**Assessment 1**

Complete the following tasks using a local Git server & GitHub to handle the version control in your development. Before this Assessment 1, you must have a GitHub account.

Write the user name of your GitHub account here: Wyatt Coff

*Note: When submitting:*

* *Use a word document named* ***your\_name\_assessmet1.docx*** *to store* the screen shots for each task described below. Clearly label which Task number you are attempting, and the name of the screen shot.
* Also, submit the C# program that you have developed with the additional features (specified in the following tasks) using the Git version control system.

**Task 1.** Create a repository and add a collaborator to the repository.

In your GitHub account, create a public repository called “**TafeProductApp**”. In the Description enter “**Tafe product Application**”.

Add the lecturer (e.g. **tafesakt**) as the collaborator in your repository.

Your submission should capture and submit the following screen shots:

* The GitHub repository has been created.

A screenshot of a computer

Description automatically generated

* The Screen shot must show the collaborator in the repository.

A screenshot of a computer

Description automatically generated

**Task 2.** Add the following files into the remote repository.

* Add the **.gitignore** file with the appropriate *ignore pattern* for the C# .net application. Upload and commit this file into the GitHub repository. The file can be found from the student assessment files.
* Add an empty **README.md** file to the repository. Add the following line to the content - “**This application calculates the total price of the product.**” Commit this file into the GitHub repository.
* Add a **LICENSE.md** file to the repository with the contents given in the student files. Enter your name in the content. Commit this file into the GitHub repository.
* Show the files under the <code> in the GitHub.

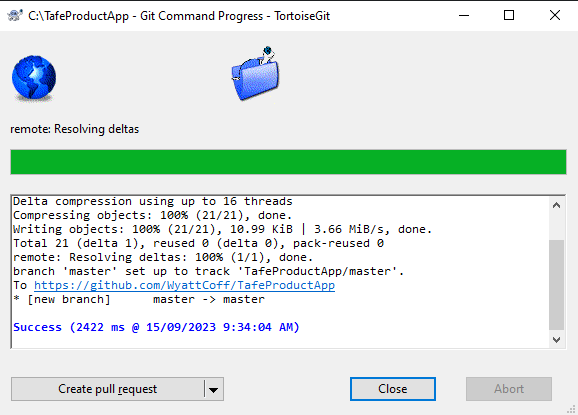
Your submission should capture the following screen shots:

* A screen shot must show that the README.md, LICENSE.md and .gitignore files have been committed to the repository.

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

**Task 3.** Configure TortoiseGit as the client tool.

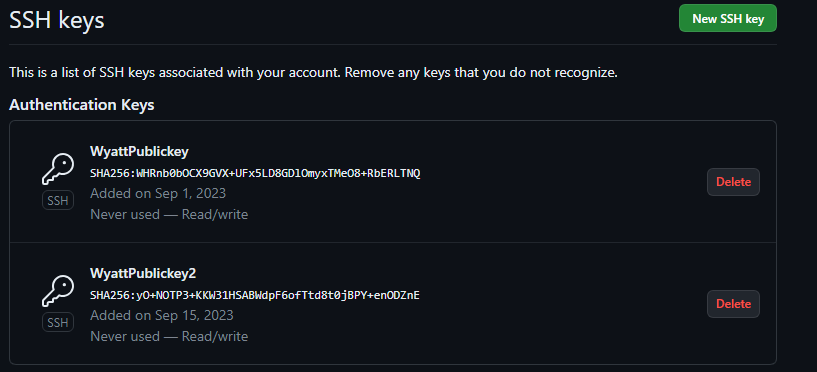
Use PuttyGen to create a SSH private and public key pair.

* Save the private key with the name “**PM\_Private.ppk**” in the c:\temp folder.

A screenshot of a computer

Description automatically generated

* Add the public key into the GitHub of your account.



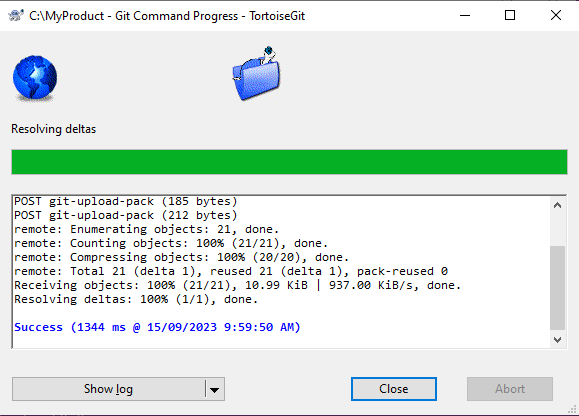
Your submission should capture the following screen shots:

* A screen shot must show the private putty key (.ppk file) generated in the C:\temp folder.
* A screen shot must show the public key uploaded to the GitHub.

