KERJA PRAKTIK

Mobile Apps Development: Starred, PlanetPal, FLog, PukaBoo, Robust, Echoes

Apple Developer Academy @UC

DIREKTORAT PENDIDIKAN INSITUT TEKNOLOGI SEPULUH NOPEMBER

Periode: 4 Maret 2024 - 13 Desember 2024

Oleh:

Pelangi Masita Wati 5025221051

Pembimbing Departemen

Prof. Tohari Ahmad, S.Kom., MIT., Ph.D.

Pembimbing Lapangan

Ari Kurniawan

DEPARTEMEN TEKNIK INFORMATIKA

Fakultas Teknologi Elektro dan Informatika Cerdas (FT-EIC)

Institut Teknologi Sepuluh Nopember

Surabaya 2024

KERJA PRAKTIK

Mobile Apps Development: Starred, PlanetPal, FLog, PukaBoo, Robust, Echoes

Apple Developer Academy @UC

DIREKTORAT PENDIDIKAN INSITUT TEKNOLOGI SEPULUH NOPEMBER

Periode: 4 Maret 2024 - 13 Desember 2024

Oleh:

Pelangi Masita Wati 5025221051

Pembimbing Departemen

Prof. Tohari Ahmad, S.Kom., MIT., Ph.D.

Pembimbing Lapangan

Ari Kurniawan

DEPARTEMEN TEKNIK INFORMATIKA

Fakultas Teknologi Elektro dan Informatika Cerdas (FT-EIC)

Institut Teknologi Sepuluh Nopember

Surabaya 2024

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

KERJA PRAKTIK

IOS Mobile Apps Development:

Starred, PlanetPal, FLog, PukaBoo, Robust, Echoes

Oleh:

Pelangi Masita Wati 5025221051

Mengetahui,	Menyetujui,
Pembimbing Lapangan	Dosen Pembimbing
Kerja Praktik	Kerja Praktik
Ari Kurniawan	Prof. Tohari Ahmad, S.Kom., MIT., Ph.D.
	NIP. 197505252003121002

SURABAYA

Desember, 2024

[Halaman ini sengaja dikosongkan]

Mobile Apps Development:

Starred, PlanetPal, FLog, PukaBoo, Robust, Echoes

Nama Mahasiswa : Pelangi Masita Wati

NRP : 5025221051

Departemen : Teknik Informatika

Pembimbing Jurusan : Tohari Ahmad, S.Kom., MIT., Ph.D.

Pembimbing Lapangan : Ari Kurniawan

ABSTRAK

Apple Developer Academy pertama kali diluncurkan di Brasil pada tahun 2013 dengan tujuan memberikan alat serta pelatihan bagi para calon wirausahawan, pengembang, dan perancang untuk menciptakan peluang dalam ekosistem aplikasi iOS yang terus berkembang. Sejak saat itu, Apple telah mendirikan lebih dari belasan akademi di berbagai belahan dunia, termasuk Indonesia, dan berencana untuk memperluas ke tempat-tempat baru seperti Korea dan Detroit, Michigan, yang menjadi lokasi pertama di Amerika Serikat. Program ini telah memberdayakan ribuan siswa di berbagai negara melalui pelatihan pengembangan aplikasi dan wirausaha. Sejumlah lulusan dari program ini telah sukses mendirikan usaha, mengembangkan dan menjual aplikasi di App Store, serta memberikan kontribusi positif untuk komunitas mereka.

Apple Developer Academy menyediakan dua program pelatihan utama. 30 hari yang meliputi area tema tertentu, termasuk pengantar dasar bagi yang ingin berkarir dalam pengembangan aplikasi, serta program akademi intensif selama 10 hingga 12 bulan yang mendalami keterampilan pemrograman, desain, kompetensi profesional, dan pemasaran. Kurikulum juga menggabungkan nilai-nilai Apple, mendorong peserta untuk menciptakan aplikasi yang inklusif dan memberikan dampak baik bagi dunia.

Sebagai peserta magang di Apple Developer Academy Indonesia, saya berkesempatan mendalami ekosistem pengembangan aplikasi dengan pendekatan kolaboratif. Program ini tidak hanya mengajarkan keterampilan teknis, tetapi juga membangun komunitas yang mendukung kreativitas dan inovasi. Dalam program ini, saya belajar bekerja sama dengan rekan-rekan dari latar belakang yang beragam, yang membawa perspektif baru ke dalam proses desain aplikasi dan pengembangan rencana bisnis.

Kata Kunci: Apple Developer Academy, Pengembangan Aplikasi, iOS, Wirausaha, Kolaborasi

KATA PENGANTAR

Saya mengucapkan puji syukur kepada Allah SWT karena dengan rahmat-Nya saya mampu menjalankan salah satu tanggung jawab saya sebagai mahasiswa Departemen Infomatika, yaitu Kerja Praktik (KP).

Saya menyadari bahwa masih terdapat kekurangan dalam pelaksanaan kerja praktik serta penyusunan laporan buku ini. Namun, saya berharap laporan buku ini dapat memperluas pengetahuan pembaca dan berfungsi sebagai sumber referensi. Saya menginginkan umpan balik dan saran yang konstruktif untuk meningkatkan kualitas buku laporan kerja praktik ini.

Lewat buku ini, saya juga ingin menyampaikan rasa terima kasih kepada semua yang telah berkontribusi, baik secara langsung maupun tidak, dalam pelaksanaan kerja praktik serta penyusunan laporan. Orang-orang itu antara lain terdiri dari:Kedua orang tua penulis.

1. Bapak Tohari Ahmad, S.Kom., MIT., Ph.D. selaku dosen pembimbing Kerja Praktik.
2. Bapak Ari Kurniawan, selaku pembimbing lapangan Kerja Praktik.
3. Rekan-rekan tim di Apple Developer Academy.

Surabaya, 26 Desember 2024

Pelangi Masita Wati

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR PSEUDOCODE

BAB I

PENDAHULUAN

1.1 Latar Belakang

Apple Developer Academy pertama kali didirikan di Brasil pada tahun 2013 dengan tujuan menyediakan pelatihan dan fasilitas bagi calon *developer*, desainer, dan wirausahawan untuk berkarya dalam ekosistem aplikasi iOS yang terus berkembang. Sejak saat itu, akademi telah hadir di berbagai negara, termasuk Indonesia, dan terus memberdayakan ribuan pelajar dengan keterampilan teknis dan profesional dalam pengembangan aplikasi dan inovasi teknologi.

Program ini menawarkan kurikulum yang berfokus pada pemrograman, desain aplikasi, pengembangan bisnis, serta pemasaran. Selain itu, peserta juga diajarkan untuk menerapkan nilai-nilai inklusivitas dan dampak positif terhadap masyarakat melalui aplikasi yang dikembangkan. Dalam lingkungan kolaboratif ini, peserta belajar bekerja sama dengan tim lintas disiplin untuk menciptakan solusi digital yang kreatif dan inovatif.

Sebagai salah satu peserta di Apple Developer Academy Indonesia, saya memiliki tanggung jawab untuk mengembangkan aplikasi berbasis iOS yang relevan dengan kebutuhan pengguna. Proses pengembangan melibatkan berbagai tahapan, mulai dari identifikasi masalah, perencanaan solusi, hingga implementasi teknis dan pengujian aplikasi. Dalam setiap tahapan, saya bekerja sama dengan rekan satu tim, memanfaatkan keterampilan yang diperoleh selama program untuk menciptakan produk yang dapat memberikan dampak positif bagi pengguna dan masyarakat secara luas.

Melalui program ini, saya berkesempatan untuk mendalami teknologi Apple, seperti pengembangan aplikasi menggunakan Swift dan SwiftUI, serta praktik terbaik dalam desain dan pengelolaan

aplikasi. Pengalaman ini diharapkan tidak hanya memberikan pemahaman mendalam tentang pengembangan aplikasi, tetapi juga membekali saya dengan keterampilan untuk menghadapi tantangan dalam ekosistem teknologi global.

1.2 Tujuan

Tujuan dari kerja praktik ini adalah untuk menyelesaikan kewajiban kuliah kerja praktik Teknik Informatika di Institut Teknologi Sepuluh Nopember. Selain itu, tujuan lainnya adalah untuk mengembangkan aplikasi berbasis *mobile* yang kreatif dan inovatif sebagai bagian dari program Apple Developer Academy.

1.3 Manfaat

Dengan aplikasi *mobile* yang dikembangkan, diharapkan dapat memberikan solusi nyata untuk memecahkan masalah di masyarakat, sekaligus mendukung visi, misi, dan tujuan Apple Developer Academy. Hasil dari pengembangan aplikasi ini diharapkan dapat menjadi dasar bagi inovasi lebih lanjut, memberikan manfaat yang luas bagi pengguna, serta memperkuat kontribusi teknologi dalam kehidupan sehari-hari.

1.4 Rumusan Masalah

Berikut ini rumusan masalah pada kerja praktik pengembangan aplikasi di Apple Developer Academy:

1. Apa masalah yang dihadapi masyarakat yang dapat diselesaikan melalui aplikasi yang akan dikembangkan?
2. Fitur apa saja yang dapat diimplementasikan untuk memberikan solusi yang efektif dan relevan bagi pengguna?
3. Bagaimana desain antarmuka pengguna (UI) yang inklusif dan mudah digunakan dapat diterapkan dalam aplikasi ini?

4. Teknologi apa yang paling sesuai untuk memastikan performa dan stabilitas aplikasi?
5. Bagaimana aplikasi ini dapat mendukung visi dan misi Apple Developer Academy untuk menciptakan dampak positif di masyarakat?

1.5 Lokasi dan Waktu Kerja Praktik

Kerja praktik ini dilaksanakan pada waktu dan tempat sebagai berikut:

Lokasi : Offline

Waktu : 4 Maret 2024 – 13 Desember 2024

Hari Kerja : Senin – Jumat

Jam Kerja : 08:00 – 12:00

1.6 Metodologi Kerja Praktik

1.6.1 Perumusan Masalah

Dalam tahap ini, kami menggunakan pendekatan Challenge Based Learning (CBL) untuk setiap proyek pengembangan aplikasi. Pendekatan ini dimulai dengan mengidentifikasi tantangan yang relevan dengan permasalahan yang dihadapi masyarakat. Setelah itu, kami mengeksplorasi kebutuhan-kebutuhan yang muncul dari tantangan tersebut dan merumuskan solusi yang dapat diimplementasikan melalui pengembangan aplikasi.

Melalui CBL, kami tidak hanya fokus pada pemahaman masalah, tetapi juga mendorong proses eksplorasi ide secara kolaboratif, melibatkan riset mendalam, serta iterasi desain dan

pengembangan untuk menciptakan solusi yang berdampak positif. Pendekatan ini memastikan bahwa setiap aplikasi yang dikembangkan memiliki dasar yang kuat untuk memecahkan masalah yang nyata dan relevan.

1.6.2 Studi Literatur

Setelah ditentukan rumusan masalah mengenai aplikasi yang akan dikembangkan, dilakukan studi literatur untuk mendukung implementasinya. Pada tahap ini, proses pencarian, pembelajaran, dan pengumpulan informasi dilakukan secara menyeluruh, baik dari sumber-sumber terpercaya di internet maupun dari aplikasi-aplikasi sebelumnya yang serupa dan relevan, serta beberapa sesi *interview* bersama dengan ahli di bidang terkait.

Studi literatur ini mencakup analisis teknologi yang digunakan, pendekatan desain aplikasi, serta strategi pengembangan yang sesuai dengan tantangan yang dihadapi. Informasi yang diperoleh diharapkan dapat memberikan landasan yang kuat untuk proses pengembangan aplikasi, memastikan bahwa solusi yang dihasilkan efektif, inovatif, dan dapat diimplementasikan secara optimal.

1.6.3 Analisis dan Perancangan

Tahap ini mencakup penjelasan mengenai hasil dari studi literatur yang telah dilakukan. Dari berbagai metode yang ditemukan selama proses literasi, dilakukan analisis untuk menentukan metode yang paling tepat dan efektif dalam menyelesaikan permasalahan yang telah dirumuskan.

Dalam tahap ini, diputuskan pula teknologi dan *framework* Apple yang akan digunakan. Selain itu, ditentukan juga batasan-batasan data yang akan digunakan untuk memastikan fokus pengembangan dan hasil yang diharapkan dapat tercapai. Proses analisis dan perancangan ini bertujuan untuk memastikan bahwa solusi yang dirancang tidak

hanya relevan dengan permasalahan, tetapi juga *feasible* untuk diimplementasikan secara teknis.

1.6.4 Implementasi Framework

Pada tahap ini, dijelaskan implementasi *framework* yang digunakan dalam proses pengembangan aplikasi. Bagian ini mencakup langkah-langkah dalam membangun aplikasi menggunakan bahasa pemrograman Swift dengan berbagai *framework* Apple, mulai dari pengembangan fitur inti hingga integrasi desain antarmuka pengguna (UI).

Implementasi dilakukan berdasarkan pendekatan Challenge Based Learning, dengan memastikan setiap solusi yang diterapkan selaras dengan hasil analisis dan perancangan yang telah dilakukan sebelumnya. Selain itu, proses pengujian dilakukan secara iteratif untuk memastikan aplikasi yang dikembangkan memiliki performa yang optimal dan mampu memenuhi kebutuhan pengguna.

1.6.5 Pengujian dan Evaluasi

Pengujian sistem yang dilakukan bertujuan untuk mengevaluasi kinerja aplikasi yang telah dikembangkan. Pada tahap ini, pengujian difokuskan pada fitur-fitur utama dalam aplikasi, seperti fungsionalitas, stabilitas, dan performa. Pengujian juga mencakup analisis terhadap antarmuka pengguna (UI) untuk memastikan aplikasi mudah digunakan dan sesuai dengan standar desain yang diinginkan.

Evaluasi dilakukan dengan mengumpulkan feedback dari pengguna dan sesi *Learning Evidence Discussion*. Selain itu, dilakukan analisis terhadap hasil yang diperoleh untuk memastikan aplikasi memenuhi tujuan dan dapat memberikan solusi yang diharapkan bagi pengguna. Hasil pengujian ini akan digunakan sebagai dasar untuk perbaikan dan pengembangan lebih lanjut.

1.6.6. Kesimpulan dan Saran

Pada bab ini, dipaparkan kesimpulan yang dapat diambil dari seluruh proses pengembangan aplikasi yang telah dilakukan. Kesimpulan mencakup pencapaian tujuan yang telah dirumuskan sebelumnya, serta evaluasi terhadap keberhasilan aplikasi dalam menyelesaikan permasalahan yang dihadapi. Selain itu, dibahas juga tantangan yang dihadapi selama proses pengembangan dan bagaimana solusi-solusi yang diterapkan dapat mengatasi hambatan tersebut.

Selanjutnya, saran diberikan untuk perbaikan dan pengembangan aplikasi di masa depan. Saran ini meliputi aspek teknis, desain, serta fitur tambahan yang dapat meningkatkan kualitas dan performa aplikasi. Saran juga dapat mencakup rekomendasi untuk pengembangan lebih lanjut atau penerapan pendekatan baru yang lebih efektif dalam menghadapi tantangan serupa.

1.7 Sistematika Laporan

1.7.1 Bab I Pendahuluan

Dalam bab ini, penulis menjelaskan latar belakang masalah, tujuan penelitian, waktu pelaksanaan, serta sistematika organisasi kerja praktik dan laporan tertulis dari kerja praktik.

1.7.2 Bab II Profil Perusahaan

Bab ini akan menjelaskan secara mendalam mengenai profil Apple Developer Academy, tempat di mana saya menjalani kerja praktik.

1.7.3 Bab III Tinjauan Pustaka

Pada bahasan ini, diuraikan mengenai kajian pustaka dan referensi yang dipakai dalam penyelesaian kerja praktik di Apple Developer Academy.

1.7.4 Bab IV Implementasi Framework

Sedangkan bab IV, menjabarkan raingkaian fase yang ditempuh agar dapat melakukan implementasi *framework* Apple yang digunakan dalam aplikasi-aplikasi *mobile* yang dikembangkan.

1.7.5 Bab V Pengujian dan Evaluasi

Kemudian bab V, terdapat penguraian dari *output* evaluasi dan *feedback* dari *softwares* yang selama kerja praktik dikembangkan.

1.7.6 Bab VI Kesimpulan dan Saran

Di bab terakhir, terdapat kesimpulan dan saran untuk aplikasi-aplikasi yang dibuat dalam proses kerja praktik tersebut.

BAB II

LATAR BELAKANG PERUSAHAAN

2.1 LATAR BELAKANG Apple Developer Academy

Apple Developer Academy adalah sebuah program pendidikan yang didirikan oleh Apple untuk memberikan pelatihan dan sumber daya kepada calon pengembang aplikasi, desainer, dan pengusaha di seluruh dunia. Akademi pertama kali dibuka di Brazil pada tahun 2013 dengan tujuan untuk memberikan pelatihan kepada individu yang ingin berkarier di industri aplikasi iOS yang berkembang pesat. Sejak itu, Apple telah membuka lebih dari sepuluh akademi di berbagai negara, termasuk di Indonesia, Italia, dan kini di Korea dan Detroit, Amerika Serikat.

Program ini bertujuan untuk memberikan keterampilan pengembangan aplikasi yang mendalam dan pengetahuan kewirausahaan kepada para peserta. Apple Developer Academy tidak hanya fokus pada pengajaran teknologi, tetapi juga melibatkan peserta dalam pembelajaran desain, pengembangan produk, dan pemahaman mendalam tentang pasar aplikasi. Dengan lebih dari 1.000 lulusan setiap tahunnya, banyak alumni Apple Developer Academy yang telah menciptakan aplikasi yang sukses di App Store atau bahkan memulai bisnis mereka sendiri.

Di Indonesia, salah satu Apple Developer Academy berlokasi di Surabaya, di mana para peserta dapat mengakses berbagai fasilitas dan sumber daya yang disediakan oleh Apple. Program ini menggunakan pendekatan Challenge Based Learning untuk mendorong peserta menghadapi tantangan nyata dan mengembangkan solusi yang inovatif melalui aplikasi. Dengan pengajaran langsung dari para ahli Apple dan kolaborasi dengan pengembang global lainnya, akademi ini telah menjadi salah satu pusat pendidikan teknologi di dunia.

Tim pengajar di Apple Developer Academy terdiri dari profesional berpengalaman di bidang teknologi, desain, dan bisnis yang siap memberikan wawasan dan bimbingan untuk membantu peserta mengembangkan keterampilan teknis dan profesional mereka. Selain itu, Apple juga menawarkan akses ke berbagai alat dan *framework* terbaru yang digunakan dalam pengembangan aplikasi, seperti Swift, SwiftUI, Xcode, dan banyak lagi.

Dengan semangat inovasi dan pemberdayaan, Apple Developer Academy berkomitmen untuk mencetak generasi baru pengembang aplikasi yang dapat menciptakan solusi yang mengubah dunia.

2.2 Simbol Perusahaan

Gambar 2.2 Logo Apple Developer Academy

2.3 Visi & Misi Perusahaan

2.3.1 Visi Perusahaan

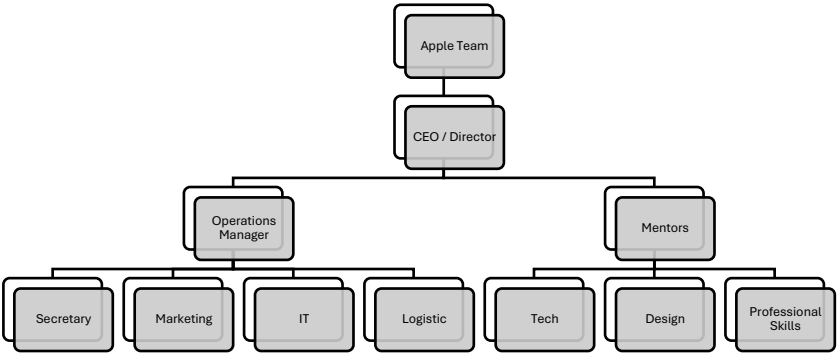
Menghasilkan *developer* aplikasi kelas dunia yang mampu menyelesaikan permasalahan di masyarakat dan mendorong inovasi melalui keterampilan pengembangan perangkat lunak, untuk

mendukung kemajuan teknologi yang bermanfaat bagi masyarakat global.

2.3.2 Misi

Menyediakan pelatihan dan pengembangan keterampilan pengembangan perangkat lunak kelas dunia melalui pendekatan berbasis tantangan, untuk mempersiapkan para peserta dalam menciptakan solusi inovatif yang dapat memecahkan permasalahan nyata di masyarakat dan mendukung kesuksesan mereka dalam industri teknologi global.

2.4 Struktur Organisasi



Gambar 2.4 Struktur Organisasi Apple Developer Academy

BAB III

TINJAUAN PUSTAKA

Pada bab III ini, diuraikan tentang prinsip-prinsip teori yang diterapkan selama pelaksanaan kerja praktik.

3.1 XCode

Xcode adalah lingkungan pengembangan terpadu (IDE) yang dikembangkan oleh Apple untuk membuat aplikasi di seluruh ekosistemnya, termasuk macOS, iOS, iPadOS, watchOS, dan tvOS. Xcode menyediakan rangkaian *tools* yang komprehensif bagi *developer* untuk merancang, membangun, menguji, dan mendistribusikan perangkat lunak dengan efisien (Vajpai, 2024).

Xcode menggabungkan editor kode dengan *debugging tools*, kemampuan manajemen aplikasi, dan Interface Builder. Lingkungan terpadu ini menyederhanakan proses pengembangan mulai dari membuat kode hingga pengujian dan *debugging* aplikasi. Xcode mendukung kerangka kerja pengujian unit dan UI, memungkinkan pengembang untuk mengotomatisasi pengujian aplikasi mereka guna memastikan kualitas sebelum diproduksi (Hada, 2023).

Interface Builder, visual *tool* dalam Xcode, memungkinkan pengembang mendesain antarmuka pengguna menggunakan antarmuka drag-and-drop. Alat ini mendukung UIKit dan SwiftUI, sehingga mempermudah pengembang dalam menciptakan tata letak responsif. Xcode juga dilengkapi dengan debugger dengan berbagai fitur, sehingga pengembang dapat menetapkan breakpoints, memeriksa variabel, dan menganalisis masalah performa *real-time*. Di samping itu, terdapat juga fitur yang berguna untuk mendiagnosis penggunaan memori dan performa runtime lainnya. Xcode mendukung Git dan *sistem kontrol versi lainnya* secara bawaan, sehingga pengembang

dapat mengelola repositori kode mereka langsung di dalam IDE. Xcode secara rutin diperbarui dengan teknologi, kerangka kerja, dan API terbaru dari Apple, sehingga pengembang dapat memanfaatkan fitur-fitur baru yang tersedia (Mannotra, 2024).

IDE ini memiliki kemampuan pengeditan kode, seperti syntax highlighting, auto-completion, dan code folding. Pengembang dapat menavigasi aplikasi mereka secara efisien menggunakan file navigator dan berbagai keyboard *shortcut*. Xcode juga menyediakan simulator untuk semua perangkat Apple, memungkinkan pengembang menguji aplikasi mereka pada berbagai ukuran layar dan konfigurasi tanpa memerlukan perangkat fisik (Ekren, 2024).

3.2 SwiftUI & UIKit

SwiftUI adalah *framework* deklaratif yang dikembangkan oleh Apple Inc. untuk membangun antarmuka pengguna di berbagai platform, termasuk iOS, iPadOS, macOS, watchOS, tvOS, dan visionOS. *Framework* ini pertama kali diperkenalkan pada 3 Juni 2019 dan dirancang untuk menyederhanakan proses pengembangan UI dengan memungkinkan pengembang untuk mendeskripsikan apa yang harus dilakukan oleh antarmuka pengguna mereka, bukan bagaimana cara mengimplementasikannya. SwiftUI dirancang agar dapat digunakan bersama UIKit. Misalnya, jika elemen UI tertentu tidak tersedia di SwiftUI, pengembang masih dapat menggunakan komponen UIKit dalam aplikasi SwiftUI mereka (Apple, 2024).

Menggunakan SwiftUI, *developer* hanya perlu menentukan hasil yang diinginkan dari komponen UI mereka. Misalnya, pengembang dapat mendefinisikan daftar item dan menyesuaikan tampilannya dengan sedikit kode. Terintegrasi dengan Xcode, SwiftUI mendukung *realtime preview* yang langsung memperlihatkan perubahan saat kode diperbarui. Fitur ini memungkinkan pengembang melihat bagaimana antarmuka pengguna mereka terlihat di berbagai

perangkat dan konfigurasi, seperti *dark mode* atau berbagai ukuran layar. *Framework* ini juga menyederhanakan penambahan animasi pada elemen UI. Pengembang dapat menerapkan animasi dengan beberapa baris kode, sementara SwiftUI menangani kompleksitas di balik interaksi pengguna dan perubahan status selama animasi berlangsung (WWDC, 2024).

Untuk mengembangkan aplikasi menggunakan SwiftUI, pengembang utamanya menggunakan Xcode. IDE ini menyediakan alat desain intuitif yang mendukung fungsi drag-and-drop untuk membangun antarmuka. Saat mendesain di kanvas, kode terkait diperbarui secara otomatis, sehingga pengalaman pengembangan menjadi lebih lancar (Jacq, 2021).

UIKit adalah framework lain yang dikembangkan oleh Apple Inc. yang menyediakan *tools* untuk membuat antarmuka pengguna di aplikasi iOS, iPadOS, dan tvOS. *Framework* ini diperkenalkan oleh Apple pertama kali di tahun 2008 dan telah menjadi komponen dasar bagi pengembang yang membuat aplikasi di platform tersebut. UIKit bekerja mengikut arsitektur perangkat lunak MVC (*Model-View-Controller*), yang membagi aplikasi menjadi tiga komponen yang saling terhubung. Struktur ini meningkatkan organisasi kode dan memudahkan proses *maintenance*. UIKit juga menyediakan pengenalan gestur (gesture recognizers) yang menyederhanakan implementasi interaksi berbasis sentuhan, sehingga meningkatkan pengalaman pengguna pada perangkat berbasis sentuh (SW Team, 2024).

Pengembang dapat membuat antarmuka pengguna secara visual menggunakan Storyboards, yang memungkinkan pengaturan elemen UI dan koneksinya ke kode melalui IBOutlets dan IBActions. UIKit juga mencakup berbagai komponen bawaan seperti tombol, label, *text field*, dan *collection views*. Komponen-komponen ini dapat

dikustomisasi dan digabungkan untuk menciptakan antarmuka pengguna yang kompleks (Kramer, 2024).

Fitur UIKit mendukung desain responsif dengan membuat pengembang bisa mendefinisikan *constraint* yang menentukan bagaimana elemen UI harus diposisikan dan diukur relatif satu sama lain dan ukuran layarnya. UIKit bisa digunakan dalam Xcode untuk menulis kode dalam Swift atau Objective-C, serta menguji aplikasi di simulator atau perangkat fisik. Framework ini juga mendukung fitur seperti fungsi drag-and-drop dan penyesuaian tipe dinamis untuk aksesibilitas. Walaupun SwiftUI telah muncul sebagai alternatif modern untuk membangun antarmuka pengguna dengan sintaks deklaratif, UIKit tetap banyak digunakan karena kematangannya dan dokumentasinya yang luas. Banyak aplikasi yang sudah ada masih bergantung pada UIKit, sehingga penting bagi pengembang untuk memahami kedua framework ini saat bekerja pada aplikasi baru atau memelihara kode lama (Ndungu, 2024).

3.3 Figma Prototyping

Figma menyediakan alat prototyping yang membuat desainer bisa menciptakan prototipe interaktif dengan fidelitas tinggi tanpa perlu menulis kode. Figma memungkinkan pengguna untuk merancang dan membuat prototipe dalam lingkungan yang sama. Hal ini memudahkan iterasi dan penyesuaian yang cepat berdasarkan masukan atau *feedback* (Figma, 2024).

Pengguna dapat membuat interaksi kompleks dengan mudah menggunakan komponen interaktif. Fitur ini memungkinkan transisi otomatis antara berbagai status atau varian elemen UI, sehingga meningkatkan realisme prototipe. Prototipe dapat dilihat pada berbagai ukuran layar, memastikan desain dapat beradaptasi dengan perangkat yang berbeda. Fitur ini sangat penting untuk menguji bagaimana antarmuka akan berfungsi pada lingkungan mobile dibandingkan

desktop. Selain itu, Figma memungkinkan pembuatan beberapa prototipe dalam satu file, memudahkan desainer untuk memetakan berbagai perjalanan pengguna atau segmen spesifik dari sebuah aplikasi (Fedorenko, 2024).

Fitur Smart Animate di Figma mendukung transisi dan animasi yang halus antar frame, sehingga mempermudah visualisasi interaksi pengguna seperti *scrolling* dan *tapping*. Desainer dapat menentukan interaksi berdasarkan tindakan pengguna, seperti klik, *hover*, dan *scroll*. Fitur ini membantu mensimulasikan skenario penggunaan nyata, memberikan gambaran yang lebih jelas tentang pengalaman pengguna. Untuk menyiapkan interaksi, desainer dapat memilih frame dan mendefinisikan apa yang terjadi saat pengguna berinteraksi dengan elemen tertentu (misalnya, navigasi ke frame lain). Hal ini meliputi penentuan jenis animasi dan pengaturan waktu (Figma, 2024).

Untuk membuat koneksi antar *frame*, pengguna dapat beralih ke tab Prototype di *sidebar* kanan. Setiap koneksi terdiri dari *hotspot* (titik awal), detail interaksi, dan *frame* tujuan. Setelah koneksi dibuat, prototipe dapat dipratinjau dalam mode presentasi untuk menguji alur dan interaksi pengguna. Proses ini memungkinkan masukan langsung dan penyesuaian yang diperlukan sebelum desain difinalisasi (Figma, 2024).

3.4 On-Device Data Storage (AppStorage and SwiftData)

Ketika mengembangkan aplikasi iOS, pengelolaan penyimpanan data secara efektif merupakan hal yang esensial untuk kualitas pengalaman pengguna yang baik. Dua metode yang dikenal untuk penyimpanan data di perangkat dalam ekosistem Apple adalah AppStorage dan SwiftData.

AppStorage adalah *property wrapper* dalam SwiftUI yang mempermudah pengelolaan UserDefaults. AppStorage memungkinkan

pengembang untuk menyimpan tipe data sederhana secara persisten di seluruh aplikasi tanpa perlu mengelola API UserDefaults secara langsung. AppStorage menyediakan antarmuka yang mudah digunakan untuk menyimpan dan mengambil nilai seperti string, integer, dan boolean (Lee, 2024).

Ketika nilai yang disimpan di AppStorage berubah, semua tampilan yang bergantung pada nilai tersebut secara otomatis diperbarui untuk menampilkan perubahan tersebut. AppStorage mendukung tipe-tipe data dasar seperti String, Int, dan Bool, sementara tipe yang lebih kompleks mungkin memerlukan penyesuaian tambahan (Fatbobman, 2024). Data yang disimpan menggunakan AppStorage tetap ada meskipun aplikasi ditutup dan dibuka kembali (Pereira, 2024). Namun, AppStorage tidak dirancang untuk dataset besar. Jadi lebih cocok untuk jumlah data kecil, seperti menyimpan preferensi atau pengaturan kustom dari pengguna. Untuk dataset yang lebih besar atau hubungan data yang lebih kompleks, Core Data atau SwiftData lebih sesuai untuk digunakan (Reddit, 2024).

SwiftData adalah *framework* yang lebih baru yang diperkenalkan oleh Apple. *Framework* ini dibangun berdasarkan prinsip Core Data, tetapi dirancang untuk menyederhanakan manajemen data dengan fokus pada integrasi dengan SwiftUI. SwiftData menggunakan sintaks deklaratif yang lebih sederhana, selaras dengan filosofi desain Swift, sehingga mempermudah pengembang dalam mendefinisikan model data dan hubungan antar data. Selain itu, SwiftData mendukung sinkronisasi data antar perangkat menggunakan iCloud, memastikan konsistensi data pengguna meskipun pengguna beralih perangkat atau memulihkan data dari cadangan. Dengan dukungan bawaan untuk pemfilteran dan pengurutan, pengembang dapat mengelola dataset yang kompleks dengan lebih efisien. SwiftData dirancang untuk bekerja secara *seamless* dengan SwiftUI, sehingga pembaruan reaktif pada antarmuka

pengguna saat data dasar berubah dapat diimplementasikan (Hudson, 2023).

Untuk penggunaannya AppStorage bisa diimplementasikan dengan pasangan *k*ey-*value* sederhana (seperti pengaturan pengguna) dan SwiftData untuk model data yang lebih kompleks yang memerlukan hubungan atau kueri lanjutan. Jika sinkronisasi data antar perangkat diperlukan, integrasi iCloud pada SwiftData menjadi solusi yang tepat. AppStorage tidak mendukung sinkronisasi ini secara bawaan. Kedua metode memastikan data pengguna dapat bertahan di seluruh sesi aplikasi. Namun, jika aplikasi dihapus tanpa sinkronisasi cloud, semua data lokal akan hilang kecuali dicadangkan secara eksplisit.

