Senior Design Team 15: Debugger and Visualizer for a Shared Sense of Time on Batteryless Sensor Networks

Adam Ford
Allan Juarez
Riley Thoma
Anthony Rosenhamer
Quentin Urbanowicz
Maksym Nakonechnyy

Client/Advisor: Dr. Henry Duwe

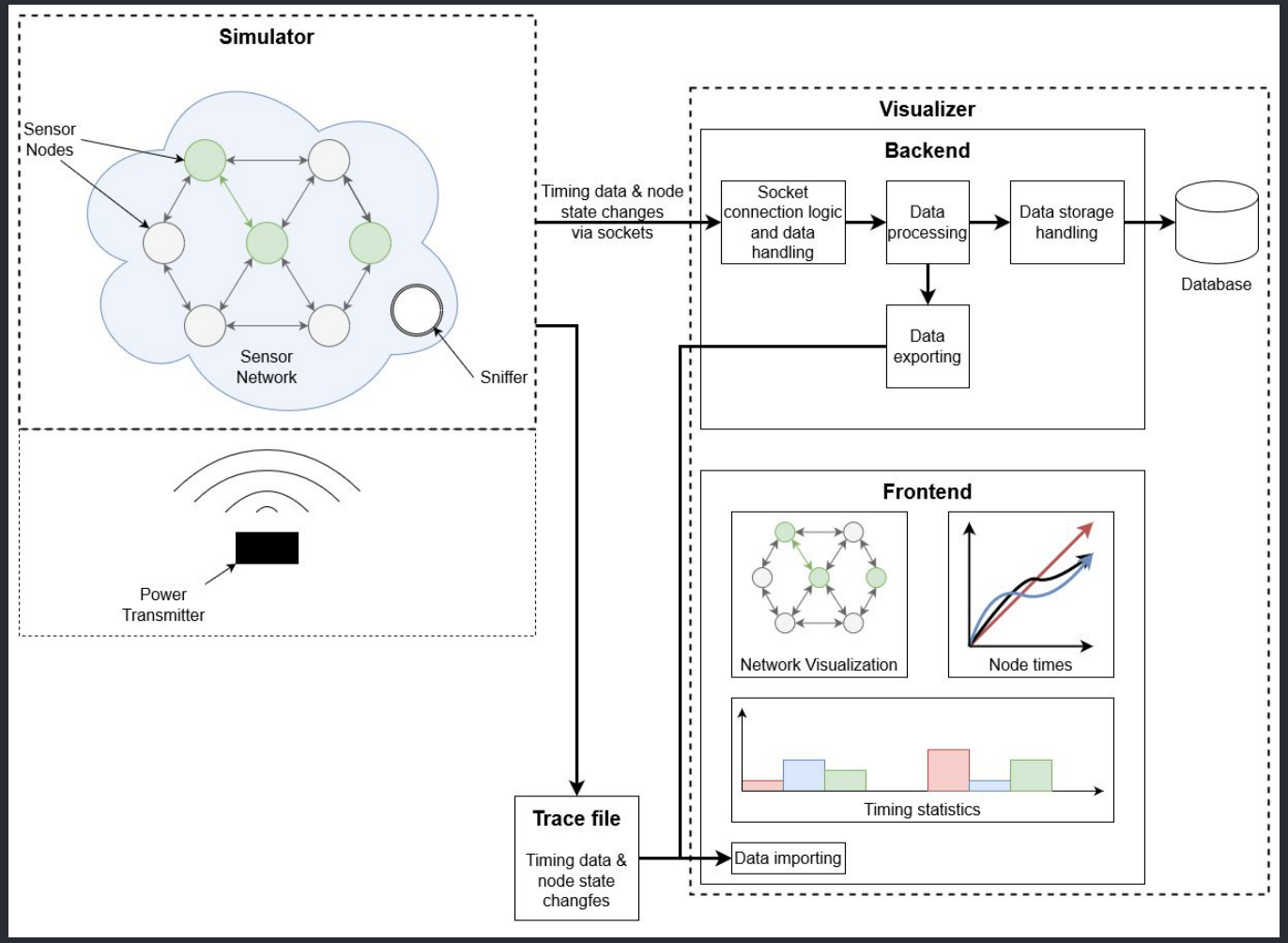
Project Vision

Our goal is to create a set of software tools enabling researchers to simulate, visualize, and debug shared timekeeping in batteryless sensor networks. Furthering this research may enable a wider range of sensing applications and a better connected, more sustainable world through the Internet of Things.

Conceptual Diagram

- Simulator
  - Models a sensor network
- Visualizer
  - Displays details about the sensor network

- Built for a research team
- Customized to test various approaches for maintaining a shared sense of time



System Overview

Functional Requirements

Simulator

- Shall generate the data in the same format as real data.
- Shall accept a seed value for pseudo-random simulation.

Visualizer (Frontend)

- Shall “replay” past data.
- Shall visualize the statistics of system communication.

System-Wide

- Shall monitor which nodes are currently communicating.
- Shall store past data.

Non-functional Requirements

Simulator

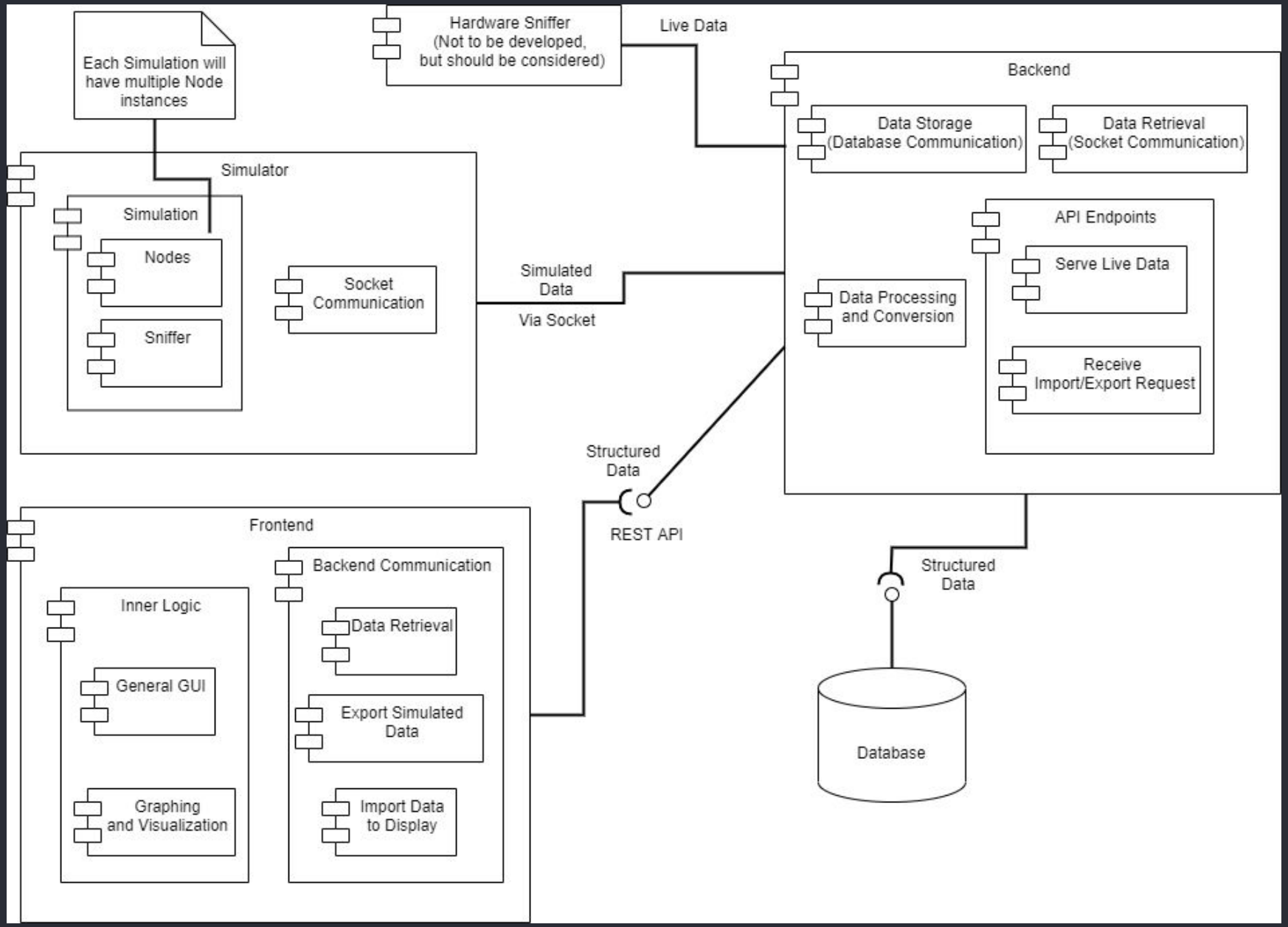
- The simulator shall run natively in a Linux environment.
- The simulator shall maintain sub-second accuracy of timing.
- The simulator shall produce on-time/off-time data from a user-provided function.

Visualizer

- The visualizer shall update node status every second.
- The visualizer shall be implemented as a web application.

System-wide

- The system shall be modular to allow for maintainability.
- The system shall not lose any sensor readings.



System Block Diagram

Project Plan - Task Decomposition

- Tasks are subdivided into design, simulator, frontend, and backend work
- Simulator
  - Data inputting and outputting
  - Node development
  - Sniffer development
- Frontend
  - One main path with some parallels
- Backend
  - One path for data handling
  - One path for API-related work

Task Graphs

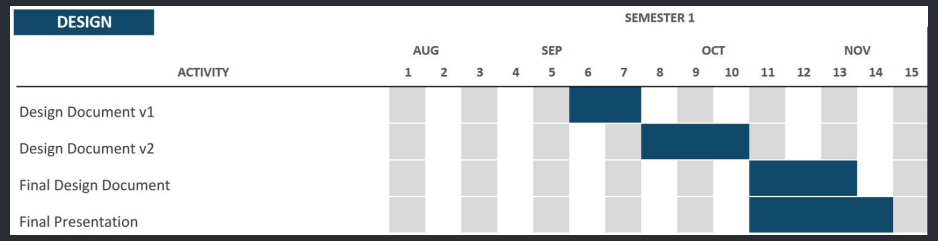
Project Plan - Risk Management

- Provide live current data
  - Data needs to be generated, processed, and displayed
  - Main components working in real time
  - Sub-one second latency requirement
  - Conscious of efficiency when coding the components

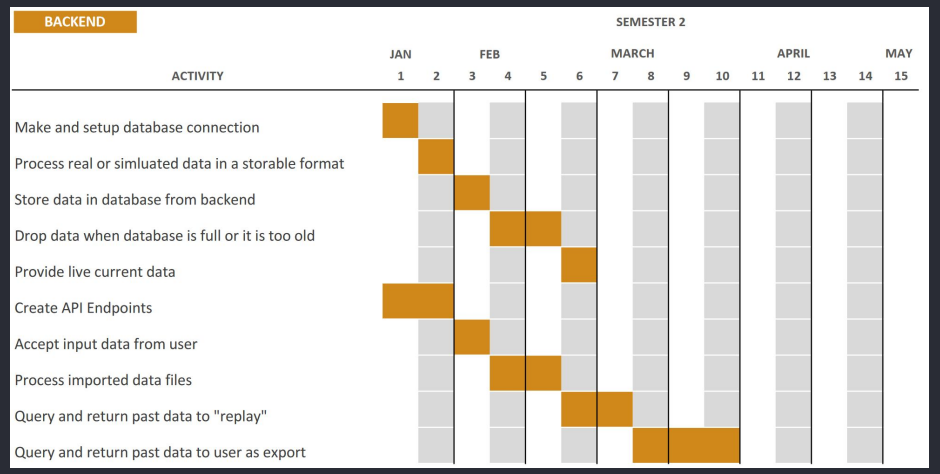
- Query and return past data to “replay”
  - Retrieve old data to display again
  - Mitigate by prototyping the replay feature
  - Future proof to make sure data is saved and easily retrievable

Project Plan - Timeline

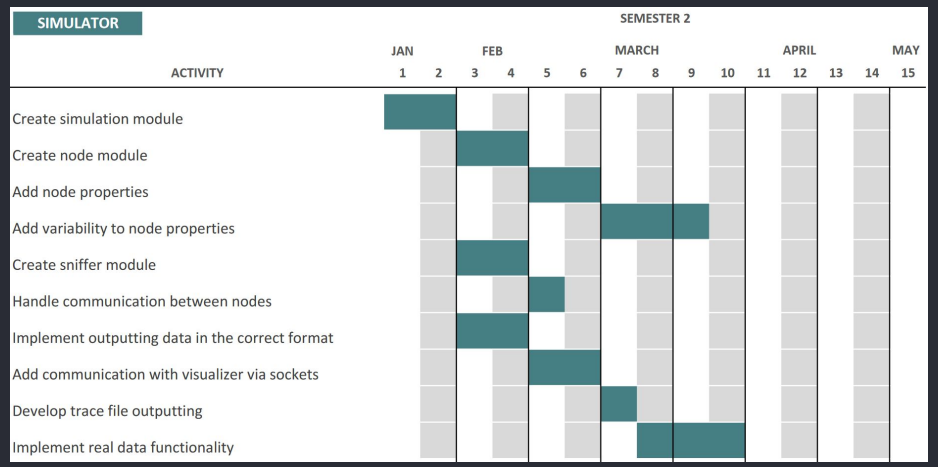
- Gantt charts show task timelines for each section of the project



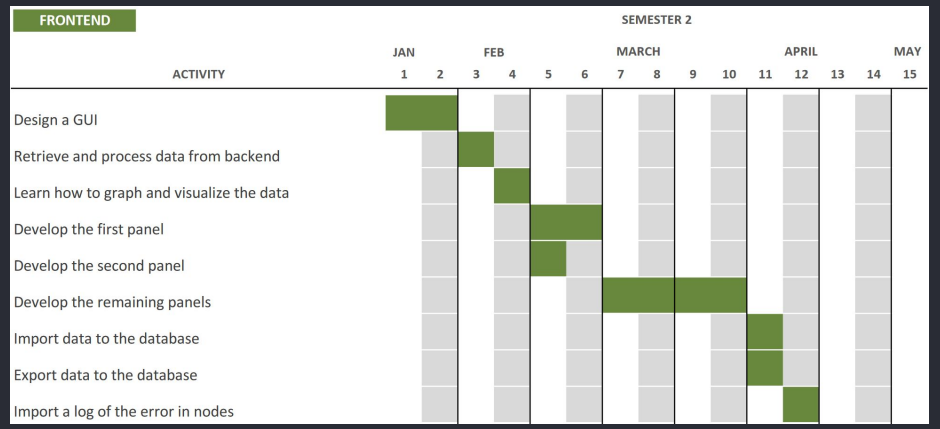
Design Gantt Chart



Backend Gantt Chart



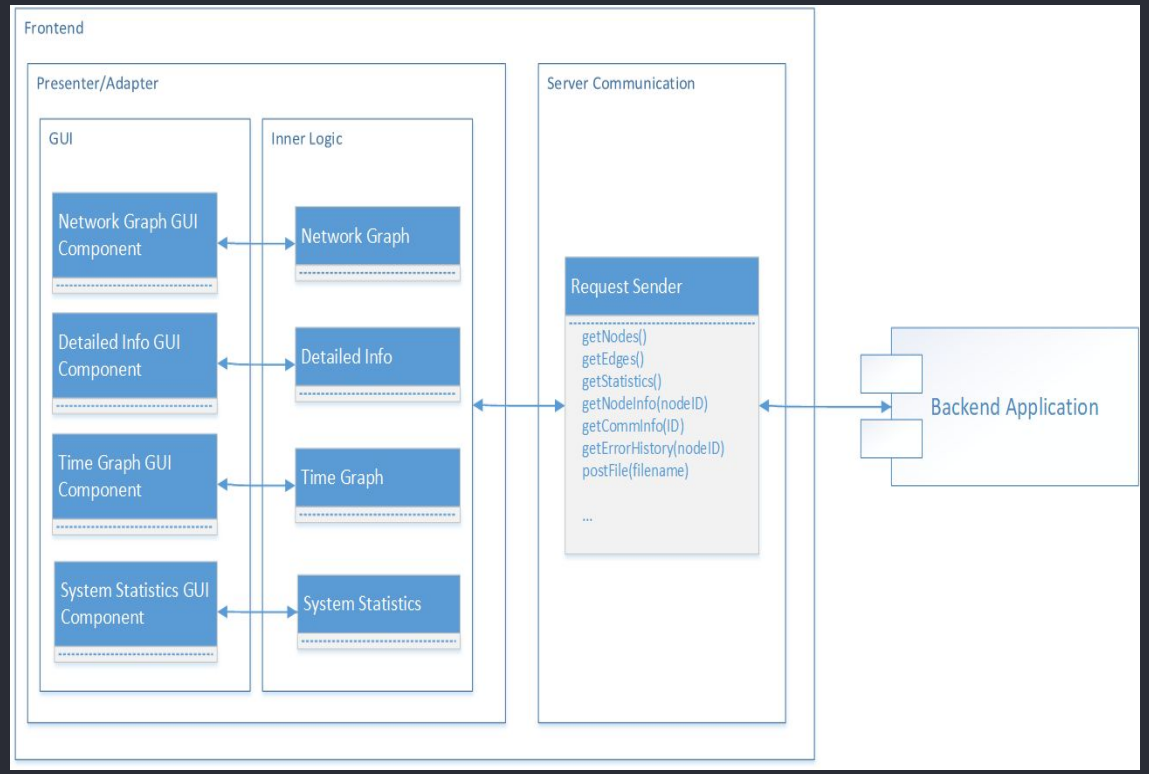
Simulator Gantt Chart



Frontend Gantt Chart

System Design - Frontend

- Consists of two logical components:
  - Presenter/Adapter
  - Server Communicator
    - Independent
    - Communicate via callbacks and EventBus
- Communication with the backend via HTTP.
- Technologies:
  - HTML & CSS
  - React JavaScript
  - Jest, Selenium, Mirage JS

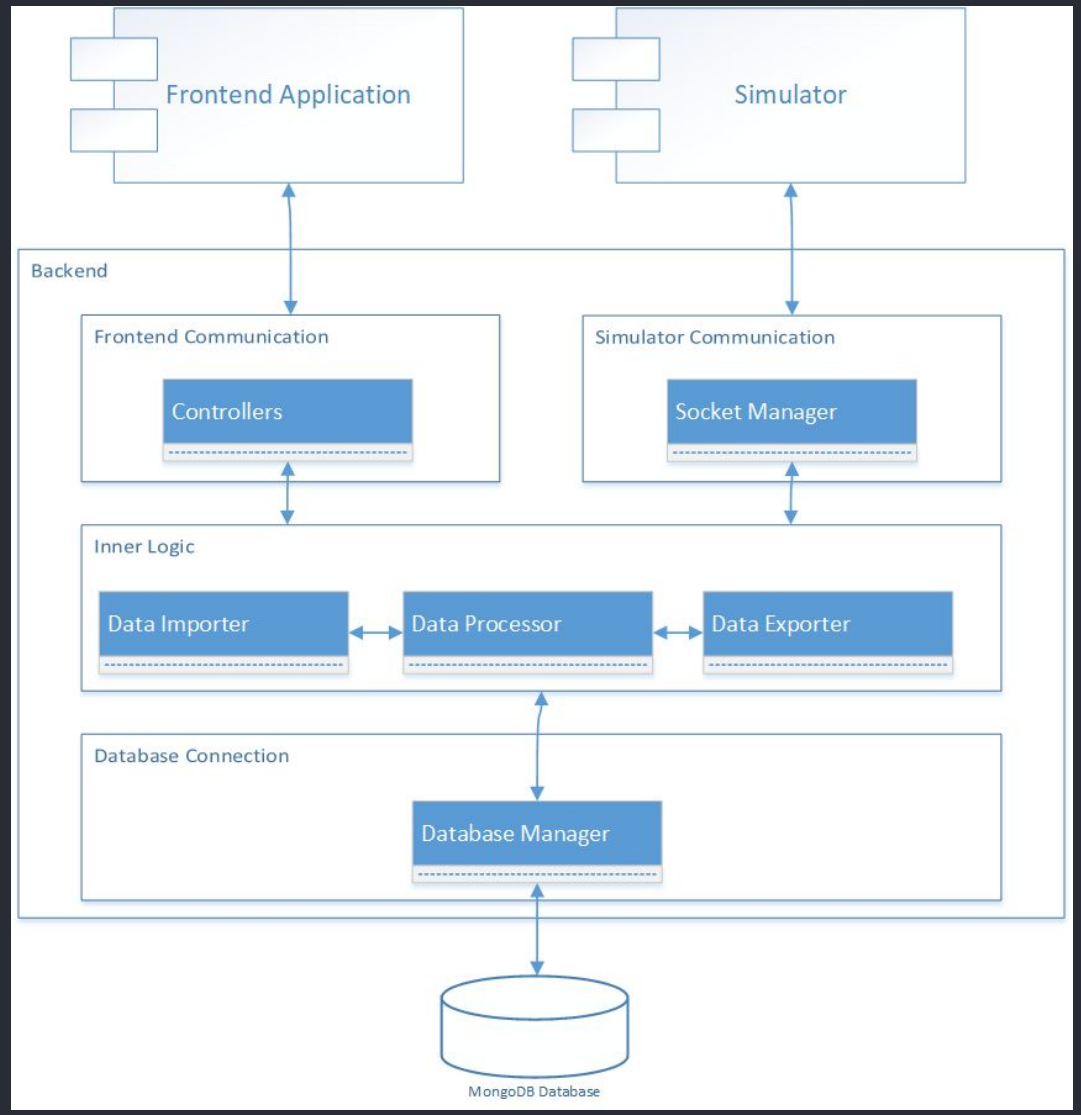


Frontend Block Diagram

System Design - Backend

- In the backend we will have two developers
  - One for working on the API's to the frontend and simulator
  - The other developer will work on the data handling making sure the data is cleaned and stored.

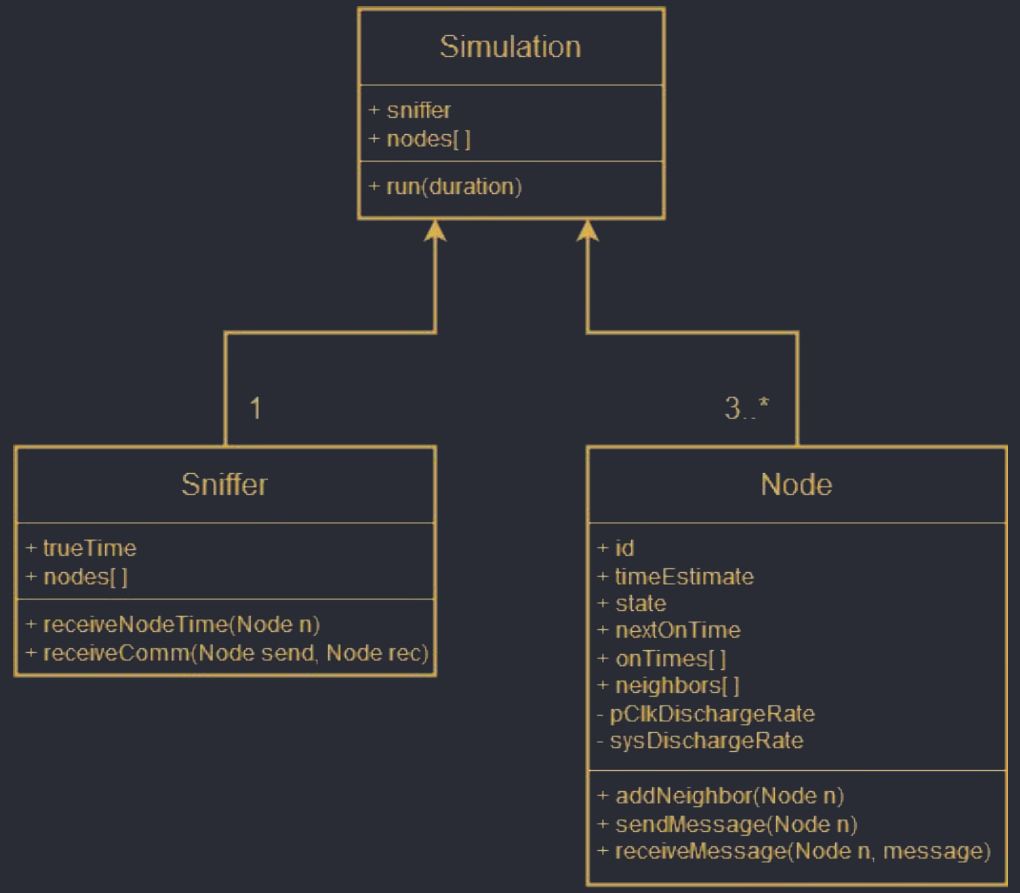
- Technologies
  - ExpressJS
  - Jest
  - MongoDB



Backend Block Diagram

System Design - Simulator

- Python classes to represent the simulation, sniffer, and nodes
- Uses discrete-event simulation to model the life of the sensor network
- Technologies
  - Python
  - SimPy
  - socket library
  - pytest



Simulator Class Diagram

Prototype Demo

Project Plan – Milestones

Major

- Design Document Final Draft
- Minimum Viable Product built
- Simulator Complete
- Frontend Complete
- Backend Complete
- Integration Complete (Final Release)

Minor

- Simulator algorithm working and producing data
- Individual Frontend panels created and working
- GUI is fully designed
- Importing and Exporting functionality is complete
- Database can store simulator data and provide it to the frontend
- Database can store past data for replayability
- Database set up and working

Test Plan

- Frequently unit test each application component individually with mock objects
- Interface and integration testing with mock objects and example data
  - Visualizer frontend and backend (REST API)
  - Simulator and visualizer backend (sockets)
  - Simulator and visualizer frontend (trace file export & import)
- System-level testing with example data and generated data
- Acceptance testing with client for each feature implemented

Current Plans

- Wrapping up the designing of the project
- Also started working on all three sections of the project(Simulator, Backend, Frontend)

What's next?

- We are on schedule to finish the project
- Quentin and Tony will work on the Simulator module
- Adam and Allan will work on and finish the Backend module
- Maksym and Riley will work on and finish the Frontend module

Thank you!

Questions?