**Task 4.** Clone a repository.

Create a local working directory called “**C:\MyProduct**”. Clone the remote repository to the local working directory.

* Copy the “**ProductMaintenance**” program from the given student assesment files which can be found from LEARN into local repository. (See Fig. 1)



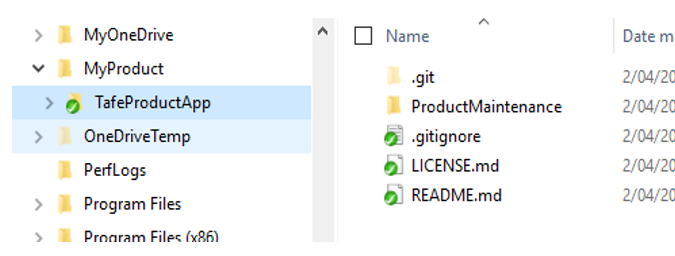


Fig 1. Screen showing ProductMaintenance program copied to the local repository

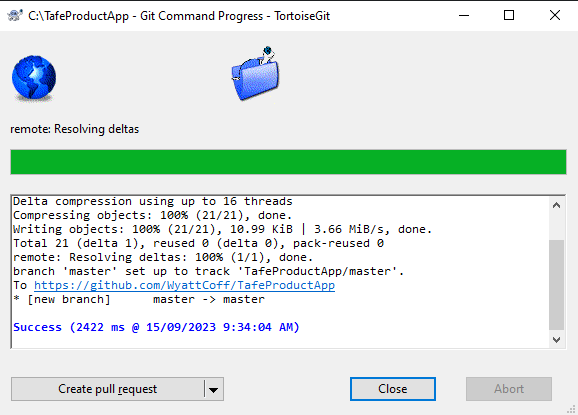
* Stage and commit the addition files to the **local repository** with message “**Commit the Product Maintenance program to the local repository**”.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

* Use the ‘show log option’, write down the commit SHA hash index (i.e. the first 7 hexadecimal digit will do) here: 3e9d45a
* In TortoistGit, push the local changes to the remote repository.



Your submission should capture the following screen shots:

* A screen shot must show the revision number. (i.e. SHA hash index) in the show log option.

A screenshot of a computer

Description automatically generated

* A screen shot must show the pushing of the local change appears in the remote repository.

A screenshot of a computer

Description automatically generated

**Task 5.** Develop and build application using then VCS.

Open the project, on the **MainWindow -** you are required to add a label and a text block to display the **Total Charge** after **Adding the delivery charge ($25.00)**. (See Fig. 2 below)

i.e. **Total Charge = TotalPayment + $25.00**

***Note: &#xA;*** *means new line in the XAML label.*

* Add a label with Content: **Total Charge after** **adding &#xA;the delivery charge ($25)**
* Add a textblock “**totalChargeTextBox**” to output the total charge of payment with delivery charge on screen.
* Modify the code to calculate the total charge so that it will include the delivery charge.

.

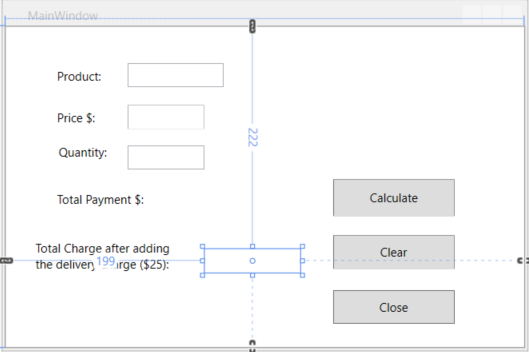
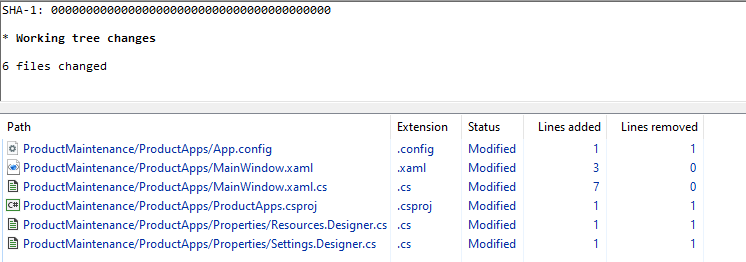


