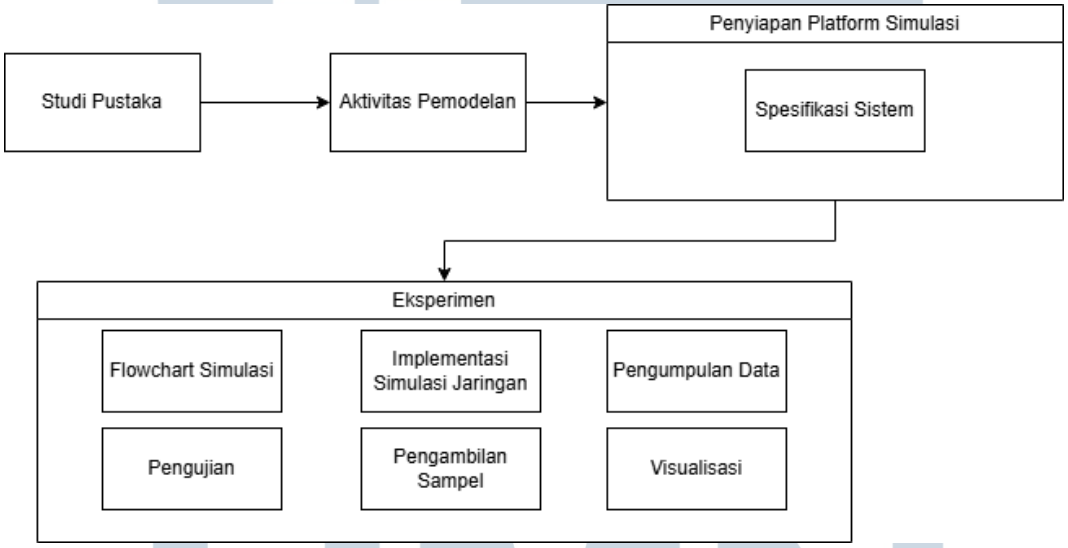
BAB 3
METODOLOGI PENELITIAN

Bagian ini menguraikan secara sistematis tahapan yang akan ditempuh dalam pelaksanaan penelitian yang digunakan seperti pada gambar 3.1 . Penelitian ini bertujuan untuk menganalisis kinerja protokol MQTT terhadap transmisi data sensor menggunakan parameter QoS, *bottleneck bandwidth*, dan jenis data (BPR dan Acc). Untuk itu, digunakan pendekatan simulasi menggunakan *Python script* dan *network simulator* yang menghasilkan data kuantitatif berupa latensi.



Gambar 3.1. Alur Metodologi Penelitian

3.1 Studi Pustaka

Penelitian ini didasarkan pada berbagai studi sebelumnya yang membahas sistem komunikasi INA-CBT, penggunaan protokol MQTT, serta pemodelan dan evaluasi kinerja jaringan komunikasi data. Kajian pustaka dilakukan untuk menelusuri pendekatan, metode, dan temuan utama dari penelitian-penelitian yang relevan, sekaligus mengidentifikasi celah dan keterbatasan yang masih ada. Ringkasan hasil kajian pustaka yang digunakan sebagai dasar perumusan penelitian ini disajikan pada Tabel 3.1.

Tabel 3.1. Studi pustaka

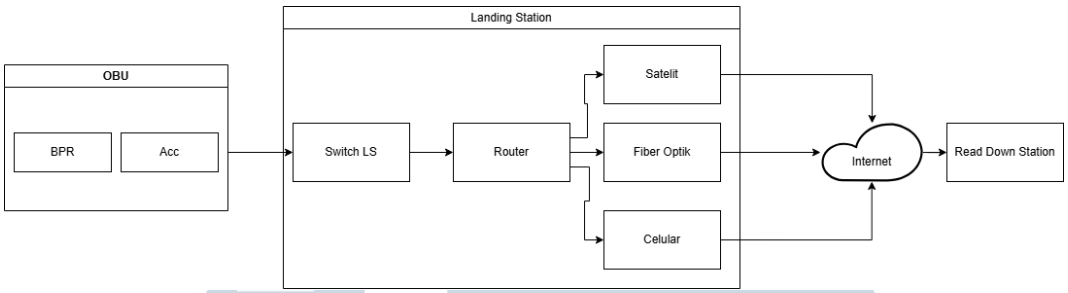
No	Judul	Fokus dan Metode	Temuan dan Keterbatasan
1	Estimating MQTT performance in a virtual testbed of INA-CBT communication sub-system [4]	Evaluasi kinerja MQTT pada sistem INA-CBT menggunakan virtual testbed berbasis perangkat fisik, simulator OBU, dan router mikrotik dengan variasi *bottleneck bandwidth*.	*Bottleneck bandwidth* berpengaruh signifikan terhadap latensi dan antrian pesan MQTT pada berbagai level QoS. Pengujian masih dilakukan pada lingkungan jaringan yang terbatas.
2	MQTT live performance on the INA-CBT communication system: a measurement-based evaluation [1]	Evaluasi performa MQTT pada sistem INA-CBT melalui pengukuran langsung (*live measurement*) antara *Landing Station* dan *Read Down Station*.	Memberikan gambaran performa aktual MQTT pada jaringan operasional INA-CBT. Namun, penelitian tidak mengeksplorasi skenario *bottleneck bandwidth* dan tidak memanfaatkan simulator jaringan.
3	NS-3 based open-source implementation of MQTT protocol for smart building IoT applications [6]	Implementasi protokol MQTT berbasis *open-source* pada simulator jaringan NS-3 untuk aplikasi IoT *smart building*.	Menunjukkan bahwa NS-3 mampu memodelkan komunikasi MQTT secara fleksibel. Studi difokuskan pada IoT umum dan tidak secara spesifik menganalisis skenario *bottleneck bandwidth*.
4	Kajian kerentanan fasilitas darat INA-CBT terhadap guncangan gempa bumi [2]	Membahas kerentanan fasilitas darat INA-CBT terhadap guncangan gempa bumi berdasarkan standar ketahanan struktur.	Membahas aspek ketahanan fisik fasilitas darat INA-CBT. Penelitian tidak mencakup aspek kinerja jaringan komunikasi maupun protokol data seperti MQTT.

No	Judul	Fokus dan Metode	Temuan dan Keterbatasan
5	Indonesia cable-based tsunameter (INA-CBT): tsunami detection and identification on other seismic wave signals [3]	Pengembangan dan pengujian sistem Indonesia Cable-Based Tsunameter (INA-CBT) berbasis Ocean Bottom Unit (OBU) dan kabel serat optik bawah laut untuk deteksi tsunami secara *near real-time*.	Penelitian ini menunjukkan bahwa INA-CBT dapat mendeteksi sinyal seismik dan indikasi tsunami secara efektif melalui pemrosesan sinyal dengan metode penyaringan Lanczos, namun penelitian ini hanya berpusat pada analisis sinyal dan kinerja deteksi, tanpa membahas protokol komunikasi data aplikasi seperti MQTT maupun parameter QoS jaringan.

Berdasarkan Tabel 3.1, penelitian-penelitian terdahulu telah membahas sistem INA-CBT dari berbagai aspek, termasuk kinerja protokol MQTT dan implementasi sistem komunikasi. Namun demikian, masih terdapat keterbatasan dalam pengujian kinerja jaringan pada kondisi *bottleneck bandwidth* yang terkontrol serta pemodelan jaringan yang fleksibel. Oleh karena itu, penelitian ini memfokuskan pada analisis *average latency* protokol MQTT melalui pemodelan jaringan berbasis NS-3 yang terkontrol dengan tujuan memberikan kontribusi baru dalam evaluasi performa komunikasi data INA-CBT.

