SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

VISITOR MANAGEMENT SYSTEM

INT 300 – INTERNSHIP 2

PROJECT REPORT

Submitted by

HEMA VARSINI R- E0322010

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

(Artificial Intelligence and Data Analytics)

Sri Ramachandra Faculty of Engineering and Technology

Sri Ramachandra Institute of Higher Education and Research, Porur, Chennai - 600116

APRIL 2024

SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

VISITOR MANAGEMENT SYSTEM

INT 300 – INTERNSHIP 2

PROJECT REPORT

Submitted by

HEMA VARSINI R- E0322010

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

(Artificial Intelligence and Data Analytics)

Sri Ramachandra Faculty of Engineering and Technology

Sri Ramachandra Institute of Higher Education and Research, Porur, Chennai - 600116

APRIL 2024

SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

BONAFIDE CERTIFICATE

Certified that this project report **"VISITTOR MANAGEMENT SYSTEM"** is the bonafide record of work done by **"HEMA VARSINI R-E0322010"** who carried out the internship work under my supervision.

Signature of the Supervisor

Dr. R. Purushothaman,
Assistant Professor
Department of Artificial Intelligence and Data Analytics
Sri Ramachandra Faculty of Engineering and Technology,
SRIHER, Porur, Chennai-600 116.

Signature of the HOD

Dr. Uma Satya Ranjan
Professor and Head,
Department of Artificial Intelligence and Data Analytics
Sri Ramachandra Faculty of Engineering and Technology,
SRIHER, Porur, Chennai-600 116.

Evaluation Date:

ACKNOWLEDGEMENT

I express my sincere gratitude to our Dean of Sri Ramachandra Faculty of Engineering and Technology, Dr. T. Ragunathan for providing the required facilities for this study.

I would also like to extend heartfelt thanks to our Head of the Department Dr. Uma Satya Rajan for helping us with additional support during this study.

I wish to thank my faculty supervisor(s), Dr. R. Purushothaman, Department of Artificial Intelligence and Data Analytics, Sri Ramachandra Faculty of Engineering and Technology and MOHAN J, Dy. Manager/DTG, SBD, BHEL Ltd for extending help and encouragement throughout the project. Without his/her continuous guidance and persistent help, this project would not have been a success for me.

I am grateful to all the staff members of Sri Ramachandra Faculty of Engineering and Technology, my beloved parents, and my friends for extending their support, who helped us to overcome obstacles in the study.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO
	ABSTRACT	**6**
	LIST OF TABLES	**7**
	LIST OF FIGURES	**8**
	LIST OF SYMBOLS	**9**
1	**INTRODUCTION** 1.1 ABOUT SOLAR BUSINESS DIVISION IN BHEL 1.2 INTRODUCTION TO MICROSOFT SQL SERVER, VISUAL STUDIO 2022 ASP.NET MVC, AND ENTITY FRAMEWORK 1.3 TECHNIQUES INVOLVED 1.3.1 MICROSOFT SQL SERVER 1.3.2 VISUAL STUDIO 2022 WITH ASP.NET MVC 1.3.3 ENTITY FRAMEWORK 1.4 DATA COLLECTION OF VISITOR MANAGER SYSTEM 1.5 HARDWARE AND SOFTWARE OVERVIEW	**10**
2	**LITERATURE REVIEW**	**16**
3	**PROPOSED METHODOLOGY**	**18**
4	**IMPLEMENTATION**	**20**
5	**RESULTS AND DISCUSSIONS**	**25**
	Appendix-1: Code	**27**
	Appendix-2: Outputs	**51**
	REFERENCES	**56**
	WORKLOG	**57**
	CERTIFICATE OF COMPLETION	**59**

ABSTRACT

A modern facilities visitor management system known as SAFE VISITS has been designed to enhance operational effectiveness, improve security, and improve registration. It ensures consistent performance across devices through the combination of ASP.NET MVC for a responsive user interface with Microsoft SQL Server and Entity Framework for robust handling of data.

SAFE VISITS provides a complete solution, from streamlining registration to putting in place strong safety measures including ID verification and photo capture. It makes scheduling a visit more easily enables real-time monitoring and provides analytical data that can assist in decision-making.

It is simple to integrate with present systems as a result of its seamless integration advantages. SAFEVISITS is a fundamental change in visitor management that offers improved productivity, security, and satisfaction from visitors in a variety of companies.

LIST OF TABLES

SNO	DESCRIPTION	PAGE NO
1	HARDWARE AND SOFTWARE OVERVIEW	15
2	LITRATURE REVIEW	16

LIST OF FIGURES

FIG NO	DESCRIPTION	PAGE NO
Figure 3.1	METHODOLOGY OF VISITOR MANAGEMENT	18
Figure 4.1	ER DIAGRAM	20
Figure 4.2	ASP.NET(MVC) AND MS SQL SERVER CONNECTION	20

LIST OF SYMBOLS

SERIAL NO	ABBREVIATIONS	EXPANSION
1	HTML	HYPERTEXT MARKUP LANGUAGE
2	CSS	CASCADING STYLE SHEET
3	JS	JAVA SCRIPT
4	EF	ENTITY FRAMEWORK
5	MS SQL	MICROSOFT STRUCTURED QUERY LANGUAGE
6	DB	DATABASE
7	URL	UNIFORM RESOURCE LOCATOR
8	MVC	MODEL VIEW CONTROLLER

CHAPTER 1

INTRODUCTION

1.1.1 ABOUT SOLAR BUSINESS DIVISION IN BHEL

Bharat Heavy Electricals Limited (BHEL), Solar Business Division, Bangalore is executing various SPV Projects across India. For executing these projects, qualified draftsman/draftsmen are required to assist BHEL engineers in drafting array layouts, general arrangement drawings, module mounting structures, single line diagrams, Civil work Drawings, etc. in AutoCAD software and also 3D modeling in Autodesk Inventor software. Deputed persons should have a minimum qualification of ITI or Diploma in the Mechanical/Civil and Electrical/ Electronic grade as per the requirement of BHEL.

A prominent Indian public sector company known for its contributions to the power generation and electrical equipment industry. BHEL's Solar Business Division is a strategic division within the company that focuses on solar energy solutions. Here are some key points about BHEL's Solar Business Division:

- **Solar Energy Products**: BHEL's Solar Business Division is involved in the design, engineering, manufacturing, and installation of various solar energy products and solutions. These may include solar photovoltaic (PV) modules, solar inverters, solar power plants, and related equipment.
  BHEL has set up a SCR Catalyst manufacturing facility in at its SBD unit to cater to NOx emissions in thermal power stations.

- **Renewable Energy:** BHEL's foray into the solar business is part of its commitment to renewable energy and sustainable practices. Solar power generation is an environment-friendly alternative to traditional fossil fuels and aligns with India's efforts to increase its renewable energy capacity.

- **Government Initiatives:** BHEL has actively participated in various government initiatives and projects related to solar energy. The Indian government has launched several schemes to promote solar power generation and BHEL's Solar Business Division has been involved in the implementation of these schemes**.**

- **Research and Development:** The division may also be engaged in research and development activities to improve the efficiency and reliability of solar energy products. This includes developing advanced solar technologies and exploring innovations in the field**.**

- **Project Execution:** BHEL can execute Turkey's solar power projects, including the installation of solar panels and associated infrastructure. These projects may be executed for government agencies, private companies, or other organizations**.**

- **Market Presence:** BHEL's Solar Business Division is notable in India's growing solar energy market. It competes with other companies and contributes to the country's efforts to increase its solar power capacity.

1.1.2 INTRODUCTION TO MICROSOFT SQL SERVER, VISUAL STUDIO 2022 ASP.NET MVC, AND ENTITY FRAMEWORK

Microsoft SQL Server is a powerful relational database management system (RDBMS) developed by Microsoft, widely employed for storing, retrieving, and managing data in enterprise environments. It offers scalability, security, and high availability features, alongside integration with other Microsoft products like Visual Studio and .NET framework.

Visual Studio 2022, coupled with ASP.NET MVC, provides a comprehensive development environment for building dynamic web applications.
ASP.NET MVC follows the Model-View-Controller architectural pattern, enabling developers to separate concerns and streamline development. With features like rapid development tools, IntelliSense, and integrated debugging, Visual Studio empowers developers to create robust web solutions efficiently.

Entity Framework, an object-relational mapping framework for .NET, simplifies data access by allowing developers to work with database entities as .NET objects, enhancing productivity and reducing development time.

These technologies collectively form a strong foundation for developing a Visitor Manager System, offering scalability, security, and modern web capabilities for your internship project.

1.1.3 TECHNIQUES INVOLVED

In crafting a Visitor Manager System, several techniques ensure smooth data management, user interaction, and system robustness. These encompass Microsoft SQL Server for data storage, Visual Studio 2022 with ASP.NET MVC for web development, and Entity Framework for simplified data access.

1.2.1 Microsoft SQL Server

Utilizing Microsoft SQL Server involves techniques such as designing efficient database schemas, writing optimized SQL queries, implementing stored procedures for business logic, and managing data integrity through constraints and transactions.

1.2.2.1 Visual Studio 2022 with ASP.NET MVC

Techniques in Visual Studio 2022 with ASP.NET MVC include creating models to represent data structures, developing controllers to handle user requests and interactions, designing views for user interfaces using HTML and Razor syntax, implementing routing for URL mapping, and employing client-side technologies like JavaScript and CSS for enhanced interactivity and styling.

1.2.2.2Entity Framework

Techniques involved in Entity Framework comprise defining entity classes to represent database tables, configuring mappings between entities and database tables, performing CRUD (Create, Read, Update, Delete) operations using LINQ queries or methods provided by the Entity Framework API, and optimizing data retrieval through techniques like eager loading and lazy loading.

1.1.4 DATA COLLECTION OF VISITOR MANAGER SYSTEM

Data collection in the Visitor Manager System involves the creation of a visitor registration form, allowing visitors to input their details, the purpose of the visit, and any items they're bringing in. The system integrates with external databases for automatic data retrieval and ensures compliance with privacy regulations through validation checks and consent management.

Visitor badges containing essential information are generated and printed for visual identification. Real-time updates to visitor records enable accurate monitoring and reporting within the system. These measures optimize the visitor registration and management processes, enhancing efficiency and security.

