

# AI-Driven Personalized Learning Platform

Adam Hisel, Carter Parks, Morgan Prieskorn, Zac  
Mueterthies, Ryan Johnson



SD-14

1

# 01

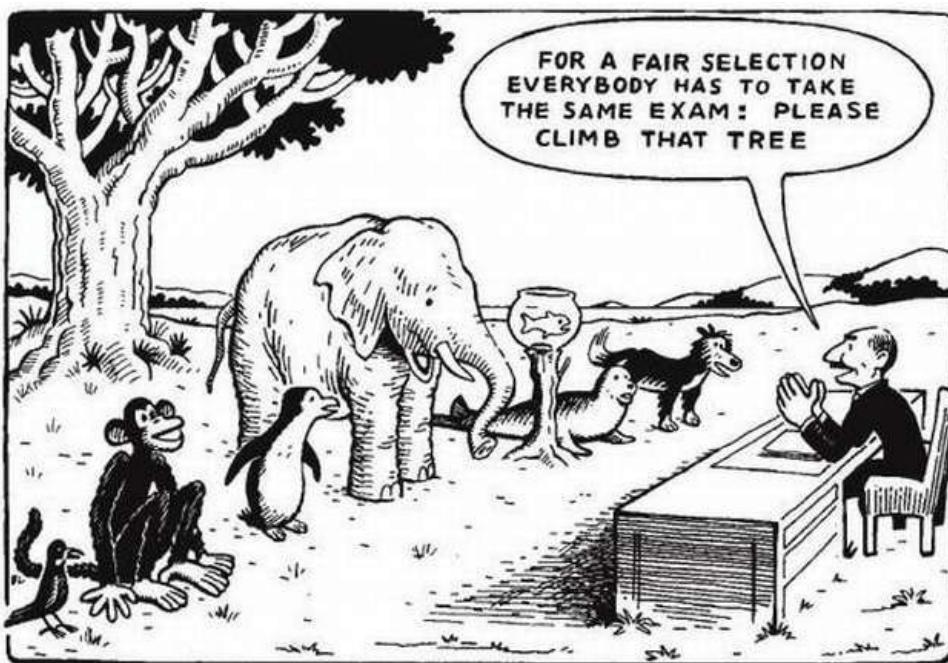
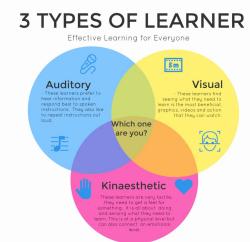
## Problem Being Addressed

2

# Problem

3

- Traditional and online learning use a **one-size-fits-all approach** that ignores different learning styles.
- **Lack of personalization** leads to disengagement, poor retention, and frustration.
- Students need a **learning system that adapts** to their unique preferences and challenges.



4

# 02

## Issues

5

### Issue 1 - Personalized Content Generation

6

#### **Issue:**

- Traditional learning materials (PDFs, videos, lectures) do not dynamically adjust to individual needs.
- Manually personalizing content for each learner is time-consuming and unrealistic at scale.

#### **Solution:**

- Use **AI that adapts content** (prompt engineering + RAG) to transform learning materials:
  - Convert text-based materials (ebooks, PDFs) into customized learning modules.
  - Generate interactive learning modules (quizzes, flashcards, video explanations).
  - Personalize examples and quiz questions based on the user's interests

## Issue 2 - AI Accuracy & Reliability

### Issue:

- AI-generated content must be accurate, relevant, and aligned with the subject matter

### Solution:

- RAG (Retrieval-Augmented Generation) AI approach:**

- AI retrieves verified sources before generating responses
- Real-time validation using educational databases and reference textbooks.
- Continuous human feedback and user reporting to improve AI accuracy.

The RAG process



## Issue 3 - Scalability & Performance

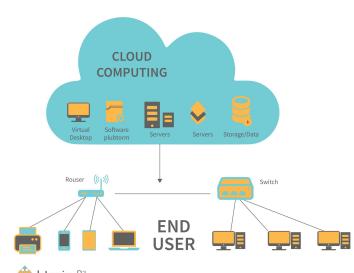
### Issue:

- Processing large amounts of data (ebooks, user inputs, learning history) requires efficient computing power.

### Solution:

- Cloud-based architecture** to handle data and scale efficiently.

- As our program gets larger with more educational dashboards the cloud can store that data on demand
- Allows for secure storage and traffic management to improve performance



# 03

## Tools and Practices

9

### Tools

10

Frontend Development:



PostgreSQL Database:



Hosting and Deployment:



Firebase

Containerization:



# Practices

11

Practices	Description
Agile Development	We follow an iterative and flexible development process (sprint plan), making adjustments based on feedback and testing.
CI/CD	Automated deployment and testing using Vercel & Docker, ensuring stable and frequent updates.
Learning Persona	Understanding how each user learns best; based on performance and feedback.
Prompt Engineering	Used to refine AI-generated quizzes, explanations, and personalized content formats.
RAG	Used to process ebooks and generate personalized content with accurate, structured learning paths (same Info-different styles).

# 04

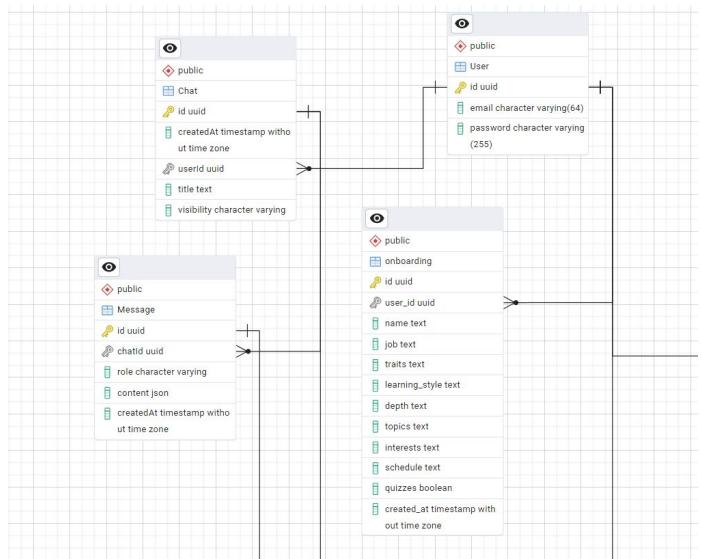
## Current Progress and Challenges

12

# Current Progress (Database)

13

- **Operational Tables:**
  - User
  - Chat
  - Message
  - Onboarding
- API is set up to successfully handle data posting and retrieval, ensuring seamless communication between the frontend and backend.  
(Authentication, chats, persona)



