MICHIGAN STATE UNIVERSITY

Project Plan Presentation

High Frequency Ingestion System

The Capstone Experience

Team GM

Dave Yonkers
David Karlavage
Kory Gabrielson
Yunxiang Zhang
Kevin Zhong
Joseph Kasza

Department of Computer Science and Engineering
Michigan State University

Spring 2022

From Students…
…to Professionals

Functional Specifications

- Primary goal is to ingest CSV data files from network attached storage into a database
  - Approximately 12,000 OnStar call agents moved from call centers to at-home offices two years ago
  - Network statistics from OnStar agents are regularly sent to GM servers, however, the current database ingestion process easily becomes backlogged
  - When backlogged, data is lost, so an efficient ingestion application must be researched and created
  - Network and hardware bottlenecks need to be efficiently mitigated
- Simulation of data influx is necessary
  - No access to GM servers, so the capstone team must create a high-bandwidth CSV data generator
- Descriptive logging statistics and network data must be visualized

Design Specifications

- CSV Data generator
  - Command line interface
  - Customization options:
    - Rate of generation
    - Number of files with errata
    - Number of threads to use to create the files
    - Where the files should be stored
- Data Ingestion Application
  - Command line interface
  - Customization options:
    - Maximum number of threads to use
    - File indexing batch size
    - Location of the files
- Data visualization
  - Interactive web dashboard
  - Built using Flask and Python
  - Can review user data and ingestion application statistics

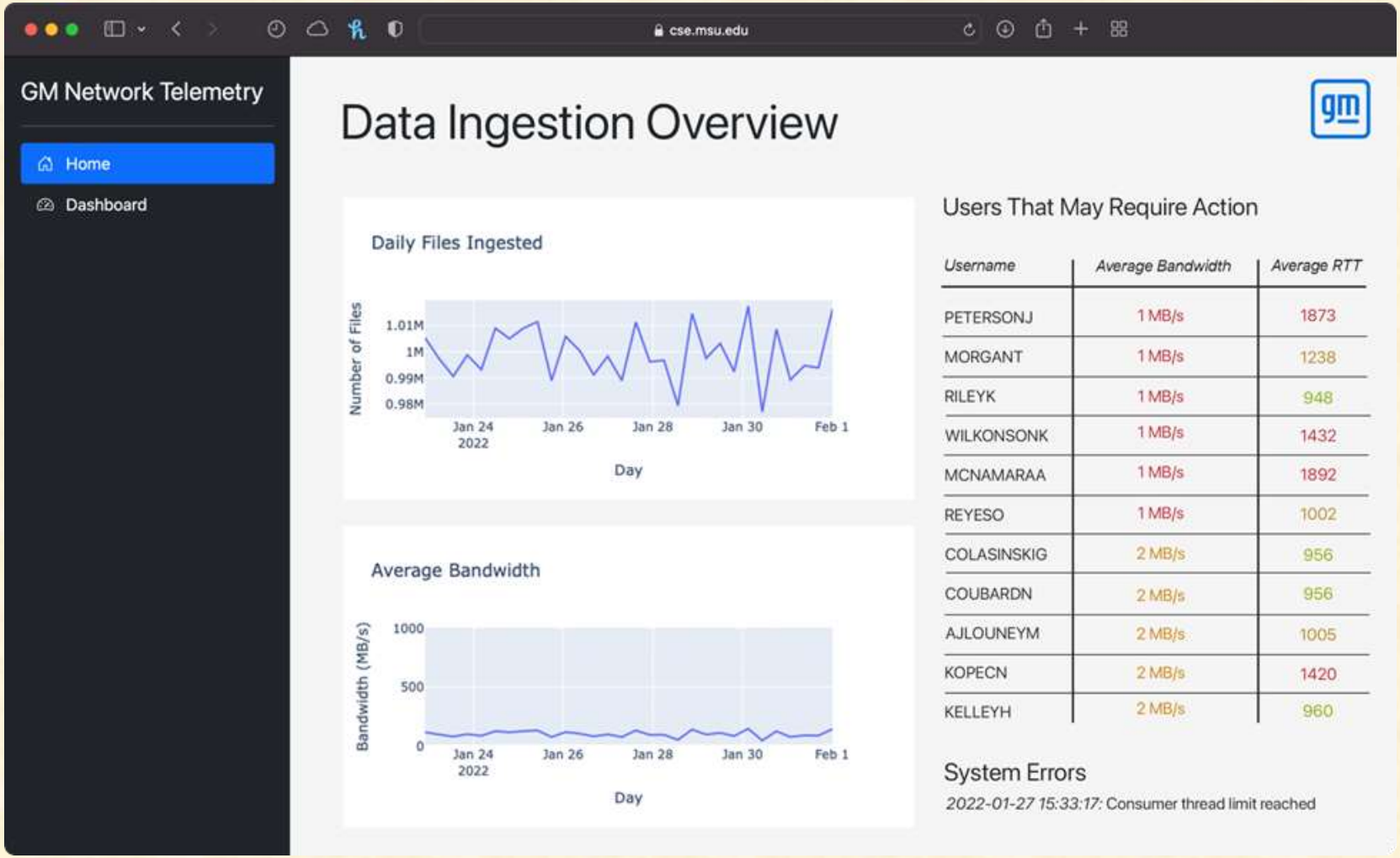
Screen Mockup: Data Generator CLI

```
1   Windows PowerShell
2   Copyright (C) Microsoft Corporation. All rights reserved.
3
4   Try the new cross-platform PowerShell https://aka.ms/pscore6
5   PS C:\TelemetryData\DataGenerator> py dataGenerator.py --help
6
7   usage:
8
9   dataGenerator.py [-f <path>] [-h | --h] [-r | --rate] [-t | --thread]
10
11  PS C:\TelemetryData\DataGenerator> py dataGenerator.py -f C:/TelemetryData/DataGenerator/DataFiles/ -r 100 -t 8
12  PS C:\TelemetryData\DataGenerator>
```

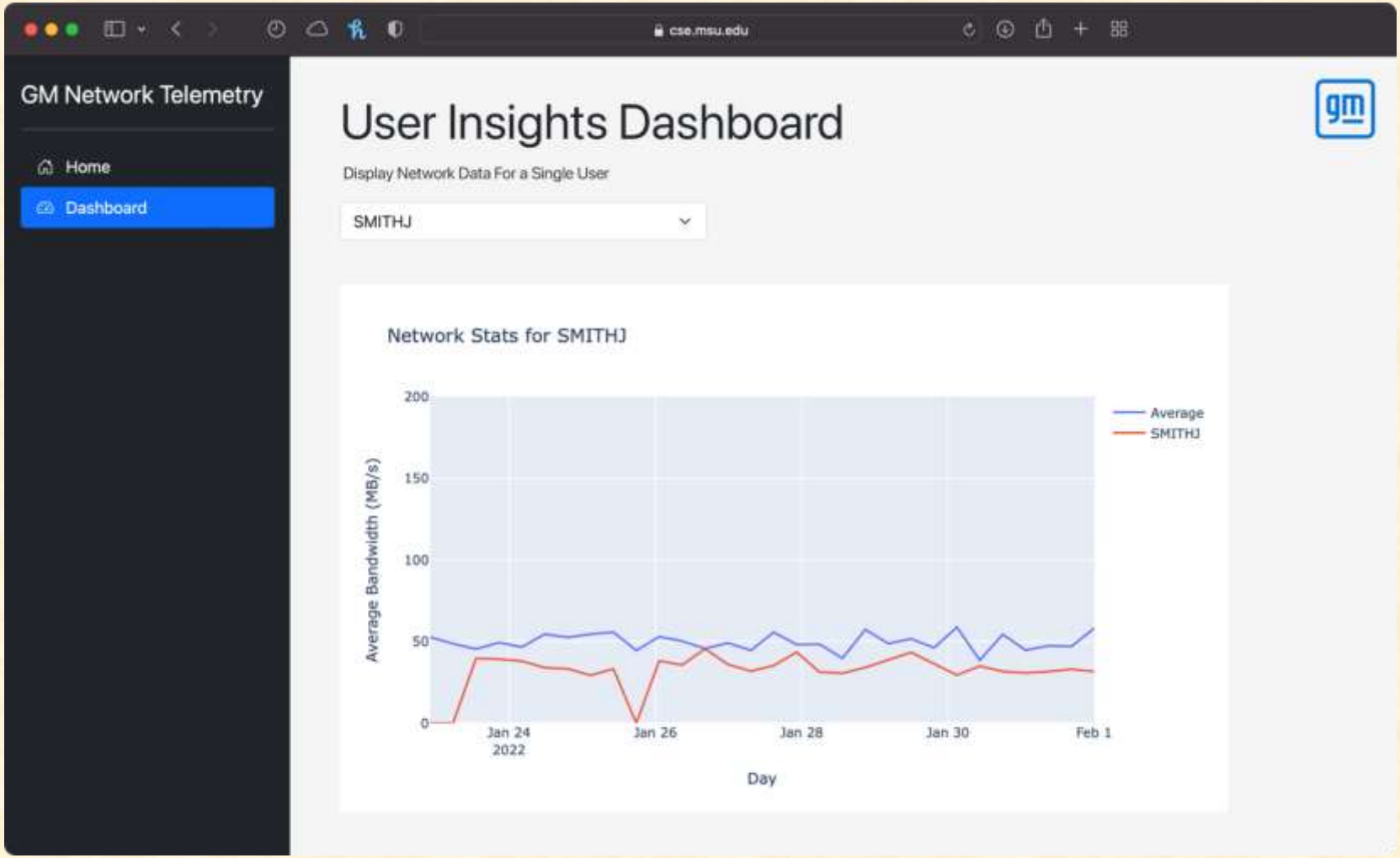
Screen Mockup: Data Ingestion CLI

```
1   Windows PowerShell
2   Copyright (C) Microsoft Corporation. All rights reserved.
3
4   Try the new cross-platform PowerShell https://aka.ms/pscore6
5   PS C:\TelemetryData\DataIngestion> gcc dataIngestion.cs --help
6
7   usage:
8
9   dataIngestion.cs [-f <path>] [-h | --h] [-t | --thread] [-b | --batch]
10
11  PS C:\TelemetryData\DataIngestion> gcc dataIngestion.cs -f C:\TelemetryData\DataGenerator\DataFiles -t 8 -b 50
12  PS C:\TelemetryData\DataIngestion>
```

Screen Mockup: Ingestion Dashboard



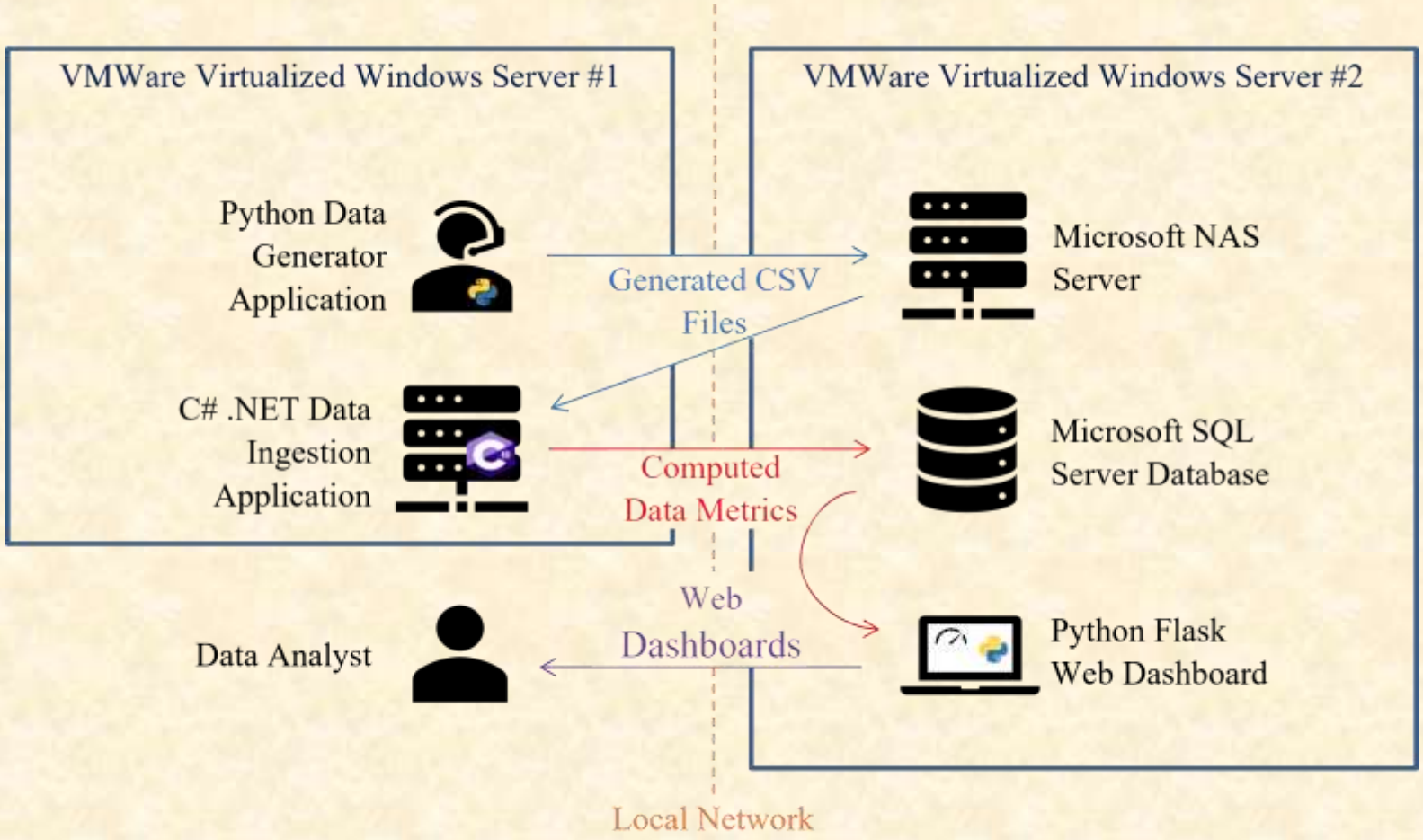
Screen Mockup: User Specific Dashboard



Technical Specifications

- Data Generator
  - Written in Python
  - Must generate at least 3,000 files per minute
  - Will be capable of generating data errata
  - Utilizes multithreading to increase throughput
- Data Ingestion Application
  - Written in C# .NET environment to utilize Microsoft file indexing utilities
  - Will feature status, speed, and error logging
  - Must be able to ingest files equivalent to the rate that they are generated (upper goal of 1,000,000 files per minute for future program expansion)
- Web Dashboard
  - Written in Python Flask with interactive plotting by Plotly
  - Will display average user data, individual user data, file ingestion statistics, and ingestion errors
  - Interactive elements to select individual users will be present
  - Ability to add on business insights in the future

System Architecture



System Components

- Hardware Platforms
  - Capstone Lab iMac #1
    - Contains the data generator and sends all generated files to the network file share on iMac #2
    - Encompasses the data ingestion program that pulls the data files from iMac #2
  - Capstone Lab iMac #2
    - Hosts the network file share, Microsoft SQL server, and Flask dashboard
- Software Platforms / Technologies
  - Microsoft SQL Server
    - Stores all valid telemetry data sent from the data ingestion
  - Python Flask Web Dashboard
    - Tool used to display the data from the SQL database in a user-friendly way

Risks

- Replicating the Bottleneck
  - We may not be able to replicate GM's file I/O bottleneck issue with the NAS
  - We may be able to artificially create a bottleneck by limiting VM resources
  - We will also be able to test our system on GM's hardware
- Working Around Hardware, Network, and Software Constraints
  - Since Microsoft's software is proprietary, the basic file I/O operations cannot be modified. Therefore, the perfect solution may not exist.
  - The team will find the optimal solution given the hardware, network, and software constraints. Since the optimal solution may not be perfect, an efficient way to manage backlogged file operations must be found.
- Status Logging Congestion
  - Introducing other processes, such as status logging, may further congest the file and database I/O.
  - If the logging is within the existing SQL server, the tables and logging metrics must be efficient. If the logging is external to the SQL server, a solution must be chosen that provides minimal impact to the file I/O and data processing.

Questions?