

Elyas Binothman • Radical AI

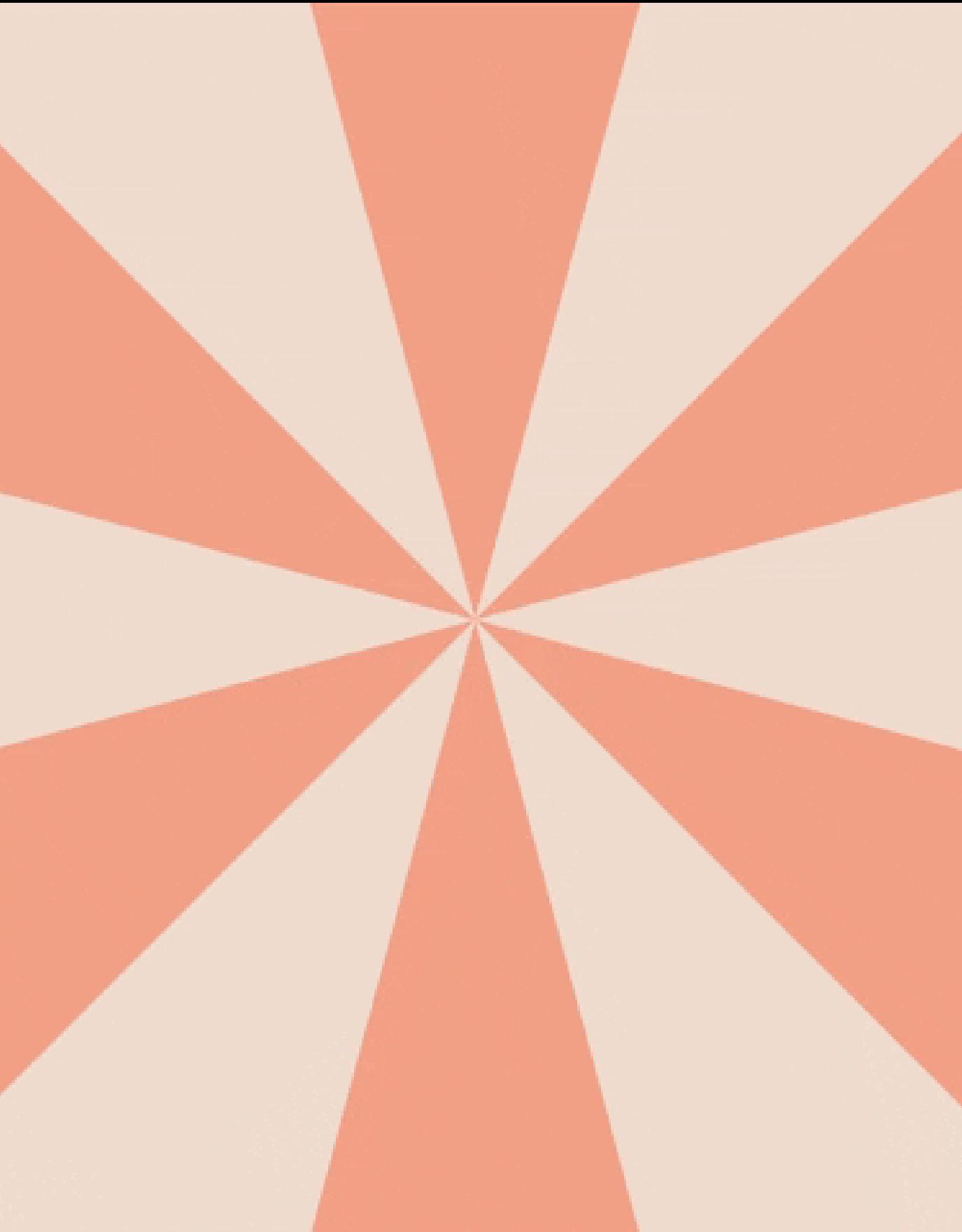
# GEMINI QUIZIFY QUIZ GENERATOR

# Introduction

Welcome, everyone.

This project focuses on generating quiz questions from user-uploaded documents.

I'll cover the approach, key features, and results.



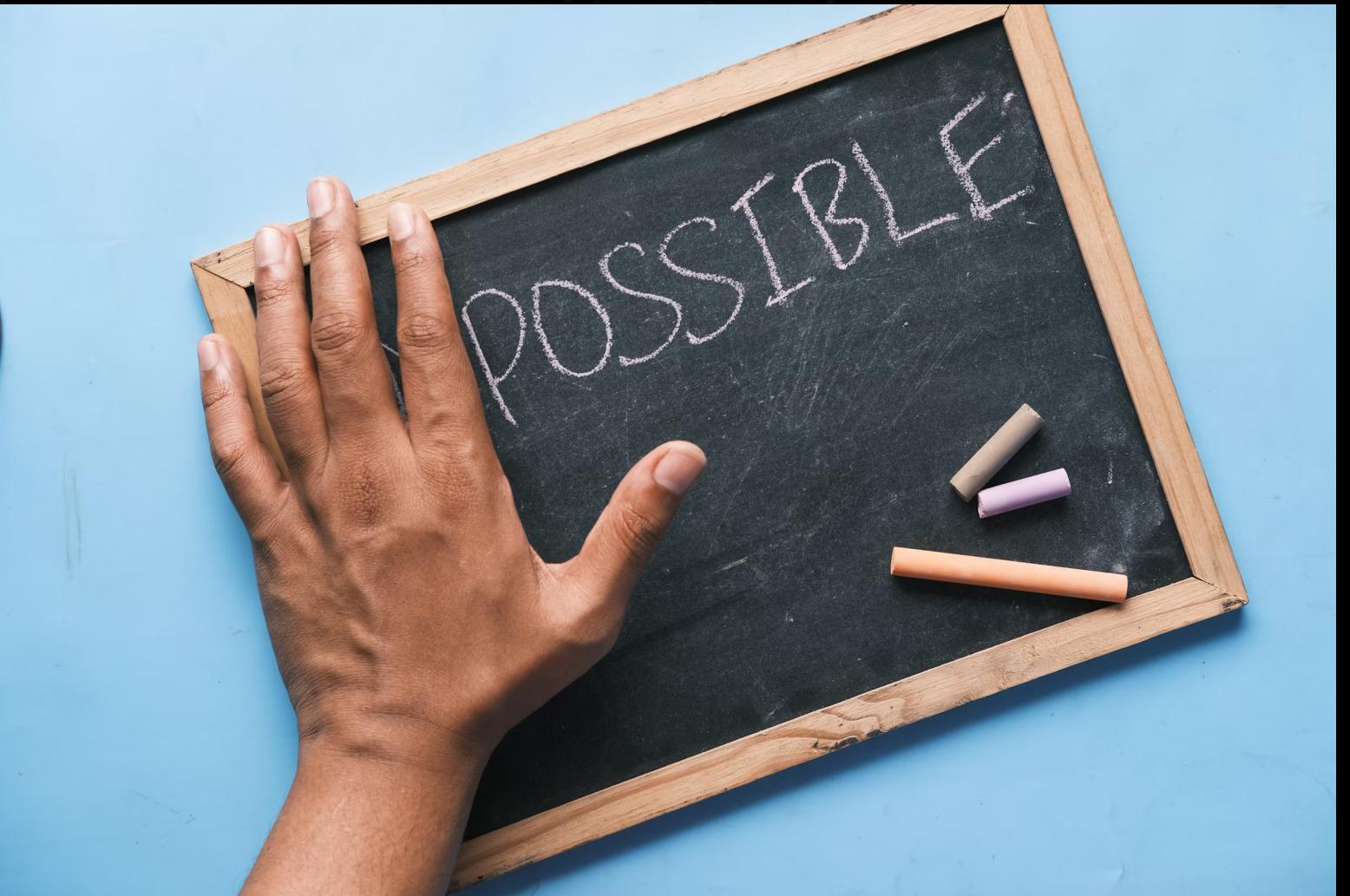
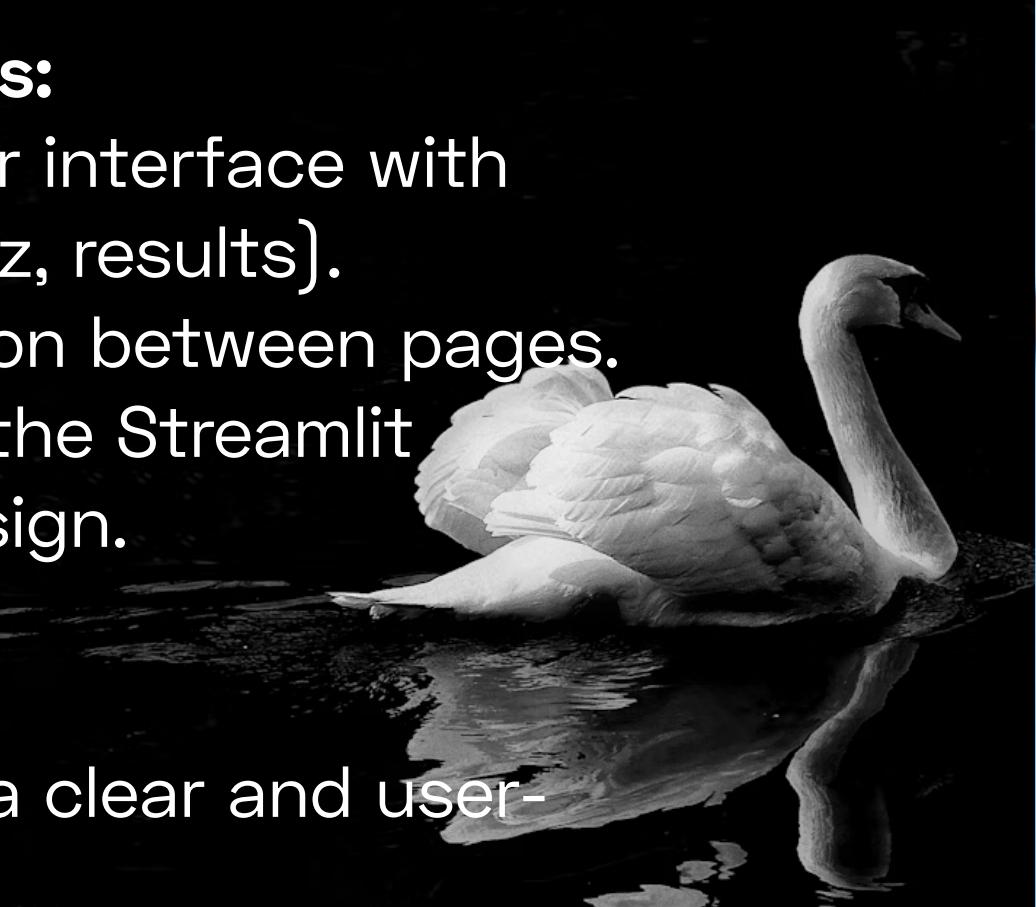
# The Challenge

