**PERSONAL BLOG WEBSITE**

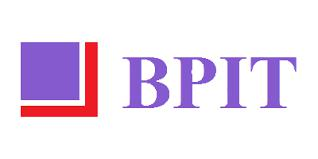
**WEB DEVELOPMENT BY CODING BLOCKS**

**Summer Training Report Submitted in partial fulfillment of the requirement for the degree of**

**B.Tech**

**In**

**Computer Science and Engineering**



Training Coordinator By

**Ms. Palak Girdhar Tripti Vijayan**

Bhagwan Parshuram Institute of Technology

Psp-4, Sec-17, Rohini, Delhi-89

**June – 2021 – August – 2021**

**DECLARATION**

This is to certify that Report titled “**Personal Blog Website**”, is submitted by me in partial fulfilment of the requirement for the award of degree B.Tech. in Computer Science & Engineering to BPIT, GGSIP University, Dwarka, Delhi. It comprises of my original work. The due acknowledgement has been made in the report for using others work.

**Date: Name of Student: Tripti Vijayan**

**ACKNOWLEDGEMENT**

Presentation inspiration and motivation have always played a key role in the success of any venture. The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning the people who made it possible.

I would also like to express thanks to my course mentor **Mr. Arnav Gupta,** for his guidance and valuable mentorship which helped me to grasp and apply new concepts of Web Development. I am extremely thankful for Coding Blocks Private Limited for designing such a course.

I am feeling obliged in taking the opportunity to sincerely thank our project guide and mentor **Prof. Achal Kaushik** for providing timely assistance to our query and guidance that he gave owing to his experience in this field for the past many years. He has indeed been a lighthouse for us in this journey.

We extend our sincere appreciation to all our Professors from BPIT for their valuable inside and tip during the designing of the project. Their contributions have been valuable in so many ways that we find it difficult to acknowledge them individually.

I am also grateful to our **Principal Prof. Payal Pahwa** for extending their help directly and indirectly through various channels in our project work.

At last I would also like to extend our gratitude to my family and all the people who have contributed to the successful completion of the project.

**COMPLETION CERTIFICATE**

****

# TRAINING COORDINATOR CERTIFICATE

This is to certify that Report titled “**Personal Blog Website**” which is submitted by Tripti Vijayan in partial fulfilment of the requirement for the award of degree B.Tech. in Computer Engineering to BPIT, GGSIP University, Dwarka, Delhi is a record of the candidate own work and the matter embodied in this report is adhered to the given format.

## Date: Coordinator

# TABLE OF CONTENTS

[Declaration II](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_bookmark0)

[Acknowledgement III](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_bookmark1)

[Completion Certificate IV](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_TOC_250012)

[Training Coordinator Certificate V](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_bookmark2)

[Abstract](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_bookmark3) 8

CHAPTERS

1. INTRODUCTION
   1. Introduction to Web Development 9
   2. Introduction to Project 10
   3. Scope of the project 10
   4. Product / Services & process 11
2. IMPLEMENTATION
   1. Hardware Used……………………………………………………………………13
   2. Software Used…………………………………………………………………….13
3. LANGUAGES AND TOOLS USED
   1. Visual Studio Code……………………………………………………………….13
   2. Hypertext Markup Language (HTML)…………………………………………..14
      1. Anatomy of HTML element 14
      2. Anatomy of HTML document 15
      3. Versions of HTML 16
   3. Web Browser……………………………………………………………………..19
   4. CSS: Cascading Style Sheets……………………………………………………..19
      1. History of CSS………………………………………………………………20
      2. Variations of CSS……………………………………………………………20
      3. Features of CSS3…………………………………………………………….21
         1. Borders 21
         2. Fonts 21
         3. Background Effects 21
         4. Text Effects 22
         5. 2D Transformations 23
         6. 3D Transformations 23
         7. Animations 23
         8. Transitions 23
         9. Browser Compatibility 24
   5. Bootstrap 24

3.5.1. History of Bootstrap………………………………………………………24

3.5.2. Features of Bootstrap……………………………………………………...25

* 1. Javascript…………………………………………………………………………26

3.6.1. History of Javascript……………………………………………………….26

3.6.2. Features of Javascript………………………………………………………27

* 1. Node.js .28

3.7.1. npm packages installed……………………………………………………..28

* 1. Express……………………………………………………………………………29
  2. EJS………………………………………………………………………………..30

3.9.1. ejs tags……………………………………………………………………..30

3.10. Git and GitHub…………………………………………………………..………30

1. SOFTWARE ANALYSIS AND DESIGN………………………………………33
   1. Software Requirement Specifications (SRS)……………………………………33

4.1.1. [Introduction 33](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_TOC_250011)

4.1.1.1 Purpose……………………………………………………………………33

4.1.1.2 Scope………………………………………………………………………34

4.1.2. [Software Requirements](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_TOC_250010)…………………………………………………….34

4.1.2.1. [Functional Requirements 34](file:///C:\Users\tript\Downloads\STP_Report21_RajatVerma.docx#_TOC_250009)

4.1.2.2. Non-Functional Requirements……………………………………………35

1. WEBSITE LAYOUT…………………………………………………………….38
2. BIBLIOGRAPHY & REFERENCES…………………………………………..41

**ABSTRACT**

I have completed my online training from an online course made available by Coding Blocks India Pvt Ltd. This Web Development course included introduction to Node.js and express.js as well as EJS, along with the utilization of HTML, CSS, Bootstrap. Throughout the duration of the course, I learnt more in depth about various Web Development Technologies.

Web development is the work involved in developing a website for the Internet or an intranet. It is a dynamic, never dying field which constantly undergoes innovations and transformations in form of new tools being developed every day for improving overall user experience. Web development can range from developing a simple single static page of plain text to complex web applications, electronic businesses, and social network services. One such web application built here is our Personal Blog Web Application. This Application can be used as a Digital Personal Daily Journal.

The **Daily Journal** or **Blog** can assist the user to concentrate on their other activities rather concentrate on the record-keeping. It can help user keep track of their daily activities. It can be used for personal as well as professional purposes.

Although designed for personal use, with further modifications and advancements, this web application can be used on a larger scale by organizations to keep track of their activities.

The project is primarily an implementation of backend web development services express.js with node.js and ejs. Its primary purpose is to digitize journaling and be used as a personal blog website.

**CHAPTER 1**

**INTRODUCTION**

1. **Introduction to Web Development**

[Web development](https://careerfoundry.com/en/events/introduction-to-web-development/) is the process of building websites and applications for the internet, or for a private network known as an intranet. Web development is not concerned with the design of a website; rather, it’s all about the coding and programming that powers the website’s functionality.

From the most simple, static web pages to social media platforms and apps, from e-commerce websites to content management systems (CMS); all the tools we use via the internet on a daily basis have been built by web developers.

Web development can be broken down into three layers: client-side coding (frontend), server-side coding (backend) and database technology.

Let’s look into the three layers with a little more detail.

* **Client Side:** Client-side scripting, or frontend development, refers to everything that the end user experiences directly. Client-side code executes in a web browser and directly relates to what people see when they visit a website. Things like layout, fonts, colours, menus and contact forms are all driven by the frontend.
* **Server Side:** Server-side scripting, or backend development, is all about what goes on behind the scenes. The backend is essentially the part of a website that the user doesn’t actually see. It is responsible for storing and organizing data, and ensuring that everything on the client-side runs smoothly. It does this by communicating with the frontend.

Whenever something happens on the client-side—say, a user fills out a form—the browser sends a request to the server-side. The server-side “responds” with relevant information in the form of frontend code that the browser can then interpret and display.

* **Database Technology:** Websites also rely on database technology. The database contains all the files and content that are necessary for a website to function, storing it in such a way that makes it easy to retrieve, organize, edit, and save. The database runs on a server, and most websites typically use some form of relational database management system (RDBMS).

1. **Introduction to the Project**

In the Personal Blog/Daily Journal Project, we created web application using HTML, CSS, Javascript and Bootstrap on the front end and Node.js with express.js and EJS on the backend.

The **Daily Journal** allows user to Compose a post with a Title and publish it on the Posts Page. The website is hosted locally. The project website is intended to be used as a personal daily journal to keep track of daily activities. It can be used as a mood tracker as well as habit tracker. It can also be used as a blog website for posting blog posts about a certain topic.

