✅ 1. What the course looks like pedagogically

From the document :

Key characteristics of your course

- **Audience:** Teenagers (14–18) with *basic programming experience*.
- **Method:** Strongly *hands-on, learn-by-doing*. The “Pedagogical Notes” (p.18) explicitly state this.
- **Structure:** 15 sessions, each building a concrete Unity/VRChat feature.
- **Progression:**
    - From setup → basic world → HUD → gameplay logic → networking → multiplayer → game loop → restart system → handling late joiners.
- **Technological demands:** High—Unity, VRChat SDK3, UdonSharp, networking logic, VR interactions.
- **Continuous testing & iteration:** Every exercise requires testing in Unity Editor *and* multiple VRChat instances.
- **Final assessment:** Students create and present their own multiplayer VRChat game.

Together these indicate:
✔ Active learning
✔ Situated, contextualised problem-solving
✔ Technology-dependent instruction
✔ Clear step-by-step performance tasks
✔ Incremental competencies leading to a final project

✅ 2. Evaluating each instructional model against your course

🔷 ADDIE (Analysis – Design – Development – Implementation – Evaluation)

Strengths for your course

- Helps structure the *entire* course lifecycle.
- Good for complex technical content that must be planned in stages.
- Works well for project-based curricula.

Limitations for your use case

- ADDIE is *too high-level* for session-by-session classroom guidance.
- It does not focus specifically on *technology integration* nor *classroom execution*.
- The course is already created, so ADDIE's main value (big design planning) is partially done.

Fit: ★★★☆☆ (medium)
ADDIE is useful but not the most accurate match.

◈ ASSURE (Analyze Learners – State Objectives – Select Media & Materials – Utilize Media – Require Learner Participation – Evaluate & Revise)

Why it fits very well
ASSURE is **built specifically for technology-mediated, hands-on, classroom instruction**, which matches your course perfectly.

Your course explicitly follows ASSURE-like patterns:

A – Analyze learners

Your document defines:

- Age range
- Prior knowledge (basic programming)
- VR/non-VR usage issues
- Multiplayer coordination needs
  (See page 1 for target audience details.)

S – State objectives

Each session has:

- A clear *goal*
- A clear *deliverable*
  Examples:
  ✔ "Goal: In this exercise we will create the basic layout…" (p.2)
  ✔ "Deliverable: A simple enclosed arena…" (p.2)

S – Select methods, media, materials

Your course requires:

- Unity Hub
- VRChat SDK
- VCC

- VR Testing instances
  (See Step 3 of Session 1.)

This is pure ASSURE methodology.

U – Utilize media & technology

Students work *directly* with:

- Unity
- VR hardware
- VRChat client
- Networking
- Game testing tools

Technology is not a complement—**it *is* the medium of learning**.

R – Require learner participation

Every session includes:

- Construction tasks
- Debugging
- Multiplayer testing
- Rebuilding and retesting
  (Strong participatory learning.)

E – Evaluate & revise

Students evaluate functionality every step:

- “Test in Unity Editor”
- “Test in multiple instances”
- “Fix behaviour…”
  Also final assessment: functional multiplayer game (p.18).

Fit: ★★★★★ (excellent)
ASSURE matches your course *almost perfectly*.

◈ CD/CI (Content Development / Cognitive Instruction)

This model is useful for:

- Cognitive sequencing
- Scaffolding conceptual knowledge

Your course, however, is:

- Highly practical
- Less theory-driven
- Centered on *technical tasks*, not cognitive conceptualisation

Fit: ★★☆☆☆ (low)
Some elements apply (progressive scaffolding), but not enough.

◈ ARCS (Attention – Relevance – Confidence – Satisfaction)

ARCS is a **motivation model**, not a complete instructional design model.

Your course already implements motivational strategies:

- **Attention:** VR, hands-on, game building
- **Relevance:** Teenagers love VR/Unity/games
- **Confidence:** Clear step-by-step tasks with predictable results
- **Satisfaction:** Students build a full multiplayer game

But ARCS cannot structure the whole course by itself.

Fit: ★★★☆☆ (medium)
Great as a *supplement* but not as the primary model.

☑ 3. Final conclusion

☆ The best tecnopedagogical instructional model for your Unity & VRChat programming course is: *ASSURE*.

Why ASSURE is the strongest match:

- Your course is **fundamentally technology-based** → ASSURE specializes in tech-enhanced instruction.
- Each session already aligns with ASSURE steps (goals → media → participation → evaluation).
- The learning process is **active, iterative, and test-driven**, a core expectation of ASSURE.
- Learners must work with VR devices, Unity Editor, VRChat instances → ASSURE explicitly integrates media utilization.
- Ideal for **practical, procedural skills** with real-time testing and peer collaboration.

Secondary Support Models:

- **ADDIE** → Useful for your overall course planning (macro level).

- **ARCS** → Great to enhance motivation but not sufficient alone.
- **CD/CI** → Limited relevance for this practice-heavy course.