Construction Project Generator Setup & Usage

📁 File Structure Setup

1. **Create the management command directory structure:**

```bash
# In your Django project root
mkdir -p management
mkdir -p management/commands

# Create __init__.py files
touch management/__init__.py
touch management/commands/__init__.py
```

2. **Place the command file:**
   - Save the generator code as: `management/commands/generate_construction_projects.py`

📦 Install Required Dependencies

```bash
pip install faker
```

🚀 How to Run the Command

Basic Usage (Generate 10 projects)

```bash
python manage.py generate_construction_projects
```

Advanced Options

Generate specific number of projects:

```bash
python manage.py generate_construction_projects --projects 25
```

Generate with tasks and scopes:

```bash
python manage.py generate_construction_projects --projects 15 --with-tasks
```

Generate with budgets and allocations:

```bash
python manage.py generate_construction_projects --projects 20 --with-budgets
```

Generate with notifications:

```bash
python manage.py generate_construction_projects --projects 10 --with-notifications
```

Generate everything:

```bash
python manage.py generate_construction_projects --projects 30 --with-tasks --with-budgets --with-notifications
```

Clear existing data first (⚠️ Be careful!):

```bash
python manage.py generate_construction_projects --clear-existing --projects 20 --with-tasks --with-budgets
```

📊 What Gets Generated

🏢 Project Types

- Residential Building
- Commercial Building
- Industrial Facility
- Infrastructure
- Renovation
- Mixed-Use Development

👥 Users & Profiles

- 2 Operations Managers

- 5 Project Managers
- 8 Engineers
- 3 View Only users

🏗️ Clients

- Major Philippine construction companies (DMCI, Ayala Land, etc.)
- Government entities (DPWH, LGUs)
- Private individuals

📈 Projects (Per Project)

- Realistic project names based on Philippine locations
- Appropriate budgets (₱500K to ₱500M depending on type)
- Realistic timelines (30-730 days)
- Progress tracking
- Status distribution (60% Ongoing, 20% Completed, 15% Planned, 5% Cancelled)

📋 Tasks & Scopes (if --with-tasks)

- Project-specific scopes (e.g., Foundation Work, MEP Installation)
- Detailed tasks within each scope
- Realistic weights and progress tracking
- PM assignments

💰 Budgets (if --with-budgets)

- Category-based budgets (Labor, Materials, Equipment, etc.)
- Fund allocations with realistic distributions
- Budget tracking per scope

🔔 Notifications (if --with-notifications)

- Role-based notifications
- Sample alerts for budget overruns, approvals, inspections
- Read/unread status tracking

🎯 Sample Output

```
Generating 15 construction projects...
  Created project type: Residential Building
  Created project type: Commercial Building
  Created Operations Manager: Maria Santos
  Created Project Manager: Juan Dela Cruz
  Created client: DMCI Holdings
  Created project: The Makati Tower (GC-001)
  Created project: BGC Corporate Center (DC-002)
  Created scopes and tasks for: The Makati Tower
  Created budgets for: The Makati Tower
  Created notification: Project budget exceeded for Metro Manila Tower...
Successfully generated 15 construction projects!
```

⚙️ Customization

You can modify the command by editing these sections:

- **Project Types**: Update `create_project_types()` method
- **Philippine Locations**: Modify the `locations` list in `create_projects()`
- **Budget Ranges**: Adjust `budget_ranges` dictionary
- **Scope Templates**: Update `scope_templates` and `task_templates`
- **Client Companies**: Modify `clients_data` in `create_clients()`

🔍 Verification

After running, check your Django admin or database to verify:

1. **Projects**: Go to Project Profiles
2. **Tasks**: Check Project Scopes and Tasks
3. **Budgets**: Verify Project Budgets and Fund Allocations
4. **Users**: Confirm User Profiles were created
5. **Notifications**: Check notification system

🚨 Important Notes

- **Database**: This adds data to your database - use `--clear-existing` carefully
- **Users**: Command creates sample users with password `password123`
- **Philippine Context**: All data uses Philippine locations, companies, and context

- **Realistic Data**: Budgets, timelines, and progress are based on actual construction patterns
- **Dependencies**: Tasks have logical date dependencies within projects

🛠️ Troubleshooting

Import Errors:

- Ensure your app names match the import statements
- Update import paths if your apps are named differently

Permission Errors:

- Make sure you have database write permissions
- Run with appropriate Django settings

Missing Dependencies:

```bash
pip install faker
```

Memory Issues:

- Generate smaller batches if you have memory constraints
- Use `--projects 10` instead of larger numbers