1. **Scope of the Project**

* As we know digital world would be more demanding in future even nowadays when many people engage with social sites or other internet activities, we can understand the value of its in upcoming time.  
  There are many option for many fields as far as blogging is concerned it is a great idea without any doubt, blogging would be nice option if you have idea about writing.
* Currently, there are many online platforms where you can apply or register as a member and write blogs over there as over websites demand.  The young generation are now moving from static content to dynamic content like flash and updating information.
* Blogging is not just the way of writing effectively, it has transformed into a lucrative business involving in depth understanding of a domain, digital marketing and sales content writing can be learnt through the process of blogging. There are numerous companies having been successful based on the marketing skills.
* The newspaper websites have now also become like blogs. Most authors write on weekly basis and you can see the article of any blogger by clicking on author name. The blogging is more taking attention to tech news like iphone, ipad, laptops, ipod, mobiles and other technology news. I have seen some blogs are discussing about medical treatment and how to remain healthy and smart, so that mean you can write on other information also. The scope of blogging is bright as more and more users are getting access to the internet and computers.
* As we are seeing a continuous rise in the number of businesses going online, the demand for web designers and developers is also increasing by the minute. However, you need to be different in the sense that you should be able to help businesses create a unique presence online. The competitive nature of the market demands business websites to understand buyer behavior and be developed in a way that makes buyers decide in favor of buying from that business.
* With newer development methodologies like agile development, the time of development has come down quite significantly, and the quality of the final product has gone up a few notches as well. Web developers nowadays are more adept at using different technologies to develop websites that are in line with business needs and customer expectations.

# Product / Services & process

Web development generally works along with the following products/services:

# Web designing and Development

* Website is an online platform dedicated to a particular topic, ranging from entertainment to networking. It represents a centrally managed group of web pages, containing text, images, and all types of multi-media files.
* Its main purpose is to display content/offerings for its visitors. Customers find business information and product/service offerings on websites. Wordsmith shall hire qualified web designers and developers to create websites for clients.

# Website Localization

* Website localization is the process of adapting an existing website to local language and culture in the target market. It is the process of adapting a website into a different linguistic and cultural context involving much more than the simple translation of text.
* This modification process must reflect specific language and cultural preferences in the content, images and overall design and requirements of the site – all while maintaining the integrity of the website.

# SEO: Search Engine Optimization

* Search engine optimization (SEO) is the process of affecting the online visibility of a website or a web page in a web search engine's unpaid results—often referred to as "natural", "organic", or "earned" results.
* In general, the earlier (or higher ranked on the search results page), and more frequently a website appears in the search results list, the more visitors it will receive from the search engine's users; these visitors can then be converted into customers.

# Social Media Marketing Campaign

* Using various social media channels such as Face book, you tube, linked in, Instagram etc. to create business brand awareness and attract target customers. This involves creating multimedia content to engage customer base, running campaigns on social media for specific offers or promotions.

# Branding and Strategy consultation

* Brand development, promotion and consultation for various online marketing strategies suitable for a particular business.

**CHAPTER 2**

**IMPLEMENTATION**

**2.1 Hardware Used**

Laptop/PC

Processor : Intel® Core™2 Duo

RAM : 8.00 GB

**2.2 Software Used**

Operating System : Windows 10

Code Editor : Visual Studio Code

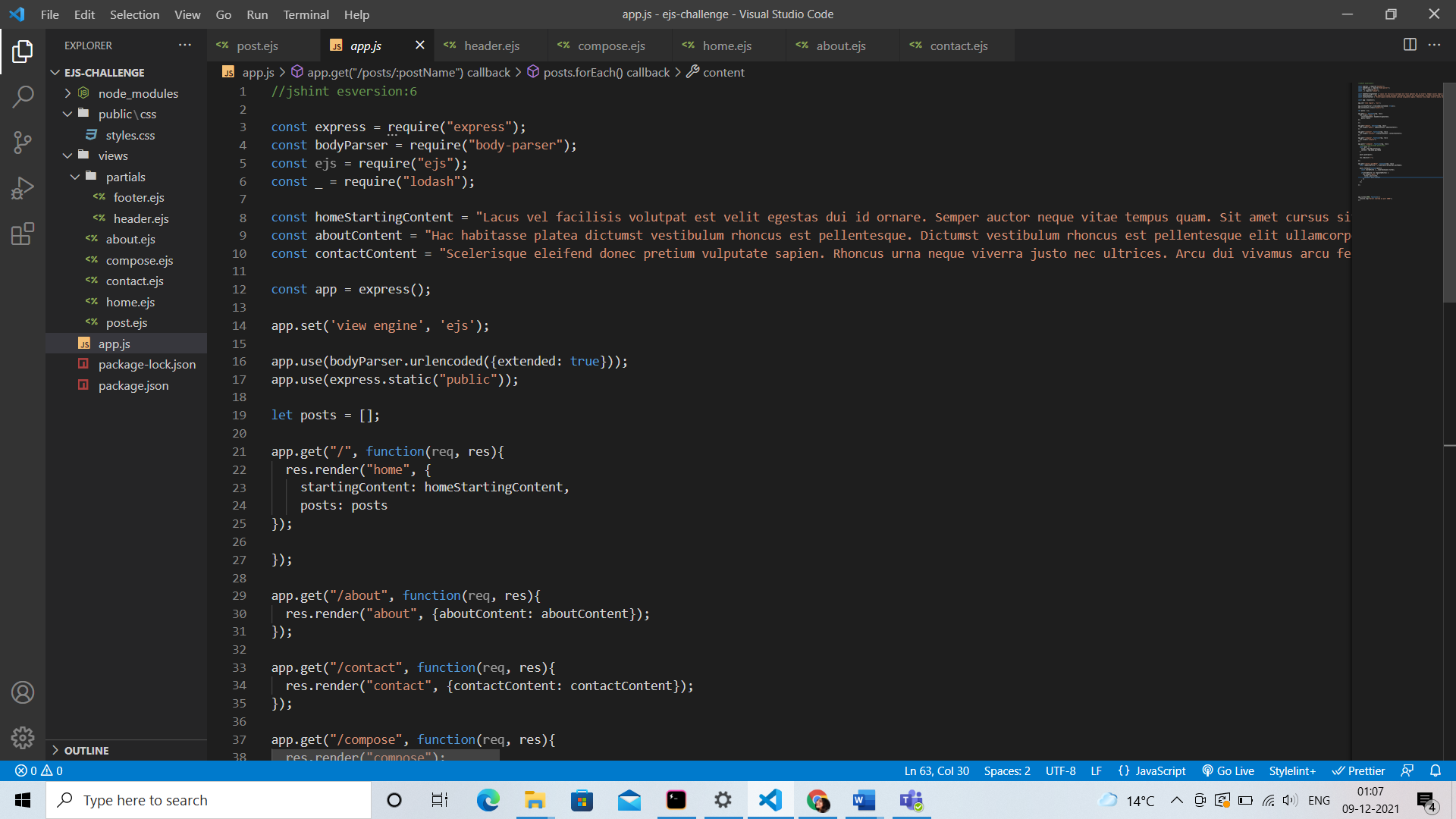
**CHAPTER 3**

**LANGUAGES AND TOOLS USED**

**3.1. Visual Studio Code**

About VS Code:

* Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.
* Microsoft has released Visual Studio Code's source code on the VSCode repository of GitHub.com, under the permissive MIT License, while the compiled binaries are freeware.



**3.2. Languages and Frameworks Used:**

Many languages and frameworks were used in the making of this project, as listed below:

1. **HyperText Markup Language (HTML)**

**About HTML:**

* + HTML is a markup language that defines the structure of your content. HTML consists of a series of **elements**, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or image hyperlink to somewhere else, can italicize words, can make the font bigger or smaller, and so on.
  + HTML (**H**yper**t**ext **M**arkup **L**anguage) is the code that is used to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables.
  + First developed by Tim Berners-Lee in 1990, HTML is used to create electronic documents (called pages) that are displayed on the World Wide Web. Each page contains a series of connections to other pages called hyperlinks. Every web page you see on the Internet is written using one version of HTML code or another.
  1. **Anatomy of an HTML Element**

An HTML element is defined by a start tag, some content, and an end tag:

<tagname>Content goes here...</tagname>

Take, for example the following line of code:



The main parts of our element are as follows:

1. **The opening tag:** This consists of the name of the element (in this case, p), wrapped in opening and closing **angle brackets**. This states where the element begins or starts to take effect — in this case where the paragraph begins.
2. **The closing tag:** This is the same as the opening tag, except that it includes a *forward slash* before the element name. This states where the element ends — in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.
3. **The content:** This is the content of the element, which in this case, is just text.
4. **The element:** The opening tag, the closing tag, and the content together comprise the element.



Attributes contain extra information about the element that you don't want to appear in the actual content. Here, class is the attribute *name* and editor-note is the attribute *value*. The class attribute allows you to give the element a non-unique identifier that can be used to target it (and any other elements with the same class value) with style information and other things.

An attribute should always have the following:

1. A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
2. The attribute name followed by an equal sign.
3. The attribute value wrapped by opening and closing quotation marks
   1. **Anatomy of an HTML Document**

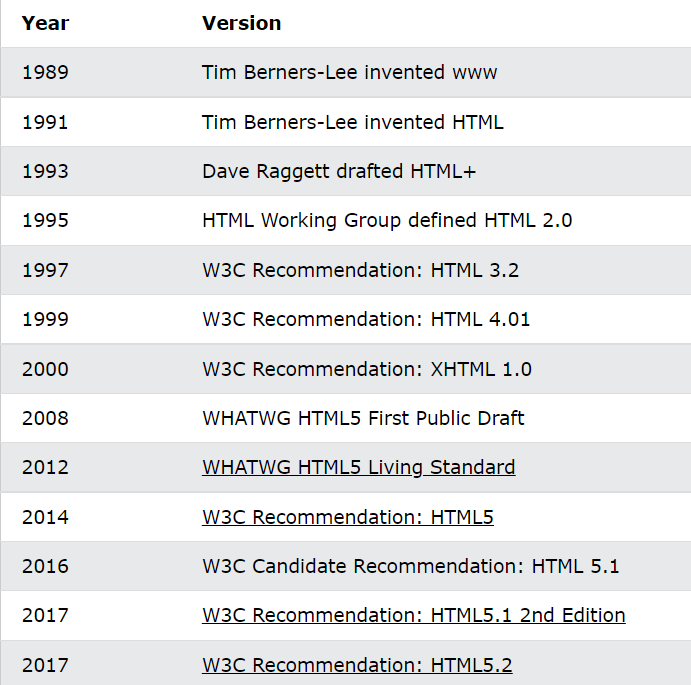
Now we'll look at how individual elements are combined to form an entire HTML page. Take for example, the following piece of code:

****

Here, we have the following:

* <!DOCTYPE html> — doctype. It is a required preamble. In the mists of time, when HTML was young (around 1991/92), doctypes were meant to act as links to a set of rules that the HTML page had to follow to be considered good HTML, which could mean automatic error checking and other useful things. However these days, they don't do much and are basically just needed to make sure your document behaves correctly. That's all you need to know for now.
* <html></html> — the [<html>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/html) element. This element wraps all the content on the entire page and is sometimes known as the root element.
* <head></head> — the [<head>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/head) element. This element acts as a container for all the stuff you want to include on the HTML page that isn't the content you are showing to your page's viewers. This includes things like [keywords](https://developer.mozilla.org/en-US/docs/Glossary/Keyword) and a page description that you want to appear in search results, CSS to style our content, character set declarations, and more.
* <meta charset="utf-8"> — This element sets the character set your document should use to UTF-8 which includes most characters from the vast majority of written languages. Essentially, it can now handle any textual content you might put on it. There is no reason not to set this and it can help avoid some problems later on.
* <title></title> — the [<title>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/title) element. This sets the title of your page, which is the title that appears in the browser tab the page is loaded in. It is also used to describe the page when you bookmark/favorite it.
* <body></body> — the [<body>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/body) element. This contains all the content that you want to show to web users when they visit your page, whether that's text, images, videos, games, playable audio tracks, or whatever else.
  1. **Versions of HTML**

Since the early days of the World Wide Web, there have been many versions of HTML:

****

Let’s see the significance of the individual Versions of Html in detail-

#### 1. HTML 1.0

* The basic version of HTML has support for basic elements like text controls and images. This was the very basic version of HTML with less support for a wide range of HTML elements. It does not have rich features like styling and other things that were related to how content will be rendered in a browser.
* The initial version of HTML does not provide support for tables, font support, etc., as it provides us in the latest version.
* We would also like to discuss that W3C did not exist before HTML 2.0; hence it does not show details about HTML 1.

#### 2. HTML 2

* HTML version 2.0 was developed in 1995 with basic intention of improving HTML version 1.0
* Now a standard got started to develop so as to maintain common rules and regulations across different browsers. HTML 2.0 has improved a lot in terms of the markup tags. In HTML 2.0 version concept of form came into force. Forms were developed, but still, they had basic tags like text boxes, buttons, etc.
* Also, the table came as an [HTML tag](https://www.educba.com/basic-html-tags/). Now, in HTML tag 2.0, browsers also came with the concept of creating their own layers of tags that were specific to the browser itself. W3C was also formed. The main intention of W3C is to maintain standard across different web browsers so that these browsers understand and render HTML tags in a similar manner.

#### 3. HTML 3.2

* It was developed in 1997. After HTML 2.0 was developed, the next version of HTML was 3.2
* With version 3.2 of HTML, HTML tags were further improved. It is worth noting that because of W3C standard maintenance, the newer version of HTML was 3.2 instead of 3.
* Now, HTML 3.2 has better support for new form elements. Another important feature what HTML 3.2 implemented was support for CSS. CSS stands for [Cascading Style Sheet](https://www.educba.com/what-is-css/). It is CSS that provides features to make HTML tags look better on rendering it on browsers. CSS helps to style HTML elements.
* With the upgradation of browsers to HTML 3.2, the browser also supported for [frame tags](https://www.educba.com/frame-tag-in-html/), although HTML specifications still do not support frame markup tags.

#### 4. HTML 4.01

* It was developed in 1999. It extended the support of cascading styling sheets. In version 3.2, CSS were embedded in HTML page itself. Therefore, if the website has various web pages to apply to the style of each page, we must place CSS on each web page. Hence there was a repetition of the same block of CSS.
* To overcome this thing, in version 4.01 concept of an external styling sheet emerged. Under this concept, an external CSS file could be developed, and this external styling file could be included in HTML itself. HTML 4.01 provided support for further new tags of HTML.

#### 5. HTML5

* This is the latest version of HTML. For a developer, it could be used in 2014. It came up with lots of HTML tags support. [HTML5](https://www.educba.com/html5-elements/) provided support for new form elements like input element s of different types; geolocations[support tags](https://www.educba.com/html-geolocation/), etc.

We are currently using HTML5 for our work.

More about HTML5:

* As HTML5 is the update made to HTML from HTML4 (XHTML follows a different version numbering scheme). It uses the same basic rules as HTML4, but adds some new tags and attributes which allow for better semantics and for dynamic elements that are activated using JavaScript. New elements added include:

## <article>, <aside>, <audio>, <bdi>, <canvas>, <datalist>, <details>, <embed>,

**<figure>, <figcaption>, <footer>, <header>, <keygen>, <mark>, <meter>,**

**<nav>, <output>, <progress>, <rp>, <rt>, <ruby>, <time>, <track>, <video>,**

and **<wbr>**.

* With the increasing movement to keep structure and style separate, some styling elements were removed, along with those that had accessibility issues or saw very little use. These following elements have been removed:

## <acronym>, <applet>, <basefont>, <big>, <center>, <dir>, <font>, <frame>,

**<frameset>, <noframes>, <strike>,** and **<tt>.**

* HTML5 also simplifies the doctype declaration.

1. **Web Browser**

* The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read HTML documents and display them correctly.
* A browser does not display the HTML tags, but uses them to determine how to display the document:



1. **Cascading Style Sheets (CSS)**

**About CSS:**

* CSS was first proposed by Håkon Wium Lie on October 10, 1994. At the time, Lie was working with Tim Berners-Lee at CERN. Style sheets have existed in one form or another since the beginnings of Standard Generalized Markup Language (SGML) in the 1980s, and CSS was developed to provide style sheets for the web.
* **Cascading Style Sheets** (**CSS**) is a [style sheet language](https://en.wikipedia.org/wiki/Style_sheet_language) used for describing the [presentation](https://en.wikipedia.org/wiki/Presentation_semantics) of a document written in a [markup language](https://en.wikipedia.org/wiki/Markup_language) such as [HTML](https://en.wikipedia.org/wiki/HTML). CSS is a cornerstone technology of the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), alongside HTML and [JavaScript](https://en.wikipedia.org/wiki/JavaScript).
* CSS is designed to enable the separation of presentation and content, including [layout](https://en.wikipedia.org/wiki/Page_layout), [colors](https://en.wikipedia.org/wiki/Color" \o "Color), and [fonts](https://en.wikipedia.org/wiki/Typeface).  This separation can improve content [accessibility](https://en.wikipedia.org/wiki/Accessibility); provide more flexibility and control in the specification of presentation characteristics; enable multiple [web pages](https://en.wikipedia.org/wiki/Web_page) to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be [cached](https://en.wikipedia.org/wiki/Cache_(computing)) to improve the page load speed between the pages that share the file and its formatting.
* Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or [screen reader](https://en.wikipedia.org/wiki/Screen_reader)), and on [Braille-based](https://en.wikipedia.org/wiki/Braille_display) tactile devices. CSS also has rules for alternate formatting if the content is accessed on a [mobile device](https://en.wikipedia.org/wiki/Mobile_device).

