STUDENTS DROPOUT ANALYSIS

**CHAPTER-1**

**1.PROJECT DESCRIPTION**

**1.1 Description**

Right to education is key concern for government and at school level. Covid outbreak has affected the education sector badly, due to which there is an increased student dropout especially from Government schools and the traditional methods (surveys) couldn’t predict the accurate dropout rate.

To overcome this problem, we are developing a software where the admin can analyse the dropout rate by school wise, grade wise and gender wise. This helps the government to take necessary actions and to increase the literacy rate.

The factors contributing to student attrition, dropout analysis facilitates early intervention strategies. Identifying at-risk students in their formative stages allows educational institutions to implement targeted support systems, addressing challenges such as financial constraints, health issues, or lack of interest before they escalate.

**CHAPTER 2**

**2.PROBLEM STATEMENT**

**2.1 Problem Statement**

Right to education is key concern for government and at school level. Covid outbreak has affected the education sector badly, due to which there is an increased student dropout especially from Government schools and the traditional methods (surveys) couldn’t predict the accurate dropout rate.

To overcome this problem, we are developing a software where the admin can analyse the dropout rate by school wise, grade wise and gender wise. This helps the government to take necessary actions and to increase the literacy rate.

The factors contributing to student attrition, dropout analysis facilitates early intervention strategies. Identifying at-risk students in their formative stages allows educational institutions to implement targeted support systems, addressing challenges such as financial constraints, health issues, or lack of interest before they escalate.

**TECHNICAL FEASIBILITY**:

* Data sources: Collecting data on student’s demographics, attendance, and behaviour.
* Data storage: Utilize databases for efficient data retrieval.
* Data Analysis: Using Python framework to find out the dropout analysis.
* Visualization: Visualizing the findings through graphs.

**ECONOMICAL FEASIBLITY**:

* Technology infrastructure: Expenses for hardware (computer) need for the project development.
* Infrastructure Maintenance: On-going cost for maintaining the software and hardware.
* Risk assessment: Changes in the enrolment patterns or the effectiveness of interventions.

**3.REQUIRED SPECIFICATION**

**3.1 Hardware Specifications**

Basic Requirements:

Processor (CPU): A modern multi-core processor (i5 or higher for mid-range analysis, or i7/i9 for more extensive computational tasks) to handle data processing efficiently.

Memory (RAM): At least 8GB RAM is recommended for basic analysis. For larger datasets or complex computations, 16GB or more would be beneficial.

Storage (Hard Drive/SSD): A decent-sized hard drive (500GB or higher) or SSD for storing datasets, software, and analysis results. SSDs provide faster data access, which can be beneficial for data-intensive tasks.

Recommended Additions:

Graphics Card (GPU): Depending on the nature of the analysis (machine learning, deep learning, or visualization tasks), a dedicated GPU (NVIDIA GeForce GTX or RTX series, or AMD Radeon RX series) can significantly accelerate computations.

Multiple Monitors: Having multiple monitors can enhance productivity, especially when working with large datasets or performing complex analyses.

Server-Side Considerations (for larger-scale analyses):

Server Hardware: If the analysis involves handling extensive datasets or requires high computational power, setting up a server with high-performance CPUs (such as Intel Xeon or AMD EPYC processors), large RAM (32GB or more), and storage capacity (RAID-configured drives or SSDs) might be necessary.

Cloud Services: For scalability and flexibility, utilizing cloud services (such as AWS, Google Cloud, or Azure) can be beneficial. These platforms provide scalable computing resources and storage for large-scale analysis without the need for upfront hardware investment**.**

**3.2 Software Specifications**

DESCRIPTION AND PRIORITY:

The student’s dropout analysis software maintains information on schools in a particular region, students details, the dropout count per school per year and the reason for the dropout.

CLIENT/SERVER SYSTEM

The term client/server refers primarily to an architecture or logical division of responsibilities, the client is the application (also known as the front-end), and the server is the DBMS (also known as the back end).

A client/server system is a distributed system in which,

Some sites are client sites and others are server sites.

All the data resides at the server sites.

All applications execute at the client sites.

USER INTERFACE DESIGN

Student Information Form:

- Collects crucial details about the student, including name, financial status, health status, and interest status.

- Facilitates comprehensive data collection for a holistic analysis.

Financial Status Dropdown:

- Allows users to select the financial status of the student.

- Options include Poor, Average, and Good, providing a clear categorization.

Health Status Dropdown:

- Enables the selection of the student's health status.

- Choices encompass Poor, Average, and Good, aiding in understanding health-related aspects.

Interest Status Dropdown:

- Provides options for selecting the interest status of the student.

- Choices range from Low to High, helping to gauge the student's level of interest in education.

Submit Button:

- Triggers the analysis process upon clicking.

- Submits entered data to the server for further processing.

Result Display Section:

- A designated area to showcase the analysis results.

- May include predictions, recommendations, or insights based on the entered data.

- Enhances user understanding by presenting actionable information.

Styling and Design:

- Clean and Simple Design: The user interface adopts a minimalist design for straightforward interaction.

Responsive Layout:

-Ensures a consistent experience across various devices, accommodating both desktop and mobile users.

- Adequate Spacing and Labels: Clear labels and sufficient spacing contribute to the overall clarity of the interface, making it user-friendly.

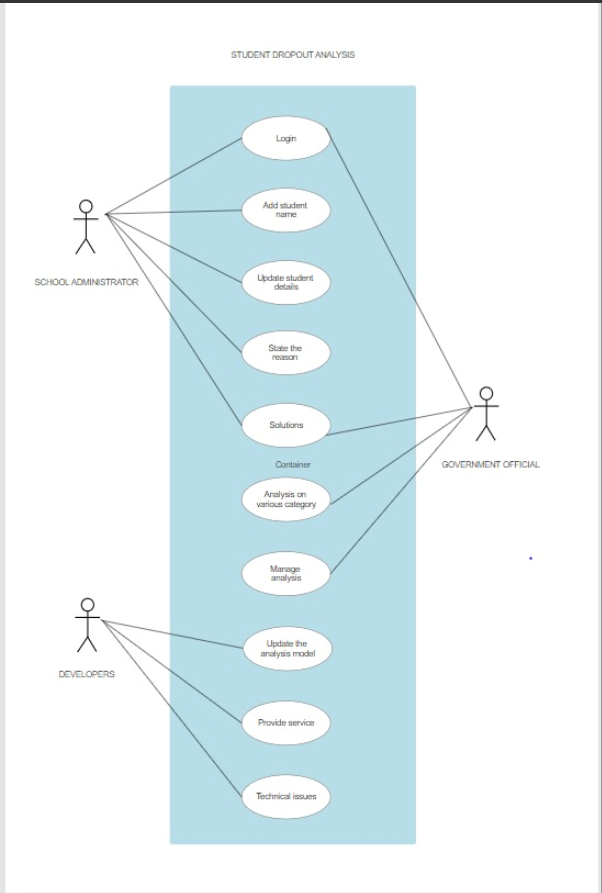
Project Implementation:

- HTML Structure: Developed a structured HTML document with appropriate form elements to capture student information.

- CSS Styling: Implemented a CSS stylesheet to achieve a clean design, responsive layout, and clear spacing.

**4.DESIGN**

**4.1 USE CASE DIAGRAM :**



**Actors:**

1. Administrator: The person in charge of the system, overseeing everything.

2. Student: The user providing personal and academic information to the system.

3. Developer: Analyses collected data and draws insights.

**Use Cases (What the System Can Do):**

1. Submit Student Information: Students can give their details to the system.

2. View Dropout Statistics: Admins and Data Analysts can check stats and visualizations.

3. Generate Reports: Data Analysts can make reports for analysis.

4. Update Student Information: Admins can modify student details.

5. Receive Dropout Alerts: Admins get notifications about potential dropouts.

6. Visualize Dropout Trends: Data Analysts can see trends using charts and graphs.

7. Monitor System Health: Admins keep an eye on the system's overall health.

**4.2 CLASS DIAGRAM**

A diagram of a student model

Description automatically generated

A class diagram for a school dropout analysis system is like a visual plan showing different parts of the system and how they relate. Imagine you have different "parts" like students, dropouts, schools, data analysis, and administrators.

Each part (class) has specific things it knows about itself (attributes) and actions it can do (methods). For example, a student knows their ID, name, age, and grade, and can enroll in a school or study. This diagram helps us see how these parts are connected.

This diagram serves as a blueprint for the system's architecture, emphasizing the static elements and their relationships, laying the groundwork for further development and implementation of the student dropout analysis system.

**4.3 INTERACTION DIAGRAM**

A diagram of a data flow

Description automatically generated

* An interaction UML diagram for student dropout analysis provides a dynamic view of the system by illustrating the communication and collaboration among various components during specific scenarios.
* The diagram shows the sequence of messages exchanged between these actors and the system components, emphasizing the flow of information and control during key processes such as submitting student information, generating reports, and visualizing dropout trends.
* In essence, the interaction diagram captures the dynamic behavior and communication patterns within the student dropout analysis system, aiding in a comprehensive understanding of how different elements collaborate to achieve specific tasks.

**4.4 STATE CHART DIAGRAM**

A diagram of a student information

Description automatically generated

* A State Chart UML diagram for student dropout analysis captures the various states and transitions that a student's data may undergo throughout the analysis process.
* Transitions between these states depict the flow of activities: from the initial idle state, the system transitions to the data input state when a user submits information.
* Following this, the system moves to the analysis processing state, signifying the backend processing of the entered data.
* Finally, the results display state shows the presentation of analysis outcomes.
* This visual representation helps to understand the dynamic behavior and sequence of actions within the student dropout analysis system, offering insights into its operational flow**.**

**4.5 PACKAGE DIAGRAM**

**A diagram of a software structure

Description automatically generated**

* A UML package diagram for student dropout analysis provides a structured overview of the system's components and their interrelationships.
* Dependencies illustrate how these components interact: the user interface depends on the analysis engine for data processing, which, in turn, relies on data access components to retrieve student information.
* Additionally, reporting components depend on the analysis engine to generate meaningful reports.
* The diagram offers a visual representation of the system's architecture, aiding in understanding the organization and collaboration between key modules for effective student dropout analysis.

**5.IMPLEMENTATION AND DATABASE CONNECTIVITY**

**5.1 Source code**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

public class DropoutAnalysis {

private static final String JDBC\_URL = "jdbc:mysql://localhost:3306/studentsdb";

private static final String JDBC\_USER = "your\_username";

private static final String JDBC\_PASSWORD = "your\_password";

public static void main(String[] args) {

Connection connection = null;

try {

// Establish database connection

connection = DriverManager.getConnection(JDBC\_URL, JDBC\_USER, JDBC\_PASSWORD);

// Perform operations

insertStudent(connection, "John Doe", "Good", "Average", "High", "Financial difficulties");

insertStudent(connection, "Jane Smith", "Average", "Good", "Medium", "Health issues");

// Retrieve data for graph visualization

visualizeDropoutReasons(connection);

} catch (SQLException e) {

e.printStackTrace();

} finally {

// Close the connection in the finally block to ensure it's always closed

if (connection != null) {

try {

connection.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

private static void insertStudent(Connection connection, String name, String financialStatus,

String healthStatus, String interestStatus, String dropoutReason) throws SQLException {

// Insert a new student record

String sql = "INSERT INTO students (name, financial\_status, health\_status, interest\_status, dropout\_reason) " +

"VALUES ()";

try (PreparedStatement statement = connection.prepareStatement(sql)) {

statement.setString(1, name);

statement.setString(2, financialStatus);

statement.setString(3, healthStatus);

statement.setString(4, interestStatus);

statement.setString(5, dropoutReason);

statement.executeUpdate();

}

}

private static void visualizeDropoutReasons(Connection connection) throws SQLException {

// Query to retrieve dropout reasons

String sql = "SELECT dropout\_reason, COUNT(\*) AS count FROM students " +

"WHERE dropout\_reason IS NOT NULL GROUP BY dropout\_reason";

try (PreparedStatement statement = connection.prepareStatement(sql);

ResultSet resultSet = statement.executeQuery()) {

while (resultSet.next()) {

String reason = resultSet.getString("dropout\_reason");

int count = resultSet.getInt("count");

// Perform graph visualization or store data for further processing

System.out.println("Reason: " + reason + ", Count: " + count);

}

}

}

}

import org.jfree.chart.ChartFactory;

import org.jfree.chart.ChartPanel;

import org.jfree.chart.JFreeChart;

import org.jfree.chart.axis.CategoryAxis;

import org.jfree.chart.axis.NumberAxis;

import org.jfree.chart.plot.CategoryPlot;

import org.jfree.chart.plot.PlotOrientation;

import org.jfree.data.category.CategoryDataset;

import org.jfree.data.category.DefaultCategoryDataset;

import javax.swing.\*;

import java.awt.\*;

public class DropoutAnalysisWithGraph extends JFrame {

public DropoutAnalysisWithGraph() {

super("Dropout Reasons Visualization");

CategoryDataset dataset = createDataset();

JFreeChart chart = createChart(dataset);

ChartPanel chartPanel = new ChartPanel(chart);

chartPanel.setPreferredSize(new Dimension(800, 600));

setContentPane(chartPanel);

}

private CategoryDataset createDataset() {

DefaultCategoryDataset dataset = new DefaultCategoryDataset();

// Dummy data for illustration

dataset.addValue(5, "Reasons", "Financial difficulties");

dataset.addValue(8, "Reasons", "Health issues");

dataset.addValue(3, "Reasons", "Other");

return dataset;

}

private JFreeChart createChart(CategoryDataset dataset) {

return ChartFactory.createBarChart(

"Dropout Reasons",

"Reasons",

"Count",

dataset,

PlotOrientation.VERTICAL,

true,

true,

false

);

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> {

DropoutAnalysisWithGraph example = new DropoutAnalysisWithGraph();

example.setSize(800, 600);

example.setLocationRelativeTo(null);

example.setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

example.setVisible(true);

});

}

}

**5.2 Screenshots**

**5.2.1 User Login Page**

**A screenshot of a login page

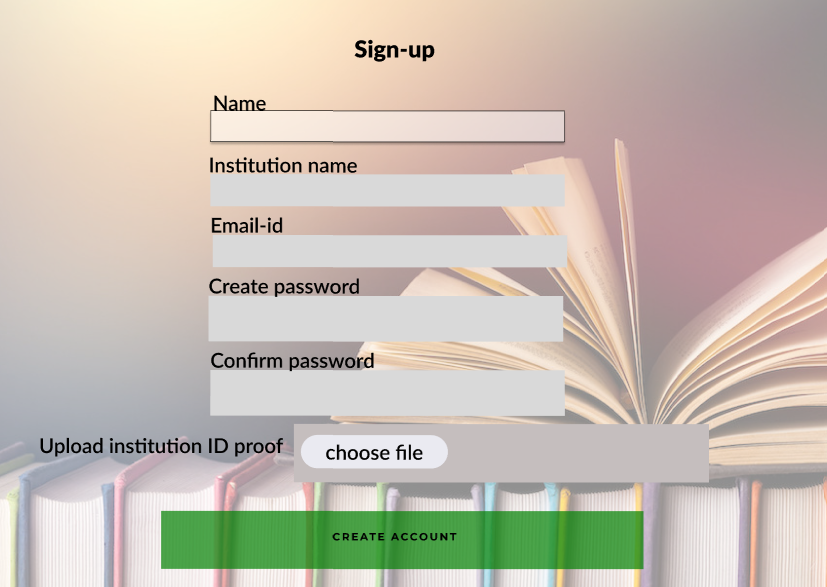
Description automatically generated**

**5.2.2 Sign-up page**

**A child sitting at a desk

Description automatically generated**

**5.2.3. Institutional Signup details**

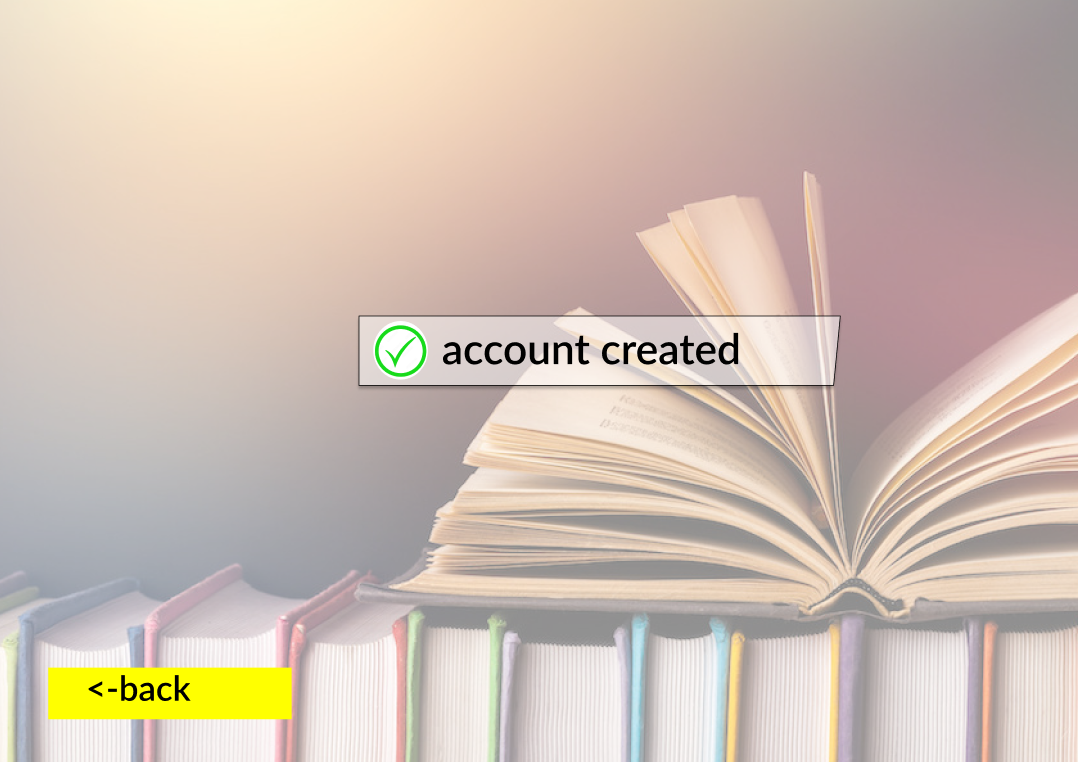
****

**5.2.4 Govt.officials signup details**

**A screenshot of a login form

Description automatically generated**

**5.2.5 Account created page**

****

**5.2.6 School Permission page**

**A screenshot of a school application

Description automatically generated**

**5.2.7 School Status Page**

**A screenshot of a school application

Description automatically generated**

**5.2.8 New Student Page**

**A screenshot of a student application

Description automatically generated**

**5.2.9 Dropout Student Page**

**A screenshot of a student application

Description automatically generated**

**A close up of a book

Description automatically generated**

**5.2.10 School Dropout Analysis**

**A screenshot of a computer

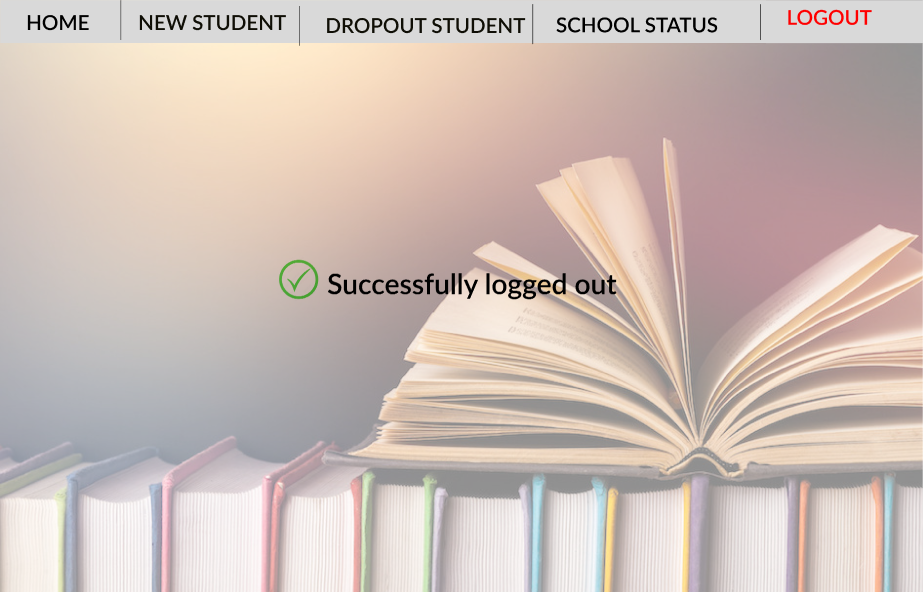
Description automatically generated**

**5.2.11 School Status Page**

**A graph on a book

Description automatically generated**

**5.2.12. Logout**

****

**6.CONCLUSION**

In conclusion, the student dropout analysis project represents a crucial initiative in the education sector, aiming to understand and address the challenges associated with student disengagement. By employing data-driven insights, the project offers a comprehensive view of dropout trends, enabling educators and administrators to proactively identify at-risk students. The integration of user roles, such as administrators, students, and data analysts, facilitates a collaborative approach in managing student information, conducting analyses, and generating actionable reports.

Through features like submitting student information, visualizing dropout statistics, and generating reports, the project empowers stakeholders to make informed decisions and implement targeted interventions. The seamless interaction depicted in use-case diagrams underscores the user-friendly nature of the system, ensuring accessibility for various roles. Interaction diagrams illustrate dynamic processes, showing how users, including administrators and data analysts, engage with the system's functionalities like login, data retrieval, analysis, and solution generation.