



BIT21503 – WEB DEVELOPMENT

Group 11

Learn-Hub a Tutorial Web App

Name	Metric No
ABDULLAHI MOHAMED ABDULLAHI	BI220050
ABDIKARIN OSMAN MOHAMED	BI220042
ABOBAKR EISA OMER SULIMAN	BI210052
HUSSEIN AHMED SALEH BAHLOL	BI210022
ZAKARIYE SAHAL ABDI JAMA	AI200329
MUHAMMAD RAFI RIDWANSYAH	AI210033
AHMAD SYAHRUL HADI SAN	AI210043
Section	01
GitHub Repo	https://github.com/Abdallemo/Web-Dev-Final-Project-UTHM.git
Website Link	https://learn-hub-nine.vercel.app

Abstract

This project aimed to develop a user-friendly website tailored for developers to share tutorials on programming languages and various technical subjects using Markdown, akin to GitHub README files. The primary objective was to create a platform where users could effortlessly create and access tutorials, fostering a community of knowledge sharing.

The development process involved leveraging Node.js for the server-side operations and Express.js for the frontend, ensuring a seamless, responsive, and secure user experience. Key methodologies included user-centric design principles, agile development practices, and thorough testing to guarantee functionality and performance.

Main findings from the project highlighted the effectiveness of using modern web technologies to create an intuitive and efficient platform. The website supports user authentication, allowing individuals to register, log in, and contribute tutorials. Additionally, the site offers features for managing tutorials and submitting reviews with media uploads, enhancing the overall user experience.

In conclusion, the project successfully delivered a stable, secure, and fast platform where users can join, create accounts, and contribute tutorials. This website not only meets the initial objectives but also provides a robust foundation for future enhancements and community growth.

Chapter 1: Introduction

1.1 Welcome to Our Tutorial Website Project

Welcome to our Tutorial Website project! This project aims to create a platform where users can share tutorials on various topics, such as programming, cooking, DIY projects, and more. The website provides a space where users can sign up, log in, submit tutorials, and view tutorials submitted by others. One of the key features is the ability for users to upload their tutorials in Markdown format, making it easy to format and style their content. Markdown is a lightweight markup language with plain text formatting syntax, making it ideal for writing tutorials that include sample code, images, and other elements.

1.2 Objectives

The primary objectives of this project are as follows:

1. **Create a User-Friendly Platform:** Develop a website that is intuitive and easy to navigate, allowing users of all skill levels to share and access tutorials.
2. **Support Markdown Formatting:** Enable users to submit tutorials in Markdown format, providing them with a simple way to format their content, include code samples, images, and more.
3. **Ensure Secure User Authentication:** Implement a robust authentication system allowing users to securely sign up, log in, and manage their tutorials.
4. **Facilitate Knowledge Sharing:** Foster a community where users can share their expertise on various topics, helping others to learn and grow.
5. **Deliver High Performance:** Ensure the website is fast, stable, and reliable, providing a seamless user experience.

1.3 Scope

The scope of the project includes the following:

- **User Registration and Authentication:** Implementing secure user registration, login, and profile management features.
- **Tutorial Submission and Management:** Allowing users to create, edit, and delete their tutorials, with the ability to format them using Markdown.
- **Review and Feedback System:** Enabling users to review tutorials, provide feedback, and upload media related to their tutorials.
- **Content Viewing and Searching:** Providing features for users to search, view, and navigate through the tutorials submitted by others.
- **Responsive Design:** Ensuring the website is accessible and functional on various devices, including desktops, tablets, and smartphones.

1.4 Significance of Developing the Website

The significance of developing this website lies in its potential to democratize knowledge sharing across various domains. Here are the key points highlighting its importance:

1. **Universal Access to Knowledge:** By allowing users to share their knowledge in an organized and easily accessible format, the website promotes the dissemination of information to a global audience.
2. **Empowerment through Education:** Users can learn new skills, enhance their knowledge, and solve problems by accessing a diverse range of tutorials.
3. **Community Building:** The platform fosters a sense of community among users, encouraging them to share their expertise and learn from each other.
4. **Ease of Use with Markdown:** Markdown's simplicity and flexibility make it an ideal choice for writing tutorials. Users can include code samples, images, and other elements without needing complex formatting tools.
5. **Encouraging Content Creation:** By providing a user-friendly platform, we encourage more individuals to share their knowledge and experiences, contributing to a rich repository of tutorials.

In conclusion, the Tutorial Website project is a significant initiative aimed at creating a comprehensive, user-friendly platform for sharing and accessing tutorials. It leverages modern web technologies to provide a secure, efficient, and engaging user experience, ultimately contributing to the broader goal of knowledge democratization.

Chapter 2: Technical Details & Requirements

2.1 Introduction

This chapter provides a detailed overview of the technical aspects of the Tutorial Website project. It covers the user requirements, system requirements, hardware and software requirements, programming languages, and databases used in the development of the website.

2.2 User Requirements

The website is designed to be user-friendly and accessible to a wide range of users, including those with minimal technical knowledge. Key user requirements include:

- **User Registration and Authentication:** Users must be able to securely sign up, log in, and manage their profiles.
- **Tutorial Submission and Management:** Users should be able to create, edit, and delete tutorials, with support for Markdown formatting.
- **Responsive Design:** The website must be accessible and functional on various devices, including desktops, tablets, and smartphones.
- **Search and Navigation:** Users should be able to easily search for and navigate through tutorials.
- **Review and Feedback System:** Users must be able to review and provide feedback on tutorials.

2.3 System Requirements

The system requirements for the website are designed to ensure optimal performance and user experience. They include:

- **Internet Connection:** A stable internet connection is required for accessing the website and its features.
- **Browser Compatibility:** The website must be compatible with all modern web browsers, including Chrome, Firefox, Safari, and Edge.

2.4 Hardware Requirements

The website is designed to work efficiently on a wide range of devices, including low-end hardware. Specific hardware requirements include:

- **Low-End Devices:** The website is optimized to work on low-end devices without compromising performance.
- **Mobile and Desktop Compatibility:** The website is responsive and performs well on both mobile and desktop devices.

2.5 Software Requirements

The software requirements for developing and running the website include:

- **Node.js:** A JavaScript runtime used for server-side development [1].(Node.js — Run JavaScript Everywhere, n.d.)

- **Express.js:** A web application framework for Node.js, used for building the frontend [2].(*Express - Node.js Web Application Framework*, n.d.)
- **Passport.js:** A middleware for user authentication, providing robust security [3].(*Passport.js*, n.d.)
- **Marked.js:** A library for parsing Markdown, allowing users to format their tutorials [4].(*Marked Documentation*, n.d.)
- **Bcrypt:** A library for hashing passwords, ensuring user security [5].(*Bcryptjs - Npm*, n.d.)
- **DOMPurify:** A library for sanitizing HTML to prevent malicious code injection [6].(*Dompurify - Npm*, n.d.)
- **Tailwind CSS:** A utility-first CSS framework for styling the website [7].(*Tailwind CSS - Rapidly Build Modern Websites without Ever Leaving Your HTML.*, n.d.)
- **Mongodb:** An Object Data Modeling (ODM) library for MongoDB, used for database management [8].(*MongoDB: The Developer Data Platform | MongoDB*, n.d.)

2.6 Programming Languages

The primary programming language used in the development of the website is JavaScript. JavaScript is used for both server-side and client-side development, ensuring a seamless integration between the frontend and backend components.

2.7 Databases

The website uses MongoDB as the database. MongoDB is a NoSQL database that provides high performance, high availability, and easy scalability. It is accessed and managed using Mongoose, an ODM library for Node.js.

2.8 Performance

The website is designed to provide a fast and efficient user experience. Key performance metrics include:

- **Routing System:** The website uses a routing system that loads only the required components when a user navigates to a new page, improving load times and reducing server load.
- **Lighthouse Performance Test:** The website achieves a performance score of 89 for mobile devices and 100 for desktop devices in the Lighthouse performance test, indicating high efficiency and responsiveness.

2.9 Conclusion

In conclusion, the technical details and requirements of the Tutorial Website project ensure that it is a robust, secure, and user-friendly platform. By leveraging modern web technologies and frameworks, the website provides an optimal user experience while maintaining high performance and security standards.

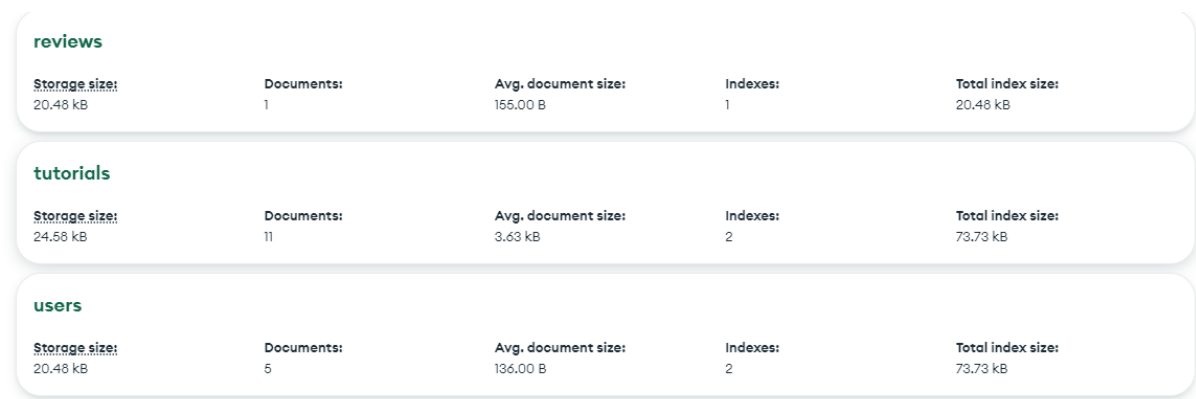
Chapter 3: Design

3.1 Database Design

Our project utilizes a NoSQL database, specifically MongoDB, to manage and store data efficiently. The database is structured into three main collections:

- Tutorials Collection:** This collection contains all the tutorials submitted by users. Each tutorial document includes the following fields:
 - Title:** The title of the tutorial.
 - Description:** A brief description of the tutorial content.
 - Markdown:** The tutorial content written in Markdown format.
 - SanitizedHtml:** A field generated from the Markdown content, containing sanitized HTML to ensure that no malicious code is injected.
- Users Collection:** This collection stores user information securely. Each user document includes:
 - Email:** The user's email address.
 - Password:** The user's password, which is hashed using the Bcrypt library to ensure security.
- Reviews Collection:** This collection holds feedback and contact information submitted by users who encounter issues or wish to provide reviews. Each review document includes:
 - ReviewEmail:** The email address of the user providing the feedback.
 - ReviewMedia:** An optional field that can contain an image of the issue or bug.
 - ReviewMessage:** The message detailing the user's feedback or issue.

































Below is an illustration of the database design:


















Reviews Collection

```
_id: ObjectId('66712c5a017f99c85ad81d25')
reviewEmail: "learn3038it@gmail.com"
reviewMedia: "1718692954345-Tutorial Website (2).png"
reviewMessage: "nice work"
__v: 0
```

Tutorials Collection

★ tutorials						
	_id ObjectId	createdAt Date	user ObjectId	title String	Description String	
1	ObjectId('6668b680224bfe...	2024-06-11T20:41:36.406+...	ObjectId('6667e9954a746e...	"Github"	"discription"	  
2	ObjectId('6668c15a224bfe...	2024-06-11T21:27:54.297+...	ObjectId('66683b325b7e2a...	"Github By abdalle"	"this is not a good Desc...	  
3	ObjectId('6668cf034d4879...	2024-06-11T22:26:11.808+...	ObjectId('6667e9954a746e...	"Another One by Me"	"Testing"	  
4	ObjectId('666956f848b667...	2024-06-12T08:06:16.831+...	ObjectId('66683b325b7e2a...	"gothub"	"heeh"	  
5	ObjectId('666aa0f14f4593...	2024-06-13T07:34:09.889+...	ObjectId('6667e9954a746e...	"sdsdsd"	"dss"	  
6	ObjectId('666b32e8bf343c...	2024-06-13T17:56:56.716+...	ObjectId('6667e9954a746e...	"Some visual"	""	  
7	ObjectId('666b68509d9b33...	2024-06-13T21:44:48.381+...	ObjectId('66683b325b7e2a...	"Testing"	"testing"	  
8	ObjectId('666b692c9d9b33...	2024-06-13T21:48:28.836+...	ObjectId('66683b325b7e2a...	"tITL"	""	  
9	ObjectId('666fe078f0fbee...	2024-06-17T07:06:32.707+...	ObjectId('66683b325b7e2a...	"My firt page"	"some testing"	  
10	ObjectId('667044e2a7225c...	2024-06-17T14:14:58.639+...	ObjectId('66683b325b7e2a...	"Title"	""	  
11	ObjectId('66705aa66acd19...	2024-06-17T15:47:50.430+...	ObjectId('66683b325b7e2a...	"Clipload app"	""	  

Users Collection

★ users						
	_id ObjectId	username String	password String	__v Int32		
1	ObjectId('6667e3f54a746e...	"abd@gmail.com"	"\$2a\$10\$dc7ZLJcgeK8zRG9...	0		  
2	ObjectId('6667e9954a746e...	"fahad11@gmail.com"	"\$2a\$10\$.spNWgvFzFf/Cdgj...	0		  
3	ObjectId('66683b325b7e2a...	"abdalle@gmail.com"	"\$2a\$10\$0jmUV8d1JiapbQAH...	0		  
4	ObjectId('666995cc28f13e...	"hatim@gmail.com"	"\$2a\$10\$LcerH6a/vPPG4GYk...	0		  
5	ObjectId('666998150a7ba3...	"mahir@gmail.com"	"\$2a\$10\$5VJ.tB7Bd7pG7qA1W...	0		  

3.2 Web-Tree Diagram

The website's structure is designed to be clear and intuitive, facilitating easy navigation and management. The web-tree diagram illustrates the hierarchical structure of the website, demonstrating how users can navigate through different sections and features.

3.3 Directory Structure

The project is organized into a single main directory called backend. This directory consolidates all the components required for both backend and frontend functionalities, ensuring streamlined development and deployment processes. The directory structure is as follows:

- **Backend Directory:** The main directory that contains all the subdirectories and files for the project.
 - **Models Directory:** Contains all the Mongoose models representing the database collections.
 - tutorialModel.js: Model for the Tutorials collection.
 - userModel.js: Model for the Users collection.
 - reviewsModel.js: Model for the Reviews collection.
 - **Routes Directory:** Contains all the route files that define the application's endpoints.
 - tutorialRouter.js: Defines routes for tutorial-related operations.
 - **Views Directory:** Contains all the EJS view files that render the frontend pages.
 - **Middleware Directory:** Contains middleware functions for the application.

- **authMiddleware.js**: Middleware for handling user authentication, ensuring that routes requiring authentication are protected and accessible only to authenticated users.
- **Server.js**: The main server file that initializes and configures the application, including middleware, routes, and database connection.

The following is a visual representation of the directory structure:



(Ubuntu Manpage: *Tree - List Contents of Directories in a Tree-like Format.*, n.d.)

3.4 Conclusion

The design of the Tutorial Website project is structured to ensure ease of use, security, and efficient data management. By organizing the project into clear and distinct collections, routes, and views, we have created a robust and scalable platform that supports user engagement and content sharing. The consolidated directory structure under the backend directory simplifies development and deployment, ensuring that all necessary components are easily accessible and manageable.

Chapter 4: Implementation

In this chapter, we provide a detailed look into the implementation of the Tutorial Website project, including partial coding examples with explanations to illustrate key aspects of the development process.

Setting Up the Server

The first step in our implementation was setting up the server using Node.js and Express.js. This involved configuring the necessary middleware and initializing the application.

```
const express = require('express')
const app = express()
const mongoose = require('mongoose')
const Tutorial = require('./models/tutorialModel')
const methodOverride = require('method-override')
const User = require('./models/userModel');
const Reviewsmdl = require('./models/reviewsModel');
const passport = require('passport');
const session = require('express-session');
const flash = require('express-flash');
const multer = require('multer');
const { ensureAuthenticated } = require('./middleware/authMiddleware');
require('./config/passportConfig')
const tutorialRouter = require('../backend/router/tutorialRouter')
const path = require('path')
const imageMimeTypes = ['image/jpeg', 'image/png', 'image/gif']
const uploadPath = path.join('backend', Reviewsmdl.coverImageBasePath);
const e = require('express')

if(process.env.NODE_ENV !== 'production'){

  require('dotenv').config();
}
/* Database Connection
const databaseurl = process.env.DATABASE_URL
mongoose.connect(process.env.DATABASE_URL)
  .then(() => console.log('MongoDB connected'))
  .catch(err => console.error('MongoDB connection error:', err));
```

- ❑ **Imports and Setup:** We start by importing necessary modules such as express, mongoose, and passport. We also import our custom middleware and router.
- ❑ **Express Application:** We create an instance of an Express application.
- ❑ **Database Connection:** We connect to a MongoDB database using mongoose.connect, we used “.env” to store our secret keys for database and then we require it if we are in the deployment production.


```

app.use(session({
  secret: 'H2HSS$HS',
  resave: false,
  saveUninitialized: false
}));
app.use(express.urlencoded({extended:false}))
app.use(methodOverride('_method'))
app.use(flash());

```

❑ Middleware Configuration:

We configure various middleware including session for handling sessions, flash for flash messages, and passport for authentication.



```

//? All setters
app.use(passport.initialize());
app.use(passport.session());
app.set('view engine','ejs')
app.set('views', path.join(__dirname, './views'))

```

❑ View **Engine**: We set ejs as the view engine and specify the directory for our view files.

```

app.use('/tutorials',tutorialRouter)

app.listen(3000)

```

❑ Start **Server**: Finally, we start the server on port 3000.

