

Senkou cards

Flashcards to memorize new topics

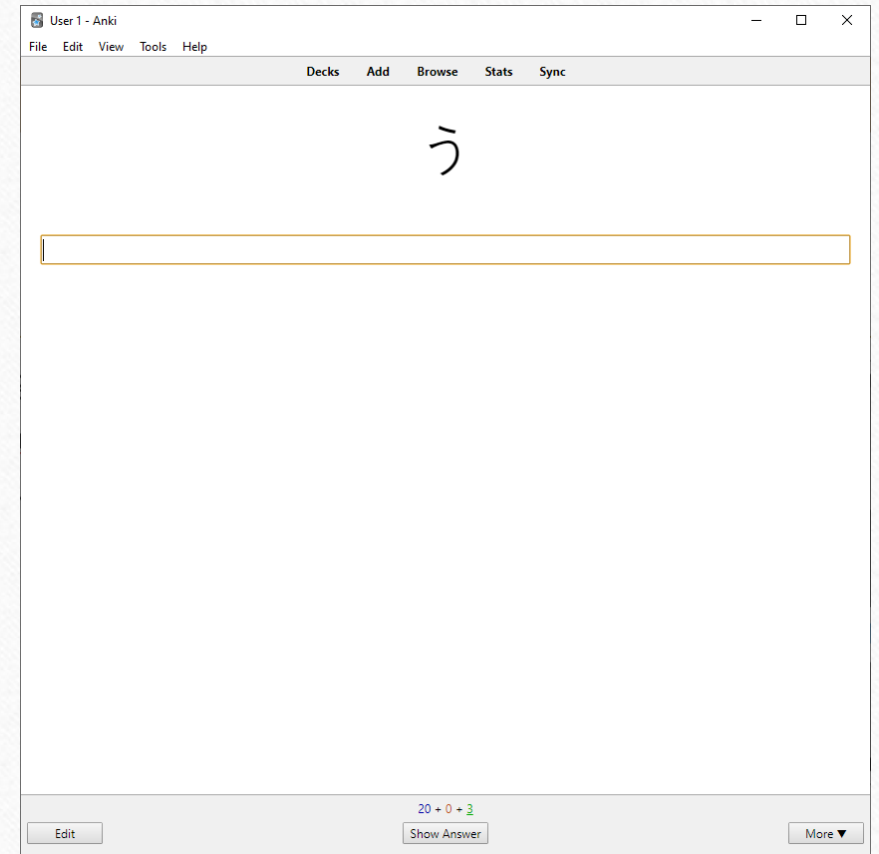
By

Wps Por-Favor (WPS)

Rami Chaouki, Caelan Seto, Ali Nehme

Background

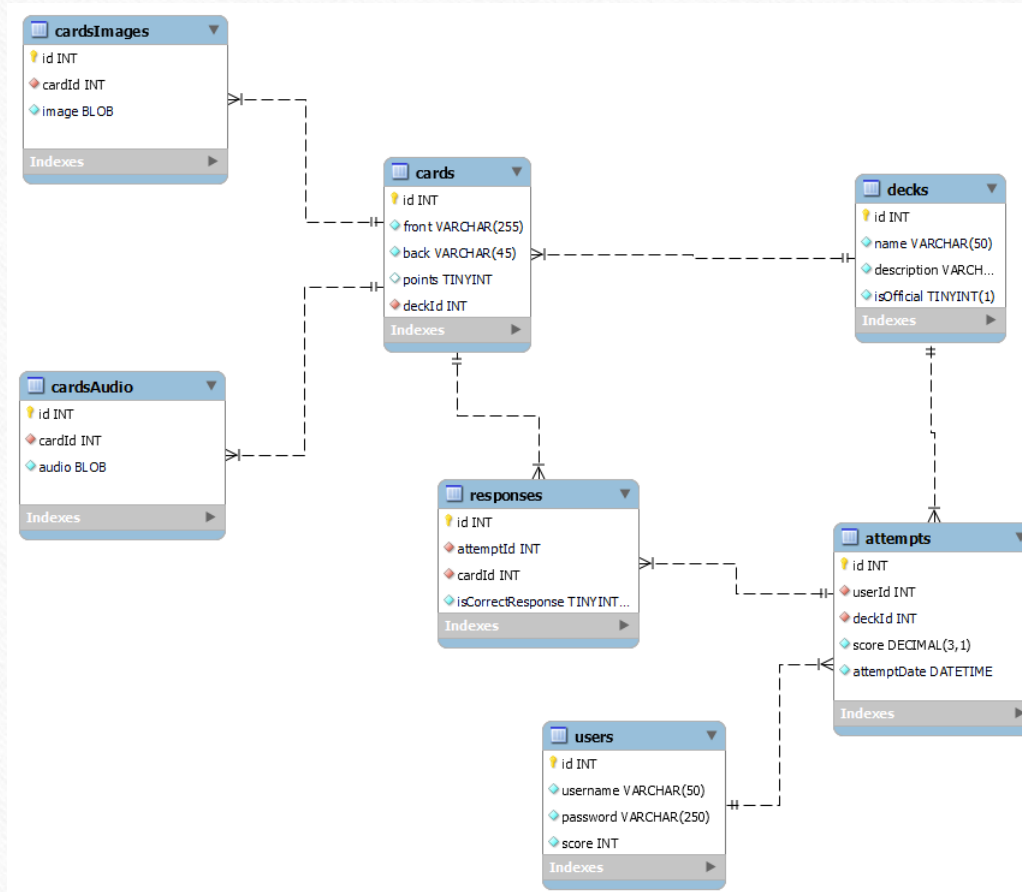
- Language learning App:
 - Top-Down: Memrize, Duolinguo
 - Bottom-up: Anki
- Cloud based vs. Locally Run
- Origin of Senkou (閃光) Cards



Senkou Cards

- Availability: All information stored in Azure Cloud Database
- Decks = sets of cards
- Card = front (question) + back (answer = card name) + points (weight)
 - Rich text
 - Audio and images
- User login, registration and offline
- User score
- Penalty on multiple attempts
 - Avoids increasing score by reattempting same quiz multiple times

Database



User

- Run of the mill login - registration
- Global user once logged-in
- Offline mode - currently rudimentary

```
Globals.ActiveUser = new users { id = -1, username="Local User", password=""};
```

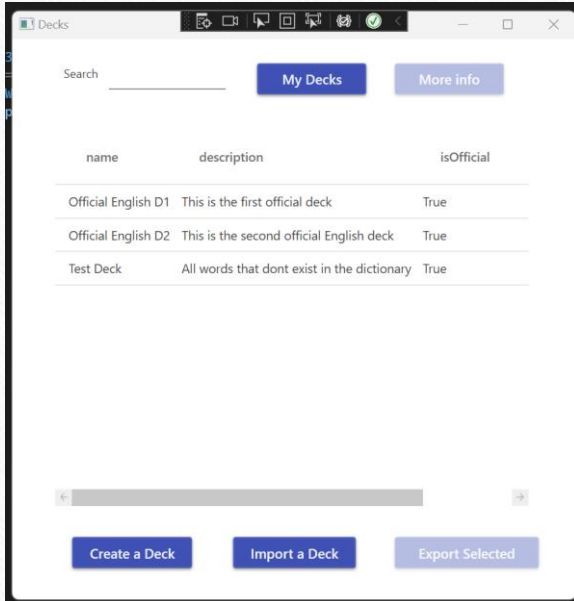
```
private static users _activeUser;

14 references
public static users ActiveUser
{
    set
    {
        if (_activeUser == null)
        {
            _activeUser = value;
        }
    }

    get
    {
        return _activeUser;
    }
}
```


Decks

All Decks

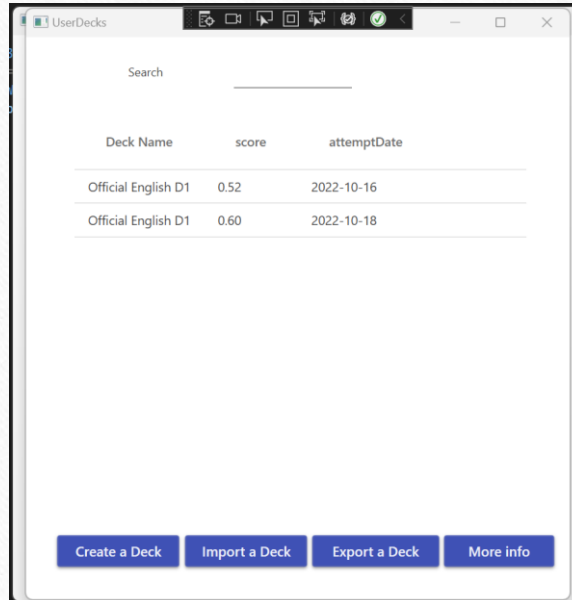


Search [My Decks](#) [More info](#)

name	description	isOfficial
Official English D1	This is the first official deck	True
Official English D2	This is the second official English deck	True
Test Deck	All words that dont exist in the dictionary	True

[Create a Deck](#) [Import a Deck](#) [Export Selected](#)

User Attempts

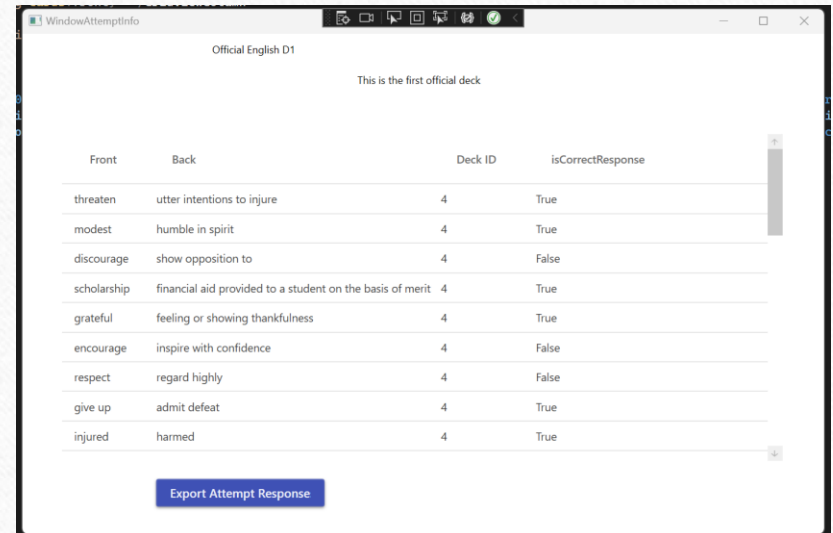


Search

Deck Name	score	attemptDate
Official English D1	0.52	2022-10-16
Official English D1	0.60	2022-10-18

[Create a Deck](#) [Import a Deck](#) [Export a Deck](#) [More info](#)

Attempt responses



Official English D1

This is the first official deck

Front	Back	Deck ID	isCorrectResponse
threaten	utter intentions to injure	4	True
modest	humble in spirit	4	True
discourage	show opposition to	4	False
scholarship	financial aid provided to a student on the basis of merit	4	True
grateful	feeling or showing thankfulness	4	True
encourage	inspire with confidence	4	False
respect	regard highly	4	False
give up	admit defeat	4	True
injured	harmed	4	True

[Export Attempt Response](#)

Decks

Select a Deck

The 'Decks' application window features a search bar at the top left. To its right are two buttons: 'My Decks' and 'More info'. Below these is a table with three columns: 'name', 'description', and 'isOfficial'. The table contains three rows of data. At the bottom of the window, there is a horizontal scrollbar and three buttons: 'Create a Deck', 'Import a Deck', and 'Export Selected'.

name	description	isOfficial
Official English D1	This is the first official deck	True
Official English D2	This is the second official English deck	True
Test Deck	All words that dont exist in the dictionary	True

Deck Info

The 'DeckInfo' application window displays details for a selected deck. At the top, the deck name 'Official English D1' is shown in a grey header. Below this, the 'No. of Cards' is listed as '17'. The 'Description' field contains the text 'This is the first official deck'. At the bottom of the window, there are three buttons: 'Edit', 'Attempt', and 'Attempts History'.

Official English D1

No. of Cards 17

Description This is the first official deck

Deck-specific attempts

The 'DeckHistory' application window shows the attempt history for the 'Official English D1' deck. At the top, the deck name is displayed, followed by the text 'This is the first official deck'. Below this is a table with two columns: 'score' and 'attemptDate'. The table contains two rows of data. At the bottom of the window, there are two buttons: 'Export History' and 'More info'.

Official English D1

This is the first official deck

score	attemptDate
0.52	2022-10-16
0.60	2022-10-18

Cards

Create Card

CardCreation

Upload Image

Upload Audio

B

I

U

Points:

Import from Server

Upload to Server

View All Cards

View All Cards

Search

front	back	points
embarrassed	feeling uneasy and self-conscious	10
remind	put in the mind of someone	10
introduce	confident	10

Deck Name:

Official English D1

Description:

change again

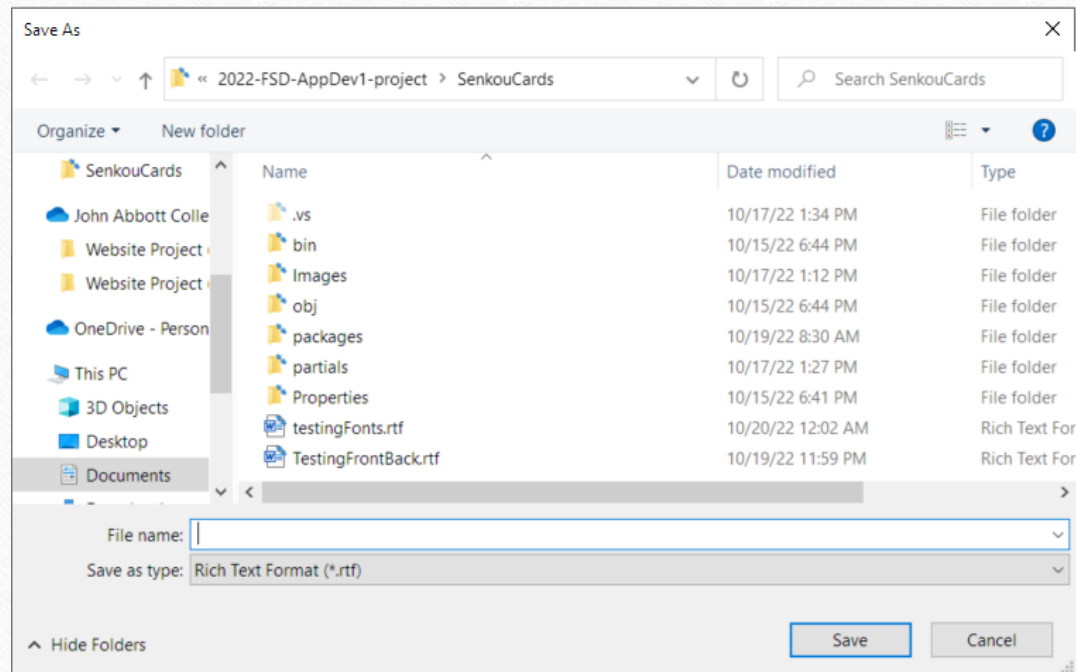
Update Deck Info

Create a Card

Delete a Card

Rich Text

Bold *Italics* Underline **All Three** Different font Different size



Quiz

- Buttons Randomized
- Oxford API
 - Securely Storing Keys
 - HTTP client
- Weighted Scoring

```
private void RegisterAttempt()
{
    SenkocardsConfig Sc = new SenkocardsConfig();
    int prevAttempts=(from a in Sc.attempts where a.deckId == currentDeck.id && a.userId == Globals.ActiveUser.id select a).Count();
    decimal finalScore =
        (prevAttempts == 0) ?
        (decimal)unweightedScore :
        (prevAttempts == 1) ?
        (decimal)unweightedScore * (decimal)0.5 :
        (prevAttempts == 2) ?
        (decimal)unweightedScore * (decimal)0.25 :
        0;
    attempts attempt=new attempts { userId=Globals.ActiveUser.id,deckId=currentDeck.id,score=finalScore, attemptDate = DateTime.Now };
    Sc.attempts.Add(attempt);
    foreach(response r in response)
```


Challenges and solutions

Things we have learned

Environmental Variables

- AppSettings

```
var appSettings = ConfigurationManager.AppSettings;  
string key = appSettings["OX_API_KEY"];  
string id = appSettings["OX_API_ID"];
```

```
<appSettings>  
  <add key="OX_API_ID" value="a6732c54" />  
  <add key="OX_API_KEY" value="7f3c550204f187870da7f4833cf736af" />  
</appSettings>
```

Field Validation

- Database First Requires Partial Classes and Metadata

```
[MetadataType(typeof(UsersMetadata))]  
16 references  
public partial class users  
{  
    private string _confirmPassword;  
    [NotMapped]  
    1 reference  
    public string confirmPassword  
    {  
        get  
        {  
            return _confirmPassword;  
        }  
        set  
        {  
            if (value == "")  
            {  
                throw new ArgumentException("You must confirm the password you wrote.");  
            }  
            else if (value != password)  
            {  
                throw new ArgumentException("Passwords Must Match");  
            }  
            else  
            {  
                _confirmPassword = value;  
            }  
        }  
    }  
}
```

```
1 reference  
public class UsersMetadata  
{  
    // [Required(AllowEmptyStrings=false, ErrorMessage = "Must enter a username")]  
    [StringLength(16, ErrorMessage = "Username must at least be 6 characters", MinimumLength = 6)]  
    0 references  
    public string username  
    {  
        get; set;  
    }  
    [StringLength(16, ErrorMessage = "Password must at least be 4 characters", MinimumLength = 4)]  
    0 references  
    public string password  
    {  
        get; set;  
    }  
}
```

Optional Parameters

reference

```
private void Window_Loaded(object sender, RoutedEventArgs e)
{
    setButtonsStatus();

    try
    {
        List<string> excluded = new List<string>() { "attempts", "cards", "users" };
        Globals.AddListViewColumns<decks>(GvDecks, excluded);
        LvDecks.ItemsSource = Globals.SenkouDbAuto.decks.ToList(); // equivalent of SELECT * FROM People
    }
    catch (SystemException ex)
    {
        MessageBox.Show(this, "Unable to access the database:\n" + ex.Message, "Fatal database error", MessageBoxButton.OK, MessageBoxImage.Error);
        Environment.Exit(1);
    }
}
```


Optional Parameters

```
4 references
public partial class WindowCreateDeck : Window
{
    Decks decks;
    1 reference
    public WindowCreateDeck(Decks wd=null)
    {
        decks = wd;
        InitializeComponent();
    }

    1 reference
    private void Btn_Create_Click(object sender, RoutedEventArgs e)
    {
        bool deckOfficial = Rbn_Yes.IsChecked == true;
        string deckName = Tbx_Name.Text;
        string deckDescription = Tbx_Description.Text;

        try
        {
            SenkoucardsConfig Sc = new SenkoucardsConfig();
            decks newDeck = new decks { ownerId = Globals.ActiveUser.id, name = deckName, description = deckDescription, isOfficial = deckOfficial };
            Sc.decks.Add(newDeck);
            Sc.SaveChanges();
            decks.LvDecks.ItemsSource = Globals.SenkouDbAuto.decks.ToList(); //updates the list with new deck
            this.DialogResult = true;
        }
        catch (Exception ex) when (ex is DataException || ex is SystemException)
        {
            MessageBox.Show("Unexpected Error... Who are we kidding, we expected all sorts of errors. "+ex.Message,"Error",MessageBoxButton.OK,MessageBoxImage.Error);
        }
    }
}
```

Generic Methods: Add columns

< T > stands for "Type"

Method (Globals.ca)

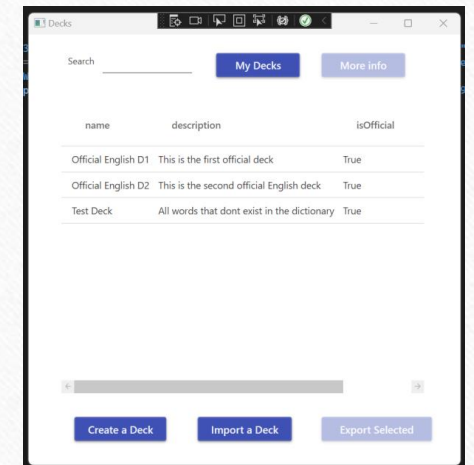
```
/**
 * Overloaded function to dynamically change column names based on class properties
 * with an excluded list to exclude from headers
 * */
5 references
public static void AddListViewColumns<T>(GridView GvFOO, List<string> excluded)
{
    foreach (System.Reflection.PropertyInfo property in typeof(T).GetProperties().Where(p => p.CanWrite)) //loop through the fields of the object
    {
        if (property.Name != "Id" && property.Name != "id" && !excluded.Contains(property.Name)) //if you don't want to add the id in the list view
        {
            GridViewColumn gvc = new GridViewColumn(); //initialize the new column
            gvc.DisplayMemberBinding = new Binding(property.Name); // bind the column to the field
            if (property.PropertyType == typeof(DateTime)) { gvc.DisplayMemberBinding.StringFormat = "yyyy-MM-dd"; } //optional if you want to display dates only for DateTime data
            gvc.Header = property.Name; //set header name like the field name
            GvFOO.Columns.Add(gvc); //add new column to the Gridview
        }
    }
}
```

XAML

```
<ListView Name="LvDecks" Height="388" VerticalAlignment="Top" Margin="44,92,39,0" SelectionChanged="LvUserDecks_SelectionChanged" MouseDoubleClick="LvUserDecks_MouseDoubleClick" SelectionMode="Extended"
    GridViewColumnHeader.Click="LvHeader_Click">
    <ListView.View>
        <GridView x:Name="GvDecks">
        </GridView>
    </ListView.View>
</ListView>
```

Code behind

```
List<string> excluded = new List<string>() { "attempts", "cards", "users" };
Globals.AddListViewColumns<decks>(GvDecks, excluded);
LvDecks.ItemsSource = Globals.SenkouDbAuto.decks.ToList(); // equivalent of SELECT * FROM People
```



Generic Methods: Export to CSV

Create header and lines (strings)

```
/**
 * Creates a file header with a specific delimiter for any class type, and excluding given list of properties
 * */
1 reference
public static string CreateFileHeaderFromProperties<T>(List<string> excluded, string delimiter = ",")
{
    var headerList = typeof(T).GetProperties()
        .Where(x => (!excluded.Contains(x.Name)))
        .Select(x => x.Name).ToList();
    string headerString = String.Join(delimiter, headerList);
    return headerString;
}

/**
 * Creates a delimiter-separated string of properties from any object, and excluding given list of properties
 * */
1 reference
public static string CreateStringFromProperties<T>(T source, List<string> excluded, string delimiter = ",")
{
    var propertiesList = typeof(T).GetProperties()
        .Where(x => (!excluded.Contains(x.Name)))
        .Select(x => x.GetValue(source) ?? "").ToList();
    string propertyString = String.Join(delimiter, propertiesList);
    return propertyString;
}
```

write a list of elements into a

```
/**
 * Writes a list of objects into a file
 * Takes a list of elements (objects) to be written and a list of properties to be excluded from the writing
 * */
4 references
public static void SaveToCSV<T>(List<string> excluded, List<T> elementsList)
{
    SaveFileDialog saveFileDialog = new SaveFileDialog();
    saveFileDialog.Filter = "CSV files (*.csv)|*.csv";
    if (saveFileDialog.ShowDialog() == true)
    {
        List<string> lines = new List<string>();
        lines.Add(Globals.CreateFileHeaderFromProperties<T>(excluded));
        foreach (T element in elementsList)
        {
            string newline = Globals.CreateStringFromProperties<T>(element, excluded);
            //Console.WriteLine(newline);
            lines.Add(newline);
        }
        File.WriteAllLines(saveFileDialog.FileName, lines); // ex IO/Sys
    }
}
```

Exporting cards

```
List<cards> cardsList = Globals.SenkouDbAuto.cards
    .Where(card => card.deckId == deckId)
    .ToList();

List<string> excluded = new List<string>() { "responses", "decks", "cardsAudios", "cardsImages" };

Globals.SaveToCSV<cards>(excluded, cardsList);
```

Exporting responses

```
List<responses> responsesList = currentAttempt.responses.ToList();
List<string> excluded = new List<string>() { "attempts" };

Globals.SaveToCSV<responses>(excluded, responsesList);
```


Blobs handling

```
private byte[] _imageBytes = null;
1 reference
private void BtnUploadImage_Click(object sender, RoutedEventArgs e)
{
    try
    {
        //TODO: EXCEPTIONS
        OpenFileDialog dlg = new OpenFileDialog
        {
            Filter = "Image files (*.jpg;*.png;*.jpeg)|*.jpg;*.png;*.jpeg;|All Files (*.*)|*.*",
            RestoreDirectory = true
        };
        if (dlg.ShowDialog() == true)
        {
            string selectedImageName = dlg.FileName;
            TbxImagePath.Text = selectedImageName;

            using (var fs = new FileStream(selectedImageName, FileMode.Open, FileAccess.Read)) //IOException, ArgumentException, UnauthorizedAccessException
            {
                _imageBytes = new byte[fs.Length];
                fs.Read(_imageBytes, 0, Convert.ToInt32(fs.Length));
            }
        }
    }
    catch (Exception ex) when (ex is IOException || ex is ArgumentException || ex is UnauthorizedAccessException)
    {
        MessageBox.Show(this, "Image upload failed\n" + ex.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);
    }
}
```

Rich text editor

```
private void RtbFront_SelectionChanged(object sender, RoutedEventArgs e)
{
    object temp = RtbFront.Selection.GetPropertyValue(Inline.FontWeightProperty);
    btnBold.IsChecked = (temp != DependencyProperty.UnsetValue) && (temp.Equals(FontWeights.Bold));
    temp = RtbFront.Selection.GetPropertyValue(Inline.FontStyleProperty);
    btnItalic.IsChecked = (temp != DependencyProperty.UnsetValue) && (temp.Equals(FontStyles.Italic));
    temp = RtbFront.Selection.GetPropertyValue(Inline.TextDecorationsProperty);
    btnUnderline.IsChecked = (temp != DependencyProperty.UnsetValue) && (temp.Equals(TextDecorations.Underline));

    temp = RtbFront.Selection.GetPropertyValue(Inline.FontFamilyProperty);
    cmbFontFamily.SelectedItem = temp;
    temp = RtbFront.Selection.GetPropertyValue(Inline.FontSizeProperty);
    cmbFontSize.Text = temp.ToString();
}
```

Future work

- Import deck and cards
- Develop offline (unranked) test from imported decks
- Image- and audio-based quiz
- Show answer
- Deploy project
- Rearrange the firewall settings to allow for anyone with the app to access it
- Material Design

```
public BitmapImage ConvertByteArrayToBitMapImage(byte[] imageByteArray)
{
    BitmapImage img = new BitmapImage();
    using (MemoryStream memStream = new MemoryStream(imageByteArray))
    {
        img.StreamSource = memStream;
    }
    return img;
}
```

