

**BT4301 AY23/24 Semester 2**

**Prediction of Property Prices in the Housing Industry**

**Project Group 8**

| Oliver Gui Chin Wee | A0218079X |
| --- | --- |
| Khoo Cho Yaw, Valentin | A0217757U |
| Tan Wei Hao Bandy | A0214104B |
| Koh Quan Wei Ivan | A0217585W |
| Hang Jia Hui, Brenda | A0223617J |

[**1. Formulation of Analytics Project 4**](#_mrow08pvax9t)

[1.1 Problem Statement 4](#_kmdu0dy6kxpz)

[1.2 Business Value and Solution Statement 4](#_lep97gzek82q)

[**2. Agile Analytics Project Management with Scrum 5**](#_gd0ycjux6tj8)

[2.1 Overall Project Planning 5](#_jfus8w13y5t8)

[2.2 Sprint planning for DataOps and MLOps 6](#_lfb5x2m6xwal)

[**3. DataOps 6**](#_1swk0yuweu27)

[3.1 Dataset Ingestion 7](#_dhcl09g4tkni)

[3.2 Data Processing 9](#_rw027wvyrrt9)

[3.2.1 Data Cleansing 9](#_csy90fw36y0u)

[3.2.2 Data Transformation 10](#_kfasd0lhljj)

[3.2.2.1 Private Residential Property Transactions 10](#_n57jflx5q0jh)

[3.2.2.1 Postal Districts 11](#_lcnv9y8llt9z)

[3.2.2.3 HDB Resale Flat Transactions 11](#_5nx4c31cnrlw)

[3.3 Dataset Integration 11](#_srapwy4j054f)

[3.3.1 Data Quality and Data Profiling 12](#_8zd1uwus3epz)

[3.4 Data Lineage 13](#_mk9xeur02z2)

[3.5 Automation and orchestration of DataOps process 14](#_ijtnp3qc1uyb)

[**4. MLOps 15**](#_l4y39j2pv155)

[4.1 Model Development 16](#_rdqi21p3qcwt)

[4.1.1 Data Exploration 16](#_nomj40wam7vu)

[4.1.2 Feature Engineering 17](#_vtb9l7y71wy8)

[4.1.2.1 Encoding 17](#_es6kovky0y1g)

[4.1.2.2 Window Aggregations 17](#_lna603mgvqhk)

[4.1.3 Experimentation with MLFlow 17](#_qrgodnhopu2u)

[4.1.3.1 Model Training 17](#_qjopzg2u3w9v)

[4.1.3.2 Model Evaluation 17](#_u4t7oyrqul96)

[4.1.3.3 Reproducibility and Version Control 18](#_m79wvux60p81)

[4.2 Preparing for Production 18](#_e0v8jxn1ta15)

[4.2.1 Runtime Environments and Considerations 18](#_kdk0n37ybekm)

[4.2.2 Model Risk Evaluation 19](#_3maizueb2bh7)

[4.2.3 Quality Assurance and Model Testing 19](#_wt3rb2lirgtr)

[4.2.4 Reproducibility and Audibility 20](#_7h8w8w81fx7n)

[4.2.5 Risk Mitigation 20](#_bxo9fyhssnm8)

[4.3 Deploying to Production 20](#_mwh2kceultu1)

[4.3.1 Containerisation and Scaling Deployments 20](#_ghqtzimeilpt)

[4.3.2 CI/CD Pipeline and Testing 21](#_8vd91okxdoa)

[4.4 Monitoring and Feedback Loop 22](#_7xuhcgetqz5y)

[4.4.1 Ground Truth Evaluation 22](#_qhtrbap5415v)

[4.4.2 Input Drift Detection 23](#_rydewjhjhrp1)

[4.4.3 Feedback Loop 23](#_x9ucwn547g8q)

[4.5 Model Governance 24](#_ptjexa84q34o)

[**5. Conclusion 25**](#_15anu7p1o9bw)

[**6. Appendix 27**](#_vbfylnt26p5g)

[6.1 Kanban board 27](#_dbjfuogiofx2)

[6.2 Data quality and profiling email report 27](#_mcr9dkk3ap51)

[6.2.1 Example of data profiling email 27](#_wxv0m3hewkln)

[6.2.2 Example of descriptive statistics generated 27](#_u7wcit6vmp85)

[6.2.3 Example of visualisations generated for numerical variables 28](#_454zoq1frbrq)

[6.2.4 Example of visualisations generated for categorical variables 29](#_civph6v2v8j3)

# 

# 1. Formulation of Analytics Project

## 1.1 Problem Statement

The housing industry in Singapore faces several challenges, such as the fragmentation of information across different sources, and the volatility of the market due to external factors, as well as inflation and interest rates. These multifaceted challenges hinder realtors from making accurate valuations of property prices. As such, there is a need for an end-to-end analytics solution that merges housing information from various sources and analyses it to provide a comprehensive and up-to-date overview of the state of the market in Singapore.

## 1.2 Business Value and Solution Statement

The project aims to deliver an analytics solution to predict HDB resale flat prices given relevant data such as flat location and size while taking into consideration other factors like bank loan rates as well as the consumer price index. Given the surge in resale prices in recent years, as well as resale flat transactions making up a significant proportion of all property transactions in Singapore, this project is of value to real estate agencies in various ways.

Gathering and analysing data from different sources is integral to real estate agencies in reducing information asymmetry and increasing transparency of the property market, thus increasing their confidence in decision-making regarding property acquisitions, sales, and investments. This could result in enhanced pricing strategies and the ability to identify lucrative investment opportunities, leading to increased revenue generation for the company.

Prediction of housing prices also enables real estate agencies to mitigate risks. By anticipating fluctuations in resale flat prices, realtors can proactively manage risks associated with market volatility, such as identifying potential areas of overvaluation or downturns in certain segments of the housing market.

Real estate agencies could also utilise this solution to provide clients with premium services such as personalised market analysis or investment advisory services, thereby strengthening their brand image as go-to experts in the real estate industry to grow their clientele and revenue.

# 2. Agile Analytics Project Management with Scrum

## 2.1 Overall Project Planning

The overall project planning for the development of the analytics solution to predict HDB resale flat prices in Singapore followed an agile approach using the scrum methodology. The project was divided into two main sprints: DataOps and MLOps.

DataOps Sprint (Sprint 1):

The DataOps sprint focused on the data lifecycle, ensuring high-quality data for the analytics solution. The sprint goal was to establish a robust data pipeline that ingested, processed, integrated, and managed data lineage. The sprint backlog consisted of user stories related to data ingestion, cleansing, transformation, integration, quality checks, profiling, and automation using Airflow. The team members were assigned specific tasks based on their expertise and the sprint timeline.

MLOps Sprint (Sprint 2):

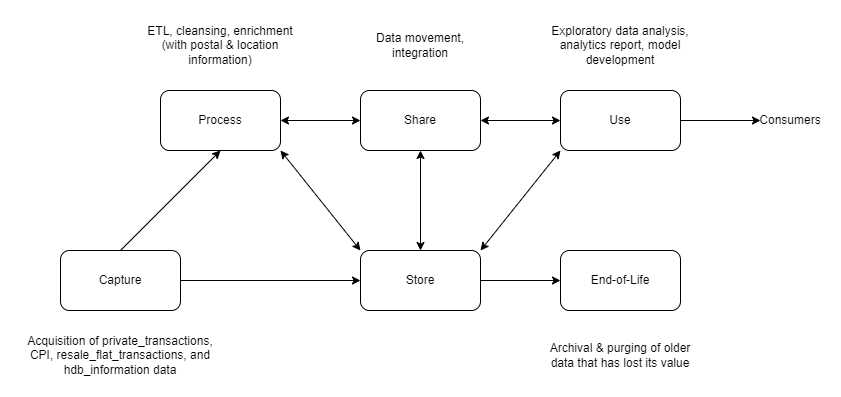
The MLOps sprint aimed to develop, deploy, and monitor the machine learning model for predicting HDB resale flat prices. The sprint goal was to create an efficient and reliable MLOps pipeline that encompassed model development, experimentation, deployment, monitoring, and governance. The sprint backlog included user stories related to data exploration, feature engineering, model training, evaluation, containerization, CI/CD pipeline setup, monitoring, and model explainability. Team members were allocated tasks based on their skills and the sprint timeline.

## 2.2 Sprint planning for DataOps and MLOps

The detailed sprint planning can be found in our [scrum documentation Google sheet workbook](https://docs.google.com/spreadsheets/d/1lI6CupiEpJVNsM4YY8-4WsVL5lWXVlgjgNTufRkJUq0/edit?usp=sharing). The sheets related to DataOps sprint planning are the sprint 1 sheets, whereas the sheets related to MLOps print planning are the sprint 2 sheets. Both sprint plannings include the sprint goal, sprint backlog with full documentation of user stories, and assignment to team members. Additionally, the planning, daily stand-up, and sprint retrospective can also be found. We also used [Trello as a Kanban board tool](https://trello.com/invite/b/cFCGgCAj/ATTIcee0b85d664e00250d8d1f2665fc7ffcD439C2DB/bt4301-project-sprint-board) (click this link to join and view the Kanban board) and screenshots of how we used it can be found in the appendix below. Additionally, the deliverable of each user story is evidenced using source code and can be found in our Github repo. Our source code can be found under the `source` directory in our Github repo, with the source code for the DataOps sprint found under the `dataops` subdirectory and the source code for the MLOps sprint found under the `mlops` subdirectory.

# 3. DataOps

DataOps is a process that focuses on the data life cycle to enhance data quality and metadata management, with the ultimate aim of providing business-ready data that is quickly available for use. Our team has identified the modern data lifecycle as consisting of the following main stages: capture, process, share, store, use, and end-of-life. The connection between these stages is not linear but each stage can be visited as needed to fulfil the business requirements. This modern data lifecycle adopts the Agile methodology as data is built up in an iterative manner that aims to be flexible to meet the changing business needs. Figure 1 provides a more detailed visualisation of the various stages of our data lifecycle.



*Figure 1: Modern Data Lifecycle*

Additionally, the data pipeline that constructs our final datasets will be explained in greater detail in the following sections of our report. Additionally, each stage of the DataOps process our team has implemented will be illustrated.

## 3.1 Dataset Ingestion

Data was acquired from several Singapore government agencies: URA (Urban Redevelopment Authority), SingStat, MAS (Monetary Authority of Singapore), and GovTech (data.gov.sg) and initially stored in CSV files as our first layer of data loading after ingestion of the data. The data that was ingested had a relatively simple tabular structure and the size of the files were not too big. Hence, we decided against storing the raw data in external third-party data warehouses or data lakes but decided to store the raw data in CSV files as an initial data landing area, for ease of processing before loading the transformed data into our destination data warehouse.

Additionally, due to the nature of our business problem as well as the nature of the data sources, the data we extracted, cleansed, and processed would be structured and can be stored in a relational database. The relational database management system we used was PostgreSQL, which is a server-based, open-source object-relational database management system. Storing the data in a tabular format would also be useful for the later stages of our project, such as the exploratory data analysis and model development phases.

The description of the data ingested can be found in Figure 2 below.

| **Dataset** | **Source** | **No. of observations** | **No. of variables** |
| --- | --- | --- | --- |
| CPI | SingStat | 758 | 2 |
| Postal Districts | URA | 28 | 3 |
| HDB Information | data.gov.sg | 12877 | 25 |
| Private Residential Property Transactions | URA | 131585 | 14 |
| HDB Resale Flat Transactions | data.gov.sg | 176487 | 12 |
| SIBOR (Singapore Interbank Offered Rate) and SORA (Singapore Overnight Rate Average) | MAS | 2549 | 9 |

*Figure 2: Description of datasets ingested*

## 3.2 Data Processing

After acquiring the required data for solving our business problem in its original raw format, our team has identified the following data cleansing and data transformation steps required to improve data quality and ensure that business value can be extracted from the data stored:

### 3.2.1 Data Cleansing

After initial data exploration, our team realised that the raw data ingested from the data sources were already well-formatted and there were no missing values in the data. There were no data correctness or data completeness issues likely due to the strict data quality enforced by the upstream data sources, which are the Singapore government agencies. Hence, after the raw data was ingested, there were no data cleansing steps required.

However, while there were no data cleansing steps implemented, our team adopted a monitoring method. After the completion of the data ingestion job, an email with descriptive statistics containing the measures as listed in Figure 3 below will be sent to the Data Engineers. Additionally in the report, a box plot and histogram will be generated for each numerical variable, and for each categorical variable a bar chart is generated. For some columns, only values within a certain range (for numerical variables) or set (for categorical variables) are deemed valid. More details regarding this email will be provided in section 3.3.1 on data quality and profiling below. If after examining the email, the Data Engineer discovers that there are values out of the range, this will prompt the Data Engineer to investigate the potential reasons behind the anomaly.

| **Numerical Variables** | **Categorical Variables** |
| --- | --- |
| Count | Count |
| Mean | Unique Values |
| Standard Deviation | Top (mode) |
| Minimum | Frequency (of mode) |
| 25th percentile | Missing values |
| 50th percentile | Present values |
| 75th percentile | Mode frequency (%) |
| Maximum |  |
| Data Type |  |
| Missing Values |  |
| Present Values |  |

*Figure 3: List of the descriptive statistics generated*

### 3.2.2 Data Transformation

#### 3.2.2.1 Private Residential Property Transactions

In this dataset, the dates are in the format of either MYY or MMYY which makes it hard to interpret. Several processing steps were taken to convert this to two columns, MM and YYYY, allowing downstream apps to group or sort the data by month or year easily. While private property transactions are not within the scope of our current project, it was included in our ETL pipeline due to their relevance in further developments of the project.

#### 3.2.2.1 Postal Districts

The data was obtained from the URA List of Postal Districts site and stored in a spreadsheet format. To allow the lookup of postal district information, we transformed the dataset to split each row into multiple rows based on the columns that contained lists of data, using the Pandas explode function.

#### 3.2.2.3 HDB Resale Flat Transactions

We formatted the dates and filtered them for the last 5 years. To obtain more information about each transaction, we used the OneMap API to retrieve the other details such as postal code, latitude, and longitude, while caching the retrieved information. This has improved the efficiency of the transformation process to complete. Additionally, to obtain the district from each transaction, we do a lookup from the transformed Postal Districts dataset.

## 3.3 Dataset Integration

As mentioned in the previous section on Dataset Ingestion, our team created a centralised analytics data warehouse in PostgreSQL that stores the processed data from multiple sources. Integrating multiple data sources which have already been transformed into the same data warehouse maximises the value of data as data scientists or data analysts would be able to easily access the data in the same location to conduct their analysis or modelling. Due to the nature of our datasets, where the data stored in each dataset are already structured and from different domains, we did not join the datasets together but loaded them into the data warehouse as in their current state to give the data analysts and data scientists greater flexibility when it comes to analysis or modelling. Furthermore, this would also ensure that data redundancy is reduced.

### 3.3.1 Data Quality and Data Profiling

Before the multiple datasets were loaded into the data warehouse, data profiling was carried out on the transformed data. This refers to the process of collecting statistics and summary information from existing data to generate metadata. Through this data profiling, descriptive summary statistics are generated, which can help the team detect poor data quality and create metadata for further data cleansing or investigations into any anomalies in the upstream data sources. Our team conducted data profiling for both numerical variables and categorical variables separately. The descriptive statistics generated can be seen in Figure 2 above. Additionally, for each numerical variable, a box plot and histogram are generated, and for each categorical variable, a bar chart is generated. Examples of these visualisations can be seen in the appendix below. While histograms and boxplots generally provide a visual representation of the distribution of numerical variables, they are both required and useful as they reveal different aspects of the data distribution. Histograms provide a detailed view of the distribution of the data, giving a more granular view of how data is spread across different ranges. Additionally, histograms are useful in assessing the shape of the distribution to determine if the distribution is symmetrical, skewed, etc. On the other hand, boxplots not only provide a concise summary of the distribution of the data using the summary statistics of minimum, first quartile, median, third quartile, and maximum, which allow Data Engineers to better understand the central tendency and spread of the data, but they are also particularly effective for outlier detection as outliers are shown as points beyond the whiskers of the boxplot.

These statistics and visualisations would allow the Data Engineers to visualise the distribution of both numerical and categorical values and check if any anomalies need to be rectified. Our team created a job to send these statistics and charts would be sent to the Data Engineers in charge of the data after every scheduled job to ingest and transform the source data. If low-quality data has been detected by the Data Engineers, it would be possible to investigate the source of the anomalies, apply any data cleansing or data transformation jobs, and backfill the data for the affected dates. As can be seen during both the data ingestion and data pre-processing stages, data profiling metadata is captured and monitored continuously along the entire data pipeline.

## 3.4 Data Lineage

As our team is not using commercial BI or ETL tools due to the scale of our project, lineage information is not automatically captured as data is processed. We have documented the business-level lineage, which is a more manual process of creating and maintaining the data lineage, but one we deem is suitable as the nature of our business and data sources are unlikely to change drastically shortly, and we will unlikely need to make many manual changes to this data lineage diagram. This data lineage diagram can be found in Figure 4 below.



