WEB-ENGINEERING

By: Sagar Chhabriya (Batch CS-2k22)

SEM-5

SUBMITTED TO:
Sir Mohammad Faiz Lakhani

Contents

Introduction to Web Engineering

Web Engineering is the discipline that applies engineering principles to the development of web applications. It covers all aspects of web development including design, programming, architecture, testing, deployment, and maintenance. The goal is to ensure that web applications are scalable, secure, efficient, and user-friendly.

Types of Web Applications

Web applications can be classified into various types based on their functionality and user interaction. Below are some of the primary types of web applications:

Document Centric

- Document-centric web applications focus on delivering documents or content to users. Examples include news websites, blogs, and educational sites. These sites primarily serve static or semi-static content.

Interactive

- These web applications enable real-time interaction between the user and the application. Examples include online banking, e-commerce websites, and customer service platforms. They allow dynamic content updates and user input handling.

Workflow-Based

- Workflow-based web applications manage business processes that involve multiple tasks, roles, or approval stages. Common examples include enterprise resource planning (ERP) systems and business process management (BPM) applications.

Portal-Oriented

- Portal-oriented web applications aggregate information from multiple sources, providing a central hub for users. Examples include online education portals, employee intranets, and government service portals.

Social and Collaborating Web

- These applications focus on social interaction and collaboration. Social media platforms like Facebook and Twitter, as well as collaborative tools such as Google Docs or Slack, are examples of this category.

Semantic Web

- The semantic web enhances the traditional web by adding metadata to data, making it easier to interpret and process by machines. It is designed to facilitate interoperability across different systems and applications. Technologies like RDF, OWL, and SPARQL are key components of the semantic we

Web Application Architecture

Web application architecture is crucial for building scalable and maintainable web applications. Below are the common types of architectures used in web development:

Single Tier

- In a single-tier architecture, the application's presentation layer and data layer are combined into a single entity. This is typically used for small-scale applications or testing purposes, where the server and client are located on the same system.

Client-Server (Two Tier)

- A client-server architecture consists of two layers: the client (user interface) and the server (data storage and processing). In this architecture, the client sends requests to the server, and the server responds with the requested data.

Three Tier

- The three-tier architecture separates the presentation, logic, and data layers. It enhances scalability and maintainability by distributing each layer to separate components or servers. The three tiers are:
  - **Presentation Tier** (User interface)
  - **Application Tier** (Business logic)
  - **Data Tier** (Database)

Attributes of Web Applications

Web applications possess various attributes that ensure they are user-friendly, scalable, and efficient. These attributes include:

- **Scalability**: The ability to handle increasing numbers of users or data.
- **Reliability**: The capacity of an application to operate without failures over time.
- **Security**: Measures to protect the application from malicious attacks and unauthorized access.
- **Performance**: The ability of the application to respond quickly to user actions.
- **Usability**: The ease with which users can interact with the application.

Design Guidelines for Web Application Development

1. **User-Centered Design**: Focus on the needs, preferences, and limitations of users.
2. **Responsive Design**: Ensure the application is usable across a range of devices with varying screen sizes.
3. **Consistency**: Maintain uniformity in the design to make the application intuitive.

4. **Performance Optimization**: Ensure fast loading times and minimal delays.
5. **Security Best Practices**: Implement robust security measures to protect user data and privacy.

Introduction to HTML (Hypertext Markup Language)

HTML is the standard markup language for creating web pages. It structures the content of a webpage, defining elements like headings, paragraphs, links, tables, and multimedia.

Types of HTML Versions

1. **HTML 4.01**: The last version of HTML before the advent of XHTML.
2. **XHTML**: A stricter version of HTML, based on XML.
3. **HTML5**: The latest version, supporting multimedia, graphics, and mobile devices.

HTML Tags

1. **Meta Tags**
    - Meta tags provide metadata about the HTML document, such as author information, character encoding, and keywords for search engines.

    Example: <meta charset="UTF-8">
2. **Table Tags**
    - Tables are created using <table>, with rows (<tr>) and cells (<td>).

    Example:

    ```
    <table>
     <tr>
      <td>Header 1</td>
      <td>Header 2</td>
     </tr>
     <tr>
      <td>Row 1, Cell 1</td>
      <td>Row 1, Cell 2</td>
     </tr>
    </table>
    ```
3. **Form Elements**
    - Forms allow users to input data. Key elements include <input>, <select>, <textarea>, and <button>.

