

INFO1111: Computing 1A Professionalism

2022 Semester 1

Practice: Team Project Report

Submission number: 02

Team Members:

Name	Student ID	Levels being attempted in this submission
Finn Gladstone	510448570	3,4
Anh Ho	510442958	1,2
Adam Jankelowitz	510441180	3,4

Contents

1.	Level 1: Basic Skills	4
1.1.	Developing industry skills	4
1.2.	Skills: Anh Ho : Computer Science	5
1.3.	Skills: Adam Jankelowitz : Data Science	7
1.4.	Skills: Finn Gladstone : Software Development	8
2.	Level 2: Basic Technology	11
2.1.	LAMP Tech Stack: Adam Jankelowitz	11
2.2.	MEAN Tech Stack: Anh Ho	11
2.3.	ASP.NET Tech Stack: Finn Gladstone	12
3.	Level 3: Advanced Skills	14
3.1.	Advanced features: Adam Jankelowitz	14
3.2.	Advanced features: Finn Gladstone	18
3.3.	Advanced features: Anh Ho	19
4.	Level 4: Advanced Knowledge	20
4.1.	Advanced Knowledge: Adam Jankelowitz	20
4.2.	Advanced Knowledge: Finn Gladstone	22
4.3.	Advanced Knowledge: add student 3 name here	23
5.	References	24

General Instructions

You should use this L^AT_EX template to generate your team project report. Keep in mind the following key points:

- When we assess your report, you are not given a mark. Instead we will indicate (separately, for each team member) whether each level is "achieved".
- In order to pass the unit, you must achieve at least level 1.
- In order to achieve level 2, you must first have achieved level 1, and so on for each level up to level 4. This means that we will not assess a higher level until a lower level has been achieved (though we will review one level higher and give you feedback to help you in refining your work).
- Some parts of the report are completed as a team and other parts require each student to complete a different section. This means that for each submission, some members of the team may have completed their work for a given section, but other members may not. It also is therefore possible that some members of the team may achieve a specified level and other members of the team may not yet have achieved that level.
- Even if some members are completing their material for a given level, and others are not, your team members will still need to work together to edit and compile the report. The only exception to this is where a member of the team has already achieved the level they are targeting in a previous submission and has decided to not attempt higher levels, and so is not contributing any further (this should be obvious because no level is indicated for that student on the cover page).
- When completing each section you should remove the explanation text and replace it with your material.

For each submission you will add new details to this report, and/or update previous sections (where previous work was not good enough to have achieved the relevant level). In particular:

- **General:** For each submission, each student can attempt up to 2 levels. You must also successfully achieve each lower level before you can be assessed at a higher level. For example, in the first submission you might attempt only level 1, but not be successful in achieving that level. You then reattempt level 1 and add in level 2 in the second submission and are successful in achieving level 1 but not level 2. For the third and final submission you could then attempt level 2, or levels 2 and 3 - or even just choose to not submit anything further and remain at level 1).
- **Submission 1:** You should complete at least the material for level 1 (since achieving level 1 is required to pass the unit). Each member of the team can also optionally choose to complete the material for level 2.

Note 1: If you do not complete the level 2 information then you obviously cannot achieve level 2 at this stage. This does not stop you from attempting level 2 in Deliverable 2 or 3, but it will make it more difficult to achieve the higher levels later in the semester. Note 2: To be able to achieve Level 1 in submission one your team has to achieve level 1 in the group component (Section 1.1) and you have to achieve Level 1 in the individual component (i.e. your assigned section 1.2, 1.3, 1.4 or 1.5)

- **Submission 2:** Each member of your team will complete additional sections, but because you are submitting a single document, you need to work together to compile your results together and generate the final submission.

If you did not achieve level 1 in your first submission, then you should revise the material for level 1 based on the feedback, and optionally you can also complete level 2.

If you achieved level 1 in your first submission, then each team member can optionally complete the material for levels 2 and 3. *Note: If you do not achieve level 1 with this submission then the highest level you will be able to achieve in the final submission will be level 2. If you achieve level 1, but not level 2, with this submission then the highest level you will be able to achieve with the final submission is level 3.*

- **Submission 3:** Again, you can correct sections where you did not achieve the specified level in the previous submission, and you complete additional sections.

If you still have not achieved level 1, then you should revise the material for level 1 based on the feedback, and again optionally you can also complete level 2.

For those at level 1, you can choose to complete the material for levels 2 and 3.

For those at level 2, you can choose to complete the material for levels 3 and 4.

For those at level 3, you can choose to complete the material for level 4.

Whilst the team project is just that – a team project – it has been designed to also allow different members of the team to achieve different outcomes. We do expect you to work together as a team. If you do come across problems working together then the first step should be to discuss this with your tutor. Note: If you are having problems you should approach your tutor as soon as you can to make them aware of the difficulties you are having with your team.

Finally, you should also ensure that any resources you use are suitably referenced, and references are included into the reference list at the end of this document. You should use APA 6th reference style (?, ?).

1. Level 1: Basic Skills

Level 1 focuses on basic technical skills (related to \LaTeX and Git) and the types of skills used in different computing jobs.

1.1. Developing industry skills

Five approaches to continual learning:

Personal projects in unfamiliar territory: Undertaking a personal interest project in a new language or package is an effective example of continual learning.

In the project format, the user needs to not only obtain a certain technical proficiency but be able to transfer those skills into a real-world program (Ikechukwu, 2020).

This approach would be effective when the user needs to achieve a high technical proficiency in a certain area, more so in niche spaces (for e.g., derivations of neural networks; RNN, CNN, ANN etc).

The main downside of this approach is the lack of external motivation or a mentor, and the user needs to direct themselves on project timeframe, work efficiency.

2. Online bootcamps & certs: Online coding workshops are aimed at equipping the user with some of the most in-demand skills in the industry in a time-efficient manner (Columbia, 2019).

Bootcamps teach a variety of skills, such as popular front end languages (HTML, CSS, JavaScript) all the way to back end development (SQL, APIs).

Bootcamps are advantageous in their fast-paced environment and use of a structured learning path for the user.

However, some fail to teach more conceptual skills, such as Data Structures & algos (Shipley, 2019), and they generally come with a high upfront cost.

3. Conversations & questions in the workplace: Asking more questions and observing mentors in the workplace, coined as Social Learning, is a more dynamic and casual process through which individuals can work to continuously improve their skillset (Eira, 2019).

This is an effective method for continuous learning in a less strict or confining environment, and one that helps to foster soft skills (communication, team dynamics, collaborative problem solving).

Hence this approach might lack the technical comprehensiveness of a course but has the side benefit of training the user's soft skills in the workplace, which are crucial for success (Erstad, 2017).

4. Going to seminars and talks: Seminars are a great way of being introduced to new projects, techniques and industry news.

They allow the user to gain more knowledge specific to their profession and stay on the bleeding edge of industry progression, as well as network and get introduced to new fields and peers (Allconference, 2021).

Seminars also provide a learning environment that highly differentiates itself from the classroom (Panigrahi).

Therefore seminars are effective in keeping users up-to-date and providing networking opportunities, but can lack a comprehensive technical facet.

5. Reading articles and journals: Rote reading is another method to continually learn about one's field.

This includes a variety of media (books, scientific journals, news edits) and presents an easily and effectively digested source of information.

While reading is effective to gather a deep conceptual or theoretical understanding of a certain topic, follow-up practical exercises would be necessary to achieve the greatest learning outcome.

1.2. Skills: Anh Ho : Computer Science

Mathematics Mathematics is important for computer science because it teaches students how to use abstract languages, how to work with algorithms, how to self-analyze computational thinking, and how to accurately model real-world solutions (Sedlacek, 2016). Such practical mathematical skills can be applied to computer programming which is an integral part of computer science.

Furthermore, widely used programming languages within the field of computer science are very similar to the mathematical language that students learn in math class. From simple equalities to complex mathematical formulas, having practical mathematical skills allows students to conduct reading, comprehending, formulating thoughts, and communicating with abstract language.

Data Analysis Basic data analysis skills are necessary for computer scientists as the field of data analysis is growing year by year. Data plays a key role in many advanced computer science practices, revolving around artificial intelligence, predictive analysis and many more.

Computer scientists working in the fields of probabilistic, deterministic, and statistical machine learning must apply statistical algorithms and probabilistic models to train computers and software (MTU, 2018). This allows programs to make efficient decisions and collate the right data. A computer scientist who understands applied statistics will benefit greatly as this will improve their work in programming.

Software Development Computer science professionals or students must possess programming skills to conduct software development and engineering principles, this is because developing software utilising programming skills is an integral part of the field of computer science.

Such skills include the end-to-end software development process - writing programs using common software languages, usability testing, and ultimately executing programs. Programming skills can be in multiple different languages such as C++, JavaScript, Python and HTML5, with such skills computer science professionals or students can create interface programs with utility.

Programmers must utilise their practical skills in software development to consider the expected outcomes of their software, using their creativity and problem-solving skills to meet design specifications and technical challenges.

Data Visualisation While the ability to analyse data effectively is important, it is also essential that computer science professionals be able to properly visualize data. This is recommended for computer science students or professionals as in the future employers/stakeholders may want an efficient method to visualise the effectiveness of certain programs. This includes translating raw data into graphs, charts, and other visual tools to help communicate results.

Furthermore, visualization is at the heart of advanced analytics (Brush, 2020). When computer scientists are writing advanced predictive analytics or machine learning (ML)

algorithms, it becomes important to visualize the output to monitor the results and ensure the model is performing as expected.

Website Development Web development is the process of creating a website, from design to implementation to maintenance. The result of a web development process has changed significantly over the last few decades. Practical skills revolving around website development can be very beneficial to computer scientists because it gives them the ability to creatively express an idea, displaying and exhibiting thoughts and plans to the entire globe.

Web development is also becoming a very profitable industry, in which if someone has website development skills, there are a plethora of opportunities in well known companies. Website development ensures professional growth that is incomparable to any other practical skills in the IT industry. And considering how most computer scientists have high critical thinking and programming skills under the belt, learning how to develop websites aligns with their skill set.

Stock Trading The nano-speed computers utilised by traders today make it possible to mediate transactions in nanoseconds, much faster than humans. Since the algorithm was introduced to the market, stockbrokers have been able to make better decisions about which stocks to buy or sell.

This presents a huge opportunity for computer scientists to take advantage of their programming skills and also learn how to trade stocks on a brokerage site. One practical skills of stock trading are learnt, computer scientists can develop algorithms and programs which can accurately predict the expected trend of a certain stock either short term or long term. Such programs can also manage risk and quickly calculate numerous ratios which indicate whether a certain investment is profitable on the long run or not.

Furthermore, computer scientists can pursue algorithmic trading which is commonly referred to as ‘automatic trading’, this process uses a computer program that follows a defined set of instructions and automatically places trade.

Technical Writing Technical writing is the ability to put the technical details of your work and insights into simple words (Vadapalli, 2020). It’s an underrated skill that is in high demand, and corporations require computer scientists to be good at technical writing. As a computer science professional, you need to write briefings, proposals, reports, and other important technical documents. As such, mastery of technical writing is a commendable skill for aspiring candidates.

Being able to master technical writing can be a make-or-break factor in regard to promotions in a job. Therefore, technical writing is a practical skill that is important for computer scientists in order to climb the ladders of the professional world.

Electrical Engineering Many computer science students may want to focus their careers onto computer hardware, as such, being proficient with computer components will help. Such skills can include understanding the relationships between components and knowing how to connect parts such as the processor, RAM, cache memory and graphics cards.

This practical skill aligns well with computer science because computer software assists engineers in creating and optimizing electrical equipment systems. Therefore, understanding how to program as well as building computing equipment can be beneficial in achieving long-term career goals.

1.3. Skills: Adam Jankelowitz : Data Science

SQL Queries The ability to write SQL Queries and build data pipelines is a crucial skill for a data science graduate. The first reason why this skill is important is that the graduate will be flexible, having the ability to build fundamental pipelines, thus allowing one to improve insight into the data derived (Shin, 2022) Another reason that this is a key skill for a graduate is to gain independence. With this skill, one will be able to write queries for a project or model that does not exist. This makes the graduate more valuable for employees as their independence and ability to not rely on others' previous work will prove to be efficient for the graduate.

Teamwork Teamwork is another critical practical skill that a graduate who majors in data science should have. As a data scientist, one will be working with business executives to devise strategies, working as part of a team to create a new product, and also working alongside software developers to improve efficiency and create data pipelines and structures. These different tasks require strong teamwork skills in order to achieve positive outcomes for the company. Whilst working as a data scientist, one will face many challenges, so the ability to work as a team and collaborate, as well as listening to one another will guide one to be successful whilst working in the industry of data science.

Version Control Version Control, and specifically the ability to use Git is a key skill that a data science graduate should have. Git is a cloud-based repository that can store files and folders, being one of the main version control systems used worldwide (Shin, 2022) . Git is an essential practical skill that one should know as it allows collaboration, the ability to bounce off each other's ideas, and also the capability to work on projects in conjunction with others. Furthermore, Git will also allow the entire team to keep track of old versions of their project, the productivity of the individual/team, and also allows one to go back to older versions of the project if required.

Programming Knowledge Another key skill that a graduate of a data science major should have is programming knowledge. Python is one of the most popular programming languages that is used by data scientists due to its easy-to-use nature as well as its access to multiple libraries which are suited for data science projects. If a graduate has a good sense of programming knowledge, they will be able to use different platforms with emerging technologies, enabling one to be efficient when developing programs and projects. A data scientist with a background in programming will be extremely self-sufficient, not relying on others and outside resources to complete tasks and work with data sets. For example, with the knowledge of programming, one will be able to query data individually, without requiring to collaborate with a software engineer.

Data Wrangling Data wrangling is the process of cleaning and unifying complex data sets to ensure easy access, readability, and analysis. In the workplace, a data scientist will receive large data sets, sometimes being stored in different formats or on different devices. Due to this, one will require to merge the data into one clean set, ensuring accessibility and readability. Furthermore, if a business wishes to analyse incomplete or unclean data, the analysis made will be inefficient and unreliable. Despite sometimes being time-consuming, data wrangling is a crucial skill that a graduate requires as it ensures the data is reliable and readable before being analysed. Therefore, this practical skill is crucial for a data science graduate and will be used constantly when working in the data industry.

Statistics Statistics and statistical analysis are crucial skills that one should know in the modern world. The ability to predict, estimate and analyse trends is a key skill that a data science graduate should know, and will use every day in the industry. Through the knowledge of statistics, one will have the power to derive key insights and solve problems in all industries (University of Virginia, 2022). The ultimate goal of a data scientist is to create value and gain more insight out of data. This can only be achieved by understanding the trends in the data extremely well. In the industry, finding structure and making predictions are the most important steps in data science (Weihs Ickstadt, 2018)). One branch in statistical analysis is classification methods, which is the principal for finding and predicting subpopulations from data. This key concept will be used constantly in the industry along with regression and statistical modelling.

Data Visualisation Data Visualisation is the ability to transfer data into a visual image such as graphs or charts. This skill makes the data readable and makes it easier to comprehend, giving one the ability identify trends and outliers within data sets. As a data science graduate working in the industry, data visualisation is a key skill that employers will be looking for as they will require an efficient way to analyse data. (ANALYTIKS) Data visualisation has a positive affect on the decision making of a company as the data is represented visually, allowing companies to recognise trends quickly. Without the ability to express data in visual form, one may find it challenging to identify relationships within data sets, as well as trends that may form over time. Therefore, a graduate who has the skill of data visualisation will be extremely successful in the industry as data visualisation ‘represents one of the fundamental tools of modern data science (Unwin, 2020).

Intellectual Curiosity Intellectual Curiosity is a crucial, non-technical skill that a graduate who majored in data science will require when working in the industry. When working as a data scientist, one will need to be able to ask questions in order to gain further insight into the specific project or data they are analysing. Intellectual curiosity is defined as “one’s desire to acquire more knowledge” (9 Must-have skills you need to become a Data Scientist, updated - KDnuggets, 2022). Data science is an area that is constantly evolving and in order to keep up with new advancements, one must be inquisitive and seek to constantly update their knowledge through research. Furthermore, curiosity will enable one to search through the data to find answers in multiple ways, as well as gain more insights into the trends that have been analysed.

1.4. Skills: Finn Gladstone : Software Development

Communication The ability to communicate in the workplace, both written and orally, is crucial to achieve effective collaboration (Rosello, 2017).

Tasks that require successful communication stretch from basic problem solving (e.g. gathering advice on a coding problem) all the way to delegating tasks and conversing with clients and higher management.

Communication also extends to team psychology and understanding team members. Being able to listen to others and restrain from interrupting, giving and receiving feedback without ego and asking well-structured questions can drastically influence how well a team gets along and, in turn, bring results to the business (Vlasiuk, 2021).

Problem Solving Software engineering boils down to problem solving: how one takes a problem, identifies how a person would solve it, and then translates their method into code (Hakes, 2021).

Employers will assess one's ability to abstract; that is, separate the problem from its real-world connotations and translate entirely to a logical or mathematical function, through which a computer can understand and process a solution (Runestone).

Problem solving also encapsulates *Complexity Hiding*; correctly identifying the unnecessary and potentially convoluting details from a task, and removing them to perform a correct implementation (Utah CS).

Therefore software engineers should aim to continually develop their ability to abstract and approach tasks in a confident and efficient way to maintain high productivity.

Object-Oriented Design Object oriented design is a school of thought for programming which organises programs around data fields, or Objects (S. Gillis, 2021).

This approach to programming is more applicable to certain languages than others (e.g. Java) and can be implemented to reduce redundancy and make code more modular.

These attributes are key in large scale projects where minimising runtime through modular code leads to better space and time complexity.

Even working outside traditional object-oriented languages, the ability to rescale ones work into multiple separate working entities is good working practice.

Data Structures & Algorithms This refers to the conceptual understanding of how a piece of code, or an algorithm, interacts with the various types of data structures programmers use, and how efficient and correct said algorithm is (Gupta, 2020).

The ability to work with data structures indicates to employers that the programmer understands not just how their piece of code works, but also why it works the way it does (runtime, memory usage) and how to improve and optimise its efficiency and usage.

Similarly to object-oriented design, it is also more a conceptual skill which means it is transferable across many languages and improves the programmer's overall aptitude.

Data Analysis The ability to interpret, clean and analyse data efficiently over a variety of platforms is very useful to the modern programmer and, in turn, large companies.

It provides a framework for refinement, evolution or even a new product launch as companies continue to farm data on what their customers want, and how their current iteration meets, or doesn't meet, their demands.

Hence most programmers will need to have some technical & conceptual understanding of how to approach data science; that is, not only how to implement certain tools, but how to decide when and why an analysis tool is used on a dataset, to help make difficult business decisions.

Version Control Collaborating and iterating changes over code is challenging and can roadblock progress if done inefficiently.

Working with software like git, a distributed version control software, helps to facilitate this collaboration through three distinct features; its long-term history of source files, its ability to branch to help delegation, and providing a method for users to trace how a file has changed over time, which is invaluable to debugging (Atlassian, 2019).

Hence a programmer needs to aptly use git to integrate properly into a team and collaborate on larger, important projects.

Defensive Programming Testing software & defensive coding means writing programs that perform consistently and correctly (Wiesen, 2022).

This ensures that the end user has a polished and clean experience with the program, even

when the data given to the function is invalid as a result of user error.

This is an important skill to possess as it can save hundreds of hours in testing, debugging and re-implementing of code, making large-scale projects significantly more efficient. On the flip side, a poor attention to defensive programming can create a colossal waste of time and resources.

Self-Learning & Adaptability A proficient software engineer aims to continue learning and become familiar with new technologies and tools to ensure they work with efficiency at all times (Corbin, 2018).

This can also stretch to making compromises on one's coding nuances to better integrate into a team, project or assignment, thus improving overall efficiency and smoothing the collaboration process.

Self learning takes many forms, as discussed earlier, and helps to keep a developer's skills moving with the evolution of the industry.

2. Level 2: Basic Technology

2.1. LAMP Tech Stack: Adam Jankelowitz

LAMP The LAMP tech stack, one of the most common software stacks for web applications is a collection of loose open-source components in which developers can build various types of web applications and programs from (Lawton, 2005). These tech stacks usually consist of the Linux Operating System, Apache HTTP server, MySQL database system and also the PHP programming language.

Linux Operating System The main purpose of an open source software is to make the source code available and readable. The Linux operating system gives one the ability to add to, edit or further develop the code. With the ability to do these enhancements, Linux ultimately ensures software security, as well as giving users the choice as to which version they want to use (Fox, n.d.). Linux's role in the lamp tech stack is to ultimately manage hardware, memory management, process management and software security for the entire tech stack.

Apache HTTP Server Apache is an open-source web server that plays a large role in the LAMP tech stack. Apache is responsible for accepting directory (HTTP) requests from Internet users and sending them their desired information in the form of files and Web pages ("Apache | Definition Facts", 2022). Apache constitutes of various features which can range from authentication schemes to supporting-side programming languages such as Python and Pearl. Although Apache is referred to as a web server, it is rather a software that runs on a HTTP server. As the web server, its primary role in the stack is to process requests and also to establish a connection between a server and the browsers of website visitors.

MySQL Database System MySQL database is the third layer of the LAMP stack. It is a database management system which stores application data ("What is LAMP Stack?", 2022). Being an open-source database system, the role of MySQL in the stack is to store all the information in a format, which can then be queried easily using the SQL language. Due to the robust nature of MySQL, it proves to be an extremely important part of the stack, being an efficient database system for complex and large websites, giving developers the ability to build dynamic database systems.

PHP Programming PHP (Hypertext Pre-processor) is a programming language which merges all the elements of the stack together which allows the websites and web applications to run smoothly and efficiently (home, support </form>, 2022). When the webpage is opened, the PHP commands are processed and sends the results to the user's browser. As HTML is a static processor, and also because the LAMP stack uses the Apache HTTP server, inserting PHP programming into the stack enables the web applications to be dynamic (Education, 2022). Once MySQL stores all the information, PHP uses the code to create the HTML which is then sent to the Apache web server to send to the browser.

2.2. MEAN Tech Stack: Anh Ho

MEAN MEAN stands for MongoDB, Express.js, AngularJS, and Node.js, and is a relatively new stack. MEAN is a JavaScript stack that is mostly used for cloud-ready apps.

MongoDB (NoSQL database) MongoDB is a NoSQL database that is open source and geared for cloud applications. Instead of a relational paradigm, it employs object-oriented structure.

MongoDB stores the application's data in the MEAN stack. Because of MongoDB's role in storing data, there is no need to translate the object as it moves from the application to the database and back because both the application and the database use JavaScript. The application may push and pull objects between the back end and the database.

Express.js backend web framework Express is a Node.js web application framework. It strikes a mix between ease of use and a robust feature set.

Express is the MEAN stack's backend, and it manages all interactions between the frontend and the database, ensuring a smooth data transmission to the end user. Because it's made to work with Node.js, it maintains the stack's consistent use of JavaScript.

Angular.js frontend framework The Angular JS open-source front-end web framework leverages JavaScript development and is used as the front-end for the MEAN stack or other programming languages and frameworks.

AngularJS is included in the MEAN stack to assist developers in creating the user-facing portion of the application. Because the application's backend, frontend, and database are all created with JavaScript, data flows smoothly across all areas of the application

Node.js server side JavaScript Node.js is an open source JavaScript framework that uses asynchronous events to process multiple connections simultaneously.

The MEAN stack's backbone is Node.js. AngularJS integrates to Node.js for data serving, and Express is purpose-built to run on top of Node.js. Node.js includes a built-in web server, making it simple to migrate the MongoDB database and application to the cloud.

2.3. ASP.NET Tech Stack: Finn Gladstone

ASP.NET The ASP.NET Core stack is a framework developed by Microsoft that is known for its lightweight structure and compatibility (runs on Windows and Linux) (FullScale, undated). It is used to build web apps, websites and APIs.

Common Language Runtime The .NET run-time environment is responsible for compiling & executing the code, and smoothing the development process (Microsoft Docs, undated)

This is helped through its ability to model components, compile and error-check across multiple languages, and use of metadata to manage classes and utilise memory.

It is also responsible for garbage collection and managed data.

Framework Class Library The class library is a variety of classes, both in concrete code or abstract interfaced implementations, that are drawn upon and modified by the developer to their specific needs (MS Docs). These interfaces help with;

- Representing base data types
- Data Structures
- Application security
- Managing GUI implementation on client and server side

Metadata and Assemblies The .NET framework is shaped to let compilers merge declarative information into modules and assemblies, thus facilitating operation between modules that use different languages (MS Docs).

This is achieved through Metadata, binary information that is loaded into memory and is used on execution to pass information about code, classes, etc, to allow multi-language implementations. This is particularly useful for larger projects at big companies.

The use of metadata means that the program runtime is inheritantly more customisable and more referenced to your code, making the program more reliable and easily diagnosed.

3. Level 3: Advanced Skills

Level 3 focuses on more advanced technical skills (L^AT_EX and Git) and analysis of linkages and relationships between the items in the company tech stack.

The following is a list of advanced Git and L^AT_EX skills/features. Each student should select one pair of items from each list and demonstrate actual use of each item (either through activity in Git, or through including items in this report). (Target = ~100 words per student for each feature).

- Git
 - Rebasing and Ignoring files
 - Forking and Special files
 - Resetting and Tags
 - Reverting and Automated merges
 - Hooks and Tags
- L^AT_EX
 - Cross-referencing and Custom commands
 - Footnotes/margin notes and creating new environments
 - Floating figures and editing style sheets
 - Graphics and advanced mathematical equations
 - Macros and hyperlinks

3.1. Advanced features: Adam Jankelowitz

Git Advanced Skill: Forking Forking in Git is an advanced skill which allows one to copy a repository without making any changes to the original project and will create a completely new version of the original (McKenzie, 2021). The ability to fork a project is extremely helpful in version control, ultimately ensuring that conflict is avoided. When a fork of the repository is created, only the person who performed the fork will know it exists and will have ultimate control over the forked project. For example, forking is a perfect way to propose changes to the original project. Instead of logging an issue of a bug in the original project, you can fork the repository, debug the project and then submit a pull request which will immediately update the main repository (Fork a Repo, n.d.).

