

Blue Ink House — Author & Book Management Platform

Candidate Submission — SDE Intern Assignment

Confidential — For Blue Ink House Recruitment Only

1. Overview & Objective

This project is developed as part of the **SDE Intern Assignment** provided by Blue Ink House. The objective of the platform is to manage **Authors** and **Books** with the core functionalities defined in the assignment:

- Create authors (default status: pending)
- Approve authors
- Add books for approved authors
- List & search books by title, genre, or author
- Clean frontend pages for user flows
- Clean backend architecture, validation, and documentation

This implementation uses **Django**, chosen for its rapid development capability, built-in ORM, form-handling, and support for file uploads.

2. Tech Stack

Layer	Technology
Framework	Django 5.x
Templates	Django Template Engine + Bootstrap
Database	SQLite 3
Media Storage	Local server (MEDIA_ROOT)
Language	Python 3.10+
Testing	Django Test Framework

3. Project Structure

```
blueinkhouse-author-book-platform/
├── blueinkhouse/          # Project Settings + Root URLs
│   ├── authors/           # Author Module
│   │   ├── models.py
│   │   ├── forms.py
│   │   ├── views.py
│   │   └── urls.py
│   └── books/              # Book Module
│       ├── models.py
│       ├── forms.py
│       ├── views.py
│       └── urls.py
└── media/                 # Uploaded images (authors/books)
    ├── templates/
    │   ├── base.html
    │   ├── authors/
    │   └── books/
    └── manage.py
    └── README.md
    └── DOCS.md
```

4. Data Models

4.1 Author Model

Field	Type	Description
id	AutoField	Primary Key
name	CharField(255)	Author name
email	EmailField (unique)	Unique email
bio	TextField	Author biography
profile_image	ImageField	Uploaded profile photo
status	pending / approved	Default: pending
created_at	DateTime	Auto timestamp
number_of_books	property	Count of related books

Business	Rule:
Books can only be added when status = approved.	

4.2 Book Model

Field	Type	Description
id	AutoField	Primary Key
title	CharField(255)	Book title
description	TextField	Book description
genre	CharField(100)	Genre/category
cover_image	ImageField	Cover image
author	FK → Author	Only approved authors
created_at	DateTime	Auto timestamp

5. Frontend Pages

URL	Purpose
/authors/add/	Create new author
/authors/list/	View all authors
/authors/<id>/approve/	Approve author
/books/add/	Add new book
/books/list/	List all books
/books/list/?q=	Search books

Each page uses **Bootstrap UI** for a clean, responsive interface.

6. Backend Business Logic

✓ Author Creation

- Saves with status = *pending*
- Email must be unique
- Image upload handled using Django ImageField

✓ Author Approval

- A dedicated action updates the author's status
- Protected with get_object_or_404()

✓ Book Creation

- Only approved authors appear in the dropdown
Implemented in BookForm.__init__():

```
self.fields['author'].queryset = Author.objects.filter(status="approved")
```

✓ Book Search

Searches across:

- Title
- Genre
- Author name

Implemented using icontains filters.

7. Installation & Setup

Windows

- python -m venv venv
- Venv\Scripts\activate
- pip install -r requirements.txt
- python manage.py migrate
- python manage.py runserver

macOS / Linux

- python3 -m venv venv
- source venv/bin/activate
- pip install -r requirements.txt
- python manage.py migrate
- python manage.py runserver

8. API Endpoints (as per assignment)

Method	Endpoint	Description
POST	/authors/	Create an author
GET	/authors/	List all authors
PATCH	/authors/<id>/approve	Approve an author
POST	/books/	Create book
GET	/books/	List books
GET	/books/search?q=	Search books

Though the project is template-based, the backend logic mirrors these API behaviors.

9. Testing

Tests include:

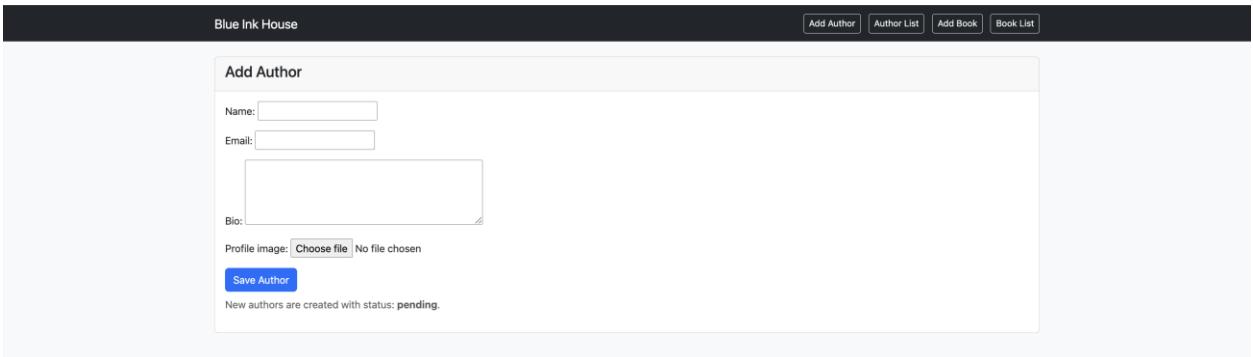
- Author creation
- Duplicate email validation
- Approving an author
- Book creation for approved authors
- Rejection for pending authors
- Search functionality

