HACETTEPE UNIVERSITY

Computer Engineering
BBM487 Software Engineering Laboratory

Library Book Loan System
Architecture Notebook

GÜLŞAH ERDOĞAN	**21327953**
GÜLENDAM BUKET GÜNDÜZ	**21328026**
İREM ÖZEN	**21328337**

Book Loan System
Architecture Notebook

1. Purpose

The main purpose of this document is to make it easier for third parties to think about not using or examining our system, by giving an overview of the system's architectural structures, component breakdowns and modeling of these connections.

This document explains the design of the system, its progress, its constraints, and its more extensive parts that shape its implementation. We make the architectural features of the system more comprehensible with diagrams and paragraphs.

In short we are talking about the structure of this document system. So what we call architectural design is control and communication of the main system components that make up the system.

2. Architectural goals and philosophy

The purpose of this system is to meet the library use needs, so the requirements and complexity of this system are generally the same everywhere.

The goal of this program is to develop a convenient, fast, reliable and practical program. Some ways to help you do this include: The database can be updated continuously, We can easily keep information in the database (about books and people), many people can process it. So the database is continuously updated. It is not expected that the system will provide load performance or be in contact with other systems.

We have designed all the requirements of the system and the architecture of this system in detail according to the requirements, so major changes can push the flexibility of our architecture, but if it needs to be maintained in general, our architecture can easily be changed.

This system is designed for use in Windows and Windows operating systems and should be able to use system features so that it can use hardware and system programs.

Our system does not have to work under unusual conditions in a fully efficient manner.

3. Assumptions and dependencies

The assumptions are;

- The system should be easy to use for users.
- The system must be working 24 hours a day, 7 days a week.
- The system must have a fast access to the database.
- Coding must be error free.
- The storage capacity of the system must be high.
- The information of all users, books and libraries must be stored in database that is accessible by the desktop application
- The system should provide search facility and support quick transactions

The dependencies are;

- The specific hardware and software due to which the product will be run

- All the processes become generated according to user requests via user interfaces, so the system wants correct and consistent requests from the user, if there is not system might not be enough useful as expected and generated for.

- On the basis of listing requirements and specification the project will be developed and run

- The system should have the general report stored

- The information of all users must be stored in a database and input file that is accessible by the Library System

4. Architecturally significant requirements

Requirement

Software Configuration:

This system requires Windows operating system and enough memory space,CPU power etc. hardware necessarities if they are not be met the program execution might not be done or be wrong.This software package id developed using c# as front end which is supported by Microsoft system.Microsoft SQL Server and data file end as the back end to the database.

Operating System:Windows 7,Windows XP

Language:C#(Microsoft Visual Studio)

Database: MS Access server(back end), data file

Hardware Configuration;

Hard disk:40GB Ram:256 Mb or mores

Data Requirement

The inputs consist of the query to the database and the output consists of the solutions for query. The output also includes the user receiving the details of their accounts. In this project the inputs will be the queries as fired by the users like create an account selecting books and putting into account.

Security Requirements

System will use secured database. System will have different types of users and every user has access constraints. No one should be able to hack users ‘password '. Normal users can just read information but they cannot edit or modify anything except their personal and some other information .

Performance Requirements

In the proposed system to be developed, the database is expected to functionally fulfill all the conditions determined by the university because our system will be used as the head performance of Hacettepe University which interact with the university staff and students. System performance must be fast and accurate

Security Requirement

The database must acquire a database backup so that the database does not get lost after a possible problem (possibly due to a virus or operating system crash).

5. Decisions, constraints, and justifications

- For memory management we doubt about using database or not, in conclusion we decided to use.
- We used the MCV model to separate the view from the application logic and easy maintenance on the system.

- This system must execute just one process at a time; because system didn’t designed for concurrent usage.
- The system must be suitable for rules that we described before in part 4, they are generally about hardware and usage requirements of the system like response time.
- The system must be flexible and updatable as we cannot easily maintain it.
- The system has an operating system restriction; because all the execution tests will be done according to Windows OS.
- This library system work into desktop applications.

6. Architectural Mechanisms

Architectural Mechanisms are common solutions to common problems that can be used during development to minimize complexity. An architectural mechanism can have three states: analysis, design, and nmplementation. These categories reflect the maturity of the mechanism's description. The state changes as successive levels of detail are uncovered when you refine architecturally significant requirements into working software. The categories are summarized in the table that follows.

States of an Architectural Mechanism

State	**Description**
Analysis	A conceptual solution to a common technical problem. For example, persistence is an abstract solution to the common requirement to store data. The purpose of this category is simply to identify the need to design and implement an architectural mechanism, and to capture basic attributes for that mechanism.
Design	Refining an analysis mechanism into a concrete technology (for example, an RDBMS). The purpose of this category is to guide precise product or technology selection.
Implementation	A further refinement from a design mechanism into a specification for the software. This can be presented as a design pattern or example code.

Architectural mechanisms represent key aspects of the technical solution that need to be standardized across the project. Everyone on the project should handle these concepts in the same way, and re-use the same mechanisms in their code.

7. Key abstractions

Users:

- Member: Who can take and return a book, make reservation for a book or pay fine for late return.
- Librarian: Who can manages books and members

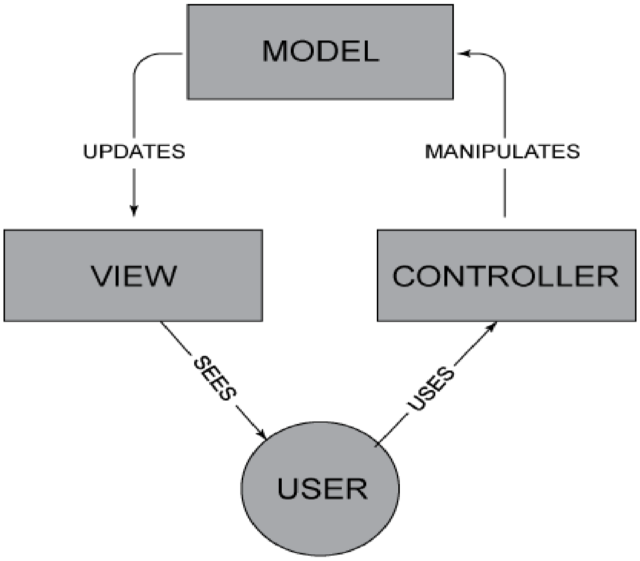
Interfaces: a point of interaction between our system and users;
Database: Where books and user information are kept

8. Layers or architectural framework

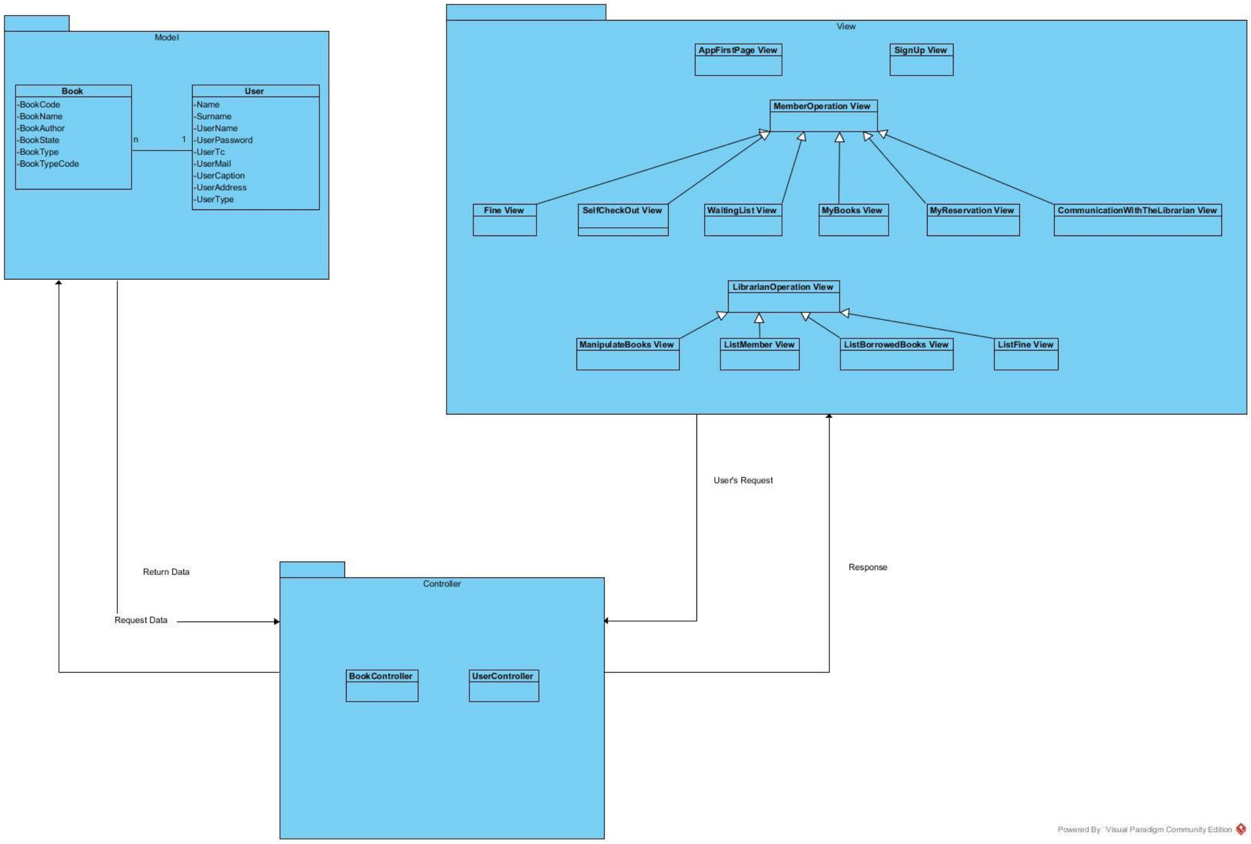
We use Model-View-Controller (MVC) pattern to describe our Book Loan System. MVC is a software architectural pattern for implementing user interfaces on computers. It divides a given application into three interconnected parts in order to separate internal representations of information from the ways that information is presented to and accepted from the user.

Model: It represents the data used in application.
View: It is what users see during use the application.
Controller: It provide connection with Model and View.

It is package diagram of our Book Loan System with MVC architecture pattern.

