A PROJECT REPORT

ON

“ DIPLOEDGE: A PLATFORM TO GUIDE DIPLOMA ENGINEERING STUDENTS”

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF

DIPLOMA IN COMPUTER ENGINEERING

ELECTRICAL ENGINEERING SECTION

SUBMITTED TO

UNIVERSITY POLYTECHNIC, AMU, ALIGARH

SUBMITTED BY

Name of Student(s)	Enrollment No.
1. Shravan Raj Saraswat	GL9021
2. Vatsal Jain	GM4205
3. Aryan Saxena	GM4120
4. Krishna Yadav	GM4148

GUIDED BY

Mr. S.M. Zakariya
Mr. Naved Qureshi

UNIVERSITY POLYTECHNIC,
FACULTY OF ENGINEERING & TECHNOLOGY,
ALIGARH MUSLIM UNIVERSITY, ALIGARH, INDIA

SESSION:2022-2023

UNIVERSITY POLYTECHNIC, AMU, ALIGARH

CERTIFICATE

This is to Certify that the project report entitled "DIPLOEDGE: A PLATFORM TO GUIDE DIPLOMA ENGINEERING STUDENT**" Was successfully completed by Student of sixth semester Diploma in Computer Engineering.**

1) Shravan Raj Saraswat (… .....)
2) Vatsal Jain (… .....)
3) Aryan Saxena (……..)
4) Krishna Yadav (……..)

in partial fulfillment of the requirements for the award of the Diploma in Computer Engineering (Electrical Engineering Section) and submitted to the Electrical Engineering Section of University Polytechnic, AMU, Aligarh, work carried out during a period for the academic year 2022-23 as per curriculum.

Mr. S.M. Zakariya
Mr. Naved Qureshi

(----------)

Dr. Salim Ishtyaq

(----------)

External Examiner

ACKNOWLEDGMENT

This project is done as a semester project, as a part course titled **"Project and Seminar Lab-II PCO694C".** We are really thankful to our course the **Principal Prof. Arshad Umar**. And the **HOD Ms. Kaushar Jahan, Electrical Engineering Section**, Aligarh Muslim University, University Polytechnic, Aligarh for his invaluable guidance and assistance, without which the accomplishment of the task would have never been possible

We also thank **Mr. S.M. Zakariya & Mr. Naved Qureshi** for giving this opportunity to explore into the real world and realize the interrelation without which a Project can never progress. In our present project we have chosen the topic- **"DIPLOEDGE: A PLATFORM TO GUIDE DIPLOMA ENGINEERING STUDENTS".**

We are also thankful to parents, friend and all staff of **Electrical Engineering department**, for providing us relevant information and necessary clarifications, and great support.

ABSTRACT

This project report is about DiploEdge, a platform designed to provide a one-stop-shop for Diploma in Engineering students to access study materials, job openings, higher studies entrance exam updates and articles to guide them through their studies. The primary aim of the platform is to provide a comprehensive online resource for students to find all the information they need in one place. The report outlines the problem that motivated the creation of DiploEdge, which is the lack of reliable, up-to-date resources for Diploma in Engineering students on the web.

The report details the design and development of DiploEdge, including the user interface, content management system, and database management system. The report also describes the features and functionality of the platform, including the ability to access study materials, job openings, higher studies entrance exam updates, and articles on a wide range of topics. The report highlights the importance of DiploEdge in helping Diploma in Engineering students stay up-to-date with the latest information and have access to a wide range of resources to help them succeed in their academic and professional pursuits.

The report concludes by discussing the potential impact of DiploEdge on the Diploma in Engineering community, including its potential to help students succeed in their studies and advance their careers. The report also highlights future plans for the platform, including the addition of new features and the expansion of its content to cover other areas of interest for Diploma in Engineering students. Overall, the report provides a comprehensive overview of DiploEdge and its potential to make a positive impact on the lives of Diploma in Engineering students.

Table of Contents

List of Figures

List of Abbreviations

Abbreviations	**Description**
JS	JavaScript
CSS	Cascading Style Sheet
HTML	Hypertext Markup Language
EJS	Embedded JavaScript
SQL	Structured Query Language
NoSQL	Not Structured Query Language

CHAPTER 1: INTRODUCTION

1.1. Aim & Objective

The aim of our project is to create DiploEdge, a comprehensive platform for Diploma in Engineering students to access study materials, job openings, higher studies entrance exam updates, and articles to guide them through their studies. Our objective is to provide a one-stop-shop for students to find all the information they need in one place. The platform is designed to be user-friendly and easy to navigate, with all the information organized in a logical and intuitive manner.

DiploEdge was developed in response to the lack of reliable, up-to-date resources for Diploma in Engineering students on the web. We recognized that many students struggle to find the information they need to succeed in their studies and careers, and we wanted to create a platform that would make it easier for them to access this information.

To achieve our aim, we conducted extensive research to identify the specific needs and challenges faced by Diploma in Engineering students. We then worked to design and develop a platform that would address these needs and provide a comprehensive online resource for students.

1.2. Purpose of the Project

Diploma in Engineering is a popular course of study that attracts many students in different parts of the world. Despite the vastness and complexity of the course, the resources available online for students are limited and often outdated. This scarcity of reliable information poses a significant challenge for students seeking to excel in their academic and professional pursuits. It is against this backdrop that DiploEdge was created, a web platform that provides a comprehensive online resource for Diploma in Engineering students to find all the information they need in one place.

The purpose of DiploEdge is to bridge the gap between the limited resources available for Diploma in Engineering students and the vast amount of information available on the web. The platform aims to provide students with easy access to

study materials, job openings, higher studies entrance exam updates, and articles to guide them through their studies. DiploEdge seeks to provide a single point of access for students to stay informed and updated with the latest developments in their field.

The need for such a platform cannot be overemphasized. Diploma in Engineering students often face significant challenges in accessing reliable and up-to-date resources, which can hinder their academic and professional growth. DiploEdge is designed to fill this gap by providing students with a wealth of information and resources that they can access at any time, from any location. The platform ensures that students have the necessary tools to succeed in their academic and professional pursuits.

In conclusion, DiploEdge is a necessary platform that provides Diploma in Engineering students with comprehensive resources and guidance to excel in their studies and professional endeavors. The scarcity of reliable resources for Diploma in Engineering students necessitates the creation of such a platform to provide students with access to the latest information and help them stay informed and updated with the latest developments in their field. With DiploEdge, students can stay on top of their studies, achieve their academic goals, and advance their careers.

1.3. Scope of Project

The scope of DiploEdge is vast and includes various features and functionalities that cater to the needs of Diploma in Engineering students. The platform provides access to study materials, job openings, higher studies entrance exam updates, and articles to guide students through their studies.

One of the primary features of DiploEdge is the provision of comprehensive study materials. The platform provides students with access to a wide range of resources, including notes, textbooks, and reference materials. These resources are carefully curated to ensure that they are relevant and up-to-date, and cover all the essential

topics in Diploma in Engineering.

Another critical aspect of DiploEdge is the provision of job openings for Diploma in Engineering students. The platform features a job board where students can search for relevant job opportunities in their field of study. This feature helps students identify job opportunities that are aligned with their skills and interests and helps them transition smoothly into their professional careers.

DiploEdge also provides updates on higher studies entrance exams, which is essential for students who want to pursue higher education in their field. The platform provides students with information on the latest entrance exams, application deadlines, and study materials to help them prepare for these exams.

Additionally, DiploEdge features articles that provide guidance and insights to Diploma in Engineering students. These articles cover a wide range of topics, including career advice, exam preparation, study techniques, and industry trends. These articles help students gain a better understanding of the field, and equip them with the knowledge and skills needed to succeed.

Overall, the scope of DiploEdge is to provide Diploma in Engineering students with a comprehensive online resource and guidance platform that caters to their needs. The platform aims to help students stay up-to-date with the latest information and have access to a wide range of resources to help them succeed in their academic and professional pursuits.

CHAPTER 2: OVERALL DESCRIPTION

2.1. User Classes and Characteristics

The website provides different types of services based on the type of users. Here users are accessing the website as an admin or user.
The features that are available to the Users are:

- Navigate through the Home page.
- Visit Notes page and navigate through each division.
- User can visit Articles page where all the articles are listed.
- User can also visit each listed article on Articles page.
- User can visit Notices page where all notices are listed.

The features that are available to the Admins after login are:

- Navigate through the Home page.
- Visit Notes page and navigate through each division.
- User can visit Articles page where all the articles are listed.
- User can also visit each listed article on Articles page.
- User can visit Notices page where all notices are listed.
- Visit Compose page.
- Admin can compose articles, notes and notices through compose page.

2.1.1. Operating Environment

The DiploEdge is a web application and can be operated through browser in web pages. The only requirement to use this website would be the internet connection and a device which can access high speed internet.

2.1.2. Assumptions and Dependencies

The assumptions are:

- The coding should be error free.
- The application should be user-friendly so that it is easy to use for the users.
- Valid information must be stored in database that is accessible by the website.
- The system should provide more storage capacity and provide fast access to the database.
- The DiploEdge is running 24 hours a day.
- Users may access from any browser that has Internet browsing capabilities and an Internet connection.
- Admins must have their correct usernames and passwords to enter into their accounts and do actions.

The dependencies are:

- The specific hardware and software due to which the product will be run.
- On the basis of listing requirements and specification the project will be developed and run.
- The end users should have proper understanding of the product.
- The information of all the must be stored in a database that is accessible by the system.
- Any update regarding the articles, notes, notices and any other update is to be recorded to the database and the data entered should be correct.

2.2. Requirements

- Software Configuration:

This website is developed using HTML, CSS, Bootstrap, EJS and Javascript as front end, Node js (Express) as server side validation, MongoDB as to store the database and Mongod as web server and Mongoose used as an interface between Nodejs and MongoDB.