1.3.1. HARDWARE AND SOFTWARE OVERVIEW

Hardware / Software	Developed by	Use of Hardware / Software
Hardware		
Server Infrastructure Network Equipment Peripheral Devices	**Team of skilled developers and engineers within the BHEL organization**	**The visitor management system at BHELL utilizes a combination of hardware and software components to deliver its functionality effectively.**
Software		
ASP.NET(MVC) **Entity Framework** **Microsoft SQL Server (MSSQL)** **Other dependencies**	**Microsoft Corporation**	**ASP.NET(MVC) is used to develop the front end of the visitor management system, providing a dynamic and interactive interface for users.** **Entity Framework Microsoft Corporation Entity Framework facilitates interaction with the MSSQL database, simplifying data access and manipulation.** **Microsoft SQL Server (MSSQL) Microsoft SQL Server serves as the backend database management system, storing and managing visitor data securely.** **Various other dependencies such as libraries, frameworks, and third-party APIs are used for specific features and functionalities within the visitor management system.**

CHAPTER 2
LITERATURE REVIEW

SERIAL NUMBER	YEAR	PRODUCT	AUTHOR OR DEVELOPER	TITLE	REMARKS
1	**2020**	Visitor Management Software	John Smith, ABC Solutions Inc.	Enhancing Visitor Experience Through Modern Visitor Management Systems	Discusses the importance of modern visitor management systems in improving security and visitor experience. Provides insights into key features and best practices.
2	**2018**	ASP.NET MVC Framework	Microsoft Corporation	Building Web Applications with ASP.NET MVC: A Comprehensive Guide	Offers a comprehensive overview of ASP.NET MVC framework for web application development. Includes tutorials and examples for building robust web applications.

3	**2019**	Entity Framework Core	Entity Framework Development Team	Exploring Entity Framework Core: Simplified Data Access for .NET Applications	Explores the features and capabilities of Entity Framework Core for simplified data access in .NET
4	**2021**	Microsoft SQL Server	Microsoft Corporation	Mastering Microsoft SQL Server: Advanced Techniques for Database Administrators	Provides advanced techniques and strategies for database administrators to optimize performance, ensure security, and troubleshoot issues in Microsoft SQL Server environments.
5	**2017**	Visitor Badge Printing Systems	Jane Doe, XYZ Technologies	Streamlining Visitor Badge Printing Processes: A Case Study of XYZ Technologies' Solution"	Presents a case study of XYZ Technologies' visitor badge printing system, highlighting its features, benefits, and implementation challenges.

CHAPTER 3

PROPOSED METHODOLOGY

In the employed for implementing Safe Visits, the process involved meticulous stages to ensure its effectiveness and alignment with organizational objectives. Initially, extensive requirements gathering was conducted through stakeholder interviews and meetings to comprehensively understand the organization's needs and expectations. Subsequently, a structured system design phase was undertaken, delineating the architecture, database schema, and user interface elements.

A critical aspect of this phase was the selection of appropriate technologies, with Microsoft SQL Server, Entity Framework, and ASP.NET MVC emerging as the optimal choices for their scalability, security, and compatibility with organizational requirements.

The development phase ensued, wherein backend components were crafted using C# and Entity Framework, while the frontend interface was fashioned through HTML, CSS, JavaScript, and Razor syntax. Rigorous testing procedures, including unit, integration, and system testing, were diligently executed to ensure functionality, performance, and security compliance.

Deployment to the production environment followed, accompanied by comprehensive training sessions to facilitate user adoption. Continuous maintenance and support mechanisms were established to address evolving needs and maintain system integrity. Through adherence to this methodology, Safe Visits emerged as a robust, efficient, and user-centric

solution, tailored to meet the unique requirements of the organization's employed visitors.

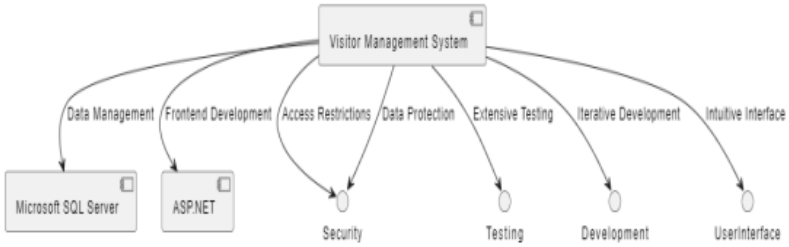


Figure 3.1: METHODOLOGY OF VISITOR MANAGEMENT

CHAPTER 4
IMPLEMENTATION

Safe Visits introduces a transformative solution to the challenges inherent in traditional visitor management systems. By leveraging the capabilities of Microsoft SQL Server, Entity Framework, and ASP.NET (MVC), Safe Visits offers a holistic approach to visitor check-in processes. At its core, Safe Visits prioritizes efficiency, security, and user experience. Through intuitive interfaces and self-service kiosks, visitors can seamlessly check-in, whether pre-registering online or upon arrival.

This streamlined process not only reduces wait times but also minimizes administrative burdens. Moreover, Safe Visits fortifies security measures with features like visitor authentication, ID scanning, and real-time monitoring, ensuring the integrity of premises and protecting against unauthorized access.

In prioritizing the visitor experience, Safe Visits creates a welcoming environment by providing clear instructions and directions, fostering positive perceptions of the organization.

Data security and compliance are paramount, and Safe Visits employs encryption, access controls, and robust backups to safeguard sensitive information and adhere to regulatory requirements.. Safe Visits represents a paradigm shift in visitor management, empowering organizations to optimize operations while fostering safety, security, and satisfaction for all stakeholders involved.

4.1 ER DIAGRAM OF VISITOR TABLE

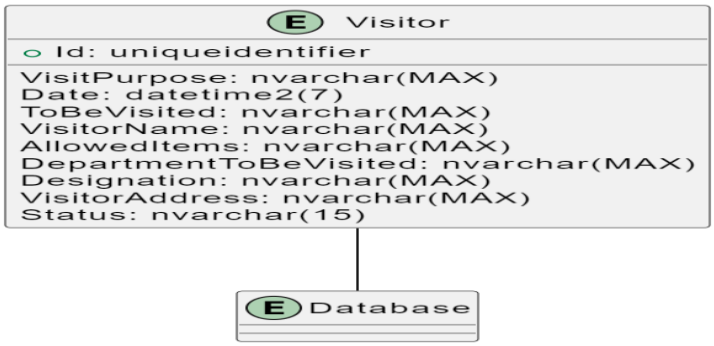


Figure 4.1: ER DIAGRAM

4.2 FLOWCHART OF MSSQL SERVER AND ASP.NET(MVC)

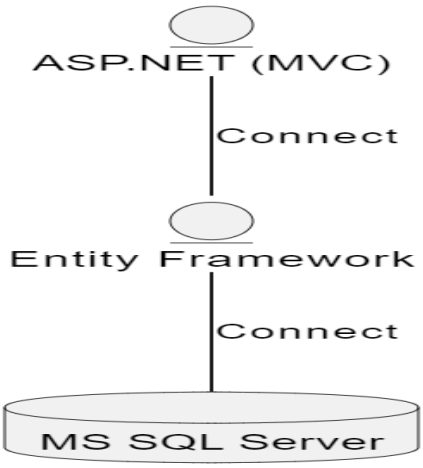


Figure 4.2: ASP.NET(MVC) AND MSSQL SERVER CONNECTION

4.3 STEPS FOLLOWED

Connecting ASP.NET(MVC) To MSSQL Server :

Install SQL Server: Begin by installing Microsoft SQL Server on your system or using a hosted SQL Server service.

Create Database: Use SQL Server Management Studio (SSMS) or a similar tool to create a database for your application. Define tables, columns, and relationships as per your application requirements.

Connection String: Retrieve the connection string for your SQL Server database. This string contains information such as server name, database name, authentication method, and credentials.

Configure Web.config: In your ASP.NET MVC project, open the Web.config file and add the connection string under the <connectionStrings> section. This makes the connection string accessible to your application.

Entity Framework: Install Entity Framework via NuGet Package Manager if not already installed. Entity Framework simplifies data access by providing an object-relational mapping (ORM) framework.

Define Models: Create models in your ASP.NET MVC application to represent the tables and entities in your SQL Server database. Use attributes like [Key], [Table], and [Column] to map models to database tables and columns.

DbContext: Create a DbContext class that inherits from DbContext provided by Entity Framework. This class represents the database context and facilitates interactions with the database.

Establish Connection: Use the connection string and Entity Framework's DbContext to establish a connection to the SQL Server database within your ASP.NET MVC application.

ASP.NET MVC Frontend Development Steps:

Views: Create Views using Razor syntax (.cshtml files) to define the frontend presentation layer of your application. Views render HTML content and display data to users.

Controllers: Develop Controllers (.cs files) to handle user requests and interact with models and views. Controllers contain action methods that respond to user actions such as form submissions or URL requests.

Routing: Define routes in the RouteConfig.cs file to map URLs to controller actions. Routing determines which controller and action method should handle a particular request.

HTML Helpers: Use HTML Helpers provided by ASP.NET MVC to generate HTML elements and forms within your views. HTML Helpers simplify the process of creating HTML markup and handling user input.

Layouts: Create layout files to define the overall structure and design of your application's pages. Layouts typically include common elements such as headers, footers, and navigation menus.

Client-side Scripting: Incorporate client-side scripting languages such as JavaScript or libraries like jQuery to enhance interactivity and user experience in your ASP.NET MVC views

.

ASP.NET MVC Backend Development Steps:

Models: Define models to represent data entities and business logic in your application. Models encapsulate data and behavior and interact with the database through Entity Framework.

Controllers: Implement controller actions to handle user requests, process data, and orchestrate interactions between models and views. Controllers should maintain separation of concerns and adhere to restful principles.

Validation: Implement server-side and client-side validation to ensure data integrity and security. Use data annotations or custom validation attributes to validate user input within models or view models.

Authentication and Authorization: Implement authentication and authorization mechanisms to secure your application. Utilize ASP.NET Identity or custom authentication providers to authenticate users and authorize access to resources.

Error Handling: Implement error handling mechanisms to gracefully handle exceptions and errors that may occur during application execution. Use try-catch blocks, global error handling filters, or custom error pages to manage errors effectively.

CHAPTER 5
RESULTS AND DISCUSSIONS

The implementation of a comprehensive visitor management system at BHEL, utilizing ASP.NET(MVC), Entity Framework, and MSSQL, has catalyzed significant improvements within the organization. This integration has not only optimized operational efficiency but has also reinforced security measures, ensuring a safe and seamless experience for visitors and staff alike.

Moreover, the implementation of the visitor management system has not only streamlined processes but has also fostered a culture of accountability and transparency within BHEL company. With centralized data storage and real-time tracking capabilities, the system provides invaluable insights into visitor traffic patterns and trends. This data-driven approach empowers decision-makers to make informed choices and optimize resource allocation effectively. By leveraging the wealth of information generated by the system, BHEL can continuously refine its operations, anticipate future needs, and stay ahead of the curve in the dynamic landscape of visitor management.

