**ROHIT JYOTIBA CHOUGULE**

**Student Number: 19200240**

**Cloud Computing Project**

**Online Event Organizer**

**Event Management**

The Event Management is a small web application which will allow a closed group of users to create, edit, delete, archive and share events.

The small web application for Event management has been developed using the Django web development framework. Django is a high-level Python Web framework for web application development.

I have used Django framework version 2.2.7

Python version 3.6 has been used in this project.

Being a small web application to be used with a small group of users, SQLite Database would be a suitable for such an application which is supposed to be used with a closed group of users:

The lite in SQLite means light weight in terms of setup, database administration, and required resource.

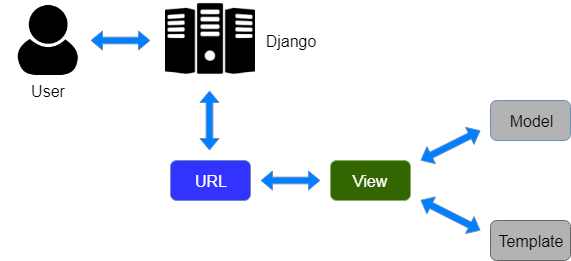
SQLite has the following noticeable features: self-contained, serverless, zero-configuration, transactional.

The web application can be deployed on an AWS EC2 instance.

The web application consists of 5 basic blocks in the event management lifecycle:

1. Create a new Event/Add a new Event.
2. Edit and delete posted event
3. Archive old events
4. Search events in the application
5. Share the event with users.

Django is a web application framework that follows the MVC-MVT pattern:



The Event Management web application has the default *event\_management* app which has then plugged in with features for search and other operations like Edit/Delete/Archive/Share.

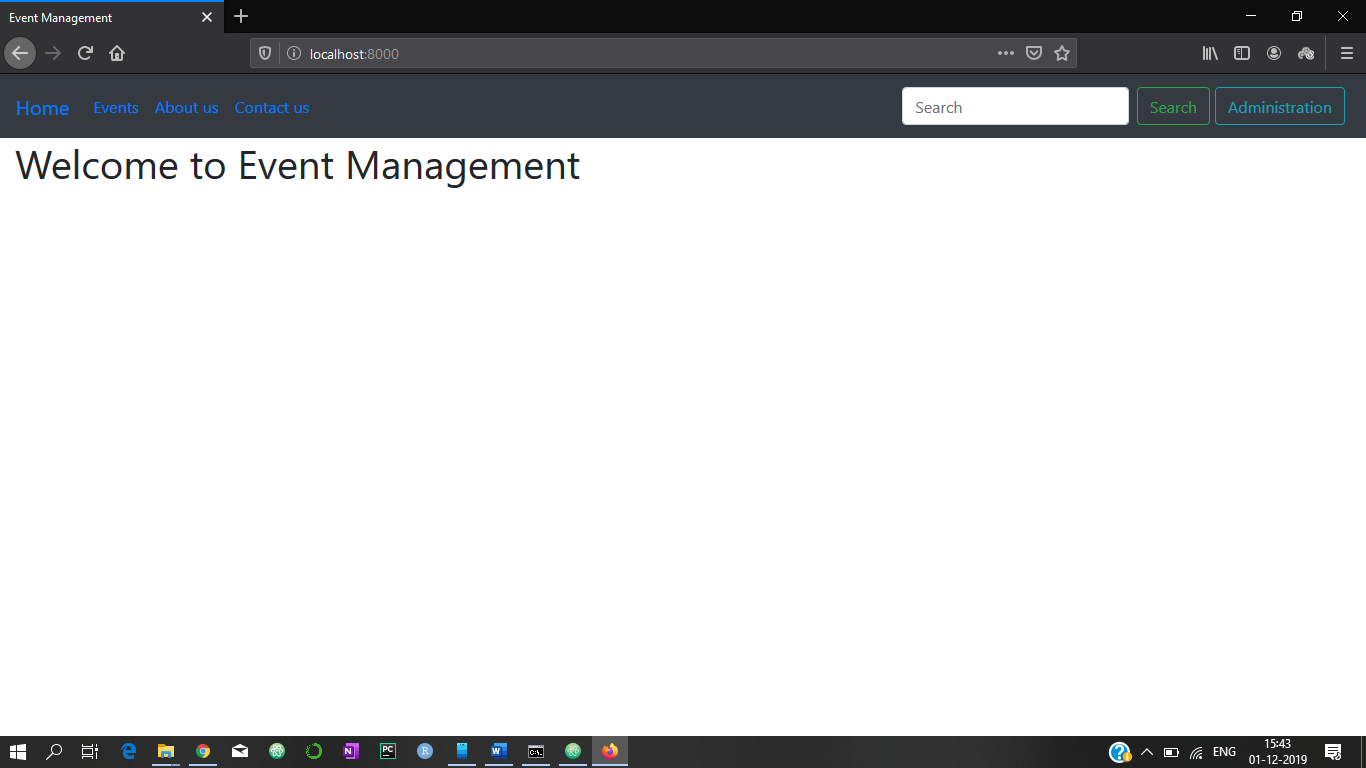
The app plugin- *event* has been written to implement the Edit, Delete, Archive and Share feature in the web application.

The app plugin- *search* has been written to implement the search functionality in the web application.

Various functions implemented in the web application has been captured in the below screenshots:

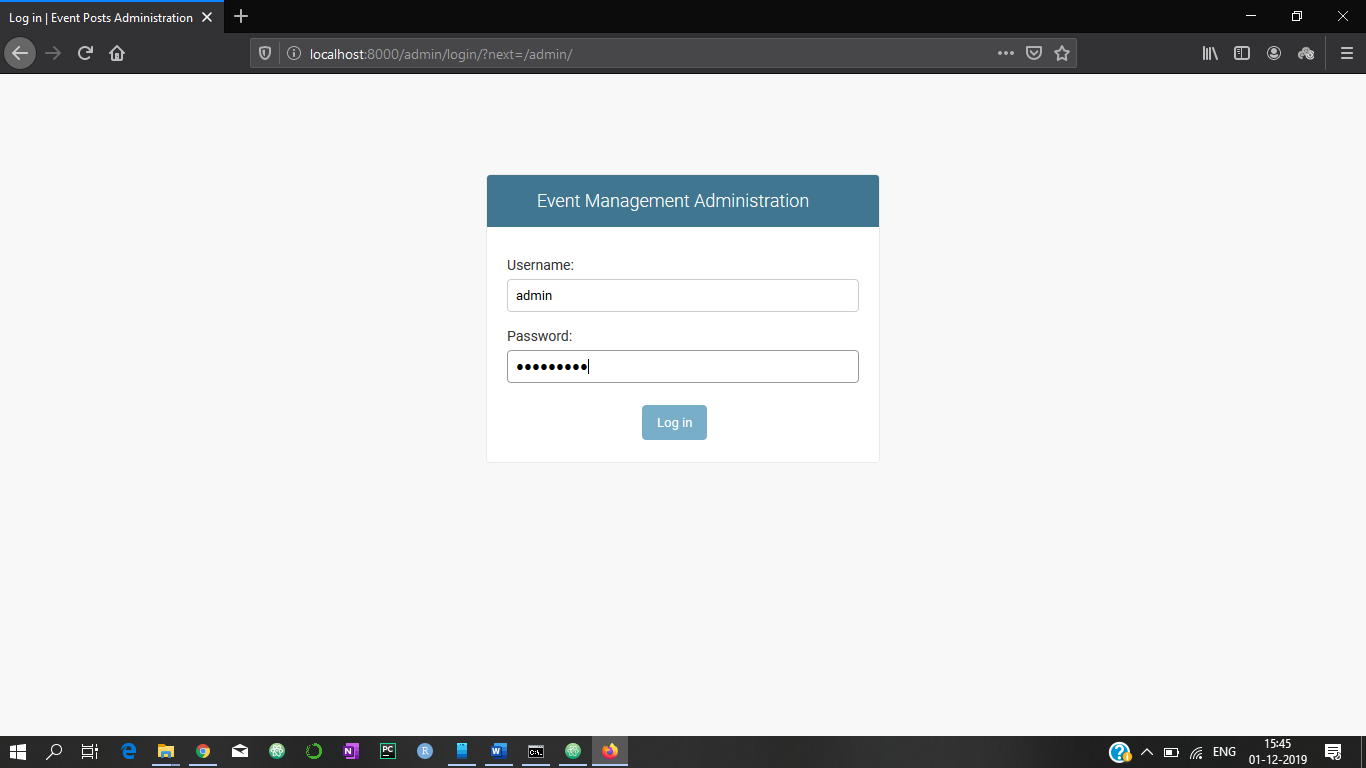
Page flow: *home\_page(event\_management) - >base.html*

