



CTU

TECHNICAL MANUAL

CTUCare Applicaton

Jarrood Venter

STB16JV9306045062082

MCSD C# FINAL APPLICATION

Contents

Introduction	3
MainWindow (Splash Screen)	4
C#	4
XAML	5
Login	5
C#	6
XAML	7
Home	8
C#	8
XAML	9
Add User	10
C#	10
XAML	11
History	12
C#	12
XAML	16
Accounts	18
C#	18
XAML	22
Information	24
C#	24
XAML	25
Appointments	26
C#	26
XAML	27
New Appointment	28
C#	28
XAML	32
Open Appointments	34
C#	34
XAML	46
New Patient	48
C#	48
XAML	50
New Procedure	51

C#	51
XAML.....	52
New Medicine	53
C#	53
XAML.....	54
New Doctor	55
C#	55
New Medical Aid Company.....	56
C#	56
XAML.....	57
PatientToken.cs.....	58
Update.cs	58
UserType.cs.....	59
Database Diagram.....	60
Error Report	61
Error 1	61
Solution 1	61
Error 2	62
Solution 2	62
Error 3	62
Solution 3	63
Conclusion.....	63

Introduction

When starting out this application, there was a heavy emphasis on ensuring that one followed the planning manual due to the sheer size of the application. Multiple windows were planned as well as an extensive database that would require strong integration.

One element of variability (where one was unlikely to follow their own plan) that was likely to come through was for the appointment system which had been given absolutely no guidelines and was up to the interpretation of the programmer

MainWindow (Splash Screen)



C#

```
using System.Threading.Tasks;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();

            splashScreenStart();

        }

        //Logic for the progressBar to load for 2 seconds
        private async void splashScreenStart()
        {
            loader.Value = 0;
            loader.Maximum = 100;

            for (int i = 0; i < 101; i++)
            {
                await Task.Delay(5);
                loader.Value += 1;
            }

            Login initialize = new Login();

            initialize.Show();

            this.Close();
        }
    }
}
```

XAML

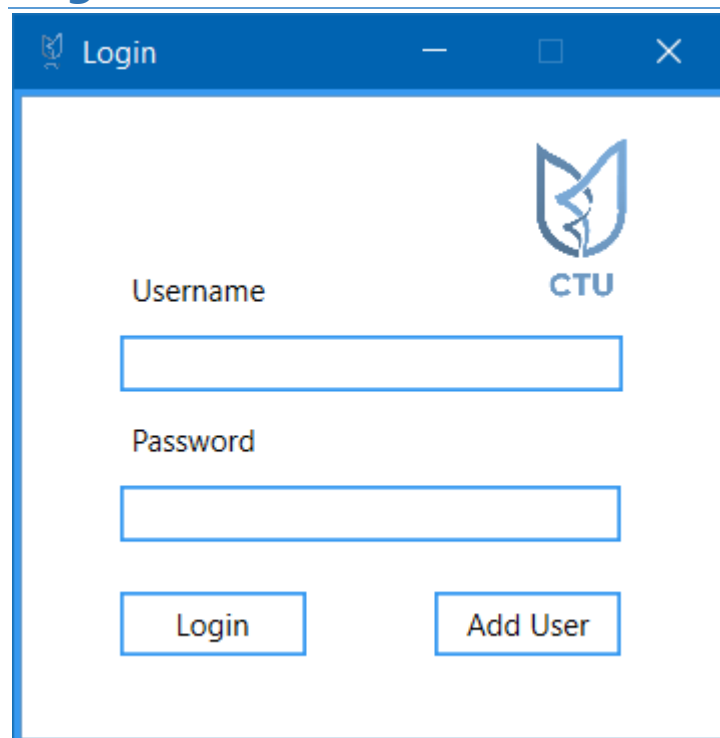
```

<Window x:Class="CTUCare.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    WindowStyle="None" Background="White" BorderBrush="White"
    WindowStartupLocation="CenterScreen"
    ResizeMode="NoResize" Height="199.2" Width="402.133" BorderThickness="3">
    <Grid Background="#FF3699F1" Margin="0,0,0.2,0.4">
        <Grid.RowDefinitions>
            <RowDefinition/>
        </Grid.RowDefinitions>
        <Rectangle HorizontalAlignment="Left" Height="102" Margin="39,46,0,0"
VerticalAlignment="Top" Width="90">
            <Rectangle.Fill>
                <ImageBrush ImageSource="Images/Picture4.png"/>
            </Rectangle.Fill>
        </Rectangle>
        <TextBlock x:Name="textBlock" HorizontalAlignment="Left" Margin="220,96,0,0"
TextWrapping="Wrap" Text="CTU Care" VerticalAlignment="Top" Height="75" Width="159"
Foreground="White" FontSize="24" FontFamily="SnasmW00-Regular"/>
        <ProgressBar x:Name="loader" HorizontalAlignment="Left" Height="22"
Margin="220,126,0,0" VerticalAlignment="Top" Width="128" BorderBrush="White"
Background="White" Foreground="#FF3699F1"/>

    </Grid>
</Window>

```

Login



Login

CTU

Username

Password

Login Add User

C#

```
using System;
using System.Linq;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for Login.xaml
    /// </summary>
    public partial class Login : Window
    {
        public Login()
        {
            InitializeComponent();
        }

        private void btnLogin_Click(object sender, RoutedEventArgs e)
        {
            ValidateCredentials();

            this.Close();
        }

        AddUser newUserWindow = new AddUser();

        private void btnAddUser_Click(object sender, RoutedEventArgs e)
        {
            try
            {
                newUserWindow.Show();
            }
            catch (Exception)
            {
                AddUser newUserWindow = new AddUser();
                newUserWindow.Show();
            }
        }

        Home newHome = new Home();

        private void ValidateCredentials()
        {
            string userUserName = txtUserName.Text;
            string userPassword = txtPassword.Password;

            var userList = CTUCareDB.Instance.Users.ToList();
            var dbUser = userList.Find(x => x.Username == userUserName);

            //1 for admin user, 2 for regular user
            int adminUser = 1;

            if (dbUser != null)
            {
                if (dbUser.Password == userPassword)
                {
                    //Acts as a token for the users
                    if (dbUser.IsUserAdmin == adminUser)
                    {
                        UserType.UserKind = 1;
                    }
                }
            }
        }
    }
}
```

```

    }
    else
        UserType.UserKind = 2;

    try
    {
        newHome.Show();
    }
    catch(Exception)
    {
        Home newHome = new Home();
        newHome.Show();
    }
}
}
}
}
}
}
}
}

```

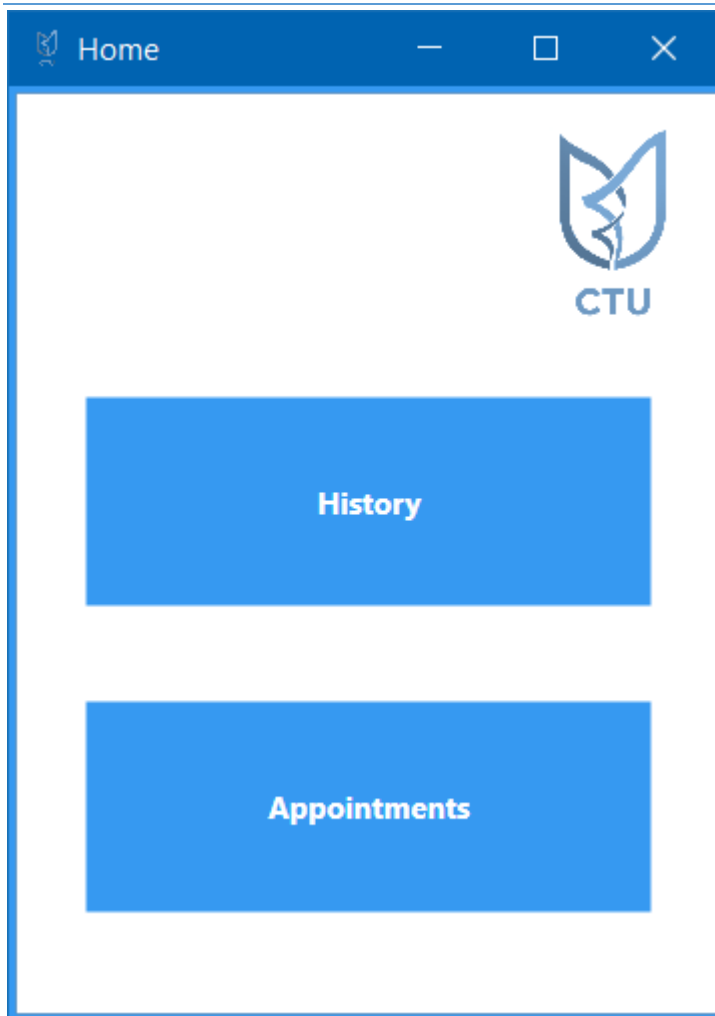
XAML

```

<Window x:Class="CTUCare.Login"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="Login" Height="300" Width="300"
    Icon="images/picture5.png" BorderThickness="3" BorderBrush="#FF3699F1"
    ResizeMode="CanMinimize">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
        </Grid.ColumnDefinitions>
        <TextBox x:Name="txtUserName" Height="23" Margin="39,95,39.6,0"
            TextWrapping="Wrap" VerticalAlignment="Top" BorderBrush="#FF3699F1"
            BorderThickness="1.5"/>
        <Label x:Name="label" Content="Username" HorizontalAlignment="Left"
            Margin="39,64,0,0" VerticalAlignment="Top" Width="83"/>
        <Label x:Name="label_Copy" Content="Password" HorizontalAlignment="Left"
            Margin="39,124,0,0" VerticalAlignment="Top" Width="83"/>
        <PasswordBox x:Name="txtPassword" Margin="39,155,40,0" VerticalAlignment="Top"
            RenderTransformOrigin="0.5,0.5" Height="23" BorderBrush="#FF3699F1"
            BorderThickness="1.5"/>
        <Button x:Name="btnLogin" Content="Login" Margin="39,198,166,0"
            VerticalAlignment="Top" BorderBrush="#FF3699F1" BorderThickness="1.5"
            Background="White" Height="25" Click="btnLogin_Click"/>
        <Button x:Name="btnAddUser" Content="Add User" Margin="165,198,40,0"
            VerticalAlignment="Top" BorderBrush="#FF3699F1" BorderThickness="1.5"
            Background="White" Height="25" Click="btnAddUser_Click"/>
        <Rectangle Height="79" Margin="202,10,34,0" VerticalAlignment="Top">
            <Rectangle.Fill>
                <ImageBrush ImageSource="Images/Picture5.png" Stretch="Uniform"/>
            </Rectangle.Fill>
        </Rectangle>
    </Grid>
</Window>

```


Home



C#

```
using System;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for Home.xaml
    /// </summary>
    public partial class Home : Window
    {
        public Home()
        {
            InitializeComponent();
        }

        private void btnAppointments_Click(object sender, RoutedEventArgs e)
        {
            Appointments newApp = new Appointments();

            newApp.Show();

            this.Close();
        }
    }
}
```

```

        Appointments openAppointments = new Appointments();
        private void btnHistory_Click(object sender, RoutedEventArgs e)
        {
            try
            {
                openAppointments.Show();
            }
            catch (Exception)
            {
                Appointments openAppointments = new Appointments();
                openAppointments.Show();
            }
        }
    }
}

```

XAML

```

<Window x:Class="CTUCare.Home"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:CTUCare"
        mc:Ignorable="d"
        Title="Home" Height="411.2" Width="300"
        Icon="images/Picture5.png" BorderBrush="#FF3699F1" BorderThickness="3">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="106*" />
            <RowDefinition Height="127*" />
            <RowDefinition Height="143*" />
        </Grid.RowDefinitions>
        <Button x:Name="btnHistory" Content="History" Margin="26,15.4,24,20.8"
            BorderThickness="2" Background="#FF3699F1" BorderBrush="White" Foreground="White"
            FontWeight="Bold" Grid.Row="1" Click="btnHistory_Click"/>
        <Button x:Name="btnAppointments" Content="Appointments" Margin="26,13.2,24,39"
            BorderThickness="2" Background="#FF3699F1" BorderBrush="White" Foreground="White"
            FontWeight="Bold" Grid.Row="2" Click="btnAppointments_Click"/>
        <Rectangle HorizontalAlignment="Left" Margin="189,12,0,9.6" Width="99">
            <Rectangle.Fill>
                <ImageBrush ImageSource="Images/Picture5.png" Stretch="Uniform"/>
            </Rectangle.Fill>
        </Rectangle>
    </Grid>
</Window>

```

Add User

The image shows a 'New User' dialog box with the following fields and controls:

- Username**: A text input field.
- Password**: A text input field.
- ☐ **New User has Admin Rights**: A checkbox.
- Admin Password**: A text input field.
- Add User**: A button at the bottom.

C#

```
using System;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for Home.xaml
    /// </summary>
    public partial class Home : Window
    {
        public Home()
        {
            InitializeComponent();
        }

        private void btnAppointments_Click(object sender, RoutedEventArgs e)
        {
            Appointments newApp = new Appointments();

            newApp.Show();
        }
    }
}
```

```

        this.Close();
    }

    History openHistory = new History();
    private void btnHistory_Click(object sender, RoutedEventArgs e)
    {
        try
        {
            openHistory.Show();
        }
        catch (Exception)
        {
            History openHistory = new History();
            openHistory.Show();
        }

        this.Close();
    }
}

```

XAML

```

<Window x:Class="CTUCare.AddUser"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="New User" Height="404.898" Width="300"
    Icon="images/Picture5.png"
    ResizeMode="CanMinimize" BorderBrush="#FF3699F1" BorderThickness="3">
    <Grid>
        <TextBox x:Name="txtUserName" Height="23" Margin="42,71,41.6,0"
            TextWrapping="Wrap" VerticalAlignment="Top" BorderBrush="#FF3699F1"
            BorderThickness="1.5"/>
        <Label x:Name="label" Content="Username" HorizontalAlignment="Left"
            Margin="42,40,0,0" VerticalAlignment="Top" Width="83"/>
        <Label x:Name="label_Copy" Content="Password" HorizontalAlignment="Left"
            Margin="42,100,0,0" VerticalAlignment="Top" Width="83"/>
        <PasswordBox x:Name="txtPassword" Margin="42,131,42.6,0"
            VerticalAlignment="Top" RenderTransformOrigin="0.5,0.5" Height="23"
            BorderBrush="#FF3699F1" BorderThickness="1.5"/>
        <Label x:Name="label_Copy1" Content="Admin Password"
            HorizontalAlignment="Left" Margin="42,223,0,0" VerticalAlignment="Top" Width="101"
            IsEnabled="False"/>
        <PasswordBox x:Name="txtAPassword" Margin="42,254,42.6,0"
            VerticalAlignment="Top" RenderTransformOrigin="0.5,0.5" Height="23"
            BorderBrush="#FF3699F1" BorderThickness="1.5"
            IsEnabled="False"/>
        <Button x:Name="btnAddUser" Content="Add User" Margin="106,305,104.6,0"
            VerticalAlignment="Top" BorderBrush="#FF3699F1" BorderThickness="1.5"
            Background="White" Height="25" Click="btnAddUser_Click"/>
        <CheckBox x:Name="checkBox" Content="New User has Admin Rights"
            HorizontalAlignment="Left" Margin="57,188,0,0" VerticalAlignment="Top"
            Checked="checkBox_Checked" Unchecked="checkBox_Unchecked"/>
    </Grid>
</Window>

```

History

C#

```

using System;
using System.Linq;
using System.Windows;
using System.Windows.Controls;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for History.xaml
    /// </summary>
    public partial class History : Window
    {
        public History()
        {
            InitializeComponent();

            populateList();
        }

        public static int statementPatient;

        private void search()
        {
            listPatients.Items.Clear();
        }
    }
}

```

```
if (cboxSearchBy.SelectedItem == iName)
{
    listPatients.Items.Clear();

    foreach (var item in CTUCareDB.Instance.Patients)
    {
        if (item.Name.ToLower().Contains(searchBox.Text.ToLower()))
        {
            listPatients.Items.Add(item.Name);
        }
    }
}
else if (cboxSearchBy.SelectedItem == iSurname)
{
    listPatients.Items.Clear();

    foreach (var item in CTUCareDB.Instance.Patients)
    {
        if (item.Name.ToLower().Contains(searchBox.Text.ToLower()))
        {
            listPatients.Items.Add(item.Surname);
        }
    }
}
else if (cboxSearchBy.SelectedItem == iIdNo)
{
    listPatients.Items.Clear();

    foreach (var item in CTUCareDB.Instance.Patients)
    {
        if (item.IDNumber.Contains(searchBox.Text.ToLower()))
        {
            listPatients.Items.Add(item.IDNumber);
        }
    }
}
else if (cboxSearchBy.SelectedItem == iMedicalAidNo)
{
    listPatients.Items.Clear();

    foreach (var item in CTUCareDB.Instance.Patients)
    {
        if (item.MedicalAidNo.Contains(searchBox.Text.ToLower()))
        {
            listPatients.Items.Add(item.MedicalAidNo);
        }
    }
}
else if (cboxSearchBy.SelectedItem == iAccNo)
{
    listPatients.Items.Clear();

    foreach (var item in CTUCareDB.Instance.Patients)
    {
        if
(item.AccountNumber.ToString().Contains(searchBox.Text.ToLower()))
        {
            listPatients.Items.Add(item.AccountNumber);
        }
    }
}
else if (cboxSearchBy.SelectedItem == iFileNo)
```

```
{
    listPatients.Items.Clear();

    foreach (var item in CTUCareDB.Instance.Patients)
    {
        if (item.ID.ToString().Contains(searchBox.Text.ToLower()))
        {
            listPatients.Items.Add(item.ID);
        }
    }
}

private void populateList()
{
    listPatients.Items.Clear();

    if (cboxSearchBy.SelectedItem == iName)
    {
        foreach (var item in CTUCareDB.Instance.Patients)
        {
            listPatients.Items.Add(item.Name);
        }
    }
    else if (cboxSearchBy.SelectedItem == iSurname)
    {
        foreach (var item in CTUCareDB.Instance.Patients)
        {
            listPatients.Items.Add(item.Surname);
        }
    }
    else if (cboxSearchBy.SelectedItem == iIdNo)
    {
        foreach (var item in CTUCareDB.Instance.Patients)
        {
            listPatients.Items.Add(item.IDNumber);
        }
    }
    else if (cboxSearchBy.SelectedItem == iMedicalAidNo)
    {
        foreach (var item in CTUCareDB.Instance.Patients)
        {
            listPatients.Items.Add(item.MedicalAidNo);
        }
    }
    else if (cboxSearchBy.SelectedItem == iAccNo)
    {
        foreach (var item in CTUCareDB.Instance.Patients)
        {
            listPatients.Items.Add(item.AccountNumber);
        }
    }
    else if (cboxSearchBy.SelectedItem == iFileNo)
    {
        foreach (var item in CTUCareDB.Instance.Patients)
        {

```

```

        listPatients.Items.Add(item.ID);
    }
}

private void setPatient()
{
    if (cboxSearchBy.SelectedItem == iName)
    {
        int user = CTUCareDB.Instance.Patients.Where(x => x.Name ==
listPatients.SelectedItem.ToString()).FirstOrDefault().ID;
        PatientToken.patientID = user;
    }
    else if (cboxSearchBy.SelectedItem == iSurname)
    {
        int user = CTUCareDB.Instance.Patients.Where(x => x.Surname ==
listPatients.SelectedItem.ToString()).FirstOrDefault().ID;
        PatientToken.patientID = user;
    }
    else if (cboxSearchBy.SelectedItem == iIdNo)
    {
        int user = CTUCareDB.Instance.Patients.Where(x => x.IDNumber ==
listPatients.SelectedItem.ToString()).FirstOrDefault().ID;
        PatientToken.patientID = user;
    }
    else if (cboxSearchBy.SelectedItem == iMedicalAidNo)
    {
        int user = CTUCareDB.Instance.Patients.Where(x => x.MedicalAidNo ==
listPatients.SelectedItem.ToString()).FirstOrDefault().ID;
        PatientToken.patientID = user;
    }
    else if (cboxSearchBy.SelectedItem == iAccNo)
    {
        int user = CTUCareDB.Instance.Patients.Where(x => x.AccountNumber ==
Convert.ToInt32(listPatients.SelectedItem.ToString())).FirstOrDefault().ID;
        PatientToken.patientID = user;
    }
    else if (cboxSearchBy.SelectedItem == iFileNo)
    {
        int user = CTUCareDB.Instance.Patients.Where(x => x.ID ==
Convert.ToInt32(listPatients.SelectedItem.ToString())).FirstOrDefault().ID;
        PatientToken.patientID = user;
    }
}

#region Event Methods
private void cboxSearchBy_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    populateList();
}

private void searchBox_TextChanged(object sender, TextChangedEventArgs e)
{
    search();
}

Information openInfo = new Information();
private void btnInformation_Click(object sender, RoutedEventArgs e)
{
    if (listPatients.SelectedIndex != -1)

```



```

        {
            setPatient();

            try
            {
                openInfo.Show();
            }
            catch (Exception)
            {
                Information openInfo = new Information();
                openInfo.Show();
            }
        }
        else
        {
            MessageBox.Show("Please select a patient");
        }
    }

    Accounts HistoryNAccounts = new Accounts();
    private void btnAccountsNHistory_Click(object sender, RoutedEventArgs e)
    {
        if (listPatients.SelectedIndex != -1)
        {
            statementPatient = CTUCareDB.Instance.Patients.Where(x => x.Name ==
listPatients.SelectedItem.ToString()).FirstOrDefault().ID;

            setPatient();

            try
            {
                HistoryNAccounts.Show();
            }
            catch (Exception)
            {
                Accounts HistoryNAccounts = new Accounts();
                HistoryNAccounts.Show();
            }
        }
        else
        {
            MessageBox.Show("Please select a patient");
        }
    }
    private void btnBack_Click(object sender, RoutedEventArgs e)
    {
        Home goBack = new Home();
        goBack.Show();
        this.Close();
    }

    #endregion
}
}

```

XAML

```

<Window x:Class="CTUCare.History"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

```

```

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
xmlns:local="clr-namespace:CTUCare"
mc:Ignorable="d"
Title="History" Height="427.211" Width="531.773"
Icon="images/Picture5.png" BorderBrush="#FF3699F1" BorderThickness="4">
<Grid>
    <ListBox x:Name="listPatients" Margin="37,71,0,0" BorderBrush="#FF3699F1"
BorderThickness="1.5" Height="219" VerticalAlignment="Top" Width="138"
HorizontalAlignment="Left"/>
    <TextBox x:Name="searchBox" Margin="37,47,0,0" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="1.5" Height="19" VerticalAlignment="Top"
Width="138" HorizontalAlignment="Left" TextChanged="searchBox_TextChanged" />
    <Label x:Name="label_Copy3" Content="Patient" HorizontalAlignment="Left"
Margin="32,21,0,0" Height="26" VerticalAlignment="Top"/>
    <ComboBox x:Name="cboxSearchBy" HorizontalAlignment="Left" Margin="37,334,0,0"
VerticalAlignment="Top" Width="138" SelectionChanged="cboxSearchBy_SelectionChanged">
        <ComboBoxItem x:Name="iName" Content="Name" HorizontalAlignment="Left"
Width="136"/>
        <ComboBoxItem x:Name="iSurname" Content="Surname"
HorizontalAlignment="Left" Width="136"/>
        <ComboBoxItem x:Name="iIdNo" Content="ID Number"
HorizontalAlignment="Left" Width="136"/>
        <ComboBoxItem x:Name="iMedicalAidNo" Content="Medical Aid Number"
HorizontalAlignment="Left" Width="136"/>
        <ComboBoxItem x:Name="iAccNo" Content="Account Number"
HorizontalAlignment="Left" Width="136"/>
        <ComboBoxItem x:Name="iFileNo" Content="File Number"
HorizontalAlignment="Left" Width="136"/>
    </ComboBox>
    <Button x:Name="btnAccountsNHistory" Content="Accounts / History"
HorizontalAlignment="Left" Margin="224,47,0,0" VerticalAlignment="Top" Width="267"
Height="144" Background="White" BorderBrush="#FF3699F1" BorderThickness="2"
Click="btnAccountsNHistory_Click" />
    <Button x:Name="btnInformation" Content="Information"
HorizontalAlignment="Left" Margin="224,222,0,0" VerticalAlignment="Top" Width="267"
Height="135" BorderBrush="#FF3699F1" Background="White" BorderThickness="2"
Click="btnInformation_Click"/>
    <Label x:Name="label_Copy" Content="Search by" HorizontalAlignment="Left"
Margin="37,303,0,0" Height="26" VerticalAlignment="Top"/>
    <Button x:Name="btnBack" Content="Button" Foreground="{x:Null}"
BorderBrush="{x:Null}" Margin="0,2,0,0" Click="btnBack_Click"
HorizontalAlignment="Left" Width="27" Height="22" VerticalAlignment="Top">
        <Button.Background>
            <ImageBrush ImageSource="Images/Picture1.png" Stretch="Uniform"/>
        </Button.Background>
    </Button>

</Grid>
</Window>

```

Accounts

C#

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for Accounts.xaml
    /// </summary>
    public partial class Accounts : Window
    {
        public Accounts()
        {
            InitializeComponent();

            loadStatements();
        }

        #region Loading Methods
        private void loadProPrice()
        {
            var selectedPro = procedureList.SelectedItem.ToString();
            var procedure = CTUCareDB.Instance.Procedures.Where(x => x.Name ==
selectedPro).FirstOrDefault().Cost;
            double money = Convert.ToDouble(procedure);

            txtProPrice.Text = "R" + money.ToString();
        }

        private void loadMediPrice()
        {
            var selectedMed = medicineList.SelectedItem.ToString();
            var medicine = CTUCareDB.Instance.Medicines.Where(x => x.Name ==
selectedMed).FirstOrDefault().Price;
            double money = Convert.ToDouble(medicine);

```

```
        txtMediPrice.Text = "R" + money.ToString();
    }

    private void loadStatements()
    {
        //If dispatch date is null it means that the appointment has not been
        closed yet and there will be no statement
        var statements = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
History.statementPatient).Where(x => x.DispatchDateID != null);

        foreach (var item in statements)
        {
            patientList.Items.Add(item.EntryDate.DateEntered);
        }
    }

    private void loadProcedures()
    {
        procedureList.Items.Clear();

        //Need to find the procedure with a matching entry date and patient ID
        var dateID = CTUCareDB.Instance.EntryDates.Where(x => x.DateEntered ==
date).FirstOrDefault().ID;
        var proID = CTUCareDB.Instance.AdministeredProcedures.Where(x =>
x.EntryDateID == dateID).Where(x => x.PatientID ==
History.statementPatient).FirstOrDefault().ProcedureID;
        var procedure = CTUCareDB.Instance.Procedures.Where(x => x.ID == proID);

        foreach (var item in procedure)
        {
            procedureList.Items.Add(item.Name);
        }

        procedureList.Items.Refresh();
    }

    private void loadDayIn()
    {
        txtDayIn.Text = date.ToShortDateString();
    }

    private void loadDayOut()
    {
        var entryDateID = CTUCareDB.Instance.EntryDates.Where(x => x.DateEntered
== date).FirstOrDefault().ID;
        var dispatchID = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
History.statementPatient).Where(x => x.IntakeDateID ==
entryDateID).FirstOrDefault().DispatchDateID;
        var dispatchD = CTUCareDB.Instance.DispatchDates.Where(x => x.ID ==
dispatchID).FirstOrDefault().DispatchDate1;
        DateTime day = (DateTime)dispatchD;

        txtDayOut.Text = day.ToShortDateString();
    }

    private void loadMedicines()
    {
        medicineList.Items.Clear();

        var dateID = CTUCareDB.Instance.EntryDates.Where(x => x.DateEntered ==
date).FirstOrDefault().ID;
```

```

        var medicineID = CTUCareDB.Instance.AdministeredMedicines.Where(x =>
x.DateAdministeredID == dateID).Where(x => x.PatientID ==
History.statementPatient).FirstOrDefault().MedicineID;
        var medicine = CTUCareDB.Instance.Medicines.Where(x => x.ID ==
medicineID);

        foreach (var item in medicine)
        {
            medicineList.Items.Add(item.Name);
        }

        medicineList.Items.Refresh();
    }

    private void loadDrsPrice()
    {
        var dateID = CTUCareDB.Instance.EntryDates.Where(x => x.DateEntered ==
date).FirstOrDefault().ID;
        var doctorID = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
History.statementPatient).Where(x => x.IntakeDateID == dateID).FirstOrDefault().DrID;
        var doctor = CTUCareDB.Instance.Doctors.Where(x => x.ID ==
doctorID).FirstOrDefault().ConsultationFee.ToString();
        double price = Convert.ToDouble(doctor);

        txtDrsConsultation.Text = "R" + price.ToString();
    }

    private void loadDr()
    {
        var dateID = CTUCareDB.Instance.EntryDates.Where(x => x.DateEntered ==
date).FirstOrDefault().ID;
        var doctorID = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
History.statementPatient).Where(x => x.IntakeDateID == dateID).FirstOrDefault().DrID;
        var doctor = CTUCareDB.Instance.Doctors.Where(x => x.ID ==
doctorID).FirstOrDefault().Name;

        txtDr.Text = doctor;
    }
#endregion

#region Event Methods
//Linked to the selected item in the statements list
public static DateTime date;
private void btnLoad_Click(object sender, RoutedEventArgs e)
{
    date = (DateTime)patientList.SelectedItem;

    loadProcedures();
    loadDrsPrice();
    loadMedicines();
    loadDayIn();
    loadDayOut();
    calculateTotal();
    loadDr();
}

private void btnProPrice_Click(object sender, RoutedEventArgs e)
{
    loadProPrice();
}

```

```
private void btnMediPrice_Click(object sender, RoutedEventArgs e)
{
    loadMediPrice();
}
#endregion

#region Total Calculation Methods
private void calculateTotal()
{
    double total = calculateProcedureTotal() + calculateMedicineTotal() +
calculateDrsTotal();

    txtTotalCost.Text = "R" + total.ToString();
}

private double calculateProcedureTotal()
{
    List<string> procedures = new List<string>();
    List<decimal> proPrices = new List<decimal>();

    foreach (var item in procedureList.Items)
    {
        procedures.Add(item.ToString());
    }

    foreach (var item in procedures)
    {
        proPrices.Add(CTUCareDB.Instance.Procedures.Where(x => x.Name ==
item).FirstOrDefault().Cost);
    }

    double procedureTotal = proPrices.Sum(x => Convert.ToDouble(x));
    return procedureTotal;
}

private double calculateMedicineTotal()
{
    List<string> medicines = new List<string>();
    List<decimal> medPrices = new List<decimal>();

    foreach (var item in medicineList.Items)
    {
        medicines.Add(item.ToString());
    }

    foreach (var item in medicines)
    {
        medPrices.Add(CTUCareDB.Instance.Medicines.Where(x => x.Name ==
item).FirstOrDefault().Price);
    }

    double medicineTotal = medPrices.Sum(x => Convert.ToDouble(x));
    return medicineTotal;
}

private double calculateDrsTotal()
{
    var dateID = CTUCareDB.Instance.EntryDates.Where(x => x.DateEntered ==
date).FirstOrDefault().ID;
    var doctorID = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
History.statementPatient).Where(x => x.IntakeDateID == dateID).FirstOrDefault().DrID;
```

```

        var doctor = CTUCareDB.Instance.Doctors.Where(x => x.ID ==
doctorID).FirstOrDefault().ConsultationFee.ToString();
        double price = Convert.ToDouble(doctor);
        return price;
    }
    #endregion
}
}

```

XAML

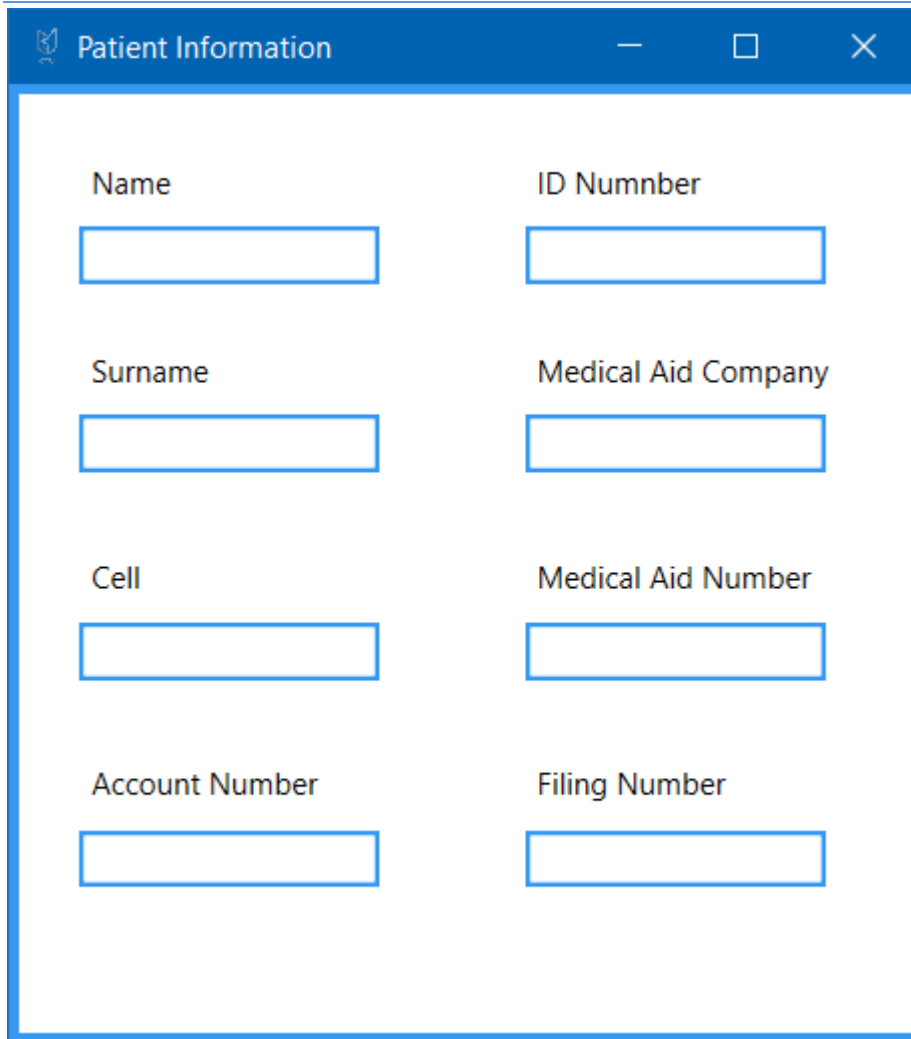
```

<Window x:Class="CTUCare.Accounts"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="Accounts and History" Height="411.733" Width="793.333"
    Icon="images/Picture5.png" BorderBrush="#FF3699F1" BorderThickness="4">
    <Grid>
        <ListBox x:Name="patientList" Margin="20,51,0,0" BorderBrush="#FF3699F1"
BorderThickness="1.5" Height="230" VerticalAlignment="Top" HorizontalAlignment="Left"
Width="150" />
        <Label x:Name="label" Content="Statement" HorizontalAlignment="Left"
Margin="20,25,0,0" Height="26" VerticalAlignment="Top" Width="150"/>
        <ListBox x:Name="procedureList" Margin="211,51,0,0" BorderBrush="#FF3699F1"
BorderThickness="1.5" Height="198" VerticalAlignment="Top" HorizontalAlignment="Left"
Width="150" />
        <Label x:Name="label_Copy" Content="Procedure" HorizontalAlignment="Left"
Margin="211,25,0,0" Height="26" VerticalAlignment="Top" Width="87"/>
        <TextBox Focusable="False" x:Name="txtProPrice" HorizontalAlignment="Left"
Height="23" Margin="211,313,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150" BorderThickness="2" BorderBrush="#FF3699F1"/>
        <ListBox x:Name="procedureList_Copy" Margin="397,51,0,0"
BorderBrush="#FF3699F1" BorderThickness="1.5" Height="198" VerticalAlignment="Top"
HorizontalAlignment="Left" Width="151"/>
        <Label x:Name="label_Copy2" Content="Medicines" HorizontalAlignment="Left"
Margin="397,25,0,0" Height="26" VerticalAlignment="Top" Width="151"/>
        <TextBox Focusable="False" x:Name="txtMediPrice" HorizontalAlignment="Left"
Height="23" Margin="397,313,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="151" BorderThickness="2" BorderBrush="#FF3699F1"/>
        <Label x:Name="label_Copy4" Content="Dr's Consultation"
HorizontalAlignment="Left" Margin="591,26,0,0" Height="26" VerticalAlignment="Top"
Width="150"/>
        <ListBox x:Name="medicineList" Margin="397,51,0,0" BorderBrush="#FF3699F1"
BorderThickness="1.5" Height="198" VerticalAlignment="Top" HorizontalAlignment="Left"
Width="151" />
        <TextBox Focusable="False" x:Name="txtDrsConsultation"
HorizontalAlignment="Left" Height="23" Margin="591,52,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="150" BorderThickness="2" BorderBrush="#FF3699F1"/>
        <Label x:Name="label_Copy5" Content="Day In" HorizontalAlignment="Left"
Margin="591,158,0,0" Height="26" VerticalAlignment="Top" Width="150"/>
        <TextBox Focusable="False" x:Name="txtDayIn" HorizontalAlignment="Left"
Height="23" Margin="591,184,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150" BorderThickness="2" BorderBrush="#FF3699F1"/>
        <Label x:Name="label_Copy6" Content="Day Out" HorizontalAlignment="Left"
Margin="591,225,0,0" Height="26" VerticalAlignment="Top" Width="150"/>
        <TextBox Focusable="False" x:Name="txtDayOut" HorizontalAlignment="Left"
Height="23" Margin="591,251,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150" BorderThickness="2" BorderBrush="#FF3699F1"/>
    