3.5 3rd Party Framework Integration (URLSession)

Mengintegrasikan *framework* pihak ketiga dalam aplikasi Swift sering kali melibatkan permintaan jaringan ke API eksternal. Salah satu *tool* utama untuk tujuan ini adalah URLSession, yang menyediakan cara untuk menangani permintaan dan respons HTTP. Berikut ini adalah cara kerja URLSession untuk mengintegrasikan API pihak ketiga serta keunggulannya dibandingkan metode lain.

URLSession adalah sebuah kelas dalam *framework* Foundation yang menyediakan API untuk mengunduh data dari dan mengunggah data ke endpoint yang ditentukan oleh URL. URLSession dirancang untuk menangani berbagai tugas jaringan, seperti mengambil data dari layanan web, mengunggah file ke server, dan mengunduh file dari lokasi eksternal.

URLSession beroperasi secara asinkron, memungkinkan aplikasi tetap responsif saat operasi jaringan berlangsung. Hal ini sangat penting untuk menjaga pengalaman pengguna yang bagus. Kelas ini menyediakan beberapa jenis tugas (*data tasks, upload tasks, download*

tasks) yang dapat dikelola dengan mudah, memberi fleksibilitas kepada pengembang dalam menangani komunikasi jaringan.

Selain itu, URLSession mendukung *caching* respons yang dapat meningkatkan kinerja dengan mengurangi kebutuhan untuk melakukan permintaan jaringan berulang. Pengembang dapat menyesuaikan permintaan HTTP dengan menetapkan header, konten body, dan metode permintaan seperti GET, POST, dan sebagainya. Sehingga mempermudah interaksi dengan berbagai API. URLSession juga menawarkan kemampuan penanganan *error* yang komprehensif, sehingga pengembang mengelola kesalahan jaringan dengan baik.

Untuk penggunaannya, sebelum membuat permintaan, pastikan telah mendapatkan kunci API atau token autentikasi yang diperlukan oleh layanan pihak ketiga. Setelah respons diproses, perbarui antarmuka pengguna aplikasi sesuai dengan data yang diperoleh (Lagadhir, 2023).

Gunakan URLSession untuk membuat objek permintaan yang menentukan endpoint dan header atau parameter yang diperlukan. Gunakan completion handler untuk memproses data respons yang dikembalikan oleh API. Hal ini sering kali melibatkan parsing data JSON menjadi objek Swift menggunakan protokol Codable. Untuk tugas yang mungkin memakan waktu lama atau perlu dilanjutkan saat aplikasi berjalan di latar belakang, pertimbangkan menggunakan background sessions. Implementasikan monitoring untuk aktivitas jaringan dan kesalahan guna meningkatkan pengalaman pengguna dan memperbaiki masalah dengan lebih efektif. Selalu gunakan endpoint HTTPS untuk melindungi data pengguna selama transmisi dan terapkan mekanisme autentikasi yang tepat (Crudu, 2024).

3.6 Machine Learning (Core ML and Create ML)

Apple menyediakan dua *framework* yang dapat digunakan agar *developer* dapat mengimplementasikan *machine learning* dalam aplikasi, yaitu Core ML dan Create ML. *Framework* ini mempermudah proses penerapan model hasil *training machine learning* ke aplikasi berbagai *device* ekosistem Apple, sehingga pengembang menciptakan aplikasi cerdas yang adaptif.

Core ML sendiri merupakan sebuah *framework machine learning* yang memungkinkan *developer* mengintegrasikan berbagai jenis model *machine learning* ke dalam aplikasi. *Framework* ini dioptimalkan untuk performa di perangkat, memastikan model berjalan dengan efisien sambil meminimalkan penggunaan memori dan konsumsi daya (Apple, 2024). Core ML mendukung berbagai jenis model, termasuk *deep learning, tree ensembles, support vector machines* (SVMs), dan *generalized linear models* (Bao, 2024).

Dengan memanfaatkan Apple Silicon, Core ML mengoptimalkan eksekusi model *machine learning*, memungkinkan model berjalan lebih cepat dan efisien di perangkat. Fitur baru dalam Core ML, yaitu MLTensor, menyediakan API yang familiar untuk operasi pada *array multi-dimensional*, meningkatkan performa dan fleksibilitas dalam eksekusi model (Apple, 2024).

Pengembang dapat ah mengintegrasikan kemampuan *machine learning* ke dalam aplikasi menggunakan Swift. *Framework* ini menyediakan API yang sederhana untuk melakukan prediksi serta menangani input dan output model. Model yang dibuat dengan Create ML dapat diekspor dalam format .mlmodel dan diintegrasikan ke dalam aplikasi Xcode (Siwal, 2024).

Create ML adalah *tool* yang dirancang untuk menyederhanakan proses *training model machine learning*. Create ML menyediakan antarmuka yang ramah pengguna sehingga pengembang dapat membuat model kustom tanpa memerlukan pengetahuan mendalam

tentang algoritma machine learning. Framework ini menawarkan antarmuka grafis yang bisa membuat pengembang membangun dan melakukan *training* model menggunakan fungsi drag-and-drop (Ramadhan, 2024).

Create ML juga menyediakan *template* untuk tugas-tugas umum seperti *image classification, object detection*, dan *text classification*, sehingga lebih mudah diakses, terutama bagi pemula (Prasatya, 2024).

3.7 Audio/Video (AVFoundation)

AVFoundation adalah *framework* yang dikembangkan oleh Apple untuk menangani media audiovisual berbasis waktu di iOS, macOS, watchOS, dan tvOS. *Framework* ini menyediakan rangkaian *tool* yang komprehensif untuk memutar, membuat, mengedit, dan memproses konten audio dan video. Dengan AVFoundation, pengembang dapat membangun aplikasi media dengan berbagai fitur untuk menangani berbagai macam tugas terkait data audiovisual.

AVFoundation menyediakan *tool* untuk mengedit aset audio dan video. Pengembang dapat membuat komposisi dengan menggabungkan beberapa trek, menyesuaikan properti seperti volume atau opasitas, dan mengekspor hasil akhirnya dalam berbagai format (Apple, 2024).

Framework ini memungkinkan pengembang untuk memutar file audio dan video, termasuk mendukung protokol streaming seperti HTTP Live Streaming (HLS). AVFoundation juga dapat menangani format seperti QuickTime movies dan MPEG-4 files, memberikan fleksibilitas dalam konsumsi media.

Kelas utama dalam AVFoundation adalah AVAsset, yang merepresentasikan data media. Kelas ini menyediakan opsi bagi pengembang untuk memeriksa properti file media, seperti durasi dan

format, tanpa harus memuat seluruh file ke dalam memori. Fitur ini sangat berguna untuk menangani file berukuran besar secara efisien. *Framework* ini juga mencakup kelas seperti AVAudioPlayer untuk pemutaran audio dan AVAudioRecorder untuk perekaman audio. Selain itu, AVFoundation mendukung pemrosesan audio tingkat lanjut melalui AVAudioEngine, untuk pengaturan efek seperti reverb atau penyesuaian pitch (Apple Documentation, 2024).

Framework ini mendukung perekaman audio dan video dari kamera dan mikrofon perangkat. Pengembang dapat membuat aplikasi yang merekam media secara real-time, menawarkan fitur seperti pratinjau langsung dan konfigurasi khusus untuk pengambilan media. AVFoundation juga menyediakan opsi untuk integrasi metadata berjangka waktu ke dalam komposisi video, yang dapat digunakan untuk subtitle atau closed captions. Kemampuan ini meningkatkan aksesibilitas dan pengalaman pengguna dalam aplikasi media (Apple, 2024).

3.8 Notification Center

Notification center Apple adalah fitur yang terintegrasi dalam iOS, iPadOS, macOS, dan watchOS, yang menyediakan pengguna untuk dapat mengelola dan melihat pemberitahuan dari berbagai aplikasi di satu lokasi terpusat. Fitur ini meningkatkan pengalaman pengguna dengan menyediakan cara yang efisien untuk menangani pemberitahuan, meminimalkan gangguan, sekaligus menjaga pengguna tetap terinformasi tentang pembaruan penting (Apple, 2024).

3.9 Apple Pencil (PencilKit)

PencilKit adalah framework dari Apple yang dirancang sehingga pengembang dapat mengintegrasikan kemampuan menggambar dan menulis tangan ke dalam aplikasi mereka, dengan memanfaatkan presisi dari Apple Pencil. Diperkenalkan pada WWDC

2019, PencilKit menyediakan serangkaian *tool* lengkap untuk membuat, mengedit, dan mengelola gambar dengan cara yang mudah digunakan. Pengembang dapat mengimplementasikan kanvas gambar menggunakan PKCanvasView, yang menyediakan area responsif bagi pengguna untuk menggambar. Kanvas ini mendukung berbagai fitur seperti fungsionalitas *undo/redo* dan kemampuan untuk menghapus gambar. PencilKit mendukung input baik menggunakan jari maupun Apple Pencil, memungkinkan pengguna untuk beralih antara gaya menggambar dengan mudah. Pengembang dapat mengonfigurasi pengaturan untuk memprioritaskan input dari Apple Pencil (Apple, 2020).

Dengan PencilKit, pengembang dapat mengakses informasi detail tentang goresan dalam gambar, termasuk jalur, tinta, dan titik. Sehingga fitur-fitur canggih seperti pengenalan bentuk dan modifikasi dinamis pada gambar berdasarkan input pengguna bisa berjalan. *Framework* ini meliputi palet *tools* yang memberikan pengguna opsi untuk memilih berbagai alat gambar misalnya, pena, spidol dan menyesuaikan warna juga. Sehingga meningkatkan pengalaman pengguna dengan menyediakan berbagai opsi untuk membuat karya seni (Apple, 2020).

3.10 Gesture Recognition (UIGestureRecognizer)

UIGestureRecognizer adalah kelas dalam UIKit yang menyediakan fitur untuk mengenali dan merespons berbagai gerakan yang dilakukan oleh pengguna pada perangkat yang mendukung sentuhan. *Framework* ini menyederhanakan proses menangani interaksi sentuhan yang kompleks, seperti ketukan, gesekan, cubitan, dan rotasi, sehingga pengalaman pengguna bisa lebih intuitif. Protokol UIGestureRecognizerDelegate berfungsi untuk menyesuaikan perilaku pengenal Gerakan yang mencakup metode untuk menentukan apakah pengenal gerakan harus mulai mengenali suatu gerakan saja atau

apakah pengenal gerakan harus mengenali beberapa gerakan secara bersamaan.

Setiap pengenal gerakan memiliki status yang menunjukkan keadaan saat ini misalnya, mungkin, dimulai, berubah, selesai, atau gagal. Pengembang dapat menggunakan status-status ini untuk mengelola interaksi secara efektif dan merespons dengan tepat berdasarkan tindakan pengguna. Pengenal gerakan menyediakan properti yang dapat dikonfigurasi untuk menyesuaikan perilaku mereka. Misalnya, properti seperti cancelsTouchesInView, delaysTouchesBegan, dan delaysTouchesEnded dapat diatur untuk mengontrol bagaimana sentuhan ditangani ketika gerakan tersebut terdeteksi.

3.11 Geo Location (MapKit)

MapKit adalah *framework* yang disediakan oleh Apple untuk mengintegrasikan peta dan fitur berbasis lokasi ke dalam aplikasi iOS, macOS, dan watchOS. *Framework* ini menyediakan *tool* untuk menampilkan peta, menambahkan anotasi, dan melakukan geocoding, yang sangat penting untuk mengonversi alamat menjadi koordinat geografis dan sebaliknya (Mattos, 1962).

Pengembang dapat menyematkan peta interaktif dalam aplikasi menggunakan MKMapView. Tampilan ini mendukung berbagai jenis peta, termasuk tampilan standar, satelit, dan hibrida. Framework ini juga mencakup layanan geocoding yang mengonversi alamat yang dapat dibaca manusia menjadi koordinat geografis seperti garis lintang dan garis bujur menggunakan CLGeocoder. Fitur ini sangat penting untuk aplikasi yang perlu menampilkan lokasi berdasarkan input pengguna (Apple, 2024).

MapKit menyediakan kemampuan perutean sehingga pengembang dapat menghitung arah antar lokasi dan menampilkan rute

di peta. Dengan integrasi Core Location, MapKit dapat melacak lokasi pengguna saat ini dan memberikan pembaruan waktu nyata seiring pergerakan *device* (Apple, 2024).

3.12 Game Development (SceneKit)

SceneKit adalah *framework* grafis 3D yang disediakan oleh Apple, dirancang untuk memfasilitasi pembuatan pengalaman 3D yang imersif dan interaktif untuk aplikasi iOS, macOS, dan watchOS. *Framework* ini sangat digunakan dalam pengembangan game karena kemampuan kinerjanya yang tinggi dan kemudahan penggunaannya. SceneKit menggunakan arsitektur berbasis node, yang memudahkan untuk mengatur dan mengelola objek game dalam scene 3D. Struktur ini memungkinkan manipulasi node secara fleksibel, memudahkan pengaturan scene yang kompleks. SceneKit juga terintegrasi dengan ARKit, sehingga pengembang bisa menggabungkan konten digital dengan dunia nyata, meningkatkan pengalaman realitas pengguna. Gunakan Xcode Scene Editor untuk alur kerja yang lebih baik. Editor ini menyediakan alat visual yang membantu dalam merancang dan debugging *scene* 3D, sehingga dapat meningkatkan kecepatan dan efektivitas pengembangan secara keseluruhan (Apple, 2017).

Dioptimalkan untuk grafis 3D, SceneKit menyediakan *rendering* cepat dan penggunaan sumber daya sistem yang efisien, menjadikannya cocok untuk aplikasi seperti game. Organisasikan dunia game menggunakan instance SCNScene. Kelola node secara hierarkis untuk menjaga kompleksitas tetap terkontrol dan mengoptimalkan efisiensi *rendering* (30DaysCoding, 2024).

SceneKit dilengkapi dengan mesin fisika bawaan, sehingga pengembang dapat menambahkan efek fisika realistis ke dalam *scene* 3D. *Framework* ini juga mencakup sistem animasi yang mampu membuat animasi dan simulasi kompleks, meningkatkan daya tarik visual game secara keseluruhan. Pisahkan logika game ke dalam lapisan

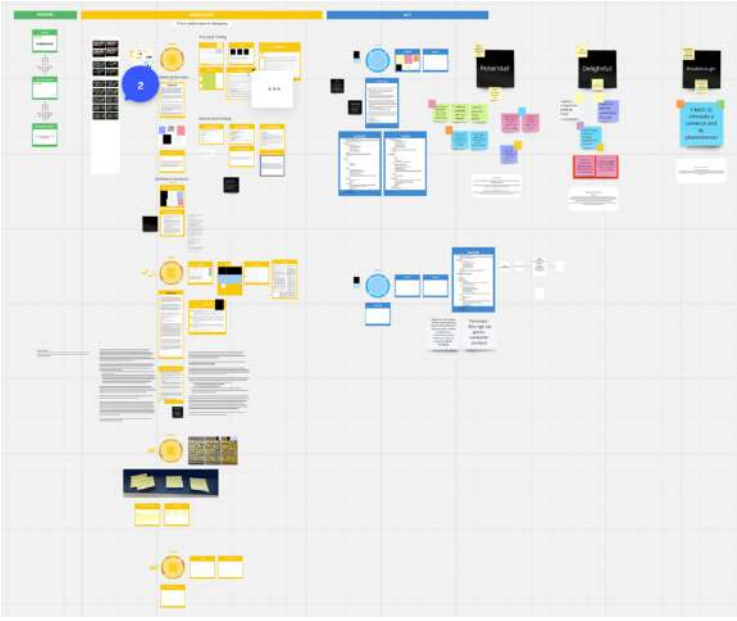
yang independen dari platform menggunakan kelas Swift. Hal ini membantu dalam pengujian dan mempertahankan modularitas (Sulevus, 1962).

BAB IV

IMPLEMENTASI FRAMEWORK

4.1 Starred

Starred adalah sebuah aplikasi *gamification* yang menggabungkan keindahan alam semesta dengan *gameplay* yang seru, memperkenalkan pengalaman mengamati bintang dengan cara yang menyenangkan dan menghadirkan eksplorasi langit malam ke ujung jari pemain. Dengan fitur-fitur yang beragam, pemain dapat menjelajahi karakteristik benda-benda langit, mulai dari menangkap objek-objek angkasa hingga berpartisipasi dalam pertempuran PvP Bersama teman.



