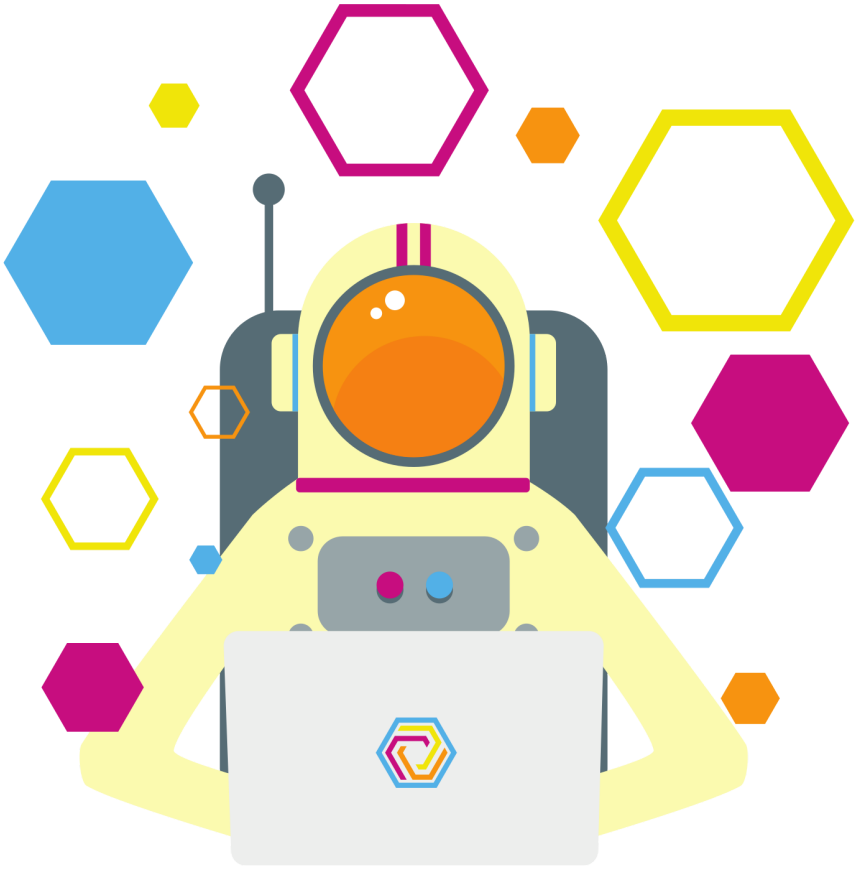
VIS con
SYMPOSIUM
HACKATHON



# HACKATHON

## 2023

# PROBLEM STATEMENTS

VIS - Verein der Informatik Studierenden an der ETH Zürich
CAB E 31, Universitätsstrasse 6, ETH Zentrum, CH-8092 Zürich

# Table of Contents

# ETHereal: Campus Ghost Hunt

**Stakeholder:** Robert Veres

## Overview

*Remember Pokemon GO? Well...*

Some days, not even the prospect of Mothy Roscoe unraveling the wonders of operating systems in between his coffee breaks is enough to lure you out of bed. But we both know you won't be as productive at home, even if you like to tell yourself that despite it never working. What if you had the chance to capture a hauntingly elusive Einstein Ghost on your way to HG F7? Or the Röntgen ghost in line at the Polymensa? Or the von Braun Ghost... maybe not that one.

Lo and behold, the Campus Ghost Hunt to your rescue when you need that extra push! Because nothing is spookier than ETH.

## Requirements

- **Ghost Spawning:** Develop a system to spawn virtual ghosts (e.g. of notable alumni) randomly around the ETH Zurich campus.
- **Ghost Hunting:** Develop a location-based mechanic to hunt ghosts.
- **Game Progression:** Implement a way for players to collect virtual items or earn rewards as they catch more ghosts. Ensure the game can save players' progress.

## Optional enhancements

- **Augmented Reality (AR):** Add AR features to enhance the gaming experience.
- **Social Features:** Include a leaderboard or basic social sharing options for players to compare their progress with friends.
- Make the hauntings time-sensitive.
- Team hunts: Enable team-based gameplay where groups can join forces to catch a particularly elusive ghost.
- Link the ghosts to their relevant locations. For instance, the Einstein ghost might be hanging around his locker more often than other locations.

# MeetMate: Rendez-Vous Planner

**Stakeholder:** Davud Evren

## Overview

Develop an application that simplifies the process of scheduling meetups for busy individuals with varying schedules. The goal is to enable each user to upload their calendar data and then have the app find the optimal meeting times.

## Requirements

- **Calendar Data Upload:** Create a feature that allows users to upload their calendar data directly from exported calendar files. Ensure compatibility with common calendar formats.
- **Optimal Meeting Time:** Implement a function to analyze the uploaded calendars and identify the best available times for meetups within a specified timeframe. Prioritize compatibility with all users' schedules.
- **Privacy:** Ensure that the application stores the bare minimum amount of data possible. Privacy and data security are paramount.

## Optional enhancements

- **User Time Preferences:** Include user-controlled parameters for time preferences, allowing users to specify preferred meeting times such as mornings, evenings, or weekends.
- **Granularity Control:** Offer an option for users to control the granularity of meeting times, enabling them to choose specific time intervals for scheduling.
- **Free-Time Calendar:** Create a “free-time calendar” that displays available slots based on the uploaded calendars (essentially, the intersection of all users’ free time slots), making it easier for users to identify mutual free time slots.
    - **Heat Map (Optional):** Visualize the free-time calendar with a heat map, where each time slot is colored in accordance with how many people are available during that time slot.

# NerdHerd: Study Group Finder

**Stakeholder:** Davud Evren

## Overview

Create a web application to help university students find and form study groups for their courses. The application should provide an intuitive, easy-to-use interface that minimizes potential social awkwardness and embarrassment, encouraging more collaborative learning.

## Requirements

- Secure student authentication using university credentials.
- Listing of all university courses to allow students to select the ones they're studying.
- Functionality for students to create/join study groups associated with specific courses.
- Ability for users to indicate their availability, preferred group size, and preferred study methods.
- A rudimentary matching system to suggest potential study groups based on course, availability, and preferences.
- Compliance with data privacy laws and university policies.

## Optional Enhancements

- A chat functionality for initial introductions and ongoing communication within the group.
- In-app scheduling tool for coordinating study sessions.
- A mechanism to provide feedback on group sessions to continuously improve the matching system.
- A 'Study Tips' section where students can share helpful resources or strategies.

# Welp: Real-Time Mensa Food Reviewer

**Stakeholder:** Federico Mantovani

## Overview

*My gosh, the lines for this mensa are long... Is the food even worth it?*

Create a real-time food review platform for the university's cafeteria (Mensa). With long lines and limited time, students need an efficient way to determine if the cafeteria food is worth the wait.

## Requirements

- **Real-time Food Reviews:** Develop a platform that allows students who have purchased food from the Mensa to submit quick and concise reviews about their dining experience. Reviews should be collected promptly after each meal service.
- **User-Friendly Interface:** Design an intuitive and user-friendly interface for both submitting and viewing food reviews. This should be accessible via a mobile app or a web platform.

## Optional Enhancements

- **Recommendation Engine:** If feasible, consider adding a recommendation engine that suggests dishes based on user preferences and popular reviews.
- **Notifications:** Notify users when highly-rated meals (or meals they've faved) are being served.

# TLDR: Lecture Summarizer

**Stakeholder:** Kwok Wai Lui

## Overview

*This lecture is so long... and I don't know what's actually important.*

Lectures and slides don't actually always cover the same topics, and it isn't always clear what is important and what isn't.

Develop an AI-based program that can automatically generate concise summaries of lengthy lectures, helping students identify the key topics and important information. This tool aims to bridge the gap between lecture content and slides, providing clarity on the lecture's core subjects.

## Requirements

- **Transcript Generation:** Most lectures at ETH Zurich don't provide transcripts. Use existing speech recognition software to generate transcripts for lecture recordings.
- **Transcript Analysis:** Create a system capable of processing lecture transcripts, extracting the most significant topics and discussions covered during the lecture.
- **Topic Triage:** Implement a mechanism to identify and prioritize the main topics discussed within the lecture, highlighting their relevance.

## Optional Enhancements

- **Customization:** Allow users to customize how summaries are generated, e.g. controlling the level of detail.
- **LaTeX:** Support LaTeX expressions in generated summaries.
- **GUI:** Wrap the application in an easy-to-use GUI.

# ThesisMingle: Thesis Matching App

**Stakeholder:** Robert Veres

## Overview

Create a centralized platform that enables ETH supervisors to advertise available thesis topics to students. The platform should also provide features for students to apply for these topics directly through the system. As an optional enhancement, the platform can also crawl ETH-associated websites for additional thesis opportunities.

