Part 1: AI Development Workflow

3. Model Development

Model Choice:
We selected a Random Forest classifier for its robustness, ability to handle both numerical and categorical data, and resistance to over-fitting. It also provides feature importance scores, which help interpret model behavior—especially useful in domains like education or healthcare.

Data Splitting Strategy:

- **Training Set (70%)**: Used to train the model.
- **Validation Set (15%)**: Used for hyper-parameter tuning and model selection.
- **Test Set (15%)**: Used to evaluate final model performance on unseen data.

Hyperparameters to Tune:

- n_estimators: Number of trees in the forest. A higher number can improve accuracy but increases computation time.
- max_depth: Controls the depth of each tree. Helps prevent over-fitting by limiting complexity.

4. Evaluation & Deployment

Evaluation Metrics:

Precision: Measures the proportion of true positives among predicted positives. Important when false positives are costly (e.g., predicting a student will drop out when they won't).

- **Recall**: Measures the proportion of true positives identified among all actual positives. Crucial when missing a true case has serious consequences (e.g., missing a high-risk patient).

Concept Drift:
Concept drift occurs when the statistical properties of input data change over time, reducing model accuracy.
Monitoring Strategy:

- Implement periodic model evaluation using recent data.
- Use drift detection algorithms (e.g., DDM, ADWIN).
- Retrain the model regularly with updated datasets.

Technical Challenge – Scalability:
Deploying the model to serve thousands of predictions per minute may strain resources.
Solution:
Use model compression, batch inference, and deploy via scalable cloud services (e.g., AWS Lambda, Azure Functions).