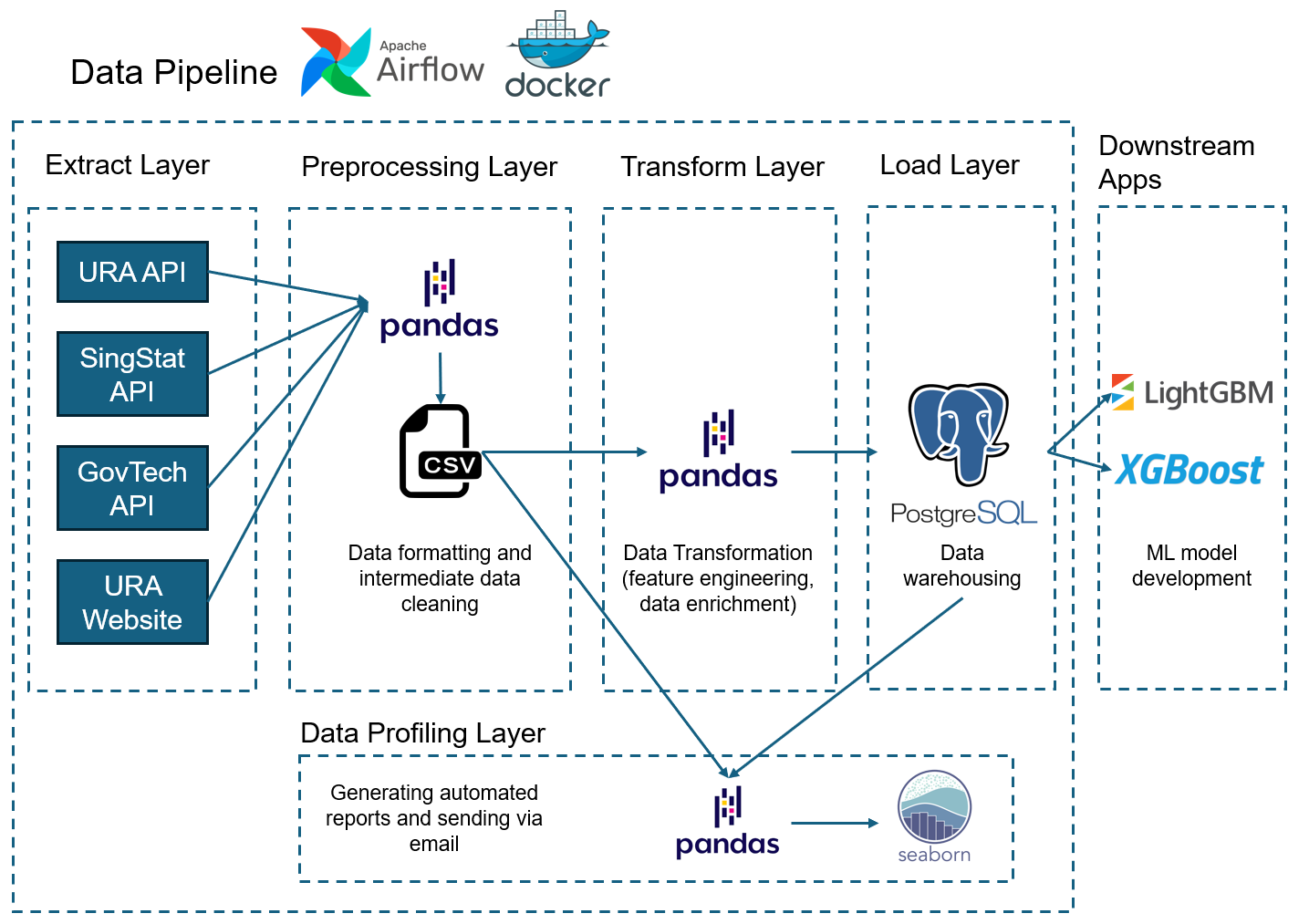
*Figure 4: Data Lineage Diagram*

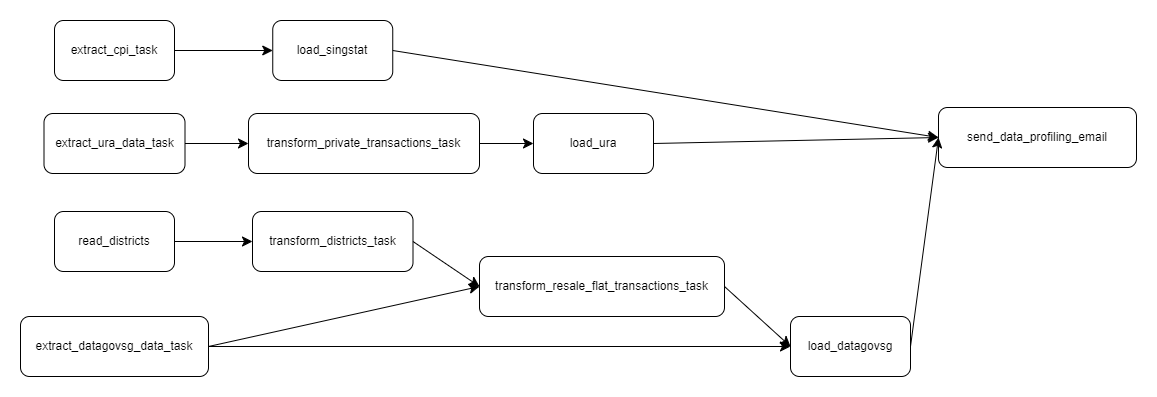
## 3.5 Automation and orchestration of DataOps process

Our team used Airflow, an open-source workflow management platform for data engineering pipelines, for the automation and orchestration of our DataOps processes. This allowed us to streamline workflows, reduce manual intervention, and ensure consistent execution of data extraction, data transformation, data loading, and data quality check jobs.

As mentioned in the previous sections, our data pipeline would first extract data from URA, GovTech, MAS, and SingStat data sources. Following this extraction, the data would be pre-processed. Data quality checks would be run to highlight any anomalies in the data with automated reports being generated and sent in email alerts to Data Engineers. This would then allow Data Engineers to intervene manually if required, as such data cleansing processes due to data anomalies will be judged on a case-by-case basis, and hence should not be automated but highlighted to Data Engineers to investigate manually. Following this, data would be transformed and enriched with additional location (OneMap) data before being loaded to the destination PostgreSQL data warehouse. As the data is loaded, data profiling jobs would also be executed, with descriptive statistics visualisations being generated and sent in email alerts to Data Engineers for them to have a quick overview of the most recent data loaded into the data warehouse.

An overview of the overall pipeline can be found in Figure 5 below while the specific pipeline tasks and DAG can be found in Figure 6 below.

*Figure 5: Overview of Data Pipeline and Technologies used*



*Figure 6: Data Pipeline DAG*

# 4. MLOps

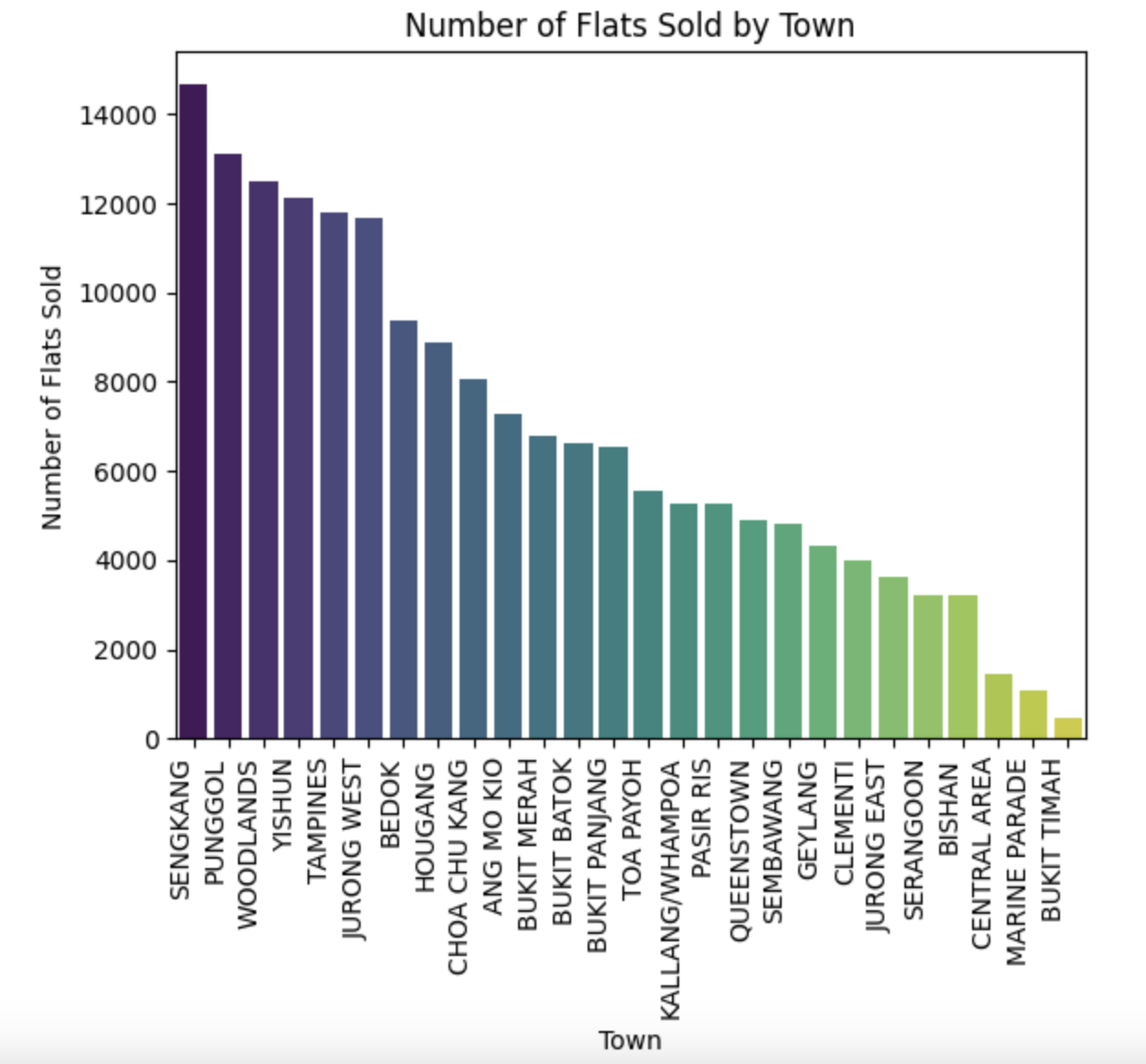
This portion of the project discusses the implementation of the MLOps pipeline, illustrated in Figure 6.

## 4.1 Model Development

*Figure 7: MLOps Pipeline*

### 4.1.1 Data Exploration

We utilised exploratory data analysis to learn more about our data, including overall trends and information about the features. Figure 7 visualises the breakdown of the number of flats sold by the town in the last five years.



*Figure 8: Quantity of flats sold by town*

### 4.1.2 Feature Engineering

#### 4.1.2.1 Encoding

The categorical features in the dataset were encoded for input to the models. The type of encoding used was feature-specific to ensure that the inherent information within each feature is preserved. For instance, the feature *flat\_type* is determined by the flat’s number of rooms, indicating that there is an inherent hierarchy in the different types. This feature was transformed with ordinal encoding. Other features such as the flat model were transformed with one-hot encoding as they do not have a set order.

