Guided Exploration 01: Set Up Environment, Version Control and Learning

**Purpose**:

* Communicate effectively in a variety of professional contexts
* Apply various tools and principles
* Apply computer science theory

**Points**: 40 (**see rubric in canvas**)

**Due Date**: See Canvas

**Effort: Collaborative and Individual**

* You will be collaborating with your team but individually **write your own code and answer the questions in your own words** . When you ask questions and explain to others you get a deeper understanding.
* Handing in something that is not your own work is not beneficial to your learning and will be a violation of ([Academic Integrity](https://docs.google.com/document/d/1w-JOtg74mWMsuwHzq-NuErUZCVKAl2fo/edit#heading=h.gjdgxs)) .
* Do not use someone else’s work or AI to write your answers or code. You can use it to clarify information or understand concepts better.

**Deliverables:**

* Upload this document as a pdf or word document with your answers for your Guided Exploration and a separate document for your professional communication. in canvas for this assignment.

[Description](#_qa44qyesmhm)

[Applying technology](#_6l9inb4e2cie)

[Explaining technical concepts and professional writing submission](#_wy9xwe7al9ev)

[Part 1: Learning New Technologies and Methodologies (Individual)](#_xv6jyo4j9nzt)

[1.1 Learning How to Learn](#_en4i4r167opd)

[1.2 Learning Computer Science and AI Tools](#_v1ok5ruo3jx3)

[Part 2: Developer Technical Documentation (Individual)](#_4lr6hykc4ng6)

[Part 3: Development Environment Exploration (Collaborative)](#_ycma38s0k73f)

[3.1 IDE](#_d31mzkq1qqxj)

[3.2 Command Tool](#_xcwanqikhxaf)

[3.3 Docker](#_qquuyavhq36x)

[3.4 Version Control (Git) and Cloud Based Storage (Github)](#_bvh2jr5ncf4l)

[4 Introduction to Web Apps](#_ofcxmllf3ppt)

[5 Professional Communication (Individual)](#_e3n5kf230hzu)

# 

# Description

## Applying technology

If you can’t complete the technical steps for a part of the assignment give a brief explanation of what went wrong, put a screenshot and meet with Deb.

## Explaining technical concepts and professional writing submission

* Be Concise When Answering
* Include Code Snippets and Command Line examples
* Add Media links such as videos and websites
* Include links AI resources used to understand technologies or concepts
* Make sure you answer in your own words

# Part 1: Learning New Technologies and Methodologies (Individual)

The reality of computer science is that there are so many languages, technologies and methodologies available and it is constantly evolving with new ones.

## 1.1 Learning How to Learn

Learning how to learn is an important component of this course as your career will be full of learning something new. One of my favorite websites is [Train Ugly - How To Get More Out Of Your Practice](https://thelearnerlab.com/train-ugly/).

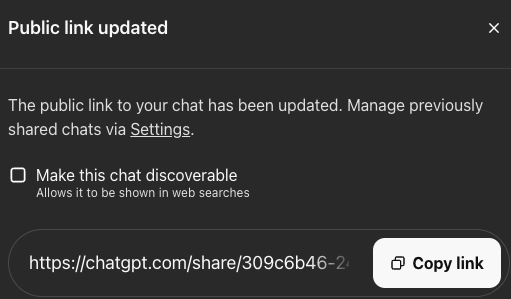
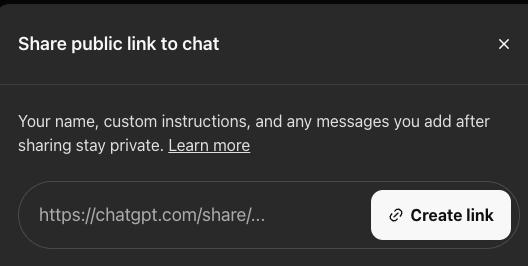
I have selected the following podcast for you to listen [**Desirable Difficulties - The Learner Lab**](https://thelearnerlab.com/desirable-difficulties/)

