
Test Document

for

What's Next



Version 1.1

Group #:

Aditya Kumar	210060
Akkinepally Kruthi	210088
Aman Arya	210106
Apoorva Gupta	210179
Chitwan Goel	210295
Krish Sharma	210392
	210530
Paras Sikarwar	210699
Talin Gupta	211095
Varun Tokas	211152
Siddharth Kalra	211032

Group Name:Codecrafters

adityakum21@iitk.ac.in
akruthi21@iitk.ac.in
aarya21@iitk.ac.in
apoorvag21@iitk.ac.in
chitwang21@iitk.ac.in
geetika21@iitk.ac.in
krish21@iitk.ac.in
sparas21@iitk.ac.in
taling21@iitk.ac.in
varuntokas21@iitk.ac.in
siddharthk21@iitk.ac.in

Course: CS253

Mentor TA: Archi Gupta

Date: 30/3/2023

Contents



CONTENTS	ii
REVISIONS	ii
1 INTRODUCTION	4
2 UNIT TESTING	5-22
3 INTEGRATION TESTING	23-49
4 SYSTEM TESTING	50-54
5 CONCLUSION	55-56
APPENDIX A - GROUP LOG	57

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.1	Varun Tokas Paras Sikarwar Kruthi Geetika Apoorva Gupta Aditya Kumar Chitwan Goel Talin Gupta Krish Sharma Siddharth Kalra Aman Arya	First draft	01/04/2023

1 Introduction

Test Strategy:

We are using Manual testing to test our software.

Testing period:

We have done unit testing during the making of the website, after the completion of every unit.

System testing of all the units was done once again after all the units and their changes were completed.

Testers:

Developers were itself the testers

Coverage criteria:

We mainly used decision coverage(which is a type of functional coverage) to test this software.

Have you used any tool for testing?

Tools:

We used Thunder Client to test our software. Thunder Client is a popular HTTP client extension for Visual Studio Code that allows developers to test APIs and web services directly from their code editor. It's advantages include:

- **Ease of use:** Thunder Client has a user-friendly interface that makes it easy for developers to quickly set up and send requests without having to deal with complex configurations.
- **Integration with Visual Studio Code:** Thunder Client integrates seamlessly with Visual Studio Code, allowing developers to switch between coding and testing without leaving the code editor.
- **Advanced features:** Thunder Client offers several advanced features such as automatic cookie management, request history, and code snippets that help developers save time and increase productivity.
- **Free and open source:** Thunder Client is free and open source, which means that developers can use it without any licensing fees and contribute to its development.

2 Unit Testing

1. Sign-up-

Check the functioning of signup functionality using ThunderClient.

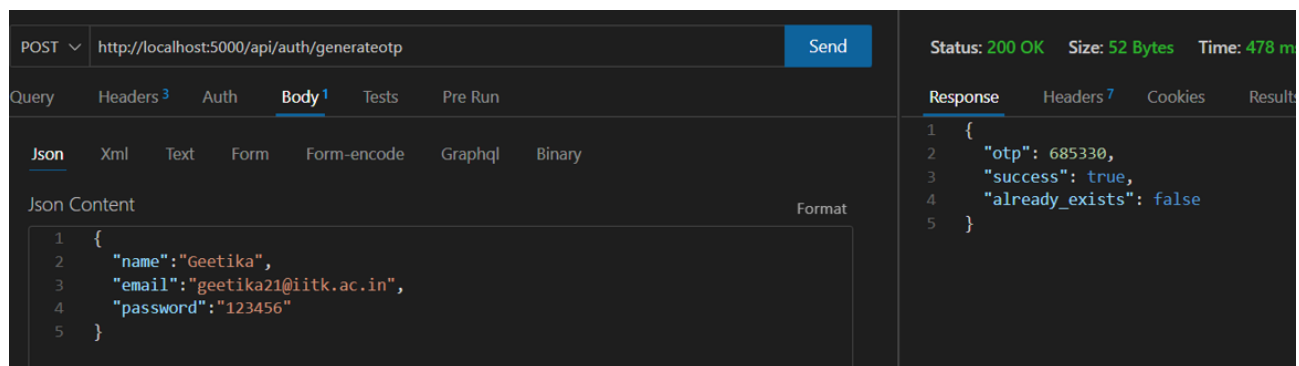
Unit Details: Signup (routes/auth.js) generateotp, createuser

Test Owner: Geetika and Varun Tokas

Test Date: 16-03-2023

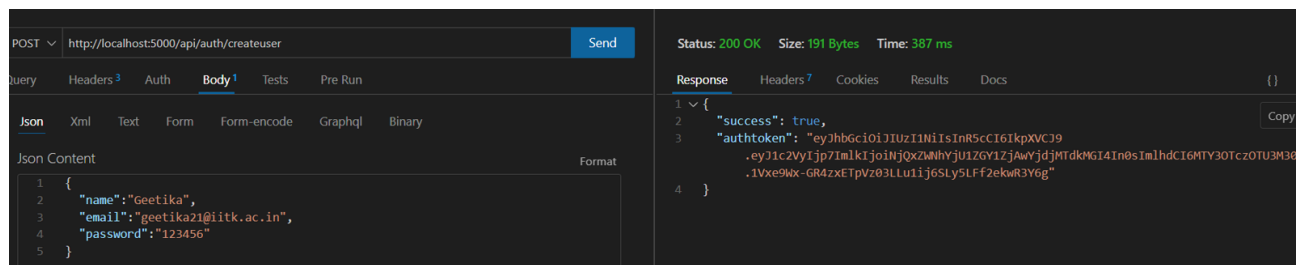
Test Results:

1)generateotp: Generating otp if the user with given email-id doesn't



The screenshot shows a ThunderClient interface for a POST request to `http://localhost:5000/api/auth/generateotp`. The status is 200 OK, with a size of 52 Bytes and a time of 478 ms. The request body is a JSON object: `{ "name": "Geetika", "email": "geetika21@iitk.ac.in", "password": "123456" }`. The response is a JSON object: `{ "otp": 685330, "success": true, "already_exists": false }`.

2)createuser: gives an authtoken



The screenshot shows a ThunderClient interface for a POST request to `http://localhost:5000/api/auth/createuser`. The status is 200 OK, with a size of 191 Bytes and a time of 387 ms. The request body is a JSON object: `{ "name": "Geetika", "email": "geetika21@iitk.ac.in", "password": "123456" }`. The response is a JSON object: `{ "success": true, "authtoken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2Vybm9udjoiOiJjQzZlbnhyZjU1ZGY1ZjAwvjdjMTdkRkGI4In81mlhdCI6MTY3OTczOTU3M3B6.1Vxe9Nx-GR4zETpVz03LLu1j6SLy5LF2ekwR3Y6g" }`.

2. Login-

Unit Details: login(routes/auth.js) login, getuser

Test Owner: Geetika and Varun Tokas

Test Date: 16-03-2023

Test Results:

1)Login: Works fine when the id and password is correct(generates auth-token), otherwise returns error.

The screenshot shows a REST client interface for a POST request to `http://localhost:5000/api/auth/login`. The request body is a JSON object with email `geetika21@iitk.ac.in` and password `123456`. The response is a 200 OK with a success status and a long auth token.

```

POST http://localhost:5000/api/auth/login
{
  "email": "geetika21@iitk.ac.in",
  "password": "123456"
}

Response: 200 OK, Size: 191 Bytes, Time: 481 ms
{
  "success": true,
  "authToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjp7ImkiOiJoinjQxZWlhYjU1ZGY1ZjAwYjdmJmtdkMGI4In0sIm1hdCI6MTY3MTUzOTc2OTkxN30.XwP54UNBKB8I3Ri5SgtZPLccZV5EHa69BzekwhTM"
}

```

2)Wrong password:

The screenshot shows a REST client interface for a POST request to `http://localhost:5000/api/auth/login`. The request body is a JSON object with email `geetika21@iitk.ac.in` and password `123476`. The response is a 400 Bad Request with an error message.

```

POST http://localhost:5000/api/auth/login
{
  "email": "geetika21@iitk.ac.in",
  "password": "123476"
}

Response: 400 Bad Request, Size: 72 Bytes, Time: 1.73 s
{
  "success": false,
  "error": "Please try to login with correct credentials"
}

```

3)getuser: takes auth token and gives details of the user

The screenshot shows a REST client interface for a POST request to `http://localhost:5000/api/auth/getuser`. The request includes an auth token in the headers. The response is a 200 OK with user details.

```

POST http://localhost:5000/api/auth/getuser
Headers:
  Accept: */*
  User-Agent: Thunder Client (https://www.thunderclient.com)
  auth-token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjp7ImkiOiJoinjQxZWlhYjU1ZGY1ZjAwYjdmJmtdkMGI4In0sIm1hdCI6MTY3MTUzOTc2OTkxN30.XwP54UNBKB8I3Ri5SgtZPLccZV5EHa69BzekwhTM

Response: 200 OK, Size: 156 Bytes, Time: 328 ms
{
  "_id": "641ecab55df5f00b7c17d0b8",
  "name": "Geetika",
  "email": "geetika21@iitk.ac.in",
  "isAdmin": true,
  "likedEvents": [],
  "date": "2023-03-25T10:19:33.034Z",
  "__v": 0
}

```

if the user already exists then `already_exists` is true in `generateotp`

The screenshot shows a REST client interface for a POST request to `http://localhost:5000/api/auth/generateotp`. The request body is a JSON object with name `geetika`, email `geetika21@iitk.ac.in`, and password `123456`. The response is a 200 OK with success status and `already_exists` set to true.

```

POST http://localhost:5000/api/auth/generateotp
{
  "name": "geetika",
  "email": "geetika21@iitk.ac.in",
  "password": "123456"
}

Response: 200 OK, Size: 38 Bytes, Time: 482 ms
{
  "success": true,
  "already_exists": true
}

```

Structural Coverage:

- We did Decision coverage. We covered cases of user already exists, wrong password, authToken generation

3. View events by the user-

Checked the functioning of allevents functionality using ThunderClient.

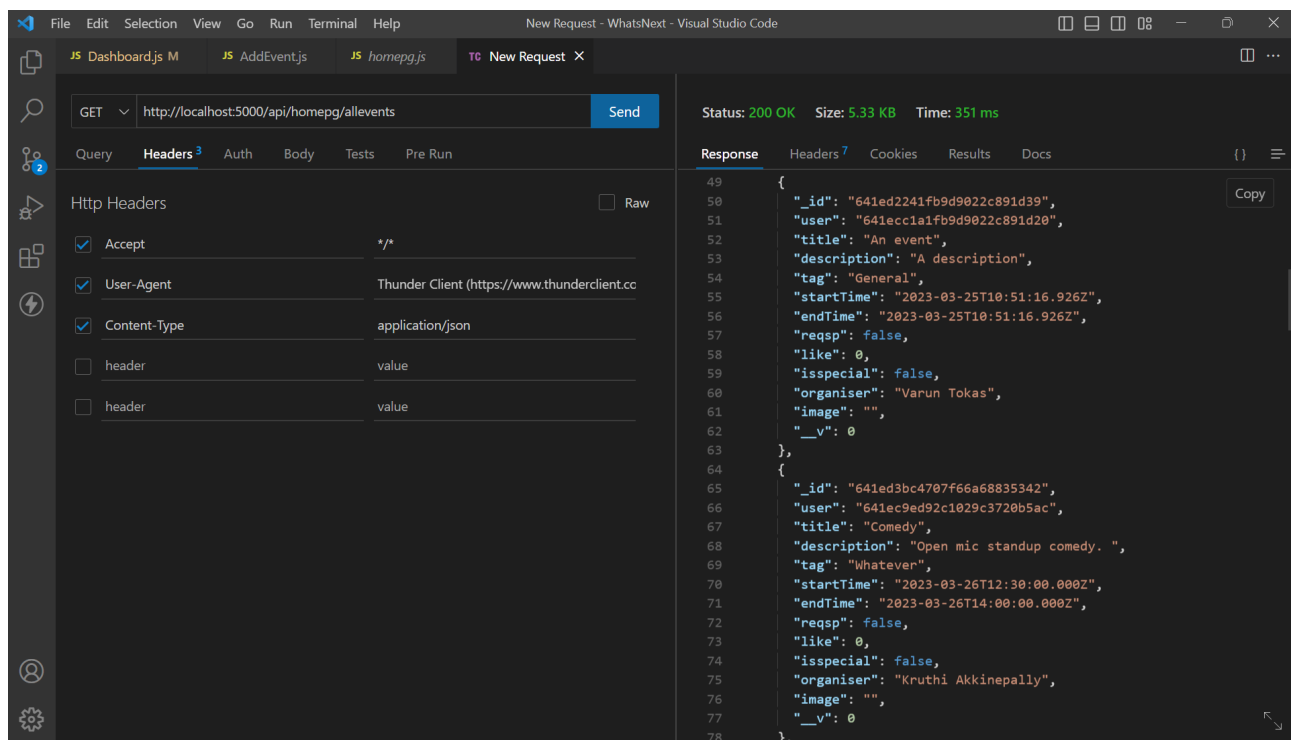
Unit Details: homepg/allevents(GET)

Test Owner: Chitwan and Kruthi

Test Date: 16-03-2023

Test Results:

1.)Display all the events added till then-



Structural Coverage:

All the cases covered in this testing:

- Called all events url and obtained all the events seen by the user which is all the events added till then.

4. Add Events-

Check the functioning of Add Events functionality using ThunderClient.

Unit Details: events/addevent(POST)

Test Owner: Varun Tokas and Geetika

Test Date: 03/25/2023 - 03/25/2023

Test Results: Checked clash algorithm and basic requests

Structural Coverage(Decision): events not correct in their start and end date covered, empty input case covered, and no duplicate clashes must occur (same event appearing twice in clashes) case checked, non clashing input covered. We have covered all types of clashes(5 different types)

Additional Comments: Basic Input

POST http://localhost:5000/api/events/addevent Send

Status: 200 OK Size: 391 Bytes Time: 2.63 s

Query Headers³ Auth¹ Body¹ Tests Pre Run

Http Headers Raw

- Accept */*
- User-Agent Thunder Client (https://www.thunderclient.com)
- auth-token eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjpb7li
- header value

Response Headers⁷ Cookies Results Docs

```
1 {
2   "success": true,
3   "warning": "Event Clashes with the following events:\n",
4   "clash": [],
5   "savedEvent": {
6     "user": "641ecc1a1fb9d9022c891d20",
7     "title": "An event",
8     "description": "A description",
9     "tag": "General",
10    "startTime": "2023-03-25T10:51:16.926Z",
11    "endTime": "2023-03-25T10:51:16.926Z",
12    "reqsp": false,
13    "like": 0,
14    "isspecial": false,
15    "organiser": "Varun Tokas",
16    "image": "",
17    "_id": "641ed2241fb9d9022c891d39",
18    "__v": 0
19  }
20 }
```

Empty Input

POST http://localhost:5000/api/events/addevent Send

Status: 400 Bad Request Size: 167 Bytes Time: 7 ms

Query Headers³ Auth¹ Body¹ Tests Pre Run

Json Content Format

```
1 {
2 }
3 }
```

Response Headers⁷ Cookies Results Docs

```
1 {
2   "errors": [
3     {
4       "msg": "Enter a valid title",
5       "param": "title",
6       "location": "body"
7     },
8     {
9       "msg": "Description must be atleast 5 characters",
10      "param": "description",
11      "location": "body"
12    }
13  ]
14 }
```

Non Clashing Input

The screenshot shows a REST client interface with the following details:

- Request:** Method: POST, URL: http://localhost:5000/api/events/addevent. The body is a JSON object:

```
{ "title": "An event1", "description": "A description", "tag": "a,b,c,d", "startTime": "2054-04-10T11:00:58.386Z", "endTime": "2054-04-20T12:00:58.386Z", "isspecial": true }
```
- Response:** Status: 200 OK, Size: 388 Bytes, Time: 2.58 s. The response body is a JSON object:

```
{ "success": true, "warning": "Event Clashes with the following events:\n", "clash": [], "savedEvent": { "user": "641ecab55df5f00b7c17d0b8", "title": "An event1", "description": "A description", "tag": "a,b,c,d", "startTime": "2054-04-10T11:00:58.386Z", "endTime": "2054-04-20T12:00:58.386Z", "reqsp": false, "like": 0, "isspecial": false, "organiser": "Geetika", "image": "", "_id": "641ee06cacce1a23c0146ff2", "_v": 0 } }
```

Various test cases for clashing input :

1. Start time of added event between start and end time of already added event

The screenshot shows a REST client interface with the following details:

- Request:** Method: POST, URL: http://localhost:5000/api/events/addevent. The body is a JSON object:

```
{ "title": "An event2", "description": "A description", "tag": "a,b,c,d", "startTime": "2054-04-11T11:00:58.386Z", "endTime": "2054-04-19T12:00:58.386Z", "isspecial": true }
```
- Response:** Status: 200 OK, Size: 1.52 KB, Time: 2.53 s. The response body is a JSON object:

```
{ "success": false, "warning": "Event Clashes with the following events:\n", "clash": [ { "_id": "641ee06cacce1a23c0146ff2", "user": "641ecab55df5f00b7c17d0b8", "title": "An event1", "description": "A description", "tag": "a,b,c,d", "startTime": "2054-04-10T11:00:58.386Z", "endTime": "2054-04-20T12:00:58.386Z", "reqsp": false, "like": 0, "isspecial": false, "organiser": "Geetika", "image": "", "_v": 0 }, { "_id": "641ee06cacce1a23c0146ff2", "user": "641ecab55df5f00b7c17d0b8", "title": "An event1",
```

2. End time of added event between start and end time of already added event

The screenshot shows a REST client interface with a POST request to `http://localhost:5000/api/events/addevent`. The request body is a JSON object:

```
1 {
2   "title": "An event4",
3   "description": "A description",
4   "tag": "a,b,c,d",
5   "startTime": "2054-04-09T11:00:58.386Z",
6   "endTime": "2054-04-21T12:00:58.386Z",
7   "isspecial": true
8 }
```

The response is a 200 OK with a size of 2.67 KB and a time of 2.82 s. The response body is a JSON object:

```
1 {
2   "success": false,
3   "warning": "Event Clashes with the following events:\n",
4   "clash": [
5     {
6       "_id": "641ee06cacce1a23c0146ff2",
7       "user": "641ecab55df5f00b7c17d0b8",
8       "title": "An event1",
9       "description": "A description",
10      "tag": "a,b,c,d",
11      "startTime": "2054-04-10T11:00:58.386Z",
12      "endTime": "2054-04-20T12:00:58.386Z",
13      "reqsp": false,
14      "like": 0,
15      "isspecial": false,
16      "organiser": "Geetika",
17      "image": "",
18      "_v": 0
19    },
20    {
21      "_id": "641ee0d3acce1a23c0146ffc",
22      "user": "641ecab55df5f00b7c17d0b8",
23      "title": "An event1",
24      "description": "A description"
```

3. Both events are at exact same time

The screenshot shows a REST client interface with a POST request to `http://localhost:5000/api/events/addevent`. The request body is a JSON object:

```
1 {
2   "title": "An events5",
3   "description": "A description",
4   "tag": "a,b,c,d",
5   "startTime": "2054-04-09T11:00:58.386Z",
6   "endTime": "2054-04-19T12:00:58.386Z",
7   "isspecial": true
8 }
```

The response is a 200 OK with a size of 2.95 KB and a time of 2.59 s. The response body is a JSON object:

```
1 {
2   "success": false,
3   "warning": "Event Clashes with the following events:\n",
4   "clash": [
5     {
6       "_id": "641ee06cacce1a23c0146ff2",
7       "user": "641ecab55df5f00b7c17d0b8",
8       "title": "An event1",
9       "description": "A description",
10      "tag": "a,b,c,d",
11      "startTime": "2054-04-10T11:00:58.386Z",
12      "endTime": "2054-04-20T12:00:58.386Z",
13      "reqsp": false,
14      "like": 0,
15      "isspecial": false,
16      "organiser": "Geetika",
17      "image": "",
18      "_v": 0
19    },
20    {
21      "_id": "641ee135acce1a23c0147010",
22      "user": "641ecab55df5f00b7c17d0b8",
23      "title": "An event3",
24      "description": "A description"
```

4. Start time of already added event between the start and end time of event to be added

```

POST http://localhost:5000/api/events/addevent
Body
{
  "title": "An event6",
  "description": "A description",
  "tag": "a,b,c,d",
  "startTime": "2054-04-10T11:00:58.386Z",
  "endTime": "2054-04-19T12:00:58.386Z",
  "isspecial": true
}
Response
{
  "success": false,
  "warning": "Event Clashes with the following events:\n",
  "clash": [
    {
      "_id": "641ee135acce1a23c0147010",
      "user": "641ecab55df5f00b7c17d0b8",
      "title": "An event3",
      "description": "A description",
      "tag": "a,b,c,d",
      "startTime": "2054-04-11T11:00:58.386Z",
      "endTime": "2054-04-21T12:00:58.386Z",
      "reqsp": false,
      "like": 0,
      "isspecial": false,
      "organiser": "Geetika",
      "image": "",
      "_v": 0
    },
    {
      "_id": "641ee166acce1a23c014701a",
      "user": "641ecab55df5f00b7c17d0b8",
      "title": "An event4",
      "description": "A description"
    }
  ]
}
    
```

5. End time of already added event between the start and end time of event to be added

```

POST http://localhost:5000/api/events/addevent
Body
{
  "title": "An event7",
  "description": "A description",
  "tag": "a,b,c,d",
  "startTime": "2054-04-11T11:00:58.386Z",
  "endTime": "2054-04-20T12:00:58.386Z",
  "isspecial": true
}
Response
{
  "success": false,
  "warning": "Event Clashes with the following events:\n",
  "clash": [
    {
      "_id": "641ee166acce1a23c014701a",
      "user": "641ecab55df5f00b7c17d0b8",
      "title": "An event4",
      "description": "A description",
      "tag": "a,b,c,d",
      "startTime": "2054-04-09T11:00:58.386Z",
      "endTime": "2054-04-21T12:00:58.386Z",
      "reqsp": false,
      "like": 0,
      "isspecial": false,
      "organiser": "Geetika",
      "image": "",
      "_v": 0
    },
    {
      "_id": "641ee166acce1a23c014701a",
      "user": "641ecab55df5f00b7c17d0b8",
      "title": "An event4",
      "description": "A description"
    }
  ]
}
    
```

5. Delete events -

Checking the functionality of deleting an event using ThunderClient

Unit Details: Event, deleteevent(DELETE)

Test Owner: Aditya and Krish Sharma

Test Date: 03/25/2023 - 03/25/2023

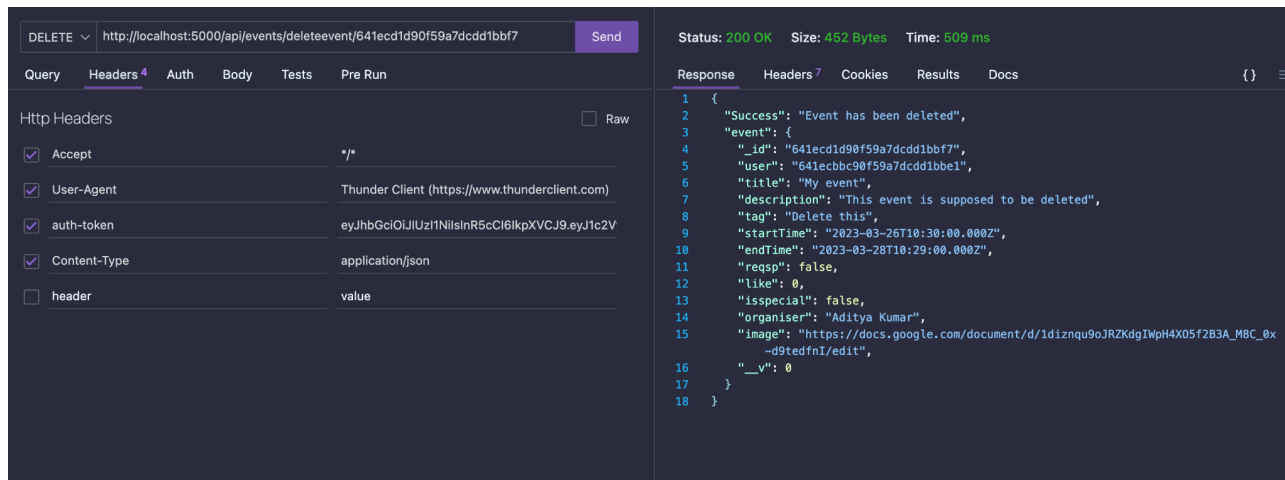
Test Result:

Case 1: If the admin is logged in:

If the admin tries to delete his own event

Expected: Event deleted from database

Response: HTTP_200_OK



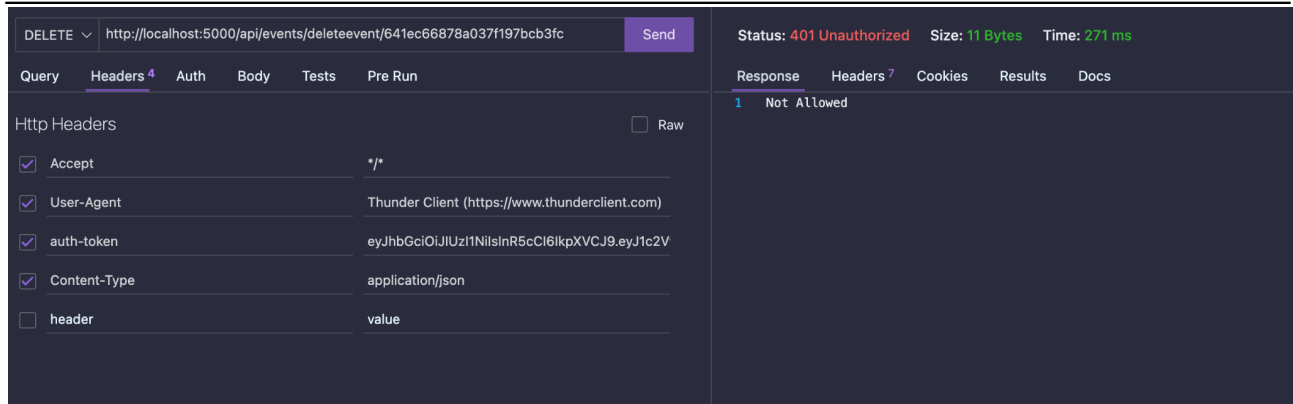
The screenshot displays the ThunderClient interface for a DELETE request. The URL is `http://localhost:5000/api/events/deleteevent/641ecd1d90f59a7dcd1bbf7`. The status is **200 OK**, with a size of **452 Bytes** and a time of **509 ms**. The response is a JSON object:

```
1 {
2   "Success": "Event has been deleted",
3   "event": {
4     "_id": "641ecd1d90f59a7dcd1bbf7",
5     "user": "641ecb90f59a7dcd1bbe1",
6     "title": "My event",
7     "description": "This event is supposed to be deleted",
8     "tag": "Delete this",
9     "startTime": "2023-03-26T10:30:00.000Z",
10    "endTime": "2023-03-28T10:29:00.000Z",
11    "reqsp": false,
12    "like": 0,
13    "isspecial": false,
14    "organiser": "Aditya Kumar",
15    "image": "https://docs.google.com/document/d/1dizqu9oJRZKdIWpH4X05f2B3A_M8C_0x-d9tedfnI/edit",
16    "_v": 0
17  }
18 }
```

If the admin tries to delete other admin's event

Expected: Error: Not allowed

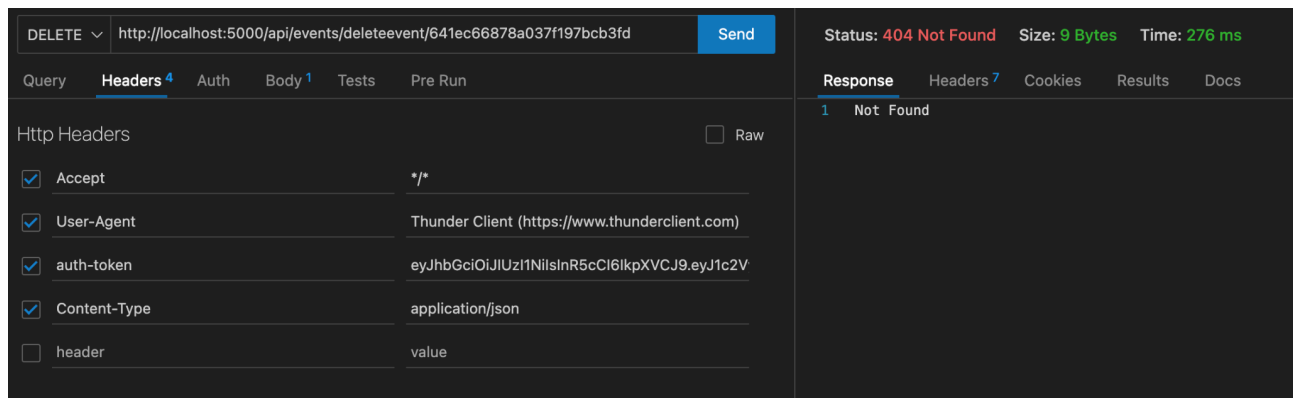
Response: HTTP_401_Unauthorize



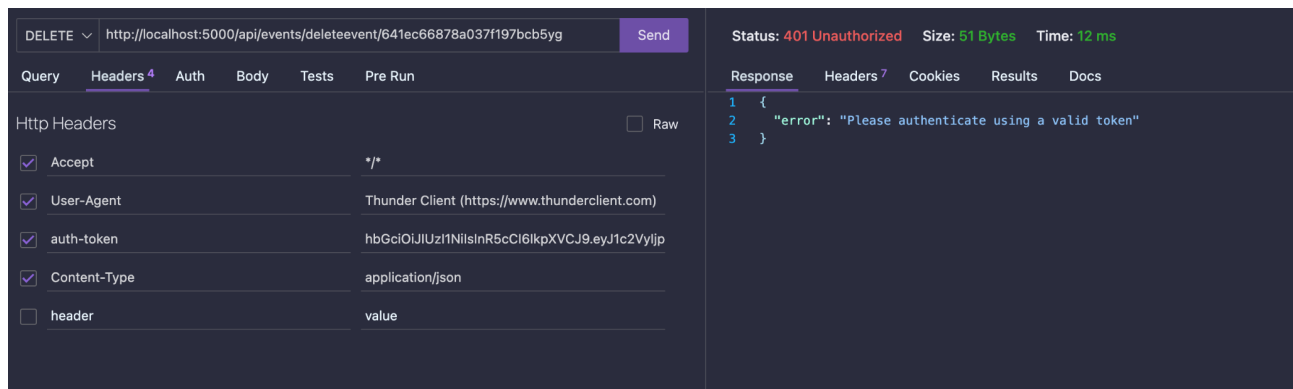
If the admin tries to delete an event that does not exist

Expected: Not Found

Result: HTTP_404_Not_Found



Case 2: If the user is not logged in or tries to delete using the wrong auth-token.



Structural Coverage : Decision Coverage. Covered cases of user not logged in, when the user tries to delete other user's events, when the user tries to delete his own event

6. Edit events-

Check the functioning of update events functionality using ThunderClient.

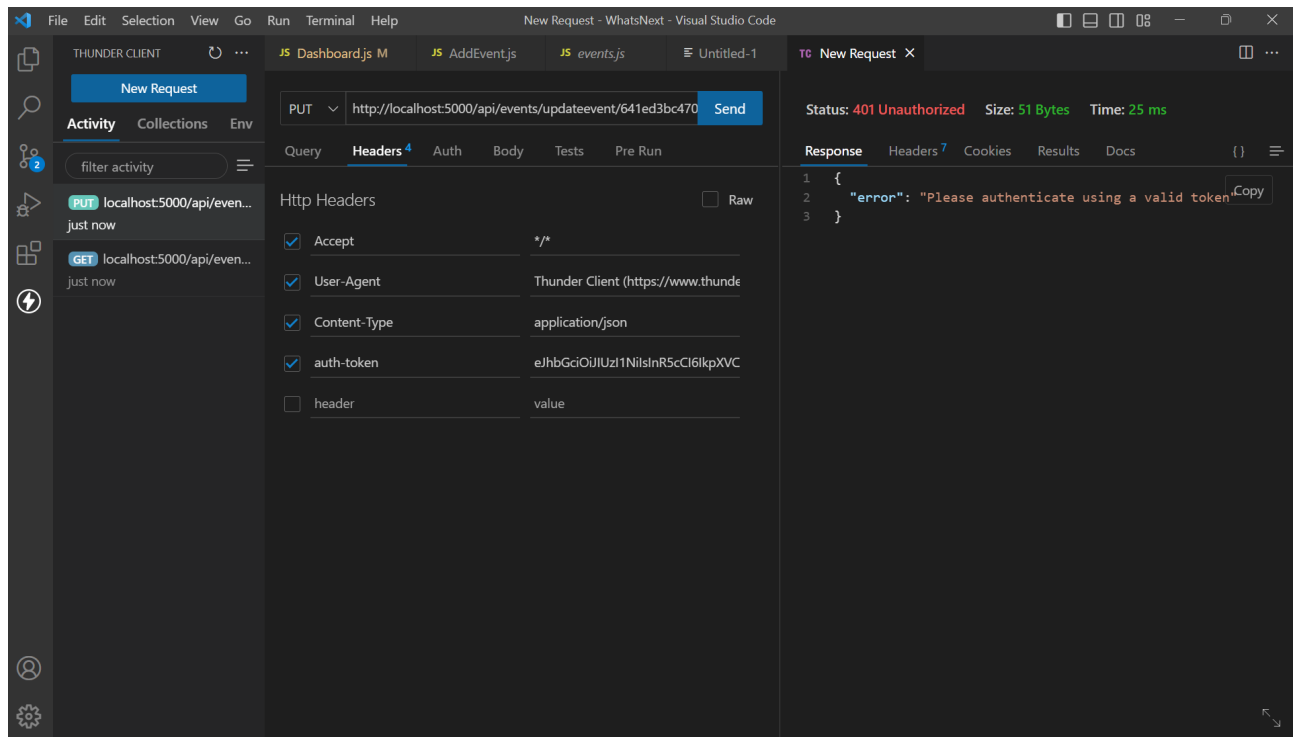
Unit Details: updateevents(routes/event.js) (PUT)

Test Owner: Kruthi and Chitwan

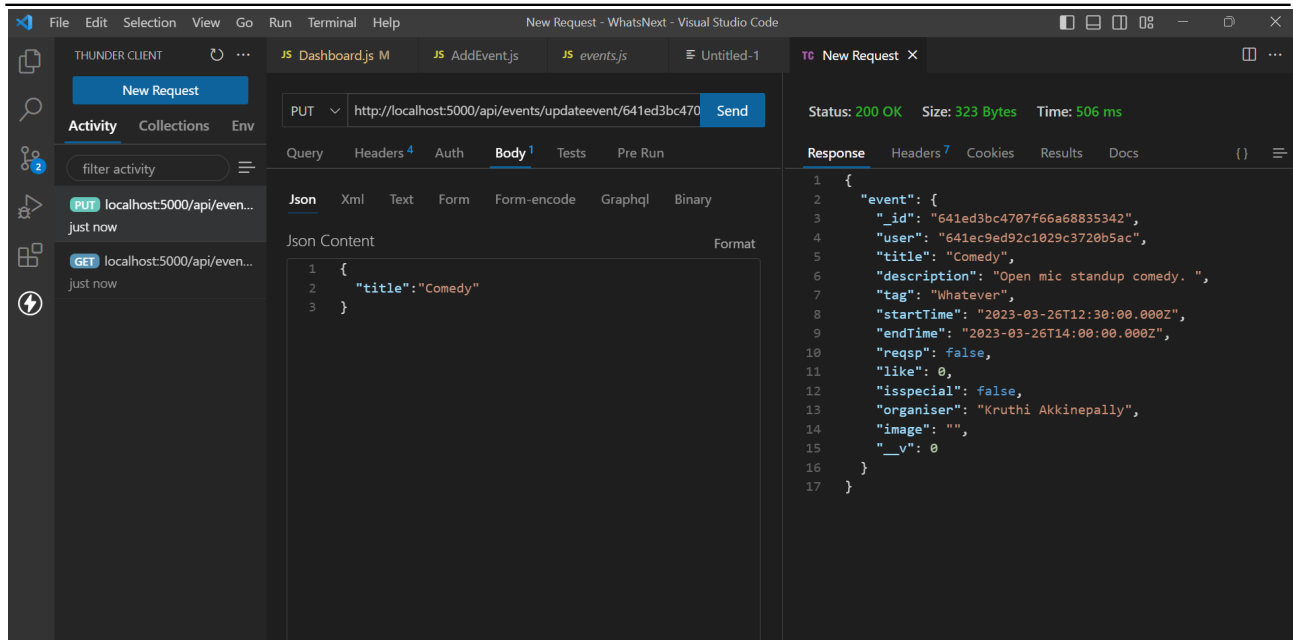
Test Date: 03/25/2023 - 03/25/2023

Test Results:

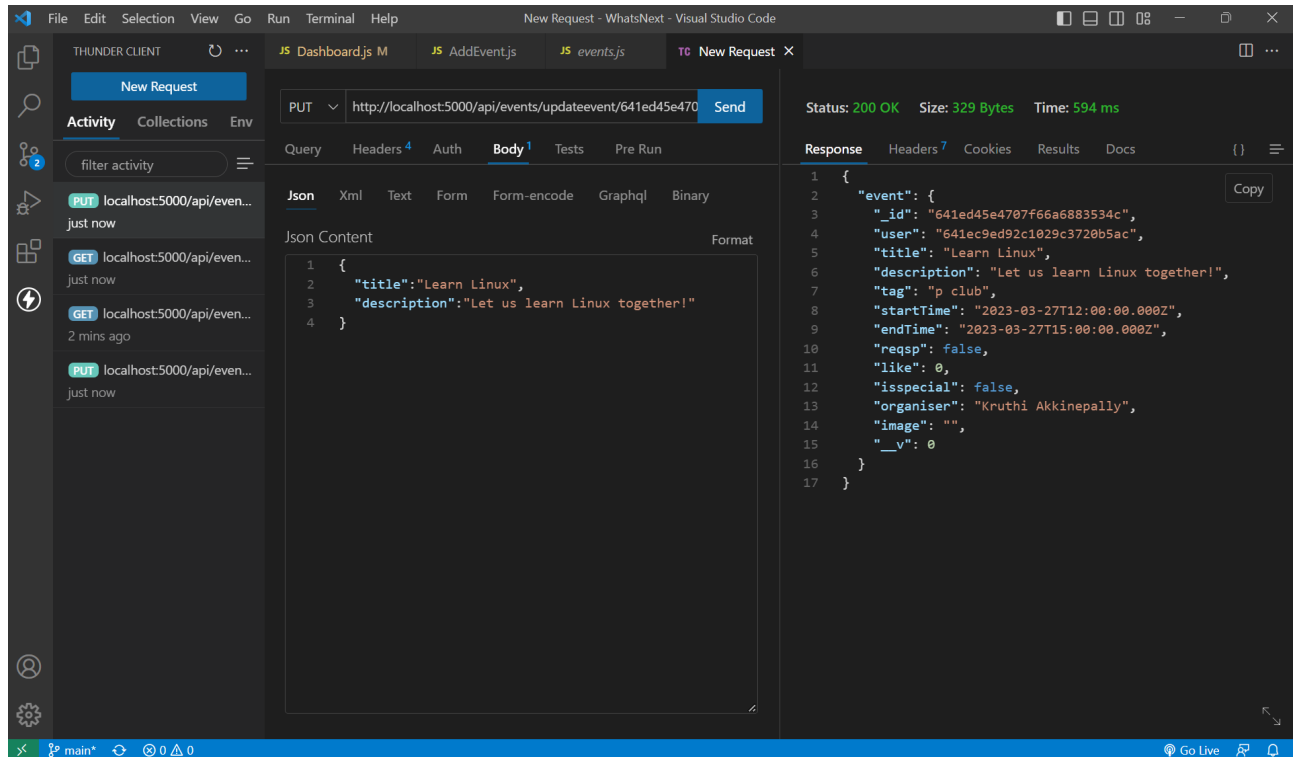
1) Enter a wrong authorization token and check if an error is being displayed.



2) Tried to edit the title of one of the events added by the corresponding user-



3) Tried to edit the title and title description of one of the events added by the corresponding user-



Structural Coverage: (Decision coverage)

All the cases covered in this testing:

- Entered a wrong authorization token.
- Changing one of the components of one of the events added by the corresponding user.
- Changing more than one component(simultaneously) of one of the events added by the corresponding user.

7. See Special Requests

Unit Details: routes/superadmin.js seerequests(GET)

Test Owner: Aditya Kumar and Siddharth Kalra

Test Date: 25-03-2023

Test Result:

Expected Result: All requests for special events will be returned to the user.

