SUNSTONE

Module 3: Front End Development Framework & Tools

Frontend Development

Topics Covered

- Web Client Server Architecture
- HTML Structure Tags
- Bootstrap
- jQuery-ajax-promises
- API Integration

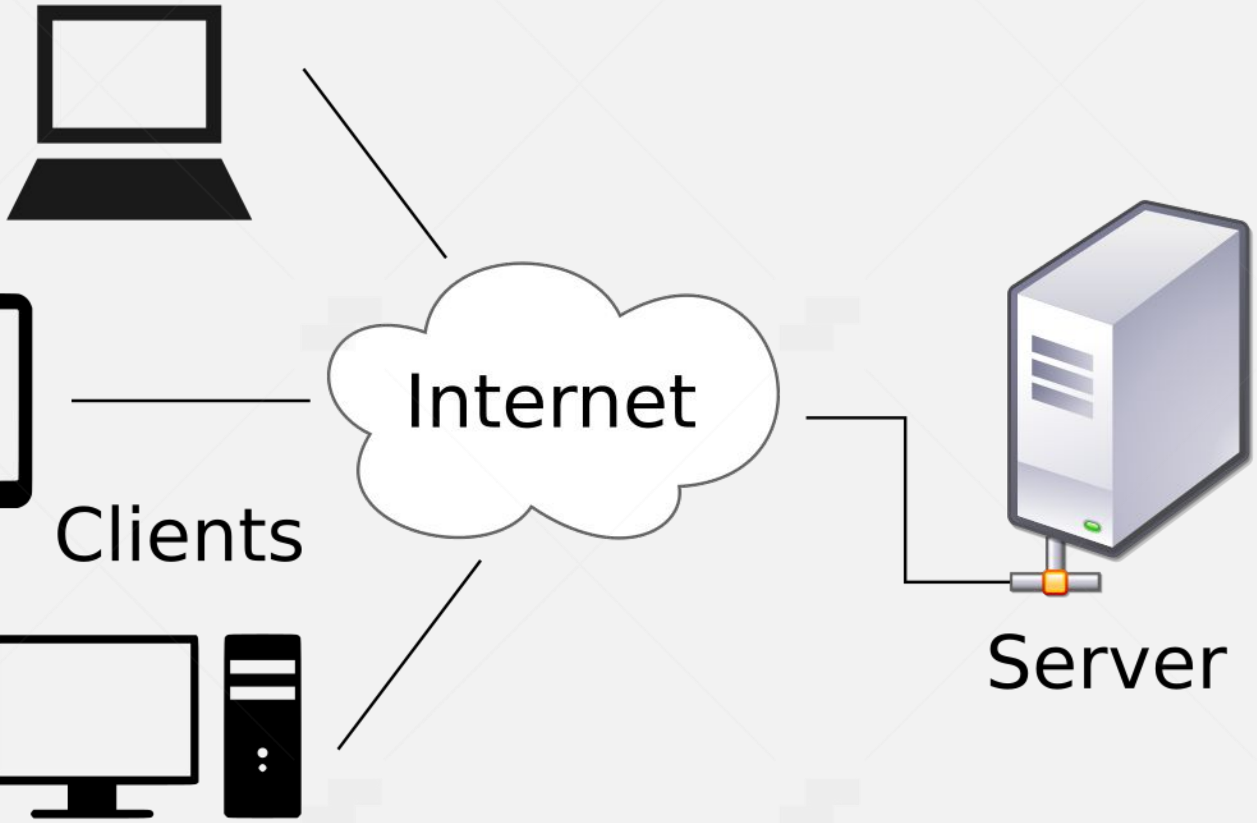
Client Server Architecture

- In a client-server architecture, every process or computer in a network functions as a server or client. Client servers are potent computers whose sole purpose is to handle printers, disk drives, and network traffic. The clients run their applications on their workstations or personal computers. The servers primarily provide resources, such as devices, files, and processing power.

- But Before understanding better client-server architecture, let's first understand what this client and server are all about.

- Client – The client can be any computer that requests something from the server. For example – visiting any website we request the webpage from its domain. So here we act as a client.

- Server – On the other hand, the Server is the computer that is designed to serve the requests to the client. For the same example as above, the client asks for the webpage then the server responds with the webpage to the client.


Internet
Clients
Server

- The client-server architecture or model is an application network separating tasks between clients and servers that are either within the same system or need to communicate over a network. In order to access the service provided by the server, the server-client sends the request to another program, which runs a few programs that distribute work among the clients & share resources with them.

- A client-server relationship corresponds to the request-response pattern and should adhere to a standard communications protocol that defines the language and rules used for the communications. The client-server communication adheres to the TCP protocol suite.

- Client/server messages are exchanged via the TCP protocol until the connection is complete. TCP protocol determines how data should be distributed in packets that networks will deliver, transfers packets to and receives packets from networks, and manages flow control or retransmission of garbled and dropped packets.

- The Internet Protocol is a connectionless protocol, which means that each packet traveling on the Internet is an independent piece of data, unconnected to any other packet.

How the browser interacts with the servers ?

There are few steps to follow to interacts with the servers a client.

- User enters the URL(Uniform Resource Locator) of the website or file. The Browser then requests the DNS(DOMAIN NAME SYSTEM) Server.
- DNS Server lookup for the address of the WEB Server.
- DNS Server responds with the IP address of the WEB Server.
- Browser sends over an HTTP/HTTPS request to WEB Server's IP (provided by DNS server).
- Server sends over the necessary files of the website.
- Browser then renders the files and the website is displayed. This rendering is done with the help of DOM (Document Object Model) interpreter, CSS interpreter and JS Engine collectively known as the JIT or (Just in Time) Compilers.

Advantages of Client-Server Architecture

1. Management is Easy – It is rather easy to manage the files because they are all stored on one server. Client-server networks have the best management to track and find records of required files.
2. Easily Accessible – Clients can log into the system regardless of their location or their platform of choice, allowing each employee to access their corporate information without having to use a terminal mode or a processor.
3. Servers are Scalable – A client-server network is highly scalable. Whenever the user needs, they can add more resources such as clients and servers, thus increasing the size of the server without any interruption. Due to the fact that the server is centralized, permission to network resources is not an issue, so very few staff members are required for configurations even if the size increases.
4. Centralized Control – Client-server networks have the advantage of centralized control since all information is stored in a single location. Since the network administrator has full control over management and administration, this is especially beneficial. As a result, any problems that arise throughout the entire network can be solved from one central location. As a result, it has also become much easier to update data and resources.
5. Security – Due to its centralized architecture, client-server networks protect data well. A single backup can be used to recover all of the files if the data are lost, such as imposing credentials like username and passwords. Another method of enforcing access controls is imposing credentials like username and password.

Disadvantages of Client-Server Architecture

1. Less Robust – Due to client-server networks' centralized nature, in case the main server undergoes failure or interference, the entire network will be interrupted. Thus, client-server networks are less robust.

1. Requires Regular Maintenance – Servers will run continuously on the network. This means they must be properly maintained. If there are any problems, they should be fixed immediately. A network manager should be appointed to maintain the server.

1. Requires Cost – In a client-server network, the cost of setting up and maintaining the server is usually very high, just as it is on the network operations. Because the networks are powerful, they can be expensive to purchase. Thus, not all users will be able to take advantage of them.

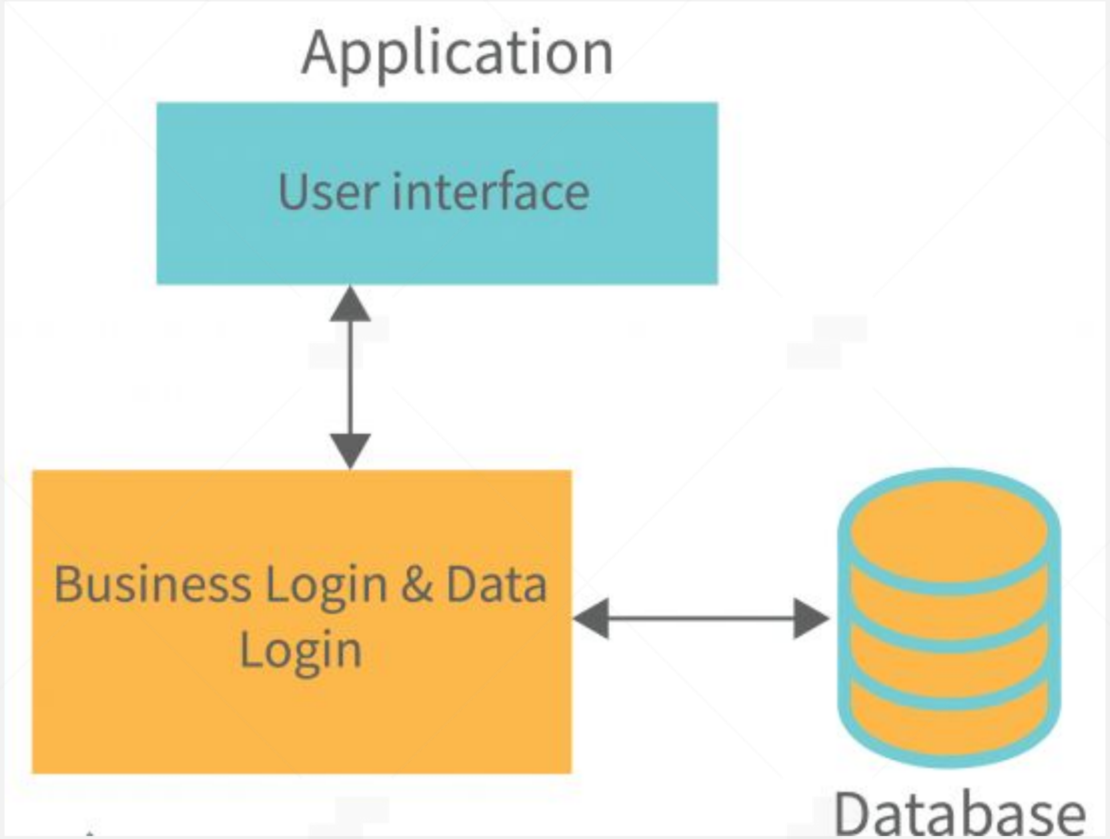
1. Network Congestion – If too many clients access a single server, it may result in crashes or slowed-down connectivity. An overloaded server presents many problems when accessing information.

Types of Client-Server Architecture

1-Tier Architecture

In 1-Tier Client Server Architecture, Everything related to the application is grouped and used as a single package to deliver the application. All the logic related to the User Interface, Business Logic, Database Logic, and Database are grouped to make a single entity.

A 1-tier architecture offers various services that make it a reliable source, but it is complex to manage. The data variance is the primary problem. Work is often duplicated. A 1-tier architecture involves several layers, including the presentation layer, business layer, and data layer, which are combined using a unique software package. This layer usually stores data within local systems or on a shared drive.

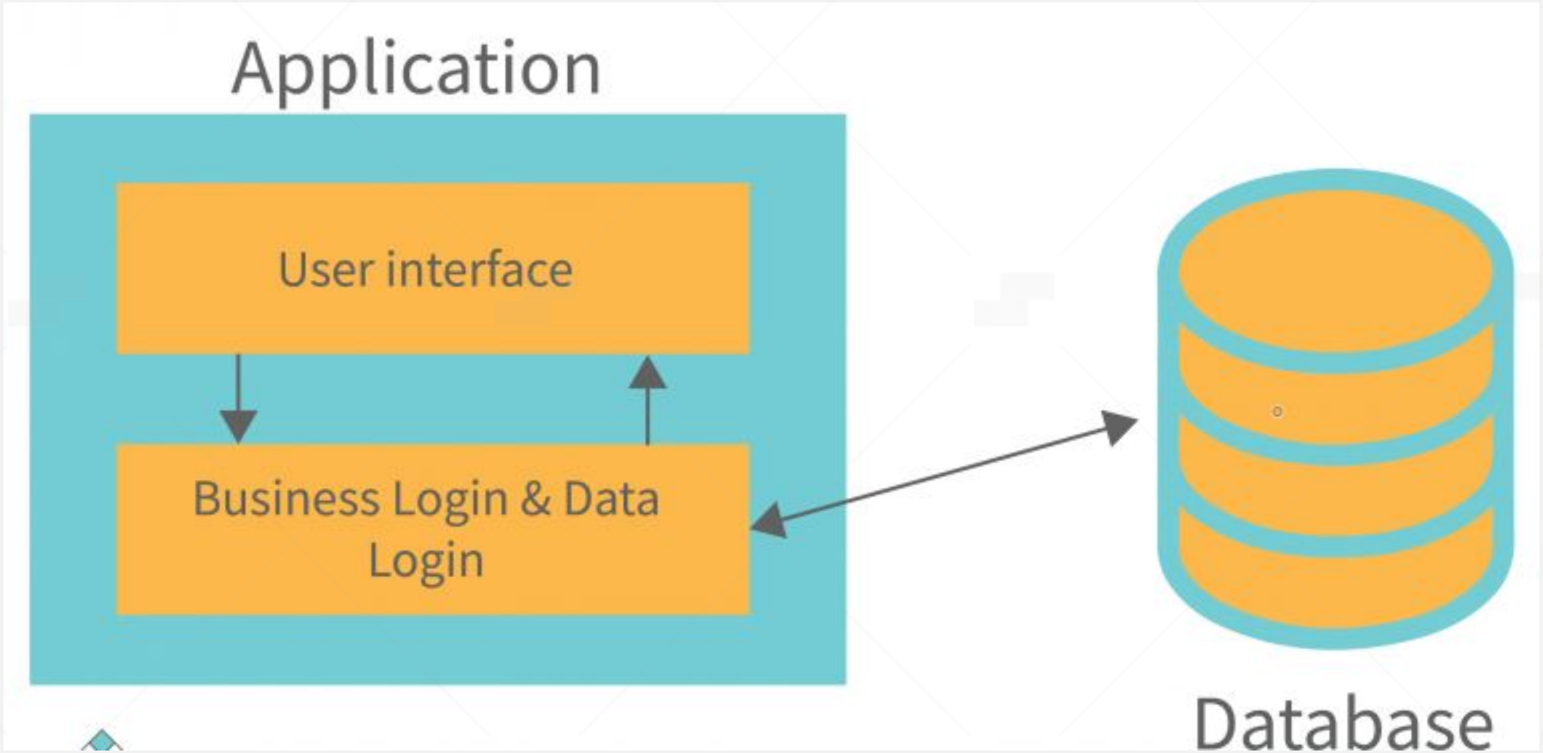

Application
User interface
Business Login & Data Login
Database

Types of Client-Server Architecture

2-Tier Architecture

In 2-tier Client Server Architecture, the whole application logic is divided into 2 layers. Majorly database acts as a separate entity in this architecture. Databases are designed separately and then the main application has all the logic related to the user interface and the business logic and database logic to communicate with the database and process the applications. This architecture has a better environment than the 1-tier architecture.

In comparison to 1-tier architectures, 2-tier architectures are faster since there is no intermediary between the client and the server. The 2-tier architecture helps avoid client confusion. The online reservation system is a popular example of a 2-tier architecture.

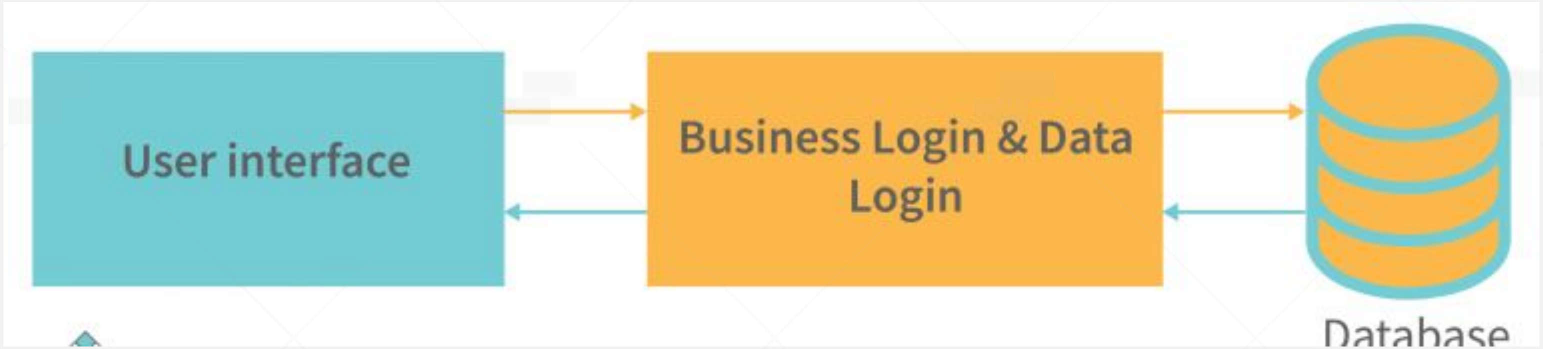

Application
User interface
Business Login & Data Login
Database

Types of Client-Server Architecture

3-Tier Architecture

In contrast to the architecture of a 2-tier system, which has no middleware, a 3-tier system has a middleware between the client and the server. When a client requests information from the server, the request is first received by the middleware. The request will then be sent to the server for processing. Similarly, the server will send a response to the client.

In 3-tier architecture, there are three main layers: presentation layer, application layer, and database layer. Different ends of each layer control each other. Client devices control the presentation layer, whereas the middleware and server control the application layer and database layer, respectively. The existence of a third layer that provides data control makes the 3-tier architecture more secure, has invisible database structures, and ensures data integrity.


User interface
Business Login & Data Login
Database

Server-side Programming :

It is the program that runs on server dealing with the generation of content of web page.

1) Querying the database

2) Operations over databases

3) Access/Write a file on server.

4) Interact with other servers.

5) Structure web applications.

6) Process user input. For example if user input is a text in search box, run a search algorithm on data stored on server and send the results.

Examples :

The Programming languages for server-side programming are :

1) PHP

2) C++

3) Java and JSP

4) Python

5) Ruby on Rails

Client-side Programming :

It is the program that runs on the client machine (browser) and deals with the user interface/display and any other processing that can happen on client machine like reading/writing cookies.

1) Interact with temporary storage

2) Make interactive web pages

3) Interact with local storage

4) Sending request for data to server

5) Send request to server

6) work as an interface between server and user

The Programming languages for client-side programming are :

1) Javascript

2) VBScript

3) HTML

4) CSS

5) AJAX

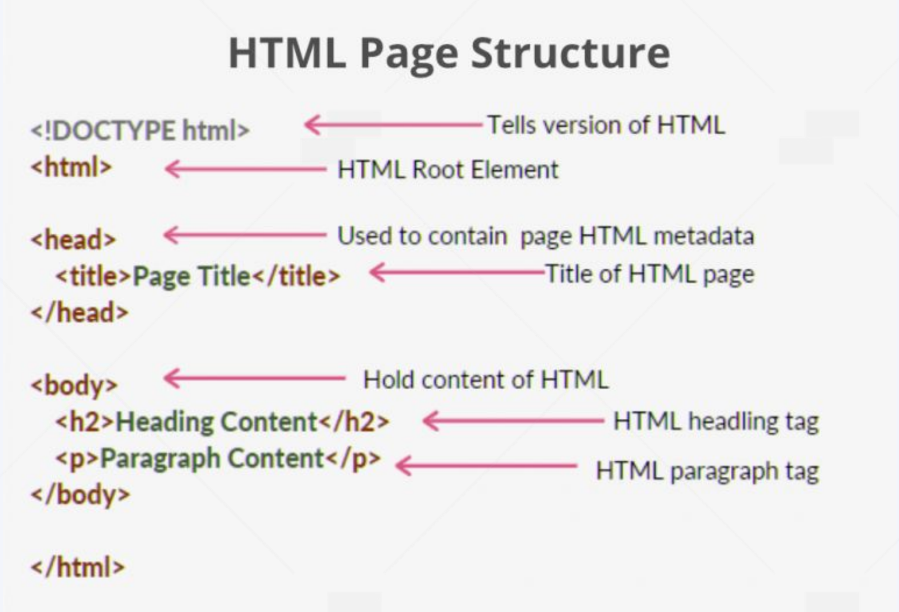
HTML Tags

- **HTML** stands for HyperText Markup Language. It is used to design web pages using a markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between web pages. A markup language is used to define the text document within the tag which defines the structure of web pages. This language is used to annotate (make notes for the computer) text so that a machine can understand it and manipulate text accordingly. Most markup languages (e.g. HTML) are human-readable. The language uses tags to define what manipulation has to be done on the text.

