

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belagavi-560014



Web Technology and Applications

A Mini Project

On

“BlogIT Website”

**Submitted in partial fulfillment of the requirement of VII semester
WTA Laboratory**

Submitted by,

R Shreyas

1DT17IS064

R Ajay

1DT17IS063

Under the guidance of

Mrs. Anusha Preetam

Asst. Professor

Dept. of ISE

DSATM, Bangalore.



DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

(Affiliated to Visvesvaraya Technological University, Belagavi & Approved by AICTE, New Delhi)

All B.E branches Accredited 3 years by NBA, New Delhi, (Validity: 26-07-2018 to 30-06-2021)

Opp. Art of Living, Udayapura, Kanakapura Road, Bangalore- 560082

2020-2021

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

(Affiliated to Visvesvaraya Technological University, Belagavi & Approved by AICTE, New Delhi)
All B.E branches Accredited 3 years by NBA, New Delhi, (Validity: 26-07-2018 to 30-06-2021)
Opp. Art of Living, Udayapura, Kanakapura Road, Bangalore- 560082

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING



Certificate

This is to certify that the mini-project work entitled **“BlogIT Website”** is carried out by **R SHREYAS(1DT17IS064) and R Ajay(1DT17IS063)**) in partial fulfillment for the requirement of VII semester Web Technology and Applications Laboratory in **Information Science and Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year 2020-2021. It is certified that all the corrections/ suggestions indicated for the given internal assessment have been incorporated in the report. This report has been approved as it satisfies the academic requirements with respect to the mini-project work.

Signature of the Guide

Mrs. Anusha Preetam

Asst. Professor, Dept. of ISE
DSATM, Bangalore.

Signature of the HOD

Dr. Sumithra Devi K A

Prof. & Head, Dept. of ISE
DSATM, Bangalore.

External Viva

Name of the Examiners

1.

2.

Signature with date

ACKNOWLEDGEMENT

We would like to thank the **VTU, Belagavi**, for having this mini project work as part of its curriculum, which gave us a wonderful opportunity to work on our research and presentation abilities.

We extend our deep sense of sincere gratitude to **Dr. B R Lakshmikantha, Principal, DSATM, Bengaluru**, for providing us an opportunity to finish the necessary work.

We are thankful to all the people who have directly and indirectly involved in this project work. I/We am/are grateful to **Dr. Sumithra Devi K A, Dean Academics and Head of Department, Information Science and Engineering, DSATM, Bangalore**, for her valuable suggestions and advice throughout my/our work period.

I would like to express my deepest gratitude and sincere thanks to our guides **Mrs. Anusha Preetam, Assistant Professor, Department of Information Science and Engineering, DSATM, Bangalore**, for her keen interest and encouragement in the project whose guidance made the project into reality.

I/We would like to thank all the staff members of **Department of Information Science and Engineering** for their support and encouragement during the course of this project.

Definitely most, I/We would like to thank my/our parents, all my family members and friends, without whose help and encouragement this project would have been impossible.

R SHREYAS 1DT17IS064

ABSTRACT

A blog is a discussion or informational website published on the World Wide Web consisting of discrete, often informal diary-style text entries. Despite its grassroots beginnings, blogging has also become a popular platform for business, from companies trying to humanize their brand to solopreneurs seeking to make a full-time income online. But with increased opportunity comes increased competition, and it takes more to stand out now than in the early days of blogging. Still, there are so many more people online today, so the potential rewards are higher for bloggers who break through.

Blogging in industry is used connect and build relationships with leads and customers. A blog is a great place to provide content that adds value for readers and helps them better understand how to solve their greatest challenges. Also through a blog, one can develop a community of like-minded users. Blogging is also a great way to help set a business apart from competitors that offer similar products or services. In the blog content, one is able to show a little personality while also demonstrating their experience in the industry. This can be the deciding factor for a consumer who is on the fence about whether to buy from their company or one of their competitors.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ACKNOWLEDGEMENT	i
	ABSTRACT	ii
	TABLE OF CONTENTS	iii
1	INTRODUCTION	1
2	REQUIRMENT ANALYSIS	2
3	DESIGN	3
4	IMPLEMENTATION	4
5	TESTING	22
6	SNAPSHOTS	24
7	CONCLUSION AND FUTURE ENHANCEMENT	30
8	REFERENCE	31

CHAPTER 1

INTRODUCTION

The BlogIt is a blog website where users can post customized content that they want to share with the Internet. This website is the combination of technology (hardware and software) and processes and procedures that helps users to express their thoughts and stories in a neatly formatted and styled form that would pique the interest of the readers. This website, allows the users to write blog content with the help of an interactive and custom on-site text editor, allows them to insert images from their PCs/ mobile devices and even allows them to embed video and website reference links.

A blog is a discussion or informational website published on the World Wide Web consisting of discrete, often informal diary-style text entries. Despite its grassroots beginnings, blogging has also become a popular platform for business, from companies trying to humanize their brand to solopreneurs seeking to make a full-time income online. But with increased opportunity comes increased competition, and it takes more to stand out now than in the early days of blogging. Still, there are so many more people online today, so the potential rewards are higher for bloggers who break through.

Blogging in industry is used connect and build relationships with leads and customers. A blog is a great place to provide content that adds value for readers and helps them better understand how to solve their greatest challenges. Also through a blog, one can develop a community of like-minded users. Blogging is also a great way to help set a business apart from competitors that offer similar products or services. In the blog content, one is able to show a little personality while also demonstrating their experience in the industry. This can be the deciding factor for a consumer who is on the fence about whether to buy from their company or one of their competitors.

CHAPTER 2

REQUIREMENTS ANALYSIS

The requirement analysis specifies the requirements needed to develop a graphic project. In this phase, we collect the requirements needed for designing the project. The requirements collected are then analyzed and carried to the next phase.

2.1 SOFTWARE REQUIREMENTS:

1. OPERATING SYSTEM : Windows XP or Higher version
2. FRONT END : HTML, CSS, JavaScript, JQuery, & Bootstrap
3. BACK END : MongoDB, NodeJs, JavaScript
4. DEV TOOLS : Goorm Cloud IDE

2.2 HARDWARE REQUIREMENTS

1. Processor – Pentium IV or above
2. RAM – 2 GB or more
3. Hard disk – 3 GB or more

CHAPTER 3

DESIGN

3.1 USECASE DIAGRAM

The boundary, which defines the system of interest in relation to the world around it. The actors, usually individuals involved with the system defined according to their roles. The use cases, which are the specific roles played by the actors within and around the system.

