Assignment Code: FSD-AG-004

Version Control & Basics of JavaScript | **Assignment**

Instructions: Carefully read each question. Use Google Docs, Microsoft Word, or a similar tool to create a document where you type out each question along with its answer. Save the document as a PDF, and then upload it to the LMS. Please do not zip or archive the files before uploading them. Each question carries 20 marks.

Total Marks: 180

Question 1 : Explain the purpose of version control systems like Git. Why are they essential in modern software development?

Answer:

Version control systems like Git are tools that help developers track changes, collaborate efficiently, and manage code history. Every update to the code is recorded, allowing teams to see what changed, who made the change, and when it happened.

Git enables multiple developers to work simultaneously on different features using branches, which can later be merged safely. It also allows teams to revert to earlier versions if something goes wrong, ensuring code stability.
In modern software development, Git is essential for team collaboration, safe experimentation, version tracking, and continuous integration, making development faster, more organized, and less error-prone.

Question 2: List the typical steps you would follow when working with Git to make changes to a project and share them on GitHub.

Answer:

The typical steps you would follow when working with Git to make changes to a project and share them on GitHub:-

1. Clone the repo – git clone <url> to copy it locally.
2. Create a branch – git checkout -b <branch-name> for your changes.
3. Make and save changes to the code.
4. Stage files – git add . to prepare them.
5. Commit – git commit -m "message" to save locally.

6. Push to GitHub – git push origin <branch-name>.
7. Open a Pull Request on GitHub for review and merging.
8. Pull latest changes – git pull origin main to stay updated.

Simple flow: clone → branch → change → add → commit → push → PR → pull.

Question 3: What is the purpose of a `.gitignore` file in a Git repository? Provide an example of how to ignore all `.log` files.

Answer:

The purpose of a .gitignore file is to tell Git which files or directories to ignore so they are not tracked or committed to the repository. This is useful for files like build outputs, logs, or sensitive data that shouldn't be version-controlled.

Example: To ignore all .log files, you would add this line to your .gitignore file:

```
*.log
```

//This ensures that any file ending with .log will be ignored by Git.

You can also ignore directories or specific files, for example:

```
/node_modules/    // ignore the node_modules folder
```

secret.txt   //ignore a specific file

Question 4: You are currently on the `main` branch and want to switch to a branch named `feature-login`. Write the Git command to do this.

Answer:

To switch from the main branch to the feature-login branch, you can use:

git switch feature-login

Question 5: Declare a variable named `age` using `let` and assign it the value `25`. Then, update its value to `30`.

Answer:

Declaring a variable named ageusing letand assign it the value 25. Then, update its value to 30.

```
// Declare the variable and assign it a value
let age = 25;

// Update the variable's value
age = 30;
```

After this, age will hold the value 30.

Question 6: Declare a constant named `PI` with the value `3.14159`. Attempting to reassign it should result in an error. Demonstrate this behavior by writing out the outcome.

Answer:

```
 // Declare a constant

const PI = 3.14159;



// Attempt to reassign it

PI = 3.14; // This will cause an error
```

Question 7: You are given the following array:

```
const colors = ["red", "green", "blue", "yellow", "purple"];
```

1. Access and print the first element of the array.

2. Access and print the last element of the array.
3. Access and print the third element of the array.

Answer:

```
const colors = ["red", "green", "blue", "yellow", "purple"];

// 1. Access and print the first element
console.log(colors[0]); // Output: "red"

// 2. Access and print the last element
console.log(colors[colors.length - 1]); // Output: "purple"

// 3. Access and print the third element
console.log(colors[2]); // Output: "blue"
```

Question 8: Given `x = 10` and `y = 5`, write expressions to:

1. Add `x` and `y`
2. Check if `x` is greater than `y`
3. Check if `x` is equal to `y`

Answer:

```
let x = 10;
let y = 5;

// 1. Add x and y
console.log(x + y); // Output: 15

// 2. Check if x is greater than y
console.log(x > y); // Output: true
```

// 3. Check if x is equal to y
console.log(x === y); // Output: false

Question 9: Identify the data types of the following values:

1. `"Hello"`
2. `42`
3. `true`
4. `undefined`

Answer :

Data types for the given values in JavaScript:

1. "Hello" → string
2. 42 → number
3. true → boolean
4. undefined → undefined