3002 Project Presentation

Magnificent 7

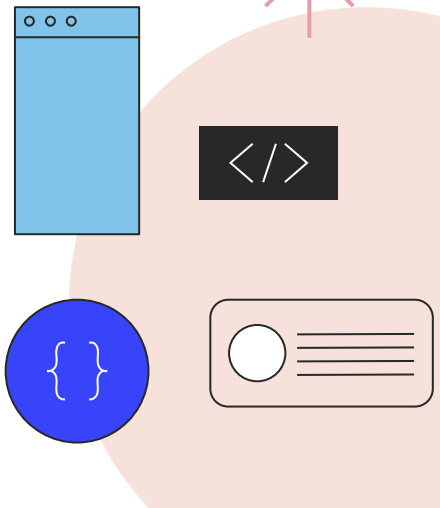


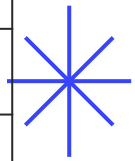
Table of contents

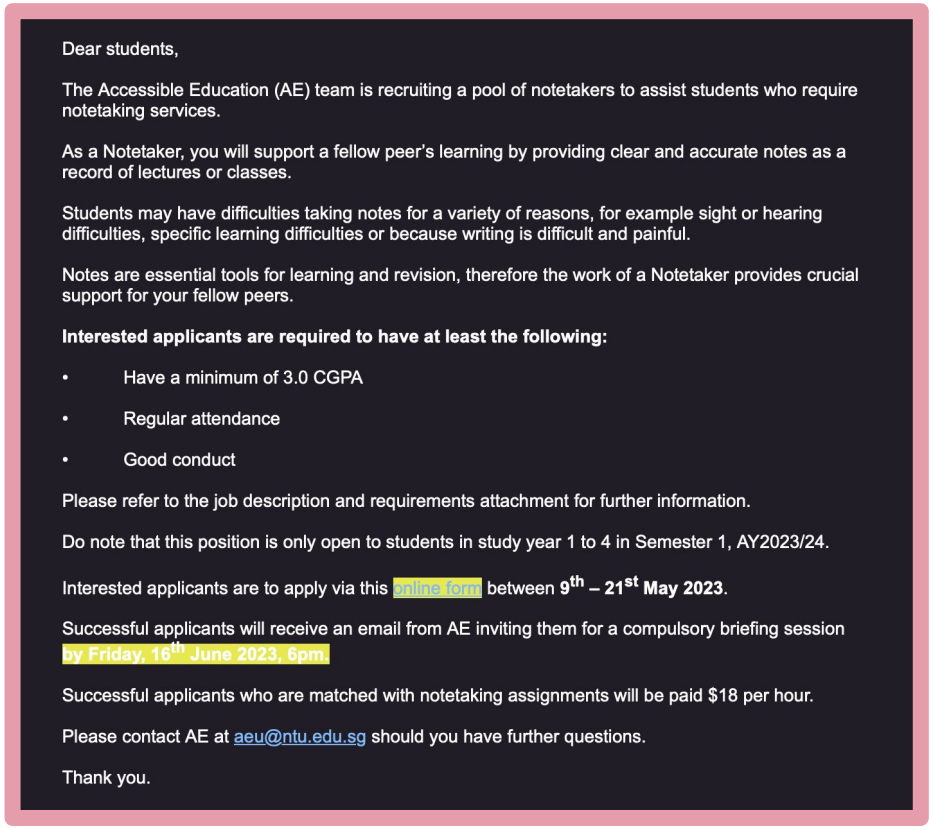
01.

Introduction

Team Members

Name	Matric Number	Role
Sherwin Samson	U2020911J	QA Engineer / Manager
Liu Changsong	U2022059F	Front-end Engineer
Yang Yida	U2022689D	Front-end Engineer
Jiang Jiaxi	U2022209K	Back-end Engineer
Sharan S/O Gopalakrishnan	U2022218F	Back-end Engineer
Zhang Mengao	U2023060L	Release Engineer/ Manager
Nathanael Axel Wibisono	U2021673B	Project Manager, Lead Developer





Dear students,

The Accessible Education (AE) team is recruiting a pool of notetakers to assist students who require notetaking services.

As a Notetaker, you will support a fellow peer's learning by providing clear and accurate notes as a record of lectures or classes.

Students may have difficulties taking notes for a variety of reasons, for example sight or hearing difficulties, specific learning difficulties or because writing is difficult and painful.

Notes are essential tools for learning and revision, therefore the work of a Notetaker provides crucial support for your fellow peers.

Interested applicants are required to have at least the following:

- Have a minimum of 3.0 CGPA
- Regular attendance
- Good conduct

Please refer to the job description and requirements attachment for further information.

Do note that this position is only open to students in study year 1 to 4 in Semester 1, AY2023/24.

Interested applicants are to apply via this online form between **9th – 21st May 2023**.

Successful applicants will receive an email from AE inviting them for a compulsory briefing session **by Friday, 16th June 2023, 6pm.**

Successful applicants who are matched with notetaking assignments will be paid $18 per hour.

Please contact AE at aeu@ntu.edu.sg should you have further questions.

Thank you.

NTU's Current Problem

- Manual note-taking in lectures
- Low-quality of notes due to lack of understanding of materials

Anti-study Study Tool

Automated Note-Taking & Summarization Solution

Magnificent 7

Star on GitHub

113,217 summaries generated so far

Anti Study Study Tool

GPT-3.5 GPT-4

1 Input information of the lecture

SC3040/CZ3002 | Select AY | Select semester

Select week | Select lecture | Search

Existing transcripts in the database

ps. click to check the transcript

Course code	AY	Sem	Week	Lec	Votes	Like it?
SC3040/CZ3002	AY23/24	1	1	1	4	
SC3040/CZ3002	AY23/24	1	13	1	1	
SC3040/CZ3002	AY23/24	1	12	1	0	