1. The home page for the web app when there are no events added:

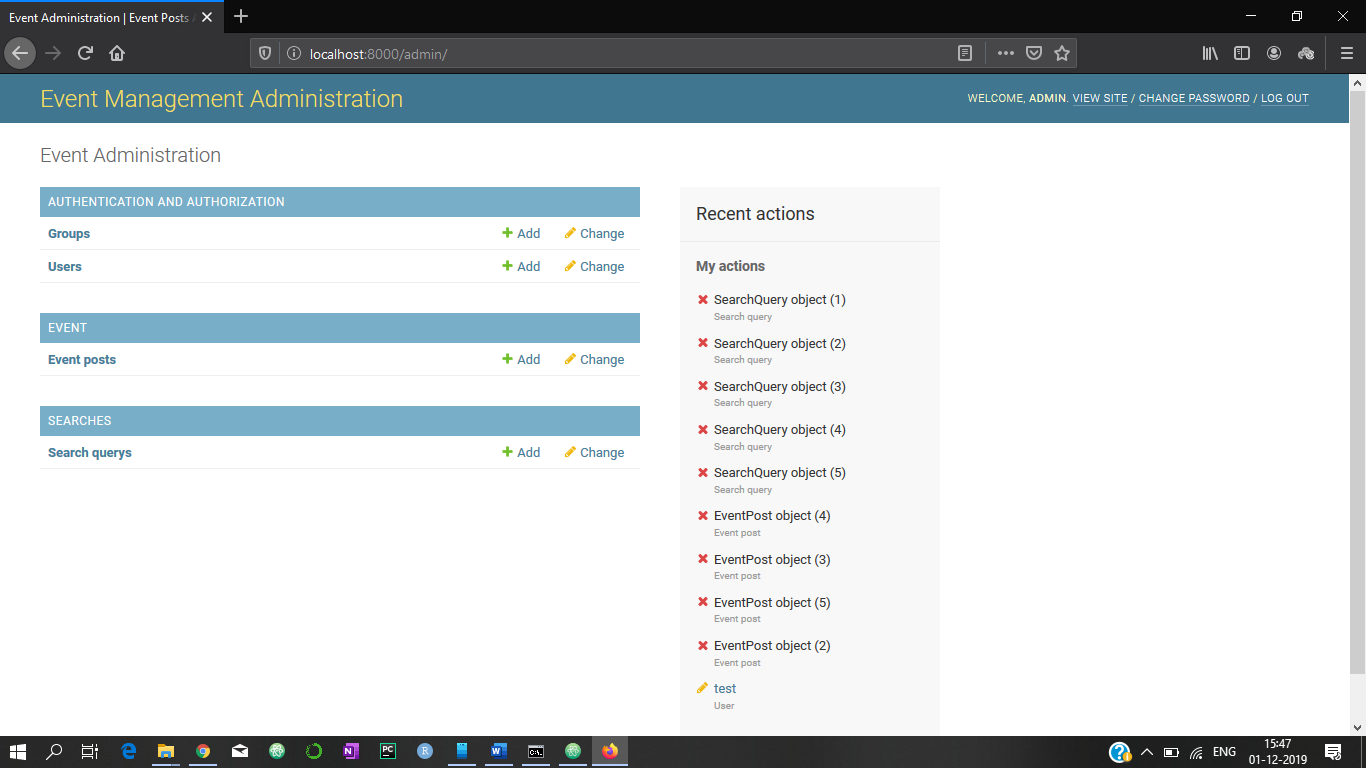


1. Event Management Administration panel to Create new events:

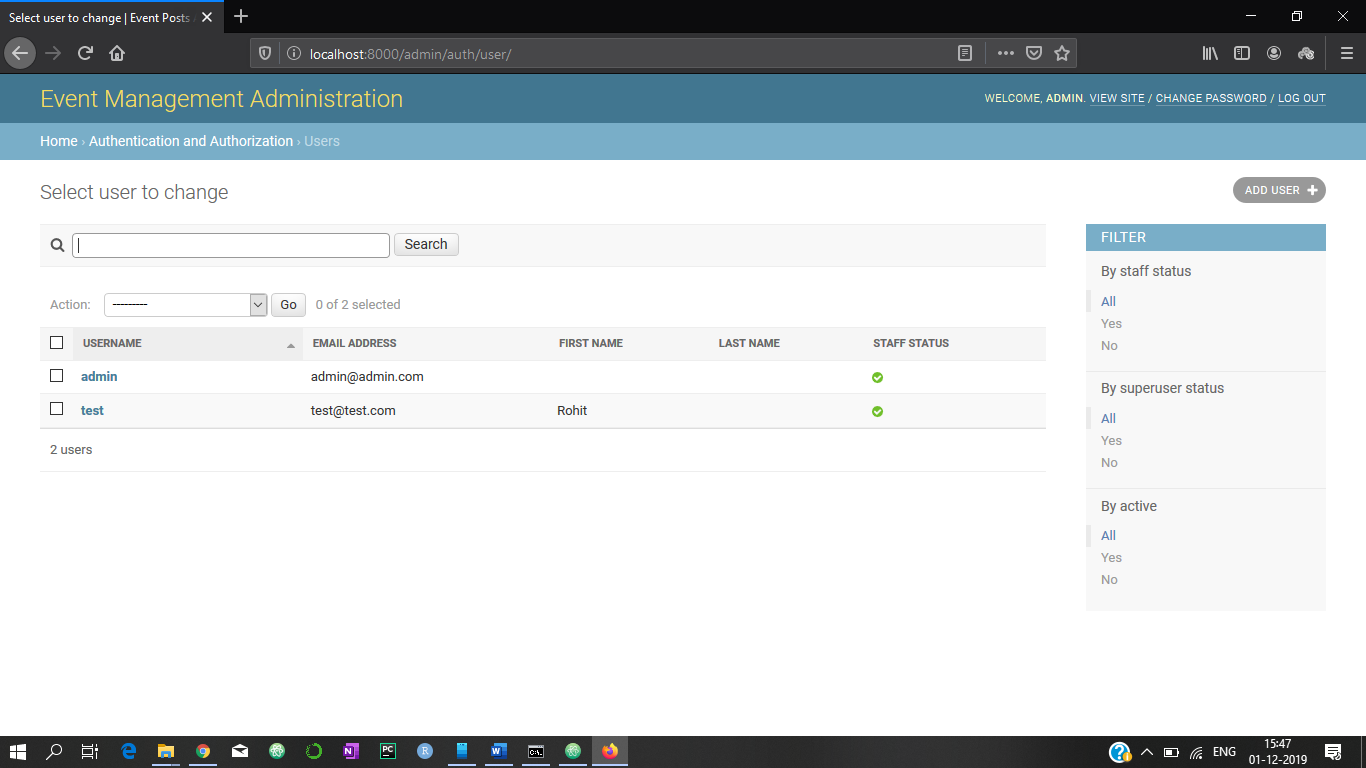
Flow: *Administration Button->admin.site.urls (Default Administration Panel in Django framework)*



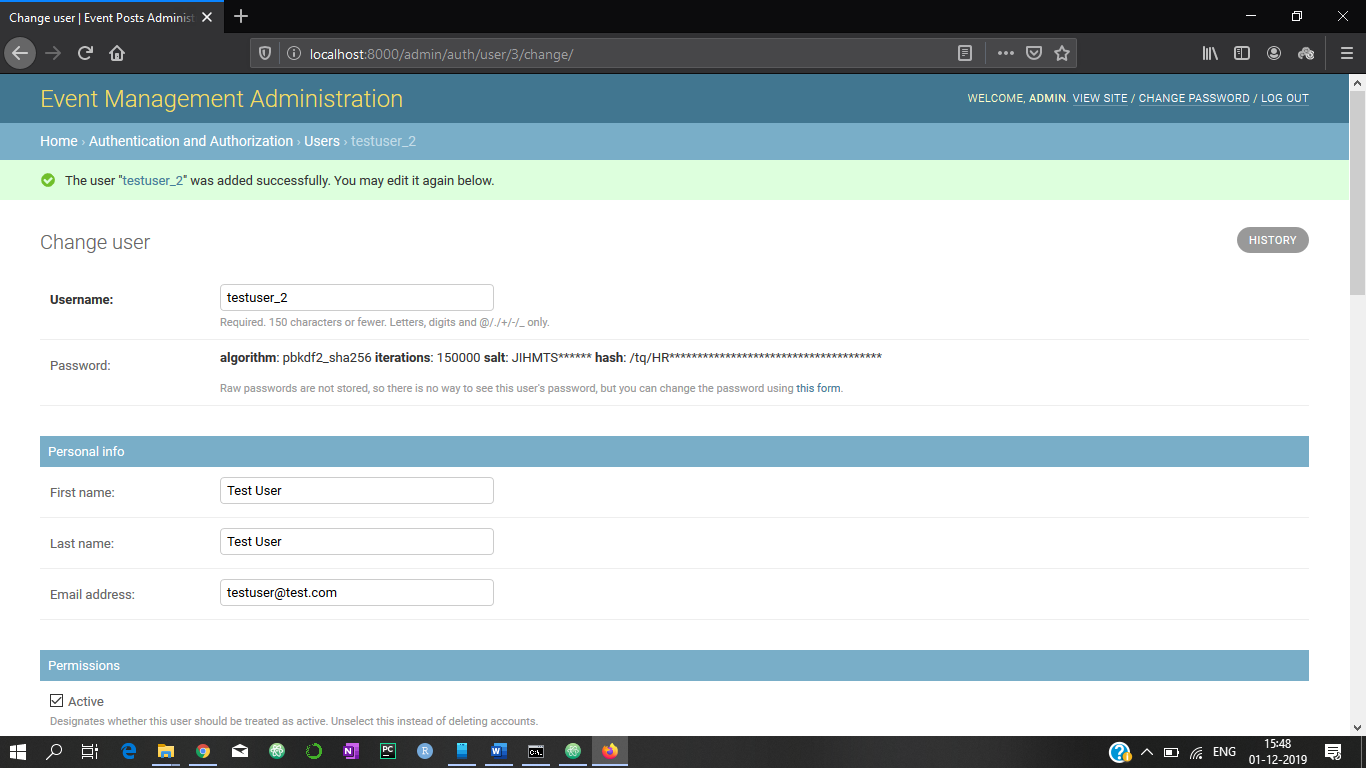
Administration Console for the web application Admin (From the default *admin.site*):



Authentication, Authorization for users:

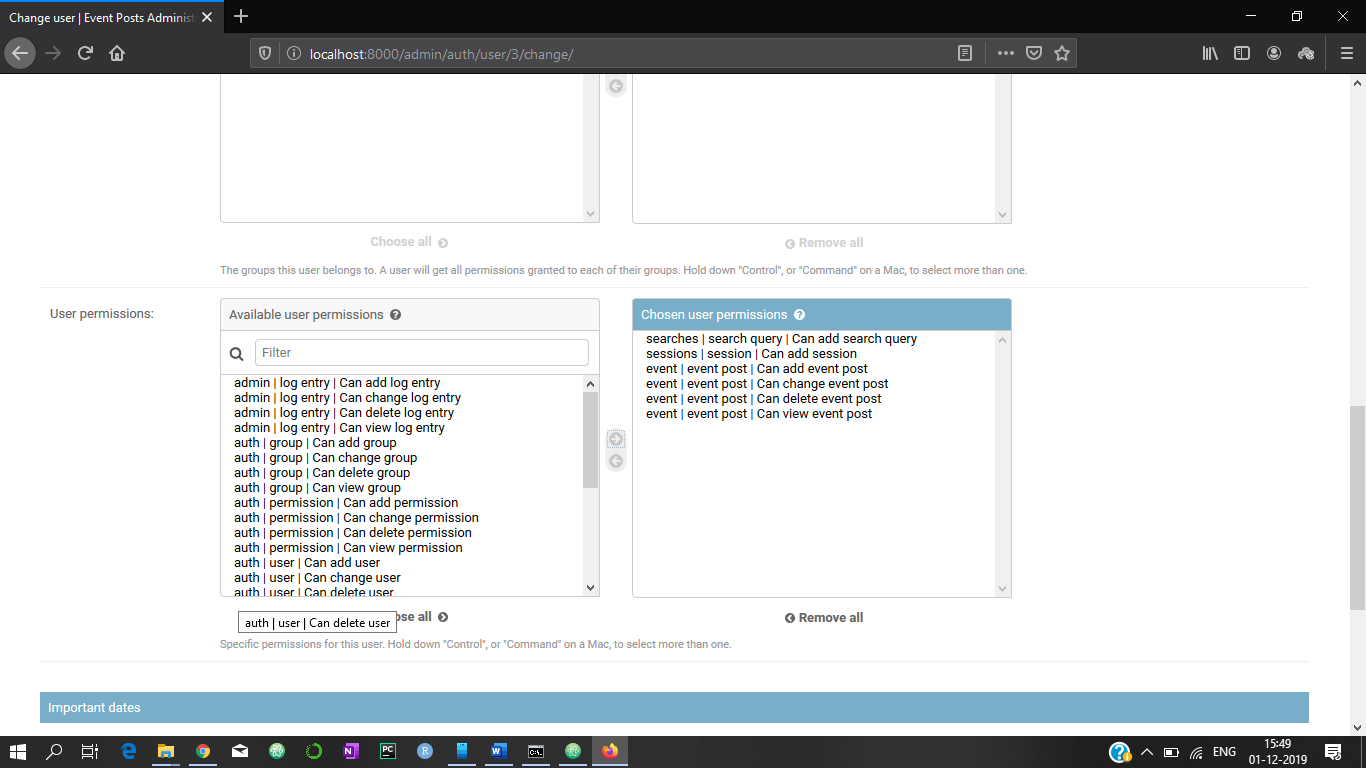


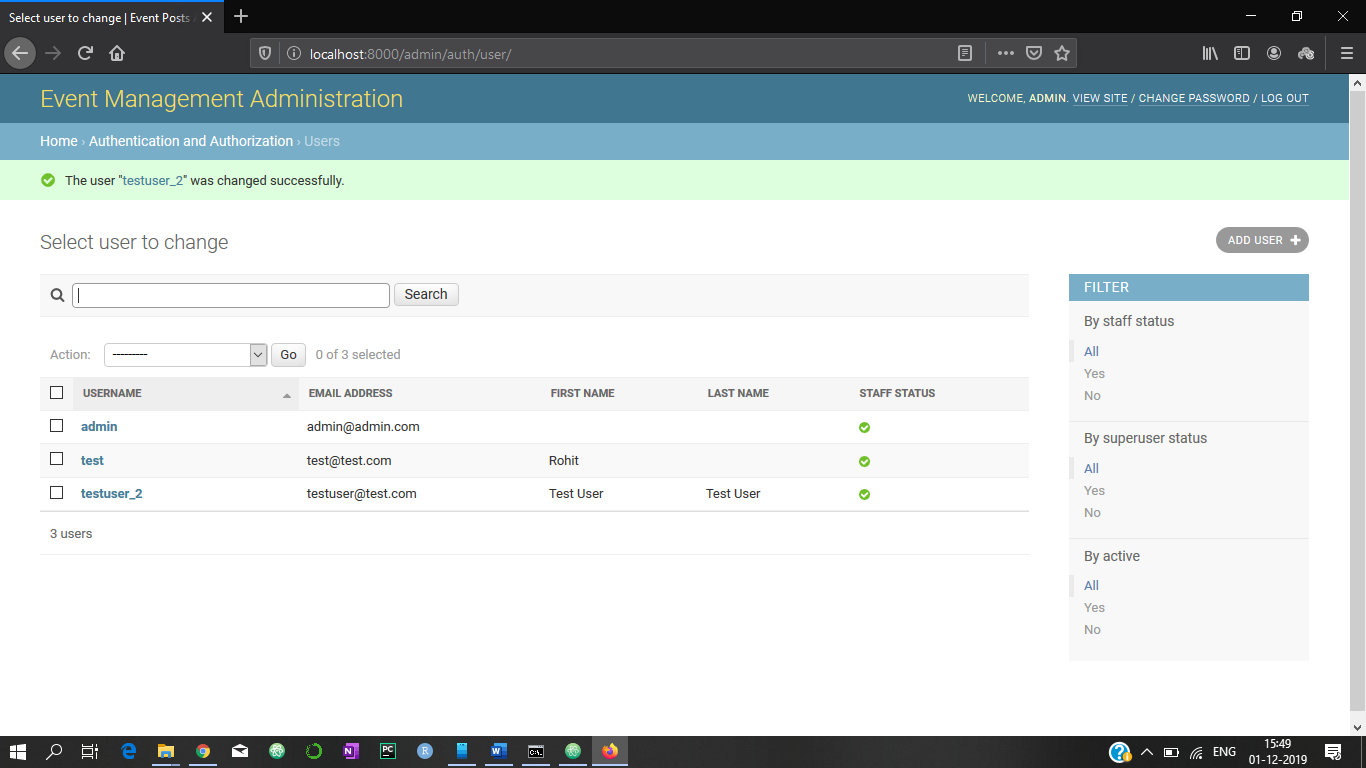
A new user: *testuser\_2* is added in the system who has permissions to *create/edit/modify/delete* and share events.