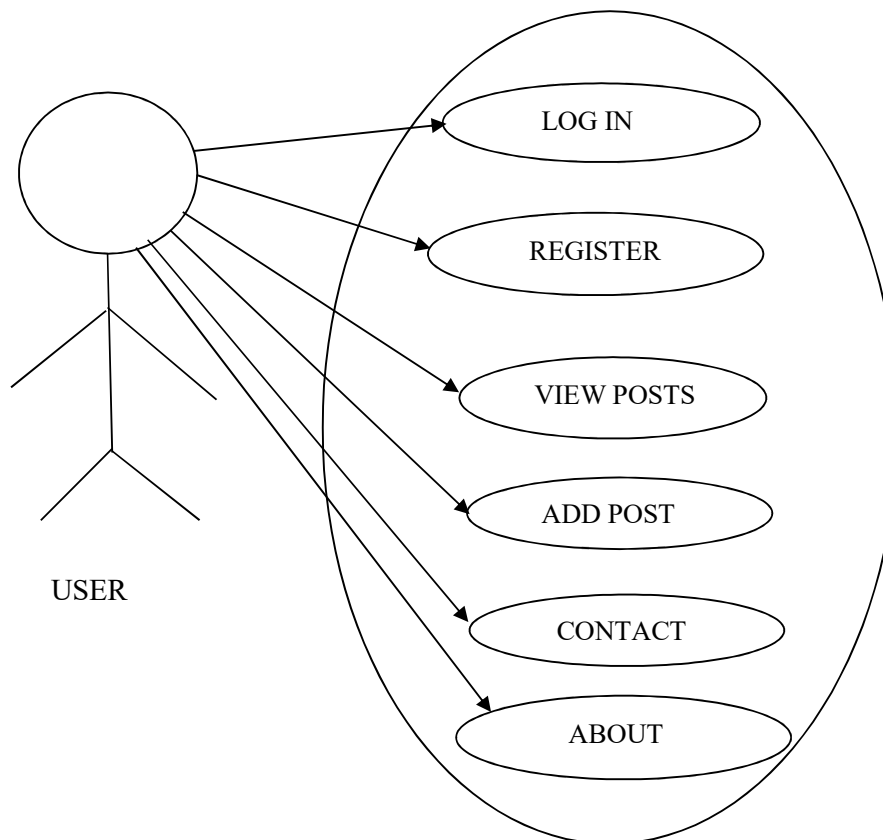


Figure 3.2: Use case diagram of BlogIt website

Use case diagram BlogIt website: The main purpose of a use case diagram is to show who interacts with your system, and the main goals they achieve with it.

CHAPTER 4

IMPLEMENTATION

4.1 INTRODUCTION TO FRONT END TOOL

4.1.1 HTML, CSS JAVASCRIPT, & BOOTSTRAP

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

HTML is a markup language which is used by the browser to manipulate text, images and other content to display it in required format. HTML was created by Tim Berners-Lee in 1991. The first ever version of HTML was HTML 1.0 but the first standard version was HTML 2.0 which was published in 1999.

CSS: Cascading Style Sheets, fondly referred to as **CSS**, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of HTML that makes up each web page. CSS is easy to learn and understood but it provides powerful control over the presentation of an HTML document.

JAVASCRIPT: JavaScript, often abbreviated as JS, is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm. Alongside HTML and CSS, JavaScript is one of the three core technologies of the World Wide Web. JavaScript enables interactive web pages and thus is an essential part of web applications.

The vast majority of websites use it, and all major web browsers have a dedicated JavaScript engine to execute it. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has an API for working with text, arrays, dates, regular expressions, and basic manipulation of the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

BOOTSTRAP: Bootstrap is a free and open-source front end development framework for the creation of websites and web apps. The Bootstrap framework is built on HTML, CSS, and JavaScript (JS) to facilitate the development of responsive, mobile-first sites and apps. Responsive design makes it possible for a web page or app to detect the visitor's screen size and orientation and automatically adapt the display accordingly; the mobile first approach assumes that smartphones, tablets and task-specific mobile apps are employees' primary tools for getting work done and addresses the requirements of those technologies in design.

4.2 INTRODUCTION TO BACK END TOOL

4.2.1 NODE.JS & MONGO DB

NODE.JS: Node.js is an open-source, cross-platform, back-end, JavaScript runtime environment that 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. 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.

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. Node.js can be combined with a browser, a database that supports JSON data (such as Postgres, MongoDB, or CouchDB) and JSON for a unified JavaScript development stack.

MONGO DB: MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License (SSPL). MongoDB supports field, range query, and regular-expression searches. Queries can return specific fields of documents and also include user-defined JavaScript functions. Queries can also be configured to return a random sample of results of a given size. Fields in a MongoDB document can be indexed with primary and secondary indices.

MongoDB can be used as a file system, called GridFS, with load balancing and data replication features over multiple machines for storing files. This function, called grid file system, is included with MongoDB drivers. MongoDB exposes functions for file manipulation and content to developers. GridFS can be accessed using mongofiles utility or plugins for Nginx and lighttpd. GridFS divides a file into parts, or chunks, and stores each of those chunks as a separate document.

Document:

A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

_id is a 12 bytes hexadecimal number which assures the uniqueness of every document. You can provide _id while inserting the document. If you don't provide then MongoDB provides a unique id for every document. These 12 bytes first 4 bytes for the current timestamp, next 3 bytes for machine id, next 2 bytes for process id of MongoDB server and remaining 3 bytes are simple incremental VALUE.

Advantages of MongoDB over RDBMS

- **Schema less** – MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- Structure of a single object is clear.

- No complex joins.
- Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
- Tuning.
- **Ease of scale-out** – MongoDB is easy to scale.
- Conversion/mapping of application objects to database objects not needed.
- Uses internal memory for storing the (windowed) working set, enabling faster access of data.

CONNECTIVITY OF THE DATABASE

Connecting to MongoDB database using Node.JS

1. Using Mongoose: We can use mongoose.connect() to connect to the required by mentioning the URL for the database.

Syntax:

```
const mongoose = require('mongoose');

function connect (config) {

  mongoose.Promise = require('bluebird');
  mongoose.set('useNewUrlParser', true);
  mongoose.set('useFindAndModify', false);
  mongoose.set('useCreateIndex', true);
  const dbURL = process.env.DATABASEURL || config.db.url;
  mongoose.connect("mongodb://localhost:27017/student", function(err) {
    if (err) {
      console.log(err.message);
    }
  });

  console.log('Database is connected on: ', dbURL);
}

module.exports = {
  connect: connect
};
```

Explanation: Mongoose connects to the database mentioned in the mongoose.connect() parameter if it already exists, or else a new database is created dynamically and connected to it.

Defining the User and Blogpost schemas for Mongoose:

Blogpost Schema:

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
const BlogPostSchema = new Schema({
  title: String,
  body: String,
  userid: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true,
  },
  datePosted: {
    type: Date,
    default: new Date(),
  },
  image: String,
})
const BlogPost = mongoose.model('BlogPost', BlogPostSchema);
module.exports = BlogPost;
```

User Schema:

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
var uniqueValidator = require('mongoose-unique-validator');
const bcrypt = require('bcrypt');
const UserSchema = new Schema({
  username: {
    type: String,
    required: [true, 'Please provide a username.'],
    unique: true,
  }, password: {
    type: String,
    required: [true, 'Please provide a password.'],
  },
});
UserSchema.plugin(uniqueValidator, {
  message: 'The {PATH} already exists.',
});
UserSchema.pre('save', function (next) {
  const user = this;
  bcrypt.hash(user.password, 10, (error, hash) => {
    user.password = hash; next();
  });
});
const User = mongoose.model('User', UserSchema);
module.exports = User;
```

4.4 MODULES

1. User Registration

```
<!DOCTYPE html>
<html lang="en">
  <%- include('layouts/header'); -%>

  <body>
    <%- include('layouts/navbar'); -%>

    <!-- Page Header -->
    <header
      class="masthead"
      style="background-image: url('/img/blog-post-bg.jpg');"
    >
      <div class="overlay"></div>
      <div class="container">
        <div class="row">
          <div class="col-lg-8 col-md-10 mx-auto">
            <div class="page-heading">
              <h1>Register a new account</h1>
            </div>
          </div>
        </div>
      </div>
    </header>

    <!-- Main Content -->
    <div class="container">
      <div class="row">
        <div class="col-lg-8 col-md-10 mx-auto">
          <% if(errors != null && errors.length > 0){ %>
            <ul class="list-group">
              <% for (var i = 0; i < errors.length; i++) { %>
                <li class="list-group-item list-group-item-danger">
                  <%= errors[i] %>
                </li>
              <% } %>
            </ul>
          <% } %>
          <form
            action="/users/register"
            method="POST"
            enctype="multipart/form-data"
```

```

controls"
    >
        <div class="control-group">
            <div
                class="form-group          floating-label-form-group
            >
                <label>User Name</label>
                <input
                    type="text"
                    class="form-control"
                    placeholder="User Name"
                    id="username"
                    name="username"
                    value="<%=username%>"
                />
            </div>
        </div>
        <div class="control-group">
            <div
                class="form-group          floating-label-form-group
            >
                <label>Password</label>
                <input
                    type="password"
                    class="form-control"
                    placeholder="Password"
                    id="password"
                    name="password"
                    value="<%=password%>"
                />
            </div>
        </div>
        <br />
        <div class="form-group">
            <button
                type="submit"
                class="btn btn-primary"
                id="sendMessageButton">
                Register
            </button>
        </div>
    </form>
</div>
</div>

```

```

    </div>

    <hr />
    <%- include('layouts/footer'); -%> <%- include('layouts/scripts'); -%>
  </body>
</html>

```

```

const User = require('../models/User');
const path = require('path');

module.exports = (req, res) => {
  User.create(req.body, (error, user) => {
    if (error) {
      const validationErrors = Object.keys(error.errors).map(
        (key) => error.errors[key].message
      );
      req.flash('validationErrors', validationErrors);
      req.flash('data', req.body);
      // req.session.validationErrors = validationErrors;
      return res.redirect('/auth/register');
    }
    res.redirect('/');
  });
};

```

2.User Login

```

<!DOCTYPE html>
<html lang="en">
  <%- include('layouts/header'); -%>

  <body>
    <%- include('layouts/navbar'); -%>

    <!-- Page Header -->
    <header
      class="masthead"
      style="background-image: url('/img/blog-post-bg.jpg');"
    >
      <div class="overlay"></div>
      <div class="container">
        <div class="row">
          <div class="col-lg-8 col-md-10 mx-auto">
            <div class="page-heading">
              <h1>Login</h1>

```



```

        </div>
    </div>
</div>
</div>
</header>

<!-- Main Content -->
<div class="container">
    <div class="row">
        <div class="col-lg-8 col-md-10 mx-auto">
            <!-- for browser to know tha tthe form contains mutli media. will
encripy multie media and send to server -->
            <form
                action="/users/login"
                method="POST"
                enctype="multipart/form-data"
            >
                <div class="control-group">
                    <div
                        class="form-group          floating-label-form-group
controls"
                    >
                        <label>User Name</label>
                        <input
                            type="text"
                            class="form-control"
                            placeholder="User Name"
                            id="username"
                            name="username"
                        />
                    </div>
                </div>
                <div class="control-group">
                    <div
                        class="form-group          floating-label-form-group
controls"
                    >
                        <label>Password</label>
                        <input
                            type="password"
                            class="form-control"
                            placeholder="Password"
                            id="password"
                            name="password"
                        />
                    </div>
                </div>
            </form>
        </div>
    </div>
</div>

```

```

        </div>
    </div>
    <br />
    <div class="form-group">
        <button
            type="submit"
            class="btn btn-primary"
            id="sendMessageButton"
        >
            Login
        </button>
    </div>
</form>
</div>
</div>
</div>

<hr />
<%- include('layouts/footer'); -%> <%- include('layouts/scripts'); -%>
</body>
</html>

```

```

module.exports = (req, res) => {
    const { username, password } = req.body;

    User.findOne({ username: username }, (error, user) => {
        if (user) {
            bcrypt.compare(password, user.password, (error, same) => {
                if (same) {
                    // if password match
                    // store user session
                    req.session.userId = user._id;
                    res.redirect('/');
                } else {
                    res.redirect('/auth/login');
                }
            });
        } else {
            res.redirect('/auth/login');
        }
    });
};

```

3.Home page

```
<html lang="en">
```

```

<%- include('layouts/header') %>

<body>
  <%- include('layouts/navbar') %>

  <!-- Page Header -->
  <header
    class="masthead"
    style="background-image: url('img/home-bg.png');"
  >
    <div class="overlay"></div>
    <div class="container">
      <div class="row">
        <div class="col-lg-10 col-md-12 mx-auto">
          <div class="site-heading">
            <h1>Welcome to BlogIt</h1>
            <span class="subheading">Blogging is a conversation
❏❏❏</span>
          </div>
        </div>
      </div>
    </div>
  </header>

  <!-- Main Content -->
  <div class="container">
    <div class="row">
      <div class="col-lg-8 col-md-10 mx-auto flex-wrap">
        <% for (var i = 0; i < blogposts.length; i++) { %>
          <div class="post-preview">
            <a href="/post/<%= blogposts[i]._id %>">
              <h2 class="post-title">
                <%= blogposts[i].title %>
              </h2>
              <div class="thumbnail">
                
              </div>
            </a>
            <p class="post-meta">
              Posted by
              <a href="#"><%= blogposts[i].userid.username %></a>
              on <%= blogposts[i].datePosted.toString() %>
            </p>
          </div>
        </div>
      </div>
    </div>
  </div>

```

```
<hr style="width:800px" />
<% } %>

<!-- Pager -->
<div class="clearfix">
  <a class="btn btn-primary float-right" href="#"
    >Older Posts &rarr;</a
  >
</div>
</div>
</div>
</div>

<hr />

<%- include('layouts/footer') %> <%- include('layouts/scripts') %>
</body>
</html>
```

Manipulation Details

1.Creating a blog post:-

```
<!DOCTYPE html>
<html lang="en">
  <%- include('layouts/header'); -%>

  <body>
    <%- include('layouts/navbar'); -%>

    <!-- Page Header -->
    <header
      class="masthead"
      style="background-image: url('/img/blog-post-bg.jpg');"
    >
      <div class="overlay"></div>
      <div class="container">
        <div class="row">
          <div class="col-lg-8 col-md-10 mx-auto">
            <div class="page-heading">
              <h1>New Post</h1>
            </div>
          </div>
        </div>
      </div>
    </header>
```

```

<!-- Main Content -->
<div class="container">
  <div class="row">
    <div class="col-lg-8 col-md-10 mx-auto">
      <!-- for browser to know tha tthe form contains mutli media. will
encripy multie media and send to server -->
      <form
        action="/posts/store"
        method="POST"
        enctype="multipart/form-data">
        <div class="control-group">
          <div
            class="form-group          floating-label-form-group
controls"

            <label>Title</label>
            <input
              type="text"
              class="form-control"
              placeholder="Title"
              id="title"
              name="title"
              required/>
          </div>
        </div>
        <div class="control-group">
          <div
            class="form-group          floating-label-form-group
controls">
            <label>Description</label>
            <textarea
              id="body"
              name="body"
              class="form-control"
              required
            ></textarea>
            <script>
              $('#body').summernote({
                placeholder: 'Hello there, blogger!',
                tabsize: 2,
                height: 200,
              });
            </script>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>

```

```

        <div class="control-group">
          <div
            class="form-group          floating-label-form-group
controls">
            <label>Image (use dark themes for better
experience)</label>
            <input
              type="file"
              class="form-control"
              id="image"
              name="image"
              required/>
            </div>
          </div>
          <br />
          <div class="form-group">
            <button
              type="submit"
              class="btn btn-primary"
              id="sendMessageButton">
              Send
            </button>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>

  <hr />
  <%- include('layouts/footer'); -%> <%- include('layouts/scripts'); -%>
</body>
</html>

```

```

module.exports = (req, res) => {
  if (req.session.userId) {
    return res.render('create', {
      createPost: true,
    });
  }
  res.redirect('/auth/login');
};

const BlogPost = require('../models/BlogPost');
const path = require('path');

module.exports = (req, res) => {

```

```
let image = req.files.image;
image.mv(
  path.resolve(__dirname, '..', 'public/img', image.name),
  async (error) => {
    await BlogPost.create({
      ...req.body,
      image: '/img/' + image.name,
      userid: req.session.userId,
    });
    console.log('req.body', req.body);
    res.redirect('/');
  }
);
};
```

3.Contact:-

```
<!DOCTYPE html>
<html lang="en">
  <%- include('layouts/header') %>
  <body>
    <%- include('layouts/navbar') %>
    <!-- Page Header -->
    <header
      class="masthead"
      style="background-image: url('img/post-bg.jpg');">
      <div class="overlay"></div>
      <div class="container">
        <div class="row">
          <div class="col-lg-8 col-md-10 mx-auto">
            <div class="page-heading">
              <h1>Contact Us</h1>
              <span class="subheading">
                Have questions? we have answers.</span>
            </div>
          </div>
        </div>
      </div>
    </header>
    <div class="container">
      <div class="row">
        <div class="col-lg-8 col-md-10 mx-auto">
          <p>
            Want to get in touch? Fill out the form below to send
            me a message and we will get back to you as soon as possible!
          </p>
```

```

<form name="sendMessage" id="contactForm" novalidate>
  <div class="control-group">
    <div
      class="form-group      floating-label-form-group
controls">
      <label>Name</label>
      <input
        type="text"
        class="form-control"
        placeholder="Name"
        id="name"
        required
        data-validation-required-message="Please
enter your name."/>
      <p class="help-block text-danger"></p>
    </div>
  </div>
  <div class="control-group">
    <div
      class="form-group      floating-label-form-group
controls">
      <label>Email Address</label>
      <input
        type="email"
        class="form-control"
        placeholder="Email Address"
        id="email"
        required
        data-validation-required-message="Please
enter your email address."/>
      <p class="help-block text-danger"></p>
    </div>
  </div>
  <div class="control-group">
    <div
      class="form-group col-xs-12 floating-label-form-
group controls">
      <label>Phone Number</label>
      <input
        type="tel"
        class="form-control"
        placeholder="Phone Number"
        id="phone"
        required

