers with prompt to enter good practice.	User is in the Pro Influencer Account Menu.	User enters 5 at the keyboard, followed by a high value for followers.	User prompted to enter number of followers they had this week. Rank is then displayed, and user is prompted to enter what led them to such a high grade.	8/12/20	Passed	Fearghal O’Boyle	
TCase_41	Testing option 6 of Pro Influencer Account – Show Steps You Took to Achieve Great Reach Grades. If user has low grades.	User is in the Pro Influencer Account Menu and has completed options 3,4,5. User has low or average grades in all ranks.	User enters 6 at the keyboard.	Program prints that the user did not gain a high enough grade in any rankings to give reach tips.	8/12/20	Passed	Fearghal O’Boyle	
TCase_42	Testing option 6 of Pro Influencer Account – Show Steps You Took to Achieve Great Reach Grades. If user has high	User is in the Pro Influencer Account Menu and has completed options 3,4,5. User has a high grade in accounts reached.	User enters 6 at the keyboard.	Program prints users entered Reach Tip for Accounts Reached and that they did not have high	8/12/20	Passed	Fearghal O’Boyle	

	accounts reached grade.			enough rankings in Interactions and Followers to give a tip.				
TCase_43	Testing option 6 of Pro Influencer Account – Show Steps You Took to Achieve Great Reach Grades. If user has high Interactions grade.	User is in the Pro Influencer Account Menu and has completed options 3,4,5. User has a high grade in accounts reached and interactions.	User enters 6 at the keyboard.	Program prints users entered Reach Tip for Accounts Reached, Interactions and that they did not have high enough ranking in Followers to give a tip.	8/12/20	Passed	Fearghal O'Boyle	
TCase_44	Testing option 6 of Pro Influencer Account – Show Steps You Took to Achieve Great Reach Grades. If user has high Followers grade.	User is in the Pro Influencer Account Menu and has completed options 3,4,5. User has a high grade in accounts reached, interactions and followers.	User enters 6 at the keyboard.	Program prints users entered Reach Tip for Accounts Reached, Interactions and Followers.	8/12/20	Passed	Fearghal O'Boyle	
TCase_45	Testing option 7 of Pro Influencer Account – Get Reach Tip.	User is in the Pro Influencer Account Menu.	User enters 7 at the keyboard.	Program prints a random Reach Tip.	8/12/20	Passed	Fearghal O'Boyle	
TCase_46	Testing option 8 of Pro Influencer Account – Account Summary.	User is in the Pro Influencer Account Menu and has completed all other options within the menu.	User enters 8 at the keyboard.	Program prints a summary of the user's account – Reach Username, Instagram username,	8/12/20	Passed	Fearghal O'Boyle	

				followers, average likes, rank in accounts reached, interactions and followers.				
TCase_48	Testing option 9 of Pro Influencer Account – Log out.	User is in the Pro Influencer Account Menu.	User enters 9 at the keyboard.	Program exits the Pro Influencer Menu	8/12/20	Failed	Fearghal O'Boyle	ProUser_Defect1 Selecting option 9 which should terminate Pro Influencer Menu and return user to Main Menu not working. Checked switch statement and the case was "0" rather than "9".
TCase_49	Testing option 9 of Pro Influencer Account – Log out.	User is in the Pro Influencer Account Menu.	User enters 9 at the keyboard.	Program exits the Pro Influencer Menu	8/12/20	Passed	Fearghal O'Boyle	

Appendix C

Reach - User Guide

On Welcome Screen choose to 1. Login, 2. Register or 3. Exit.

Enter the relevant number for how you would like to proceed.

You will be prompted to provide your Username and Password to access your account.

Create an account by supplying a Username, Password, Your Instagram Handle and the Total Number of Your Followers.

You are asked how you use Instagram, for fun (Casual Influencer) or for business (Pro Influencer). Reach will be tailored to provide you with maximum benefit depending on what you select.

Casual Influencer

Once you login as a Casual Influencer you have a number of options to choose from including:

1. Print Your Details

2. Update Followers

3. Update Likes

4. Get User interaction

5. Receive Tips

6. Log Out

Select the option you would like, and the prompts will guide you through use of the program.

Pro Influencer

Once you login as a Pro Influencer you have several options to choose from including:

1. Enter Last 3 Posts Likes

2. Get Average Likes

3. Rank Accounts Reached

4. Rank Interactions

5. Rank Followers

6. Show Steps You Took to Achieve Great Reach Grades

7. Get a Reach Tip

8. Account Summary (View After All Options Completed)

9. Log Out

Select the option you would like and the prompts will guide you through use of the program.

Appendix D

Team Member Involvement

	Design	Code	Test	Demo	Presentation
Donal Doherty	Came up with Reach idea. Provided Instagram insight data. Class diagram.	User class. User interface. Method in read class.	Tested overall completed code for user readability and friendliness.	Initial user menu demo. Registration of new user.	Requirements Future improvements What is Reach?
Fearghal O'Boyle	Came up with Reach idea. Classes. Program preparation sheets. Class diagram.	User class. Pro class. User interface Method in user registration.	TCase_30-49. (See Appendix A)	Pro user menu. Login as pro user and demonstrate options.	Development methodology Concept Final Product
Emma Mason	Program preparation sheets Class diagram User case diagram	Casual class. Read class. LoginTransaction class RegistrationTransation class.	TCase_1-29. (See Appendix A)	Casual user menu. Login as casual user and demonstrate options.	Design Testing Member Involvement