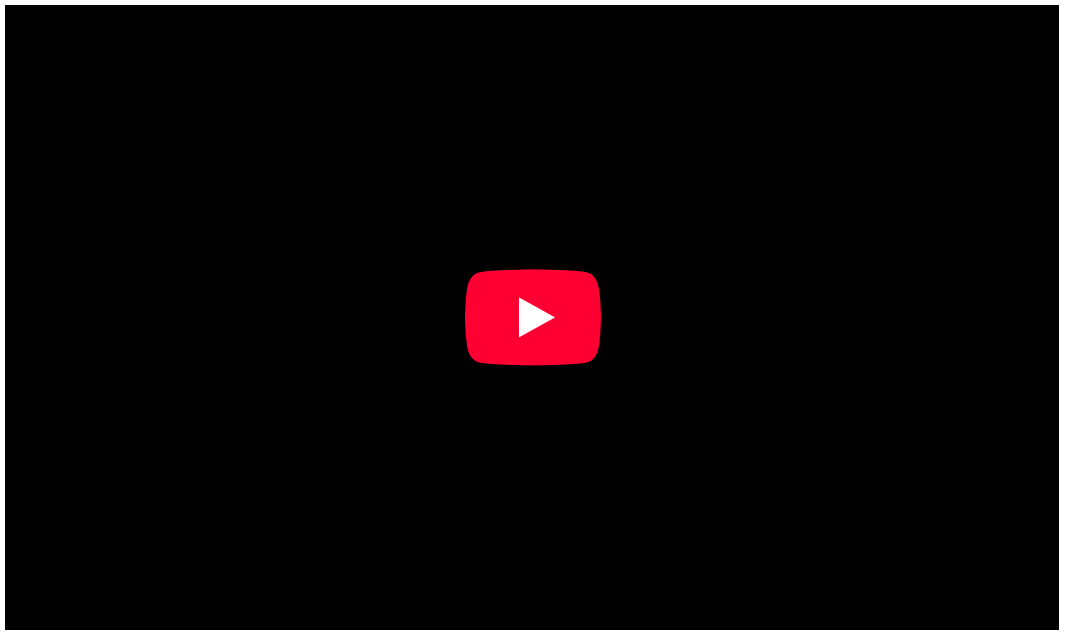
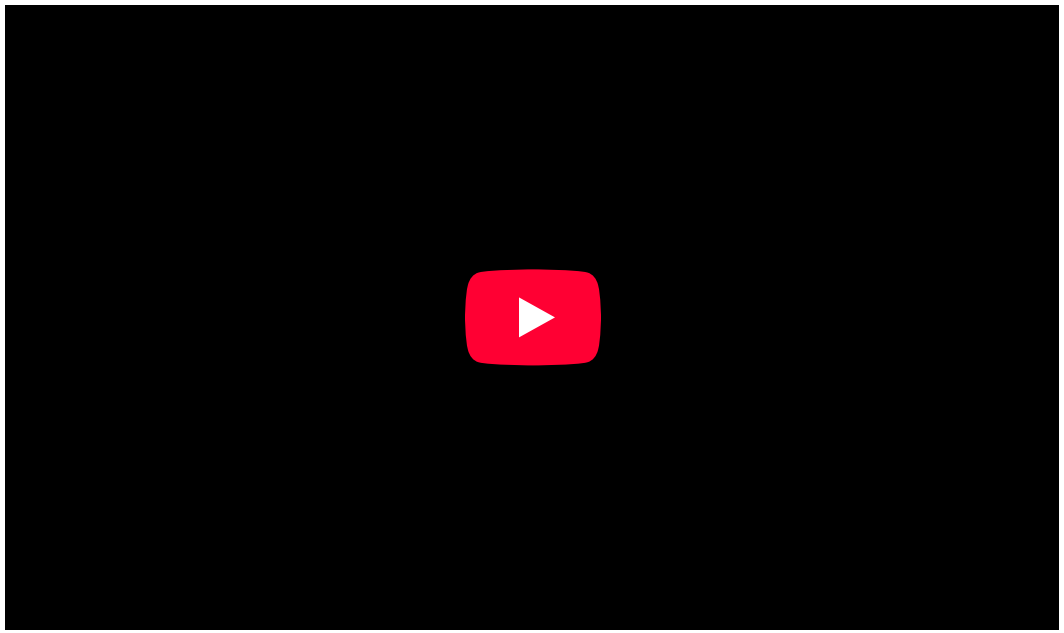
# 🎓 Create your first multiplayer game with VRChat + Unity

