CA400 - Technical Manual

A close up of a logo

Description automatically generated

# Project Title: Mini Mental State Examination Application

|  |  |
| --- | --- |
| Student Name: | Matthew Nolan |
| Student ID: | 16425716 |
| Student Name: | Michael O’Hara |
| Student ID: | 16414554 |
| Project Supervisor: | Prof. Gareth Jones |
| Date Completed: |  |

# Table of Contents

## Overview

* 1. Motivation
  2. Research

## Design

* 1. Design Concept
  2. User Interface
  3. Backend

## Implementation

3.1 Querying the DB during login

3.2 Adding Doctor Info to the DB

## Challenges Overcome

4.1 Real World Challenges

4.2 Development Challenges

## Future Work

5.1 Converting application to companion

5.2 Adding more examinations

## Appendices

# Overview

## 1.1 Motivation

Our project is based on the Mini Mental State Exam (MMSE). The MMSE is a 30-point questionnaire which is used in clinical and research practices to measure cognitive impairment in patients. It is also used in medicine to screen for dementia. The main purpose of this exam is to estimate the severity and progression of cognitive impairment and to monitor the course of cognitive changes in a patient over time. This makes the test an effective method to document a patient’s response to treatment. The examination includes questions that gauge the patient’s sense of date and time, sense of location, short-term memory, basic mathematics, naming objects and complex cognitive functions like drawing.

The way in which the exam is currently administered is with pen and paper. Our idea to modernize the way in which the MMSE is conducted by creating an application which will allow the test to be carried out in a more efficient manner. The application we hope to produce will digitise the MMSE and allow for the option, in some cases, for the patient to be able to carry out the exam themselves. We plan on constructing a reliable and user-friendly application that will be able to efficiently predict and monitor levels of cognitive impairments in patients, making the application a suitable information management system for any institution which uses the MMSE

## 1.2 Research

Before we began work on our application, we first had to do some research and further our knowledge in a few areas.

* Creating and hosting Databases
* Cross platform client development
* Application Security
* Modernize pen and paper to application
* Accessibility and User-Friendly UI Design

We felt we had a basic knowledge and understanding in each of these areas but in order to develop a full stack application encompassing these areas we needed to do some research.

We looked online and found that developing the application using Xamarin and C# would allow us to seamlessly develop for iOS and Android. Xamarin is an open source app platform from Microsoft for building modern & performant iOS and Android apps with C# and .NET. It allowed us to do this by providing tools and libraries for features such as:

* Libraries for common patterns, such as ModelView ViewModel (MVVM)
* **Editor extensions** to provide syntax highlighting, code completion, designers, and other functionality specifically for developing mobile pages

(https://dotnet.microsoft.com/learn/xamarin/what-is-xamarin)

We also found that AWS has a free tier which would allow us to run and host a MySQL database which we could integrate with our code in order to store/update patient and doctor information. AWS is the world’s most comprehensive and broadly adopted cloud platform, offering over 175 fully and featured services from data centres globally. Some of the services offered by AWS cover the areas of:

* Machine Learning
* Databases
* Gaming Technologies
* Blockchain

(<https://aws.amazon.com/products/?pg=WIAWS-N&tile=learn_more>)

The service we were most interested in learning about was what they could offer in terms of Database hosting and management. We investigated it and on the beginner tier they offer 12 months of Relational Database Service. This allowed us to set up a MySQL database for storing records and have them be accessible from any instance of the application (with correct login credentials).

We also needed to do a good bit of research into methods to make our User Interface accessible and user friendly for both medical professionals and the cognitively impaired, while retaining a high level of functionality and productivity. Research into this topic was done by reading a number of articles on Google Scholar and IEEE Xplore. One of the key articles we used which outlines the key factors which affect users suffering from cognitive impairment from 2013 can be found here:

(<https://www.researchgate.net/publication/236861806_Issues_with_Designing_Dementia-Friendly_Interfaces>)

It outlines key factors that need to be taken account of such as limited motor function or decreased visual impairment. We used the information we found in the articles we used to aid in the design of our User Interface.

Another article we found and used can be found here:

[(https://ieeexplore.ieee.org/abstract/document/4224212?casa\_token=RkBdbm47t5cAAAAA:uF8TCELN-8u\_6J743PfXXUcywNFVGXq0cdrgevydqyXdEltpaS6Oo0VS0y8lD9770ouuG5YXPA)]((https:/ieeexplore.ieee.org/abstract/document/4224212?casa_token=RkBdbm47t5cAAAAA:uF8TCELN-8u_6J743PfXXUcywNFVGXq0cdrgevydqyXdEltpaS6Oo0VS0y8lD9770ouuG5YXPA))

# Design

## 2.1 Design Overview/Concept

This is an overview of what the design concept and use case of the application for a medical professional would be when dealing with a patient.

When the app is initially launched the user is greeted by a Login Screen. Here the user will enter their login details and be given access to the application. If the user does not have an account, they will create one and will be given an ID number which will be used to link the doctor to the patient when carrying out the test.

Once the user enters the application, they will be greeted by a home screen or dashboard. From here the user will be able to access the functions which our application offers.

They will also be able access a list of patients in their care. This was to facilitate the reviewing of results after the fact to gather an appropriate diagnosis. This option allows the doctor to review and compare results to previous results (if the patient has carried out the exam before). In this section the doctor is able to add a new patient to the database. In this section the doctor will add in the patients first and last name, their age. Then as the patients carries out exams their profile will keep a record of when that test was and what score they got in that test.

The doctor can also view a calendar from this screen which will allow them to see upcoming consultations with patients or it will allow them to schedule the next consultation with a patient by adding it into the calendar.

The user can start up the test and enter in the patient’s ID and name in order to link the results to said patient. This will bring the medical professional to the exam page. The exam is broken down into 6 sections, which are *Orientation, Registration, Attention and Calculation, Recall, Language and Copying.* These sections each contain a number of questions which are designed to test the mental faculties of the patient in order to determine the level of cognitive impairment in the patient. The questions range from questions like *“What is todays date? Or What is the name of this hospital/location”* all the way to instructions like “*Copy this image or follow this 3-stage command”.* The answers will then be saved and graded by the application and return the result on the screen to the medical professional. This eliminates the need for the doctor to manually correct the exam and calculate the results for the patient.

## 2.2 Technologies Used

### **User Interface**

Our Design is built using Xamarin for designing the User Interface. We decided to this for a number of reasons.

#### Single Technology Stack

As Xamarin uses C# complemented with the .NET framework it allows for easy creating apps of each of the mobile platforms such as iOS and Android. This allowed us to reuse a lot of source code and sped up the development life cycle. It also meant we did not need to switch development environments when developing for each platform as we were able to develop for all platform natively from Visual Studio.

#### Simplified Maintenance

It also simplified the maintenance and updating life cycle of our application. Due to its cross-platform nature all we needed to do was deploy our changes or updates to the source file and then they were automatically applied to both iOS and Android.

#### Native User Experiences

It allowed us to create user experiences that mirror native android or iOS development due to the ability to use platform-specific UI elements. This is achieved by using Xamarin forms which converts app UI components into the platform-specific interface elements at runtime.

#### Hardware Support

From using Xamarin our applications had native-level app functionality. It eliminated all hardware compatibility issues, by using plugins and specific APIs, to work with common devices functionality across the platforms we were developing for. Along with the access to platform-specific APIs, Xamarin supports linking with native libraries. This gave us more options for better customization and native-level functionality.

### **Backend**

Our backend is a combination of a MySQL Database which is hosted by AWS and the source code which is written in C#.

### MySQL

We decided to use MySQL database as the relational database management system for our project. We also decided to host our database using the free tier of Amazon Web Service. This allowed us to keep the database hosted and running at all times. We did this for a number of reasons

#### Ease of Use

We decided to use AWS as it is a very powerful web hosting platform. It also does not require a huge amount of tech skill or expertise to use. The AWS Management Console or well-documented web services APIs to can be used to easily access AWS’s application hosting platform.

#### Security & Encryption

Another reason for using AWS is because of its high level of security and ability to protect information. AWS provides a more reliable security measure that is guaranteed to keep your data safe and secure. With 12 data centres scattered across the globe, and another 5 slated to open this year, this is as safe as it gets for your private data and information.

#### Scalable and high performance

Using AWS tools, Auto Scaling, and Elastic Load Balancing, your application can scale up or down based on demand. Backed by Amazon’s massive infrastructure, you have access to compute and storage resources when you need them. (<https://aws.amazon.com/application-hosting/benefits/>)

### **C#**

The reasons we chose C# for our back-end code logic are outlined below

#### Object-Oriented

C# is an object-oriented programming language. It supports Data Encapsulation, inheritance, polymorphism, interfaces. C# being object-oriented makes development and maintenance easier when compared to other languages which are procedure-oriented programming language

#### Ease of Development

C# language has a rich class of libraries that make many functions easy to be implemented. This compounded with the ability to add more libraries through the use of NuGet packages meant we had a wide range of possibilities to aid the development of our application

# Implementation

## 3.1 Adding Doctor Info to the DB

A screenshot of a cell phone

Description automatically generated

The code above shows the backend of our application when a doctor is registering to use our application. At the top of the function you can see the string called ‘CS’ which is the connections string for our AWS hosted database. In a future version of the application we wouldn’t have our database log in details in our connection string but due some difficulties with SQLite not being an adequate database solution we thought, and at this late a stage in the project we weren’t able to implement that.

Moving on through the code we used the MySqlConnection package for managing any interaction with our database. We wrapped connecting into our database inside and try and catch which would allow our code catch ‘MySqlExceptions’ and throw an error message to the user telling them to try again. Inside the try we had an ‘if’ statement to check the connection state to our database and if it were closed then it would move inside the if and open the connection to our database. Then we insert that data into the database using the ‘INSERT’ command and “Parameters.AddWithValue()” method to add the values that the user enters. It also sets the result success parameter to true and then in the finally block we call connection.Close() to close the connection to our database and finally we return ‘res’ be used later.

## 3.2 Querying the DB during login

A screenshot of a computer

Description automatically generated

This code is the backend code which runs when an already registered user tries to login to our application. We initialise a list of users which have the parameters UserName and PassWord. This method re-uses the same code for connecting to the database. Once again using ‘MySqlCommand’ but this time we use a string called ‘query’ which gets all the usernames and passwords in our User table. We then used the MySqlDataReader in order to read through the results which were returned. We then create a new user for each row in the database which was returned and set the UserName and Password for that user we then add that user to the list of users.

After each of the users has be set and added to the list. We read through all the usernames in the list and see if one matches the one the user entered. We then do the same thing for password. If we get a match for both we return a message saying, “Login successful”. We also set result to true and return that to be used later. If one of the parameters doesn’t match, we return a message saying “Incorrect Username/Password” (depending on which one was wrong).

# Challenges Overcome

## 4.1 Real World Challenges

Even though this is not considered a direct challenge to the development of our application the Covid-19 outbreak was a challenge which we had to overcome during the life cycle of our project. Had the outbreak not occurred we would have worked on our project in the School of Computing computer labs but as it did occur, we needed to adjust to working together remotely. We used Zoom in order to communicate via video and co-ordinate what each of us would do in order to continue development.

Michael is also a part time worker in a shop which is deemed an essential service and as a result of lockdown the shop got busier as everyone was out purchasing their items day to day needs meaning that all workers were asked to do more hours meaning his time became divided between working and completing studies. Through careful planning and time management I was able to complete the tasks which we set out in order to complete work on our application.

## 4.2 Development Challenges

#### C# Full Stack Application

The first challenge we had during this project was developing a full stack application in C#. During the different modules in our 4 years we had done a number of different languages, but C# was not one of them, so we decided to take it upon ourselves to try to develop our application in C#. We had both gotten a taste for it during our INTRA placements but not to the level of experience we had with Python. We knew about the wide range of libraries available in C# and this is why we decided to use it.

#### Hosting the Database

An initial challenge we faced was where would we host the Database in order to allow connectivity at all times whenever it is needed. We knew this would be an issue from the start of the project but we thought that we would be able to use SQLite as the database of choice but this would lead to separate instances of DB for every different device the app is used on rather than one shared database each device connects to.

This led us to finding out about using AWS as the method for hosting a MySQL database which each instance of the application will connect to and send data to and retrieve data from. This would also for a user to login on multiple devices if one were to stop working or to be misplaced.

#### Integrating Source Code/DB with UI

Another big challenge we faced was the integrating our source code containing the backend functions and the code for connecting to, querying, and writing the database with code written in Xamarin which defined our User Interface. We had worked with each part of the code in isolation in previous modules and even on our INTRA placements, but we had never worked to integrate the two so this was a new challenge for us

We spent a couple of hours working through each element of the UI and linking the backend code to that related to that element.

# Future Work

### 5.1 Converting application to companion

Due to the way hospitals are run, they use pen and paper to track of everything. Given the wide range of technologies available in today’s age the field of medicine should also be able to follow suit. We feel that our application could be updated to be transformed into a companion application for medical professionals. The medical professional would be able to take notes and add information to a patient file straight from the application. They would be able to view data on all the patients in their care all in the one place rather than needing to go their office to get the files on different patient. This would reduce the time between seeing different patients.

### 5.2 Adding more examinations

Another avenue that we could venture down would be adding more exams that the medical professional can carry out. These could be added into the application under a separate menu heading and the results for these tests could be added into different tables in the database. Given the different nature of these exams, they would need some tinkering to be adapted to the application format.

Some of these could include:

#### NIH Stroke Scale

The national Institutes of Health Stroke Scale is a tool that is used by all major healthcare providers to objectively quantify the impairment caused by a stroke. Its composed of 11 different items, each of which scores a specific ability between a 0 and 4. Scores in each area of 0 indicate normal function in that area and that a higher score is indicative of a level of impairment. After each section has been completed the scores are added up and a result is determined. The maximum possible score is 42, with the minimum score being a 0.

#### Self-administered Gerocognitive Examination

The self-administered Gerocognitive Examination is a brief cognitive assessment instrument for mild cognitive impairment (MCI) and early dementia. It was created by Douglas Scharre, who is a Professor of Clinical Neurology and Psychiatry at Ohio State University Wexner Medical Centre

#### Informant Questionnaire on Cognitive Decline in the Elderly

This a questionnaire that can be filled out by relative or other supporter of an elderly person and this can determine whether that person has declined in cognitive functioning. This exam is used as a screening test for dementia. This questionnaire contains 26 everyday situations where a person has to use their memory or intelligence. Example of such situations include: “Remembering where to find things which have been put in a different place from usual”. The scored for IQCODE is scored by averaging the ratings across the 26 situations. A person with no cognitive decline will score an average of 3 across the board. This exam could be use in combination with the Mini-Mental State Examination to improve the detection of dementia.