WEBD-3012
Business System Build and Testing

Term 3

Full Stack Web Development Program

Maninder Kaur

Instructor

Course Information

Course Code and Title:	WEBD-3012 Business System Build and Testing
Course Section:	273795
Department/Program:	Full Stack Web Development
Total Hours:	90 Hours
Credit Hours:	6
Instructor’s name:	Maninder Kaur
Email:	maninderkaur@rrc.ca
Office hours:	Available by appointment /online/ email/MS Team

Textbook(s):

- Kim, G. (2019).The Unicorn Project: A Novel about Developers, Digital Disruption, and Thriving in the Age of Data (1st ed.) IT Revolution Press.

LEARNING OUTCOMES AND ELEMENTS OF PERFORMANCE

By the end of this course of study, you should be able to...

1. Install and set up local environments with necessary software tools for course participation.
2. Understand and apply DevOps principles, including deployment strategies and software testing approaches.
3. Create a portfolio website with a test and deployment pipeline.
4. Demonstrate knowledge of DevOps history, pain points in code deployment, and DevOps cultural practices.
5. Implement automation tests and integrate testing frameworks into development processes.
6. Utilize tools like Docker and Jenkins for continuous integration and delivery.
7. Comprehend the role of Site Reliability Engineers and their responsibilities in ensuring site reliability in a production environment.
8. Analyze and apply various testing strategies in different project scenarios.
9. Evaluate and implement deployment strategies for getting code to production servers efficiently.
10. Calculate and work with TSLAs and SLOs, and understand their significance in DevOps and Site Reliability Engineering.

Date	Module/Unit/Week or Important Event	Topic and Learning Outcome(s)	Assessment and Evaluation	Weig ht
Week 1	Module 1: Introduction and Software Tools Set-Up	Installing a package manager, code versioning system, creating a development environment, and installing DevOps tools.	Unicorn Project Assignment 1	5%
Week 2	Module 2: Course Introduction and Assignment Descriptions	Unicorn Assignments Coding Assignment	Unicorn Project Assignment 2	5%
Week 3 & 4	Module 3: Introduction to DevOps	Pain Points of Deploying Code, What is DevOps, DevOps Process, Culture, and Practices.	Unicorn Project Assignment 3 Coding Assignment 11	5% 10%
Week 5	Module 4: Testing Methodologies	Types of tests and their applications, testing strategies for different scenarios.	Unicorn Project Assignment 4	5%
Week 6 & 7	Module 5: Introduction to Test Frameworks and Automation	Integration of test framework, UI and behavior testing, bug prevention.	Unicorn Project Assignment 5 Coding Assignment 12	5% 10%
Week 8	Module 6: Test Automation	Unit, Integration, and End-to-End Tests.	Unicorn Project Assignment 6	5%
Week 9 & 10	Module 7: Continuous Integration	Continuous Delivery and Integration, Automation in DevOps.	Unicorn Project Assignment 7 Coding Assignment 13	5% 10%
Week 11	Module 8: DevOps in Practice	Docker for CI/CD, Jenkins for continuous delivery, implementing DevOps with Docker.	Unicorn Project Assignment 8	5%
Week 12 & 13	Module 9: DevOps in Production	Deployment Strategies, Production Testing.	Unicorn Project Assignment 9 Coding Assignment 14	5% 20%
Week 14 & 15	Module 10: DevOps in Production: Site Reliability Engineering	Site Reliability Engineering, TSLAs and SLOs calculation, DevOps and SRE integration.	Unicorn Project Assignment 10	5%

Assessment	Weight	Module
Unicorn Project Assignment 1	5%	Module 1
Unicorn Project Assignment 2	5%	Module 2
Unicorn Project Assignment 3 Coding Assignment 11	5% 10%	Module 3
Unicorn Project Assignment 4	5%	Module 4
Unicorn Project Assignment 5 Coding Assignment 12	5% 10%	Module 5
Unicorn Project Assignment 6	5%	Module 6
Unicorn Project Assignment 7 Coding Assignment 13	5% 10%	Module 7
Unicorn Project Assignment 8	5%	Module 8
Unicorn Project Assignment 9 Coding Assignment 14	5% 20%	Module 9
Unicorn Project Assignment 10	5%	Module 10
Total	100%	

