

A Project Report

on

EVENT IT

*Submitted in partial fulfilment of the  
requirement for the award of the degree of*

Bachelor of Technology in Computer Science and  
Engineering



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of

**Ms. Kiran Singh:**

**Assistant Professor**

**Department of Computer Science and Engineering**

**Submitted By**

**Shivam Kumar**

**18SCSE1010744**

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING  
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GALGOTIAS UNIVERSITY, GREATER NOIDA**

**INDIA**

**OCTOBER, 2021**

## CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis / project / dissertation, entitled “**Event it**” in partial fulfilment of the requirements for the award of the B. Tech (CSE) submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of July 2021 to December 2021, under the supervision of **Ms. Kiran Singh**, Assistant Professor, Department of Computer Science and Engineering of School of Computing Science and Engineering, Galgotias University, Greater Noida.

The matter presented in the thesis / project / dissertation has not been submitted by me for the award of any other degree of this or any other places.

Shivam Kumar

18SCSE1010744

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Ms. Kiran Singh

Assistant Professor

## ACKNOWLEDGEMENT

It gives me immense pleasure in bringing out this synopsis of the project entitled “**Event it**”. Firstly, I would like to thank my teacher and guide **Ms. Kiran Singh**, Assistant Professor, Galgotias University who gave me his valuable suggestions and ideas when we were in need of them. She encouraged me to work on this project. I’m also grateful to Galgotias University for giving me the opportunity to work with him and providing me the necessary resources for the project.

I am immensely grateful to all involved in this project as without their inspiration and valuable suggestion it would not have been possible to develop the project within the prescribed time.

With sincere thanks,

Shivam Kumar

## **CERTIFICATE**

The Final Thesis/Project/ Dissertation Viva-Voce examination of Shivam Kumar 18SCSE1010253 has been held on \_\_\_\_\_ and his/her work is recommended for the award of Bachelor of Technology in Computer Science and Engineering.

**Signature of Examiner(s)**

**Signature of Supervisor(s)**

**Signature of Project Coordinator**

**Signature of Dean**

Date:

Place: Greater Noida

## **Abstract**

A calendar is a system of organizing days. This is done by giving names to periods of time, typically days, weeks, months, and years. A date is the designation of a single, specific day within such a system. Some dates have a special meaning attached to them due to the event that occurs on that date, for e.g., 25<sup>th</sup> December is Christmas Day. However, there are times when you want to create your own specific events that have some meaning to you. This website provides a platform for creating those events and broadcasting them to the users related to that organization. Reminders can be enabled for events, with options available for type and time. Event locations can also be added and users can view more information about the event.

My motivation for creating this website is to provide a hassle-free platform to my university and student clubs where they can add events like exams or holidays or cultural events and students can see information related to these events. I have used React.js, HTML, CSS, Node.js, MongoDB, Express, JavaScript and VS code to create this project. All the code for this project is open-source and it available on GitHub.

## List of Figures

<b>Figure No.</b>	<b>Title</b>	<b>Page No.</b>
1	MVC Architecture	5
2	Backend Architecture	6
3	Frontend Architecture	7
4	MERN stack	9
5	Home Page 1	25
6	Home Page 2	25
7	Day Page	26
8	Event Description Page	27
9	Login Page	28
10	After Logging in	28
11	Sign-up Page	29
12	Admin Navbar	30
13	Create Page	31
14	MongoDB Collections	33
15	Events Collections	34

---

16	Users Collections	35
17	get all events	36
18	get single event	36
19	get past events	37
20	get future events	37
21	Signup Route	38
22	Login Route	38
23	Running Server	
24	Open App in Browser	

---

# Table of Contents

CANDIDATE'S DECLARATION .....	ii
ACKNOWLEDGEMENT .....	iii
CERTIFICATE .....	iv
Abstract.....	v
List of Figures .....	vi
CHAPTER 1 .....	1
1.1 Introduction .....	1
1.2 Aim .....	2
1.3 Problem Formulation .....	2
1.4 Literature Survey.....	2
1.4.1 Comparative Study.....	2
1.4.2 Feasibility Analysis .....	3
CHAPTER 2 .....	4
2.1 MVC Architecture.....	4
2.2 Backend Architecture.....	6
2.3 Frontend Architecture .....	7
CHAPTER 3 .....	8
3.1 Frontend.....	10
3.1.1 HTML .....	10
3.1.2 Tailwind CSS .....	10
3.1.3 React .....	12
3.1.4 JavaScript .....	13
3.2 Backend.....	14
3.2.1 NodeJS.....	14
3.2.2 ExpressJS .....	15
3.3 Database .....	16
3.3.1 MongoDB .....	16
3.3.2 Mongoose .....	17
3.4 Tools .....	18
3.4.1 VS Code .....	18
3.4.2 NPM (Node Package Manager).....	19
3.4.3 GitHub .....	20
3.4.4 Postman .....	22



---

CHAPTER 4 .....	24
4.1 Module Description .....	24
4.1.1 Home Page .....	24
4.1.2 Day .....	26
4.1.3 Event Description .....	27
4.1.4 Login .....	28
4.1.5 Register .....	29
4.1.6 Create Event .....	30
4.2 Database .....	32
4.2.1 MongoDB Atlas and Compass: .....	32
4.2.2 Collections: .....	33
4.3 API Testing .....	36
4.4 Instructions .....	39

## CHAPTER 1

### INTRODUCTION

#### 1.1 Introduction

The primary practical use of a calendar is to identify days: to be informed about or to agree on a future event and to record an event that has happened. Days may be significant for agricultural, civil, religious, or social reasons. For example, a calendar provides a way to determine when to start planting or harvesting, which days are religious or civil holidays, which days mark the beginning and end of business accounting periods, and which days have legal significance, such as the day taxes are due or a contract expires. Also, a calendar may, by identifying a day, provide other useful information about the day such as its season. Calendars are also used to help people manage their personal schedules, time, and activities, particularly when individuals have numerous work, school, and family commitments. People frequently use multiple systems and may keep both a business and family calendar to help prevent them from overcommitting their time. A calendar is a system of organizing days. This is done by giving names to periods of time, typically days, weeks, months, and years. A date is the designation of a single, specific day within such a system. Some dates have a special meaning attached to them due to the event that occurs on that date, for e.g., 25th December is Christmas Day. However, there are times when you want to create your own specific events that have some meaning to you. This website provides a platform for creating those events and broadcasting them to the users related to that organization. Reminders can be enabled for events, with options available for type and time. Event locations can also be added and users can view more information about the event.

## **1.2 Aim**

To provide a hassle-free platform to my university and student clubs where they can add events like exams or holidays or cultural events. And to give students a platform where they can see information related to these events.

## **1.3 Problem Formulation**

The problem that our web-app counters is a problem that probably many Universities and students are facing, which is lack of a place which contains all of the academic, social, cultural and all other events that are happening or will happen in future. There are many times students are left unaware of these events and they miss out on opportunities. This also yields bad outcome for the event as it is not able to get enough participants. This should not happen to anyone and that is the main goal of my project, to provide a platform to record all the events and broadcast it to the students for which these events are meant for.

## **1.4 Literature Survey**

### **1.4.1 Comparative Study**

On conducting a comparative study for the project, it was found that there are a lot of universities which have this event calendar thing built in their website. But only the famous and big Universities are doing this. There are a lot of small Universities which don't have anything remotely similar to my project.

Comparative research, simply put, is the act of comparing two or more things with a view to discovering something about one or all of the things being compared. This technique often utilizes multiple disciplines in one study. When it comes to method, the majority agreement is that there is no methodology peculiar to comparative research. The multidisciplinary approach is good for the flexibility it offers, yet comparative programs do have a case to answer against the call that their research lacks a "seamless whole."

#### **1.4.2 Feasibility Analysis**

On running feasibility analysis, the project was found to be technically feasible for the users as this application is a web-based application which means its platform independent and can be easily accessed using any operating system as windows, MacOS, Linux, Chrome OS and this application is also built with responsive design so it would be easily accessible using operating system such Android, IOS using just a web browser such Google Chrome, Mozilla Firefox, Safari, Opera etc. So, it would be fairly easy for the users to access the this without the limitation of any technology or device.

## **CHAPTER 2**

### **PRODUCT DESIGN**

#### **2.1 MVC Architecture**

The Design used here for the proposed system is MVC architecture, which separates the application into three main logical components: The Model, The View and The Controller.

1. Model- The Component corresponds to all the data related logic. In our system it manages the user information to access the application.
2. View- The View component is used for all the UI logic of the application. For example, the Student view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with.
3. Controller- Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data.