2 Not satisfied? Upload your own video/audio

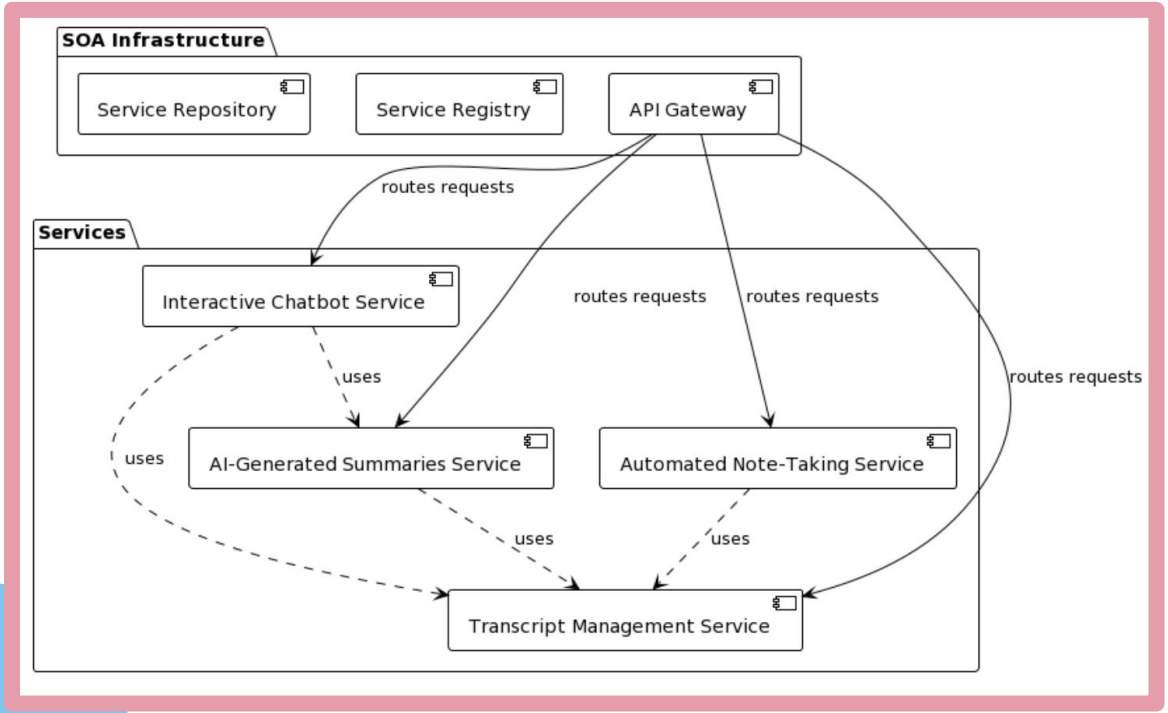
All Major Functionalities

Team	Functionalities
1 & 2	**Upload / download** saved transcripts in **Postgres DB**
1	Upload audio / video to generate **transcript** using **Whisper**
2	**Upvote** / **Downvote** uploaded transcripts
1	Transcript summarization based on **complexity** & **length** with **GPT-4**
1	Personalised **Chatbots** for clarifications
2	Design **REST API** request/response to access **postgreSQL DB**

*For complete functionalities, refer to documentation

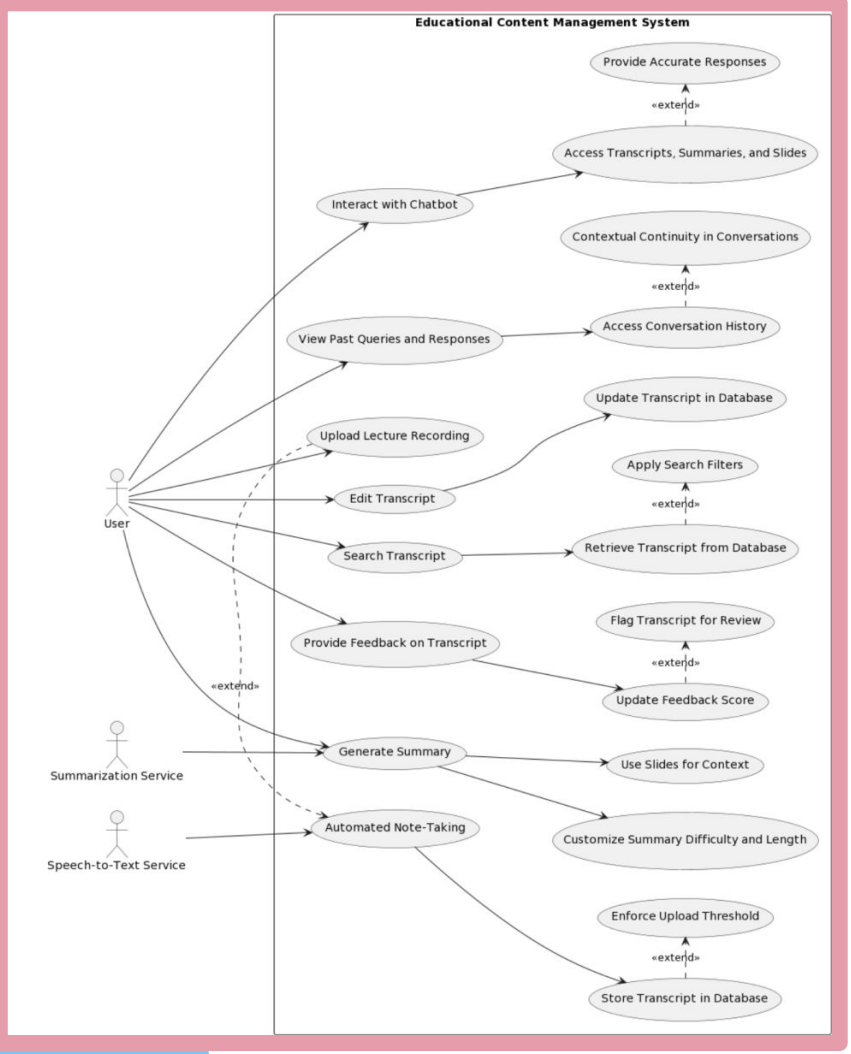
Design for Maintainability

SOA Infrastructure



- **Loosely Coupled**

- **Maximise interoperability**



Design for Maintainability

Use Case Diagram

Software Quality Assurance

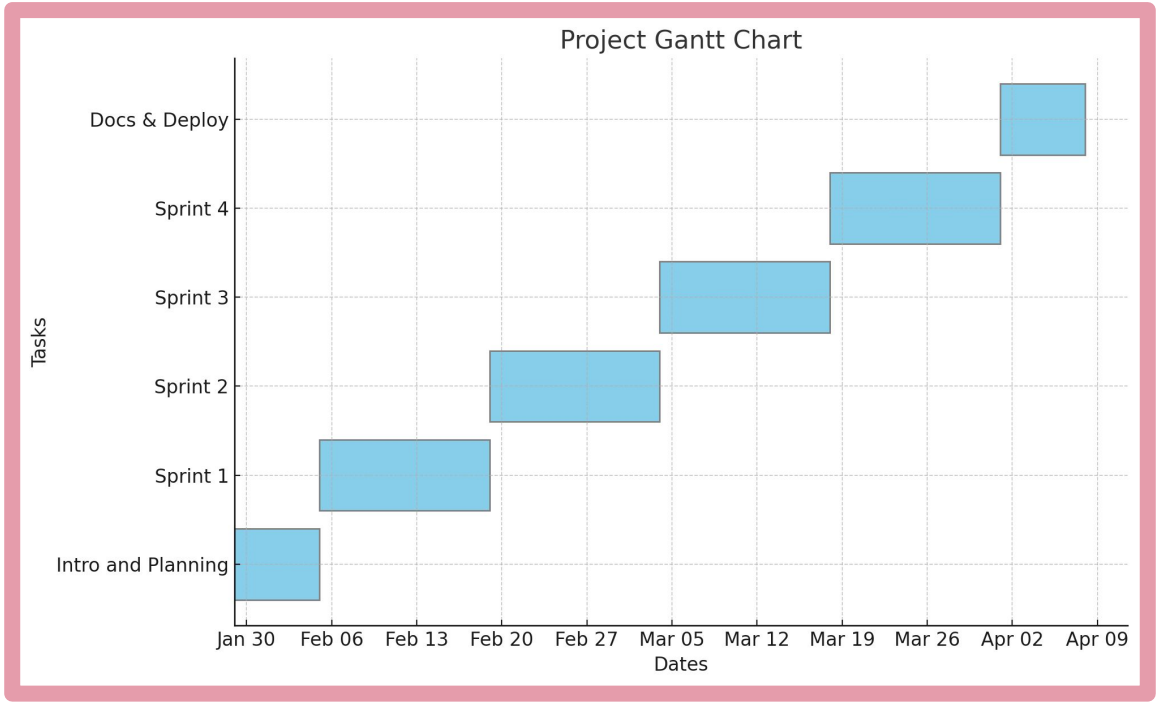
Features to be Tested

Feature	User Viewpoint Description	Risk Level
Automated Note-Taking	The system transcribes spoken words from lectures into written text, aiming for high accuracy to ensure reliable lecture notes.	M
AI-Generated Summaries	Users can obtain concise summaries of lecture notes tailored to their preferred complexity and length, enhancing study efficiency.	H
Interactive Chatbots	Chatbots provide personalized assistance, answering students' queries based on lecture content and summaries for enhanced comprehension.	M
Transcript and Summary Access	Students and faculty can easily access, search, and review transcripts and summaries, facilitating revision and knowledge retrieval.	L
Editable Transcripts	Users have the capability to edit and annotate transcripts for personalization and correction of any inaccuracies in the transcription.	M
Compatibility with Multiple Formats	The system supports a range of audio and video formats for uploading lecture content, ensuring wide accessibility.	L

- Unit Testing
(White Box Testing)

- Integration Testing
(Black Box Testing)

Project Management: Agile



Adjusted Function Points

Influence Factors	Score	Detail
Data Communications	3	The application supports a standard communication protocol for data transfer.
Distributed Functions	2	The project involves some level of distributed processing, particularly with cloud storage and processing.
Performance	4	High performance is crucial due to real-time processing of audio and large text data.
Heavily Used Configuration	2	The system will be regularly used by students and educators, with moderate load expected.
Transaction Rate	2	The system will handle a moderate number of transactions, particularly during lecture hours.
Online Data Entry	4	A significant portion of the system's functionality involves interactive data entry, such as uploads and edits.
End-User Efficiency	3	The system is designed with several features to enhance end-user efficiency, like summary generation and search functionalities.
Online Update	4	Real-time update and editing of transcripts and summaries are critical features.
Complex Processing	3	The system includes complex processing, such as speech-to-text conversion and generative summary.
Reusability	2	Components of the system are designed to be reusable, although this is not the primary focus.
Installation Ease	1	The system is web-based, requiring minimal installation effort from the end-user perspective.
Operational Ease	2	The system includes features for easy operation, but some manual configuration is necessary for setup.
Multiple Sites	1	The system is designed to be accessed from multiple locations, but does not require significant customization for each site.
Facilitate Change	3	The system is designed to be flexible, with features allowing for easy updates and changes based on user feedback.
Total score		**36**
Influence Multiplier		**= Total score × 0.01 + 0.65 = 36 × 0.01 + 0.65 = 1.01**
Adjusted FP		**= Unadjusted FP × Influence Multiplier = 58 × 1.01 = 58.58**

Efforts, Duration and Team Size Estimation

Estimated LOC: 58 FP × 29 LOC/FP = **1682 LOC**

Effort = Size / Production Rate = (1682 LOC) / (39 LOC/PD) = **43 PD**

Duration = 3 × (Effort)^(1/3) = 3 × (43)^(1/3) = **5.05 Days**

Team size = 43 PD / 5.05 D = 8.51 P ≈ **8 Persons**

Total person-hours (PH) = 43 PD × 8 hours = **344 PH**

Qualitative Risk Analysis

Component	Risk Description	Impact/Probability	Classification
Automated Note-Taking Process	API Dependency for Transcription might limit accuracy for diverse accents.	High/Medium	High (Prioritize)
	Processing time might exceed 10 minutes for a 1-hour lecture in some cases.	Medium/Medium	Moderate (Keep under review)
	Inaccurate transcription due to background noise or poor audio quality.	High/High	Critical (Act immediately)
Database Utilization for Transcript Management	Upvote/Downvote system may not accurately reflect transcript quality.	Medium/Low	Low (Monitor)
	Manual metadata entry could introduce human error and inefficiencies.	High/Medium	High (Prioritize)
	System performance issues with search functionality.	Medium/Medium	Moderate (Keep under review)
Summary Generation with Generative AI	Summarization effectiveness might vary with complex lecture content.	Medium/Medium	Moderate (Keep under review)
	Difficulty in integrating optional slides for added context.	Low/Medium	Low (Monitor)
	Inaccuracies in summaries due to generative AI limitations.	High/Medium	High (Prioritize)
Interactive Chatbots for Enhanced Learning	Chatbots might provide inaccurate or irrelevant information.	Medium/High	High (Prioritize)
	Contextual memory and conversation history may not work seamlessly.	Medium/Medium	Moderate (Keep under review)

