

**TAKSHSHILA INSTITUTE OF ENGINEERING & TECHNOLOGY,
JABALPUR (M.P.)**



Session 2019

Department of Computer Science & Engineering

**Major Project Report
On
“Journal Blog Application”**

**Submitted For the partial fulfilment of the requirement
For the award of the degree of
Bachelor of Engineering**

By

Amit Mehra (0207CS191008),

Danish beg (0207CS191022),

Aakash Kumar (0207CS191001)

And Abhay jhariya (0207CS191002),

Under the Guidance of

Prof. Swati Soni



**RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA, BHOPAL (M.P.)
(UNIVERSITY OF TECHNOLOGY OF MADHYA PRADESH)**



**TAKSHSHILA INSTITUTE OF ENGINEERING & TECHNOLOGY,
JABALPUR (M.P.)**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Major Project report entitled, “***Double vertical slider***” is the bonafide record of the work done by **Amit Mehra, Danish beg, Aakash Kumar and Abhay jhariya** from Takshshila Institute of Engineering & Technology, Jabalpur, under the guidance of **Prof. Swati Soni** and submitted to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal MP, in accordance with the requirement for the award of the Bachelor Engineering from Computer Science & Engineering.

I certify that the above declaration is true to the best of my knowledge and belief.

**Prof. Deepak Agrawal
Department Coordinator**



RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA, BHOPAL (M.P.)
(UNIVERSITY OF TECHNOLOGY OF MADHYA PRADESH)

APPROVAL CERTIFICATE

This is to certify that project entitled, “**Journal Blog Application**”, submitted by **Amit Mehra, Danish Beg, Aakash Kumar, and Abhay jhariya** is accepted for the award of degree of Bachelor Engineering in Computer Science & Engineering Department.

INTERNAL EXAMINER

Date:

EXTERNAL EXAMINER

Date:

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I would like to express my sincere gratitude to my guide **Prof. Swati Soni** encouraging me throughout the research work and for providing me various opportunities at different stages of my studies.

I thank him especially for the help that he extended to me in the form of valuable discussions and reading materials.

I would like to thank to **Prof. Deepak Agrawal**, Department Coordinator. Takshshila Institute of engineering & Technology, Jabalpur, for his kind support and valuable suggestions wherever necessary.

I am always grateful to **Dr. B. K. Sahu**, Principal Takshshila Institute of Engineering & Technology, for permitting me to avail the facilities needed for this work.

I am grateful to all my colleagues at **Takshshila Institute of Engineering & Technology, Jabalpur** for encouraging me to complete the work.

Date.....

Students Name

Amit Mehra, Danish Beg,

Aakash Kumar, Abhay jhariya

TABLE OF CONTENTS

- 1. College certificate**
- 2. Declaration**
- 3. Acknowledgement**
- 4. Introduction**
 - **Project overview**
 - **Project module**
 - **Software development life cycle**
 - **Functional and non-functional requirement**
- 5. Technology used**
 - **Hardware specification**
 - **Software specification**
- 6. Analysis and design**
 - **Database design**
 - **Data flow diagram**
 - **E-R diagram**
 - **Class diagram**
 - **Use case diagram**
 - **Sequence diagram**
- 7. Screen layout**
- 8. Testing procedure**
- 9. Cost estimation**
- 10. Conclusion**
- 11. Future enhancement**
- 12. Time chart**
- 13. Bibliography**

ABOUT PROJECT

Journal Blog Website

This is a website enables its users to save your journal and maintain your blogs on cloud, this site is safe and secure. Through this website we have provided you a tool by which you can access all the blogs of yours from anywhere in the world.

User Module:

1. User registration
2. Login
3. Home/ feed
4. Compose post
5. Logout

HARDWARE AND SOFTWARE REQUIREMENTS

• Minimum hardware requirements

1. Intel i3 11th gen processor
2. Primary ram - 4gb
3. Secondary storage – lees than 64gb

• Software requirements

1. Visual studio code
2. MySQL server
3. Code Editor

TECH STACK USED:

HTML

Hyper Text Markup Language is the computer language that facilitates website creation. The language, which has code words and syntax just like any other language, is relatively easy to comprehend and, as time goes on, increasingly powerful in what it allows someone to create. HTML continues to evolve to meet the demands and requirements of the Internet under the guise of the World Wide Web Consortium, the organization that designs and maintains the language; for instance, with the transition to Web 2.0

CSS

CSS stands for Cascading Style Sheets. It is the language for describing the presentation of Web pages, including colours, layout, and fonts, thus making our web pages presentable to the users.

