



[Hackathon Notion page](#)

The idea

An educational app that generates helpful materials for personal studying.

The pain

For the last hundred years, the formula for learning was 16 to 20 years of education followed by 40 years of work; the world is changing, and we all experienced constant upskilling and learning new things, which is a new paradigm. However, during the education period, the school provided us with various materials like exams and exercises to test our knowledge and make it stick.

The gap in the market

Currently, a lot of materials are widely available, but there are no tools that help you test, practice, or recall your knowledge.

Our solution

This is where our EduAction app comes into the picture. Users can upload any form of content, including YouTube lectures or part of the book, and the app, powered by Large Language Models, generates supportive materials to ensure the consumed knowledge sticks.

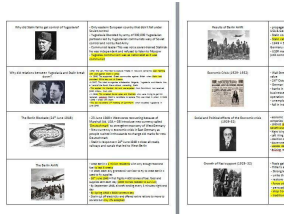
User story

History student upload a ww2 lecture link and gets a flash cards or a quiz to help him/her memories dates or leaders names, on this book/lecture.

API integration

We aim to use [stability api](#) to add visuals to the quiz and flash cards - we users memories visuals better with text than text only

Signal for market need



This [aweful] example of history flash cards to learn GCSE: they sell this s**t for £4! [\[here\]](#)

UX

<https://otter.ai/>

Users record their meetings and AI summaries the meetings. Seamless cx and solves pain points to the user and the business.

1st draft of the user journey [NDV]

- **Homepage** Establish value
- **Onboarding "Tour"** How this works
- **Step 1** Upload your lecture link!
- **Step 2** Validation link is legit
- **Step 3** Glimpse of "a-ha moment"
- **Step 4** Action items from the lecture
- **Step 5** Filter just what YOU NEED
- **Step 6** Save/share/"EduAction" again/other action

Team

- Nadav@link-big.com +447522123942 @Nadavoosh at telegram
- maciej@nannyml.com +44 7863 686881
- emil42larsen@gmail.com +45 29 89 11 39 @emil_at_telegram
- jeremycschan090204@gmail.com +44 7818 893157 @jeremycsc at telegram
- info@ebda3is.com +5511987915900 @BasemBrazil at telegram
- lucienlmp@hotmail.com +447522717166 @lucien_lin at telegram

Hunting for front ender/s

If we get 2 FE devs, we can split the work in Fe&Be working on the input bit ("add a link") and the other couple will work on the output bit (quiz/flash cards) - while NDV will design for both couples.

Stability AI - request API credits - **DONE** - [Signing up](#) for a Stability AI account

Back end & front end flow

1. The user uploads a YouTube link
2. YouTube API produce a transcript
3. EduAction platform ML scans the text
4. AI creates summary and tags content to dates/people/etc

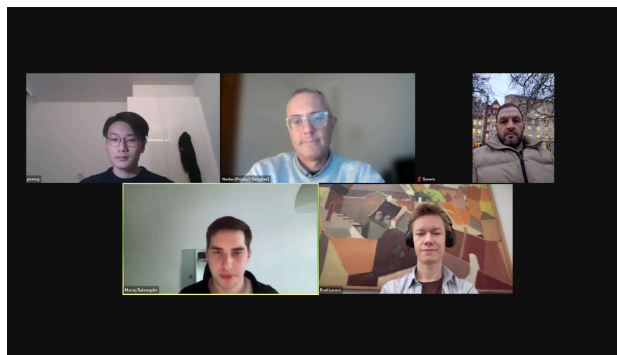
5. User gets a buffet of those tags to show online the AI has already “digested” the content
 6. AI asks the user which part they want to focus on: Summaries? Remembering dates/people?
 7. After the user picks the content they want to focus on, AI sees this as a prompt and create couple of “action items” like quizzes and flashcards
 8. EduAction presents the users a buffet of educational aids for them
 9. USer choose and have a play with them [quiz/flashcards/etc]
 10. USer can share with friends
-

Meeting Agenda

1. Intro
2. Craft idea
3. Craft tasks [domain. Server. screens.]
4. Action items
5. High 5 to us

Brainstorming ideas of functions in the app:

- Bring user into a state of flow(productivity)
 - Change the color of the screen, add a certain sound
- Addiction
 - Add daily streak
 - Memorization tasks
 - Leaderboard
 - Share flashcards with others
- Memorising schedule
 - Add a connection to the calendar and book time for studying (reminder) to build a habit

