A TECHNICAL REPORT

ON STUDENT INDUSTRIAL WORK TRAINING EXPERIENCE SCHEME

(SIWES)

UNDERTAKING AT

FEMTECH INFORMATION TECHNOLOGY INSTITUTE

IBRAHIM TAIWO ROAD, OPPOSITE ILLORIN STADIUM COMPLEX,ILLORIN, KWARA

BY

Jamiu Habeeb Adesola

1910310067

DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF SCIENCE

USMANU DANFODIYO UNIVERSITY SOKOTO

FROM

FEBUARY 2023 TO APRIL 2023

ABSTRACT

This industrial training report presents what I learned and experience gained during my 3 months of industrial training undertaken at FEMTECH INFORMATION TECHNOLOGY INSTITUTE.

My training was on java programming, CSS (cascading style sheet), and Networking.

This report discusses what I learned about the java and CSS (cascading style sheet)during the training period and its relevance of the language in equipping people with programming knowledge in order to thrive in the real world.

The report also includes definition, component hardware of networking, basics and uses of hardware components of networking.

TABLE OF CONTENTS

First industrial attachment report

CHAPTER 1

INTRODUCTION TO TRAINING PROGRAM

1.0 PURPOSE OF TRAINING

The student industrial work experience scheme(SIWES) was initiated in 1973 by the Industrial Training Fund(ITF). This was to update practical knowledge of students in the universities, polytechnics and college of technology. It was aimed at bridging the gap between the theoretical knowledge in the industry by providing students with the opportunities to apply their educational knowledge in real world situations.

Over the years, SIWES has contributed immensely to building the common pool of technical and allied skills available in the Nigeria economy which are needed for the nation's industrial development.

It gives students the opportunity to blend the theoretical knowledge acquired in the class room and with practical hands on application of knowledge required to perform work in the industry. Also it prepares students for employment and makes the transition from school to the world of work easier after graduation.

I undertook my SIWES at Femtech Information Technology Institute located at Ibrahim Taiwo Road, Opposite Illorin Stadium Complex,Ilorin, Kwara

1.3 FEMTECH'S PROFILE

FEMTECH Information Technology Institute(FITI) is one of the best ICT institutes in Nigeria. FITI deploys efficient I.T solutions that are tailored towards providing effective I.T training for career development.

As a global service company providing IT training and solutions, They offer top-notch yet affordable services for individuals and businesses. Providing quality practical & technical hands-on learning experience for building topflight ICT professionals with certification, building befitting modern websites for personal and corporate brands and also help business growth with online presence & promotions.

CHAPTER 2

THE TRAINING PROGRAM

2.0 DESCRIPTION OF WORK DONE

During my stay in FEMTECH I was assigned to the ICT department as an IT student, where I was taught java, CSS (cascading style sheet). I learned about the basics of java and CSS (cascading style sheet) how to write programs and how to solve problems, and practicals where I put everything I have learned into play by writing semi complex programs.

2.JAVA PROGRAMMING

2.10 INTRODUCTION TO JAVA AND IT'S APPLICATION

Java is object oriented programming language,which is built around the concept of objects,which have their own state and behavior.this make it easier to organize and manage complex code.

Java program are platform independent, which means it can runs on any platform thst has a JVM (java virtual machine) installed. which makes it easier to develop applications that can runs on variety of devices and platforms.

Phases of java development life cycle

In Java application development, the Java development life cycle comprises the following phases:

- Planning
- Requirements Gathering
- Design
- Implementation
- Testing
- Deployment.

planning

Planning: Here, project owners and project managers collaborate to define the timeline of the project, come up with cost estimates, and determine the resources required for the project. They define quality metrics, milestones, deliverables, and dependencies, and decide on the responsibilities of each member.

Requirement gathering

Requirements gathering: The second phase of the Java development process is requirements gathering. In this phase, the team works together to gather all of the information necessary to start the project. This includes understanding the business goals, determining the project's scope, and creating user stories.

