# Welcome to this CoGrammar tutorial:

# Django Foundations

The session will start shortly...

Questions? Drop them in the chat.

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
- Report a **safeguarding** incident: www.hyperiondev.com/safeguardreporting
- We would love your **feedback** on lectures: Feedback on Lectures

# Enhancing Accessibility: Activate Browser Captions

## Why Enable Browser Captions?

- Captions provide **real-time text for spoken content**, ensuring inclusivity.
- Ideal for individuals in noisy or quiet environments or for those with **hearing impairments**.

## How to Activate Captions:

1. YouTube or Video Players:
   - Look for the CC (Closed Captions) icon and click to enable.
2. Browser Settings:
   - Google Chrome: Go to *Settings > Accessibility > Live Captions* and toggle ON.
   - Edge: Enable captions in *Settings > Accessibility*.

# 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

# Learning Outcomes

- Define what a web framework is.
- Describe Django
- Explain the benefits of Django
- Describe the MVT structure of Django
- Explain what a model is in Django.
- Create models for your Django project
- Execute database migrations

# Learning Outcomes

- Explain what a view is in Django.
- Route views to specific urls.
- Create views for your Django project.
- Explain what a template is in Django.
- Create templates for your Django project.

SKILLS FOR LIFE

SKILLS BOOTCAMPS

Department for Education

CoGrammar

# Django

# What is a Web Framework?

- A web framework is a collection of tools, libraries, and best practices designed to simplify the process of building and maintaining web applications.
- A web framework allows developers to focus on writing application-specific code rather than reinventing the wheel for every new project.
- Promotes code reuse.

CoGrammar

# What is Django?

- High-level, open-source web framework for Python.
- Used for developing secure and scalable web applications.

# Why use Django?

- Designed to help developers take applications from concept to completion as quickly as possible.
- Architecture is highly scalable, allowing it to handle large amounts of traffic and data efficiently.
- Includes numerous security features out of the box, which help developers protect their applications from common vulnerabilities.

# Why use Django?

- ORM allows developers to interact with the database using Python code instead of SQL.
- Django’s templating engine allows developers to create dynamic HTML pages.

# Model-View-Template (MVT) Architecture

- Variation of Model-View-controller architecture
- Three main components
  - **Model**: Represents the business logic and data structure of the application.
  - **View**: Handles the interaction between the user and the application, managing the presentation logic.
  - **Template**: Deals with the presentation layer, defining the structure and appearance of the HTML content.

# Models

- Models serve as the blueprint for your database schema.
- Each model is a Python class/table that subclasses django.db.models.Model, and its attributes represent the fields/columns of the database table.
- Models are crucial for defining the structure of your data, including field types, default values, and validation rules.

# Views

- Views handle the logic of processing user requests and returning responses.
- Views define the behaviour of our URL patterns.
- Views are Python functions or class methods, that takes a web request and returns a web response.
- They interact with models to retrieve or update data.

# Templates

- Templates define the structure and layout of the HTML pages.
- Templates can incorporate dynamic data, using placeholders and template tags.
- Templates receive data from views through context dictionaries.
- Templates are stored in the templates directory.

# Django: Naming Conventions

- Project and Application Names:
  - Use lowercase letters and underscores to separate words.
  - Hyphens are not recommended because they can cause issues in import statements.
  - Choose a name that clearly describes the purpose of the project.

# Django: Naming Conventions

- Model Names:
  - Models are represented as classes and should use the PascalCase or also known as CamelCase.
  - Model names should generally be singular, as each instance of the model represents a single record in the database.
  - Example: UserProfile

# Django: Naming Conventions

- View Names:
  - For Function-Based Views, use lowercase with underscores to separate words.
  - Function names should clearly describe the action or response of the view.
  - For Class-Based Views (Where methods are used), use PascalCase for class names, and the class name should describe what the view does or what it represents.
  - Examples: def register_user(request): for user registration, class ProductListView(ListView): for listing products.

# Django: Naming Conventions

- Template Names:
  - Use lowercase letters with underscores to separate words.
  - Template names should be descriptive of the view they are associated with.
  - Example: blog_detail.html, user_profile.html.
  - Organise templates into directories that mirror your application's structure.
  - Create reusable template components (partials) and place them in a partials or includes directory, ie. header.html, footer.html

# Let’s get coding!

CoGrammar

# Let’s take a short break

CoGrammar

# Final Assessment

# Polls

- *Refer to the polls section to vote for you option.*

1. What is the correct command to create a new Django Project?
    a. django-admin startproject projectname
    b. django-admin startapp projectname
    c. django-admin createproject projectname
    d. django-admin createapp projectname

# Polls

- *Refer to the polls section to vote for you option.*

2. What is the correct command to create a new Django Application?

    a. django-admin startproject appname

    b. django-admin startapp appname

    c. django-admin createproject appname

    d. django-admin createapp appname

# Polls

- *Refer to the polls section to vote for you option.*

3. What is the correct command to run a Django Project?

    a. python manage.py startproject

    b. python manage.py startapp

    c. python manage.py runproject

    d. python manage.py runserver

CoGrammar

# Polls

- *Refer to the polls section to vote for you option.*

4. **What is the correct command to run a database migration?**
    a. python manage.py migrate
    b. python manage.py runmigrations
    c. python manage.py startmigration
    d. python manage.py makemigrations

# Lesson Conclusion

# Django Recap

- Web Framework
- Django
- Models
- Views
- Templates

# Questions and Answers

CoGrammar

# Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar