# SOFTWARE REQUIREMENTS SPECIFICATION

Project Title: Transition Analysis of Course Interaction Data

Submitted by: [Your Name] ([Your Roll No])

Submitted to: [Supervisor Name]

Department: Information Technology

Institution: The Islamia University of Bahawalpur

## 1. INTRODUCTION

### 1.1 Purpose

The purpose of this document is to define the functional and non-functional requirements for the "Transition Analysis System." This desktop-based application aims to analyze student interaction logs from the Deeds Simulator to identify learning behaviors. By applying Process Mining techniques, the system will visualize how students transition between different learning activities (e.g., studying, simulating, testing) and correlate these patterns with their academic performance.

### 1.2 Scope

The system will process historical educational datasets (CSV logs and Excel grade sheets).

* **Input:** Raw event logs containing timestamps, activity names, and student IDs; Grade sheets containing exam scores.
* **Processing:** Data cleaning, session aggregation, and application of the Heuristics Miner algorithm.
* **Output:** Interactive Directed Graphs (Transition Maps), statistical charts comparing high vs. low performers, and efficiency reports.
* **Constraint:** The system focuses on post-analysis of historical data, not real-time monitoring of live classes.

### 1.3 Definitions, Acronyms, and Abbreviations

* **SRS:** Software Requirements Specification.
* **EDM:** Educational Data Mining.
* **Transition:** The movement of a student from one specific activity (Activity A) to the next (Activity B).
* **Event Log:** A collection of records where each record represents an event (Case ID, Activity Name, Timestamp).
* **DFG:** Directly-Follows Graph (A visual map of transitions).

## 2. OVERALL DESCRIPTION

### 2.1 Product Perspective

This is a standalone data analysis dashboard developed using Python. It bridges the gap between raw log data and instructor insights. It utilizes the **PM4Py (Process Mining for Python)** framework for the analysis engine and **Streamlit** for the graphical user interface.

### 2.2 User Characteristics

* **Administrator/Instructor:** The primary user. They possess domain knowledge of the course material but require a simplified visual interface to interpret complex data logs. They are responsible for uploading datasets and interpreting the graphs.

### 2.3 System Environment

* **OS:** Windows 10/11, Linux, or macOS.
* **Runtime:** Python 3.9+.
* **Browser:** Chrome/Edge/Firefox (for rendering the Streamlit interface).

## 3. SYSTEM FEATURES AND FUNCTIONAL REQUIREMENTS

### 3.1 Module 1: Data Ingestion & Preprocessing

**Description:** The system must handle raw CSV files generated by the Deeds Simulator and standard Excel grade sheets.

* **FR-01:** The system shall allow the user to upload a batch of CSV files (Session Logs) and an XLSX file (Grades).
* **FR-02:** The system shall automatically clean the data by handling missing values and converting timestamps into a standard ISO-8601 format.
* **FR-03:** The system shall merge separate session files into a single "Event Log" dataframe suitable for process mining.
* **FR-04:** The system shall anonymize student data by using distinct Student\_IDs rather than real names.

### 3.2 Module 2: Transition Analysis (Process Mining)

**Description:** The core logic that converts list data into visual maps.

* **FR-05:** The system shall utilize the **Heuristics Miner** algorithm to discover the control flow of student activities.
* **FR-06:** The system shall calculate the frequency of specific transitions (e.g., How often did students go from *TextEditor* directly to *Simulation*?).
* **FR-07:** The system shall identify "Self-Loops" (students repeating the same activity continuously without progress).

### 3.3 Module 3: Visualization Dashboard

**Description:** The user interface for interacting with the analysis.

* **FR-08:** The system shall display a **Directly-Follows Graph (DFG)** where:
  + **Nodes** represent activities (Study, Simulate, Idle).
  + **Edges (Arrows)** represent the transition path.
  + **Edge Thickness** represents the frequency of that path.
* **FR-09:** The system shall provide interactive filters to slice data by "Session ID" or "Student Group" (High Achievers vs. Low Achievers).
* **FR-10:** The system shall display statistical bar charts showing the average time spent on specific activities.

### 3.4 Module 4: Performance Correlation

**Description:** Linking behavior to grades.

* **FR-11:** The system shall categorize students into groups (Pass/Fail or High/Low) based on the uploaded Excel grade sheet.
* **FR-12:** The system shall generate a comparative view showing two transition maps side-by-side to highlight behavioral differences between groups.

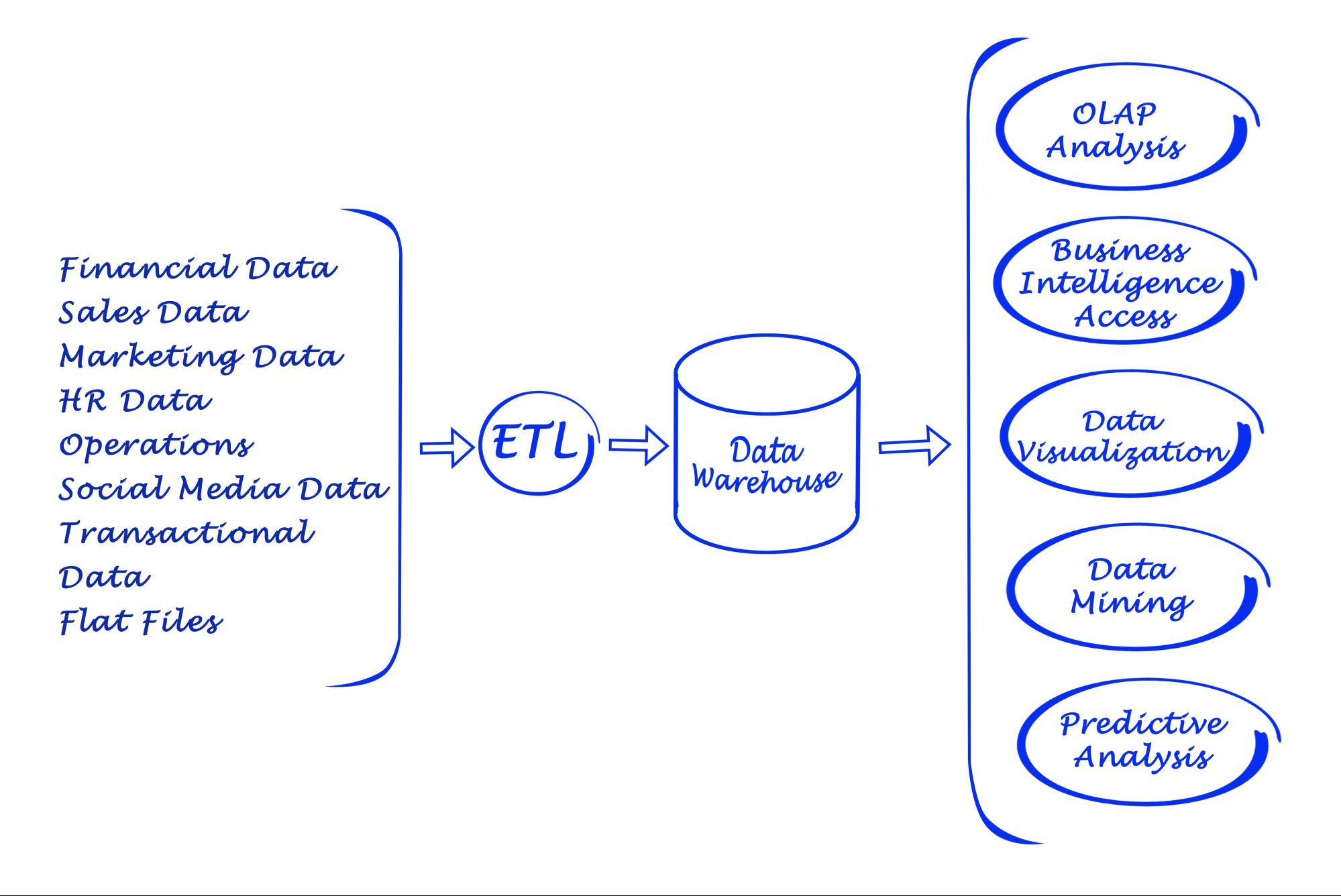
## 4. SYSTEM MODELS AND DIAGRAMS

### 4.1 Use Case Diagram

The following diagram illustrates the primary interactions between the Instructor and the System.

### 4.2 Data Flow Diagram (DFD) - Level 1

This diagram details how data moves from raw CSVs through the cleaning pipeline into the visualization engine.



vaeenma/Indiapicture

### 4.3 Entity Relationship Diagram (ERD)

The logical structure of the processed data stored within the application memory.

## 5. NON-FUNCTIONAL REQUIREMENTS

### 5.1 Usability

* The interface shall be web-based (running locally) with a clean, dashboard-style layout.
* No coding knowledge should be required to operate the dashboard.

### 5.2 Performance

* The system shall parse and process a dataset of up to 100,000 log rows in under 30 seconds.
* Graph visualizations shall render in under 5 seconds.

### 5.3 Reliability

* The system shall implement exception handling to prevent crashes if a malformed CSV file is uploaded (e.g., incorrect date formats).

### 5.4 Maintainability

* The codebase shall be modular, separating the User Interface (app.py) from the Logic (processing.py), allowing for easy updates.

## 6. TECHNOLOGY STACK

To ensure industry relevance and ease of implementation, the following widely-used technologies are selected:

| **Component** | **Technology** | **Reasoning** |
| --- | --- | --- |
| **Language** | Python 3.x | Industry standard for Data Science and Mining. |
| **Frontend/UI** | Streamlit | Allows rapid creation of data dashboards in pure Python. |
| **Data Manipulation** | Pandas | Efficient handling of large CSV/Excel datasets. |
| **Process Mining** | PM4Py | The leading open-source library for Process Mining algorithms. |
| **Visualization** | Graphviz / Plotly | For rendering directed graphs and interactive charts. |
| **IDE** | VS Code / PyCharm | Standard development environments. |

## 7. PROJECT TIMELINE (GANTT CHART SUMMARY)

1. **Requirement Gathering:** Week 1-2
2. **Data Preprocessing Module:** Week 3-4
3. **Process Mining Implementation (PM4Py):** Week 5-6
4. **UI Development (Streamlit):** Week 7-8
5. **Testing & Validation:** Week 9
6. **Final Documentation:** Week 10

## 8. CONCLUSION

The "Transition Analysis of Course Interaction Data" project will provide a vital tool for educators to understand the "process" of learning, not just the outcome. By leveraging modern Python-based Process Mining technologies, this system will offer actionable insights into student behavior patterns without requiring complex infrastructure.