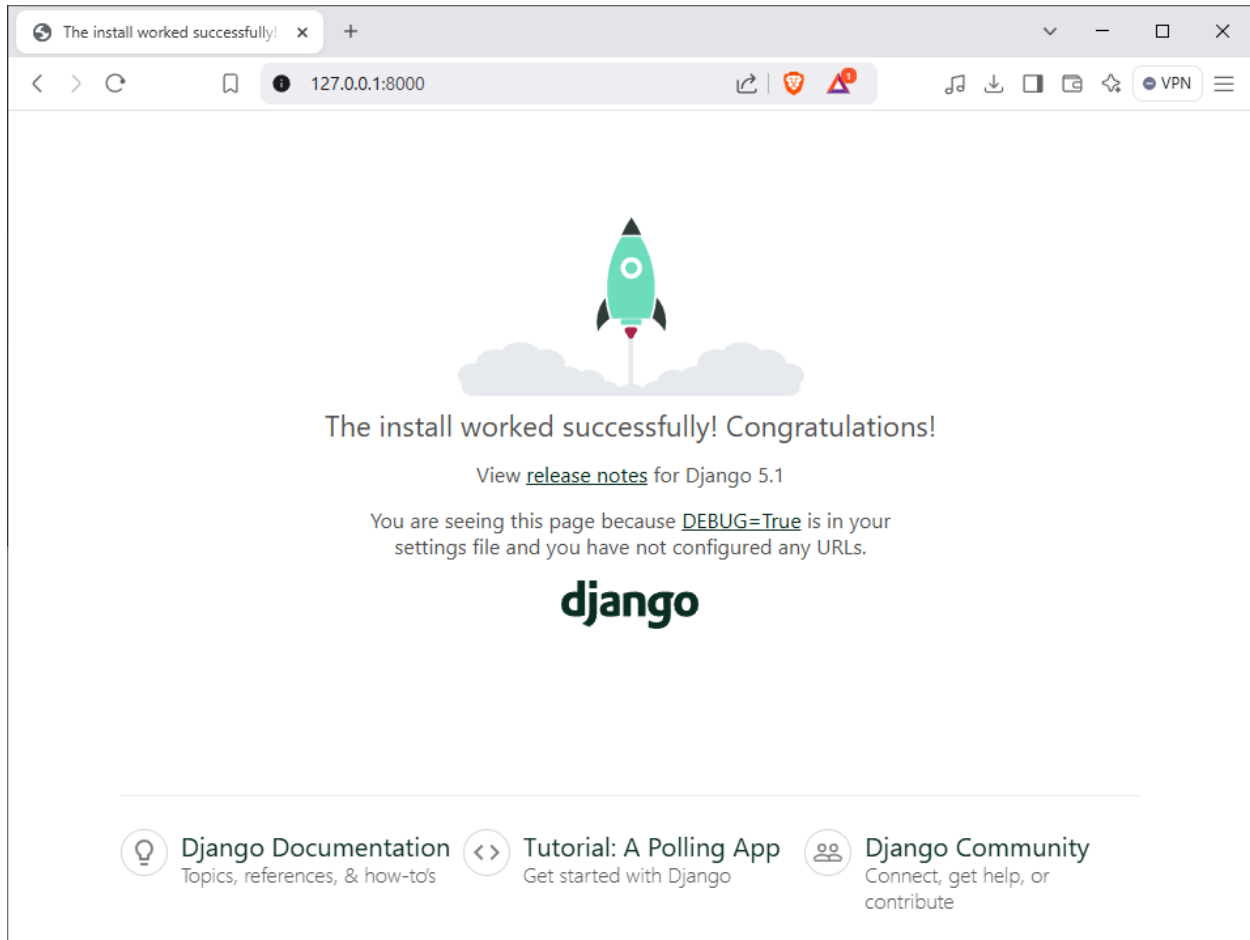**NAME: MANLIGUIS, Sandyr**

COURSE: BSCpE – 2B2

LABORATORY ACTIVITY NO. 1:

RESULT:

FOLLOW-UP QUESTIONS:

1. What is the role of a virtual environment in Django development?
   - It keeps all the required packages isolated, so different projects don't interfere with each other. This helps avoid version conflicts, keeps things organized, and ensures the project runs the same way on any system.
2. What are the advantages of using Django for web development over other frameworks?

- Django makes web development faster and easier with its "batteries-included" approach, meaning it comes with everything you need, like authentication, database management, and security features.

Findings

- A virtual environment keeps project dependencies separate, avoiding conflicts.
- Django is a secure, scalable, and efficient framework with built-in tools for rapid development.
- Its strong community and batteries-included approach make development easier.

Summary

A virtual environment ensures smooth Django development by managing dependencies. Django itself is a powerful, beginner-friendly framework with built-in features that save time and enhance security.

Conclusion

If you want a fast, secure, and reliable web development framework, Django is a solid choice. Pairing it with a virtual environment keeps projects organized and hassle-free.

LABORATORY ACTIVITY NO. 2:

RESULT:
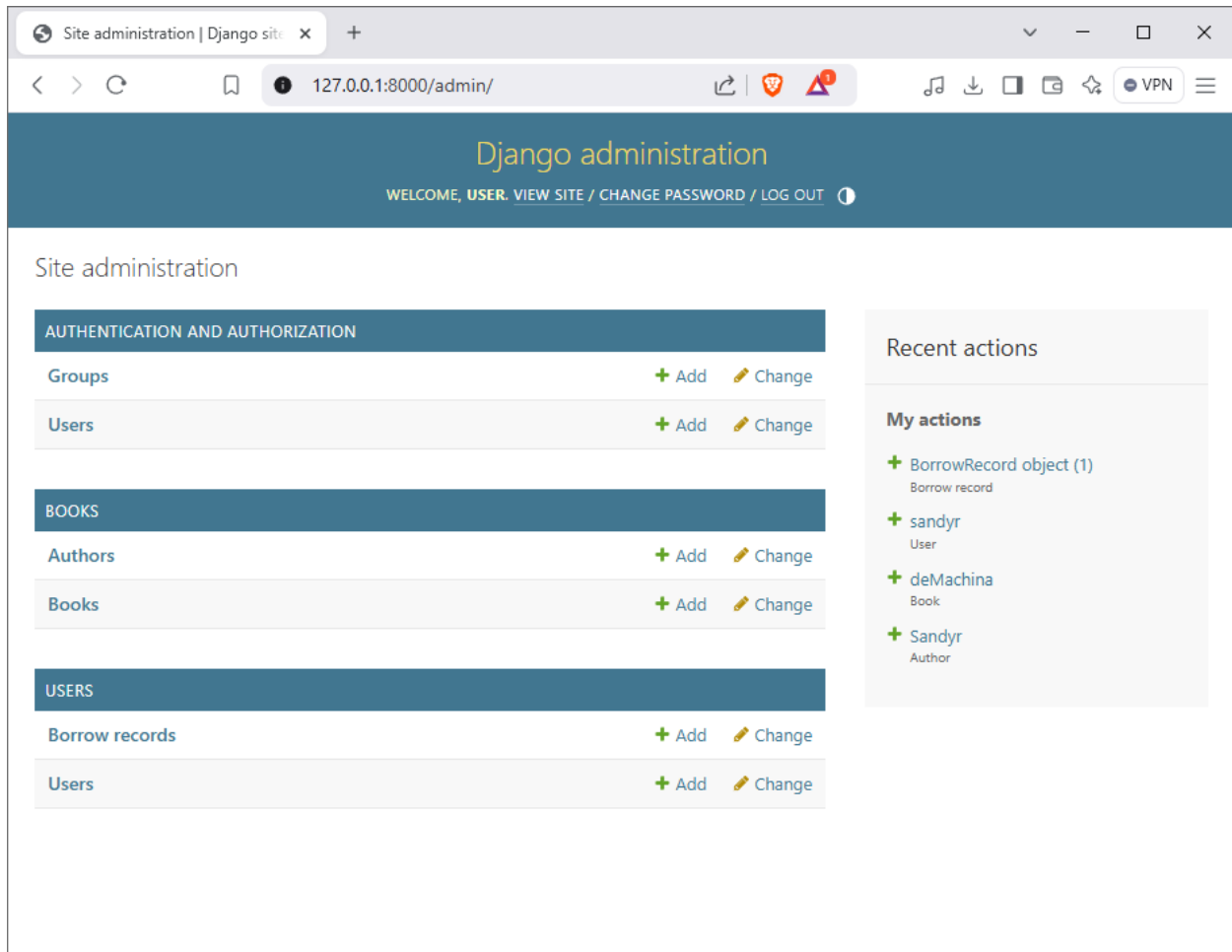
The Books App includes models for Author and Book, where each book is linked to an author. The Users App has models for User and BorrowRecord, tracking which user borrowed which book and when.

Django's ForeignKey connects related models, and migrations apply these changes to the database. Finally, the models are registered in the admin panel for easy management.

FOLLOW-UP QUESTIONS:

1. What is the purpose of using ForeignKey in Django models?
    -   It establishes relationships between tables, linking one model to another, enabling efficient data retrieval.

2. How does Django's ORM simplify database interaction?
   - It allows developers to use Python code instead of SQL queries, making database operations easier and more intuitive.

Findings:

Successfully designed the database schema and implemented models in Django.

Summary:

The activity focused on creating Django models for books, users, and borrowing records, defining relationships using ForeignKey, and applying migrations.

Conclusion:

Django's ORM simplifies database management by providing an intuitive way to define and manipulate data structures.