User Authentication

To ensure secure access to certain parts of the website, we implemented user authentication using Passport.js.

```
router.post('/', ensureAuthenticated, async (req, res, next) => {
  req.tutorial = new Tutorial();
  req.tutorial.user = req.user.id;
  next();
}, saveArticleThenRedirect('new'));

router.put('/:id', ensureAuthenticated, checkTutorialOwnership, async (req, res, next) => {
  req.tutorial = await Tutorial.findById(req.params.id);
  next();
}, saveArticleThenRedirect('edit'));

router.delete('/:id', ensureAuthenticated, checkTutorialOwnership, async (req, res) => {
  await Tutorial.findByIdAndDelete(req.params.id);
  res.redirect('/');
});
```

Router Configuration and Middleware

This code is part of an Express router that handles various routes related to tutorials. It ensures that only authenticated users can perform certain actions and that they can only modify their own tutorials.

```
// used passport.js to insure user authentication and authorization
const passport = require('passport');
const LocalStrategy = require('passport-local').Strategy;
const User = require('../models/userModel');

passport.use(new LocalStrategy(async (username, password, done) => {
  const user = await User.findOne({ username });
  if (!user) {
    return done(null, false, { message: 'Incorrect username.' });
  }
  const isMatch = await user.comparePassword(password);
  if (!isMatch) {
    return done(null, false, { message: 'Incorrect password.' });
  }
  return done(null, user);
}));

passport.serializeUser((user, done) => {
  done(null, user.id);
});

passport.deserializeUser(async (id, done) => {
  const user = await User.findById(id);
  done(null, user);
});
```

This code snippet leverages Passport.js to establish a secure authentication system for your Node.js application, enabling you to control user access to specific resources based on their credentials.

Username Verification:

- **const user = await User.findOne({ username });**: This line attempts to find a matching user in your database using the User model. It retrieves the user document where the username property matches the provided username.
- **if (!user)**: If no user is found with the given username, the authentication fails. The done callback is executed with the following arguments:
 - **null**: No error occurred.
 - **false**: Authentication failed.
 - **message**: 'Incorrect username.': An informative message indicating an incorrect username.

Serialization and Deserialization functions :

Serialization:

passport.serializeUser((user, done) => {...}); This function is responsible for converting the user object into a format suitable for storage in a session (typically a database or cookie). In this case, it simply extracts the user's ID using `user.id`.

done(null, user.id): The done callback is called with:

- ☐ **null** (no error occurred)
- ☐ **user.id** (the user's ID, which will be stored in the session)

```
app.post('/register', async (req, res) => {
  const { username, password } = req.body;
  try {

    const newUser = new User({ username, password });
    await newUser.save();
    res.redirect('/login');
  } catch (error) {
    // Handle errors appropriately
    req.flash('error', 'An error occurred. Please try again later. ');
    console.error(error);
    res.redirect('/register');
  }
});
```

In this Part of the code is where we created or register the user by simply calling `req.body` in our from we only have two name in the input field called `username` and `password` we store them in the object `const {username,password}` and then

```
app.use((req, res, next) => {  
  res.locals.user = req.user || null;  
  next();  
});
```

We used the locals(type of cookie to store the user current in the session `res.locals.user = req.user`
If no user is logged in we assign it to null `|| null`;

Chapter 5: Results and Discussion

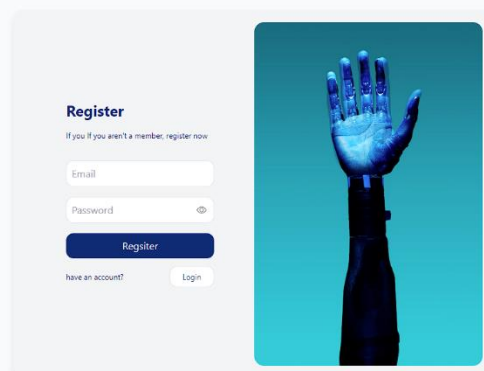
User Manual

This section provides a comprehensive guide for users on how to navigate and utilize the tutorial website. The user manual is divided into several key areas: account creation, tutorial submission, tutorial viewing, and contact support.

Account Creation

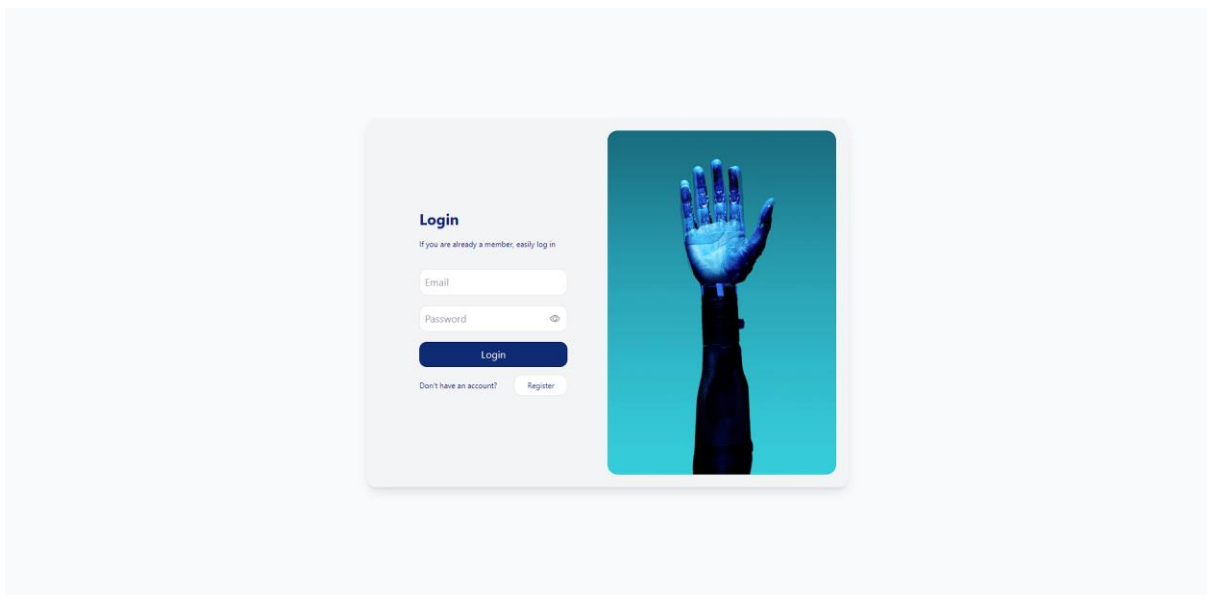
1. Sign Up:

- Navigate to the Sign-Up page.
- Fill in the required fields: username, email, and password.
- Click on the "Sign Up" button.



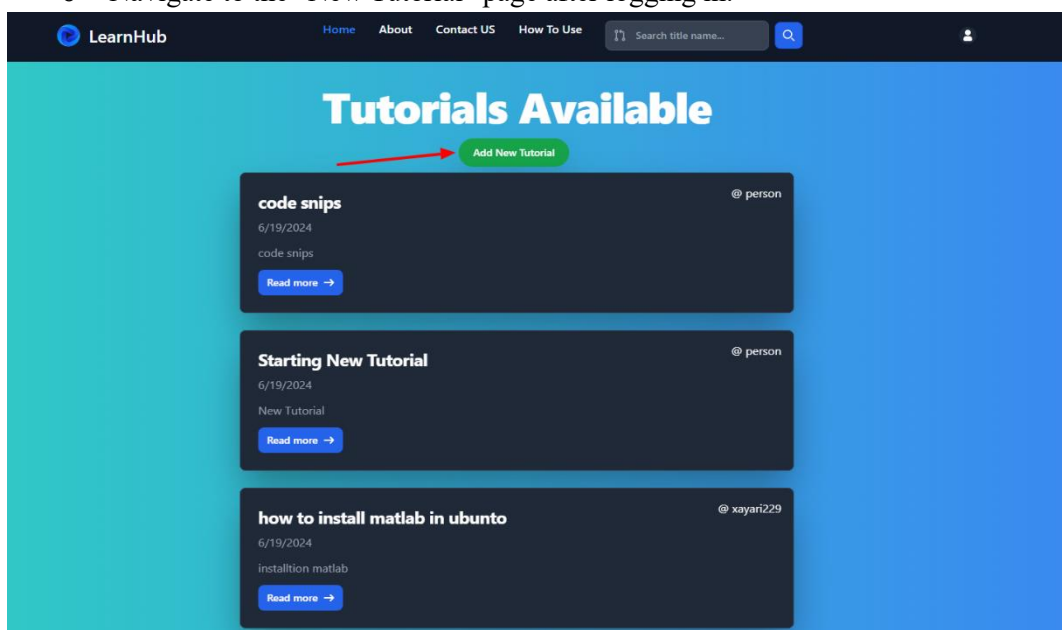
2. Log In:

- Navigate to the Log-In page.
- Enter your email and password.
- Click on the "Log In" button.
- Or in the signing up page click login
- If credentials are correct, you will be redirected to the homepage.
- First time signing up will automatically redirect You to the Login page



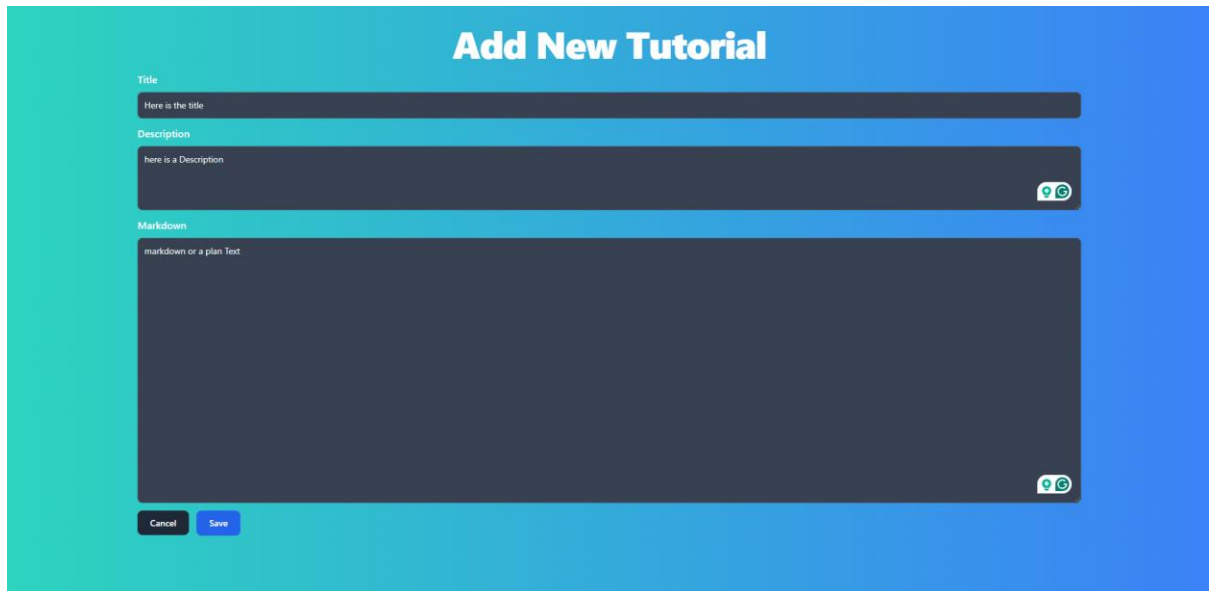
Tutorial Submission

- **Create New Tutorial:**
- Navigate to the "New Tutorial" page after logging in.



2.

- Fill in the title and description fields.
- Write your tutorial content using Markdown syntax for formatting.
- Click on the "Submit" button to save your tutorial.



Add New Tutorial

Title
Here is the title

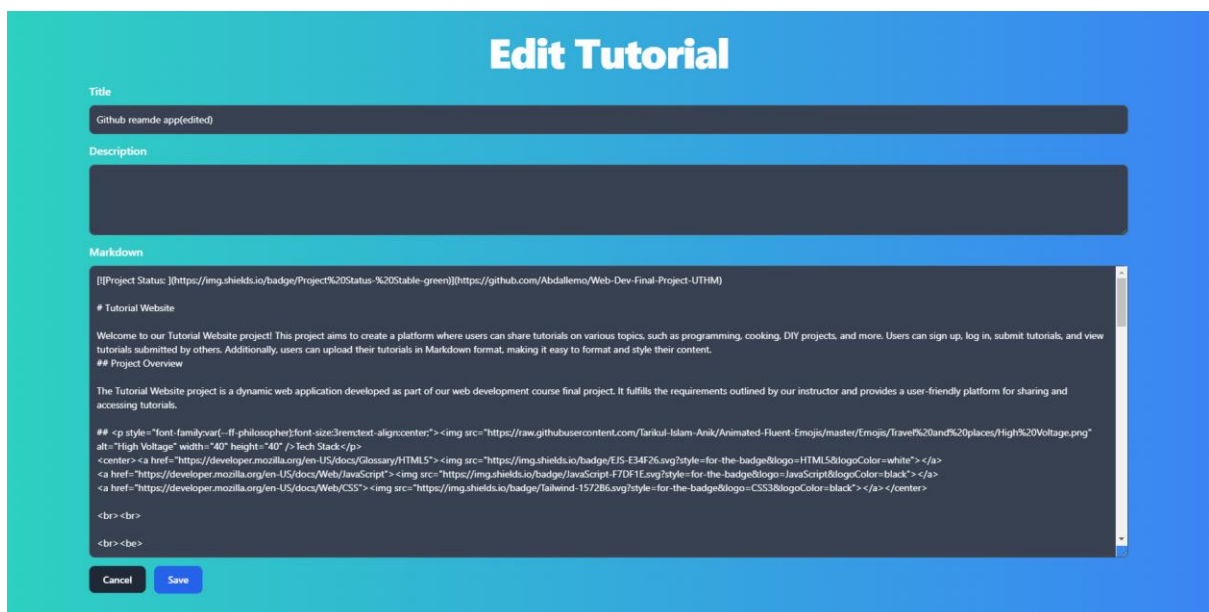
Description
here is a Description

Markdown
markdown or a plain Text

Cancel Save

3. Edit Tutorial:

- Navigate to the tutorial you want to edit.
- Click on the "Edit" button.
- Modify the content as needed.
- Click on the "Save" button to update the tutorial.
- Or if You don't want to click cancel



Edit Tutorial

Title
Github readme app(edited)

Description

Markdown

```
[[Project Status: ](https://img.shields.io/badge/Project%20Status-%20Stable-green)](https://github.com/Abdallema/Web-Dev-Final-Project-UTHM)

# Tutorial Website

Welcome to our Tutorial Website project! This project aims to create a platform where users can share tutorials on various topics, such as programming, cooking, DIY projects, and more. Users can sign up, log in, submit tutorials, and view tutorials submitted by others. Additionally, users can upload their tutorials in Markdown format, making it easy to format and style their content.

## Project Overview

The Tutorial Website project is a dynamic web application developed as part of our web development course final project. It fulfills the requirements outlined by our instructor and provides a user-friendly platform for sharing and accessing tutorials.

## <img alt="High Voltage" width="40" height="40" /> Tech Stack </p>
<center> <a href="https://developer.mozilla.org/en-US/docs/Glossary/HTML5">  </a>
<a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript">  </a>
<a href="https://developer.mozilla.org/en-US/docs/Web/CSS">  </a> </center>

<br> <br>
<br> <br>
```