      Example:

      ```
      <form action="/submit">
      ```

```
  <input type="text" name="name" placeholder="Enter your name">
  <input type="submit" value="Submit">
</form>
```

4. **Div and Span**
   - <div> is a block-level element used to group content, while <span> is an inline element.

     Example:

     ```
     <div>
       <span>Example text</span>
     </div>
     ```

5. **Lists**
   - HTML supports both ordered (<ol>) and unordered (<ul>) lists.

     Example:

     ```
     <ul>
       <li>Item 1</li>
       <li>Item 2</li>
     </ul>
     ```

Introduction to CSS (Cascading Style Sheets)

CSS is used to style HTML elements and control their layout, appearance, and positioning on the web page. It can be applied inline, internally within the HTML document, or externally in a separate file.

Inserting CSS

- **Inline CSS**: CSS applied directly within an HTML element using the style attribute.
  - **Example:** `<p style="color: red;">This is a red paragraph.</p>`
- **Internal CSS**: CSS defined within the <style> tag in the <head> section of the HTML document.
  - **Example:**

    ```
    <style>
      p { color: blue; }
    </style>
    ```
- **External CSS**: CSS defined in a separate .css file and linked to the HTML document.

  Example:

  ```
  <link rel="stylesheet" href="styles.css">
  ```

CSS Selectors

Selectors are used to target HTML elements for styling. Examples include:

- **Element Selector**: p { color: red; } targets all <p> elements.
- **Class Selector**: .className { font-size: 14px; } targets elements with a specified class.
- **ID Selector**: #idName { background-color: blue; } targets an element with a specific ID.

Classes and IDs

- **Class**: Used to apply styles to multiple elements with the same class name.

  Example: <div class="box">Content</div>

- **ID**: Used to apply styles to a unique element on the page.

  Example: <div id="uniqueBox">Content</div>

Strings

Strings are a sequence of characters that are used to represent text. In web development, strings are crucial for tasks like storing user input, displaying messages, and manipulating content.

Pseudo Classes

Pseudo-classes in CSS are used to define the special state of an element. They allow developers to style an element when it is in a particular state, such as when a user hovers over a link or selects an option in a form.

Examples:

- :hover — Applies when the user hovers over an element.
- :active — Applies when the element is in an active state.
- :focus — Applies when an element has focus (e.g., a text input).

Introduction to JavaScript Coding

JavaScript is a client-side scripting language that enables interactive web pages. It allows developers to create dynamic effects, control multimedia, and handle user events on the page.

Finding Elements

JavaScript can be used to find elements in an HTML document using methods like getElementById, getElementsByClassName, and querySelector. These methods allow you to select HTML elements for further manipulation.

Example:

```
document.getElementById("myElement");
```

Interactive Pages

JavaScript makes it possible to create interactive web pages where the content can change dynamically based on user input or events. This includes responding to mouse clicks, keyboard actions, and other user-driven interactions.

Animation

JavaScript can also be used to create animations on web pages. Using libraries like CSS transitions or JavaScript libraries such as jQuery, developers can animate properties like position, size, and color of elements.

Example:

```
element.style.transition = "all 2s ease-in-out";
```

Introduction to jQuery

jQuery is a popular JavaScript library that simplifies tasks like DOM manipulation, event handling, and animation. It allows developers to write less code to achieve complex effects and functionalities.

Example:

```
$(document).ready(function(){
  $("button").click(function(){
    alert("Button clicked!");
  });
});
```

Introduction to PHP

PHP is a server-side scripting language primarily used for web development. It is often used for creating dynamic web pages and interacting with databases.

Data Types

PHP supports various data types, such as:

- Integer
- Float
- String
- Boolean
- Array
- Object

Variables

Variables are used to store data that can be changed during program execution. In PHP, variables start with a dollar sign ($).

Example:

$myVariable = "Hello, world!";

Constants

Constants are values that cannot be changed once they are defined. PHP constants are defined using the define() function.

Example:

define("MY_CONSTANT", 3.14159);

Operators

PHP operators are used to perform operations on variables and values. There are various types of operators in PHP, including:

- **Arithmetic Operators**: +, -, *, /, %
- **Comparison Operators**: ==, !=, <, >, <=, >=
- **Logical Operators**: &&, ||, !
- **Assignment Operators**: =, +=, -=, *=, /=

About PHP Operators and Expressions

Operators in PHP are used to perform operations on variables. Expressions are combinations of variables, values, and operators that are evaluated to produce a result.

What Is a String?