# MVC Architecture Pattern

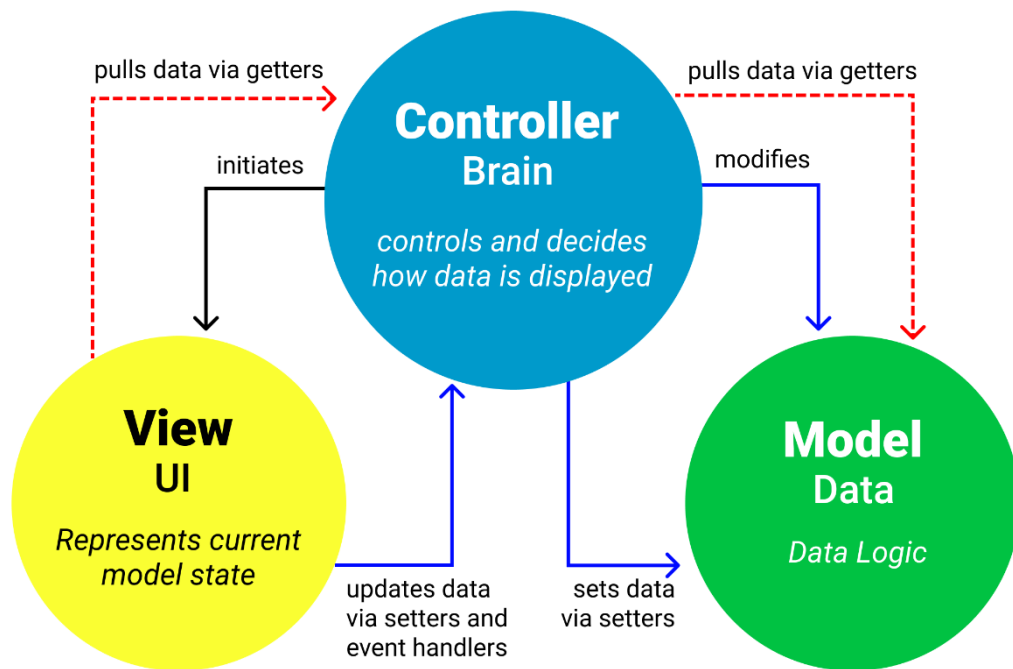


Figure 1: MVC Architecture

## 2.2 Backend Architecture

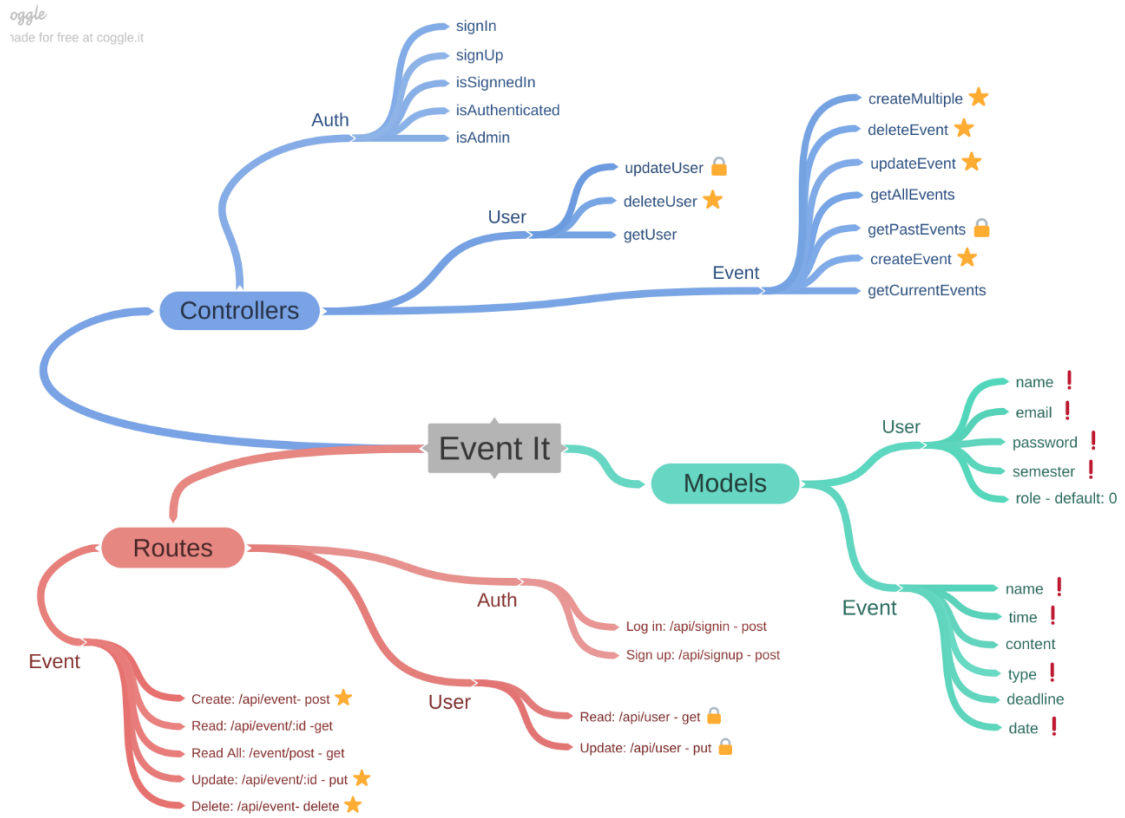


Figure 2: Backend Architecture of Project

## 2.3 Frontend Architecture

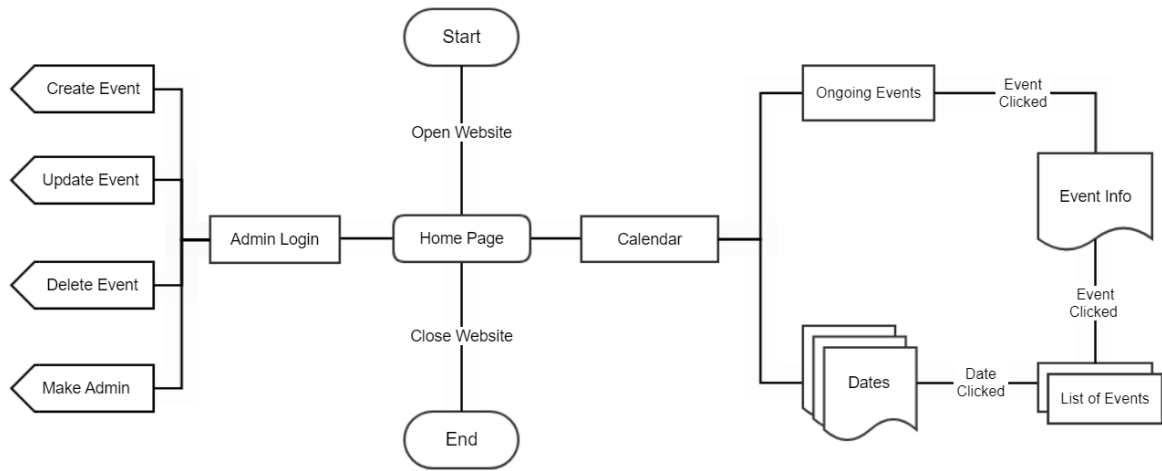


Figure 3: Frontend Architecture of Project



## CHAPTER 3

### TOOLS AND TECHNOLOGIES USED

The key part about an website development is the trunk end functionality. You may receive hundreds of tens of thousands of requests that ping every minute from all around the globe. The backend must support the data retrieving from the server and show it on the front end. In addition to that, there are undoubtedly a lot of additional factors such as for instance logistics, payment gateways, supplier management, and more. These complex features take a toll on the total performance of the site. So, to convey out such complicated activities and to stay powerful round the clock, the trunk end should be durable and expandable; otherwise, the front end also collapses.

Following are the technologies utilized for the creation of this project:

**MERN Stack:** MERN stack is a web development framework. It consists of MongoDB, ExpressJS, ReactJS, and NodeJS as its working components. Here are the details of what each of these components is used for in developing a web application when using MERN stack:

*MongoDB:* A document-oriented, No-SQL database used to store the application data.

*NodeJS:* The JavaScript runtime environment. It is used to run JavaScript on a machine rather than in a browser.

*ExpressJS:* A framework layered on top of NodeJS, used to build the backend of a site using NodeJS functions and structures. Since NodeJS

was not developed to make websites but rather run JavaScript on a machine, ExpressJS was developed.

*ReactJS*: A library created by Facebook. It is used to build UI components that create the user interface of the single page web application.

The user interacts with the ReactJS UI components at the application front-end residing in the browser. This frontend is served by the application backend residing in a server, through ExpressJS running on top of NodeJS.

Any interaction that causes a data change request is sent to the NodeJS based Express server, which grabs data from the MongoDB database if required, and returns the data to the frontend of the application, which is then presented to the user.



Figure 4: MERN stack

## 3.1 Frontend

### 3.1.1 HTML

HTML is a markup language that specifies the structure of your content. HTML consists of a sequence of components, which you employ to encapsulate, or wrap, various portions of the material to make it seem a given way, or perform a certain way. The surrounding tags may make a word or picture hyperlink to someplace else, can italicize words, can make the font larger or smaller, and so on.

HTML (HyperText Markup Language) is the most fundamental building component of the Web. It determines the meaning and organization of online content. "Hypertext" refers to linkages that connect online pages to one another, either inside a single website or across websites. Links are a basic feature of the Web.

### 3.1.2 Tailwind CSS

Tailwind is a utility-first CSS framework. In contrast to other CSS frameworks like

Bootstrap or Materialize CSS it doesn't come with predefined components. Instead, Tailwind CSS operates on a lower level and provides you with a set of CSS helper classes. By using these classes, you can rapidly create custom design with ease. Tailwind CSS is not opinionated and lets you create your own unique design.

Tailwind CSS works by scanning all of your HTML files, JavaScript components, and any other templates for class names, generating the corresponding styles and then writing them to a static CSS file.

It's fast, flexible, and reliable — with zero-runtime.

### *Installation:*

The simplest and fastest way to get up and running with Tailwind CSS from scratch is with the Tailwind CLI tool.

#### 1. Install Tailwind CSS:

Install `tailwindcss`` via npm, and create your `tailwind.config.js`` file.

