Welcome to the CoGrammar Django III

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.

CoGrammar

Software Engineering Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** throughout this session, should you wish to ask any follow-up questions.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: [Questions]()

Software Engineering Session Housekeeping cont.

- For all **non-academic questions**, please submit a query:

  www.hyperiondev.com/support

- We would love your **feedback** on lectures: **Feedback on Lectures**

- If you are hearing impaired, please kindly use your computer's function through Google chrome to enable captions.

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

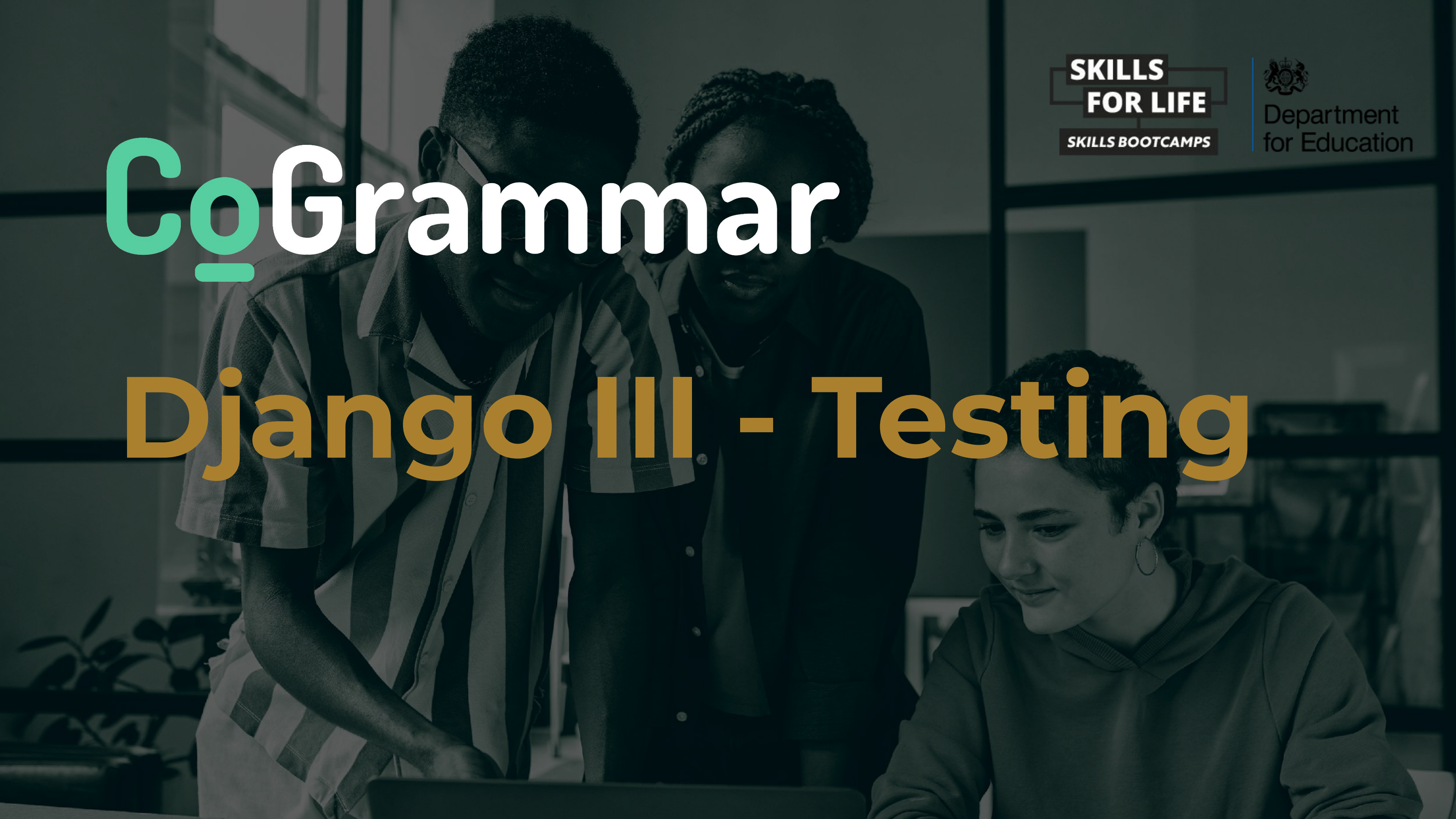
Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Livestorm Tip

- **Livestorm Tip** 👉To allow your attendees to register faster, for example, for small internal meetings, don't require their first and last names. **Attendees without first and last names will then show up in the room as "Anonymous user 1", "Anonymous user 2", etc...** As a Team Member, you'll still be able to see their email address in the room and in your People dashboard.



CoGrammar

SKILLS FOR LIFE

SKILLS BOOTCAMPS

Department for Education

Django III - Testing

Learning Objectives

- Explain the importance of testing in software development and specifically in Django applications.
- Set up and configure Django’s built-in testing framework.
- Write unit tests for Django models, views, and forms.
- Use Django’s TestCase class to create and run tests.
- Apply best practices for writing effective and maintainable tests.

