

# SOCIAL POST IT

## Introduzione:

Il progetto "Postit Social" è un'applicazione web costruita utilizzando Node.js con il framework Express.

2 e il motore di templating EJS. L'applicazione consente agli utenti di registrarsi, effettuare il login e pubblicare post, che vengono visualizzati sulla home page. I dati, come gli utenti e i post, sono memorizzati in un file JSON locale.

## Funzionalità principali:

1-Registrazione degli utenti: Gli utenti possono registrarsi con un nome utente e una password.

La password è memorizzata in chiaro (in un'applicazione reale dovrebbe essere criptata).

2-Login: Gli utenti possono accedere all'applicazione tramite il login, inserendo il loro nome utente e password.3-Pubblicazione di post: Gli utenti, dopo aver effettuato il login, possono scrivere e pubblicare post sulla home page.

4-Visualizzazione dei post: Tutti i post pubblicati dagli utenti vengono visualizzati sulla home page dell'applicazione.

## Tecnologie Utilizzate:

Node.js: Un runtime JavaScript per costruire il server web.

Express: Un framework web minimalista per Node.js.

EJS (Embedded JavaScript): Un motore di templating per generare HTML dinamico.

Bootstrap 5: Una libreria CSS per creare un'interfaccia utente responsive.

JSON: Per memorizzare i dati relativi agli utenti e ai post.

Express-session: Per gestire le sessioni degli utenti.

Body-parser: Per leggere i dati inviati tramite POST.

## Descrizione dei File:

## 1. File app.js

Questo è il punto d'ingresso principale per l'applicazione. In questo file configuriamo il server Express, le route per la registrazione, il login, l'aggiunta di post e la visualizzazione dei post.

Il file app.js è il cuore del progetto "Postit - Social". Esso contiene la logica server-side, che gestisce le richieste HTTP, l'autenticazione degli utenti, la gestione dei post, la registrazione degli utenti e altre funzionalità di backend. Utilizza Node.js e il framework Express, e si occupa anche di collegare il front-end (le pagine EJS) con il backend, usando JSON per memorizzare i dati.

```
const express = require('express');
const ejs = require('ejs');
const bcrypt = require('bcryptjs');
const bodyParser = require('body-parser');
const session = require('express-session');
const fs = require('fs');
const app = express();
const port = 3000;

// Middleware per gestire le richieste POST
app.use(bodyParser.urlencoded({ extended: true }));
app.set('view engine', 'ejs'); // Impostiamo EJS come motore di templating
app.use(express.static('public')); // Cartella per i file statici come CSS e JS

// Configurazione della sessione
app.use(session({
  secret: 'secret-key',
  resave: false,
  saveUninitialized: true
}));

// Funzione per caricare i post da JSON
const loadPosts = () => {
  try {
    return JSON.parse(fs.readFileSync('posts.json')); // Legge i dati dal file posts.json
  } catch (err) {
    return [];
  }
};

// Funzione per salvare i post in JSON
const savePosts = (posts) => {
  fs.writeFileSync('posts.json', JSON.stringify(posts, null, 2)); // Salva i post nel file JSON
};

// Home Page: mostra tutti i post
app.get('/', (req, res) => {
  const posts = loadPosts(); // Carica i post
  res.render('home', { posts, user: req.session.user }); // Rende la pagina home con i post e l'utente attivo
});

// Pagina di login
app.get('/login', (req, res) => {
  res.render('login'); // Rende il template di login
});

// Gestione del login
app.post('/login', (req, res) => {
```

```

// Gestione del login
app.post('/login', (req, res) => {
  const { username, password } = req.body;
  let users = loadPosts(); // Carica gli utenti dai post.json
  const user = users.find(u => u.username === username); // Trova l'utente per nome utente

  if (user && user.password === password) { // Se le credenziali sono corrette
    req.session.user = user; // Salva l'utente nella sessione
    res.redirect('/'); // Ritorna alla home
  } else {
    res.redirect('/login'); // In caso di errore, ritorna alla pagina di login
  }
});

// Pagina di registrazione
app.get('/register', (req, res) => {
  res.render('register'); // Rende il template di registrazione
});

// Gestione della registrazione
app.post('/register', (req, res) => {
  const { username, password } = req.body;
  let users = loadPosts(); // Carica gli utenti
  const existingUser = users.find(u => u.username === username); // Controlla se l'utente esiste già

  if (existingUser) {
    return res.send("Username already exists."); // Se l'utente esiste già
  }

  const newUser = { username, password }; // Crea un nuovo utente
  users.push(newUser); // Aggiungi l'utente all'array
  savePosts(users); // Salva gli utenti nel file JSON
  res.redirect('/login'); // Redirect alla pagina di login
});

// Aggiunta di un nuovo post
app.post('/addPost', (req, res) => {
  const { content } = req.body;
  if (!req.session.user) { // Se l'utente non è loggato
    return res.redirect('/login'); // Redirect alla pagina di login
  }

