BAB V

IMPLEMENTASI DAN PENGUJIAN

5.1. Implementasi

Implementasi sistem dilakukan setelah merancang aplikasi. Implementasi sistem mengimplementasikan kebutuhan - kebutuhan sistem yang dapat menunjang pembuatan sistem tersebut yang meliputi implementasi pembuatan sertifikat *Tizen*, implementasi *Wireless Programming System*, implementasi *Run Deploy Apps*, dan implementasi pengambilan dan penyimpanan data. Implementasi pada aplikasi *Samsung Smartwatch* dalam pengenalan aktivitas permainan bola basket menggunakan *tools* yaitu bahasa pemrograman *Javascript*, menggunakan aplikasi *Tizen SDK*, dan penyimpanan hasil sensor *acceleration gravity* yang dimana meliputi sensor *Accelerometer* dan *Gyroscope* nya menggunakan *Firebase Console*.

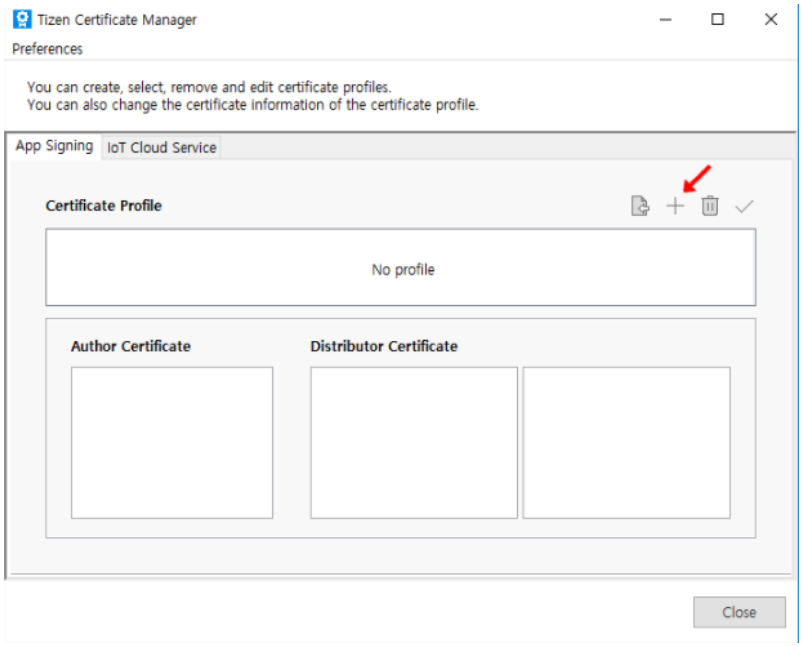
5.2. Implementasi Pembuatan *Tizen* Sertifikat

Pengembangan aplikasi *Samsung Smartwatch* pengenalan aktivitas permainan bola basket membutuhkan *Tizen* sertifikat terlebih dahulu yang dimana berfungsi sebagai menandatangani dan memverifikasi aplikasi, apabila penulis tidak melakukan pembuatan *Tizen* sertifikat maka pada saat melakukan pendeployan tidak akan berhasil yang dimana dengan kata lain *Tizen* sertifikat ini adalah hak akses utama untuk melakukan penggunaan keseluruhan fitur *Tizen SDK*. Dalam *Tizen* sertifikat sendiri terdapat 2 jenis sertifikat yaitu *Author Certificate* dan *Distributor Certificate*, penulis dalam melakukan pengembangan aplikasi *Samsung Smartwatch* pengenalan aktivitas permainan bola basket menggunakan jenis sertifikat yang *Distributor Certificate* dikarenakan untuk mengidentifikasi dan mendapatkan hak istimewa yang memungkinkan penginstalan ke perangkat yang terdafta.

Berikut ini adalah langkah - langkah dari implementasi pembuatan *Tizen* sertifikat sehingga dapat berhasil menggunakan keseluruhan fitur *Tizen SDK* :

5.2.1. Membuat Profil Sertifikat Baru

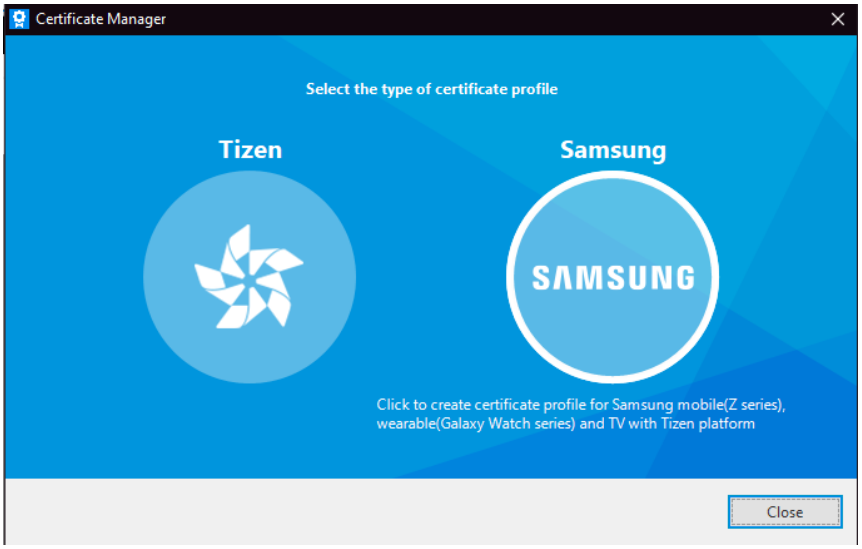
a. Profil sertifikat baru dengan mengklik tombol “+” untuk membuat profil sertifikat, seperti yang ditunjukkan oleh Gambar 5.1 berikut :



Gambar 5. 1 Profil Certifitace Baru

(docs.tizen.org, 2021)

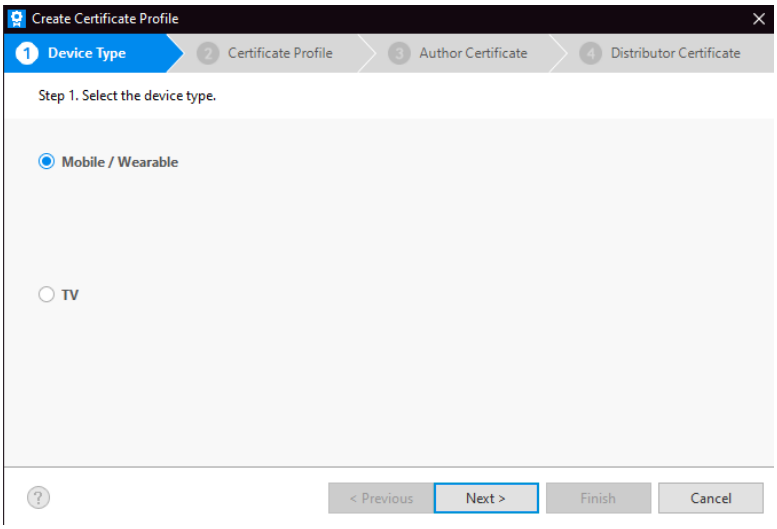
b. Jenis sertifikat yang digunakan oleh penulis adalah jenis *Samsung* yang berfungsi sebagai profil sertifikat untuk menghasilkan sertifikat untuk mengembangkan dan menginstal apliasi ke perangkat *Samsung*, seperti yang ditunjukkan oleh Gambar 5.2 berikut :



Gambar 5. 2 Type Certificate

(docs.tizen.org, 2021)

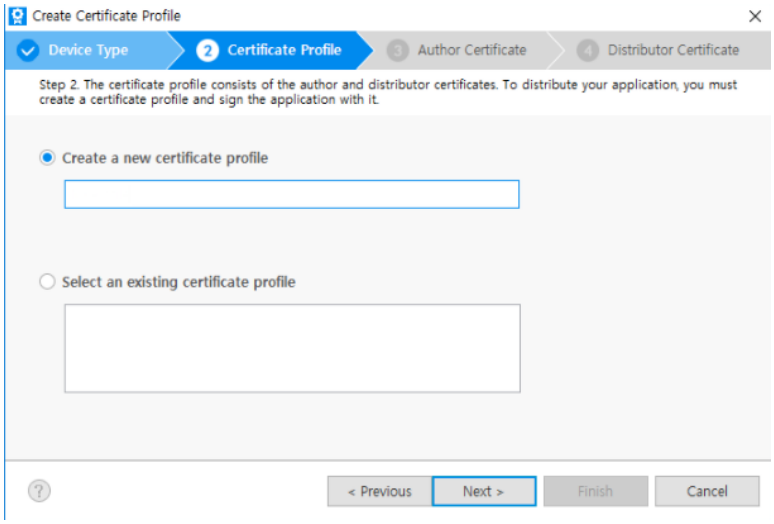
c. Jenis perangkat yang dipilih oleh penulis adalah jenis perangkat *Mobile/Wearable*, seperti yang ditunjukkan oleh Gambar 5.3 berikut :



Gambar 5. 3 Type untuk Device yang akan digunakan

(docs.tizen.org, 2021)

d. Membuat profil sertifikat baru untuk membuat profil baru, seperti yang ditunjukkan oleh Gambar 5.4 berikut :

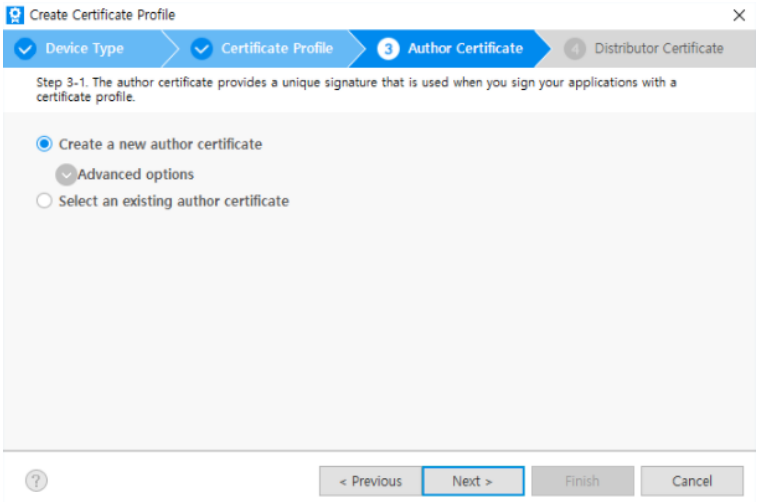


Gambar 5. 4 Membuat nama profile baru

(docs.tizen.org, 2021)

5.2.2. Membuat *Author Certificate* Baru

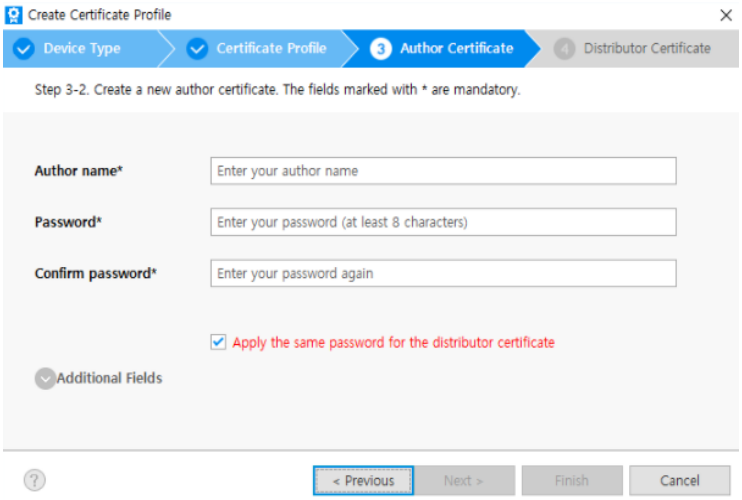
a. Membuat baru atau memilih sertifikat yang ada, penulis disini melakukan pembuatan *Author Certificate* baru, seperti yang ditunjukkan oleh Gambar 5.5 berikut :



Gambar 5. 5 Membuat author certificate baru

(docs.tizen.org, 2021)

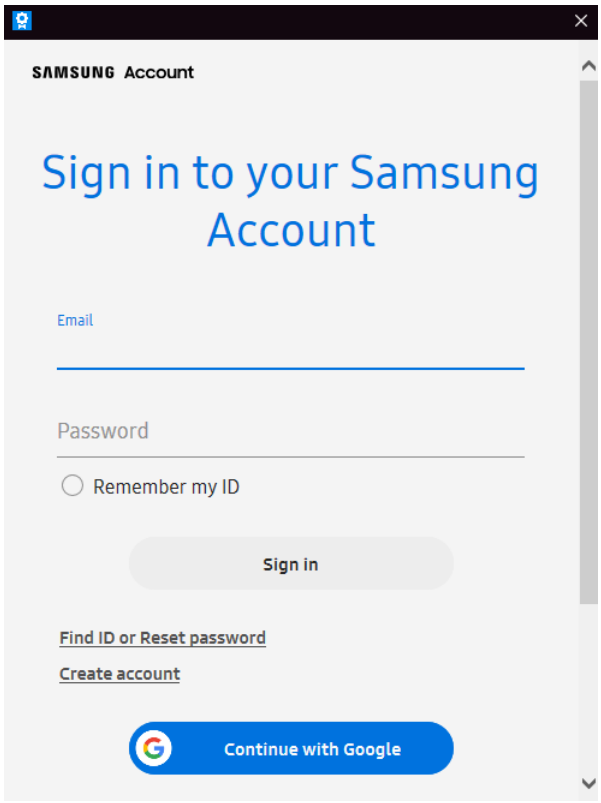
b. Mengisi informasi tentang *Author Certificate* dengan memasukkan nama *Author* dan kata sandi, seperti yang ditunjukkan oleh Gambar 5.6 berikut :



Gambar 5. 6 Informasi untuk author certificate

(docs.tizen.org, 2021)

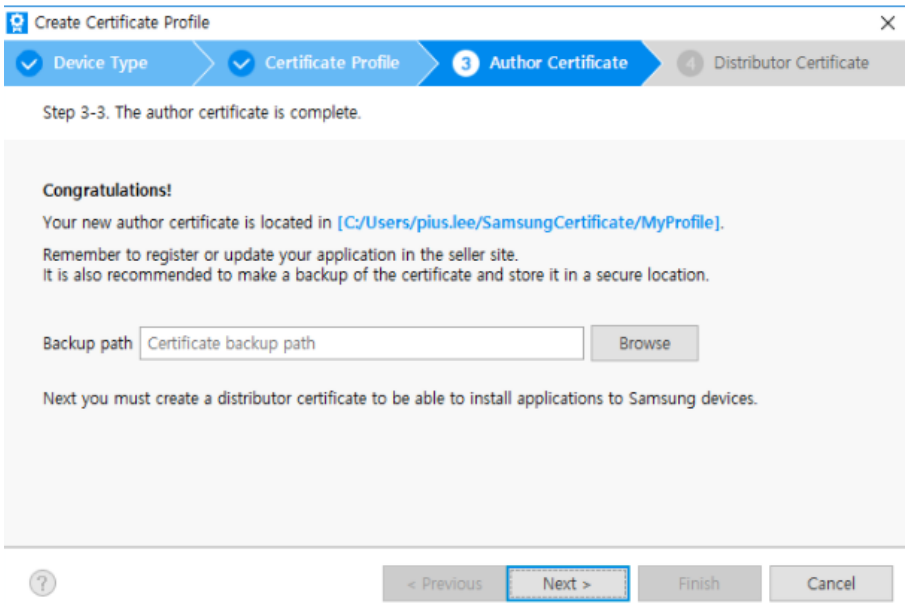
c. Masuk kedalam akun *Samsung* / masuk dengan akun *Google*, penulis disini masuk dengan menggunakan akun *Google*, seperti yang ditunjukkan oleh Gambar 5.7 berikut :



Gambar 5. 7 Masuk ke Akun Samsung

(docs.tizen.org, 2021)

d. Mencadangan pembuatan *Author Certificate* yang telah dilakukan, seperti yang ditunjukkan oleh Gambar 5.8 berikut :

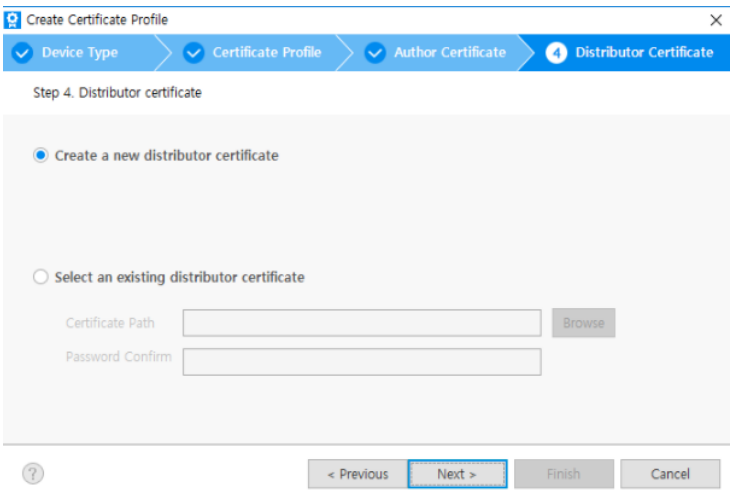


Gambar 5. 8 Menyimpan cadangan author certificate

(docs.tizen.org, 2021)

5.2.3. Membuat *Distributor Certificate* Baru

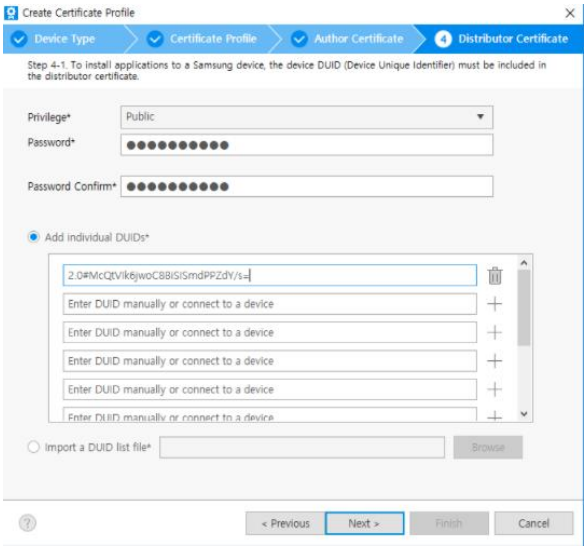
a. Membuat *Distributor Certificate* baru untuk membuat sertifikat baru, seperti yang ditunjukkan oleh Gambar 5.9 berikut :



Gambar 5. 9 Membuat Distributor Certificate baru

(docs.tizen.org, 2021)

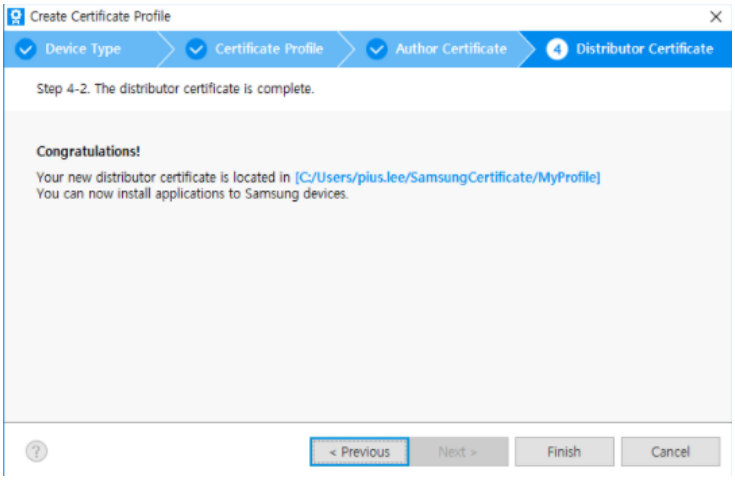
b. Mengisi informasi sertifikat *Distributor* yang memiliki tingkat hak istimewa yang sesuai untuk memastikan bahwa *API* yang diterapkan berfungsi diperangkat, seperti yang ditunjukkan oleh Gambar 5.10 berikut :



Gambar 5. 10 Informasi untuk distributor certificate

(docs.tizen.org, 2021)

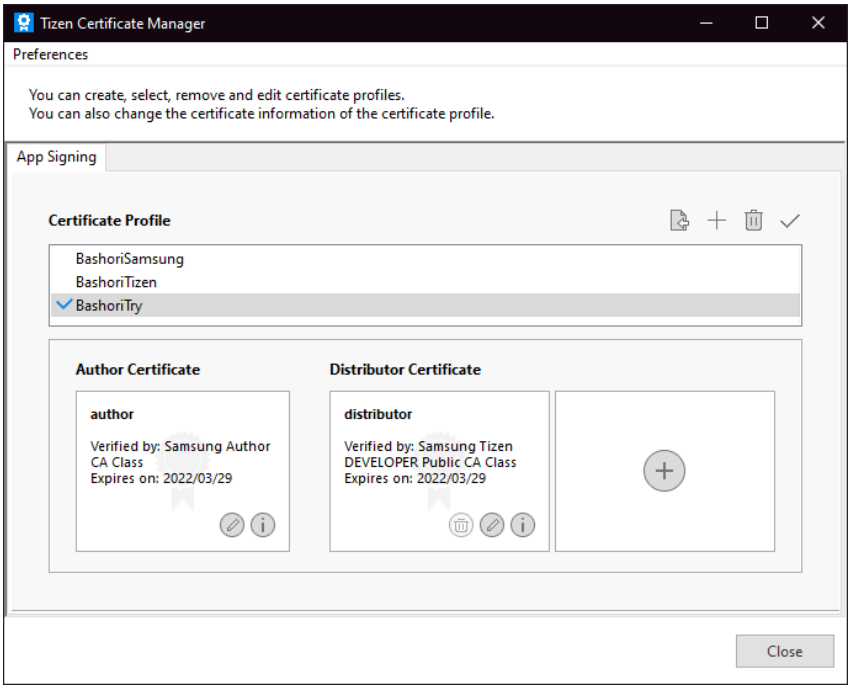
c. Selesai membuat *Distributor Certificate*, klik *next* untuk mendapatkan *Distributor Certificate*, seperti yang ditunjukkan oleh Gambar 5.11 berikut :



Gambar 5. 11 Selesai membuat distributor certificate

(docs.tizen.org, 2021)

d. Klik *finish* untuk menemukan profil yang dibuat beserta informasinya, seperti yang ditunjukkan oleh Gambar 5.12 berikut :



Gambar 5. 12 Mendapatkan certificate profil yang telah dibuat beserta informasinya

(docs.tizen.org, 2021)

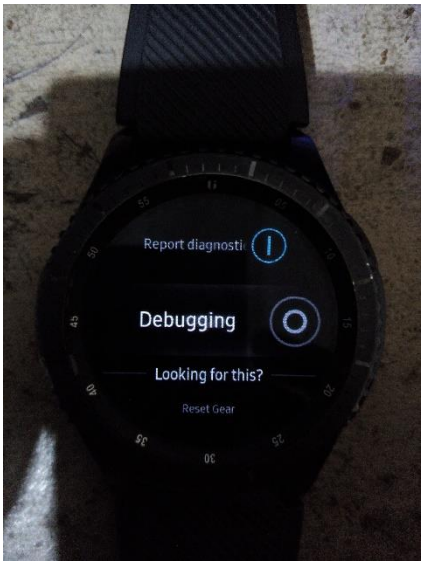
5.3. Implementasi *Wireless Programming System*

Tahap *Wireless Programming System* merupakan proses penyambungan antara *Tizen SDK* dengan perangakat yang digunakan yaitu *Wearabale Samsung Smartwatch Gear S3* dengan adanya *Ip Address* jaringan yang sama dan mengaktifkan *mode debugging* pada *Wearable Samsung Smartwatch Gear S3*. Pada tahap ini bertujuan mengelola perangkat yang terhubung dari komputer *host* ke perangkat *Wearable Samsung Smartwatch Gear S3.*

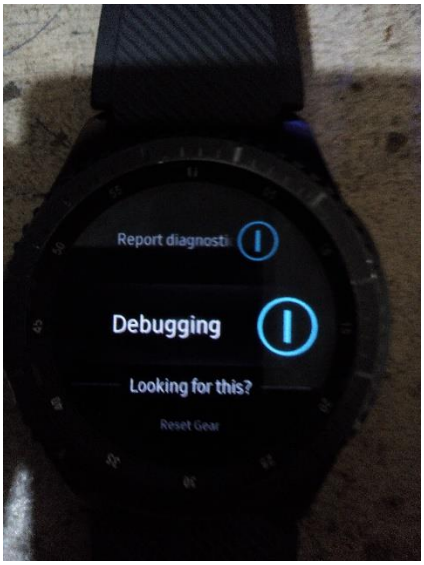
Berikut ini adalah langkah - langkah dari implementasi *Wireless Programming System* sehingga dapat dilakukannya proses pendeployan aplikasi pada *Wearable Samsung Smartwatch Gear S3* :

5.3.1. Proses mengaktifkan terlebih dahulu *Mode Debugging* pada *Wearable Samsung Smartwatch Gear S3* dengan cara klik *Setting/About*

Gear/Debugging, seperti yang ditunjukkan oleh Gambar 5.13 dan Gambar 5.14 berikut :

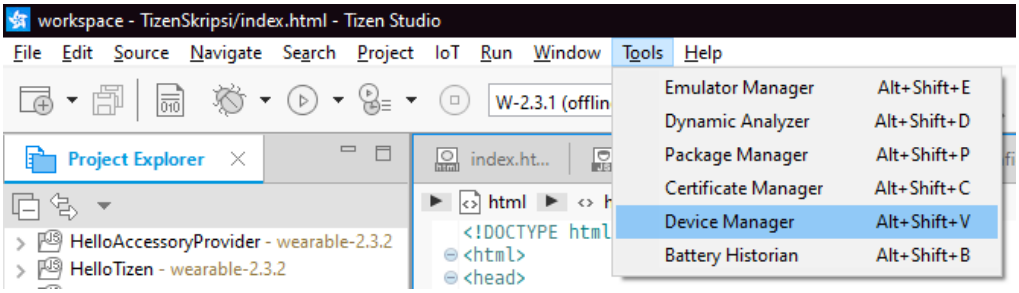


Gambar 5. 13 Sebelum diklik / Debug Off



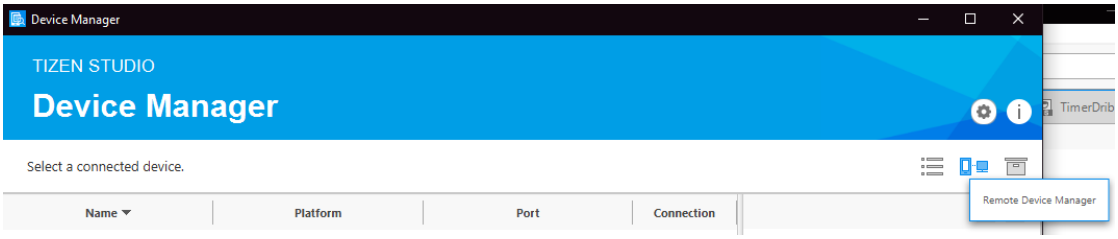
Gambar 5. 14 Setelah diklik / Debug On

5.3.2. Kemudian pada *Tizen SDK* klik pada *tab* bagian *Tools* dan klik pada bagian *Device Manager*, seperti yang ditunjukkan oleh Gambar 5.15 berikut:



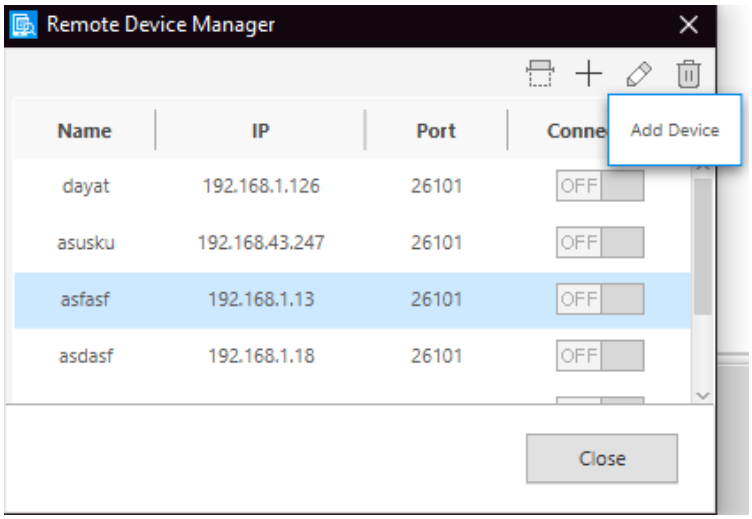
Gambar 5. 15 Menmbuka Device Manager

5.3.3. Setelah keluar sebuah *Figure: Device Manager* kemudian klik pada bagian *Remote Device Manager*, seperti yang ditunjukkan oleh Gambar 5.16 berikut :



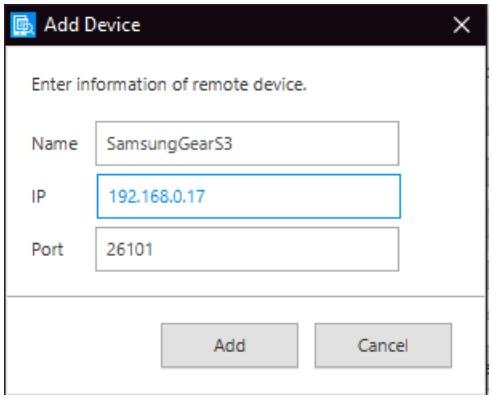
Gambar 5. 16 Device Manager

5.3.4. Setelah keluar sebuah *Figure: Remote Device Manager* kemudian klik "+" yang dimana sebagai menambahkan *Device*, seperti yang ditunjukkan oleh Gambar 5.17 berikut :

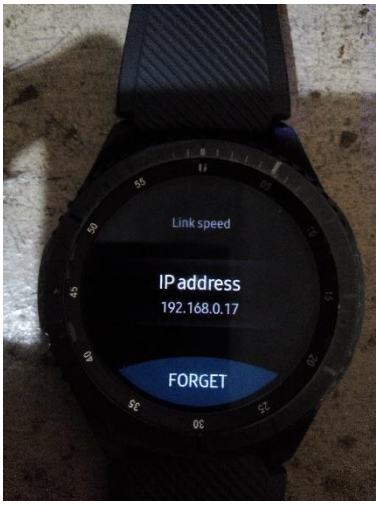


Gambar 5. 17 Remote Device Manager

5.3.5. Setelah keluar *Figure: Add Device* kemudian isikan Nama *Device* dan *IP address* yang terhubung oleh *device* atau dengan kata lain menyamakan *IP Address* yang berada pada *device seperti* Gambar 5.19 dengan yang berada pada *Add Device* seperti Gambar 5.18, setelah itu klik tombol *Add* pada *Figure: Add Device* untuk menambahkan *Device* yang akan diremote oleh *Device Manager* sendiri.

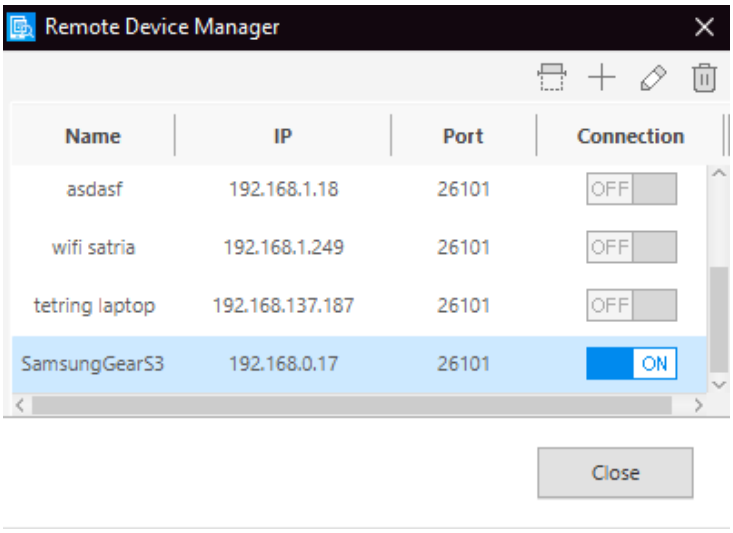


Gambar 5. 18 Add Device



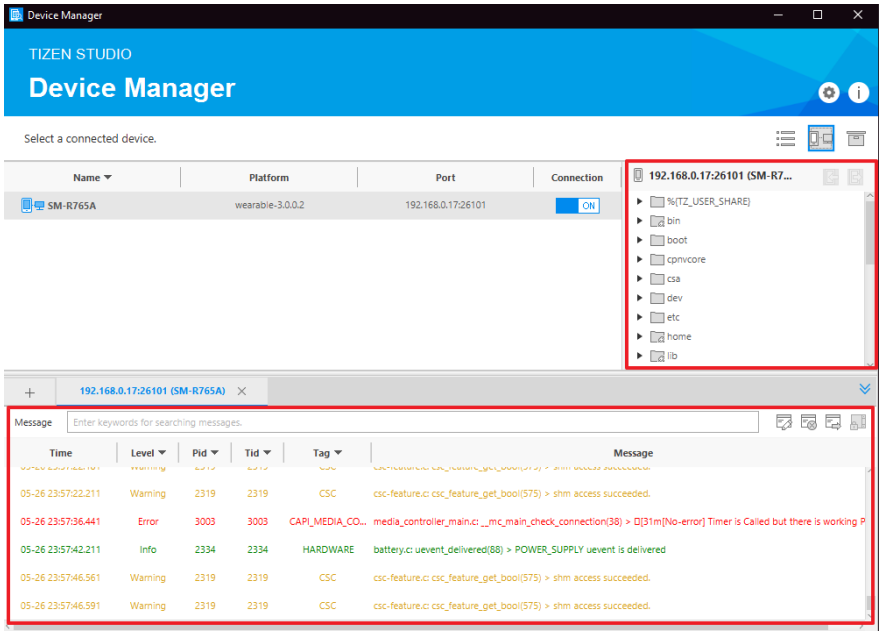
Gambar 5. 19 Device

5.3.6. Setelah melakukan klik tombol *Add* kemudian *Device* yang baru ditambahkan akan muncul pada *Figure: Remote Device Manager* dan koneksikan *SamsungGearS3* dengan *Device Manager Tizen SDK*, seperti yang ditunjukkan oleh Gambar 5.20 berikut :



Gambar 5. 20 Menkoneksikan Ip Address dengan Device

5.3.7. Apabila *SamsungGearS3* berhasil terkoneksi dalam 1 jaringan maka akan muncul beberapa folder dan beberapa pesan yang membuktikan bahwa *SamsungGearS3* berhasil terkoneksi dengan baik, seperti yang ditunjukkan oleh Gambar 5.21 berikut :



Gambar 5. 21 Berhasil terkoneksikan

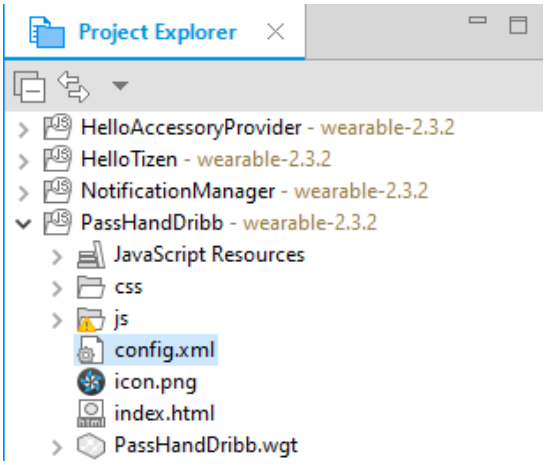
5.4. Implementasi *Run Deploy Apps*

Tahap *Run Deploy Apps* merupakan proses *deploy* pada bagian *Backend* dan bagian *Frontend*, dengan kata lain penulis disini memerlukan 2 aplikasi yang memiliki fungsi dan tujuan berbeda. Dikarenakan penulis belum bisa menyatukan 2 halaman yang disebabkan oleh sedikitnya referensi *Tizen SDK* sendiri dalam menyatukan bagian *Frontend* dan bagian *Backend*-nya.

5.4.1. Bagian Backend (PassHandDribb)

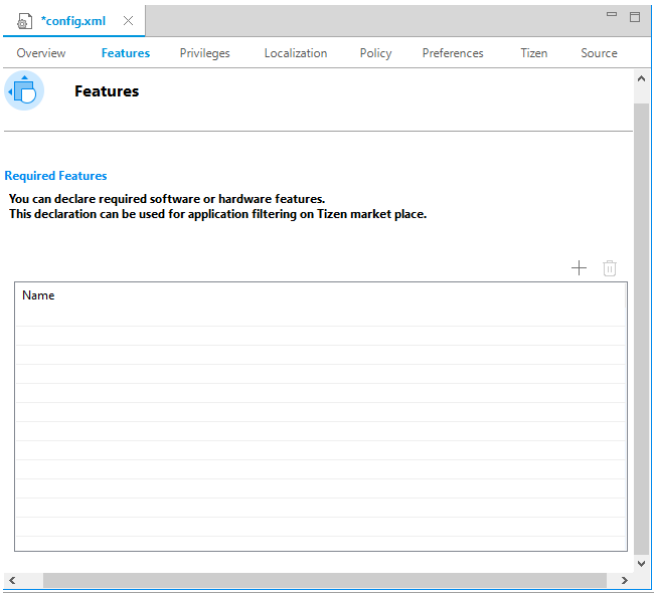
Pada bagian *Backend*, penulis memberi nama dengan *PassHandDribb* yang dimana dapat tersambung dengan *Firebase* untuk proses penyimpanan hasil kedua sensor dalam bentuk sumbu X, Y, Z. Sebelum proses *Run Deploy Apps*, terlebih dahulu menambahkan beberapa fitur penting kedalam *project* bagian *Backend* dengan nama *PassHandDribb* dengan cara:

a. Klik 2 kali pada bagian *config.xml*, seperti yang ditunjukkan oleh Gambar 5.22 berikut :



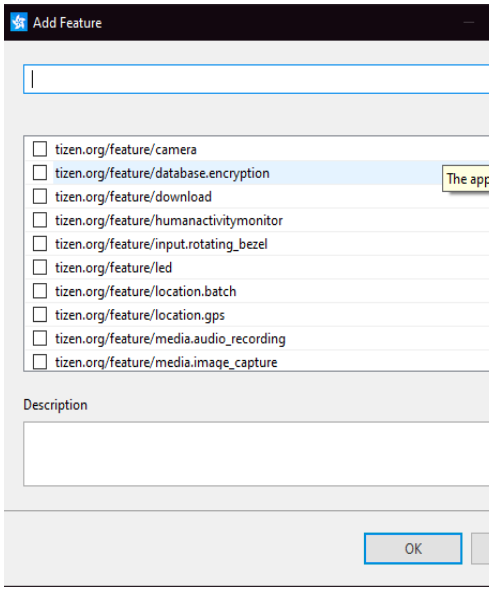
Gambar 5. 22 config.xml pada Backend

b. Kemudian klik pada bagian *Features* dan klik tanda "+" yang berada pada atas *table*, seperti yang ditunjukkan oleh Gambar 5.23 berikut :

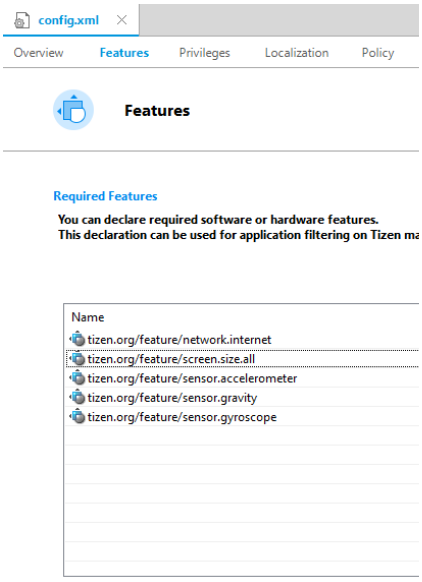


Gambar 5. 23 Tab Features pada config.xml

c. Kemudian memilih beberapa fitur yang digunakan dalam proses bagian *Backend* seperti *fitur screen.size.all*, *sensor.gravity*, *sensor.gyroscope*, *network.internet*, *dan sensor.accelerometer* dengan cara klik OK dibawah *table Description*, seperti yang ditunjukkan oleh Gambar 5.24 dan Gambar 5.25 berikut :



Gambar 5. 24 Add Features



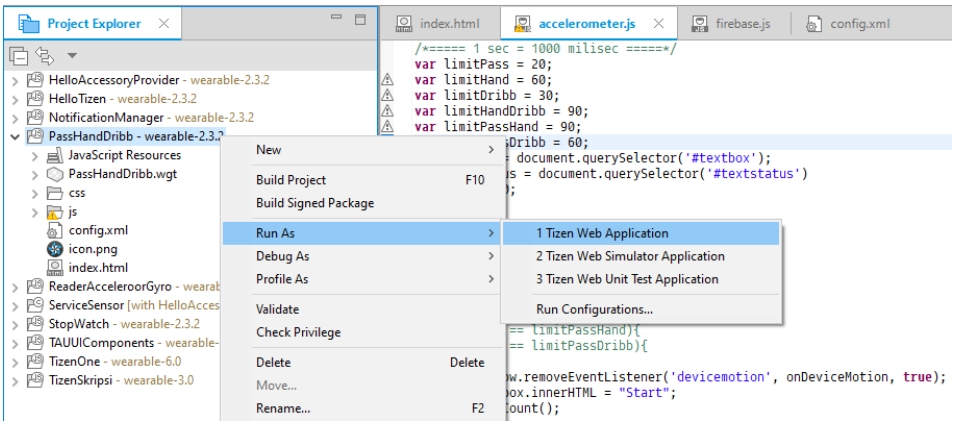
Gambar 5. 25 Beberapa features yang telah ditambahkan

Pada gambar diatas menunjukkan beberapa fitur yang digunakan dalam *project* bagian *Backend* dengan nama *PassHandDribb*. Adapun fungsi dari beberapa fitur tersebut antara lain :

- Fitur *screen.size.all* : untuk mendukung semua kemungkinan ukuran layar pada penelitian pengembangan ini dan penelitian selanjutnya yang dilakukan oleh mitra penulis dan semua kemungkinan untuk setiap layar.
- Fitur *sensor.gravity* : untuk mendukung akses sensor *Acceleration Gravity* yang dimana mencakup *sensor Gyroscope dan sensor Accelerometer*-nya sendiri.
- Fitur *sensor.gyroscope* : untuk mendukung akses *sensor Gyroscope*-nya pada bagian *Backend.*
- Fitur *network.internet* : untuk mendukung akses internet, pada saat proses *wireless programming system.*
- Fitur *sensor.accelerometer* : untuk mendukung akses *sensor Accelerometer*-nya pada bagian *Backend.*

Apabila tidak melakukan konfigurasi fitur pada *config.xml*, maka *sensor Acceleration Gravity* yang mencakup *Accelerometer* dan *Gyroscope* tidak akan bisa berjalan. Oleh sebab itu sebelum melakukan *Run Deploy Apss* pada bagian *Backend* harus terlebih dahulu melakukan konfigurasi fitur pada *config.xml* agar sensor *Acceleration Gravity* yang mencakup *sensor Accelerometer dan Gyroscope* bisa berjalan dengan baik dan bisa membaca sumbu X, Y, dan Z.

Proses *Run Deploy Apps*-nya pada bagian *Backend* dengan klik kanan pada *PassHandDribb* dan pilih *Run As* dan pilih bagian *1 Tizen Web Application*, seperti yang ditunjukkan oleh Gambar 5.26 berikut :



Gambar 5. 26 Run Deploy Apps bagian Backend

Kemudian *source code* pada bagian *Backend* yang berfungsi untuk menampilkan atau melakukan pendeteksian sumbu X, Y, Z pada *user interface*-nya, seperti yang ditunjukkan oleh Tabel 5.1 berikut. Yang dimana pada baris ke 2 sampai baris ke 4 digunakan sebagai pendeklarasian hasil sensor yang akan dikirim pada saat merubah akselerasi yang akan ditambahkan, baris ke 6 sampai baris ke 8 digunakan sebagai pendeklarasian aktivitas, baris ke 10 digunakan sebagai pengiriman hasil sensor ke database yang menggunakan *firebase console* dan baris ke 12 sampai baris ke 16 digunakan sebagai menampilkan hasil sensor pada *user interface wearable Samsung Smartwatch Gear S3*

Tabel 5. 1 Pendeteksian Sumbu X, Y, Z

No.	Source Code
1	**function** onDeviceMotion(event){
2	**var** x = event.acceleration.x;
3	**var** y = event.acceleration.y;
4	**var** z = event.acceleration.z;
5	
6	**var** hand = ("Passing Ball");
7	// **var** hand = ("Handling Ball");
8	// **var** hand = ("Dribbling Ball");
9	
10	send_dbPass(x, y, z);
11	
12	textstatus.innerHTML = " " + "x = " + x.toFixed(4) +
13	" " + "y = " + y.toFixed(4) +
14	" " + "z = " + z.toFixed(4) +
15	" " + "Aktivitas = " + hand ;
16	}

Adapun *source code* pada bagian *Backend* yang berfungsi untuk mengirimkan hasil pendeteksian sumbu X, Y, Z kedalam *firebase console* dan juga sudah terintegrasi oleh *Realtime Database* pada *Firebase Console*, seperti yang ditunjukkan oleh Tabel 5.2 berikut. Yang dimana pada baris ke 2 digunakan sebagai menampilkan hasil sensor pada Log firebase console, baris ke 4 sampai baris ke 6 menambahkan ke daftar data dalam database dengan nama *Passing Ball / Handling Ball / Dribbling Ball*, dan baris ke 8 sampai baris ke 10 digunakan sebagai pengaturan koma, yang dimana ditentukan 4 angka dibelakang koma.

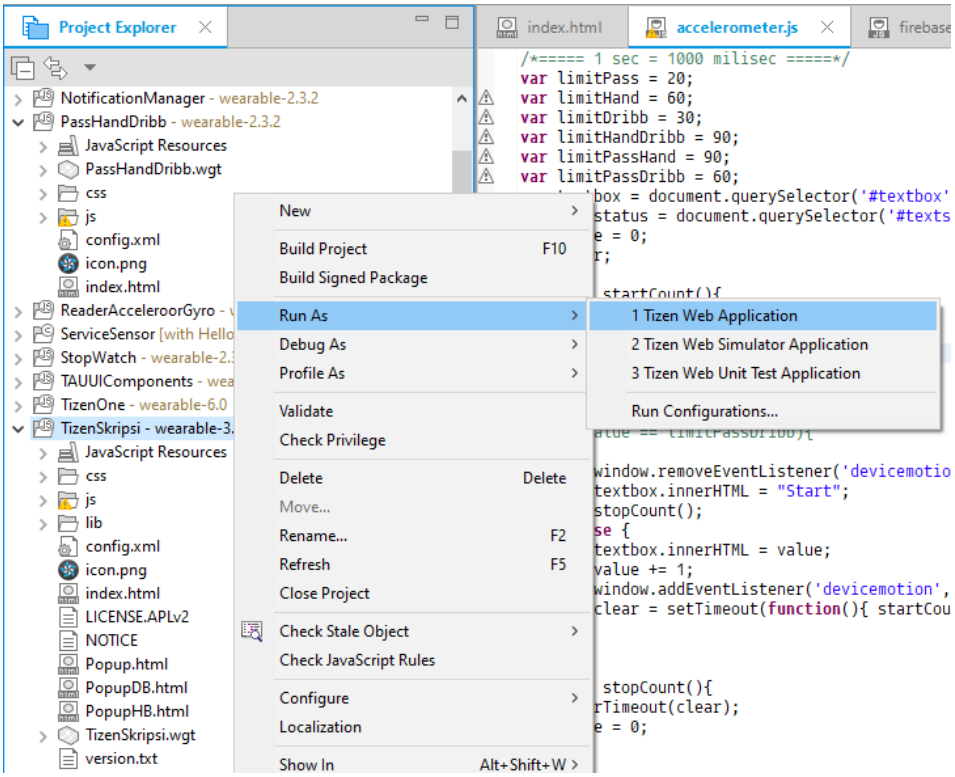
Tabel 5. 2 Pengiriman ke Firebase

No.	Source Code
1	**function** send_dbPass(x,y,z){
2	console.log(" x = " + x + " y = " + y + " z = " + z);
3	
4	firebase.database().ref('Passing Ball').push().set({
5	// firebase.database().ref('Handling Ball').push().set({
6	// firebase.database().ref('Dribbling Ball').push().set({
7	
8	x: x.toFixed(4),
9	y: y.toFixed(4),
10	z: z.toFixed(4)
11	});;
12	}

5.4.2. Bagian Frontend (TizenSkripsi)

Pada bagian *Frontend*, penulis memberi nama dengan *TizenSkripsi* yang dimana sebagai fitur *Timer* yang bertujuan untuk mencocokan standar waktu minimal latihan bola basket dengan aktivitas *Passing Ball, Handling Ball, dan Dribbling Ball*. Pada bagian *Frontend* ini melakukan pencocokannya secara terpisah dengan bagian *Backend* dikarenakan penulis belum bisa melakukan penggabungan antara bagian *Backend* dengan bagian *Frontend* pada *Tizen SDK* sendiri, dikarenakan minimnya referensi dan minimnya sumber dokumentasi *Tizen SDK*.

Perhitungan waktu untuk mencocokan standar waktu minimal latihan bola basket secara berkelanjutan, yang dimana setelah melakukan pencocokan aktivitas *Passing Ball* sebesar 20 detik maka selanjutnya mencocokan aktivitas *Handling Ball* sebesar 60 detik, kemudian yang terakhir mencocokan aktivitas *Dribbling Ball* sebesar 30 detik. Proses *Run Deploy Apps*-nya pada bagian *Frontend* dengan klik kanan pada *TizenSkripsi* dan pilih *Run As* dan pilih bagian *1 Tizen Web Application*, seperti yang ditunjukkan oleh Gambar 5.27 berikut



Gambar 5. 27 Run Deploy Apps bagian Frontend

Kemudian *source code* pada bagian *Frontend* yang berfungsi sebagai perhitungan waktu apabila detik sudah menunjukkan angka 20 (00:20) maka *Timer* akan otomatis berhenti. Dikarenakan standar waktu minimal latihan bola basket dengan aktivitas *Passing Ball* sebesar 20 detik, seperti yang ditunjukkan oleh Tabel 5.3 berikut. Adapun fungsi dari beberapa baris source code dalam *function TimerCycle()* tersebut antara lain :

- Baris ke 6 digunakan sebagai penambahan 1 detik pada setiap *variable sec.*
- Baris ke 7 sampai baris ke 10 digunakan apabila *variable sec* menunjukan angka 60 / sama dengan 60 detik maka *variable min* menambahkan 1 menit,
- Baris 21 sampai baris 24 digunakan apabila *variable sec* menunjukkan angka 10 / sama dengan 10 detik maka otomatis *vibration* menyala yang berfungsi sebagai penanda apabila standar minimal waktu latihan aktivitas *Passing Ball* kurang 10 detik lagi,
- Baris 26 sampai baris 30 digunakan apabila *variable sec* menunjukkan angka 20 / sama dengan 20 detik maka otomatis berhenti dan berganti halaman menuju *Popup.html*,
- Baris ke 36 digunakan sebagai menampilkan *variable min dan sec* pada bagian *frontend*, dan
- Angka 1000 pada baris ke 38 yang dimaksud ialah kecepatan *millisecond* yang digunakan, jadi 1000 *millisecond* sama dengan 1 *second*/detik.

Tabel 5. 3 Source Code Passing Ball

No.	Source Code
1	`function timerCycle() {`
2	`    if (stoptime == false) {`
3	`    sec = parseInt(sec);`
4	`    min = parseInt(min);`
5	
6	`    sec = sec + 1;`
7	`    if (sec == 60) {`
8	`        min = min + 1;`
9	`        sec = 0;`
10	`    }`
11	
12	`    if (min == 60) {`
13	`        min = 0;`
14	`        sec = 0;`
15	`    }`
16	
17	`    if (sec < 10 || sec == 00) {`
18	`        sec = '0' + sec;`
19	`    }`
20	
21	`    if (sec == 10){`
22	`        patternVibrationPB();`
23	`        stoptime = false;`
24	`    }`
25	
26	`    if (sec == 20) {`
27	`        sec = '' + sec;`
28	`        stoptime = true;`
29	`        location.replace("Popup.html");`
30	`    }`
31	
32	`    if (min < 10 || min == 0) {`
33	`        min = '0' + min;`
34	`    }`
35	
36	`    timer.innerHTML = min + ':' + sec;`
37	
38	`    setTimeout("timerCycle()", 1000);`
39	`    }`
40	`}`

Kemudian *source code* pada bagian *Frontend* yang berfungsi sebagai perhitungan waktu apabila menit sudah menunjukkan angka 1 dan detik sudah menunjukkan angka 20 (01:20) maka *Timer* akan otomatis berhenti. Dikarenakan standar waktu minimal latihan bola basket dengan aktivitas *Handling Ball* sebesar 60 detik, seperti yang ditunjukkan oleh Tabel 5.4 berikut. Adapun fungsi dari beberapa baris source code dalam *function TimerCycleHB()* tersebut antara lain :

- Baris ke 6 digunakan sebagai penambahan 1 detik pada setiap *variable secHB,*
- Baris ke 7 sampai baris ke 10 digunakan apabila *variable secHB* menunjukan angka 60 / sama dengan 60 detik maka *variable minHB* menambahkan 1 menit,
- Baris 21 sampai baris 24 digunakan apabila *variable secHB* menunjukkan angka 10 / sama dengan 10 detik dan *variable minHB* menunjukkan angka 1 / sama dengan menit ke 1 maka otomatis *vibration* menyala yang berfungsi sebagai penanda apabila standar minimal waktu latihan aktivitas *Handling Ball* kurang 10 detik lagi,
- Baris 26 sampai baris 30 digunakan apabila *variable secHB* menunjukkan angka 20 / sama dengan 20 detik dan *variable minHB* menunjukkan angka 1 / sama dengan menit ke 1 maka otomatis berhenti dan berganti halaman menuju *PopupHB.html*,
- Baris ke 36 digunakan sebagai menampilkan *variable minHB dan secHB* pada bagian *frontend*, dan
- Angka 1000 pada baris ke 38 yang dimaksud ialah kecepatan *millisecond* yang digunakan dalam perhitungan detik, jadi 1000 *millisecond* sama dengan 1 *second*/detik.

Tabel 5. 4 Source Code Handling Ball

No.	Source Code
1	`function timerCycleHB() {`
2	`    if (stoptimeHB == false) {`
3	`    secHB = parseInt(secHB);`
4	`    minHB = parseInt(minHB);`
5	
6	`    secHB = secHB + 1;`
7	`    if (secHB == 60) {`
8	`        minHB = minHB + 1;`
9	`        secHB = 0;`
10	`    }`
11	
12	`    if (minHB == 60) {`
13	`        minHB = 0;`
14	`        secHB = 0;`
15	`    }`
16	
17	`    if (secHB < 10 || secHB == 00) {`
18	`        secHB = '0' + secHB;`
19	`    }`
20	
21	`    if (secHB == 10){`
22	`        patternVibrationHB();`
23	`        stoptimeHB = false;`
24	`    }`
25	
26	`    if (secHB == 20) {`
27	`        stoptimeHB = true;`
28	`        secHB = '' + secHB;`
29	`        location.replace("PopupHB.html");`
30	`    }`
31	
32	`    if (minHB < 10 || minHB == 0) {`
33	`        minHB = '0' + minHB;`
34	`    }`
35	
36	`    timerHB.innerHTML = minHB + ':' + secHB;`
37	
38	`    setTimeout("timerCycleHB()", 1000);`
39	`    }`
40	`}`

Kemudian *source code* pada bagian *Frontend* yang berfungsi sebagai perhitungan waktu apabila menit tetap menunjukkan angka 1 dan detik sudah menunjukkan angka 50 (01:50) maka *Timer* akan otomatis berhenti. Dikarenakan standar waktu minimal latihan bola basket dengan aktivitas *Dribbling Ball* sebesar 30 detik, seperti yang ditunjukkan oleh Tabel 5.5 berikut. Adapun fungsi dari beberapa baris source code dalam *function TimerCycleDB()* tersebut antara lain :

- Baris ke 6 digunakan sebagai penambahan 1 detik pada setiap *variable secDB,*
- Baris ke 7 sampai baris ke 10 digunakan apabila *variable secDB* menunjukan angka 60 / sama dengan 60 detik maka *variable minDB* menambahkan 1 menit,
- Baris 21 sampai baris 24 digunakan apabila *variable secDB* menunjukkan angka 40 / sama dengan 40 detik dan *variable minDB* menunjukkan angka 1 / sama dengan menit ke 1 maka otomatis *vibration* menyala yang berfungsi sebagai penanda apabila standar minimal waktu latihan aktivitas *Dribbling Ball* kurang 10 detik lagi,
- Baris 26 sampai baris 30 digunakan apabila *variable secDB* menunjukkan angka 50 / sama dengan 50 detik dan *variable minDB* menunjukkan angka 1 / sama dengan menit ke 1 maka otomatis berhenti dan berganti halaman menuju *PopupDB.html*,
- Baris ke 36 digunakan sebagai menampilkan *variable min dan secDB* pada bagian *frontend*, dan
- Angka 1000 pada baris ke 38 yang dimaksud ialah kecepatan *millisecond* yang digunakan dalam perhitungan detik, jadi 1000 *millisecond* sama dengan 1 *second*/detik.

Tabel 5. 5 Source Code Dribbling Ball

No.	Source Code
1	`function timerCycleDB() {`
2	`    if (stoptimeDB == false) {`
3	`    secDB = parseInt(secDB);`
4	`    min = parseInt(min);`
5	
6	`    secDB = secDB + 1;`
7	`    if (secDB == 60) {`
8	`        min = min + 1;`
9	`        secDB = 0;`
10	`    }`
11	
12	`    if (min == 60) {`
13	`        min = 0;`
14	`        secDB = 0;`
15	`    }`
16	
17	`    if (secDB < 10 || secDB == 00) {`
18	`        secDB = '' + secDB;`
19	`    }`
20	
21	`    if (secDB == 40){`
22	`        patternVibrationDB();`
23	`        stoptimeDB = false;`
24	`    }`
25	
26	`    if (secDB == 50) {`
27	`        stoptimeDB = true;`
28	`        secDB = '' + secDB;`
29	`        location.replace("PopupDB.html");`
30	`    }`
31	
32	`    if (min < 10 || min == 0) {`
33	
34	`    }    min = '0' + min;`
35	
36	`    timerDB.innerHTML = min + ':' + secDB;`
37	
38	`    setTimeout("timerCycleDB()", 1000);`
39	`    }`
40	`}`

5.5. Implementasi Pengambilan dan Penyimpanan Data

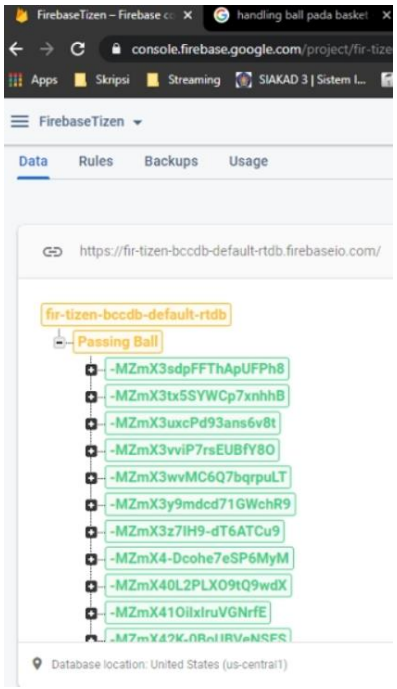
Implementasi pengambilan data sumbu X, Y, Z dilakukan pada saat proses *Backend* berlangsung yang dimana atlet/pemain bola basket melakukan aktivitas *Passing Ball* seperti Gambar 5.28, *Handling Ball* seperti Gambar 5.29, *dan Dribbling Ball* seperti Gambar 5.30 dengan memakai *Wearable Device Samsung Smartwatch Gear S3* pada pergelangan tangannya. Tampak seperti dibawah ini.

Gambar 5. 28 Aktivitas Passing Ball

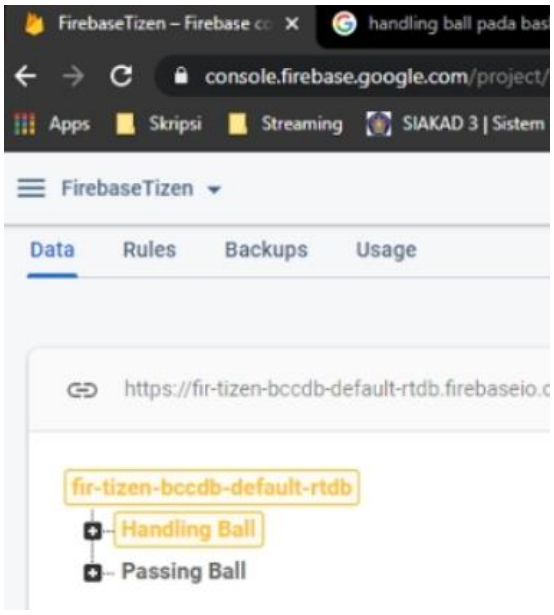
Gambar 5. 29 Aktivitas Handling Ball

Gambar 5. 30 Aktivitas Dribbling Ball

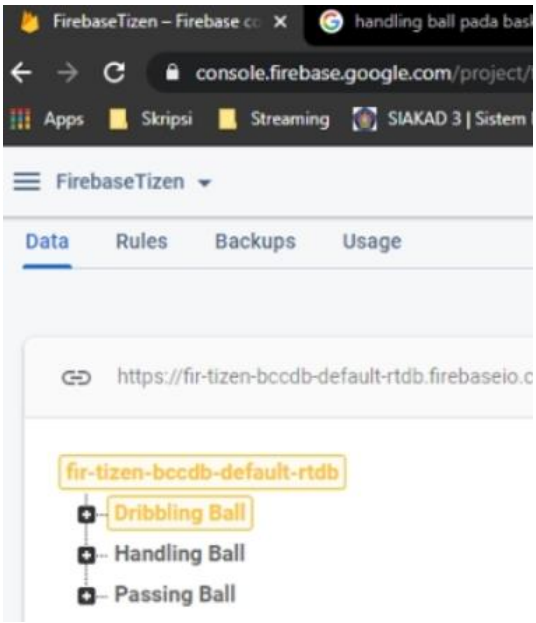
Pada saat melakukan aktivitas, proses pengambilan pada *Firebase Console* dilakukan secara *Realtime Database*. Jadi apabila atlet/pemain melakukan aktivitas maka *Firebase*-nya juga mengambil data sumbu X, Y, Z secara langsung, akan tetapi atlet/pemain berhenti melakukan aktivitas maka *Firebase*-nya juga berhenti. Berikut adalah hasil dari pengambilan data sumbu X, Y, Z dengan melakukan aktivitas *Passing Ball, Handling Ball, dan Dribbling Ball*. Seperti yang ditunjukkan oleh Gambar 5.31, Gambar 5.32, dan Gambar 5.33 berikut.



Gambar 5. 31 Database Passing Ball



Gambar 5. 32 Database Handling Ball



Gambar 5. 33 Database Dribbling Ball