

# Kamran Rasim Asgarov

askerovk@gmail.com ❖ +65 81260560 ❖ Singapore ❖ [LinkedIn](#) ❖ [Github](#)

## YARA INTERNATIONAL

---

<b>Active Campaign ETL</b>	<i>Python (pandas, requests), Airflow, Zuora, Active Campaign, Redshift</i>	2023
<ul style="list-style-type: none"><li>• Wrote python code to automate setup of a staging Redshift database, with identical schema, populated by sample data.</li><li>• Extended, documented and diagrammed legacy Airflow DAGS.</li></ul>		
<b>Amplitude Historical Data</b>	<i>Amplitude, RudderStack, AWS S3, Python (boto3, pandas)</i>	2023
<ul style="list-style-type: none"><li>• Built a RudderStack ETL to parse RudderStack logs in an S3 bucket and push archived application data to an Amplitude project.</li></ul>		
<b>User Data Anonymization Pipelines</b>	<i>Python (pandas, boto3, requests), Airflow, S3, Redshift, Amplitude</i>	2022
<ul style="list-style-type: none"><li>• Wrote an Airflow DAG to download RudderStack logs in json/parquet formats from S3, parse and anonymize data for a given user and re-upload the file to S3.</li><li>• Wrote an Airflow DAG to anonymize Active Campaign data in Redshift.</li><li>• Wrote an Airflow DAG to send and track progress of anonymization requests to Amplitude API</li></ul>		
<b>Amplitude Historical Data</b>	<i>Amplitude, RudderStack, AWS S3, Python (boto3, pandas)</i>	2022
<ul style="list-style-type: none"><li>• Built a RudderStack ETL to parse RudderStack logs in an S3 bucket and push archived application data to an Amplitude project.</li></ul>		
<b>Metrics Layer POC</b>	<i>Transform/Metrics Flow, PostgreSQL</i>	2022
<ul style="list-style-type: none"><li>• Conducted a Proof of Concept project for implementing a Metrics Layer in the company, using the Transform platform and a sample SQL database.</li><li>• Showcased Transform's metric visualization capabilities to the Analyst Team.</li></ul>		
<b>Data Collectors Analytics</b>	<i>Python (pandas, boto3), MongoDB, Power Bi, SharePoint</i>	2021
<ul style="list-style-type: none"><li>• Built a Dockerized ETL to collect agent data from MongoDB, calculate performance metrics and write it as a .csv to SharePoint.</li><li>• Built a Power BI dashboard to highlight high/low performing data collector agents, as well as overall progress towards team goal.</li></ul>		
<b>5 Amigos Analytics</b>	<i>Python (pandas, sklearn, requests, boto3, GPSPPhoto) AWS S3, Zendesk</i>	2021
<ul style="list-style-type: none"><li>• Scraped 25k comment threads from Zendesk and experimented with Latent Dirichlet Allocation to develop an automatic topic tagging system.</li><li>• Coded an ETL pipeline to parse metadata from images stored in S3 bucket, utilizing multiprocessing to cut down code execution time.</li><li>• Performed translation comparison study between Microsoft Azure, IBM Watson, Google and AWS translation APIs.</li></ul>		
<b>Agonomic Content Delivery Framework</b>	<i>Data modelling, knowledge graphs,</i>	2021
<ul style="list-style-type: none"><li>• Developed a solution-agnostic framework which governs the ingestion, customization, delivery and localization of agronomic content.</li><li>• Dynamic, farm specific roster of advisories/alerts based on crop type, region and farmer knowledge/available tech.</li><li>• Shared microservices for location mapping, crop calendar, fertilizer recommendation, etc...</li><li>• Agile content localization of agronomic advisories, by reducing them to fact bundles.</li><li>• Advice triggered by farm specific calendar, regional alerts or farmer activity.</li></ul>		
<b>Crop Calendar</b>	<i>Python (pandas, matplotlib, seaborn)</i>	2020
<ul style="list-style-type: none"><li>• Developed and implemented a data framework for unifying regional crop calendars, based on BBCH crop stages.</li><li>• Created a series of visualizations to critique legacy crop calendar data, obtained from external sources.</li></ul>		

<b>Soil Modelling</b>	<i>Python (pandas, sklearn, folium, geopandas, shapely, plotly, matplotlib), PostgreSQL</i>	2020
<ul style="list-style-type: none"> <li>• Used a folium, a python based API for the Leaflet JavaScript library to generate interactive soil maps of several Indian States.</li> <li>• Trained kriging and inverse-distance mean interpolation methods for generating soil composition models.</li> <li>• Built a algorithm for visualizing and evaluating the performance of fertilizer recommendation models, given real world soil data.</li> <li>• Used findings to critique the use of district level soil compositions in India.</li> </ul>		
<b>Soil Health Cards</b>	<i>Python (boto3, requests, pandas), PostgreSQL, AWS RDS, S3, Docker, Proxy server</i>	2019
<ul style="list-style-type: none"> <li>• Wrote scraping code to acquire 20 million soil composition samples from an Indian government website as excel tables, stored in S3 bucket.</li> <li>• Scraping code was containerized with Docker. Each container was self-sufficient and obtained tasks from a centralized SQL table, allowing to run any number of parallel scraping jobs.</li> <li>• Wrote parsing script, which read excel files in S3, parsed tables within, de-normalized data and uploaded to a PostgreSQL database on RDS.</li> <li>• Spearheaded a landmark agreement with Indian government for direct access to the data.</li> </ul>		
<b>Intern Projects</b>	<i>Python (boto3, voronoi), AWS Sagemaker</i>	2019
<ul style="list-style-type: none"> <li>• Supervised creation of a Voronoi Diagram, in order to find the nearest major city for any location in India. Our approach was 3 times faster than previously used algorithm.</li> <li>• Supervised creation of an S3 to S3 file migration code, optimised to run on several processes, deployed on AWS Sagemaker.</li> </ul>		
<b>Crop Disease Models</b>	<i>Python (unit testing, object oriented programming), API</i>	2018
<ul style="list-style-type: none"> <li>• Wrote python implementation of 5 crop diseases forecasting models for a farmer app.</li> <li>• Used regional call center data to estimate crop disease outbreaks, along with IBM weather API, in order to validate the predictive power of crop disease models.</li> </ul>		
<b>FarmPulse</b>	<i>Python (requests, beautifulsoup), PostgreSQL, cronjob, AWS (Lambda, RDS, S3)</i>	2018
<ul style="list-style-type: none"> <li>• Wrote a python script to scrape 200k call centre records from a web widget.</li> <li>• Used serverless framework to deploy scraping/parsing code to AWS Lambda, triggered by a cronjob. Raw files were stored in S3 bucket, while parsed data was added to a PostgreSQL database on RDS.</li> </ul>		