## Application Layout Challenges:

- Designing an intuitive user interface with multiple pages (home, quiz, results).
- Ensuring smooth navigation between pages.
- Learning and integrating the Streamlit library for effective UI design.

## Adding Results Page:

- Displaying quiz results in a clear and user-friendly manner.
- Calculating and showing correct and incorrect answers with explanations.
- Challenge of learning Streamlit for dynamic updates and interactivity.



# Application layout

## 1. Multi-Page Navigation:

- **Pages:**
  - **Home:** Users upload documents and specify quiz parameters.
  - **Quiz:** Users answer the generated quiz questions.
  - **Results:** Users review their results and explanations for each question.
- **Benefit:** Ensures a smooth and intuitive workflow from document upload to quiz completion.

## 2. Easy Navigation:

- **Sidebar Navigation:**
  - **Description:** A sidebar with navigation buttons for easy access to different application sections.
  - **Buttons:** Home, Next Question, Previous question and Results (only if the user answered all of the questions).
- **Benefit:** Allows users to easily navigate between pages and keep track of their progress.

## 3. Dynamic Updates:

- **Real-Time Interaction:**
  - **Description:** The application dynamically updates based on user inputs and actions.
  - **Examples:** Generating questions after document upload, displaying results after quiz completion.

The Quiz Builder interface shows a dark-themed dashboard. At the top, there's a navigation bar with a 'Home' button. Below it is a 'Quiz Builder' section with a sub-header 'Select PDFs for Ingestion, the topic for the quiz, and click Generate!'. It features a 'Drag and drop files here' area with a 200MB limit, where 'Lecture 7 Probability.pdf' (0.6MB) is listed. There's also a 'Browse files' button. Underneath, a 'Topic for Generative Quiz' input field contains 'probability'. A slider for 'Number of Questions' is set to 4, with a range from 1 to 10. A 'Submit' button is present. Below the form, two green success messages are displayed: 'Successfully split documents to 18 pages!' and 'Successfully created Chroma Collection!'. A status message at the bottom says 'Generating 4 questions for topic: probability'.

The Quiz Results page has a header 'Quiz Results'. It displays a message 'You got 50.00%' and 'You answered 2 out of 4 questions correctly.' Below this, a question is shown: 'Question 1: What is the definition of probabilistic inference?'. A red error box indicates 'Incorrect!' with the message 'Your answer: A) Computing a desired probability from other unknown probabilities.' A correct answer is given as 'B'. An explanation follows: 'Explanation: Probabilistic inference involves computing a desired probability from other known probabilities, such as calculating a conditional probability from a joint probability distribution. This process is crucial for agents to update their beliefs and make informed decisions based on new evidence.' Another question is partially visible below: 'Question 2: What is probabilistic inference?' with a green success box indicating 'Correct!'.

```
def next_page():
    st.session_state.page += 1

Codeium: Refactor | Explain | Generate Docstring | X
def previous_page():
    st.session_state.page -= 1

Codeium: Refactor | Explain | Generate Docstring | X
def go_home():
    st.session_state.page = 0
    st.session_state['question_bank'] = []
    st.session_state['display_quiz'] = False
    st.session_state['question_index'] = 0
    st.session_state.correct_answers = 0
    st.session_state.answers = {}
```

## Generated Quiz Question:

3. What is the probability that a fair six-sided die will land on a number greater than 3?

Choose an answer

A) 1/6  
 B) 1/2  
 C) 2/3  
 D) 5/6

Submit Answer  
Next Question  
Previous Question  
Results

# Approach

## 1. User Input:

- Users upload their PDF documents through an intuitive interface.

## 2. Document Processing:

- The system extracts text from the uploaded PDFs for analysis.

## 3. Embedding Generation:

- Extracted text is converted into embeddings to understand the content.

## 4. Quiz Generation:

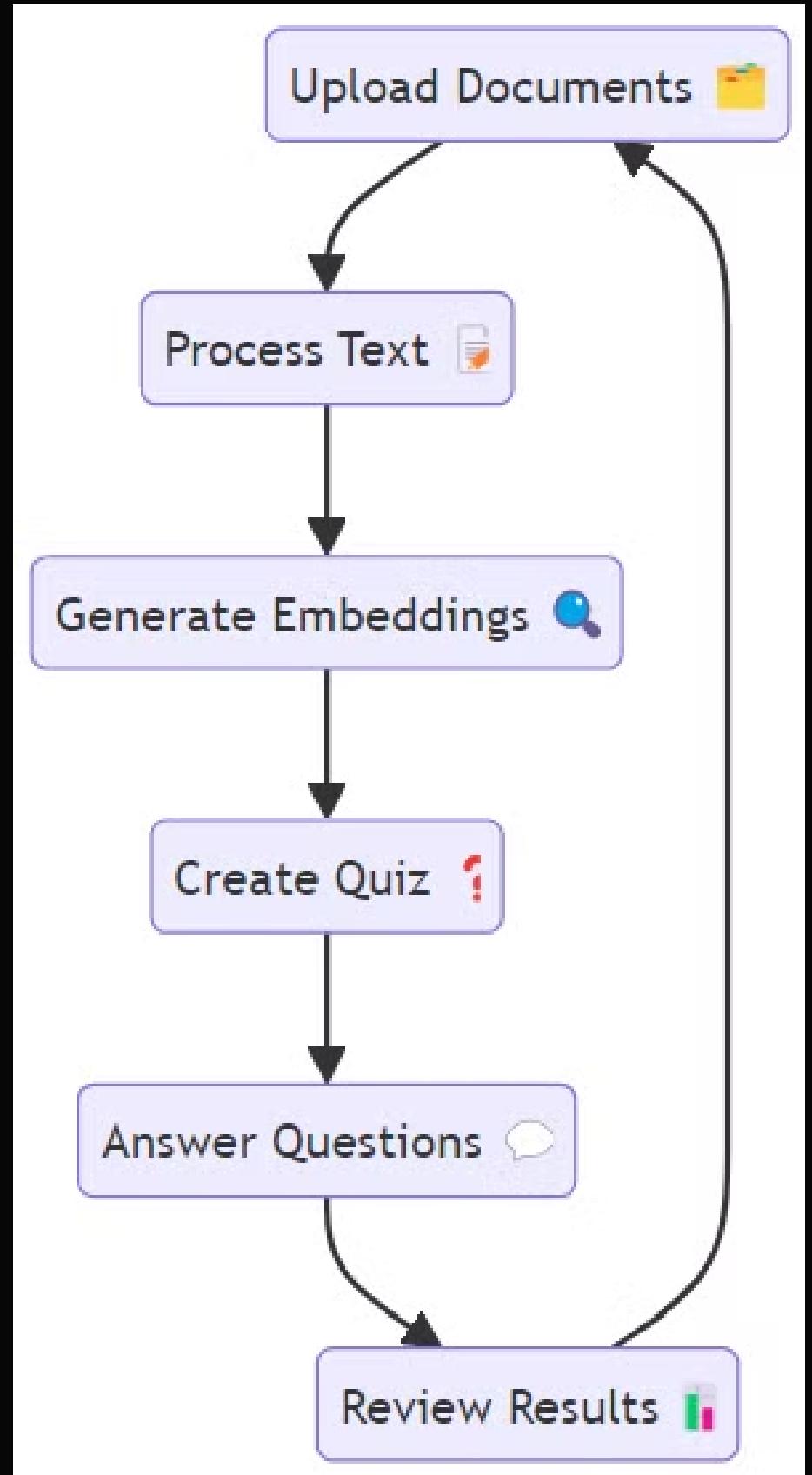
- Based on the topic and number of questions specified by the user, the system generates quiz questions.

## 5. User Interaction:

- Users interact with the generated quiz by answering the questions.

## 6. Review and Results:

- **New Feature:** After answering all questions, there will be a results page allowing users to review their performance.



# Key features

## Content:

### 1. Automated Quiz Generation:

- **Description:** Streamlined process that automatically generates quiz questions from uploaded documents.

### 2. User-Friendly Interface:

- **Description:** Intuitive and easy-to-use interface for uploading documents, specifying quiz parameters, and navigating between quiz pages.

### 3. Locking answers:

- **Description:** After the user submits the answer the question is then locked and the user cannot change their answer and can only review it on the results page.

### 4. Result Button and Detailed Results Page:

- **Description:** The result button appearing after answering the questions which will show a comprehensive results page that shows the percentage of correct answers, detailed explanations, and overall performance.

```
if 'submitted_answers' not in st.session_state:  
    st.session_state.submitted_answers = 0
```

```
if st.form_submit_button("Submit Answer") and answer is not None:  
    st.session_state.answers[st.session_state["question_index"]] = answer  
    st.session_state.submitted_answers += 1
```

```
if st.session_state.submitted_answers >= len(st.session_state['question_bank']):  
    if st.button("Results"):  
        next_page()  
        st.rerun()
```

# DEMONSTRATION



THANK  
YOU



# Want to make a presentation like this one?

Start with a fully customizable template, create a beautiful deck in minutes, then easily share it with anyone.

[Create a presentation \(It's free\)](#)