**3.1. History of CSS**

* CSS was first proposed by Håkon Wium Lie on October 10, 1994. At the time, Lie was working with Tim Berners-Lee at CERN. Several other style sheet languages for the web were proposed around the same time, and discussions on public mailing lists and inside World Wide Web Consortium resulted in the first W3C CSS Recommendation (CSS1) being released in 1996. In particular, a proposal by Bert Bos was influential; he became co-author of CSS1, and is regarded as co-creator of CSS.

**3.2. Variations of CSS**

CSS has various levels and profiles. Each level of CSS builds upon the last, typically adding new features and typically denoted as CSS 1, CSS 2, CSS 3, and CSS 4. Profiles are typically a subset of one or more levels of CSS built for a particular device or user interface. Currently there are profiles for mobile devices, printers, and television sets. Profiles should not be confused with media types, which were added in CSS 2.

#### **CSS 1**

The first CSS specification to become an official W3C Recommendation is CSS level 1, published on December 17, 1996. [Håkon Wium Lie](https://en.wikipedia.org/wiki/H%C3%A5kon_Wium_Lie" \o "Håkon Wium Lie) and [Bert Bos](https://en.wikipedia.org/wiki/Bert_Bos) are credited as the original developers. Among its capabilities are support for

* [Font](https://en.wikipedia.org/wiki/Typeface) properties such as typeface and emphasis
* Color of text, backgrounds, and other elements
* Text attributes such as spacing between words, letters, and lines of text
* [Alignment](https://en.wikipedia.org/wiki/Alignment_(typesetting)) of text, images, [tables](https://en.wikipedia.org/wiki/Table_(HTML)) and other elements
* Margin, border, padding, and positioning for most elements
* Unique identification and generic classification of groups of attributes

The W3C no longer maintains the CSS 1 Recommendation.

#### **CSS 2**

CSS level 2 specification was developed by the W3C and published as a recommendation in May 1998. A superset of CSS 1, CSS 2 includes a number of new capabilities like absolute, relative, and fixed positioning of elements and [z-index](https://en.wikipedia.org/wiki/Z-index), the concept of media types, support for aural style sheets (which were later replaced by the CSS 3 speech modules) and bidirectional text, and new font properties such as shadows.

The W3C no longer maintains the CSS 2 recommendation.

#### CSS 2.1

CSS level 2 revision 1, often referred to as "CSS 2.1", fixes errors in CSS 2, removes poorly supported or not fully interoperable features and adds already implemented browser extensions to the specification. To comply with the W3C Process for standardizing technical specifications, CSS 2.1 went back and forth between Working Draft status and Candidate Recommendation status for many years. CSS 2.1 first became a [Candidate Recommendation](http://www.w3.org/TR/2004/CR-CSS21-20040225/) on February 25, 2004, but it was reverted to a Working Draft on June 13, 2005 for further review. It returned to Candidate Recommendation on 19 July 2007 and then updated twice in 2009. However, because changes and clarifications were made, it again went back to Last Call Working Draft on 7 December 2010.

CSS 2.1 went to Proposed Recommendation on 12 April 2011. After being reviewed by the W3C Advisory Committee, it was finally published as a W3C Recommendation on 7 June 2011

CSS 2.1 was planned as the first and final revision of level 2—but low priority work on CSS 2.2 began in 2015.

#### **CSS 3**

Unlike CSS 2, which is a large single specification defining various features, CSS 3 is divided into several separate documents called "modules". Each module adds new capabilities or extends features defined in CSS 2, preserving backward compatibility. Work on CSS level 3 started around the time of publication of the original CSS 2 recommendation. The earliest CSS 3 drafts were published in June 1999.

**3.3 Features of CSS3**

1. **CSS3 Borders**
   * CSS3 contain following border properties:

1. border-radius

2. box-shadow

3. border-image

* + With CSS3, you can create rounded borders, add shadow to boxes, and use an image as a border - without using a design program, like Photoshop.
  + The CSS3 border-radius property allows web developers to easily utilise rounder corners in their design elements, without the need for corner images or the use of multiple div tags and is perhaps one of the most talked about aspects of CSS3.

# CSS3 Fonts

* + Before CSS3, web designers had to use fonts that were already installed on the user's computer. With CSS3, web designers can use whatever font he/she likes. When you have found/bought the font you wish to use, include the font file on your web server, and it will be automatically downloaded to the user when needed.

# CSS3 Background Effects

* + CSS3 contains the following background properties:

1. background-size

2. background-origin

* + CSS3 contains several new background properties, which allow greater control of the background element. CSS3 allows web designers to specify multiple background images for box elements, using nothing more than a simple comma-separated list. The property adds new functionality to CSS allowing designers to specify the size of background images using either lengths, percentages, or by using one of two keywords; contain or cover.

# CSS3 Text Effects

* + CSS3 contain following Text Effect properties:

1. text-shadow

2. word-wrap

* + The new CSS3 properties give developers a wonderful chance to enhance their designs in a way that‘s quick and easy, yet visually impressive. To give a website a visually impressive look, designers put emphasis on neat and stylish typography. For years designers have depended on Photoshop, but CSS3 is a revolution with easy-to-create text effects. Almost all of the major browsers now support most of the CSS3 features so that‘s another reason for mastering the new techniques.

# 2D Transforms

* + A transform is an effect that lets an element change shape, size and position. CSS3 contain following 2D Transforms properties:
    - translate(), rotate(), scale(), skew(), matrix()

# 3D Transforms

* + CSS3 allows you to format your elements using 3D transforms. CSS3 contain following 3D transforms properties:
    - rotateX() and rotateY()

# CSS3 Animations

# Elements are moved across the screen with the assistance of JavaScript and flash. By using the elements, we also change the size and color. All kinds of transformations and animations are performed by using CSS3.

# Transitions

* We can add an effect when changing from one style to another, producing transition effects.

# The only real caveat of CSS3 is the fact that not all features are compatible in all browsers. CSS3 is constantly in development and browsers simply need time to catch up to all the latest features. However, the features mentioned above should work fine across the latest iterations of all the major browsers.

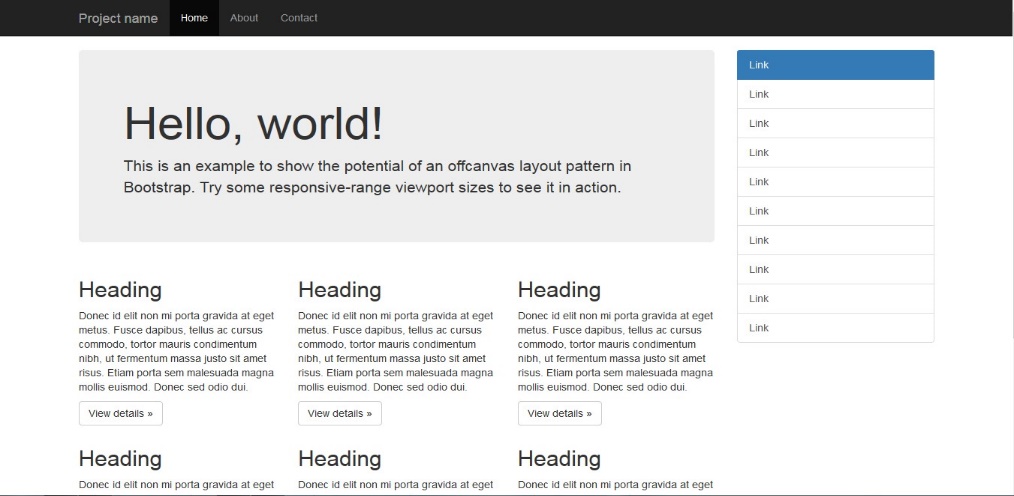
# A Quick Guide To CSS3 - Cheat Sheet and Browser Support | Witspry Tech-novate

