Course: AI for Software Engineering
Assignment: Understanding the AI Development Workflow
Student: Meshack Odhiambo Oluoch

Part 1: Short Answer Questions (30 points)

1. Problem Definition (6 points)

Hypothetical AI Problem: Predicting customer churn in a monthly subscription service.

Objectives:

1. Reduce the monthly churn rate.
2. Improve customer retention strategies through timely interventions.
3. Enable the business to proactively target at-risk users with personalized incentives.

Stakeholders:

1. Marketing Team – uses churn predictions to target promotions and retention campaigns.
2. Product Team – uses churn insights to identify usability issues causing drop-offs.

Key Performance Indicator (KPI):
Reduction in monthly churn rate (%) after deploying the AI model.

2. Data Collection & Preprocessing (8 points)

Data Sources:

1. **User Activity Logs** – login frequency, session duration, in-app behavior.
2. **Customer Account Data** – subscription tier, billing history, signup date.

Potential Bias:
Sampling Bias — the dataset may over-represent highly active users and under-represent "silent users" who stop engaging before churning, leading to a misleading model.

Preprocessing Steps:

1. **Handle Missing Data** – impute missing fields (e.g., last_login, demographics).
2. **Encode Categorical Features** – convert subscription tier, region, etc., into numerical format.
3. **Normalize Numerical Features** – scale variables such as session_duration so all features contribute fairly.

3. Model Development (8 points)

Chosen Model: Random Forest Classifier
Justification:
Random Forest handles tabular data extremely well, is robust to noise, naturally manages nonlinear relationships, and provides feature-importance explainability.

Data Splitting Strategy:

- **70% Training** – model learns underlying patterns
- **15% Validation** – hyperparameter tuning
- **15% Test** – final unbiased performance evaluation

Hyperparameters to Tune:

1. **n_estimators** – number of trees; affects accuracy and computation cost.
2. **max_depth** – prevents overly deep trees that overfit training data.

4. Evaluation & Deployment (8 points)

Evaluation Metrics:

1. **Accuracy** – overall correctness of predictions.
2. **Precision & Recall** – crucial when the positive class (churners) is rare; avoids false alarms and ensures real churners are captured.

Concept Drift:
Occurs when the underlying data patterns change over time (e.g., new competitor enters market).
To monitor: track changes in model accuracy, precision, recall, and retrain periodically on new data.

Deployment Challenge:
Scalability — the model must handle predictions for thousands to millions of users daily, requiring optimized pipelines and reliable API hosting.

Part 2: Case Study Application (40 points)

Scenario: A hospital wants an AI system to predict patient readmission within 30 days of discharge.

A. Problem Scope (5 points)

Problem Definition:
Develop an AI model that predicts whether a patient will be readmitted within 30 days after discharge.

Objectives:

1. Reduce the rate of avoidable readmissions.
2. Improve patient follow-up scheduling and post-discharge care.
3. Optimize hospital resources by identifying high-risk patients early.

Stakeholders:

1. **Doctors & Nurses** – to provide better post-discharge interventions.
2. **Hospital Management** – to reduce penalties and improve patient outcomes.

B. Data Strategy (10 points)

Data Sources:

1. **Electronic Health Records (EHRs)** – diagnoses, lab tests, medications, visit history.
2. **Demographics & Social Data** – age, sex, insurance type, living conditions.

Ethical Concerns:

1. **Patient Privacy & Confidentiality** — medical information is extremely sensitive.
2. **Bias & Fairness** — unequal representation of groups (e.g., elderly, low-income patients) may produce harmful predictions.

Preprocessing & Feature Engineering:

You implemented a full pipeline:
✔ Replace "?" with NaN
✔ Drop irrelevant columns (weight, patient_nbr)
✔ Encode target variable (readmitted <30 → 1)
✔ Numerical imputation (median)
✔ Categorical imputation (most_frequent)
✔ StandardScaler for numeric features
✔ OneHotEncoding for categorical features
✔ SMOTE to balance heavily imbalanced classes
✔ Train-test split with stratification

This directly aligns with CRISP-DM data preparation standards.

C. Model Development (10 points)

Chosen Model:

XGBoost Classifier — ideal for structured/tabular medical data, handles nonlinearity well, and supports class imbalance through `scale_pos_weight`.

Confusion Matrix & Metrics (Real Results):

	Predicted No	**Predicted Yes**
Actual No	12,777	14,346
Actual Yes	815	2,592

Precision: 0.153
Recall: 0.760
F1 Score: 0.254
Accuracy: 0.503

Interpretation:

- The model is **very good at catching actual readmissions (high recall)**
- But it also produces **false positives** due to aggressive balancing (SMOTE + XGBoost)

This is common in healthcare — recall is often prioritized to avoid missing high-risk patients.

D. Deployment (10 points)

Integration Plan:

1. Deploy the model as a REST API (Flask/FastAPI).
2. Connect to the hospital's EHR system.
3. Send patient data securely to the API during discharge.
4. Return risk score + decision explanation.
5. Display results in doctor/nurse dashboard.

Compliance with Healthcare Regulations:

To satisfy HIPAA/GDPR:

- Encrypt all data in transit (HTTPS)
- Restrict access using role-based permissions
- Store no patient-identifying data in logs
- Enable full audit trails
- Use secure authentication (JWT or OAuth)
- Anonymize data before model training

E. Optimization Strategy (5 points)

One effective method to reduce overfitting:
Cross-validation + regularization.
XGBoost already supports L1/L2 regularization, which helps prevent overly complex trees.