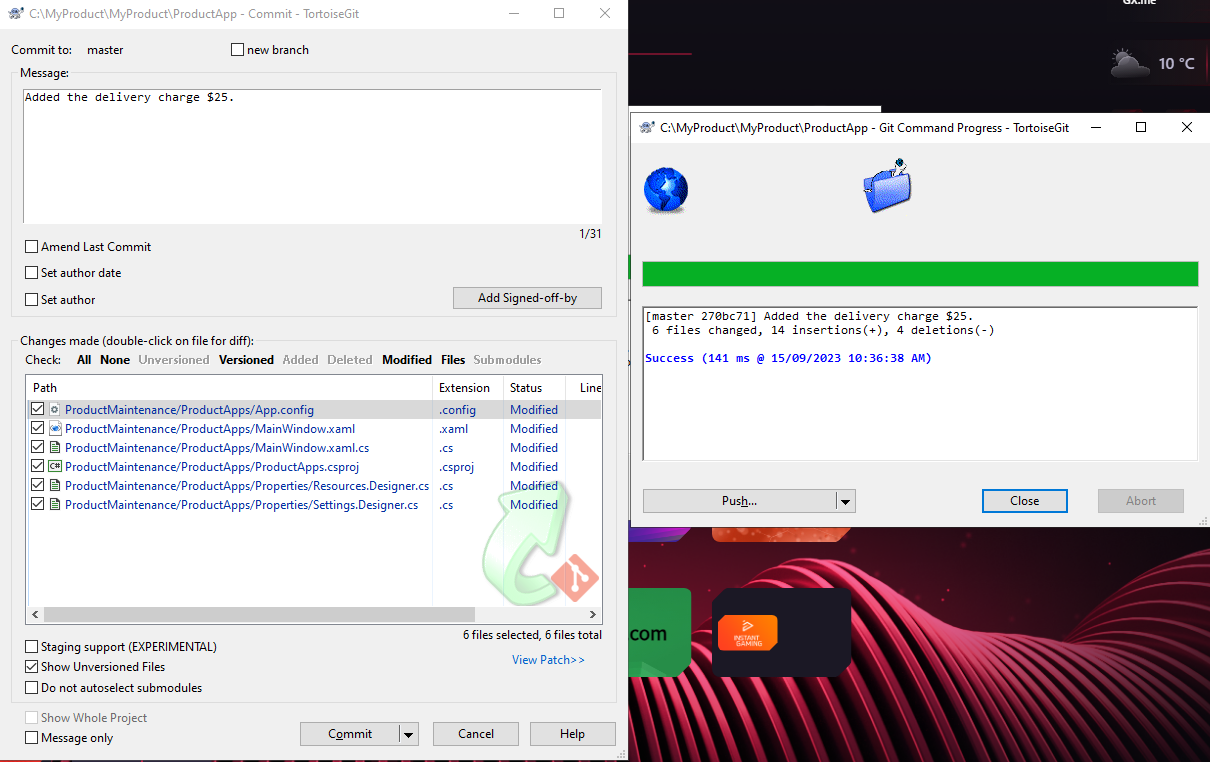
Fig 2. Screen display with Total Charge

* After re-building the solution successfully. Save the solution.
* Capture a screen shot to show the **show log** option, and note below the files that have been changed.

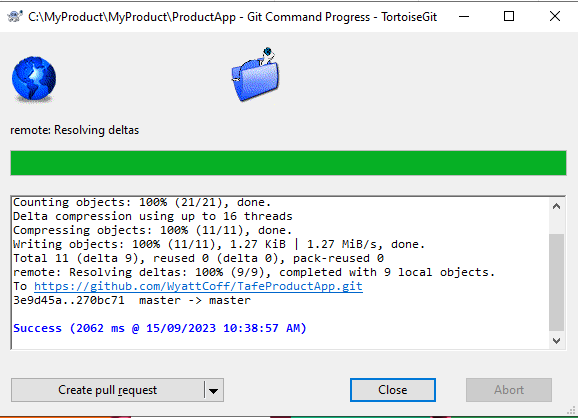


1. App.config
2. MainWindow.xaml
3. MainWindow.xaml.cs
4. ProductApps.csproj
5. Resources.Designer.cs
6. Settings.Designer.cs

* Commit the changes to the local repository and add the message as “**Added the delivery charge $25”.**



* Use the show log, write down the commit SHA hash index (i.e. write down the first 7 hexadecimal digit will do) here: 270bc71
* In TortoiseGit, push the local changes to the remote repository.



When submitting this Task, capture the following screen shots:

* A screen shot must show those files being changed.

A computer screen with a screen and a computer screen

Description automatically generated with medium confidence

* A screen shot must show the execution of the code with total charge calculated after the $25 delivery charge has been added and committed to the repository.

A screenshot of a computer

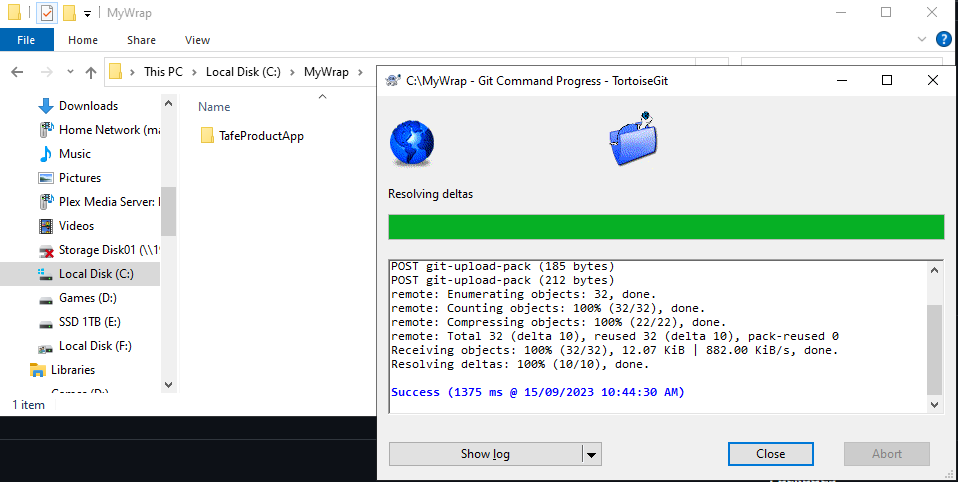
Description automatically generated

* A screen shot must show the revision no. (i.e. SHA hash index) in the show log option.
* A screen shot must show the pushing of the local changes to the remote repository successfully.

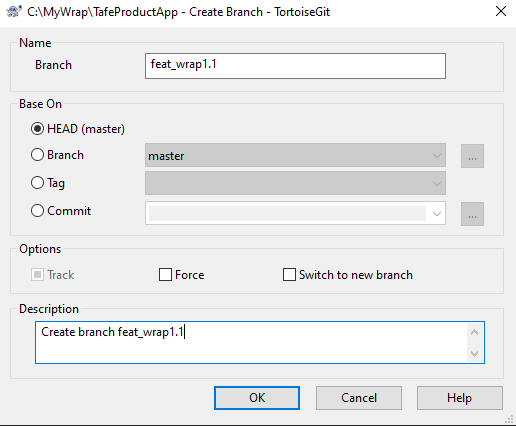
**Task 6.** Create a branch for developing a feature

Now, the system has a **feat\_wrap1.1** which has been approved by the project manager. The user likes to incorporate the calculation of **wrapping cost (i.e. additional $5.00)** as well. You are required to develop this in the feature branch. To do this, you need to:

* Create a working directory called “**C:\MyWrap”.** Clone the remote repository to the form a local repository.



* Create a branch called **feat\_wrap1.1** based on the **Head** of the **master (or main)** branch. Add a message as “**Create branch feat\_wrap1.1**”.



* In the local repository, checkout the **feat\_wrap1.1**. Modify the code to calculate the wrapping charge (i.e. add another $5.00). (See Fig 3)

A screenshot of a computer

Description automatically generated

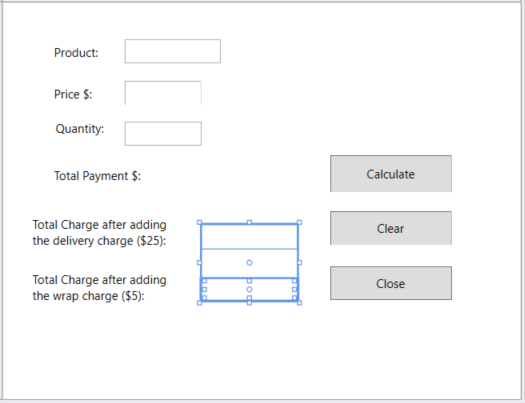
`

Fig 3. Screen display with the Total Charge after the wrapping charge

A computer screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Total Charge after wrap = TotalPayment + $25.00 + $5.00**

* Rebuild code successfully and save it. Commit the code to the **feat\_wrap1.1** branch. Add message as “**Commit the branch feat\_wrap1.1**”.

A screenshot of a computer

Description automatically generated

* Use the show log, write down the commit SHA hash index (i.e. write down the first 7 hexadecimal digit will do) here: 5706a78

A screenshot of a computer

Description automatically generated

* **Push all branch** the local feature changes to the remote repository. (See. Fig 4)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* Show the revision graph with the checkout of the head revision of the feat\_wrap1.1 branch. (See Fig 5)

A diagram of a program

Description automatically generated

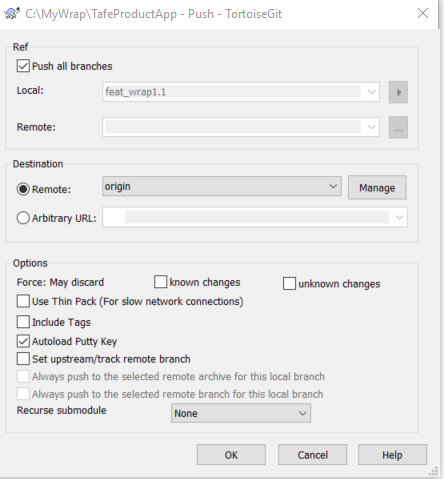


Fig 4. Screen setting for pushing branches to remote repository

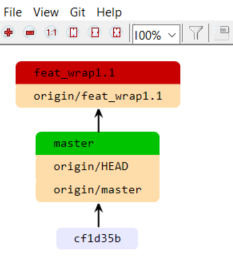


Fig 5 example of revision graph

Your submission should, capture the following screen shots:

* A screen shot must show the execution of the code with total charge after wrap calculation ($5.0) has been added and committed to the repository.
* A screen shot must show the revision no. (i.e. SHA hash index) in the show log option.
* A screen shot must show the pushing of the local change to the remote repository successfully.
* A screen shot must show the checkout of the head revision of the feat\_wrap1.1 branch.

**Task 7.** Create branch based on the previous branch in Task 6

The requirement **feat\_GST1.2** has added further functionality to include a GST calculation based on the **feat\_wrap1.1.**

***Note:*** *Make sure that your Task 6 changes have been committed to the local repository, as well as to the remote GitHub repository.*

* Create a new working directory “**C:\MyGST”** and clone the remote repository.

A screenshot of a computer

Description automatically generated

* In the local repository, checkout the **feat\_wrap1.1** branch and use it to base your further development. (See Fig 6)

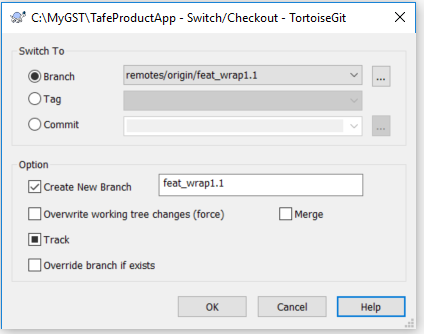


Fig 6. Checkout before creating a branch

* Create a new branch called “**feat\_GST1.2**”. Add a message “**Create a branch feat\_GST1.2**”. (See Fig 7)

A screenshot of a computer

Description automatically generated

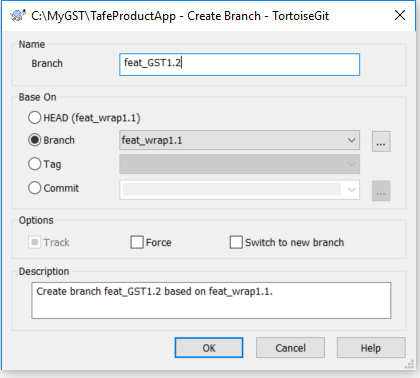


Fig 7. Create a feature branch

* + Checkout the local repository to **feat\_GST1.2**. (See Fig 8.)

