## **Project Architecture**

The provided code demonstrates a machine learning pipeline for building a Random Forest Classifier to predict customer reviews (positive or negative) based on order data. Here's a breakdown of the architecture:

**1. Data Loading:**

* Several datasets related to orders, products, sellers, reviews etc. are loaded using pandas.read\_csv.

**2. Data Merging:**

* The datasets are merged sequentially based on relevant keys (e.g., order\_id, product\_id) to create a comprehensive dataset data.

**3. Data Preprocessing:**

* Missing values are removed using data.dropna().
* The review\_score is converted to a binary classification target (positive: 1, negative: 0) using a lambda function.

**4. Feature Engineering:**

* Specific features are selected from the merged data for model training:
  + product\_id
  + seller\_id
  + price
  + freight\_value
  + product\_category\_name
* Categorical variables like product\_category\_name are encoded using one-hot encoding with pd.get\_dummies.

**5. Data Splitting:**

* The data is split into training and testing sets using train\_test\_split.
* This ensures the model is trained on a portion of the data and evaluated on unseen data.

**6. Data Normalization:**

* A StandardScaler from sklearn.preprocessing is used to normalize the features in the training and testing sets.
* Normalization helps improve model performance by scaling features to a similar range.

**7. Model Training (not shown in current code):**

* A RandomForestClassifier from sklearn.ensemble will likely be instantiated with desired parameters.
* The model will be trained using the training data (X\_train, y\_train).

**8. Model Evaluation (not shown in current code):**

* The trained model will be used to make predictions on the testing data (X\_test).
* Evaluation metrics like accuracy score and classification report will be calculated using sklearn.metrics to assess the model's performance.