A
Summer Internship Report
On
"Software Development"

(IT446 Summer Internship-II)

Prepared by

AYUSH SHAH (19IT127)

Under the guidance of

Prof. Priteshkumar Prajapati

Submitted to

Charotar University of Science and Technology

for Partial Fulfillment of the Requirements for the

Degree of Bachelor of Technology

In Information Technology

(6th Semester IT446 Summer Internship-II)

Submitted at

Accredited with Grade A by NAAC
Accredited with Grade A by KCG

SMT. KUNDANBEN DINSHA PATEL DEPARTMENT OF INFORMATION TECHNOLOGY

(NBA Accredited)

Chandubhai S. Patel Institute of Technology (CSPIT)

Faculty of Technology & Engineering (FTE), CHARUSAT

At: Changa, Dist: Anand, Pin: 388421.

July, 2022

SUMMER INTERNSHIP CERTIFICATE

This is to certify that Ayush Shah, B.Tech (I.T.) student of Chandubhai S. Patel Institute of Technology, CHARUSAT, Changa is pursuing summer internship as a Software Developer Intern, under the mentorship of Mr. Amin Raiyan and Mr. Ronak Thacker at smartSense Consulting Solutions Pvt. Ltd.

Respectfully,

Adarsh Dave

Manager - Humar Resource

smartSense Consulting Solutions Pvt. Ltd.

CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CERTIFICATE

This is to certify that the report entitled "**Software Development**" is a bonafied work carried out by **Mr. Ayush Shah (19IT127)** under the guidance and supervision of **Prof. Priteshkumar Prajapati , Mr. Amin Raiyani and Ronak Thacker** for the subject **IT446 Summer Internship-II (IT446)** of 6th Semester of Bachelor of Technology in **Information Technology** at Faculty of Technology & Engineering – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidates himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation, and language for being referred to the examiner.

Under supervision of,

Mr. Ronak Thacker
Team lead
smartSense Solutions

Mr Amin Raiyani
Frontend Team lead
FullStack Development
smartSense Solutions

Dr. Parth Shah
Head & Professor
Smt. Kundanben Dinsha Patel
Department of Information Technology

Prof. Priteshkumar Prajapati
Assistant Professor
Smt. Kundanben Dinsha Patel
Department of Information Technology

At: Changa, Ta. Petlad, Dist. Anand, PIN: 388 421. Gujarat

ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many hands. We would like to extend our sincere thanks to all of them.

We are highly indebted to Dr. Parth Shah, Prof Nirav Bhatt and Prof. Pritesh Prajapati for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We are grateful to our external guide Mr. Ronak Thacker , Mr Adarsh Dave, Mr Amin Raiyani and Mr Anand Tanna in smartSense Solutions for giving us the support and encouragement that was necessary for the completion of this project.

We would like to express our gratitude to H.O.D. Dr. Parth Shah, and we are also grateful to all our faculty members of Chandubhai S. Patel Institute of Technology for their kind cooperation and encouragement which help us in completion of this project and preparing the report.

Last but not the least, we would also like to thank our colleagues, who have co – operated during the preparation of our report and without them this project would not have been possible. Their ideas helped us a lot to improve our project report.

“We may not achieve everything we dream, but we cannot achieve anything unless we dream of it.” –

Ayush Shah (19IT127)

ABSTRACT

In essence, HTML is used for creating the primary content of a webpage, giving it structure. You start by writing words, then apply tags or elements to these words. The web browser then reads this and can then understand the heading of a page, any paragraphs, and where the page starts and finishes, thus filling your web page with content.

HTML is supported by every single browser and is established on pretty much every webpage in existence. You don't need any licenses, you don't need to pay for it, and it can be pretty easy to learn and code. If we can compare a webpage to the human body, then HTML is the bones of the body.

If HTML is the bones of the body, then CSS is the skin that covers it. It's used for background color, styling, layout, borders, shadowing – all the essential design bits and bobs that make a webpage look slick and smart. CSS enables you to distinguish between presentation and content by modifying the design and display of HTML elements.

Presentation and ease of use are a couple of the main things that CSS has bought to web design by translating the way content looks on a webpage and what else goes on it to complement that content. While frequently used in conjunction with HTML, it is actually independent of it, and can be used with any XML-based markup language.

The purpose of this internship is to know how websites and whole web is working and which architectures are used to make a websites.So,this is the main purpose of internship as a Software Developer Intern.

Table of Contents

List of Figures

Company Details

smartSense Consulting Solutions Pvt Ltd (smartSense solutions) is no common corporate. It stands unique in every aspect of working environment. We are a team/organization consisting of members who are passionate and pursuant of excellence in what we do. We are eminent team that signs their work with excellence.

We work based on milestones and always reach them before the deadline does. We promote overt discussions with our dealers, clients, employees and all other business relationships.

We seek, create and thrive at opportunities to develop what people may have never thought of before and will not think of else after using it. We do not code to provide service, we code for “joie de vivre”.

Our aim is to remain one of the most innovative organizations around the world that provides concrete solutions to the clients in various domains around the globe. And we believe that it always makes smart sense if the customer wins. That’s why our product solutions, ranging from Education domain, well-developed organisations, to smartHome, have been customized to extend the benefits of our customers.

Our mission is to continue implementing smart development Solutions for the customers and providing them outstanding IT services. We always keep the business perspective along with the technical knowledge in order to help our clients develop robust platform technically as well as practically. All in all “Customer’s Win is our Mission”.

1. HTML

1.1 Introduction

Fig 1.1 Logo

The web is constantly evolving. New and innovative websites are being created every day, pushing the boundaries of HTML in every direction. HTML 4 has been around for nearly a decade now, and publishers seeking new techniques to provide enhanced functionality are being held back by the constraints of the language and browsers. To give authors more flexibility and interoperability, and enable more interactive and exciting websites and applications, HTML 5 introduces and enhances a wide range of features including form controls, APIs, multimedia, structure, and semantics.
Work on HTML 5, which commenced in 2004, is currently being carried out in a joint effort between the W3C HTML WG and the WHATWG. Many key players are participating in the W3C effort including representatives from the four major browser vendors: Apple, Mozilla, Opera, and Microsoft; and a range of other organisations and individuals with many diverse interests and expertise.

Work on HTML 5, which commenced in 2004, is currently being carried out in a joint effort between the W3C HTML WG and the WHATWG. Many key players are participating in the W3C effort including representatives from the four major browser vendors: Apple, Mozilla, Opera, and Microsoft; and a range of other organisations and individuals with many diverse interests and expertise.

HTML5 is a specification for how the web's core language, HTML, should be formatted and utilized to deliver text, images, multimedia, web apps, search forms, and anything else you see in your browser. In some ways, it's mostly a core set of standards that only web developers really need to know. In other ways, it's a major revision to how the web is

put together. Not every web site will use it, but those that do will have better support across modern desktop and mobile browsers (that is, everything except Internet Explorer).

1.2 List & Tables

HTML lists allow web developers to group a set of related items in lists.

1.2.1 Unordered HTML List

An unordered list starts with the <ul> tag. Each list item starts with the <li> tag.

The list items will be marked with bullets (small black circles) by default.

1.2.2 Ordered HTML List

An ordered list starts with the <ol> tag. Each list item starts with the <li> tag.

The list items will be marked with numbers by default:

Tag	Description
	Defines an unordered list
	Defines an ordered list
	Defines a list item
<dl>	Defines a description list
<dt>	Defines a term in a description list
<dd>	Describes the term in a description list

Fig 1.2 List Tags

Tables:

A table in HTML consists of table cells inside rows and columns

Table Cell: Each table cell is defined by a <td> and a </td> tag.

Table Row: Each table row starts with a <tr> and end with a </tr> tag.

Table Header: Sometimes you want your cells to be headers, in those cases use the <th> tag instead of the <td> tag.

