Ollscoil
Teicneolaíochta
an Atlantaigh

Atlantic
Technological
University

Catalogue ya deck

Card scanner and cataloguing system

Eoin Gallagher

BEng(H) in Software & Electronic Engineering

Project Engineering

Atlantic Technological University
2023/2024

Declaration:

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Software & Electronic Engineering at Galway-Mayo Institute of Technology.
This project is my own work, except where otherwise accredited. Where the work of others has been used, or incorporated during this project, this is acknowledged and referenced.

Acknowledgements:

I would like to thank my supervisor Brian for his help throughout the year as well as the other lecturers, they all gave me some advice over the course of the year and helped me reach where I am now.

Summary:

The project is a database and card scanning website for the trading card game Magic the Gathering. It uses many of the skills and knowledge I've gained over the 4 years of study. The project is python code for the card recognition using YOLO which was gotten from their own GitHub and edited for my own purposes, this is along side react code for the front end which was developed in Visual Studio IDE. Some of the features of the app include:

- Card logging.
- Sorting of card data.
- Extended card use and rulings.
- Mongoose SQL database.
- Card searching.

The user runs a local database that will hold the images of the cards that they upload to the app, the app then uses those images to gather the necessary data such as name of card, type etc.

Table of contents

1. INTRODUCTION:

1.1. Project Goals:

- The goals of the project are as follows:
- To be able to scan in Trading cards from the game Magic the Gathering.
- To store the data from the cards to a local database.
- To develop a website to view the stored data.
- To be able to remove/update the data from the webpage.

1.2. Scope:

The project aims to be a easy to use and convenient way for people to catalogue and store data on their card collections.

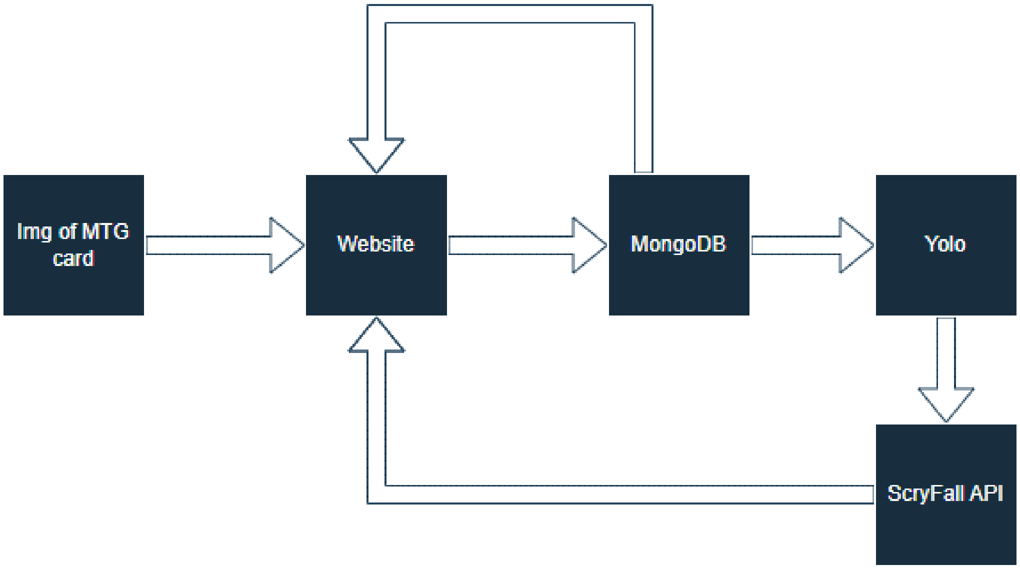
Should be easy to use and easy to search through to find the cards you want.

1.3. Report Overview:

Where ever there is code shown or explained it'll be shown using screenshots and snippets like the example below.

```
const mongoose = require('mongoose');
```

2. BLOCK DIAGRAM:



3. IMAGE RECOGINITION:

3.1. Usage:

There are a few reasons why I wanted to use this such as:

- Gathering info from the card itself so users don't need to input as much data.
- Having the image of the card so it could be displayed along side the gathered info on the home page.
- Would make the site easier to use as all the user would need to do is snap a picture of their card and upload that.

3.2. YOLO:

YOLO also known as You Only Look Once, Built on PyTorch it is an object detection model.

By training it on some data in the form of images, you can have it be able to detect what is within an image like stop signs, playing card suits, animals and more.

YOLO comes pretrained on the above mentioned items so I needed to train it to be able to detect MTG (Magic the Gathering) cards.

3.3. Python:

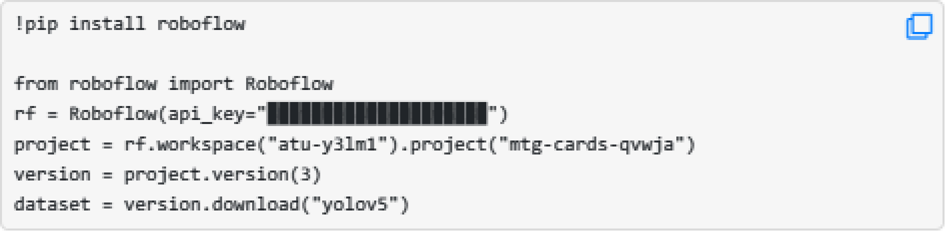
YOLO is coded using python, Python is a interpreted object-oriented, high-level programming language, its pretty modular so makes sense it was used for YOLO to be able to load in the different data sets for it to learn what the images are from.

3.4. Dataset:

I had to make my own custom data set for the cards that I personally own.

There were some online on sites like Kaggle, but majority of them were either older sets, so using cards I don't personally own or were just listed in a Json format using words only.

The dataset I ended up making was done using RoboFlow, its a website specially used for making image datasets for Image recognition/OCR (Optical character recognition), RoboFlow has built in YOLO support, so after I added the images I wanted to train it on it would supply me some code that o was to use to load that into my version of YOLO as shown below.



```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="[redacted]")
project = rf.workspace("atu-y3lm1").project("mtg-cards-qvwja")
version = project.version(3)
dataset = version.download("yolov5")
```

4. MONGODB DATABASE:

4.1. MongoDB:

MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need.

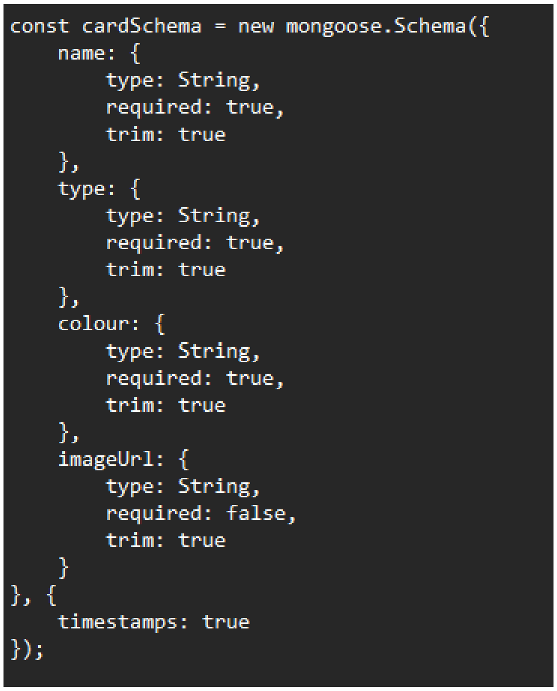
4.2. Schema:

The schema of the database is as shown below, Name is string that is given by the user/gotten from YOLO, it is the name that is printed on the top of the card.

Type is again another string, this is the card type printed below the card art, this is normally something like “monster”

Colour is a string and is on the tip right of the card itself and is represented by a few different colour words like, black, green, blue etc.

And finally, ImageURL is the image of the full card uploaded by the user, it is convereted to a URL by the code is accessed by YOLO and is posted to the homepage alongside the other 3 members of the schema

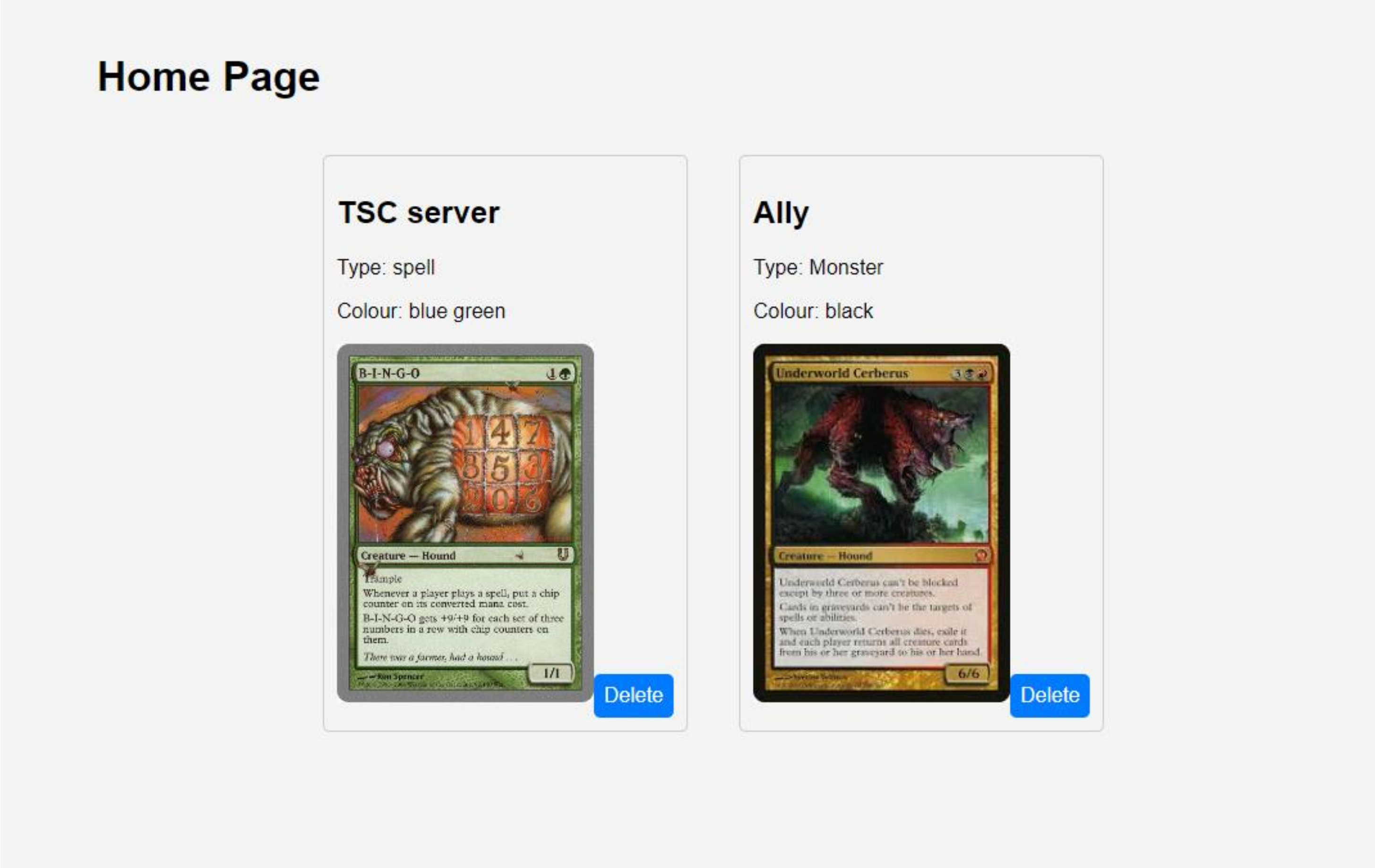


```
const cardSchema = new mongoose.Schema({
    name: {
        type: String,
        required: true,
        trim: true
    },
    type: {
        type: String,
        required: true,
        trim: true
    },
    colour: {
        type: String,
        required: true,
        trim: true
    },
    imageUrl: {
        type: String,
        required: false,
        trim: true
    }
}, {
    timestamps: true
});
```

5. WEBSITE:

A website we developed to allow users to access the information and to add more cards to the database. Other features of the site are to allow the users to manually delete entries of the database and to view and sort through what is already saved to the database.

Below is a sample of how the home page looks with some data already on the site.



Home Page

TSC server

Type: spell

Colour: blue green

B-I-N-G-O
Creature — Hound
Trample
Whenever a player plays a spell, put a chip counter on its converted mana cost.
B-I-N-G-O gets +9/+9 for each set of three numbers in a row with chip counters on them.
There was a farmer, had a hound . . .
1/1
Delete

Ally

Type: Monster

Colour: black

Underworld Cerberus
Creature — Hound
Underworld Cerberus can't be blocked except by three or more creatures.
Cards in graveyards can't be the targets of spells or abilities.
When Underworld Cerberus dies, exile it and each player returns all creature cards from his or her graveyard to his or her hand.
6/6
Delete

As well as the main parts of the site I mentioned above, I also made a bottom tab bar for navigation between the add card page and the home page, this was so if later I wanted to port the site to mobile it would easily work on phone as well as PC.

6. PROBLEM SOLVING:

6.1. YOLO:

I had planned to actually use YOLO, while I spent a long time working on it and researching it I sadly couldn't get it to work within the time I had, being as it isn't something I'm too knowledgeable in, nor is it using a language I'm too confident in (python) it made trying to use it quite tough, maybe with more time I could possibly get it working.

Using YOLO to get the car name and what set it was from I was going to use the API which is connected to a large scale database of all official MTG cards to get extra info on the cards such as rules, prices and more.

6.2. ScryFall API:

I didn't really talk about this much through out the report but, this was to be a huge part of the project.

At first it seemed like it would be possible but the more I looked into ScryFall the more it seemed like I would need to be approved by the dev team behind it before I was ever given access to use it.

I didn't attempt to try getting access as I found from a few different sources that getting access would take some time, with no idea how long it would take I felt like hoping I could get access would be a waste of time so I decided to drop it.

6.3. Time Management:

I'm not particularity good at time keeping, so trying to keep to a schedule and getting stuff done in a timely manner along with other project and assignments was one of my largest struggles.

Getting time to work on this project has been a huge struggle for myself between working and other college commitments. So I was only able to put in the work when I had some spare time as I kept this at the back of my mind most days.

7. CONCLUSION:

Getting an OCR to work is way more work then I expected, but the application of the site is much easier to use then having to carry the cards around with me.

Overall I would call the site a mild success, I couldn't get YOLO to work as I had hoped but I was at least able to get the main site to work as I hoped it would work. It takes in data from the user and its able to post the data to a separate page and has a way for users to delete said data.

In the future I hope to get the OCR working to help increase the functionality of the site and possibly find a different API as the one I wanted to use was restricted to developers who requested access to it from the team.

If I had more time or if I knew a bit more about how OCR or YOLO itself works.

8. REFERENCES:

YOLOv5: https://github.com/ultralytics/yolov5/tree/master/data

Scryfall API: https://github.com/ultralytics/yolov5/tree/master/data