FAKULTÄT FÜR INFORMATIK

DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

# Analyzing Neurodegenerative Diseases with Web Chatbot Typing Behavior

Nicolas Othmar Theodarus

FAKULTÄT FÜR INFORMATIK

DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

# Analyzing Neurodegenerative Diseases with Web Chatbot Typing Behavior

# Analyse neurodegenerativer Krankheiten anhand des Tippverhaltens von Web-Chatbots

| | |
|---|---|
| Author: | Nicolas Othmar Theodarus |
| Supervisor: | Prof. Dr.-Ing. J org Ott |
| Advisor: | Yifan Yang, M.Sc. |
| Date: | 03.12.2024 |

I assure the single handed composition of this bachelor thesis only supported by declared resources,

Munich, 03.12.2024 Nicolas Othmar Theodarus

# Acknowledgements

Thank you everyone :D!

**Abstract**

Neurodegenerative diseases are chronic conditions that destroy and damage part of nervous system of the sufferer over time, especially the brain. This diseases pose a significant challenge for general public health, since the damages are permanent and incurable. This condition happens mainly on elderly people, given that aging is the greatest risk factor. Moreover, early detection of these diseases are inefficient, impractical and only have minuscule success percentage. There is a need for better detection methods that are cost-effective, user-friendly and accurate.

This thesis aims to pave the way of developing the aforementioned better detection methods. This thesis proposes a solution that involve developing a mobile optimized web application to gather typing data from users of different age groups. A clean and robust architecture structure is utilised to guarantee reliability, scalability and maintainability. It should also be ensured that the application is able to effectively process and save the collected data, so that the data can be used for research purposes. The application can also then be developed further with more advanced features. An example of such additional feature would be an analysis section, where typing behaviour data of a person can instantly be analysed with a click of a button.

An analysis of the data will be performed with the goal to find statistical properties. This statistical properties can then be used to categorize each user into their corresponding age groups. Understanding whether certain biomakers, e.g. typing pattern, can be used to differentiate characteristics of a person is the main focus of this thesis. The conclusion derived from this thesis could give insight into the feasibility of utilising biomarkers to effectively monitor health condition of the user. Specifically, the author hopes that this findings would be beneficials for research on detecting early signs of neurodegenerative disorders effectively with a simple method of collecting typing pattern data.

**Zusammenfassung**

Neurodegenerative Krankheiten sind chronische Erkrankungen, die im Laufe der Zeit Teile des Nervensystems der Betroffenen, insbesondere das Gehirn, zerstören und schädigen. Diese Krankheiten stellen eine große Herausforderung für die allgemeine öffentliche Gesundheit dar, da die Schäden dauerhaft und unheilbar sind. Sie treten vor allem bei älteren Menschen auf, da das Älterwerden der größte Risikofaktor ist. Darüber hinaus ist die Früherkennung dieser Krankheiten ineffizient, unpraktisch und hat nur einen verschwindend geringen Erfolgsanteil. Es besteht ein Bedarf an besseren Erkennungsmethoden, die kostengünstig, benutzerfreundlich und genau sind.

Ziel dieser Arbeit ist es, den Weg für die Entwicklung besserer Erkennungsmethoden zu ebnen. In dieser Arbeit wird eine Lösung vorgeschlagen, die die Entwicklung einer für Mobilgeräte optimierten Webanwendung beinhaltet, um Tippdaten von Benutzern verschiedener Altersgruppen zu sammeln. Es wird eine saubere und robuste Architekturstruktur verwendet, um Zuverlässigkeit, Skalierbarkeit und Wartbarkeit zu gewährleisten. Es sollte auch sichergestellt werden, dass die Anwendung in der Lage ist, die gesammelten Daten effektiv zu verarbeiten und zu speichern, so dass die Daten für Forschungszwecke verwendet werden können. Die Anwendung kann dann auch mit erweiterten Funktionen weiterentwickelt werden. Ein Beispiel für eine solche zusätzliche Funktion wäre ein Analysebereich, in dem die Daten zum Tippverhalten einer Person mit einem Klick auf eine Schaltfläche sofort analysiert werden können.

Die Analyse der Daten wird mit dem Ziel durchgeführt, mathematische Eigenschaften zu finden. Diese mathematischen Eigenschaften können dann verwendet werden, um jeden Nutzer in die entsprechenden Altersgruppen einzuteilen. Das Hauptaugenmerk dieser Arbeit liegt auf der Frage, ob bestimmte Biomacher, wie z.B. das Tippverhalten, zur Unterscheidung von Merkmalen einer Person verwendet werden können. Die aus dieser Arbeit abgeleiteten Schlussfolgerungen könnten Aufschluss darüber geben, inwieweit Biomarker zur effektiven Überwachung des Gesundheitszustands des Nutzers eingesetzt werden können. Insbesondere hofft der Autor, dass diese Erkenntnisse für die Forschung zur Erkennung früher Anzeichen von neurodegenerativen Erkrankungen mit einer einfachen Methode zur Erfassung von Tippmusterdaten von Nutzen sein könnten.

# Contents

**AD** Alzheimer's Disease

**PD** Parkinson's Disease

**UI** user interface

**DAO** Data Access Object

**DTO** Data Transfer Object

**LLM** large language model

# Outline of the Thesis

Chapter 1: Introduction
Text

Chapter 2: Background
Text

Chapter 3: Related Work
Text

Chapter 4: Requirements Elicitation
Text

Chapter 5: System Design
Text

Chapter 6: Case Study/Evaluation
Text

Chapter 7: Summary
Text

# Chapter 1

# Introduction

Alzheimer's Disease (AD) and Parkinson's Disease (PD), are chronic and progressive neurodegenerative diseases that primarily affect the nervous system. This diseases lead to the degeneration of motoric and cognitive abilities. These diseases are often irreversible and incurable, posing significant public health challenges. As populations age, the risk of such disorders increase substantially, making early detection crucial.

Historically, the diagnosis of neurodegenerative diseases has been done throuh clinical observations, imaging, and biomarkers. Even though medical technologies have improved significantly over the years, there are still many challenges for early diagnosis. One of the main reason for this is because neurodegenerative diseases develop gradually over time. Early symptoms of these diseases can easily be overlooked or mistaken as normal aging [VRW23]. As a result, these early symptons are often ignored until the disease itself has reached a critical stage, where the chance of treatment declines significantly. Furthermore, current diagnostic methods are expensive, invasive, and often inaccessible to a large portion of the population.

A better method is obviously needed to battle these insidious diseases. The rise of technologies such as smartphones open up new possibilities for early detection of these conditions. One example of such possibility is to utilise a phone application as a mean to detect early neurodegenerative diseases. Studies have shown that one of the effect of these diseases, i.e. impairments of motoric functions, will be reflected on how a person types [MFQM18]. This changes in typing behaviors, such as typing speed, accuracy, and keystroke patterns can then be recorded by the application. The data acquired by this application might be able to be used to analyse early symptons of AD and PD.

## 1.1 Problem

It is clear from the facts mentioned above, that neurodegenerative diseases are problems that need to be addressed. World Health Organisation estimates that there are approximately 50 million people worldwide affected by these disorders. Most of the sufferer are elderly, since age is one of the main risk factor. As the most common neurogedenerative disorder, AD still lacks an effective cure. It is even harder to treat the more progressive the disease progress. That is why the importance of an effective way to diagnose the disease early cannot be overstated. In the current state, however, misdiagnosis rates are still high, reaching up to 20%. Not only that, current diagnostic methods are either invasive, costly, or impractical for widespread use.

Similarly, PD, the second most common neurodegenerative disorder, has no cure and limited treatment options. For this disease too an effective mean for early diagnosis is of utmost importance. Since with a successful early detection, the progression of the disease can be slowed significantly, improving the quality of life of patients greatly. The current diagnostic methods for this disease, however, rely mostly on the observation of changes of motoric symptoms. These methods, as previously discussed, are unreliable for many reasons.

In both conditions, early intervention can significantly improve patient outcomes, but existing diagnostic tools fail to provide a practical and accurate solution for early detection. There is a pressing need for non-invasive, cost-effective, and widely accessible methods to detect early signs of neurodegenerative diseases before the onset of significant symptoms.

## 1.2 Motivation

The motivation for this thesis comes from the urgent need to find better diagnostic methods for neurodegenerative diseases. Finding methods to effectively detect early these diseases would significantly improve general public health. Early detection allows for earlier interventions, which will then slow the progression of the diseases. This would improve treatment outcomes, and ultimately reduce the burden on healthcare systems.

From a scientific point of view, the research on using digital biomarker, i.e. typing pattern, as a mean to detect neurodegenerative diseases is underexplored. This research could give insights into how effective common tools and activities can be used to improve public health. Typing has became a common daily activity for most modern human, especially with the widespread use of smartphones and chat applications. This means that it can be a low-cost and non-invasive method to detect subtle motoric or cognitive impairments. In the most ideal case, the subject would not even notice that they are being monitored for these diseases and can go on about enjoying their daily lives.

Taking advantage of these common daily activities could help reduce the risk of neurodegenerative diseases, by diagnosing them as early as possible. Furthermore, these methods would also be accessible to people that lives in less developed countries with less developed medical technologies. This would ensure equal chances to fight against these neurodegenerative diseases. By developing a mobile-optimized web application that can collect and analyze typing data in real time, it would also become easier to monitor people's condition. Especially the condition of those that are more prone to this diseasesm, i.e. the elderly.

## 1.3 Objectives

Developing a web-based chat application that can capture and analyze typing behavior for early detection of neurodegenerative diseases is an ambitious goal. That goal is unfortunately not in the scope of this thesis. The primary objective of this thesis is to explore whether it is possible to determine the age of a user based on their typing patterns. The author believe that this thesis will pave the way for a more advanced research on this matter. This thesis wants to show that typing pattern can be used to identify the characteristics of the user, in this case, the age group.

Specifically, this thesis aims to:

1. Develop mobile-optimized chat application that collects the user's typing data. The main focus is to collect samples from individuals across different age groups.

2. Practice clean architecture and secure coding practices to make sure the application is reliable, scalable, and maintainable. The chat application will be designed to be user-friendly for all age groups, specifically the elderly.

3. Analyze the gathered typing behavior data to identify patterns or statistical distributions that may correlate with the user's age. Metrics such as typing speed, keystroke intervals, and error rates will be examined to determine if they can give indication of the user's age group.

4. Evaluate the accuracy of using typing behavior as a predictor of age. The identified patterns need to be consistent enough to be able to reliably be used to estimate the user's age group.

The goal of this thesis is to research the feasibility of using data of typing behavior gathered by the application to profile the user in an age group. If this is achieved, the author hopes that this could give insights that would be valuable for future research in user profiling or cognitive assessments. The application could also be further developed to be able to analyse more complex matters, such as early signs of neurodegenerative disorders. Another possible improvement would be adding real-time analysis of the typing pattern and integration with healthcare systems. This would be beneficials for patients and clinicians alike.

# Chapter 2

# Background

## 2.1 Mobile-Optimized Applications and Accessibility for Elderly Users

It is crucial that the application is both mobile-optimized and also accessible to elderly users. Since the author wants to use data of people typing on their smartphones, the application needs to be mobile-optimized. Mobile-optimized application ensures that the data gathered by this application captures typing pattern of its users correctly. Irregularities, such as typing mistakes or longer typing interval, should not be caused by the difficulty of using this application. Instead this irregularities should reflect human error, that most likely to happens more frequently with older subjects. Among other thigs, the user interface (UI) design of the application should be clean and minimalist, focusing on essential features and contents. Unnecessary elements and visual clutters should be removed to create an intuitive UI that is easy to navigate on smaller screens. Since there are many types and sizes of smartphones, it is also important to build a responsive web application that is able to adjust to common screen sizes.

The application also need to be accessible for elderly users. This is done to prevent a false positive condition, where significantly more typing errors happen on elderly subjects because of the difficulty of using the application. Designing an accessible UI for the elderly requires some considerations as suggested by Gomez-Hernandez et. al. in their research in 2023 [GHFMVM23]:

1. **Bigger Interactive Elements:** The size of interactive elements, such as buttons and forms, need to be bigger to make them more accessible. This improvement could help with possible vision or motoric impairments.

2. **High contrast:** Another method to help with vision impairment. This could help increase readability.

3. **Font Selection:** The font used should be easily readable, especially on smaller screens.

4. **Spacing:** Enough spacing should be added to improve readability und user experience.

## 2.2 Backend Architecture: Java Spring Boot

On the backend side of the application, Java Spring Boot framework is utilised to manage and process data that are sent by the frontend of the application. The Spring Framework is an application framework and inversion of control container for the Java programming language. Spring Boot is an open-source Java framework for applications based on Spring to help project startup and management easier. This framework provides libraries that help to develop a scalable, maintainable and secure web applications. It is important to develop the application in this manner so that new features and improvement could easily be added during or after the writing of this thesis. If the occassion arises, this application would also be ready to be reused for a possible follow-up research on this topic.

Spring Boot provides functionalities that help to ensure clean architecture and adherence to coding principles, such as DRY (Don't Repeat Yourself). Another important feature provied by Spring Boot is RESTful services, that support separation of the client and the server. Other than aiding on building a clean architecture, RESTful services also further ensure the possibility of reusing the code for further researches. This application uses Data Access Object (DAO) and Data Transfer Object (DTO) patterns to manage data efficiently. This separation of concerns between DAO and DTO adheres to clean architecture principles, improving maintainability and testability.

## 2.3 Database Management: PostgreSQL

PostgreSQL is an open-source, object-relational database system known for its scalability, reliability and support for complex queries. These attributes make PostgreSQL ideal for storing a huge amount typing data and keystroke logs. Features such as JSONB data type that is offered by PostgreSQL also help to simplify data processing and storing, especially data of keystroke logs. Another reason why PostgreSQL is used in this project is its indexing and search capabilities. This will be crucial for fast retrieval of user data, allowing efficient typing pattern analysis that would be useful if instant analysis feature is ever built.

## 2.4 Large Language Models: Llama3

The Llama3 is the latest large language model (LLM) develop by Meta inc. In addition to Llama3's impressive performance compared to other LLMs, Llama3 is multilingual and can support both english and german language well. This makes Llama3 the perfect language model to use for this thesis. In this thesis, Llama3 will be used to chat with the users in real-time. The language model is prompted to try to get as much response from the users as possible, so that the typing data can be collected.

It is interesting to see whether LLMs such as Llama3 would be able to also analyze the users' typing data. It can potentially help identify changes and irregularities in typing pattern that might indicate the age of the user. If this is possible, a real-time analysis of the typing pattern might be able to be implemented with the help of language models. This topic is, however, not in the scope of this thesis and is only a possible future research.

## 2.5 Containerization: Docker

To further promotes the scalability of the application, the author utilizes Docker. Docker is a platform for delivering software in packages called containers. Containerization is an important part of deploying a scalable application across various environments. Docker allows applications to be deployed separately (frontend, backend, database and LLM) in their own separated containers. This makes it easier for the application to be reused and rebuild in other environments, such as on the university servers.

## 2.6 Pattern Recognition and Analysis of Typing Pattern

To recognize typing pattern of users from different age groups, a statistical analysis of the typing pattern will be carried out.

TODO: find and explain statistical analysis correlation methods

Metrics such as keystroke intervals, typing speed, and error rates will be analyzed to distinguish typing patterns between each age groups.

TODO: find correlation between typing metrics and age group based on the statistical analysis.

TODO: find pattern recognition method or algo such as Hidden Markov Models (HMM), Dynamic Time Warping (DTW), or Neural Networks

## 2.7 Security and Privacy in Data Collection

TODO: do we need this part? Privacy is not as important since the data is not from patients. Maybe explain about anonimity and HTTPS

# Chapter 3

# Related Work

## 3.1 Fine Motor Decline and Psychomotor Impairment Detection

Similar research has been done in the work of Kapsecker et al [KOJ22]. In this research, data of typing behaviors are also accumulated, such as typing speed and variation in character usage. Kapsecker's research, however, used a modified version of the iOS default keyboard to able to gather these data. This modified version of the iOS keyboard brings forwards a limitation in this research, specifically that it shows deviation from the standard keyboard which cause more frequent use of backspace due to typos and different typing behaviors in general. Since the default keyboard of iOS devices is highly optimized, even the slightest changes could affect the user significantly. Differences in structure and layout from this default keyboard, however minor, could cause noticeable changes in the behavior of the users and thus the gathered data

This research has three main findings. The first finding show that the uniform statistical property can be found in the subjects' typing patterns, i.e. their typing speed and their associated overall distribution. The second finding is also regarding the typing speed. The results of the research shows that there is a strong consistency in typing speed between healthy subjects regardless of potential impact factors, such as daytime. This implies that the method of recording typing behavior, in this case with a custom keyboard, is suitable for measuring baseline deviations for both short and long term. The third finding shows that there is a high correlation of approximately 0.8 between frequency and average transition time. It implies that subjects show different transition time during typing characters that are rarely used and more often used. The system used in this research seems sensitive enough to notice these differences.

These findings suggest that it is possible to detect cognitive and psychomotor impairments through recording and analyzing typing behavior. In the effort of extending this research, the author is hoping to achieve similar results while decreasing the limitations, specifically the limitation caused by using custom keyboard.

Van Waes et al. suggested in 2017 that typing tasks might provide a more accessible alternative for both patients and clinicians. Additionally, the research explores the use of keystroke dynamics as digital biomarkers, which could enhance the diagnostic accuracy for detecting fine motor decline associated with neuropsychiatric disorders [VWLME17]. The research highlights how these tasks could serve as a valuable tool in assessing typing and motor skills, which may decline in patients with Alzheimer's disease.

Mastoras et al. suggested in their research in 2019 that typing patterns can be indicative of psychomotor impairment associated with depressive tendencies [MIH+19]. This research contributes to the development of unobtrusive, high-frequency monitoring tools for depressive tendencies, providing a potential method for early detection and intervention in everyday settings. The findings highlight the potential of using everyday interactions with mobile devices as a source of data for mental health monitoring.

A newer research in 2023 by Tripathi et al. showed the recognition of neurodegenerative diseases, such as PD, using typing patterns is an emerging field that leverages keystroke dynamics [TAGG23]. This approach involves analyzing the time it takes for individuals to press and release keyboard keys during typing, known as hold time, as well as the time between keystrokes, referred to as flight time. These metrics can be used to detect signs of PD in an ecologically valid setup, such as at the subject's home.

These researches highlighted the possibility of detecting fine motor decline and psychomotor impairment through typing pattern on a keyboard. They showed that through analysing keystroke dynamics, flight time and hold time, psychomotor impairment can be detected.

## 3.2 Age Deduction Based on Typing Pattern

In an article published in 1984, Salthouse revealed some interesthing insights on his research on the effect of age on typing pattern [Sal84]. Older typists are generally not slower in overall typing speed compared to younger typists. However, older typists show slower performance in some areas: tapping rate (the speed of repetitive finger movements) and choice reaction time (the time between a stimulus and an action). Older typists tend to compensate this by being more conscious about characters farther ahead in the text they are typing. This in turn makes it so that older typists are able to maintain a more constant typing speed compared to their younger counterparts.

TODO: add more research on this topic

To the researcher knowledge, there has not been a study about using typing behavior on a mobile optimized application to infer the age group of a user.

# Chapter 4

# Requirements Elicitation

This chapter follows the Requirements Analysis Document Template based on Bernd Bruegge's *Object Oriented Software Engineering Using UML, Patterns, and Java* [BD09]. The goal is to illustrate the concepts, taxonomies, and relationships of the application domain independent of the chosen technology and development platform. The following sections provide an overview of the system, describe functional and nonfunctional requirements, and outline important system models.

## 4.1 Overview

The system being developed has the purpose of collecting typing pattern of its users with the purpose of inferring their age group by analysing their typing patterns. Potential future improvements of the application include the addition of real-time analysis and result. It is also possible to use the system to gather the data for follow-up research on detecting early signs of neurodegenerative diseases with typing pattern. These potential future improvements are, however, beyond the scope of this thesis.

The system's primary objectives are to collect, process and save keystroke data efficiently, so that the data can be analysed. To achieve this, the application also needs to provide a condusive condition for the users, so that they are able to type as they usually would. This means that the application would need to integrate and prompt the LLM to elicit response from the users. The LLM will be prompted to elicit as many response from the users as possible, so that more data can be gathered and the analysis can be more accurate.

User experience is also an important part of the application. The user should feel comfortable using the application as they would usually use another chatting application. Accessibility, especially for the elderly users, needs to be taken into account. This is important to make sure that differences in typing pattern are only caused by internal factors of the user themselves. Success of this application will be measured by the ability of the system to gather, process and save keystroke data accurately and effectively.

## 4.2 Requirements

The application will provides a condusive condition for the user to type. Typing pattern of the user will be collected, processed and saved in the database for later to be analyzed. The proposed system will be split into functional and nonfunctional requirements.

### 4.2.1 Functional Requirements

The functional requirements (FR) define the specific actions that the system must perform. Below is a list of the primary functional requirements:

FR1 **Condusive Typing Environment**: The user should feel comfortable using the system, as they would using other chatting application.

FR2 **Elicite User Response**: The system must be able to elicit response from the users by using LLM.

FR3 **Collect Typing Data**: The system must capture and log keystroke data from the user in real time, including keypresses and timestamps.

FR4 **Store Typing Data**: The system must securely store all captured typing data in the database for future analysis.

FR5 **Store Users' Age**: The system must gather users' age to act as a comparison between infered and actual age group of the user.

FR6 **Ensure Data Anonymization**: The system must anonymize user data before analysis to ensure compliance with privacy standards.

### 4.2.2 Nonfunctional Requirements

The nonfunctional requirements (NFR) define system constraints, performance criteria, and standards the system must adhere to. These requirements are categorized using the FURPS+ model, as described in Bernd Bruegge's book [BD09], excluding functionality, which was covered in the functional requirements section.

TODO: describe unrealisable functionalities

NFR1 **Usability**: The user interface must be easy to use, especially for elderly users. Design of the application should be accessible for the elderly with features such as large fonts, enough spacing, and clear visual indicators.

NFR2 **Reliability**: The system must be able to run and save users' typing pattern without loss of data or functionality.

NFR3 **Performance**: The system should give response to the user within 10 seconds after the user send a message.

NFR4 **Security**: All communication between the frontend and backend must be encrypted using HTTPS. The user's information other than their age should be anonym. Chat session of users should be able to be locked, so that it cannot be accessed anymore.

NFR5 **Adaptability**: The system must be able to scale to support more users, e.g. by adding more servers. It should also be possible to add functionalities such as real-time analysis.

## 4.3 System Models

This section includes key system models for requirements analysis, including scenarios, use case models, object models, dynamic models, and user interface.

### 4.3.1 Scenarios

A scenario is "a narrative description of what people do and experience as they try to make use of computer systems and applications" [Car95]. The following section will show concrete, focused and informal description of features of the application from the viewpoint of the user.

#### Visionary Scenarios

The following scenario describes an ideal scenario that might be achieved in a future research.

| | |
|---|---|
| *Scenario name* | neurodegenerativeDiseaseDetection |
| *Participating actor instances* | bob: User |
| *Flow of events* | 1. Bob, a man in his late 50s, use a chatting application to chat with his grandson. This chatting application is already integrated with a system to capture and analyze typing pattern.<br>2. Even though Bob does not notice it at all, the application records his typing pattern while simultaneously analysing it. After some times, the application notifies Bob that the system has detected with 90% certainty early symptoms of AD on Bob. Bob then seeks medical help to confirm this analysis. |

**Table 4.1:** Neurodegenerative diseases detection using typing pattern analysis

**Demo Scenarios**

The following scenario is implemented by the current system. It is the main purpose and functionality of the application.

| | |
|---|---|
| *Scenario name* | ageGroupDetection |
| *Participating actor instances* | bob: User |
| *Flow of events* | 1. Bob, a man in his late 50, uses the application to chat with a LLM. The application logs and stores Bob's keystrokes.<br>2. This keystrokes data is then later analyzed. After the analysis is done, it is correctly inferred that Bob is in the age group of 50 to 60 years old. |

**Table 4.2:** Age group detection using typing pattern analysis

### 4.3.2 Use Case Model

The following use case model show the interactions between the user, the server and the LLM.

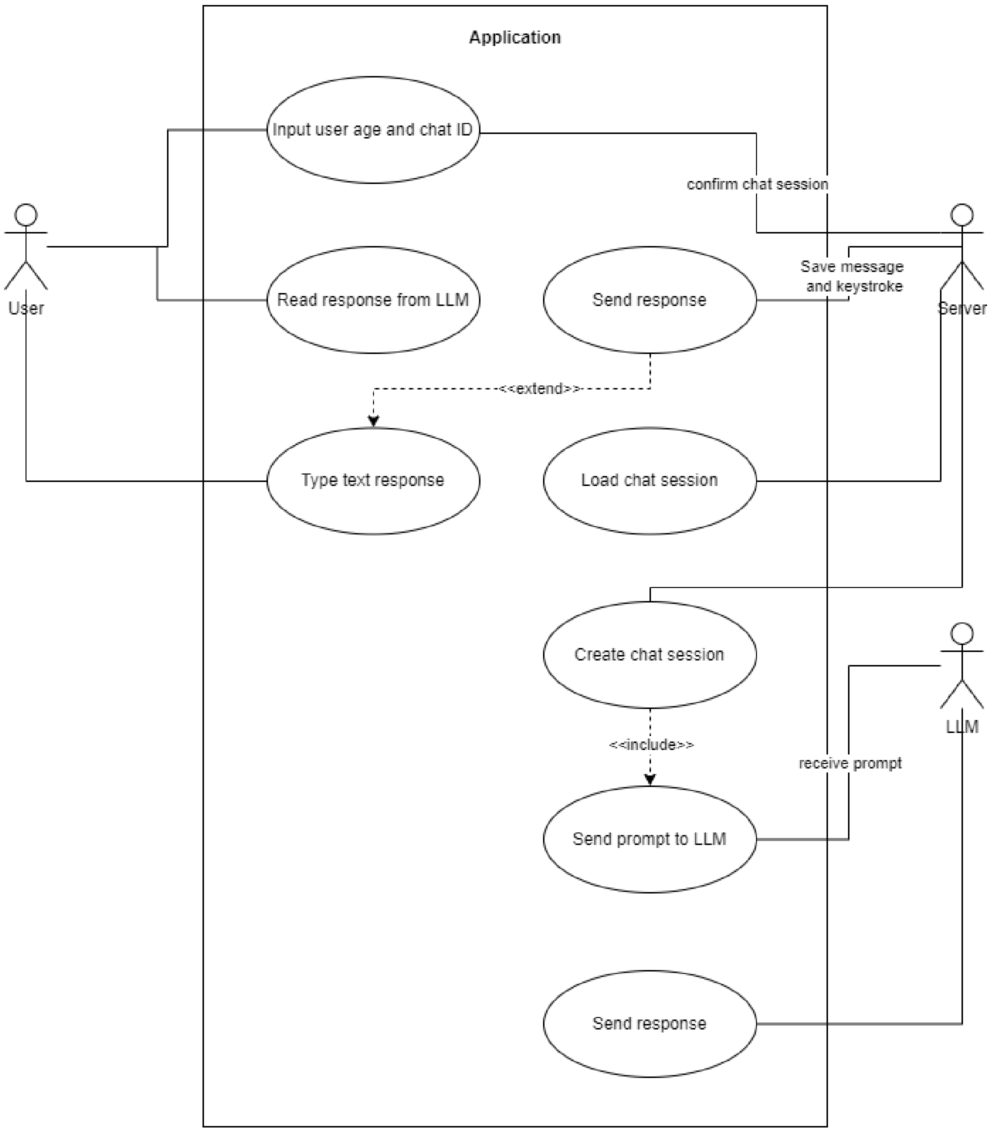


**Figure 4.1:** UML use case diagram of the chatting feature

| | |
|---|---|
| *Use case name* | initialization of chat session. |
| *Participating actors* | Initiated by user.<br>Communicates with server and LLM. |

| | |
|---|---|
| *Use case name* | initialization of chat session. |
| *Flow of events* | 1. User inputs age and chat ID.<br>2. Backend server confirms that the ID is valid and accessible.<br>3. Backend server checks that no corresponding session with this ID has been created and initialises a new chat session.<br>3. Backend server gives a prompt to LLM.<br>4. LLM loads and send the first message after it is finished loading.<br>5. User reads the message from the LLM |
| *Entry condition* | • User opens the login page of the application. |
| *Exit condition* | • User reads the first message from the LLM. |
| *Quality requirements* | • The server should be able to checks that the corresponding chat of the given ID is accessible and has no chat history.<br>• The server should initialise the chat session correctly and save the age user data corresponding to this chat session in the database.<br>• The server should send a prompt to the LLM, wait for the response, and forward the response of the LLM to the user. |

**Table 4.3:** Use case initialization of chat session

| | |
|---|---|
| *Use case name* | loading of an already existing chat session. |
| *Participating actors* | Initiated by user.<br>Communicates with server and LLM. |

| | |
|---|---|
| *Use case name* | loading of an already existing chat session. |
| *Flow of events* | 1. User inputs age and chat ID.<br>2. Backend server confirms that the ID is valid and accessible.<br>2. Backend server checks that there exists already a chat session with the corresponding ID.<br>3. Backend server loads all the messages history of this chat session and sends it to the user.<br>4. User reads the chat history and continues the chat as they left it. |
| *Entry condition* | • User opens the login page of the application. |
| *Exit condition* | • User is able to read all of the chat history belonging to the session with the ID given by the user. |
| *Quality requirements* | • The server should be able to checks that the corresponding chat of the given ID is accessible and has chat history.<br>• The server should load all chat history of this chat session correctly.<br>• The server should send the chat history data to the user with right information of whether a message is sent by the user or by the LLM. |

**Table 4.4:** Use case loading of an alredy existing chat session

| | |
|---|---|
| *Use case name* | user sends a message. |
| *Participating actors* | Initiated by user.<br>Communicates with server and LLM. |

| | |
|---|---|
| *Use case name* | user sends a message. |
| *Flow of events* | 1. User types a response to the last message from the LLM.<br>2. User sends the message.<br>3. Backend server saves the message and its keystroke data in the database.<br>4. Backend server sends the message to the LLM.<br>5. LLM responses to the user's message.<br>6. Backend server saves the message from the LLM in the database and sends it to the user. |
| *Entry condition* | • User is already in a chat session. |
| *Exit condition* | • User is able to read the message from the LLM. |
| *Quality requirements* | • The server should save the message from the user and its keystroke data in the database correctly.<br>• The server should forward the message to the LLM.<br>• The server should wait for the LLM to response, . |

**Table 4.5:** Use case user sends a message

### 4.3.3 Analysis Object Model

The following diagram shows the key objects and their relations with other objects in the application.

**Classes**: Message, Conversation, Keystroke

Conversation: represents the whole chatting session with a unique ID. When the user enters an ID, the backend server will look for a conversation that has the same ID. If such object exists, then the server will fetch the conversation along with the Messages that belong to this conversation object. If the object does not exist, then the server will initialise a new conversation object with ID that just has been given by the user. Each conversation object also has age attribute, which give information about the age of the user to whom this object belongs to. To improve security, each conversation is also

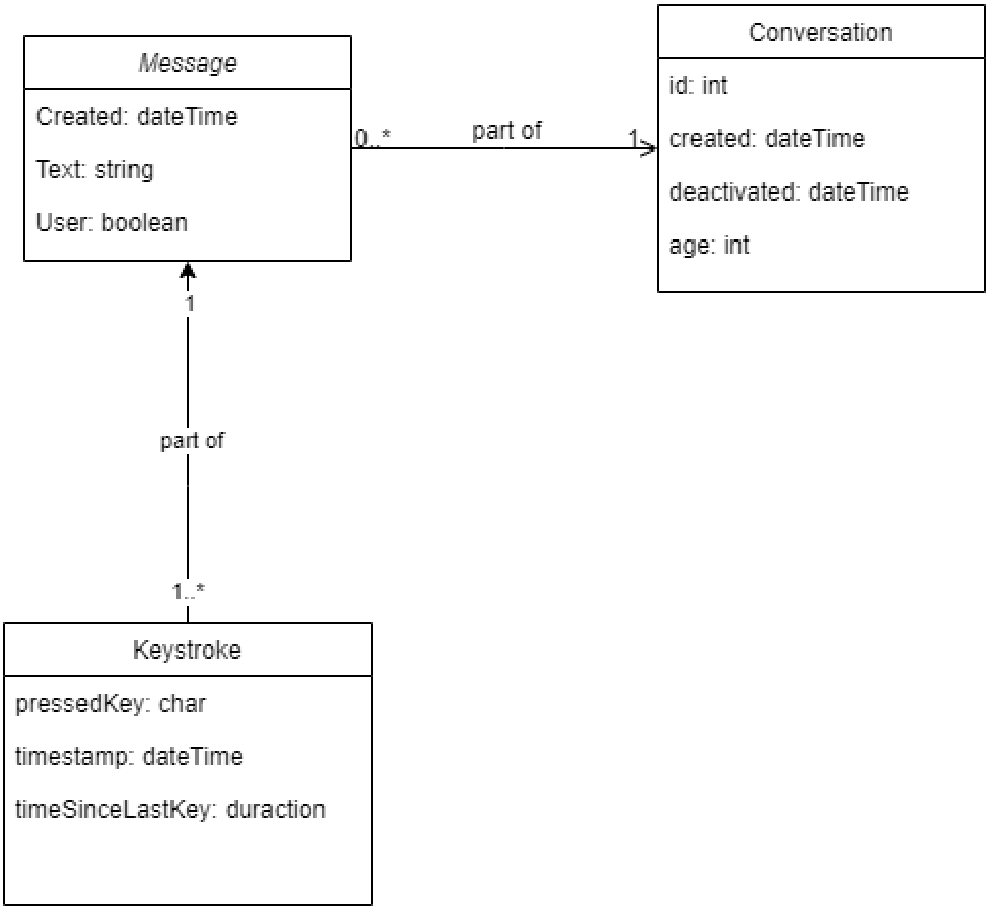


**Figure 4.2:** UML class diagram of the application domain

given the attribute deactivated. If this attribute is set, then the conversation will not accessible anymore. This helps prevent unwanted actors to access chat sessions of users that are already finished.

Message: represents each message that the user or the LLM sent. Each message object is related to one conversation. Whereas each conversation contains between zero and multiple messages. The messages from the LLM also need to be saved, so that the whole conversation can be loaded again, after the user exits a chat session. As a result, the message objects need to have an attribute to give information, whether this message was created by the user or by the LLM. This information is saved in the boolean attribute "user".

Keystroke: represents each keystroke that the user types. Each message object will have between one and multiple keystrokes. The keystroke object is the most important object for this thesis, since the reseach will be focused on the keystrokes of the user. Each keystroke object will have three attributes: pressedKey, timestamp, and timeSinceLastKey. Each key the user pressed will be saved as a keystroke object, such as alphabets, numbers, and control characters, such as backspace or horizontal tab. This information is saved on the attribute pressedKey. The attribute timeSinceLastKey will show the duration between each key presses. Statistical analysis of the keystrokes will be based on the value saved on the attribute timeSinceLastKey.

### 4.3.4 Dynamic Model

Figure 4.3 depicts UML state diagram of the application. This diagram shows the states the application will be in based on certain triggers. The entry point of the application would be the login page. This will be the first page the user accessed. If the user inputted the correct ID, i.e. accessible ID with the right format, then the second condition will be checked. If the ID is not correct, then a warning will be shown.

The second condition consists of checking whether the chat session corresponding to the ID has been initialised. An initialised chat session means that the chat session has already been accessed before. This chat session possibly already has chat history that needs to be loaded. If, on the other hand, the entered ID is not associated with any chat session, this means that this is the first time the user entered this ID. In this case, the chat session needs to be initialised first. After the chat session is initialised, then a prompt is given to the LLM. The first message from the LLM is then sent to the user.

Either after the first message from the LLM or the whole chat history of the chat session is loaded, the application enters the next state. In this state, the application waits for the user to type and send the message. Message that is sent by the user is then processed in the backend server. To begin with, the message is sent to the LLM. The server then waits for the LLM to respond to the text. It should be mentioned here that the server sends the whole chat history to the LLM, so that the context of the previous messages is not lost. After the LLM responded, both the new message from the user and from the LLM are then saved in the database. In the end, the new message from the LLM is forwarded to the user. This bring the application back to the state of waiting for the user to send a message.

### 4.3.5 User Interface

This is the user interface

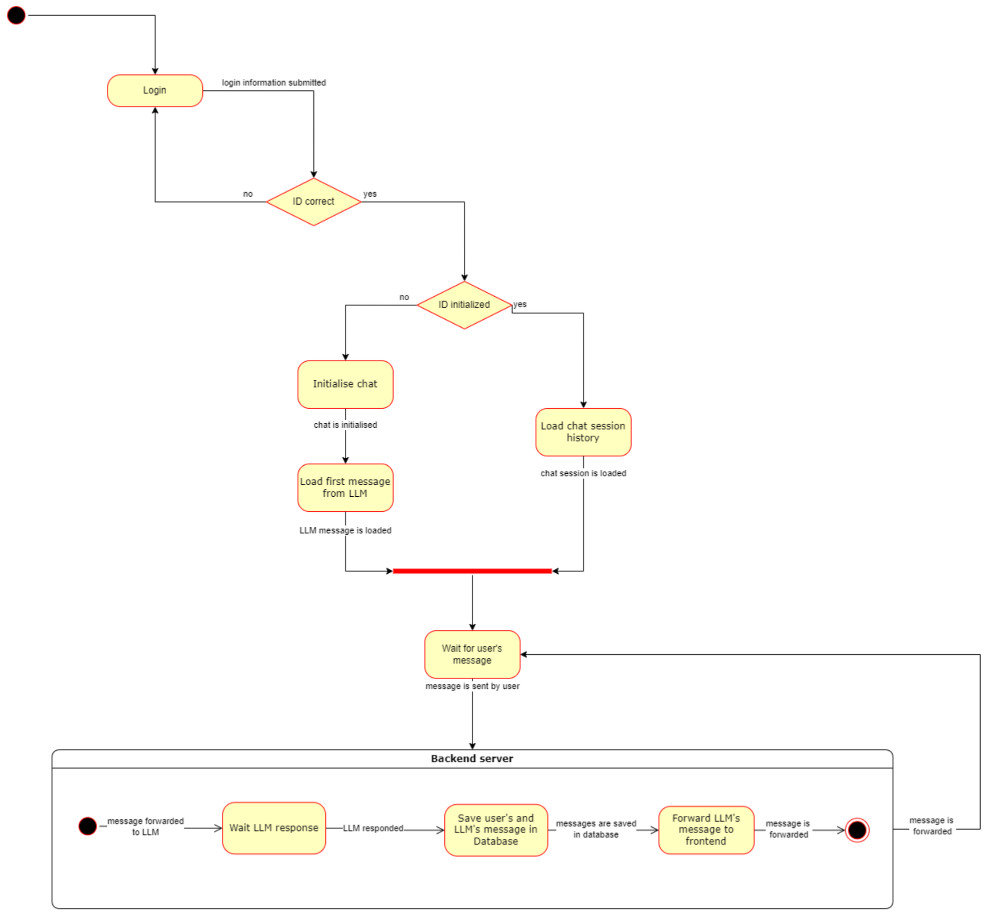


**Figure 4.3:** UML class diagram of the application domain

# Chapter 5

# System Design

*Note: This chapter follows the System Design Document Template in [BD09]. You describe in this chapter how you map the concepts of the application domain to the solution domain. Some sections are optional, if they do not apply to your problem. Cite [BD09] several times in this chapter.*

## 5.1 Overview

*Note: Provide a brief overview of the software architecture and references to other chapters (e.g. requirements analysis), references to existing systems, constraints impacting the software architecture.*

## 5.2 Design Goals

*Note: Derive design goals from your nonfunctional requirements, prioritize them (as they might conflict with each other) and describe the rationale of your prioritization. Any trade-offs between design goals (e.g., build vs. buy, memory space vs. response time), and the rationale behind the specific solution should be described in this section*

## 5.3 Subsystem Decomposition

*Note: Describe the architecture of your system by decomposing it into subsystems and the services provided by each subsystem. Use UML class diagrams including packages / components for each subsystem.*

## 5.4 Hardware Software Mapping

*Note: This section describes how the subsystems are mapped onto existing hardware and software components. The description is accompanied by a UML deployment diagram. The existing components are often off-the-shelf components. If the components are distributed on different nodes, the network infrastructure and the protocols are also described.*

## 5.5 Persistent Data Management

*Note: Optional section that describes how data is saved over the lifetime of the system and which data. Usually this is either done by saving data in structured files or in databases. If this is applicable for the thesis, describe the approach for persisting data here and show a UML class diagram how the entity objects are mapped to persistent storage. It contains a rationale of the selected storage scheme, file system or database, a description of the selected database and database administration issues.*

## 5.6 Access Control

*Note: Optional section describing the access control and security issues based on the nonfunctional requirements in the requirements analysis. It also describes the implementation of the access matrix based on capabilities or access control lists, the selection of authentication mechanisms and the use of encryption algorithms.*

## 5.7 Global Software Control

*Note: Optional section describing describing the control flow of the system, in particular, whether a monolithic, event-driven control flow or concurrent processes have been selected, how requests are initiated and specific synchronization issues*

## 5.8 Boundary Conditions

*Note: Optional section describing the use cases how to start up the separate components of the system, how to shut them down, and what to do if a component or the system fails.*

# Chapter 6

# Case Study / Evaluation

*Note: If you did an evaluation / case study, describe it here.*

## 6.1 Design

*Note: Describe the design / methodology of the evaluation and why you did it like that. E.g. what kind of evaluation have you done (e.g. questionnaire, personal interviews, simulation, quantitative analysis of metrics, what kind of participants, what kind of questions, what was the procedure?*

## 6.2 Objectives

*Note: Derive concrete objectives / hypotheses for this evaluation from the general ones in the introduction.*

## 6.3 Results

*Note: Summarize the most interesting results of your evaluation (without interpretation). Additional results can be put into the appendix.*

## 6.4 Findings

*Note: Interpret the results and conclude interesting findings*

## 6.5 Discussion

*Note: Discuss the findings in more detail and also review possible disadvantages that you found*

## 6.6 Limitations

*Note: Describe limitations and threats to validity of your evaluation, e.g. reliability, generalizability, selection bias, researcher bias*

# Chapter 7

# Summary

*Note: This chapter includes the status of your thesis, a conclusion and an outlook about future work.*

## 7.1 Status

*Note: Describe honestly the achieved goals (e.g. the well implemented and tested use cases) and the open goals here. if you only have achieved goals, you did something wrong in your analysis.*

### 7.1.1 Realized Goals

*Note: Summarize the achieved goals by repeating the realized requirements or use cases stating how you realized them.*

### 7.1.2 Open Goals

*Note: Summarize the open goals by repeating the open requirements or use cases and explaining why you were not able to achieve them.* ***Important:*** *It might be suspicious, if you do not have open goals. This usually indicates that you did not thoroughly analyze your problems.*

## 7.2 Conclusion

*Note: Recap shortly which problem you solved in your thesis and discuss your* ***contributions*** *here.*

## 7.3 Future Work

*Note: Tell us the next steps (that you would do if you have more time. be creative, visionary and open-minded here.*

# Appendix A

# e.g. Questionnaire

*Note: If you have large models, additional evaluation data like questionnaires or non summarized results, put them into the appendix.*

# Appendix B

# Tips for writing a thesis in TeX

## B.1 using this template

This template tries to achieve a separation of the template itself and the parts that are specific to the thesis. Ideally, the template does not have to be edited.

The content of the thesis shall be added to the following files and folders:

- the `.tex`-files in the `chapters`-folder shall contain the description of your scientific work.
- the `.tex`-files in the `resources`-folder contain templates and examples, into which metadata, settings and organisational information about the thesis can be entered.
- the `thesis.bib`-file shall contain a list of the literature, that you cite in your thesis.

## B.2 General tips

Track your work on this thesis with a version control system such as git.

In your TeX source code, use one line per sentence. This facilitates spotting excessively long sentences. Also, it makes the tracking of changes by the version control system more useful. If you add line breaks after a fixed number of columns instead, a change affects all subsequent lines of the paragraph, even though the actual contend has not been changed.

It is recommended to create a folder, in which all images, that are included in this document are stored. See the `resources/settings.tex`-file, on how to add this folder to the default graphics path, so only the filenames have to be entered, when including an image.

# List of Figures

# List of Tables

# Bibliography


[BD09] Bernd Bruegge and Allen H Dutoit. *Object Oriented Software Engineering Using UML, Patterns, and Java.* Prentice Hall, 2009.

[Car95] John M. Carroll, editor. *Scenario-based design: envisioning work and technology in system development.* John Wiley & Sons, Inc., USA, 1995.

[GHFMVM23] M. Gomez-Hernandez, X. Ferre, C. Moral, and E. Villalba-Mora. Design guidelines of mobile apps for older adults: Systematic review and thematic analysis. *JMIR Mhealth Uhealth*, 11:e43186, 2023.

[KOJ22] Maximilian Kapsecker, Simon Osterlehner, and Stephan M. Jonas. Analysis of mobile typing characteristics in the light of cognition. In *2022 IEEE International Conference on Digital Health (ICDH)*, pages 87–95, 2022.

[MFQM18] Tara L. McIsaac, E. Nora Fritz, Lori Quinn, and Lisa M. Muratori. Cognitive-motor interference in neurodegenerative disease: A narrative review and implications for clinical management. *Frontiers in Psychology*, 9(02061), 2018.

[MIH+19] R.-E. Mastoras, D. Iakovakis, S. Hadjidimitriou, V. Charisis, S. Kassie, and T. et al. Alsaadi. Touchscreen typing pattern analysis for remote detection of the depressive tendency. *Scientific Reports*, 9(1):1–12, 2019.

[Sal84] T. A. Salthouse. Effects of age and skill in typing. *J Exp Psychol Gen*, 113(3):345–371, 1984.

[TAGG23] S. Tripathi, T. Arroyo-Gallego, and L. Giancardo. Keystroke-dynamics for parkinson's disease signs detection in an at-home uncontrolled population: A new benchmark and method. *IEEE Trans. Biomed. Eng.*, 70(1):182–192, 2023.

[VRW23] Manera Valerias, Erika Rovini, and Peter Wais. Early detection of neurodegenerative disorders using behavioral markers and new technologies: New methods and perspectives. *Frontiers in Aging Neuroscience*, 15(1149886), 2023.

[VWLME17] L. Van Waes, M. Leijten, P. Mariën, and S. Engelborghs. Typing competencies in alzheimer's disease: An exploration of copy tasks. *Computers in Human Behavior*, 73:311–319, 2017.