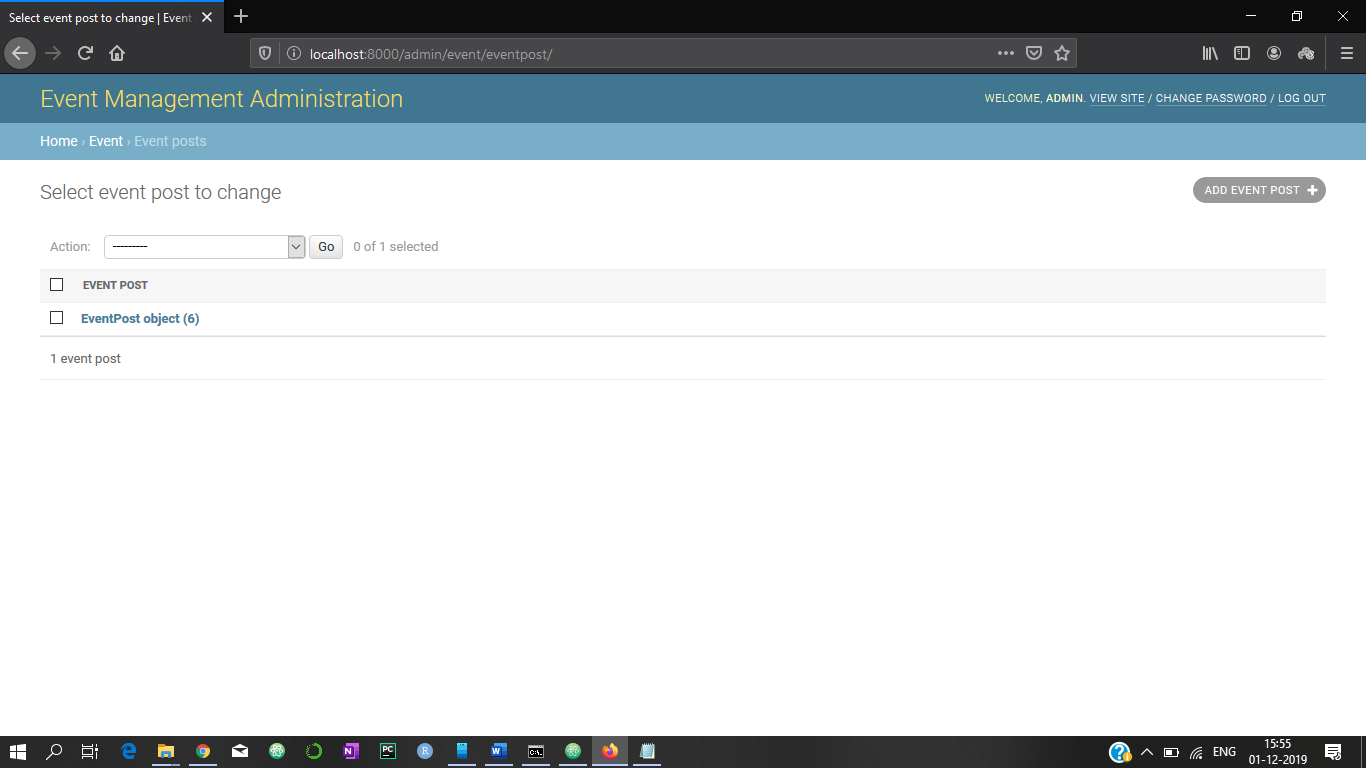


*Managing user permissions to create/edit/delete posts*

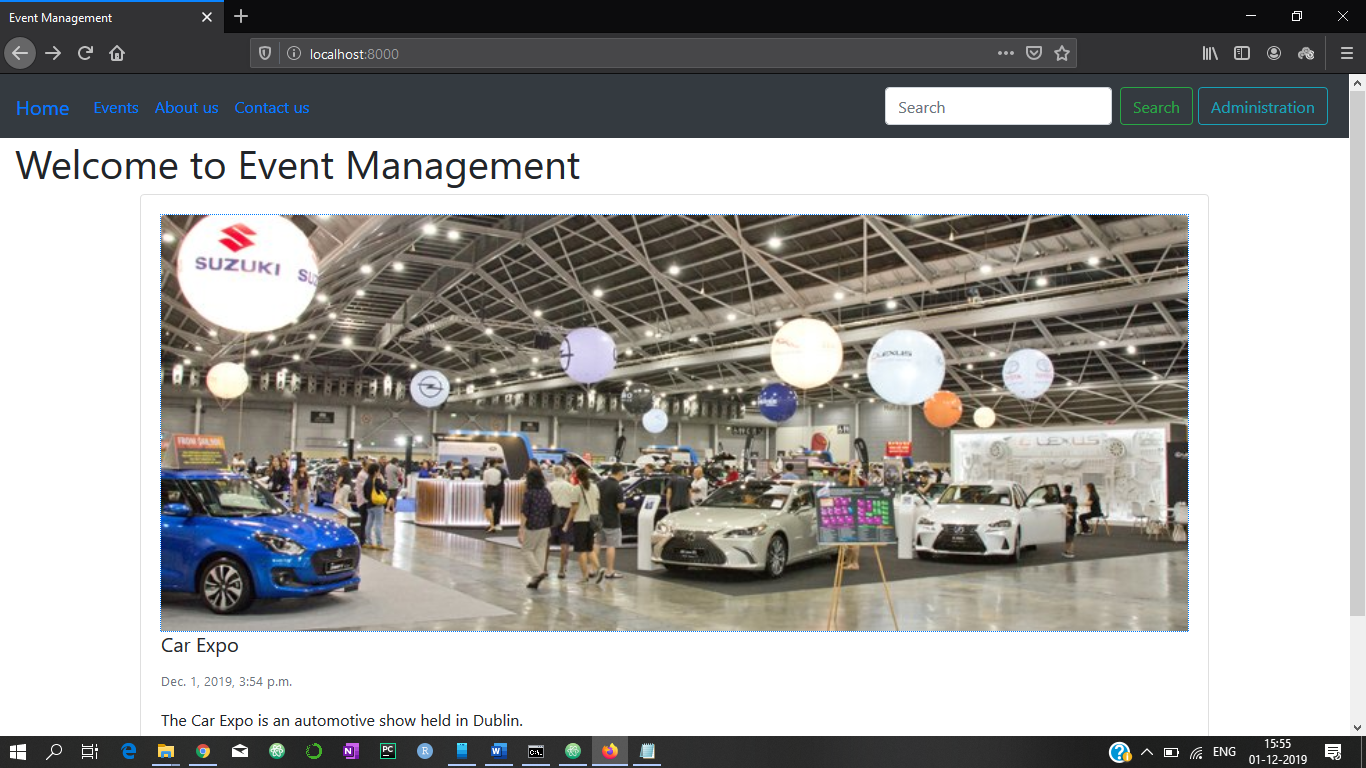




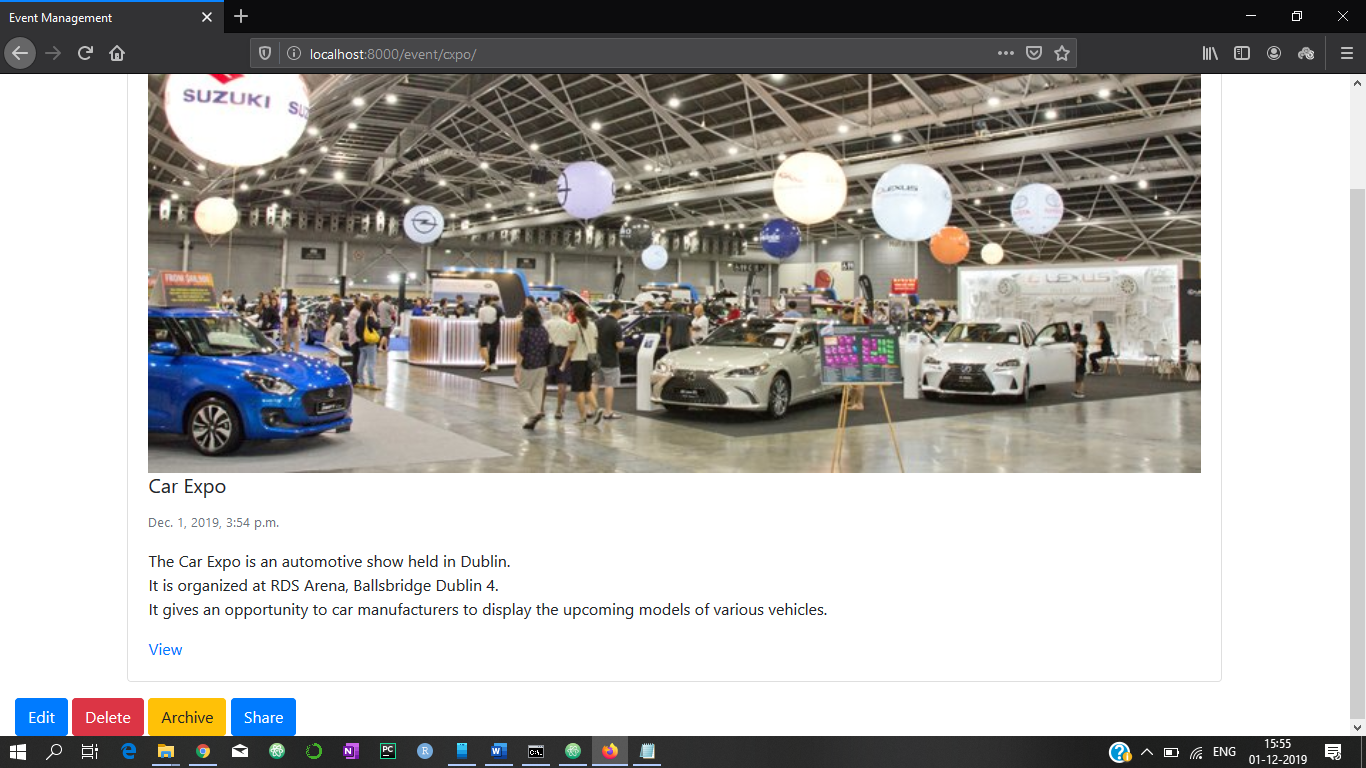
Event Added from the Event section in the web application:



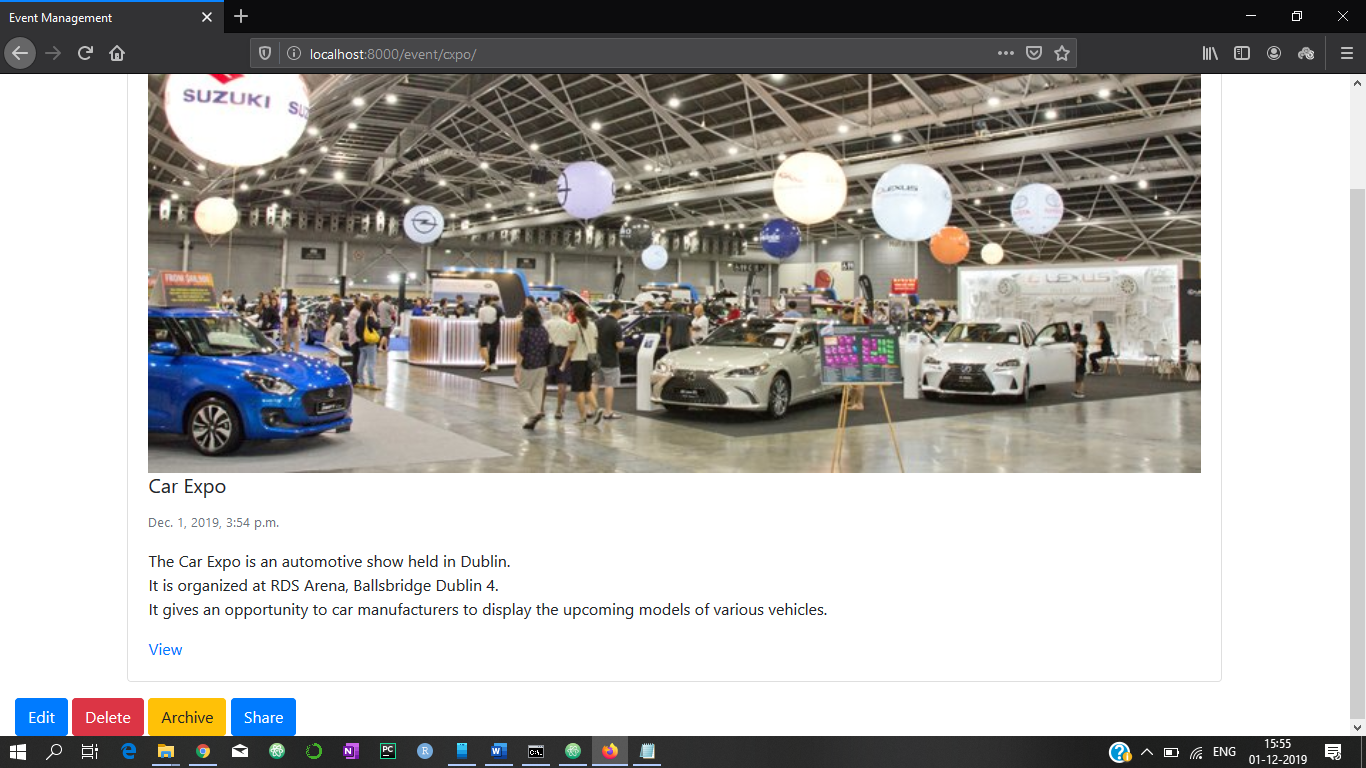
Site home page after adding the event:

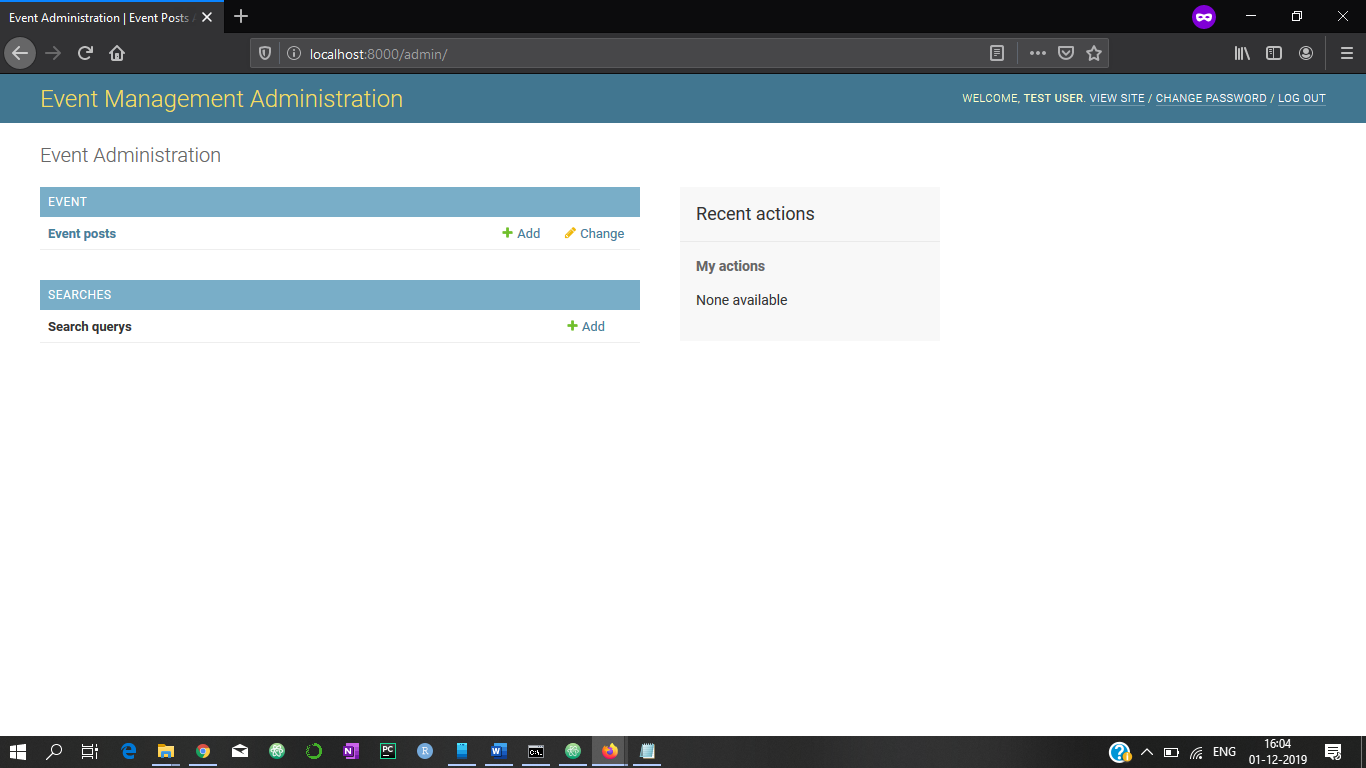


Detailed view of the web application: *event\_post\_detail\_view (event) -> render (detail.html)*



