NTNDU List Application

Vision

Version 1

Ntndu List Application	Version:	1.0
Main report	Date: 30.04.2021	
NTNDU.MR		

Revision History

Date	Version	Description	Author
29/04/2021	0.1	Main Report Started	All team members
30/04/2021	1.0	Finished Main Report	All team members

Table of Contents

Main report

1. Preface

Software Engineering is a project-oriented study program, where we learn different software engineering models, patterns in software architecture, principles in software design and universal design, prototyping, and user testing. From an academic/educational view the purpose of a project like this is to get hands-on experience with software engineering and teamwork. We also get a lot of experience with the documentation of a project like this and get a hold of the essential components that make both the teamwork and the finished product great and working. We have learned a whole lot about how to work together in a group. To rely on each other for help, as well as taking initiative when you have either knowledge or experience to offer. Throughout the semester we also learned a lot about software architecture and its common patterns, as well as system development methodology, and testing.

2. Introduction

Introduces important elements of the document in five sections: Its - Purpose, Scope, (Definitions, Acronyms and Abbreviations), References, and Overview.

The Purpose section will clarify the process behind the project.

The Scope section explains the scope of the Main Report and how it applies to the NDL.

The (Definitions, Acronyms and Abbreviations) section is a list of not commonly used or new acronyms and their definitions.

The References section lists all references, in addition to authors, dates, and locale.

The overview section explains contents of the remaining pages of the Main Report.

2.1 Purpose

The purpose of the Main Report is to clarify the process behind the project, both in relation to the assignment description and the implementation of the project. In this document you will also find the different definitions, acronyms and abbreviations that we have used throughout our project and in this report. The report will work as a summary of the purpose behind the project, how we intended to solve the project, and how we ultimately did solve it. It will also the theory behind both the finished product, and the theory used to journey on the way to a product that we are all satisfied with.

2.2 Scope

The scope of this Main Report applies to the design, documentation and implementation of the NTNDU List Application (NDL). This document's scope includes an Assignment description where the project task gets explained, theory which is about our theoretical background that the project is based on, what methods and standards were used for how this assignment was solved, how we implemented our vision in the project some further work for our project and a repository where you can find all our work.

2.3   Definitions, Acronyms, and Abbreviations

NDL – NTNDU List Application

UD – Universal Design

CUD – Universal Design at North Carolina State University

DOGA – Design and Architecture Norway

GUI – Graphical User Interface

UI – User Interface

WCAG – Web Content Accessibility Guidelines

2.4   References

[1]   (2007) Todoist [online] London: Doist inc ©.

Available from: https://todoist.com/features [accessed 03.07.2021]

[2]   (2006) Listonic[online] Łódź: Listonic ©.

Available from: https://listonic.com/features [accessed 03.07.2021]

[3]   (2017) Microsoft To Do[online] Washington: © Microsoft 2021.

Available from: https://www.microsoft.com/nb-no/microsoft-365/microsoft-to-do-list-app?rtc=1 [accessed 03.07.2021]

[4]   Howard L.A.O., Emmet N. (2012) Airtable [Online] San Francisco: Airtable ©.

Available from: https://airtable.com/product [accessed 03.07.2021]

[5]   (2017) Microsoft Excel[online] Washington: © Microsoft 2021.

Available from: https://support.microsoft.com/en-us/excel [accessed 03.07.2021]

[6]   (2016) Notion [online] San Francisco: ©2021 Notion Labs, Inc.

Available from: https://www.notion.so/product [accessed 03.07.2021]

[7]   (2007) Things 3[online] Stuttgart: Cultured Code GmbH & Co.

Available from: https://www.culturedcode.com/things/ [accessed 03.07.2021]

[8]   Yevgeniy Z. (2017) SQLite ORM [online]

Available from: https://github.com/fnc12/sqlite_orm [accessed 04.06.2021]

[9]   Arthur S. (2019) FTXUI [online]: Copyright © 2019 Arthur Sonzogni.

