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

MEAN STACK

Target Audience: Fresher DEVELOPER (0-2 year experience)
Duration: 120 Hrs
Course Coverage:

- Frontend Technology: HTML4/5, CSS2/3, Bootstrap, JavaScript, Typescript and Angular
- Backend Technology: NPM, NodeJS, ExpressJS
- Database: MySQL, MongoDB

SYLLABUS

UNIT_001: 05 HRS

(HTML, CSS, BHOOTSRAP, REACT/ANGULAR) The project, Domain, Platform,Product, Technology, Tech Stack, Process,Frontend Introduction

UNIT_002 : HTML 06 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_003: CSS 05 HRS

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

UNIT_004 : BOOTSTRAP: 07 HRS

History, Why Bootstrap, Containers, Grid Basic, Typograpy, Colours, Tables, Images, Alerts, Buttons, Progress Bar, Spinner, Pagination, Listgroups, Cards, Dropdowns, Collapse, Navs, Carousel, Modal, Tooltip, Popovers, Toast, Scrollspy, Offcanvas, Utilities, Bootstrap 5 Forms.

UNIT_005 : JAVASCRIPT: 08 HRS

Introduction, History, Features, Browser, Browser Object Model (BOM), How Does Java Script Works?, Document Object Model (DOM), Jsbasics, False Values, Var, Let, Const, Programming Fundamentals, String, Date, Math, Arrays, Dense Array, Forms, Events, DHTML

UNIT_006 : JQUERY : 07 HRS

Introduction, Syntax, Merits and Demerits, JQuery Functions

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.

20 HRS

ANGULAR

UNIT_007 : INTRODUCTION TO ANGULAR:

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

UNIT_008 : 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_009 : 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_010 : 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_011 : 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_012 : 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_013 : 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_014 : 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_015 : 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_016 : 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_017 : 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.

BACKEND TECHNOLOGY

NODEJS BACKEND

UNIT_018 : 05 HRS

Introduction, Setup, Advantages And Disadvantages, Why Nodejs?, Nodejs , Architecture

UNIT_019 : 05 HRS

NodeJs Modules - Functions,Buffer,Module,Module Types, Core Modules,Local , Modules,Module.Exports

UNIT_020 : 07 HRS

(What Is NPM?Installing Packages Locally,Adding Dependency In Package.Json, Installing Packages Globally,Updating Packages, Debugging

UNIT_021 : 05 HRS

Creating Web Sever, Handling Http Methods And File System Reading, Writing, Opening And Deleting Files

UNIT_022 : 05 HRS

Event Driven Framework, Eventemitter Class, Inheriting Events,

UNIT_023 : 06 HRS

Express Framework , Installation, Middleware, How Does Express Work ? ,

UNIT_024 : 05 HRS

Error Handling, Routing, Handling Static Files,CORS,

UNIT_025 : 07 HRS

Database Integration with MySQL/MongoDV, Logging,

UNIT_026 :

CAPSTONE PROJECT

LAB SET NODEJS BACKEND

LAB 1 : SETTING UP NODE.JS ENVIRONMENT

- Install Node.js and set up the development environment.
- Create a basic Node.js project structure.
- Initialize a package.json file to manage project dependencies.

LAB 2 : BUILDING RESTFUL APIS WITH EXPRESS.JS

- Set up an Express.js application to handle HTTP requests.
- Define routes and endpoints for various CRUD operations.
- Implement middleware for request validation, authentication, and error handling.

LAB 3 : DATABASE INTEGRATION WITH MONGODB

- Install MongoDB and set up a local or remote database.
- Connect Node.js application to the MongoDB database.
- Perform CRUD operations on MongoDB collections using Mongoose.

LAB 4 : USER AUTHENTICATION AND AUTHORIZATION

- Implement user registration and login functionality.
- Use libraries like Passport.js for authentication strategies.
- Implement role-based access control for restricting access to certain routes.

