Welcome To 7i Tech Solutions

Designing → Development → Deployment

Full Stack Python

Course Content

Warm Up For Programing

- Computer Basic
- Short cuts
- E-mail
- Software Installations

Designing

Figma

1. What is Figma?
2. Versions and setup
3. Figma tools
4. What is WireFrames
5. Creating wireframes
6. What is Components
7. Creating components
8. Color theory
9. Typography
10. High fedility
11. Template

Development

HTML

HTML is the foundation of every web page — it structures the content you see online.

HTML stands for HyperText Markup Language. It is the standard language used to create and structure content on the web.

Topics List:

1) Introduction to HTML
2) Basic HTML Syntax
3) HTML Document Structure
4) Text Formatting Elements
5) Lists in HTML
6) Links (Anchor Tags)
7) Images in HTML
8) Tables in HTML
9) Forms and Input Elements
10) Semantic HTML

11) Multimedia Elements

12) HTML Entities

13) Block-level vs Inline Elements

14) IDs and Classes

15) HTML5 Features

16) Accessibility Basics

17) Best Practices in HTML

18) Project Work

CSS (Cascading Style Sheet)

CSS styles the web — turning plain HTML into beautiful, responsive designs.

CSS stands for Cascading Style Sheets. It is used to style and design the appearance of HTML content on a web page.

Topics List:

1) Introduction to CSS
2) Basic CSS Syntax
3) CSS Selectors
4) Colors & Backgrounds

5) Text and Font Styling
6) CSS Box Model
7) Borders and Outlines
8) Sizing and Units
9) Backgrounds and Borders
10) CSS Units
11) Overflow Control
12) Display & Visibility
13) Layout Basics
14) Transitions and Animations
15) Pseudo-classes and Pseudo-elements
16) Positioning Elements
17) CSS Variables (Intro)
18) CSS Flexbox
19) CSS Grid
20) CSS Pseudo-classes and Pseudo-elements
21) Transitions and Basic Animations
22) Media Queries (Responsive Design)
23) CSS Best Practices
24) Mini Projects for Practice (Figma file will be shared)

Bootstrap (No JS Knowledge Required)

Bootstrap simplifies responsive web design with ready-to-use components and a powerful grid system.

It provides a collection of pre-designed HTML, CSS, and JavaScript components—such as buttons, forms, navigation bars, modals, and more—so developers don't have to build everything from scratch.

Topics List:

1) Introduction to Bootstrap
2) Setting Up Bootstrap
3) Bootstrap Grid System
4) Typography
5) Colors and Backgrounds
6) Buttons
7) Images and Figures
8) Tables
9) Forms
10) Navigation Components
11) Layout Utilities
12) Responsive Helpers

13) Cards
14) Modals and Popups
15) Components and UI Elements
16) Helpers and Utilities
17) Icons (Bootstrap Icons)
18) Responsive Design in Bootstrap
19) Customizing Bootstrap
20) Custom CSS
21) Mini Projects for Practice (Figma file will be shared)

JavaScript

JavaScript brings web pages to life with interactivity, logic, and dynamic behavior.

JavaScript is a programming language used to make web pages interactive and dynamic.

Topics List:

1) Introduction to JavaScript
2) Variables
3) Data Types

4) Operators
5) Control Flow Statements
6) Functions
7) Scope and Hoisting
8) Template Literals
9) Type Conversion and Coercion
10) Error Handling
11) Basic Debugging

DOM (Document Object Model)

1) What is the DOM?
2) Selecting Elements
3) Traversing the DOM
4) Manipulating Element Content
5) Manipulating Element Attributes
6) Manipulating Styles
7) Creating and Removing Elements

8) Event Handling
9) Event Delegation
10) Form Handling and Validation
11) Keyboard and Mouse Events
12) Working with Timers and Animation
13) Best Practices for DOM Manipulation

Asynchronous JavaScript

1) Understanding Asynchronous JavaScript
2) Callbacks
3) Promises
4) Async/Await
5) Fetching Data with Fetch API
6) Other Asynchronous APIs and Timers
7) Event Loop and Call Stack (Conceptual)
8) Advanced Promise Concepts
9) Error Handling in Asynchronous Code

ES6 Features and Topics List

1) Arrow Functions
2) Template Literals

3) Default Parameters
4) Destructuring Assignment
5) Spread and Rest Operators
6) Enhanced Object Literals
7) Classes
8) Modules
9) Promises
10) Iterators and Generators
11) Symbols
12) New Built-in Methods
13) Map and Set Data Structures
14) Block-scoped Functions

LocalStorage Topics List

1) Introduction to Web Storage
2) LocalStorage Basics
3) Data Types and Serialization
4) Common Use Cases
5) Handling Storage Limits and Errors
6) Best Practices

React

React builds fast, interactive user interfaces with reusable components and efficient updates.

React JS is a JavaScript library used for building user interfaces, especially single-page applications (SPAs).

It was developed by Facebook and is widely used because it makes UI development faster, modular, and easier to manage through components.

Topics List

1) What is React?
2) React App Architectures
3) Component Based Development
4) React Vs Vanilla JS
5) React Ecosystem
6) How React Works
7) Environment Setup
8) Introduction to JSX
9) Vite Dev Server 7 Build Tool

10) Creating Components
11) Styling in React
12) Lists
13) Handling Events
14) Introduction to State
15) useState Hook
16) Conditional Rendering and Styling
17) Component Props
18) Component Composition
19) Controlled Inputs
20) Form Data Object
21) Form Submission
22) Resuable Components
23) Prop Drilling
24) Component Lifecycle
25) useEffect and side effects
26) useRef Hook in Action
27) Making HTTP Request
28) Environment Variables
29) React Router

30) Deployment to Vercel

Projects to be Done:

1) Crypto Dash Project
2) Shopping Cart UI
3) Github User Finder

Note: The above projects included loaders, Filtering and Pagination

Git & GitHub

From first commit to final release — Git & GitHub make it happen.

Topics List:

1) What is Git and why use it?
2) Installing Git
3) Basic Git configuration (git config)
4) Git workflow
5) Viewing and comparing changes
6) Ignoring files with .gitignore

7) Working with Repositories
8) Branching and Merging
9) Remote Repositories (GitHub Focus)
10) Collaboration Workflow
11) Advanced Git Topics
12) GitHub Pages

Deployment (Vercel)

Topics List:

1) Project Preparation
2) Vercel Account Setup
3) Deploying Your Project
4) Post-Deployment
    a. Debug build errors
    b. Handle routing with react-router (configure vercel.json or rewrites)
    c. Add a custom 404 page
    d. Optimize build size or performance

"You don't have to be great to start, but you have to start to be great."
— *Zig Ziglar*

Backend

Python

Basics:

Introduction:

1. What is Python?
2. Why we can use Python?
3. IDE's for Python?

Topics:

1. Introduction
2. Installing server and IDE
3. Variables
4. Types of variables
5. Data Types
   a. Numeric
      i. Int
      ii. Float
      iii. complex
   b. Textual
      i. String → str
   c. Boolean

6. Type conversions
7. Strings
8. String modifiers
9. Operators
    a. Arithmetic operators
    b. Assignment operators
    c. Comparison operators
    d. Logical operators
    e. Bitwise operators
10. User input
11. Conditional statements
    a. If
    b. Else
    c. Elif
    d. Nested if
    e. programs
12. Loops
    a. For loop
    b. While loop
    c. Break statement
    d. Continue statement
    e. Programs
13. Functions
    a. Declaration and syntax
    b. Parameters and arguments
    c. Arbitrary arguments

d. Keyword arguments
e. Arbitrary keyword arguments

14. Lambda functions
15. Return statement
16. Arrays
17. Complex data types
    a. List
    b. Tuple
    c. Dictionary
    d. Set
    e. List compharance
    f. Dictionary compharance
18. Debugging
19. Error handling
20. File handling
    a. Creating
    b. Write and append
    c. delete

Advance:

1. OOPS
2. Class and objects
    a. Init() function
    b. Str() function
    c. Object methods
    d. Modify objects

3. Inheritance
    a. Single level
    b. Multi level
    c. Multiple level
    d. Hirarical
    e. Add properties and methods
    f. Super() function
4. Iterators
5. Polymorphism
    a. Class polymorphism
    b. Inheritance polymorphism
6. Scope
    a. Local scope
    b. Global scope
7. Operator overloading
8. Method overloading
9. Method overriding
10. Abstraction
11. Encapsulation
12. Modules
    a. Math modules
    b. Date modules
    c. Time modules
13. JSON
14. Regular expressions
15. Generators

16. Decorators
17. PIP

Django

1. What is framework?
2. Why we can use framework?
3. Django intro
4. Installation
5. Creating project
6. Create app
7. HTTP response
8. MVT structure
    a. Model
    b. View
    c. Template
9. Creating admin
10. Migrations and superuser
11. Creating templates → html files
12. Creating static → css and js files
13. Creating media → images and videos
14. Linking css and html
15. Settings configurations
16. Media configurations
17. Creating views and urls
18. Linking views and urls
19. Modules

20. Importing modules
21. Create a table in data base
22. Adding and delate in database
23. Creating forms
24. Fetching database data
25. Rest API
26. Sqlite3
27. CRUD operations
    a. Creating
    b. Read
    c. Update
    d. Delete
28. Authentication
29. Registration page
30. Login page
31. Logout
32. Dynamic errors from Django

SQL(Structured Query Language):

Basic level

1. Introduction to Databases
2. What is SQL?
3. SQL Syntax & Statements Overview
4. Data Types in SQL
5. CREATE, DROP, and ALTER Tables
6. INSERT INTO Statement

7. SELECT Statement
8. WHERE Clause
9. AND, OR, NOT Operators
10. ORDER BY Clause
11. LIMIT / TOP Clause
12. DISTINCT Keyword
13. UPDATE Statement
14. DELETE Statement
15. IN, BETWEEN, LIKE Operators
16. IS NULL / IS NOT NULL
17. Aggregate Functions
18. SUM(), AVG(), MIN(), MAX(), COUNT()
19. GROUP BY Clause
20. HAVING Clause
21. Aliases (AS)
22. Joins
23. INNER JOIN
24. LEFT JOIN
25. RIGHT JOIN
26. FULL JOIN
27. SELF JOIN
28. CROSS JOIN
29. UNION and UNION ALL
30. Subqueries
31. Scalar subquery
32. Correlated subquery

33. Nested subquery
34. Views
35. Creating and using views
36. Updatable vs non-updatable views
37. Indexes
38. Single-column and multi-column indexes
39. Unique indexes
40. Constraints
41. PRIMARY KEY, FOREIGN KEY
42. UNIQUE, NOT NULL, CHECK, DEFAULT

Advanced Level:

1. Triggers – Introduction
2. Types of Triggers
3. BEFORE INSERT, AFTER INSERT
4. BEFORE UPDATE, AFTER UPDATE
5. BEFORE DELETE, AFTER DELETE
6. Creating Triggers
7. Using Triggers for Auditing/Logging
8. Nested and Recursive Triggers
9. Disabling and Dropping Triggers
10. INSTEAD OF Triggers (SQL Server, etc.)
11. Performance Considerations with Triggers

Create one complete Full Stack web Application

The End