CSC 648/848 Software Engineering – Spring 2023
Milestone 0

02-03-23

Anthony John Souza, Class CTO, SFSU
***ajsouza@sfsu.edu* and discord: @ajsouza25**

Dragutin Petkovic, Class CEO, SFSU
Petkovic@sfsu.edu

1.Introduction

The high-level purpose of this milestone is for student teams to choose, install and prepare IT infrastructure for the development of team final project in CSC 648-848. Specifically, in M0 student teams must achieve the following objectives working together:

- Configuring GitHub to enable team SW development
- Choosing a SW stack and delivery platform and service provider from given list in this document
- Getting your choices for SW stack and delivery platform approved by Class CTO Anthony via e-mail
- Installing all necessary tools and SW for a chosen stack and delivery platform
- Creating a joint WWW page with info about team members using above infrastructure (this page will be graded but will also serve as ABOUT page for your team application)

The deployment server and *all* team applications for final team project must be running on a remote host. *Localhost hosting will not be accepted.*

In this class we will have **class CTO** who will deal with high-level technical issues (Anthony Souza) and **class CEO** who deals with schedules etc. (Prof. Petkovic).

The work on M0 shall commence only after the teams have been formed and basic accounts set up – you will get e-mail on this from Class CTO Anthony

Note that your team will be choosing most of the IT infrastructure for developing the team project from the list provided in Milestone 0 – then the team choices will have to be

approved by class CTO Anthony. After choosing the infrastructure and after it is approved it has to be installed and all team members have to follow M0 and learn how to use the chosen tools. M0 and class IT involves infrastructure designed to give teams more freedom but more responsibility in maintaining your infrastructure. Hence, there may be problems. Please contact class CTO Anthony ajsouza@sfsu.edu or class TA if you encounter any problems that you cannot solve yourself or with your team.

Your responsibility:
Since CSC 648-848 is a capstone senior course, students are expected to learn necessary tools themselves and in working with their team members. Use of on-line resources and asking around (team members, other classmates) is encouraged. Class instructors will only help in case there are problems, and only at high level. Also, specific coding details will be the duty of students to figure out with the help of on-line resources and asking around. Class instructors will also post some relevant resources (use of git/hub in team setting, several code templates for backbone infrastructure of the team project etc.).

The IT environment and setup we envision for the class is close to what you will be required to know to be successful SW engineer, so M0 is a great experience to improve your marketable skills. While some of the work is to be done *individually*, you must work together with your team to complete the task. Note that your learning will consist of: a) individual usage of tools; and b) most important: how to use the tools in team setting (e.g. code merge etc.).

Your choices of IT stack and hosting platform will be from a limited list: this is how it is in most companies and it also simplifies our supervision and support to you.

M0 carries 10 grade points. Information about grading can be found at the end of the document. Again, each team member must complete his/her M0 tasks but to accomplish this each team member must also work with the rest of the team. **This means that you can help each other, ask for help, work in pairs and are in fact encouraged to do this to not only help with M0 but also build the teamwork.**

Summary of M0 tasks (the flowing sections describe the details)

Your first task is to read through the entire document. It contains very important and valuable info and requirements for M0. The tasks for M0 follow and must be done in the order listed below:

1. Set up GitHub for team SW development
2. Appoint team GitHub master

3. Select Server/Platform Provider, Software stack for Deploying Your Team Web Application and ask class CTO for approval

4. Get Server/Platform and Software Stack Approved by class CTO

5. Install and configure approved Software Stack (to be done after the above approval by class CTO).

6. Correctly Populate Credentials Folder in Git Repository

7. Create a Team Website and ABOUT page

8. Submit M0 for grading

Again, M0 is a great vehicle to help you learn a very important part of SW engineering work which is setup, use and maintenance of professional development and deployment infrastructure.

Note that we use *group* or *team* interchangeably.

Please read this document carefully and follow all the required steps. Follow the process for all submission and communication (e.g. approvals) exactly as described in this document, Mo submission is NOT done via Canvas assignment submission functions.