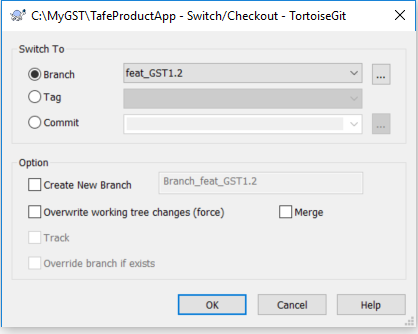


Fig 8. Checkout the branch with the head revision.

Further develop the application to incorporate the GST feature**.**

Add labels to display “**Total Charge After GST (@10%”.** You need to modify the code to suit the design.

i.e. **Total Charge After GST@10%** = (**Total payment + $25.00 + $5.00) \* 1.1**

A screenshot of a computer

Description automatically generated

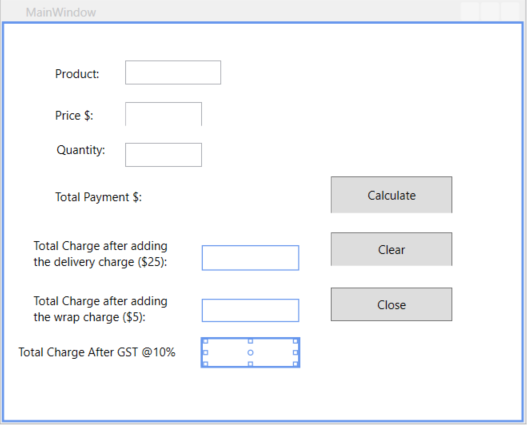


Fig 9. Screen display with the Total Charge after the GST.

* Save the code if rebuild successful.
* Commit the code to the **feat\_GST1.2** branch. Add message as “**Commit the branch feat\_GST1.2**”.

A screenshot of a computer

Description automatically generated

* Push the local **feat\_GST1.2** branch to the remote repository.

A screenshot of a computer

Description automatically generated

Your submission should capture the following screen shots:

* A screen shot must show the results of the execution of the code with a total charge after GST calculation (\*1.1) that has been added and committed to the repository.
* A screen shot must show the revision number (i.e. SHA hash index) in the show log option.
* 5b89079

A screenshot of a computer

Description automatically generated

* A screen shot must show the pushing of the local change to the remote repository successfully.
* A screen shot must show the checkout of the head revision of the feat\_wrap1.2 branch.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Task 8**. Merge the branch development back to the master (or main) branch.

* Create a local working directory “C:\MyMerge”.
* Clone the remote repository.

A screenshot of a computer

Description automatically generated

* Checkout the head revision of the master branch. (See Fig. 10)

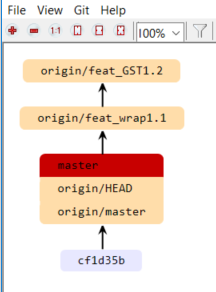


Fig 10. Revision graph showing the checkout of the head revision of the master (or main) branch.

My Revision Graph:

A diagram of a code

Description automatically generated

* Choose **TortoiseGit Merge**. In the **From Branch**: choose **remote/origin/feat\_GST1.2**. (See Fig 11)

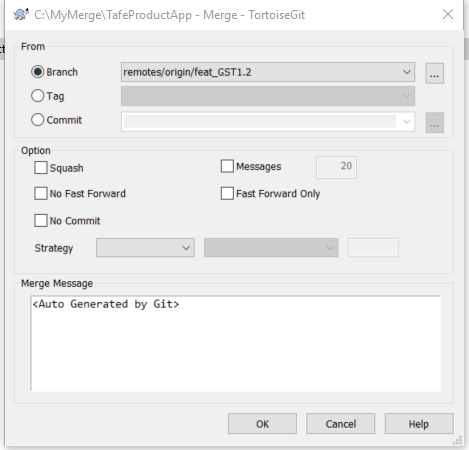


Fig 11. Merge feature branch to master branch

Capture the screen shot of the merged message.