1. **Bootstrap**

**About Bootstrap**

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

* 1. **History of Bootstrap**
* Bootstrap, originally named Twitter Blueprint, was developed by Mark Otto and Jacob Thornton at Twitter as a framework to encourage consistency across internal tools. Before Bootstrap, various libraries were used for interface development, which led to inconsistencies and a high maintenance burden. Bootstrap has now become one of the most popular front-end frameworks and open source projects in the world.
* It served as the style guide for internal tools development at the company for over a year before its public release, and continues to do so today.
* Originally released on  August 19, 2011, it has since had over twenty releases, including two major rewrites with v2 and v3. With Bootstrap 2, it added responsive functionality to the entire framework as an optional stylesheet. Building on that with Bootstrap 3, we rewrote the library once more to make it responsive by default with a mobile first approach.
* With Bootstrap 4, the developers once again rewrote the project to account for two key architectural changes: a migration to Sass and the move to CSS’s flexbox.
  1. **Features of Bootstrap**
* Bootstrap is an HTML, CSS & JS Library that focuses on simplifying the development of informative web pages (as opposed to [web apps](https://en.wikipedia.org/wiki/Web_Apps)). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all [HTML elements](https://en.wikipedia.org/wiki/HTML_element). The result is a uniform appearance for prose, tables and form elements across [web browsers](https://en.wikipedia.org/wiki/Web_browser). In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent [pull quotes](https://en.wikipedia.org/wiki/Pull_quote), and text with a highlight.
* Bootstrap also comes with several JavaScript components in the form of [jQuery](https://en.wikipedia.org/wiki/JQuery) plugins. They provide additional user interface elements such as [dialog boxes](https://en.wikipedia.org/wiki/Dialog_box), [tooltips](https://en.wikipedia.org/wiki/Tooltip), and carousels. Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields.



Example of a webpage using Bootstrap framework rendered in [Firefox](https://en.wikipedia.org/wiki/Firefox)

* The most prominent components of Bootstrap are its layout components, as they affect an entire web page. The basic layout component is called "Container", as every other element in the page is placed in it. Developers can choose between a fixed-width container and a fluid-width container. While the latter always fills the width of the web page, the former uses one of the five predefined fixed widths, depending on the size of the screen showing the page:
  + Smaller than 576 pixels
  + 576–768 pixels
  + 768–992 pixels
  + 992–1200 pixels
  + Larger than 1200 pixels
* Once a container is in place, other Bootstrap layout components implement a CSS Flexbox layout through defining rows and columns.
* A precompiled version of Bootstrap is available in the form of one CSS file and three JavaScript files that can be readily added to any project. The raw form of Bootstrap, however, enables developers to implement further customization and size optimizations. This raw form is modular, meaning that the developer can remove unneeded components, apply a theme and modify the uncompiled [Sass](https://en.wikipedia.org/wiki/Sass_(stylesheet_language)) files.

1. **Javascript**

**About Javascript:**

JavaScript is a cross-platform, object-oriented scripting language used to make webpages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.).  There are also more advanced server side versions of JavaScript such as Node.js, which allow you to add more functionality to a website than downloading files (such as realtime collaboration between multiple computers). Inside a host environment (for example, a web browser), JavaScript can be connected to the objects of its environment to provide programmatic control over them.

**5.1. History of Javascript**

* Brendan Eich created JavaScript in 1995 while he was at Netscape Communications Corporation, the creators of the legendary Netscape Navigator web browser.
* Back in 1995, Netscape Communicator (a paid internet browser) was by far the most popular web browser in the world. The founder of Netscape Communications, Marc Andreeseen, wanted to make the web more dynamic by making animations, user interaction, and other types of automation a standard part of any website.
* Before the introduction of JavaScript, web pages were static. To achieve any level interaction, multiple requests had to be made to the server, upon which, whole new documents would be generated, even for the smallest change. JavaScript was conceived as an embedded scripting language for the web browser by Brendan Eich at Netscape. It provided the means of manipulating HTML documents, circumventing the need for multiple page requests, thus providing richer experiences without delay or page refreshes. This facilitated functionality like client-side form validation, animation and reaction to user input (e.g. clicks and key presses). It first shipped with the Netscape browser in 1995, and from that point onward, every new browser came equipped with some version of JavaScript. Now a fully-fledged programming language, it is ubiquitous; since it is the language of the web, any device with a respectable form of web access will have a JavaScript implementation. This includes all major operating systems, and devices ranging from desktops to mobiles. Most browser vendors implement their own JavaScript engine—a piece of software that will interpret and execute JavaScript code, according to the ECMAScript1 specification

**5.2. Features of Javascript**

**1. Imperative and Structured**

JavaScript supports much of the structured programming syntax from C (e.g., if statements, while loops, switch statements, do while loops, etc.). One partial exception is scoping: originally JavaScript only had function scoping with var; then block scoping was added in ECMAScript 2015 with the keywords let and const. Like C, JavaScript makes a distinction between expressions and statements. One syntactic difference from C is automatic semicolon insertion, which allow semicolons (which terminate statements) to be omitted.

**2. Dynamic**

**Typing**

JavaScript is dynamically typed like most other scripting languages. A type is associated with a value rather than an expression. For example, a variable initially bound to a number may be reassigned to a string.  JavaScript supports various ways to test the type of objects, including duck typing.

**Run-time evaluation**

JavaScript includes an “eval” function that can execute statements provided as strings at run-time.

**3. Object Oriented**

Like most modern programming languages, JavaScript is object-oriented. Objects are data-structures- containers for state and behaviour. They facilitate many established software engineering practices including: abstraction—hiding complexity and implementation details; encapsulation—privacy and logical grouping of functionality; and modularity- separation of distinct functionality. In JavaScript, objects are unordered sets of named properties, which can be of various data types including other objects. Everything is an object, except the primitive types: undefined, null, boolean, string and number. Differing from objects, these primitives have only one value and can’t have properties- however, the latter three appear to have properties and can be interacted with like objects; this is because they have object wrappers and JavaScript will coerce them. Sometimes, the functionality of multiple objects will overlap. In order to facilitate code reuse in such cases, object-oriented languages generally offer inheritance mechanisms. Inheritance allows an object to inherit behaviour from another object, preventing duplication of their shared functionality. There are two prevailing types of inheritance: classical and prototypical. The classical approach has the concept of a class, which can be thought of as a blueprint or mould for objects; it defines the properties and behaviour for a certain type of object, and it can be used to create objects of that type. Inheritance is achieved by declaring that a class ‘extends’ another class. The prototypical approach is conceptually simpler in that objects are not created from classes, and they may inherit directly from other objects, but its use is less widespread and therefore the techniques are less well known. Most popular programming languages, including C++, Java, Ruby and Python use classical inheritance techniques. JavaScript is the only mainstream language the employs the prototypical method. In JavaScript, all objects have a hidden link to a prototype, which is another object whose behaviour and state they inherit and extend.

1. **Node.js**

**About Node js:**

* Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser.
* Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts.
* Though .js is the standard filename extension for JavaScript code, the name "Node.js" does not refer to a particular file in this context and is merely the name of the product. Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time Web applications

**6.1. npm packages used:**

* express

Express.js, or simply Express, is a back end web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs.

****

* body-parser

The bodyParser object exposes various factories to create middlewares. All middlewares will populate the req.body property with the parsed body when the Content-Type request header matches the type option, or an empty object ({}) if there was no body to parse, the Content-Type was not matched, or an error occurred.

****

* lodash

****

* **nodemon**

nodemon is a tool that helps develop node.js based applications by automatically restarting the node application when file changes in the directory are detected.

nodemon does **not** require any additional changes to your code or method of development. nodemon is a replacement wrapper for node. To use nodemon, replace the word node on the command line when executing your script.



1. **Express**

**About express**

* Express is basically a web framework for Node JS. The framework brings efficiency and excellence in the entire process of developing an application.
* It is a debugging mechanism, template engines, handles HTTP request and encompasses within itself numerous features that aid application development.
* Let’s understand Express in Node JS in a more detailed manner. A website dedicated to Express says that it is a fast, unopinionated and minimalist framework that delivers high performance and fast delivery. The framework has a couple of new features but makes sure that the features you are already familiar with don’t get lost with the advent of new ones. Express can be thought of as a layer of an additional and cool set of functionalities built on top of Node JS.
* When we say that Express is a minimalistic framework, we mean that the philosophy followed is 'Less is more.' Also, doesn't come in the way of a developer's expression.
* Another word used to describe Express, is unopinionated. An unopinionated framework means that it doesn’t come with a defined set of rules which a developer has to follow. In Express, developers have the freedom to augment their project with innumerable possibilities and minimal restrictions and create codes with the help of varied components together. Developers can add middleware packages, structure the application into one or multiple files or use any directory structure.
* More often than not, all other frameworks provide a robust set of features but are way too complicated for beginners. Express takes care that there is no excess of functions or unnecessary code as suggestion as Express is quite a flexible framework that allows developers to add features and functions only when they need to and not otherwise.

1. **EJS**

**About EJS**

EJS or Embedded Javascript Templating is a templating engine used by Node.js. Template engine helps to create an HTML template with minimal code. Also, it can inject data into HTML template at the client side and produce the final HTML. EJS is a simple templating language which is used to generate HTML markup with plain JavaScript. It also helps to embed JavaScript to HTML pages.

To begin with, using EJS as templating engine we need to install EJS using given command:

npm install ejs

**8.1. EJS Tags:**

* <% 'Scriptlet' tag, for control-flow, no output
* <%\_ ‘Whitespace Slurping’ Scriptlet tag, strips all whitespace before it
* <%= Outputs the value into the template (HTML escaped)
* <%- Outputs the unescaped value into the template
* <%# Comment tag, no execution, no output
* <%% Outputs a literal '<%'
* %> Plain ending tag
* -%> Trim-mode ('newline slurp') tag, trims following newline
* \_%> ‘Whitespace Slurping’ ending tag, removes all whitespace after it

1. **Git and GitHub**

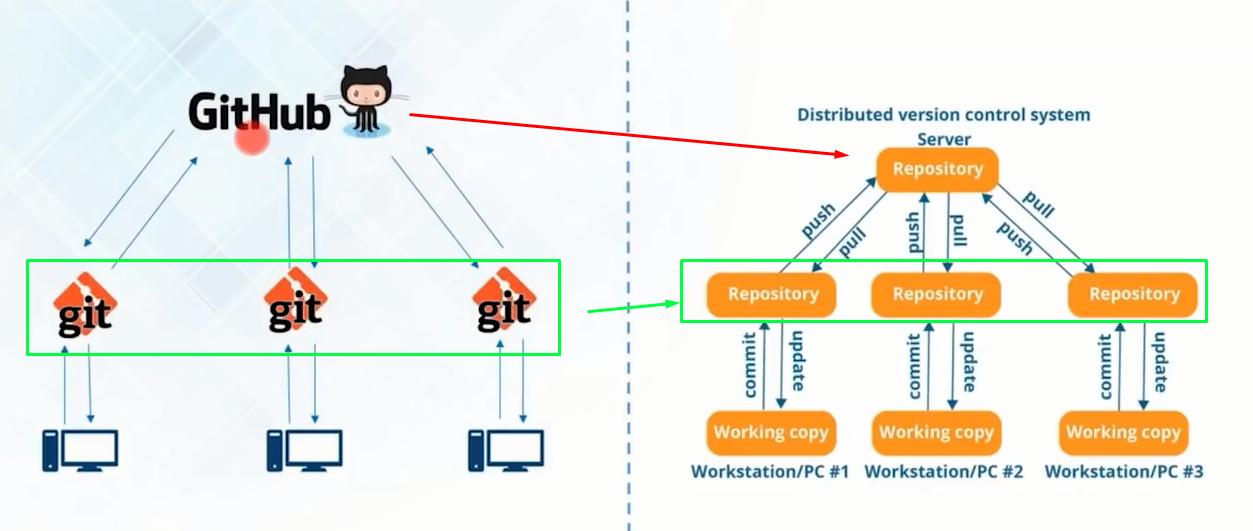
**Git:**

* First developed back in 2005, Git is an extremely popular version control system that is at the heart of a wide variety of high-profile projects. Git is installed and maintained on your local system (rather than in the cloud) and gives you a self- contained record of your ongoing programming versions. It can be used completely exclusive of any cloud-hosting service — you do not even need internet access, except to download it.
* Compared to other version control systems, Git is responsive, easy to use, and inexpensive (free, actually). Git is also specially designed to work well with text files — which, if you think about it, is what code actually is. But one thing that really sets Git apart is its branching model. Branching allows you to create independent local branches in your code. This means you can try out new ideas, set aside branches for production work, jump back to earlier branches, and easily delete, merge, and recall branches at the click of a button.

**GitHub**

* GitHub is designed as a Git repository hosting service, an online database that allows you to keep track of and share your Git version control projects outside of your local computer/server.
* Unlike Git, GitHub is exclusively cloud-based. Also unlike Git, GitHub is a for- profit service (although basic repository-hosting features are available at no cost to those who are willing to create a user profile, making GitHub a popular choice for open-source projects).
* That’s because, in addition to offering all of the features and advantages of Git, GitHub expands upon Git’s basic functionality. It presents an extremely intuitive, graphically represented user interface, and provides programmers with built-in control and task-management tools.
* Additional features can be implemented via the GitHub Marketplace service. And because GitHub is cloud-based, an individual’s Git repositories can be remotely accessed by any authorized person, from any computer, anywhere in the world (provided it has an internet connection).
* Through GitHub, you can share your code with others, giving them the power to make revisions or edits on your various Git branches. This makes it possible for entire teams to coordinate together on single projects in real-time. As changes are introduced, new branches are created, allowing the team to continue to revise the code without overwriting each other's work.
* These branches are like copies, and changes made on them do not reflect in the main directories on other users’ machines unless users choose to push/pull the

changes to incorporate them. There is also a GitHub desktop application available, which offers some additional functionality for experienced developers.



**CHAPTER 4**

**SOFTWARE ANALYSIS AND DESIGN**

**4.1 SOFTWARE REQUIREMENT SPECIFICATIONS (SRS)**

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders (business, users) needs.

A software requirements specification is the basis for your entire project. It lays the framework that every team involved in development will follow.

It’s used to provide critical information to multiple teams — development, quality assurance, operations, and maintenance. This keeps everyone on the same page.

Using the SRS helps to ensure requirements are fulfilled. And it can also help you make decisions about your product’s lifecycle — for instance, when to retire a feature. Writing an SRS also minimizes overall development time and costs.

Now, for large software systems, requirements analysis is perhaps the most difficult activity and the most error prone. The requirement analyst must identify the requirements by talking to these people and understanding their needs.

We will now look through the Software Requirement Specifications for our Blog/Daily Journal project.

**4.1.1 Introduction**

**4.1.1.1 Purpose**

The “Personal Blog Website” project is a model Blogging/Online Journaling Site. This site enables the users to create blog posts/online journal entries by sitting at their office or at home through a PC or laptop. The system provides the access to the customer to create an account, create blog posts, and also view their posts. The users can access the website for viewing their blogs as per their requirements. The brick-and-mortar structure of traditional banking gets converted into a click and portal model, thereby giving a concept of virtual banking a real shape. Online journalling/blogging transactions round the clock globally. The primary aim of this is to provide an improved design methodology, which envisages the future expansion, and modification

**4.1.1.2 Scope**

As we know digital world would be more demanding in future even nowadays when many people engage with social sites or other internet activities, we can understand the value of its in upcoming time.  
There are many option for many fields as far as blogging is concerned it is a great idea without any doubt, blogging would be nice option if you have idea about writing.

Currently, there are many online platforms where you can apply or register as a member and write blogs over there as over websites demand.  The young generation are now moving from static content to dynamic content like flash and updating information.

Blogging is not just the way of writing effectively, it has transformed into a lucrative business involving in depth understanding of a domain, digital marketing and sales content writing can be learnt through the process of blogging. There are numerous companies having been successful based on the marketing skills.

The newspaper websites have now also become like blogs. Most authors write on weekly basis and you can see the article of any blogger by clicking on author name. The blogging is more taking attention to tech news like iphone, ipad, laptops, ipod, mobiles and other technology news. I have seen some blogs are discussing about medical treatment and how to remain healthy and smart, so that mean you can write on other information also. The scope of blogging is bright as more and more users are getting access to the internet and computers.

