HELPER

USER:
pastet problematischen Code in Editor

USER:
klickt Analyse:

INTERN:
Helper_InitialPrompt an LLM: User Code mit description (Kommentare oder bezeichnende Namensgebungen etc.)

Response von LLM: Text Hint, Solution, opt. Code Hint

auf Client Seite:
neues ***helperSession*** Objekt mit Daten (was initial im Editor stand für später generate Flashcard zb *initialEditorContent*, Inhalte aus Response: *texthint*, *codehint*, *solution*) -> auf Clientseite, nicht über MongoDB + Server -> Helpersession.js für Klasse Helpersession also im Helpermode bewusst kein Zugriff auf Daten über Server API, weil nur im Helpermode gebraucht und Helpermodezustand wird auch nicht gespeichert

USER: klick Text-Hint/Code-Hint

INTERN: Zugriff über **helperSession** objekt und Ausgabe (z.B. DuggyBuggy sagt Text hint, Codehint steht in dritter uneditierbaren Codebox)

USER: editiert Code and klickt auf Re-Analyse

INTERN:
Helper_AnalysePrompt an LLM: neuer User Code (muss nicht mehr so descriptive sein) und zuvor generierte solution als Referenz

Response: neuer Text-Hint, neuer Code-Hint zugeschnitten auf aktuellen Code im Editor, evtl. zusätzliche Duggy Feedbackmessage, die aber nicht wie ein Texthint ist (z.B. „du bist nah dran“) ;

Überschreibe *texthint*, *codehint* in Objekt ***helperSession***

USER: nach ein paar Iteration von reanalyse usw. hat keine Lust mehr und klickt auf Solution

INTERN: Zugriff auf Lösungscode wieder über ***helperSession*** Objekt und im Editor

USER: entdeckt neu aufgepoppten generateFlashcard Button und klickt

INTERN: neue Flashcard generieren durch:
Helper_GenerateFlashcardPrompt an LLM; Infos für Prompt entnehme ***helperSession*** Objekt (z.B. *initialEditorContent*)

Response: alle Infos, die zur Erstellung der Flashcard benötigt werden

USER: wechselt zu anderer Page

INTERN: Helper Page wird unmounted, inkl. HelperSession objekt, was verschwindet, alles sauber und aufgeräumt

➔ **Brauchen drei Prompts: Helper_InitialPrompt, Helper_AnalysePrompt und Helper_GenerateFlashcardPrompt**

Probleme:

- es ist nicht klar, was das Ziel vom User ist, Solution kann nicht generiert werden.
  (User pastet zB initial eine kompilierende Methode method(int a, int b){return a+b;} rein und LLM weiß gar nicht, was Ziel der Methode ist: soll method addieren? oder soll was anderes mit a und b passieren?) -> wie sehen Hints und Solution aus? kann dieser Code benutzt werden für später generateFlashcard?
  Lösungsvorschlag: falls Unklarheiten da sind, soll LLMs response eine Rückfrage oder Spezifizierungsaufforderung enthalten, Duggy Buggy soll an User übermitteln. User macht refined input (z.B. sprechender Methodenbezeichner oder Kommentare etc) und dann wieder Helper_InitialPrompt an LLM.

Erst wenn klar ist, was das Ziel ist, d.h. LLM das Problem verstanden hat, kann HelperSessionobjekt (für Hints, Solution und Flashcard) und Flashcard zum Problem erstellt werden

- Analyse bei leerem/unzureichendem Editor Inhalt nachdem Solution festgelegt wurde: z.b. { int x = 15; } oder { infjcnsLDKCN } oder {}
  Lösungsvorschlag: wie bei o.g. Zielunklarheit nochmal nachfragen. insbesondere *texthint* und *codeHint* nicht überschreiben

Fragen:

- soll generate Flashcard Button wirklich erst aufpoppen, wenn Solution betätigt wurde? Oder sobald generating Flashcard möglich ist, z.B. wenn Problem von LLM verstanden wurde

TRAINER

USER:
hat in Flashcard Mode auf eine neue Flashcard (im Backlog oder Repeat) geklickt

INTERN:
Zugriff auf Daten (z.B. Aufgabenstring für linke Box *task*) über ***Flashcard.*** Texthint, Codehint und Solution Buttons sind alle schon für User benutzbar: Texthint ist ein generischer Hint von Duggy, Codehint ist zB. pseudocode als comment und/oder Codegerüst in linker Box. -> Felder in ***Flashcard*** *textHint* und *codeHint*
wenn Flashcard in Backlog oder Repeat geöffnet wurde, setze *status* in ***Flashcard*** auf „in Progress“. *triedCount* in ***Flashcard*** um 1 inkrementieren

USER: braucht erst mal einen Hint: Texthint

INTERN: Duggy gibt den Hint. setze *textHintUsed* auf true; *hintsUsed* um eins erhöhen (*textHintUsed*kann nur einmal pro Session auf true gestellt werden. Jedes mal, wenn der User nochmal den Texthint innerhalb der Session anfragt, wird geschaut, ob *textHintUsed* auf true ist, also der hint schon aktiviert wurde. Wenn ja, wird *hintsUsed* nicht inkrementiert -> maximal kann *hintsUsed* um 2 pro Session inkrementiert werden)

USER:
Texthint gibt User Idee. User macht Eingabe in Editor und klickt auf Analyse

INTERN:
(eine neue Session wurde gestartet. Wenn der User aus der Page rausgeht und später wieder auf die begonnene Flashcard geht, soll der Zustand vom letzten analyse geladen werden können.
Wir brauchen also ein Feld in ***Flashcard***, das den Inhalt des Editors beinhaltet, damit der Client den alten Zustand wiederherstellen kann. Zb *editorContent*)

Speichere Editorinhalt von User in ***Flashcard*** in *editorContent*.

Trainer_AnalysePrompt an LLM

Response: Feedback von Editorinhalt als String für Duggy -> entsprechendes Feld *duggyFeedback* und
Feedback, ob Aufgabe gemeistert oder noch nicht (*mastered*, boolean) -> übertrage in ***Flashcard*** Statusänderung in DONE

timesAnalysed in ***Flashcard*** um 1 erhöhen.

falls Eingabe äquivalent zu Solution, setze entsprechendes Feld bei ***Flashcard*** auf Done, falls nicht,

...

USER:
nach paar Iterationen frustiert, klickt auf Solution

INTERN:
zeige *solution* von ***Flashcard*** auf linker Seite. Alle Button (Hints und Analyse) verschwinden. ***Flashcard*** bekommt Flag "repeat" in *status*. Inhalt der Felder used*Texthint* und *usedCodehint* sowie *editorContent* werden wieder auf false gesetzt bzw geleert. -> Flashcardfortschritt wurde zurückgesetzt. User kann die Flashcard in repeat von vorne beginnen.

➔ **Brauchen nur eine Prompt: Trainer_AnalysePrompt**

Probleme:

- bei o.g. Vorschlag würde man den Editorinhalt nur bei "check" in der Flashcard speichern. Ein Speichern vom geschriebenen Code ist also nur in Verbindung mit LLM API call möglich. Was, wenn User seit letztem Ceck schon wieder viel Arbeit in Code reingesteckt hat, aber noch nicht neu checken wollte? Geht aus Trainer raus, wieder rein, Fortschritt weg
  Lösungsvorschlag: zusätzlicher Save Button oder automatisches Speichern alle paar Sekunden oder nach anderem Trigger