SMTP (Simple Mail Transfer Protocol) in Django is a mechanism for sending emails from your Django web application. Django provides a built-in email functionality that abstracts the complexities of working directly with SMTP servers.

Here's how SMTP works in Django:

1. Configuration: In your Django project settings, you specify the SMTP server details such as host, port, username, password, etc. Django uses this information to connect to the SMTP server when sending emails.

2. EmailMessage: In your Django application code, you create an instance of Django's `EmailMessage` class, specifying details like sender, recipient, subject, body, attachments, etc.

3. Sending: Once you've created an `EmailMessage` instance, you call its `send()` method. Django takes care of connecting to the SMTP server using the provided configuration and sends the email.

4. Feedback: After attempting to send the email, Django provides feedback about whether the email was sent successfully or if any errors occurred during the process.

Regarding security features, Django's SMTP support includes several mechanisms to ensure secure email communication:

1. TLS/SSL Encryption: Django supports TLS (Transport Layer Security) and SSL (Secure Sockets Layer) encryption for communication with SMTP servers. This ensures that the email content is encrypted during transmission, protecting it from interception or tampering by unauthorized parties.

2. Authentication: SMTP servers often require authentication before allowing email transmission. Django supports various authentication methods such as plain text, CRAM-MD5, and LOGIN. By providing valid credentials, Django ensures that only authorized users can send emails through the SMTP server.

3. Secure Password Storage: Django provides utilities for securely storing sensitive information like passwords. It's recommended to use Django's built-in `django.core.mail.backends.smtp.EmailBackend` with settings configured to use TLS/SSL and provide authentication credentials securely from settings or environment variables.

4. Configuration Validation: Django validates SMTP configuration settings to ensure they comply with security best practices. For example, it checks whether TLS/SSL encryption is enabled, whether authentication credentials are provided, etc.

By leveraging Django's built-in SMTP support and configuring it correctly with appropriate security measures, you can ensure that your Django application sends emails securely and reliably.