FULL STACK DEVELOPMENT

TECHNOLOGY

GREENS TECHNOLOGIES

ANNA NAGAR

89399 82022

GREENS
TECHNOLOGIES

ABOUT GREEN TECHNOLOGIES

Greens Technology: Your Path to Success in Full Stack Development
Greens Technology stands out as a premier provider of project-oriented Full Stack Development programs, offering 100% job placement support upon completion. Our curriculum features mentorship from industry leaders at renowned companies such as Google, Microsoft, Flipkart, Zoho, and Freshworks, ensuring our students secure lucrative positions in top organizations.The Full Stack Developer Program is thoughtfully crafted to thoroughly cover every technology, equipping participants with the essential skills to thrive as successful Full Stack Developers.

Our Mission
To make technology education accessible to everyone around the globe by adhering to the standards set by the EdTech industry.

Our Vision
To transform lives by equipping learners with advanced tech skills in their native languages and bridging the gap between aspiring tech professionals and the corporate industry.

ABOUT TRAINER

SENIOR FULL STACK TRAINER

Miss Divya shree S is a passionate and experienced Full Stack Developer and AI/ML trainer with experience in software development and mentoring. Having worked with leading tech companies and startups, she brings real-world expertise in Java, Spring Boot, React.js, Node.js, MongoDB, and AI/ML tools like Python She has built and deployed scalable web applications and integrated AI models into real-time systems for smart solutions.

FULL STACK TRAINER

Mr. Raghul Krishnan J specializes in end-to-end development, including frontend design, REST API development, backend architecture, database management, and cloud deployment. In the AI/ML space, he focuses on making AI practical for developers—teaching students how to embed machine learning models into full stack applications such as chatbots, image classifiers, and recommendation systems.

Miss DIVYA SHREE S
FULL STACK TRAINER
GREEN TECHNOLGIES
9944842073

Mr. Raghul Krishnan J
FULL STACK TRAINER
GREEN TECHNOLGIES
9080058961

GREENS
TECHNOLOGIES

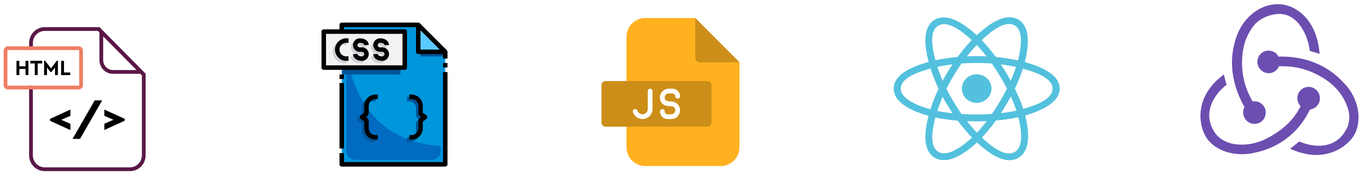
COURSE WE OFFERED

- **JAVA FULL STACK DEVELOPMENT**
- **PYTHON FULL STACK DEVELOPEMENT**
- **MERN STACK DEVELOPMENT**
- **MEAN STACK DEVELOPMENT**
- **AI ENABLED FULL STACK DEVELOPEMENT**
- **BEGINNER COURSE (C , C++ , JAVA , PYTHON)**

