**Bachelor of Computer and Information Sciences**

**Contemporary Issues in Software Engineering**

**Semester 2, 2024**

**ASSIGNMENT 1A: Set up Development Environment**

**Worksheet 1 (20% of Ass1A)**

*VS Code, Git, GitHub, Nest*

# Deliverables and Due dates:

You are required to complete the Worksheet and keep evidence as you do it by selectively taking screenshots of your work, as well as explanations.

**Each worksheet should ideally be checked off by the TA by the end of the week’s tutorial**

**This worksheet should be Checked off and Uploaded on CANVAS ideally by end of Tutorial Week 1 – all four worksheets for Assignment 1a are due by week 6, and the knowledge develops cumulatively so don’t leave it to the end – that will also make it hard for the TA’s to mark and give you feedback.**

*EXPERIMENT – BE CURIOUS – TEACH OTHERS – TAKE SELECTED SCREENSHOTS FOR KEY ASPECTS*

The worksheets will have some theory, a practical exercise, and a worksheet for answers to questions and at least three selectively captured screenshots as evidence. The aim is to be able to learn from the exercise, and evidence that. For each of your three selected screenshots (or sequence of shots) in a brief paragraph or two reflect on why you have selected it. What have you learnt in this part of the worksheet? What was new or surprising? What useful external resource(s) did you consult and why? Provide a link(s) to the resource.

**By the end of this Worksheet you should be able to….**

1. Create branches and work with them appropriately in GitHub and Git
2. Keep a local repo synchronised with a GitHub repo using pull and push commands
3. Write useful commit messages
4. Use the pull request feature before merging code to Master branch in GitHub
5. Merge pull requests
6. Work with Git and GitHub from VS Code locally (make sure you know what is happening in Git!)

(Use the “GitHub Pull Requests and Issues extension)

<https://code.visualstudio.com/docs/editor/github>

1. Understand what NEST is and how to use the Nest CLI to create a project.

**Introduction**

This worksheet introduces you to:

The toolset you will need for the team project to enable you to: 1) develop and share code within the team, 2) apply practices to manage local versions and migrate changes to the team repository, 3) ensure the quality of the code developed and released through the workflow to be established, 4) become exposed to the tech stack for the project.

**Git**, a version control system (VCS) to manage code changes locally for each developer,

**GitHub**, a cloud-based version control system (VCS) to manage Git code repositories and share code

**Visual Studio Code** (VS Code), an IDE and text editor with some intelligent features and plug-ins for coding and integrating with other tools.

These tools are used to write, test, integrate and share code so developers can collaborate.

**Exercise using VS Code, Git, GitHub & Nest**

**Do the following exercise:**

You will need to create folders to store the repository content for each week’s worksheet. Previous examples of the typical folder structure that you will have for the worksheets is given below, with the top-level folder being called CISE\_Repos:

A screenshot of a computer

Description automatically generated

1. Create a Github account if you have not already (note your token somewhere safe)
2. Create a new GitHub repository called CISE\_Repos to hold the worksheets, [and note that a later team repository may be named as a [Main Repository]--- CISE + Team Number + ProjectName]
3. Install Git on your local machine (<https://github.com/git-guides/install-git> )
4. Create a folder for your projects called “CISE\_Repos” on your local machine
5. Install VS Code on your local machine <https://code.visualstudio.com/>
6. Install the “GitHub pull requests and Issues” extension for VS Code and authorise if necessary
7. Clone a copy of this empty repo to your local machine in the “CISE\_Repos” folder using the following:

From a terminal window in VS Code, at the CLI type:

>git clone https://github.com/**<Your *GitHub username*>**/CISE\_Repos

>cd CISE\_Repos to change the working folder

>git status to check git is working.

1. Create a README.md file and push it to the main branch in GitHub

>echo "# CISE\_Repos" >> README.md creates a README.md file with “CISE\_Repos” as the title.

>git add README.md stages the README.md

>git commit -m "first commit" saves a snapshot of the changes to README.md with commit message

>git push -u origin main pushes the latest changes to GitHub remote main branch from the local main branch (You should see your README.md in GitHub now).