LAB 5 : HANDLING FILE UPLOADS AND DOWNLOADS

- Handle file uploads from clients using libraries like Multer.
- Store uploaded files on the server or in cloud storage (e.g., AWS S3).
- Enable file downloads or streaming for clients.

LAB 6 : ERROR HANDLING AND LOGGING

- Implement error handling middleware to handle runtime errors.
- Set up logging using libraries like Winston or Morgan.
- Capture and log application errors and exceptions.

LAB 7 : TESTING AND TEST-DRIVEN DEVELOPMENT (TDD)

- Write unit tests using frameworks like Mocha or Jest.
- Implement test suites to cover different parts of the application.
- Practice test-driven development by writing tests before writing application code.

LAB 8 : AUTHENTICATION WITH JWT (JSON WEB TOKENS)

- Implement authentication using JWT for stateless authentication.
- Generate and verify JWT tokens for securing API routes.
- Use middleware to validate and decode JWT tokens.

LAB 9 : CACHING AND PERFORMANCE OPTIMIZATION

- Integrate caching mechanisms using libraries like Redis.
- Implement caching strategies to improve application performance.
- Optimize API response times and reduce database queries.

LAB 10 : DEPLOYMENT AND CONTINUOUS INTEGRATION

- Deploy Node.js application to cloud platforms like AWS or Heroku.
- Set up continuous integration and deployment pipelines using tools like Jenkins or Travis CI.
- Monitor and manage the deployed application using monitoring and logging tools.

EXPRESS JS

UNIT_019: INTRODUCTION TO EXPRESS.JS — 05 HRS

- Overview of Express.js and its features
- Setting up a basic Express.js application
- Understanding the request-response cycle

UNIT_020 : ROUTING IN EXPRESS.JS — 06 HRS

- Handling different HTTP methods (GET, POST, PUT, DELETE)
- Route parameters and query parameters
- Route handlers and middleware functions
- Route chaining and order of routes

UNIT_021: HANDLING REQUESTS AND RESPONSES — 05 HRS

- Parsing request data (query parameters, request body)
- Handling JSON and form data
- Sending different types of responses (HTML, JSON, files)
- Working with response headers and status codes

UNIT_022: TEMPLATE ENGINES IN EXPRESS.JS — 05 HRS

- Integrating template engines (e.g., EJS, Handlebars, Pug)
- Rendering dynamic views and passing data to templates
- Layouts and partials

UNIT_023 : MIDDLEWARE IN EXPRESS.JS — 06 HRS

- Understanding middleware functions and their role
- Built-in middleware (e.g., static, body-parser)
- Creating custom middleware functions
- Error handling middleware

UNIT_024: WORKING WITH DATABASES AND MODELS — 05 HRS

- Integrating databases (e.g., MongoDB, MySQL) with Express.js
- Using database drivers or ORMs (e.g., Mongoose, Sequelize)
- CRUD operations with database models
- Implementing authentication and authorization

UNIT_025: TEMPLATE ENGINES IN EXPRESS.JS — 05 HRS

- Integrating template engines (e.g., EJS, Handlebars, Pug)
- Rendering dynamic views and passing data to templates
- Layouts and partials

UNIT_026 : MIDDLEWARE IN EXPRESS.JS — 06 HRS

- Understanding middleware functions and their role
- Built-in middleware (e.g., static, body-parser)
- Creating custom middleware functions
- Error handling middleware

UNIT_027: WORKING WITH DATABASES AND MODELS — 05 HRS

- Integrating databases (e.g., MongoDB, MySQL) with Express.js
- Using database drivers or ORMs (e.g., Mongoose, Sequelize)
- CRUD operations with database models
- Implementing authentication and authorization

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

UPL
UNLIMITED POWER FULL LEARNING

UNIT_028: EXPRESS.JS AND RESTFUL APIS — 05 HRS

- Designing and implementing RESTful APIs
- Routing for API endpoints
- Handling JSON data and responses
- Authentication and authorization for APIs

