Welcome to this

CoGrammar

Task Walkthrough: Task 15

The session will start shortly...

Questions? Drop them in the chat.

CoGrammar

SKILLS FOR LIFE

SKILLS BOOTCAMPS

Department for Education

CoGrammar

Task Walkthrough: Software Design

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

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

- Identify and explain the purpose of a use case diagram, sequence diagram, and class diagram.
- Design and construct software design diagrams to show interactions between components in a system, user interactions, and the static structure of the system's classes.
- Apply the CRUD functionality and MVC pattern to create a system design.
- Transfer learnings to complete the Software Design task.

Polls

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

1. **Have you completed the Software Design task?**

    a) Yes

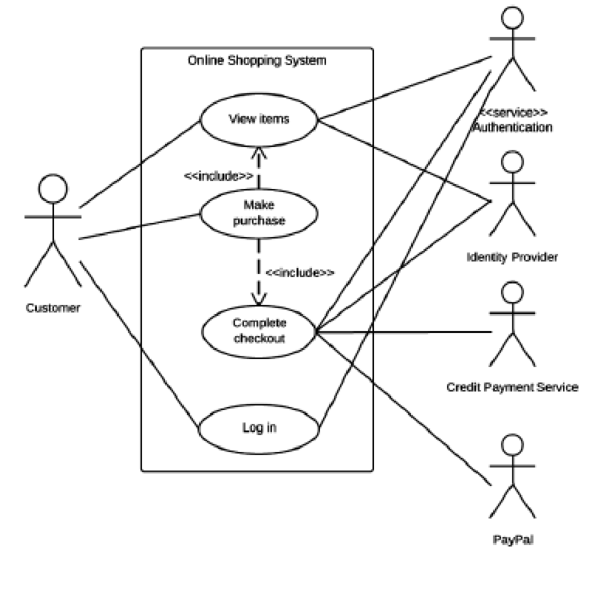
    b) No, but I attempted and I’m stuck

    c) No, but I am almost done

    d) No, I have not yet started

Software Design

- Use Case Diagram: Shows the system's functionality from the perspective of the user. It focuses on what the system does by listing the use cases (features) and actors (users).

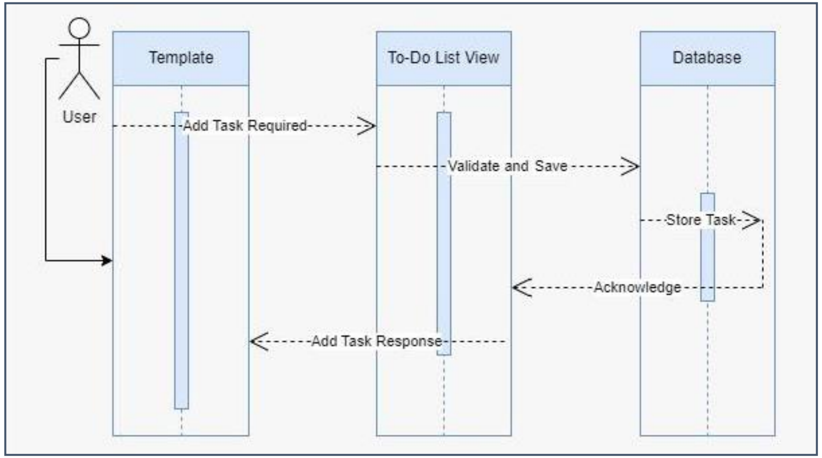


UML use case analysis diagram (Lucidchart, n.d.)

CoGrammar

Software Design

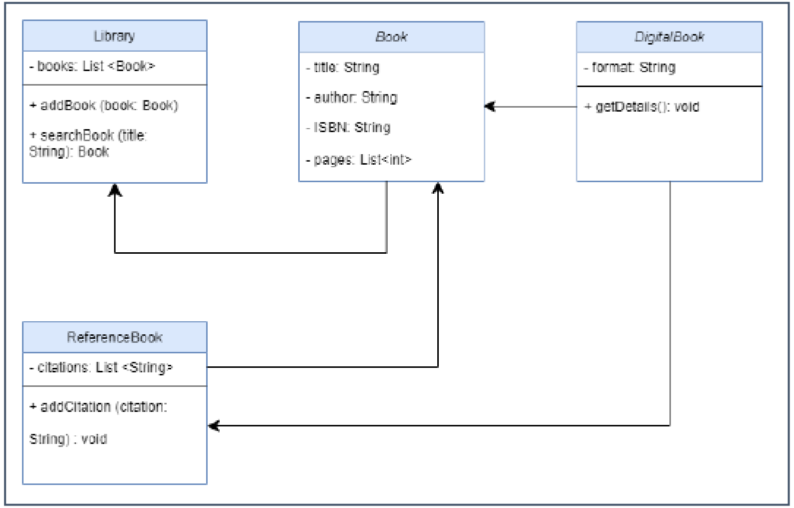
- Sequence Diagram: Shows the sequence of messages exchanged between objects to perform a specific operation. Focuses on how the system behaves over time.