You must deliver M0 on schedule. Any delays must be approved by class CTO Anthony (Class CEO, Dr. Petkovic CC'd) and asked for via e-mail at latest 24 H before the M0 deadline.

2. Setting up GitHub for team SW development

2.1 GitHub Team (Private) Repo

The purpose of this part of the exercise is for your team to set up the team private GitHub repository that is going to be used for storing your team's project (SW, documentation, formal milestone deliverables etc.) and be accessible only to the team members and instructors. Only one member needs to set up the private repository.

(Check also GitHub usage slides posted on class Canvas, created by class CTO Anthony)

As a first step, all team members need to have their own GitHub account. This is mandatory, NO EXCEPTIONS - you need to have your own GitHub these days anyway.

2.2 Creating GitHub Team private repo.

1. Select one team member to create the private repo. Selected team member then uses the following link to create the private repo:
   Project Repository Creation Link
2. A chosen team member (team leads must ensure this happens) will then ADD ALL MEMBERS of the team to the private repo. Each invited team member has to accept the invite. **(For each non-invited team member, 1 point will be deducted from M0 Grade for the whole team to promote teamwork.)**
   a. **Simply inviting the team member is not enough**. They need to accept the invite. Non-confirmed invites will get the same penalty as no invite for those who do not follow up. **Team leads have to make sure team members respond to this**
3. All team members are required to fill out the table in the readme.md file in the root of the repository with the following information:
   a. Student name
   b. Student email (please use SFSU e-mail)
   c. GitHub username

Team members are strongly **encouraged to** practice creation of branches and code merge, which turned out to be occurring problems with previous teams. Please consult on-line resources for **GitHub** best practices on this topic and the slides we posted on class Canvas (created by Anthony).

The team repository created with the above link will be used to store your team's source code and milestone documents. This repo is already private and configured for you to begin work right away. After the class you can copy this to make your own portfolio very useful in job search

***BOTH* instructors are already attached to the repo so you do not need to invite them. Only your team members need to be added.**

There is *NO* need for a paid subscription for the private repo. This has been taken care of for you.

2.3 Appoint team GitHub master

Teams are also requested to appoint a GitHub master who will help organize and maintain GitHub according to best practices – M0 is a good time to do this.

3. M0 tasks

This section describes how to develop actual M0 tasks, after you have completed all the previous steps.

Task 1: Selecting Server/Platform Provider, Software stack for Deploying Your Team Web Application and ask class CTO for approval

The purpose of this task is for you as a team to do the following:

- Select a *Server/Platform provider*
  Some possible and strongly recommended choices are below:
  - Amazon AWS
  - Google Compute Engine
  - (Please refrain from using ***Heroku***. While simple, it has caused issues for many teams in the past that have delayed application development that has affected teams' final grade.)
- Select *a Software Stack*. An example would be a LAMP Stack. This includes a) the OS choice; b) web server choice; c) database choice; d) and server-side language choice, and e) any other important technologies needed your Software Stack.
- The server-side language for your web application must be one of the following:
  a. Python
  b. JavaScript
  c. PHP

  You may ***NOT*** use another server-side language that is not listed above. If you strongly disagree you may contact Class CEO and CTO with your suggestion.

IMPORTANT: When using AWS or Google Compute Engine please refrain from using their app features. These auto-deployment tools cause some issues for students and in some cases caused students to be charged (Most of it not all charges were refunded). If you do use them, use at your own risk we are not responsible for accrued costs for running the server.

Note: you will first select your choices then you will need them approved by class CTO before proceeding, see below.

For the DB we only allow MySQL for several reasons (well supported, standard SQL, available setup and management tools including Workbench, and has facilities we will use for search like %Like etc.). Some helpful hints re: connecting to MySQL are in Appendinx III

When deciding on the technology to be used in your Software Stack, besides functionality (a common factor to start from), you should keep a few additional factors in mind when comparing technologies side by side. Some key factors may be:

- How often is the technology updated?
- How well is the technology supported?
- How mature is the technology?
- How well is the documentation?
- How good is the community using this tool?
- What features do is offer compare to other technologies.
- And very important: How easy is it to use/learn for all your team members given specific class schedules?
- Please avoid choosing the technologies where there is no expertise in the team. Remember: your focus is to deliver the application and NOT necessarily learn/use newest technologies

Note: your choices must be made such as not to incur any costs, which is possible today given many free offerings on the market. Appoint somebody to regularly (twice a week at least) checks cloud server billing to be reasonable and if there are issues act on it and/or talk to class CTO) (often additional billing is caused by automatic rstart of apps so please disable it)

Task 2: Getting Server/Platform and Software Stack from Task 1 Approved:

When your team has decided on a software stack, **team lead** needs to email the class CTO Anthony (cc'ing the class CEO Prof. Petkovic as well) detailing your software stack. The email **must have the format as in Appendix I:**

- Receivers: Class CTO and CEO
- E-mail to be sent only by team lead
- E-mail Subject line: "CSC 648 848 Spring 2023 Approval of IT choices Team M"
- List the Technologies used in your software stack.
- This includes the following:
    - Server Host, Instance size (CPU and RAM)
    - Operating System and Version Number
    - Database and Version Number
        - ***Only MySQL Database is allowed***
    - Webserver and Version Number
    - Server-Side language and Version Number

    - Also list any technologies or packages you will need that you think is important. (you can exclude things like git and SSH). There is no wrong answer here, just list what you can. The more the better, it'll help me determine how sound your software stack is.
- Then list each member's familiarity with the backend language. A sample is given in the Appendix of this document.

A sample email template for this task has been given in the Appendix I of this document. Please follow this format. This will ensure a speedy response. If the class CTO has any questions about your software stack, make sure that your team replies promptly. Delayed responses can delay Software Stack approval.

MAKE SURE TO FOLLOW EMAIL FORMAT. MOST IMPORTANTLY USE THE CORRECT SUBJECT HEADING. ANY EMAILS SENT WITH WRONG SUBJECT HEADING WILL BE EITHER RETURNED OR MISSED (IGNORED).

Once your software stack has been approved via e-mail from Anthony you must begin installing and configuring your server and completion of M0 immediately.

Task 3: Installing and Configuring Approved Software Stack (to be done after the above approval by class CTO).

After your server/platform provider and software stack have been approved by e-mail from Anthony it is now time to begin installing everything.

There will be no detailed instructions given for this as there are too many possible configurations. But you may follow these simple steps:

1. Start server Instance with your server host
2. SSH/log into your server
3. Update your server
4. Install Webserver
5. Install DB
6. Install Server-Side language
7. Install remaining needed packages
8. Make any needed configurations.

Some notable Stack installation instructions:

- AWS w/ Node
- LAMP Stack AWS EC2
- LAMP Google Cloud Platform

This may not be the most detailed set of instructions be it gives you an order of items to work through. **Please use on-line resources and follow the documentation!** If you run into any issues, there is help available. You may ask anyone in your team, your class, and class CTO Anthony.

THE CLASS CTO ANTHONY MUST BE GIVEN ROOT ACCESS TO THE TEAM SERVER (SSH ACCOUNT) AND DATABASE (DB USER WITH FULL PRIVELEGES). THERE IS NO EXCEPTION TO THIS.

Task 4: Correctly Populate Credentials Folder in Git Repository PLEASE DO NOT DO THIS TASK UNTIL TASK 3 IS COMPLETED AND SW TESTED.

The purpose of this task if for your team convey all the needed information for logging into your team's server and database. Failure to correctly give the required information will cause points to be deducted from your team's M0 grade as well as other milestones as well.

In your team's repository there is a folder named credentials. The folder contains a README.md file with a set of instructions and required information. Please read this carefully and follow ALL instructions give ALL required information.

You may store all the required information in the README.md itself EXCEPT for SSH keys. These should be stored as files inside the credentials folder in your team's repository.

Task 5: Create a Team Website and ABOUT page

Now that the setup and titillations are done, we can move on the M0 task. The purpose of this part of the exercise is to get you to work individually to create your own webpage within your team's framework context, and then to work with your team to join these pages together using your teams GitHub account and chosen framework into a single site. We recommend that you in fact create ABOUT WWW page which introduces the team members. This then is to be part of your final team application and is great for your portfolio.

Everyone in your team should clone the team's private GitHub repository into his/her individual shell account or onto your local computer, and using only the chosen team framework create their own WWW page that at a minimum displays their name and their or some other image (if you are comfortable it is good idea to use your own photo which is useful for your ABOUT page and portfolio. **Photo is optional**). This work must be

completed by individual student and then pushed to the team's private GitHub repository. The file(s) created/added should then be merged into the team's private GitHub repository.

The website must be deployed onto the team's remote server setup as in Task 3. ABOUT page and all student individual pages created for M0 task must also be deployed on the server.

Make sure your individual test environment is identical to the one on the server, see end of this section.

One of your teammates will need to modify your framework's home page to point to all the different team member pages (all residing at the server). The finished site must look something reasonably close to the example in Figure 1. The user 1...user 6 buttons should be replaced with buttons pointing to the individual pages labeled with the username of the person who created that page. **The website (home and individual pages) shall be deployed onto the team's group shell account**.

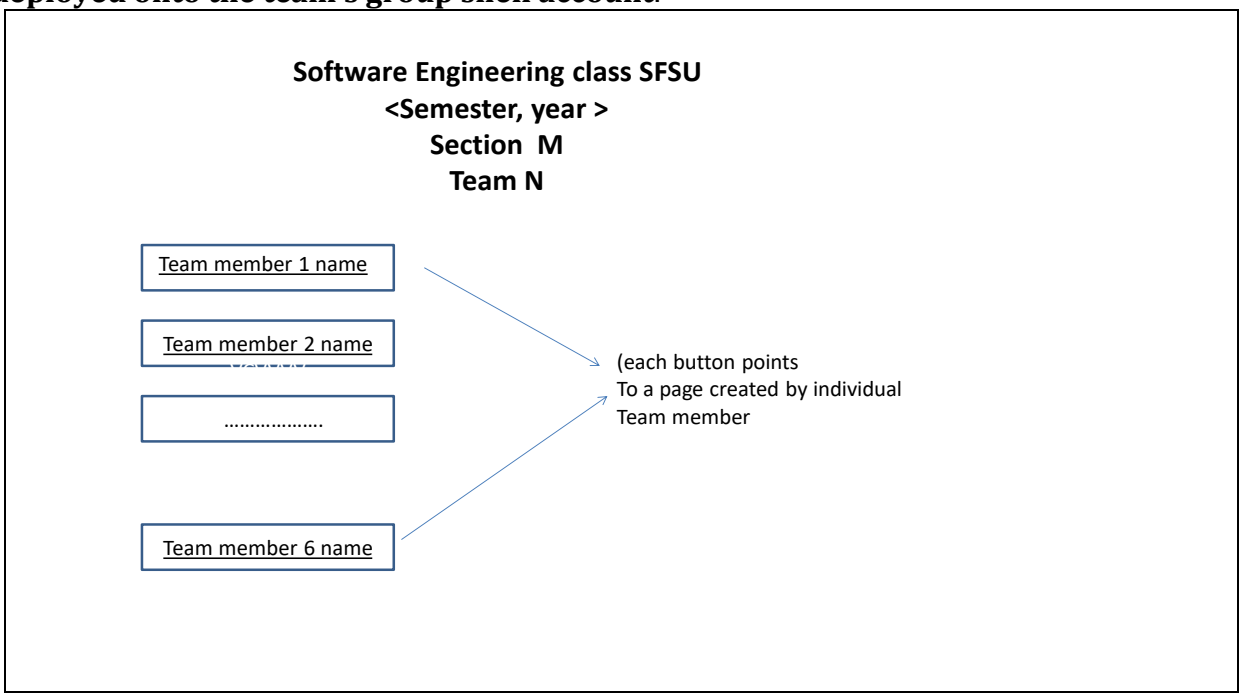


Figure 1. Example mockup of Milestone 0 team home page (**replace semester and team number with yours**).

