Project Report on Telemetrix Insighthub

Submitted in partial fulfilment of the course of
PG-Diploma
In

Advanced Computing
From C-DAC ACTS (Bangalore)

Guided by:
Mr. Ravikumar Reddy

Presented by:

Mr. Shivam Suresh Mhasaye	PRN: 230950120135
Ms. Mrinmayee Patil	PRN: 230950120079
Ms. Anushka Nagvekar	PRN: 230950120030
Ms. Gautami Hirawat	PRN: 230950120052
Ms. Shalini Shere	PRN: 230950120130

Candidate's Declaration

We hereby certify that the work being presented in the report titled: **Telemtrix Insighthub**, in partial fulfilment of the requirements for the award of PG Diploma Certificate and submitted to the department of PG-DAC of the C-DAC ACTS Bangalore, is an authentic record of our work carried out during the period, 1st Feb 2024 to 13th March 2024 under the supervision of Mr. B Reddy Ravikumar, C-DAC Bangalore. The matter presented in the report has not been submitted by us for the award of any degree of this or any other Institute/University.

Name and Signature of Candidate:

Mr. Shivam Suresh Mhasaye	PRN: 230950120135
Ms. Mrinmayee Patil	PRN: 230950120079
Ms. Anushka Nagvekar	PRN: 230950120030
Ms. Gautami Hirawat	PRN: 230950120052
Ms. Shalini Shere	PRN: 230950120130

CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

Mr.Shivam Suresh Mhasaye

Ms. Mrinmayee Patil

Ms. Anushka Nagvekar

Ms. Gautami Hirawat

Ms. Shalini Shere

Have successfully completed their project on

Telemterix Insighthub

Under the guidance of

Mr. B Reddy Ravikumar

Mr. B Reddy Ravikumar	Ms. Garima Singh
(Project Guide)	(Course Co-ordinator)

Acknowledgement

This project **"Telemetrix Insighthub"** we was a great learning experience for us and are submitting this work to Advanced Computing Training School (CDAC ACTS), Bengaluru.

We all are very glad to mention the name of **Mr. B Reddy Ravikumar** for his valuable guidance to work on this project. His guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

Our most heartfelt thanks goes to **Ms. Garima Singh** (Course Coordinator, **PG-DAC**) who gave all the required support and kind coordination to provide all the necessities like extra Lab hours to complete the project and throughout the course up to the last day at C-DAC ACTS, Bangaluru.

From,
Mr. Shivam Suresh Mhasaye
Ms. Mrinmayee Patil
Ms. Anushka Nagvekar
Ms. Gautami Hirawat
Ms. Shalini Shere

ABSTRACT

"Technology makes the world a new place" is what people say. To make our lives easier new innovations are always made.

The rapid advancement of technology has paved the way for sophisticated telemetry monitoring systems that enable real-time data collection, analysis, and control in various domains. This project proposes the development of a web-based telemetry monitoring system designed to enhance the efficiency and reliability of remote data monitoring and control processes.

The proposed system will utilize modern web technologies to create an intuitive and user-friendly interface accessible through standard web browsers. The core functionality of the system will include the collection, visualization, and analysis of telemetry data from diverse sources, such as sensors, devices, and instruments, facilitating seamless monitoring of critical parameters.

The proposed web-based telemetry monitoring system is intended to find applications in various sectors, including industrial automation, environmental monitoring, healthcare, and infrastructure management. By offering a scalable and adaptable solution, this project aims to contribute to the optimization of remote monitoring processes, enhancing overall system reliability and performance.

Contents

1. Introduction and Overview

Project Introduction

Telemetrix InsightHub is a project that utilizes a stack consisting of InfluxDB, Node.js, and React.js to collect system metrics such as CPU usage, free memory, etc., and visualize them for monitoring purposes. Telemetry is a communication process used to collect data from remote or inaccessible points and transmit it to receiving equipment for monitoring and analysis. It plays a critical role in aerospace, healthcare, environmental monitoring, industrial automation, and IoT. A typical telemetry system comprises sensors, data acquisition units, transmitters, receivers, and data processing software. Telemetry systems can be remote, wired, wireless, or satellite-based. Applications of telemetry include aerospace and defense, healthcare, environmental monitoring, industrial automation, and transportation. Telemetry enables remote monitoring, control, and analysis of data, leading to improved efficiency, safety, and decision-making across.

Project Overview

Existing System

The existing system consists of inbuilt features i.e. Task manager which typically relies on manual data collection, leading to inefficiencies, data silos, and limited real-time monitoring capabilities. Without advanced visualization tools, scalability, and accessibility, traditional systems hinder effective data analysis and decision-making. Upgrading to modern technologies like InfluxDB and Grafana enables automated data collection, centralized storage, real-time monitoring, and intuitive visualization, addressing these limitations and enhancing operational efficiency.

Problem Definition- Need of Computerization

1. The Telemetrix Insighthub System is designed to eliminate the problem of the current system. This accessibility of the information will be a great advantage as it reduces effort.
2. It addresses challenges such as inefficient data collection processes, lack of real-time monitoring capabilities, and scalability constraints. These limitations impede operational efficiency, decision-making, and the ability to derive actionable insights from telemetry data. Transitioning to a computerized system is essential to overcome these challenges, offering benefits such as improved data accuracy, real-time monitoring, scalability, and enhanced operational efficiency.
3. Telemetry monitoring systems offer real-time data insights, remote accessibility, and predictive maintenance capabilities, enhancing operational efficiency, cost savings, and safety across various industries.

2.Literature Survey

This literature review explores the potential of using Node.js, InfluxDB, and Grafana to build a telemetry system for collecting and visualizing system metrics from client computers on a server-side dashboard.

Node.js for Telemetry Systems

Node.js is a popular choice for building real-time applications due to its event-driven, non-blocking architecture. Studies like [1] showcase its effectiveness in handling large volumes of data streams, making it suitable for collecting and processing telemetry data. Additionally, its extensive ecosystem of libraries, like [2], simplifies tasks like network communication and data manipulation, further supporting its use in telemetry projects.

InfluxDB for Time-Series Data

InfluxDB is a time-series database (TSDB) optimized for storing and querying large amounts of data with timestamps. Studies like [3] highlight its performance and scalability in handling high-frequency data streams, which is crucial for efficiently storing and managing telemetry data. Furthermore, its built-in query language (InfluxQL) allows for flexible data retrieval and analysis, enabling the creation of insightful dashboards.

Grafana for Data Visualization

Grafana is a popular open-source platform for visualizing time-series data. Studies like [4] demonstrate its effectiveness in creating interactive dashboards and real-time visualizations, allowing users to easily monitor and understand complex datasets. Its support for various data sources, including InfluxDB, makes it an ideal choice for visualizing the telemetry data collected from client machines

3.System and Software Requirements

Hardware Specifications

Machine: Desktop/Laptop

Operating system: Windows 8 or above

Processors: Intel core i3 or AMD 5 series (minimum)

Memory: 8 GB RAM or above

Hard Disk (SSD): 250 GB or more

Video Card (optional): Intel Integrated Graphics (suggested – 4 GB graphics card - NVIDIA)

Network: Ethernet / Wi-Fi with 25 Mbps Speed Connection (UL/DL)

Software Specifications

Platform: Windows 10/11

Frontend: Reactjs, html, bootstrap

Backend: Nodejs.

Database: Influxdb, Mysql

Visualization Tool: Grafana

4.Architecture

1. Collect System Metrics (Node.js):

- This step involves using Node.js to develop a backend service responsible for collecting system metrics such as CPU usage, free memory, disk usage, network traffic, etc.
- Node.js libraries such as os-utils or system information can be used to gather system metrics.
- The backend service periodically retrieves system metrics data from the operating system and prepares it for storage.

2. Store Metrics in InfluxDB:

- Once the system metrics are collected, they are stored in InfluxDB, a time-series database optimized for handling high volumes of time-stamped data.
- InfluxDB efficiently stores the system metrics data and provides fast querying capabilities for data retrieval and analysis.
- The collected metrics are stored in InfluxDB using its write API or client libraries available for Node.js.

3. Visualize Data with Grafana:

- Grafana, an open-source visualization and monitoring platform, is used to create customizable dashboards for visualizing system metrics data stored in InfluxDB.
- Grafana connects to InfluxDB as a data source and retrieves the stored metrics data.
- Users can create interactive dashboards with various visualization options such as graphs, gauges, tables, and heatmaps to monitor system performance in real-time.

4. Display Data on Frontend (React.js):

- The React.js frontend application interacts with the backend service to retrieve system metrics data stored in InfluxDB.
- The retrieved data is displayed on the frontend interface, providing users with a comprehensive overview of system performance metrics.
- Users can view and interact with the system metrics data, customize the display, and derive actionable insights from the visualizations presented on the frontend.

This flowchart outlines the basic workflow of the telemetry project, from collecting system metrics to visualizing and displaying the data for monitoring and analysis purposes. Each step in the process contributes to creating a robust monitoring solution that enables users to monitor system performance effectively in real-time. Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.

Project flowchart

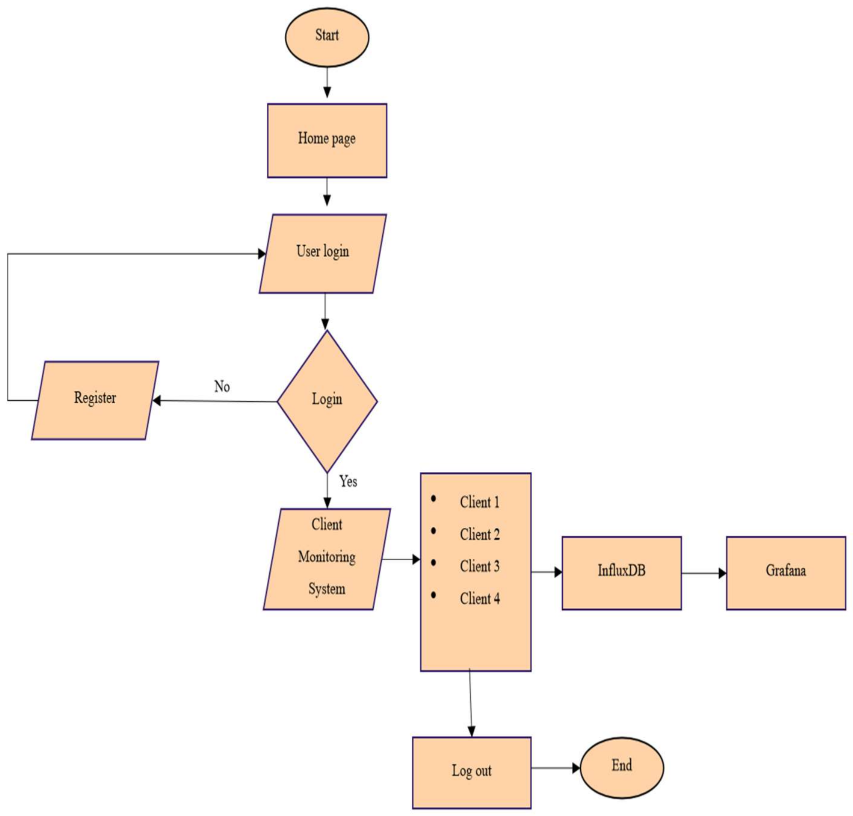


Fig: Project Flowchart

5. System Design

Assessing the feasibility of a telemetry product involves evaluating its technical, economic, operational, and schedule feasibility. Here's how you can assess each aspect for the telemetry project:

1. Technical Feasibility:

- Technology Stack: Evaluate the suitability and compatibility of the chosen technologies (InfluxDB, Node.js, React.js, Grafana) for building the telemetry solution. Ensure that these technologies can effectively collect, store, visualize, and display system metrics.
- Integration: Assess the feasibility of integrating the various components of the telemetry solution (backend, database, visualization, frontend) to ensure seamless communication and data flow.
- Scalability and Performance: Determine whether the chosen technologies can handle the expected volume of telemetry data and provide the required performance and scalability for real-time monitoring and analysis.

2. Economic Feasibility:

- Cost Analysis: Estimate the costs associated with developing, deploying, and maintaining the telemetry solution. Consider factors such as hardware, software licenses, development resources, infrastructure, and ongoing operational expenses.
- Return on Investment (ROI): Evaluate the potential benefits of the telemetry solution, including cost savings, efficiency improvements, risk mitigation, and revenue generation opportunities. Determine whether the expected ROI justifies the investment required to develop and deploy the product.

3. Operational Feasibility:

- User Requirements: Ensure that the telemetry solution meets the needs and requirements of its intended users, including system administrators, operations teams, and other stakeholders. Conduct user interviews, surveys, or focus groups to gather feedback and validate user requirements.
- Resource Availability: Assess the availability of skilled personnel, expertise, and resources required to develop, deploy, and support the telemetry solution. Identify any gaps or constraints that may impact the project's feasibility and develop mitigation strategies.

By evaluating the technical, economic, operational, and schedule feasibility of the telemetry project, you can make informed decisions about its viability and determine the best course of action for moving forward with development and deployment.

6. Implementation

:

Development Environment:

The telemetry project was developed in a collaborative environment using modern development tools and platforms. Visual Studio Code (VS Code) served as the primary Integrated Development Environment (IDE) due to its lightweight, extensible, and feature-rich environment. Git was used as the Version Control System (VCS), enabling seamless collaboration among team members, code versioning, and tracking changes throughout the development lifecycle. The project repository was hosted on GitHub, providing a centralized location for code management and collaboration.

Technology Stack:

The telemetry project leveraged a modern technology stack comprising various programming languages, frameworks, libraries, and tools. Node.js was chosen as the backend runtime environment for its asynchronous and event-driven architecture, enabling real-time data processing and communication between clients and servers. InfluxDB served as the time-series database for storing telemetry data, offering scalability and high-performance querying capabilities. Grafana was utilized for data visualization and dashboarding, providing an intuitive interface for monitoring system metrics in real-time.

Module Implementation:

The implementation of the telemetry project comprised several modules, each responsible for specific functionalities within the system. The key modules included Telemetry Agents, Data Processing Server, Database Management, and Data Visualization. Telemetry Agents were developed using Node.js to collect system metrics from client machines. The Data Processing Server, also implemented in Node.js, received telemetry data, processed it, and stored it in InfluxDB. Database Management involved designing the database schema and implementing data retention policies. Data Visualization was achieved using Grafana, where custom dashboards were created to visualize system metrics.

Database and Data Storage:

InfluxDB was integrated into the telemetry system as the primary database for storing time-series telemetry data. The database schema was designed to accommodate various types of telemetry metrics, with appropriate retention policies and data retention strategies implemented to manage data lifecycle and optimize storage utilization. Data was stored in InfluxDB in a structured format, allowing for efficient querying and retrieval of telemetry data.

User Interface (UI) Design and Development:

The telemetry project included a user interface developed using HTML, CSS, and JavaScript. The UI design followed principles of simplicity, intuitiveness, and responsiveness to ensure a seamless user experience. The frontend was implemented using modern frontend frameworks such as React.js, enabling modular, component-based UI development. User interface components were designed and implemented to display real-time system metrics fetched from the backend.

7 Conclusion

7.1 Limitations & Drawbacks:

Performance Bottlenecks: Complex queries or visualizations can lead to performance issues, necessitating careful optimization.
Complex Implementation: Setting up and configuring the system can be complex, requiring expertise and proper documentation.
Dependency on Third-party Tools: Reliance on external tools introduces dependency risks if support or development changes.

7.2 Future Enhancement:

Machine Learning and Predictive Analytics: Incorporate machine learning algorithms to analyze telemetry data patterns and predict future events or anomalies, enabling proactive maintenance and optimization.

Advanced Visualization Techniques: Explore advanced visualization techniques such as 3D visualization, heatmaps, or geographic mapping to provide more intuitive and insightful representations of telemetry data.

Integration with External Systems: Enhance interoperability by integrating the telemetry monitoring system with other enterprise systems, such as asset management, maintenance scheduling, or IoT platforms, to enable data sharing and streamline workflows.

Automated Alerting and Notification: Implement automated alerting and notification mechanisms to promptly notify users of critical events, threshold breaches, or anomalies detected in telemetry data, facilitating timely responses and decision-making.

Enhanced Security Features: Strengthen security measures by implementing additional authentication methods, encryption techniques, and audit logging to protect sensitive telemetry data and mitigate security risks.

Data Compression and Storage Optimization: Optimize data storage and retrieval processes by implementing data compression techniques, data aggregation, or tiered storage strategies to reduce storage costs and improve system performance.

Enhanced User Collaboration Features: Incorporate features such as user collaboration tools, shared dashboards, and data annotation capabilities to facilitate collaboration and knowledge sharing among users, improving teamwork and decision-making processes.

7.3 Conclusions:

Our telemetry monitoring project, leveraging InfluxDB and Grafana, has revolutionized real-time data visualization, accuracy, and operational efficiency. Addressing scalability, performance, and security challenges, we integrated advanced features like machine learning and customizable alerts. Emphasizing our project's contributions underscores its significance in advancing telemetry monitoring practices for enhanced decision-making and operational excellence. By streamlining data collection and analysis processes, our system offers proactive maintenance insights and facilitates timely responses to critical events. The integration with external systems ensures seamless data sharing and interoperability, while adherence to industry best practices safeguards data integrity and privacy. Overall, our project represents a significant step forward in modernizing telemetry monitoring, empowering organizations to optimize operations and drive innovation in a rapidly evolving digital landscape.

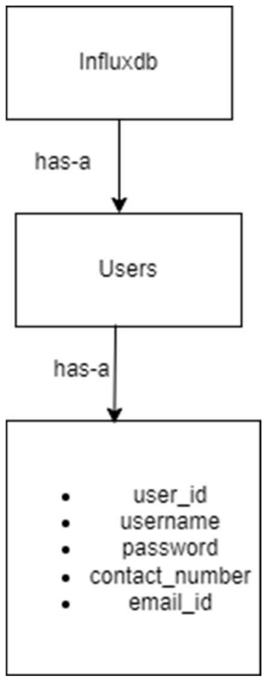
In conclusion, machine learning is an important tool for making predictions and solving complex problems. It can be used to improve decision making and create more efficient processes. With the advances in technology, machine learning will continue to be used in many industries and become an integral part of our lives. Even though it is still in its early stages, its potential is enormous. Its applications will continue to expand and become more versatile as technology continues to advance.

8.References

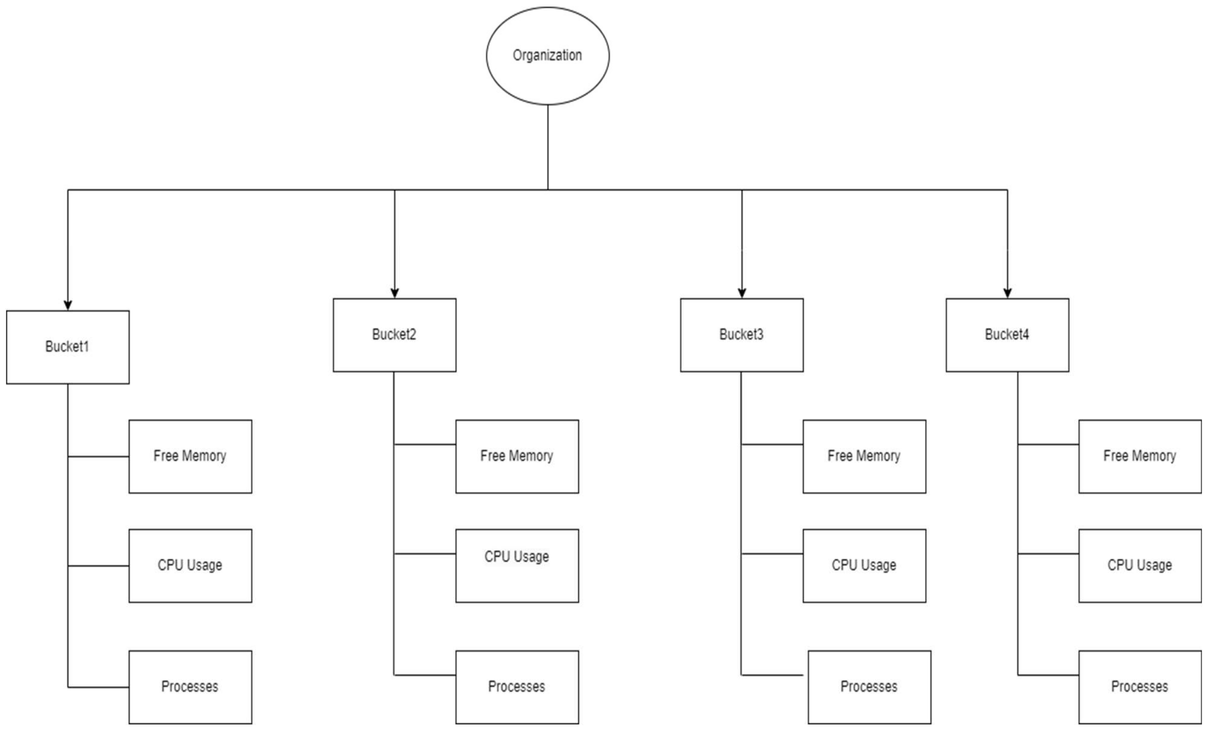
- [1] Chen, S., Wang, Y., & Liu, X. (2018). An Efficient and Scalable Real-time Big Data Processing Platform Based on Node.js. https://www.sciencedirect.com/science/article/pii/S1877050918317927
- [2] Node.js https://node.js.org/
- [3] Anastasova, A., & Zlatev, Z. (2016). Performance Analysis of Time-Series Databases for Industrial Big Data. https://arxiv.org/pdf/2208.13982
- [4] Grafana Labs https://grafana.com/
- InfluxData. (2023). InfluxDB: Time Series Database. https://www.influxdata.com/
- Grafana Labs. (2023). Grafana: The open and composable analytics platform. https://grafana.com/
- McNett, J. (2015). Node.js in Action. Manning Publications Co.
- Kumar, S., & Rastogi, R. (2019). Security considerations for IoT-based telemetry systems. 2019 9th International Conference on Cloud Computing, Data Science & Engineering (CONFLUENCE), 512-517. https://ieeexplore.ieee.org/document/7412116
- Gronager, M. (2022). Security best practices for access control in serverless applications. Nordic APIs Platform Summit, 2022.
- NIST. (2023). Cybersecurity Framework. https://www.nist.gov/cyberframework
    - Zhang, Y., Li

List of figures

ER-Diagram:



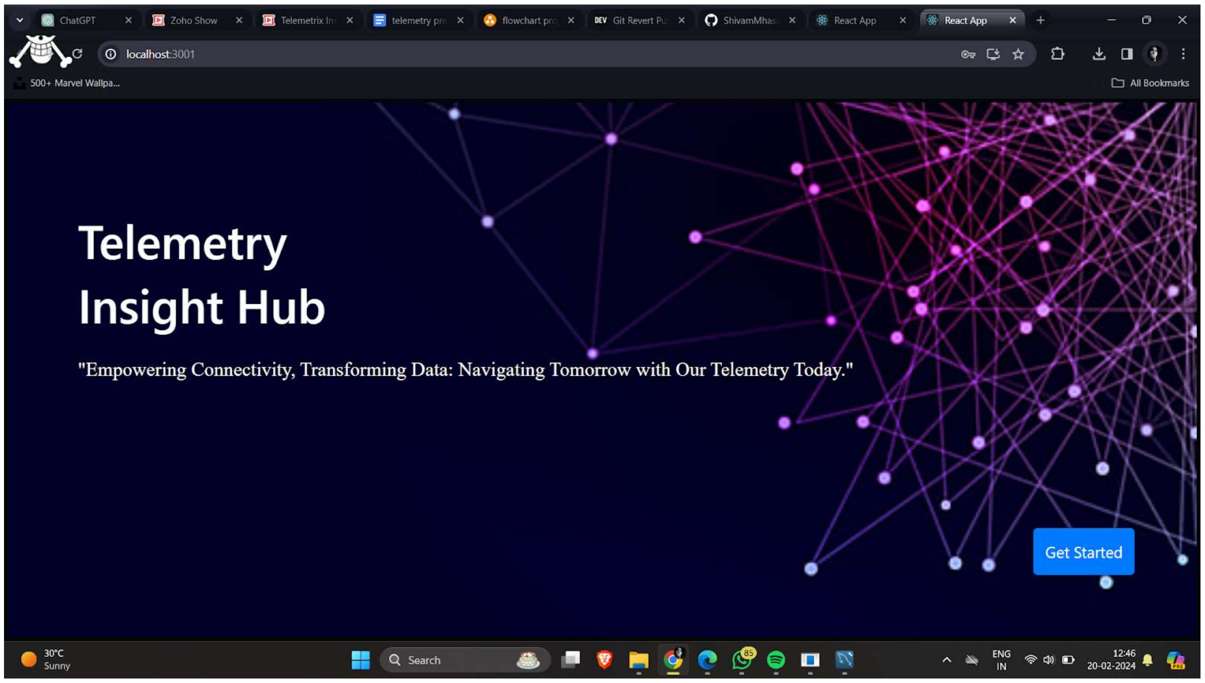
E-R diagram:-Admin Login



E-R diagram:-InfluxDB

Screen Shots:

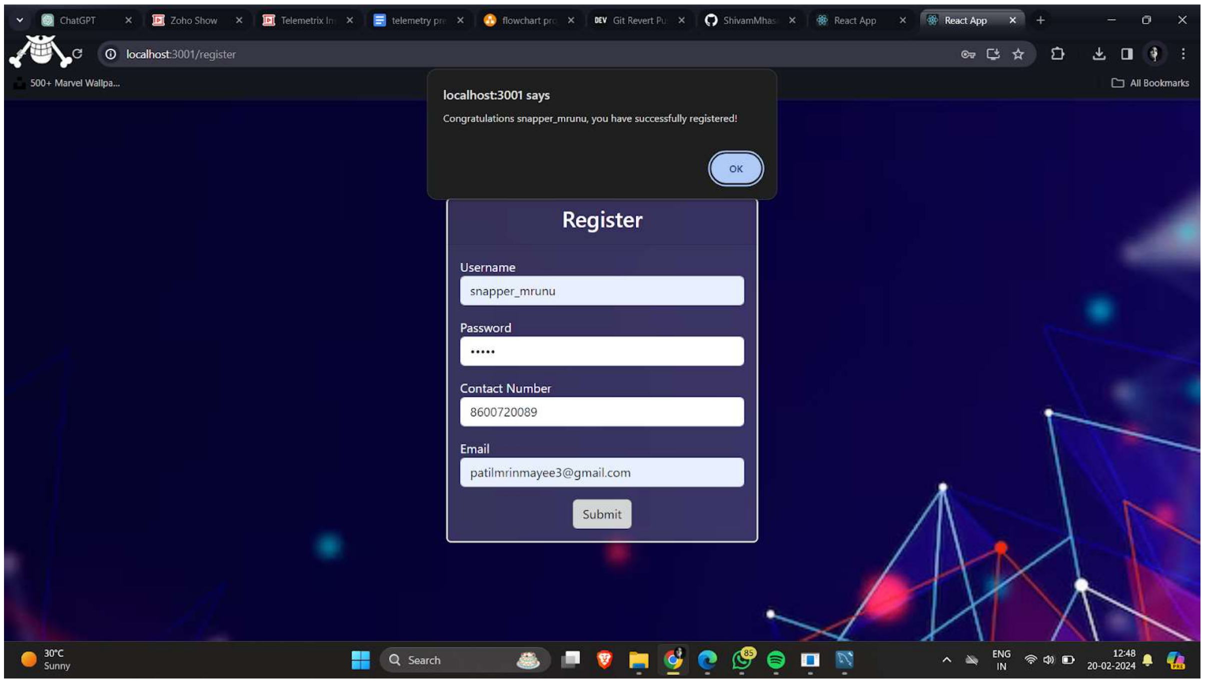
Home Page:


Telemetry
Insight Hub
"Empowering Connectivity, Transforming Data: Navigating Tomorrow with Our Telemetry Today."
Get Started

This page contains following controls

- Home

Register Page



localhost:3001 says

Congratulations snapper_mrunu, you have successfully registered!

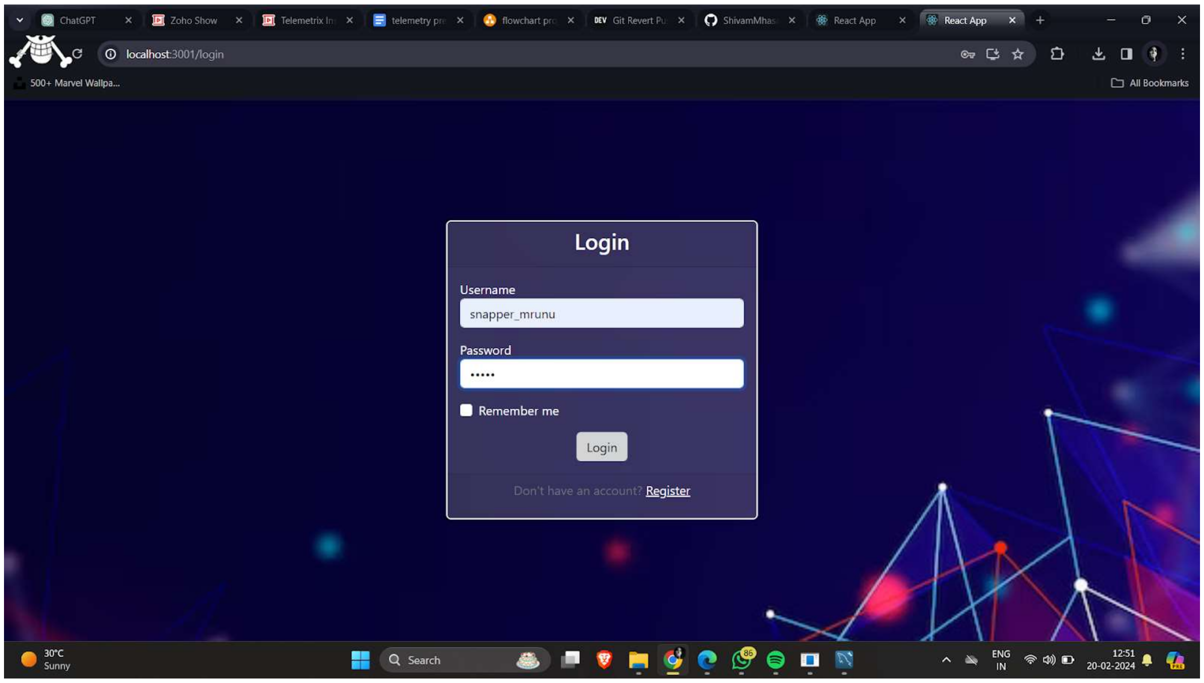
Username: snapper_mrunu

Password: •••••

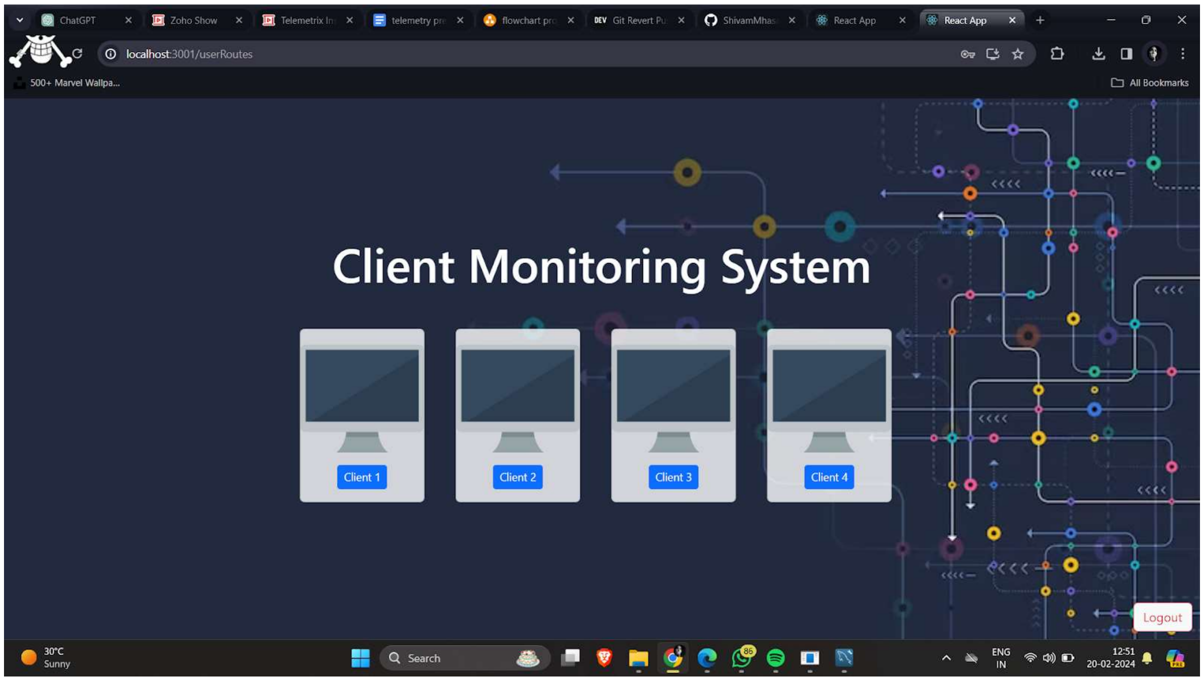
Contact Number: 8600720089

Email: patilmrinmayee3@gmail.com

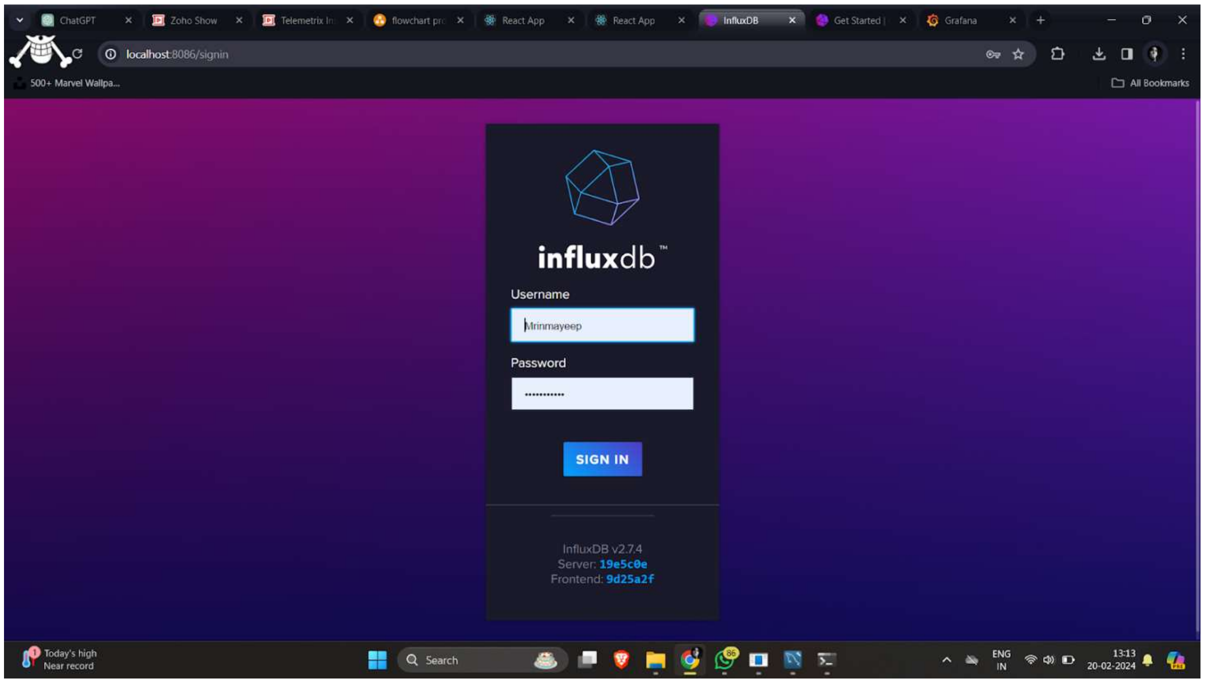
Login Page



Client Monitoring System



Influxdb Login Page



localhost:8086/signin

Username

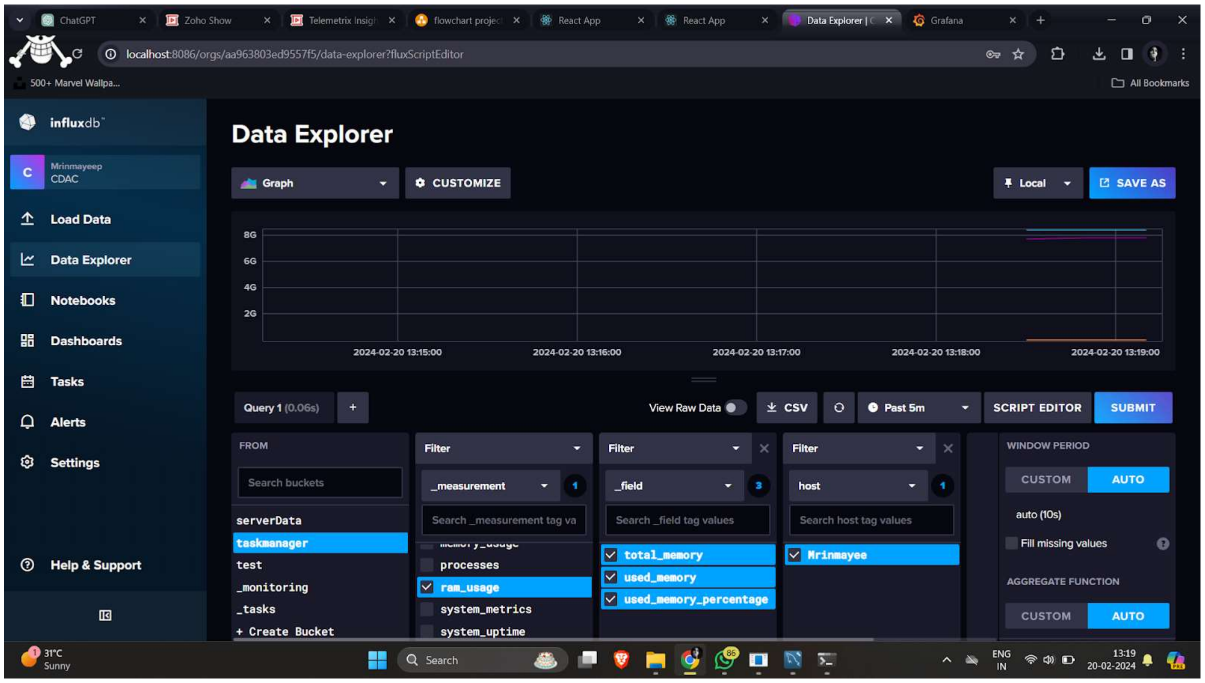
Mrinmayeep

Password

••••••••••

InfluxDB v2.7.4
Server: 19e5c0e
Frontend: 9d25a2f

Influxdb data Explorer Page



localhost:8086/orgs/aa963803ed9557f5/data-explorer?fluxScriptEditor

Data Explorer

Mrinmayeep
CDAC

Graph | Local

Query 1 (0.06s) | View Raw Data | Past 5m

FROM: Search buckets
serverData
taskmanager
test
_monitoring
_tasks
+ Create Bucket

Filter: _measurement 1
Search _measurement tag va
memory_usage
processes
ram_usage ✓
system_metrics
system_uptime

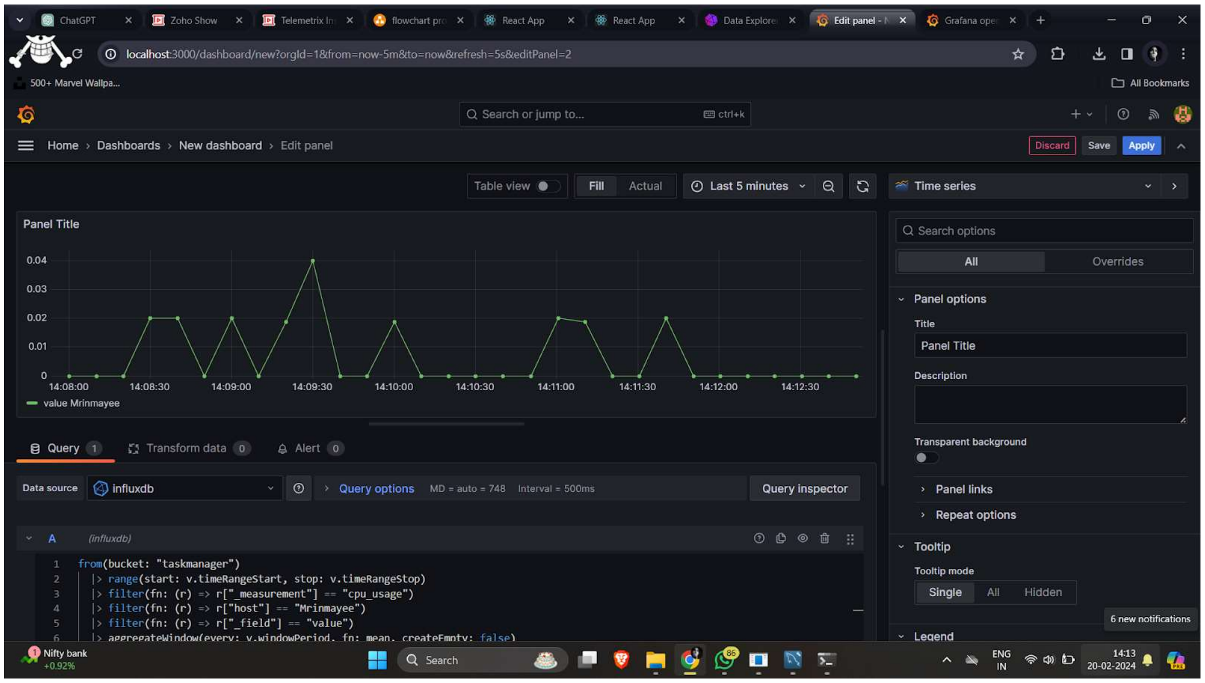
Filter: _field 3
Search _field tag values
total_memory ✓
used_memory ✓
used_memory_percentage ✓

Filter: host 1
Search host tag values
Mrinmayee ✓

WINDOW PERIOD: CUSTOM [✓ AUTO]
auto (10s)
Fill missing values

AGGREGATE FUNCTION: CUSTOM [✓ AUTO]

Grafana setup Dashboard Page



localhost:3000/dashboard/new?orgId=1&from=now-5m&to=now&refresh=5s&editPanel=2

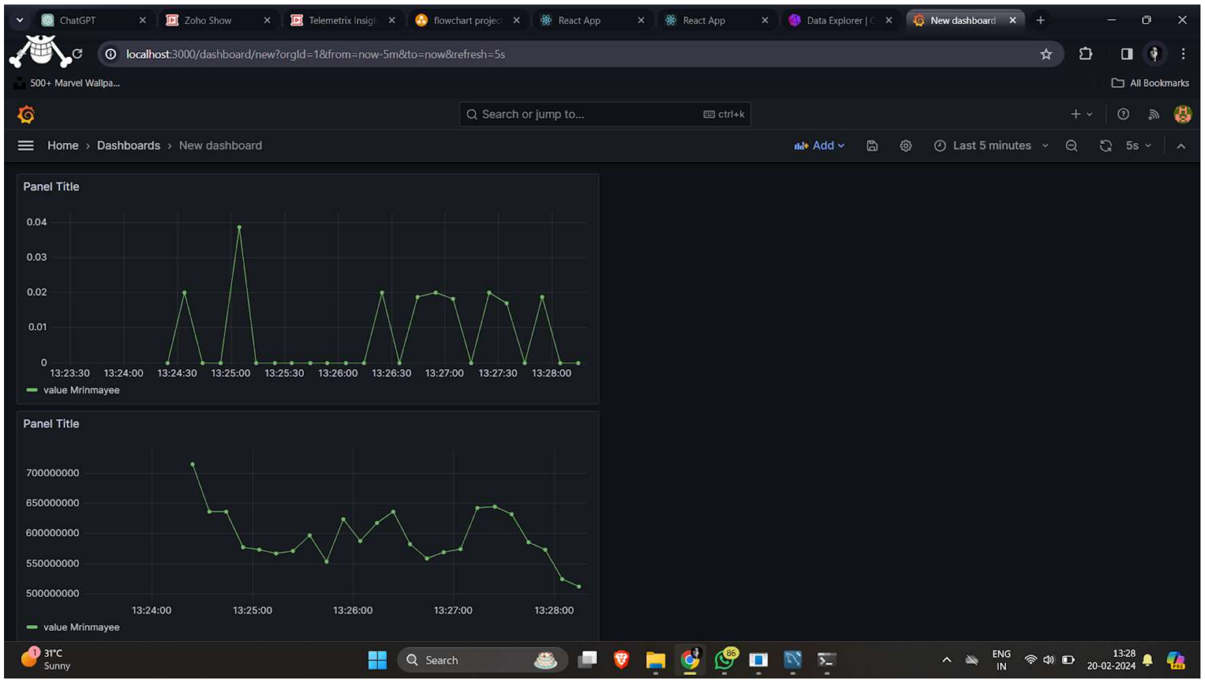
Home › Dashboards › New dashboard › Edit panel

Panel Title

Data source: influxdb — Query options — MD = auto = 748 — Interval = 500ms

```
from(bucket: "taskmanager")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "cpu_usage")
  |> filter(fn: (r) => r["host"] == "Mrinmayee")
  |> filter(fn: (r) => r["_field"] == "value")
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
```

Grafana visualization Page



localhost:3000/dashboard/new?orgId=1&from=now-5m&to=now&refresh=5s

Home › Dashboards › New dashboard

List of Tables

Field	Type	Null	Key	Description
user_id	Int	No	Primary Key	User Id
username	Varchar(50)	No		User Name
password	Varchar(15)	No		Password
contact_number	Varchar(12)	No		Mobile Number
email_id	Varchar(20)	No		Email

Acronyms and Abbreviation

SRS - Software Requirement Specification

InfluxDB - Influx DataBase (time-series database)

API - Application Programming Interface.

MQTT – MQ telemetry Transpor

t HTTP – Hyper Text Transfer Protocol

LDAP - Lighweight directory Access Protocol

CPU- Central Processsing Unit