---------- DJANGO TUTORIAL 12 ----------

In this video we'll create the model for the items people will sell on the site. We'll also create the directories for images and static files. We'll also install Pillow, create filler items and create our views.

1. Create the items app and add it to settings

```
python manage.py startapp items
```

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'crispy_forms',
    'users.apps.UsersConfig',
    'pages.apps.PagesConfig',
    'items.apps.ItemsConfig',
]
```

2. Create directory for css files mkdir BobsList/static/css

3. Create media/gallary directory and place default image in it no-image.png

```
# Add this to settings
STATIC_ROOT = ''
STATIC_URL = '/static/'
STATICFILES_DIRS = (os.path.join('static'),)

MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = '/media/'
```

4. Update bobs_list/urls.py

```
from django.contrib import admin
from django.urls import path, include
from django.views.generic.base import TemplateView

from . import settings
from django.contrib.staticfiles.urls import static
from django.contrib.staticfiles.urls import staticfiles_urlpatterns

urlpatterns = [
    path('admin/', admin.site.urls),
    path('users/', include('users.urls')),
    path('users/', include('django.contrib.auth.urls')),
    path('', include('pages.urls')),
    path('items/', include('items.urls')),
]
```

```
urlpatterns += staticfiles_urlpatterns()
urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

4. Install Pillow so we can use images in our model

```
Ctrl + C
exit
pipenv install pillow==7.0.0
pipenv shell
```

5. Create /items/models.py

```
from django.db import models
from django.conf import settings
# Gets us access to our custom user model data
from django.contrib.auth import get_user_model
from django.urls import reverse

class Item(models.Model):
    title = models.CharField(max_length=200)
    description = models.TextField()
    date = models.DateTimeField(auto_now_add=True)
    price = models.FloatField()
    photo = models.ImageField(upload_to="gallary", default='no-image.png')
    # Get cascading list of all stored users
    seller = models.ForeignKey(
    get_user_model(),
    on_delete=models.CASCADE,)

def __str__(self):
    return self.title

# You should add a __str__ and get_absolute_url to every
# model that you make
# This sends a user to the item_detail page after
# data is submitted to the DB
def get_absolute_url(self):
    return reverse('item_detail', args=[str(self.id)])
```

Create URLs, Views and Templates for Edit and Delete

6. Add URLs to items/urls.py

```
from django.urls import path
from .views import(
ItemListView,
ItemUpdateView,
ItemDetailView,
ItemDeleteView,
)

# URLs will take the pattern of items/PrimaryKey/edit
urlpatterns = [
path('<int:pk>/edit/', ItemUpdateView.as_view(), name='item_edit'),
path('<int:pk>/', ItemDetailView.as_view(), name='item_detail'),
path('<int:pk>/delete/', ItemDeleteView.as_view(), name='item_delete'),
path('', ItemListView.as_view(), name='item_list'),
]
```

7. Update items/views.py

```
from django.views.generic import ListView, DetailView
from django.views.generic.edit import UpdateView, DeleteView
from django.urls import reverse_lazy
from .models import Item

class ItemListView(ListView):
  model = Item
  template_name = 'item_list.html'

class ItemDetailView(DetailView):
  model = Item
  template_name = 'item_detail.html'

class ItemUpdateView(UpdateView):
  model = Item
  fields = ('title', 'description', 'price')
  template_name = 'item_edit.html'

class ItemDeleteView(DeleteView):
  model = Item
  template_name = 'item_delete.html'
  success_url = reverse_lazy('item_list')
```

8. Make app show in items/admin

```
from django.contrib import admin
from .models import Item

admin.site.register(Item)
```

9. Create our template in templates/item_list.html
ListView returns an object called object_list which we'll cycle through with a for loop.

```
{% extends 'base.html' %}
{% block title %}Items for Sale{% endblock title %}
{% block content %}
<div class="card-columns">
{% for item in object_list %}
<div class="card">
  <div class="card-body text-center">
    <img class="card-img-top" src="{{ item.photo.url }}" alt="{{ item.title }}">
    <h4 class="card-title">{{ item.title }}</h4>
    <p class="card-text">{{ item.price }}</p>
    <a href="{% url 'item_detail' item.pk %}" class="btn btn-primary">More Info</a>
  </div>
</div>
{% endfor %}
</div>
{% endblock content %}
```

10. Migrate the model

```
python manage.py makemigrations items
python manage.py migrate
```