UNLIMITED POWERFUL LEARNING

UPL PROSPECTOUS

early break even | best experience

2023

@unlimited powerfull learning

ABOUT COMPANY:

UNLIMITED POWER FULL LEARNING (UPL) aims to solve the challenges and minimize the gap between students with IT industries' expectations. This organization is built by a strong team who are having good academic and industry experience of more than two decades. The founder of this experience G.D. Mallikarjuna has 20+ plus started as a technologist having diverse experience in the education sector as Trainer and Developer.

VISION:

At UPL@SNIPE, we make the best experience in technology learning with career guidance for their life journey

MISSION:

Learn with Live experience and career values.

PROGRAMS OFFERED:

PROGRAMS	DURATION	AMOUNT + GST
CODING BOOT CAMP	4 TO 6 MONTHS	Rs.30000/-
CERTIFICATION COURSE	3 SEMESTERS 1 YEAR COURSE	Rs. 25000/- per semester Rs. 10000/- final semester
CAREER BRIDGE	3 MONTHS	Rs. 50000/-
INDUSTRY READINESS PROGRAM	3 MONTHS	Rs.20000/-

*Terms and Conditions are subjected to further discussion before signing the legal agreement

UPL
UNLIMITED POWER FULL LEARNING

CODING BOOT CAMP

ABOUT THIS MODEL

- **Category:** Virtual Program
- **Target Audience:** Fresher & Experienced
- **Duration:** 4 To 6 Months
- Cost: Rs. 30,000/Candidate (**Registration:** 10K + GST **After 6 Weeks:** 10K + GST **Live Project:** 10K + GST)
- **Course Coverage:** 2 Months training in a relevant discipline, 1 capstone project & followed by involving in live project for duration 4 months.
- **Outcome:** Build their careers feature strong growth projections & lucrative salaries
- **Career Opportunities:** The best jobs you can secure after completing one of these programs such as, Technical Support Specialist, Digital Marketer, Junior Developer, Data Analyst, Web Developer, Project Manager, User Interface/Xxperience (UI/UX) designer, Application Developer, Product Manager, Software Engineer, Full Stack Developer, Data Scientist, Development Operations (DevOps) Engineer, Back End Eeveloper, Teach Others, also Freelancer

COURSES ARE :

- JAVA FULLSTACK
- FULL STACK C# .NET
- FRONT END DEVELOPER IN (REACT/ANGULAR)
- MEAN STACK
- PYTHON
- DATA-SCIENCE
- AUTOMATION TESTING WITH JAVA
- UI/UX DESIGN
- DIGITAL MARKETING
- JENKINS
- MACHINE LEARNING
- DATASTRUCTURE IN PYTHON
- TABLEOU
- POWER BI
- PSPARK
- DEVOPS

*Terms and Conditions are subjected to further discussion before signing the legal agreement

UPL
UNLIMITED POWER FULL LEARNING

BENEFITS IN THIS PROGRAM :

- Uplsnipe Coding Bootcamp Certificate.
- Program Transcript For The Entire Learning Path.
- Coding Bootcamps Can Open Doors To Exciting Technical Career Opportunities.
- Mastering Programming Languages And Associated Technologies Can Prepare You To Work As A Software Or Web Developer.
- Strong Growth Projections And Lucrative Salaries

JAVA FULL-STACK

CORE JAVA

UNIT_001 : 03 HRS

The Project, Domain, Platform, Product, Technology, Java Tech Stack, Process, Development Environment, Java Introduct

UNIT_002 : 03 HRS

Java Fundamentals, Tokens, Conditional Statements, Looping Constructs

UNIT_003 : 05 HRS

Java Fundamentals, Tokens, Conditional Statements, Looping Constructs

UNIT_004: 05 HRS

Arrays, Strings, Exception Handling-Try, Catch, Finally, Custom Exception, Checked And Unchecked Exception

*Terms and Conditions are subjected to further discussion before signing the legal agreement

UPL
UNLIMITED POWER FULL LEARNING

UNIT_005 :

08 HRS

Multithreading, Life Cycle, Synchronized Block, And Method, Wait, Notify, notify all, Concurrency -ThreadPool, ThreadPool Types, FixedThreadPool, CachedThreadPool, ScheduledThreadPool, SingleThreadExecutor, Threadpool size,ThreadLocal, Lock

UNIT_006 :

08 HRS

IOStreams, Inputstream, Outputstreams, Character Vs Byte Streams, Node And Filter Streams, Collection, Collection Framework, Collection Interface, Bulk Operations , Internals Of List, Set And Map, Arraylist, Hashset, Hashmap AndImportant Concepts Of Implementations Of Collection Frameworks

UNIT_007 :

016 HRS

Annotations -Build-In Annotations, Custom Annotations, Market, Singlevalue And Multivalue Annotations, Regular Expressions

UNIT_008 :

03 HRS

JAVA 1.8 and 11 features, LAMBDA Expressions, Functional Interface, Streams API, JDBC Programming.

LAB SET CORE JAVA

LAB 1 :

- learn to compile and run a very simple Java program
- To know how to use environment
- Learn to use scanner class and take user input
  - (a) WAP print “Welcome to Java Programming” in console
  - (b) WAP print “addition of two numbers”
  - (c) WAP to generate Fibonacci Series
  - (d) WAP to generate Prime Number generation
  - (e) WAP to convert given Celsius to Fahrenheit

LAB 2 :

-learn to compile and run a very simple Java program
- To know how to use environment
- To learn basic principles of Object , Class
- To learn default constructor, parametrized constructor, POJO (Encapsulation)
  - (a)WAP display employee information using class and object(default constructor)
  - (b)WAP display employee information using class and object(parameterized constructor)
  - (c)WAP display employee information using class and object(POJO-setter/getter)
  - (d) WAP to generate Prime Number using object and class (constructor)
  - (e)WAP to build simple calculator(class , object and constructor)

LAB 3 :

- learn to compile and run a very simple Java program
- learn to Arrays

(a) WAP to find the sum of ‘n’ Numbers
(b) WAP to Sort ‘n’ Numbers
(c) WAP to Addition, Subtraction and Multiplication of two matrices

LAB 4 :

- learn to compile and run a very simple Java program
- To learn Inhertance and Polymorphism concepts
  - (a)WAP display Dog information reusing Animal(Single Inheritance)
  - (b) WAP to build Parttime employee and full-time employees from the derived class Employee and do necessary changes and implement the same(Hierarchical Inheritance)
  - (c) WAP to build FullTime employee from derived Employee class which is inherithe ted from Person to display Employee information(Multilevel Inheritance)
  - (d) WAP to demonstrate access modifier default, private, protected and public modifier
  - (e) WAP to demonstrate display method of person class as abstract extending Employee class and display the same (abstraction)
  - (f) WAP to build a calculator to display the Addition, Subtraction, Multiplication and Division calculator using interface Calculator and CalculatorImpl(Interface and class implementations)
  - (g) WAP to demonstrate to display compute operation using method overloading (Polymorphism)
  - (h) WAP to demonstrate to override addition, subtraction, division and multiplication method in subclass of calculator (Runtime Polymorphism, interface)
  - (f) WAP to demonstrate abstract class using person class extending the Employee concreate class to display information(abstract class)

LAB 5 :

- learn to use of MATH library
- learn passing command line arguments
  - Demonstrate an example for Constants, Truncating, comparison, power , Trigonometric ,square roots and generate random number using Math library

LAB 6 :

- learn to compile and run a very simple Java program
- learn to use String, StringBuffer, and StringBuilder
  - (a) Demonstrate an example extract substring from a string
  - (b) WAP to parse String using String Tokenizer
  - (c) WAP to reverse a String (By character)
  - (d) WAP to reverse a String (By words)
  - (e) Demonstrate an example StringBuffer and String and StringBuilder
  - (f) Demonstrate difference between == and equals operator on String
  - (g) WAP to convert cases of given String
  - (h) WAP to sort given 'N' Strings
  - (i) WAP to the concatenation of String
  - (j) WAP to string comparison

LAB 7 :

- learn the basic jargon of object-oriented programming and how it appears in code
- learn how a Java program is organized into multiple source files
- learn to compile and run a very simple Java program
  - (a) Create an application program that consists of 2 classes, a "startup class" and a second class that prints out the values of at least 3 instance variables that are initialized in one method and printed from another. The initialization method should have two forms. One of them will have no arguments and the other 3 arguments corresponding to the data types of the 3 instance variables. The second class should also contain a "class variable" of one of the 8 primitive data types (you choose). Create 2 instances of the second class. Using one instance, set the class variable to some value and, using the second instance, print that value out. Also, using either or both of the 2 instances, call the methods that set and print the 3 instance variables.
  - (b) Demonstrate a Bank Application
  - Demonstrate 1.Create an Account 2. Deposit 3. Withdraw and 3. Display the account details

LAB 8 :

- learn to use inheritance, Exception Handling, Packaging
- learn to design your own hierarchy
- learn how to write and use a constructor method
- learn how to use access specifiers (public, protected, private, and default or package)

- (a) Write an application with a hierarchy composed of at least 2 classes. One of these classes will be a subclass derived from the other. The subclass is to have at least 3 constructor methods. Each of these will be called during the creation of 3 subclass objects. In the example shown, the subclass student has a default constructor, a constructor with one string argument, and a constructor with a string argument giving the student's name and an integer argument giving an ID number.
- (b) Create an hierarchy of Person
- (c) Assume necessary hierarchy and demonstrate computation of Student Result generation/Marks sheet generation.
- (d) WAP to create custom exception handling.
- (e) WAP Develop a Calculator using layered architecture with appropriate exception handling

LAB 9 :

- learn to use I/O Streams

- (a) WAP to demonstrate Reading/Writing to file using FileInputStream/OutputStream and Reader/Writer classes
- (b) WAP to reverse file content
- (c) WAP to concatenate two files
- (d) WAP to copy file to another
- (e) WAP to copy all files from src directory to destination directory
- (f) WAP to demonstrate create, Delete and modify a file.
- (g). WAP Reading and writing the data using DataInput and DataOutput Streams.
- (h). WAP Reading and writing using File Input and File output Streams
- (I). WAP Reading and writing a files using FileReader and FileWriter classes

LAB 10 :

- learn to use Multithreading
  - (a) WAP to demonstrate an example for Thread extending class and Implementing Runnable Interface
  - (b) WAP to demonstrate thread life cycle
  - (c) WAP to demonstrate thread priority
  - (d) WAP to demonstrate different types of Thread Pool
  - (e) WAP to demonstrate Thread call/future

LAB 11 :

- learn to use Collection class
  - (a) WAP to demonstrate an ArrayList
  - (b) WAP to demonstrate Hashtable and Hash Map
  - (c) WAP to implementation Stack and Queue
  - (d) WAP to demonstrate Tree Set and Sorted Set
  - (e) Demonstrate examples with ArrayList
  - (f). Demonstrate the Linked List
  - (g). Demonstrate the Vector
  - (h). Demonstrate the stack implementations
  - (i). Demonstrate the collections utility class
  - (j). Demonstrate the Sort by comparator
  - (k). Demonstrate Deque operations
  - (l). Demonstration of Map interface using HashMap
  - (m). Demonstarate Map using HashTable
  - (n). Demonstrate the ENUMERATIONS

LAB 12 :

- learn to use Java Applet/Swing/AWT
  - (a) WAP to creation of GUI and by assuming necessary steps for student marks sheet generation
  - (b) WAP to Copy file from one directory to another
  - (c) WAP to develop a simple calculator.

LAB 13 :

- learn to use Java Networking

- **Write a chat application:**
- **One-One:** By opening socket connection and displaying what is written by one party to the other.
- **Many-Many (Broad cast) :** Each client opens a socket connection to that chat server and writes to the socket. What ever is written by one party can be seen by all other parties.

LAB 14 :

-learn to use lambda expressions and Streams API

- (a).WAP to demonstrate simple with and without lambda expression
- (b).Display lambda expression with one argument
- (c).Perform addition of two input numbers showcase two arguments
- (d).Perform showcase default and static keywords usage in functional interface
- (e).Display the square of number forms of return statement
- (f).Demonstrate the multithreading example using lambda expression

LAB 15:

- learn to use Annotations

- (a). Demonstrate @Override annotations
- (b). Demonstrate the suppress warning annotations
- (c). Demonstrate the deprecated annotations
- (d). write a example to demonstrate functional interface
- (e). Write a custom annotations to show Course details

LAB 16:

- learn to use JDBC Programming

(a) WAP to build calculator operations such create, add, mod and delete operations

(b) WAP to build simple Employee database management system

LAB 17 :

- CAPSTONE PROJECT – Use Core Java and JDBC

(a) Develop and Design the HR Department which maintains the employee details with basic payroll information. Also Generate necessary report information.

It may be having the following tables

Employee {employee identification number, first and last name, designation, home address, contact number, hire date, work location, and such details.}

Address { like building ID, company's physical locations, zip code, address, name of the manager, etc., for each physical location etc}

Payroll {employee Id, payroll info-Basic, HRA, DA,TA}

Department{ DeptId, dname, Address}

Whenever a new hire takes place, data is added to relevant records on the relevant tables like payroll, employees table, department table etc. The HR department creates a new record and updates it to reflect the changes. When the enterprise needs to send a letter, it simply reads the employee's table to select the relevant personal details, or when the employee leaves their service.

Expected features

- Persisting employee, department, and Payroll information
- Modify the employee, department, and payroll information
- Delete employee, department
- Display the all employee works for a certain department
- Listof all departments

Note:

Maintain the layered architecture

User interface will be mocked Mock Controller, with services and dao

Dao layer should be using JDBC to connect mysql database

DATABASE PROGRAMMING

UNIT_009 :

03 HRS

Database, DBMS, RDBMS,SQL, DDL,DML,DCL, JOINS, MYSQL, POSTGRESQL.

FOUNDATION WEB-PROGRAMMING

UNIT_010 : 03 HRS

Introduction, Webserver, Servlet, Jobs of Servlet, Life cycle, SingleThreadModel, Context and Config, HTTPStatus codes, Session Handling-cookies, hiddenform fields, URL Rewritting and SessionTracking API, forward and redirect

UNIT_011 : 03 HRS

JSP, JSP PROCESS, JSP Features , Declarations, Scriptlets And Expressions, Standard Actions, Implicit Objects, Custom Tag Libraries, Forward, Redirect

LAB SET SERVLET

LAB 1 :

Write a simple Servlet program to display welcome message.

LAB 2 :

- Write a program to change the background color selected by the user.

LAB 3 :

- Write a Servlet program to display “Hello World” in HTML/Text format.

LAB 4 :

Write a servlet program to display current date in server.

LAB 5 :

- Write a servlet program to display information about server.

LAB 6 :

Write servlet program to read init parameter values.

LAB 7 :

- Write a servlet program to submit information from html to servlet and display the message.

LAB 8 :

Write a servlet program to pass multiple values.

LAB 9 :

Write a Servlet program to demonstrate sendRedirect loginForSendRedirect.jsp

LAB 10 :

- (a)Write a Servlet program to display session.
- (b)To develop a simple program to get session elements.

LAB 11 :

- (a) Write a program how to send cookie in servlets.
- (b)Write a program to read cookies from Servlets.

LAB 12 :

Write a program to demonstrate URL-Rewriting.

LAB 13 :

- (a)Write a program to demonstrate Hidden Form Fields.
- (b) To demonstrate a servlet program to show Multiple Hidden formFileds.

LAB 14 :

Write a servlet program using RequestDispathcher to forward the request to an Error page.

LAB 15 :

Write a servlet program to submit information from html to servlet and connection to the database.

LAB 16 :

Write a servlet program for handling database connections using SingleThreadModel.

LAB 17 :

To develop a mini project using servlet (ATM BANK).

LAB SET JSP

LAB 1 : INTRODUCTION TO JSP

1. Set up the development environment with a servlet container (e.g., Apache Tomcat).
2. Create a simple JSP file that displays "Hello, JSP!" in the browser.
3. Execute the JSP file and verify the output.

LAB 2 : JSP BASICS

- Create a JSP page that includes an HTML form.
- Retrieve form data using request parameters in JSP.
- Display the submitted form data on the JSP page.

LAB 3 : JSP SCRIPTING ELEMENTS

- Use scriptlets to write Java code within a JSP page.
- Retrieve request parameters and perform server-side processing using scriptlets.
- Display the processed data on the JSP page.

LAB 4 : JSP EXPRESSION LANGUAGE (EL)

1. Use EL expressions to access and display variables and attributes in a JSP page.
2. Perform arithmetic calculations and conditional statements using EL.
3. Use EL to access request parameters and display their values.

LAB 5 : JSP STANDARD ACTIONS

- Use JSP standard actions like <jsp:include> and <jsp:forward> to include and forward JSP pages.
- Use <jsp:useBean> to create and use JavaBeans in JSP.
- Use <jsp:setProperty> and <jsp:getProperty> to set and get JavaBean properties.

LAB 6 : JSP CUSTOM TAGS

1. Create a custom tag handler class and define a custom tag in a JSP.
2. Implement custom tag attributes and body content in the tag handler class.
3. Use the custom tag in a JSP page and observe its behavior.

LAB 7 : JSP AND JDBC

- Set up a database connection using JDBC in a JSP page.
- Retrieve data from a database using JDBC queries in a JSP page.
- Display the retrieved data on the JSP page.

LAB 8 : JSP AND SESSION MANAGEMENT

- Use session attributes to store and retrieve user-specific data in a JSP page.
- Implement login/logout functionality using session management in JSP.
- Display user-specific data based on session attributes in the JSP page.

LAB 9 : JSP AND ERROR HANDLING

1. Handle exceptions and errors in JSP using the <%@ page errorPage="..." %> directive.
2. Create an error handling JSP page to display custom error messages.
3. Test the error handling mechanism by deliberately throwing exceptions in a JSP page.

LAB 10 : JSP AND INTERNATIONALIZATION

- Implement internationalization in JSP using resource bundles and <fmt:message> tag.
- Create multiple property files for different languages and configure them in the JSP.
- Display localized messages and labels in the JSP based on the user's preferred language.

BUILD TOOLS, PROCESS, VERSION MANAGEMENT

UNIT_012 : 08 HRS

Maven, Features, Life Cycle, pom.xml, Agile, scrum process, source code management (SCM), GIT, GITHUB, GIT Commands

HIBERNATE

UNIT_013 : 05 HRS

Introduction, Environment setup, Architecture, Annotations, Reading data from DB, Mapping-one-to-one,one-to-many,many-to-manyInheritance, HQL,
Criteria API, Caching

LAB SET ORM HIBERNATE

LAB 1 :

1. Write a program to save user details username, password in the mysql using simple hibernate object
2. Write a program to save user details username, password in the mysql using simple hibernate object without using hibernate.cfg.xml

LAB 2 :

Write a program the user defined table name and column name.

LAB 3 :

Write a program to use with annotations Id, table, column, Generated Value, Transient and Lob in user table.

LAB 4 :

Write a program to use ValueType and Embedded object using nested classes scenario such as employee works for department

LAB 5 :

Write a program to use AttributeOverride and Embedded object keys using nested classes scenario such as employee works for department located in multiple address

LAB 6 :

Write a program to showcase association company linking with multiple address

LAB 7 :

Write a program to showcase fetch proxy object, Eager and Lazy object

LAB 8 :

Write a program association from employee to department one -to-one mapping

LAB 9 :

Write a program association from one employee works for more than one department showcase one -to-many mapping

LAB 10 :

- Write a program association from many to many using employee and dept relationship

LAB 11 :

Write a program inherit from single table person_info to employee and student(default inheritance)

LAB 12 :

- Write a program polymorphic inheritance from base class person to employee and student as concrete implementations using MappedSuperClass annotations

LAB 13 :

Write a program polymorphic inheritance from base class person to employee and student as concrete implementations using SingleTableStrategy annotations

LAB 14 :

- Write a program polymorphic inheritance from base class person to employee and student as concrete implementations using Table Per class Strategy annotations

LAB 15 :

Write a program polymorphic inheritance from base class person to employee and student as concrete implementations using JOIN Strategy annotations

LAB 16 :

Write a program to develop simple crud operation

LAB 17 :

Write a program to showcase Transient, Persistent and Detached Object

LAB 18 :

Write a program to demonstrate to load user details using HQL

LAB 19 :

Write a program to showcase the pagination,update and delete

LAB 20 :

Write a program to showcase the Named Query

LAB 21 :

Write a program to demonstrate Criteria API

LAB 22 :

Write a program to demonstrate 2ND level cache in hibernate

SPRING FRAMEWORK

UNIT_014 :

03 HRS

INTRODUCTION, Spring Benefits, Inversion Of Control (IOC) and Dependency Injection, Advantages of Spring Framework, Spring Modules, IoC Container, Spring AOP, Spring Jdbc Template, Spring with ORM Frameworks, Hibernate and Spring Integration, Spring MVC, Spring boot, Components, Spring Data JPA

LAB SET SPRING AND SPRING BOOT

LAB 1 :

Write a application program to instantiate bean object using bean factory

LAB 2 :

Write a application program to instantiate bean object using application context.

LAB 3 :

Write application of bean objects from context by setting the attributes

LAB 4 :

Write application constructor injection to bean object indexing, type in spring.xml as an argument to build employee information

LAB 5 :

Write a application program to instantiate bean object using bean factory

LAB 6 :

Write a application program to instantiate bean object using application context.

LAB 7 :

Write application of bean objects from context by setting the attributes

LAB 8 :

Write application constructor injection to bean object indexing, type in spring.xml as an argument to build employee information

LAB 9 :

Write application injecting object build the employee details

LAB 10 :

Demonstrate with employee details using inner bean, alias and idref

LAB 11 :

Demonstrate with company details with initialization as collections

LAB 12 :

Demonstrate the autowiring of bean objects of employee by name, type and constructor

LAB 13 :

Demonstrate bean scopes singleton, prototype using employee object

LAB 14 :

WAP to implement ApplicationContextAwar and BeanNameAware

LAB 15 :

WAP to implement ApplicationContextAwar and BeanNameAware

LAB 16 :

WAP to demonstrate call back life cycle bean methods

LAB 17:

WAP reading from properites file using PropertyPlaceholder

FRONT END TECHNOLOGIES

UNIT_015 : HTML

08 HRS

INTRODUCTION,History,Text-editor,Webbrowser, Html page structure, Html document format, Basic tags list, Html elements and tags,, Types of tags, Html attributes

UNIT_016 : CSS

03 HRS

Introduction, Selector, Style, Background, Border, Display, CSS Float, Font, Text, Button, Table, Form

UNIT_017 : BOOTSTRAP

03 HRS

History,Why bootstrap,Getting started,containers,Grid basic,Typography Colours,Tables,Images,Alerts,Buttons,Progressbar,Spinner,Pagination,List groups,Cards,Dropdowns,Collapse,Navs,Carousel,Modal,Tooltip,Popovers, Toast,Scrollspy,Offcanvas,Utilities,Bootstrap 5 forms

UNIT_018 : JAVA SCRIPT

05 HRS

Introduction,History,Features,Browser,Browser Object Model (BOM),How does 03 Hrs Java script works?,Document Object Model (DOM) ,Data type, undefined, falsy values, conditional and control statements,with statement,looping constructs,string, Arrays,window, forms, events,DHTML

UNIT_019 : JQUERY

03 HRS

JQUERY HOW TO USE JQUERY?, Merits,methods,Introduction,History,What Is Typescript,Why We Use,Features,Ts & Ecmasccript,Components Of Ts,Declaration Files,Environment Setup,Basic Syntax,Compile & Exe ,Example,Classes,Modules

LAB SET FOR HTML ,CSS,BOOTSTRAP,AND JAVA SCRIPT

LAB 1 : HTML BASICS

- Create an HTML page with a header, footer, and navigation menu.
- Add different types of HTML elements such as headings, paragraphs, lists, and images.
- Create a simple form with input fields and a submit button.

LAB 2 : CSS STYLING

- Style the HTML elements using inline CSS, internal CSS, and external CSS.
- Apply different font styles, colors, and backgrounds to elements.
- Create a CSS-based layout for a web page using divs and CSS properties like float and margin.

LAB 3 : RESPONSIVE DESIGN WITH BOOTSTRAP

- Integrate Bootstrap into an HTML page using the Bootstrap CDN.
- Use Bootstrap classes to create responsive layouts and grids.
- Apply Bootstrap components such as navigation bars, buttons, and forms.

LAB 4 : CSS FLEXBOX AND GRID

- Use CSS Flexbox to create a flexible layout for a web page.
- Create a grid layout using CSS Grid to arrange elements in a structured manner.
- Combine Flexbox and Grid to create complex layouts.

LAB 5 : JAVASCRIPT BASICS

- Write JavaScript code to display a popup message or alert on a web page.
- Create a JavaScript function to validate a form before submitting it.
- Manipulate HTML elements dynamically using JavaScript.

LAB 6 : DOM MANIPULATION

- Use JavaScript to access and manipulate the Document Object Model (DOM) of a web page.
- Create interactive features like event listeners and handlers using JavaScript.
- Build a simple slideshow or image carousel using JavaScript.

LAB 7 : FORM VALIDATION WITH JAVASCRIPT

- Write JavaScript code to validate form input fields, such as checking for empty fields or valid email addresses.
- Display error messages dynamically based on user input using JavaScript.
- Implement client-side form validation before submitting data to the server.

LAB 8 : HANDLING EVENTS WITH JAVASCRIPT

- Write JavaScript code to handle different types of events, such as button clicks or mouse movements.
- Implement functionality to show and hide elements based on user actions using JavaScript event handlers.
- Use JavaScript to create interactive elements like dropdown menus or to

LAB 9 : AJAX AND API INTEGRATION

- Use JavaScript and AJAX to make asynchronous requests to a server and update web page content dynamically.
- Integrate an external API into a web page to fetch and display data using JavaScript.
- Implement functionality to search and filter data on a web page using AJAX and JavaScript.

LAB 10 : MINI PROJECT

Design and implement a simple web application that combines HTML, CSS, Bootstrap, and JavaScript.

ANGULAR

UNIT_020 : INTRODUCTION TO ANGULAR: 20 HRS

Understanding the fundamentals of Angular.
Overview of Angular architecture and its key features.
Setting up the development environment for Angular.

UNIT_021 : TYPESCRIPT:

Introduction to TypeScript, the language used in Angular development.TypeScript syntax, data types, variables, and functions.
Classes, interfaces, and modules in TypeScript.
TypeScript decorators and their usage in Angular.

