

Roles, Groups & Permissions Mapping

Core Idea

- **Role** = logical label stored on the user (Reader / Editor / Journalist)
- **Group** = Django auth construct that holds permissions
- **Permissions** = what actions a user is allowed to perform

Each **role maps to one group**, and each group has specific permissions.

Roles → Groups Mapping

Role	Django Group
Reader	Readers
Editor	Editors
Journalist	Journalists

When a user is created:

- Their **role field** is set
- They are automatically added to the **matching group**

Permission Mapping per Group

1. Reader Group

Purpose: Consume content only

Permissions

- view_article
- view_newsletter

Restrictions

- Cannot create, edit, or delete articles
- Cannot approve content

2. Editor Group

Purpose: Moderate and publish content

Permissions

- view_article
- change_article
- delete_article
- view_newsletter
- change_newsletter
- delete_newsletter
- approve_article (custom permission)

Restrictions

- Cannot create articles
- Cannot subscribe to content

3. Journalist Group

Purpose: Create content

Permissions

- add_article
- view_article
- change_article
- delete_article
- add_newsletter
- view_newsletter
- change_newsletter
- delete_newsletter

Restrictions

- Cannot approve articles
- Cannot edit other journalists' content

Custom Permission (Required)

approve_article

This permission:

- Is assigned **only to Editors**
- Controls who can approve articles
- Is checked in views/templates before approval actions

Enforcement Points in the App

1. Model Level

- Use Meta.permissions for custom permissions
- Use ForeignKeys to restrict ownership

2. View Level

- @login_required
- @permission_required
- DRF permission classes for API endpoints

3. Template Level

- Hide buttons/actions based on permissions
- Prevent UI access even if URL is known

Automatic Group Assignment Logic

When a user is created:

- If role = Reader → add to Readers group
- If role = Editor → add to Editors group
- If role = Journalist → add to Journalists group

This can be implemented using:

- Django signals (post_save)
- Or overridden save() method