In the terminal run the following commands:

```
npm install -D tailwindcss
npx tailwindcss init
```

#### 2. Configure your template path:

Add the paths to all of your template files in your `tailwind.config.js`` file

```
module.exports = {
  content: ["/src/**/*.{html,js}"],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

#### 3. Add the Tailwind directives to your CSS:

Add the `@tailwind` directives for each of Tailwind's layers to your main CSS file.

```
@tailwind base;
```

`@tailwind components;`

```
@tailwind utilities;
```

#### 4. Start the Tailwind CLI build process:

Run the CLI tool to scan your template files for classes and build your CSS. In the terminal run the following commands

```
npx tailwindcss -i ./src/input.css -o ./dist/output.css -watch
```

### 3.1.3 React

React is a JavaScript library for building user interfaces. React stands at the intersection of design and programming. It lets you take a complex user interface, and break it down into nestable and reusable pieces called “components” that fit well together.

- React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes. Declarative views make your code more predictable, simpler to understand, and easier to debug.
- Build encapsulated components that manage their own state, then compose them to make complex UIs. Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep state out of the DOM.

#### *Installation:*

Create React App is a comfortable environment for learning React, and is the best way to start building a new single-page application in React.

It sets up your development environment so that you can use the latest JavaScript features, provides a nice developer experience, and optimizes your app for production. You'll need to have Node  $\geq$  14.0.0 and npm  $\geq$  5.6 on your machine.

To create a project, run:

```
npx create-react-app my-app  
cd my-app npm start
```

Then open <http://localhost:3000/> to see your app.

When you're ready to deploy to production, create a minified bundle with

```
npm run build
```

### 3.1.4 JavaScript

JavaScript is a high-level, interpreted scripting language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions. Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it, and major web browsers have a dedicated JavaScript engine to execute it. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has APIs for working with text, arrays, dates, regular expressions, and the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities. It relies upon the host environment in which it is embedded to provide these features.

Initially only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets.

The terms Vanilla JavaScript and Vanilla JS refer to JavaScript not extended by any frameworks or additional libraries. Scripts written in Vanilla JS are plain JavaScript code. Google's Chrome extensions, Opera's extensions, Apple's Safari 5 extensions, Apple's Dashboard Widgets, Microsoft's Gadgets, Yahoo! Widgets, Google Desktop Gadgets, and Serence Klipfolio are implemented using JavaScript.

## **3.2 Backend**

### **3.2.1 NodeJS**

Node.js is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project!

Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant.

A Node.js app runs in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.

When Node.js performs an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back. This allows Node.js to handle thousands of concurrent connections with a single server without introducing the burden of managing thread concurrency, which could be a significant source of bugs.

Node.js has a unique advantage because millions of frontend developers that write JavaScript for the browser are now able to write the server-side code in addition to the client-side code without the need to learn a completely different language.

### **3.2.2 ExpressJS**

Express is the most popular Node web framework, and is the underlying library for a number of other popular Node web frameworks. It provides mechanisms to:

- Write handlers for requests with different HTTP verbs at different URL paths (routes).
- Integrate with "view" rendering engines in order to generate responses by inserting data into templates.
- Set common web application settings like the port to use for connecting, and the location of templates that are used for rendering the response.
- Add additional request processing "middleware" at any point within the request handling pipeline.



While Express itself is fairly minimalist, developers have created compatible middleware packages to address almost any web development problem. There are libraries to work with cookies, sessions, user logins, URL parameters, POST data, security headers, and many more.

### *Installation:*

Use the following command to install express:

```
npm install express --save
```

## **3.3 Database**

### **3.3.1 MongoDB**

MongoDB is a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. Documents consist of keyvalue pairs which are the basic unit of data in MongoDB. Collections contain sets of documents and function which is the equivalent of relational database tables.

MongoDB is a database which came into light around the mid-2000s.

Features of MongoDB:

1. Each database contains collections which in turn contains documents. Each document can be different with a varying number of fields. The size and content of each document can be different from each other.
2. The document structure is more in line with how developers construct their classes and objects in their respective programming languages.

Developers will often say that their classes are not rows and columns but have a clear structure with key-value pairs.

3. The rows (or documents as called in MongoDB) doesn't need to have a schema defined beforehand. Instead, the fields can be created on the fly.
4. The data model available within MongoDB allows you to represent hierarchical relationships, to store arrays, and other more complex structures more easily.
5. Scalability – The MongoDB environments are very scalable. Companies across the world have defined clusters with some of them running 100+ nodes with around millions of documents within the database.

### **3.3.2 Mongoose**

Mongoose is an Object Document Mapper (ODM). This means that Mongoose allows you to define objects with a strongly-typed schema that is mapped to a MongoDB document.

Mongoose provides an incredible amount of functionality around creating and working with schemas. Mongoose currently contains eight Schema Types that a property is saved as when it is persisted to MongoDB. They are:

- String
- Boolean
- Number
- Mixed
- Date
- ObjectId
- Buffer
- Array

## 3.4 Tools

### 3.4.1 VS Code

Visual Studio Code is a source-code editor 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. Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 Build conference. A preview build was released shortly thereafter.

On November 18, 2015, the source of Visual Studio Code was released under the MIT License, and made available on GitHub. Extension support was also announced. On April 14, 2016, Visual Studio Code graduated from the public preview stage and was released to the Web.[12] Microsoft has released most of Visual Studio Code's source code on GitHub under the permissive MIT License, while the releases by Microsoft are proprietary freeware.

In the Stack Overflow 2021 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 70% of 82,000 respondents reporting that they use it.

### 3.4.2 NPM (Node Package Manager)

NPM is the standard package manager for Node.js. In January 2017 over 350000 packages were reported being listed in the npm registry, making it the biggest single language code repository on Earth, and you can be sure there is a package for (almost!) everything.

It started as a way to download and manage dependencies of Node.js packages, but it has since become a tool used also in frontend JavaScript.

The public npm registry is a database of JavaScript packages, each comprised of software and metadata. Open-source developers and developers at companies use the npm registry to contribute packages to the entire community or members of their organizations, and download packages to use in their own projects.

A package is a file or directory that is described by a package.json file. A package must contain a package.json file in order to be published to the npm registry. For more information on creating a package.json file, see "Creating a package.json file". npm manages downloads of dependencies of your project.

Installing all dependencies: `npm install`

it will install everything the project needs, in the `node_modules` folder, creating it if it's not existing already.

Installing a single package: `npm install <package-name>`

### 3.4.3 GitHub

GitHub, Inc. is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management (SCM) functionality of Git, plus its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, continuous integration and wikis for every project. Headquartered in California, it has been a subsidiary of Microsoft since 2018.

It is commonly used to host open-source projects. As of November 2021, GitHub reports having over 73 million developers and more than 200 million repositories (including at least 28 million public repositories). It is the largest source code host as of November 2021.

GitHub is a web-based interface that uses Git, the open source version control software that lets multiple people make separate changes to web pages at the same time. As Carpenter notes, because it allows for real-time collaboration, GitHub encourages teams to work together to build and edit their site content.

Some common terms related to GitHub are:

- **Repository (repo)** — a folder in which all files and their version histories are stored.
- **Branch** — a workspace in which you can make changes that won't affect the live site.

- **Markdown (.md)** — a way to write in Github that converts plain text to GitHub code. Sites such as Atom and Sublime Text are examples of free resources for developers using Markdown.
- **Commit Changes** — a saved record of a change made to a file within the repo.
- **Pull Request (PR)** — the way to ask for changes made to a branch to be merged into another branch that also allows for multiple users to see, discuss and review work being done.
- **Merge** — after a pull request is approved, the commit will be pulled in (or merged) from one branch to another and then, deployed on the live site
- **Issues** — how work is tracked when using git. Issues allow users to report new tasks and content fixes, as well as allows users to track progress on a project board from beginning to end of a specific project.

In order to use GitHub, we need to make an empty repository on the GitHub account. Then use the following commands on the command line to successfully upload your code to this repository.