```



```

                                data-validation-required-message="Please
enter your phone number."
                                />
                                <p class="help-block text-danger"></p>
                                </div>
                                </div>
                                <div class="control-group">
                                <div
                                class="form-group          floating-label-form-group
controls" >
                                <label>Message</label>
                                <textarea
                                rows="5"
                                class="form-control"
                                placeholder="Message"
                                id="message"
                                required
                                data-validation-required-message="Please
enter a message."></textarea>
                                <p class="help-block text-danger"></p>
                                </div>
                                </div>
                                <br />
                                <div id="success"></div>
                                <button    type="submit"    class="btn    btn-primary"
id="sendMessageButton">
                                Send
                                </button>
                                </form>
                                </div>
                                </div>
                                </div>
                                <hr />
                                <%- include('layouts/footer') %> <%- include('layouts/scripts') %>
                                </body>
                                </html>

```

4.About:-

```

<!DOCTYPE html>
<html lang="en">
    <%- include('layouts/header') %>
    <body>
        <%- include('layouts/navbar') %>
        <!-- Page Header -->

```

```
<header class="masthead" style="background-image: url('img/home-
bg.png');">
  <div class="overlay"></div>
  <div class="container">
    <div class="row">
      <div class="col-lg-8 col-md-10 mx-auto">
        <div class="page-heading">
          <h1>About Us</h1>
          <span class="subheading">This is what we do.</span>
        </div>
      </div>
    </div>
  </div>
</header>
<!-- Main Content -->
<div class="container">
  <div class="row">
    <div class="col-lg-8 col-md-10 mx-auto">
      <p>
        Lorem ipsum dolor sit amet, consectetur adipisicing
        elit. Saepe nostrum ullam eveniet pariatur voluptates
        odit, fuga atque ea nobis sit soluta odio, adipisci quas
        excepturi maxime quae totam ducimus consectetur?
      </p>
      <p>
        Lorem ipsum dolor sit amet, consectetur adipisicing
        elit. Eius praesentium recusandae illo eaque architecto
        error, repellendus iusto reprehenderit, doloribus, minus
        sunt. Numquam at quae voluptatum in officia voluptas
        voluptatibus, minus!
      </p>
      <p>
        Lorem ipsum dolor sit amet, consectetur adipisicing
        elit. Aut consequuntur magnam, excepturi aliquid ex
        itaque esse est vero natus quae optio aperiam soluta
        voluptatibus corporis atque iste neque sit tempora!
      </p>
    </div>
  </div>
</div>
<hr />
<%- include('layouts/footer') %> <%- include('layouts/scripts') %>
</body>
</html>
```

CHAPTER 5

TESTING

5.1 TESTING

Testing is the process of executing a program to find the errors. A good test has the high probability of finding a yet undiscovered error. A test is vital to the success of the system. System test makes a logical assumption that if all parts of the system are correct, then goal will be successfully achieved.

5.2 TYPES OF TESTING

5.2.1 Module Testing.

5.2.2 Integration Testing.

5.2.1 Module Testing

Module testing is the testing of complete code objects as produced by the compiler when built from source.

A library may be composed of a single compiled object or several compiled objects. There is only a slight difference between unit testing and module testing. Modules are fully formed chunks of coherent source code that can typically be tested by driving a few functions signatures with various stimuli. On the other hand, unit testing (which is considered as part of the implementation phase for this software development process) may involve testing one small part of a function that will never formally implement any function interface.

As a result of modules being more self-contained, module testing will likely require less testing infrastructure such as test harness and test stubs. The testing of modules could perhaps even be automated so that they can be included in regression test suites or a acceptance test suites.

5.2.2 Integration Testing

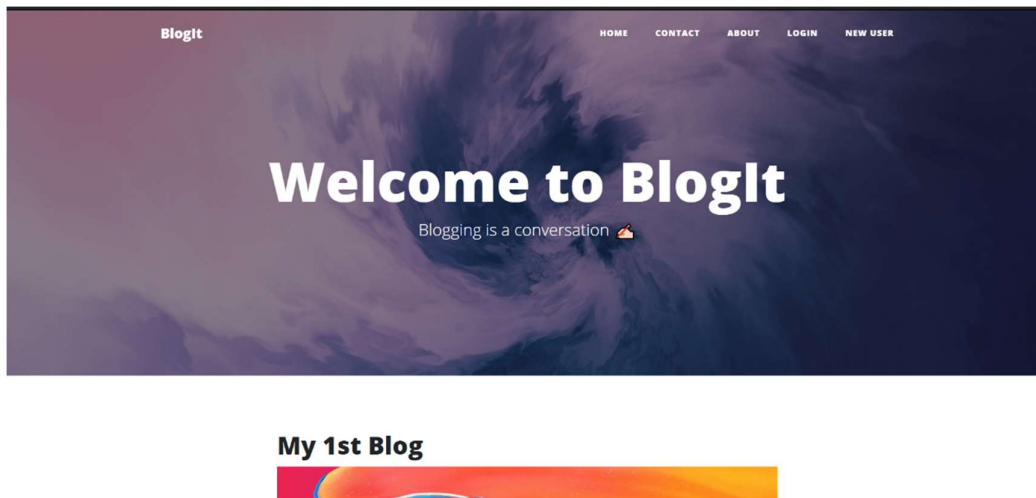
Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

Test Case Id	Description	Input Data	Expected Output	Actual Output	Status
1	Login Page	Username Password	Successfully Logged in	Successfully Logged in	Pass
2	Register Page	Username Password	Successfully Registered	Successfully Registered	Pass
3	Create Post	_id Title Body UserId Date posted Image	Entry Successful	Entry Successful	Pass
6	Users	_id Username Password	Entry Successful	Entry Successful	Pass
7	Contact	_id Name Email Phone No. Message	Entry Successful	Entry Successful	Pass

Figure 5.1: "Integration testing"