- Operating System: Windows XP and later versions of Windows, ChromeOS and MacOS

- Front end: HTML, CSS, Bootstrap, Javascript, EJS
- Client side Validation: Nodejs
- Server side Validation: Node js
- Business Logic: Node js
- Database: MongoDB
- Web Server: Mongod
- Hardware Configuration:
    - Processor: Core i3, 1.5MHz
    - Hard Disk: 150 GB
    - RAM: 2GB
    - Resolution: 480 X 800

2.2.1. Data Requirement

The inputs consist of the query to the database and the output consists of the solutions for the query. In this project the inputs will be queries as fired by the admins like create a post. Now the output will be visible when the user requests the server to get details of that post and also notices, downloads posted by admins.

2.2.2. External Interface Requirement GUI

This application provides good graphical interface for the user and the administrator can operate on the system, performing the request task such as create, update, delete and also view the every detail.

- The user interface must be customizable by the administrator.
- All the modules provided with the software must fit into this graphical user interface and accomplish to the standard defined.
- The design should be simple and all the different interfaces should follow a standard template.
- The user interface should be able to read the content by the admins.
- Design of login page should be easy to understand for every admin.

2.2.3. System Features

This is possible by providing:-

- User accessibility available to all the visitors..
- Every admin can have individual & unique log in credential.
- This website provides separate access to admin (viz. Compose page to post articles, notes and notices)
- Admin can post including photo & include any other kind of file.
- Admin functionality will be added

2.2.4. Performance Requirement

- The performance of the system should be fast and accurate.
- It shall handle expected and nonexpected errors in way that prevent loss in information and long downtime period.
- The system should be able to handle large amount of data. Thus it should accommodate large number of data entry of a articles, notes and notices without any fault.

2.2.5. Satisfy Requirement

The database may get crashed at any certain time due to virus or operating system failure. Therefore, it is required take the database backup.

2.2.6. Security Requirements

- System will use secured database.
- Normal users can just read the content but they cannot edit or modify anything.
- System will have two type of user and user has access constraints.
- Proper admin authentication should be provided.
- Admin has the rights to update the database.

2.2.7. User Requirement

The users are assumed to have basic knowledge of the computers and internet browsing. The administrators of the system should have more knowledge of the internals of the system and is able to rectify the small problems that may arise due to disk crashes, power failures and other catastrophes to maintain the system.

2.3. System Planning and System Development Lifecycle

We have chosen "Iterative Life Cycle Model" for developing this application, because an iterative life cycle model does not attempt to start with a full specification pf requirements. Instead, development begins by specifying and implementing just part of the software, which can then be reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software for each cycle of the model. Here is the diagram of the iterative life cycle model which depicts its working flow.[1]

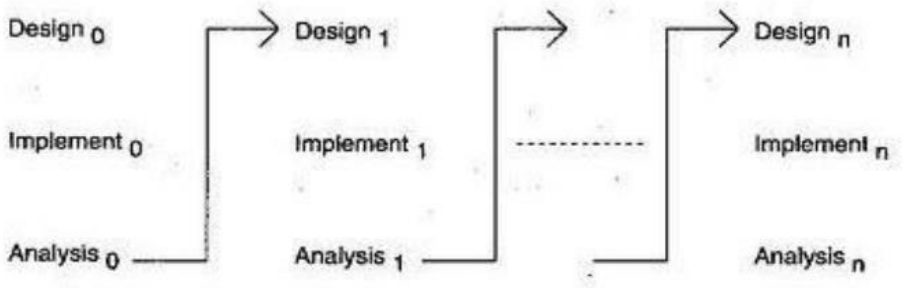

Figure 2.1: Iterative Enhancement Model[1]

Few advantages for choosing the SDLC are –

- In iterative model we are building and improving the product step by step. Hence, we can track the defects at early stages. This avoids the downward flow of the defects.
- Testing and debugging in smaller iteration is easy.

- In iteration model we can get the reliable user feedback. When presenting sketches and blueprints of the product to user for their feedback, we are effectively asking them to imagine how the product will work.
- Progress can be measured.
- In iterative model less time is spent on documentation and more time is given for designing.
- Risk are identified and resolved during iteration; and each iteration is an easily managed milestone. It supports changing requirement.[1]

2.4. Constraints of Use

- Admin and user must remember login id and password.
- User need to have a personal computer or cell phone with internet connection.

CHAPTER 3: TECHNOLOGY USED

3.1. Front End

3.1.1. HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.[2]

3.1.2. CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content. [3]

3.1.3. Bootstrap

Bootstrap is a free and open-source front-end library for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.[4]

3.1.4. JavaScript

JavaScript is a scripting or programming language that allows you to implement complex features on web pages — every time a web page does more than just sit there and display static information for you to look at — displaying timely content updates, interactive maps, animated 2D/3D graphics, scrolling video jukeboxes, etc.[5]

3.1.5 EJS

EJS (Embedded JavaScript Templating) is one of the most popular template engines for JavaScript. As the name suggests, it lets us embed JavaScript code in a template language that is then used to generate HTML.

Aa template engine is software designed to combine templates with a data model to produce, in our case, real HTML code. Template engines handle the task of interpolating data into HTML code while providing some features (like partials in EJS) that would have been difficult to replicate by concatenating strings.[6]

Another key feature of EJS is its flexibility. It can be used with various web frameworks, including Express and Koa, and can be integrated with other Node.js modules to create a complete web application. EJS also supports a range of options, such as caching and layout support, allowing developers to customize the behavior and performance of their templates.

EJS also has excellent support for data binding, allowing developers to pass data from the server to the client-side of their application. This makes it easy to create dynamic and interactive web pages that respond to user input and display real-time data.[7]

3.2. Client-Side Validation

JavaScript often abbreviated as JS, is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm. Alongside HTML and CSS, JavaScript is one of the three core technologies of the World Wide Web. JavaScript enables interactive web pages and thus is an essential part of web applications. The vast majority of websites use it, and all major web browsers have a dedicated JavaScript engine to execute it. JQuery is a cross-platform JavaScript library designed to simplify the clientside scripting of HTML. It is free, open-source software using the permissive MIT License. Web analysis indicates that it is the most widely deployed JavaScript library by a large margin.[8]

3.3. Server Side Validation

Node.js is a runtime environment to allow JavaScript to not only be run in the browser, but also on the server (or almost any environment, really). That also expanded the types of applications that could be built with the language since it wasn't tied to only the client-side anymore. After that, the popularity of the language exploded. Node is available on Windows, Linux, Mac, and many other platforms. Developers started using the power of the platform, which allowed them to access APIs such as the filesystem and to use powerful applications. Popular projects have been created to ease the development of software, web, desktop, mobile, and other types of applications. Thanks to Node, Chromium, and Electron we can now have sophisticated IDEs like Visual Studio Code, programming languages like TypeScript, powerful command-line applications like React Native, and Expo to create mobile apps with React. This is all possible because JavaScript is now available everywhere.

We used the Express framework of Nodejs. Express.js is a free and open-source web application framework for Node.js. It is used for designing and

building web applications quickly and easily. Web applications are web apps that you can run on a web browser. Since Express.js only requires javascript, it becomes easier for programmers and developers to build web applications and API without any effort. Express.js is a framework of Node.js which means that most of the code is already written for programmers to work with. You can build a single page, multi-page, or hybrid web applications using Express.js. Express.js is lightweight and helps to organize web applications on the server-side into a more organized MVC architecture.

It is important to learn javascript and HTML to be able to use Express.js. Express.js makes it easier to manage web applications.It is a part of a javascript based technology called MEAN software stack which stands for MongoDB, ExpressJS, AngularJS, and Node.js. Express.js is the backend part of MEAN and manages routing, sessions, HTTP requests, error handling, etc. The JavaScript library of Express.js helps the programmers to build efficient and fast web apps. Express.js enhances the functionality of the node.js. In fact, if you don’t use Express.js, then you have to do a lot of complex programming to build an efficient API. It has made programming in node.js effortless and has given many additional features.[9]

3.4. Database MongoDB

NoSQL databases are non relational database management systems. It does not mean that we are saying no to SQL. It means that apart fromm SQL, there could be other solutions too that are relevant in our applications that we use. Hence, the NoSQL can be interpreted as ‘Not Only SQL’. Now, let us see the architeture of NoSQL and how its been evolving over the years: In the 1980’s: All the applications used the same database. For example, considering the database MySQL, all the applications that were being developed used MySQL but each used it individually. In the 1990’s: Database was used as an integration hub where a group of

applications all used the same database. In the 2000's and hereafter: Each application had its own database, and each database was different from the other.[10] NoSQL databases differ from the traditional NoSQL systems in many ways:

• NoSQL databases are unstructured unorganized and have unpredictable data, whereas SQL databases have structured and organized data.

• NoSQL can be classified further as key value/tuple store, column store, graph db, document store, whereas SQL databases can be classified as DDL, DML.

• NoSQL systems do not have any predefined schema, whereas SQL databases do.

• NoSQL systems relax one or more of the ACID properties, whereas SQL systems strictly follow the ACID properties Some other important characteristics of NoSQL databases are:

• Next generation databases

• Open source

• Distributed

• Large data volumes

• Non relational

• Scalable replication and distribution.

