Product Requirements and Specification Document

Project Name

ClassifAI - Logistic Regression Email Spam Classifier

Description

ClassifAI is a Streamlit web application that allows users to input email text and receive a real-time classification (spam or not spam) using a logistic regression model. The app is designed for educational purposes, demonstrating basic machine learning concepts.

---

1. Goals & Objectives

Goal	Description
Educational Tool	Demonstrate logistic regression for text classification
User-Friendly Interface	Simple, intuitive input and output for non-technical users
Real-Time Prediction	Immediate feedback on email classification

---

2. Target Users

User Type	Description
Students	Learning about ML and text classification
Educators	Teaching ML concepts interactively
General Audience	Exploring spam detection basics

---

3. Functional Requirements

ID	Requirement
FR1	User can input email text via a text box
FR2	App displays classification result: “Spam” or “Not Spam”
FR3	App shows model confidence/probability score
FR4	Logistic regression model is trained on a standard spam dataset
FR5	Model and preprocessing pipeline are loaded at app startup
FR6	App provides brief educational content on logistic regression and spam
FR7	App handles invalid/empty input gracefully

---

4. Non-Functional Requirements

ID	Requirement
NFR1	App loads and predicts within 2 seconds
NFR2	UI is responsive and accessible
NFR3	Codebase is modular and well-documented
NFR4	App runs locally with minimal dependencies

---

5. Technical Specifications

5.1 Architecture Overview

- **Frontend:** Streamlit web app
- **Backend:** Python (scikit-learn, pandas)
- **Model:** Logistic Regression (scikit-learn)
- **Data:** Public spam email dataset (e.g., UCI SMS Spam Collection)

5.2 Data Flow

```
flowchart TD
    A[User Input Email Text] --> B[Preprocessing]
    B --> C[Vectorization (e.g., CountVectorizer)]
    C --> D[Logistic Regression Model]
    D --> E[Prediction Output]
```

5.3 Key Components

Component	Technology	Description
UI	Streamlit	Input box, result display, info sections
Preprocessing	pandas, sklearn	Text cleaning, vectorization
Model	scikit-learn	Logistic regression, trained offline
Data	pandas	Load and process dataset

---

6. User Interface (UI) Requirements

Element	Description
Email Input Box	Multiline text area for user input
Predict Button	Triggers classification
Result Display	Shows “Spam” or “Not Spam” and confidence score
Educational Section	Brief info on logistic regression and spam filtering
Error Handling	User-friendly messages for invalid input

7. Success Criteria

Metric	Target
Prediction Accuracy	≥ 90% on test set
App Response Time	≤ 2 seconds per prediction
User Satisfaction	Positive feedback from test users
Code Quality	Passes code review and basic tests

8. Out of Scope

- No user authentication or data storage
- No email attachments or HTML parsing
- No advanced model tuning or deep learning

9. Milestones & Timeline

Milestone	Description	Target Date
Data Preparation	Dataset cleaning & preprocessing	Week 1
Model Training	Train & validate logistic regression	Week 1
Streamlit UI Development	Build core app interface	Week 2
Integration & Testing	Connect model, test app	Week 2
Documentation	User guide & code comments	Week 3
Release	Final app delivery	Week 3

10. Dependencies

Dependency	Version/Notes
Python	3.8+
Streamlit	Latest stable
scikit-learn	Latest stable
pandas	Latest stable

11. Acceptance Criteria

- App runs locally with `streamlit run app.py`
- User can input text and receive a prediction
- Model achieves ≥ 90% accuracy on test data

- UI is clear, responsive, and error-tolerant
- Code is documented and modular

End of Document

Product Requirements and Specification Document

Project Name

ClassifAI - Logistic Regression Email Spam Classifier

Description

ClassifAI is a Streamlit web application that allows users to input email text and receive a real-time classification (spam or not spam) using a logistic regression model. The app is designed for educational purposes, demonstrating basic machine learning concepts.

---

1. Goals & Objectives

Goal	Description
Educational Tool	Demonstrate logistic regression for text classification
User-Friendly Interface	Simple, intuitive input and output for non-technical users
Real-Time Prediction	Immediate feedback on email classification

---

2. Target Users

User Type	Description
Students	Learning about ML and text classification
Educators	Teaching ML concepts interactively
General Audience	Exploring spam detection basics

---

3. Functional Requirements

ID	Requirement
FR1	User can input email text via a text box
FR2	App displays classification result: “Spam” or “Not Spam”
FR3	App shows model confidence/probability score
FR4	Logistic regression model is trained on a standard spam dataset
FR5	Model and preprocessing pipeline are loaded at app startup
FR6	App provides brief educational content on logistic regression and spam
FR7	App handles invalid/empty input gracefully