Gambar 4.1 Proses *ideation* dan *concepting* untuk aplikasi Starred

Fitur seperti Starstop menawarkan pengalaman mengamati bintang dengan tingkat polusi cahaya yang minimal, sementara Stardex memberikan informasi terperinci mengenai benda-benda langit. Selain itu, fitur Personal Universe memungkinkan pemain untuk berinteraksi dengan sistem tata surya mereka sendiri, menjanjikan perjalanan edukatif melalui keindahan kosmos. Menyediakan petualangan luar biasa di alam semesta dengan *Starred* untuk pengguna.

Aplikasi pertama ini sangat bergantung pada prototyping menggunakan Figma untuk mendesain antarmuka dan alur permainan. Untuk pengembangan lebih lanjut, diperlukan teknologi berikut.

- Augmented Reality (AR): Untuk memberikan pengalaman interaktif yang imersif.
- Game Development: Menggunakan SceneKit atau SpriteKit untuk menciptakan elemen permainan yang menarik.
- Geo Location: Dengan teknologi Maps, Beacons, dan MapKit untuk menjelajahi lokasi bintang berdasarkan posisi geografis.
- Custom Camera: Menggunakan AVFoundation dan AVCAM untuk menangkap gambar bintang dan mengintegrasikan realitas dengan permainan.

4.2 PlanetPal

PlanetPal adalah aplikasi yang dirancang untuk membantu pengguna memahami dan mengelola jejak karbon mereka. Dengan fitur-fitur seperti perhitungan jejak karbon, pembaruan berita lingkungan terkini, dan rekomendasi tindakan berdasarkan ukuran jejak karbon yang dimiliki pengguna. *PlanetPal* memberdayakan pengguna untuk membuat keputusan bijak serta mengambil langkah-langkah menuju gaya hidup yang lebih berkelanjutan.

Gambar 4.2 Proses *ideation* dan *concepting*

Dalam proses pengembangannya, aplikasi ini menggunakan teknologi berikut

- On-Device Data Storage (AppStorage): Digunakan untuk menyimpan data kalkulasi *carbon footprint* pengguna secara lokal di perangkat.
- SwiftUI: Digunakan untuk merancang antarmuka pengguna yang intuitif dan responsif.

```
import SwiftUI

struct GasBillView: View {
    @State private var gasBill: Double = 0
    @AppStorage("gasBill") private var gas = 0.0

    var body: some View {
        Form {
            Section(header: Text("Gas Bill")) {
                TextField("Enter gas bill", value: $gasBill, formatter:
NumberFormatter())
                    .keyboardType(.decimalPad)
                    .onChange(of: gasBill) { newValue in
                        gas = newValue
                    }
                    .onAppear {
                        gasBill = gas
                    }
            }
        }
    }
}

struct GasBillView_Previews: PreviewProvider {
    static var previews: some View {
        GasBillView()
    }
}
```

Pseudocode 4.2 Implementasi AppStorage

4.3 FLog

FLog adalah aplikasi yang dirancang untuk membantu pengguna melacak asupan nutrisi harian mereka. Dengan aplikasi ini, pengguna dapat mencatat makanan yang mereka konsumsi melalui berbagai input, seperti mengambil foto, mengetik secara manual, atau menggunakan perintah suara. Hanya dengan satu klik, teknologi pembelajaran mesin di latar belakang akan menganalisis dan memberikan informasi tentang kandungan makronutrisi dari makanan tersebut.

Gambar 4.3.1 Proses *ideation* dan *concepting*

Gambar 4.3.2 *Workflow* dari aplikasi FLog

Lebih dari itu, pengguna juga dapat mempertimbangkan akumulasi makronutrisi mereka melalui grafik, sehingga membantu mereka memutuskan apakah makanan tersebut cocok dengan kebutuhan mereka. Aplikasi ini membantu pengguna untuk melacak asupan makronutrisi harian mereka serta melihat riwayat konsumsi untuk mendorong kebiasaan makan yang sehat.

Untuk mengahadirkan semua fitur tersebut, *Flog* mengimplementasikan berbagai teknologi berikut:

- Machine Learning (Core ML dan Create ML): Digunakan untuk mengidentifikasi kandungan nutrisi makanan melalui gambar atau teks yang dimasukkan pengguna.
- Computer Vision: Untuk mengenali makanan dari gambar yang diunggah pengguna melalui foto dari galeri atau tangkapan *camera* secara langsung.
- On-Device Data Storage: Memastikan data pengguna disimpan dengan aman di perangkat, melindungi privasi pengguna, dan memudahkan akses data riwayat konsumsi.
- SwiftUI: Untuk merancang antarmuka pengguna yang interaktif dan intuitif.
- Voice Assistance: Memberikan kemampuan input data menggunakan perintah suara, menjadikan aplikasi lebih ramah pengguna.
- Audio/Video (AVFoundation): Mendukung fitur suara dan media dalam aplikasi, seperti perintah suara.
- Notification: Mengingatkan pengguna untuk mencatat makanan mereka, *eat reminder*, atau memantau kemajuan harian.
- 3rd-Party Library Framework (URLSession): Digunakan untuk integrasi API eksternal jika diperlukan, seperti mengakses basis data nutrisi global.

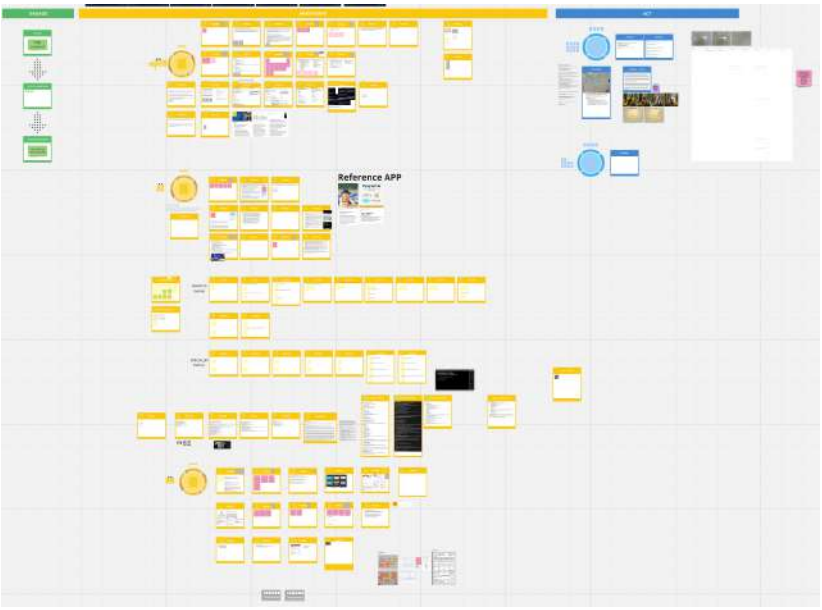
Flog siap menjadi solusi pintar bagi yang ingin menjaga pola makan sehat dan terinformasi setiap hari.

Pseudocode 4.3 Implementasi CoreML

4.4 PukaBoo

PukaBoo adalah sebuah game edukasi yang dirancang khusus untuk membantu anak-anak prasekolah belajar menulis melalui permainan interaktif. Dikembangkan dalam kurun waktu 6 minggu, game ini mengajak anak-anak untuk berinteraksi dengan karakter menarik seperti *Terry The Dinosaur* dan *Trixie The Unicorn*. Anak-anak dapat menyelesaikan tugas harian seperti makan, bermain, dan

minum, di mana setiap tugas memerlukan mereka untuk menelusuri kata-kata dari objek terkait.



Gambar 4.4 Proses *ideation* dan *concepting* aplikasi PukaBoo

Game ini memanfaatkan berbagai *sensory*—auditori, kinestetik, dan visual—untuk menciptakan pengalaman belajar yang imersif. *PukaBoo* terinspirasi dari penelitian tentang *Dysgraphia* serta konsultasi dengan para ahli, dengan tujuan untuk menggabungkan unsur hiburan dan pendidikan secara efektif.

Dalam pengembangan aplikasi *PukaBoo*, digunakan berbagai teknologi berikut:

- SwiftUI: Untuk membangun antarmuka pengguna yang menarik dan mudah digunakan.

- Game Development Concept (UIKit): Memungkinkan pengembangan mekanisme permainan dan animasi yang menarik dan interaktif.
- Apple Pencil (PencilKit): Mendukung aktivitas menelusuri kata-kata dengan presisi dan sensitivitas yang bisa disesuaikan menggunakan Apple Pencil.
- Audio/Video (AVFoundation): Menyediakan suara dan visual yang imersif untuk meningkatkan keterlibatan anak-anak selama bermain dan belajar.
- Gesture Recognition: Untuk mendeteksi gerakan pengguna saat bermain, seperti gerakan menggambar atau memilih objek.
- Computer Graphics (2D, 3D, Animation): Membuat karakter, objek, dan animasi yang menyenangkan untuk mendukung pengalaman bermain yang kaya visual.

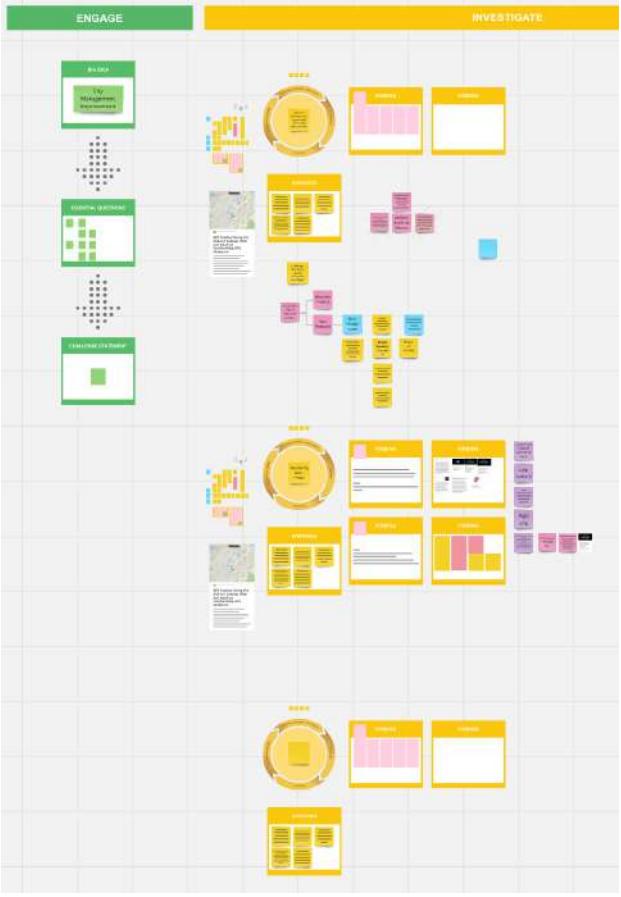
PukaBoo menggabungkan hiburan dengan pendidikan, memberikan anak-anak pengalaman belajar yang seru dan efektif. Dengan pendekatan inovatif ini, anak-anak dapat meningkatkan keterampilan menulis mereka sambil bersenang-senang.

```
import PencilKit

class PracticeViewController: UIViewController, PKCanvasViewDelegate {
    @IBOutlet weak var backgroundCanvasView: PKCanvasView!
    @IBOutlet weak var canvasView: PKCanvasView!

    var difficulty: CGFloat = 7.0 {
        didSet {
            generateText()
        }
    }

    var incorrectStrokeCount = 0

    override func viewDidLoad() {
        super.viewDidLoad()

        canvasView.tool = PKInkingTool(.pen, color: .systemBlue, width: 10)
        canvasView.backgroundColor = .clear
        canvasView.delegate = self

        generateText()
    }

    // MARK: - Text generation

    func generateText() {
        let text = "Example Text".uppercased()
        backgroundCanvasView.drawing = TextGenerator().synthesizeTextDrawing(
            text: text,
            practiceScale: 6.0,
            lineWidth: view.bounds.width
        )
        resetPractice()
    }

    // MARK: - Handwriting matching

    func resetPractice() {
        incorrectStrokeCount = 0
        canvasView.drawing = PKDrawing()
    }

    func canvasViewDrawingDidChange(_ canvasView: PKCanvasView) {
        let testDrawing = backgroundCanvasView.drawing
        let strokeIndex = canvasView.drawing.strokes.count - 1

        guard let lastStroke = canvasView.drawing.strokes.last else { return }
        guard strokeIndex < testDrawing.strokes.count else { return }

        let threshold: CGFloat = difficulty * 6.0
        let distance = lastStroke.discreteFrechetDistance(to:
testDrawing.strokes[strokeIndex], maxThreshold: threshold)

        if distance < threshold {
            canvasView.drawing.strokes[strokeIndex].ink.color = .green
        } else {
            canvasView.drawing.strokes.removeLast()
            incorrectStrokeCount += 1
        }
    }
}
```

Pseudocode 4.4 Implementasi PencilKit

4.5 Robust

Robust adalah aplikasi pelaporan keamanan *all-in-one* yang dirancang untuk mengoptimalkan dan meningkatkan operasional keamanan di CitraLand. Aplikasi ini menggantikan metode pelaporan manual yang tidak efisien dan tidak berkelanjutan dengan proses digital yang lebih terorganisir dan efektif. Dengan *Robust*, data *security* dapat dikelola dengan lebih baik, memberikan wawasan yang dapat ditindaklanjuti untuk meningkatkan keamanan.

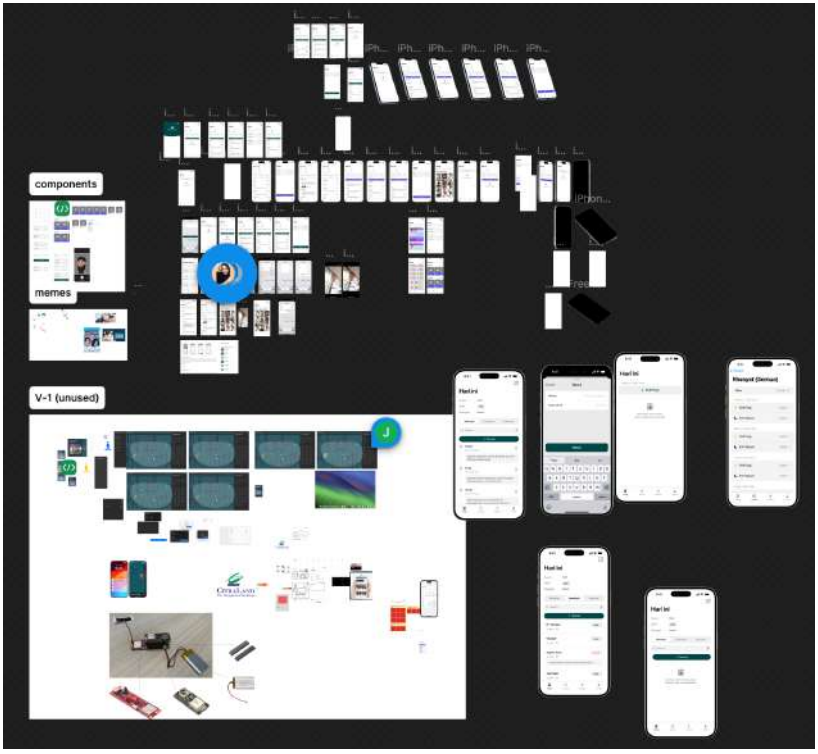


Gambar 4.5.1 Proses *ideation* dan *concepting* aplikasi Robust

Untuk pengembangan selanjutnya, aplikasi *Robust* akan menggunakan teknologi berikut:

- On-Device Data Storage: Untuk menyimpan data laporan secara lokal, menjamin akses cepat dan keamanan informasi.
- SwiftUI: Untuk menciptakan antarmuka yang sederhana, intuitif, dan responsif.
- Voice Assistance: Memungkinkan petugas keamanan membuat laporan menggunakan perintah suara, meningkatkan efisiensi operasional.
- Geo Location (Maps, Beacons): Memberikan pelacakan lokasi dan integrasi peta untuk membantu identifikasi lokasi insiden secara akurat.
- Haptic Feedback: Memberikan pengalaman pengguna yang lebih interaktif dengan umpan balik taktil saat menggunakan aplikasi.
- Custom Camera (AVCam): Untuk dokumentasi visual yang presisi, seperti mengambil foto atau video insiden keamanan secara langsung melalui aplikasi.
- Notification: Memberikan pemberitahuan real-time kepada petugas keamanan terkait laporan baru atau insiden prioritas tinggi.

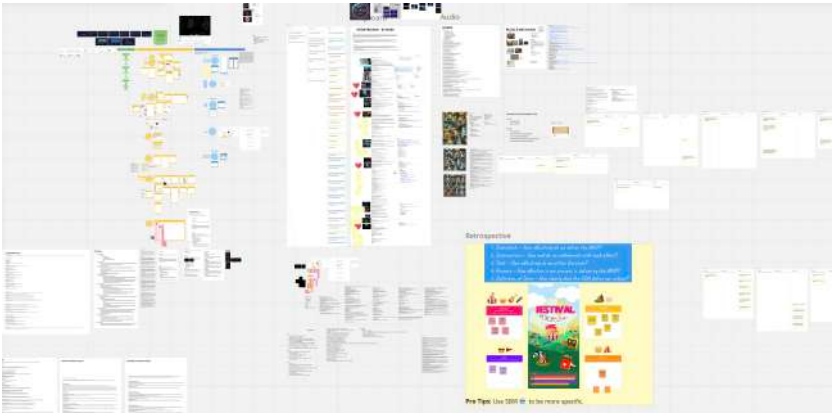
Robust menghadirkan pendekatan modern untuk pelaporan keamanan, memberikan solusi digital yang efisien dan relevan untuk menjaga keamanan operasional di CitraLand. Aplikasi ini tidak hanya membantu petugas keamanan bekerja lebih produktif tetapi juga memastikan setiap data dapat dimanfaatkan untuk analisis dan pengambilan keputusan yang lebih baik.



Gambar 4.5.2 Proses pembuatan *prototype* aplikasi Robust

4.6 Echoes

Echoes mengajak pengguna untuk berperan sebagai seorang detektif *visually impaired* dengan kemampuan langka menggunakan *echolocation*. Dalam game ini, pengguna diberi tugas untuk mengungkap kebenaran dari suatu kasus kematian. Dengan menggali kehidupan korban, mengungkap kebenaran yang tersembunyi, dan menjembatani dunia yang hidup dan yang mati. Pengguna harus bisa menemukan jawaban tiga pertanyaan yaitu pelaku pembunuhan, senjata apa yang digunakan, dan motifnya.

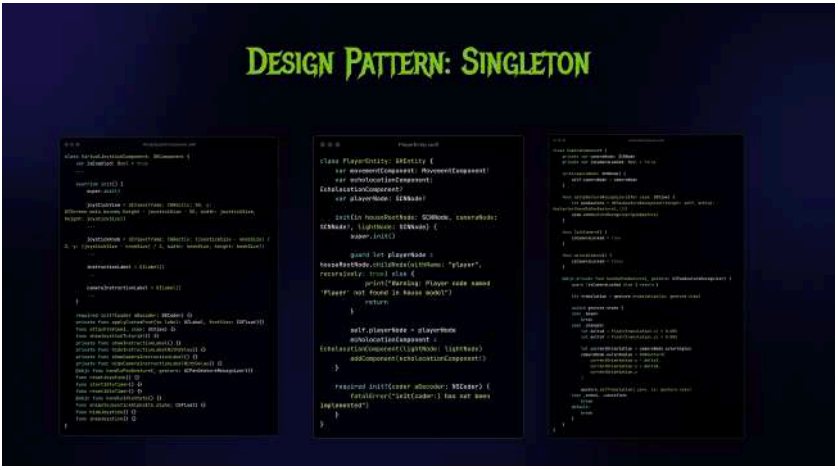


Gambar 4.6 Proses *ideation* dan *concepting* aplikasi Echoes

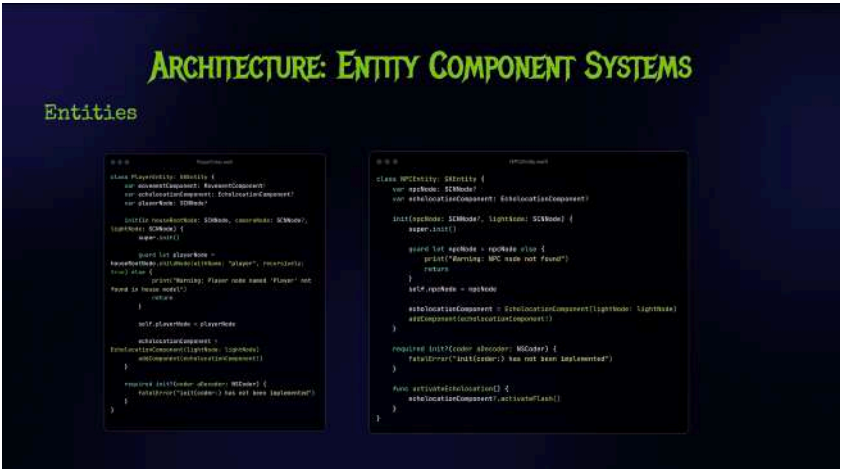
Aplikasi *Echoes* dikembangkan dengan arsitektur ECS (Entity-Component-System) untuk efisiensi dan fleksibilitas dalam pengembangan game. Berikut adalah teknologi yang digunakan:

- Game Development (SceneKit): Untuk menciptakan dunia 3D yang imersif dan detail, mendukung mekanisme *echolocation* yang menjadi inti permainan.
- Audio/Video (AVFoundation): Untuk menghasilkan pengalaman audio yang realistis dan membantu pemain melihat melalui suara.
- Computer Graphics (2D, 3D, Animation): Untuk visualisasi grafis yang memukau dan animasi yang halus, menciptakan atmosfer misteri yang mendalam.
- SwiftUI dan UIKit: Untuk antarmuka pengguna yang interaktif dan mendukung pengalaman bermain yang intuitif.
- GamePlayKit: Untuk manajemen logika game dan sistem mekanika yang kompleks seperti teka-teki dan interaksi pemain.

Pseudocode 4.6.1 Implementasi fitur *echolocation* & *spatial audio*



Pseudocode 4.6.2 Implementasi *design pattern singleton*



Pseudocode 4.6.3 Implementasi arsitektur ECS bagian *entities*

Pseudocode 4.6.4 Implementasi arsitektur ECS bagian *components*

Echoes menghadirkan pengalaman bermain yang imersif, dengan kombinasi cerita yang menarik, mekanisme permainan yang inovatif, dan teknologi Apple terkini. Game ini menguji kemampuan

analisis, kreativitas, dan sensitivitas pemain untuk mengungkap kebenaran di balik misteri yang tersembunyi.

BAB V

PENGUJIAN DAN EVALUASI

Bab berikut ini memaparkan tentang fase pengujian coba serta evaluasi terhadap aplikasi-aplikasi yang telah dikembangkan.

5.1 Tujuan Proses Pengujian

Pengujian pertama-tama dilakukan melalui tahap *Usability Testing*, di mana aplikasi yang telah dikembangkan diuji langsung oleh pengguna nyata. Tujuan dari pengujian ini adalah untuk mengukur kesesuaian dan kenyamanan pengguna dalam menggunakan aplikasi, serta mengumpulkan *feedback* yang berguna untuk mengidentifikasi perbaikan dan pengembangan lebih lanjut. *Usability Testin*g ini juga bertujuan memastikan bahwa aplikasi mampu memenuhi kebutuhan pengguna secara efektif dan memberikan pengalaman yang optimal.

5.2 Kriteria Proses Pengujian

Pengujian aplikasi sendiri penilaiannya didasarkan atas pencapaian tujuan didasarkan pada keberhasilan aplikasi dalam memenuhi *Minimum Viable Product* (MVP) yang telah ditentukan pada tahap awal pengembangan. Kriteria pengujian mencakup kemampuan aplikasi untuk memenuhi kebutuhan pengguna dan memberikan solusi terhadap permasalahan yang dihadapi. Karena *feedback* yang diterima berbentuk kualitatif, evaluasi juga dilakukan berdasarkan respon pengguna terhadap kegunaan, efisiensi, dan kenyamanan aplikasi dalam penggunaan sehari-hari. Selain *Usability Testing*, ada juga sesi *Learning Evidemce Discussion* yang diadakan di setiap akhir pengembangan aplikasi sebagai evaluasi pembelajaran.

5.3 Skenario Pengujian

Gambar 5.3.1 *Idea testing* pertama untuk aplikasi Starred

Target user pertama kami bernama Richard, seorang mahasiswa Informatika ITS yang suka menikmati kegiatan stargazing sendirian, namun merasa lebih menyenangkan jika dilakukan bersama teman. Ia baru melakukan stargazing beberapa hari lalu tanpa peralatan, hanya menggunakan mata telanjang. Lokasi stargazing meliputi tengah jalan dan tempat dengan kondisi terbaik seperti setelah hujan, langit bersih, rendah polusi cahaya, atau menggunakan teleskop. Pengalaman stargazing saat pandemi di Medan, di mana ia menemukan 4 bintang berkedip meski ada polusi cahaya. Tempat terbaik untuk stargazing menurutnya adalah pegunungan yang minim cahaya atau dataran kering seperti gurun pasir karena tidak lembab. Sehingga dapat disimpulkan bahwa lingkungan yang optimal untuk melakukan *stargazing* adalah langit tanpa polusi cahaya atau ketika cahaya sekitar seperti lampu jalan diminimalkan.

Manfaat dan refleksi setelah *stargazing* bagi Richard adalah sebagai berikut. Kepuasan melihat bintang dan pengalaman reflektif. Menikmati keindahan bintang, seperti menghubungkan titik-titik bintang atau menyerupai kebiasaan orang melihat bentuk awan. Maka

dari itu bisa disimpulkan bahwa target pengguna kami adalah orang yang menyukai astronomi atau memiliki ketertarikan filosofis dan sering melakukan *stargazing*.

Gambar 5.3.2 *Idea testing* kedua untuk aplikasi Starred

Target pengguna kedua kami adalah Siddhartha Tiwari, seorang lulusan *computer science* yang bekerja di Astroport Global. Berikut ini tanggapannya terkait dengan masalah utama dalam *stargazing.* Langit harus cukup gelap untuk melihat objek yang jauh. Polusi cahaya dari kota menjadi hambatan utama. Dibutuhkan sekitar 40 menit bagi mata untuk beradaptasi sepenuhnya dengan kegelapan. Awan di langit menjadi penghalang besar dalam *stargazing*. Lokasi terbaik adalah tempat yang jauh dari kota dengan polusi cahaya rendah.

Kemudian untuk pengalaman optimal *stargazing,* bisa dilakukan dengan berikut ini. Resort seperti Astroport Sariska di India menawarkan pengalaman stargazing optimal, termasuk mematikan semua lampu saat sesi dimulai. Prosesnya meliputi mempelajari dasar-dasar langit malam, menggunakan mata telanjang untuk melihat rasi bintang, dan alat seperti teropong atau teleskop untuk objek yang lebih

besar. Dibutuhkan waktu hingga satu bulan untuk meneliti lokasi baru dan 4-5 hari untuk memutuskan apakah lokasi tersebut layak digunakan.

Stargazing memiliki makna budaya yang mendalam di India, sehingga minat terhadap kegiatan ini sangat besar. Banyak orang lebih suka melakukan *stargazing* sendirian karena dianggap sebagai momen penuh kedamaian. Inspirasi pribadi seperti keindahan bulan menjadi motivasi utama bagi beberapa *stargazer*.

Siddharta menyarankan beberapa hal berikut terkait dengan *stargazing*. Fokus pada elemen yang bisa dikontrol, seperti alat dan pengalaman pengguna, daripada memprediksi lokasi terbaik untuk *stargazing* karena terlalu banyak variabel yang memengaruhinya. Misalnya, kualitas kamera atau cuaca. Menjaga kontak dengan Siddharta yang memiliki keahlian dalam mengelola teleskop dan memberikan wawasan tentang *stargazing* menggunakan berbagai alat.

Untuk aplikasi *Starred* sendiri, menurutnya tidak ada aplikasi serupa yang saat ini tersedia, sehingga peluang untuk berinovasi terbuka lebar. Perhatikan batasan teknologi, seperti ketidakmampuan untuk memprediksi lokasi secara akurat.

Gambar 5.3.3 *Idea testing* ketiga untuk aplikasi Starred

Kemudian kami juga melakukan sesi *interview* bersama dengan Claudette Renee Lyons yang merupakan seorang Operatio Manager dari Sedona Stargazing Tours di Arizona, Amerika Serikat. Musim sibuk untuk tur stargazing di Sedona terjadi pada bulan Maret. Setiap tur berdurasi satu jam dan dijadwalkan setiap hari. Dalam satu sesi, enam orang berbagi satu teleskop, dipandu oleh seorang astronom dan seorang asisten. Objek langit yang diamati dipilih berdasarkan belahan bumi yang relevan, dengan 5-6 objek diprioritaskan untuk setiap sesi.

Mayoritas pengunjung tur adalah orang tua muda yang membawa keluarga mereka dan berbagai generasi. Meskipun terdapat variasi usia, kelompok usia utama berada pada rentang awal 30-an hingga awal 40-an. Akibat regulasi COVID-19, proses operasional mengalami banyak perubahan, termasuk pengurangan staf dan modifikasi metode pelaksanaan tur.

Claudette memandang *stargazing* bukan hanya sebagai aktivitas pengamatan objek langit, tetapi juga sebagai pengalaman filosofis yang mendalam. *Stargazing*, menurutnya, membantu individu terhubung dengan alam semesta dan memahami konteks yang lebih besar. Konsep seperti alkimia, spiritualitas, dan fisika kuantum sering ia masukkan dalam pendekatannya untuk menciptakan pengalaman yang lebih bermakna. Claudette percaya bahwa *stargazing* memiliki kekuatan untuk membangkitkan rasa kagum dan keterhubungan yang mendalam dengan kosmos.

Tantangan yang sering dihadapi dalam tur mencakup kerusakan teleskop, gangguan koneksi satelit, dan badai matahari. Hal ini membutuhkan fleksibilitas dan kemampuan beradaptasi untuk memastikan pengalaman pengunjung memiliki impresi yang bagus. Sebagai manajer operasional, Claudette mengelola aktivitas harian dengan baik, tetapi ia tetap menikmati momen *stargazing* sendirian

setiap malam, yang menurutnya merupakan pengalaman pribadi yang sangat berharga.

Secara teknis, para astronom membawa teleskop favorit mereka dan menyesuaikan lensa berdasarkan tujuan pengamatan. Sebagai contoh, pengaturan khusus diperlukan untuk mengamati nebula Orion. Selain itu, Claudette menekankan manfaat kesehatan dari stargazing, yang ia anggap tidak hanya menenangkan secara emosional tetapi juga memperkaya spiritual. Menurutnya, menghabiskan waktu untuk mengamati alam semesta adalah pengalaman yang penting bagi semua orang.

Claudette dan timnya sedang mengembangkan aplikasi untuk meningkatkan pengalaman *stargazing*. Aplikasi ini dirancang untuk memungkinkan pelanggan menjadi astronom mereka sendiri, dilengkapi dengan fitur visualisasi yang lebih mendalam dan pengalaman multi-dimensi seperti teater. Ide ini bertujuan untuk memperkaya pengalaman *stargazing*, menjadikannya lebih interaktif dan bermakna bagi semua kalangan.

Claudette percaya bahwa alam semesta memiliki resonansi organik yang memengaruhi emosi dan pikiran manusia secara mendalam. Filosofinya tentang hidup untuk mencintai kehidupan tercermin dalam dedikasinya terhadap *stargazing*. Baginya, pengalaman di bawah langit malam adalah momen berharga yang menghubungkan manusia dengan alam semesta dan menciptakan rasa syukur terhadap kehidupan.

Selain dari ketiga orang tersebut, setelah sesi *Learning Evidence Discussion* kami juga mendapatkan *feedback* untuk lebih memperhatikan *constraint* usia dan lokasi saran *stargazing* demi keamanan pengguna.

Gambar 5.3.4 *Feedback session* untuk aplikasi PlanetPal

Pada aplikasi kedua ini, saya berhasil mencapai MVP (Minimum Viable Product) dalam hal perhitungan dasar jejak karbon. Namun, saya juga menerima beberapa umpan balik terkait antarmuka aplikasi, terutama mengenai alur aplikasi yang perlu diperhatikan lebih lanjut untuk memastikan pengalaman pengguna yang lebih baik.

Dari sisi desain, saya menyadari bahwa saya sebenarnya telah menerapkan prinsip-prinsip desain yang benar, meskipun awalnya saya tidak tahu bahwa hal-hal tersebut memiliki sebutan khusus. Sekarang, saya memiliki pemahaman yang lebih jelas tentang prinsip-prinsip desain tersebut dan bagaimana cara mengimplementasikannya secara efektif.

Di sisi teknis, saya belajar banyak tentang pengembangan dalam lingkungan SwiftUI. Selain itu, saya mempelajari cara menghitung logika jejak karbon dengan menggunakan SwiftUI, serta memanfaatkan AppStorage dan SwiftData untuk penyimpanan data di perangkat. Saya juga belajar bagaimana cara mengambil data dari API berita dan menampilkan informasi tersebut di aplikasi, serta mengimplementasikan grafik menggunakan *framework* Charts untuk memvisualisasikan data yang didapatkan.