```

```

  }

  const posts = loadPosts(); // Carica i post
  const newPost = {
    username: req.session.user.username, // Aggiunge il nome dell'utente al post
    content: content, // Contenuto del post
    date: new Date().toISOString() // Data del post
  };
  posts.push(newPost); // Aggiungi il post all'array
  savePosts(posts); // Salva i post nel file JSON
  res.redirect('/'); // Ritorna alla home
});

// Avvio del server
app.listen(port, () => {
  console.log(`Server is running on 
```

## 1-Home.ejs

La home page di questa applicazione è un elemento chiave, in quanto è dove gli utenti possono interagire con i contenuti pubblicati. Nella home page vengono visualizzati i post di tutti gli utenti e, se l'utente è autenticato, è possibile creare un nuovo post.

**Funzionalità principali:**

**Navbar dinamica:** La navbar cambia a seconda che l'utente sia loggato o meno. Se l'utente è loggato, viene visualizzato il pulsante Logout; altrimenti, sono visibili i pulsanti per il Login e per la Registrazione.

**Post visibili nella home:** Nella sezione centrale della pagina, vengono mostrati tutti i post pubblicati. Ogni post contiene:

Username dell'autore del post.

Data di pubblicazione.

Contenuto del post.

**Form di pubblicazione del post:** Se l'utente è loggato, un modulo consente di scrivere un nuovo post. I post vengono inviati al server tramite il metodo POST e salvati nel file JSON.

**Layout Responsive:** L'interfaccia utilizza Bootstrap 5 per garantire che la pagina sia visibile correttamente su dispositivi diversi, inclusi smartphone, tablet e desktop.

```

<html lang="en">
<head>
  <title>Postit - Social</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-light bg-light">
    <div class="container">
      <a class="navbar-brand" href="/">Postit</a>
      <div class="d-flex">
        <a href="/login" class="btn btn-primary">Login</a>
        <a href="/register" class="btn btn-secondary ms-2">Sign Up</a>
      </div>
    </div>
  </nav>

  <div class="container my-4">
    <h2>Welcome, <%= user ? user.username : 'Guest' %>!</h2>
    <% if (user) { %>
      <form action="/addPost" method="POST">
        <textarea name="content" class="form-control" rows="3" placeholder="Write a post..."></textarea>
        <button type="submit" class="btn btn-primary mt-3">Post</button>
      </form>
    <% } else { %>
      <p>Please <a href="/login">login</a> to post something.</p>
    <% } %>

    <div class="my-4">
      <h3>Recent Posts</h3>
      <% posts.forEach(function(post) { %>
        <div class="post">
          <p><strong><%= post.username %></strong> <span class="text-muted"><%= post.date %></span></p>
          <p><%= post.content %></p>
        </div>
        <hr>
      <% }); %>
    </div>
  </div>

  <footer class="bg-light text-center py-3">
    <p>&copy; 2025 Postit</p>
  </footer>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>

```

## 2-Login.ejs

La pagina di login consente agli utenti di autenticarsi nell'applicazione. Se l'utente ha già un account, può inserire il nome utente e la password per accedere al sistema.

**Funzionalità principali:**

**Modulo di login:** L'utente deve inserire il proprio username e password.

**Gestione degli errori:** Se l'utente inserisce credenziali errate, verrà visualizzato un messaggio di errore.

**Design professionale:** La pagina di login è progettata con un design moderno grazie a Bootstrap 5, che include una card centrata con ombre leggere per migliorare l'esperienza visiva.

```

> <> login.ejs > ...
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
  <div class="container">
    <h2 class="my-4">Login</h2>
    <form action="/login" method="POST">
      <div class="mb-3">
        <label for="username" class="form-label">Username</label>
        <input type="text" class="form-control" id="username" name="username" required>
      </div>
      <div class="mb-3">
        <label for="password" class="form-label">Password</label>
        <input type="password" class="form-control" id="password" name="password" required>
      </div>
      <button type="submit" class="btn btn-primary">Login</button>
    </form>
  </div>
</body>
</html>

```

Inizia con la dichiarazione del tipo di documento (`<!DOCTYPE html>`), che indica che il file è scritto in HTML5, e il tag `<html lang="en">` che specifica che la lingua principale della pagina è l'inglese. Nella sezione `<head>`, vengono impostate la codifica dei caratteri UTF-8, la visualizzazione ottimizzata per dispositivi mobili e il titolo della pagina, che è "Login". Inoltre, viene incluso il file CSS di Bootstrap tramite un tag `<link>`, il quale fornisce uno stile predefinito alla pagina. Il corpo della pagina è racchiuso nel tag `<body>` e contiene un modulo di login. Questo modulo è avvolto in un `<div class="container">`, che aiuta a centralizzare il contenuto e ad adattarlo alle dimensioni del dispositivo. Il modulo include due campi di input, uno per l'username e uno per la password, entrambi con l'attributo `required` per garantire che l'utente li compili prima di inviare il modulo. I campi sono preceduti da etichette (`<label>`) e sono stilizzati con la classe `form-control` di Bootstrap. Infine, il modulo ha un pulsante di invio (`<button type="submit" class="btn btn-primary">Login</button>`) che, quando cliccato, invia i dati al server tramite il metodo POST all'URL `/login`. La pagina si avvale di Bootstrap per garantire un design responsivo e moderno, ma non include il JavaScript per funzionalità avanzate, che potrebbe essere aggiunto se necessario.

### 3-Register.ejs

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Register - Postit</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
  <style>
    body {
      background-color: #f8f9fa;
    }

    .form-container {
      margin-top: 50px;
    }

    .form-card {
      background-color: white;
      border-radius: 10px;
      box-shadow: 0 2px 15px rgba(0, 0, 0, 0.1);
      padding: 30px;
    }

    .btn-custom {
      background-color: #007bff;
      color: white;
    }

    .btn-custom:hover {
      background-color: #0056b3;
    }

    footer {
      background-color: #007bff;
      color: white;
      padding: 15px 0;
      text-align: center;
      margin-top: 40px;
    }
  </style>
</head>