Cancel Save

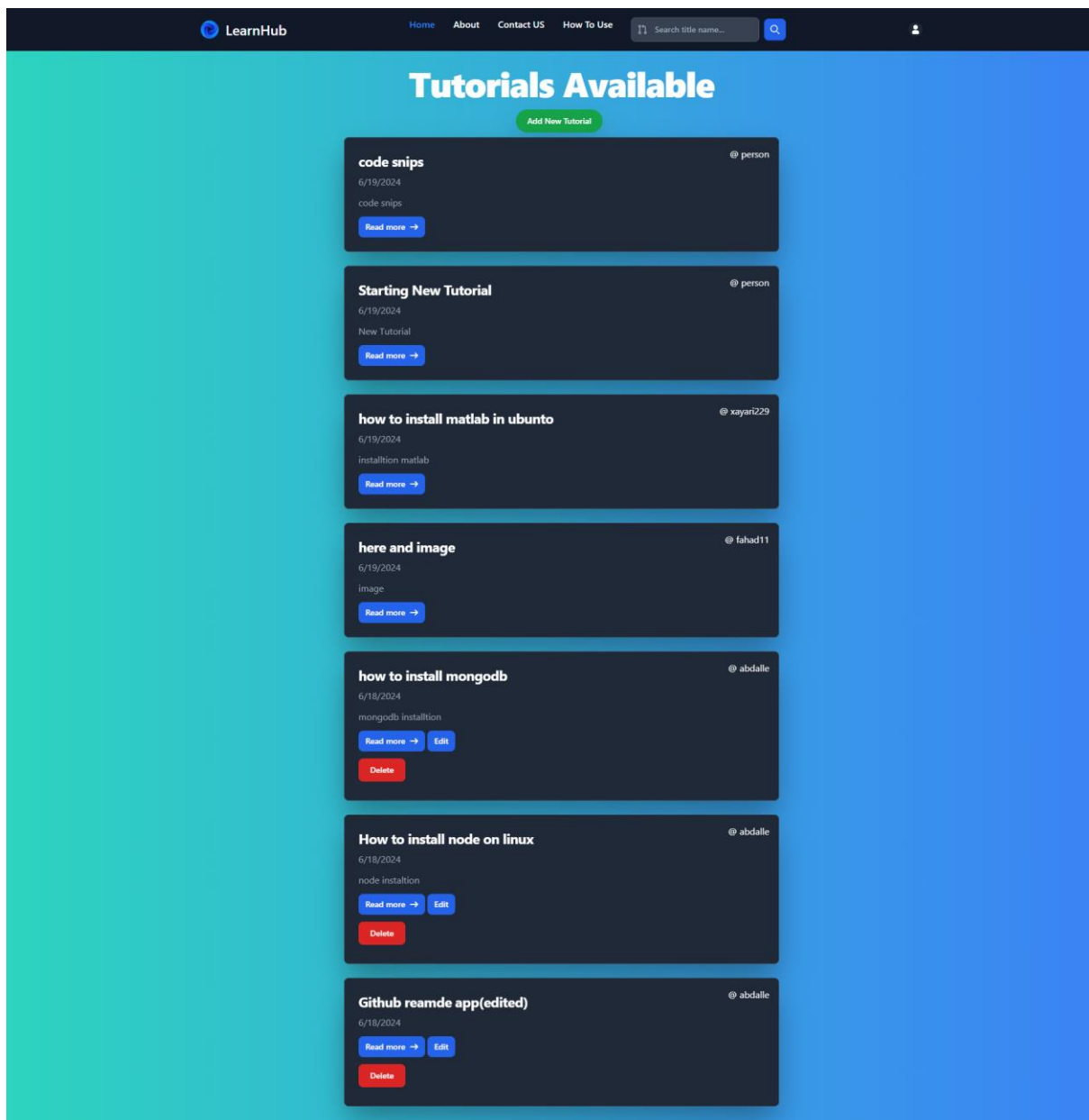
4. Delete Tutorial:

- Navigate to the tutorial you want to delete.
- Click on the "Delete" button.
- Confirm the deletion in the prompt that appears.
- You can Only delete Your Own Tutorials

Tutorial Viewing

1. Browse Tutorials:

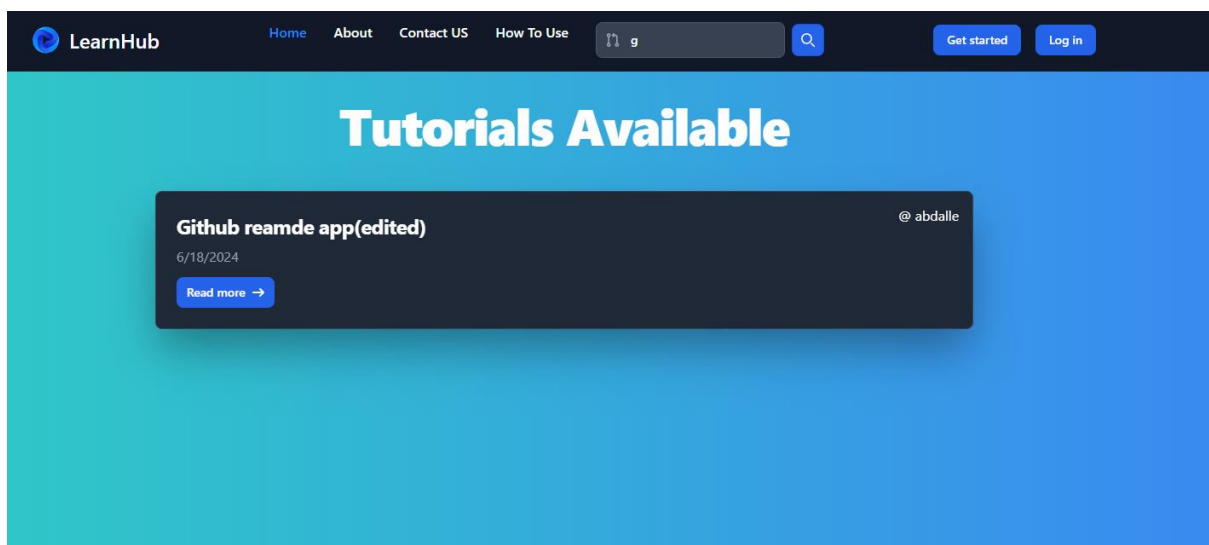
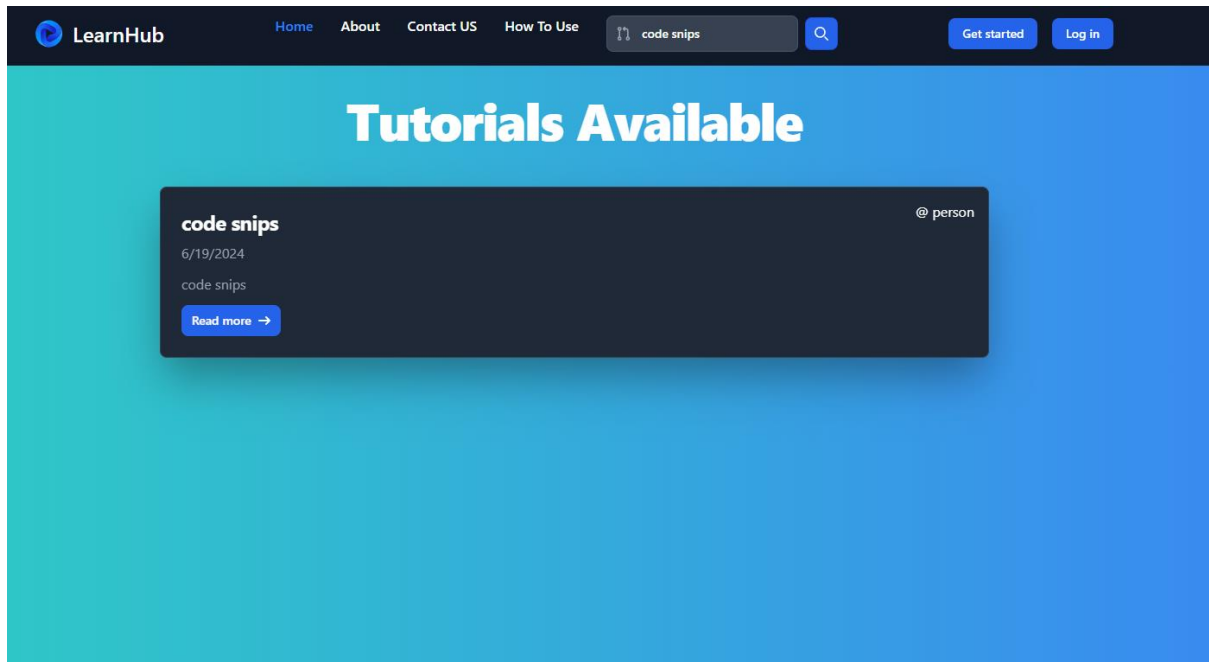
- From the homepage, you can browse through all the tutorials.



- Click on any tutorial title to view its content in detail.

2. Search Tutorials:

- Use the search bar at the top of the page to find tutorials by keywords or topics.



Contact Support

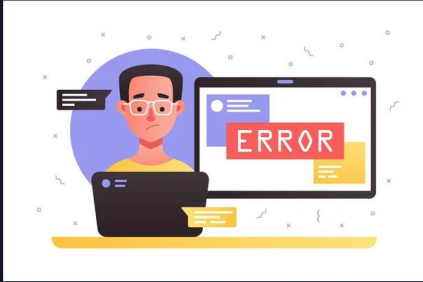
1. Submit an Issue or Feedback:

- Navigate to the "Contact Us" page.
- Fill in the required fields: name, email, and message.
- Optionally, you can attach an image or screenshot. (experimental feature)
- Click on the "Submit" button to send your message to the support team.

LearnHub

[Home](#)[About](#)[Contact Us](#)[How To Use](#)

[Get started](#)[Log in](#)



Contact Us

Got a technical issue? Want to send feedback about a beta feature? Need details about our Business plan? Let us know.

Upload file

Choose File

No file chosen

PNG, JPG or GIF.

Your email

name@flowbite.com

Your message

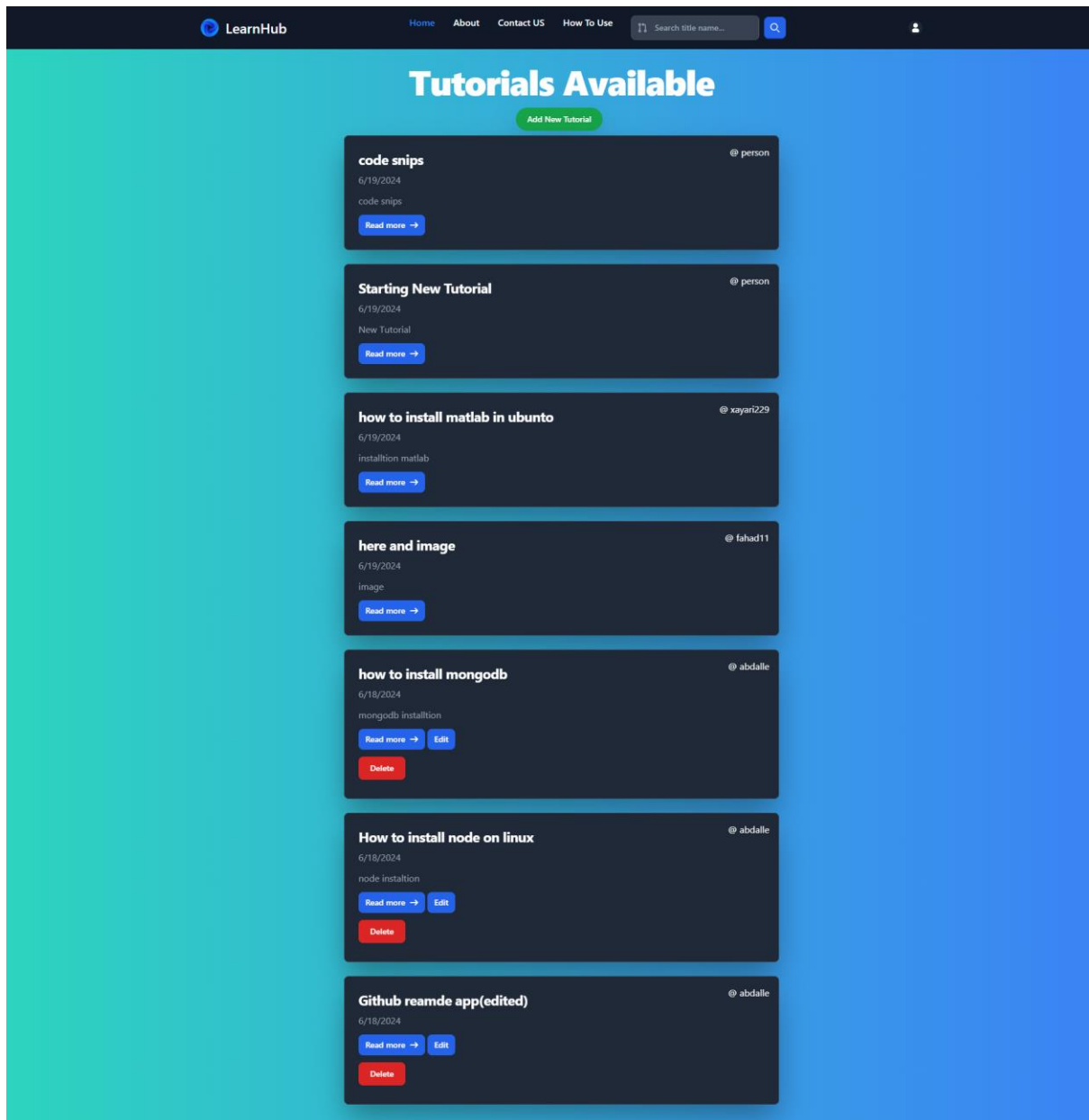
Leave a comment...

Send message

Flow of the Website

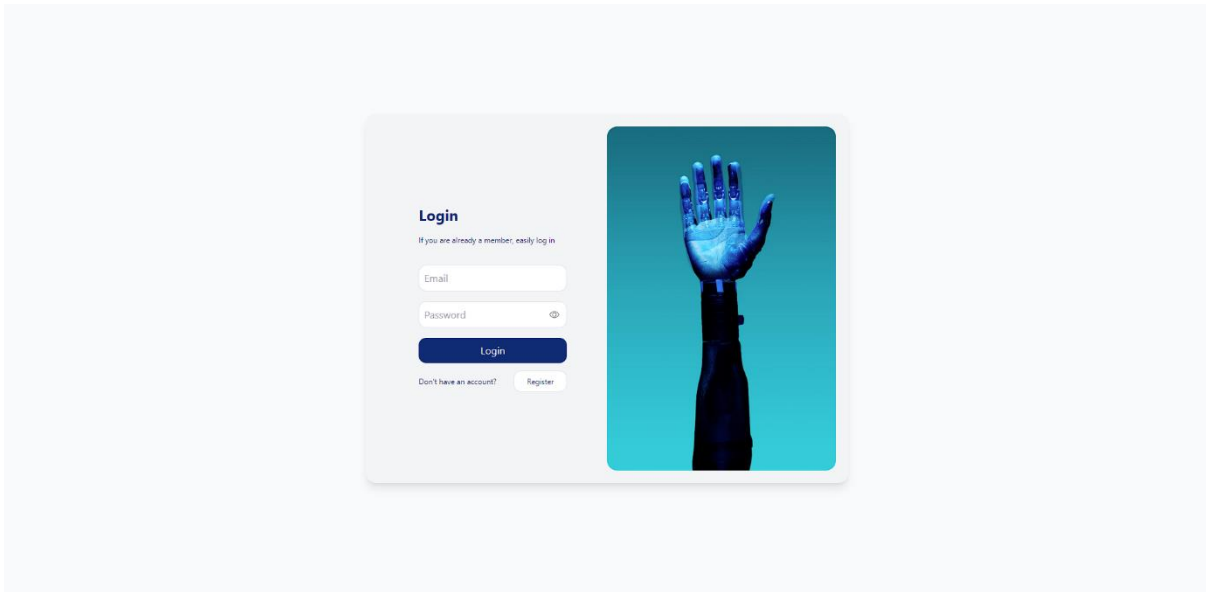
The flow of the website is designed to be intuitive and user-friendly. Below is a step-by-step outline of the primary interactions:

- Homepage:**
 - Displays a list of recent and popular tutorials.
 - Provides options for users to log in or sign up



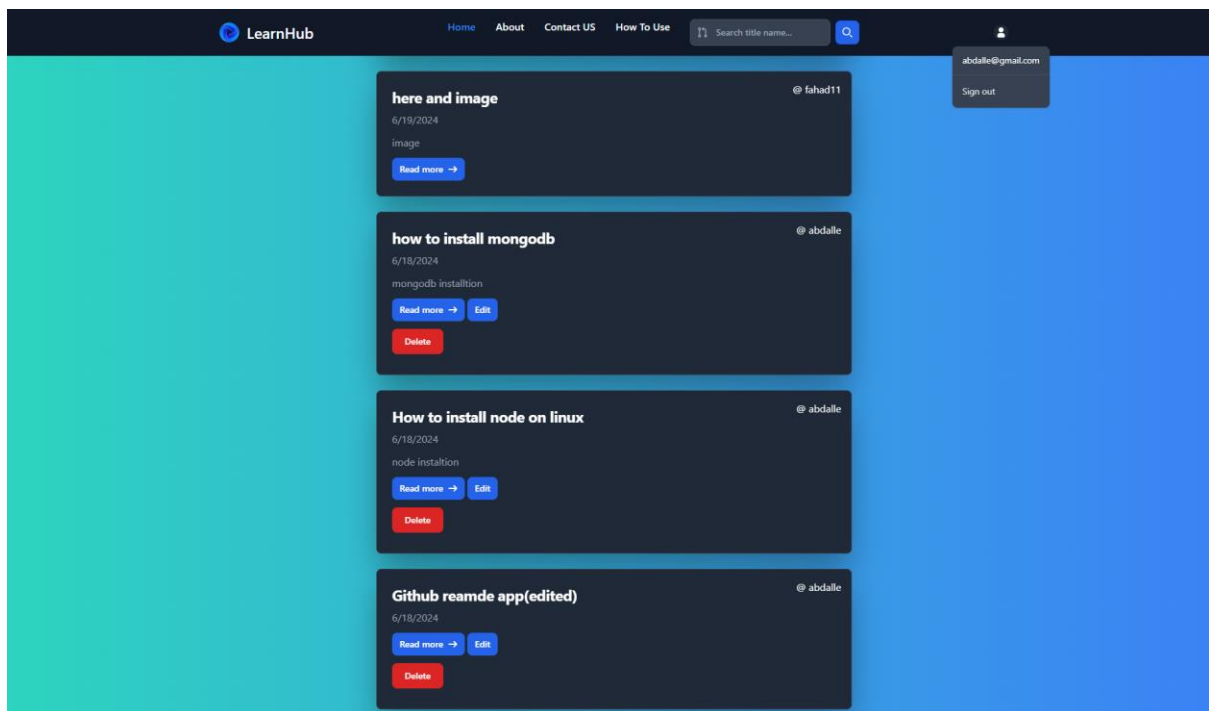
2. Authentication:

- Users can create an account or log in using their credentials.
- Successful authentication redirects users to their dashboard.




