LAPORAN AKHIR MOBILE APP PROGRAMMING

Anggota Kelompok StroPis:

Mishel Milen - 00000055747

Nayasha Clarisa Dwisutrisna - 00000056883

Ayu Febriana Lingga - 00000057105

Gilbert Henry Munayang - 00000066484

KELAS IF 570-D

SEMESTER GENAP 2022-2023

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK DAN INFORMATIKA

UNIVERSITAS MULTIMEDIA NUSANTARA

TANGERANG

2023

Pengembangan Aplikasi Mobile Let's Train Buddy: Sebagai Aplikasi untuk Panduan Berolahraga

1. PENDAHULUAN

1.1 Latar Belakang Masalah

Dalam kehidupan sehari hari, semakin banyak orang yang mulai sadar terhadap pentingnya kondisi kesehatan dan kebugaran tubuh yang harus dijaga dengan melalui berolahraga. Namun, terdapat beberapa orang yang mengalami kesulitan dalam merencanakan dan melaksanakan rutinitas untuk berolahraga yang efektif dan sesuai dengan kebutuhannya. Kendala lainnya seperti terbatasnya fasilitas/alat untuk berolahraga, biaya, dan waktu. Oleh karena itu, tanpa memerlukan pengeluaran untuk alat yang mahal, mereka dapat berolahraga sendiri di rumah dengan beberapa alternatif *home workout*. Hal ini dapat dilakukan dengan adanya bantuan panduan dari instruktur olahraga. Maka, diperlukan sebuah aplikasi *mobile* yang dapat memberikan panduan olahraga personal dan dapat diakses dengan mudah oleh siapa dan dimana saja.

1.2 Penjelasan Data/Fakta Terkait Masalah

Menurut penelitian Solikin dan Muradi pada jurnal penelitiannya, kurangnya aktivitas fisik dapat meningkatkan kadar kolesterol dalam tubuh yang menjadi faktor risiko penyakit jantung dan pembuluh darah [1]. Hal ini membuktikan bahwa pentingnya aktivitas fisik selain aktivitas utama dalam kehidupan sehari hari seperti berolahraga olahraga ringan. Dengan melakukan olahraga walau dengan frekuensi dan intensitas yang tidak terlalu tinggi, setidaknya telah memiliki kesadaran betapa pentingnya kesehatan tubuh untuk selalu dirawat dan dijaga. Namun, halangan yang terkadang dihadapi oleh individu yang ingin berolahraga adalah gerakan olahraga seperti apa yang harus dilakukannya jika tidak mengikuti suatu komunitas ataupun kelompok berolahraga. Sehingga, sangat dibutuhkan panduan sederhana bagi mereka yang ingin berolahraga dari rumah saja secara mandiri. Kemudian, menurut data BPS, warga Indonesia yang telah berumur lebih dari 5 tahun telah menggunakan telepon seluler sebagai alat komunikasinya [2]. Melihat dari penggunaan ponsel yang tinggi dan terus meningkat, membuktikan bahwa hampir seluruh individu telah memiliki dan menggunakan *handphone* untuk mengakses informasi apapun. Sehingga, aplikasi *mobile* yang dibutuhkan dalam menjadi panduan berolahraga bagi *user* yang membutuhkannya akan

sangat didukung dengan keberadaan penggunaan *handphone* yang sudah tidak langka lagi.

1.3 Kajian Literatur Penelitian Terdahulu

Terdapat beberapa aplikasi *home workout mobile* yang dapat ditemukan di berbagai platform unduh seperti *Google Play Store* maupun *App Store*. *Workout* adalah latihan *calisthenics* atau *bodyweight training* yang hanya menggunakan berat tubuh kita sendiri untuk melatih otot [3]. Maka dari itu, *workout* sangat cocok digunakan untuk orang-orang yang ingin berolahraga secara mandiri di rumah karena tidak memerlukan alat-alat berat seperti yang ada di gym atau tempat-tempat fitnes. Latihan workout dapat memberikan banyak dampak baik pada kebugaran fisik dan juga bentuk tubuh seseorang. Orang dengan proporsi lemak dan otot yang sesuai dapat membuat seseorang lebih bersemangat dan sehat.

Pada penelitian yang dilakukan oleh Setyo Wibowo dengan judul "Perancangan Aplikasi Latihan Fitness untuk Pemula Berbasis Multimedia" di tahun 2016, Setyo merancang sebuah aplikasi berbasis website untuk latihan fitnes bagi para pemula. Latihan-latihan yang diberikan berupa gerakan basic atau dasar karena ditujukan bagi orang yang benar-benar baru mulai olahraga atau ingin membangun massa otot. Telah dilakukan pengujian keandalan sistem dan seberapa manfaat perancangan aplikasi web latihan fitness ini. Hasilnya sebesar 74% responden menjawab bahwa aplikasi web ini bermanfaat [4]. Penelitian ini membuat aplikasi berbasis website, namun jika user sedang berada dalam kondisi diluar jaringan atau offline, maka user tidak dapat menggunakan aplikasi ini. Untuk itu, penulis membuat sebuah aplikasi mobile, yaitu aplikasi workout bernama Let's Train Buddy yang berbasis android dengan memanfaatkan *smartphone* sebagai media untuk membantu memudahkan setiap orang yang ingin berolahraga secara mandiri dimana dan kapan pun.

1.4 Analisis Kondisi Saat Ini

Disisi lain, terdapat beberapa hal yang berkaitan dengan kondisi saat ini berdasarkan permasalahan dan kebutuhan orang-orang yang ingin berolahraga secara mandiri di rumah mereka terhadap aplikasi pemandu workout. Berikut adalah beberapa kondisinya:

A. Banyak orang yang mencari panduan untuk berolahraga melalui internet, tetapi merasa kesulitan dalam menemukan sumber yang terpercaya dan sesuai dengan kondisi maupun kebutuhan mereka

B. Terdapat individu yang merasa kesulitan untuk memotivasi dirinya dalam berolahraga secara disiplin/teratur
C. Mulai bertambahnya kesadaran untuk berolahraga, sehingga banyak orang mencari solusi untuk menjaga kesehatan melalui aktivitas olahraga
D. Keterbatasan pengetahuan mengenai gerakan-gerakan olahraga yang efektif untuk kondisinya
E. Kemajuan teknologi dan penggunaan perangkat mobile telah menjadi bagian dalam kehidupan yang terus semakin maju. Sehingga, membuka peluang untuk memberikan solusi yang inovatif dalam dunia kesehatan.

1.5 Analisis Komparatif Aplikasi Serupa

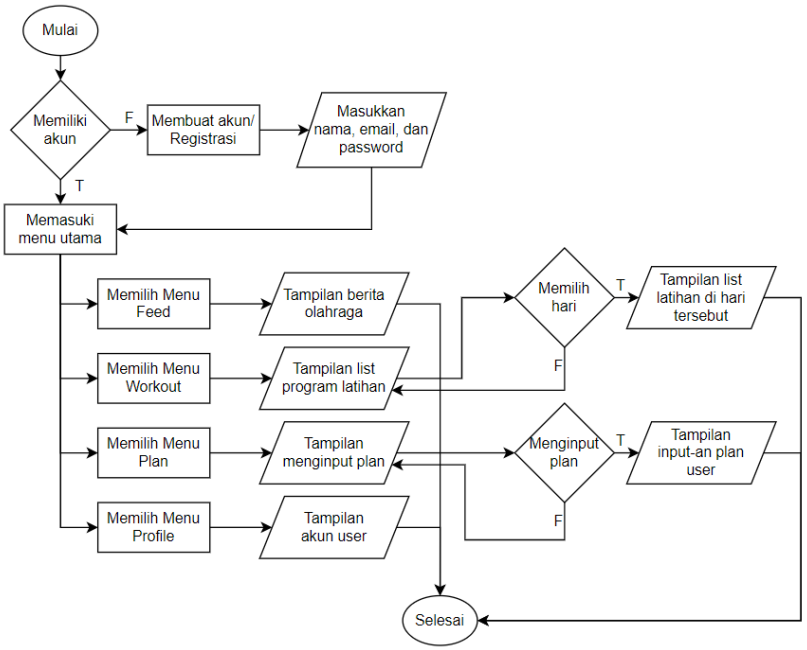
Pada aplikasi serupa yang sudah ada di pasaran, memiliki kelebihan dan kekurangannya masing masing. Namun, hal yang mungkin dialami pada aplikasi yang sudah ada di pasaran dan cukup dijadikan sebagai analisa perbandingan seperti:

A. Aplikasi yang terlalu fokus pada nutrisi daripada aspek kebugaran fisiknya
B. Panduan olahraga kurang sesuai personalisasi user
C. Beberapa konten premium sehingga butuh biaya langganan
D. Terdapat panduan yang kurang sesuai untuk pemula
E. Fitur-fitur yang memerlukan biaya tambahan premium
F. UI yang memerlukan user untuk beradaptasi kembali
G. Kurang fleksibel dalam plan aktivitas user

2. PERANCANGAN

2.1 Alur Aplikasi

Alur aplikasi ini dibuat menggunakan *software draw.io* dalam bentuk *flowchart* sehingga memudahkan penulis dan pembaca untuk lebih memahami cara kerja aplikasi Let’s Train Buddy. Berikut adalah *flowchart* alur aplikasi Let’s Train Buddy:



2.2 Target Pengguna dan Role User

A. Pengguna awam yang berminat untuk berolahraga. Individu yang baru mengenal dunia kebugaran dan tertarik untuk memulai program latihan. Mereka dapat menggunakan aplikasi untuk mendapatkan panduan latihan, merencanakan rutinitas, update informasi berita terkini seputar kesehatan.

B. Pengguna yang sudah berpengalaman. Orang orang yang telah berpengalaman dalam berolahraga dan mencari tantangan baru atau variasi dalam latihan berolahraga. Mereka dapat menggunakan aplikasi untuk mendapatkan rutinitas latihan yang lebih beragam sebagai ide latihan baru, dan memantau sejauh mana kekuatannya dengan variasi latihan.

C. Pengguna yang memiliki tujuan spesifik seperti menurunkan berat badan, membentuk tubuh, dsb. Individu dengan tujuan kesehatan atau kebugaran tertentu yang ingin merancang program latihan olahraga sesuai dengan kekuatan dan target mereka. Mereka dapat menggunakan aplikasi yang dapat membantu untuk menjadi perencanaan dan mengikuti program latihan sesuai yang dituju.

3. IMPLEMENTASI

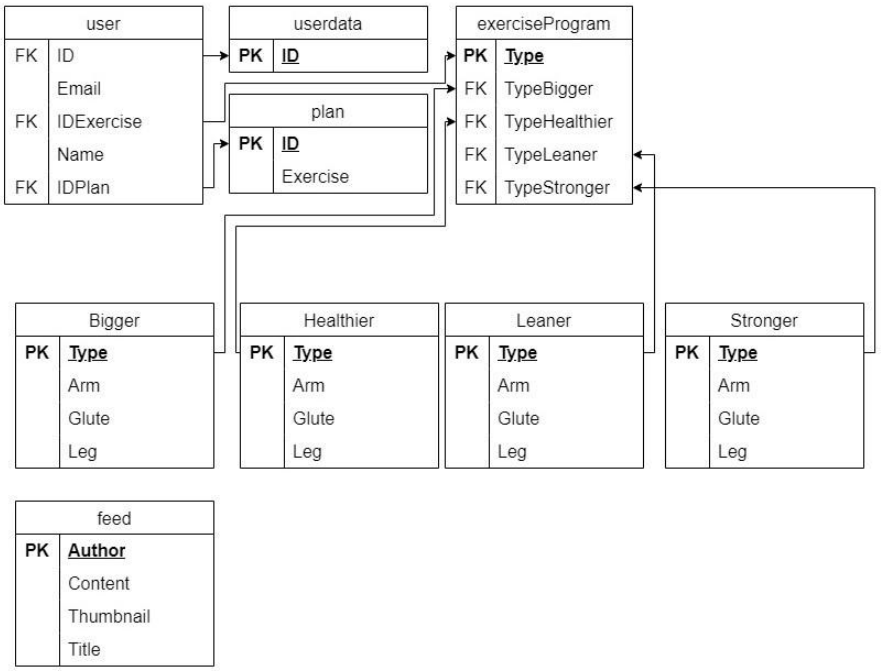
3.1 Arsitektur Sistem Client-Server

Aplikasi "Let's Train Buddy" menggunakan jenis arsitektur *client-server* terpusat. Pengguna atau *user* akan terhubung ke server pusat melalui aplikasi klien pada perangkat *mobile* mereka. Server bertanggung jawab atas seluruh logika bisnis, manajemen pengguna, dan penyimpanan data. Aplikasi klien pada perangkat *mobile* bertugas untuk

menyediakan antarmuka pengguna dan mengirimkan permintaan ke server untuk mendapatkan program latihan serta menerima respons, pembaruan, atau rekomendasi. Dalam manfaat skalabilitas, server pusat dapat diperkuat untuk menangani peningkatan jumlah pengguna. Komunikasi antara klien dan server dienkripsi untuk melindungi data pengguna. Otorisasi dan otentikasi digunakan untuk mengamankan akses dan aktivitas pengguna.

3.2 Rancangan Skema Database

Rancangan skema database pada entitas utama memiliki dua tujuan utama yaitu pengguna untuk menyimpan informasi pribadi pengguna seperti nama, *email*, dan *password* pada *database*. Pada aplikasi ini, penulis menggunakan *Firebase* sebagai *database* dari aplikasi Let's Train Buddy. Kemudian, program latihannya yang berisi jenis latihan dan panduan latihan latihan gerakannya. Relasi antar tabel pengguna dan program latihan saling berkaitan satu sama lain untuk menjalankan program latihan. Berikut merupakan skema *database* dari Let's Train Buddy.



3.3 Kode Implementasi Fitur Natif

Kode implementasi fitur natif mengacu pada bagian kode dalam pengembangan perangkat lunak yang ditulis menggunakan bahasa pemrograman spesifik untuk platform tertentu. Dalam konteks pengembangan aplikasi mobile, fitur natif merujuk pada fungsionalitas yang diimplementasikan dengan menggunakan bahasa dan alat

pengembangan khusus seperti iOS atau Android. Pada aplikasi Let's Train Buddy, penulis menggunakan alat pengembangan Android dengan bahasa pemrograman Kotlin untuk mengimplementasikan fitur-fitur pada aplikasi. Namun, penulis juga menggunakan ekstensi '.xml' dalam mendesain dan membuat tampilan fitur-fitur untuk aplikasi. Berikut beberapa potongan kode pada pembuatan aplikasi Let's Train Buddy yang menggunakan bahasa pemrograman kotlin:

A. DayListFragment.kt:

```
private fun fetchAndDisplayExerciseMoves(chosenProgram: String, dayName: String, sportName: String) {
    // ...
    dbReferences_exercise.addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(dataSnapshot: DataSnapshot) {
            // ...
            for (exerciseSnapshot in exerciseListSnapshot.children) {
                // ...
                val exerciseMove = ExerciseMove(exerciseName, description, reps, set)
                exerciseMovesList.add(exerciseMove)
            }
            val adapter = ExerciseListAdapter(requireContext(), exerciseMovesList)
            moveListView.adapter = adapter
        }

        override fun onCancelled(databaseError: DatabaseError) {
            // ...
        }
    })
}
```

Penjelasan:

-Fungsi **fetchAndDisplayExerciseMoves** digunakan untuk mengambil data gerakan latihan dari Firebase dan menampilkannya dalam ListView yang tampilannya di desain di fragment_day_list.xml.

-Fungsi ini akan mengeksekusi query ke Firebase untuk mendapatkan data gerakan latihan berdasarkan program latihan, hari, dan jenis olahraga yang dipilih.

-Fungsi ini akan mengonversi data dari Firebase ke objek '**ExerciseMove**'.

-Fungsi ini akan menyiapkan adapter ('**ExerciseListAdapter**') untuk '**ListView**' dan menetapkannya ke '**moveListView**'.