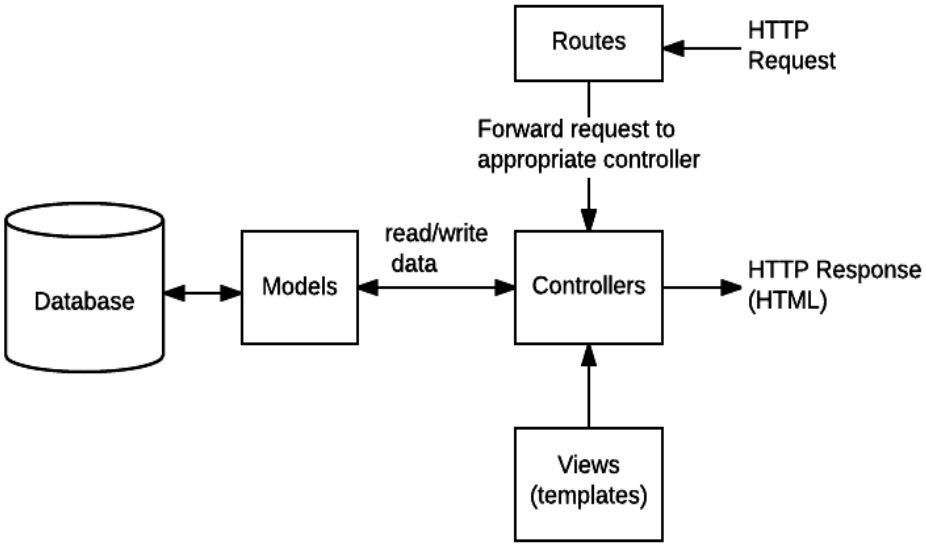

Figure3.1: Routes and Controllers[10]

MongoDB is a popular NoSQL document-oriented database that stores data in flexible, semi-structured JSON-like documents. Unlike traditional relational databases, MongoDB does not require a predefined schema, allowing for more dynamic and adaptable data structures.

MongoDB is often used in web applications and big data projects, where fast read and write operations are required. Its flexible document model allows for faster and more efficient querying, making it a good fit for real-time analytics, content management systems, and e-commerce platforms.

MongoDB also has a rich set of features, including horizontal scaling for high availability and automatic sharding, which helps distribute data across multiple servers to improve performance and ensure redundancy. It also supports a range of programming languages, including Python, Java, and Node.js, making it easy to integrate with other tools and frameworks.

Overall, MongoDB is a powerful and flexible database solution that can be used in a variety of applications and use cases. However, like any technology, it has its own strengths and weaknesses, and choosing the right database solution depends on the specific needs of the project.

MongoDB is a document-oriented NoSQL database. It has steadily been gaining popularity due to its great functionality and ease of use. It is currently ranked as the top choice for NoSQL databases according to most rankings including DB-Engines and KDNuggets.

MongoDB document is an ordered collection of key-value pairs where the keys are strings and values can be objects like integers, 6 strings or other documents. MongoDB represents its data as JSON (JavaScript Object Notation) documents.

Figure 1.2: JSON Objects for storing dependencies

A group of related MongoDB documents is called a collection. When a document is embedded or nested within another document, it is called an Embedded document. It is particularly useful to denormalize data by storing multiple related documents in a single document as it improves query performance.

Some other important terminologies associated with MongoDB are:

• mongod: mongod is the main process of MongoDB that runs in the background and performs all majors tasks including management and handling requests. A mongod runs on each node running MongoDB.

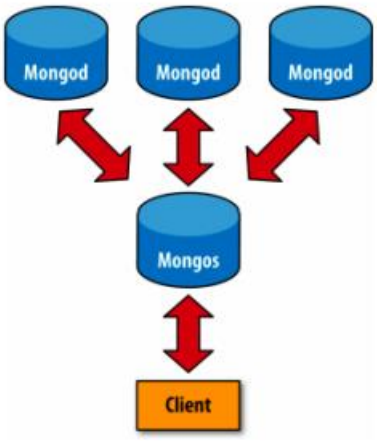

Figure 3.3: Sharded MongoDB Cluster[10]

Figure 3.4: Non-sharded MongoDB[10]

• mongos: MongoDB is designed to run on a cluster of multiple nodes through the concept of sharding. Data is sharded and distributed to different nodes in 7 the cluster. mongos provides a layer of abstraction between the user and the sharded cluster by acting as an interface between them. When writes and reads are performed on data in a MongoDB cluster, mongos routes them based on the sharding information.

• config server: All metadata regarding the sharded cluster is stored in the config server. This includes information such as the distribution of data across shards

3.5. Web Server Mongod

MongoDB's "mongod" server is the primary component of the MongoDB database system. It is responsible for managing data storage, retrieval, and processing of queries.

The "mongod" server is designed to be highly scalable and distributed, making it ideal for handling large datasets and high-traffic applications. It supports a number of advanced features like sharding and replication, which allow for horizontal scaling and high availability.

The server stores data in the form of BSON documents, which are similar to JSON objects but are encoded using a binary format. BSON is used to optimize data storage and retrieval operations, making MongoDB faster and more efficient than traditional relational databases in many cases.

In addition to data management, the "mongod" server provides a powerful query language called MongoDB Query Language (MQL). MQL allows developers to perform complex queries and aggregations on data, and supports a wide range of operators and functions for filtering and manipulating data.

The "mongod" server can be run on a variety of platforms, including Windows, macOS, and Linux. MongoDB also provides drivers and tools for many programming languages, including Python, Java, and Node.js, making it easy to integrate with other applications and systems.[11]

3.6. Creating Database connection

Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js. It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.

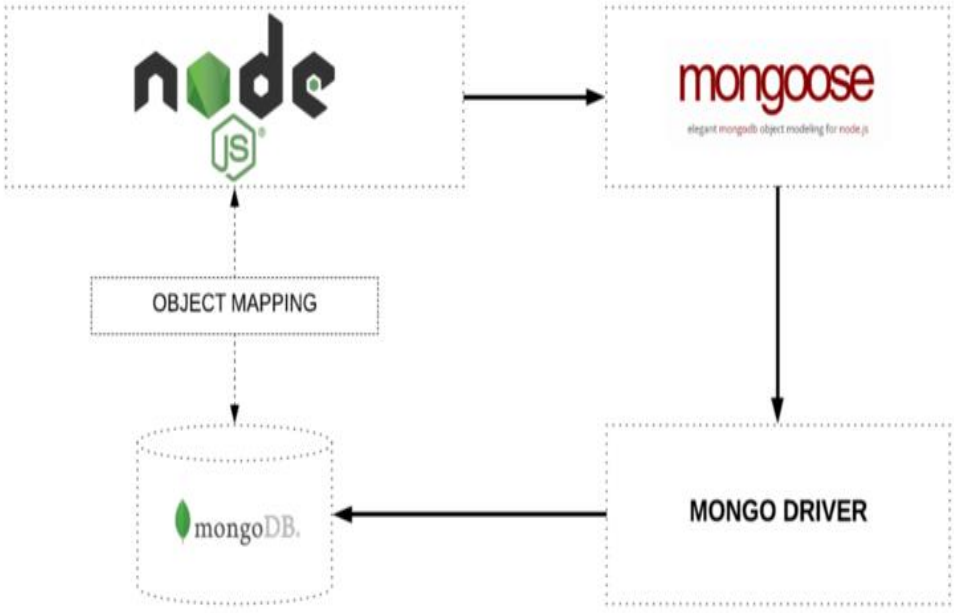

Figure 3.5: Object Mapping between Node and MongoDB managed via Mongoose[12]

Object Mapping between Node and MongoDB managed via Mongoose MongoDB is a schema-less NoSQL document database. It means you can store JSON documents in it, and the structure of these documents can vary as it is not enforced like SQL databases. This is one of the advantages of using NoSQL as it speeds up application development and reduces the complexity of deployments.[12]

3.7. Use Case Diagram

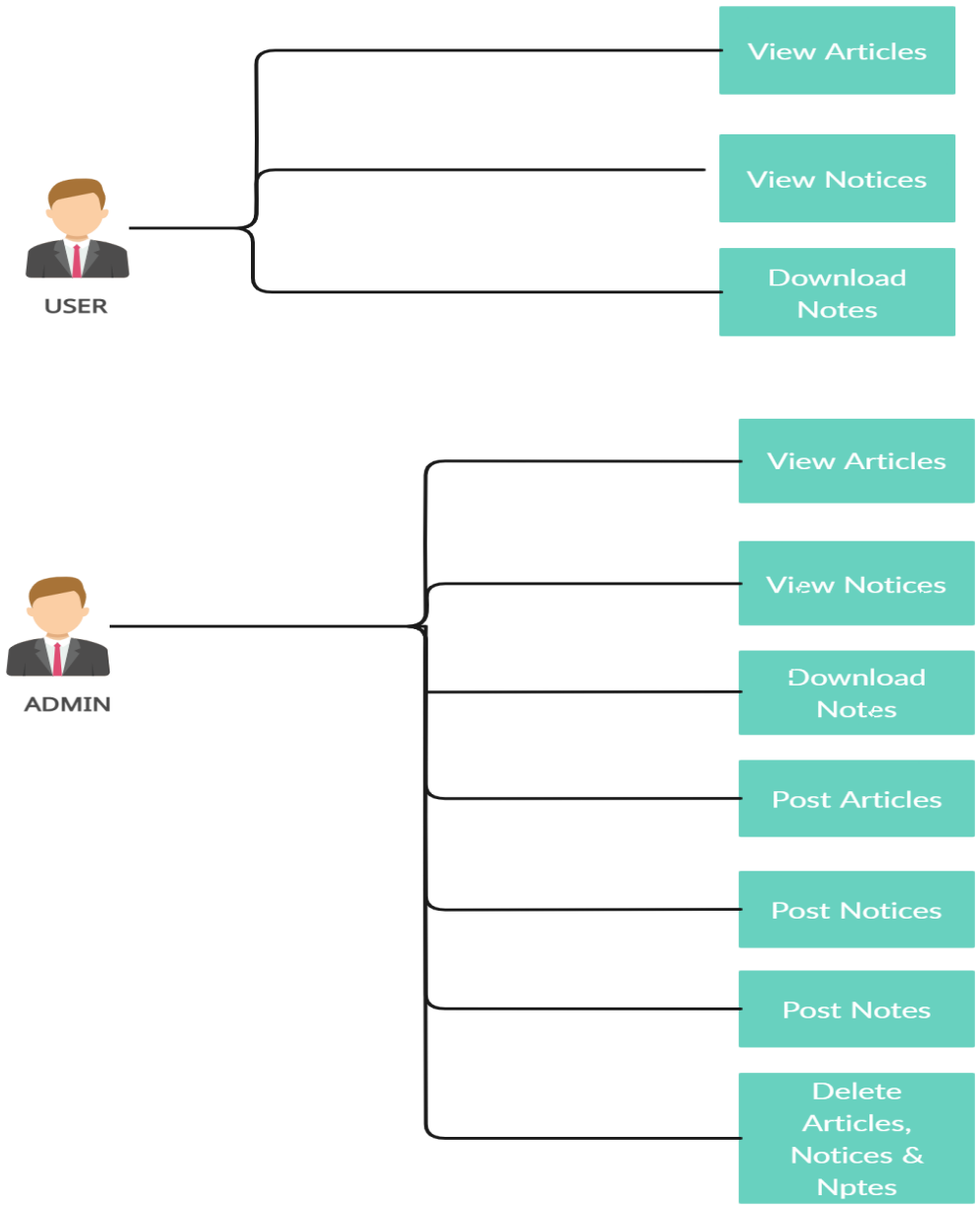

Figure 3.6: Use Case Diagram

3.8. Source Code

3.8.1. Creating Database connection using Mongoose

```
1   //jshint esversion:6
2
3   const express = require("express");
4   const bodyParser = require("body-parser");
5   const ejs = require("ejs");
6   const _=require("lodash");
7   const mongoose = require("mongoose");
8
9
10  //=================================Data Base development=================
11
12  mongoose.connect("mongodb://127.0.0.1:27017/mainDB");//to solve buffer tir
    localhost,mainDB is the name of main Data base
13
14
```

Figure 3.7: Creating Database connection using Mongoose

The require('mongoose') call above returns a Singleton object. It means that the first time you call require('mongoose'), it is creating an instance of the Mongoose class and returning it. On subsequent calls, it will return the same instance that was created and returned to you the first time because of how module import/export works in ES6.

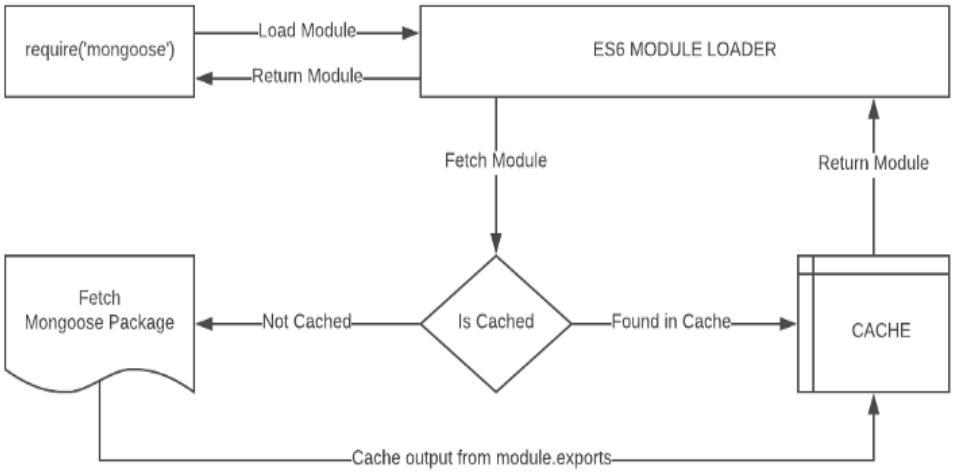

Figure 3.8: Module import/require work-flow[13]

Similarly, we have turned our Database class into a singleton by returning an instance of the class in the module.exports statement because we only need a single connection to the database.

ES6 makes it very easy for us to create a singleton (single instance) pattern because of how the module loader works by caching the response of a previously imported file.

3.8.2. Defining Schema

```
16  //==================defining schema for all the miscellaneous collections
17
18  const downloadSchema =new mongoose.Schema({
19    contentTitle: String,
20    contentLink:String,
21    contentBranch:String,
22    contentSem:Number
23  });
24
25  const noticeSchema ={
26    noticeTitle: String,
27    noticeContent:String,
28    noticeLink:String
29  };
30
31  const articleSchema =new mongoose.Schema({
32    articleTitle: String,
33    articleContent:String,
34    articleLink:String,
35    articleImg:String
36  });
37
```

Figure 3.9: Defining Schema

While Mongo is schema-less, SQL defines a schema via the table definition. A Mongoose schema is a document data structure (or shape of the document) that is enforced via the application layer.

A schema defines document properties through an object where the key name corresponds to the property name in the collection.

3.8.3. Post, Compose and View Routes

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


  const download = new Download({
    contentTitle: req.body.Title,
    contentLink:req.body.Link,
    contentBranch:req.body.Branch,
    contentSem:req.body.Semester
  });
  Download.insertMany([download]);

  res.redirect("/downloads");
})
```

Figure 3.10: Post route for Download

```
app.get("/downloads",function(req,res)                    //view download routes
{
  Download.find({},function(err,downloadarray){
    if(err){
      console.log(err);
    }
    else{
      res.render("download",{downloadarray:downloadarray});
    }
  })

})
```

Figure 3.11: View Route for Download

```
app.get("/compose",function(req,res){        //view compose route
  res.render("compose");
})
```

Figure 3.12: Compose Route for Download

Other Schemas follow the similar structure except a complex parameter route has been defined for articles.

```
//======================complex parameter route for articles=================

app.get("/post/:articleTitle",function(req,res) // getting value in parameter
{

    const requestedTitle = _.lowerCase(req.params.articleTitle);
    console.log(requestedTitle);

    Article.find({},function(err,articlearray){
      if(err){
        console.log(err);
      }
      else{
        articlearray.forEach(function(article){
          console.log(article);
          if(_.lowerCase(article.articleTitle)==requestedTitle){
            res.render("post",{
              articleTitle:article.articleTitle,
              articleContent:article.articleContent,
              articleImg:article.articleImg,
              articleLink:article.articleLink
            })

          }

        });
      }
    })
})
```

Figure 3.13: Complex parameter route for articles

3.8.4. Display of Articles page, Notices page and individual notices and articles

views > article.ejs > ? > ? > ? > ?

```
<%- include("partials/header") -%>
<div id="article-notice-page"></div>

<% posts.forEach(function(post){ %>
        <h1 class="article-notice-page-title"><%= post.articleTitle %></h1>
        <p> <%= post.articleContent.substring(0,100)+"..." %>
        <a href="post/:<%= post.articleTitle %>"> click me</a>
        </p>
        <hr>
    <% }) %>
<%- include("partials/footer") -%>
```

Figure 3.14: EJS for displaying the article page

```
<%- include("partials/header") -%>
<div id="article-notice-page"></div>

<% posts.forEach(function(post){ %>
    <h1 class="article-notice-page-title"><%= post.noticeTitle %></h1>
    <p> <%= post.noticeContent %>
    <a href="post/:<%= post.noticeLink %>"> Read me</a>
    </p>
    <hr>
    <% }) %>
<%- include("partials/footer") -%>
```

Figure 3.15: EJS for displaying the notice page

views > post.ejs > ?

```
<%- include("partials/header") -%>
    <h1 class="post-title"><%= articleTitle %></h1>
    <p class="post-content"> <%= articleContent %></p>
    <img src="<%= articleImg %>">
    <button class="post-button" href="<%= articleLink %>"> Download </button>

<!--Footer Removed--->
```

Figure 3.16: EJS for displaying the individual article page

3.8.5. Compose Form

```
<%- include("partials/header"); -%>
    <form class="article" action="/article"  method="post" autocomplete="off">
        <div class="form-group">
            <h1>Article</h1>
            <label>Title</label>
            <input type="text" name="Title" class="form-control">
            <label>content</label>
            <textarea name="Content" rows="5" collon="50" class="form-control"></textarea>
            <label>link</label>
            <input type="text" name="Link" class="form-control">
            <label>img link</label>
            <input type="text" name="imgLink" class="form-control">
            <button class="btn btn-primary" type="submit" name="button">Add</button>
        </div>
    </form>

    <form class="notice" action="/notices"  method="post" autocomplete="off">
        <div class="form-group">
            <h1>Notice</h1>
            <label>Title</label>
            <input type="text" name="Title" class="form-control">
            <label>content</label>
            <textarea name="Content" rows="5" collon="50" class="form-control"></textarea>
            <label>link</label>
            <input type="text" name="Link" class="form-control">
            <button class="btn btn-primary" type="submit" name="button">Add</button>
        </div>
    </form>

    <form class="download" action="/download"  method="post" autocomplete="off">
        <div class="form-group">
            <h1>Downloads</h1>
            <label>Title</label>
            <input type="text" name="Title" class="form-control">
            <label>Branch</label>
            <input type="text" name="Branch" class="form-control">
            <label>Semester</label>
            <input type="number" name="Semester" class="form-control">
            <label>link</label>
            <input type="text" name="Link" class="form-control">
            <button class="btn btn-primary" type="submit" name="button">Add</button>
        </div>
    </form>