A string is a sequence of characters. In PHP, strings are enclosed in either double quotes (") or single quotes (').

Example:

$myString = "Hello, world!";

String Functions

PHP provides various functions for working with strings, such as:

- strlen() — Returns the length of a string.
- strtoupper() — Converts a string to uppercase.

- strtolower() — Converts a string to lowercase.
- substr() — Extracts a portion of a string.

Example:

```
$length = strlen("Hello, world!");
```

Other String Functions

PHP offers several other functions for manipulating strings:

- str_replace() — Replaces all occurrences of a substring with another substring.
- strpos() — Finds the position of the first occurrence of a substring.
- trim() — Removes whitespace from the beginning and end of a string.

Conditionals and Loops

Control Structures, Blocks, and Compound Statements

Control structures in PHP allow you to control the flow of your program based on certain conditions. Examples of control structures are if, else, elseif, switch.

Example:

```
if ($x > 0) {
  echo "Positive number!";
} else {
  echo "Non-positive number!";
}
```

Loops

Loops allow you to repeat a block of code multiple times. Common loop types in PHP are:

For Loop

Executes a block of code a specified number of times. Example:

```
for ($i = 0; $i < 10; $i++) {
  echo $i;
}
```

While Loop

Executes a block of code as long as a condition is true. Example:

```
while ($i < 10) {
  echo $i;
  $i++;
}
```

Do-While Loop

Similar to the while loop, but it guarantees at least one iteration. Example:

```
do {
  echo $i;
  $i++;
} while ($i < 10);
```

Foreach Loop

Iterates over arrays or objects. Example:

```
$array = ["apple", "banana", "cherry"];
foreach ($array as $fruit) {
  echo $fruit;
}
```

What Is an Array?

An **array** is a data structure that allows you to store multiple values in a single variable. Instead of using multiple variables to store similar data, you can use an array to store multiple values under one variable name. Arrays can store elements of any data type including numbers, strings, or even other arrays.

Example of an array:

```
$fruits = array("Apple", "Banana", "Cherry");
```

Arrays

Arrays in PHP can be either **indexed** or **associative**:

- **Indexed arrays**: Arrays with numerical indices starting from 0. Example:
- `$fruits = array("Apple", "Banana", "Cherry");`
- **Associative arrays**: Arrays where each element has a custom key (often strings). Example:

- $person = array("name" => "John", "age" => 25);

Modifying Arrays (Unsetting, Deleting, Adding, and Changing Elements)

PHP provides several functions for modifying arrays:

- **Unsetting/Deleting an Element**: Use the unset() function to delete an element from an array. Example:
- unset($fruits[1]); // Removes "Banana" from the array
- **Adding Elements**: You can add elements to an array using the array syntax or array functions like array_push(). Example:
- array_push($fruits, "Grape"); // Adds "Grape" to the end of the array
- **Changing Elements**: Simply reassign a value to an existing array index. Example:
- $fruits[1] = "Blueberry"; // Changes "Banana" to "Blueberry"

User-Defined Functions

Functions are blocks of code that can be reused throughout a program. They allow you to group code into a single entity, making your program more organized and modular.

What Is a Function?

A function is a named block of code that performs a specific task and can be called (executed) whenever needed.

Define a Function

To define a function in PHP, you use the function keyword followed by the function name and a set of parentheses. The code inside the curly braces {} will execute when the function is called.

Example:

```
function greet($name) {
   echo "Hello, $name!";
}
```

You can call the function like this:

```
greet("John"); // Outputs: Hello, John!
```

PHP Forms

Forms are a critical aspect of web development as they allow user input to be submitted to the server for processing.

Introduction

PHP forms allow users to send data to the server where it can be processed and stored. Forms are created using HTML, and PHP can handle form data to interact with databases, send emails, and perform other server-side tasks.

Review of HTML Forms

An HTML form is created using the <form> tag. It can include various input elements like text fields, radio buttons, checkboxes, and submit buttons.

Example of an HTML form:

```
<form method="POST" action="process.php">

  <input type="text" name="username" placeholder="Enter username">

  <input type="password" name="password" placeholder="Enter password">

  <input type="submit" value="Submit">

</form>
```

PHP and Forms

Once the form is submitted, PHP can access the form data through the superglobal arrays $_GET, $_POST, and $_REQUEST.

- $_GET: Retrieves data sent via the HTTP GET method (usually through the URL).
- $_POST: Retrieves data sent via the HTTP POST method (usually via form submission).
- $_REQUEST: A combination of $_GET, $_POST, and $_COOKIE.