ASSESSMENT (Total 14)	WEIGHT
Unicorn Project Assignment (Total 10: 5%each)	**50**
Coding Assignment (Total 4)	**50**
Total:	**100%**

Note: All coding assignments will be evaluated in-person Only

Course AI guidelines

- The use of GenAI for academic work must follow the program and course expectations, as identified in this section and must adhere to Policy A17 – Academic Integrity.
- All tasks in this course must be completed independently, without the use of tools or assistance. Specifically, the use of Generative AI, Large Language Models (LLMs), or any AI-based platforms is not permitted. This ensures that students develop and demonstrate their own skills and understanding. No acknowledgment of AI use is required, as it should not be part of the work.
- Depending on the level of breach, consequences may include an alternate or additional assignment, a grade of zero, failure of the course, or academic suspension

Book overview:
Kim, G. (2019). *The unicorn project: a novel about developers, digital disruption, and thriving in the age of data*. IT Revolution.

Novel explores the challenges developers face in a traditional IT environment and how modern practices like DevOps and agile can transform a company

Key points of book:

- Follow-up to *The Phoenix Project*
- Tells the story of a struggling organization, **Parts Unlimited**, from the perspective of **Maxine**, a senior developer

Why Should Students Read This Book?

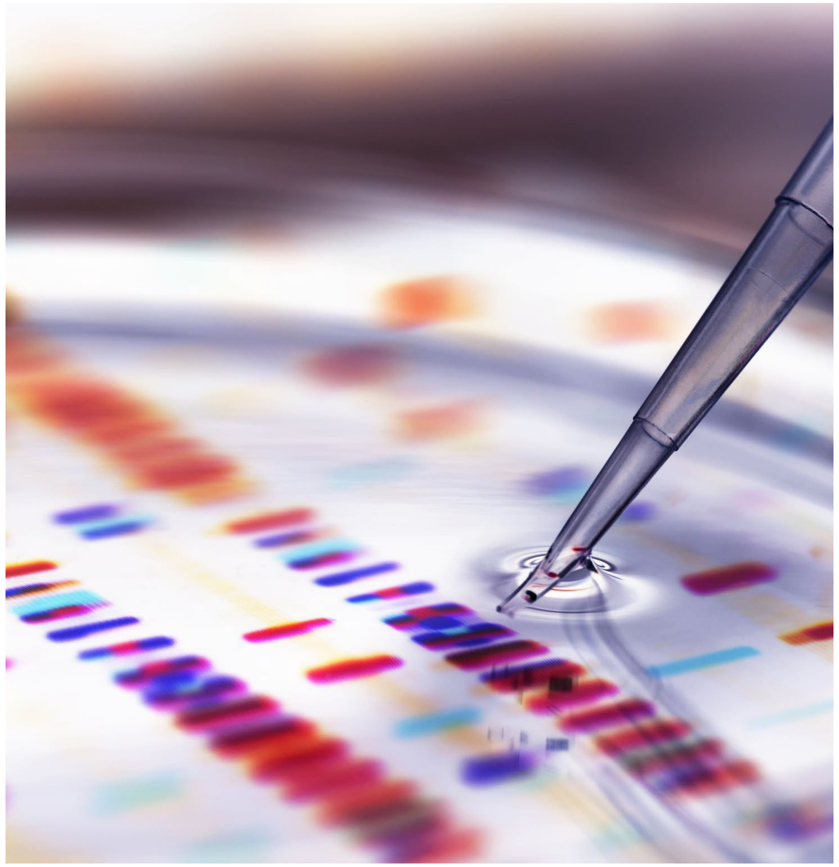
- **Understanding Real-World Challenges**
- **Developing Problem-Solving Skills**
- **Learning the Five Ideals:**
    - Locality and Simplicity
    - Focus
    - Flow
    - Joy
    - Improvement of Daily Work
    - Psychological Safety, and Customer Focus

What Can You Learn from This Book?

- Collaboration Matters
- Adaptability is Key
- Empathy for Developers
- Problem-Solving Frameworks

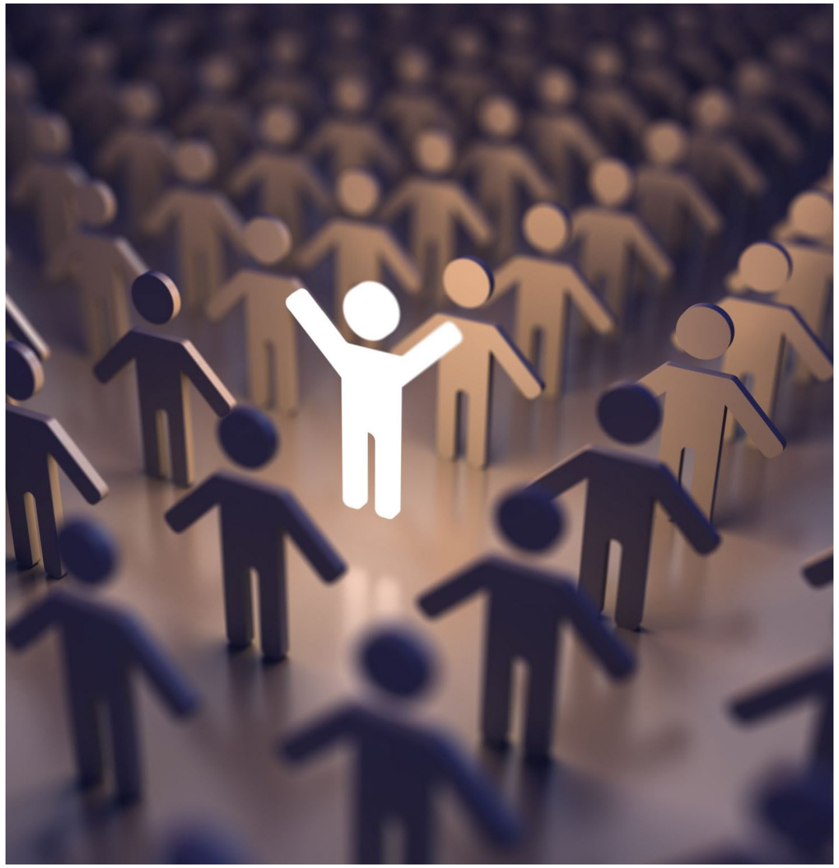
Assignment Connection

- Analyzing Perspectives
- Reflecting on Tools and Processes:



Key Takeaways for the Assignment (Whole Book Perspective)

- Organizational Health
- Importance of Leadership
- Empowering Employees
- Continuous Improvement



Module 1

- Install and prepare the local environment with the software required to successfully participate in the course.
- Install/run the following software tools:
    - Install git
    - Install nvm
    - Install code editor (vscode)
    - Install docker
    - Install typescript
- Read Chapter 1 and 2 of Textbook: Kim, G. (2019). *The unicorn project: A novel about developers, digital disruption, and thriving in the age of data* (1st ed.): IT Revolution Press.
- Assignment 1:Unicorn Project Assignment 1 (Based on chapter 1 and 2 of Textbook)

Important :

- Install nvm-window(instead of WSL to use nvm desktop)
- If you are not able to install latest version of Create React app due to compatibility issue, downgrade the version to 18

Unicorn Assignment 1

Assignment 1 is a reflection on reading materials assigned for the specific chapter, particularly book chapters (i.e., Unicorn Project Ch 1-2). It is worth 5% of the final grade

- Please read the Unicorn Project, **Chapter 1-2** and answer the following questions:

Question 1: What would the c suite position say are the top three issues the company is facing. What would middle management say is the main issue facing the company? What would Maxine say is the main issue facing the company?

Question 2:Compare and contrast Maxine's reaction to having to track activities in a timecard vs using her developer journal? What are the contributing factors to the difference in her reaction.

Version control

- Version control is a tool that tracks changes to source code over time.
- It's also known as source code management.
- Types of version control
    - Centralized Version Control: Example: Subversion (SVN)
    - Distributed Version control: Git
- Some benefits of version control include:
    - **Collaboration**: Developers can work together without fear of code conflicts.
    - **Quick access**: Users have quick access to the latest version of a file.
    - **Reduced duplication**: Users are less likely to overwrite each other's work or make changes to files that can't be merged.

Git

- Git is a **distributed version control system** created by Linus Torvalds in 2005.
- It is designed to handle everything from small to very large projects with speed and efficiency.
- **Uses and Purpose**:
  - Git is used for tracking changes in source code during software development.
  - It allows multiple developers to work on the same project simultaneously without overwriting each other's changes.
- **Advantages**:
  - **Distributed System**
  - **Branching and Merging**
  - **Performance**.
- **Disadvantages**:
  - **Complexity**:
  - **Storage**:
- **Applications**:
  - Git is widely used in software development for version control, collaboration, and maintaining the history of changes.