#### 4.1.2.2 Window Aggregations

The CPI and SIBOR rate data were lagged and aggregated into simple and exponential moving averages for three, six and twelve months to create features. Lag features are useful for predictive modelling as they provide the model with information about the variable's past behaviour, and moving averages create more stable features due to smoothing out fluctuations and noise.

### 4.1.3 Experimentation with MLFlow

#### 4.1.3.1 Model Training

The data was split into train and test sets by date to maintain its temporal order. The model was trained with data ranging from 2017 to 2022, and evaluated with data from 2023 onwards.

#### 4.1.3.2 Model Evaluation

We utilised three metrics to evaluate the models’ performance, namely the mean absolute error, the mean squared error, and the mean absolute percentage error. We prioritised the mean absolute error (MAE) metric as it calculates the absolute values of the differences between the predicted and actual value, making it inherently more robust to outliers as compared to the MSE.

The XGBoost model yielded significantly better performance compared to the LightGBM model, with an MAE of 0.196.

#### 4.1.3.3 Reproducibility and Version Control

For reproducibility, our tracked experiments are automatically saved into a folder that contains artifacts and runs. Our folder is tracked by Git, which is pushed into a central repository provider (GitHub). This ensures the reproducibility of our experiments and artifacts seamlessly.



*Figure 9: Showing the versioned artifacts that are stored for reproducibility*

## 4.2 Preparing for Production

### 4.2.1 Runtime Environments and Considerations

In our case, we segregate our development and production workflows into their respective runtimes. In both cases, they would both be running as containers (section 4.3).

The tooling considerations are also straightforward when dev and prod environments are containers, abstractions of the same platform that they are based on, MLFlow serving. Thus no conversions are needed. First, the models are saved as .pkl files as a result of MLFlow experiments. Then, they are packaged as containers for serving.

As for performance considerations, We do not need to perform model compression because, in XGBoost, the model is trained in a way that reduces information loss. In other words, XGBoost already outputs an optimised model

### 4.2.2 Model Risk Evaluation

If the model performs poorly, it could lead to inaccurate predictions of housing prices. This could result in the company making poor investment decisions, such as overpaying for properties or missing out on profitable opportunities.

Inaccurate predictions could lead to poor investment decisions, financial losses, and disruptions to business operations. For example, if the model overestimates housing prices, the house owner could end up being unable to sell the house for a long time, resulting in opportunity costs such as rent. For potential buyers, the overestimated house prices could incur financial losses, such as them taking up larger loans than necessary to purchase the house. These losses for the company’s customers could damage the company’s reputation in the long run.

### 4.2.3 Quality Assurance and Model Testing

Artificial data with various data types, extreme values, and null values is created for quality assurance purposes to test the data pipeline's ability to handle diverse scenarios. By including different data types, the pipeline is tested for its flexibility in processing various data structures. Extreme values test the pipeline's ability to handle outliers effectively, while null values test its capability to manage missing data without errors. Testing the pipeline with this artificial data helps identify and address potential issues, ensuring its reliability and robustness in handling real-world data.

### 4.2.4 Reproducibility and Audibility

MLflow's Model Registry component allows the logging of datasets and model files, ensuring that all relevant files are saved and accessible for future reference. Its Tracking feature together with the versioning function tracks all runs, including parameters, metrics, and artifacts, providing a clear overview of experiment progress and allowing for comparison of results. These features can support the company in maintaining reproducibility, audibility, and trustworthiness in its machine-learning operations.

### 4.2.5 Risk Mitigation

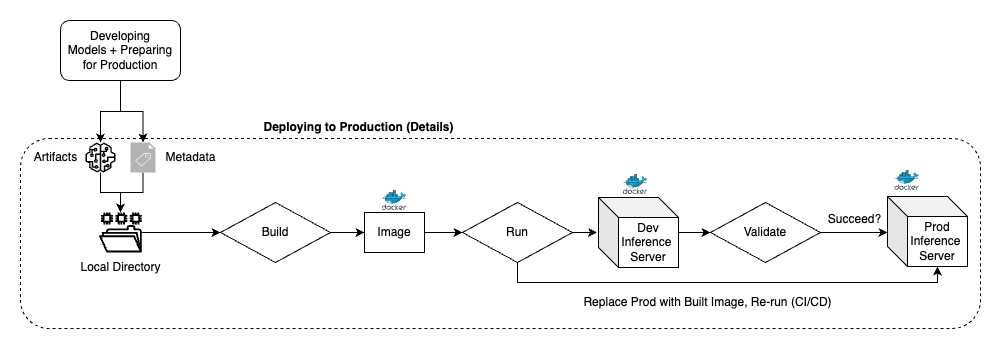
In the Singapore housing market, house prices can change rapidly due to government policies, particularly in the resale HDB market. It's important to address the challenges posed by rapidly changing environments to maintain profitability. To mitigate these risks, the company's MLOps strategy should be responsive enough to trigger alerts and take necessary actions, such as temporarily halting the model's operation, retraining the model with updated data, or making adjustments to the model to better adapt to the new environment. This proactive approach helps ensure the model's effectiveness and reliability amidst potentially drastic market conditions, minimising the impact of unexpected changes on its performance.

## 4.3 Deploying to Production

### 4.3.1 Containerisation and Scaling Deployments

To create model-serving environments, we conceptualise our development and production environments as separate running containers that perform model-serving. Such deployments can be scaled easily through container management platforms such as Kubernetes - but are out of the scope of our defined MLOps sprint.

### 4.3.2 CI/CD Pipeline and Testing

To simulate a CI/CD pipeline, we define a development environment and a production environment as running containers. The abstraction provided by containerisation makes it suitable for our simulation of a CI/CD pipeline

*Figure 10: Diagram showing a simulation of a CI/CD pipeline trigger when the latest model artifact is pushed into the repository - triggering the build and test process*

Our CI/CD workflow is defined with the following steps as can be seen from the figure 10:

1. Deploy the development model server by running the development model server container
2. Run tests on the development model server
3. If any of the tests fail, stop. Otherwise, use the development model server image to run as a production model server container - thereby deploying to production

The testing pipeline consists of automated tests on the model pipeline with various data types, including extreme or null values. Any entries with invalid data types or extreme values are ignored and an error message will be printed to indicate the corresponding error. This approach allows the testing pipeline to identify and address issues related to data type compatibility and the model's ability to handle extreme or unexpected inputs, ensuring its robustness and reliability in production environments..

## 4.4 Monitoring and Feedback Loop

For the monitoring portion of our project, we implemented model performance monitoring through ground truth evaluation, and also input drift monitoring through measuring data drift of the features and target variable. The feedback of the monitoring part is done through an email alert to the relevant team members. The monitoring and feedback process is done through a script, which is intended to be scheduled to run at a consistent interval.

