TEMPLE ANALYTICS SYSTEM

USING MERN STACK

MINI PROJECT REPORT

Submitted by

BHUVANIKA S (9517202109011)
RAJAKUMARI S (9517202109042)
SUJI S (9517202109051)

in

19AD581 –NO-SQL DATABASES LABORATORY

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MEPCO SCHLENK ENGINEERING COLLEGE

SIVAKASI

OCTOBER 2023

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI

AUTONOMOUS

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

BONAFIDE CERTIFICATE

This is to certify that it is the bonafide work of **RAJAKUMARI.S (9517202109042), SUJI.S (9517202109051) , BHUVANIKA.S(9517202109011)** for the mini project titled **"LOAD BALANCING USING SOCKET "** in 19AD551 – Computer Networking Laboratory during the fifth semester July 2023 – October 2023 under my supervision.

SIGNATURE
Dr.P.Thendral M.E.,Ph.D
Associate Professor
Artificial Intelligence and Data Science
ScienceMepco Schlenk Engineering College
College Sivakasi - 626 005
Virudhunagar District

SIGNATURE
Dr.J.Angela Jennifa Sujana M.E.,Ph.D
Professor&Head
Artificial Intelligence and Data
Mepco Schlenk Engineering
Sivakasi – 626 005
Virudhunagar District

ABSTRACT

The Temple Analytics System is a comprehensive software solution designed to streamline and enhance the administrative and operational processes of temples and religious institutions. Temples hold significant cultural and religious importance, and managing their day-to-day activities efficiently is essential. This project aims to provide a user-friendly, integrated platform that automates various temple management tasks, ultimately improving overall operational efficiency and visitor experience. It enables the secure recording and analytics of between two years total number of temples and revenue and gold and number of visitor state-wise and deity-wise .Financial transparency and accountability are achieved through detailed transaction logs. Generate reports and analytics to gain insights into the temple's performance, including financial reports, attendance records, and more. Access to sensitive data is tightly controlled, with different roles and permissions for admin and client.The user interface is designed to be intuitive, allowing admin with varying technical expertise to analyse and use the system effectively. In a world increasingly dependent on technology, the Temple Management System is a crucial tool to modernize and enhance the management of religious institutions. It ensures transparency, efficiency, and convenience for both temple administrators and visitors, ultimately fostering a deeper connection to the spiritual and cultural heritage these temples represent.

TABLE OF CONTENTS

ACKNOWLEDGEMENT

First and foremost we **praise and thank "The Almighty",** the lord of all creations, who by his abundant grace has sustained us and helped us to work on this project successfully. We really find unique pleasure and immense gratitude in thanking our respected management members**,** who is the backbone of our college. A deep bouquet of thanks to respected Principal **Dr.S.Arivazhagan M.E.,Ph.D.,** for having provided the facilities required for our mini project.

We sincerely thank our Head of the Department **Dr. J. Angela Jennifa Sujana M.E.,Ph.D** Professor & Head, Department of Artificial Intelligence and Data Science, for her guidance and support throughout the mini project.

We also thank our guide **Dr.P.Thendral., M.E.,Ph.D**, Associate Professor,Department of Artificial Intelligence and DataScience for their valuable guidance and it is great privilege to express our gratitude to them.

We extend our heartfelt thanks and profound gratitude to all the faculty members of Artificial Intelligence and Data Science department for their kind help during our mini project work.We also thank our parents and our friends who had been providing us with constant support during the course of the mini project work.

CHAPTER 1

INTRODUCTION

1.1 Introduction

. Temples and religious institutions have been an integral part of human civilization for centuries, serving as centers of spirituality, community, and cultural heritage. Managing these sacred places efficiently is of paramount importance to ensure the smooth conduct of religious activities, administration, and the overall welfare of the devotees and visitors. To address the complex and evolving needs of temple management, the Temple Management Project was conceived. Historically, temple management involved a myriad of manual processes, extensive paperwork, and intricate record-keeping, often leading to inefficiencies and challenges in maintaining transparency and accountability. With the advent of digital solutions, the Temple Management Project endeavors to bridge this gap by providing a comprehensive software system that caters to the diverse requirements of temple administration. Transparent financial management is a cornerstone of this project. By maintaining meticulous records of donations, expenses, and income, the system ensures that the financial affairs of the temple are conducted with the utmost transparency and accountability. Security is a paramount concern when dealing with sensitive temple data. The Temple Management Project includes robust security measures to protect against unauthorized access or data breaches.

1.2 Objective for the project

- To streamline administrative tasks and reduce manual effort, enhancing the overall efficiency of temple analytic processes. This includes analysis of between two years state-wise ,deity-wise total number of visitors and total revenue and gold received and deity-wise total number of temples

- To assist in managing temple finances effectively, enabling the institution to maintain its operations .

- To provide a user-friendly interface that is accessible to individuals with varying levels of technical expertise, making it easier for admin and administrators to manage temple details.

- To implement robust security measures to safeguard sensitive temple data, ensuring that it remains protected from unauthorized access or breaches.

1.3 Problem Statement

- To analyze the number of visitors and total revenue and total golds that are received in the period of two years grouping by state-wise and deity-wise

1.4 Scope of the project

The scope of the Temple Management Project encompasses a wide range of features and functionalities aimed at modernizing and enhancing the administration of temples and religious institutions.

- Event and Pooja Management:
  This helps in Scheduling rituals with the Maintenance of a calendar of activities and reporting analysis of visitor statistics.
- Online Donations:
  It helps to integrate with online payment gateways for convenient and secure online

CHAPTER 2

ARCHITECTURE

2.1 Aggregate Model

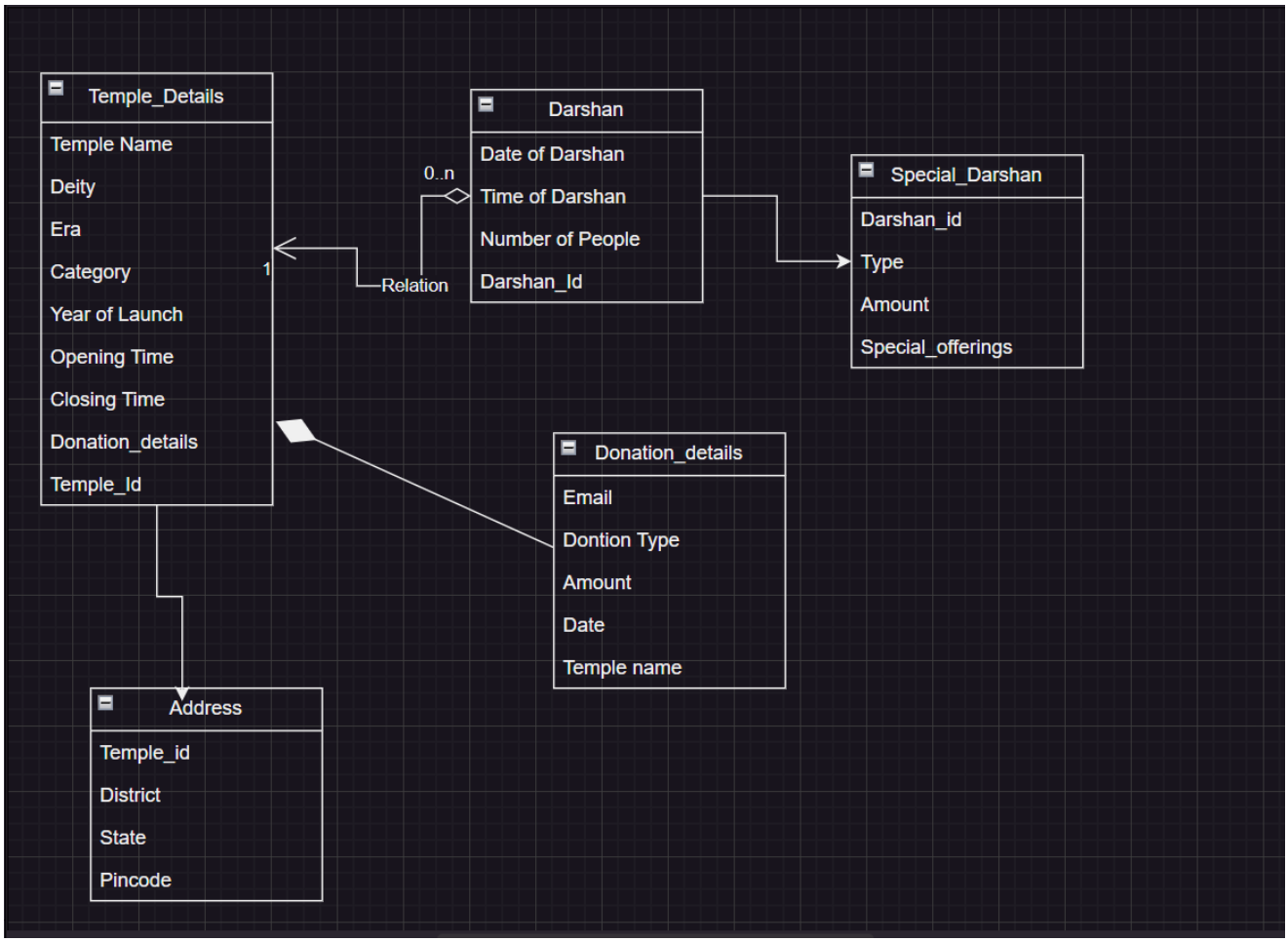


Figure 2.1.1 – Mongo DB collection model

2.2 FRONTEND DESIGN

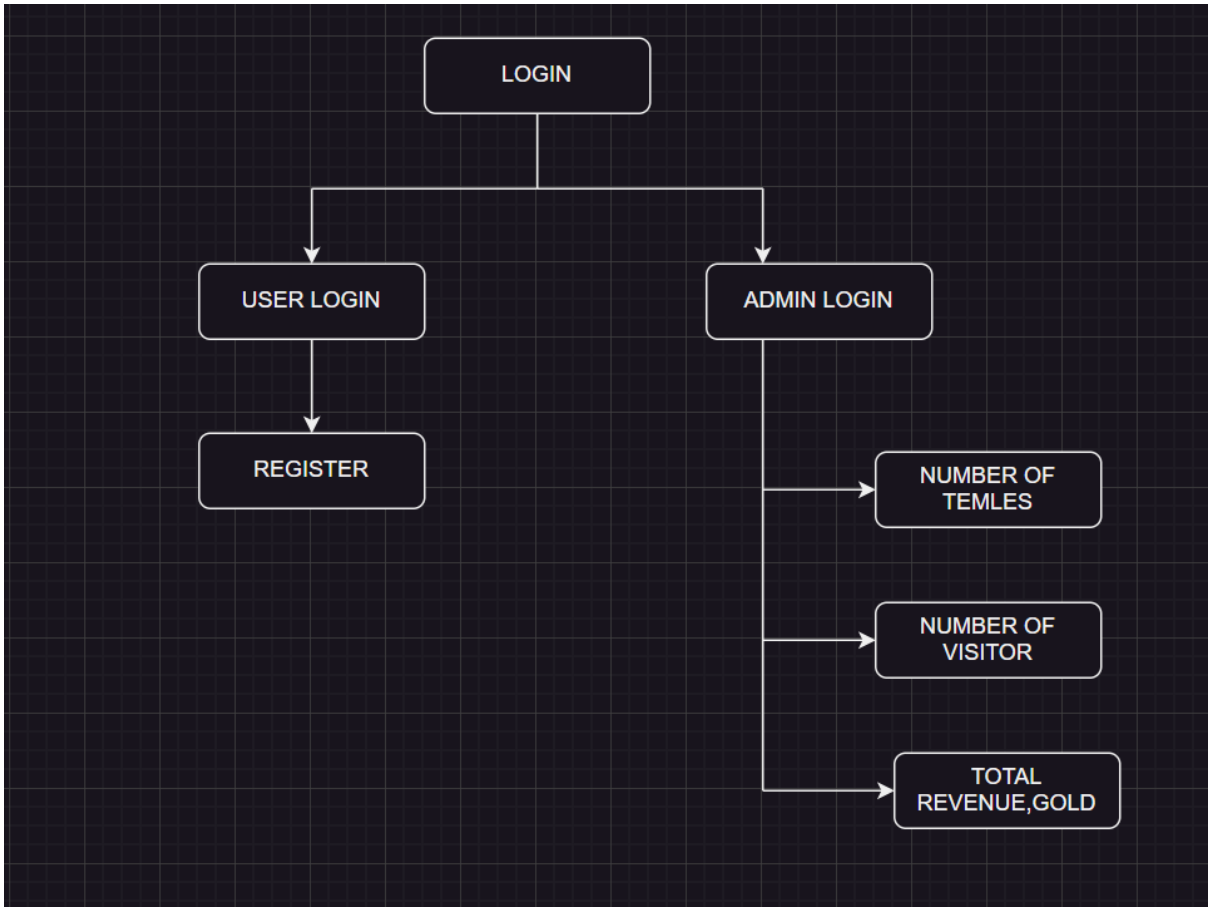


Figure 2.1.2 – FrontEnd Design

2.2UseCase Aggregate

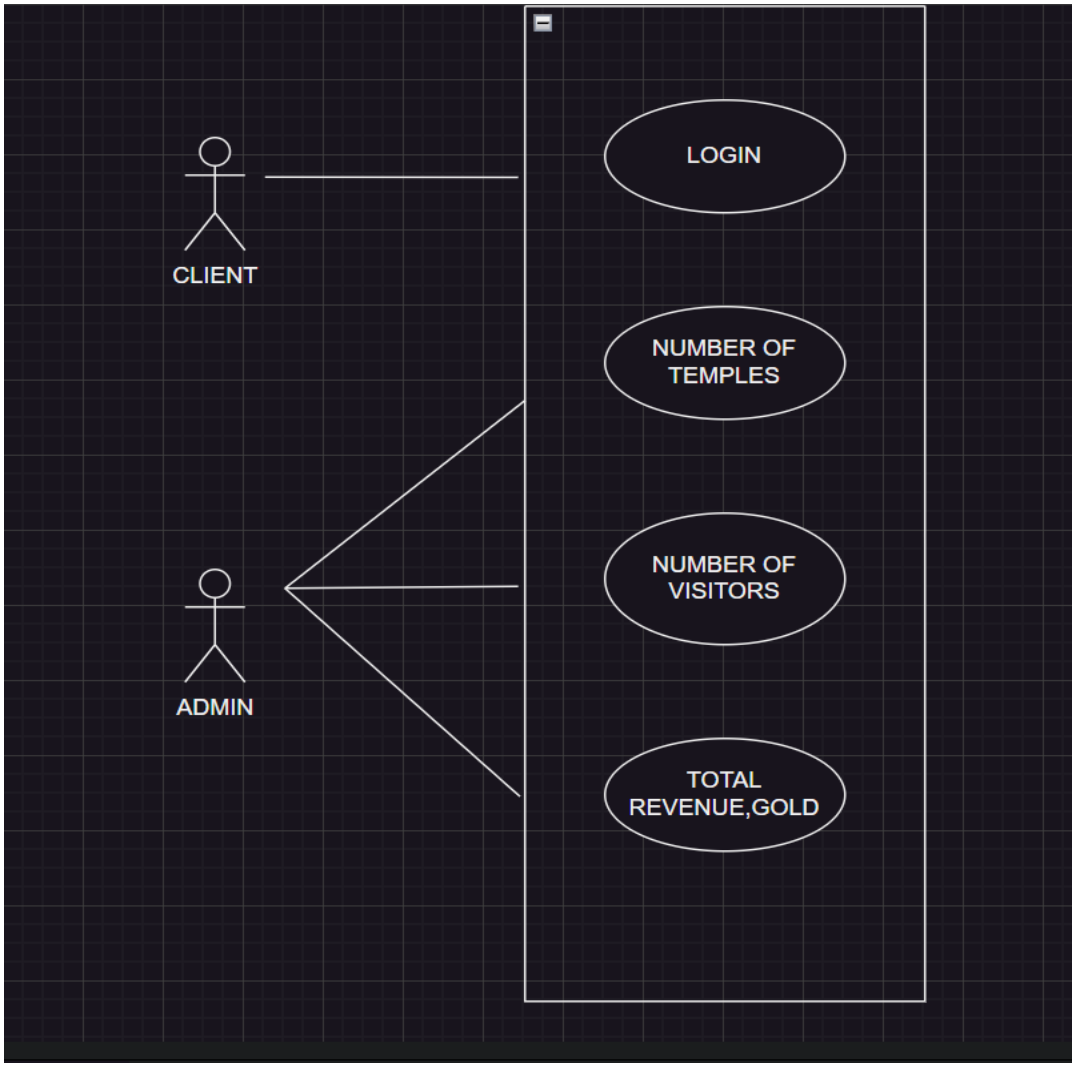


Figure 2.2.1 – Use-Case Diagram

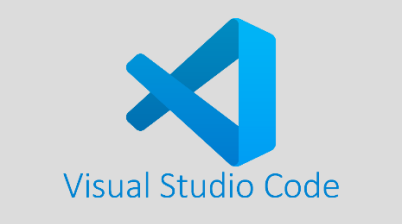
CHAPTER 3

SYSTEM REQUIREMENTS

3.1 Software Component

VISUAL STUDIO CODE

Visual Studio Code is an integrated development environment made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality



MONGO DB

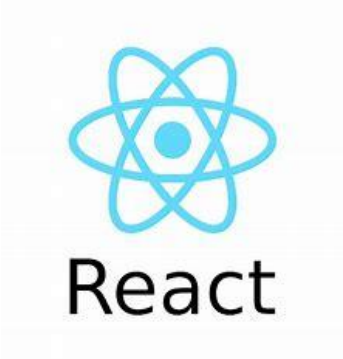
MongoDB is a popular, open-source, NoSQL database management system known for its flexibility, scalability, and ease of use. It belongs to the family of document-oriented databases, which are designed to store, retrieve, and manage data in a way that is more aligned with modern application development. MongoDB is schema-less, you can change the structure of your documents as your application evolves, making it suitable for agile development and accommodating evolving data requirements. Database - MongoDB is a NoSQL, document-oriented database that stores data in a flexible, JSON-like format called BSON. It is well-suited for handling unstructured or semi-structured data and provides scalability and high performance. MongoDB is used to store and manage the application's data.

NODE JS

Node.js is an open-source, server-side JavaScript runtime environment that allows developers to build and run web applications on the server. It is built on the V8 JavaScript engine from Google and is designed to be lightweight, efficient, and scalable. It is cross-platform, which means it can be run on various operating systems, making it versatile for server development. It can be easily scaled horizontally to handle increased traffic by adding more server instances.

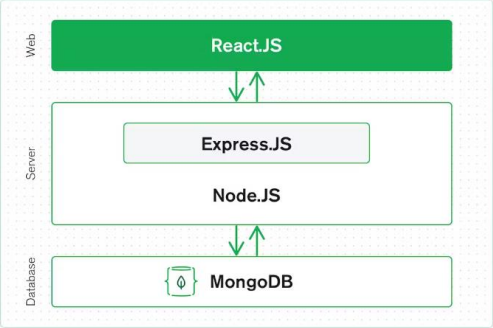
REACT JS

React (also known as React.js or ReactJS) is an open-source JavaScript library created and maintained by Facebook. It is a popular tool for building user interfaces (UIs) for web applications. React is known for its efficiency, performance, and ease of use, and it has become a fundamental part of modern web development. React enforces a one-way data flow, which means data flows downward from parent components to child components. React uses JSX, an extension of JavaScript that allows developers to write HTML-like code within their JavaScript files. JSX is transpiled into JavaScript and is used to define the structure and appearance of components.



MERN STACK

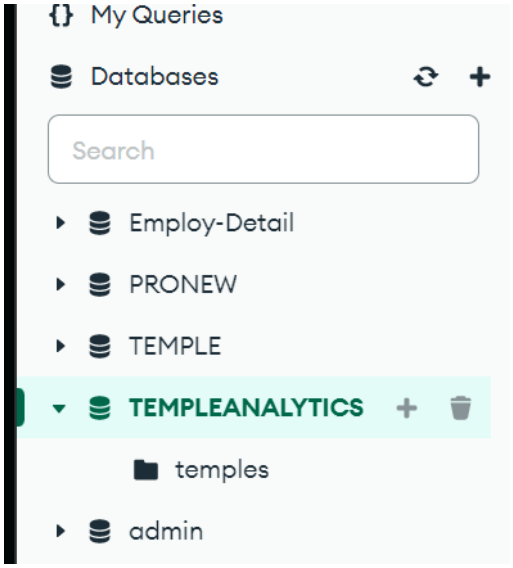
The MERN Stack is a popular and powerful web development stack that consists of four key technologies: MongoDB, Express.js, React, and Node.js. Each component plays a unique role in building full-stack web applications.

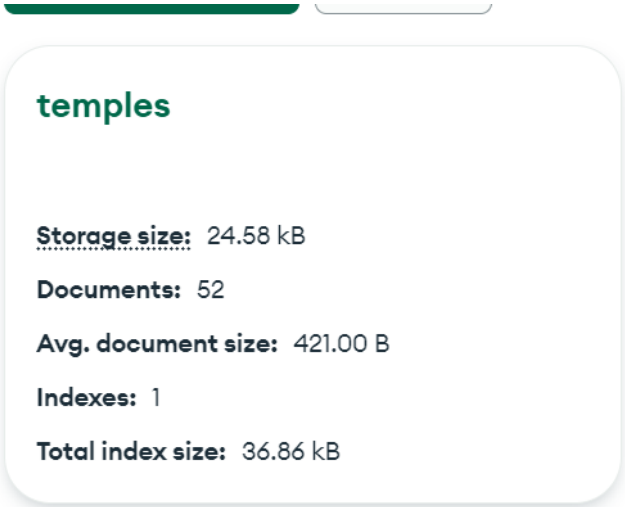


CHAPTER 4

IMPLEMENTATION

DATABASE:TEMPLEANALYTICS





COLLECTIONS:

1.TEMPLES

This Collection stores information about temples.

d_templer	d_deity	d_district	d_state	d_pincode	d_era	d_categor	d_count	d_year_of	d_revenue	d_gold	d_opening	d_closing_	d_email	d_descript
Tirumala \	Vishnu	Other	Andhra Pr	123456	Chera	Small Size	5	1973-10-2	10000	50	06:30	08:00	Venkatesv	Venkatesv
Kashi Vish	Shiva	Other	Uttar Prad	123456	Chera	Midsize	48	2022-10-2	15000	69.99	06:30	08:00	Vishwanat	Kashi Vish
Meenaksh	Parvati	Madurai	Tamil Nad	123456	Pandya	Large Size	90	1996-10-2	150000	59.98	06:30	08:00	Meenaksh	Meenaksh
Jagannath	Shiva	Other	Uttar Prad	123456	Chera	Small Size	100	1999-10-2	199997	3.98	06:30	08:00	Jagannath	Jagannath
Ramanath	Shiva	Rameswa	Tamil Nad	123456	Chola	Midsize	147	2006-10-2	25000	4.48	06:30	08:00	Ramanath	Ramanath
Nataraja	Shiva	Coimbato	Tamil Nad	123456	Pandya	Large Size	200	2009-10-2	25000	4.99	06:30	08:00	Nataraja@	Nataraja :)
thanumal	Vishnu	Kanyakum	Tamil Nad	787878	Chola	Midsize	500	1996-10-2	49999	2.48	07:00	23:00	thanumal	thanumal
Sree Padr	Vishnu	Other	Kerala	898976	Chola	Midsize	5000	1986-10-2	199999	100	06:00	20:00	Padmanat	Padmana
Brihadees	Shiva	Thanjavur	Tamil Nad	123456	Pandya	Small Size	7890	1995-10-2	600000	70.64	07:00	18:00	Brihadees	Brihadees
Thirupath	Murugan	Other	Tamil Nad	123456	Chola	Midsize	7000	2005-10-2	600000	70.64	08:00	20:22	Thirupath	Thirupath
Tirupati Ba	Balaji	Other	Andhra Pr	123456	Chola	Small Size	200	2003-10-2	50000	100	04:00	07:00	TirupatiBa	Tirupati Ba
Siddhivina	Vinayaka	Other	Maharash	567898	Pandya	Large Size	500	1993-10-2	69999	50	04:00	07:00	Siddhivina	Siddhivina
Kedarnath	Kedareshv	Other	Uttarakha	987656	Chera	Small Size	800	1993-10-2	69999	50	05:00	07:00	Kedarnath	Kedarnath
Harmandi	Sikh Guru	Other	Puducherr	987656	Pandya	Midsize	800	1997-10-2	70011	99.98	05:00	07:00	Harmandi	Harmandi
Badrinath	Badrinara	Other	Uttar Prad	456789	Chola	Small Size	800	1983-10-2	90011	99.98	05:00	07:00	Badrinath	Badrinath

Data Types:

d_templename: String,

d_deity: String,

d_district: String,

d_state: String,

d_pincode: String,

d_era: String,

d_category: String,

d_count:String,

d_year_of_built: String,

d_revenue:String,

d_gold:String,

d_opening_time: String,

d_closing_time: String,

d_email: String,

d_description: String,

d_start_date:String,

d_end_date:String,

templeCount:String,

4.1 SAMPLE OUTPUT:

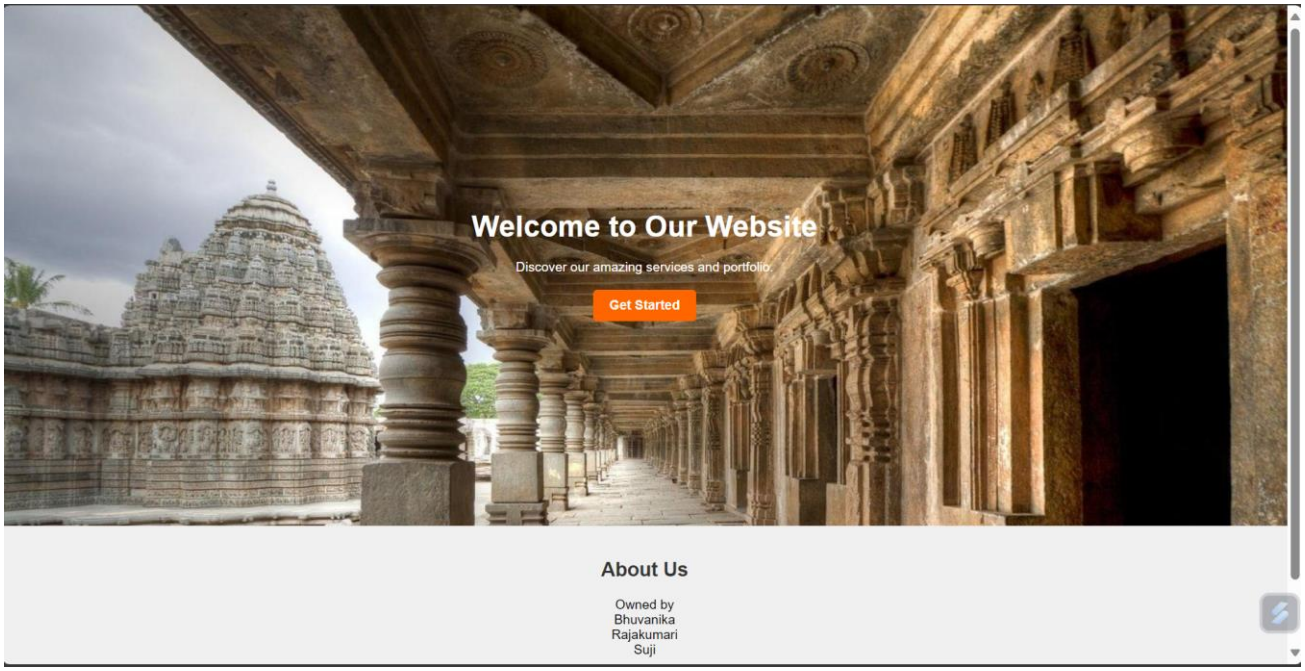


Fig 4.1.1 Home Page

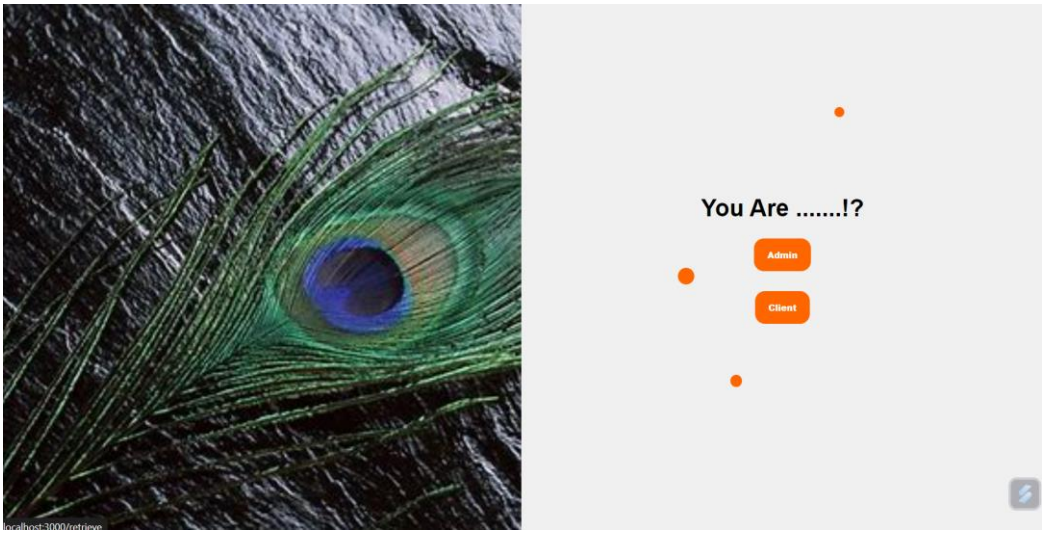


Fig 4.1.2 Next Page

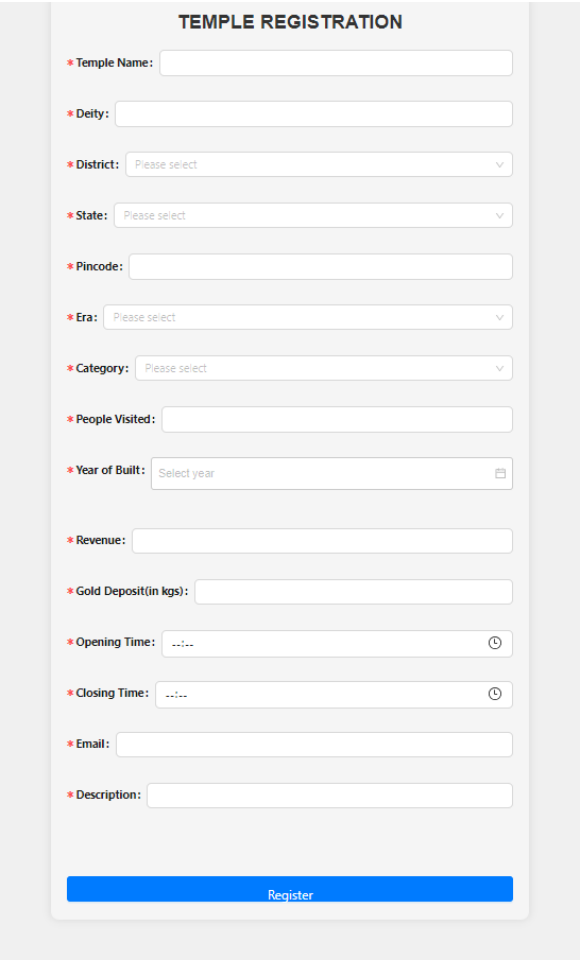


Fig 4.1.3 Registration Page

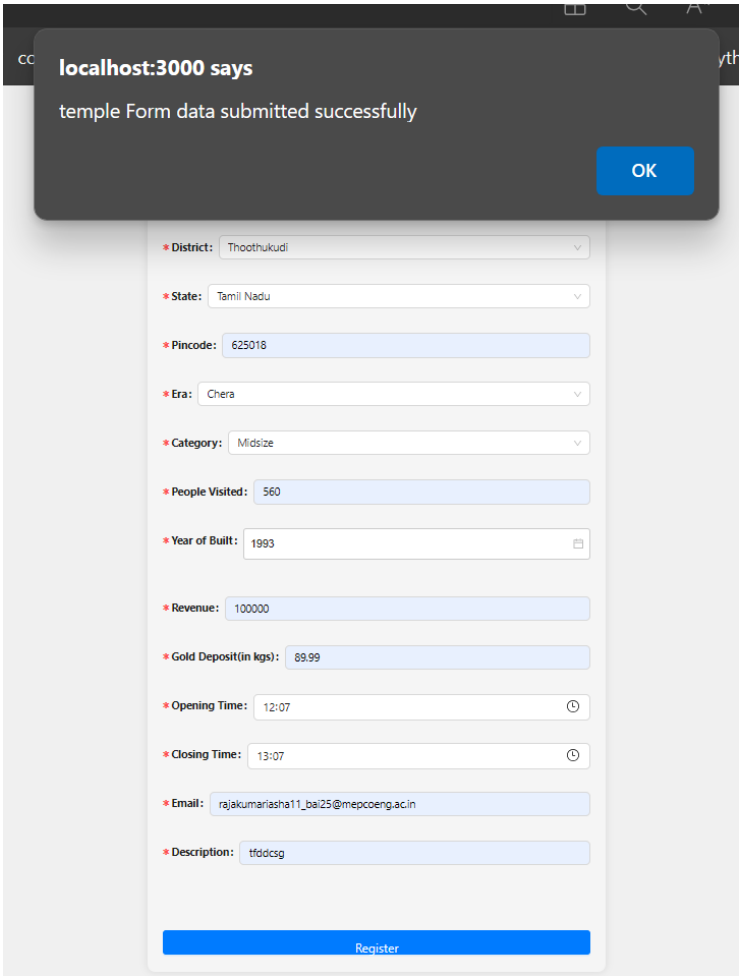


Fig 4.1.4 Registration Page II

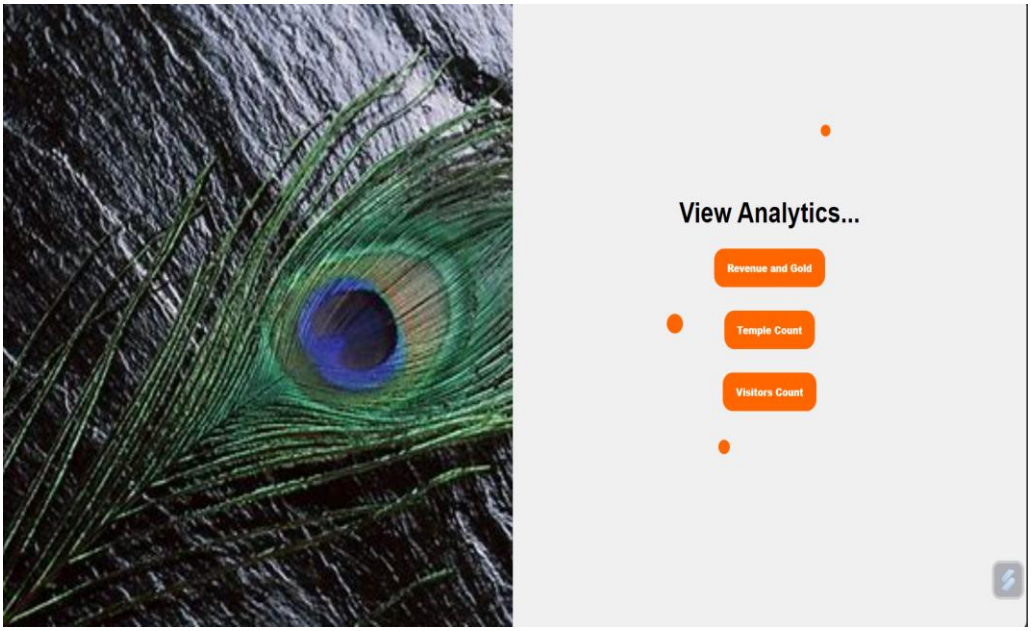


Fig 4.1.5 Analytics

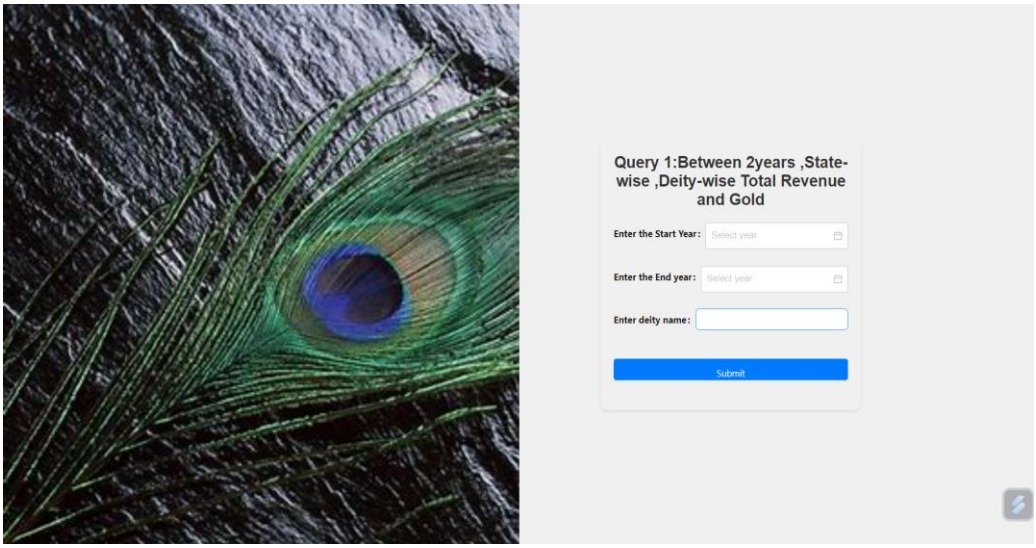


Fig 4.1.6 Query1

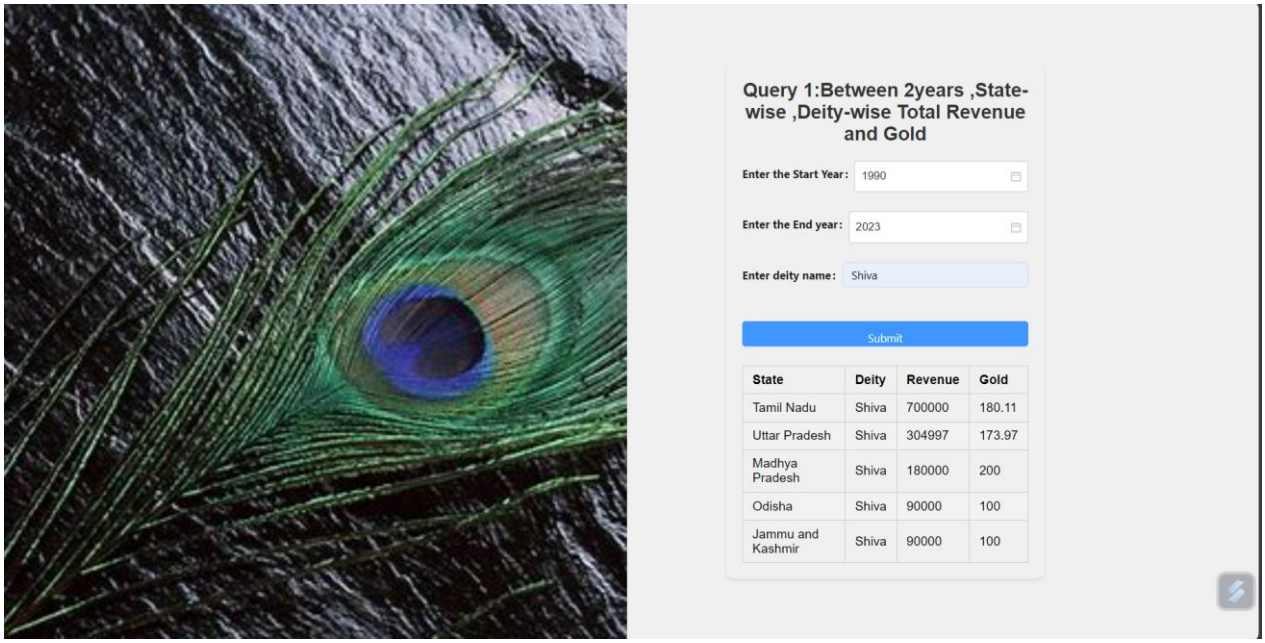


Fig 4.1.7 Query1 Result

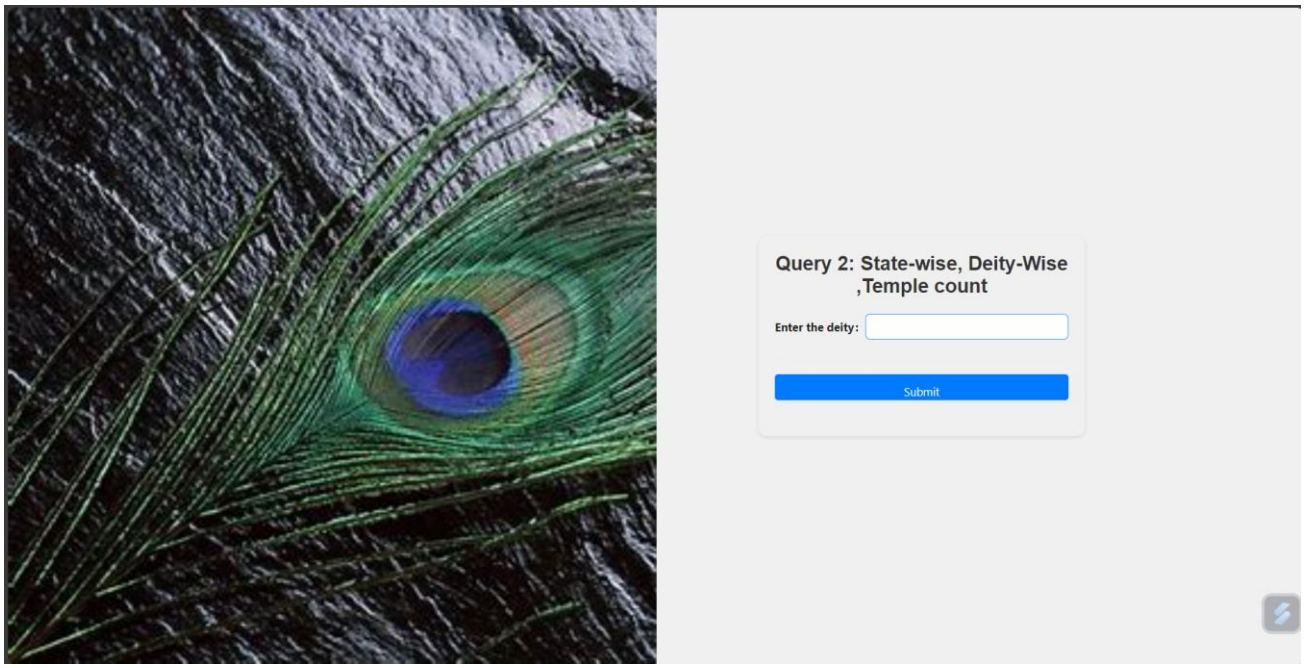


Fig 4.1.8 Query2

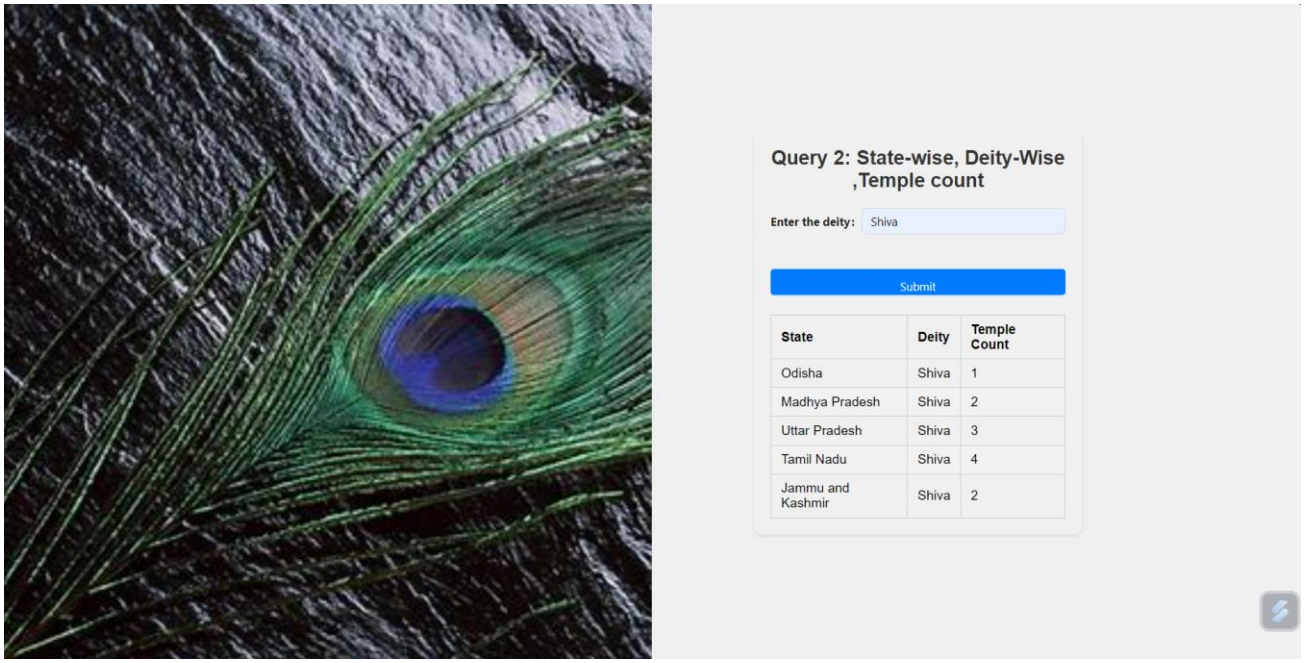


Fig 4.1.9 Query2 Result

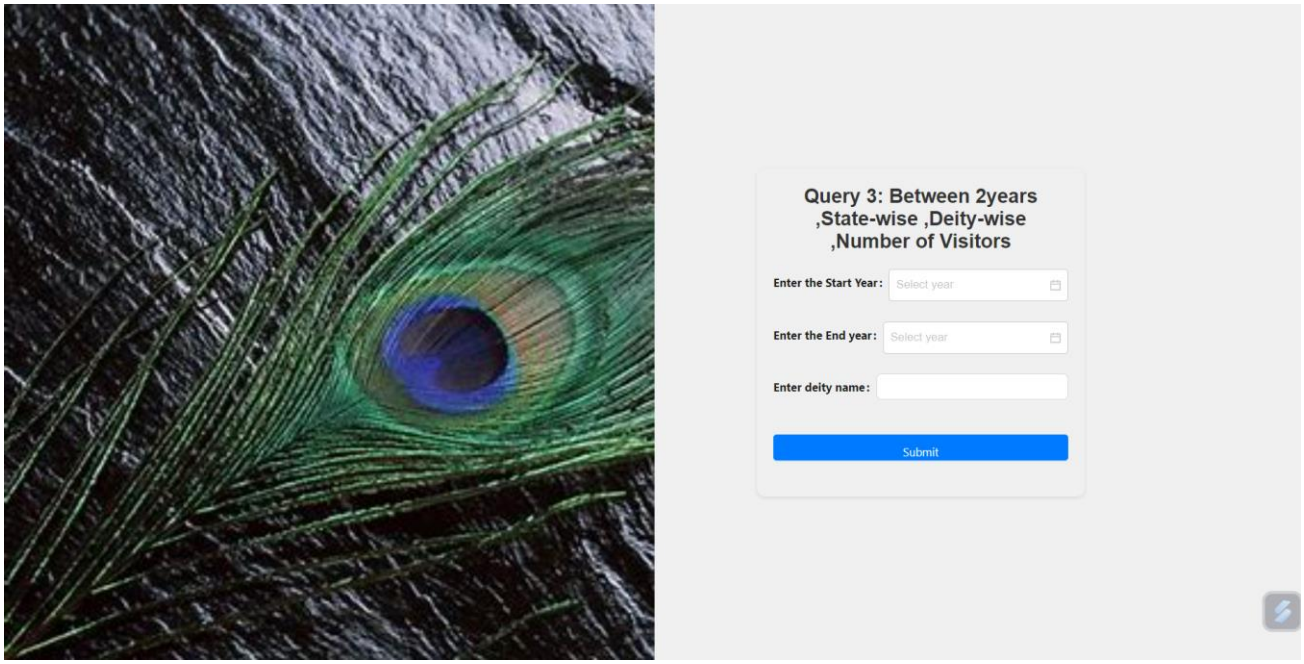


Fig 4.1.10 Query3

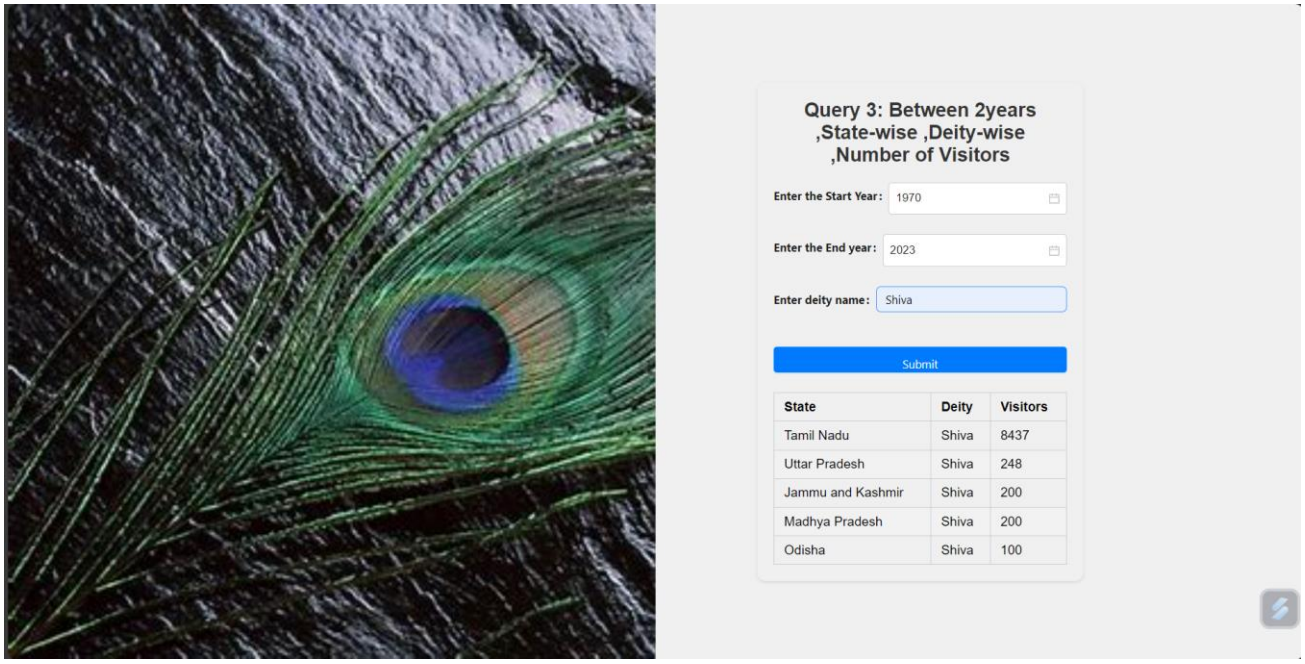


Fig 4.1.11 Query4

4.2 IMPLEMENTATION

//Database Connection

```
const mongoose = require('mongoose')
const colors = require('colors')

const connectDB = async () => {
    try {
        await
mongoose.connect("mongodb://127.0.0.1:2701/TEMPLEANALYTICS")
        console.log('Mongodb connected'.bgGreen.white);
    } catch (error) {
        console.log(error)
    }
}

module.exports = connectDB;
```

//Schema Definitions

```
const userModel = require('../models/userModel');

const registerController = async (req, res) => {
  try {
        const newUser = new userModel({
      d_templename: req.body.d_templename,
      d_deity: req.body.d_deity,
      d_district: req.body.d_district[0],
      d_state: req.body.d_state[0],
      d_pincode: req.body.d_pincode,
      d_era: req.body.d_era[0],
      d_category: req.body.d_category[0],
      d_count: req.body.d_count,
      d_year_of_built: req.body.d_year_of_built,
      d_revenue: req.body.d_revenue,
      d_gold: req.body.d_gold,
      d_opening_time: req.body.d_opening_time,
      d_closing_time: req.body.d_closing_time,
      d_email: req.body.d_email,
      d_description: req.body.d_description,
        });
    await newUser.save();
    res.status(201).send({
      message: 'Registered Successfully',
      success: true,
    });
  } catch (error) {
    console.error('Error in registerController:', error);
    res.status(500).send({
      success: false,
      message: 'Register controller',
```

```
    });
  }
};
module.exports = { registerController };
```

//Queries

#BETWEEN TWO YEARS,STATE-WISE ,DEITYWISE TOTAL REVENUE AND TOTAL GOLD

```
const userModel = require('../models/userModel');
const query1Controller = async (req, res) => {
const { startDate, endDate,d_deity} = req.query;
  const startYear = new Date(startDate).getFullYear();
  const endYear = new Date(endDate).getFullYear();
    try {
    const result = await userModel.aggregate([
      {
        $match: {
          $expr: {
            $and: [
              { $gte: [{ $year: { $toDate: '$d_year_of_built' } }, startYear] },
              { $lte: [{ $year: { $toDate: '$d_year_of_built' } }, endYear] },
            ],
          },
        },
      },
      {
        $match: {
          d_deity: d_deity,
        },
      },
      {
        $group: {
          _id: {
            state: '$d_state',
            deity: '$d_deity'
          },
          count: { $sum: { $toDouble: '$d_revenue' } },
          count1: { $sum: { $toDouble: '$d_gold' } },
        },
      },
      {
        $sort: { count: -1 },
      },
      {

          $project: {
```

```
            startYear: startYear,
            endYear: endYear,
            state: '$_id.state',
            deity: '$_id.deity',
            count: 1,
            count1: 1,
            _id: 0
        }
      }
    ]);
    res.json(result);
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: 'An error occurred' });
  }
};
module.exports = { query1Controller };
```

#STATE-WISE,DEITY-WISE TEMPLE COUNT TEMPLE COUNT

```
const userModel = require('../models/userModel');
const query2Controller = async (req, res) => {
  const { d_state, d_deity } = req.query;
  try {
    let pipeline = [
      {
        $group: {
          _id: {
            state: '$d_state',
            deity: '$d_deity',
          },
          totaltemples: { $sum: 1 },
        },
      },
      {
        $project: {
          state: '$_id.state',
          deity: '$_id.deity',
          totaltemples: 1,
          _id: 0,
        },
      },
    ];
     if (d_deity) {
      pipeline = [
        {
          $match: {
            'd_deity': d_deity,
          },
        },
        ...pipeline,
      ];
    }
```

```
    const result = await userModel.aggregate(pipeline);


    res.status(200).json(result);
  } catch (error) {
    console.error(error);
    res.status(500).json({ success: false, message: 'Error fetching data' });
  }
};

module.exports = { query2Controller };
```

#BETWEEN TWO YEARS ,STATE-WISE ,DEITY-WISE NUMBER OF VISITORS

```
const userModel = require('../models/userModel');
const query3Controller = async (req, res) => {
const { startDate, endDate,d_deity} = req.query;
  const startYear = new Date(startDate).getFullYear();
  const endYear = new Date(endDate).getFullYear();
  try {
    const result = await userModel.aggregate([
      {
        $match: {
          $expr: {
            $and: [
