# Aukshan

### **November 30,2020**

Youtube link (demo) :django E Aukshan project

Drive Link:

https://drive.google.com/file/d/1896icTy4GvRxZsvGBh5e61LpIm1QaB60/view?usp=sharing

Github link: <a href="https://github.com/melodyPereira05/Aukshan">https://github.com/melodyPereira05/Aukshan</a>

Note: Installing guide is present on the github links readme.md file.

# Made By:

70 - Sukhada Morgaonkar

78- Chetna Nihalani

84- Melody Pereira

<u>**Project Description:**</u> This project aims towards solving a common problem faced by customers - how to bid on a lot being auctioned at a remote location?

Looking at the current pandemic situation, safe distancing is must due to which in-person auctions cannot take place in order to avoid public gathering.

So we have designed a solution for this problem. An E-Auction website-'Aukshan'

#### Overview

Aukhan is an online E-bidding website mainly designed for buying, reselling antiques and luxurious items primarily focusing on providing maximum comfort to the customers.

### Why Aukshan?

- 1. **Convenience:** The primary benefit of online auction sites lies in the fact that you can bid on a product which is being auctioned remotely.you can make offers on your perfect home while relaxing at sofa or while on your work during lunch break.
- 2. **Save your precious time and money:** While there might be an auction of a luxurious item of your interest taking place around another

- corner of the world, you need not book tickets or waste money and time instead sit at your comfort of your home and bid online.
- 3. **Open 24/7:** countdown timers and regular updates might help you bid at any time zone.

# **Technology stack Used:**

#### Front End:

HTML,CSS, Sass,JavaScript, Bootstrap

Template credits : Aroma colorlib

#### Back End:

Framework: Django , Django channels

Database: Sqlite, Redis

# Why Django?

- Compatible with major operating systems and databases like (mongodb,postgres, sqlite,oracle,Mysql). It has official documentation explaining the connectivity of various db with django.
- Developers do not have to get their hands dirty in learning databases.
   Django's ORM helps convert python codes to database queries.
- Follows a specific structure (unless changed manually) which helps other developers understand which part of code is where thereby saving a huge amount of time.
- In built Admin Interface.
- As it is written in python, easy to learn and minimum lines are needed to code.

# **Django Channels:**

Channels is a project that takes Django and extends its abilities beyond HTTP - to handle WebSockets, chat protocols, IoT protocols, etc. It's built on a Python specification called ASGI . (Asynchronous Server Gateway Interface, is the specification which Channels are built upon, designed to untie Channels apps from a specific application server and provide a common way to write application and middleware code.)

# Redis:

Redis is an in-memory data structure store that can be used as a caching engine. Since it keeps data in RAM, Redis can deliver it very quickly. Redis is not the only product that we can use for caching.

# **Project Working:**

Customers can visit **Aukshan's website** to know about the latest ,trending antique and luxuries items being auctioned.

Categories range from Art, Coins, Luxuries cars, Historic and antique properties, fundraising, Coins and many more.

To know more about a product see the specification of an item, read comments about the product being auctioned, read the blog to know more about the upcoming items that are going to be auctioned.

Visit the trending page to know which item is currently bidded the most.

Search for any category, name of product, seller name or any descriptive keyword to get relevant search results.

To know more about the item's seller and their authenticity ,visit their page where their personal information and items they have auctioned successfully or are currently being auctioned are displayed.

A Customer interested in bidding has to register and subsequently login. Once logged in, the user is free to bid on any live auction present. The bidded amount will be visible to all the customers visiting that specific item.

Remember once bidded, the bid cannot be taken back, read the terms and conditions for more.

The Customer with the highest bid before the countdown ends wins the Item. An email will be sent to him/her for further communication.

A Customer can make enquiry regarding any product being auctioned. The enquiry details will be available on the enquiry page of the Customer. And will be sent to the seller of the item. Since it is an auction site only items to be auctioned are available.

To get an item, the customer has to fight for it, bid the highest and take the item home, unlike e-commerce websites where products are brought without any constraints.

### **Django Channels and Redis working:**

With normal E-commerce websites customer cart information is stored in session cookies. And displayed to the user when they log in. This functionality stays intact to that specific customer.

But the main challenge in our project was when one customer who is logged in bids, the rest of the people watching the auction or will watch should be able to see this bid and the next value to be bidded should be updated and the value should stay even if the page is refreshed.

With only django this is not possible and here comes django Channels and redis.

With WebSockets (via Django Channels) managing the communication between the client and the server, whenever a user is authenticated, an event will be broadcasted to every other connected user to that specific room(in our case auctioned product page). <u>Each user's screen will change automatically, without them having to reload their browsers.</u>

#### **Use of Redis:**

Application performance is vital to the success of your product. In an environment where users expect website response times of less than a second, the consequences of a slow application can be measured in dollars and cents. Even if you are not selling anything, fast page loads improve the experience of visiting your site. As redis keeps data in ram, excessing the data as quick as possible becomes easy.

Note: Due to time constraint we couldn't implement few features we thought of.Will update soon.Thank you

### **Database Tables:**

Database Schema:

Since django follows ORM pattern, we do not need to get our hands dirty in learning any database, instead write code in python and django does the rest for us.

Description of the schema -

#### Lot -

```
class Lot(models.Model):
   category=models.ForeignKey(Category, related name='lots',
on delete=models.CASCADE)
   slug=models.SlugField(max length=200,db index=True)
   product name=models.CharField(max length=200,
                 db index=True)
main photo=models.ImageField(upload to='photos/main/%Y/%m/%d/')
photo1=models.ImageField(upload to='photos/optional/%Y/%m/%d/',
blank=True)
```

```
photo2=models.ImageField(upload_to='photos/optional/%Y/%m/%d/',
blank=True)

photo3=models.ImageField(upload_to='photos/optional/%Y/%m/%d/',
blank=True)

photo4=models.ImageField(upload_to='photos/optional/%Y/%m/%d/',
blank=True)

is_trending=models.BooleanField(default=False)

on_banner=models.BooleanField(default=False)

year_published=models.DateTimeField(default=datetime.now())
```

# Category -

```
class Category(models.Model):

name=models.CharField(unique=True, max_length=200, db_index=True)
slug=models.SlugField(max_length=200, unique=True)
```

### Seller -

```
class Seller(models.Model):
    name=models.CharField(max_length=200)

seller_photo=models.ImageField(upload_to='photos/seller/%Y/%m/%d/')
    contact_no =models.CharField(max_length=200)
    email=models.CharField(max_length=200)
```

#### Auction -

```
class Auction(models.Model):
    start = models.DateTimeField(auto_now=False)
    curr_time = models.DateTimeField(auto_now=False)
    item = models.OneToOneField(
    Lot, on_delete=models.CASCADE, related_name="auction")
```

#### Wishlist -

```
class Wishlist(models.Model):
    lot=models.CharField(max_length=1000)
    name=models.CharField(max_length=200)

slug=models.SlugField(max_length=200,db_index=True,null=True,blank=True)
    lot_id=models.IntegerField()

Wishlisted_date=models.DateTimeField(default=datetime.now,blank=True)
    user_id=models.IntegerField(blank=False)
```

#### Contact -

```
class Contact(models.Model):
    lot=models.CharField(max_length=1000)
    lot_id=models.IntegerField()
    name=models.CharField(max_length=200)
    email=models.CharField(max_length=200)

slug=models.SlugField(max_length=200,db_index=True,null=True,blank=True)
    message=models.TextField(blank=True)

contact_date=models.DateTimeField(default=datetime.now,blank=True)
    user_id=models.IntegerField(blank=True)
```

### Message -

```
class Message(models.Model):
    auction=models.ForeignKey(Auction,
on_delete=models.CASCADE, related_name='auction')

author=models.ForeignKey(User, related_name='author_messsages', on_delete=models.CASCADE)

timecap = models.DateTimeField(auto_now_add=True)
```

```
price = models.TextField()
```

#### Subscribe -

```
class Subscribe(models.Model):
   email=models.EmailField()
```

#### Post for blog -

```
created=models.DateTimeField(default=timezone.now)

updated=models.DateTimeField(default=timezone.now)

status=models.CharField(max_length=10,choices=STATUS_CHOICES,default='draft')

objects=models.Manager()

published=PublishedManager()

tags=TaggableManager()
```

