



Inspiring Excellence

CSE471: System Analysis and Design

Project Report

Project Title: Storytelling Website

Group: 03

Section: 04

Summer 2024

Submitted to:

Mr. Hamim Ibne Nasim (HBN)

Mr. MD Asif (MDS)

Submitted by:

ID	Name
21301062	Jannatul Ferdaus
21301191	Anonna Dev Nipa
21301105	Tangena Islam
21301196	Swachcha Podder

Table of Contents

Section	Content	Page No
1	Introduction	3
2	Functional Requirements	4
3	User Manual	5
4	Frontend Development	16
5	Backend Development	39
6	Technology (Framework, Languages)	44
7	Github Repo Link	44
8	Individual contribution	45

Introduction

The SOYO (Story of Your Own) storytelling project is a functional website built using MERN Stack framework. The goal of this project is to allow users to take part in a story community either being a reader or an uprising writer. The users here can write their own stories or search for stories written by other users and build a community.

The project involves several features like:

- User authentication and profiles,
- A vast library of user-generated stories with advanced search and filtering options,
- A rich text editor with styling tools for enhancing narrative structure,
- A robust content management system for authors,
- And an overall emphasis on providing an engaging, user-friendly platform for storytelling

The rapid evolution of digital platforms has redefined the way readers and writers connect with literature. The SOYO Writing Platform is a comprehensive initiative designed to bring readers and writers together in a seamless online environment, offering an extensive range of stories, and creative works that cater to diverse tastes and genres.

SOYO envisions an intuitive online platform where users can effortlessly explore, discover, and engage with content that resonates with their interests. By focusing on a user-centric interface and interactive features, SOYO replicates the intimate experience of traditional reading while embracing the flexibility and convenience of digital access. The platform aims to foster an engaging environment where readers can easily search for their preferred genres, authors, and topics, while writers can publish, share, and grow their audience.

This project encapsulates the principles of successful online platforms, blending user experience optimization, content management, and community-building tools. Through innovation, accessibility, and personalization, SOYO seeks to transform the way modern readers consume and interact with written content, enabling a dynamic, digital literary community.

Functional Requirements

Module 1:

User management (author/readers):

- -login/signup/authentication
- -profile picture upload/change/delete
- -Followers/following
- -Works(published/drafts)
- -Add to list(add stories in your list) (read/reading/want to read)

Module 2:

Writing:

- -Modify story/save
- -Add tags
- -Default title and summary font : Standardize the appearance of titles and summaries.
- -Comment section under each chapter: Enable comments on each chapter with features to reply, like, delete, report, copy, and block users.
- -Translations

Module 3:

Content Discovery and Organization:

- Genre: use tags to organize works into genre
- Search Functionality
- Sorting stories
- Top reads: Display top read stories of the day in front page
- Bookmarking: Users can bookmark stories to receive update on new chapters
- Display uprising authors

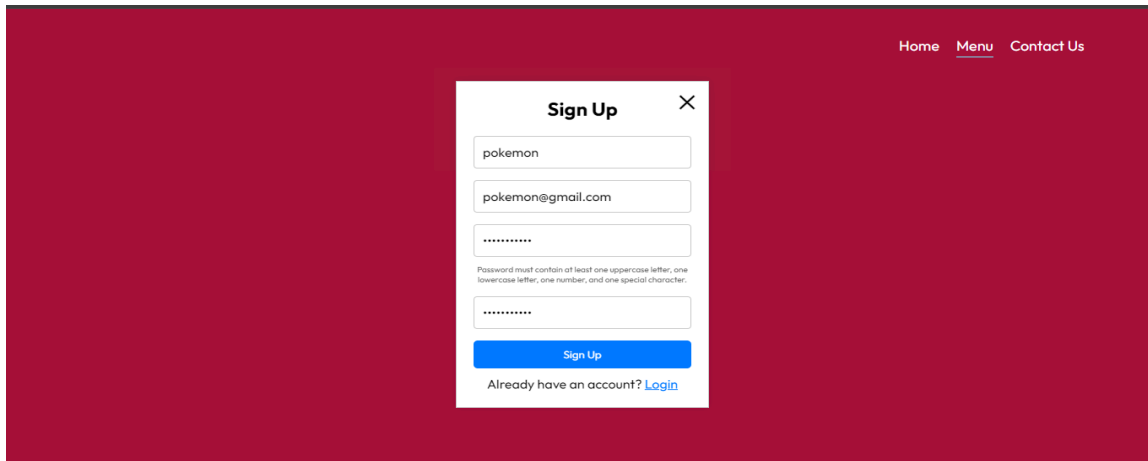
Module 4:

Story discussion forum:

- Create/delete post
- Reply to post
- Like, delete, copy post/replies
- Idea Polling: Create polls within posts to gather opinions on story directions or character development.
- Resource Sharing: Allow users to share writing resources using QR code within the forum.
- Real time chat

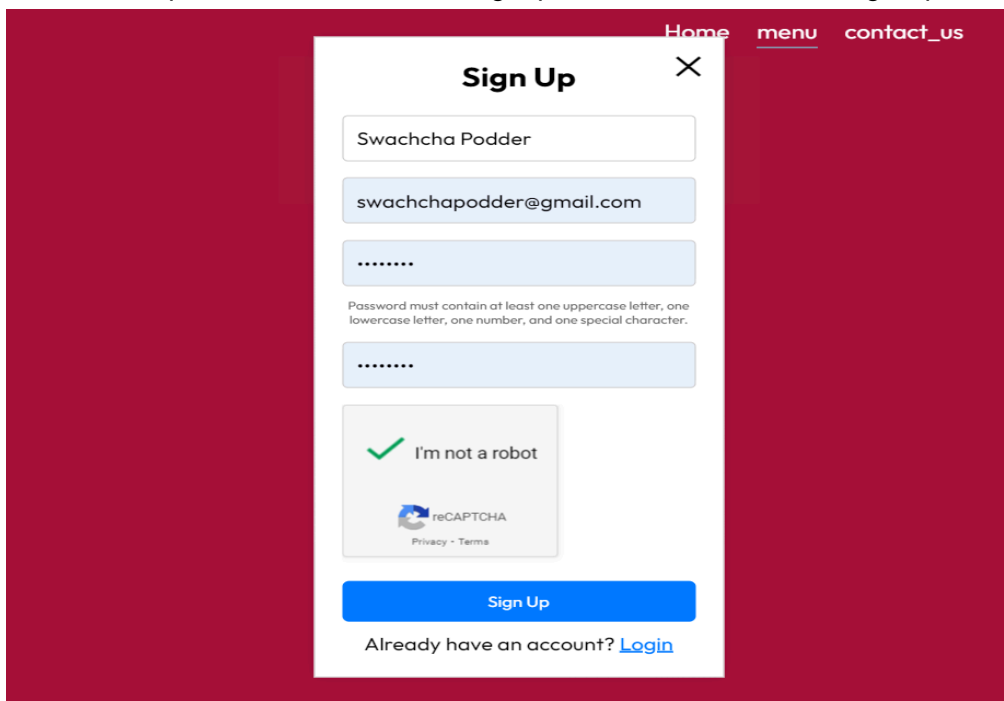
User Manual

This is the **sign up page** where the user creates a new account. Username, email and password is required. The password must be 8 characters long with uppercase letters, lowercase letters, numbers and special characters.



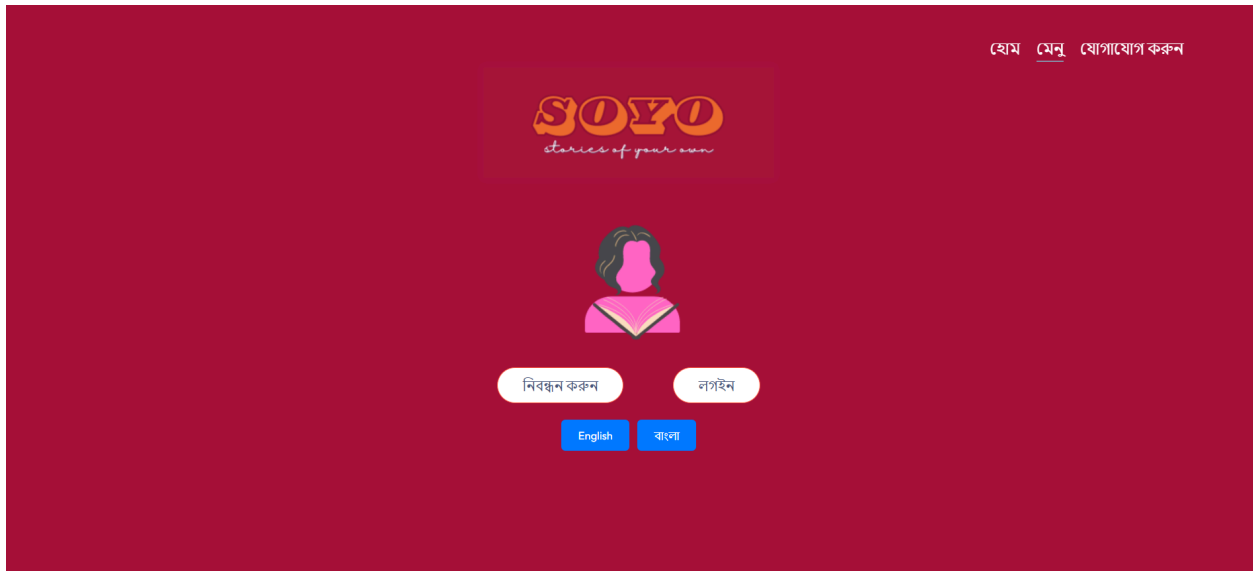
The screenshot shows a 'Sign Up' modal form centered on a dark red background. The form has a title 'Sign Up' and a close button 'X'. It contains three input fields: a username field with 'pokemon', an email field with 'pokemon@gmail.com', and a password field with masked characters. Below the password field is a small text requirement: 'Password must contain at least one uppercase letter, one lowercase letter, one number, and one special character.' There is another masked password field below this. At the bottom of the form is a blue 'Sign Up' button and a link 'Already have an account? Login'. In the top right corner of the background, there are links 'Home', 'Menu', and 'Contact Us'.

This is the **Authentication page** where it is detected if the user is already signed in or not. It also resets the Captcha after a successful signup or once clicked on the Sign Up button.

