Setting Up Email Configuration (Self-Hosted)

Why Email Configuration is Essential

Email enables critical store functions:

- **Order confirmations** sent to customers after purchase
- **Download delivery** emails with product access
- **Password resets** for customer accounts
- **Account verification** emails during signup

Without email configuration, your store will not send emails to customers.

Generate Encryption Key

Email passwords are encrypted in the database. Generate your encryption key first.

Run this command from your project directory with the virtual environment activated.

```
python manage.py generate_encryption_key
```

Add the generated key to your `.env` file:

```
ENCRYPTION_KEY=your-generated-key-here
```

⚠️ **CRITICAL:** Store this key securely. Losing it means losing access to encrypted passwords.

Configuration Method 1: Environment Variables (.env)

Add to your `.env` file:

```
# Email Configuration
EMAIL_BACKEND=django.core.mail.backends.smtp.EmailBackend
EMAIL_HOST=smtp.example.com
EMAIL_PORT=587
EMAIL_HOST_USER=your-email@example.com
EMAIL_HOST_PASSWORD=your-password
EMAIL_USE_TLS=True
DEFAULT_FROM_EMAIL=noreply@example.com
```

Common SMTP Provider Settings

Gmail:

```
EMAIL_HOST=smtp.gmail.com
EMAIL_PORT=587
EMAIL_USE_TLS=True
EMAIL_HOST_USER=your-email@gmail.com
EMAIL_HOST_PASSWORD=your-app-password  # Not your Gmail password!
```

*Note: Use an **App Password,** not your regular Gmail password.*

Microsoft 365:

```
EMAIL_HOST=smtp.office365.com
EMAIL_PORT=587
EMAIL_USE_TLS=True
EMAIL_HOST_USER=your-email@outlook.com
EMAIL_HOST_PASSWORD=your-password
```

SendGrid:

```
EMAIL_HOST=smtp.sendgrid.net
EMAIL_PORT=587
EMAIL_USE_TLS=True
EMAIL_HOST_USER=apikey
EMAIL_HOST_PASSWORD=your-sendgrid-api-key
```

Mailgun:

```
EMAIL_HOST=smtp.mailgun.org
EMAIL_PORT=587
EMAIL_USE_TLS=True
EMAIL_HOST_USER=your-mailgun-smtp-username
EMAIL_HOST_PASSWORD=your-mailgun-smtp-password
```

NameCheap:

```
EMAIL_HOST=mail.privateemail.com
EMAIL_PORT=587
EMAIL_USE_TLS=True
EMAIL_HOST_USER=your-email-address e.g. support@yourdomain.com
EMAIL_HOST_PASSWORD=your-email-password
```

GoDaddy:

```
EMAIL_HOST=smtpout.secureserver.net
EMAIL_PORT=587
EMAIL_USE_TLS=True
EMAIL_HOST_USER=your-email-address e.g. support@yourdomain.com
EMAIL_HOST_PASSWORD=your-email-password
```

After updating `.env`, restart

```
sudo systemctl restart ebuilder
```

Configuration Method 2: Admin Dashboard

Navigate to: **Admin › Shop › Shop Settings**

Fill in the Email Configuration section:

- **SMTP Host:** Your mail server (e.g., `smtp.gmail.com`)
- **SMTP Port:** Usually `587` for TLS or `465` for SSL
- **Use TLS:** Check this (recommended)
- **SMTP Username:** Your email address
- **SMTP Password:** Your email password or app password
- **From Email Address:** The sender address for outgoing emails

Dashboard settings override .env settings if both are configured.

Choosing Between Methods

Use .env if:

- You want configuration in version control (excluding passwords)
- You manage multiple environments (dev, staging, production)
- You prefer infrastructure-as-code

Use Admin Dashboard if:

- You want to change settings without restarting containers
- Non-technical staff need to update email settings
- You prefer GUI configuration

Use Both: Set fallback credentials in .env, allow dashboard overrides for production.

Testing Your Configuration

Test the connection:

```
python manage.py test_email --to your-email@example.com
```

This command:

1. Validates SMTP connection
2. Tests authentication
3. Sends a test email to verify delivery

Expected output:

```
✓ Connected to SMTP server
✓ Authentication successful
✓ Test email sent to your-email@example.com
```

Troubleshooting

"Authentication failed"

- Verify username/password are correct
- For Gmail: Use an App Password, not your regular password
- For Microsoft 365: Ensure SMTP is enabled for your account

"Connection timed out"

- Check your firewall allows outbound SMTP (port 587/465)
- Verify SMTP host and port are correct
- Try `EMAIL_USE_TLS=False` and `EMAIL_PORT=465` for SSL

"Emails not arriving"

- Check spam/junk folders
- Verify `DEFAULT_FROM_EMAIL` is a valid address
- Ensure your SMTP provider allows sending from that address

"ENCRYPTION_KEY not configured"

- Generate the key: `python manage.py generate_encryption_key`
- Add it to `.env`
- Restart: `sudo systemctl restart ebuilder`

Security Best Practices

1. **Never commit passwords to git** - use environment variables
2. **Use App Passwords** for Gmail/Google Workspace accounts
3. **Restrict SMTP credentials** to sending only (no inbox access)
4. **Monitor email quotas** to avoid service disruptions
5. **Backup your ENCRYPTION_KEY** securely - it cannot be recovered

---

Next Steps

After configuring email:

1. Place a test order to verify customer emails work - ***see docs on setting up Stripe***
2. Test password reset functionality
3. Configure your DNS SPF/DKIM records for better deliverability (optional) - see your email provider for details
4. Set up monitoring for email delivery failures (optional)