```

```
<Label x:Name="label_Copy7" Content="Total" HorizontalAlignment="Left"
Margin="591,279,0,0" Height="26" VerticalAlignment="Top" Width="150"/>
<TextBox Focusable="False" x:Name="txtTotalCost" HorizontalAlignment="Left"
Height="23" Margin="591,314,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150" BorderThickness="2" BorderBrush="#FF3699F1"/>
<Button x:Name="btnLoad" Content="Load" HorizontalAlignment="Left"
Margin="20,314,0,0" VerticalAlignment="Top" Width="150" Height="23"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White" Click="btnLoad_Click"/>
<Button x:Name="btnProPrice" Content="See Price" HorizontalAlignment="Left"
Margin="211,258,0,0" VerticalAlignment="Top" Width="150" Height="23"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White"
Click="btnProPrice_Click" />
<Button x:Name="btnMediPrice" Content="See Price" HorizontalAlignment="Left"
Margin="397,258,0,0" VerticalAlignment="Top" Width="151" Height="23"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White"
Click="btnMediPrice_Click" />
<Label x:Name="label_Copy1" Content="Dr" HorizontalAlignment="Left"
Margin="591,92,0,0" Height="26" VerticalAlignment="Top" Width="150"/>
<TextBox x:Name="txtDr" HorizontalAlignment="Left" Height="23"
Margin="591,118,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="150"
BorderThickness="2" BorderBrush="#FF3699F1" Focusable="False"/>

</Grid>
</Window>
```


Information



The screenshot shows a Windows-style application window titled "Patient Information". Inside the window, there are eight text input fields arranged in a 4x2 grid. The labels for the fields are: Name, ID Numnber, Surname, Medical Aid Company, Cell, Medical Aid Number, Account Number, and Filing Number. Each label is positioned above its corresponding text box.

C#

```
using System.Linq;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for Information.xaml
    /// </summary>
    public partial class Information : Window
    {
        public Information()
        {
            InitializeComponent();

            loadFields();
        }

        private void loadFields()
        {
            txtName.Text = CTUCareDB.Instance.Patients.Where(x => x.ID ==
PatientToken.patientID).FirstOrDefault().Name;
```

```

        txtSurname.Text = CTUCareDB.Instance.Patients.Where(x => x.ID ==
PatientToken.patientID).FirstOrDefault().Surname;
        txtIDNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID ==
PatientToken.patientID).FirstOrDefault().IDNumber;
        txtMedicalAidNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID ==
PatientToken.patientID).FirstOrDefault().MedicalAidNo;
        txtAccNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID ==
PatientToken.patientID).FirstOrDefault().AccountNumber.ToString();
        txtCell.Text = CTUCareDB.Instance.Patients.Where(x => x.ID ==
PatientToken.patientID).FirstOrDefault().Cell;
        txtFileNo.Text = PatientToken.patientID.ToString();

        var medicID = CTUCareDB.Instance.Patients.Where(x => x.ID ==
PatientToken.patientID).FirstOrDefault().MedicalAidID;
        txtMedicalAidCo.Text = CTUCareDB.Instance.MedicalAids.Where(x => x.ID ==
medicID).FirstOrDefault().Name;

        if (UserType.UserKind == 2)
        {
            txtAccNo.Visibility = Visibility.Collapsed;
            lblAccNo.Visibility = Visibility.Collapsed;

            txtCell.Visibility = Visibility.Collapsed;
            lblCell.Visibility = Visibility.Collapsed;

            txtMedicalAidCo.Visibility = Visibility.Collapsed;
            lblMedicalAidCo.Visibility = Visibility.Collapsed;
        }
    }
}
}

```

XAML

```

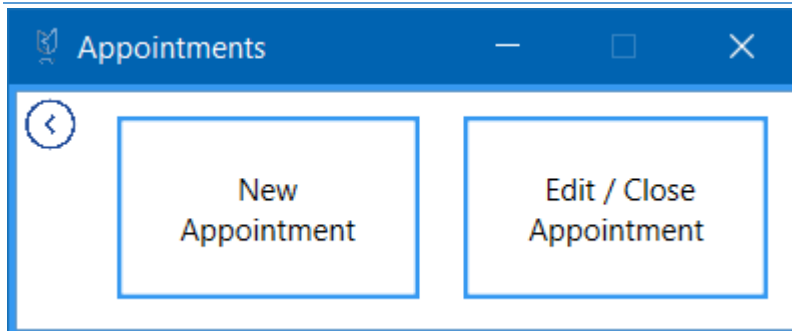
<Window x:Class="CTUCare.Information"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="Patient Information" Height="420.4" Width="380"
    Icon="images/Picture5.png" BorderBrush="#FF3699F1" BorderThickness="4">
    <Grid>
        <TextBox x:Name="txtName" HorizontalAlignment="Left" Height="23"
Margin="24,53,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <Label x:Name="label" Content="Name" HorizontalAlignment="Left"
Margin="24,22,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtSurname" HorizontalAlignment="Left" Height="23"
Margin="24,128,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <Label x:Name="label_Copy" Content="Surname" HorizontalAlignment="Left"
Margin="24,97,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtCell" HorizontalAlignment="Left" Height="23"
Margin="24,211,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <TextBox x:Name="txtAccNo" HorizontalAlignment="Left" Height="23"
Margin="24,294,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <Label x:Name="lblCell" Content="Cell" HorizontalAlignment="Left"
Margin="24,180,0,0" VerticalAlignment="Top"/>
    
```

```

        <TextBox x:Name="txtIDNo" HorizontalAlignment="Left" Height="23"
Margin="202,53,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <Label x:Name="label_Copy2" Content="ID Numnber" HorizontalAlignment="Left"
Margin="202,22,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtMedicalAidCo" HorizontalAlignment="Left" Height="23"
Margin="202,128,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <Label x:Name="lblMedicalAidCo" Content="Medical Aid Company"
HorizontalAlignment="Left" Margin="202,97,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtMedicalAidNo" HorizontalAlignment="Left" Height="23"
Margin="202,211,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <Label x:Name="label_Copy4" Content="Medical Aid Number"
HorizontalAlignment="Left" Margin="202,180,0,0" VerticalAlignment="Top"/>
        <Label x:Name="lblAccNo" Content="Account Number" HorizontalAlignment="Left"
Margin="24,262,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtFileNo" HorizontalAlignment="Left" Height="23"
Margin="202,294,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"
BorderBrush="#FF3699F1" Background="White" BorderThickness="2"/>
        <Label x:Name="label_Copy6" Content="Filing Number" HorizontalAlignment="Left"
Margin="202,262,0,0" VerticalAlignment="Top"/>
    </Grid>
</Window>

```

Appointments



C#

```

using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for Appointments.xaml
    /// </summary>
    public partial class Appointments : Window
    {
        public Appointments()
        {
            InitializeComponent();

            NewAppointment newApp = new NewAppointment();

            private void btnNewAppointment_Click(object sender, RoutedEventArgs e)

```

```

    {
        NewAppointment newApp = new NewAppointment();
        newApp.Show();

        this.Close();
    }

    private void btnCloseAppointment_Click(object sender, RoutedEventArgs e)
    {
        OpenAppointments openApp = new OpenAppointments();
        openApp.Show();

        this.Close();
    }

    private void btnBack_Click(object sender, RoutedEventArgs e)
    {
        Home goBack = new Home();
        goBack.Show();
        this.Close();
    }
}

```

XAML

```

<Window x:Class="CTUCare.Appointments"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="Appointments" Height="138.462" Width="331.49"
    Icon="images/Picture5.png" BorderBrush="#FF3699F1" BorderThickness="3"
    ResizeMode="CanMinimize">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="178*"/>
            <ColumnDefinition Width="141*"/>
        </Grid.ColumnDefinitions>
        <Button x:Name="btnNewAppointment" Content="New &#xA;Appointment"
            Margin="40,10,12.6,12.4" Background="White" Foreground="Black" BorderBrush="#FF3699F1"
            BorderThickness="2" Click="btnNewAppointment_Click"/>
        <Button x:Name="btnCloseAppointment" Content="Edit / Close &#xA;Appointment"
            Margin="5.4,10,10.4,12.4" Background="White" BorderBrush="#FF3699F1"
            BorderThickness="2" Grid.Column="1" Click="btnCloseAppointment_Click"/>
        <Button x:Name="btnBack" Content="Button" Foreground="{x:Null}"
            BorderBrush="{x:Null}" Margin="0,2,0,0" Click="btnBack_Click"
            HorizontalAlignment="Left" Width="27" Height="22" VerticalAlignment="Top">
            <Button.Background>
                <ImageBrush ImageSource="Images/Picture1.png" Stretch="Uniform"/>
            </Button.Background>
        </Button>
    </Grid>
</Window>

```

New Appointment

C#

```

using System;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for NewAppointment.xaml
    /// </summary>
    public partial class NewAppointment : Window
    {
        public NewAppointment()
        {
            InitializeComponent();

            populatePatientList();
            populateDrBox();
            populateProcedureList();
        }

        private void populatePatientList()
        {
            Update.updatePatients();
            patientList.Items.Clear();
        }
    }
}

```

```
        foreach (var item in Update.PatientList)
        {
            patientList.Items.Add(item.Name);
        }

        patientList.Items.Refresh();
    }

    private void populateDrBox()
    {
        Update.updateDoctors();
        cBoxDr.Items.Clear();

        foreach (var item in Update.DoctorList)
        {
            cBoxDr.Items.Add(item.Name);
        }

        cBoxDr.Items.Refresh();
    }

    private void populateProcedureList()
    {
        Update.updateProcedures();
        cBoxProcedure.Items.Clear();

        foreach (var item in Update.ProcedureList)
        {
            cBoxProcedure.Items.Add(item.Name);
        }

        cBoxProcedure.Items.Refresh();
    }

    private void search()
    {
        if (searchBox.Text == "")
        {
            populatePatientList();
        }
        else
        {
            patientList.Items.Clear();

            foreach (var item in Update.PatientList)
            {
                if (item.Name.ToLower().Contains(searchBox.Text.ToLower()))
                {
                    patientList.Items.Add(item.Name);
                }
            }
        }
    }

    #region Event Methods
    CTUCareDB db = new CTUCareDB();

    //Enables the dispatch options
    private void checkBox_Checked(object sender, RoutedEventArgs e)
    {
        lblDispatch.IsEnabled = false;
        dateDispatch.IsEnabled = false;
    }
}
```

```
}

private void checkBox_Unchecked(object sender, RoutedEventArgs e)
{
    lblDispatch.IsEnabled = true;
    dateDispatch.IsEnabled = true;
}

NewDr newDoc = new NewDr();

private void btnNewDr_Click(object sender, RoutedEventArgs e)
{
    try
    {
        newDoc.Show();
    }
    catch (Exception)
    {
        NewDr newDoc = new NewDr();
        newDoc.Show();
    }
}

NewPatient newPat = new NewPatient();

private void btnNewPatient_Click(object sender, RoutedEventArgs e)
{
    try
    {
        newPat.Show();
    }
    catch
    {
        NewPatient newPat = new NewPatient();
        newPat.Show();
    }
}

NewProcedure newPro = new NewProcedure();

private void btnNewProcedure_Click(object sender, RoutedEventArgs e)
{
    try
    {
        newPro.Show();
    }
    catch (Exception)
    {
        NewProcedure newPro = new NewProcedure();
        newPro.Show();
    }
}

private void searchBox_TextChanged(object sender, TextChangedEventArgs e)
{
    search();
}

private void searchBox_GotFocus(object sender, RoutedEventArgs e)
{
    searchBox.Foreground = Brushes.Black;
}
```

```
        searchBox.Text = "";
    }

    private void btnSaveAppointment_Click(object sender, RoutedEventArgs e)
    {
        try
        {
            var patient = patientList.SelectedItem.ToString();
            var appPatient = CTUCareDB.Instance.Patients.Where(x => x.Name ==
patient).FirstOrDefault();

            var doctor = cBoxDr.SelectedItem.ToString();
            var appDoctor = CTUCareDB.Instance.Doctors.Where(x => x.Name ==
doctor).FirstOrDefault();

            var procedure = cBoxProcedure.SelectedItem.ToString();
            var appProcedure = CTUCareDB.Instance.Procedures.Where(x => x.Name ==
procedure).FirstOrDefault();

            var intakeDate = dateIntake.SelectedDate;

            var dispatchDate = dateDispatch.SelectedDate;

            EntryDate loadNewEntry = new EntryDate();
            Appointment loadNewApp = new Appointment();
            AdministeredProcedure loadNewPro = new AdministeredProcedure();

            loadNewEntry.PatientID = appPatient.ID;
            loadNewEntry.DateEntered = intakeDate;

            db.EntryDates.Add(loadNewEntry);
            db.SaveChanges();

            //Done after entry dates were added so that an ID can be assigned
            var entryDate = CTUCareDB.Instance.EntryDates.Where(x => x.PatientID
== loadNewEntry.PatientID).ToList();
            var entryDateID = entryDate.Last().ID;

            loadNewApp.PatientID = appPatient.ID;
            loadNewApp.IntakeDateID = entryDateID;
            loadNewApp.DrID = appDoctor.ID;

            //Only adds if the checkbox is unchecked
            if (checkBox.IsChecked == false)
            {
                loadNewApp.ScheduledDispatchDate = (DateTime)dispatchDate;
            }

            db.Appointments.Add(loadNewApp);
            db.SaveChanges();

            //Done after appointments to generate an ID
            loadNewPro.PatientID = appPatient.ID;
            loadNewPro.ProcedureID = appProcedure.ID;
            loadNewPro.EntryDateID = entryDateID;

            db.AdministeredProcedures.Add(loadNewPro);

            db.SaveChanges();
        }
        catch (Exception)
```



```

        {
            MessageBox.Show("One or more fields still require entries, refer to
user manual for more details");
        }

        Appointments goBack = new Appointments();
        goBack.Show();
        this.Close();
    }

    private void cBoxDr_DropDownOpened(object sender, EventArgs e)
    {
        populateDrBox();
    }

    private void btnReloadPatients_Click(object sender, RoutedEventArgs e)
    {
        populatePatientList();
    }
#endregion

    private void btnBack_Click(object sender, RoutedEventArgs e)
    {
        Appointments goBack = new Appointments();
        goBack.Show();
        this.Close();
    }
}
}

```

XAML

```

<Window x:Class="CTUCare.NewAppointment"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="New Appointment" Height="428.8" Width="568.8"
    Icon="images/Picture5.png" BorderBrush="#FF3699F1" BorderThickness="3">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="73*"/>
            <RowDefinition Height="150*"/>
            <RowDefinition Height="79*"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="162*"/>
            <ColumnDefinition Width="154*"/>
            <ColumnDefinition Width="40*"/>
            <ColumnDefinition Width="125*"/>
            <ColumnDefinition Width="77*"/>
        </Grid.ColumnDefinitions>
        <ListBox x:Name="patientList" Margin="34,1.8,9.6,0" BorderBrush="#FF3699F1"
BorderThickness="1.5" Grid.Row="1" Height="202" VerticalAlignment="Top"
Grid.RowSpan="2"/>
        <TextBox x:Name="searchBox" Margin="34,0,9.6,3.2" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="1.5" Height="19" VerticalAlignment="Bottom"
TextChanged="searchBox_TextChanged" Text="Search" GotFocus="searchBox_GotFocus"
Foreground="Gray" />
    </Grid>

```

```

        <Rectangle x:Name="logo" Height="60" Margin="27,10,10.8,0"
VerticalAlignment="Top" RenderTransformOrigin="0.44,0.447" Grid.Column="4">
            <Rectangle.Fill>
                <ImageBrush ImageSource="Images/Picture5.png" Stretch="Uniform"/>
            </Rectangle.Fill>
        </Rectangle>
        <Label x:Name="label" Content="Patient" HorizontalAlignment="Left"
Margin="29,0,0,21.8" Height="26" VerticalAlignment="Bottom"/>
        <Label x:Name="label_Copy" Content="Doctor" HorizontalAlignment="Left"
Margin="15.4,0,0,21.8" Grid.Column="1" Height="26" VerticalAlignment="Bottom"/>
        <ComboBox x:Name="cBoxDr" Margin="15.4,0,29.2,2.8" BorderBrush="#FF3699F1"
Background="#FF3699F1" Grid.Column="1" Height="19" VerticalAlignment="Bottom"
DropDownOpened="cBoxDr_DropDownOpened"/>
        <Label x:Name="label_Copy1" Content="Procedure" HorizontalAlignment="Left"
Margin="21.8,0,0,21.8" Grid.Column="2" Grid.ColumnSpan="2" Height="26"
VerticalAlignment="Bottom"/>
        <ComboBox x:Name="cBoxProcedure" Margin="21.8,0,34,2.8" BorderBrush="White"
Grid.Column="2" Grid.ColumnSpan="2" Height="19" VerticalAlignment="Bottom"/>
        <DatePicker x:Name="dateIntake" Margin="58.8,62.2,34,0"
VerticalAlignment="Top" Height="26" BorderBrush="#FF3699F1" BorderThickness="2"
Grid.Column="3" Grid.Row="1"/>
        <Label x:Name="label_Copy2" Content="Scheduled Intake Day"
HorizontalAlignment="Left" Margin="33.4,62,0,0" VerticalAlignment="Top"
Grid.Column="1" Grid.ColumnSpan="2" Grid.Row="1"/>
        <DatePicker x:Name="dateDispatch" Margin="58.8,106.2,34,0"
BorderBrush="#FF3699F1" BorderThickness="2"
            IsEnabled="True" Grid.Column="3" Grid.Row="1" Height="26"
VerticalAlignment="Top"/>
        <Label x:Name="lblDispatch" Content="Scheduled Dispatch Day"
HorizontalAlignment="Left" Margin="33.4,105.2,0,0" VerticalAlignment="Top" Width="140"
            IsEnabled="True" Grid.Column="1" Grid.ColumnSpan="2" Grid.Row="1"/>
        <CheckBox x:Name="checkBox" Content="Appointment Only" Margin="109.4,29.2,0,0"
VerticalAlignment="Top" Checked="checkBox_Checked" Unchecked="checkBox_Unchecked"
Grid.Column="1" Grid.ColumnSpan="3" Grid.Row="1" HorizontalAlignment="Left"
Width="118"/>
        <Button x:Name="btnNewPatient" Content="New Patient" Margin="34,40.6,2.6,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White" Grid.Row="2"
Height="25" VerticalAlignment="Top" Click="btnNewPatient_Click"/>
        <Button x:Name="btnNewProcedure" Content="New Procedure"
Margin="48.4,40.6,20.2,0" BorderBrush="#FF3699F1" BorderThickness="2"
Background="White" Grid.Column="1" Grid.ColumnSpan="2" Grid.Row="2"
Click="btnNewProcedure_Click" Height="25" VerticalAlignment="Top"/>
        <Button x:Name="btnNewDr" Content="New Doctor" Margin="30.8,40.6,45.8,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White" Grid.Column="3"
Grid.ColumnSpan="2" Grid.Row="2" Click="btnNewDr_Click" Height="25"
VerticalAlignment="Top"/>
        <Button x:Name="btnSaveAppointment" Content="Save Appointment"
Margin="39.4,168.8,34,0" BorderBrush="#FF3699F1" BorderThickness="2"
Background="#FF3699F1" Grid.ColumnSpan="3" Grid.Row="1" Height="35"
VerticalAlignment="Top" Foreground="White" Grid.Column="1" Grid.RowSpan="2"
Click="btnSaveAppointment_Click"/>
        <Button x:Name="btnReloadPatients" Content="" HorizontalAlignment="Left"
Height="24" Margin="122,176.8,0,0" Grid.Row="1" VerticalAlignment="Top" Width="27"
BorderBrush="{x:Null}" Foreground="{x:Null}" Grid.RowSpan="2"
Click="btnReloadPatients_Click">
            <Button.Background>
                <ImageBrush ImageSource="Images/reload_icon.jpg" Stretch="Uniform"/>
            </Button.Background>
        </Button>
        <Button x:Name="btnBack" Content="Button" Foreground="{x:Null}"
BorderBrush="{x:Null}" Margin="-5,2,0,73.2" Click="btnBack_Click"
HorizontalAlignment="Left" Width="32">

```

```

        <Button.Background>
            <ImageBrush ImageSource="Images/Picture1.png" Stretch="Uniform"/>
        </Button.Background>
    </Button>
</Grid>
</Window>

```

Open Appointments

C#

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Windows;
```

```
namespace CTUCare
```

```
{
```

```
    /// <summary>
```

```
    /// Interaction logic for OpenAppointments.xaml
```

```
    /// </summary>
```

```
    public partial class OpenAppointments : Window
```

```
    {
```

```
        public OpenAppointments()
```

```
        {
```

```
            InitializeComponent();
```

```
        updateProcedureBox();

        updatePatientsAppointments();

        updateMedicineBox();

        updateDrs();
    }

    public int currentPatient;

    private void setDate()
    {
        var intakeDatum = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
currentPatient).FirstOrDefault().IntakeDateID;

        var selectedDate = CTUCareDB.Instance.EntryDates.Where(x => x.ID ==
intakeDatum).FirstOrDefault().DateEntered;

        txtCurrentDay.Text = selectedDate.ToString();

        dateIntake.SelectedDate = selectedDate;
    }

    private void goBack()
    {
        Appointments goBack = new Appointments();
        goBack.Show();
        this.Close();
    }

    private void saveNewDate()
    {
```

```
var intakeDatum = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
currentPatient).FirstOrDefault().IntakeDateID;
```

```
DispatchDate newExitDate = new DispatchDate();
```

```
newExitDate.PatientID = currentPatient;
```

```
newExitDate.DispatchDate1 = DateTime.Now;
```

```
CTUCareDB.Instance.DispatchDates.Add(newExitDate);
```

```
CTUCareDB.Instance.SaveChanges();
```

```
var lookupDate = CTUCareDB.Instance.DispatchDates.Where(x => x.PatientID ==
currentPatient).ToList();
```

```
CTUCareDB.Instance.Appointments.Where(x => x.EntryDate.ID == intakeDatum).Where(x =>
x.PatientID == currentPatient).FirstOrDefault().DispatchDateID = lookupDate.LastOrDefault().ID;
```

```
CTUCareDB.Instance.SaveChanges();
```

```
}
```

```
#region Update/Loading Methods
```

```
public void updateProcedureBox()
```

```
{
```

```
Update.updateProcedures();
```

```
cboxProcedures.Items.Clear();
```

```
foreach (var item in Update.ProcedureList)
```

```
{
```

```
cboxProcedures.Items.Add(item.Name);
```

```
}
```

```
        cboxProcedures.Items.Refresh();
    }

    public void updateMedicineBox()
    {
        Update.updateMedicines();

        cboxMedicines.Items.Clear();

        foreach (var item in Update.MedicineList)
        {
            cboxMedicines.Items.Add(item.Name);
        }

        cboxMedicines.Items.Refresh();
    }


    private void updateDrs()
    {
        cboxDrs.Items.Clear();

        Update.updateDoctors();

        foreach (var item in Update.DoctorList)
        {
            cboxDrs.Items.Add(item.Name);
        }

        cboxDrs.Items.Refresh();
    }
```

```
}

private void updateDate()
{
    var intakeDatum = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
currentPatient).FirstOrDefault().IntakeDateID;

    CTUCareDB.Instance.EntryDates.Remove(CTUCareDB.Instance.EntryDates.Where(x => x.ID ==
intakeDatum).FirstOrDefault());

    EntryDate updatedDate = new EntryDate();

    updatedDate.ID = (int)intakeDatum;
    updatedDate.PatientID = currentPatient;
    updatedDate.DateEntered = dateIntake.SelectedDate;

    CTUCareDB.Instance.EntryDates.Add(updatedDate);

}

private void updatePatientsAppointments()
{
    List<Appointment> openAppointments = new List<Appointment>();

    var Appointments = CTUCareDB.Instance.Appointments.Where(x => x.DispatchDate == null);

    foreach (var item in Appointments)
    {
        openAppointments.Add(item);
    }

    foreach (var item in openAppointments)
```

```
{
    listPatients.Items.Add(item.Patient.Name);
}
}

private void loadDoctor()
{
    var intakeDatum = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
currentPatient).FirstOrDefault().IntakeDateID;

    txtCurrentDoc.Text = CTUCareDB.Instance.Appointments.Where(x => x.IntakeDateID ==
intakeDatum).FirstOrDefault().Doctor.Name;
}

private void updateMedicinesList()
{
    try
    {
        listMedicinesAdmin.Items.Clear();

        List<AdministeredMedicine> appliedMedicines = new List<AdministeredMedicine>();

        var appMedicines = CTUCareDB.Instance.AdministeredMedicines.Where(x => x.PatientID ==
currentPatient).FirstOrDefault().DateAdministeredID;

        var currentMeds = CTUCareDB.Instance.AdministeredMedicines.Where(x =>
x.DateAdministeredID == appMedicines).ToList();

        foreach (var item in currentMeds)
        {
            listMedicinesAdmin.Items.Add(item.Medicine.Name);
        }

        listMedicinesAdmin.Items.Refresh();
    }
    catch (Exception)
```



```
{  
  
}  
}  
  
private void updateProcedureList()  
{  
    listProceduresAdmin.Items.Clear();  
  
    List<AdministeredProcedure> appliedProcedures = new List<AdministeredProcedure>();  
  
    var appProcedures = CTUCareDB.Instance.AdministeredProcedures.Where(x => x.PatientID ==  
currentPatient).FirstOrDefault().EntryDateID;  
  
    var currentProcedures = CTUCareDB.Instance.AdministeredProcedures.Where(x =>  
x.EntryDateID == appProcedures).ToList();  
  
    foreach (var item in currentProcedures)  
    {  
        listProceduresAdmin.Items.Add(item.Procedure.Name);  
    }  
  
    listProceduresAdmin.Items.Refresh();  
}  
  
private void saveProcedures()  
{  
    var appProcedures = CTUCareDB.Instance.AdministeredProcedures.Where(x => x.PatientID ==  
currentPatient).FirstOrDefault().EntryDateID;  
  
    var currentProcedures = CTUCareDB.Instance.AdministeredProcedures.Where(x =>  
x.EntryDateID == appProcedures).ToList();  
  
    foreach (var item in currentProcedures)
```

```
{
    Update.AdminProcedureList.Remove(item);
}
}
#endregion

#region Event Methods

private void btnChangeDoctor_Click(object sender, RoutedEventArgs e)
{
    var intakeDatum = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
currentPatient).FirstOrDefault().IntakeDateID;

    CTUCareDB.Instance.Appointments.Where(x => x.IntakeDateID ==
intakeDatum).FirstOrDefault().DrID =

    CTUCareDB.Instance.Doctors.Where(x => x.Name ==
cboxDrs.SelectedItem.ToString()).FirstOrDefault().ID;

    txtCurrentDoc.Text = cboxDrs.SelectedItem.ToString();
}

private void btnCloseApp_Click(object sender, RoutedEventArgs e)
{
    saveNewDate();

    goBack();
}

private void btnChangeDate_Click(object sender, RoutedEventArgs e)
{
    updateDate();

    txtCurrentDay.Text = dateIntake.SelectedDate.ToString();
}
```

```
private void btnSave_Click(object sender, RoutedEventArgs e)
{
    CTUCareDB.Instance.SaveChanges();

    goBack();
}

private void btnLoadAppointment_Click(object sender, RoutedEventArgs e)
{
    currentPatient = CTUCareDB.Instance.Patients.Where(x => x.Name ==
listPatients.SelectedItem.ToString()).FirstOrDefault().ID;

    updateMedicinesList();
    updateProcedureList();
    loadDoctor();
    setDate();
}

NewMedicine newMed = new NewMedicine();

private void btnNewMedicine_Click(object sender, RoutedEventArgs e)
{
    try
    {
        newMed.Show();
    }
    catch (Exception)
    {
        NewMedicine newMed = new NewMedicine();
        newMed.Show();
    }
}
```

```
    }  
}
```

```
NewProcedure newPro = new NewProcedure();
```

```
private void btnNewProcedure_Click(object sender, RoutedEventArgs e)
```

```
{  
    try  
    {  
        newPro.Show();  
    }  
    catch (Exception)  
    {  
        NewProcedure newPro = new NewProcedure();  
        newPro.Show();  
    }  
}
```

```
private void btnAddProcedure_Click(object sender, RoutedEventArgs e)
```

```
{  
    listProceduresAdmin.Items.Add(cboxProcedures.SelectedItem);
```

```
AdministeredProcedure newAdminPro = new AdministeredProcedure();
```

```
var intakeDatum = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==  
currentPatient).FirstOrDefault().IntakeDateID;
```

```
newAdminPro.PatientID = currentPatient;
```

```
newAdminPro.ProcedureID = CTUCareDB.Instance.Procedures.Where(x => x.Name ==  
cboxProcedures.SelectedItem.ToString()).FirstOrDefault().ID;
```

```
newAdminPro.EntryDateID = intakeDatum;
```

```
CTUCareDB.Instance.AdministeredProcedures.Add(new AdminPro);  
}
```

```
private void btnRemoveProcedure_Click(object sender, RoutedEventArgs e)  
{  
    var selected = listProceduresAdmin.SelectedItem;
```

```
    listProceduresAdmin.Items.Remove(listProceduresAdmin.SelectedItem);
```

```
CTUCareDB.Instance.AdministeredProcedures.Remove(CTUCareDB.Instance.AdministeredProcedures.Where
```

```
(x => x.Procedure.Name == selected.ToString()).FirstOrDefault());  
}
```

```
private void cboxProcedures_DropDownOpened(object sender, EventArgs e)  
{  
    updateProcedureBox();
```

```
    cboxProcedures.Items.Refresh();  
}
```

```
private void cboxMedicines_DropDownOpened(object sender, EventArgs e)  
{  
    updateMedicineBox();
```

```
    cboxMedicines.Items.Refresh();  
}
```

```
private void btnAddMedicine_Click(object sender, RoutedEventArgs e)  
{
```

```
listMedicinesAdmin.Items.Add(cboxMedicines.SelectedItem);

AdministeredMedicine newAdminMedicine = new AdministeredMedicine();

var intakeDatum = CTUCareDB.Instance.Appointments.Where(x => x.PatientID ==
currentPatient).FirstOrDefault().IntakeDateID;

newAdminMedicine.PatientID = currentPatient;

newAdminMedicine.MedicineID = CTUCareDB.Instance.Medicines.Where(x => x.Name ==
cboxMedicines.SelectedItem.ToString()).FirstOrDefault().ID;

newAdminMedicine.DateAdministeredID = intakeDatum;

CTUCareDB.Instance.AdministeredMedicines.Add(newAdminMedicine);

}

private void btnRemoveMedicine_Click(object sender, RoutedEventArgs e)
{
    var selected = listMedicinesAdmin.SelectedItem;

    CTUCareDB.Instance.AdministeredMedicines.Remove(CTUCareDB.Instance.AdministeredMedicines.
Where
    (x => x.Medicine.Name == selected.ToString()).FirstOrDefault());

    listMedicinesAdmin.Items.Remove(listMedicinesAdmin.SelectedItem);
}

private void cboxDrs_DropDownOpened(object sender, EventArgs e)
{
    updateDrs();
}
```

#endregion

```
private void btnBack_Click(object sender, RoutedEventArgs e)
{
    Appointments goBack = new Appointments();
    goBack.Show();
    this.Close();
}
}
```

XAML

```
<Window x:Class="CTUCare.OpenAppointments"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="Open Appointment" Height="406.767" Width="858.773"
    Icon="images/Picture5.png">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
        </Grid.ColumnDefinitions>
        <ListBox x:Name="listProceduresAdmin" Margin="234,52,0,0"
HorizontalAlignment="Left" Width="142" BorderBrush="#FF3699F1" BorderThickness="2"
Height="138" VerticalAlignment="Top"/>
        <ComboBox x:Name="cboxProcedures" Margin="234,223,0,0"
DropDownOpened="cboxProcedures_DropDownOpened" HorizontalAlignment="Left" Width="142"
Height="22" VerticalAlignment="Top" />
        <Button x:Name="btnAddProcedure" Content="Add Procedure" Margin="234,266,0,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White"
Click="btnAddProcedure_Click" HorizontalAlignment="Left" Width="142" Height="24"
VerticalAlignment="Top" />
        <Button x:Name="btnNewProcedure" Content="New Procedure" Margin="234,305,0,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White"
Click="btnNewProcedure_Click" HorizontalAlignment="Left" Width="142" Height="24"
VerticalAlignment="Top" />
        <Label x:Name="label" Content="Procedures Administered" Margin="234,21,0,0"
VerticalAlignment="Top" Height="26" HorizontalAlignment="Left" Width="142"/>
        <Label x:Name="label_Copy2" Content="Intake Day" Margin="0,19,126,0"
VerticalAlignment="Top" Height="26" HorizontalAlignment="Right" Width="88"/>
        <ListBox x:Name="listMedicinesAdmin" Margin="0,51,274,0"
HorizontalAlignment="Right" Width="142" BorderThickness="2" BorderBrush="#FF3699F1"
Height="138" VerticalAlignment="Top"/>
        <Label x:Name="label_Copy" Content="Medicines Administered"
Margin="0,20,278,0" VerticalAlignment="Top" Height="26" HorizontalAlignment="Right"
Width="138"/>
    </Grid>
</Window>
```

```

        <ComboBox x:Name="cboxMedicines" Margin="0,222,274,0"
HorizontalAlignment="Right" Width="142" DropDownOpened="cboxMedicines_DropDownOpened"
Height="22" VerticalAlignment="Top"/>
        <Button x:Name="btnAddMedicine" Content="Add Medicine" Margin="0,265,274,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White"
Click="btnAddMedicine_Click" HorizontalAlignment="Right" Width="142" Height="24"
VerticalAlignment="Top" />
        <Button x:Name="btnNewMedicine" Content="New Medicine" Margin="0,304,274,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White"
Click="btnNewMedicine_Click" HorizontalAlignment="Right" Width="142" Height="24"
VerticalAlignment="Top" />
        <DatePicker x:Name="dateIntake" Margin="0,87,179,0" VerticalAlignment="Top"
Height="26" BorderBrush="#FF3699F1" BorderThickness="2" HorizontalAlignment="Right"
Width="35"/>
        <ComboBox x:Name="cboxDrs" Margin="0,194,60,0" VerticalAlignment="Top"
Height="22" DropDownOpened="cboxDrs_DropDownOpened" HorizontalAlignment="Right"
Width="154"/>
        <Button x:Name="btnChangeDoctor" Content="Change Doctor" Margin="0,221,58,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White" Height="22"
VerticalAlignment="Top" HorizontalAlignment="Right" Width="156"
Click="btnChangeDoctor_Click" />
        <Button x:Name="btnSave" Content="Save" Margin="0,264,58,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="#FF3699F1" Foreground="White"
HorizontalAlignment="Right" Width="156" Height="24" VerticalAlignment="Top"
Click="btnSave_Click" />
        <Button x:Name="btnCloseApp" Content="Close Appointment" Margin="0,303,60,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="#FF3699F1" Foreground="White"
HorizontalAlignment="Right" Width="154" Height="24" VerticalAlignment="Top"
Click="btnCloseApp_Click" />
        <Label x:Name="label_Copy1" Content="Current Dr" Margin="0,139,118,0"
VerticalAlignment="Top" Height="26" HorizontalAlignment="Right" Width="98"/>
        <Button x:Name="btnRemoveProcedure" Content="-" HorizontalAlignment="Left"
Margin="355,162,0,0" VerticalAlignment="Top" Width="15" RenderTransformOrigin="-
0.455,0.396" BorderBrush="White" Background="White" Foreground="Red" FontWeight="Bold"
FontSize="20" Height="23" Click="btnRemoveProcedure_Click"/>
        <Button x:Name="btnRemoveMedicine" Content="-" HorizontalAlignment="Right"
Margin="0,161,279,0" VerticalAlignment="Top" Width="15" RenderTransformOrigin="-
0.455,0.396" BorderBrush="White" Background="White" Foreground="Red" FontWeight="Bold"
FontSize="20" Height="23" Click="btnRemoveMedicine_Click"/>
        <ListBox x:Name="listPatients" Margin="37,71,0,0" BorderBrush="#FF3699F1"
BorderThickness="1.5" Height="219" VerticalAlignment="Top" HorizontalAlignment="Left"
Width="138"/>
        <TextBox x:Name="searchBox" Margin="37,47,0,0" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="1.5" Height="19" VerticalAlignment="Top"
HorizontalAlignment="Left" Width="138" />
        <Label x:Name="label_Copy3" Content="Patient" Margin="32,21,0,0" Height="26"
VerticalAlignment="Top" HorizontalAlignment="Left" Width="46"/>
        <Button x:Name="btnLoadAppointment" Content="Load Appointment"
Margin="37,305,0,0" BorderBrush="#FF3699F1" BorderThickness="2" Background="White"
HorizontalAlignment="Left" Width="138" Click="btnLoadAppointment_Click" Height="24"
VerticalAlignment="Top" />
        <TextBox x:Name="txtCurrentDoc" Margin="0,170,60,0" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="1.5" Height="19" VerticalAlignment="Top"
Width="154" HorizontalAlignment="Right" Focusable="False"/>
        <TextBox x:Name="txtCurrentDay" Margin="0,50,60,0" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="1.5" Height="19" VerticalAlignment="Top"
Width="154" HorizontalAlignment="Right" Focusable="False"/>
        <Button x:Name="btnChangeDate" Content="Change" Margin="0,87,60,0"
BorderBrush="#FF3699F1" BorderThickness="2" Background="White" Height="26"
VerticalAlignment="Top" HorizontalAlignment="Right" Width="99"
Click="btnChangeDate_Click" />

```



```

        <Button x:Name="btnBack" Content="Button" Foreground="{x:Null}"
BorderBrush="{x:Null}" Margin="-5,2,0,354.8" Click="btnBack_Click"
HorizontalAlignment="Left" Width="32">
            <Button.Background>
                <ImageBrush ImageSource="Images/Picture1.png" Stretch="Uniform"/>
            </Button.Background>
        </Button>

    </Grid>
</Window>

```

New Patient

The screenshot shows a 'New Patient' form window. The form is organized into a grid with the following elements:

- Name**: Text input field
- Surname**: Text input field
- Cell No**: Text input field
- ID No**: Text input field
- Address**: Text input field
- Medical Aid No**: Text input field
- Medical Aid Co**: Dropdown menu
- Account No**: Text input field
- New Medical Aid Co**: Button
- Save New Patient**: Button

C#

```

using System;
using System.Linq;
using System.Windows;

```

```
namespace CTUCare
{
    /// <summary>
    /// Interaction logic for NewPatient.xaml
    /// </summary>
    public partial class NewPatient : Window
    {
        public NewPatient()
        {
            InitializeComponent();

            updateMedicCompanies();

            NewMedicalAidCo newMed = new NewMedicalAidCo();

            private void btnNewMedicalAid_Click(object sender, RoutedEventArgs e)
            {
                try
                {
                    newMed.Show();
                }
                catch (Exception)
                {
                    NewMedicalAidCo newMed = new NewMedicalAidCo();
                    newMed.Show();
                }

                updateMedicCompanies();
            }

            private void updateMedicCompanies()
            {
                Update.updateCompanies();

                cbxMedicalAidCo.Items.Clear();

                foreach (var item in Update.MedicalAidCoList)
                {
                    cbxMedicalAidCo.Items.Add(item.Name);
                }
            }

            private void cbxMedicalAidCo_DropDownOpened(object sender, EventArgs e)
            {
                updateMedicCompanies();

                cbxMedicalAidCo.Items.Refresh();
            }

            private void btnSaveNew_Click(object sender, RoutedEventArgs e)
            {
                try
                {
                    string name = txtName.Text;
                    string surname = txtSurname.Text;
                    string cellNo = txtCell.Text;
                    string idNo = txtIDNo.Text;
                    string address = txtAdd.Text;
                    string medicalAidNo = txtMedicalAidNo.Text;
                    string medicalAidCo = cbxMedicalAidCo.SelectedItem.ToString();
                    int accNo = Convert.ToInt32(txtAccNo.Text);
                }
            }
        }
    }
}
```

```

        Patient newPatient = new Patient
        {
            Name = name,
            Surname = surname,
            Cell = cellNo,
            IDNumber = idNo,
            Address = address,
            MedicalAidNo = medicalAidNo,
            MedicalAidID = CTUCareDB.Instance.MedicalAids.Where(x => x.Name ==
medicalAidCo).FirstOrDefault().ID,
            AccountNumber = accNo,
        };

        CTUCareDB.Instance.Patients.Add(newPatient);
        CTUCareDB.Instance.SaveChanges();

        this.Close();
    }
    catch (Exception)
    {
        MessageBox.Show("One or more values are still invalid or missing");
    }
}
}
}

```

XAML

```

<Window x:Class="CTUCare.NewPatient"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="New Patient" Height="483.6" Width="364.03"
    Icon="images/Picture5.png">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="93*"/>
            <RowDefinition Height="96*"/>
            <RowDefinition Height="91*"/>
            <RowDefinition Height="88*"/>
            <RowDefinition Height="85*"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="172*"/>
            <ColumnDefinition Width="185*"/>
        </Grid.ColumnDefinitions>
        <TextBox x:Name="txtName" Margin="29,47,23,22.8" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2"/>
        <TextBox x:Name="txtSurname" Margin="25,47,40.6,22.8" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Grid.Column="1"/>
        <TextBox x:Name="txtCell" Margin="29,48.2,23,24.8" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Grid.Row="1"/>
        <TextBox x:Name="txtIDNo" Margin="25,48.2,40.6,24.8" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Grid.Column="1" Grid.Row="1"/>
        <TextBox x:Name="txtAdd" Margin="29,48.2,23,20" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Grid.Row="2"/>
        <TextBox x:Name="txtMedicalAidNo" Margin="25,48.2,40.6,20" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Grid.Column="1" Grid.Row="2"/>
    </Grid>

```

```

        <Button x:Name="btnNewMedicalAid" Content="New Medical Aid Co"
Margin="25,29,40.6,26" Background="#FF3699F1" BorderBrush="#FF3699F1"
BorderThickness="2" Foreground="White" Grid.Column="1" Grid.Row="3"
Click="btnNewMedicalAid_Click"/>
        <Label x:Name="label_Copy" Content="Medical Aid Co" HorizontalAlignment="Left"
Margin="29,8,0,0" VerticalAlignment="Top" Grid.Row="3"/>
        <ComboBox x:Name="cboxMedicalAidCo" Margin="30,34,23,26" Grid.Row="3"
DropDownOpened="cboxMedicalAidCo_DropDownOpened"/>
        <Button x:Name="btnSaveNew" Content="Save New Patient"
Margin="24,36,41.6,21.6" Background="White" BorderBrush="#FF3699F1"
BorderThickness="2" Grid.Row="4" Click="btnSaveNew_Click" Grid.Column="1"/>
        <Label x:Name="label_Copy1" Content="Medical Aid No"
HorizontalAlignment="Left" Margin="25,17.2,0,0" VerticalAlignment="Top"
Grid.Column="1" Grid.Row="2"/>
        <Label x:Name="label_Copy2" Content="Address" HorizontalAlignment="Left"
Margin="29,17.2,0,0" VerticalAlignment="Top" Grid.Row="2"/>
        <Label x:Name="label_Copy3" Content="ID No" HorizontalAlignment="Left"
Margin="25,17.2,0,0" VerticalAlignment="Top" Grid.Column="1" Grid.Row="1"/>
        <Label x:Name="label_Copy4" Content="Cell No" HorizontalAlignment="Left"
Margin="29,17.2,0,0" VerticalAlignment="Top" Grid.Row="1"/>
        <Label x:Name="label_Copy5" Content="Surname" HorizontalAlignment="Left"
Margin="25,16,0,0" VerticalAlignment="Top" Grid.Column="1"/>
        <Label x:Name="label_Copy6" Content="Name" HorizontalAlignment="Left"
Margin="30,21,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtAccNo" Margin="29,41,23,21.6" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Grid.Row="4"/>
        <Label x:Name="label_Copy7" Content="Account No" HorizontalAlignment="Left"
Margin="29,15,0,0" VerticalAlignment="Top" Grid.Row="4"/>

    </Grid>
</Window>

```

New Procedure

The screenshot shows a standard Windows-style dialog box titled 'New Pro...'. It contains two text boxes for data entry. The first text box is preceded by the label 'Procedure Name'. The second text box is preceded by the label 'Price'. At the bottom of the dialog, centered, is a button with the text 'Save New Procedure'.

```

C#
using System;

```

```

using System.Linq;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for NewProcedure.xaml
    /// </summary>
    public partial class NewProcedure : Window
    {
        public NewProcedure()
        {
            InitializeComponent();

            CTUCareDB db = new CTUCareDB();

            private void btnSaveNew_Click(object sender, RoutedEventArgs e)
            {
                if (txtName.Text.Count() > 0)
                {
                    try
                    {
                        Procedure newPro = new Procedure
                        {
                            Name = txtName.Text,
                            Cost = Decimal.Parse(txtPrice.Text),
                        };

                        db.Procedures.Add(newPro);
                        db.SaveChanges();

                        Update.updateProcedures();

                        this.Close();
                    }
                    catch (Exception)
                    {
                        MessageBox.Show("Please enter a monetary value");
                    }
                }
                else
                {
                    MessageBox.Show("Please enter a valid name");
                }
            }
        }
    }
}

```

XAML

```

<Window x:Class="CTUCare.NewProcedure"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="New Procedure" Height="267.2" Width="240"
    Icon="images/Picture5.png"
    ResizeMode="CanMinimize">
    <Grid>

```

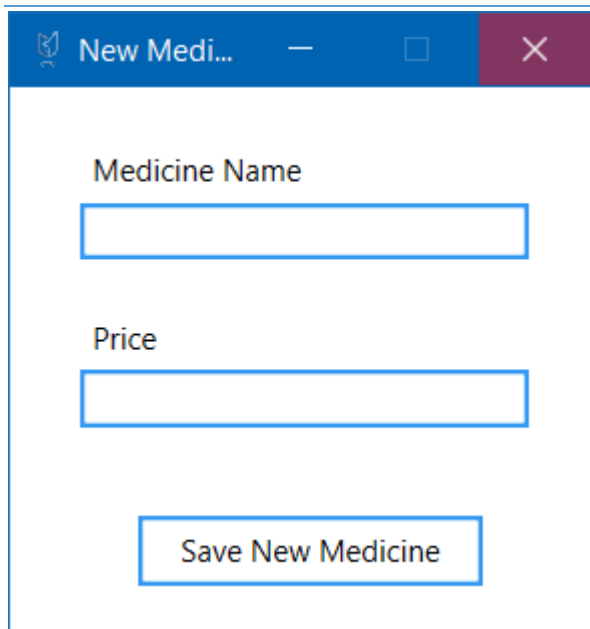
```

        <TextBox x:Name="txtName" Margin="28,46,26.6,0" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Height="23" VerticalAlignment="Top"/>
        <Label x:Name="label" Content="Procedure Name" HorizontalAlignment="Left"
Margin="28,20,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtPrice" Margin="28,113,26.6,0" TextWrapping="Wrap"
BorderBrush="#FF3699F1" BorderThickness="2" Height="23" VerticalAlignment="Top"/>
        <Label x:Name="label_Copy" Content="Price" HorizontalAlignment="Left"
Margin="28,87,0,0" VerticalAlignment="Top"/>
        <Button x:Name="btnSaveNew" Content="Save New Procedure"
Margin="51,171,44.6,0" Background="White" BorderBrush="#FF3699F1" BorderThickness="2"
Height="28" VerticalAlignment="Top" Click="btnSaveNew_Click"/>

    </Grid>
</Window>

```

New Medicine



C#

```

using System;
using System.Linq;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for NewMedicine.xaml
    /// </summary>
    public partial class NewMedicine : Window
    {
        public NewMedicine()
        {
            InitializeComponent();

            CTUCareDB db = new CTUCareDB();

            private void btnSaveNew_Click(object sender, RoutedEventArgs e)

```

```

{
    if (txtName.Text.Count() > 0)
    {
        try
        {
            Medicine newMedi = new Medicine
            {
                Name = txtName.Text,
                Price = Decimal.Parse(txtPrice.Text),
            };

            db.Medicines.Add(newMedi);
            db.SaveChanges();

            Update.updateMedicines();

            this.Close();
        }
        catch (Exception)
        {
            MessageBox.Show("Please enter a monetary value");
        }
    }
    else
    {
        MessageBox.Show("Please enter a valid name");
    }
}
}
}
}

```

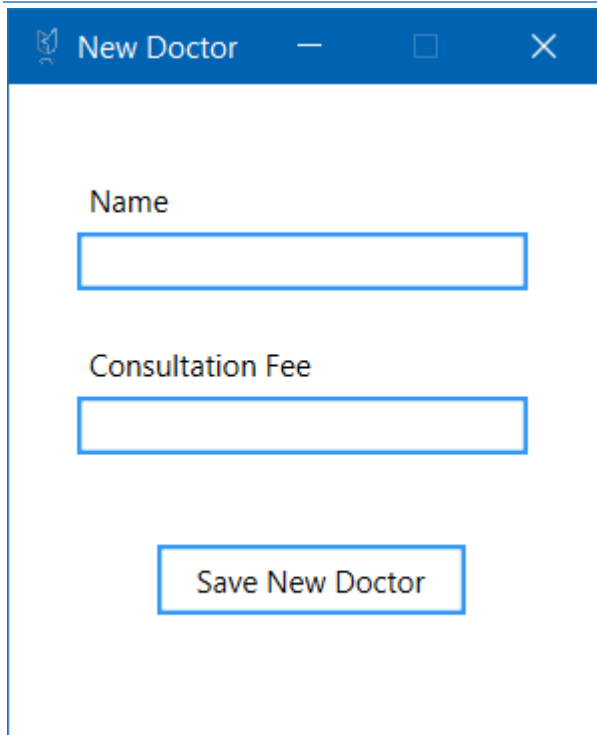
XAML

```

<Window x:Class="CTUCare.NewMedicine"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="New Medicine" Height="256.667" Width="247.778"
    Icon="images/Picture5.png"
    ResizeMode="CanMinimize">
    <Grid>
        <TextBox x:Name="txtName" Margin="28,46,0,0" TextWrapping="Wrap"
            BorderBrush="#FF3699F1" BorderThickness="2" Height="23" VerticalAlignment="Top"
            HorizontalAlignment="Left" Width="179"/>
        <Label x:Name="label" Content="Medicine Name" HorizontalAlignment="Left"
            Margin="28,20,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtPrice" Margin="28,113,0,0" TextWrapping="Wrap"
            BorderBrush="#FF3699F1" BorderThickness="2" Height="23" VerticalAlignment="Top"
            HorizontalAlignment="Left" Width="179"/>
        <Label x:Name="label_Copy" Content="Price" HorizontalAlignment="Left"
            Margin="28,87,0,0" VerticalAlignment="Top"/>
        <Button x:Name="btnSaveNew" Content="Save New Medicine" Margin="51,171,0,0"
            Background="White" BorderBrush="#FF3699F1" BorderThickness="2" Height="28"
            VerticalAlignment="Top" HorizontalAlignment="Left" Width="138"
            Click="btnSaveNew_Click"/>
    </Grid>
</Window>

```

New Doctor



C#

```
using System;
using System.Linq;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for NewMedicine.xaml
    /// </summary>
    public partial class NewMedicine : Window
    {
        public NewMedicine()
        {
            InitializeComponent();

            CTUCareDB db = new CTUCareDB();

            private void btnSaveNew_Click(object sender, RoutedEventArgs e)
            {
                if (txtName.Text.Count() > 0)
                {
                    try
                    {
                        Medicine newMedi = new Medicine
                        {
                            Name = txtName.Text,
                            Price = Decimal.Parse(txtPrice.Text),
                        };
                    }
                }
            }
        }
    }
}
```



```

        db.Medicines.Add(newMedi);
        db.SaveChanges();

        Update.updateMedicines();

        this.Close();
    }
    catch (Exception)
    {
        MessageBox.Show("Please enter a monetary value");
    }
}
else
{
    MessageBox.Show("Please enter a valid name");
}
}
}
}
}

```

New Medical Aid Company

The screenshot shows a standard Windows application window with a blue title bar. The title bar text is 'New Medi...'. The window content area is white and contains three elements: a text box with the label 'Name' above it, another text box with the label 'Email' above it, and a button with the text 'Save New Comapny' below the 'Email' text box. The button text contains a typo.

C#

```

using System;
using System.Linq;
using System.Windows;

namespace CTUCare
{
    /// <summary>
    /// Interaction logic for NewMedicine.xaml
    /// </summary>
    public partial class NewMedicine : Window
    {
        public NewMedicine()
        {

```

```

        InitializeComponent();
    }

    CTUCareDB db = new CTUCareDB();

    private void btnSaveNew_Click(object sender, RoutedEventArgs e)
    {
        if (txtName.Text.Count() > 0)
        {
            try
            {
                Medicine newMedi = new Medicine
                {
                    Name = txtName.Text,
                    Price = Decimal.Parse(txtPrice.Text),
                };

                db.Medicines.Add(newMedi);
                db.SaveChanges();

                Update.updateMedicines();

                this.Close();
            }
            catch (Exception)
            {
                MessageBox.Show("Please enter a monetary value");
            }
        }
        else
        {
            MessageBox.Show("Please enter a valid name");
        }
    }
}

```

XAML

```

<Window x:Class="CTUCare.NewMedicalAidCo"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:CTUCare"
    mc:Ignorable="d"
    Title="New Medical Aid Company" Height="280" Width="247.778"
    Icon="images/Picture5.png"
    ResizeMode="CanMinimize">
    <Grid>
        <TextBox x:Name="txtName" Margin="27,51,26.6,0" TextWrapping="Wrap"
            BorderBrush="#FF3699F1" BorderThickness="2" Height="23" VerticalAlignment="Top"/>
        <Label x:Name="label" Content="Name" HorizontalAlignment="Left"
            Margin="27,25,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="txtEmail" Margin="27,125,26.6,0" TextWrapping="Wrap"
            BorderBrush="#FF3699F1" BorderThickness="2" Height="23" VerticalAlignment="Top"/>
        <Label x:Name="label_Copy" Content="Email" HorizontalAlignment="Left"
            Margin="27,98,0,0" VerticalAlignment="Top"/>
        <Button x:Name="btnSaveNew" Content="Save New Comapny" Margin="59,184,59.6,0"
            Background="White" BorderBrush="#FF3699F1" BorderThickness="2" Height="28"
            VerticalAlignment="Top"/>
    </Grid>

```

</Window>

PatientToken.cs

```
namespace CTUCare
{
    //Used to carry information from one window to another about the patient
    public static class PatientToken
    {
        public static string patientName;
        public static int patientID;
    }
}
```

Update.cs

```
using System.Collections.Generic;

namespace CTUCare
{
    public static class Update
    {
        /* These lists are updated using the methods below */
        public static List<Procedure> ProcedureList = new List<Procedure>();
        public static List<Medicine> MedicineList = new List<Medicine>();
        public static List<Doctor> DoctorList = new List<Doctor>();
        public static List<Patient> PatientList = new List<Patient>();
        public static List<AdministeredProcedure> AdminProcedureList = new
List<AdministeredProcedure>();
        public static List<MedicalAid> MedicalAidCoList = new List<MedicalAid>();

        /* These methods are used to update comboboxes/listboxes and anything that
contains items
        throughout the application. They are static so they can be accessed
anywhere */
        #region Methods
        public static void updateCompanies()
        {
            MedicalAidCoList.Clear();

            foreach (var item in CTUCareDB.Instance.MedicalAids)
            {
                MedicalAidCoList.Add(item);
            }
        }

        public static void updateProcedures()
        {
            ProcedureList.Clear();

            foreach (var item in CTUCareDB.Instance.Procedures)
            {
                ProcedureList.Add(item);
            }
        }
    }
}
```

```
    }

    public static void updateAdminProcedures()
    {
        AdminProcedureList.Clear();

        foreach (var item in CTUCareDB.Instance.AdministeredProcedures)
        {
            AdminProcedureList.Add(item);
        }
    }

    public static void updateMedicines()
    {
        MedicineList.Clear();

        foreach (var item in CTUCareDB.Instance.Medicines)
        {
            MedicineList.Add(item);
        }
    }

    public static void updateDoctors()
    {
        DoctorList.Clear();

        foreach (var item in CTUCareDB.Instance.Doctors)
        {
            DoctorList.Add(item);
        }
    }

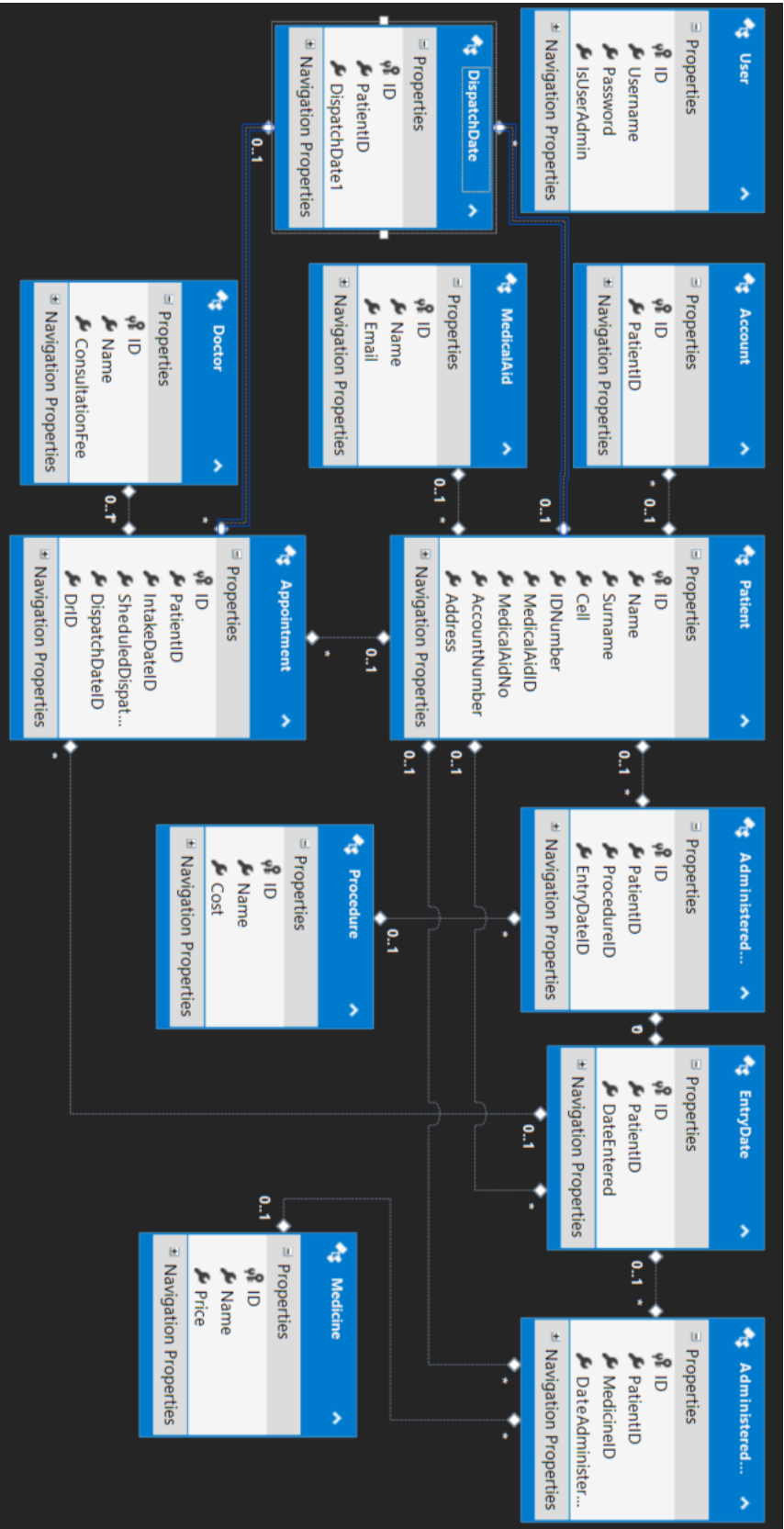
    public static void updatePatients()
    {
        PatientList.Clear();

        foreach (var item in CTUCareDB.Instance.Patients)
        {
            PatientList.Add(item);
        }
    }
    #endregion
}
}
```

UserType.cs

```
namespace CTUCare
{
    public static class UserType
    {
        //Used as a token to enable and disable features
        public static int UserKind;
    }
}
```

Database Diagram



Error Report

Error 1

```

10 references
public partial class Information : Window
{
    2 references
    public Information()
    {
        InitializeComponent();

        loadFields();
    }

    1 reference
    private void loadFields()
    {
        txtName.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().Name;
        txtSurname.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().Surname;
        txtIDNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().IDNumber;
        txtMedicalAidNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().MedicalAidNumber;
        txtAccNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().AccountNumber;
        txtCell.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().Cell;
        txtFileNo.Text = PatientToken.patientID.ToString();
    }
}

```

NullReferenceException was thrown
An exception of type 'System.NullReferenceException' was thrown in 'CTUCare.exe' but was not handled.
Additional information: Object reference not set to an instance of an object.

When loading the login (2nd page to appear when starting the application), the compiler initializes the window without me requesting this

Solution 1

```

try
{
    txtName.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().Name;
    txtSurname.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().Surname;
    txtIDNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().IDNumber;
    txtMedicalAidNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().MedicalAidNumber;
    txtAccNo.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().AccountNumber;
    txtCell.Text = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().Cell;
    txtFileNo.Text = PatientToken.patientID.ToString();

    var medicID = CTUCareDB.Instance.Patients.Where(x => x.ID == PatientToken.patientID).FirstOrDefault().MedicalAidNumber;
    txtMedicalAidCo.Text = CTUCareDB.Instance.MedicalAids.Where(x => x.ID == medicID).FirstOrDefault().Name;

    if (UserType.UserKind == 2)
    {
        txtAccNo.Visibility = Visibility.Collapsed;
        lblAccNo.Visibility = Visibility.Collapsed;

        txtCell.Visibility = Visibility.Collapsed;
        lblCell.Visibility = Visibility.Collapsed;

        txtMedicalAidCo.Visibility = Visibility.Collapsed;
        lblMedicalAidCo.Visibility = Visibility.Collapsed;
    }
}
catch(Exception)
{
}

```

Placed the loadFields() method inside of a try block with an empty catch

Error 2

```
NewMedicine newMed = new NewMedicine();

1 reference
private void btnNewMedicine_Click(object sender, RoutedEventArgs e)
{
    newMed.Show();
}
```

I was getting double instances of windows when launching them like this

Solution 2

```
NewMedicine newMed = new NewMedicine();

1 reference
private void btnNewMedicine_Click(object sender, RoutedEventArgs e)
{
    try
    {
        newMed.Show();
    }
    catch (Exception)
    {
        NewMedicine newMed = new NewMedicine();
        newMed.Show();
    }
}
```

Similarly to the last solution I placed the code block inside try catch block that re-initializes the window if it has been closed

Error 3

```
2 references
public void updateProcedureBox()
{
    Update.updateProcedures();

    cboxProcedures.Items.Clear();

    foreach (var item in Update.ProcedureList)
    {
        cboxProcedures.Items.Add(item.Name);
    }
}
```

Every now and comboboxes and listboxes would not update their items when event triggers were called

Solution 3

```
public void updateProcedureBox()
{
    Update.updateProcedures();

    cboxProcedures.Items.Clear();

    foreach (var item in Update.ProcedureList)
    {
        cboxProcedures.Items.Add(item.Name);
    }

    cboxProcedures.Items.Refresh();
}
```

I simply refreshed the items of whatever box I was trying to update

Conclusion

Unfortunately, the sheer size of the programme made it more and more difficult to follow a plan as the layout evolved along the way. Many originally planned features did come through, however other key features were integrated together and changed drastically from what was set out

The appointment system evolved the most as expected and was the main feature of the application in the end. One can assume this is due to the fact that no strict constraints were to be followed