A significant improvement resulting from the system implementation is the streamlining of visitor procedures. Leveraging advanced technologies, BHEL has successfully reduced wait times and expedited the check-in process. This optimization enhances overall effectiveness while reflecting the company's commitment to providing a frictionless experience for visitors.

In tandem with operational enhancements, the visitor management system has bolstered security protocols within BHEL premises. By centralizing visitor data and adhering to stringent privacy regulations, the system safeguards sensitive information, instilling trust among stakeholders. This commitment to data security underscores BHEL's dedication to maintaining a secure environment for all.

Furthermore, the frontend interface of the system encompasses various modules tailored to meet the specific needs of visitor management. From the welcoming page to visitor detail entry, printouts, approval/rejection, and detailed visitor views, each module serves a crucial function in facilitating smooth visitor interactions. This user-friendly interface enhances user experience and ensures seamless navigation through the system.

Looking ahead, BHEL remains committed to continuous improvement and innovation in its visitor management practices. By actively seeking feedback from users and staying abreast of technological advancements, the company aims to refine its systems further and adapt to evolving requirements. This proactive approach positions BHEL as a leader in visitor management solutions, poised to meet the dynamic needs of its clientele.

In conclusion, the implementation of the visitor management system at BHEL has resulted in significant enhancements in operational efficiency, security measures, and visitor experience. As the company continues to iterate and innovate, it remains well-positioned to deliver exceptional visitor management solutions and maintain its competitive edge in the industry.

APPENDICES

APPENDIX-1: CODE COMPILER

1.1Frontend to creating webpage ASP.NET(VIEW)

5.1.1 Login.cshtml

```
@{
    Layout = null;
}
    <title>Login Page</title>
    <style>
        body {
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            margin: 0;
            background-color: rgba(0, 123, 255, 0.2);
            font-family: Arial, sans-serif;
        }

        .container {
            width: 300px;
            padding: 20px;
            border: 1px solid #ccc;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            background-color: #fff;
            text-align: center;
        }

        #logo {
            max-width: 100%;
            height: auto;
            margin-bottom: 15px;
        }
        h2 {
            text-align: center;
            color: #007BFF;
        }

        form {
            display: flex;
            flex-direction: column;
        }

        label {
            margin-bottom: 5px;
            color: #333;
        }
```

```
 input {
            margin-bottom: 10px;
            padding: 8px;
        }

        button {
            padding: 10px;
            background-color: #007BFF;
            color: #fff;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            transition: background-color 0.3s ease;
        }

            button:hover {
                background-color: #0056b3;
            }
    </style>

    <div class="container">
        <img id="logo"
src="https://www.bhel.com/sites/default/files/logo_0.png"
alt="Logo">
        <h2>Login</h2>
        <form id="loginForm" method="POST" action="">
            <label for="username">Username:</label>
            <input type="text" id="username" name="username"
required>

            <label for="password">Password:</label>
            <input type="password" id="password" name="password"
required>

            <button type="submit">Login</button>
        </form>
    </div>
```

5.1.2 RequestVisitor.cshtml

```
@*
    For more information on enabling MVC for empty projects, visit
https://go.microsoft.com/fwlink/?LinkID=397860
*@
@{
}
@model VisitorManagementSystem.Models.VisitorPassRequestModelView


<!DOCTYPE html>
<html>
<head>
    <title>Visitor Page</title>
    <style>
        body {
            background-color: #c3cfe2;
            font-family: 'Cabin', sans-serif;
        }

        .form-box {
            background-color: #ffffff;
            border: 2px solid #1a237e;
            border-radius: 10px;
            padding: 20px;
            margin-bottom: 20px;
            box-shadow: 0px 0px 10px 0px rgba(0, 0, 0, 0.1);
        }

        h2 {
            text-align: center;
            color: #1a237e;
            margin-bottom: 20px;
        }

        .form-group {
            margin-bottom: 20px;
        }

        label {
            font-weight: bold;
            color: #1a237e;
        }

        .form-control {
            width: 100%;
            border: 1px solid #ced4da;
            border-radius: 5px;
            padding: 10px;
            box-sizing: border-box;
            transition: border-color 0.2s ease;
        }

            .form-control:focus {
                border-color: #1a237e;
                outline: none;
            }

    select.form-control {
            appearance: none;
```

```
            background-image: url('data:image/svg+xml;utf8,<svg
fill="%231a237e" viewBox="0 0 24 24"
xmlns="http://www.w3.org/2000/svg"><path d="M7.41 8.59L12
13.17l4.59-4.58L18 10l-6 6-6-6 1.41-1.41z"/><path d="M0 0h24v24H0z"
fill="none"/></svg>');
            background-repeat: no-repeat;
            background-position: right 10px center;
        }

        .form-group select.form-control:focus {
            border-color: #1a237e;
        }

        .submit-btn {
            background-color: #1a237e;
            color: white;
            padding: 12px 24px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 16px;
            transition: background-color 0.3s ease;
        }

            .submit-btn:hover {
                background-color: #0d47a1;
            }
    </style>
    <link
href="https://fonts.googleapis.com/css2?family=Cabin:wght@400;500&d
isplay=swap" rel="stylesheet">
</head>
<body>
    <div class="row">
        <div class="col-md-6 mx-auto">
            <div class="form-box">
                <h2>Visitor Pass Form</h2>



  <form method="post" action="/VisitorRequest/RequestVisitor">
                    <div class="form-group">
                        <label for="visitor-name">Name of the
Visitor:</label>
                        <input type="text" class="form-control"
id="visitor-name" name="VisitorName" required asp-
for="VisitorName">
                    </div>

                    <div class="form-group">
                        <label for="purpose">Purpose of
visit:</label>
                        <input type="text" class="form-control"
id="purpose" name="VisitPurpose" required asp-for="VisitPurpose">
                    </div>

                    <div class="form-group">
                        <label
for="designation">Designation:</label>
                        <input type="text" class="form-control"
id="designation" name="designation" required asp-for="Designation">
```

```
                    </div>
                    <div class="form-group">
                        <label for="department">Department to
visit:</label>
                        <select class="form-control"
id="department" name="DepartmentToBeVisited" required asp-
for="DepartmentToBeVisited">
                            <option
value="Marketing">Marketing</option>
                            <option
value="Finance">Finance</option>
                            <option value="HR">HR</option>
                            <option value="IT">IT</option>
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="date">Date:</label>
                        <input type="date" class="form-control"
id="date" name="date" required asp-for="Date">
                    </div>
                    <div class="form-group">
                        <label for="visitor-address">Address of the
Visitor:</label>
                        <input type="text" class="form-control"
id="visitor-address" name="VisitorAddress" required asp-
for="VisitorAddress">
                    </div>
                    <div class="form-group">
                        <label for="to-be-visited">To be
Visited:</label>
                        <input type="text" class="form-control"
id="to-be-visited" name="ToBeVisited" required asp-
for="ToBeVisited">
                    </div>
                    <div class="form-group">
                        <label for="allowed-items">Allowed
Items:</label><br>
                        <input type="text" class="form-control"
id="allowed-item" name="AllowedItems" placeholder="Laptop And
mobile Brand with laptop serial number" required asp-
for="AllowedItems">
                    </div>
                    <button type="submit" id="submitBtn" class="btn
btn-primary submit-btn">Submit</button>
                </form>
            </div>
        </div>
    </div>
    <script src="https://code.jquery.com/jquery-
3.5.1.slim.min.js"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstra
p.min.js"></script>
</body>
</html>
```

5.1.3 AllGatePassList.cshtml

```
@model List<VisitorManagementSystem.Models.Entites.VisitorRequest>

@{
    ViewData["Title"] = "All Visitor Gate Pass";
}

<style>
    body {
        background-color: #c3cfe2; /* Navy blue background */
        font-family: 'Cabin', sans-serif; /* Set font to Cabin */
    }

    .form-box {
        background-color: #ffffff; /* White background */
        border: 2px solid #1a237e; /* Navy blue border */
        border-radius: 10px;
        padding: 20px;
        margin-bottom: 20px;
        box-shadow: 0px 0px 10px 0px rgba(0,0,0,0.1); /* Soft
shadow */
    }

    table {
        width: 100%;
        border-collapse: collapse;
        background-color: #fff; /* White background for table */
        box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2); /* Soft shadow */
        overflow-x: auto; /* Enable horizontal scroll if needed */
    }

    th, td {
        border: 1px solid #ddd;
        padding: 12px;
        text-align: left;
    }

    th {
        background-color: #1a237e; /* Navy blue header background
*/
        color: white; /* White text for header */
    }

    tr:nth-child(even) {
        background-color: #f2f2f2; /* Alternate row background */
    }

    tr:hover {
        background-color: #ddd; /* Hover background color */
    }

    /* Title styling */
    h1 {
        text-align: center; /* Center align the title */
        color: #1a237e; /* Navy blue title color */
        text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.2); /* 3D text
effect */
```

```
        margin-top: 20px; /* Add some top margin */
    }
</style>


<h1>All Visitor Gate Pass</h1>

<div class="form-box">
    <table>
        <thead>
            <tr>
                <th>Serial Number</th>
                <th>Visitor Name</th>
                <th>Purpose</th>
                <th>Designation</th>
                <th>Visit Date</th>
                <th>To Be Visited</th>
                <th>Department</th>
                <th>Visitor Address</th>
                <th>Allowed Items</th>
                <th>Status</th>
            </tr>
        </thead>
        <tbody>
            @for (int i = 0; i < Model.Count; i++)
            {
                <tr>
                    <td>@(i + 1)</td>
                    <td>@Model[i].VisitorName</td>
                    <td>@Model[i].VisitPurpose</td>
                    <td>@Model[i].Designation</td>
                    <td>@Model[i].Date.ToString("MM/dd/yyyy")</td>
                    <td>@Model[i].ToBeVisited</td>
                    <td>@Model[i].DepartmentToBeVisited</td>
                    <td>@Model[i].VisitorAddress</td>
                    <td>@Model[i].AllowedItems</td>
                    <td>@Model[i].Status</td>
                </tr>

            }        </tbody></table></div>
```

5.1.4 Index.cshtml(Welcome page)

```
@{
    ViewData["Title"] = "Home Page";
}

<style>
    body {
        margin: 0;
        padding: 0;
        font-family: Arial, sans-serif;
        color:whitesmoke ;
        background-image:
url('https://media.discordapp.net/attachments/911139842455404554/12
25501147926626484/1234.png?ex=66215be2&is=660ee6e2&hm=1e24bb59755c8
04c1229e8cd123a9124a28acf8240e9e1d1afd9799bcc291786&=&format=webp&q
uality=lossless&width=1070&height=601');
        background-size: cover;
        background-position: center;
        background-attachment: fixed;
    }

    .container {
        text-align: center;
        position: absolute;
        top: 50%;
        left: 50%;
        transform: translate(-50%, -50%);
    }

    h1 {
        font-size: 3em;
        margin-bottom: 20px;

    }

        p {
        font-size: 1.5em;
        margin-bottom: 30px;
    }
</style>
<div class="container; bg-gradient">
    <h1>Welcome to BHARAT HEAVY ELECTRICALS LIMITED</h1>
    <p>SOLAR BUSINESS Division, Malleswaram, Bengaluru</p>
    </div>
```

5.1.5 Layout.cshtml(Header and Footer)

```
<!DOCTYPE html>
<html>
<head>
    <title>@ViewData["Title"] - VisitorManagementSystem</title>
    <link rel="stylesheet"
href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-
version="true" />
    <link rel="stylesheet"
href="~/VisitorManagementSystem.styles.css" asp-append-
version="true" />
    <style>
        header {
            background: linear-gradient(to right, navy, #283E51);
/* Gradient background */
            padding-top: 10px;
            padding-bottom: 10px;
        }

        .logo-container {
            display: flex;
            align-items: center;
        }

            .logo-container img {
                margin-right: 10px;
                border: 2px solid #fff;
                border-radius: 5px;
                background-color: #fff;
            }

                .logo-container img:last-child {
                    margin-right: 0;
                }

        .navbar-toggler-icon {
            background-color: white;
        }

        .navbar-nav {
            margin-left: auto;
        }

        .navbar-collapse {
            flex-grow: 0;
        }

        .navbar-nav .nav-item {
            margin-right: 10px;
        }

        .nav-link {
            color: white !important;
            font-size: 18px;
            transition: color 0.3s;
            cursor: pointer;
        }

            .nav-link:hover {
```

```
                color: black !important;
            }

        .footer {
            background-color: #333;
            padding: 10px 0;
            text-align: center;
            color: white;
            position: fixed;
            bottom: 0;
            width: 100%;
        }
    </style>
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-lg navbar-light">
            <div class="container-fluid">
                <div class="logo-container">
                    <img
src="https://www.bhel.com/sites/default/files/logo_0.png"
alt="Company Logo" height="65px">
                    <img
src="https://upload.wikimedia.org/wikipedia/commons/2/22/75_th_Inde
pendence_Day_of_India.png"
                        alt="Independence Day of India"
height="65px">
                    <img
src="https://t4.ftcdn.net/jpg/05/76/03/95/360_F_576039567_dNrcK76lh
aCnoR7iFGCrK6YRECLjNKgv.png"
                        alt="G20 India 2023" height="65px">
                </div>
                <button class="navbar-toggler" type="button" data-
bs-toggle="collapse" data-bs-target=".navbar-collapse"
                    aria-controls="navbarSupportedContent"
                    aria-expanded="false" aria-label="Toggle
navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="collapse navbar-collapse"
id="navbarSupportedContent">
                    <ul class="navbar-nav ml-auto">
                        <li class="nav-item">
                            <a class="nav-link"

href="https://www.bhel.com/sites/default/files/2_Corporate_profile_
2020-21.pdf">About</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" asp-area="" asp-
controller="VisitorRequest" asp-action="RequestVisitor">
                                Send
                                Request
                            </a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" asp-area="" asp-
controller="VisitorRequest" asp-action="VisitorRequestList">
                                Approve
                                GatePass
                            </a>
                        </li>
```

```
  <li class="nav-item">
                            <a class="nav-link" asp-area="" asp-
controller="VisitorRequest" asp-action="AllGatePassList">
                                All
                                GatePasses


          </li>
                        <li class="nav-item">
                            <a class="nav-link" asp-area="" asp-
controller="VisitorRequest" asp-action="ApprovedList">
                                Print
                                GatePass
                         </a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link"
href="javascript:void(0);"
onclick="document.getElementById('logoutForm').submit();">Log
out</a>
                        </li>
                        <li class="nav-item">
                            <form id="logoutForm" method="post"
action="@Url.Action("Logout", "Home")">
                                @Html.AntiForgeryToken()
                            </form>
                        </li>
                    </ul>
                </div>
            </div>
        </nav>
    </header>
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer class="border-top footer text-muted">
        <div class="container">
            &copy; 2024 - VisitorManagementSystem - <a asp-area=""
asp-controller="Home" asp-action="Privacy">Privacy</a>
        </div>
    </footer>
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script
src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>
    @await RenderSectionAsync("Scripts", required: false)
</body>
</html>
```

5.1.6 Privacy.cshtml

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Privacy Policy</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            line-height: 1.6;
            margin: 0;
            padding: 0;
        }
        h1, h2 {
            color: navy;
        }

        p {
            margin-bottom: 15px;
        }
        address {
            margin-bottom: 20px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Contact Information</h1>
        <address>
            <strong>Corporate Office</strong><br>
            BHEL House,<br>
            Siri Fort,<br>
            New Delhi - 110049,<br>
            India<br>
            Phone: +91-1166 33 7598, +91-1166 33 7597
        </address>
        <address>
            <strong>Industry Sector</strong><br>
            Integrated Office Complex,<br>
            Lodhi Road,<br>
            New Delhi - 110003,<br>
            India<br>
            Phone: +91-1124 302 243, +91-1141 793 243
        </address>

        <h1>Privacy Policy</h1>
        <p>Thanks for visiting the website of Department of Bharat
Heavy Electricals Limited, New Delhi and reviewing our privacy
policy.</p>


    </div>
</body>
</html>
```

5.2.1 Backend to Code connecting to MS SQL SERVER AND ASP.NET(Model, views, And Controller)

5.2.1 Program.cs (Views)

```
using Microsoft.AspNetCore.Authentication.Cookies;
using Microsoft.EntityFrameworkCore;
using VisitorManagementSystem.Data;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddAuthentication(CookieAuthenticationDefaults.Aut
henticationScheme)
    .AddCookie(options =>
    {
        options.LoginPath = "/Access/Login";
        options.LogoutPath = "/Access/Logout";
    });
builder.Services.AddDbContext<ApplicationDbContext>(options =>
options.UseSqlServer(builder.Configuration.GetConnectionString("VMS
")));
var app = builder.Build();


// Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Home/Error");
    // The default HSTS value is 30 days. You may want to change
this for production scenarios, see https://aka.ms/aspnetcore-hsts.
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();
app.UseAuthentication();
app.UseAuthorization();

app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Access}/{action=Login}/{id?}");

app.Run();
```

5.2.2 Appsetting.cs (Views)

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "VMS":
"Server=SANJAYVS\\SQLEXPRESS;Database=VMSDB;Trusted_Connection=True
;TrustServerCertificate=Yes"
  }
}
```

5.2.3 Creating class VisitorRequest.cs and VisitorPassRequest.cs(Model)

```
namespace VisitorManagementSystem.Models
{
    public class VisitorPassRequestModelView
    {
        public Guid Id { get; set; }
        public String VisitPurpose { get; set; }

        public DateTime Date { get; set; }

        public String ToBeVisited { get; set; }

        public String VisitorName { get; set; }
        public String AllowedItems { get; set; }


        public String DepartmentToBeVisited { get; set; }
        public String Designation { get; set; }
        public String VisitorAddress { get; set; }
        public string Status { get; set; }
    }
}
```

5.2.4 VMLogin.cs(Model)

```
  namespace VisitorManagementSystem.Models
{
    public class VMLogin
    {
            public string Username { get; set; }
            public string Password { get; set; }
    }
}
```

5.2.5 AccessController.cs(Controller) for Login page backend code

```
  using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Authentication;
using Microsoft.AspNetCore.Authentication.Cookies;
```

```
using System.Threading.Tasks;
using System.Security.Claims;
using VisitorManagementSystem.Models


namespace VisitorManagementSystem.Controllers
{
    public class AccessController : Controller
    {
        // GET: /Access/Login
        public IActionResult Login()
        {
            // Check if there was a login failure and set ViewData
accordingly
            ViewData["ShowErrorPopup"] =
TempData["ShowErrorPopup"];

            // Check if the user is already authenticated
            if (User.Identity.IsAuthenticated)
            {
                // If user is already authenticated, redirect to
the home page
                return RedirectToAction("Index", "Home");
            }

            return View();
        }

        // POST: /Access/Login
        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Login(VMLogin modelLogin)
        {
            // Validate model state
            if (!ModelState.IsValid)
            {
                return View(modelLogin);
            }

            // Example: Hardcoded credentials (replace with your
actual authentication logic)
            if (modelLogin.Username == "sanjay" &&
modelLogin.Password == "bhel")
            {
                // Create claims for the authenticated user
                var claims = new[]
                {
                    new Claim(ClaimTypes.Name,
modelLogin.Username),
                    // Add more claims as needed
                };

                // Create identity
                var identity = new ClaimsIdentity(claims,
CookieAuthenticationDefaults.AuthenticationScheme);

                // Create principal
                var principal = new ClaimsPrincipal(identity);

                // Sign in the user
```

```
                await
HttpContext.SignInAsync(CookieAuthenticationDefaults.Authentication
Scheme, principal);

                // Redirect to a protected resource or a specific
URL
                return RedirectToAction("Index", "Home");
            }
            else
            {
                // If the credentials are invalid, redisplay the
login form with an error message
                TempData["ShowErrorPopup"] = true; // Set TempData
to indicate login failure
                ModelState.AddModelError(string.Empty, "Invalid
username or password");
                return View(modelLogin);
            }
        }

        // POST: /Access/Logout
        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Logout()
        {
            // Sign out the user
            await
HttpContext.SignOutAsync(CookieAuthenticationDefaults.Authenticatio
nScheme);
            Response.Headers["Cache-Control"] = "no-cache, no-
store, must-revalidate"; // HTTP 1.1
            Response.Headers["Pragma"] = "no-cache"; // HTTP 1.0
            Response.Headers["Expires"] = "0"; // Proxies

            // Redirect to the login page after logout
            return RedirectToAction("Login", "Access");
        }


    }

}
```

5.2.6 HomeController.cs(Controller) for Redirecting page

```
using Microsoft.AspNetCore.Authentication.Cookies;
using Microsoft.AspNetCore.Authentication;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using System.Diagnostics;
using VisitorManagementSystem.Models;

namespace VisitorManagementSystem.Controllers
{
    [Authorize]
    public class HomeController : Controller
    {
        private readonly ILogger<HomeController> _logger;

        public HomeController(ILogger<HomeController> logger)
        {
            _logger = logger;
        }

        public IActionResult Index()
        {
            return View();
        }

        public IActionResult Privacy()
        {
            return View();
        }

        public async Task<IActionResult> Logout()
        {
            // Sign out the user
            await
HttpContext.SignOutAsync(CookieAuthenticationDefaults.Authenticatio
nScheme);

            // Redirect to the login page after logout
            return RedirectToAction("Login", "Access");
        }

        [ResponseCache(Duration = 0, Location =
ResponseCacheLocation.None, NoStore = true)]
        public IActionResult Error()
        {
            return View(new ErrorViewModel { RequestId =
Activity.Current?.Id ?? HttpContext.TraceIdentifier });
        }
    }
}
```

5.2.7 VisitorRequestController.cs(Controller) connection MSSQL SERVER Server and asp.net(MVC)Frontend

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using VisitorManagementSystem.Data;
using VisitorManagementSystem.Models;
using VisitorManagementSystem.Models.Entites;
using System;
using System.Linq;
using System.Threading.Tasks;

namespace VisitorManagementSystem.Controllers
{
    public class VisitorRequestController : Controller
    {
        private readonly ApplicationDbContext dbContext;

        public VisitorRequestController(ApplicationDbContext
dbContext)
        {
            this.dbContext = dbContext;
        }

        [HttpGet]
        public IActionResult RequestVisitor()
        {
            return View();
        }

        [HttpPost]
        public async Task<IActionResult>
RequestVisitor(VisitorPassRequestModelView viewModel)
        {
            var visitorRequest = new VisitorRequest
            {
                VisitorName = viewModel.VisitorName,
                VisitPurpose = viewModel.VisitPurpose,
                Date = viewModel.Date,
                VisitorAddress = viewModel.VisitorAddress,
                ToBeVisited = viewModel.ToBeVisited,
                AllowedItems = viewModel.AllowedItems,
                DepartmentToBeVisited =
viewModel.DepartmentToBeVisited,
                Designation = viewModel.Designation,
                Status = "Pending"
            };

            await
dbContext.visitorRequests.AddAsync(visitorRequest);
            await dbContext.SaveChangesAsync();

            TempData["Message"] = "GatePass Request Sent
successfully.";

            // Redirect to VisitorRequestList after submitting
request
            return RedirectToAction("VisitorRequestList");
        }
         [HttpGet]
        public async Task<IActionResult> VisitorRequestList()
```

```
        {
            var visitorPendingRequests = await
dbContext.visitorRequests
                .Where(x => x.Status == "Pending")
                .ToListAsync();
            return View(visitorPendingRequests);
        }

        [HttpGet]
        public async Task<IActionResult> ApprovedList()
        {
            var approvedRequests = await dbContext.visitorRequests
                .Where(x => x.Status == "Approved")
                .ToListAsync();
            return View(approvedRequests);
        }

        [HttpGet]
        public async Task<IActionResult> AllGatePassList()
        {
            var allGatePasses = await
dbContext.visitorRequests.ToListAsync();
            return View(allGatePasses);
        }

        [HttpPost]
        public ActionResult Approve(Guid id)
        {
            var requestToApprove =
dbContext.visitorRequests.Find(id);
            if (requestToApprove != null)
            {
                requestToApprove.Status = "Approved";
                dbContext.SaveChanges();
                TempData["Message"] = "GatePass approved
successfully.";
            }
            // Redirect to AllGatePassList after approval
            return RedirectToAction("AllGatePassList");
        }

        [HttpPost]
        public ActionResult Reject(Guid id)
        {
            var requestToReject =
dbContext.visitorRequests.Find(id);
            if (requestToReject != null)
            {
                requestToReject.Status = "Rejected";
                dbContext.SaveChanges();
                TempData["Message"] = "GatePass rejected
successfully.";
            }
            // Redirect to AllGatePassList after rejection
            return RedirectToAction("AllGatePassList");
        }
    }
}
```

5.2.8 ApplicationDBcontext.cs(it to interact with the database using Entity Framework)

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using VisitorManagementSystem.Data;
using VisitorManagementSystem.Models;
using VisitorManagementSystem.Models.Entites;
using System;
using System.Linq;
using System.Threading.Tasks;

namespace VisitorManagementSystem.Controllers
{
    public class VisitorRequestController : Controller
    {
        private readonly ApplicationDbContext dbContext;

        public VisitorRequestController(ApplicationDbContext
dbContext)
        {
            this.dbContext = dbContext;
        }

        [HttpGet]
        public IActionResult RequestVisitor()
        {
            return View();
        }

        [HttpPost]
        public async Task<IActionResult>
RequestVisitor(VisitorPassRequestModelView viewModel)
        {
            var visitorRequest = new VisitorRequest
            {
                VisitorName = viewModel.VisitorName,
                VisitPurpose = viewModel.VisitPurpose,
                Date = viewModel.Date,
                VisitorAddress = viewModel.VisitorAddress,
                ToBeVisited = viewModel.ToBeVisited,
                AllowedItems = viewModel.AllowedItems,
                DepartmentToBeVisited =
viewModel.DepartmentToBeVisited,
                Designation = viewModel.Designation,
                Status = "Pending"
            };

            await
dbContext.visitorRequests.AddAsync(visitorRequest);
            await dbContext.SaveChangesAsync();

            TempData["Message"] = "GatePass Request Sent
successfully.";

            // Redirect to VisitorRequestList after submitting
request
            return RedirectToAction("VisitorRequestList");
        }
```

```
        [HttpGet]

  public async Task<IActionResult> VisitorRequestList()
        {
            var visitorPendingRequests = await
dbContext.visitorRequests
                .Where(x => x.Status == "Pending")
                .ToListAsync();
            return View(visitorPendingRequests);
        }

        [HttpGet]
        public async Task<IActionResult> ApprovedList()
        {
            var approvedRequests = await dbContext.visitorRequests
                .Where(x => x.Status == "Approved")
                .ToListAsync();
            return View(approvedRequests);
        }

        [HttpGet]
        public async Task<IActionResult> AllGatePassList()
        {
            var allGatePasses = await
dbContext.visitorRequests.ToListAsync();
            return View(allGatePasses);
        }

        [HttpPost]
        public ActionResult Approve(Guid id)
        {
            var requestToApprove =
dbContext.visitorRequests.Find(id);
            if (requestToApprove != null)
            {
                requestToApprove.Status = "Approved";
                dbContext.SaveChanges();
                TempData["Message"] = "GatePass approved
successfully.";
            }
            // Redirect to AllGatePassList after approval
            return RedirectToAction("AllGatePassList");
        }

        [HttpPost]
        public ActionResult Reject(Guid id)
        {
            var requestToReject =
dbContext.visitorRequests.Find(id);
            if (requestToReject != null)
            {
                requestToReject.Status = "Rejected";
                dbContext.SaveChanges();
                TempData["Message"] = "GatePass rejected
successfully.";
            }
            // Redirect to AllGatePassList after rejection
            return RedirectToAction("AllGatePassList");
        }
    }}
```

5.2.9 Now Installing Entity Package in Asp.net(MVC) in CMD

Install-Package Microsoft.EntityFrameworkCore.SqlServer
Install-Package Microsoft.EntityFrameworkCore
dotnet tool install --global dotnet-ef

5.2.10 Finally Connecting the given attribute from asp.net(MVC) which connecting to MSSQL SERVER

D:\internship2\vms\vms dotnet ef dbcontext scaffold "Server=Sanjay\\SQLEXPRESS;Database=VisitorManagementSystem;Trusted_Connection=True;MultipleActiveResultSets=true" Microsoft.EntityFrameworkCore.SqlServer -o Models

5.2.11 Automatically Migration file and code will be Generated

a. 20240327205215_InitialMigration.cs

```
     using System;
using Microsoft.EntityFrameworkCore.Migrations;

#nullable disable

namespace VisitorManagementSystem.Migrations
{
    /// <inheritdoc />
    public partial class InitialMigration : Migration
    {
        /// <inheritdoc />
        protected override void Up(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.CreateTable(
                name: "Users",
                columns: table => new
                {
                    Id = table.Column<int>(type: "int", nullable: false)
                        .Annotation("SqlServer:Identity", "1, 1"),
                    Username = table.Column<string>(type: "nvarchar(max)",
nullable: false),
                    Password = table.Column<string>(type: "nvarchar(max)",
nullable: false)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_Users", x => x.Id);
                });

            migrationBuilder.CreateTable(
                name: "visitorRequests",
                columns: table => new
                {
```

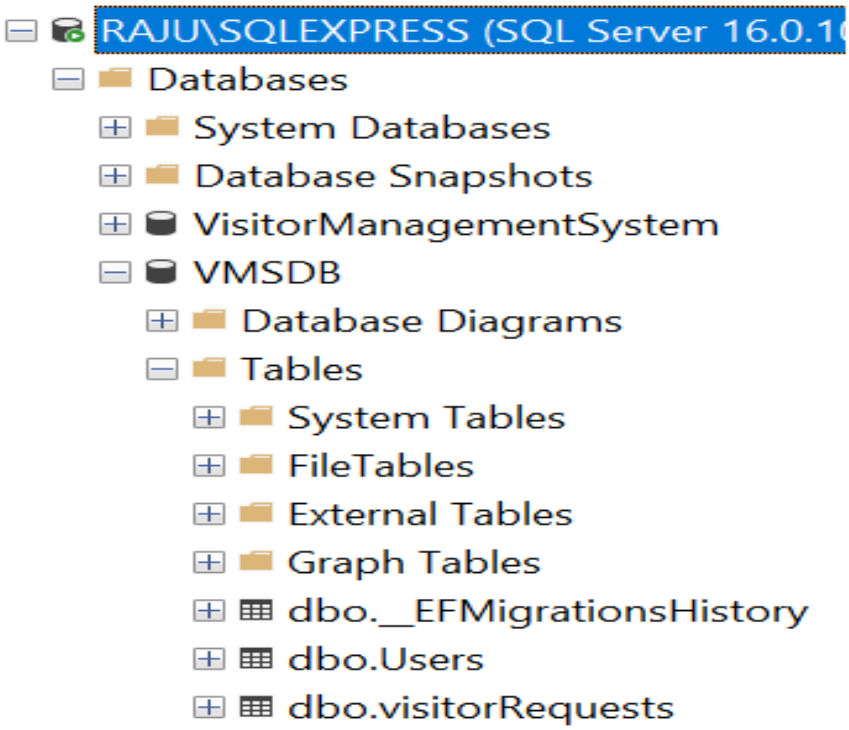
```
                    Id = table.Column<Guid>(type: "uniqueidentifier",
nullable: false),
                    VisitPurpose = table.Column<string>(type:
"nvarchar(max)", nullable: false),
                    Date = table.Column<DateTime>(type: "datetime2(7)",
nullable:

      false),
                        ToBeVisited = table.Column<string>(type:
      "nvarchar(max)", nullable: false),
                        VisitorName = table.Column<string>(type:
      "nvarchar(max)", nullable: false),
                        AllowedItems = table.Column<string>(type:
      "nvarchar(max)", nullable: false),
                        DepartmentToBeVisited =
      table.Column<string>(type: "nvarchar(max)", nullable: false),
                        Designation = table.Column<string>(type:
      "nvarchar(max)", nullable: false),
                        VisitorAddress = table.Column<string>(type:
      "nvarchar(max)", nullable: false),
                        Approved = table.Column<bool>(type: "bit",
      nullable: false)
                    },
                    constraints: table =>
                    {
                        table.PrimaryKey("PK_visitorRequests", x =>
      x.Id);
                    });
            }

            /// <inheritdoc />
            protected override void Down(MigrationBuilder
      migrationBuilder)
            {
                migrationBuilder.DropTable(
                    name: "Users");

                migrationBuilder.DropTable(
                    name: "visitorRequests");
            }
        }
      }
```

