WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Result Analyzer

[1]Veeraj Kiran Panchal, [2]Soham Naresh Batawle, [3]Asma Fakhruddin Khan, [4]Prof. Mithun Mhatre

[1,2,3] Students, [4] Professor (Guide)

[1,2,3,4] Computer Department, *Bharati Vidyapeeth Institute Of Technology, Kharghar , Navi Mumbai , India*

1Sohambatawle@gmail.com 2veerajhome11@gmail.com 3asmakhan98558@gmail.com 4bvit.mithun@gmail.com

ABSTRACT—

The “Result Analyzer” is an automation tool de- signed to streamline the process of fetching and analyzing student results from the Maharashtra State Board of Technical Education (MSBTE) website. Traditionally, teachers manually retrieve and compile student results, which is time-consuming and error- prone. This project automates the process by extracting result data directly from the MSBTE website, converting JSON data into Excel format, and making the analyzed data available for download. The system employs Spring Boot, Selenium, JSP, JavaScript, and Apache POI to ensure efficient data handling. By eliminating manual efforts, the Result Analyzer enhances accuracy, reduces workload, and improves efficiency.

***Keywords*— Web Scraping, Selenium Automation , Spring Boot Application**

I. Introduction

Analyzing student results is a crucial yet tedious task for educators. The existing manual process requires teachers to visit the MSBTE website, enter enrollment numbers one by one, solve CAPTCHAs, and extract results for multiple students. This approach is inefficient, leading to delays and possible human errors.

To address these challenges, the ”Result Analyzer” automates the entire process. The user uploads an Excel file containing enrollment numbers, and the system fetches corresponding results using Selenium web scraping techniques. The extracted data is then processed and formatted into an Excel file, ensuring seamless access and analysis.

This paper discusses the architecture, working mechanism, technologies used, and real-world benefits of the Result An- alyzer system, highlighting its role in revolutionizing result analysis for educational institutions.

II. Methodology

The methodology outlines the systematic approach used to develop the Result Analyzer system. It consists of multiple phases, from data extraction to processing and result generation.

A. System Workflow

The workflow is divided into frontend interaction, backend processing, and automation execution:

1) **User Input Handling:** The user uploads an Excel file containing enrollment numbers or enters an individual enrollment number manually on the web interface.

The system validates input and enables/disables options dynamically using JavaScript. [1]

2) **Backend Processing:** The request is sent to the Spring Boot backend, where controllers handle incoming data. Data is mapped to an Automation Parameter class using Spring Boot’s automatic request binding.

3) **Automation Execution :** Selenium WebDriver launches a Chrome browser instance and navigates to the MSBTE result page. The script fills in the enrollment number and waits for user CAPTCHA input.

After CAPTCHA verification, the system extracts the result data using DOM parsing.

4) **Data Conversion and Storage :** Extracted data is stored in an array and processed using Apache POI for Excel file modification.

The writeInto Excel method appends data into an Excel file while preserving existing content.

5) **File Delivery :** The processed Excel file is con- verted into a downloadable response (ResponseEn- tity<Resource>).

JavaScript handles the file download operation on the user's browser.

B. Workflow Breakdown

This section provides a simple representation of the system's working, helping to understand how different components interact. It explains the logical flow, showing how the input data (enrollment numbers or Excel file) is processed, how results are fetched, and how they are written back into the Excel file for the user to download.

By automating this process, the system eliminates manual result searching, reduces errors, and significantly improves efficiency in academic result analysis.

1) User uploads an Excel file / enters enrollment number.
2) Data Extraction:
   - If an Excel file is uploaded → Extract roll numbers.
   - If an enrollment number is entered → Proceed with fetching.
3) Automated System Fetches Results:
   - Selenium opens MSBTE website.
   - Captcha is manually entered.
   - Results are extracted and stored.
4) Data is written back into the Excel file.
5) User downloads the updated Excel file.