Git

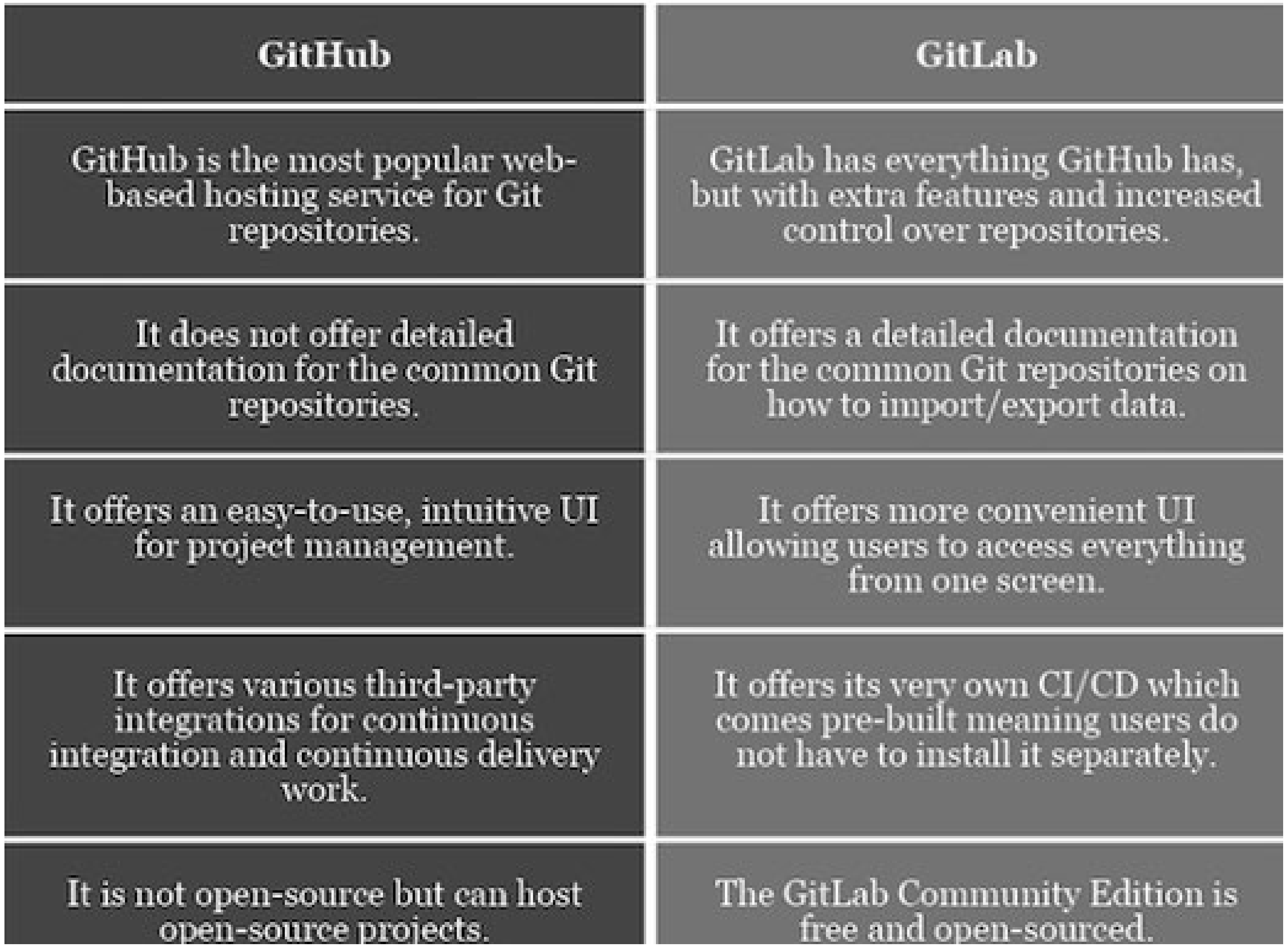
Devops tool used for source code management