<%- include("partials/footer"); -%>
```

Figure 3.17: Compose Form source code

3.8.6. Screenshots of the Web Application

3.8.6.1. Home Page

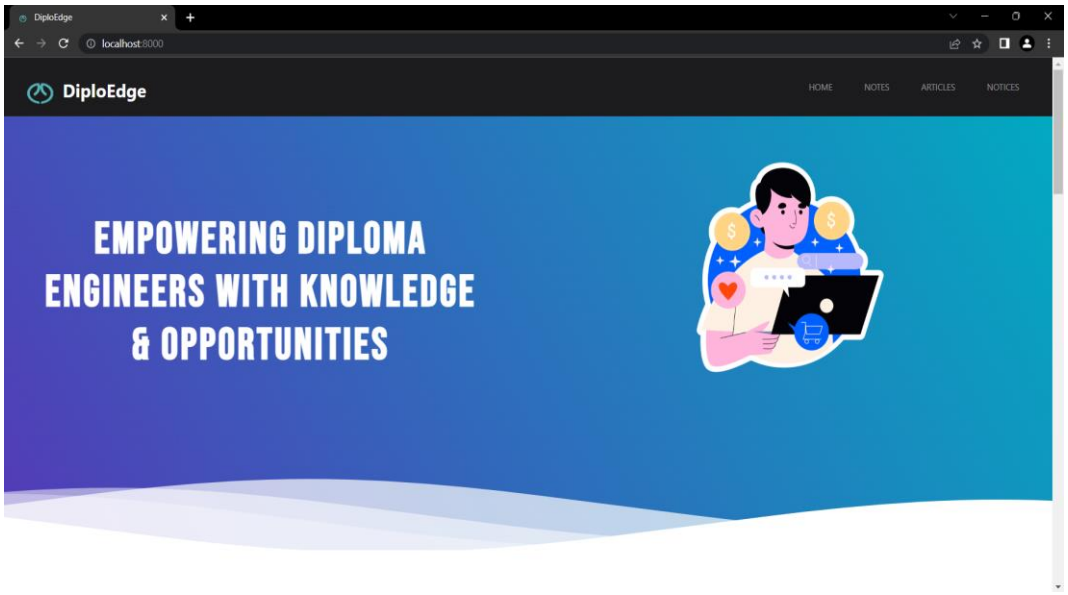

Figure 3.18: Home Page (A)

In the above image of Main Home page it can be seen that it consists of functional navbar containing navigational links to Notes Page, Articles Page and Notices Page. It also consists of multiple animations applied to the image and the background.
The parallax effect is a technique used in web design to create a sense of depth and movement on a webpage. It involves layering multiple images or elements on top of each other and animating them at different speeds, creating an illusion of depth and movement as the user scrolls through the page. In this specific example, the parallax effect is achieved by applying an animation to the "use" elements within the "parallax" container. The animation moves the elements horizontally using the "transform" property and the "translate3d" function. The "animation-delay" and "animation-duration" properties are used to stagger the animation of each "use" element, creating a more complex and dynamic parallax effect. This technique can be used in various web design projects, such as creating dynamic landing pages, product showcases, or visual storytelling elements.

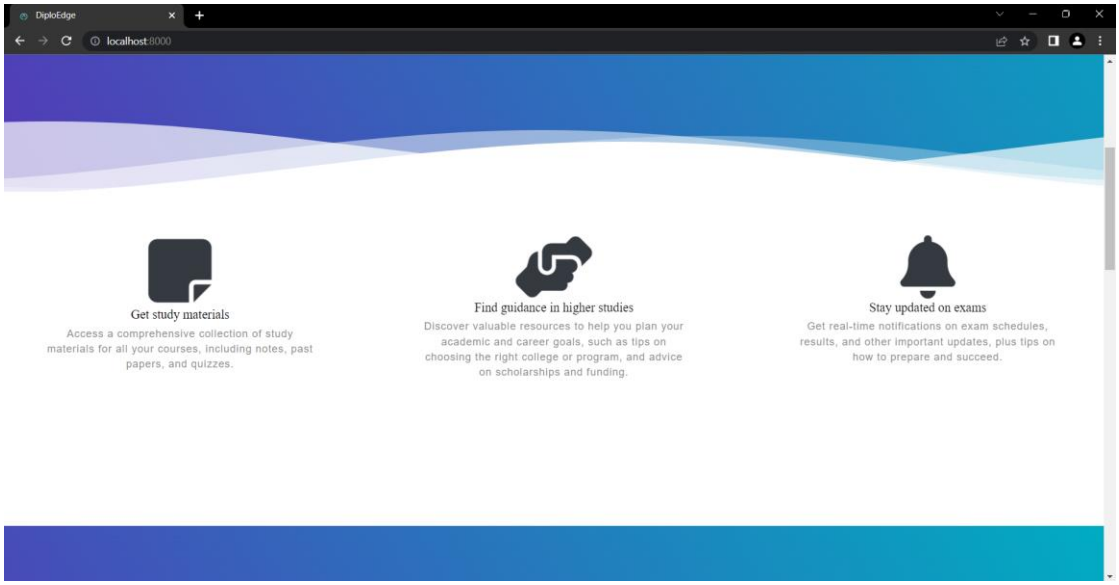

Figure 3.19: Home Page (B)

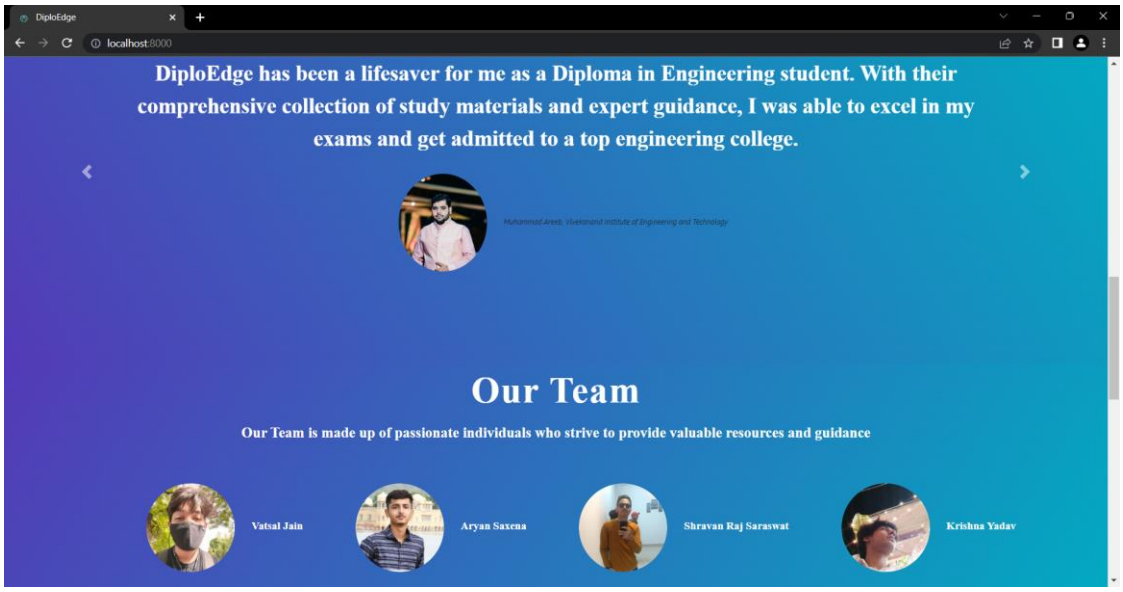

Figure 3.20: Home Page (C)

In Fig. 3.20 can be seen that a carousel is used for displaying testimonials. Bootstrap Carousel is a popular UI component provided by the Bootstrap framework, which is widely used in web development for creating responsive, mobile-friendly, and interactive carousels. It is an easy-to-use and customizable solution that allows developers to create stunning visual experiences for their web pages without writing extensive code.

The Bootstrap Carousel supports a variety of content types, including images, videos, text, and HTML elements. It is built on top of the standard HTML carousel functionality, allowing developers to quickly create a fully functional carousel with just a few lines of code.

The Bootstrap Carousel comes with a range of pre-built features, including autoplay, pause on hover, indicators, and navigation controls, making it easy for users to interact with the carousel. Additionally, the carousel supports multiple transitions, including slide, fade, and zoom, enabling developers to create unique and engaging effects for their content.