- **Elements and Tags:** HTML uses predefined tags and elements which tell the browser how to properly display the content. Remember to include closing tags. If omitted, the browser applies the effect of the opening tag until the end of the page.

HTML page structure:

The basic structure of an HTML page is laid out below. It contains the essential building-block elements (i.e. doctype declaration, HTML, head, title, and body elements) upon which all web pages are created.



```
<!DOCTYPE html>
<html>

<head>
  <title>Page Title</title>
</head>

<body>
  <h2>Heading Content</h2>
  <p>Paragraph Content</p>
</body>

</html>
```

<!DOCTYPE html>: This is the document type declaration (not technically a tag). It declares a document as being an HTML document. The doctype declaration is not case-sensitive.

<html>: This is called the HTML root element. All other elements are contained within it.

<head>: The head tag contains the “behind the scenes” elements for a webpage. Elements within the head aren’t visible on the front-end of a webpage. HTML elements used inside the <head> element include:

<style>-This html tag allows us to insert styling into our webpages and make them appealing to look at with the help of CSS.

<title>-The title is what is displayed on the top of your browser when you visit a website and contains title of the webpage that you are viewing.

<base>-It specifies the base URL for all relative URL’s in a document.

<noscript>– Defines a section of HTML that is inserted when the scripting has been turned off in the users browser.

<script>-This tag is used to add functionality in the website with the help of JavaScript.

<meta>-This tag encloses the meta data of the website that must be loaded every time the website is visited. For eg:- the metadata charset allows you to use the standard UTF-8 encoding in your website. This in turn allows the users to view your webpage in the language of their choice. It is a self closing tag.

<link>– The ‘link’ tag is used to tie together HTML, CSS and JavaScript. It is self closing.

<body>: The body tag is used to enclose all the visible content of a webpage. In other words, the body content is what the browser will show on the front-end.

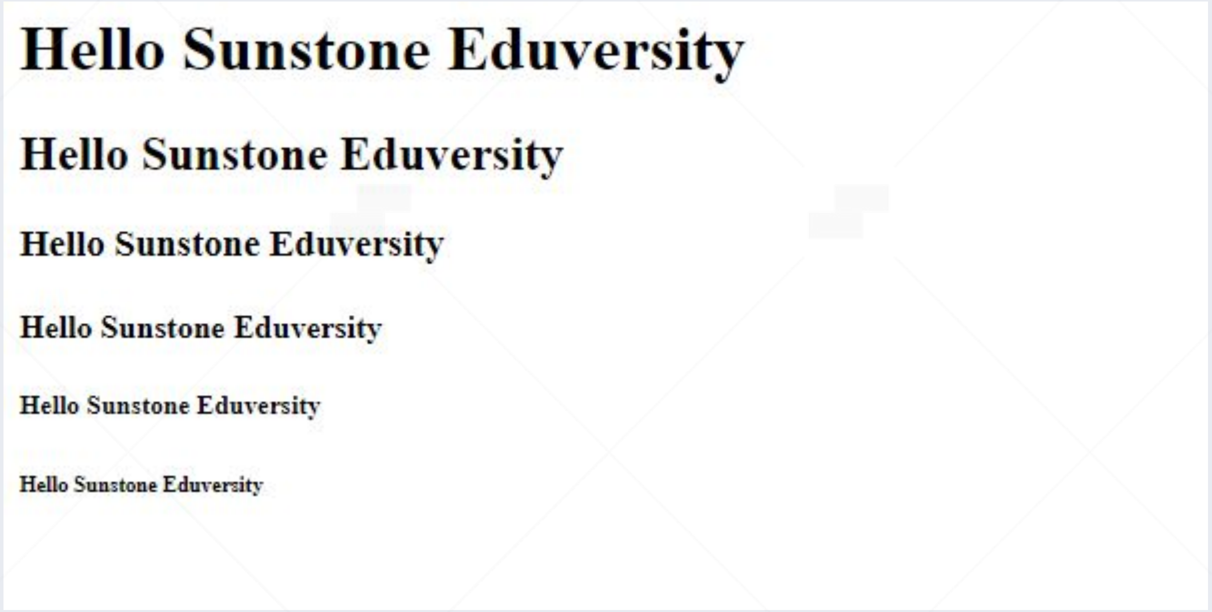
An HTML document can be created using any text editor. Save the text file using .html or .htm. Once saved as an HTML document, the file can be opened as a webpage in the browser.

Note: Basic/built-in text editors are Notepad (Windows) and TextEdit (Macs). Basic text editors are entirely sufficient for when you're just getting started. As you progress, there are many feature-rich text editors available which allow for greater function and flexibility.

Example: This example illustrates the basic structure of HTML code.

```html
<!DOCTYPE html>

<html>

<head>

      <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!--The above meta characteristics make a website compatible with different devices. -->

    <title>Demo Web Page</title>
```

```
</head>


<body>

  <h1>Sunstone Eduversity</h1>



<p>Best Job Placement Assistance</p>


</body>

</html>
```



```
Hello Sunstone Eduversity
Hello Sunstone Eduversity
Hello Sunstone Eduversity
Hello Sunstone Eduversity
Hello Sunstone Eduversity
Hello Sunstone Eduversity
```

Characteristics of HTML:

- **Easy to understand:** It is the most straightforward language you can say, very easy to grasp this language and easy to develop.
- **Flexibility:** This language is so much flexible that you can create whatever you want, a flexible way to design web pages along with the text.
- **Linkable:** You can make linkable text like users can connect from one page to another page or website through these characteristics.
- **Limitless features:** You can add videos, GIFs, pictures, or sound anything you want that will make the website more attractive and understandable.
- **Support:** You can use this language to display the documents on any platform like Windows, Linux, or Mac.
- **Not a Programming Language:** HTML is not a programming language as it is only concerned with presenting the information on the web. It is not used to program any logic but to give structure and semantically meaning to our website. Though we can link JavaScript code to it which is a programming language.
- **Language Support:** HTML can support various other languages like JavaScript, Ruby, PHP, Perl, and many more. You can also able to run embed python during the runtime.

Features of HTML:

- It is easy to learn and easy to use.
- It is platform-independent.
- Images, videos, and audio can be added to a web page.
- Hypertext can be added to the text.
- It is a markup language.

Why learn HTML?

- It is a simple markup language. Its implementation is easy.
- It is used to create a website.
- Helps in developing fundamentals about web programming.
- Boost professional career.

Advantages:

- HTML is used to build websites.
- It is supported by all browsers.
- It can be integrated with other languages like CSS, JavaScript, etc.

Disadvantages:

- HTML can only create static web pages. For dynamic web pages, other languages have to be used.
- A large amount of code has to be written to create a simple web page.
- The security feature is not good.

HTML Basics

- **HTML Headings:** These tags help us to give headings to the content of a webpage. These tags are mainly written inside the body tag. HTML provides us with six heading tags from <h1> to <h6>. Every tag displays the heading in a different style and font size.

- Most HTML heading tag that we use :-

  Heading 1

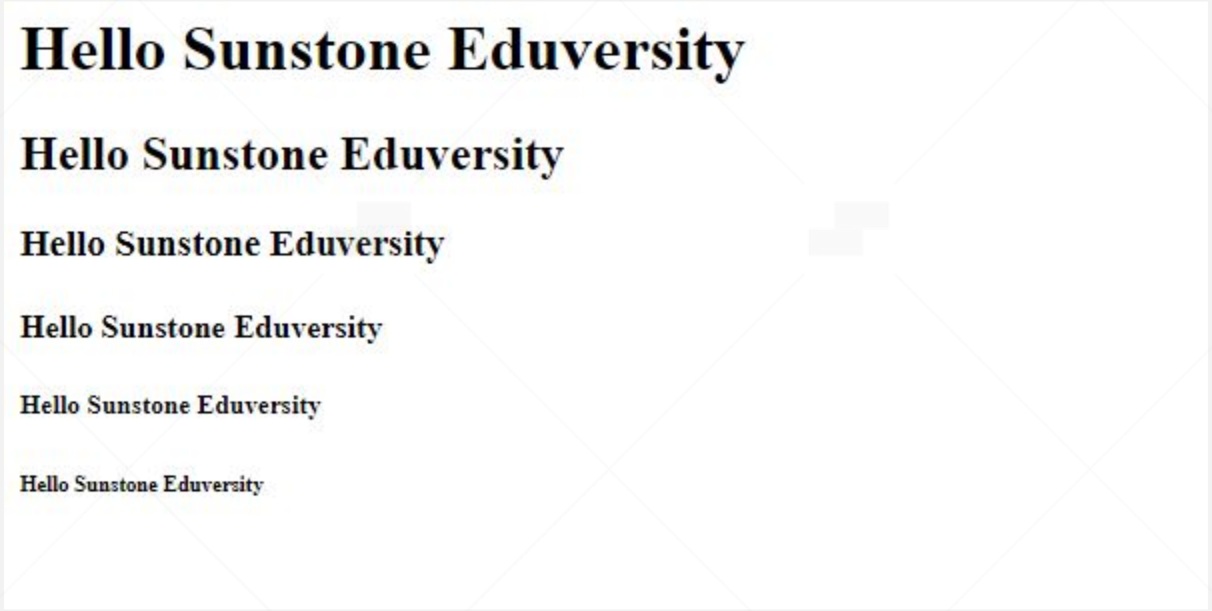
  Heading 2

  Heading 3

  Example: This example illustrates the use of 6 heading tags from <h1> to <h6> in HTML.

```
<html>

<head>

   <title>Sunstone Eduversity</title>

</head>


<body>

   <h1>Hello Sunstone Eduversity</h1>

   <h2>Hello Sunstone Eduversity</h2>
```

```
 <h3>Hello Sunstone Eduversity</h3>

    <h4>Hello Sunstone Eduversity</h4>

    <h5>Hello Sunstone Eduversity</h5>

    <h6>Hello Sunstone Eduversity</h6>

</body>


</html>
```


Hello Sunstone Eduversity

Hello Sunstone Eduversity

Hello Sunstone Eduversity

Hello Sunstone Eduversity

Hello Sunstone Eduversity

Hello Sunstone Eduversity

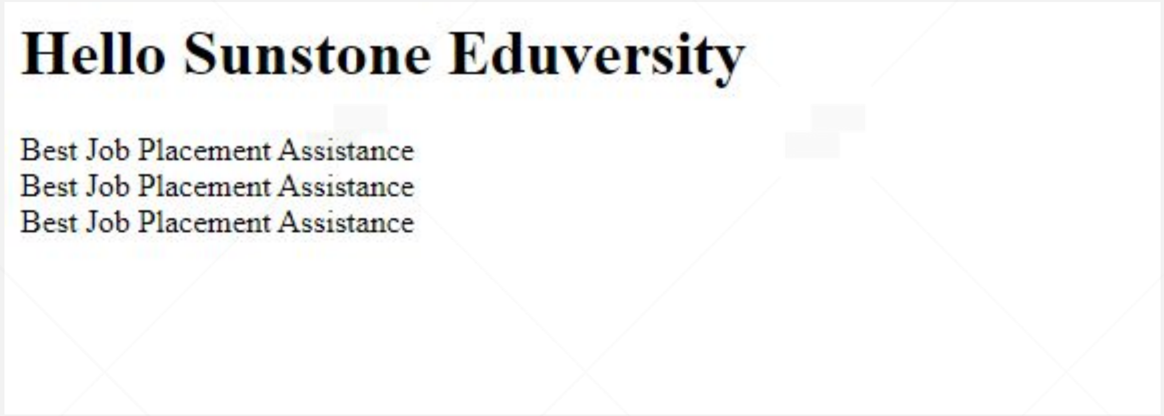
- **HTML Paragraph:** These tags help us to write paragraph statements on a webpage. They start with the <p> tag and ends with </p>.

- **HTML Break: –** These tags are used for inserting a single line type break. It does not have any closing tag. In HTML the break tag is written as <br>.

- Example: This example illustrates the use of the <p> tag for writing a paragraph statement in HTML.

```
<html>


<head>
    <title>Sunstone Eduversity</title>
</head>


<body>
    <h1>Hello Sunstone Eduversity</h1>
```

```
<p> Best Job Placement Assistance<br>

    Best Job Placement Assistance<br>

    Best Job Placement Assistance<br>

  </p>

</body>

</html>
```



```
Hello Sunstone Eduversity

Best Job Placement Assistance
Best Job Placement Assistance
Best Job Placement Assistance
```

- **HTML Horizontal Line:** The <hr> tag is used to break the page into various parts, creating horizontal margins with help of a horizontal line running from the left to right-hand side of the page. This is also an empty tag and doesn't take any additional statements.

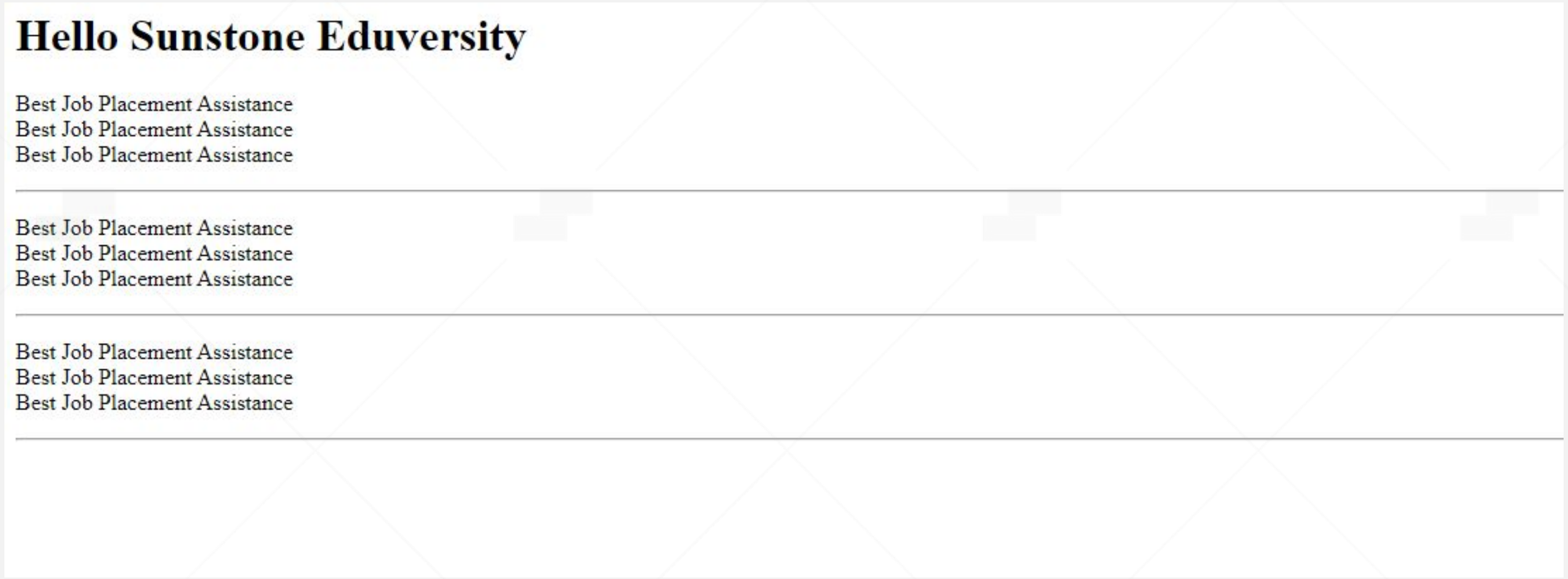
- Example: This example illustrates the use of the <hr> tag for the horizontal line in HTML.

```
<html>


<head>
   <title>Sunstone Eduversity</title>
</head>


<body>
   <h1>Hello Sunstone Eduversity</h1>
```

```
<p>

    Best Job Placement Assistance<br>

    Best Job Placement Assistance<br>

    Best Job Placement Assistance<br>

  </p>



 <hr>
```

```
<p>

    Best Job Placement Assistance<br>

    Best Job Placement Assistance<br>

    Best Job Placement Assistance<br>

  </p>




  <hr>
```

```
<p>

      Best Job Placement Assistance<br>

      Best Job Placement Assistance<br>

      Best Job Placement Assistance<br>

   </p>





   <hr>

</body>


</html>
```


Hello Sunstone Eduversity

Best Job Placement Assistance
Best Job Placement Assistance
Best Job Placement Assistance

Best Job Placement Assistance
Best Job Placement Assistance
Best Job Placement Assistance

Best Job Placement Assistance
Best Job Placement Assistance
Best Job Placement Assistance

- **HTML Images:** The image tag is used to insert an image into our web page. The source of the image to be inserted is put inside the <img src=”source_of_image“> tag.

- Image can be inserted in the image tag in two formats: –

    - If the image is in the same folder, then we can just write the name of the image and the format as the path.

    - If the image is in another folder, then we do need to mention the path of the image and the image name as well as the format of the image.

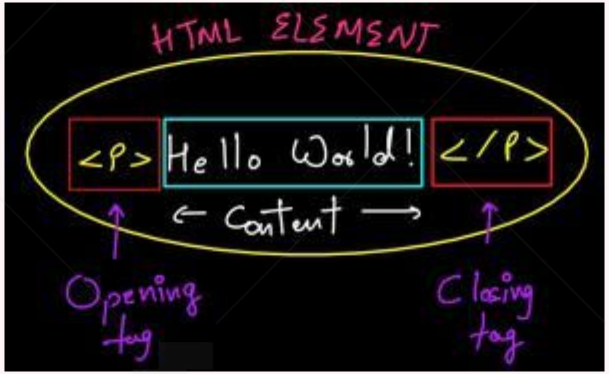
HTML Elements

- An HTML element is the collection of start and end tags with the content inserted in between them.
- HTML Element: The HTML element consist of 3 parts.

i) Opening tag: It is used to tell the browser where the content material starts.

ii)Closing tag: It is used to tell the browser where the content material ends.

iii)Content: It is the actual content material inside the opening and closing tag.

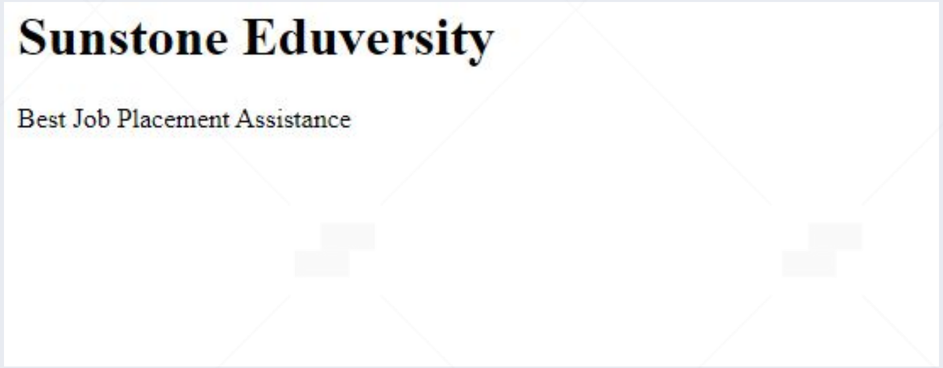


- **Nested HTML Elements:** The HTML element is use inside the another HTML Element is called nested HTML elements.

- Example 3: This example describes the use of the Nested HTML elements. Here, <html> tag contains the <head> and <body>. The <head> and <body> tag contains another elements so it is called nested element.

```
<!DOCTYPE html>

<html>

<head>

<title>HTML Elements</title>

</head>

<body style="text-align: center">
```

```
<h1>Sunstone Eduversity</h1>

<p>Best Job Placement Assistance</p>

</body>

</html>
```



Sunstone Eduversity

Best Job Placement Assistance

- **Necessary to add end tag:** It is necessary to add the end tag of an element. Otherwise, the displayed content may or may not be displayed correctly. It is a good practice if you add closing tags to the non-void HTML Elements but nowadays browsers are getting more and more advanced and forgiving in nature and that's why if you somehow forget to apply the closing tag in the non-void Element, the browser will not throw any error but the problem will arise as you insert more and more HTML elements after that.
- **Empty HTML Elements:** HTML Elements without any content i.e, that do not print anything are called Empty elements. Empty HTML elements do not have an ending tag. For instance. <br>, <hr>, <link>, <input> etc are HTML elements.

HTML Attributes

- All HTML elements have attributes that will provide additional information about that particular element. It takes 2 parameters, ie, a name & a value which define the properties of the element and are placed inside the element tag.
- The name parameter takes the name of the property we would like to assign to the element and the value takes the properties value or extent of the property names that can be aligned over the element. Every name has some value that must be written within quotes.
- Syntax:

  <element attribute_name="attribute_value">

- The below are some of the most commonly used Attributes in HTML.

HTML Attributes

- **HTML src Attribute:** If we want to insert an image into a webpage, then we need to use the <img> tag and the src attribute. We will need to specify the address of the image as the attribute's value inside the double quote.

```
<body>

   <img src=
"https://media.sunstone.in/wp-content/cdn-uploads/Geek_logi_-low_res.png">
</body>
```

- **HTML alt Attribute:** This is an alternate tag that is used to show or display something if the primary attribute i.e., the <img> tag, fails to display the value assigned to it. This can also be used to describe the image to a developer who is actually sitting at the coding end.

```
 <img src=

 "https://media.sunstone.in/wp-content/cdn-uploads/Geek_logi_-low_res.png"

        alt="The Logo"><br>

   <img src="" alt="Since the src value is blank,the alt value is displayed">
```

HTML Attributes

- **HTML width and height Attribute:** This attribute is used to adjust the width and height of an image

```
<body>
    <img src=
"https://media.sunstone.in/wp-content/cdn-uploads/Geek_logi_-low_res.png"
         width="300px" height="100px" >
</body>
```

- **HTML id Attribute:** This attribute is used to provide a unique identification to an element. Situations may arise when we will need to access a particular element that may have a similar name as the others. In that case, we provide different id’s to various elements so that they can be uniquely accessed. The properties extending the use of id are generally used in CSS, which we will be learning later.

```
<body>
    <h1 id="geeks">Welcome to Sunstone Eduversity</h1> </body>
```

HTML Attributes

- **HTML title Attribute:** The title attribute is used to explain an element on hovering the mouse over it. The behavior differs with various elements but generally, the value is displayed while loading or hovering the mouse pointer over it.

```
<body>

   <h3 title="Hello Sunstone Eduversity">Hover to see the effect</h3>

</body>
```

- **HTML href Attribute:** This attribute is used to specify a link to any address. This attribute is used along with the <a> tag. The link put inside the href attribute gets linked to the text displayed inside the<a> tag. On clicking on the text we will be redirected to the link. By default, the link gets opened in the same tag but by using the target attribute and setting its value to “_blank”, we will be redirected to another tab or another window based on the browser’s configuration.

```
<a href="https://www.sunstone.in/">

   </a><br>
```

HTML Attributes

- **HTML style Attribute:** This attribute is used to provide various CSS effects to the HTML elements such as increasing font-size, changing font-family, coloring, etc.

```
<body>

   <h2 style="font-family:Chaparral Pro Light;">Hello GeeksforGeeks.</h2>

   <h3 style="font-size:20px;">Hello GeeksforGeeks.</h3>

   <h2 style="color:#8CCEF9;">Hello GeeksforGeeks.</h2>

   <h2 style="text-align:center;">Hello GeeksforGeeks.</h2>

</body>
```

HTML Attributes

- **HTML lang attribute:** The language is declared with the lang attribute. Declaring a language is can be important for accessibility applications and search engines.

```
<!DOCTYPE html>

<html lang="en-US">

<body>



...



</body>

</html>
```

CSS- Bootstrap

- Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. Nowadays, the websites are perfect for all browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers – Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

CSS- Bootstrap

- By using this framework we can easily manipulate the styling of any web page, like font style, text color, background color, flex, grid system, etc. Bootstrap Vesrion 4 & Vesrion 5 are the most popular versions. There are lots of other CSS frameworks like Tailwind CSS, Bulma, and Foundation but among them, this framework is the most popular because of below mentioned features:
  - It is Faster and Easier way for Web-Development.
  - It creates Platform-independent web-pages.
  - It creates Responsive Web-pages.
  - It designs responsive web pages for mobile devices too.
  - It is a free and open-source framework available on www.getbootstrap.com

How to use Bootstrap on the webpage:

- There are two ways to include Bootstrap in the website.

1. Include Bootstrap through CDN links

1. Download Bootstrap from getbootstrap.com and use it

- **Include Bootstrap through CDN links:**

```
<!– CSS library –>

<link rel=”stylesheet” href=”https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css”
integrity=”sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T”
crossorigin=”anonymous”>



<!– jQuery library –>

<script src=”https://code.jquery.com/jquery-3.3.1.slim.min.js”
integrity=”sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo”
crossorigin=”anonymous”></script>
```

```
<!– JavaScript library –>

<script src=”https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js”
integrity=”sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1″
crossorigin=”anonymous”></script>



<!– Latest compiled JavaScript library –>

<script src=”https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js”
integrity=”sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM”
crossorigin=”anonymous”></script>
```

- **Example:**

```
<!DOCTYPE html>

<html lang="en">



<head>

  <meta charset="utf-8">

  <meta name="viewport"

     content="width=device-width, initial-scale=1">



+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"

     crossorigin="anonymous">
```

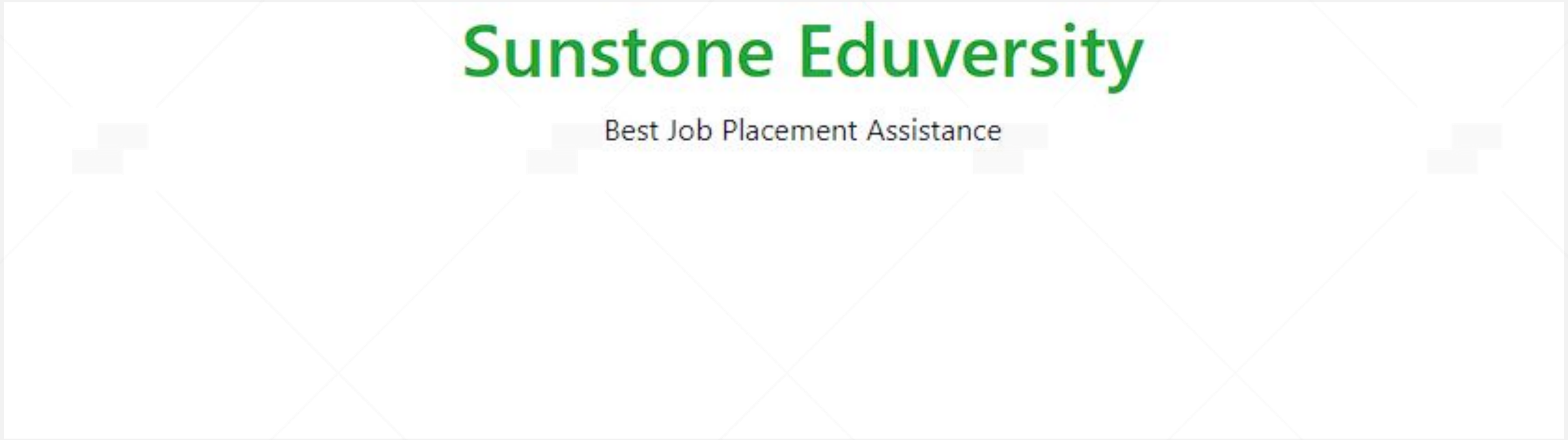
```
 <!-- Bootstrap CSS library -->

   <link rel="stylesheet"

      href=

"https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"

      integrity=

"sha384-ggOyR0iXCbMQv3Xipma34MD
```

```
  <!--jQuery library-->

  <script src=

"https://code.jquery.com/jquery-3.3.1.slim.min.js"

        integrity=

"sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"

        crossorigin="anonymous">

  </script>
```

```
    <!-- JavaScript library -->

    <script src=

  "https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"

          integrity=

  "sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"

          crossorigin="anonymous">

    </script>
```

```
 <!-- Latest compiled JavaScript library -->

   <script src=

"https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"

           integrity=

"sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"

           crossorigin="anonymous">

   </script>

</head>
```

```html
<body>

   <div class="container text-center">

      <!-- Text color class used -->

      <h1 class="text-success">Sunstone Eduversity</h1>

      <p>Sunstone Eduversity</p>

   </div>

</body>



</html>
```

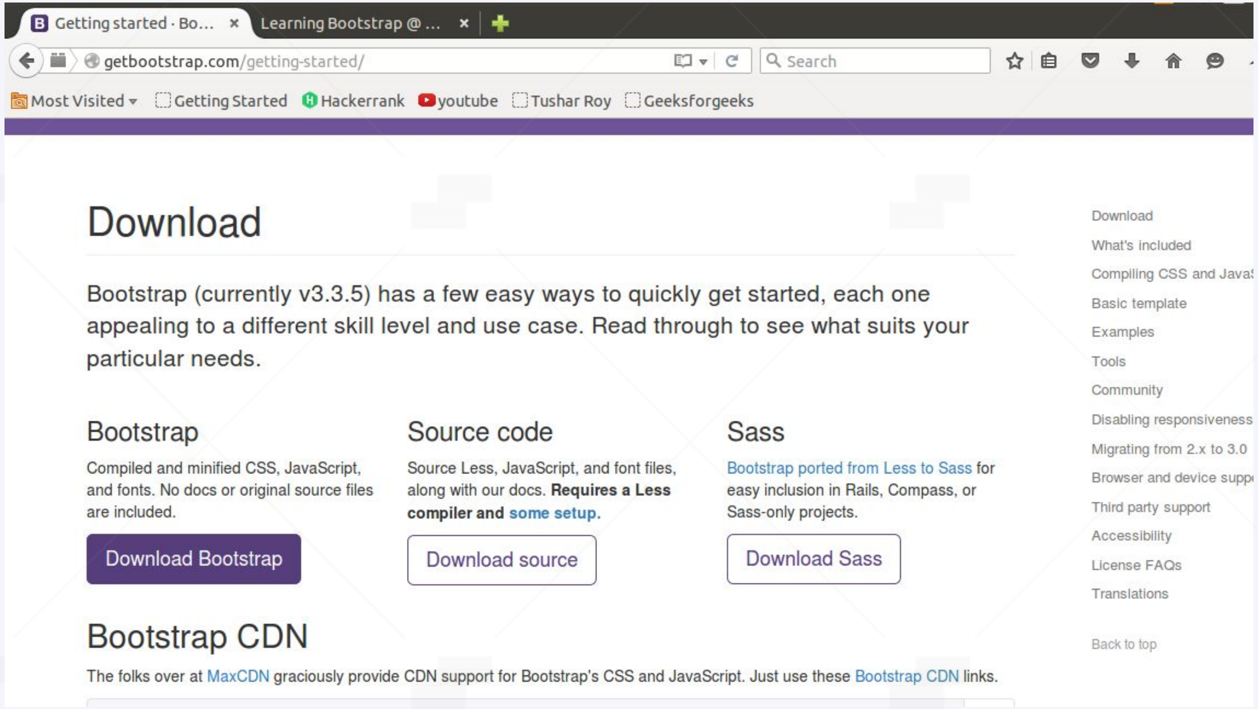


Sunstone Eduversity

Best Job Placement Assistance

Download Bootstrap from getbootstrap.com and use it:

Goto www.getbootstrap.com and click Getting Started. Click on the Download Bootstrap button.



Download

Bootstrap (currently v3.3.5) has a few easy ways to quickly get started, each one appealing to a different skill level and use case. Read through to see what suits your particular needs.

Bootstrap

Compiled and minified CSS, JavaScript, and fonts. No docs or original source files are included.

Download Bootstrap

Source code

Source Less, JavaScript, and font files, along with our docs. **Requires a Less compiler and** some setup.

Download source

Sass

Bootstrap ported from Less to Sass for easy inclusion in Rails, Compass, or Sass-only projects.

Download Sass

Bootstrap CDN

The folks over at MaxCDN graciously provide CDN support for Bootstrap's CSS and JavaScript. Just use these Bootstrap CDN links.

A.zip file would get downloaded. Extract the zip file and go to the distribution folder. It contains two folders named CSS and JS.

```
<link rel=”stylesheet” type=”text/css” href=”css/bootstrap.min.css”>

<script src=”js/bootstrap.min.js”> </script>

<script src=”https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js”></script>
```

Add the file link to the HTML document and then open the web page using web browsers.

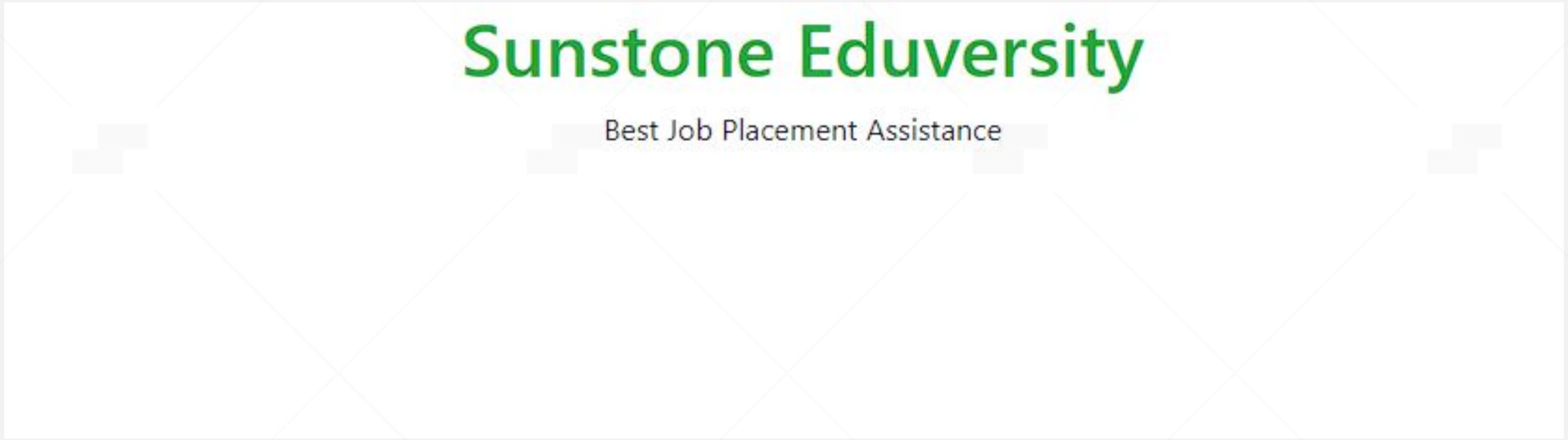
- **Example:**

```
<!DOCTYPE html>

<html lang="en">



<head>

  <meta charset="utf-8">

  <meta name="viewport"

      content="width=device-width, initial-scale=1">
```

```
<!-- Bootstrap CSS library -->
 <link rel="stylesheet"
     type="text/css"
     href="css/bootstrap.min.css">
```

```
<!-- JavaScript library -->

   <script src="js/bootstrap.min.js"></script>

</head>



<body>

   <div class="container text-center">
```

```
<!-- Text color class used -->
      <h1 class="text-success">Sunstone Eduversity</h1>
      <p>Best Job Placement Assistance</p>
   </div>
</body>


</html>
```



Sunstone Eduversity

Best Job Placement Assistance

JQuery

- jQuery is an open-source JavaScript library that simplifies the interactions between an HTML/CSS document, or more precisely the Document Object Model (DOM), and JavaScript. Elaborating the terms, it simplifies HTML document traversing and manipulation, browser event handling, DOM animations, Ajax interactions, and cross-browser JavaScript development.

Why jQuery?

Some of the key points which support the answer for why to use jQuery:

- It is incredibly popular, which is to say it has a large community of users and a healthy amount of contributors who participate as developers and evangelists.
- It normalizes the differences between web browsers so that you don't have to.
- It is intentionally a lightweight footprint with a simple yet clever plugin architecture.
- Its repository of plugins is vast and has seen steady growth since jQuery's release.
- Its API is fully documented, including inline code examples, which in the world of JavaScript libraries is a luxury. Heck, any documentation at all was a luxury for years.
- It is friendly, which is to say it provides helpful ways to avoid conflicts with other JavaScript libraries.

Advantages:

- Wide range of plug-ins that allows developers to create plug-ins on top of the JavaScript library.
- Large development community.
- It is a lot easier to use compared to standard javascript and other javascript libraries.
- It lets users develop Ajax templates with ease. Ajax enables a sleeker interface where actions can be performed on pages without requiring the entire page to be reloaded.
- Being Light weight and a powerful chaining capabilities makes it more strong.

Disadvantages:

- While jQuery has an impressive library in terms of quantity, depending on how much customization you require on your website. The functionality may be limited thus using raw javascript may be inevitable in some cases.

- The JQuery javascript file is required to run the commands, while the size of this file is relatively small (25-100KB depending on the server). It is still a strain on the client computer and maybe your web server as well if you intend to host the script on your own web server.

Using jQuery (JS) library on HTML page

There are several ways to start using jQuery on your web site.

- Use the Google-hosted/ Microsoft-hosted content delivery network (CDN) to include a version of jQuery.
- Download own version of jQuery from jQuery.com and host it on own server or local filesystem.
- Basic syntax for any jQuery function is:

  $(selector).action()

  A $ sign is to define/access jQuery

  A (selector) is to “query (or find)” HTML elements in html page

  A jQuery action() is the action to be performed on the selected element(s)

jQuery ajax()

The ajax() method in jQuery is used to perform an AJAX request or asynchronous HTTP request..

$.ajax({name:value, name:value, ... })

Parameters: The list of possible values are given below:

- **type:** It is used to specify the type of request.
- **url:** It is used to specify the URL to send the request to.
- **username:** It is used to specify a username to be used in an HTTP access authentication request.
- **xhr:** It is used for creating the XMLHttpRequest object.
- **async:** It's default value is true. It indicates whether the request should be handled asynchronous or not.
- **beforeSend(xhr):** It is a function which is to be run before the request is being sent.
- **dataType:** The data type expected of the server response.

- **error(xhr, status, error):** It is used to run if the request fails.
- **global:** It's default value is true. It is used to specify whether or not to trigger global AJAX event handles for the request.
- **ifModified:** It's default value is false. It is used to specify whether a request is only successful if the response has changed since the last request.
- **jsonp:** A string overriding the callback function in a jsonp request.
- **jsonpCallback:** It is used to specify a name for the callback function in a jsonp request.
- **cache:** It's default value is true. It indicates whether the browser should cache the requested pages.
- **complete(xhr, status):** It is a function which is to be run when the request is being finished.
- **contentType:** It's default value is: "application/x-www-form-urlencoded" and it is used when data send to the server.
- **context:** It is used to specify the "this" value for all AJAX related callback functions.
- **data:** It is used to specify data to be sent to the server.
- **dataFilter(data, type):** It is used to handle the raw response data of the XMLHttpRequest.
- **password:** It is used to specify a password to be used in an HTTP access authentication request.
- **processData:** It's default value is true. It is used to specify whether or not data sent with the request should be transformed into a query string.
- **scriptCharset:** It is used to specify the charset for the request.
- **success(result, status, xhr):** It is to be run when the request succeeds.
- **timeout:** It is the local timeout for the request. It measured in terms of milliseconds.
- **traditional:** It is used to specify whether or not to use the traditional style of param serialization.

- **Example 1: This example illustrates the ajax() method in jQuery.**

```
<!DOCTYPE html>

<html>



<head>

  <title>

    jQuery ajax() Method

  </title>



  <script src=

"https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">

  </script>
```

```
<script>
    $(document).ready(function() {
      $("li:parent").css("background-color", "green");
    });
  </script>
</head>


<body style="text-align:center;">


  <h1 style="color:darkblue">
    Sunstone Eduversity
  </h1>
```

```
<h2>jQuery ajax() Method</h2>

  <h3 id="h11"></h3>

  <button>Click</button>

  <!--Script to use ajax() method to
     add text content -->
  <script>
     $(document).ready(function(){
```

```
$("button").click(function(){

            $.ajax({url: "geeks_exp.txt", async: false,

                        success: function(result) {

               $("h11").html(result);

            }});

         });

      });

   </script>

</body>



</html>
```

jQuery | ajaxComplete() Method

- The ajaxComplete() method is used to specify function to be run when an AJAX request completes.

Syntax:

```
$(document).ajaxComplete(function(event, xhr, options))
```

Parameter:

event: It contains the event object.

xhr: It contains the XMLHttpRequest object.

options: It contains the used options in AJAX request.

jQuery | ajaxComplete() Method

- The ajaxComplete() method is used to specify function to be run when an AJAX request completes.

Syntax:

```
$(document).ajaxComplete(function(event, xhr, options))
```

Parameter:

event: It contains the event object.

xhr: It contains the XMLHttpRequest object.

options: It contains the used options in AJAX request.

Example-1:This example changes the content of < p > element, by taking the data from server. When the AJAX request completes, the page says AJAX request completes..

```
<!DOCTYPE html>

<html>

<head>

    <script src=

"https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">

    </script>

    <script>

        $(document).ready(function() {

            $(document).ajaxComplete(function() {

                alert(" AJAX request completes.");

            });
```

Example-1:This example changes the content of < p > element, by taking the data from server. When the AJAX request completes, the page says AJAX request completes..

```
<!DOCTYPE html>

<html>

<head>

    <script src=

"https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">

    </script>

    <script>

        $(document).ready(function() {

            $(document).ajaxComplete(function() {

                alert(" AJAX request completes.");

            });
$("button").click(function() {

        $("#paragraph").load("demo.txt");

      });

    });

  </script>


  <style>

    body {

      text-align: center;

    }

  </style>

</head>
```

```
<body>
   <div id="div_content">
      <h1 style="color: darkblue;">
      Sunstone Eduversity
    </h1>
      <p id="paragraph"
        style="font-size: 20px;">
         Best Job Placement Assistance
      </p>
   </div>
   <button>Change Content</button>
</body>
</html>
```

jQuery | ajaxError() Method

The ajaxError() method in jQuery is used to specify function to be run when an AJAX request fails.

Syntax:

```
$(document).ajaxError( function(event, xhr, options, exc) )
```

Parameter:: This method accepts single parameter function which is mandatory. This function accepts four parameters which are listed below:

event: This parameter holds the event object.

xhr: It holds the XMLHttpRequest object.

options: It contains the used options in AJAX request.

exc: It holds the JavaScript exception.

Example 1: This example changes the content of <p> element, by taking the data from server. When the AJAX request fails due to an error, the page says AJAX request fail<!DOCTYPE html>

```
<html>

    <head>

        <script src=

"https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">

        </script>


        <!--Script to use ajaxError() method-->

        <script>

            $(document).ready(function() {

                $(document).ajaxError(function() {

                    alert("AJAX request fails.");

                });s..
$("button").click(function() {

            $("#paragraph").load("demo.txt");

          });

        });

      </script>

    </head>



    <body style="text-align:center;">



      <div id="div_content">



        <h1 style = "color: green;">
```

```
Sunstone Eduversity

        </h1>



        <p id = "paragraph" style= "font-size: 20px;">

         Best Job Placement Assistance

        </p>

      </div>



      <button>

        Change Content

      </button>

    </body>

  </html>
```

jQuery | ajaxSend() Method

The ajaxSend() method in jQuery is used to specify the function to be run when an AJAX request is about to send.

Syntax:

```
$(document).ajaxSend( function(event, xhr, options) )
```

Parameters: This method accepts single parameter function which is mandatory. The function accepts three parameters as mentioned above and described below:

event: It holds the event object.

xhr: It holds the XMLHttpRequest object.

options: It holds the used options in AJAX request.

Example 1: This example changes the content of <p> element, by taking the data from server. When the AJAX request is ready to send, the page says AJAX request is about to send.

```
<!DOCTYPE html>

<html>

  <head>

    <script src=

"https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">

    </script>

    <!-- Script to use ajaxSend() method -->

    <script>

      $(document).ready(function() {

        $(document).ajaxSend(function() {

          alert("AJAX request is about to send");

        });
$("button").click(function() {

          $("#paragraph").load("demo.txt");

        });

      });

    </script>

  </head>


  <body style="text-align:center;">


    <div id="div_content">


      <h1 style = "color: green;">
```

```
Sunstone Eduversity

        </h1>



        <p id = "paragraph" style= "font-size: 20px;">

         Best Job Placement Assistance

        </p>

      </div>



      <button>

        Change Content

      </button>

    </body>

  </html>
```

jQuery | ajaxSetup() Method

The ajaxSetup() method in jQuery is used to set the default values for future AJAX requests.

Syntax:

```
$.ajaxSetup( {name:value, name:value, ... } )
```

Parameters:

- type: It is used to specify the type of request.
- url: It is used to specify the URL to send the request to.
- username: It is used to specify a username to be used in an HTTP access authentication request.
- xhr: It is used for creating the XMLHttpRequest object.
- async: It's default value is true. It indicates whether the request should be handled asynchronous or not.
- beforeSend(xhr): It is a function which is to be run before the request is being sent.

jQuery | ajaxSetup() Method

- dataType: The data type expected of the server response.
- error(xhr, status, error): It is used to run if the request fails.
- global: It's default value is true. It is used to specify whether or not to trigger global AJAX event handles for the request.
- ifModified: It's default value is false. It is used to specify whether a request is only successful if the response has changed since the last request.
- jsonp: A string overriding the callback function in a jsonp request.
- jsonpCallback: It is used to specify a name for the callback function in a jsonp request.
- cache: It's default value is true. It indicates whether the browser should cache the requested pages.
- complete(xhr, status): It is a function which is to be run when the request is being finished.
- contentType: It's default value is: "application/x-www-form-urlencoded" and it is used when data send to the server.
- context: It is used to specify the "this" value for all AJAX related callback functions.

jQuery | ajaxSetup() Method

- data: It is used to specify data to be sent to the server.
- dataFilter(data, type): It is used to handle the raw response data of the XMLHttpRequest.
- password: It is used to specify a password to be used in an HTTP access authentication request.
- processData: It’s default value is true. It is used to specify whether or not data sent with the request should be transformed into a query string.
- scriptCharset: It is used to specify the charset for the request.
- success(result, status, xhr): It is to be run when the request succeeds.
- timeout: It is the local timeout for the request. It measured in terms of milliseconds.
- traditional: It is used to specify whether or not to use the traditional style of param serialization.

Example 1: This example uses ajaxSetup() method to call data from other file.

sunstone_data.txt: This text file is called within HTML file.

Welcome to Sunstone Eduversity

```
<!DOCTYPE html>

<html>

<head>

    <title>jQuery ajaxSetup() Method</title>

    <script src=

"https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">

    </script>

    <script>

        $(document).ready(function(){

            $("li:parent").css("background-color", "green");

        });

    </script>

</head>
```

```
<body style="text-align:center;">


    <h1 id="geeks1" style="color:green">GeeksForGeeks</h1>

    <h2 id="geeks2">jQuery ajaxSetup() Method</h2>

    <h3></h3>


    <button>Click</button>


    <!--Script to use ajaxSetup() method -->
```

```
<script>
      $(document).ready(function() {
        $("button").click(function() {
          $.ajaxSetup({url: "geeks1_data.txt",
                success: function(result) {
            $("h3").html(result);
          }});
          $.ajax();
        });
      });
  </script>
</body>
</html>
```

jQuery | ajaxStart() Method

The ajaxStart() method is used to specify function to be run when an AJAX request starts.

Syntax:

$(document).ajaxStart(function())

Parameter:: It takes only one parameter.

function(): It specifies the function to run when the Ajax request starts.

The demo.txt file stored on server and it will load after clicking the change content button.

The content of demo.txt are:

This is Sunstone Eduversity.

Example-1: This example changes the content of < p > element, by taking the data from server. When the request starts the page says AJAX requ<!DOCTYPE html>

```
<html>



<head>

    <script src=

"https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">

    </script>

    <script>

        $(document).ready(function() {

            $(document).ajaxStart(function() {

                alert("AJAX request started");

            });est started.
$("button").click(function() {

            $("#paragraph").load(

             "demo.txt");

         });

      });

   </script>

   <style>

      body {

         text-align: center;

      }

   </style>

</head>
```

```
<body>

  <div id="div_content">

    <h1 style="color: green;">

      Sunstone Eduversity</h1>

    <p id="paragraph"

      style="font-size: 20px;">

      Best Job Placement Assistance

    </p>

  </div>

  <button>Change Content

 </button>

</body>

</html>
```

API Integration

- API (Application Programming Interface) integration refers to the process of connecting different software applications so they can exchange data and functions. This allows organizations to automate their processes, streamline workflows, and create new business opportunities by connecting to external services and data sources.

- APIs are typically implemented as RESTful (Representational State Transfer) or SOAP (Simple Object Access Protocol) web services, which use HTTP (Hypertext Transfer Protocol) to exchange data. RESTful APIs use a standard set of HTTP methods (GET, POST, PUT, DELETE, etc.) to perform operations, while SOAP APIs use XML (Extensible Markup Language) to exchange data.

