HelpMeHelp Application

Main Report

Version 1.1



Group members:

Adrian Paul Limpiado Balunan

Roar Andre Fagerkind

Meenakshi Jayachandran

Cathrine Kristiansen

Robin Strand Prestmo

Revision History

Date	Version	Description	Author
24.03.2026	0.1	- Set up the document	- Robin Strand Prestmo
26.03.2026	0.2	- Written Description assignment - Written Cap 1. Introduction - Started Cap 1.1 Definitions, Acronyms, and Abbreviations - Written Cap 2.1 Agile Development - Written Cap 2.2 Software Testing	- Robin Strand Prestmo - Robin Strand Prestmo - Robin Strand Prestmo - Robin Strand Prestmo - Robin Strand Prestmo
30.03	0.3	- Written Cap 2.3 User-centered design - Written Cap 2.4 Universal design	- Robin Strand Prestmo - Robin Strand Prestmo
31.03	0.4	- Written Cap 3.1 Methods and Standards used - Written Cap 3.2 Use of Literature and Internet	- Robin Strand Prestmo - Robin Strand Prestmo
01.04	0.5	- Written Cap 3.3 Hardware used - Written Cap 3.4 Software used - Written Cap 3.5 Sustainability analysis method	- Robin Strand Prestmo - Robin Strand Prestmo - Robin Strand Prestmo
07.04	0.6	- Written Cap 2.5 GDPR	- Cathrine Kristiansen
09.04	0.7	-Written Cap 3.7 Work and Role Distribution -Written Cap 5.2 Evaluation of Requirements -Written Cap 5.4 Technology, Process and Tools	- Cathrine Kristiansen - Cathrine Kristiansen - Cathrine Kristiansen

		-Written Cap 5.5 Technical Challenges and Limitations -Written Cap 5.6 Reflection on Team Process -Written Cap 5.8 Risk and Sustainability	- Cathrine Kristiansen - Cathrine Kristiansen - Cathrine Kristiansen
10.04	0.8	-Written Cap 5.7 -Written Cap 5.3	- Cathrine Kristiansen - Cathrine Kristiansen
14.04	0.9	- Started writing Cap 3.6 Documentation Overview	- Robin Strand Prestmo
15.04	0.9	- Finished writing Cap 3.6 Documentation Overview - Written Cap 4.1 Project Results	- Robin Strand Prestmo - Robin Strand Prestmo
17.04	0.10	- Written Cap 4.2 Administrative Results	- Cathrine Kristiansen
20.04	0.11	- Written Appendix -Written Cap 5.1 Final product and result	- Cathrine Kristiansen and Robin Strand Prestmo - Cathrine Kristiansen
21.04	0.12	- Written Appendix - Written Cap 6 Conclusion - Written Preface	- Cathrine Kristiansen - Cathrine Kristiansen - Robin Strand Prestmo
22.04	0.13	- Written Abstract - Updated Cap 1.1 Definitions, Acronyms, and Abbreviations -Written Appendix	- Cathrine Kristiansen - Robin Strand Prestmo - Cathrine Kristiansen

23.04	1.0	- Written Appendix	- Cathrine Kristiansen -Robin Strand Prestmo -Roar Fagerkind
23.04	1.1	- Removed Appendix, and moved it to new file	- Robin Strand Prestmo

Abstract

This report describes the development process of a Java-based desktop application, Help Me Help (HmH). The application has been developed as part of the course IDATT1005 System Development at NTNU. The project aims to develop a solution that makes it easier for users to find, evaluate and support charitable organizations through a user-friendly application.

The project was implemented using an iterative process inspired by Scrum, which was held in several sprints. The development process was carried out with good planning, monthly work requirements, prototyping, system implementation and multiple tests. The solution was developed with user-centered design, universal design, privacy and sustainability in mind. At the same time, the team used tools such as GitHub, Microsoft Teams, Figma, IntelliJ, Scene Builder and JUnit throughout the project.

The result of the project is a functional prototype that includes key features such as user profiles, organization profiles, search, filtering, simulated donation flow, and communication through messaging. The application presents verified information about organizations and aims to increase trust and facilitate informed donation choices.

The report also discusses challenges and limitations related to database management, GitHub workflows, external integration, and the lack of a real payment solution. It also reflects on teamwork, project execution, risk, sustainability, and further development opportunities.

Working on the project has provided valuable experience in collaboration and technical software development. Through a structured and iterative development process, the project led to a functional prototype that can engage more people in charitable work, contributing positively to society.

Preface

This report is written as part of the course IDATT1005 System Development at NTNU, as a part of the bachelor's degree in data engineering. The project has been carried out during the spring semester of 2026. The main purpose of the project was to gain experience working in a team to develop a software system. The group chose to develop the HelpMeHelp application due to its relevance and potential to support meaningful causes, which is a Java-based desktop application designed to make it easier for users to find and support charitable organizations.

The development process was structured using agile methods and organized into multiple sprints. The work included planning, design, implementation and testing. Tools like GitHub were used for version control, and Microsoft Teams were used for sharing documentation. Regular meetings ensured steady progress through the project.

Through the project, we gained valuable experience in teamwork, project planning and iterative software development. We have also learned a lot within Java programming and teamwork using agile methods.

We would like to thank our teaching assistant Tri Tac Lee for his guidance and support through the project, and to Oshadha Ratnayake, Asbjørn Risøy, Mats Vestvik and Onur Zubari, for giving us good feedback on our user tests.