Response: HTTP_200_OK

Result: An array of all the events for which reqsp = true is returned. (reqsp means requested for special event)

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:5000/api/superadmin/seerequests
- Status:** 200 OK
- Size:** 1.75 KB
- Time:** 490 ms
- Headers:** Accept: */*, User-Agent: Thunder Client (https://www.thunderclient.com), auth-token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2V... Content-Type: application/json
- Response (JSON):**

```
1 {
2   {
3     "_id": "641edeb9acce1a23c0146fd4",
4     "user": "641ecab55df5f00b7c17d0b8",
5     "title": "not happy hour",
6     "description": "will try to make u sad",
7     "tag": "cse",
8     "startTime": "2023-04-01T18:29:00.000Z",
9     "endTime": "2023-04-02T14:53:00.000Z",
10    "reqsp": true,
11    "like": 0,
12    "isspecial": false,
13    "organiser": "Geetika",
14    "image": "",
15    "_v": 0
16  },
17  {
18    "_id": "6423e7fad5575ca5ee31157c",
19    "user": "641ecab55df5f00b7c17d0b8",
20    "title": "Linux Install fest",
21    "description": "we will install ubuntu in your pcs",
22    "tag": "pclub",
23    "startTime": "2023-04-03T04:30:00.000Z",
24    "endTime": "2023-04-03T06:30:00.000Z",
25    "reqsp": true,
26    "like": 0,
27    "isspecial": false,
28    "organiser": "Geetika",
29    "image": ""
30  }
31 }
```

Structural Coverage: Statement coverage was used.

8. Special Request accepted

Unit Details: routes/superadmin.js approveevent(PUT)

Test Owner: Aditya Kumar and Krish Sharma

Test Date: 25-03-2023

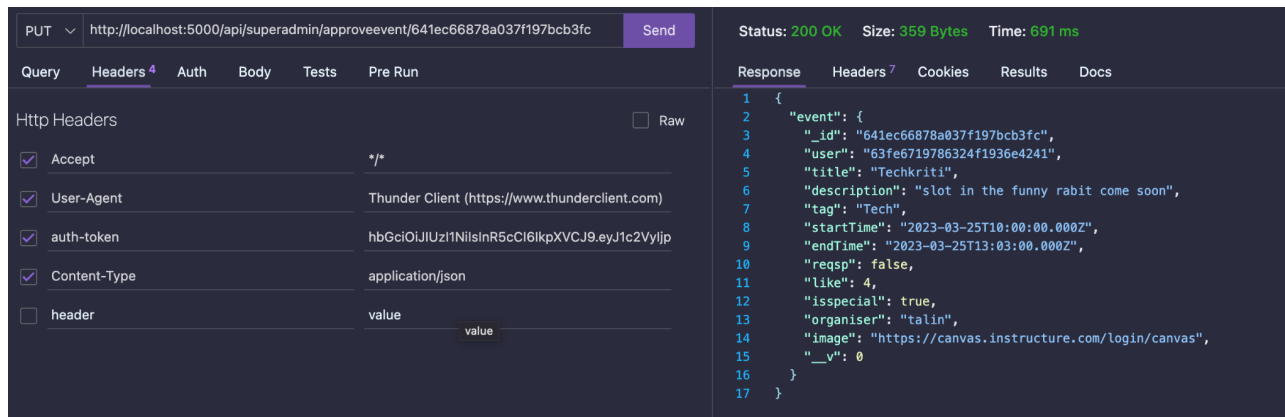
Test Result:

Case -1 - The event whose request was approved exists.

Expected Result - The event was approved to be special

Response - HTTP_200_OK

Result - The event's isspecial field was returned to be true.

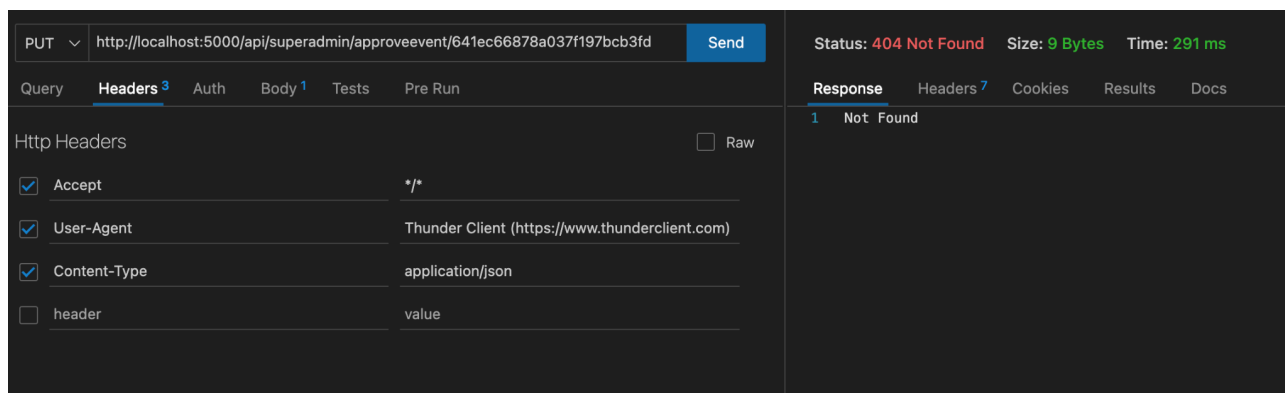


Case - 2 The event whose request was approved does not exist

Expected result - The event does not exist

Response - HTTP_404_Not_Found

Result - Not Found



Structural Coverage: Decision coverage was used

9. Special Request denied -

Unit Details: routes/superadmin.js denyevent(PUT)

Test Owner: Aditya Kumar and Siddharth Kalra

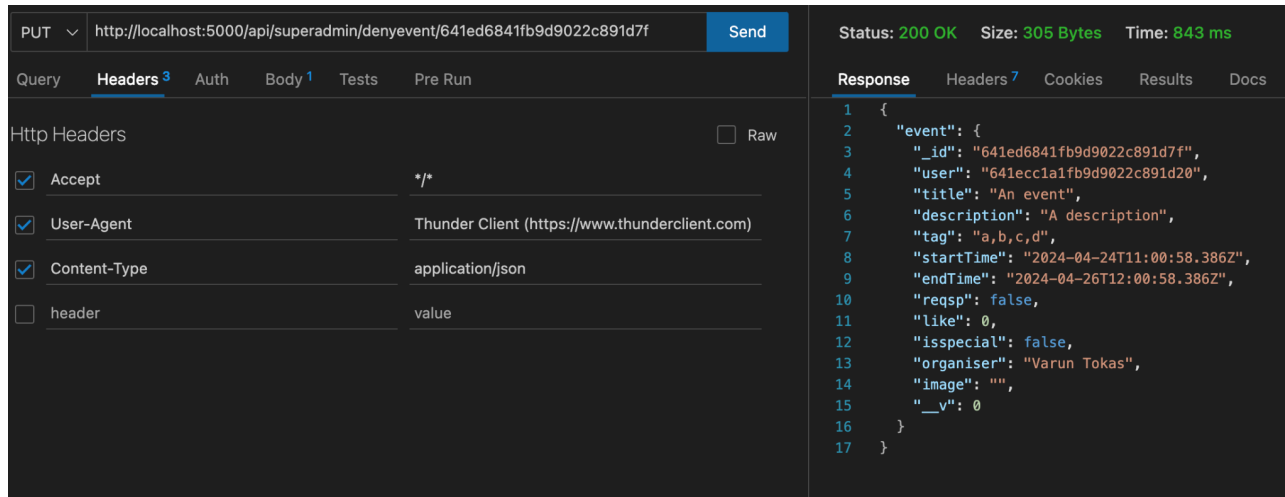
Test Date: 25-03-2023

Test Result:

Case -1 - The event whose request was denied exists.

Expected Result - The event was denied to be special

Response - HTTP_200_OK

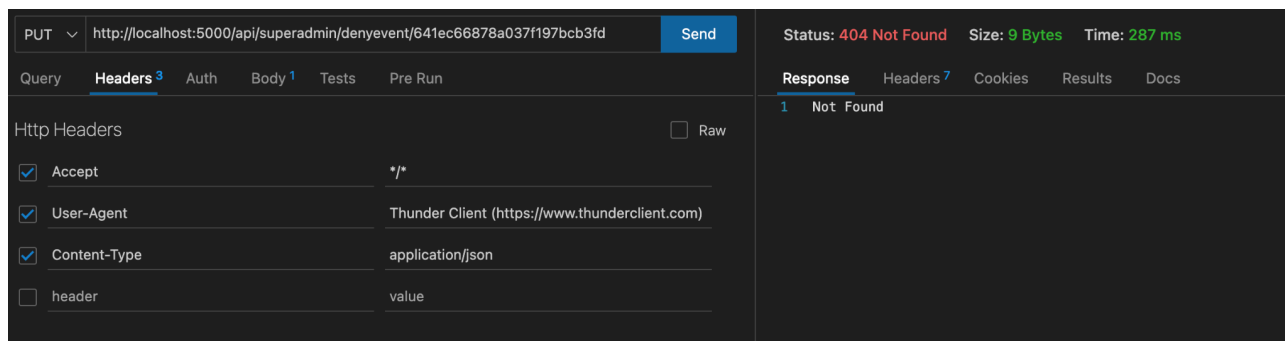


Case - 2 The event whose request was denied does not exist

Expected result - The event does not exist

Response - HTTP_404_Not_Found

Result - Not Found



Result - The event's isspecial field was returned to be false.

Result: An array of all the events for which reqsp = true is returned. (reqsp means requested for special event)

Structural Coverage: Branch Coverage was used.

9. Password is encrypted

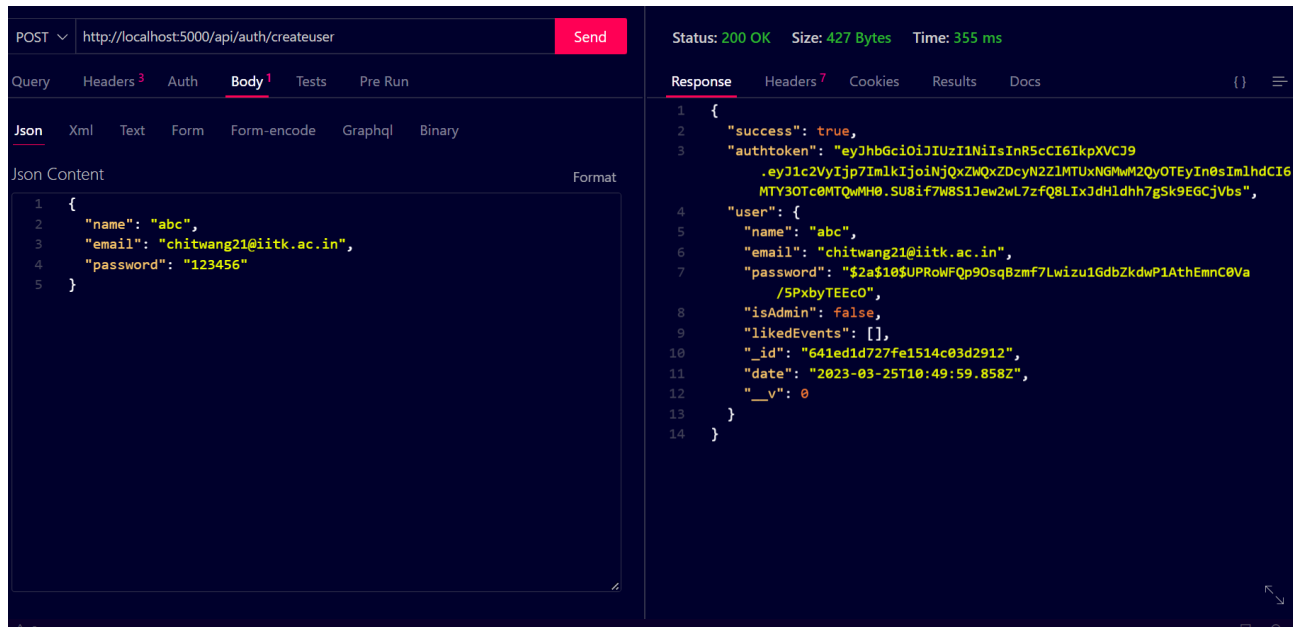
Check that password is indeed encrypted

Unit Details: routes/auth.js createuser (POST)

Test Owner: Apoorva Gupta and Paras Sikarwar

Test Date: 25-03-2023

Test Results: Password stored in database is not the same as that entered by the user. Statement Coverage has been used.



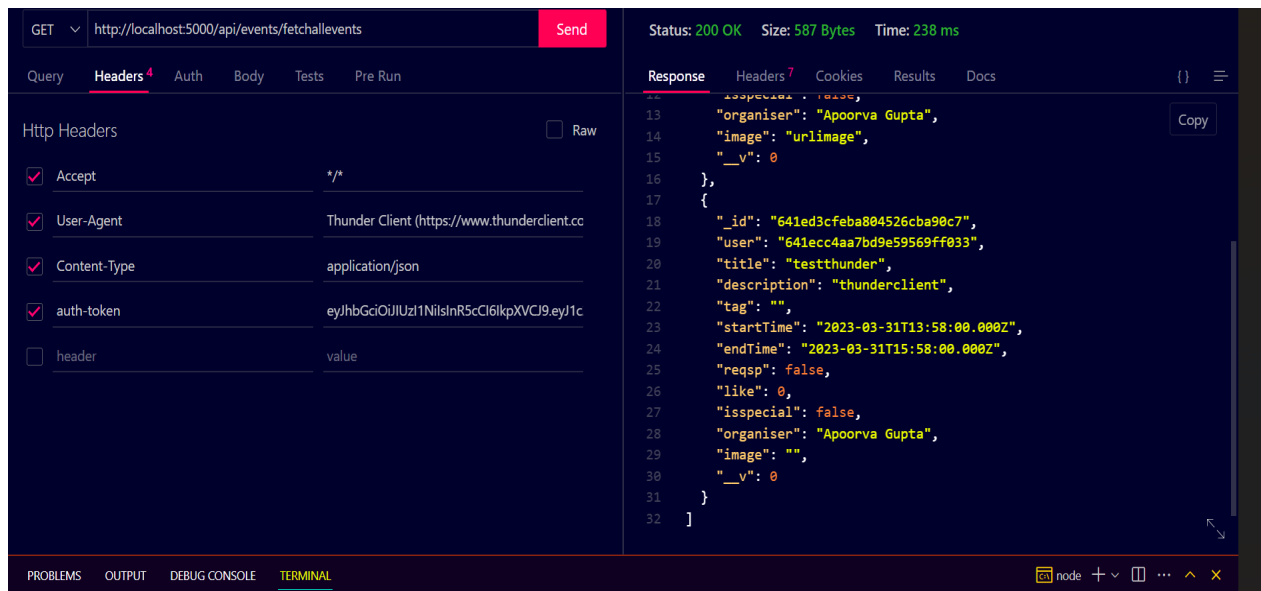
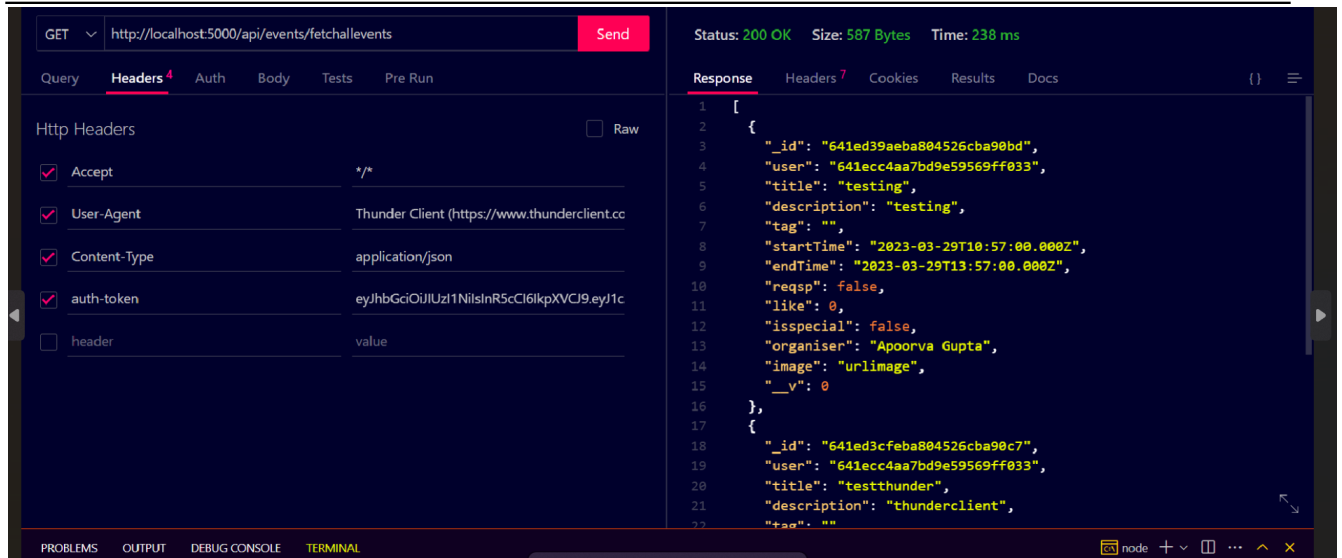
10. Fetching all events

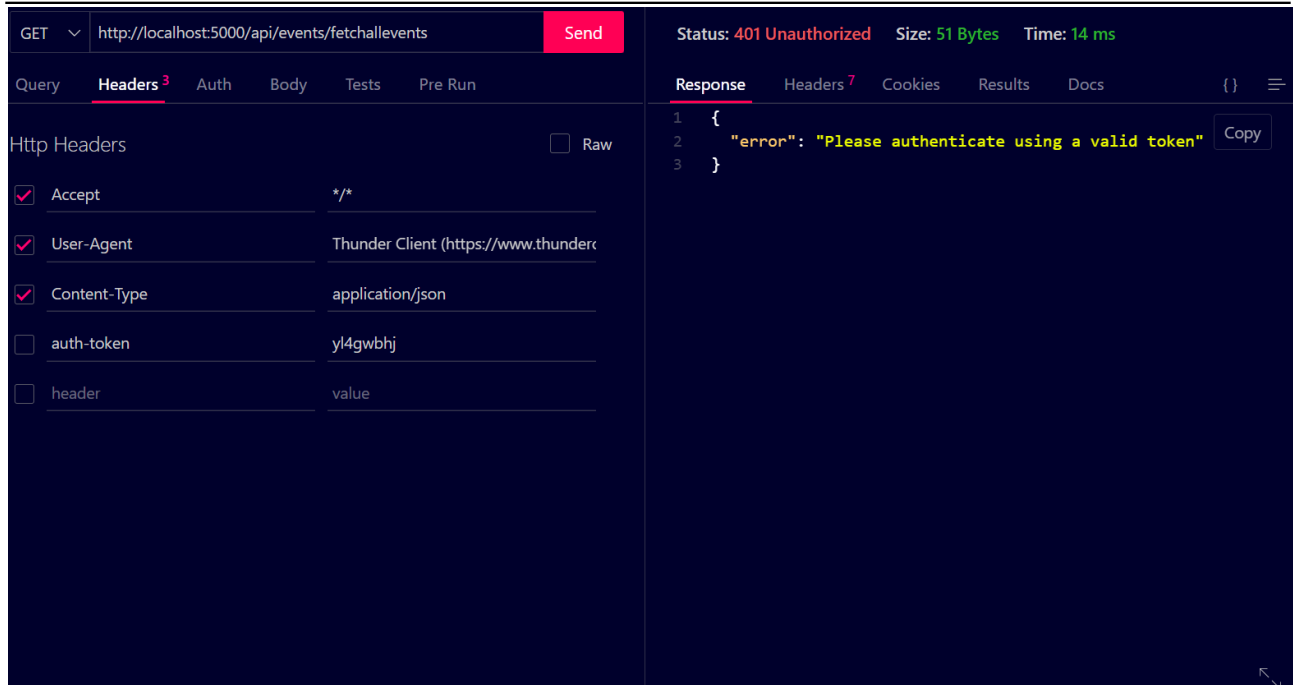
Unit Details : routes/events.js fetchallevents (GET)

Test Owner: Apoorva Gupta and Chitwan Goel

Test Date: 25-03-2023

Test Results: admin is able to view all its events





Structural Coverage : We have used Decision coverage. Events will be shown only if the user has valid auth-token (case of invalid also covered)

11. View Liked Events

Unit Details: routes/generaluser.js showlikedevents (GET)