Run tests:

python manage.py test

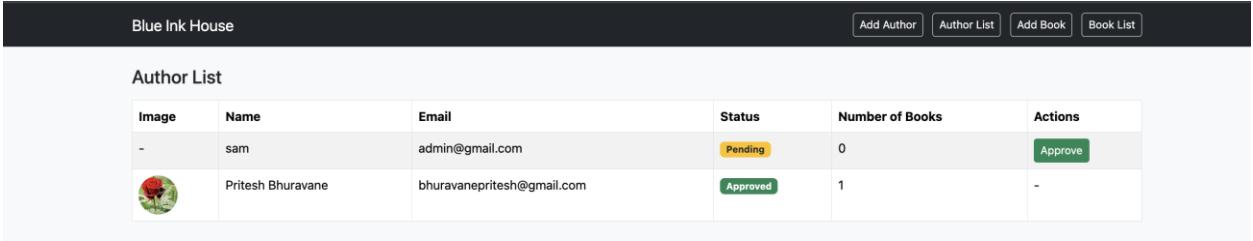
10. Screenshots

- Add Author screen



The screenshot shows the 'Add Author' form. It has fields for Name, Email, and Bio, each with a text input box. Below these is a file input field for Profile image with the placeholder 'Choose file' and the message 'No file chosen'. A 'Save Author' button is at the bottom, and a note below it says 'New authors are created with status: pending.'

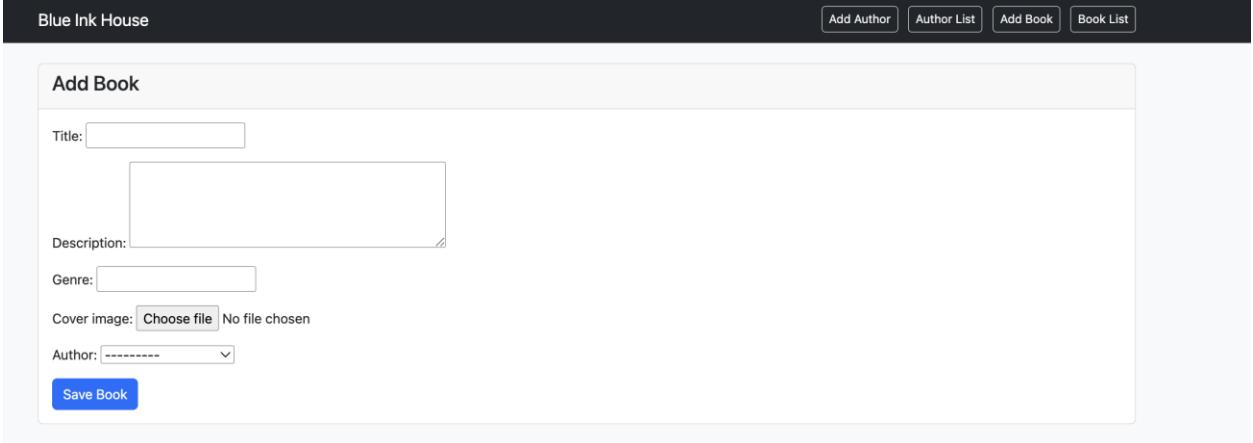
- Approve Author page



The screenshot shows the 'Author List' page. It displays a table with columns: Image, Name, Email, Status, Number of Books, and Actions. There are two rows: one for 'sam' with status 'Pending' and one for 'Pritesh Bhuravane' with status 'Approved'. The 'Actions' column contains a green 'Approve' button for the pending author.

Image	Name	Email	Status	Number of Books	Actions
-	sam	admin@gmail.com	Pending	0	<button>Approve</button>
	Pritesh Bhuravane	bhuravanepritesh@gmail.com	Approved	1	-

- Add Book form



The screenshot shows the 'Add Book' form. It includes fields for Title, Description (with a large text area), Genre, Cover image (with a file input 'Choose file' and 'No file chosen' message), Author (a dropdown menu), and a 'Save Book' button.