CSS is designed to make style sheets for the web.

JAVASCRIPT

JavaScript is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document. It was introduced in the year 1995 for adding programs to the webpages in the Netscape Navigator browser. Since then, it has been adopted by all other graphical web browsers. With JavaScript, users can build modern web applications to interact directly without reloading the page every time. The traditional website uses Javascript to provide several forms of interactivity and simplicity.

EJS

EJS is a simple templating language that lets you generate HTML markup with plain JavaScript. No religiousness about how to organize things. No reinvention of iteration and control-flow. It is just plain JavaScript.

NodeJS

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in

JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

EXPRESSJS

Express.js is a small framework that works on top of Node.js web server functionality to simplify its APIs and add helpful new features. It makes it easier to organize your application's functionality with middleware and routing. It adds helpful utilities to Node.js HTTP objects and facilitates the rendering of dynamic HTTP objects.

MySQL

MySQL is a relational database management system (RDBMS) developed by oracle that is based on structured query language (SQL).

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or a place to hold the vast amounts of information in a corporate network. A relational database is a digital store collecting data and organizing it according to the relational model. In this model, tables consist of rows and columns, and relationships between data elements all follow a strict logical structure. An RDBMS is simply the set of software tools used to implement, manage, and query such a database.

DEVELOPMENT LIFE CYCLE

The **Systems Development Life Cycle (SDLC)**, or Software Development Life Cycle in systems engineering and software engineering, is the process of creating or altering systems, and the models and methodologies that people use to develop these systems. The concept generally refers to computer or information systems. In software engineering the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system the software development process. Systems Development Life Cycle (SDLC) is a logical process used by a systems analyst to develop an information system, including requirements, validation, training, and user (stakeholder) ownership. Any SDLC should result in a high-quality system that meets or exceeds customer expectations, reaches completion within time and cost estimates, works effectively and efficiently in the current and planned Information

Technology infrastructure, and is inexpensive to maintain and cost-effective to enhance.

Computer systems are complex and often (especially with the recent rise of Service Oriented Architecture) link multiple traditional systems potentially supplied by different software vendors. To manage this level of complexity, several SDLC models have been created: "waterfall"; "fountain"; "spiral"; "build and fix"; "rapid prototyping"; "incremental"; and "synchronize and stabilize". SDLC models can be described along a spectrum of agile to iterative to sequential. Agile methodologies, such as XP and Scrum, focus on light-weight processes which allow for rapid changes along the development cycle. Iterative methodologies, such as Rational Unified Process and Dynamic Systems Development Method, focus on limited project scopes and expanding or improving products by multiple iterations. Sequential or big-design-upfront (BDUF) models, such as Waterfall, focus on complete and correct planning to guide large projects and risks to successful and predictable results. Some agile and iterative proponents confuse the term SDLC with sequential or "more traditional" processes; however, SDLC is an umbrella term for all methodologies for the design, implementation, and release of software.

1. Initiation/planning

2. Requirements gathering and analysis

3. Design

4. Build or coding

5. Testing

- **Unit testing:** unit testing is a software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is the smallest testable part of an application
- **Integration testing:** The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware.
- **Black box testing:** Black-box testing uses external descriptions of the software, including specifications, requirements, and designs to derive test cases.
- **White box testing:** White box testing (a.k.a. clear box testing, glass box testing, and transparent box testing or structural testing) uses an internal

perspective of the system to design test cases based on internal structure.

- **Regression testing:** It is any type of software testing that seeks to uncover software errors by partially retesting a modified program.

- Automation testing

- User acceptance testing & Performance testing

- **System testing:** System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

6. operations and maintenance:

DATABASE DESIGN:

TABLES:

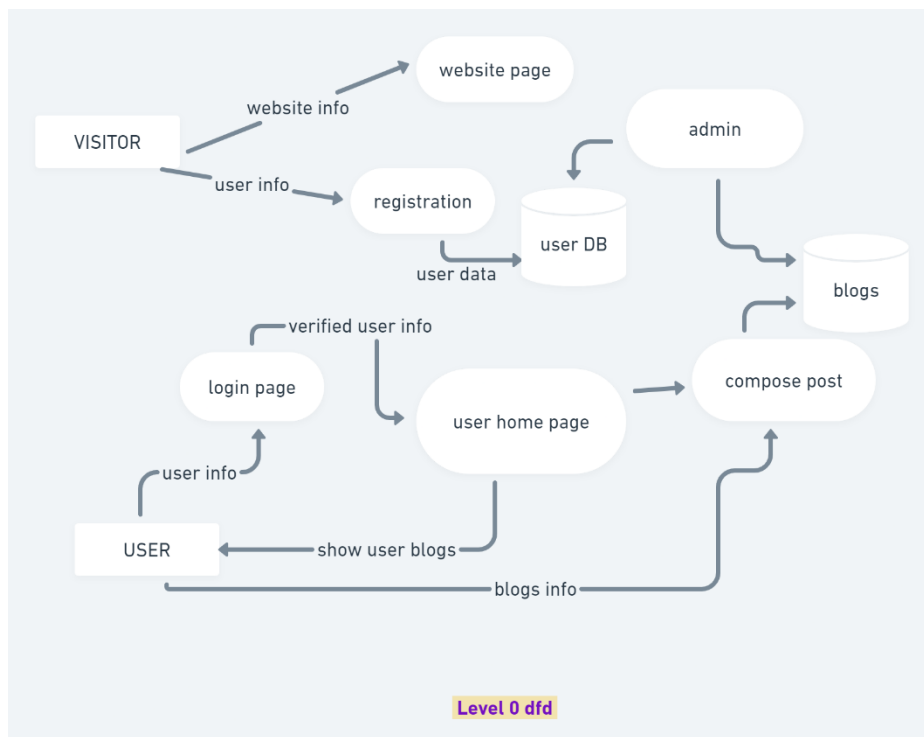
Users table

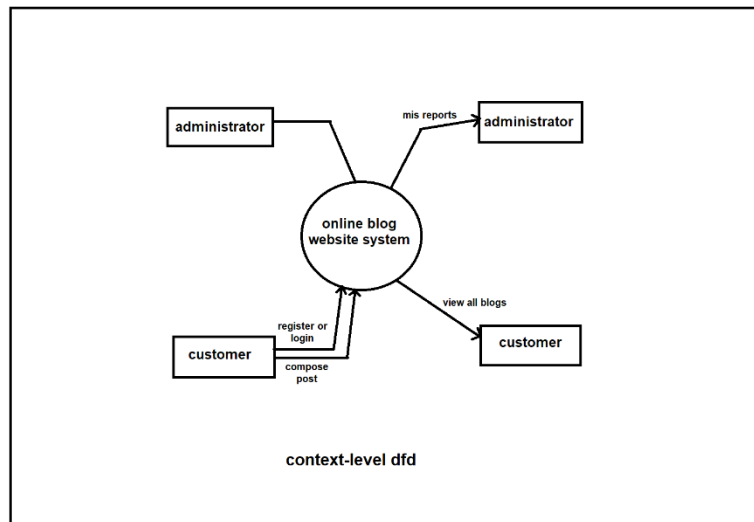
Column	Type	Default Value	Nullable
◇ name	varchar(255)		NO
◇ email	varchar(255)		NO
◇ passwordl	varchar(255)		NO

Blog table

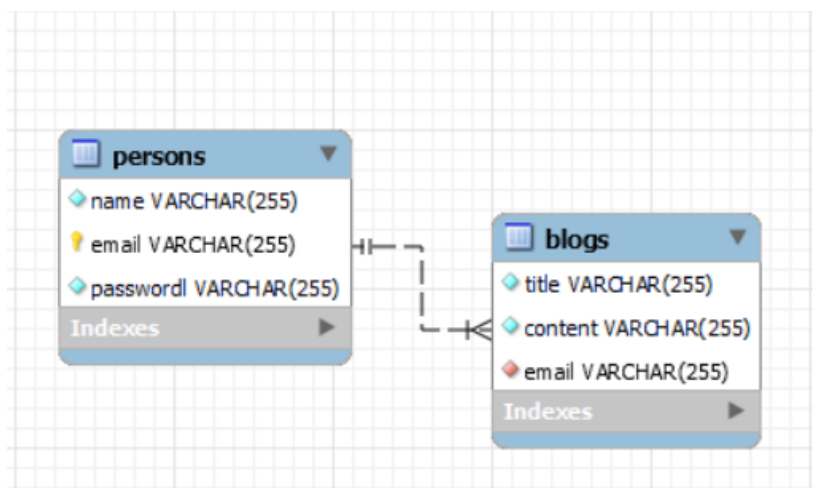
Column	Type	Default Value	Nullable
◇ title	varchar(255)		NO
◇ content	varchar(255)		NO
◇ email	varchar(255)		NO

DFD DIAGRAMS:

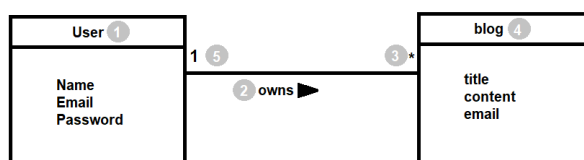




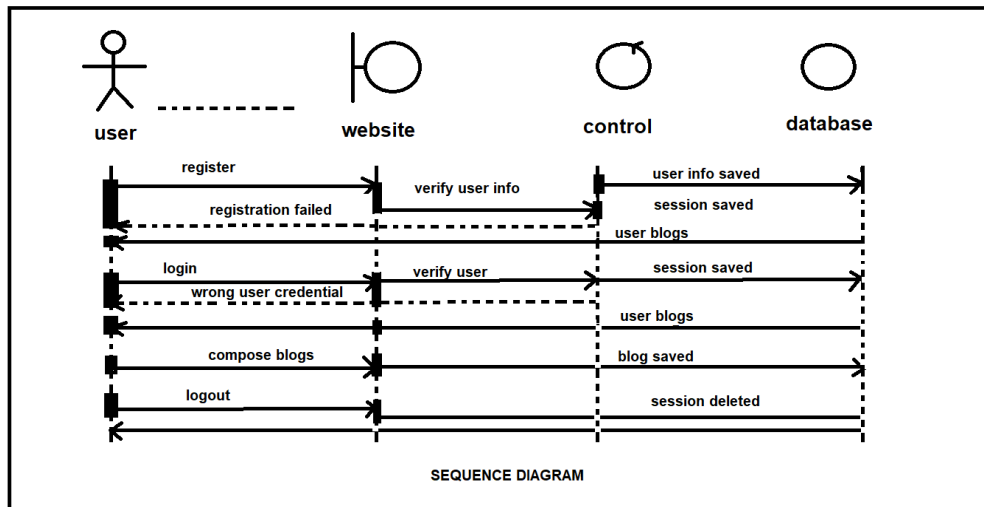
EER DIAGRAM:



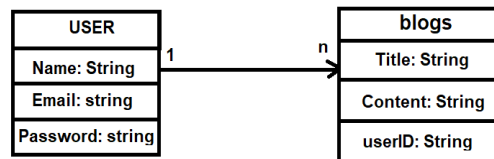
Class diagram:



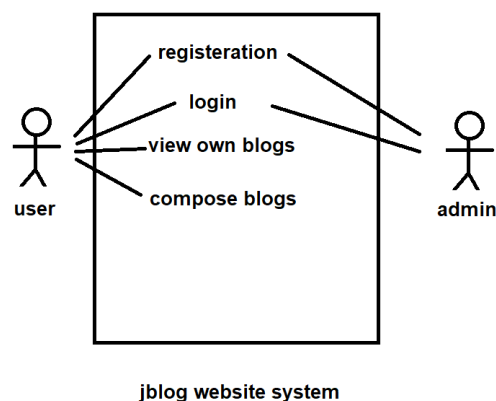
Sequence diagram:



UML diagram:



Use case diagram:



SCREEN LAYOUT



[Home](#) [Features](#) [Contact Us](#)

welcome everyone

Now you can store all your journals securely.

[Get Started](#)



Features

EASY TO USE

This site is very easy to use, And beginner friendly.

SAFE AND SECURE

This site store all your data securely and safely.

FREE OF COST

This site charges nothing but an honest feedback.