3. User Dashboard:

- Displays user-specific content, including their submitted tutorials and account settings with other people Tutorials.
- Provides options to create, edit, or delete tutorials that they own.




4. Tutorial Pages:

- Users can view the detailed content of any tutorial by clicking on its title.
- Tutorials are displayed with Markdown formatting for easy reading.

 LearnHub

[Home](#)[About](#)[Contact US](#)[How To Use](#)



abdalle@gmail.com

Sign out

how to install mongodb


6/18/2024

[All Tutorials](#)[Edit](#)


step one

```
console.log(error)
```

```
sudo apt innstall monoose
```

 LearnHub

[Home](#)[About](#)[Contact US](#)[How To Use](#)



abdalle@gmail.com

Sign out


Starting New Tutorial

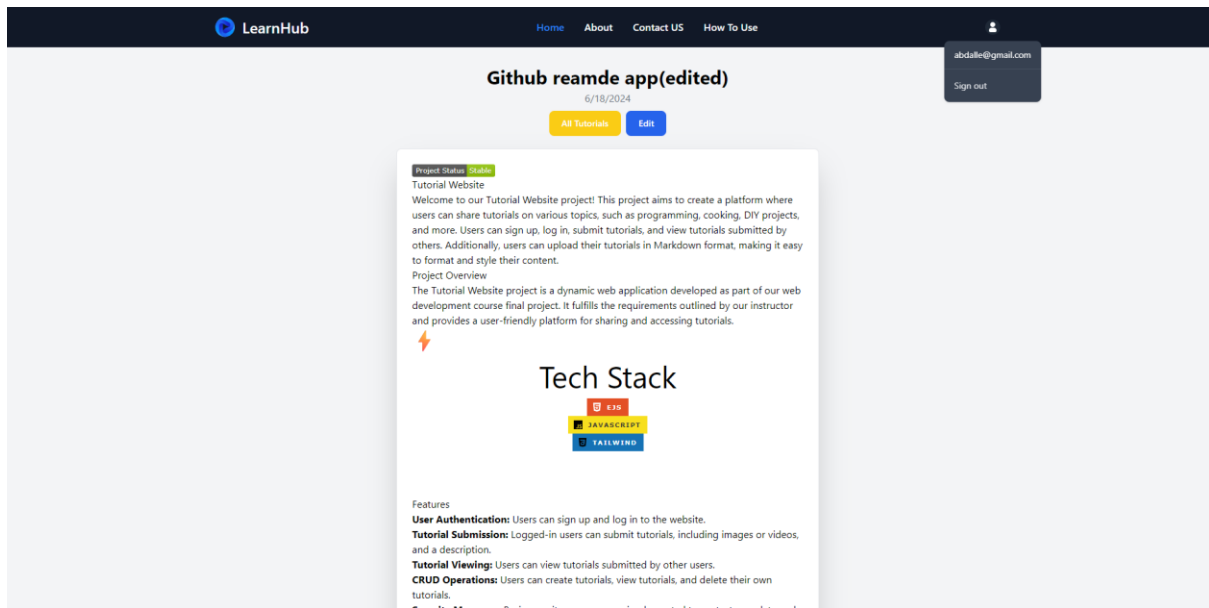
6/19/2024

[All Tutorials](#)

Hello word

☒ checked Box





5. Submission Pages:

- Users can create or edit tutorials using a Markdown editor.
- The submission is processed and saved to the database.

Add New Tutorial

Title

Here is the title

Description

here is a Description

Markdown

markdown or a plain Text

Cancel Save

6. Edit Page

- Users can edit their Previous tutorials by clicking edit.

Edit Tutorial

Title

Github reamde app(edited)

Description

Markdown

```
[[Project Status: ](https://img.shields.io/badge/Project%20Status-%20Stable-green)](https://github.com/Abdallema/Web-Dev-Final-Project-UTHM)

# Tutorial Website

Welcome to our Tutorial Website project! This project aims to create a platform where users can share tutorials on various topics, such as programming, cooking, DIY projects, and more. Users can sign up, log in, submit tutorials, and view tutorials submitted by others. Additionally, users can upload their tutorials in Markdown format, making it easy to format and style their content.

## Project Overview

The Tutorial Website project is a dynamic web application developed as part of our web development course final project. It fulfills the requirements outlined by our instructor and provides a user-friendly platform for sharing and accessing tutorials.

## <p style="font-family:var(--ff-philosopher),font-size:3rem;text-align:center;"> Tech Stack</p>
<center><a href="https://developer.mozilla.org/en-US/docs/Glossary/HTML5"></a>
<a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript"></a>
<a href="https://developer.mozilla.org/en-US/docs/Web/CSS"></a></center>

<br><br>
<br><br>
```

Cancel

Save

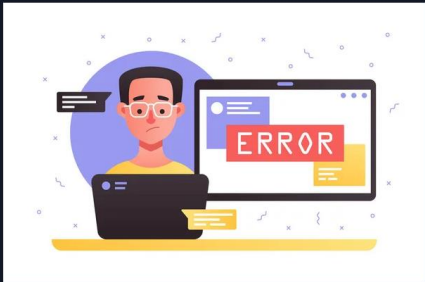
7. Contact Us:

- Users can submit feedback or report issues via a contact form.
- The form submission is handled by the backend and stored in the database.

LearnHub

Home About Contact US How To Use

Get started Log in



Contact Us

Got a technical issue? Want to send feedback about a beta feature? Need details about our Business plan? Let us know.

Upload file

Choose File

No file chosen

PNG, JPG or GIF.

Your email

name@flowbite.com

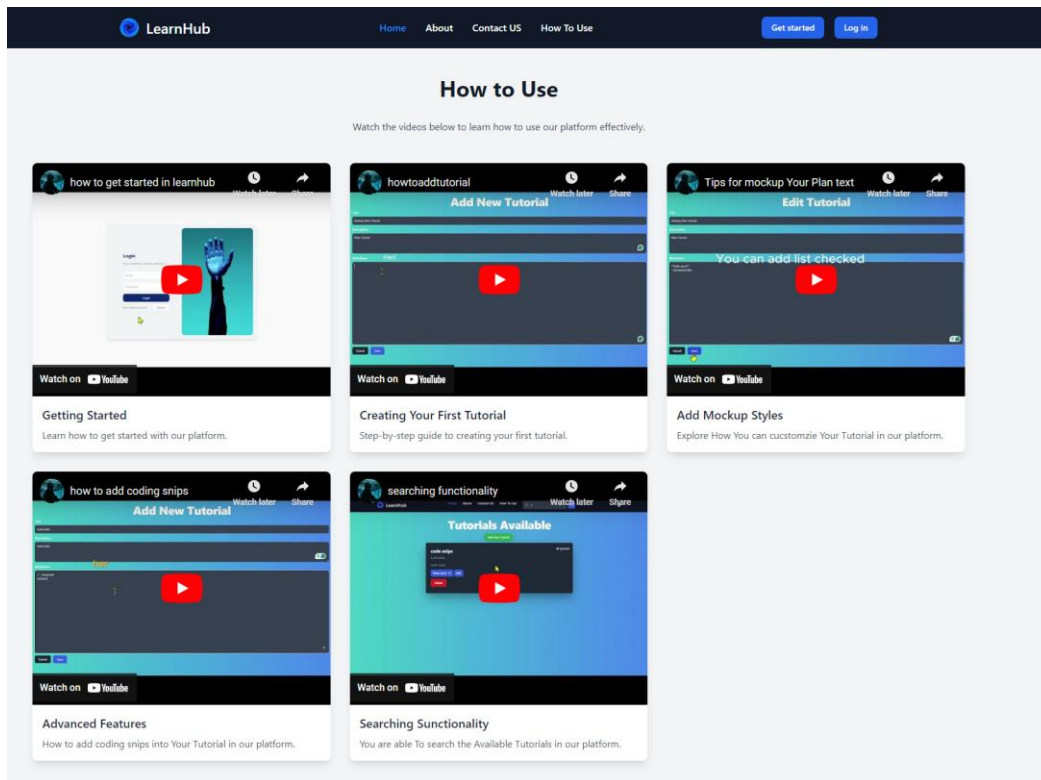
Your message

Leave a comment...

Send message

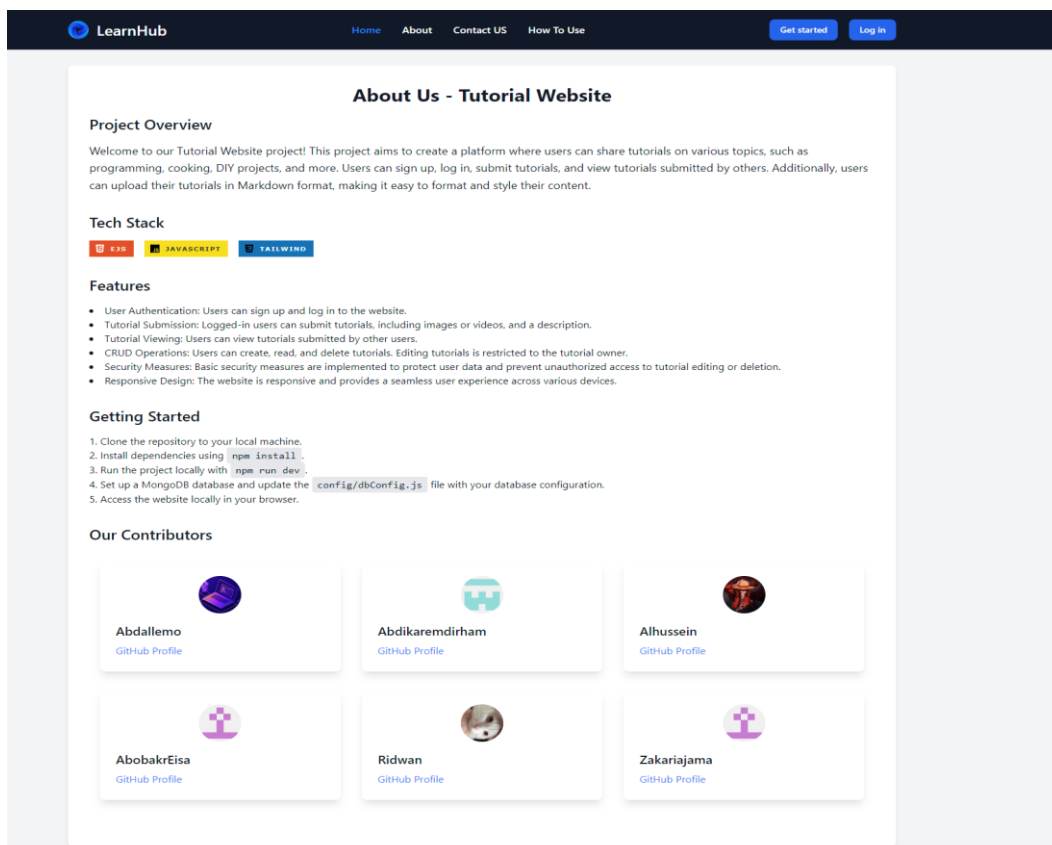
8. How to use page

- In this Page there are Videos that explains to the user How to do things in Our Platform



9. About Us page

- This page contains information About the Team And their GitHub



Discussion

The development of this tutorial website achieved its primary objective of creating a platform where users can share knowledge through well-formatted tutorials. Here are some key points of discussion:

1. User Experience:

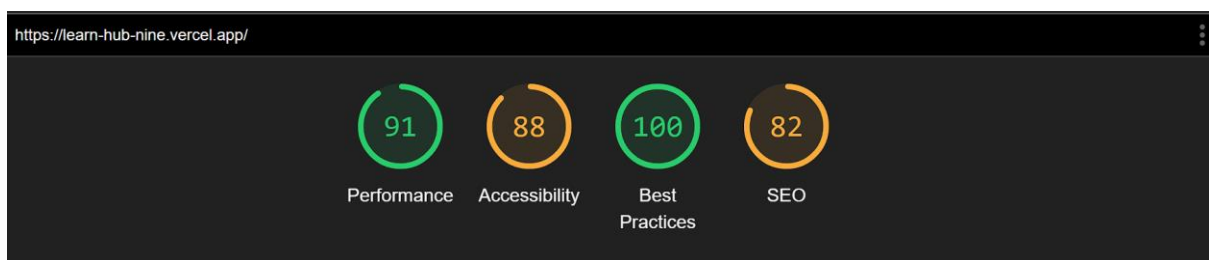
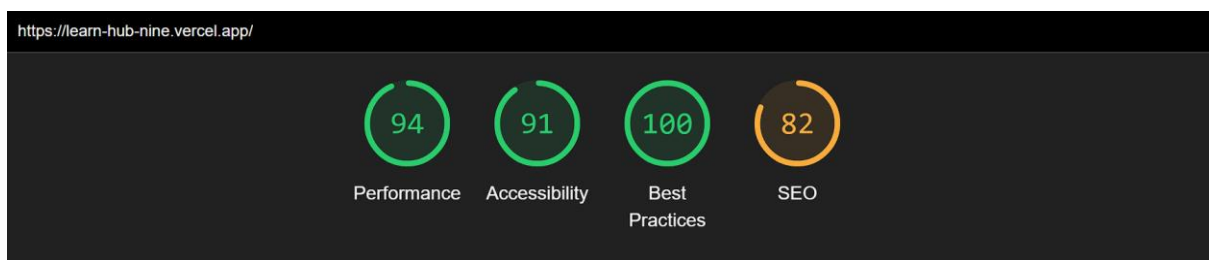
- The website offers a smooth and intuitive user experience, from account creation to tutorial submission.
- Markdown support allows users to format their tutorials easily, including code snippets and images.

2. Security:

- The use of Passport.js for authentication ensures that user data is securely handled.
- Bcrypt is employed to hash passwords, adding an extra layer of security.
- DOMPurify helps prevent malicious code injection by sanitizing user inputs.

3. Performance:

- The website is optimized for performance, with a Lighthouse score of 91 for mobile devices and 94 for desktop.



- The use of Node.js and Express.js ensures a fast and responsive backend.

4. Scalability:

- The choice of MongoDB as the database allows for easy scalability as the user base grows.
- The modular structure of the codebase facilitates the addition of new features and maintenance.

5. Challenges:

- One challenge faced was integrating the frontend and backend into a single directory to streamline deployment. This was resolved by restructuring the directory layout.
- Ensuring cross-browser compatibility and responsive design required careful testing and adjustments.

6. Future Enhancements:

- Implementing more advanced search functionality, including filtering by tags and categories.
- Adding social features, such as comments and likes, to increase user engagement.
- Developing a mobile application to complement the web version for on-the-go access.

In conclusion, the tutorial website successfully provides a platform for users to share and access a wide range of tutorials. It leverages modern web technologies to ensure a secure, performant, and user-friendly experience. The project demonstrates the potential for further growth and enhancement, making it a valuable resource for knowledge sharing.

References:

bcryptjs - npm. (n.d.). Retrieved June 19, 2024, from <https://www.npmjs.com/package/bcryptjs>

dompurify - npm. (n.d.). Retrieved June 19, 2024, from <https://www.npmjs.com/package/dompurify>

Express - Node.js web application framework. (n.d.). Retrieved June 19, 2024, from <https://expressjs.com/>

Marked Documentation. (n.d.). Retrieved June 19, 2024, from <https://marked.js.org/>

MongoDB: The Developer Data Platform | MongoDB. (n.d.). Retrieved June 19, 2024, from <https://www.mongodb.com/>

Node.js — Run JavaScript Everywhere. (n.d.). Retrieved June 19, 2024, from <https://nodejs.org/en>

Passport.js. (n.d.). Retrieved June 19, 2024, from <https://www.passportjs.org/>

Tailwind CSS - Rapidly build modern websites without ever leaving your HTML. (n.d.). Retrieved June 19, 2024, from <https://tailwindcss.com/>

Ubuntu Manpage: tree - list contents of directories in a tree-like format. (n.d.). Retrieved June 19, 2024, from <https://manpages.ubuntu.com/manpages/trusty/man1/tree.1.html>