Trondheim, April 2026
Team 6

Signatures

Cathrine Kristiansen

Adrian Paul Limpiado Bauman

Roar Andre Fagerkind

Meenakshi Jayachandran

Robin Strand Prestmo

Description Assignment

Due to increasing global instability, including wars, conflicts and disasters, there is a growing need for humanitarian support. At the same time, the number of charitable organizations is increasing, making it difficult for individuals to decide which causes to support.

This project concerns the development of a Java desktop application called Help Me Help (HmH). The goal of the application is to provide users with an overview of relevant charities and make it easier to search, filter and select organizations to support. The application will also enable donations directly through the platform.

The system is developed from scratch, with a focus on usability, accessibility and intuitive design. Emphasis is placed on creating a clear and user-friendly interface that is accessible to a wide range of users.

The application is designed to present users with featured organizations on the front page, as well as provide functionality for filtering and searching among charities. In addition, the system includes login functionality, allowing users to create profiles and manage their interactions with the platform.

Registered users will be able to donate, view previous donations, and mark favorite charities. Charitable organizations will be able to create their own pages, where they can describe their purpose and define donation goals.

Further requirements are described in the vision document and use case specifications.

Contents

1. Introduction 3

    1.1 Definitions, Acronyms, and Abbreviations 4

2. Theory and relevant literature 5

    2.1 Agile Development 5

    2.2 Software Testing 6

    2.3 User-centered design 7

    2.4 Universal Design 8

    2.5 GDPR 9

3. Method 12

    3.1 Methods and Standards Used 12

    3.2 Use of Literature and Internet 13

    3.3 Hardware used 13

    3.4 Software used 14

    3.5 Sustainability analysis method 15

    3.6 Documentation overview 15

    3.7 Work and Role Distribution 16

4. Results 19

    4.1 Project results 19

    4.2 Administrative results 20

5. Discussion 22

    5.1 Final Product and Results 22

    5.2 Evaluation of Requirements 23

        5.2.1 Functional Requirements 23

5.2.2 Non-Functional Requirements 24
5.3 Deviations from Plan and Project Outcome 25
5.4 Technology, Process and Tools 26
5.5 Technical Challenges and Limitations....27
5.6 Reflection on Team Process 27
5.7 Risk and Sustainability 28
5.8 Recommendations for Future Work 29

6. Conclusion....31

Repository 33

References 34

Table of Figures and Tables

Table 1 - Definitions, Acronyms and Abbreviations 4

1. Introduction

This report presents the development process and results of the Help Me Help (HmH) application, created as part of the IDATT1005 Software Engineering course at NTNU. The goal of the project is to develop a desktop application where users can easily find and donate to charitable organizations.

The project was carried out through a planning phase followed by three development sprints. The planning phase included team organization, establishment of a collaboration agreement, configuration of GitHub, and distribution of roles within the team.

The first sprint focused on creating key project documentation, including the project manual and vision document, as well as developing initial wireframes and conducting user testing based on these prototypes. The second sprint involved the development of core system models and diagrams, such as the domain model, use case diagram, sequence diagrams and class diagram, in addition to implementing and testing a Minimum Viable Product (MVP). The third and final sprint focused on completing the application and preparing it for presentation.

The remainder of this report is structured as follows. Chapter 3 covers relevant theory and literature. Chapter 4 describes the methods used in the project, including work and role distribution. Chapter 5 presents the project results, including both technical and administrative outcomes. Chapter 6 discusses the results, and Chapter 7 concludes the report and suggests further work. The repository, references and appendix are included at the end of the report.

1.1 Definitions, Acronyms, and Abbreviations

Table 1 - Definitions, Acronyms and Abbreviations

Term/Acronym	Definition
HmH	Help Me Help, the application developed in the project.
NTNU	Norwegian University of Science and Technology.
IDATT1005	Course code for Software Engineering at NTNU.
MVP	Minimum Viable Product.
UI	User Interface.
UX	User Experience.
WCAG	Web Content Accessibility Guidelines.
GDPR	General Data Protection Regulation.
IK	Innsamlingskontrollen.
IDE	Integrated Development Environment.
CI	Continuous Integration.
JUnit	Java testing framework.
FXML	Markup language is used in JavaFX.
JavaFX	Java framework for GUI development.
Git	Version control system.
GitHub	Code hosting and collaboration platform.
GitHub Actions	Automation tool for CI/CD workflows in GitHub.
Figma	UI design and prototyping tool.
SusAF	Sustainability Awareness Framework.
SusAD	Sustainability Awareness Diagram.
Scrum	Agile framework for iterative development.

2. Theory and relevant literature

2.1 Agile Development

Agile development is a software development approach where work is divided into smaller tasks that are completed in short, iterative cycles. Instead of developing the entire system at once, the work is split into manageable parts with shorter deadlines (Norozi, 2026). This approach makes it possible to identify errors and issues early in the process, allowing them to be corrected before they have a larger impact.

Scrum is a widely used agile framework. In Scrum, one team member takes the role of Scrum Master, who is responsible for monitoring progress and facilitating daily stand-up meetings (Norozi, 2026). The Scrum Master helps ensure that the team stays on track and addresses any obstacles that arise during development.

Development in Scrum is organized into cycles called sprints, which typically last between one and four weeks. Each sprint includes planning activities where tasks and features are identified and prioritized, and where the team decides what to work on during the sprint (Norozi, 2026). At the end of each sprint, the results are demonstrated, and the team evaluates the process to identify what worked well and what could be improved.

Another important concept in agile development is Minimum Viable Product (MVP). An MVP is a version of the product that includes only the core functionality that is required for the application to be stable. The purpose of the MVP is early testing and feedback.