Gambar 5.3.5 *Tech exploration showcase* aplikasi FLog

Pada aplikasi Flog, kami menemukan bahwa untuk mencapai akurasi yang lebih baik dalam mendeteksi komponen makanan menggunakan *machine learning*, dibutuhkan dataset yang lebih besar. Dengan peningkatan dataset, kami dapat meningkatkan akurasi deteksi, yang akan membantu aplikasi lebih efektif dalam mengenali dan mengidentifikasi makanan.

Antarmuka pengguna aplikasi sudah dirancang dengan bersih dan sesuai dengan kebutuhan. Secara keseluruhan, kami berhasil mencapai MVP (Minimum Viable Product) pada beberapa fitur utama aplikasi. Salah satu fitur yang berhasil dikembangkan adalah deteksi dan identifikasi makanan menggunakan kombinasi teknologi Machine Learning, Vision, AVFoundation, dan UIKit di platform iOS. Fitur ini memungkinkan aplikasi untuk mengenali gambar makanan yang diambil oleh pengguna.

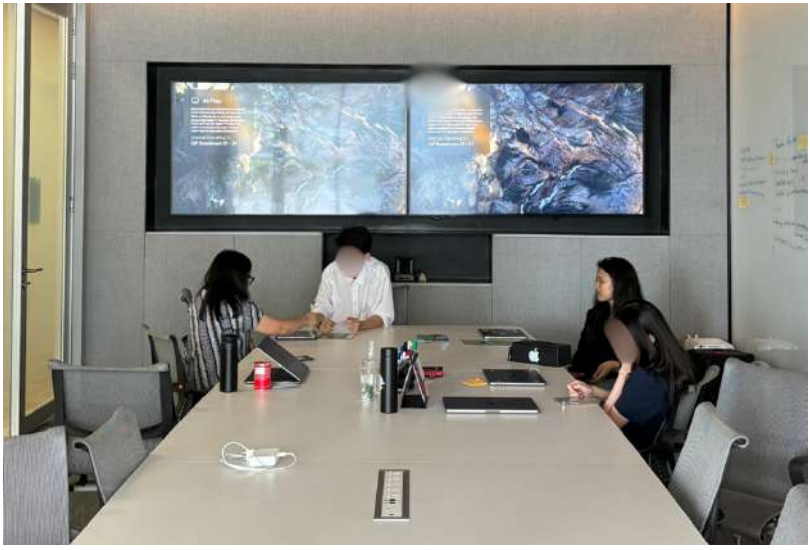
Selain itu, fitur pengingat makan untuk pengguna juga telah berhasil diimplementasikan. Fitur ini menggunakan teknologi Notification pada platform iOS untuk mengingatkan pengguna

mengenai jadwal makan mereka, memberikan pengalaman yang lebih interaktif dan mendukung gaya hidup yang lebih sehat.

Fitur lainnya adalah tampilan informasi nutrisi yang menggunakan SwiftUI pada platform iOS dan WatchOS. Fitur ini memungkinkan pengguna untuk melihat kandungan nutrisi dari makanan yang mereka akan konsumsi, sehingga pengguna akan dapat memilih mana yang lebih tepat tentang asupan gizi mereka.

Terakhir, aplikasi juga menampilkan informasi tentang komposisi makanan melalui SwiftUI pada iOS. Fitur ini memberikan pemahaman yang lebih dalam tentang setiap makanan yang dipilih oleh pengguna, meningkatkan pengetahuan mereka tentang nutrisi dari makanan yang mereka konsumsi.

Dengan semua teknologi ini, aplikasi Flog berhasil mencapai tujuan dasar fungsionalitas yang diinginkan. Namun, untuk meningkatkan kualitas pengalaman pengguna, kami menyadari bahwa akurasi deteksi makanan berbasis *machine learning* perlu diperbaiki lebih lanjut.



Gambar 5.3.6 Proses *usability testing* aplikasi PukaBoo

Gambar 5.3.7 Sesi u*sability testing* aplikasi PukaBoo

Gambar 5.3.8 *Usability testing* iterasi kedua untuk aplikasi PukaBoo

Awalnya, beberapa umpan balik dari pengguna menunjukkan bahwa aplikasi perlu lebih menarik secara visual. Misalnya, menggunakan karakter manusia dengan topi dinosaurus atau tema berbasis gender dapat meningkatkan daya tarik visual bagi anak-anak. Hal tersebut membuat pengalaman bermain sambil belajar menjadi lebih menyenangkan dan dapat membantu memperkenalkan elemen kreatif dalam proses belajar menulis.

Selain itu, ada juga saran agar aplikasi mengenali huruf terlebih dahulu sebelum anak mulai melakukan *tracin*g. Kegiatan tersebut tentunya akan memberikan dasar yang lebih kokoh bagi anak-anak untuk mengenali dan menulis huruf dengan benar, sebelum mereka melanjutkan ke tahap berikutnya dalam permainan. Umpan balik lainnya adalah pentingnya memberi kesempatan kepada anak untuk meninjau cerita yang telah mereka buat. Dengan memberikan akses untuk melihat kembali hasil karya mereka, anak-anak dapat lebih menghargai dan memahami proses penciptaan cerita, serta merasakan kebanggaan atas pencapaian mereka.

Pengguna juga mengusulkan untuk menambahkan efek suara untuk setiap karakter atau saat anak mencapai sesuatu dalam permainan. Efek suara ini dapat memberikan umpan balik positif dan membuat pengalaman belajar menjadi lebih interaktif dan menyenangkan.

Terakhir, ada permintaan agar aplikasi menyediakan kemampuan untuk menghasilkan cerita dengan berbagai hasil atau output. Hal ini akan memberikan lebih banyak variasi dan meningkatkan keterlibatan anak, karena mereka bisa melihat berbagai versi cerita yang berbeda berdasarkan pilihan atau keputusan yang mereka buat selama permainan.

Gambar 5.3.9 *Idea testing* untuk aplikasi Robust

Berdasarkan hasil dari wawancara dengan pihak *security* di CitraLand, proses pelaporan barang hilang di wilayah timur dilakukan dengan mencatat beberapa informasi penting seperti KTP pemilik, waktu kejadian, dan kapan terakhir barang tersebut terlihat. Semua informasi ini kemudian dicatat di buku jurnal yang digunakan untuk dokumentasi lebih lanjut.

Barang yang ditemukan diserahkan kepada petugas keamanan dan harus dalam kondisi yang sama saat dikembalikan. Jika diperlukan, bukti kondisi barang akan dicatat untuk menghindari permasalahan lebih lanjut. Barang yang belum diakui akan disimpan di pos satpam hingga pihak yang kehilangan atau pihak kepolisian mengambilnya. Di pos satpam terdapat loker dan markas utama dengan apel rutin yang dilakukan untuk memastikan semua prosedur berjalan lancar.

Petugas yang memiliki kewajiban dalam pengelolaan barang hilang dan temuan adalah petugas dengan jabatan danru (setara dengan kasiv), yang memimpin sekitar 30 personel. Semua laporan kehilangan dan penemuan barang akan dibahas melalui apel serah terima untuk memastikan kelancaran koordinasi antar personel.

Mereka memiliki grup WhatsApp untuk koordinasi jika ada penemuan barang, sehingga seluruh tim dapat memantau dan memberikan update terkait perkembangan. Semua laporan barang hilang dibuat dan disampaikan kepada koordinator lapangan (korlap) saat apel rutin. Barang yang sering ditemukan di wilayah ini antara lain tas dan jas hujan. Selain itu, meskipun kejadian kecelakaan lalu lintas jarang terjadi, apabila ada kecelakaan, pelaporan dan penanganan dilakukan sesuai dengan prosedur yang berlaku.

Apel serah terima dilakukan setiap pagi dan malam sebagai bagian dari pergantian shift. Patroli juga dilakukan menggunakan kendaraan dinas (marki), dan laporan patroli dikirimkan ke grup WhatsApp untuk koordinasi lebih lanjut. Sistem pelaporan keamanan melibatkan filter berdasarkan wilayah (kasiv), waktu kejadian, pelapor, dan wilayah.

Dari *feedback* tersebut, kami menggagas sebuah app bernama Robust di mana laporan dapat diakses melalui sistem login dengan pengaturan *roles* dan *segmented control* yang memudahkan dalam navigasi antara jurnal atau patroli.

Gambar 5.3.10 *Usability testing* untuk aplikasi Echoes

Untuk iterasi pertama, pengguna diminta untuk memainkan game yang telah kami kembangkan. Berikut merupakan *feedback* yang mereka berikan. Transisi antara *cutscene* terasa tiba-tiba dan kurang mulus, membuat pemain terkejut. Menambahkan penanda perpindahan *scene* seperti yang ada di GTA bisa membantu memandu pemain dengan lebih jelas. Meskipun atmosfer tegang berhasil tercipta, elemen *jumpscare* perlu diperbaiki. *Timer* juga sebaiknya lebih cepat dan intens, agar menciptakan ketegangan yang lebih efektif.

Efek *echolocation* sudah ada, namun suara *echo* terlalu keras. Efek suara di dalam *scene* perlu disesuaikan lagi untuk menciptakan pengalaman yang lebih realistis. Kurangnya penggunaan joystick di beberapa scene, terutama saat dialog, menyebabkan pengguna tetap diam tanpa petunjuk. Interaksi seharusnya lebih intuitif, dengan antarmuka pengguna yang lebih jelas, seperti penanda *proximity* yang lebih baik.

Karakter nenek kurang mengekspresikan emosi. Menambahkan petunjuk atau elemen pengecoh akan meningkatkan plot dan ketegangan. Ketegangan lebih banyak ditentukan oleh suara dan *pacing*, yang perlu diperbaiki. Secara keseluruhan, masalah-masalah ini telah diidentifikasi dan diperbaiki di versi akhir aplikasi setelah melalui beberapa iterasi pengembangan dengan sesi *usability testing* dengan pengguna-pengguna yang lain.

5.4 Evaluasi Pengujian

Echoes Protoype MVP	**Terpenuhi**
Create a landing page with an interactive board for celestial object hunting.	100%
Implement a mechanism to capture constellations by connecting stars.	90%

Randomly generate constellations for players to discover and collect.	90%
Develop a player profile system with basic details like username and stats.	100%
Set up a basic turn-based PvP system with HP and skills.	100%
Build a Stardex to display captured stars and constellations with descriptions and images.	100%
Create a map feature showing landmarks as “Starstops” for exploration.	90%
Add a low light pollution indicator to the map for optimal stargazing locations.	70%
Include a weather indicator with warnings for rain or unfavorable conditions.	70%
Enable map navigation to guide players to Starstops.	85%
Design an inventory system to store collected stars and constellations.	95%
Add an upgrade system to enhance stars using Stardust.	60%
Implement an interaction system where users clear a screen (e.g., wiping clouds) to earn rewards.	100%

Tabel 5.4.1 Evaluasi Pencapaian Aplikasi Starred

PlanetPal MVP	**Terpenuhi**
Interactive Carbon Footprint Calculator	100%
Educational Content	90%
Carbon Offset Suggestions	100%
Local Sustainability News	50%
Data Storage	100%
Clean and Intuitive Interface	80%
Visualizations with Charts	70%

Tabel 5.4.2 Evaluasi Pencapaian Aplikasi PlanetPal

FLog MVP	**Terpenuhi**
Food Detection and Identification	100%
Nutrient Display	100%
User Eat Reminder	100%
Food Composition Database	100%
Integration with SwiftData	50%
Interactive Charts	100%
Clean and User-Friendly Interface	100%
Voice Input for Logging	100%
Dynamic Filtering for Food Search	100%

Tabel 5.4.3 Evaluasi Pencapaian Aplikasi Flog

PukaBoo MVP	**Terpenuhi**
Letter Recognition Interface	100%
Letter Tracing	100%
Interactive Story Review	100%
Engaging Visual Themes	100%
Sound Effects	100%
User Progress Tracking	90%
Gamification Features	100%
Simple and Intuitive UI	100%
Error Correction Assistance	80%

Tabel 5.4.4 Evaluasi Pencapaian Aplikasi PukaBoo

Robust Prototype MVP	**Terpenuhi**
Lost Item Reporting	90%
Security Patrol Reporting	100%
Notification System	70%
Search and Filter Functionality	100%

Role-Based Access Control	90%
Patrol Logs	100%
User Authentication and Login	90%
Incident Reporting	90%
Region-Based Coordination	60%
Shift-Based Reporting	70%
Location-Based Features	60%

Tabel 5.4.5 Evaluasi Pencapaian Aplikasi Robust

Echoes MVP	**Terpenuhi**
Cutscene Integration	100%
Directions Support	100%
Atmosphere and Suspense Features	100%
Echolocation Mechanic	100%
Scene Transition Indicators	100%
Player Blindness Narrative	100%
Timer Adjustment	100%
Clue Integration	100%
Joystick-Free Interaction	90%
Scene-Specific Sound Effects	100%
Enhanced NPC Behavior	90%
Jump Scare Mechanics	100%
Proximity Interaction Enhancements	100%
Player Choices Impact	90%

Tabel 5.4.6 Evaluasi Pencapaian Aplikasi Echoes

BAB VI

KESIMPULAN & SARAN

6.1 Kesimpulan

6.1.1 Starred

Aplikasi ini menunjukkan bahwa pengalaman stargazing memiliki potensi besar untuk dikembangkan menjadi aplikasi interaktif yang memberikan panduan serta pengalaman mendalam kepada pengguna. Melalui wawancara dan observasi, kami memahami bahwa tantangan utama seperti polusi cahaya, cuaca, dan lokasi dapat diminimalisir dengan fitur aplikasi yang inovatif. Penggunaan teknologi untuk visualisasi langit dan prediksi kondisi optimal menjadi fokus utama. Aplikasi ini juga dapat meningkatkan apresiasi terhadap astronomi dan menciptakan pengalaman reflektif yang bermakna.

6.1.2 PlanetPal

Aplikasi PlanetPal berhasil mencapai MVP dengan implementasi perhitungan jejak karbon dasar. Dalam proses ini, saya belajar bahwa pengalaman pengguna dapat ditingkatkan dengan menyederhanakan alur aplikasi dan menyesuaikan desain antarmuka agar lebih intuitif. Penerapan teknologi seperti AppStorage, SwiftData, dan visualisasi data dengan framework Charts menunjukkan bahwa aplikasi ini dapat menjadi alat edukasi yang efektif untuk kesadaran lingkungan.

6.1.3 FLog Project

Pada aplikasi ini, kami berhasil mengembangkan fitur utama seperti deteksi makanan berbasis *machine learning*, pengingat makan, dan informasi nutrisi. Meskipun demikian, tantangan dalam meningkatkan akurasi deteksi makanan menunjukkan kebutuhan akan

dataset yang lebih besar. Dengan kombinasi teknologi seperti SwiftUI, Vision, dan AVFoundation, aplikasi ini memberikan kontribusi dalam mendukung gaya hidup sehat melalui kemudahan identifikasi makanan dan perencanaan nutrisi.

6.1.4 PukaBoo

Aplikasi PukaBoo berhasil mengimplementasikan elemen-elemen gamifikasi untuk pendidikan anak, seperti efek suara, karakter interaktif, dan fitur *tracing* huruf. Namun, feedback dari pengguna menunjukkan bahwa daya tarik visual dan variasi cerita dapat ditingkatkan untuk memberikan pengalaman belajar yang lebih menyenangkan. Di mana di akhir iterasi sudah diperbaiki sesuai dengan *feedback* tersebut. Aplikasi ini memiliki potensi besar untuk memadukan pembelajaran dengan hiburan secara efektif.

6.1.5 Robust

Sistem pelaporan barang hilang di Robust dirancang untuk mempermudah pengelolaan informasi, mulai dari pencatatan barang hingga koordinasi antar personel. Fitur seperti segmentasi laporan berdasarkan wilayah dan penggunaan sistem *roles* memberikan efisiensi dalam operasional. Proses dokumentasi dan koordinasi menggunakan teknologi digital menghilangkan kebutuhan untuk berpindah-pindah aplikasi dalam melakukan tugas keamanan sehingga meningkatkan efisiensi.

6.1.6 Echoes

Aplikasi ini menunjukkan keberhasilan dalam menciptakan atmosfer yang imersif melalui efek suara dan elemen *gameplay*. Namun, transisi antar *scene* dan desain interaksi pemain membutuhkan penyempurnaan untuk meningkatkan pengalaman bermain. Dengan penyesuaian pada *pacing*, desain karakter, dan penggunaan joystick, aplikasi ini memiliki potensi untuk menjadi game interaktif yang lebih

menarik dan mendalam. Yang di mana pada iterasi terakhir, semua masalah tersebut sudah teratasi dan memperoleh respon pengguna yang positif.

