**Final project documentation**

Christopher Hardamek  
CS1410

Due Date04.23.22

Inhaltsverzeichnis

[Introduction 2](#_Toc101549269)

[Tasks 2](#_Toc101549270)

[Task 1: Program does something fun or useful 2](#_Toc101549271)

[Task 2: Program handles invalid input 2](#_Toc101549272)

[Task 3: Logic and IO cleanly separated 3](#_Toc101549273)

[Task 4: UI layer calls down into logic layer for processing 4](#_Toc101549274)

[Task 5: Read and/or write data from one or more: files, database, network 6](#_Toc101549275)

[Task 6: Use interfaces to define access to external services (e.g., your storage service, database, network, etc.) 6](#_Toc101549276)

[Task 7: Use bogus/test implementations of those interfaces in your unit tests (e.g., do \_not\_ actually write to the filesystem or call the network from a unit test). 6](#_Toc101549277)

[Retrospective 6](#_Toc101549278)

[Retrospective: What you learned in the process of completing this project 6](#_Toc101549279)

[Retrospective: Share which principles/patterns/technologies you enjoyed learning about and why you feel they're valuable/beneficial. 6](#_Toc101549280)

# 

# Introduction

This documentary is about my final project in CS1410 Object-oriented Programming. The target of this final project is to know how to program and how to handle different situations. Here is my [Repository](https://github.com/christopherhardamek/Banksystem_final).

# Tasks

## Task 1: Program does something fun or useful

This program can create accounts. The user can create a saving account, make a deposit, withdraw, and get credit from the bank in the accounts.

## Task 2: Program handles invalid input

On the Index page, there are some input fields. One of the Input fields is Name, it isn’t possible to create an account with just numbers. This counts for Lastname and name.

The other inputs are for the day, month, and year. It isn’t possible to create a name as a date. And this program will check if the date exists. The Account number must be between 8 and 12 Digits, otherwise, it shows an error.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Screenshot 1

On screenshot 1, you can see there is an Error message because the date is not valid.

## 

## Task 3: Logic and IO cleanly separated

On Screenshot, there is 3 Folder. One folder Bank\_lib. In this folder is the whole program. The folder BankBlazor, in this folder, is the Web application. The last folder is Bank\_test. There is all the test inside.

In the Finalproject.sln there is the reference between the folder. It is necessary to have this because without the reference is not possible to use all folder and the logic.

Screenshot

Screenshot

## Ein Bild, das Text enthält. Automatisch generierte BeschreibungTask 4: UI layer calls down into logic layer for processing

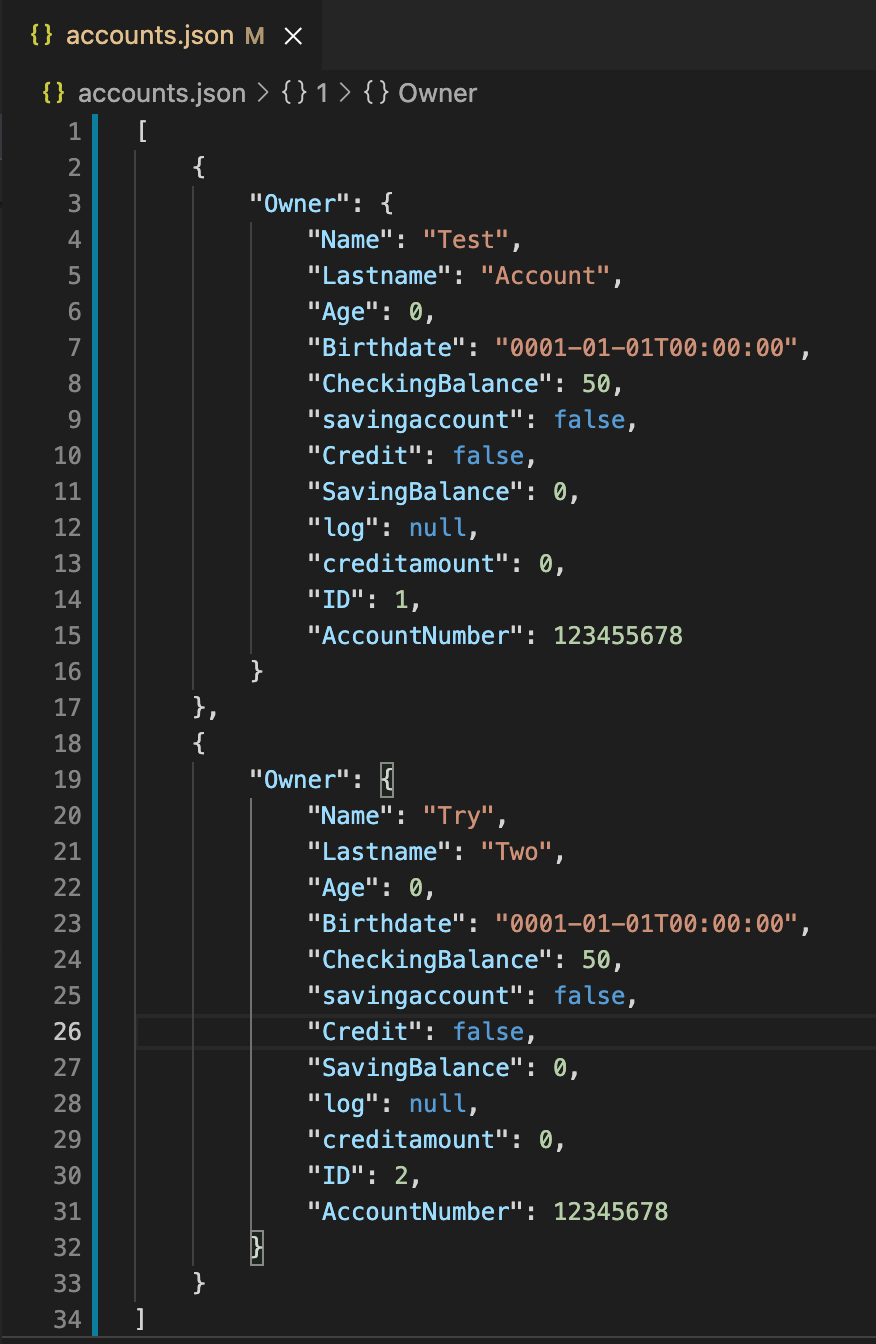
In screenshot 3 there is the index page. On line 96 is called the function GetAccountNumber(Accountnumber). In screenshot 4 there is the function of screenshot 3Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Screenshot 4

## Task 5: Read and/or write data from one or more: file, database, network

Screenshot 5

In my project, I work with 2 files. The first file is responsible for the and the other one is responsible for the logs.

In screenshot 5 there are two accounts. One of them is Test Account and the other one is Try Two.

They both have a different customerID. Test Account has ID 1 and “Try two” has ID 2. This number is unique.

Test Account has a balance of $50 and the account number is 123455678. The Try Two account has a balance of $50 and the account number is 12345678.

On the other hand, there is more information about the accounts. For example, they were both born on the 1st of January in the year 0001. Both accounts don’t have a Credit. The saving balance has a balance of $0 in both accounts. The Age function is not yet implemented.

## Task 6: Use interfaces to define access to external services (e.g., your storage service, database, network, etc.)

Fixen

## Task 7: Use bogus/test implementations of those interfaces in your unit tests (e.g., do \_not\_ actually write to the filesystem or call the network from a unit test).

# Retrospective

## Retrospective: What you learned in the process of completing this project

I learned much about programming. I learned how to work with files, how to work with exceptions, Blazor, debugging, Query expression, and Test-driven development. During I did my final project, it was sometimes hard because I stuck in problems that I had never.

I think that is normal for a final project respectively.

## Retrospective: Share which principles/patterns/technologies you enjoyed learning about and why you feel they're valuable/beneficial.

I enjoyed it. I think some of the functions I will need in the future. I enjoyed the time working with JSON. In this case, I learned the format of JSON. In the IT world, there are often files in JSON. I think JSON is important a lot of config files are in JSON format and that is the first step of the introduction to JSON. In this project I did, there is a lot of option that can be improved, for example. With a log-in system, or transfer money between customers. But the basic is done. What else can be improved is the data get saved in the database. In most cases is the pattern the same.

## Retrospective: Discuss things you'll do differently next project.

In the next project, I will inform more about which options I have to implement some functions instead of using just the basics. During this time, I will set some milestones. A mile is for me, I will set a milestone for example, until the weekend I want to have a log-in system, and the next weekend I want to have a User Interface