C. Technical Details

The development of the Result Analyzer system involves multiple technologies that work together to automate result extraction and processing. Each component plays a crucial role in ensuring seamless data retrieval, conversion, and user interaction. The frontend provides an intuitive interface, the backend processes requests and manages automation, and the data handling layer ensures efficient storage and retrieval of results. The table below summarizes the key technologies used in the system.

TABLE I

Key Technologies Used in the Result Analyzer System

Technology	**Description**
Spring Boot	Handles backend logic, request mapping, and data processing.
Selenium WebDriver	Automates interaction with the MSBTE website.
JSP and JavaScript	Provides the user interface and manages frontend actions.
Apache POI	Enables Excel file creation and modification.
HTTP Requests and Re- sponse Entity	Ensures seamless data transfer be- tween client and server.

III. Key Features

Following are some of the key features of this application along with real - world examples which will help you to understand the concept in simpler and easier way.

A. Automated Data Extraction

- The system is designed to fetch student results directly from the MSBTE website using automation techniques. Instead of manually searching for results, the tool auto- mates the process by extracting necessary data such as student names, marks, grades, and pass/fail status.
- Example: Imagine a college with 500+ students; manually fetching results would take hours. This tool automates the process, reducing errors and saving time.

B. ***Bulk Data Processing for Multiple Students***

- Instead of processing one student at a time, the tool allows teachers to upload an Excel file containing multiple stu- dent enrollment numbers. The system then processes each record sequentially and retrieves corresponding results.
- Example: A university professor needs results for an entire class of 100 students. Uploading an Excel file simplifies this, allowing the system to process data au- tomatically without manual intervention.

C. ***JSON to Excel Conversion***

- The MSBTE website provides student results in JSON format [2], which is not user-friendly for most educators. The tool converts this data into a structured Excel sheet for easy readability and analysis.
- Example: Instead of copying and pasting raw text from the website, teachers receive a well-organized Excel sheet, making it easy to analyze pass/fail trends, subject performance, and overall results.

D. ***Web-Based Interface for Ease of Use***

- The tool provides a web-based UI that allows users to enter individual enrollment numbers or upload Excel files for bulk processing. It ensures user-friendliness and eliminates the need for command-line operations.
- Example: Teachers or non-technical staff can easily use the tool via a simple webpage instead of writing code or running scripts manually.

E. ***Secure and Efficient Data Handling***

- The tool ensures data security [3] and efficient processing by handling requests systematically, avoiding redundancy, and ensuring user data is not misused.
- Example: A teacher uploads an Excel file with confi- dential student information. The system ensures this data is not stored permanently but only processed during the session, protecting sensitive data.

F. ***Downloadable Report Generation***

- After processing, the system generates a final Excel report containing the results of all students, which can be downloaded directly from the web portal.
- Example: A school administrator needs a compiled report of all students for printing. The system generates an Excel sheet in seconds, reducing manual effort.

IV. Implementation Strategy

The implementation of the Result Analyzer tool involves a structured workflow, from system initialization to data extrac- tion and file processing. This section details both the internal and external working of the system, explaining how user inputs are handled, data is fetched from the MSBTE website, and results are processed into downloadable Excel files.

A. ***Phase 1: System Initialization***

1) Starting the Tool:
   - The tool is designed specifically for extracting and analyzing data from the MSBTE website.
   - The application is launched from Eclipse (temporary setup for development and testing).
   - The user opens Chrome (currently the tool supports only Chrome) and navigates to http://localhost:8080/.
   - The home page is displayed, allowing the user to select the input type (Enrollment/Seat Number or file upload).

B. ***Phase 2: User Input Handling and Validation***

2) User Interaction with the Web Interface:
   - The home page is built using JSP and displays HTML components.
   - A background JavaScript file (dynamicHandler.js) manages user interactions dynamically.
   - When the user selects the input mode (Enroll- ment/Seat Number or file upload), the JavaScript dynamically enables or disables relevant fields and buttons.
   - The “Quick Test” button is enabled only when a file is uploaded, and the “Get Data” button is enabled when an Enrollment/Seat Number is entered.