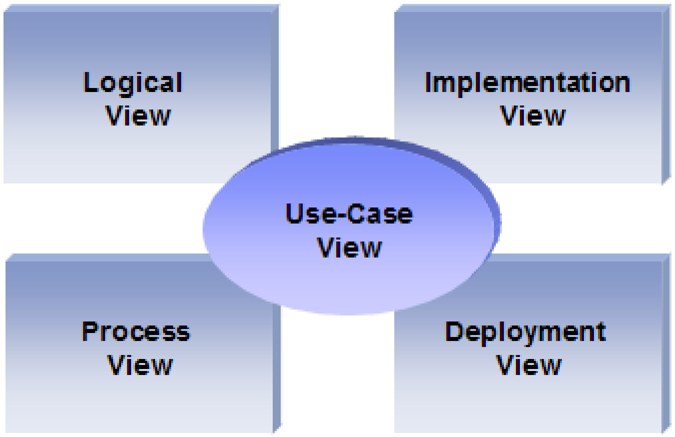


Package diagram of our system is as following:

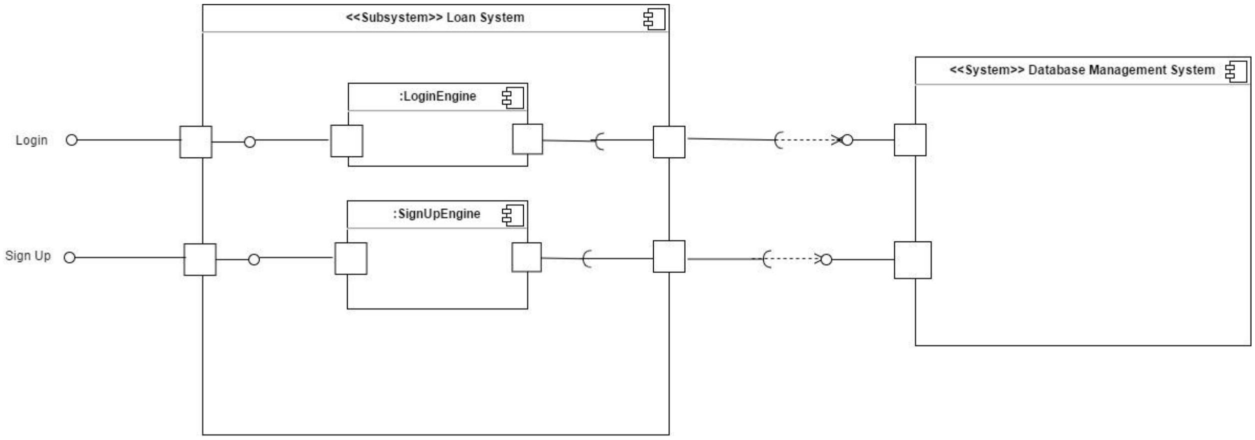


9. Architectural views

The “4+1” View Model allows various stakeholders to find what they need in the software architecture. The four views of the model are logical, development, process and physical view.



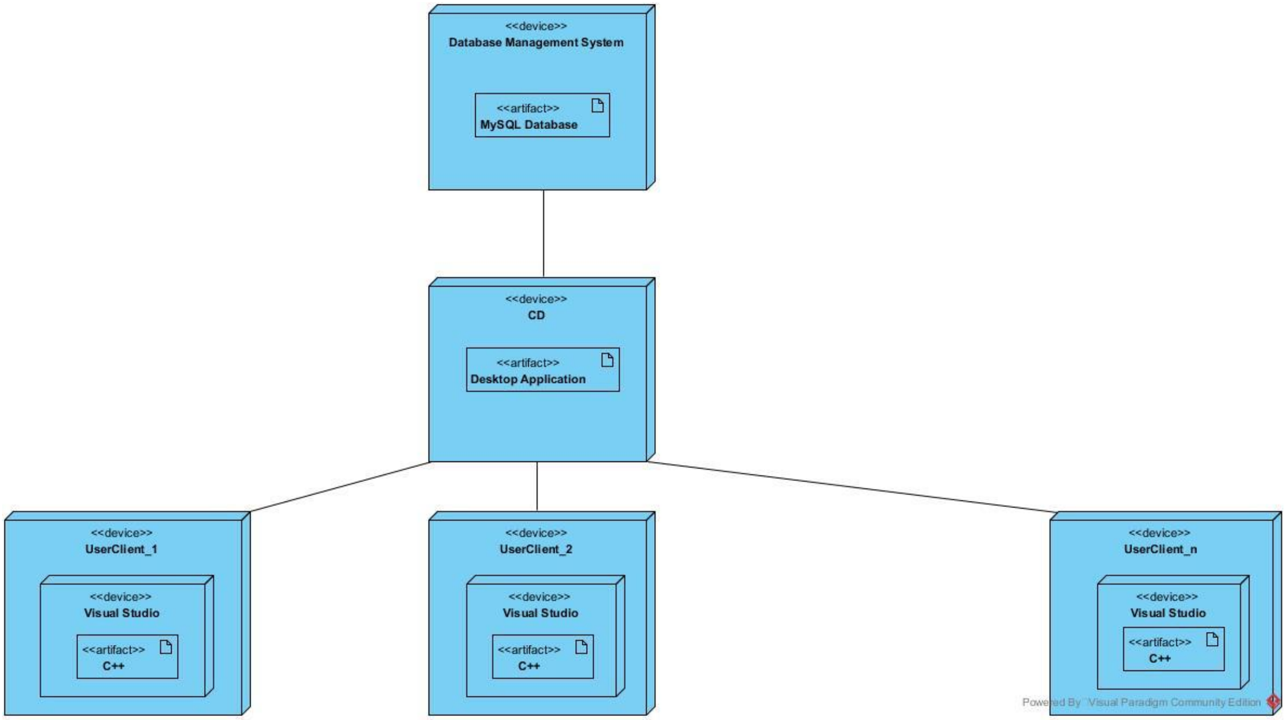
In this delivery, we implement out first demo for two use cases such as Login and Sign up. Their **component diagram** is as following:


<<Subsystem>> Loan System
:LoginEngine
:SignUpEngine
<<System>> Database Management System
Login
Sign Up

When user login, first user must enter username and password, system check correctness of username and password from database. If correct, user enter the system.

When user signs up, user must fill the Sign Up Form. System create a user depend on the information from the form.

Deployment diagram of our system is as following:

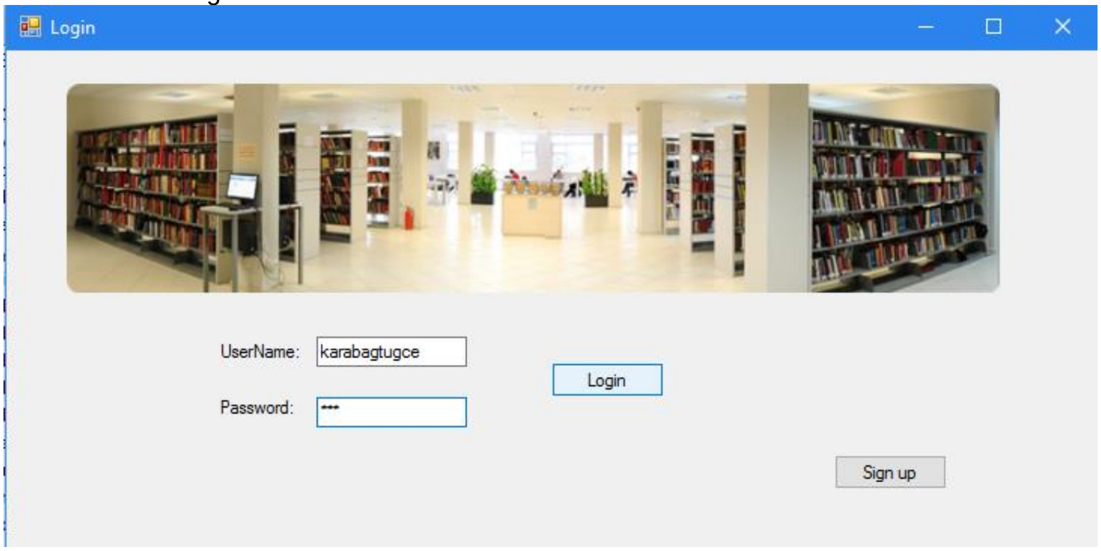


Recommended views

- **Logical:** The logical view is concerned with the functionality that the system provides to end-users. It is considerably simplified to take into account only the items that are architecturally significant.
- **Operational:** It focuses on how the system is built.
- **Use case:** It is necessity when using 4+1 View Model.

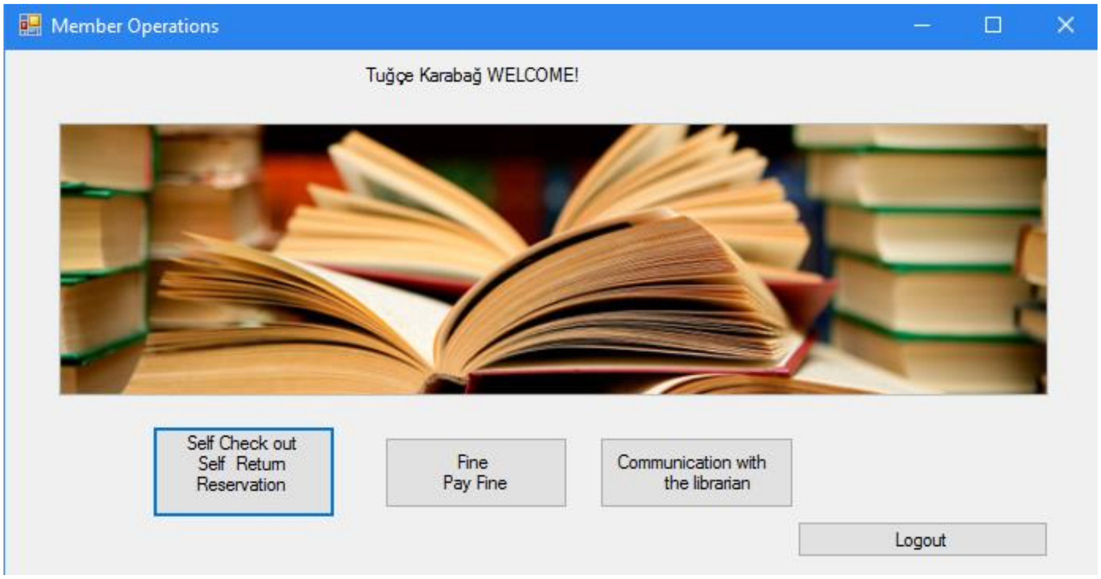
DEMO INTERFACES

When member logins:



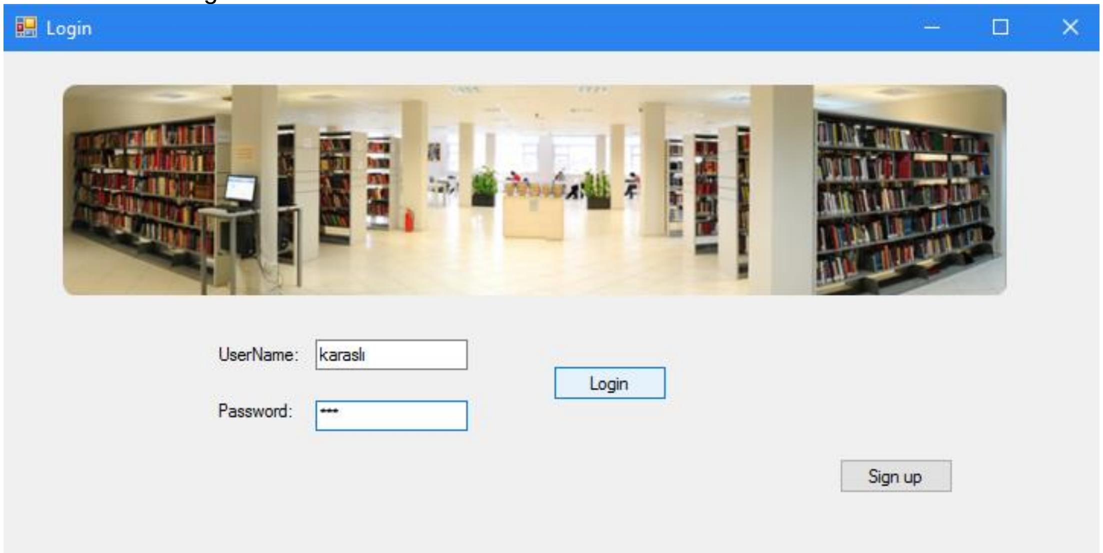
UserName: karabagtugce

Password: ***



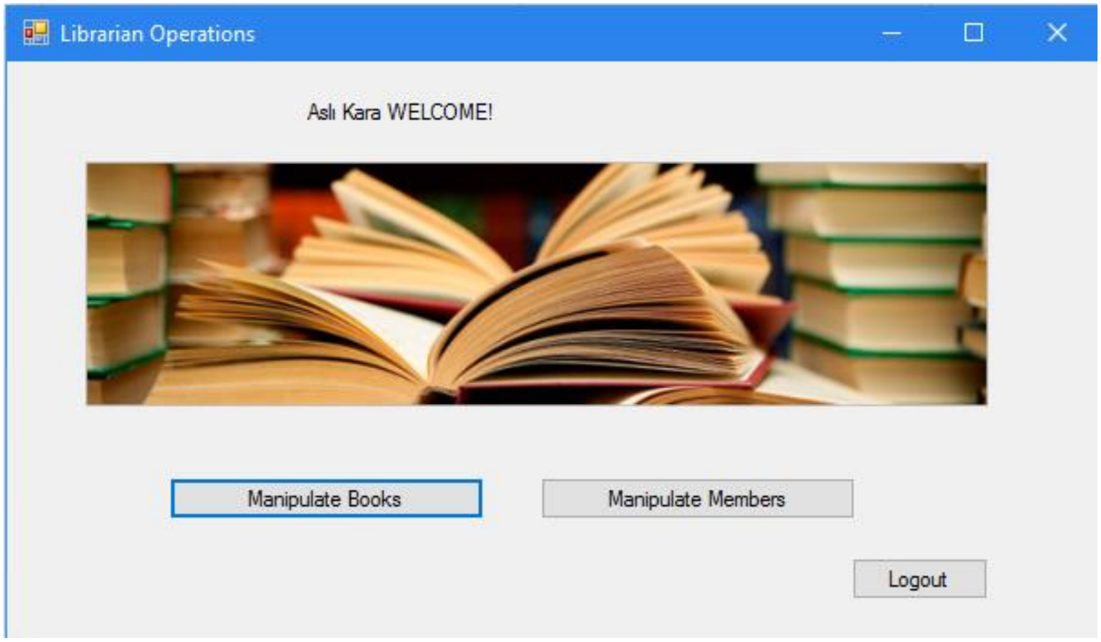
Tuğçe Karabağ WELCOME!

When librarian login:



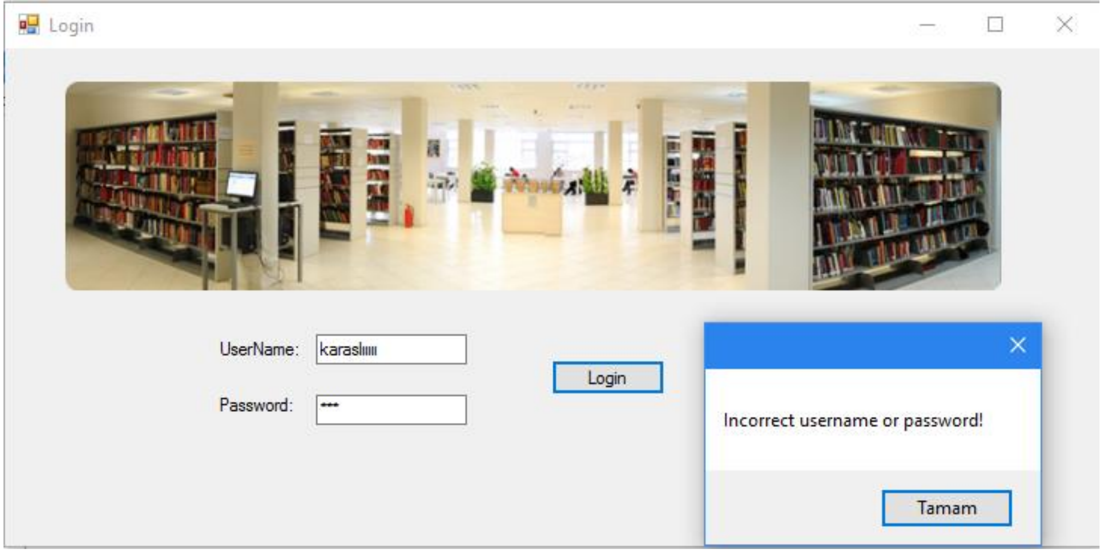
UserName: karaslı

Password: ***



Aslı Kara WELCOME!

If username or password is entered wrong:

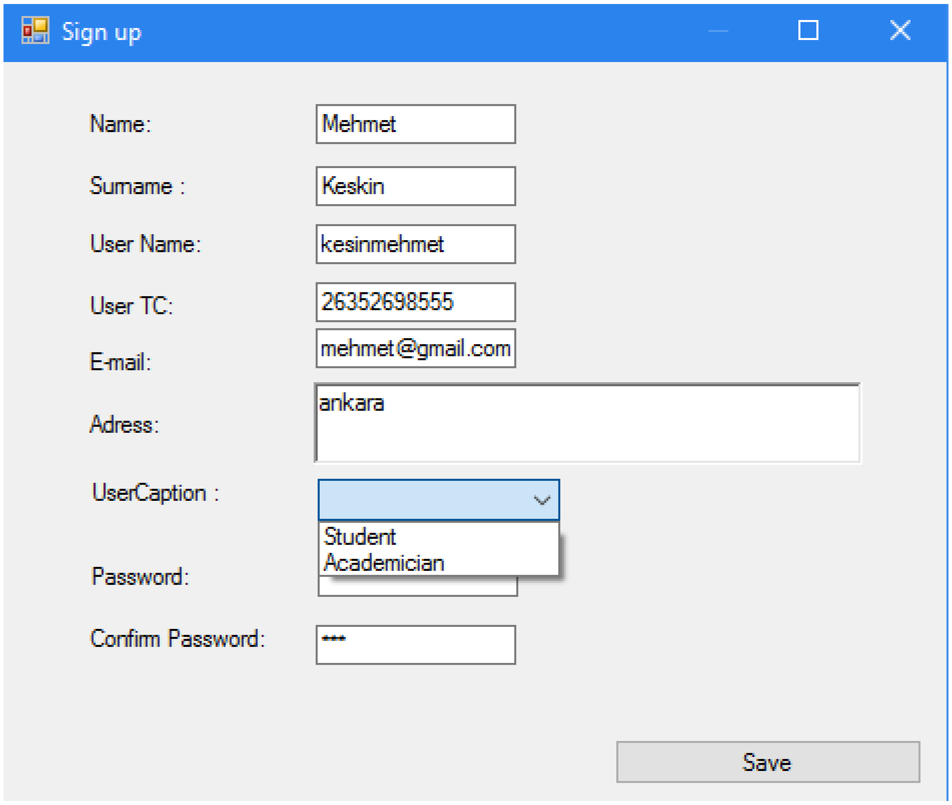


UserName: karaslıııııı

Password: ***

Incorrect username or password!

If user signs up:



Name: Mehmet

Surname : Keskin

User Name: kesinmehmet

User TC: 26352698555

E-mail: mehmet@gmail.com

Adress: ankara

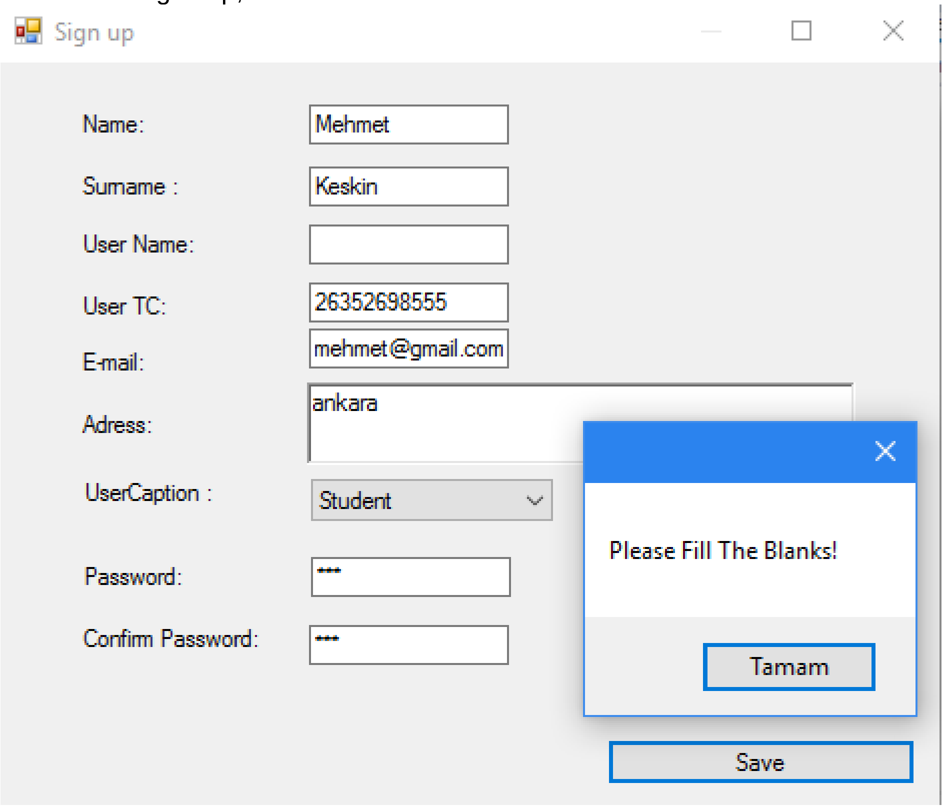
UserCaption :

- Student
- Academician

Password:

Confirm Password: ***

When user signs up, if leave a blank in the form:



Name: Mehmet

Surname : Keskin

User Name:

User TC: 26352698555

E-mail: mehmet@gmail.com

Adress: ankara

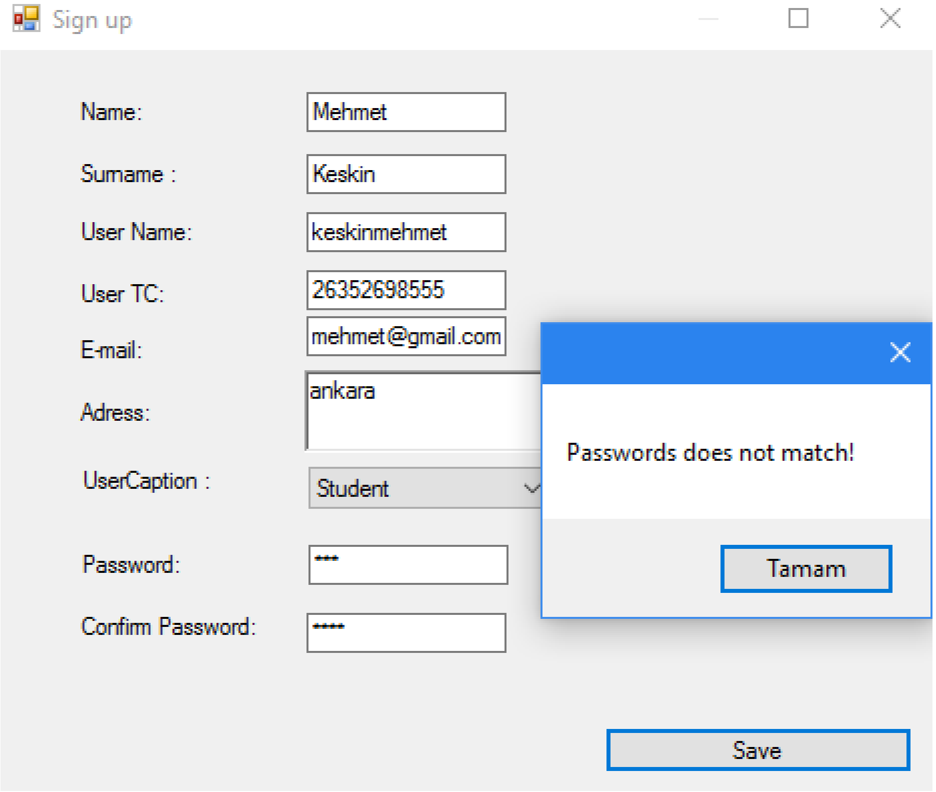
UserCaption : Student

Password: ***

Confirm Password: ***

Please Fill The Blanks!

If passwords do not match:



Name: Mehmet

Surname : Keskin

User Name: keskinmehmet

User TC: 26352698555

E-mail: mehmet@gmail.com

Adress: ankara

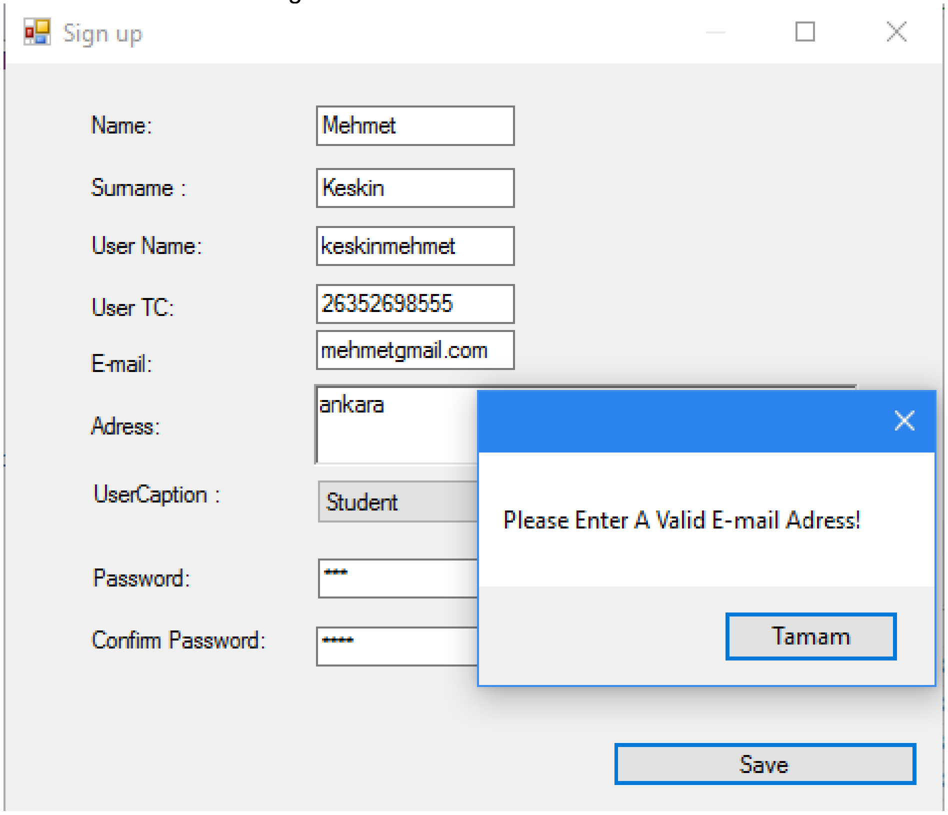
UserCaption : Student

Password: ***

Confirm Password: ****

Passwords does not match!

If e-mail is entered wrong:



Name: Mehmet

Surname : Keskin

User Name: keskinmehmet

User TC: 26352698555

E-mail: mehmetgmail.com

Adress: ankara

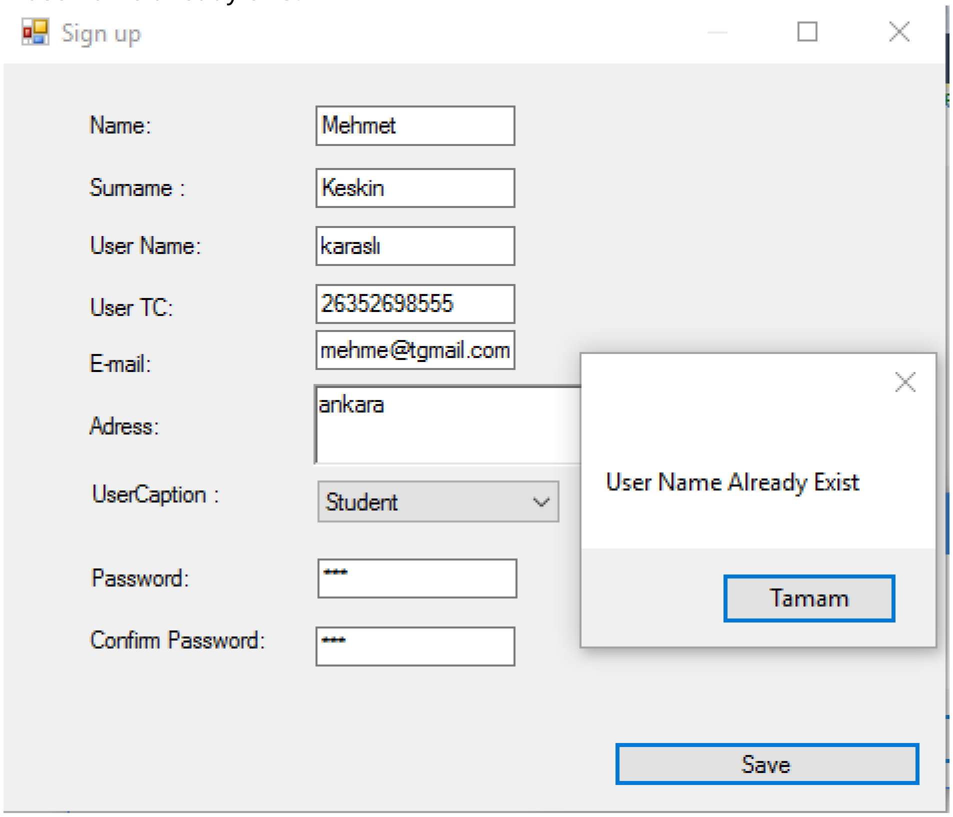
UserCaption : Student

Password: ***

Confirm Password: ****

Please Enter A Valid E-mail Adress!

If username already exist:



Name: Mehmet

Surname : Keskin

User Name: karaslı

User TC: 26352698555

E-mail: mehme@tgmail.com

Adress: ankara

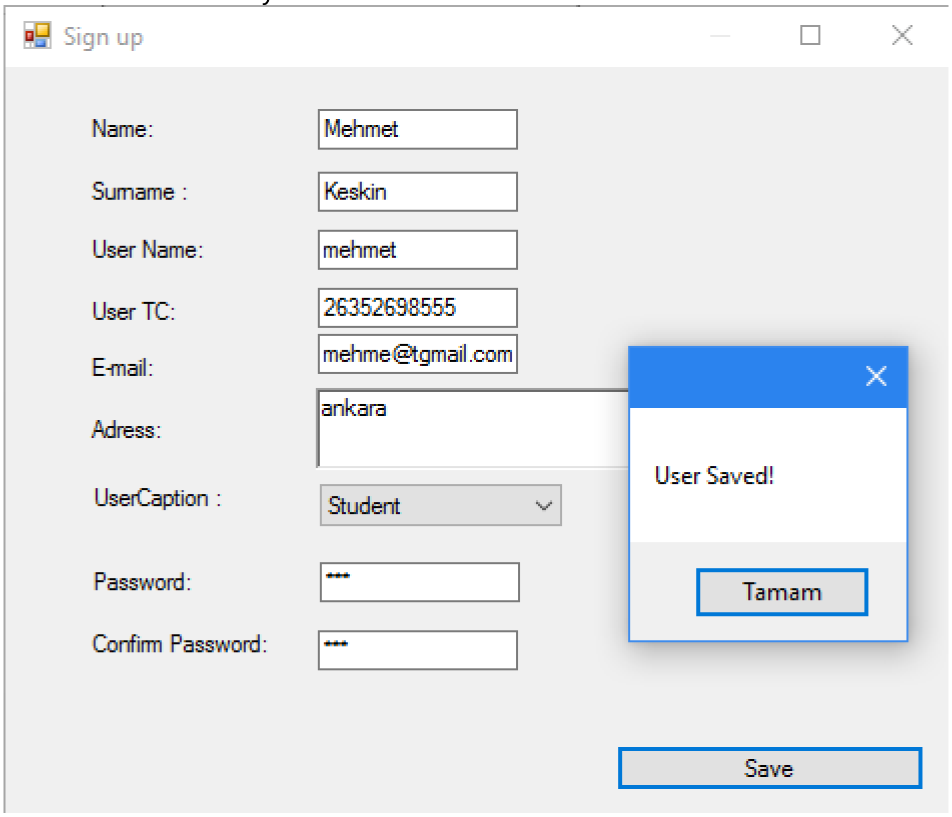
UserCaption : Student

Password: ***

Confirm Password: ***

User Name Already Exist

When user saved to system:



Name: Mehmet

Surname : Keskin

User Name: mehmet

User TC: 26352698555

E-mail: mehme@tgmail.com

Adress: ankara

UserCaption : Student

Password: ***

Confirm Password: ***

User Saved!

10. References:

http://epf.eclipse.org/wikis/openup/core.tech.common.extend_supp/guidances/concepts/arch_mechanism_2932DFB6.html

https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

https://en.wikipedia.org/wiki/Change_management_(engineering)