```
git init
git add .
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/<account-name> /<repo>.git
git push -u origin main
```

### 3.4.4 Postman

Postman is an application used for API testing. It is an HTTP client that tests HTTP requests, utilizing a graphical user interface, through which we obtain different types of responses that need to be subsequently validated. Postman offers many endpoint interaction methods. The following are some of the most used, including their functions:

- GET: Obtain information
- POST: Add information
- PUT: Replace information
- PATCH: Update certain information
- DELETE: Delete information

When testing APIs with Postman, we usually obtain different response codes.

Some of the most common include:

100 Series: Temporal responses, for example, '102 Processing'.

200 Series: Responses where the client accepts the request and the server processes it successfully, for instance, '200 Ok'.

300 Series: Responses related to URL redirection, for example, '301 Moved

Permanently.'

400 Series: Client error responses, for instance, '400 Bad Request'.

500 Series: Server error responses, for example, '500 Internal Server Error.'

Postman gives the possibility to group different requests. This feature is known as

‘collections’ and helps organize tests. These collections are folders where requests are stored and can be structured in whichever way the team prefers. It is also possible to export-import them. Postman also allows us to create different environments through the generation/use of variables; for example, a URL variable that is aimed towards different test environments (dev-QA), enabling us to execute tests in different environments using existing requests.



## **CHAPTER 4**

### **IMPLEMENTATIONS AND RESULTS**

#### **4.1 Module Description**

The project has a large scope and has number of functionalities to it. Thus, project have a multiple number of modules to it. This website is divided up in modules according to the user flow. There are many pages in this website, but it is very easy to navigate and the user can instinctively go between different pages. This website's main focus is its home page, this page contains a calendar and the user can go to different months or different years. In each of the month there are events that are held on that particular date. Then the user can click any date and view the events of that particular day. In the day page we have all the events listed with their title and to get more details on a particular event, user can click and go to the event's details page. Here is all related information that the user has provide while creating the event. Users are allowed to login or register to use some advanced features of the website. If you login as Admin, you are allowed to make changes to an existing event or create a new event.

##### **4.1.1 Home Page**

This is the main page of the website and this will be viewed by any user firstly when coming on this website. This page contains a calendar and the user can go to different months or different years to view the calendar of that particular month. The calendar contains a grid containing all the days of that month. Any events occurring on that date is shown on the day inside the grid and the user can click any date to view all the events occurring on that day.