Design

Design: The third phase of the Java development process is design. In this phase, the team works together to create a high-level design for the application. This design will guide the implementation process and help ensure that all necessary functionality is included in the final products.

Implementation

Implementation: The fourth phase of the Java development process is implementation. In this phase, the team works together to turn the design into a working code. This involves writing classes, methods, and tests.

Testing

Testing: The fifth phase of the Java development process is testing. In this phase, the team works together to test the application to ensure it meets all requirements. This includes unit testing, integration testing, and user acceptance testing.

Deployment

Deployment: The sixth and final phase of the Java development process is deployment. In this phase, the team collaborates to deploy the application to a production environment. This includes setting up servers, configuring networking, and more.

2.12 STRING AND STRING LITERALS, COMMENTS, NUMERIC DATA TYPES, READING INPUT, VARIABLES

String and string literals

Programs almost always work with some sort of data type. In programming a sequence of characters that is used as data is called a *string*. When a string appears in the actual code of a program, it is called a *string literal,* which must be enclosed in a quote mark either single or double quotes.

Comments in java

Comments in java are the statements that are not executed by the compiler and interpreter. It can be used to provide information or explanation about the variable, method, class or any statement. It can also be used to hide program code for a specific time. They are various types of comments in java programming which include:

1 Single line comments

This type of comment is mainly used for describing the code functionality. It is the easiest typed comments. Single-line comments start with two forward slashes (//). Any text between // and the end of the line is ignored by Java implies it will not be executed.

```
1
                    public class Example1 {
2
                public static void main(String[] args) {
3
                   int u=100; //Here, u is a variable
4
            System.out.println(u); // Prints the value of the variable u
                                    }
5
                                    }
6
```

2 Multi-line comments

To describe a full method in a code or a complex snippet single line comments can be tedious to write since we have to give '//' at every line. So to overcome this multi-line comments can be used. Multi-line comments start with /* and end with */. Any text between /* and */ will be ignored by Java. It is used to comment multiple lines of code.

Example:

```
1
2                   public class Example2 {
3               public static void main(String[] args) {
4                  /* Here, let's declare and print
5                        a variable in java. */
6                           int u=100;
7                        System.out.println(u);
8                                    }
                                     }
```

3 Documentation comments

This type of comments is used generally when you are writing code for a project/ software package. It helps you to generate a documentation page for reference, which can be used for getting information about methods present, its parameters, etc.

Example

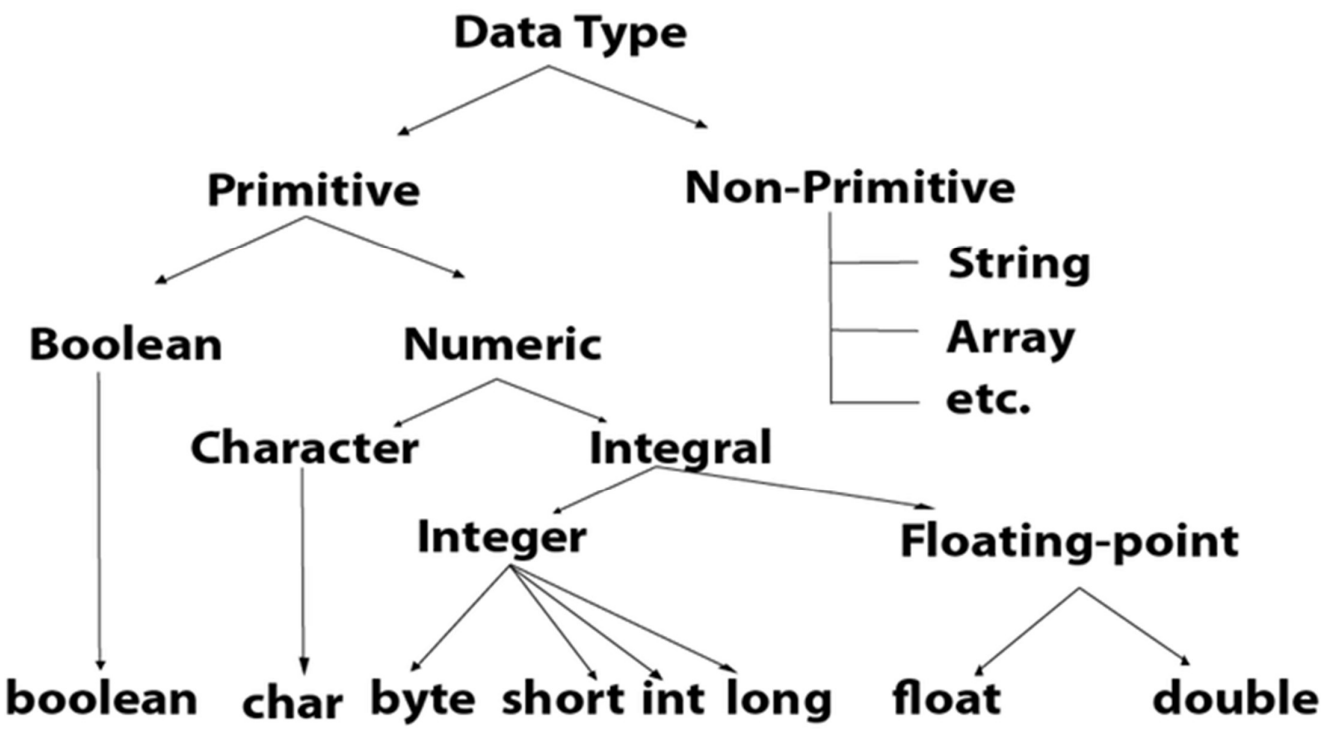
```
1                   public class Example3 {
2               public static void main(String[] args) {
3                  /* Here, let's declare and print
```

```
4                                   a variable in java. */
5                                         int u=100;
6                                   System.out.println(u);
7                                                 }
8                                                 }
```

Java data types

There are 8 types of primitive data types:

- Boolean data type
- byte data type
- char data type
- short data type
- int data type
- long data type
- float data type
- double data type



Boolean Data Type

The Boolean data type is used to store only two possible values: true and false. This data type is used for simple flags that track true/false conditions.

The Boolean data type specifies one bit of information, but its "size" can't be defined precisely.

Byte Data Type

The byte data type is an example of primitive data type. It isan 8-bit signed two's complement integer. Its value-range lies between -128 to 127 (inclusive). Its minimum value is -128 and maximum value is 127. Its default value is 0.

The byte data type is used to save memory in large arrays where the memory savings is most required. It saves space because a byte is 4 times smaller than an integer. It can also be used in place of "int" data type.

Example:

```
1. byte a = 10, byte b = -20
```

Short Data Type

The short data type is a 16-bit signed two's complement integer. Its value-range lies between -32,768 to 32,767 (inclusive). Its minimum value is -32,768 and maximum value is 32,767. Its default value is 0.

The short data type can also be used to save memory just like byte data type. A short data type is 2 times smaller than an integer.

Example:

```
1. short s = 20000, short r = -5000
```

Int Data Type

The int data type is a 32-bit signed two's complement integer. Its value-range lies between -2,147,483,648 (-2^31) to 2,147,483,647 (2^31 -1) (inclusive). Its minimum value is - 2,147,483,648and maximum value is 2,147,483,647. Its default value is 0.

The int data type is generally used as a default data type for integral values unless if there is no problem about memory.

Example:

```
1. int a = 600000, int b = 400000
```

Long Data Type

The long data type is a 64-bit two's complement integer. Its value-range lies between -9,223,372,036,854,775,808(-2^63) to 9,223,372,036,854,775,807(2^63 -1) (inclusive). Its minimum value is - 9,223,372,036,854,775,808and maximum value is 9,223,372,036,854,775,807. Its default value is 0. The long data type is used when you need a range of values more than those provided by int.

Example:

```
long a = 100000L, long b = -200000L
```

Float Data Type

The float data type is a single-precision 32-bit IEEE 754 floating point.Its value range is unlimited. It is recommended to use a float (instead of double) if you need to save memory in large arrays of floating point numbers. The float data type should never be used for precise values, such as currency. Its default value is 0.0F.

Example:

```
float f1 = 234.5f
```

Double Data Type

The double data type is a double-precision 64-bit IEEE 754 floating point. Its value range is unlimited. The double data type is generally used for decimal values just like float. The double data type also should never be used for precise values, such as currency. Its default value is 0.0d.

Example:

```
double d1 = 12.3
```

Char Data Type

The char data type is a single 16-bit Unicode character. Its value-range lies between '\u0000' (or 0) to '\uffff' (or 65,535 inclusive).The char data type is used to store characters.

Example:

```
char letter A = 'A'
```

Variables

A variable is a container which holds the value while the Java program is executed. A variable is assigned with a data type.

More also variable is the name of a reserved area allocated in memory. In other words, it is a name of the memory location. It is a combination of "vary + able" which means its value can be changed.

Variable is a name of memory location. There are three types of variables in java: local, instance and static.

There are two types of data types in Java: primitive and non-primitive.

Types of Variables

There are three types of variables in Java:

- local variable
- instance variable
- static variable

Local Variable

A variable declared inside the body of the method is called local variable. You can use this variable only within that method and the other methods in the class aren't even aware that the variable exists.

A local variable cannot be defined with "static" keyword.

Instance Variable

A variable declared inside the class but outside the body of the method, is called an instance variable. It is not declared as static.

It is called an instance variable because its value is instance-specific and is not shared among instances.

Static variable

A variable that is declared as static is called a static variable. It cannot be local. You can create a single copy of the static variable and share it among all the instances of the class. Memory allocation for static variables happens only once when the class is loaded in the memory.

Variables are created by using an *assignment operator* to create a variable and make it reference to a piece of data

There are rules to follow when naming a variable, which must be followed:

- Java keywords cannot be used as variable name
- A variable name cannot contain spaces
- The first character must be one of the a through z, or Z through A or underscores
- Every variable name is unique
- After the first character, you may use letters, underscores or numbers

In addition, variable should give an indication of what they are used for.

Java scanner class

Java **Scanner class** allows the user to take input from the console. It belongs to **java.util** package. It is used to read the input of primitive types like int, double, long, short, float, and byte. It is the easiest way to read input in Java program.

1. Scanner sc=**new** Scanner(System.in);

The above statement creates a constructor of the Scanner class having **System.inM** as an argument. It means it is going to read from the standard input stream of the program. The **java.util** package should be import while using Scanner class.

It also converts the Bytes (from the input stream) into characters using the platform's default charset.

Example of String Input from user:

```
1. import java.util.*;
2. class UserInputDemo1
3. {
4. public static void main(String[] args)
5. {
6. Scanner sc= new Scanner(System.in); //System.in is a standard input stream
```

```
7.  System.out.print("Enter a string: ");
8.  String str= sc.nextLine();              //reads string
9.  System.out.print("You have entered: "+str);
10. }
11. }
```

2.13 ARITHMETIC OPERATIONS, TYPE CASTING, ESCAPE CHARACTERS

Arithmetic operations

Java has a numerous operators that can be used to perform mathematical operations such as :

+ addition * multiplication

- subtraction ** exponent

/ division

% modulus

Java Type casting

In Java, **type casting** is a method or process that converts a data type into another data type in both ways manually and automatically. The automatic conversion is done by the compiler and manual conversion performed by the programmer.

Types of Type Casting

There are two types of type casting:

- Widening Type Casting
- Narrowing Type Casting

Widening Type Casting

Converting a lower data type into a higher one is called **widening** type casting. It is also known as **implicit conversion** or **casting down**. It is done automatically. It is safe because there is no chance to lose data. It takes place when:

- Both data types must be compatible with each other.
- The target type must be larger than the source type.