|  |
| --- |
| 1.1.1 Explain desirable difficulties and how you think it relates to the learning in this class and in your future career.   * A difficulty that I would see that could relate to this class is learning to understand the difference in work load between individual work, and work divided between teammates. Also, this could relate to a future career, because a company will want to see how well you work either on your own with a team, and what is better for the company. |
| 1.1.2 Explain the difference between learning and performing   * Learning is understanding new concepts and material to better your knowledge with a certain concept or concepts. For example, learning does not mean you are suppose to know and understand the problem immediately, but instead find and learn you ways to get to a solution to a problem/concept you are wanting to better grasp information on. * Performing is being able to use these resources and knowledge gain in order to create something new/better out of a situation. An example of this would be trying to create a new app with the knowledge you have or have gained from previous research done. Then continuing to pursue this until you have a desired outcome. |
| 1.1.3 Summarize three strategies you could incorporate in your learning to help you learn different languages, technologies and methodologies.   * Integrate learning from other domains to make connections between new and existing knowledge. So, when learning a new programming language, for instance, make comparisons and contrasts with the languages you are already familiar with. This method helps could help us to recognize the wider applications of what you would be studying in addition to deepening the grasp of it. * Over time, possibly revisiting content at progressively longer intervals by using a method known as spaced repetition technique(s). In doing so, information could be transferred from short-term to long-term memory. For example, Flashcards would pose a relative idea as a long-term study method. * Another idea would be to work on practical tasks or exercises that are connected to the material you are learning, and get a hands-on experience. In terms of languages, this could entail using the language for writing, speaking, or coding. Constructing little projects or finding solutions to issues can help to better increase critical thinking on future ideas/problems. |

## 1.2 Learning Computer Science and AI Tools

Read

* [9 Practical Uses of ChatGPT in Programming](https://www.makeuseof.com/chatgpt-programming-practical-uses/)
* [Companies Banning AI Platforms Like ChatGPT in 2023](https://tech.co/news/tech-companies-banning-generative-ai)
* Include resources for example for ChatGPT
  1. Click upload button in upper right corner
  2. Click Create link
  3. Copy link
  4. Paste link



|  |
| --- |
| 1.2.1 Explain how AI such as chatgpt can be used to help you learn.   * A great example of how it can help you learn is if you do not understand a problem and you do know which approach to take into tackling the problem; you can ask AI to give a suggestion and background information of the best method for doing so. * Also, if information that is needed to be learned, but you have a harder time understanding it, then you have AI go ahead and try to list example and have it explain it better to yourself. * AI can also provided great resources that are or could be relevant to what is need for your learning. |
| 1.2.2 Explain how AI such as chatgpt can be used that would hinder your learning.   * ASKING IT FOR A STRAIGHT ANSWER! I have been known in the past to do so when AI first became new to us, but quickly learned that I was not learning anything at all and would seem to start to fall behind from others. Now I have learned that AI should be used only for gathering info and different perspectives on ideas/problems. |
| 1.2.3 What do you think is meant by not blindly copying and pasting code and explanation from websites such as stack overflow or AI tools such as chatgpt?   * Not blindly copying and pasting code or explanations from websites like Stack Overflow or AI tools like ChatGPT means that you should take the time to understand the code or information before even using it. * Using any piece of code would mean that you should ensure you understand what it does, how it works, and why it is an appropriate solution for your problem. This would involve possibly reading through the code, understanding the logic, and maybe even thinking about how it integrates with the project. |
| 1.2.4 Use an A.I. tools like ChatGPT and ask it the following   1. Describe docker, what it provides and the benefits. 2. Provide me with free online resources to learn more?   Summarize a description of docker and the benefits. Include the AI source url and any other web resources you used   * Docker is a platform that makes managing, growing, and deploying applications easier. Applications are bundled with their code, dependencies, and runtime libraries in these isolated environments known as containers, which allow the application to execute consistently on a variety of computer setups. * <https://chatgpt.com/share/93a4a6b7-6485-4c4e-ace1-a244b473854e> |

