Student:Vid Alexandru

**Group:30234**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 3

1.3 Non-functional Requirements 3

2. Use-Case Model 3

3. System Architectural Design 3

4. UML Sequence Diagrams 3

5. Class Design 3

6. Data Model 3

7. System Testing 3

8. Bibliography 3

1. Requirements Analysis

# Assignment Specification

Use the C# API to design and implement a client-server application for managing the consultations of doctors in a clinic. The application has three types of users: the clinic secretary, the doctors and an administrator.

# Functional Requirements

The clinic secretary can perform the following operations:

* Add/update patients (patient information: name, identity card number, personal numerical code, date of birth, address).
* CRUD on patients’ consultations (e.g. scheduling a consultation, assigning a doctor to a patient based on the doctor’s availability).

The doctors can perform the following operations:

* Add/view the details of a patient’s (past) consultation.

The administrator can perform the following operations:

* CRUD on user accounts.

# Non-functional Requirements

There will be 3 applications, on for every type of user and including only the functionality for that type of user. The architecture should be client-server and the data will be stored in a database. The client applications will be Windows Forms applications. Use a .Net Web API to expose the server functionality to the client applications. Create a pooling mechanism in a separate Thread which will query the server at predefined short intervals to find out if a new patient has checked in.

2. Use-Case Model

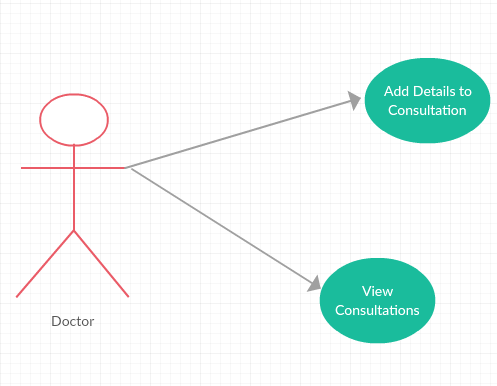
Use case: adding a product

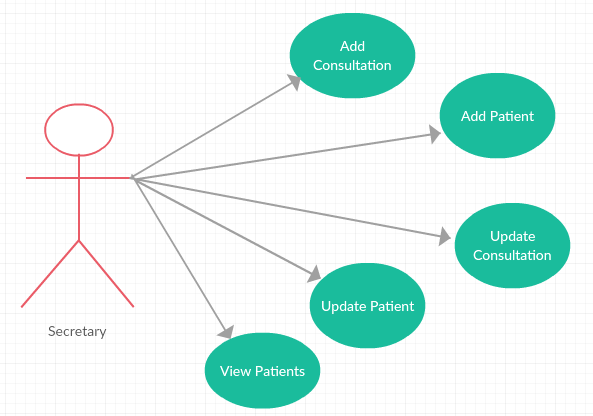
Level: user-goal level

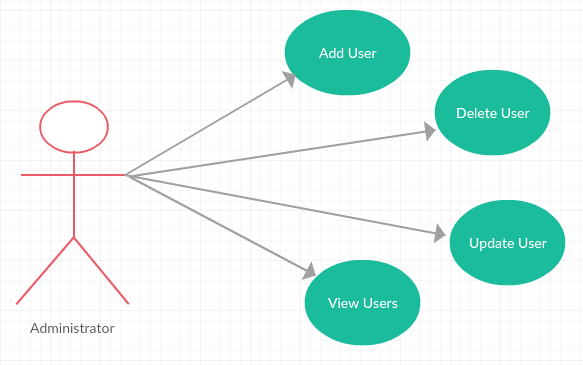
Primary actor: regular user

Main success scenario: login (username and password required)-go to products tab – fill in the information about the product – click “Add Product” button

Extensions: failure : the product already exists in the database







3. System Architectural Design

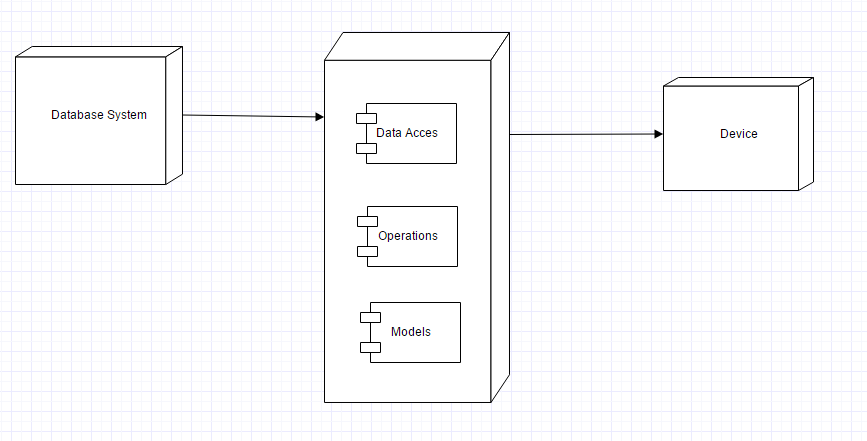
**3.2 Diagrams**

**3.2.1. Conceptual Diagram**



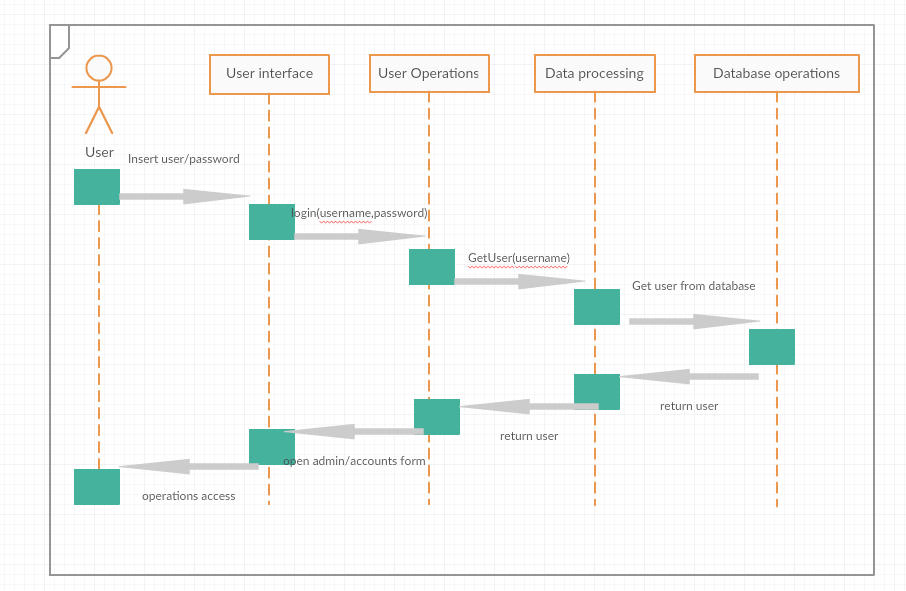
[3-Layer architecture conceptual diagram]

**3.2.2 Package Diagram**



[Deployment Diagram]

4. UML Sequence Diagrams

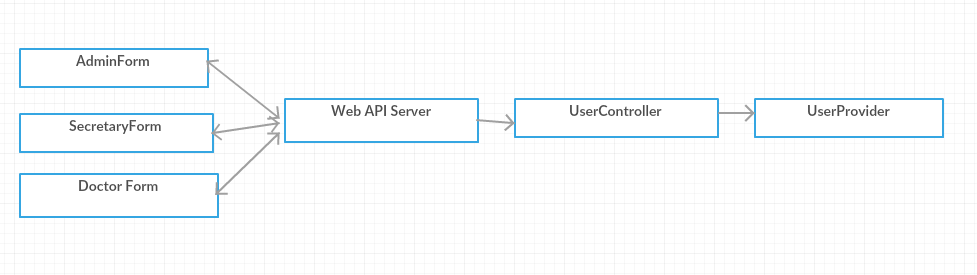


5. Class Design

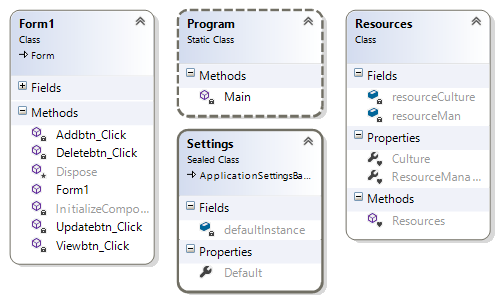
**5.1 Design Patterns Description**

The three-tier architecture model, which is the fundamental framework for the [logical design model](https://msdn.microsoft.com/en-us/library/windows/desktop/ms682280(v=vs.85).aspx), segments an application's components into three tiers of services. These tiers do not necessarily correspond to physical locations on various computers on a network, but rather to logical layers of the application. How the pieces of an application are distributed in a physical topology can change, depending on the system requirements.

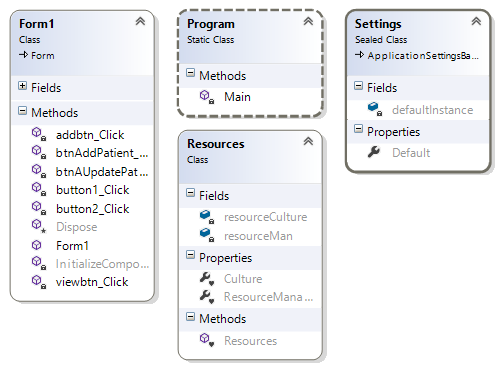
**5.2 UML Class Diagram**

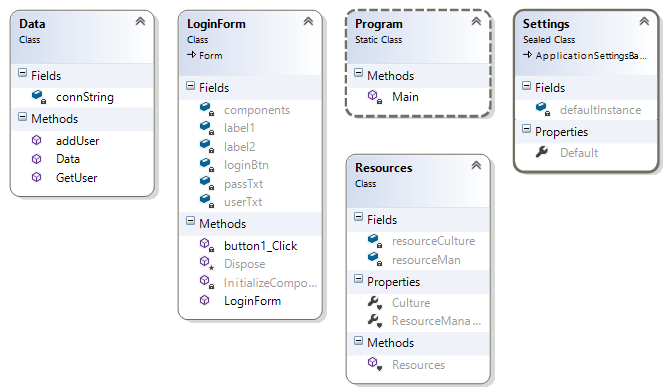


Admin form:

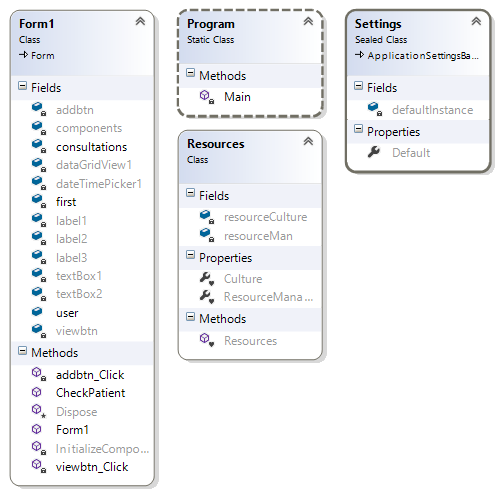


Secretary form:



Login form:

Doctor form:

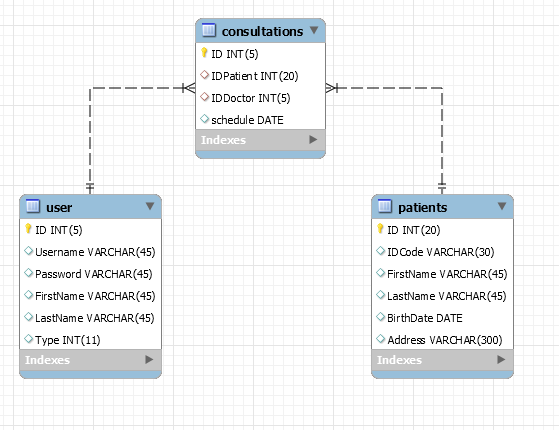


[UML Class Diagram]

6. Data Model

As data models I used in my implementation the following:

* Product
* Order
* Employee
* Order products
* History



8. Bibliography

<https://www.techopedia.com/definition/24649/three-tier-architecture>

<https://en.wikipedia.org/wiki/Multitier_architecture>

<http://exponential.io/blog/2015/03/05/3-layer-architecture-in-detail/>