COLLEGE CODE: 5118

COLLEGE NAME: PODHIGAI COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT : B.Tech., IT

STUDENT NM-ID :e0b1c9446e74e6381fc46330bae7fd21

ROLL NO : 511823205007

DATE : 11-10-2025

IBM-NJ-BLOGGING PLATFORM

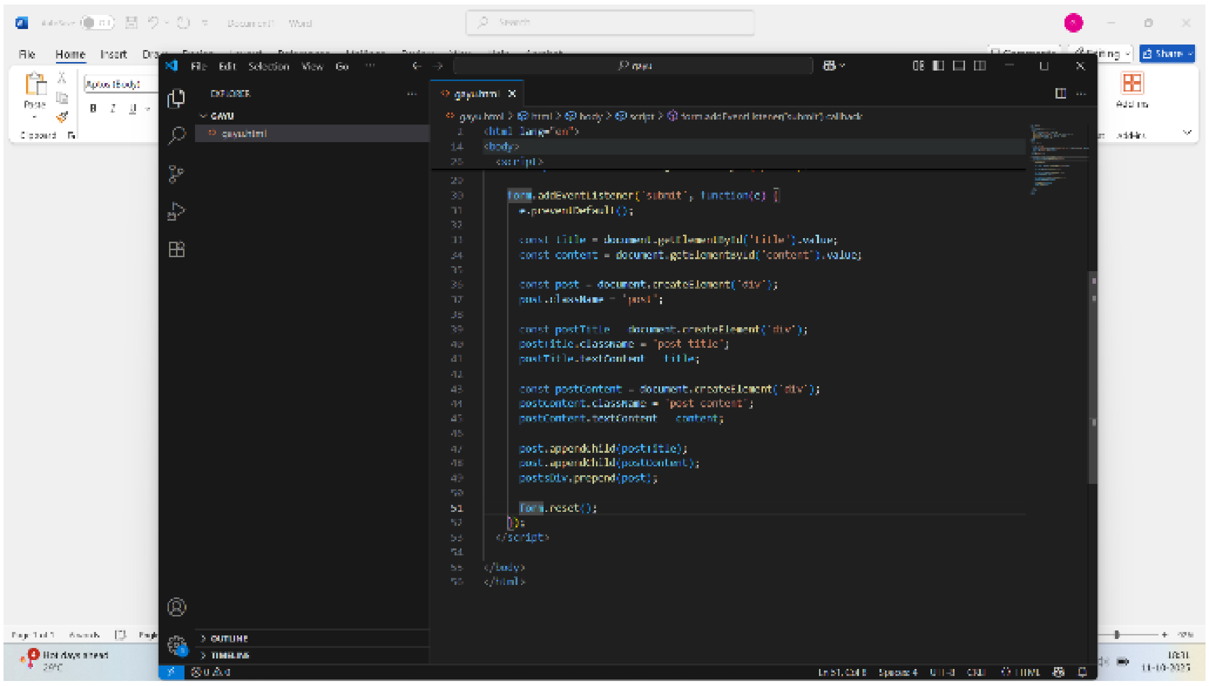
Phase-V

TECHNOLOGY PROJECT NAME : PROJECT DEMONSTRATE &DOCUMENT

SUBMITTED BY,
NAME: GAYATHRI. V
PHONENO:6369709185

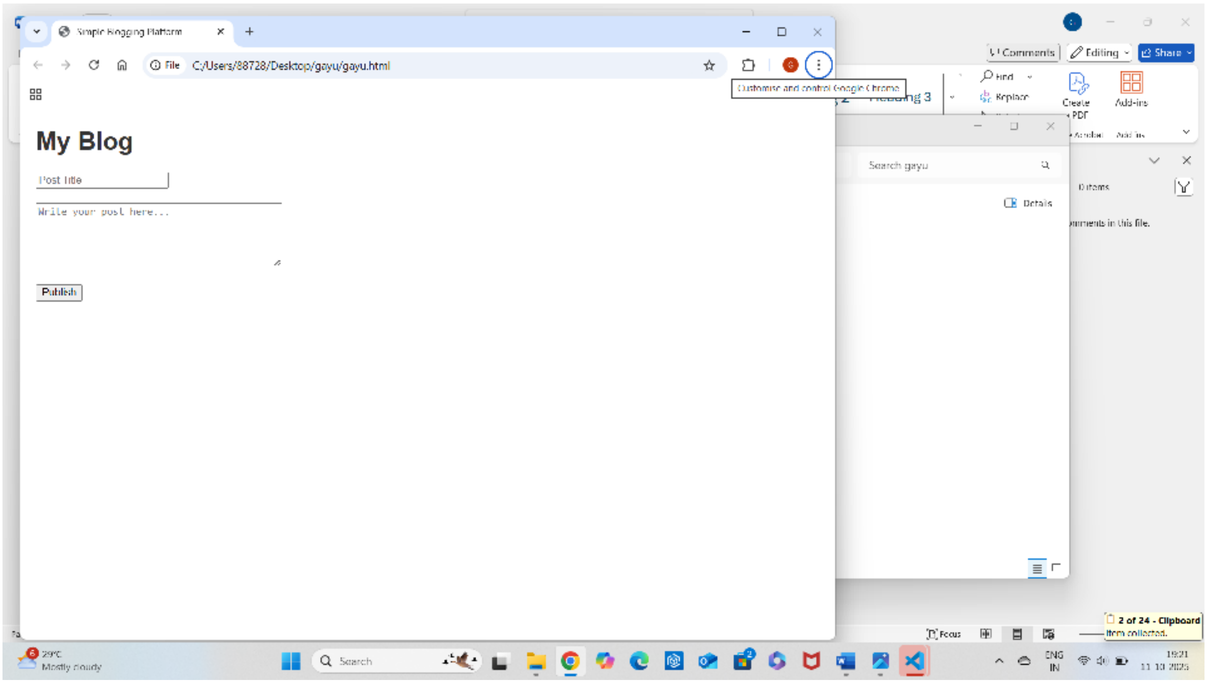
NJ-BLOGGING PLATFORM

FINAL DEMOWALK THROUGH:



gayu.html > html > body > script > form.addEventListener('submit') callback

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

    form.addEventListener('submit', function(e) {
      e.preventDefault();

      const title = document.getElementById('title').value;
      const content = document.getElementById('content').value;

      const post = document.createElement('div');
      post.className = 'post';

      const postTitle = document.createElement('div');
      postTitle.className = 'post-title';
      postTitle.textContent = title;

      const postContent = document.createElement('div');
      postContent.className = 'post-content';
      postContent.textContent = content;

      post.appendChild(postTitle);
      post.appendChild(postContent);
      postsDiv.prepend(post);

