Events Frontend

What is an event?

In Sportify, every user can create or join a sport “event”. Each user will be able to create an event with the following details:

1. Title
2. Description of the event
3. Sport
4. Date and time of the event
5. Location name
6. Number of players who can join the event

Event Location

In the create event form, the user should be able to enter the location in a search field, and be able to select the location from the google maps api results. As the user is typing, google maps api will be called to fetch results, and the user should be able to select one.

This is then sent to the backend where it'll be stored as location name, latitude and longitude. when a GET call is made to the backend, it will return this latitude and longitude and frontend has to reverse geocode using gmaps api to get the location from the latitude and longitude and display to the user.

References:

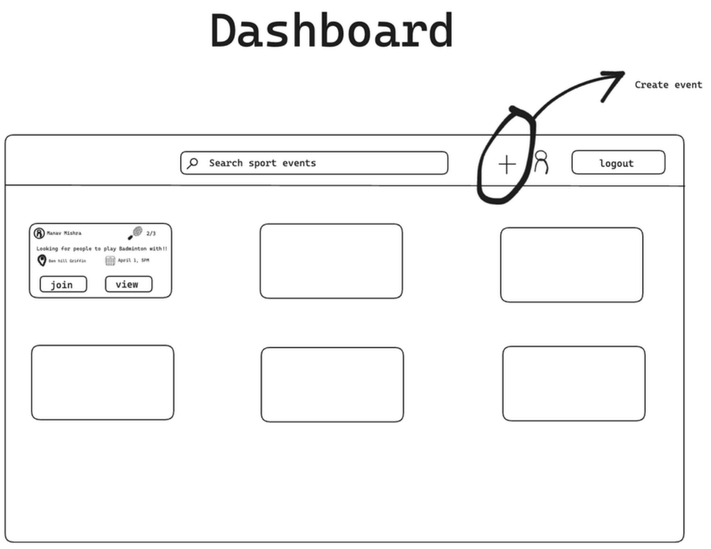
Place Autocomplete | Maps JavaScript API | Google for Developers

How to add Google Maps Autocomplete search box?

Dashboard

In the dashboard, the user will be able to see all the events, which will later on be optimised to show events closer to the user. For now, the user will see all the events, in a small viewing box, with the title, sport logo, date, time and number of spots filled for the event.

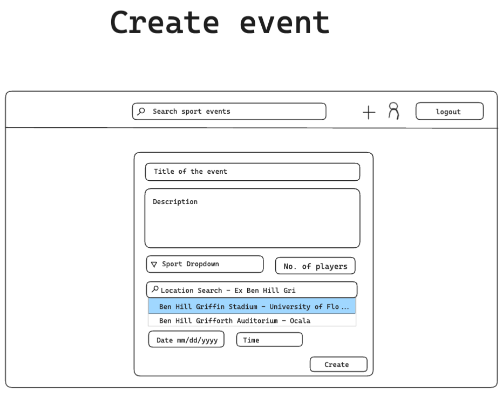
Rough UI:



Event Create

Beside the profile button, a “+” (plus) symbol signifies the create event button. The user can click it to create their own sport event. Once the button is clicked, the user will be shown a form, with the details of the sport event they are creating. The user can add the details and click “Create”.

Rough UI:



Once the user clicks “create”, the frontend makes a POST /event call to the backend with the following headers and parameters:

URL: /events

Method: POST

headers:

```
'Content-Type': 'application/json',
'Authorization': `Bearer ${localStorage.getItem('jwt_token')}`
```

parameters:

```
{
  "sport": "Football",
  "event_date": "2025-03-25T15:00:00",
  "max_players": 20,
  "location_name": "Central Park, NY",
  "latitude": 40.785091,
  "longitude": -73.968285,
  "description": "Friendly football match",
  "title": "Need players for football match"
}
```

If the backend responds with 201, it means that the event creation is successful.

Events Get - Dashboard

URL: `/events`

Method: `GET`

```
{
  sport: "Football" #(optional)
  event_date: "2025-03-25T15:00:00" #(optional)
  max_players: 3 #(optional)
}
```

For the dashboard, no filters (parameters) are necessary. Make a call to GET /events to fetch all events and display it on the dashboard.

Event View - Get Single Event

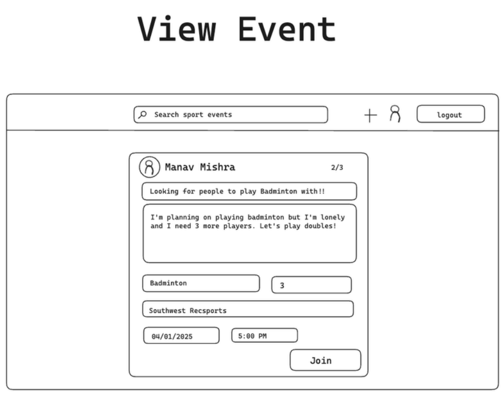
URL: `/events/:id`

Method: `GET`

```
GET /events/1
```

When a user wants to view an event from the dashboard or a user profile, once they click the “view” button, frontend must make a call to the backend to fetch the details of the event with the event id.