One of the most significant advantages of the Bootstrap Carousel is its responsiveness. It is designed to work seamlessly on various screen sizes, making it an ideal solution for creating mobile-first web applications. The carousel is also highly customizable, allowing developers to tweak the styling, layout, and functionality to suit their specific needs.

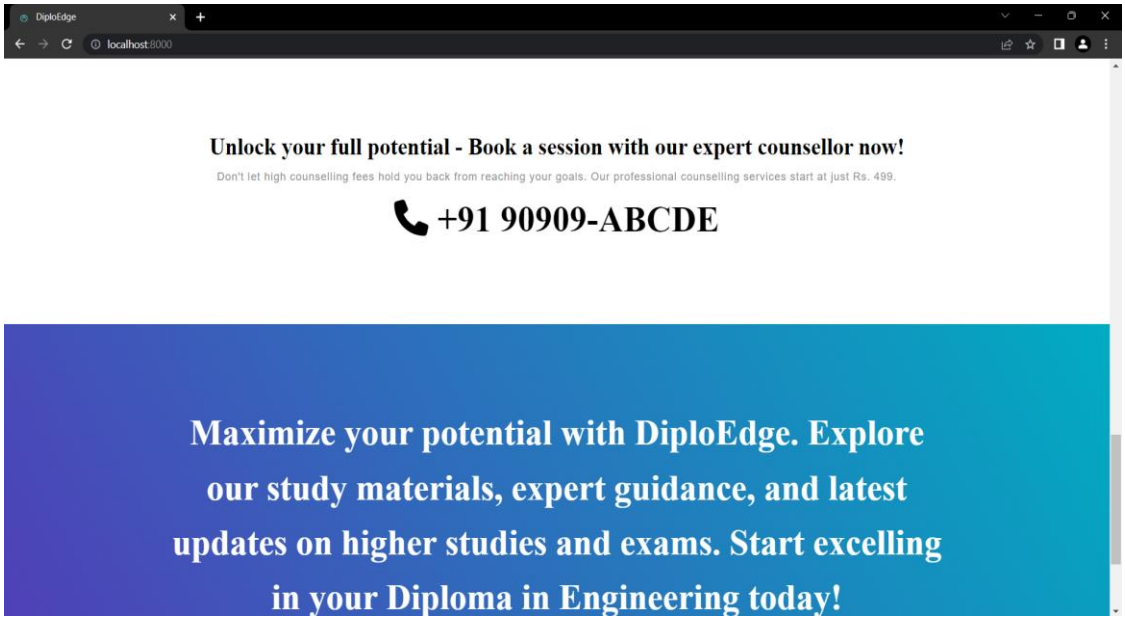

Figure 3.21: Home Page (D)

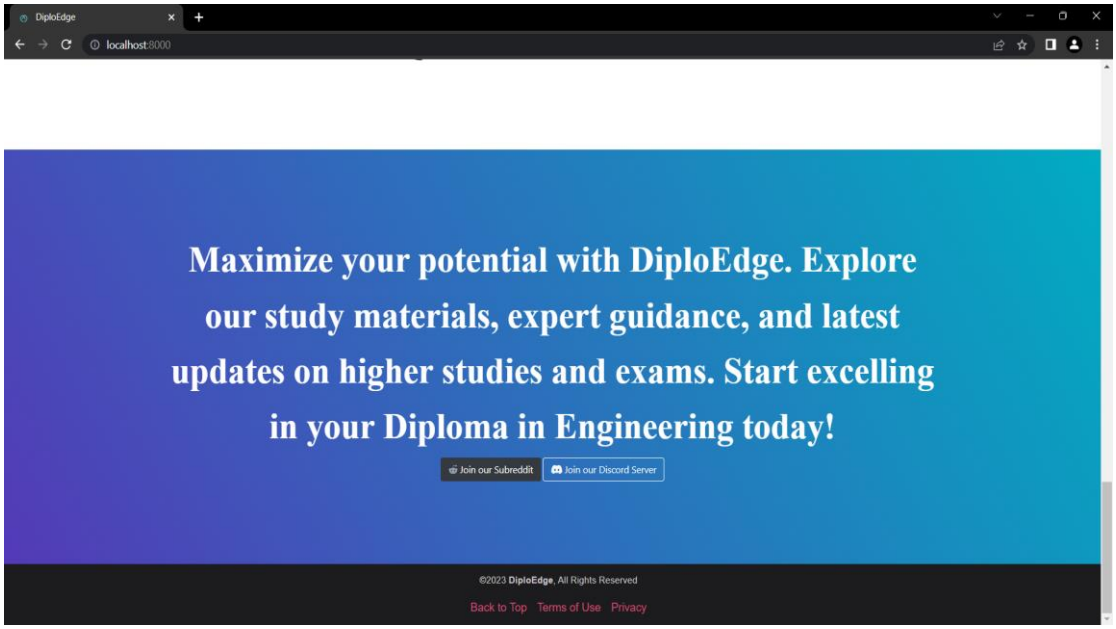

Figure 3.22: Home Page (E)

Various animations have been applied to icons and buttons using the features of fontawesome.com. Fontawesome.com is a popular icon library that offers a wide range of scalable vector icons that can be customized and styled to suit your website's needs. One of the exciting features of Fontawesome is the ability to add animations to icons and buttons. These animations can bring a dynamic and engaging element to your website's design, making it more interactive and visually appealing.

By using Fontawesome's built-in animation classes, you can easily add animations such as spin, pulse, flip, and many more to your icons and buttons. These classes are designed to work seamlessly with Fontawesome icons and can be applied to any element on your page, giving you complete control over the animation's timing and style.

Whether you are creating a landing page, a portfolio site, or an online store, Fontawesome's animation classes can help you add an extra layer of creativity and interactivity to your design. With the ability to customize the animation duration, delay, and easing, you can create unique and engaging animations that bring your website to life.

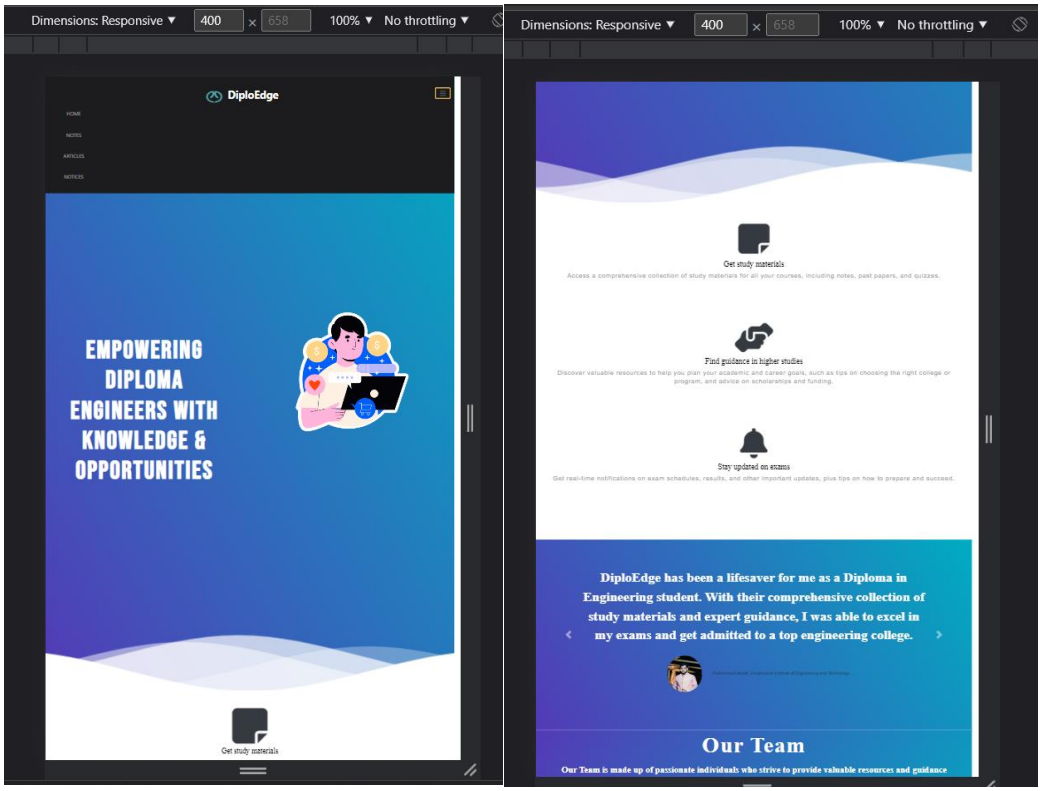

Figure 3.23: Responsive Home Pages (A)

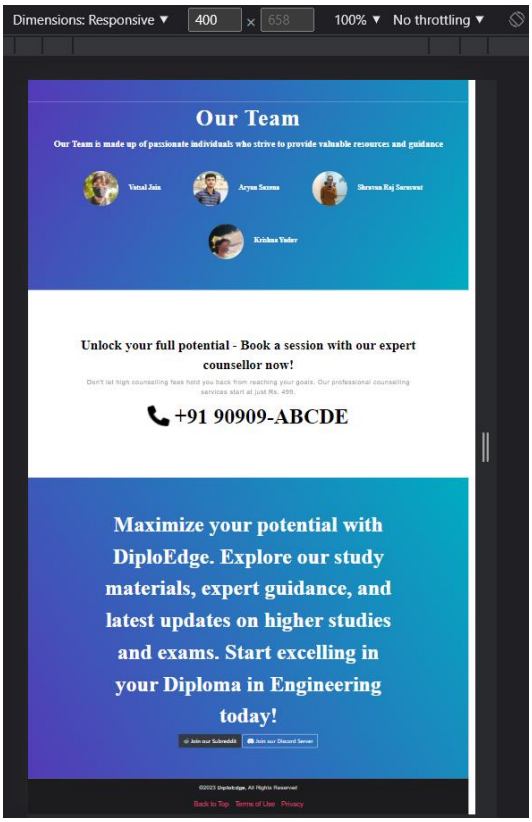

Figure 3.24: Responsive Home Pages (B)

Bootstrap was an essential tool in creating our website. We used it to ensure that our website has a consistent and responsive layout across all devices and screen sizes. Bootstrap's breakpoint system allowed us to define the minimum and maximum widths at which certain CSS styles should be applied. This enabled our website to adjust its layout and design based on the device it's being viewed on, whether it's a desktop computer, tablet, or mobile phone.

In addition to breakpoints, Bootstrap provided us with a range of pre-built UI components that we could easily customize and integrate into our website. These components included navigation menus, forms, buttons, modals, and much more, saving us a lot of time and effort in building these elements from scratch.

3.8.6.2. Notes Page

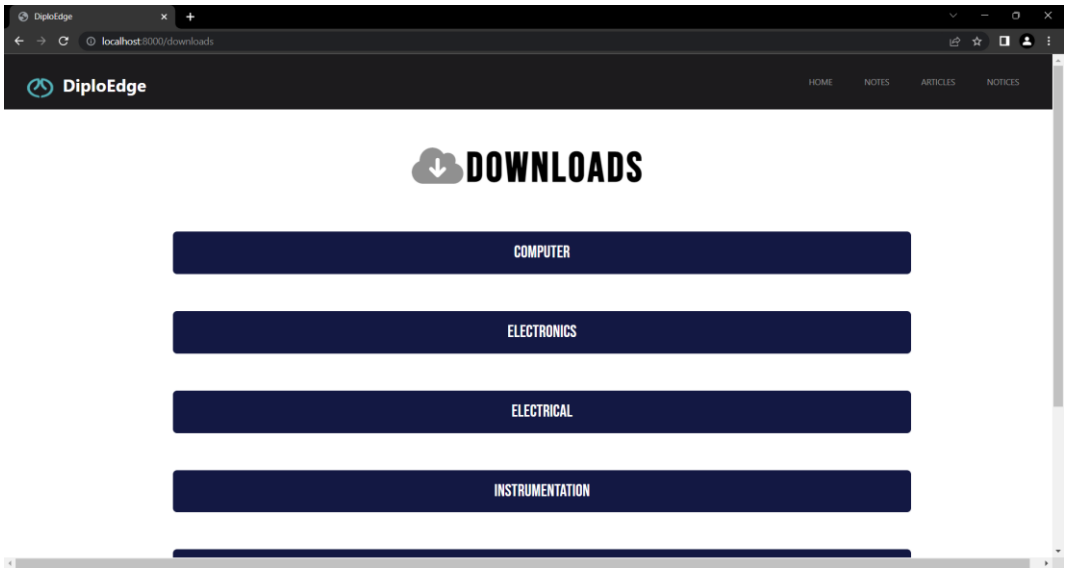