Depends on:
Probability & Impact

Quantitative Risk Analysis

Risk Event Description	Estimated Impact on Project Activities	Numerical Rating
API Dependency for Transcription might limit accuracy for diverse accents.	Could lead to significant rework of transcripts, affecting project timelines.	8
Processing time might exceed 10 minutes for a 1-hour lecture.	Delays in transcript availability could reduce user satisfaction.	6
Inaccurate transcription due to background noise or poor audio quality.	High rework rates and decreased user trust in the system.	9
Upvote/Downvote system may not accurately reflect transcript quality.	Misleading quality indicators could result in poor content curation.	5
Manual metadata entry could introduce human error.	Inaccuracies in search functionality, affecting user experience.	7
System performance issues with search functionality.	Frustration and potential loss of users due to slow response times.	6
Summarization effectiveness might vary with complex lecture content.	Summaries may not meet user expectations, affecting study efficiency.	7
Difficulty in integrating optional slides for added context.	Reduced summary quality and relevance, impacting user satisfaction.	4
Inaccuracies in summaries due to generative AI limitations.	Could lead to misinformation and reduced credibility of the system.	8
Chatbots might provide inaccurate or irrelevant information.	Decreased user engagement and trust in the platform.	7
Contextual memory and conversation history may not work seamlessly.	Impacts user experience and perceived intelligence of the system.	5

Depends on:

RE = Probability * Impact

Risk Response Planning

Risk Event	Selected Response Strategy	Specific Action Plans
API Dependency for Transcription might limit accuracy for diverse accents.	Mitigate	Implement additional accent training modules for the speech-to-text engine. Conduct regular accuracy checks and updates.
Processing time might exceed 10 minutes for a 1-hour lecture.	Accept	Inform users of potential delays during peak hours. Explore options for asynchronous content delivery.
Inaccurate transcription due to background noise or poor audio quality.	Mitigate	Integrate advanced noise-cancellation technologies. Provide guidelines for optimal audio recording to users.
Manual metadata entry could introduce human error.	Mitigate	Automate metadata extraction where possible. Implement a review system for manual entries.
Upvote/Downvote system may not accurately reflect transcript quality.	Accept	Regularly review upvote/downvote trends and adjust algorithms as necessary. Encourage user engagement for more representative feedback.
System performance issues with search functionality.	Mitigate	Optimize database queries and server performance. Consider scaling up infrastructure to handle peak loads.
Summarization effectiveness might vary with complex lecture content.	Mitigate	Continuously train the AI on diverse content. Allow user feedback to inform adjustments to the summarization algorithms.
Difficulty in integrating optional slides for added context.	Accept	Provide clear guidelines for slide uploads. Work on incremental improvements to slide integration capabilities.
Chatbots might provide inaccurate or irrelevant information.	Mitigate	Enhance AI training datasets and improve natural language processing capabilities. Establish a feedback loop for user-reported inaccuracies.
Contextual memory and conversation history may not work seamlessly.	Mitigate	Implement more sophisticated memory management and user session tracking. Test and refine based on user interactions.

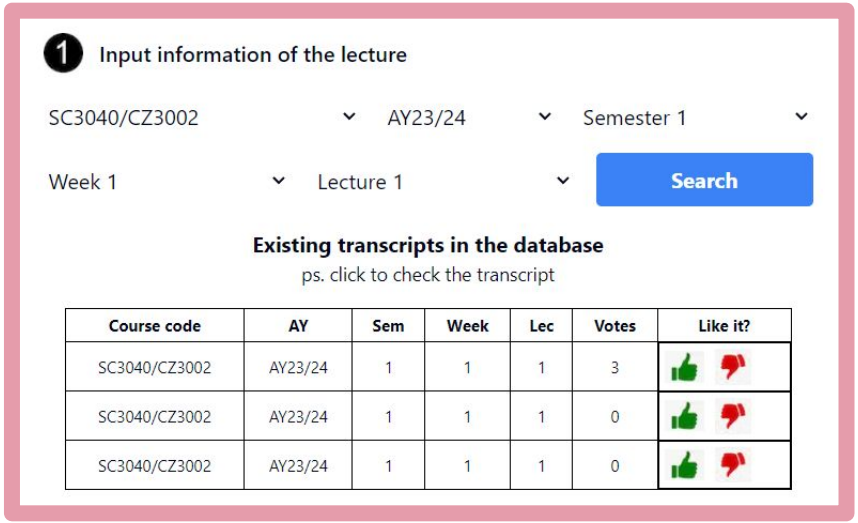
02.

Functional Requirement Team 1

Step 1: input information of the lecture

Step 1 allows the user to search for existing transcripts in database:

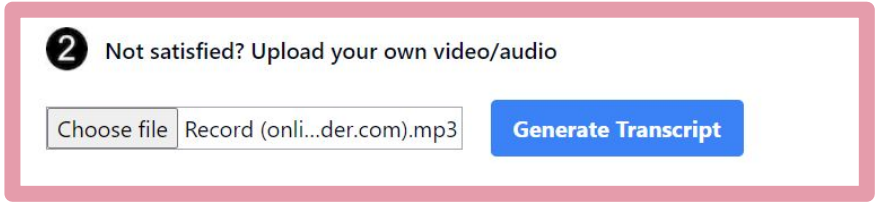
- **Select Information:** the user specify related information of the lecture through dropdown menu
- **Search from Database:** related lectures are retrieved from database when user presses the search button.
- **Display Lecture Information:** search results together with votes are display in the list. The user may click the row to display the transcript and upvote/downvote



Step 2: upload audio/video

In step 2, user upload their own audio/video if user is not satisfied with the search results:

- **Upload Video/audio:** The user uploads file from local
- **Generate Transcript:** Once uploading completes, the user presses generate transcript button. The backend then calls Whisper API to transcribe the video/audio.