1. **byte** -> **short** -> **char** -> **int** -> **long** -> **float** -> **double**

For example, the conversion between numeric data type to char or Boolean is not done automatically. Also, the char and Boolean data types are not compatible with each other.

Narrowing Type Casting

Converting a higher data type into a lower one is called **narrowing** type casting. It is also known as **explicit conversion** or **casting up**. It is done manually by the programmer. If we do not perform casting then the compiler reports a compile-time error.

1. **double** -> **float** -> **long** -> **int** -> **char** -> **short** -> **byte**

an example of narrowing type casting.

In the following example, the narrowing type casting are performed two times. First, we have converted the double type into long data type after that long data type is converted into int type.

2.14 Method in java, Java conditional statement, logical operators, Boolean expression, random numbers.

Method in Java

method is a way to perform some task. Similarly, the **method in Java** is a collection of instructions that performs a specific task. It provides the reusability of code. We can also easily modify code using **methods**. In this section, we will learn **what is a method in Java, types of methods, method declaration,** and **how to call a method in Java**.

method is a block of code or collection of statements or a set of code grouped together to perform a certain task or operation. It is used to achieve the **reusability** of code. We write a method once and use it many times. We do not require to write code again and again. It also provides the **easy modification** and **readability** of code, just by adding or removing a chunk of code. The method is executed only when we call or invoke it.

Method Declaration

The method declaration provides information about method attributes, such as visibility, return-type, name, and arguments. It has six components that are known as **method header** .

Naming a Method

While defining a method, remember that the method name must be a **verb** and start with a **lowercase** letter. If the method name has more than two words, the first name must be a verb followed by adjective or noun. In the multi-word method name, the first letter of each word must be in **uppercase** except the first word.

Single-word method name: sum(), area()

Multi-word method name: areaOfCircle(), stringComparision()

It is also possible that a method has the same name as another method name in the same class, it is known as **method overloading**.

Types of Method

There are two types of methods in Java:

- Predefined Method
- User-defined Method

Predefined Method

In Java, predefined methods are the method that is already defined in the Java class libraries is known as predefined methods. It is also known as the **standard library method** or **built-in method**. We can directly use these methods just by calling them in the program at any point. Some pre-defined methods are **length(), equals(), compareTo(), sqrt(),** etc. When we call any of the predefined methods in our program, a series of codes related to the corresponding method runs in the background that is already stored in the library.

Each and every predefined method is defined inside a class. Such as **print()** method is defined in the **java.io.PrintStream** class. It prints the statement that we write inside the method. For example, **print("Java")**, it prints Java on the console.

Example

```
1. public class Demo
2. {
3. public static void main(String[] args)
4. {
5. // using the max() method of Math class
6. System.out.print("The maximum number is: " + Math.max(9,7));
7. }
8. }
```

Output:

```
The maximum number is:9
```

User-defined Method

The method written by the user or programmer is known as **a user-defined** method. These methods are modified according to the requirement.

How to Create a User-defined Method

Let's create a user defined method that checks the number is even or odd. First, we will define the method.

```
1. //user defined method
2. public static void findEvenOdd(int num)
3. {
4. //method body
5. if(num%2==0)
6. System.out.println(num+" is even");
7. else
8. System.out.println(num+" is odd");
9. }
```

Output

```
Enter the number: 12
12 is even
```

Java Conditional statements

If statement the if statement is used to create a decision structure,which allows a program to have more than one part of execution.

Example:

```
1. if(condition) {
2. statement 1; //executes when condition is true
3. }
4.
```

elif statement to test more than one condition, a decision structure can be nested inside another

example:

```
1. if(condition 1) {
2. statement 1; //executes when condition 1 is true
3. }
4. else if(condition 2) {
5. statement 2; //executes when condition 2 is true
6. }
7. else {
8. statement 2; //executes when all the conditions are false
```

```
9. }
```

else statement will execute if the preceding conditional statements are false.