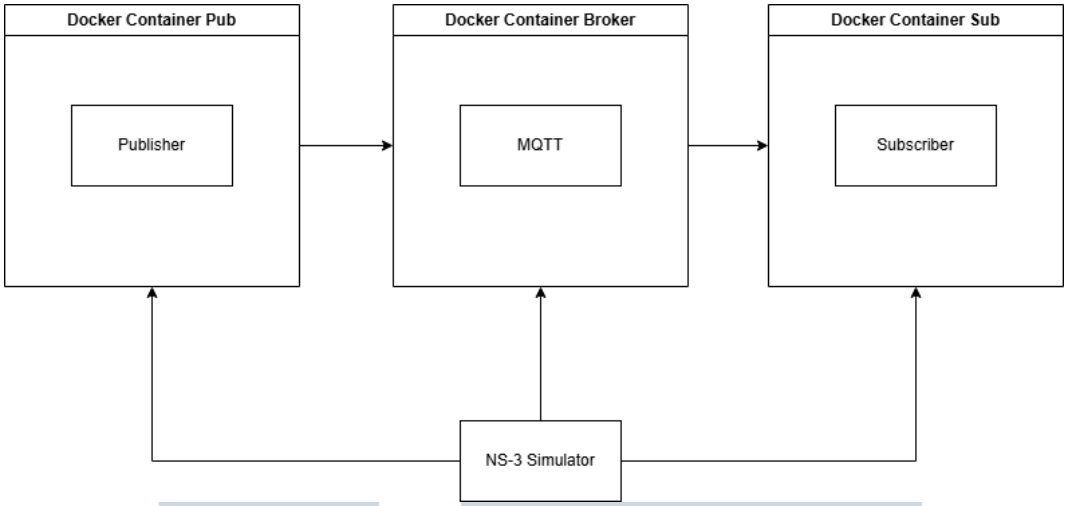
3.2 Aktivitas Pemodelan

Gambar 3.2. menunjukkan arsitektur sistem komunikasi antara OBU dan *read down station* melalui *Landing Station*. Pada sisi OBU, data dihasilkan oleh sensor BPR dan Acc, yang kemudian dikirimkan ke *landing station*. Data dari OBU terlebih dahulu diteruskan ke *switch landing station* sebelum diproses oleh *router*. Router berfungsi sebagai pengatur jalur komunikasi yang mendistribusikan data melalui beberapa media transmisi, yaitu satelit, *fiber optic*, dan *cellular*. Seluruh jalur komunikasi tersebut terhubung ke jaringan *internet* sebagai media transportasi utama, sehingga data hasil pengukuran dari OBU dapat dikirimkan secara andal menuju *read down station* untuk proses pemantauan dan analisis lebih lanjut.



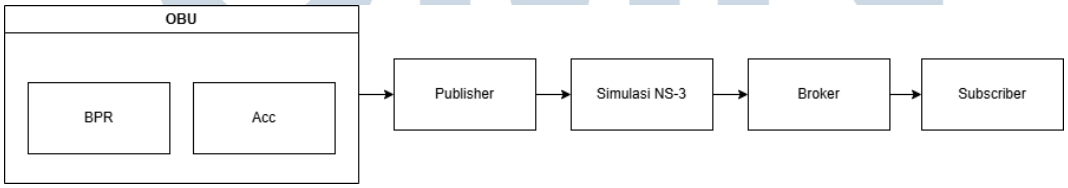
Gambar 3.2. Arsitektur INA-CBT [1]

Pada tahap pemodelan sederhana ini, alur komunikasi konseptual pada sistem INA-CBT ditunjukkan pada gambar 3.2 disederhanakan dalam bentuk diagram sistem berbasis MQTT yang diintegrasikan dengan *network simulator* menggunakan pendekatan *container*, seperti ditampilkan pada gambar 3.3. Sistem pemodelan ini terdiri atas tiga *container*, yaitu *publisher*, *broker*, dan *subscriber*, yang masing-masing dijalankan sebagai entitas terisolasi dan terpisah. *Publisher* berperan dalam menghasilkan data dummy yang merepresentasikan data sensor, kemudian mengirimkannya ke *broker* menggunakan protokol MQTT. Selanjutnya, *broker* bertindak sebagai pusat distribusi pesan yang meneruskan data tersebut kepada *subscriber*. *Subscriber* menerima data dari *broker* dan melakukan pengukuran performansi jaringan, yang meliputi analisis latensi serta pencatatan data log. Ketiga *container* tersebut dihubungkan melalui antarmuka jaringan virtual yang terintegrasi dengan NS-3, sehingga seluruh lalu lintas komunikasi MQTT tidak berjalan langsung pada jaringan fisik, melainkan dimasukkan ke dalam skenario simulasi jaringan. Dalam hal ini, NS-3 berperan mengendalikan karakteristik jaringan.



Gambar 3.3. Diagram NS-3 dan Container

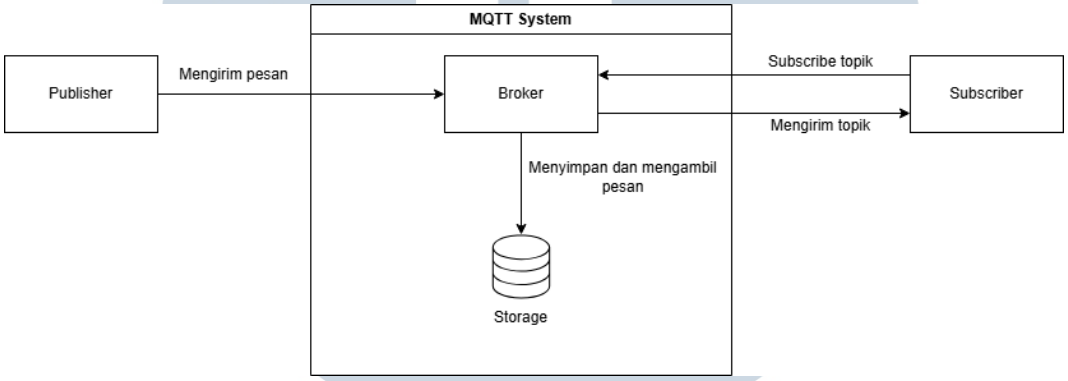
Untuk merealisasikan sistem pada Gambar 3.3, implementasi dilakukan melalui integrasi antara NS-3 dan teknologi *container*. Setiap komponen utama, yaitu *publisher*, *broker*, dan *subscriber*, diwujudkan sebagai *container* yang berjalan secara terisolasi namun saling terhubung melalui antarmuka jaringan virtual. Dalam hal ini, NS-3 berperan sebagai pengendali kondisi jaringan dengan mensimulasikan karakteristik jaringan, seperti *bottleneck bandwidth* dan latensi, sehingga seluruh lalu lintas komunikasi MQTT antar *container* dapat diarahkan dan dikontrol sepenuhnya di dalam lingkungan simulasi. Dengan pendekatan ini, *broker* tetap berfungsi sebagaimana pada sistem nyata sebagai pusat distribusi pesan, sementara *publisher* dan *subscriber* menjalankan aplikasi MQTT sesungguhnya, sehingga sistem pemodelan yang dibangun tidak hanya bersifat konseptual, tetapi juga dapat digunakan sebagai lingkungan uji yang terkontrol.



Gambar 3.4. Arsiktektur NS-3

Gambar 3.4 menunjukkan arsitektur simulasi sistem komunikasi INA-CBT yang dimodelkan menggunakan NS-3. Pada sisi sumber data, OBU menghasilkan data sensor BPR dan Acc yang merepresentasikan data hasil pengukuran di dasar laut. Data tersebut dikirimkan ke modul *publisher*, yang dalam konteks sistem nyata merepresentasikan fungsi *Landing Station* sebagai pengumpul dan pengirim

data sensor. Selanjutnya, proses transmisi data melalui jaringan dimodelkan pada lingkungan NS-3 sebagai representasi jaringan internet, yang memungkinkan pengujian karakteristik komunikasi seperti latensi dan keandalan jaringan. Data kemudian diteruskan ke broker, yang secara konseptual merepresentasikan *read down station* pada sistem INA-CBT di lapangan, sebelum akhirnya diterima oleh *subscriber* sebagai pihak yang melakukan penerimaan dan pemrosesan data. Arsitektur simulasi ini digunakan untuk merepresentasikan alur komunikasi end-to-end INA-CBT secara terkontrol dan terukur dalam lingkungan simulasi.



Gambar 3.5. Arsiktektur MQTT

Gambar 3.5 menunjukkan cara kerja alur komunikasi pada sistem MQTT. Pada sistem ini, *publisher* berfungsi menghasilkan pesan lalu mengirimkannya ke *broker*. *Broker* bertugas sebagai perantara utama untuk menerima pesan dari *publisher* ke *subscriber*, lalu mengatur pesan berdasarkan topik MQTT (BPR dan Acc), dan menyimpan pesan tersebut di *storage*. Topik pada MQTT digunakan sebagai penanda atau kategori pesan, sehingga *broker* dapat menentukan pesan mana yang harus diteruskan kepada *subscriber* yang sesuai. Selain itu, topik MQTT dalam penelitian ini merepresentasikan OBU, karena satu OBU terdapat BPR dan Acc. Lalu, *subscriber* terlebih dahulu melakukan *subscribe* pada topik tertentu, kemudian akan menerima pesan secara otomatis dari *broker* sesuai dengan topik yang dipilih, tanpa harus berkomunikasi langsung dengan *publisher*. Dengan penggunaan topik ini, proses distribusi data menjadi lebih terstruktur, efisien, dan mudah dikelola.

3.3 Penyiapan Platform Simulasi

Buka Windows PowerShell sebagai Administrator lalu ketik wsl –install untuk mengaktifkan WSL, dan mengunduh Ubuntu secara otomatis. Setelah proses selesai PowerShell akan meminta restart. Untuk menginstall Ubuntu ketik wsl –install -d Ubuntu-20.04, saat sudah terinstall ubuntu akan secara otomatis muncul. Ubuntu akan meminta username dan password saat selesai akan masuk ke prompt seperti `username@DESKTOP: $`.

1. Menginstall NS-3 ketik
    - git clone https://gitlab.com/nsnam/ns-3-dev.git.
    - cd ns-3-dev.
    - ./ns3 configure –enable-examples –enable-tests.
    - ./ns3 build.
2. Menginstall Mosquitto MQTT Broker
    - sudo apt update
    - sudo apt install mosquitto mosquitto-clients
3. Membuat Publisher dan Subscriber
    - pip install paho-mqtt
    - nano docker-compose.yml
    - mkdir pub sub
    - nano pub/Dockerfile
    - nano pub/pub.py
    - nano sub/Dockerfile
    - nano sub/sub.py
4. Membuat interface TAP di host LINUX
    - sudo ip tuntap add mode tap tap-mqtt
    - sudo ip addr add 10.0.0.10/24 dev tap-mqtt # IP Host
    - sudo ip link set tap-mqtt up
    - cd /ns-3-dev
    - ./ns3 run scratch/mqtt-ns3.cc

3.3.1 Spesifikasi Sistem

Dalam pelaksanaan penelitian ini, dipakai sejumlah perangkat lunak (*software*) dan perangkat keras (*hardware*) yang mendukung. Adapun rincian *software* yang digunakan selama proses pengembangan sistem sebagai berikut:

1. *Operating System*: Windows 11
2. *Integrated Development Environment (IDE)*: Ubuntu 20.04 (melalui WSL) dan Visual Studio Code
3. *Others*: Google Chrome

Adapun spesifikasi *hardware* dari komputer yang dimanfaatkan selama proses penelitian adalah:

1. *Processor*: Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz
2. *Memory (RAM)*: 8 GB
3. *Storage*: 429 GB Solid State Drive (SSD)

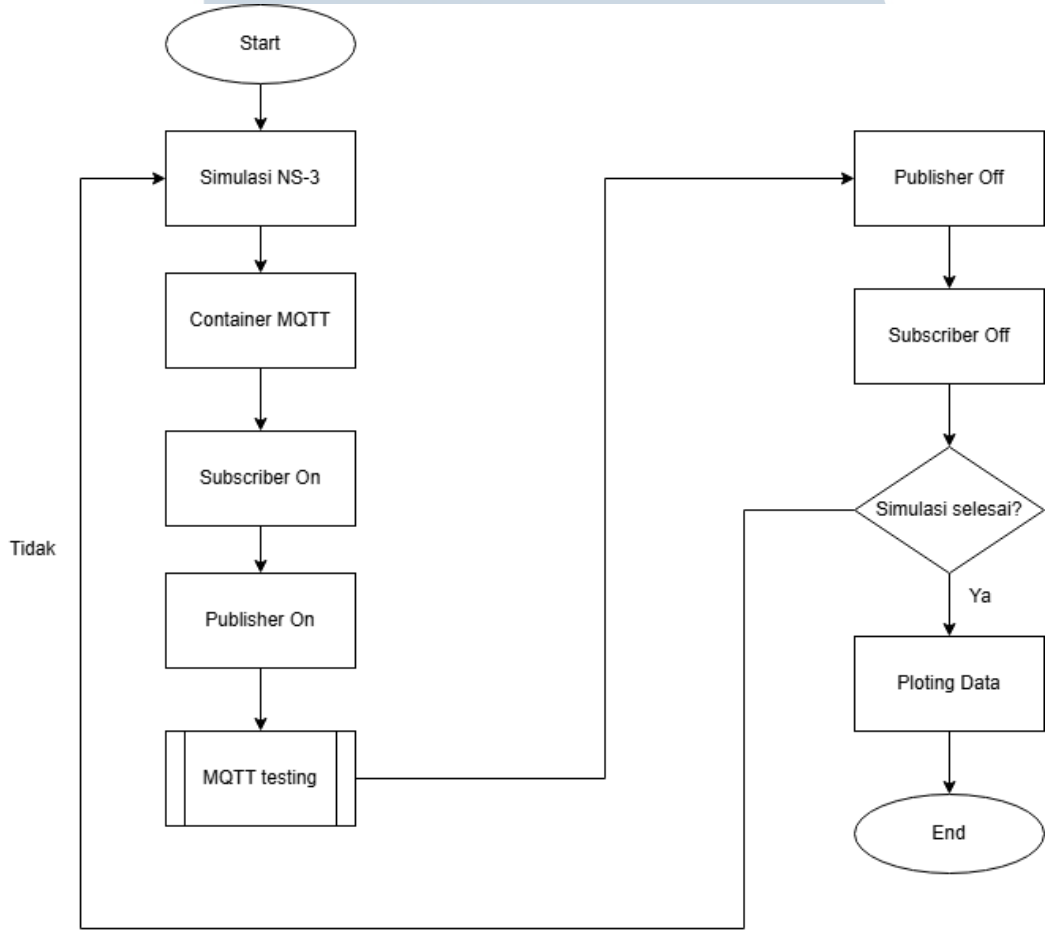
3.4 Eksperimen

Penelitian ini melakukan serangkaian eksperimen yang terstruktur untuk mengevaluasi kinerja komunikasi data menggunakan protokol MQTT pada lingkungan simulasi. Setiap tahapan eksperimen dirancang untuk mengukur efektivitas pengiriman dan penerimaan data sensor melalui berbagai konfigurasi parameter seperti QoS, jenis data, serta variasi *bottleneck bandwidth*.

3.4.1 Flowchart Simulasi

Gambar 3.6 menunjukkan *flowchart* simulasi sistem secara keseluruhan yang terintegrasi dengan NS-3. Proses simulasi diawali dengan menjalankan simulasi NS-3 untuk membentuk lingkungan jaringan sesuai dengan parameter yang telah ditentukan. Setelah itu, *container* MQTT diinisialisasi agar komunikasi antara *publisher*, *broker*, dan *subscriber* dapat berjalan di dalam lingkungan simulasi. Tahap berikutnya adalah mengaktifkan *subscriber* terlebih dahulu, kemudian *publisher* supaya data yang dikirimkan dapat langsung diterima dan dicatat. Selanjutnya, proses MQTT *testing* dilakukan dengan mengirimkan data

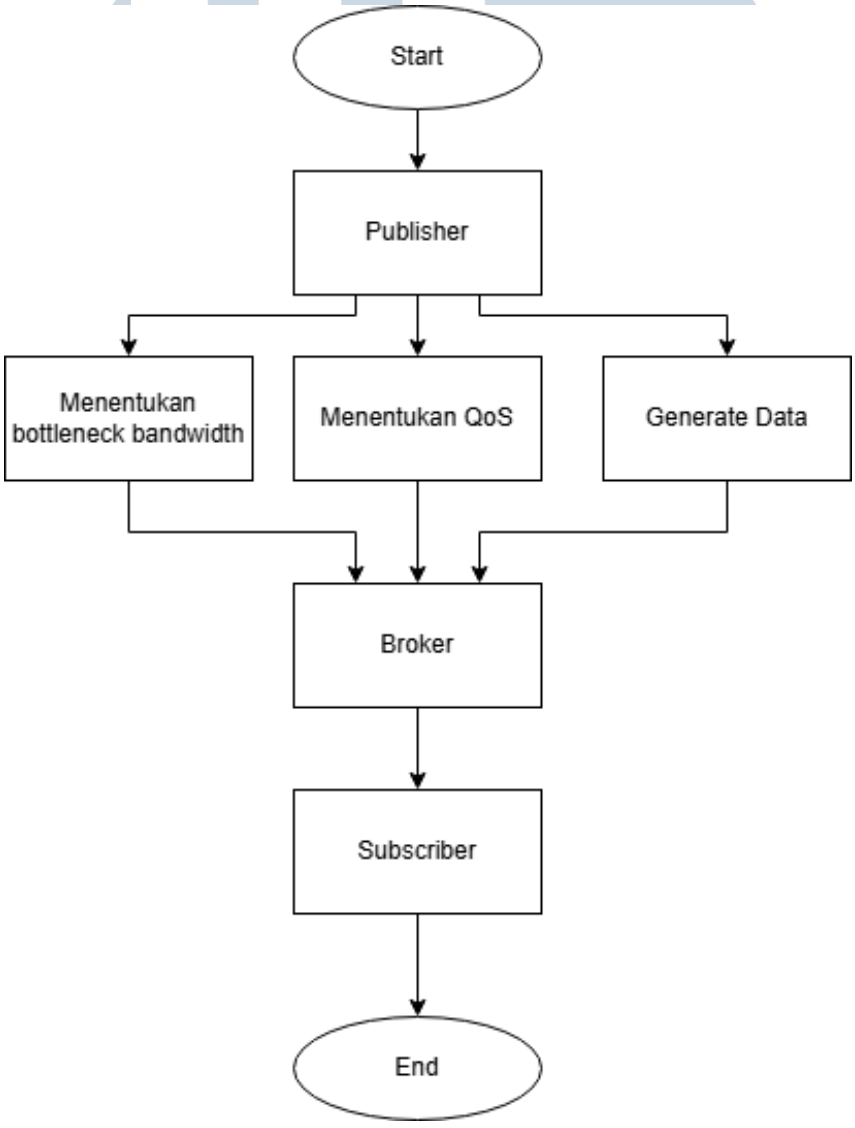
BPR dan Acc menggunakan variasi *bottleneck bandwidth* dan QoS yang telah ditentukan. Setelah proses pengiriman pesan selesai, *publisher* dan *subscriber* akan dinonaktifkan sebagai bagian dari mekanisme kontrol simulasi. Selama simulasi berlangsung, sistem akan melakukan pengecekan apakah simulasi telah selesai atau belum. Apabila simulasi belum selesai, proses akan kembali ke tahap simulasi NS-3. Jika simulasi telah selesai, data hasil pengujian diproses pada tahap plotting data untuk dianalisis lebih lanjut, kemudian simulasi diakhiri.



Gambar 3.6. Flowchart Simulasi NS-3

Berdasarkan gambar 3.7, *flowchart* modul MQTT *testing* menggambarkan alur pengujian komunikasi MQTT yang dilakukan secara terstruktur dengan variasi parameter jaringan dan pengiriman data. Proses dimulai dari *publisher* yang terlebih dahulu menetapkan nilai *bottleneck bandwidth* untuk merepresentasikan kondisi keterbatasan jaringan, yaitu sebesar 128 Kbps, 256 Kbps, dan 512 Kbps. Setelah itu, *publisher* menentukan tingkat QoS yang digunakan dalam pengiriman pesan yang masing-masing menunjukkan tingkat keandalan pengiriman data, mulai

dari pengiriman tanpa jaminan hingga pengiriman dengan jaminan tepat satu kali. Selanjutnya, *publisher* melakukan proses generate data berupa data BPR dan Acc sebagai data uji, yang kemudian dikirimkan ke *broker*. Pesan yang diterima oleh *broker* selanjutnya diteruskan kepada *subscriber* sesuai dengan mekanisme *publish–subscribe* pada protokol MQTT.

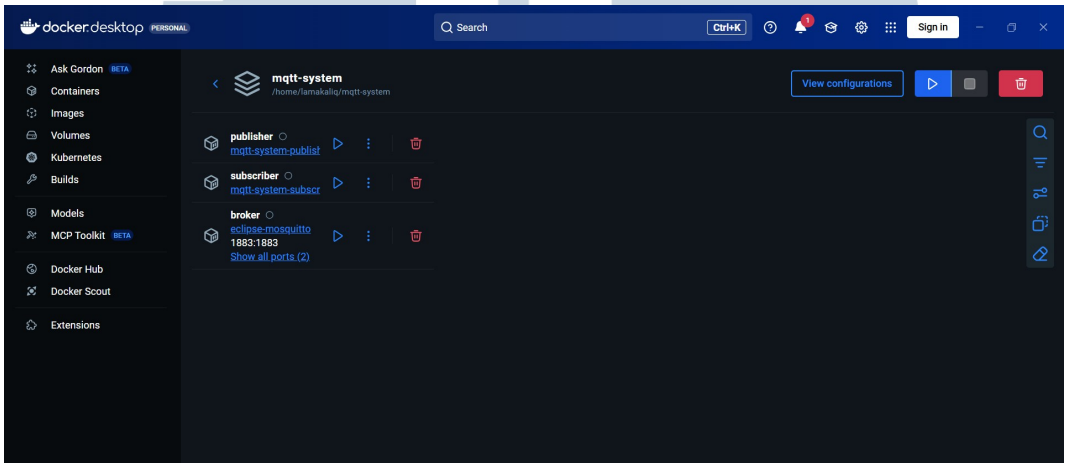


Gambar 3.7. Flowchart modul MQTT testing

3.4.2 Implementasi Simulasi Jaringan

Gambar 3.8 menunjukkan implementasi simulasi jaringan MQTT yang dijalankan dalam *container* menggunakan Docker, di mana setiap komponen sistem direpresentasikan sebagai *container* terpisah. Broker MQTT diimplementasikan

menggunakan Eclipse Mosquitto di dalam *container*, sedangkan *publisher* dan *subscriber* dijalankan pada *container* masing masing untuk melakukan pengiriman dan penerimaan data. Simulasi jaringan dalam penelitian ini dibangun menggunakan NS-3 untuk merepresentasikan kondisi jaringan, seperti *bottleneck bandwidth* yang memengaruhi proses komunikasi MQTT. Integrasi antara NS-3 dan *container* memungkinkan pemisahan antara simulasi jaringan dan aplikasi, sehingga kinerja komunikasi MQTT dapat diamati secara terstruktur dan mendekati kondisi sistem nyata.



Gambar 3.8. Implementasi simulasi jaringan MQTT

3.4.3 Pengumpulan Data

Penelitian ini memakai data dummy yang dikirim dari *publisher* dan diterima melalui protokol MQTT, di mana setiap data yang diterima oleh *subscriber* dilengkapi dengan informasi waktu pengiriman untuk menghitung nilai latensi. Seluruh data yang diterima direkam dalam berkas berformat `csv` yang memuat informasi keterlambatan pengiriman dan jumlah data yang diterima. Penggunaan data dummy dipilih karena memungkinkan pengendalian kondisi eksperimen secara konsisten, memudahkan pengaturan frekuensi dan ukuran pesan, serta mendukung replikasi pengujian tanpa ketergantungan pada data lapangan atau sensor fisik.

3.4.4 Pengambilan Sampel

Pengambilan sampel dalam penelitian ini berdasarkan kombinasi tiga parameter utama. Pertama, level QoS yang terdiri dari nilai 0, 1, dan 2. Kedua, variasi *bottleneck bandwidth* yang disimulasikan yaitu 128 Kbps, 256 Kbps, dan

512 Kbps. Ketiga, jenis data yang dikirimkan, yaitu data BPR mengirim data dengan frekuensi 1 Hz dan data Acc mengirim data dengan frekuensi 125 Hz. Masing-masing kombinasi parameter tersebut diuji selama durasi 20 menit, dan dari setiap uji tersebut dikumpulkan dalam bentuk latensi. Selain itu, variasi besaran data yang ditunjukkan ada perbedaan ukuran data *payload* yang dihasilkan dari dua jenis data yang dikirimkan. Data BPR memiliki ukuran *payload* rata-rata sebesar 80 *bytes*, sedangkan data Acc memiliki ukuran *payload* rata-rata sebesar 800 *bytes*. Untuk ukuran data yang ada di lapangan data *payload* BPR sebesar 83 *bytes* dan data *payload* Acc sebesar 7365 *bytes* [4].

```
if data_type == "BPR":
    interval = 1.0
elif data_type == "ACC":
    interval = 1.0 / 125
else:
    print(f"Data type tidak dikenali: {data_type}")
    sys.exit(1)
```

Kode 3.1. Frekuensi pengiriman data BPR dan Acc

Kode pada gambar 3.1 digunakan untuk mengatur interval pengiriman data berdasarkan jenis data yang digunakan. Apabila jenis data BPR, maka interval pengiriman data ditetapkan sebesar 1 detik, yang merepresentasikan frekuensi pengiriman sebesar 1 Hz. Sementara itu, jika jenis data Acc, interval pengiriman data ditetapkan sebesar 1 / 125 detik, yang sesuai dengan frekuensi pengiriman sebesar 125 Hz. Selain itu, jenis data yang diberikan tidak sesuai dengan kedua kategori tersebut, program akan menampilkan pesan kesalahan dan menghentikan proses eksekusi.

3.4.5 Pengujian

Tahap pengujian dilakukan dengan menjalankan file `sub.py` untuk setiap kombinasi parameter yang telah dirancang sebelumnya. Setelah selesai menerima data dari broker, file yang dihasilkan berupa .csv untuk memastikan data terekam dengan benar. Lalu, grafik-grafik yang akan membandingkan kinerja tiap tingkat QoS digambarkan menggunakan file `plot_csv.py` untuk keperluan visualisasi.

3.4.6 Visualisasi

Visualisasi dilakukan dengan memanfaatkan file `plot_csv.py`, yang membaca seluruh file `csv` yang dihasilkan untuk setiap kombinasi parameter. Skrip ini akan menggabungkan dan memetakan data ke dalam grafik performa, untuk membandingkan nilai rata-rata latensi antar berbagai tingkat QoS. Hasil visualisasi ditampilkan dalam bentuk grafik garis dan disimpan dalam format gambar `png`.