

# TaskEase: Documentation

## 1. Project Overview

TaskEase is a personal task management system, specially designed to the employees of Miles of Style garment, to create and manage a list of tasks. It allows users to register, log in, and then use the system. The user interface consists of a form to add new task details like title, description, due date, and priority, and a list of all task details, and controls to mark tasks as completed, delete and update buttons and a logout button. The application uses MongoDB for data storage, Express.js for server-side logic, and EJS (Embedded JavaScript) for rendering dynamic HTML pages.

## 2. Architecture

### Frontend

- **HTML/CSS/JavaScript:** Frontend interface for user interaction and task management.
- **EJS:** Templating engine for server-side rendering of dynamic content.

### Backend

- **Node.js:** Runtime environment for server-side JavaScript execution.
- **Express.js:** To create the routes for the application, allowing the user to interact with the application through a web browser.
- **MongoDB:** NoSQL database for storing user data and tasks.
- **Mongoose:** ODM (Object Data Modeling) library for MongoDB and Node.js.
- **bcrypt:** Library for hashing passwords securely.
- **express-session:** Middleware for managing user sessions.
- **connect-mongodb-session:** MongoDB session store for storing session data.

**Nodemon:** To monitor changes to the code and automatically restart the server, making it easy to develop and test the application.

When the user adds a new task using the form, Node.js and Express.js handle the request and store the task in the database using Mongoose. When the user views the list of task details, EJS displays the tasks from the database in a list on the web page. When the user marks a task as completed or deletes a task, Node.js and Express.js handle the request and update the database using Mongoose.

### 3. APIs

#### ❖ Authentication Routes (`authentication.js`)

- GET /register (Renders the registration form)
  - Response: HTML page containing the registration form
- POST /register (Registers a new user)
  - Request Body: {"username": "string", "password": "string"}
  - Response: Redirect to `/login` on successful registration or {"error": "Error registering new user"}
- GET /login (Renders the login form)
  - Response: HTML page containing the login form
- POST /login (Authenticates a user)
  - Request Body: {"username": "string", "password": "string"}
  - Response: Redirect to `/tasks` on successful login or {"error": "Error logging in"}
- GET /logout (Logs out the authenticated user)
  - Response: Redirect to `/login`

#### ❖ Task Routes (`tasks.js`)

- GET / (Check the authentication)
  - Response: Redirect to `/tasks` if the user is authenticated and to `/login` if the user is not authenticated
- POST /tasks (Creates a new task)
  - Request Body: {"title": "string", "description": "string", "dueDate": "date", "priority": "string"}
  - Response: Redirect to /tasks on success or {"error": "Error message"}
- GET /tasks (Retrieves all tasks for the authenticated user)
  - Response: Render HTML page with tasks data or {"error": "Error message"} on failure

- GET /task/:id (Retrieves details of a specific task)
  - Request Params: Task ID
  - Response: `{"_id": "string", "title": "string", "description": "string", "dueDate": "date", "priority": "string", "completed": "boolean"} or {"error": "Error fetching task"}`
- POST /tasks/:id (Updates details of a specific task)
  - Request Params: Task ID
  - Request Body: `{"title": "string", "description": "string", "dueDate": "date", "priority": "string", "completed": "boolean"}`
  - Response: Redirect to `/tasks`` on success or `{"error": "Error message"}` on failure
- POST /tasks/:id/delete (Deletes a specific task)
  - Request Params: Task ID
  - Response: Redirect to `/tasks`` on success or `{"error": "Error message"}` on failure

## 4. Usage Instructions

### Prerequisites

1. **Node.js:** Ensure that you have Node.js installed on your system.
2. **MongoDB:** Ensure that MongoDB is installed and running on your system.

### Run Locally

1. Clone the Repository

```
git clone https://github.com/SGopinath89/I22342024TaskEase
```

2. Go to the project directory

```
cd TaskEase
```

3. Install Dependencies

```
npm install
```

#### 4. Start MongoDB

Make sure MongoDB is running. By default, it will run on

```
'mongodb://localhost:27017'
```

#### 5. Run the application

```
npm start
```

#### 6. View the Website

Open the browser and go to 'http://localhost:1070'

#### Application Structure

- **Views:** The EJS templates are located in the 'views' directory. These templates render the HTML pages for different routes.
- **Public:** The 'public' directory contains static files like CSS stylesheets.
- **Routes:** The 'routes' directory contains route definitions for authentication and task management.
- **Models:** The 'models' directory contains the Mongoose schemas for the `User` and `TodoTask` models.

#### User Registration

##### 1. Navigate to the registration page

<http://localhost:1070/register>

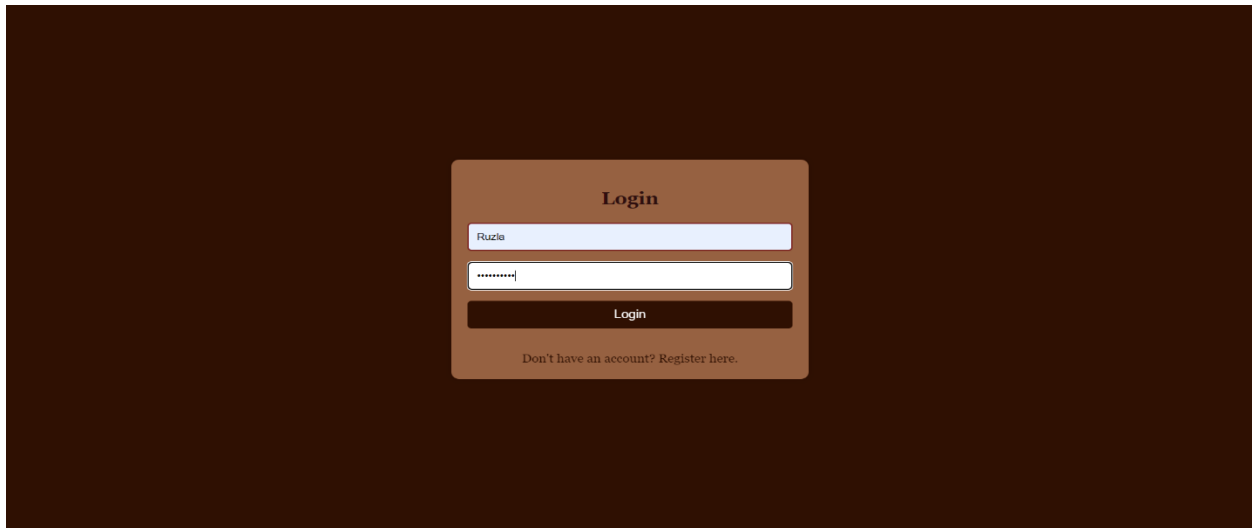
##### 2. Fill out the form with your desired username and password.

##### 3. Click "Register" to create a new account.

##### 4. You will be redirected to the login page upon successful registration.

## User Login

1. Navigate to the login page  
<http://localhost:1070/login>
2. Enter your registered username and password.
3. Click "Login" to authenticate.
4. You will be redirected to the tasks page upon successful login.

