**Gregorio, Mark C.**

**BSIT-3A**

**MODULE 1**

1**. Explain the concept of web development tools and how is important in web application development?**

Web development tools allow developers to work with a variety of web technologies, including HTML, CSS, the DOM, JavaScript, and other components that are handled by the web browser. Due to increasing demand from web browsers to do more, popular web browsers have included more features geared for developers.

2**. Discuss the difference between server-side scripting and client-side scripting and cite examples.**

The server-side scripting has complete access to all the files present in any web server. The client-side scripting has no access to the files that exist in a web server. Languages like Ruby on Rails, Perl, ASP, Python, ColdFusion, PHP, etc., come into play in the case of server-side scripting.

3**. What is the purpose of the framework in developing an application?**

Software engineers and developers use a framework as a template to create websites and applications. Developers do this by adding code to a framework, then personalizing it for their specific purpose. A framework can combine multiple resources, such as an image or document file, to create a package unique to a project.

4**. Cite at least one example of why a framework is important in building web applications and explain.**

Some examples would be Nextjs, Django and Express. Web application frameworks are essential for software engineering because they provide a set of reusable components, libraries, and tools that simplify the development process, increase productivity, and improve the overall quality of web applications.

5**. Aside from the listed standard web technology tools cite at least one example of each tool.**

**Web Browser:**

Example: Google Chrome

**HTML (Hypertext Markup Language):**

Example: <h1>Hello, World!</h1>

**CSS (Cascading Style Sheets):**

Example: color: #3498db;

**JavaScript:**

Example: const greeting = "Hello, World!";

**HTTP (Hypertext Transfer Protocol):**

Example: GET /index.html HTTP/1.1

**Web Server:**

Example: Apache HTTP Server

**Database:**

Example: MySQL

**AJAX (Asynchronous JavaScript and XML):**

Example: Making asynchronous requests with XMLHttpRequest in JavaScript.

**JSON (JavaScript Object Notation):**

Example: {"name": "John", "age": 25}

**API (Application Programming Interface):**

Example: Twitter API for accessing and interacting with Twitter data.

**MODULE 2**

1. **Create a Database and name it as db1_sample.**

   CREATE DATABASE db1_sample;

2. **Create a Table inside the database that you have just created and name it as tbl1_sample. Your table should look like the one given below:**

| Id | LastName | FirstName | CurrentYear | Course |
|---|---|---|---|---|
| | | | | |

   **Note: The Id field should be mark with the AUTO_INCREMENT flag so that the modifier will tell the MySQL to automatically assign a value to this field if it is left unspecified, by incrementing the previous value by 1.**

```
CREATE TABLE tbl1_sample (
  Id INT AUTO_INCREMENT PRIMARY KEY,
  LastName VARCHAR(255),
  FirstName VARCHAR(255),
  CurrentYear INT,
  Course VARCHAR(255)
);
```

3. **Using the Insert Into statement, insert the following data below on the database you have just created.**

| | | | |
|---|---|---|---|
| **Lee** | **Peter** | **2nd** | **BSIT** |
| **Edwards** | **Jonathan** | **3rd** | **BSCS** |
| **Johnson** | **Marilyn** | **1st** | **BSEd** |
| **Kim** | **Joe** | **1st** | **BSCE** |
| **Martinez** | **Haley** | **2nd** | **BSEE** |
| **Smith** | **John s** | **3rd** | **BSCA** |
| **Letty** | **David** | **4th** | **BSECE** |

```
INSERT INTO tbl1_sample (LastName, FirstName, CurrentYear, Course)
VALUES
  ('Lee', 'Peter', 2, 'BSIT'),
  ('Edwards', 'Jonathan', 3, 'BSCS'),
  ('Johnson', 'Marilyn', 1, 'BSEd'),
  ('Kim', 'Joe', 1, 'BSCE'),
  ('Martinez', 'Haley', 2, 'BSEE'),
  ('Smith', 'John', 3, 'BSCA'),
  ('Letty', 'David', 4, 'BSECE');
```

4. **Display all the data from the database using table.**

```
MariaDB [db1_sample]> SELECT * FROM tbl1_sample;
+----+----------+-----------+-------------+--------+
| Id | LastName | FirstName | CurrentYear | Course |
+----+----------+-----------+-------------+--------+
|  1 | Lee      | Peter     |           2 | BSIT   |
|  2 | Edwards  | Jonathan  |           3 | BSCS   |
|  3 | Johnson  | Marilyn   |           1 | BSEd   |
|  4 | Kim      | Joe       |           1 | BSCE   |
|  5 | Martinez | Haley     |           2 | BSEE   |
|  6 | Smith    | John      |           3 | BSCA   |
|  7 | Letty    | David     |           4 | BSECE  |
+----+----------+-----------+-------------+--------+
7 rows in set (0.001 sec)
```

***Assessment Task***

1. **Create a database and name it as db_info.**

   CREATE DATABASE db_info;

2. **Create a table and name it as tbl_stud and input all the records the same as the table above.**

```
CREATE TABLE tbl_stud (
  id INT PRIMARY KEY,
  first_name VARCHAR(50),
  last_name VARCHAR(50),
  email VARCHAR(100),
  age INT
);

INSERT INTO tbl_stud (id, first_name, last_name, email, age) VALUES
(1, 'Peter', 'Parker', 'peterparker@mail.com', 17),
(2, 'John', 'Rambo', 'johnrambo@mail.com', 18),
(3, 'Clark', 'Kent', 'clarkkent@mail.com', 18),
(4, 'John', 'Carter', 'johncarter@mail.com', 19),
(5, 'Harry', 'Potter', 'harrypotter@mail.com', 18),
(6, 'Henry', 'Sy', 'henrysy@mail.com', 17);
```

3. **Using the WHERE CLAUSE, display in a table where only students have ages of 18.**

```
SELECT * FROM tbl_stud WHERE age = 18;
```

*Assessment Task*

1. **Using the LIKE Clause, create a PHP script that will return all the records from the tbl1 for which the last name contains er.**

```
<?php
// Database connection parameters
$servername = "hostname";
$username = "root ";
$password = " ";
$dbname = db_info"";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// SQL query with the LIKE clause
$sql = "SELECT * FROM tbl_stud WHERE last_name LIKE '%er%'";

// Execute the query
$result = $conn->query($sql);

// Check if any records were found
if ($result->num_rows > 0) {
  // Output data of each row
  while ($row = $result->fetch_assoc()) {
    // Print or process the data as needed
    echo "ID: " . $row["id"] . " - First Name: " . $row["first_name"] . " - Last Name: " .
$row["last_name"] . "<br>";
  }
} else {
  echo "No records found";
}

// Close connection
$conn->close();
?>
```

**MODULE 3**

1. **What are the different SQL commands? Enumerate and provide concrete example of its usage.**

- **Data Definition Language**

Data definition language (DDL) refers to SQL commands that design the database structure. Database engineers use DDL to create and modify database objects based on the business requirements. For example, the database engineer uses the CREATE command to create database objects such as tables, views, and indexes.

- **Data Query Language**

Data query language (DQL) consists of instructions for retrieving data stored in relational databases. Software applications use the SELECT command to filter and return specific results from a SQL table.

- **Data Manipulation Language**

Data manipulation language (DML) statements write new information or modify existing records in a relational database. For example, an application uses the INSERT command to store a new record in the database.