B. DetailFragment.kt:

```
fun setSportData(id: Int){
    when(id){
        R.id.sport1 -> {
            sportTitle?.text = getString(R.string.sport1_title)
            sportDesc?.text = getString(R.string.sport1_desc)
        }
        R.id.sport2 -> {
            sportTitle?.text = getString(R.string.sport2_title)
            sportDesc?.text = getString(R.string.sport2_desc)
        }
        R.id.sport3-> {
            sportTitle?.text = getString(R.string.sport3_title)
            sportDesc?.text = getString(R.string.sport3_desc)
        }
    }
}
```

Penjelasan:

-Fungsi **setSportData** mengatur data olahraga (judul dan deskripsi) berdasarkan ID olahraga yang diteruskan.

-Fungsi ini menggunakan struktur '***when***' untuk menentukan data olahraga yang sesuai dengan ID yang diberikan. Sehingga jika *user* memilih olahraga *Leg Exercise*, maka judul olahraga (*Leg Workout*) dan deskripsi tentang *Leg Exercise* akan muncul.

C. LandingActivity.kt:

```
public override fun onStart() {
    super.onStart()
    val currentUser = mAuth.currentUser
    if (currentUser != null) {
        val intent = Intent(this, MainActivity::class.java)
        startActivity(intent)
        finish()
    }
}
```

Penjelasan:

-Fungsi **onStart** dipanggil ketika aktivitas dimulai.

-Fugsi ini memeriksa apakah *user* sudah memiliki akun dan sudah log in sebelumnya atau belum. Jika sudah, maka langsung diarahkan ke MainActivity.kt dan aktivitas LandingActivity.kt akan diakhiri.

D. PlanFragment.kt:

```
private fun setupPlanListView() {
    val adapter = CustomAdapter(requireContext(), planList)
    planListView.adapter = adapter

    dbReferences.child("plan").addValueEventListener(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            val newPlanList = mutableListOf<String>()
            for (postSnapshot in snapshot.children) {
                val planName = postSnapshot.getValue(String::class.java)
                planName?.let {
                    newPlanList.add(it)
                }
            }
            adapter.updatePlanList(newPlanList)
        }

        override fun onCancelled(error: DatabaseError) {
            Log.e("FirebaseError", "Error fetching plans: ${error.message}") // Logging Firebase error
            // Handle error
        }
    })
}
```

Penjelasan:

-Fungsi **setupPlanListView** digunakan untuk mengatur '**ListView**' dan mengambil data dari Firebase Realtime Database menggunakan '**addValueEventListener**'.

-Data plan/rencana latihan diambil dari child '**plan**' dalam database dan diperbarui menggunakan adapter.

E. SportFragment.kt:

```
override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)
    mAuth = FirebaseAuth.getInstance()
    val userId = mAuth.currentUser?.uid
    dbReferences = FirebaseDatabase.getInstance().reference.child("userdata").child(userId ?: "")

    // ...

    dbReferences.addValueEventListener(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            // ...
        }

        override fun onCancelled(error: DatabaseError) {
            Log.e("FirebaseError", "Error fetching email: ${error.message}") // Logging Firebase error
            // Handle error
        }
    })

    val addButton = view.findViewById<Button>(R.id.update_button)
    addButton.setOnClickListener {
        showPreferencesInputDialogFragment()
    }
}
```

Penjelasan:

-Fungsi **onViewCreated** dipanggil setelah tampilan fragment telah dibuat.

-Fungsi ini menginisialisasi elemen-elemen UI seperti tombol '**addButton**' dan elemen-elemen '**TextView**'.

-Fungsi ini menambahkan '**ValueEventListener**' untuk mengambil data *user* dari Firebase dan memperbarui tampilan sesuai dengan data terbaru.

3.4 Kode Proses CRUD

Kode proses CRUD mengacu pada bagian kode dalam pengembangan perangkat lunak yang ditulis menggunakan bahasa pemrograman spesifik untuk platform tertentu. Dalam hal pengembangan aplikasi, CRUD memiliki fungsi untuk melakukan membuat data, membaca data, memperbarui data dan menghapus data dari **database**. Berikut merupakan potongan kode dari proses CRUD pada aplikasi Let's Train Buddy.

A. RegisterActivity.kt

- *Create*

```
mAuth.createUserWithEmailAndPassword(emailInput, passwordInput)
    .addOnCompleteListener() { task ->
        if (task.isSuccessful) {
            val currentUser = mAuth.currentUser
            val uid = currentUser?.uid

            uid?.let {userId ->
                val userRef = dbreferences.child(userId)
                val userData = HashMap<String, Any>()
                userData["name"] = nameInput
                userData["email"] = emailInput
                userData["exerciseProgram"] = "none"

                userRef.setValue(userData)
                    .addOnSuccessListener { it: Void!
                        val intent = Intent( packageContext: this, LandingActivity::class.java)
                        startActivity(intent)
                    }
                    .addOnFailureListener { exception ->
                        Log.e(TAG,  msg: "Failed to register: $exception")
                        Toast.makeText( context: this,  text: "Failed to register.", Toast.LENGTH_SHORT).show()
                    } ^let

            }
        } else {
            Toast.makeText(
                 context: this,  text: "Authentication failed.", Toast.LENGTH_SHORT
            ).show()
        }
    }
```

Penjelasan :

1. **mAuth.createUserWithEmailAndPassword** digunakan untuk membuat akun baru di *firebase* menggunakan *email* dan *password* yang diberikan.
2. Apabila pembuatan akun berhasil, maka akan mendapatkan ID *user* dari pengguna tersebut. Kemudian, data pengguna yaitu *name*, *email* dan *exerciseProgram* akan disimpan dalam *hashmap* **userData**.
3. **userRef.setValue(userData)** digunakan untuk menyimpan data ke *database*, yaitu *firebase*.

- *Read*

```
val currentUser = mAuth.currentUser
if (currentUser != null) {
    val intent = Intent( packageContext: this, MainActivity::class.java)
    startActivity(intent)
    finish()
}
```

Penjelasan :

1. **currentUser** akan membaca *database* apakah pengguna sudah terdaftar atau belum. Jika sudah terdaftar, maka akan memulai halaman MainActivity.

B. PlanFragment.kt

- *Create*

```
private fun showPlanInputDialogFragment() {
    val planInputDialogFragment = PlanInputDialogFragment()
    planInputDialogFragment.setOnSubmitListener(object : PlanInputDialogFragment.InputFormListener {
        override fun onSubmitClicked(inputText: String) {
        }
    })
    planInputDialogFragment.show(childFragmentManager, tag: "PlanInputDialogFragment")
}
```

Penjelasan :

1. Menggunakan metode **showPlanInputDialogFragment** untuk menampilkan dialog agar pengguna dapat memasukkan *plan* kegiatan yang baru.

- *Read*

```
dbReferences.child( pathString: "plan").addValueEventListener(object : ValueEventListener {
    override fun onDataChange(snapshot: DataSnapshot) {
        val newPlanList = mutableListOf<String>()
        for (postSnapshot in snapshot.children) {
            val planName = postSnapshot.getValue(String::class.java)
            planName?.let { it: String
                newPlanList.add(it)
            }
        }
        adapter.updatePlanList(newPlanList)
    }

    override fun onCancelled(error: DatabaseError) {
        Log.e( tag: "FirebaseError", msg: "Error fetching plans: ${error.message}") // Logging Firebase error
        // Handle error
    }
})
```

Penjelasan :

1. Melakukan pemantauan perubahan data pada *node* **plan** di *database* yang digunakan.
2. Apabila data tersedia, kode akan mengambil isi dari setiap plan dan menambahkannya ke **newPlanList**.
3. Daftar plan yang diperbarui akan diteruskan ke adapter untuk kemudian ditampilkan di **ListView.**

- *Update*

```
fun updatePlanList(newPlanList: MutableList<String>) {
    planList.clear()
    planList.addAll(newPlanList)
    notifyDataSetChanged()
}
```

Penjelasan :

1. **planList.clear** memiliki fungsi untuk menghapus semua data plan yang ada dari daftar.
2. **planList.addAll** memiliki fungsi untuk menambahkan semua data plan baru ke daftar.
3. **notifyDataSetChanged** memiliki fungsi untuk mengabarkan adapter bahwa data sudah diperbarui.