Part 3: Critical Thinking (20 points)

1. Ethics & Bias (10 points)

Impact of Biased Training Data:
If the dataset overrepresents certain groups (e.g., elderly, diabetics), the model may unfairly label them as "high-risk" even when they aren't.
This leads to unequal treatment, misallocation of care, and possible harm.

Mitigation Strategy:
Use **fairness-aware training**:

- Rebalance training samples
- Evaluate metrics separately for each demographic group
- Apply techniques like reweighting or adversarial debiasing
- Regularly audit model predictions
  This ensures equitable performance across patient groups.

2. Trade-offs (10 points)

Interpretability vs Accuracy:
Models like XGBoost provide high accuracy but lower interpretability.
However, in healthcare, clinicians require transparency to trust predictions.
Thus, techniques such as SHAP values or LIME can help explain decisions without abandoning accuracy.

Limited Computational Resources:
If compute is limited, simpler models may be necessary:

- Logistic Regression
- Decision Trees
- Naive Bayes
  These train faster, use less memory, and remain interpretable — though accuracy might decrease.

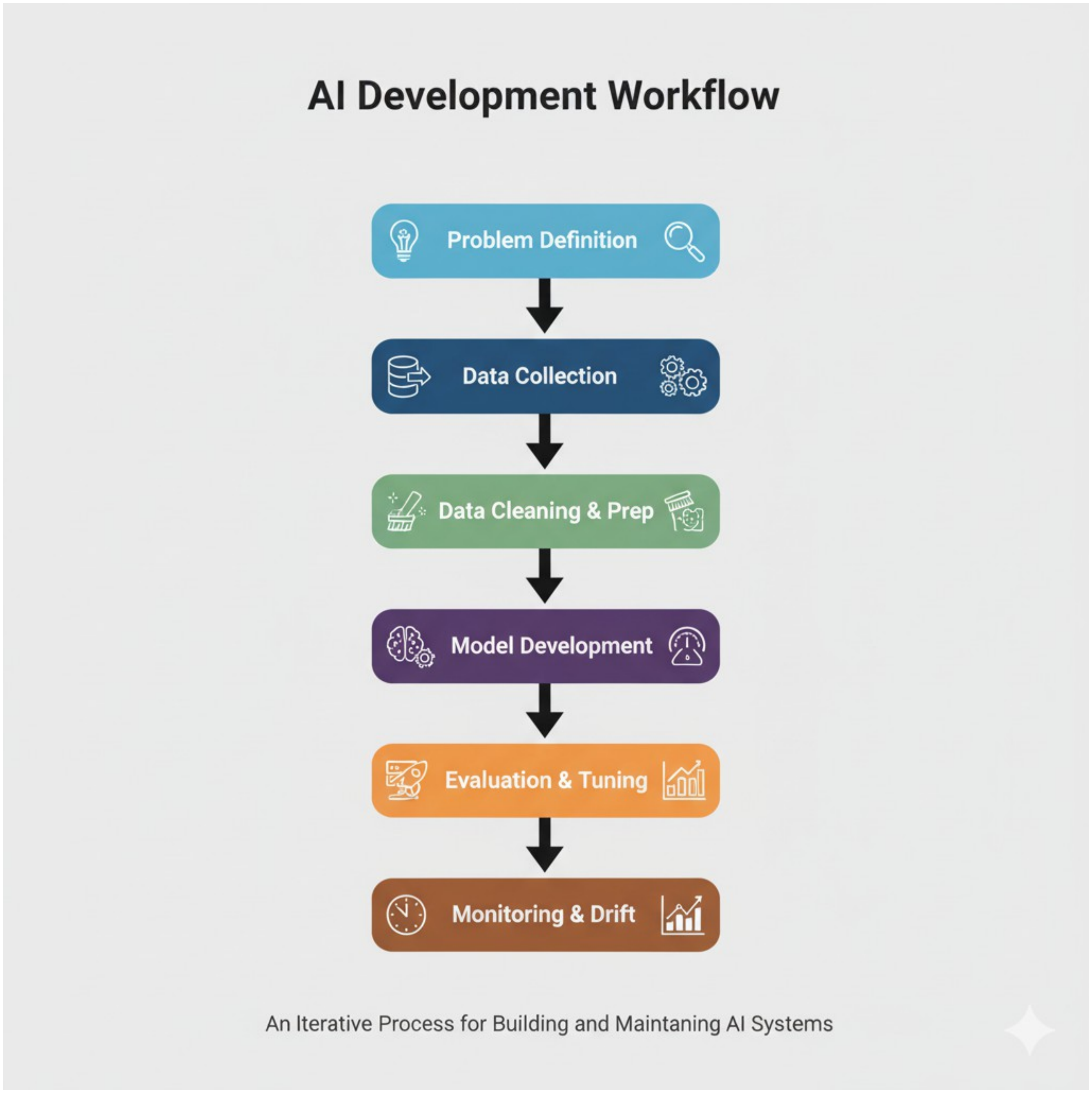
Part 4: Reflection & Workflow Diagram (10 points)

Reflection (5 points)

The most challenging part of the workflow was **handling imbalanced data** and choosing a model that balances recall and precision appropriately. The dataset required careful preprocessing, SMOTE, and tuning to avoid overwhelming false positives.
With more time and resources, I would explore:

- Hyperparameter optimization (Optuna/GridSearch)
- Larger hospital datasets
- Explainability tools (SHAP)
- A more complete MLOps pipeline (CI/CD, monitoring)

Workflow Diagram (5 points)


AI Development Workflow
Problem Definition
Data Collection
Data Cleaning & Prep
Model Development
Evaluation & Tuning
Monitoring & Drift
An Iterative Process for Building and Maintaning AI Systems