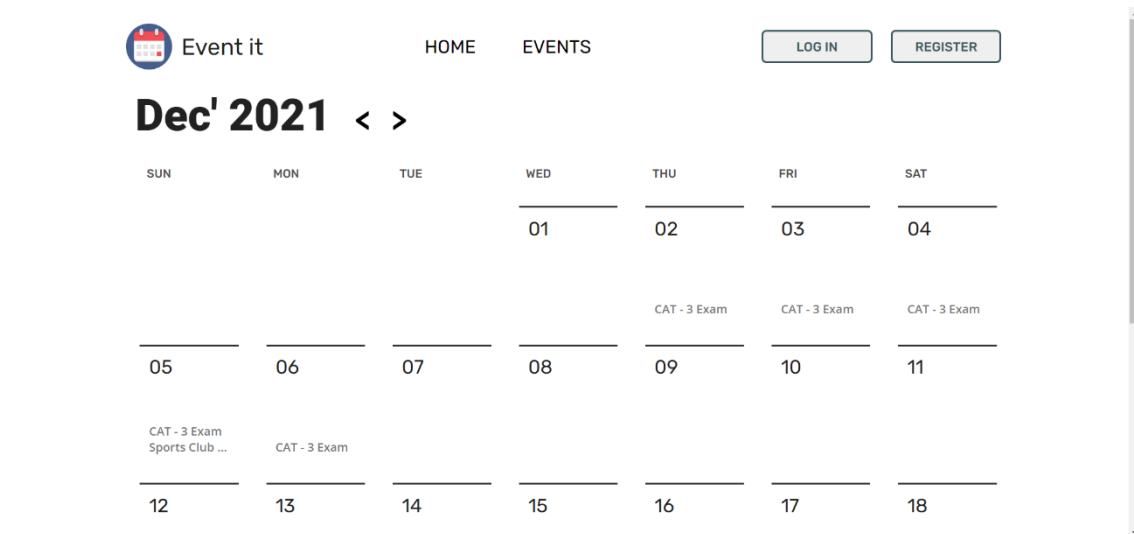


Figure 5: Home Page 1

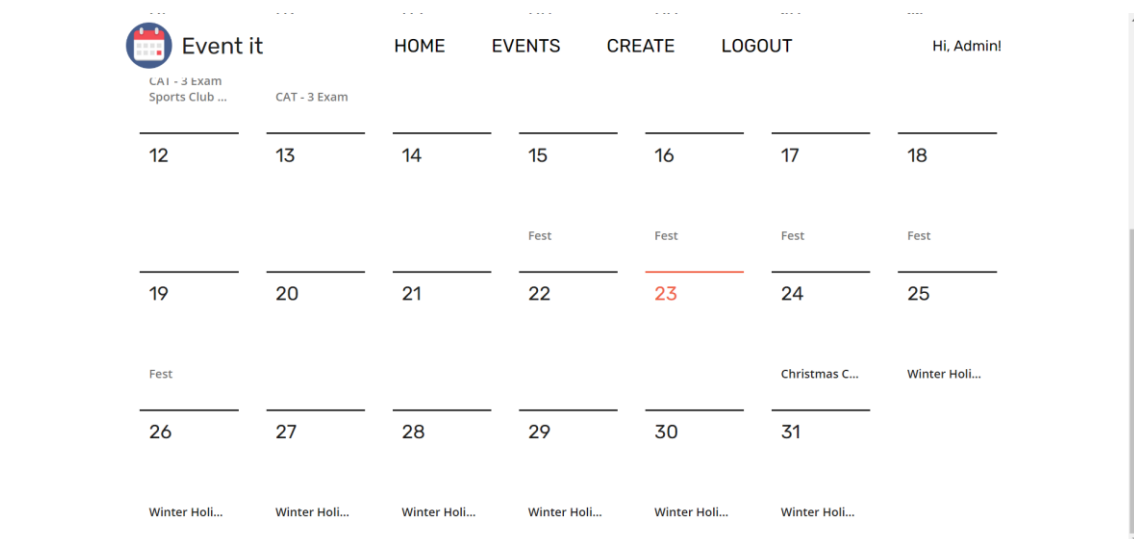
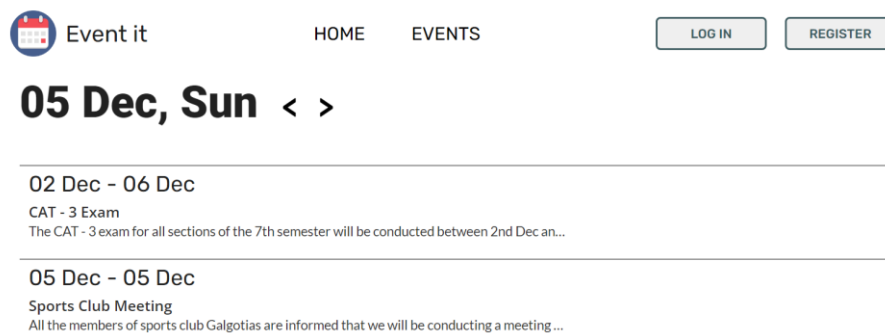


Figure 6: Home Page 2

### 4.1.2 Day

In this page we have all the events listed that occurs on this particular day. The user can come to this page by clicking on any day inside the calendar on the home page of the website. This has all the event's basic information like the start date and the end date, title of the event and a first 100 characters of the description of the project. Users can click any event from the list and navigate to the detailed information of the event. There are also left and right arrow buttons beside the date, users can use these to change date directly form here. There is no need to visit home page again and again.



Event it      HOME    EVENTS         

**05 Dec, Sun** < >

---

02 Dec - 06 Dec  
CAT - 3 Exam  
The CAT - 3 exam for all sections of the 7th semester will be conducted between 2nd Dec an...

---

05 Dec - 05 Dec  
Sports Club Meeting  
All the members of sports club Galgotias are informed that we will be conducting a meeting ...

Figure 7: Day Page

### 4.1.3 Event Description

This is the description page for events, it contains all related information that the user has provide while creating the event. To navigate to this page, users need to click one of the many listed events in the day page. Once the user clicks it, the detailed version i.e., this page pops up. It provides information like Event title, from and till, category of the event, Author of the event, and the details provided by the admin while creating the event.

---

Event it

HOME

EVENTS

LOG IN

REGISTER

---

## Sports Club Meeting

05 Dec 02:30 PM - 05 Dec 02:30 PM

Category: other

Author: Sports Club Galgotias

**Details:**

All the members of sports club Galgotias are informed that we will be conducting a meeting on 5th Dec. Presence of every member is compulsory. As per Covid guidelines, all of you are requested to wear masks and maintain social distancing.

Venue: C-block, Auditorium, Galgotias University

Time: 2:30pm to 4:30pm

For any query feel free to contact the undersigned.

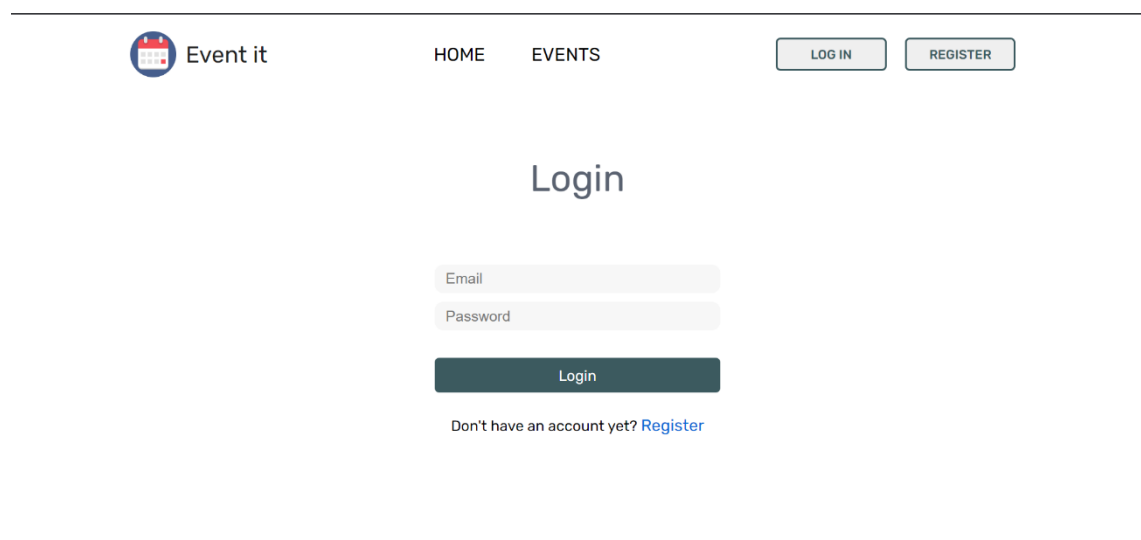
Gaurav Rana  
Sports Head  
+91 8890782907

Figure 8: Event Description Page

#### 4.1.4 Login

This website offers the users with the login functionality. Here the users need to enter the registered email address and password. If the provided email and password exists inside the database, the user is logged in and his name is shown on the top right corner of the website.

---



Event it      HOME      EVENTS      LOG IN      REGISTER

### Login

Email

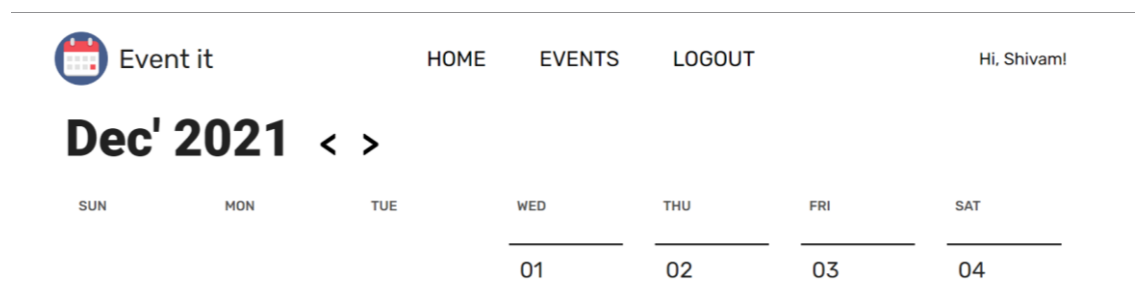
Password

Login

Don't have an account yet? [Register](#)

Figure 9: Login Page

---



Event it      HOME      EVENTS      LOGOUT      Hi, Shivam!

## Dec' 2021 < >

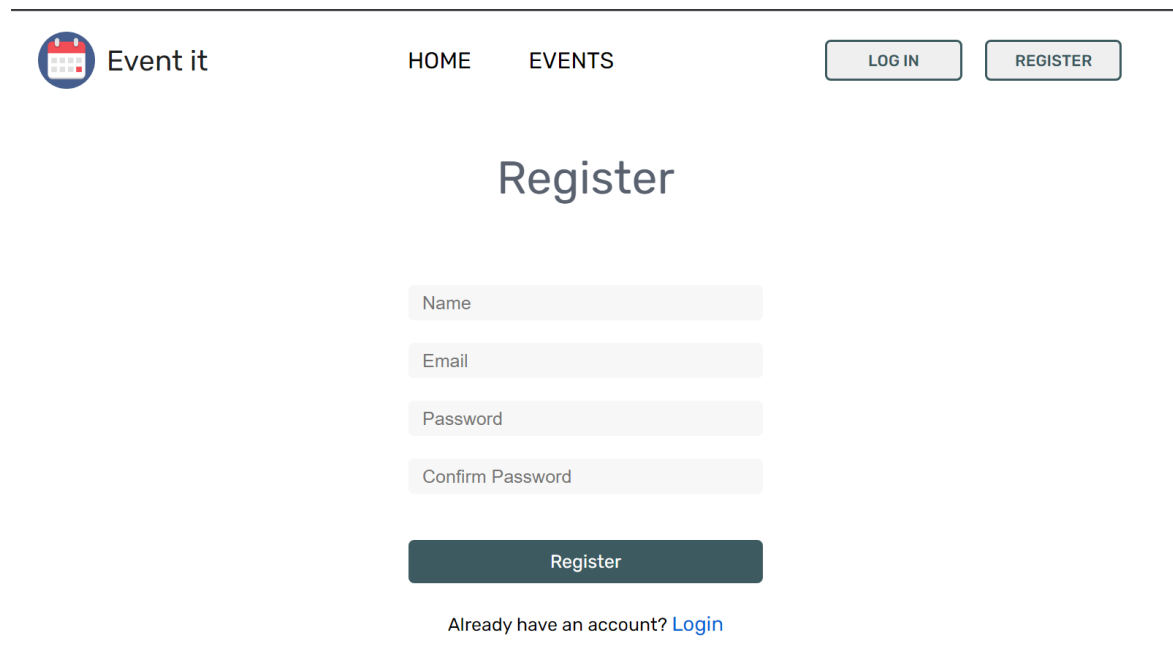
SUN	MON	TUE	WED	THU	FRI	SAT
			01	02	03	04

Figure 10: After Logging in

