

# Pädagogische Steuerung des Nutzerverhaltens in der Testumgebung

---

Ein zentrales Designziel der Anwendung ist die Balance zwischen summativer Bewertung (Testen) und formativem Lernen. Um oberflächliches Bearbeitungsverhalten (*Rapid Guessing*) zu minimieren und metakognitive Kompetenzen im Zeitmanagement zu fördern, implementiert das System eine Reihe von verhaltenssteuernden Mechanismen (*Nudges* und *Constraints*).

## 1. Differenzierung durch Tempo-Modi

Um unterschiedlichen Lernständen und Zielen gerecht zu werden, bietet das System drei Tempo-Modi (*Normal*, *Speed*, *Power*). Dies ermöglicht ein **Scaffolding** der Anforderung: Während der *Normal*-Modus (Basiszeit) die kognitive Durchdringung und das tiefe Verständnis in den Vordergrund stellt, fokussieren die Modi *Speed* (50 % Zeit) und *Power* (25 % Zeit) auf die Automatisierung und Abrufschwindigkeit (*Fluency*) von Wissen. Die Wahl des Modus überträgt die Verantwortung für das Anforderungsniveau an den Lernenden und stärkt das Autonomieerleben.

## 2. Metakognitives Feedback durch Pacing-Indikatoren

Statt den Nutzer lediglich mit einem ablaufenden Countdown zu konfrontieren, bietet das System eine kontinuierliche, qualitative Rückmeldung zum Bearbeitungsfortschritt. Ein dynamischer Pacing-Algorithmus vergleicht die verstrichene Zeit mit der idealen Bearbeitungszeit für die aktuelle Frage (basierend auf deren Komplexitätsgewichtung). Visuelle Indikatoren (z. B. "*Im Zeitplan*", "*Leichter Rückstand*") dienen als externer Taktgeber, der die Selbstregulation unterstützt, ohne kognitive Ressourcen durch eigene Zeitkalkulationen zu binden.

## 3. Förderung tiefer Verarbeitung durch Cooldown-Mechanismen

Um impulsives Klickverhalten zu unterbinden und eine minimale kognitive Auseinandersetzung mit dem Lernmaterial zu erzwingen, setzt die App dynamische Sperrzeiten (Cooldowns) ein:

- **Lese-Cooldown (Pre-Answer):** Der Antwort-Button bleibt für eine dynamisch berechnete Zeitspanne deaktiviert. Diese Zeit skaliert mit der Komplexität der Frage (Gewichtung) und der Anzahl der Antwortoptionen. Dies erzwingt eine minimale Rezeptionsphase und verhindert, dass Fragen beantwortet werden, bevor der Text vollständig erfasst werden konnte.
- **Reflexions-Cooldown (Post-Answer):** Nach der Antwortabgabe wird die Navigation zur nächsten Frage kurzzeitig verzögert. Dies schafft ein zeitliches Fenster für die Rezeption des formativen Feedbacks (Erklärungstexte), das in stressigen Testsituationen sonst oft übersprungen wird.

## 4. Adaptive Fairness ("Panic Mode")

Um Frustration in kritischen Zeitphasen zu vermeiden, überwacht das System die verbleibende Gesamttestzeit relativ zur Anzahl der offenen Fragen. Fällt das Zeitbudget unter einen kritischen Schwellenwert (z. B. < 15 Sekunden pro verbleibender Frage), werden alle Cooldown-Mechanismen automatisch deaktiviert (*Panic Mode*). Dies priorisiert in der Endphase die Fertigstellung des Tests

gegenüber der didaktischen Steuerung und verhindert, dass die pädagogischen Bremsen zum Nachteil des Nutzers werden.

---

### Zusammenfassung der Implementierung:

Maßnahme	Didaktisches Ziel	Technische Umsetzung
<b>Tempo-Modi</b>	Binnendifferenzierung, Fluency-Training	Globale Zeitfaktoren (1.0, 0.5, 0.25) auf Testdauer und Cooldowns.
<b>Pacing-UI</b>	Metakognition, Zeitmanagement	Echtzeit-Vergleich von <code>elapsed_time</code> vs. <code>ideal_time</code> (gewichtet).
<b>Cooldowns</b>	Deep Processing, Vermeidung von Raten	Button-Sperre ( <code>disabled=True</code> ) basierend auf <code>time.monotonic()</code> .
<b>Panic Mode</b>	Fairness, Frustrationsvermeidung	Override aller Cooldowns bei <code>remaining_time &lt; threshold</code> .