RFI Tower Data Collection and Plotting Tools Development

Student Project By:

Janhavi Rajesh Dixit

Usha Mittal Institute of Technology,

Santacruz (W), Mumbai

Under guidance of:

Dr. Shubhendu Joardar

Mr. Yogesh Gaikwad

NCRA • TIFR

GIANT METERWAVE RADIO TELESCOPE

NATIONAL CENTER FOR RADIO ASTROPHYSICS

TATA INSTITUTE OF FUNDAMENTAL RESEARCH

Narayangaon, Tal-Junnar, Dist-Pune

June 2024 – August 2024

Acknowledgement

I would like to express my sincere gratitude to Mr. Yashwant Gupta, Distinguished Professor (J) and Centre Director at NCRA-TIFR and Professor C. H. Ishwara Chandra, Professor H GMRT, for his support for providing me with the invaluable opportunity to work in the esteemed organization. The opportunity to collaborate and work here has been an enriching experience, significantly contributing to my professional and personality growth.

I would like to express my heartfelt appreciation and gratitude to my mentors,
Dr. Shubhendu Joardar and Mr. Yogesh Gaikwad, for their invaluable guidance, encouragement, and insightful suggestions throughout this project. Their unwavering support, constructive feedback, and expert advice were instrumental in the successful completion of this report. Their dedication and commitment to fostering my understanding of the subject have significantly contributed to the depth and quality of this work.

I am also profoundly grateful to N. S. Deshmukh Ma'am, the STP Coordinator, for her exceptional support and coordination throughout the project. Her assistance in navigating the procedural aspects and her encouraging words greatly facilitated the progress of my work. Her dedication to ensuring a smooth and productive research experience was invaluable.
Thank you all for your constant assistance, direction, and the positive impact you have had on my academic journey.

Janhavi Dixit

Abstract:

Radio astronomy is increasingly hindered by Radio Frequency Interference (RFI), which complicates the accurate collection and analysis of astronomical data. Despite substantial efforts from governmental bodies and advancements in geographical site selection, man-made interference continues to escalate. To combat this challenge, an effective Omni-Directional RFI Monitoring System has been developed to support the establishment of RFI-free zones for radio telescopes. This system is specifically designed to study the effects of RFI on data collected by the Giant Metrewave Radio Telescope (GMRT) and to identify potential RFI sources.
The monitoring system is equipped with four log-periodic antennas, oriented towards the East, West, North, and South directions, and positioned at a height of 20 meters. These antennas operate in the E-plane to ensure comprehensive coverage of the surrounding environment. The system utilizes a cyclic antenna switching mechanism, controlled by a computer through a parallel port and decoder circuit. The antennas are sequenced in a fixed order—North, West, South, and East—during observations.

Each antenna captures incoming RFI signals, which are then amplified by a Low Noise Amplifier (LNA) before undergoing spectrum analysis. The analyzed data, representing the RFI levels in each direction, is stored in separate files corresponding to the four directional views. This cyclic process of antenna switching, amplification, spectrum analysis, and data recording is repeated continuously throughout the observation period.

By systematically analyzing the RFI data captured from multiple directions, this monitoring system facilitates a detailed understanding of the interference environment. The collected data supports statistical analysis and helps in identifying patterns and sources of RFI. Consequently, it contributes significantly to efforts in mitigating RFI, thereby enhancing the quality of radio astronomical observations and the effectiveness of RFI -free zone establishment.

INDEX

CHAPTER - 1

INTRODUCTION

1.1 Introduction:

The Giant Metre-wave Radio Telescope (GMRT) is a recent world class instrument for studying astrophysical phenomena at low radio frequencies (40 to 1450 MHz). GMRT has been designed and built in India by NCRA- TIFR as a national project at a low cost of about 15 million US (1992) $. The array telescope consists of 30 antennas of 45 metres diameter, each, operating at metre wavelengths, being collectively, the largest in the world at these frequencies. These antennas are spread over 25 km in diameter, with about half the antennas randomly located in a central one square km. area and the remaining ones along the three 14 km arms of an approximate 'Y' configuration. The longest baseline is 26 km and the shortest about 100 m. GMRT is located (Fig. 1a), (Lat. = 19.1° , Long. = -74.05° , Alt.=650 m) near Khodad Village, about 80 km north of Pune city and 100 kms. east of Mumbai (Bombay). The array schematic is shown in Figure 1b. Each of the GMRT antenna is an alt-az mounted dish (Figure 2). The dish has 16 parabolic frames which give the basic shape. The reflecting surface consists of a "Stretched Mesh Attached to Rope Trusses" (SMART) (Figure 3). The wire mesh size is matched to the large wavelengths of operation and varies from 10mm to 10 mm inside to 20 mm to 20 mm in the outer one third of the dish surface. As a result the efficiency varies from 60% below 1 GHz to 40% at 1.4 GHz. The dishes can give ae used to improve storage efficiency.

- **Scientific Achievements**: GMRT has made significant contributions to astrophysics, including detailed studies of pulsars, galactic structures, and cosmic radio sources. Its sensitivity and resolution are particularly valuable for understanding phenomena in the low-frequency radio spectrum.
- **Data Handling and Analysis**: The data collected by GMRT is immense and requires advanced processing techniques. The observatory utilizes sophisticated software and hardware for data storage, analysis, and visualization, enabling detailed scientific research and discoveries.
- **Technical Specifications**: The GMRT antennas are equipped with advanced receivers and electronics to ensure accurate signal detection and measurement. The design allows for high sensitivity and dynamic range, critical for observing faint celestial objects.

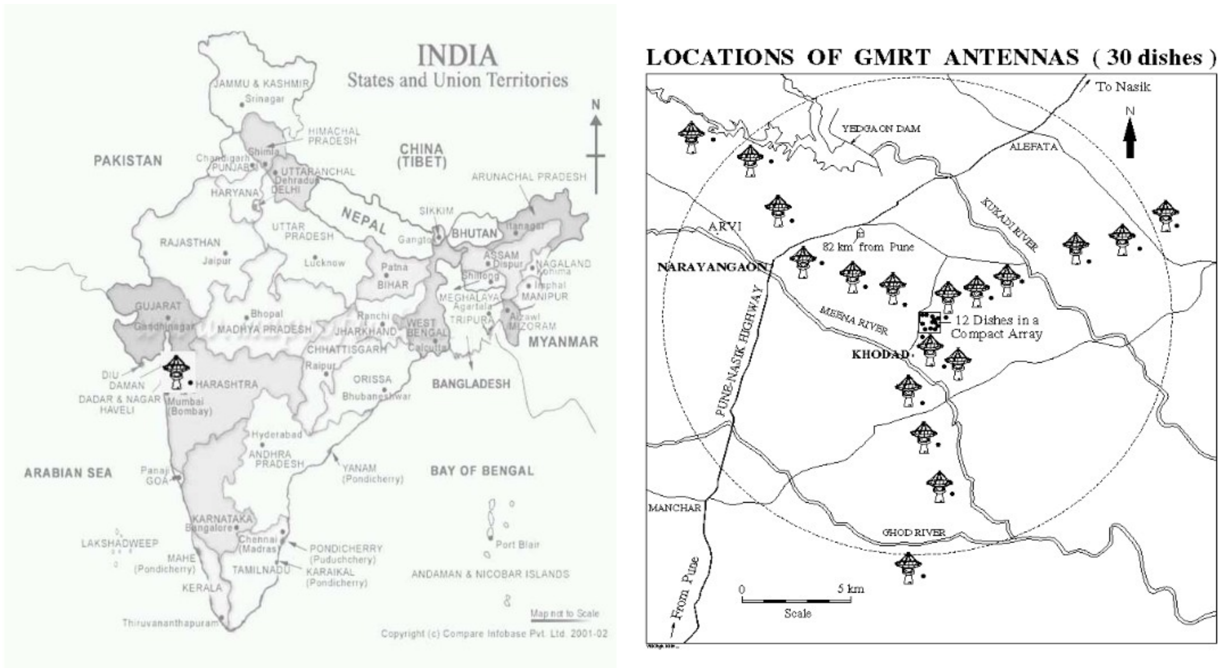


Fig 1. a) Location and (b) Configuration of the GMRT

Figure 2. One of the 30 GMRT Antennas

1.2 Necessity:

Self-learning, enhancement of knowledge, time saving.

1.3 Objectives:

To overcome drawbacks of redundancy of MATLAB , keeping free space, time saving and secured system. Easy to use. User-friendly GUI.

1.4 Theme:

The theme of the project, based on the project, revolves around **Real-Time Spectrum Analysis and Monitoring for Radio Frequency Interference (RFI) .**

1. **Real-Time Spectrum Monitoring**:
    - The application is designed to interface with a spectrum analyzer to monitor radio frequency spectra in real-time. It fetches data from the spectrum analyzer and updates the plot periodically, allowing users to observe live changes in the radio frequency environment.
2. **User Interface for Spectrum Analysis**:
    - The project includes a graphical user interface (GUI) using PyQt5 that provides controls for various spectrum analyzer settings (e.g., center frequency, span, resolution bandwidth). Users can interact with the spectrum analyzer through this interface, making it user-friendly for adjusting settings and visualizing data.
3. **Data Management and Visualization**:
    - The system allows users to save spectrum data and plot images. It also supports loading and saving settings configurations, providing a way to manage and document different analysis scenarios.
4. **Error Handling and Notifications**:
    - The application features robust error handling and notifications through QMessageBox, ensuring that users are informed about connection issues with the spectrum analyzer or any errors during data processing.
5. **Customization and Flexibility**:
    - Users can customize spectrum analyzer parameters through the GUI and save these settings for future use. The ability to reset parameters to default values and update them dynamically reflects the flexibility of the application.

1.5 Organization:

The report divided into Nine Chapters. In Chapter 2, it is all discussion about the Existing System which focuses on the hardware and software requirements. In chapter 3, 4 and 5 the discussion about the proposed system and requirement specification is mentioned. In chapter 6, Workflow Diagram of the user interface is given in simplest manner. In Chapter 7, Advantages of PyQt5 over MATPLOTLIB are given. Chapter 9 refers to the Conclusion and Future Work for the proposed system. At last 9th chapter refers to the References which I had used for taking references for relevant content.

----------Chapter 1 ends here----------

CHAPTER 2

EXISTING SYSTEM

* The Omnidirectional RFI monitoring system of GMRT.

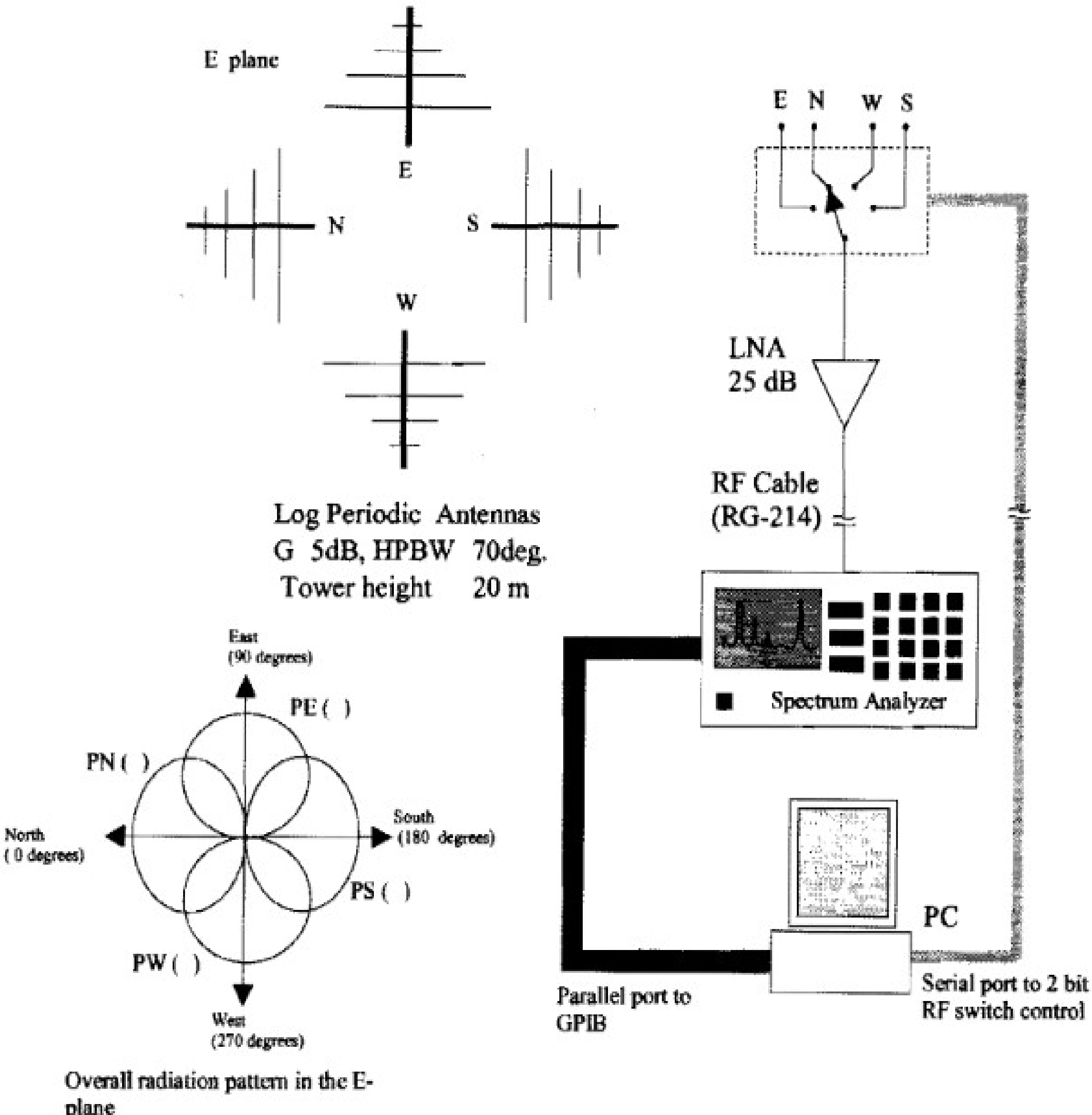
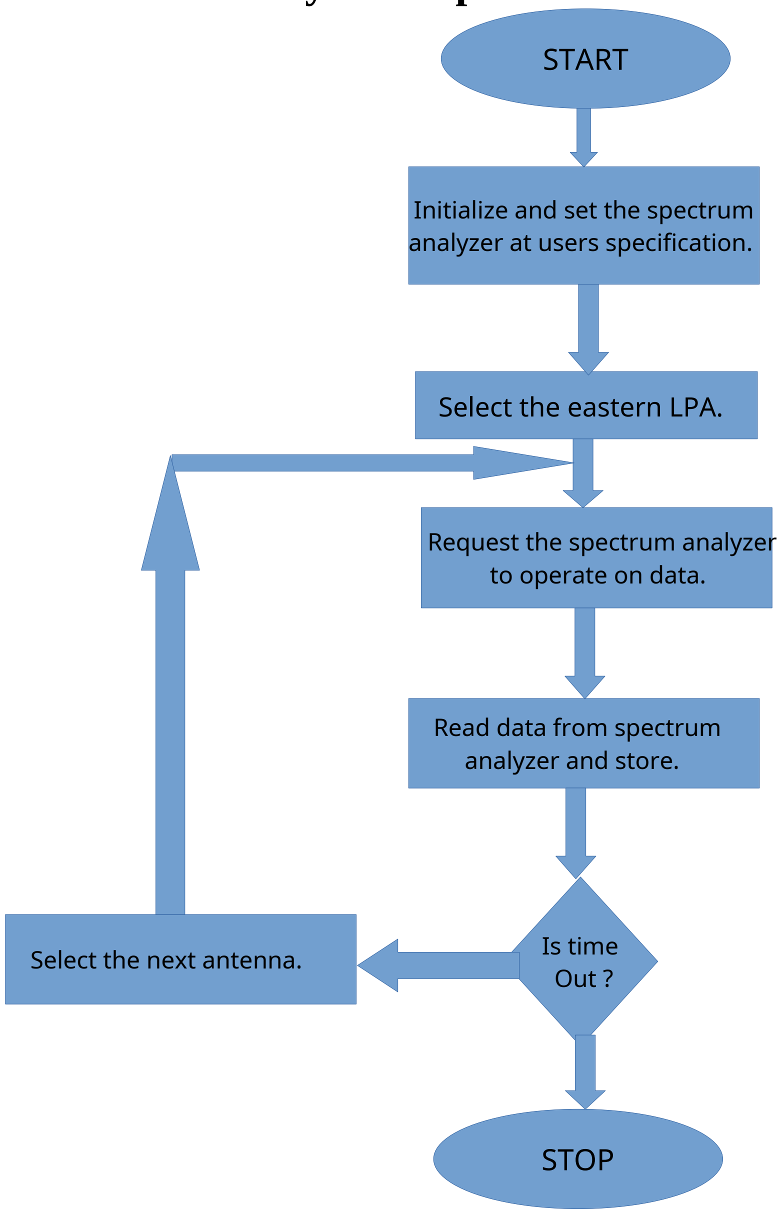
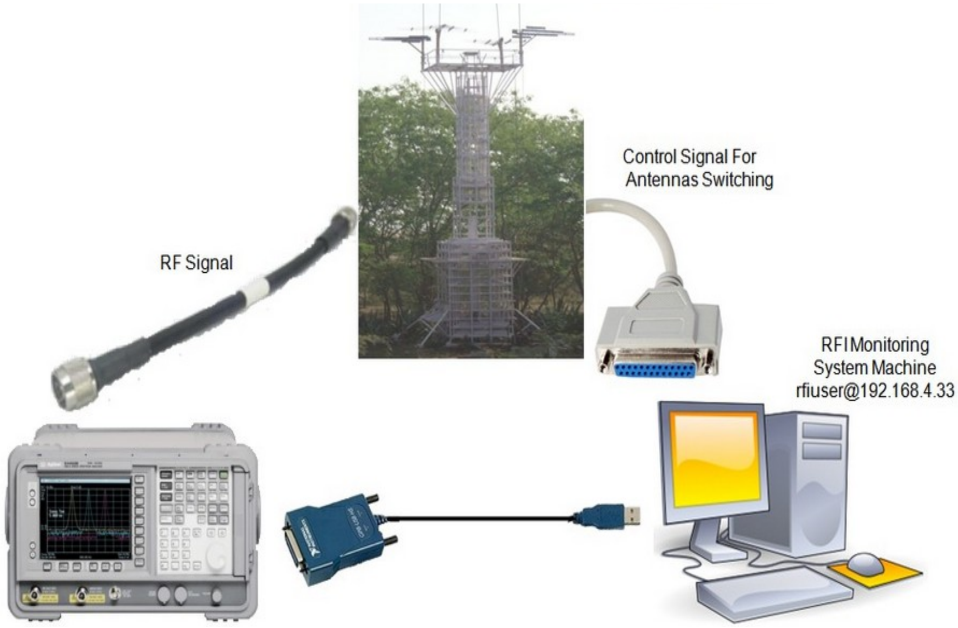


Fig.1: The Omnidirectional RFI monitoring system of GMRT

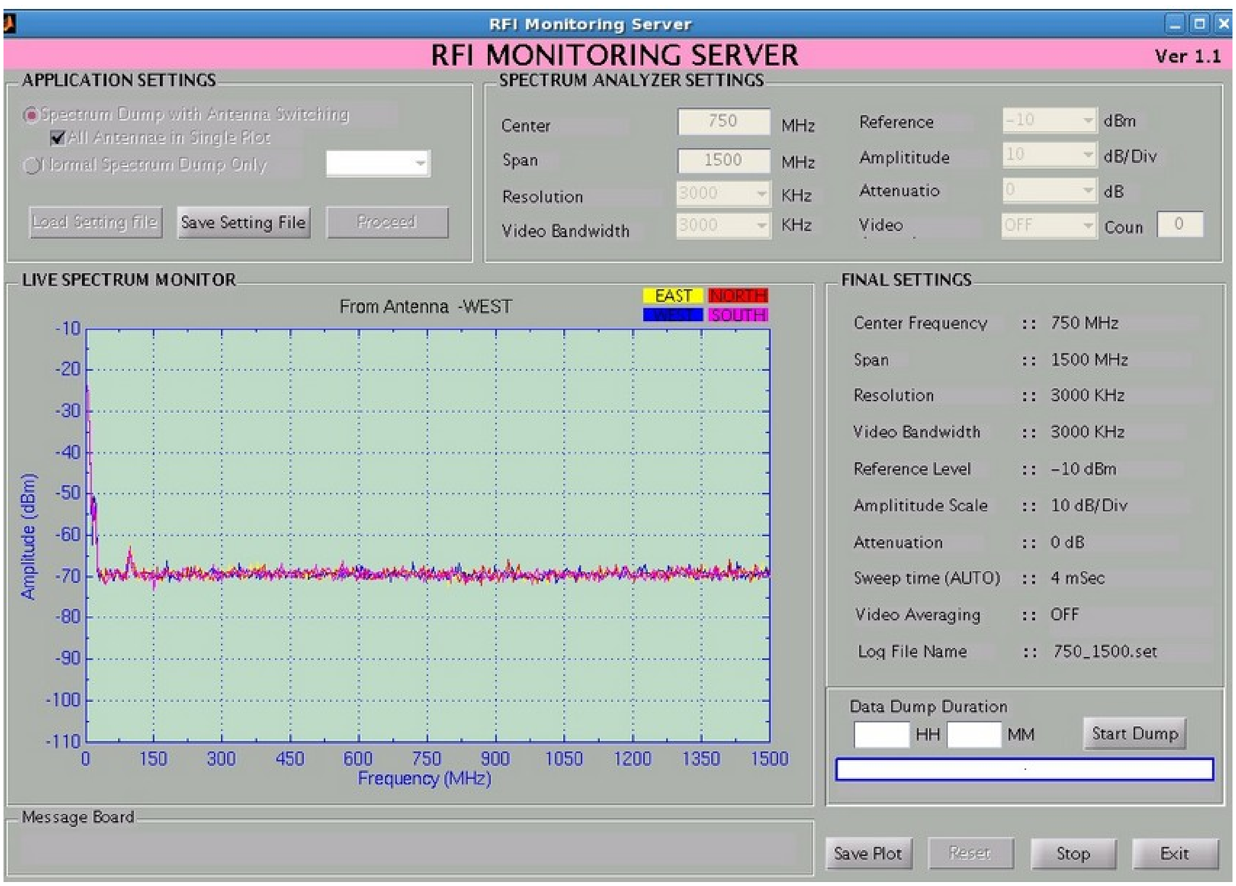
The above fig 1. depicts the diagram of the omnidirectional RFI monitoring system of GMRT . The 4 antennas are placed in the E-plane in orthogonal directions, viz. EAST, WEST, NORTH and SOUTH. The computer selects the antenna's sequence of EAST-WEST-NORTH- SOUTH using the SP4T switch. After each selection the incoming RFI power is amplified (LNA) followed by spectrum analysis and finally recorded in the corresponding direction file in the computer. This cyclic process is continued throughout the observation.

* The Flowchart for System Operation.

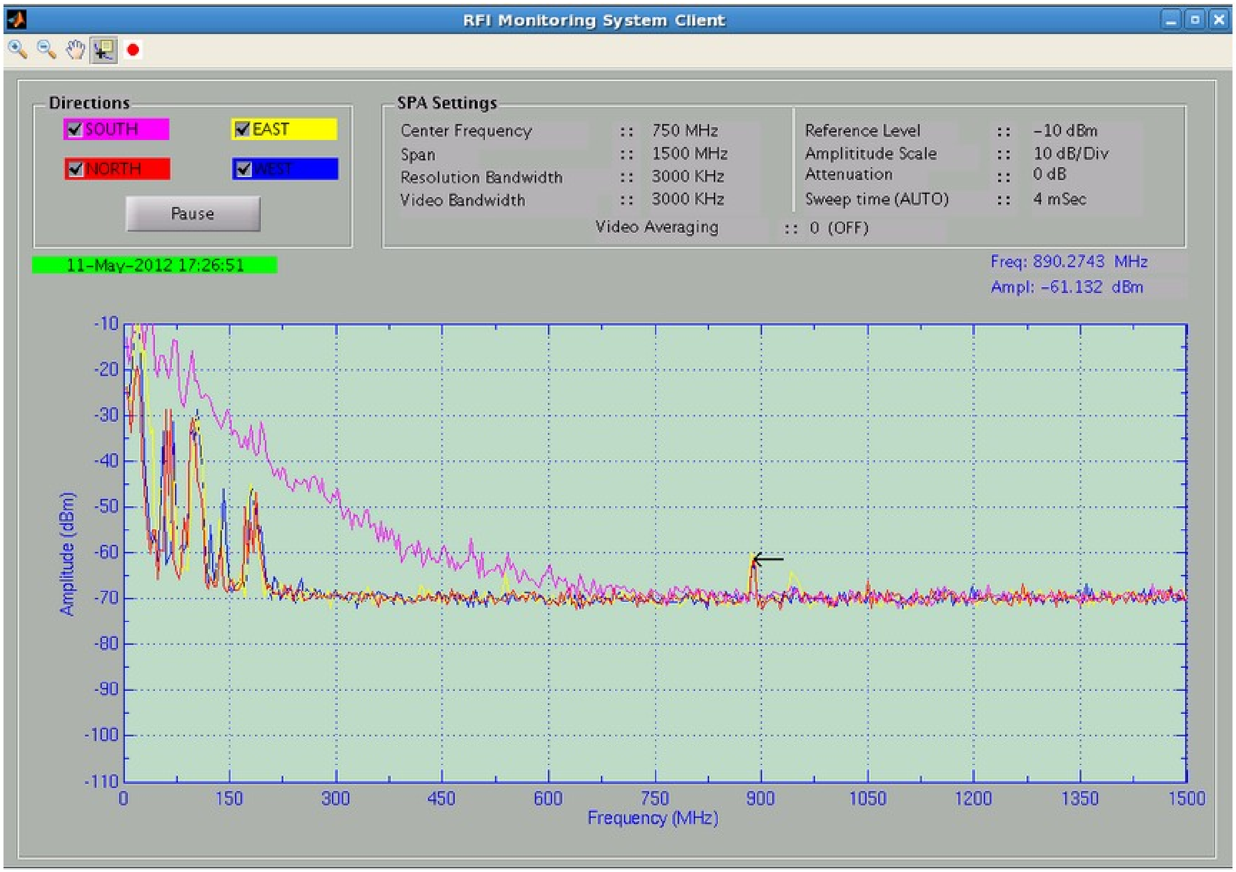




* Software Requirements of the Existing System



*** GUI of the Existing System.**



*** GUI of the Existing System.**

----------Chapter 2 ends here----------

CHAPTER 3

PURPOSE OF PROPOSED SYSTEM

1. **Data Visualization and Analysis:**
   - The application is designed not only to visualize real-time spectrum data but also to allow for detailed analysis through graphical representation.
   - Users can observe changes in spectrum data dynamically and make real-time decisions based on the visual output.

2. **Parameter Customization:**
   - Users have the ability to customize a wide range of spectrum analyzer settings, including frequency center, span, RBW, VBW, reference level, and attenuation.
   - This flexibility allows users to adapt the spectrum analyzer to various testing and measurement scenarios.

3. **Advanced Data Logging:**
   - The system logs detailed spectrum data to a file with timestamped entries.
   - This logging capability supports historical analysis and tracking of spectral measurements over time, which is crucial for monitoring trends and detecting anomalies.

4. **User-Friendly Interface:**
   - The application features a user-friendly graphical interface that simplifies interaction with complex spectrum analyzer settings.
   - Controls are organized logically, and visual feedback is provided to guide users through various tasks.

5. **Error Handling and Notifications:**
   - Built-in error handling mechanisms ensure that users are informed about issues, such as connection failures or errors in data processing.
   - This helps in troubleshooting and maintaining the reliability of the system.

6. **Settings Management:**
   - Users can easily manage settings by loading from or saving to configuration files.

- This functionality allows for quick reconfiguration and replication of settings across different sessions or systems.

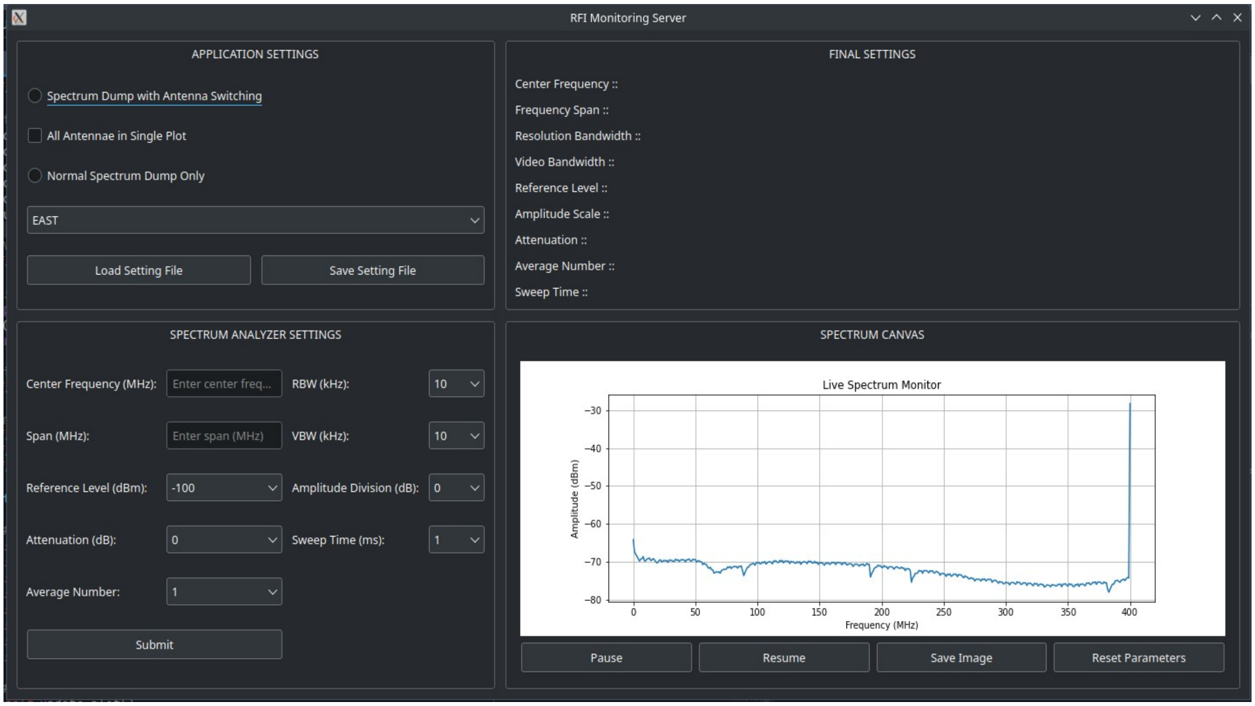
7. **Real-Time Data Update:**
    - The application supports real-time updates of the spectrum plot, with periodic data retrieval and plot refresh.
    - This ensures that users have the most current view of the spectrum data.

8. **Graphical Export:**
    - The ability to save the spectrum plot as an image file facilitates reporting and documentation.
    - Users can capture and share visual representations of their measurements for analysis or presentation.

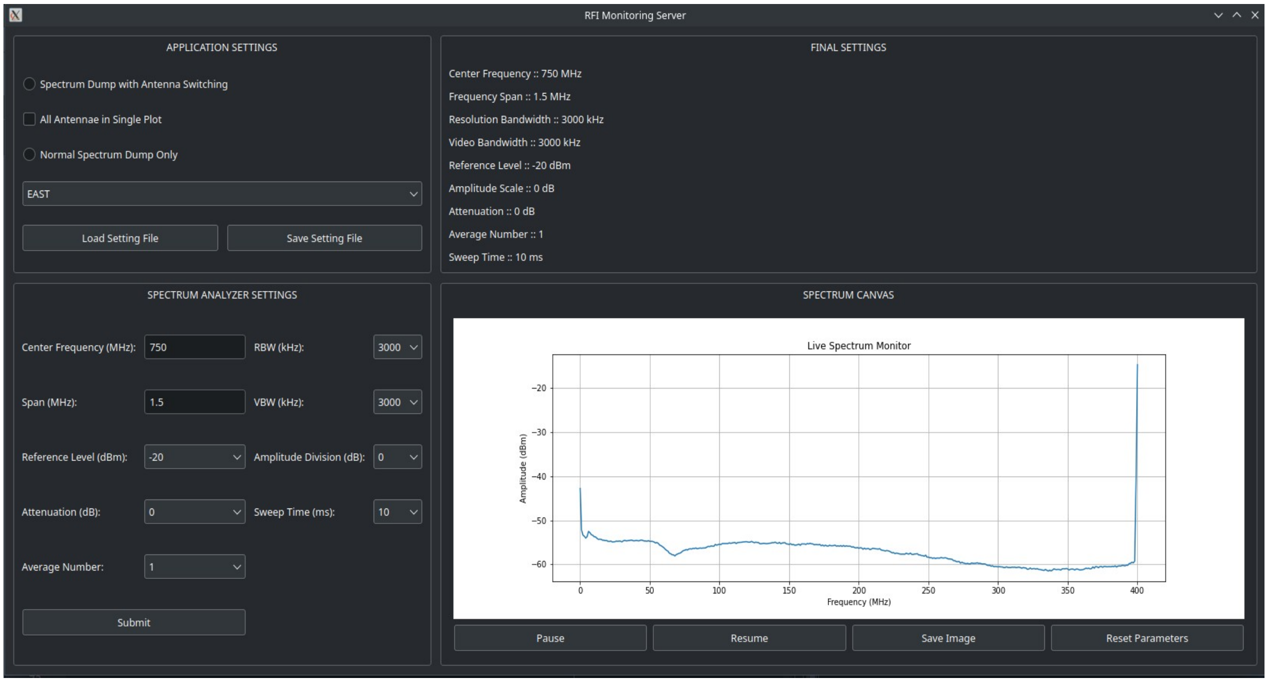
9. **System Reset Capability:**
    - The system provides a reset function to return the spectrum analyzer to its default settings.
    - This is useful for quickly restoring standard configurations or starting fresh measurements.

10. **Customizable Measurement Settings:**
    - Through the application, users can configure and adjust various measurement parameters such as sweep time and averaging number, providing control over the precision and duration of spectrum measurements.

The proposed system offers a comprehensive tool for spectrum analysis with **real-time visualization, customizable settings, robust data management, and user-friendly interactions.** It supports both immediate analysis and long-term tracking of spectrum data, making it suitable for various applications in research, testing, and monitoring.



*** User Interface for the Proposed System**



*** GUI with entered values and hence graph is changed respectively.**

----------Chapter 3 ends here----------

CHAPTER 4

SYSTEM REQUIREMENTS

A. Hardware Requirements:-

- Processor : 8 × Intel® Core™ i7-2600 CPU @ 3.40GHz
- Monitor : Color Monitor.
- Mesa Intel® HD Graphics 2000
- Agilent N9310A RF Signal Generator 9 kHz – 3.0 GHz.

- Rohde & Schwarz FSV Signal Analyzer 10Hz ---- 3.6GHz.

B. Software Requirements:-

- Operating System: LINUX
- KDE Plasma version: 5.27.11
- Development Tool: PYTHON, PyQt5 Library, PYVISA Library.
- Spyder (Python 3.12)

----------Chapter 4 ends here----------

CHAPTER 5

PROPOSED SYSTEM

The proposed system is a graphical user interface (GUI) application designed to interface with a spectrum analyzer for real-time monitoring and analysis of spectrum data. Here's a breakdown of its purpose and functionality:

Purpose:

1. **Real-Time Spectrum Monitoring:**
    - The primary purpose of this application is to provide a real-time graphical display of spectrum data obtained from a spectrum analyzer.
    - The spectrum analyzer is used to measure the amplitude of signals across a range of frequencies.
2. **Spectrum Analyzer Control:**
    - The application allows users to configure and control various settings of the spectrum analyzer, such as frequency center, frequency span, resolution bandwidth (RBW), video bandwidth (VBW), reference level, and attenuation.
    - These settings can be adjusted through the GUI to tailor the measurements according to specific requirements.
3. **Data Logging:**
    - The system logs spectrum data to a file (.rfi format) periodically.
    - This includes timestamped data on the frequency spectrum, which can be used for further analysis or documentation.
4. **Image Saving:**
    - Users can save the plotted spectrum data as an image file (PNG, JPEG) for reporting or documentation purposes.
5. **Parameter Management:**
    - The application provides options to reset the spectrum analyzer to default settings, load settings from a file, and save current settings to a file. This helps in managing and reusing configurations easily.

Functionality:

1. **User Interface:**
    - **Spectrum Canvas:** Displays live spectrum data in a plot using Matplotlib. The plot updates at regular intervals (every second) to reflect the latest data from the spectrum analyzer.

- **Control Buttons:** Includes buttons for pausing and resuming updates, saving the plot image, and resetting parameters.

- **Settings Management:** Provides options to load and save settings from/to files and adjust various parameters like frequency center, span, RBW, VBW, reference level, and more.

- **Final Settings Display:** Shows the currently applied settings on the GUI for easy reference.

2. **Spectrum Analyzer Integration:**
    - Utilizes the PyVISA library to communicate with the spectrum analyzer via TCP/IP.
    - The application sends commands to configure the analyzer and queries it for data.
3. **Error Handling:**
    - Includes mechanisms for error handling and user notifications via message boxes.
    - This ensures users are informed if there are issues with connecting to the analyzer or updating the plot.
4. **Customization:**
    - Allows users to select and apply different settings for the spectrum analyzer through a form-based interface.
    - These settings are applied immediately and reflected in the live plot.

In summary, the system is designed to monitor, control, and visualize the performance of a spectrum analyzer, providing users with an interactive tool for spectrum analysis and data management.

----------Chapter 5 ends here----------

CHAPTER 6

DESIGNS AND WORKFLOW DIAGRAM

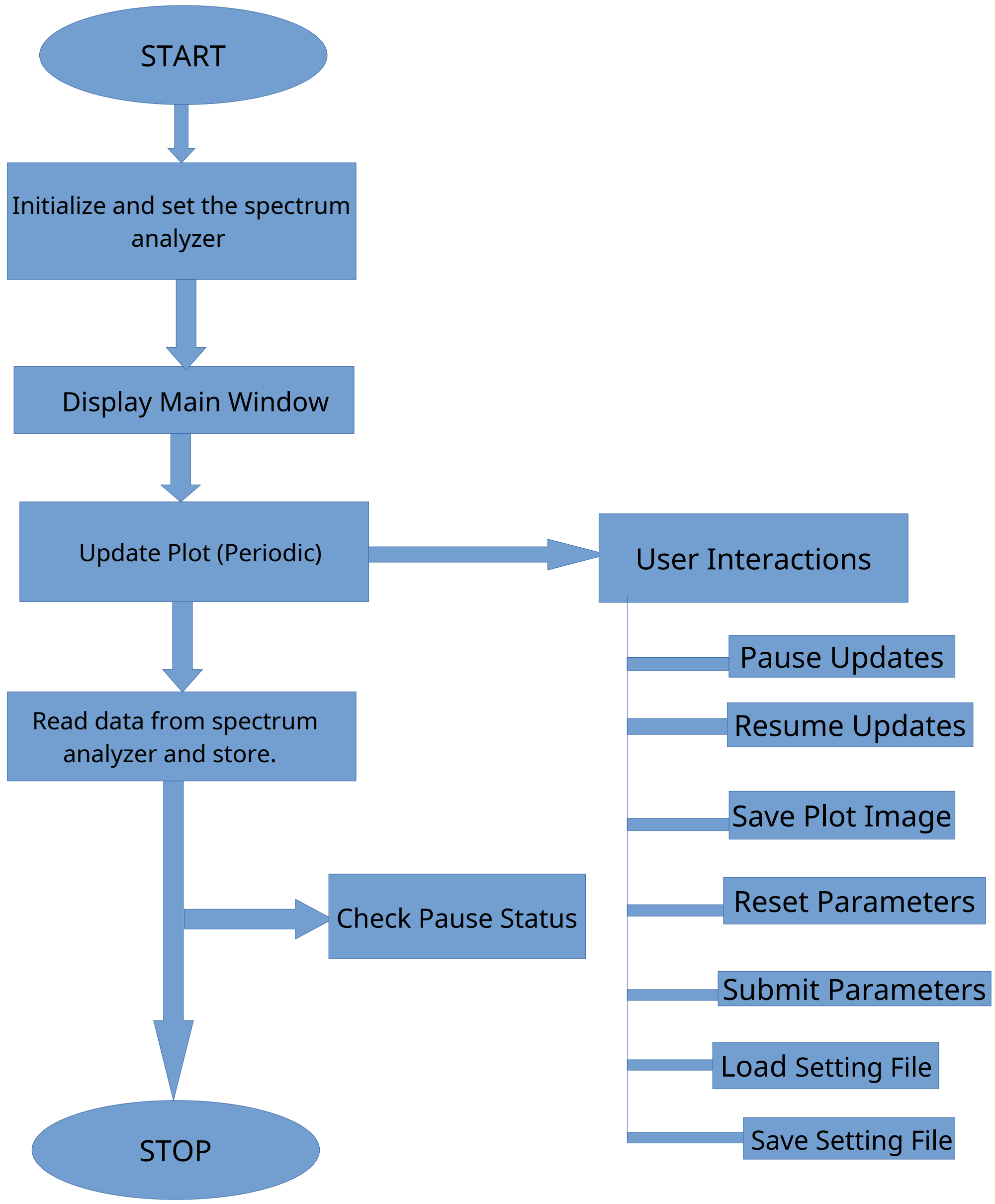


fig. Workflow Diagram for the Proposed System

The diagram represents the flow of the spectrum analyzer application. It begins with the **Start** and proceeds to **Initialize Application**, setting up the user interface and connecting to the spectrum analyzer. Once initialized, the **Display Main Window** shows the application to the user. The system then enters a loop where it **Update Plot (Periodic)** to refresh the spectrum data continuously. During this period, **User Interactions** are handled, allowing users to pause or resume updates, save plot images, reset parameters, and manage settings files. Finally, the process concludes with the **End** of the application.

----------Chapter 6 ends here----------

CHAPTER 7

ADVANTAGES

Using PyQt5 over MATLAB for developing a spectrum analysis and monitoring application presents several advantages. The key benefits of PyQt5:

1. Cost and Licensing

- **PyQt5**: PyQt5 is open-source and free to use under the GPL license. It provides a cost-effective solution for developing and distributing applications.

- **MATLAB**: MATLAB is a commercial product that requires a license, which can be expensive. This may not be ideal for all users or organizations, especially those with budget constraints.

2. Integration with Other Libraries

- **PyQt5**: PyQt5 integrates well with other Python libraries, such as MATPLOTLIB for plotting and PYVISA for instrument control. This flexibility allows you to leverage a wide range of tools and libraries in Python, enhancing the functionality of your application.

- **MATLAB**: While MATLAB has its own extensive set of toolboxes, integrating with external libraries or systems can be more complex and may require additional MATLAB toolboxes or custom interfaces.

3. Flexibility and Customization

- **PyQt5**: PyQt5 offers greater flexibility and customization options for designing the user interface. You can create highly customized widgets and layouts, and integrate with various Python modules for specialized functionalities.

- **MATLAB**: MATLAB’s GUI development environment (GUIDE) offers built-in components but can be less flexible compared to PyQt5. Customizing MATLAB GUIs beyond the standard components may require more effort and programming.

4. Cross-Platform Compatibility

- **PyQt5**: PyQt5 is cross-platform and works on Windows, macOS, and Linux. This allows you to develop applications that run consistently across different operating systems.

- **MATLAB**: MATLAB is also cross-platform, but its performance and features might vary slightly between different operating systems. Additionally, MATLAB applications often require MATLAB Runtime to be installed on the target system.

5. Open-Source Ecosystem

- **PyQt5**: Being part of the Python ecosystem, PyQt5 benefits from a vast range of open-source libraries and tools. This open-source environment fosters community support, frequent updates, and a wealth of resources.

- **MATLAB**: MATLAB's ecosystem is more closed and relies on proprietary toolboxes. While MATLAB offers extensive built-in functionalities, expanding capabilities beyond these toolboxes can be limited and costly.

6. Development and Deployment

- **PyQt5**: Python applications, including those developed with PyQt5, can be easily packaged and distributed as standalone executables using tools like PyInstaller or cx_Freeze. This simplifies deployment to end-users without requiring them to have Python installed.

- **MATLAB**: MATLAB applications often require MATLAB Runtime or a full MATLAB installation for deployment, which can complicate the deployment process and add additional overhead for end-users.

7. Community and Support

- **PyQt5**: PyQt5 benefits from a large and active Python community. There are extensive resources, forums, and documentation available to help with development and troubleshooting.

- **MATLAB**: MATLAB also has a strong community and official support from MathWorks, but the resources are more specialized to MATLAB's ecosystem.

PyQt5 offers a cost-effective, flexible, and highly customizable alternative to MATLAB for developing GUI applications. It benefits from open-source libraries, cross-platform compatibility, and an active community, making it a powerful choice for integrating with various Python-based tools and technologies.

----------Chapter 7 ends here----------

CHAPTER 8

CONCLUSION AND FUTURE WORK

The provided code implements a spectrum analyzer application using PyQt5 for the graphical user interface (GUI) and PYVISA for interfacing with a spectrum analyzer. The application allows users to:

- Connect to and control a spectrum analyzer.
- Configure spectrum analyzer settings (ex. frequency, span, resolution bandwidth).
- Monitor and visualize spectrum data in real time.
- Save plots and settings, and reset parameters to default.

The application is structured with a clean separation of concerns, with distinct classes handling different parts of the functionality:

- Spectrum Analyzer App: Manages the spectrum analyzer interface and plotting.
- Form: Provides the main GUI layout for user interactions, including settings management and display.

This design ensures that the application is modular and maintainable, with a focus on providing an intuitive user experience.

Future Scopes

1. **Enhanced Data Analysis**
    - **Feature**: Incorporate advanced data analysis features such as statistical analysis, data smoothing, or filtering.
    - **Benefit**: Allows users to derive more insights from the spectrum data beyond basic visualization.
2. **Extended Instrument Control**
    - **Feature**: Support for additional spectrum analyzer commands and functionalities, such as automated sweeps or advanced measurement modes.
    - **Benefit**: Expands the range of applications and provides more detailed control over measurements.
3. **User Customization**
    - **Feature**: Implement user profiles or configuration presets for saving and loading different setups.
    - **Benefit**: Enhances user experience by allowing quick access to frequently used settings.
4. **Real-Time Data Streaming**

- **Feature**: Develop real-time data streaming capabilities, including live data export or integration with data logging systems.
- **Benefit**: Facilitates continuous monitoring and logging of spectrum data for long-term analysis.

5. **Improved Error Handling and Diagnostics**

   - **Feature**: Enhance error handling and diagnostics, including detailed logs and troubleshooting guides.
   - **Benefit**: Provides better support for users by making it easier to identify and resolve issues.

6. **Graphical Enhancements**

   - **Feature**: Add more advanced graphical features such as multiple plots, zooming, panning, or interactive elements.
   - **Benefit**: Improves data visualization and user interaction with the plots.

7. **Cross-Platform Deployment**

   - **Feature**: Create and test deployment packages for various platforms, ensuring consistent performance across different operating systems.
   - **Benefit**: Makes the application accessible to a wider audience with different operating systems.

8. **Integration with Other Systems**

   - **Feature**: Integrate the application with other scientific or data analysis tools, such as databases or cloud services.
   - **Benefit**: Enhances the application's utility by connecting it with other parts of a data analysis pipeline.

9. **User Interface Improvements**

   - **Feature**: Refine the GUI with more intuitive controls, better layout organization, and additional help or documentation.
   - **Benefit**: Increases usability and reduces the learning curve for new users.

----------Chapter 8 ends here----------

CHAPTER 9

REFERENCES

[1] Tata Institute of Fundamental Research. *Handle/2301/321*. n.d. http://library.ncra.tifr.res.in:8080/jspui/handle/2301/321.

[2] Spyder Project. (n.d.). *Spyder*. GitHub. Retrieved August 6, 2024, from https://github.com/spyder-ide/spyder

[3] Rohde & Schwarz. (2021). *FSW user manual* (Rev. 5.3). Retrieved August 6, 2024, from https://scdn.rohde-schwarz.com/ur/pws/dl_downloads/pdm/cl_manuals/user_manual/1173_9411_01/FSW_UserManual_en_53.pdf

[4] KDE. (n.d.). *Tutorials*. KDE User Base. Retrieved August 6, 2024, from https://userbase.kde.org/Tutorials

[5] TutorialsPoint. (n.d.). *Ubuntu tutorial*. Retrieved August 6, 2024, from https://www.tutorialspoint.com/ubuntu/index.htm

[6] PyVISA Documentation. (n.d.). *PyVISA read the docs*. Retrieved August 7, 2024, from https://pyvisa.readthedocs.io/

[7] Riverbank Computing. (n.d.). *PyQt: Introduction*. Retrieved August 7, 2024, from https://riverbankcomputing.com/software/pyqt/intro

[8] Matplotlib. (n.d.). *Matplotlib: Python plotting*. Retrieved August 7, 2024, from https://matplotlib.org/

[9] W3Schools. (n.d.). *Python JSON*. Retrieved August 7, 2024, from https://www.w3schools.com/python/python_json.asp

[10]Python Software Foundation. (n.d.). *sys — System-specific parameters and functions*. Retrieved August 7, 2024, from https://docs.python.org/3/library/sys.html

----------End of the Report----------