### 4.1.5 Register

If the user doesn't already have an account, they are redirected to the Sign-Up page where they can register. The user is required to provide username, email, password and then confirm password to register for a user account. This information is saved inside the database so if the user wants to log in again, he can use the provided email and password.

---



The screenshot shows the 'Register' page of the 'Event It' application. At the top left is the 'Event it' logo. The navigation menu includes 'HOME' and 'EVENTS'. On the right, there are 'LOG IN' and 'REGISTER' buttons. The main heading is 'Register'. Below it are four input fields: 'Name', 'Email', 'Password', and 'Confirm Password'. A dark green 'Register' button is positioned below the fields. At the bottom, there is a link: 'Already have an account? [Login](#)'.

Figure 11: Sign-up Page

#### 4.1.6 Create Event

If user wants to create an event, firstly he needs the admin privileges. These can be verified by logging in as admin, using admin's email and password. Once you login as admin, the home page has advanced options enabled. One of them is, 'Create' will pop up in the main navbar of the page.

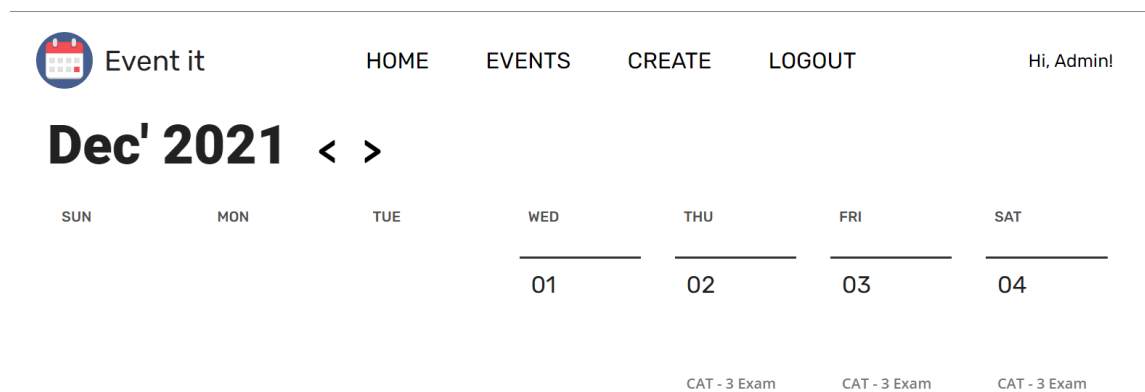


Figure 12: Admin Navbar

These options won't pop up until you login as admin. Here you can click on the create button in the navbar to navigate to the create page. This 'Create page' has a lot of forms where you have to enter all the necessary details. You can not skip any field as in leave it empty, as all the fields are mandatory and must be filled with the correct information.

Once you fill in all the information, there is a publish button on the top right corner of this page. The user can click this Publish button to successfully create an event. This event gets saved to the database and the information can be retrieved whenever needed.

---

Event it

HOME

EVENTS

CREATE

LOGOUT

Hi, Admin!

## Create Events

PUBLISH

Your Title Here

Author Name

Category

Start Date: dd-mm-yyyy --:--

End Date: dd-mm-yyyy --:--

Your Details Here...

Figure 13: Create Page



## 4.2 Database

A “cloud database” can be one of two distinct things: a traditional or NoSQL database installed and running on a cloud virtual machine (be it public cloud, private cloud, or hybrid cloud platforms), or a cloud provider’s fully managed database-as-a-service (DBaaS) offering. The former, running your own self-managed database in a cloud environment, is really no different from operating a traditional database. Cloud DbaaS, on the other hand, is the natural database equivalent of software-as-a-service (SaaS): pay as you go, and only for what you use, and let the system handle all the details of provisioning and scaling to meet demand, while maintaining consistently high performance.

### 4.2.1 MongoDB Atlas and Compass:

#### MongoDB Atlas

The project uses MongoDB Atlas as the cloud database provider. MongoDB Atlas, a fully managed cloud database for modern applications. Atlas is the best way to run MongoDB, the leading modern database. MongoDB’s document model is the fastest way to innovate, bringing flexibility and ease of use to the database.