2.2 Software Testing

Software testing is important to ensure that a system works as intended. It is used to identify and remove defects before the system is used by end users (Kathayat, 2026). When few or no errors are discovered by users, the application is perceived as reliable and of high quality. In agile development, testing is performed continuously throughout the development process.

There are several reasons why software testing is essential. Early detection of defects helps reduce development costs, as errors become more difficult and expensive to fix later in the process (Kathayat, 2026). Testing also reduces risk, for example by preventing issues related to data loss or security vulnerabilities. As a result, effective testing contributes to higher software quality and increased user trust.

There are multiple types of software testing. Unit testing focuses on testing small parts of the system in isolation (Kathayat, 2026). This makes it easier to identify where defects occur. Integration testing verifies that different components of the system work correctly together. In addition, positive testing ensures that the system behaves as expected with valid input, while negative testing verifies that the system handles invalid input appropriately.

User testing is particularly important for applications developed for end users. Users often interact with a system differently than developers, which can reveal usability issues and unexpected problems. This type of testing provides valuable feedback that can be used to improve the system.

Software testing is a shared responsibility within the development team. Everyone must contribute to ensure the quality of the system is at a high level.

2.3 User-centered design

User-centered design is an approach where the system is developed with a strong focus on the users and their needs. The goal is to create a system that is visually appealing, efficient, and easy to use. To achieve this, the system is developed with continuous feedback from user testing (Sandstrak, 2026b).

This approach is important because it improves usability and user satisfaction. A system that is easy to use reduces the risk of user errors and makes the application more effective. There are several methods used in user-centered design.

Wireframes are used to create prototypes of the user interface (UI). This helps visualize the application early in the development process, allowing developers to design and adjust the system at an early stage (Sandstrak, 2026a). When the prototype is tested on users, feedback can be collected on what works well and what needs improvement, allowing the design to be continuously refined.

Another important aspect of user-centered design is universal design. This ensures that the system is accessible to as many users as possible, including users with disabilities (Sandstrak, 2026b). This can include aspects such as color choices, clear layout, appropriate font sizes, and accessible interaction options.

In this project, user-centered design was applied through the use of wireframes and user testing. The design focused on simplicity, accessibility, and making it easy to find and support charities.

2.4 Universal Design

Universal Design is important to consider when designing an application. It focuses on making the application accessible to as many users as possible, not only users with disabilities, but everyone. There are guidelines for this called WCAG.

WCAG stands for Web Content Accessibility Guidelines and includes four main principles for accessibility (Sandstrak, 2026b). The first principle is “Perceivable”, which means that information and user interface elements must be presented in ways that users can detect. The second principle is “Operable”, meaning that users must be able to interact with the interface as intended. The third principle is “Understandable”, which means that content and navigation should be clear. The fourth principle is “Robust”, which ensures that the system works across different devices, browsers, and assistive technologies.

These principles can be applied in practice in several ways. Perceivable content can be achieved through sufficient contrast between text and background, and by providing alternative text for images (Sandstrak, 2026b). Operable interfaces ensure that users can navigate and interact with the system, for example by using a keyboard. Understandable systems rely on consistent design and predictable behavior. Robust systems are designed to function across different platforms and technologies.

Universal design is an important part of modern software development, as it ensures that digital solutions can be used by a wider audience and improves overall usability.

2.5 GDPR

GDPR (General Data Protection Regulation) is an EU regulation that Norway has been obligated to follow since 2018 through the EEA agreement. The regulation applies to all organizations that process personal data and aims to protect individuals' privacy and ensure control over their personal data stored in digital systems (Tranvik, Introduksjon GDPR, 2020).

GDPR sets requirements for how personal data is collected, stored, and used. This includes both general information, such as name and email address, and more sensitive data such as health information, political opinions, and religious beliefs. Such data can be vulnerable if it is exposed or misused (Tranvik, Personopplysninger GDPR, 2020).

Personal data is defined as any information that can be linked to an identifiable individual, such as images, video, audio recordings, text, and IP addresses (Tranvik, Personopplysninger GDPR, 2020).

Privacy is therefore particularly important in the development of digital services and user applications. In the *Help Me Help* application, personal data is processed in connection with registration, login, storage, and data transfer. This makes it necessary to comply with GDPR regulations in order to ensure users' rights and security (Tranvik, Personopplysninger GDPR, 2020).

To meet the requirements of GDPR, we have emphasized several key privacy principles in the development of *Help Me Help*. First and foremost, consent is essential, and users must actively accept terms before their personal data is collected and used within the application. This is particularly important during user registration and when making donations (Tranvik, Grunnleggende Prinsipper GDPR, 2020c).

Furthermore, we focus on data minimization, meaning that we only collect necessary information, such as name and email, and avoid storing unnecessary sensitive data. In addition, it is

important that users have control over their own data, and therefore they should be able to access, modify, or delete information related to their account.

The application also offers an anonymous mode, allowing users to donate without linking their identity to the action. This is an important measure to protect privacy and provide users with greater security.

To further ensure data protection, information must be stored securely, for example through encryption and protected systems. This reduces the risk of data breaches and unauthorized access.

Ethical Reflection:
In addition to complying with GDPR, it is important to consider ethical aspects in the development of the application. Users must be able to trust that their data is handled in a responsible and transparent manner. This includes being clear about how data is used and avoiding solutions that may be perceived as manipulative, such as pressuring users to donate. By prioritizing transparency, security, and voluntariness, the application can help build trust among users.

To build trust between the user and the platform, credibility is essential, particularly through accurate information about organizations. The “verified” label must be genuine and not misleading. Therefore, we rely on accurate data from Innsamlingskontrollen when marking an organization as “verified by IK.” This introduces an external actor into the trust framework, meaning that part of the responsibility for credibility is shared with Innsamlingskontrollen.

Credibility is also strengthened through transparency by informing users where their money goes and how data and personal information are handled. This information must be presented clearly and not hidden within long and inaccessible terms and conditions.

Regarding anonymity and privacy, it is important that users have the option to donate anonymously if they wish to protect their privacy. This can increase security and lower the threshold

for donating. However, anonymity can also create challenges related to control and security. It may become more difficult to trace transactions and verify the origin of funds, which can potentially enable misuse such as money laundering. Thus, anonymity enhances privacy but may also make the system more vulnerable to misuse.

To prevent users from feeling pressured to donate, it is important to avoid manipulative design (dark patterns). The application should not pressure users into donating but instead facilitate voluntary and informed choices. UX design plays a key role in ethical considerations, as the design of the interface can influence users' decisions and emotions. To address this, we have chosen a balanced design of donation buttons that does not create stress or urgency through wording such as "donate now." Additionally, a conscious decision has been made not to clearly display how much each individual donates, in order to avoid creating feelings of guilt.

It is important to avoid donations being driven primarily by strong emotional reactions, as this may negatively affect the user's financial situation. Good UI/UX should therefore support users in making thoughtful decisions without excessively influencing their emotions, for example through the overuse of large "Donate" buttons or time-limited countdowns. Users should not feel guilty for choosing not to donate.

Data security is important for users, organizations, and the application itself. A data breach would represent a serious breach of trust, particularly concerning personal data and payment information. This could have significant consequences for both individuals and organizations if sensitive data or financial resources are compromised. Users expect their data to be handled securely, which is essential for maintaining trust and protecting individual integrity.

Overall, this demonstrates that the development of the application is not only about technical solutions but also about ethical decisions that affect users' safety, trust, and freedom of choice.

3. Method

3.1 Methods and Standards Used

This project applied iterative and incremental development approaches inspired by Scrum. The work was divided into multiple iterations, enabling continuous improvement of both functionality and design.

User requirements identified in the Vision document, such as the need for user-friendliness, secure data handling, and trustworthy presentation of information, were addressed throughout the entire development process.

Prototyping was introduced early in the process to visualize the system and evaluate usability. This ensured that the requirement of providing a clear and intuitive user experience was considered from an early stage.

User testing was conducted on early prototypes to evaluate usability and identify potential improvements. This approach helped reveal issues at an early stage, reducing the time and effort required to address them later in the development process.

Furthermore, established standards and best practices, including modular development, version control, and unit testing, were applied to ensure maintainability and reliability of the system.

3.2 Use of Literature and Internet

Literature and internet resources were used to support both theoretical understanding and practical implementation throughout the project.

Course lectures provided the primary theoretical foundation, covering topics such as Scrum, software testing, and user-centered development. These concepts guided the overall development process and supported decisions related to system design and quality.

Internet resources were mainly used to solve implementation-related challenges. This included documentation and online forums related to Java and JavaFX, as well as general troubleshooting during development.

One challenge was the large amount of available information online, which made it difficult to identify the most relevant solutions. This was addressed by comparing multiple sources and testing different approaches before selecting a suitable implementation.

This approach ensured that both theoretical knowledge and practical problem-solving contributed effectively to the development process.

3.3 Hardware used

The development of the application was carried out using personal computers. No specialized hardware was required, as the project focused on developing a desktop application.

The application was developed and tested in standard computing environments, including Windows and macOS. This ensures that the system can run on typical user devices without requiring advanced hardware.

3.4 Software used

The development of the application was carried out using Java as the primary programming language. JavaFX was used to develop the graphical user interface, enabling the creation of a structured and interactive desktop application. Scene Builder and FXML were used to design and organize the user-interface components.

The project was developed using an integrated development environment (IDE), which supported efficient coding, debugging, and project management. Maven was used as a build and dependency management tool, allowing for easy handling of external libraries and consistent project structure.

Version control was managed using Git and GitHub, which supported collaboration between team members and ensured proper version tracking throughout the development process. Continuous Integration (CI) was implemented using GitHub Actions, which automatically ran tests and verified the build when changes were pushed to the repository.

Testing was performed using JUnit, focusing on unit testing of core functionality. This helped ensure that individual components behaved as expected. Code quality was supported through tools such as Checkstyle, which enforced coding standards and helped identify potential issues early.

Javadoc was used to document the codebase, improving readability and maintainability. This made it easier to understand the structure and functionality of the system for both developers and future contributors.

Design and prototyping of the user interface were carried out using Figma, which allowed early visualization of the system and supported user-centered design decisions.

3.5 Sustainability analysis method

The sustainability analysis of the system was based on the Sustainability Awareness Framework (SusAF). This framework was used to identify and evaluate potential sustainability impacts of the application.

To support the analysis, the Sustainability Awareness Diagram (SusAD) was used as a tool to visualize both negative and positive impacts. The analysis considered environmental, social, economic, individual, and technical aspects of sustainability.

This framework made it possible to structure the analysis and identify how design choices may affect sustainability in both short and long term. A more detailed analysis is presented in the Vision document.

3.6 Documentation overview

The project documentation consists of several structured documents that together guide and support the development and validation of the Help Me Help (HmH) application. These documents ensure that the system is developed in accordance with both the assignment requirements and the overall project vision.