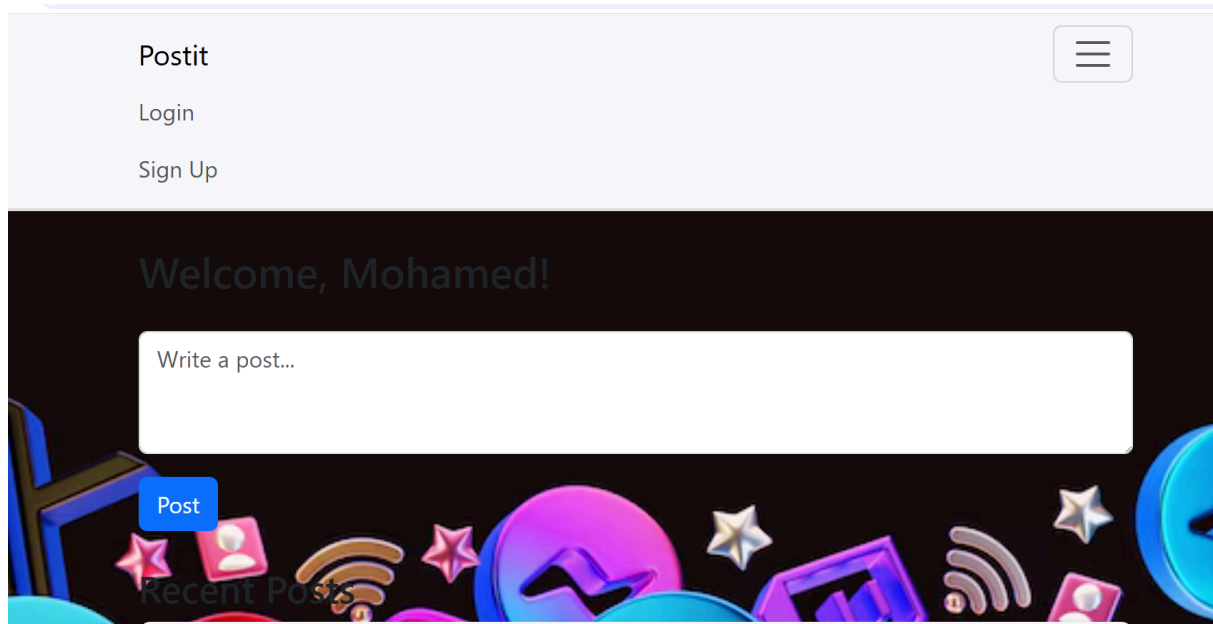
```



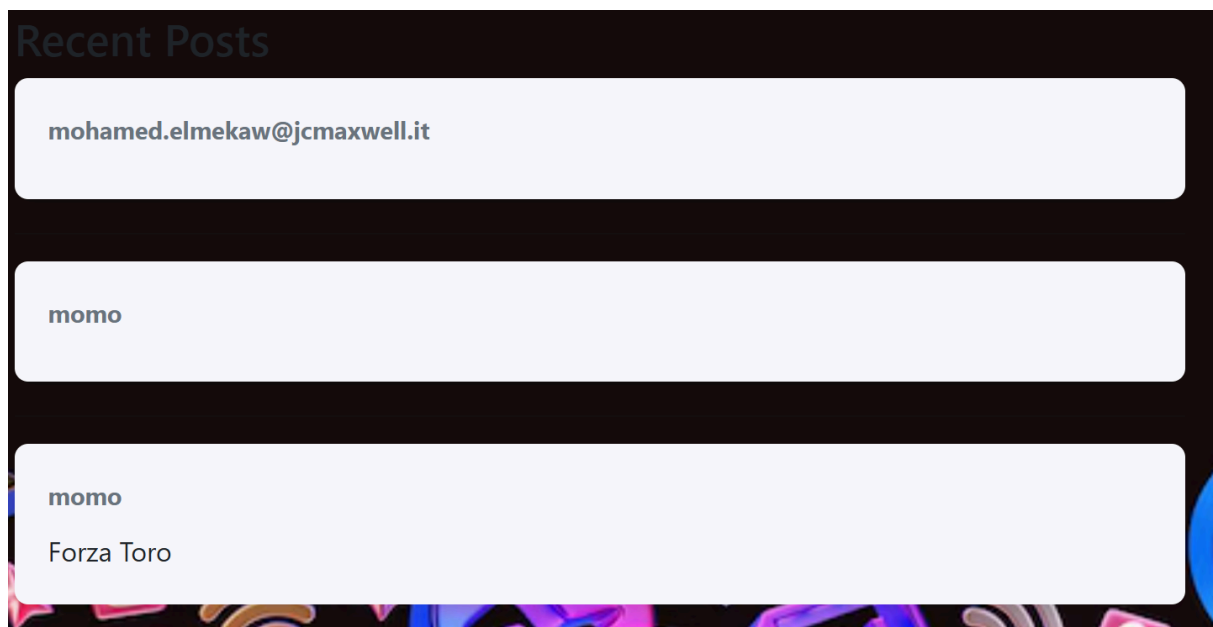


VIDEATE

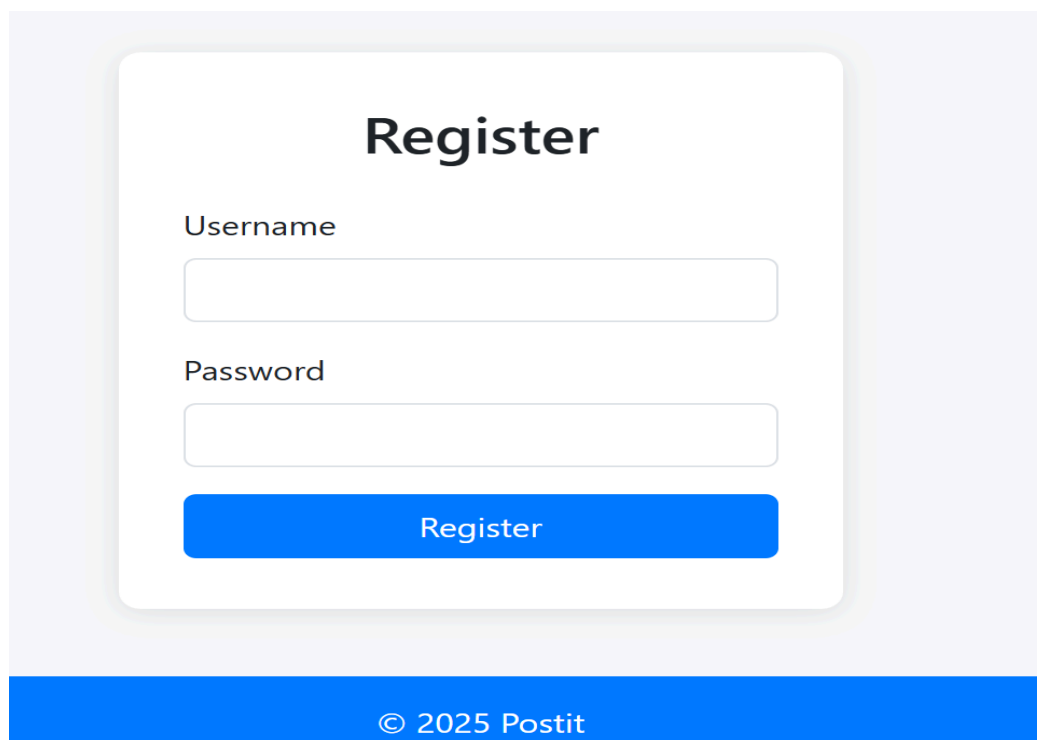
## Pagina Home



Sotto abbiamo tutti i post recenti inseriti



Poi abbiamo le 2 condizioni di login e registrazione

A registration form titled "Register" is centered on a light gray background. The form is contained within a white rounded rectangle with a subtle drop shadow. It features two input fields: "Username" and "Password", each with a light gray border and rounded corners. Below the password field is a solid blue button with the text "Register" in white. At the bottom of the page, a solid blue footer bar contains the copyright text "© 2025 Postit" in white.

## Register

Username

Password

Register

© 2025 Postit

# Login

Username

Password

Login

Non hai un account? [Registrati qui](#)