Git: Special Files Within Git, there are multiple special files which are used to improve repository management and interactions between developers (Henderson, 2022). These files can be committed to the repository and each serve a different purpose.

The first special file is `.gitignore`. This file is used when there is a specific file that Git should ignore. These files are usually files created by the machine in which serve no purpose in the repository. Some examples are files generated at runtime, hidden file systems and also personal IDE config files (Atlassian, 2019). As these files serve no true purpose in the repository, the `.gitignore` file can be edited and specifically specified with files that should be ignored.

Another special file which can be added to a repository is `.gitattributes`. The `gitattributes` file is extremely useful as it allows one to state the files and attributes that should be used when performing actions in git (Git - Gitattributes Documentation, n.d.). This can include

git commit and git add. When these attributes are included in the special file, git will automatically save the file according to the attributes provided whenever a file is created or saved within that repository.

`.gitmailmap` is a special file that can prove to be extremely useful when using Git. Essentially, this file is to tell Git about user aliases. In Git, one may add multiple email addresses under one account, as well as usernames. Ultimately, the `.gitmailmap` file is used to map the usernames of members within the repository and the email addresses used to the user's actual name (jramya, n.d.). For example, the history in a repository contains commits by Adam and Fred whose names appear in several forms (seen below):

Adam Developer <adam@test.com>

Adam F. Developer <adam@test.com>

Fred Smith <fred@test.com>

Fred Smith <fred@home.com>

Fred S <fred@work.com>

The mailmap of this example is seen below where Adam would like to include his middle name and Fred would like only his last name:

Adam F. Developer <adam@test.com>

Fred Smith <fred@test.com>

Fred Smith <fred@home.com>

All three of these special files are extremely useful and there are many others that can serve multiple different purposes.

L^AT_EX Advanced Skill: Graphics Graphics and images are important elements to include in documents and reports to further develop and express an idea. As L^AT_EX cannot handle images by itself, the graphic package must be used, as well as multiple key commands.

As seen below, there are specific commands that one must use to include graphics in a L^AT_EX document. In this example, I will upload the University of Sydney logo. The first line of code is included at the beginning of the file to install the package. The second command seen below is also required to insert graphics. This command tells the document that the image is kept in a folder in the directory of the file path written. Finally, one must include the final command, which includes the name of the image to be uploaded and must be written where you would like the image to be.

```
\usepackage{graphicx}
\graphicspath{ {./filepath/} }
\includegraphics{logo}
```



L^AT_EX Advanced Skill: Mathematical Equations Due to the nature of L^AT_EX being focused on plain text as opposed to presentation, it is extremely useful when using mathematical equations. L^AT_EX has the ability to use advanced mathematical symbols, and can

perform equations efficiently and effectively. In order to do so, one needs to import two key packages;

```
\usepackage{amsmath} and \usepackage{amssymb}
```

The `amsmath` package provides features in which one can write and perform math equations, as well as improving the style of the output.

The `amssymb` package imports a large variety of mathematical symbols into the document which can be used by the user.

With these commands, one can perform advanced mathematical equations when required. For example, if one was required to include a matrix in a report, they would use the following command (Parker and Schwein, 2015):

```
\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
\]
```

This command will give the following matrix:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Another example is if someone had to perform or express multiple mathematical equations and wanted to label them. This is seen below and would be performed using the following commands (Parker and Schwein, 2015):

```
\begin{equation}
x^n + y^n \neq z^n
\label{fermat}
\end{equation}
\begin{align}
z^n &= x^n + y^n
\nonumber \\
&\neq (x+y)^n
\end{align}
```

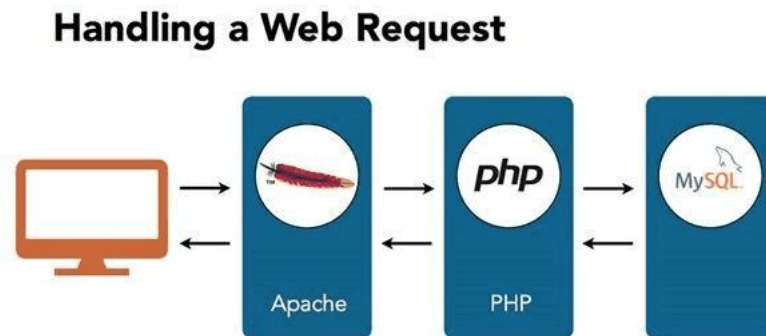
$$x^n + y^n \neq z^n \tag{1}$$

$$\begin{aligned} z^n &= x^n + y^n \\ &\neq (x+y)^n \end{aligned} \tag{2}$$

Ultimately, the mathematical packages that one can import into L^AT_EX are extremely useful and provide endless opportunity to express mathematical equations and symbols.

LAMP Stack Interactions As seen above in section 2.1, the LAMP tech stack is one of the most common software stacks for web applications. The successful nature of the LAMP stack relies on each part of the stack interacting with one another in which web applications and programs can be built successfully and efficiently.

The first interaction occurs between the Apache HTTP server and PHP programming. The Apache server is responsible for accepting HTTP request from internet users and then sending this information in the form of web pages and files. This interaction commences when the Apache web server receives the requests for web pages. If the request is for a PHP programmed file, Apache will send the request to PHP. PHP will then load the file and execute the code given. After processing and running the code, PHP will then send the processed data back to the Apache web server.



Another interaction which occurs in the LAMP tech stack is between the PHP programming language and MySQL. This interaction occurs since MySQL makes data retrieving automatic and also provides good support in PHP web application development. In a similar way to the interaction between PHP and Apache, when PHP receives the file, data may be contained within it. If data is referenced in the PHP file, PHP will communicate with MySQL to fetch and derive this data reference. The functions within PHP make communicating with MySQL simple and efficient as these functions are used to send these data queries to the database.

Within the LAMP tech stack, there is an over-arching interaction which occurs and this is by the Linux Operating System. Linux gives one the ability to add to, edit or further develop the code. Linux's role in the lamp tech stack is to ultimately manage hardware, memory management, process management and software security for the entire tech stack. The role of Linux in the stack is to interact with every part of the stack, involving both the systems hardware and software.

These key interactions are crucial to the LAMP tech stack and without these occurring, the stack would not be able to deliver its strong platform for developing and hosting web applications (LAMP: Which Stack Is Right for You?, 2019).

3.2. Advanced features: Finn Gladstone

Rebasing in Git is an alternative to merging and it helps users to implement changes from one branch into another. When rebase is called, the secondary branch is moved into the main branch, *but the main branch commit history is rewritten*, meaning a linear project history is maintained (git SCM, N.D.).

Having a linear project history helps to improve readability and tracing as users can follow a feature's development through a single branch without having to follow forks. This also means commands such as `git log`, `git bisect`, `gitk` are more applicable (Atlassian, N.D.).

There are tradeoffs to using rebasing, however. Namely, executing a rebase correctly is challenging, and it removes tracing for feature updates, meaning it is hard to see how the branch evolved separately to main.

Ignoring files in Git is used to disregard tracking for certain files in a repository. This is used to make sure machine generated files or build artifacts are not pushed to the repository and obstructing the source code (Atlassian, N.D.).

The types of files that users will typically choose to signal Git to ignore include:

- Log files
- Build files
- Dependency caches
- Build output directories
- Hidden system files
- Runtime files

I used this feature when editing this document on my local machine, and then updating our group repository.

When editing this document, my editor (TexStudio) produces approximately 6 files in the render process that I would prefer not to add to the git repo. Using the ignore feature allowed me to mitigate this issue.

Floating figures in \LaTeX allow us to change the positioning of an image or object in our document.

This allows us to achieve correct centering, position, any desired caption and add references to the object (Overleaf, N.D.). Latex uses the figure environment to achieve this with a series of specifiers to change the object's position.

This is a floating figure. It has been parsed with the 'h' argument in its location specifiers, meaning it will appear in roughly the same location in the export as in the source code (h meaning *here*).

Style sheets in \LaTeX determine the formatting rules for the given document, changing the default look or adding more methods through packages (Overleaf, N.D.). They are additional files loaded using `\documentclass` and `\usepackage` commands, and will help define the structure of the document (Robertson, 2005).

In this document, the style formatting runs from lines 1-22 and it incorporates packages such as:

- `blindtext` which allows the user to test new classes and packages
- `fontenc` which allows the user to select font encodings and;
- `fancyhdr` which provides tools to manipulate page headers and footers.

Interactions in ASP.NET Tech Stack:

- **CLR + Framework Library:** While the CLR is responsible for executing code and smoothing the development process, the Framework Library provides a broad body of classes and methods that developers integrate with their own projects in the runtime.

This integration is aided by the CLR's jack-of-all-trades coverage that manages exceptions, multiple languages, security, version control and models for component interaction.

- **Framework Library + Assemblies:** Assemblies act as standalone function executables, which are often built from one or more classes from the Framework Library. This integration is facilitated by the global assembly cache (GAC), which helps to integrate classes and also distribute assemblies across applications.

We can think of assemblies like mini-programs; they are executable, and can be run individually, but often act as a puzzle piece in a larger project. Assemblies are also highly flexible, and facilitate the integration of the framework library in a larger project through an object-oriented manner.

- **CLR + Assemblies / Metadata:** When a program is executed in ASP.NET, the CLR utilises the Metadata of the assemblies within the project scope to obtain information about how the assembly works, its dependencies and its permissions.

The integration of these two technologies means compilation is much more refined; metadata allows for self-describing files, meaning two modules can interact seamlessly at compilation. In addition, our assemblies can be written in a breadth of supported languages and the compilation process is no different, bringing unprecedented levels of flexibility.

3.3. Advanced features: Anh Ho

Explain your use of the advanced Git and \LaTeX features.

4. Level 4: Advanced Knowledge

Level 4 focuses on analysing your particular tech stack and considering alternatives. Each student should consider the tech stack they described for Level 2, and then discuss each of the following points:

- What are the strengths and limitations of this stack? (Target = ~ 200 words).
- What alternatives exist, and under what situations might these alternatives be a better choice? (Target = ~ 200 words).

4.1. Advanced Knowledge: Adam Jankelowitz

The LAMP tech stack, being the most common software stacks for web applications is a collection of open-source components in which can build various types of web applications. This tech stack consists of the following;

- Linux Operating System
- Apache HTTP server
- MySQL database system
- PHP programming language

The LAMP stack has many strengths but also has some limitations due to the nature of its different stack components.

Strengths

Open Source The first strength of the LAMP tech stack is that it is open-source. An open-source software means that it's source code is available and accessible to the public. As well as having access to view it, open source software gives anyone the ability to modify and distribute the source code. This is a strength of the LAMP since the source code can be modified and further developed, ultimately improving the stacks overall performance (PhoenixNAP KB, 2022).

Easily Customizable Another advantage of using the LAMP tech stack is its ability to be customized. Due to the flexibility of LAMP, users can replace components of the stack with another open-source software in order to suit the needs of the project or application development. For example, rather than using PHP programming, users can interchange this with Python or Perl which are other programming languages. This change may occur as Python is a general-purpose programming language and can be learnt relatively quickly.

Despite these strengths of the LAMP tech stack, there are multiple limitations and disadvantages when using this stack.

Limitations

Limit in Operating Systems One main limitation in the LAMP tech stack is that the stack does not support any other operating system other than Linux. Being the backbone of the stack, Linux is used due to its flexibility and customizable nature, therefore being efficient when running the other LAMP components of Apache, MySQL and PHP. However, there are multiple limitations to the Linux operating system. The first disadvantage of using Linux is the lack of expertise in the operating system. When being compared to other operating systems such as Windows and Mac, Linux is difficult to learn due to having to learn about the command-line interface. Furthermore, when seeking for advice in Linux, there are fewer experts as compared to Windows and Mac. Another limitation of Linux is the lack of Hardware drivers. Being one of the biggest issues when using Linux is that most hardware drivers are not available for the Linux operating system. This is because other systems such as Windows and Mac have more users and is more accessible.

Relational Property of MySQL Another limitation in the LAMP tech stack is due to the relational property of MySQL as it makes the entire LAMP stack more inefficient and inflexible when compared to other database management systems. Some disadvantages of the relational property is that the relational data model does not fit with every domain (Hauer, 2015). To expand on this point, when one is required to traverse a graph, the relational database will make this inefficient due to the amount of joins required. Also, when dealing with nested domain models that require tables and joins, the relational property will make MySQL difficult to work with in this situations (Hauer, 2015). Non-relational databases such as Mongo DB are able to store data in objects that can be deeply nested and can contain arbitrary fields which are two situations that MySQL cannot handle.

Other Limitations Some other limitations include that Apache can run into performance problems when it is under heavy workloads, as well as switching between Python and PHP on the server-side and using Javascript on the client-side can cause a disruption to the development workflow.

Alternatives There are many possible alternatives to the LAMP tech stack and each alternative would be used in different scenarios. One possible tech stack alternative is the MEAN stack which consists of the following components:

- **M**ongo DB
- **E**xpress JS
- **A**ngular JS
- **N**ode JS

When being compared to the LAMP tech stack, there are a number of differences between the two. MongoDB is the database system used in the MEAN tech stack. In a similar way to MySQL in LAMP, MongoDB is also open-source, however, it employs object-oriented (non-relational database) structure, opposing to the relational MySQL in LAMP. Just like MySQL, MongoDB stores the data in the stack but does not require to translate it as both

the application and database use JavaScript making it efficient to use.

Express.js and Angular.js are the backend and front-end frameworks respectively. As the back-end of the MEAN stack, Express.js manages the interactions between the database and the front-end, whilst ensuring an efficient transmission of data. On the other end of the stack, Angular.js assists the developers in creating the applications user-face. Another advantage of using MEAN is that the entire stack uses the programming language of JavaScript, allowing developers to reuse code, and can work on both front-end and back-end development simultaneously.

When deciding whether to use LAMP or MEAN for a project in which collaboration would take place, the MEAN tech stack is preferable. Due to the use of JavaScript in MEAN would make the project more time efficient and productive. As JavaScript is used for both the front and back end, this will allow developers to switch easily between the two, as well as creating a homogeneous and readable workflow.

Another case where the MEAN stack should be used over LAMP is when creating a start-up business. The demand for MEAN stack development has increased due to its capability to create fast and vigorous application (MEAN Stack for Startups: What Does It MEAN for Startups?, n.d.). As stated above, the MEAN stack helps to achieve an easy to use system as developers only require to code in JavaScript. Also, the stack comes with additional packages and frameworks which can further develop the application. Furthermore, when a new start-up is being formed, using cost-effective systems is extremely important. The MEAN stack is extremely cost-effective due to its use of JavaScript being the core language across all frameworks. The stack is also cost effective due to its open-source nature, meaning a licence fee is not required. The open-source technology will save development costs and will also result in an extremely high-functioning development.

4.2. Advanced Knowledge: Finn Gladstone

Strengths of ASP.NET :

- **Object-Oriented design:** The ASP.NET framework is very supportive of object-oriented programming, and this is particularly advantageous in ASP.NET's desired use case, where large companies tackle big projects with multiple teams working on different features (SoftwareDeveloperIndia, 2018). It also takes inspiration from MVC architecture which facilitates effective web design.
- **Language Independent:** Thanks to the CLR compiler, there are a variety of languages that can be utilised in the ASP stack, meaning programmers have free reign to choose the optimum language that suits their project, on grounds of efficiency, portability etc (Net-informations.com, N.D.).
- **Massive libraries:** The framework library holds an abundance of methods and objects which can be seamlessly implemented, allowing programmers to save time and take advantage of industry standard code across many different languages. In addition, the use of Metadata and Assemblies means accessing these libraries on compilation is much more straightforward.

Limitations of ASP.NET :

- **Constricting design principles** Given ASP.NET is built around Microsoft's Windows operating system, ASP.NET requires users to adopt some design procedures, such as MVC as mentioned above. For some use cases, this is a bridge too far, hence why some may avoid adopting ASP.NET.
- **Imminent redesign:** As of 2019, ASP.NET is undergoing a paradigm shift in its identity and function as Microsoft attempts to migrate .NET and .NET Core into a new framework. The new .NET Core 5.0 (Nov 2020) replaces the .NET Framework and modernises the ASP.NET structure (Kanan, 2021).

However, 5.0 eliminates key features of previous builds, of which many current tools and projects are maintained on today, leaving a rift in the userbase. Hence this identity crisis may turn programmers away from using ASP.NET for a new project.

Alternatives to ASP.NET : A popular alternative to the ASP.NET framework is the **Django Framework**, a python-based web development framework. Its features include:

- **Documentation:** Django has held a long-standing reputation for excellent documentation, which has improved since its move to open source in 2005 (Data Flair, N.D.). Extensive documentation helps to overcome the learning curve of adapting a new platform, which makes Django a competitive option.
- **SEO Optimised:** When publishing a website on the Django framework, it is maintained on the network via its URL as opposed to the server IP. For a variety of reasons, this means its contents are easier read by a search engine, thus increasing the chances of the web project being seen and gaining traction.
- **High Security:** Django is one of the most secure web platforms, as it was written from ground up with security and back-end loopholes in mind (Data Flair, N.D.). This means Django is an easier choice for platforms that are designed to hold and distribute sensitive data, or receive high traction.

We can see Django is relatively easy to learn, being based in Python and also benefiting from comprehensive documentation. It also benefits from SEO optimisation and a good reputation for security.

Hence, if learning C is a barrier and the project is on web design, Django could be a more appropriate choice than ASP.NET.

4.3. Advanced Knowledge: add student 3 name here

Your text goes here

5. References

- Ikechukwu, L. (2020). *The Self-Taught Developer's Guide to Learning How to Code*. (See <https://www.freecodecamp.org/news/the-self-taught-developers-guide-to-coding/>)
- Atlassian. (2019). .gitignore file - ignoring files in Git | Atlassian Git Tutorial. Atlassian. <https://www.atlassian.com/git/tutorials/saving-changes/gitignore>
- Colombia Universtiy (2020). *Are Coding Bootcamps Worth It? [What the Numbers Say]* - *Columbia Engineering Boot Camps* (See <https://bootcamp.cvn.columbia.edu/blog/are-coding-bootcamps-worth-it/>)
- Shipley, B. (2019). *A Realistic Perspective Of The Pros And Cons Of Coding Bootcamps* (See <https://medium.com/swlh/a-realistic-perspective-of-the-pros-and-cons-of-coding-bootcamps-527a1e4b8fb2>)
- Eira, A. (2018). What Is Continuous Learning: Benefits of Adopting It in Your Workplace - Financesonline.com. Retrieved 27 March 2022, from <https://financesonline.com/what-is-continuous-learning/>
- Erstad, W. (2017). Computer Programmer Skills: The Perfect Balance of Hard & Soft Skills Employers Are Seeking | Rasmussen University. Retrieved 27 March 2022, from <https://www.rasmussen.edu/degrees/technology/blog/5-soft-skills-programmers-need/>
- 3 Reasons Why Data Scientists Should Learn Statistics Well. (2022). Retrieved 3 April 2022, from <https://towardsdatascience.com/3-reasons-why-data-scientists-should-learn-statistics-well-90e80ae6c68f>
- 9 Must-have skills you need to become a Data Scientist, updated - KDnuggets. (2022). Retrieved 3 April 2022, from <https://www.kdnuggets.com/2018/05/simplilearn-9-must-have-skills-data-scientist.html>
- How Much Do Data Scientists Need to Know about Statistics? — School of Data Science. (2022). Retrieved 3 April 2022, from <https://datascience.virginia.edu/news/how-much-do-data-scientists-need-know-about-statistics>
- Shin, T. (2022). 11 Most Practical Data Science Skills for 2022 - KDnuggets. Retrieved 3 April 2022, from <https://www.kdnuggets.com/2021/10/11-most-practical-data-science-skills-2022.html>
- Top 8 Tech Stacks: Choosing the Right Tech Stack. (2022). Retrieved 3 April 2022, from <https://fullscale.io/blog/top-5-tech-stacks/>
- Unwin, A. (2020). Why is Data Visualization Important? What is Important in Data Visualization?. 2.1. doi: 10.1162/99608f92.8ae4d525
- Weihs, C., & Ickstadt, K. (2018). Data Science: the impact of statistics. *International Journal Of Data Science And Analytics*, 6(3), 189-194. doi: 10.1007/s41060-018-0102-5

Why Data Visualization Is Important - Analytik. (2022). Retrieved 3 April 2022, from <https://analytik.co/importance-of-data-visualization/>

Sedlacek, L. (2016). *Math Education: The Roots of Computer Science*. (See <https://www.edutopia.org/blog/math-education-roots-computer-science-lincoln-sedlacek/>)

Michigan Technological University. (2018). *Why Computer Scientists Can Benefit from Studying Applied Statistics*. (See <https://onlinedegrees.mtu.edu/computer-scientists-can-benefit-studying-applied-statistics/>)

Brush, K. (2020). *What is data visualization and why is it important?*. (See <https://www.techtarget.com/searchbusinessanalytics/definition/data-visualization/>)

Vadapalli, P. (2020). *Top 10 Skills For Every Computer Science Professional in 2022*. (See <https://www.upgrad.com/blog/skills-for-every-computer-science-professional/>)
Lawton, G. (2005). LAMP lights enterprise development efforts. *Computer*, 38(9), 18-20. doi: 10.1109/mc.2005.304

1.3. What Is Computer Science? — Problem Solving with Algorithms and Data Structures. Retrieved 3 April 2022, from <https://runestone.academy/ns/books/published/pythonds/Introduction/WhatIsComputerScience>

Common Language Runtime (CLR) overview - .NET. Retrieved 3 April 2022, from <https://docs.microsoft.com/en-us/dotnet/standard/clr>

Corbin, A. (2018). What Makes For Great Software Engineers? Technical Skills Are Just The Start. Retrieved 3 April 2022, from <https://www.profiq.com/what-makes-for-great-software-engineers-technical-skills-are-just-the-start/>

Gupta, S. (2020). What Are Data Structures and Algorithms?. Retrieved 3 April 2022, from <https://www.springboard.com/blog/software-engineering/data-structures-and-algorithms/>

LAMP Stack - What Is It, Advantages Alternatives | phoenixNAP KB. (2022, January 6). Knowledge Base by PhoenixNAP. <https://phoenixnap.com/kb/what-is-a-lamp-stack>

Johnson, D. (2022). What is Data Analysis? Research | Types | Methods | Techniques. Retrieved 3 April 2022, from <https://www.guru99.com/what-is-data-analysis>

Problem Solving. Retrieved 3 April 2022, from <https://www.cs.utah.edu/~germain/PPS/Topics/problemsolving.html>

Rosello, N. (2019). Effective Communication Skills In Software Engineering | Santex. Retrieved 3 April 2022, from <https://santexgroup.com/blog/communication-skills-software-engineering/>

S. Gillis, A., Lewis, S. (2021). What is Object-Oriented Programming (OOP)?. Retrieved 3 April 2022, from <https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP>

Top 8 Tech Stacks: Choosing the Right Tech Stack. (2020). Retrieved 3 April 2022, from <https://fullscale.io/blog/top-5-tech-stacks/>

Vlasiuk, A. (2021). Soft Skills for Software Engineers —Part 1: Communications. Retrieved 3 April 2022, from [:/medium.com/geekculture/soft-skills-for-software-engineers-part-1-communications](https://medium.com/geekculture/soft-skills-for-software-engineers-part-1-communications)

What is version control | Atlassian Git Tutorial. Retrieved 3 April 2022, from <https://www.atlassian.com/git/is-version-control>

Hauer, P. (2015, May 14). Why Relational Databases are not the Cure-All. Strength and Weaknesses. Philipp Hauer's Blog. <https://phauer.com/2015/relational-databases-strength-weaknesses-mongodb/>

Wiesen. (2022). What Is Defensive Programming? (with pictures). Retrieved 3 April 2022, from <https://www.easytechjunkie.com/what-is-defensive-programming.htm>

Unwin, A. (2020). Why is Data Visualization Important? What is Important in Data Visualization?. 2.1. doi: 10.1162/99608f92.8ae4d525

Weihs, C., Ickstadt, K. (2018). Data Science: the impact of statistics. International Journal Of Data Science And Analytics, 6(3), 189-194. doi: 10.1007/s41060-018-0102-5

Why Data Visualization Is Important - Analytiks. (2022). Retrieved 3 April 2022, from <https://analytiks.co/importance-of-data-visualization/>

home, P., support, c., </form>. (2022). LAMP Stack - What Is It, Advantages Alternatives | phoenixNAP KB. Retrieved 3 April 2022, from [https://phoenixnap.com/kb/what-is-a-lamp-stack: :text=PHP](https://phoenixnap.com/kb/what-is-a-lamp-stack#:~:text=PHP)

Education, I. (2022). LAMP stack explained: Master the basics and get started quickly. Retrieved 3 April 2022, from <https://www.ibm.com/cloud/learn/lamp-stack-explained>

Fork a repo. (n.d.). GitHub Docs. <https://docs.github.com/en/get-started/quickstart/fork-a-repo>

McKenzie, C. (2021, July 28). Git fork vs. clone: What's the difference? TheServerSide.com. <https://www.theserverside.com/answer/Git-fork-vs-clone-Whats-the-difference>

Git - gitattributes Documentation. (n.d.). Git-Scm.com. Retrieved April 27, 2022, from <http://git-scm.com/docs/gitattributes>

jranya. (n.d.). What the mailmap file is. Library.netapp.com. Retrieved April 27, 2022, from <https://library.netapp.com/ecmdocs/ECMP1222473/html/GUID-9B5C60EF-F21D-4C17-9A39-1540D3590F98.html>

Parker, D., Schwein, D. (2015). Advanced L A T E X. <https://www.brown.edu/academics/science-center/sites/brown.edu/academics/science-center/files/uploads/advancedLaTeX0.pdf>

LAMP vs. MEAN : Which stack is right for you? (2019, May 2). Bitbucket. <https://bitbucket.org/blog/lamp-vs-mean-which-stack-is-right-for-you> : : text = LAMP