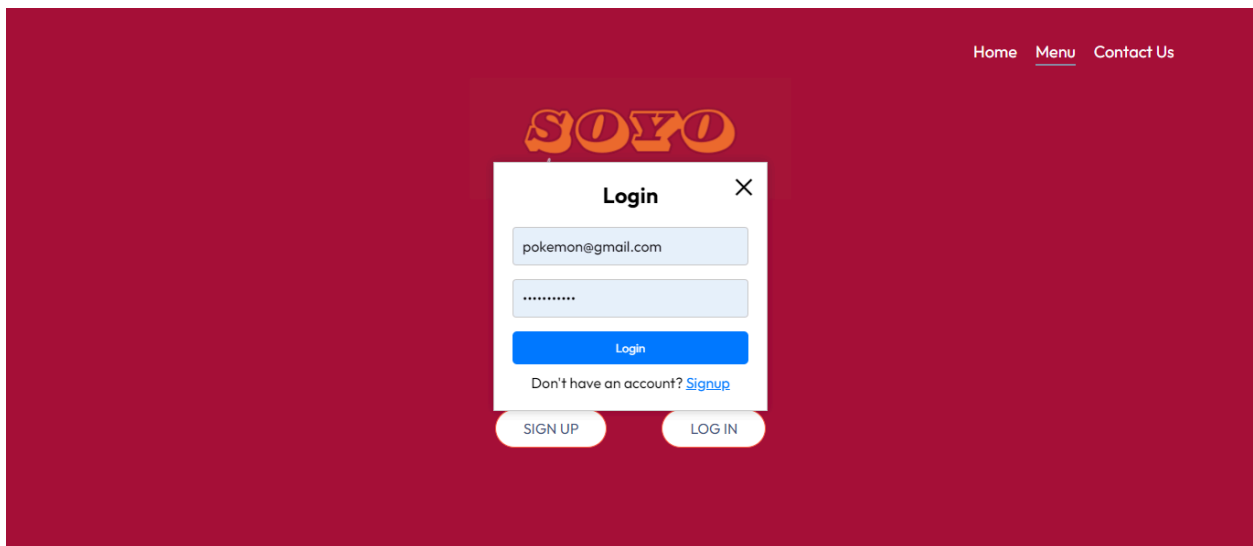


This screenshot shows the 'Sign Up' form after a successful submission or captcha reset. The form fields now contain: 'Swachcha Podder' for the username, 'swachchapodder@gmail.com' for the email, and two masked password fields. Below the password fields is a reCAPTCHA section with a green checkmark and the text 'I'm not a robot'. Below the reCAPTCHA is the 'reCAPTCHA' logo and links for 'Privacy' and 'Terms'. The blue 'Sign Up' button and the 'Login' link are still present at the bottom. The background links 'Home', 'menu', and 'contact_us' are also visible.

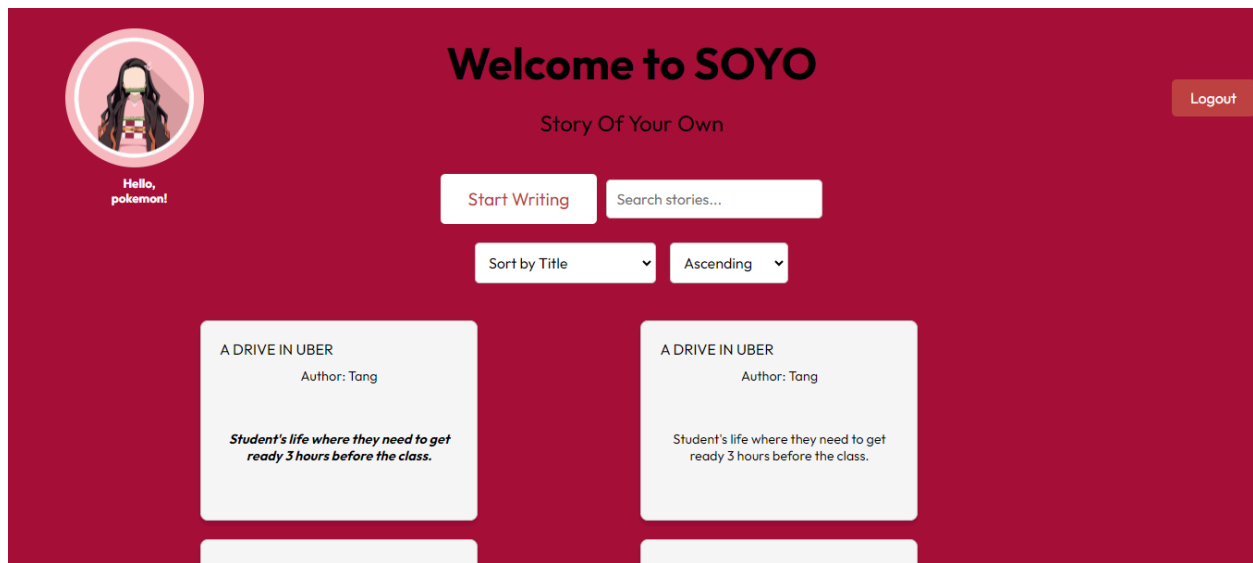
Options to switch languages, the user can translate the feature instructions to either of these languages



This is the **login** popup



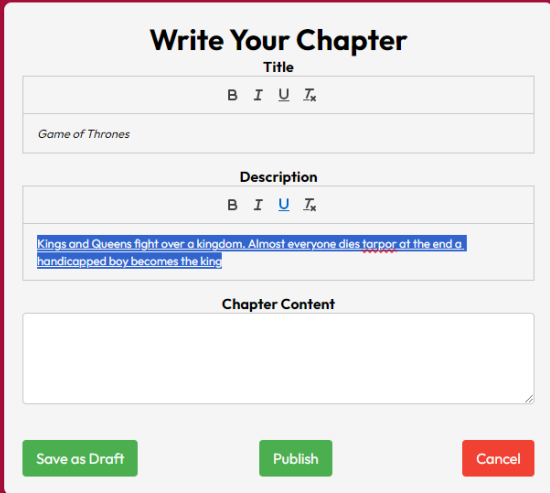
This is the homepage



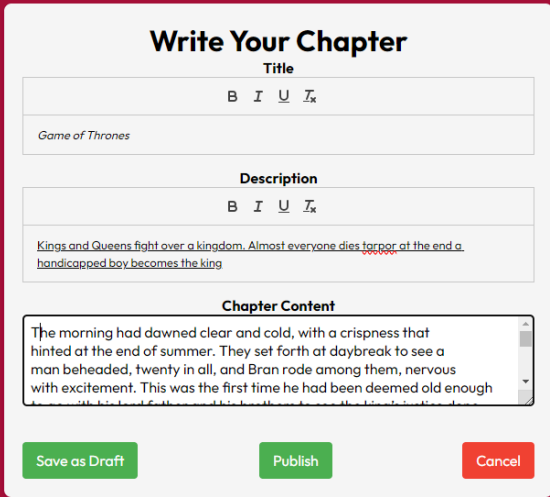
After clicking the start writing button, the user is redirected to the writing page where they fill up the story information to be displayed.

The screenshot shows a form titled "Create a New Story" on a dark red background. The form is a light gray box with a white border. It contains several input fields and dropdown menus. The first field is "Topic Name" with the text "Game of Thrones" entered. The second field is "Description" with the text "Kings and Queens fight over a kingdom. Almost everyone dies tarpor at the end a handicapped boy becomes the king" entered. The third field is "Category" with a dropdown menu showing "Fantasy". The fourth field is "Tags" with the text "dragon, jon snow, throne" entered. The fifth field is "Language" with a dropdown menu showing "English". At the bottom of the form, there are two buttons: a green "Next" button and a red "Cancel" button.

After clicking the next button, the user is redirected to a chapter writing page. Here they can also **style the title and description** according to their choice.

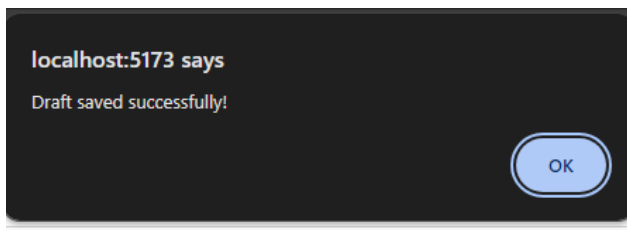


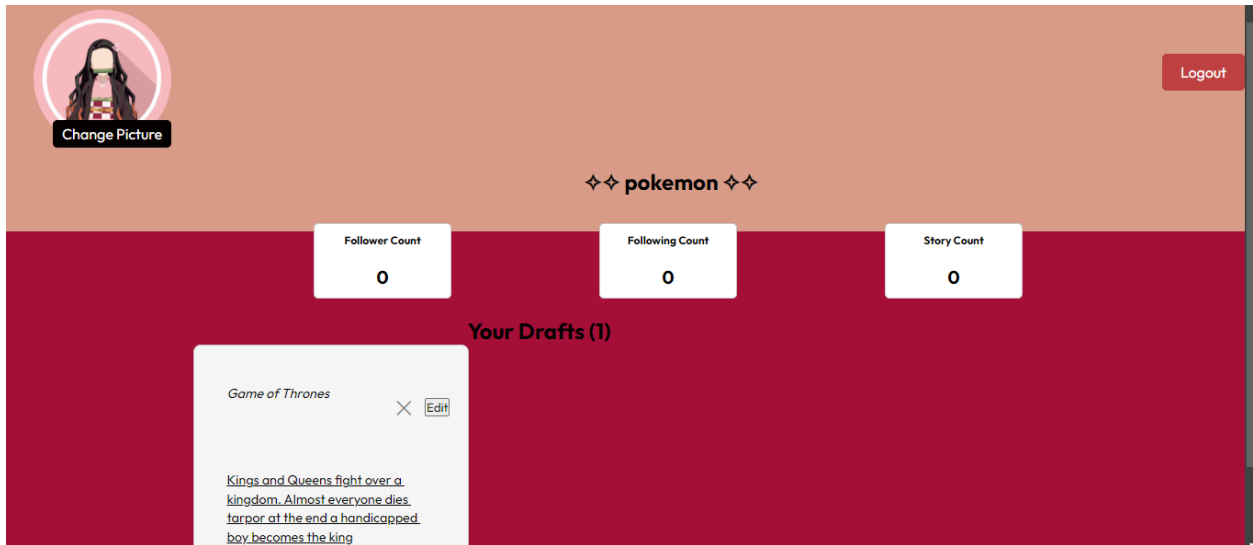
The screenshot shows a web form titled "Write Your Chapter" on a dark red background. The form has three main sections: "Title", "Description", and "Chapter Content". The "Title" section has a text input field containing "Game of Thrones" and a toolbar with icons for bold (B), italic (I), underline (U), and strikethrough (ABC). The "Description" section has a text input field containing "Kings and Queens fight over a kingdom. Almost everyone dies for poor at the end a handicapped boy becomes the king" and a similar toolbar. The "Chapter Content" section has a large, empty text area. At the bottom of the form are three buttons: "Save as Draft" (green), "Publish" (green), and "Cancel" (red).



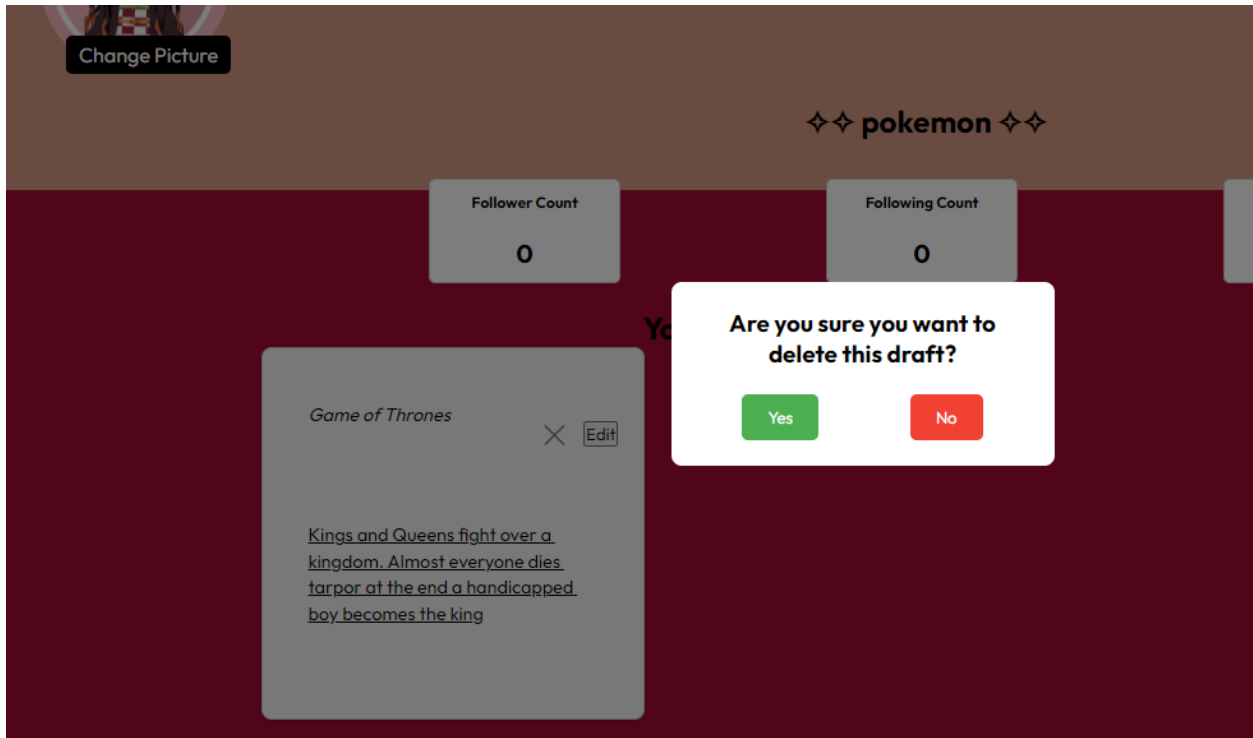
This screenshot shows the same "Write Your Chapter" form, but now the "Chapter Content" section is filled with text: "The morning had dawned clear and cold, with a crispness that hinted at the end of summer. They set forth at daybreak to see a man beheaded, twenty in all, and Bran rode among them, nervous with excitement. This was the first time he had been deemed old enough". The text is displayed in a light gray box with a scrollbar on the right. The "Title" and "Description" sections remain unchanged. The "Save as Draft", "Publish", and "Cancel" buttons are still at the bottom.

After writing a chapter, the user can click the **save as draft button** and then their stories will be saved in their profile as drafts.





The user can choose to **delete their draft** if they wish to by clicking the cross button.



Write Your Chapter

Title

B I U *↵*

Game of Thrones

Description

B I U *↵*

Kings and Queens fight over a kingdom. Almost everyone dies tarpor at the end a handicapped boy becomes the king

Chapter Content

The morning had dawned clear and cold, with a crispness that hinted at the end of summer. They set forth at daybreak to see a man beheaded, twenty in all, and Bran rode among them, nervous with excitement. This was the first time he had been deemed old enough

Save as Draft

Publish

Cancel

However, they can also choose to **edit their draft** and click the publish button. Doing so will display their story in the homepage among all the stories

Game of Thrones

Author: pokemon

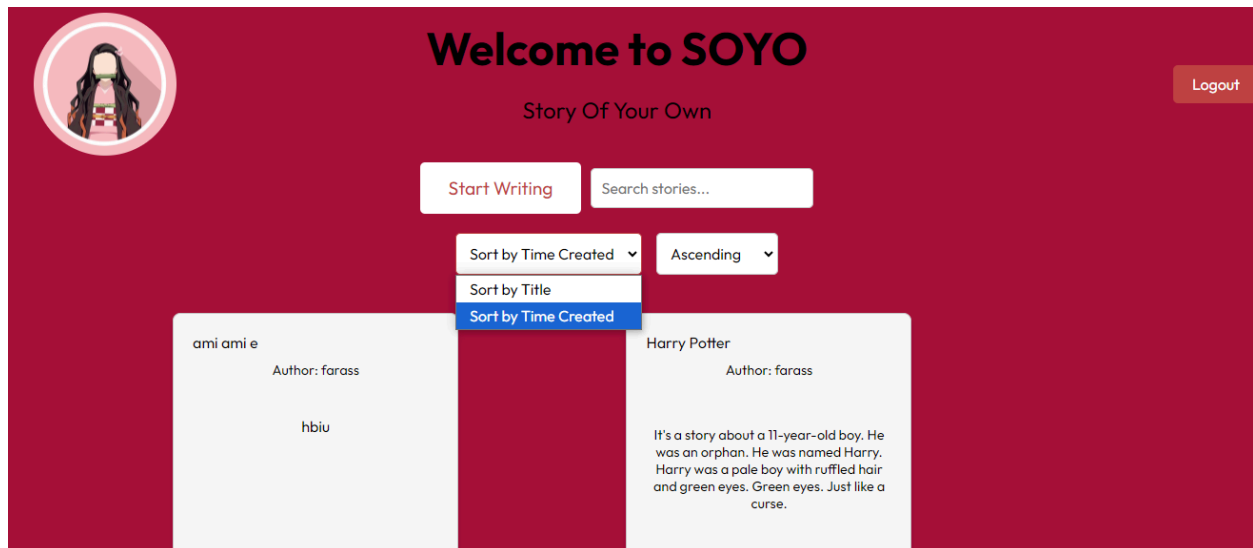
Kings and Queens fight over a kingdom. Almost everyone dies tarpor at the end a handicapped boy becomes the king

Harry Potter

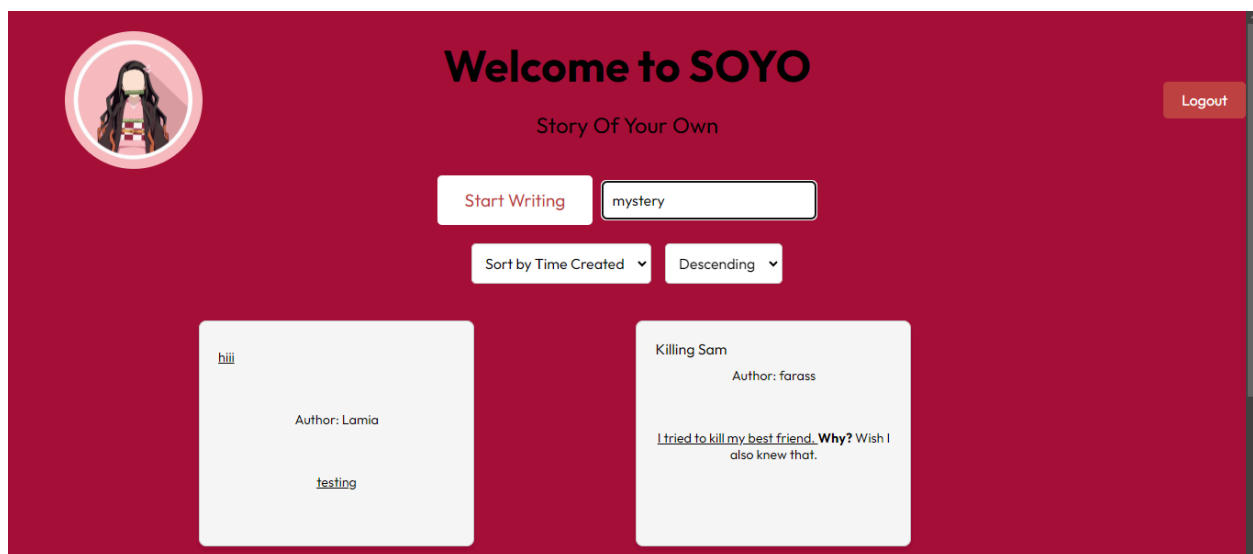
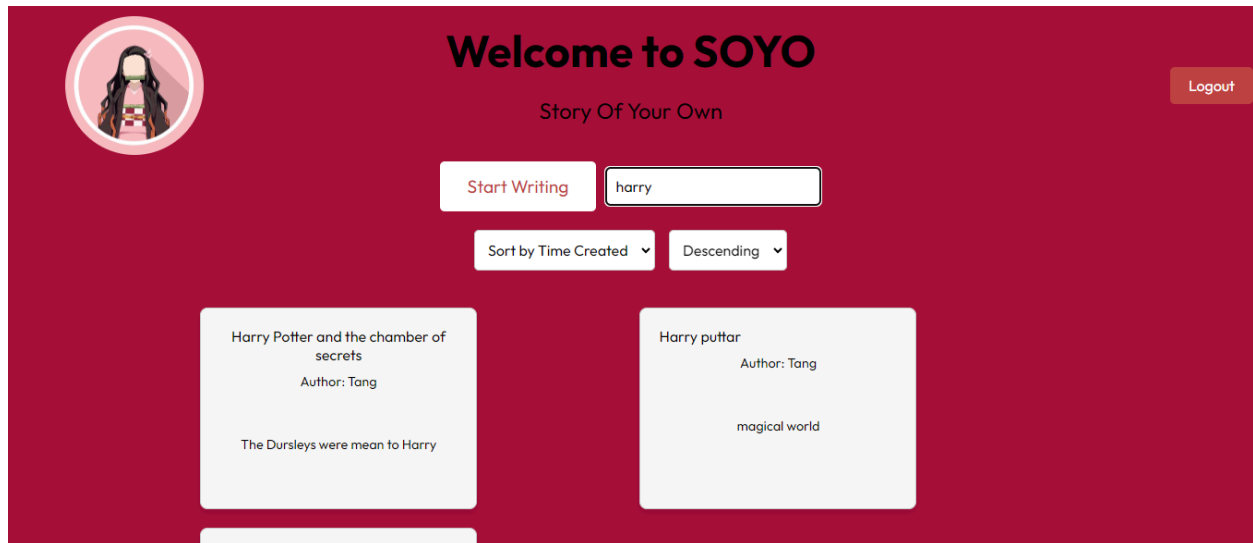
Author: farass

It's a story about a 11-year-old boy. He was an orphan. He was named Harry. Harry was a pale boy with ruffled hair and green eyes. Green eyes. Just like a curse.

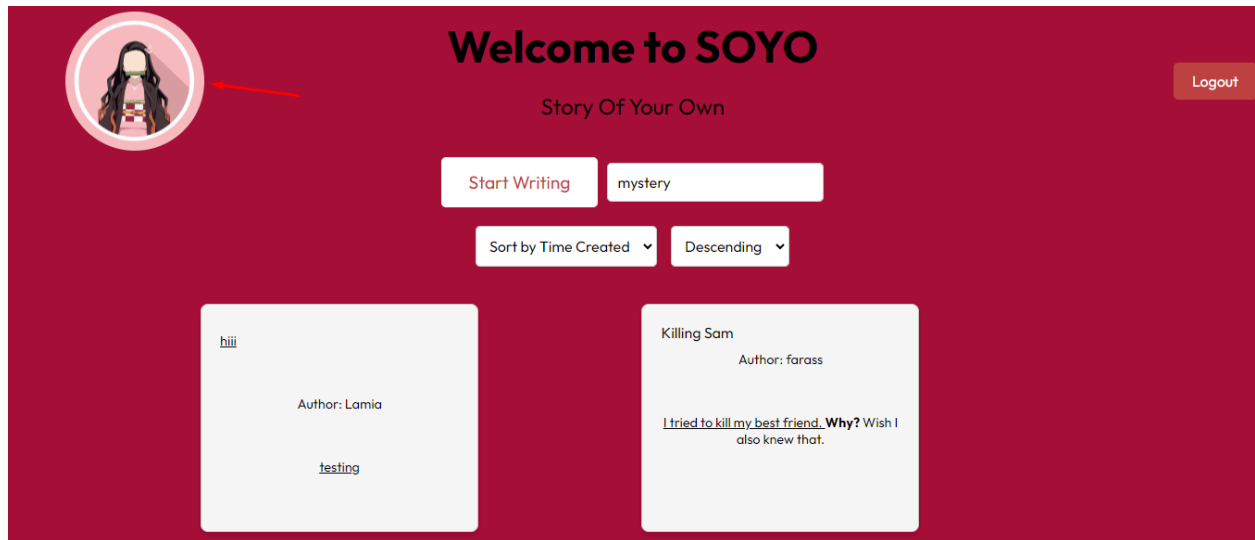
The readers can also **sort the published stories** according to date and title in both ascending and descending order.



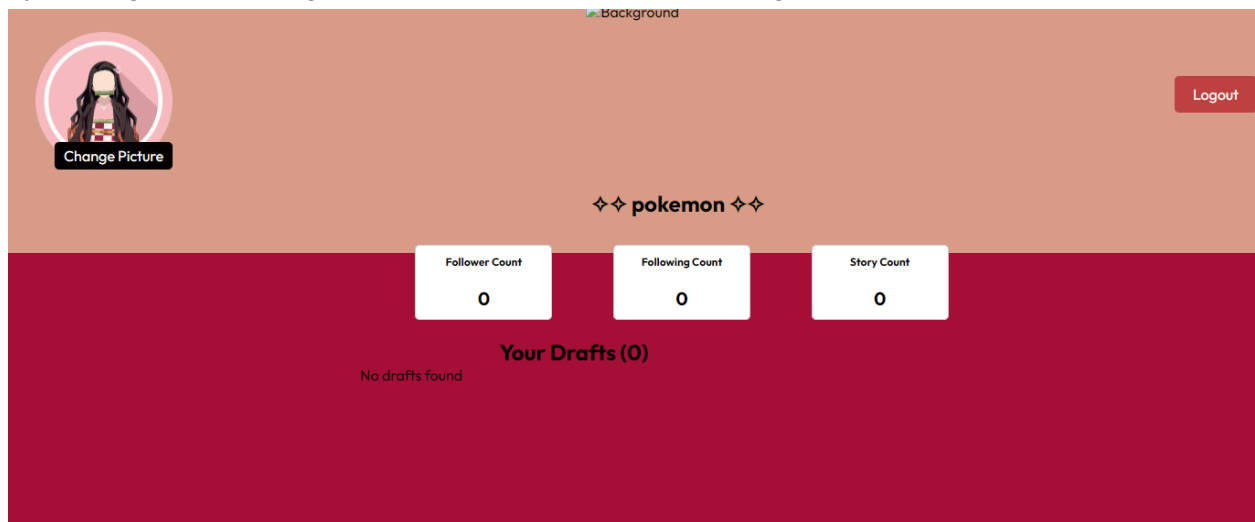
The users can **search the stories** they want to read according to their titles, author names, or categories.

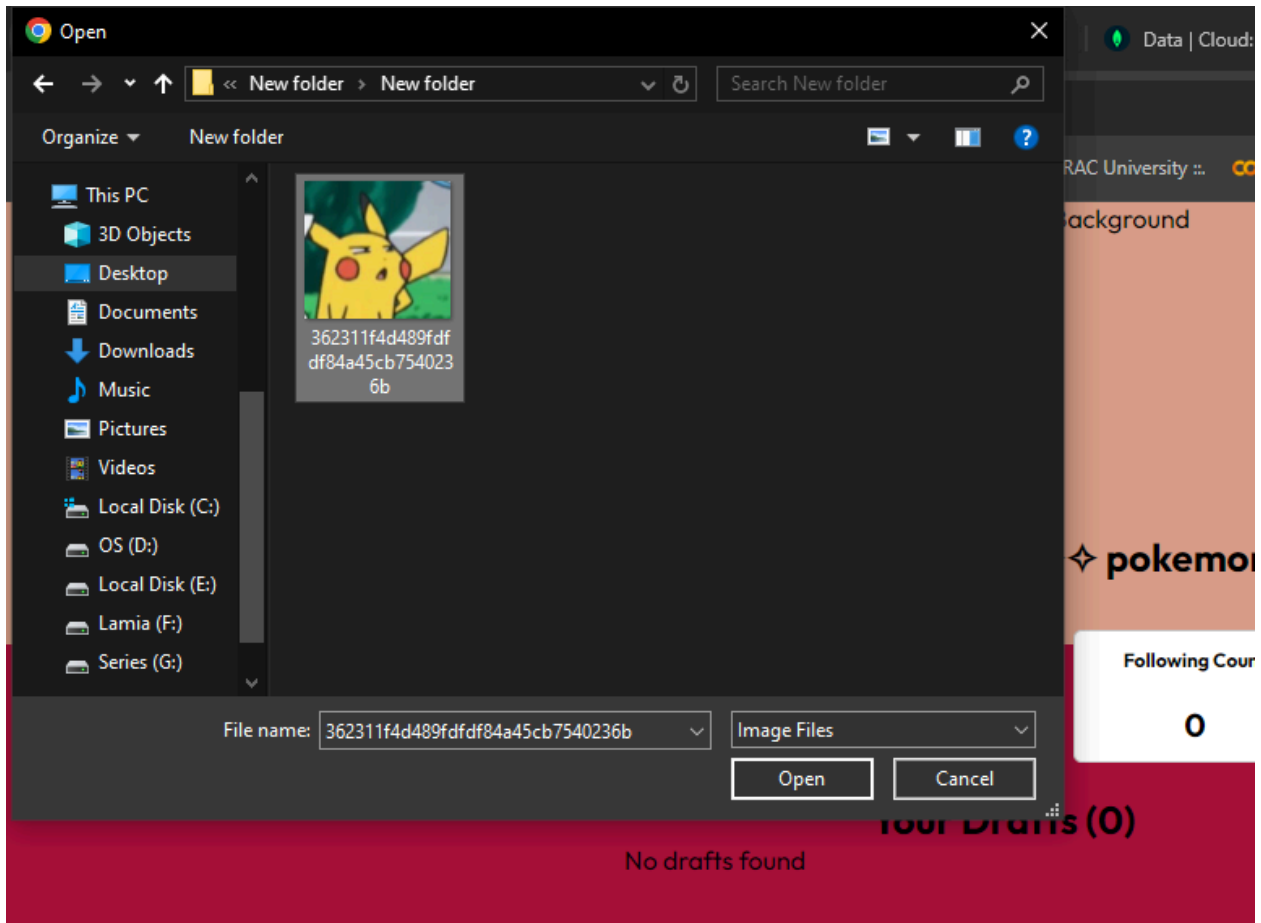


The users can visit their **profile page** by clicking on the icon shown below

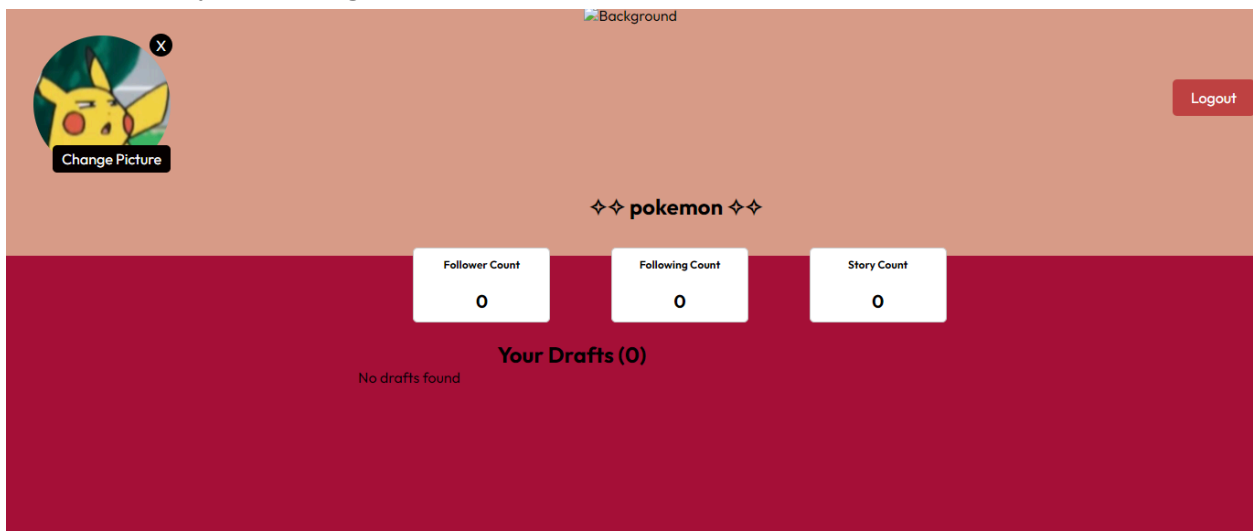


By clicking on the change picture button, the user can change their profile picture.

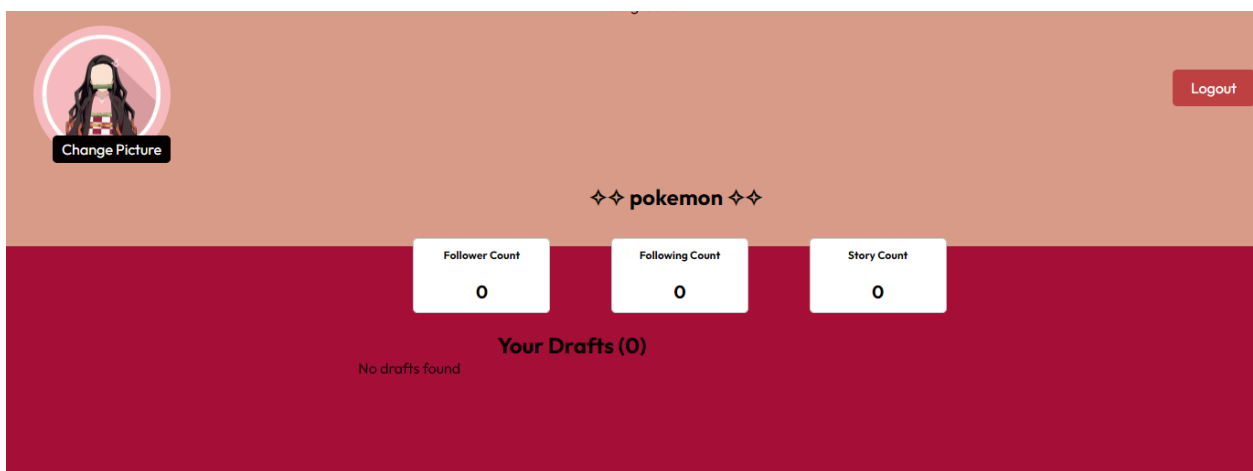
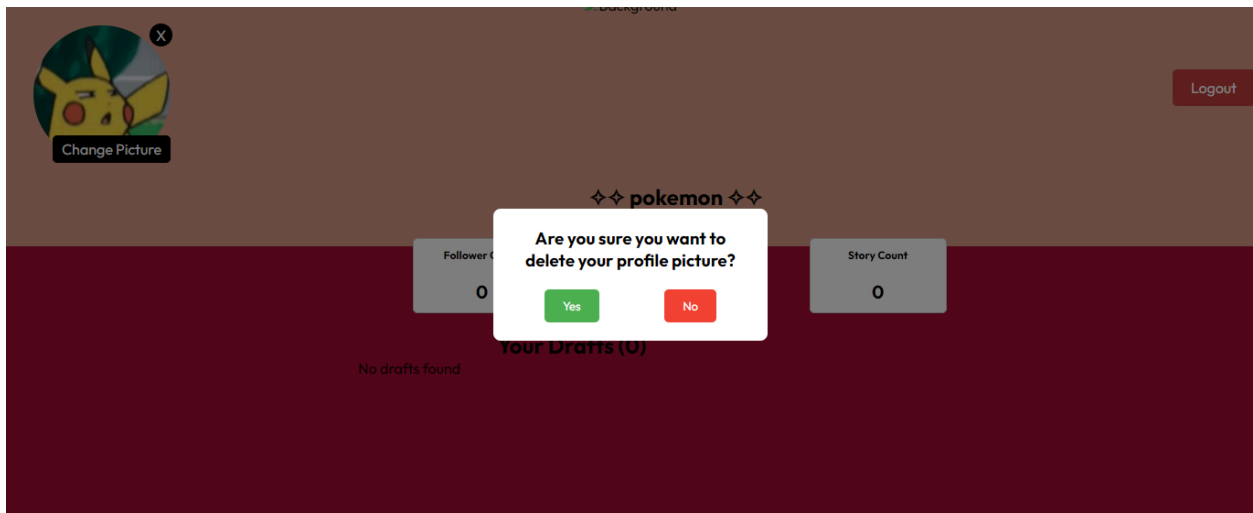




This is how they can **change** their profile picture



By clicking on the cross button, the user can **delete their current profile picture**.



Frontend Development

- Login/signup/authentication

```
frontend > src > components > Login > Login.jsx > Login > handleLoginClick
88  const Login = ({ onClose, onSignupClick, setIsLoggedIn, setName }) => {
89    const history = useHistory();
90    const [email, setEmail] = useState('');
91    const [password, setPassword] = useState('');
92    const [error, setError] = useState('');
93    const [success, setSuccess] = useState('');
94
95    const handleLoginClick = async (e) => {
96      e.preventDefault();
97      try {
98        const response = await axios.post('/api/user/login', { email, password });
99        console.log('Received response:', response.data);
100
101        if (response.status === 200) {
102          localStorage.setItem('token', response.data.token);
103          localStorage.setItem('userId', response.data.userId);
104          localStorage.setItem('userName', response.data.userName);
105
106          setSuccess('Login successful!');
107          setError('');
108          setIsLoggedIn(true);
109          setName(response.data.userName);
110
111          history.push('/home');
112        } else {
113
114
115          setError(response.data.error || 'Login failed. Please try again.');
```



```

    // Redirect to home page
    history.push('/home');
  } else {
    // Handle login failure
    setError(response.data.error || 'Login failed. Please try again.');
```

```

    setSuccess('');
  }
} catch (error) {
  // Handle any other errors (e.g., network or server errors)
  setError('Login failed. Please try again.');
```

```

import React, { useState } from 'react';
import './SignUpForm.css';
import axios from 'axios'

const SignUpForm = ({ onClose, onLoginClick, setUser }) => {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [confirmPassword, setConfirmPassword] = useState('');
  const [error, setError] = useState('');
  const [success, setSuccess] = useState('');

  const handleSignUpClick = async (e) => {
    e.preventDefault();

    if (password !== confirmPassword) {
      setError('Passwords do not match');
      return;
    }

    try {
      const response = await axios.post('/api/user/signup', { name, email, password });
      console.log('Received response:', response.data);

      if (response.status === 201) {
        localStorage.setItem('token', response.data.token);
        setSuccess('Sign up successful!');
        onClose();
      } else {
        setError(response.data.error);
      }
    }
  }

  return (
    <div className="sign-up-form">
      <button className="close-button" onClick={onClose}>X</button>
      <h2>Sign Up</h2>
      <form onSubmit={handleSignUpClick}>
        <div className="form-group">
          <input type="text" placeholder="Name" value={name} onChange={(e) => setName(e.target.value)} />
        </div>
        <div className="form-group">
          <input type="email" placeholder="Email" value={email} onChange={(e) => setEmail(e.target.value)} />
        </div>
        <div className="form-group">
          <input type="password" placeholder="Password" value={password} onChange={(e) => setPassword(e.target.value)} />
          <p className="password-instructions">Password must contain at least one uppercase letter, one lowercase letter, one number, and one special character</p>
        </div>
        <div className="form-group">
          <input type="password" placeholder="Confirm Password" value={confirmPassword} onChange={(e) => setConfirmPassword(e.target.value)} />
        </div>
        {error && <div className="error-message">{error}</div>}
        <button type="submit">Sign Up</button>
      </form>
      <p style={{ marginTop: '10px', textAlign: 'center' }}>
        Already have an account?{' '}
        <span onClick={onLoginClick} style={{ color: '#007bff', textDecoration: 'underline', cursor: 'pointer' }}>Login</span>
      </p>
    </div>
  );
};

```

- Profile picture upload/change/delete

```

7
8  const Profile = ({ setIsLoggedIn }) => {
9    const [imageFile, setImageFile] = useState(null);
10   const [imageFileUrl, setImageFileUrl] = useState(localStorage.getItem('profileImageUrl') || assets.defaultpfp2);
11   const [name, setName] = useState(localStorage.getItem('userName') || '');
12   const [drafts, setDrafts] = useState([]);
13   const [draftCount, setDraftCount] = useState(0);
14   const [showDeleteModal, setShowDeleteModal] = useState(false);
15   const [draftToDelete, setDraftToDelete] = useState(null);
16   const [showProfilePictureDeleteModal, setShowProfilePictureDeleteModal] = useState(false);
17
18   const history = useHistory();
19   const userId = localStorage.getItem('userId');
20
21   useEffect(() => {
22     if (!localStorage.getItem('profileImageUrl') || !localStorage.getItem('userName')) {
23       const fetchData = async () => {
24         try {
25           const response = await axios.get('/api/user/profile', {
26             headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }
27           });
28           const profilePictureUrl = response.data.profilePicture || assets.defaultpfp2;
29           const fetchedName = response.data.name || '';
30

```

```

63   const handleProfilePictureChange = (e) => {
64     const file = e.target.files[0];
65     if (file) {
66       setImageFile(file);
67       const previewUrl = URL.createObjectURL(file);
68       setImageFileUrl(previewUrl);
69     }
70   };
71
72   const handleProfilePictureDeleteClick = () => {
73     setShowProfilePictureDeleteModal(true);
74   };
75
76   const confirmResetProfilePicture = async () => {
77     try {
78       const response = await fetch('/api/user/delete-profile-picture', {
79         method: 'DELETE',
80         headers: {
81           'Content-Type': 'application/json',
82           Authorization: `Bearer ${localStorage.getItem('token')}`
83         },
84         body: JSON.stringify({ userId }),
85       });

```

```

useEffect(() => {
  if (imageFile) {
    uploadImage();
  }
}, [imageFile]);

const uploadImage = async () => {
  const formData = new FormData();
  formData.append('file', imageFile);
  formData.append('userId', userId);

  try {
    const response = await axios.post('http://localhost:5000/upload', formData, {
      headers: { 'Content-Type': 'multipart/form-data' }
    });
    if (response.data.success) {
      const uploadedImageUrl = `http://localhost:5000/public/Images/${response.data.result.image}`;
      setImageFileUrl(uploadedImageUrl);
      localStorage.setItem('profileImageUrl', uploadedImageUrl);
    }
  } catch (error) {
    console.error('Error uploading image:', error);
  }
}

```

- Published/Drafted stories

```

</div>
<div className="story-cards-container">
  {sortedStories.length === 0 ? (
    <p></p>
  ) : (
    sortedStories.map((story) => (
      <div
        key={story._id}
        className="story-card"
        onClick={() => handleStoryClick(story._id)}
      >
        <div
          className="story-card-header"
          dangerouslySetInnerHTML={{ __html: story.topicName || 'No title available' }}
        />
        <p>Author: {story.userId?.name || 'Unknown'}</p>
        <div
          className="story-description"
          dangerouslySetInnerHTML={{ __html: story.description || 'No description available' }}
        />
      </div>
    ))
  )
}

```

```

4
5  const Writing = () => {
6    const [topicName, setTopicName] = useState('');
7    const [description, setDescription] = useState('');
8    const [category, setCategory] = useState('');
9    const [tags, setTags] = useState('');
10   const [language, setLanguage] = useState('');
11   const history = useHistory();
12
13   const handleNext = (e,status) => {
14     e.preventDefault();
15
16     const storyData = {
17       topicName,
18       description,
19       category,
20       tags,
21       language,
22       status
23     };
24
25     history.push('/chapters', { storyData });
26   };

```

```

useEffect(() => {
  const fetchDrafts = async () => {
    try {
      const response = await axios.get('/api/stories/drafts', {
        headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }
      });
      setDrafts(response.data);
      setDraftCount(response.data.length);
    } catch (error) {
      console.error('Error fetching drafts:', error);
    }
  };
  fetchDrafts();
}, []);

```

```

const handleDeleteDraftClick = (id) => {
  setDraftToDelete(id);
  setShowDeleteModal(true);
};

const confirmDeleteDraft = async () => {
  try {
    await axios.delete(`/api/stories/${draftToDelete}`, {
      headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }
    });
    setDrafts(drafts.filter(draft => draft._id !== draftToDelete));
    setDraftCount(draftCount - 1);
    setShowDeleteModal(false);
  } catch (error) {
    console.error('Error deleting draft:', error);
  }
};

const cancelDeleteDraft = () => {
  setDraftToDelete(null);
  setShowDeleteModal(false);
};

const handleEditDraft = (draft) => {
  history.push({
    pathname: '/chapters',
    state: { storyData: draft }
  });
};

return (

```

```

<div className="drafts-section">
  <h2>Your Drafts ({draftCount})</h2>
  <div className="stories-container">
    {drafts.length === 0 ? (
      <p>No drafts found</p>
    ) : (
      drafts.map((draft) => (
        <div key={draft._id} className="story-card">
          <div className="story-card-header">
            <h2>{draft.topicName}</h2>
            <button onClick={() => handleDeleteDraftClick(draft._id)} className="delete-draft-icon">X</button>
            <button className="edit-btn" onClick={() => handleEditDraft(draft)}>Edit</button>
          </div>
          <ReactQuill
            value={draft.description}
            readOnly={true}
            theme="bubble"
          />
        </div>
      ))
    )}
  </div>

```

- Add tags

```

<div className="form-group">
  <label htmlFor="description">Description</label>
  <textarea
    id="description"
    value={description}
    onChange={(e) => setDescription(e.target.value)}
    required
  />
</div>

<div className="form-group">
  <label htmlFor="category">Category</label>
  <select
    id="category"
    value={category}
    onChange={(e) => setCategory(e.target.value)}
    required
  >
    <option value="" disabled>Select Category</option>
    <option value="Action">Action</option>
    <option value="Adventure">Adventure</option>
    <option value="Fanfiction">Fanfiction</option>
    <option value="Fantasy">Fantasy</option>
    <option value="Horror">Horror</option>
    <option value="Humor">Humor</option>
    <option value="Mystery">Mystery</option>
    <option value="Poetry">Poetry</option>
    <option value="Romance">Romance</option>
    <option value="Science fiction">Science fiction</option>
  </select>
</div>

<div className="form-group">
  <label htmlFor="tags">Tags</label>
  <input
    type="text"
  />
</div>

```

```

    <label htmlFor="tags">Tags</label>
    <input
      type="text"
      id="tags"
      value={tags}
      onChange={(e) => setTags(e.target.value)}
      required
    />
  </div>

  <div className="form-group">
    <label htmlFor="language">Language</label>
    <select
      id="language"
      value={language}
      onChange={(e) => setLanguage(e.target.value)}
      required
    >
      <option value="" disabled>Select Language</option>
      <option value="English">English</option>
      <option value="Bangla">Bangla</option>
    </select>
  </div>

  <div className="button-group">
    <button type="submit" className="submit-btn">
      Next
    </button>
    <button type="button" className="cancel-btn" onClick={() => history.push('/')}>
      Cancel
    </button>
  </div>
</form>
</div>
);
};

```


- Modify/save stories

```
const confirmDeleteDraft = async () => {
  try {
    await axios.delete(`/api/stories/${draftToDelete}`, {
      headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }
    });
    setDrafts(drafts.filter(draft => draft._id !== draftToDelete));
    setDraftCount(draftCount - 1);
    setShowDeleteModal(false);
  } catch (error) {
    console.error('Error deleting draft:', error);
  }
};
```

```
const cancelDeleteDraft = () => {
  setDraftToDelete(null);
  setShowDeleteModal(false);
};
const handleEditDraft = (draft) => {
  history.push({
    pathname: '/chapters',
    state: { storyData: draft }
  });
};
return (
  <div className="profile-container">
```

```
</div>
<div className="drafts-section">
  <h2>Your Drafts ({draftCount})</h2>
  <div className="stories-container">
    {drafts.length === 0 ? (
      <p>No drafts found</p>
    ) : (
      drafts.map((draft) => (
        <div key={draft._id} className="story-card">
          <div className="story-card-header">
            <ReactQuill
              value={draft.topicName}
              readOnly={true}
              theme="bubble"
            />
            <button onClick={() => handleDeleteDraftClick(draft._id)} className="delete-draft-icon">X</button>
            <button className="edit-btn" onClick={() => handleEditDraft(draft)}>Edit</button>
          </div>
            <ReactQuill
              value={draft.description}
              readOnly={true}
              theme="bubble"
            />
          </div>
        </div>
      ))
    )}
  </div>
</div>
{showDeleteModal && (
  <div className="modal">
```

- Changeable title/description styles

```
const Chapters = () => {
  const location = useLocation();
  const history = useHistory();
  const { storyData } = location.state || {};

  const [chapters, setChapters] = useState(storyData?.chapters || '');
  const [styledTitle, setStyledTitle] = useState(storyData?.topicName || '');
  const [styledDescription, setStyledDescription] = useState(storyData?.description || '');

  // Toolbar options for title and description
  const titleModules = { toolbar: ['bold', 'italic', 'underline', 'clean'] };
  const descriptionModules = { toolbar: ['bold', 'italic', 'underline', 'clean'] };
}
```

```
return (
  <div className="chapter-writing-container">
    <h1>Write Your Chapter</h1>
    <form className="chapter-writing-form">
      <div className="form-group">
        <label htmlFor="styledTitle">Title</label>
        <ReactQuill
          id="styledTitle"
          theme="snow"
          value={styledTitle}
          onChange={setStyledTitle}
          modules={titleModules}
        />
      </div>

      <div className="form-group">
        <label htmlFor="styledDescription">Description</label>
        <ReactQuill
          id="styledDescription"
          theme="snow"
          value={styledDescription}
          onChange={setStyledDescription}
          modules={descriptionModules}
        />
      </div>
    </form>
  </div>
)
```

- Search functionality

```
const handleSearch = (e) => {
  setSearchQuery(e.target.value);
};
```

```

</div>
<div className="welcome-section">
  <h1>Welcome to SOYO</h1>
  <p>Story Of Your Own</p>
  <div className="button-container">
    <button className="btn" onClick={handleWritingClick}>Start Writing</button>
    <input
      type="text"
      placeholder="Search stories..."
      className="search-input"
      value={searchQuery}
      onChange={handleSearch}
    />
  </div>
</div>

```

```

const filteredStories = stories.filter((story) => {
  const lowerQuery = searchQuery.toLowerCase();
  return (
    story.topicName.toLowerCase().includes(lowerQuery) ||
    story.category.toLowerCase().includes(lowerQuery) ||
    (story.userId?.name && story.userId.name.toLowerCase().includes(lowerQuery))
  );
});

```

- Sorting stories

```

8  const Home = ({ isLoggedIn, setIsLoggedIn }) => {
81
82    const stripHtml = (html) => {
83      const doc = new DOMParser().parseFromString(html, 'text/html');
84      return doc.body.textContent || '';
85    };
86
87    const sortStories = (stories, criteria, order) => {
88      return stories.sort((a, b) => {
89        let valueA, valueB;
90        if (criteria === 'title') {
91          valueA = stripHtml(a.topicName || '').toLowerCase();
92          valueB = stripHtml(b.topicName || '').toLowerCase();
93        } else if (criteria === 'createdAt') {
94          valueA = new Date(a.createdAt);
95          valueB = new Date(b.createdAt);
96        }
97
98        if (order === 'asc') {
99          return valueA < valueB ? -1 : valueA > valueB ? 1 : 0;
100        } else {
101          return valueA > valueB ? -1 : valueA < valueB ? 1 : 0;
102        }
103      });

```

```

const filteredStories = stories.filter((story) => {
  const lowerQuery = searchQuery.toLowerCase();
  return (
    story.topicName.toLowerCase().includes(lowerQuery) ||
    story.category.toLowerCase().includes(lowerQuery) ||
    (story.userId?.name && story.userId.name.toLowerCase().includes(lowerQuery))
  );
});

const sortedStories = sortStories(filteredStories, sortCriteria, sortOrder);

```

```

</div>
<div className="sorting-container">
  <select value={sortCriteria} onChange={handleSortCriteriaChange}>
    <option value="title">Sort by Title</option>
    <option value="createdAt">Sort by Time Created</option>
  </select>
  <select value={sortOrder} onChange={handleSortOrderChange}>
    <option value="asc">Ascending</option>
    <option value="desc">Descending</option>
  </select>
</div>
</div>

```

Translation(json files and codes)

```
App.jsx bn.json SignUpForm.jsx JS userRoutes.js
frontend > src > locales > bn.json > ...
1  {
2    "welcome": "স্বাগতম",
3    "greetings": "এখানেই আপনার সৃজনশীলতাকে মুক্তি দিন",
4    "login": "লগইন",
5    "logout": "লগআউট",
6    "signup": "নিবন্ধন করুন",
7    "home": "হোম",
8    "Menu": "মেনু",
9    "Contact Us": "যোগাযোগ করুন",
10   "profile": "প্রোফাইল",
11   "writing": "লিখা শুরু করুন",
12   "login_success": "লগইন সফল",
13
14   "createStory": "নতুন গল্প তৈরি করুন",
15   "TopicName": "বিষয়ের নাম",
16   "Description": "বর্ণনা",
17   "Category": "শ্রেণী",
18   "Action": "অ্যাকশন",
19   "Adventure": "অ্যাডভেঞ্চার",
20   "Fanfiction": "ফ্যানফিকশন",
21   "Fantasy": "ফ্যান্টাসি",
22   "Horror": "হরর",
23   "Humor": "হাস্যরস",
24   "Mystery": "রহস্য",
25   "Poetry": "কবিতা",
26   "Romance": "রোমান্স",
27   "Science Fiction": "বিজ্ঞান কল্পকাহিনী",
28   "Tags": "ট্যাগসমূহ",
29   "Language": "ভাষা",
30   "English": "ইংরেজি",
31   "Bangla": "বাংলা",
32
33   "Writechapter": "আপনার অধ্যায় লিখুন",
34   "title": "শিরোনাম",
35   "ChapterContent": "অধ্যায়ের বিষয়বস্তু",
36
37   "SaveDraft": "খসড়া হিসাবে সংরক্ষণ করুন",
```

```
App.jsx bn.json x SignUpForm.jsx JS userRoutes.js
ntend > src > locales > bn.json > ...
5 "ChapterContent": "অধ্যায়ের বিষয়বস্তু",
6
7 "SaveDraft": "খসড়া হিসাবে সংরক্ষণ করুন",
8 "Publish": "প্রকাশ করুন",
9 "Cancel": "বাতিল করুন",
10
11 "1": {
12   "topicName": "কান্না নয়",
13   "description": "জীবন আমার কান্না। তাই জীবন মানেই কান্না",
14   "chapters": "*কান্নার শব্দ বাড়ছে*"
15 },
16 "2": {
17   "topicName": "মিউ মিউ",
18   "description": "মিউ মিউ মিউ মিউ মিউ মিউ মিউ মিউ",
19   "chapters": ""
20 },
21 "3": {
22   "topicName": "দুঃস্বপ্ন",
23   "description": "অনেক ভয়ের একটি কাহিনী",
24   "chapters": "একটি সাদা পোশাক পরা মহিলা প্রাসাদের করিডোরে চলাফেরা করছে..."
25 },
26 "4": {
27   "topicName": "দুঃস্বপ্ন",
28   "description": "অনেক ভয়ের একটি কাহিনী",
29   "chapters": "একটি সাদা পোশাক পরা মহিলা প্রাসাদের করিডোরে চলাফেরা করছে..."
30 },
31 "5": {
32   "topicName": "দুঃস্বপ্ন",
33   "description": "অনেক ভয়ের একটি কাহিনী",
34   "chapters": "যা লিখেছি তা দেখানোর কথা ছিল, সংরক্ষিত থাকার কথা। এখন..."
35 },
36 "6": {
37   "topicName": "একটি গল্প",
38   "description": "df",
39   "chapters": "প্রকাশিত?"
40 },
41 "7": {
```

```

},
"7": {
  "topicName": "একটি গল্প",
  "description": "df",
  "chapters": "প্রকাশিত?"
},
"8": {
  "topicName": "হাই",
  "description": "hkjhgv",
  "chapters": "okokokokokol"
},
"9": {
  "topicName": "হাই",
  "description": "hkjhgv",
  "chapters": "okokokokokol"
},
"10": {
  "topicName": "আমার জীবনের গল্প",
  "description": "মারা খাই",
  "chapters": "মৃত"
},
"11": {
  "topicName": "আমার জীবনের গল্প",
  "description": "মারা খাই",
  "chapters": "মৃত"
},
"12": {
  "topicName": "স্বপোর একটা স্টুপিড",
  "description": "খুব খারাপ বন্ধু (মজা করছি)",
  "chapters": "সে গ্রিন রোডে থাকে। সে একমাত্র সন্তান। সে একজন ভালো বন্ধু..."
},
"13": {
  "topicName": "উবরে এক সফর",
  "description": "শিক্ষার্থীদের জীবন, যেখানে ক্লাসের তিন ঘন্টা আগে প্রস্তুতি নিতে হয়...",
  "chapters": "অনেকেই ক্লাসের তিন ঘন্টা আগে প্রস্তুত হতে হয়..."
},

```

```

    },
    "14": {
      "topicName": "উবরে এক সফর",
      "description": "শিক্ষার্থীদের জীবন, যেখানে ক্লাসের তিন ঘন্টা আগে প্রস্তুতি নিতে হয়...",
      "chapters": "অনেকেই ক্লাসের তিন ঘন্টা আগে প্রস্তুত হতে হয়..."
    },
    "15": {
      "topicName": "কেউ জানে না",
      "description": "হেহহিহিহি হেহহে হাহাহা",
      "chapters": "দয়া করে আমাকে কিছু বলুন। মানুষ এসব কীভাবে ম্যানেজ করে?!! আমার..."
    },
    "16": {
      "topicName": "হারি পুটার",
      "description": "জাদুকরী জগৎ",
      "chapters": "নাস্তায় একটি তর্ক হয়েছিল..."
    },
    "17": {
      "topicName": "হারি পটার এবং চেম্বার অফ সিক্রেটস",
      "description": "ডারসলিরা হারির সাথে খারাপ আচরণ করেছিল",
      "chapters": "ডারসলিরা ভুলেও আজকের দিনটি যে হারির জন্মদিন তা মনে রাখেনি..."
    },
    "18": {
      "topicName": "ডেমো",
      "description": "ডেমো",
      "chapters": ""
    },
    "19": {
      "topicName": "fbirebf",
      "description": "ndsjvbfdbhfr",
      "chapters": "blaaaaaaaa"
    },
    "20": {
      "topicName": "পার্টেসিনা",
      "description": "সাহায্য করুন",
      "chapters": ""
    }
  }
}

```

```

    },
    "21": {
      "topicName": "সাম কে হত্যা করা",
      "description": "আমি আমার সেরা বন্ধুকে হত্যা করার চেষ্টা করেছিলাম।",
      "chapters": "অধ্যায় ১: এটি ২০১৪ সালে শুরু হয়েছিল।"
    },
    "loading": "লোড হচ্ছে...",
    "storyerror": "গল্প আনতে ত্রুটি",
    "untitled_story": "শিরোনামহীন গল্প",
    "no_description_available": "কোন বর্ণনা নেই",
    "no_chapter_available": "কোন অধ্যায় নেই",
    "story_not_found": "গল্প পাওয়া যায়নি"
  }
}

```



```
Run terminal Help > CSE471_merged
App.jsx {} en.json x SignUpForm.jsx JS userRoutes.js
ontend > src > locales > {} en.json > ...
1 {
2   "welcome": "Welcome to SOYO",
3   "greetings": "This is where you unleash your creativity",
4   "login": "Login",
5   "logout": "Logout",
6   "signup": "Sign Up",
7   "home": "Home",
8   "Menu": "menu",
9   "Contact Us": "contact_us",
10  "profile": "Profile",
11  "writing": "Start Writing",
12  "login_success": "log in success",
13
14  "createStory": "Create a New Story",
15  "TopicName": "Topic Name",
16  "Description": "Description",
17  "Category": "Category",
18    "Action": "Action",
19    "Adventure": "Adventure",
20    "Fanfiction": "Fanfiction",
21    "Fantasy": "Fantasy",
22    "Horror": "Horror",
23    "Humor": "Humor",
24    "Mystery": "Mystery",
25    "Poetry": "Poetry",
26    "Romance": "Romance",
27    "Science Fiction": "Science Fiction",
28  "Tags": "Tags",
29  "Language": "Language",
30    "English": "English",
31    "Bangla": "Bangla",
32
33  "Writechapter": "Write your chapter",
34  "title": "Title",
35  "ChapterContent": "Chapter Content",
36
37  "SaveDraft": "Save as Draft",
```

```
App.jsx  enjson  SignUpForm.jsx  JS userRoutes.js
frontend > src > locales > enjson > ...
35 "ChapterContent": "Chapter Content",
36
37 "SaveDraft": "Save as Draft",
38 "Publish": "Publish",
39 "Cancel": "Cancel",
40
41
42 "1": {
43   "topicName": "Save no tears",
44   "description": "jibon amar kanna. so jibon manei kanna",
45   "chapters": "**Sobbing noise intensified**"
46 },
47 "2": {
48   "topicName": "Mew mew",
49   "description": "mew mew mew mew mew mew mew mew",
50   "chapters": ""
51 },
52 "3": {
53   "topicName": "Nightmare",
54   "description": "onek bhoi er ekta kahini",
55   "chapters": "A woman wearing a white gown is gliding in the hallways of the palace..."
56 },
57 "4": {
58   "topicName": "Nightmare",
59   "description": "onek bhoi er ekta kahini",
60   "chapters": "A woman wearing a white gown is gliding in the hallways of the palace..."
61 },
62 "5": {
63   "topicName": "Nightmare",
64   "description": "onek bhoi er ekta kahini",
65   "chapters": "blabla jeta likhsi sheta show korar kotha as saved thakar kotha. Ekhon..."
66 },
67 "6": {
68   "topicName": "a storyas",
69   "description": "df",
70   "chapters": "published?"
71 },
```

```
App.jsx  enjson  SignUpForm.jsx  JS userRoutes.js
frontend > src > locales > enjson > ...
67 "6": {
68   "topicName": "a storyas",
69   "description": "df",
70   "chapters": "published?"
71 },
72 "7": {
73   "topicName": "a storyas",
74   "description": "df",
75   "chapters": "published?"
76 },
77 "8": {
78   "topicName": "hii",
79   "description": "hkjhgv",
80   "chapters": "okokokokokol"
81 },
82 "9": {
83   "topicName": "hii",
84   "description": "hkjhgv",
85   "chapters": "okokokokokol"
86 },
87 "10": {
88   "topicName": "story of my life",
89   "description": "mara khai",
90   "chapters": "ded"
91 },
92 "11": {
93   "topicName": "story of my life",
94   "description": "mara khai",
95   "chapters": "ded"
96 },
97 "12": {
98   "topicName": "SWOPER EKTA STUPID",
99   "description": "Very bad friend (JK)",
100   "chapters": "She lives in Green road. She is the only child. She is a good friend b..."
101 },
102 "13": {
103   "topicName": "A DRIVE IN UBER",
104   "description": "Student's life where they need to get ready 3 hours bef...",
105   "chapters": "The fact that many people have to get ready 3 hours before the class s..."
106 },
```

```
un Terminal Help ← → cse471_merged
enjson x SignUpForm.jsx JS userRoutes.js
nd > src > locales > enjson > {} > chapters
},
"14": {
  "topicName": "A DRIVE IN UBER",
  "description": "Student's life where they need to get ready 3 hours before the clas...",
  "chapters": "The fact that many people have to get ready 3 hours before the class s..."
},
"15": {
  "topicName": "nobody knows",
  "description": "hehheeh ee heeh hahaha",
  "chapters": "Please tell me something. How do people manage all these things?!! My ..."
},
"16": {
  "topicName": "Harry puttar",
  "description": "magical world",
  "chapters": "Not for the first time an argument had broken out over breakfast at nu..."
},
"17": {
  "topicName": "Harry Potter and the chamber of secrets",
  "description": "The Dursleys were mean to Harry",
  "chapters": "The Dursleys hadn't even remembered that today happened to be Harry's..."
},
"18": {
  "topicName": "demo",
  "description": "demo",
  "chapters": ""
},
"19": {
  "topicName": "fbirebf",
  "description": "ndsjvbhfdhbfr",
  "chapters": "blaaaaaaaa"
},
"20": {
  "topicName": "partesinaaa",
  "description": "HELPPP",
  "chapters": ""
},
"21": {
```

```
    "chapters": ""
  },
  "21": {
    "topicName": "Killing Sam",
    "description": "I tried to kill my best friend. ",
    "chapters": "Chapter 1: It all started back in 2014."
  },

  "loading": "Loading...",
  "storyerror": "error fetching story",
  "untitled_story": "untitled story",
  "no_description_available": "no description available",
  "no_chapter_available": "no chapter available",
  "story_not_found": "story not found"
}
```

```
App.jsx JS i18n.js X SignUpForm.jsx JS userRoutes.js
frontend > src > JS i18n.js > ...
1 // src/i18n.js
2 import i18n from 'i18next';
3 import { initReactI18next } from 'react-i18next';
4 import LanguageDetector from 'i18next-browser-languagedetector';
5 import translationEN from './locales/en.json';
6 import translationBN from './locales/bn.json';
7
8
9 const resources = {
10   en: {
11     translation: translationEN,
12   },
13   bn: {
14     translation: translationBN,
15   },
16 };
17
18 i18n
19   .use(LanguageDetector)
20   .use(initReactI18next)
21   .init({
22     resources,
23     lng: 'en', |
24     fallbackLng: 'en',
25     interpolation: {
26       escapeValue: false,
27     },
28   });
29
30 export default i18n;
31
```

translation.js

```
App.jsx JS i18n.js JS translation.js X SignUpForm.jsx JS userRoutes.js
frontend > src > JS translation.js > ...
import React from 'react';
import ReactDOM from 'react-dom/client';
import './i18n.js'; // Import i18n here to initialize it globally

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Navbar.jsx

```

<div className="language-switcher">
  <button onClick={() => changeLanguage('en')}>English</button>
  <button onClick={() => changeLanguage('bn')}>বাংলা</button>
</div>

```

Recaptcha

```

App.jsx  JS i18n.js  SignUpForm.jsx  JS userController.js  Navbar.jsx  JS userRoutes.js
frontend > src > components > SignUpForm > SignUpForm.jsx > [e] SignUpForm > [e] handleSignUpClick
46
47 import React, { useState, useRef } from 'react'; //userref
48 import { useNavigate } from 'react-router-dom';//
49 import './SignUpForm.css';
50 import axios from 'axios'
51 import ReCAPTCHA from 'react-google-recaptcha';//
52
53 const SITE_KEY = '6LfWnkQqAAAAAL06jAkNTRQny1kYGW1A0V_TyRQH'; //
54
55 const SignUpForm = ({ onClose, onLoginClick, setUser, setIsLoggedIn }) => {
56
57   const navigate= useNavigate();//
58
59   const [name, setName] = useState('');
60   const [email, setEmail] = useState('');
61   const [password, setPassword] = useState('');
62   const [confirmPassword, setConfirmPassword] = useState('');
63   const [captchaValue, setCaptchaValue] = useState(null);//
64   const [error, setError] = useState('');
65   const [success, setSuccess] = useState('');
66   const [recaptchaValue, setRecaptchaValue] = useState('');//
67
68   const captchaRef = useRef();//
69
70
71   //
72   const handleSignUpClick = async (e) => {
73     e.preventDefault();
74     if (password !== confirmPassword) {
75       setError('Passwords do not match');
76       captchaRef.current.reset();
77       return;
78     }
79
80     if (!recaptchaValue) {
81       setError('Please complete the CAPTCHA');
82       return;

```

```

    if (!recaptchaValue) {
      setError('Please complete the CAPTCHA');
      return;
    }

    try {
      const response = await axios.post('/api/user/signup', { name, email, password, recaptchaValue });
      console.log('Signup response:', response.data);

      if (response.status === 201) {
        localStorage.setItem('token', response.data.token); //
        setSuccess('Sign up successful!'); //
        setError(''); //
        setIsLoggedIn(true); //
        setUser(response.data.user.name); //
        navigate('/Home');
        onClose(); //
      } else {
        console.log('Error response data:', response.data);
        setError(response.data.error);
      }
    }
    catch (error) {
      console.error('Signup error response:', error.response?.data || error);
      setError(error.response?.data?.error);
    }
    finally {
      captchaRef.current.reset();
    }
  }
}

const onChange = (value) => {

```

```

const onChange = (value) => {
  console.log('Captcha value:', value);
  setRecaptchaValue(value); //string backend a pathabe, backend google er shathe a secret key pathabe and verification a true ashbe tahole recaptc
};

return (
  <div className="sign-up-form">
    <button className="close-button" onClick={onClose}>X</button>
    <h2>Sign Up</h2>
    <form onSubmit={handleSignUpClick}>
      <div className="form-group">
        <input type="text" placeholder="Name" value={name} onChange={(e) => setName(e.target.value)} />
      </div>
      <div className="form-group">
        <input type="email" placeholder="Email" value={email} onChange={(e) => setEmail(e.target.value)} />
      </div>
      <div className="form-group">
        <input type="password" placeholder="Password" value={password} onChange={(e) => setPassword(e.target.value)} />
        <p className="password-instructions">Password must contain at least one uppercase letter, one lowercase letter, one number, and one special character</p>
      </div>
      <div className="form-group">
        <input type="password" placeholder="Confirm Password" value={confirmPassword} onChange={(e) => setConfirmPassword(e.target.value)} />
      </div>
      <div className="form-group mt-2">
        <ReCAPTCHA
          sitekey={SITE_KEY}
          onChange={onChange}
          ref={captchaRef}
          size="compact" />
      </div>
      {error && <div className="error-message">{error}</div>}
      <button type="submit">Sign Up</button>
    </form>
  </div>
);

```

Backend Development

- Login/signup
- Profile picture upload/change/delete

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

const userImageSchema = new Schema({
  userId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true
  },
  image: {
    type: String,
    required: true
  }
});

module.exports = mongoose.model('UserImage', userImageSchema);
```

```
22 const storage = multer.diskStorage({
23   destination: (req, file, cb) => {
24     cb(null, 'public/Images');
25   },
26   filename: (req, file, cb) => {
27     cb(null, `${file.fieldname}_${Date.now()}${path.extname(file.originalname)}`);
28   }
29 });
30
31 const upload = multer({ storage: storage });
32
33 app.post('/upload', upload.single('file'), (req, res) => {
34   const { userId } = req.body;
35
36   console.log('File received:', req.file);
37   if (!userId) {
38     return res.status(400).json({ success: false, error: 'User ID is required' });
39   }
40
41   const imageUrl = `/public/Images/${req.file.filename}`;
42
43   UserImage.create({ userId, image: req.file.filename })
44     .then(result => res.json({ success: true, result }))
45     .catch(err => {
46       console.log(err);
47       res.status(500).json({ success: false, error: 'Database error' });
48     });
49 });
```

- Published/Drafted stories

```
// POST a new story (Draft or Publish)
router.post('/', async (req, res) => {
  const { topicName, description, category, tags, language, chapters, status, _id } = req.body;

  if (!topicName || !description || !category || !tags || !language || !status) {
    return res.status(400).json({ error: 'All fields are required' });
  }

  try {
    const storyData = {
      userId: req.user._id,
      topicName,
      description,
      category,
      tags,
      language,
      chapters: chapters || '', // Use chapters field
      status,
      author: req.user._id
    };

    let story;
    if (_id) {
      // Update an existing story (either draft or publish)
      story = await Story.findByIdAndUpdate(_id, storyData, { new: true, runValidators: true });
      return res.status(200).json({ message: 'Story updated successfully', story });
    } else {
      // Create a new story (either draft or publish)
      story = await Story.create(storyData);
      return res.status(201).json({ message: 'Story created successfully', story });
    }
  } catch (error) {
    console.error('Error creating or updating story:', error);
    res.status(500).json({ error: 'Server error, please try again later' });
  }
});
```


- Modify/save stories

```
router.get('/published', async (req, res) => {
  const { searchQuery } = req.query;
  const searchRegex = new RegExp(searchQuery, 'i');

  try {
    let query = { status: 'published' };

    if (searchQuery) {
      query = {
        status: 'published',
        $or: [
          { topicName: searchRegex },
          { category: searchRegex },
          { tags: searchRegex }
        ]
      };
    }

    const stories = await Story.find(query).populate('userId', 'name');
    const filteredStories = stories.filter(story => story.userId.name.match(searchRegex));

    res.status(200).json(filteredStories);
  } catch (error) {
    console.error('Error fetching published stories:', error);
    res.status(500).json({ error: 'Failed to fetch stories' });
  }
});

router.get('/drafts', async (req, res) => {
  try {
    const drafts = await Story.find({ userId: req.user._id, status: 'draft' });
    res.status(200).json(drafts);
  } catch (error) {
    console.error('Error fetching drafts:', error);
    res.status(500).json({ error: 'Failed to fetch drafts' });
  }
});
});
```

- Add tags

```
4 const router = express.Router();
5
6 router.use(requireAuth);
7
8 router.post('/', async (req, res) => {
9   const { topicName, description, category, tags, language, chapters, status, _id } = req.body;
10
11   if (!topicName || !description || !category || !tags || !language || !status) {
12     return res.status(400).json({ error: 'All fields are required' });
13   }
14
15   try {
16     const storyData = {
17       userId: req.user._id,
18       topicName,
19       description,
20       category,
21       tags,
22       language,
23       chapters: chapters || '',
24       status,
25       author: req.user._id
26     };
27   }
28 }
```

- Changeable title/description styles

```
// GET a specific story by ID
router.get('/:id', async (req, res) => {
  try {
    const story = await Story.findById(req.params.id);
    if (!story) {
      return res.status(404).json({ error: 'Story not found' });
    }
    res.status(200).json(story);
  } catch (error) {
    console.error('Error fetching story:', error);
    res.status(500).json({ error: 'Failed to fetch story' });
  }
});
```

- Search functionality

```
3 router.get('/published', async (req, res) => {
4   const { searchQuery } = req.query;
5   const searchRegex = new RegExp(searchQuery, 'i');
6
7   try {
8     let query = { status: 'published' };
9
10    if (searchQuery) {
11      query = {
12        status: 'published',
13        $or: [
14          { topicName: searchRegex },
15          { category: searchRegex },
16          { tags: searchRegex }
17        ]
18      };
19    }
20
21    const stories = await Story.find(query).populate('userId', 'name');
22    const filteredStories = stories.filter(story => story.userId.name.match(searchRegex));
23
24    res.status(200).json(filteredStories);
25  } catch (error) {
26    console.error('Error fetching published stories:', error);
27    res.status(500).json({ error: 'Failed to fetch stories' });
28  }
29 });
```

- Sorting stories

```
// GET a specific story by ID
router.get('/:id', async (req, res) => {
  try {
    const story = await Story.findById(req.params.id);
    if (!story) {
      return res.status(404).json({ error: 'Story not found' });
    }
    res.status(200).json(story);
  } catch (error) {
    console.error('Error fetching story:', error);
    res.status(500).json({ error: 'Failed to fetch story' });
  }
});
```

Recaptcha

```
2
3 import User from '../models/userModel.js';
4 import jwt from 'jsonwebtoken';
5 import Story from '../models/storyModel.js';
6 import axios from 'axios';
7
8
9 const RECAPTCHA_SECRET_KEY = process.env.RECAPTCHA_SECRET_KEY;
10
11 // Create JWT token
12 const createToken = (_id) => {
13   return jwt.sign({ _id }, process.env.SECRET, { expiresIn: '3d' }); // Expires in 3 days
14 };
15
```

```

export const signupUser = async (req, res) => {
  const { name, email, password, recaptchaValue } = req.body;

  if (!recaptchaValue) {
    return res.status(400).json({ error: "ReCAPTCHA token is missing" });
  }

  try {
    //reCAPTCHA verification token with Google's API
    const recaptchaResponse = await axios.post(
      `https://www.google.com/recaptcha/api/siteverify`,
      null,
      {
        params: {
          secret: process.env.RECAPTCHA_SECRET_KEY,
          response: recaptchaValue
        }
      }
    );

    if (!recaptchaResponse.data.success) {
      console.error('reCAPTCHA verification error:', recaptchaResponse.data['error-codes']);
      return res.status(400).json({
        error: "ReCAPTCHA verification failed",
        details: recaptchaResponse.data['error-codes']
      });
    }

    // Proceed with user signup
    const user = await User.signup(name, email, password);
    const token = createToken(user._id);

    res.status(201).json({ email, user, token });
  } catch (error) {

```

Technology (Framework, Languages)

Framework: MERN stack

Languages: Javascript, CSS

GitHub Repository

Link: <https://github.com/Lamiaaaaaaaaaa/CSE471-Project.git>

Individual Contribution:

ID	Name	Contribution
21301062	Jannatul Ferdous	Profile picture upload/change/delete, changeable title/description styles, publish stories, sorting stories
21301191	Annonna Dev Nipa	login/signup, change/save/delete drafts,
21301105	Tangena Islam	Modify/save stories, add tags, search functionalities
21301196	Swachcha Podder	Authentication/Recaptcha Translation using i18next & .json file