- **Data Control Language**

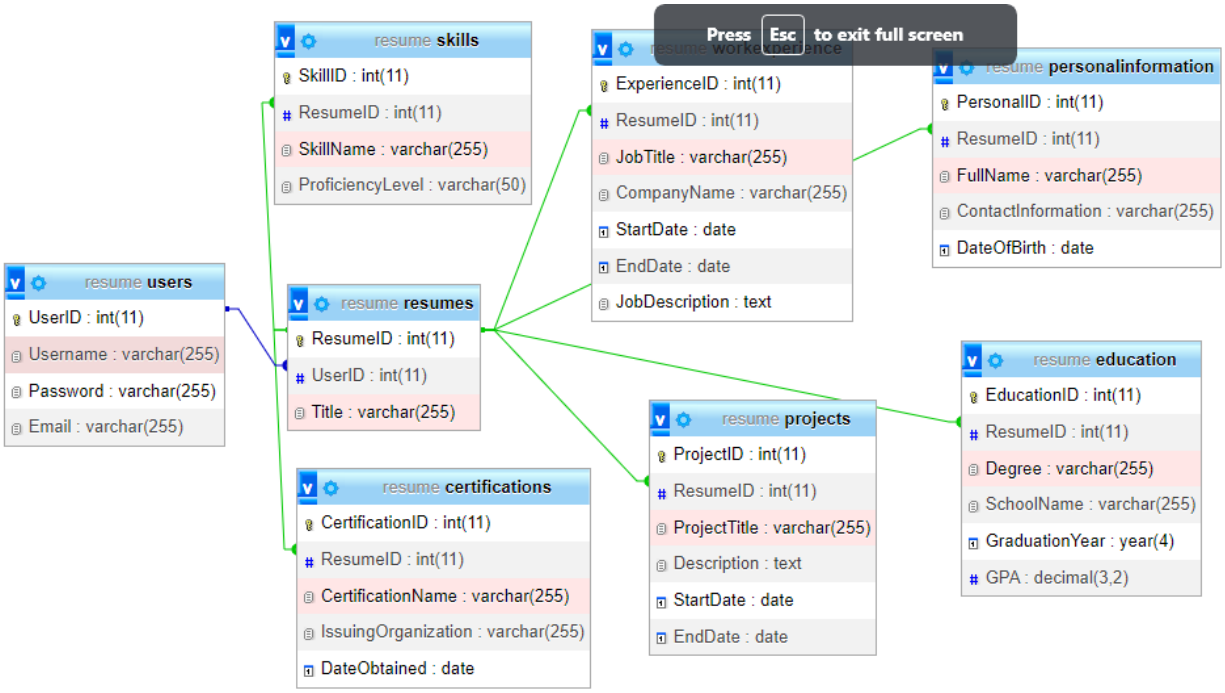
Database administrators use data control language (DCL) to manage or authorize database access for other users. For example, they can use the GRANT command to permit certain applications to manipulate one or more tables.

- **Transaction Control Language**

The relational engine uses transaction control language (TCL) to automatically make database changes. For example, the database uses the ROLLBACK command to undo an erroneous transaction.

**MODULE 4**

1. **Create a database schema of your final website project.**




2. **Create a website layout or wireframe design as you conceptualize how your webpages will look like.**

**MODULE 5**

1. **What is AJAX and explain about it?**

   Ajax is a set of web development techniques using many web technologies on the client side to create asynchronous web applications. With Ajax, web applications can send and retrieve data from a server asynchronously (in the background) without interfering with the display and behavior of the existing page. By decoupling the data interchange layer from the presentation layer, Ajax allows web pages and, by extension, web applications, to change content dynamically without the need to reload the entire page. In practice, modern implementations commonly utilize JSON instead of XML.

2. **What are the different technologies used in AJAX?**

   Ajax is a collection of technologies all of which have been around for a number of years. Each of these technologies was developed for various reasons that had nothing to do with Ajax. However, because of their complementary nature, web developers have discovered that when used together, these technologies provide a robust and powerful environment for creating and running web applications. The individual technologies that make up Ajax include:

   • **JavaScript.** The programming language used to develop Ajax applications, tying together the interaction of all of the other Ajax technologies.
   • **XML**. Provides a means of exchanging structured data between the web server and client. • **The XMLHttpRequest object**. Provides the ability to asynchronously exchange data between web browsers and a web server.
   • **HTML and CSS.** Provides the ability to mark up and style the display of web page text.
   • **The Document Object Model or DOM.** Provides the ability to dynamically interact with and alter the web page layout and content.

