Plan at a glance

Awesome—let's switch from the console "email" to **real SMTP email** to you/Ed. We'll do it safely with a `.env` file so you don't commit secrets, and I'll cover both common cases: **authenticated SMTP** (e.g., Office 365 / Gmail) and **campus relay** (no login).

---

1) Add an env loader (so secrets aren't committed)

In your activated venv (inside `codebase/`):

```
bash

pip install python-dotenv
```

Update `.gitignore` (repo root) if it doesn't already ignore `.env`:

```
bash

.env
```

Create `codebase/.env` (do not commit this file):

```
ini

# --- PICK ONE CONFIG ---

# A) Authenticated SMTP (e.g., Office 365, Gmail, etc.)
EMAIL_BACKEND=django.core.mail.backends.smtp.EmailBackend
EMAIL_HOST=smtp.office365.com
EMAIL_PORT=587
EMAIL_USE_TLS=true
EMAIL_HOST_USER=sklarz@berea.edu
EMAIL_HOST_PASSWORD=YOUR_SMTP_PASSWORD_OR_APP_PASSWORD
DEFAULT_FROM_EMAIL=Printing Services <sklarz@berea.edu>

# Where to send new ticket notifications (comma-separated)
EMAIL_TO=ed@berea.edu,sklarz@berea.edu


# B) Campus relay (no auth) — comment out the A-section above and use this:
# EMAIL_BACKEND=django.core.mail.backends.smtp.EmailBackend
# EMAIL_HOST=campus.smtp.server  # ask IT for the hostname
# EMAIL_PORT=25
# EMAIL_USE_TLS=false
# EMAIL_HOST_USER=
# EMAIL_HOST_PASSWORD=
# DEFAULT_FROM_EMAIL=Printing Services <no-reply@berea.edu>
# EMAIL_TO=ed@berea.edu,sklarz@berea.edu
```

> If you use **Gmail**, you'll need an **App Password** (not your normal password) and may need `EMAIL_PORT=587` + `EMAIL_USE_TLS=true`.

---

2) Load `.env` in Django settings

Edit: `codebase/printer_system/settings.py`
Add near the top (after the existing imports):

```
python


from pathlib import Path
import os
from dotenv import load_dotenv

BASE_DIR = Path(__file__).resolve().parent.parent

# Load .env in development
env_path = BASE_DIR / ".env"
if env_path.exists():
    load_dotenv(env_path)
```

Replace your prior console backend with env-driven SMTP:

```
python


EMAIL_BACKEND = os.getenv("EMAIL_BACKEND", "django.core.mail.backends.console.EmailBackend")
EMAIL_HOST = os.getenv("EMAIL_HOST", "")
EMAIL_PORT = int(os.getenv("EMAIL_PORT", "25"))
EMAIL_USE_TLS = os.getenv("EMAIL_USE_TLS", "false").lower() == "true"
EMAIL_HOST_USER = os.getenv("EMAIL_HOST_USER", "")
EMAIL_HOST_PASSWORD = os.getenv("EMAIL_HOST_PASSWORD", "")
DEFAULT_FROM_EMAIL = os.getenv("DEFAULT_FROM_EMAIL", "no-reply@berea.local")

# Comma-separated list to Python list
EMAIL_TO = [e.strip() for e in os.getenv("EMAIL_TO", "").split(",") if e.strip()]
```

Commit (safe—no secrets are in the repo):

```
bash


git add codebase/printer_system/settings.py .gitignore
git commit -m "feat(email): load SMTP settings from .env; keep secrets out of git"
git push
```

---

3) Send to your real recipients from the view

Edit: `codebase/tickets/views.py`
Use the **`EMAIL_TO`** list from settings:

```
python


from django.conf import settings
# ...
send_mail(subject, body, settings.DEFAULT_FROM_EMAIL, settings.EMAIL_TO or ["you@example.com"])
```

Commit:

```
bash


git add codebase/tickets/views.py
git commit -m "feat(email): send ticket notifications to configured recipients"
git push
```

4) Test the SMTP config

1. **Restart the server** (if running): CTRL+C, then:

```
bash

python manage.py runserver
```

2. Submit a test ticket from **`http://127.0.0.1:8000/p/<qr_token>/`**.
3. Check:
   - Terminal: if there's an SMTP error, Django prints a traceback.
   - Inbox of addresses in **`EMAIL_TO`**: you should receive the message.

If you prefer a quick shell test:

```
bash

python manage.py shell
```

```
python

from django.core.mail import send_mail
from django.conf import settings
send_mail("Test from Django", "Hello from the capstone app.", settings.DEFAULT_FROM_EMAIL,
settings.EMAIL_TO)
exit()
```

---

5) Troubleshooting cheatsheet

- **TLS vs SSL**
  - Most providers: **`PORT=587`** + **`EMAIL_USE_TLS=true`**.
  - Some legacy servers: **`PORT=465`** and use SSL (Django uses TLS flag for STARTTLS; for 465 you may need additional config—try 587 first).
- **Office 365**
  - **`EMAIL_HOST=smtp.office365.com`** + **`PORT=587`** + **`TLS=true`**.
  - Account must be allowed to send SMTP or use authenticated relay.
- **Campus relay**
  - Usually **`PORT=25`**, **no auth**, **no TLS**, and the server whitelists your IP. Clear **`EMAIL_HOST_USER/PASSWORD`**.
- **Gmail**
  - Use an **App Password**, not your normal password; account must allow SMTP.
- **Firewalls**
  - If it silently hangs, outbound SMTP might be blocked on your network. Try from campus IT-approved networks or ask IT to allow it.

6) Optional: fail loudly during setup

While testing, make Django raise if email fails:

```
python

# in views.py just for debugging:
send_mail(subject, body, settings.DEFAULT_FROM_EMAIL, settings.EMAIL_TO, fail_silently=False)
```

(Flip back to **`True`** or omit the arg after you're confident.)

---

7) Update your README (good for your professor & handoff)

Add a section like:

> **Email setup**
>
> 1. Copy **`.env.example`** → **`.env`**.
> 2. Fill SMTP settings (Office 365 or campus relay).
> 3. Run **`python manage.py runserver`**, submit a ticket, and verify email received.

If you want, I can generate a **`.env.example`** file for you now so teammates/admins know what to fill in—just say the word.

❞ Ask ChatGPT