

Faculty of Engineering and Information Technology

School of Computer Science

31927 - Applications Development with .NET
32998 - .NET Applications Development

SPRING 20120

ASSIGNMENT - 1 SPECIFICATION

Due date	Monday 11:59pm, 21st September 2020
Demonstrations	Required in the lab/tutorial session
Marks	35% of the total marks for this subject
Submission	Complete project folder zip (Code, solution files, etc.), any instructions to run the program in a text file, all in 1 single zip file.
Submit to	Canvas assignment submission
Note: This assignment is individual work.	

Summary

This assessment requires you to develop a Simple Bank Management Systems using C#. Bank account details, user data, banking transactions etc. are to be stored in files. It has to be a C# console application. The specification/requirements are detailed in the rest of the document.

Students need to submit the complete project folder in zip format, which will have the complete C# code, solution file, data files etc. required to run/test the program. Any special instructions required to run the code has to be provided in a text file.

Assignment Objectives

The purpose of this assignment is to demonstrate competence in the following skills.

- ☐ Ensure firm understanding of the .Net framework, C# basic and syntax
- ☐ Understand how the .NET framework implements OO concepts and the implications this has for new language design
- ☐ Array and string manipulation
- ☐ Creating custom classes and methods in C#
- ☐ File operations and handing in C#
- ☐ Creating interactive console applications
- ☐ Create good OO design.

Tasks:

In this assignment you need to develop a menu driven Simple Bank Management Systems using C#. Bank account details, user data, banking transactions etc. has to be stored in files. The components/menu items described below that should be in the screen/console, but you are free to add more options as appropriate based on your application design. It should be a console app.

1. Login Menu:

Functionality: Provide secured access to the banking system.

Input fields required:

- *Username* : Display the typed characters as it is
- *Password* : Display "*" instead of the actual characters

Field checks required:

1. Username and Password to be cross checked with the valid Credentials available in the file to store the login details (**login.txt**)
2. Username should be unique

Display appropriate error message if the credentials are invalid and allow to re-enter the values.

Sample Console Screen for login:

```
WELCOME TO SIMPLE BANKING SYSTEM

LOGIN TO START

User Name: nabin
Password: *****

Valid credentials!... Please enter
```

2. Main Menu:

Functionality: Used to navigate through the system using certain options related to common banking operations.

Menu item required:

1. **Create a new account:** To create a new account and store it in the respective account file
2. **Search for an account:** Search for an account using account number
3. **Deposit:** Deposit money in a valid account
4. **Withdraw:** Withdraw a valid amount for an account
5. **A/C statement:** Display account statement for a valid account number
6. **Delete account:** Delete the account
7. **Exit:** Exit from the application or return back to the login screen
8. Ask user to select an item number(1-7) from the menu (See sample screen)

Field checks required:

- The item number selected should be an integer
- Check for a valid input (1-7) else return back to the menu without any error.
- Check for a non-integer input and return back to the menu without any error.

Sample screen for Main Menu

```
WELCOME TO SIMPLE BANKING SYSTEM

1. Create a new account
2. Search for an account
3. Deposit
4. Withdraw
5. A/C statement
6. Delete account
7. Exit

Enter your choice (1-7):
```

3. Create a new account:

Functionality: Create a new account in the banking system

Input fields required:

- *First Name* : Text field
- *Last Name*: Text field
- *Address*: Text field
- *Phone*: Integer
- *Email*: Text field

Field checks required:

1. Phone: Integer, should not be more than 10 characters
2. Email: Check if the entered string has "@" (required), "gmail.com", "outlook.com", and "uts.edu.au" in the domain (optional)

If the information was correct in the input fields:

- a. Create a file with the name **<account_number>.txt** and save all the information in a proper format, which makes it easy to retrieve.

Sample screen for Account Creation:

```
CREATE A NEW ACCOUNT

ENTER THE DETAILS

First Name:
Last Name:
Address:
Phone:
Email:
```

Sample screen after account creation:

```
CREATE A NEW ACCOUNT

ENTER THE DETAILS

First Name: Nabin
Last Name: Sharma
Address: UTS, Building-11, Sydney
Phone: 951411835
Email: Nabin@uts.edu.au

Is the information correct (y/n)?

Account Created! details will be provided via email.
Account number is: 100001
```

- b. Generate a unique account number (6-8 digits) and display it.
- c. Email the account details to the email ID provided.
- d. Return to the Main menu upon-key press.

4. Search for an account:

Functionality: Search for a valid account and display the account details if an account is found

Input fields required:

- *Account Number:* integer

Field checks required:

1. *Account number:* Integer, should not more than 10 characters.
If an invalid account number is provided, appropriate error message should be provide, with an option to re-enter/check another account.
If the account number is valid, display the account details similar to as shown in the sample screen.

Once the search is complete, return to the Main Menu.

Sample screen for account search

```

SEARCH AN ACCOUNT
-----
ENTER THE DETAILS
Account Number:
  
```

Sample screen for invalid account

```

SEARCH AN ACCOUNT
-----
ENTER THE DETAILS
Account Number: 1001414

Account not found!
Check another account (y/n)?
  
```

Sample screen for valid account

```

SEARCH AN ACCOUNT
-----
ENTER THE DETAILS
Account Number: 100001

Account found!

ACCOUNT DETAILS
-----
Account No: 100001
Account Balance: $51000
First Name: Nabin
Last Name: Sharma
Address: UTS, Broadway, NSW 2007
Phone: 0450977815
Email: Nabin.Sharma@uts.edu.au

Check another account (y/n)?n
  
```

5. Deposit:

Functionality: Search for a valid account and deposit the provided amount in the account.

