[ITADINFO]

CONVEGNO ITALIANO
SULLA DIDATTICA DELL'INFORMATICA

4ª edizione

Padova, 9-11 ottobre 2026



inTestAI

Valuta e migliora le competenze informatiche con il feedback assistito dall'AI

Gaetana de Gioia, Enrica Gentile,
Paola Plantamura, Francesca Pia Travisani
Dipartimento di Informatica
Università degli Studi di Bari Aldo Moro

AI nell'insegnamento dell'Informatica

Insegnamento dell'Informatica

Allenare le abilità tecniche e di problem solving degli studenti affrontando problematiche di varia natura

AI nella formazione

Migliorare le modalità di progettazione, erogazione e valutazione delle attività didattiche supportando il docente e lo studente

Valutazione Formativa con AI

L'AI permette la generazione automatica di problemi e domande di varia natura a partire dai materiali didattici messi a disposizione dal docente



Le tecnologie adottate: LLM e RAG

Large Language Model (LLM)

Modello linguistico addestrato su grandi quantità di input testuali, generando del testo a partire dalle istruzioni e dal contesto fornito

Retrieval Augmented Generation (RAG)

Tecnica che permette a un LLM di generare automaticamente del testo sulla base di informazioni pertinenti ottenute in input

LLM + RAG

Generazione automatica di problemi e domande a partire dai materiali didattici messi a disposizione dal docente nel corso nella piattaforma Moodle

mTestAI Generazione delle domande per l'autovalutazione

1

Materiali

Basati sui contenuti presenti all'interno del corso Moodle

2

Chunk

I contenuti sono divisi in frammenti brevi, significativi e coerenti

3

Embedding

Ogni frammento viene rappresentato attraverso un valore numerico

4

Database vettoriale

L'LLM recupera i frammenti per somiglianza dei contenuti

5

Prompting

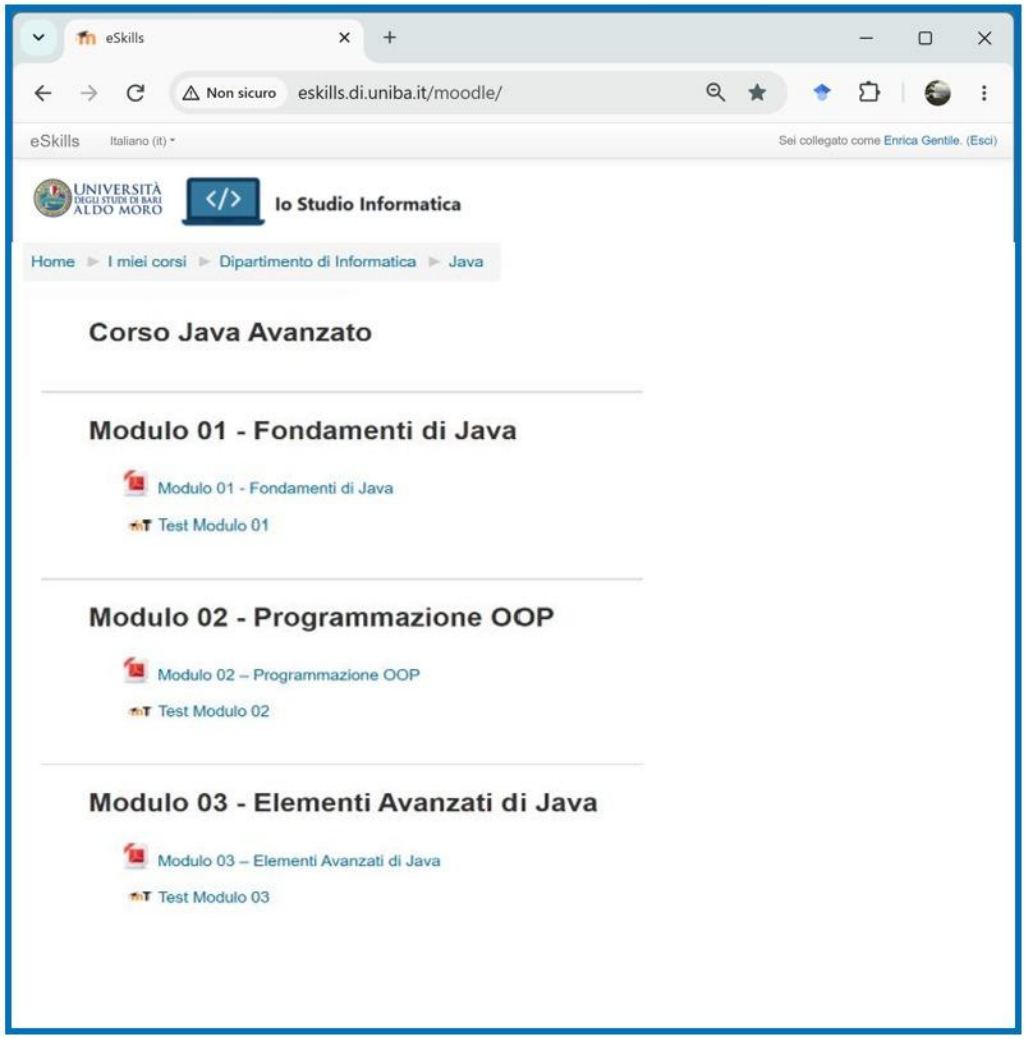
Le istruzioni per LLM sono arricchite dai frammenti pertinenti rispetto ai contenuti

