Documentation
Github address:
https://github.com/MohammadHasani/TheaterSeating/tree/develop
Live Demo: http://185.235.40.196/
Username: mohammad_hasani@live.com
Password: .U_YB+KT}r

1. Models:

If you look at the diagram picture attached, you will find out the app Is composed of:

1. Hall
   1.1. Desc: a saloon(theater, cinema, …)
2. Section
   2.1. Desc: a piece of the hall, like back hall, balcony …
3. Row
   3.1. Desc: every row that depends to a section
4. Seat
   4.1. Desc: every chair on a row
5. Rank
   5.1. Specific ranking for a seat (golden, VIP, …)
6. Seance
   6.1. Desc: a piece of time with a name. (for example, 22 December from 2 o'clock until 7 in Gran Theatre). Depends on hall
7. User

What is dynamic?:

You can define Hall As much as you want with different sections every section can have
Every row is in the middle, left, and right. You can define which section has curved left_side or right_ide rows. Also, the colors of the ranks are different.

A live demo with data is up in:http://185.235.40.196/

For example
http://185.235.40.196/tickets/?seance_name=The%20Sepration

You will see the Tehran Theatre Salon in “The Separation” Seance.
I defined the red color for VIP chairs. You can change it to this address.

Seating Algorithm
In the “Test Seating Algorithm,” you can pick a seance and send a list of int with JSON format. And it will run in non-blocking mode with celery. And then click on the below link and see what the magic happened.

Also, you can manually select a chair or multiple chairs and click on “buy ticket” for buying it.

If there is enough time, it could be better, more dynamic, and more efficient.

Restful API
I Used DRF to create a restful API.

1. In the address /tickets/wallet/<int:user_id>/<str:seance_name>/
   You can get the list of tickets bought by the user with entered user_id

Mohammad Hasani