**Secure User Registration and Login**

**1. Introduction**

Welcome to the documentation for the **Secure User Registration and Login** project. This project demonstrates a secure and efficient user registration and login system using Node.js, Express, MongoDB, and bcrypt for password hashing.

**2. Project Overview**

The project aims to create a web application that provides a user-friendly interface for user registration and login, with secure password storage and authentication. The user's information is stored in a MongoDB database, and sensitive data like passwords are securely hashed using the bcrypt library.

**3. Installation and Setup**

To run the project locally, follow these steps:

1. Clone the project repository from GitHub.
2. Navigate to the project directory using a terminal.
3. Run **npm install** to install the required dependencies.
4. Ensure that you have MongoDB installed and running on your system.
5. Update the MongoDB connection details in **src/db/db.js**.
6. Start the server using **npm run dev**.
7. Open your web browser and access the application at **http://localhost:3000**.

**4. Roadmap**

1. **Express Server Setup and Middleware Configuration:** Setting up the Express server, configuring middleware such as static files and form data parsing.
2. **Database Model and Connection:** Defining a MongoDB schema for user data and establishing a connection to the database.
3. **User Registration:** Handling user registration data, hashing passwords, and saving user information to the database.
4. **User Login:** Implementing user login functionality, comparing hashed passwords, and creating secure authentication tokens.
5. **Logout and Cookies:** Managing user sessions, handling logouts, and using cookies for session persistence.

**5. Key Code Explanations**

* **Express Server Setup**

The **app.js** file sets up the Express server and configures middleware:

const express = require("express"); const cookieParser = require('cookie-parser'); const app = express(); const path = require('path'); const port = 3000; app.use(express.static(path.join(\_\_dirname, 'public'))); // ... Other middleware configurations ... app.listen(port, () => { console.log(`Server is running at port no ${port}`); });

* **Database Model**

The **model/model.js** defines the MongoDB schema and model for user data:

const mongoose = require('mongoose'); const formDataSchema = new mongoose.Schema({ name: { type: String, required: true }, email: { type: String, unique: true, required: true }, phone: { type: Number, required: true }, password: { type: String, required: true } }); const FormDataModel = mongoose.model('FormData', formDataSchema); module.exports = FormDataModel;

* **User Registration**

The **app.js** handles user registration by hashing passwords and saving data to the database:

app.post('/formdata', async(req, res) => { // ... Code for password hashing and data saving ... });

* **User Login**

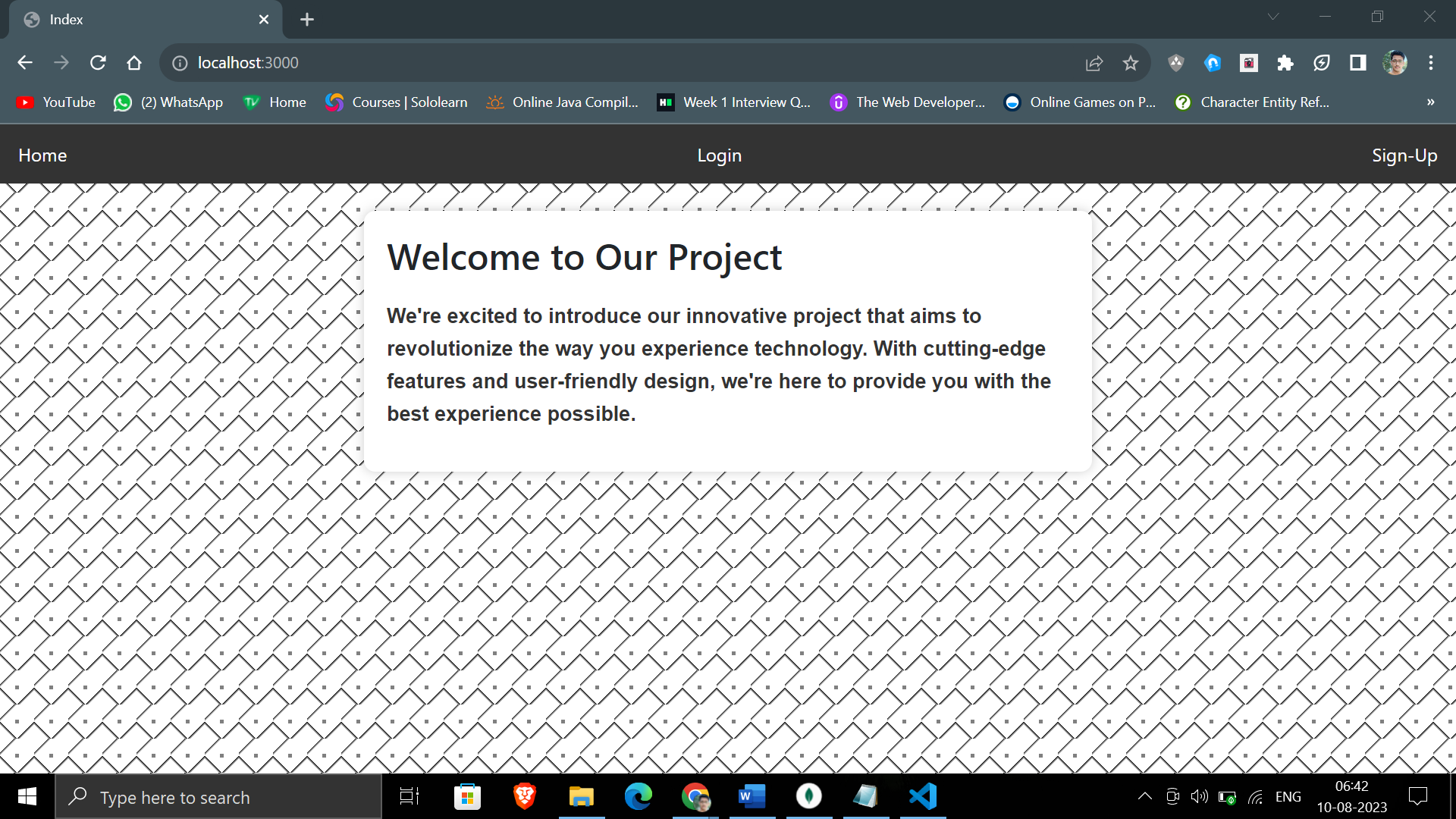
User login is managed in the **app.js** file, including password validation and token creation:

app.post('/loginPage', async(req, res) => { // ... Code for user login, password validation, and token creation ... });

**6. Conclusion**

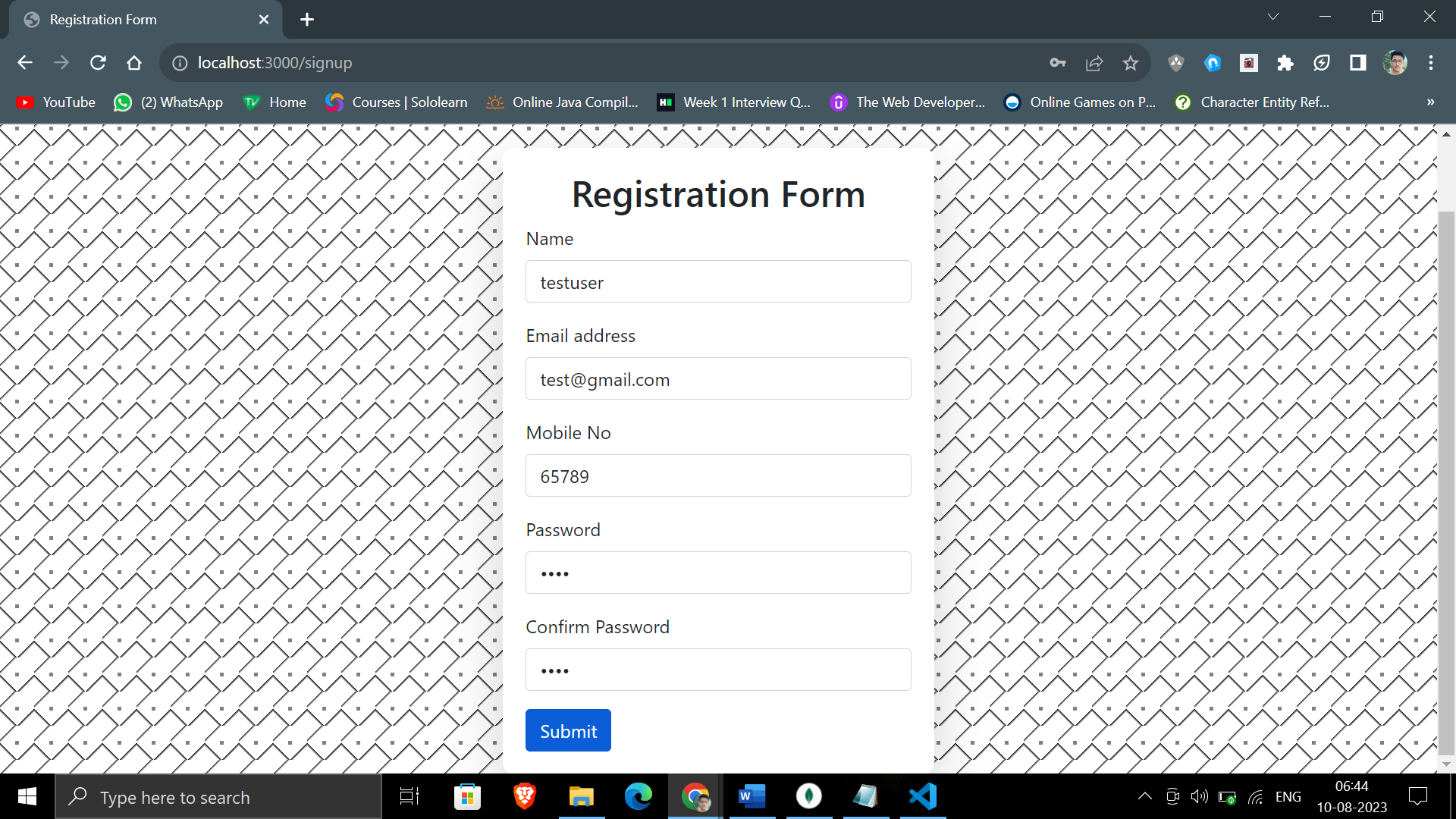
The **Secure User Registration and Login** project demonstrates a secure and efficient user authentication system using Node.js, Express, MongoDB, and bcrypt. By following this documentation and analyzing the provided code snippets, you have gained a deeper understanding of how to implement secure user registration and login functionality in your web applications.