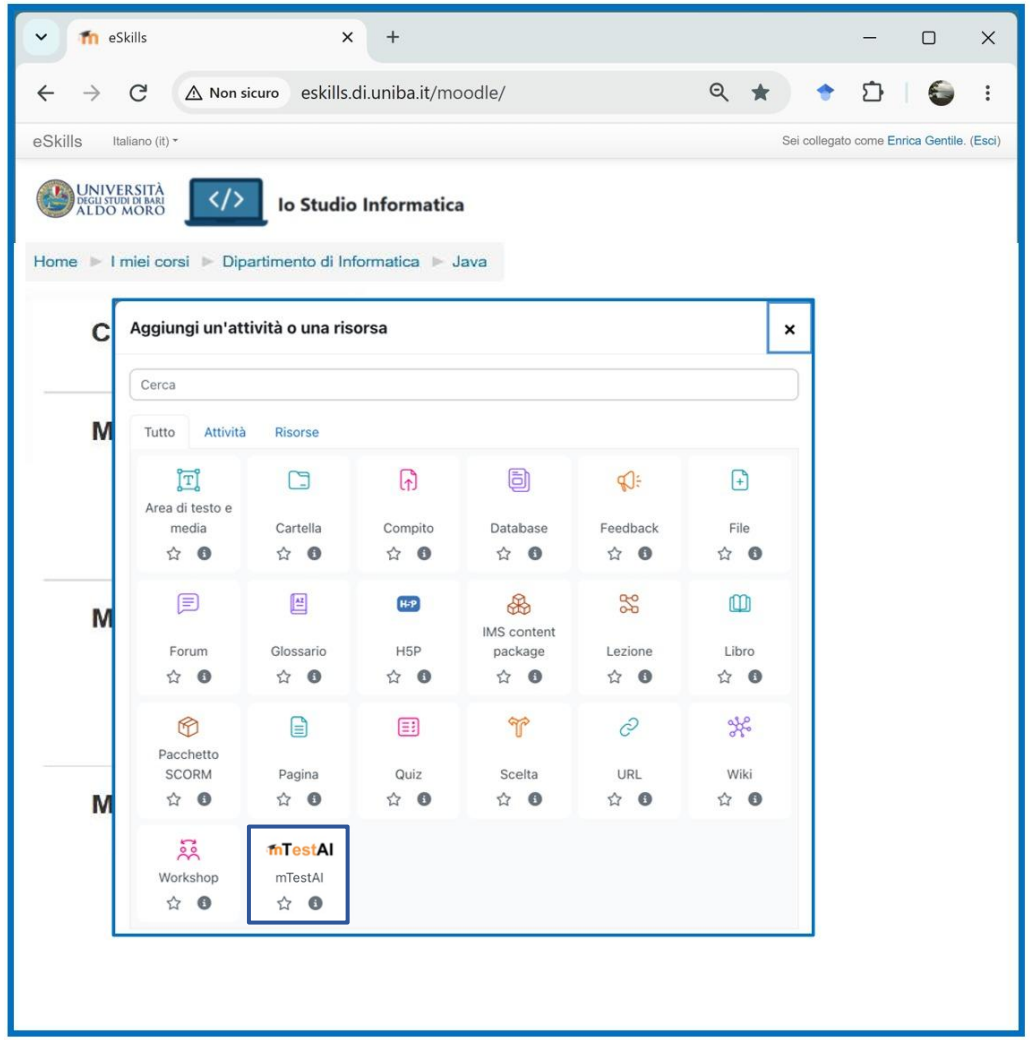
6

Test

Generazione delle domande aperte relative ai contenuti

mTestAI Il plug-in di Moodle





mTestAI Prompt di generazione delle domande

```
// Struttura Istruzioni di Generazione
Sei un docente universitario. Genera 5 domande aperte per:
Modulo 4 - Concorrenza.

Seed: 47291

CONCETTI GIÀ STUDIATI
• M01 - Tipi primitivi e struttura Java
• M02 - Classi ed Ereditarietà
• M03 - Collections Framework

REGOLA:
Non usare mai concetti di moduli non presenti nella lista.
```