GET, POST, REQUEST

- **GET**: Used for retrieving data. Data is visible in the URL. Example:
- $name = $_GET['name'];
- **POST**: Used for sending data securely. Data is not visible in the URL. Example:
- $name = $_POST['name'];
- **REQUEST**: Combines data from both GET and POST. Example:

- $name = $_REQUEST['name'];

Files and Directories

PHP provides functions for working with files and directories on the server. You can create, read, write, and delete files, as well as manage directories.

Files

The file system allows for storing and managing files on the server. Some of the functions used for file handling include:

- fopen(): Opens a file.
- fread(): Reads the contents of a file.
- fwrite(): Writes data to a file.
- fclose(): Closes a file.

The Web Server, PHP, and Permissions

Web servers are responsible for serving web pages and handling HTTP requests. PHP is used on the server-side to generate dynamic content. Permissions determine what actions PHP can perform on files and directories, such as reading or writing data.

Directories

A directory is a container used to organize files. PHP has built-in functions for directory management:

- opendir(): Opens a directory.
- readdir(): Reads the contents of a directory.
- closedir(): Closes a directory.

Managing Content with Include Files

Include files are used to insert content or PHP code into a PHP script. This is useful for reusing code or separating logic from presentation.

Example:

```
include 'header.php'; // Includes the contents of 'header.php'
```

Regular Expressions and Pattern Matching

Regular expressions (regex) are used to find specific patterns in text. They can be used for tasks like validating input, searching for substrings, and performing text replacements.

What Is a Regular Expression?

A regular expression is a sequence of characters that forms a search pattern. It is used to match strings or portions of strings according to specific patterns.

Example:

```
$pattern = "/^abc/";

$string = "abcdef";

if (preg_match($pattern, $string)) {

   echo "Match found!";

}
```

Pattern-Matching Functions

PHP offers several functions for working with regular expressions:

- preg_match(): Checks if a pattern matches a string.
- preg_replace(): Replaces text that matches a pattern with a replacement string.
- preg_split(): Splits a string by a regular expression pattern.

Example:

```
$text = "Hello 123 World";

$pattern = "/\d+/"; // Match digits

preg_match($pattern, $text, $matches);

print_r($matches); // Outputs: Array ( [0] => 123 )
```

Introduction to MySQL

MySQL is an open-source relational database management system (RDBMS) that is widely used in web development. It allows for the creation, management, and manipulation of databases and is typically used alongside PHP for web applications.

About Databases

A **database** is an organized collection of data that can be easily accessed, managed, and updated. It is used to store and retrieve large amounts of data, and it ensures that data is stored in a structured way.

Databases are typically managed by database management systems (DBMS) like MySQL, PostgreSQL, Oracle, and SQL Server.

The Anatomy of a Relational Database

A **relational database** organizes data into tables, which are composed of rows and columns. Each table represents a specific entity (like "Users" or "Orders"), and each row contains information about an instance of that entity. Tables can be related to each other using keys (primary keys and foreign keys).

Connecting to the Database

To connect to a MySQL database, you need to use a **database connection**. In PHP, this is typically done using the mysqli or PDO (PHP Data Objects) extension.

Example:

```
$connection = mysqli_connect("localhost", "username", "password", "database_name");


if (!$connection) {
  die("Connection failed: " . mysqli_connect_error());
}
```

The MySQL Privilege System

The MySQL privilege system controls who can access the database and what actions they can perform. Privileges are granted to database users and can define whether a user can read, write, or modify data. The system is highly flexible and allows for fine-grained access control.

Common privileges include:

- SELECT: Allows a user to read data.

- INSERT: Allows a user to add data.
- UPDATE: Allows a user to modify existing data.
- DELETE: Allows a user to delete data.

What Is SQL?

SQL (Structured Query Language) is a standard language used to manage and manipulate databases. SQL allows you to perform operations such as querying data, updating records, and creating or modifying tables and databases.

SQL Data Manipulation Language (DML)

SQL Data Manipulation Language (DML) refers to the SQL commands used to manage data in a database. The most common DML commands include:

- **SELECT**: Retrieves data from one or more tables. Example:
- SELECT * FROM users;
- **INSERT**: Adds new data into a table. Example:
- INSERT INTO users (name, email) VALUES ('John', 'john@example.com');
- **UPDATE**: Modifies existing data in a table. Example:
- UPDATE users SET email = 'john.doe@example.com' WHERE name = 'John';
- **DELETE**: Deletes data from a table. Example:
- DELETE FROM users WHERE name = 'John';

SQL Data Definition Language

SQL Data Definition Language (DDL) refers to the SQL commands used to define and manage database structures, such as tables, views, and schemas. Some common DDL commands are:

- **CREATE**: Creates a new table, database, or view. Example:
- CREATE TABLE users (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(255), email VARCHAR(255));
- **ALTER**: Modifies an existing table. Example:
- ALTER TABLE users ADD COLUMN age INT;
- **DROP**: Deletes a table or database. Example:
- DROP TABLE users;

SQL Functions

SQL functions are built-in operations that can be used to perform various calculations or operations on data. Some common SQL functions include:

- **COUNT()**: Counts the number of rows. Example:
- SELECT COUNT(*) FROM users;

- **SUM()**: Adds up the values in a column. Example:
- SELECT SUM(age) FROM users;
- **AVG()**: Calculates the average value of a column. Example:
- SELECT AVG(age) FROM users;
- **MAX()** and **MIN()**: Retrieves the maximum or minimum value of a column. Example:
- SELECT MAX(age) FROM users;

PHP and MySQL Integration

PHP is often used in combination with MySQL to create dynamic web applications. PHP can connect to a MySQL database to retrieve, manipulate, and display data. Using PHP with MySQL allows developers to build data-driven websites like content management systems, e-commerce platforms, and social networks.

Example of connecting PHP to MySQL:

```
$connection = mysqli_connect("localhost", "username", "password", "database_name");


if (!$connection) {
   die("Connection failed: " . mysqli_connect_error());
}


$result = mysqli_query($connection, "SELECT * FROM users");


while ($row = mysqli_fetch_assoc($result)) {
   echo $row['name'] . " - " . $row['email'] . "<br>";
}


mysqli_close($connection);
```

Cookies and Sessions

What Is Stateless?

A **stateless** application does not store any information about the previous interactions of the user. Every request is treated as independent. HTTP, the protocol used for web communication, is inherently stateless, meaning that the server does not retain any memory of previous requests.

What Are Cookies?

Cookies are small pieces of data that are stored in the user's browser. They can be used to remember user preferences, login information, and track user behavior across sessions.

In PHP, cookies can be set using the setcookie() function:

setcookie("user", "John", time() + 3600); // Cookie expires in 1 hour

---

PHP and Cookies

PHP provides functions to work with cookies. You can set, retrieve, and delete cookies in PHP.

- **Setting a cookie**:
- setcookie("username", "John", time() + 3600);
- **Accessing a cookie**:
- echo $_COOKIE["username"];
- **Deleting a cookie**:
- setcookie("username", "", time() - 3600); // Expire the cookie

---

What Is a Session?

A **session** is a way to store user data on the server across multiple requests. Sessions are more secure than cookies because they are stored on the server, not the client's browser. PHP uses the $_SESSION superglobal to store session data.

- **Starting a session**:
- session_start();
- **Storing session data**:
- $_SESSION["user"] = "John";
- **Retrieving session data**:
- echo $_SESSION["user"];
- **Destroying a session**:
- session_destroy();

PHP and E-Mail

The Mail Server

A **mail server** is a software that sends, receives, and stores emails. It uses the **SMTP** (Simple Mail Transfer Protocol) for sending emails and **POP3** (Post Office Protocol 3) or **IMAP** (Internet Message Access Protocol) for receiving them.

MIME (Multipurpose Internet Mail Extensions)

MIME is an extension to the original email protocol that allows emails to contain multimedia content such as images, audio, and video, as well as text in character sets other than ASCII. It also allows for the transmission of email attachments.

Runtime Configuration Options

In PHP, certain configurations can be adjusted to enhance the behavior of the script, such as configuring the mail sending options. These configurations can be found in the **php.ini** file or set dynamically within the PHP script using functions like ini_set().

The mail() Function

The mail() function in PHP is used to send email messages. It requires at least three arguments: the recipient's email address, the subject of the email, and the message body.

Example:

mail("recipient@example.com", "Subject", "Message body");

You can also specify additional headers like the sender's email address and MIME type for HTML emails.

Sending a Simple E-Mail Message

Example of sending a simple email:

```
<?php
$to = "recipient@example.com";
$subject = "Test Email";
$message = "Hello, this is a test email.";
$headers = "From: sender@example.com";

mail($to, $subject, $message, $headers);
?>
```

Example: Sending an HTML Message

To send an HTML email, you need to set the correct content type in the email headers.