Rough UI:



Manav Mishra 2/3

Looking for people to play Badminton with!!

I'm planning on playing badminton but I'm lonely and I need 3 more players. Let's play doubles!

Badminton

3

Southwest Recsports

04/01/2025

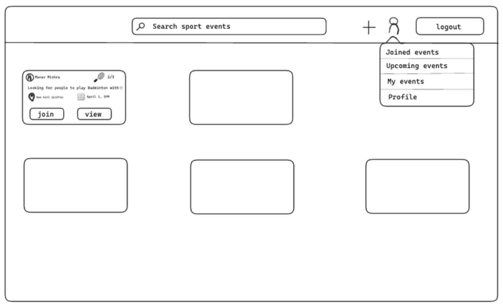
5:00 PM

Event View - Own and Joined Events

A user can click on a profile, or their own profile to view all events a user (or themselves) are a part of. For the owner, when the profile button is clicked, 4 options are shown

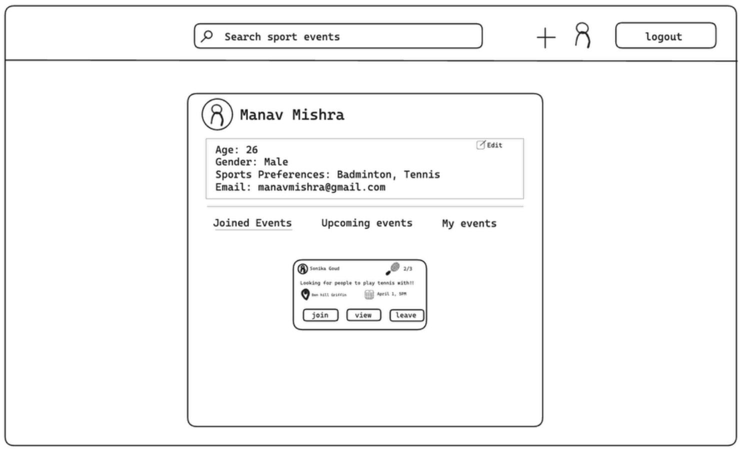
1. Joined events - All events that they have joined so far
2. Upcoming events - Events that have not yet taken place, but have joined
3. My events - Events that the user created
4. Profile - User details

User Profile and Events

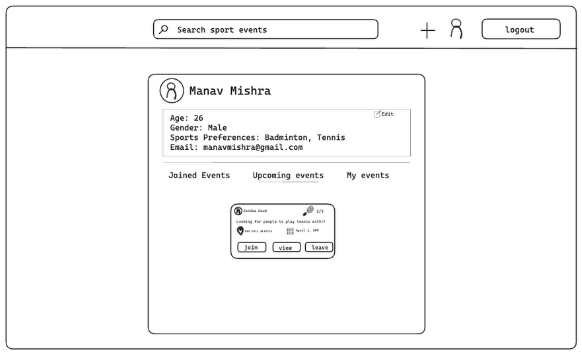


If a user clicks their own profile logo (as shown above) and selects any of the 4 options.

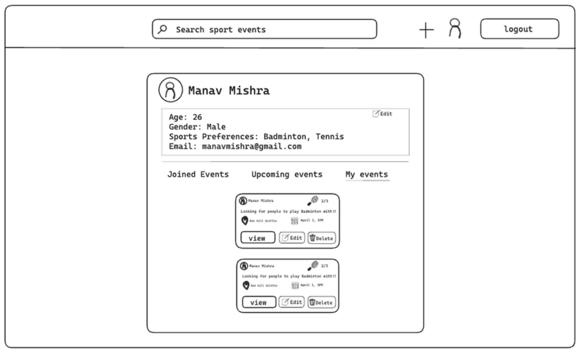
Rough UI:



Age: 26
Gender: Male
Sports Preferences: Badminton, Tennis
Email: manavmishra@gmail.com



Age: 26
Gender: Male
Sports Preferences: Badminton, Tennis
Email: manavmishra@gmail.com



Age: 26
Gender: Male
Sports Preferences: Badminton, Tennis
Email: manavmishra@gmail.com

If a user X clicks on user Y’s Profile, they will be shown the Y’s name, profile details and the events associated with Y. To get joined events and created events of a user, this is the call:

URL: `/events`

Method: `GET`

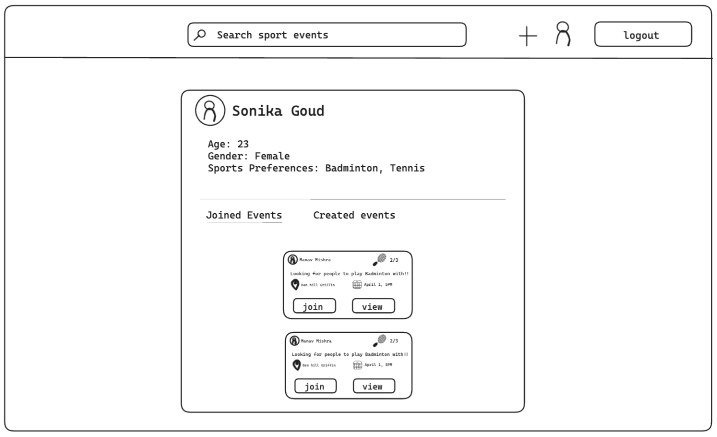
GET created events: Events Frontend | Events Get Dashboard

```
{
  event_owner_id: 1
}
```

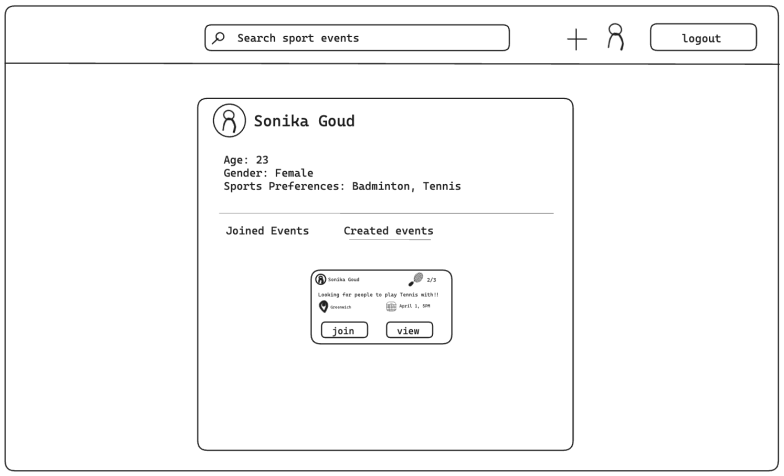
GET Joined events:

TBD

Rough UI:



Age: 23
Gender: Female
Sports Preferences: Badminton, Tennis



Event Update:

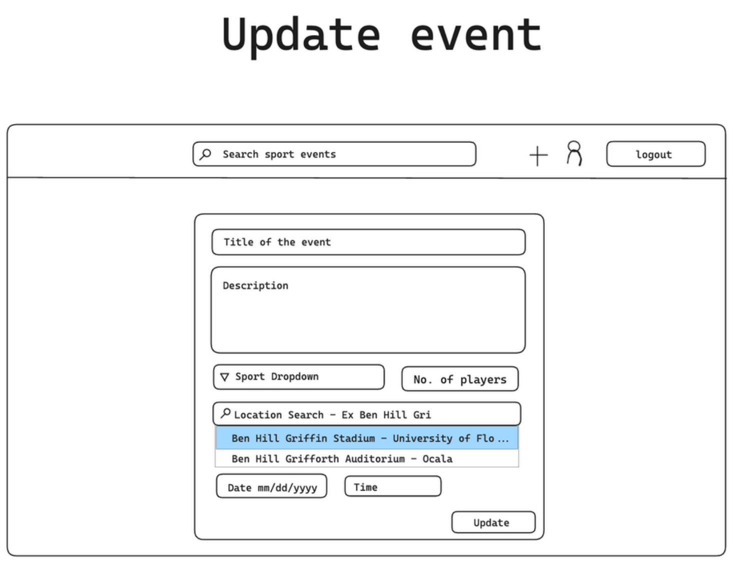
A user can update an event they created. Frontend makes a PUT call to the backend. API spec is:

URL: `/events/:id`

Method: `PUT`

```
{
  "sport": "Basketball",
  "event_date": "2025-03-26T15:00:00",
  "max_players": 25,
  "location_name": "Madison Square Garden",
  "latitude": 40.748817,
  "longitude": -73.985428,
  "description": "Basketball tournament"
}
```

Rough UI:



Join Event

When the user clicks “Join” button on the event, frontend makes a POST call to the backend. API spec:

URL: `/events/:id/join`

Method: `POST`

Leave Event

When the event owner clicks “delete” button on the event, frontend makes a DELETE call to the backend. API spec:

URL: `/events/:id/`

Method: `DELETE`

Delete Event

When the user wants to delete an event, frontend makes a DELETE call to the backend. API spec:

URL: `/events/:id/delete`

Method: `DELETE`