6.2 Saran

6.2.1 Starred

Terapkan *constraint* usia dan jarak lokasi *stargazing* dengan jelas untuk menjaga keamanan pengguna dalam bermain. Serta pastikan untuk menerapkan visualisasi yang bisa menjaga ketertarikan penggemar *stargazing* melalui aplikasi meskipun kondisi cuaca sedang tidak optimal untuk melakukan *stargazing.*

6.2.2 PlanetPal

Perbarui algoritma perhitungan jejak karbon untuk mencakup aktivitas yang lebih kompleks. Perbaiki alur navigasi aplikasi agar lebih user-friendly.

6.2.3 FLog

Kembangkan dataset *machine learning* untuk meningkatkan akurasi deteksi makanan. Pertimbangkan untuk menggunakan sensor tambahan lain untuk meningkatkan akurasi ukuran komponen makanan yang akan dikonsumsi pengguna.

6.2.4 PukaBoo

Penataan objek mungkin harus lebih diperhatikan lagi di beberapa ukuran *device* yang berbeda.

6.2.5 Robust

Selain digitalisasi dari proses pencatatan keamanan yang sudah berjalan saat ini untuk Kawasan CitraLand, mungkin bisa ditambahkan inovasi lain di pengembangan selanjutnya.

6.2 6 Echoes

Keseluruhan performa dan suasana game sudah sesuai dengan desain dan konsep awal. Hal yang perlu diperhatikan adalah penggunaan memori yang bisa dibuat lebih efisien lagi dari sisi penggunaan aset 3D.

DAFTAR PUSTAKA

30DayCoding. (2024, June 2). Getting started with building IOS games using SpriteKit and SceneKit. 30 Days Coding - Learn Live. https://30dayscoding.com/blog/building-ios-games-with-spritekit-and-scenekit

Apple. (2024). AVCAM: Building a camera app. Apple Developer Documentation. https://developer.apple.com/documentation/welcome/avcam-building-a-camera-app

Apple. (2024). Avfoundation. Apple Developer. https://developer.apple.com/av-foundation/

Apple. (2024). Core ML - machine learning. Apple Developer. https://developer.apple.com/machine-learning/core-ml/

Apple. (2024, June 11). Deploy machine learning and AI models on-device with core ML - WWDC24 - videos. Apple Developer. https://developer.apple.com/videos/play/wwdc2024/10161/

Apple. (2020, June 23). Inspect, modify, and construct PencilKit drawings - WWDC20 - videos. Apple Developer. https://developer.apple.com/videos/play/wwdc2020/10148/

Apple. (2024d). Mapkit.geocoder. Apple Developer Documentation. https://developer.apple.com/documentation/mapkitjs/mapkit.geocoder

Apple. (2024). Notifications. Apple Developer. https://developer.apple.com/notifications/

Apple. (2017, June 7). Scenekit: What's new - WWDC17 - videos. Apple Developer. https://developer.apple.com/videos/play/wwdc2017/604/

Apple. (2024). Swiftui Overview. Xcode - Apple Developer. https://developer.apple.com/xcode/swiftui/

Apple. (2024, June 12). Unlock the power of places with mapkit - WWDC24 - videos. Apple Developer. https://developer.apple.com/videos/play/wwdc2024/10097/

Apple. (2020, June 23). What's new in pencilkit - WWDC20 - videos. Apple Developer. https://developer.apple.com/videos/play/wwdc2020/10107/

Apple Documentation. (2024). Avfoundation. Apple Developer Documentation. https://developer.apple.com/documentation/avfoundation/

Bao, A. C. (2024, May 20). Swift and coreml: Machine learning magic. Swift and CoreML: Machine Learning Magic - Alibaba Cloud. https://www.alibabacloud.com/tech-news/a/swift/gv9vmmzr5i-swift-and-coreml-machine-learning-magic

Crudu, A. (2024, March 13). Integrating third-party apis with Apple Developer for IOS Apps. MoldStud. https://moldstud.com/articles/p-integrating-third-party-apis-with-apple-developer-for-ios-apps

Ekren, E. K. (2024, May 25). What is Xcode and how to use it?. Digital Acceleration Company.

https://www.netguru.com/blog/what-is-xcode-and-how-to-use-it

Fatbobman. (2024, January 25). Exploring Swiftui property wrappers - @appstorage, @scenestorage, @FocusState, @gesturestate and @scaledmetric: Fatbobman's blog. fatbobman.com. https://fatbobman.com/en/posts/exploring-swiftui-property-wrappers-2/

Fedorenko, E. (2024). Features: Prototyping. Figmalion. https://figmalion.com/topics/prototyping

Figma. (2024a). Beginner 3: Build prototypes. https://help.figma.com/hc/en-us/articles/4405337257751--Beginner-3-Build-prototypes

Figma. (2024). Free prototyping tool: Build interactive prototype designs. https://www.figma.com/prototyping/

Figma. (2024b). Guide to prototyping in Figma – Figma learn - help center. https://help.figma.com/hc/en-us/articles/360040314193-Guide-to-prototyping-in-Figma

Hada, B. (2023, September 20). What is Xcode: Features, installation, and how to use it. Next-Generation Mobile Apps and Cross Browser Testing Cloud. https://www.lambdatest.com/learning-hub/what-is-xcode

Hudson, P. (2023, October 31). How to store swiftdata attributes in an external file. Hacking with Swift. https://www.hackingwithswift.com/quick-start/swiftdata/how-to-store-swiftdata-attributes-in-an-external-file

Jacq. (2021, June 7). Tutorial SWIFTUI: Mengenal basic layout - GITS.ID - jasa pembuatan aplikasi: Software house enterprise: Mobile application developer, google cloud partner. GITS.ID - Jasa Pembuatan Aplikasi | Software House Enterprise | Mobile Application Developer, Google Cloud Partner. https://gits.id/blog/tutorial-swiftui-basic-layout/

Kramer, N. (2024, May 17). Build your first IOS app with UIKit: Beginner's guide. RSS. https://daily.dev/blog/build-your-first-ios-app-with-uikit-beginners-guide

Lagadhir, M. (2023, August 2). Can swift applications integrate with third-party apis?. GTCSYS. https://gtcsys.com/faq/can-swift-applications-integrate-with-third-party-apis/

Lee, A. van der. (2021, August 24). @AppStorage explained and replicated for a better alternative. SwiftLee. https://www.avanderlee.com/swift/appstorage-explained/

Mannotra, V. (2024, November 14). What is Xcode: Features, installation, Uses, Pros & Cons. BrowserStack. https://www.browserstack.com/guide/what-is-xcode

Mattos, P., & Cristina, A. (1962, June 1). Open map in a given address using MapKit and swift. Stack Overflow. https://stackoverflow.com/questions/43918842/open-map-in-a-given-address-using-mapkit-and-swift

Ndungu, D. (2024). Swiftui vs UIKit: The best choice for IOS in 2024. Sendbird. https://sendbird.com/developer/tutorials/swiftui-vs-uikit

Pereira, T. G. (2024, May 9). Storing information using user defaults and @appstorage. Create with Swift.

https://www.createwithswift.com/storing-information-using-user-defaults-appstorage/

Prasatya. (2024, December 16). Core ML UNTUK PEMULA: Membangun aplikasi IOS Yang Pintar dengan machine learning. CODEPOLITAN. https://www.codepolitan.com/blog/core-ml-untuk-pemula-membangun-aplikasi-ios-yang-pintar-dengan-machine-learning/

Ramadhan, G. (2024, December 13). Core ML: Membangun aplikasi IOS berbasis machine learning. Dicoding Blog. https://www.dicoding.com/blog/core-ml-membangun-aplikasi-ios-berbasis-machine-learning/

Reddit. (2024). R/swiftui on reddit: @appstorage question. https://www.reddit.com/r/SwiftUI/comments/181v7ag/appstorage_question/

Siwal, S. (2024, July 10). Swift machine learning: Using Apple Core ML. Bugfender. https://bugfender.com/blog/swift-machine-learning-using-apple-core-ml/

Sulevus, & Mueller 7, H. (1962, April 1). Scenekit Game Architecture. Stack Overflow. https://stackoverflow.com/questions/43081988/scenekit-game-architecture

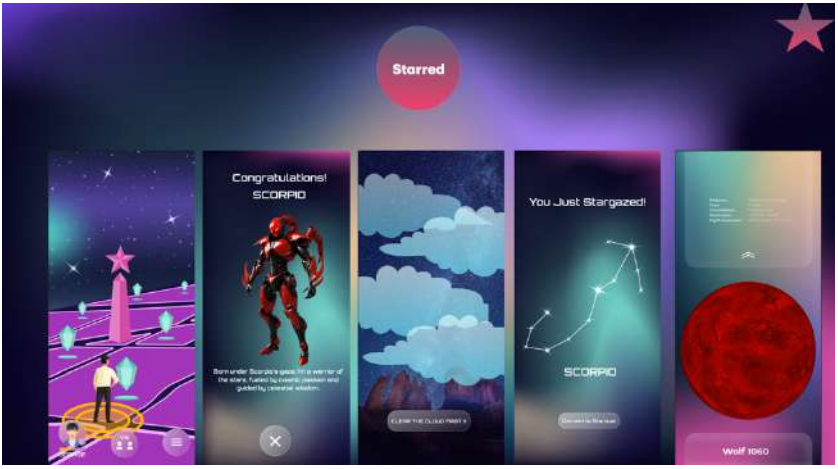
SW Team. (2024, April 7). What is UIKit and what is it for?: SW hosting. SW Hosting's Blog. https://www.swhosting.com/en/blog/what-is-uikit-and-what-is-it-for

Vajpai, S. (2024, November 25). What is Xcode: A best guide. ContextQA. https://contextqa.com/what-is-xcode/

WWDC. (2024, June 10). Swiftui Essentials - WWDC24 - videos. Apple Developer. https://developer.apple.com/videos/play/wwdc2024/10150/

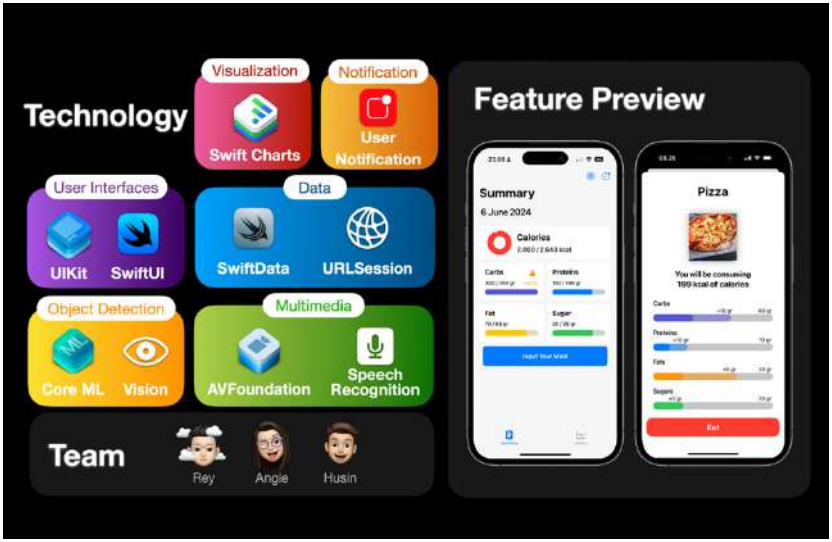
LAMPIRAN

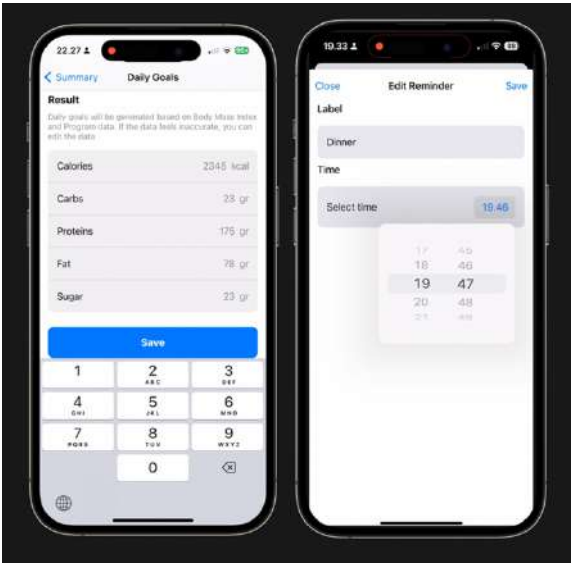
LAMPIRAN 1. Tampilan fitur-fitur dari Starred



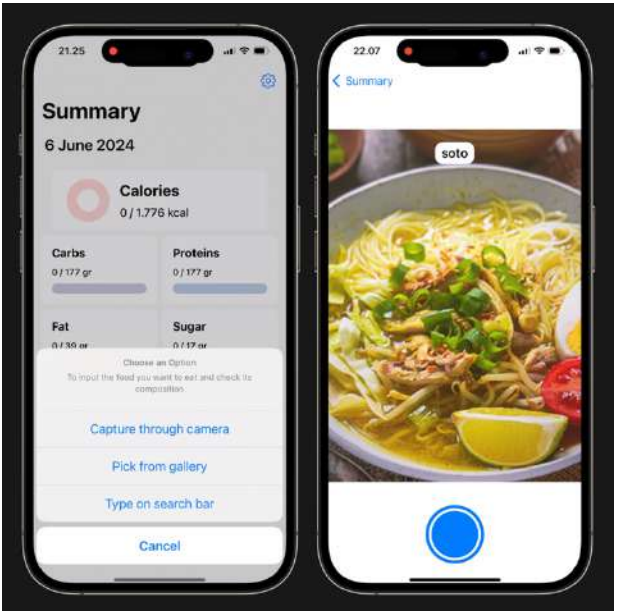
LAMPIRAN 2. Tampilan dari PlanetPal

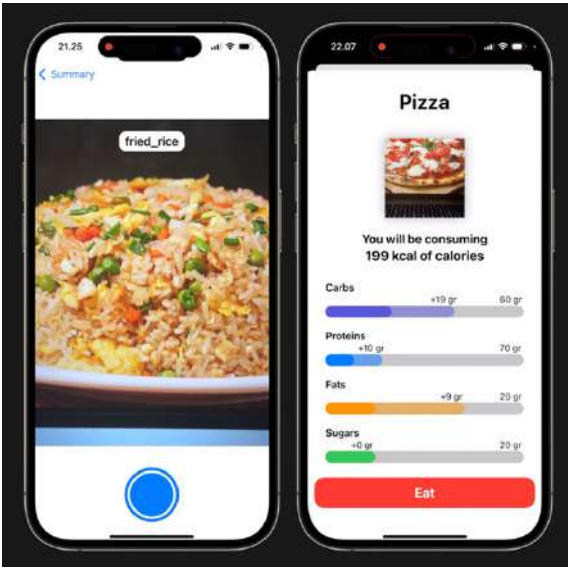
LAMPIRAN 3. Tampilan dari FLog

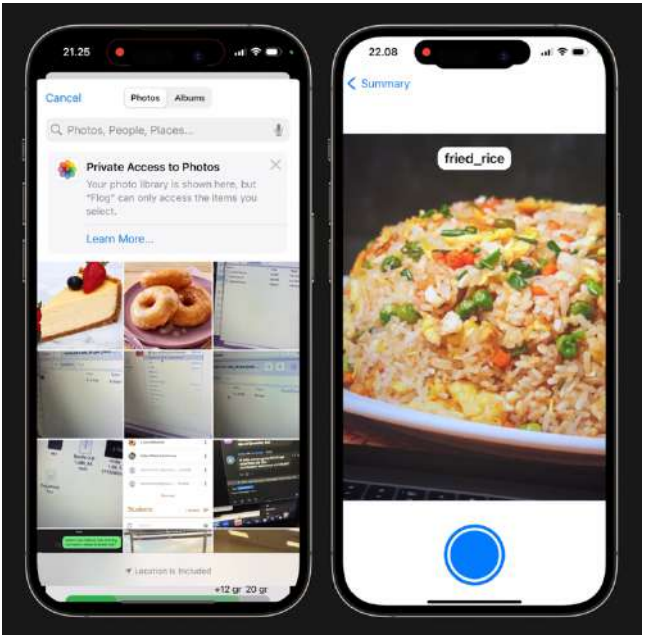

Technology
Visualization
Swift Charts
Notification
User Notification
User Interfaces
UIKit
SwiftUI
Data
SwiftData
URLSession
Object Detection
Core ML
Vision
Multimedia
AVFoundation
Speech Recognition
Team
Rey
Angie
Husin
Feature Preview
Summary
6 June 2024
Calories
Pizza
You will be consuming
1899 kcal of calories