Start

## 🎮 From playing games...





...to creating worlds 🌍

Next

# Who is this course for?

## 🎯 This course is designed for:

- Students aged 15–22
- Curious about:
  - Technology
  - Video games
  - Digital creation
  - Programming or engineering
- With basic computer skills
- No prior experience in Unity or VRChat required

## ✔ Ideal for students who:

- ☒ Enjoy learning by doing
- ☒ Like solving challenges
- ☒ Want to understand how digital systems work
- ☒ Are exploring future academic or professional paths in technology

Next

# Learning Objectives + Deliverables

## 5 skill categories

### 🔹 Digital & Technical Skills

- Programming logic (C# fundamentals)
- Game engines (Unity)
- Multiplayer systems
- Virtual Reality environments
- Debugging and testing

### 🧠 Cognitive Skills

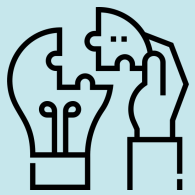
- Logical thinking
- Problem decomposition
- Systems thinking
- Cause–effect analysis

### 🎨 Creative Skills

- Game mechanics design
- User experience (UX)
- Visual and interactive design
- Prototyping ideas

### 🤝 Social & Collaborative Skills

- Teamwork
- Communication
- Peer feedback
- Collaborative problem-solving



### 🔄 Transversal Skills

- Autonomy
- Perseverance
- Adaptability
- Learning from errors
- Project ownership

# A guided learning journey

## Phases:

### Phase 1: Foundations

- Lesson 1–2:
  - Introduction to VRChat & Unity
  - Development environment setup
  - Building the game arena

### Phase 3: Core Gameplay

- Lesson 5–7:
  - Ball pickup & physics
  - Networking & synchronization
  - Scoring system
  - Player-to-player interaction

### 🎮 A complete Multiplayer VRChat game

- Designed
- Programmed
- Tested
- Played by students

### Phase 2: Game Structure

- Lesson 3–4:
  - Game state machine
  - Game HUD
  - Team formation & player identification

### Phase 4: Game Loop

- Lesson 8–9:
  - Game timer & end-game logic
  - Reset systems
  - Spectator mode & late joiners

# Learning Objectives + Deliverables

## ◆ Learning by Doing

- Students learn through **hands-on practice**
- Every concept is applied immediately
- No passive learning

## ◆ Problem-Based Learning

- Lessons are structured as **real technical challenges**
- Students solve concrete problems:
  - *"How do we synchronize players?"*
  - *"How do we prevent game-breaking bugs?"*
- Encourages analytical thinking

## ◆ Iterative Development

- Build → Test → Fix → Improve
- Mistakes are part of the learning process
- Students see visible progress every session

## ◆ Project-Based Learning

- All lessons contribute to a **single evolving project**
- Knowledge is contextualized
- Students understand why each concept matters

## ◆ Collaborative Learning

- Peer discussion and testing
- Shared debugging
- Team-based thinking inspired by real tech teams

# One course. Three challenge levels.

🎯 Students choose how deep they want to go — and can change level at any time.

## 🟤 LEVEL 1: Advanced Challenge

For students with prior Unity or programming experience

+info

## 🟤 LEVEL 2: Guided Challenge

For students with average programming skills

+info

## ☒ LEVEL 3: Step-by-Step Support

For students new to programming or Unity

+info

# ☒ Step-by-Step Support

**Each step is developed together with the instructor**

- Exact actions are shown:
  - Where to click
  - What to write
  - How to test
- No student is left behind

**Focus:**

- Learning foundations
- Reducing frustration
- Building confidence through support

☒ **Comparable to scaffolded learning in introductory CS courses**

# Advance Challenge

**At the start of each lesson**, students receive:

- A high-level description of the features to implement
- Clear goals and expected behavior
- No step-by-step guidance
- Students design and implement their own solutions

**Focus:**

- Autonomy
- System design
- Creative problem-solving
- Advanced technical thinking

☒ **Comparable to challenge-based tasks used in advanced engineering programs**

# Guided Challenge

**Each lesson is divided into clear steps**

- For each step:
  - The goal is explained
  - The expected result is shown
- Students decide how to implement the solution

**Focus:**

- Understanding concepts
- Applying logic independently
- Building confidence through guided autonomy

☒ **Comparable to standard university-level lab exercises**

# Instant support, when students need it (1/2)

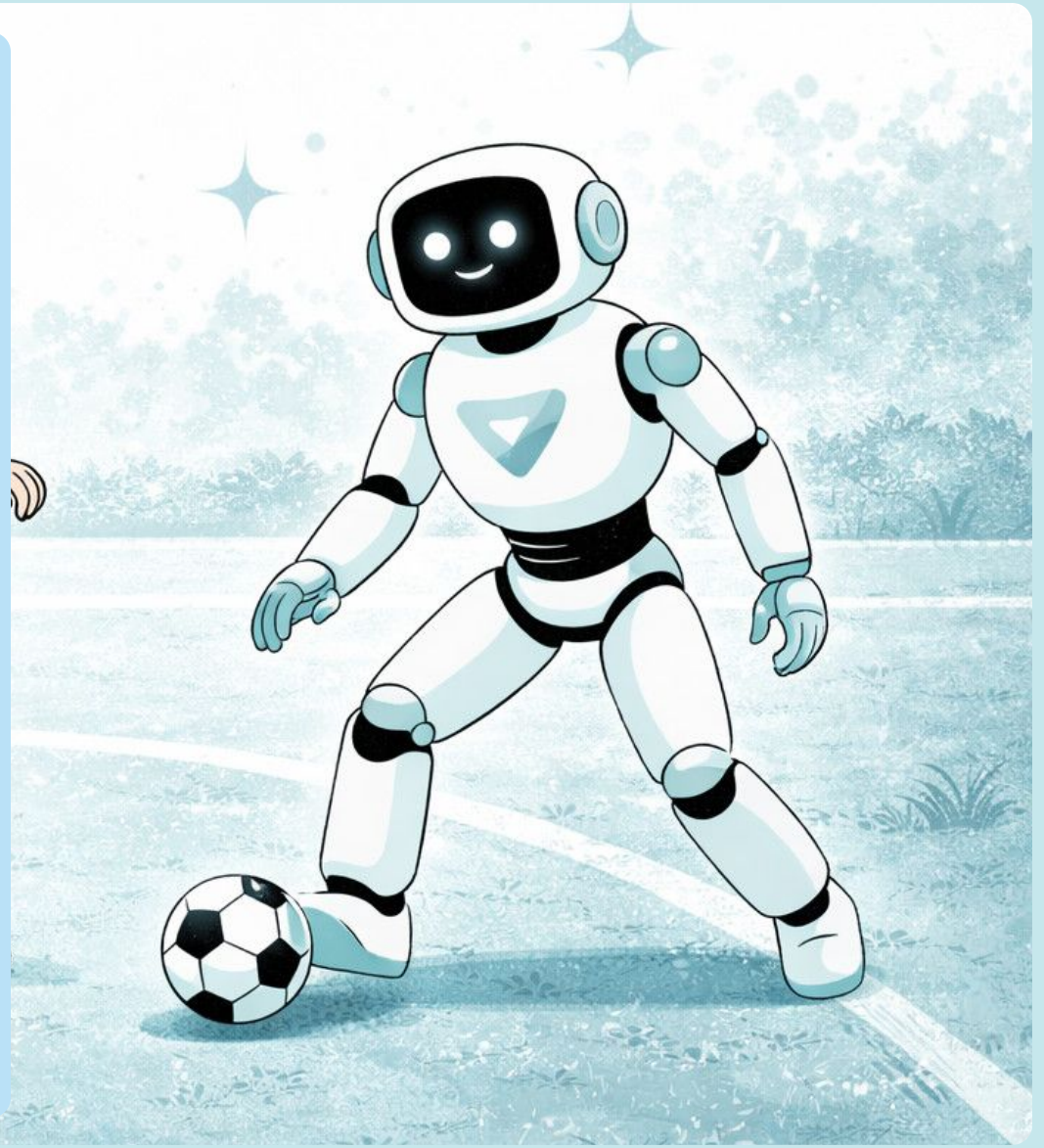
## Guided help without giving away the solution

- **Prepared Learning Context**
  - For each lesson, students receive a prepared prompt
  - The prompt contains:
    - Lesson objectives
    - Technical context
    - Current step description
    - Constraints and rules
  - Students don’t start from a blank question
- **AI as a Learning Assistant**
  - Students can ask:
    - Conceptual questions
    - Clarification questions
    - Debugging guidance
  - The assistant:
    - Explains why things work
    - Suggests approaches
    - Encourages reasoning

🚫 It does not replace learning

🚫 It does not provide copy-paste answers by default

Next



# Instant support, when students need it (2/2)

- **Immediate, Personalized Support**
  - Available at any moment
  - Adapts to the student's level
  - Reduces frustration and blocking points
  - Encourages autonomy instead of dependency

- **Responsible Use of AI**
  - AI is used as:
    - A tutor
    - A guide
    - A thinking partner
  - Not as:
    - An automatic solution generator
    - A replacement for effort