## Requirements

- **Supervisor Dashboard:** Develop a dashboard where ETH supervisors can post details about available thesis topics. Include fields such as title, description, department, and contact information.
- **Data Organization:** Organize the advertised thesis topics in a structured database, making it simple for students to search and explore the available opportunities.
- **User Interface:** Create an intuitive platform where students can search, filter, and explore thesis topics based on criteria like department, keyword, or supervisor.
- **Application Process:** Implement a feature that allows students to apply directly for thesis topics through the platform. Include the option to attach resumes, cover letters, and any other necessary documents.
- **Regular Updates:** Enable supervisors to update or remove their advertised thesis topics, ensuring that students have access to current information.

## Optional Enhancements

- **Web Crawling:** Develop a web crawling mechanism to periodically scan ETH-associated websites and gather information about thesis opportunities that are not advertised on the platform. Extract key details like title, description, department, supervisor, URL, and contact information.
- **Recommendation System:** Build a recommendation engine that suggests thesis topics to students based on their academic interests and previous selections.
- **Alerts and Notifications:** Provide an option for students to set up alerts or notifications based on specific criteria, such as new thesis opportunities in their department.
- **Feedback Mechanism:** Include a feature that allows students to rate or review the thesis topics they have pursued, providing valuable insights for future students.

# ETHoot!: Quiz Master

**Stakeholder:** Davud Evren

## Overview

Develop an interactive web application where students can collaboratively create and share quizzes and flashcards.

## Requirements

- **Course Registration:** Create a system where ETH Zurich students can register for the courses they are taking during a semester or academic term.
- **Quiz Creation:** Enable students to create quizzes for their registered courses. They should be able to add questions and provide answer options, and make the quiz available to other students.
- **Quizzing:** Enable students to take each other's quizzes.

## Optional Enhancements

- **Rating & Commenting:** Allow students to rate and comment on each other's quizzes, possibly even on individual questions within a quiz.
- **Multiplayer Mode:** Implement local and/or online multiplayer modes, allowing students to compete or collaborate in quizzes with their peers.
- **Rewards & Points:** Reward students for their submissions, especially if others found them helpful, and also for answering quiz questions correctly.

# LectureLinker: Lecture notes update service

**Stakeholder:** Federico Mantovani

## Overview

Everybody solves the problem of lecture notes differently. Some people manually download all lecture notes for all their lectures, some people write scripts to crawl the website's HTML and some on the other hand buy domains such as wasisteinevorlesungswebsite.ch and don't download a single bit.

None of these methods however solve the problem of centralizing the available information. In an ideal world, all lecture notes would magically appear in a centralized place, which is what this app tries to achieve.

## Requirements

- Display list of lectures to sync lecture notes and exercises from
- Allow subscription to selected lectures and automatic synchronization into the user's polybox
- Do NOT save login credentials in any persistent data backend
- Proof of concept for Moodle integration where users only select a lecture based on a name or an (easy to find out) ID

## Optional enhancements

- Detection of new versions and locally modified versions and sane conflict handling
- Notify all subscribed people of a lecture if new content was updated (e.g. via notification to a VIS chat channel or via daily mail) as soon as this is recognized by a service
- Integration of all other commonly used lecture websites
- Anything else you can come up with

## Notes

- The polybox has a WebDAV interface for storing and fetching files programmatically.

# Güntherslist: Marketplace Hub

**Stakeholder:** Federico Mantovani

## Overview

Desperately need a new couch for those long study sessions? On the hunt for a van for that one-day escape from the mysteries of quantum mechanics to the deterministic lands of IKEA? Fancy an electric saw for some... totally educational purposes, we swear? Hey, we get it—life at ETH isn't all algorithms and mathematics and crying. However, going to random strangers for your eclectic needs is like asking Schrödinger's cat to babysit—unpredictable at best. But fear not! ETH students are a trusty bunch, offering a cornucopia of goods, services, and yes, even electric saws. Welcome to Günther's List, where one student's textbook is another student's doorstop. Let the bartering begin!

## Requirements

- **Login:** Implement a basic login system that allows registered users to access the platform securely.
- **Item Listing:** Create a feature that allows users to post items or services for sale, including detailed descriptions, photos, categories, and pricing.
- **Search and Filters:** Develop an intuitive search and filtering system so users can easily browse items based on various criteria such as category, price, or location.
- **Messaging System:** Enable secure and private messaging between buyers and sellers to discuss transaction details.

## Optional Enhancements

- **SSO:** Secure student authentication using university credentials.
- **Ratings and Reviews:** Implement a ratings and reviews system for both buyers and sellers, which could contribute to a safer and more reliable marketplace.
- **Alerts and Notifications:** Provide an option for users to set up alerts or notifications for items they are interested in, based on specific criteria like category or price range.
- **Transaction Monitoring:** Include a transaction status system that tracks the stages of a deal (e.g., listed, in discussion, sold) and allows users to mark transactions as completed.

# StudyQuest: Study Planner

**Stakeholder:** Kwok Wai Lui

## Overview

We all know that feeling, at the beginning of the “lecture-free time”: The exams are coming closer and we should begin to study. But how? There are a lot of topics, maybe you even missed some during the semester, and every lecture has different resources: slides, lecture notes, exercises, old exams.... A perfect study plan doesn’t exist, but it would be nice to have some support for creating one - based on lectures, other commitments, and study preferences.

## Requirements

- Enable lecture additions
- Incorporate key dates like exams, holidays, and other obligations
- Allow students to specify the beginning and conclusion dates for their study sessions
- Provide options for students to input study time preferences and prioritize specific lectures
- Feature the ability to enter different lecture topics for organization
- Allow the integration of study resources such as slides, exercises, past exams, and midterm assessments
- Generate a customized study plan based on the student's input parameters
- Offer flexibility for students to modify their personalized study plans manually
- Include an iCal export function to facilitate study plan integration into student calendars

## Optional enhancements

- Gamification
- Link it with your ETH account, for automatic import of lectures and exam dates
- When adding a new appointment, update the study plan to accommodate for less time
- Prioritize resources (with what does the student prefer to learn)
- Include designated “buffer times”
- Import other calendars containing e.g. holidays or other commitments
- Sharing calendars with others
- Combination of different study plans (e.g. if you have common lectures with a friend, and you would like to study together)

# SceneSleuth: Lecture Scene Search

**Stakeholder:** Kwok Wai Lui

## Overview

Ah, the age-old conundrum of having to be in the Algolab tutorial, NLP exercise, and somehow at ZWEISTEIN — all at 4 PM. Thank goodness for lecture recordings; they're like the MCU of academia: expansive, filled with easter eggs, and dauntingly long. But let's be honest – finding that one golden nugget of wisdom at the 79th minute is like finding a needle in a haystack. If only Shazam and Ctrl+F had a baby...

## Requirements

- **Full-Text Search:** Implement a full-text search feature capable of searching through the lecture. Use natural language processing to improve search accuracy.
- **Scene Playback:** Identify and play back the exact scenes corresponding to the full-text search results.
- **Time-Stamped Links:** Generate links that lead directly to the specific timestamp of the scene within the full video.
- **Advanced Text Search:** Integrate features like fuzzy search and synonym recognition for more comprehensive search results.

## Optional Enhancements

- **Cross-Platform Support:** Make the application compatible with ETH's own video platform: https://video.ethz.ch/.
- **Scene-Specific Questions and Comments:** Incorporate a feature where students can ask questions or make comments tied to specific scenes.
- **Usage Analytics for Lecturers:** Use machine learning to generate insights about which parts of the lectures are being revisited most frequently, helping professors understand student engagement.

## Notes

- Potentially useful resources:
  - https://support.google.com/youtube/answer/6373554
  - https://github.com/yt-dlp/yt-dlp
- An ETH lecture on YouTube:
  - https://youtube.com/playlist?list=PL5Q2soXY2Zi8J58xLKBNFQFHRO3GrXxA9

# HeartWare: Emotional Support Machine

**Stakeholder:** Robert Veres

## Overview

Staying inside and studying all day during the summer can be quite depressing, especially when your friends from UZH have already finished their exams and are having the time of their life. To help CS students get through this, we would like to provide emotional support to make this time of the year more bearable.

Your goal is to build a chatbot that will find encouraging (or maybe not so encouraging, AI might turn evil...) words for students who aren't having the best day. It's up to you to decide whether the AI is actually trying to help or whether it's just a secret weapon with the only goal of eliminating your competition. As long as you're having fun, you're winning!

## Requirements

- Deploy a reasonably sophisticated and emotionally intelligent chatbot.
    - Ensure the bot has access to information about student life at ETH to be able to contextualize user input.
    - Ensure the bot has human contact information to which it can refer the user if it feels unable to manage the situation.
- Make the chat persistent (i.e. refreshing/restarting app shouldn't delete the chat history).
- Wrap everything in a friendly UI with calming audiovisual elements.

## Optional enhancements

- Let the bot have different modes/moods, e.g. supportive mode, evil mode, romantic mode, passive-aggressive mode, ...
- Have the bot impersonate someone/something.
- AI group chat: Talk to multiple bots in the same chat.
- Feel free to scrap all of the above and come up with your own enhancements.