
WoC
shadow of Syntax
django
HR MANAGER
CareerSphere
– Recruitment platform

Winter Of Code: *Shadows of syntax*

❄️ CATEGORY: Django

🕸️ Mentors: Shamit Gandhi, Kulanjay Chavda, Aayush Goyal

🕸️ Project: CareerSphere : <video link>

🕸️ Prerequisites: Basics of Python

INTRODUCTION

Recruitment platform in Django connecting employers and job seekers through job postings along with application tracking for the job seekers

---

WHY SHOULD YOU TAKE THIS CATEGORY?

- Django helps you go from idea → working web app fast
- Covers core backend concepts like databases, authentication, and APIs
- Beginner-friendly, but powerful enough for production-level projects

---

BENEFITS FOR PARTICIPANTS

Participants will get hands-on experience by working on the CareerSphere project with both basic frontend and backend components. The project is beginner-friendly. By the end, you'll have a working project, clearer backend fundamentals, and the confidence to build and explore more on your own.

TECH STACK USED

- Backend / Core Technology: Django (Python)
- Framework / Library: Django
- Database / Storage: Postgres with Supabase
- Authentication / Security: JWT

EVENT PROJECT DETAILS

🔐 User Authentication

- Sign up as either an Employer or Job Seeker.
- User profiles with basic info (phone, location, bio).

💼 For Employers

- Post job listings with all the details (title, description, requirements, salary range) along with edit and delete job posts.
- View all applications received for your jobs
- Update application status.
- Dashboard to manage all your posted jobs

👔 For Job Seekers

- Browse all available jobs with search and filters
- View detailed job descriptions
- Apply to jobs with an optional cover letter
- Track all your applications and their status

🛠️ Admin Panel

- Pre-built Django admin interface for managing everything
- View and manage users, jobs, and applications
- Bulk actions for updating application statuses

🎃Theme Special Features

Ditch the boring standard forms and job listings! Get creative with a spooky makeover:

- Dark, eerie color schemes.
- Creepy fonts, Spooky icons, Ghost animations.
- Flickering text effects, etc.

Bonus points awarded for:

- Overall spooky aesthetic consistency.
- Creative use of Halloween elements.
- Scary, but not confusing!
- **Code Horror-fication**
- Actually implementing a working Halloween feature (not just renaming stuff)

TENTATIVE EVALUATION CRITERIA

1. Backend: 40%
2. Frontend: 30%
3. Bonus: 30%
    a. Basic renaming (spooky variable names): 5%
    b. Horror-themed UI: 15%
    c. Easter eggs and hidden horror references: 10%

FREQUENTLY ASKED QUESTIONS (FAQs)

Q: Do I need prior Django experience?

A: Nope! This category is designed for beginners. Basic Python knowledge is helpful, but we'll cover Django fundamentals from scratch.

Q: What should I have installed before the workshop?

A:

- Python 3.8+
- A code editor (VS Code recommended)
- Git (optional but useful)
- pip (comes with Python)

Q: Can I use SQLite instead of PostgreSQL/Supabase?

A: Absolutely! SQLite works great for learning. Just comment out the PostgreSQL settings and use Django's default SQLite config. Supabase is just a bonus if you want cloud database experience.

Q: How long will this project take?

A: The basic version can be built in 3-4 hours. With Halloween features and custom UI, expect 6-8 hours total. Take your time and have fun with it!

Halloween Bonus Questions 🎃

Q: What exactly counts as "Halloween themed UI"?

A: Anything that gives spooky vibes! Dark colors, creepy fonts, Halloween icons (pumpkins, ghosts, bats), eerie animations, gothic designs. Put your creative hats on - if it feels haunted, you're on the right track!

Q: Do I need to implement ALL the Halloween feature ideas mentioned?

A: No way! Pick ONE feature that sounds fun and implement it well. Quality over quantity. A working "Zombie Job Resurrection" feature is better than five broken ones.

Q: Can I just rename variables to horror terms without adding new features?

A: You can, but you'll get minimal points (5). To score higher, implement at least one functional Halloween feature with horror-themed naming. Make the code spooky AND useful!

Q: What's the difference between "horror indentation" and regular naming?

A:

- **Regular:** `def delete_application(app_id):`
- **Horror:** `def banish_soul_to_void(cursed_soul_id):`

It's about making your code read like a horror story while still being functional!

Q: Can I use Halloween CSS frameworks or must I code everything?

A: Use whatever you want! Pre-made spooky CSS, Bootstrap with custom overrides, Tailwind with dark themes, or hand-coded CSS. Just make it look Halloween-y!

Q: My Halloween feature breaks the basic functionality. Will I still get points?

A: Nope! Basic features must work perfectly. Halloween stuff is BONUS. If your spooky code breaks job applications, fix the core first, then add the scary stuff.

Q: Can I add horror sound effects or music?

A: That's next level! Creaky door sounds on login, thunder when rejecting applications, spooky background music - absolutely! Just make sure there's a mute button (some people are in public places).

Q: Is there a theme restriction for Halloween UI?

A: No! Classic horror (Dracula, Frankenstein), modern creepy, cute-spooky, cosmic horror, zombie apocalypse - pick your horror subgenre and run with it!

CONCLUSION

This Django category is all about learning by building. Whether you're new to backend development or just getting started with Django, CareerSphere gives you a solid, hands-on introduction to how real-world web applications work. With a beginner-friendly approach, creative Halloween twists, and practical features, this project is a great way to build skills, confidence, and a project you can proudly call your own.