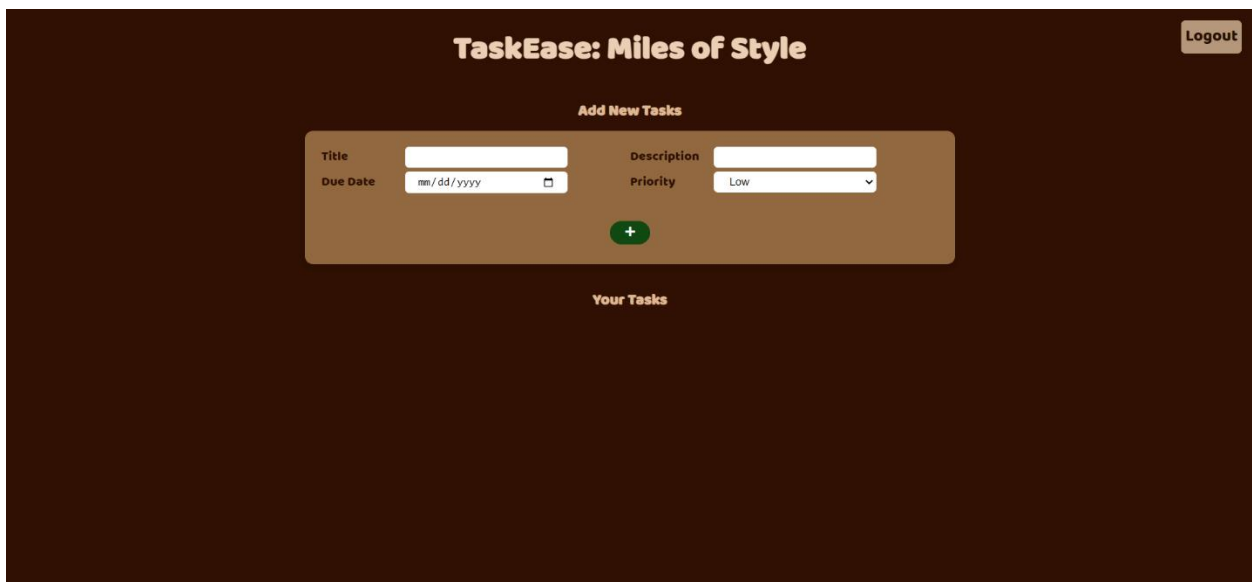


A screenshot of a login page with a dark brown background. In the center is a light brown rectangular box titled "Login". Inside the box, there are two input fields: the first is labeled "Ruzla" and contains the text "Ruzla"; the second is a password field with masked characters "\*\*\*\*\*". Below these fields is a dark brown button labeled "Login". At the bottom of the box, there is a link that says "Don't have an account? Register here."

## Managing Tasks

### ❖ Add New Task

- On the tasks page, fill out the form to add a new task.
- Click the "+" button to save the task.



A screenshot of the "TaskEase: Miles of Style" application. The header shows the title "TaskEase: Miles of Style" and a "Logout" button. Below the header is a section titled "Add New Tasks" containing a form with four fields: "Title" (empty), "Description" (empty), "Due Date" (with a date picker showing "mm/dd/yyyy"), and "Priority" (a dropdown menu set to "Low"). A green button with a white "+" sign is positioned below the form. Below the form is a section titled "Your Tasks" which is currently empty.

## ❖ View Tasks

- The tasks page displays a list of all your tasks.

The screenshot shows the 'TaskEase: Miles of Style' application interface. At the top right is a 'Logout' button. Below the header is the 'Add New Tasks' section, which includes a form with fields for 'Title' (Sewing), 'Description' (Stitching fabric pieces together), 'Due Date' (07/15/2024), and 'Priority' (High). A green '+' button is at the bottom of this form. Below this is the 'Your Tasks' section, which displays a list of three tasks. Each task row includes a checkbox, a title, a description, a due date, a priority dropdown, an edit button (pencil icon), and a delete button (X icon).

Task	Description	Due Date	Priority	Edit	Delete
<input type="checkbox"/> Designing	Creating sketches & selecting materials	07/05/2024	High		
<input type="checkbox"/> Pattern making	Developing templates for cutting fabric	07/08/2024	Medium		
<input type="checkbox"/> Fabric cutting	Cutting fabric pieces based on patterns	07/10/2024	Medium		

## ❖ Edit Task

- Click the edit button (pencil icon) next to a task to open the edit modal.
- Modify the task details and click "Save" to update the task.

The screenshot shows the 'TaskEase: Miles of Style' application with the 'Edit Task' modal open. The modal is a light blue box with a close button (X) in the top right corner. It contains a form with fields for 'Title' (Designing), 'Description' (Creating sketches & selecting materials), 'Due Date' (07/05/2024), and 'Priority' (Low). There is a 'completed' checkbox which is checked, and a 'Save' button at the bottom. The background shows the 'Add New Tasks' form and the 'Your Tasks' list, which is partially obscured by the modal.

Task	Description	Due Date	Priority	Edit	Delete
<input type="checkbox"/> Designing	Creating sketches & selecting materials	07/05/2024	Low		
<input type="checkbox"/> Pattern making	Developing templates for cutting fabric	07/08/2024	Medium		
<input type="checkbox"/> Fabric cutting	Cutting fabric pieces based on patterns	07/10/2024	Medium		
<input type="checkbox"/> Sewing	Stitching fabric pieces together	07/15/2024	High		

- Updated task details

TaskEase: Miles of Style

Logout

Add New Tasks

Title

Description

Due Date

mm/dd/yyyy

Priority

Low

+

Your Tasks

☒

Designing

Creating sketches & selecting ma

07/05/2024

Low

☐

Pattern making

Developing templates for cutting f

07/08/2024

Medium

☐

Fabric cutting

Cutting fabric pieces based on pa

07/10/2024

Medium

☐

Sewing

Stitching fabric pieces together

07/15/2024

High

## ❖ Delete Task

- Click the delete button (cross icon) next to a task to delete it.

