NOTES :

- **FUNZOA** is a website which maps fun places all over the map of India.
- The website consists of a welcoming home page which has a button of viewing all the fun places which are already present in the database .
- Website allows the unsigned user to view all the places and see the location on the map .
- User can create an account and use the site with the respective login.
- User with an account is allowed to access numerous other features.
- For **authentication** we have a login page -User login with username and password.

AUTHORIZATION IN OUR WEBSITE

- One cannot manage posts and view user profile without being authenticated.
- One cannot edit or delete posts and comments created by other users.
- A registered user can:

1. Make a new campground , Upload campground photos.
2. Display campground location on the Map that we have required from Map-box.
3. Can give the review on the campground created by others and rate it with stars .
4. If the user himself is the author of a particular campground or a review he can do all the **CRUD functionalities** with that particular campground or a review.

DATABASE USED : MongoDB

SCHEMAS MADE :

IMAGE SCHEMA :

```
const ImageSchema = new Schema({
    url: String,
    filename: String
});
```

CAMPGROUND SCHEMA:

```
const CampgroundSchema = new Schema({
    title: String,
    images: [ImageSchema],
    geometry: {
        type: {
            type: String,
            enum: ['Point'],
            required: true
        },
        coordinates: {
            type: [Number],
            required: true
        }
    },
    price: Number,
    description: String,
    location: String,
    author: {
        type: Schema.Types.ObjectId,
        ref: 'User'
    },
    reviews: [
        {
            type: Schema.Types.ObjectId,
            ref: 'Review'
        }
    ]
}, opts);
```

REVIEWS SCHEMA :

```
const reviewSchema = new Schema({
    body: String,
    rating: Number,
    author: {
        type: Schema.Types.ObjectId,
        ref: 'User'
    }
});
```

USER SCHEMA :

```
const UserSchema = new Schema({
    email: {
        type: String,
        required: true,
        unique: true
    }
});

UserSchema.plugin(passportLocalMongoose);
```

DATABASE USED FOR LOCATING FUNPLACES :

1. Database consists of 406 prominent cities in India which is the subset of World Cities Database which was released under MIT license.

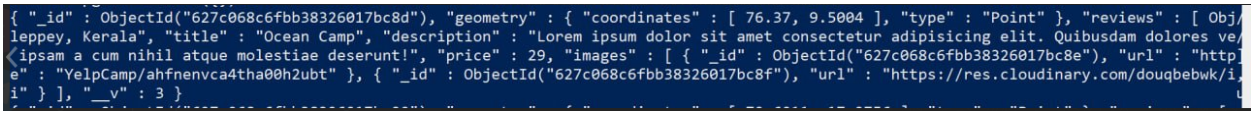
Following is the snipe of the database which we have used :

city	lat	lng	country	iso2	admin_name	capital	population	population_proper
Delhi	28.6600	77.2300	India	IN	Delhi	admin	29617000	16753235
Mumbai	18.9667	72.8333	India	IN	Mahārāshtra	admin	23355000	12478447
Kolkāta	22.5411	88.3378	India	IN	West Bengal	admin	17560000	4496694
Bangalore	12.9699	77.5980	India	IN	Karnātaka	admin	13707000	8443675
Chennai	13.0825	80.2750	India	IN	Tamil Nādu	admin	11324000	6727000
Hyderābād	17.3667	78.4667	India	IN	Telangana	admin	9746000	6993262
Pune	18.5196	73.8553	India	IN	Mahārāshtra		7764000	3124458
Ahmedabad	23.0300	72.5800	India	IN	Gujarāt	minor	7410000	5570585
Sūrat	21.1700	72.8300	India	IN	Gujarāt		5807000	4466826
Lucknow	26.8470	80.9470	India	IN	Uttar Pradesh	admin	3382000	3382000
Jaipur	26.9167	75.8667	India	IN	Rājasthān	admin	3073350	3073350
Cawnpore	26.4725	80.3311	India	IN	Uttar Pradesh		2701324	2701324

2. For storing images we are using **CLOUDINARY**

Cloudniary provides cloud-based image and video management services. It enables users to upload, store, manage, manipulate, and deliver images and video.

INFORMATION OF ALL THE PLACES AND ITS REVIEWS MADE BY THE USERS ARE GETTING ADDED IN MONGO DATABASE:



INFORMATION OF REGISTERED USER TAKEN BY THE FORM IS STORED IN DATABASE :

We have added query to search the campground : here we have written codes in our files which will search a key among all the campgrounds and provide the result of all the campgrounds which contain that key.

```
const Campground = require('../models/campground');
router.get("/", function(req, res){
    var noMatch = null;
    if(req.query.search) {
        const regex = new RegExp(escapeRegex(req.query.search), 'gi');
        // Get all campgrounds from DB
        Campground.find({title: regex}, function(err, allCampgrounds){
           if(err){
               console.log(err);
           } else {
              if(allCampgrounds.length < 1) {
                  noMatch = "No campgrounds match that query, please try again.";
              }
              res.render("campgrounds/index",{campgrounds:allCampgrounds, noMatch: noMatch});
           }
        });
    } else {
        // Get all campgrounds from DB
        Campground.find({}, function(err, allCampgrounds){
           if(err){
               console.log(err);
           } else {
              res.render("campgrounds/index",{campgrounds:allCampgrounds, noMatch: noMatch});
           }
        });
    }
});
```