UNIT_029 : EXPRESS.JS AND REAL-TIME COMMUNICATION — 06 HRS

- Introduction to real-time communication protocols (e.g., WebSocket)
- Integrating real-time functionality with Express.js
- Building real-time chat applications

UNIT_030: TESTING AND DEBUGGING EXPRESS.JS APPLICATIONS — 05 HRS

- Writing unit tests for Express.js routes and middleware
- Using testing frameworks (e.g., Mocha, Jest)
- Debugging Express.js applications with logging and debugging tools

UNIT_031: DEPLOYMENT AND SCALING — 05 HRS

- Preparing Express.js application for production
- Deploying Express.js applications to hosting platforms
- Scaling and load balancing strategies

LAB SET NODEJS BACKEND

LAB 1 : SET UP YOUR NODE.JS ENVIRONMENT:

- Install Node.js on your machine by downloading the installer from the official Node.js website (Install Node.js and set up the development environment.
- Create a basic Node.js project structure.
- Initialize a package.json file to manage project dependencies.

). Choose the appropriate version for your operating system and follow the installation instructions.

LAB 2 : CREATE A NEW PROJECT DIRECTORY:

Open a terminal or command prompt and navigate to the directory where you want to create your Express.js project. Use the following command to create a new directory and initialize a new Node.js project

LAB 3 : INSTALL EXPRESS.JS:

In your project directory, install Express.js by running the following command

LAB 4 : SET UP A BASIC EXPRESS.JS APPLICATION:

In the 'index.js' file, write the code to set up a basic Express.js application.

LAB 5 : TEST AND RUN THE EXPRESS.JS APPLICATION:

Save the changes to your index.js file and run the application using the node command followed by the file name. Use the following command in the terminal:

LAB 6 : ACCESS THE APPLICATION:

Open a web browser and visit http://localhost:3000. You should see the message "Hello, Express!" displayed in the browser.

LAB 7 : EXPAND AND EXPERIMENT WITH THE APPLICATION:

From this starting point, you can continue to build upon the basic Express.js application. Add routes, middleware, and explore different features of Express.js such as handling form submissions, working with databases, implementing authentication, and more. Refer to the official Express.js documentation (Expand and experiment with the application:) for detailed explanations and examples of each feature.

LAB 8 : TEST AND RUN EACH LAB EXERCISE:

Save the changes to your lab files and run them using the node command followed by the file name. Observe the output in the terminal and test the functionality in your web browser.

DATA VISUALIZATION

UNIT_020 : DATA VISUALIZATION 06 HRS

Understanding Exports And Require, Creating Modules, Importing Modules ,Npm ,Data Visualization Principles and Design Fundamentals, Data Preparation and Exploration, Basic Visualization Techniques, Interactive Visualizations, Geographic and Spatial Visualization, Time Series Visualization, Dashboard Design and Data Storytelling, Ethical and Responsible Data Visualization, Data Visualization Best Practices and Evaluation.

LAB SET DATA VISUALIZATION

LAB 1 : SET UP YOUR PROGRAMMING ENVIRONMENT:

Install the necessary software and libraries for data visualization. Some popular options include Python, Jupyter Notebook, and libraries such as Matplotlib and Seaborn.

LAB 2 : INSTALL THE REQUIRED PACKAGES:

Open a terminal or command prompt and install the required data visualization libraries. You can use the following commands for installing Matplotlib and Seaborn in Python:

LAB 3 : CREATE A NEW JUPYTER NOTEBOOK:

Open Jupyter Notebook by running the command jupyter notebook in the terminal. This will open the Jupyter Notebook interface in your web browser.

LAB 4 : CREATE A NEW NOTEBOOK:

In the Jupyter Notebook interface, click on "New" and select "Python 3" to create a new notebook.

