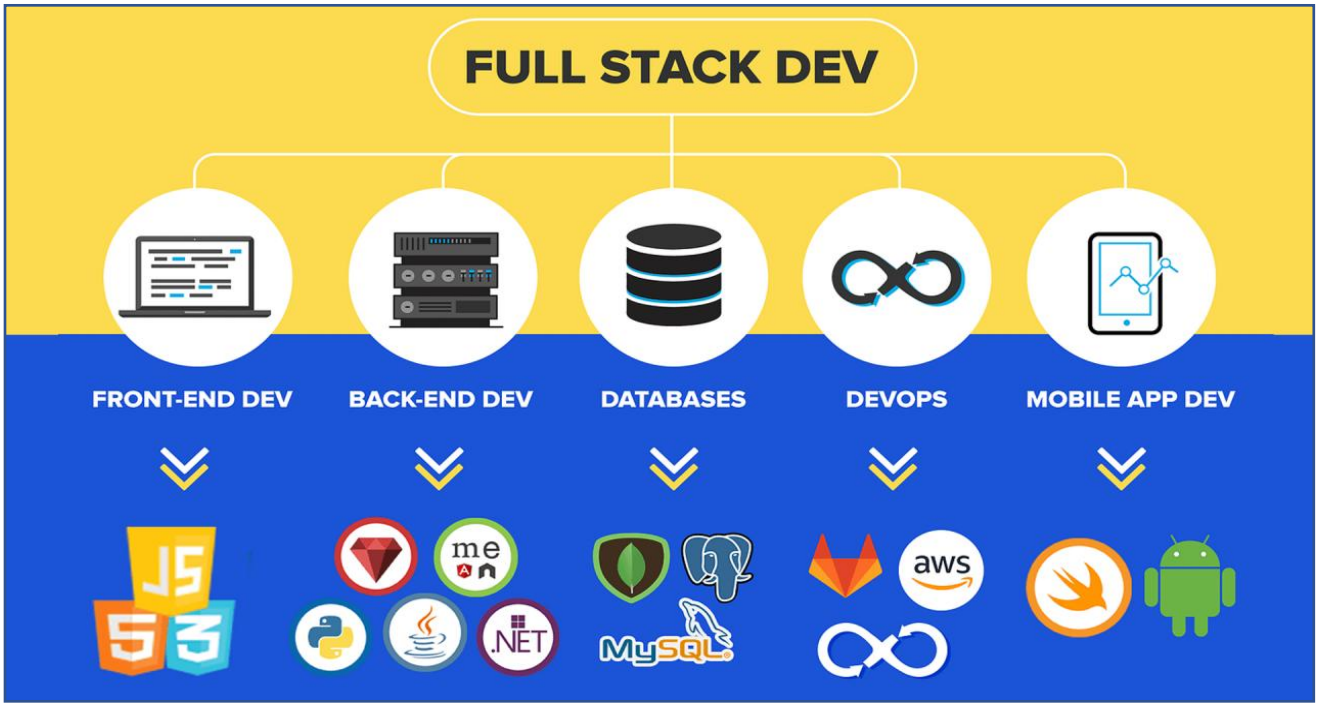
1. What is Full Stack Development?

Full Stack Development is a system of creating, developing, and sustaining each component of a web application, i.e., the front-end (what the user experiences) and the back-end (how the system processes under the hood), and the database where all data is stored and managed.

A full-stack developer has the ability to work at all the layers of this process, which means that they can develop an entire wide product entirely or be able to enhance a previously developed product without depending fully on disjointed specialists in different sections.

Full-stack developers are desirable as they can combine the aesthetic and functional code and make it communicate well with other layers of the program.



