**Software Development and Testing IT7320**

**LAB WORK**

Team Members: Submitted to:

Anjali Khandelwal Chalinor Baliuag

Kwinno Pineda

Ralph Saplan

Thanh Huynah

# Surveying the GitHub Platform (Kwinno)

## Difference between the Git DVCS and GitHub

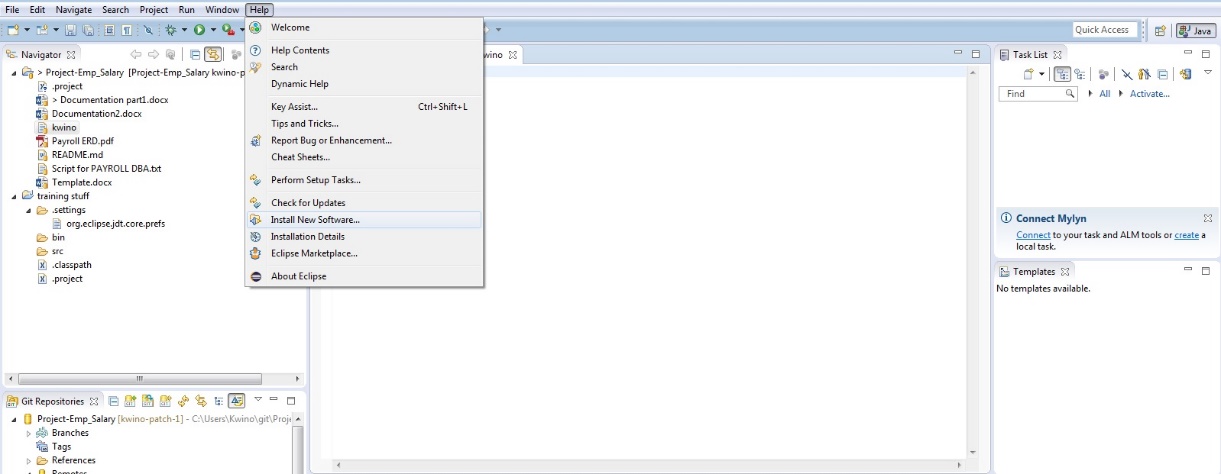
Git DVCS (Distributed Revision Control System) uses GitBash, a **command-line tool** to connect to your work locally and its working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server. (Wikipedia, 2015)[[1]](#footnote-1).

While on the other hand Github, is a web based Git repository hosting system which is free and open source used commonly with team collaborating with a project. It is a **web based graphical interface system** and desktop as well as mobile integration. It also provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project. (Wikipedia, 2015)[[2]](#footnote-2). Github also provides private account which needed to pay only few people used it. (Wikipedia, 2015). Github use can be used remotely using Gitbash or other application (Netbeans, Eclipse and etc.)

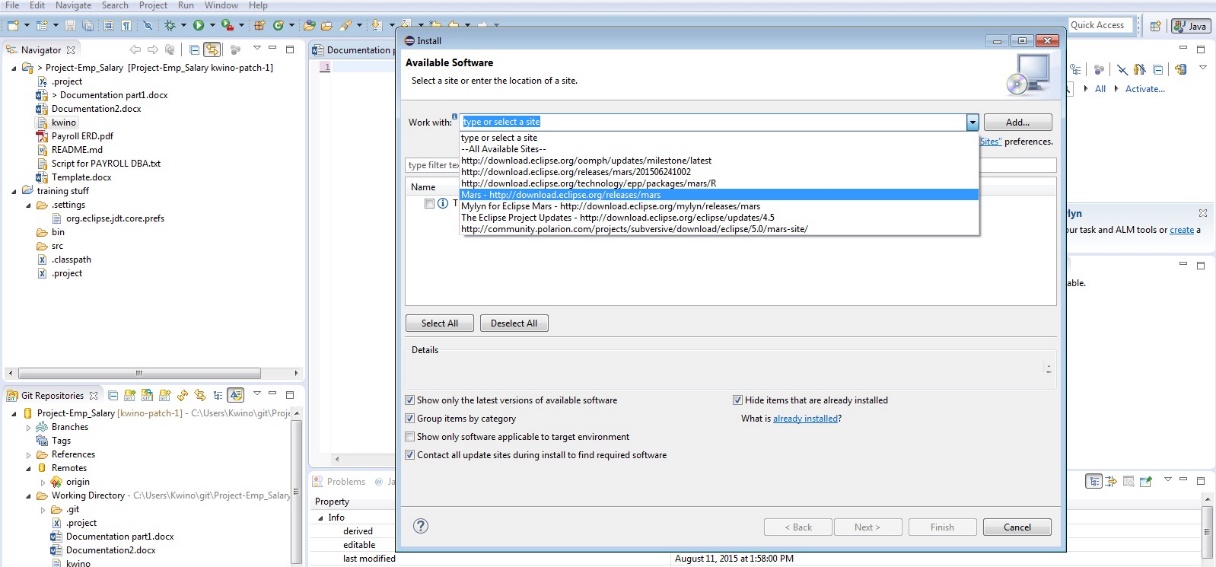
## Access the common project components of daily GitHub interactions