- Book List with search filter

Blue Ink House

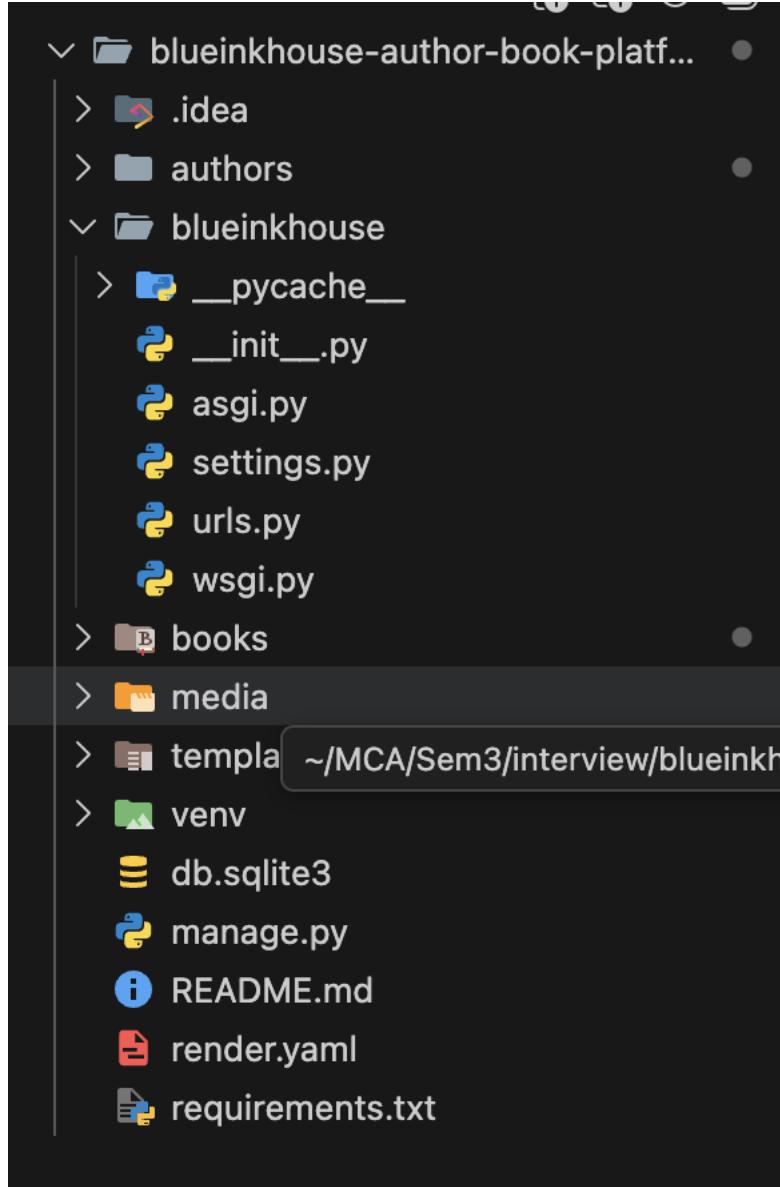
Add Author Author List Add Book Book List

Book List

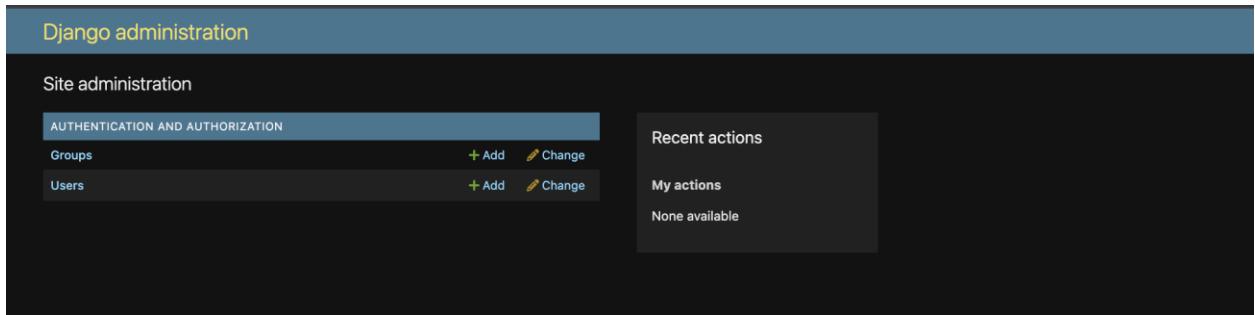
The Q Search

Cover	Title	Author	Genre	Description
	The Quantum Bloom	Pritesh Bhuravane	Science Fiction / Eco-Thriller	In a future where bio-engineered plants provide Earth's primary power source, a...

- Directory structure screenshot



- Admin panel screenshot



11. ☀ Design Decisions

✓ Why Django?

- Rapid development
- ORM for clean database handling
- Built-in form validation
- Image uploads are straightforward
- Easy to maintain for small/medium teams

✓ Clean Architecture

- Each module is isolated into apps
- URLs modular
- Forms manage validation
- Views handle only required logic

✓ Media File Handling

Stored under /media/, configurable for production (e.g., AWS S3).

12. ★ Bonus Features Implemented

- Profile image upload
- Cover image upload
- Search functionality

- Clean Bootstrap UI
- Organized modular structure

13. Recommended Enhancements (Not required but suggested)

- JWT Authentication (for production use)
- Pagination
- Deleting or editing authors/books
- Logging & error tracking
- REST API version using Django REST Framework
- Containerization (Docker)
- Deployment to Render / Railway / AWS

14. Confidentiality Note

This project and its contents are created solely for **Blue Ink House SDE Intern Assignment** and must **not be published or shared publicly**.