Laporan UTS IF4051
Pengembangan Sistem IoT

Oleh:

Aditya Prawira Nugroho - 13520049

Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2024

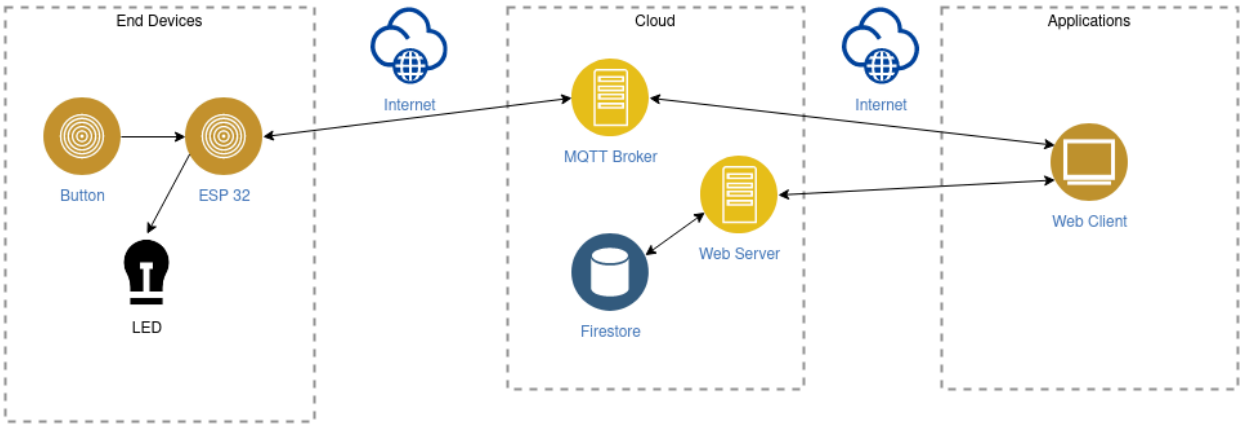
Daftar Isi

Deskripsi Sistem

Secara umum, sistem terdiri dari satu buah *node*, sebuah broker MQTT, dan satu buah *client* web. *Node* digunakan untuk melakukan pembayaran sebesar dua puluh ribu rupiah. Kemudian, *node* mem-*publish* pesan hasil transaksi, baik berhasil maupun gagal. Kemudian, *client* akan menerima pesan yang di-*publish* dan menampilkannya di layar. *Client* dapat digunakan untuk melihat hasil transaksi, histori transaksi, dan melakukan top up saldo. Secara rinci, berikut adalah fitur-fitur yang diimplementasikan:

1. Melakukan pembayaran pada *node* dengan menekan tombol
2. Menampilkan status pembayaran pada *client* web
3. Melihat histori transaksi, baik pembayaran atau *top up*
4. Melakukan *top up* sebesar Rp20.000

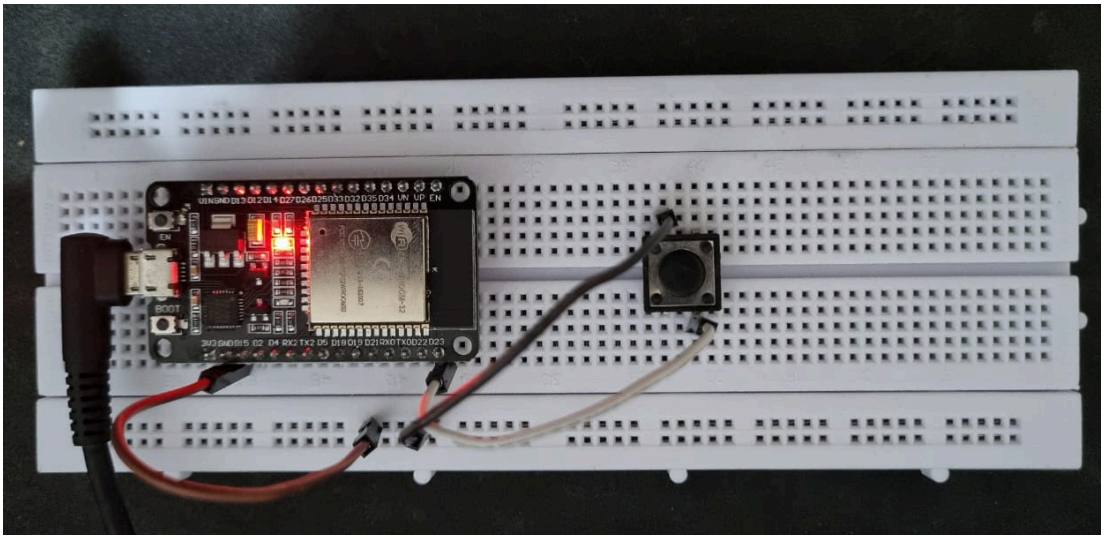
Arsitektur Hardware IoT End-to-End System



Gambar 1 Arsitektur *hardware* IoT sistem *end-to-end*

Arsitektur *hardware* IoT ditunjukkan pada Gambar 1. Terdapat tiga *layer* pada arsitektur, yaitu layer *end devices*, layer *cloud*, dan layer *applications*. Layer *end devices* berisi *node* yang terdiri dari beberapa komponen fisik seperti tombol dan LED. Kemudian, layer *cloud* terdiri dari beberapa layanan seperti layanan basis data Firestore, layanan MQTT *broker*, dan *web server* yang menjadi pengolah data dan berisi logika aplikasi. Terakhir adalah layer *applications* yang berisi *client* web untuk melihat transaksi dan melakukan *top up*. Komunikasi antar layer menggunakan internet.

Rancangan Hardware



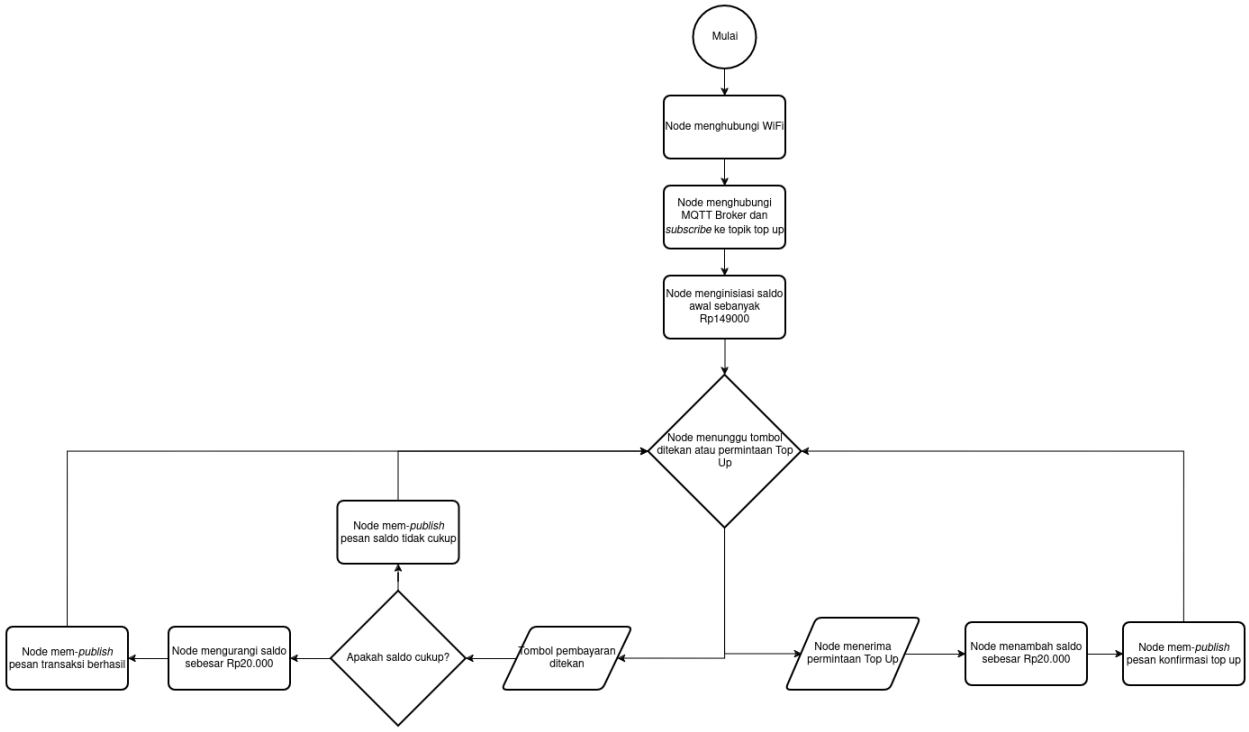
Gambar 2 Rancangan *hardware end device*

Hardware yang digunakan pada proyek ini adalah ESP32 Devkit V1, *bread board*, *push button*, dan LED bawaan ESP32. *Push button* dihubungkan dengan PIN GPIO23 dan PIN *ground* pada ESP32. PIN GPIO23 digunakan dengan mode INPUT_PULLUP, yaitu mode input yang memanfaatkan impedansi bawaan ESP32 sehingga keadaan *default button* adalah HIGH dan menjadi LOW ketika ditekan. LED bawaan yang digunakan ada pada PIN 2 pada ESP32.

Rancangan Software

Rancangan perangkat lunak pada proyek ini dibagi menjadi dua, yaitu perangkat lunak pada *end device* dan *web*.