Registration page:



Already have an account? [LOG IN](#)

Sign Up


Name

Email

Password

[Sign Up](#)

LOGIN PAGE:

 Don't have an account? [SIGN UP](#)


Log In

Email

Password

Log In

DASHBOARD PAGE.

 [Home](#) [Compose Post](#) [Logout](#)

Home

Lacus vel facilisis volutpat est velit egestas dui id ornare. Semper auctor neque vitae tempus quam. Sit amet cursus sit amet dictum sit amet justo. Viverra tellus in hac habitasse. Imperdiet proin fermentum leo vel orci porta. Donec ultrices tincidunt arcu non sodales neque sodales ut. Mattis molestie a iaculis at erat pellentesque adipiscing. Magnis dis parturient montes nascetur ridiculus mus mauris vitae ultrices. Adipiscing elit ut aliquam purus sit amet luctus venenatis lectus. Ultrices vitae auctor eu augue ut lectus arcu bibendum at. Odio euismod lacinia at quis risus sed vulputate odio ut. Cursus mattis molestie a iaculis at erat pellentesque adipiscing

day1

this was a very nice day

Day 2

this is very nice day too , had some fun with friends.

Day 3


this day I have completed my major project.

Day 4


I am going to become world best developer.

Day 5

this is my first major project

Made with  by Amit mehra

COMPOSE POST PAGE:

 JBLog

Home logout

Compose Post

Title

Post

Publish

Made with ❤️ by Amit mehra

CODING

App.js

```
// packages
require('dotenv').config()
const express = require("express");
const bodyParser = require("body-parser");
const mysql = require('mysql2');
const bcrypt = require('bcrypt');
const session = require('express-session');

// use of express
const app = express();
app.set('view engine', 'ejs');

app.use(bodyParser.urlencoded({ extended: true }));
app.use(express.static("views"));
const saltRounds = 10;

/** session middleware */
app.use(session({
  secret: process.env.SECRET,
  resave: false,
  saveUninitialized: true,
  cookie: {
    secure: false, maxAge: 60*60000
  }
}));

/** ends here */
const connection = mysql.createConnection({
  host: process.env.HOST,
  port: process.env.PORT,
  database: process.env.DATABASE,
  user: process.env.USER,
  password: process.env.PASSWORD
});

/** db connection */
connection.connect(function (err) {
  if (err) {
    console.log(err);
  }
});
```



```

/** db connection */
connection.connect(function (err) {
  if (err) {
    console.log(err);
  }
  else {
    console.log("connection successfull");
  }
});

app.get("/", function (req, res) {
  if(req.session.userID){
    res.redirect("home");
  }
  else res.render("index");
});

app.get("/register", function (req, res) {
  if(req.session.userID){
    res.redirect("home");
  }
  else{
    res.render("register");
  }
})
app.post("/register", function(req, res){

  let name = req.body.name;
  let email = req.body.email;
  let password = req.body.password;

  bcrypt.hash(password, saltRounds, function(err, hash) {
    if(err) throw err;
    else{

      connection.query(`INSERT INTO persons values ('${name}', '${email}', '${hash}')`, function(err, result){
        if(err) {
          res.render("failure2");
        }
        else{

```

```

/** LOGIN ROUTES */
app.get("/login", function (req, res) {
  if(req.session.userID){
    res.redirect("home");

  }
  else{
    res.render("login");
  }
})

app.post("/login", function(req, res){

  let email = req.body.email;
  let plainpassword = req.body.password;

  connection.query(`SELECT password1 FROM persons WHERE email = '${email}'`, function(err, result){

    if(err) console.log(err);

    else
    {
      bcrypt.compare(plainpassword, result[0].password1, function(error, match) {
        if(error){
          console.log(error);
          res.redirect("login");
        }
        else if (match) {
          req.session.userID = email;
          res.redirect("home");}
        else{
          res.render("failure1");
        }
      }
    }
  })
})
});