#### Comments for blog -

```
class Comment(models.Model):

post=models.ForeignKey(Post,on_delete=models.CASCADE,related_name='comments')

name=models.CharField(max_length=100)

email=models.EmailField()

body=models.TextField()

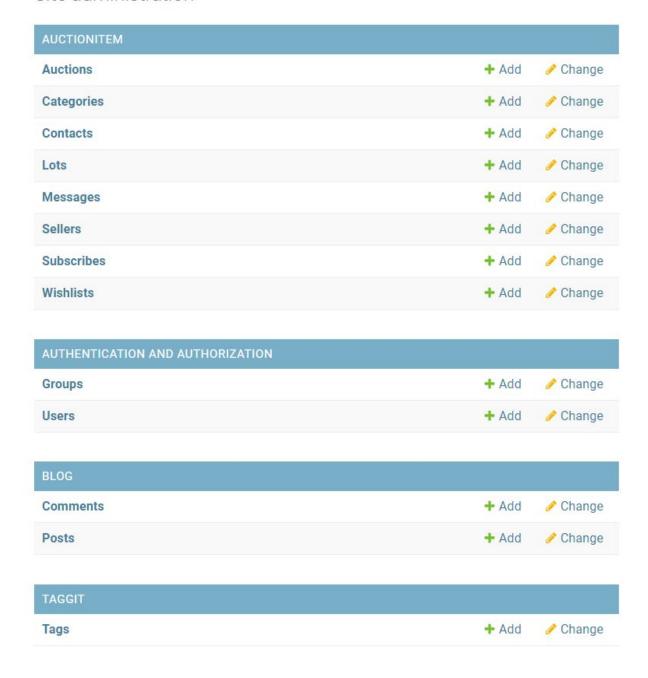
created=models.DateTimeField(auto_now_add=True)

updated=models.DateTimeField(auto_now_add=True)

active=models.BooleanField(default=True)
```

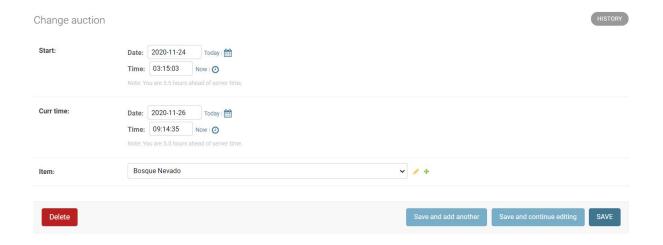
#### Screenshots of Aukshan Database.

#### Site administration

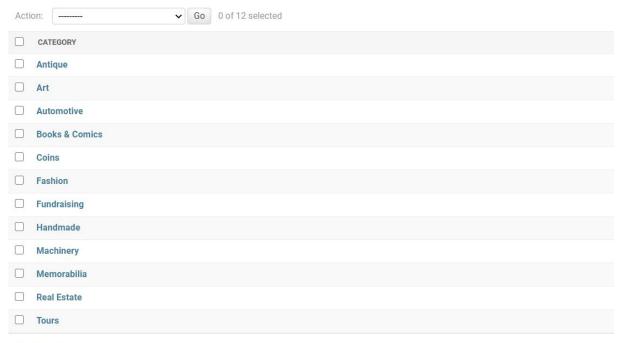


# Select auction to change

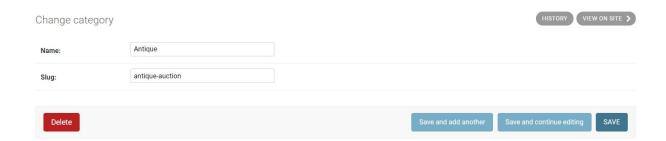
Acti	on: Go 0 of 34 selected
	AUCTION
	Bosque Nevado
	Indian 200BC Rupee
	Wicker bagit
	Whiskey Made Me do it Black capADD TO CART WHISKEY MADE ME DO IT BLACK CAP
	Vintage book
	Vintage Fruit Bowl
	Villa Flowers
	Sulpicia, 106 a.C.
	Stainless Pocket Watch
	Roman & Byzantine Coin
	Rumours Remaster
	Roman & Byzantine Coin
	Red Vintage Car
	Pink Floyd - The Dark Side Moon
	Organico Monaco
	Onxy Black Pearl
	Miletos, c. 295-275