Definizione del ruolo e contenuti ammessi

LLM è un docente e deve generare domande coerenti con il contenuto del modulo

Autovalutazioni diversificate

Un *Seed* numerico casuale rende ogni autovalutazione diversa dalle precedenti

Regola di creazione del test

LLM non utilizzerà argomenti non ancora seguiti

Tipologie di domanda

Analizzare

Applicare

Ricordare

Sintetizzare

Concettuale



Verifica la conoscenza teorica della terminologia e la comprensione dei costrutti

Confronto



Verifica la capacità di valutazione paragonando due o più approcci per risolvere lo stesso problema

Implementazione



Verifica l'applicazione pratica di regole sintattiche e logiche per scrivere o correggere il codice

Scenario



Verifica la capacità di analisi difronte ad una situazione problematica realistica

Livelli di difficoltà

Lunghezza delle domande	
Bassa	3–5 righe su un singolo concetto
Media	5–10 righe su un collegamento fra più concetti
Alta	10–15 righe su progettazione, implementazione o analisi critica

Punteggio Massimo	
Bassa	Tra 1 e 1.5 pt
Media	Tra 2 e 2.5 pt
Alta	Tra 3 e 3.5 pt



Invio per feedback

Prompt di generazione del feedback dell'AI

```
// Struttura Istruzioni di Generazione e Input modello
Sei un docente universitario. Genera le risposte alle domande aperte:
        Testo della domanda
        Risposta dello studente
        Frammento Modulo nro – Concorrenza pp. nro

VINCOLI DI VALUTAZIONE:
• criteri di giudizio
• punteggio massimo per livello di difficoltà
• gestione delle risposte lasciate in bianco
```

Definizione del ruolo

Docente esperto che corregge una prova scritta sullo specifico modulo di input

Input al modello AI

E' composto dal testo della domanda, dalla risposta dello studente e dal chunk del contenuto con le pagine del PDF

Vincoli di valutazione

LLM considererà il punteggio, i criteri di giudizio, il punteggio massimo per livello di difficoltà e gestirà le risposte lasciate in bianco

I criteri di valutazione



Completezza

Verifica della presenza di tutti gli elementi e i requisiti richiesti dalla traccia



Terminologia Tecnica

Controllo dell'uso appropriato e rigoroso dei termini tecnici specifici del dominio



Proprietà Linguistica

Analisi della chiarezza, della struttura logica e della comprensibilità del testo



Correttezza Concettuale

Validazione dell'accuratezza tecnica e scientifica delle spiegazioni fornite

Feedback formativo positivo



✅ Domanda 3 — m01_c03 100%

✓ Terminologia ✓ Linguaggio

Scrivi un programma Java che dichiari tre variabili usando rispettivamente un tipo primitivo, la parola chiave var e una costante final. Per ciascuna variabile, spiega dove viene allocata in memoria (stack o heap) e descrivi il comportamento dell'autoboxing se converti uno dei valori primitivi nel corrispondente wrapper.

La tua risposta:

```
public class Esempio {
public static void main(String[] args) {
int numero = 42; // tipo primitivo
var testo = "ciao"; // var → inferito come String
final double PIGRECO = 3.14; // costante final (primitivo)

Integer boxed = numero; // autoboxing: int → Integer
System.out.println(numero + " " + testo + " " + PIGRECO + " " + boxed);
}
}
```

*`numero` e `PIGRECO` sono primitivi, quindi i loro valori stanno sullo **stack** (il `final` impedisce solo la riassegnazione, non cambia l'allocazione). `testo` è inferito come `String`: la referenza sta sullo stack, ma l'oggetto vive nell'**heap** (nel String pool). Con `Integer boxed = numero` scatta l'**autoboxing**: il compilatore inserisce `Integer.valueOf(numero)`, creando un wrapper `Integer` nell'heap con la referenza sullo stack — tieni presente che `valueOf` ha una cache per i valori da −128 a 127, quindi per `42` riusa un oggetto esistente.*

Valutazione:

Risposta completa ed eccellente. Il codice presenta correttamente tutte e tre le dichiarazioni richieste: primitivo (int), var (String) e final (double). La spiegazione dell'allocazione in memoria è precisa: primitivi sullo stack, oggetti nell'heap con reference sullo stack, String pool correttamente menzionato. L'autoboxing è descritto in modo accurato, specificando Integer.valueOf() e la cache [-128,127]. Terminologia tecnica impeccabile (autoboxing, wrapper, inferenza di tipo, String pool). Linguaggio chiaro e ben strutturato.