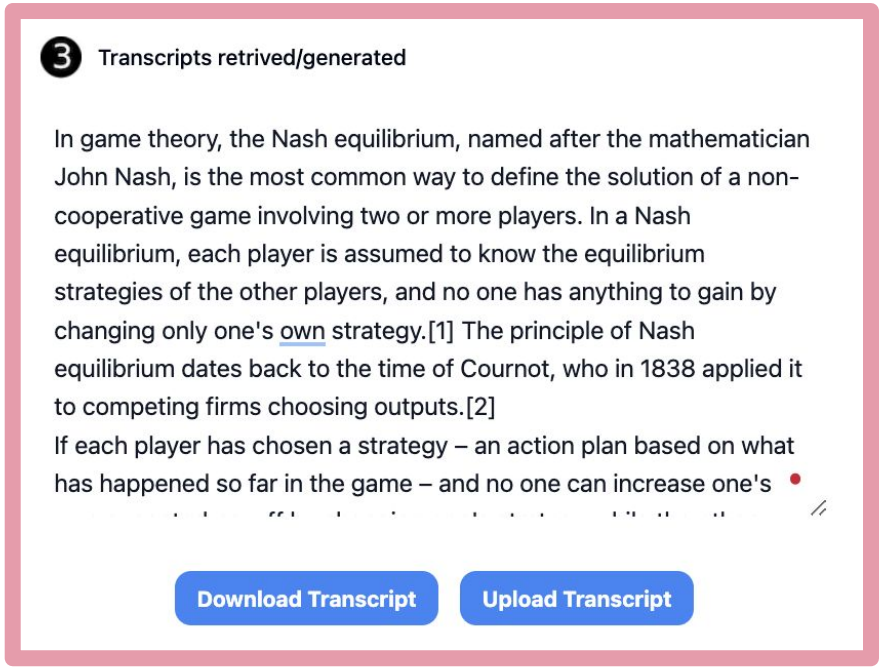



```
const transcribeAudio = async (audioFile: File) => {
  try {
    const transcription = await openai.audio.transcriptions.create({
      file: audioFile,
      model: "whisper-1",
    });
    console.log(transcription);
    setFinalTranscript(transcription.text);
  } catch (error) {
    console.error('Error transcribing audio:', error);
  }
};
```

Step 3: transcript retrieval / generation

Step 3 serves as a display area for both step 1 and 2:

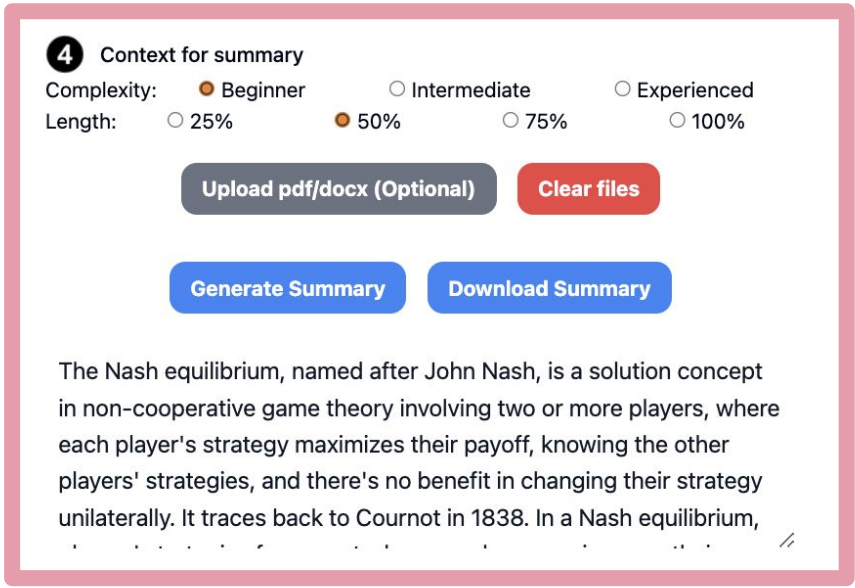
- **Connection with Step 1**: display saved transcripts retrieved from database
- **Connection with Step 2**: display the Whisper-generated transcripts from user's upload
- **Download Transcript:** save the retrieved/generated contents locally
- **Upload Transcript:** save the transcripts to database if user wants to share



Step 4: transcript summarization

Step 4 summarizes the transcripts in step 3:

- **Context for Summary**: customize the complexity and length for summarization
- **Upload Files**: offer an option for supplementary materials such as slides/tutorials; texts will be extracted from the files
- **Generate Summary:** send a query to gpt-4 api together with supplementary materials (if any) and display the results
- **Download Transcript:** save the generated summary locally if user wants to keep a copy



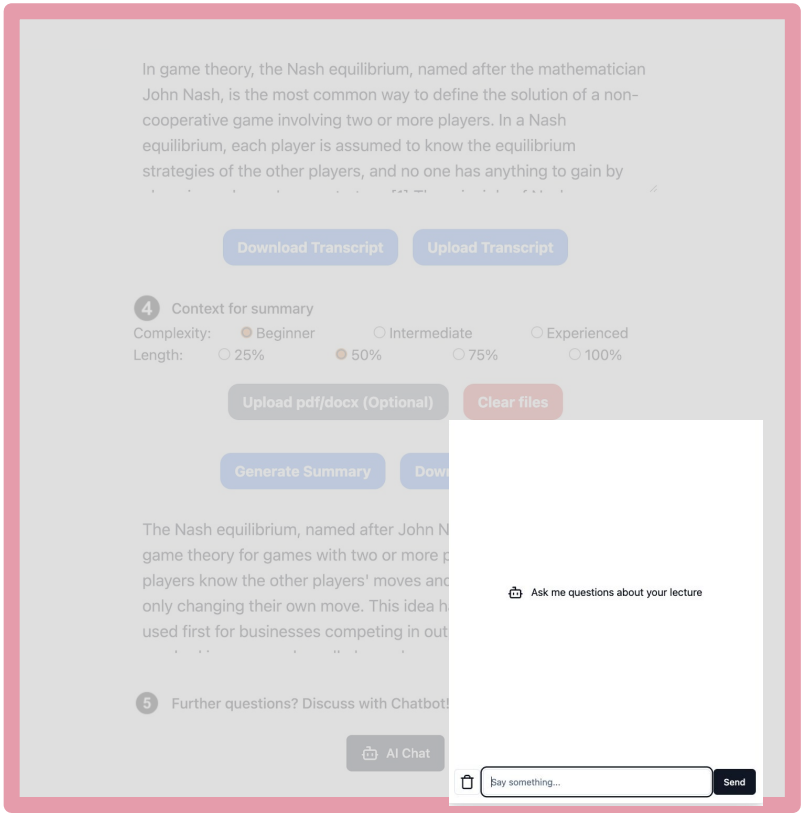
Step 5: AI chatbot

/components/AIChatBox.tsx

```tsx
export default function AIChatBox({ open, onClose }: AIChatBoxProps) {
  const {
    messages,
    input,
    handleInputChange,
    handleSubmit,
    setMessages,
    isLoading,
    error,
  } = useChat();
```

useChat () hook:

- Handles conversational UI
- Calls chat API endpoint



Step 5: AI chatbot

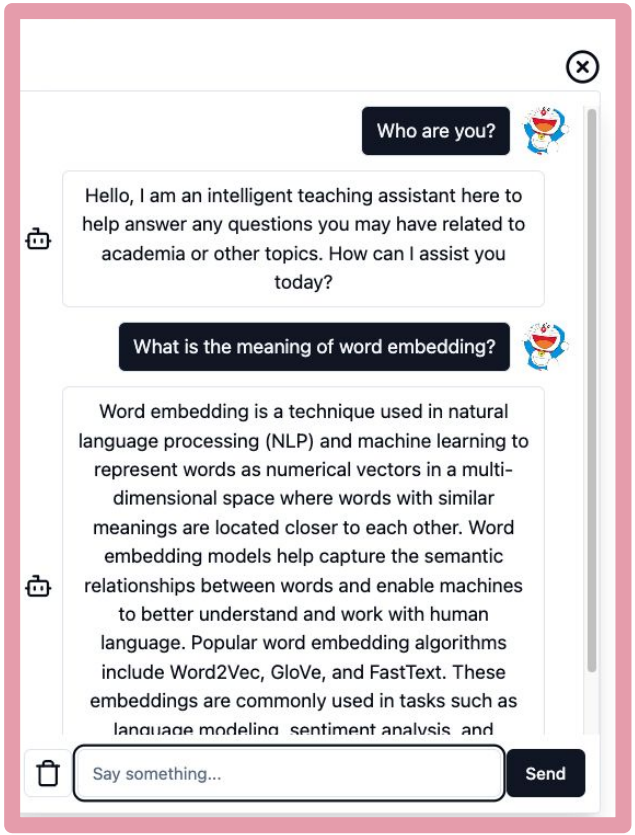
pages/api/chat/index.ts

```
const messagesTruncated = messages.slice(-6);

const systemMessage: ChatCompletionMessage = {
  role: 'assistant',
  content: 'You are an intelligent teaching assistant in university.' +
           'You may receive questions from students. Here is the transcript from lecture: ' + summary,
};

const response = await openai.chat.completions.create({
  model: 'gpt-3.5-turbo',
  stream: true,
  messages: [systemMessage, ...messagesTruncated],
});
```

messages

⇧

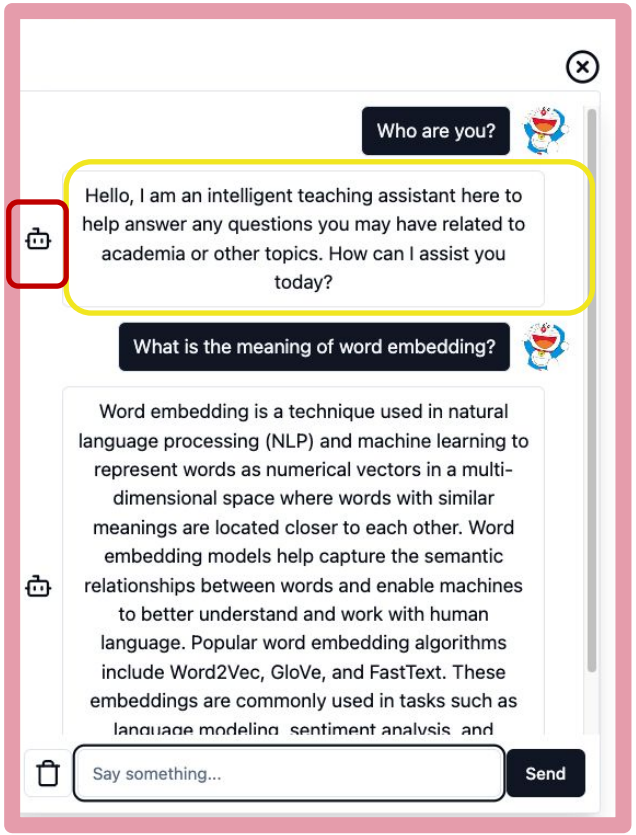
Initial prompt + The last 6 chat messages



Step 5: AI chatbot

/components/AIChatBox.tsx

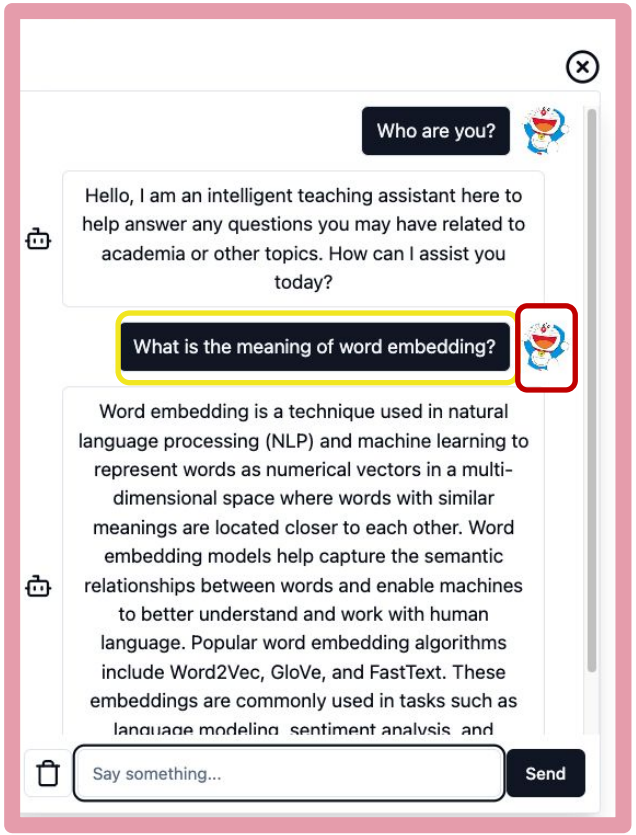
```tsx
function ChatMessage({
  message: { role, content },
}: {
  message: Pick<Message, "role" | "content">;
}) {
  const isAiMessage = role === "assistant";

  return (
    <div
      className={cn(
        "mb-3 flex items-center",
        isAiMessage ? "me-5 justify-start" : "ms-5 justify-end"
      )}
    >
      {isAiMessage && <Bot className="mr-2 shrink-0" />}
      <p
        className={cn(
          "whitespace-pre-line rounded-md border px-3 py-2",
          isAiMessage ? "bg-background" : "bg-primary text-primary-foreground"
        )}
      >
        {content}
      </p>
      {!isAiMessage && (
        <Image
          src="/avatar.jpeg"
          alt="User image"
          width={100}
          height={100}
          className="ml-2 h-10 w-10 rounded-full object-cover"
        />
      )}
    </div>
  );
}
```



Step 5: AI chatbot

/components/AIChatBox.tsx

```tsx
function ChatMessage({
  message: { role, content },
}: {
  message: Pick<Message, "role" | "content">;
}) {
  const isAiMessage = role === "assistant";

  return (
    <div
      className={cn(
        "mb-3 flex items-center",
        isAiMessage ? "me-5 justify-start" : "ms-5 justify-end"
      )}
    >
      {isAiMessage && <Bot className="mr-2 shrink-0" />}
      <p
        className={cn(
          "whitespace-pre-line rounded-md border px-3 py-2",
          isAiMessage ? "bg-background" : "bg-primary text-primary-foreground"
        )}
      >
        {content}
      </p>
      {!isAiMessage && (
        <Image
          src="/avatar.jpeg"
          alt="User image"
          width={100}
          height={100}
          className="ml-2 h-10 w-10 rounded-full object-cover"
        />
      )}
    </div>
  );
}
```



03.

Functional Requirement Team 2

Table “courses”

The table contains all the courses in SCSE.

Column	Datatype	Example
code	varchar	SC3040/CZ3002
title	varchar	“Advanced Software Engineering”

Table “histories”

Column	Datatype	Example
title	varchar	“Advanced Software Engineering”
academic_num	varchar	“2023-2024”
semester_num	int	1 or 2
week_num	int	1, 2, 3, …., 13
lec_num	int	1 or 2
transcript	text	
positive_num	int	
negative_num	int	

APIs

Get courses

GET


```
[
  {
    "id": 1,
    "code": "SC3040/CZ3002",
    "title": "Advanced Software Engineering"
  },
.....
]
```


Search histories

GET


```
"message": "Entry found",
"entry": {
  "id": 1,
  "title": "CZ3002",
  "week_num": 3,
  "lec_num": 1,
  "upload_time": "2024-02-18T09:20:28.658Z",
  "transcript": "This is an example transcript for CZ3002 week 3 lecture 1.",
  "academic_year": null,
  "semester_num": null,
  "positive_num": 0,
  "negative_num": 0,
  "is_showing": true
```


Store transcript

POST


```
"title": "CZ3002",
"week_num": 3,
"lec_num": 2,
"transcript": "This is an example transcript for CZ3002 week 3 lecture 2. This is generated from postman by Jiaxi.  this year, this semester. all the best for your test.",
"academic_year": "2023-2024",
"semester_num": 2
```


Upvote/downvote

PUT

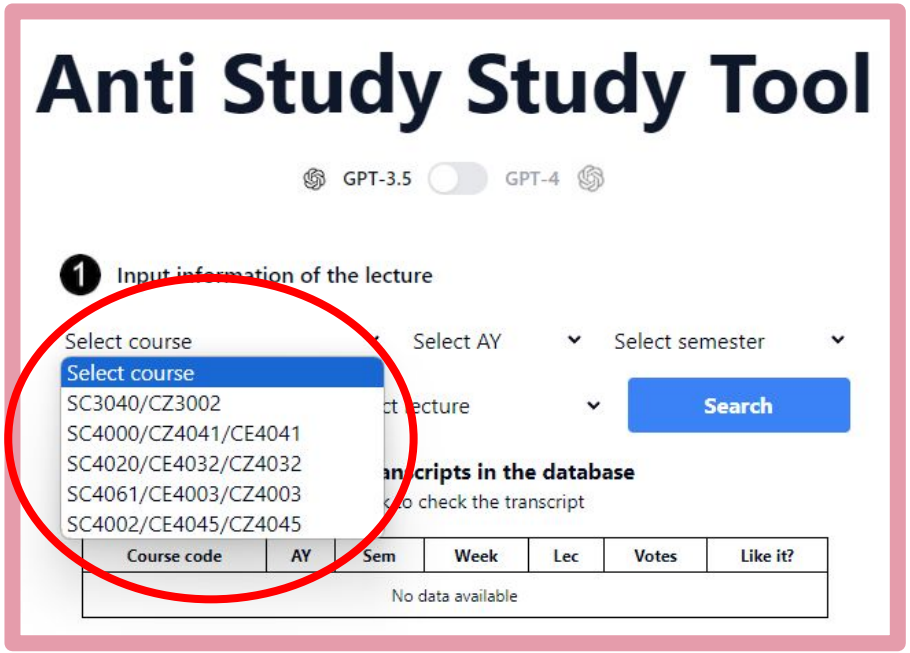

```
{
"id": "id"
"incrementPositive": +=1
}
```

```
{
"id": "id"
"incrementNegative": +=1
}
```

Get courses

Functionality to retrieve list of courses from table ‘courses’.