CoGrammar

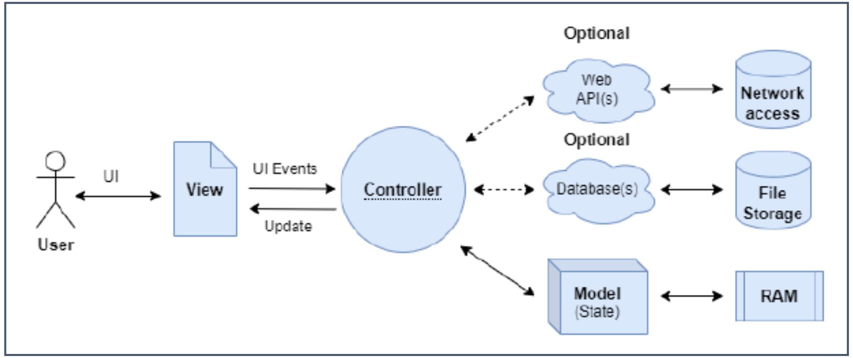
Software Design

- Class Diagram: Shows the static structure of the system, focusing on the classes, their attributes, methods, and relationships.



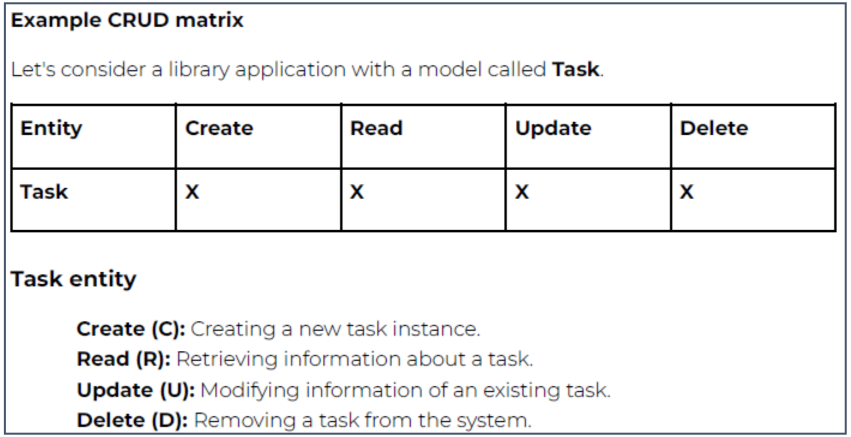
Software Design

- MVC Components:
  - Model: Represents the data and logic
  - View: Displays the user interface
  - Controller: Handles user input and updates the model and view



Software Design

- CRUD Matrices:
  - Purpose: Identifying the operations (Create, Read, Update, Delete) needed for each entity



Example CRUD matrix

Let's consider a library application with a model called **Task**.