1.3 Divs & Spans, Classes & Ids

The tag is a block level element.	The tag is an inline element.
It is best to attach it to a section of a web page.	It is best to attach a CSS to a small section of a line in a web page.
It accepts align attribute.	It does not accept align attribute.
This tag should be used to wrap a section, for highlighting that section.	This tag should be used to wrap any specific word that you want to highlight in your webpage.

1.4 HTML Semantic Tags

1 A semantic element clearly describes its meaning to both the browser and the developer.

2 Examples of **non-semantic** elements: and - Tells nothing about its content.

3 Examples of **semantic** elements: <form>, <table>, and <article> - Clearly defines its content.

In HTML there are some semantic elements that can be used to define different parts of a web page:

Fig 1.3 HTML Structure

- <article>
- <aside>
- <details>
- <figcaption>
- <figure>
- <footer>
- <header>
- <main>
- <mark>
- <nav>
- <section>
- <summary>
- <time>

Tag	Description
<article>	Defines independent, self-contained content
<aside>	Defines content aside from the page content
<details>	Defines additional details that the user can view or hide
<figcaption>	Defines a caption for a <figure> element
<figure>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
<footer>	Defines a footer for a document or section
<header>	Specifies a header for a document or section
<main>	Specifies the main content of a document
<mark>	Defines marked/highlighted text
<nav>	Defines navigation links
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time

Fig 1.4 Semantic Tags

2. Basic CSS

2.1 Intoduction

- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once
- External stylesheets are stored in CSS files

2.2 Box Model, Margin & Padding

In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:

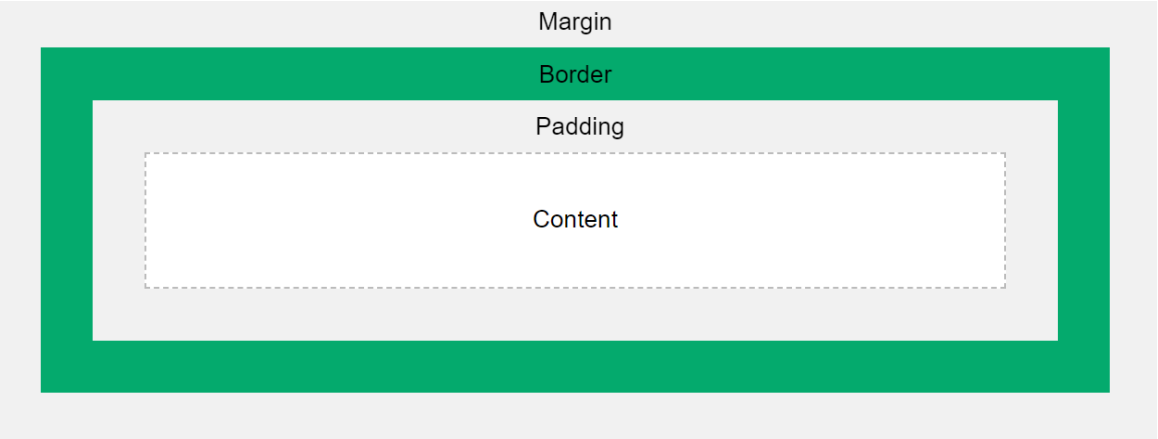


Fig 2.1 Box Model

Explanation of the different parts:

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent

- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

2.3 Inline, Block & Inline-Block Display

Inline: An inline element does not start on a new line.

An inline element only takes up as much width as necessary.

Here are the inline-level elements in HTML:

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>		<dl>
<dt>	<fieldset>	<figcaption>	<figure>	<footer>	<form>	<h1>-<h6>	<header>
<hr>		<main>	<nav>	<noscript>		<p>	<pre>
<section>	<table>	<tfoot>		<video>			

Fig 2.2 Inline Tags

Block: A block-level element always starts on a new line, and the browsers automatically add some space (a margin) before and after the element.

A block-level element always takes up the full width available (stretches out to the left and right as far as it can).

Here are the block-level elements in HTML:

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>		<dl>
<dt>	<fieldset>	<figcaption>	<figure>	<footer>	<form>	<h1>-<h6>	<header>
<hr>		<main>	<nav>	<noscript>		<p>	<pre>
<section>	<table>	<tfoot>		<video>			

Fig 2.3 Block Tags

Inline-Block: Compared to display: inline, the major difference is that display: inline-block allows to set a width and height on the element.

Also, with display: inline-block, the top and bottom margins/paddings are respected, but with display: inline they are not.

3. Advance CSS

3.1 Intro To Flexbox

Flexbox is tool to build layouts. It provides a cross browser-compatible method to implement layouts and makes it easier to add responsive design to your application, without you having to explicitly deal with positioning the components of your page. The only prerequisite to learn flexbox is CSS selectors.

Property	Description
align-content	Modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines
align-items	Vertically aligns the flex items when the items do not use all available space on the cross-axis
display	Specifies the type of box used for an HTML element
flex-direction	Specifies the direction of the flexible items inside a flex container
flex-flow	A shorthand property for flex-direction and flex-wrap
flex-wrap	Specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line
justify-content	Horizontally aligns the flex items when the items do not use all available space on the main-axis

Fig 3.1 Flex Properties

The flex container becomes flexible by setting the display property to flex

3.2 Intro To Grid

CSS Grid operates within a grid layout: a set of rows and columns (not unlike a table). To create a grid with CSS Grid, you specify how many columns wide it should be, and how many of those columns any particular piece of content should occupy.

Property	Description
column-gap	Specifies the gap between the columns
gap	A shorthand property for the *row-gap* and the *column-gap* properties
grid	A shorthand property for the *grid-template-rows*, *grid-template-columns*, *grid-template-areas*, *grid-auto-rows*, *grid-auto-columns*, and the *grid-auto-flow* properties
grid-area	Either specifies a name for the grid item, or this property is a shorthand property for the *grid-row-start*, *grid-column-start*, *grid-row-end*, and *grid-column-end* properties
grid-auto-columns	Specifies a default column size
grid-auto-flow	Specifies how auto-placed items are inserted in the grid
grid-auto-rows	Specifies a default row size
grid-column	A shorthand property for the *grid-column-start* and the *grid-column-end* properties
grid-column-end	Specifies where to end the grid item
grid-column-gap	Specifies the size of the gap between columns
grid-column-start	Specifies where to start the grid item
grid-gap	A shorthand property for the *grid-row-gap* and *grid-column-gap* properties
grid-row	A shorthand property for the *grid-row-start* and the *grid-row-end* properties
grid-row-end	Specifies where to end the grid item

Fig 3.2 Grid Properties

3.3 Intro To Responsive Layout

Responsive web design makes your web page look good on all devices.Responsive web design uses only HTML and CSS.Responsive web design is not a program or a JavaScript.It uses the @media rule to include a block of CSS properties only if a certain condition is true.

3.4 Animation, Transition & Transform Property

CSS transitions allows you to change property values smoothly, over a given duration.

Property	Description
transition	A shorthand property for setting the four transition properties into a single property
transition-delay	Specifies a delay (in seconds) for the transition effect
transition-duration	Specifies how many seconds or milliseconds a transition effect takes to complete
transition-property	Specifies the name of the CSS property the transition effect is for
transition-timing-function	Specifies the speed curve of the transition effect

Fig 3.3 Transition Properties

Animation:

An animation lets an element gradually change from one style to another.

You can change as many CSS properties you want, as many times as you want.

Keyframes hold what styles the element will have at certain times.

Property	Description
@keyframes	Specifies the animation code
animation	A shorthand property for setting all the animation properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards, backwards or in alternate cycles
animation-duration	Specifies how long time an animation should take to complete one cycle
animation-fill-mode	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies the name of the @keyframes animation
animation-play-state	Specifies whether the animation is running or paused
animation-timing-function	Specifies the speed curve of the animation

Fig 3.4 Animation Properties

Transforms: CSS transforms allow you to move, rotate, scale, and skew elements.

Function	Description
matrix(*n,n,n,n,n,n*)	Defines a 2D transformation, using a matrix of six values
translate(*x,y*)	Defines a 2D translation, moving the element along the X- and the Y-axis
translateX(*n*)	Defines a 2D translation, moving the element along the X-axis
translateY(*n*)	Defines a 2D translation, moving the element along the Y-axis
scale(*x,y*)	Defines a 2D scale transformation, changing the elements width and height
scaleX(*n*)	Defines a 2D scale transformation, changing the element's width
scaleY(*n*)	Defines a 2D scale transformation, changing the element's height
rotate(*angle*)	Defines a 2D rotation, the angle is specified in the parameter
skew(*x-angle,y-angle*)	Defines a 2D skew transformation along the X- and the Y-axis
skewX(*angle*)	Defines a 2D skew transformation along the X-axis
skewY(*angle*)	Defines a 2D skew transformation along the Y-axis

Fig 3.5 Tranform Properties

3. SASS

3.1 Introduction To SASS

Sass (Syntactically awesome style sheets) is a style sheet language initially designed by Hampton Catlin and developed by Natalie Weizenbaum. Sass is a preprocessor scripting language that is interpreted or compiled into Cascading Style Sheets (CSS). SassScript is the scripting language itself. Embrace Sass once, and you may never want to go back to vanilla CSS again. I didn't realize how much I was enjoying working with Sass until recently when I had to switch back to vanilla CSS in a project.

3.2 Variables

A variable declaration looks a lot like a property declaration: it's written <variable>: <expression>. Unlike a property, which can only be declared in a style rule or at-rule, variables can be declared anywhere you want. To use a variable, just include it in a value.

Variables declared at the top level of a stylesheet are *global*. This means that they can be accessed anywhere in their module after they've been declared. But that's not true for all variables. Those declared in blocks (curly braces in SCSS or indented code in Sass) are usually *local*, and can only be accessed within the block they were declared.

Local variables can even be declared with the same name as a global variable. If this happens, there are actually two different variables with the same name: one local and one global. This helps ensure that an author writing a local variable doesn't accidentally change the value of a global variable they aren't even aware of.

SCSS Sass

```scss
$base-color: #c6538c;
$border-dark: rgba($base-color, 0.88);

.alert {
  border: 1px solid $border-dark;
}
```

CSS

```css
.alert {
  border: 1px solid rgba(198, 83, 140, 0.88);
}
```

Fig 4.1 Variables

3.3 Nesting & Structuring

Nesting is one most popular features of SCSS. With nesting, you can add classes between the braces of a declaration. SCSS will compile and handle the selectors quite intuitively. You can even use the "&" character to get a reference to the parent selector.

Base: contained within this file are all your resets, variables, mixins, and any utility classes. Layout: contains all the CSS that handles the layout, such as the container and any grid systems. Components: anything reusable such as buttons, navbars, cards etc.

3.4 Functions, Mixins & More

Mixins and extends are powerful features that help to avoid a lot of repetition. With mixins, you can make parameterized CSS declarations and reuse them throughout your stylesheets.

Import:

@import will be handled by Sass and all our CSS and SCSS files will be compiled to a single file that will end up on our live site. You can create partial Sass files that contain little snippets of CSS that you can include in other Sass files, i.e. variable.scss, fonts.scss, buttons.scss, etc., and then we can include all SCSS files in the main/style.scss folder. If you don't import the partial files, then reusable components like mixin and variable will not work.

Functions are defined using the @function at-rule, which is written @function <name>(<arguments...>) { ... }. A function's name can be any Sass identifier. It can only contain universal statements, as well as the @return at-rule which indicates the value to use as the result of the function call. Functions are called using the normal CSS function syntax.

SCSS Sass

```scss
@function pow($base, $exponent) {
  $result: 1;
  @for $_ from 1 through $exponent {
    $result: $result * $base;
  }
  @return $result;
}

.sidebar {
  float: left;
  margin-left: pow(4, 3) * 1px;
}
```

CSS

```css
.sidebar {
  float: left;
  margin-left: 64px;
}
```

Fig 4.2 Functions

4. Projects

5.1 Hotel Website

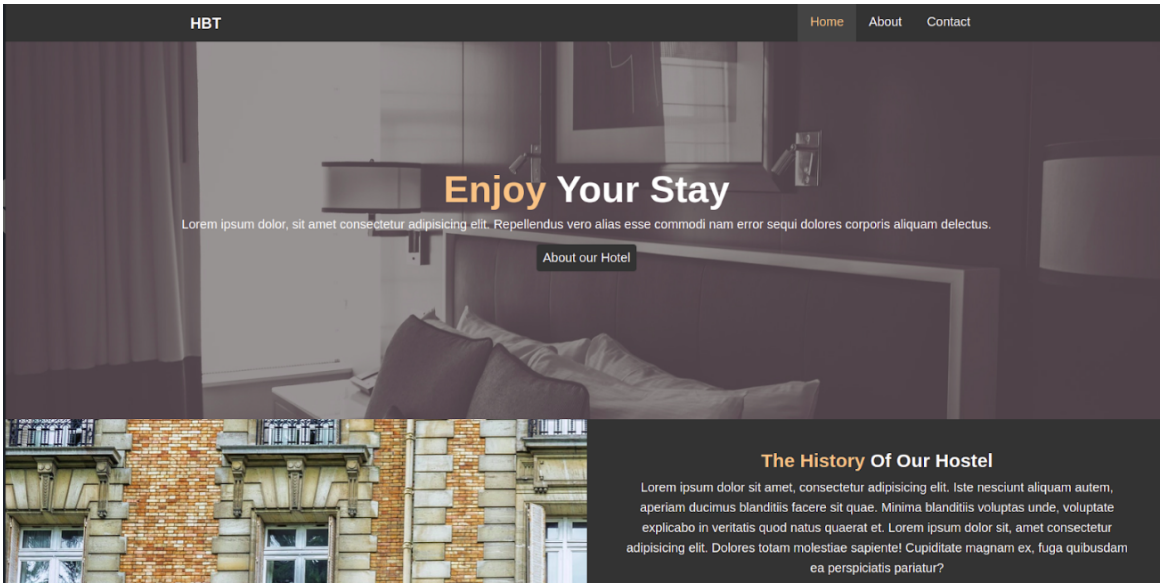


Fig 5.1 Hotel Home

This website is made with the use HTML & CSS.There are some Advance CSS concepts are used in this project like flexbox, grid etc

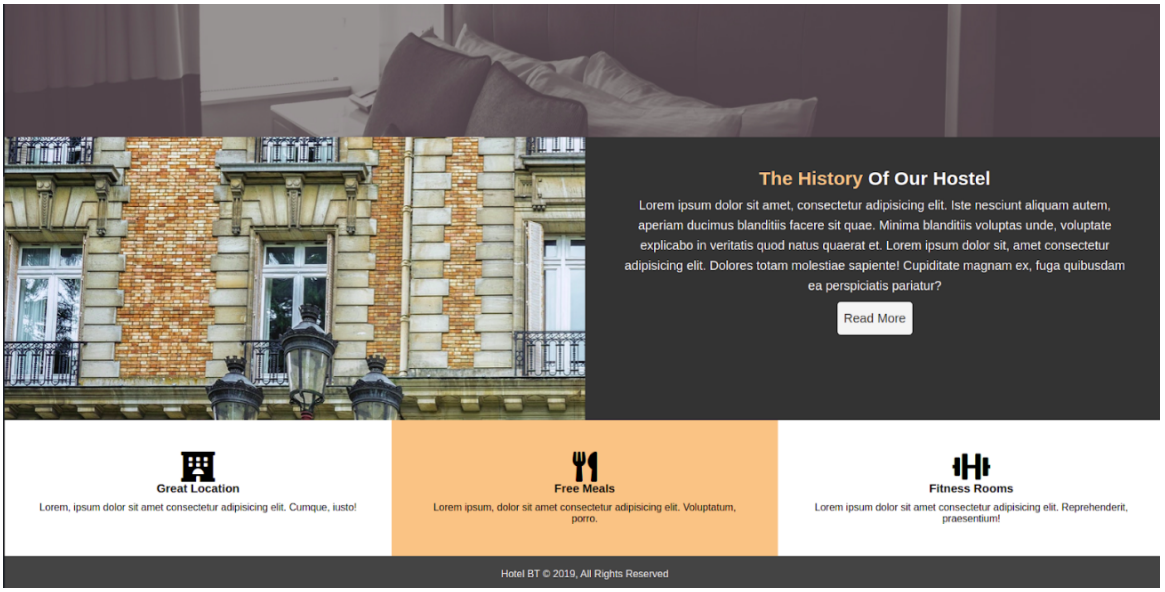


Fig 5.2 Hotel Flex

In fig 5.2 I had the concept of flexbox in thos there boxes which sre in row.

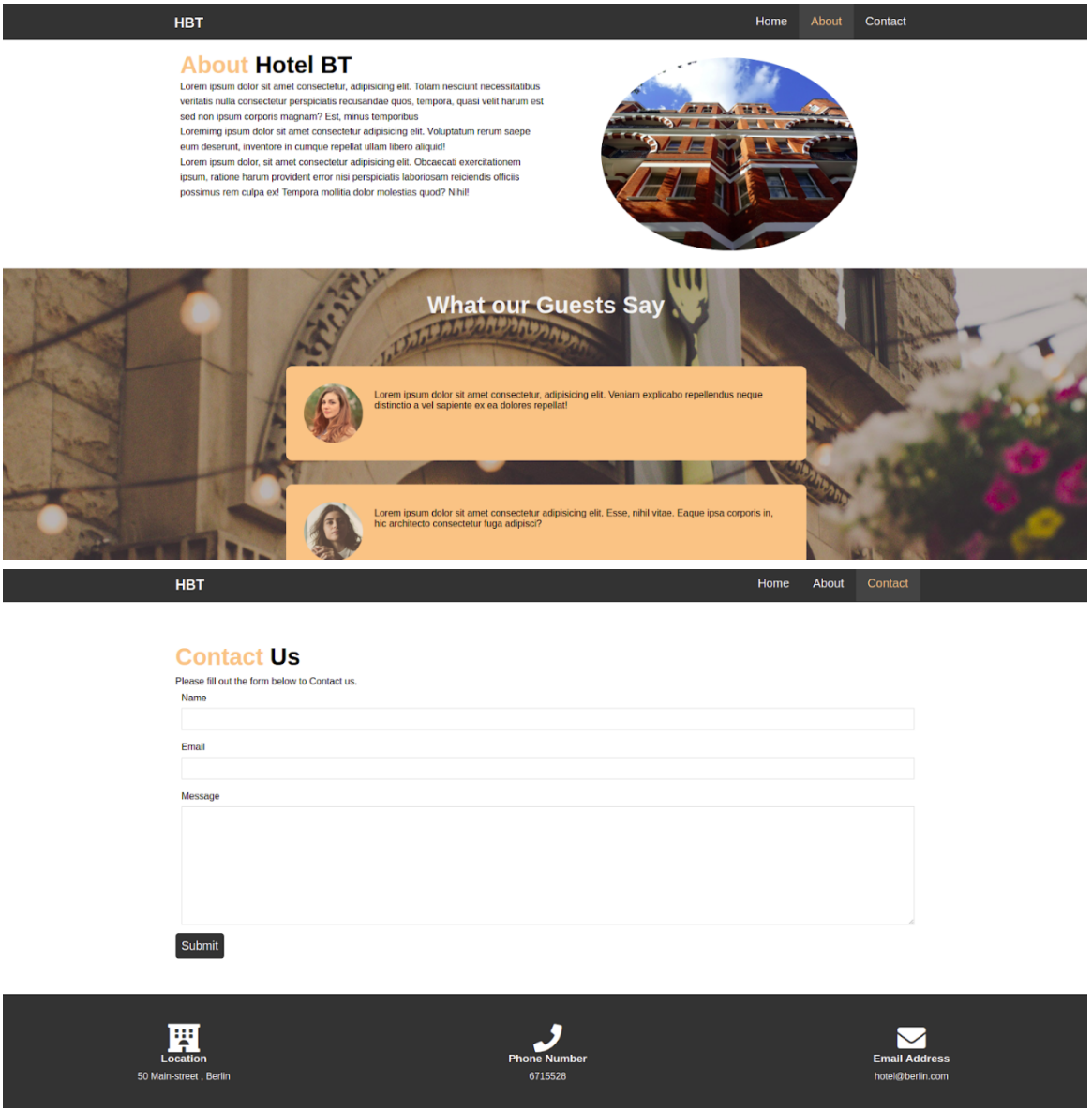


Fig 5.3 Hotel Contact

In fig 5.3 contact page I had used flexbox in 3 boxes of footer

5.2 Edgeledger

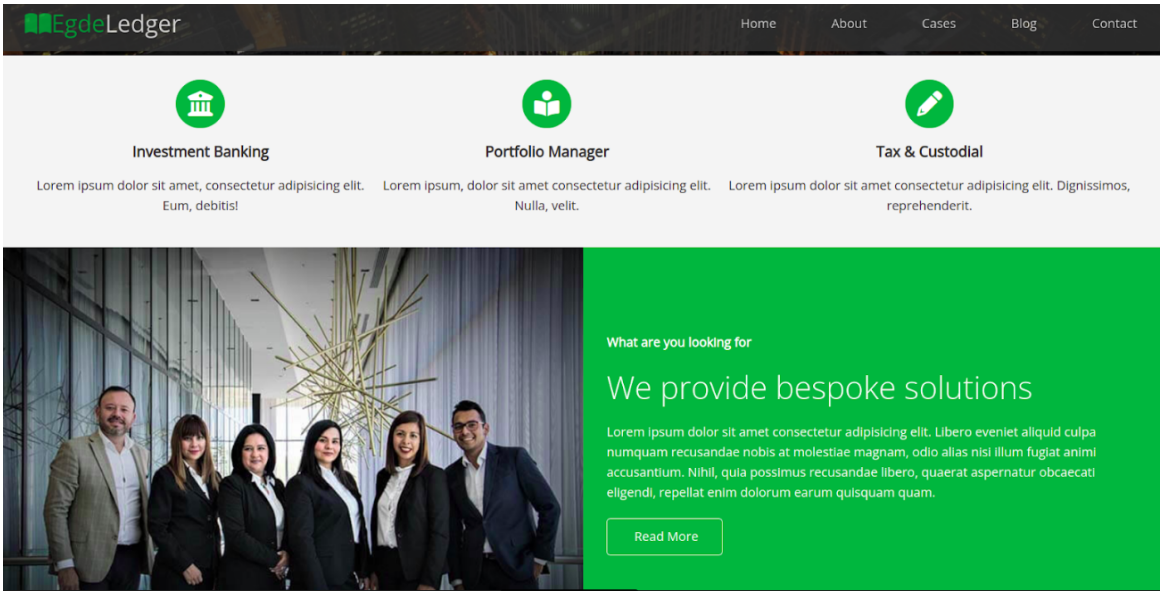


Fig 5.4 Edgeleger Website

In this page I had used flex in upper three boxes

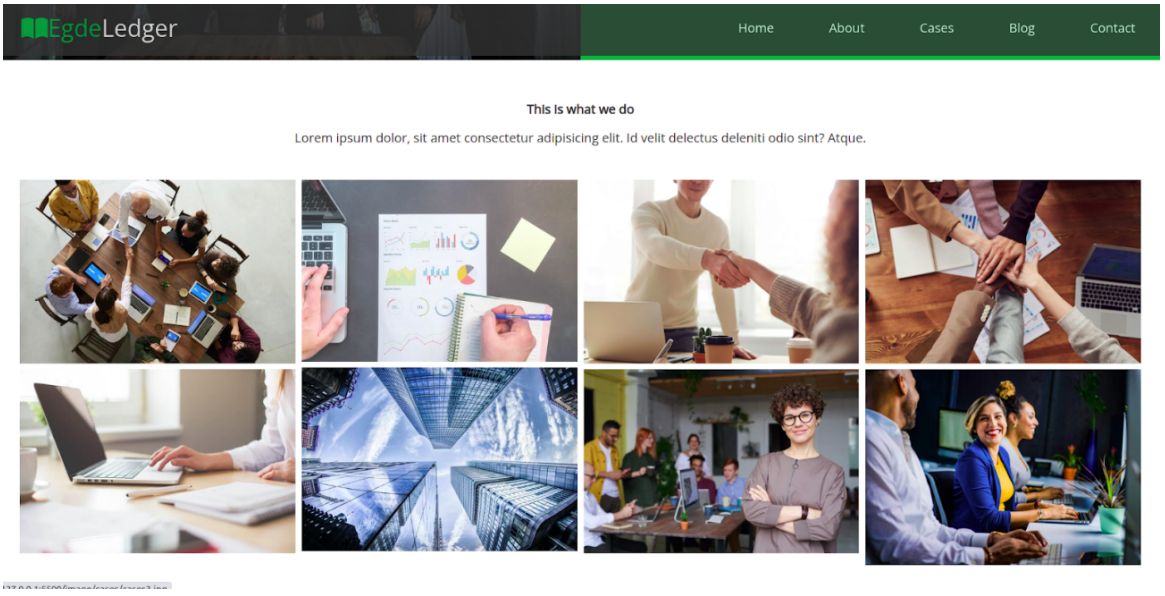


Fig 5.5 Edgeledger Grid

I had used grid in this 8 images because it is easy to select grid for 2 dimensions.

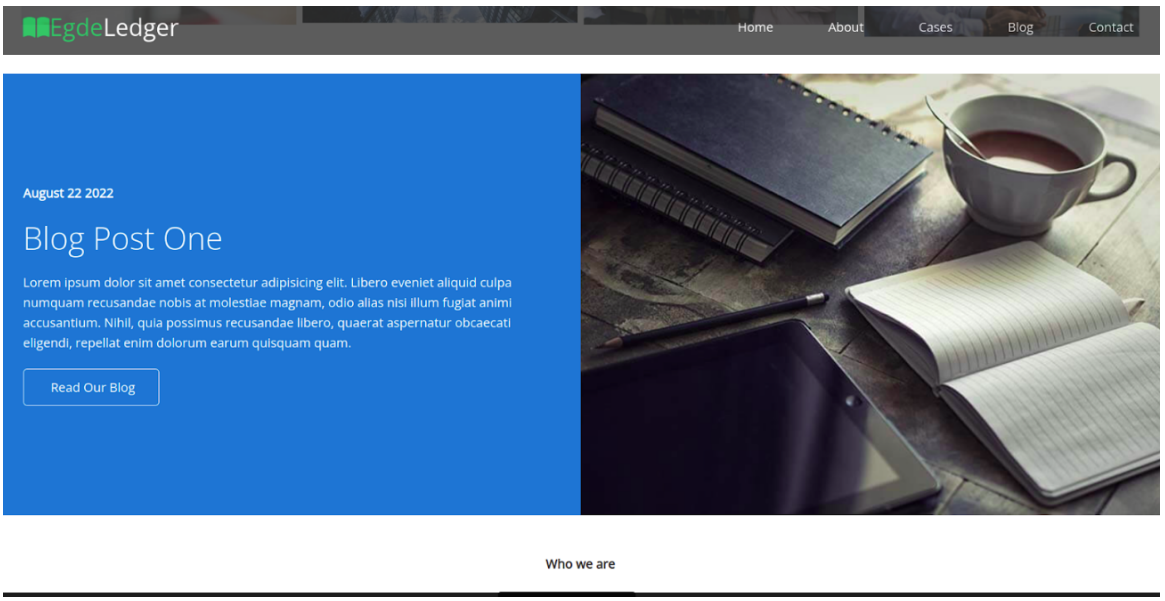


Fig 5.6 Edgeledger Flex

In fig 5.5 I had used flexbox.

Fig 5.7 Edgeledger Blog

In fig 5.6 I had used flexbox.

5.3 NewsGrid Website

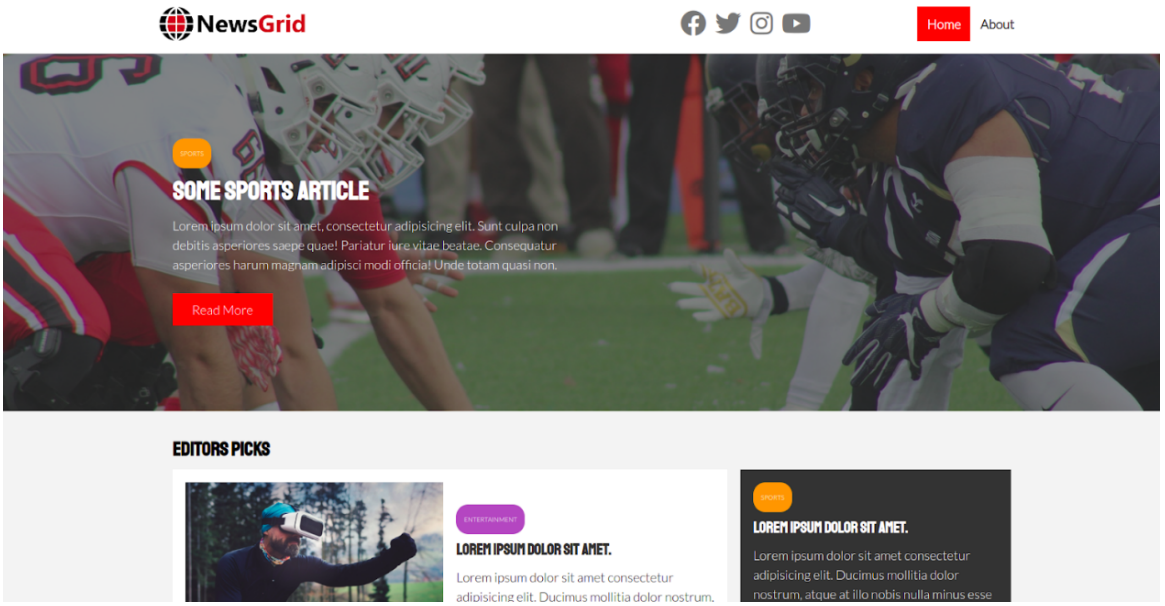


Fig 5.8 NewsGrid

In this website many advance properties of css like before,after flex, grid etc properties are used.

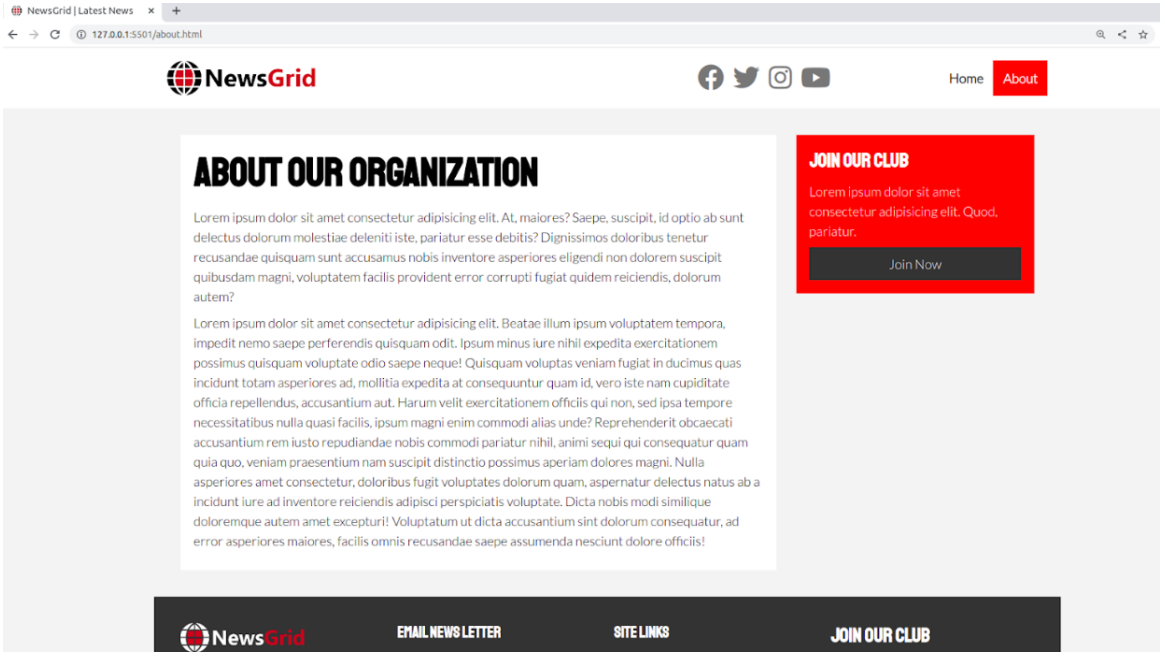


Fig 5.9 Newsgrid Details

This pages shows the organization details.I used float property of css here.

5.4 Portfolio Website

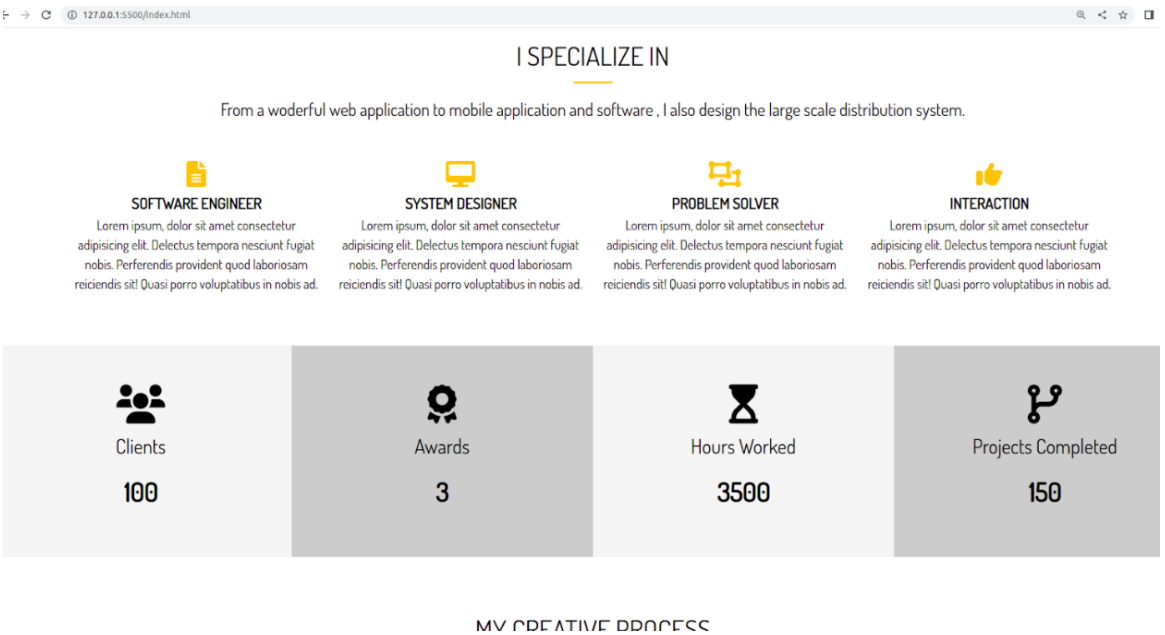


Fig 5.10 Porfolio Website

In this I had used flexbox property to keep boxes in one row.

