# ETL Project

By: Al Faust

## Objective

Create an item lookup that gathers in-game market data from Old School RuneScape and saves it to a sql database.

## Tools/Methods

Screen scrape: Splinter, Beautiful Soup

Data Cleaning: API/JSON, Pandas, datetime

SQL Loading: SQLalchemy

## Extract

Information was first taken from the OSRS apis and itemdb.biz. To retrieve information from the api, an item id is first needed. Since a user would have no way of knowing the item id without doing research themselves, I scraped the itemdb.biz site base on the user’s search input. This user input is first standardized and then inserted in the itemdb.biz web url so the correct table information is gathered. Once the item id is saved, it is then used in the OSRS apis to get a json response of the item’s marketplace information.

## Transform

Once the json responses are saved, each section is separated into its own variable and reassembled in two pandas DataFrames. The gen\_df required certain characters to be striped from the responses so that they could be imported properly into the table. This was handled by a simple if/else statement that used a type check to determine in the formatting changes needed to be made to the data. The graph\_daily dictionary was a bit trickier to work with. The keys and values first needed to be separated into two lists. The keys list then needed to be reformatted from the milliseconds since epoch format to a standard date. Thankfully, I found a simple conversion using datetime and once complete, these values could be inserted into their own DataFrame as daily\_graph\_df. Lastly a graph is created showing the daily values from the last six months of the searched item using data from daily\_graph\_df.

## Load

Once each table was cleaned, they were loaded into sql using sqlalchemy. This had a small issue of not updating pre-existing rows with the same item id. To work around this, a temp\_holding table was created and a try/except statement was put in place that compared the two tables (temp\_holding and general\_info). If the id in temp\_holding matched an entry in general\_info, that entry was deleted and the temp\_holding entry was used to updated the general\_info table.