

## 4. Reflection – Our own research question

### Main question:

How can we design and implement a real-time transcription and translation pipeline for one MIJNIP TV live channel that outputs multilingual subtitles (at least four languages) with  $\leq 8$  seconds delay, in a way that is scalable and maintainable within Nidaros' infrastructure?

### Sub-questions:

1. What functional and non-functional requirements (latency, accuracy, accessibility, scalability) must the real-time subtitle system satisfy for MIJNIP TV and its users?
2. Which speech-to-text and machine translation technologies best fit these requirements and can be integrated into Nidaros?
3. How should the system architecture and deployment pipeline be designed so that the subtitle service is stable, observable (logging/health checks), and future-ready for additional workflows?
4. How does the implemented solution perform in realistic test streams, in terms of subtitle delay, continuity, and accuracy for different languages and network conditions?

### Checklist

You can tick these in your worksheet. Here is how we see them:

- **Is my question concrete and specific?**  
Yes. It specifies *one MIJNIP TV live channel, real-time transcription and translation,  $\geq 4$  languages,  $\leq 8$ s delay, and fit with Nidaros' infrastructure.*
- **Can I collect data for this?**  
Yes. We can measure latency, continuity, accuracy and stability using our PoC/MVP tests, simulations in Docker, and system tests described in the research report.
- **Is my question neutral?**  
Yes. It does not assume a specific technology is best. Instead, it asks *how* we can design and implement a solution that meets clearly defined criteria.
- **Is it feasible?**  
Yes. The scope is limited to one channel and an MVP within our project period and builds on an existing PoC instead of starting from zero.

- **Does it contribute to improvement or new insight?**

Yes. It improves accessibility for non-native speakers and hearing-impaired users and gives Nidaros a scalable architecture and concrete performance data they can build on.

### **Reflection (from the team perspective)**

At the start of the project, our thinking about the research question was much more vague: we mainly thought about adding subtitles to the livestream. During the analysis phase, and especially while writing the Fishbone diagram, Requirements Analysis, and Project Plan, we realized that this was not specific or measurable enough for professional ICT research.

As a team, we refined the question step by step. We made it more context-specific by explicitly linking it to *one MIJNIP TV live channel* and Nidaros' infrastructure. We made it more measurable by including concrete criteria such as  $\leq 8$  seconds delay, at least four languages, and quality targets for continuity and accuracy. We also ensured it stayed neutral: instead of asking whether a particular tool (e.g. Whisper or a cloud API) is the best, we ask how we can design and implement a pipeline that meets the requirements, based on evidence from literature, experiments, and tests.

Looking back, we think our current main question fits the checklist well: it is specific, realistic within the project duration, and strongly connected to our research pattern and cards (literature study, user requirements, domain modelling, prototyping, simulation, and system tests). At the same time, we see that the question is still broad enough to allow technical design freedom and iteration. As a team, we learned that formulating a good research question is not a one-time step, but something that evolves as we better understand the client, the users, and the technical constraints.