### 4.4.1 Ground Truth Evaluation

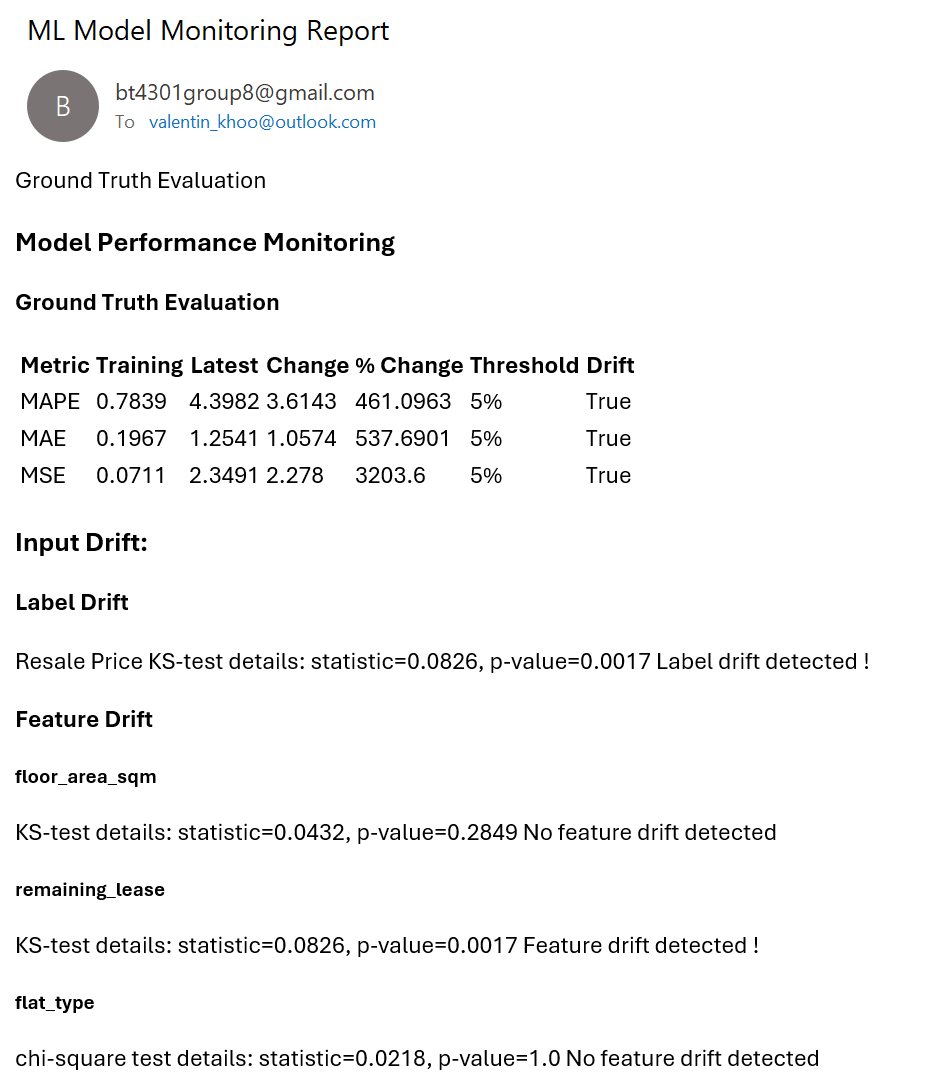
We initially logged the relevant key metrics during the model development phase, in the earlier segment of the MLOps process. These metrics are the MSE, MAE and MAPE. For our implementation or simulation of ground truth evaluation in this project, we simulated the model making predictions on unseen data (using a subset of the most recent data), using the deployed prediction model as mentioned in the earlier stage. As the model makes predictions for the set of input data, the features and predictions are saved temporarily to a table, which constitutes one set of new predictions. We then simulated collecting the ground truth for this set, where we obtained the ground truth from the actual resale prices. Once ground truth has been collected, we then evaluate the model performance with the key metrics on this new set of data. If any of the error metrics exhibit a degradation of 5% or greater over its value during training, this information will be presented through an email alert sent out to notify the team, in the feedback loop stage.

### 4.4.2 Input Drift Detection

In addition to monitoring model performance through ground truth evaluation, we also monitored input data drift in the form of feature and label drift. We believe this is an essential component because any drift in inputs would most likely affect the performance of the model over time, as what is learnt on the initial dataset may not be applicable to another dataset that has drifted. For the continuous features and label, we utilise the two-sample Kolmogorov-Smirnov test to compare the underlying distribution between corresponding features of the initial and the new dataset. For categorical features, we used the chi-square test on the value counts of the corresponding features for the initial and new dataset. For the label and features, if they meet the criteria of being considered as significantly different, this information will be recorded and presented in an email alert to the team.

### 4.4.3 Feedback Loop

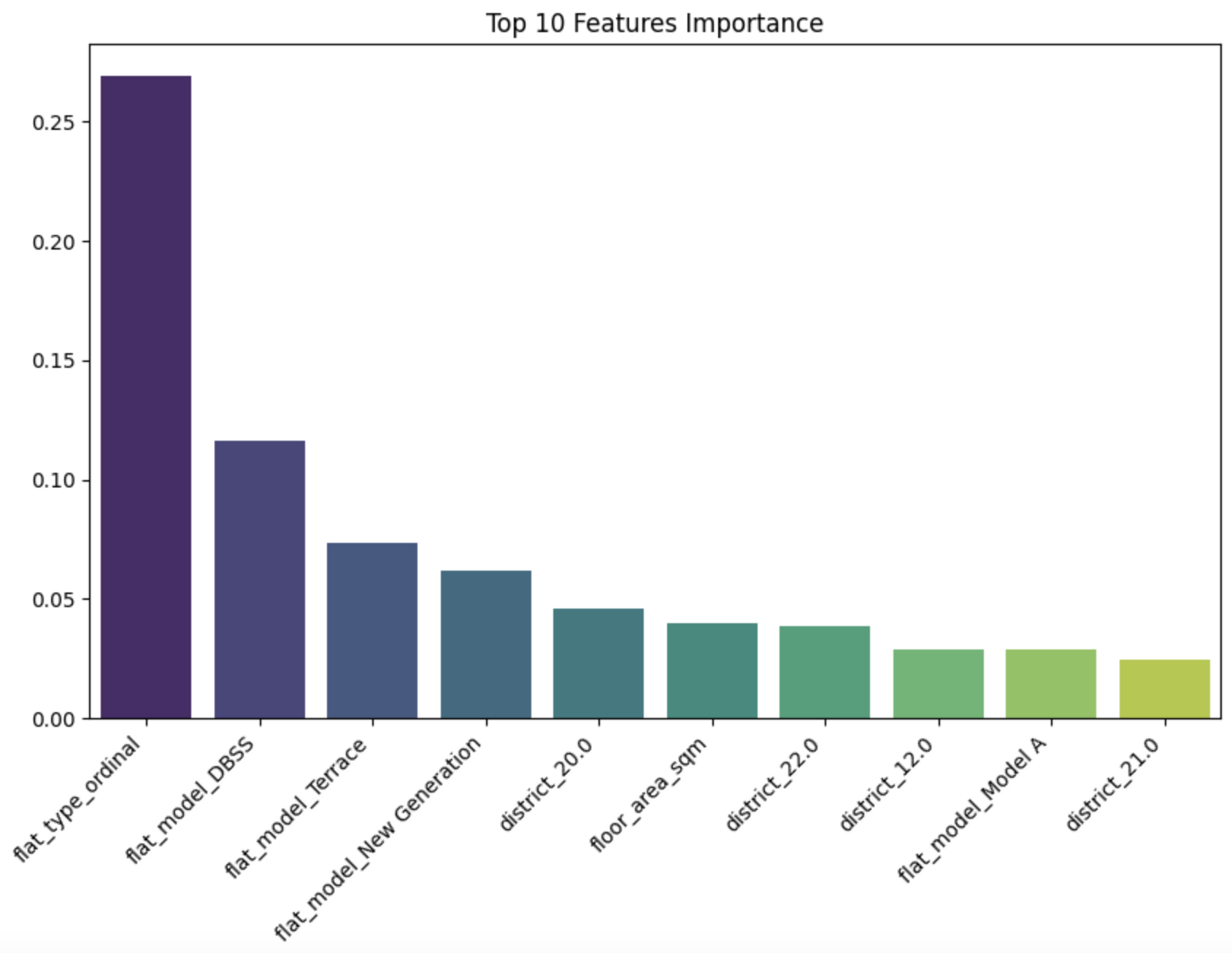
After the monitoring steps have completed, the information throughout every stage is recorded and prepared to be presented. These information consisting of ground truth evaluation details for each metric, input drift consisting of label drift and feature drift, is presented in an email that is sent to the relevant machine learning team. The relevant results of the statistical tests are included as well. An example of one such email that we have implemented is shown in the screenshot below. As this project utilises agile principles, our aim at this stage was to create an email that has sufficient details for the team. Should we, hypothetically, receive feedback on the email format or the way the information is presented, we would make improvements to the email template in subsequent sprints.