#### MongoDB Compass

MongoDB Compass is a powerful GUI for querying, aggregating, and analysing your MongoDB data in a visual environment. Compass is free to use and source available, and can be run on macOS, Windows, and Linux.

## 4.2.2 Collections:

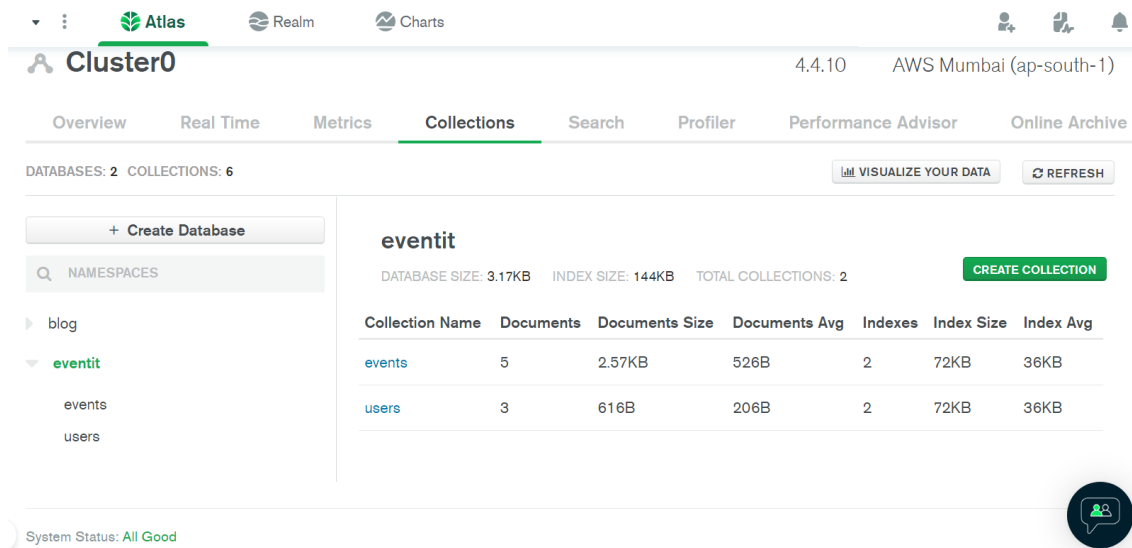
Collections is grouping of MongoDB documents. A collection is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields.

Typically, all documents in a collection have a similar or related purpose.

In the project database there are currently different collections to store different database from the website.

The current used collections are as under:

- Users Collection
- Events Collection



The screenshot displays the MongoDB Atlas interface for a cluster named 'Cluster0'. The 'Collections' tab is active, showing the 'eventit' database. The database contains two collections: 'events' and 'users'. The 'events' collection has 5 documents, a size of 2.57KB, and an average document size of 526B. The 'users' collection has 3 documents, a size of 616B, and an average document size of 206B. The interface also shows a 'CREATE COLLECTION' button and a table with columns for Collection Name, Documents, Documents Size, Documents Avg, Indexes, Index Size, and Index Avg.

Collection Name	Documents	Documents Size	Documents Avg	Indexes	Index Size	Index Avg
events	5	2.57KB	526B	2	72KB	36KB
users	3	616B	206B	2	72KB	36KB

Figure 14: MongoDB Collections

## Events Collection:

Events Collection is the collection of all the events listed on the website with the entry field of data as `_id`, `title`, `details`, `category`, `startTime`, `endTime`, `expired`. This Collection has default value of `expired` set to `false` and all the other fields are required.

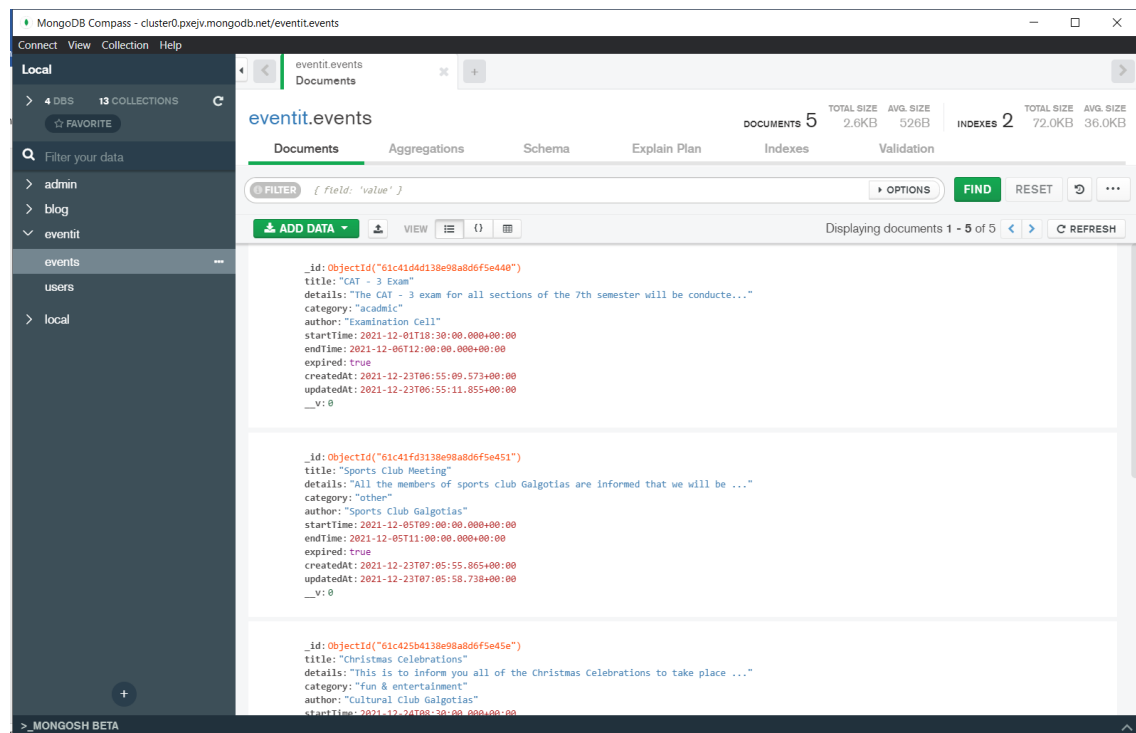


Figure 15: Events Collections

## Users Collection:

User collection is the collection of all the registered user with the field of their credential such as email, name, password, user id, role, Sem etc.

Passwords are first encrypted then stored in the data based in its encoding form. The project uses bcrypt to hash the passwords. To match the password from the user at the time of authentication, the provided password is also hashed and then compared with the stored password. This increases the security of the users data.

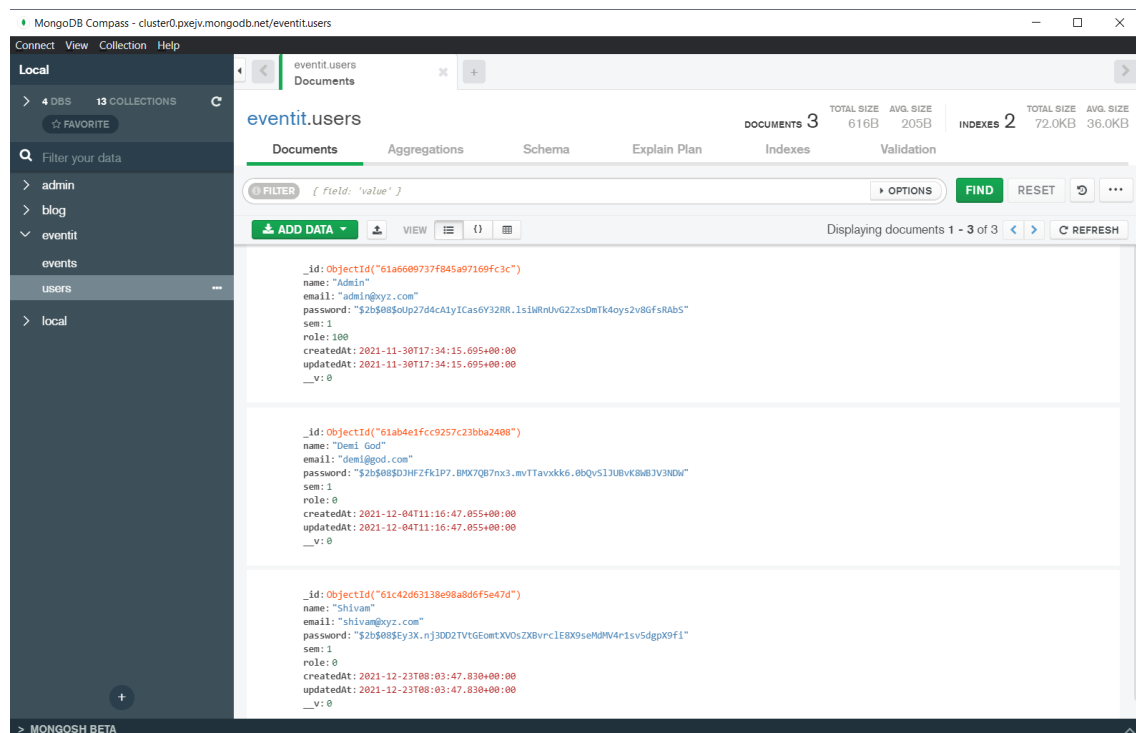


Figure 16: Users Collections

## 4.3 API Testing

### Fetching events from api/events route

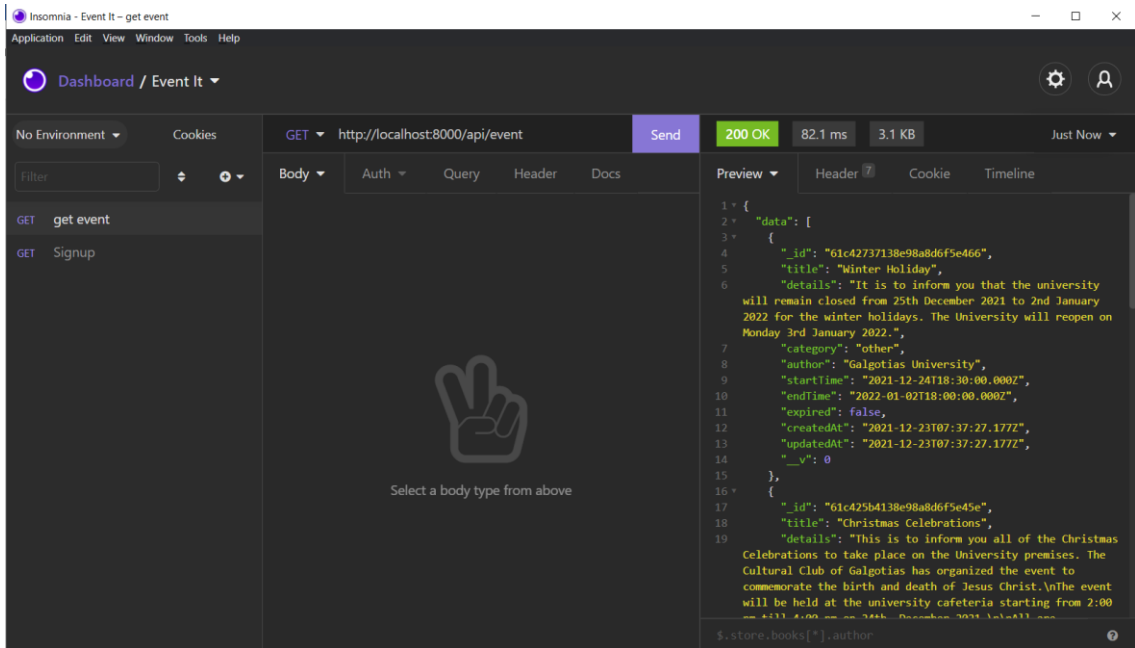


Figure 17: get all events

### Fetching a single event using id

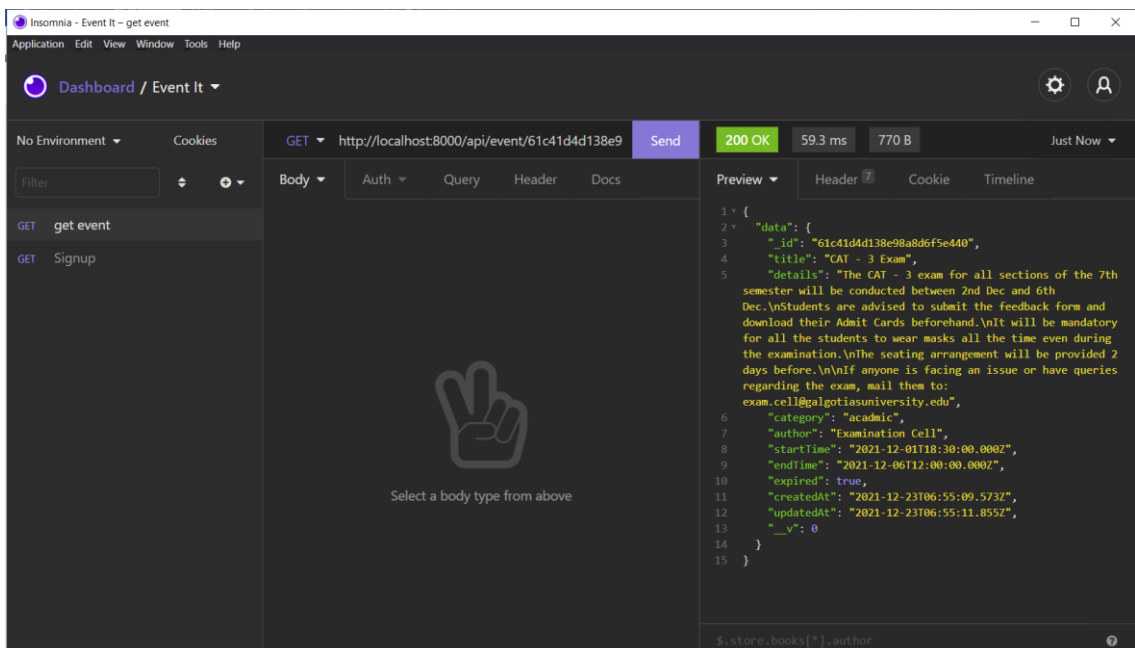


Figure 18: get single event

## Fetching Past events

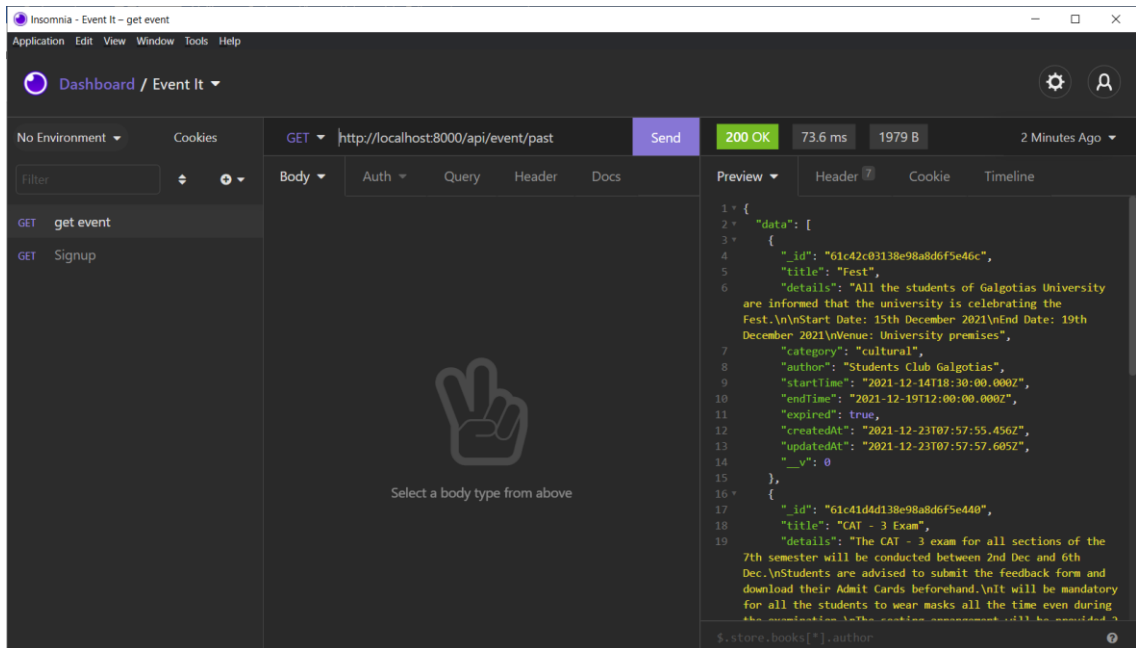


Figure 19: get past events

## Fetching Ongoing events

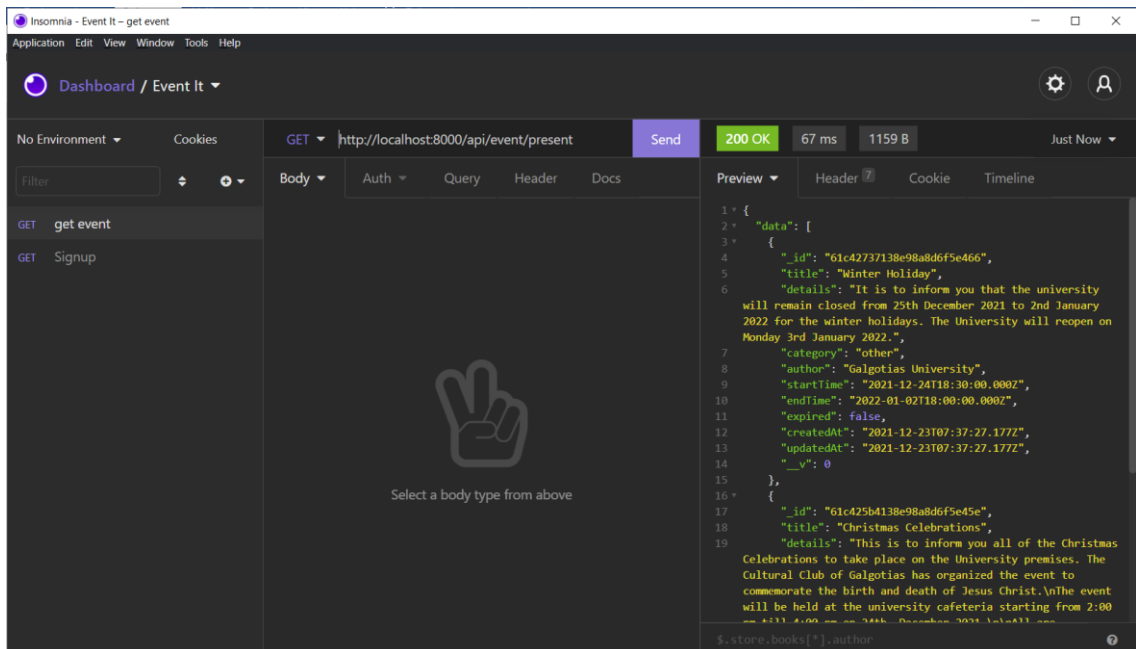


Figure 20: get future events

## Sign-up Route

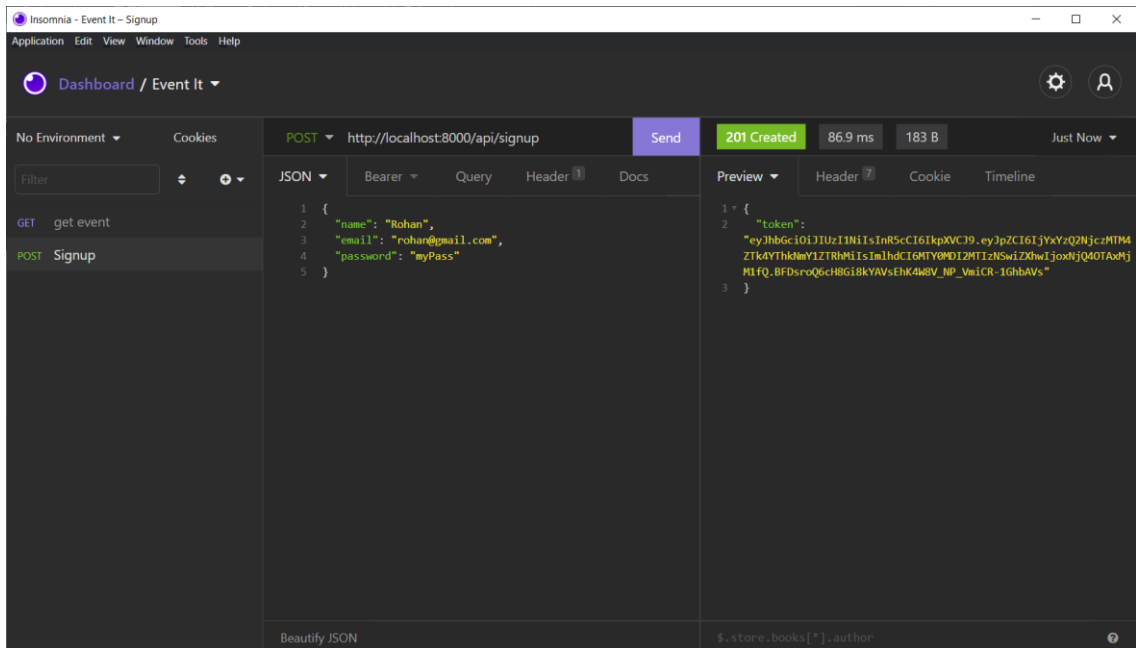


Figure 21: signup route

## Login Route

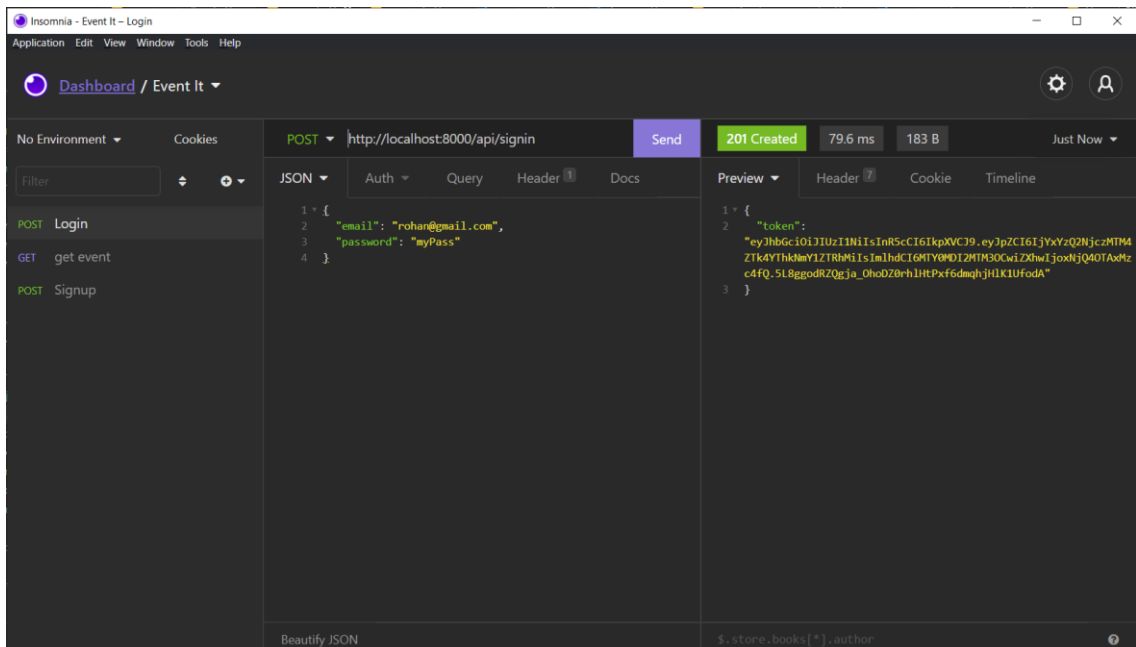


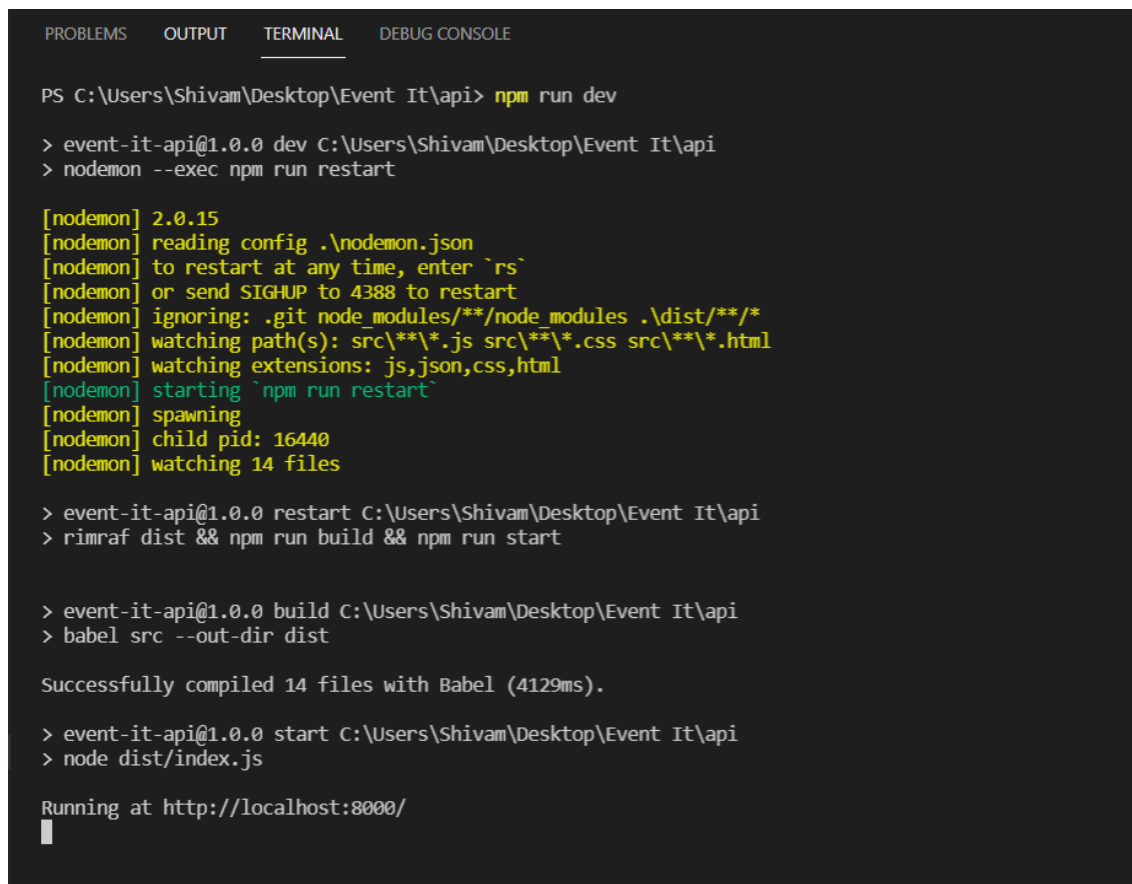
Figure 22: login route

## 4.4 Instructions

Following are the instructions to run the project

In the terminal once all packages are installed run the below command to run the development server.