Available from: https://github.com/ArthurSonzogni/FTXUI [accessed 03.18.2021]

[10]   Dwayne R.H. (2000) SQLite [online]

         Available from: https://www.sqlite.org/index.html [accessed 03.07.2021]

2.5   Overview

The remaining pages of the Main Report contains the following:

- An assignment description, where the project and the client are presented, as well as the format and nature of the work.
- A theoretical background for the work that the team put in, and how we addressed the different aspects of the project, such as user testing, prototyping, and universal design. As well as how we chose to implement it.
- An overview of what sources, methods, and standards we used, as well as the hardware, and standard software used. An overview of the documentation of the project, including user testing, user case diagrams, sequence diagrams etc.
- How the workload was divided between the team members, and what each member either oversaw or focused on.

> **Commented [JL1]:** Revurder om det bør stå at team member har hatt "ansvar" for noe, kan tolkes som dårlig teamarbeid?

3.   Assignment description

Our task was to create a to-do list application. The aspects of the finished product would be that the user would be able to register new tasks in the application. Each task would hold different relevant information such as task description, priority, category, status, deadline, and start and finish date. In addition, the user would be able to mark tasks as done, change the priority of the task, and reorder them.

The application is a standalone C++ application targeted towards the windows platform, using SQLite as a database and FTXUI as a UI library.

> **Commented [JL2]:** Kanskje legge til noe mer her?

The client is a Computer Consulting Company, and the System Consultant Team has been in contact with Seyed Ali Amirshahi for the client meetings and progress updates. The nature of our work was to be an acting team of systems consultants responsible for designing and implementing the system.

Client Information: Seyed Ali Amishahi
                            s.ali.amirshahi@ntnu.no

4.   Theory

This part of the Main Report will cover the theoretical background that we have based our project on. This will include all standards, methods, and guidelines that we have implemented in our product and what we based the process on. Universal Design will cover the usability of the product and how accommodating it is for users with any

form of impaired functioning. It will also touch base on two main sources of guidelines, WCAG (Web Content Accessibility Guidelines), and CUD (Universal Design at North Carolina State University). Prototyping will cover our approach to prototypes and what whether our prototypes are evolutionary or use and discard. In addition, it will cover our use of low and high-fidelity prototypes. Finally, User Testing will cover how we chose to approach user testing regarding demographic, classical user tests, and the development of testing and feedback throughout the project.

Universal Design

Universal Design is term widely used in IT, design, architecture, society planning, and development of products and services. It is used regarding whether a product, service, building, or institution is adaptable or designed so that any person, regardless of their functional ability, can participate in said activity or operate the product. Web Content Accessibility Guidelines (WCAG) provides the guidelines to make sure you meet their recommendations for usability. The structure offers four main principles.

- Possible to perceive.
- Possible to operate.
- Understandable
- Robust

As well as four main principles, WCAG offers guidelines and success criteria. The guidelines are based on the main principles but offers a more in-depth perspective. There are 12 guidelines which marks what any web content developers or producers should work to meet. Because the guidelines are not testable, WCAG offers success criteria that are. There are success criteria for each of the guidelines. Here, "testable" means you can test if your product complies with the set requirements, agreements, design specifications etc. Success criteria is again divided into three levels of compliance, Level A, Level AA, and Level AAA. What level you need to meet depends on the group or situations a user might find themselves in.

The Project Description specified as one of its basic project elements, that "*the application must be designed according to WCAG 2.1 principle 1 – Perceivable".* The Perceivable principle, according to w3.org covers

> *"Information and user interface components must be*
> *presentable to users in ways they can perceive."*

The principle has different subcategories, or sections, that covers the specifics. As well as the success criteria belonging to each guideline. All success criteria related to audio and video is not considered here as the assignment does not cover anything related to this. The application has no images, nor audio or video implemented.

In relation to the success criterion 1.3.3 Sensory Characteristics and the 1.4.1 Use of Color, the application has a fairly simple design with contrasts between letters and background. Black and white is the main color scheme used, with an inverted scheme where you are currently located as a user in the application. However, other colors are used in the output window for the Tasks, Priority and Completed tab. We considered the 1.4.1 criterion where it says

> *"Color is not used as the only visual means of conveying*
> *information, indicating an action, prompting a response,*
> *or distinguishing a visual element"*

We decided to use a green color for the output window when what you have executed in the program is correct, as well as a red color when what you have executed is incorrect, but we also included the error message with clear instructions on what you either must change or can leave as it is. That way the message is conveyed regardless of the user's inability to see color. Both the 1.3.3 criterion and the 1.4.1 criterion fulfill a Level A standard.

Success criterions 1.4.4 Resize text says.

> *"Except for captions and images of text, text can be resized*
> *without assistive technology up to 200 percent without loss*
> *of content or functionality."*

In our application this is achievable through 'Ctrl +' or 'Ctrl -', or Ctrl + scrolling back and forth. Since our application meets this criterion as well, it fulfills a Level AA standard here.

In relation to success criteria like 1.4.8 Visual Presentation, 1.4.12 Text Spacing and 1.4.13 Content on Hover or Focus, we decided not to focus on them as they would require plenty more working hours and aimed towards a Level AA or Level AAA in most cases. In general, we decided to focus on the Level A goals as well as some of the Level AA ones.

The course content also mentions DOGA, Design and Architecture Norway, and CUD, Universal Design at North Carolina State University, which has seven main principles regarding UD. These are

1. Simple and intuitive to use.
2. Understandable information.
3. Tolerance of errors.
4. Equal opportunities for everyone.
5. Flexible to use.
6. Make physical effort.
7. Size and space for access and use.

We feel like a team that we have fulfilled almost every one of these principles. Although you could argue to what degree. Every application can almost always be more user-friendly. Still, we feel like the application is both *Simple and intuitive to use*. This also goes for *Equal opportunities for everyone* since there are no technical requirements to understand a software of this type. The only requirement is basic keyboard skills. This is also the case when it comes to *Flexible to use* and *Make physical effort.*

Understandable information is also covered. Both error messages that appears and messages that confirm a successfully implemented change in one of the tasks, conveys the necessary information to the user in an effective manner. NTNDU is also *Tolerant of errors*, because it is very easy to fix anything you have accidentally missed or done incorrectly. The last principal *Size and space for access and use* is also met to some degree. As mentioned earlier we do have a zoom function that enables any user with impaired vision to use the application more effectively. Still, we do not have a lot of other features that allow the user to customize their experience.

As mentioned in the introduction to Universal Design further up in the report, the Project Description specifies that the application must be designed according to WCAG 2.1 principle 1 – Perceivable. All audio and video success criteria has to be overlooked, as the application does not offer either audio or video options. Regarding the other success criteria, we feel that as a team we meet to the best of our abilities. It is user-friendly, with contrasts and highlighting. offers zooming and does not require any skill other than basic keyboard skills, although a Level AAA standard has not been met on all accounts, we still feel it complies to the 2.1 Principle 1 – Perceivable.

Prototyping

Our approach to prototyping was severely influenced by the theory presented during the lectures, and the principles of Ernest Hemingway that "the first draft of everything is shit." Therefore, we decided to mainly use wireframes in

the start of the project. Everyone in the team played around with different versions and designed different layouts and together we came up with something we wished final product would look like.

After that we decided on a specific library to use for the project, and then after thoroughly testing out the library, we designed a wireframe based on that. In general, we also started with the most basic functions of our program for us to be 100% sure it worked, before moving on to more detail-oriented work. This way we didn't have to go back and fix the fundamentals but could rather focus on the finer things of the program, when we moved towards the end of the project.

One can say we used a horizontal approach in the beginning of the project, where we tested many features, but not with a lot of depth. Then, later on, we used a vertical approach, where we selected fewer features but tested them in depth. We also used the vertical approach when we wanted to answer specific questions about the application, e.g., if an error message appears after you have already created a task and given it a description but failed to give it a finish date.

Though prototype development can be divided into two different categories; use and discard prototypes, and evolutionary prototypes, we did use the latter one of the two. Evolutionary prototypes as said in the course content is.

> *"...where the prototype is gradually developed during different iterations. The feedback provided at each iteration will then have an important effect on the next iteration. In evolutionary prototypes, the last version of the prototype is the final product."*

The earlier prototypes, the first wireframes, had very low fidelity. As time went by and we gained feedback, and our skills progressed, we reached higher fidelity. The further we got in the project, the higher fidelity we had on each prototype and the more we brought with us into the designing of the next one. A contributing factor to the higher fidelity was user testing, and the feedback from there. That was very helpful though the whole process.

User testing

In our approach to user testing, we wanted to get a diverse demographic to test our product, both in age, gender, and occupation. As a team we feel like we achieved this, even though the situation we are in with travel restrictions and social distancing etc. made it a bit harder than it initially had to be.

When referring to the lectures again, Laura Arlov said in the GUI Guide II from 1999 that

> *"User testing is the closest thing to a magic formula you can have to improve usability. It is the fastest way I know to acquire increased competence as a designer."*

We used classical user tests, where the main purpose is to increase the quality of use. As said in the course content, these kinds of tests are qualitative in nature, where instead of a measurable outcome the results consist of impressions, opinions, and views. When planning the tests, we wanted to set clear simple goals, so that they would not take too long, and would provide an easy and readable result. Goals like "see if the user knows how to use the side menu", "how to create a task", "how to confirm a made task", "how to change the priority of a task" etc., are what we qualified as good goals.

When we executed the user testing, we did several of them before the first iteration within the wireframe. After doing that, we realized that the first wireframe did not fit with a terminal-based program, so we discarded the first initial wireframe and started over. The first batch of user tests was therefore naturally not that much of use for us. The user tests done before the second iteration, however, worked very well. We got a lot of great feedback. The

documentation from this can be found in our Git repo. Here we did both a user test with the wireframe, as well as the minimal viable prototype.

5. How the assignment was solved

The assignment was solved while working closest to the Kanban method where we used the Gitlab Issue board to have an overview over the tasks we are working on. The standards we worked with were mainly Don Norman's Principles of Interaction design and WCAG (Web Content Accessibility Guidelines.

When it comes to use of literature and internet the main thing would be Don Normans Principles of Interaction Design, Wikipedia and PROG1004 Software Engineering course main documentation file. For our hardware everyone except one person used standard windows 10 machines. One person used mac and a virtual windows machine.

The software we used was Balsamiq Wireframes, which is a graphical user interface where you can build and test a simple application, to make a really early prototype and try to figure out our programs GUI and the functions we wanted, we used Visual Studio Code, which is a source-code editor by Microsoft, to program C++ as that was the program the group was most used to, we used Gitlab as our repository to keep track of all our files and to be able to share and edit our program in an efficient way and we used Discord and Teams for communication in the group. The work was divided based on interest of group members. If a task seemed interesting to a particular team member or a task fit that specific team members skill, we tried to make sure that team member got to do the task.

We did not encounter any major problems in the ways we worked. The biggest problem was that our first wireframe was too GUI based for a terminal-based program, so we had to make a new wireframe from scratch.

Documentation:

- Collaboration agreement (Gitlab)
- Vision document (Gitlab)
- User Testing MVP v.2 (Gitlab)
- User Testing Wireframe v.2 (Gitlab)
- NTNDU domain model (Gitlab)
- NTNDU Class Diagram (Gitlab)

6. Implementation of the project

Our vision for the application was quite different than what it would eventually become, however what has not changed is the objective and goals we had set for the project. An example of one of those goals, would be the UI, albeit miniscule in comparison to the necessity of the functions, still plays a big part in the project.

The application itself became somewhat different than what was originally planned, what we managed to make was something much different than what was described in the vision document. Technically we managed every aspect of what we wanted with few exceptions.

We managed to make the application communicate with a .sql database as planned. And we implemented everything we described in the 5.2 summary of capabilities from the vision document. Effectively fulfilling all the features planned for our client.

Overview over features and benefits:

- Register new tasks.
- Add additional information like task description, priority, category, status, deadline, and start and finish date.
- Mark tasks as completed.
- Edit any information about the task.
- Delete any tasks.
- Reorder tasks
- Sort and filter tasks by description, priority, category, status, deadline, and start and finish date.

Of course, in the case of our application, the task description, and the name of the task are the same. We decided to do it this way because we felt it was not needed to both have a description and a name, as the character limit is set to 30, which should suffice. We also chose not to implement a finish date and a deadline, but rather only a deadline. This is because we did not deem it necessary to have both. The reason for that is there is a full overview over all completed tasks, and the status clearly shows the progress. We therefore decided to overlook that requirement, as we think a start date and a deadline will serve the same purpose as all three would together. And reordering and sorting would be equivalent.

We managed to complete everything according to the result goals we set for our project as well (3.2 Result goals Vision document)

- Create a functional to-do list application of high quality which fulfills all the features our client wants.
- Ensuring that the application is easy to use and syncs seamlessly between devices.
- Complete vision report, wireframe prototype and domain model by 07.03
- Finish MVP, GitLab and WIKI by 11.04
- Finish application, all documentation and final report by 30.04

As compared to what was depicted on the Gantt diagram.

The risk analysis was deemed to be correct, however the risk in of itself was not consequential, the reason being that the project is a school project.

We were unfortunately unable to add more color to the functions of the application, in addition to running early user tests before the MVP. We also did not have enough time to implement every change we wanted after the user testing as we were pressed for time. We were therefore not capable of such a feat within the given time, in addition to other projects from different classes. We could have done things differently, instead of using time in the beginning to create lots of different wireframes based on GUI we could have started a little bit earlier in order to allot more time into creating a finished product.

When the planning did not match up with the members schedules, we had good vibes and filled in for each other, it just worked out. We mostly communicated through Discord, and we used Teams for sharing files. Thereafter we committed and pushed the changes to Gitlab.

After thoroughly going through trying to convert our wiki to pdf a dozen times, we could not find a viable solution to embedding multiple pdf documents to the main wiki pdf. Therefore, we went with using links to the git-download server instead.

Effort and the result went accordingly as what was planned, the teamwork was overall efficient and well distributed. All the members performed effectively, and we were able to deliver a final product in accordance with the specifications provided and what was planned. In this project we were most pleased with how our project turned out, we think we had creative and unique solution to the project assignment. There are however some things we could have done differently like used less time on Wireframe in the beginning and started on our documentation earlier. We were extremely satisfied with the UI of the product seeing as it was an experimental library (FTXUI), and we had to teach it to ourselves from scratch. The time and effort allotted into the project was carefully and meticulously planned and executed to an acceptable degree. This is especially true when time was allotted to other group members when unexpected health complications impaired a team member.

7.   Further work

Product perspective

The product perspective is a terminal-based to-do list application that can add a simple task, edit a task, delete a task and sort a task. The product is meant for primarily businesses to keep track of tasks but can also be used by private persons.

Overview over features and benefits:

- Register new tasks.
- Add additional information like task description, priority, category, status, deadline, and start and finish date.
- Mark tasks as completed.
- Edit any information about the task.
- Delete any tasks.
- Reorder tasks
- Sort and filter tasks by description, priority, category, status, deadline, and start and finish date.

Assumptions and dependencies

- We assume you are running the latest version of Windows 10.
- We assume that you have an archiver installed (like WinRAR).

It saves changes to the database. Thereafter it can load the changes.

8.   Repository

https://gitlab.stud.iie.ntnu.no/anp/prog1004-gruppe-25