3. **Differentiate synchronous request and asynchronous request in AJAX?**

   Synchronous Ajax request is the process in which execution of the request stops until a response is received and Asynchronous Ajax request means the script continue the process without waiting for the server to reply. It will handle the reply if it arrives.

4. **What are different ready states in AJAX?**

**0 (UNSENT)**: The request has been created, but the open() method has not been called yet. This is the initial state.

**1 (OPENED**): The open() method has been called. During this state, you can use the setRequestHeader() method to set HTTP headers, but the request has not been sent yet.

**2 (HEADERS_RECEIVED):** The send() method has been called, and the headers of the response are available. You can access them using the getAllResponseHeaders() method.

**3 (LOADING)**: The response is being received. You can access the partial response using the responseText property. This state is useful for showing progress updates to the user.

**4 (DONE):** The operation is complete. The entire response has been received, and you can use the responseText or responseXML properties to get the complete response.5. What are the different stages and processes in AJAX ready states?

5. **What is XMLHTTPRequest Object?**
   XMLHttpRequest (XHR) objects are used to interact with servers. You can retrieve data from a URL without having to do a full page refresh. This enables a Web page to update just part of a page without disrupting what the user is doing.

6. **What are the uses of XMLHTTPRequest Object in AJAX?**
   XMLHttpRequest is an object that is used to send a request to the webserver for exchanging data or transferring and manipulating to it and from the server behind the scenes. You can use the received data to update the data present on the web page without even reloading the page.

7. **What are AJAX applications in web development?**

   **Asynchronous Requests:** AJAX enables the sending and receiving of data from the server without requiring a complete page refresh. This is typically achieved using the XMLHttpRequest object in JavaScript or more modern approaches like the Fetch API.

   **Data Interchange Format:** While XML is part of the acronym, it's not always used in modern AJAX applications. JSON (JavaScript Object Notation) has become more popular due to its simplicity and ease of use with JavaScript. JSON is often used to format the data sent between the server and the client.

   **DOM Manipulation:** After receiving data from the server, JavaScript can manipulate the Document Object Model (DOM) to update specific parts of the page without reloading the entire page. This provides a smoother and more responsive user experience.

   **Event Handling:** AJAX applications often involve handling events triggered by user interactions. These events can trigger asynchronous requests to the server, update the DOM, or perform other actions to enhance the interactivity of the web page.

   **Cross-browser Compatibility:** Developers need to ensure that AJAX applications work consistently across different web browsers. This involves considering variations in the implementation of JavaScript and XMLHttpRequest across browsers.

8. **What are the real-time AJAX web applications?**
   AJAX is the cornerstone of real-time functionality, enabling chat systems, live updates, and collaborative applications. It fosters dynamic user engagement by allowing users to interact with each other and with the application in real-time.

9. **What are the advantages of AJAX?**

Some key advantages include a more responsive user experience, faster page loading, efficient bandwidth usage, and support for interactive features like live search, notifications, and chat systems.

**10. Provide disadvantages of AJAX?**

- Open-source. View source is allowed, and anyone can view the code source written for Ajax, which makes it less secure compared to other technologies

- Search Engines cannot index Ajax pages can not be indexed by Google as well as other search engines

- The usage of Ajax can cause difficulties for your web pages to debug as well as make them prone to possible security issues in the future

- Most importantly, Ajax has a considerable dependency on JavaScript, so only browsers that support Javascripts or XMLHttpRequest can use pages with Ajax techniques

- Users will find it challenging to bookmark a specific state of the application due to the dynamic web page

- From the users' perspective, when you click the back button on the browser, you may not return to the previous state of the page but the entire page. This happens because the pages with successive Ajax requests are unable to register with the browser's history

**MODULE 6**

1. **What is web security?**

   Web Security is very important nowadays. Websites are always prone to security threats/risks. Web Security deals with the security of data over the internet/network or web or while it is being transferred to the internet. For e.g. when you are transferring data between client and server and you have to protect that data that security of data is your web security. Hacking a Website may result in the theft of Important Customer Data, it may be the credit card information or the login details of a customer or it can be the destruction of one's business and propagation of illegal content to the users while somebody hacks your website they can either steal the important information of the customers or they can even propagate the illegal content to your users through your website so, therefore, security considerations are needed in the context of web security.

2. **What is the purpose of web security?**

   The massive importance of the internet for modern enterprises — and the accompanying growth in the sophistication, frequency, and impact of cyberattacks — has made web security critical to business continuity. It's your first line of defense against threats that can lead to the exposure of sensitive data, costly ransoms, reputational harm, compliance violations, and a host of other consequences. Once the domain of mostly small-time hackers, internet-borne threats have evolved into a massive black-market business that touches the worlds of organized crime as well as state-sponsored espionage and sabotage. Some of the latest threats are incredibly sophisticated, able to easily fool the untrained eye or bypass legacy security. Plus, with an array of ready-made tools, exploit kits, JavaScript modules, and even fully developed campaigns for sale, even a novice actor can easily launch an attack. Cybersecurity Ventures estimates that, by 2025, global cybercrime will cost US$10.5 trillion annually — a greater profit than the entire world's major illicit drug trade — and half the world's data will live in the cloud. Given what's at stake, it's easy to see why effective web security is so important today. To comply with internal policies, government-imposed criteria, or Open Web Application Security Project (OWASP) standards, security professionals consider a variety of factors. Keeping abreast with OWASP standards helps security staff stay up to date with industry-standard web safety expectations. In addition, encryption must be kept up to date, the latest threats in the Web Hacking Incident Database (WHID) monitored, and user authentications properly managed. When vulnerabilities emerge, security personnel must install the most recent patches to address them. To secure data, software development teams must implement protocols that shield code from being stolen during or after writing it.

3. **What are web security threats?**

   Web security casts a wide net to protect users and endpoints from malicious emails, encrypted threats, malicious or compromised websites and databases, malicious redirects, hijacking, and more. Let's look at a few of the most common threats in more detail:

   • **Ransomware:** These attacks encrypt data, and then demand a ransom payment in exchange for a decryption key. In a double-extortion attack, your data is also exfiltrated.

• **General malware:** Countless variants of malware exist that can lead to anything from data leaks, spying, and unauthorized access to lockouts, errors, and system crashes.
• **Phishing:** Often carried out through email, text messages, or malicious websites, these attacks trick users into things like divulging login credentials or downloading spyware.
• **SQL injection:** These attacks exploit an input vulnerability in a database server, allowing an attacker to execute commands that let them retrieve, manipulate, or delete data.
• **Denial of service (DoS):** These attacks slow or even shut down a network device such as a server by sending it more data than it can process. In distributed DoS—that is, a DDoS attack—this is carried out by many hijacked devices at once.
• **Cross-site scripting (XSS):** In this type of injection attack, an attacker introduces malicious code to a trusted website by entering it in an unprotected user input field

4. **What are the three most common security threats?**

**Malware:** Malicious software, including viruses, worms, trojans, ransomware, and spyware, poses a significant threat. Malware is designed to infiltrate and damage computer systems or steal sensitive information.

**Phishing Attacks:** Phishing involves tricking individuals into providing sensitive information such as usernames, passwords, or financial details by posing as a trustworthy entity. Phishing attacks often come in the form of deceptive emails, messages, or websites.

**Distributed Denial of Service (DDoS) Attacks:** DDoS attacks aim to overwhelm a network, system, or website with a flood of traffic, rendering it inaccessible to users. Attackers may use botnets or other means to generate a massive volume of requests, disrupting normal service.

5. **What is security risk?**

The possibility of damage or loss as a result of a security breach or vulnerability is referred to as security risk. It includes the possibility that a danger will take advantage of a weakness and the possible effects it may have on the resources, business, and standing of an organization. By putting web security measures into place, security risks can be reduced and possible threats can be avoided.