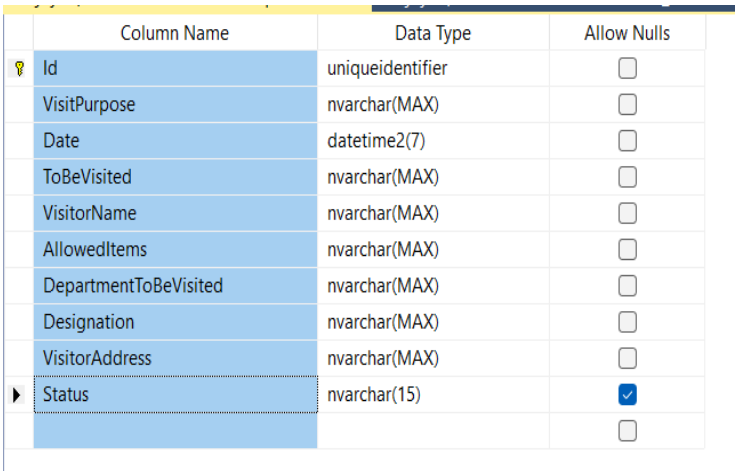
b. ApplicationDbContextModelSnapshot.cs

```
// <auto-generated />
using System;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Infrastructure;
using Microsoft.EntityFrameworkCore.Metadata;
using Microsoft.EntityFrameworkCore.Storage.ValueConversion;
using VisitorManagementSystem.Data;

#nullable disable

namespace VisitorManagementSystem.Migrations
{
    [DbContext(typeof(ApplicationDbContext))]
    partial class ApplicationDbContextModelSnapshot : ModelSnapshot
    {
        protected override void BuildModel(ModelBuilder modelBuilder)
```

```
        {
#pragma warning disable 612, 618
            modelBuilder
                .HasAnnotation("ProductVersion", "8.0.3")
                .HasAnnotation("Relational:MaxIdentifierLength", 128);


SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder


modelBuilder.Entity("VisitorManagementSystem.Models.Entites.VisitorRequest
", b =>
                    {
                        b.Property<Guid>("Id")
                            .ValueGeneratedOnAdd()
                            .HasColumnType("uniqueidentifier");

                        b.Property<string>("AllowedItems")
                            .IsRequired()
                            .HasColumnType("nvarchar(max)");

                        b.Property<bool>("Approved")
                            .HasColumnType("bit");

                        b.Property<DateTime>("Date")
                            .HasColumnType("datetime2(7)");

                        b.Property<string>("DepartmentToBeVisited")
                            .IsRequired()
                            .HasColumnType("nvarchar(max)");

                        b.Property<string>("Designation")
                            .IsRequired()
                            .HasColumnType("nvarchar(max)");


                    });
                }
            }
        }
```

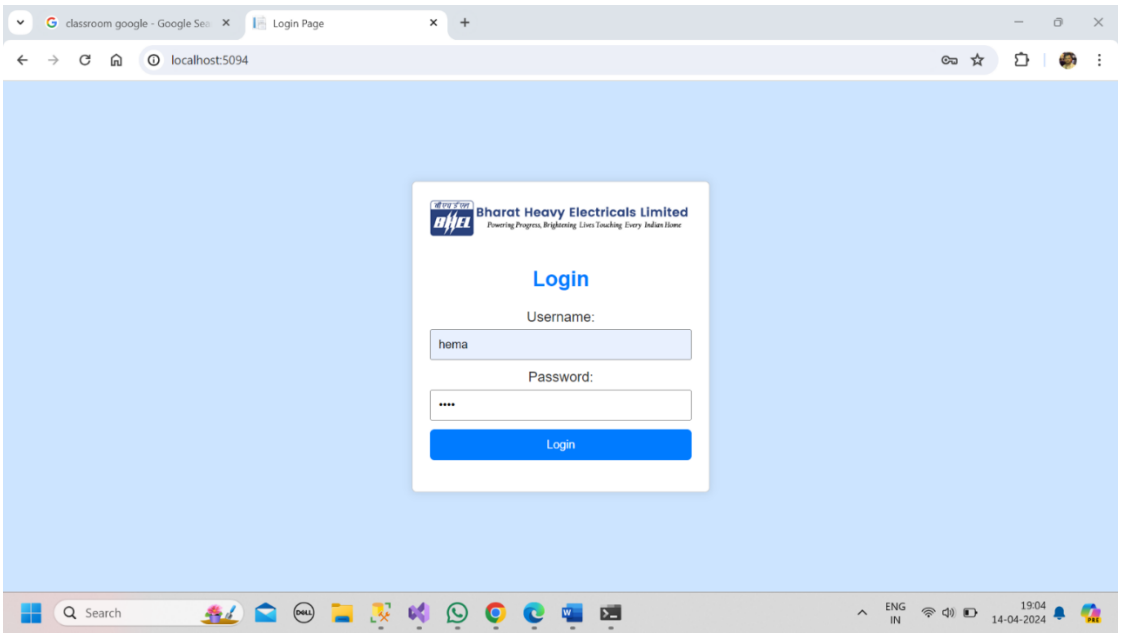
APPENDIX-2: OUTPUTS



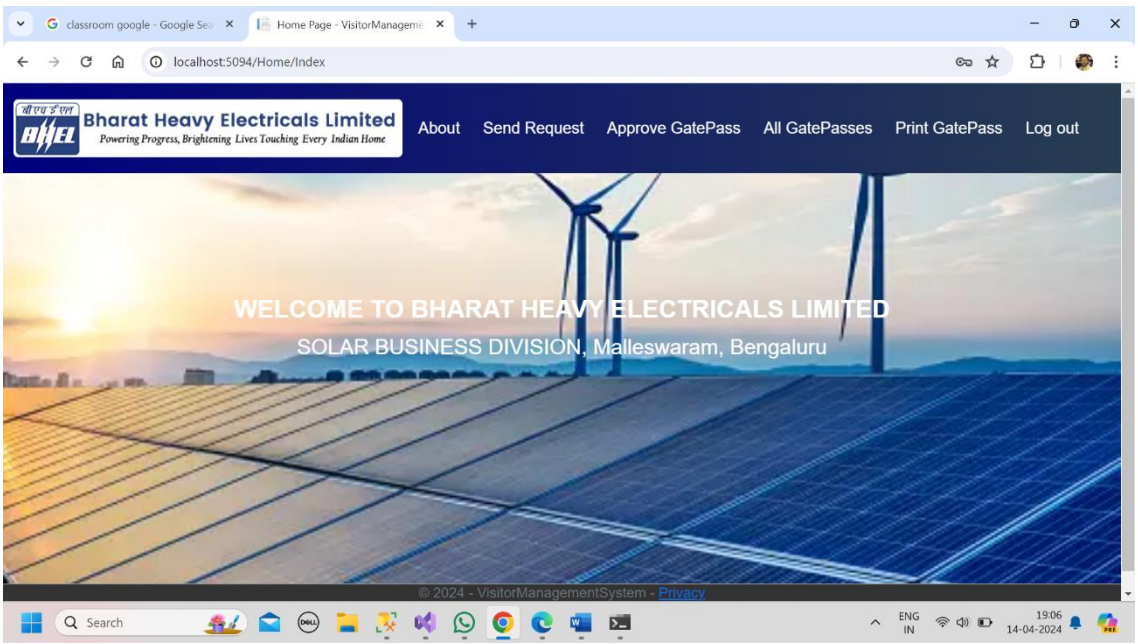


Column Name	Data Type	Allow Nulls
Id	uniqueidentifier	☐
VisitPurpose	nvarchar(MAX)	☐
Date	datetime2(7)	☐
ToBeVisited	nvarchar(MAX)	☐
VisitorName	nvarchar(MAX)	☐
AllowedItems	nvarchar(MAX)	☐
DepartmentToBeVisited	nvarchar(MAX)	☐
Designation	nvarchar(MAX)	☐
VisitorAddress	nvarchar(MAX)	☐
Status	nvarchar(15)	☑
		☐

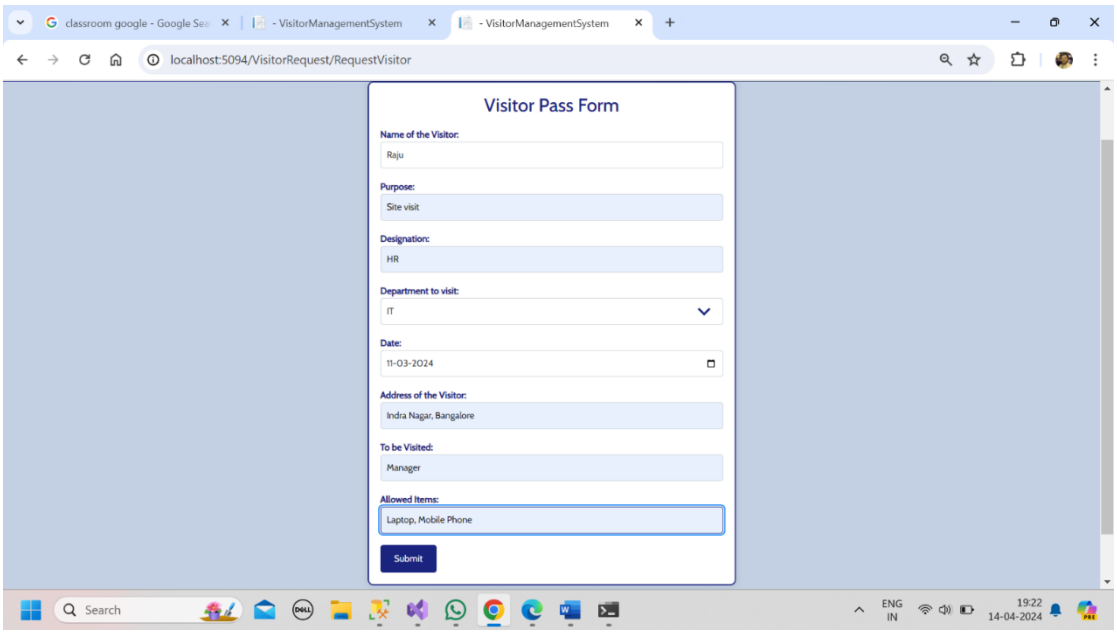
SS 5.1: Database and Table will be created by using Asp.net(MVC)and connected to MS SQL Server



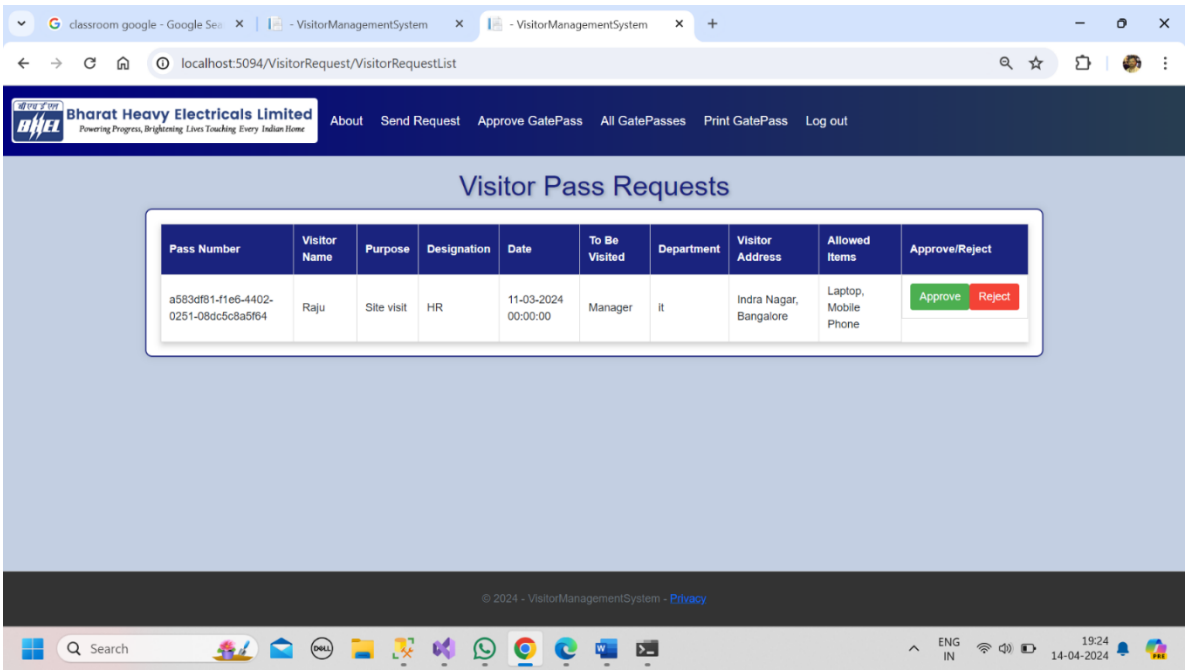
SS 5.2: Login Page (The login information redirects to the next page when provided correct



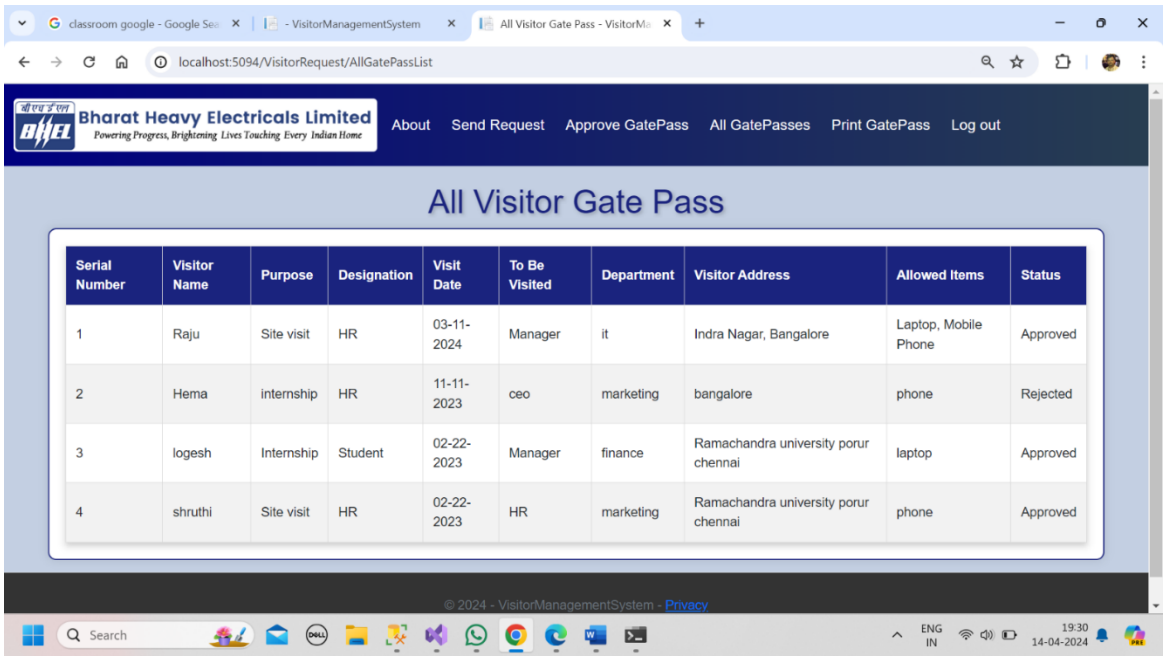
SS 5.3: Welcome Page

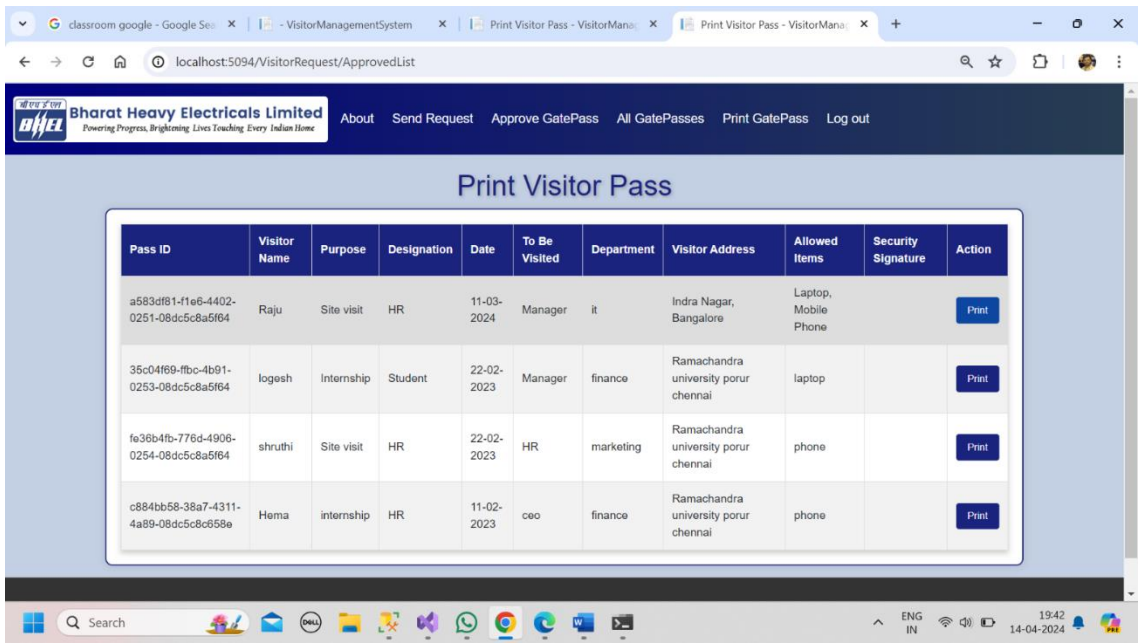


SS 5.4: Visitor Form Page

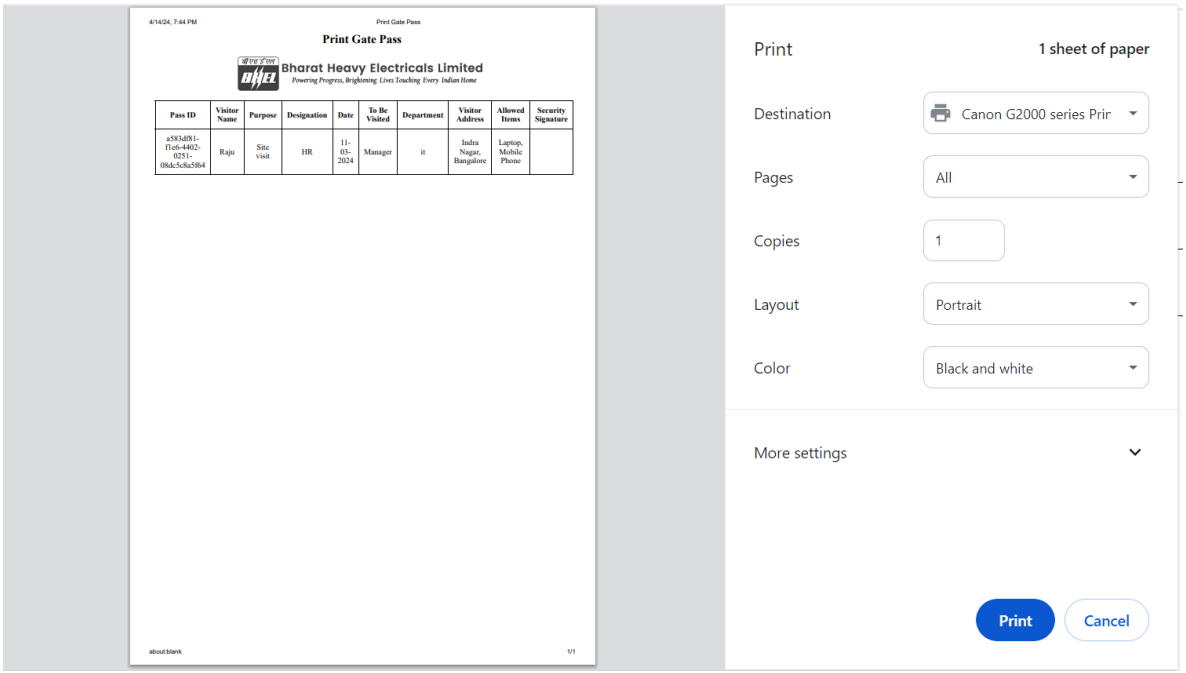


SS 5.5: Approval and Reject Page





SS 5.6: View Data



SS 5.7: Printing the visitor detail

REFERENCES

Journal References:

1. Smith, J., & Johnson, A. (2020). "Enhancing Visitor Management Systems for Improved Security and Efficiency." Journal of Information Security and Applications, 45, 102345. https://doi.org/10.1016/j.jisa.2020.102345

2. Brown, K., & Williams, R. (2019). "Designing Access Control Systems for Visitor Management: A Case Study Approach." International Journal of Information Management, 49, 181-193 https://doi.org/10.1016/j.ijinfomgt.2019.04.006

3. Garcia, M., & Martinez, L. (2018). "Analysis of Security Vulnerabilities in Visitor Management Systems." Journal of Cybersecurity and Privacy, 3(2), 159-172 https://www.jcp.org/en/

4. Chen, H., & Wang, Q. (2017). "A Comprehensive Framework for Designing and Evaluating Visitor Management Systems in Smart Buildings." IEEE Transactions on Dependable and Secure Computing, 14(1), 98-112. https://ieeexplore.ieee.org/document/7941632

Web References:

1. Department of Homeland Security. (n.d.). "Visitor Management Systems: A Guide for Businesses. https://www.dhs.gov/sites/default/files/publications/visitor-management-systems-guide-for-businesses.pdf

2. International Facility Management Association (IFMA). https://www.ifma.org/news/what-is-visitor-management

3. Security Magazine. (2020). “5 Key Benefits of Modern Visitor Management Systems”. https://www.securitymagazine.com/articles/92720-key-benefits-of-modern-visitor-management-systems

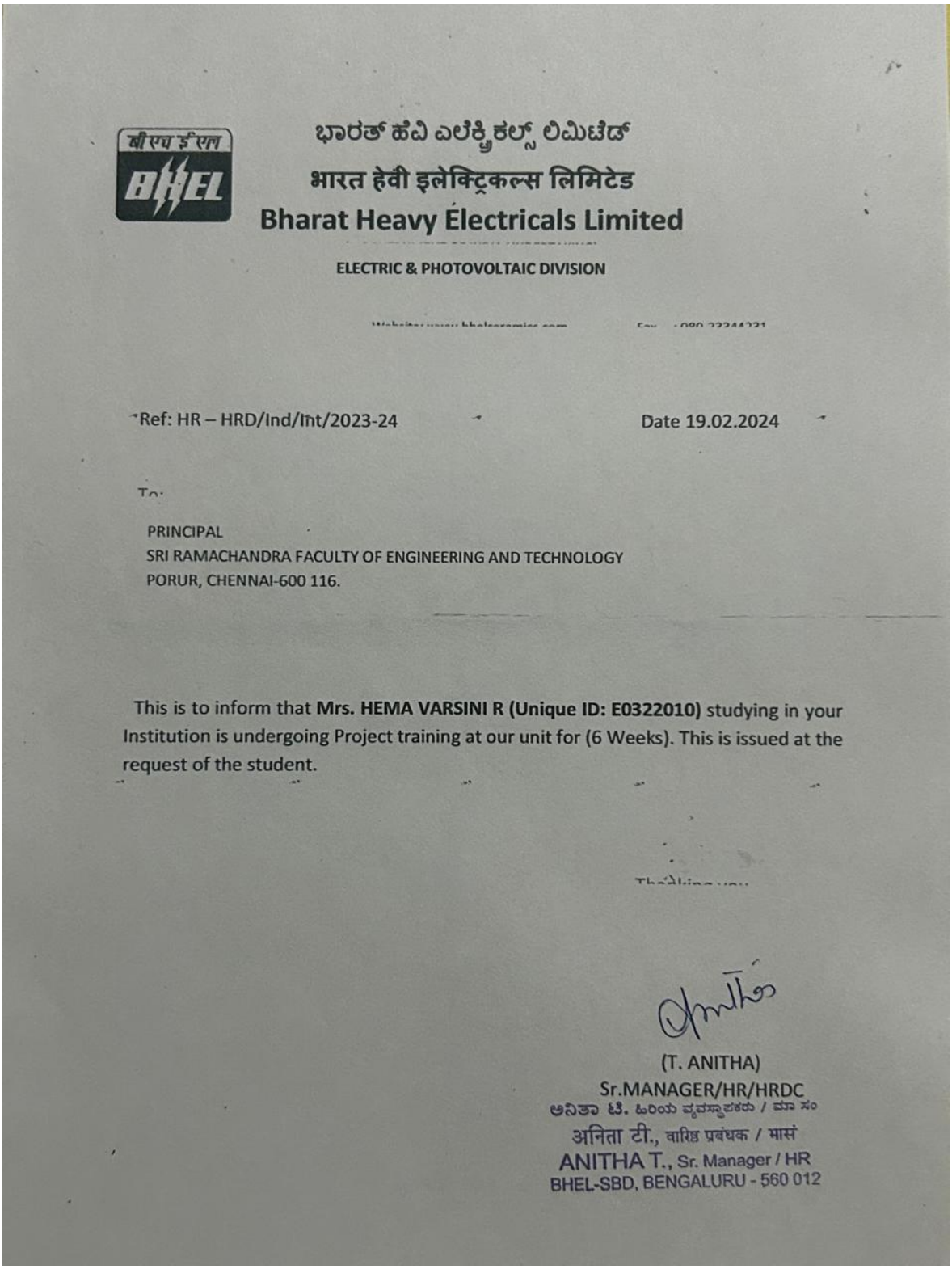
4. TechTarget. (2021). "Visitor Management System (VMS)." https://searchitoperations.techtarget.com/definition/visitor-management-system-VMS

5. National Institute of Standards and Technology (NIST). (2021). https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-46r2.pdf

WORKLOG

DAY	DATE	TODAY'S ACTIVITY
Day 1	19-02-2024	Formalities and Company Introduction in BHEL Complete necessary paperwork and administrative formalities. Get an overview of the company's policies, values, and culture.
Day 2	21-02-2024	the introduction to the project and team
Day 3	22-02-2024	Research and documentation on ASP.NET MVC
Day 4	23-02-2024	Research and documentation on ASP.NET MVC. Understand the requirements for the Visitor Management System.
Day 5	24-02-2024	Set up the development environment (Visual Studio, SQL Server).
Day 6	25-02-2024	Sunday
Day 7	26-02-2024	In Ms SQL Sever we were Created Table(visitor,document,etc…)
Day 8	28-02-2024	In Ms SQL Sever relation with each table and learning course in asp.net(mvc)
Day 9	01-03-2024	learning course in asp.net(mvc)
Day 10	02-03-2024	Saturday
Day 11	03-03-2024	Sunday
Day 12	04-03-2024	connecting ms sql server to asp.net
Day 13	06-03-2024	connecting ms sql server to asp.net
Day 14	08-01-2024	set up the development asp.net (Entity Framework EF) and learning course in asp.net(mvc) in Frontend code
Day 15	09-03-2024	Saturday
Day 16	10-03-2024	Sunday
Day 17	11-03-2024	Leave
Day 18	12-03-2024	College First Review
Day 19	14-03-2024	Leave
Day 20	16-01-2024	Saturday
Day 21	17-01-2024	Sunday
Day 22	18-01-2024	Frotend UI Part
Day 23	20-01-2024	Frotend UI Part
Day 24	21-01-2024	Frotend connection to mssql
Day 25	22-01-2024	Frotend connection to mssql
Day 26	23-01-2024	Frotend connection to mssql
Day 27	24-01-2024	Frotend connection to mssql
Day 28	25-01-2024	Creating controler parts and models
Day 29	27-01-2024	Creating controler parts and models and Model view creation
Day 30	28-01-2024	UI+ Backend(Database) in mssql
Day 31	29-01-2024	UI+ Backend(Database) in mssql
Day 32	30-01-2024	UI+ Backend(Database) in mssql
Day 33	31-01-2024	Sunday
Day 34	01-02-2024	Leave
Day 35	02-02-2024	College Second Review
Day 36	03-02-2024	College Second Review

OFFER LETTER



ಭಾರತ್ ಹೆವಿ ಎಲೆಕ್ಟ್ರಿಕಲ್ಸ್ ಲಿಮಿಟೆಡ್
भारत हेवी इलेक्ट्रिकल्स लिमिटेड
Bharat Heavy Electricals Limited

ELECTRIC & PHOTOVOLTAIC DIVISION

Ref: HR – HRD/Ind/Int/2023-24 Date 19.02.2024

To

PRINCIPAL
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY
PORUR, CHENNAI-600 116.

This is to inform that **Mrs. HEMA VARSINI R (Unique ID: E0322010)** studying in your Institution is undergoing Project training at our unit for (6 Weeks). This is issued at the request of the student.

(T. ANITHA)
Sr.MANAGER/HR/HRDC
ಅನಿತಾ ಟಿ. ಹಿರಿಯ ವ್ಯವಸ್ಥಾಪಕರು / ಮಾ ಸಂ
अनिता टी., वरिष्ठ प्रबंधक / भासं
ANITHA T., Sr. Manager / HR
BHEL-SBD, BENGALURU - 560 012

CERTIFICATE OF COMPLETION

ಭಾರತ್ ಹೆವಿ ಎಲೆಕ್ಟ್ರಿಕಲ್ಸ್ ಲಿಮಿಟೆಡ್
भारत हेवी इलेक्ट्रिकल्स लिमिटेड
Bharat Heavy Electricals Limited
(A GOVERNMENT OF INDIA UNDERTAKING)
SOLAR BUSINESS DIVISION
Prof. C.N.R. Rao Circle, Science Institute Post,
Malleswaram, BANGALORE-560012 INDIA

REF: SBD: INT: 23-24/194 **Date: 30.03.2024**

CERTIFICATE

This is to certify that

HEMAVARSINI R E0322010

Pursuing FOURTH SEMESTER **B. TECH IN COMPUTER SCIENCE ENGINEERING (ARTIFICIAL INTELLIGENCE AND DATA ANALYTICS)** at SRI RAMACHANDRA INSTITUTE OF HIGHER EDUCATION & RESEARCH, PORUR, CHENNAI-600116 has undergone Internship Training at BHEL-SBD, Bangalore for 6 weeks i.e, from 19.02.2024 To 30.03.2024

Inputs regarding the various Products and Functions and Production Shop Floor details on power point presentation were provided to the student during the Program.

The interacting session with student during the period of Internship was found good.

Date: 30.03.2024
Place: Bangalore

Signature of Issuing Authority
with Seal
ಅನಿತಾ ಟಿ. ಹಿರಿಯ ವ್ಯವಸ್ಥಾಪಕರು / मा सं
अनिता टी., वरिष्ठ प्रबंधक / मासं
ANITHA T., Sr. Manager / HR
BHEL-SBD, BENGALURU - 560 012