```

```

/** HOME ROUTE */
app.get("/home", function (req, res){
  if(req.session.userID){
    // console.log(req.session.cookie);
    // console.log(req.session.userID);
    // console.log(req.session.user);
    // console.log(req._read.name);
    // console.log(req.cookies);

    connection.query('select * from blogs where email = '${req.session.userID}' ', function(err, result){
      if(err) console.log(err);
      else {
        res.render("home", {
          dummy:result
        });
      }
    });
  }
  else{
    res.statusCode = 404;
    console.log("cannot find login page please check.");
    res.redirect("login");
  }
});

/** COMPOSE ROUTES */
app.get("/compose", function (req, res){
  if(req.session.userID){
    res.render("compose");
  }
  else{
    console.log("cannotfind login page please check.");
    res.redirect("login");
  }
});

app.post("/compose", function(req, res){

```

```

    },
    else{
      console.log("cannotfind login page please check.");
      res.redirect("login");
    }
  });

  app.post("/compose", function(req, res){
    let title = req.body.title;
    let content = req.body.content;

    console.log(title, content);

    connection.query('INSERT INTO blogs values ('${title}', '${content}', '${req.session.userID}')', function(err, result){
      if(err) {
        console.log("sorry some error occurred");
        console.log(err);
      }
      else{
        console.log("succesfully indserted blog.");
        res.redirect("/home");
      }
    });
  });

  /** LOGOUT ROUTES */
  app.get("/logout", function(req, res){
    req.session.destroy();
    res.redirect("/");
  })

  /** LISTENING AT PORT */
  app.listen(process.env.LISTENING_PORT, function () {
    console.log("app is listening on port",process.env.LISTENING_PORT);
  });

```

Index.ejs

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Poppins&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <!-- NAVBAR -->
  <div class="container" id="home">
    <div class="logo">
      
      <p style="font-size: 28px;">Blog</p>
    </div>
    <div class="hamburger">
      
    </div>
    <div class="list">
      <li class="item"><a href="#about"> Home </a></li>
      <li class="item"><a href="#features"> Features </a></li>
      <li class="item"><a href="#contact"> Contact Us </a> </li>
    </div>
  </div>
  <!-- HERO SECTION -->
  <div class="hero" id="about">
    <div class="box2">
      <h1 class="heading">welcome everyone</h1>
      <p>Now you can store all your journals securely.</p>
      <h3><button class="btn"> <a href="/register"> Get Started </a></button></h3>
    </div>
    <div class="box1">
      
    </div>
  </div>
  <!-- FEATURES SECTION -->
  <div class="main-head">Features</div>

```

```

</div>
</div>
<!-- FEATURES SECTION -->
<h2 class="main-head">Features</h2>
<div class="content" id="features">
  <div class="elem">
    <h3 class="headd">EASY TO USE</h3>
    <hr>
    <p class="headd">This site is very easy to use, And beginner friendly.</p>
  </div>
  <div class="elem">
    <h3 class="headd">SAFE AND SECURE</h3>
    <hr>
    <p class="headd">This site store all your data securely and safely.</p>
  </div>
  <div class="elem">
    <h3 class="headd">FREE OF COST</h3>
    <hr>
    <p class="headd">This site charges nothing but an honest feedback.</p>
  </div>
</div>
<h2 class="main-head">Contact Us</h2>
<div class="foot logo">
  
  <p>Blog</p>
</div>
<!-- FOOTER -->
<div class="footer" id="contact">
  <!-- LOGO -->
  <div class="etc">
    <div class="ulist">
      <li><a href="#home"> Home</a></li>
      <li><a href="#about"> About</a></li>
      <li><a href="#features"> Features</a></li>
    </div>
    <div class="socials">

```

```

      
    </div>
    <div class="insta">
      
    </div>
    <div class="twitter">
      
    </div>
    <div class="github">
      
    </div>
  </div>
  <hr class="hrhr">
  <div class="copyrights">
    <p>© Copyright 2022, All Rights Reserved by Amit Mehra. </p>
  </div>
</div>
</body>
</html>

```

Style.css

```
body{
  margin: 0;padding: 0;
  font-family: 'Poppins', sans-serif;
  background-color: white;
  max-width: 70vw;
  margin: auto;
  scroll-behavior: smooth;
}
p{
  display: inline-block;
}
.container {
  display: flex;
  padding: 20px;
  justify-content: space-between;
}
.list{
  padding: 25px 0 0 5px;
}
.logo{
  display: flex;
  gap: 10px;
}
img{
  width: 35px;
  display: inline-block;
}
a{
  text-decoration: none;
  font-size: large;
  color: white;
}
.list{
  margin: 5px;
}
li{
  color: black;
}
a{
```