---

4. Non-Functional Requirements

ID	Requirement
NFR1	App loads and predicts within 2 seconds
NFR2	UI is responsive and accessible
NFR3	Codebase is modular and well-documented
NFR4	App runs locally with minimal dependencies

---

5. Technical Specifications

5.1 Architecture Overview

- **Frontend:** Streamlit web app
- **Backend:** Python (scikit-learn, pandas)
- **Model:** Logistic Regression (scikit-learn)
- **Data:** Public spam email dataset (e.g., UCI SMS Spam Collection)

5.2 Data Flow

```
flowchart TD
    A[User Input Email Text] --> B[Preprocessing]
    B --> C[Vectorization (e.g., CountVectorizer)]
    C --> D[Logistic Regression Model]
    D --> E[Prediction Output]
```

5.3 Key Components

Component	Technology	Description
UI	Streamlit	Input box, result display, info sections
Preprocessing	pandas, sklearn	Text cleaning, vectorization
Model	scikit-learn	Logistic regression, trained offline
Data	pandas	Load and process dataset

---

6. User Interface (UI) Requirements

Element	Description
Email Input Box	Multiline text area for user input
Predict Button	Triggers classification
Result Display	Shows “Spam” or “Not Spam” and confidence score
Educational Section	Brief info on logistic regression and spam filtering
Error Handling	User-friendly messages for invalid input

7. Success Criteria

Metric	Target
Prediction Accuracy	≥ 90% on test set
App Response Time	≤ 2 seconds per prediction
User Satisfaction	Positive feedback from test users
Code Quality	Passes code review and basic tests

8. Out of Scope

- No user authentication or data storage
- No email attachments or HTML parsing
- No advanced model tuning or deep learning

9. Milestones & Timeline

Milestone	Description	Target Date
Data Preparation	Dataset cleaning & preprocessing	Week 1
Model Training	Train & validate logistic regression	Week 1
Streamlit UI Development	Build core app interface	Week 2
Integration & Testing	Connect model, test app	Week 2
Documentation	User guide & code comments	Week 3
Release	Final app delivery	Week 3

10. Dependencies

Dependency	Version/Notes
Python	3.8+
Streamlit	Latest stable
scikit-learn	Latest stable
pandas	Latest stable

11. Acceptance Criteria

- App runs locally with `streamlit run app.py`
- User can input text and receive a prediction
- Model achieves ≥ 90% accuracy on test data

- UI is clear, responsive, and error-tolerant
- Code is documented and modular

End of Document

Low Level Design Document

Introduction

This Low Level Design (LLD) document outlines the implementation details for **ClassifAI - Logistic Regression Email Spam Classifier**. The project is a Streamlit web application that allows users to input email text and receive a spam/not-spam classification using a logistic regression model.

---

1. System Components

Component	Description	Key Responsibilities
UI Layer (Streamlit)	Web interface for user input/output	Collect input, display results
Model Layer	Logistic regression classifier (scikit-learn)	Predict spam/not-spam
Preprocessing Layer	Text cleaning and vectorization	Prepare input for model
Data Layer	(Optional) Load model/vectorizer artifacts	Persist/load trained objects

---

2. Class/Interface Overview

Class/Module	Description	Key Methods/Attributes
SpamClassifier	Encapsulates model & vectorizer	`predict(email_text: str) -> str`
Preprocessor	Handles text preprocessing	`transform(text: str) -> vector`
app.py (Streamlit)	Main app script	`main()`

Relationships:

- `app.py` uses `SpamClassifier` for predictions.
- `SpamClassifier` uses `Preprocessor` for input transformation.

---

3. Data Structure Overview

Data Model	Structure/Type	Description
Email Input	`str`	Raw email text from user
Feature Vector	`scipy.sparse` or `np.ndarray`	Vectorized email for model input
Prediction Output	`str` (`"Spam"` / `"Not Spam"`)	Classification result

---

4. Algorithms/Logic

Prediction Flow (Pseudocode):

```
def predict(email_text: str) -> str:
    cleaned = Preprocessor.transform(email_text)
    vector = vectorizer.transform([cleaned])
    label = model.predict(vector)
    return "Spam" if label == 1 else "Not Spam"
```

Streamlit App Flow:

1. User enters email text.
2. On submit, call `SpamClassifier.predict()`.
3. Display result to user.

---

5. Error Handling

Scenario	Handling Approach
Empty input	Show validation error in UI
Model/vectorizer load failure	Log error, display generic error message
Unexpected prediction error	Catch exception, show error in UI
Invalid input type	Validate and prompt user

End of Document