A screenshot of a computer

Description automatically generated

* Do git sync and push to the remote repository. (See Fig. 12 & 13)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

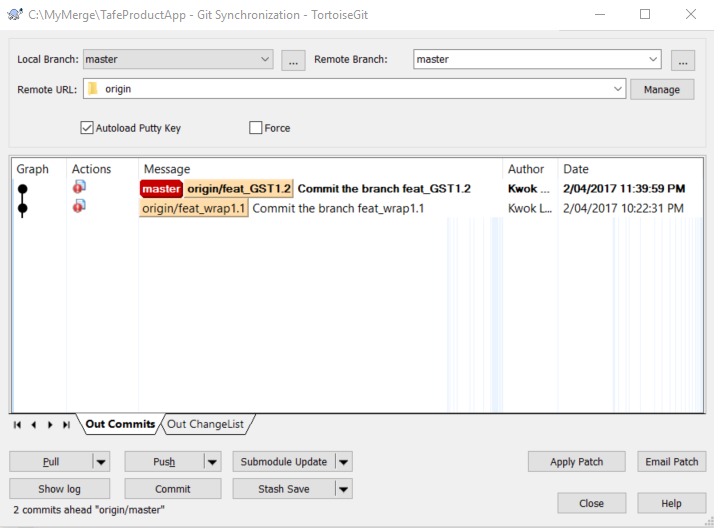


Fig 12. Git Sync setting

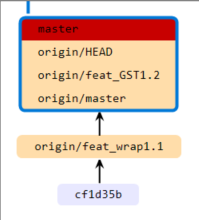


Fig 13. Revision graph showing the head revision of the remote master.

My Revision graph.

A diagram of a computer code

Description automatically generated

Find out the revision numbers. (note: write down the first 7 hexadecimal will do) and record them in the following space:

|  |  |
| --- | --- |
| Committed feat\_wrap1.1 | 5706a78 |
| Committed feat\_GST1.2 | 5b89079 |

Your submission should capture the following screen shots:

* A screen shot must show the merging has been completed successfully.
* A screen shot must show the revision numbers. for the branch **feat\_wrap1.1** and **feat\_GST1.2** using the show log option.

**Task 9.** Create serial patches.

The patches created will be based on the code starting from the **Added the delivery charges $ 25** to the code finished in the **feat\_GST1.2.**

* Write down the revision number of **the Added the delivery Charge $25** and **the feat\_GST1.2** in the following space:

|  |  |
| --- | --- |
| Committed Added the delivery charge $25. | 270bc7152f95120456c574a03032cea96e104a29 |
| Committed feat\_GST1.2 | 5b89079bccd16f8d8bb61c06dc5970226db1bd21 |

* In the MyMerge directory, switch to the committed revision of **Added the delivery charges $ 25.** (See Fig 14)

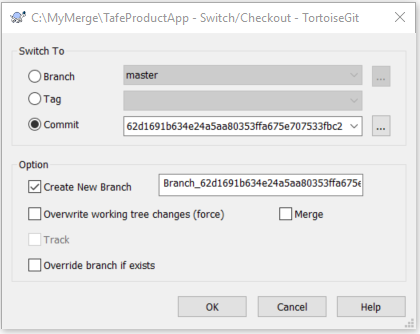


Fig 14. Check out the base version for creating patch

A screenshot of a computer

Description automatically generated

A screenshot of a computer

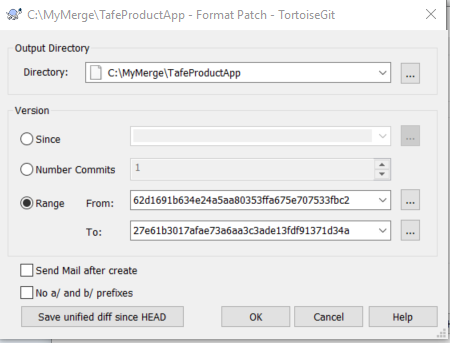
Description automatically generated

* Generate a patch: (See Fig 15)

**From:** **the Added the delivery Charge $25**

**To:** **the feat\_GST1.2**

Note: Pay attention to where the patches are generated. You need to submit this working directory with the serial patches to LEARN for marking.



Use your own revision numbers.

Fig 15. Marking the range of revision numbers in creating patch

A screenshot of a computer

Description automatically generated

Your submission for this task should capture the following screen shots:

* A screen shot must show the revision number of **the Added the delivery Charge $25** and **the feat\_GST1.2**
* A screen shot must show the serial patch files have been created in the directory.

A screenshot of a computer

Description automatically generated

**Submission requirements:**

Upload a zip file to LEARN with the following items:

* A Word document with the screen shots and your answer for each task.
* The zipped C# code in the MyMerge directory.
* The generated patch files.