UNIT_022 : ANGULAR COMPONENTS:

Understanding Angular components and their role in building applications.
Creating components using the Angular CLI (Command Line Interface).
Data binding and event handling in components.
Component lifecycle hooks and their usage.

UNIT_023 : ANGULAR TEMPLATES AND DIRECTIVES:

Angular template syntax and interpolation.
Working with built-in directives (e.g., ngIf, ngFor, ngSwitch).
Creating custom directives in Angular.

UNIT_024 : ANGULAR SERVICES AND DEPENDENCY INJECTION:

Creating and using services in Angular.
Understanding dependency injection and its importance.
Injecting services into components and other Angular constructs.
Sharing data and functionality across components using services.

UNIT_025 : ROUTING AND NAVIGATION:

Implementing client-side routing in Angular.
Configuring routes and route parameters.
Implementing nested routes and child components.
Using router guards for authentication and authorization.

UNIT_026 : ANGULAR FORMS:

Building forms in Angular using template-driven and reactive approaches
Form validation and error handling.
Working with form controls and form groups.
Implementing custom form validators.

UNIT_027 : ANGULAR MODULES AND LAZY LOADING:

Understanding Angular modules and their role in organizing the application.
Creating and configuring modules in Angular.
Implementing lazy loading for optimizing application performance.

UNIT_028 : ANGULAR HTTP AND API INTEGRATION:

Making HTTP requests using the Angular HttpClient module.
Consuming RESTful APIs and handling responses.
Error handling and interceptors in HTTP requests.
Mocking HTTP requests for testing purposes.

UNIT_029 : ANGULAR TESTING AND DEBUGGING:

Writing unit tests for Angular components, services, and directives.
Using testing utilities and frameworks (e.g., Jasmine, Karma).
Debugging Angular applications using browser developer tools.
Performance optimization and debugging techniques.

UNIT_030 : ANGULAR DEPLOYMENT AND BUILD OPTIMIZATION:

Building and bundling Angular applications for production.
Optimizing application size and performance.
Deploying Angular applications to various hosting platforms.
Continuous integration and deployment (CI/CD) for Angular projects.

LAB SET ANGULAR

LAB 1 : SETTING UP ANGULAR DEVELOPMENT ENVIRONMENT

- Install Node.js and Angular CLI.
- Create a new Angular project using Angular CLI.
- Run and test the default Angular application in the browser.

LAB 2 : COMPONENTS AND TEMPLATES

- Create a new component and template.
- Use data binding to display dynamic content in the template.
- Implement event binding to handle user interactions.

LAB 3 : SERVICES AND DEPENDENCY INJECTION

- Create a service to handle data retrieval and manipulation.
- Inject the service into a component and use it to fetch and display data.
- Implement dependency injection to provide the service to the component.

LAB 4 : ROUTING AND NAVIGATION

- Configure routing in the Angular application.
- Create multiple components and associate them with different routes.
- Implement navigation between components using router links and programmatically.

LAB 5 : FORMS AND VALIDATION

- Create a form with input fields and submit functionality.
- Implement form validation using Angular's built-in validators.
- Display validation errors and provide user feedback.

LAB 6 : HTTP REQUESTS AND APIS

- Use Angular's HttpClient module to make HTTP requests to an API.
- Fetch data from an external API and display it in the application.
- Implement CRUD operations (create, read, update, delete) using HTTP requests.

LAB 7 : ANGULAR MATERIAL

- Install and configure Angular Material in the project.
- Use Angular Material components like buttons, cards, and forms.
- Apply styles and themes provided by Angular Material.

LAB 8 : STATE MANAGEMENT WITH NGRX

- Set up NgRx for state management in the Angular application.
- Create actions, reducers, and effects to manage application state.
- Use selectors to retrieve and display data from the store.

LAB 9 : AUTHENTICATION AND AUTHORIZATION

- Implement user authentication using a mock or real authentication service.
- Protect routes and components based on user authentication status.
- Restrict access to certain features based on user roles or permissions.

LAB 10 : DEPLOYMENT AND PRODUCTION BUILD

- Build the Angular application for production.
- Deploy the application to a hosting platform or server.
- Configure routing and server-side rendering for SEO optimization.

REACT

20 HRS

UNIT_031 : INTRODUCTION TO REACT:

Overview of React and its benefits.
Setting up the development environment for React.
Understanding the virtual DOM and React's rendering process.
Introduction to JSX (JavaScript XML) syntax.

UNIT_032 : COMPONENTS AND PROPS:

Understanding React components and their lifecycle.
Creating functional and class components.
Passing and accessing props in components.
Using state and managing component state.