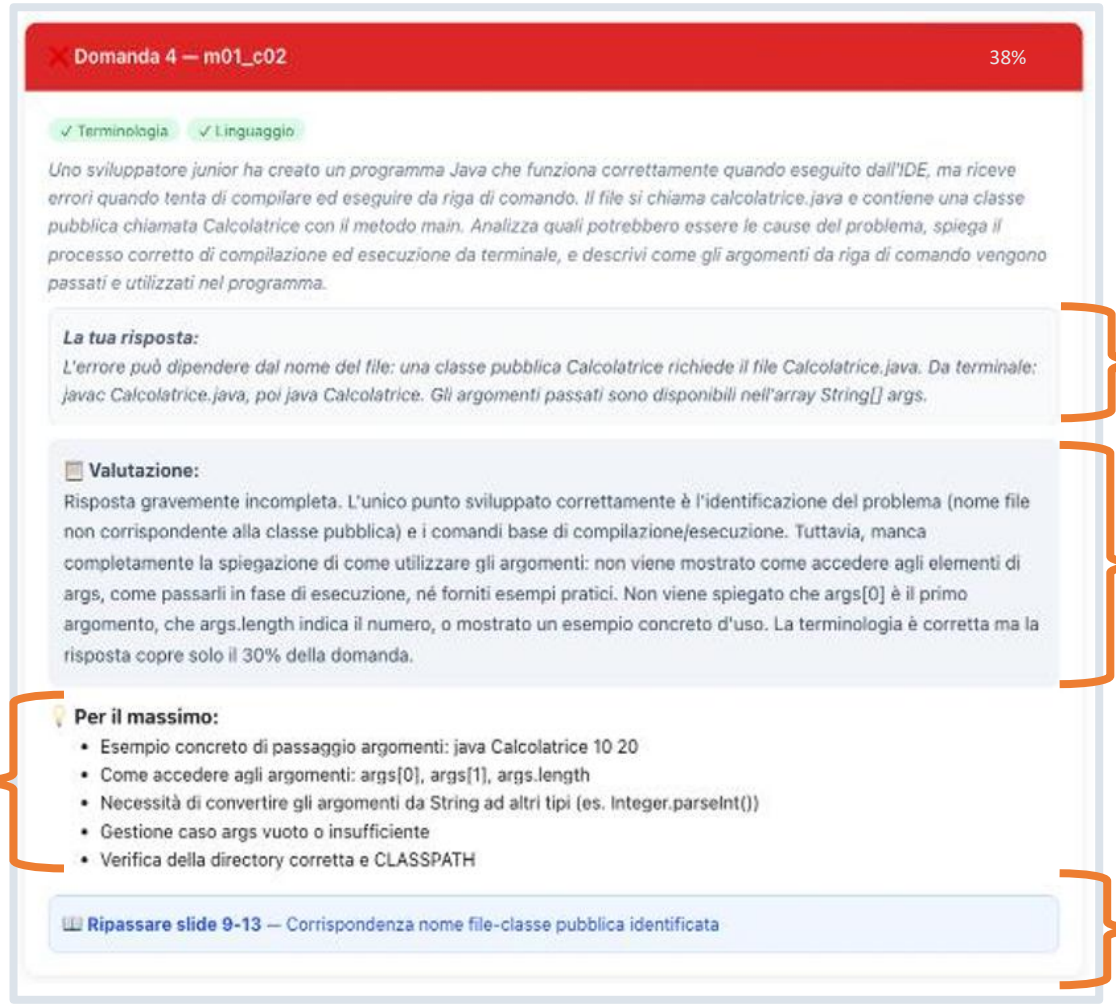
Il punteggio si riferisce alla singola domanda

Il parere qualitativo si basa sui criteri di valutazione

Feedback formativo con suggerimenti



Se la lascio in bianco?
L'AI indica le pagine del PDF da studiare e propone spunti di riflessione sull'argomento da trattare



Risposta dello studente

Valutazione dell'AI

L'AI elenca i concetti da trattare perché lo studente dia una risposta completa

L'AI riporta le pagine del PDF del corso da ripassare

Conclusioni





Generazione automatica delle domande aperte diversificate

Autovalutazione formativa immediata

Valutazione fornita dall'AI sulle risposte

AI-tutor per la regolazione consapevole dello studio

TEST DI FINE CORSO

L'AI genera il test finale sulla base delle scelte del docente rispetto a: difficoltà, numero e argomento delle domande

[ITADINFO]

CONVEGNO ITALIANO
SULLA DIDATTICA DELL'INFORMATICA

4ª edizione

Padova, 9-11 ottobre 2026



InTestAI

Valuta e migliora le competenze informatiche con il feedback assistito dall'AI

Grazie per l'attenzione

Gaetana de Gioia, Enrica Gentile,
Paola Plantamura, Francesca Pia Travisani
Dipartimento di Informatica
Università degli Studi di Bari Aldo Moro

paola.plantamura@uniba.it