TaskEase: Miles of Style

Logout

Add New Tasks

Title

Description

Due Date

mm/dd/yyyy

Priority

Low

+

Your Tasks

☐

Pattern making

Developing templates for cutting f

07/08/2024

Medium

☐

Fabric cutting

Cutting fabric pieces based on pa

07/10/2024

Medium

☐

Sewing

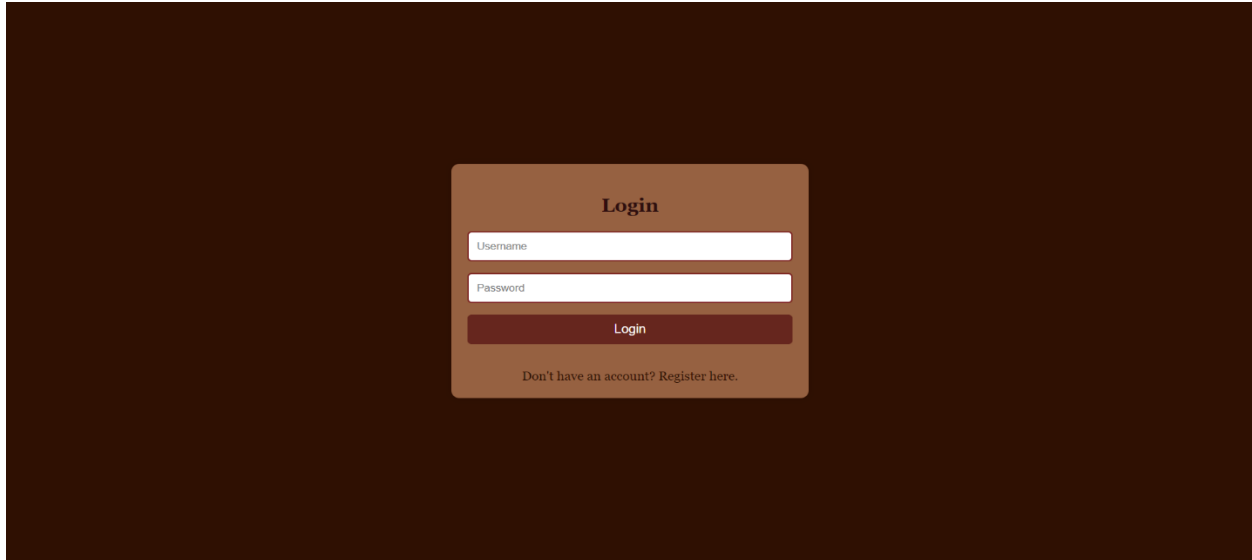
Stitching fabric pieces together

07/15/2024

High

## **Logging Out**

1. Click the "Logout" button to log out of your account.
2. You will be redirected to the login page upon successful logout.



## **5.Demo**

<https://drive.google.com/file/d/1HVNkibO2aiLBD94Owiqx9xbJdfwyX-81/view?usp=sharing>

## **6. Development Process**

- Planning: Defined project requirements and architectural design.
- Implementation: Developed backend using Node.js, Express.js, and MongoDB. Integrated frontend with EJS for dynamic rendering.
- Testing: Conducted unit testing for routes and functionalities.
- Deployment: Deployed on local environment for testing and refinement.

## **7. Challenges Faced**

- Session Management: Ensuring secure session handling and implementing login/logout functionality.
- Error Handling: Managing errors effectively, especially in async operations with MongoDB.
- Frontend Integration: Seamless integration of frontend (EJS templates) with backend APIs.
- Security: Ensuring password hashing, data validation, and preventing common vulnerabilities.

## **8. Conclusion**

TaskEase provides a robust solution for managing tasks with user authentication and session management. By following the provided documentation, users can easily set up and utilize the application for efficient task management.