```python

from django.contrib.auth import authenticate, login, logout
from django.db import IntegrityError
from django.http import HttpResponse, HttpResponseRedirect
from django.shortcuts import render
from django.urls import reverse

from .models import User, AuctionListing, Bid, Comment, Category, Info


def index(request):
    auction_listings = AuctionListing.objects.exclude(listing_open=False)

    for listing in auction_listings:
        try:
            Bid.objects.get(bid_identifier=listing.id, watchBidder=request.user)
            listing.display_button = False
        except:
            listing.display_button = True

    return render(request, "auctions/index.html",{
        "listings": auction_listings,
        "type": "Active Listings"
    })


def login_view(request):
    if request.method == "POST":

        # Attempt to sign user in
        username = request.POST["username"]
        password = request.POST["password"]
        user = authenticate(request, username=username, password=password)

        # Check if authentication successful
        if user is not None:
            login(request, user)
            return HttpResponseRedirect(reverse("index"))
        else:
            return render(request, "auctions/login.html", {
                "message": "Invalid username and/or password."
            })
    else:
        return render(request, "auctions/login.html")


def logout_view(request):
    logout(request)
    return HttpResponseRedirect(reverse("index"))


def register(request):
    if request.method == "POST":
        username = request.POST["username"]
        email = request.POST["email"]

        # Ensure password matches confirmation
        password = request.POST["password"]
        confirmation = request.POST["confirmation"]
        if password != confirmation:
            return render(request, "auctions/register.html", {
                "message": "Passwords must match."
            })

        # Attempt to create new user
        try:
            user = User.objects.create_user(username, email, password)
            user.save()
        except IntegrityError:
            return render(request, "auctions/register.html", {
                "message": "Username already taken."
            })
        login(request, user)
        return HttpResponseRedirect(reverse("index"))
    else:
        return render(request, "auctions/register.html")

def create(request):
    if request.method == "POST":
        title = request.POST['listing_title']
        description = request.POST['listing_description']
        price = request.POST['start_bid']
        photo = request.POST['listing_image']
        creator = request.user

        if request.POST['listing_category']:
            listing_category = Category.objects.get(pk=int(request.POST['listing_category']))
        else:
            listing_category = Category.objects.get(pk=1)

        new_listing = AuctionListing(
            title=title,
            description=description,
            start_price=price,
            current_bid=price,
            photo=photo,
            creator=creator,
            listing_category=listing_category)

        new_listing.save()

        return HttpResponseRedirect(reverse("index"))
    return render(request, "auctions/create.html",{
        "categories": Category.objects.exclude(pk=1)
    })

def listing_page(request, listing_id):
    listing = AuctionListing.objects.get(pk=listing_id)
    min_bid = listing.current_bid + 5
    comments = listing.listing_comment.all()

    if listing.listing_open == False:
        close = "CLOSED"
        try:
            info = Info.objects.get(won_listing=listing)
        except:
            info = ""
    else:
        close = ""
        info = ""
    try:
        condition = listing.bids.get(bidder__isnull=False)
    except:
        condition =""

    if not condition:
        close_button = False
    else:
        close_button = True

    try:
        Bid.objects.get(bid_identifier=listing_id)
        button = False
    except:
        button = True

    context = {
        "listing": listing,
        "status": listing.listing_open,
        "min_bid": min_bid,
        "info": info,
        "closed": close,
        "button": button,
        "cButton": close_button,
        "comments": comments
    }
    return render(request, "auctions/listing.html", context)

def watchlist(request):
    if request.method == "POST":
        wList = request.POST['wList']
        page = request.POST['page']
        watch_list = AuctionListing.objects.get(pk=wList)
        Bidder = request.user

        new_bid = Bid(
            bid_item=watch_list,
            bid_price=watch_list.current_bid,
            watchBidder=Bidder,
            bid_identifier=wList
        )

        new_bid.save()

        if page == 1:
            return HttpResponseRedirect(reverse("listing", kwargs={"listing_id":wList}))
        else:
            return HttpResponseRedirect(reverse("index"))


    else:
        listings = []
        bid_listings = request.user.listing_watchBidder.all()
        for bid in bid_listings:
            listings.append(AuctionListing.objects.get(pk=bid.bid_identifier))

        for listing in listings:
            try:
                Bid.objects.get(bid_identifier=listing.id, watchBidder=request.user)
                listing.display_button = False
            except:
                listing.display_button = True

        return render(request, "auctions/index.html",{
            "listings": listings,
            "type": "my Watchlist"
        })


def bidlist(request):
    bid_listings = request.user.listing_bidder.all()
    if request.method == "POST":
        bItem = request.POST['bItem']
        bid_list = AuctionListing.objects.get(pk=bItem)
        bid_list.current_bid = request.POST["bid_price"]
        bid_list.save()
        Bidder = request.user

        try:
            update_bid = Bid.objects.get(bid_item=bid_list, watchBidder=Bidder)
```

```
                update_bid.bidder = Bidder
                update_bid.bid_price = bid_list.current_bid
                update_bid.save()
            except:
                new_bid = Bid(
                    bid_item=bid_list,
                    bid_price=bid_list.current_bid,
                    bidder=Bidder,
                    watchBidder=Bidder,
                    bid_identifier=bItem
                )
                new_bid.save()


            return HttpResponseRedirect(reverse("listing", kwargs={"listing_id":bItem}))

        else:
            listings = []

            for bid in bid_listings:
                listings.append(AuctionListing.objects.get(pk=bid.bid_identifier))

            for listing in listings:
                try:
                    Bid.objects.get(bid_identifier=listing.id, watchBidder=request.user)
                    listing.display_button = False
                except:
                    listing.display_button = True

            return render(request, "auctions/index.html",{
                "listings": listings,
                "type": "my Bids"
            })

    def close_lisiting(request):
        if request.method == "POST":
            bItem = request.POST['lastBid']
            listing = AuctionListing.objects.get(pk=bItem)
            listing.listing_open = False
            listing.save()

            last_bid = Bid.objects.get(bid_item=listing, bid_price=listing.current_bid)
            winner = last_bid.bidder

            a_info = User.objects.get(username=listing.creator)
            w_info = User.objects.get(username=winner)

            closing_info = Info(won_listing=listing,
            author_Username=listing.creator,author_Email = a_info.email, winner_Username =
            last_bid.bidder,winner_Email = w_info.email)
            closing_info.save()

            return HttpResponseRedirect(reverse("listing", kwargs={"listing_id":bItem}))


    def my_listing(request):
        listings = request.user.listing_creator.all()
        return render(request, "auctions/index.html",{
            "listings": listings,
            "type": "my Listings"
        })

    def comments(request):
        if request.method == "POST":
            comment = request.POST['comment']
            listing_id = request.POST['listing_id']
            item = AuctionListing.objects.get(pk=listing_id)
```

```python
        new_comment = Comment(comment=comment,commentor=request.user, comment_item=item)
        new_comment.save()

        return HttpResponseRedirect(reverse("listing", kwargs={"listing_id":listing_id}))

def category(request):
    if request.method == "POST":
        listing_category = request.POST['category']
        category = Category.objects.get(list_category=listing_category)

        listings = category.item_category.exclude(listing_open=False)
        return render(request, "auctions/index.html",{
            "listings": listings,
            "type": f"Listings on: {category}"
        })

    categories = Category.objects.all()
    return render(request, "auctions/category.html", {
        "categories": categories
    })

```