# Current Progress (UI)

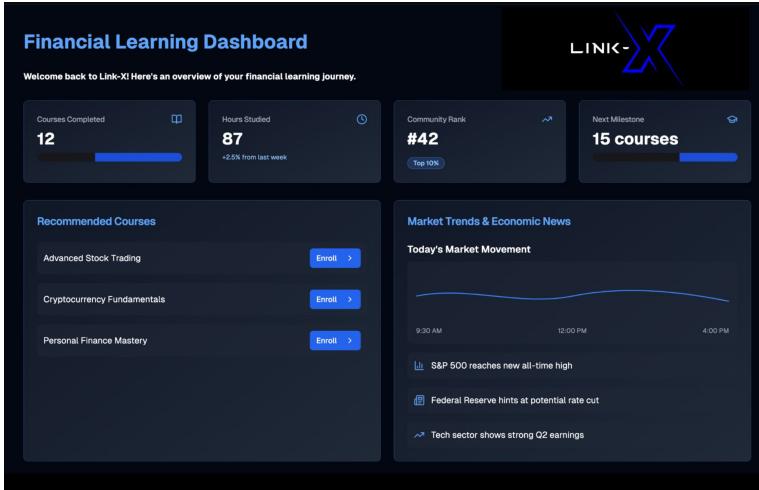
14

- **Major Pages Implemented:** We have successfully developed four of our key pages.
- **Standardized Design:** Our goal is to align these pages with industry best practices for usability and aesthetics.
- **Client Collaboration:** We are actively working with the client to refine the design based on their vision and feedback.
- **Design Tools:** We are leveraging V0 by Vercel to streamline the design process and ensure consistency.



# UI Components

15



User Dashboard

This form is titled 'Personalized Learning Setup'. It consists of several input fields: 'What should Link-X call you?' (text input), 'What do you do?' (text input with placeholder 'e.g., Student, Engineer'), 'What traits should Link-X have?' (text input with placeholder 'e.g., witty, encouraging'), 'Preferred Learning Style' (dropdown menu with placeholder 'Select a learning style'), 'Depth of Explanation' (dropdown menu with placeholder 'Select depth'), 'Topics of Interest' (text input with placeholder 'e.g., Investing, Finance'), 'Interests, Values, or Preferences for Personalization' (text input with placeholder 'e.g., Basketball, Video Games'), 'Preferred Study Schedule' (dropdown menu with placeholder 'Select schedule'), and a checkbox for 'Include quizzes for progress tracking'.

User Onboarding

# UI Components

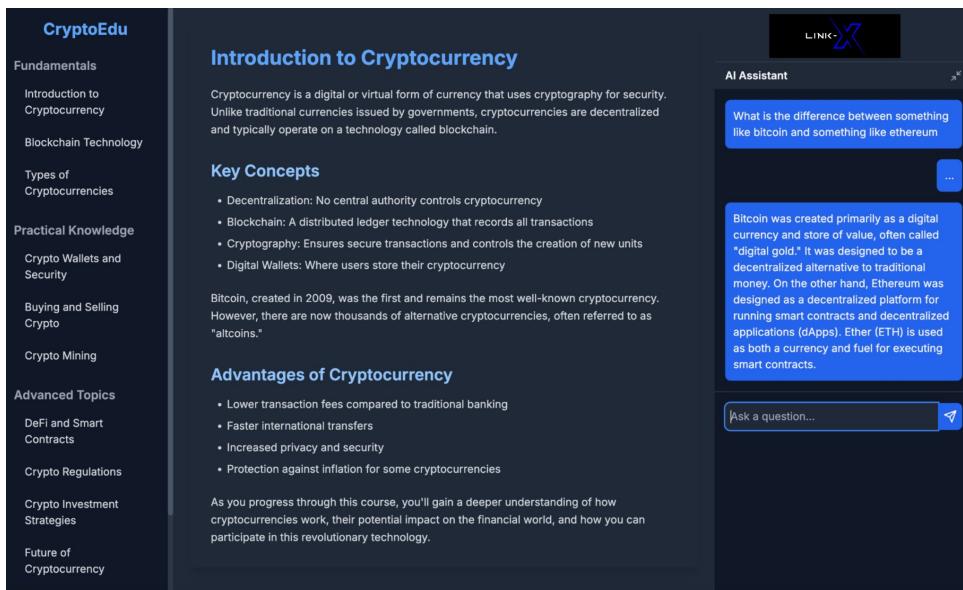
16



Modular Navbar Component

# Course Screen Sketch

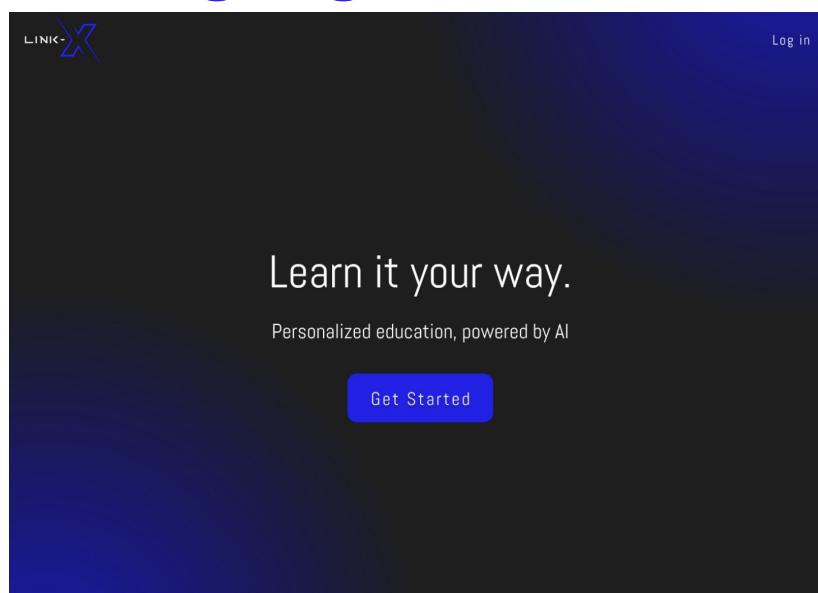
17



The screenshot shows a course page titled "Introduction to Cryptocurrency". On the left, there's a sidebar with a navigation menu for "CryptoEdu" containing sections like Fundamentals, Practical Knowledge, and Advanced Topics. The main content area has a heading "Key Concepts" with a bulleted list about decentralization, blockchain, cryptography, and digital wallets. Below that is a section titled "Advantages of Cryptocurrency" with another bulleted list. A sidebar on the right is titled "AI Assistant" and contains a question "What is the difference between something like bitcoin and something like ethereum" and a detailed answer about the differences between Bitcoin and Ethereum. At the bottom right is a "Ask a question..." input field.

# Landing Page Screen Sketch

18



The screenshot shows a landing page with a dark blue gradient background. At the top left is the "LINK-X" logo. At the top right is a "Log in" button. In the center, the text "Learn it your way." is displayed above the tagline "Personalized education, powered by AI". At the bottom center is a large blue "Get Started" button.

# Challenges (Codebase & Design)

19

- **Adapting to the Existing Codebase:** Understanding the structure, dependencies, and best practices for working within it.
- **Running the Backend Properly:** Initial setup issues, API key integration for OpenAI, and backend filing confusions.
- **Refactoring & Optimization:** Cleaning up and restructuring code to improve maintainability and efficiency.
- **Website Design:** Working with our clients to design a professional website that exceeds their expectations.



# Challenges (AI & Prompt Engineering)<sup>20</sup>

- **Limited Experience in ML & Prompt Engineering:** Learning how to optimize prompts and integrate AI effectively.
- **Configuration Hurdles:** Understanding how to fine-tune responses and manage OpenAI's API parameters.
- **Ongoing Research & Experimentation:** Finding the best strategies to implement prompt engineering within our system.



# 05

## Remaining Work Timeline

21

## Sprint Plan

22

Week Starting	Sprint Plan	
2/2/2025	Onboarding and understanding code base	
2/9/2025	Onboarding and understanding code base	
2/16/2025	Account creation and Authentication	
2/23/2025	Persona Creation and Database	
3/2/2025	Landing Page and Dashboard	
3/9/2025	Topics Page and Study Page	
3/16/2025	Spring Break	
3/23/2025	Testing Personalization & Backend	UI & Onboarding Complete
3/30/2025	Testing Personalization & Backend	
4/6/2025	Make Database Backend to save personalization progress	
4/13/2025	Testing RAG for Uploading pdfs	Have Working Personalization for any topic
4/20/2025	Certificate Report for completion	
4/27/2025	Deployment	
5/4/2025	Extra ~ Analytics for Dashboard (user progress)	
5/11/2025	Finals Week	

# Future Plans: Dashboard & Study Page <sup>23</sup>

- **Enhancing Dashboard Functionality:** Improve user experience by adding interactive elements, progress bars, and dynamic data displays.
- **Expanding Study Page Features:** Implement better organization, filtering options, and personalized study recommendations.
- **Seamless Integration:** Ensure all pages work cohesively with existing components for a smooth workflow.



# Future Plans: Universal Styling <sup>24</sup>

- **Consistent UI/UX:** Apply a uniform design system across all pages for a professional and cohesive look.
- **Theme & Responsiveness:** Ensure accessibility across different devices and screen sizes.
- **Efficiency & Maintainability:** Use reusable styling components to simplify future development.



# Future Plans: Prompt Engineering for Personas<sup>25</sup>

- **Persona-Specific Responses:** Fine-tune AI behavior to align with different user needs and expectations.
- **Adaptive Learning:** Improve model accuracy based on user interactions and feedback.
- **Optimized API Usage:** Experiment with various prompt techniques to enhance response quality and efficiency.



# 06

## Demo

# THANK YOU



# AI-Driven Personalized Learning Platform

Adam Hisel, Carter Parks, Morgan Prieskorn, Zac  
Mueterthies, Ryan Johnson



SD-14

1

# 01

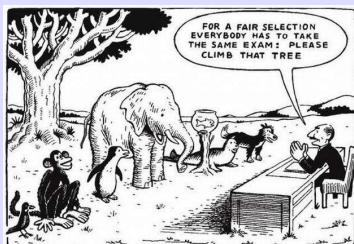
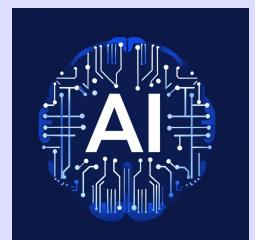
## Topic

2

# Topic

3

- Goal: Personalized supplemental learning
- Solution: AI-powered content
- **"Learn the way you want." - LINK-X**



# 02

## Design & Implementation

4

# Styling And Feel

- Universal Styling
- Business / Educational Feel

# Solution (Quick Overview)

- Gather Persona (Onboarding)
- Course Topic and Expertise Level
- Generate Course Outline
- Content with Persona
- Interactive (smart) chat

# Prompt 1

## Issue:

- Gather Learning Subject
- Based on knowledge/experience

## User Input:

- e.g., "I'm a sophomore in finance and I want to learn about investing"

```
data = request.form or request.get_json()
question = data.get("question")
if not question:
    return jsonify({"error": "Missing question"}), 400

# Use OpenAI to extract topic and expertise
client = openai.Client(api_key=os.getenv("OPENAI_API_KEY"))
response = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {
            "role": "system",
            "content": (
                "You are an education assistant. Extract a topic and the user's level of expertise from the question."
                "Reply ONLY with a JSON object containing \"topic\" and \"expertise\" (one of: beginner, intermediate, advanced)."
            )
        },
        {"role": "user", "content": question}
    ]
)
import json
parsed = json.loads(response.choices[0].message.content)
topic = parsed.get("topic")
expertise = parsed.get("expertise")
if not topic or not expertise:
    return jsonify({"error": "Invalid API response"}), 400
outline = generate_course_outline(topic, expertise)
```

7

# Prompt 2

## Issue:

- When generating a course, first need an outline of the modules

## Prompt:

- You are an AI Assistant ...
  - user has provided a topic: {topic} and their expertise on the subject: {expertise} ...
  - pdf possibly included (FAISS & RAG)
  - task is to retrieve all relevant content based on their expertise and summarize it ...
  - For each chapter:
    - Provide a concise title (3-7 words)
    - Include an array of relevant metadata or key points
  - return as JSON"

```
{
  "chapters": [
    {
      "chapterTitle": "string",
      "metadata": [
        "string",
        "string",
        ...
      ],
      ...
    }
  ],
  ...
}
```

8

# FAISS

**Issue:**

- Convert PDFs into a Knowledge-Base(KB)

**FAISS:**

- FaceBook AI Similarity Search
  - PDF divided into chunks & stored as vector embeddings
  - Generate references & citations
  - Store in index.faiss & index.pkl

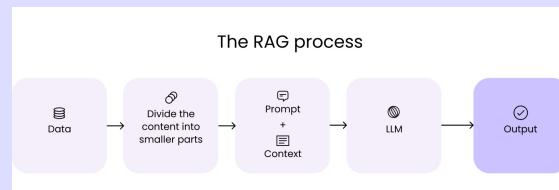
# RAG

**Issue:**

- Use vector embeddings for course creation
- Extract relevant chunks

**RAG:**

- Retrieval-Augmented Generation
  - LLM performs retrievals from FAISS KB
  - Generates relevant knowledge based on prompt & KB



## Prompt 3

**Issue:**

- For each module, the content needs to be generated by the LLM using Persona & RAG

**Prompt:**

- "You are a helpful and friendly AI tutor. {full\_persona}{expertise}...
  - *full\_persona*: Name, traits, learning style, interests, etc
  - *expertise*: Description of knowledge level
  - Now explain this topic: {module\_content}"
  - *module\_content*: Title & key points

```
expertise_map = {
    "beginner": "They prefer simple, clear explanations suitable for someone new to the topic.",
    "intermediate": "They have some prior experience and prefer moderate technical depth.",
    "advanced": "They want in-depth explanations with technical language."
}
```

## Prompt 4

**Issue:**

- Questions asked in the AI chat should be relevant to the current course

**Prompt:**

- "You are an AI assistant ...
  - access to a knowledge base on a topic ...
  - provided the following query: {user\_query} ...
  - task is to:
    - query is relevant → respond with a concise answer (w/ RAG & Persona)
    - query is **not** relevant → 'Query is not relevant to the Course. Please try again.'
- Output should only provide one of these two outcomes based on the relevance to content"

# Prompts Put Together (Frontend)

13

The screenshot shows a mobile application interface for an AI Assistant. At the top, there's a navigation bar with a back arrow and the title "AI Assistant". Below it is a blue input field containing the text "Tell me a little more about AI perception". The main area is a dark-themed chat window. On the left, a user message reads: "Hey Adam! Ready to flex those mental muscles and dive into the all-star game of Artificial Intelligence? Let's break it down like a tactical playbook!". On the right, the AI's response is: "I Intelligence? Let's revolutionizing how we ic human intelligence reasoning (using". A red arrow points from the user's first message to the AI's response. Below the main window, there's a footer with a green "Upload PDF" button and a blue "Learn" button.

Prompt 3:  
I am a Freshmen in Computer Science

Prompt 4:  
**Definition of art**

Hey Adam! Ready to flex those mental muscles and dive into the all-star game of Artificial Intelligence? Let's break it down like a tactical playbook!

Artificial Intelligence, often abbreviated as AI, is like the MVP of the tech world, and it's revolutionizing how we interact with technology. In essence, AI refers to computer systems or machines that mimic human intelligence processes. These processes include learning (acquiring information and rules for using it), reasoning (using rules to reach approximate or definite conclusions), and self-correction.

1. **\*\*Learning:\*\*** Just like a basketball player fine-tunes their jump shot through practice, AI systems improve through experience. This is where machine learning (ML) comes into play, which involves algorithms that allow systems to learn from data. The system receives inputs, like a coach's feedback, and adjusts its actions to improve results.
2. **\*\*Reasoning:\*\*** Think of this as the playbook of strategies that a quarterback memorizes for decision-making on the field. Similarly, AI systems apply rules and logical reasoning to process information and make decisions.
3. **\*\*Perception:\*\*** AI's ability to interpret inputs from the world, akin to how athletes use their senses during a game. For example, computer vision allows AI to 'see' and understand images, which is like a player visualizing the field or court.
4. **\*\*Language Understanding:\*\*** Just like understanding the communication among teammates, AI has Natural Language Processing (NLP) to understand and generate human language. This enables AI to interface with users in a natural and meaningful way.

AI can be broadly classified into two types: **"Narrow AI"** and **"General AI"**.

- **"Narrow AI"**: This is like a specialized athlete, highly skilled in one area, such as AI systems that excel at playing chess or identifying objects in images.

# Final Learning Page

14

The screenshot shows a final learning page with a sidebar menu on the left and a main content area on the right.

**Sidebar Menu:**

- LINK-X
- INTRODUCTION TO ARTIFICIAL INTELLIGENCE
  - Definition of artificial intelligence
  - Brief history of AI development
  - Importance and applications of AI
- TYPES OF ARTIFICIAL INTELLIGENCE
  - Narrow AI vs. General AI
  - Reactive machines, limited memory
  - Examples of each type in real-world applications
- MACHINE LEARNING BASICS
  - Definition and significance of machine learning
  - Supervised, unsupervised, and reinforcement learning
  - Common algorithms used in ML
- DEEP LEARNING EXPLAINED
  - Overview of deep learning and neural networks

**Main Content Area:**

## Definition of artificial intelligence (AI)

Hey Adam! Ready to flex those mental muscles and dive into the all-star game of Artificial Intelligence? Let's break it down like a tactical playbook!

Artificial Intelligence, often abbreviated as AI, is like the MVP of the tech world, and it's revolutionizing how we interact with technology. In essence, AI refers to computer systems or machines that mimic human intelligence processes. These processes include learning (acquiring information and rules for using it), reasoning (using rules to reach approximate or definite conclusions), and self-correction.

1. **\*\*Learning:\*\*** Just like a basketball player fine-tunes their jump shot through practice, AI systems improve through experience. This is where machine learning (ML) comes into play, which involves algorithms that allow systems to learn from data. The system receives inputs, like a coach's feedback, and adjusts its actions to improve results.
2. **\*\*Reasoning:\*\*** Think of this as the playbook of strategies that a quarterback memorizes for decision-making on the field. Similarly, AI systems apply rules and logical reasoning to process information and make decisions.
3. **\*\*Perception:\*\*** AI's ability to interpret inputs from the world, akin to how athletes use their senses during a game. For example, computer vision allows AI to 'see' and understand images, which is like a player visualizing the field or court.
4. **\*\*Language Understanding:\*\*** Just like understanding the communication among teammates, AI has Natural Language Processing (NLP) to understand and generate human language. This enables AI to interface with users in a natural and meaningful way.

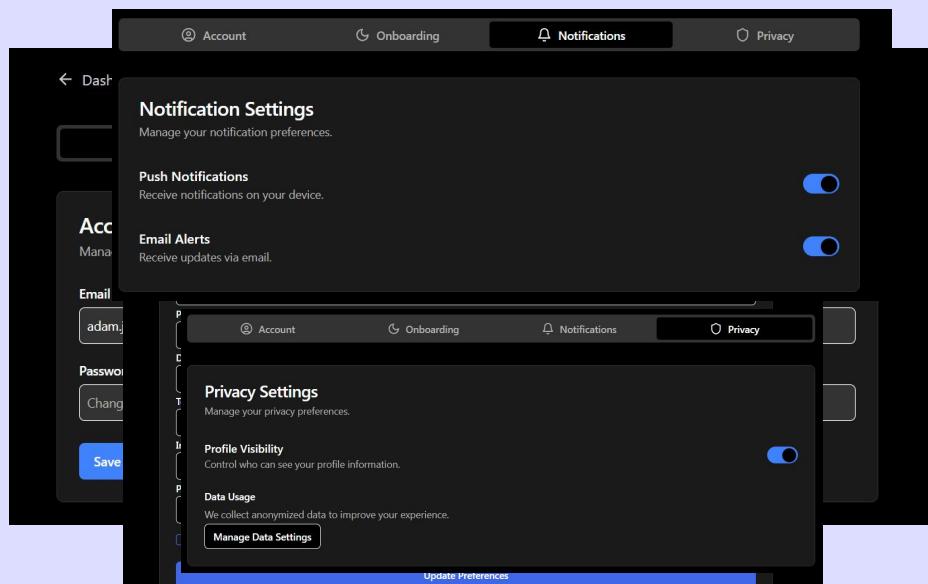
AI can be broadly classified into two types: **"Narrow AI"** and **"General AI"**.

- **"Narrow AI"**: This is like a specialized athlete, highly skilled in one area, such as AI systems that excel at playing chess or identifying objects in images.

The right side of the screen features an AI Assistant interface with a blue input field "Tell me a little more about AI perception" and a detailed AI response below it.

# Settings Page

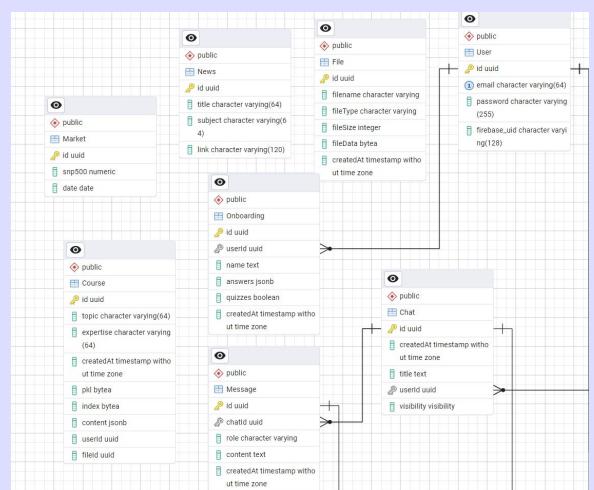
15



# Database Current Progress

16

Identifier	Providers	Created	Signed In	User UID
morgan4@gmail.com	✉️	Apr 14, 2025	Apr 14, 2025	i6zIHuJGfkgeIYYGNtFGJabN...
morgan5@gmail.com	✉️	Apr 14, 2025	Apr 14, 2025	QzqSwLGFVLvGuF0TUR02db...
test2@iastate.edu	✉️	Apr 14, 2025	Apr 14, 2025	ttVvwsRhNCYRXEDoiMUnknj...
practice@iastate.edu	✉️	Apr 14, 2025	Apr 14, 2025	AiC7LIGu4VfNMu3mbkG0S5...
adamtest@iastate.edu	✉️	Apr 14, 2025	Apr 14, 2025	M4AiOq7dPjgXZUwnbWQAH...
test@iastate.edu	✉️	Apr 14, 2025	Apr 14, 2025	eSCNFBv6Z0gh0ZnJ7MPMF4...
a@iastate.edu	✉️	Apr 14, 2025	Apr 14, 2025	JnglSG6ZXcLN0tJRFV59rsH...
adamhisel12@yahoo.c...	✉️	Apr 14, 2025	Apr 14, 2025	mjV6sLpJHTaKuSYMjRP4cpcl...
carterp@iastate.edu	✉️	Apr 13, 2025	Apr 14, 2025	rJxr50IDqeew9c1b5v5NIVEQp...
demo@iastate.edu	✉️	Apr 13, 2025	Apr 14, 2025	GaduYsgGfPZPanDfmkUHAQ...
carter.p@iastate.edu	✉️	Apr 13, 2025	Apr 13, 2025	rXat1AvoxSZLyJnuy3y5UN0g...
ahisel@iastate.edu	✉️	Apr 13, 2025	Apr 13, 2025	rpuguqlfMzgqf3ZuBQADS9NY...
parks@iastate.edu	✉️	Apr 13, 2025	Apr 13, 2025	Ay5rdvU7lyzdPiY0nmvVEar0...



```

# Generate a course from a user question.
# Store the course in the Postgres database.
# Return the course id for use in the frontend.
@app.route('/create-course', methods=['POST'])
def learn_from_question():
    try:
        # Verify session cookie
        user = get_user_from_session()
        if "error" in user:
            return jsonify(user), 401

        # Get question from request body
        data = request.form or request.get_json()
        question = data.get("question")
        if not question:
            return jsonify({"error": "Missing question"}), 400

        # Use OpenAI to extract topic and expertise
        client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))
        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages=[
                {
                    "role": "system",
                    "content": (
                        "You are an education assistant. Extract a topic and the user's level of expertise. Reply ONLY with a JSON object containing 'topic' and 'expertise' (one of: basic, intermediate, advanced)."
                    )
                },
                {"role": "user", "content": question}
            ]
        )
        import json
        parsed = json.loads(response.choices[0].message.content)
        topic = parsed.get("topic")
        expertise = parsed.get("expertise")
        if not topic or not expertise:
            return jsonify({"error": "Invalid GPT response"}), 400
    except Exception as e:
        print(f"Error in /create-course: {str(e)}")
        return jsonify({"error": str(e)}), 500

    # Generate course outline using the provided topic and expertise
    outline = generate_course_outline(topic, expertise)
    db_session = Session()

    # Get Postgres UID from Firebase UID
    postgres_user = get_user_by_firebase_uid(db_session, user["uid"])
    if not postgres_user:
        return jsonify({"error": "User not found"}), 404

    # Create the new course record
    new_course = create_course(
        db=db_session,
        user_id=str(postgres_user.id),
        topic=topic,
        expertise=expertise,
        content_outline=outline, # Course outline (JSON)
        pk=None,
        index=None,
        file_id=None
    )
    db_session.close()

    # Return the new course ID to the client
    return jsonify({"message": "Course created successfully", "courseId": str(new_course.id)}), 200

```

# 03

## Challenges

# Team Challenges

19

- Build times vary.
  - Moving backend out of Nextjs.
  - Shifting directives.
  - Fast-moving work environment.
  - Moving from frontend to backend.
  - Working with AI.

```
jiangruo@JRC-OptiPlex-5040:~/Desktop/MERN&K8 ~/OneDrive/Documents/Senior Design/LINK X (ryan)
$ bash run-backend.sh
Step 1: Building Backend...
[1] Building 2476.89 (16/17)
--> [internal] load build definition from Dockerfile
--> transferring dockerfile: 1.5kB
--> [internal] load metadata for docker.io/library/python:3.10-slim
--> [internal] load .dockerignore
--> transferring context: 2B
--> [1/12] FROM docker.io/library/python:3.10-slim@sha256:65c083635408a1ba2c12b5d083a022f4aeaf77497f4f0f7f0cbf8ce0931a96
--> [2/12] RUN pip install /library/python:3.10-slim@sha256:65c083635408a1ba2c12b5d083a022f4aeaf77497f4f0f7f0cbf8ce0931a96
--> [3/12] [internal] build context: 13.41MB
--> transferring context: 13.41MB
--> CACHED [ 2/12 ] WORKDIR /app
--> CACHED [ 3/12 ] COPY src/requirements.txt .
--> CACHED [ 4/12 ] RUN pip install --upgrade pip
--> [ 5/12 ] RUN pip install --no-cache-dir -r requirements.txt
--> [ 6/12 ] COPY src/.env .
--> [ 7/12 ] COPY src/firebasekey.json .
--> [ 8/12 ] COPY src/index.faiss ./app/index.faiss
--> [ 9/12 ] COPY src/index.pkl ./app/index.pkl
--> [10/12 ] COPY src ./app/src
--> [11/12 ] RUN pip install -U Flask flask-cors types-flask-cors
--> [12/12 ] RUN pip install faiss-cpu
--> exporting to image
--> exporting to image
--> exporting layers

1 warning found (use docker --debug to expand).

docker@desktop-linux:~
```

04

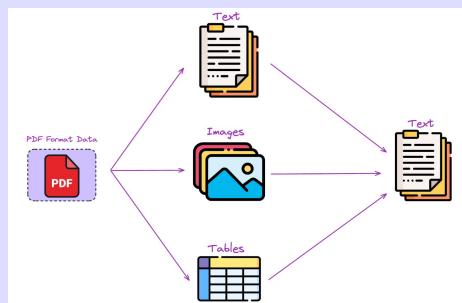
# Remaining Work Timeline

# Sprint Plan

Week Starting	Sprint Plan	
2/2/2025	Onboarding and understanding code base	
2/9/2025	Onboarding and understanding code base	
2/16/2025	Account creation and Authentication	
2/23/2025	Persona Creation and Database	
3/2/2025	Landing Page and Dashboard	
3/9/2025	Topics Page and Study Page	
3/16/2025	Spring Break	
3/23/2025	Testing Personalization & Backend	UI & Onboarding Complete
3/30/2025	Testing Personalization & Backend	
4/6/2025	Make Database Backend to save personalization progress	
4/13/2025	Testing RAG for Uploading pdfs	Have Working Personalization for any topic
4/20/2025	Certificate Report for completion	
4/27/2025	Deployment	
5/4/2025	Extra ~ Analytics for Dashboard (user progress)	
5/11/2025	Finals Week	

# Future Plans: Learn-X

- **Integration With Canvas:** Pulling course content from canvas.
- **Analytics:** Allow for relationship between student and instructor. Course instructors can view students' progress.
- **Expanding Study Page Features:** Implement RAG and pdf content scraping. Connect chatbot to content.
- **Seamless Integration:** Ensure all pages work cohesively with existing components for a smooth workflow.



# 05

Demo

23

# 06

Demo

24

# THANK YOU



25

# AI-Driven Personalized Learning Platform

Adam Hisel, Carter Parks, Morgan Prieskorn, Zac Mueterthies, Ryan Johnson

SD-14

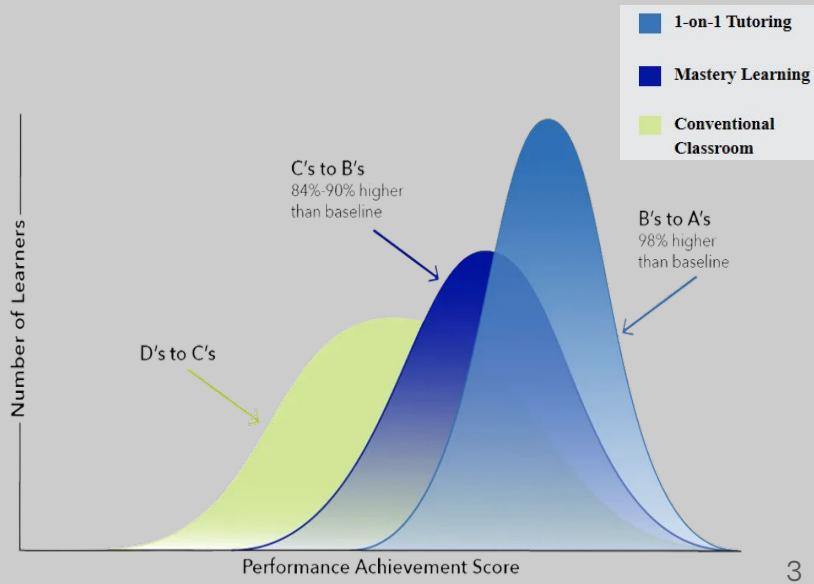
1



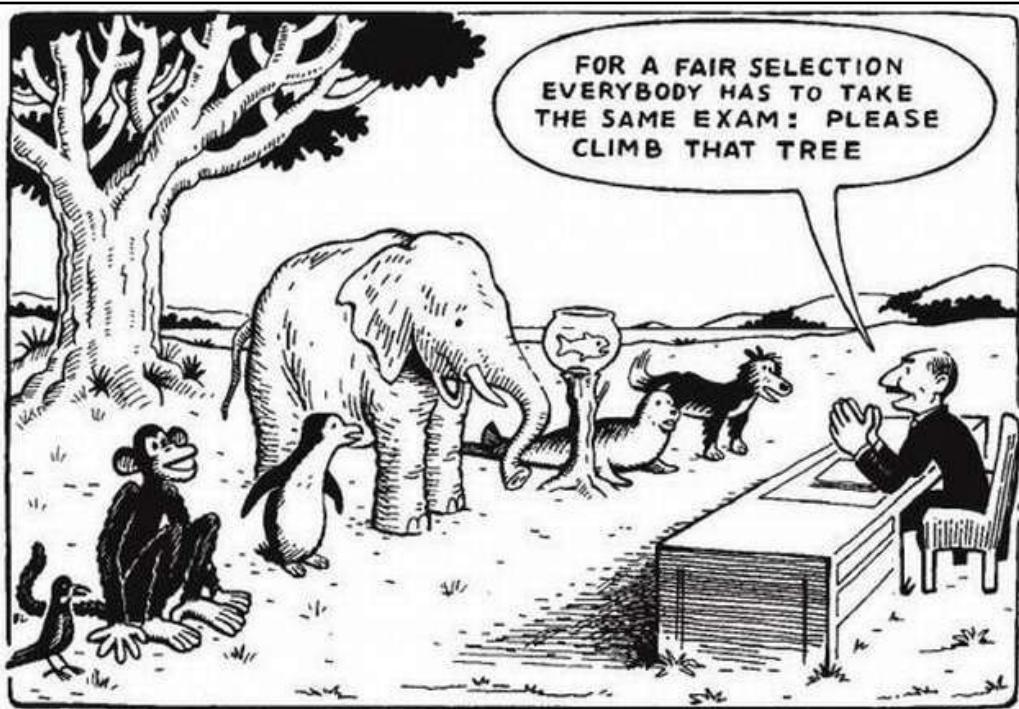
2

# Problem

- Not enough personalized learning
- Not all students learn the same way

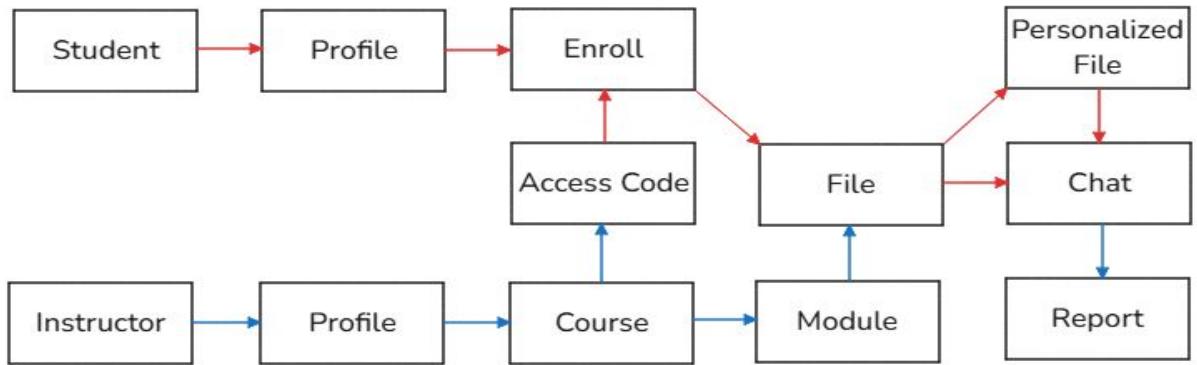


3



4

# User Flow



5

## Design Challenge 1: Users and User Types

- Rewrite backend to handle multiple user types
- Rewrite frontend to use new backend
- Working with multiple databases



6

# Design Challenge 2: OpenAI Prompting

- Reliability and Instruction Adherence
  - Consistent Outputs
  - Utilizing ALL content
  - Relevance & Accuracy
- Personalization
  - Adapting to Persona
- Specificity!

```
rows = db.execute(sql, [{"id": course_id, "query_vec": pvector_to_hex}], fetchall())
retrieved_chunks = [row[0] for row in rows if row[0]]
```

```
# 5. Build messages for OpenAI
messages = [
    {
        "role": "system",
        "content": [
            "You are a helpful and knowledgeable AI tutor assisting a student.",
            "Please use the student's background and interests to personalize each explanation and response.",
            "If course content is relevant to the user's message, you must use it to answer.",
            "If course content is not relevant, do not fabricate an answer. Instead, respond with \"I'm here to help with this course, but that question isn't related to the material we've covered.\"", 
            "I'm here to help with this course, but that question isn't related to the material we've covered."
        ]
    }
]
```

```
if retrieved_chunks:
    context_string = "\n".join([
        f"({i+1}) {chunk}" for i, chunk in enumerate(retrieved_chunks)
    ])
    material_prompt = {
        "role": "user",
        "content": [
            "The following concepts are from course materials. You must use them to answer the student's question if relevant:",
            f"({context_string})"
        ]
    }
    messages.append(material_prompt)
```

7

# Design Challenge 3: RAG Implementation

- Integrate RAG w/ Backend
  - Separately Running Container
- FAISS vs PGVector

## Definition of artificial intelligence (AI)

Hey Adam! Ready to flex those mental muscles and dive into the all-star game of Artificial Intelligence? Let's break it down like a tactical playbook!

Artificial Intelligence, often abbreviated as AI, is like the MVP of the tech world, and it's revolutionizing how we interact with technology. In essence, AI refers to computer systems or machines that mimic human intelligence processes. These processes include learning (acquiring information and rules for using it), reasoning (using rules to reach approximate or definite conclusions), and self-correction.

1. \*\*Learning\*\* Just like a basketball player fine-tunes their jump shot through practice, AI systems improve through experience. This is where machine learning (ML) comes into play, which involves algorithms that allow systems to learn from data. The system receives inputs, like a coach's feedback, and adjusts its actions to improve results.

2. \*\*Reasoning\*\* Think of this as the playbook of strategies that a quarterback memorizes for decision-making on the field. Similarly, AI systems apply rules and logical reasoning to process information and make decisions.

8

# Design Challenge 4: Smart Chatbot

- Prompting OpenAI
- Relevant Course Material
  - PGVector
- User Personalization
  - Proper Personalization
- Reliability

can you tell me how lizards interact with one another?

Lizards interact with one another primarily through visual displays, which are often related to territorial defense and mating behaviors. Here's a detailed look at how these interactions typically occur: 1. \*\*Territorial Defense\*\*: Many lizards are territorial and will defend their area from intruders of the same or closely related species. This defense doesn't always involve physical combat. Instead, lizards often use visual signals to assert dominance and ward off rivals. For example, they might erect crests along their back and neck or puff up their bodies to appear larger. 2. \*\*Visual Displays\*\*: Lizards use a variety of visual cues to communicate. They may display bright colors by extending a throat fan or exposing a colored patch of skin. These displays are often accompanied by stereotyped movements such as push-ups, head bobbing, and tail waving. These behaviors are not only for defense but also play a crucial role in mating rituals. 3. \*\*Mating Rituals\*\*: Male lizards use visual displays to attract females. For example, the green anole lizard inflates a rust-colored throat sack to win over a female, sometimes maintaining this display for hours. Males may also change colors to signal their presence and readiness to mate, with some species exhibiting dramatic color changes during the breeding season. 4. \*\*Aggressive Interactions\*\*: In some cases, male lizards may engage in physical fights to establish dominance or secure mating rights. These fights usually end when the weaker male retreats. 5. \*\*

can you tell me about how to drive a car

I'm here to help with this course, but that question isn't related to the material we've covered.

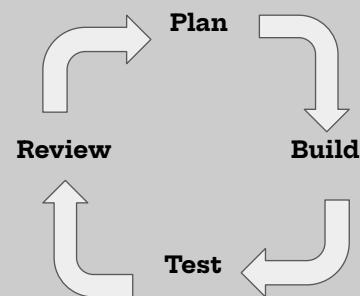
9

## Practices

- One week sprints
- Weekly client meetings

### Benefits:

- Immediate feedback
- Continuous alignment with expectations
- New tasks weekly



10

# Sprint Plan

Week Starting	Sprint Plan	
2/2/2025	Onboarding and understanding code base	
2/9/2025	Onboarding and understanding code base	
2/16/2025	Account creation and Authentication	
2/23/2025	Persona Creation and Database	
3/2/2025	Landing Page and Dashboard	
3/9/2025	Topics Page and Study Page	
3/16/2025	Spring Break	
3/23/2025	Testing Personalization & Backend	
3/30/2025	Testing Personalization & Backend	
4/6/2025	Make Database Backend to save personalization progress	
4/13/2025	Testing RAG for Uploading pdfs	
4/20/2025	Certificate Report for completion	
4/27/2025	Deployment	
5/4/2025	Extra ~ Analytics for Dashboard (user progress)	
5/11/2025	Finals Week	
		UI & Onboarding Complete
		Have Working Personalization for any topic

11

# Tools

## Frontend Tools:

- Next.js
- V0
- Loveable

## Database Tools:

- PostgreSQL
- pgvector

## Backend Tools:

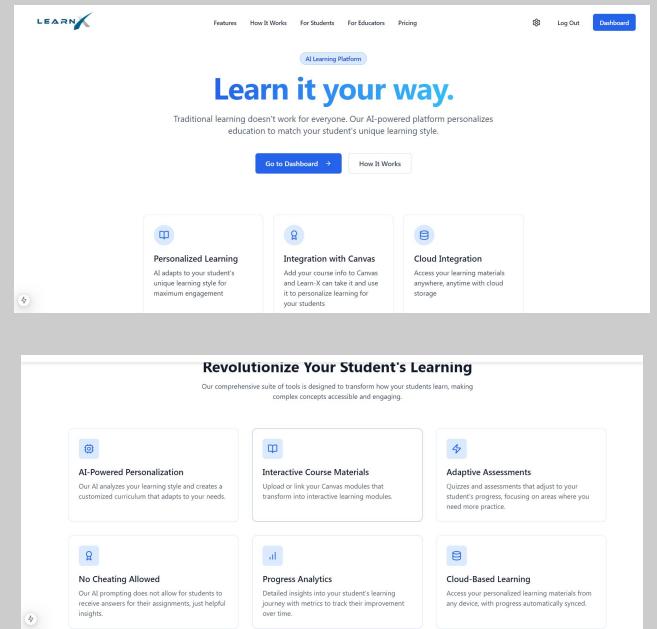
- Docker
- FAISS + RAG
- Firebase
- OpenAI
- Flask



12

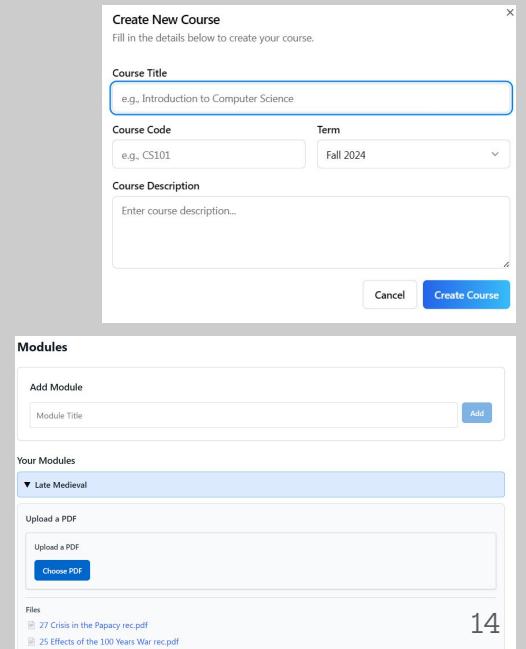
# Landing Page

- Draw potential clients in
- Showcase strengths of project
- Highlight features



# Professor Functionality

- Create courses for students
- Course appears on dashboard
- Add modules and documents



## Personalization

- Students take an onboarding quiz
- Results used to personalize the modules and chat bot

Personalized Learning Setup

What should Learn-X call you?

What do you do?  
e.g., Student, Engineer

What traits should Learn-X have?  
e.g., witty, encouraging

Preferred Learning Style

Select a learning style

Depth of Explanation

Select depth

Topics of Interest

e.g., Investing, Finance

Interests, Preferences

e.g., Basketball, Video Games

15

## Settings

- Edit Onboarding
- Change email and password

← Dashboard **Settings**

Account      Onboarding

**Account Settings**  
Manage your account information and preferences.

Email  
 morgan2@gmail.com

Password  
 Change your password

**Save Changes**

16

# Student Functionality

- Join course with access code
- Course appears on dashboard
- Can view modules and files professors uploaded

The screenshot shows a student's dashboard. At the top, there is a modal window titled "Join with Access Code" with a text input field "Enter code..." and a blue "Submit" button. Below this, a course card for "HISTORY" (HIST212 • Spring 2025) is displayed with a "View Course" button. A section titled "Modules" shows "Your Modules" with a dropdown menu currently set to "Late Medieval". Under "Files", two documents are listed: "27 Crisis in the Papacy rec.pdf" and "25 Effects of the 100 Years War rec.pdf".

17

# Personalized Learn Page

- Adds personalization, taken from onboardings quiz, to files the professor uploads

The screenshot shows a personalized learn page. On the left, a sidebar lists topics under three main sections: "DEVELOPMENTS DURING THE WAR" (Changes in Warfare, Changes in Armies, Political Problems, The Birth of Nations), "THE AFTERMATH OF WAR IN FRANCE" (The Recovery of Royal Power, The Threat of Burgundy, Beginnings of French Absolutism), and "THE AFTERMATH OF WAR IN ENGLAND" (The War of the Roses, The Rise of the Tudors). On the right, the main content area is titled "Changes in Warfare". It features a heading "Ah, the Hundred Years War, where knights were slowly shown the door, and infantry and archers took the spotlight. Imagine a video game where you swap out your overpowered knight character for a bunch of archers and foot soldiers. After 1400, cannons crashed the party, making those once-impenetrable fortresses about as useful as a horror movie villain who can't run. Boom, there goes the neighborhood!" Below this, there is an AI Assistant window with a message: "why were cannons so effective" and a response: "Ah, the mighty cannon! Let's delve into why these booming behemoths were so effective on the battlefield, with a hint of evil flair, just for fun. Cannons were like the bosses of the battlefield, and here's why: 1. 'Devastating Firepower': Imagine you're in a medieval battle, all decked out in your shiny armor. Along comes a cannonball, and boom! Your shiny armor is now a twisted mess. Cannons could easily demolish fortifications, crush infantry lines, and even make armored knights look like they were wearing paper suits. Their sheer destructive power was unmatched. 2. 'Long Range': Think of cannons as the long-range snipers of the past. They could hit targets from a distance that archers and musketeers could only dream of. This meant they could start wreaking havoc on the enemy". At the bottom, there is a "Ask a question..." input field and a small "Ask" icon.

18

# Prompting - File JSON Generation

- Generate JSON of entire content
- No Personalization
- Do not create facts, examples, or interpretations
- Preserve all information

```
**OUTPUT FORMAT**
Return a valid JSON matching this structure:

{
  "chapters": [
    {
      "chapterTitle": "string",
      "subsections": [
        {
          "title": "string",
          "fullText": "string"
        },
        ...
      ],
      ...
    }
  ]
}
```

```
**STRUCTURE**
For each chapter:
1. Provide a **concise chapter title** (3-7 words).
2. Include **2-4 comprehensive subsections**, each with:
   - A **short title** summarizing the subsection's focus.
   - A **fullText** explanation that presents the relevant information in **at least two sentences**.

**INSTRUCTIONS**
- Use language that is **clear, precise, and faithful** to the original material. Rephrase only to improve structure or flow.
- Use the **entire content** unless something is clearly redundant.
- You may **reorder or group** related points for clarity, but you must not omit any meaningful content.
- Do **not invent** any facts, examples, or interpretations. Work only with the content provided.
- If the original text contains specific names, terms, dates, steps, or examples, they **must be preserved** in the output.
```

19

# Prompting - Personalized File

- Take File JSON
- Modify fullText using Persona:
  - Tone
  - Depth
  - Learning Style
  - Interests

```
You are an AI assistant tasked with personalizing structured educational content.

You have been provided with a JSON object containing chapters and subsections: {JSON_response}
Each subsection includes:
- A **title** describing its focus
- A **fullText** field containing the original explanation

You will also receive:
- A description of the **user's persona**

**Your task is to revise ONLY the fullText fields to match the user's tone and background preferences.**
**INSTRUCTIONS**
1. You must **retain the original explanation and meaning** in every subsection.
2. Do **not change** any subsection titles, chapter titles, or the number of items.
3. You may personalize language, tone, depth, and examples based on the persona and expertise, but:
   - Do **not invent** new facts, terms, or interpretations
   - Do **not remove** information unless it is explicitly redundant
4. If a subsection includes an example, you may **adapt its context or framing** to be more relatable to the user, but:
   - The example must still teach the same lesson
   - The core logic or takeaway must be unchanged
```

20

## Prompting - Question Validation

- Chatbot
- PGVector
- Do not answer irrelevant questions
- Personalize answers

```
"You are a helpful and knowledgeable AI tutor assisting a student. "
"You must use the student's background and interests to personalize each explanation and response. "
"If course content is relevant to the user's message, you must use it to answer. "
"If the question is relevant to course material, but not specifically included, you can use your greater knowledge outside of course content. "
"If it is not relevant, do not fabricate an answer. Instead, respond with:\n\n"
"\n\"I'm here to help with this course, but that question isn't related to the material we've covered.\n\n"
"Avoid speculation or answering based on general knowledge if the topic isn't in the course context."
```

21

## Challenges Encountered

- Multiple database and backend redesigns
- New tools
  - Firebase, PostgreSQL, pgvector, FAISS + RAG
- AI Prompting
- Industry standard UI design



22

## Contributions

- Adam - Professor-side functionality, learn page design, student-side functionality, transcriber
- Ryan - Database(s) design, development and management, website backend
- Zac - FAISS, RAG, prompt engineering, course generation, content personalization
- Morgan - Onboarding page, landing page, courses page, student dashboard, settings page
- Carter - PGVector integration with chatbot, professor and student functionality, learn page