Version control system

Allows developers to work independently on different parts of a project, track the history of changes, and merge their work together efficiently

GitHub/GitLab

- GitHub and GitLab are both web-based platforms that help developers host, manage, and collaborate on code:
- **GitHub:** A popular platform for open-source projects and community development. It's known for its code hosting and collaboration tools, including:
  - Repository management and hosting
  - Code review and management
  - Issue tracker
  - Project management tools
- **GitLab :** An all-in-one DevOps platform that includes Git repository hosting, project management, and CI/CD pipelines. It's known for its comprehensive set of DevOps tools and security features.
- [Learn shell: Git · GitHub]()



GitHub	GitLab
GitHub is the most popular web-based hosting service for Git repositories.	GitLab has everything GitHub has, but with extra features and increased control over repositories.
It does not offer detailed documentation for the common Git repositories.	It offers a detailed documentation for the common Git repositories on how to import/export data.
It offers an easy-to-use, intuitive UI for project management.	It offers more convenient UI allowing users to access everything from one screen.
It offers various third-party integrations for continuous integration and continuous delivery work.	It offers its very own CI/CD which comes pre-built meaning users do not have to install it separately.
It is not open-source but can host open-source projects.	The GitLab Community Edition is free and open-sourced.

Some Git Resources

- Git Guide :- Official git Learning Guides from the folks at GitHub.
- Git CheetSheet -- The most popular git commands in one page.
- Pro Git book - Free ebook provides a deep dive into everything git.

Node.js

Javascript runtime environment

Lets developers create servers, web apps, command line tools and scripts

NVM (Node Version Manager

NVM is a version manager for Node.js, designed to manage multiple active versions of Node.js on a single machine.

NVM allows you to quickly install and use different versions of node via the **command line.**

Node.js is a JavaScript runtime environment that allows developers to run JavaScript code on the server-side, essentially enabling them to build backend applications using JavaScript;

Node.js — Introduction to Node.js

- **Uses and Purpose**:

  NVM allows developers to easily switch between different versions of Node.js,

- **Advantages**:
  - **Flexibility**
  - **Isolation**
- **Disadvantages**:
  - **Overhead**
  - **Compatibility**
- **Applications**: NVM is used by developers who need to test their applications across different Node.js versions or maintain projects that depend on different versions.
- GitHub - nvm-sh/nvm: Node Version Manager - POSIX-compliant bash script to manage multiple active node.js versions

nvm

- Node Version Manager
- Command-line tool
- Allows installation of Node.js

NPM

- NPM (Node Package Manager) a free, open-source registry and tool for JavaScript software packages
- Tool used to install, manage, and share packages (libraries) for Node.js projects,
- Open-source developers from every continent use npm to share and borrow packages, and many organizations use npm to manage private development as well
- NPM manages project dependencies.

npm

- Package manager for javascript
- Allows installation of third-party libraries

NVM versus NPM

•NVM (Node Version Manager) is a tool used to manage different versions of Node.js on your system, allowing you to switch between them easily, while NPM (Node Package Manager) is a package manager that lets you **install and manage dependencies** for your Node.js projects;
•NVM manages which version of Node.js you are using, while NPM manages the packages within your Node.js project.

•Key Difference:

- **Function:** NVM is for managing Node.js versions, while NPM is for managing packages within a Node.js project.
- **Use case:** If you need to test your application with different Node.js versions, you will use NVM to switch between them. If you need to install a library for your project, you will use NPM.
- **Relationship :**When you install Node.js, NPM is included as part of the package, but NVM is a separate tool that needs to be installed separately.

GitHub and NVM/NVP

- GitHub is the online platform where the source code for "nvm" (Node Version Manager) is hosted, allowing developers to access and download the latest version of the nvm tool to manage different Node.js versions on their systems.
- GitHub acts as the repository for the nvm project, making it accessible to the public for download and contribution
- Key points about the relationship:
  - **NVM code on GitHub:b**The code for the nvm tool is publicly available on GitHub under the repository "nvm-sh/nvm".
  - **Installation via Git:** When installing nvm, users typically use a Git command to clone the repository from GitHub, effectively downloading the latest version of the nvm code.
  - **Version control:** GitHub allows for version control of the nvm code, enabling developers to track changes and updates to the tool

Visual Studio Code (VSCode)

- Visual Studio Code is a free, open-source code editor developed by Microsoft.
- It is available for Windows, macOS, and Linux.
- **Uses and Purpose**:
  - VSCode is used for writing and debugging code.
  - It supports a wide range of programming languages and comes with built-in Git support.
- **Advantages**:
  - **Extensibility**
  - **Integrated Terminal**
- **Disadvantages**:
  - **Performance**
  - **Learning Curve**

- **Applications**:
  - VSCode is used by developers for coding, debugging, and version control.
  - It is popular for web development, data science, and many other programming tasks.