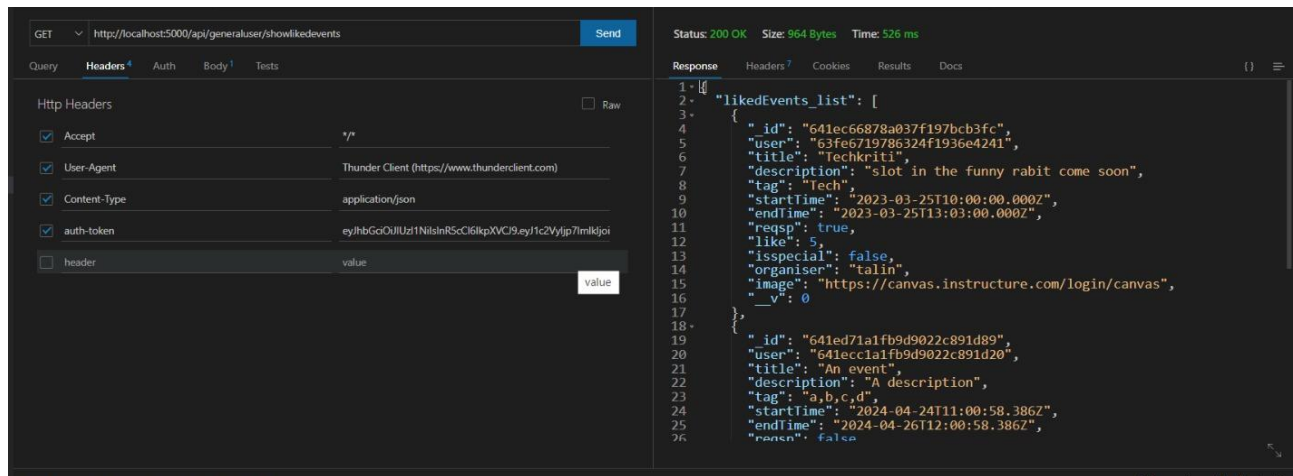
Test Owner: Talin Gupta

Test Date: 25-03-2023

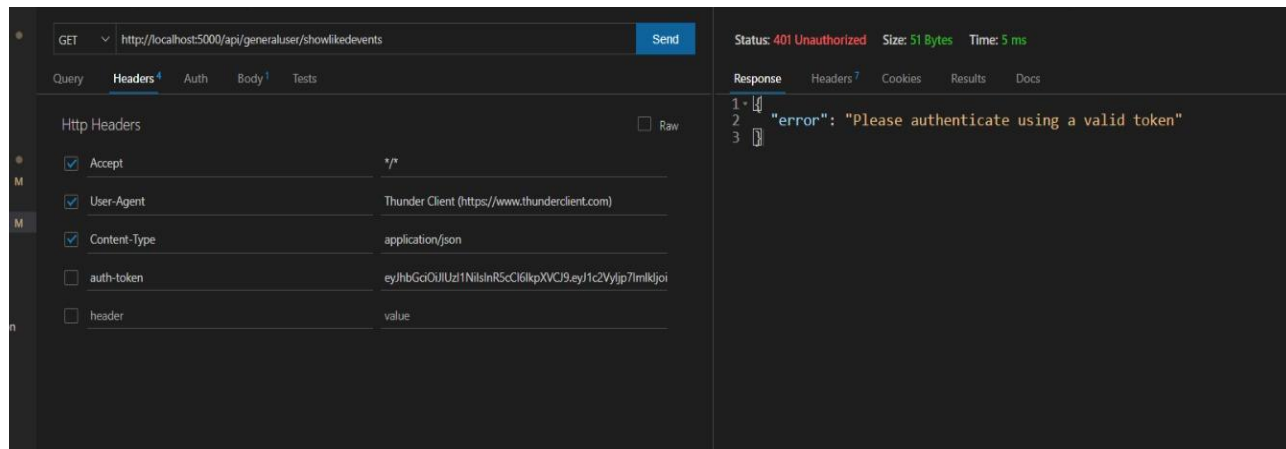
Test Results: user is able to view all his/her liked events

Structural Coverage: (Decision Coverage) Covered cases where user is logged in as well as not logged in

Additional Comments: All liked events are visible, in sorted order of time of happening.



In case user isn't logged in, it will give error:



12. Liking an event

Unit Details: `routes/generaluser.js likeevent (POST)`

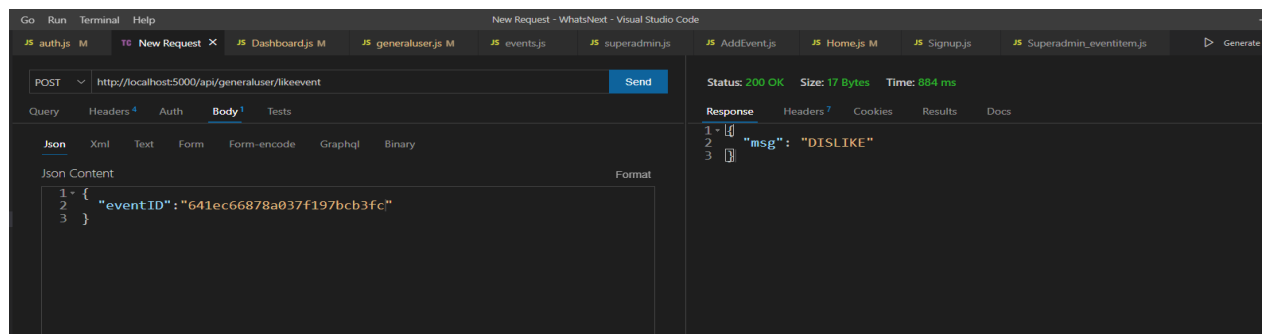
Test Owner: Talin Gupta

Test Date: 25-03-2023

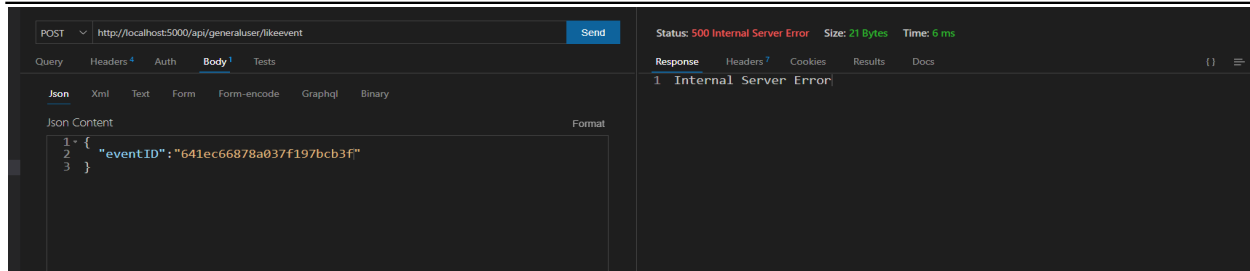
Test Results: user is able to like an event

Structural Coverage (Decision Coverage): Covered cases where event id exist and not exists

Additional Comments: The msg indicates the text on button to be displayed after clicking, in this case it should be Dislike



In case event id doesn't exist, internal server error will be given.



13. Dislike an event

Unit Details: routes/generaluser.js likeevent (POST)

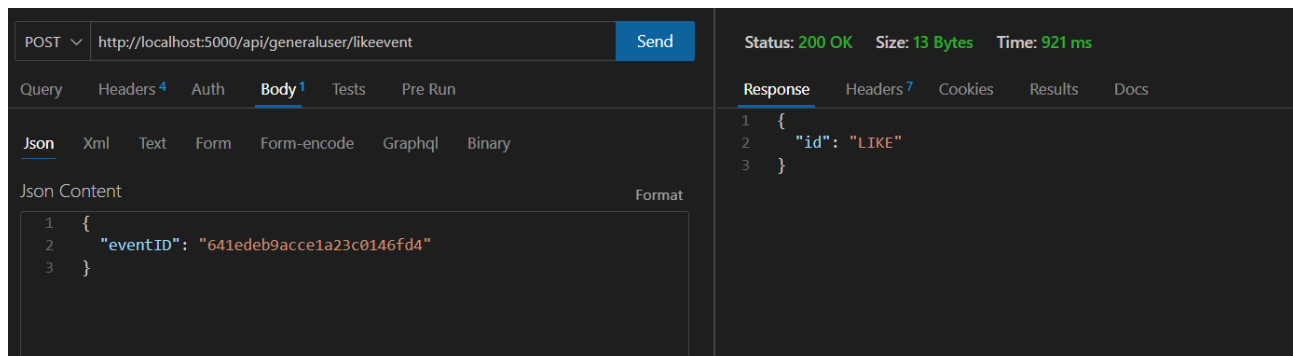
Test Owner: Talin Gupta, Siddharth Kalra

Test Date: 25-03-2023

Test Results: user is able to like an event

Structural Coverage: Decision Coverage : Covered cases where event id exist and not exists

Additional Comments: Api is the same(likeevent) but serves different purpose here. The msg here, as expected is "LIKE"



3 Integration Testing

1. Home Page

Several api's were integrated with the frontend of the dashboard and they are fetchallevents, addevent, deleteevent and updateevent. The fetchallevents api was found to be correctly working as the frontend was showing the correct details of all the events added by an admin on his dashboard . Upon the integration of the frontend and backend components of dashboard every functionality was found to be working properly and that too within the consistency of the requirements.

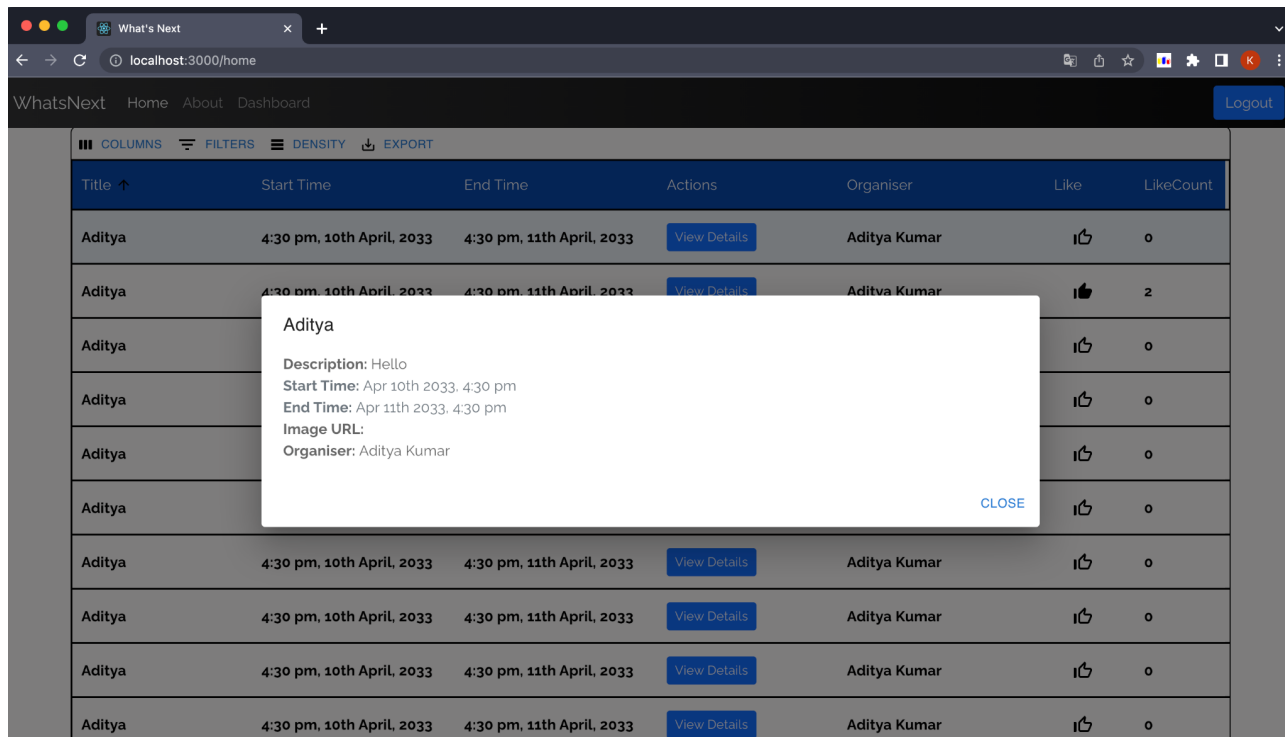
Module Details: Integrated the frontend of dashboard and multiple apis such as addevent, fetch all events, updateevent, deleteevent, etc.

Test Owner: *Krish and Apoorva*

Test Date: 25/03/2023

Test Results: In this test we successfully showcased that the "event details" button is working properly. Also, the sorting functionality is working correctly on each column i.e the list of event is getting sorted according to start time, end time , organizers and like count. The export functionality is working properly i.e the schedule can be downloaded as a .csv file.

Test 1: event details working properly i.e description of the event is popping up by clicking on the event details button.



Test 2: Sorting functionality is working correctly

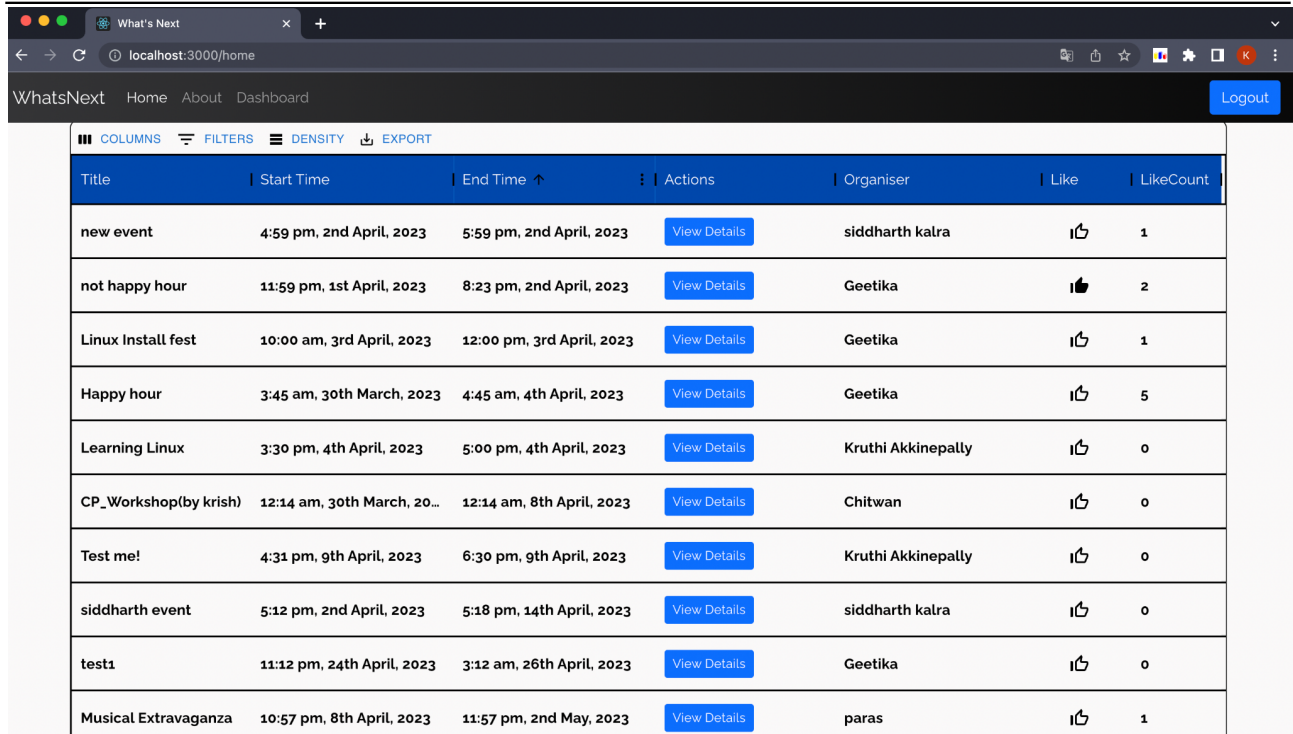
2.1 Sorting according to title

Title ↑	Start Time	End Time	Actions	Organiser	Like	LikeCount
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	2
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	0

2.2 Sorting according to start time

Title	Start Time ↑	End Time	Actions	Organiser	Like	LikeCount
CP_Workshop(by krish)	12:14 am, 30th March, 20...	12:14 am, 8th April, 2023	View Details	Chitwan	👍	0
Happy hour	3:45 am, 30th March, 2023	4:45 am, 4th April, 2023	View Details	Geetika	👍	5
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika	👍	2
new event	4:59 pm, 2nd April, 2023	5:59 pm, 2nd April, 2023	View Details	siddharth kalra	👍	1
siddharth event	5:12 pm, 2nd April, 2023	5:18 pm, 14th April, 2023	View Details	siddharth kalra	👍	0
Linux Install fest	10:00 am, 3rd April, 2023	12:00 pm, 3rd April, 2023	View Details	Geetika	👍	1
Learning Linux	3:30 pm, 4th April, 2023	5:00 pm, 4th April, 2023	View Details	Kruthi Akkinepally	👍	0
Musical Extravaganza	10:57 pm, 8th April, 2023	11:57 pm, 2nd May, 2023	View Details	paras	👍	1
Test me!	4:31 pm, 9th April, 2023	6:30 pm, 9th April, 2023	View Details	Kruthi Akkinepally	👍	0
test1	11:12 pm, 24th April, 2023	3:12 am, 26th April, 2023	View Details	Geetika	👍	0

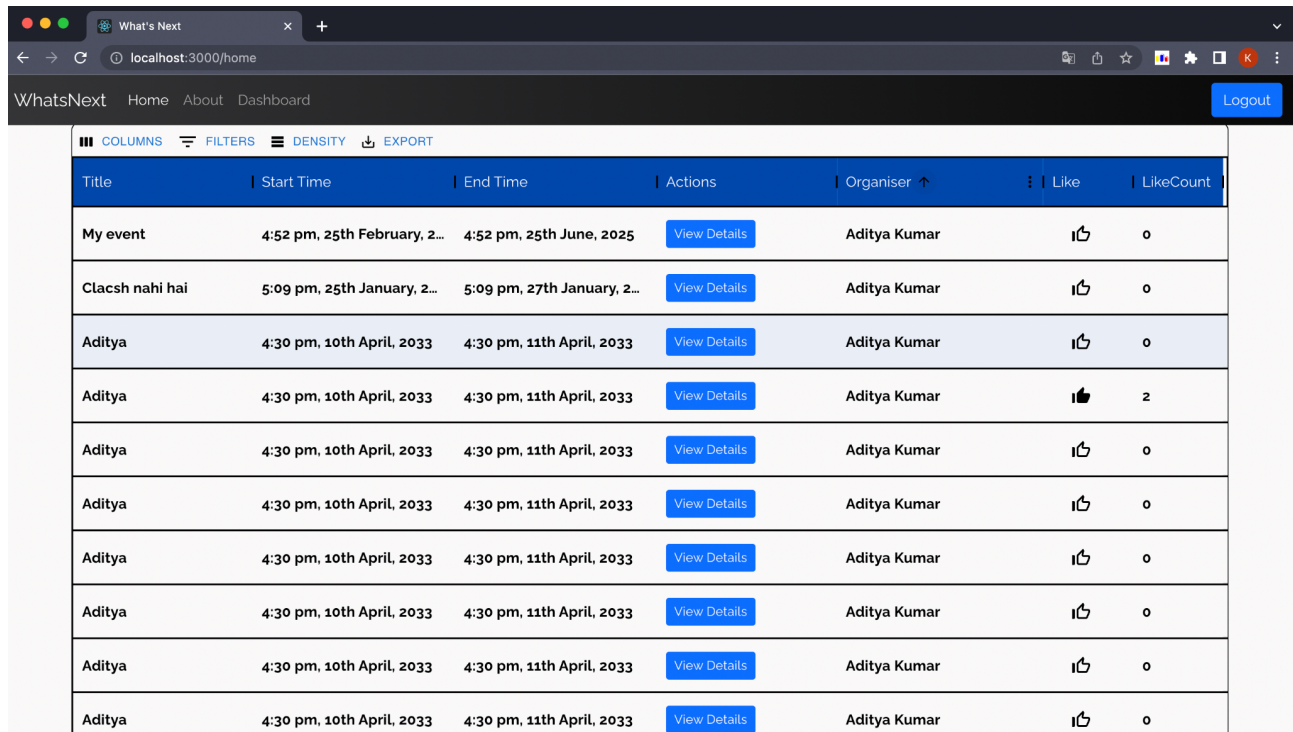
2.3 Sorting according to end time



The screenshot shows a web browser window with the URL localhost:3000/home. The page title is 'WhatsNext' and it has navigation links for Home, About, and Dashboard, along with a Logout button. Below the navigation is a table with columns: Title, Start Time, End Time, Actions, Organiser, Like, and LikeCount. The table is sorted by the 'Organiser' column. The data rows are as follows:

Title	Start Time	End Time	Actions	Organiser	Like	LikeCount
new event	4:59 pm, 2nd April, 2023	5:59 pm, 2nd April, 2023	View Details	siddharth kalra		1
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika		2
Linux Install fest	10:00 am, 3rd April, 2023	12:00 pm, 3rd April, 2023	View Details	Geetika		1
Happy hour	3:45 am, 30th March, 2023	4:45 am, 4th April, 2023	View Details	Geetika		5
Learning Linux	3:30 pm, 4th April, 2023	5:00 pm, 4th April, 2023	View Details	Kruthi Akkinepally		0
CP_Workshop(by krish)	12:14 am, 30th March, 20...	12:14 am, 8th April, 2023	View Details	Chitwan		0
Test me!	4:31 pm, 9th April, 2023	6:30 pm, 9th April, 2023	View Details	Kruthi Akkinepally		0
siddharth event	5:12 pm, 2nd April, 2023	5:18 pm, 14th April, 2023	View Details	siddharth kalra		0
test1	11:12 pm, 24th April, 2023	3:12 am, 26th April, 2023	View Details	Geetika		0
Musical Extravaganza	10:57 pm, 8th April, 2023	11:57 pm, 2nd May, 2023	View Details	paras		1

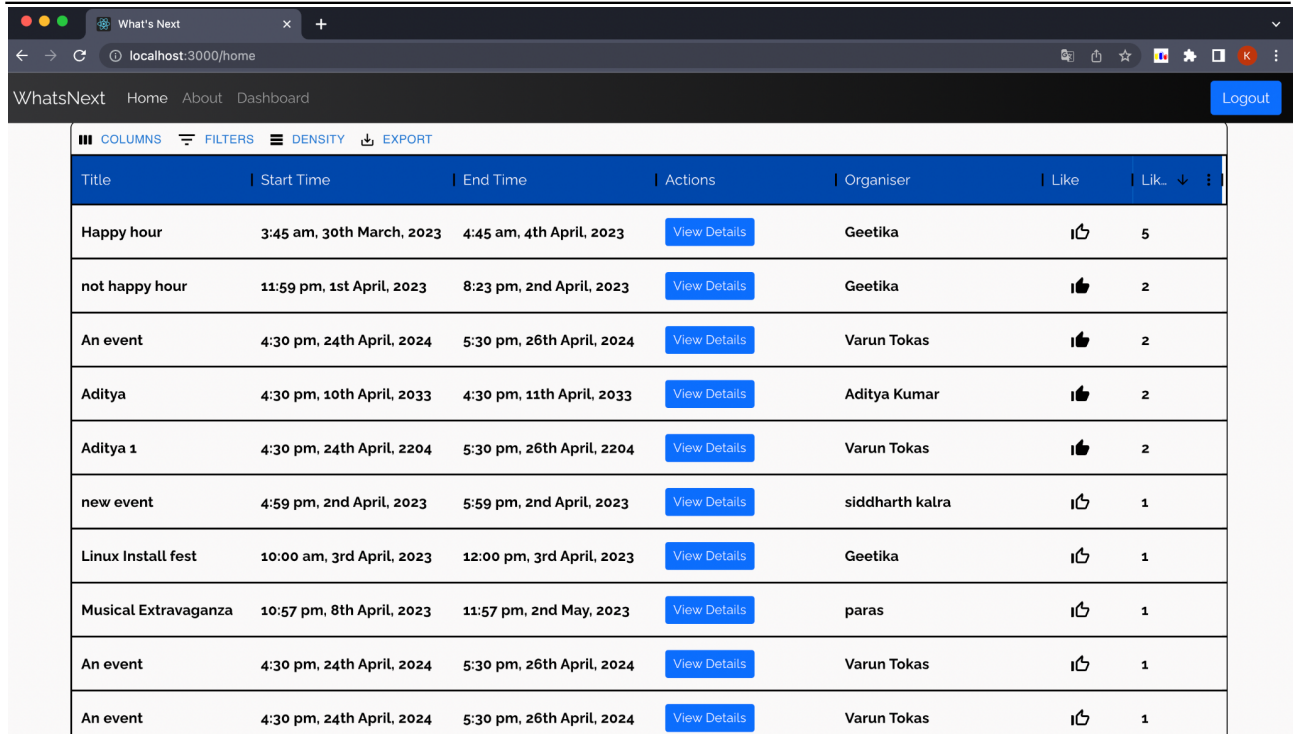
2.4 Sorting according to organizers



The screenshot shows a web browser window with the URL localhost:3000/home. The page title is 'WhatsNext' and it has navigation links for Home, About, and Dashboard, along with a Logout button. Below the navigation is a table with columns: Title, Start Time, End Time, Actions, Organiser, Like, and LikeCount. The table is sorted by the 'Organiser' column. The data rows are as follows:

Title	Start Time	End Time	Actions	Organiser	Like	LikeCount
My event	4:52 pm, 25th February, 2...	4:52 pm, 25th June, 2025	View Details	Aditya Kumar		0
Clacsh nahi hai	5:09 pm, 25th January, 2...	5:09 pm, 27th January, 2...	View Details	Aditya Kumar		0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		2
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		0
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar		0

2.5 Sorting according to like counts



The screenshot shows a web browser window with the URL localhost:3000/home. The page title is 'WhatsNext' and it has a navigation menu with 'Home', 'About', and 'Dashboard'. A 'Logout' button is in the top right. Below the navigation is a table with columns: Title, Start Time, End Time, Actions, Organiser, Like, and Lik. The table contains 11 rows of event data.

Title	Start Time	End Time	Actions	Organiser	Like	Lik. ↓
Happy hour	3:45 am, 30th March, 2023	4:45 am, 4th April, 2023	View Details	Geetika	👍	5
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika	👍	2
An event	4:30 pm, 24th April, 2024	5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	2
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	2
Aditya 1	4:30 pm, 24th April, 2204	5:30 pm, 26th April, 2204	View Details	Varun Tokas	👍	2
new event	4:59 pm, 2nd April, 2023	5:59 pm, 2nd April, 2023	View Details	siddharth kalra	👍	1
Linux Install fest	10:00 am, 3rd April, 2023	12:00 pm, 3rd April, 2023	View Details	Geetika	👍	1
Musical Extravaganza	10:57 pm, 8th April, 2023	11:57 pm, 2nd May, 2023	View Details	paras	👍	1
An event	4:30 pm, 24th April, 2024	5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	1
An event	4:30 pm, 24th April, 2024	5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	1

Test 3: Export functionality working properly that is the correct schedule is getting downloaded.

Software Design Document for Codecrafters

28

COLUMNS FILTERS DENSITY EXPORT							
Title	Start Time	Download as CSV	End Time	Actions	Organiser	Like	LikeCo. ↓
Happy hour	3:45 am, 30th March, 2023	Print	4:45 am, 4th April, 2023	View Details	Geetika	👍	5
not happy hour	11:59 pm, 1st April, 2023		8:23 pm, 2nd April, 2023	View Details	Geetika	👍	2
An event	4:30 pm, 24th April, 2024		5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	2
Aditya	4:30 pm, 10th April, 2033		4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	2
Aditya 1	4:30 pm, 24th April, 2204		5:30 pm, 26th April, 2204	View Details	Varun Tokas	👍	2
new event	4:59 pm, 2nd April, 2023		5:59 pm, 2nd April, 2023	View Details	siddharth kalra	👍	1
Linux Install fest	10:00 am, 3rd April, 2023		12:00 pm, 3rd April, 2023	View Details	Geetika	👍	1
Musical Extravaganza	10:57 pm, 8th April, 2023		11:57 pm, 2nd May, 2023	View Details	paras	👍	1
An event	4:30 pm, 24th April, 2024		5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	1
An event	4:30 pm, 24th April, 2024		5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	1
Aditya	4:30 pm, 2nd April, 2036		4:30 pm, 5th April, 2036	View Details	Aditya Kumar	👍	1

COLUMNS FILTERS DENSITY EXPORT							
Title	Start Time	End Time	Actions	Organiser	Like	LikeCo. ↓	
Happy hour	3:45 am, 30th March, 2023	4:45 am, 4th April, 2023	View Details	Geetika	👍	5	
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika	👍	2	
An event	4:30 pm, 24th April, 2024	5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	2	
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar	👍	2	
Aditya 1	4:30 pm, 24th April, 2204	5:30 pm, 26th April, 2204	View Details	Varun Tokas	👍	2	
new event	4:59 pm, 2nd April, 2023	5:59 pm, 2nd April, 2023	View Details	siddharth kalra	👍	1	
Linux Install fest	10:00 am, 3rd April, 2023	12:00 pm, 3rd April, 2023	View Details	Geetika	👍	1	
Musical Extravaganza	10:57 pm, 8th April, 2023	11:57 pm, 2nd May, 2023	View Details	paras	👍	1	
An event	4:30 pm, 24th April, 2024	5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	1	
An event	4:30 pm, 24th April, 2024	5:30 pm, 26th April, 2024	View Details	Varun Tokas	👍	1	

2. Like and show liked events

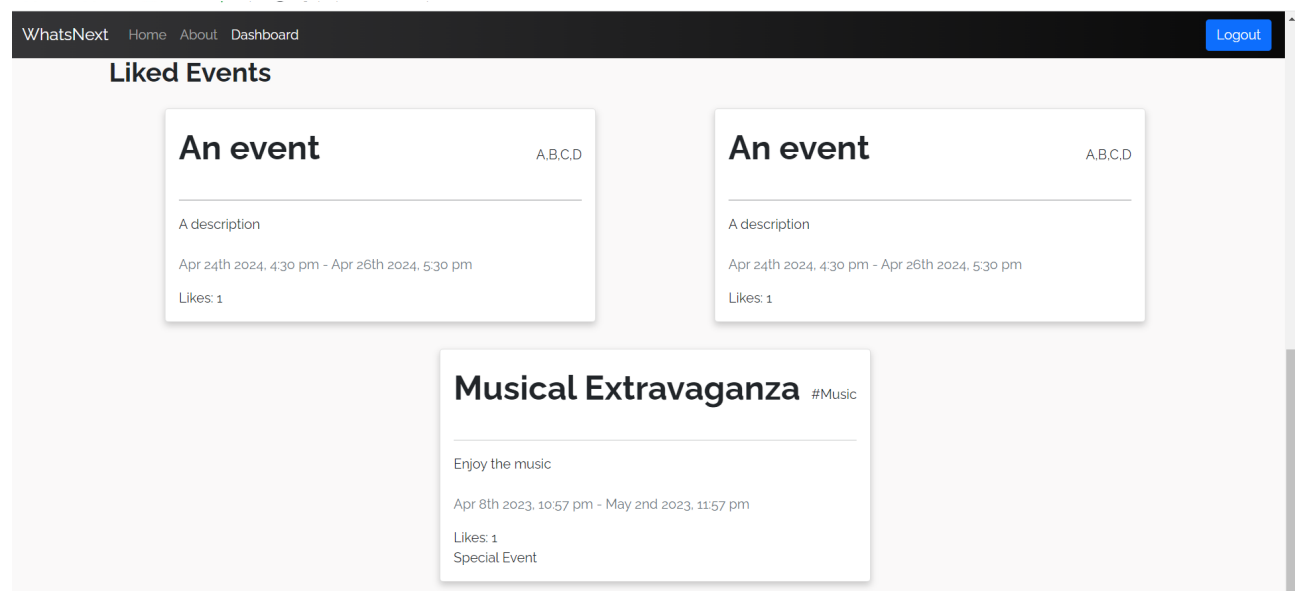
Module Details: In this test we tested the implementation of api's such as showlikedevents and likeevent on the frontend of this component.

Test Owner: Krish and Aman

Test Date: 25/03/2023

Test Results: In this test we successfully showcased that the events like count were working correctly, there were no false increments in the likes of the event and the correct number of likes were also to be displayed correctly. Also, on dislike, the event gets disliked. Apart from this, all other functionalities such as displaying time start and end time , and the details of the organisers which were stored in the database were successfully fetched by the api's and displayed correctly on the frontend.

Liked events on dashboard :



Like an event :

Test case: Like button of an event clicked

All Events

Title	Start Time	End Time	Actions	Organiser	Like	LikeCount
Tired	10:00 am, 29th March, 20...	1:30 pm, 31st March, 2023	View Details	Kruthi Akkinepally		0
Silent Disco	9:58 pm, 30th March, 2023	9:58 pm, 31st March, 2023	View Details	paras		0
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika		0

Test result: Like count of that event and only that event is increased, and the like button is activated and the event liked is also shown on dashboard.

Title	Start Time	End Time	Actions	Organiser	Like	LikeCount
Tired	10:00 am, 29th March, 20...	1:30 pm, 31st March, 2023	View Details	Kruthi Akkinepally		1
Silent Disco	9:58 pm, 30th March, 2023	9:58 pm, 31st March, 2023	View Details	paras		0
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika		0

WhatsNext Home About Dashboard [Logout](#)

Liked Events

An event A.B.C.D

A description

Apr 24th 2024, 4:30 pm - Apr 26th 2024, 5:30 pm

Likes: 1

Musical Extravaganza #Music

Enjoy the music

Apr 8th 2023, 10:57 pm - May 2nd 2023, 11:57 pm

Likes: 1
Special Event

An event A.B.C.D

A description

Apr 24th 2024, 4:30 pm - Apr 26th 2024, 5:30 pm

Likes: 1

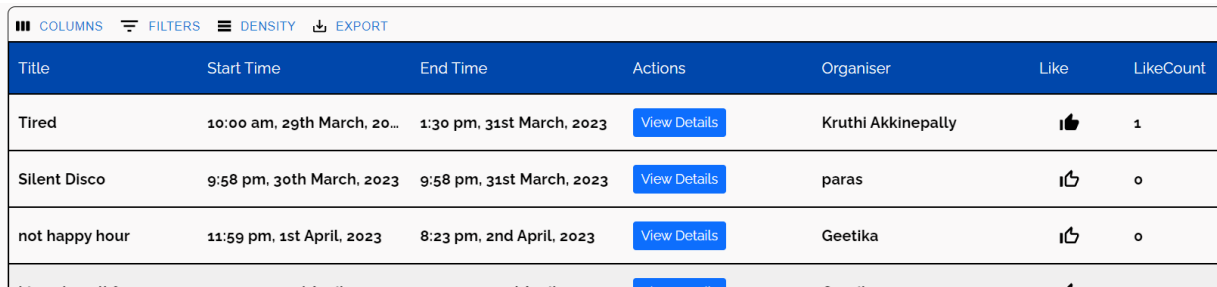
Tired Let's Go!

Sab bhaag jao yaha se!

Mar 29th 2023, 10:00 am - Mar 31st 2023, 1:30 pm

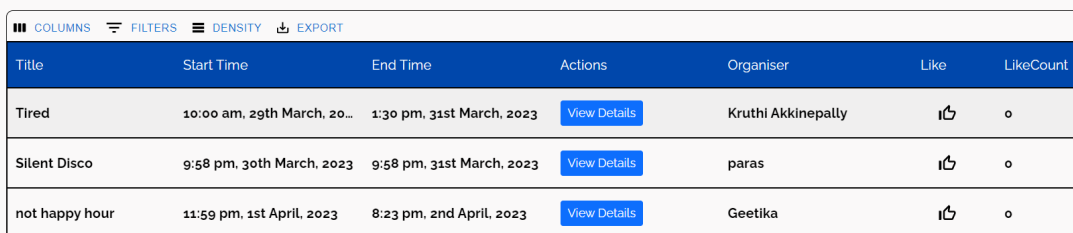
Likes: 1

Test case: Liked event is disliked by clicking on the like button again:



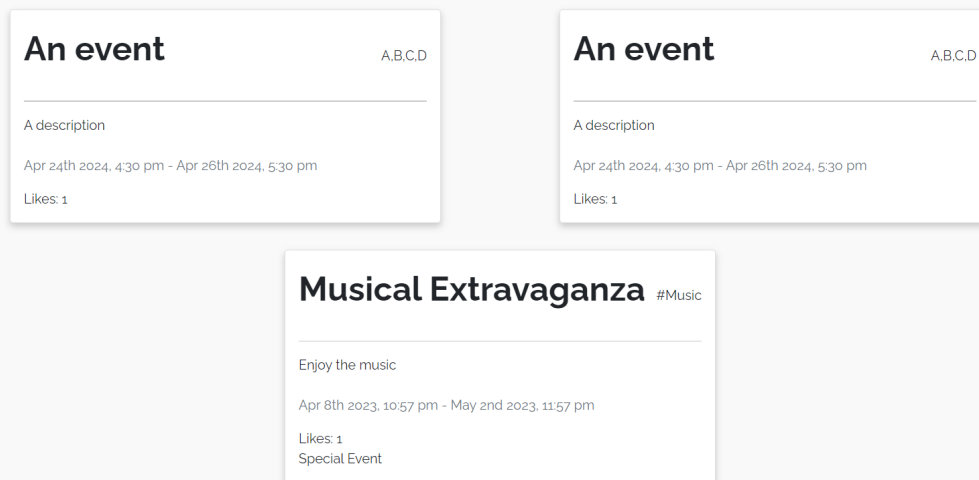
Title	Start Time	End Time	Actions	Organiser	Like	LikeCount
Tired	10:00 am, 29th March, 20...	1:30 pm, 31st March, 2023	View Details	Kruthi Akkinepally		1
Silent Disco	9:58 pm, 30th March, 2023	9:58 pm, 31st March, 2023	View Details	paras		0
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika		0

Test result: The event’s like count is decreased by one, like button is restored to its original state and the event is removed from the “Liked events” in user dashboard.



Title	Start Time	End Time	Actions	Organiser	Like	LikeCount
Tired	10:00 am, 29th March, 20...	1:30 pm, 31st March, 2023	View Details	Kruthi Akkinepally		0
Silent Disco	9:58 pm, 30th March, 2023	9:58 pm, 31st March, 2023	View Details	paras		0
not happy hour	11:59 pm, 1st April, 2023	8:23 pm, 2nd April, 2023	View Details	Geetika		0

Liked Events



The 'Liked Events' dashboard displays three event cards. The top two cards are identical, each titled 'An event' with a truncated title 'A.B.C.D' and a description 'A description'. The dates for these events are 'Apr 24th 2024, 4:30 pm - Apr 26th 2024, 5:30 pm' and they each have 'Likes: 1'. The third card is titled 'Musical Extravaganza #Music' with a description 'Enjoy the music'. Its dates are 'Apr 8th 2023, 10:57 pm - May 2nd 2023, 11:57 pm' and it also has 'Likes: 1' and is categorized as a 'Special Event'.

Test case : Multiple users liking a single event while logged in

Test result : Like of one user was shown to other user only on manual reloading the page.

Test case : Liking an event when a filter was added

Test result : Like works fine but on automatic reloading of page the filter gets removed.