Example:

```
<?php
$to = "recipient@example.com";
$subject = "HTML Email Test";
$message = "<html><body><h1>This is a test email</h1><p>Hello, this is an HTML email.</p></body></html>";
$headers = "From: sender@example.com\r\n";
$headers .= "Content-type: text/html\r\n";

mail($to, $subject, $message, $headers);
?>
```

PHP and Date/Time

Formatting Dates and Times

In PHP, you can use the date() function to format the date and time in various formats. The date() function takes a string that specifies the format.

Example:

```
echo date("Y-m-d H:i:s"); // Outputs: 2024-12-04 14:30:00
```

Getting the Timestamp

You can get the current Unix timestamp (the number of seconds since January 1, 1970) using the time() function.

Example:

```
echo time(); // Outputs the current timestamp
```

Security and Debugging

About Security

Security is a crucial aspect of any web application. Common security concerns in PHP include:

- SQL Injection: Always use prepared statements and parameterized queries to prevent SQL injection.

- Cross-Site Scripting (XSS): Sanitize user inputs to prevent malicious scripts from executing in the browser.
- Cross-Site Request Forgery (CSRF): Use anti-CSRF tokens to prevent unauthorized requests from being made on behalf of the user.

Securing PHP and MySQL

To secure PHP and MySQL:

- Use secure password hashing (e.g., bcrypt) for user passwords.
- Enable prepared statements in MySQL queries to avoid SQL injection attacks.
- Ensure your PHP version and MySQL version are up-to-date with security patches.

Debugging

PHP offers several tools to help with debugging, including:

- **Error Reporting**: Set error_reporting(E_ALL); and ini_set('display_errors', 1); to display all errors during development.
- **Logging**: Errors can also be logged to a file instead of being displayed on the screen using ini_set('log_errors', 1); and ini_set('error_log', '/path/to/error.log');.
- **Xdebug**: A PHP extension for powerful debugging and profiling.

Asynchronous JavaScript and XML (AJAX)

AJAX Introduction

AJAX (Asynchronous JavaScript and XML) is a technique used to send and receive data asynchronously without refreshing the web page. This allows for faster, more interactive web applications.

AJAX with PHP

AJAX is commonly used with PHP to create dynamic, real-time interactions. JavaScript handles the request and response asynchronously while PHP processes the request on the server.

Example of an AJAX request in JavaScript:

```
var xhr = new XMLHttpRequest();

xhr.open("GET", "server.php?name=John", true);

xhr.onreadystatechange = function() {

 if (xhr.readyState == 4 && xhr.status == 200) {

  document.getElementById("response").innerHTML = xhr.responseText;

 }
};
xhr.send();
```

In server.php, PHP processes the request and sends a response:

```
<?php
$name = $_GET['name'];
echo "Hello, " . $name;
?>
```

AJAX with Database

AJAX can also be used to retrieve data from a database without refreshing the page. This is useful for applications such as real-time chat systems, live search suggestions, or updating data dynamically.

Content Management System (CMS): WordPress and Bootstrap

Introduction to CMS

A **Content Management System (CMS)** is a software application used to manage digital content. WordPress is one of the most popular CMS platforms, known for its ease of use, flexibility, and extensibility.

Installation of CMS

To install WordPress, follow these steps:

1. Download the WordPress package from the official website.
2. Upload the files to your web server.
3. Create a MySQL database and user for WordPress.
4. Configure the wp-config.php file with the database details.
5. Run the WordPress installation script by visiting your site.

Create a Website with CMS

Once WordPress is installed, you can create a website by choosing a theme, installing plugins, and adding content such as posts, pages, and media.

Change and Install the CMS Template

WordPress comes with several themes. You can change or install a new theme by navigating to the **Appearance** section in the admin dashboard and selecting a new theme.

CMS Extensions

WordPress offers a wide range of extensions (called **plugins**) that add functionality to your website. These plugins can be used for SEO, security, performance optimization, and more.

Upgrading of CMS

Upgrading WordPress to the latest version is important for security and performance. You can upgrade WordPress via the admin dashboard or by manually downloading the latest version and replacing the existing files.

CMS Backups

Regular backups of your CMS are essential to prevent data loss. WordPress offers several plugins for automated backups, or you can manually back up your database and files.

Subject Work Code

The code for this subject is available on GitHub. You can access and review the code repository using the following link:

GitHub Repository: Web Engineering

Please refer to this link for all related code and further development details.

The End