Example:

```
1. if(condition) {
2. statement 1; //executes when condition is true
3. }
4. else{
5. statement 2; //executes when condition is false
6. }
7.
```

Rational operators

Rational operators determines whether a specific relationship exist between two values.

- > greater than

- < less than

- >= greater than or equal to

- <= less than or equal to

- == equal to

- != not equal to

Boolean expression

Conditional statements test an expression to determine whether it is true or false. The expression that are tested are called *Boolean expression,* named in honor of the English mathematician George Boole.

Logical operators

The logical operator *and* and *or* allows you to connect multiple Boolean expression. The logical *not* reverses the truth of a Boolean expression.

- and

- or

- not

2.15 Java loop statement

Loop statement

In Java, we have three types of loops that execute similarly. However, there are differences in their syntax and condition checking time.

1. for loop
2. while loop
3. do-while loop

Java for loop

In Java, for loop is similar to C and C++. It enables us to initialize the loop variable, check the condition, and increment/decrement in a single line of code. We use the for loop only when we exactly know the number of times, we want to execute the block of code.

Example

```
1.    for(initialization, condition, increment/decrement) {
2. //block of statements
3. }
```

Java while loop

The while loop is also used to iterate over the number of statements multiple times. However, if we don't know the number of iterations in advance, it is recommended to use a while loop. Unlike for loop, the initialization and increment/decrement doesn't take place inside the loop statement in while loop.

Example

```
1. while (condition){
2. //code to be executed
3. Increment / decrement statement
4. }
```

2.16 EXCEPTIONS

Exception

An exception Is an error that occurs while a program is running, causing the program to abruptly halt. Exceptions can be handled by using the *try/except* statement.

2.CSS (cascading style sheets)

2.21 INTRODUCTION TO CSS AND IT'S APPLICATION

CSS stands for **CASCADING STYLE SHEETS** it's a language used to described the visual presentation of a webpage. It works by defining the styles of HTML elements, such as font, colors, layouts, and more.

CSS has a simple and intuitive syntax. It consists of selectors target specific HTML elements to apply styles to, and declarations specify the styles that should be applied. CSS also supports a variety of layout techniques, such as flexbox and grind, which make it easier to create complex page layouts.

Some of the key features and benefit of CSS include:

- Selectors: CSS allows developers to target specific HTML elements, classes, or IDs with selectors, making it easy to apply styles selectively.
- Box model: CSS defines how boxes are displayed on a web page, including their size, padding, and border.
- Responsive design: CSS offers flexible layouts that can adapt to different screen sizes and device types, allowing developers to create responsive designs that look great on desktops, tablets, and mobile devices.
- Modular development: CSS allows developers to organize their styles into separate files and modules, making it easier to manage complex projects.
- Browser compatibility: CSS is supported by all modern browsers, which means that developers can create consistent styles across different platforms and devices.

2.2 APPLICATIONS OF CSS

Controlling Effects and Flash Animation

The use of CSS streamlines the process of placing and handling flash elements on websites. By applying the built-in Flash files, styling sheets, and frameworks in CSS, developers can easily create animations, make movies, and update effects on web pages.

Dynamic Web Templates Managing

Dynamic web templates are HTML-based parent copies of web pages that contain various page elements and settings, such as styles, page layouts, graphics, and text. The use of CSS frameworks and extensions enables the creation of dynamic web templates, which allow developers to seamlessly add, edit, and manage the dynamic elements of websites that help style web pages.

Image File Management

Features integrated into CSS frameworks not only allow developers to format, edit, and update different image formats but also enable image cropping and the creation of watermarks and thumbnails. Styling, editing, and updating a wide variety of image types, such as png, gif, and jpeg, were challenging tasks before CSS made its appearance in the web development domain.

Improved User Experience

Cascading Style Sheets not only delivers web pages that are visually pleasing but also helps developers do user-friendly page formatting. When text, images, and buttons are well-organized, the user experience automatically gets enriched.