The Vision Document functions as the central foundation for the project. It defines the purpose and overall goals of the system, as well as identifying key system requirements. It also includes both functional and non-functional requirements, providing a clear specification of what the system should do. In addition, the document addresses important aspects such as sustainability, risk analysis, and system constraints, ensuring that both technical and societal considerations are taken into account during development.

The requirements are further detailed through use-case descriptions and domain models. These models describe how users interact with the system and how core entities are structured. Together, they contribute to a clearer understanding of the system vision and support the implementation of its functionality.

Design documentation, including wireframes and usability considerations, was developed to support the creation of a user-friendly interface. Particular emphasis was placed on universal design principles, ensuring that the system can be used by a wide range of users. This is also reflected in the non-functional requirements, where usability and accessibility are explicitly addressed.

Evaluation and quality assurance are documented through user testing and technical testing. User testing provides feedback on usability and interaction design, while automated testing, such as unit tests, ensures that the system behaves as expected. These activities support continuous improvement and help ensure that the final system meets both functional and non-functional requirements.

Together, these documents form a coherent set of artifacts that support the entire development process, from planning and implementation to evaluation. They also ensure traceability between requirements, design decisions, and the final system.

3.7 Work and Role Distribution

This subsection describes how the team organized itself throughout the project and how responsibilities and tasks were distributed among the team members.

The team adopted a structured approach based on defined roles from the project contract. The main roles included Product Owner, Scrum Master, Archive/Document Responsible, and Software Quality Responsible, in addition to rotating roles such as Meeting Leader and Reporter.

The Scrum Master (Adrian Paul Limpiado Balunan) was responsible for facilitating collaboration within the group, ensuring that the team followed the Scrum framework, and maintaining a productive and supportive working environment.

The Archive and Document Responsible (Cathrine Kristiansen) had overall responsibility for maintaining and organizing project documentation. This included ensuring consistency and quality across documents, although all team members contributed to writing.

The role of Software Quality Responsible (Roar Andre Fagerkind, Robin Strand Prestmo, and Meenakshi Jayachandra) focused on ensuring that the technical implementation met required standards throughout the project. This included reviewing code, testing functionality, and supporting overall software quality.

The Product Owner role was defined as an external user perspective, with responsibility for defining and prioritizing product features.

In addition to the fixed roles, the team implemented rotating responsibilities during meetings. Each meeting had a Meeting Leader, responsible for guiding the agenda and ensuring structure, and a Reporter, responsible for documenting key points and decisions. These roles rotate between members to ensure equal participation and shared responsibility.

Although specific roles were assigned, the team worked collaboratively across all areas of the project. All members contributed to both development and documentation, but with some natural distribution based on individual strengths and interests.

Some team members contributed more heavily to programming tasks, while others focused more on documentation and structure. However, no tasks were strictly isolated, and collaboration was emphasized to ensure shared understanding and knowledge across the team.

To avoid overly one-sided work distribution, tasks were actively rotated within the team. For example, members who had worked extensively with documentation were given the opportunity to take on programming tasks, and vice versa. This approach helped ensure a more balanced workload, increased overall competence within the team, and provided all members with experience across various parts of the project.

Work progress and task allocation were managed through a progress plan, GitHub issue boards, meetings, and regular group discussions. This allowed the team to continuously monitor progress and redistribute tasks when necessary.

The team followed a structured workflow with weekly meetings with the student assistant and regular internal meetings and stand-ups. Stand-up meetings were used to provide quick updates on progress, while more formal meetings were used to discuss challenges, motivation, and planning. Communication was primarily handled through Messenger, while Microsoft Teams and GitHub were used for documentation, collaboration, and version control. All work was reviewed before submission to ensure quality and consensus within the group.

A key aspect of the team's approach was flexibility and mutual support. If a task became too demanding or a member experienced difficulty, responsibilities were redistributed within the group. This collaborative approach ensures steady progress and reduces the risk of delays or individual overload.

4. Results

4.1 Project results

The main goal of the project is to develop a Java-based desktop application that makes it easier for users to discover, evaluate, and support charitable organizations. The system was designed to present reliable charities and provide users with a simple way to donate. In addition, the system aimed to follow principles of universal design, ensuring that it is accessible and usable for a wide range of users.

At the time of delivery, the core functionality of the system has been implemented. The application allows users to create and manage user profiles, explore available charities, and view relevant information about each organization. The system presents this information in a clear and structured way, making it easy for users to understand the purpose and activities of each charity. Functionality for donating to charities is also included in the system as a simulated donation process.

The user interface has been developed with a focus on usability and accessibility. Feedback from user testing indicates that the system is generally perceived as structured and easy to navigate. However, as the testing was conducted on a limited number of users, the evaluation of universal design is somewhat limited. The results still provide an indication that the design decisions have contributed positively to the overall user experience.

Testing has been an important part of the development process. Automated unit tests have been implemented using JUnit to verify the behavior of key methods and components in the system. These tests contribute to ensuring that the core functionality works as intended. In addition, user testing has been conducted, where users interacted with the system and provided feedback on usability and functionality.

Overall, the project has resulted in a functional system that meets the primary goals defined at the beginning of the project. The system demonstrates the intended functionality and provides a foundation for further development and improvements.

4.2 Administrative results

The administrative aspects of the project were handled through structured planning, collaboration, and the use of digital tools. The team followed an iterative development process based on Scrum. This helped organize the work into manageable phases and maintain steady progress throughout the project.

