Part 1: Problem Definition (6 points)

AI Problem:
Predict which employees are likely to leave the company in the next six months.

Objectives:

1. Identify employees who might leave early so HR can take action.
2. Help managers plan retention programs to reduce turnover.
3. Lower the costs of hiring and training new employees.

Stakeholders:

1. Human Resources (HR) department – they use the model to act on high-risk employees.
2. Company management – they use the results to plan staffing and budgeting.

Key Performance Indicator (KPI):
F1-score of the model when predicting employee turnover.
The F1-score balances precision and recall, showing how accurate the model is overall.

Part 2: Data Collection & Preprocessing (8 points)

Data Sources:

1. Company HR records – include data such as age, job role, salary, years at the company, and performance ratings.
2. Employee satisfaction or engagement surveys – include feedback about workload, satisfaction, and relationship with managers.

Potential Bias:
The data could be biased if surveys are only answered by some employees.
For example, office workers may respond more often than field workers, which could make the model less accurate for certain groups.

Preprocessing Steps:

1. **Handle missing data:** fill in missing survey responses or remove incomplete rows.
2. **Encode categorical variables:** convert text data (like job title or department) into numeric form.
3. **Normalize numerical data:** scale values like salary and performance score so they are on a similar range for the model.

Part 3: Model Development (8 points)

Chosen Model:
Random Forest Classifier.

Justification:
A Random Forest works well for classification tasks like predicting whether an employee will stay or leave.
It can handle both numerical and categorical data, reduce overfitting, and show which features (like salary or satisfaction) are most important.

Data Split:

- 70% Training data – used to train the model.
- 15% Validation data – used to adjust model parameters.
- 15% Test data – used to check final performance before deployment.

Hyperparameters to Tune:

1. **Number of trees:** controls how many decision trees the forest has. More trees can improve accuracy but slow the model.
2. **Maximum tree depth:** limits how deep each tree can go. It helps prevent overfitting so the model performs well on new data.

Part 4: Evaluation & Deployment (8 points)

Evaluation Metrics:

1. **Precision:** measures how many employees predicted to leave actually do. This avoids unnecessary HR actions on employees who are not at risk.
2. **Recall:** measures how many employees who left were correctly identified. This ensures at-risk employees are not missed.

Concept Drift:
Concept drift happens when the data pattern changes over time.
For example, new management policies, salary changes, or remote work options could change why employees leave.
To handle this, the model should be monitored regularly. If accuracy drops, retrain it using the most recent employee data.

Technical Challenge During Deployment:
Scalability: as the company grows, the model must process more data quickly. Deploying it on cloud infrastructure or using batch predictions can help handle large datasets and real-time updates efficiently.

Part 2: Case Study Application — Predicting Patient Readmission Risk

A. Problem Scope (5 points)

Hospitals face financial and operational challenges from patients being readmitted within 30 days of discharge.
The goal is to build an AI system that predicts the likelihood of a patient being readmitted, allowing healthcare
staff to prioritize post-discharge care and reduce unnecessary readmissions.

- Objectives:

1. 1. Predict 30-day readmission risk using historical and current patient data.
2. 2. Identify key clinical and demographic factors influencing readmissions.
3. 3. Support physicians in early interventions to improve patient outcomes.

- Stakeholders:
- • Primary: Hospital management and healthcare providers.
- • Secondary: Patients and insurance agencies.

Success KPI: Readmission Reduction Rate — percentage decrease in 30-day readmission rates after implementing the AI system.

B. Data Strategy (10 points)

Proposed Data Sources

- Electronic Health Records (EHRs): Diagnostic codes, lab results, vitals, medications, discharge summaries — core dataset for medical patterns.
- Demographic Data: Age, gender, socioeconomic indicators — help identify population-level influences.
- Pharmacy Data: Prescription adherence, drug interactions — captures medication compliance.
- Appointment & Administrative Records: Follow-ups, missed appointments — reflects care continuity.
- Wearable / IoT Data: Heart rate, sleep, activity — captures post-discharge recovery.
- Laboratory Information Systems (LIS): Test results like HbA1c, cholesterol, CRP — clinical health indicators.
- Social & Behavioral Data: Lifestyle, diet, smoking, social support — behavioral predictors.
- Public Health & Environmental Data: Air quality, socioeconomic indices — external health determinants.
- Patient Feedback & Surveys: Recovery experiences, satisfaction — qualitative risk indicators.