CoGrammar

Poll

1. Which of the following is NOT a benefit of Django?

    i. It has built-in security features.
    ii. It requires writing SQL queries manually for all database operations.
    iii. It provides an admin interface for managing data.

Poll

2. What does a Django database migration do?

a. It transfers data from one database to another.

b. It applies changes in models to the database schema.

c. It removes all data from the database.

CoGrammar

Types of Software Testing (Quick Overview)

- **Unit Testing** – Tests individual components (e.g., models, views)
- **Integration Testing** – Verifies how different parts work together
- **System Testing** – Tests the entire application as a whole
- **Acceptance Testing** – Ensures the app meets user requirements
- **Compliance Testing** – Checks legal/security standards

CoGrammar

Django's Built-in Testing Framework

- Building on unittest knowledge
- Django's TestCase vs Python's **unittest.TestCase**
- Test directory structure and naming conventions
- Running tests with `python manage.py test`
- Test database creation and isolation

What to Test in Django?

CoGrammar

Models

- Field Validation:
  - Data types, max lengths, default values.
  - Example: Verifying that a `CharField` rejects inputs exceeding `max_length`.
- Model Methods:
  - Custom methods that perform calculations or logic.
  - Example: A method that calculates the total price of an order.
- Relationships and Constraints:
  - One-to-one, one-to-many, many-to-many relationships.
  - Cascading deletes and updates.

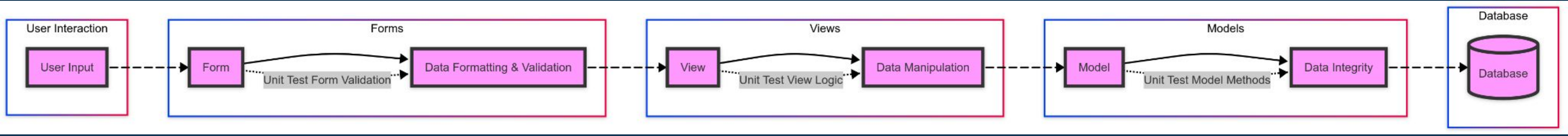
Views

- HTTP Response Codes: Use correct status codes (e.g., 200 OK, 404 Not Found, 302 Redirect). Example: A protected view should return 403 Forbidden for unauthorized users.
- Templates & Context: Views must render the right template with the correct data. Example: A list view should pass the proper queryset to the template.
- URL Routing: Ensure URLs map to the correct views. Example: /profile/ should route to profile_view.
- Authentication & Permissions: Enforce access control as needed. Example: An admin-only view should restrict access to admins.

Templates

- Data Rendering:
  - Variables from context are displayed correctly.
  - Example: The user's name appears correctly on the dashboard.
- Template Logic:
  - Control structures (loops, conditionals) work as intended.
  - Example: A for-loop correctly iterates over a list of items.
- Static Content and Assets:
  - Links to CSS, JavaScript, and images are correct.
  - Example: The site logo displays properly on all pages.
- Internationalization and Localization:
  - Templates render correctly in different languages if supported.
  - Example: Date and currency formats adjust based on locale.

Big Picture



CoGrammar

Best Practices in Django Testing

CoGrammar

Django Testing

- Test-Driven Development (TDD):
    - Write tests before writing the code they validate.
    - Benefits include clearer requirements and cleaner code.
- Writing Effective Tests:
    - **Isolated:** Tests should not depend on each other.
    - **Repeatable:** Tests should produce the same results every time.
    - **Descriptive:** Test names should clearly state what they are testing.
- Common Pitfalls to Avoid:
    - Over-mocking can lead to tests that pass but don't reflect reality.
    - Neglecting edge cases and error conditions.

Final Assessment

CoGrammar

Poll

1. Which command initializes Django's built-in testing framework?

   a. python manage.py runtests
   b. python manage.py test
   c. pytest django

CoGrammar

Poll

2. How do you test if a Django view returns a 200 status code?

    a. self.assertEqual(response.status_code, 200)

    b. self.assertTrue(response.status_code)

    c. self.assertContains(response, 200)

CoGrammar

Lesson Conclusion and Recap

CoGrammar

Lesson Conclusion and Recap

- **Why Testing Matters**
    - Ensures code works as expected and catches bugs early.
    - Critical for reliable Django apps.
- **Django's Testing Framework**
    - Built-in tools for writing and running tests.
    - Test database created automatically.
- **Writing Unit Tests**
    - Models: Test database interactions.
    - Views: Check responses and request handling.
    - Forms: Validate data and error handling.
- **Best Practices**
    - One test = one function
    - Use Django's TestCase class.
    - Write small, focused, and descriptive tests.
    - Keep tests independent and maintainable.

CoGrammar

References

- Testing in Django
- Django Tutorial in Visual Studio Code

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar