BAB 2
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Dalam membangun proyek ini, terdapat beberapa referensi dari penelitian terdahulu yang menjadi acuan dalam penulisan proyek akhir ini di antara lain: Pertama penelitian yang dilakukan oleh Gamaliel & Arliyanto (2021) dengan judul Perancangan Aplikasi Ujian Online Berbasis Website. Penelitian tersebut bertujuan untuk membahas perancangan aplikasi ujian online berbasis website dengan bahasa pemrograman php dan mysql sebagai basis datanya. Metode yang digunakan menggunakan Extreme Programming. Hasil dari penelitian ini adalah sistem informasi ujian secara online pada perguruan tinggi.

Penelitian kedua dilakukan oleh (Khazanah & Purnama, 2024) dengan judul Rancang Perancangan Sistem Ujian Online Menggunakan Metode Pengembangan Waterfall Berbasis Web. Penelitian ini merancang dan nmenggunakan metode Waterfall. Sistem ini bertujuan menggantikan ujian tradisional berbasis kertas, mengurangi biaya, waktu, dan tenaga, serta meningkatkan efisiensi, keamanan, dan aksesibilitas. Hasil pengujian Black Box menunjukkan 23 dari 24 skenario berhasil, dengan satu kekurangan pada validasi penginputan soal kosong.

Ketiga, menurut penelitian oleh Malik & Amijoyo (2023) dengan judul Sistem Informasi Ujian Online Pada Universitas Saintek Muhammadiyah. penelitian ini memberikan solusi melalui pengkajian beberapa teori dan observasi dilapangan dalam rangka perancangan sistem ujian Online berbasis web. Metode yang digunakan berupa SDLC (System Development Life Cycle) dengan memanfaatkan salah satu modelnya yaitu model waterfall. Hasil yang diperoleh adalah terciptanya sebuah integrasi aplikasi yang dapat mempermudah proses ujian tanpa berkas dan mendapatkan kandidat yang terpilih secara cepat, tepat dan terpercaya.

Keempat, menurut penelitian oleh Yudhistira (2024) dengan judul Perancangan Sistem Informasi Ujian Online Menggunakan Metode Extreme Programming. Penelitian ini merancang dan mengembangkan sistem informasi ujian online yang bertujuan untuk memfasilitasi pelaksanaan ujian secara daring.

Sistem ini dirancang untuk mengatasi masalah yang sering terjadi dalam ujian konvensional, seperti risiko kecurangan, keterbatasan geografis, dan waktu. Sistem ini memungkinkan siswa untuk mengakses dan mengerjakan ujian dari mana saja dengan koneksi internet, serta memfasilitasi pengelolaan ujian secara efisien bagi institusi pendidikan.

Kelima, menurut penelitian Febriansyah & Voutama (2024) dengan judul Rancang Bangun Sistem Ujian Online Berbasis Website Menggunakan Metode Waterfall. Penelitian tersebut bertujuan untuk membuat sistem ujian berbasis online menggunakan media website yang dapat. Hasil akhir dari penelitian ini adalah terciptanya aplikasi ujian online berbasis website yang menawarkan berbagai manfaat seperti efisiensi administrasi, fleksibilitas waktu, dan aksesibilitas bagi siswa untuk mengerjakan ujian dari mana saja.

Tabel 2. 1 Perbandingan Penelitian Terdahulu

Penulis	Metode	Kelebihan	Kekurangan
Gamaliel & Arliyanto (2021	XP	Seluruh fungsi ada telah berhasil sesuai dengan fungsinya.	Belum ada integrasi platform mobile. memerlukan tim pengembang atau sumber daya yang lebih besar
(Khazanah & Purnama, 2024)	Waterfall	Mengurangi biaya, waktu, dan tenaga, serta meningkatkan efisiensi, keamanan, dan aksesibilitas	Hasil pengujian Black Box menunjukkan kekurangan pada validasi penginputan soal kosong.
Malik & Amijoyo (2023)	Waterfall	Sistem informasi ujian Online dapat diterapkan untuk mempermudah pengajar dan pelajar dalam melakukan ujian dan memastikan kesesuaian dengan tingkat validitasnya	Pengujian sistem hanya dilakukan pada 10 mahasiswa, sehingga belum diketahui performa sistem apabila sistem digunakan oleh pengguna dengan skala lebih besar.
Yudhistira (2024)	XP	Hasil pengujian black box menunjukkan 100% keberhasilan, menandakan bahwa semua fitur telah berfungsi	Tidak dijelaskan bagaimana sistem yang ada dapat mengurangi kemungkinan dilakukannya kecurangan dalam mengerjakan ujian
Febriansyah & Voutama (2024)	Waterfall	mengurangi penggunaan kertas dalam ujian, yang	Belum ada fitur untuk mengatasi kecurangan

Penulis	Metode	Kelebihan	Kekurangan
		dapat menghemat biaya dan waktu serta mendukung keberlanjutan lingkungan.	yang dilakukan oleh peserta ujian

Berdasarkan peninjauan terhadap penelitian terdahulu, terdapat beberapa kesenjangan penelitian yang dapat diidentifikasi. Pertama, semua penelitian yang sudah disebutkan sebelumnya hanya berfokus pada pengembangan sistem ujian berbasis website, sementara integrasi dengan platform mobile belum diimplementasikan. Padahal, kebutuhan akan fleksibilitas akses melalui perangkat mobile semakin meningkat, terutama di lingkungan pendidikan. Kedua, metode pengembangan yang digunakan, seperti Extreme Programming (XP) dan Waterfall, cenderung memerlukan tim pengembang atau sumber daya yang lebih besar, sehingga kurang sesuai untuk proyek yang dikelola oleh pengembang tunggal. Ketiga, meskipun beberapa penelitian menyebutkan efisiensi biaya dan waktu, seperti pada penelitian Yudhistira (2024) dan Febriansyah & Voutama (2024), belum ada pendekatan yang secara khusus mengoptimalkan kedua aspek tersebut dengan metodologi yang dirancang untuk pengembang tunggal. Selain itu pada penelitian yang dilakukan oleh Malik & Amijoyo (2023), hanya melakukan pengujian sistem pada skala kecil (10 mahasiswa), sehingga performa sistem dalam skala besar masih belum teruji. Terakhir, aspek keamanan dan pencegahan kecurangan dalam ujian online, seperti yang diungkapkan oleh Yudhistira (2024) dan Febriansyah & Voutama (2024), belum dijelaskan secara mendetail, padahal hal ini sangat penting untuk menjaga integritas ujian.

Proyek akhir ini, berjudul "Rancang Bangun Sistem Ujian Berbasis Mobile dan Website di SMP Negeri 4 Tilatang Kamang dengan Metode RSSD", hadir sebagai solusi untuk mengatasi beberapa kekurangan yang sudah disebutkan sebelumnya. Sistem ini dirancang untuk dapat diakses melalui website dan perangkat mobile, memberikan fleksibilitas bagi siswa dan guru dalam mengelola ujian. Dengan menggunakan metode RSSD (Rapid Solo Software Development), proyek ini memungkinkan pengembangan yang cepat, efisien, dan terfokus tanpa memerlukan tim besar, sehingga sesuai dengan kebutuhan SMP Negeri 4 Tilatang

Kamang yang membutuhkan solusi tepat waktu dengan biaya terbatas. Selain itu, penggunaan framework Laravel dipilih karena kemampuannya dalam mempercepat pengembangan, menyediakan fitur-fitur modern, dan memudahkan pengelolaan basis data serta antarmuka pengguna. Dengan demikian, proyek ini tidak hanya menjawab kebutuhan sistem ujian berbasis daring di SMP Negeri 4 Tilatang Kamang, tetapi juga memberikan kontribusi baru dalam pengembangan sistem ujian dengan pendekatan yang lebih efisien dan terintegrasi.

2.2 Landasan Teori

2.2.1 Ujian

Ujian atau tes adalah sebuah cara dalam pengumpulan informasi, tetapi apabila dikomparasi dengan instrumen lainnya, ujian lebih bersifat formal sebab ada banyak batasan-batasan (Arikunto, 2021). Ujian akan digunakan untuk mengetahui sejauh mana peserta didik telah memahami materi pelajaran yang telah diperoleh sebelumnya. Hasil dari ujian ini nantinya akan menghasilkan sebuah informasi mengenai hasil dari proses pembelajaran.

2.2.2 *Rapid Solo Software Development* (RSSD)

Solo Software Development merujuk pada proses pengembangan perangkat lunak yang dilakukan oleh seorang individu (developer tunggal) yang bertanggung jawab penuh atas seluruh siklus pengembangan, mulai dari pengumpulan persyaratan, desain, pengkodean, pengujian, hingga peluncuran produk. Pendekatan ini sering digunakan oleh pengembang freelance atau start-up yang memiliki tim kecil atau bahkan hanya satu orang, terutama untuk proyek-proyek kecil hingga menengah seperti aplikasi web atau mobile. Pengembang solo menghadapi tantangan seperti keterbatasan sumber daya, motivasi, dan self-efficacy, namun dapat mengatasinya dengan mengadopsi metodologi yang sesuai seperti Personal Software Process (PSP), Personal Extreme Programming (PXP), atau Solo Scrum (Kusuma dkk., 2024).

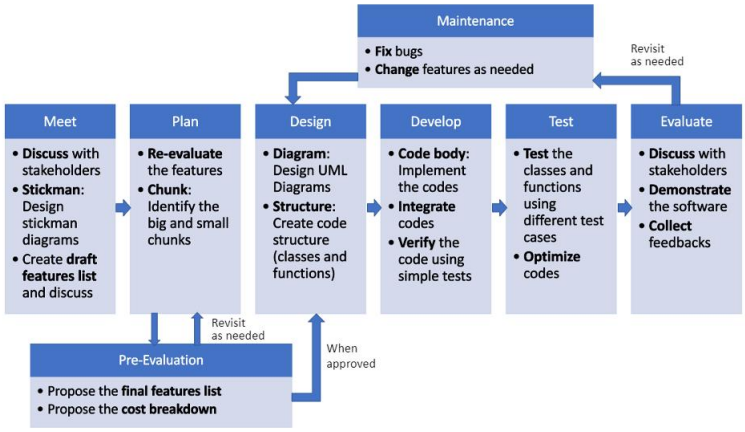
Dalam penyusunan proyek akhir ini, penulis memilih metodologi Rapid Solo Software Development (RSSD) sebagai pendekatan utama dalam proses pengembangan perangkat lunak. Pemilihan RSSD didasarkan pada kebutuhan dan karakteristik proyek yang dikembangkan secara individual (solo development).

Sebagaimana dijelaskan oleh Purba & Ramli (2022), sebagian besar metodologi pengembangan perangkat lunak yang tersedia seperti Agile, Scrum, maupun Waterfall, lebih ditujukan untuk tim pengembang. Padahal, sekitar 60% perusahaan pengembang perangkat lunak memulai aktivitas mereka dengan pengembang tunggal (Pagotto dkk., 2016). Dalam konteks tersebut, RSSD hadir sebagai metodologi yang secara khusus dirancang untuk pengembang tunggal dengan mengedepankan tiga nilai utama yaitu efisiensi, modularitas, dan revisitability (kemudahan untuk ditinjau kembali) (Purba & Ramli, 2022).

Berbeda dengan metodologi Waterfall yang bersifat linier dan tidak fleksibel terhadap perubahan kebutuhan pengguna di tengah proses pengembangan (Boiangiu dkk., 2019). RSSD dirancang dengan pendekatan iteratif seperti Agile, namun lebih disesuaikan untuk kebutuhan individu. Waterfall juga mengasumsikan bahwa seluruh kebutuhan pengguna telah diketahui sejak awal, yang dalam praktiknya sering tidak realistis karena perubahan sangat mungkin terjadi (Cáliz dkk., 2016).

Studi yang dilakukan oleh Purba & Ramli (2022) menunjukkan bahwa penerapan RSSD pada sepuluh proyek yang dikerjakan oleh pengembang individu dari berbagai negara menghasilkan skor rata-rata 4.19 dari 5.0, yang menunjukkan bahwa para responden mengakui efektivitas RSSD dalam menyederhanakan proses pengembangan dan meningkatkan kualitas produk akhir. Selain itu, nilai-nilai utama RSSD seperti efisiensi dalam perencanaan dan pembuatan struktur kode, modularitas untuk meminimalkan kompleksitas, serta kemampuan untuk merevisi dan memahami kode kembali setelah waktu lama sangat membantu dalam konteks proyek jangka panjang maupun proyek akademik.