Ethical Concerns

- Patient Privacy and Confidentiality: Sensitive health data must be encrypted, anonymized, and access-restricted.
- Informed Consent: Patients must understand how their data is used; ensure transparent consent.
- Algorithmic Bias and Fairness: Audit models for demographic bias; ensure equitable outcomes.
- Explainability and Transparency: Use SHAP/LIME to make AI predictions interpretable to clinicians.
- Accountability: Define clear governance and responsibility for AI-related errors.
- Data Ownership and Sharing: Clarify who owns patient data and prevent unauthorized reuse.
- Security Vulnerabilities: Prevent breaches via encryption and regular security audits.
- Automation Bias: Ensure AI supports, not replaces, human judgment.
- Equity of Access: Develop affordable and scalable models for all hospital types.
- Continuous Learning and Consent Renewal: Renew patient consent when models are updated or retrained.

Preprocessing Pipeline & Feature Engineering

- Data Cleaning: Handle missing values, duplicates, and standardize medical codes.
- Outlier Detection: Filter extreme values using z-score or IQR.
- Feature Encoding: One-hot encode categorical data like diagnoses or insurance type.
- Scaling: Normalize continuous variables such as heart rate and glucose.
- Imbalanced Data Handling: Apply SMOTE or class weighting.
- Feature Engineering: Derive features like average hospital stay or comorbidity count.
- Dimensionality Reduction: Apply PCA or feature selection for performance improvement.

C. Model Development (10 points)

Chosen Model: Gradient Boosting Classifier (e.g., XGBoost or LightGBM)
Justification: Handles mixed data types, strong performance on tabular data, interpretability via feature importance and SHAP values.

Data Split: 70% training, 15% validation, 15% test sets.

- Key Hyperparameters to Tune:
- • Learning Rate (η): Controls tree contribution to prevent overfitting.
- • Max Depth: Adjusts model complexity for better generalization.

Confusion Model

```
Training Readmission Prediction Model...
```

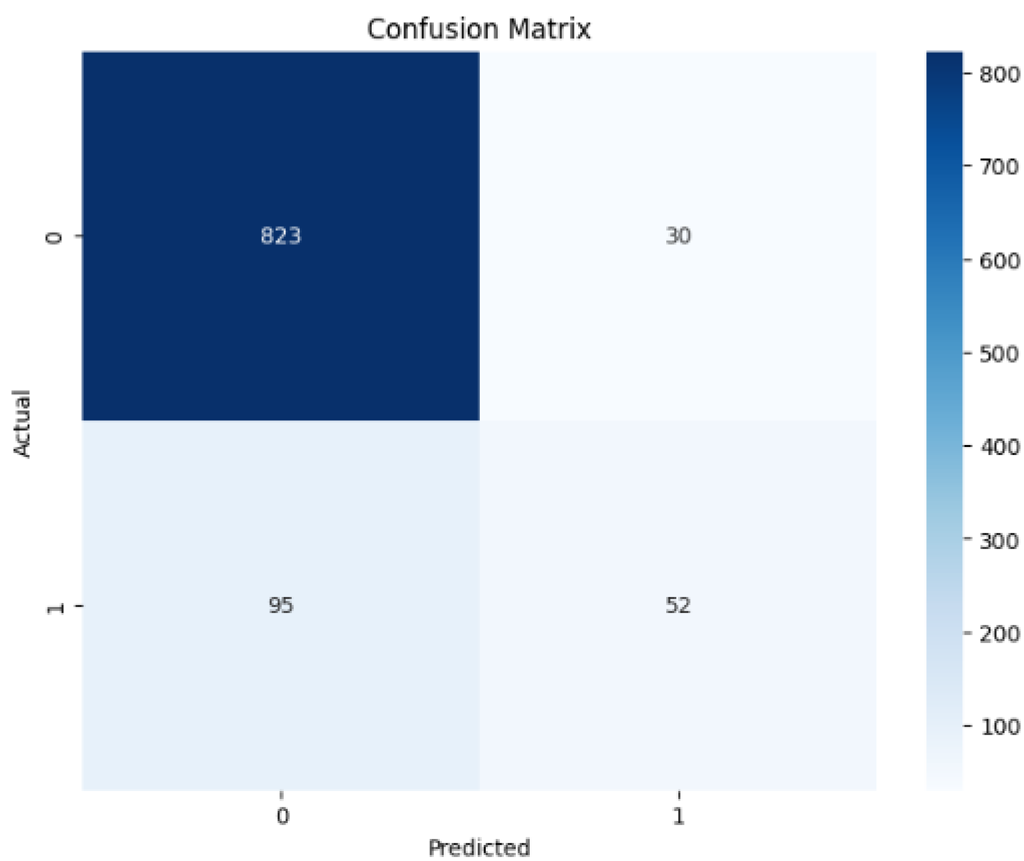
```
Training set: 4000 records
Test set: 1000 records
✅ Model training completed!
📈 Accuracy: 0.875

📋 Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.96      0.93       853
           1       0.63      0.35      0.45       147

    accuracy                           0.88      1000
   macro avg       0.77      0.66      0.69      1000
weighted avg       0.86      0.88      0.86      1000


💾 Model saved to readmission_model.pkl
```



D. Deployment (10 points)

Integration Steps:

1. Package model using MLflow or TensorFlow Serving.

2. Expose REST API for EHR/HMS integration.
3. Embed predictions in clinician dashboard showing risk score and contributing factors.
4. Automate data ingestion and retraining.
5. Monitor model performance and detect concept drift.

- Compliance with Regulations (HIPAA/GDPR):
- • Encrypt data at rest and in transit.
- • Implement role-based access control and audit trails.
- • Use anonymized data for model development.
- • Ensure explainability for clinical transparency.

E. Optimization (5 points)

To address overfitting:
- Apply L1/L2 regularization.
- Use k-fold cross-validation.
- Implement early stopping.
- Prune trees to reduce model complexity.

Part 3: Critical Thinking (20 points)

Ethics & Bias (10 points)

- **Impact of Biased Data:**
  - The model could disproportionately misclassify readmissions for certain subgroups (e.g., older adults, certain racial/ethnic groups, or patients in lower socioeconomic quintiles), leading to unequal access to interventions.
  - Miscalibration could erode trust in the system and worsen disparities if high-risk patients are not identified or are overburdened with unnecessary follow-ups.

- **Mitigation Strategy: Fairness-aware machine learning** techniques
  - Measure subgroup performance (precision, recall, calibration by race, sex, age, socioeconomic status).
  - Use reweighting or constrained optimization to balance metrics across groups.
  - Deploy post-hoc calibration within subgroups or use equalized odds/adaptive thresholds to ensure similar error rates across groups.
  - Involve diverse clinical stakeholders in model reviews and maintain an ongoing bias audit.

Trade-offs (10 points)

- **Interpretability vs. Accuracy:**
  - **High Accuracy (e.g., Deep Learning) vs. High Interpretability (e.g., Logistic Regression):** In healthcare, the trade-off is critical. Clinicians need to understand *why* a patient is flagged as high-risk to trust the prediction and take appropriate action.
  - A black-box model, even if highly accurate, might be unusable in a clinical setting due to lack of trust and inability to provide actionable insights or a clear justification to the patient.
  - Interpretability helps in clinical validation, ethical accountability, and regulatory compliance.
  - Therefore, a balance is often sought, perhaps using models like GBMs that offer a good blend of both, or using **post-hoc interpretability methods** (like SHAP or LIME) on black-box models to explain individual predictions.
- **Limited Computational Resources Impact on Model Choice:**
  - Favor lighter models (logistic regression, shallow trees) that require less computing and memory.
  - Batch processing during off-peak hours or edge-hosted inference with optimized code. Implement model compression and quantization if feasible; consider cloud-based inference with strict data governance and latency requirements, ensuring regulatory compliance.
  - Balance: prioritize models that deliver acceptable performance with low latency and resource usage, possibly sacrificing a bit of accuracy for reliability and speed.
  - Efficiency in deployment (fast prediction times on standard CPUs) would become a primary constraint.

Part 4: Reflection & Workflow Diagram

A. Reflection (5 points)

The most challenging part of the AI Development Workflow was ensuring data quality and ethical compliance.
Integrating multiple hospital data sources—each with different formats and privacy

constraints—required careful planning,
data validation, and anonymization to protect patient confidentiality. Additionally, balancing model performance
with interpretability was difficult, as clinicians need transparent insights rather than opaque predictions.

With more time and resources, the approach could be improved by developing an automated data pipeline
for real-time ingestion and preprocessing. This would reduce manual work, maintain data freshness, and support
continuous model retraining. Additional efforts would also go toward stakeholder training to ensure responsible AI use
and enhance adoption across departments.

B. Workflow Diagram (5 points)

The following outlines the AI Development Workflow as applied to the hospital readmission prediction project.
It aligns with the CRISP-DM (Cross-Industry Standard Process for Data Mining) framework.

- 1. Problem Definition: Define the AI problem, objectives, stakeholders, and success criteria (KPI).
- 2. Data Collection: Aggregate data from EHRs, demographics, pharmacy, and behavioral sources.
- 3. Data Preprocessing: Clean, encode, normalize, and engineer features for modeling.
- 4. Model Development: Train and tune Gradient Boosting Classifier with cross-validation.
- 5. Evaluation: Assess performance using metrics such as AUC, recall, and precision.
- 6. Deployment: Integrate model into hospital systems through secure REST APIs.
- 7. Monitoring & Maintenance: Continuously monitor model drift, update data pipelines, and ensure compliance.

Diagram Description:
A flowchart can be visualized as follows — arrows indicate progression between stages:

Problem Definition → Data Collection → Data Preprocessing → Model Development → Evaluation → Deployment → Monitoring