© Hak Cipta milik Politeknik Negeri Jakarta
Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta
tanpa izin Politeknik Negeri Jakarta

BAB II
TINJAUAN PUSTAKA

2.1 Internet Of Things (IoT)

Internet Of Things, yang biasa disingkat IoT adalah sebuah konsep yang bertujuan untuk memperluas manfaat dari koneksi internet yang selalu aktif, menghubungkan mesin, perangkat, dan objek fisik lainnya ke sensor dan aktuator untuk memperoleh data dan mengelola kinerjanya sendiri. Memungkinkan mesin untuk bekerja sama dan beroperasi secara independen berdasarkan informasi yang baru diperoleh (Efendi, 2018).

Internet of Things merupakan suatu konsep dimana suatu objek dapat mempunyai kemampuan dalam hal komunikasi via jaringan, seperti proses pentransferan data tanpa adanya proses komunikasi yang dilakukan antar manusia (manusia ke manusia) maupun antar manusia ke perangkat sistem seperti komputer atau sebuah kontroler. Dengan adanya teknologi *Internet of Things* ini proses kerja sebuah sistem dapat dilakukan semakin luas, jarak jangkauannya juga semakin luas, proses pengolahan data dan analisis data terhadap sebuah sistem juga semakin bagus. Teknologi IoT ini mendukung kerja sistem sebagai suatu kesatuan meliputi komponen/elemen dalam hal memudahkan proses aliran informasi data. Sistem pada penelitian ini mengabungkan tiga bagian penting, yaitu mekanik, *hardware* (elektronik) dan algoritma kontrol, dimana ketiga bagian tersebut saling berinteraksi dan tidak dapat dipisahkan dalam satu kesatuan sistem (Abdullah, 2021).

2.2 Pemograman

Pemrograman adalah proses menulis, menguji, memperbaiki dan memelihara kode yang membangun sebuah program komputer. Kode ini ditulis dengan menggunakan bahasa pemrograman tertentu. Terdapat banyak jenis bahasa pemrograman seperti C#, C++, javascript, PHP, dan lain-lain. Tujuan dari pemrograman adalah untuk membuat suatu program yang dapat melakukan suatu perhitungan atau 'pekerjaan' sesuai dengan keinginan si pemrogram. Untuk melakukan pemrograman, diperlukan keterampilan dalam algoritme, logika, bahasa pemrograman, dan pada banyak kasus, pengetahuan-pengetahuan lain sepert

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

matematika.

Pemrograman adalah suatu seni dalam menggunakan satu atau lebih algoritme yang saling berhubungan dengan menggunakan suatu bahasa pemrograman tertentu sehingga menjadi suatu program komputer. Bahasa pemrograman yang berbeda mendukung gaya pemrograman yang berbeda pula. Gaya pemrograman ini biasa disebut paradigma pemrograman. Terdapat banyak teknik dan metode dalam mempelajari dan mengajarkan pemrograman. Salah satunya yaitu yang paling umum dengan cara tatap muka langsung didalam kelas. Dimana dalam satu kelas biasannya diisi banyak siswa dengan 1 pengajar. Pengajar menjelaskan materi dengan memberikan tutorial menyelesaikan masalah. Setelah itu pengajar akan memberikan latihan kepada siswa sesuai dengan materi yang telah diajarkan.

Selain itu, teknik yang biasa digunakan oleh para pengajar, yaitu dengan memberikan project membuat program yang dikerjakan secara berkelompok. Dengan begini pengajar akan mengetahui feedback berupa tingkat pemahaman siswa apakah sudah melampaui target atau belum. Dengan berkembangnya teknologi yang sangat pesat, maka teknik pembelajaran dan pengajaran pun banyak berubah. Teknik yang lama dinilai terlalu biasa dan cenderung membosankan. Baik untuk siswa maupun pengajar.

Oleh karena itu, didalam jurnal ini akan membahas teknik-teknik dan alat yang digunakan dalam mempelajari dan mengajarkan pemrograman berdasarkan penelitian-penelitian yang telah dilakukan. Nantinya, teknik-teknik ini akan dibandingkan untuk mengetahui teknik yang paling efektif dan efisien. Di dalam jurnal ini akan dijelaskan bahwa teknik dan metode pembelajaran dan pengajaran koding pada universitas dan industri yang paing efektif dan efisien adalah menggunakan game interaktif berbentuk open source bernama pex4fun.

Teknik ini dinilai efektif dan efisien karena memiliki beragam metode yang menarik dan tidak monoton. Selain itu, teknik penilaian yang digunakan berbeda dengan penilaian pada umumnya (M.F. Naufal,2018).Gambar 2.1 menunjukkan logo bahasa pemrograman C++ yang termasuk dalam bahasa pemrograman dan terdapat pada Arduino IDE.

Gambar 2. 1 *Logo C++*

Sumber : Ramadhana, 2018

2.3 Android Studio

Android Studio adalah lingkungan pengembangan terpadu (IDE) yang dirancang khusus untuk membangun aplikasi yang berjalan di platform Android. IDE ini berbasis pada IntelliJ IDEA, sebuah IDE yang populer untuk bahasa pemrograman Java. Bahasa utama yang digunakan untuk pengembangan aplikasi adalah Java, sementara untuk mengatur tampilan atau layout, digunakan bahasa XML. Android Studio juga terintegrasi dengan Android *Software Development Kit* (SDK) untuk mengimplementasikan aplikasi ke perangkat Android.

Android Studio merupakan evolusi dari *Eclipse*, yang telah dikembangkan menjadi lebih kompleks dan profesional. IDE ini menyertakan Android SDK tools yang penting untuk pengembangan aplikasi Android. Setiap proyek di Android Studio terdiri dari satu atau lebih modul yang mencakup file kode sumber dan sumber daya. Jenis-jenis modul termasuk modul aplikasi Android, modul pustaka, dan modul *Google App Engine*.

Android Studio memiliki banyak fitur yang dapat digunakan untuk mengembangkan aplikasi berbasis Android. Beberapa fitur yang dapat digunakan adalah:

a. Memiliki open source build system yang fleksibel.

b. Emulator yang mudah, cepat, dan kaya akan fitur.

c. Dapat digunakan untuk mengembangkan semua jenis perangkat Android.

d. Memiliki fitur Instan Run yaitu adalah fitur menjalankan

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

aplikasi tanpa membuat APK baru.

e. Mengimpor kode dengan mudah karena memiliki template kode dan terintegrasi dengan GitHub.
f. Alat pengujian dan frameworks yang ekstensif.
g. Lint tools untuk menganalisis kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain saat pembuatan aplikasi.

Di dalam Android Studio, setiap projeknya memiliki satu atau beberapa jenis modul seperti modul aplikasi, modul library, dan modul *Google Cloud* yang berisi *source code files* dan *resource files*.

Android Studio memiliki beberapa kelebihan. Proses iterasi dalam pengkodean dilakukan dengan cepat karena Android Studio didasarkan pada IntelliJ IDEA, yang memfasilitasi penggunaan kode dan alur kerja yang efisien. Konfigurasi pembangunan aplikasi tidak terbatas berkat struktur proyek dan sistem pembangunan berbasis Gradle, yang memberikan fleksibilitas dalam menghasilkan APK untuk berbagai jenis perangkat. Fitur-fitur yang ada di Android Studio juga mendukung pengembang untuk mengkode dengan percaya diri, menciptakan aplikasi yang kaya fitur dan terhubung dengan baik. Selain itu, Android Studio menyediakan alat GUI yang mempermudah perancangan tampilan aplikasi, mengurangi tugas-tugas yang memakan waktu.

Android Studio menggunakan Gradle sebagai sistem dasar untuk membangun proyek perangkat lunak Android. Gradle adalah sistem manajemen build yang memungkinkan pengembang untuk mengelola proses kompilasi, pengujian, dan penggabungan kode sumber menjadi file Android Package (APK) yang siap digunakan. Plugin Android Gradle menyediakan kemampuan khusus untuk pengembangan aplikasi Android di dalam Android Studio.

Gradle memungkinkan pengaturan detail tentang bagaimana setiap bagian dari proyek harus dikompilasi dan diintegrasikan, serta mengelola dependensi dan versi dari pustaka yang digunakan dalam proyek Android. Library dalam konteks Gradle merujuk pada kumpulan kode yang dikompilasi menjadi file Java Archive (JAR) atau Android Archive (AAR), yang dapat diintegrasikan dan digunakan oleh pengembang dalam pengembangan aplikasi mereka. Dependensi terhadap library

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

ditentukan dalam file build.gradle proyek, di mana Gradle akan mengelola proses pengunduhan dan integrasi library tersebut ke dalam proyek aplikasi yang sedang dikembangkan. (Ida, Bagus, 2018).Contoh logo android studio dapat dilihat pada Gambar 2.2

Gambar 2. 2 Android Studio

Sumber : Prayoga, 2017

2.4 Kotlin

Kotlin adalah bahasa pemrograman dengan tipe statis yang menggabungkan prinsip-prinsip object-oriented dengan fitur-fitur fungsional, dirancang untuk berjalan di atas *Java Virtual Machine* (JVM). Bahasa ini pertama kali dikembangkan oleh JetBrains sejak tahun 2011 dan mulai didukung secara resmi oleh Google untuk pengembangan aplikasi Android sejak Mei 2017, diumumkan pada acara Google I/O 2017 . Sejak saat itu, popularitas Kotlin mengalami peningkatan yang signifikan. Kotlin dapat digunakan untuk pengembangan aplikasi Android, pengembangan server-side, dan pengembangan client-side.

Salah satu fitur unggulan Kotlin adalah interoperabilitas penuh dengan Java, memungkinkan Kotlin dan Java dapat digunakan bersama dalam satu proyek aplikasi. Kotlin juga dirancang dengan konsep null-safety, yang mengatasi masalah umum seperti NullPointerException (NPE) dalam bahasa pemrograman Java. Dalam Kotlin, masalah NPE dapat dideteksi pada saat kompilasi, berbeda dengan Java yang menangani NPE pada saat runtime. Selain itu, Kotlin mendukung paradigma pemrograman fungsional dengan fitur-fitur seperti ekspresi lambda, fungsi higher-order, evaluasi malas (lazy evaluation), dan berbagai metode pada koleksi data seperti filtering, pemetaan (mapping), pengurutan (ordering), dan lain-lain. (Susanto. S, 2020)

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.4.1 Tipe Data

Dalam Kotlin semuanya adalah sebuah objek dalam artian bahwa *member functio*n dan *property* dapat dipanggil pada variable apapun. Setiap nilai atau *value* dalam Kotlin pasti memiliki sebuah tipe data. Dan dengan kemampuan *type inference* Kotlin dapat membaca tipe data dari sebuah nilai atau *value* secara otomatis. Tipe data dasar dalam Kotlin antara lain numbers, character, Booleans, array dan string.

a. *Numbers*

Tipe data *numbers* dalam Kotlin adalah kategori yang digunakan untuk menyimpan nilai numerik. Tipe data ini mencakup bilangan bulat dan bilangan pecahan, yang masing-masing memiliki varian dan batasan tertentu. Berikut adalah penjelasan lebih rinci mengenai tipe data numbers dalam Kotlin:

```
fun main() {
val integer = 5 // tipe integer
val   long   =   5000000000   //tipe   longkarena melebihi
       nilai maksimal integer
val  long1  =  5L  //  long,  dengan  menambahkan  L diakhir
       value
val byte: Byte = 75L // menetapkan value dengan tipe byte
}
```

Selain penulisan dan penentuan tipe data numbers, ada hal lain yang perlu diperhatikan pada baris kode di atas, yaitu variable. Umumnya variable digunakan untuk menyimpan informasi atau nilai yang akan dikelola di dalam sebuah program.

1. **var atau val** : var atau val adalah kata kunci untuk mengontrol nilai dari sebuah variable. Dengan kata kunci var sebuah nilai dari variable yang sudah diinisialisasikan dapat diubah.
2. **integer, long, long1, dan byte** : Identifier yang merupakan nama dari sebuah variable
3. **Byte atau akhiran L pada variable long1** : Bagian dari variable yang menetapkan tipe datanya secara eksplisit. Tipe data dibutuhkan agar compiler dapat mengetahui bagaimana sebuah data akan digunakan. Karena Kotlin mendukung type inference, maka variable juga diperbolehkan untuk tidak menetapkan tipe datanya secara eksplisit.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4. **5000000000 dan 75L** : Nilai awal dari sebuah variable disebut juga dengan *initialization*.

b. *Character*

Tipe data **character** dalam Kotlin, yang dideklarasikan dengan kata kunci **Char**, digunakan untuk merepresentasikan karakter tunggal. Karakter ini dapat berupa huruf, angka, atau simbol khusus. Dalam Kotlin, karakter dinyatakan dengan menggunakan tanda kutip tunggal (' ').

```
fun main() { val letter: Char = 'A' // Mendeklarasikan
karakter 'A'
val digit: Char = '5' // Mendeklarasikan karakter '5'
val specialChar: Char = '@' // Mendeklarasikan karakter '@'
println(letter) // Output: A
println(digit) // Output: 5
println(specialChar) // Output: @ }
```

c. *Booleans*

Tipe data *Boolean* adalah tipe yang mewakili memiliki dua nilai: *True or False*. *Boolean* memiliki operasi bawaan antara lain :

1. || atau lazy conjunction atau AND

   Operator ini akan mengembalikan nilai true jika semua hasil evaluasi expression yang diberikan bernilai true.

```
fun main() {
val berangkat = 7 val pulang = 5 val sekarang = 12
//operator AND atau Conjunction
val posisi = sekarang >= berangkat && sekarang <=
pulang
print("sekarang si Edo sudah $posisi sekolah")
}
```

2. && atau lazy disjunction atau OR

   Operator ini akan mengembalikan nilai true jika hasil evaluasi dari salah satu expressions yang diberikan bernilai true.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

```
//operator OR atau Disjunction
val posisi = sekarang >= berangkat || sekarang <=
pulang
print ("Sekarang si Edo sudah $posisi sekolah")
```

3. ! atau negation atau NOT

Operator NOT(!) digunakan untuk melakukan negasi (mengubah nilai true/false menjadi kebalikannya) pada hasil evaluasi expression yang diberikan.

```
fun main() {
// operator NOT atau Negation
If(!berangkat) { Print("Edo sudah pulang")
}else {
Print("Edo sudah berangkat sekolah")
}
}
```

d. *Array*

Array dalam Kotlin diwakili oleh class Array, yang memiliki fungsi *get* dan set serta *property size*. Array adalah tipe data yang dapat menyimpan beberapa objek sekaligus di dalam sebuah variable. Untuk membuat sebuah array dalam program gunakan fungsi library *arrayOf()* dan berikan nilai yang akan dimasukan ke dalam fungsi seperti berikut.

```
fun main() {
// Arrays
val array = array0f(1, 2, 3)
val arrayMix = array0f(5,6,7, "ody", false) }
```

e. *String*

Tipe data yang satu ini sebenarnya telah banyak digunakan pada

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

program- program sebelumnya yaitu nilai yang diapit dengan tanda “ ” kutip ganda atau ditunjukan oleh variable yang menetapkan tipenya dengan diwakili oleh tipe String. Mirip dengan array tipe data String juga memiliki index untuk setiap elemennya dan dimulai dari index 0.

```
fun main() {
val name = “ody” for(a in name) {
print(a)
}
}
```

Pada program di atas, jika index elemen dari string basmallah dipecah maka index E=0, d=[1], o=[2]. Jika dalam suatu kondisi program butuh untuk
menndapatkan karakter tunggal dari sebuah String manfaatkanlah indexing tersebut.

```
fun main() {
val name = “Ody”
for karakterAwal = name [0]
printIn(karakterAkhir)
valkarakterAkhir=name[3]println(karakterAkh)
//mengambil karakter    terakhir menggunakan fungsi
val charAkhir = name.last()}
```

Nilai *String* juga dapat digabungkan dengan nilai dari tipe data lain dengan menggunakan operator + plus tapi dengan satu syarat yaitu elemen pertama dari expression harus bertipekan *String*.

```
fun main() {
val name = “Kautsar” + 12 println(name + ”tahun”
}

}
```

2.4.2 Struktur Data

Struktur program kotlin adalah kerangka dasar yang digunakan untuk mengorganisir dan mengatur kode dalam bahasa pemrograman kotlin. Struktur ini membantu dalam mengontrol alur eksekusi program dan membuat kode lebih terstruktur dan mudah dipahami.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

a. *Classes* (Kelas)

Kelas dalam Kotlin adalah struktur dasar untuk mengorganisir kode yang menggabungkan data (dalam bentuk properti) dan perilaku (melalui metode atau fungsi). Kelas dalam Kotlin mendefinisikan objek dengan atribut-atribut yang mewakili keadaan atau informasi tertentu, serta operasi yang dapat dilakukan terhadap objek tersebut. Kelas dapat memiliki konstruktor untuk menginisialisasi objek saat dibuat, dan dapat mewarisi properti serta metode dari kelas lain untuk memfasilitasi penggunaan kembali kode.

b. *Data Classes* (Kelas Data)

Data classes adalah kelas khusus dalam Kotlin yang disediakan untuk menyimpan data. *Data classes* dirancang untuk menyederhanakan definisi dan penggunaan objek yang hanya digunakan untuk menyimpan nilai dan tidak memiliki perilaku tambahan. Kotlin secara otomatis menyediakan implementasi fungsi standar seperti *toString()*, *equals()*, *hashCode()*, dan *copy()* berdasarkan properti yang dideklarasikan dalam konstruktor primer. *Data classes* sangat berguna dalam pengkodean yang berorientasi pada data seperti penggunaan model atau transfer objek data (DTO).

c. (Enumerasi)

Enumerasi dalam Kotlin digunakan untuk mendefinisikan tipe data dengan kumpulan nilai-nilai yang terbatas dan spesifik. Setiap nilai dalam enum memiliki nama dan nilai yang dapat diakses menggunakan sintaks yang jelas. Enumerasi berguna untuk membatasi pilihan nilai yang valid untuk sebuah variabel atau parameter dan mengelompokkan konstanta-konstanta terkait yang memiliki hubungan semantik yang kuat.

d. (Antarmuka)

Antarmuka adalah kontrak yang mendefinisikan perilaku atau kemampuan yang harus diimplementasikan oleh kelas lain. Antarmuka menyediakan cara untuk mendeklarasikan fungsi-fungsi tanpa memberikan implementasi konkret, yang kemudian diimplementasikan oleh kelas-kelas yang menerapkannya. Antarmuka memisahkan spesifikasi (apa yang harus dilakukan) dari implementasi (bagaimana harus dilakukan), sehingga

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

mengadopsi pola desain seperti *Dependency Injection* dan *polimorfisme* tanpa perlu ketergantungan pada implementasi kelas konkret.

e. *Object Expressions* (Ekspresi Objek)

Ekspresi objek dalam Kotlin membuat objek anonim yang dapat langsung digunakan tanpa perlu mendefinisikan kelas secara eksplisit. Eksperesi objekcocok untuk situasi di mana pengguna hanya membutuhkan objek sementara dengan kemampuan tertentu tanpa perlu membuat kelas baru. Ekspresi objek biasanya digunakan dalam konteks seperti implementasi antarmuka tanpa membuat kelas baru atau dalam pengujian unit untuk membuat objek tiruan.

f. *Type Aliases* (Alias Tipe)

Alias tipe adalah mekanisme yang memberikan nama lain pada tipe data yang sudah ada atau kombinasi dari tipe-tipe yang ada. Alias tipe berguna untuk menyederhanakan penggunaan tipe-tipe yang kompleks atau panjang, sehingga membuat kode lebih mudah dibaca dan dipahami. Alias tipe tidak menciptakan tipe data baru, tetapi hanya memberikan nama yang dapat digunakan sebagai sinonim untuk tipe yang sudah ada.

g. *Inline Classes* (Kelas Inline)

Kelas inline adalah fitur eksperimental dalam Kotlin yang menciptakan kelas yang dijalankan secara inline. Kelas inline dirancang untuk mengurangi overhead memori yang terkait dengan penciptaan objek, terutama ketika objek tersebut hanya digunakan sebagai wadah untuk tipe data tunggal.

2.5 Firebase

Firebase adalah penyedia layanan *cloud* dengan *back-end* sebagai servis yang berbasis di San Fransisco, California. *Firebase* membuat sejumlah produk untuk pengembangan aplikasi *mobile* ataupun *website*. *Firebase* didirikan oleh Andrew Lee dan James Tamplin pada tahun 2011 dan diluncurkan dengan *cloud database* secara *real-time* di tahun 2012. Produk utama dari *Firebase* yakni suatu *database* yang menyediakan API untuk memungkinkan pengembang menyimpan dan mensinkronisasi data lewat *multiple client*. Perusahaan ini diakusisi oleh Google pada Oktober 2014.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Firebase adalah penyedia layanan *real-time database* dan *backend* sebagai layanan. Suatu aplikasi yang memungkinkan pengembang membuat API untuk disinkronisasikan untuk *client* yang berbeda-beda dan disimpan pada *cloud*-nya *Firebase*. *Firebase* memiliki banyak *library* yang memungkinkan untuk mengintegrasikan layanan ini dengan Android, Ios, Javacript, Java, Objective-C dan Node.JS. *Database Firebase* juga bersifat bisa diakses lewat REST API. REST API tersebut menggunakan protokol *Server-Sent Event* dengan membuat koneksi HTTP untuk menerima *push notification* dari server.

Pengembang menggunakan REST API untuk post data yang selanjutnya *Firebase client library* yang sudah diterapkan pada aplikasi yang dibangun yang akan mengambil data secara *real-time*. Pengembang juga dapat menggunakan *database* ini untuk mengamankan data menggunakan *server Firebase* dengan *rules* yang ada. Untuk *hosting file firebase* menyediakan *hosting* untuk *static file* dengan fasilitas CND dan SNL. (Yulia, 2019)

2.6 Quality Of Service

Quality of service (QOS) adalah metode analisa jaringan yang digunakan untuk melakukan manajemen bandwidth. Manajemen bandwidth perlu dilakukan agar koneksi internet di satu titik akses tidak digunakan secara penuh oleh satu user maupun satu kelompok user saja. QOS diterapkan dalam jaringan komputer agar dapat memberikan layanan yang optimal dan efisien bagi para pengguna jaringan komputer. Seorang administrator jaringan dapat memprioritaskan sub jaringan tertentu dalam sebuah layanan dengan menggunakan QOS. (Muhamad Hasbi, 2021)

1. Latency (Waktu Pengiriman): Waktu yang dibutuhkan untuk mengirimkan paket dari pengirim ke penerima. Latency yang rendah sangat penting untuk aplikasi real-time..

2. Jitter: Variasi dalam waktu pengiriman paket. Jitter yang tinggi dapat menyebabkan masalah dalam aplikasi seperti VoIP atau video streaming. Meskipun tidak ditampilkan secara langsung pada gambar, pengukuran jitter dapat dilakukan berdasarkan perubahan latency.

3. Throughput (Kecepatan Jaringan): Ukuran seberapa banyak data yang dapat dikirimkan dalam satuan waktu tertentu.

4. Packet Loss (Kehilangan Paket): Jumlah paket yang hilang selama pengiriman. Dalam kasus IoT, kehilangan paket dapat menyebabkan kegagalan sistem atau hasil pengukuran yang tidak akurat

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB III

PERENCANAAN DAN REALISASI

Pada bab ini akan dijelaskan mengenai perencanaan dan realisasi alat yang dibuat dalam tugas akhir. Rancangan alat yang dibahas yaitu rancang bangun pemurninan air laut menjadi air bersih berbasis android.

3.1 Rancang Alat

Pada tugas akhir ini dibuat "Perancangan Aplikasi Android Pemurnian Air Laut Menjadi Air Bersih Berbasis Android" yang dapat digunakan untuk masyarakat pesisir yang kekurangan air bersih . Sistem alat ini terkoneksi secara *real-time* pada *firebase* dan di integrasikan dengan aplikasi Android *Water Filter*. Bagian yang akan dibahas adalah perencanaan pada program untuk bagian sistem transmisi dan pemrograman aplikasi Android dengan sensor yang digunakan adalah sensor PH, Sensor Ultrasonik, Sensor Salinitas dan Sensor Turdibity. Pada bagian pengiriman berfungsi untuk mengirim data sensor menggunakan ESP32 dan hasilnya akan ditampilkan melalui aplikasi Android.

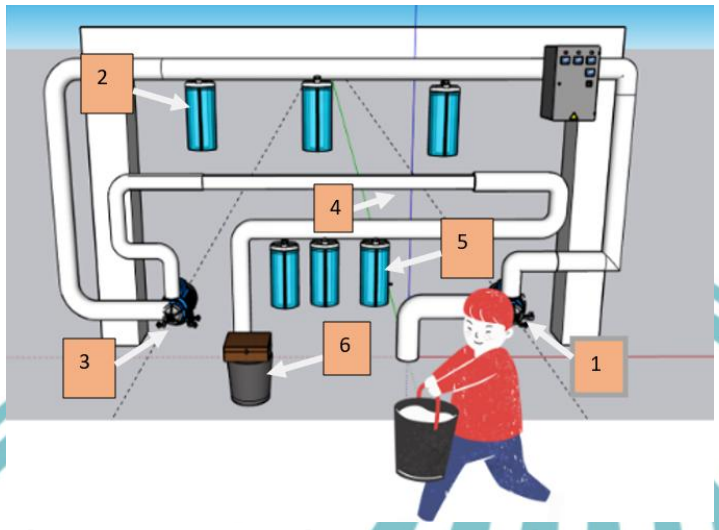
3.1.1 Deskripsi Alat

Alat yang dibuat ini digunakan untuk mengubah air laut menjadi air yang dapat digunakan sehari hari khusus nya pada masyarakat pesisir yang kekurangan air bersih. Perancangan dan realisasi sistem yang dibuat pada tugas akhir ini berupa alat pemurninan air laut menjadi air bersih berbasis android.. Alat tersebut nantinya akan digunakan untuk menfiltrasi air laut menjadi air yang dapat digunakan untuk kebutuhan sehari hariData pembacaan dari sensor akan dikirimkan ke *database* untuk menyimpan data dari sensor kemudian ditampilkan pada aplikasi Android. Aplikasi ini memanfaatkan platform Kodular sebagai perancangan aplikasi dan *Firebase* sebagai *database* untuk menciptakan sistem yang efisien dan terhubung dengan internet.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

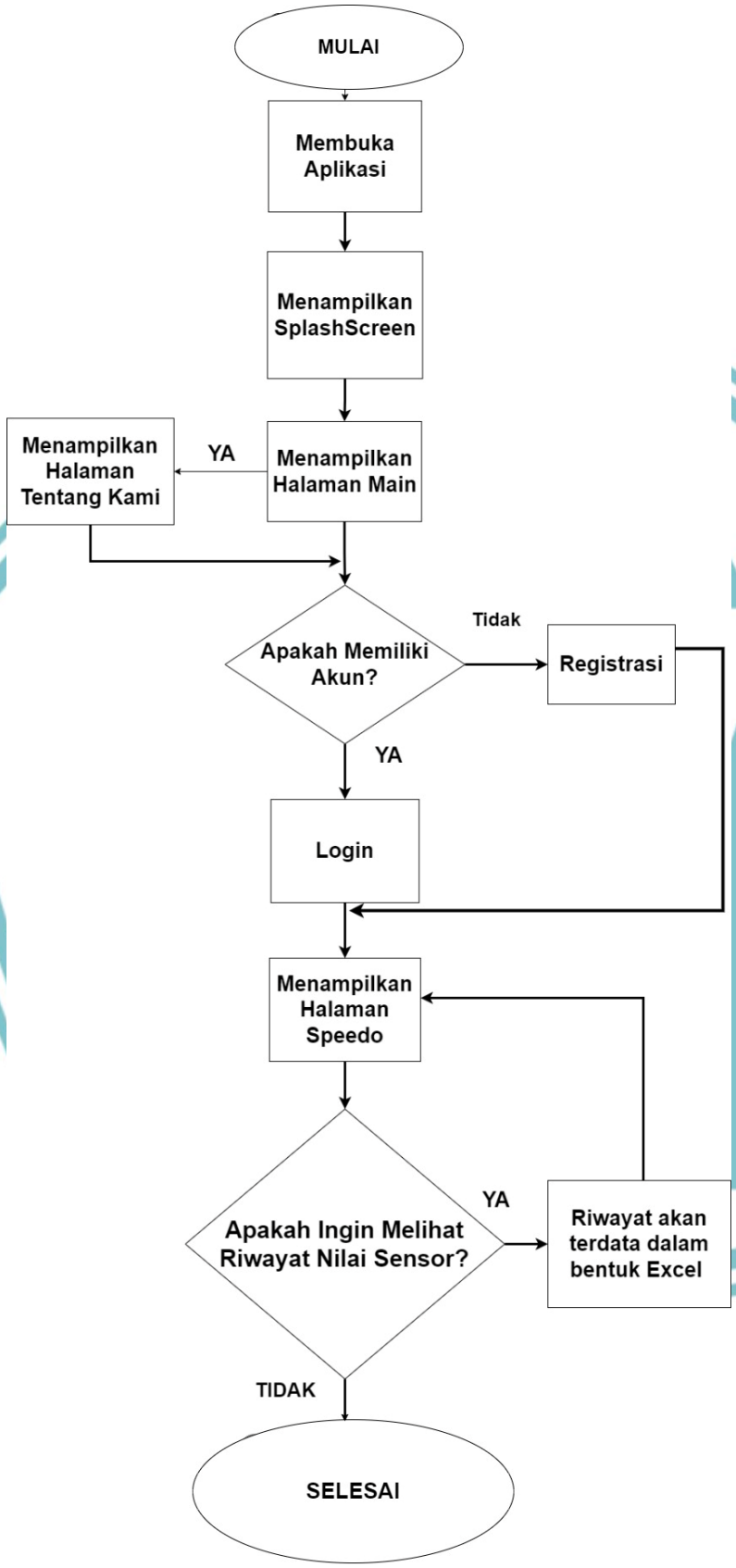
3.1.2 Cara Kerja Alat



Gambar 3. 1 Ilustrasi penggunaan alat pemurnian air laut menjadi air bersih

Menunjukan ilustrasi sistem saat warga akan melakukan filterisasi mengunakan alat, terdapat 1 operator untuk memantau dan mengontrol penggunaan pompa air. Yang dapat dijelaskan alur kerjanya pada langkah berikut:

1. Keterangan pada nomer 1 menjelaskan yaitu pompa air akan menyedot air yang akan diteruskan pada filter karbon
2. Pada keterangan nomer 2 yaitu filter karbon berfungsi untuk menjernihkan air lalu diteruskan oleh pompa air pada keterangan nomer 3
3. Setelah air diteruskan pada pompa pada keterangan nomer 3 air akan disaring oleh membran Ro yang terdapat pada keterangan nomer 4 yang berfungsi untuk menyaring kadar garam pada air laut
4. Setelah melewati nomer 4 (membran Ro) maka air akan disaring kembali oleh filter karbon pada keterangan nomer 5
5. Setelah melewati filter karbon maka air akan tertampung pada bak penampung air dan akan dibaca hasilnya menggunakan sensor salinitas,sensor pH dan sensor turdibity pada keterangan nomer 6



Gambar 3. 2 Flowchart Aplikasi Water Filter

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
   a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
   b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Flowchart di atas menggambarkan alur kerja sebuah aplikasi, dimulai dari pengguna membuka aplikasi dan menampilkan layar pembuka atau *splashscreen*. Setelah *splashscreen*, aplikasi menampilkan halaman utama di mana pengguna dapat mengakses berbagai fitur. Pada tahap ini, pengguna dihadapkan pada pertanyaan apakah mereka sudah memiliki akun. Jika pengguna sudah memiliki akun, mereka akan diarahkan ke proses login. Jika belum, mereka akan diarahkan ke halaman registrasi untuk membuat akun baru. Setelah berhasil login, aplikasi menampilkan halaman "Speedo", yang mungkin berisi informasi penting atau dashboard terkait aplikasi. Selanjutnya, pengguna diberikan opsi untuk melihat riwayat nilai sensor. Jika pengguna memilih untuk melihatnya, riwayat tersebut akan didata dalam bentuk Excel. Jika tidak, aplikasi akan tetap berada pada halaman "Speedo". Alur ini menunjukkan bagaimana pengguna berinteraksi dengan aplikasi dari awal hingga menggunakan fitur untuk melihat riwayat nilai sensor.

3.1.3 Spesifikasi Alat

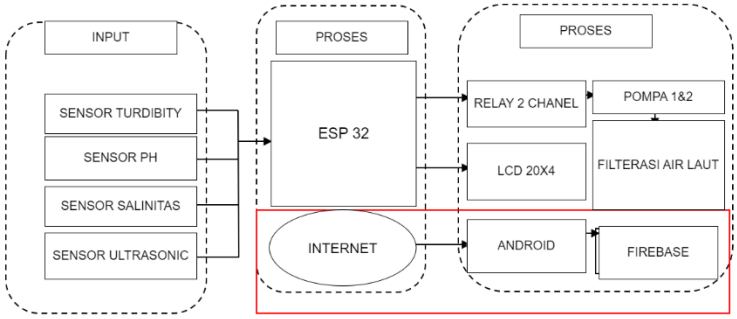
Perangkat keras maupun lunak yang digunakan pada tugas akhir memiliki spesifikasi tertentu. Tabel 3.1 menunjukkan spesifikasi perangkat untuk merancang aplikasi *Water Filter*.

Tabel 3. 1 Spesifikasi perangkat untuk merancang aplikasi

Nama Alat	Spesifikasi
Smartphone	ITEL P55 8/128 GB
Laptop	OS Windows
Android Studio	Koala 2024.1.1
Firebase	*Firebase* Android BoM 33.1.1

3.1.4 Diagram Blok

Diaram blok terdapat input, proses, dan output. Diagram blok sistem alat dapat dilihat pada Gambar 3.3



Gambar 3. 3 Diagram Blok Alat Pendeteksi Kandungan Air

Bagian proses sistem terdiri dari ESP32 yang menjalankan program di Arduino IDE untuk mengukur kadar garam, kejernihan air, pH, dan mengatur debit air otomatis. Outputnya mencakup relay 2 channel yang mengontrol dua pompa air untuk distribusi air ke filter reverse osmosis, dan filter karbon aktif. LCD 20x4 menampilkan hasil pembacaan dari tiga sensor. Aplikasi Android memungkinkan monitoring dan kontrol jarak jauh melalui internet, termasuk pengaturan pompa air dan pembacaan hasil filtrasi seperti pH, kadar garam, kejernihan air, dan ketinggian air.

3.2 Realisasi Alat

Pada sub bab ini akan membahas mengenai realisasi yang meliputi realisasi pembuatan *Firebase*, menyambungkan *Firebase* ke aplikasi Android Studio, membuat rancangan notifikasi pop up, dan realisasi program aplikasi Android. Berikut ini merupakan realisasi alat dari tugas akhir "Rancang Bangun Alat Pemurnian Air Laut Berbasis Android"

3.2.1 Realisasi Pembuatan *Database*

Pada bagian penerima, media transmisi yang digunakan adalah *Firebase* sebagai *monitoring* data secara *real-time* sistem pendeteksi kualitas air. Setelah data dari sensor pengirim terbaca, ESP32 mengirimkan data tersebut ke internet, kemudian data akan diterima kembali melalui internet. *Firebase* yang digunakan memiliki nilai *Realtime database*, yang berarti data yang diterima akan terus diperbarui secara otomatis sesuai dengan kondisi terkini dari sensor pengirim. Dalam hal ini, setiap perubahan data sensor yang terdeteksi oleh ESP32 akan segera memperbarui data yang ada di *Firebase*

© Hak Cipta milik Politeknik Negeri Jakarta
Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

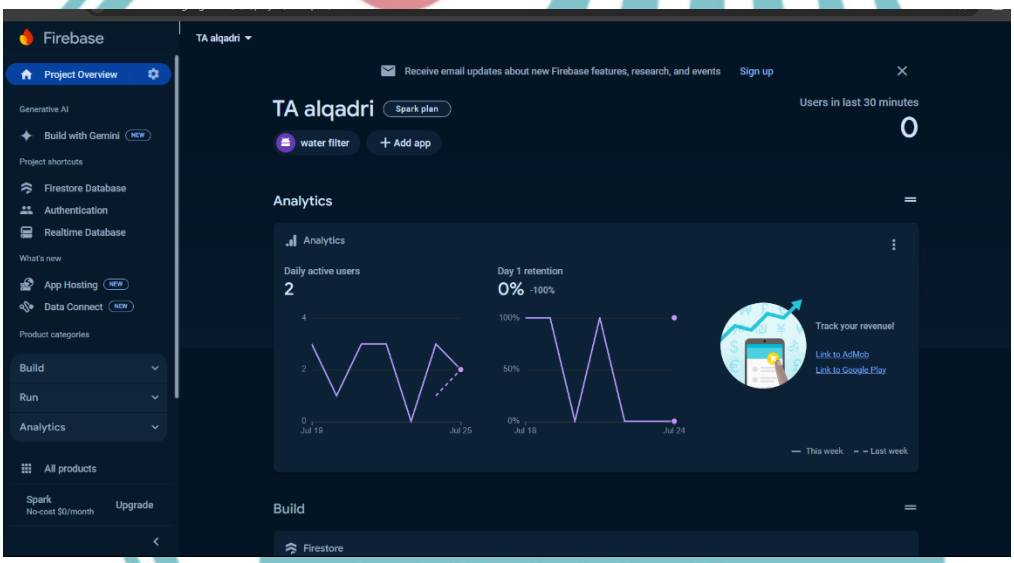
tanpa perlu menunggu proses penyimpanan manual. Memastikan bahwa data yang ditampilkan di aplikasi Android selalu merupakan data terbaru.

Pembuatan *database Firebase* dilakukan dengan cara seperti berikut:

1. Membuat *project* baru

Membuat *project* baru *database* pada *Firebase* dengan cara sebagai berikut:

a. Membuka halaman web *Firebase* yaitu https://firebase.google.com/?hl=id
b. Memulai *project Firebase* dengan mengklik "Go Console"
c. Membuat *project Firebase* dengan mengklik "Add Project"
d. Memberi nama *project*



Gambar 3. 4 Tampilan Awal Firebase

Pada Gambar 3.4 menunjukkan tampilan awal *project Firebase* jika sudah membuat projek baru.

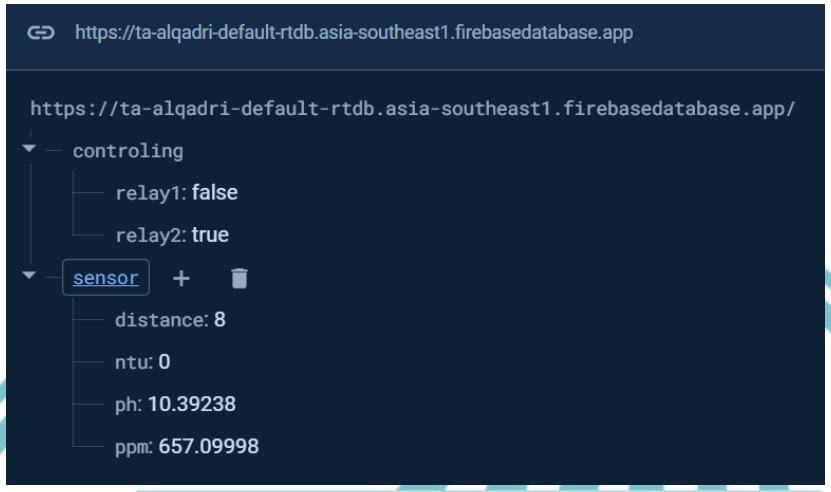
1. Membuat *Real-time Database*

Database dibuat menggunakan *Firebase* secara *real-time*, sehingga data yang diterima dapat diproses dengan cepat dan mudah dalam pengelolaannya. *Real-time database* akan menyimpan data berdasarkan data yang dikirimkan oleh sensor pengirimnya. *Database* digunakan untuk menyimpan data sensor sementara, yang kemudian ditampilkan oleh aplikasi Android. Setelah disimpan ke *database*, nilai-

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

nilai ini akan dikirim ke aplikasi Android untuk ditampilkan. Pada Gambar 3.5 menunjukkan tampilan *database* pada *Firebase*.

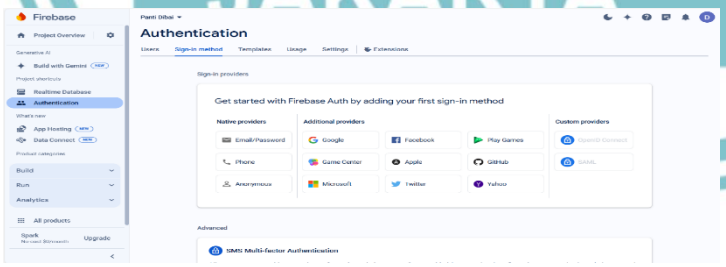


Gambar 3. 5 Tampilan *Database Real-time Firebase*

Pada Gambar 3.5 menunjukkan tampilan pada *database real-time*. *Real-time database* yang dipakai adalah *test mode*. *Test mode* dapat mengakses data dan melakukan perubahan data tanpa *private*.

2. Membuat *Authentication*

Firebase authentication menyediakan *Web API Key*. *Web API Key* untuk menyambungkan *real-time database* ke platform Android Studio. Jika sudah masuk pada menu *authentication* maka untuk mengaktifkannya pilih "*Get started*". Pada Gambar 3.6 tampilan pada *authentication*.



Gambar 3. 6 Tampilan pada *Authentication*

Pada Gambar 3.6 menunjukkan tampilan *authentication* jika sudah diaktifkan. *Web API Key* akan terlihat di *project setting* pada bagian *General*.

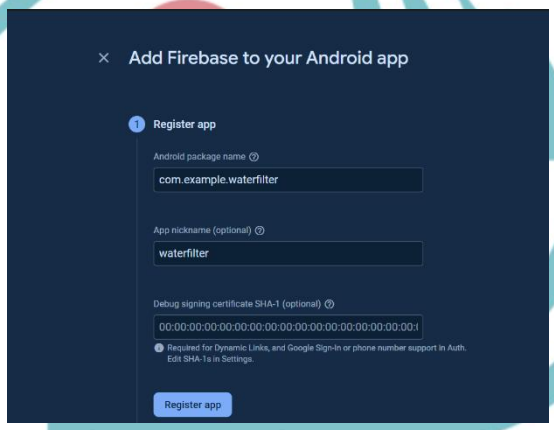
© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3.2.2 Penyambungan database ke program Aplikasi Android

1. Menyambungkan *Firebase* ke Aplikasi Android

Pada tampilan awal *Firebase*, pilihan Android untuk mendaftar aplikasi Android yang sudah dibuat pada *Firebase* untuk menyambungkan *database* dengan aplikasi. Pada bagian *register* app, diarahkan untuk mengisi informasi mengenai aplikasi Android yang akan didaftarkan. *Firebase* memerlukan identitas unik aplikasi untuk memastikan bahwa data yang dikirimkan dan diterima berasal dari aplikasi yang benar. Pada Gambar 3.7 menunjukkan tampilan *register* untuk aplikasi.

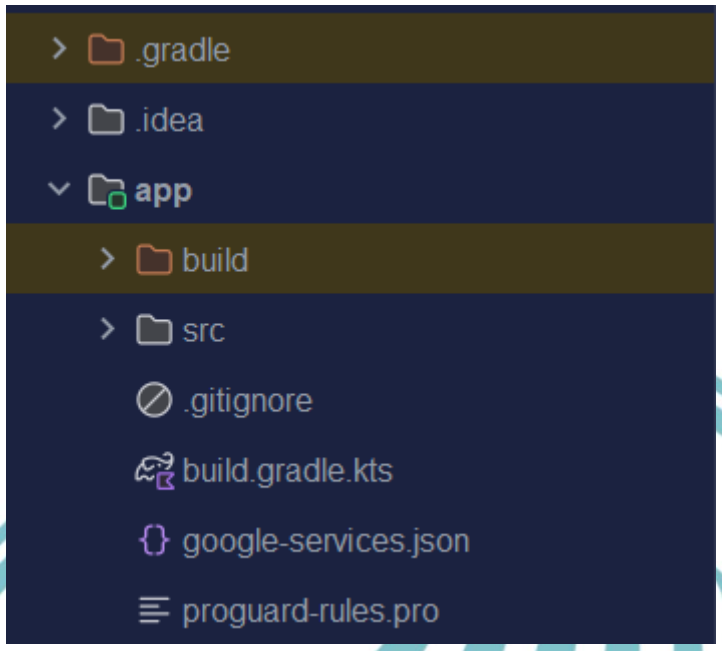


Gambar 3. 7 Tampilan Register Aplikasi

Pada Gambar 3.7 menunjukkan pendaftaran aplikasi yang akan disambungkan. *android package name diisi* dengan *package name* yang sudah dibuat pada Android Studio di publishing pada menu *project setting*. Package name dibuat manual dengan package name yang unik. App nickname merupakan nama aplikasi yang dibuat dan debug signing certificate SHA-1 diisi dengan Keystore pada Android Studio di bagian *Firebase*. Jika pendaftaran sudah dilakukan maka diarahkan untuk mendowload *google-services.json* dan akan di *upload* pada Android Studio di bagian *.gradle/app/*

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 3. 8 B*uild gradle json*

2. Menyambungkan Aplikasi Android ke Realtime Database

Menyambungkan Android di halaman *Realtime* pada *Firebase* yang akan ditampilkan dengan cara membuat *Activity* untuk memulai , kunjungi *Firebase Console* dan membuat *project* baru , setelah *project* dibuat klik "*add app*" dan pilih platform android.