Input fields required:

- *Account Number:* integer

Field checks required:

1. *Account Number:* Integer, should not more than 10 characters.
If an invalid account number is provided, appropriate error message should be display, with an option to re-enter/check another account.

If the account number is valid allow user to enter the amount to deposit. Update the account balance and update the information in the files for the account. Once the deposit is successful, return to the Main Menu.

Sample screen for Deposit option:

```

DEPOSIT
-----
ENTER THE DETAILS
Account Number: 100001
Amount: $

Account found! Enter the amount...
  
```

Sample screen for deposit successful

```

DEPOSIT
-----
ENTER THE DETAILS
Account Number: 100001
Amount: $2000

Account found! Enter the amount...
Deposit successful!
  
```

Sample screen for account not found error.

```

DEPOSIT
-----
ENTER THE DETAILS
Account Number: 100004
Amount: $

Account not found!
Retry (y/n)?y
  
```

6. Withdrawal:

Functionality: Search for a valid account and withdraw the provided amount from the account.

Input fields required:

- *Account Number:* integer

Field checks required:

1. *Account number:* Integer, should not more than 10 characters.
If an invalid account number is provided, appropriate error message should be displayed, with an option to re-enter/check another account.

If the account number is valid allow user to enter the amount to withdraw. If the withdrawal amount is less the balance, display appropriate error message. Update the account balance after withdrawal and in the files for the account. Once the withdrawal is successful, return to the Main Menu.

7. Account Statement:

Functionality: Search for a valid account, display and email the statement to the email address provided in the account.

Input field required:

- *Account Number:* integer

Field checks required:

1. *Account number:* Integer, should not more than 10 characters.
If an invalid account number is provided, appropriate error message should be displayed, with an option to re-enter/check another account.

If the account number is valid: display the account statement with last five transactions. *Email the statement to the user based on the preference.*

Once the statement is successfully generated, return to the Main Menu.

8. Delete and Account

Functionality: Search for a valid account, display the account details, and delete an account.

Input field required:

- *Account Number:* integer

Field checks required:

1. *Account Number:* Integer, should not more than 10 characters.
If an invalid account number is provided, appropriate error message should be displayed, with an option to re-enter/check another account.

If the account number is valid: display the account details and delete the account based on the user preference. Return to the main menu upon deletion.

9. Exit in Main Menu:

Functionality: exit from the simple banking system program.

Sample screen for withdraw option

```
WITHDRAW
-----
ENTER THE DETAILS
Account Number: 100001
Amount: $
Account found! Enter the amount...
```

Sample screen for successful withdraw

```
WITHDRAW
-----
ENTER THE DETAILS
Account Number: 100001
Amount: $400
Account found! Enter the amount...
Withdraw successful!
```

Sample screen for statement option

```
STATEMENT
-----
ENTER THE DETAILS
Account Number:
```

Sample screen for statement display

```
STATEMENT
-----
ENTER THE DETAILS
Account Number: 100001
Account found! The statement is displayed below...
SIMPLE BANKING SYSTEM
Account Statement
Account No: 100001
Account Balance: $1000
First Name: Nabin
Last Name: Sharma
Address: UTS, Broadway, NSW 2007
Phone: 0450977815
Email: Nabin.Sharma@uts.edu.au
Email statement (y/n)?y
Email sent successfully!...
```

Sample screen for delete account option

```
DELETE AN ACCOUNT
-----
ENTER THE DETAILS
Account Number: 100001
```

Sample screen after delete operation

```
DELETE AN ACCOUNT
-----
ENTER THE DETAILS
Account Number: 100001
Account found! Details displayed below...
ACCOUNT DETAILS
Account No: 100001
Account Balance: $1000
First Name: Nabin
Last Name: Sharma
Address: UTS, Broadway, NSW 2007
Phone: 0450977815
Email: Nabin.Sharma@uts.edu.au
Delete (y/n)?y
Account Deleted!...
```

Additional requirements:

1. All login related data to be stored in the **login.txt**
2. All user account data to be stored in the **<account_number>.txt** file. There should be one file per account in the system. E.g: 100001.txt
3. The console should be interactive and easy to use.

Note: The sample console screens provided are just for reference only.

Additional Information:

Assessment Submission

You must upload a zip file of the C# project folder, data files and the solution file to Canvas. **This must be done by the Due Date.** You may submit as many times as you like until the due date. The final submission you make is the one that will be marked. If you have not uploaded your zip file within 7 days of the Due Date, or it cannot be run in the lab, then your assignment will receive a zero mark.

PLEASE NOTE 1: It is your responsibility to make sure you have thoroughly tested your program to make sure it is working correctly.

PLEASE NOTE 2: Your final submission to Canvas is the one that is marked. It does not matter if earlier submissions were working; they will be ignored. Download your submission from Canvas and test it thoroughly in your assigned laboratory.

Return of Assessed Assignment

It is expected that marks will be made available 2 weeks after the submission via Canvas.

Queries

If you have a problem such as illness which will affect your assignment submission contact the subject coordinator as soon as possible.

Dr. Nabin Sharma
Room: CB11.07.124
Phone: 9514 1835
Email: Nabin.Sharma@uts.edu.au

If you have a question about the assignment, please post it to the Canvas discussion board for this subject so that everyone can see the response.

If serious problems are discovered in assignment specification the class will be informed via an announcement on Canvas. It is your responsibility to make sure you frequently check Canvas.

PLEASE NOTE : If the answer to your questions can be found directly in any of the following

- ☐ Subject outline
- ☐ Assignment specification
- ☐ Canvas FAQ
- ☐ Canvas discussion board

You will be directed to these locations rather than given a direct answer.

Extensions and Special Consideration

Please refer to subject outline.

Academic Standards and Late Penalties

Please refer to subject outline.