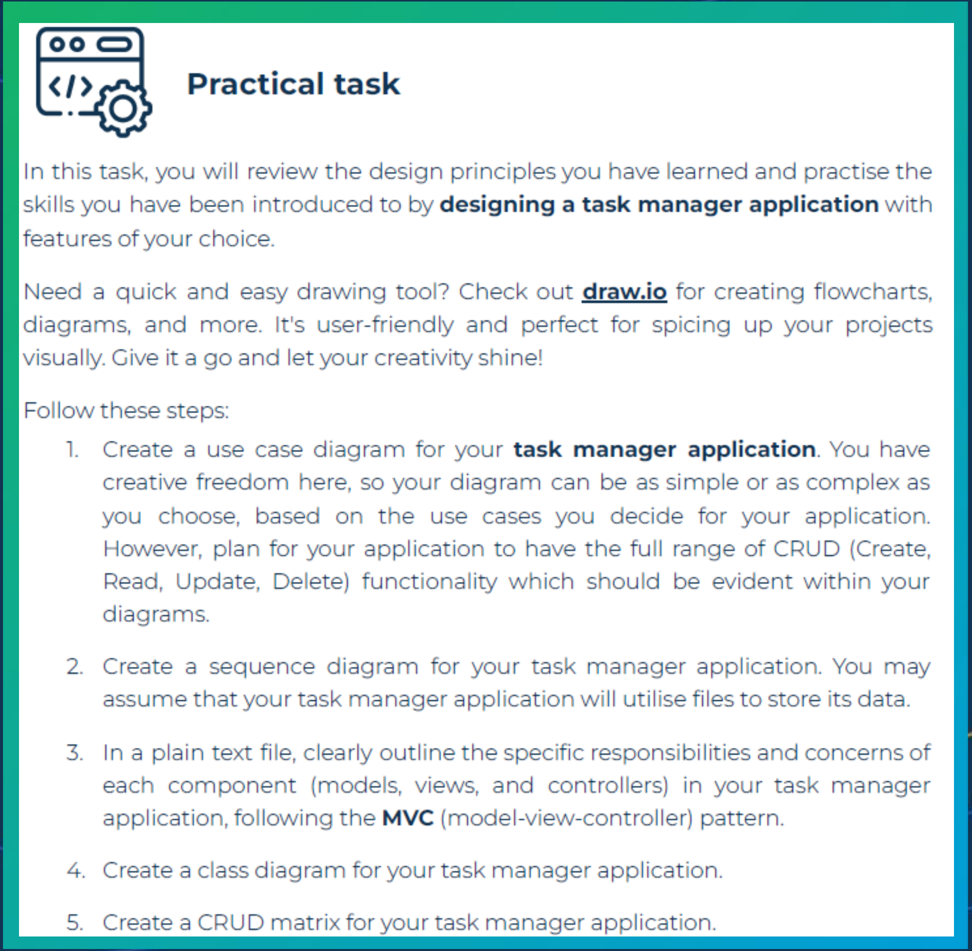
Entity	**Create**	**Read**	**Update**	**Delete**
Task	**X**	**X**	**X**	**X**

Task entity

Create (C): Creating a new task instance.
Read (R): Retrieving information about a task.
Update (U): Modifying information of an existing task.
Delete (D): Removing a task from the system.

Task Walkthrough: Practical Task

CoGrammar



Practical task

In this task, you will review the design principles you have learned and practise the skills you have been introduced to by **designing a task manager application** with features of your choice.

Need a quick and easy drawing tool? Check out **draw.io** for creating flowcharts, diagrams, and more. It's user-friendly and perfect for spicing up your projects visually. Give it a go and let your creativity shine!

Follow these steps:

1. Create a use case diagram for your **task manager application**. You have creative freedom here, so your diagram can be as simple or as complex as you choose, based on the use cases you decide for your application. However, plan for your application to have the full range of CRUD (Create, Read, Update, Delete) functionality which should be evident within your diagrams.

2. Create a sequence diagram for your task manager application. You may assume that your task manager application will utilise files to store its data.

3. In a plain text file, clearly outline the specific responsibilities and concerns of each component (models, views, and controllers) in your task manager application, following the **MVC** (model-view-controller) pattern.

4. Create a class diagram for your task manager application.

5. Create a CRUD matrix for your task manager application.

Task Manager App: Ideas

A **Work Task Manager** application helps users organise, track, and manage tasks efficiently. Here are some ideas for features and functionalities you can include:

- **Task Creation & Management**
  - Create, update, and delete tasks
  - Assign tasks to users
  - Set due dates and priorities (Low, Medium, High, Urgent)
  - Add descriptions and attachments
  - Users can add comments under tasks

Task Manager App: Ideas

- **Task Status**
  - "To-Do", "In Progress", "Completed", "Blocked“
- **Task Categories & Tags**
  - Organise tasks by category (e.g., Development, Marketing, HR)
- **Recurring Tasks**
  - Daily, weekly, or monthly task recurrence
  - Automate reminders for repeated tasks

Task Manager App: Ideas

- User Roles & Permissions
  - Admins, Managers, Team Members, and View-Only roles
  - Restrict who can edit/delete tasks
- Task Reports & Analytics
  - Visual charts for completed vs pending tasks
  - Productivity reports for users
- Time Tracking
  - Log time spent on tasks

Task Manager App: Ideas

- Task Dependencies
  - Link tasks that depend on each other
  - "Task B cannot start until Task A is completed"

Summary of the Task Steps

1. Use Case Diagram:
   - Include full range of CRUD functionality
2. Sequence Diagram:
   - Utilising files to store data
3. MVC Components:
   - Text file outlining the responsibilities and concerns
   - NOTE: You are not asked to draw a diagram (Refer: Pg 11-12)
4. Class Diagram
5. CRUD Matrix

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar