Discover Workshop: Modern Cloud Apps with Micro Service architecture on Azure with Immeo: Lab setup guide

# Create repository on GitHub

1. Go to <https://github.com/> and sign-in with your existing account or create a new account
2. Create repository with name "Fabmedical". For detailed instructions on how to create new repository see Task 7, step 1-5 on <https://github.com/microsoft/MCW-Cloud-native-applications/blob/master/Hands-on%20lab/Before%20the%20HOL%20-%20Cloud-native%20applications.md#task-7-create-a-github-repository>
3. Create personal GitHub access token with "repo" access. - <https://docs.github.com/en/github/authenticating-to-github/creating-a-personal-access-token>

# Commit lab code to your own repository

Install git client if you do not have one, <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

1. Clone source repository

You can choose either to clone the MS source repository or the Immeo prepared repository.

The difference is that the Immeo repository includes many of the files to be created during the lab with .bak extension.

**If using the Immeo repository beware** that the top folder structure will be slightly different. The content-api, content-web folders are in ~/Fabmedical instead of ~/MCW-Cloud-native-applications/Hands-on\ lab/lab-files/developer/

MS source files: git clone https://github.com/microsoft/MCW-Cloud-native-applications.git

Immeo source files: git clone <https://github.com/immeorfj/Fabmedical.git>

1. Commit the code to own GitHub repository

*Run the following commands where the <> parts are replaced with your data:*

git remote add origin <your GitHub URL>

git push -u origin master

git config user.email "<your email>"

git config user.name "<your name>"

git config --unset credential.helper

git config credential.helper store

# Install VS Code and Remote - SSH extension (optional)

This step is optional but if you are familiar with VS Code or want a richer editor experience than a bash editor (vim) can offer the combination of VS Code with the remote ssh extension can offer this.

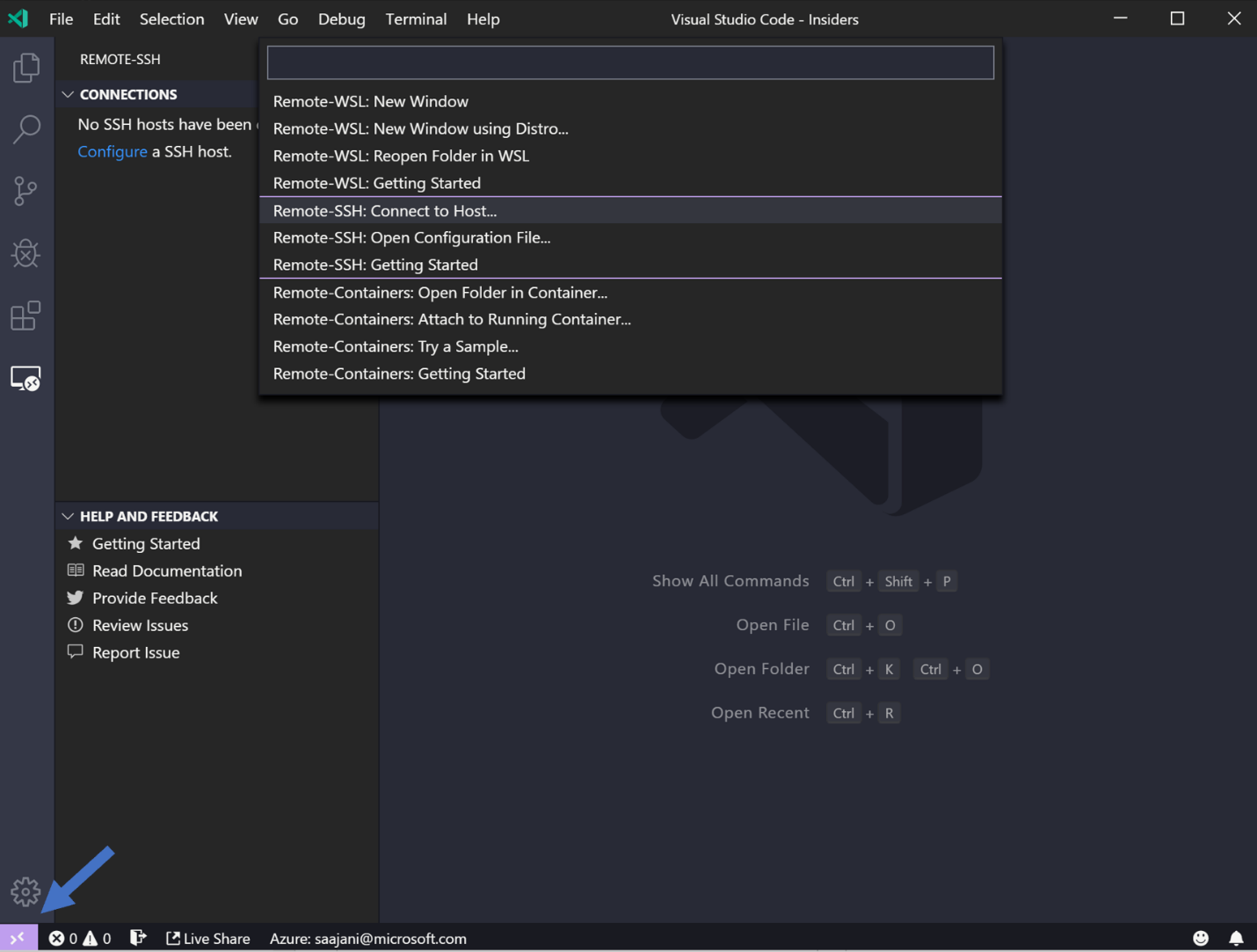
1. Install VS Code if you do not already have it installed.

<https://code.visualstudio.com/>

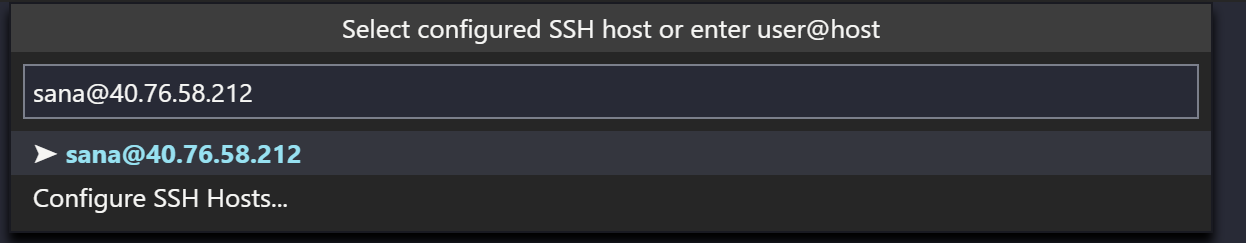
1. Install an OpenSSH compatible SSH client (PuTTY is not supported).

<https://code.visualstudio.com/docs/remote/troubleshooting#_installing-a-supported-ssh-client>

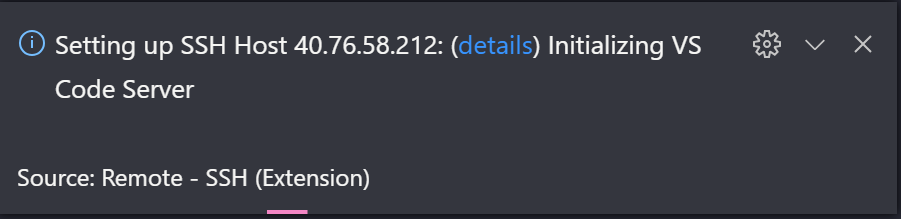
1. Open VS Code
2. If you do not already have the Remote - SSH extension installed, you can search for "remote ssh" in the Extensions view (Ctrl + Shift + X)
3. When asked during the lab to ssh to the build agent choose the Remote-SSH: Connect to Host command and connect to the host by entering connection information for your VM in the following format: user@hostname.



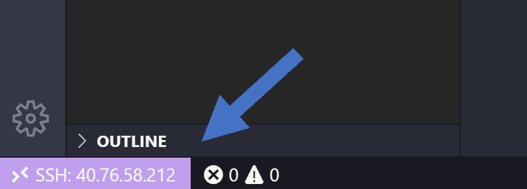
1. Set the user and hostname in the connection information text box.



1. VS Code will now open a new window (instance). You'll then see a notification that the "VS Code Server" is initializing on the SSH Host. Once the VS Code Server is installed on the remote host, it can run extensions and talk to your local instance of VS Code.



1. You will know you're connected to your VM by looking at the indicator in the Status bar. Now it shows the hostname of our VM!



1. Once you are connected to the build agent, you can interact with files and open folders on the remote machine. If you open the integrated terminal, you will see you are working inside a bash shell.