

INTERNSHIP REPORT

AUG-SEP, 2023



APOORV D
COMPUTER SCIENCE & ENGINEERING
VIT

ACKNOWLEDGEMENT

I would like to extend my sincere gratitude to Dr. B. VENKATRAMAN, Director of IGCAR, whose visionary leadership has created an environment truly conducive to learning and research. I would also like to express my gratitude to Mr. JEHADEESAN R, Head, Computer Division. Under their guidance, I have had the privilege to excel and innovate in my academic pursuits.

To my esteemed guide, Dr. M.L. Jayalal, Head, Computer Systems Section, and Ms. Thirupurasundari D, SO/E, I owe an immeasurable debt of gratitude. Their constant encouragement, profound insights, and dedication to teaching have not only steered me through this project but have also ignited in me a passion for research. I consider myself incredibly fortunate to have had the privilege of working under their mentorship. Their invaluable insights and unwavering encouragement have been instrumental in elevating the quality of my work. Their expertise in their respective fields has truly enriched my academic journey.

In conclusion, I am immensely grateful for the exceptional guidance and support I have received during my tenure at IGCAR, Kalpakkam. The influence has been pivotal in shaping my academic journey.

OVERVIEW

IGCAR, founded in 1971, is India's primary center for Sodium Cooled Fast Breeder Reactor (FBR) research. Its mission is to advance FBR technology and associated fuel cycle facilities. This includes research in materials, techniques, and systems for FBRs, and fundamental breakthroughs in Fast Reactor technology.

Key achievements:

- FBTR, a 40 MWt reactor, was the first of its kind globally and is powered by Plutonium Uranium mixed carbide.
- IGCAR is constructing a 500 MWe Prototype Fast Breeder Reactor (PFBR) and a Fast Reactor Fuel Reprocessing Plant.
- KAMINI, a 30 KWt U²³³ fueled mini reactor, supports neutron-based research.
- IGCAR collaborates with national and international institutions, contributing to sectors like Defense and Space.
- It houses a modern library, a well-equipped Central Workshop, and a Computer Division with advanced servers.
- Dr. B. Venkatraman became Director in September 2021, overseeing an annual budget of around 800 crore rupees.

IGCAR's multifaceted work spans nuclear technology, various scientific disciplines, and international partnerships, with a focus on advancing India's nuclear capabilities.

COMPUTER DIVISION

The Computer Division at IGCAR is equipped with state-of-the-art technology, including Silicon Graphics Power Challenge L servers, SGI workstations, and 8-node Xeon Servers, to effectively meet the computational needs of its users. The center employs a dedicated workforce of 2814, comprising 1187 engineers and scientists. With an annual budget of approximately 754.47 crore rupees, the center sustains its multifaceted activities and plans.

This division's advanced technology and comprehensive services contribute significantly to IGCAR's research and operational excellence, ensuring efficient support for various computational and communication needs.

The Computer Division plays a vital role in various aspects, such as the development of a PFBR operator training simulator, high-performance computing applications, 3D models, and plant walkthroughs. Additionally, it specializes in wireless sensor-based systems and computational intelligence-based systems for reactor applications, alongside creating application software tailored to user requirements. Furthermore, the division offers services like a high-performance computational facility, internet access, email services, local area networking, video conferencing, national knowledge network connectivity, web services, knowledge management, and user consultancy.

Contents

CHAPTER 1	6
GETTING STARTED	6
Understanding MVC	6
Setting Up WAMP Server	6
Building a PHP MVC Application	7
Testing and Debugging	8
Deployment	8
Conclusion	8
CHAPTER 2	9
EDUSYNC	9
MVC Architecture	9
Authentication and Session Management	9
Data Validation and Error Handling	9
Data Management	10
Technology Stack	10
Screenshots and Visualization	11
Conclusion	12
CHAPTER 3	13
HPCS PORTAL	13
User Privileges and Access Control	13
Data Collection and Monitoring	13
Database Design and Management	14
Web Design and User Interface	15
Screenshots and Visualization	16
Conclusion	16
CHAPTER 4	17
DOMAIN OF WORK.....	17
Abaqus Unified FEA	17
ANSYS	17
COMSOL Multiphysics	18
CHAPTER 5	19
LOG PARSING & ANALYSIS	19
MVC Architecture	19
Authentication and Session Management	19
Data Validation and Error Handling	19

Software-Specific Log Parsing and Analysis	20
Custom Time Interval Analysis	21
Technology Stack	21
Log Parsing Methodology	21
Screenshots and Visualization	23
Conclusion	28
CHAPTER 6	29
LICENSE HISTORY ANALYSIS	29
MVC Architecture	29
Authentication and Session Management	29
Data Validation and Error Handling	29
View Activity	30
Software Interaction Timeline	30
Day-to-Day Capability Analysis	30
Technology Stack	31
Screenshots and Visualization	31
Conclusion	31
CHAPTER 7	32
WORKSTATION ANALYSIS	32
MVC Architecture	32
Authentication and Session Management	32
Data Validation and Error Handling	33
Utilization Analysis	33
Technology Stack	33
Screenshots and Visualization	34
Conclusion	35
CHAPTER 8	36
CONCLUSION	36

CHAPTER 1

GETTING STARTED

Understanding MVC

Components in the PHP MVC architecture:

1.1. Model:

- Represents the application's data and logic.
- Interacts with databases (e.g., MySQL) to manage data.
- Manages data storage, retrieval, updates, and validation.

1.2. View:

- Deals with how data is presented to users.
- Utilizes HTML templates with embedded PHP code for dynamic web page generation.
- Defines the structure and layout of the user interface.

1.3. Controller:

- Manages user input, processes requests, and acts as a mediator.
- Receives data from the View.
- Interacts with the Model to retrieve or manipulate data.
- Determines and renders the appropriate View for the user.

Setting Up WAMP Server

To embark the PHP MVC journey with WAMP, I had to set up my development environment:

2.1. Install WAMP: Download and install WAMP Server, which includes Apache (the web server), MySQL (the database server), and PHP (the scripting language). One can choose a WAMP distribution like WampServer or XAMPP, which includes these components and simplifies the installation process.

2.2. Configure PHP: Customize PHP settings according to project requirements by editing the `php.ini` file. One can adjust parameters like file upload limits, error reporting levels, and database connection settings.

2.3. Create Virtual Hosts: Set up virtual hosts in the Apache configuration to define the project directories and URLs. This allowed me to work on multiple projects simultaneously, each with its own domain.

Building a PHP MVC Application

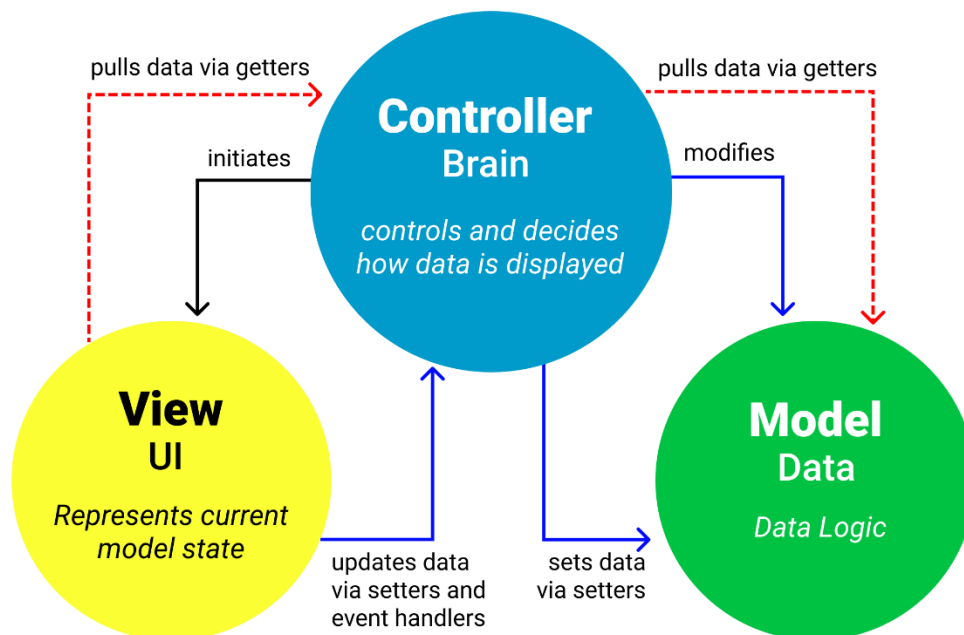
Directory Structure:

- Organize project with a structured directory layout to maintain clarity and organization.
- Consider the following example structure:

- /your-project/
 - /app/
 - /controllers/
 - /models/
 - /views/
 - /public/
 - /css/
 - /js/
 - /images/
 - /config/
 - /vendor/

- In this layout, `/app` contains directories for controllers, models, and views, which keeps your application logic separated.
- The `/public` directory is meant for publicly accessible assets like CSS, JavaScript, and images.
- `/config` is where one can store configuration files.
- `/vendor` can be used for third-party libraries or dependencies.

MVC Architecture Pattern



Model: Create PHP classes in the `/models/` directory to handle data manipulation, interactions with the database, and business logic.

View: Design user interfaces using HTML templates with embedded PHP code. Place these templates in the `/views/` directory.

Controller: Build PHP controllers in the `/controllers/` directory to manage user requests, interact with models, and render views.

Routing: Implement URL routing to map URLs to specific controllers and actions. One can use tools like Apache's `.htaccess` file or routing libraries to achieve this.

Database Connectivity: Configure database connections in your application. PHP offers MySQLi and PDO extensions for secure and efficient database interactions.

Templating: Consider using a templating engine like Smarty or Blade to separate PHP logic from HTML presentation in your views.

Testing and Debugging

PHP MVC applications on WAMP can be effectively tested and debugged:

Debugging Tools: Utilize debugging tools like Xdebug or built-in PHP debugging to identify and resolve issues in your code.

Testing Frameworks: Implement testing frameworks such as PHPUnit for unit and integration testing to ensure the reliability of your application.

Deployment

When PHP MVC application is ready for deployment:

Hosting: Choose a web hosting provider that supports PHP and MySQL. Transfer the application files and database to the hosting server.

Server Configuration: Configure the production server environment to match the development environment, including PHP settings and database connections.

Security: Implement security measures such as input validation, output escaping, and user authentication to protect your application from common web vulnerabilities.

Conclusion

By harnessing the power of PHP MVC and the convenience of the WAMP server stack, one can develop robust, modular, and maintainable web applications. With a well-structured directory layout, effective use of MVC components, and rigorous testing, one can be well on their way to creating successful PHP applications for the web.

CHAPTER 2

EDUSYNC

In this chapter, we will delve into the core functionalities of EduSync, a robust Student Database Management System built on the Model-View-Controller (MVC) architecture. EduSync is designed to streamline student data management, provide powerful data analysis tools, and ensure data security and integrity.

MVC Architecture

EduSync adheres to the MVC architectural pattern, which enhances maintainability and modularity in our application. Here's a brief overview of how it is implemented:

- Model (M): The Model represents the data layer of our system. It interacts with the database, performing operations such as inserting, retrieving, updating, and deleting student records. All data-related operations are centralized within the Model to ensure consistency and data integrity.
- View (V): The View is responsible for the presentation layer. It displays student data using HTML templates and interacts with the user through an intuitive web interface. User interactions trigger actions in the Controller to maintain a separation of concerns.
- Controller (C): The Controller acts as an intermediary between the Model and View. It receives user input, processes requests, performs validations, and communicates with the Model to retrieve and manipulate data. The Controller ensures that the right data is presented to the View and that the user's actions are appropriately handled.

Authentication and Session Management

EduSync implements secure session management using PHP's `session_start()` function. This ensures that only authorized users can access the system. When users log in, a session is initiated, and their credentials are verified against the stored data in the database. If authentication is successful, users gain access to the system's functionalities based on their roles.

Data Validation and Error Handling

Robust data validation mechanisms are in place to ensure that user-input data is accurate and safe. Validation rules are enforced both on the client-side using JavaScript and on the server-side using PHP. This double layer of validation enhances data integrity.

EduSync also incorporates comprehensive error handling. Any errors that occur during user interactions or database operations are gracefully handled, preventing unexpected application crashes and providing informative error messages to users.

Data Management

Data Addition

EduSync allows authorized users, particularly administrators, to add new student records seamlessly. User-provided data is sanitized and validated before insertion into the MySQL database via PHP. This ensures the integrity and consistency of the stored data.

Viewing, Filtering, and Sorting

Students and administrators can view and manage student data effortlessly. Data is presented in an organized and user-friendly manner using DataTables, a dynamic JavaScript library. DataTables facilitate features like sorting, searching, and pagination, providing a smooth user experience.

Computational Graphs and Data Analysis

One of the standout features of EduSync is its ability to generate computational graphs based on student data. We leverage Chart.js, a popular JavaScript library, to create interactive and visually appealing graphs. Users can analyze data trends, such as academic performance, attendance, or demographic distributions, through these graphs. This analytical capability empowers educators and administrators to make data-driven decisions.

Admin Access

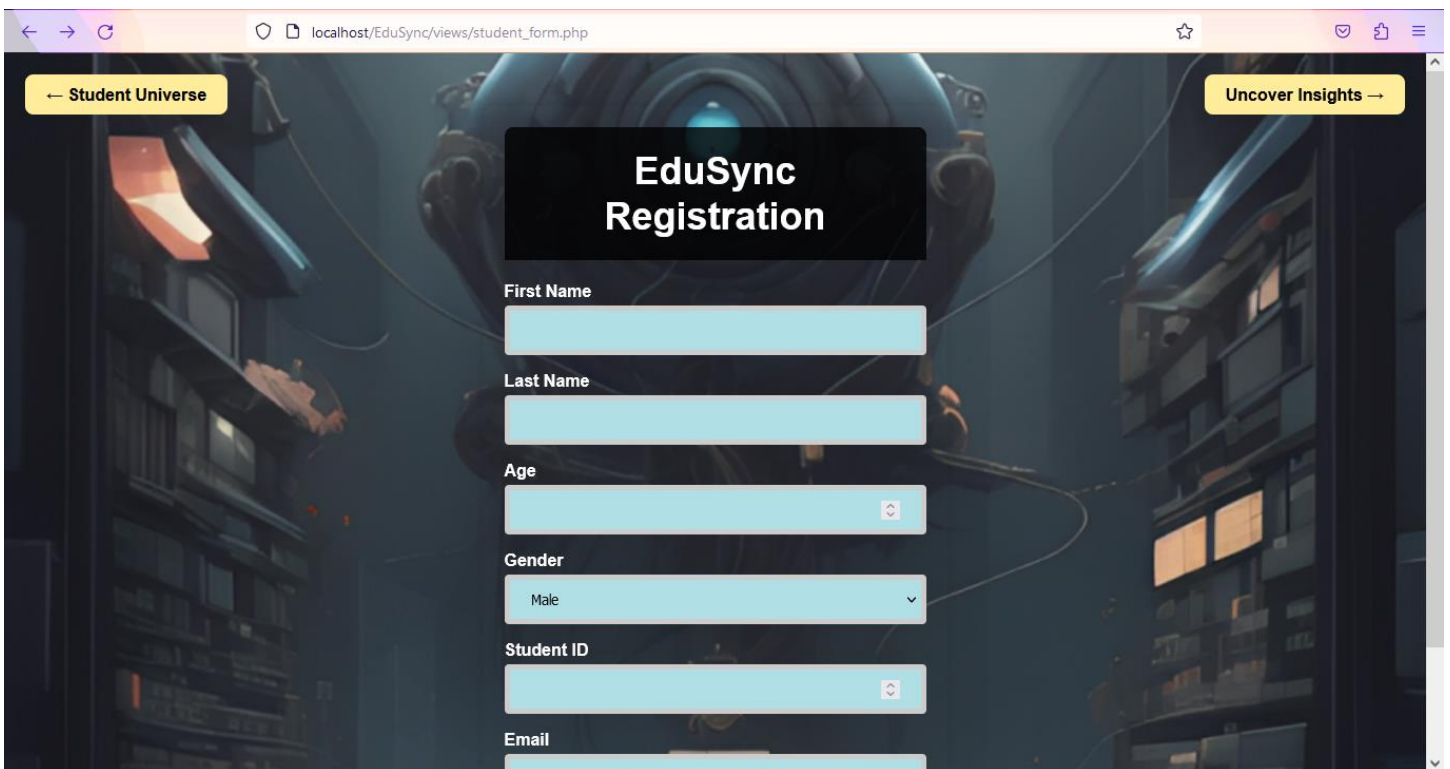
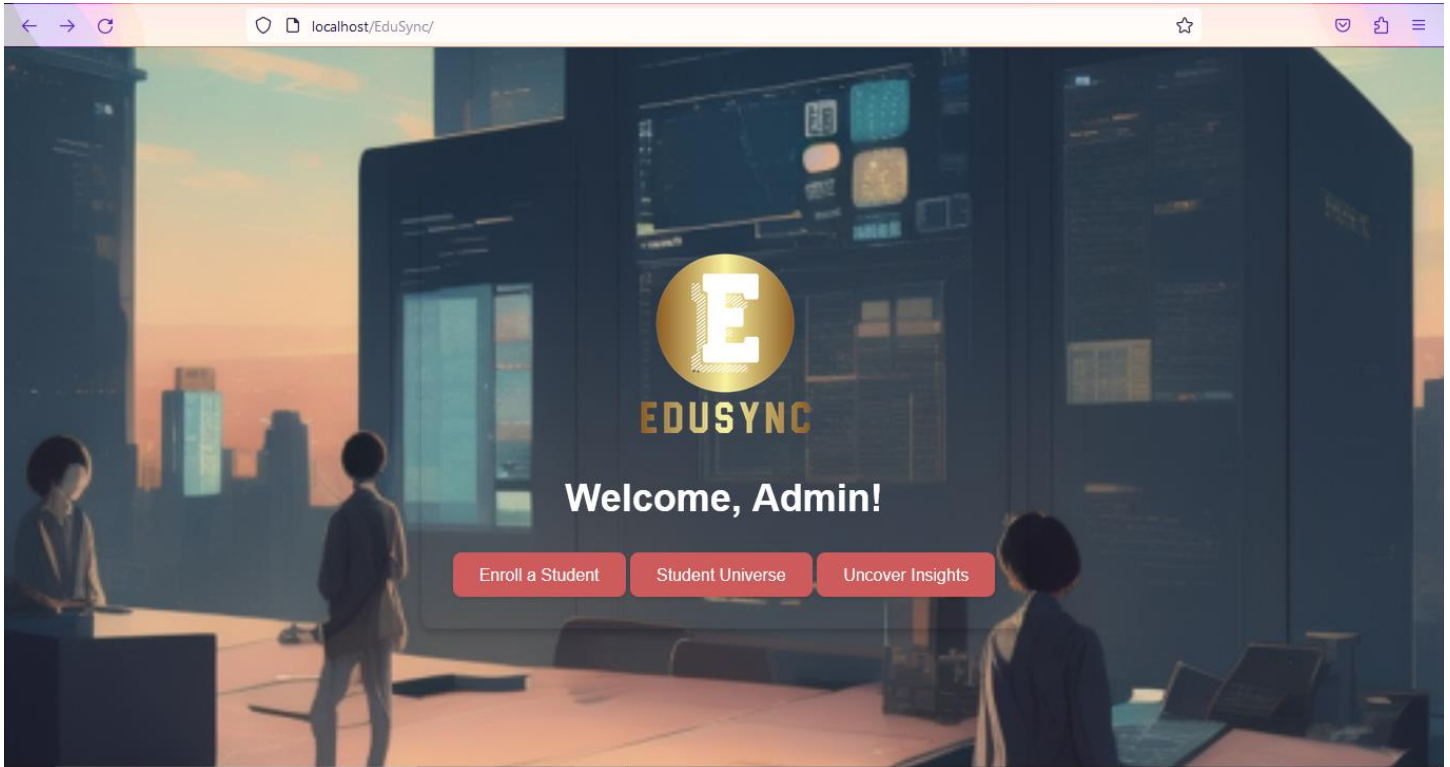
EduSync includes role-based access control, allowing administrators to manage user roles and permissions. Admins can create, modify, or delete user accounts, ensuring that only authorized individuals can access and manipulate student data. This granular control enhances data security and privacy.

Technology Stack

EduSync leverages a diverse technology stack to provide a seamless and feature-rich experience. Key technologies and tools used in the development of EduSync include:

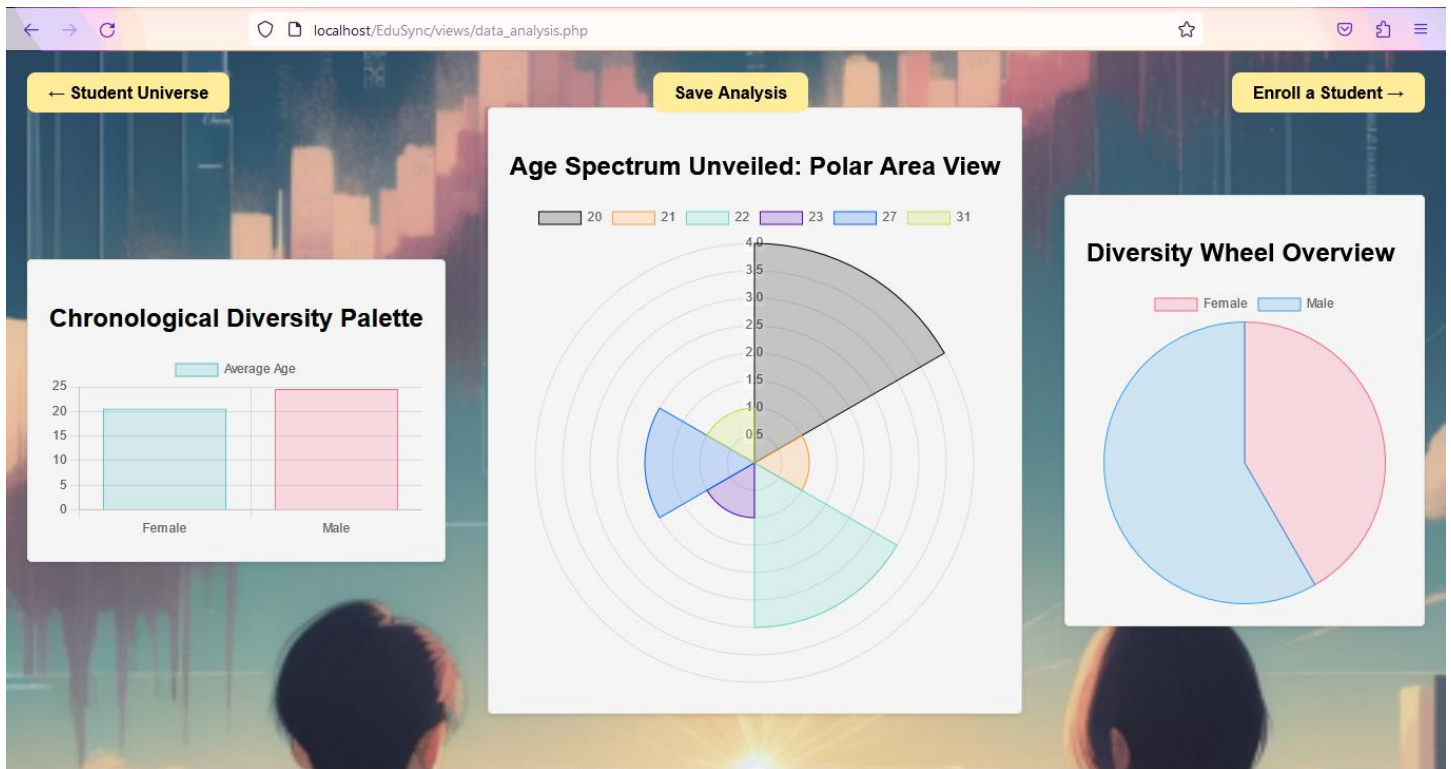
- HTML for structuring web pages.
- PHP for server-side scripting and database interactions.
- CSS for styling and layout.
- JavaScript for client-side interactivity and graph plotting using Chart.js.
- MySQL database for data storage and retrieval.
- DataTables for data presentation and manipulation.

Screenshots and Visualization



This screenshot shows the 'Student Universe' registration form. The browser address bar indicates the URL is `localhost/EduSync/views/student_form.php`. The form is titled 'EduSync Registration' and is set against a background of a futuristic city with a large robot. Navigation links '← Student Universe' and 'Uncover Insights →' are at the top. The form fields are as follows:

- First Name:
- Last Name:
- Age: (with a small age selection icon)
- Gender: (with a dropdown arrow)
- Student ID: (with a small ID selection icon)
- Email:



Conclusion

EduSync is designed to simplify the complex task of managing student data. It employs a secure MVC architecture, implements robust authentication and data validation, and provides a range of data management and analytical tools. With EduSync, educational institutions can harness the power of data to enhance decision-making and student outcomes.

CHAPTER 3

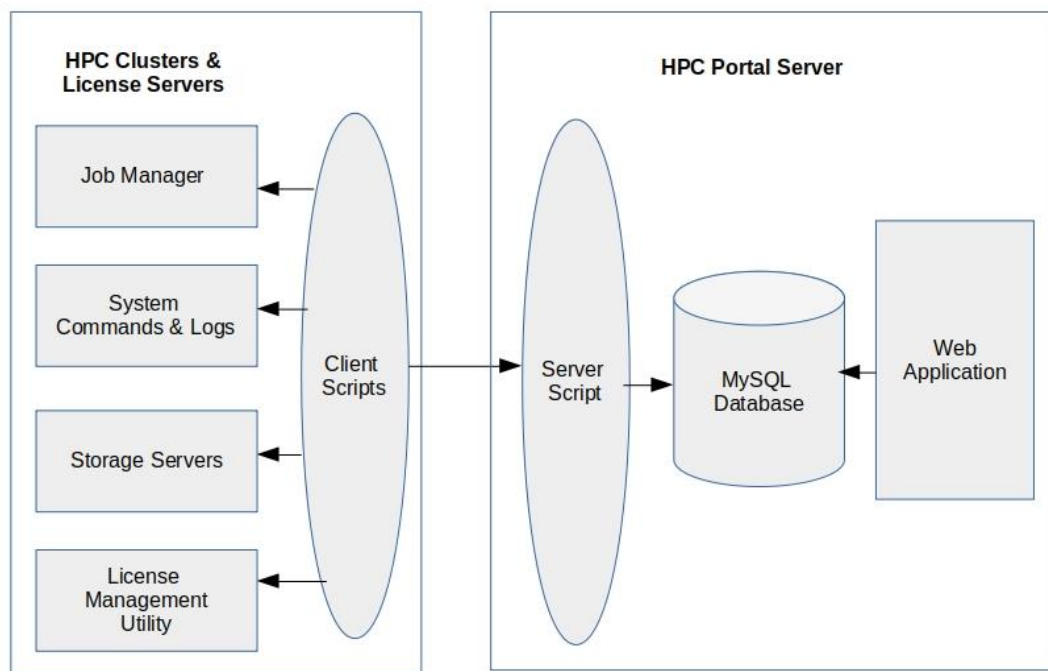
HPCS PORTAL

User Privileges and Access Control

The HPC portal serves various user roles, each with different privileges. These roles include HPC users, shift operators, shift-in-charge, and HPC administrators. Access control is integrated with a centralized LDAP authentication mechanism, ensuring that only authorized personnel can access and interact with the portal. User authentication and authorization are fundamental to maintaining the security and integrity of the HPC system.

Data Collection and Monitoring

The data collection system in the HPC portal is essential for gathering critical information from HPC clusters and license servers. It follows a client-server architecture with client scripts written in Perl. These scripts collect data about cluster status, job queues, user activities, and software licenses. Some scripts run frequently to provide real-time data updates, while others run at longer intervals. The collected data is sent to a server script running on the HPC portal server, which stores it in a MySQL database. This structured storage allows for organized data management and facilitates reporting and historical analysis. In essence, the data collection system ensures that the HPC portal has accurate and up-to-date information to support decision-making and system optimization.



Database Design and Management

The MySQL database plays a pivotal role in storing and managing the data collected from various sources. The database is designed with multiple tables, each dedicated to specific aspects of HPC monitoring. These tables have columns for timestamps, cluster or server names, and data related to the particular aspect they represent. To prevent data clutter, irrelevant information is periodically purged from the database, leaving only essential historical data for further analysis.

hpcmonitor.clusterusage #clustername : varchar(50) #slotstotal : int(11) #slotsdown : int(11) #slotsallocated : int(11) #slotside : int(11) #timeofmon : datetime #UsedPercent : double	hpcmonitor.slurmfinishedjobsinfo @clustername : varchar(30) @timeofmon : timestamp @jobid : int(11) @jobname : varchar(500) @username : varchar(100) @groupname : varchar(100) @queue : varchar(100) @starttime : datetime @endtime : datetime @jobstate : varchar(30) #slots : int(11) #ElapsedTime : double @NodeList : varchar(200) #Nodes : int(11) #cpuusage : double #memusage : double #iusage : double @processname : varchar(500) @jobtype : varchar(500)	hpcmonitor.slurmjobstatus @timeofmon : datetime @clustername : varchar(30) @partitionname : varchar(30) @availstatus : varchar(30) #nodes : int(11) @state : varchar(50) @nodelist : varchar(100) #nodesallocated : int(11) #nodesidle : int(11) #cpuallocated : int(11) #cpuidle : int(11) #cpuother : int(11) #cputotal : int(11)
hpcmonitor.slurmjobs @timeofmon : datetime @clustername : varchar(20) @jobid : int(11) @partitionname : varchar(30) @jobname : varchar(100) @username : varchar(50) @state : varchar(20) @starttime : datetime @elapsedtime : varchar(100) #nodes : int(11) @nodelist : varchar(100) #cpus : int(11) @processname : varchar(500) @jobtype : varchar(500)	hpcmonitor.slurmnodelist @timeofmon : datetime @clustername : varchar(20) @nodename : varchar(40) @partitionname : varchar(40) @state : varchar(40) #cpualloc : int(11) #cpuidle : int(11) #cpuother : int(11) #cputotal : int(11) #loadavg : double	
hpcmonitor.logmessages @cluster : varchar(20) @time : datetime @host : varchar(20) @message : varchar(1500) #msgfreq : int(11) @msgcomment : varchar(150) @admin : varchar(50)	hpcmonitor.loggedinusers @systemname : varchar(30) @timeofmon : datetime @username : varchar(30) @IPaddress : varchar(100) @logintime : varchar(100) @process : varchar(300) @phoneno : varchar(20) @emailid : varchar(50) @userfullname : varchar(200) @usergroup : varchar(100)	hpcmonitor.storageservers @timeofmon : datetime @clustername : varchar(30) @servername : varchar(30) @status : varchar(30)
	hpcmonitor.storagestatus @timeofmon : datetime @clustername : varchar(30) @partitionname : varchar(100) @total : varchar(30) @used : varchar(30) @available : varchar(30) @percent : varchar(30) @mountpoint : varchar(50)	
hpcmonitor.softwarelicenses @licenseserver : varchar(30) @timeofmon : datetime @software : varchar(50) @feature : varchar(50) #total : int(11) #used : int(11)	hpcmonitor.licenseusers @licenseserver : varchar(30) @software : varchar(50) @feature : varchar(50) @username : varchar(50) @machinename : varchar(100) @starttime : varchar(100) #licenses : int(11)	

Web Design and User Interface

The web development for the HPC portal follows a well-organized and maintainable Model-View-Controller (MVC) pattern. This architectural pattern is widely used in software development to separate different aspects of an application for clarity and maintainability.

1. Model-View-Controller (MVC) Pattern:

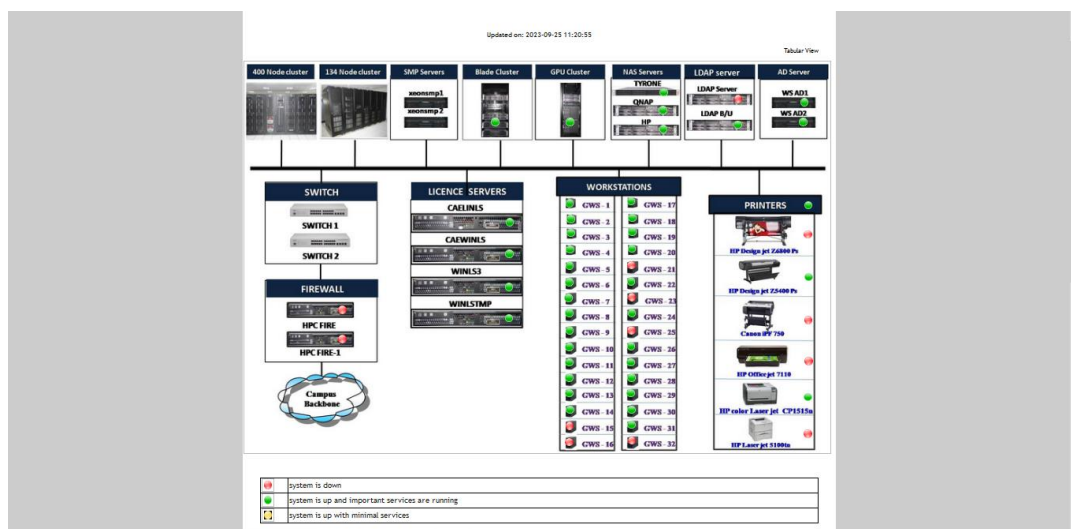
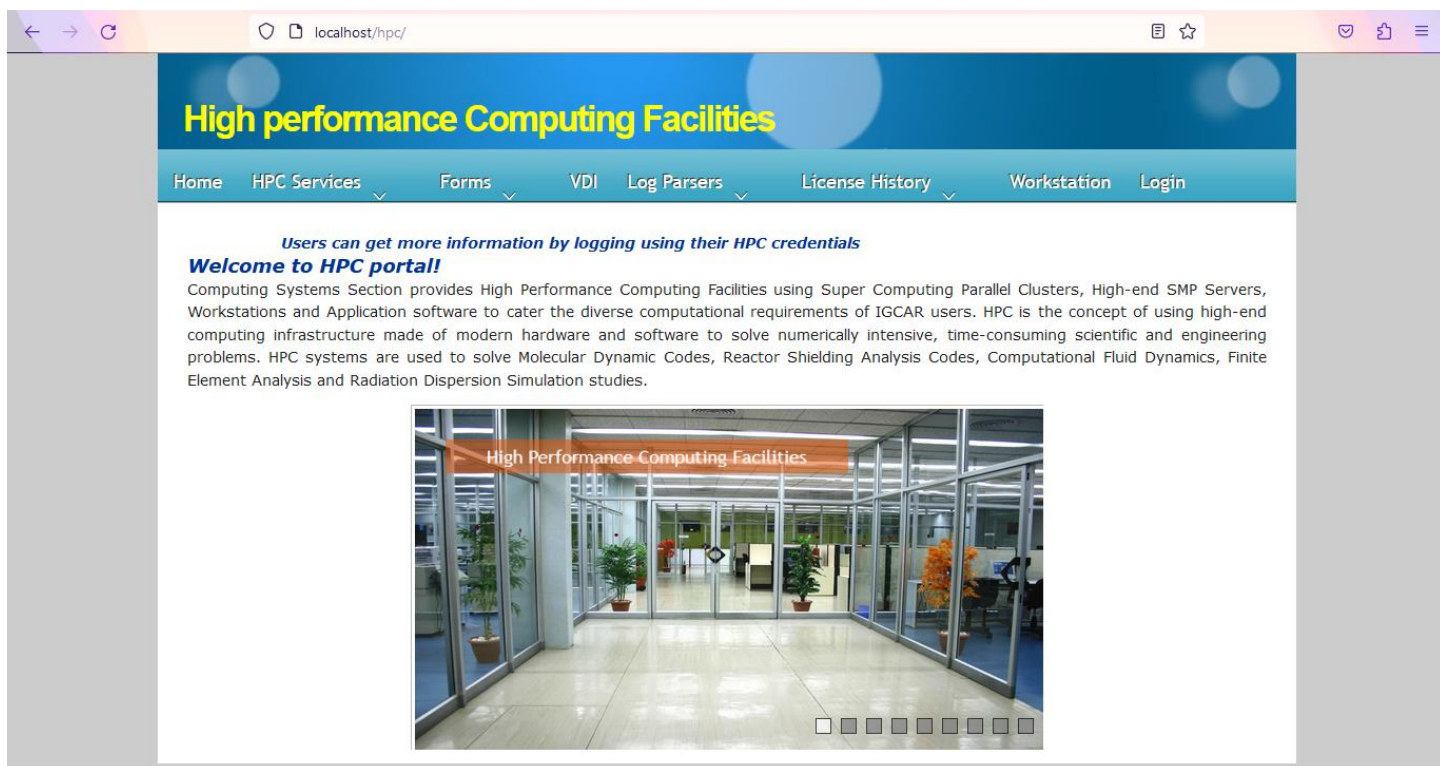
- **Model:** The Model represents the data and the business logic of the application. In the case of the HPC portal, Model classes are responsible for interacting with the MySQL database. This includes retrieving data (e.g., cluster status, job queue details), inserting new data, and updating existing records. Essentially, the Model manages the data and its integrity.
- **View:** The View is responsible for presenting the data to the user in a user-friendly and visually appealing manner. In the HPC portal, the View component takes the data from the Model and formats it for display. This includes rendering data as responsive HTML tables and creating graphs using libraries like 'Highcharts' and 'Datatables.' The View ensures that users can easily interpret and interact with the data.
- **Controller:** The Controller acts as an intermediary between the Model and the View. In this case, the 'SystemMonitor' controller takes on this role. It handles user requests, manages authentication, and facilitates communication between the Model and View components. When a user interacts with the portal, the Controller processes their request, retrieves the necessary data from the Model, and passes it to the View for rendering.

2. Highcharts and Datatables:

- **Highcharts:** Highcharts is a popular JavaScript library for creating interactive and visually appealing charts and graphs. In the context of the HPC portal, Highcharts is used to generate graphical representations of data, making it easier for users to visualize trends and patterns.
 - **Datatables:** Datatables is a JavaScript library that enhances the functionality of HTML tables. It allows for features like sorting, searching, and pagination in HTML tables. In the HPC portal, Datatables is employed to create responsive and user-friendly tables for presenting data.
- By adhering to the MVC pattern and leveraging libraries like Highcharts and Datatables, the HPC portal achieves several advantages:
- **Modularity:** The separation of concerns between the Model, View, and Controller makes the codebase modular and easier to maintain.
 - **Scalability:** New features and enhancements can be added with minimal disruption to existing code.
 - **User-Friendly Interface:** The View component ensures that data is presented in a clear and interactive manner, enhancing the user experience.
 - **Efficient Data Management:** The Model component efficiently manages data interactions with the MySQL database, ensuring data integrity and reliability.

Overall, the use of MVC and these libraries contributes to a robust and user-friendly web development structure in the HPC portal, facilitating effective data presentation and interaction.

Screenshots and Visualization



Conclusion

In conclusion, the development of the HPC portal represents a significant advancement in managing and monitoring high-performance computing resources at IGCAR. The portal's features, architecture, and user access controls have been detailed in this report. By providing real-time data, historical records, and user-friendly visualizations, the portal contributes to efficient resource utilization, improved decision-making, and enhanced user experience. This report serves as a comprehensive overview of the portal's capabilities and its importance within the organization.

CHAPTER 4

DOMAIN OF WORK

It's evident that Abaqus, ANSYS, and COMSOL Multiphysics are all powerful simulation software packages, each with its own unique features and capabilities. Let's delve a bit deeper into each of these software solutions:

Abaqus Unified FEA

Abaqus Overview:

- Abaqus is a powerful Finite Element Analysis (FEA) software.
- It is extensively utilized in engineering and manufacturing industries.
- Abaqus offers a comprehensive suite of simulation tools for solving a wide range of engineering problems.

Key Features:

- **Multidisciplinary Simulations:** Abaqus provides a unified platform for simulating multiple engineering disciplines. Engineers can perform structural analysis, dynamics, thermal analysis, and more within a single environment.
- **Complex Simulations:** Abaqus excels at handling complex simulations, including dynamic vibrations, impact/crash simulations, and nonlinear static analysis. This capability is crucial for analyzing real-world engineering problems.
- **Common Model Data Structure:** Abaqus employs a common model data structure. This simplifies the process of considering full vehicle loads and various coupled phenomena. It ensures consistency and efficiency in modeling complex systems.
- **Integration:** Abaqus offers integrated solver technology. This integration makes it easier for engineers to analyze different aspects of a system in a cohesive and efficient manner. It facilitates the examination of multifaceted engineering scenarios.

Abaqus is valued for its versatility and ability to simulate intricate engineering challenges, making it a valuable tool in various industries for understanding and optimizing designs and systems.

ANSYS

ANSYS Overview:

- ANSYS is a versatile simulation software widely used in engineering.
- It covers a broad range of physical disciplines, making it suitable for diverse engineering applications.
- ANSYS offers powerful tools for simulating and analyzing complex systems.

Key Aspects:

- **Multiphysics Simulations:** ANSYS excels at simulating interactions across different physics disciplines. It can handle structural analysis, fluid dynamics, heat transfer, and electromagnetic effects within a single platform. This capability allows engineers to study how multiple physical phenomena affect one another.
- **Virtual Testing:** ANSYS enables engineers to perform virtual testing and analysis of products before physical prototypes are built. This reduces development time and costs significantly by identifying and addressing issues early in the design process.
- **Modular Structure:** ANSYS offers a modular structure, allowing users to choose and utilize specific features and capabilities tailored to their needs. This flexibility ensures that engineers can focus on the aspects of simulation most relevant to their projects.
- **Integration:** ANSYS can be integrated with other engineering software and Computer-Aided Design (CAD) systems. This enhances its versatility and compatibility with existing tools, making it easier for engineers to incorporate ANSYS into their workflow.

ANSYS is valued for its ability to simulate complex physical phenomena, its time and cost-saving benefits through virtual testing, and its adaptability to various engineering domains. It plays a crucial role in improving product design, performance, and reliability across industries.

COMSOL Multiphysics

COMSOL Multiphysics is a platform for modeling and simulating physics-based problems, with several notable features:

- **Multiphysics Modeling:** It excels in solving coupled physics phenomena simultaneously, making it valuable for simulating complex systems.
- **Advanced Numerical Methods:** COMSOL employs advanced numerical analysis tools, adaptive meshing, and error control to provide accurate simulations.
- **Cluster Computing:** It can harness the power of multiprocessor systems and cluster computing for demanding simulations.
- **User-Friendly GUI:** The software offers an intuitive graphical user interface with a Model Builder for easy model creation and access to functionality.

In summary, these software solutions cater to different engineering needs and scenarios. Abaqus is known for its capabilities in structural analysis and complex simulations. ANSYS offers a broad range of physics disciplines and is highly adaptable. COMSOL Multiphysics specializes in multiphysics modeling and advanced numerical methods, making it suitable for researchers and engineers dealing with complex physics-based problems. The choice between them depends on the specific requirements and goals of the engineering project.

CHAPTER 5

LOG PARSING & ANALYSIS

In this chapter, we will explore the Log Parsing and Analysis project, a versatile platform that empowers users of the HPCS Portal to upload raw log files generated by ABAQUS, ANSYS, and COMSOL software license managers. Additionally, users can view detailed analysis reports derived from the parsed log data stored in the database. This project follows the Model-View-Controller (MVC) architecture, ensuring a well-organized codebase. It features robust authentication, data validation, error handling, and interactive graphs powered by Chart.js, making log analysis efficient and insightful.

MVC Architecture

The Log Parsing and Analysis project follows the MVC architectural pattern, separating the application into three primary components:

- Model (M): The Model acts as the data layer and interacts with the database. It is responsible for processing and parsing raw log files uploaded by users. Parsed log data is stored in the database for subsequent analysis. The Model differentiates between ABAQUS, ANSYS, and COMSOL log files due to their distinct parsing techniques.
- View (V): The View handles the presentation layer, rendering the user interface and displaying analysis reports. Interactive graphs powered by Chart.js provide users with visual insights. Datatables are also used for tabular data presentation.
- Controller (C): The Controller serves as the intermediary between the Model and View. It processes user requests, manages data retrieval, and ensures data validation. The Controller coordinates the parsing of uploaded log files, stores parsed data, and facilitates data analysis.

Authentication and Session Management

The project incorporates secure session management using PHP's `session_start()`. This ensures that only authorized users of the HPCS Portal can access log parsing and analysis functionalities. User credentials are verified to provide seamless and secure access to log data.

Data Validation and Error Handling

Robust data validation mechanisms are implemented to ensure the integrity of user-input data. Both client-side and server-side validation processes are employed to guarantee data accuracy and security.

Error handling is a fundamental aspect of the project, allowing for graceful handling of errors that may occur during user interactions, log parsing, and analysis. Users are provided with informative error messages to assist in issue resolution.

Software-Specific Log Parsing and Analysis

ABAQUS

For ABAQUS log files, the project extracts and processes the following information:

- Date
- Time
- Software
- Status
- Feature
- UserMachine
- Tokens

Analysis functionalities for ABAQUS include:

- Day-wise FEATURE TOKEN USAGE
- DENIED Records
- Day-wise DENIAL COUNT
- Cumulative Feature Tracker (Feature usage duration)
- Feature Flux: Charting Daily Engagement (Day-wise Feature usage duration)

Datatables are employed to present analysis results in a structured manner.

ANSYS

ANSYS log files are processed to retrieve information such as:

- Time
- Software
- Status
- Feature
- UserMachine
- UniqueID
- Date

Analysis functionalities for ANSYS align with those of ABAQUS, with the exception of Day-wise FEATURE TOKEN USAGE. Datatables are utilized to present the analysis reports.

COMSOL

COMSOL log files contain the following data:

- Date
- Time
- Software
- Status
- Feature
- UserMachine

Analysis functionalities for COMSOL, like ANSYS, exclude Day-wise FEATURE TOKEN USAGE. Instead, the count of usage is displayed. Datatables are employed for data presentation.

Custom Time Interval Analysis

By default, the system displays analysis reports for the entire duration of data. However, users have the flexibility to specify custom time intervals for analysis. This feature empowers users to focus on specific periods of interest and gain insights into trends, usage patterns, and resource allocation during those periods.

Technology Stack

The Log Parsing and Analysis project utilizes a versatile technology stack to deliver a seamless and feature-rich experience:

- HTML: Structures web pages and creates the user interface.
- PHP: Powers server-side scripting, database interactions, raw data parsing and data retrieval.
- CSS: Provides styling and layout for an aesthetically pleasing interface.
- JavaScript: Enhances client-side interactivity and drives Chart.js for dynamic graph generation.
- MySQL: Stores parsed log data efficiently and securely.
- Chart.js: Facilitates the creation of interactive utilization graphs for insightful analysis.
- Datatables: Presents analysis results in a structured tabular format.

Log Parsing Methodology

Log Parsing is a crucial aspect of the Log Parsing and Analysis project. It involves the extraction and processing of raw log files from ABAQUS, ANSYS, and COMSOL software license managers. Each software's log file follows a specific format, necessitating distinct parsing techniques.

ABAQUS Log Parsing

The ABAQUS log file parsing method begins by splitting the raw log content into individual lines. Here is a step-by-step explanation of the parsing methodology:

- Initialization: Initialize an empty data array and a variable to store the current date.
- Line-by-Line Processing: Iterate through each line of the log content.
- Pattern Matching: Use regular expressions to match and extract relevant information from each line. The regular expression pattern used in ABAQUS parsing identifies key components, such as date, time, software, status, feature, user machine, and token counts.
- Timestamp Conversion: Convert the extracted timestamp to a DateTime object for standardized formatting.
- Data Storage: If a valid timestamp is found, create a data entry in the data array with the extracted information, including date, time, software, status, feature, user machine, and token counts.

- Date Detection: Detect lines containing the "TIMESTAMP" keyword to determine the current date.
- Database Insertion: If a current date is identified, insert the parsed data into the database using the `insertDataToDatabase` method.
- Return Data: Return the parsed data array for further processing and analysis.

ANSYS Log Parsing

The ANSYS log file parsing method follows a similar structure to ABAQUS parsing. Here's a breakdown of the parsing methodology for ANSYS logs:

- Initialization: Initialize an empty data array and a variable to store the current date.
- Line-by-Line Processing: Iterate through each line of the log content.
- Pattern Matching: Utilize regular expressions to match and extract relevant information from each line. The ANSYS parsing pattern identifies elements such as time, software, status, feature, user machine, and a unique ID.
- Timestamp Conversion: Convert the extracted timestamp to a DateTime object for standardized formatting.
- Data Storage: If a valid timestamp is found, create a data entry in the data array with the extracted information, including date, time, software, status, feature, user machine, and unique ID.
- Date Detection: Detect lines containing the "TIMESTAMP" keyword to determine the current date.
- Database Insertion: If a current date is identified, insert the parsed data into the database table for ANSYS.
- Error Handling: Handle any database insertion errors and display error messages if necessary.

COMSOL Log Parsing

COMSOL log file parsing is similar in structure to ANSYS parsing, with some variations in data format. Here's an overview of the parsing methodology for COMSOL logs:

- Initialization: Initialize an empty data array and a variable to store the current date.
- Line-by-Line Processing: Iterate through each line of the log content.
- Pattern Matching: Apply regular expressions to match and extract relevant information from each line. The COMSOL parsing pattern identifies elements like date, time, software, status, feature, and user machine.
- Timestamp Conversion: Convert the extracted timestamp to a DateTime object for standardized formatting.

- Data Storage: If a valid timestamp is found, create a data entry in the data array with the extracted information, including date, time, software, status, feature, and user machine.
- Date Detection: Detect lines containing the "TIMESTAMP" keyword to determine the current date.
- Database Insertion: If a current date is identified, insert the parsed data into the database table for COMSOL.
- Error Handling: Handle any database insertion errors and display error messages if necessary.

Unified Database Insertion

Throughout the parsing process, once a valid timestamp and data are extracted, the parsed data is inserted into the respective database table for the software (ABAQUS, ANSYS, or COMSOL).

By implementing these parsing methodologies, the Log Parsing and Analysis project ensures that raw log files are processed accurately, and meaningful data is available for detailed analysis and insights.

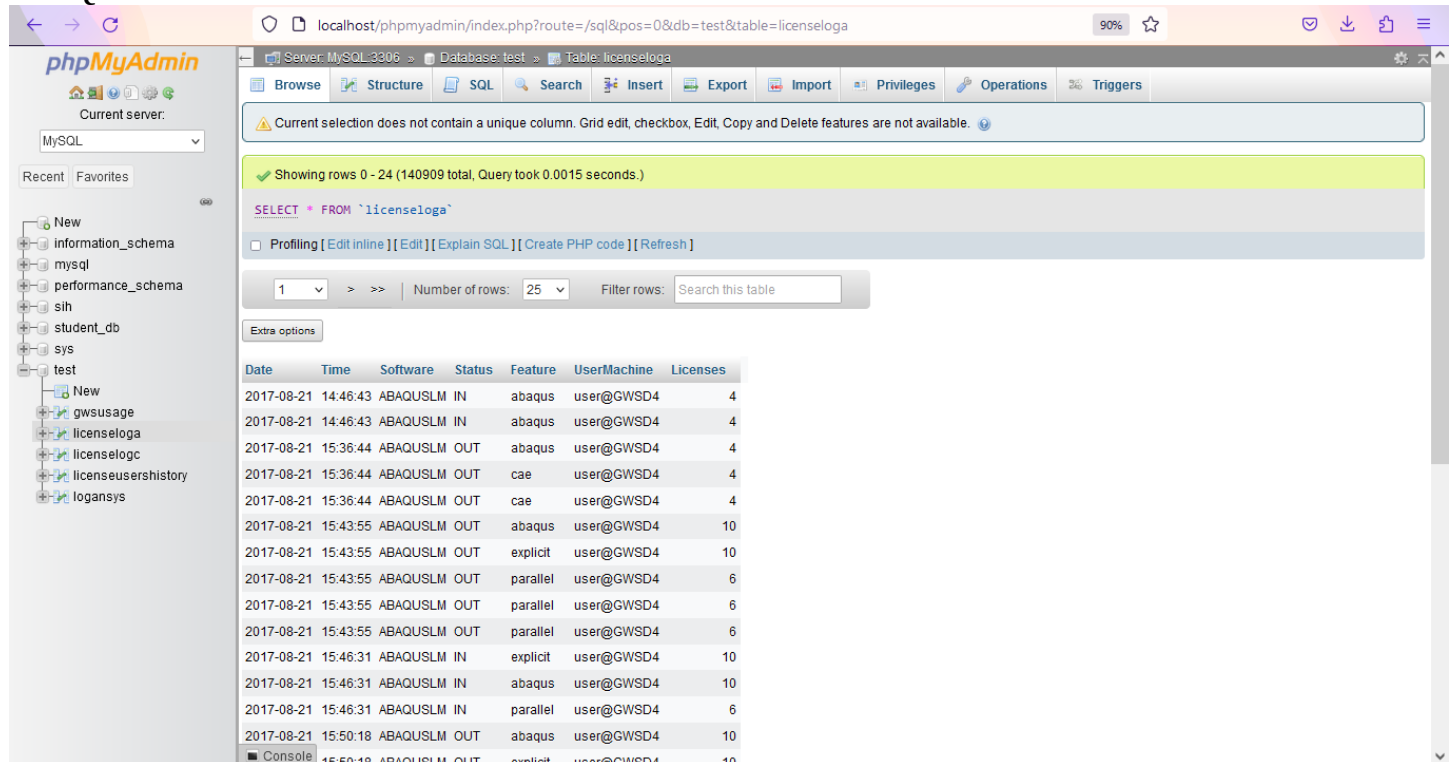
Screenshots and Visualization

Raw Log File:



```
license.log - Notepad
File Edit Format View Help
12:49:27 (ABAQUSLM) EXTERNAL FILTERS are Off12:49:27 (lmgrd) ABAQUSLM using TCP-port 4595012:49:33 (ABAQUSLM) TCP_NODELAY NOT enabled12:49:46 (ABAQUSLM) OUT: "abaqus" a
167/kulbir/unknown/node352.igcar.gov.in13:38:20 (ABAQUSLM) DENIED: "abaqus" kulbir@node350.igcar.gov.in (25 licenses) (Licensed number of users already reached. (-4,34
@node350.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))14:24:50 (ABAQUSLM) DENIED: "standard" kulbir@node350.igcar.gov.in (25 licenses) (Us
car.gov.in (25 licenses) 15:08:27 (ABAQUSLM) IN: "parallel" kulbir@node352.igcar.gov.in (48 licenses) 15:08:50 (ABAQUSLM) OUT: "parallel" kulbir@node350.igcar.gov.in
Licensed number of users already reached. (-4,342))16:36:06 (ABAQUSLM) DENIED: "cae" prince@GWS-9 (Licensed number of users already reached. (-4,342))16:36:06 (ABAQUSL
eady reached. (-4,342:104 "Connection reset by peer"))16:36:11 (ABAQUSLM) DENIED: "cae" prince@GWS-9 (Licensed number of users already reached. (-4,342:104 "Connection
))16:59:51 (ABAQUSLM) DENIED: "abaqus" askmanson@node353.igcar.gov.in (25 licenses) (Licensed number of users already reached. (-4,342))16:59:51 (ABAQUSLM) DENIED: "ex
gcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))17:46:22 (ABAQUSLM) DENIED: "explicit" askmanson@node353.igcar.gov.in (25 licenses) (Users are
qus" askmanson@node353.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))18:48:22 (ABAQUSLM) DENIED: "explicit" askmanson@node353.igcar.gov.in
: "abaqus" kulbir@node350.igcar.gov.in (30 licenses) 19:23:41 (ABAQUSLM) OUT: "abaqus" askmanson@node353.igcar.gov.in (25 licenses) 19:23:41 (ABAQUSLM) IN: "parallel"
"parallel" prince@GWS-9 (4 licenses) 21:11:39 (ABAQUSLM) 1682696515/0/6.19-1/0/64/8,62/8/sta_par/55124/prince/unknown/GWS-921:18:20 (ABAQUSLM) 1682696917/1/6.19-1/4/ca
:49:27 (lmgrd) TIMESTAMP 4/30/2023 1:19:33 (ABAQUSLM) TIMESTAMP 4/30/2023 6:49:27 (lmgrd) TIMESTAMP 4/30/2023 7:24:34 (ABAQUSLM) TIMESTAMP 4/30/2023 8:37:35 (ABAQUSLM)
ses) 21:24:44 (ABAQUSLM) 1682870148/0/6.19-1/0/64/4,29/4/cae/53080/prince/unknown/GWS-921:27:57 (ABAQUSLM) 1682870341/1/6.19-1/4/cae/53080/prince/unknown/GWS-921:27:57
:57:00 (ABAQUSLM) IN: "parallel" kulbir@node355.igcar.gov.in (72 licenses) 9:57:15 (ABAQUSLM) OUT: "abaqus" kulbir@node355.igcar.gov.in (30 licenses) 9:57:15 (ABAQUSL
3/1/6.17-1/30/sta_par/5227/kulbir/unknown/node355.igcar.gov.in12:29:38 (ABAQUSLM) IN: "standard" kulbir@node355.igcar.gov.in (30 licenses) 12:29:38 (ABAQUSLM) IN: "aba
in@node355.igcar.gov.in (30 licenses) 15:10:52 (ABAQUSLM) OUT: "standard" kulbir@node355.igcar.gov.in (30 licenses) 15:12:09 (ABAQUSLM) 1682937584/1/6.19-1/25/exp_par
"abaqus" askmanson@node352.igcar.gov.in (25 licenses) 17:19:57 (ABAQUSLM) OUT: "explicit" askmanson@node352.igcar.gov.in (25 licenses) 17:34:57 (ABAQUSLM) DENIED: "ab
d for this feature. (-24,332))18:21:27 (ABAQUSLM) DENIED: "explicit" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))18:36:5
kmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))19:23:27 (ABAQUSLM) DENIED: "explicit" askmanson@node352.igcar.gov.in (25 lic
his feature. (-24,332))20:25:27 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))20:25:27 (ABAQUS
(Users are queued for this feature. (-24,332))21:27:27 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-2
r.gov.in (25 licenses) (Users are queued for this feature. (-24,332))22:29:27 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queu
" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))23:31:28 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 l
QUSLM) DENIED: "explicit" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332)) 0:33:28 (ABAQUSLM) DENIED: "abaqus" askmanson@nod
(Users are queued for this feature. (-24,332)) 1:19:58 (ABAQUSLM) DENIED: "explicit" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (
DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332)) 2:21:58 (ABAQUSLM) DENIED: "explicit" askmanson@node352.ig
2)) 3:23:58 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332)) 3:23:58 (ABAQUSLM) DENIED: "explici
or this feature. (-24,332)) 4:25:58 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332)) 4:25:58 (AE
ses) (Users are queued for this feature. (-24,332)) 5:27:58 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature
.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332)) 6:29:58 (ABAQUSLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are
2)) 7:15:58 (ABAQUSLM) DENIED: "explicit" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332)) 7:31:28 (ABAQUSLM) DENIED: "abaqu
v.in (25 licenses) (Users are queued for this feature. (-24,332)) 8:17:58 (ABAQUSLM) DENIED: "explicit" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued
kmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332)) 9:19:58 (ABAQUSLM) DENIED: "explicit" askmanson@node352.igcar.gov.in (25 lic
SLM) DENIED: "abaqus" askmanson@node352.igcar.gov.in (25 licenses) (Users are queued for this feature. (-24,332))10:21:58 (ABAQUSLM) DENIED: "explicit" askmanson@node3
3008254/0/6.19-1/0/64/25,59/25/exp_par/32482/askmanson/unknown/node352.igcar.gov.in11:17:31 (ABAQUSLM) OUT: "abaqus" kulbir@GWS-24 (4 licenses) 11:17:31 (ABAQUSLM) OUT
already reached. (-4,342:104 "Connection reset by peer"))11:47:54 (ABAQUSLM) DENIED: "abaqus" kulbir@GWS-17 (4 licenses) (Licensed number of users already reached. (-4
"Connection reset by peer"))11:51:57 (ABAQUSLM) DENIED: "cae" kulbir@GWS-17 (Licensed number of users already reached. (-4,342:104 "Connection reset by peer"))11:52:16
WS-17 (4 licenses) 11:57:45 (ABAQUSLM) 1683008919/0/6.19-1/0/64/4,63/4/cae/9492/kulbir/unknown/GWS-1712:49:27 (lmgrd) TIMESTAMP 5/2/202313:40:57 (ABAQUSLM) DENIED: "ab
eached. (-4,342:104 "Connection reset by peer"))13:42:14 (ABAQUSLM) DENIED: "cae" jagruti@GWS-2 (Licensed number of users already reached. (-4,342:104 "Connection rese
```


Parsed Data ABAQUS



Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 24 (140909 total, Query took 0.0015 seconds.)

```
SELECT * FROM `licenseloga`
```

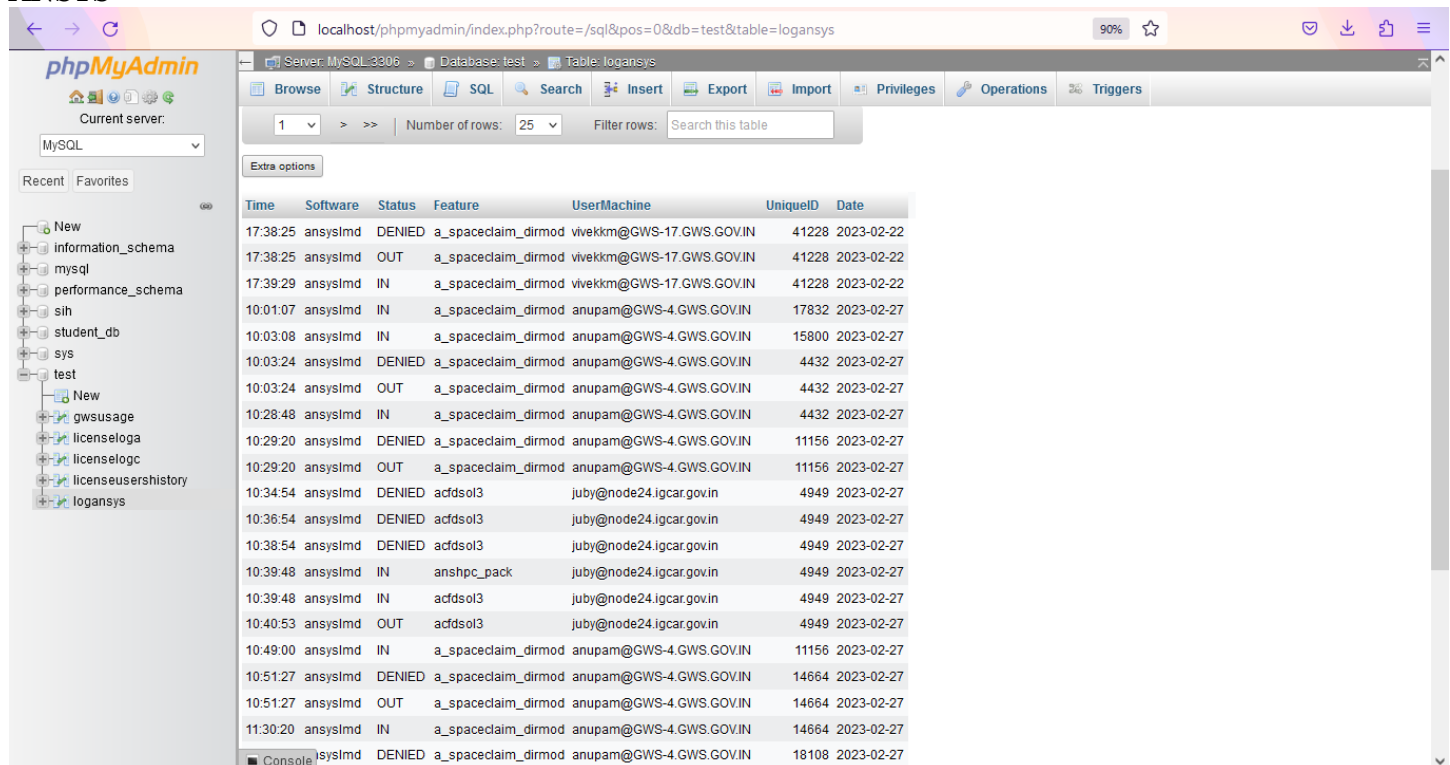
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1 > >> Number of rows: 25 Filter rows: Search this table

Extra options

Date	Time	Software	Status	Feature	UserMachine	Licenses
2017-08-21	14:46:43	ABAQUSLM	IN	abaqus	user@GWSD4	4
2017-08-21	14:46:43	ABAQUSLM	IN	abaqus	user@GWSD4	4
2017-08-21	15:36:44	ABAQUSLM	OUT	abaqus	user@GWSD4	4
2017-08-21	15:36:44	ABAQUSLM	OUT	cae	user@GWSD4	4
2017-08-21	15:36:44	ABAQUSLM	OUT	cae	user@GWSD4	4
2017-08-21	15:43:55	ABAQUSLM	OUT	abaqus	user@GWSD4	10
2017-08-21	15:43:55	ABAQUSLM	OUT	explicit	user@GWSD4	10
2017-08-21	15:43:55	ABAQUSLM	OUT	parallel	user@GWSD4	6
2017-08-21	15:43:55	ABAQUSLM	OUT	parallel	user@GWSD4	6
2017-08-21	15:43:55	ABAQUSLM	OUT	parallel	user@GWSD4	6
2017-08-21	15:46:31	ABAQUSLM	IN	explicit	user@GWSD4	10
2017-08-21	15:46:31	ABAQUSLM	IN	abaqus	user@GWSD4	10
2017-08-21	15:46:31	ABAQUSLM	IN	parallel	user@GWSD4	6
2017-08-21	15:50:18	ABAQUSLM	OUT	abaqus	user@GWSD4	10
2017-08-21	15:50:18	ABAQUSLM	OUT	explicit	user@GWSD4	10

ANSYS



Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 24 (25 total, Query took 0.0015 seconds.)

```
SELECT * FROM `logansys`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1 > >> Number of rows: 25 Filter rows: Search this table

Extra options

Time	Software	Status	Feature	UserMachine	UniqueID	Date
17:38:25	ansyslmd	DENIED	a_spacedaim_dirmod	vivekkm@GWS-17.GWS.GOV.IN	41228	2023-02-22
17:38:25	ansyslmd	OUT	a_spacedaim_dirmod	vivekkm@GWS-17.GWS.GOV.IN	41228	2023-02-22
17:39:29	ansyslmd	IN	a_spacedaim_dirmod	vivekkm@GWS-17.GWS.GOV.IN	41228	2023-02-22
10:01:07	ansyslmd	IN	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	17832	2023-02-27
10:03:08	ansyslmd	IN	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	15800	2023-02-27
10:03:24	ansyslmd	DENIED	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	4432	2023-02-27
10:03:24	ansyslmd	OUT	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	4432	2023-02-27
10:28:48	ansyslmd	IN	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	4432	2023-02-27
10:29:20	ansyslmd	DENIED	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	11156	2023-02-27
10:29:20	ansyslmd	OUT	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	11156	2023-02-27
10:34:54	ansyslmd	DENIED	acfdsol3	juby@node24.igcar.gov.in	4949	2023-02-27
10:36:54	ansyslmd	DENIED	acfdsol3	juby@node24.igcar.gov.in	4949	2023-02-27
10:38:54	ansyslmd	DENIED	acfdsol3	juby@node24.igcar.gov.in	4949	2023-02-27
10:39:48	ansyslmd	IN	anshpc_pack	juby@node24.igcar.gov.in	4949	2023-02-27
10:39:48	ansyslmd	IN	acfdsol3	juby@node24.igcar.gov.in	4949	2023-02-27
10:40:53	ansyslmd	OUT	acfdsol3	juby@node24.igcar.gov.in	4949	2023-02-27
10:49:00	ansyslmd	IN	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	11156	2023-02-27
10:51:27	ansyslmd	DENIED	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	14664	2023-02-27
10:51:27	ansyslmd	OUT	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	14664	2023-02-27
11:30:20	ansyslmd	IN	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	14664	2023-02-27
11:30:20	ansyslmd	DENIED	a_spacedaim_dirmod	anupam@GWS-4.GWS.GOV.IN	18108	2023-02-27

COMSOL

phpMyAdmin

Current server: MySQL

Recent Favorites

New

information_schema

mysql

performance_schema

sih

student_db

sys

test

New

gwsusage

licenseloga

licenseelogc

licenseusershistory

logansys

localhost/phpmyadmin/index.php?route=/sql&pos=0&db=test&table=licenseelogc

Browse

Structure

SQL

Search

Insert

Export

Import

Privileges

Operations

Triggers

1

>

>>

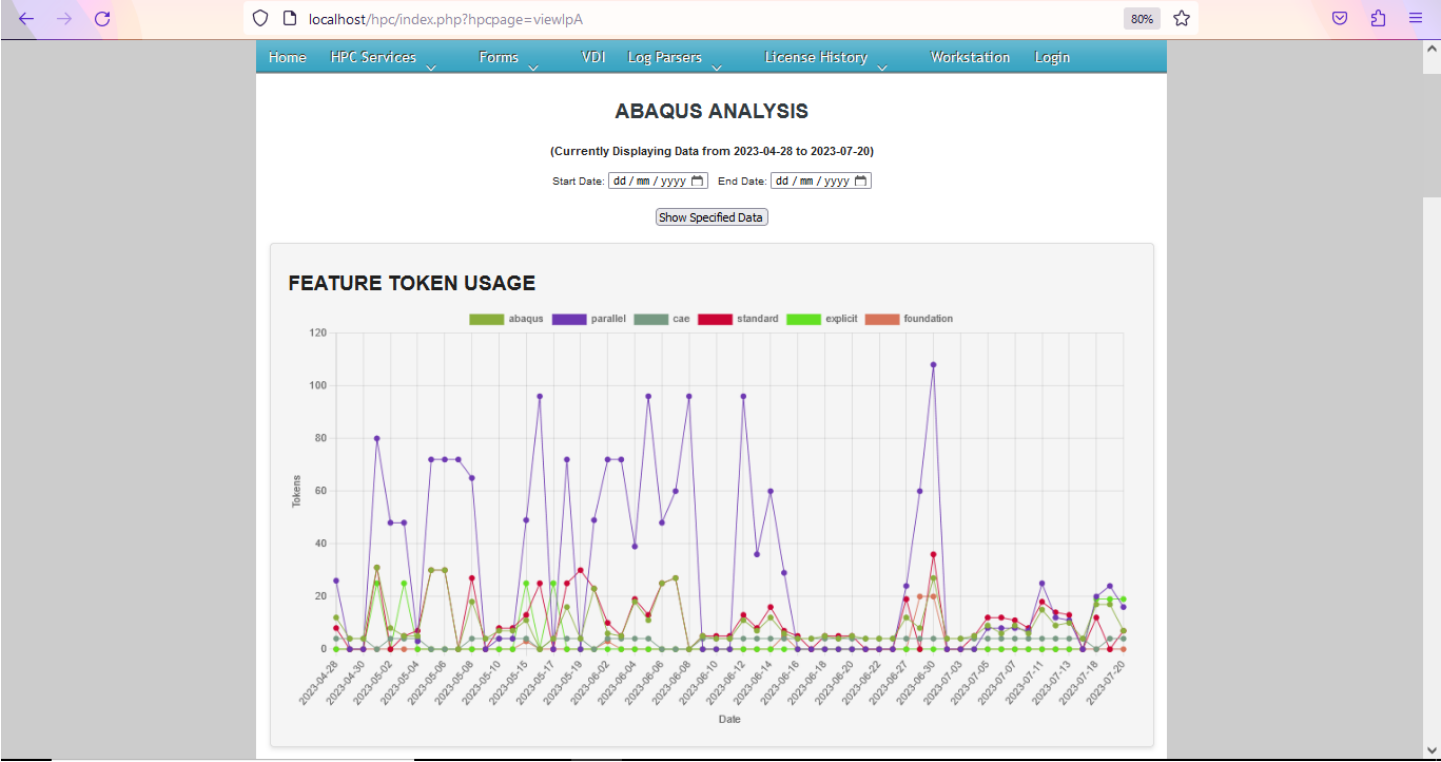
Number of rows: 25

Filter rows: Search this table

Extra options

Date	Time	Software	Status	Feature	UserMachine
2023-07-21	15:12:42	LMCOMSOL	OUT	CADIMPORT	avijit@GWS-1
2023-07-22	15:12:42	LMCOMSOL	OUT	CADIMPORT	avijit@GWS-1
2023-07-23	15:12:42	LMCOMSOL	OUT	CADIMPORT	avijit@GWS-1
2023-07-24	15:12:42	LMCOMSOL	OUT	CADIMPORT	avijit@GWS-1
2023-07-24	10:38:29	LMCOMSOL	IN	SERIAL	sabarish@GWS-12
2023-07-24	10:38:29	LMCOMSOL	IN	COMSOL	sabarish@GWS-12
2023-07-24	10:38:29	LMCOMSOL	IN	COMSOLGUI	sabarish@GWS-12
2023-07-24	10:38:29	LMCOMSOL	IN	CFD	sabarish@GWS-12
2023-07-24	10:38:29	LMCOMSOL	IN	COMSOLUSER	sabarish@GWS-12
2023-07-24	10:38:30	LMCOMSOL	IN	COMSOL	sabarish@GWS-12
2023-07-24	10:38:30	LMCOMSOL	IN	COMSOLGUI	sabarish@GWS-12
2023-07-24	10:38:30	LMCOMSOL	IN	CFD	sabarish@GWS-12
2023-07-24	10:38:30	LMCOMSOL	IN	COMSOLUSER	sabarish@GWS-12
2023-07-24	11:39:41	LMCOMSOL	OUT	CADIMPORTUSER	avijit@GWS-1
2023-07-24	11:39:41	LMCOMSOL	OUT	COMSOLGUI	avijit@GWS-1
2023-07-24	11:39:42	LMCOMSOL	OUT	CADIMPORT	avijit@GWS-1
2023-07-24	11:39:42	LMCOMSOL	OUT	ACDC	avijit@GWS-1
2023-07-24	11:39:42	LMCOMSOL	OUT	COMSOLUSER	avijit@GWS-1
2023-07-24	11:39:42	LMCOMSOL	OUT	CFD	avijit@GWS-1
2023-07-24	11:39:42	LMCOMSOL	OUT	CADIMPORTUSER	avijit@GWS-1
2023-07-24	11:39:42	LMCOMSOL	OUT	CFD	avijit@GWS-1

Console



localhost/hpc/index.php?hpcpage=viewIpA

DENIED Records

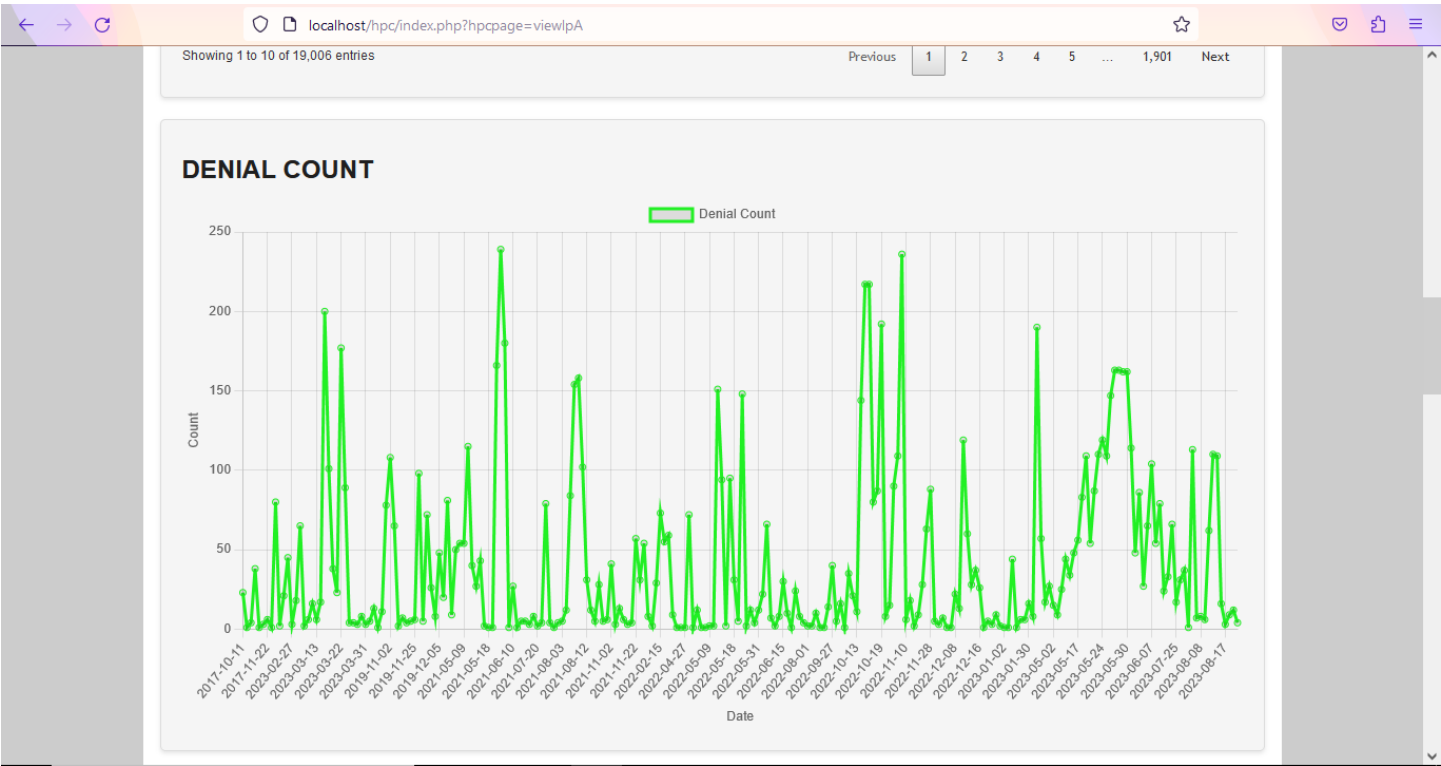
Show 10 entries

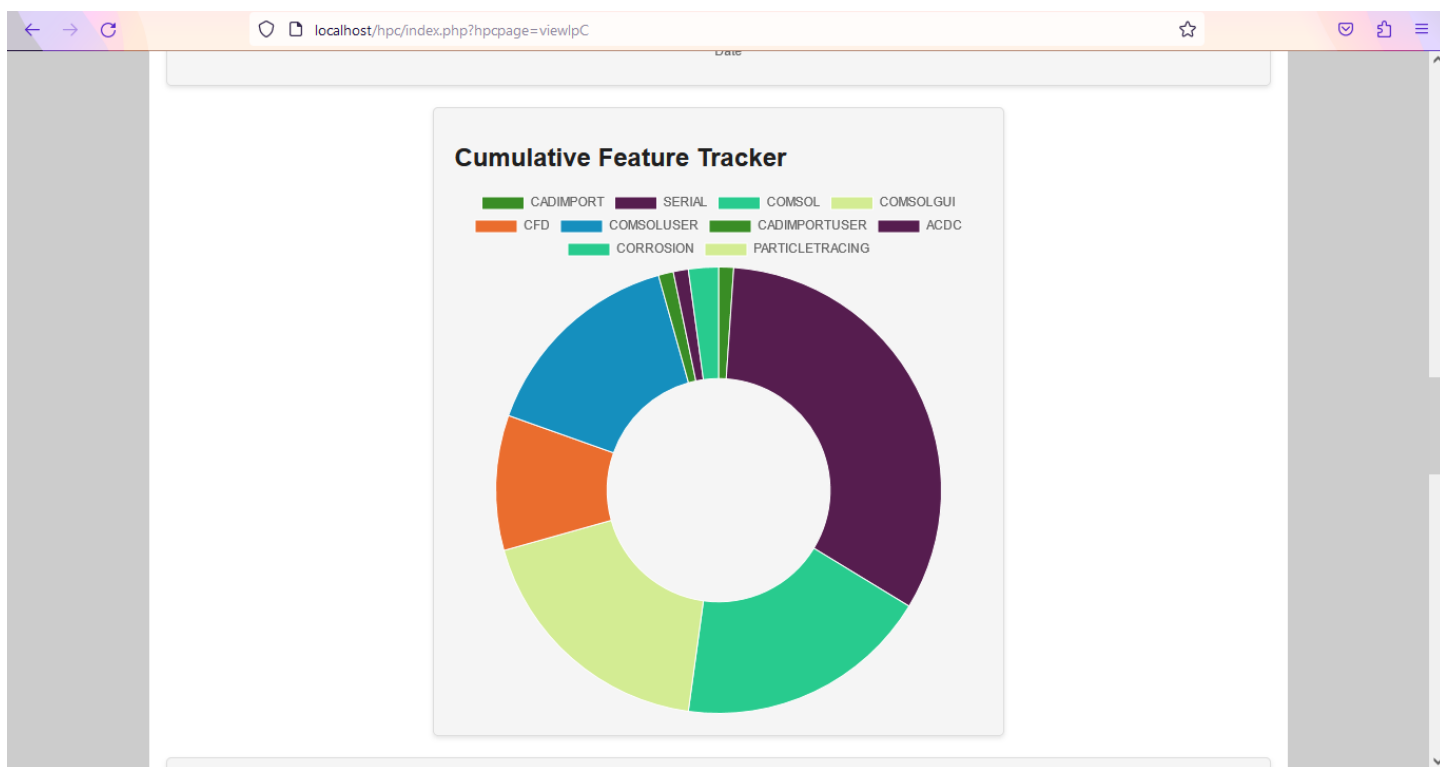
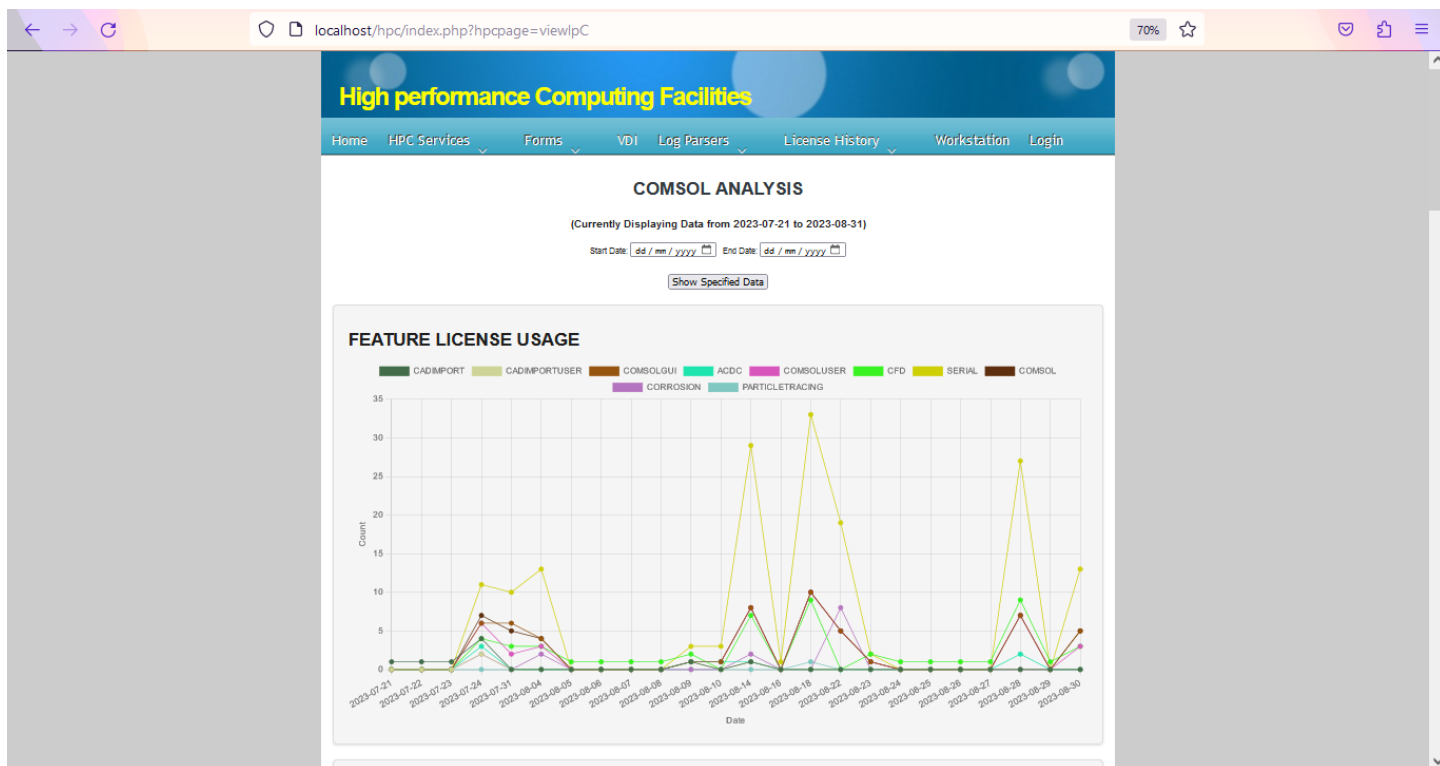
Search:

DATE	TIME	FEATURE	USERMACHINE	TOKENS
2017-10-11	14:44:38	abacus	nikil@GWSD-2-PC	19
2017-10-11	14:44:38	standard	nikil@GWSD-2-PC	19
2017-10-11	14:50:11	abacus	admin@xsftn8	3
2017-10-11	14:50:11	foundation	admin@xsftn8	3
2017-10-11	14:52:56	abacus	nikil@GWSD-2-PC	14
2017-10-11	14:52:56	standard	nikil@GWSD-2-PC	14
2017-10-11	14:58:06	abacus	admin@xsftn8	4
2017-10-11	14:58:06	abacus	nikil@GWSD-2-PC	16
2017-10-11	14:58:06	standard	nikil@GWSD-2-PC	16
2017-10-11	15:01:41	abacus	admin@xsftn8	4

Showing 1 to 10 of 19,006 entries

Previous12345...1,901Next





← → ↺

localhost/hpc/index.php?hpcpage=viewlpA

90% ☆

🔍 📄 ☰

2023-04-02

2023-05-02

2023-06-02

2023-07-02

2023-08-02

2023-09-02

2023-10-02

2023-11-02

2023-12-02

2024-01-02

2024-02-02

2024-03-02

2024-04-02

2024-05-02

2024-06-02

2024-07-02

2024-08-02

2024-09-02

2024-10-02

2024-11-02

2024-12-02

Date

Engagement Dynamics: Daily Feature Activity

Show 10 entries

Search:

DATE	EXPLICIT	ABAQUS	STANDARD	PARALLEL	CAE	FOUNDATION
2023-06-02	00:00:00	09:57:35	09:52:12	02:08:15	02:05:30	00:15:16
2023-05-15	00:00:00	04:13:32	04:02:21	07:38:15	01:34:30	00:03:00
2023-06-30	00:00:00	07:54:27	03:00:45	04:58:41	00:08:50	00:01:41
2023-06-15	00:00:00	08:36:21	08:15:33	07:48:38	05:20:54	00:00:28
2023-06-28	00:00:00	04:01:56	00:00:00	00:00:16	00:23:33	00:00:16
2023-04-28	00:00:00	01:54:39	00:00:44	01:47:03	01:14:55	00:00:00
2023-04-29	00:00:00	02:01:11	00:00:00	00:00:00	01:25:02	00:00:00
2023-04-30	00:00:00	00:03:13	00:00:00	00:00:00	00:03:13	00:00:00
2023-05-01	02:07:48	09:13:20	00:15:01	02:57:31	00:00:00	00:00:00
2023-05-02	00:00:00	07:22:44	00:00:00	00:00:21	06:54:50	00:00:00

Showing 1 to 10 of 84 entries

Previous

1

2

3

4

5

...

9

Next

Conclusion

The Log Parsing and Analysis project offers a powerful and user-friendly platform for analyzing log data generated by ABAQUS, ANSYS, and COMSOL software license managers. With its adherence to the MVC architecture, robust authentication, data validation, error handling, and interactive visualizations, the project enables users to make data-driven decisions, optimize resource allocation, and gain valuable insights from log data.

CHAPTER 6

LICENSE HISTORY ANALYSIS

In this chapter, we delve into the License History Analysis project, a multifaceted system designed to provide insights into license usage history. The project is categorized into three main components: View Activity, Software Interaction Timeline, and Day-to-Day Capability Analysis. Each of these segments offers users a unique perspective on license utilization patterns and allows for data-driven decision-making. The project adheres to the Model-View-Controller (MVC) architecture, ensuring a well-structured and maintainable codebase. It includes robust authentication, data validation, error handling, and dynamic data visualization powered by Chart.js. The technology stack comprises HTML, PHP, CSS, JS, and other essential web technologies.

MVC Architecture

The License History Analysis project follows the MVC architectural pattern, breaking the application into three fundamental components:

- Model (M): The Model serves as the data layer, responsible for interacting with the database. It facilitates data retrieval and processing, ensuring that the system can provide users with meaningful insights. The project's database structure accommodates critical information such as license server details, software types, features, usernames, machine names, usage timestamps, and license counts.
- View (V): The View is responsible for the presentation layer. It constructs the user interface and displays the analysis reports in an organized and visually appealing manner. Interactive graphs powered by Chart.js and Datatables enhance data presentation.
- Controller (C): The Controller acts as a mediator between the Model and View. It manages user requests, processes input, and communicates with the Model to retrieve the necessary data. The Controller ensures seamless data flow and handles data validation, maintaining data integrity.

Authentication and Session Management

Secure authentication using PHP's `session_start()` ensures that only authorized users can access the License History Analysis system. User credentials are verified, safeguarding sensitive data and enhancing overall system security.

Data Validation and Error Handling

Data validation mechanisms are implemented both on the client and server sides to guarantee the accuracy and security of user-provided data. Robust error handling ensures that issues encountered during user interactions, data retrieval, and analysis are managed gracefully. Informative error messages assist users in resolving problems effectively.

View Activity

Active Features & Users

The View Activity section is divided into two parts, offering detailed insights into license activity. Users can select a specific date to view the following:

- Active Features: This section displays the details of features that were active on the selected date.
- Active Users: Users can also view the users who were actively utilizing licenses on the chosen date.

Analysis parameters in this section are accompanied by their respective graphs and Datatables, providing a comprehensive view of license activity.

Software Interaction Timeline

Usage Statistics

The Software Interaction Timeline segment empowers users with granular usage statistics based on specific criteria, including:

- Start Date: Users can specify the start date for the analysis.
- End Date: The end date marks the conclusion of the selected analysis period.
- Software Type: Users can choose between ABAQUS or COMSOL software.
- Distribution Type: This parameter allows users to differentiate between user-based or machine-based license distribution.

The analysis results are complemented by interactive graphs and Datatables, enabling users to gain insights into license utilization patterns over time.

Day-to-Day Capability Analysis

Day-wise Feature Duration Analysis

The Day-to-Day Capability Analysis component is designed for in-depth analysis of feature utilization over specific time intervals. Users can:

- Select the desired features for analysis.
- Specify the time interval within which they want to analyze feature duration.
- Access detailed day-wise feature duration data.

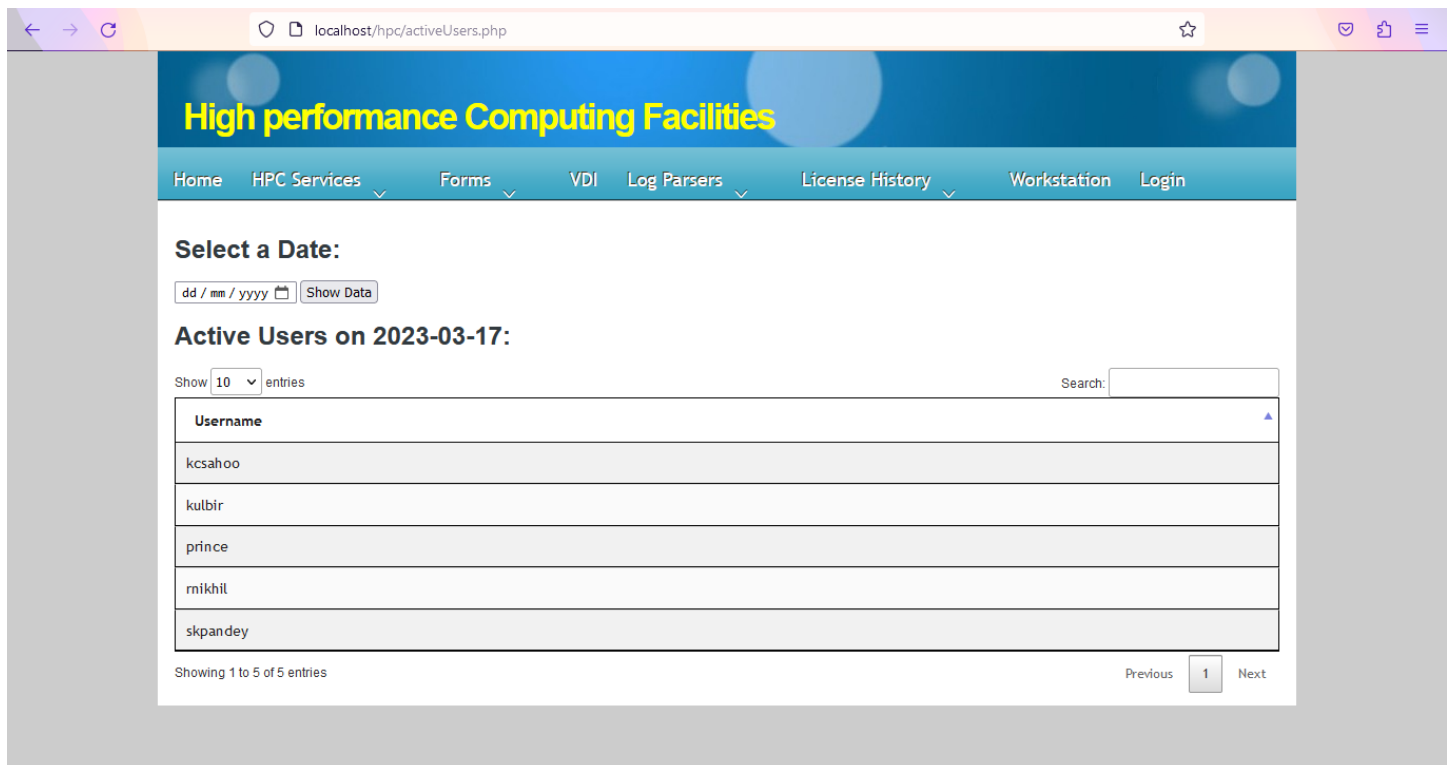
The project generates interactive graphs and Datatables that provide a clear understanding of how selected features are used day by day within the chosen time frame.

Technology Stack

The License History Analysis project leverages a versatile technology stack to deliver a seamless user experience:

- HTML: Structures web pages and constructs the user interface.
- PHP: Powers server-side scripting, manages database interactions, and handles data retrieval.
- CSS: Provides styling and layout for an aesthetically pleasing interface.
- JavaScript: Enhances client-side interactivity and drives Chart.js for dynamic graph generation.
- MySQL: Stores license history data efficiently and securely.
- Chart.js: Facilitates the creation of interactive visualizations for comprehensive data analysis.
- Datatables: Presents analysis results in a structured tabular format.

Screenshots and Visualization



Conclusion

The License History Analysis project is a powerful tool for gaining insights into license usage history. With its adherence to the MVC architecture, robust authentication, data validation, error handling, and interactive visualizations, the project empowers users to make data-driven decisions, optimize license allocation, and uncover valuable insights from license history data.

In the next chapter, we will provide a detailed walkthrough of the user interface and demonstrate how users can effectively utilize the system to access analysis reports and gain valuable insights for informed decision-making and resource management.

CHAPTER 7

WORKSTATION ANALYSIS

In this chapter, we will explore the Workstation Analysis project, a robust system that provides insightful utilization analysis of workstations (GWS) based on CPU Load, Memory Load, and overall utilization. Additionally, it offers granular insights into user-specific workstation usage. The system adheres to the Model-View-Controller (MVC) architecture, ensuring a modular and maintainable codebase. Workstation Analysis empowers users with a user-friendly interface, interactive graphs powered by Chart.js, and advanced features for detailed analysis.

MVC Architecture

Workstation Analysis embraces the MVC architectural pattern, dividing the application into three key components:

- Model (M): The Model represents the data layer. It interacts with the database and handles data operations. In this project, the Model manages key data attributes such as `gwsname`, `loggedinuser`, `CPULoad`, `MemLoad`, `CDrive`, and `Timemon`. The utilization data from GWS workstations (1-32) is collected and stored in the database.
- View (V): The View is responsible for the presentation layer. It renders the user interface, displaying data in an organized and visually appealing manner. The View also leverages Chart.js to create interactive graphs that visually represent the utilization data.
- Controller (C): The Controller acts as an intermediary between the Model and View. It handles user requests, processes input, manages data retrieval, and communicates with the Model to obtain the required data. The Controller ensures seamless data flow between the Model and View while maintaining data security and validation.

Authentication and Session Management

To ensure secure access to the system, Workstation Analysis implements robust authentication using PHP's `session_start()`. User credentials are verified against stored data in the system, allowing only authorized users to access utilization analysis. This security measure safeguards sensitive data and maintains privacy.

Data Validation and Error Handling

Workstation Analysis incorporates comprehensive data validation mechanisms. Input data is rigorously validated both on the client-side using JavaScript and on the server-side using PHP. This double layer of validation ensures that data is accurate and secure.

The system also features advanced error handling. It gracefully handles errors that may occur during user interactions, data retrieval, or graph generation. Informative error messages assist users in troubleshooting issues effectively.

Utilization Analysis

Overall Utilization

Workstation Analysis provides an overview of overall utilization for all workstations (GWS) throughout the specified duration. Utilization is measured on a scale of 0-100, allowing users to quickly assess the efficiency of their workstations.

CPU Load and Memory Load

The system offers detailed insights into CPU Load and Memory Load for each workstation. Users can analyze how CPU and memory resources are being utilized over time. Graphs generated by Chart.js offer interactive visualizations to facilitate data interpretation.

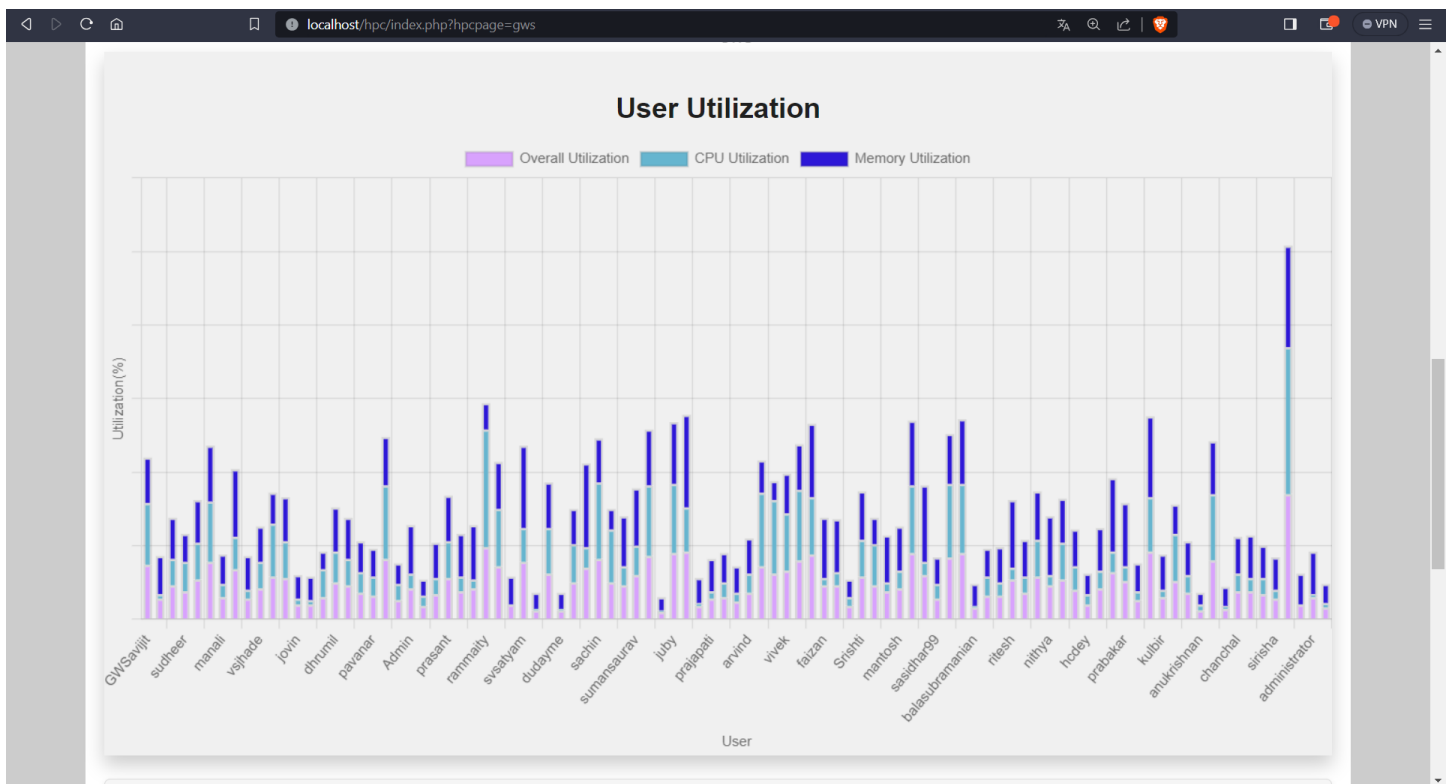
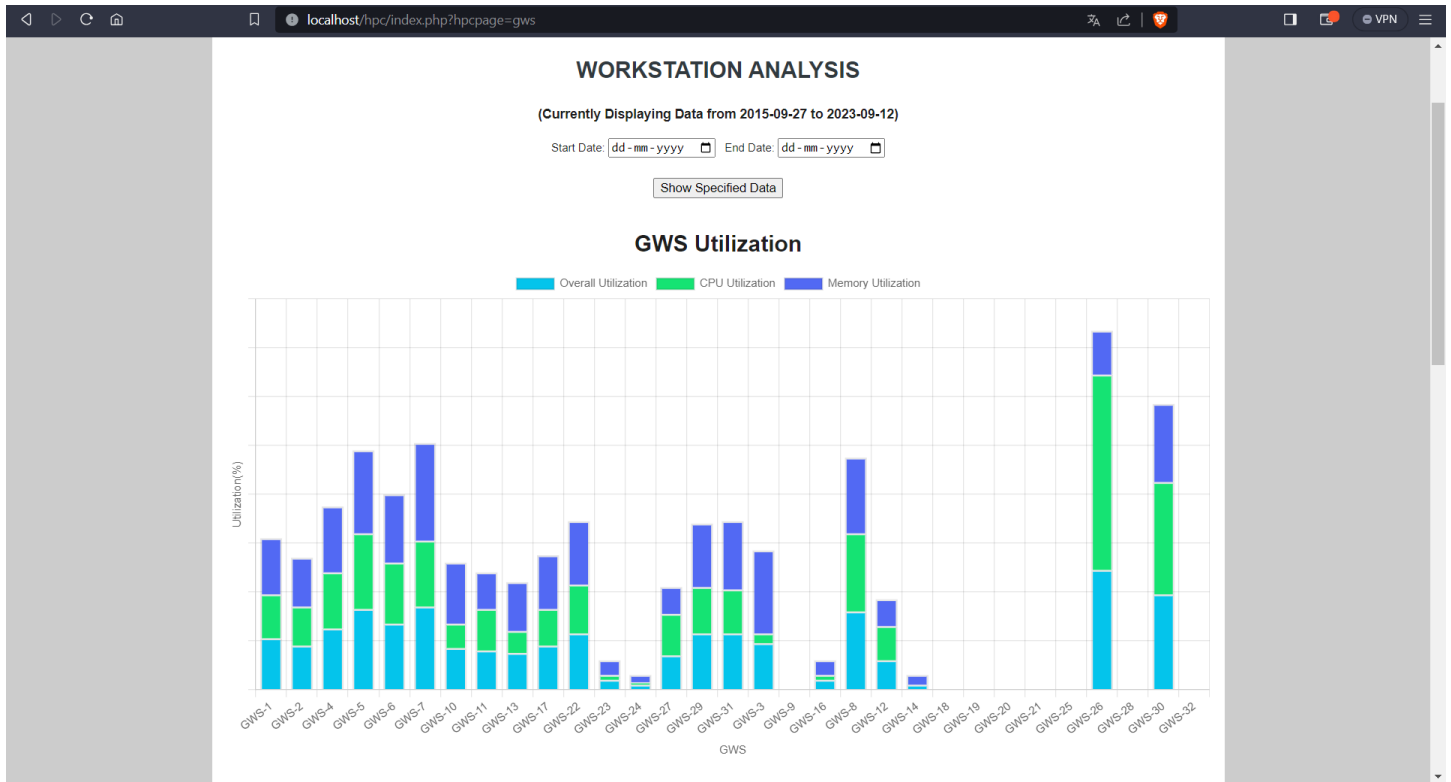
Workstation Analysis allows users to drill down into utilization data for specific durations. By default, the system displays data for the entire duration, but users can specify custom time intervals for analysis. This feature empowers users to identify trends, anomalies, and resource bottlenecks during specific periods.

Technology Stack

Workstation Analysis utilizes a diverse technology stack to deliver a powerful and user-friendly experience:

- HTML: Used for structuring web pages and creating the user interface.
- PHP: Empowers server-side scripting, database interactions, and data retrieval.
- CSS: Provides styling and layout for a visually appealing interface.
- JavaScript: Enhances client-side interactivity and powers Chart.js for dynamic graph generation.
- MySQL: Stores utilization data efficiently and securely.
- Chart.js: Drives the creation of interactive utilization graphs for insightful analysis.

Screenshots and Visualization



localhost/hpc/index.php?hpcpage=gws

User

Show

10

entries

Search:

USERNAME	GWS	CDRIVE (%)	MEMORY (%)	CPU (%)	OVERALL (%)
anupam	GWS-4	19	29	25	27
arav	GWS-13	43	17	2	9
arbabu	GWS-6	27	31	31	31
arjun	GWS-7	44	50	39	44
arulb	GWS-5	51	38	41	39
arvind	GWS-13	52	24	13	18
askmanson	GWS-22	62	30	25	28
athikrish	GWS-6	31	23	12	18
avijit	GWS-1	52	28	18	23
balasubramanian	GWS-13	9	15	1	8

Showing 11 to 20 of 95 entries

Previous

1

2

3

4

5

...

10

Next

Conclusion

Workstation Analysis is a comprehensive tool for analyzing workstation utilization, making it invaluable for system administrators and IT professionals. With its adherence to MVC architecture, robust authentication, advanced data validation, error handling, and interactive graphs, Workstation Analysis provides a seamless and efficient way to optimize workstation performance and resource allocation.

CHAPTER 8

CONCLUSION

Throughout this internship, I embarked on a comprehensive exploration of various web development aspects, frameworks, and applications. The journey began with an in-depth understanding of the Model-View-Controller (MVC) architecture, laying the foundation for structuring robust and scalable web applications. The internship commenced with the setup of a local development environment using WAMP (Windows, Apache, MySQL, PHP/Python/Perl) server. Understanding the intricacies of this environment proved pivotal for subsequent tasks, ensuring a seamless development and testing process. In the realm of MVC architecture, the creation of EduSync exemplified the practical implementation of theoretical concepts. The project showcased the importance of authentication, session management, data validation, and error handling in crafting a secure and reliable Student Database Management System.

The exploration of the HPCS Portal provided insights into user privileges, access control, data collection, and monitoring. The robust database design, coupled with an intuitive web design, underscored the significance of an organized and user-friendly interface for effective data management. An understanding of Finite Element Analysis (FEA) software, including Abaqus, ANSYS, and COMSOL, broadened the scope. These insights laid the groundwork for Log Parsing and Analysis, delving into software-specific log parsing methodologies and analysis techniques.

The Log Parsing & Analysis chapter elucidated the intricacies of parsing raw log files, implementing MVC architecture, and deploying technologies like PHP, HTML, and Chart.js. The project emphasized the importance of custom time interval analysis for nuanced insights into software usage patterns.

In the License History Analysis chapter, we explored a multifaceted system designed to provide detailed insights into license usage patterns. The project demonstrated the practical application of MVC architecture, authentication, and data validation, while incorporating interactive visualizations for effective data presentation.

The Workstation Analysis chapter showcased the development of a system for analyzing workstation utilization. MVC architecture, authentication, and utilization analysis were central to the project, providing valuable insights into overall system performance.

The technology stacks employed throughout the internship, encompassing HTML, PHP, CSS, JS, and various frameworks, highlighted the versatility and interoperability of these tools in crafting robust and dynamic web applications.

In conclusion, this internship has been a transformative journey, offering hands-on experience in web development, database management, and system analysis. The practical implementation of theoretical concepts, coupled with exposure to real-world applications, has fortified foundational skills and fostered a deeper appreciation for the intricacies of web development. As I conclude this internship report, it is evident that the acquired knowledge and practical insights gained throughout these chapters will serve as a solid foundation for future endeavors in the dynamic and ever-evolving field of web development.

Thank you for the enriching experience!