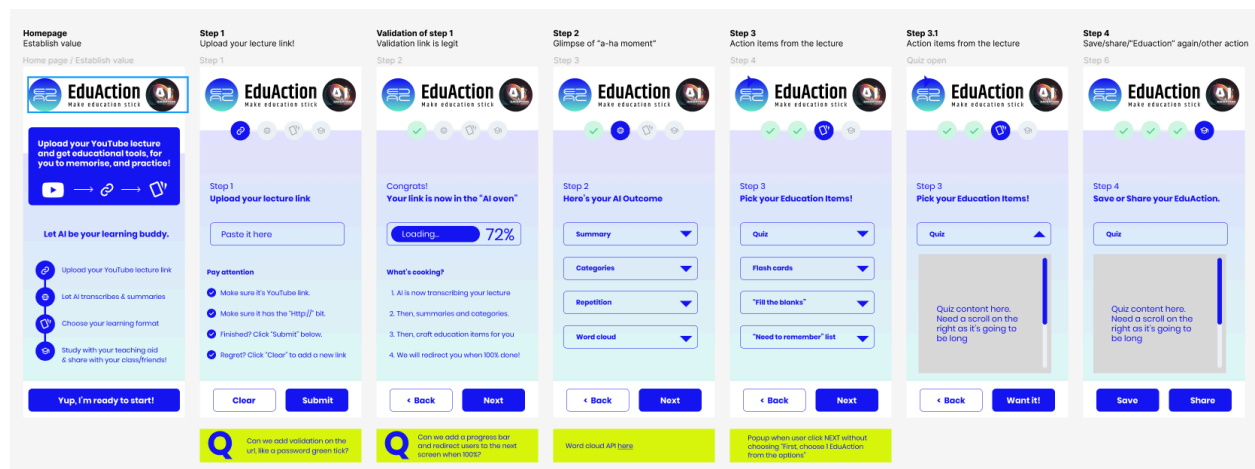


[Figma flow](#) [user journey]

Tasks:

Action items

Nadav: add content and theme to the figma screens.



Started to work on the screens content and made it 2 screens shorter 😊 NDV. [Link here](#)
[Mobile mockup](#)

Testing a new AI [mistral] chat app, and it crafted this flow for us:

a basic outline of how the back-end flow for such an application could look like:

1. **Video Submission:** The user submits a YouTube link through the front-end interface. This link is then sent to the back-end server.
2. **Video Retrieval:** The back-end server retrieves the video from the provided YouTube link using YouTube's API.
3. **Transcription:** The audio from the video is extracted and sent to a transcription service. This could be a third-party service like Google's Speech-to-Text API, or a custom-built solution. The transcription service returns a text transcript of the video's audio.
4. **Summarization:** The transcript is then sent to a summarization service. This service, which could be built using natural language processing (NLP) techniques, returns a summarized version of the transcript.
5. **Quiz Generation:** The summarized transcript is sent to a quiz generation service. This service, which could also be built using NLP techniques, generates a set of quiz questions and answers based on the content of the summary.
6. **Data Storage:** The original transcript, the summary, and the quiz are stored in a database. They are associated with the user who submitted the video and the video itself.

7. **Quiz Presentation:** When the user is ready to take the quiz, the back-end server retrieves the quiz from the database and sends it to the front-end interface.

Flash cards examples:

- 1 >>> <https://www.brainscape.com/>
2 >>> <https://quizlet.com/features/flashcards>

Prompt examples:

- 1 >>> <https://www.moreusefulthings.com/student-exercises>

Submission information

Deadline: Sunday 12:00pm, announcement at 15:00

Next steps: If we're in top 10 -> 16:00 pitch and demo live,

- 1 >>> Project Details
- name
 - Short description
 - Explanation of how sponsor tech was used
- 2 >>> Code base -> keep repo public for the jury from 12:00-15:00 Sunday with a detailed readme
- 3 >>> Video demo max 3 minutes -> walkthrough of the project, brief info about the codebase
- 4 >>> Additional materials

[Optional]

- Presentation/Deck
- Live website
- Technical Paper

Bounties:

- Overall Bounty for the best project
- Best AI use-case for Gaming, Social, or Entertainment
- Best use-case for Stability AI

More info: <https://encodeclub.notion.site/Bounties-0ec6308ddedc41329b77a09c0637276f>

Backend tasks(high-level):

- Figure out how to get the transcript from YouTube link - Maciek
- Pick the LLM model to use
- Upload the transcript to the LLM - Emil

- Test Stability AI API - Maciek
- Generate a Flashcard image from the prompt generated by LLM before
- Figure out what database to use for Flashcard and Quizzes storage - Emil
- Store the Flashcards and Quizzes in the database - Emil
- Find YT lectures to test on - Maciek
- Create a nice readme page - Maciek
- Optimizations, pa endpoint calls
- Add support for longer videos by chunking the input prompt
- REST API
- Experiment with different prompts to get the FlashCards and Quiz
- Application deployment on website e.g. Vercel
-

Extra nice-to-do things if time allows

- Use other models
 - Mistral Large model
 - Mixtral 8X7B
- Fine-tune LLM
- User feedback system
- Authentication (e.g. Auth0)

Resources:

https://www.reddit.com/r/Anki/comments/129khha/best_chatgpt_prompt_for_flashcards/
<https://www.reddit.com/r/Anki/comments/11306zf/comment/j8n46il/>
<https://super-memory.com/articles/20rules.htm>
<https://www.brainscape.com/flashcards/evolution-5705938/packs/8920128>
<https://www.datasciencetrivia.com/>

Crypto lecture >> maciej@nannyml.com

<https://www.youtube.com/watch?v=hEEWMiMuEF8>

Project details:

- Use-case
- Target audience
- Potential market impact

Prototype:

- Explain what it is able to do today

Code repo:

- Update readme