A red text on a black background

AI-generated content may be incorrect.

**PROJECT AND TEAM INFORMATION**

## Project Title

|  |
| --- |
| Smart Data Pre-processor: A Web-Based Tool for Cleaning and Visualizing Datasets |

## Student/Team Information

|  |  |
| --- | --- |
| Team Name: | Sapphire |
| Team member 1 (Team Lead) | Gangwar, Atharv – 230112386  [atharvgangwar8@gmail.com](mailto:atharvgangwar8@gmail.com) |
| Team member 2 | Negi, Dhruv – 23011451  [negi67291@gmail.com](mailto:negi67291@gmail.com) |
| Team member 3 | Negi, Abhishek – 23011732  [abhisheknegi75054@gmail.com](mailto:abhisheknegi75054@gmail.com) |

**PROJECT PROGRESS DESCRIPTION (35 pts)**

## Project Abstract (2 pts)

|  |
| --- |
| The Smart Preprocessor and Visualizer is a web application developed in Java that enhances data preprocessing and visualization workflows. It integrates a MongoDB database to efficiently store and manage preprocessed data, ensuring users can handle large datasets effectively.  Data Preprocessing: The application enables users to clean and structure their data, improving the quality of insights derived from subsequent analyses.  MongoDB Integration: Utilizing MongoDB allows for robust and scalable data storage, accommodating various user-defined requirements.  Interactive Visualization: A React-based visualizer, powered by Chart.js, provides an intuitive interface for users to create dynamic and interactive charts. This feature simplifies the interpretation of complex datasets.  User -Centric Design: The application is designed with a focus on user experience, offering a seamless interface for both preprocessing and visualization tasks. |

## Updated Project Approach and Architecture (2 pts)

|  |
| --- |
| File Upload and Parsing: Users can upload their CSV files through the Java interface. Upon upload, the application parses the CSV data to prepare it for preprocessing.  Data Processing Options: After parsing, users are directed to a dedicated route (/process) where they can select various preprocessing operations to apply to their data. This flexibility allows users to tailor the preprocessing steps according to their specific needs.  Data Storage: Once the preprocessing is complete, the cleaned and structured data is saved to a MongoDB Atlas database. This ensures efficient storage and management of the data for future use.  Data Visualization: For users interested in visualizing their processed data, a "Visualize" button is provided. Clicking this button redirects users to a separate React.js application that utilizes Chart.js for dynamic data visualization. This application connects to the same MongoDB database, allowing seamless access to the preprocessed data. |

## 

## Tasks Completed (7 pts)

|  |  |
| --- | --- |
| Task Completed | Team Member |
| 1. Codebase Design and Workflow 2. MongoDB Database Setup 3. Route and MVC Architecture 4. HTML Template Development 5. CSV File Parsing Implementation 6. Model Layer Code 7. Exception Handling Mechanisms | Atharv Gangwar and Abhishek Negi  Atharv Gangwar  Atharv Gangwar  Dhruv Negi  Dhruv Negi  Abhishek Negi  Abhishek Negi |

## 

## Challenges/Roadblocks (7 pts)

|  |
| --- |
| One of the primary challenges encountered during the development of the Smart Preprocessor and Visualizer was the lack of prior knowledge among team members regarding Java, the core programming language used for the project. This presented a significant hurdle, as understanding Java's syntax, libraries, and frameworks was essential for successful implementation. Another challenge arose from managing the numerous dependencies required for the project. With a variety of libraries and frameworks needed for both the Java backend and the React frontend, ensuring compatibility and proper configuration became a complex task. Additionally, the creation of mathematical functions for data preprocessing posed another significant challenge. Developing accurate and efficient algorithms to handle various preprocessing tasks, such as normalization, scaling, and data transformation, required a deep understanding of both the mathematical concepts involved and their practical implementation in code. Despite these challenges, the team's commitment to learning and problem-solving ultimately led to a successful project outcome, demonstrating resilience and adaptability in the face of obstacles. |

## Tasks Pending (7 pts)

|  |  |
| --- | --- |
| Task Pending | Team Member (to complete the task) |
| * Implementing the React-Based-Data-Visualizer | Atharv Gangwar and Dhruv Negi |

## 

## Project Outcome/Deliverables (2 pts)

|  |
| --- |
| User -Friendly Interface: A Java-based interface that allows users to easily upload CSV files and select preprocessing operations, enhancing user experience.  Efficient Data Processing: Implementation of robust data parsing and preprocessing functionalities, ensuring that users can clean and structure their data effectively.  MongoDB Integration: A fully functional MongoDB database that securely stores preprocessed data, enabling efficient data management and retrieval.  Dynamic Data Visualization: A separate React.js application utilizing Chart.js for interactive data visualization, allowing users to gain insights from their processed data seamlessly.  Learning Experience: The project fostered significant learning among team members, particularly in Java programming, dependency management, and mathematical function development. |

# Progress Overview (2 pts)

|  |
| --- |
| The Smart Preprocessor and Visualizer project has made significant strides since its inception. Initially, the team focused on understanding Java, which was a new language for all members. Through collaborative learning and hands-on practice, we quickly gained proficiency, enabling us to build the core functionalities of the application.  We successfully implemented the file upload feature, allowing users to upload CSV files for preprocessing. The parsing and data cleaning processes were developed, ensuring that users can effectively prepare their data for analysis.  The integration of MongoDB was completed, providing a reliable database solution for storing preprocessed data. We also designed the application’s routes and established a Model-View-Controller (MVC) architecture, promoting organized code management.  In parallel, the React.js frontend was developed, featuring an interactive visualizer powered by Chart.js. This allows users to visualize their data seamlessly.  As of now, we are finalizing the documentation and conducting thorough testing to ensure the application functions as intended. Overall, the project is on track, with all major components in place and a clear path toward completion. |

# Codebase Information (2 pts) (Repository link, branch, and information about important commits.)

|  |
| --- |
| Repository Link: <https://github.com/AtharvGangwar48/Smart_Preprocessor_and_Visualizer>  Branch: Origin Commits: “[Initial commit and Project Done](https://github.com/AtharvGangwar48/Smart_Preprocessor_and_Visualizer/commit/07471b7bbb33a57f180d0e78645bb316cefd0169)” (20th May 2025) (Pushed in a single commit to origin via IntelliJ IDE) |

## Testing and Validation Status (2 pts)

|  |  |  |
| --- | --- | --- |
| Test Type | Status (Pass/Fail) | Notes |
| Unit Testing  Integration Testing  Route Testing  Responsive Testing  Error Handling Validation | Pass  Pass  Pass  Pass  Pass | Individual components of the application, such as the CSV parsing functions and data preprocessing algorithms.  Integration tests were conducted to verify that different components of the application, like Java backend worked seamlessly together. This included testing the data flow from file upload to MongoDB storage.  Each route in the application was thoroughly tested to ensure proper functionality. This included verifying that users could successfully navigate from the file upload page to the processing page and then to the visualization page.  All pages of the application were tested for responsiveness across different devices and screen sizes. The layout and functionality were verified to ensure a consistent and user-friendly experience on desktops, tablets, and mobile devices. |

# Deliverables Progress (2 pts)

|  |
| --- |
| The project is tracking well, with significant progress achieved on key deliverables:   * **Java-based Preprocessing:** Completed successfully. * **React-based Visualizer:** Near completion. Integration is the next step to finalize this deliverable.   We anticipate completing the full project shortly following the successful integration of these components. |