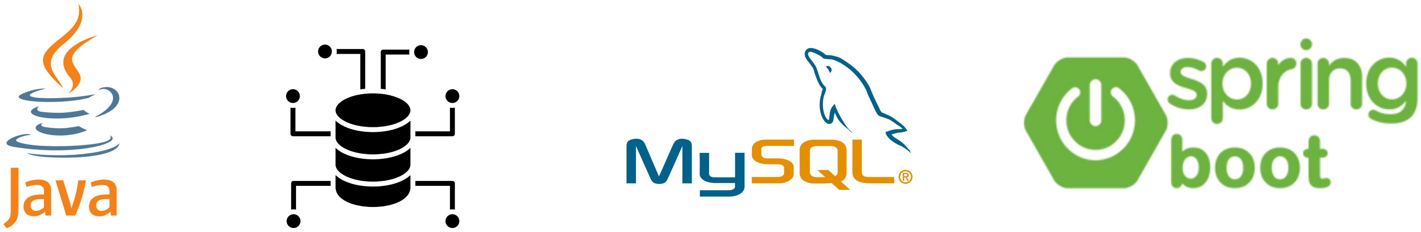
GREENS TECHNOLOGIES

JAVA FULL STACK DEVELOPEMENT

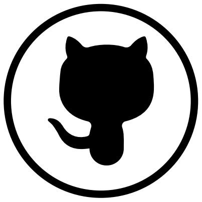
FRONTEND DEVELOPMENT



BACKEND DEVELOPMENT



VERSION CONTROL

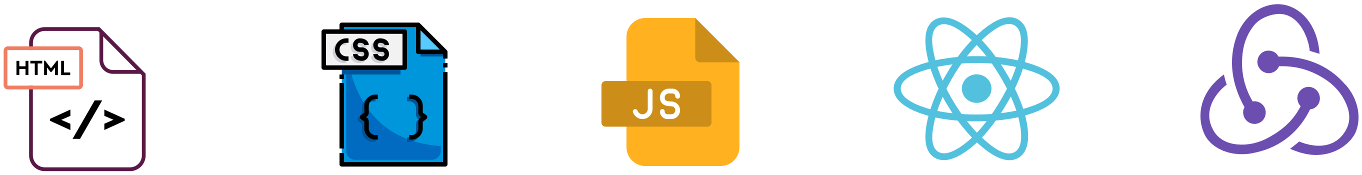


OTHER TOOL

GREENS TECHNOLOGIES

PYTHON FULL STACK DEVELOPEMENT

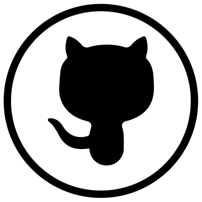
FRONTEND DEVELOPMENT



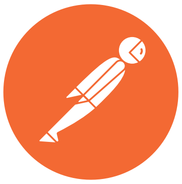
BACKEND DEVELOPMENT

MySQL®
Flask
web development,
one drop at a time

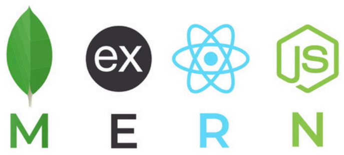
VERSION CONTROL



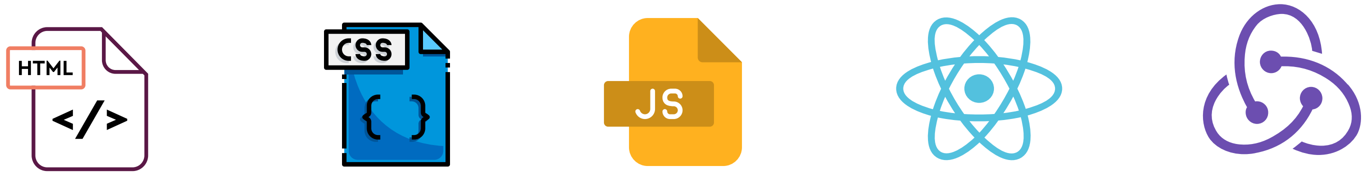
OTHER TOOL



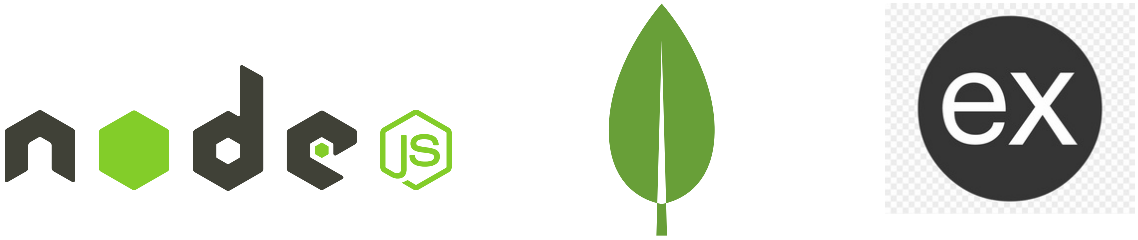
MERN STACK DEVELOPEMENT



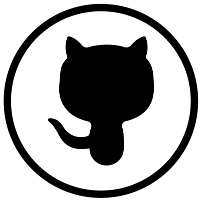
FRONTEND DEVELOPMENT



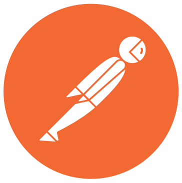
BACKEND DEVELOPMENT



VERSION CONTROL

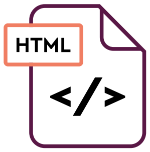


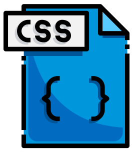
OTHER TOOL



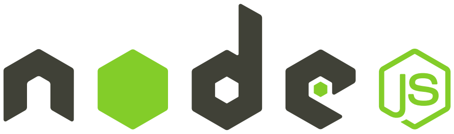
MEAN STACK DEVELOPEMENT

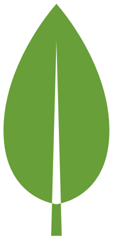
FRONTEND DEVELOPMENT



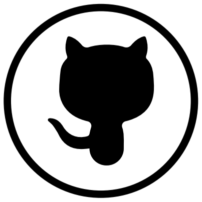


BACKEND DEVELOPMENT

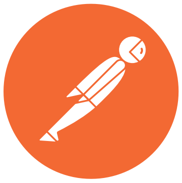




VERSION CONTROL

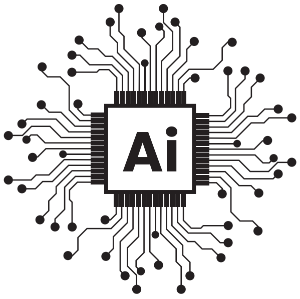


OTHER TOOL

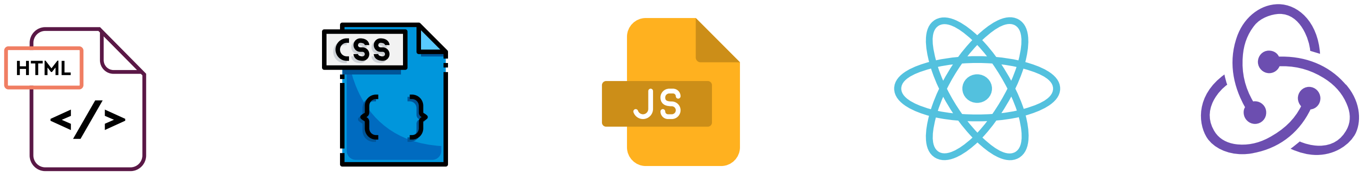


GREENS
TECHNOLOGIES

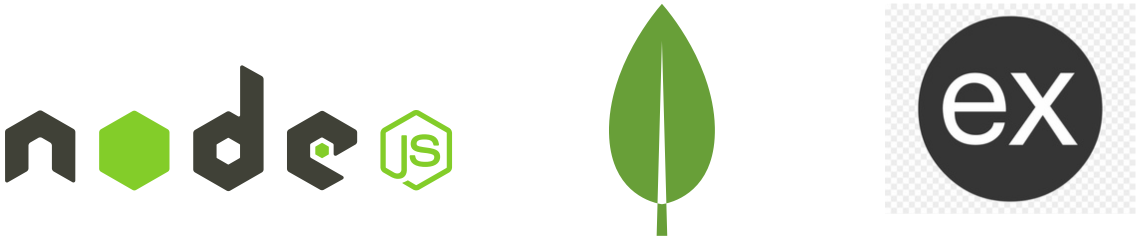
AI ENABLED FULL STACK DEVELOPEMENT



FRONTEND DEVELOPMENT

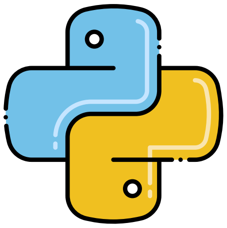


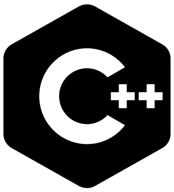
BACKEND DEVELOPMENT



AI TOOLS

BEGINNER COURSE





PROJECTS

PROJECT -1

Portfolio using Html and css

PROJECT -2

Clone of real time web application

PROJECT -3

E Commerce Application
Using React-js

PROJECTS

PROJECT -4

AI Chat bot Application usinfg AI model

PROJECT -5

Nextflix clone application

PROJECT -6

BookMyshow Clone

HTML

- <!DOCTYPE html>, <html>, <head>, <body>
- Title tag and metadata
- Structure of a basic HTML document

📝 Headings, Paragraphs & Line Breaks

- <h1> to <h6> headings
- <p> paragraph tag
- <br> line breaks and <hr> horizontal rule

🔤 Text Formatting Tags

- <b> Bold, <i> Italic, <u> Underline
- <strong>, <em> for semantic emphasis
- <mark>, <small>, <sub>, <sup>, <del>, <ins>

📝 Lists

- Ordered List: <ol>, <li>
- Unordered List: <ul>, <li>
- Description/List tag: <dl>, <dt>, <dd>

🔗 Links (Anchor Tags)

- <a href="URL">Link Text</a>
- Target attribute for opening in new tab (_blank)
- Linking to sections with id anchors

HTML

🖼️ Images

- <img src="image.jpg" alt="description">
- Image attributes: height, width, title
- Accessibility: Proper use of alt

📊 Tables

- <table>, <tr>, <td>, <th>
- Table attributes: colspan, rowspan, border

🧾 Forms

- <form>, action, method
- Input fields: <input>, <textarea>, <select>, <option>
- Input types: text, number, checkbox, radio, password, email, date, etc.
- Buttons: <button>, <input type="submit">
- Label association using for attribute

🧱 Semantic HTML5 Tags

- <header>, <nav>, <main>, <article>, <section>, <aside>, <footer>

CSS

📏 Width, Height, Spacing

- width, height, max-width, min-height
- margin and padding (shorthand syntax)

🎨 Colors & Backgrounds

- Color formats: name, hex, rgb(), rgba(), hsl()
- background-color, background-image
- background-repeat, background-size, background-position

🖋️ Fonts & Text Styling

- font-family, font-size, font-weight, font-style
- color, line-height, letter-spacing, text-align
- text-decoration, text-transform, text-shadow

🔲 Borders & Shadows

- border, border-radius
- box-shadow

🧰 Flexbox Basics

- display: flex
- justify-content, align-items, flex-direction, flex-wrap
- Aligning items horizontally and vertically

🧩 CSS Grid (Intro)

CSS

🎨 CSS (Cascading Style Sheets)

📌 Introduction to CSS

- What is CSS and why it’s used
- Ways to apply CSS: Inline, Internal, External

🎯 CSS Selectors

- Basic Selectors: element, .class, #id
- Grouping selectors
- Descendant & Child selectors
- Pseudo-classes: :hover, :focus, :nth-child()
- Pseudo-elements: ::before, ::after

📐 Box Model

- Content, Padding, Border, Margin
- box-sizing: content-box vs border-box
- Visual debugging using browser tools

🧱 Layout & Positioning

- display: block, inline, inline-block, none
- position: static, relative, absolute, fixed, sticky
- z-index and stacking context
- overflow: visible, hidden, scroll, auto

⚙️ JavaScript (JS) Syllabus

📌 Introduction to JavaScript

- What is JavaScript? Why is it used?

🔤 Variables & Data Types

- var, let, const (scope & usage)
- Data types: string, number, boolean, null, undefined, object
- Type checking: typeof
- Type conversion (implicit & explicit)

🔁 Operators

- Arithmetic: +, -, *, /, %
- Assignment: =, +=, -=, etc.
- Comparison: ==, ===, !=, !==, >, <, >=, <=
- Logical: &&, ||, !
- Ternary operator

🔄 Control Flow

- if, else if, else
- switch statement
- for, while, do...while loops
- break, continue

📦 Arrays

- Create arrays and access elements
- Array methods: push, pop, shift, unshift, splice, slice
- Looping through arrays: for, for...of
- map, filter, reduce (intro)

📘 Strings

- String properties and methods: length, toUpperCase(), includes(), replace(), split(), etc.

📚 Functions

- Function declaration & expression
- Parameters, return values
- Arrow functions (=>)
- Callback functions (intro)

🧱 Objects

- Creating and accessing objects
- Nested objects
- this keyword (basic use)
- Object methods and property access

Events & DOM Manipulation

- What is the DOM?
- getElementById, querySelector, etc.
- Changing content and style with JS
- Handling events: onclick, addEventListener

⏱️ Timing & Loops

- setTimeout(), setInterval(), clearInterval()

🧵 JSON & Data Handling

- JSON structure & parsing (JSON.stringify, JSON.parse)

⚛️ React.js Syllabus

📌 Introduction to React

- What is React and why use it?
- Virtual DOM vs Real DOM
- Installing Node.js and Create React App (CRA)
- Folder structure overview

🧩 JSX (JavaScript XML)

- Writing HTML inside JavaScript
- Rules of JSX (single parent, expressions, etc.)
- Fragment: <></> or <React.Fragment>

🧠 Components

- Functional Components
- Props (Passing data to components)
- Reusable and nested components
- Default props and prop types

🔄 State & Events

- useState() hook
- Updating state on events (onClick, onChange)
- Handling forms and inputs
- Controlled vs uncontrolled components

🔄 Conditional Rendering

- if-else, ternary (? :), && operator
- Rendering components or elements conditionally

🔁 Lists and Keys

- Rendering lists using map()
- Using key prop correctly
- Dynamic component rendering

🔧 useEffect Hook

- What is useEffect and when it runs
- useEffect with dependencies

🧮 Forms Handling

- Form elements in React
- Managing inputs with state
- Validation basics

🧭 Routing in React

- Installing React Router
- BrowserRouter, Routes, Route
- Link, NavLink, useNavigate, useParams

🗃️ Lifting State Up & Props Drilling

- Passing data from child to parent
- Centralizing shared state

🛠️ Custom Hooks (Basic)

- Creating and using custom hooks
- Reusability with hooks

📦 API Integration

- Fetching data using fetch() and axios
- Using useEffect for API calls
- Displaying API data
- Loading & error states

🔄 Redux

- Introduction to Redux
- Core Concepts: Store, Action, Reducer
- Creating a Redux Store
- Dispatching Actions
- Reducers and State Updates

🚀 Introduction to Java

- What is Java?
- History and features
- JDK, JRE, JVM
- How Java works (platform-independent)

💻 Java Basics

- Writing your first Java program
- main() method and structure
- Data types, variables, constants
- Type casting and type conversion
- Operators: arithmetic, logical, relational

📦 Object-Oriented Programming (OOP)

- Classes and Objects
- Constructors (default & parameterized)
- this keyword
- Inheritance (extends)
- Polymorphism (method overloading & overriding)
- Abstraction: abstract classes and interfaces
- Encapsulation: private fields & getters/setters
- final, static, super keywords

🔁 Control Statements

- if, if-else, else-if
- switch-case
- Loops: for, while, do-while
- break, continue, return

🧠 Arrays & Strings

- Single and multi-dimensional arrays
- Array operations (looping, sorting)
- String class and methods
- StringBuffer

🧰 Exception Handling

- Types of exceptions (checked & unchecked)
- Try-catch-finally blocks
- throw and throws
- Custom exceptions

📚 Packages and Access Modifiers

- Built-in and user-defined packages
- public, private, protected, default
- import keyword usage

📥 Input & Output (I/O)

- Scanner class for user input
- File handling: reading/writing files using FileReader, BufferedReader, FileWriter

🧮 Collections Framework

- Introduction to collections
- List, Set, Map interfaces
- ArrayList, LinkedList, HashSet, TreeSet
- HashMap, LinkedHashMap

🌐 JDBC (Java Database Connectivity)

- Connecting Java with MySQL
- DriverManager, Connection, Statement, ResultSet
- Insert, update, delete, read operations

🚀 Introduction to Python

- What is Python?
- Features and applications
- Installing Python and setting up IDE (VS Code, PyCharm)
- Writing and running your first Python script

🔤 Variables & Data Types

- Declaring variables
- Data types: int, float, str, bool, complex
- Type conversion and type checking
- Input/output with input() and print()

⚙️ Operators & Expressions

- Arithmetic, comparison, logical, assignment operators
- Identity and membership operators
- Operator precedence

🔄 Control Flow

- if, elif, else statements
- Nested conditions
- match (Python 3.10+ switch-case)

🔄 Loops

- for and while loops
- range() function
- break, continue, pass
- Looping through strings, lists, dictionaries

📦 Data Structures

- Lists, Tuples
- Sets, Dictionaries

🧠 Functions

- Defining and calling functions
- Parameters, return values
- Lambda functions

📚 Modules & Packages

- Importing built-in modules (math, random, datetime)
- Creating user-defined modules
- pip and installing external packages

File Handling

- Opening and reading files (open(), read(), readlines())
- Writing to files (write(), writelines())
- Using with statement for file handling

⚠️ Exception Handling

- Try-except blocks
- Finally clause
- Raising exceptions
- Handling multiple exceptions

🧱 Object-Oriented Programming (OOP)

- Classes and objects
- __init__() constructor
- Inheritance, Polymorphism
- Encapsulation and Abstraction
- @staticmethod, @classmethod

🔰 Introduction to MySQL

- What is a Database & RDBMS?
- Overview of MySQL
- MySQL Workbench / CLI setup
- Database vs Table

📁 Database Basics

- Creating and selecting a database
- Creating tables with columns
- Data types in MySQL (INT, VARCHAR, DATE, etc.)
- NULL vs NOT NULL

✏️ CRUD Operations

- Create – INSERT INTO
- Read – SELECT
- Update – UPDATE
- Delete – DELETE

🔍 Filtering & Sorting Data

- WHERE clause
- Operators: =, !=, <, >, BETWEEN, IN, LIKE
- ORDER BY, LIMIT

🧠 Functions in MySQL

- String functions: CONCAT, LOWER, UPPER, LENGTH
- Numeric functions: ROUND, CEIL, FLOOR

🧾 Aggregate Functions

- COUNT(), SUM(), AVG(), MIN(), MAX()
- GROUP BY and HAVING clauses

🔄 Joins in MySQL

- INNER JOIN
- LEFT JOIN
- RIGHT JOIN
- FULL JOIN (using UNION)
- Joining multiple tables

🔐 Constraints & Keys

- PRIMARY KEY, FOREIGN KEY
- UNIQUE, DEFAULT, CHECK, AUTO_INCREMENT

🧱 Table Management

- Altering table: ALTER TABLE
- Dropping table: DROP TABLE
- Renaming, adding/removing columns

🚀 Introduction to Node.js

- What is Node.js?
- Why use Node.js?
- Node.js vs traditional backend

🧱 Express.js Framework

- Installing and setting up Express
- Creating routes: GET, POST, PUT, DELETE
- Sending responses and handling requests
- Route parameters and query strings

📥 Handling Forms & JSON

- Parsing req.body using express.json()
- Handling form submissions with body-parser or built-in middleware
- Working with Postman for API testing

🗃️ Working with MongoDB (Intro)

- Connecting Node.js with MongoDB using mongoose
- Defining schemas and models
- CRUD operations with MongoDB
- find(), save(), updateOne(), deleteOne()

🛡️ REST API with Node + Express

- Creating RESTful endpoints
- API testing with Postman
- Structuring folders for routes, controllers, services

🔐 Authentication & Middleware

- Middleware functions in Express
- Creating custom middleware
- Intro to JWT (JSON Web Token)
- Protecting routes with JWT

⚙️ Error Handling & Validation

- Global error handlers
- Try-catch in async functions
- Input validation using express-validator or joi

🚀 Introduction to C

- What is C language?
- History and features
- Structure of a C program
- Compile & run using GCC or Code::Blocks
- Understanding #include and main()

🔤 Variables & Data Types

- Declaring variables
- Data types: int, float, char, double
- Constants and keywords
- Format specifiers and printf, scanf

⚙️ Operators & Expressions

- Arithmetic, relational, logical, assignment
- Increment/decrement (++, --)
- Conditional (? :)

🔁 Control Flow Statements

- if, else if, else
- switch-case
- Nested condition

🔄 Looping Statements

- for, while, do-while
- Using break and continue
- Nested loops

📚 Arrays and Strings

- Declaring and initializing arrays
- One-dimensional and two-dimensional arrays
- Basic string operations using arrays
- String functions: strlen, strcpy, strcmp, etc.

🧠 Functions in C

- Function declaration, definition, and calling
- Pass by value vs pass by reference
- Recursion basics
- Scope and lifetime of variables

🧱 Pointers

- Introduction to pointers
- Pointer arithmetic
- Pointers and arrays
- Pointers to functions

GREENS TECHNOLOGIES

🚀 Introduction to C++

- What is C++?
- History and features
- Difference between C and C++

🔤 Variables, Data Types & I/O

- Declaring variables
- Data types: int, float, char, double, bool
- Input/output: cin, cout

⚙️ Operators & Expressions

- Arithmetic, relational, logical, assignment
- Increment/decrement
- conditional operators

🔁 Control Flow Statements

- if, else if, else
- switch-case

🔄 Looping Statements

- for, while, do-while loops
- break, continue
- Nested loops

📚 Arrays and Strings

- Single and multi-dimensional arrays
- Character arrays and string handling

🧠 Functions in C++

- Function declaration and definition
- Parameters & return values

🧱 Object-Oriented Programming (OOP)

- Classes and objects
- Constructors and destructors
- Access specifiers: public, private, protected
- Inheritance: single, multilevel, multiple
- Polymorphism: compile-time (overloading), runtime (overriding)
- Abstraction & encapsulation

🧭 Pointers & References

- Pointer declaration and dereferencing
- Pointers with arrays and functions
- this pointer

GREENS
TECHNOLOGIES

FOR FURTHER DETAILS

CONTACT

89399 82022
ANNA NAGAR

www.greenstechnologys.com/