Sri Lanka Institute of Information Technology

R24-002

FERNANDO R.D.S.A.

IT21011948

Information System Engineering

Table of Contents

Table of Figures

Project Component and Current Status

Component: Risk-Adjusted Time Forecast

Progress

- **Data Gathering from Kaggle:** You have sourced the necessary data from Kaggle, a platform known for its rich datasets, particularly in machine learning and AI. This data is crucial for training and testing your time duration prediction model. The dataset is being used to extract patterns and trends that can aid in making accurate time forecasts for various stages of a Software Development Life Cycle (SDLC).
- **Backend Development (Completed):** The backend architecture for time duration prediction is fully implemented, utilizing a Gradient Boost machine learning model. This advanced algorithm was chosen for its ability to enhance prediction accuracy by iteratively improving upon previous models. Python, in combination with Jupyter Notebooks, was used for coding and managing the development process. This phase likely involved tasks like data preprocessing, feature selection, model training, and evaluation to ensure optimal predictive performance.
- **Frontend Development (In Progress):** The frontend for time duration prediction is currently under development. This user interface will be responsible for allowing users to interact with the system and access the predictions generated by the backend. It may involve creating intuitive and accessible ways for users to input data or view results. The design might focus on clarity and ease of use, ensuring that even non-technical users can make sense of the time predictions provided by the backend system.

Future Progress

- **Developing Risk-Adjusted Time Forecast According to SDLC Phases:** The next phase involves creating a Risk-Adjusted Time Forecast model that aligns with the different stages of the Software Development Life Cycle (SDLC). This model will account for potential risks that might occur during each phase of the SDLC, such as design delays, testing issues, or implementation challenges. The goal is to factor in these risks to provide more accurate and realistic time estimates, ensuring that the forecast reflects the complexities and uncertainties of real-world development projects.
- **Developing a Time Forecasting Model Adjusted to Mitigation Plans:** In addition to risk adjustment, another model will be developed that considers the mitigation plans identified during each SDLC phase. This involves forecasting time durations by considering the measures put in place to manage risks. For instance, if a mitigation plan is designed to handle potential delays in the testing phase, the model will adjust the predicted time based on the efficiency and impact of those plans, leading to a more dynamic and responsive time prediction.

```
Linear Regression Performance:
RMSE: 24.59
R2 Score: 0.76
MAE: 20.18
/opt/conda/lib/python3.10/site-packages/sklearn/preprocessing/_encoders.py:868: FutureWarning: `sparse` was renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you leave `sparse` to its default value.
  warnings.warn(
/opt/conda/lib/python3.10/site-packages/sklearn/preprocessing/_encoders.py:868: FutureWarning: `sparse` was renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you leave `sparse` to its default value.
  warnings.warn(

Random Forest Performance:
RMSE: 6.99
R2 Score: 0.98
MAE: 4.40
/opt/conda/lib/python3.10/site-packages/sklearn/preprocessing/_encoders.py:868: FutureWarning: `sparse` was renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you leave `sparse` to its default value.
  warnings.warn(

Gradient Boosting Performance:
RMSE: 4.78
R2 Score: 0.99
MAE: 3.34

SVR Performance:
RMSE: 49.22
R2 Score: 0.04
MAE: 43.27
/opt/conda/lib/python3.10/site-packages/sklearn/preprocessing/_encoders.py:868: FutureWarning: `sparse` was renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you leave `sparse` to its default value.
  warnings.warn(
```

Figure 1 Machine Learning modeling.

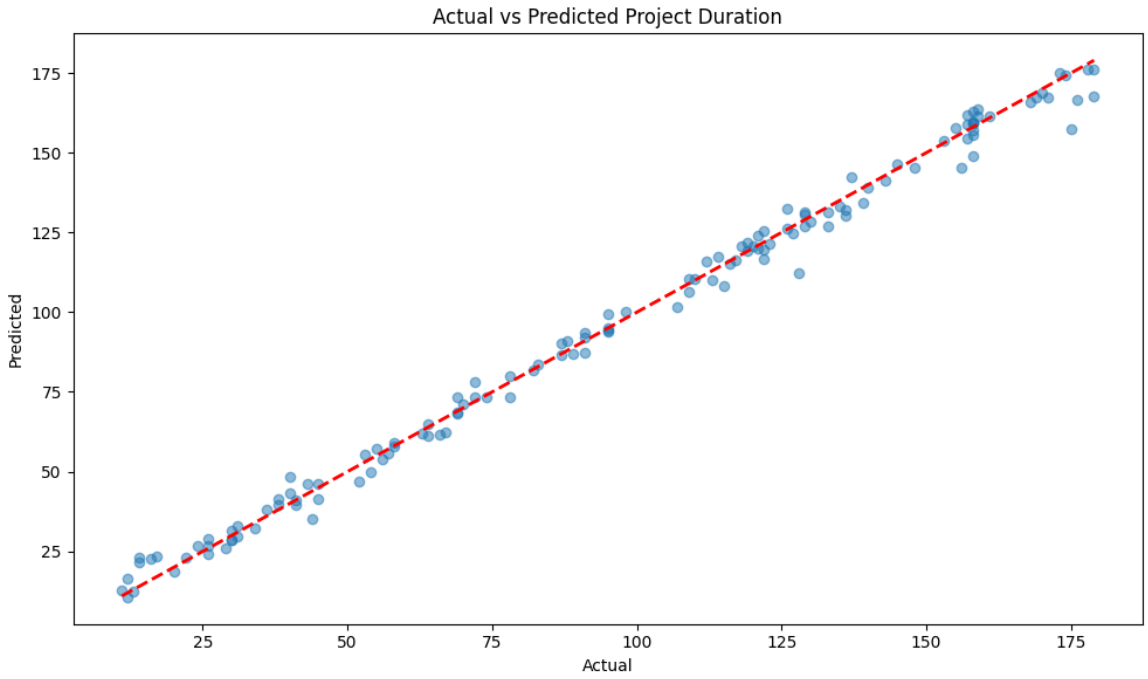


Figure 2- Actual vs predicted graph

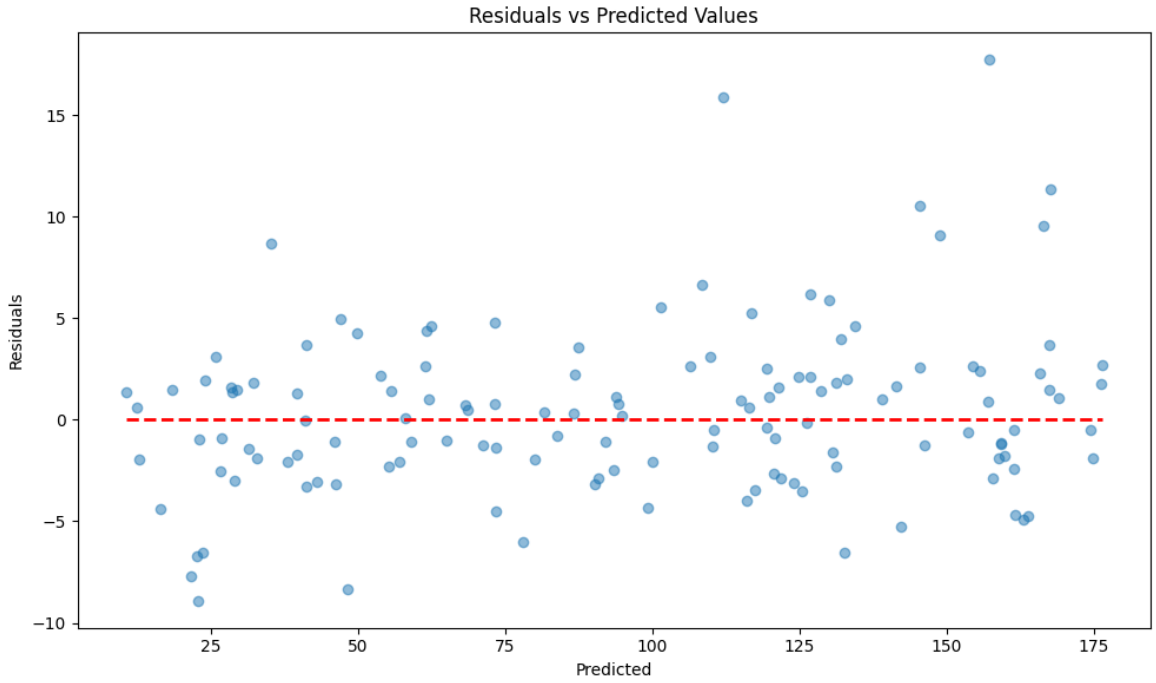


Figure 3-Residual vs predicted graph

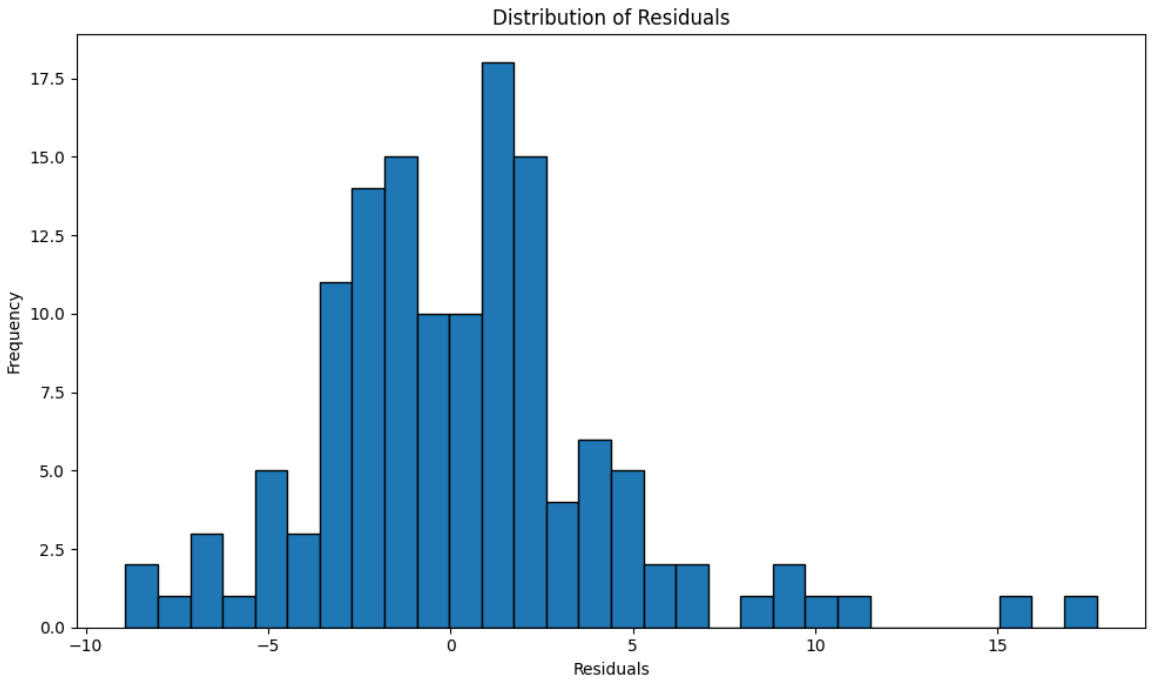


Figure 4- Distribution of residuals

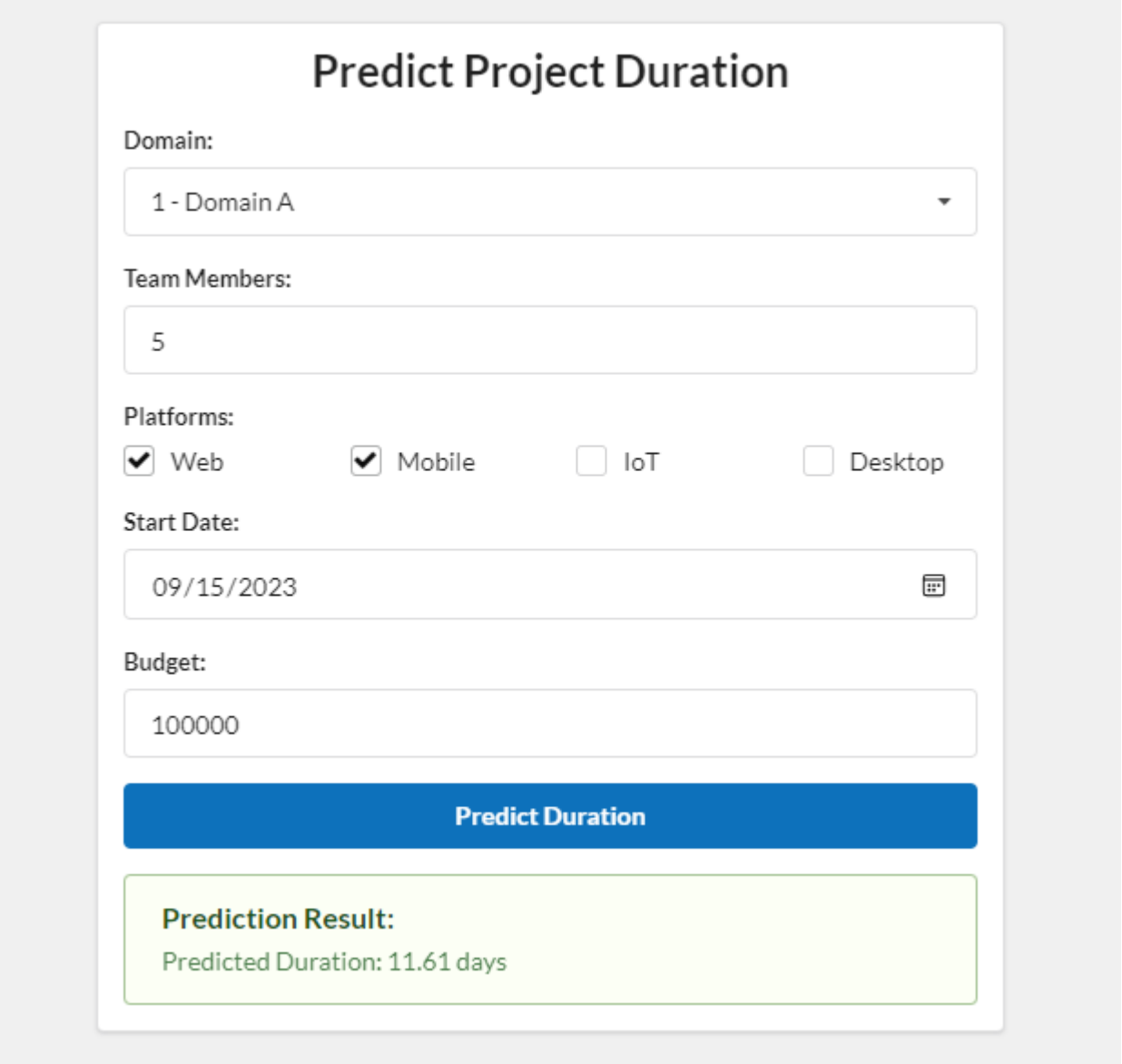


Figure 5-Prediction results

Team Communication

The team chose Microsoft Teams as their primary communication channel, forming a dedicated Team with all four group members. We also used Zoom to communicate with supervisors, provide updates, and receive comments on the project's progress. Regular team conversations were arranged to discuss, share knowledge, and plan.

The crew also used WhatsApp as an additional tool to stay in constant communication with their supervisors. This enabled timely updates and cooperation between the supervisor and co-supervisor, ensuring that everyone was informed and on the same page throughout the project.

Teams Calls with the Research Team

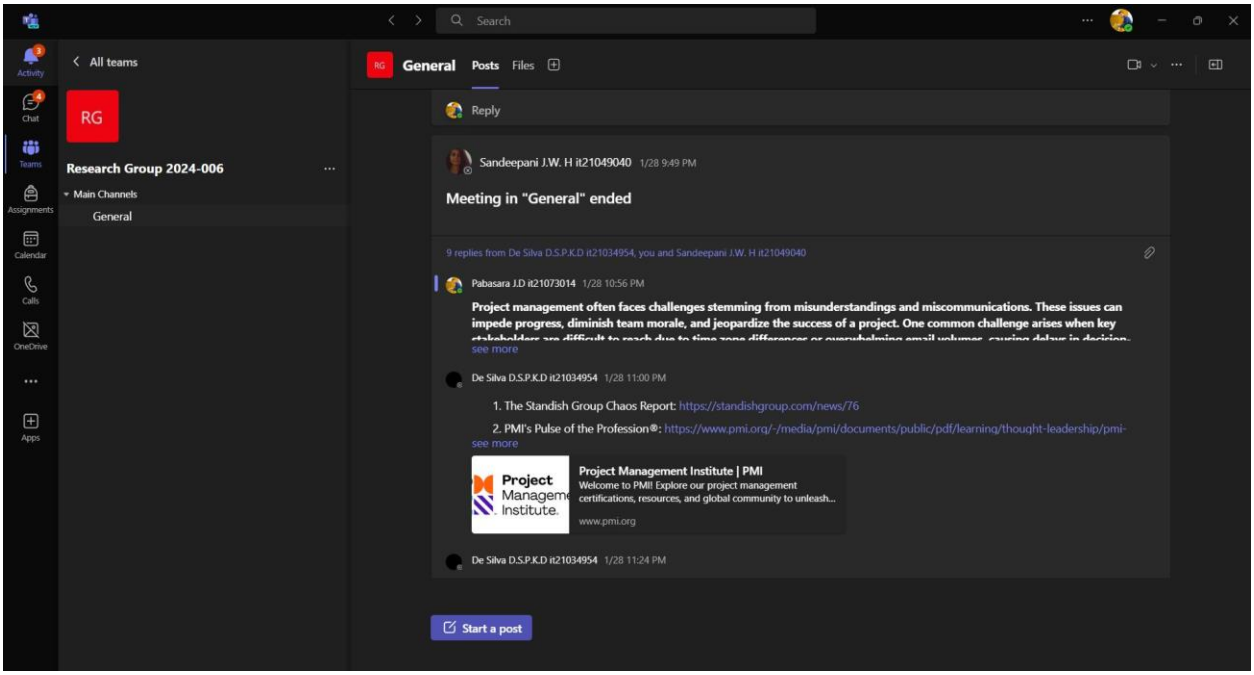


Figure 6 - Overview of Team Calls and Communication

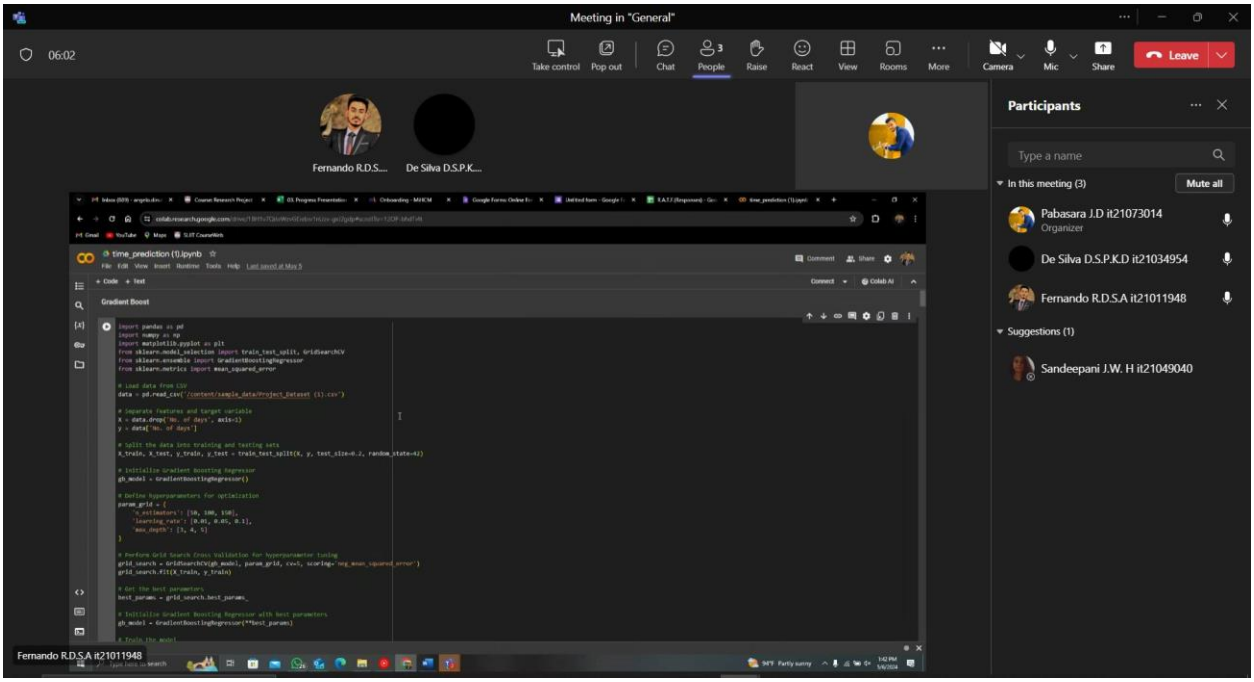


Figure 7 - Team Member Teams Calls (Example 01)

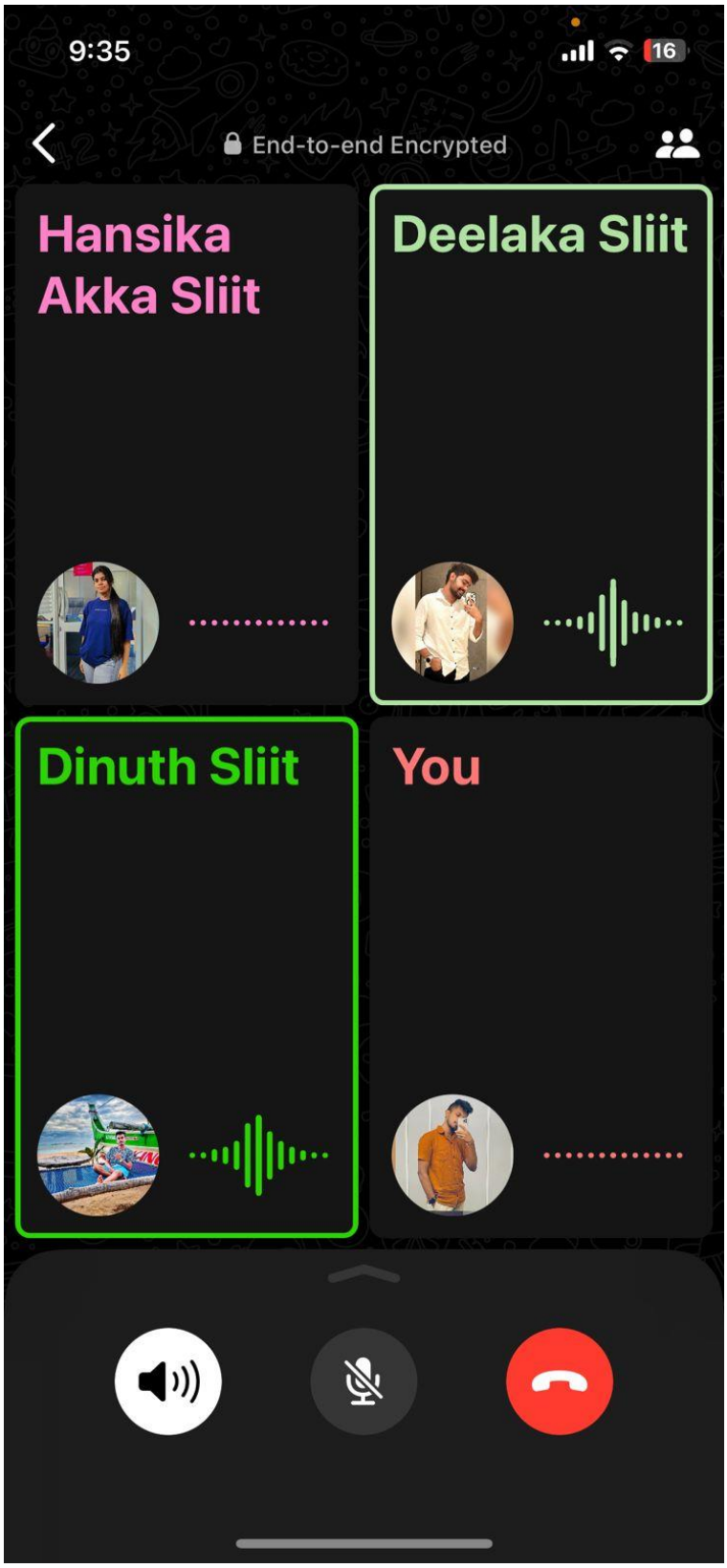


Figure 14 - Team Member Teams Calls (Example 02)

Online Calls with Supervisors

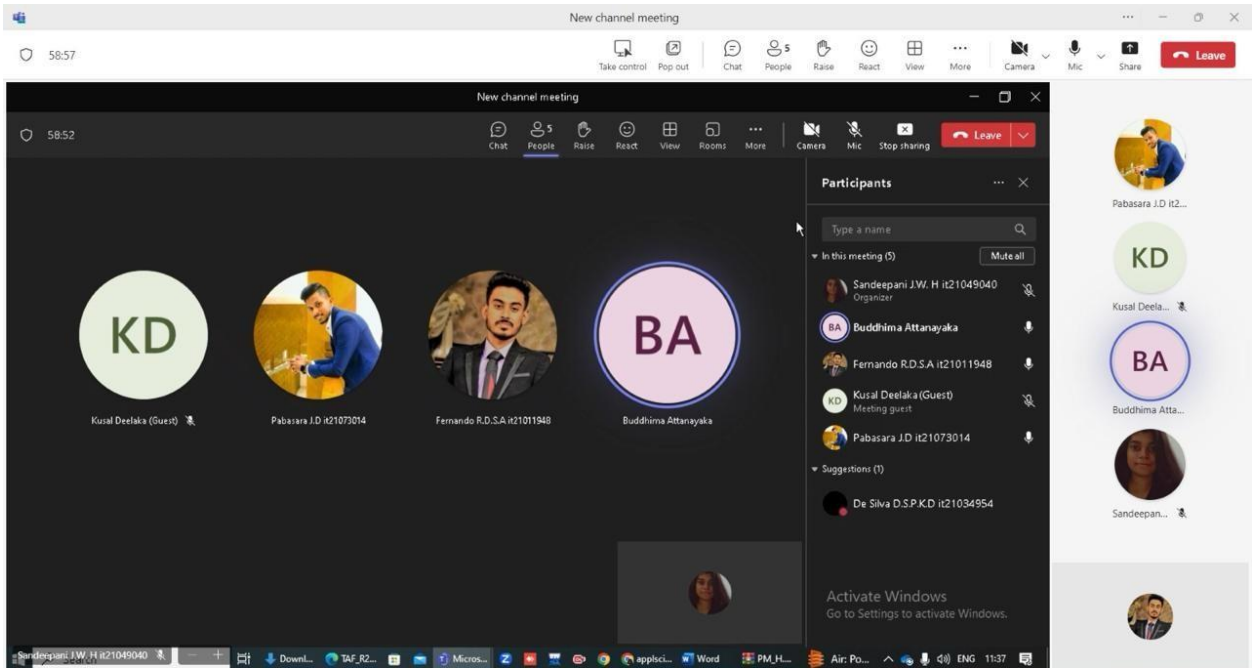


Figure 9 - Teams Meeting with the Co-Supervisor (Example 01)

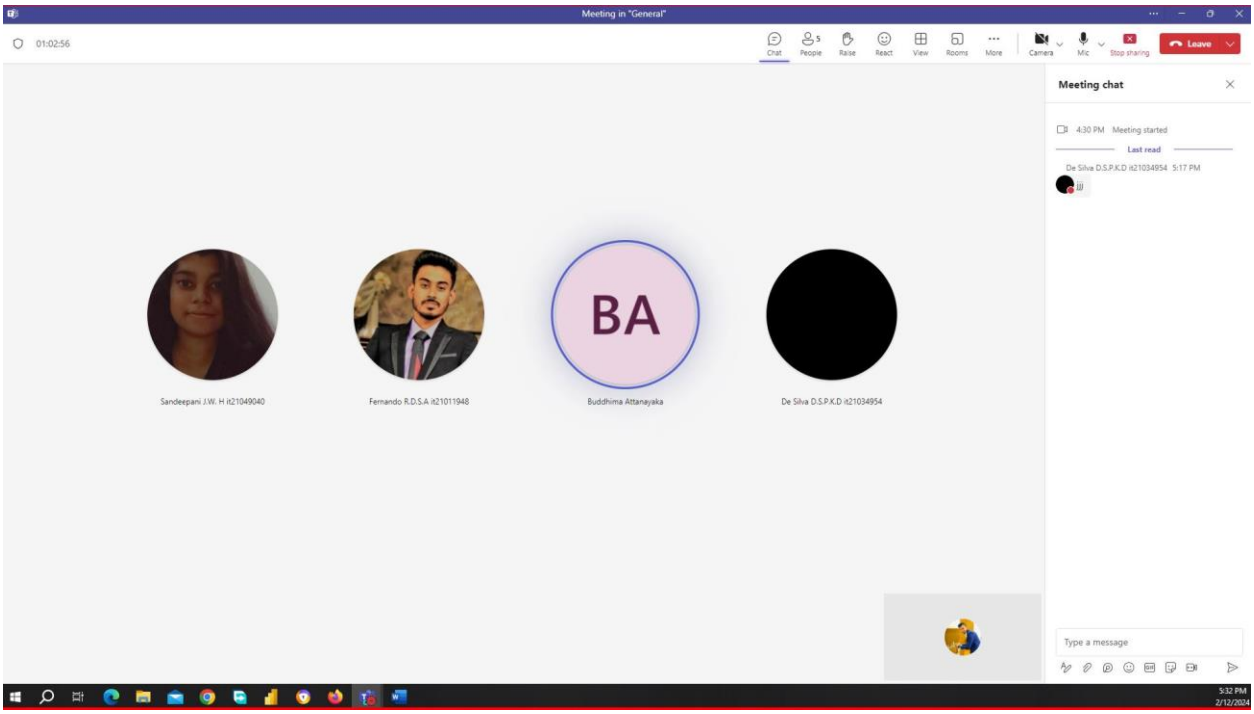


Figure 10 - Teams Meeting with the Co-supervisor (Example 02)

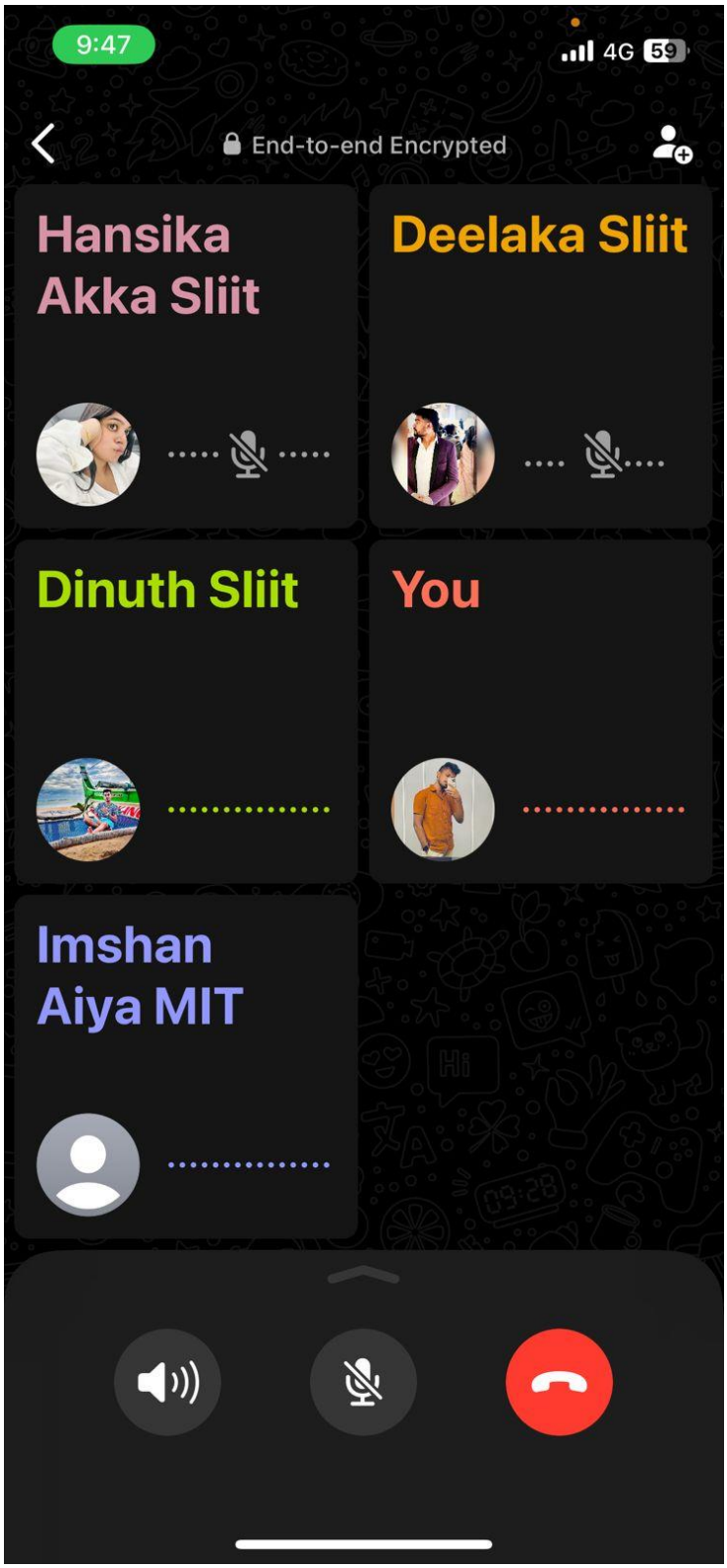


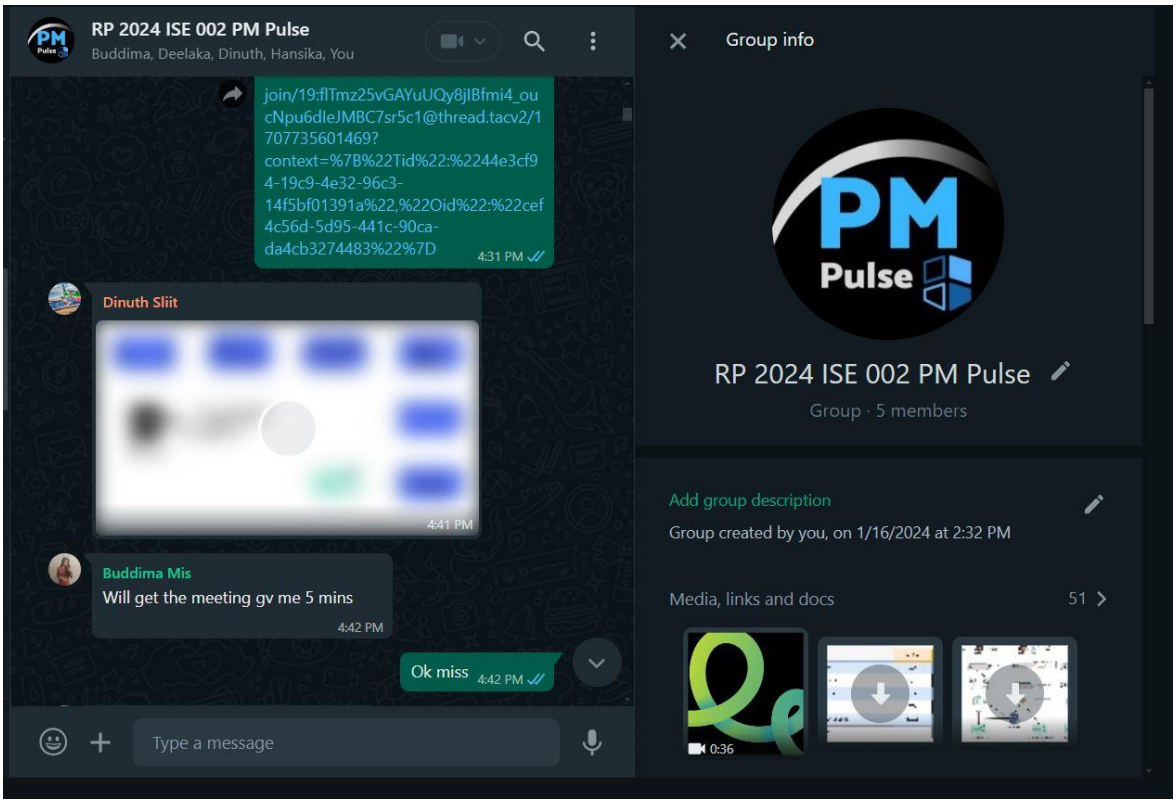
Figure 11 - WhatsApp Call with External Supervisor (LSEG)

Physical Meetings with Group Members

Figure 12 - Physical Meetings with team members.

WhatsApp Group Creation

Figure 13 - WhatsApp Group Creation with Co-Supervisor



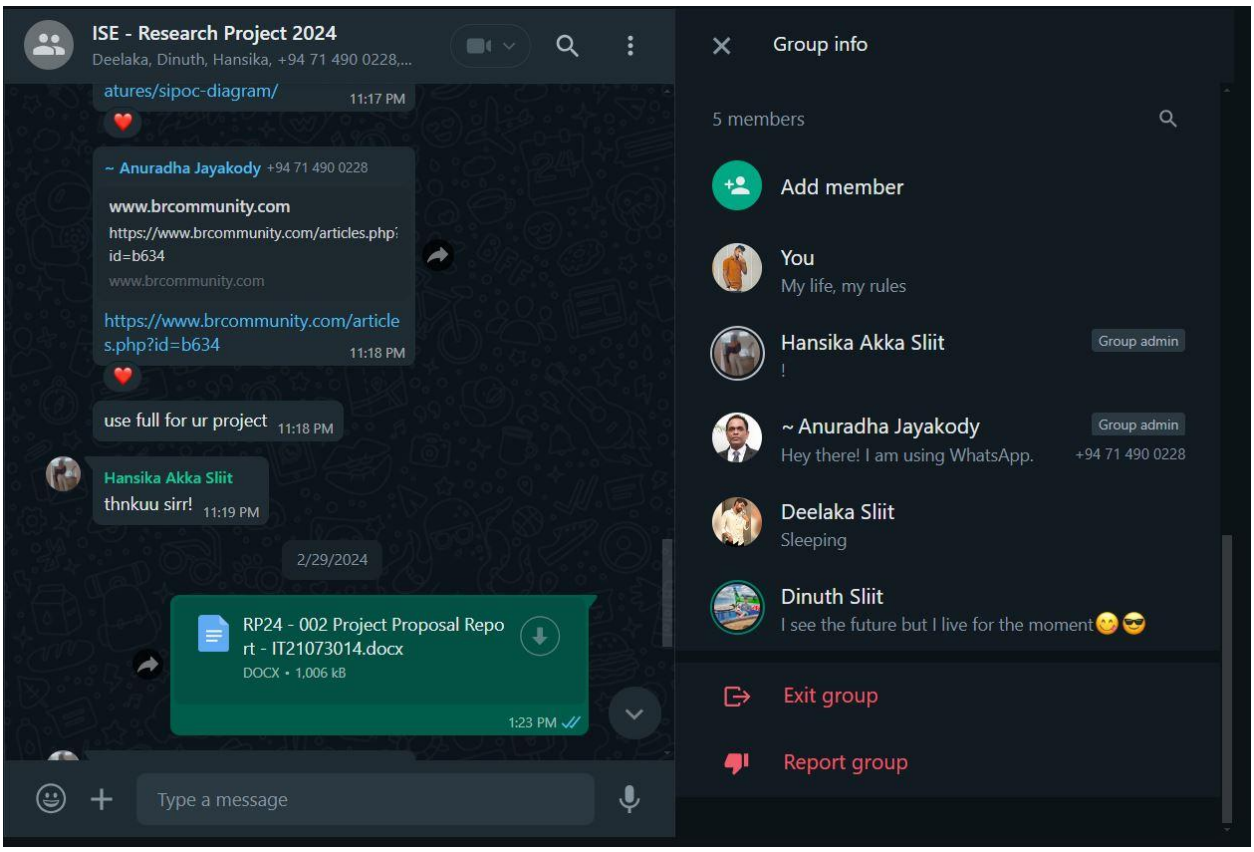


Figure 14 - WhatsApp Group Creation with Supervisor

Project Timeline

We put together a Gantt chart that outlines the entire timeline of our project, encompassing all the deadlines for our deliverables. This visual representation provides a comprehensive overview of our project's progression from its inception to its completion.

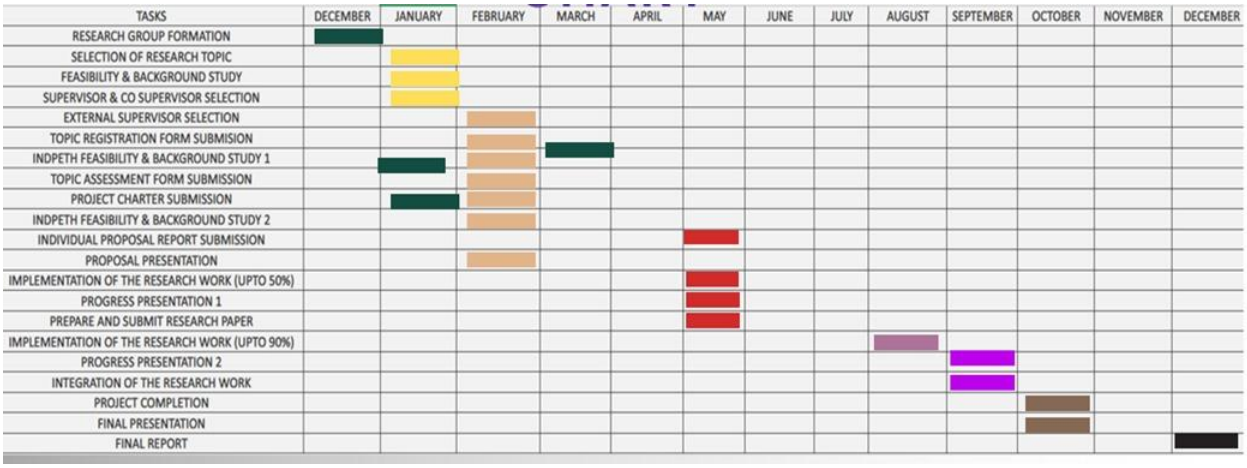


Figure 15 - Gantt Chart (Project Timeline)

Work Break-Down

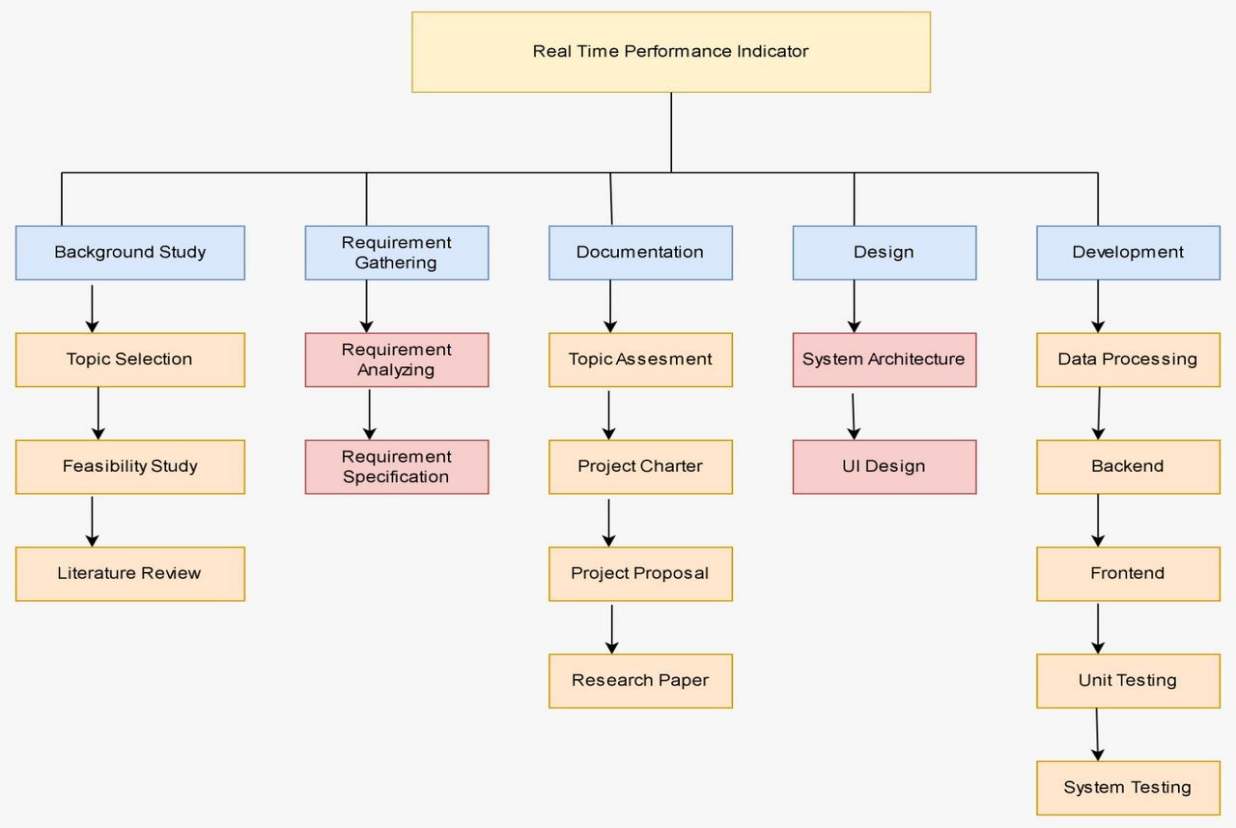


Figure 16 - Work Break-Down Structure

Version Controlling

The team implemented GitHub as their version control system, creating a repository that included all four group members. Each member was responsible for committing their code changes and progress to the repository.

To ensure transparency and collaboration, all code changes were incrementally committed and pushed to individual branches. These branches were later merged into the main branch during weekly meetings. This approach allowed for effective version control and streamlined collaboration among team members.

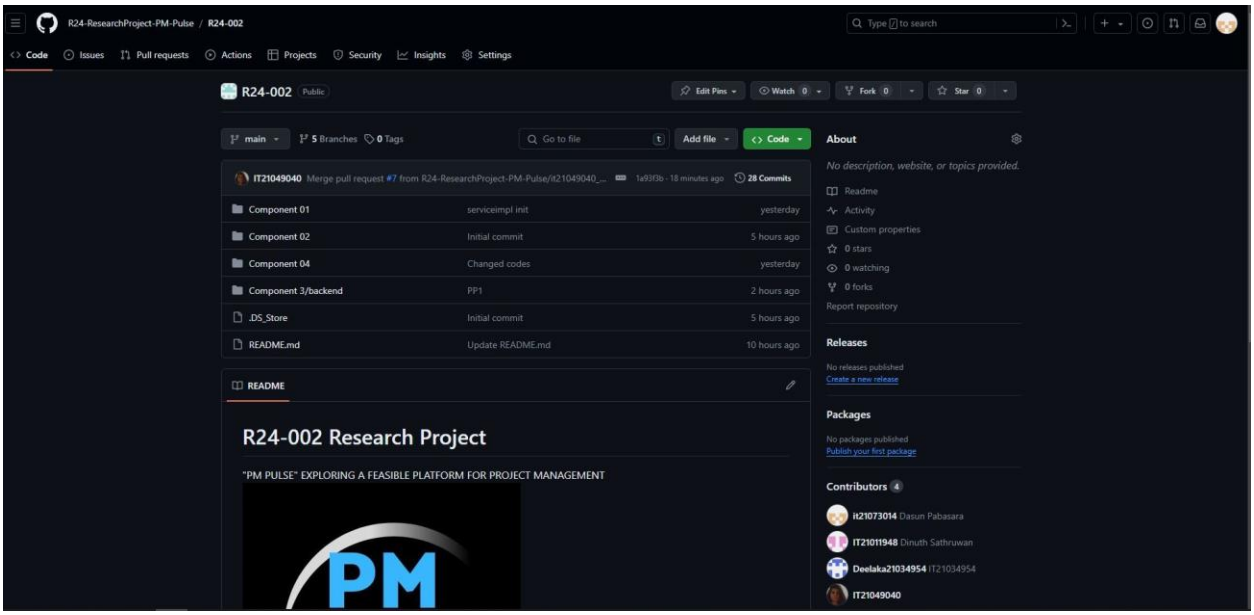


Figure 17 - GitHub Repository

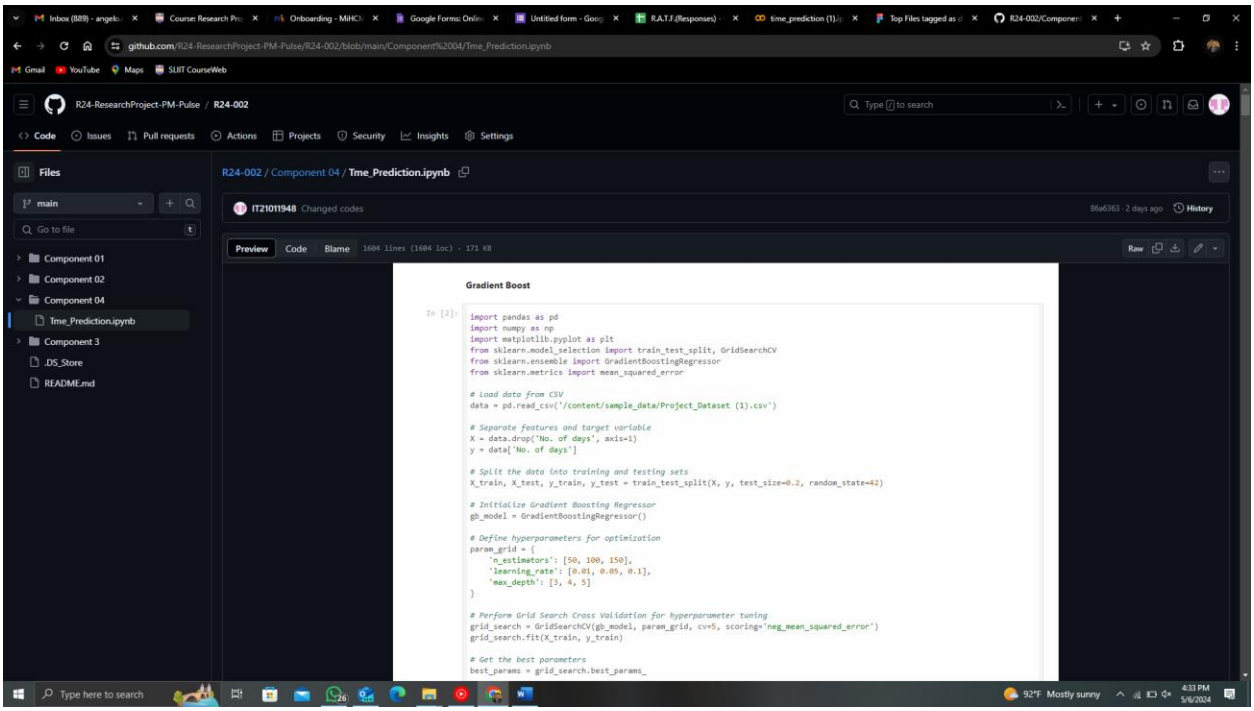
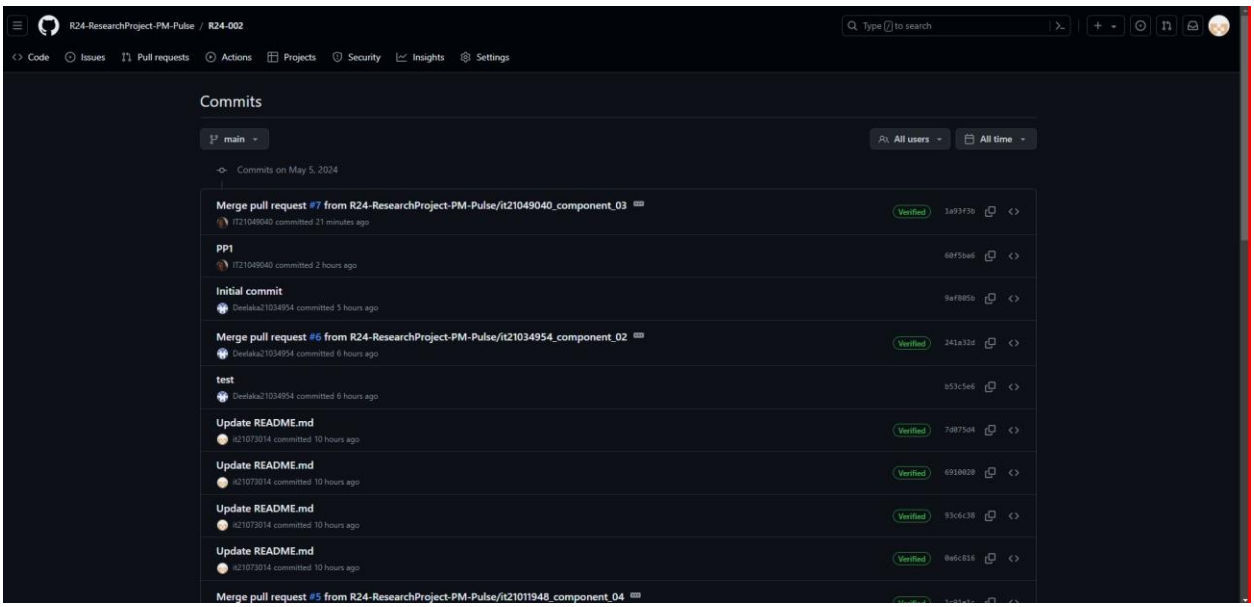
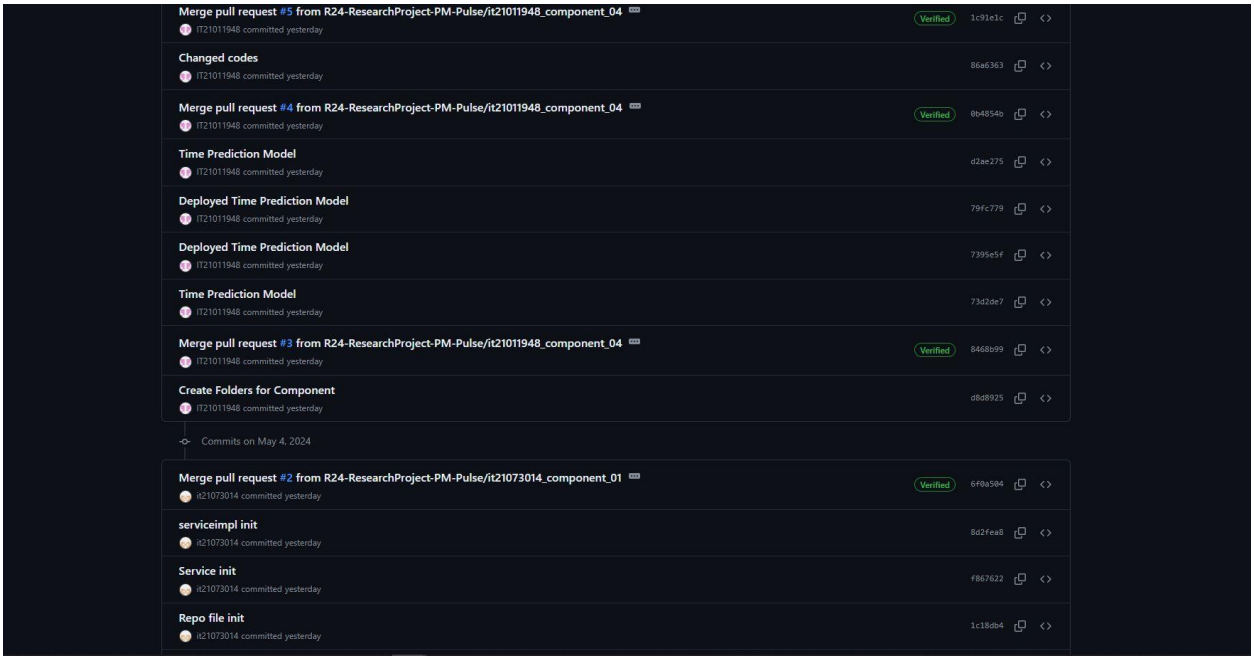


Figure 18 - GitHub Repository (Component 04 Branch)

GitHub Commits





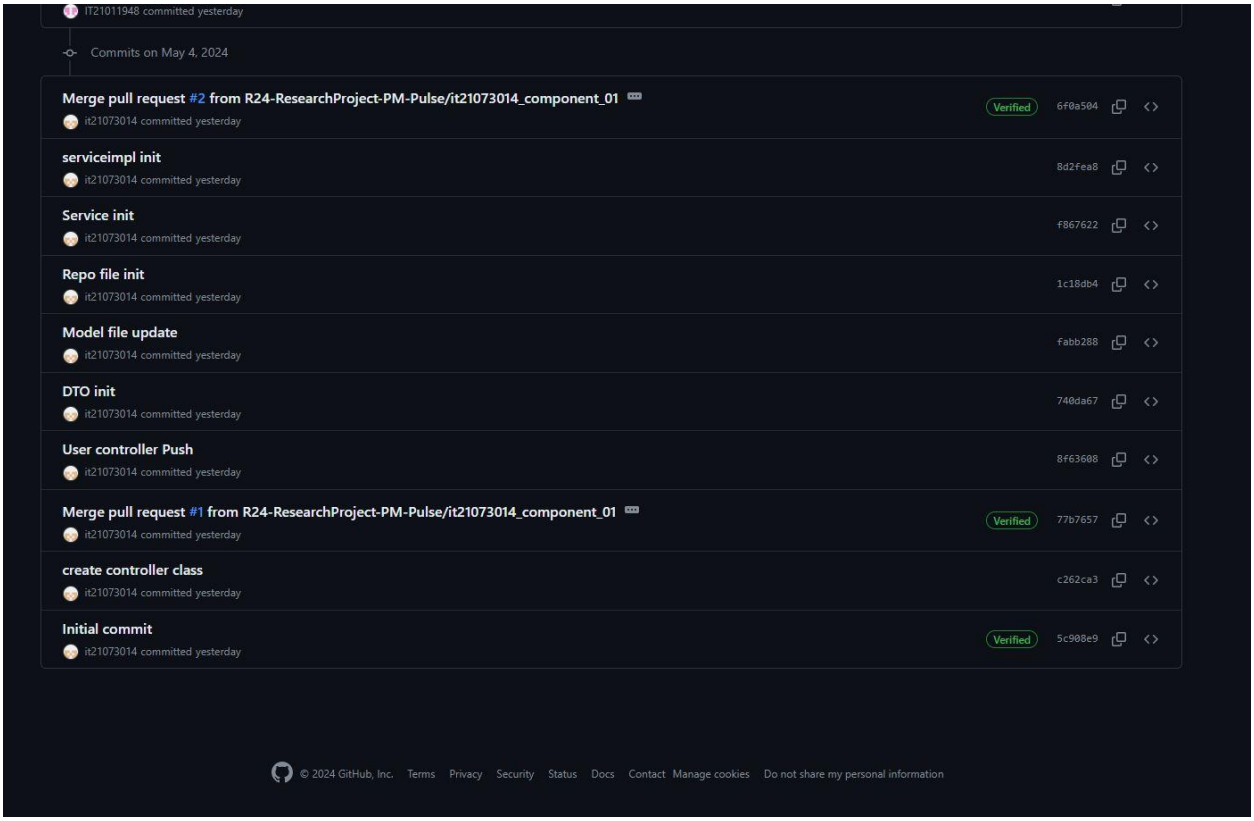


Figure 19 - GitHub Commits

GitHub Graph of Commits

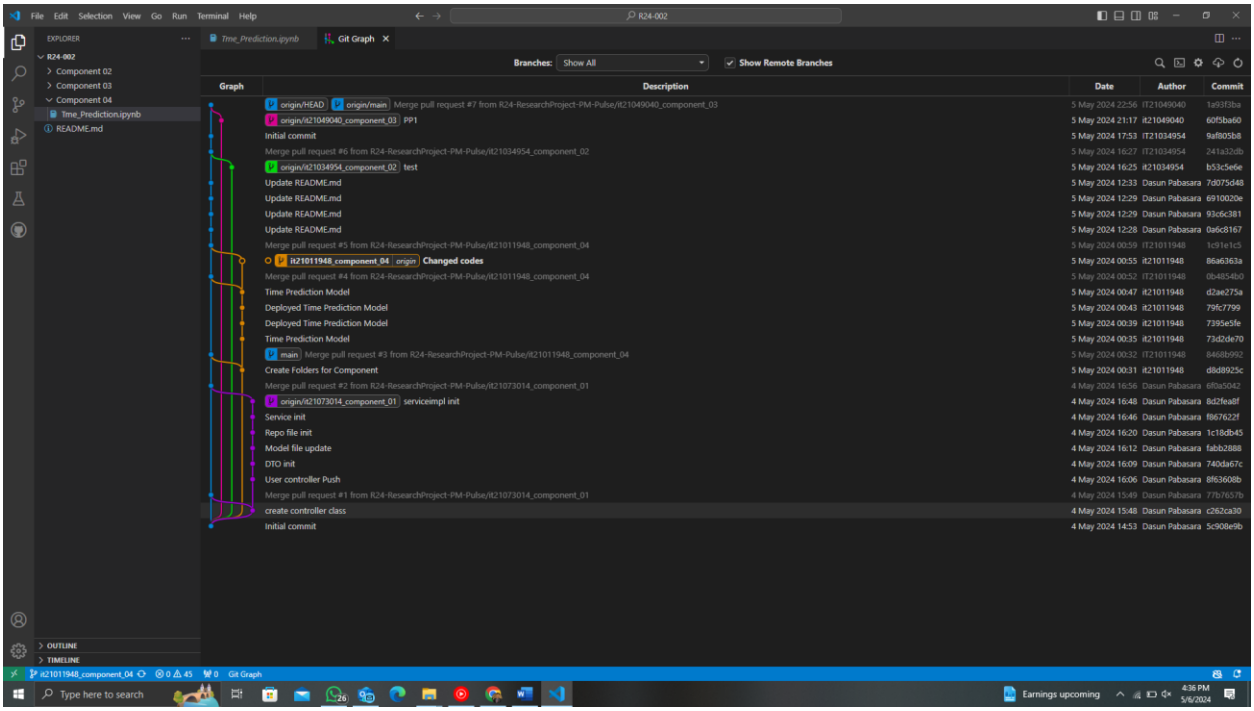
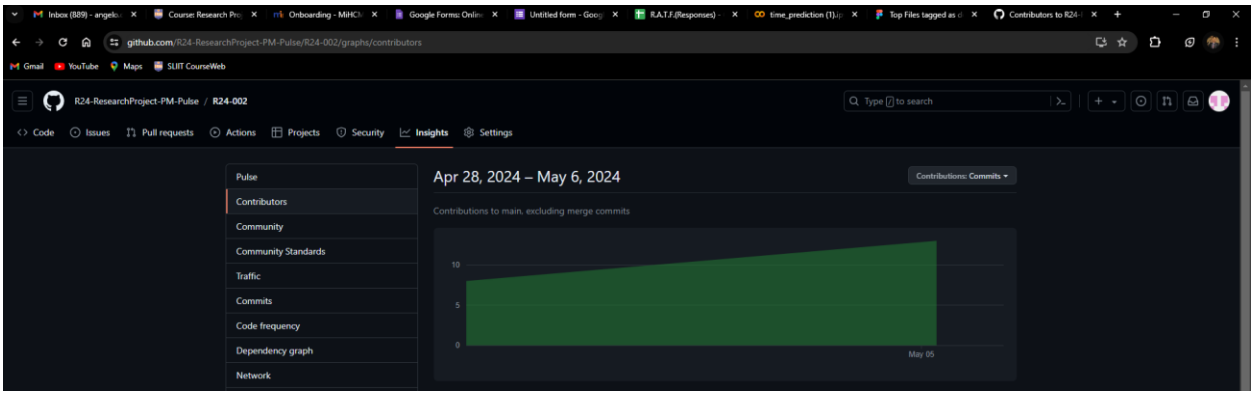
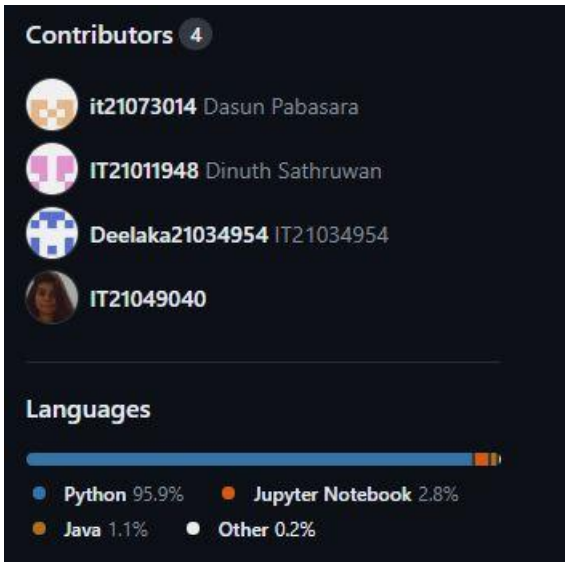


Figure 20 - Git Graph

GitHub Contributors





Contribution Charts

Figure 21 - GitHub Contributors

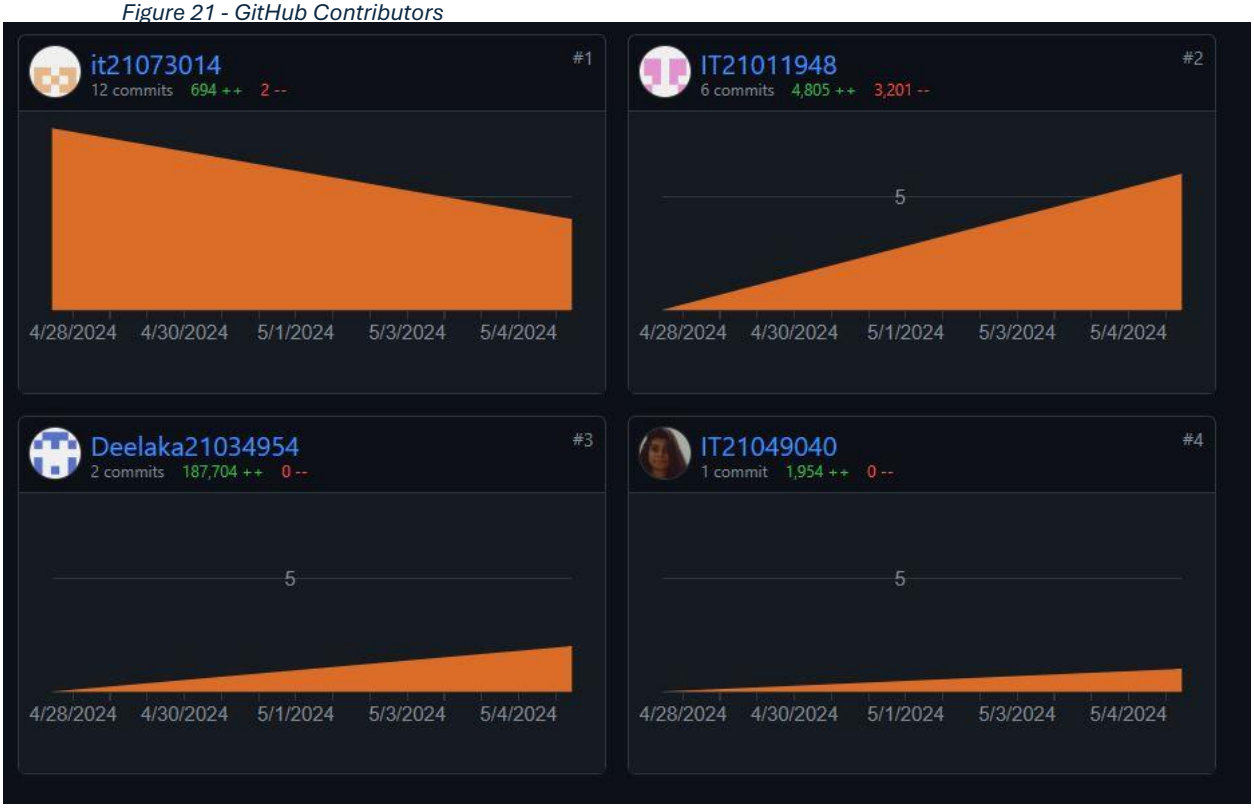


Figure 22 - Contributors Charts