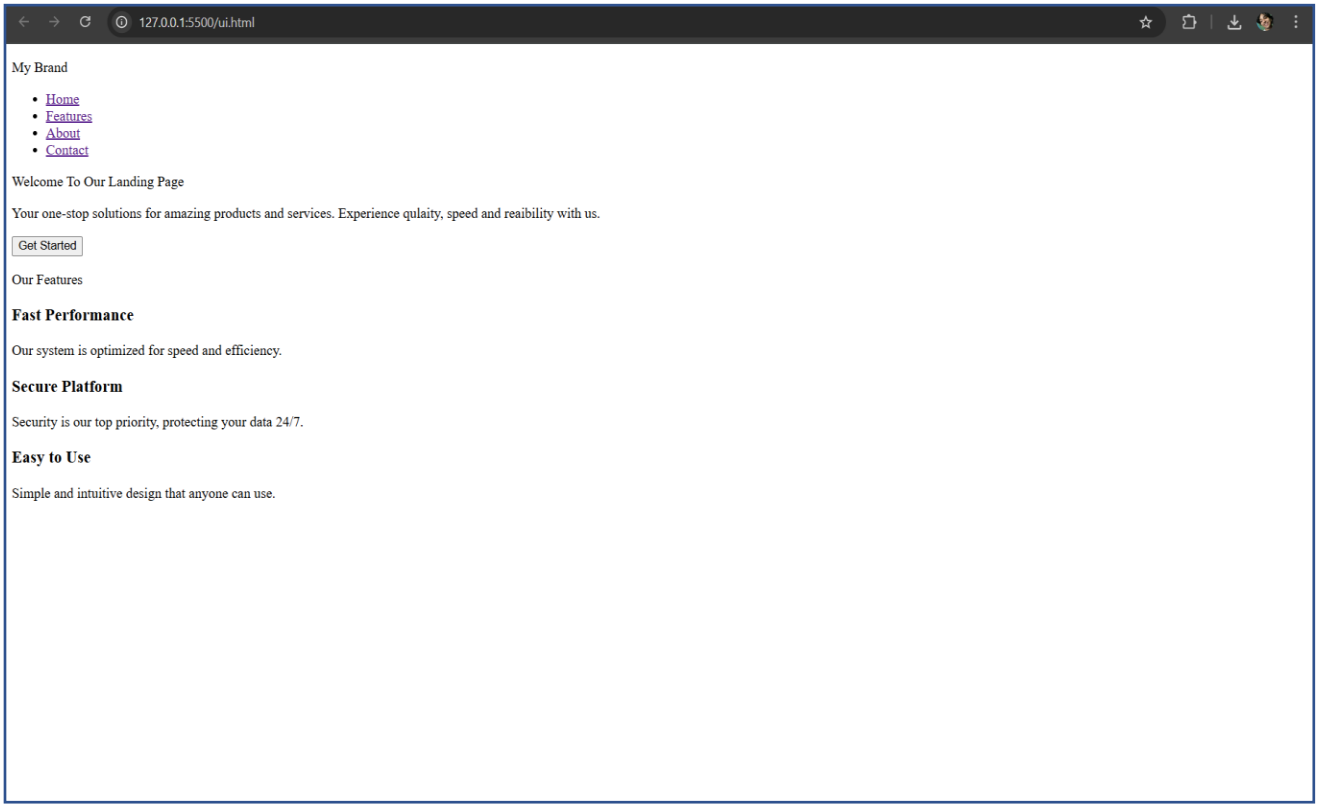
2. Layers of Full Stack Development

a) Front-End (Client Side)

- The presentation layer comprises the front-end, i.e., the bit of a website or an application that users are actually using. It aims at providing a clear, appealing, and convenient experience. The key technologies are given below:

1. Hypertext Markup Language (HTML): Think of it as the skeleton of a website. It defines headings, paragraphs, buttons, etc. An example is given below in the image:

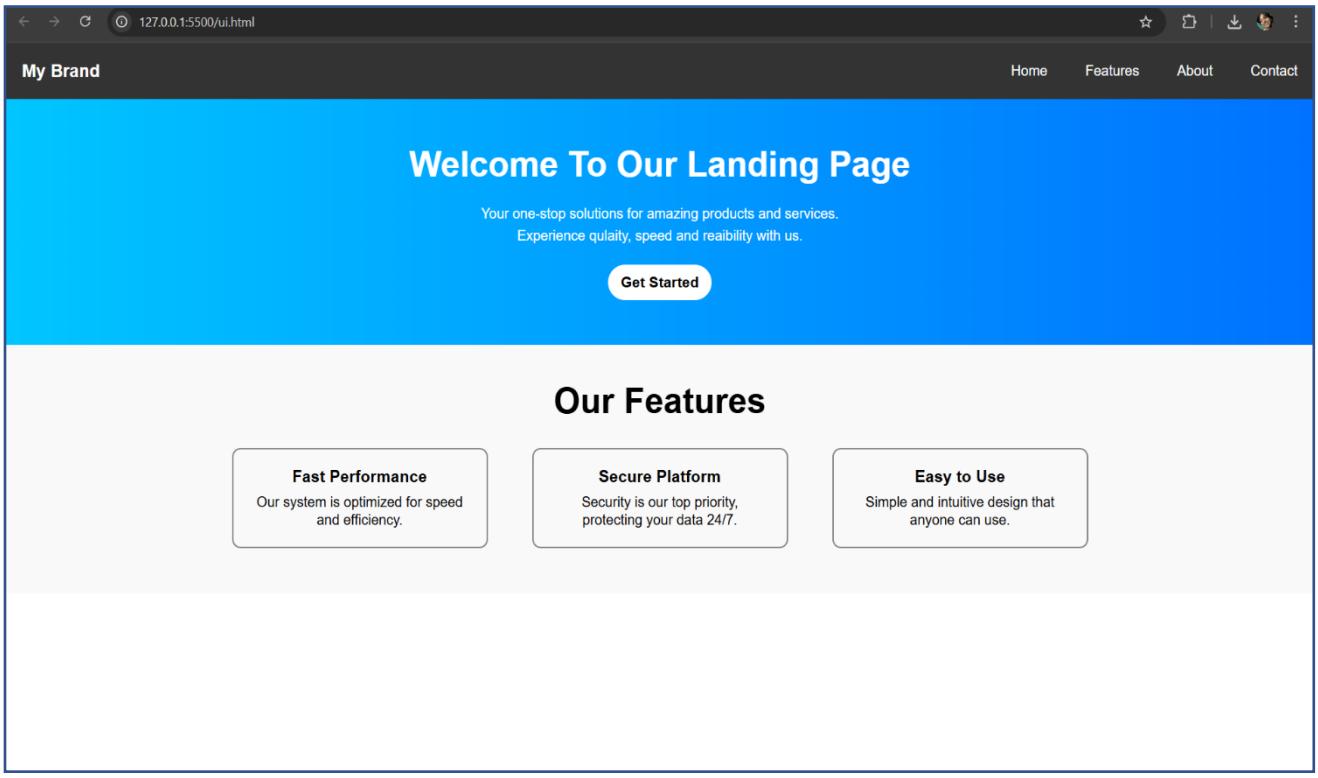
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>UI Test</title>
    <link rel="stylesheet" href="">
</head>
<body>
    <header>
        <p class="brand_name">My Brand</p>
        <nav>
            <ul class="link">
                <li><a href="">Home</a></li>
                <li><a href="">Features</a></li>
                <li><a href="">About</a></li>
                <li><a href="">Contact</a></li>
            </ul>
        </nav>
    </header>

    <div class="container1">
        <p class="p1">Welcome To Our Landing Page</p>
        <p class="p2">Your one-stop solutions for amazing products and services. Experience qulaity, s
        <button class="but1">Get Started</button>
    </div>

    <div class="features">
        <p class="h1">Our Features</p>
        <div class="box">
            <div class="Box">
                <h3>Fast Performance</h3>
                <p>Our system is optimized for speed and efficiency.</p>
```



2. Cascading Style Sheet (CSS): It makes the website look good. Colors, layouts, spacing, and fonts – all can be done with CSS. An example is given below:

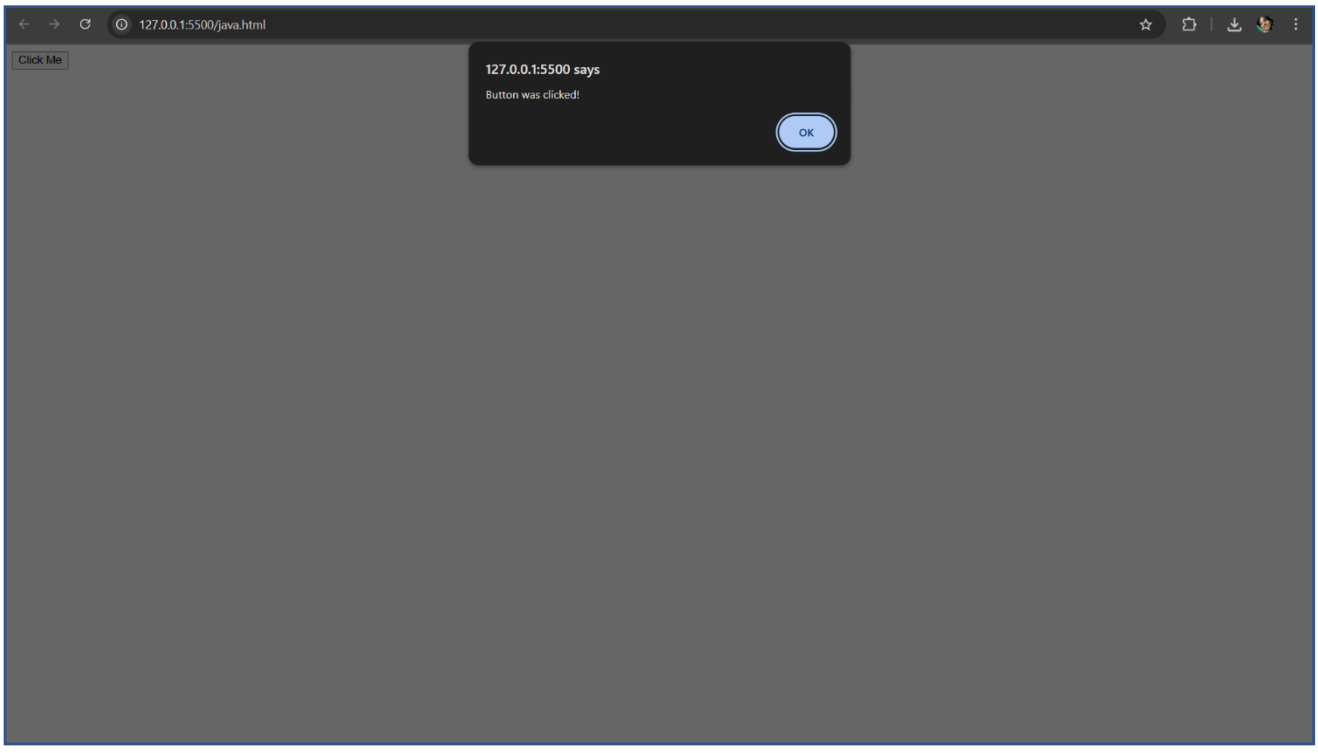
ui.css

```css
*{
    box-sizing: border-box;
    margin: 0px 0px;
    padding: 0px 0px;
    font-family: Arial, Helvetica, sans-serif;
}

header{
    background-color: #333;
    display: flex;
    align-items: center;
    justify-content: space-between;
}
header .brand_name{
    font-size: 20px;
    color: white;
    font-weight: bold;
    padding: 20px 20px;
}
.link{
    list-style: none;
}
.link li{
    display: inline-block;
    padding: 0px 20px;
}
.link a{
    color: white;
    text-decoration: none;
}

.container1{
    background: linear-gradient(to right, #00c6ff, #0072ff);
```



3. JavaScript: It makes the site interactive. Click buttons, show alerts, dynamic forms, etc, can be done by using JS. An example is given below:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <button onclick="handleClick()">Click Me</button>

    <script>
        function handleClick() {
            alert('Button was clicked!');
        }
    </script>
</body>
</html>
```



b) Back-End (Server Side)

- Back-end is where the brains and processing are done, and so is data handling. It gets requests from the user interface, handles them, and returns staff answers.

- The responsibilities of the back-end include:
  - Receiving the demands of a client.
  - Processing the gathered information.
  - Fetching, saving, or updating information in the database.
  - Application of business logic (rules that the application behaves).
  - Returning the correct response to the front-end.

- The commonly used technologies are given below:
  - Languages: Node.js (JavaScript), PHP, Python (Django, Flask), Java (Spring), Ruby on Rails.
  - Frameworks: Express.js (Node.js), Laravel (PHP), Django (Python).
  - APIs: RESTful APIs or GraphQL for structured communication between front-end and back-end.

c) Database Layer

- The database holds the information used by the application, stores the information, and makes it possible to retrieve and update the information, e.g., the information of a user account or product, or messages.

- Types of databases:
  - Relational Databases (SQL) – Data is stored in structured tables with predefined relationships. Examples: MySQL, PostgreSQL, Oracle.
  - NoSQL Databases – Store data in flexible formats such as documents or key-value pairs. Examples: MongoDB, Firebase, CouchDB.

- Database tasks in full-stack development include:
  - Designing the database structure (schema).
  - Writing queries to retrieve or update data.
  - Ensuring security and preventing unauthorized access.

3. How do the Layers work together?

- Simple examples of full-stack workflow are as follows:
  - User Action (Front-End): A visitor on the site clicks on the registration button.
  - Request sent: The browser sends an HTTP request, including the form information, to the back-end server.

- Processing (Back-End): The server checks the input, business rules, and generates instructions to the database.
- Database Interaction: New details of the user will be added to the database by the server.
- Response Sent Back: The server returns a confirmation message to the front-end.
- Update UI: The UI on the front-end updates to show a message stating that the registration went through successfully without loading the entire page.

4. Advantages of Full Stack Development

➢ The advantages of Full Stack Development are listed below:

- Versatility: Full-stack developers are able to go both front-end and back-end, giving them insight into every aspect of a project and enabling them to contribute to it.
- Cost-Effectiveness: Companies can save money when hiring a full-stack developer, since they can perform various tasks, and no individual experts need to be employed.
- Adaptability: Ability to change projects/tasks easily, and even with a change in project demand, they become efficient in fluid development activities.
- Career Opportunities: Full-stack skills are in high demand, providing developers with a wide range of job opportunities and the potential for career growth.

5. Full Stack Workflow