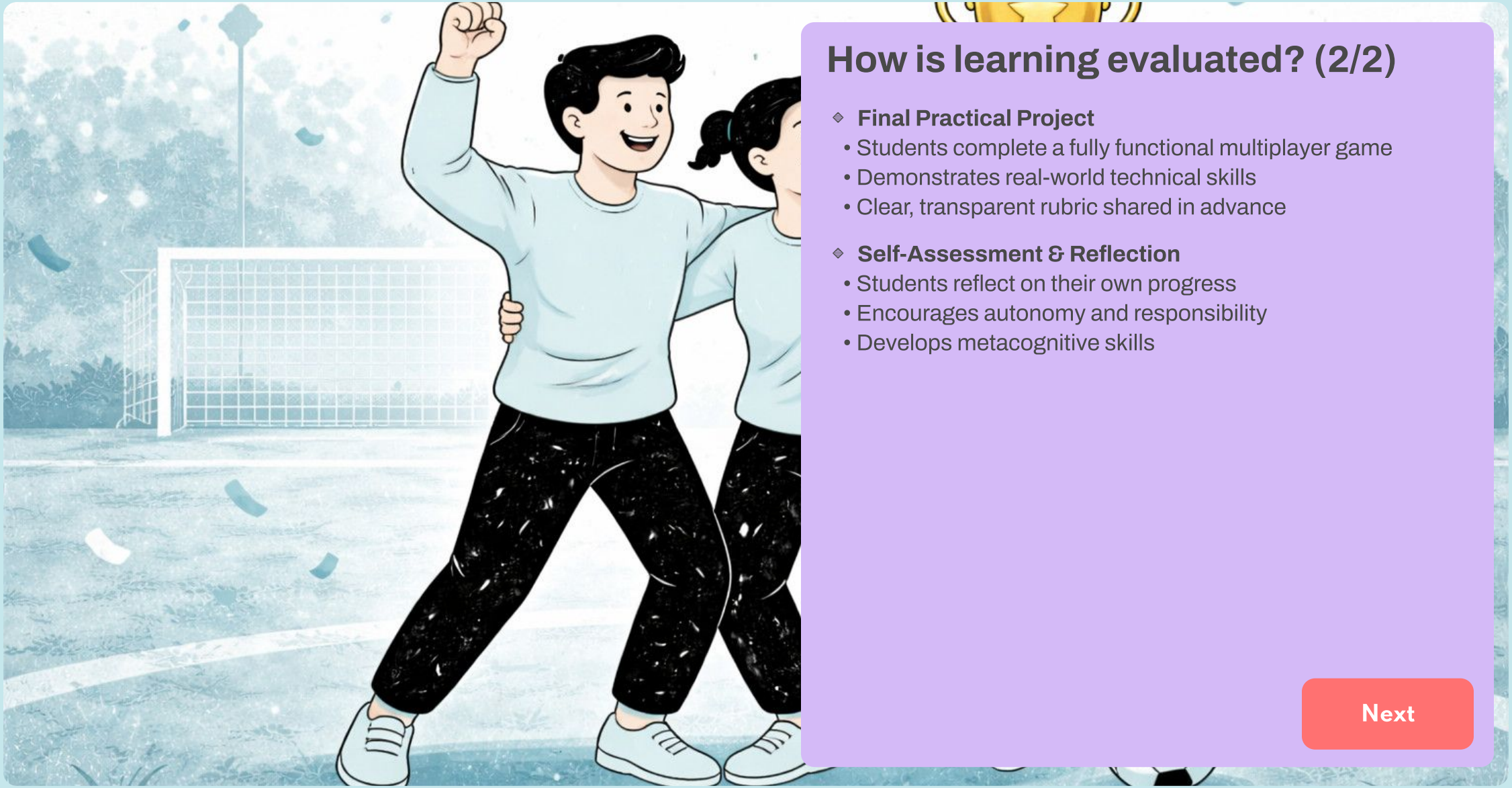
Next

# How is learning evaluated? (1/2)

## Evaluation Principles

- **Continuous Evaluation**
  - Progress is evaluated throughout the course
  - No single final exam determines success
  - Improvement over time matters
- **Competency-Based Assessment**
  - Evaluation focuses on:
    - Problem-solving ability
    - Technical understanding
    - System design
    - Collaboration
  - Inspired by engineering programs at leading technological institutions
- **Learning from Errors**
  - Errors are expected and analyzed
  - Students receive feedback at every step
  - Mistakes are part of the evaluation process

Next



# How is learning evaluated? (2/2)

- **Final Practical Project**
  - Students complete a fully functional multiplayer game
  - Demonstrates real-world technical skills
  - Clear, transparent rubric shared in advance
- **Self-Assessment & Reflection**
  - Students reflect on their own progress
  - Encourages autonomy and responsibility
  - Develops metacognitive skills

Next

# Your journey starts here

🎮 Don’t just play technology...

..Learn how to create it. 🔨