React

- **React** (also known as React.js or ReactJS) is a free and open-source JavaScript library developed by Facebook (now Meta) for building user interfaces, particularly for single-page applications.
- It allows developers to create **reusable UI components**, which can manage their own state and be composed to build complex user interfaces.
- **Key Features**
  - **Component-Based**
  - **JSX**
  - **Virtual DOM**.
  - **Unidirectional Data Flow**

- **Uses and Purpose**
    - React is primarily used for building user interfaces in web applications. It can also be used for mobile app development with React Native and for desktop applications with Electron.
- **Advantages**
    - **Reusable Components**: Components can be reused across different parts of an application, reducing code duplication and improving maintainability.
    - **Performance**: The virtual DOM improves performance by minimizing direct manipulation of the real DOM.
    - **Strong Community and Ecosystem**: React has a large community and a rich ecosystem of libraries and tools.
- **Disadvantages**
    - **Learning Curve**: React introduces concepts like JSX and component-based architecture, which can be challenging for beginners.
- **Applications**
    - **Web Development**: React is widely used for building dynamic and interactive web applications.
    - **Mobile Development**: React Native allows developers to build mobile apps using React principles.
    - **Desktop Applications**: Tools like Electron enable the creation of cross-platform desktop applications using React.

React

Javascript library for building web, mobile, and desktop applications

Enables developers to create reusable UI components and manage application state efficiently

Create React App

- **Create React App** is a tool that helps you set up a new React project with a modern build configuration.
- It simplifies the process of creating a new React application by providing a pre-configured setup that includes all the necessary tools and configurations.
- **What It Does**
  - When you use Create React App to generate a new React project, it sets up a development environment with the following features:
    - **Modern JavaScript**: Uses the latest JavaScript features.
    - **Babel**: Transpiles modern JavaScript to ensure compatibility with older browsers.
    - **Webpack**: Bundles your JavaScript files and assets.
    - **ESLint**: Provides linting to help you write clean and consistent code.
    - **Jest**: Sets up a testing environment.

- **Advantages**
    - **Zero Configuration:** No need to manually configure tools like Webpack or Babel.
    - **Quick Setup**: Get started with a new React project in minutes.
    - **Best Practices**: Encourages best practices and a standardized project structure.
    - **Community Support**: Widely used and supported by the React community.
- **Disadvantages**
    - **Limited Customization**: The default configuration may not suit all projects, and customizing it can be complex.
    - **Overhead**: Includes many tools and dependencies, which might be unnecessary for very simple projects.
- **Applications**
  - Create React App is ideal for:
    - **New React Developers**: Provides an easy way to get started with React.
    - **Prototyping**: Quickly create prototypes and proof-of-concept applications.
    - **Small to Medium Projects**: Suitable for projects that do not require extensive customization of the build process.

Docker

- Docker is an open-source platform that allows developers to **build, test, and deploy applications quickly**

- Docker is a platform for developing, shipping, and running applications **in containers**.
- **Containers:** Containers are like virtualized operating systems that remove the need to directly manage server hardware
- Docker packages software into standardized units called containers that contain everything the software needs to run.

- Containers allow developers to package an application with all its dependencies into a standardized unit for software development.

- **Uses and Purpose**:
  - **Separation of applications and infrastructure:** Docker allows you to separate your applications from your infrastructure, so you can deliver software quickly.
  - **Portability:** Docker enables portability for when these packaged containers are moved to different servers or environments
  - **Efficiency:** Docker gives software developers a faster and more efficient way to build and test containerized portions of an overall software application

- **Advantages**:
  - Portability
  - Efficiency
  - Isolation
- **Disadvantages**:
  - Security
  - Complexity

- **Applications**:
  - Docker is used in DevOps for continuous integration and deployment, microservices architecture, and ensuring consistent environments across development, testing, and production.

Key features

- **Containerized development environments**: Docker provides a consistent environment for software to run across multiple computing environments.
- **Continuous integration and deployment**: Docker automates the deployment of applications inside containers.
- **Microservices architecture**: Docker supports microservices architecture.
- **Automated testing environments**: Docker is used for automated testing environments.
- **Infrastructure as code**: Docker is used for infrastructure as code.
- **Scientific research**: Docker enhances reproducibility in scientific research.

Docker

Platform for developing, shipping, and running applications

Enables separation between applications and infrastructure so software can be delivered quickly

Feature	Docker (Containerization)	Virtual Machines (VMs)
Definition	Lightweight, OS-level virtualization that isolates applications.	Full virtualization that emulates an entire operating system.
Architecture	Shares the host OS kernel with containers.	Includes a hypervisor to manage multiple guest OSs.
Resource Usage	Low; uses only the resources needed for the application.	High; requires resources for the entire OS and hypervisor.
Startup Time	Fast; typically seconds.	Slower; can take minutes.
Isolation	Process-level isolation using namespaces and control groups.	Complete isolation with a dedicated OS for each VM.
Portability	Highly portable; runs consistently across different environments.	Less portable; depends on the hypervisor and hardware.
Use Case Example	Running a single microservice (e.g., a web server in a container).	Running a full-stack application with multiple OS dependencies.
Scalability	High; designed for dynamic scaling.	Moderate; more resource-intensive to scale.
Performance	Near-native performance as it uses the host OS kernel.	Slightly slower due to the overhead of the hypervisor.
Storage Size	Smaller; only the application and its dependencies are stored.	Larger; includes the OS image and applications.
Security	Limited by the shared kernel, but can be enhanced with tools.	Stronger isolation due to separate OS environments.
Management Tools	Docker CLI, Docker Compose, Kubernetes, etc.	Hyper-V, VMware, VirtualBox, e

TypeScript

- TypeScript is a superset of JavaScript developed by Microsoft.
- It adds static types to JavaScript, which helps in writing more robust and maintainable code.
- **Uses and Purpose**:
  - TypeScript is used to develop large-scale applications with a clear structure and fewer bugs.
  - It compiles down to plain JavaScript, which can run in any browser or JavaScript engine.

- **Advantages**:
  - **Static Typing**
  - **Improved Tooling**
  - **Maintainability**

- **Disadvantages**:
  - **Compilation Step**
  - **Learning Curve**

- **Applications**
  - TypeScript is used in web development, especially for large-scale applications.
  - It is popular in frameworks like Angular and libraries like React.

TypeScript Installation

- TypeScript can be installed through three installation routes depending on how you intend to use it:
  - an npm module,
  - a NuGet package
  - a Visual Studio Extension.

- If you are using Node.js, you want the npm version.
- If you are using MSBuild in your project, you want the NuGet package or Visual Studio extension.

TypeScript

- Strongly typed programming language that builds on JavaScript
- Adds additional syntax (catch errors early)
- Converts to JavaScript

Tool	Purpose
Git	Tracks changes in code
GitHub	Stores code online for sharing and collaboration
NVM	Manages multiple versions of Node.js
Node.js	Runs JavaScript outside the browser
npm	Installs packages for JavaScript projects
Docker	Packages apps to run anywhere
TypeScript	JavaScript with types (safer and scalable)
VS Code	Code editor
Create React App	Tool to quickly start a React project

Activity: Bonus Marks: 2 marks: Week 1 and week 2

- Group activity: Total 6 groups (3 groups 6 members and 3 groups 5 members)
- **PPTs/ hands on activity**
    - What the tool/technology is
    - Why it is used (benefits, use cases)
    - How to install and configure it
    - Live Demo or Example

  Topic:
    - **Git & GitHub**
    - **NVM (Node Version Manager)**
        - Install Node.js & npm
    - **React**
        - Create React App
    - **TypeScript**
    - **Docker**
    - **Storybook**

Example: Git ppt :Demo/example include

- Launching bash
- Making folder
- Configuring git
- Initializing Repository
- Main branch/Master Branch
- Creating branches
- Checking Repo status
- Open window explore
- Creating Readme.md file
- Staging file before we commit
- Committing Staged files
- Good commit and bad commit
- Git commit and Git log

Thank you