3) Quick Test Feature (Single Entry Validation):

- If the user is unsure about the entered details, they can use the Quick Test feature.
- This triggers the /quickTest method in the controller class.
- The system extracts the first Enrollment/Seat Num- ber from the uploaded Excel file.
- It then calls the fetchData method to fetch the corresponding result.
- The fetched result is displayed in an alert pop- up (handled by dynamicHandler.js), confirming whether the details entered are correct.
- This ensures that the user is confident in proceeding with the bulk extraction process.

C. ***Phase 3: Data Extraction Process***

4) Controller Class and Request Handling:

- The controller listens to four major request map- pings:

(i) @RequestMapping(/): Maps requests to the home page (home.jsp).

(ii) @PostMapping(/automate): Handles individual student result requests.

(iii) @PostMapping(/automateBulkData): Processes multiple student results in bulk mode.

(iv) @PostMapping(/quickTest): Handles quick test requests from uploaded files. [5]

5) Fetching and Processing Data from MSBTE Website:

- When a user submits Enrollment/Seat Number and clicks “Get Data,” the /automate method is trig- gered.
- The system collects the input data using an object- oriented approach:

(i) A custom data class (AutomationParameter) stores and retrieves user input fields dynamically.

(ii) Spring Boot automatically maps request param- eters to class properties, ensuring smooth data transfer.

- The collected data is passed to the fetchData method in the Analyzer class.
- The fetchData method initiates Selenium-based web scraping:

(i) Opens the MSBTE website.

(ii) Enters the Enrollment/Seat number.

(iii) Extracts specific result data from the webpage.

(iv) Returns the extracted data to the /automate method.

- The extracted data is then passed to the writeIntoEx- cel method:

(i) Writes the fetched results into an Excel file.

(ii) Converts the file into a Byte Data Object.

(iii) Returns the Byte Data as a (ResponseEn- tity<Resource>) to enable file download for the user.

D. ***Phase 4: Bulk Data Processing and File Download***

6) Automate Bulk Data Processing:

- If the user wants to process multiple results, they upload an Excel file containing Enrollment/Seat Numbers. [4]
- The system reads the list of numbers from the file.
- Iterates through the list, calling fetchData for each student.
- Collects and stores all results in an array.
- Passes the extracted data to writeIntoExcel to update the original Excel file.
- Converts the updated Excel file into a downloadable format and sends it as a response to the user.

*** Summary of Implementation Workflow**

1) There are two major modules In this system : label=)

a) Extractor

b) Analyzer

2) System Initialization: The application starts, and the web interface is loaded.

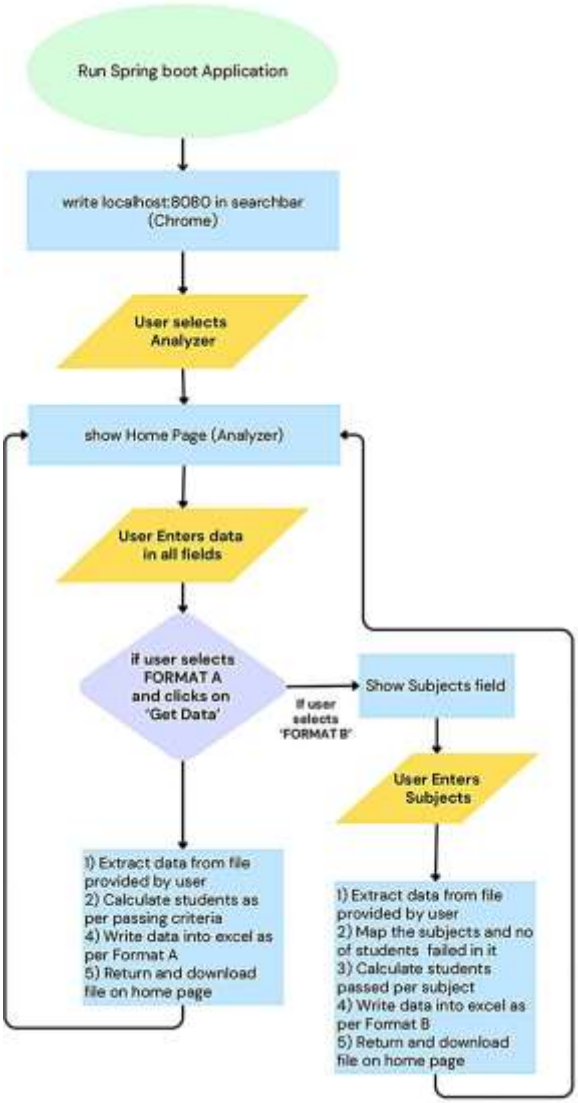

Fig. 1. Extractor Module

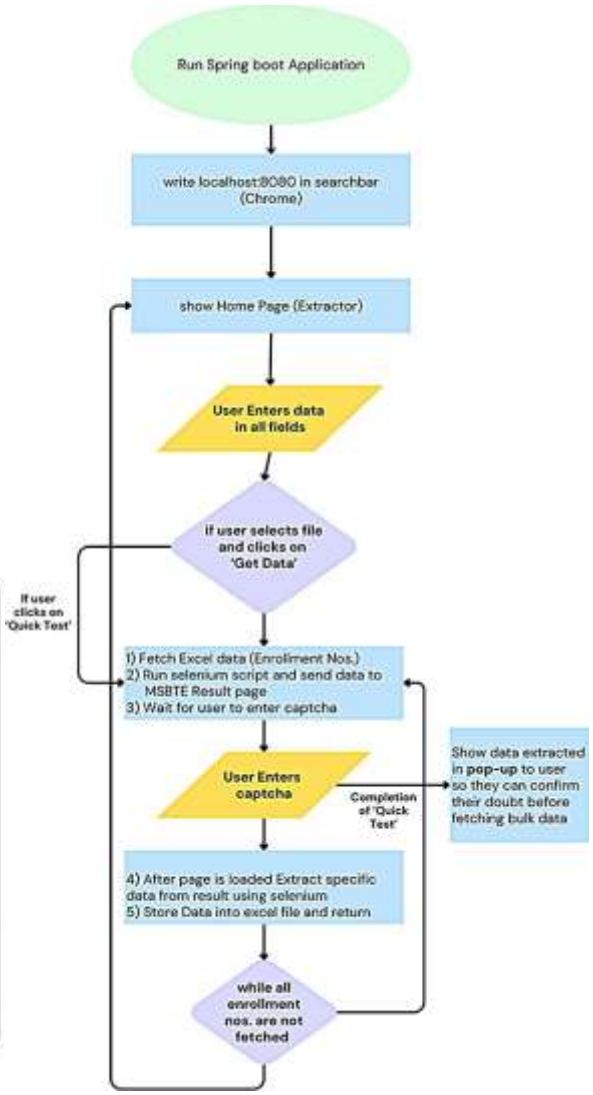

Fig. 2. Analyzer Module

3) User Input Handling and Validation: The user selects the input mode, enters details, and validates using Quick Test.

4) Data Extraction: The system fetches student results from the MSBTE website and processes them.

5) Bulk Data Processing and File Download: The fetched data is written into an Excel file, which is provided for user download.

V. Expected Results and Impact

The Result Analyzer is designed to streamline the process of fetching, analyzing, and organizing student results, ensuring accuracy, efficiency, and ease of access. By automating result retrieval and eliminating manual data entry, this system significantly reduces processing time, minimizes human errors, and enhances productivity for educators.

1) **Time Efficiency: Significantly Reduces the Time Re- quired for Result Retrieval and Processing**

- One of the major impacts of the Result Analyzer is the drastic reduction in the time required to fetch and analyze student results. Traditionally, educators or administrative staff manually search for each student's result on the MSBTE website, enter data into records, and generate reports, which is a time- consuming process.

- Example -

Consider a teacher managing results for 200 students. Manually retrieving, recording, and verifying each result may take several hours or even days.

With the Result Analyzer, this entire process is automated, retrieving all results within minutes. Just like how online banking has replaced long queues in physical banks, this system eliminates the waiting time associated with result collection.

2) **Error Reduction: Minimizes Human Errors Associated with Manual Data Entry**

- When results are recorded manually, mistakes in entering enrollment numbers, misplacing decimal points, or copying incorrect grades are common. This can lead to serious consequences, including incorrect student reports and grading errors.
- Example -

Imagine a school administrator inputting results manually into a spreadsheet and mistakenly record- ing a student's marks as 82 instead of 28. This could misrepresent student performance and af- fect decision-making in promotions or scholarships. With automation, such errors are eliminated, as data is fetched and stored programmatically without manual intervention, ensuring accuracy similar to how barcode scanning in supermarkets prevents pricing errors.

3) **Enhanced Productivity: Allows Educators to Focus on Student Performance Analysis Rather than Administrative Tasks**

- By automating the data collection and analysis pro- cess, educators and administrators can spend more time understanding student performance trends rather than spending hours retrieving and recording results.
- Example -

A college professor handling multiple classes can use the automated tool to get results instantly. Instead of spending hours fetching data, they can focus on analyzing patterns, such as identifying students struggling in specific subjects, and plan remedial sessions accordingly. Similar to how au- tomation in factories allows workers to focus on quality control instead of repetitive manual tasks, this system lets educators prioritize student growth rather than paperwork.

4) **Data Accuracy: Ensures Precise Extraction and For- matting of Results**

- The system guarantees that the fetched data is accurate, eliminating inconsistencies that often arise when results are recorded manually. The formatting of results in a structured manner ensures readability and ease of further processing.
- Example -

Consider a scenario where universities compile semester results for ranking and accreditation pur- poses. If different departments enter data manu- ally, inconsistencies in formats (e.g., some using percentages, some using CGPA) may occur. With the Result Analyzer, data is extracted directly from the official MSBTE portal and converted into a structured Excel format, ensuring uniformity and eliminating discrepancies. It works similarly to how GPS systems provide accurate directions instead of relying on manual mapping errors.

5) **Scalability: The System Can Handle Large Volumes of Student Data Efficiently**

- Educational institutions may have thousands of students, and handling such a large volume of data manually is not feasible. The Result Analyzer ensures that increasing the number of students does not affect performance.
- Example -

An engineering college with 5,000+ students con- ducts semester exams. If each faculty member has to manually download and enter results, the workload becomes unmanageable. The Result Analyzer scales effortlessly, processing thousands of records in the same time it takes to process ten. This scalability is comparable to cloud computing services like Google Drive, which allow users to store and access large amounts of data without performance issues.

6) **Improved Decision-Making: Enables Quick Access to Structured Data for Analysis and Decision-Making in Educational Institutions**

- Having organized and accurate data readily avail- able allows institutions to make informed decisions, whether for academic improvements, student counseling, or curriculum changes.
- Example -

If an institution notices through the Result Analyzer that 60% of students are failing a particular subject, they can take corrective actions, such as arranging extra lectures or revising the curriculum. This is similar to how business analytics tools help companies analyze market trends and make strategic decisions based on real-time data.

VI. Future Scope – Expanding the Impact of Result Analyzer

The future scope of the Result Analyzer project includes several enhancements that will improve its usability, efficiency, and scalability. Below is a detailed breakdown of each point:

A. Automated CAPTCHA Solving

(i) Current Issue:

Right now, users have to manually enter the CAPTCHA while fetching results, which makes the process slow and requires human intervention.

(ii) Future Enhancement:

- AI-based Optical Character Recognition (OCR) tech- niques can be used to automatically read and enter the CAPTCHA.
- Deep learning models such as Tesseract OCR or Convolutional Neural Networks (CNNs) can be trained to recognize different CAPTCHA patterns.
- This will make the process fully automated and eliminate the need for manual input.

B. Mobile Application Integration

(i) Current Issue:

Right now, the tool runs only on a desktop web browser, making it less convenient for users who need to check results on the go.

(ii) Future Enhancement:

- Develop a mobile application for Android and iOS.
- The app will allow users to upload Excel files, fetch results, and analyze data directly from their smartphones.
- It will provide push notifications when results are available, eliminating the need to check manually.

C. Cloud Storage Support

(i) Current Issue:

Currently, results are stored only in an Excel file, which can get lost or deleted.

(ii) Future Enhancement:

- Store fetched results in cloud-based storage services such as Google Drive, AWS, or Firebase.
- This will allow users to access results from any device, at any time.
- Implement user authentication to ensure secure ac- cess and prevent unauthorized use.

D. Performance Optimization

(i) Current Issue:

Processing bulk data (large sets of enrollment numbers) takes time, slowing down result fetching.

(ii) Future Enhancement:

- Use multi-threading to fetch multiple results at the same time, reducing waiting time.
- Optimize code efficiency and database queries to improve speed.
- Implement load balancing to handle large numbers of requests without server crashes.

E. Multi-Board Support

(i) Current Issue:

Currently, the tool is designed only for MSBTE results.

(ii) Future Enhancement:

- Extend the tool to support other educational boards like CBSE, ICSE, and State Boards.
- Implement a universal format converter that can handle different data structures from various board websites.

F. Graphical Analysis and Reporting

(i) Current Issue:

Right now, results are presented only in text and tabular format, making it difficult to analyze performance trends.

(ii) Future Enhancement:

- Add graphical representation of results using charts, graphs, and heatmaps.
- Introduce features like trend analysis, subject-wise comparison, and historical performance tracking.
- Provide automated performance reports for teachers, parents, and students.

Conclusion

The Result Analyzer project highlights how technology can simplify and improve traditional methods of handling student results. In many educational institutions, teachers and administrators manually retrieve, analyze, and store student results, which is a time-consuming and error-prone process. This project automates these tasks, making the entire process faster, more efficient, and more reliable.

By utilizing techniques such as web scraping, data processing, and file handling, this tool fetches student results directly from the MSBTE website and converts them into a structured Excel file. This eliminates the need for teachers to manually search for and input results, significantly reducing their workload and ensuring greater accuracy in data handling.

One of the biggest advantages of this tool is that it minimizes human errors that often occur when handling large amounts of student data manually. Additionally, since the results are stored in an Excel file, they can be easily analyzed, shared, and stored for future reference.

The Result Analyzer is not only useful for teachers but also beneficial for students and educational institutions. It provides a quick and organized way to access results, ensuring that data is processed without delays.

References

[1] The Coding Train, "Tabular Data - Working With Data & APIs in JavaScript," https://www.youtube.com/watch?v=RfMkdvN-23olist=PLRqwX-V7Uu6YxDKpFzf2*D*84*p*0*cyk*4*T* 7*Xindex* = 4

[2] The Coding Train, "JSON - Working with Data and APIs in JavaScript,"

[3] "Data from JSON - Getting Data from JSON," AnyChart Documenta- tion. https://docs.anychart.com/Working*withData/DataF romJ SON*

[4] "Data conversion from JSON to Excel," How-To Geek. https://www.howtogeek.com/775651/how-to-convert-a-json- file-to- microsoft-excell

[5] K. Dale, "*Data Visualization with Python and JavaScript: Scrape, Clean, Explore & Transform Your Data*", O'Reilly Media, 2016. https://www.pdfdrive.com/data-visualization-with- python-and- javascript-scrape-clean-explore-transform-your- data-e186354043.html