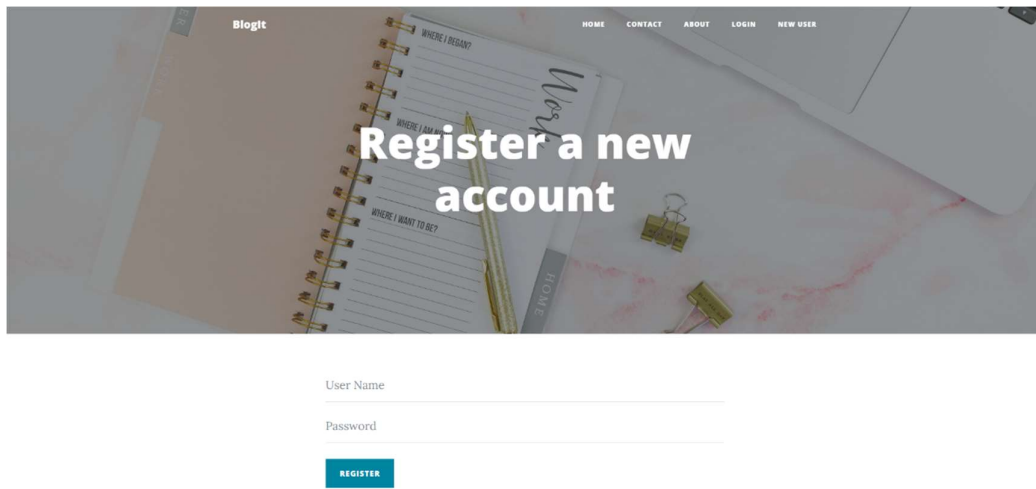
CHAPTER 6

SNAPSHOTS



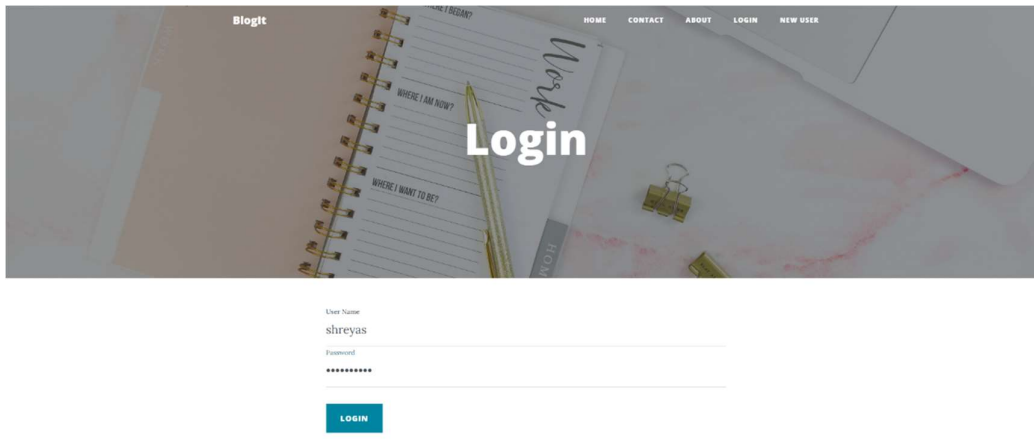
Snapshot 6.1: “Home Page”

First page of BlogIt website. This page lists all the blog posts by various users, sorted by the oldest appearing at the top.



Snapshot 6.2:”Register Page”

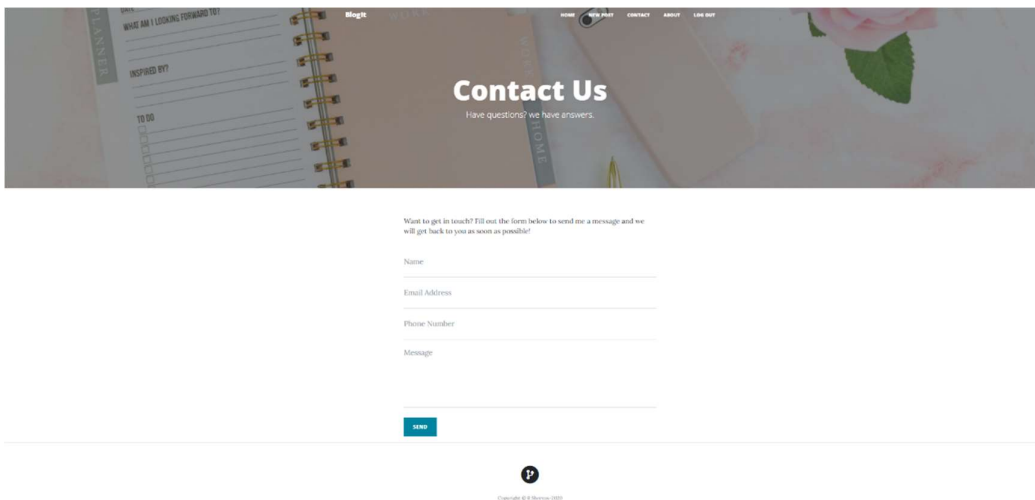
This is the register page where the user can register by providing a username and password.



The screenshot shows the login page of the BlogIt website. The header features the 'BlogIt' logo on the left and navigation links (HOME, CONTACT, ABOUT, LOGIN, NEW USER) on the right. The main content area has a background image of a desk with a spiral notebook, a pen, and paper clips. The word 'Login' is prominently displayed in the center. Below this, there are input fields for 'User Name' (containing 'shreyas') and 'Password' (masked with asterisks). A blue 'LOGIN' button is positioned at the bottom of the form.

Snapshot 6.3:”Login Page”

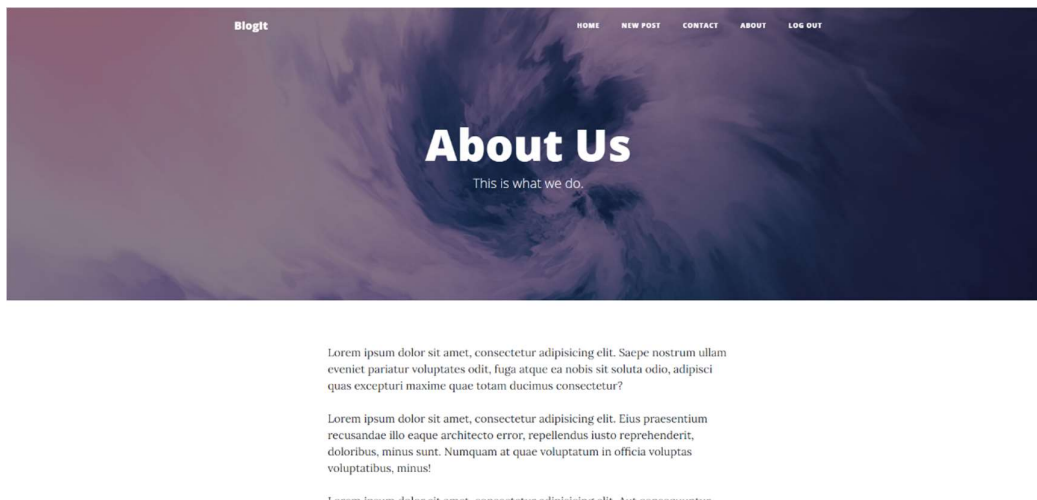
This is the login page. The user has to login in order to make their own blog posts.



The screenshot shows the 'Contact Us' page of the BlogIt website. The header is identical to the login page. The main content area features a background image of a desk with a spiral notebook, a pen, and a pink rose. The text 'Contact Us' is centered, followed by the subtitle 'Have questions? we have answers.' Below this, a message prompt reads: 'Want to get in touch? Fill out the form below to send me a message and we will get back to you as soon as possible!'. The form includes input fields for 'Name', 'Email Address', 'Phone Number', and 'Message'. A blue 'SEND' button is at the bottom of the form. At the very bottom of the page, there is a small circular logo and a copyright notice: 'Copyright © 2020 BlogIt Inc.'.

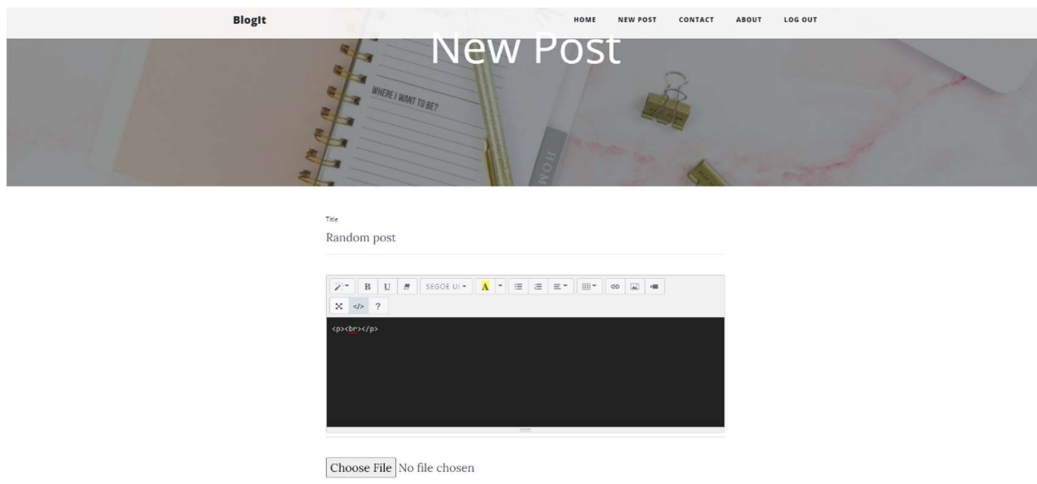
Snapshot 6.4:”Contact Page”

This is the contact page, where the users can reach out to the website support team by providing their contact details and a message that they want to convey.



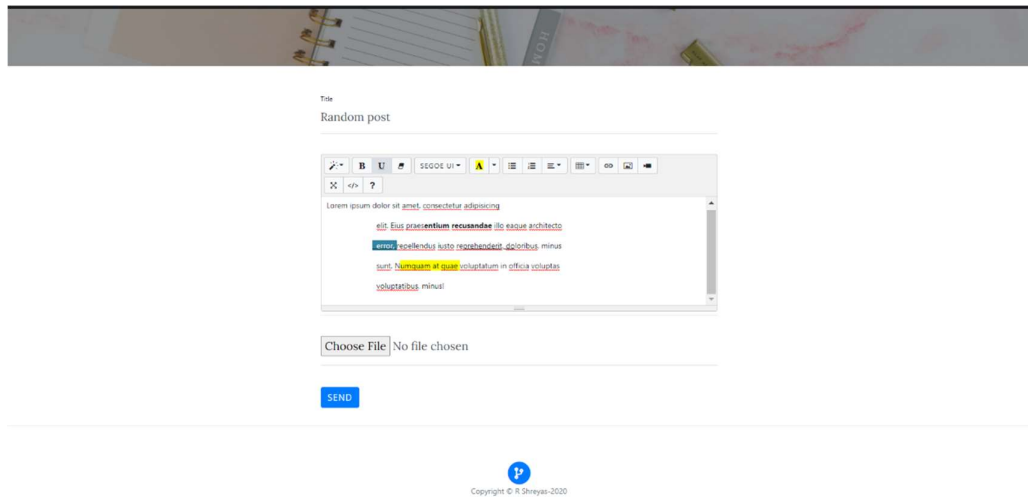
Snapshot 6.5: "About Page"

This page is used give a brief description about the website.



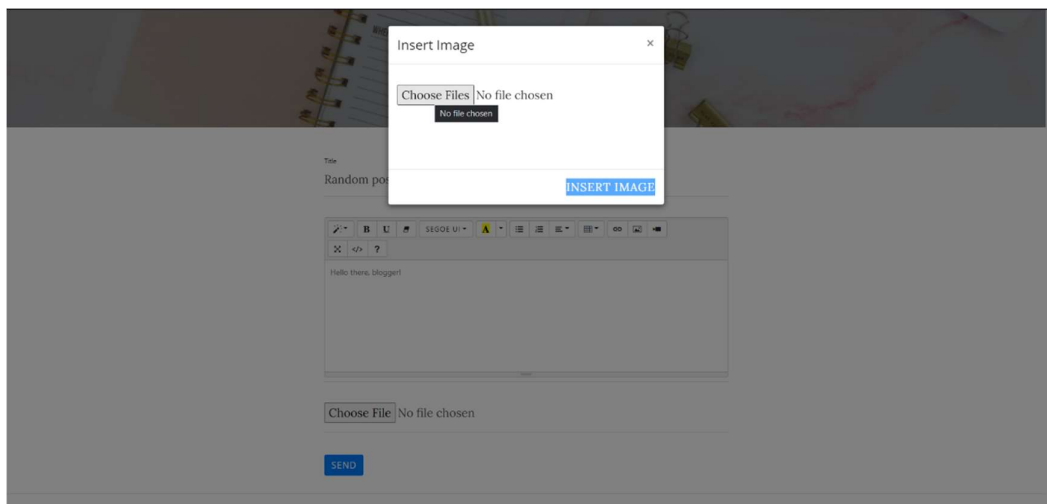
Snapshot 6.6: "New Post Page- code editor view"

This page is used write the blog by logged in users. This is the code editor view where the users can type in code snippets or even html pages.