Figure 3.25: Notes Page (A)

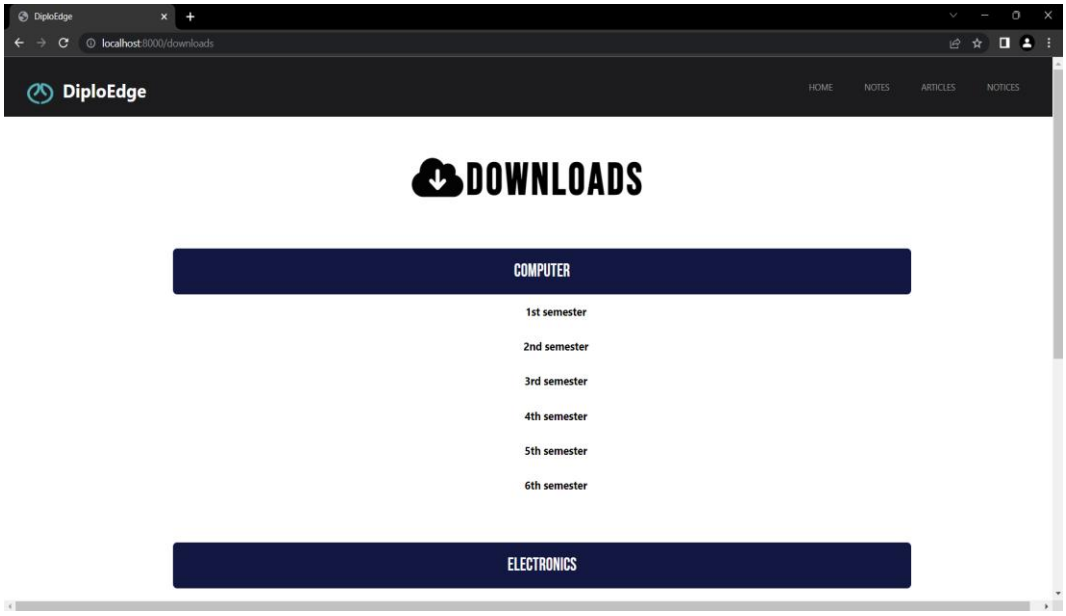

Figure 3.26: Notes Page (B)

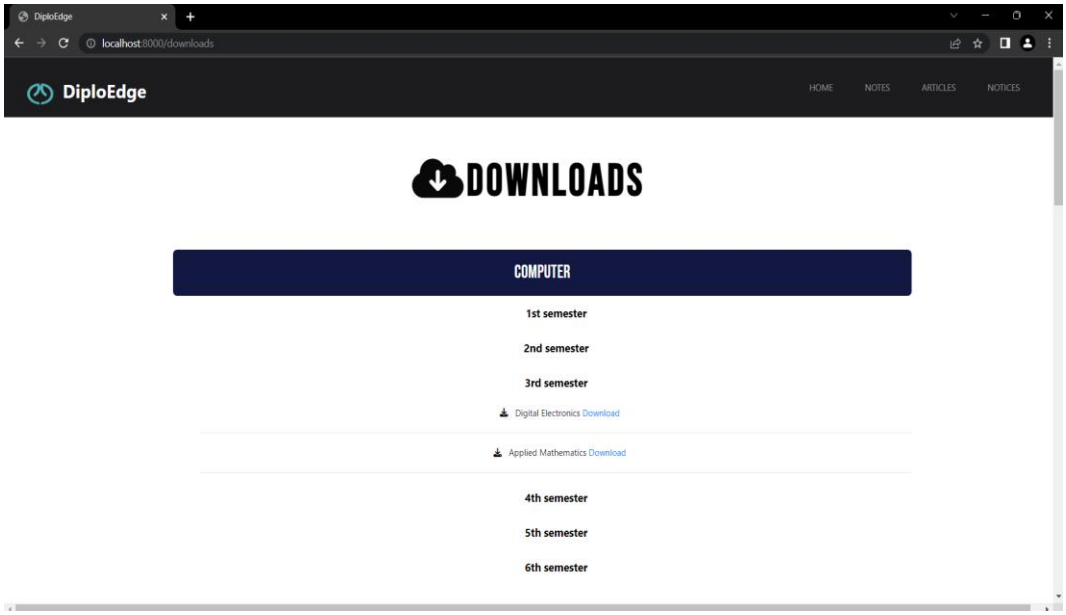

Figure 3.27: Notes Page(C)

3.8.6.3. Notices Page

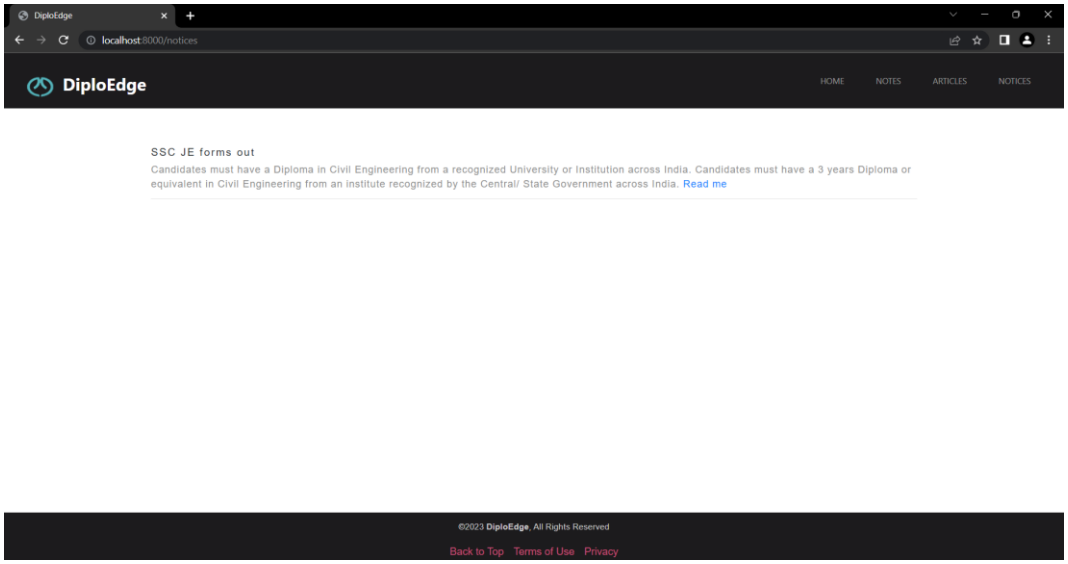

Figure 3.28: Notices Page

3.8.6.4. Compose Page

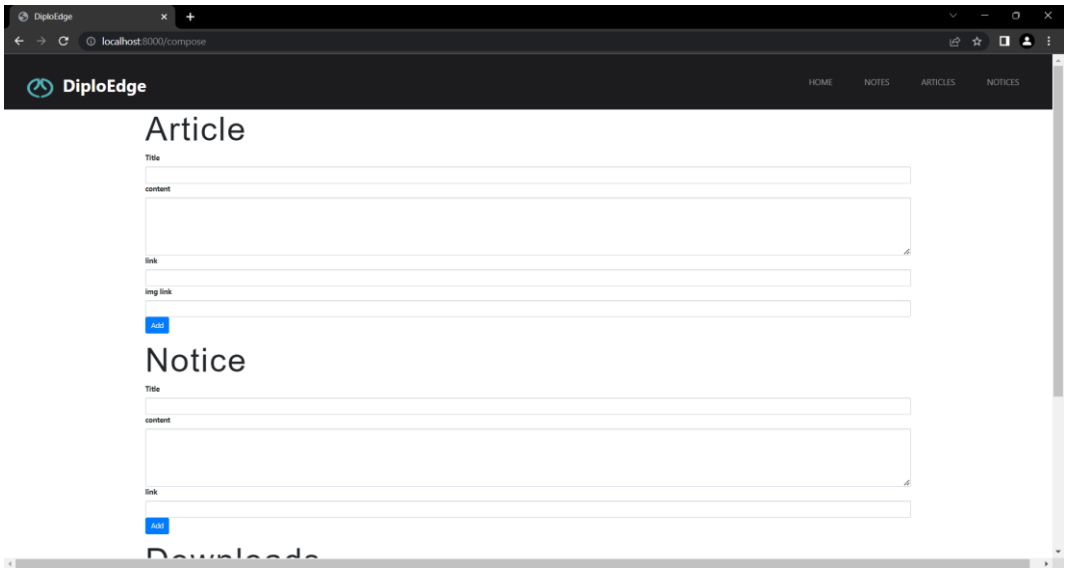

Figure 3.29: Compose Page (A)

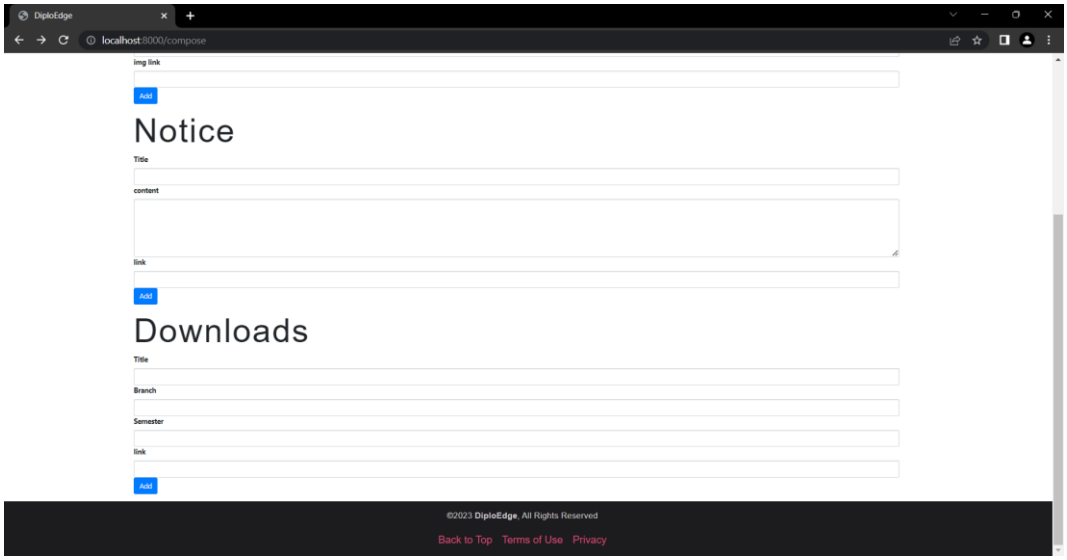

Figure 3.30: Compose Page (B)

3.8.6.5. Articles Page

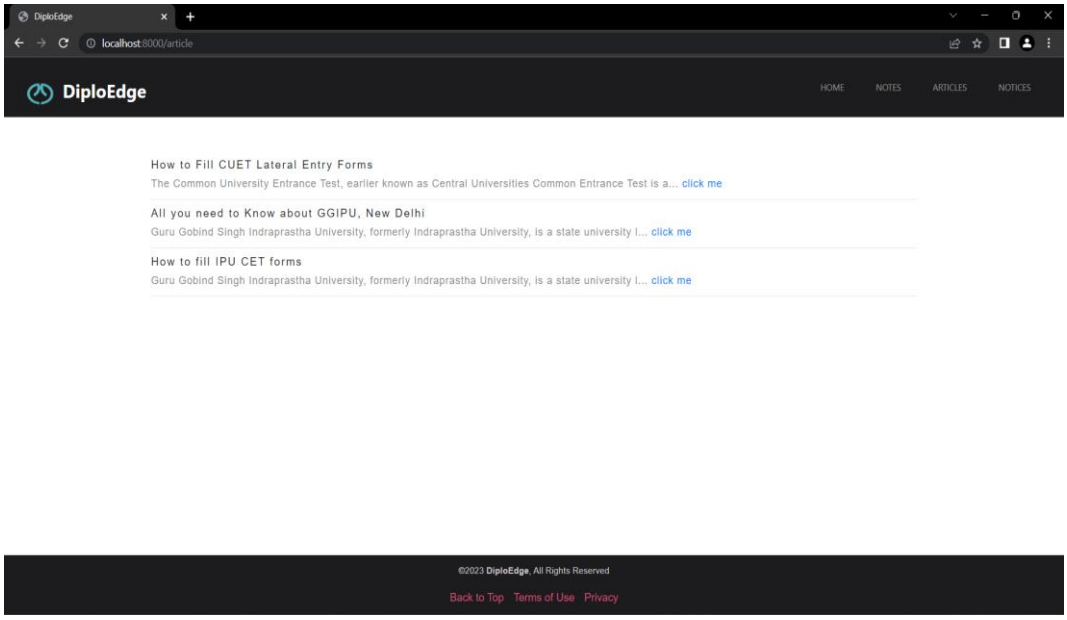

Figure 3.31: Articles Page

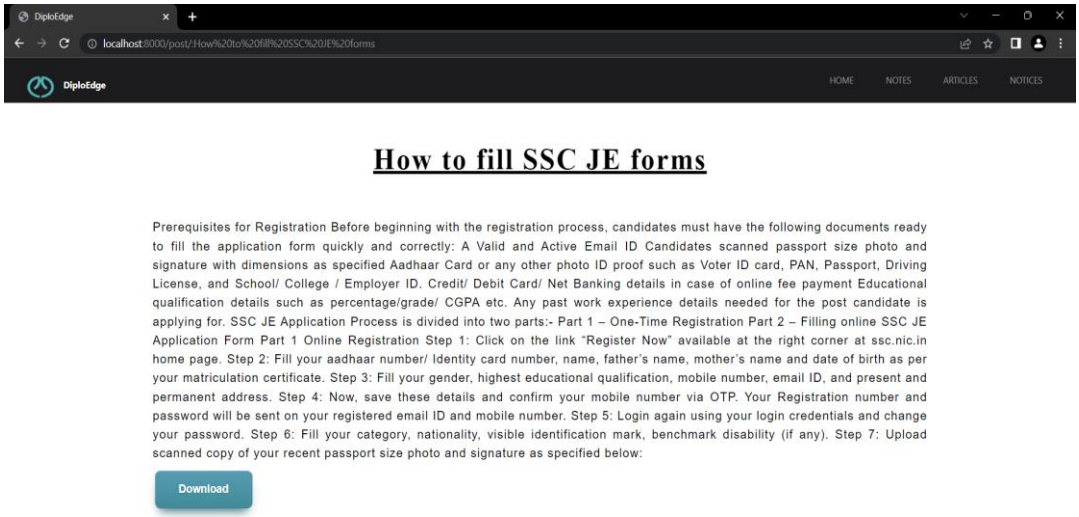

Figure 3.32: Individual Article Page

CHAPTER 4: CONCLUSION

4.1. Conclusion

In conclusion, DiploEdge is a platform designed to address the lack of reliable, up-to-date resources for Diploma in Engineering students on the web. Our aim is to provide students with a comprehensive online resource that is easy to navigate and provides all the information they need in one place.

We believe that DiploEdge has the potential to revolutionize the way Diploma in Engineering students access educational resources and connect with industry professionals. By providing a one-stop-shop for students to find all the information they need, we hope to make it easier for students to succeed in their studies and careers. Looking to the future, we are excited about the possibilities for further expansion and development of the platform. With our plans for online courses, interactive forums, and new resources, we are confident that DiploEdge will continue to be a valuable resource for Diploma in Engineering students for years to come.

4.2. Future Scope of Work

While DiploEdge is already a valuable resource for Diploma in Engineering students, we believe that there is significant scope for further expansion and development of the platform. In the future, we plan to introduce a range of new features and resources to enhance the platform's value to students.

One area of future development for DiploEdge is online courses. We recognize that many students are looking for flexible and accessible ways to enhance their skills and knowledge, and we believe that online courses could be an excellent addition to the platform. We plan to partner with leading universities and educational institutions to provide our users with access to high-quality educational resources and programs.

Another area of future development for DiploEdge is interactive forums. We recognize that students benefit from the ability to connect with each other and with industry professionals, and we plan to create a platform where students can share their experiences, ask questions, and receive guidance from their peers and experts in the field.