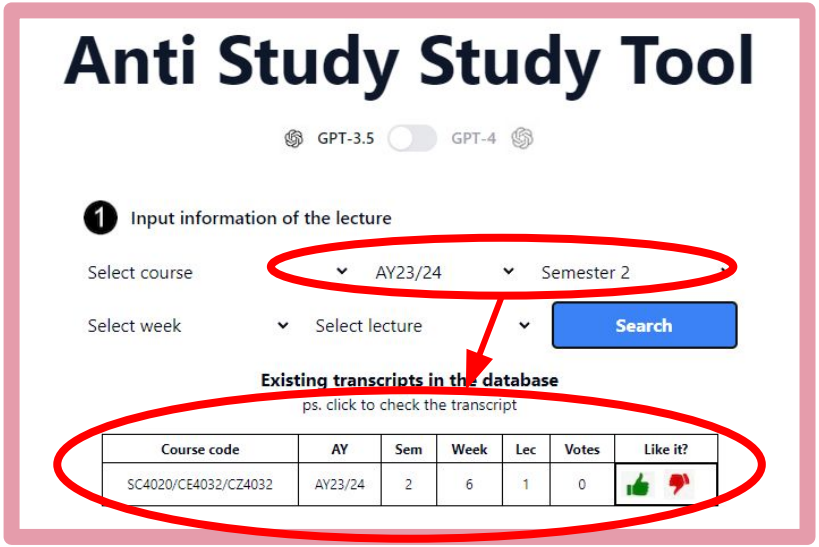
- **Parameters:** -
- **Checks:** Checks available course codes in ‘Courses’ table.
- **Use-case**:
  - Provide all courses to frontend for the dropdown list.
  - Allows students to store transcript according to the correct course code.



Search histories

Key functionality to search histories table.

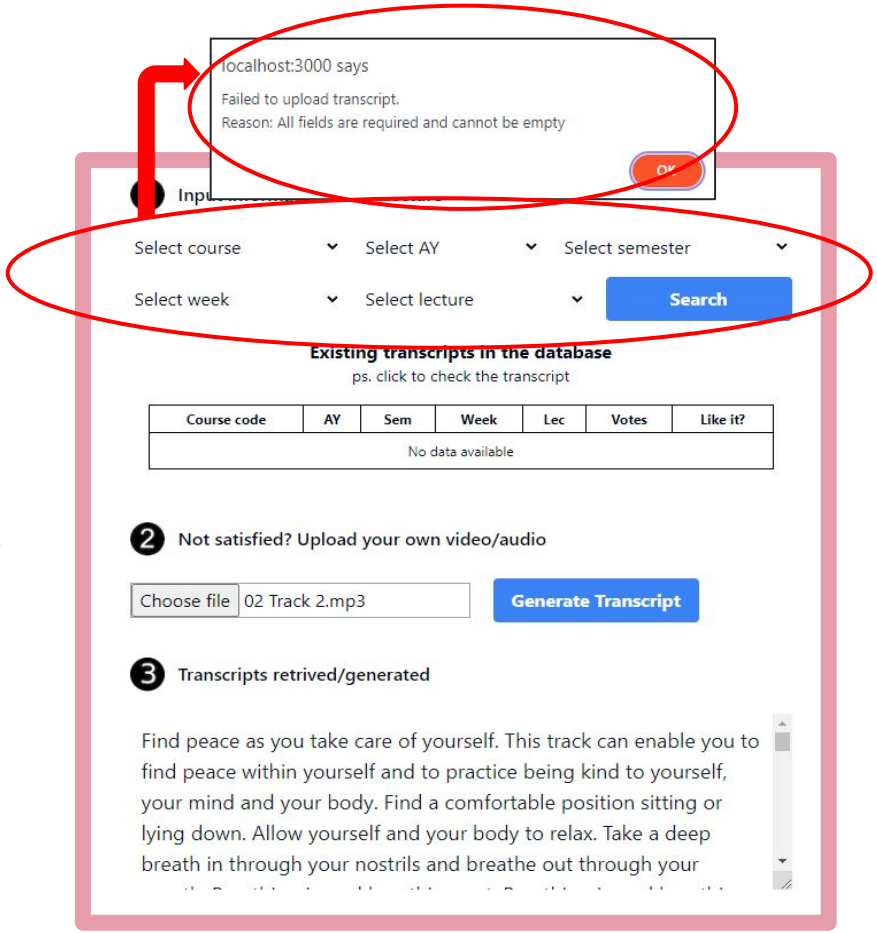
- **Parameters:** Course/AY/Sem no./Week no./Lecture no.
- **Checks:** Any of the given parameters.
- **Use-case**:
  - Search for transcripts in the table with specified parameters.
  - The more parameters set, the more specific the returned transcript.



Store transcripts

Functionality to store transcripts to Database.

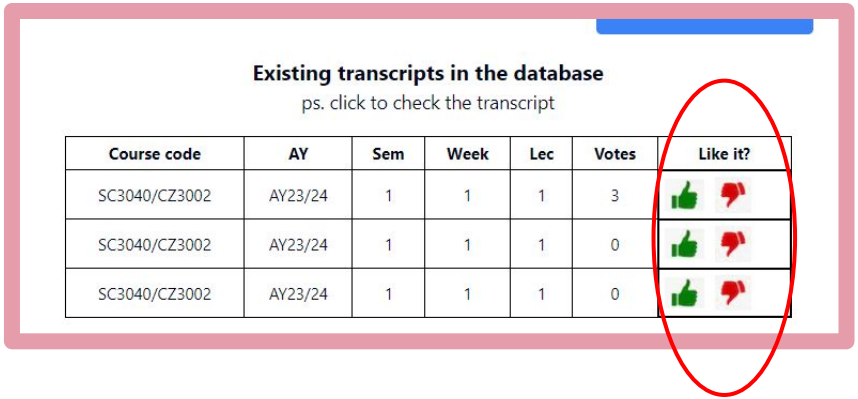
- **Parameters:**
  - course, AY, Sem, Week, lecture no.(first or second lecture of the week), transcript.
- **Checks:**
  - Need to provide all the parameters!
  - Maximum 3 slots per lecture, unless there are existing transcripts with >= 100 downvotes.
- **Use-case:**
  - Store user-provided transcript into the table “histories”.
  - Admin verifies uploaded lectures



Upvote/Downvote mechanism

Functionality to maintain relevance of Database.

- **Parameters:**
  - Id of transcript, *incrementPositive* OR *incrementNegative*
- **Checks:**
  - Need to provide both parameters to upvote or downvote!
- **Use-case:**
  - Flag outdated lectures
  - Upvote latest lectures.
  - Lectures with >100 downvotes are hidden



Existing transcripts in the database

ps. click to check the transcript

Course code	AY	Sem	Week	Lec	Votes	Like it?
SC3040/CZ3002	AY23/24	1	1	1	3	
SC3040/CZ3002	AY23/24	1	1	1	0	
SC3040/CZ3002	AY23/24	1	1	1	0	

04.

Demo