ARCHITECTURE DOCUMENT FOR THYROID DETECTION PROJECT:

1.Overview

This document provides a detailed architecture overview for the Thyroid Detection Project, which involves building a machine learning application to predict thyroid function using a logistic regression model. The system is built using a Flask-based web service, MongoDB for data storage, and is deployed on AWS using Docker containers.

Expanded Architecture Diagram Description

1. **User Interaction (UI)**
   - The user interacts with the webpage to upload a data file for prediction.
2. **Flask API (Running in AWS App Runner)**
   - The Flask API receives the uploaded data and retrieves the trained model from an S3 bucket.
   - The API uses this model to predict the uploaded data and returns the results to the user.
3. **Model Development Process**
   - **Data Ingestion**:
     - Data is ingested from MongoDB, where the thyroid data is stored. This data is fetched and passed to the preprocessing pipeline.
   - **Data Preprocessing**:
     - The ingested data is cleaned and prepared, dealing with missing values, outliers, and data types. This step ensures that the data is suitable for the next phases.
   - **Data Validation**:
     - The preprocessed data is validated against predefined schemas (such as checking data types, ranges, etc.). This ensures the integrity and consistency of the data.
   - **Data Transformation**:
     - The validated data is transformed into a format suitable for model training. Feature engineering, scaling, and encoding of categorical variables occur at this stage.
   - **Model Training**:
     - A logistic regression model is trained using the transformed data. Hyperparameter tuning and cross-validation ensure optimal performance.
   - **Model Serialization and Storage**:
     - The trained model is serialized (using joblib or pickle) and uploaded to an S3 bucket. This model is now ready to be used by the Flask API for predictions.

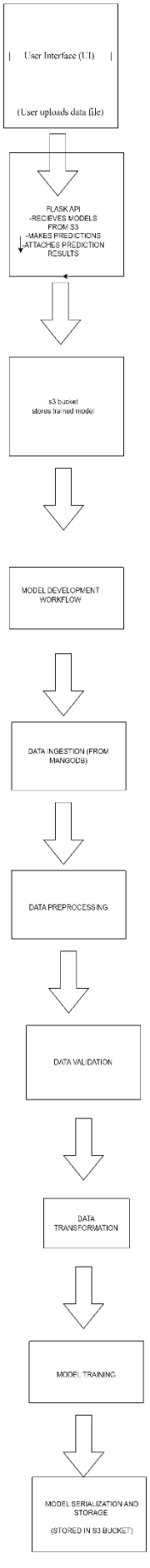
Detailed Diagram Layout

Frontend and Backend Interaction

1. **User Interface (UI)**
    - A webpage where users can upload data for prediction.
2. **Flask API (AWS App Runner)**
    - Handles the data from the UI.
    - Retrieves the trained model from S3.
    - Processes the data, makes predictions, and returns results.
3. **S3 Bucket**
    - Stores the trained model.

Model Development Workflow

4. **Data Ingestion**
    - Fetches data from MongoDB.
5. **Data Preprocessing**
    - Cleans and prepares the data.
6. **Data Validation**
    - Validates data against schemas.
7. **Data Transformation**
    - Transforms data for model training.
8. **Model Training**
    - Trains and tunes the model.
9. **Model Serialization**
    - Serializes and stores the model in s3


User Interface (UI)
(User uploads data file)
FLASK API
-RECIEVES MODELS FROM S3
-MAKES PREDICTIONS
-ATTACHES PREDICTION RESULTS
s3 bucket
stores trained model
MODEL DEVELOPMENT WORKFLOW
DATA INGESTION (FROM MANGODB)
DATA PREPROCESSING
DATA VALIDATION
DATA TRANSFORMATION
MODEL TRAINING
MODEL SERIALIZATION AND STORAGE
(STORED IN S3 BUCKET)