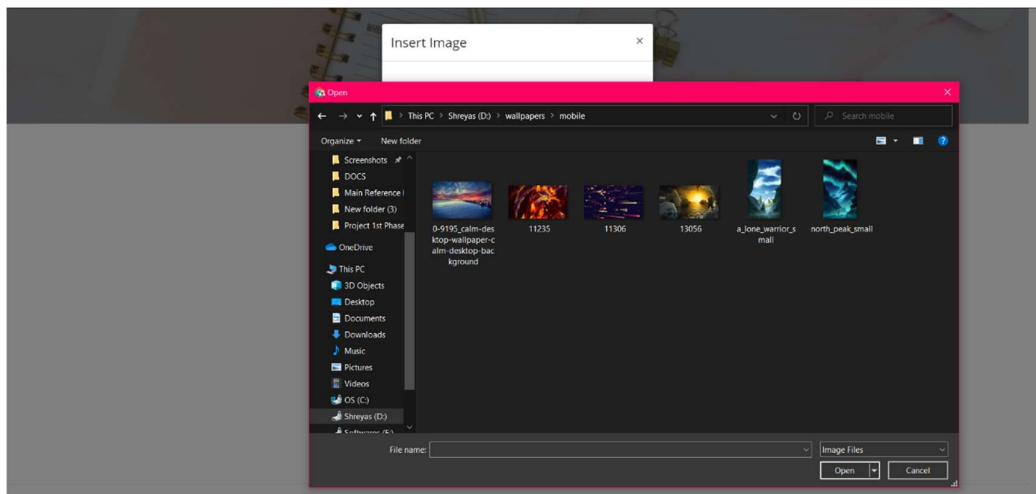
Snapshot 6.7: "New Post Page – text editor view"

This page is used write the blog by logged in users. This is the text editor view where the users can type in plain text and style it according to their wish using the given text styling options (highlight, bold, underline, etc).



Snapshot 6.8: "New Post Page- inserting images"

This is a pop up menu that allows users to choose image files form their system to be added to their blogs. Multiple images can be inserted.



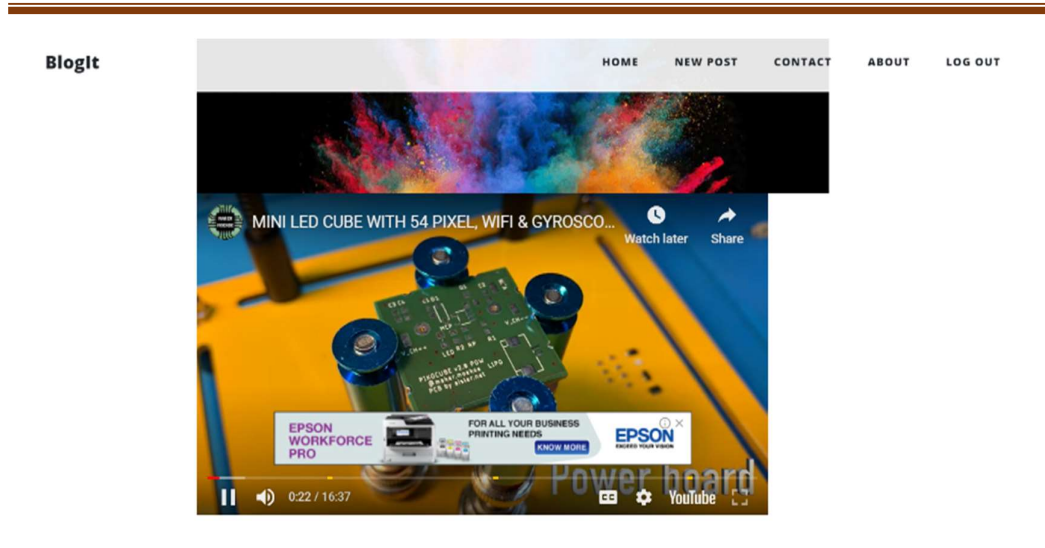
Snapshot 6.8:”Uploading image file from system”

This is the file browser pop up window that allows users to browse their system and select files for uploading.



Snapshot 6.9:”Blog post content view page”

This is the view content page that is displayed when a particular blog post is selected from the home page. It displays the complete content of the blog post.



Snapshot 6.10: "Blog post view page"

This is an example emmbedded video link, played on the view content page.

CONCLUSION AND FUTURE ENHANCEMENT

To conclude the description about the project: The project developed using Visual Studio Code, and Goorm Cloud IDE based on the requirement specification of the user and the analysis of the existing system, with a lot flexibility and scope for future enhancement.

The expanded functionality of today's software requires an appropriate approach towards software development. This website is designed for people who want to express their ideas on a definitive platform or if they want to expand their intended audience range. Blogging is also a great way to help set a business apart from competitors that offer similar products or services. In the blog content, one is able to show a little personality while also demonstrating their experience in the industry. This can be the deciding factor for a consumer who is on the fence about whether to buy from their company or one of their competitors.

There are a lot of features that can be added to this version of the website to make it even more interactive, interesting and informative. Features like a search engine to search specific blog posts or authors will greatly reduce efforts in finding the required blog post. Also, categorizing posts on the basis of authors, date, popularity, views, etc. will be helpful in reducing searching time and provide greater ease of access. Author profiling is another concept that helps users know more about the authors whose contents they like. It will also help provide an option for the viewers to be able to follow or favorite the authors of their choice in order to see their content more easily. Adding a comment section where the viewers can post their comments on a post will help the users communicate to the authors and also let the authors receive appraisals or critics from the viewing community.

REFERENCES

[1] Fundamentals of Node.JS by W3Schools

[2] Fundamentals of HTML and CSS by W3Schools

[3] Fundamentals of MongoDB by W3Schools

[4] Wikipedia:

- <https://en.wikipedia.org/wiki/Node.JS>
- <https://en.wikipedia.org/wiki/HTML>
- https://en.wikipedia.org/wiki/Cascading_Style_Sheets

[5] Google:

- <https://www.w3schools.com>
- <https://getbootstrap.com>
- <https://www.w3schools.com/js/default.asp>