3. Login and Signup

Module Details: In this we tested the different login (admin and non admin) and signup+otp verification through various api calls including 'generate otp', 'createuser', 'login' as well as the frontend for dashboard based on different logins

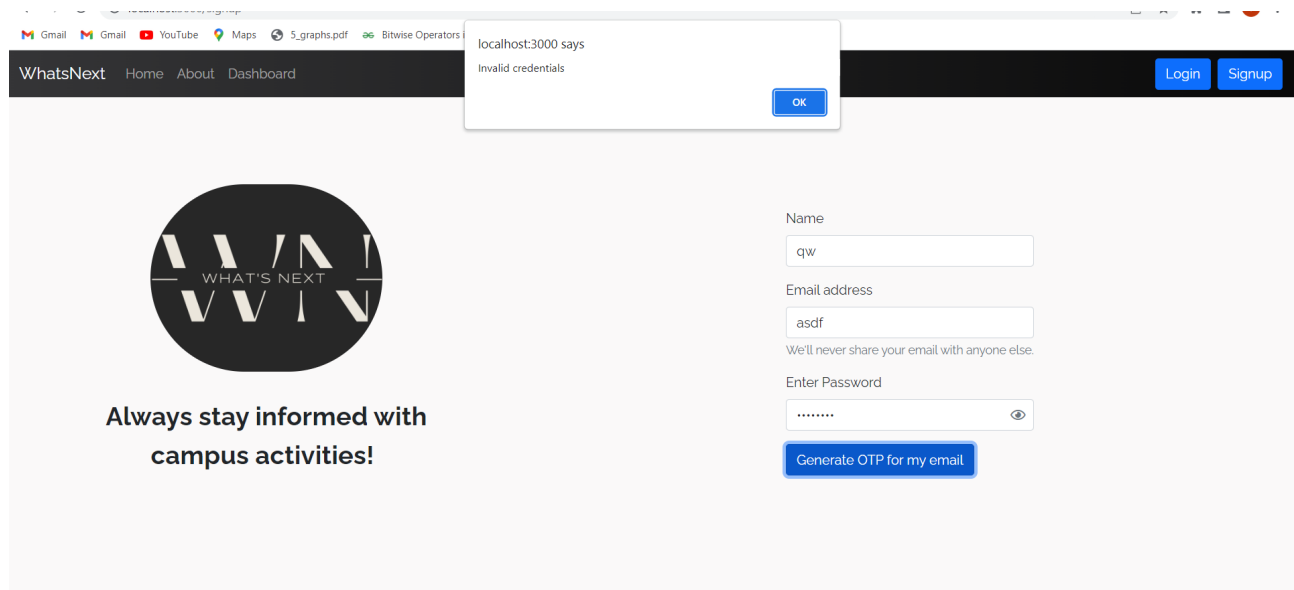
Test Owner: Talin

Test Date: 25/03/2023

Test Results: In this test we scrutinized the integration of the backend and frontend components of Signup and Login functionality of our web application . The backend was found to be correctly delivering with consistency . Here we are integrating the Login api , createuser api and generateotp api with the frontend. The integration was found to be successful, users were able to generate otps while signup and the frontend was able to send data to the database with the help of backend apis to successfully furnish the functionalities expected from Login and Signup

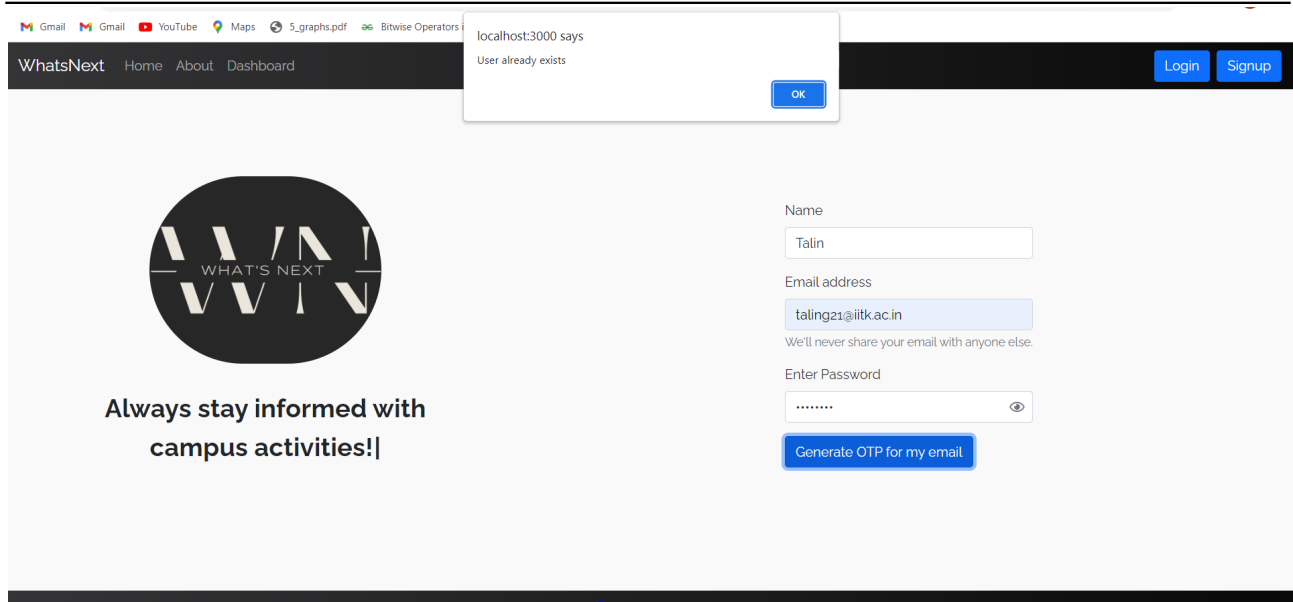
Signup generates otp requests: 30 seconds break between 2 consecutive otp requests to avoid spam.

Test : Invalid Credentials(invalid email id or password length < 5)



Result : popup showing invalid credentials appears, and no otp generated

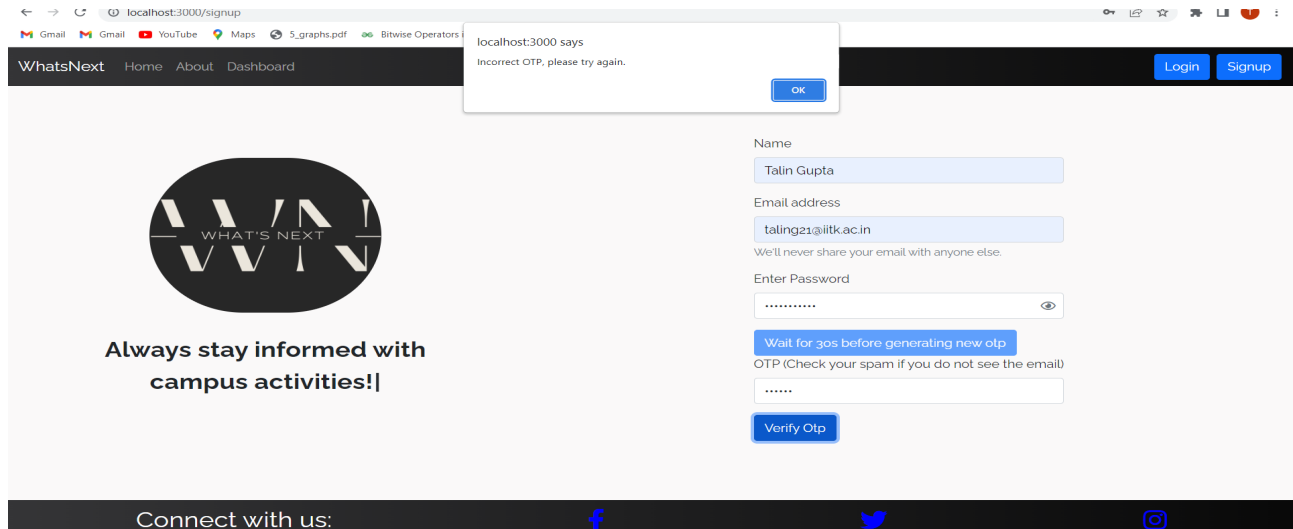
Test case : Email already present:



Result : Prompt showing user already exists appears

Otp verification : In this we tested the different login (admin and non admin) and signup+otp verification through various api calls including 'generateotp', 'createuser', 'login' as well as the frontend for dashboard based on different logins

Test case : incorrect otp given

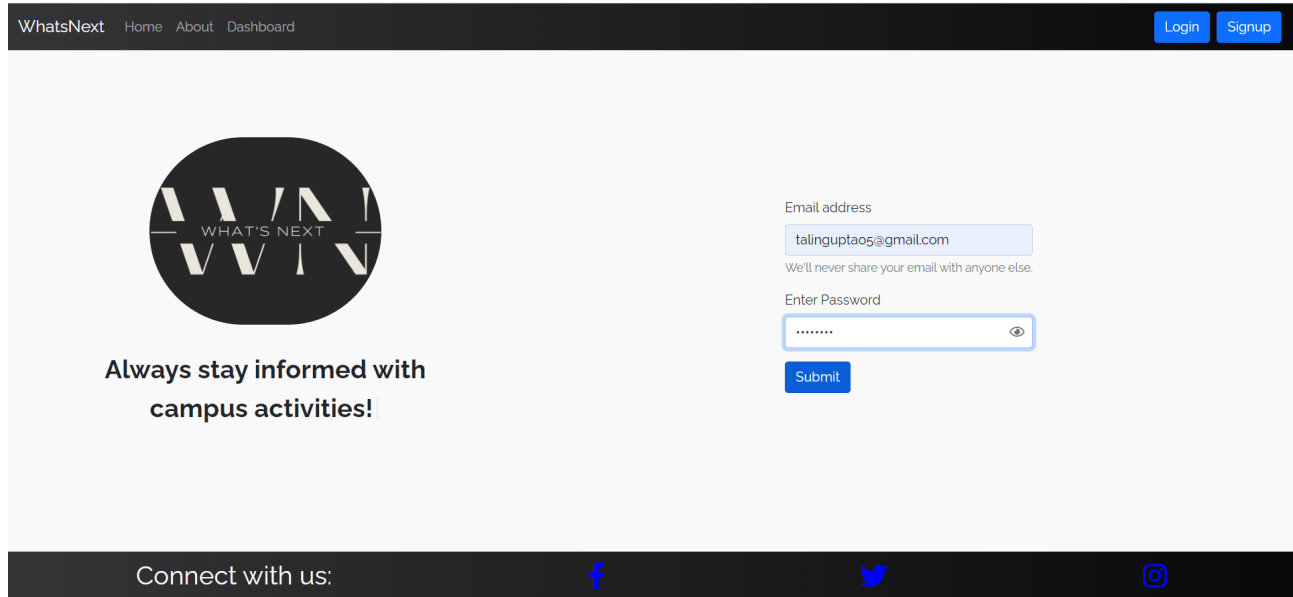


Result : prompt showing invalid otp appears

Test case: Otp verification successful

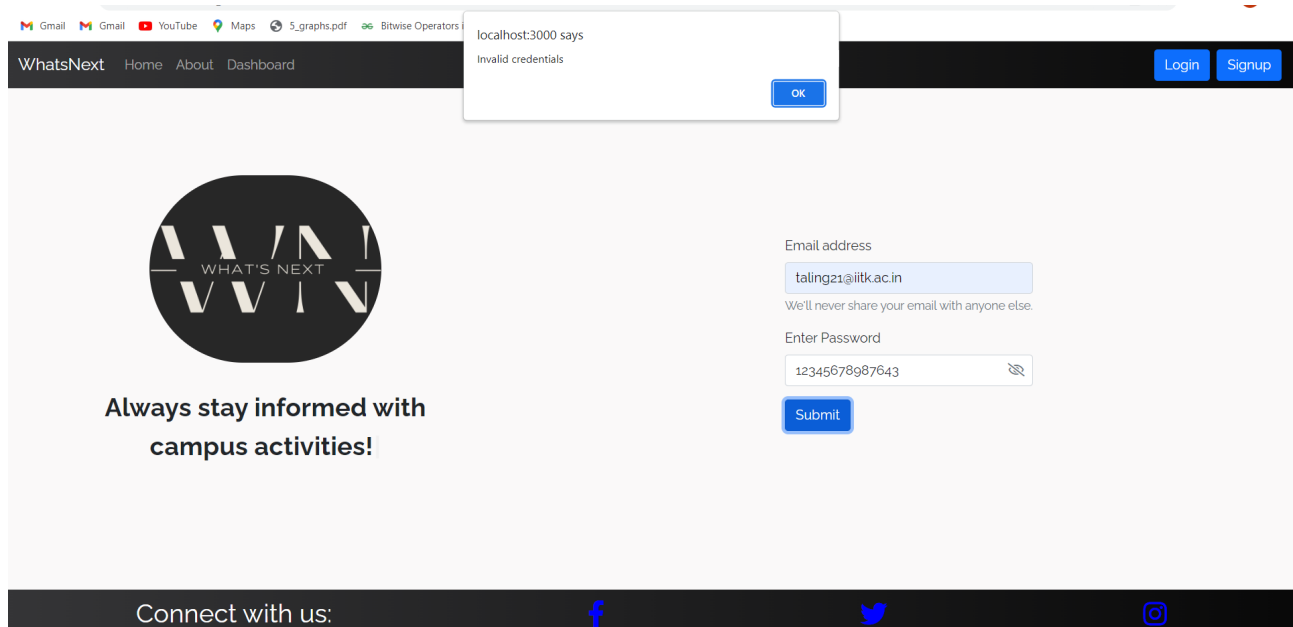
Result : redirected to Landing page

Test case : login with right credentials



Result : Taken to Landing page

Test case: In case of invalid credentials :



Result : Popup showing invalid credentials, not redirected to landing page

4. Dashboard

Module Details: Several api's were integrated with the frontend of the dashboard and they are fetchallevents, addevent, deleteevent and updateevent. The fetchallevents api was found to be correctly working as the frontend was showing the correct details of all the events added by an admin on his dashboard . Upon the integration of the frontend and backend components of dashboard every functionality was found to be working properly and that too within the consistency of the requirements.

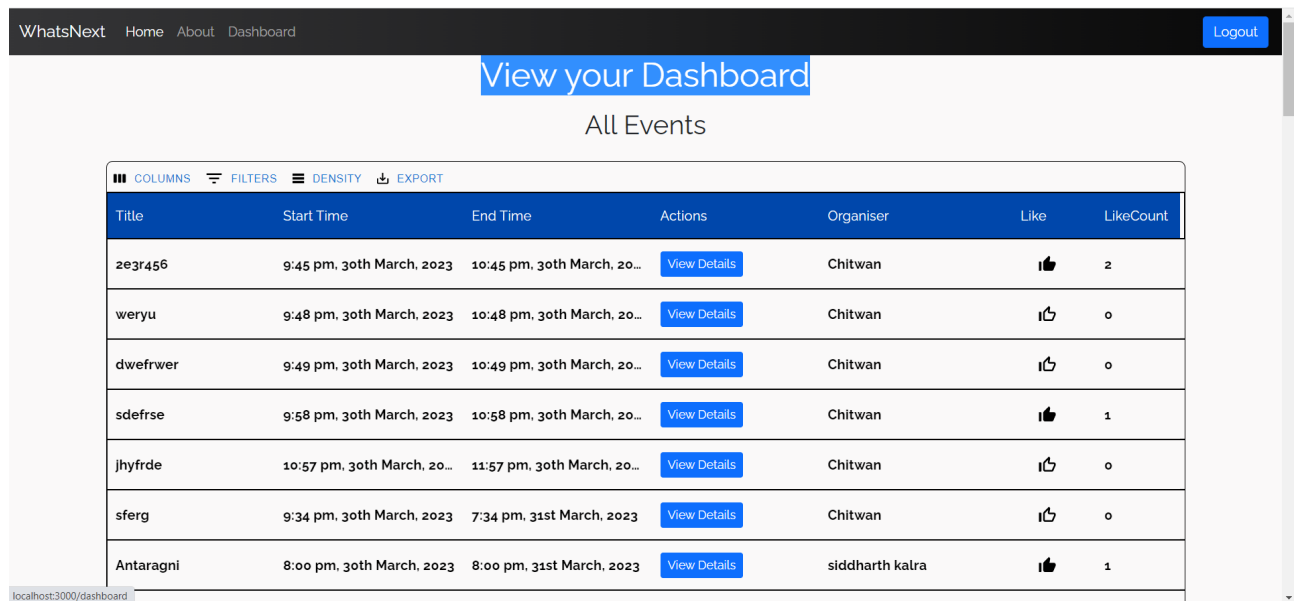
Test Owner: Talin Gupta and Siddharth Kalra

Test Date: 25/03/2023

Test Results: In this test, we were able to see that the dashboard is being

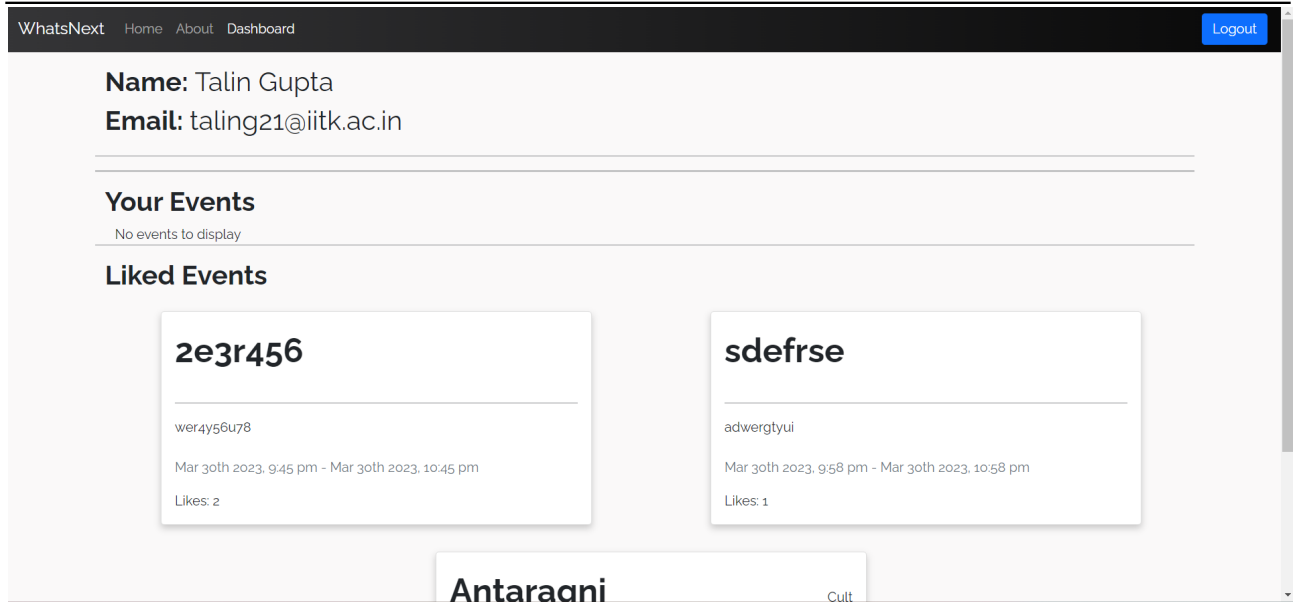
Pre-assumption : User is already logged in

Test Case : On clicking on Dashboard on the Home page we are taken to the user Dashboard.



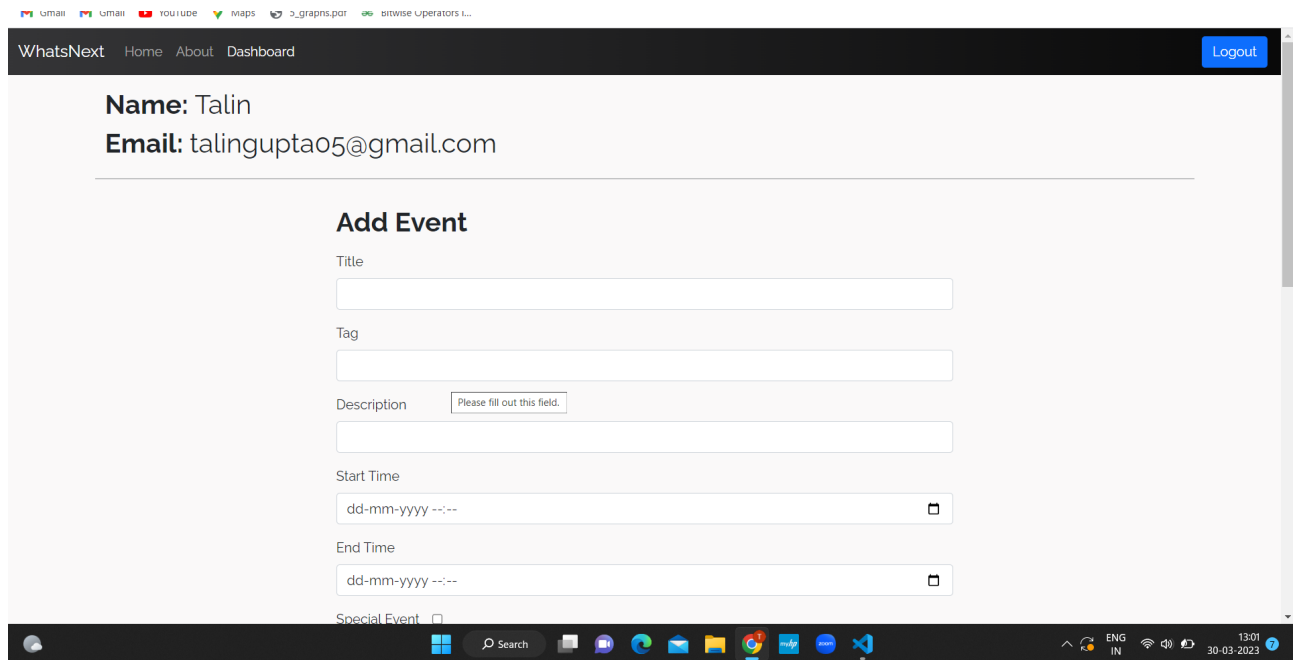
Result : Successful , dashboard appears

Test case : General user is viewing dashboard



Result : Liked events appear correctly. No option to add events. Your events section is empty as a general user can't add events.

Test case : Admin is seeing dashboard



Result : Option to add event shows up in dashboard

4. Event operations

Module Details: Here we tested the working of addition of a new event, deletion and updation of pre added events by the admin. Frontend and backend integration was tested, as well as various apis working well.

Test Owner: Kruthi and Siddharth Kalra

Test Date: 30/03/2023

Test Results:

Our website What's next allows the admins to add, edit or delete an event by going into their dashboard. While adding a new event, incomplete credentials (providing a link for the poster or image being optional) or invalid start and end times leads to not being able to add that event. Also while adding an event all the feasible clashes with respect to the time of the event will be shown to the corresponding admin using a pop up. All the events added by the user can be found at the bottom of the dashboard page, below the Add event functionality. The title, description, tag, dates, everything can be edited in an event.