Buka file " *build.gradle*" di Android Studio dan tambahkan depedensi untuk *Firebase Realtime Database*

```
implementation'com.google.firebase:firebase-database:20.3.0'
```

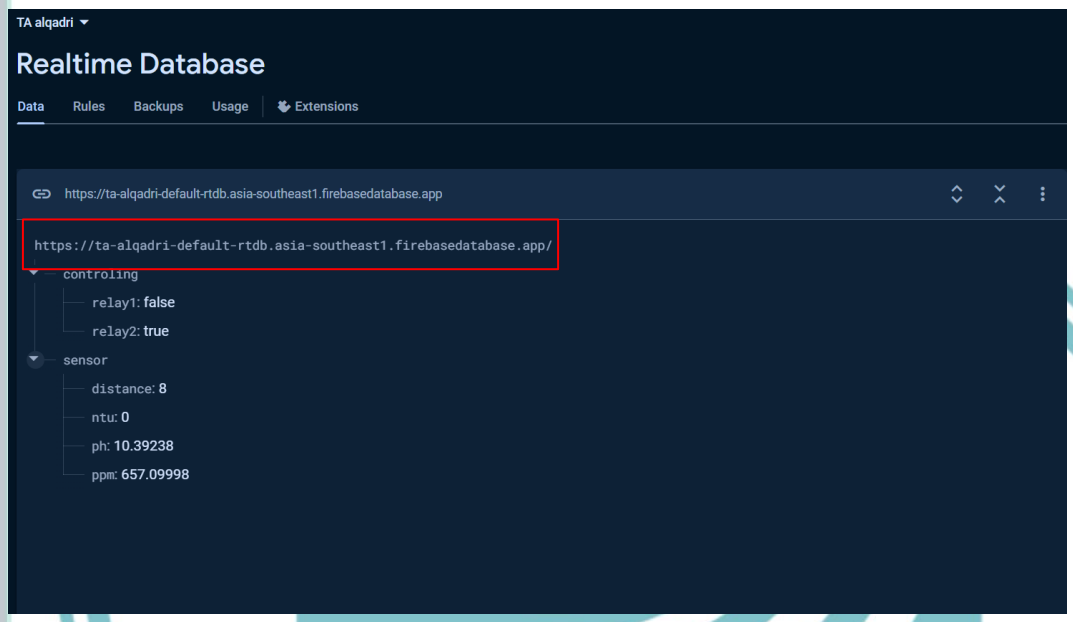
Pastikan juga untuk menambahkan plugin *Google Services* di bagian bawah file *build.gradle* (*Project*). Didalam *project*, buat *instance* dari'*FirebaseDatabase'* dan ' *DatabaseReference'*. Contoh kode untuk menulis data ke *Realtime Database*

```
DatabaseReference database

FirebaseDatabase.getInstance().getReference();

database.child("sensor").child("pH");
```

© Hak Cipta milik Politeknik Negeri Jakarta

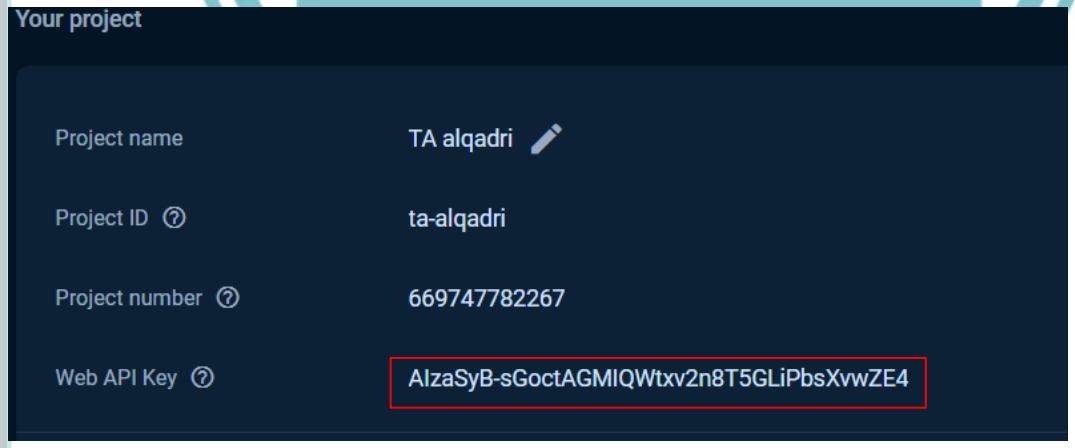
Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Firebase URL dapat dilihat pada tampilan *Realtime Database* dan *Firebase* token bisa dilihat pada *Web API Key*. Pada Gambar 3.9 menunjukkan tampilan *Realtime Database* pada *Firebase*.



Gambar 3. 9 Tampilan Realtime Database pada Firebase

Pada Gambar 3.9 menunjukkan tampilan *Realtime Database* pada *Firebase*. Pada kotak berwarna merah merupakan *Firebase* URL. URL akan diisi pada *Activity* sebagai alamat untuk menghubungkan *Firebase* dengan aplikasi di Android Studio. Pada Gambar 3.10 menunjukkan tampilan pada *project setting* pada *Firebase*.



Gambar 3. 10 Tampilan *Api Key* pada *Firebase*

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Pada gambar 3.10 menunjukkan tampilan *project setting* pada *Firebase*. Pada kotak warna merah merupakan *Web API Key* yang menunjukkan *Firebase* token. *Firebase* token diisi pada *google-service.json* di Android Studio. Untuk memunculkan *Web API Key* diperlukan untuk Membuat *Authentication* terlebih dahulu. Jika *Firebase* URL dan *Firebase* token sudah diisi pada *properties Firebase* di Android Studio maka *Firebase* sudah tersambung ke Android Studio.

3 Menghubungkan Storage Firebase

Untuk menghubungkan *Firebase Storage* ke Android Studio, pertama-tama Anda perlu mengatur *project Firebase* di *Firebase Console*. Buat proyek baru atau pilih proyek yang sudah ada, kemudian tambahkan aplikasi Android Anda ke *project* tersebut dengan memberikan nama paket aplikasi dan mengikuti petunjuk hingga. Anda mendapatkan file *google-services.json*. Selanjutnya, salin file *google-services.json* tersebut ke dalam folder app pada *project* Android Studio Anda. Setelah itu, tambahkan dependensi *Firebase* ke dalam file *build.gradle*. Pada tingkat proyek, tambahkan

```
implementation("com.google.firebase:firebase-storage:21.0.0")
```

Setelah proyek Android Studio terhubung ke *Firebase*, Anda perlu mengaktifkan *Firebase Storage* di *Firebase Console* dengan mengakses bagian ***Storage*** dan mengklik ***Get Started***. Setelah itu, atur aturan keamanan sesuai kebutuhan. Untuk menggunakan *Firebase Storage* di aplikasi Anda, inisialisasi *Firebase Storage* di kelas *Java* atau *Kotlin* dengan

```
FirebaseStorage storage = FirebaseStorage.getInstance();
StorageReference storageRef = storage.getReference();
```

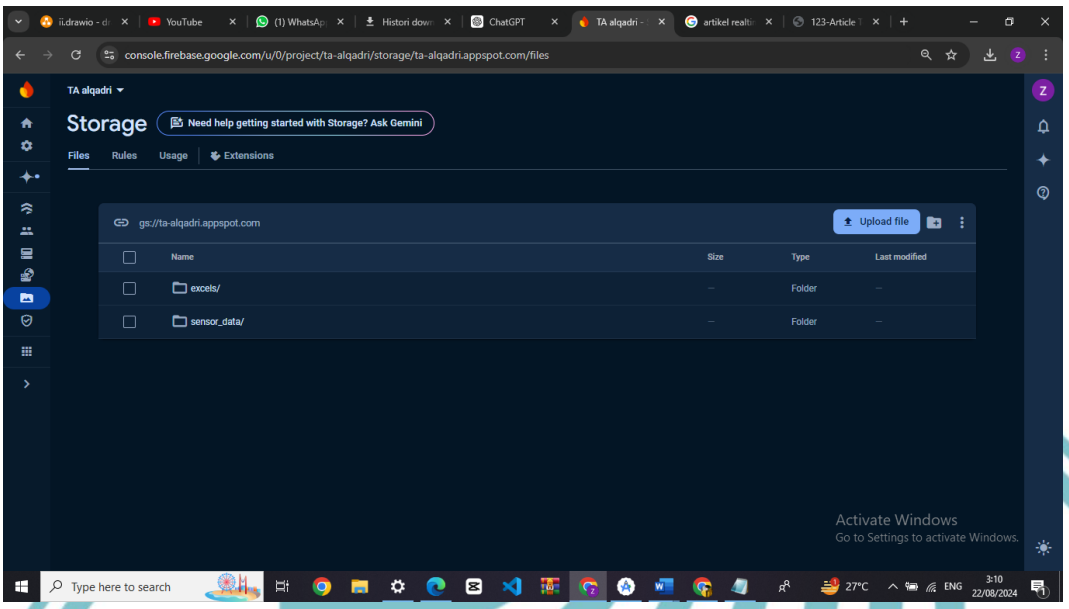
dan buat referensi penyimpanan menggunakan

```
StorageReference storageRef = storage.getReference();
```

Dan pada gambar 3.11 menunjukan tampilan pada Storage Firebase

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 3.11 Storage Firebase

3.2.3 Perancangan Aplikasi Android

Aplikasi ini dirancang dan dibuat dengan menggunakan Android Studio dengan bahasa yang digunakan yaitu bahasa *Kotlin* dan diberi nama "Water Filter" Aplikasi ini digunakan untuk *monitoring* dan *controlling* parameter yang dibutuhkan air seperti pH, NTU, PPM, dan sensor Ultrasonic secara *Realtime*. Aplikasi yang dirancang akan digunakan untuk menampilkan hasil pembacaan sensor Ph, sensor turdibity dan sensor salinitas, sensor ultrasonic dari mikrokontroler. *Smartphone* yang sudah mengunduh aplikasi harus terhubung dengan internet agar dapat menerima data sensor dari *Firebase*. Pada Gambar 3.2 yang merupakan *flowchart* perancangan aplikasi Android..

Aplikasi ini dapat menampilkan data sensor Ph, sensor Turdibity dan sensor Salinitas, sensor Ultrasonic yang tersimpan dalam *Firebase*. Sebelum membuka aplikasi, *Smartphone* android harus terhubung dengan internet. Setelah aplikasi dibuka, akan menampilkan tampilan spalshscreen dari aplikasi "Water Filter". Setelah tampilan spalshscreen akan menampilkan tampilan Main Menu dan yang terakir akan menampilkan halaman Speedo

3.2.4 Realisasi Program Aplikasi Android

Program aplikasi ini dirancang dengan menggunakan Android Studio. Aplikasi yang dibuat diberi nama "*Water Filter*". Aplikasi tersebut digunakan untuk

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

menampilkan hasil pembacaan sensor pH, sensor Turdibity , sensor Salinitas dan sensor Ultrasonic dari Mikorkontroler , *Smartphone* yang mempunyai aplikasi "*Water Filter*" harus terhubung dengan internet agar dapat mendapatkan notifikasi.

1. Membuat Tampilan *Splash Screen*

Splash Screen adalah tampilan awal pada saat membuka aplikasi Android. *Splash screen* berfungsi untuk perkenalan awal tentang aplikasi Android "*Water Filter*". *Splash screen* akan menampilkan ikon aplikasi dan nama aplikasi. Gambar 3.11 menunjukkan tampilan *splash screen* pada aplikasi.

Gambar 3. 11 Tampilan *Splash Screen*

Berdasarkan Gambar 3.11 menunjukkan tampilan *splash screen* dengan menggunakan beberapa Kode Kotlin. Pada Gambar 3.11 merupakan *sketch* untuk tampilan *splash screen* ditunjukan pada Kode Kotlin sebagai berikut.

```
private lateinit var binding:ActivitySplashScreenBinding

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

binding=ActivitySplashScreenBinding.inflate(layoutInfla

ter) setContentView(binding.root)
```

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Kode program di atas merupakan kode dasar yang akan digunakan untuk setiap halaman, menggunakan metode *ViewBinding* dengan deklarasi `private lateinit var binding: ActivitySplashScreenBinding`. Metode ini memungkinkan *variabel binding* digunakan dalam berbagai kondisi.

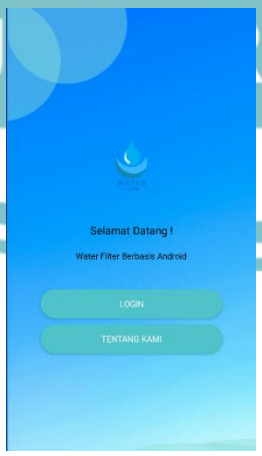
Kode berikutnya, yaitu *override* yang disertai dengan *binding* dan *layoutInflater*, memungkinkan *layout* XML untuk *activity* ini di-*inflate*, lalu *setContentView* berfungsi untuk menampilkan activity yang di dalamnya terdapat *binding.root*, menjadikan activity yang tampil sesuai dengan layout yang telah di-*binding* sebelumnya

```
Handler().postDelayed({
startActivity(Intent(this, MainActivity::class.java))
finish()
},3000)
```

Kode *Kotlin* di atas akan membuat *Activity splash screen* menutup dan melanjutkan ke *Activity login* setelah tampil selama 3000 ms di layar.

2. Membuat Tampilan Menu Masuk

Tampilan Menu Masuk pada aplikasi "*Water Filter*" akan ditampilkan setelah *splash screen* berjalan, untuk membuat tampilan perlu menambahkan *screen* baru yaitu *screen* Menu Masuk. Gambar 3.12 menunjukkan tampilan Menu Masuk pada Aplikasi.



Gambar 3. 12 Tampilan Menu Masuk

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Berdasarkan gambar 3.13 menunjukkan tampilan Menu Masuk pada aplikasi dengan tombol "*Login*" untuk membuka tampilan Speedo, dan "*About*" tentang aplikasi. Berikut ini Kode Kotlin yang digunakan

```
private lateinit var binding: ActivityMainBinding
super.onCreate(savedInstanceState)
binding = ActivityMainBinding.inflate(layoutInflater)
setContentView(binding.root)
binding.buttonlogin.setOnClickListener
val intent = Intent (this, SpeedoActivity::class.java)
startActivity(intent)}
binding.buttontentang.setOnClickListener{
val intent = Intent(this, tentangkami::class.java)
startActivity(intent)
}
}
}
```

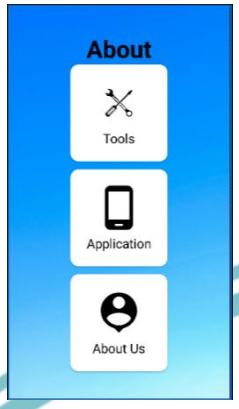
Pada fungsi onCreate, inisialisasi tampilan dilakukan dengan menggunakan `ActivityMainBinding` untuk mengikat layout. Fungsi `setOnClickListener` pada `binding.buttonlogin` dan `binding.buttontentangkami`.Ketika tombol "Tentang Kami" diklik, pengguna diarahkan ke `tentangkamiactivity` untuk tentang aplikasi . Begitu pula, jika tombol login diklik, pengguna akan diarahkan ke `loginactivity` untuk bantuan mengenai kerja alat. Kedua tindakan ini memanfaatkan Intent untuk memulai aktivitas baru

3. Membuat Tampilan *About*

Tampilan *About* berisi nama serta nim dari pembuat aplikasi dan alat. Gambar 3.14 menunjukkan tampilan *About* pada aplikasi.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
   a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
   b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Gambar 3. 13 Tampilan *About*

Tampilan *About* akan menampilkan card "*Tools*" , "Application", "About us". "Tools" untuk menampilkan halaman Tentang Alat , "Application" untuk menampilkan halaman tentang aplikasi dan tombol "about us" untuk menampilkan halaman mahasiswa. Berikut adalah Kode Kotlin yang digunakan

```
private lateinit var binding: ActivityTentangkamiBinding

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding =
ActivityTentangkamiBinding.inflate(layoutInflater)
    setContentView(binding.root)

    binding.App.setOnClickListener {
        val intent = Intent (this,
AplikasiActivity::class.java)
        startActivity(intent)
    }

    binding.Alat.setOnClickListener {
        val intent = Intent(this,
ToolsActivity::class.java)
        startActivity(intent)
    }

    binding.Pembuat.setOnClickListener {
        val intent = Intent(this,
MahasiswaActivity::class.java)
        startActivity(intent)} }
```

Pada fungsi `onCreate`, inisialisasi tampilan dilakukan dengan menggunakan `ActivityTentangKamiBinding`.Untuk mengikat layout Fungsi `setOnClickListener` pada `binding.alat, binding.app,binding.pembuat` Ketika tombol "*Tools*" diklik, pengguna diarahkan ke `toolsactivity` untuk halaman aplikasi . Begitu pula, jika tombol *"app"* diklik, pengguna akan diarahkan

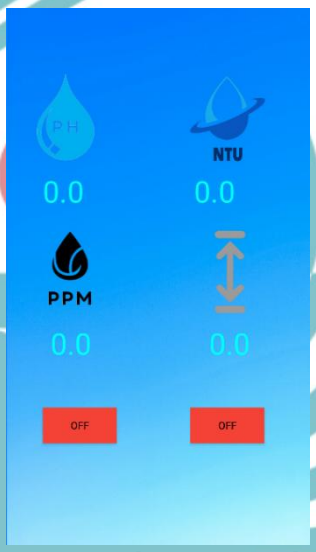
© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

ke `aplikasiactivity` untuk bantuan mengenai kerja alat. Kedua tindakan ini memanfaatkan `Intent` untuk memulai aktivitas baru

4. Membuat Halaman *SpeedoActivity*

Halaman SpeedoActivity menampilkan data yang diambil dari *Realtime Database* yang mencakup pengukuran dari sensor pH, sensor turbidity, sensor salinitas, dan sensor ultrasonik. Gambar 3.15 menunjukkan Halaman *SpeedoActivity* pada aplikasi.



Gambar 3. 14 Halaman *SpeedoActivity*

Berdasarkan gambar 3.15 menunjukkan Halaman *SpeedoActivity* pada aplikasi. Terdapat empat parameter yang menunjukan kondisi Air yang sudah terfilterisasi.

a. *pH*

Untuk membaca data pH secara Realtime digunakan Kode Kotlin seperti ini

```
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
 binding = ActivitySpeedoBinding.inflate(layoutInflater)
 setContentView(binding.root)

  val firebaseDatabase =
        FirebaseDatabase.getInstance("https://ta-alqadri-
default-rtdb.asia-southeast1.firebasedatabase.app/")
    database = firebaseDatabase.reference
```

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

```
readData()

private fun readData() {
val ph = database.child("sensor").child("ph")
ph.addValueEventListener(object : ValueEventListener {
override fun onDataChange(dataSnapshot: DataSnapshot) {
if (dataSnapshot.exists()) {
val ph: Float = dataSnapshot.getValue(Float::class.java) ?:
0.0f
                binding.nilaiph.text = ph.toString()
```

Dalam kode tersebut, terdapat proses inisialisasi koneksi ke *Firebase Realtime Database* dan pengambilan referensinya. *Variabel FirebaseDatabase* diinisialisasi dengan memanggil fungsi `getInstance` dari *FirebaseDatabase*, yang memerlukan URL spesifik sebagai parameter untuk mengakses *Database* di wilayah Asia Tenggara. Setelah itu, referensi *root* dari *Database* disimpan dalam *variabel database* melalui *firebaseDatabase.reference*. Langkah ini memfasilitasi operasi baca dan tulis pada *Firebase Realtime Database*.

Dalam kode yang diberikan, fungsi readData() digunakan untuk membaca nilai *pH* dari *Firebase Realtime Database*. Pertama, referensi ke *child* "*sensor*" dan *child* "*ph*" di dalam *database* diambil dengan menggunakan `database.child("sensor").child("ph")`.

Kemudian, sebuah `ValueEventListener` ditambahkan untuk mendengarkan perubahan data di lokasi tersebut. Di dalam metode `onDataChange`, dilakukan pengecekan apakah data ada dengan menggunakan `dataSnapshot.exists()`.

Jika data ada, maka nilai pH diambil sebagai `Float` menggunakan `dataSnapshot.getValue(Float::class.java)`, dengan nilai default 0.0f jika nilai tidak ada. Nilai ini kemudian diubah menjadi string dan ditampilkan pada elemen antarmuka pengguna yang terhubung dengan `binding.nilaiph`.

b. NTU

Untuk membaca data pH secara *Realtime* digunakan Kode Kotlin seperti ini

```
private fun readNtu() {
    val ntu = database.child("sensor").child("ntu")

    ntu.addValueEventListener(object : ValueEventListener
{
        override fun onDataChange(dataSnapshot:
DataSnapshot) {
            if (dataSnapshot.exists()) {
                val ntu: Float =
dataSnapshot.getValue(Float::class.java) ?: 0.0f
                binding.nilaintu.text = ntu.toString()
```

Pada kode yang diberikan, fungsi `readNtu()` berfungsi untuk membaca nilai NTU dari *Firebase Realtime Database*. Proses dimulai dengan mengambil referensi ke child "sensor" dan kemudian child "ntu" menggunakan `database.child("sensor").child("ntu")`. Kemudian, ditambahkan `ValueEventListener` untuk mendengarkan setiap perubahan data pada referensi tersebut. Dalam metode `onDataChange`, dilakukan verifikasi apakah data yang diambil ada `(dataSnapshot.exists())`.

Jika data ada, nilai NTU diambil dalam bentuk Float dengan menggunakan `dataSnapshot.getValue(Float::class.java)`, dan jika nilai tidak tersedia, akan menggunakan default 0.0f. Nilai NTU ini kemudian diubah menjadi string dan ditampilkan pada elemen UI yang terhubung dengan binding.nilaintu.

c. PPM

Untuk membaca data pH secara Realtime digunakan Kode Kotlin seperti ini

```
private fun readPPM() {
val ppm = database.child("sensor").child("ppm")
ppm.addValueEventListener(object : ValueEventListener {
override fun onDataChange(dataSnapshot: DataSnapshot) {
if (dataSnapshot.exists()) {
val ppm: Float =
dataSnapshot.getValue(Float::class.java) ?: 0.0f
binding.nilaippm.text = ppm.toString()
```

Pada kode yang diberikan, fungsi `readPPM()` berfungsi untuk membaca nilai PPM dari *Firebase Realtime Database*. Proses dimulai dengan mengambil referensi ke child "sensor" dan kemudian child "ppm" menggunakan

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

`database.child("sensor").child("ppm")`.Kemudian,ditambahkan `ValueEventListener` untuk mendengarkan setiap perubahan data pada referensi tersebut. Dalam metode `onDataChange`, dilakukan verifikasi apakah data yang diambil ada `(dataSnapshot.exists())`. Jika data ada, nilai PPM diambil dalam bentuk `Float` dengan menggunakan `dataSnapshot.getValue(Float::class.java)`, dan jika nilai tidak tersedia, akan menggunakan *default* 0.0f. Nilai PPM ini kemudian diubah menjadi string dan ditampilkan pada elemen UI yang terhubung dengan `binding.nilaippm`.

d. Distance

Untuk membaca data *Distance* secara *Realtime* digunakan Kode *Kotlin* seperti ini

```
private fun readDistance() {
val distance =database.child("sensor").child("distance")

distance.addValueEventListener(object :
ValueEventListener {

override fun onDataChange(dataSnapshot: DataSnapshot) {
if (dataSnapshot.exists()) {
val distance: Float =
dataSnapshot.getValue(Float::class.java) ?: 0.0f
binding.nilaidistance.text = distance.toString()
```

Pada kode yang diberikan, fungsi `readDistance()` berfungsi untuk membaca nilai jarak dari *Firebase Realtime Database*. Proses dimulai dengan mengambil referensi ke child "sensor" dan kemudian child "distance"menggunakan `database.child("sensor").child("distance")`.

Kemudian, ditambahkan `ValueEventListener` untuk mendengarkan setiap perubahan data pada referensi tersebut. Dalam metode `onDataChange`, dilakukan verifikasi apakah data yang diambil ada `(dataSnapshot.exists())`.

Jika data ada, nilai jarak diambil dalam bentuk `Float` dengan menggunakan `dataSnapshot.getValue(Float::class.java)`, dan jika nilai tidak tersedia, akan menggunakan default 0.0f. Nilai jarak ini kemudian diubah menjadi *string* dan ditampilkan pada elemen UI yang terhubung dengan `binding.nilaidistance`.

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian , penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3.2.5 Analisa Perancangan dan Realisasi Aplikasi Android

Proses *sign up* dan *sign in* pada sebuah aplikasi adalah salah satu aspek terpenting dalam menjaga keamanan dan integritas data pengguna. Penggunaan metode *authentication*, terutama yang melibatkan layanan seperti Gmail, memiliki beberapa keunggulan signifikan dibandingkan metode tradisional yang hanya mengandalkan penyimpanan username dan password di dalam *Realtime database*.

Ketika pengguna melakukan *sign up* atau *sign in* menggunakan Gmail, proses tersebut melibatkan konfirmasi identitas melalui layanan email yang terpercaya. Gmail menyediakan mekanisme verifikasi dua langkah, yang berarti selain memasukkan username dan password, pengguna juga harus mengonfirmasi identitas mereka melalui kode yang dikirim ke email atau perangkat lain yang terdaftar. Ini menambah lapisan keamanan ekstra yang sangat penting dalam melindungi akun dari akses tidak sah.

Sebaliknya, metode *sign up* dan *sign in* yang hanya mengandalkan penyimpanan *username* dan *password* dalam *Realtime Database* memiliki beberapa kelemahan. Tanpa adanya mekanisme verifikasi tambahan, siapapun dapat membuat akun dengan informasi palsu dan mengakses layanan dengan mudah. Hal ini meningkatkan risiko keamanan, karena tidak ada jaminan bahwa akun yang dibuat benar-benar dimiliki oleh orang yang sah.

POLITEKNIK NEGERI JAKARTA

© Hak Cipta milik Politeknik Negeri Jakarta
Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB IV

PEMBAHASAN

Setelah melakukan perancangan dan realisasi sistem pada android, proses selanjutnya adalah melakukan pengujian alat. Proses ini merupakan tahapan akhir yang dilakukan dalam pembuatan tugas akhir ini. Pengujian ini dilakukan untuk mengetahui alat dapat berfungsi dengan baik atau tidak. Pengujian dilaksanakan berdasarkan lokasi dan waktu sebagai berikut:

Lokasi : Jalan Margonda Raya , GG masjid alistiqomah RT02 RW 06 no 60

Waktu : 28 Juli 2024

Pelaksanaan : 1. Ahmad Al Qadri

Pembimbing : Dr. Yenniwarti Rafsyam, SST., M.T.

4.1 Pengujian Aplikasi Android

Pengujian ini dilakukan untuk membuktikan bahwa aplikasi Android dapat terhubung dengan *Realtime Firebase* dan mikrokontroler yaitu ESP32, menampilkan data dari sensor turdibity, sensor ultrasonic, sensor pH dan sensor salinitas.

4.1.1 Deskripsi Pengujian

Pengujian ini dilakukan untuk membuktikan bahwa aplikasi Android dapat terhubung dengan Realtime Firebase dan mikrokontroler yaitu ESP32, serta dapat menampilkan sensor *pH*, sensor *salinitas* , sensor *turdibity* dan sensor *ultra sonic*

Adapun alat yang digunakan dalam pengujian ini yaitu:

1. Laptop.
2. *Smartphone.*
3. Alat Pemurnian Air Laut Menjadi Air Bersih

4.1.2 Prosedur Pengujian

Berikut merupakan beberapa tahapan untuk melakukan pengujian aplikasi Android:

1. Menghubungkan *smartphone* dengan jaringan internet agar dapat terhubung dengan *Realtime database.*

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

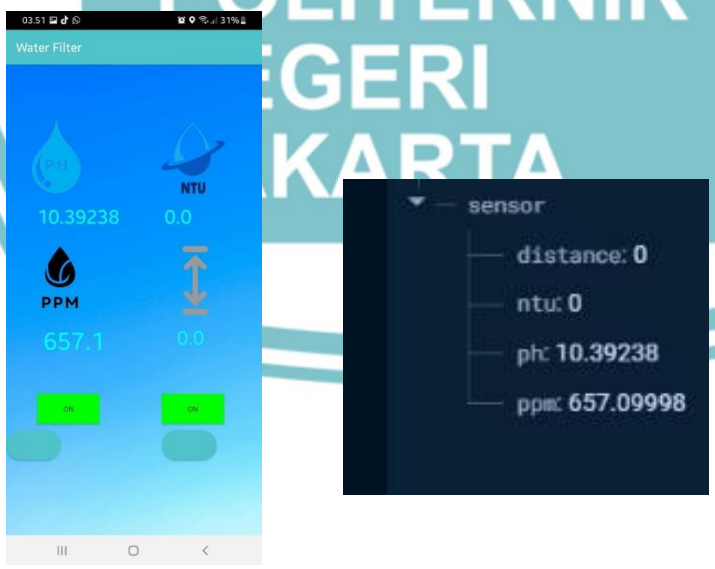
2. Membuka aplikasi Android dan *database* untuk melihat data nilai dari sensor pada mikrokontroler seperti pada Gambar 4.1.
3. Melakukan *monitoring* sensor secara realtime dan *controling* pintu air melalui Android seperti ditunjukkan pada Gambar 4.2 dan Gambar 4.4.

4.1.3 Data Hasil Pengujian Aplikasi Android

Pengujian dilakukan dengan menghubungkat alat seperti sensor dengan internet agar data dapat dikirimkan ke Android sehingga Android dapat menampilkan data hasil sensor secara *Realtime*. Dengan memanfaatkan sistem esp32 yang kemudian diteruskan ke *firebase*. Pengujian yang dilakukan adalah sebagai berikut.

1. **A.) Pengujian pertama, monitoring sensor ultrasonic, sensor pH, sensor salinitas, sensor turdibity.**

   Pengujian yang dilakukan yaitu menampilkan data monitoring sensor dengan mendeteksi sensor ketinggian air, sensor kecepatan angin, dan sensor kelembapan tanah pada sawah. Hasil monitoring dari sensor tersebut akan ditampilkan pada Android dan sistem *Realtime firebase* ditunjukkan Gambar 4.1.



Gambar 4. 1 Data Hasil Sensor Pada Android dan *Firebase*

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

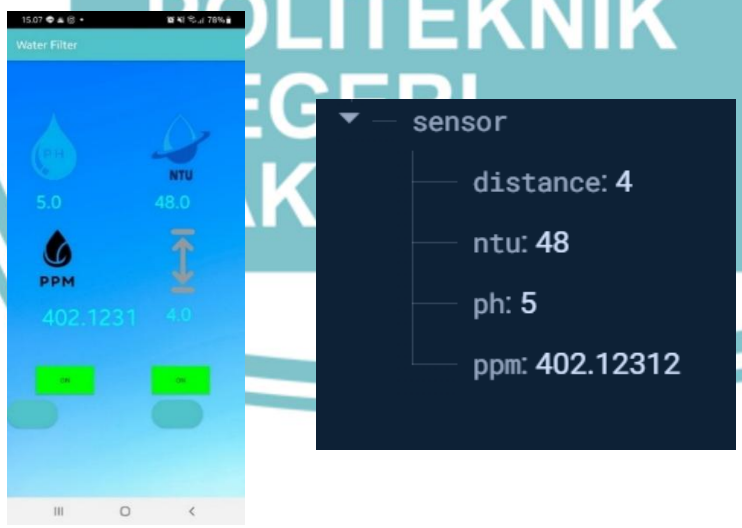
Gambar 4.1 menunjukkan data dari hasil sensor ketinggian air 0 cm dari permukaan ember, sensor pH 10 pada air , sensor salinitas NTU 0 dan sensor turdibity sebesar 657. Pada tampilan Android monitoring sensor terdapat data yang terhubung dengan *firebase* seperti pada data *firebase* pada Gambar 4.1 diatas. Data hasil pengujian sensor ultrasonic, sensor pH, sensor turdibity dan sensor salinitas ditunjukkan pada Tabel 4.1.

Tabel 4. 1 Hasil Pengujian Monitoring Sensor Percobaan Pertama

pH	Turdibity	Salinitas	Ultrasonic
10	657	0	0

B.) Pengujian kedua, monitoring sensor ketinggian air, kecepatan angin, dan kelembapan tanah.

Pengujian yang kedua kembali dilakukan yaitu menampilkan data monitoring sensor dengan mendeteksi sensor ketinggian air, sensor kecepatan angin, dan sensor kelembapan tanah pada sawah. Hasil monitoring dari sensor tersebut akan ditampilkan pada Android dan sistem *Realtime firebase* ditunjukkan Gambar 4.2.



Gambar 4. 2 Data Hasil Sensor Pada Android dan *Firebase* II

Gambar 4.2 menunjukkan data dari hasil sensor ultrasonic 4 cm dari permukaan ember, sensor turdibity 48 pada ember tersebut, dan sensor pH sebesar 5 dan sensor salinitas 402 Pada tampilan Android monitoring sensor

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

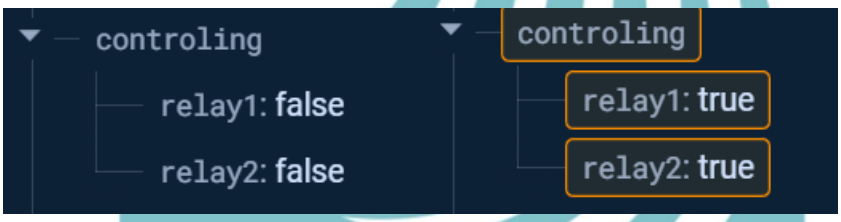
terdapat data yang terhubung dengan *firebase* seperti pada data *firebase* pada Gambar 4.2 diatas. Data hasil pengujian sensor ketinggian air, sensor kecepatan angin, dan sensor kelembapan tanah ditunjukkan pada Tabel 4.2

Tabel 4. 2 Hasil Pengujian Monitoring Sensor Percobaan Kedua

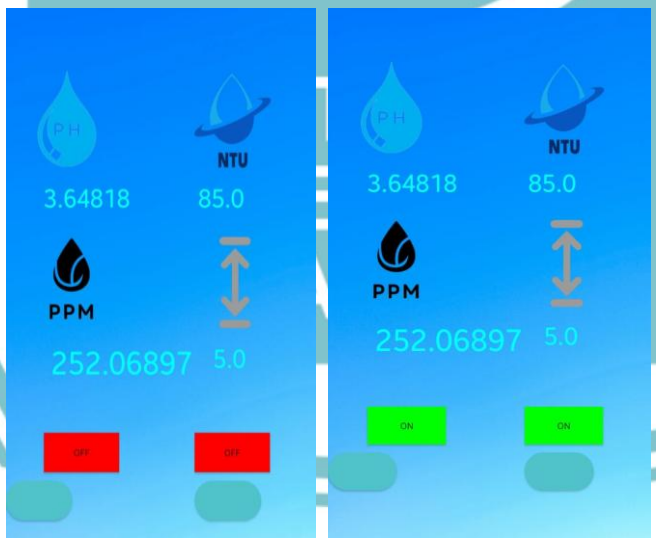
pH	Turdibity	Salinitas	Ultrasonic
5	48	402	4

2. Pengujian pada controling pompa air

Percobaan yang dilakukan yaitu menguji kontrol *switch* pada Android yang terhubung dengan sistem *Realtime firebase* sehingga pintu air dapat dibuka atau ditutup melalui Android. *Switch* kontrol pintu air ditunjukkan pada Gambar 4.3 dan Gambar 4.4



Gambar 4. 3 Tampilan Controlling pada Firebase



Gambar 4. 4 Tampilan Controling pada Aplikasi Water Filter

Gambar 4.4 diatas menunjukkan pembacaan kontrol yang dikirim pada *firebase* dengan hasil *False* yaitu *switch* dimatikan yang menunjukkan pompa air sedang dimatikan, sedangkan pada Gambar 4.4 dengan hasil *True* ini menunjukkan *switch* menyala dan pintu air sedang menyala. Pada Gambar 4.5 diatas akan menunjukkan korelasi yang selaras pada Android dan juga *firebase*

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

yaitu ketika *switch* pada Android ditekan "*ON*" maka sistem *firebase* akan mengirimkan data *switch* yaitu *True* yang berarti menyala, sedangkan ketika *swtich* pada Android ditekan "*OFF*" maka sistem *firebase* akan mengirimkan data *swtich* yaitu *False* yang berarti sedang dimatikan.

4.1.4 Analisa Hasil Data Pengujian Aplikasi Android

Berdasarkan data pengujian yang diperoleh, aplikasi Android berhasil menampilkan hasil monitoring dari sensor pH, sensor kekeruhan (turbidity), sensor salinitas, dan sensor ultrasonik yang terhubung melalui ESP32 secara real-time dengan Firebase. Pada pengujian pertama, sensor ultrasonik mendeteksi ketinggian air 0 cm, sensor pH mencatat nilai 10, sensor kekeruhan menunjukkan nilai 657 NTU, dan sensor salinitas menunjukkan nilai 0 ppm.

Pengujian kedua dilakukan dengan kondisi yang berbeda, di mana sensor ultrasonik mendeteksi ketinggian air 4 cm, sensor pH mencatat nilai 5, sensor kekeruhan menunjukkan nilai 48 NTU, dan sensor salinitas menunjukkan nilai 402 ppm.

Dari hasil pengujian ini, terlihat bahwa aplikasi dapat menangkap dan menampilkan perubahan data sensor dengan baik, sesuai dengan kondisi yang ada. Sensor-sensor yang digunakan dalam pengujian ini menunjukkan performa yang konsisten dan memberikan hasil yang dapat diandalkan untuk monitoring kualitas air. Hal ini membuktikan bahwa sistem yang dibangun, termasuk integrasi antara sensor, mikrokontroler ESP32, dan aplikasi Android, berjalan dengan efektif dan dapat digunakan untuk pemantauan kualitas air secara real-time

4.2 Pengujian Quality Of Service Pada Aplikasi Android

Dalam pengujian Quality of Service terdapat 3 bagian yang harus diperhatikan, yaitu packet loss, delay dan throughut

4.2.1 Deskripsi Pengujian Qualtiy Of Service Pada Aplikasi Android

Pengujian dilakukan dengan melihat hasil pengiriman dan penerimaan data yang dicapture oleh aplikasi wireshark. Parameter yang dicapture akan digunakan untuk melakukan perhitungan QoS sehingga data QoS bisa didapatkan. Adapun alat yang digunakan dalam pengujian ini yaitu :

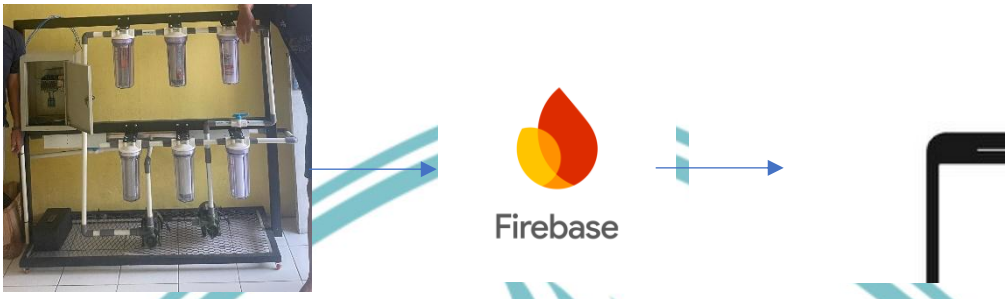
1. Konektivitas Internet yang berasal dari modem.
2. Aplikasi Wireshark

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.2.2 Prosedur Pengujian

1. Menghubungkan laptop yang digunakan dengan koneksi internet dari ORBIT seperti pada gambar 4.5



Gambar 4. 5 Persiapan Rangkaian Pengujian QoS

2. Membuka aplikasi Wireshark
3. Pada aplikasi Wireshark, memilih koneksi yang akan diuji, yaitu koneksi WiFi
4. Menjalankan proses pengambilan data selama 1 menit. Menghentikan proses capture dengan menekan tombol stop.
5. Melihat parameter pada menu bar Statistics dan memilih Capture File Properties, maka akan diperoleh data seperti pada Gambar 4.6


Details

File

Name:	C:\Users\ENDANG~1\AppData\Local\Temp\wireshark_Wi-Fi1WJTR2.pcapng
Length:	849 kB
Hash (SHA256):	1cb94b62c92baaf44c7b5774109055ff0679468462de314197dfdd59715825d9
Hash (SHA1):	82aaac5a5c71df64e69830aba7e8c51cab3e1c04
Format:	Wireshark/... - pcapng
Encapsulation:	Ethernet

Time

First packet:	2024-07-28 22:56:55
Last packet:	2024-07-28 22:57:56
Elapsed:	00:01:01

Capture

Hardware:	AMD Ryzen 5 4500U with Radeon Graphics (with SSE4.2)
OS:	64-bit Windows 11 (23H2), build 22631
Application:	Dumpcap (Wireshark) 4.2.6 (v4.2.6-0-g2acd1a854bab)

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Wi-Fi	0 (0.0%)	none	Ethernet	262144 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	1378	1378 (100.0%)	—
Time span, s	61.781	61.781	—
Average pps	22.3	22.3	—
Average packet size, B	582	582	—
Bytes	802384	802384 (100.0%)	0
Average bytes/s	12 k	12 k	—
Average bits/s	103 k	103 k	—

Gambar 4. 6 Hasil Performansi Jaringan pada Wireshark

Pada gambar 4.6 merupakan hasil dari wireshark yang sudah dijalankan. Untuk menghitung kemampuan nya baik itu througput, delay dan packet loss yang perlu kita ketahui dahulu besaran byte dan packet yang terkirim yaitu

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Bytes 8023

Packet 1378

TimeSpan: 61,781s

4.2.3 Data Hasil Pengujian Quality Of Service

Pengukuran QoS didapatkan dari data – data yang berhasil direkam oleh Wireshark selama 1 menit dengan menggunakan konektivitas internet dari modem. Untuk mengetahui nilai QoS seperti packet loss, throughput, dan delay menggunakan persamaan yang ada pada Bab 2.

a. Packet Loss

Packet loss adalah paket yang hilang dibandingkan dengan jaringan yang diuji di lab. Paket yang dikirimkan diukur dalam satuan persen. Untuk mengetahui nilai packet loss, digunakan persamaan berikut.

$$Packet\ Loss = \frac{paket\ yang\ dikirim - paket\ yang\ diterima}{paket\ yang\ dikirim} \times 100\%$$

$$Packet\ Loss = \frac{1378 - 1378}{1378} \text{ x}100\% = 0\%$$

Dari hasil perhitungan, nilai packet loss adalah 0. Ini menunjukkan bahwa tidak ada paket yang hilang dan semua paket yang dikirim berhasil diterima dengan persentase 100%.

b. Delay

Untuk mengetahui hasil delay dapat dihitung menggunakan persamaan dibawah ini:

$$Delay = \frac{Waktu\ Pengiriminan}{packet\ diterima}$$

$$Delay = \frac{61,781}{1378} = 4,48\text{s}$$

Dari hasil perhitungan, dapat dikatakan bahwa delay sebesar 4,48s detik menunjukkan performa yang kurang memadai, kemungkinan disebabkan oleh

© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

kualitas sinyal yang tidak optimal.

c. Throughput

Untuk mengetahui troughput dapat dihitung menggunakan persamaan 2.1 dibawah ini :

$$Throughput = \frac{Jumlah\ data\ yang\ dikirim}{waktu\ pengirim\ data}$$

$$Throughput = \frac{8023}{61{,}781} = 129{,}86\ bytes/s$$

Dari perhitungan yang dilakukan , diperoleh nilai throughput sebesar 129,86 *bytes*

4.2.4 Analisis Data Hasil Pengujian Quality Of Service

Berdasarkan hasil pengukuran QoS menggunakan Wireshark dengan koneksi internet dari modem, diperoleh beberapa parameter penting. Pertama, nilai packet loss sebesar 0%, menunjukkan bahwa tidak ada paket data yang hilang selama transmisi; semua paket yang dikirim berhasil diterima dengan persentase 100%. Hal ini mengindikasikan kualitas jaringan yang baik dalam hal kehandalan pengiriman data. Kedua, nilai delay yang dihitung sebesar 4,48 detik menunjukkan performa yang kurang memadai. Delay yang cukup tinggi ini kemungkinan besar disebabkan oleh kualitas sinyal yang tidak optimal, yang dapat mempengaruhi kecepatan respon dari aplikasi, terutama dalam konteks real-time monitoring. Ketiga, throughput yang diperoleh adalah sebesar 129,86 bytes/s. Nilai throughput ini menunjukkan jumlah data yang berhasil ditransmisikan dalam jangka waktu tertentu, yang meskipun cukup memadai, masih bisa ditingkatkan untuk mendapatkan performa aplikasi yang lebih baik. Secara keseluruhan, meskipun tidak ada packet loss, tingginya delay dan rendahnya throughput dapat menjadi indikasi adanya masalah dalam koneksi jaringan yang perlu diatasi untuk memastikan aplikasi dapat berjalan lebih efisien dan responsif