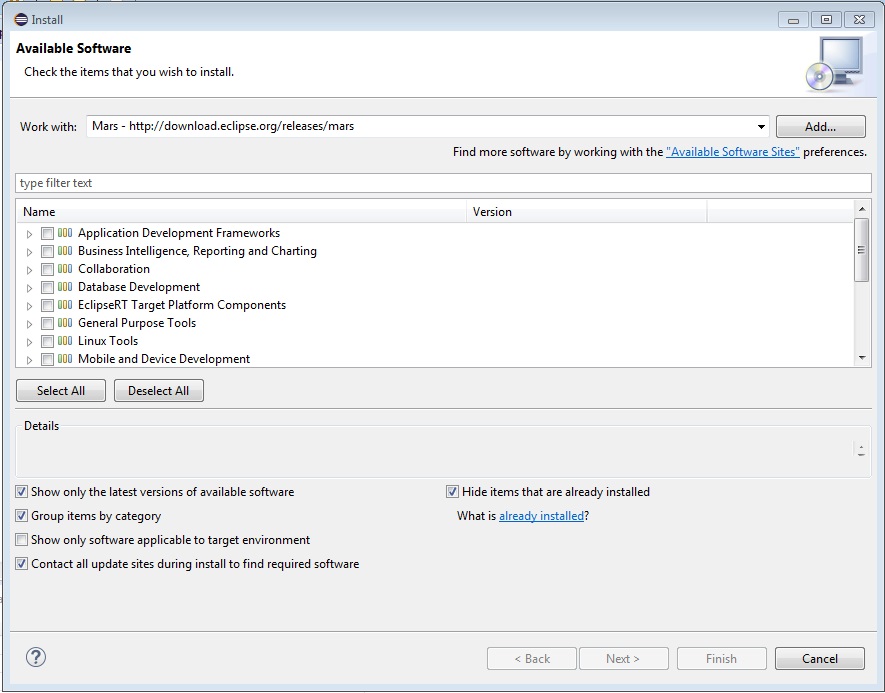
There are many ways to connect a project to Github, one of the way is using Eclipse. First install the Eclipse Mars Java IDE then after install, open eclipse and

* Go to Help menu.
* Click Install New Software.



* Choose <http://download.eclipse.org/releases/mars/> to download Github plugin
* Click Collaboration and Install all the Github plugins under Collaboration.

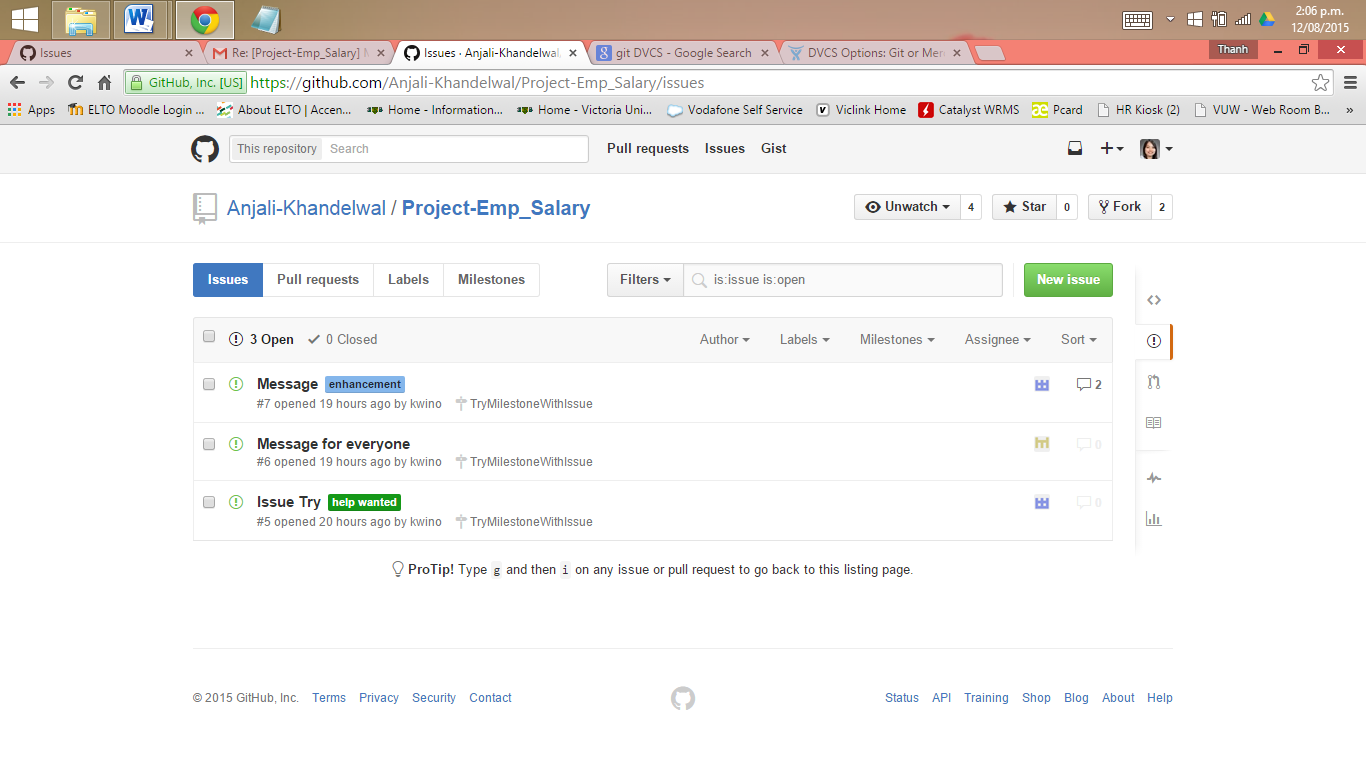




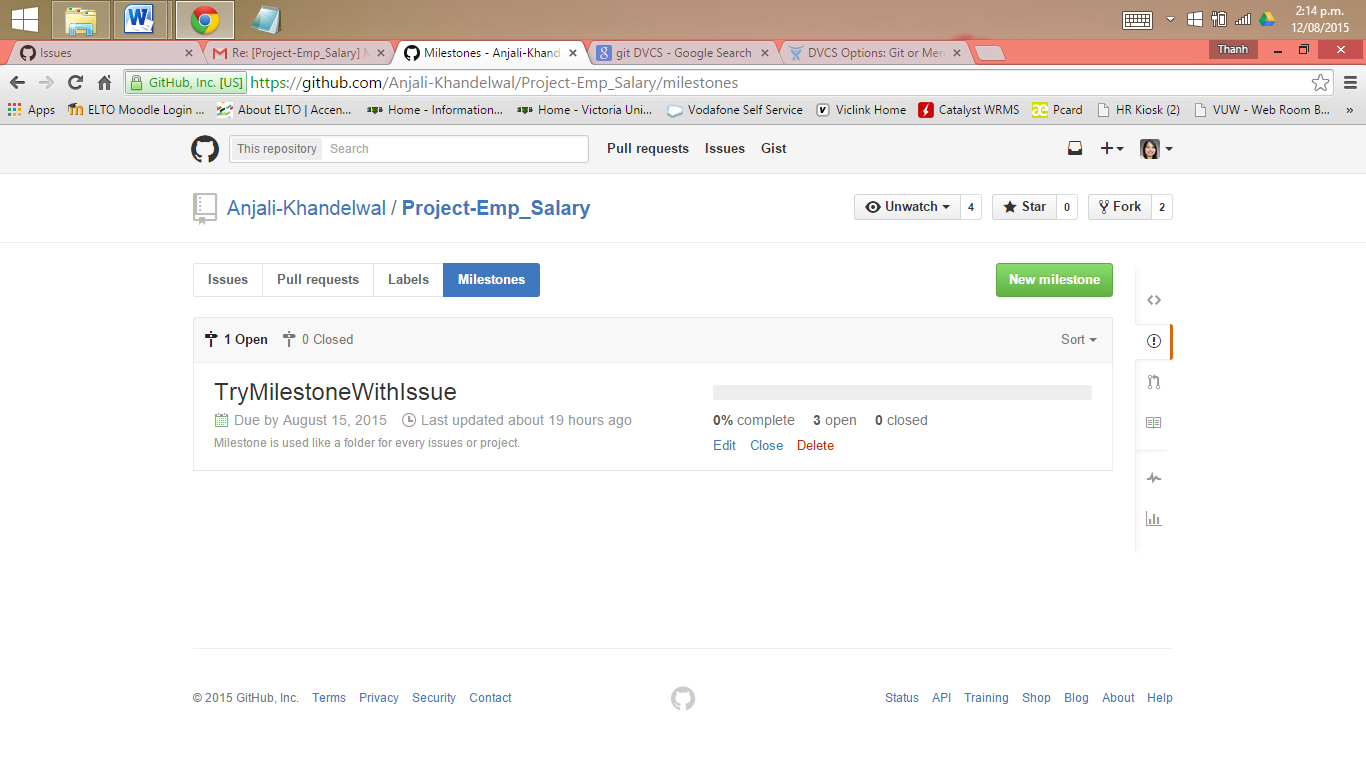
* Then Go To **File Menu** select import and copy the clone in Github paste it in the URI text box. Next is select the repository you want to fetch from Github to your local computer. And select as **Import as general project**.

## Utilize project management components (Issues, Milestones, Collaborators and Teams)

**Issues** is a feature of Github as an integrated bug and enhancement tracker where all the things such as request, suggestion and etc. are being post as message and discuss with the group. In **issues** you can assign to a specific member and put a **labels** and add color if it is a duplicate, bug, enhancement and others or make your own label to recognize what is the message all about.



You can create a **Milestones** for your issues and set the due date of your project to monitor. Creating milestone for every project is a great help all issue are being sorted or group by according to what project you are working on and all open, close of issue and completeness of the project were being recorded to display. Members of a team with issues regarding the project will be posted



In Github, a repository owned by a user account has 2 kinds of permission level (Github, n.d.). One is **repository owner** and **collaborators**. **Repository owner** has:

* all the permission
* full control of the repository
* owner can add collaborators
* change the visibility of the repository(from public to private or from private to public), and delete repository

. There is only one owner of a repository owned by a user account.

**Collaborators** are users who are assigned to work on a project its either same or different project. A **collaborator** has the permission to:

* Push to (write), pull from (read), and fork (copy) the repository
* Apply labels and milestones
* Open, close, re-open, and assign issues
* Edit and delete comments on commits, pull requests, and issues
* Merge and close pull requests
* Send pull requests from forks of the repository
* Create and edit Wikis
* Create and edit Releases
* Remove themselves as collaborators on the repository (Github Help, 2015)

Under **collaborators** is a feature called **team** wherein Github are included and group by to work on a same project. The administrator of the team can assigned team members what privileged he/she can use it’s either pull only, push and pull, pull and administrative. The administrator of the project can limit the privilege and see the information of the members without their permission.

3 Github Help, (2015). Retrieved from https://help.github.com/articles/permission-levels-for-a-user-account-repository/

## Recognize best document types for version control (code, CVS/TSV, small binaries)

**CVS (Concurrent Version System)**, uses a [client–server](https://en.wikipedia.org/wiki/Client%E2%80%93server) architecture: a server stores the current version(s) of a [project](https://en.wikipedia.org/wiki/Project) and its history, and clients connect to the server in order to "check out" a complete copy of the project, work on this copy and then later "check in" their changes (Wikipedia, 2015). CVS can be use locally to your computer and it’s free to use by everyone such as Git, Mercurial and SVN.

While Code applications for version control are the IDE applications such as **Eclipse** and **NetBeans** where **Git plugins** are needed to download and install. Doing this there’s no need to use Git to add, commit, push, pull and etc. to remotely access your Github repository. With the help of IDE applications project files can import from Github to Eclipse or NetBeans. You just need to:

* connect it by creating an account to Github
* cloning the repository of your project from Github to Eclipse
* import project files from the repository
* commit all the changes from the imported project files to remotely control, change, monitor, and store your project online using Github.

By using IDE applications coders or programmers can use Github easily and more convenient. All the things that are in Git will be available and much better with the help of code applications.

4 Wikipedia (2015). Retrieved from <https://en.wikipedia.org/wiki/Concurrent_Versions_System>.

1. Wikipedia (2015). Retrieved from https://en.wikipedia.org/wiki/Git\_(software) [↑](#footnote-ref-1)
2. Wikipedia (2015). Retrieved from https://en.wikipedia.org/wiki/GitHub. [↑](#footnote-ref-2)