      form.reset();
    });
  </script>

</body>
</html>
```



Simple Blogging Platform

C:/Users/88728/Desktop/gayu/gayu.html

My Blog

Post Title

Write your post here...

Publish

1. The user lands on the homepage with login/register options.
2. After login, users are directed to the dashboard showing

recent blog posts.

3. Users can create, edit, save as draft, and publish blog posts.

4. Admin can manage users, moderate content, and view analytics.

5. The blog editor supports text formatting, image uploads, and tags.

6. Blogs can be searched, filtered by category, or sorted by date/ popularity.

7. Responsive UI adapts to mobile and desktop screens.

8. Notifications are shown on successful actions (e.g., post published).

9. APIs are secured with authentication and return JSON responses.

10. The app is hosted (e.g., on Vercel/Render) and includes basic analytics and logs.

PROJECT REPORT:

1. Project Title: NJ Blogging Platform

2. Objective: To build a simple and user-friendly platform for creating, publishing, and managing blogs.

3. Tech Stack: Node.js (Backend), Express.js (API), MongoDB (Database), HTML/CSS/JS (Frontend), React (UI).

4. Features: User registration/login, blog creation, drafts, publish, edit, delete, and comment system.

5. Authentication: Implemented using JWT for secure access.

6. Routing: RESTful APIs for handling blog and user operations.

7. UI/UX: Clean responsive design using React and Tailwind CSS.

8. Database: MongoDB for storing users, blogs, and comments.

9. Admin Panel: Admins can moderate content and manage users.

10. API Security: Routes are protected using middleware.

11. Error Handling: Consistent error responses across endpoints.

12. Form Validation: Frontend and backend validation on inputs.

13. Testing: Manual testing of API endpoints and UI flows.

14. Hosting: Frontend on Netlify, backend on Render, database on MongoDB Atlas.

15. Deployment: Continuous deployment with GitHub integration.

16. Performance: Optimized API and lazy loading of blog lists.

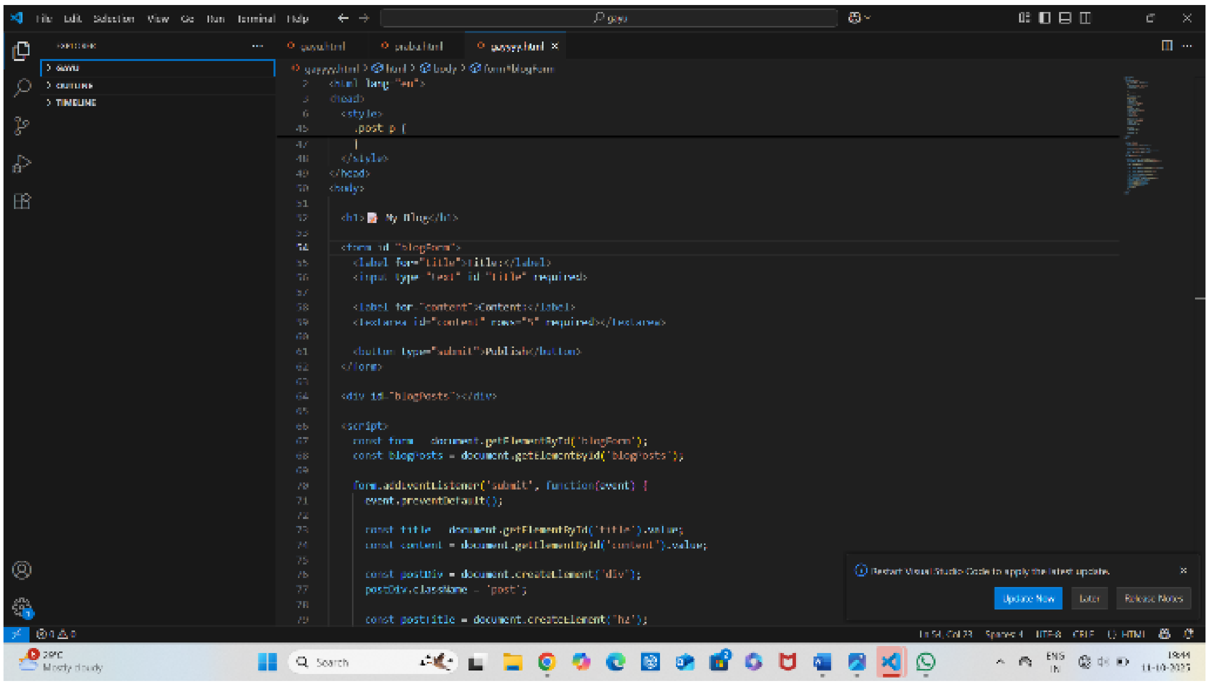
17. Monitoring: Console logging and basic uptime monitoring.

18. Accessibility: Follows basic WCAG guidelines.

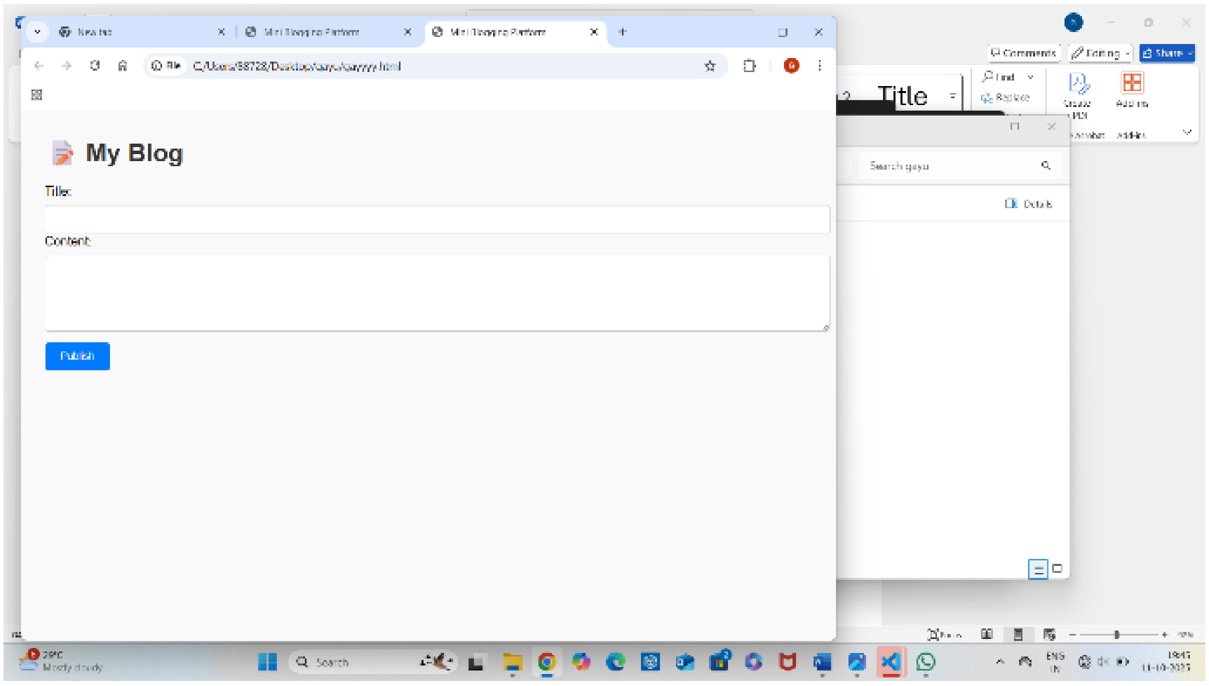
19. Outcome: Fully functional blog platform for users and admins.

20. Future Scope: Add SEO tools, multilingual support, and analytics dashboard.

SCREENSHOT/APPLICATION DOCUMENT:



```
<html lang="en">
<head>
  <style>
    .post p {
    }
  </style>
</head>
<body>

  <h1>📝 My Blog</h1>

  <form id="blogForm">
    <label for="title">Title:</label>
    <input type="text" id="title" required>

    <label for="content">Content:</label>
    <textarea id="content" rows="5" required></textarea>

    <button type="submit">Publish</button>
  </form>

  <div id="blogPosts"></div>

  <script>
    const form = document.getElementById('blogForm');
    const blogPosts = document.getElementById('blogPosts');

    form.addEventListener('submit', function(event) {
      event.preventDefault();

      const title = document.getElementById('title').value;
      const content = document.getElementById('content').value;

      const postDiv = document.createElement('div');
      postDiv.className = 'post';

      const postTitle = document.createElement('h2');
```



📝 My Blog

Title:

Content:

Publish

CHALLENGES&SOLUTION:

1. User Authentication & Security

   - Challenge: Securing user sessions & data.

   - Solution: Used JWT & password hashing (bcrypt).

2. Content Management
   - Challenge: Handling draft and publish logic.
   - Solution: Added status fields (draft / published) in blog schema.

3. Responsive UI Design
   - Challenge: Ensuring good UI on all devices.
   - Solution: Used Tailwind CSS & tested on mobile + desktop.

Features:

1. User authentication
2. Post creation and editing
3. Comment system

4. Categories and tags

Tech Stack:

1. Frontend: [HTML, CSS, JavaScript, React/Angular/Vue]
2. Backend: [Node.js, Ruby on Rails, Django]
3. Database: [MySQL, MongoDB]

GitHub README:

Description
A simple blogging platform built using [programming languages/frameworks].

Features
* User authentication
* Post creation and editing
* Comment system
* Categories and tags

Tech Stack
* Frontend: [HTML, CSS, JavaScript, React/Angular/Vue]

* Backend: [Node.js, Ruby on Rails, Django]
* Database: [MySQL, MongoDB]

Setup Guide

1. Clone the repository: `git clone https:
2. Install dependencies: npm install or pip install -r requirements.txt

3. Start the server: npm start or python manage.py runserver

Contributing

Contributions are welcome! Please submit a pull request with your changes.

Setup Guide:

Step 1: Clone the repository

```
git clone https://github.com/your-username/your-repo-name.git
```

Step 2: Install dependencies

```
npm install
```

or

```
pip install -r requirements.txt
```

Step 3: Start the server

```
npm start
```

or

```
python manage.py runserver
```

Documentation:

- You can add more detailed documentation in the README or create a separate documentation folder.
- Use tools like Swagger or API Blueprint for API documentation.

Commit messages:

- Follow the conventional commit message format: fix:, feat:, docs:, etc.

Example: feat: add user authentication

Best practices:

- Follow coding standards and best practices for your chosen programming language.

- Use meaningful variable names and comments.
- Keep your code organized and modular.

Would you like me to elaborate on any specific section?4. API Performance

   - Challenge: Slow API response on load.

   - Solution: Used pagination, indexing in MongoDB.

5. Deployment

   - Challenge: Coordinating frontend/backend/database deployment.

   - Solution: Used Render (backend), Netlify (frontend), MongoDB Atlas.

Flowchart (Text Representation)

[Start]

[User Registers/Login]

[Dashboard Loaded]

[Create/Edit Blog]

[Save as Draft or Publish]

[Blog Stored in DB via API]

[Fetch Blogs ! Show in Feed]

[Admin Moderation (if needed)]

[User Reads or Interacts]

[End]

GITHUB README &SETUP GUIDE:

Tech Stack:

1. Frontend: [HTML, CSS, JavaScript, React/Angular/Vue]
2. Backend: [Node.js, Ruby on Rails, Django]
3. Database: [MySQL, MongoDB]

GitHub README:

Description

A simple blogging platform built using [programming languages/frameworks].

 Features

* User authentication

* Post creation and editing

* Comment system

* Categories and tags

Tech Stack

* Frontend: [HTML, CSS, JavaScript, React/Angular/Vue]

* Backend: [Node.js, Ruby on Rails, Django]

* Database: [MySQL, MongoDB]

 Setup Guide

1. Clone the repository: `git clone https:

2. Install dependencies: npm install or pip install -r requirements.txt

3. Start the server: npm start or python manage.py runserver

Contributing

Contributions are welcome! Please submit a pull request with your changes.

Setup Guide:

Step 1: Clone the repository

```
git clone https://github.com/your-username/your-repo-name.git
```

Step 2: Install dependencies

```
npm install
```

or

```
pip install -r requirements.txt
```

Step 3: Start the server

```
npm start
```

or

```
python manage.py runserver
```

Documentation:

- You can add more detailed documentation in the README or create a separate documentation folder.
- Use tools like Swagger or API Blueprint for API documentation.

Commit messages: Follow the conventional commit message format: fix:, feat:, docs:, etc.

Example: feat: add user authentication

Best practices:

- Follow coding standards and best practices for your chosen programming language.

- Use meaningful variable names and comments.

- Keep your code organized and modular.

FINAL SUBMISSION (REPO+DEPLOYED LINK):

GitHub - gayathri56666/NJ-blogging-platform-project-1