1)Adding an event-

- Incomplete credentials- The 'Add events' button remains inaccessible-


Add Event

Title


Tag

Description

Start Time



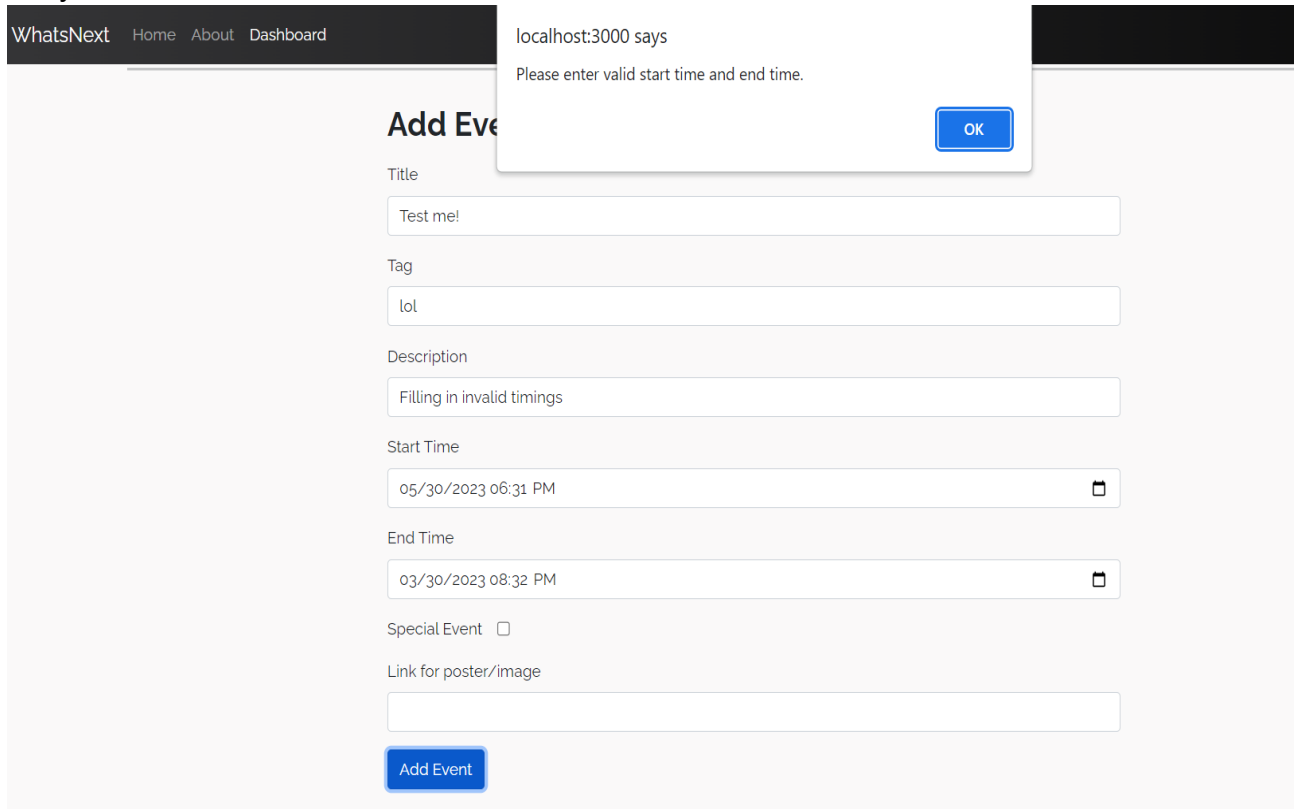
End Time



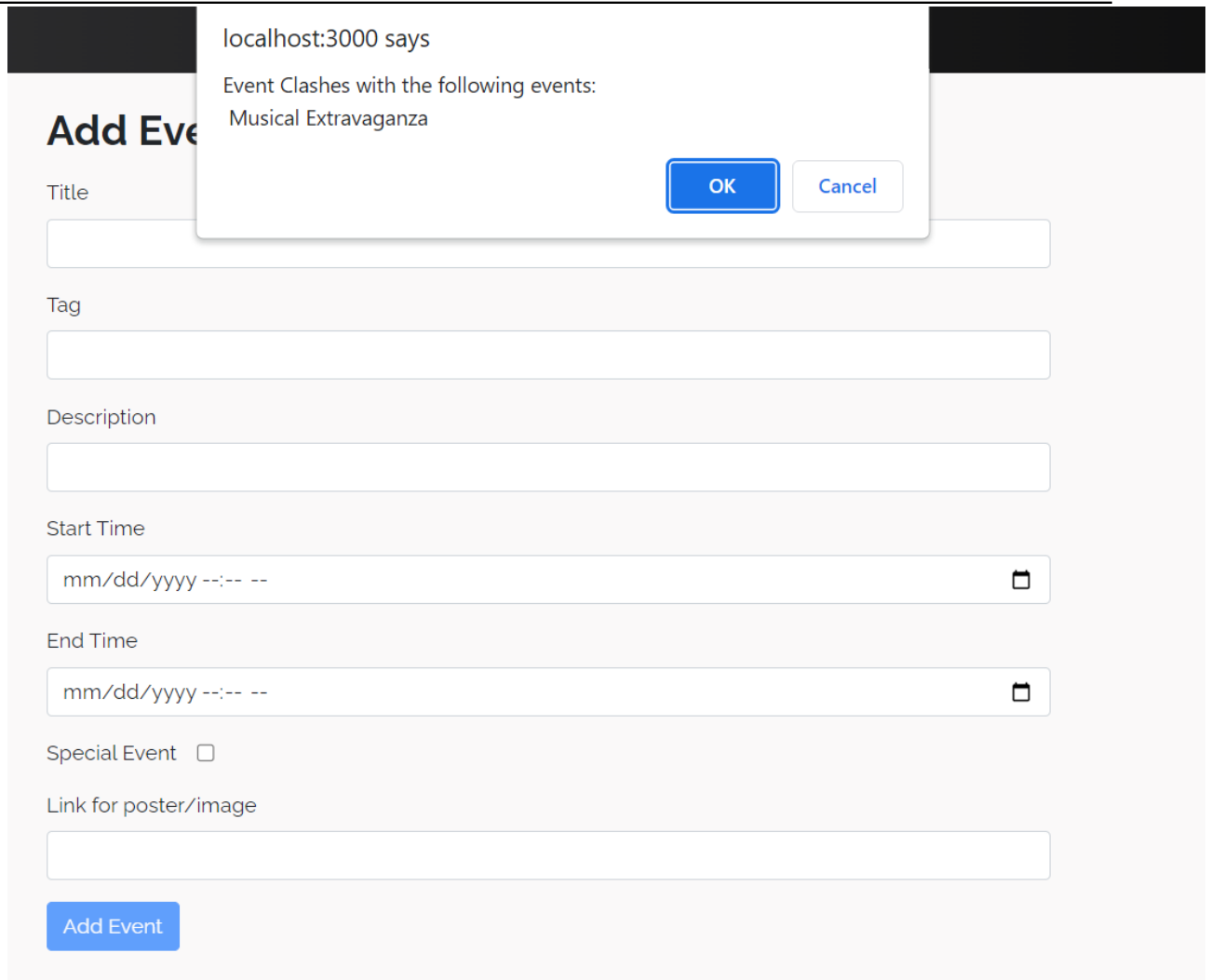
Special Event

Link for poster/image

- Test case Invalid start and end times- A pop up warning appears requesting the entry of valid time.



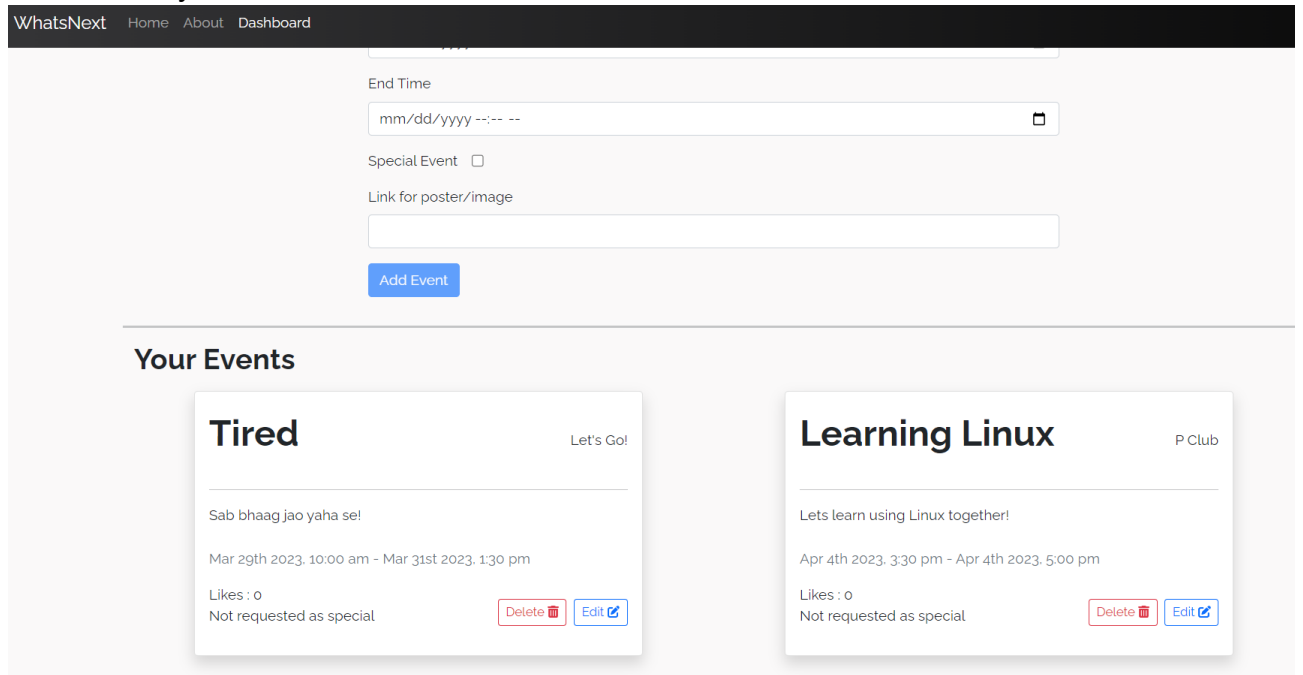
- Clash detector- A pop up appears warning the admin about the event(s) in the same date and time slot.
Clash detected-



The image shows a web form titled "Add Event" with several input fields and a modal dialog box. The form fields are: Title, Tag, Description, Start Time (with a calendar icon), End Time (with a calendar icon), Special Event (checkbox), and Link for poster/image. A blue "Add Event" button is at the bottom. The modal dialog, titled "localhost:3000 says", contains the text "Event Clashes with the following events: Musical Extravaganza" and two buttons: "OK" and "Cancel".

2)Updating an event-

- Location of 'your events'



- Now clicking the Edit button of any of the events-


Edit Event ✕

Title


Description

Tag

Start Time



End Time



- Editing the description and tag of the event-


Edit Event ✕

Title


Description

Tag

Start Time

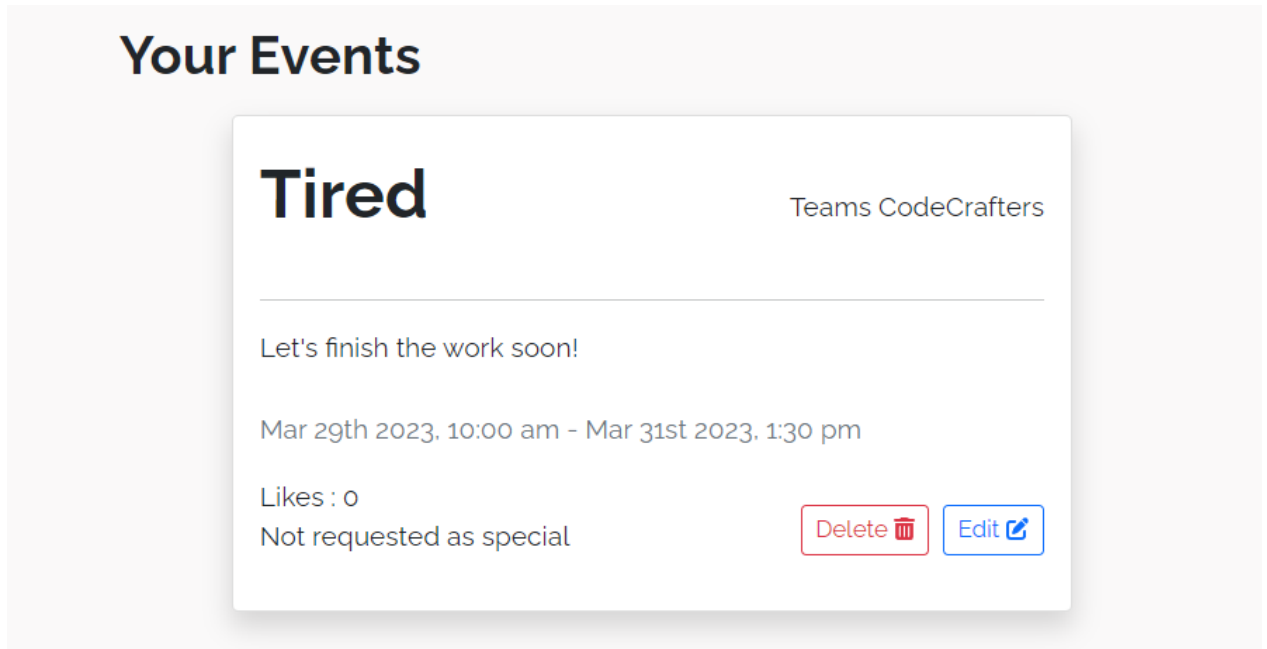


End Time



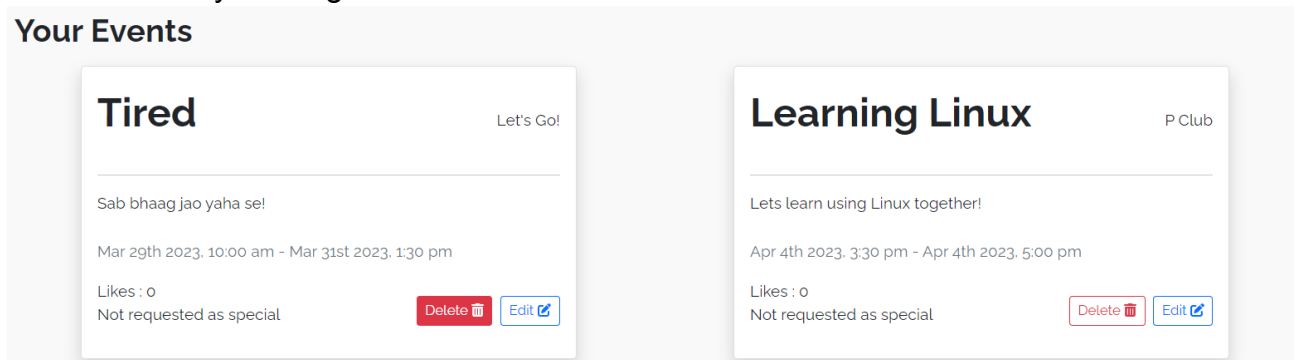
Close Update Event

- The edited event looks like-

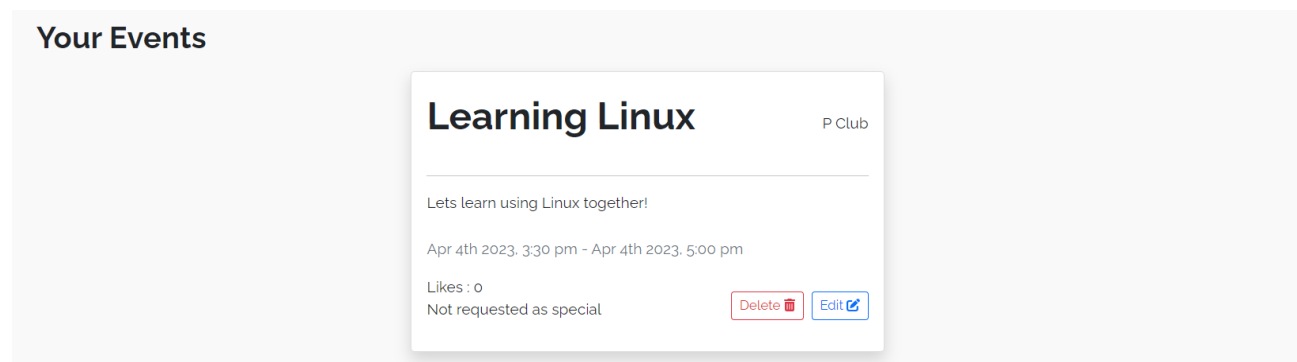


3)Deleting an event-

- Can be done by clicking the delete button on the event box-



- Now after deleting the event 'Tired', 'Your events' looks like-



5. Filter Events

Module Details: In this test we tested that the events get filtered according to the filters we apply. The scope of this testing includes filtering events based on various criteria, including title, start time, end time, actions, organizer, like and like count.

Test Owner: Varun Tokas and Apoorva Gupta

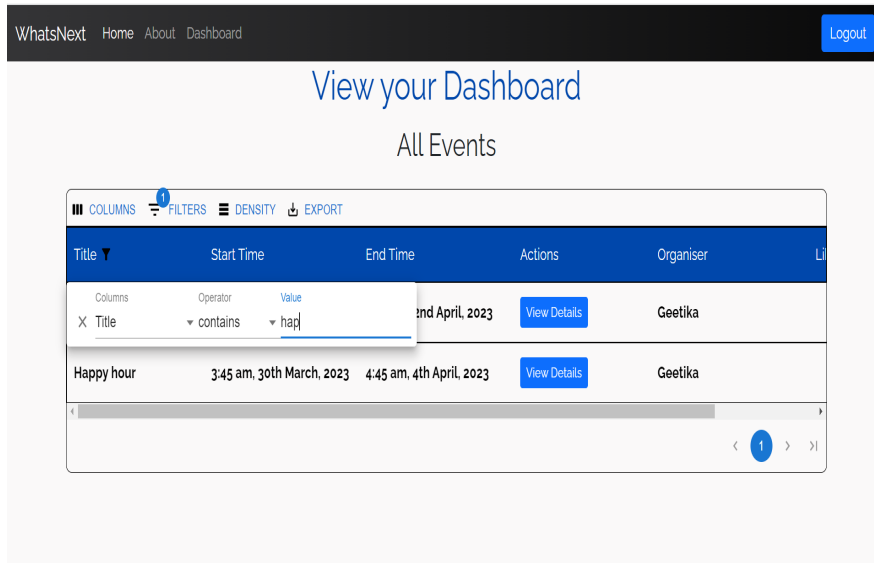
Test Date: 29/03/2023

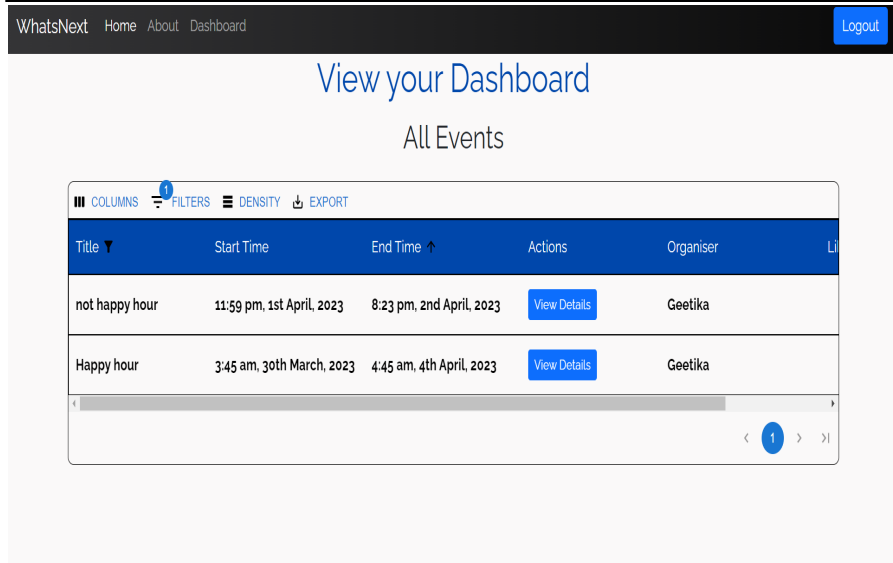
Test Case 1: Filtering events by Title

Objective: To verify that the system filters events based on the title

Input: we apply the filter Title with operator contains and value of "hap"

Expected output: we should get all events that have "hap" in their Title.