We also plan to expand the platform's resources to include more articles, tutorials, and other materials to support students in their studies and careers. By continuing to update and expand the platform, we hope to ensure that it remains a valuable resource for Diploma in Engineering students.

BIBLIOGRAPHY

1. Markus Dickinson and Dan Ioan Tufis “Iterative Enhancement” In book: Handbook of Linguistic Annotation (pp.257-276)

3. w3schools.com HTML Tutorial https://www.w3schools.com/html/

4. w3.org Cascading Style Sheetshome page https://w3/Style/css

5. getbootstrap.com https://getbootstrap.com/

6. Sanja Delcev and Drazen Draskovic “Modern JavaScript frameworks: As survey study ” Published in: 2018 Zooming Innovation in Consumer Technologies Conference (ZINC)

7. ejs.co https://ejs.co/

8. mdn web docs https://developer_mozilla.org

9. Medium.com https://medium.com/@davidpetri/server-and-client-side-validation-with-javascript-html-and-hapi-js-eccc779e448a

10. mongodb.com https://www.mongodb.com/\

11. mongoing https://mongoing.com/docs/reference/program/mongod.html

12. dev.to https://dev.to/alexmercedcoder/mongoose-connecting-to-mongo-via-javascript-pk5

13. freecodecamp.org https://www.freecodecamp.org/news/modular-programming-nodejs-npm-modules/

APPENDICES

1. EJS (Embedded JavaScript):

- "EJS Templating System in Node.js" by Adnan Rahić (https://medium.com/@adnanrahic/ejs-templating-system-in-nodejs-27ac4f9c6302)
- "EJS Tutorial" by Tutorialspoint (https://www.tutorialspoint.com/ejs/index.htm)

2. JavaScript:

- "JavaScript: The Good Parts" by Douglas Crockford
- "Eloquent JavaScript: A Modern Introduction to Programming" by Marijn Haverbeke
- "You Don't Know JS" by Kyle Simpson

3. HTML/CSS:

- "HTML and CSS: Design and Build Websites" by Jon Duckett
- "CSS Mastery: Advanced Web Standards Solutions" by Andy Budd, Simon Collison, and Cameron Moll
- "Responsive Web Design" by Ethan Marcotte

4. Node.js/Express:

- "Node.js in Action" by Mike Cantelon, Marc Harter, T.J. Holowaychuk, and Nathan Rajlich
- "Express in Action" by Evan Hahn

5. MongoDB/Mongoose/Mongod:

- "MongoDB: The Definitive Guide" by Kristina Chodorow and Michael Dirolf

- "Getting MEAN with Mongo, Express, Angular, and Node" by Simon Holmes

6. MERN stack:

- "MERN Quick Start Guide: Build web applications with MongoDB, Express.js, React, and Node" by Eddy Wilson Iriarte Koroliova
- "Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node" by Vasan Subramanian