Andreas Richardsen & Zahro-Madalina Khaji

Abstract

[Draw your reader in with an engaging abstract. It is typically a short summary of the document.   
When you’re ready to add your content, just click here and start typing.]

[Document title]

[Document subtitle]

Contents

[Introduction 2](#_Toc534365886)

[Problem statement 2](#_Toc534365887)

[Group contract 2](#_Toc534365888)

[Analysis 2](#_Toc534365889)

[Requirements 2](#_Toc534365890)

[Functional Brief use cases 2](#_Toc534365891)

[Non-functional 3](#_Toc534365892)

[Domain model 4](#_Toc534365893)

[Design 4](#_Toc534365894)

[System architecture 4](#_Toc534365895)

[User interface 4](#_Toc534365896)

[Design class diagram 4](#_Toc534365897)

[Relational model 4](#_Toc534365898)

[Implementation 4](#_Toc534365899)

[Choices of technologies 4](#_Toc534365900)

[Back-end 4](#_Toc534365901)

[Front-end 4](#_Toc534365902)

[Code 4](#_Toc534365903)

[Testing 4](#_Toc534365904)

[Conclusion 4](#_Toc534365905)

[References 4](#_Toc534365906)

# Introduction

In this report we will introduce a problem at the start, analyse the problem, show the way we decided to design it, how we decided to implement the system, how we performed our testing and at last we will say our conclusion where we will also answer the questions that was asked in the problem statement.

The system we decided to create was a Personal Task Manager to manage your tasks, ranging from daily, household, work, etc.

## Problem statement

We believe the old TO DO list method of organising to solve tasks has several deficiencies and it might be improved.

To create a TO DO list the old way one requires access to paper and a writing instrument. In our modern age, access to the internet is very common and technology is very wide spread. Therefore, a digital TO DO list could be a better option with possibly more functionality.

Here are some as the aspects that is lacks: not easily accessible. E.g., when you go grocery shopping and you forget your list at home. Therefore, an improved list should be available to access anywhere, anytime with just an internet connection.

Another issue is that of security. Suppose you want to keep you TO DO list private – a list written on a plain piece of paper might be seen by anyone around you who might be snooping or can be lost and who knows who might find it. This shows the importance of keeping your data safe.

Writing the tasks on a list on paper means that it is hard to edit or rearrange them if the need arises, and takes unnecessary space. Important tasks for the user might also not be properly evident so they might be overlooked. Therefore, a method of organising, categorising and highlighting list tasks is necessary.

* What are some ways in which we can improve the old-fashioned TO DO list?
  + How to present the tasks in a more organised manner?
  + How to highlight the user’s most important tasks?
  + What are some ways to make sure that our list is accessible to the user anytime and anywhere with an internet connection?
  + Which are some ways to store our data securely?

# Analysis

## Idea generation

## System vision

What we intend to do

Scope

Purpose

## Requirements

We used brief use cases only because we deemed none of our use cases to be complex enough for a fully dressed use case.

### Functional Brief use cases

* CRUD account
  + Create account
    - User navigates to the login page, selects the create account option, the user inputs the required information for creating the account, the system creates the users account and the user is redirected to the log in page.
  + Log in
    - The user inputs their credentials and the system redirects them to their dashboard.
  + Update account
    - User selects the account settings option, and then he changes the information he wishes to change and saves it, the system records the changes.
  + Delete account
    - The user selects the account settings option, then selects Delete account and is prompted with a popup to confirm that he wants to delete his account; the system removes the users account information and redirects the user to the login page.
* CRUD for list
  + Create list
    - The user presses the create list button and inputs the name of the list, the system creates and adds the new list to the dashboard.
  + Update list
    - The user selects the list settings, he is redirected to the view list page where he can change the name and/or add tasks.
  + Delete list
    - The user selects the list settings, he is redirected to the view list page then he selects the delete list button, the system removes the list from the records and the user is redirected to the dashboard.
* CRUD for task
  + Create task
    - The user presses the add task button on the view list page, then inserts a name for the task and presses save, the task is saved to the system and is displayed in the list.
  + Create subtask
    - The user presses the add subtask button on a selected task, then inputs the name for the subtask and presses save, the system records the task and the subtask is displayed on the task.
  + Update task
    - The user selects the task on the view list page, then updates the desired information and saves the task, the system records the task and the task is updated.
  + Delete task
    - The user selects the task on the view list page, then presses the delete task where he is prompted with a popup that ask if he is sure, when the user presses yes the task is deleted from the records and the view is refreshed.

### Use case diagram

### Prioritisation of use cases

1. Create account

### Non-functional

System must be:

* User friendly
* Accessible
* Fast
* Secure

### Domain model

# Design

## System architecture

## User interface

## Design class diagram

## Relational model

# Implementation

## Choices of technologies

### Back-end

Wcf vs ASP.NET Web API

### Front-end

Angular vs react vs vue

## Code

# Testing

# Conclusion

# References