**Screen Shots of the project**



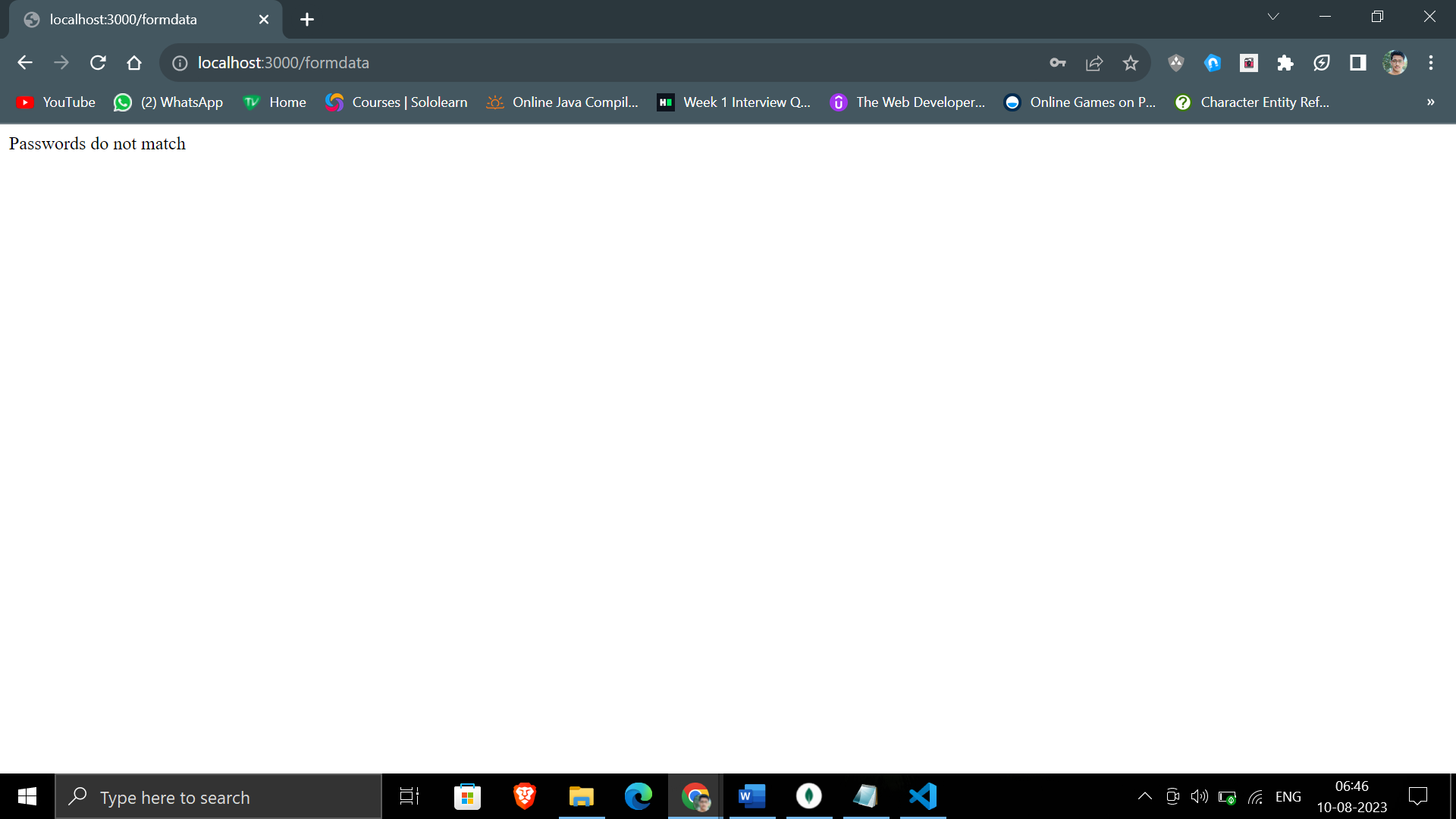
**Screenshot 1 - Homepage:**

* Welcome to Our Project: A captivating webpage with a gradient background and sleek navigation bar.
* The webpage serves as an introduction to a project, providing essential information and enticing users to learn more.
* The navigation bar offers links to different sections, including "Home," "Login," and "Sign-Up," facilitating seamless exploration.



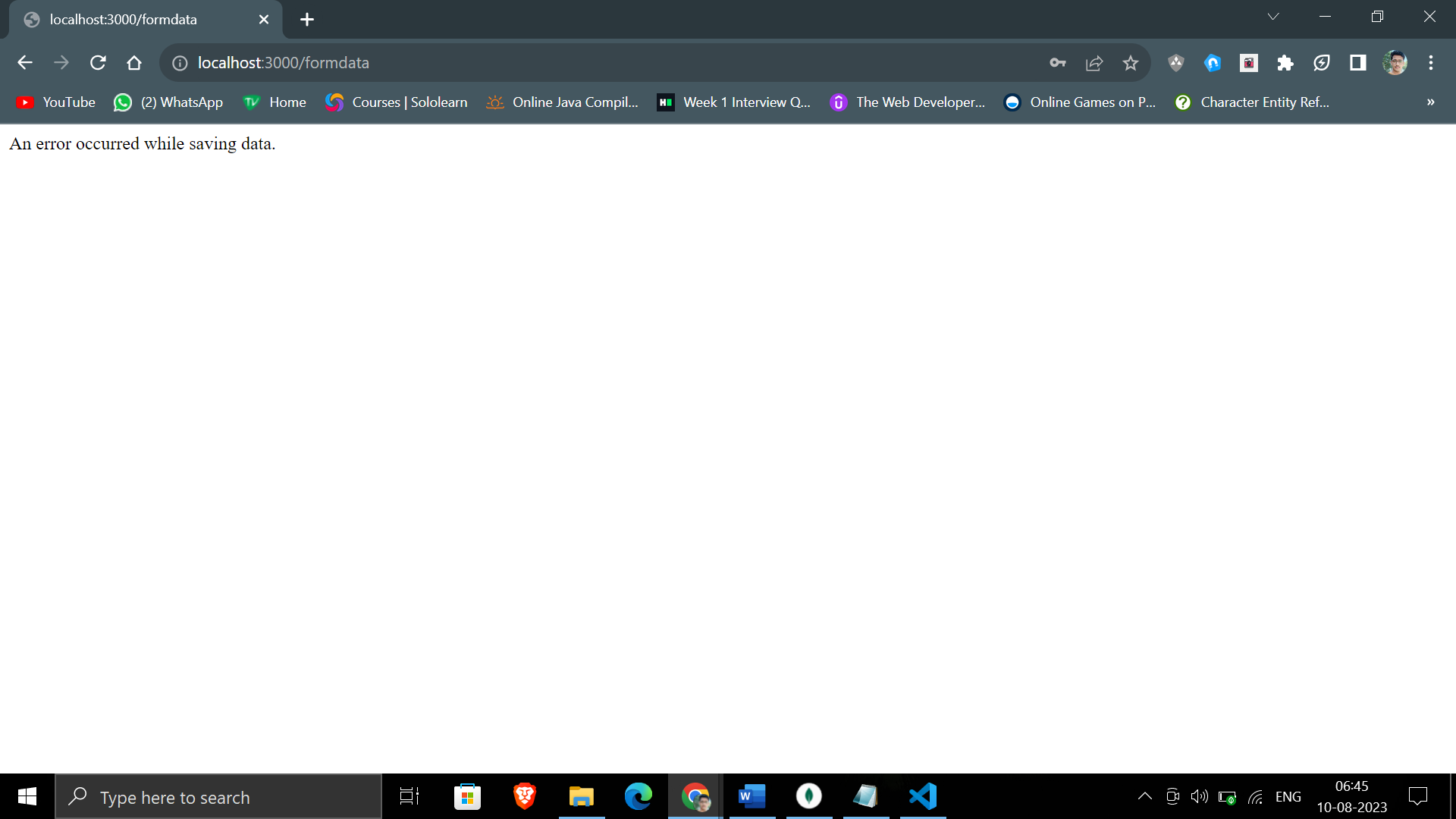
**Screenshot 2 - Registration Form Page:**

* The registration form page displays a clean layout with a white container over a background gradient pattern.
* Users can enter their name, email, mobile number, password, and confirm password.
* Navigation bar provides links to 'Home' and 'Login' pages.



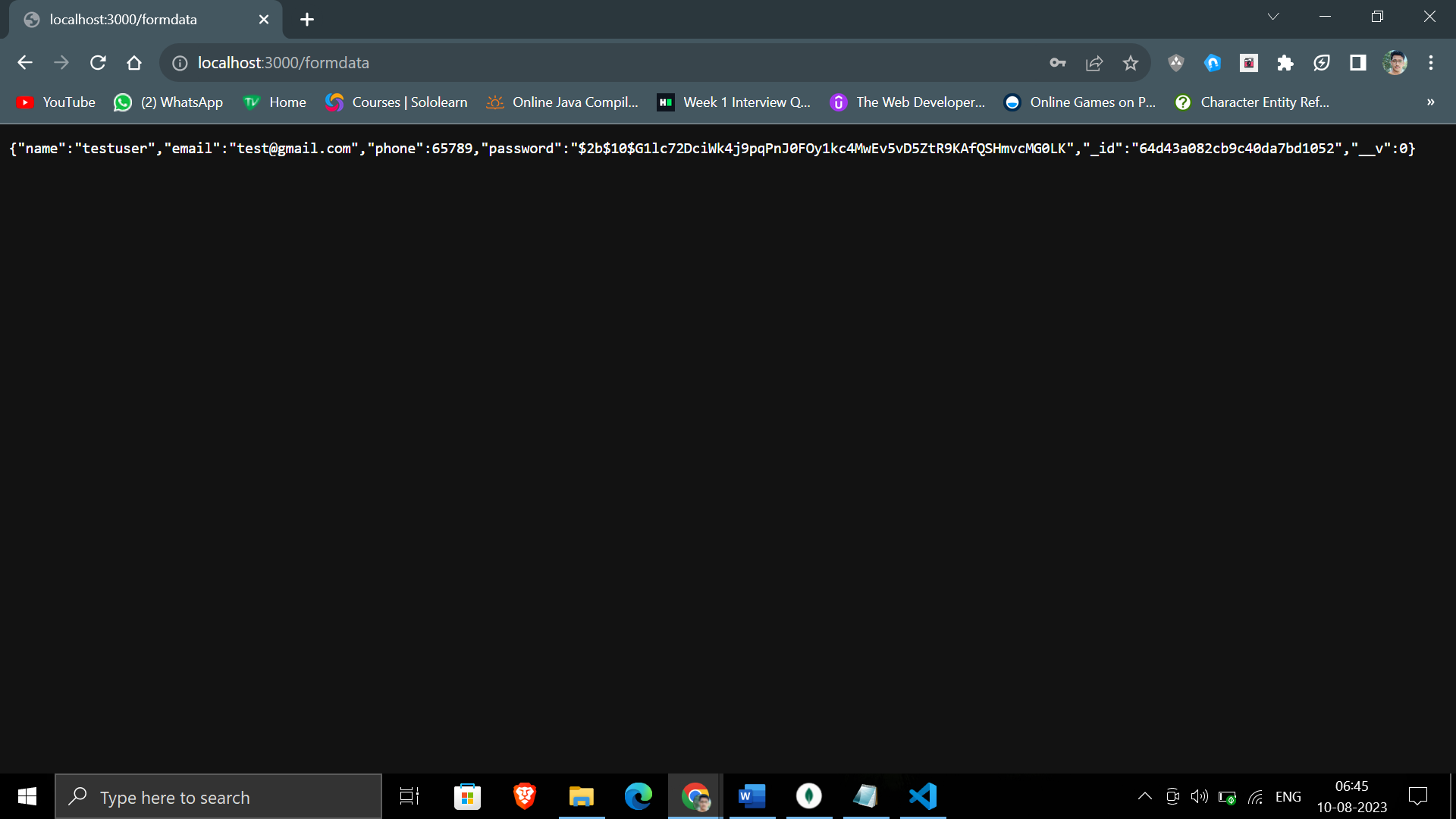
**Screenshot 3 - Passwords Do Not Match:**

* Registration error: Passwords do not match.
* Validation prevents submission and prompts the user with an error message.
* Users need to ensure password and confirm password fields match before proceeding.



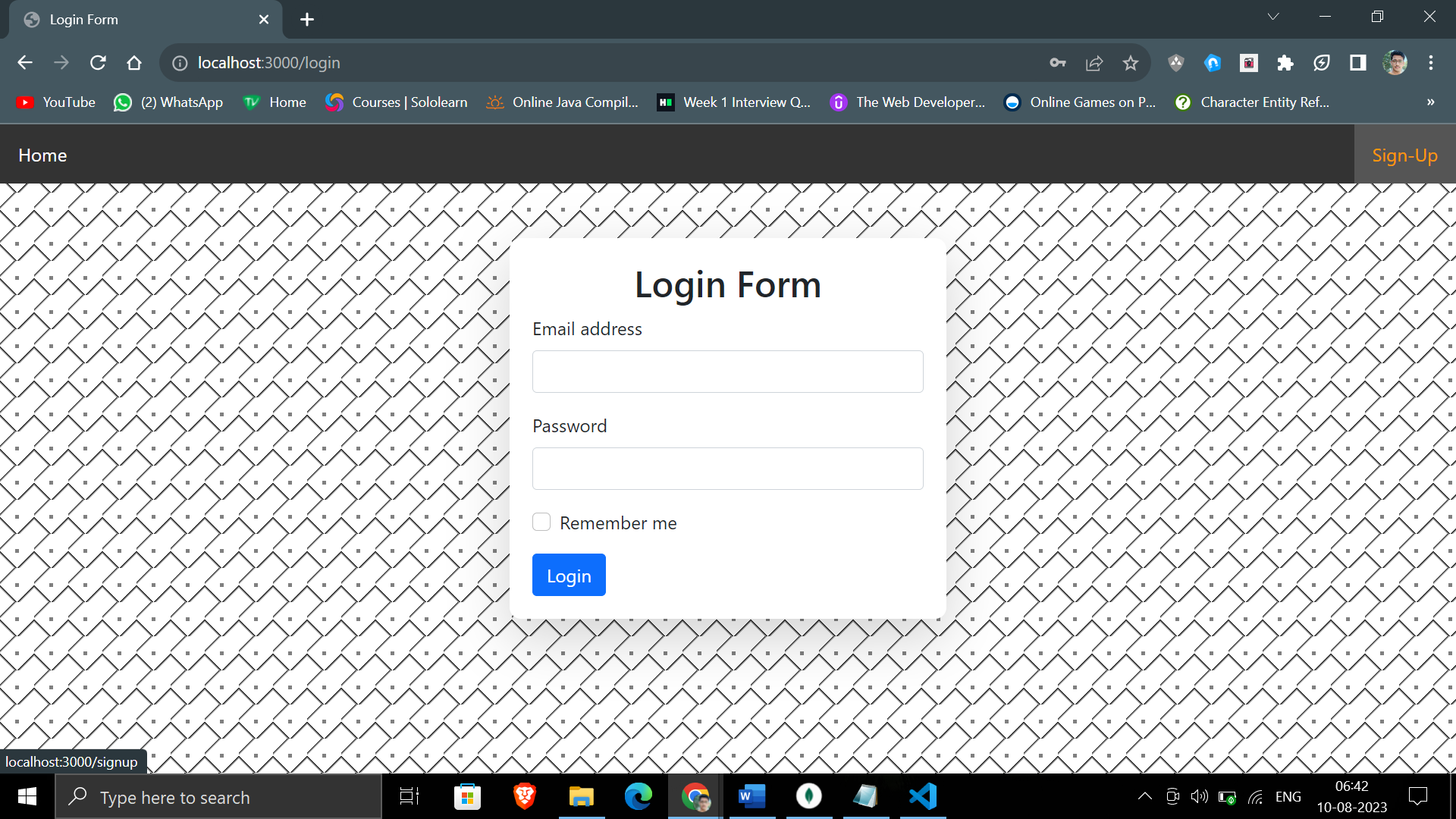
**Screenshot 4 - Error Saving Data:**

* Error message after submitting registration form.
* An issue occurred while attempting to save data to the server or database.
* Users are informed that an error occurred and may be directed to try again later.



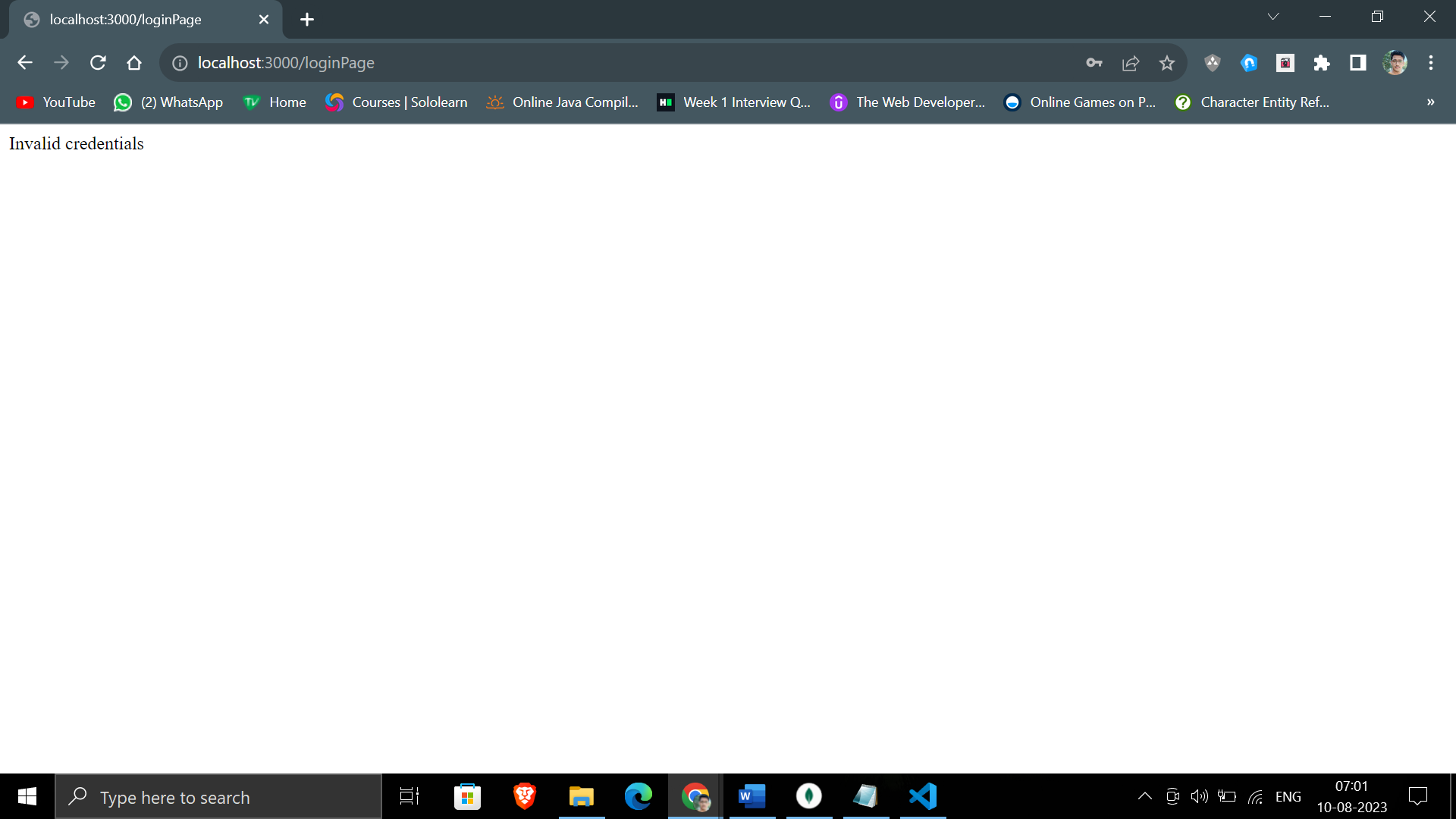
**Screenshot 5 - Data Saved with Hashed Password:**

* Confirmation of successful registration with hashed password.
* User registration data has been successfully saved in the database.
* Passwords are securely hashed before storage, ensuring data privacy and security.
* Sensitive user information, such as passwords, is stored in a secure and non-readable format.



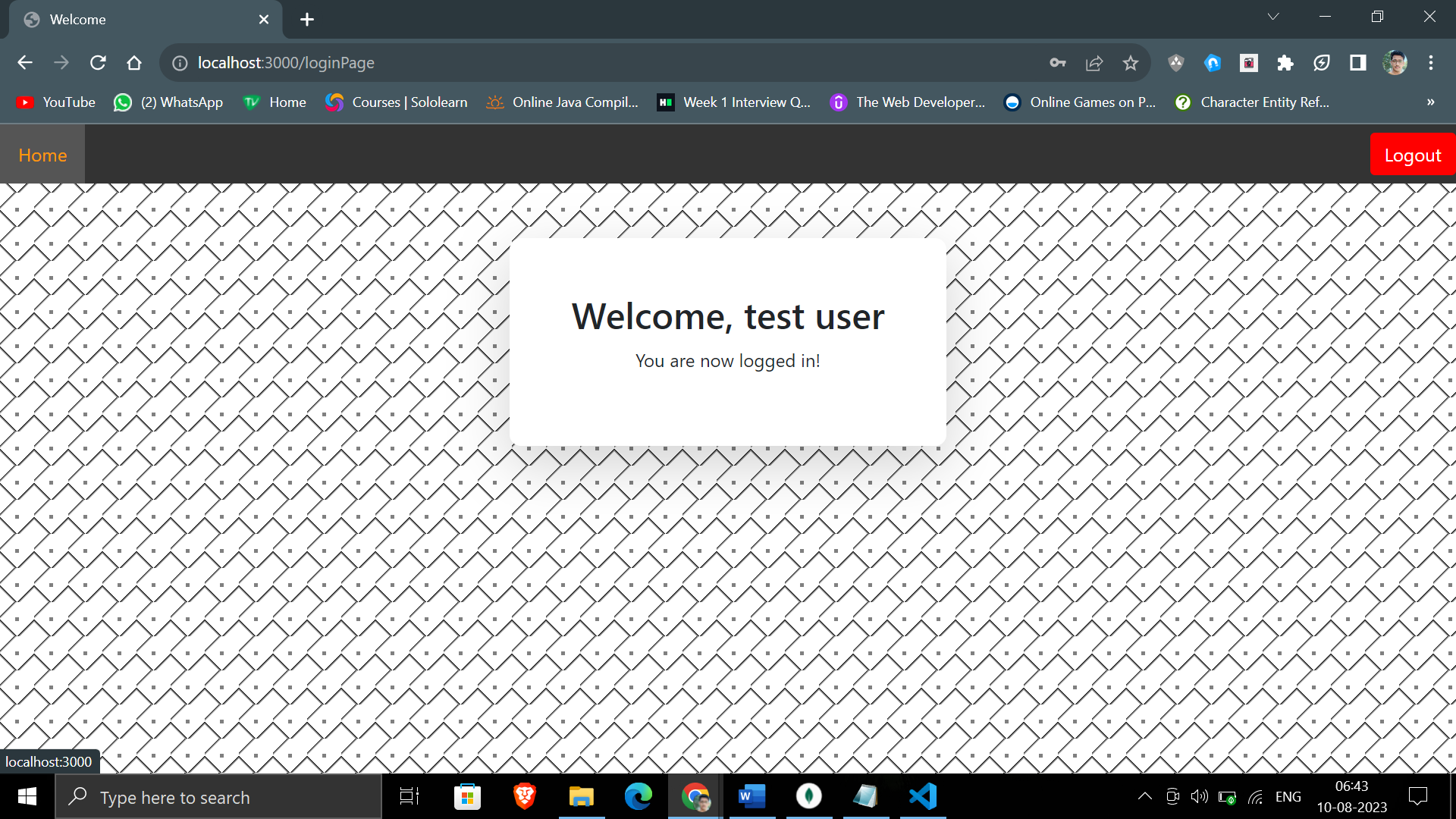
**Screenshot 6 - Login Page:**

* The login page features a visually appealing layout with a white container on a gradient background.
* Users can input their email and password to log in.
* Navigation bar provides links to 'Home' and 'Sign-Up' pages.
* A 'Remember me' option allows users to stay logged in.



**Screenshot 2 - Invalid Credentials:**

* Error message displayed after entering invalid login credentials.
* Users are notified with an 'Invalid credentials' message when incorrect email or password is submitted.
* This feature enhances security by preventing unauthorized access attempts.
* Users are encouraged to double-check their email and password and try again.



**Screenshot - Welcome Page:**

* The welcome page greets the user with a personalized message.
* The user's name, obtained from their registration, is displayed in the 'Welcome, <<user>>' message.
* This feature adds a personalized touch to the user's experience upon successful login.
* Users can navigate to the 'Home' page or log out using the navigation bar.