LAB 5 : IMPORT THE NECESSARY LIBRARIES:

In the first cell of your notebook, import the required libraries for data visualization. For example:

LAB 6 : LOAD AND EXPLORE YOUR DATASET:

Obtain a dataset that you want to visualize. You can use popular datasets available online, or import your own dataset. Read and explore the dataset using pandas or any other data manipulation library.

LAB 7 : CREATE BASIC VISUALIZATIONS:

Use the data visualization libraries to create basic visualizations such as line plots, bar charts, scatter plots, or histograms. Experiment with different visualization types to understand the characteristics of your dataset.

LAB 8 : ENHANCE VISUALIZATIONS WITH CUSTOMIZATION:

Explore various customization options such as labels, colors, legends, and axes formatting to make your visualizations more informative and visually appealing.

LAB 4 : EXPLORE ADVANCED VISUALIZATIONS:

Dive into more advanced data visualizations such as heatmaps, box plots, violin plots, or stacked area charts. Experiment with different visualization techniques to effectively represent your data.

LAB 5 : DOCUMENT AND SAVE YOUR VISUALIZATIONS:

Add appropriate titles, labels, and legends to your visualizations. Save your visualizations as image files or export them to PDF format for future reference or presentation purposes.

EVENTS & STREAMS

UNIT_021 : EVENTS & STREAMS 07 HRS

Significance Of Events, Event Emitter Class, Emitting And Listening To Events, Types Of Streams, Working With Streams, Composing Streams Using Pipe, Debugging In Express, Realtime Programming With Socket.Io, Scaling Nodejs Applications, The Child Process Model, Exec, Spawn, And Fork Functions, Using The Cluster Module, Building Reactive Systems with Streams, Event Processing and Complex Event Processing (CEP),Event-Driven Security and Resilience,Event-Driven Workflow Orchestration

LAB SET EVENTS & STREAMS

LAB 1 : CHOOSE A PROGRAMMING LANGUAGE:

Select a programming language that supports event-driven and stream-based programming paradigms. Some popular choices include JavaScript/Node.js, Java, Python, and C#.

LAB 2 : SET UP YOUR PROGRAMMING ENVIRONMENT:

Install the necessary software and libraries for event-driven and stream-based programming. This may include an Integrated Development Environment (IDE), a runtime environment, and relevant libraries or frameworks specific to your chosen programming language.

LAB 3 : CHOOSE AN EVENT-DRIVEN FRAMEWORK OR LIBRARY:

Explore event-driven frameworks or libraries that are widely used in your selected programming language. For example, in JavaScript/Node.js, you can consider frameworks like Node.js Event Emitter, RxJS, or KafkaJS.

LAB 4 : INSTALL THE REQUIRED PACKAGES:

Use a package manager specific to your chosen programming language to install the required packages or libraries. For example, in Node.js, you can use npm or yarn to install the necessary packages.

LAB 5 : SET UP A BASIC EVENT-DRIVEN APPLICATION:

Create a simple event-driven application to understand the fundamentals. Define events, event handlers, and how events can be emitted or triggered. Use the event-driven framework or library of your choice.

LAB 6 : EXPLORE STREAM PROCESSING:

Learn about stream processing and how it differs from event-driven programming. Set up a basic stream processing application using stream-based libraries or frameworks, such as Apache Kafka or Apache Flink.

LAB 7 : BUILD REAL-TIME APPLICATIONS:

Use events and streams to develop real-time applications. Explore scenarios like real-time analytics, real-time data processing, or real-time collaborative systems. Implement event-driven or stream-based solutions to handle these use cases.

LAB 8 : TEST AND DEBUG YOUR APPLICATIONS:

Perform testing and debugging of your event-driven and stream-based applications. Learn techniques and tools for unit testing, integration testing, and debugging in the event-driven and stream processing context.

LAB 9 : MONITOR AND OPTIMIZE PERFORMANCE:

Understand how to monitor and optimize the performance of event-driven and stream-based applications. Explore tools and techniques for monitoring event streams, measuring throughput, and handling high-volume event processing.

UNIT_022 :

CAPSTONE PROJECT

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