1. Features to Edit/Delete/Archive/Share when opened the detail view of the event:

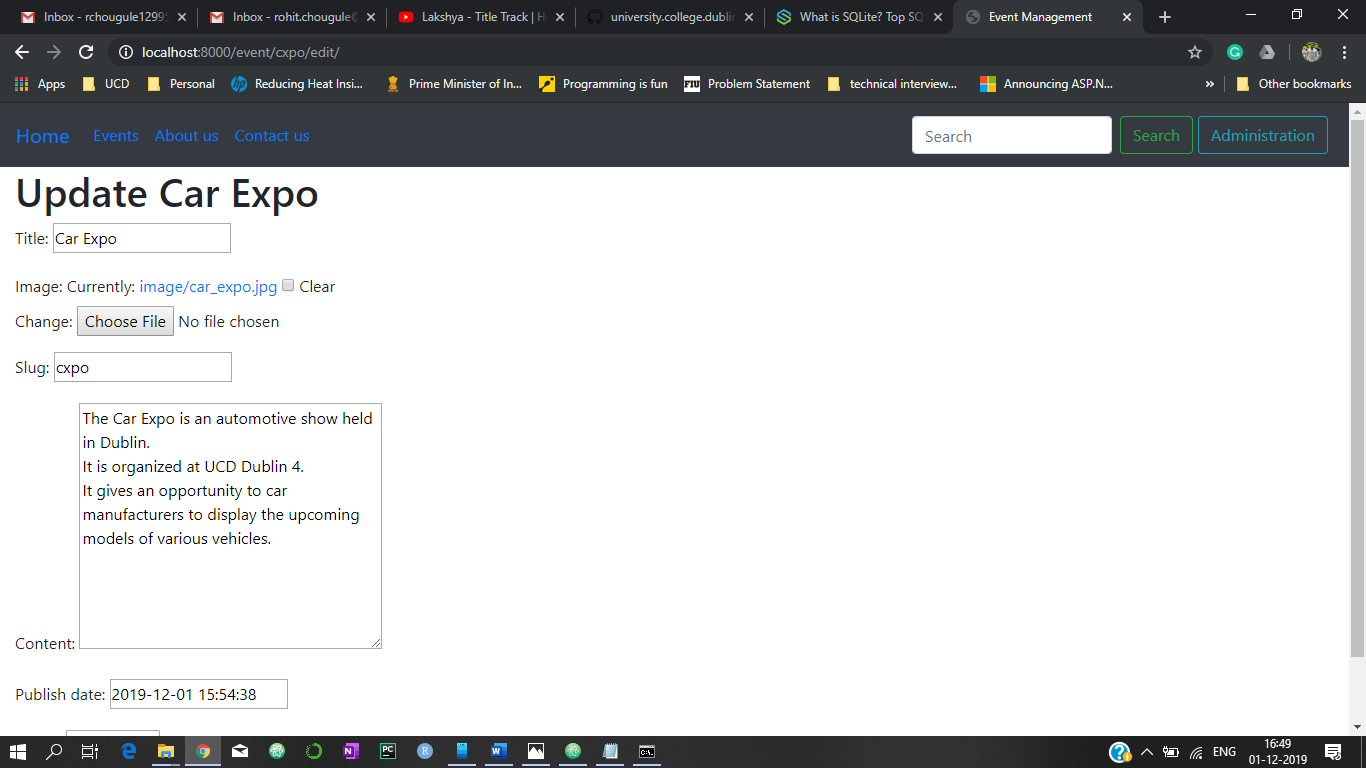




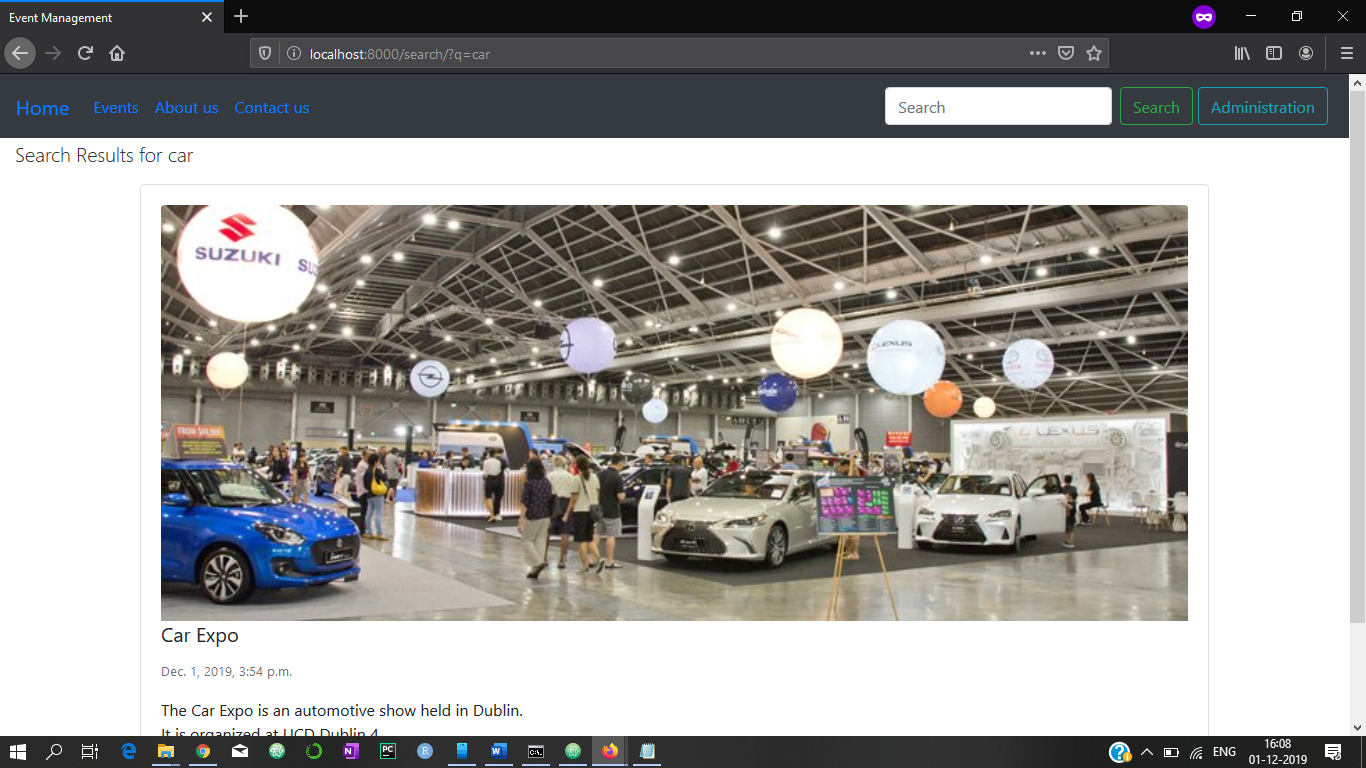
1. Edit Event:

*event\_post\_update\_view(event) -> form.html*

*event\_post\_archive\_view(event) -> form.html*



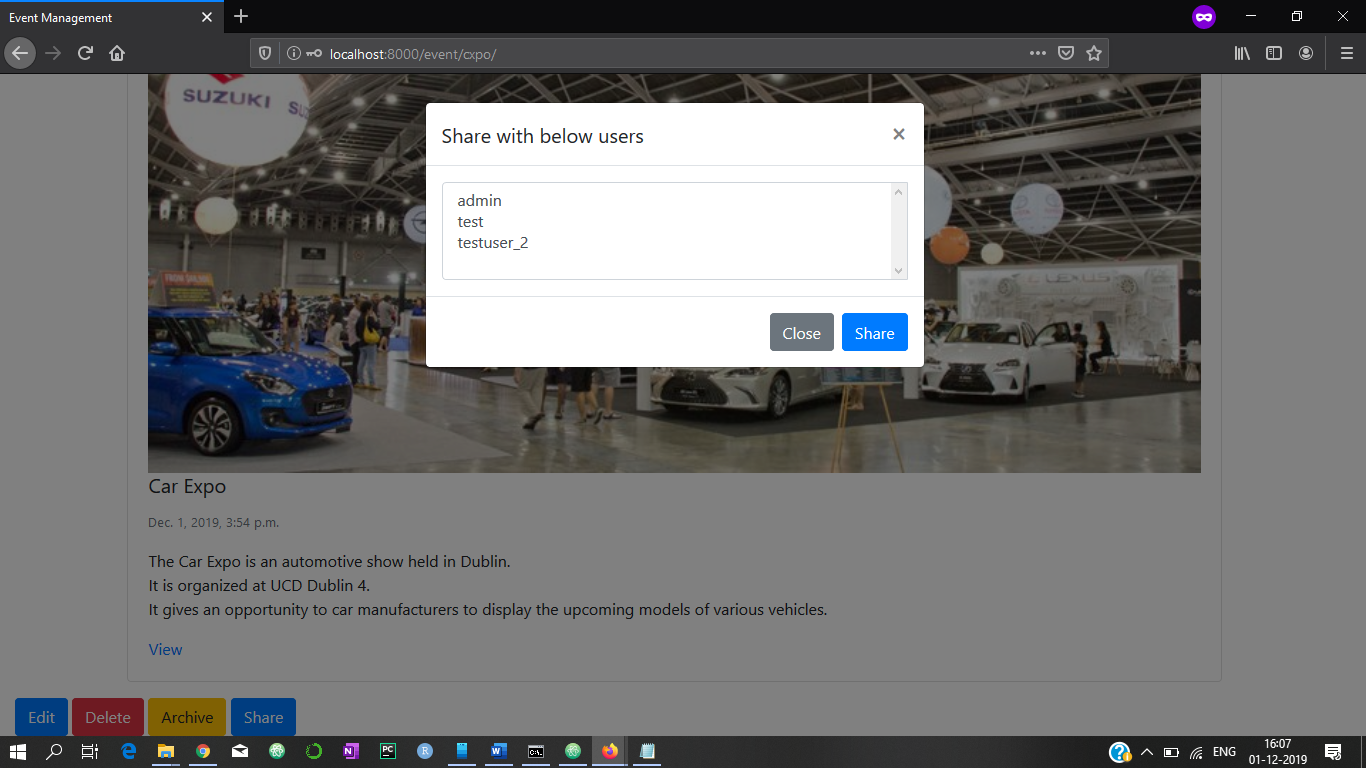
1. Search for Event: *search\_view(searches) -> searches/view.html*