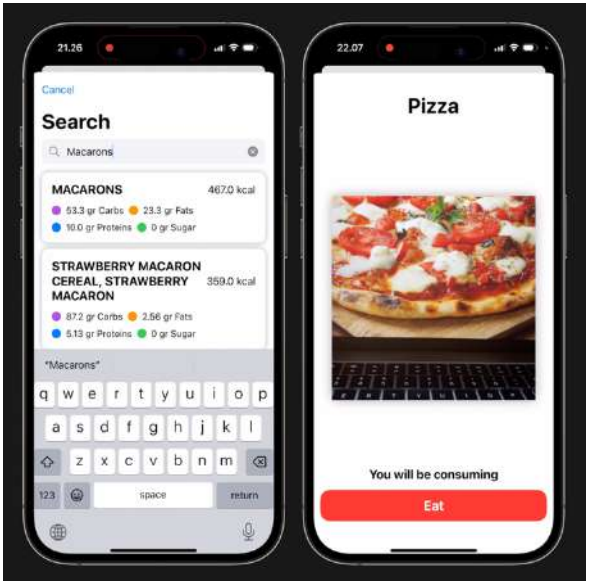

Summary
Daily Goals
Result
Calories 2345 kcal
Carbs 23 gr
Proteins 175 gr
Fat 78 gr
Sugar 23 gr
Save
Close
Edit Reminder
Save
Label
Dinner
Time
Select time 19:46

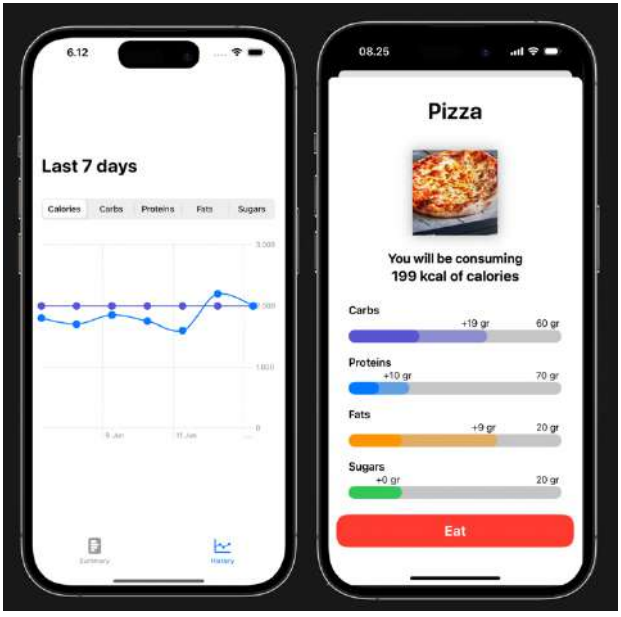

22.27
Summary
Daily Goals
Body Mass Index
Age
Gender
Male
Height
Weight
BMI
Program
19.33
Dinner
It's time to log your calories and essent...
Ecommerce
Finance
Entertainement
Social Media
Productivity
Journal
Google
soundcore
myIM3
Search


21.25
Summary
6 June 2024
Calories
0 / 1.776 kcal
Carbs
0 / 177 gr
Proteins
0 / 177 gr
Fat
Sugar
Choose an Option
To input the food you want to eat and check its composition
Capture through camera
Pick from gallery
Type on search bar
Cancel
22.07
Summary
soto

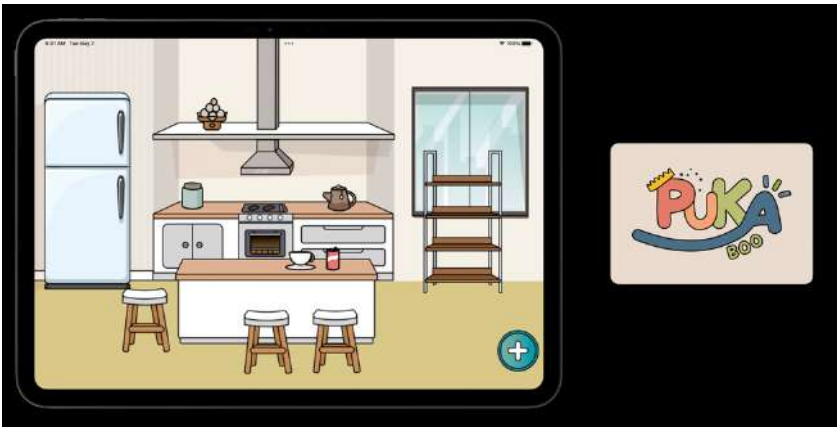

21.25
Summary
fried_rice
22.07
Pizza
You will be consuming
199 kcal of calories
Carbs
+19 gr
60 gr
Proteins
+10 gr
70 gr
Fats
+9 gr
20 gr
Sugars
+0 gr
20 gr
Eat

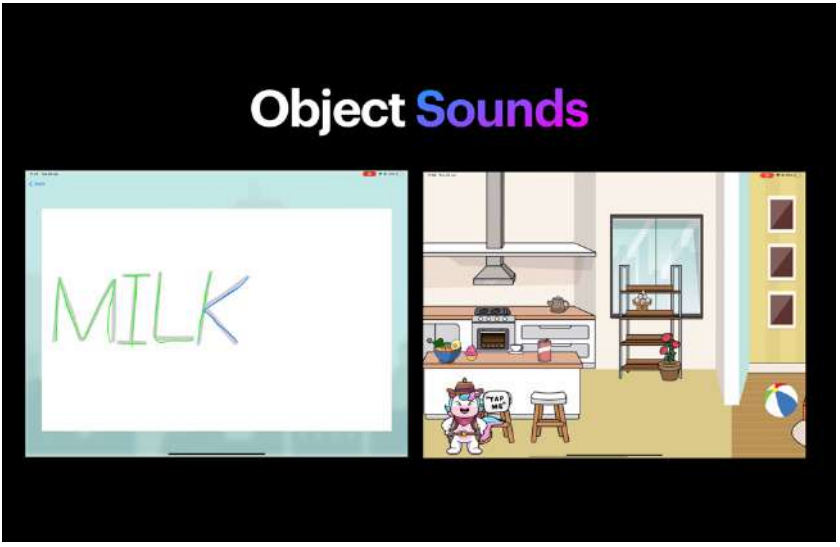

21.25
Cancel
Photos
Albums
Photos, People, Places...
Private Access to Photos
Your photo library is shown here, but "Flog" can only access the items you select.
Learn More...
Location is included
+12 gr 20 gr
22.08
Summary
fried_rice


21.26
Cancel
Search
Macarons
MACARONS
467.0 kcal
53.3 gr Carbs
23.3 gr Fats
10.0 gr Proteins
0 gr Sugar
STRAWBERRY MACARON CEREAL, STRAWBERRY MACARON
359.0 kcal
87.2 gr Carbs
2.56 gr Fats
5.13 gr Proteins
0 gr Sugar
"Macarons"
space
return
22.07
Pizza
You will be consuming
Eat


6.12
Last 7 days
Calories
Carbs
Proteins
Fats
Sugars
08.25
Pizza
You will be consuming
199 kcal of calories
Carbs
+19 gr
60 gr
Proteins
+10 gr
70 gr
Fats
+9 gr
20 gr
Sugars
+0 gr
20 gr
Eat

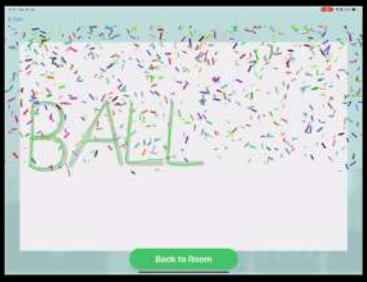
LAMPIRAN 4. Tampilan dari PukaBoo



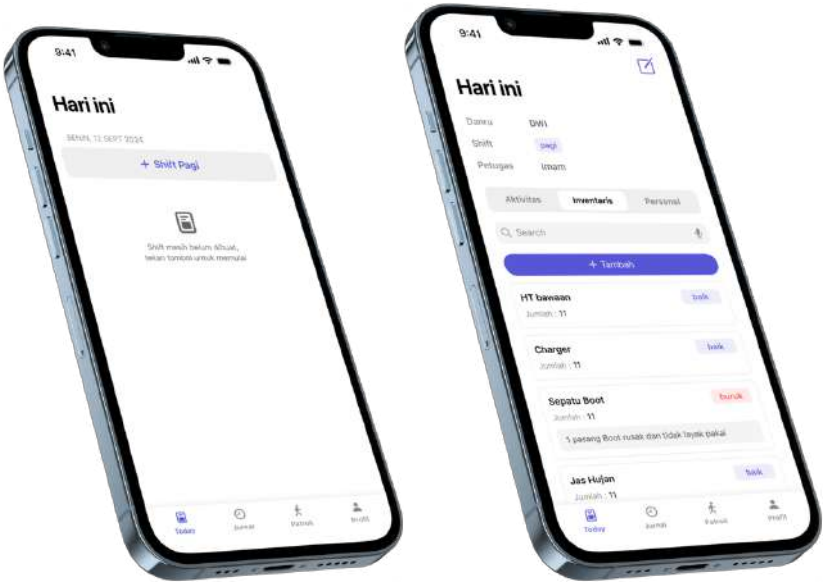


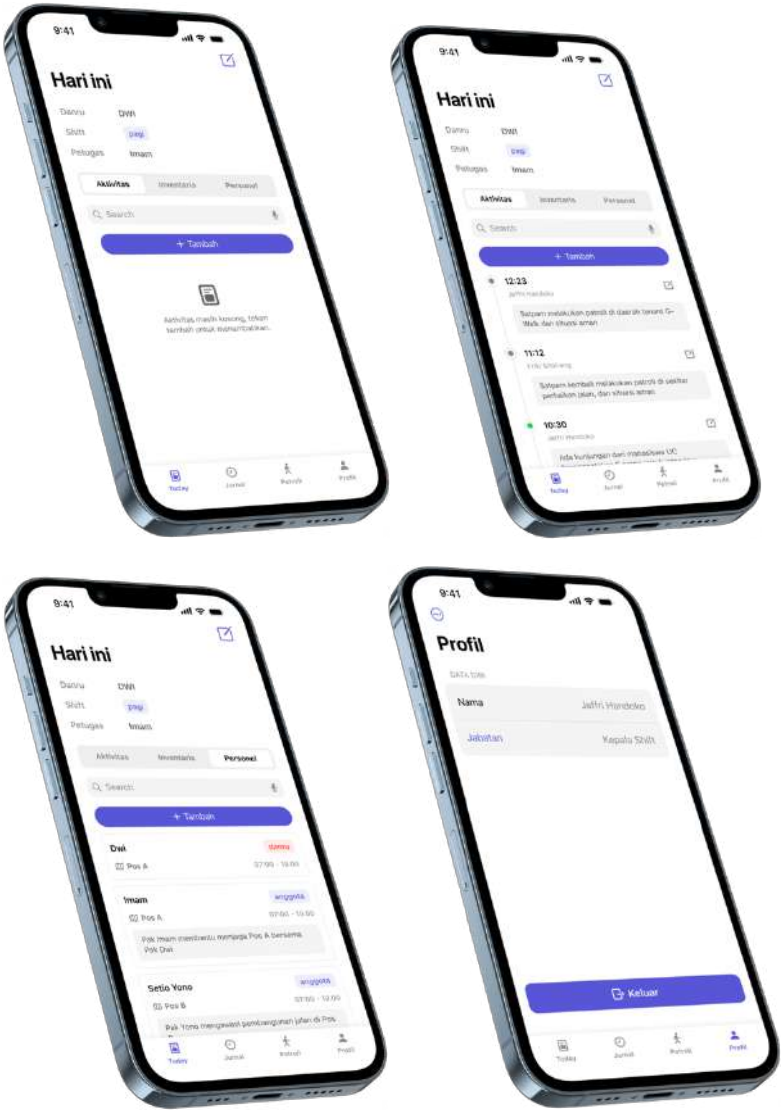
Freedom to Explore

Animated Objects

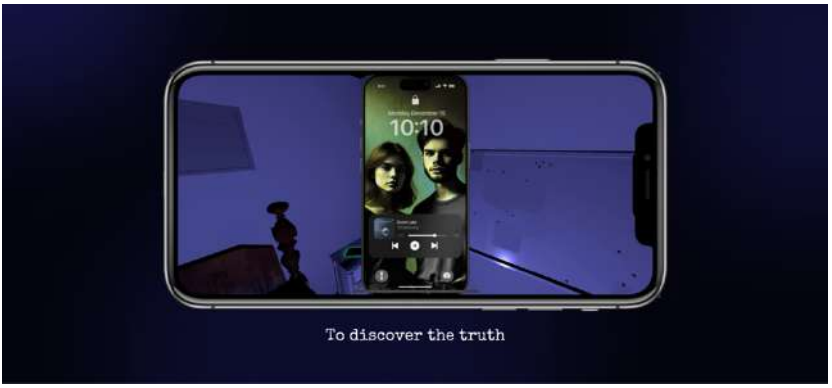


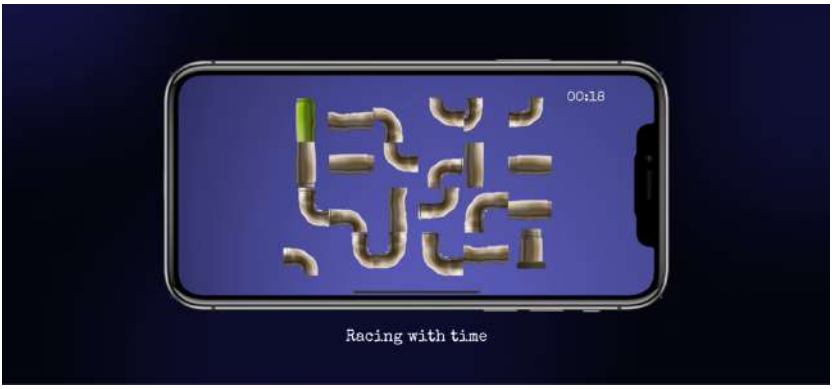
LAMPIRAN 5. Tampilan dari Robust

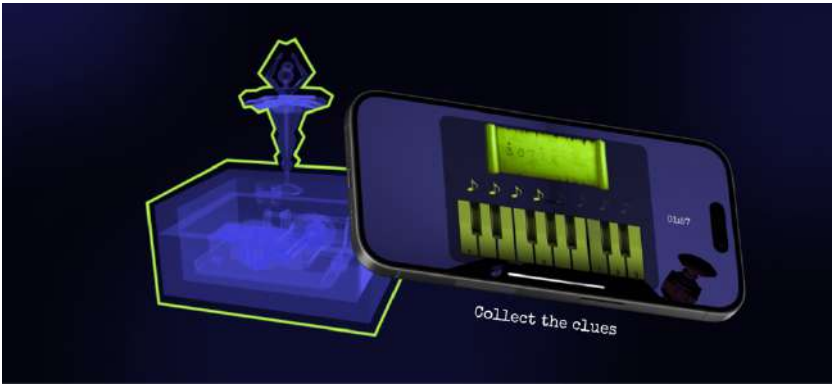



9:41
Hari ini
Danru
Dwi
Shift
pagi
Petugas
Imam
Aktivitas
Inventaris
Personel
Search
+ Tambah
Aktivitas masih kosong, tekan tambah untuk menambahkan.
Today
Jurnal
Patroli
Profil
12:23
11:12
10:30
Profil
DATA TOKO
Nama
Jabatan
Kepala Shift
Keluar

LAMPIRAN 6. Tampilan dari Echoes








Arrange photo
Solve the riddles


Explore


ECHOES
Play
Settings
Credits
Start your investigation

BIODATA PENULIS

Nama	: Pelangi Masita Wati
Tempat, Tanggal Lahir	: Gresik, 18 September 2003
Jenis Kelamin	: Perempuan
Status	: Belum Menikah
Alamat	: Jl. Gebang Kidul No. 41, Surabaya
Telepon	: +6281337504401
Email	: pelangimasita18@gmail.com

PENDIDIKAN FORMAL

2022 – 2026: S1 Teknik Informatika ITS

2019 – 2022: SMAN 6 Surabaya

2016 – 2019: SMPN 6 Gresik

2015 – 2016: SDN 1 Pangkahwetan Gresik

2014 – 2015: SDN 245 Surabaya

2011 – 2014: SDN Gunung Sari III Surabaya

PENGALAMAN

- *Web Programming* (NextJS, Javascript, Tailwind, Shadcn)
- *Programming* (Swift, C++, Python)
- *Database Management* (MySQL, MongoDB)
- *Software* (Figma, VSCode, XCode, Blender, Unity)

- *Operting system* (macOS, Windows, RasbperryPi)
- Bahasa (Indonesia, English)

AKADEMIS

Kuliah : Departemen Informatika – Fakultas Teknologi Elektro dan Informatika Cerdas (FT-EIC)

Angkatan : 2022

Semester : 5 (Lima)

IPK : 3.67