              { $gte: [{ $year: { $toDate: '$d_year_of_built' } }, startYear] },
              { $lte: [{ $year: { $toDate: '$d_year_of_built' } }, endYear] },
            ],
          },
        },
      },
      {
        $match: {
          d_deity: d_deity,
        },
      },
      {
        $group: {
          _id: {
            state: '$d_state',
            deity: '$d_deity'
          },
          count: { $sum: { $toDouble: '$d_count' } },
        },
      },
      {
        $sort: { count: -1 },

      },
```

```
      {

          $project: {
            startYear: startYear,
            endYear: endYear,
            state: '$_id.state',
            deity: '$_id.deity',
            count: 1,
            _id: 0
        }
      }
    ]);

    res.json(result);
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: 'An error occurred' });
  }
};
module.exports = { query3Controller };
```

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

the Temple Management Project has made a significant contribution to the modernization and enhancement of temple management in India. The project's success is a testament to the power of digital solutions to address the complex and evolving needs of religious institutions. The Temple Management Project is a digital solution that streamlines temple administration in India. It enhances transparency, accountability, and security, while increasing efficiency and productivity. The project has made a significant contribution to the modernization of temple management in India. The system's robust security measures also protect sensitive temple data from unauthorized access or breaches.

CHAPTER 6

REFERENCES

- https://stackoverflow.com/
- https://www.youtube.com/watch?v=7CqJlxBYj-M
- https://www.mongodb.com/languages/mern-stack-tutorial
- https://www.geeksforgeeks.org/how-to-connect-mongodb-with-reactjs/