

Logbook – Secure Aggregation Module of R25 -039



Project ID: R25 - 039

Project Title: Data-Privacy Focused Federated Learning Framework for Industrial IoT

Student Details:	
Names:	Student IDs:
Weerasinghe K.M	IT21831904

Supervisor: Mr. Amila Seneratha

Co-Supervisor: Mr. Tharaniyawarma Kumaralingam

Date of Submission: 2025

Contents

1.	Group Details.....	3
2.	Project Details.....	4
3.	Communication Methods.....	4
4.	Meetings With Supervisors.....	7
5.	Task Details	8
1.	Personal task Assigning and Completion	8
6.	System Details	9
2.	System completion status.....	9
3.	System Design	11
I.	System Architecture.....	11
II.	Module Architecture	12
4.	System Testing.....	13
5.	System Codes.....	14
7.	GithHub Upload.....	18
8.	Documentation.....	18
6.	Proposal.....	18
7.	Presentation 1.....	19
8.	Presentation 2.....	19
9.	Final Presentation.....	20
10.	Final Product.....	21
11.	Research Paper.....	22
III.	Conference Appetence	22
9.	CDAP upload.....	23
10.	Website	24
12.	Development.....	24
13.	Finalize.....	27

1. Group Details

Student Details:		
Names:	Student IDs:	Research Component
Nanayakkara Y.D.T. D	IT21826368	Attack Defense and Resillence
Mendis H.R.M	IT21822612	Privacy Preseravation
Weerasinghe K.M	IT21831904	Secure Aggrigation
Dissanayaka K.D.A.R. A	IT21828348	Secure Communicaiton and Protocol Enforcement

2. Project Details

Topic - Data-Privacy Focused Federated Learning Framework for Industrial IoT

Aim – To develop a product that going to full fill the research

Deliverables – Federated Learning Framework designed for industrial internet of things

This project was initiated to develop a secure and private **Federated Learning (FL) framework** specifically for **Industrial IoT (IIoT)** environments.

The Challenge: Traditional AI methods require centralizing sensitive factory data, which poses major **privacy risks** and clashes with the distributed nature of industrial operations. Existing FL solutions are insufficient because they fail to simultaneously provide robust security, data privacy, and efficient operation on **resource-limited IIoT devices**.

The Solution: The developed framework is a multi-layered system that provides **end-to-end protection**.

- It uses techniques like **Differential Privacy (DP)** and **Homomorphic Encryption (HE)** to guarantee data confidentiality.
- It implements a robust protocol that uses **client/server validation** to actively block cyber threats such as **Model Poisoning and Byzantine Attacks**.
- The system is optimized for **efficiency** to reduce overhead on IIoT devices.

3. Communication Methods

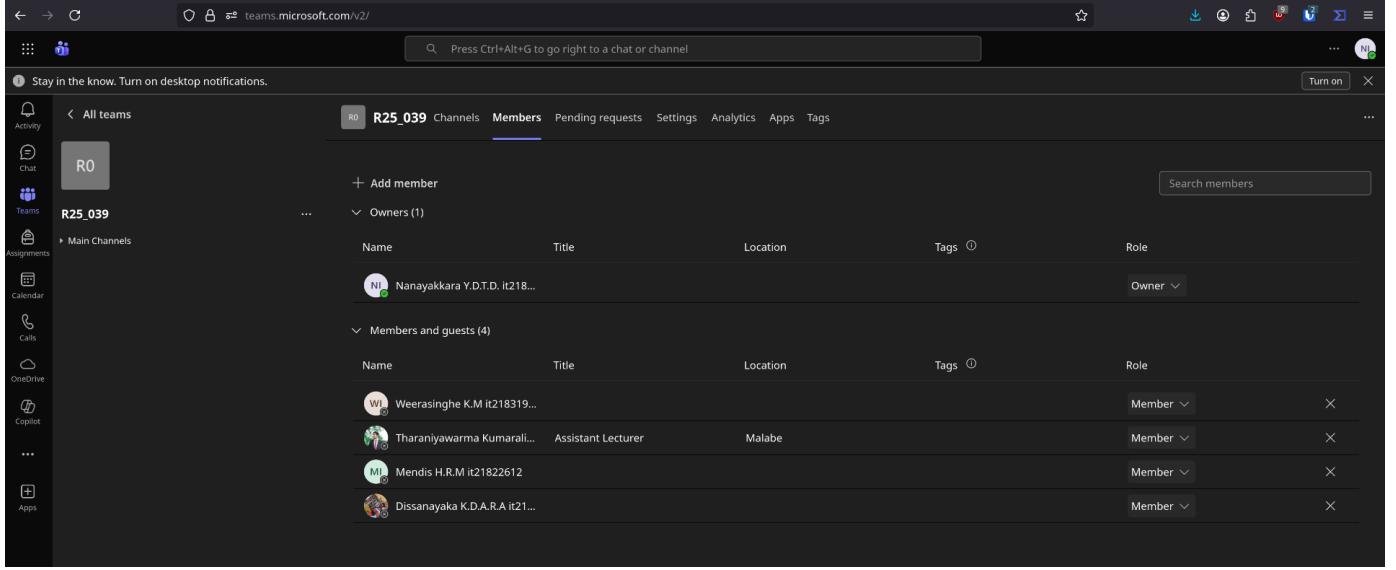
WhatsApp Group – Team

Advanced chat privacy
Off

4 members 

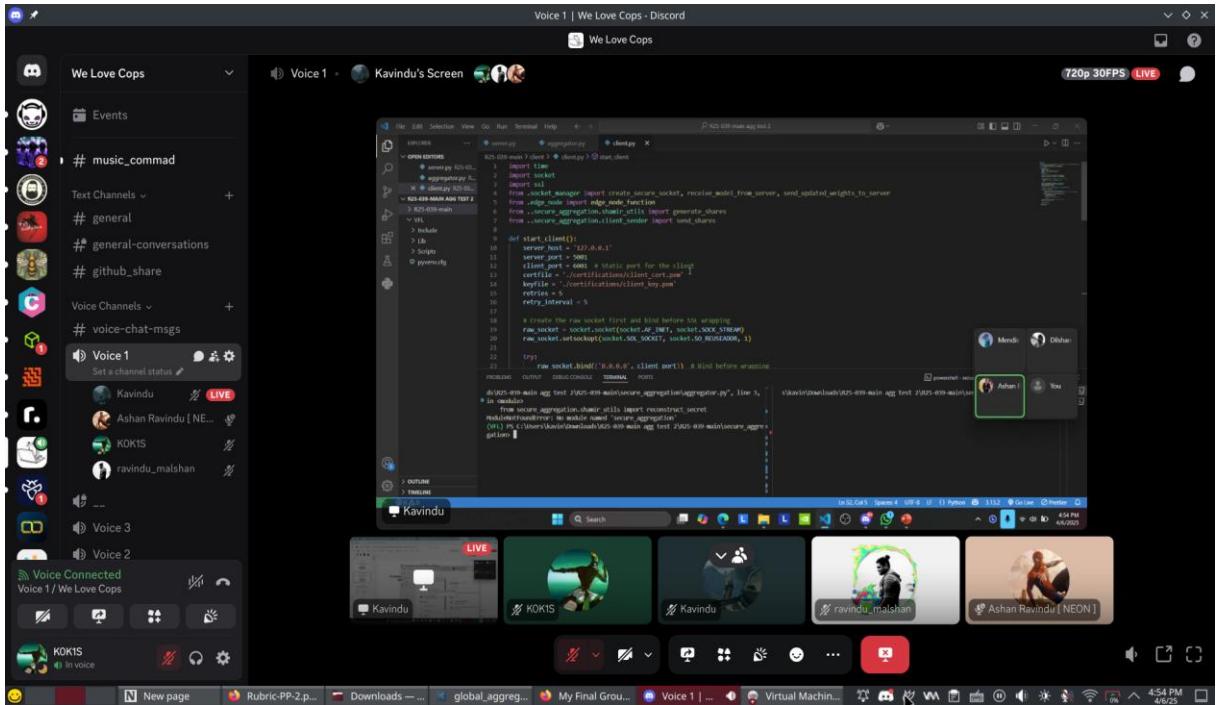
 You	
Busy	
<hr/>	
 Kavindu Weerasinghe	Group admin
Just me myself and I	
<hr/>	
 Ravindu Mendis	Group admin
<hr/>	
 Ashan Ravindu	
කම්දුවත් පාරාජයන් වලදී ඔයාගේ වාහාට වලට යන ගෙනි තෙව්තෙන්න එපා	
<hr/>	
 Add to favourites	

Microsoft Teams - All



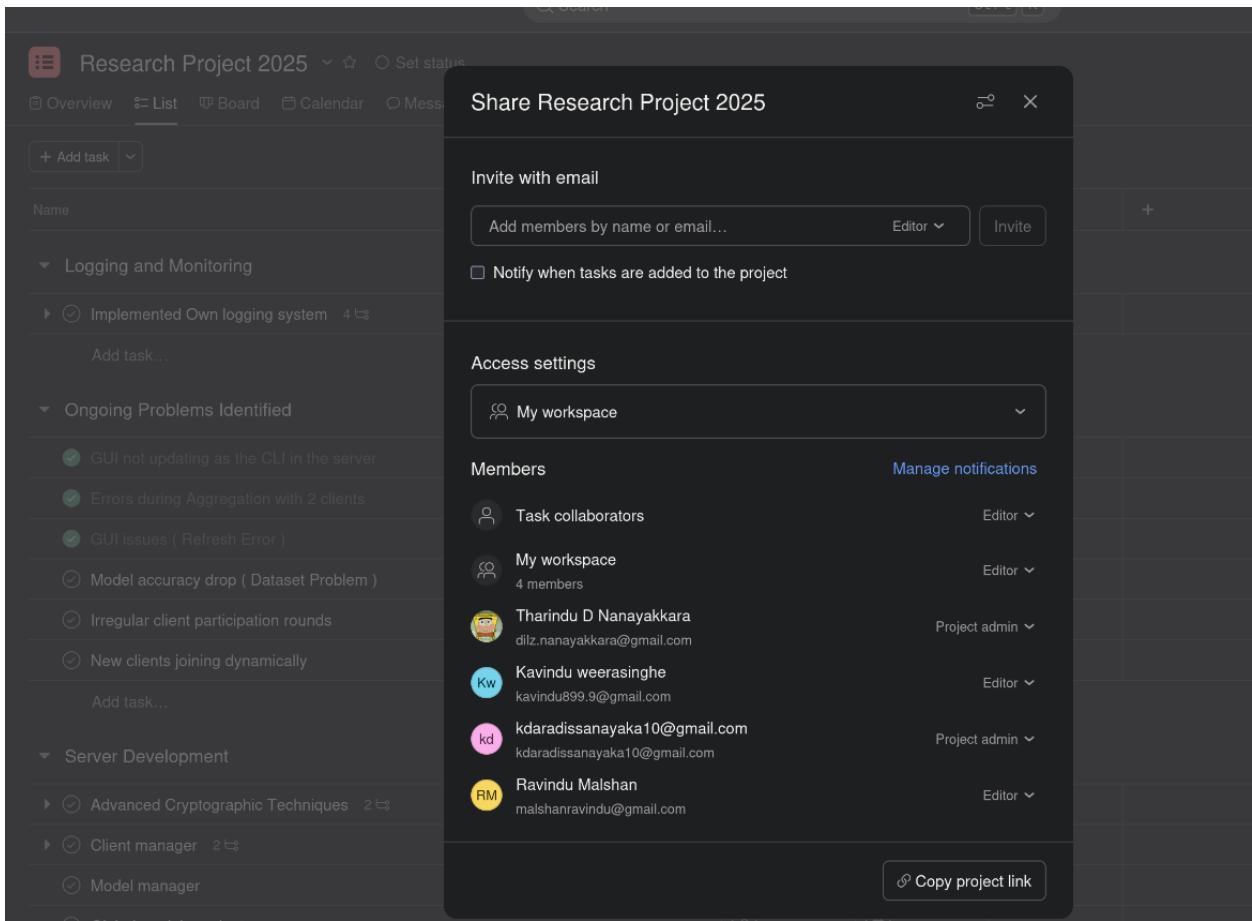
The screenshot shows the Microsoft Teams interface for the 'R25_039' channel. On the left, there's a sidebar with various team-related links like Activity, Chat, Teams, Assignments, Calendar, Calls, OneDrive, Copilot, and Apps. The main area displays the channel members. At the top, there are tabs for 'Channels' (selected), 'Members' (highlighted in blue), 'Pending requests', 'Settings', 'Analytics', 'Apps', and 'Tags'. Below the tabs, there's a search bar with the placeholder 'Search members'. Under the 'Members' tab, there are two sections: 'Owners (1)' and 'Members and guests (4)'. The 'Owners' section lists Nanayakkara Y.D.T.D. it218... as the owner. The 'Members and guests' section lists four members: Weerasinghe K.M it218319..., Tharaniyawarma Kumarali... (Assistant Lecturer from Malabe), Mendis H.R.M it21822612, and Dissanayaka K.D.A.R.A it21... All listed members have the role 'Member'.

Group Meetings – Discord – Team



The screenshot shows a Discord voice channel named 'Voice 1' for the server 'We Love Cops'. The channel interface includes a list of users (Kavindu, Ashan Ravindu [NEON], KOKIS, ravindu_malshan) and a video feed showing 'Kavindu's Screen'. The video feed is set to '720p 30FPS' and is currently 'LIVE'. In the background, there's a code editor window showing Python code related to a secure socket connection. The desktop taskbar at the bottom shows other open applications like a browser, file explorer, and system tray icons.

Asana – Task Assigning – Team

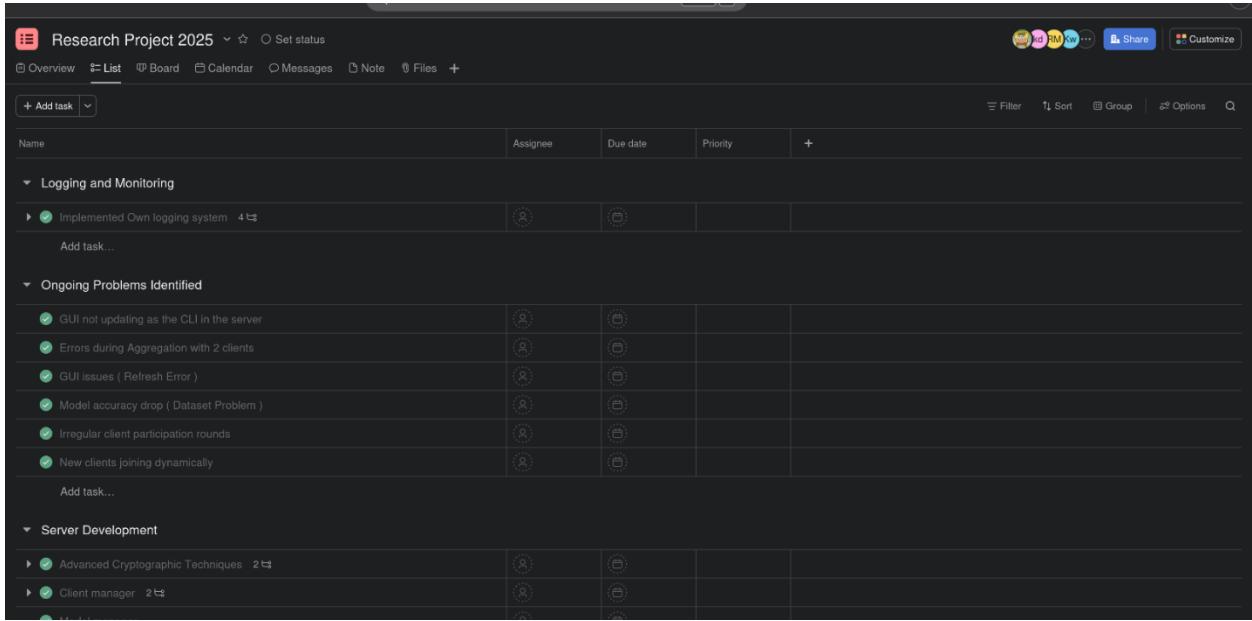


The screenshot shows a project management interface for 'Research Project 2025'. On the left, the main workspace is visible with sections like 'Logging and Monitoring', 'Ongoing Problems Identified', and 'Server Development'. A modal window titled 'Share Research Project 2025' is open on the right. It contains two tabs: 'Invite with email' (selected) and 'Access settings'. Under 'Invite with email', there's a text input for 'Add members by name or email...', a dropdown for 'Editor', and a 'Invite' button. There's also a checkbox for 'Notify when tasks are added to the project'. Under 'Access settings', it shows 'My workspace' selected. The 'Members' section lists four users with their roles: Tharindu D Nanayakkara (Project admin), Kavindu weerasinghe (Editor), kdaradissanayaka10@gmail.com (Project admin), and Ravindu Malshan (Editor). A 'Manage notifications' dropdown is shown next to each user. At the bottom of the modal is a 'Copy project link' button.

4. Meetings With Supervisors

All the meetings were conducted in person and only WhatsApp calls were taken to organize the meeting

5. Task Details



The screenshot shows a task management interface for a project titled "Research Project 2025". The interface includes a header with navigation links like Overview, List, Board, Calendar, Messages, Note, Files, and a search bar. Below the header is a toolbar with icons for filter, sort, group, options, and share.

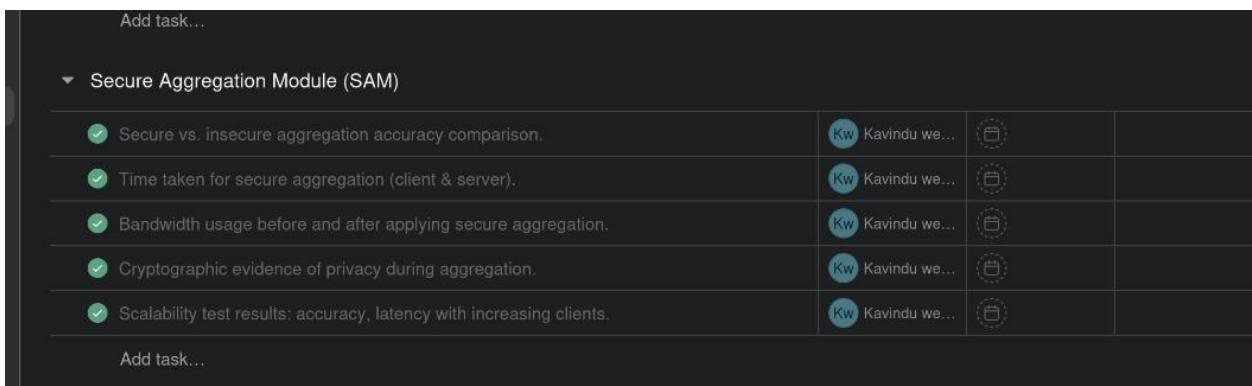
The main area displays a table with columns for Name, Assignee, Due date, Priority, and actions. Tasks are categorized into three sections:

- Logging and Monitoring:**
 - Implemented Own logging system (4 tasks)
- Ongoing Problems Identified:**
 - GUI not updating as the CLI in the server
 - Errors during Aggregation with 2 clients
 - GUI issues (Refresh Error)
 - Model accuracy drop (Dataset Problem)
 - Irregular client participation rounds
 - New clients joining dynamically
- Server Development:**
 - Advanced Cryptographic Techniques (2 tasks)
 - Client manager (2 tasks)

Each task row contains icons for user, file, and more actions. A "Add task..." button is located at the bottom of each section.

Secure Aggregation Module was mine.

5.1 Personal task Assigning and Completion



The screenshot shows a personal task list for the "Secure Aggregation Module (SAM)". The list includes the following tasks, all of which are marked as completed (green checkmark):

- Secure vs. insecure aggregation accuracy comparison.
- Time taken for secure aggregation (client & server).
- Bandwidth usage before and after applying secure aggregation.
- Cryptographic evidence of privacy during aggregation.
- Scalability test results: accuracy, latency with increasing clients.

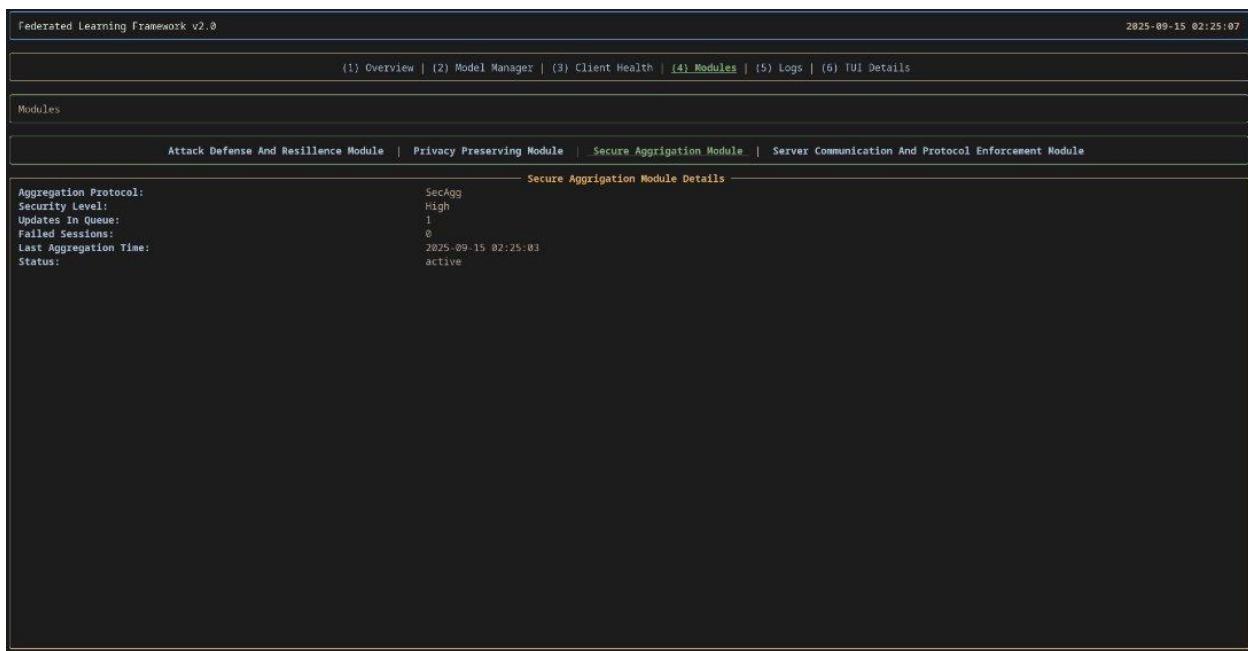
Each task row includes columns for assignee (Kavindu we...), due date, and file attachments. A "Add task..." button is located at the bottom of the list.

6. System Details

6.1 System completion status

Finished

SAM TUI



Federated Learning Framework v2.0 2025-09-15 02:25:07

(1) Overview | (2) Model Manager | (3) Client Health | (4) Modules | (5) Logs | (6) TUI Details

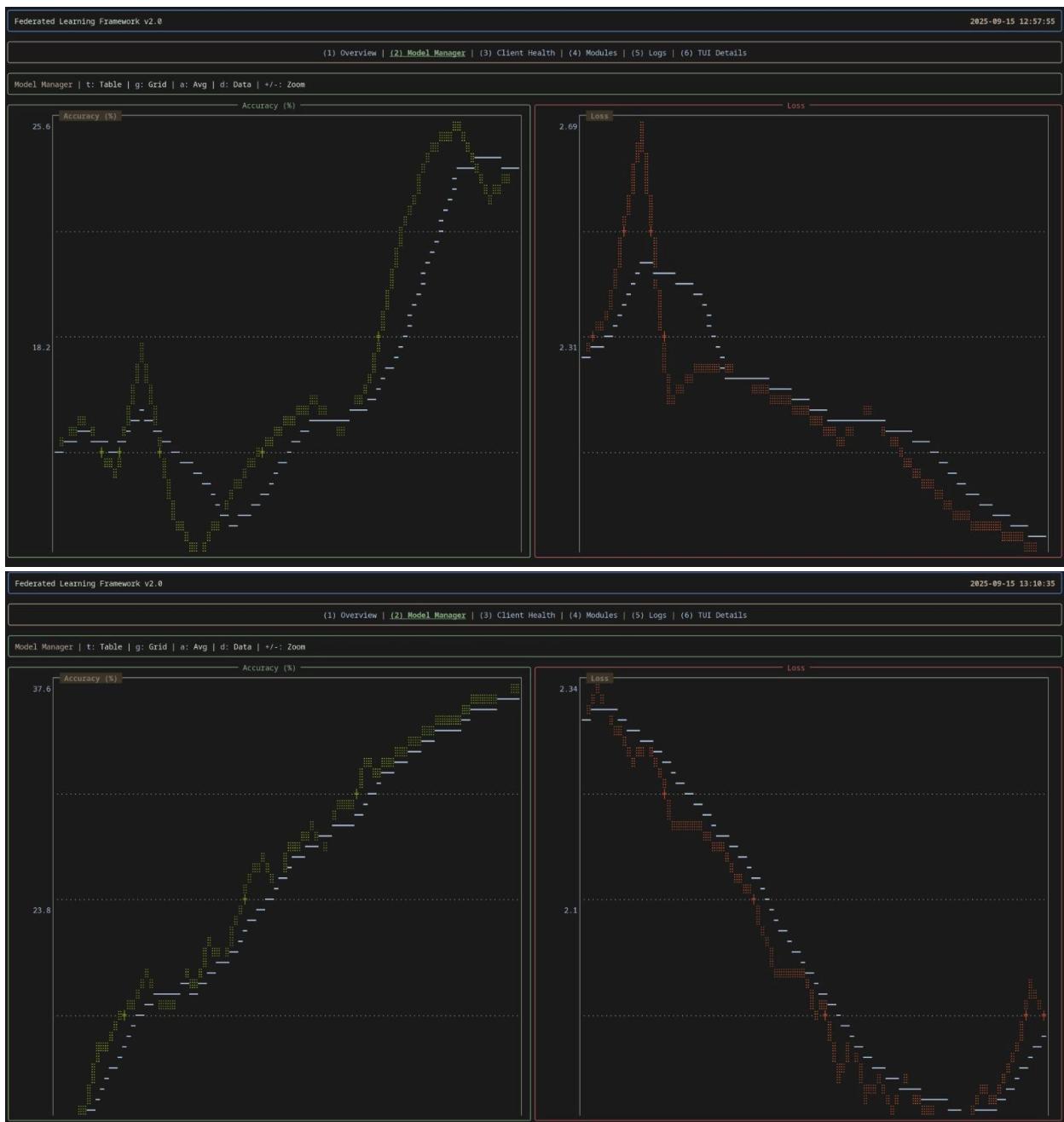
Modules

Attack Defense And Resilience Module | Privacy Preserving Module | Secure Aggregation Module | Server Communication And Protocol Enforcement Module

Secure Aggregation Module Details

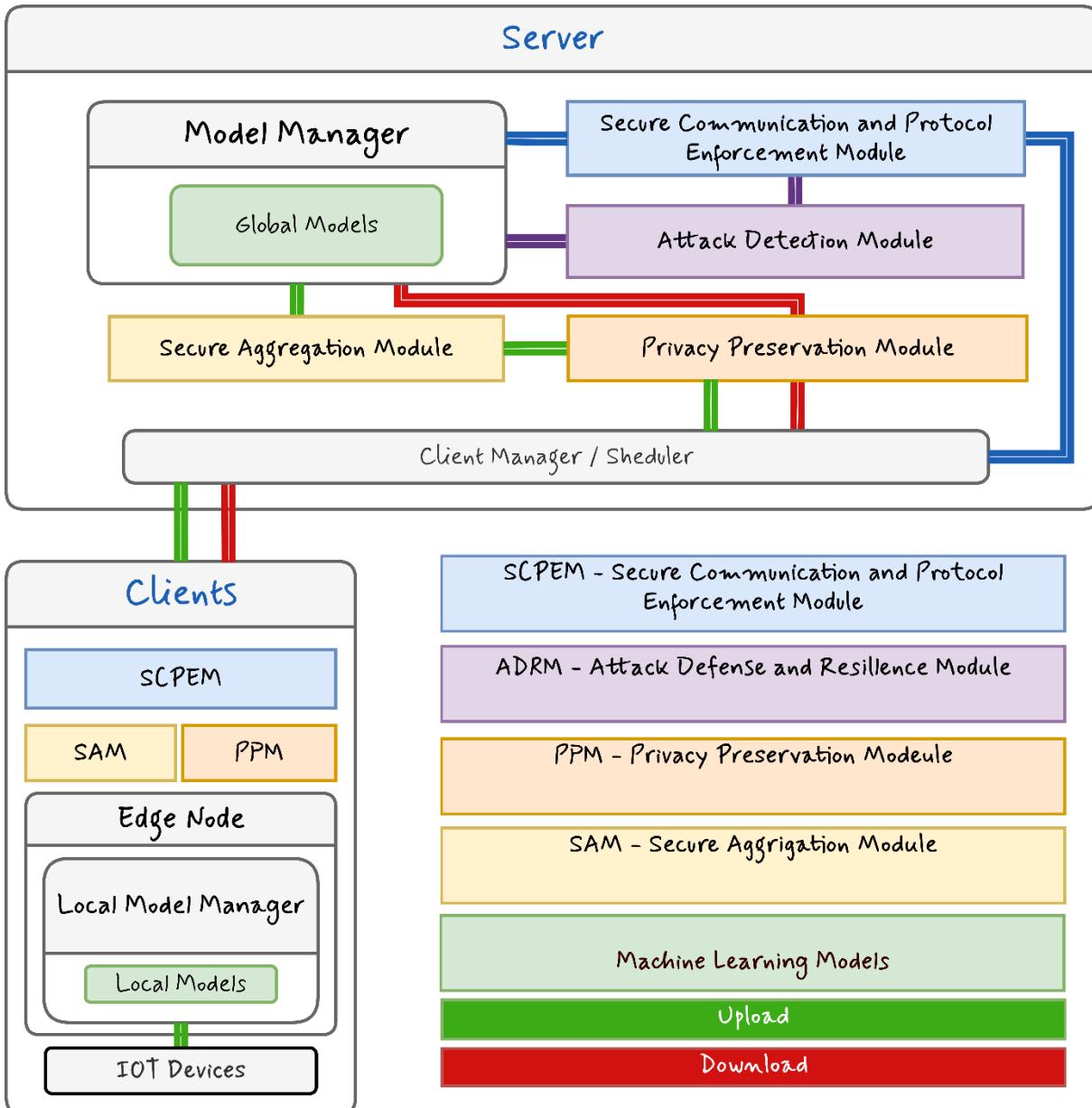
Aggregation Protocol:	SecAgg
Security Level:	High
Updates In Queue:	1
Failed Sessions:	0
Last Aggregation Time:	2025-09-15 02:25:03
Status:	active

Aggregation Process server

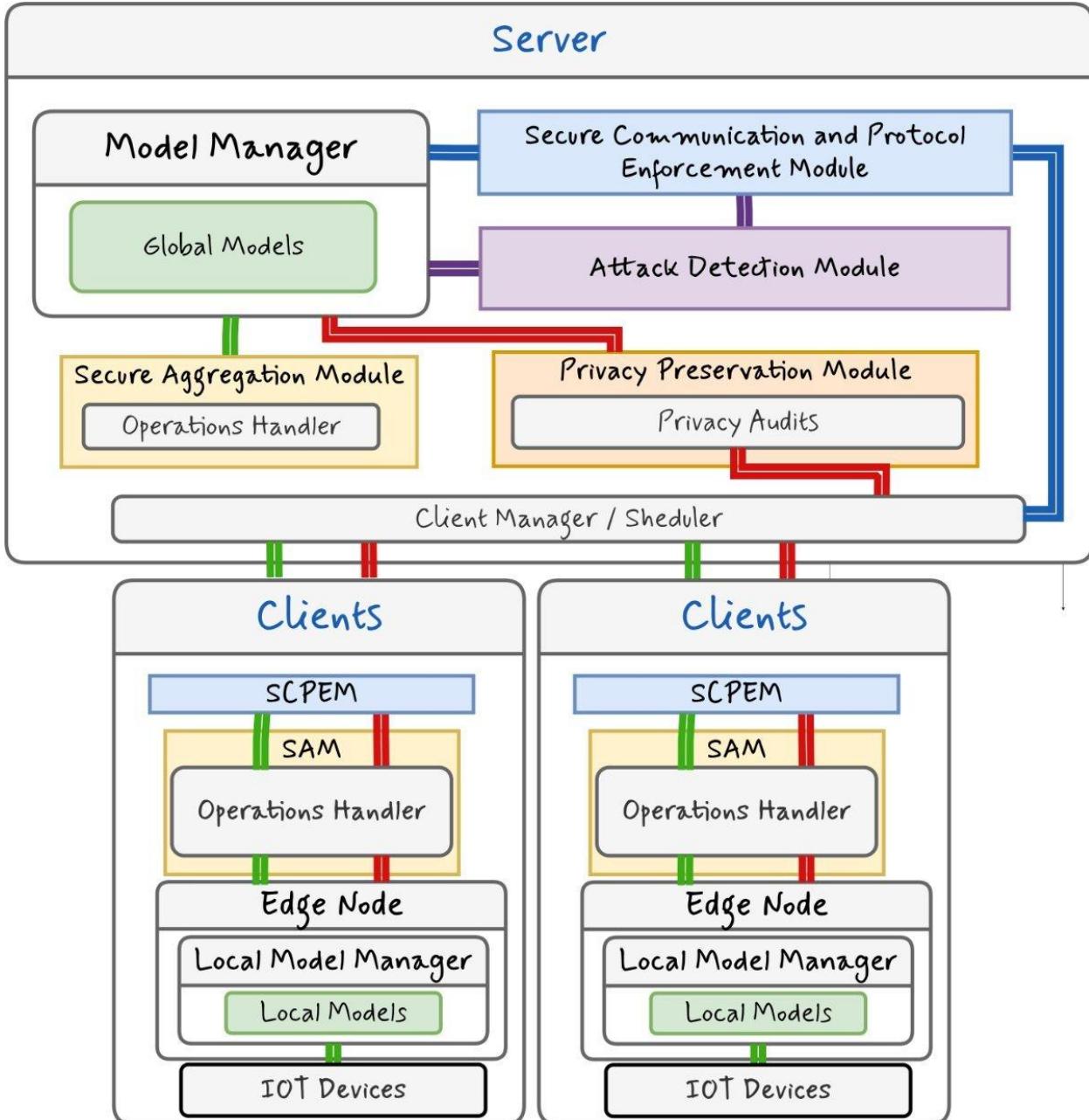


6.2 System Design

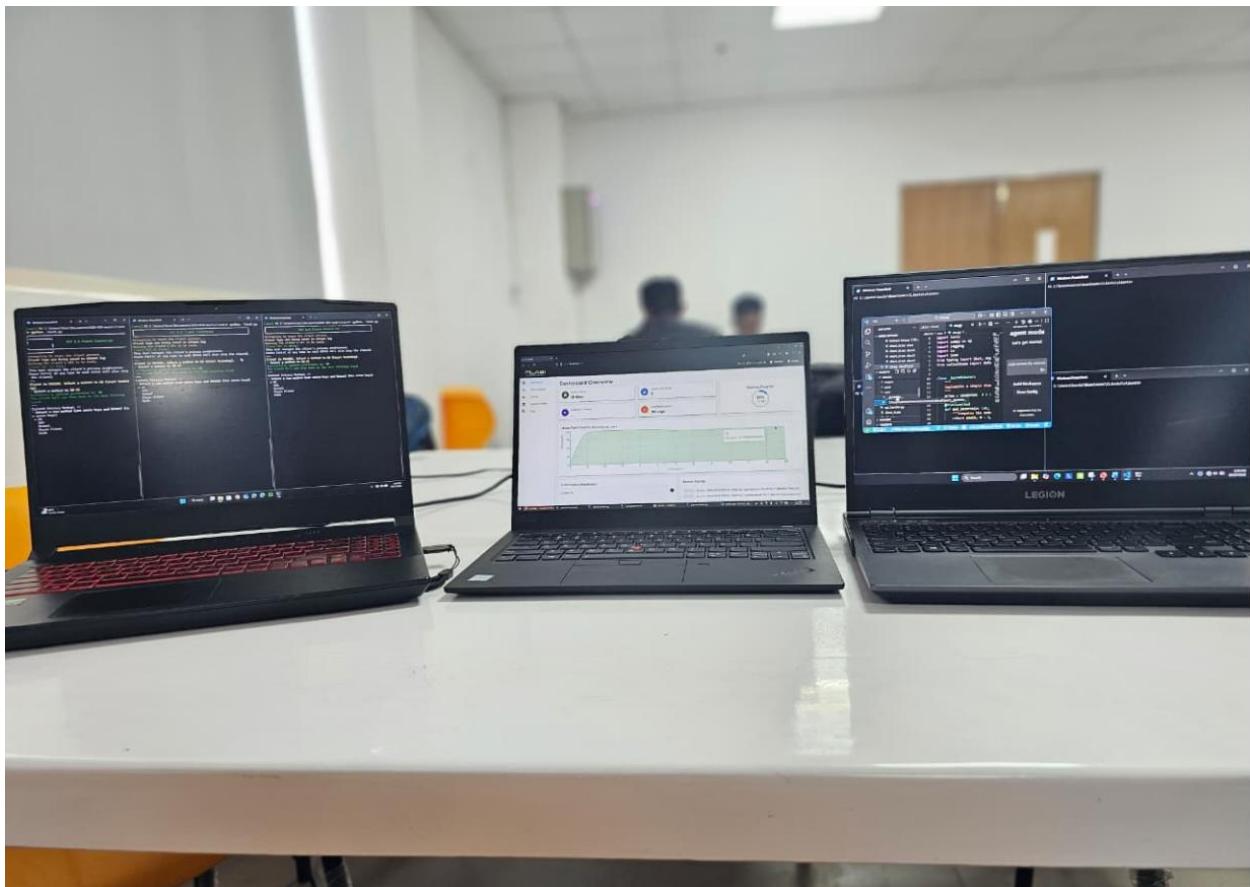
I. System Architecture



II. Module Architecture

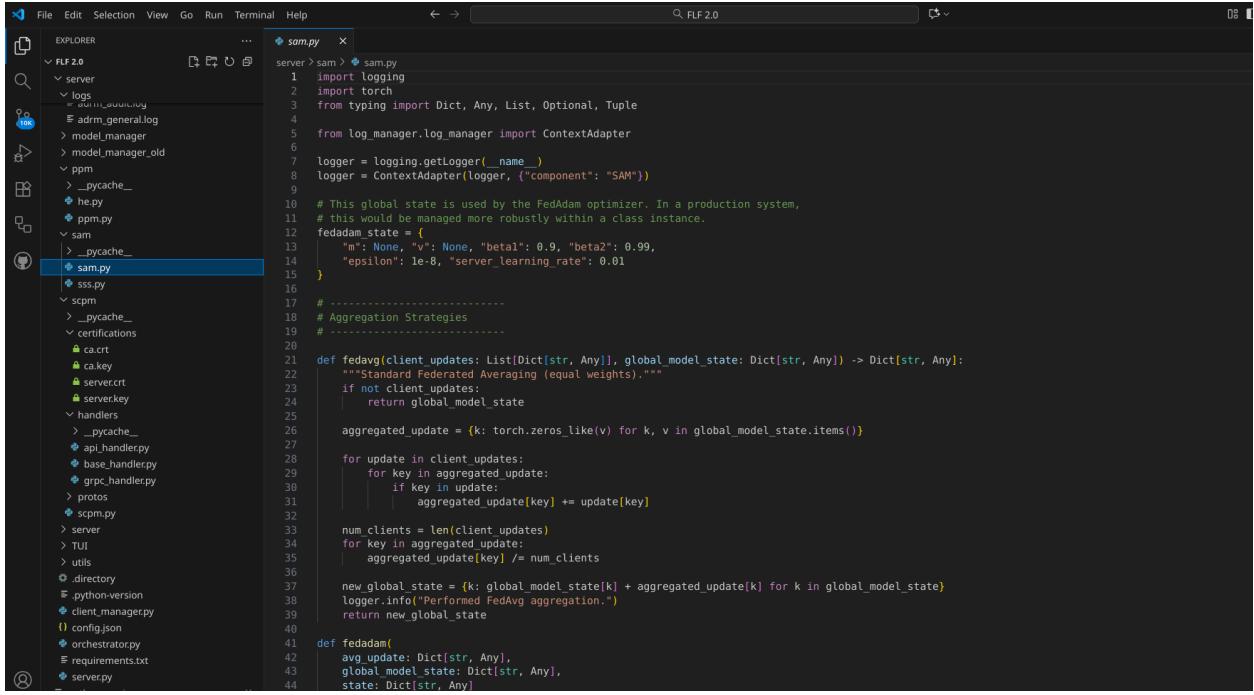


6.3 System Testing



6.4 System Codes

SAM – Server



The screenshot shows a code editor interface with the following details:

- File Path:** FLF 2.0 > server > sam > sam.py
- Code Content (sam.py):**

```

1 import logging
2 import torch
3 from typing import Dict, Any, List, Optional, Tuple
4
5 from log_manager.log_manager import ContextAdapter
6
7 logger = logging.getLogger(__name__)
8 logger = ContextAdapter(logger, {"component": "SAM"})
9
10 # This global state is used by the FedAdam optimizer. In a production system,
11 # this would be managed more robustly within a class instance.
12 fedadam_state = {
13     "m": None, "v": None, "beta1": 0.9, "beta2": 0.99,
14     "epsilon": 1e-8, "server_learning_rate": 0.01
15 }
16
17 # -----
18 # Aggregation Strategies
19 #
20
21 def fedavg(client_updates: List[Dict[str, Any]], global_model_state: Dict[str, Any]) -> Dict[str, Any]:
22     """Standard Federated Averaging (equal weights)."""
23     if not client_updates:
24         return global_model_state
25
26     aggregated_update = {k: torch.zeros_like(v) for k, v in global_model_state.items()}
27
28     for update in client_updates:
29         for key in aggregated_update:
30             if key in update:
31                 aggregated_update[key] += update[key]
32
33     num_clients = len(client_updates)
34     for key in aggregated_update:
35         aggregated_update[key] /= num_clients
36
37     new_global_state = {k: global_model_state[k] + aggregated_update[k] for k in global_model_state}
38     logger.info("Performed FedAvg aggregation.")
39     return new_global_state
40
41 def fedadam(
42     avg_update: Dict[str, Any],
43     global_model_state: Dict[str, Any],
44     state: Dict[str, Any]
45 ):

```

Model manager

File Edit Selection View Go Run Terminal Help

FLF 2.0

EXPLORER

```

server > model_manager > data_loader.py
1 # data_loader.py
2
3 import torch
4 from torch.utils.data import Dataset, Data
5 import pandas as pd
6 from sklearn.preprocessing import Standards
7 import numpy as np
8 import os
9 import logging
10
11 logger = logging.getLogger(__name__)
12
13 # New Custom Dataset Class for Tabular
14 class IiotDataset(Dataset):
15     """Custom dataset for WUSTL-IIOT-2021 t
16     def __init__(self, features, labels):
17         # Convert preprocessed numpy arrays
18         self.features = torch.tensor(features)
19         self.labels = torch.tensor(labels,
20
21     def __len__(self):
22         return len(self.labels)
23
24     def __getitem__(self, idx):
25         return self.features[idx], self.lab
26
27     def load_wustl_iiot_test_data(file_path: st
28     """
29     Loads, preprocesses, and prepares the W
30
31     Returns:
32     tuple: (DataLoader, num_features, n
33
34     if not os.path.exists(file_path):
35         logger.warning(f"Dataset file not f
36     return None, 41, 2 # Return None fo
37
38     logger.info("Loading and preprocessing
39     df = pd.read_csv(file_path)
40
41     # --- Preprocessing Step 1: Drop non-ge
42     columns to drop = ['StartTime', 'LastTi
43     df = df.drop(columns=columns_to_drop, e
44
45     # ... Feature and Label Separation ...

```

common_model.py

```

server > model_manager > common_model.py
1 # common_model.py
2
3 import torch
4 import torch.nn as nn
5 import torch.nn.functional as F
6
7 class SimpleFCN(nn.Module):
7
8     """
9     A Deeper Fully Connected Network (FCN)
10    Increased depth and width to improve mo
11    """
12    def __init__(self, input_size, num_clas
13        super(SimpleFCN, self).__init__()
14
15        # Increase the size of the first hi
16        self.fc1 = nn.Linear(input_size, 25
17
18        # Increase the size of the second h
19        self.fc2 = nn.Linear(256, 128)
20
21        # --- NEW: Added a third hidden lay
22        self.fc3 = nn.Linear(128, 64)
23
24        # Final output layer
25        self.fc4 = nn.Linear(64, num_classe
26
27    def forward(self, x):
28        # ... is a 1D feature vector for each
29        x = F.relu(self.fc1(x))
30
31        # --- NEW: Added the forward pass f
32        x = F.relu(self.fc3(x))
33
34        x = self.fc4(x)
35
36        return x
37
38    # Alias the new model to SimpleCNN for mini
39    SimpleCNN = SimpleFCN

```

model_manager.py

```

server > model_manager > model_manager.py
1 import os
2 import logging
3 import torch
4 import torch.nn as nn
5 from typing import Dict, Any, List, Optiona
6 import datetime
7 import json
8
9 # Import the new, secure aggregation functi
10 from sam.sam import aggregate_model_weights
11 from model_manager.common_model import Simp
12 from model_manager.data_loader import load
13
14 class ModelManager:
15     """
16     Manages the global model for the federat
17     This includes loading the initial model
18     and evaluating its performance on a test
19     """
20
21     def __init__(self, cfg: Dict[str, Any]):
22         """
23             Initializes the ModelManager with a
24
25             Args:
26                 cfg (Dict[str, Any]): The configuration
27
28             self.logger = logging.getLogger(self.cfg["n
29
30             # ... FIX 1: Load data and get dimensions
31             # load_cifar10_test_data is aliased
32             # (DataLoader, num_features, num_classes)
33             self.cifar10_test_loader, self.inpu
34
35             # Now, load_model() can access self.
36             self.global_model = self.load_model(
37                 self.global_model_version = 0
38
39             # Now attributes for model converge
40             self.best_accuracy = 0.0
41             self.rounds_since_last_improvement =
42             self.convergence_window = self.cfg["c
43             self.model_save_path = self.cfg.get(
44
45             # Create the model save directory

```

Shamir secret sharing

File Edit Selection View Go Run Terminal Help

FLF 2.0

EXPLORER

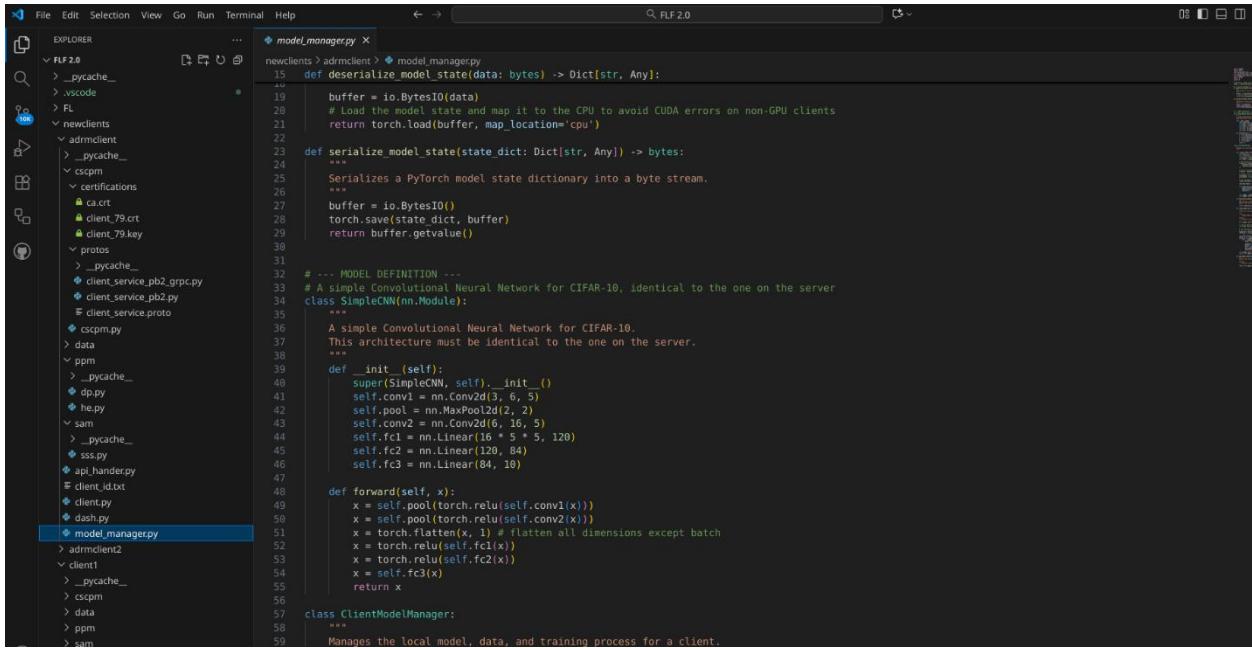
```

server > sam > sss.py
1 import torch
2 import numpy as np
3 import logging
4 import io
5 import json
6 from typing import Dict, Any, List
7 from collections import defaultdict
8
9
10 class SecretSharer:
11     """
12         Implements a simple Shamir's Secret Sharing algorithm using numpy.
13
14         prime = 104857601 # A large prime for modular arithmetic
15
16         @staticmethod
17         def mod_inverse(a: int, m: int) -> int:
18             """
19                 Computes the modular multiplicative inverse of a modulo m.
20
21             return pow(a, m - 2, m)
22
23         @staticmethod
24         def split(secret: int, k: int, n: int) -> List[tuple[int, int]]:
25             """
26                 Splits a secret into n shares, requiring k to reconstruct.
27
28             coeffs = [secret] + [np.random.randint(0, _SecretSharer.prime) for _ in range(k - 1)]
29             shares = []
30             for i in range(1, n + 1):
31                 y = sum(c * (i ** j) for j, c in enumerate(coeffs)) % _SecretSharer.prime
32                 shares.append((i, y))
33             return shares
34
35         # <<< FIX START >>>
36         # The original 'recover_secret' method had a flawed implementation of Lagrange interpolation.
37         # This new version uses the correct formula to calculate the Lagrange basis polynomials,
38         # ensuring the reconstructed integer is correct and preventing the OverflowError.
39         @staticmethod
40         def recover_secret(shares: List[tuple[int, int]]) -> int:
41             """
42                 Reconstructs the secret from a list of shares using Lagrange interpolation.
43
44             if len(shares) < 2:
45                 raise ValueError("At least two shares are required.")
46
47             secret = 0
48             x_coords, y_coords = zip(*shares)

```

SAM – Client

Model Manager

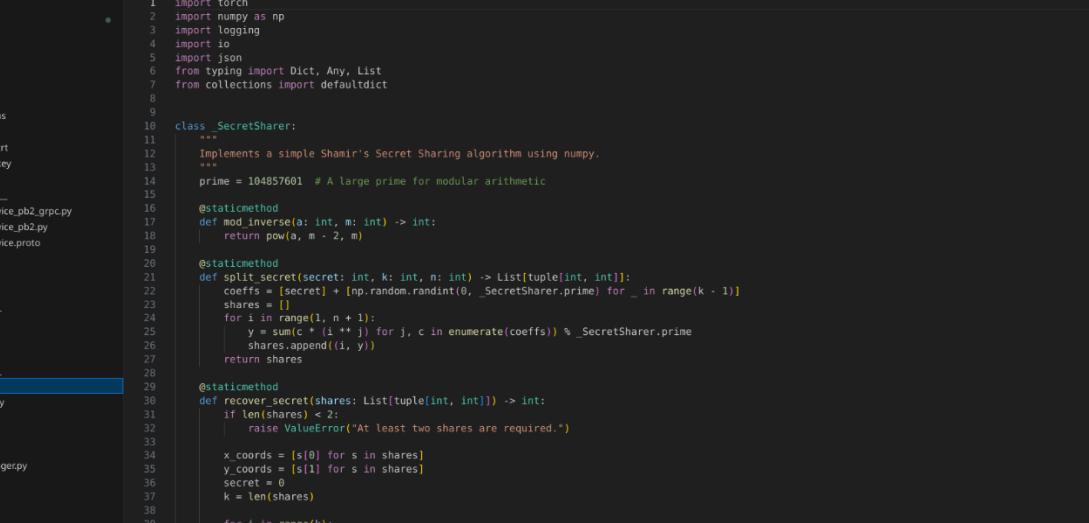


```

File Edit Selection View Go Run Terminal Help
EXPLORER model_manager.py ...
newclients > admclient > model_manager.py
15 def deserialize_model_state(data: bytes) -> Dict[str, Any]:
16     """
17         buffer = io.BytesIO(data)
18         # Load the model state and map it to the CPU to avoid CUDA errors on non-GPU clients
19         return torch.load(buffer, map_location='cpu')
20     """
21
22 def serialize_model_state(state_dict: Dict[str, Any]) -> bytes:
23     """
24         Serializes a PyTorch model state dictionary into a byte stream.
25         """
26     buffer = io.BytesIO()
27     torch.save(state_dict, buffer)
28     return buffer.getvalue()
29
30
31 # ... MODEL DEFINITION ...
32 # A simple Convolutional Neural Network for CIFAR-10, identical to the one on the server
33 class SimpleCNN(nn.Module):
34     """
35         A simple Convolutional Neural Network for CIFAR-10.
36         This architecture must be identical to the one on the server.
37     """
38
39     def __init__(self):
40         super(SimpleNN, self).__init__()
41         self.conv1 = nn.Conv2d(3, 6, 5)
42         self.pool = nn.MaxPool2d(2, 2)
43         self.conv2 = nn.Conv2d(6, 16, 5)
44         self.fc1 = nn.Linear(16 * 5 * 5, 120)
45         self.fc2 = nn.Linear(120, 84)
46         self.fc3 = nn.Linear(84, 10)
47
48     def forward(self, x):
49         x = self.pool(torch.relu(self.conv1(x)))
50         x = self.pool(torch.relu(self.conv2(x)))
51         x = torch.flatten(x, 1) # flatten all dimensions except batch
52         x = torch.relu(self.fc1(x))
53         x = torch.relu(self.fc2(x))
54         x = self.fc3(x)
55         return x
56
57     class ClientModelManager:
58         """
59             Manages the local model, data, and training process for a client.

```

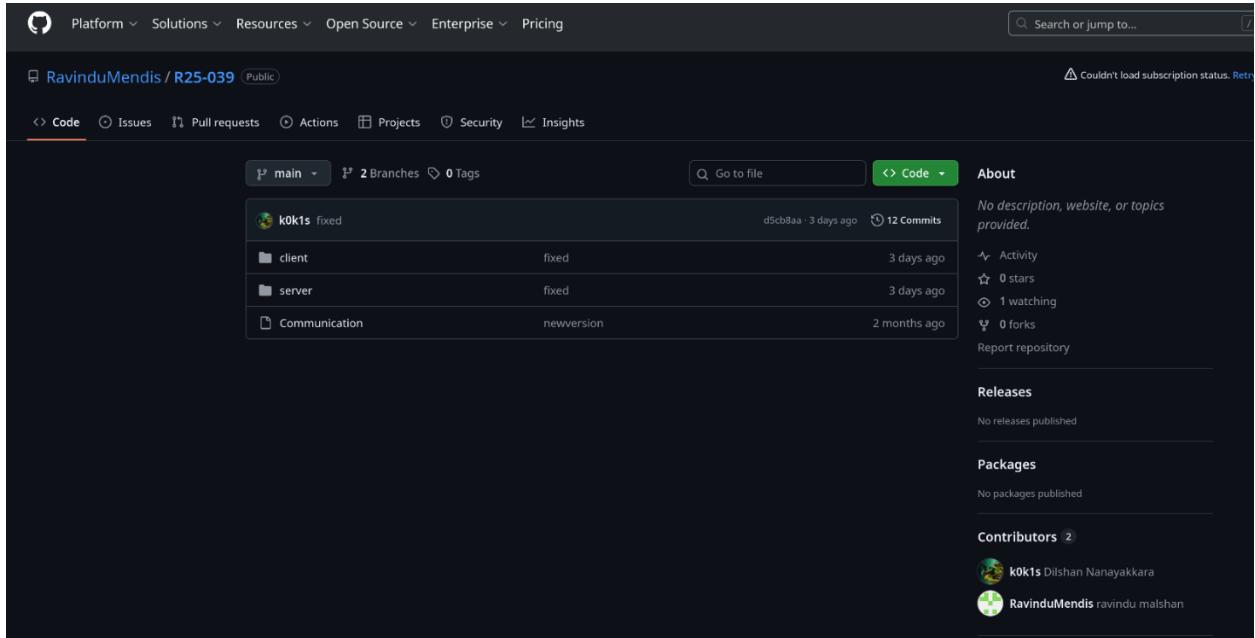
Sharmir secret Sharing



The screenshot shows a code editor with a Python file named `sss.py` open. The file contains the following code:

```
newclients > admclient > sam > sss.py
1 import torch
2 import numpy as np
3 import logging
4 import io
5 import json
6 from typing import Dict, Any, List
7 from collections import defaultdict
8
9
10 class SecretSharer:
11     """
12         Implements a simple Shamir's Secret Sharing algorithm using numpy.
13     """
14     prime = 104857601 # A large prime for modular arithmetic
15
16     @staticmethod
17     def mod_inverse(a: int, m: int) -> int:
18         return pow(a, m - 2, m)
19
20     @staticmethod
21     def split_secret(secret: int, k: int, n: int) -> List[tuple[int, int]]:
22         coeffs = [secret] + [np.random.randint(0, _SecretSharer.prime) for _ in range(k - 1)]
23         shares = []
24         for i in range(1, n + 1):
25             y = sum(c * (i ** j) for j, c in enumerate(coeffs)) % _SecretSharer.prime
26             shares.append((i, y))
27         return shares
28
29     @staticmethod
30     def recover_secret(shares: List[tuple[int, int]]) -> int:
31         if len(shares) < 2:
32             raise ValueError("At least two shares are required.")
33
34         x_coords = [s[0] for s in shares]
35         y_coords = [s[1] for s in shares]
36         secret = 0
37         k = len(shares)
38
39         for i in range(k):
40             numerator, denominator = 1, 1
41             for j in range(k):
42                 if i != j:
43                     numerator = (numerator * (-x_coords[j])) % _SecretSharer.prime
44                     denominator = (denominator * (x_coords[i] - x_coords[j])) % _SecretSharer.prime
45
46             secret += (y_coords[i] * numerator) // denominator
47
48         return secret
```

7. GitHub Upload



8. Documentation

8.1 Proposal



Data-Privacy Focused Federated Learning Framework for Industrial IoT

R25-039

Supervisor - Dr. Sanika Wijesekara
Co-Supervisor - Mr. Tharaniyawarma Kumaralingam

Project Group

Role	Team Member	ID
Supervisor		Dr. Sanika Wijesekara
Co-Supervisor		Mr. Tharaniyawarma Kumaralingam
Team		Nanayakkara Y.D.T.D IT21826368
		Mendis H.R.M IT21822612
		Weerasinghe K.M IT21831904
		Dissanayaka K.D.A.R.A IT21828348

8.2 Presentation 1

Contents

Search...

- Slide 1: Data-Priv... 1
- Slide 2: Project G... 2
- Slide 3: Introduc... 3
- Slide 4: Researc... 4
- Slide 5: Research ... 5
- Slide 6: Research ... 6
- Slide 7: System Ar... 7
- Slide 8: IT218263... 8
- Slide 9: Introduc... 9
- Slide 10: GAPS F... 10
- Slide 11 11
- Slide 12 12
- Slide 13: REQUIS... 13
- Slide 14: Propos... 14
- Slide 15: Compo... 15
- Slide 16: Compo... 16
- Slide 17 17
- Slide 18: Comple... 18
- Slide 19: Refere... 19
- Slide 20: IT2182... 20
- Slide 21: Conclus... 21
- Slide 22: Objecti... 22
- Slide 23: Resear... 23
- Slide 24: Metho... 24**
- Slide 25: System ... 25
- Slide 26: Compo... 26
- Slide 27: Compo... 27
- Slide 28: Functio... 28
- Slide 29: Visual B... 29
- Slide 30: Conclus... 30
- Slide 31: Refere... 31
- Slide 32: IT2183... 32
- Slide 33: Introduc... 33
- Slide 34: Resear... 34
- Slide 35: Resear... 35
- Slide 36: Resear... 36
- Slide 37: Objecti... 37
- Slide 38: Compos... 38
- Slide 39: Compo... 39
- Slide 40: Function... 40
- Slide 41: Work b... 41

There is a lack of large-scale IIoT datasets for testing privacy-preserving techniques under real-world conditions. Existing research often relies on synthetic data, limiting the generalizability of results.

Regulatory compliance Privacy-preserving methods need to be compliant with existing data protection laws like GDPR, but there's a lack of specific guidelines for IIoT systems.

Resistance to privacy attacks While current privacy-preserving methods are robust, the resilience of these methods against evolving privacy attacks in IIoT systems remains insufficiently addressed.

Energy consumption and efficiency Many privacy-preserving methods are computationally intensive, posing significant challenges to resource-constrained IoT devices, affecting their energy efficiency.

IT21822612 | Mendis H.R.M. | R25-039 23

Methodology

Approach:

- Analyze existing FL privacy vulnerabilities.
- Combine HE and DP for enhanced privacy.
- Optimize techniques for IIoT-specific constraints.
- Validate Using real-world Datasets

Key Techniques:

- Homomorphic Encryption (HE):** Encrypts gradients, allowing computations on encrypted data without decrypting it. Prevents data leakage even if adversaries intercept communications.
- Differential Privacy (DP):** Ensure that individual data points cannot be separated by adding controlled noise to gradients. Balances model accuracy with privacy.

IT21822612 | Mendis H.R.M. | R25-039 24

System Architecture



8.3 Presentation 2

Contents

Search...

- > OVERALL 1
- > ADMR COMPONENT... 15
- > COMPONENT 2 PRI... 23
- > COMPONENT 3 SEC... 33
- > COMPONENT 4 SCPM 40
- > GENERAL END SLIDES 48

DATA-PRIVACY FOCUSED FEDERATED LEARNING FRAMEWORK FOR INDUSTRIAL IOT

R25 - 039

PROJECT GROUP

Team	Supervisors
 <i>Nanayakkara Y.D.T.D</i> <i>IT21826368</i>	 <i>Mr. Amila Nuwan Senarathne</i> <i>Supervisor</i>

8.4 Final Presentation

Contents

Search...

- Slide 1: DATA-PRIVACY ... 1
- Slide 2: PROJECT ... 2
- Slide 3: FRAMEW... 3
- Slide 4: PROBLEMS 4
- Slide 5: SOLUTIONS 5
- Slide 6: SYSTEM W... 6
- Slide 7: System W... 7
- Slide 8: Attack De... 8
- Slide 9: RESEARC... 9
- Slide 10: SOLUTI... 10
- Slide 11: ... 11
- Slide 12: Termin... 12
- Slide 13: ... 13
- Slide 14: Attack... 14
- Slide 15: DETECT... 15
- Slide 16: Logging 16
- Slide 17: Privacy ... 17
- Slide 18: ... 18
- Slide 19: SOLUTI... 19
- Slide 20: PPM Ac... 20
- Slide 21: METHO... 21
- Slide 22: FUNCTL... 22
- Slide 23: FUNCTL... 23
- Slide 24: PROOF... 24
- Slide 25: ... 25
- Slide 26: Secure ... 26
- Slide 27: ... 27
- Slide 28: ... 28
- Slide 29: Module... 29
- Slide 30: SERVER... 30
- Slide 31: OVERA... 31
- Slide 32: Log ma... 32
- Slide 33: Secure ... 33
- Slide 34: ... 34
- Slide 35: ... 35
- Slide 36: METHO... 36
- Slide 37: ... 37
- Slide 38: SAM M... 38
- Slide 39: AGGRE... 39
- Slide 40: USER T... 40
- Slide 41: ... 41
- Slide 42: We are ... 42
- Slide 43: Minus O... 42

**DATA-PRIVACY
FOCUSED
FEDERATED LEARNING FRAMEWORK
FOR
INDUSTRIAL IOT**



Final Presentation
R25 - 039

PROJECT GROUP
Supervisors

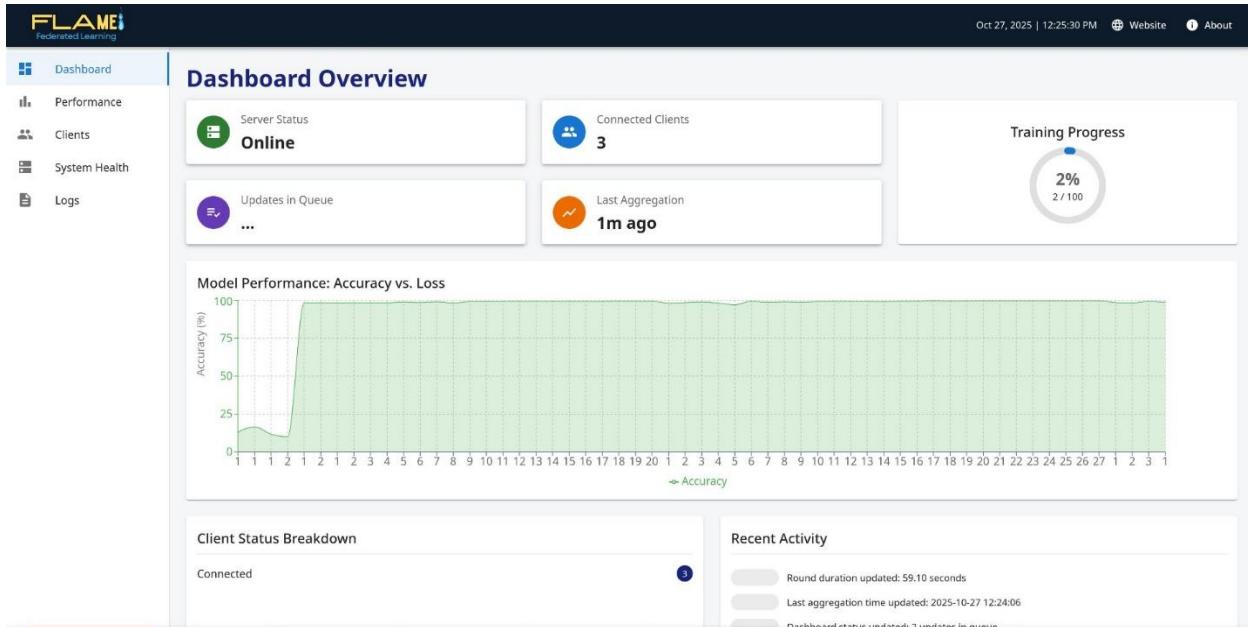
	Mr. Amila Nuwan Senarathne Supervisor		Mr. Tharanikarma Kyunaratilagam Co-Supervisor
---	--	---	---

Team

	Mendis H.R.M IT21822612	Dissanayaka K.D.A.R.A IT21828348	Weerasinghe K.M IT21831904
---	----------------------------	-------------------------------------	-------------------------------

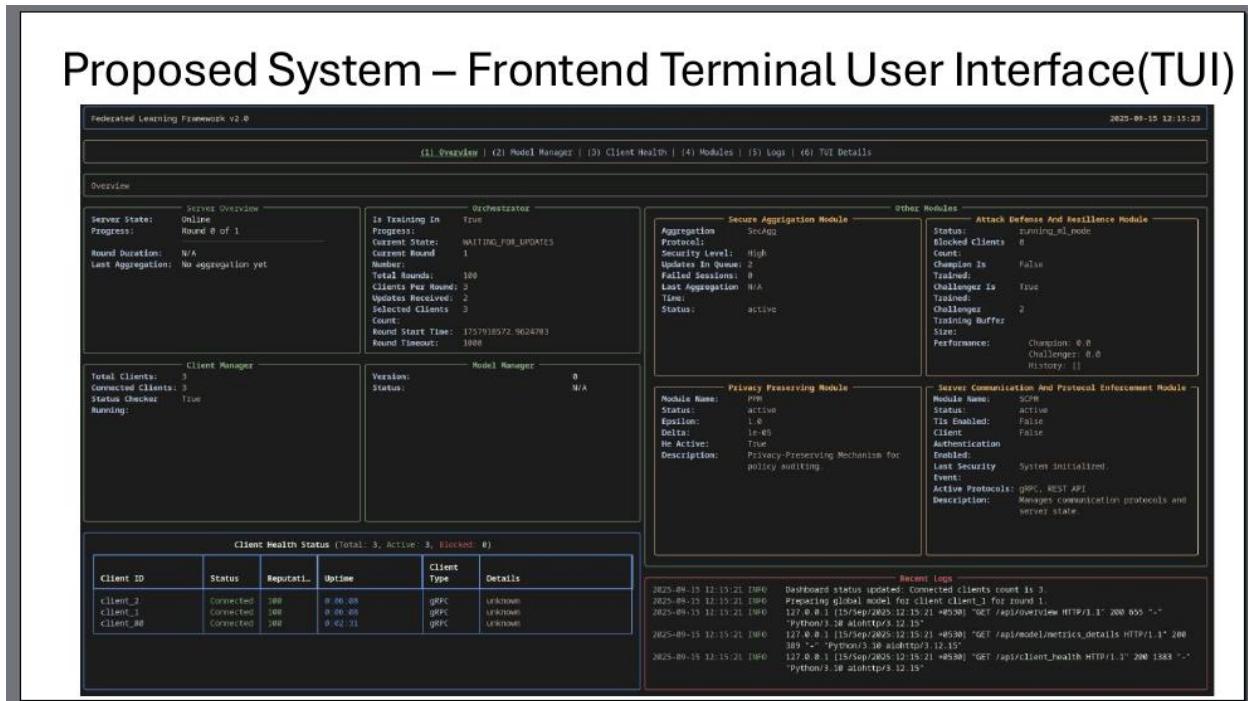
8.5 Final Product

Web Portal Frontend



Terminal User interface (Frontend)

Proposed System – Frontend Terminal User Interface(TUI)



9 Research Paper

I. Conference Apperence

To Tharindu D Nanayakkara <dilz.nanayakkara@gmail.com> @

10/29/25, 11:27 AM

Acceptance Notification

Dear Tharindu D Nanayakkara,

Congratulations! We are pleased to inform you that your paper has been accepted as a regular paper to be presented at the 7th International Conference on Advancements in Computing 2025.

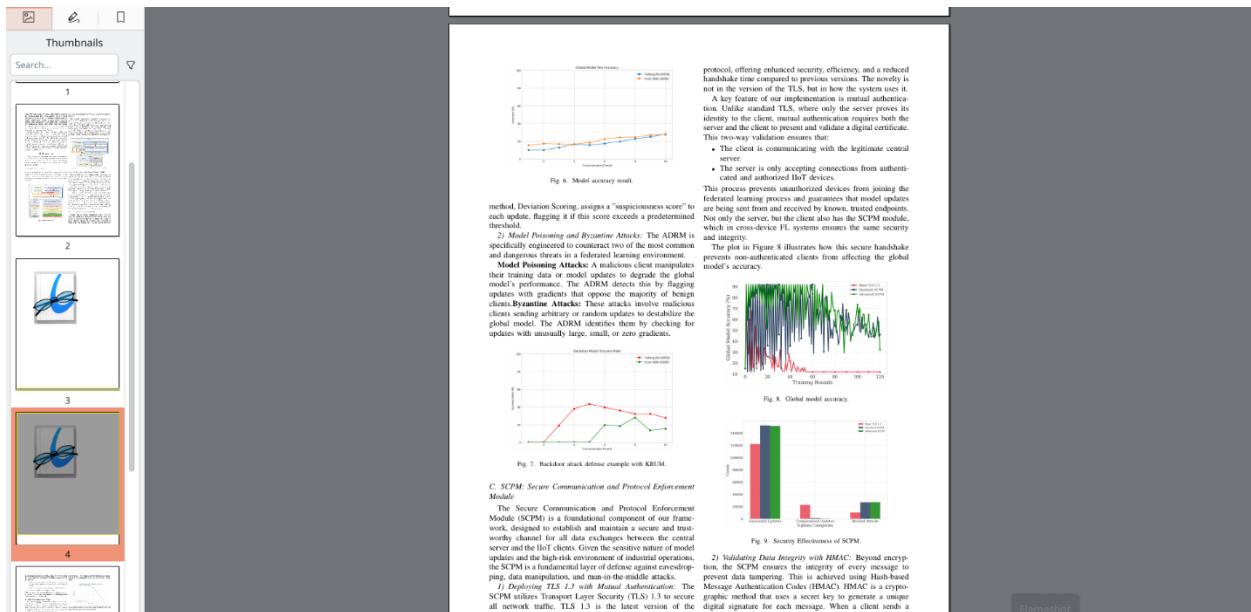
Paper ID: 469

Paper Title: Data-privacy based Federated Learning Framework for Industrial IOT

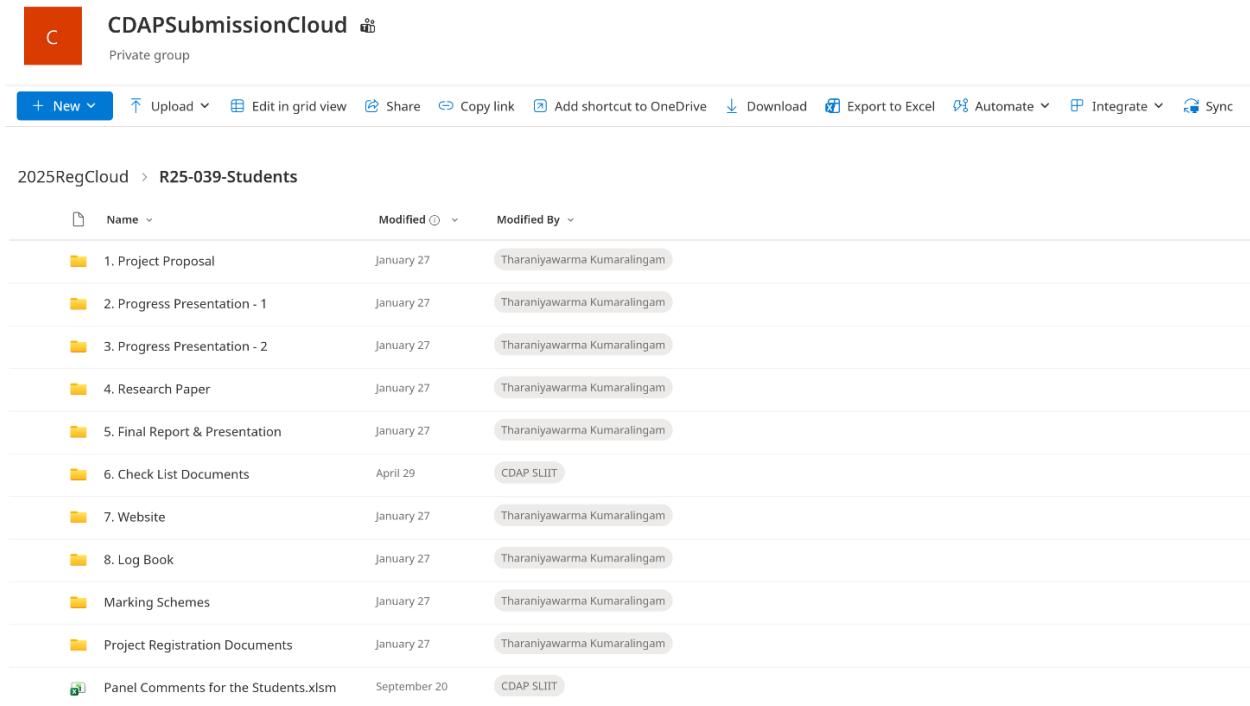
Please visit <https://cmt3.research.microsoft.com/7ICAC2025/Submission/Index> to view the reviews given during the double-blind review process.

When preparing the camera-ready version of your paper, please address all the review comments and follow the camera-ready guidelines given in the <https://icac.lk/for-authors>

Please note that the camera-ready deadline is 10th November 2025 and camera-ready submission portal on CMT will be available starting from 22nd October 2025.



9. CDAP upload



The screenshot shows a OneDrive interface with the following details:

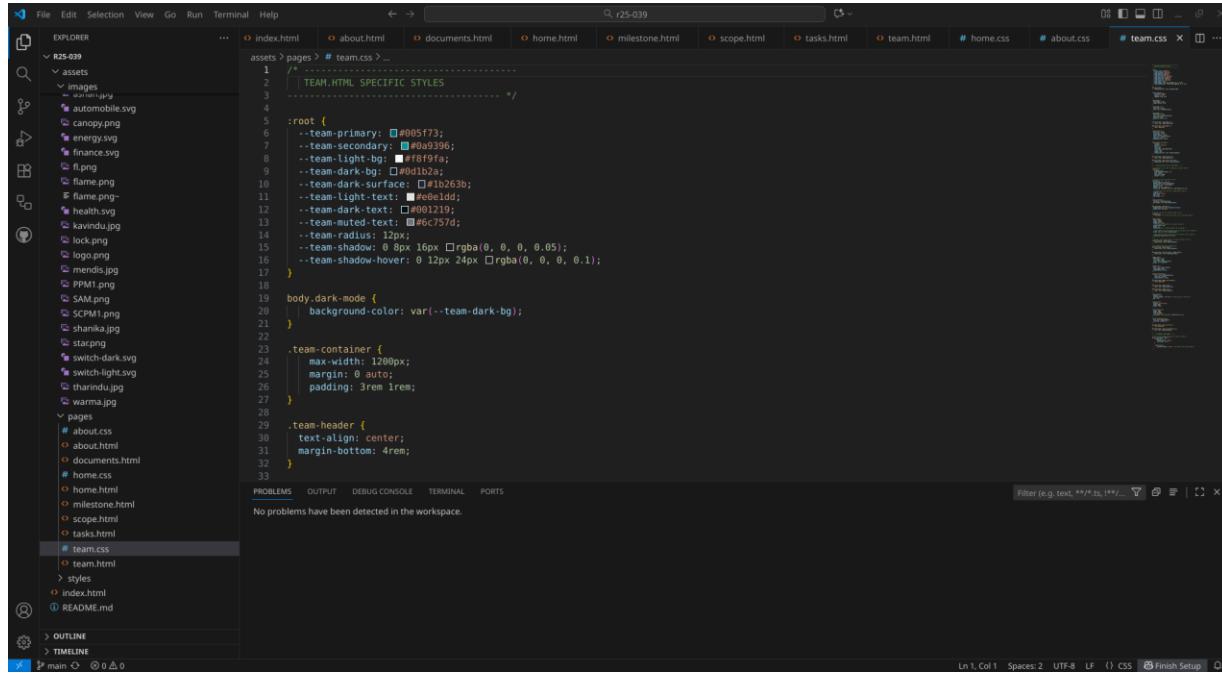
CDAPSubmissionCloud (Private group)

2025RegCloud > R25-039-Students

Name	Modified	Modified By
1. Project Proposal	January 27	Tharaniyawarma Kumaralingam
2. Progress Presentation - 1	January 27	Tharaniyawarma Kumaralingam
3. Progress Presentation - 2	January 27	Tharaniyawarma Kumaralingam
4. Research Paper	January 27	Tharaniyawarma Kumaralingam
5. Final Report & Presentation	January 27	Tharaniyawarma Kumaralingam
6. Check List Documents	April 29	CDAP SLIIT
7. Website	January 27	Tharaniyawarma Kumaralingam
8. Log Book	January 27	Tharaniyawarma Kumaralingam
Marking Schemes	January 27	Tharaniyawarma Kumaralingam
Project Registration Documents	January 27	Tharaniyawarma Kumaralingam
Panel Comments for the Students.xlsx	September 20	CDAP SLIIT

10. Website

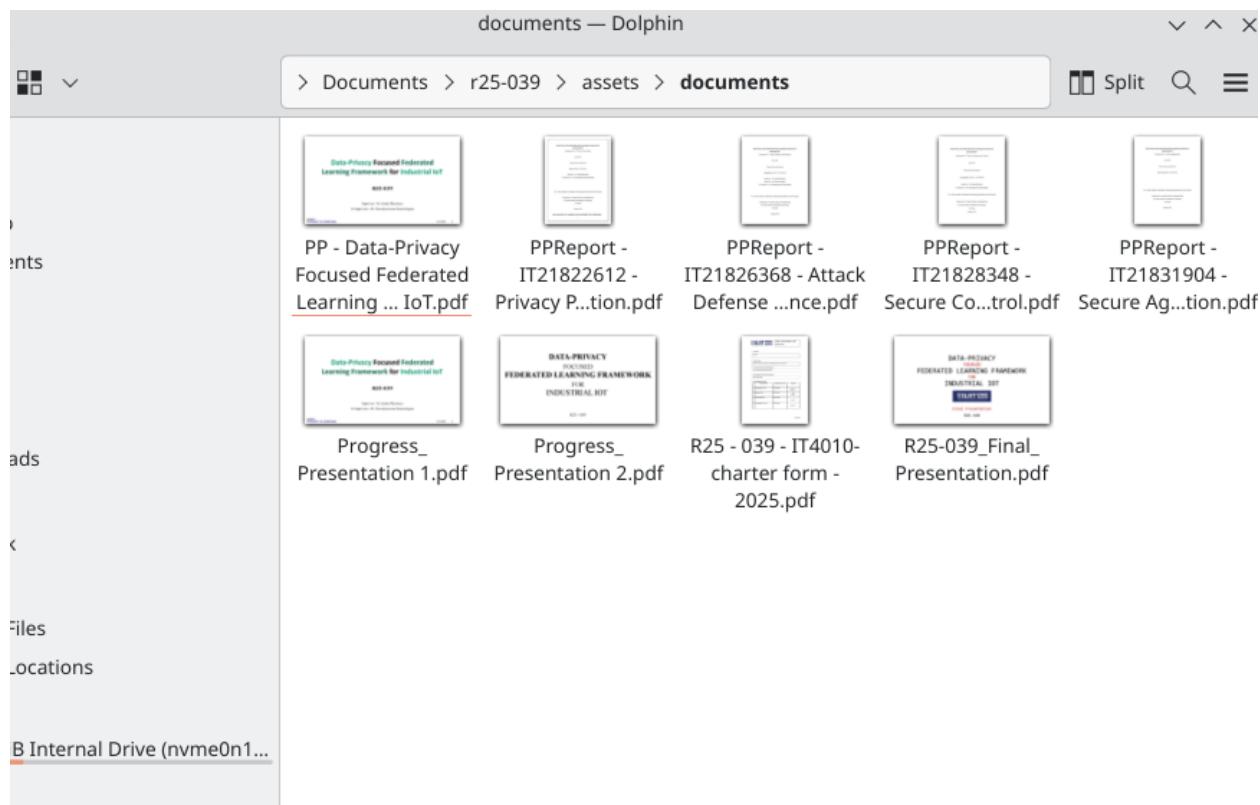
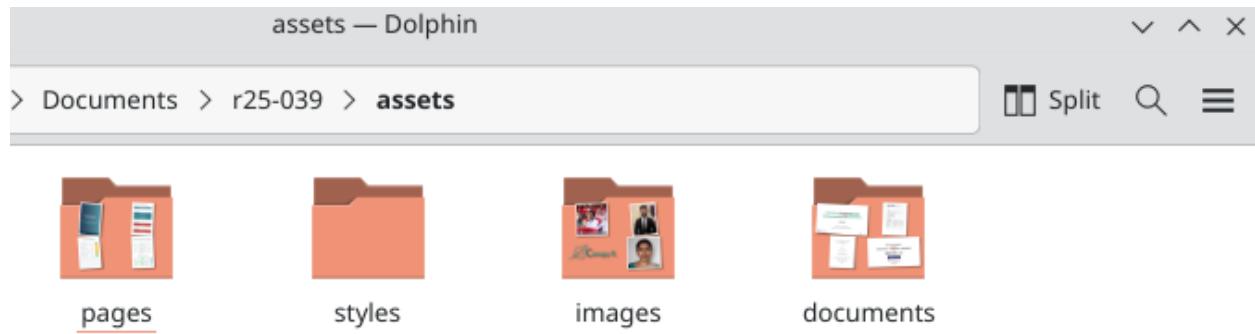
10.1 Development

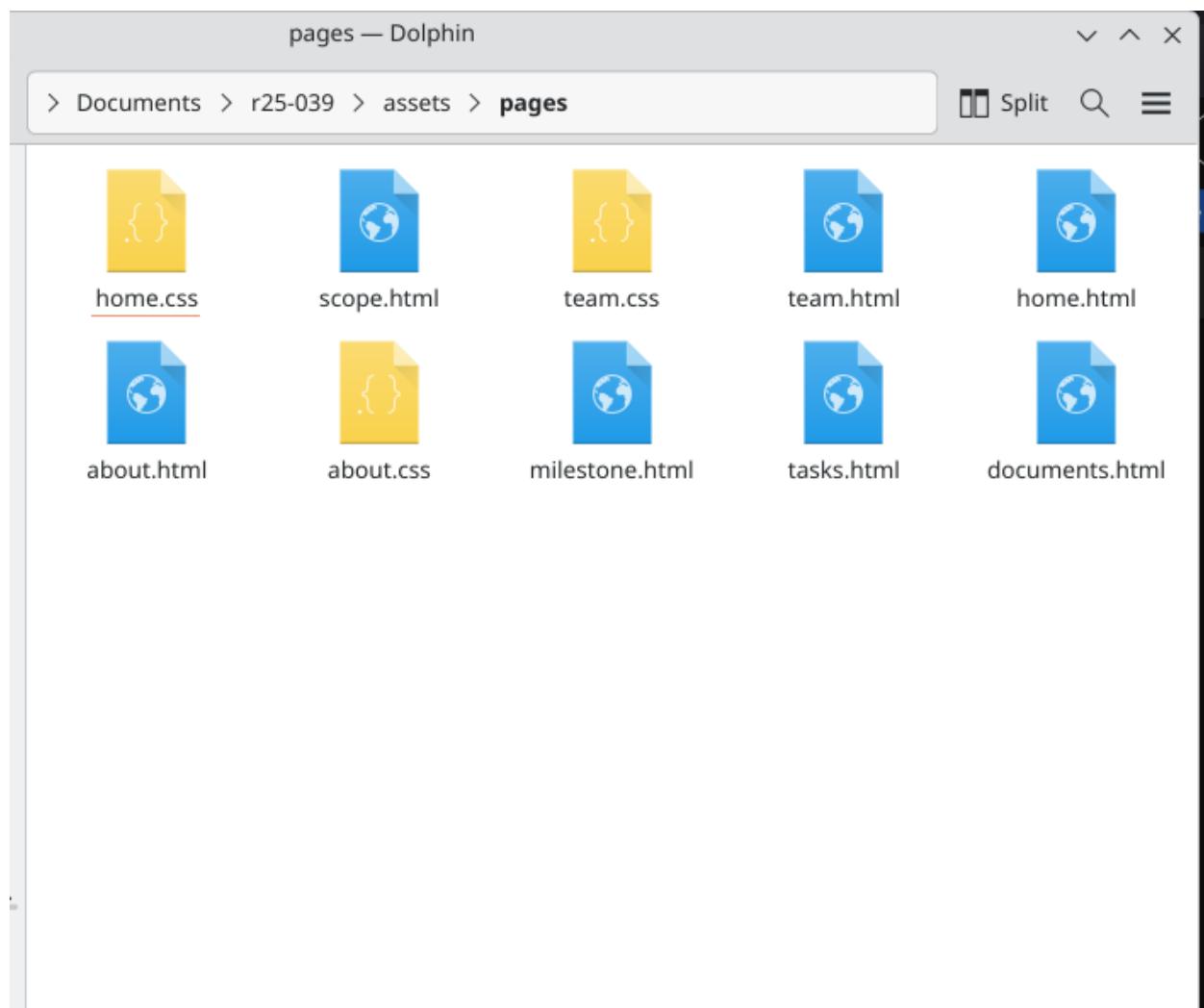


```

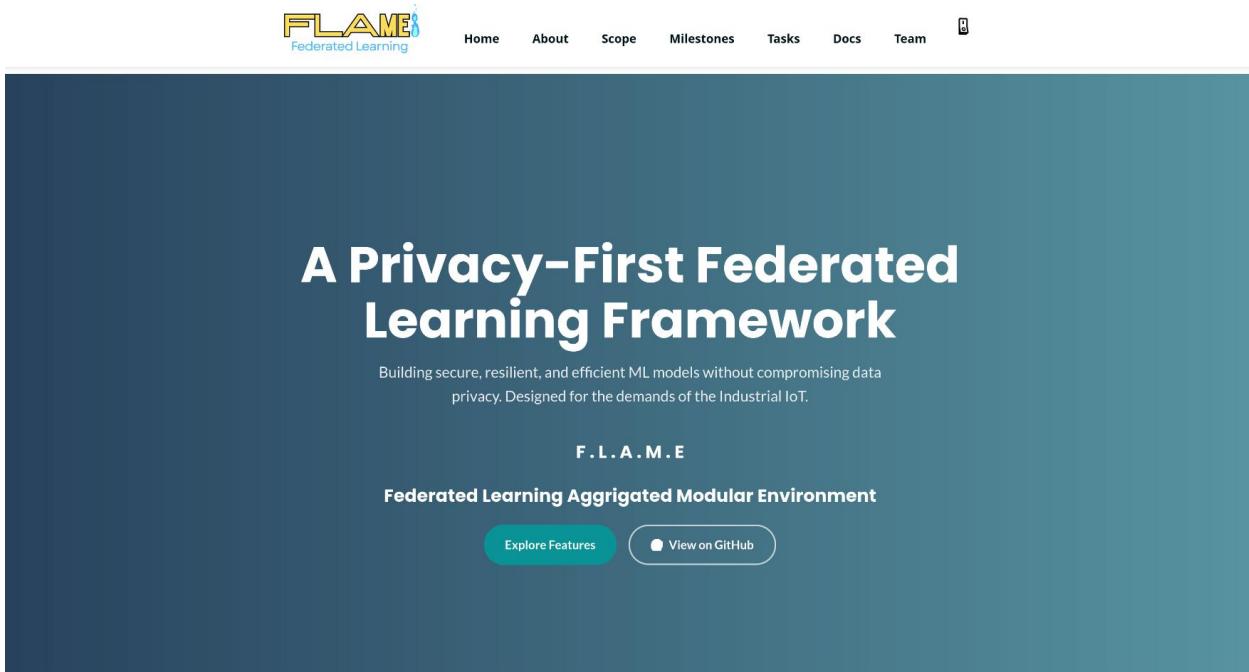
1  /*-----*
2   |TEAM.HTML SPECIFIC STYLES
3   *-----*/
4
5  :root {
6      --team-primary: #005f73;
7      --team-secondary: #0a9396;
8      --team-light-bg: #f8f9fa;
9      --team-dark-bg: #d1b2e;
10     --team-dark-surface: #1b263b;
11     --team-light-text: #e6e0d;
12     --team-dark-text: #001210;
13     --team-radius: 12px;
14     --team-shadow: 0 8px 16px rgba(0, 0, 0, 0.05);
15     --team-shadow-hover: 0 12px 24px rgba(0, 0, 0, 0.1);
16 }
17
18 body.dark-mode {
19     background-color: var(--team-dark-bg);
20 }
21
22 .team-container {
23     max-width: 1200px;
24     margin: 0 auto;
25     padding: 3rem 0;
26 }
27
28 .team-header {
29     text-align: center;
30     margin-bottom: 4rem;
31 }
32
33

```





10.2 Finalize

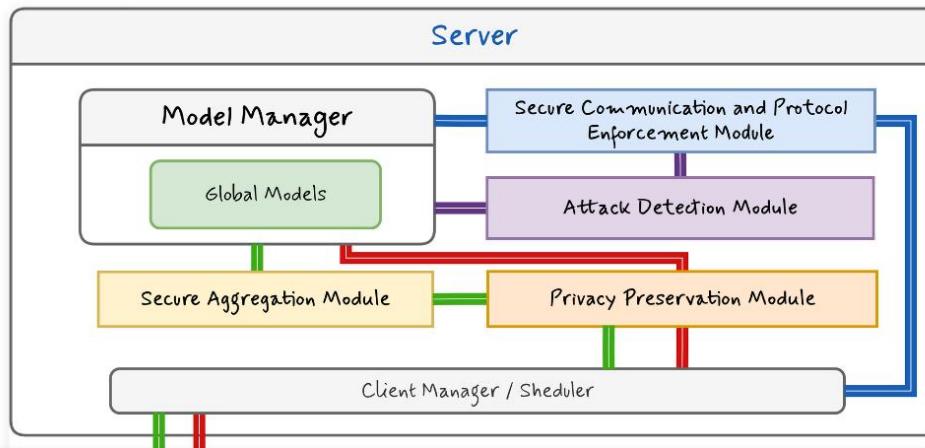


The screenshot shows the FLAME Federated Learning website. At the top, there is a navigation bar with links for Home, About, Scope, Milestones, Tasks, Docs, Team, and a search icon. The main title is "A Privacy-First Federated Learning Framework". Below the title, a subtitle reads: "Building secure, resilient, and efficient ML models without compromising data privacy. Designed for the demands of the Industrial IoT." The acronym "F.L.A.M.E" is displayed above the full name "Federated Learning Aggregated Modular Environment". There are two buttons at the bottom: "Explore Features" and "View on GitHub".

Privacy-Enhanced Federated Learning Framework

Our comprehensive Federated Learning (FL) System Framework is engineered to significantly augment the privacy, security, and operational resilience of machine learning models deployed in decentralized and distributed environments. The framework is composed of four interconnected core modules, collectively guaranteeing data integrity, defense against adversarial attacks, and authenticated inter-component communication.

Framework Overview and Architecture



**Mr. Amila Senerathne**

Supervisor

**Dr. Sanika Wijesekara**

External Supervisor

**Mr. T. Kumaralingam**

Co-Supervisor

