# CREDIT CARD FRAUD DETECTION WITH MACHINE LEARNING

## Phase 1: Problem Definition and Design Thinking

### PROBLEM DEFINITION:

The goal is to create a solution that can accurately identify fraudulent transactions while minimizing false positives. This project involves data preprocessing, feature engineering, model selection, training, and evaluation to create a robust fraud detection system. Credit card fraud is the act of using another person's credit card to make purchases or request cash advances without the cardholder's knowledge or consent. The Machine learning model aims to reduce the vulnerabilities and implement the Rule-Based Filters, Velocity Checks and some more features.

### DESIGN THINKING:

### Data collection:

The first step is to collect data that is relevant to the product demand prediction task. This data may include historical sales data, pricing data, market trends data, and other factors that may affect demand. Once the data is collected, it needs to be cleaned and prepared for modeling

The given dataset:

https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

### Data Preprocessing:

Prepare the data. Once you have gathered your data, you need to prepare it for machine learning. This may involve cleaning the data, removing outliers, and converting the data into a format that is compatible with your chosen machine learning algorithm. Once the data is collected, it needs to be cleaned and prepared for modeling.

### Feature Engineering:

Feature engineering is a critical step in building a credit card fraud detection model. It involves creating relevant features from the raw transaction data that can be used by machine learning algorithms to detect fraud effectively. Here are some common feature engineering techniques for this type of model:

1. **Transaction Amount**: Use the transaction amount as a feature. Large, unusually high or low transaction amounts may be indicative of fraud.

2. **Transaction Time**: Extract the time of the transaction, including the hour of the day, day of the week, and potentially the month. Fraudulent activity may exhibit specific temporal patterns.

3. **Merchant Information**: Incorporate information about the merchant, including the merchant category, location, and whether it's an online or in-person transaction. Fraudsters often target specific types of merchants or regions.

4. **Cardholder Information**: Include features related to the cardholder, such as their account history, spending patterns, and the number of cards associated with the account.

5. **Transaction Frequency**: Calculate the frequency of transactions for each cardholder, both in terms of the number of transactions and the time between transactions.

6. **Transaction Sequence**: Analyze sequences of transactions for each card, looking for patterns or anomalies in the order of transactions.

7. **Geographical Information**: Use geographical features like the country, city, or region where the transaction occurred. Compare this information to the cardholder's known location.

8. **Velocity and Frequency Checks**: Create features that calculate the velocity of transactions (e.g., the number of transactions per minute or hour) and the frequency of transactions within specific time windows.

9. **Card Features**: Consider features related to the card itself, such as the card type (credit, debit), the issuing bank, and the card's expiration date.

10. **Previous Transactions**: Incorporate information about the cardholder's recent transaction history, including the average transaction amount, standard deviation, and other statistics.

## Model Selection:

There are many different machine learning algorithms that can be used for credit card fraud detection. Some popular choices include Logistic Regression, Random Forest, Gradient Boosting.

Our choice of model is Logistic Regression. Logistic Regression is a popular and interpretable machine learning algorithm that can be effectively used in credit card fraud detection

## Model Training:

Once you have chosen a machine learning algorithm, you need to train it on your historical data. This process will teach the algorithm to identify patterns and trends in the data Once the machine learning algorithm is chosen, the next step is to train the model on the historical data. This involves feeding the data to the algorithm and allowing it to learn the relationships between the different variables

## Evaluation:

Once the model is trained, it is important to evaluate its performance on a held-out test set. This will help to assess how well the model will generalize to new data. it is important to evaluate its performance on a held-out test set. This will help to ensure that the model is generalizing well and is not simply overfitting the training data. This will give you an idea of how well the model will generalize to new data.