This single ABOUT page and individual team member pages) site must **be served from the remote server you setup in Task 3.** This is the same way you will deploy your final app, hence this is useful to make sure you learn and test it, especially how to deal with branching and merging code. Every team must understand how your team's application is deployed and managed.

NOTE on student individual test and dev environment:

Every student must have a correct test and dev environment on their working PC e.g. make SURE the same steps you do to configure your deployment sever you also do for your test environment. This has caused a lot of issues in the past. For example, students would setup test environments on their personal laptops, but the environment would not match what the deployment server had. If your deployment and test environment differ too much, then deployment becomes a hassle especially running the app on the deployment server.

For example, in of the recent semesters the deployment server was running Node Js with NGINX webserver in front of the Node apps. Many students would have test environments on their PCs mimicking their node app on their PC but forgot to add NGINX as well. Forgetting NGINX caused the following problems:

- Apps would not even start after being pulled from GitHub
- Apps paths did not work
- Apps routes did not work.
- Apps were missing dependencies obtained from test environment.
- Apps would perform differently compared to test environment.
- And the list goes on.

Therefore we strongly recommend that most experienced team members first create AND test and comment the SW stack and deployment and test environment on the server, then others should copy/clone it on their laptops

NOTE about using deployment server: most choices will be on some form of a cloud service. To use cloud service properly one must understand how they charge. For example, it may happen that unintentionally students create many instances of SW or applications and have them running silently (unless they are stopped), thus incurring extra costs. It is imperative to avoid this and we strongly suggest that somebody in the team be in charge of monitoring cloud usage. Also, cloud storage is very expensive so make sure you use it minimally and understand how it is charged (class project requires only minimal storage). Keep in mind that if your team uses app engine for google or the AWS equivalent stopping an instance is NOT enough. Killing an instance will only cause it to respawn since the app is active. To stop this the APP itself needs to be stopped or deleted which then will terminate all instances associated with that APP. Failure to do this may incur large costs for the service

We strongly recommend that you have one team member check cloud billing regularly.

NOTE on individual members additional tasks: once the team has chosen given infrastructure, it is the duty of all team members to learn and brush up their knowledge on

all necessary technologies on their own. This will make team development much more effective.

4. Submission and Grading of M0

4.1 Submission of M0 for grading

All projects will be inspected and graded on the due date TBD. The key for grading will be your team's correct installation and configuration of software stack, correct setup and usage of GitHub, and the deployment of your About page on ***YOUR CHOICE OF REMOTE SERVER (*** **section 2).** Emphasis for M0 grading is on correctness (not so much on UI design of the final team page) and proper usage of development tools and deployment method.

Once you are done and ready for grading team lead must send e-mail to Dr. Petkovic AND A. Souza with the following subject line and email body (**you MUST follow this protocol**)

E-mail subject line:
"CSC 648-848 Spring 2023 Team M"

E-mail body (see Appendix II):
In the e-mail say that M0 is ready for grading. Provide a link to your application and GitHub repository. Please do not send emails with no text and only a link or with empty body messages.

You must deliver M0 on schedule. Any delays must be approved by class CTO Anthony and asked for via e-mail at latest 24 H before the M0 deadline.

4.2 Grading of M0

This assignment is worth 10% of your class grade, as indicated in the syllabus. The grade for this assignment will be determined according to the following criteria:

Category	Points
Correctly Installed and Configured Software Stack	2
Correctly Populated Credentials Folder in GitHub	3
Correct use of Git and GitHub	2
Correct team WWW page functionality, deployment and proper usage of team's Software Stack for creating web page	3
Total:	10

Team or individuals who do not get full grade will be given feedback and will be asked to correct asap to get full points, because we want all team members to do this step correctly and learn from it since it emulates steps to be done in full app deployment.

Appendix I - Sample Email for Software Stack Approval (Task 2 Section 3)

e-mail subject line:
"CSC 648 848 Spring 2023 Approval of IT choices Team M"

e-mail body:

```
Hello Class CEO and CTO

     Below is a list of the technologies used in TeamNN's
software stack:
     Sever Host: Google Compute Engine 1vCPU 2 GB RAM
     Operating System: Ubuntu 16.04 Server
     Database: MySQL v……
     Web Server: NGINX 1.12.2
     Server-Side Language: Python
     Additional Technologies: Web Framework: Flask
                              IDE: PyCharm InteliJ,
                              Web Analytics: Google Analytics
                              SSL Cert: Lets Encrypt (Cert Bot)
                              SASS: 3.5.5



          Member's Familiarity with Server-Side Language on a
scale of 1 to 5,
          with 5 being very familiar and 1 being never used it.

          Jane: 5
          Jack: 2
          John: 3
          Joe: 1
          Jill: 4
          Jake: 5


          Then any extra information you think is necessary.
     Best,
          Team M <team lead name>
```

Appendix II - Email Submission for Milestone 0 (Task 4.1 section 4)

E-mail subject line:
"CSC 648-848 Spring 2023 Team M"

e-mail body

```
Hello Class CEO and CTO

     Our team has completed Millstone 0 and is ready for
grading.


     Link to application

     Link to GitHub Repo

     Please state (say YES) that your team has correctly
     populated the credentials folder.

     Then any extra information you think is necessary.


     Best,

          Team M <team lead name>
```

Appending III - Connecting to MySQL Database via cmd line or Workbench

Connecting to MySQL Server via MySQL Workbench

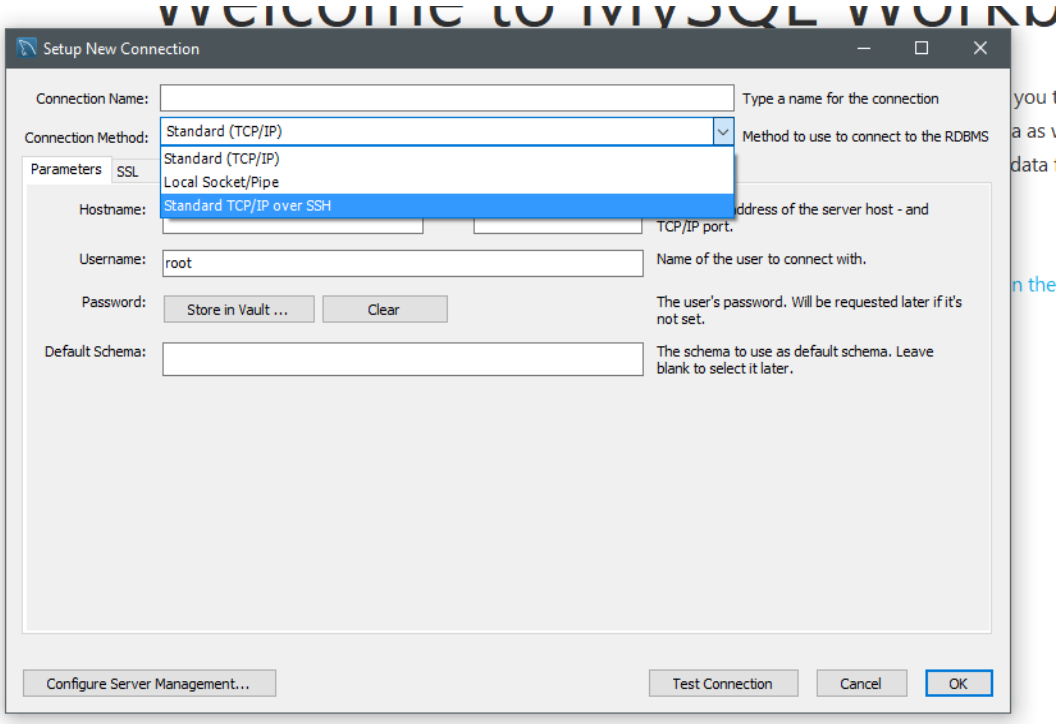
There are two ways you may access your database. One way is logging into the remote Google Compute Engine server and accessing your database via command line with the following commands:

mysql -u username -p
(where username is your database username and password is your db password)
connect db_name
(where db name is the database name you wish to connect to)

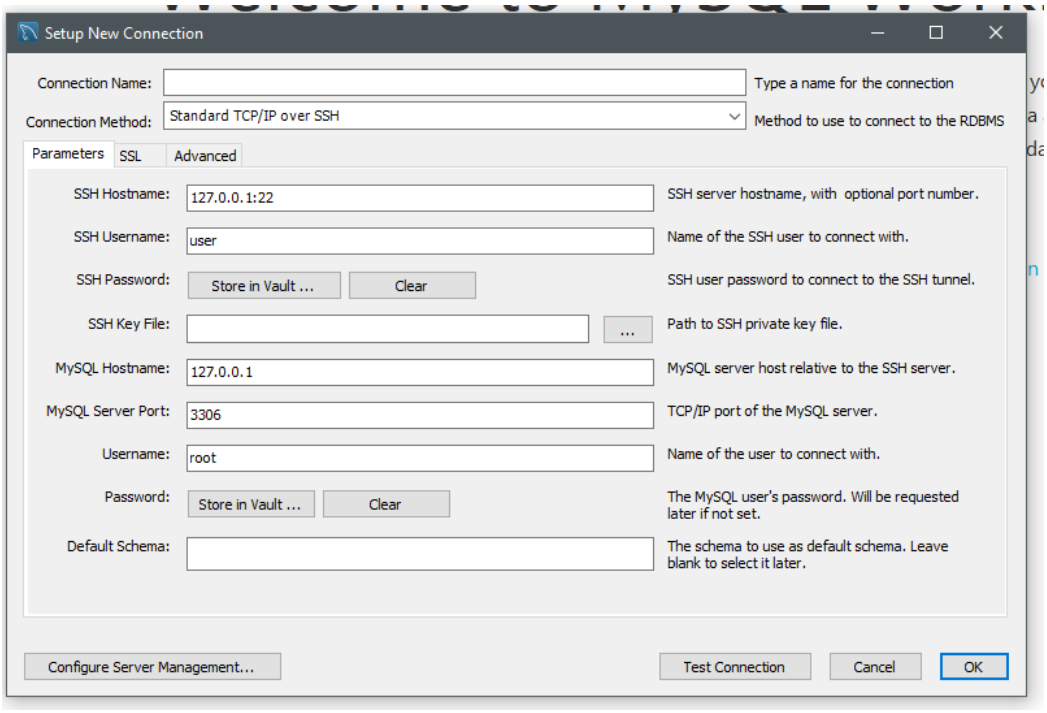
At this point you may start executing SQL commands against your connected database.

The other way is via MySQL Workbench. This is GUI application used to manage MySQL databases.

To connect to your database, you first need to create a new connection. This can be done by clicking the plus sign (+) near the text MySQL Connections. This will open a new tab. You will then select a connection type. The connection to select is "Standard TCP/IP over SSH". Figure shown below.



Next, you will need to fill in the required information for creating a new connection over SSH.



Using the above figure as an example, fill in the following information:

- Connection Name: teams_db
- SSH Hostname: your host name (could be an IP)
- SSH username: your_ssh_username
- SSH Key File: path you your key file
- MySQL Hostname: 127.0.0.1
- MySQL Server Port: 3306
- Username: database username, same as shell account name
- Password: database password, should be student id, unless changed.

Default Schema: student_username, this your database name, where username is your database account name.

Other Option is to Connect with username and passed.

This is a simple process and is the default way to connect. Therefore, the steps will not be shown. But you should note the following:

- To connect to database remotely not via SSH a port needs to be opened on the server pointed to the database. Note that this is against good security practices.

- A user account in your database needs to be made for the connection. DO NOT use root to do this, while easy and simply its lazy and goes against good security practices.