Task management was primarily handled through GitHub issue boards, allowing the team to track and update tasks continuously. However, task distribution was assigned at weekly meetings through a progress plan. The progress plan was shared publicly on Microsoft Teams so that everyone could follow the progression along the way. This contributed to a clear overview of the progress and made it easier to redistribute work when needed.

Communication within the team was maintained through a combination of various tools. Messenger was used for quick and continuous communication, while all documentation and files were managed through Microsoft Teams. Regular meetings were held weekly, and consisted of both internal meetings, retrospectives, and weekly meetings with the student assistant. This helped ensure compliance, monitor progress, and identify challenges early.

Documentation was an important part of the administrative work. The team developed key project documents such as the vision document and worked continuously on the final report together with the system development. This ensured that the documentation remained consistent with the evolving system.

Several aspects of the administrative process worked well. The use of structured workflows, regular meetings and collaboration tools contributed to effective coordination and smooth progress.

At the same time, some challenges were experienced. The use of multiple communication platforms occasionally led to fragmented information flow, where not all messages reached every team member. Weekly meetings therefore played an important role in ensuring alignment and improving communication within the team. Access restrictions in GitHub also meant that the group needed additional coordination within the team.

Overall, the administrative processes supported the development of the project, although improvements could be made in terms of communication structure and tool integration in future work.

5. Discussion

5.1 Final Product and Results

The end product of this project is a prototype of the application Help Me Help, which is developed as a Java-based desktop system. The application allows users to find and explore charities, access verified information, simulate donations in a user-friendly interface.

Several features were implemented, such as creating and managing user profiles and donation history. You can also specify interests and interact with organizations. Organizations can create their own profiles where they have the opportunity to update information, communicate with users and get an overview of donations.

At the same time, the application also includes a search and filtering function, and a carousel with selected organizations. Organizations can also receive feedback from users, in the form of comments related to the organization's page.

The organization itself has the ability to send messages to users who have donated to them. These messages are not intended as a direct one-on-one conversation, but rather in the form of general communication, such as a "thank you for the contribution" or information about upcoming events. Although the database is structured so that direct communication is technically possible, this is not implemented as part of the solution's functionality.

The donation functionality itself was implemented as a simulated flow in the application, but is not integrated with a real payment system. Some data is also handled locally, without full integration with external data sources such as the Innsamlingskontrollen (IK).

The final product contains many of the project's original ideas and solutions, and demonstrates a user-friendly solution. At the same time, the prototype has limitations related to payment solutions, external integrations and scalability, which should be worked out during further development into a production-ready application.

5.2 Evaluation of Requirements

5.2.1 Functional Requirements

Most of the functional requirements defined in the Vision Document were addressed in the final application. Core features such as browsing organizations, searching, filtering, viewing verified information, and interacting with donation functionality were successfully implemented.

In addition, both user -and organization profiles were implemented with custom functional behavior. Users can create accounts, log in, manage their profile, view donation history, define interests, and communicate through a message system. Organizations can update their information, view incoming donations, and interact with users.

Several features were implemented in the application with a working logic, including search functionality, filtering of organizations, and a dynamic highlighted section that displays randomly selected organizations.

However, some aspects of the system remain limited. The donation functionality is implemented as a simulated process and is not connected to a real payment system. Additionally, the integration with external data sources such as Innsamlingskontrollen (IK) is not fully automated, and some data is handled locally within the application.
The anonymity feature is implemented at a functional level within the system but would require further development and security considerations in a production environment.

5.2.2 Non-Functional Requirements

The non-functional requirements described in the Vision Document were considered throughout the project to varying degrees. Overall, the system aligns with several key quality attributes, although some areas remain limited due to the scope of the project.

Security and privacy were important considerations in both the design and implementation of the system. Features like authentication, profile management, and an optional anonymous donation contribute to a reduction of unnecessary exposure to personal data. GDPR principles like data minimization and user control over personal information were considered. However, the system does not implement full production-level security mechanisms, such as advanced encryption, secure authentication protocols, or real-world identity verification. To ensure that the design fully follows GDPR requirements in a real-world setting, further development of the system would be necessary.

Usability and user experience were among the most important aspects of the application. The interface was developed with consideration for UX principles and universal design. Users can easily navigate the system, search and filter organizations, and perform actions such as donating or managing their profile with minimal effort. It was important to emphasize ethical design to avoid manipulative elements such as time pressure or overly aggressive calls to action. The system supports well-informed and voluntary decision-making, which contributes to a positive user experience.

The system operates efficiently and reliably within the framework of a local Java application. Functions such as search, filtering, and navigation respond quickly, providing a smooth and responsive user experience. However, the system has not been tested under high load or real-world conditions with multiple users. Therefore, performance and reliability should be further evaluated in future development.

The use of a modular Java-based architecture supports structured development and future extensions, making the system easier to maintain and further develop over time. Since the system lacks full backend infrastructure and integration with external services, scalability is currently limited. Additional development will therefore be required to support a larger number of users and real-time data synchronization.

Overall, most non-functional requirements have been partially fulfilled, with strengths in usability and design. Areas such as scalability, security, and compliance require further development in order to reach a production-ready level.

5.3 Deviations from Plan and Project Outcome

The project was initially structured with clear phases and milestones defined in a Gantt chart. Although the team followed the original plan, some deviations occurred due to technical challenges and an underestimation of the project's size.

A prototype was developed using Figma, which included key features such as user and organization profiles, search and filtering functionality, and a well-functioning donation flow. Whit iterative development, the team achieved steady work progress and maintained effective communication. At the same time, the team encountered challenges related to backend development, database management, and system integration, which proved to be time-consuming. Limitations in the database environment and issues with GitHub workflows contributed to further delays.

As a result, some features were simplified or implemented at a prototype level. This experience highlights the importance of realistic time estimation and flexibility in development processes. Despite these deviations, the project achieved its key features, and the work provided valuable insights into both technical challenges and project execution.

5.4 Technology, Process and Tools

The development of the application was carried out using a combination of technologies, development processes, and collaboration tools, which supported both the technical implementation and the overall workflow of the project.

To meet the project requirements, the application was developed as a Java-based desktop application. Java was chosen because of its strong support for object-oriented programming, stability, and compatibility with relational databases. This simplified the implementation of features such as user management, search, filtering, and system logic. At the same time, the use of Java-based desktop application we experienced some limitations in the modern user interface framework and integration with external services.

Prototyping and design were carried out using Figma, where we developed Wireframes through iterations and user tests, which helped improve usability prior to implementation and reduced design-related issues later in development.

The iterative development process was based on Scrum, structured into three main iterations: planning and vision, development of the MVP, and finalization. This approach allowed gradual system development with continuous improvement. However, challenges were experienced, particularly related to time estimation and task planning, as some development tasks proved more complex than expected.

Version control and task management were handled through GitHub, supporting collaboration and structure. Microsoft Teams was used for documentation and file sharing, while Messenger was used for quick communication. The use of multiple platforms sometimes led to fragmented information and required additional coordination.

Overall, the combination of technologies, processes, and tools contributed positively to the project. The iterative approach and prototyping supported structured development, while tools such

as GitHub and Figma enabled collaboration and implementation. At the same time, challenges related to planning, coordination, and technical complexity provided valuable learning experiences.

5.5 Technical Challenges and Limitations

The team encountered several technical challenges that affected both implementation and system functionality.

One of the main challenges was related to the database environment, where limitations in permissions restricted the ability to create triggers and manage database users. This reduced the possibility of implementing more advanced data handling mechanisms, requiring some functionality to be simplified or handled within the application logic instead of the database.

The team also experienced issues related to version control and workflow management in GitHub. The project utilized both NTNU's GitHub environment and a repository hosted under a personal account, which limited access to repository settings. As a result, only one team member could configure features such as GitHub Actions and runners. To address this, custom runners were manually configured using authentication tokens, allowing the rest of the team to participate in the workflow.

These challenges highlight the importance of proper infrastructure and access management and demonstrate how external constraints can impact development and collaboration.

5.6 Reflection on Team Process

The team's work process was characterized by strong collaboration, shared responsibility, and frequent communication. Although roles were defined, the team worked flexibly across tasks to ensure progress and a shared understanding of the system.

A key strength of the team was its collaborative approach. Members contributed across both development and documentation, which reduced dependency on individuals and increased overall understanding. The rotation of tasks also supported skill development and engagement within the group.

Regular meetings and group discussions helped maintain progress, identify challenges early, and adjust tasks when needed. The use of digital tools such as Messenger, Microsoft Teams, and GitHub supported continuous communication and coordination.

The use of Scrum and iterative development contributed to structure and predictability, allowing the team to focus on short-term goals while progressing toward the final product. However, some challenges were experienced, particularly related to communication structure and time estimation.

Overall, the team process functioned well and contributed positively to the project outcome. The experience highlighted the importance of flexibility, collaboration, and continuous communication in handling challenges during development.

5.7 Risk and Sustainability

The risk analysis identified several potential technical, social, and organizational challenges, many of which proved relevant during development.

Technical risks, such as bugs, database limitations, and workflow issues, had the most direct impact on the project. Database restrictions and GitHub workflow challenges affected implementation and required additional coordination within the team.

Other risks, such as security breaches, identity theft, and GDPR non-compliance, did not arise during the project, but are still important to consider when implementing a real-world application.

These factors were considered in design decisions, especially regarding data minimization and user privacy.

The risk of inaccurate charity information was mitigated by relying on verified data from Innsamlingskontrollen (IK), which contributed to trust and reliability in the system.

The sustainability analysis was used to evaluate the system across multiple dimensions. From an individual perspective, the application supports informed decision-making by providing structured and verified information. From a social perspective, it contributes to increased trust and transparency, although there is still a potential risk of unequal visibility between organizations.

From a technical perspective, the system demonstrates a modular and maintainable design, but scalability is still limited due to a lack of testing under real-world conditions. Economically, the system has the potential to improve the efficiency of donation processes, while the absence of a real payment solution limits this effect.

From an environmental perspective, the system reduces the need for physical resources, although increased reliance on digital infrastructure may lead to higher energy consumption if scaled.

Overall, the analyses contributed to increased awareness of potential challenges, different impacts, and guided the team through design decisions throughout the project. While the prototype shows several positive aspects, further development is needed to fully address long-term risks and sustainability concerns.

5.8 Recommendations for Future Work

The current application demonstrates a functional prototype, but further development is required to reach a production-ready solution.

From a technical perspective, implementing a real payment system would significantly increase the value of the application, as the current donation functionality is limited to a demonstration. In addition, further development of the backend and database infrastructure is necessary to support robust data persistence, user management, and reliable performance.

The system has not been tested under real-world conditions with multiple users, and future development should therefore focus on scalability, including performance optimization, and handling increased data volumes.

Further user testing should also be conducted to improve usability and better understand user behavior. While the current design follows UX principles, real user feedback would provide valuable insights and support improvements such as enhanced filtering, recommendation systems, and personalized features.

In future development, the system architecture could be allowed for one-to-one communication between users and organizations but are by current state not fully implemented. To increase accessibility, the application could be extended to other platforms, such as mobile or web solutions. At this stage, the current system relies on data from Innsamlingskontrollen (IK), and by integrating additional trusted data sources the application could provide a better overview of trusted charities.

Finally, further work should focus on strengthening security and ensuring full GDPR compliance. The balance between anonymity and potential misuse should also be explored to maintain both user privacy and system integrity.

Overall, while the prototype meets many of the initial project goals, additional development is necessary to create a fully functional, secure, and scalable system suitable for real-world use.

6. Conclusion

The result of the project was a prototype of the Help Me Help application, which demonstrates how a digital solution, with good and reliable information, can help users make good donation choices. The system contains important features such as user and organizational profiles for better communication and direct information flow, search and filtering functions and a good donation flow that provides a user-friendly application.

The project ended up with many of the original features for the system, such as usability and implementation of core functionality. Iterative development, prototyping and continuous good collaboration facilitated steady progress, and made it possible for the theme to improve technical and design-related implementations along the way.

At the same time, the team experienced challenges and limitations in both database management and functional limitations that meant that some functions were implemented at prototype level. The lack of a real payment system and full integration with external data sources created limitations in relation to the system's current solutions.

The sustainability and risk analysis mentioned several positive contributions and potential challenges related to the system. It describes several potential changes that are possible for the application, such as more openness, trust and accessibility in the donation process, while also mentioning potential risks related to privacy, security and fairness. An attempt has been made to take these factors into account in the system solution, which should be further taken into account in future development of the application.

Overall, the project was solved through a well-structured, iterative approach and regular communication, which led to a functional and meaningful solution. The current prototype provides a good and solid foundation for further development of the project, if one wants a fully scalable, secure and production-ready system. The team has gained experience with valuable insight into both technical development and collaboration through project work, which will be useful in future software development projects.

Repository

GitHub Repository: https://git.ntnu.no/cathrkri/systemutviklingTeam6

GitHub Wiki, for Team 6: https://git.ntnu.no/cathrkri/systemutviklingTeam6/wiki

Issues/Issueboard, for Team 6:

- https://git.ntnu.no/cathrkri/systemutviklingTeam6/issues
- https://git.ntnu.no/users/cathrkri/projects/5

Javadoc, for Team 6: https://git.ntnu.no/cathrkri/systemutviklingTeam6/tree/develop/docs/Java-doc/apidocs

References

Kathayat, S. (2026, Januar 20). Software Testing [Presentation, IDATT1005, NTNU].

Norozi, M. A. (2026, Januar 13). Agile System Development Methods, [Presentation, IDATT1005, NTNU].

Sandstrak, G. (2026a, Januar 29). Brukersentrert systemutvikling, [Presentation, IDATT1005, NTNU].

Sandstrak, G. (2026b, Januar 29). Brukersentrert systemutvikling Del 2, [Presentation, IDATT1005, NTNU].

Tranvik, T. (2020, February 09). *Introduksjon GDPR*. Retrieved from Youtube: https://www.youtube.com/watch?v=AD4k622u6AY

Tranvik, T. (2020, February 09). *Personopplysninger GDPR*. Retrieved from Youtube: https://www.youtube.com/watch?v=TRdjQotZoCY

Tranvik, T. (2020c, February 09). *Grunnleggende Prinsipper GDPR*. Retrieved from Youtube: https://www.youtube.com/watch?v=Z-8JAApyGSY

Reflection note

Team 6

This section summarizes the team's experiences from the project work and collaboration throughout the development period, based on retrospectives after each sprint.

Throughout the project, the team experienced a gradual development in both the work process and collaboration. Good strengths such as work effort and communication were already present from sprint 1, but areas for improvement in the division of labor and planning of tasks proved to need to be strengthened. Here, an important foundation was laid for further work with well-structured work through carefully prepared progress plans and by getting started with the technical development.

In sprint 2, there was clear progress. Delegation of tasks had been improved and the team experienced a strengthened collaboration where members helped each other to a greater extent. The lack of a macro perspective created challenges with the distribution of work tasks over time, and planning therefore remained a challenge. A greater need was therefore expressed for a better overview of time spent and more systematic follow-up of progress. The program development took great strides in implementing several important functions in this development phase, and the project task began to take shape.

Due to the need for a better overview of time and follow-up, good communication and work effort were maintained in sprint 3. The task was now significantly more defined, and the final phase was therefore efficiently completed. The group nevertheless felt time pressure, as it became clear that several tasks were taking longer than planned. The time pressure meant that not everyone had a full overview of the entire code base, and testing was unfortunately downgraded. Although the collaboration worked well, the group still had potential for improvement related to planning, testing and knowledge sharing.

The project period clearly shows that the team's strengths lie in good collaboration, communication and the willingness to help each other. The team's challenges lie largely in planning and the distribution of work over time. This can be partly explained by limited experience with larger development projects and Scrum as a working method.

If the project had been carried out again, the group would have had a greater focus on realistic time estimates and clearer progress plans from the start. The team would have prioritized core functionality over additional functions, and therefore limited the scope of the task in order to improve the distribution of work over time. At the same time, the team had placed more emphasis on continuous testing and better distribution of knowledge about the code base.

Overall, the project period has been educational, and the group has developed both professionally and collaboratively through the work. The experiences from the retrospectives have contributed to increased awareness of both strengths and areas for improvement, which provides a good foundation for future projects.