

Web Page Mini Project Report

Executive Summary

The Web Page Mini Project is a simple web application designed to capture user input through a web page and store the submitted data in MongoDB. This report provides an overview of the project, its features, installation instructions, and future considerations.

Table of Contents

- 1. Introduction
- 2. Project Overview
- 3. Features
- 4. Installation
- 5. Usage
- 6. Technologies Used
- 7. Conclusion
- 8. Screenshots

1. Introduction

The Web Page Mini Project is a practical exploration of web development fundamentals, spotlighting the integration of a user-friendly web page with MongoDB, a NoSQL database. This initiative aims to provide a hands-on experience for beginners, emphasizing the seamless interplay between frontend and backend technologies. Focused on simplicity and user experience, the project allows users to submit data through a form, with the submitted information securely stored in MongoDB. The limited scope enables a concentrated exploration of key technologies, offering foundational insights into the collaborative efforts of React.js for the frontend, Next.js for server-side rendering, and MongoDB for data storage. This report will delve into the project's architecture, features, installation guidelines, and potential considerations for future expansion.

2. Project Overview

2.1 Architecture

The project follows a simple architecture:

- Frontend: React.js for the user interface.
- Backend: Next.js for server-side rendering and API routes.
- Database: MongoDB for storing user data.

2.2 Functionality

The main functionality includes:

- User input form with fields for full name, email, and message.
- Data submission to a MongoDB database.
- Basic error handling and feedback to users.

3. Features

The Web Page Mini Project includes the following features:

- User-friendly web page with an input form.
- Integration with MongoDB for data storage.
- Basic validation and error handling on the server-side.

4. Installation

To install the project locally, follow the steps outlined in the [Getting Started](#getting-started) section of the README file.

5. Usage

Users can visit the web page, fill out the form, and submit data. The submitted data is stored in MongoDB.

6. Technologies Used

The project leverages the following technologies:

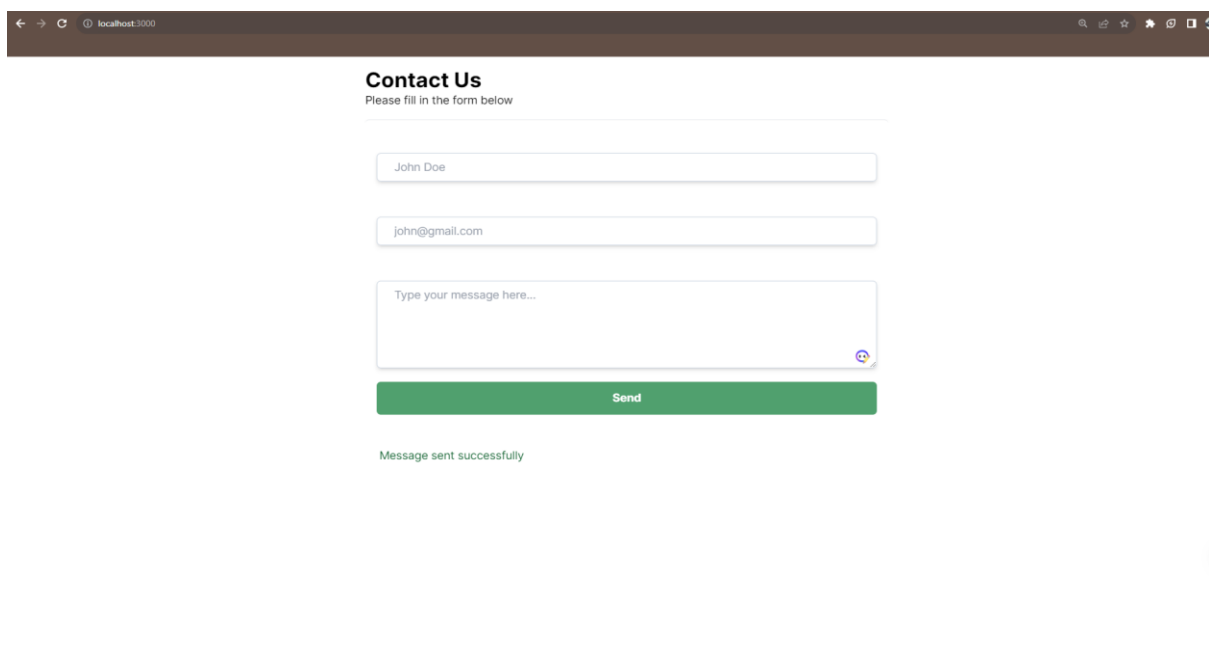
- React.js for the front end.
- Next.js for server-side rendering and API routes.
- MongoDB for data storage.

7. Conclusion

The Web Page Mini Project serves as a foundational example of web development, showcasing the integration of a web page with MongoDB. The project provides valuable insights into building interactive web applications and handling user input.

Git Hub : https://github.com/geijinchan/Web_Page

8. Screen Shot



The screenshot displays a web browser window with the address bar showing 'localhost:3000'. The main content area features a 'Contact Us' form. The form title is 'Contact Us' with a subtitle 'Please fill in the form below'. There are three input fields: a text field containing 'John Doe', an email field containing 'john@gmail.com', and a larger text area containing 'Type your message here...'. A green 'Send' button is positioned below the message field. At the bottom of the form, a green message states 'Message sent successfully'.

Contact Us

Please fill in the form below

Abhishek

abhishekravikumar24@gmail.com

Contact Us page with React and Node Js

Send

Message sent successfully

The screenshot displays the MongoDB Atlas web interface. The top navigation bar includes the Atlas logo, a user profile for 'Abhishek', and links for 'Access Manager' and 'Billing'. The main header shows 'SA_Mini_Project' and 'Data Services'. The left sidebar contains a navigation menu with sections like 'Overview', 'DEPLOYMENT', 'Database', 'SERVICES', and 'SECURITY'. The 'Database' section is expanded, showing 'Data Lake', 'Triggers', 'Data API', 'Data Federation', 'Search', 'Stream Processing', 'Backup', 'Database Access', 'Network Access', and 'Advanced'. The main content area is titled 'test.contacts' and shows a list of documents. The documents are as follows:

Document
<pre>{ "_id": "655b6a82c1d274f6b373744b", "fullName": "Abhishek", "email": "abhishekravikumar24@gmail.com", "message": "xxxx", "date": "2023-11-20T14:17:38.135+00:00", "__v": 0 }</pre>
<pre>{ "_id": "655b6b09c1d274f6b373744b", "fullName": "Abhishek", "email": "abhishekravikumar24@gmail.com", "message": "Contact Us page with react and node js", "date": "2023-11-20T14:19:53.895+00:00", "__v": 0 }</pre>
<pre>{ "_id": "655b6b2cc1d274f6b373744b", "fullName": "Abhishek", "email": "abhishekravikumar24@gmail.com", "message": "Contact Us page with React and Node Js", "date": "2023-11-20T14:20:20.513+00:00", "__v": 0 }</pre>

At the bottom of the interface, there is a 'System Status' section indicating 'All Good' and a footer with copyright information for MongoDB, Inc. and links to 'Status', 'Terms', 'Privacy', 'Atlas Blog', and 'Contact Sales'.