Collection	**Endpoint**	**CRUD**	**Comments**

/movies	`movies/filmingLocations/:cityName`
	`movies/imageURL/:title`
	`movies/director/:directorId`
	`movies/genre/:genreId`
	`movies/:title`
		Get	Returns JSON of all movies qualified by URL parameter

/movies	`/movie` `/movies/:movieId`	Post	1) Add movie 2) Add filming location
/movies	`/movies/:movieId`	Put	Update movie info from body
/movies	`/movies/:movieId`	Delete	Remove filming location
/users	`/users`	Get	Return JSON
/users	`/users` `/users/:name/movies/:MovieID`	Post	1) Add user 2) Add to “favorite movies”
/users	`/users/:id`	Put	Update user from body
/users	`/users/:id` `/users/:name/movies/:MovieID`	Delete	1) Delete user 2) Remove from “favorite movies”
/directors	`/directors/:name`	Get	Return director by name
/genres	`/genres/:name`	Get	Return genre by name

```
let movieSchema = mongoose.Schema({
  title: {type: String, required: true},
  filmingLocations: [
      {
       name: String,
       locations: [ {
         name: String,
         location: [String, String]
       }
       ]

  }
  ],
  genre:{
      type: mongoose.Schema.Types.ObjectId, ref: 'genre'
    },
  director: {
    type: mongoose.Schema.Types.ObjectId, ref: 'Movie'
  },
  ImageURL: String,
  Featured: Boolean
});

let userSchema = mongoose.Schema({
  name: {type: String, required: true},
  password: {type: String, required: true},
  email: {type: String, required: true},
  birthday: Date,
  favoriteMovies: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Movie' }]
});

let genreSchema = mongoose.Schema({
  name: String,
  description: String
})
let directorSchema = mongoose.Schema({
  name: String,
  about: String
})
```