API integration can be used in a variety of ways, including:

1. Integrating internal systems: APIs can be used to integrate different systems within an organization, allowing data and functionality to be shared between systems.

1. Connecting to external services: APIs can be used to connect to external services, such as payment processors, social media platforms, or cloud services, to access additional functionality and data.

1. Building custom applications: APIs can be used to build custom applications that combine data and functionality from multiple systems, providing a more customized and integrated experience for users.

1. Automating processes: APIs can be used to automate processes, such as triggering events or data transfers, reducing the need for manual intervention and improving efficiency.

Rest API

- REST (Representational State Transfer) is a popular architectural style for building web services and APIs. REST APIs use HTTP (Hypertext Transfer Protocol) to exchange data and are designed to be simple, flexible, and scalable.

- REST APIs use a standard set of HTTP methods (GET, POST, PUT, DELETE, etc.) to perform operations and exchange data in the form of JSON (JavaScript Object Notation) or XML (Extensible Markup Language) payloads. REST APIs are typically stateless, meaning that each request contains all the information necessary to complete the request, and do not maintain a client-server session.

The main benefits of REST APIs include:

- Simplicity: REST APIs are simple to understand and use, making it easy to integrate with other systems.
- Flexibility: REST APIs can be used to perform a wide range of operations, making them suitable for a variety of use cases.
- Scalability: REST APIs are designed to be scalable, allowing them to handle large amounts of data and traffic.
- Interoperability: REST APIs use standard HTTP methods and data formats, making them interoperable with a wide range of systems and technologies.

SOAP API

- SOAP (Simple Object Access Protocol) is a protocol for exchanging structured information in the implementation of web services. SOAP APIs use XML (Extensible Markup Language) to exchange data and are designed to be extensible, flexible, and secure.

- SOAP APIs define a standard set of rules for encoding and transmitting data, including a specific structure for the XML messages that are exchanged between systems. This structure includes a header that contains information about the message and a body that contains the actual data.

- SOAP APIs also support the use of security features, such as encryption and authentication, making them suitable for use in secure environments. Additionally, SOAP APIs support the use of attachments, allowing binary data to be included in the messages.

The main benefits of SOAP APIs include:

1. Extensibility: SOAP APIs are designed to be extensible, allowing new features and functionality to be added over time.

1. Flexibility: SOAP APIs can be used to perform a wide range of operations, making them suitable for a variety of use cases.

1. Security: SOAP APIs support the use of security features, making them suitable for use in secure environments.

1. Interoperability: SOAP APIs use standard XML formats and data structures, making them interoperable with a wide range of systems and technologies.

Step by step implementation to fetch data from an api in react.

Step 1: Create React Project

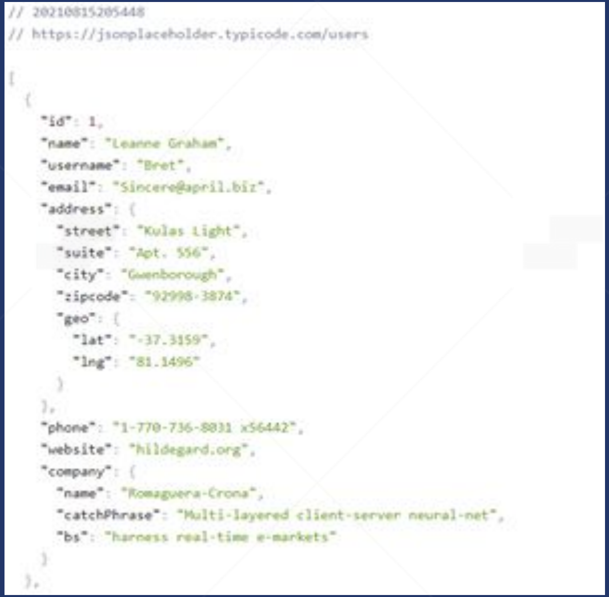
```
npm create-react-app MY-APP
```

Step 2: Change your directory and enter your main folder charting as

```
cd MY-APP
```

Step 3: API endpoint

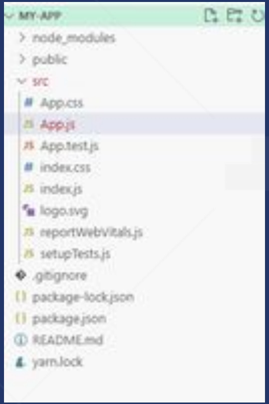
https://jsonplaceholder.typicode.com/users



```
// 20210815205448
// https://jsonplaceholder.typicode.com/users

[
  {
    "id": 1,
    "name": "Leanne Graham",
    "username": "Bret",
    "email": "Sincere@april.biz",
    "address": {
      "street": "Kulas Light",
      "suite": "Apt. 556",
      "city": "Gwenborough",
      "zipcode": "92998-3874",
      "geo": {
        "lat": "-37.3159",
        "lng": "81.1496"
      }
    },
    "phone": "1-770-736-8031 x56442",
    "website": "hildegard.org",
    "company": {
      "name": "Romaguera-Crona",
      "catchPhrase": "Multi-layered client-server neural-net",
      "bs": "harness real-time e-markets"
    }
  },
```

Step 4: Write code in App.js to fetch data from API and we are using fetch function.

Project Structure: It will look the following.



```
MY-APP
> node_modules
> public
v src
   App.css
   App.js
   App.test.js
   index.css
   index.js
   logo.svg
   reportWebVitals.js
   setupTests.js
 .gitignore
 package-lock.json
 package.json
 README.md
 yarn.lock
```

Example:

```
import React from "react";

import './App.css';

class App extends React.Component {



   // Constructor

   constructor(props) {



   }
```

```
super(props);


      this.state = {

         items: [],

         DataisLoaded: false

      };
```

```
// ComponentDidMount is used to
    // execute the code
    componentDidMount() {
        fetch(
"https://jsonplaceholder.typicode.com/users")
.then((res) => res.json())

        .then((json) => {

            this.setState({

                items: json,

                DataisLoaded: true

            });

        })

  }
```

```
render() {

        const { DataisLoaded, items } = this.state;

        if (!DataisLoaded) return <div>

            <h1> Pleses wait some time.... </h1> </div> ;


        return (

        <div className = "App">

            <h1> Fetch data from an api in react </h1>  {

                items.map((item) => (

                <ol key = { item.id } >
User_Name: { item.username },
                Full_Name: { item.name },
                User_Email: { item.email }
                </ol>
            ))
        }
    </div>
  );
}
}

export default App;
```

Write code in App.css for styling the app.js file.

```
.App {
   text-align: center;
   color: Green;
}
```

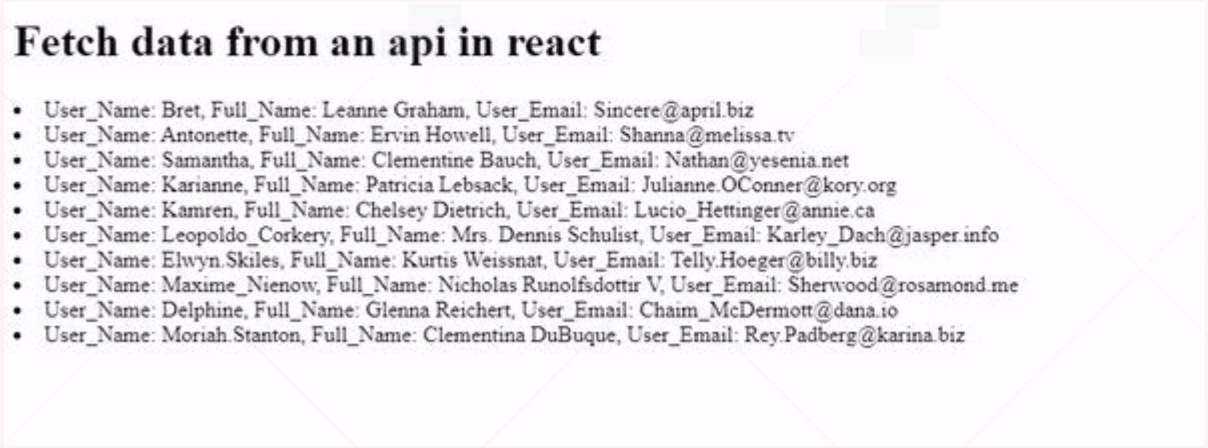
```
.App-header {
    background-color: #282c34;
    min-height: 100vh;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    font-size: calc(10px + 2vmin);
    color: white;
```

```
.App-link {
  color: #61dafb;
}


@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}
```

Step to run the application: Open the terminal and type the following command.

- npm start

Output: Open the browser and our project is shown in the URL http://localhost:3000/



```
Fetch data from an api in react

• User_Name: Bret, Full_Name: Leanne Graham, User_Email: Sincere@april.biz
• User_Name: Antonette, Full_Name: Ervin Howell, User_Email: Shanna@melissa.tv
• User_Name: Samantha, Full_Name: Clementine Bauch, User_Email: Nathan@yesenia.net
• User_Name: Karianne, Full_Name: Patricia Lebsack, User_Email: Julianne.OConner@kory.org
• User_Name: Kamren, Full_Name: Chelsey Dietrich, User_Email: Lucio_Hettinger@annie.ca
• User_Name: Leopoldo_Corkery, Full_Name: Mrs. Dennis Schulist, User_Email: Karley_Dach@jasper.info
• User_Name: Elwyn.Skiles, Full_Name: Kurtis Weissnat, User_Email: Telly.Hoeger@billy.biz
• User_Name: Maxime_Nienow, Full_Name: Nicholas Runolfsdottir V, User_Email: Sherwood@rosamond.me
• User_Name: Delphine, Full_Name: Glenna Reichert, User_Email: Chaim_McDermott@dana.io
• User_Name: Moriah.Stanton, Full_Name: Clementina DuBuque, User_Email: Rey.Padberg@karina.biz
```

SUNSTONE

Thank you!