```
npm run dev
```



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\Shivam\Desktop\Event It\api> npm run dev
> event-it-api@1.0.0 dev C:\Users\Shivam\Desktop\Event It\api
> nodemon --exec npm run restart

[nodemon] 2.0.15
[nodemon] reading config .\nodemon.json
[nodemon] to restart at any time, enter `rs`
[nodemon] or send SIGHUP to 4388 to restart
[nodemon] ignoring: .git node_modules/**/node_modules .\dist/**/*
[nodemon] watching path(s): src\**\*.js src\**\*.css src\**\*.html
[nodemon] watching extensions: js,json,css,html
[nodemon] starting `npm run restart`
[nodemon] spawning
[nodemon] child pid: 16440
[nodemon] watching 14 files

> event-it-api@1.0.0 restart C:\Users\Shivam\Desktop\Event It\api
> rimraf dist && npm run build && npm run start

> event-it-api@1.0.0 build C:\Users\Shivam\Desktop\Event It\api
> babel src --out-dir dist

Successfully compiled 14 files with Babel (4129ms).

> event-it-api@1.0.0 start C:\Users\Shivam\Desktop\Event It\api
> node dist/index.js

Running at http://localhost:8000/
█
```

Figure 23: Running Server

Once the server is up and running and database is connected. You need to launch the frontend of the project using the following command.

```
npm start
```



```
[1] Compiled successfully!  
[1]  
[1] You can now view frontend in the browser.  
[1]  
[1] Local: http://localhost:3000  
[1] On Your Network: http://192.168.29.212:3000  
[1]  
[1] Note that the development build is not optimized.  
[1] To create a production build, use npm run build.  
...
```

Figure 24: Open App in Browser

Then open <http://localhost:3000/> to see the app.