# Part 2: Developer Technical Documentation (Individual)

Another important skill is organizing and keeping technical documentation on all the different technologies, languages and methodologies. Developer documentation provides developers with comprehensive software information, which they utilize to understand, develop and interact with the software they are creating

Read the following:

[Improve Developer Documentation: Tips & Tricks](https://www.archbee.com/blog/improve-developer-documentation) and follow the first 4 tips in the article. Apply these when answering your questions for the remaining guided exploration.

|  |
| --- |
| 2.1 List the first 4 ways to improve developer documentation as listed in the article. These should be followed in your developer documentation.  - Be Concise When Writing  - Avoid Using Technical Jargon  - Include Code Snippets  - Choose a Good Documentation Tool |
| 2.2 Summarize the benefits of developer documentation.   * By offering a clear reference for every team member, developer documentation enhances code quality, speeds up onboarding, and promotes collaboration. By increasing the readability and accessibility of the codebase, it diminishes knowledge silos, facilitates external users, and eases maintenance. It also helps guarantee that security rules and standards are followed, fosters community growth in open-source projects, and increases productivity by reducing the amount of time spent searching for information. All things considered, producing high-quality, scalable software is made easier with efficient documentation. |

# Part 3: Development Environment Exploration (Collaborative)

Collaborate with others in class in person or through discord but each person should complete and answer in their own words.. Feel free to use websites, videos and A.I. tools such as chatgpt to help you and include resources with your answers.

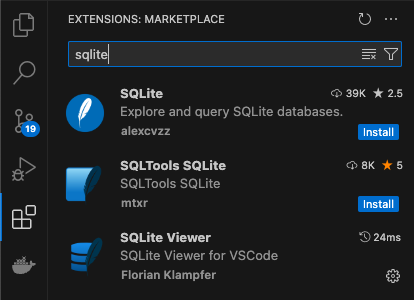
## 3.1 IDE

I recommend the following IDEs for this class if you need support but feel free to use what you prefer for a Ruby on Rails development environment.

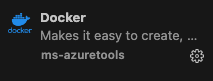
* [Visual Studio Code](https://code.visualstudio.com/) (Windows OS and Linux; note: Mac OS support ending August 31, 2024 [What's happening to Visual Studio for Mac](https://learn.microsoft.com/en-us/visualstudio/mac/what-happened-to-vs-for-mac?view=vsmac-2022) ) or [VSCodium](https://vscodium.com/) : Windows, Mac and Linux

We will utilize various extensions for Visual Studio Code or VSCodium. Here is an example of one.

* Search SQLite Viewer



* **Search Docker and install**

****

## 3.2 Command Tool

Recommendations for selecting a command tool.

* **Windows**: WSL 2 + Docker Desktop for a seamless Linux-like development experience
* **Mac/Linux**: Terminal
* **Windows/Mac/Linux** : Git Bash + Docker Desktop for a simpler setup with a Unix-like shell.

Tutorials to help you learn command line basics to navigate the file system and manage files

* [Learn the Command Line | Codecademy](https://www.codecademy.com/learn/learn-the-command-line)
* [Linux Journey](https://linuxjourney.com/) -> [1. The Shell](https://linuxjourney.com/lesson/the-shell)

Answer the following in your own words.

|  |
| --- |
| 3.2.1 Paste a screenshot of your command line tool.  - |
| 3.2.2 Find resources (websites, images, videos) for command line basics to navigate the file system and manage files. Put the resources below and include an summarize these commands  - I found that <https://www.codecademy.com/learn/learn-the-command-line/modules/learn-the-command-line-navigation/cheatsheet> provides me short and basic explanation commands listed below:  - pwd: exhibits current working directory.  - ls: Displays all files and/or any directories in systems directory.  - cd: Changes directory.  - mkdir: Creates new directory.  - rmdir: Deletes/removes a directory.  - touch: Creates or updates a file.  - rm: Deletes files or directories.  - cp: Copies files or directories.  - mv: Moves files or directories.  - find: does a search for any files or directories.  - The materials offered provide a thorough rundown of all the necessary command-line options for file management and navigation on Unix-like platforms. While the FOSS Linux article and Red Hat Enable Sysadmin tutorial offer more in-depth explanations and examples, the Codecademy Command Line Cheatsheet offers a short reference for a variety of tasks. |

## 3.3 Docker

Familiarize yourself with installing docker desktop for your operating system and architecture

* [Install Docker Desktop on Windows](https://docs.docker.com/desktop/install/windows-install/)
  + Install Linux on Windows with WSL.
  + [How to check if processor is ARM64 or x64 (64-bit) on Windows 11 - Pureinfotech](https://pureinfotech.com/check-if-processor-arm64-x64-64bit-windows-11/)
* Mac [Install Docker Desktop on Mac](https://docs.docker.com/desktop/install/mac-install/)
  + [How to Check if Your Mac Is Using an Intel or Apple Silicon Processor](https://www.howtogeek.com/706226/how-to-check-if-your-mac-is-using-an-intel-or-apple-silicon-processor/)
* [Install Docker Desktop on Linux](https://docs.docker.com/desktop/install/linux-install/)

Complete all parts of getting started

1. [Getting started | Docker Docs](https://docs.docker.com/guides/getting-started/)
2. [Get Docker Desktop](https://docs.docker.com/guides/getting-started/get-docker-desktop/)
3. [Develop with containers | Docker Docs](https://docs.docker.com/guides/getting-started/develop-with-containers/)
4. [Build and push your first image | Docker Docs](https://docs.docker.com/guides/getting-started/build-and-push-first-image/)
5. Pull the Docker Image for the next module  
   docker pull debmhteach/cs3170\_rails

Answer the following in your own words.

|  |
| --- |
| 3.3.1 Paste a screenshot of your remote docker repository.  Unfortunately, I was able to pull the information, but I was not able to get the image running and working. Reason being is because of the Chip in my computer if I am not mistaken. Gives me an error saying ARM64, which leads to me not being able to pull up the imagine properly. I have found that I can get it running on a sub-Linux system (WSL), but then once I try to go to the local host it still does not show. |
| 3.3.2 Paste a screenshot of your folder and files you accessed through the docker image after completing the task [Develop with containers | Docker Docs](https://docs.docker.com/guides/getting-started/develop-with-containers/) . |
| 3.3.3. What is an image? What is a container?  - An image is known to be like a blueprint or a template. Its basically is to help the developer create a what is known as a container. It contains the necessary instructions for running a specific application.  - A Container is more so an environment that provides a place for the application to execute. |
| 3.3.4 What is Docker, and how does it support software development and deployment?  - Docker is a useful tool for software development and deployment because it offers a strong and adaptable foundation for creating, executing, and managing applications. |
| 3.3.5 Explore these commands   * docker pull * docker run * docker ps * docker stop * docker restart   Find resources (websites, images, videos) for using the different docker commands. Put the resources below and include a brief description and examples for the commands.   * So, I decided to switch over to the AI, Gemini, to see how it differs from ChatGPT, and I am blown away. First this is a public link to me asking it with the following questions to this: <https://g.co/gemini/share/959f5e75fcbb>. This AI uses its tool very well, and I have found that it provides the best information for questions asked to it. * **1. Docker pull** * Downloads and brings (pulls) a Docker image from registry * ex: docker pull ubuntu:latest (This command pulls/downloads the latest version of the Ubuntu image) * **2. docker run** * Creates and starts a running container based on a downloaded image. * ex: docker run -it ubuntu:latest bash (runs the Ubuntu image with a bash shell) * **3. docker ps** * Lists all Docker containers on your system * ex: docker ps * **4. docker stop** * Stops a docker container that is currently running. * ex: docker stop spotted\_zebra (stops the container with ID spotted\_zebra) * **5. docker restart**   Restarts the previously stopped Docker container.   * ex: docker restart spotted\_zebra (restarts the container with ID spotted\_zebra) |

## 3.4 Version Control (Git) and Cloud Based Storage (Github)

Read [About GitHub and Git](https://docs.github.com/en/get-started/start-your-journey/about-github-and-git) and complete the following

* [Creating an account on GitHub](https://docs.github.com/en/get-started/start-your-journey/creating-an-account-on-github)
* [Hello World - GitHub Docs](https://docs.github.com/en/get-started/start-your-journey/hello-world) Learn the "GitHub Flow", and the key principles of collaborative working (branches, commits, pull requests, merges).
* [A Git/GitHub beginner tutorial – 4sysops](https://4sysops.com/archives/a-gitgithub-beginner-tutorial/)

|  |
| --- |
| 3.4.1 What is git and github and how do they support building software solutions? What is the difference between them?  - **Git**: A distributed version control system used mostly for tracking changes to source code in software development. It is a powerful tool that makes effective teamwork, version control of the code, and error correction possible when necessary.  - **GitHub**: Where developers can exchange and work together on code using Git on GitHub, an online hosting platform. It is known to be a more well-liked option for software development teams because of its many options/capabilities.  - **Difference**: The platform that offers developers a collaborative environment to use Git efficiently is called GitHub, whereas Git is the engine behind version control. These resources work together to create a potent toolkit for developing and managing software solutions. |
| 3.4.2 Paste a screenshot of your remote repository you created for [A Git/GitHub beginner tutorial – 4sysops](https://4sysops.com/archives/a-gitgithub-beginner-tutorial/) . |
| 3.4.3 Explain a .gitignore file.  - A text file that instructs Git on certain directories or files to pass by while pushing updates to a repository. This is helpful for removing temporary files, files created during your build process, and files you do not want version control to track. |
| 3.4.4 Explore the commands: status, log, clone, add, commit, push, pull, tag, branch, checkout  Find resources (websites, images, videos) for using the different commands. Put the resources below and include a brief description and examples for the commands.   * **Status** * shows current state of the working directory and staging area * <https://git-scm.com/docs/git-status> * ex: git status (Should show any untracked files, modified files, or even staged changes * **Log** * shows commit history made in repository * <https://git-scm.com/docs/git-log> * ex: git log –fine=format: ‘%h %s’ (should display the commit hash and subject line for each commit * **Clone** * Makes a local copy of a repository * <https://github.com/git-guides/git-clone> * ex: git clone <https://github.com/Ckopcik-msu/CS_3710.git> (should make a copy of the repository named CS\_3710) * **Add** * Stages changes to be committed (From what I remember almost like notes made for the team) * <https://git-scm.com/docs/git-add> * ex: git add (Stage changes) * **Commit** * Creates a commit from staged changes with a message added on of the developers choice. * <https://git-scm.com/docs/git-commit> * ex: git commit -m “This is a first-time commit” (New commit with message “This is a first-time commit”.) * **Push** * Sends local commit to repository * <https://git-scm.com/docs/git-push> * ex: git push origin master * **Pull** * Gathers and grabs changes from repository and merges them into the most current branch * <https://git-scm.com/docs/git-pull/en> * ex: git pull origin master * **Tag** * Creates a tag on a specific commit * <https://git-scm.com/book/en/Git-Basics-Tagging> * ex: git tag CS\_3710 * **Branch** * Creates a new branch * <https://git-scm.com/docs/git-branch> * ex: git branch CS\_3710\_WEB\_APP\_DEV (Will create a new branch by the name of “CS\_3710\_WEB\_APP\_DEV”) * **Checkout** * Stitch from one branch to another * <https://git-scm.com/docs/git-checkout> * ex: git checkout CS\_3710\_WEB\_APP\_DEV (will switch from main branch to “CS\_3710\_WEB\_APP\_DEV”) |

# 4 Introduction to Web Apps

Module 1 introduced you to some of the fundamentals of a web app that you will be building on in future modules. Answer the following and include resources (websites, images, videos) for future reference as we go deeper into these concepts.

|  |
| --- |
| 4.1.1 How does a web application typically interact between the client and the server?  - Online applications frequently use a client-server architecture to help clients and servers communicate with one another (usually via a web browser). The client, which can be your web browser again, then sends an HTTP request to the server. This request contains the additional information headers, the requested URL, and the HTTP method (GET, POST, PUT, or DELETE).Upon receiving this request, the server assesses it and determines the appropriate course of action. It could generate dynamic content, import data from a database, or process a form submission. The server then sends an HTTP response to the client.  - Resources:  <https://stackoverflow.com/questions/11170457/discussion-client-server-application-architecture>  <https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works> |
| 4.1.2 What are the key components of an HTTP request and response?  Request:  - Method: Specifies or should specify actions to be performed  - URL: Identifies the resource to be accessed.  - Headers: Contains additional information about the request  - Body: Optional content to be sent with the request, often used for form submissions or data uploads.  Response:   * Status Code: Indicates outcome of a request * Headers: Contains information about the response * Body: The content of the response, which can be HTML, JSON, an image, or other data   Resources:  <https://developer.mozilla.org/en-US/docs/Web/API/Response/body>  <https://www.w3schools.com/tags/ref_httpmethods.asp> |
| 4.1.3 What is a URI, and how does it relate to routing in a web application?  - A character string/list, known as a URI, is what is used to identify a resource in a unique way.  - Resources:  <https://en.wikipedia.org/wiki/Uniform_Resource_Identifier>  <https://developer.mozilla.org/en-US/docs/Glossary/URI> |
| 4.1.4 What is REST, and why is it widely used in web APIs?  - The web API architecture approach known as REST (Representational State Transfer) places a strong emphasis on scalability, simplicity, and the usage of HTTP methods to describe various tasks.  - Because of its ease of use, scalability, and interoperability with HTTP, REST is frequently utilized in online APIs. It gives users an easy-to-use interface to communicate with web services.  - Resources:  <https://www.youtube.com/watch?v=BRdcRFvuqsE>  <https://developer.mozilla.org/en-US/docs/Glossary/REST> |
| 4.1.5 What are some of the key features provided by the browser's developer tools?  - Elements panel: Allows you to examine the HTML structure of a page, edit elements, and even inspect styles. - Console panel: Provides a JavaScript console for executing code and debugging errors - Network panel: Provides details on HTTP requests - Sources panel: Allows you to view and edit source code - Performance panel: Measures the performance of a webpage  - Resource: <https://developer.chrome.com/docs/devtools/open> |

# 5 Professional Communication (Individual)

**These are individual responses and should not use AI.**

This is just a start of your understanding of different architectures, design patterns and concepts used when learning a framework to build SaaS apps.The goal is to see how you grow over the semester in your learnings.

Create a separate professional document (word or pdf) where main ideas are clearly presented with supporting evidence.

* **Summarize Key Concepts:** Clearly and concisely explain the key computing concepts covered in this module. Ensure that your explanations are in your own words, using accurate and appropriate terminology.
  + What is client/server and Web Application architecture?
  + What are the key components of an HTTP request and response?
  + What is a URI, and how does it relate to routing in a web application?
  + What are CRUD operations in relation to web apps?
  + What is REST, and why is it widely used in web APIs?
* **Include Examples:** Provide relevant examples to support your explanations. These could include code snippets, command examples, or real-world applications.
* **Visual Aids:** Create and include at least one original visual aid (such as a diagram, screenshots, or chart) that help illustrate and clarify the concepts discussed in your document.
* **Cite Sources:** Use at least four credible and relevant outside resources to support your explanations. Ensure that all sources are properly cited with active links included.
* **Organize Your Document:** Structure your document with clear headers and subheaders to guide the reader through the content. Ensure that the flow of ideas is logical and easy to follow.