UNIT_033 : JSX AND RENDERING ELEMENTS:

JSX syntax and its usage in React.
Rendering React elements to the DOM.
Conditional rendering and handling dynamic content.
Fragment and its usage for grouping multiple elements.

UNIT_034 : HANDLING EVENTS:

Adding event handlers to React components.
Binding event handlers and passing arguments.
Event delegation and event pooling in React.
Understanding synthetic events in React.

UNIT_035 : STATE MANAGEMENT WITH REACT:

DWorking with component state in React.
Updating state using setState and immutability concept.
Lifting state up and managing shared state.
Using controlled and uncontrolled components.

UNIT_036 : REACT ROUTER:

Implementing client-side routing using React Router.
Configuring routes and route parameters.
Navigating between different pages and components.
Implementing nested routes and route guards.

UNIT_037 : REACT FORMS:

Building forms in React.
Handling form inputs and managing form state.
Form validation and error handling.
Using form libraries and third-party form components.

UNIT_038 : MANAGING DATA WITH REACT:

Fetching and handling data from APIs.
Making HTTP requests using fetch or Axios.
Managing asynchronous operations with promises and async/await.
Using React hooks (e.g., useEffect, useState) for data management.

UNIT_039 : REACT CONTEXT API:

Understanding React's Context API for global state management.
Creating context providers and consumers.
Sharing data and state across components using context.
Implementing context with multiple providers and consumers.

UNIT_040 : REACT HOOKS:

Introduction to React hooks and their benefits.
Using useState for managing state in functional components.
useEffect hook for handling side effects and lifecycle events.
Custom hooks for reusable logic and code abstraction.

UNIT_039 : REACT TESTING AND DEBUGGING:

UWriting unit tests for React components using testing libraries (e.g., Jest, React Testing Library).
Debugging React applications using browser developer tools and React DevTools.
Performance optimization and debugging techniques for React applications..

UNIT_040 : REACT DEPLOYMENT AND BUILD OPTIMIZATION:

Building and bundling React applications for production.
Optimizing application size and performance.
Deploying React applications to various hosting platforms.
Continuous integration and deployment (CI/CD) for React projects.

LAB SET REACT

LAB 1 : ETTING UP REACT DEVELOPMENT ENVIRONMENT

- Set up the development environment by installing Node.js and create-react-app.
- Create a new React project using create-react-app.
- Run and test the default React application in the browser.

LAB 2 : COMPONENTS AND JSX

- Create functional and class components in React.
- Use JSX syntax to render components and HTML elements.
- Pass props to components and display dynamic content.

LAB 3 : STATE AND LIFECYCLE METHODS

- Create stateful components and manage component state.
- Implement lifecycle methods such as componentDidMount and componentDidUpdate.
- Update component state and re-render components accordingly.

LAB 4 : HANDLING EVENTS

- Implement event handlers for user interactions such as button clicks or form submissions.
- Update component state based on user input.
- Use event objects and prevent default behavior when necessary.

LAB 5 : FORMS AND CONTROLLED COMPONENTS

- Create a form with input fields and handle form submissions.
- Implement controlled components by binding form input values to component state.
- Perform form validation and display error messages.

LAB 6 : LISTS AND KEYS

- Render lists of items using the map function.
- Assign unique keys to list items for efficient updates.
- Implement dynamic rendering of lists based on component state.

LAB 7 : CONDITIONAL RENDERING

- Conditionally render components or elements based on component state or props.
- Implement conditional rendering using if statements, ternary operators, or logical operators.

LAB 8 : REACT ROUTER

- Install and configure React Router in the project.
- Create routes and associate them with different components.
- Implement navigation between routes using links or programmatic navigation.

LAB 9 : CONTEXT API

- Create a context using React's Context API.
- Provide and consume context values in different components.
- Update context values and trigger re-renders in consuming components.

LAB 10 : HOOKS AND CUSTOM HOOKS

- Use built-in React hooks such as useState and useEffect in functional components.
- Create custom hooks to encapsulate reusable logic.
- Implement functionality like fetching data, handling side effects, or managing component state using hook

DESIGN

UNIT_021 :

IDesign principle, Good code, architecture design, architecture style, design pattern.

UNIT_022 :

THANK YOU

VISIT US @

+91 63635 07858

www.uplsnipe.com

D0WLOARD THE APP

GET IT ON Google Play

/uplsnipe

uplsnipe

#123, 70Th Cross, 2Nd Floor, SVA Arcade, 5Th Block Rajajinagara, Bengaluru - 560010