

# heyprof – An AI-Powered Platform for Teaching and Learning

Architecture, Capabilities, and Pedagogical Design

Volker Reichenberger

Dirk Schieborn

23 September 2026, Version 2.0

heyprof is a web-based platform for teaching and learning built around an AI tutor that does not answer from the world knowledge of a language model but from the course: from the lecturer’s slides, script and recorded lecture, from the current exercise with its model solution and private tutor notes, and from the student’s own work so far. It guides Socratically instead of handing out solutions and stays at the level of the course. The second building block is a library of more than 590 interactive visualisations (VisuApps), which the tutor can demonstrate itself. Around this sits the full life cycle of a course: materials, exercises with automatic feedback, tests, mock exams with handwritten submission, recordings, live quizzes and a community. heyprof was developed by the authors and was used as an experimental prototype in teaching during the summer term of 2026. This paper describes the architecture, the features for lecturers and students, the VisuApps, the AI integration, the pedagogical principles, data protection, and the state of deployment.

## 1 Introduction

Large language models have arrived in higher education, mostly in the shape of general-purpose assistants that do not know the course. Two problems follow. The first is the *cognitive bypass*:<sup>1</sup> students have the task solved for them instead of understanding it. The second is the lack of fit: the answer does not know where the course stands, is too complex, or uses concepts not yet introduced. A typical example is a solution to a programming exercise that uses constructs the course has not covered, such as list comprehensions or type hints.

heyprof starts exactly here. For every question the AI tutor receives the context of the course: the relevant passages from script and slides, the current exercise with its model solution and the lecturer’s private tutor notes, the student’s code so far and, if desired, the transcript of the recorded lecture. It gives *no* complete solutions at first but guides with questions and hints matched to the level of the course, and it explains with the examples and metaphors used in the lecture hall.

---

<sup>1</sup>A technical aid takes over the cognitive work completely; the learning effect vanishes.

Since the first edition of this paper (March 2026) the platform has grown in three directions. First, a library of interactive visualisations, the *VisuApps*, with more than 590 apps in 66 subject collections that the tutor can remote-control and demonstrate. Second, deployment in primary schools and corporate training, each with its own set of rules. Third, operations: multi-tenancy, a data protection concept, Moodle import and a test project alongside production. This text describes the state of September 2026.

## 2 System Architecture

heyprof is built on a web stack chosen for rapid iteration and minimal operational overhead.

### Frontend

Next.js 16 with React 19 and TypeScript (App Router), Tailwind CSS and shadcn/ui. Mathematics is typeset client-side with KaTeX, PDFs are rendered with pdf.js, code is edited in CodeMirror (Python, SQL).

### Backend

Serverless API routes in Next.js for all AI and business logic. Supabase (PostgreSQL with row-level security) for data, authentication and file storage; vector search runs in the pgvector extension.

### AI services

OpenAI models of the GPT-4.1 and GPT-5 families for tutoring, grading, translation and summarisation, text-embedding-3-small for semantic search, Whisper for transcription, a speech model for text to speech and a vision model for handwriting and photos. Selected AI workshops (CogniApps) use Anthropic's Claude models. All calls go through the platform, never directly from the browser to the provider.

### VisuApps

Static, self-contained HTML applications built on an in-house class library (VisuKit, 2D and 3D on Three.js), embedded via iframe in courses, demos and the website, and receiving theme, colour palette, language and tutor commands through a message protocol.

The application is deployed on Vercel and needs no dedicated server infrastructure. Roles (lecturer, student, administration) are enforced at the database level through row-level security and server-side checks. Since August 2026 the platform is multi-tenant: organisations receive separate data spaces, and a dedicated test project separates development from production. A CI pipeline checks types and tenant isolation on every change. Schema changes are made exclusively as versioned migrations.

The **data model** consists of the core entities *Course*, *Session*, *Material* (with semantically indexed *Chunks*), *Exercise* (open-ended, multiple-choice, code) with *Collections* and *Labels*, *Exam*, *Attempt* or *Submission*, *Recording* with transcript, *LiveQuiz*, *CommunityPost* and the learning signals of the *learning map*.

On top of this core sits a **kit of 42 course modules**. The lecturer enables per course whatever is needed: exercises, tests and mock exams, VisuApps and CogniApps, community and anonymous

questions, live quizzes, flash rounds, action weeks and a score board, a vocabulary trainer, a data lab, micro-lessons, the learning map, knowledge rooms, heyCanvas, class administration and subject-specific workshops. A course shows only the modules that are switched on, which keeps the interface narrow although the kit is broad.

## 3 Features for Lecturers

### 3.1 Creating a course

A course comes into being in one of three ways. Classically in the dashboard, with an access code students use to enrol. From existing documents: from scripts, slide decks or exercise sheets the platform derives sessions, materials and exercise proposals, which the lecturer reviews and adopts. Or by importing an existing Moodle course in .mbz format, taking over sections, files and activities. Every course is divided into *sessions* corresponding to lecture dates; the order of all elements can be changed by drag and drop.

### 3.2 Materials and annotation

Materials are PDF slides, scripts, videos or HTML slide decks, which the platform renders as self-contained decks with formulas and embedded VisuApps. PDF pages can be annotated with a freehand tool; students see the annotations as an overlay. After upload, materials are *ingested*: the text is extracted page by page, split into chunks of at most 1500 characters, embedded and stored in the vector store, so that the tutor can quote precisely from the course material. The lecturer's handwritten annotations are converted to text and  $\text{\LaTeX}$  by handwriting recognition and given to the tutor as context.

### 3.3 Slide workshop

If no finished slide deck exists, the platform builds one. In the *slide workshop* (SlideLab) the lecturer uploads sources – a script, notes, an old deck – which the AI analyses before proposing an outline. From that an HTML deck is generated that can be revised slide by slide; matching VisuApps are offered as short profiles and embedded directly. The result is course material like any other: searchable, annotatable and indexed for the tutor.

### 3.4 Presenting in the lecture hall

The lecturer presents straight from the platform, without detouring through another program. PDF and HTML slides carry a handwriting overlay, and a laser pointer works across embedded VisuApps. A cover reveals the slide step by step. Next to it sits a digital blackboard with an adjustable board colour and a second surface for side calculations. A device switch decides whether the finger writes or turns the page – on a lecture hall touch panel a more important setting than it sounds. Whatever is written on a slide is kept and is afterwards available to students as an overlay.

### 3.5 Classes and accounts without e-mail

Not every group of learners has e-mail addresses. For school classes the teacher therefore creates the accounts, from a list of names, and prints access cards. Learners sign in through a class link with a user name and an initial password. In primary school courses a *picture password* takes its place: tap your name, tap three pictures in the right order, done – no reading required. A monster serves as the avatar. Resetting passwords, deactivating and deleting accounts rest with the teacher; there is deliberately no self-service reset. This makes it possible to run groups for whom ordinary registration is legally or practically out of the question.

### 3.6 Exercises

The exercise editor knows three types: *open-ended*, *multiple-choice* and *code* (Python, SQL). Each exercise carries a model solution (visible to students if desired), private tutor notes seen only by the AI tutor, and labels and collections for organisation, for instance by topic and difficulty. Collections can be imported from Markdown files, generated by the AI from a topic description, or translated between German and English at the press of a button.

### 3.7 Tests

A *test* is a gradable sequence of questions that runs entirely inside the platform. Question types are free text, fill-in-the-blank, multiple choice and numerical answer. Every question can carry *variants* from which the platform picks one at random per attempt; questions can also be shuffled. The lecturer sets the time limit, the number of attempts, the availability window, a restriction to particular networks, a penalty for repeat attempts, and how much feedback is shown – none, points only, or full feedback. The tutor can be switched off for a test or limited to a fixed number of hints. On request the AI generates the test from the course materials. Model solutions and attempts sit behind server-side routes so that the browser cannot read them.

### 3.8 Exams

Exams consist of numbered questions with points and a time limit. Questions are imported from scanned exam sheets; an AI pipeline extracts text, numbering and points. Students submit text, drawings or photos of handwritten solutions, which the vision model reads and grades. After submission all papers are available with an AI pre-assessment and feedback; the lecturer keeps the final say.

Exams currently serve *preparation* only; the module is accordingly called *mock exam* inside the platform. Students practise under exam conditions and then see where they stand. No formal assessment is attached to them, and none is intended at present.

### 3.9 Recordings, transcription and live quiz

A browser-based recorder captures the lecture; alternatively audio files are uploaded. Whisper transcribes in the background, a language model produces a structured summary in several steps. Transcript and summary are available to students, and the tutor can answer from the

spoken word of the lecture, with exactly the examples and metaphors used in the lecture hall. For classroom teaching there is a *live quiz*: the lecturer launches a question, students answer on their phones, the result appears immediately. Alongside it are *flash rounds* with a QR code for short polls in the room, a shared *score board* for group contests that computes round totals and ranking by itself, and *action weeks* for time-limited course activities with evaluation and CSV export.

### 3.10 Learning analytics

A course-wide dashboard shows the lecturer how the cohort works: which materials have reach, which VisuApps are used, how the vocabulary learning curve develops, how much time is invested, and which topics keep the tutor busy. A traffic light marks anomalies, a cohort comparison puts them in context, and a term report and CSV export leave the building. The evaluation is tightly bounded in data protection terms: events are condensed by day, deleted after 60 days, and whether real names are visible is decided by a separate switch. The underlying tables are closed to clients and reachable only through audited server routes.

### 3.11 Tutor configuration

At course level the lecturer stores tutor notes for material and exercise questions or disables the tutor. Its *personality* (a *soul* in the system) is chosen from profiles describing tone, style and emphasis. Besides factual and encouraging variants there are colloquial ones. A course-level tutor view shows which topics currently occupy the cohort.

## 4 Features for Students

### 4.1 Learning studio

The dashboard shows enrolled courses with progress and notes on new materials. In the *learning studio* each session presents slides with annotations, the recording with transcript and summary, and the exercises of the day on one page, with the AI tutor alongside. In the PDF viewer students mark a region of a slide or a board image; the tutor explains exactly that excerpt. A course-specific welcome tour guides first-time visitors through the interface; afterwards the tutor serves as a guide. The interface is available in German and English, for desktop and phone, in light and dark mode.

### 4.2 Exercises

Several paths are open for every exercise:

1. **Text or code** with immediate AI feedback.
2. **Drawing canvas** for handwritten solutions, also with a pen on a tablet.
3. **QR photo**: the desktop shows a QR code, the phone captures the handwritten solution, the photo appears on the desktop and is read and graded.
4. **AI tutor** in the integrated dialogue, which knows the task, the model solution, the tutor notes and the current state of the solution.

5. **Practise a variant:** the AI generates a structurally identical exercise with different numbers or names.
6. **Show solution**, after a genuine attempt.

Python runs entirely in the browser (CPython as WebAssembly, Pyodide) with syntax highlighting, a run button and an output pane; a time limit stops endless loops and dangerous imports are blocked. For SQL there is a playground with an in-browser database. Exercises follow the student's state and motivation: graded attempts, labels and the learning signals from VisuApps feed a *learning map* per competency, from which suggestions for the next exercises are derived.

### 4.3 Community and voice

Every course has a community with anonymous questions, answers and votes; lecturers see what moves the cohort without anyone having to expose themselves. The tutor speaks on request: answers and explanations in VisuApps are read aloud by a server voice, which makes the difference for primary school pupils and language learners.

### 4.4 Further learning spaces

Beside material and exercise sit course modules that follow other approaches.

#### Knowledge rooms

a walkable 3D building per person: rooms, floors and staircases are built by the learner, and VisuApps, materials, videos and notes go on the inner walls. The apps run there *live* and keep their state. The lecturer can publish their building as a course exhibition.

#### heyCanvas

an infinite, zoomable surface for lecturing and video production, holding handwriting, text and formula islands and operable VisuApps side by side; an overview shows the surface as a tree and the view flies to a chosen island.

#### Micro-lessons

short AI-guided dialogues with intermediate questions and adaptive help, which can be built around selected VisuApps of the course – the AI drafts them from the material.

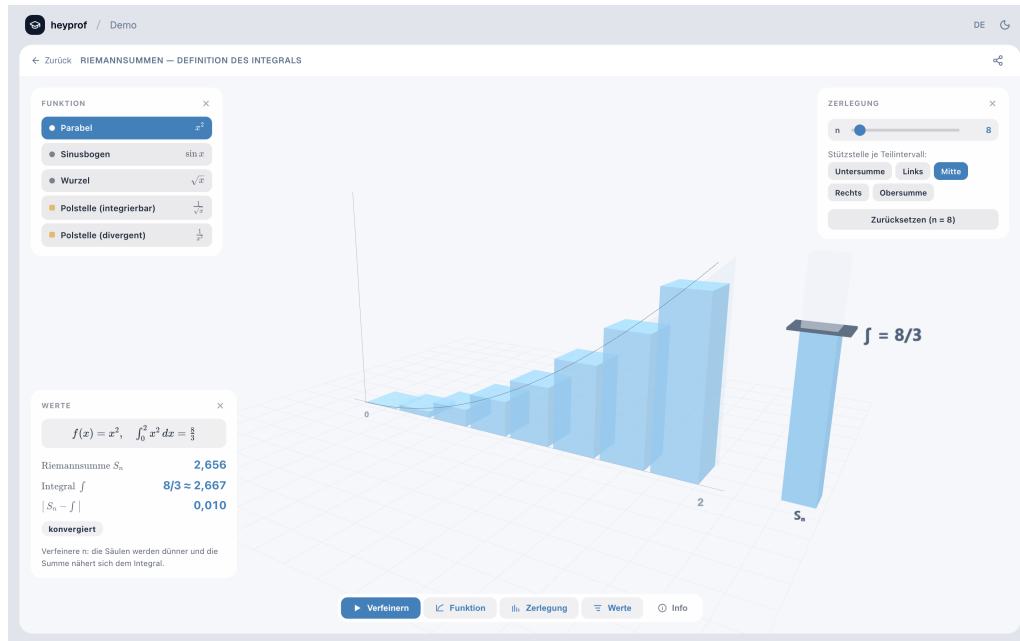
#### Data lab

visual data workflows in the manner of KNIME: read in, prepare, build models, dashboard. The nodes really compute, and the tutor helps with debugging.

#### Vocabulary trainer

sentences and vocabulary taken from a template by text recognition and practised in several modes.

For primary schools there is *Kompass 4*, a trainer for the year-four comparative assessments: AI-generated practice tests with a proficiency profile, a companion through the practice run, an adaptive difficulty balance, tasks in nine family languages, a paper mode (print, scan, grade, review) and a report for parents.



**Figure 1:** The VisuApp *Riemann sums in 3D*: partition, sample point and function are adjustable, the values card shows sum, integral and their distance. Through *Tutor Show* the tutor can operate the app itself while explaining.

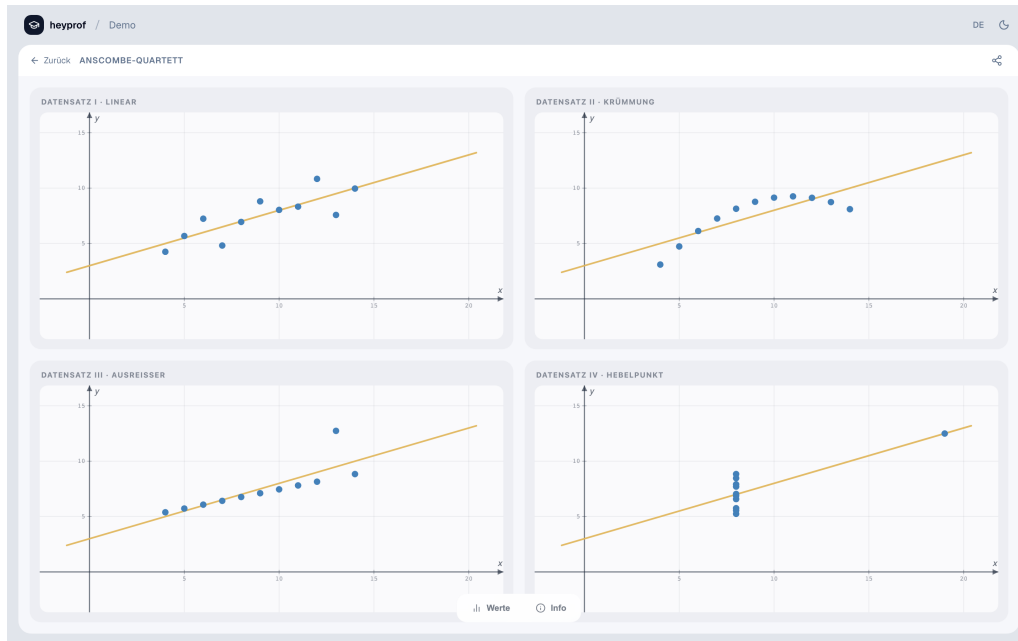
## 5 VisuApps: Knowledge You Can Touch

Many relationships are not understood from a formula but by turning their knobs. VisuApps are interactive visualisations that allow exactly that: refining Riemann sums until the columns become the integral; Anscombe's quartet with identical statistics and entirely different shapes; a neuron, a pendulum, a credit loss distribution, a high-speed rail network. In September 2026, 590 apps in 66 subject collections are registered, from mathematics, physics, chemistry, biology and computer science through economics, statistics and financial derivatives to linguistics, music, geography and academic writing.

**Common foundation.** All apps build on *VisuKit*, an in-house class library for 2D and 3D. It provides the runtime (theme, colour palette, language, throttling during a tutor stream), scenes, nodes, interchangeable visual languages and the uniform control scheme: panels in the corners, a pill bar at the bottom, an info panel with the didactic summary, bottom sheets on the phone. Colours derive from the course colour, so an app fits into any course, light or dark.

**Didactics first.** Every app starts with a didactic brief: What is the smallest correct unit of knowledge? Which misconception should the scene visibly refute? Which objects must be distinguishable, which states exist, which aha moment should arise? Whatever carries no concept stays out of the scene; no animation without a learning purpose.

**Tutor Show.** Through a machine-readable manifest an app declares which anchors, verbs and controls it offers the tutor. The tutor then does not answer with chat text but demonstrates: it places speech bubbles at the spot in question, points with a pointer, sets sliders, moves the camera



**Figure 2:** Anscombe's quartet: four data sets, the same regression line, the same statistics, four shapes. An argument for plotting that one does not forget.

and highlights panels; learners can intervene at any time. 568 of the apps can be controlled this way. A guided explanation thus always comes from the tutor and follows the question, not a script.

**CogniApps and VisuLab.** 23 *CogniApps* let the AI generate exercise material itself: unlimited citation drills, a sparring partner for the research question, a critical reader for outlines, a clinic for questionnaire items. In the *VisuLab* lecturers and students build their own visualisations with AI support; 414 such apps have been created. Public demo apps are reachable on the website without login and are embedded in course decks.

**Embedding on third-party sites.** A VisuApp can be placed as an iframe into any website, for instance that of a university or a publisher. Access runs through signed keys that are created in the administration area and come with the matching embed snippet; a parameter sets the accent colour so that the app fits the host design. Usage is counted per key, and delivery is restricted to the approved sites.

**Primary school.** For years 1 to 4 a separate set of rules applies: a speaking companion instead of text, operation without having to read, with large tap targets, family language, lessons with subject vocabulary, calm feedback without confetti. 23 apps of a 100-app programme are complete, from number decomposition to multiplication tables.

## 6 AI Features and Pedagogical Integration

### 6.1 Context instead of world knowledge

The central difference from general-purpose assistants is context. Before the tutor answers, its system prompt is enriched with

- the full text of the current exercise and the model solution,
- the lecturer's private tutor notes for this exercise and the course,
- the student's current code or solution state,
- semantically retrieved passages from the course materials and the lecture transcript,
- the recognised handwritten annotations on the current slide and the marked excerpt,
- for a VisuApp, the app's manifest and its current state,
- the chosen tutor personality.

This lets the tutor help precisely and at the right level *without* giving the answer away. A general AI knows neither the textbook nor the constructs introduced in the course. heyprof does.

### 6.2 Retrieval-Augmented Generation

Retrieval-augmented generation supplies a language model with relevant context from an external knowledge base before it answers [1]. The model itself is unchanged; the prompt is enriched with matching passages. In the *ingestion phase* every document is extracted page by page, split into chunks of at most 1500 characters and turned into vectors with `text-embedding-3-small`, stored together with the text in `pgvector`. In the *query phase* the question is embedded the same way, the *k* most similar chunks (cosine similarity) are found and inserted into the prompt. If someone asks “What is a linked list?”, the tutor answers from the course script, in the notation of the lecture, not from the model's world knowledge.

### 6.3 Socratic mode and explain mode

In **Socratic mode** the tutor asks counter-questions that lead one step further without revealing the solution:

*Student:* I don't know how to start with the recursion.

*Tutor:* Think about it: what is the simplest input for which you know the answer immediately, without computing anything?

In **explain mode** the tutor explains directly and vividly, for instance when someone does not understand a slide, with concrete examples, everyday metaphors and explicit reference to the slide content. In a VisuApp the explanation becomes a demonstration.

### 6.4 Automated grading

Text answers are graded by a structured model call returning a verdict and short feedback. Handwritten or photographed solutions are read by the vision model, checked for relevance and converted into  $\text{\LaTeX}$  notation. Grading updates progress immediately; the lecturer can inspect and override every grade.

## 6.5 Further AI services

### Exercise variants

structurally identical exercises with different numbers, names or functions for repetition.

### Exercise generation

whole collections from a topic description, released by the lecturer.

### Translation

of exercises between German and English; formulas and code are preserved.

### Exam hints

a limited number of Socratic hints per question, even during the exam.

### Handwriting recognition

for annotations, drawing canvases and photos.

### Text to speech

through a server voice, so that numbers and formulas sound the same on every device.

## 7 Pedagogical Principles

**Constructive alignment** [2]. Every exercise, every tutor note and every collection is linked to a learning objective. The AI uses the lecturer's private notes to guide towards the expected solution without giving it away.

**Zone of proximal development** [3]. The Socratic tutor looks for the boundary of current understanding and asks a question that goes exactly one step beyond it.

**Deliberate practice** [4]. The variant feature allows immediate practice after understanding: same concept, new numbers, instant feedback.

**Multimodal input.** Typing, drawing, photographing and speaking are equal; the gap between thinking and submission format becomes small.

**Visualisation with intent.** VisuApps show only what carries a concept. Every motion serves a state change or the aha moment; scientific soundness comes before photorealism. The tutor demonstrates instead of decorating.

**Learners as creators.** In the VisuLab students build their own visualisations, in the community they answer each other's questions, in peer feedback they assess against shared checklists.

## 8 Data Protection and Operations

Learning platforms process data of people who cannot choose where they learn. heyprof treats this as a design task, not a footnote. Data processing agreements exist with all service providers; the list of providers with purpose, location and legal basis is published on the platform. AI requests go exclusively through the platform, with the necessary context and without names; VisuApps load nothing from third-party servers, so no IP address reaches a third party. Users obtain their Art. 15 GDPR disclosure directly in their profile; deleting an account removes all associated data, verified by an inventory across all tables. A record of processing activities and a

road map document the state. Development and production are separate projects; organisations are isolated from one another as tenants.

Multi-tenancy also has an interface: an organisation administers its members, handles join requests and issues invitations; managed class accounts join the organisation of their course automatically. The community board is separated per organisation, and how far a post reaches is decided by the course.

Operation is funded through tiers. An account sits on a free or a paid tier to which a binding feature matrix is attached; payment runs through an external provider and activation through its callback to the platform. Payment data are not stored in heyprof.

## 9 Deployment and Figures

During the summer term of 2026 we used heyprof as an experimental prototype in our own teaching. The figures in Table 1 come from that deployment and should be read accordingly: they describe the use of a prototype over one term, not the routine operation of an established platform.

**Table 1:** Usage of the production instance as of 21 September 2026.

|                                         |                  |
|-----------------------------------------|------------------|
| Accounts (lecturers / students)         | 312 (19 / 293)   |
| Courses (published / with enrolments)   | 78 (44 / 31)     |
| Enrolments                              | 529              |
| Students active in the last 90 days     | 224              |
| Materials / indexed chunks              | 413 / 11 989     |
| Exercises                               | 2 854            |
| Exercise attempts (students)            | 11 672 (181)     |
| Lecture recordings / live quizzes       | 125 / 39         |
| Community posts                         | 191              |
| Visualisations built in the VisuLab     | 414              |
| AI calls since 31 March 2026 (users)    | 40 416 (215)     |
| of which tutor dialogues (students)     | 17 635 (167)     |
| of which text gradings / photo gradings | 8 237 / 3 938    |
| of which handwriting recognitions       | 7 454            |
| Language model tokens processed         | about 83 million |

The platform is developed and operated by Studio Quant GmbH.

## 10 Future Development

heyprof is developed continuously; an incomplete selection.

**Adaptive learning paths.** The learning map already collects signals from exercises and VisuApps. The next step is a recommender that proposes the appropriately challenging exercise from the individual history, with prerequisite relations in a competency graph.

**Native iPad app.** A standalone iPad application is under development. It carries the web interface in a native shell and adds what the browser cannot offer: handwriting through PencilKit with document-wide undo, annotation of slides directly on the device, and a native presentation mode for the lecture hall.

**Integration with institutional systems.** Single sign-on and automatic enrolment through the examination office's systems.

**Further worlds.** The primary school programme grows to 100 apps; the collections for schools, continuing education and lifelong learning are expanded.

## 11 Conclusion

heyprof shows that AI in teaching need not mean that learners bypass the learning process. A tutor deeply embedded in material, exercise and visualisation, supplied with pedagogical context and bound to Socratic guidance, turns a potentially harmful tool into an effective learning aid. Learners receive immediate, level-appropriate support around the clock without skipping the cognitive effort that genuine learning requires. Lecturers gain an authoring environment, automatic feedback, a library of vivid visualisations and a view of where their cohort stands.

Not least, heyprof is itself a product of AI-assisted development: a large part of the code and documentation was written with the help of Claude [5], practical evidence that AI assistants, used well, considerably increase the productivity of a small team.

## References

- [1] Patrick Lewis et al. 'Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020, pp. 9459–9474.
- [2] John Biggs and Catherine Tang. *Teaching for Quality Learning at University*. 4th ed. Maidenhead: McGraw-Hill / Open University Press, 2011.
- [3] Lev S. Vygotsky. *Mind in Society: The Development of Higher Psychological Processes*. Ed. by Michael Cole et al. Cambridge, MA: Harvard University Press, 1978.
- [4] K. Anders Ericsson, Ralf T. Krampe and Clemens Tesch-Römer. 'The Role of Deliberate Practice in the Acquisition of Expert Performance'. In: *Psychological Review* 100.3 (1993), pp. 363–406. DOI: [10.1037/0033-295X.100.3.363](https://doi.org/10.1037/0033-295X.100.3.363).
- [5] Anthropic. *The Claude 3 Model Family: Opus, Sonnet, Haiku*. Tech. rep. Model Card. [https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model\\_Card\\_Claude\\_3.pdf](https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf). Anthropic, 2024.