Your folder structure should now look like this:

A screenshot of a computer

Description automatically generated

1. Create locally a “Development” branch off the Master/Main branch and change to working in that branch

>git checkout -b Development This creates a new branch and moves you to the branch

>git branch -a to check what branches you have in your local repo type – green is the current one

1. Use VS Code to create a new folder called “worksheet1” (or any name you like) in the CISE\_Repos folder for worksheet1’s content. Then create a folder “CISE\_React” in folder worksheet1.

A screenshot of a computer

Description automatically generated

Create a new file called Delete\_me.html in the CISE\_Repos/worksheet1/CISE\_React folder and add the following code to the file Delete\_me.html:

<!DOCTYPE html>

<html>

<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>

</html>

A screenshot of a computer

Description automatically generated

Stage and commit this file, with a useful commit message.

If you run “git status” command, you will see that the worksheet1 folder (and everything inside) is untracked at the moment. You can use “git add .” (there is a “.”) to add all files, or “git add worksheet1/” to add everything under worksheet1, or “git add worksheet1/CISE\_React/Delete\_me.html” to add this file. Remember to commit it after adding it.

A screenshot of a computer

Description automatically generated

Note: Make sure to write useful commit messages (THIS IS VERY IMPORTANT – it documents the reasons for changing and what was changed). Read the following articles to understand:

<https://betterprogramming.pub/why-every-git-commit-message-must-include-its-commit-context-1171c0b2f710>

<https://dev.to/helderburato/patterns-for-writing-better-git-commit-messages-4ba0>

1. >git push -u origin Development to push your changes to GitHub
2. Check Delete\_me.html has been sent to GitHub in the Development branch in GitHub. You should see a message in GitHub like the following:



1. Press the green button and continue accepting until you have merged this change with the main branch on GitHub. Usually the main branch is the production branch to be deployed to users.

It is common to create a branch structure with feature branches coming off the Development branch like this (Fig 1)

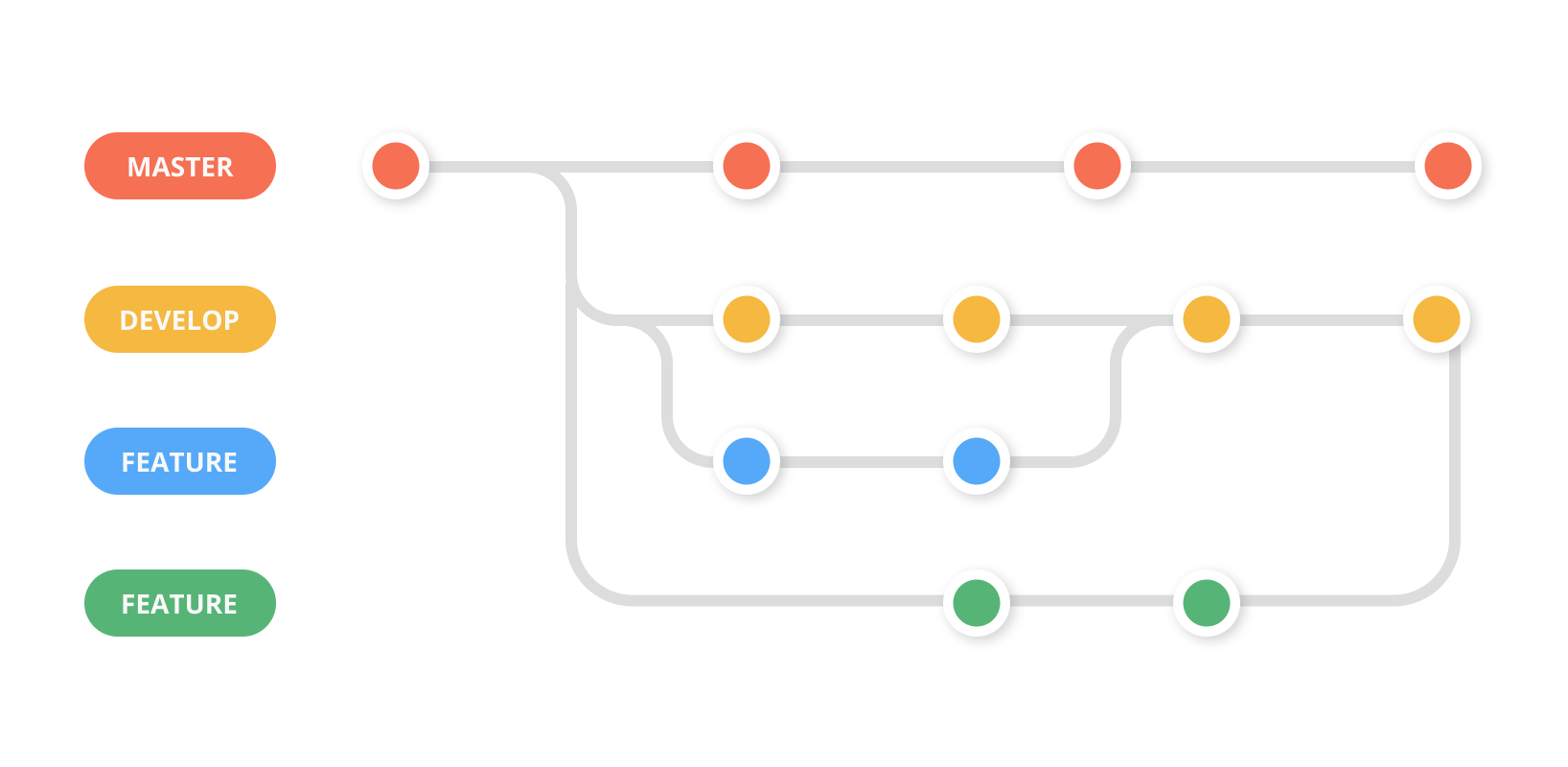


Fig 1

1. We are now going to create a simple React app using a script called “create-react-app”. In order to use the script and some other features we need to install node.js, which is a javascript run-time that allows us to run javascript outside the browser (thank you Google!)
2. Follow the instructions to install node.js on your Operating System

<https://nodejs.org/en/download/>

1. To check it has installed and the version you have, at the command prompt you can type

>node -v

1. To install all the dependencies for a React application we can install the “create-react-app”. This will create a new folder called cise-react-learn for this app. First navigate to worksheet1 folder in your terminal, then use the commands:

>npx create-react-app cise-react-learn

>cd cise-react-learn to change to the new project directory

>npm start will start the development web server and open a browser to show you the simple app it created with a spinning logo.

**README.md** is a markdown file that includes a lot of helpful tips and links that can help you while learning to use Create React App.

**node\_modules** is a folder that includes all of the dependency-related code that Create React App has installed. You will never need to go into this folder.

**package.json** that manages our app dependencies and what is included in our node\_modules folder for our project, plus the scripts we need to run our app.

**.gitignore** is a file that is used to exclude files and folders from being tracked by Git. We don't want to include large folders such as the node\_modules folder

**public** is a folder that we can use to store our static assets, such as images, svgs, and fonts for our React app.

**src** is a folder that contains our source code. It is where all of our React-related code will live and is what we will primarily work in to build our app.

A screenshot of a computer

Description automatically generated

>git add . (note the dot) to get Git to stage all files in the cise-react-learn folder.

>git commit -m “first commit” to commit all the files created

>git push origin Development to push all the files created to the Development branch of GitHub

1. Make a change to App.js file in the src folder – change “Learn React” to “Agile is about Values and Principles” and save the file. Notice the filename is yellow in VS Code – indicating it is not committed.

Note that your app should have updated in the web browser automatically.

1. Commit this change and push it to GitHub. (Remember the good quality commit message discussed in step 10!)
2. (Check that the files were updated in GitHub)

(Ctrl-c in terminal windows will stop the program running)

**Collaborate on a Repository with a team**

1. In Git create another branch “LogoLink” from the Development branch and move to it
2. Change the App.js file in VSCode so clicking the link takes you to aut.ac.nz instead of reactjs.org
3. Commit and push to GitHub creating the new “LogoLink” branch
4. Invite a classmate or friend who has a GitHub account (or will make one) to be a collaborator with your repository. (or make another GitHub account using a different machine).

“Settings>Manage Access” in GitHub. you will need to know your collaborator’s GitHub account name

1. **Create a pull request** for this merge of the “LogoLink” branch with the Development branch, so the changes are in this branch. Other developers working on the same repository would clone this branch to get the latest code to work on several times per day. You can create the pull request with Git command or use the extension you installed into VS Code (see Fig 3).
2. Go to GitHub on your browser and you should see one open pull request. The Reviewer assigned to the pull request would normally review this code to be merged to check for errors etc and there may be a discussion with the original author of the code to fix something or get clarification. **Get your reviewer to open the pull request and merge this change with the Develop branch**
3. Install the Prettier plug-in to VS Code and see how it works. Read about the purpose of a Linter – we will use ESLint. Try installing the ESlint plug-in for VS Code.

**Work with NEST**

Step 28: Install NEST CLI on your machine. Open a terminal and use the following command to install the CLI globally: <https://docs.nestjs.com/cli/overview>

>npm i -g @nestjs/cli

*Joshua gave a heads-up on an issue he picked up in Worksheet 1 for Mac users:*

*Hi tony, just wanted to quickly flag that ive had a couple mac mac users now that have had trouble installing nest js as per step 28 as they were getting a permission error despite being the only user on the device.*

*The fix for this is using "sudo npm i -g @nestjs/cli" rather than just using the step as it is written. it might be worth adding a note in future versions of worksheet 1*

Step 29: Verify the installation of NEST CLI by checking its version. Type the following command:

>nest --version

(Note you may not have permissions on your system, if you are getting errors run this command for windows OS)

> Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned

Step 30: Create a new project using NEST CLI. In your terminal, navigate to the directory CISE\_Repos/worksheet1 and then create the project and then run the following command:

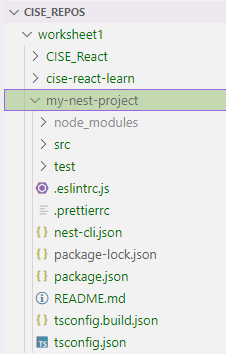
>nest new my-nest-project --skip-git

Replace "my-nest-project" with the name you want for your project. NEST CLI will create a new directory with this name and set up a new NEST project inside it (Select npm as the package manager).

**Your folder structure at this point should look like this:**

A screenshot of a computer

Description automatically generated



Note: You may want to add a “.gitignore” file into your Nest project folder. Since we are using npm as the dependencies management tool, with the config file (package.json) we can download all dependencies locally after cloning the remote repository. We do not want to upload dependencies to remote repository, as it can take a long time and a lot of space. By adding the .gitignore file, we are telling Git to ignore the contents listed in it.

You can copy and paste the ignore list provided by NestJS:

[nest/.gitignore at master · nestjs/nest (github.com)](https://github.com/nestjs/nest/blob/master/.gitignore)

A screenshot of a computer program

Description automatically generated

Step 31: Navigate into your new project directory using the command:

>cd my-nest-project

Step 32: Run the NEST application. Inside the project directory, type the following command:

>npm run start

Your NEST application is now running at `http://localhost:3000`. You can open this URL in your browser to see your application.

Step 33: Make some changes to the application. Open the project in your VS Code, and navigate to `src/app.controller.ts`. Here, you can change the string returned by the `getHello()` function. Save your changes and check your application at `http://localhost:3000` again to see the updated message.

Step 34: Once you have made your changes, stop the NEST application by pressing `Ctrl + C` in your terminal.

Now you can commit everything to this week’s repository.

**Name: Min Su Kim Date: 06/09/2024**

**Worksheet Evidence:**

This worksheet requires some answers to questions and **at least three selectively captured screenshots** as evidence. The aim is to be able to learn from the exercise, and evidence that.

**For each of your three selected screenshots (or sequence of shots) in a brief paragraph or two reflect on:**

* Why you have selected it?
* What have you learnt in this part of the worksheet?
* What was new or surprising?
* What useful external resource(s) did you consult and why? Provide a link(s) to the resource.

A computer screen shot of a program

Description automatically generated

* Why you have selected it?

I chose this screenshot to show my understanding of Git, pushing to my github repository, and knowing how to create and check branches.

* What have you learnt in this part of the worksheet?

I have learnt the baseline git workflow of adding, staging, committing and pushing changes to github.

* What was new or surprising?

I found it surprising that after learning it once, the streamlined approach to pushing to github is retained, and makes it easier for future commits.

* What useful external resource(s) did you consult and why? Provide a link(s) to the resource.

N/A

A blue and white logo

Description automatically generated

* Why you have selected it?

I chose this to show that the frontend application runs on my local server.

* What have you learnt in this part of the worksheet?

That frontend applications have to run on a local server, after thinking, I have also realised cloud platforms which host websites also run their web applications on local servers.

* What was new or surprising?

Creating a new React application was easier than expected, following the steps provided.

* What useful external resource(s) did you consult and why? Provide a link(s) to the resource.

A screen shot of a computer program

Description automatically generated

* Why you have selected it?

I selected this as this was actually the code (or file) which modified the getHello() string.

* What have you learnt in this part of the worksheet?

The AppController class is decorated with the @Controller() decorator, which marks it as a controller and allows it to handle incoming HTTP requests. The getHello method is mapped to a HTTP GET request, mapped to the root route, so the getHello() method is executed.

* What was new or surprising?

I learnt a new way of handling GET requests, as javascript and PHP have different ways of handling these with isset and using fetchAPI / async await.

* What useful external resource(s) did you consult and why? Provide a link(s) to the resource.

https://docs.nestjs.com/controllers

|  |  |
| --- | --- |
| **Evidence** | **Check** |

|  |  |
| --- | --- |
| 1. What is the purpose of Git and GitHub?   Git is a tool that helps developers manage changes and handling collaborations for their codebase, Github is a platform that hosts git repositories online, providing an environment where teams can collaborate on codebases and integrate other development tools. |  |
| 1. Explain the difference between a local repository and a remote repository in the context of Git and GitHub.   A local repo is a version controlled project stored on your local computer, a remote repository is hosted online, which enables collaboration and sharing with others. Changes are synced when you push and pull from remote repositories. |  |
| 1. What is the role of README.md file in a GitHub repository?   The README file serves as a introductory document that provides a summary and important information for the project, which can include usage examples, documentation, licensing information and installation instructions. It helps the user navigate and learn about the repository. |  |
| 1. Explain the purpose of creating branches in GitHub. A screenshot(s) to support your answer may be suitable to show your “Development” branch.     Branches allow developers to work on new features, tests and fixes without affecting the main codebase, allowing for a more streamlined approach for safer and secure codebase management, and encourages better collaborative development where multiple team members can work on different features of a project all at the same time. |  |
| 1. Explain the steps you took to merge your changes to the main branch on GitHub. A screenshot(s) to support your answer may be suitable.   Git add . git commit -m “commit message here” git push origin Development, then accept the pull request. |  |
| 1. Provide an example of one of your commit messages, adhering to a commit message standard. A screenshot here may be suitable.   “replaced homepage text, replaced app logo link” |  |
| 1. What is the "create-react-app" script used for? A screenshot(s) to support your answer may be suitable to evidence your successful creation of the cise-react-learn React application.   The “create-react-app” script is used to set up a new React application quickly. This is a readyt to use development environment which includes a React library setup bundled with the essential tools needed to contain necessary dependencies and tools for javascript compilation. However, the create-react-app dev environment has been depreciated for a while now, better recommended dev environments depending on the use case would be Vite, NestJS, NextJS, Astro, among others. |  |
| 1. What are the roles of the package.json and .gitignore files in a React application?   Package.json holds the projects metadata, dependencies and any scripts that need to be run in the application. .gitignore files ignore certain files when uploading different git versions. For example, the node\_modules folder can get extremely big in size if you continue to download different libraries. The .gitignore file also serves to ignore .env files or files that contain sensitive data. |  |
| 1. Explain the purpose of a pull request in GitHub. A screenshot(s) to support your answer may be suitable, to evidence the open pull request for merging the "LogoLink" branch with the Development branch.     A pull request used to merge a set of changes from one branch to another within a repository, which may involve code reviews, bug fixes and allows other team members to also review the pull request before confirming the changes to the main branch. It is used to ensure that any changes made and integrated into the main branch does not break the codebase and integrates successfully. |  |
| 1. A screenshot(s) to support your answer may be suitable to evidence the successful creation of a new project using NEST CLI and running of the NEST application. |  |
| 1. A screenshot(s) to support your answer may be suitable to evidence changes you made to the getHello() function in the src/app.controller.ts file of the NEST application and the updated message in the browser.     Alternatively, I could have manually changed the return string of getHello in app.controller.ts |  |