Chirp! Project Report

ITU BDSA 2025 Group 9

Kristine Nielsen krini@itu.dk
Isabella Bjerre Wahl iswa@itu.dk
Nickolaj Toft Carman nitc@itu.dk
Mikkel Dybro Hansen midh@itu.dk
Marcus Alexander Ryssel Mader almm@itu.dk

1 Design and Architecture of *Chirp!*

1.1 Domain model

Here comes a description of our domain model.

Illustration of the *Chirp!* data model as UML class diagram.

Figure 1: Illustration of the *Chirp!* data model as UML class diagram.

1.2 Architecture — In the small

1.3 Architecture of deployed application

1.4 User activities

1.5 Sequence of functionality/calls trough *Chirp!*

2 Process

2.1 Build, test, release, and deployment

2.2 Team work

2.3 How to make *Chirp!* work locally

2.4 How to run test suite locally

3 Ethics

3.1 License

3.2 LLMs, ChatGPT, CoPilot, and others

During the development of our project, Large Language Models (LLMs) were used for various parts of the project, primarily ChatGPT. Although LLMs were used throughout the project, we were not always fully transparent about their use and did not consistently add, for instance, ChatGPT as a co-author to all commits where it was actually used. The primary reason for this was that at the beginning of the project, we were not aware that this was required. Once we were informed and reminded of this, we began to apply it to our commits. However, there were still instances where we simply forgot to add it as a co-author, particularly when the use of LLMs was minimal and not a significant part of the changes in a commit.

The primary use of LLMs during our project was for error handling and debugging. There were many situations in which we encountered cases where changes to the code caused significant parts of the system to break. As a result, we often had to identify where the issues had occurred, and how to resolve them, sometimes while dealing with a large number of exceptions simultaneously. This was not always straightforward, and LLMs were therefore particularly helpful in such situations. In these cases, we would, for instance, provide ChatGPT with the broken code and the associated exceptions and prompt it to suggest how to handle these. This was for the most part useful, when frustration had begun to build, and we could not resolve the issue on our own, as it often provided us with insight into the mistakes we had made. In many instances, the underlying problems were caused by simple or easily overlooked coding mistakes. Therefore, the LLMs offered us an additional perspective that helped us resolve these issues

efficiently.

A second use of LLMs was for direct code generation. This was, however, used in moderation and primarily for inspiration and guidance on more confusing or challenging tasks, allowing us to incorporate parts of the generated code while still implementing the final solution properly on our own. Lastly, we also decided to use ChatGPT for image generation, creating the profile pictures in our project that depict different colored birds. We chose this, because we wanted images that fit the bird-themed concept of Chirp while maintaining a clean user interface design, and since none of us had the necessary skills or time to create these images ourselves, using ChatGPT provided us with a fast and easy solution.

The advantage of LLMs is that they provide a useful tool when one is uncertain or stuck, offering guidance on how to handle it, and since they are always available, they can help more quickly than, for instance, a teaching assistant, who is limited by time and day. Therefore, LLMs can in most cases speed up the development.

However, LLMs can sometimes also be misleading. We found that, since the LLM lacks full knowledge of the project that only we as developers possess, it occasionally provided incorrect guidance, which slowed our development down rather than speed it up. A significant example of this occurred during the deployment of our application to Azure. For a long time, we believed that our GitHub workflow was the reason the application would not deploy automatically and display the new content, after integrating the EF Core database. Following guidance from ChatGPT, we spent considerable time adjusting the workflow, only to later discover that the actual issue was a missing startup command and an outdated database in Azure (See GitHub Issue #24).

In conclusion, we found that LLMs, when used appropriately, can significantly speed up the development process. However, they should be used in moderation, and while convenient, they can also introduce inefficiencies if completely and uncritically relied upon.