```

}
li:hover{
  text-decoration: underline;
  cursor: pointer;
}
.main-head{
  font-size: 2.5rem;
  text-align: center;
}
.hero{
  padding: 15px;
  box-sizing: border-box;
  display: grid;
  grid-template-columns: 1.2fr 0.8fr;
}
.banner
{
  width: 300px;
  padding: 20px;
  align-items: center;
  align-self: center;
  margin: auto;
}
.heading{
  font-size: 3rem;
}
.hero> p{
  font-size: small;
}
.btn{
  padding: 15px 30px;
  border-radius: 15px;
  border: none;
  background-color: #F07DEA;
  color: white;
  font-size: 1rem;
}
```

```

}
.content{
  display: flex;
}
.elem{
  border: 2px solid black;
  width: 30%;
  height: 170px;
  margin: 10px 10px 0px 5px;
  padding: 15px;
  border-radius: 10px;
}
p.head{
  text-align: center;
}
hr{
  width: 40%;
  background-color: #000;
  border-style:dashed;
}
.head{
  justify-content: center;
  text-align: center;
}
.footer{
  padding: 10px 0;
  border-radius: 10px;
  background-color: #ECC5FB;
  margin: 5px;
}
.etc{
  display: grid;
  grid-template-columns: 1fr 1fr ;
}
.foot.logo{
  width: min-content;

```

```

list-style:none;
}
.ulist{
  width: 70%;
  margin: auto;
  padding: 15px;
  display: flex;
  font-size: small;
  justify-content: space-between;
}
.socials
{
  margin: 5px;
  display: flex;
  justify-content: space-evenly;
}
.copyrights{
  font-size: small;
  padding: 15px;
  text-align: center;
  margin: auto;
}
.btn:hover{
  background-color: #7e64cd;
}
.ulist li{
  padding: 3px 5px;
  border-radius: 10px;
}
.ulist li:hover{
  text-decoration: underline;
  cursor: pointer;
}
.socials div{

```

```

.socials div{
  background: none;
}
.hh{
  background-color: white;
}
.hamburger{
  display: none;
}
.list{
  display: flex;
  gap: 25px;
}
img{
  cursor: pointer;
}
@media (max-width:700px) {
  .list{
    display: none;
  }
  .hamburger{
    margin: 20px;
    display: block;
  }
  .elem{
    display: block;
    height: fit-content;
    width: fit-content;
  }
}
.hrhr{

```

```
}  
.hrhr{  
  width: 84%;  
}  
@media (max-width:600px) {  
  .banner{  
    display: none;  
  }  
  .content{  
    display: inline;  
  }  
  .etc{  
    flex-wrap: wrap;  
  }  
  .hh{  
    display: none;  
  }  
  .ulist{  
    margin: auto  
    ;  
  }  
}  
@media (max-width:350px) {  
  body{  
    max-width: 90vw;  
  }  
  .container{  
    padding: 0;  
  }  
}
```

TESTING

In computer programming, **unit testing** is a software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is the smallest testable part of an application. In procedural programming a unit may be an individual function or procedure.

Ideally, each test case is independent from the others: substitutes like method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended. Its implementation can vary from being very manual (pencil and paper) to being formalized as part of build automation.

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits. Unit tests find problems early in the development cycle.

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all the "integrated" software components that have successfully passed integration testing and the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limiting type of testing; it seeks to detect defects both within the "inter assemblages" and within the system as a whole.

The following examples are different types of testing that should be considered during System testing:

- GUI software testing
- Usability testing
- Performance testing
- Compatibility testing
- Error handling testing
- Load testing
- Volume testing
- Stress testing
- Security testing
- Scalability testing
- Smoke testing
- Exploratory testing
- Ad hoc testing
- Regression testing
- Reliability testing
- Installation testing
- Maintenance testing
- Recovery testing / failover
- Accessibility testing,

Bibliography

Website references:

<https://getbootstrap.com>

<https://fontawesome.com>

<https://ejs.com>

<https://npmjs.com>

<https://expressjs.com>

<https://nodejs.com>

<https://google.com>

<https://mysql.com>