Rancangan Firmware End Device



Gambar 3 Diagram alir rancangan *firmware end device*

Alur dari *firmware end device* dirincikan pada Gambar 3. Sebelum *node* menerima atau mengirim data, *node* melakukan *setup* terlebih dahulu dengan menghubungi WiFi dan MQTT *broker*. Ketika koneksi terjalin, *node* menginisiasi saldo dan menunggu adanya input berupa tombol yang ditekan atau pesan permintaan *top up*.

Jika *node* menerima pesan permintaan *top up*, saldo akan ditambah Rp20.000. Kemudian, *node* akan mengirimkan pesan konfirmasi *top up*. Jika tombol pembayaran ditekan, *node* akan mengecek apakah saldo yang tersisa cukup untuk melakukan pembayaran. Jika saldo cukup, saldo dikurangi Rp20.000 dan pesan transaksi berhasil dikirim. Jika saldo tidak cukup, *node* mengirimkan pesan bahwa transaksi gagal karena saldo tidak cukup. Selain itu, *node* akan menyalakan LED selama lima detik apabila transaksi berhasil dan LED akan berkedip selama lima detik apabila transaksi gagal.

Rancangan Software Web

Berikut adalah *list* alat yang digunakan untuk mengimplementasikan web

1. NextJS 14
2. Docker untuk menjalankan *mosquitto*
3. MQTTjs

4. Firestore

Pada web, terdapat tiga halaman yang dapat diakses pengguna:

1. Homepage
   Halaman ini akan menampilkan pesan transaksi yang berhasil atau gagal. Selain itu, halaman ini akan mengirimkan data transaksi yang berhasil ke Firestore.
2. History
   Halaman ini menampilkan histori transaksi dari Firestore. Transaksi yang ditampilkan adalah pembayaran dan *top up*.
3. Top Up
   Halaman ini digunakan untuk melakukan *top up* saldo. Halaman ini akan menampilkan pesan yang dikirim oleh *node*. Selain itu, data *top up* yang berhasil akan dikirimkan ke Firestore sebagai histori transaksi.

Hasil Pengujian dan Analisis

1. Pengujian Pembayaran

Pengujian pembayaran dilakukan dengan menekan tombol dan mengurangi saldo hingga tidak mencukupi. LED dapat menyala sesuai dengan ketentuan, yaitu menyala lima detik ketika transaksi berhasil dan berkedip lima detik ketika transaksi gagal. Selain itu, aplikasi *web client* mampu menampilkan pesan yang dikirimkan oleh *node* tanpa masalah. Aplikasi *web client* juga dapat mengirimkan histori transaksi yang sesuai. Akan tetapi, terdapat kasus yang belum ditangani, yaitu ketika aplikasi *web client* tidak aktif. Jika aplikasi *web client* tidak aktif, maka histori transaksi tidak akan diperbarui.

2. Pengujian Histori Transaksi

Histori transaksi dapat ditampilkan dengan benar. Waktu dan tanggal dari transaksi dapat ditampilkan dengan benar serta tipe dari transaksi dan nominalnya. Akan tetapi, perlu diperhatikan bahwa waktu yang digunakan pada histori transaksi bergantung pada aplikasi *web client* sehingga ada kemungkinan tidak akurat.

3. Pengujian Top Up

Top up berhasil dilakukan. Alur dari *top up* adalah aplikasi *web client* mengirimkan pesan untuk *top up*. Kemudian, *node* mengecek pesan dan menambah saldo sebesar Rp20.000 dan mengirimkan pesan bahwa *top up* berhasil dilakukan. Kemudian, aplikasi *web client* akan menampilkan pesan tersebut.

Simpulan dan Refleksi

1. Fitur utama dan fitur tambahan dapat berjalan dengan baik.
2. Pada *javascript*, fungsi yang mengubah waktu menjadi bulan dimulai dari indeks 0 sehingga harus ditambah 1 untuk menampilkan bulan yang benar.
3. Arsitektur IoT dapat diperbaiki sehingga *node* mengirimkan pesan transaksi berhasil ke sebuah *microservice* yang berfungsi untuk memperbarui histori transaksi. Dengan begitu, histori transaksi akan selalu diperbarui meskipun aplikasi *web client* tidak aktif.
4. Jangan melakukan *subscribe* dan *publish* di satu topik yang sama di satu waktu yang bersamaan, hal ini dapat menyebabkan *infinite callback loop* pada *node*.
5. Ketika menggunakan *button*, manfaatkan fitur INPUT_PULLUP atau INPUT_PULLDOWN pada esp32 sehingga tidak terjadi *behavior* yang tidak diinginkan seperti mendeteksi *button* yang ditekan ketika tidak ditekan. Selain itu, implementasikan *debounce* untuk mengurangi pembacaan yang berlebih.

Referensi

[1] MQTT on ESP32: A Beginner's Guide
[2] MQTT on Javascript

Lampiran

Link Github: https://github.com/Adityapnn811/uts-iot
Link Video: https://youtu.be/PzeDZok6i0E?si=ySUpkNsDHXApuNMe