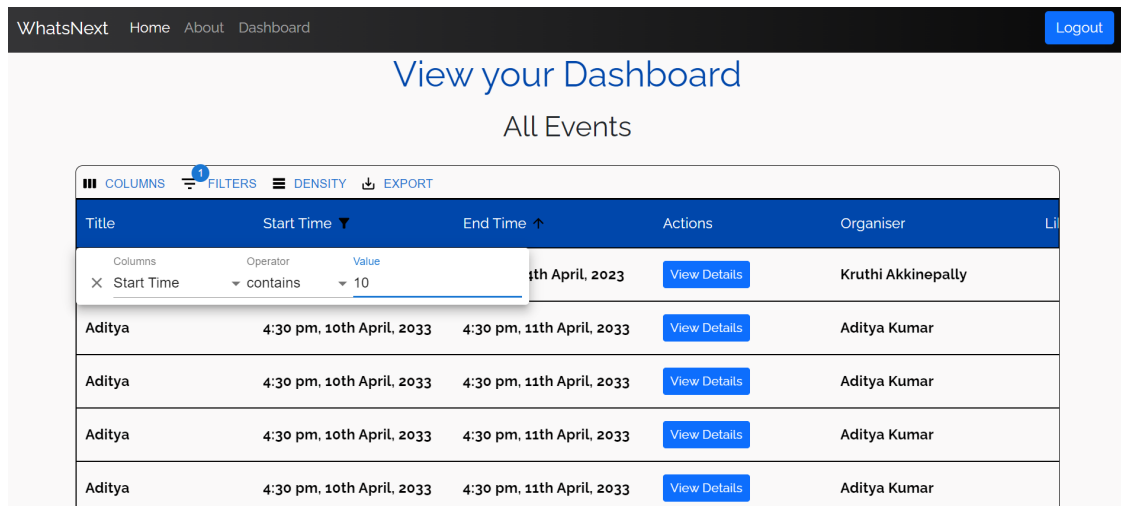
Results: we are getting all the events filtered that have “hap” in their title.

Test Case 2: Filter events by start time

Objective: To verify that the system filters events based on start time.

Input: We apply the filter start time with operator contains and value 10.

Expected Output: all the events with start time 10th April gets filtered.



WhatsNext				Home	About	Dashboard	Logout
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar			
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar			
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar			
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar			
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar			
Aditya	4:30 pm, 10th April, 2033	4:30 pm, 11th April, 2033	View Details	Aditya Kumar			
An event5	4:30 pm, 10th April, 2054	5:30 pm, 19th April, 2054	View Details	Geetika			
An event6	4:30 pm, 10th April, 2054	5:30 pm, 19th April, 2054	View Details	Geetika			
An event1	4:30 pm, 10th April, 2054	5:30 pm, 20th April, 2054	View Details	Geetika			

Results: all the events with start time 10th April gets filtered .

Bugs: a bug is found, while testing the if we give value as 10th instead of 10 , events are not filtered. Also if we give a string as input value, the events are not filtered.

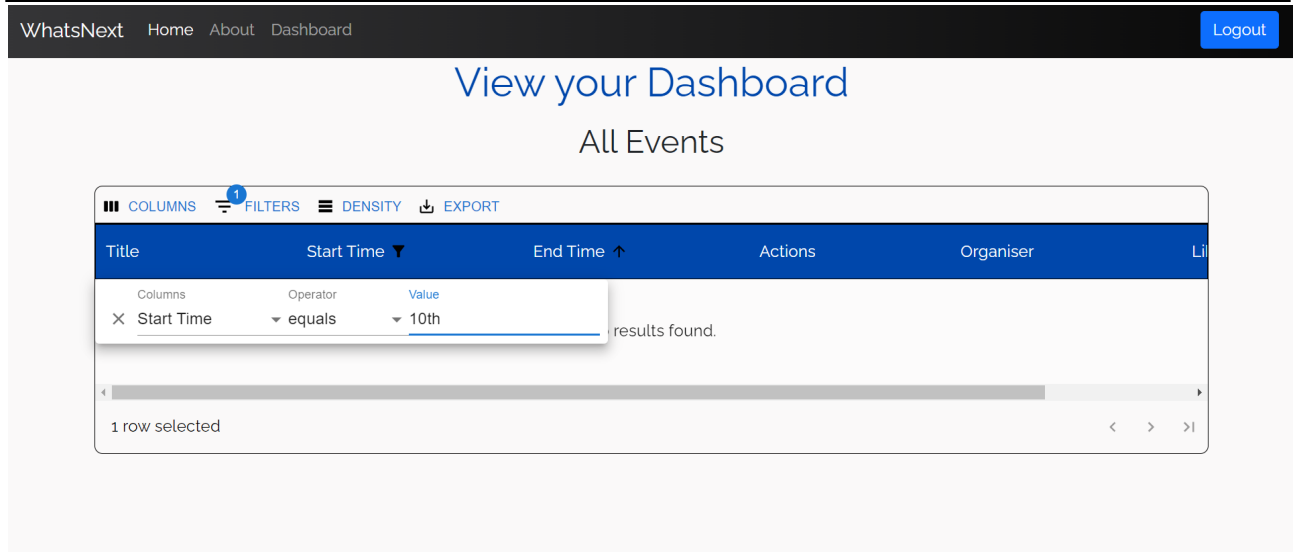
WhatsNext Home About Dashboard [Logout](#)

View your Dashboard

All Events

COLUMNS **FILTERS** **DENSITY** **EXPORT**

Title	Start Time	End Time	Actions	Organiser	Li
Columns Operator Value					
X	Start Time	▼ equals	▼ Ap	results found.	
1 row selected					

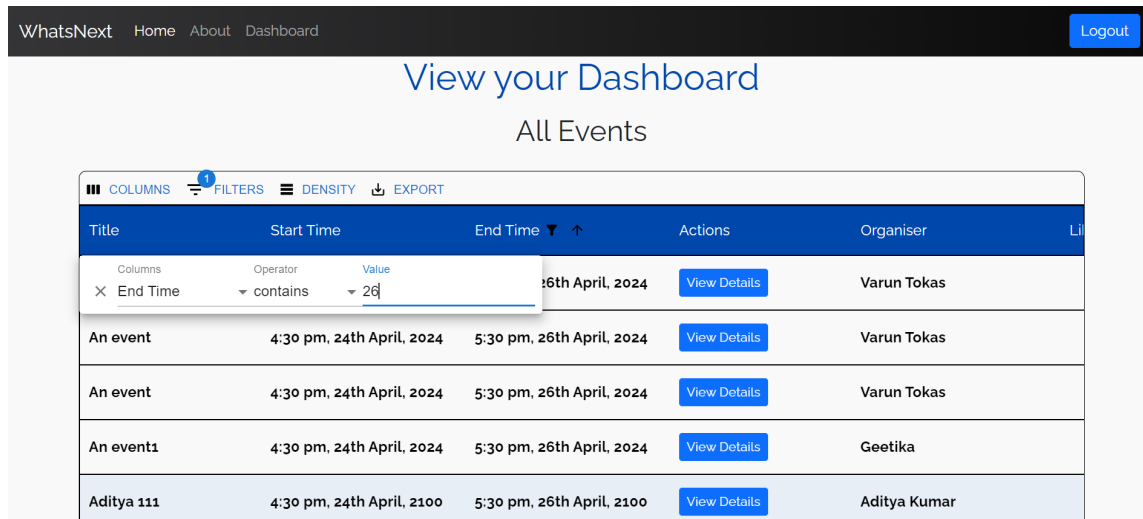


Test Case 3: Filter events by end time

Objective: To verify that the system filters events based on end time.

Input: We apply the filter end time with operator contains and value 26.

Expected Output: all the events with end time 26th April gets filtered.



WhatsNext Home About Dashboard Logout				
Aditya 112	4:30 pm, 24th April, 2101	5:30 pm, 26th April, 2101	View Details	Aditya Kumar
Aditya 114	4:30 pm, 24th April, 2102	5:30 pm, 26th April, 2102	View Details	Aditya Kumar
Aditya 114	4:30 pm, 24th April, 2102	5:30 pm, 26th April, 2102	View Details	Aditya Kumar
Aditya 114	4:30 pm, 24th April, 2102	5:30 pm, 26th April, 2102	View Details	Aditya Kumar
Aditya 1	4:30 pm, 24th April, 2204	5:30 pm, 26th April, 2204	View Details	Varun Tokas
Aditya 1	4:30 pm, 24th April, 2214	5:30 pm, 26th April, 2214	View Details	Varun Tokas
Aditya 1	4:30 pm, 24th April, 2214	5:30 pm, 26th April, 2214	View Details	Varun Tokas
Aditya 1	4:30 pm, 24th April, 2244	5:30 pm, 26th April, 2244	View Details	Varun Tokas

Results: all the events with end time 26th April gets filtered.

Bugs: a bug is found, while testing the if we give value as 26th instead of 26 , events are not filtered. Also if we give a string as input value, the events are not filtered.

WhatsNext Home About Dashboard Logout

View your Dashboard

All Events

COLUMNS FILTERS DENSITY EXPORT

Title	Start Time	End Time	Actions	Organiser	Li
Columns Operator Value					
X End Time	contains	26th	results found.		

1 row selected

WhatsNext Home About Dashboard Logout

View your Dashboard

All Events

COLUMNS FILTERS DENSITY EXPORT

Title	Start Time	End Time	Actions	Organiser	Li
Columns Operator Value					
X End Time	contains	A	results found.		

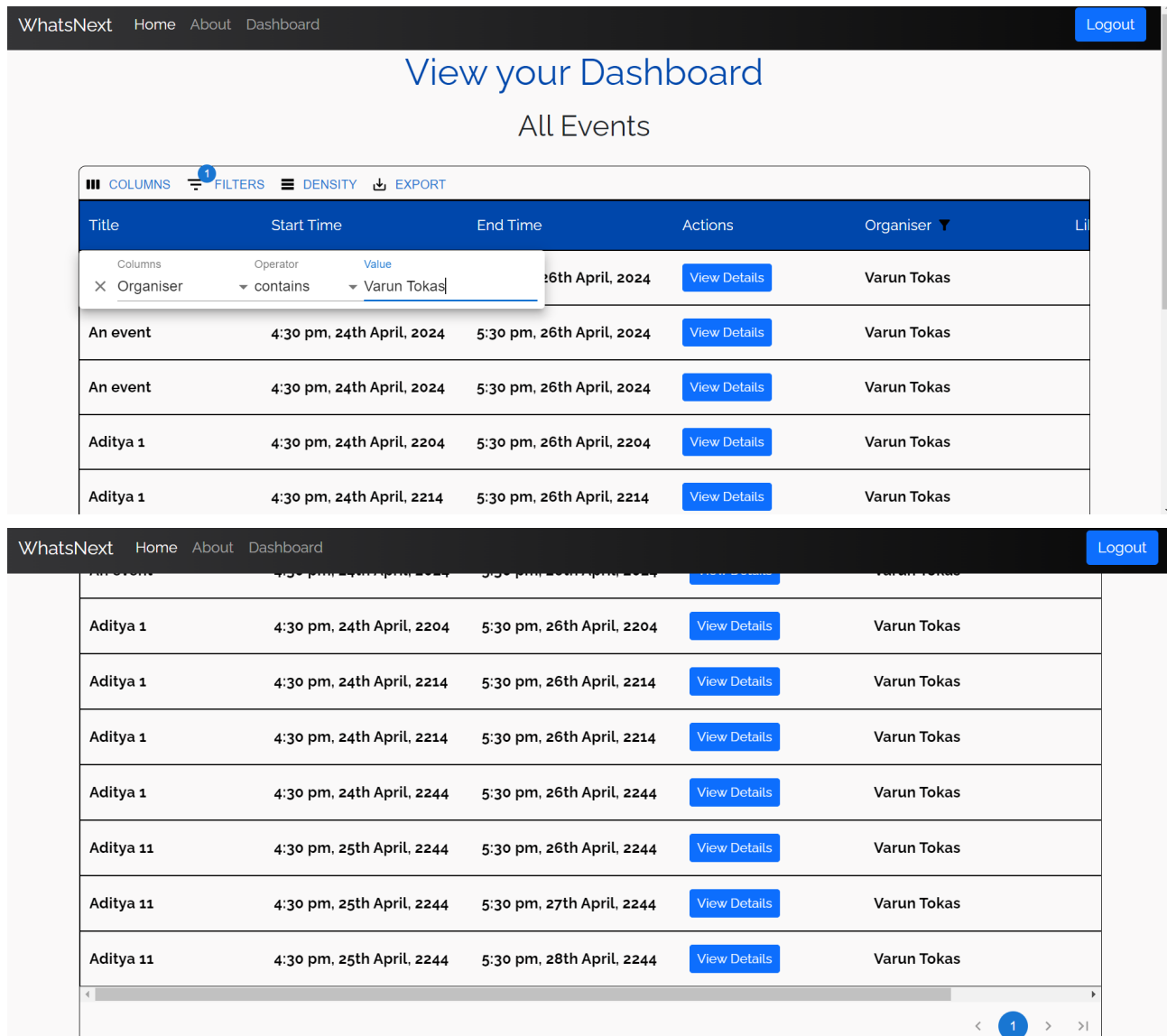
1 row selected

Test Case 4: Filter events by organizer.

Objective: To verify that the system filters events based on the organizer

Input: we apply the filter organizer with operator contains and value of "Varun Tokas"

Expected output: we should get all events that are organized by "Varun Tokas"



Results: all the events with the organizer "Varun Tokas" gets filtered.

System Testing

System testing is a type of software testing that is conducted to evaluate the complete system or software application as a whole. It is performed to ensure that the system meets the specified requirements and performs as intended. The objective of system testing is to validate and verify the software product under test and ensure that it is free from defects. The first step in defining a system testing strategy is to determine the objectives of the testing. . In this case, the objectives include ensuring that the system is functional, user-friendly, and can handle a high volume of users. The system is being tested under the “DEVELOPMENT” environment . The Scope of our system testing also defines the areas of the system that we’ll be testing. In our case it is : Process of adding events, process of approving the requests of creating a special event, process of liking an event . There are several types of testing that should be included in the system testing strategy, such as functional testing, usability testing, performance testing, security testing, and compatibility testing. It is important to ensure that all of these types of testing are conducted.

REQUIREMENTS TESTED (AT THE SYSTEM LEVEL)

1. Requirement : Adding an event from admin's side and viewing it from the perspective of an user

This involved testing both the frontend and the backend together to test whether they fulfilled the requirements. We acted as a general user and tried to like an event and view detailed descriptions of an event . All the features involved were tested from the frontend. The results were then analyzed for correctness. For this, we populated the hosted database with events data. As stated before the testing ran in parallel on multiple fronts, so the database was already populated as part of other testing routines.

Test Owner: Aman Arya and Paras Sikarwar

Test Date: 30/03/2023

Test Results: As a simulated admin, we attempted to use the functionalities as claimed in the SRS. This was done using the User Interface hosted on an actual server. The functionalities were carried out successfully.

2. Requirement : The system will help the admin to detect clashes of any sort

This test involved both the frontend and the backend as well. Multiple testers acted as admins to add events. They added an event with its description such as what is the event about and then they provided the start and end time for the event. The various test criteria used in Unit Testing - 2.3 were rigorously performed here as well owing to the large number of checks. The system successfully showed clashes if they were present in any scenario.

Test Owner: Aman Arya and Siddharth Kalra

Date: 30/03/2023

Test Results: The system was flawlessly able to detect clashes among events. Owing to rigorous unit testing during development of each and every API, almost no bugs were reported.

3. Requirement : Requesting an event as a special event

This test involved both the frontend and the backend as well. Multiple testers logged in as simulated admins. Operations such as adding an event and requesting them as a special event were performed by the simulated admins. All the implemented APIs related to the Special Event Request were tested here.

Test Owner: Aman Arya and Paras Sikarwar

Test Date: 30/03/2023

Test Results: The tests were successful. Again, owing to rigorous unit testing, the bugs found at this stage were minimal.

4. Requirement: General user can filter and sort the events according to its interest

This test involved the frontend, we acted as general users who can see the events according to their own interest. We sorted the events according to the all functionalities, like Title, start time, end time, tags, organisers, like count etc.

Test Owner: Geetika and Aditya

Test Date: 2/04/2023

Test Results: The tests were successful. Again, owing to rigorous unit testing, the bugs found at this stage were minimal.

5. Requirement: General user can view the events filtered according to the categories

This test involved the frontend, we acted as general users who can see the events according to their own interest. We filtered the events according to the all functionalities, like Title, start time, end time, tags, organisers, like count etc.

Test Owner: Krish and Apoorva Gupta

Test Date: 2/04/2023

Test Results: The tests were successful. Again, owing to rigorous unit testing, the bugs found at this stage were minimal.

6. Requirement: Like and Dislike

This test involved the frontend and backend, we acted as general users who can see the likes associated with the event. Likes will get updated when we like or dislike.

Test Owner: Geetika and Kruthi

Test Date: 2/04/2023

Test Results: The tests were successful. Again, owing to rigorous unit testing, the bugs found at this stage were minimal.

7. Non functional Requirements: This includes requirements related to how the software performs under different load conditions, response times, and scalability.

Test Owner: Geetika and Aman

Test Date: 2/04/2023

Test Results:

1. Performance

- OTP – 7sec
- Login and signup - 3 sec
- Loading of various pages such as homepage, login page, dashboard, etc -2 sec.
- Processing of requests such as adding events, deleting events, and requesting special events, etc.- 1sec

2. Scalability: The system can handle a maximum of 500-1000 concurrent users on an average depending on the server we will be using .

3. Usability: The system is very easy and user friendly to use and the website contains clear instructions and error messages to navigate through the website in a smooth manner.

4. Reliability: The system is very robust in its working and the expected down time of the website is considerably less and the maintenance is also smooth.

5. Security : Security has been taken care of.

- Password encryption is working correctly.
- Otp generation can be requested only at intervals of 30s so as to prevent spamming
- Tested successfully that superadmin page can only be accessed through a secret key
- A security bug was found : a person can directly call createuser api and register himself on the site, without making use of an otp. This is not possible from the frontend but can be done from backend. Will be corrected in future versions

Conclusion

4.1 Effectiveness and Exhaustiveness of Testing

The web application involved a large number of APIs, where each was responsible for fulfilling a particular requirement. Each of these underwent testing to ensure that they were modifying the database and returning the information when asked to do so. Initial testing allowed the developers to uncover vulnerabilities and bugs and many of them were rectified. This process of finding bugs using testing and then improving them ensured that the application completes the expectations in the SRS. The application's key features and services have undergone extensive testing and work robustly providing a pleasant experience to the user.

4.2 Inadequate Testing Components

As far as the stress testing and testing of some functional requirements were concerned we did not have adequate resources for carrying out that testing. For example, to stress test the software, we did not have adequate resources for carrying out that testing as we required many devices and signups for them.

1. Multiple User Registration simultaneously
2. Multiple Add events

These are some of the inadequacies that we faced while testing.

4.3 Difficulties Faced

The main difficulties that we faced while testing was the process of simultaneously developing and testing the software. As we were also improving the software simultaneously, few test cases were needed to be performed again.

4.4 Improvement in Testing Process

The testing process can be improved by introducing automated testing to make the process more streamlined. Independent testers can also be used for black-box testing.

4.5 Areas requiring improvements

- The security concern that createuser api can be called directly.
- On liking, any applied filters get removed
- Multiple users simultaneously liking an event : need manual reload to see other's likes
- Notifications, removing old events, frontend improvements have been made for the beta version , so we will add those during the Beta Testing phase

Appendix A - Group Log

Date	Timings	Duration	Minutes
23/03/23	9am-10:30am	1.5hrs	<ul style="list-style-type: none"> ● Meet to discuss what to be done and written in the testing doc and user manual. ● Also revisited the implementation doc to ensure everybody was on the same page.
24/03/23	6pm-10pm	4hrs	<ul style="list-style-type: none"> ● All the team members assembled to do the unit testing together. ● All the discussion about how to proceed with the unit testing, and each of its components was done. ● Each of the parts for the unit testing was done and the unit testing part in the testing doc was edited.
25/03/23	2pm-5pm	3hrs	<ul style="list-style-type: none"> ● A meet to finish unit testing. ● Also started integration testing. ● Work distribution for the user manual doc.
29/03/23	8pm-12pm	4hrs	<ul style="list-style-type: none"> ● Meet to continue the integration testing, and a few other parts of the doc like introduction etc. ● Also discussed and did a small part of the system testing. ● Also started writing the user manual.
30/03/23	5:30pm-9pm	3.5hrs	<ul style="list-style-type: none"> ● The meeting was held to finish the integration testing and started with system testing. ● Also continued writing of the user manual.
31/03/23	5:30pm-7pm	1.5hrs	<ul style="list-style-type: none"> ● Finished and submitted the user manual.
02/04/23	4pm-7pm	3hrs	<ul style="list-style-type: none"> ● Meet held to finish the Testing doc. ● Also revisited and discussed about all the components of the testing doc to make sure everybody was on the same page.