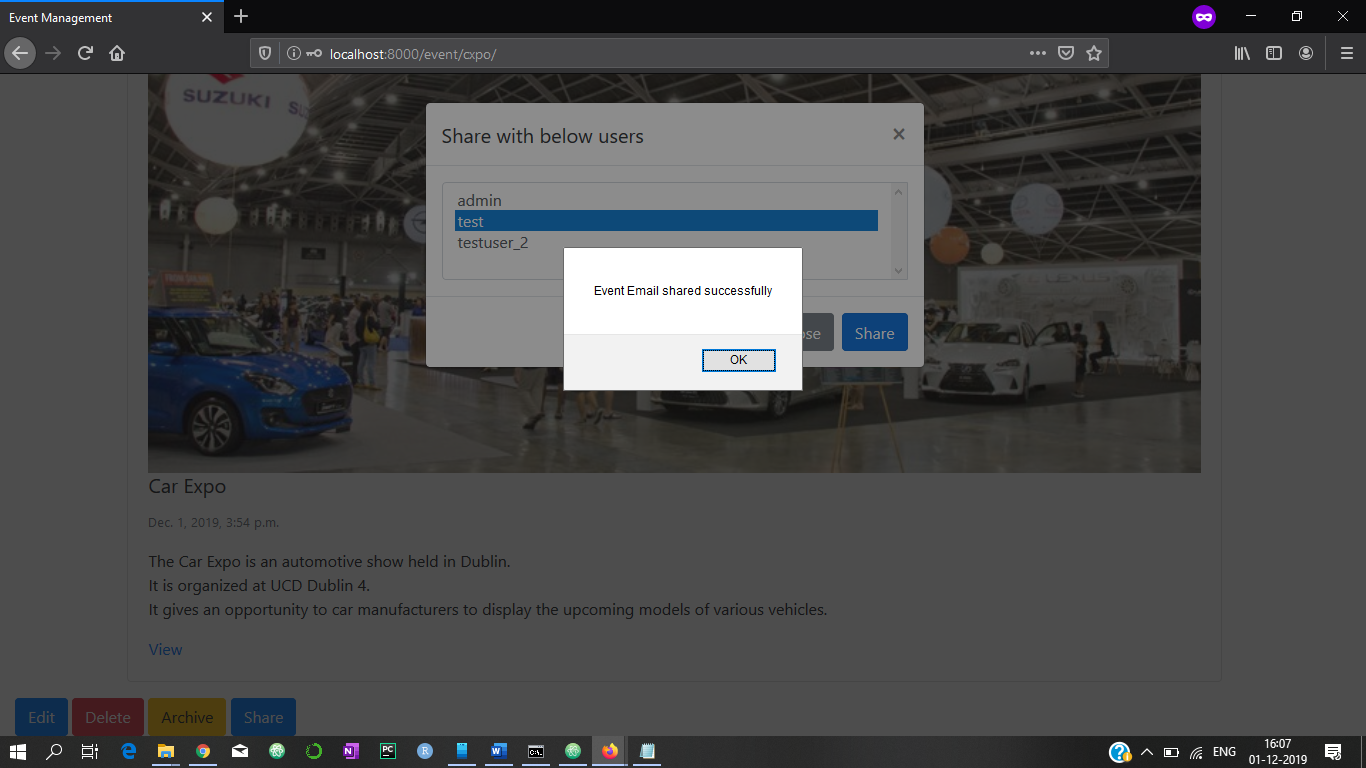


Archived Events are present in the database, its just that they are not shared with the user on the display page.

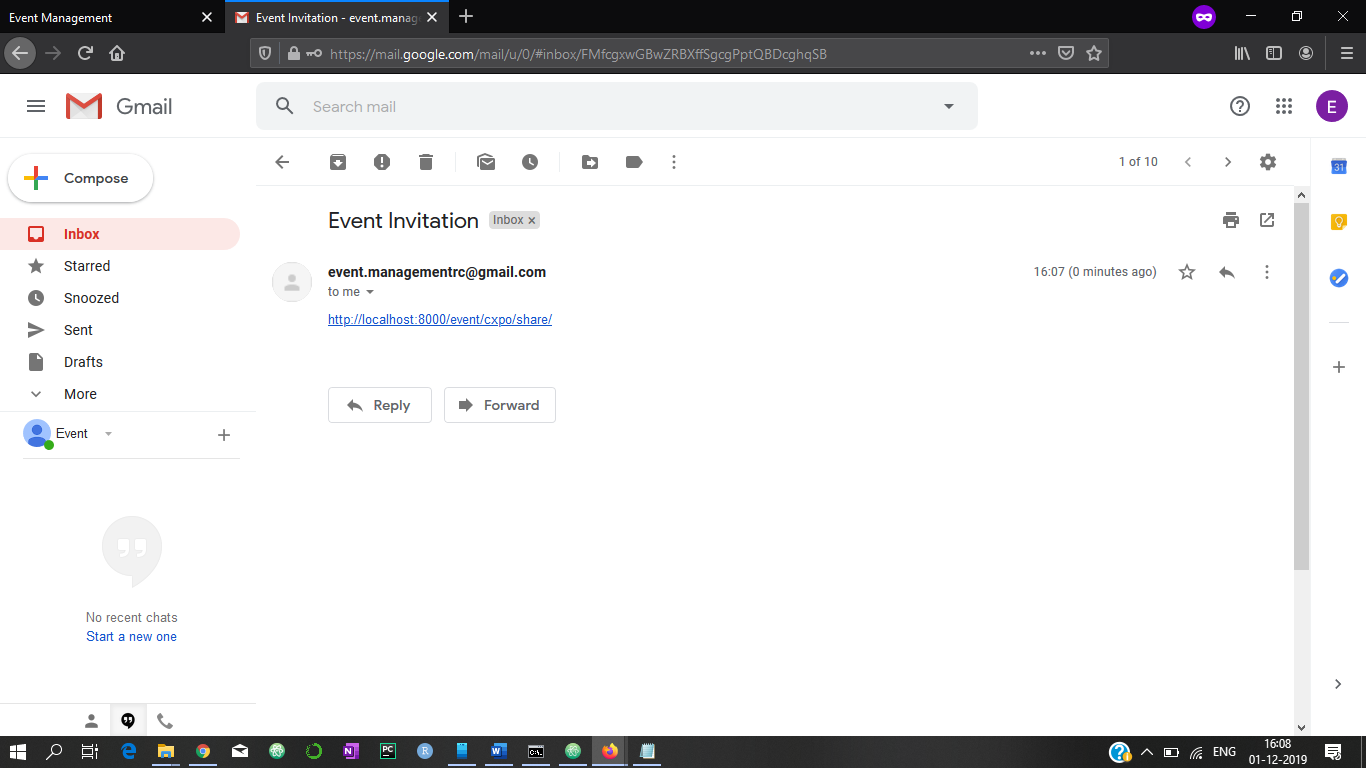
1. Share Event:

*event\_post\_share(event) -> email functionality from from django.core.mail import send\_mail*





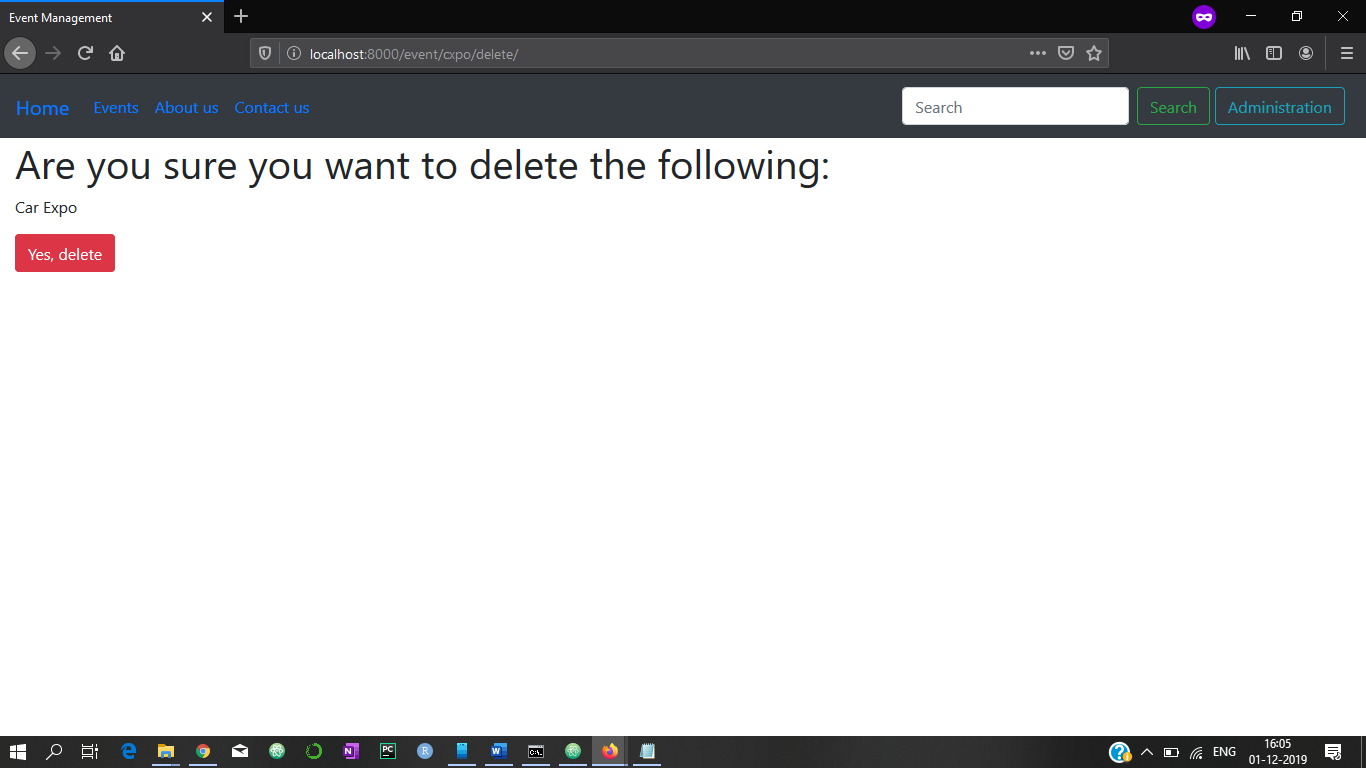
Email shared with the user for inviting to view the event:



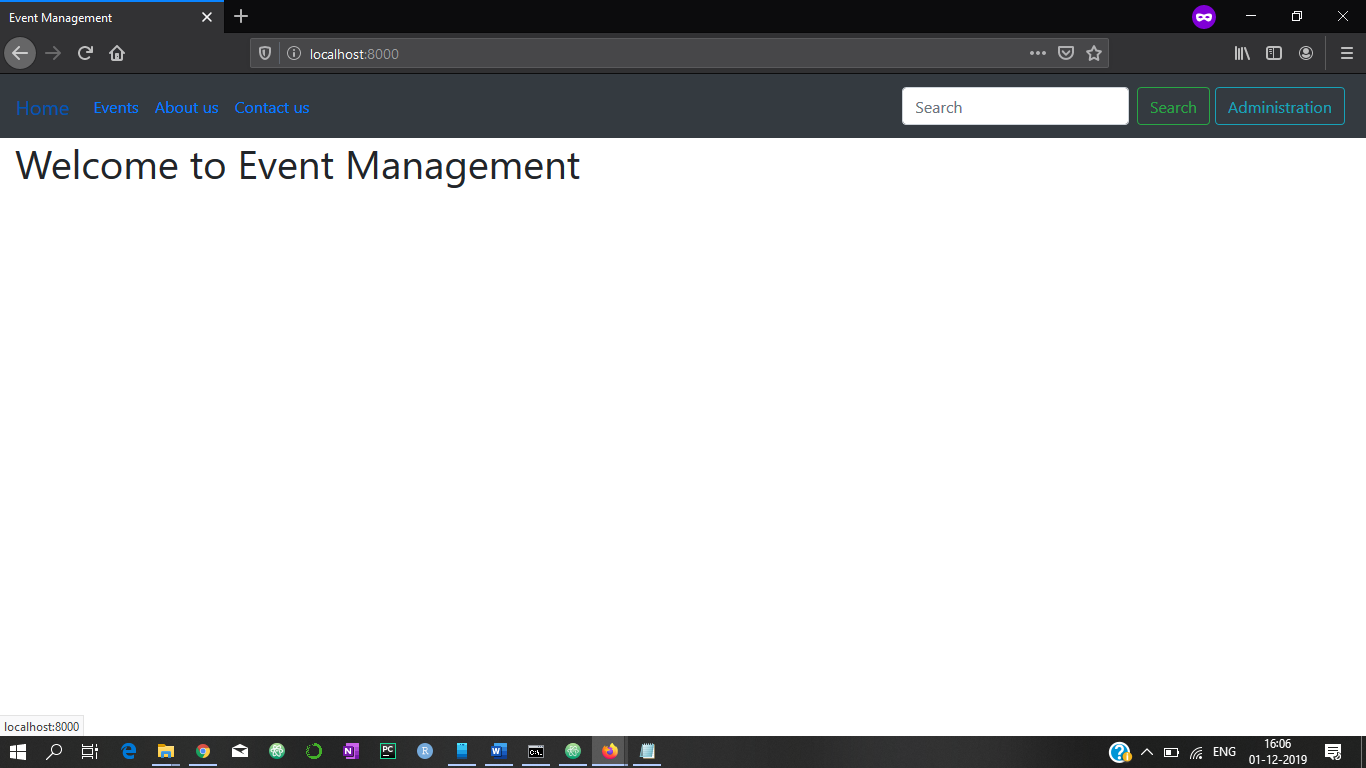
I have used the default, dummy Google account for sending and receiving the email on the same. However, we can share the event by email with whatever user we want, which will need some small modifications in the existing working of this feature.

1. Delete Event:

*event\_post\_delete\_view (event) -> event/delete.html*

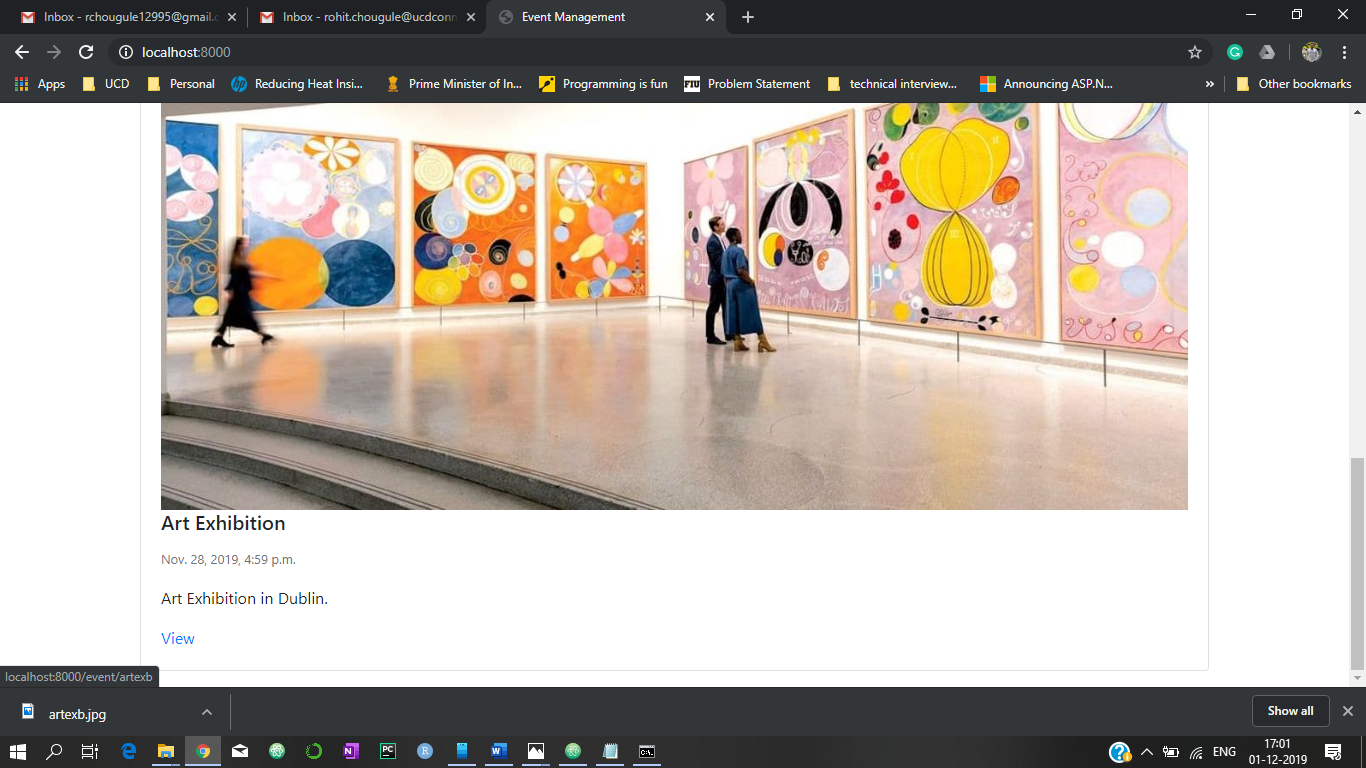


Home Page after deletion of the event:





List of multiple events:



The above screenshots show different functionality of the web application which includes, creating a new event, editing/deleting/archiving and sharing of events with users.

The media used the web application is saved at */src/static\_cdn\_test* which is defined in the *settings.py* for the app.

*MEDIA\_ROOT = os.path.join(LOCAL\_STATIC\_CDN\_PATH, 'media')*

*MEDIA\_URL = '/media/' # django-storages*