*Figure 11: Example of ML Model Monitoring Report*

## 4.5 Model Governance

As a self-service analytics project to be consumed by an internal audience, the model requires relatively lightweight governance. In addition to monitoring and drift management, we explored model explainability through the use of XGBoost’s feature importance. Figure 12 shows the ten most useful features based on their contribution to the final model prediction.



*Figure 12: Feature importances from XGBoost model that is planned to be used in production*

# 5. Conclusion

This project successfully developed an end-to-end analytics solution to predict HDB resale flat prices in Singapore by leveraging data from various government sources. The project followed an agile approach with scrum methodology, effectively managing DataOps and MLOps sprints.

The DataOps process ensured high-quality data by implementing a modern data lifecycle, which included data ingestion, processing, integration, and lineage. Data cleansing, transformation, and profiling were performed to ensure data quality and detect anomalies. The process was automated and orchestrated using Airflow, streamlining workflows and reducing manual intervention.

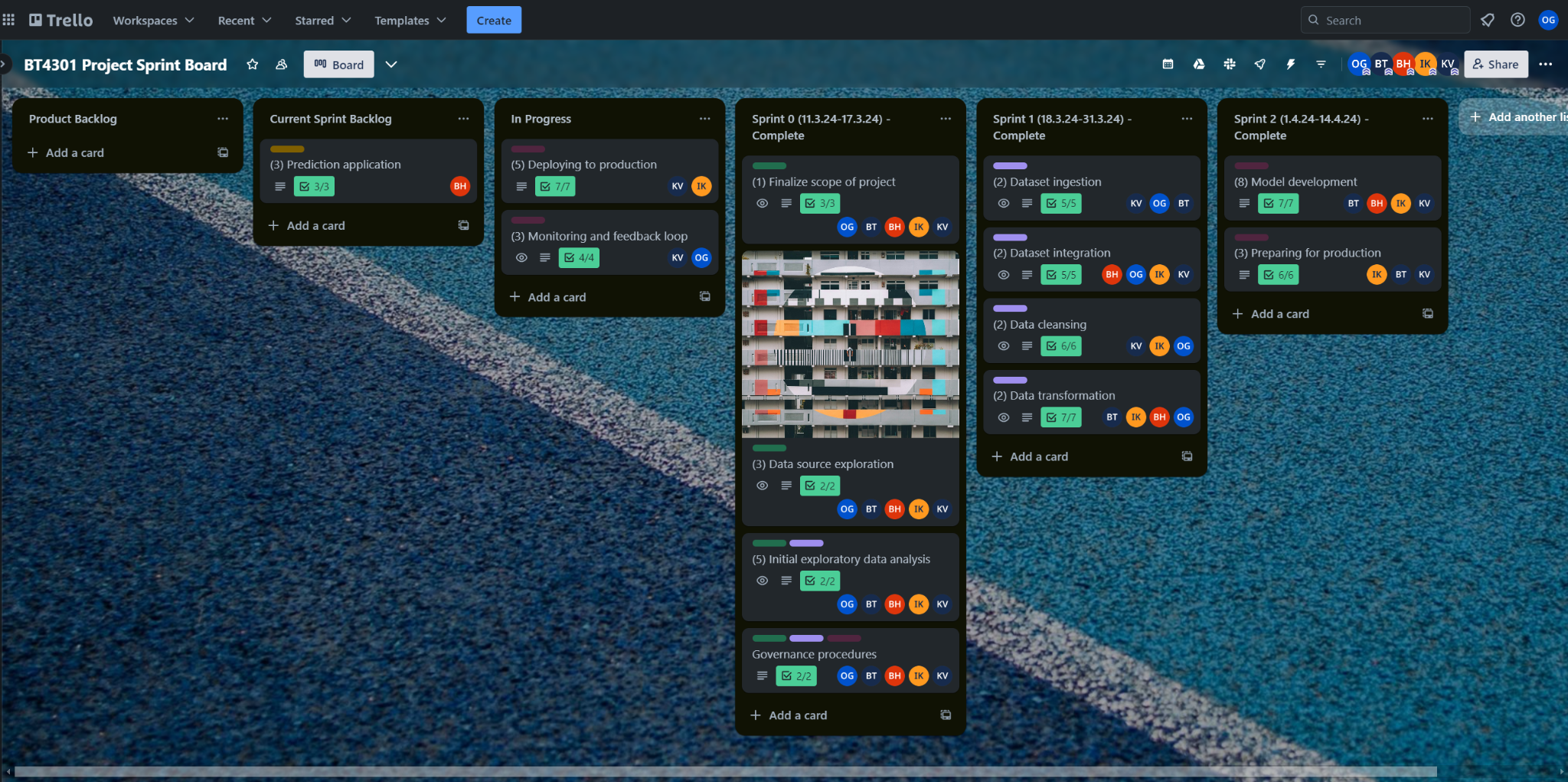
The MLOps pipeline focused on model development, preparation for production, deployment, monitoring, and governance. Exploratory data analysis and feature engineering were conducted to gain insights and prepare data for modelling. Experimentation with MLFlow allowed for efficient model training, evaluation, and version control. The best-performing model, XGBoost, was containerized and deployed in a simulated CI/CD pipeline with automated testing. A monitoring and feedback loop was established to detect model performance deviations and data drift, triggering alerts when necessary. Model governance was addressed through explainability using feature importance.

The developed solution provides valuable insights for real estate agencies, enabling them to make data-driven decisions, mitigate risks, and enhance their services. By leveraging this analytics solution, agencies can reduce information asymmetry, identify lucrative investment opportunities, and strengthen their brand image as experts in the real estate industry.

Future work could involve expanding the solution to cover other segments of the housing market. For instance, in our DataOps pipelines, we have pulled data related to private property transactions. Similar models can be developed and deployed, providing greater insights for real estate agencies. Continuous monitoring and refinement of the solution will ensure its relevance and effectiveness in the dynamic Singapore housing market.

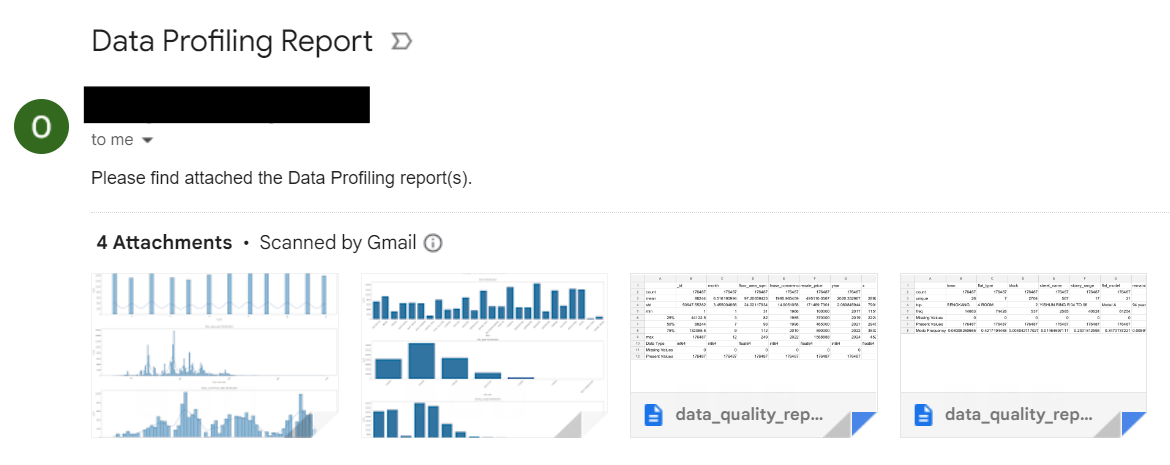
# 6. Appendix

## 6.1 Kanban board

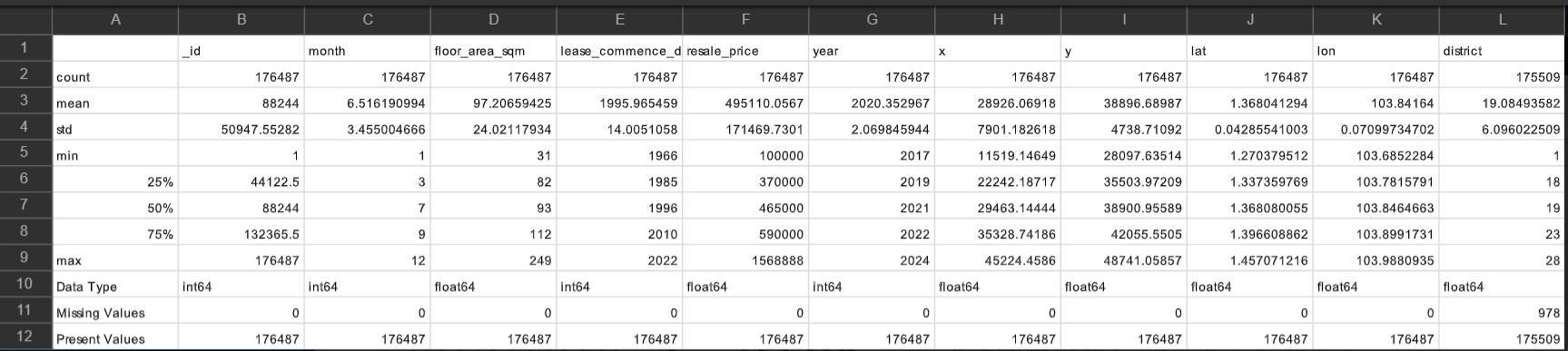


## 6.2 Data quality and profiling email report

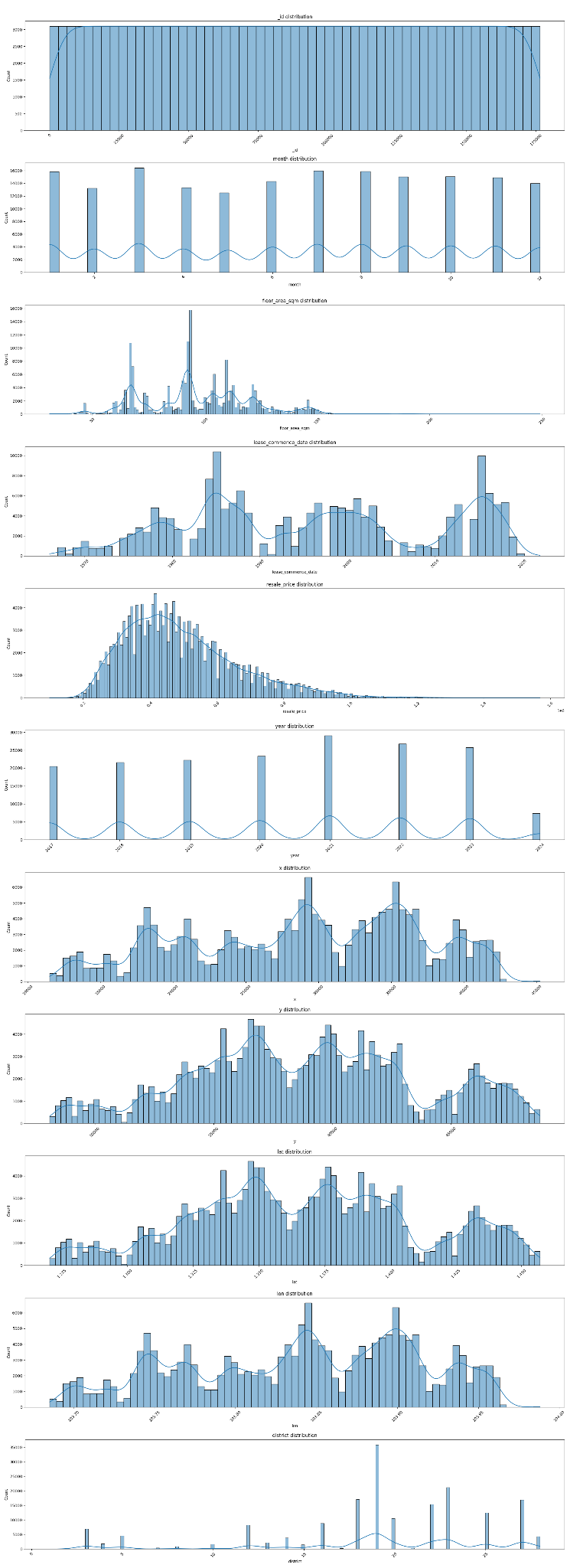
### 6.2.1 Example of data profiling email



### 6.2.2 Example of descriptive statistics generated



### 6.2.3 Example of visualisations generated for numerical variables



### 6.2.4 Example of visualisations generated for categorical variables