5.10 Emexso Website Landing Page

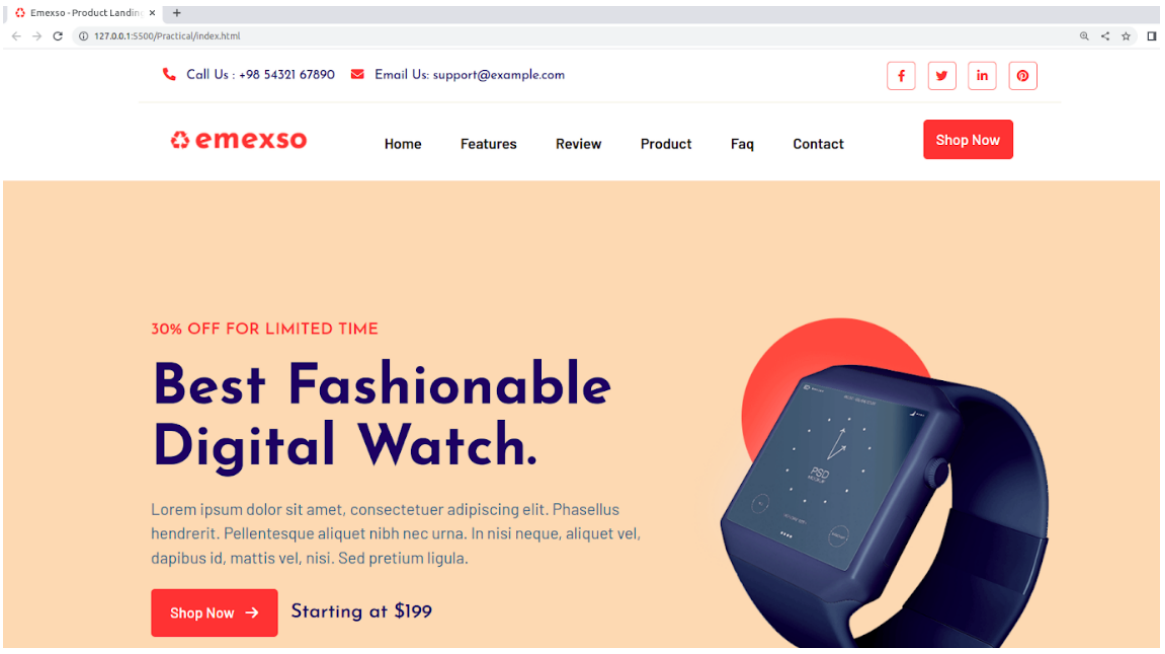


Fig 5.11 Emexso Website

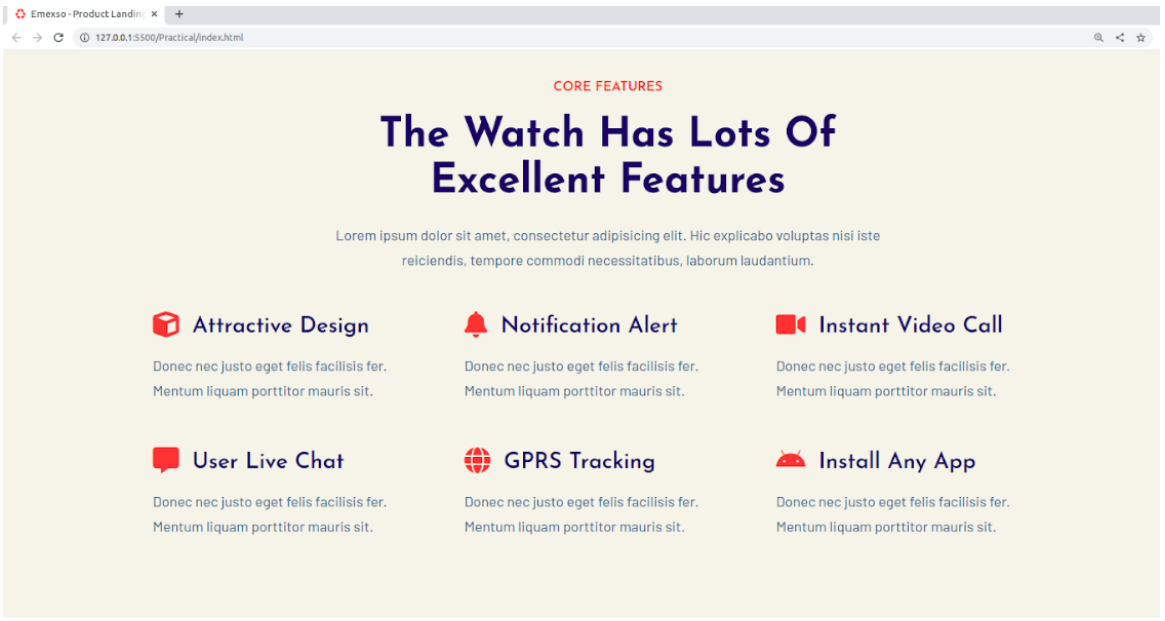


Fig 5.12 Emexso Grid

Here, in fig 5.12 I had used grid property because it is in 2D

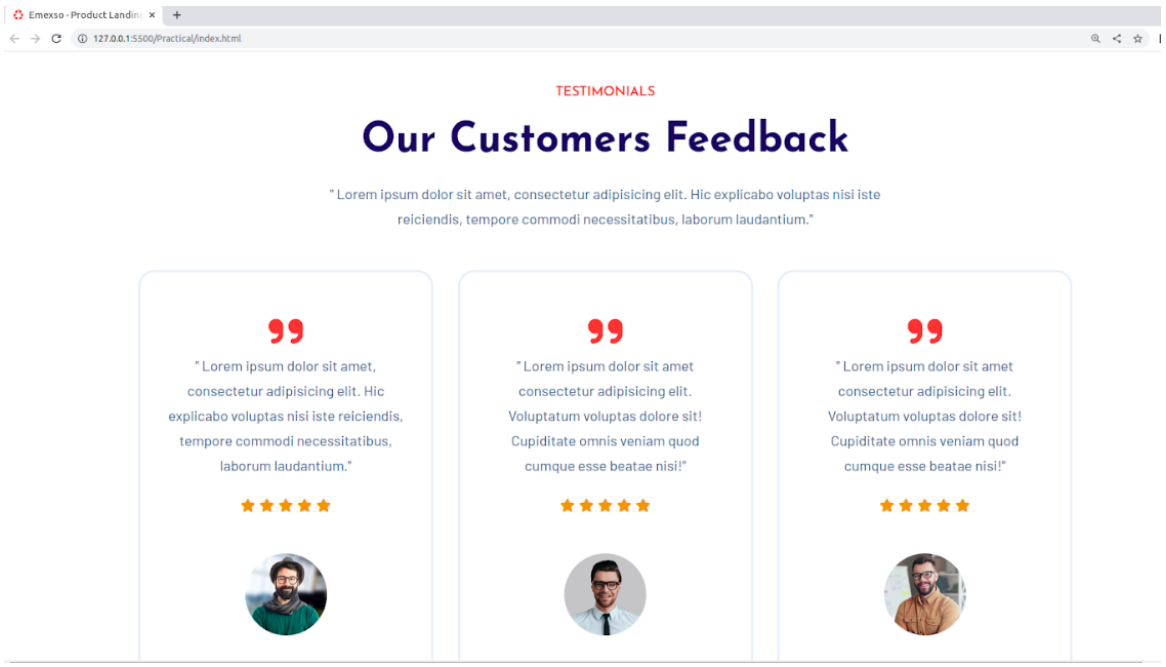


Fig 5.13 Emexso Flex

Here,in fig 5.13 I had used flex property

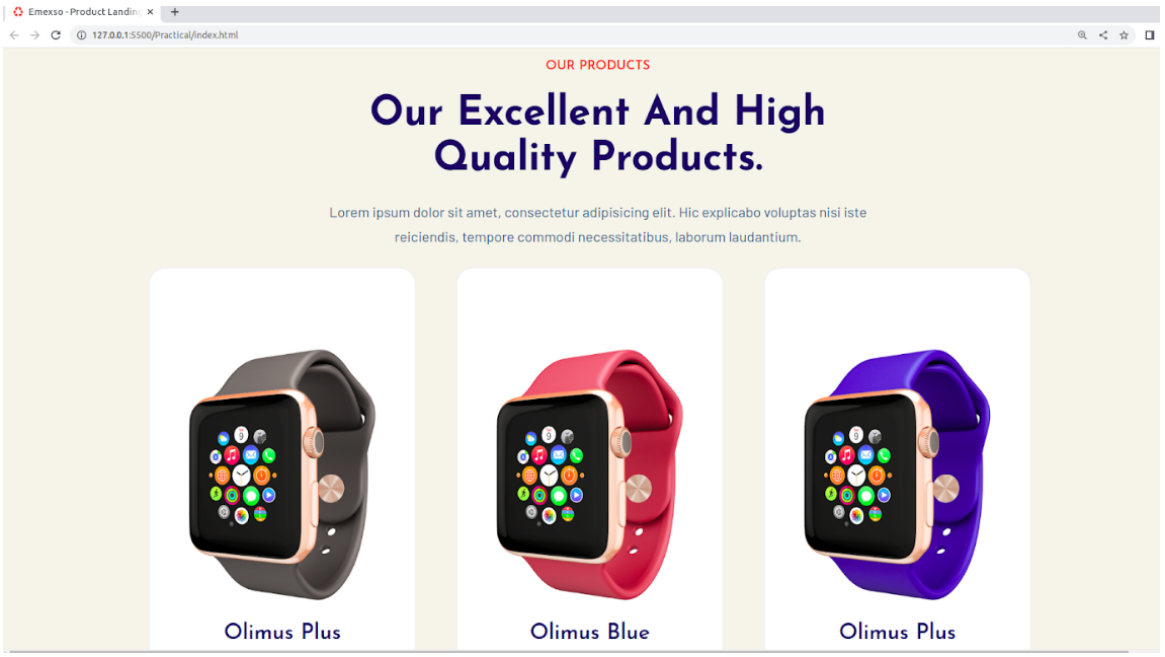


Fig 5.14 Emexso Flex2

Here,in fig 5.14 I had used flex property

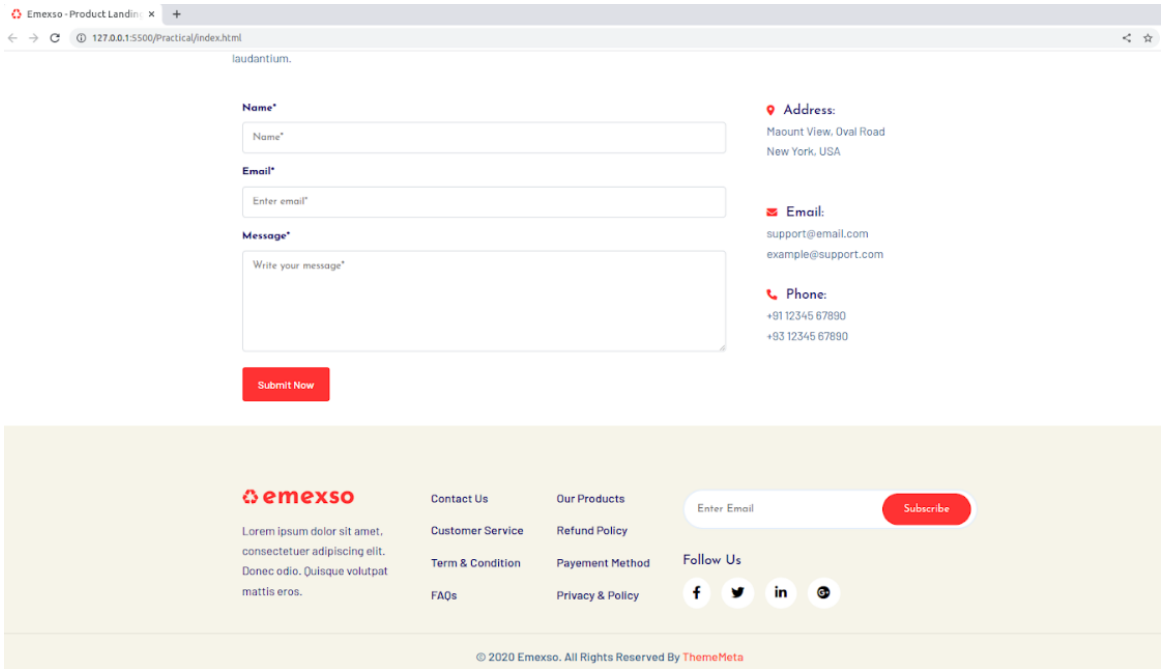
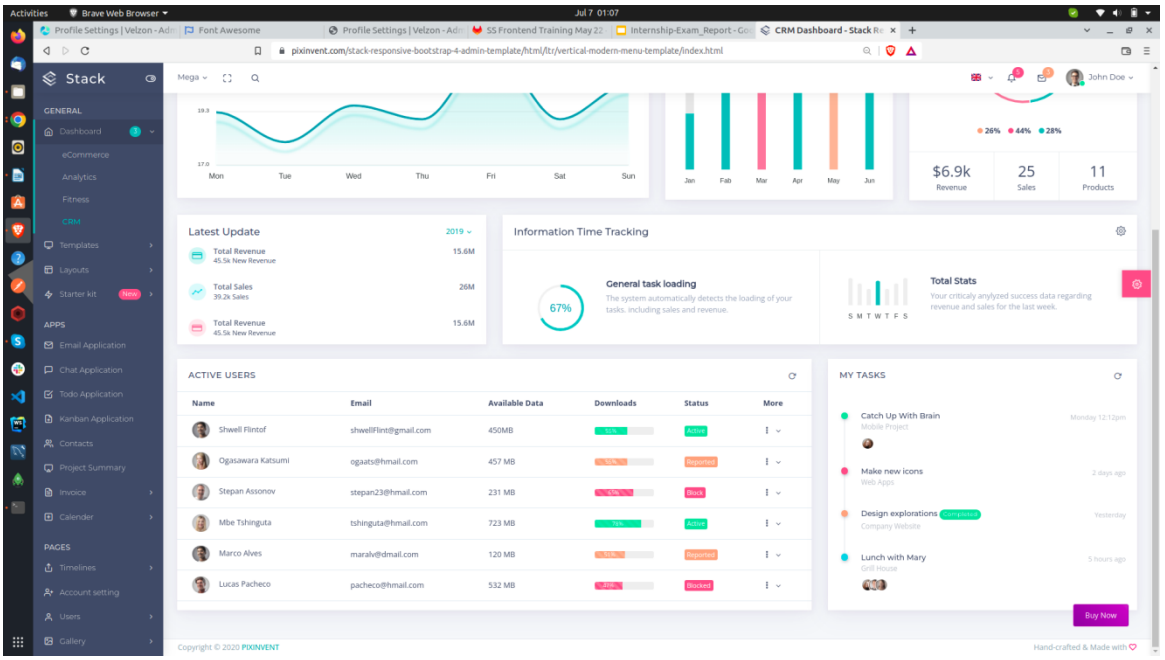


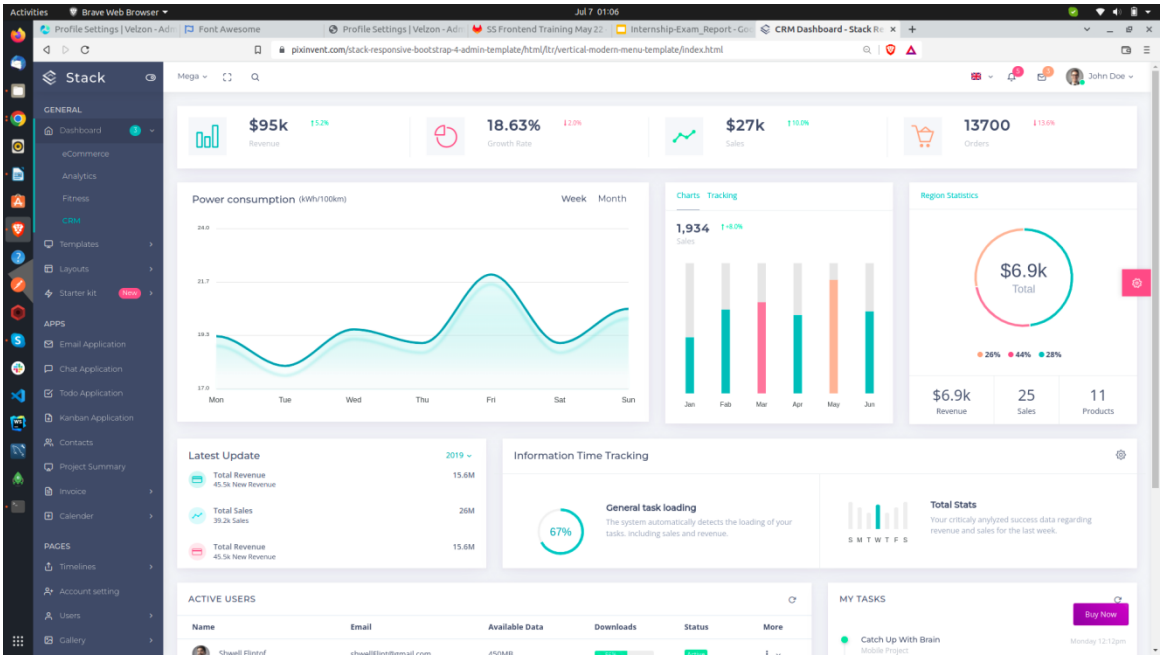
Fig 5.15 Emexso Contact

5.6 CRM Dashboard-Stack Responsive



In fig 5.16 Dashboard Table

In this website all advance CSS concepts will cover.For table I used table tags only because grid becames complicated.



In fig 5.17 Dashboard

In fig 17 2 containers are taken to keep main container and sidebar in row so here I used Flex Concept of CSS.

6. FUTURE ENHANCEMENTS

In next Phase of intership I am going to learn following things:

1. Javascript
2. React JS
3. Angular JS
4. Backend Technologies

   And After Learning this things I will going to apply in this website as this websites on not giving client server architecture.

7. CONCLUSION

7.1 SELF ANALYSIS OF PROJECT VIABILITIES

Personally, I believe that the project design and concept are very implementable and feasible; in addition to being of great importance, we believe that the suggested product fulfils almost every expectation and extracts all the many sorts of features required for Web[1]. In addition, it is It is apparent from the research done so far in this field that, in comparison, the web application designed for such a software decreases the overall time developers spend in setting up the environment.

7.2 SUMMARY OF PROJECT WORK

Our overall internship experience at smartSense was educational, and we learned a lot about various technologies. Working with a company and my coworkers was an eye-opening experience.

Thanks to VSCode services, the project are web without having all the resources locally. The user can create a project, add models, and case files to it, edit, and do any of the activities they need using the web application's integrated services.

Overall, it was a very informative and enjoyable experience. The work environment and team experience were fantastic, and we hope to work with them again in the future.

References

1 https://sass-lang.com/documentation/at-rules/function

2 https://www.w3schools.com/

3 https://dzone.com/articles/introduction-of-scss

4 https://css-tricks.com/

5 https://developer.mozilla.org/en-US/

6 https://www.udemy.com/course/modern-html-css-from-the-beginning

7 https://www.futurelearn.com/info/blog/what-are-html-css-basics-of-coding