#### Select category to change

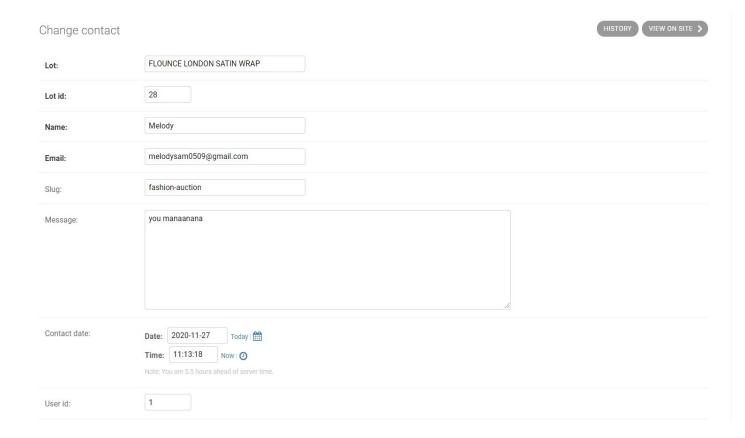


12 categories

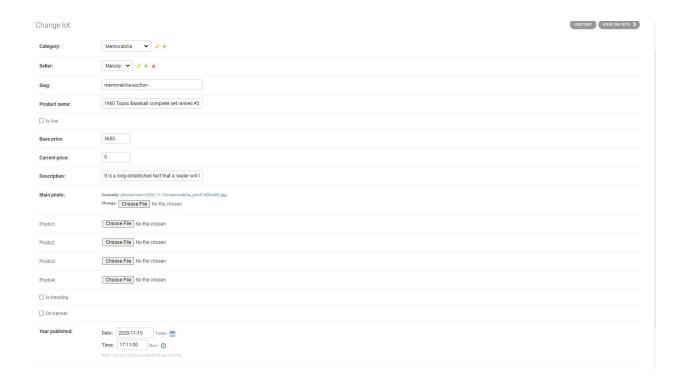


#### Select contact to change



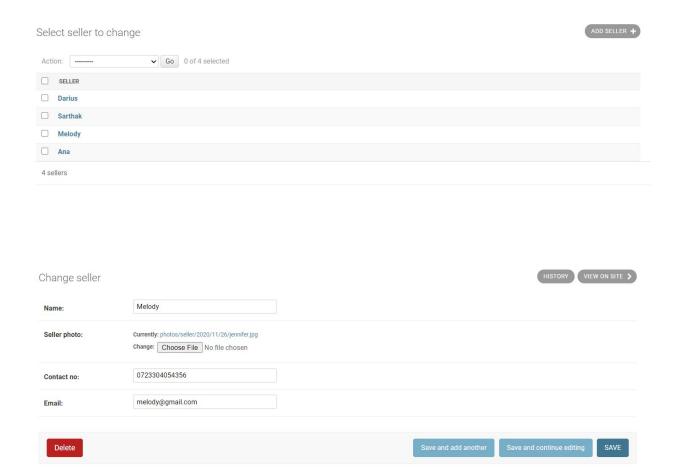


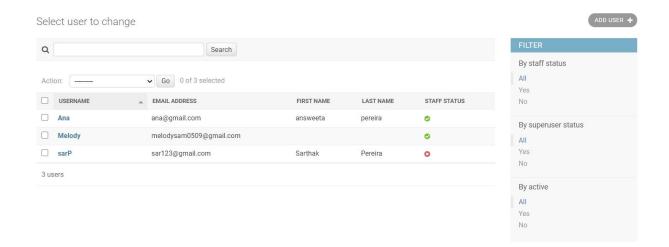
Sele	ect lot to change	ADD LOT -
Acti	ion: Go 0 of 34 selected	
	LOT	
	1960 Topss Baseball complete set ranked #20 On PSA Registry With 8.01 Set	
	21 CD Cover LP (12" album) - Adele	
	BOOHOO WRAP FRONT BODY	
	Beautiful Pyrite Stone	
	Bosque Nevado	
	Bronzo, c. 415-387	
	Bronzo, c. 415-387	
	Calvin Johnson Signed Detroit Game Worn Jersey Photo-Matched To Win	
	Classic American Car	
	FLOUNCE LONDON SATIN WRAP	
	Indian 200BC Rupee	
	Jewelry Container	
	Killer Fiat 500	
	Latitude Penthouse	
	Le bouquet de Paris	
	Longitude Wave	
	Maison Alsacienne	











# Select comment to change