**4.1.2 Software Requirements**

**4.1.2.1 Functional Requirements**

1. User Login- Since this web application is for personal use, a personal login portal may be created to enhance security and encryption.

This feature is used by the user to login into system. Usually the user is required to enter his/her credentials like username/email and password to log into the application. At each successful login user is required to redirect to the main website page where he can access other products.

*Functional requirements*

* + - * + Credentials must be provided when user registers.
        + The application must only allow user with valid credentials which is currently associated with specific user profile.
        + The application performs authorization and authentication process which decides what users can access or not.

2**.** Registering a new user- Register New User**-** This feature enhances the automation for user registration as application must check the user status itself (i.e. whether user is registered or not) and in case user is not holding a registered account then register the user for the very first time. *Functional requirements*

* + - * + App must be able to verify information.
        + App must be able to delete information if information is wrong and roll back.

**4.1.2.2 Non-Functional Requirements**

## EFFICIENCY REQUIREMENT

When this application system will be implemented admin and user will easily access the portal for searching, adding, updating product and profile.

## RELIABILITY REQUIREMENT

The system should accurately perform member registration, member validation, creating, posting and searching of posts.

## USABILITY REQUIREMENT

The system is designed for a user-friendly environment so that admin and user can perform the various tasks easily and in an effective way.

**4. Interface requirement**

**4.1. GUI**

This is interface must be highly intuitive or interactive because there will not be assistance for the user who is operating the System. At most places, a help desk should be provided for users’ convenience. The screens appearing should be designed in such a manner that it can draw User attraction towards the new plans for the customers. Also, the PIN and password confidentiality should be maintained, This can be done by using asterisks at the password panel. Proper security messages should be displayed at most places.

**4.2. Hardware Interface**

Various interfaces for the product could be

1. Touch screen/Monitor

2. Keypad

3. Printer which can produce the hard copy.

4. Interface that connects the device to the bank’s computer.

**4.3. Software Interface**

1. Any operating system.

2. For the database handling MYSQL must be installed.

3. The final application must be packaged in a setup program so that the products can be easily installed on machines. This application must be networked to corresponding banks

**5. Performance requirement**

In order to maintain an acceptable speed at the maximum number of uploads allowed from a particular customer, any number of users can access the system at any time.

Also, the connections to the servers will be based on the attributes of the user like his location and the server will be working 24X7 times.

**6. Design Requirements**

Following Non-Functional Requirements will be there in the insurance to the internet:

(i) Secure access to consumer’s confidential data

(ii) 24X7 availability.

(iii) Better component design to get better performance at peak time.

(iv) Flexible service-based architecture will be highly desirable for future extension.

**7. Performance**

7.1. Security The banking system must be fully accessible to only authentic users. It should require a PIN for entry to a new environment.

7.2. Reliability The application should be highly reliable and it should generate all the updated information in the correct order.

7.3. Availability Any information about the account should be quickly available from any computer to the authorized user. The previously visited customer’s data must not be cleared.

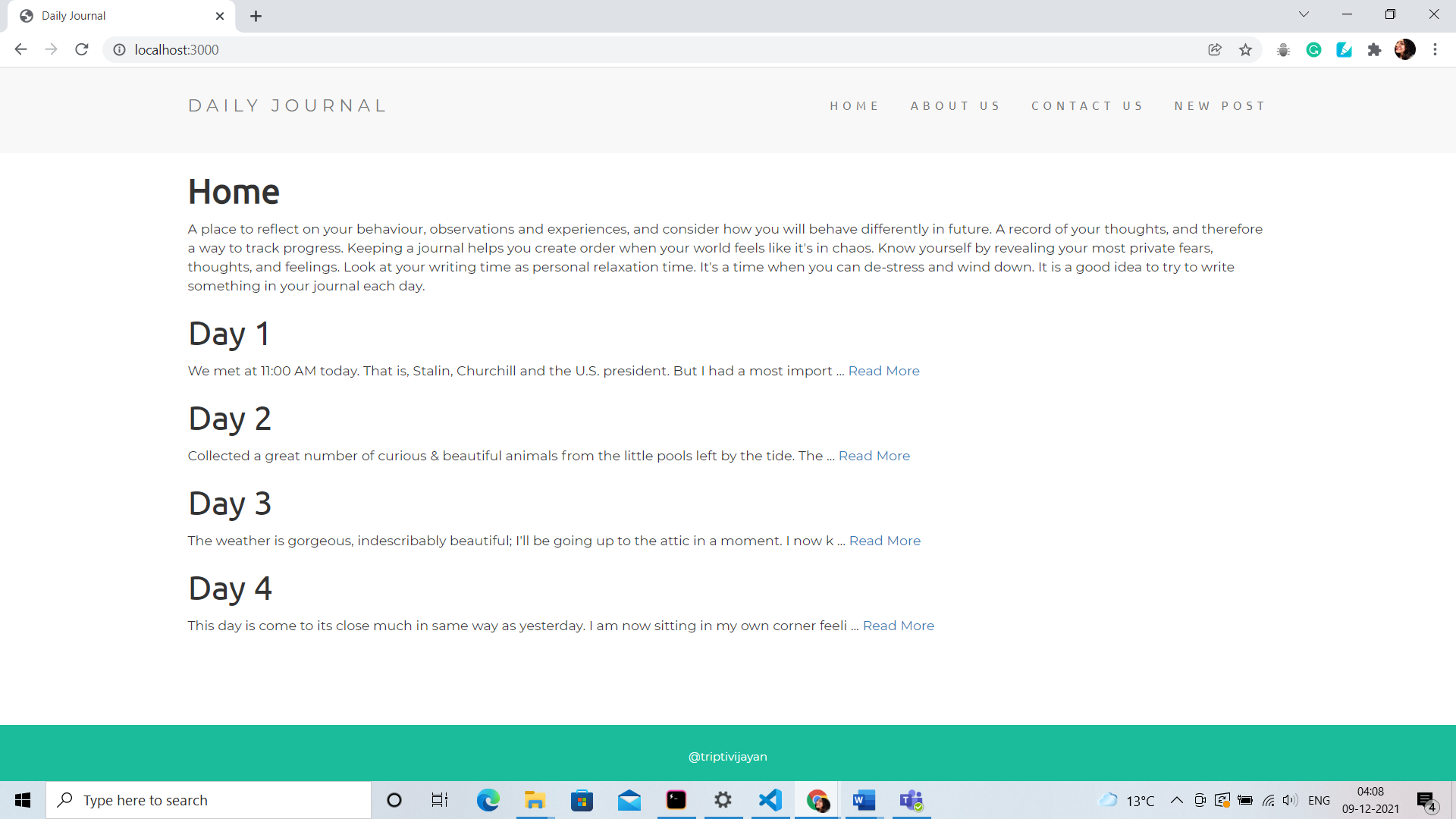
7.4. Maintenance The application should be maintainable in such a manner that if any new requirement occurs then it should be easily incorporated into an individual module.

7.5. Portability The application should be portable on any windows based system. It should not be machine-specific.

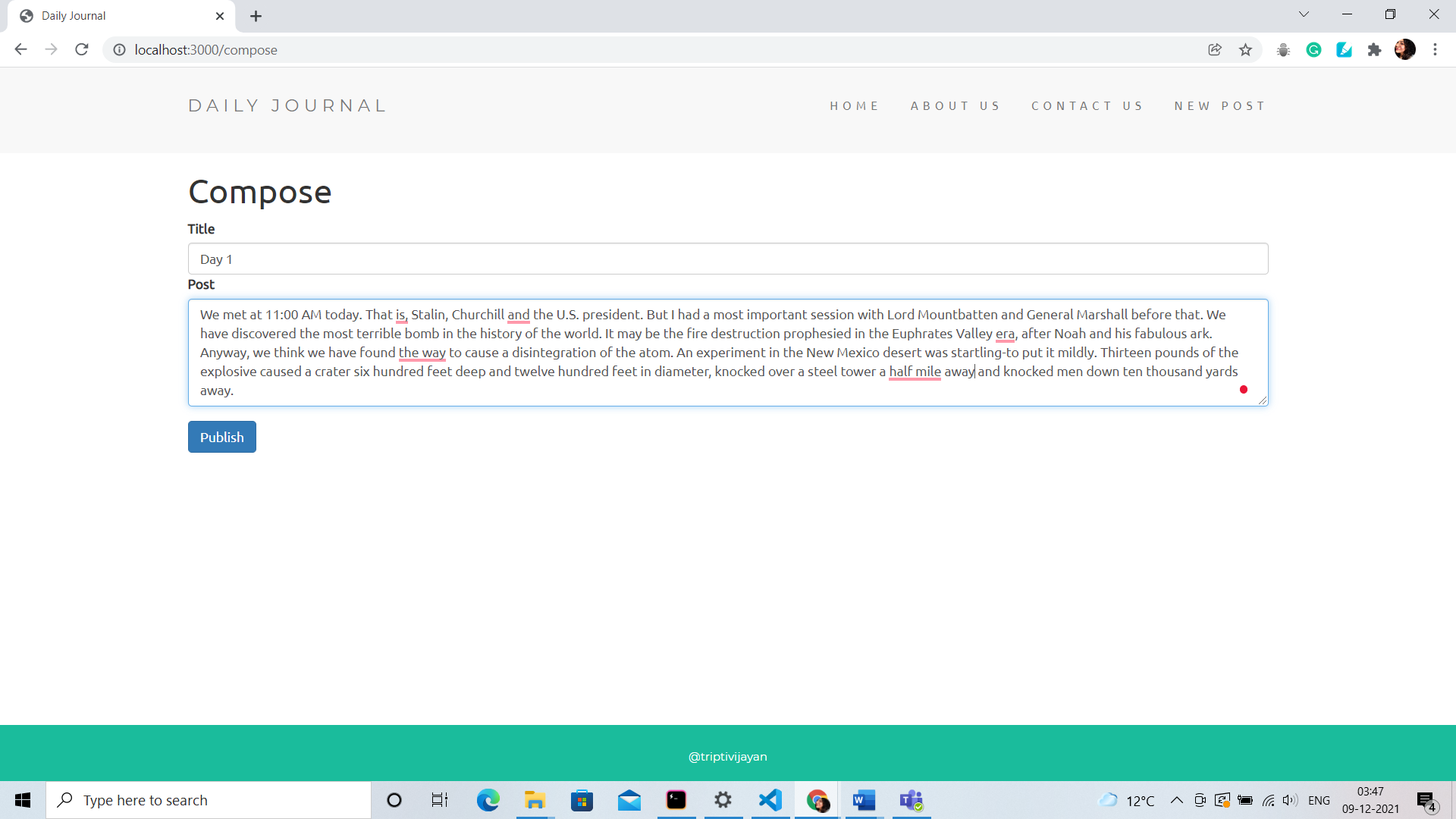
**CHAPTER 5**

**WEBSITE LAYOUT**

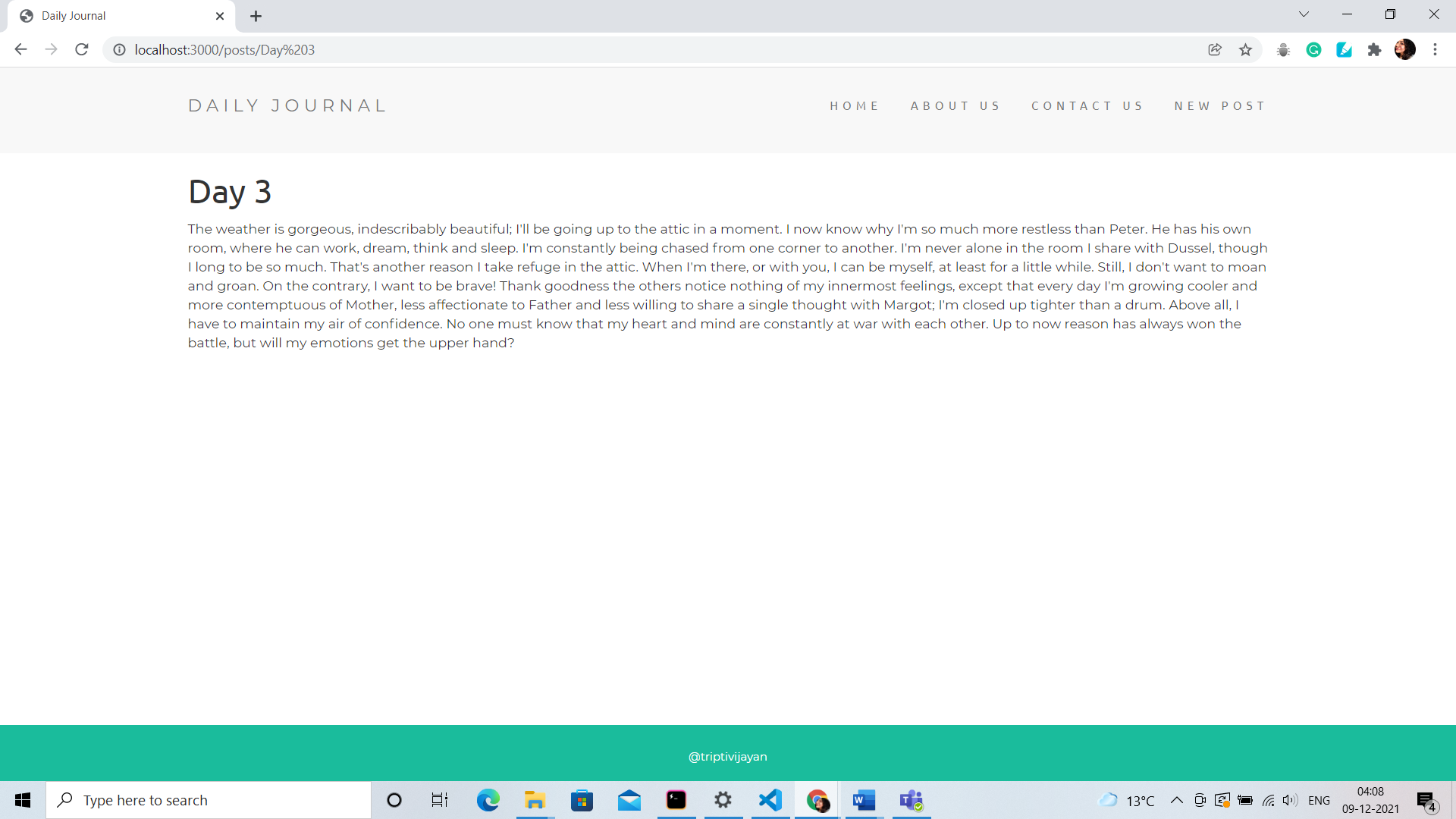
* 1. **Home Page with Daily Entries**



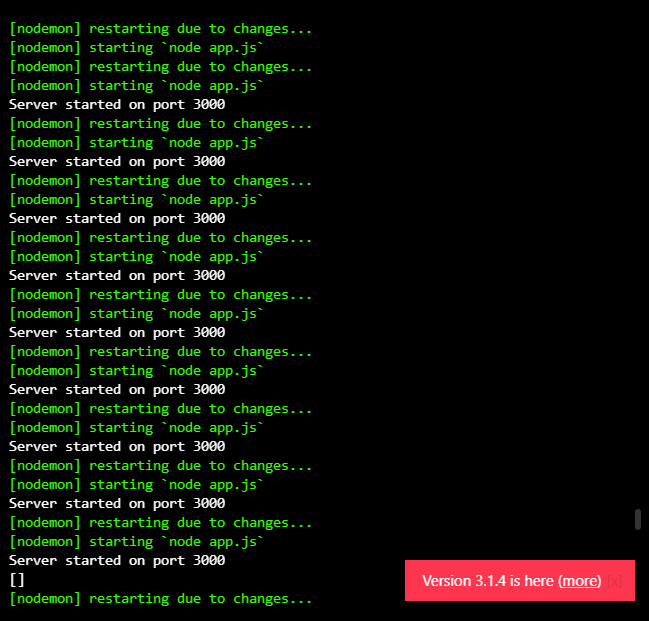
* 1. **Compose New Post Page**



* 1. **View a Post**



**Screenshot of the command line teminal**



**CHAPTER 6**

**BIBLIOGRAPHY AND REFERENCES**

* <https://developer.mozilla.org/en-US/docs/>
* <https://www.w3schools.com/>
* <https://getbootstrap.com/>
* <https://ejs.co/>
* <https://careerfoundry.com/en/blog/web-development/>
* <https://www.geeksforgeeks.org/>
* https://en.wikipedia.org/wiki/JavaScript
* <https://www.educba.com/versions-of-html/>
* <https://www.udemy.com/course/the-complete-web-development-bootcamp/>
* <https://online.codingblocks.com/>