Device Compatibility

In today's digital environment, responsive web designs matter. Nowadays, web pages should be easily navigable and fully visible on any device, be it a desktop computer, a tablet, a smartphone, or a smart TV. CSS integrates into HTML to enable responsive web designs.

CSS syntax

CSS rule set contains a selector and a declaration block.

Selector: Selector indicates the HTML element you want to style. It could be any tag like <h1>, <title> etc.

Declaration Block: The declaration block can contain one or more declarations separated by a semicolon. For the above example, there are two declarations:

1. color: yellow;
2. font-size: 11 px;

Each declaration contains a property name and value, separated by a colon.

Property: A Property is a type of attribute of HTML element. It could be color, border etc.

Value: Values are assigned to CSS properties. In the above example, value "yellow" is assigned to color property.

Types of CSS

CSS (Cascading Style Sheet) describes the HTML elements which are displayed on **screen, paper**, or in **other media**. It saves a lot of time. It controls the layout of multiple web pages at one time. It sets the **font-size, font-family, color, background color** on the page.

It allows us to add **effects** or **animations** to the website. We use **CSS** to display **animations** like **buttons, effects, loaders** or **spinners**, and also **animated backgrounds**.

Without using **CSS**, the website will not look attractive. There are **3** types of **CSS** which are below:

- Inline CSS
- Internal/ Embedded CSS
- External CSS

1. Internal CSS

The Internal CSS has **<style>** tag in the **<head>** section of the **HTML** document. This CSS style is an effective way to style single pages. Using the CSS style for multiple web pages is time-consuming because we require placing the **style** on each web page.

We can use the internal CSS by using the following steps:

1. Firstly, open the **HTML** page and locate the **<head>**

2. Put the following code after the **<head>**

```
<style type="text/css">
```

3.Add the **rules** of CSS in the new line

4. Close the style tag

2. External CSS

In external CSS, we link the web pages to the external **.css** file. It is created by **text editor**. The CSS is more efficient method for styling a website. By editing the **.css** file, we can change the whole site at once.

To use the external CSS, follow the steps, given below:

1. Create a new **.css** file with **text editor**, and add **Cascading Style Sheet** rules
2. Add a reference to the external **.css**file right after **<title>** tag in the **<head>** section of **HTML sheet**. **<link** rel="stylesheet" type="text/css" href="style.css" **/>**

Pros of External CSS:

- Our files have a cleaner structure and smaller in size.
- We use the same **.css** file for multiple web pages in external CSS.

Cons of External CSS:

- The pages cannot be delivered correctly before the external CSS is loaded.
- In External CSS, uploading many CSS files can increase the download time of a website.

3. Inline CSS

Inline CSS is used to style a specific **HTML** element. Add a **style** attribute to each HTML tag without using the selectors. Managing a website may difficult if we use only **inline CSS**. However, Inline **CSS** in HTML is useful in some situations. We have not access the **CSS files** or to apply styles to element.

Example

```
<!DOCTYPE html>
<html>
<body style="background-color:white;">
<h1 style="color:Red;padding:20px;">CSS Tutorials</h1>
<p style="color:blue;">It will be useful here.</p>
</body>
```

7. **</html>**

Pros of inline CSS:

- We can create CSS rules on the HTML page.
- We cannot create and upload a separate document in inline CSS.

Cons of inline CSS:

- Inline CSS, adding **CSS** rules to HTML elements is **time-consuming** and **messes** up the HTML structure.
- It styles multiple elements at the same time which can affect the page size and download time of the page.

SUMMARY

This technical report is written based on the knowledge and experience gained during my ten weeks industrial training performed at FEMTECH IT INSTITUTE, ILLORIN and this report is mainly on java and CSS (cascading style sheets).

CONCLUSION

Industrial training is key factor in the intellectual and social development of the students. This scheme has exposed me to the world of work, industrial practical and working methods, which will be expected of me when functioning professionally after graduation. It has also made me be aware of my strengths, witnesses and a lot of challenge ahead to attain the status of real computer scientist.

I would say that with this training I have a good knowledge of java Programming and CSS