**1. Data Collection and Preprocessing**

* **Steps:**
  1. Find a dataset like the [SpamAssassin Public Corpus](https://spamassassin.apache.org/) or from [Kaggle](https://www.kaggle.com/uciml/sms-spam-collection-dataset).
  2. Load the dataset using pandas for analysis.
  3. Clean the data:
     + Remove duplicates and null values.
     + Preprocess text (lowercase conversion, remove punctuation and stop words, and stemming/lemmatization).
* **Key Tools:**
  1. Python libraries: pandas, nltk, re (regular expressions).
* **Output:** A clean dataset with emails or messages prepared for feature extraction.

**2. Feature Extraction**

* **Steps:**
  1. Convert the text data into numerical format using techniques like:
     + **Bag of Words (BoW):** Create a frequency matrix.
     + **TF-IDF (Term Frequency-Inverse Document Frequency):** Assign importance to words based on frequency and uniqueness.
  2. Use CountVectorizer or TfidfVectorizer from sklearn to transform the data.
* **Key Tools:**
  1. Python libraries: scikit-learn.
* **Output:** A numerical representation of text data that can be fed into a machine learning model.

**3. Model Training and Evaluation**

* **Steps:**
  1. Split the data into training and testing sets using train\_test\_split.
  2. Train a classification model (e.g., Logistic Regression, Naive Bayes, Random Forest).
  3. Evaluate the model using metrics like:
     + Accuracy
     + Precision
     + Recall
     + F1-score
* **Key Tools:**
  1. Python libraries: scikit-learn.
* **Output:** A trained model that can classify emails or messages as spam or not spam.

**4. Deployment and Documentation**

* **Steps:**
  1. Save the model using joblib or pickle.
  2. Create a simple interface for your model:
     + Use a Jupyter Notebook to showcase the workflow.
     + Build a small web app using Streamlit or Flask.
  3. Document the entire process:
     + Explain your approach in a README file.
     + Include code, visuals, and examples in your portfolio or GitHub.
* **Key Tools:**
  1. Python libraries: joblib, pickle, Streamlit/Flask.
* **Output:** A user-friendly, deployed application or a well-documented GitHub repository showcasing your project.