Konzept:

Multiple choice Mathe Prüfungsquiz:

Elemente:

* Aufgaben mit je 4 antworten: Analysis, Geometrie/Algebra und Stochastik
* Auswahl zwischen den Aufgabenbereichen + Mix-Aufgaben aus allen Bereichen
* Bei Falscher Antwort Lösungsweg anzeigen + Tipp Button
* Zugänglich für Schüler, online über antonheinrich.github.io/quiz
* Speichern der User mit Punktzahl, bei richtigen Antwort +1
* Begrenzte Antwortzeit 100 sekunden
* Erweiterungen …

**Konzeptübersicht: Multiple-Choice Mathe-Prüfungsquiz**

**1. Ziel des Projekts**

Erstellen einer einfachen, zugänglichen Webanwendung, mit der Schüler Multiple-Choice-Matheaufgaben lösen können. Das System soll Ergebnisse speichern und erweiterbar sein.

**2. Elemente der Webanwendung**

**a) Inhalte**

* **Aufgabenpool**:
  + 50 Multiple-Choice-Aufgaben (Analysis und Stochastik).
  + Jede Aufgabe hat:
    - Eine Fragestellung.
    - 4 mögliche Antworten, von denen eine korrekt ist.
* **Erweiterbarkeit**:
  + Möglichkeit, neue Aufgaben hinzuzufügen (z. B. Geometrie, Matrizen).

**b) Funktionen**

1. **Quiz-Logik:**
   * Anzeige der Aufgaben nacheinander.
   * Auswahl einer Antwort aus vier Optionen.
   * Zeitlimit von 20 Sekunden pro Aufgabe (Timer).
   * Nach Ablauf der Zeit wird die Antwort als falsch gewertet.
2. **Punkteberechnung:**
   * Richtige Antwort: +1 Punkt.
   * Falsche Antwort: -1 Punkt.
   * Punktestand wird nach jeder Aufgabe aktualisiert und angezeigt.
3. **Ergebnisanzeige:**
   * Endergebnis nach Beantwortung aller Fragen.
   * Optional: Feedback (z. B. „Super gemacht!“ oder „Übe weiter!“).

**c) Benutzerverwaltung**

* **Anmeldung (optional):**
  + Nutzername (kein Passwort nötig für Einfachheit).
* **Speicherung der Ergebnisse:**
  + Punktestand wird mit dem Nutzernamen verknüpft.
  + Ergebnisse bleiben auch nach Schließen der Seite verfügbar.

**d) Zeitlimit**

* Countdown-Timer pro Frage (20 Sekunden).
* Automatisches Überspringen zur nächsten Frage, wenn Zeit abgelaufen ist.

**e) Design und Benutzeroberfläche**

* Minimalistisches Layout:
  + **Startseite**: Willkommenstext, Anmeldeoption, und Quiz starten.
  + **Quizseite**: Eine Frage mit vier Antwortbuttons und einem Timer.
  + **Ergebnisseite**: Punktestand und Möglichkeit, erneut zu spielen.
* Responsives Design für Mobilgeräte und PCs.

**f) Erweiterungsmöglichkeiten**

* Themenauswahl: Verschiedene Themengebiete (Analysis, Stochastik, etc.).
* Schwierigkeitsstufen: Einfach, Mittel, Schwer.
* Highscore-Liste: Bestenliste der Nutzer.
* Lernmodus: Erläuterungen zu den Antworten nach jeder Frage.

**3. Technische Anforderungen**

**Frontend (Benutzeroberfläche):**

* HTML: Aufbau der Seitenstruktur.
* CSS: Design (Farben, Layout, Timer).
* JavaScript: Quiz-Logik, Timer, Punkteberechnung.

**4. Aufgabenübersicht für die Umsetzung**

**Schritt 1: Aufgaben und Inhalte vorbereiten**

* Erstelle 50 Fragen mit Antworten (Analysis & Stochastik).
* Formatiere die Fragen in einem strukturierten Format (z. B. JSON).

**Schritt 2: Benutzeroberfläche erstellen**

* Startseite, Quizseite, Ergebnisanzeige.
* Timer und Interaktivität hinzufügen.

**Schritt 3: Punkte- und Ergebnislogik**

* Funktion, die Punktestand aktualisiert.
* Feedback für richtige und falsche Antworten.

**Schritt 4: Erweiterungen vorbereiten**

* Benutzerregistrierung und Speicherung der Ergebnisse.
* Themenauswahl und neue Kategorien hinzufügen.

**Schritt 5: Testen und Optimieren**

* Teste die App auf verschiedenen Geräten (PC, Smartphone).
* Überprüfe, ob Timer, Fragen, und Ergebnisse korrekt funktionieren.

**5. Erweiterungen für später**

* Authentifizierung mit E-Mail oder Google.
* Multiplayer-Modus: Schüler können gegeneinander antreten.
* Integration eines KI-basierten Empfehlungssystems für spezifische Übungsaufgaben.

Möchtest du bei einem der Punkte ins Detail gehen? Zum Beispiel das Aufgabenformat oder die Logik für den Timer? 😊