- *Delete*

```
deleteButton.setOnClickListener { it: View!
    val planToDelete = planList[position]
    val userId = mAuth.currentUser?.uid
    userId?.let { uid ->
        dbReferences.child( pathString: "plan").orderByValue().equalTo(planToDelete)
            .addListenerForSingleValueEvent(object : ValueEventListener {
                override fun onDataChange(snapshot: DataSnapshot) {
                    for (childSnapshot in snapshot.children) {
                        childSnapshot.ref.removeValue()
                            .addOnSuccessListener { it: Void!
                                Log.d( tag: "Firebase",  msg: "Successfully deleted plan: $planToDelete")
                            }
                            .addOnFailureListener { exception ->
                                Log.e( tag: "Firebase",  msg: "Failed to delete plan: $exception")
                            }
                    }
                }

                override fun onCancelled(error: DatabaseError) {
                    Log.e( tag: "FirebaseError",  msg: "Error deleting plan: ${error.message}")
                }
            })
    }
}
```

Penjelasan :

1. **deleteButton.setOnClickListener** memiliki arti fungsi apabila tombol tersebut di klik maka akan menghapus data.
2. **planToDelete** memiliki fungsi untuk mengambil *plan* yang ingin dihapus dari *list plan*.
3. **childSnapshot.ref.removeValue** memiliki fungsi untuk menghapus *node plan* yang telah ditemukan.

C. SportFragment.kt

- *Create*

```
private fun showPreferencesInputDialogFragment() {
    val preferencesDialog = PreferenceInputDialogFragment()
    preferencesDialog.setOnSubmitListener ( object :
        PreferenceInputDialogFragment.InputFormListener {
        override fun onSubmitClicked(inputText: String) {
            Log.d( tag: "PreferenceInputDialog",  msg: "Input: $inputText")
        }
    } )
    preferencesDialog.show(childFragmentManager,  tag: "PreferenceInputDialogFragment")
}
```

Penjelasan :

1. **showPreferencesInputDialogFragment** memiliki fungsi untuk menampilkan dialog atau *popup* yang digunakan untuk memasukkan preferensi *exercise* yang baru.
2. **onSubmitClicked** akan berjalan ketika tombol submit di klik.

- *Read*

```
dbReferences.addValueEventListener(object : ValueEventListener {
    override fun onDataChange(snapshot: DataSnapshot) {
        val email = snapshot.child( path: "email").getValue(String::class.java) ?: ""
        val name = snapshot.child( path: "name").getValue(String::class.java) ?: ""
        val exerciseProgram = snapshot.child( path: "exerciseProgram").getValue(String::class.java) ?: ""

        content_description.text = "You workout because you want to get $exerciseProgram"
        user_email.text = email
        user_name.text = name
    }

    override fun onCancelled(error: DatabaseError) {
        Log.e( tag: "FirebaseError",  msg: "Error fetching email: ${error.message}") // Logging Firebase error
        // Handle error
    }
})
```

Penjelasan :

1. **addValueEventListener()** digunakan untuk memantau perubahan data pada **userdata** di *firebase*.

2. **onDataChange** akan dipanggil ketika terjadi perubahan data.
3. **onDataChange** akan mengambil *email*, *name*, dan *exerciseProgram* dari *firebase*.

D. PlanInputDialogFragment.kt

- *Create*

```
val inputText = editText.text.toString().trim()
val currentUser = mAuth.currentUser
val uid = currentUser?.uid

uid?.let { userId ->
    val userRef = dbreferences.child(userId).child( pathString: "plan")

    userRef.addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            val count = snapshot.childrenCount.toInt()

            val newPlanRef = userRef.push()
            newPlanRef.setValue(inputText)
                .addOnSuccessListener { it: Void!
                    Log.d(ContentValues.TAG, msg: "Successfully added plan")
                }
                .addOnFailureListener { exception ->
                    Log.e(ContentValues.TAG, msg: "Failed to add plan: $exception")
                }
        }
        override fun onCancelled(error: DatabaseError) {
            Log.e(ContentValues.TAG, msg: "Cancelled operation: $error")
        }
})
}
```

Penjelasan :

1. **userRef.push()** akan membuat sebuah *node* baru dengan ID unik dibawah *node* **plan**.
2. **newPlanRef.setValue(inputText)** memiliki fungsi untuk menyimpan *input* dari pengguna mengenai *plan* baru ke *node* tersebut.

3.5 Proses Setup untuk Replicate Project

Proses setup untuk *replicate project* adalah serangkaian langkah-langkah atau tindakan yang harus dilakukan untuk menyiapkan lingkungan pengembangan atau produksi sehingga proyek dapat dijalankan atau dikembangkan. Berikut langkah-langkah atau proses *setup* yang harus dilakukan untuk dapat mereplikasi proyek aplikasi mobile berbasis android, seperti aplikasi Let's Train Buddy:

1. Instalasi Perangkat Lunak dan Alat Pengembangan. Pastikan untuk menginstal Android Studio yakni IDE resmi untuk pengembangan Android dan pastikan juga telah menginstal SDK yang diperlukan.
2. *Extract file* aplikasi Let's Train Buddy ke dalam folder 'AndroidStudioProjects'.
3. Buka aplikasi Android Studio, kemudian file 'File' → 'Open' dan pilih file Let's Train Buddy.
4. Pastikan file konfigurasi proyek seperti 'gradle.properties' atau 'local.properties' sudah sesuai dengan konfigurasi lokal di laptop/komputer yang digunakan.
5. Jalankan atau sinkronisasikan Gradle dengan mengklik 'File' → 'Sync Project With Gradle Files'.
6. Konfigurasikan emulator Android atau hubungkan perangkat fisik seperti *handphone* untuk melakukan uji coba atau run proyek.
7. Siapkan *Firebase*, kemudian koneksikan proyek dengan *realtime database*.
8. Siapkan *Firebase Auth* dan *Realtime Database*
9. *Import JSON Database*
10. *Run* atau jalankan aplikasi di *emulator* atau perangkat fisik (*handphone*).

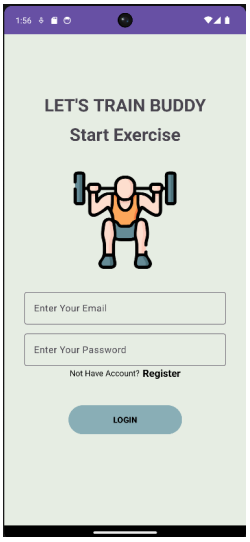
3.6 Tahapan Implementasi

A. Menganalisa kebutuhan. Melakukan identifikasi dan analisis kebutuhan pengguna serta spesifikasi teknis yang diperlukan.
B. Desain sistem. Mulai melakukan rancang bangun arsitektur, skema *database*, dan antarmuka pengguna
C. Pengembangan *Frontend*. Membuat antarmuka pengguna yang menjadi gambaran terbentuknya sebuah aplikasi Let's Train Buddy.
D. Pengembangan *backend* dan *database*. Melakukan implementasi logika server dan skema *database* yang diintegrasikan dengan *frontend* yang telah dibuat.
E. Uji sistem. Melakukan pengujian sistem secara fungsional dan kinerja.
F. Implementasi ke server produksi. Menerapkan aplikasi dengan uji coba lanjutan.
G. Peluncuran. Aplikasi mulai dirilis pada platform Google Play Store supaya dapat digunakan oleh end *user*.

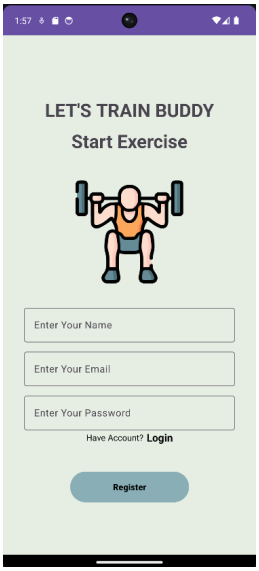
4. HASIL DAN EVALUASI

4.1 Screenshot Hasil Aplikasi

Berikut merupakan hasil dari aplikasi Let's Train Buddy yang telah dibuat.

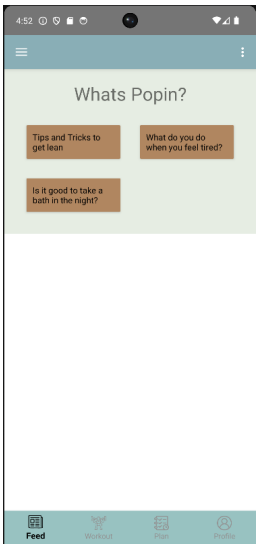


1. Pada gambar pertama, pengguna akan diarahkan untuk melakukan *login* ketika membuka aplikasi. Pengguna akan diminta untuk memasukkan *email* dan *password* yang sesuai. Apabila *email* dan *password* sudah dimasukkan, maka pengguna akan diarahkan untuk klik tombol *login*. Apabila *login* berhasil, maka akan lanjut ke tahap selanjutnya. Namun apabila gagal, maka akan muncul peringatan.

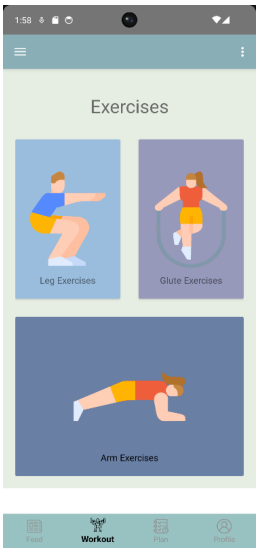


2. Pada gambar kedua, apabila pengguna belum mempunyai akun, maka akan diarahkan untuk melakukan *register* akun. Pengguna akan diminta untuk memasukkan nama,

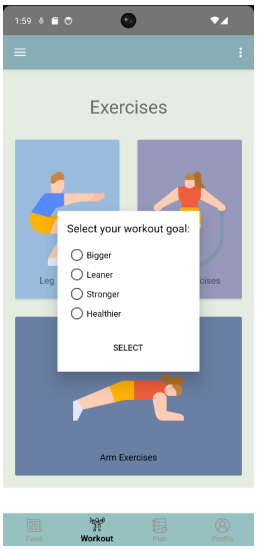
email dan *password*. Apabila semuanya sudah diisi, maka pengguna akan diarahkan untuk klik tombol *register*.



3. Pada gambar ketiga, apabila pengguna sudah melakukan *register* atau *login*, maka pengguna akan masuk ke bagian *feed*. Pada bagian *feed* terdapat berbagai macam berita mengenai kesehatan.

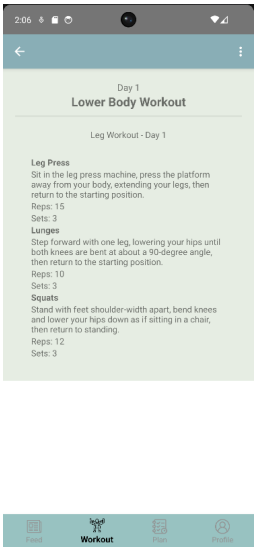


4. Pada gambar keempat, saat pengguna ke bagian *workout*, maka akan muncul tiga jenis *exercise*, yaitu *Leg Exercise*, *Glute Exercise* dan *Arm Exercise*.

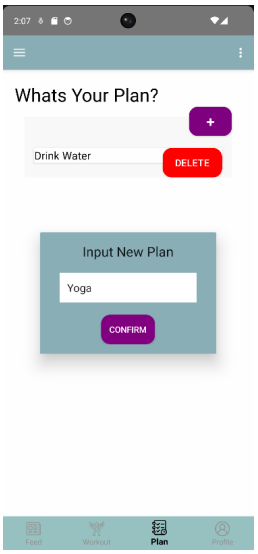


5. Pada gambar kelima, apabila pengguna klik salah satu *exercise*, maka akan muncul *pop up*. *Pop up* tersebut bertujuan mendapatkan *goal* yang ingin dicapai apabila melakukan salah satu *exercise* tersebut. Dalam hal ini, *workout goal* terdiri dari *Bigger, Leaner, Stronger* dan *Healthier*.

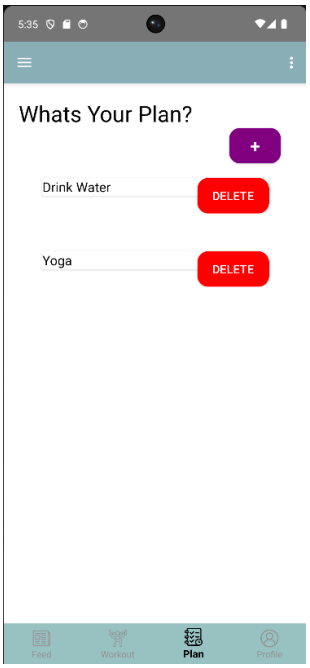
6. Pada gambar keenam, ketika *workout goal* sudah dipilih, maka akan muncul daftar hari untuk *exercise* tersebut. Dalam hal ini, daftar yang muncul mulai dari *day* 1 hingga *day* 7.



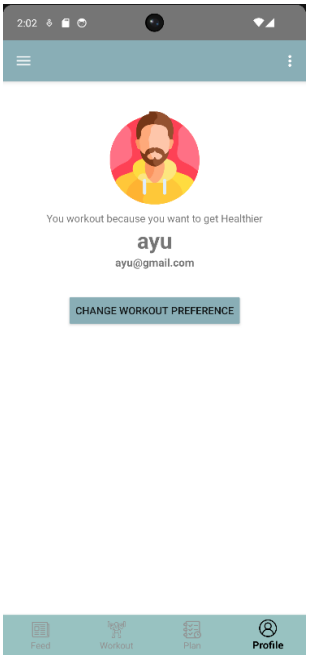
7. Pada gambar ketujuh, apabila pengguna klik salah satu hari dari yang telah dipilih pada gambar sebelumnya, maka akan muncul gerakan apa saja yang harus dilakukan untuk hari tersebut.



8. Pada gambar kedelapan, saat pengguna klik plan, pengguna dapat menambahkan *planning* kegiatan yang akan dilakukan. Pengguna akan diarahkan untuk klik tombol "+" untuk menambahkan *plan* yang akan dilakukan dan kemudian akan muncul *pop up* untuk mengisi *plan* baru. Apabila pengguna sudah mengisi *plan* baru, maka pengguna akan diarahkan untuk klik *confirm*.



9. Pada gambar kesembilan, saat pengguna sudah memasukkan *plan* yang baru, maka akan muncul tampilan seperti yang telah ditunjukkan. Pengguna dapat menambah lagi *plan* yang baru atau menghapus *plan* yang telah dikerjakan.



10. Pada gambar kesepuluh, saat pengguna klik *profile*, pengguna dapat melihat nama, *email* dan *goal* yang telah dimasukkan oleh pengguna sebelumnya. Pada halaman ini, pengguna dapat mengganti preferensi *workout* menjadi *goal* yang baru.

4.2 Skenario Pengujian Blackbox

Pengujian *Blackbox* atau *Blackbox Testing* adalah tahap pengujian sistem aplikasi dengan cara mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak [3]. Proses *Blackbox testing* yang akan penulis lakukan terhadap aplikasi Let's Train Buddy adalah dengan mengakses sistem serta mencoba satu persatu fungsi, fitur, atau menu yang terdapat di aplikasi, apakah semua fitur ini dapat berjalan dengan yang diharapkan atau tidak. Berikut adalah tabel hasil pengujian *blackbox* dari aplikasi Let's Train Buddy:

Tabel 1 *Black Box Testing*

No.	Nama Fungsi	Skenario	Status
1.	Log in	Sistem dapat melakukan autentikasi log in berdasarkan email dan password yang pernah dibuat *user* sebelumnya	Berhasil
2.	Sign up	Sistem dapat membuat akun baru	Berhasil
3.	Halaman utama (Feed)	Sistem dapat menampilkan halaman utama yang berisi berita seputar olahraga beserta navigation bar	Berhasil
4.	Menu Workout	Sistem dapat menampilkan kategori olahraga dan list olahraga setiap hari	Berhasil
5.	Menu Plan	Sistem dapat menampilkan button tambah untuk membuat plan beserta list plan yang telah dibuat *user*	Berhasil
6.	Menu Profile	Sistem dapat menampilkan informasi akun *user*	Berhasil

4.3 Dokumentasi Pengujian, Bug dan Solusi

Selama proses pengerjaan project, bug yang menjadi cukup masalah pada proses pengerjaan proyek kami yaitu pada page "Plan". Pada page ini terdapat logika untuk 'Delete', namun dalam beberapa waktu tombol tersebut masih belum dapat berfungsi. Sehingga, dengan segala percobaan trial dan error, kami melakukan Custom Adapter, sedangkan pada proses sebelumnya yang tidak kunjung berhasil kami masih menggunakan Default Adapter.

```
inner class CustomAdapter(context: Context, private val planList: MutableList<String>) : ArrayAdapter<String>(
    context,
    R.layout.item_plan_card,
    R.id.plan_item,
    planList
) {
    override fun getCount(): Int {...}

    override fun getView(position: Int, convertView: View?, parent: ViewGroup): View {
        val view = convertView ?: LayoutInflater.from(context).inflate(R.layout.item_plan_card, parent, attachToRoot: false)

        val planItem = planList[position]
        val deleteButton = view.findViewById<Button>(R.id.delete_button1)

        val planItemTextView = view.findViewById<TextView>(R.id.plan_item) // Assuming plan_item is a TextView

        planItemTextView.text = planItem // Set the plan name to the TextView


        deleteButton.setOnClickListener {...}

        // Set other views or data for the item as needed

        return view
    }

    fun updatePlanList(newPlanList: MutableList<String>) {
        planList.clear()
        planList.addAll(newPlanList)
        notifyDataSetChanged()
    }
}
```

Terdapat juga bug pada pemilihan preferensi ketika ingin memilih jenis workout. Setiap kali user ingin memilih jenis workout, pop-up akan selalu muncul untuk menanyakan preferensi. Bug ini diperbaiki dengan menambahkan properti data baru pada database user yaitu “exerciseProgram”. Ketika user telah melakukan pemilihan preferensi, maka jawaban tersebut akan disimpan pada “exerciseProgram” masing-masing user. dan pada aplikasi akan melakukan pengecekan value pada “exerciseProgram” user tersebut sebelum memasuki list workout.

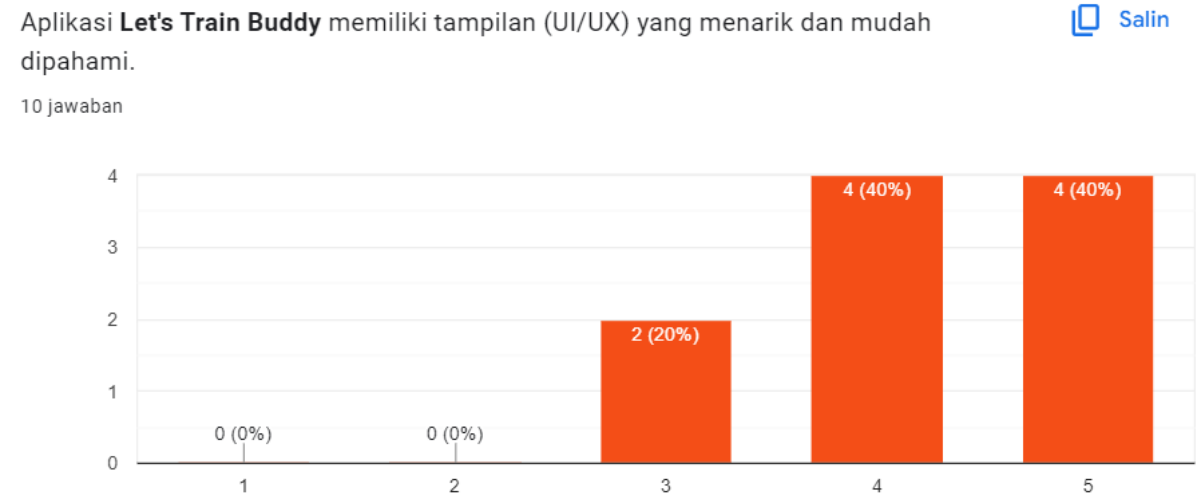
```
sport.setOnClickListener{ it: View!
    dbReferences.addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            val exerciseProgram = snapshot.getValue(String::class.java)
            if (exerciseProgram == "none") {
                showPreferencesInputDialogFragment()

            } else {
                Log.d( tag: "Firebase", msg: "Exercise program: $exerciseProgram")
                Navigation.createNavigateOnClickListener(R.id.action_workoutFragment_to_detailFragment, fragmentBundle).onClick(sport)
            }
        }

        override fun onCancelled(error: DatabaseError) {
            Log.e( tag: "FirebaseError", msg: "Error fetching exercise program: ${error.message}") // Logging Firebase error
            // Handle error
        }
    })
}
```

4.4 Evaluasi oleh Pengguna

Evaluasi yang diberikan oleh pengguna dengan melakukan survei menggunakan Google Form terhadap 10 responden, seperti:



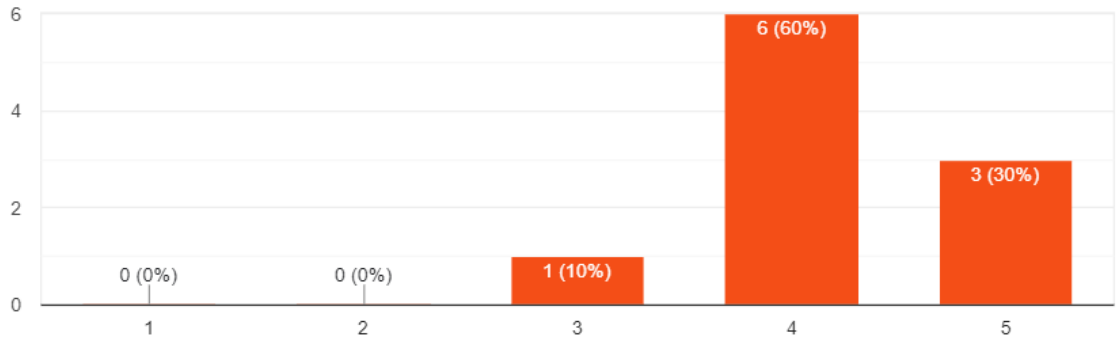
Sebanyak 80% dari responden berpendapat bahwa tampilan aplikasi menarik dan mudah untuk dipahami dalam menggunakannya. Solusi yang dapat kami berikan yaitu meningkatkan kembali desain antarmuka di lain kesempatan.

Sebanyak 70% responden berpendapat sangat setuju bahwa aplikasi ini dapat membantunya dalam aktivitas berolahraga. Solusi yang dapat kami berikan yaitu menambah variasi dari ragam latihan yang ada.

Fitur yang disediakan Aplikasi **Let's Train Buddy** sudah berfungsi dengan baik dan benar.

Salin

10 jawaban

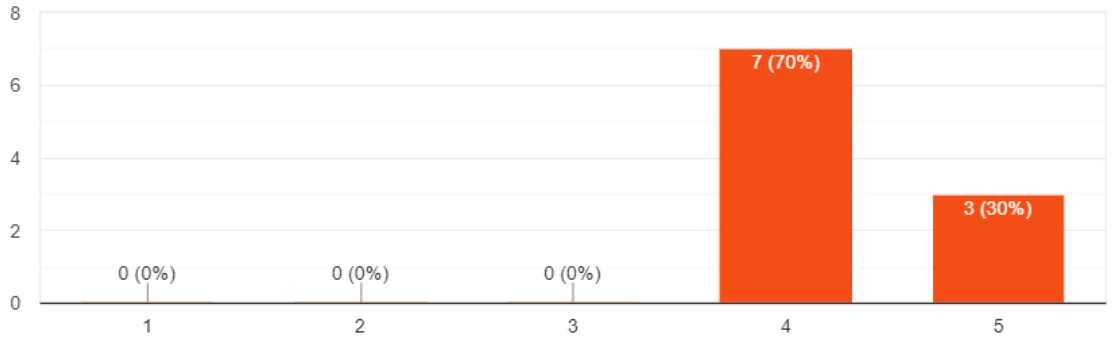


Sebanyak 90% dari responden berpendapat setuju dan sangat setuju bahwa fitur pada aplikasi sudah berfungsi dengan baik dan benar. Solusi yang dapat kami berikan yaitu meragamkan fitur fungsional untuk kebutuhan *user* kedepannya.

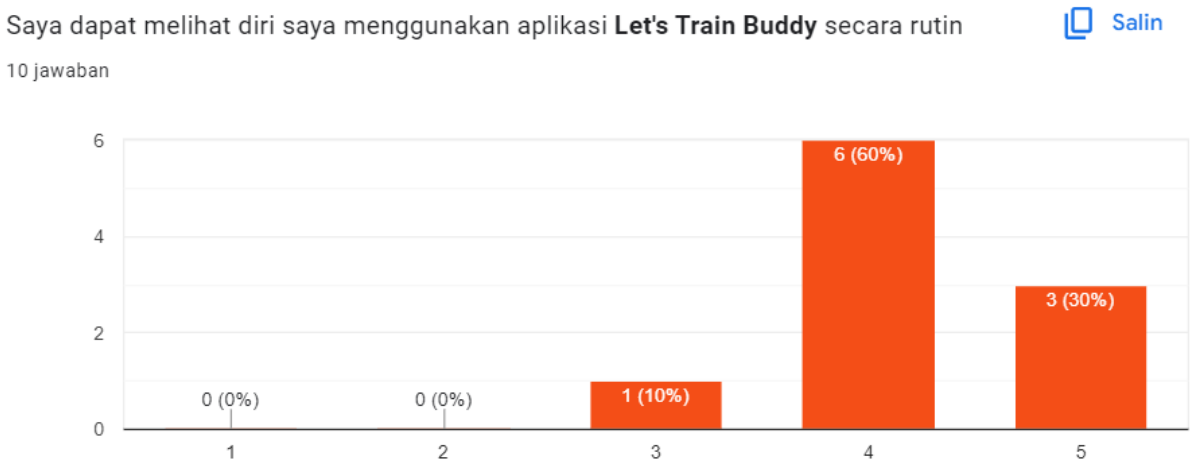
Saya merasa puas dengan responsivitas dan kecepatan aplikasi **Let's Train Buddy**.

Salin

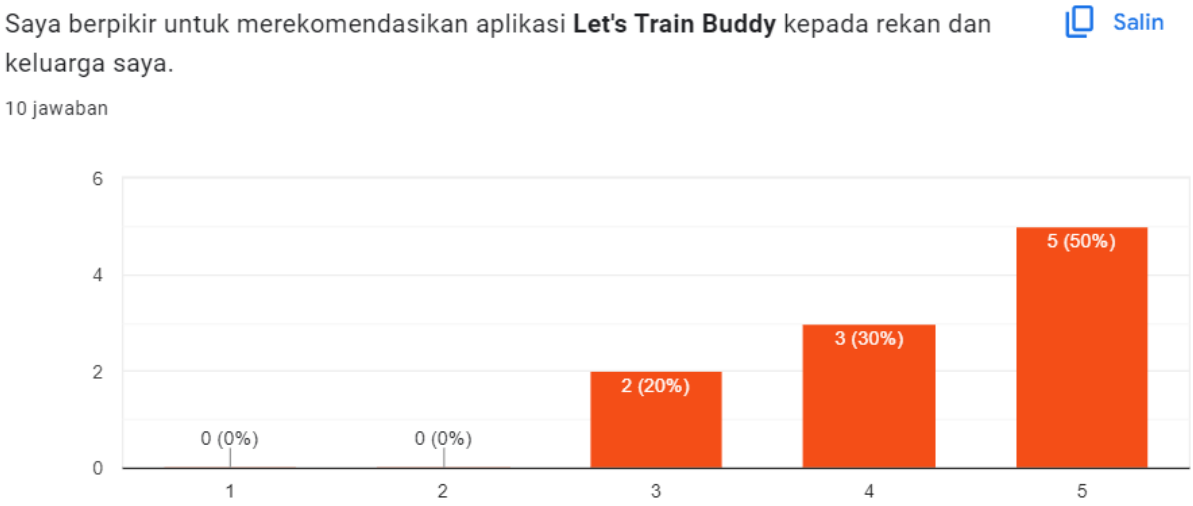
10 jawaban



Hampir seluruh responden berpendapat bahwa mereka setuju dan sangat setuju dalam tingkat kepuasan responsivitas dan kecepatan aplikasi Let’s Train Buddy. Solusi yang dapat kami berikan yaitu mempertahankan responsivitas dan kecepatan yang sudah ada terlebih dahulu karena belum terlalu menjadi masalah yang besar.



Sebanyak 90% responden berpendapat bahwa mereka setuju dan sangat setuju bahwa aplikasi ini dapat menjadi aplikasi yang digunakan secara rutin. Solusi yang dapat kami berikan yaitu terus memotivasi diri untuk hidup sehat dan bugar.



Sebanyak 80% dari responden berpendapat bahwa mereka setuju dan sangat setuju jika aplikasi ini akan mereka rekomendasikan kepada rekan dan keluarganya. Solusi yang dapat kami berikan yaitu jika *user* sudah merasa puas, dipersilahkan untuk mempromosikan ke rekan mereka supaya lebih meningkatkan kesadaran terhadap kesehatan tubuh dan aplikasinya dapat berkembang karena semakin banyaknya pengalaman dari pengguna.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

5.1.1 Ringkasan Hasil

Ringkasan hasil yang dicapai oleh aplikasi termasuk dalam aspek keberhasilan dalam memenuhi tujuan dan masalah target, seperti menilai sejauh mana aplikasi Let's Train Buddy mencapai tujuan awalnya yaitu dapat memberikan pengguna sarana untuk membuat, melacak, dan meningkatkan aktivitas latihan olahraga mereka. Harapan untuk pengguna dapat dengan mudah beradaptasi dan menggunakan aplikasi dengan lancar. Sehingga, penggunaan yang intuitif ini dapat meningkatkan kepuasan pengguna. Mengukur kualitas dan efektivitas latihan yang diberikan oleh aplikasi. Pelacakan aktivitas ini menjadi point penting dalam memberikan pemahaman mengenai perkembangan pengguna. Menilai konsistensi pengguna dengan hadirnya aplikasi ini dapat membuat mereka lebih konsisten dalam menjalankan latihan. Memperhatikan tata letak antarmuka pengguna agar mudah dipahami dan estetis. Mengevaluasi ketersediaan dan kestabilan aplikasi seperti beban yang digunakan saat aplikasi dijalankan. Hingga melihat potensi kedepannya terhadap pengembangan lanjutan fitur tambahan yang memungkinkan untuk menambah daya tarik dan fungsionalitas aplikasi.

5.1.2.Ketercapaian Tujuan

1. Membantu pengguna untuk mencapai tujuan kesehatan dan kebugaran mereka

   Aplikasi Let's Train Buddy menawarkan berbagai beberapa latihan workout yang disesuaikan dengan tujuan fitness pengguna. Selain itu, juga menyediakan fitur untuk memantau progress workout mereka dalam beraktivitas, membaca berita informasi seputar kesehatan terkini, dan *planning* terhadap aktivitas yang dilakukan. Fitur fitur ini dapat membantu pengguna untuk tetap termotivasi dan konsisten dalam menjalankan program *workout* mereka.

2. Memberikan pengalaman workout yang menyenangkan dan efektif

   Aplikasi Let's Train Buddy memiliki desain yang *user-friendly* dan mudah digunakan. Aplikasi ini juga dilengkapi berbagai macam fitur yang membuat pengalaman *workout* lebih menyenangkan. Aplikasi ini dapat disesuaikan dengan macam pilihan workout sesuai dengan minat dan kemampuan pengguna. Sehingga, mereka dapat merasa lebih nyaman dan termotivasi dalam menjalankan program workoutnya.

3. Menjadi sumber informasi dan motivasi bagi pengguna

   Aplikasi Let's Train Buddy menyediakan informasi seputar kesehatan, tips dan trik workout, serta artikel inspiratif lainnya. Informasi ini dapat membantu

pengguna untuk mendapatkan pengetahuan dan motivasi yang mereka butuhkan untuk mencapai tujuan fitness mereka.

Secara keseluruhan, aplikasi Let's Train Buddy ini merupakan aplikasi pemandu workout yang efektif dan dapat membantu pengguna untuk mencapai tujuan fitness pengguna. Aplikasi juga dirancang dengan antarmuka pengguna yang *user-friendly*, fitur dan informasi yang lengkap serta bermanfaat.

5.1.3 Kontribusi Aplikasi

Bagi Pengguna:

1. Membantu pengguna dalam mencapai tujuan fitness mereka. Aplikasi Let's Train Buddy menyediakan beberapa program *workout* yang disesuaikan dengan tujuan fitness pengguna. Dengan menggunakan aplikasi ini, pengguna dapat lebih mudah dan efektif dalam mencapai tujuan kesehatan serta kebugarannya.
2. Memberikan pengalaman workout yang menyenangkan dan efektif. Aplikasi Let's Train Buddy memiliki desain yang *user-friendly*. Selain itu, dilengkapi dengan berbagai fitur yang memberi pengalaman workout kepada pengguna menjadi lebih menyenangkan dan efektif. Sehingga, pengguna dapat menyesuaikan dengan minat dan kemampuannya serta perasaan lebih nyaman dan termotivasi dalam menjalankan programnya.
3. Menjadi sumber informasi dan motivasi bagi pengguna. Aplikasi Let's Train Buddy menyediakan berbagai informasi seputar kesehatan dan kebugaran untuk menarik minat pengguna dalam berita kesehatan. Informasi ini dapat menambah pengetahuan dan motivasi mereka dalam mencapai tujuannya.

Bagi Komunitas:

1. Mendorong budaya hidup sehat. Aplikasi ini dapat mendorong budaya hidup sehat di masyarakat. Dengan hadirnya aplikasi ini, pengguna dapat lebih mudah untuk memulai dan menjalankan program workout mereka. Sehingga, membawa dampak positif terhadap kesehatan masyarakat secara umum.
2. Menjadi sarana edukasi seputar fitness. Aplikasi Let's Train Buddy menyediakan berbagai macam fitur beserta informasi seputar fitness. Hal ini dapat membantu masyarakat untuk lebih memahami pentingnya menjaga kesehatan dan kebugaran. Sehingga, kesadaran masyarakat akan pentingnya hidup sehat dapat meningkat.

Bagi Bidang Terkait:

1. Meningkatkan industri fitness. Aplikasi workout ini dapat meningkatkan industri fitness. Aplikasi ini dapat menjadi sarana bagi pelaku industri fitness dalam menjangkau lebih banyak pengguna. Sehingga, dapat meningkatkan minat masyarakat terhadap fitness dan mendorong pertumbuhan industri fitness.
2. Menjadi sarana penelitian di bidang fitness. Data yang dikumpulkan dalam aplikasi Let's Train Buddy dapat digunakan untuk penelitian dalam bidang fitness. Data ini memberi bantuan kepada para peneliti dalam memahami perilaku pengguna dan efektivitas program workout. Sehingga. hal ini dapat mendukung pengembangan ilmu pengetahuan di bidang fitness.

5.2 Saran

5.2.1 Peningkatan Aplikasi

Saran yang dapat penulis berikan untuk peningkatan aplikasi Let’s Train Buddy di masa yang akan datang adalah sebagai berikut.

1. Mengembangkan aplikasi Let’s Train Buddy agar dapat berjalan di sistem operasi yang berbeda
2. Memperbaiki beberapa *bug* yang tidak dapat ditangani pada aplikasi Let’s Train Buddy
3. Menambah fitur yang lebih optimal sehingga dapat membantu *user* untuk melakukan *workout* secara lebih efektif

5.2.2 Strategi Implementasi

Dalam melakukan promosi aplikasi Let’s Train Buddy, diperlukan strategi yang efektif untuk menarik minat *user* agar menggunakan aplikasi Let’s Train Buddy. Strategi yang dapat digunakan untuk melakukan promosi ke khalayak umum adalah sebagai berikut.

1. Mengembangkan sosial media yang sedang diminati saat ini seperti *Instagram* atau *Tik Tok* agar dapat menarik minat dari pengguna sosial media
2. Menggunakan desain yang kekinian dengan cara yang menarik namun informatif

3. Membuat konten seperti *challenge* menggunakan aplikasi Let's Train Buddy serta memberikan *reward* kepada pemenang dari *challenge* tersebut.

5.2.3 Penelitian Lebih Lanjut

Berdasarkan pada hasil dari pembuatan aplikasi Let's Train Buddy yang telah dilakukan, beberapa rekomendasi untuk pengembangan yang terkait dengan pembuatan aplikasi Let's Train Buddy adalah sebagai berikut.

1. Melakukan integrasi dengan *artificial intelligence* dalam membuat *plan* untuk *workout* yang sesuai dengan yang diinginkan.
2. Melakukan ekspansi ke platform lain agar *user* dapat membuka Let's Train Buddy melalui *smartphone*, laptop, dan *smart* TV sehingga menjadi lebih fleksibel dan efisien.
3. Mengoptimalkan algoritma pelacakan *progress* pada setiap *user* agar dapat memberikan *feedback* yang lebih tepat dan bersifat personal kepada pengguna.

6. DAFTAR PUSTAKA

1. Solikin., Muradi. 2020. HUBUNGAN KADAR KOLESTEROL DENGAN DERAJAT HIPERTENSI PADA PASIEN HIPERTENSI DI PUSKESMAS SUNGAI JINGAH. Banjarmasin: Jurnal Keperawatan Suaka Insan.
2. Sadya, S. 2023. *Sebanyak 67,88% Penduduk RI Gunakan Telepon Genggam pada 2022*.
3. Imanulloh, H. 2017. PEMBUATAN APLIKASI WORKOUT TRAINING COACH BERBASIS ANDROID. Informatika, Universitas Amikom Yogyakarta.
4. Wibowo, S. 2016. PERANCANGAN APLIKASI LATIHAN FITNESS UNTUK PEMULA BERBASIS MULTIMEDIA. Teknik Informatika, Universitas PGRI Yogyakarta.

TIM KELOMPOK

	Mishel Milen 1. Membuat tampilan dan navigasi day 1 - 7 pada aplikasi 2. Membuat tampilan front end aplikasi 3. Membantu membuat laporan pada bagian penelitian terdahulu, alur aplikasi, arsitektur sistem, proses setup untuk replicate project, skenario pengujian blackbox, dan kode implementasi fitur natif.
	Nayasha Clarisa Dwisutrisna 1. Membangun frontend dari nol hingga terbentuk aplikasi 2. Merapikan/finalisasi frontend pada page “Plan” 3. Membuat laporan (Menjawab sebanyak 11 Point)
	Ayu Febriana Lingga 1. Menyambungkan bagian sign in dan sign up ke beranda 2. Mengupload aplikasi ke google play console 3. Membuat skema database, menulis bagian kode proses CRUD dan saran pada laporan
	Gilbert Henry Munayang 1. Mengimplementasikan firebase auth dan penggunaan database pada aplikasi (Backend) 2. Membangun populasi data pada database seperti kumpulan data untuk fitur gerakan-gerakan pada setiap jenis workout 3. Merancang Fitur CRUD seperti plan, workout preference, dan READ pada fitur pemilihan jenis workout