Dari perbandingan metodologi yang ada, RSSD menjadi pilihan yang paling relevan karena secara eksplisit didesain untuk memenuhi kebutuhan proyek yang dikerjakan secara individu. Dengan struktur fase yang jelas (Meet, Plan and Pre-evaluation, Design, Develop, Test, Evaluate) dan subfase yang terarah, RSSD mampu menjawab tantangan klasik pengembang solo, seperti keterbatasan waktu, perubahan kebutuhan stakeholder, serta manajemen dokumentasi (Purba & Ramli, 2022).



Gambar 2. 1 Metode RSSD

Metode ini memiliki enam tahapan dengan peran yang berbeda-beda selama proses perancangan aplikasi, berikut adalah tahapan-tahapan pada metode RSSD:

1) Meet Phase

Meet Phase adalah tahap awal pertemuan antara pengembang dan klien untuk memastikan kesepahaman tentang kebutuhan proyek. Tahap ini dari tiga sub-tahap: *discuss* (diskusi kebutuhan), *stickman* (pembuatan diagram sederhana untuk memvisualisasikan fitur), dan *list and decide* (penulisan dan konfirmasi fitur yang disepakati). Meet Phase berperan sebagai fondasi yang kokoh untuk memulai proses pengembangan lebih lanjut, meminimalisir kesalahpahaman, dan memastikan keselarasan antara pengembang dan klien.

2) Plan and PreEvaluation Phase

Plan and Pre-Evaluation Phase dimulai dengan mengevaluasi fitur proyek untuk menilai kelayakannya. Pengembang membuat proposal yang mencakup rincian proyek dan biaya, kemudian mengajukannya kepada pemangku kepentingan. Jika perlu, daftar fitur dan diagram dari pertemuan awal dapat dilampirkan. Tujuan fase ini adalah memastikan proyek layak dan mendapatkan persetujuan sebelum melanjutkan.

3) Design Phase

Pada fase ini pengembang akan membuat beberapa diagram utama seperti *ERD*, dan *Use Case Diagram*. Tujuan fase ini adalah merancang kerangka proyek secara sistematis sebelum tahap implementasi

4) Develop Phase

Develop Phase implementasi kode pada sistem yang akan dibuat. Fase ini mencakup tiga sub-tahap berulang: *code body* (menulis kode), *integrate* (menghubungkan fungsi dan kelas), dan *verify* (memeriksa fungsi secara mandiri). Pada sub-tahap *integrate*, pengembang memastikan *output* satu fungsi cocok dengan input fungsi lain. Sub-tahap verify fokus pada pengujian fungsi dengan ketergantungan minimal, sesuai prinsip modular *RSSD*. Tujuannya adalah menghasilkan kode yang terstruktur dan teruji sebelum lanjut ke tahap berikutnya.

5) Test Phase

Test Phase terdiri dari dua sub-tahap, yaitu *test* dan *optimize*. Pada sub-tahap test, pengembang menguji setiap fungsi menggunakan berbagai kasus uji untuk memastikan tidak ada *bug*, terutama ketika fungsi tidak menghasilkan *output* yang diharapkan. Metode *black box testing* lebih dipilih karena lebih praktis dibandingkan pembuktian matematis. Selanjutnya, pada sub-tahap *optimize*, pengembang melakukan optimasi kode, seperti memperbaiki *query SQL* atau mengoptimalkan *array* dan *loop*, untuk meningkatkan performa aplikasi dan memastikan program dapat berjalan lancar seiring pertumbuhan pengguna. Tujuan fase ini adalah memastikan program bebas *bug* dan efisien.

6) Evaluate and Maintenance Phase

Evaluate Phase melibatkan tiga sub-tahap: diskusi dengan pemangku kepentingan, demonstrasi software, dan pengumpulan umpan balik, yang dapat dilakukan dalam beberapa pertemuan. Pada saat yang sama, pengembang juga menjalankan Maintenance Phase untuk memperbaiki bug dan menyesuaikan fitur sesuai kebutuhan. Dalam fase ini, pengembang perlu menjaga hubungan baik dengan pemangku kepentingan dengan menetapkan batasan yang jelas untuk perubahan kecil, sementara perubahan besar harus dibahas karena memerlukan biaya tambahan. Tujuan fase ini adalah memastikan software berfungsi optimal dan memenuhi kebutuhan klien.

2.2.3 *Database*

Database adalah kumpulan data yang terorganisir secara sistematis untuk memudahkan penyimpanan, pengelolaan, dan pengambilan informasi dengan cepat

dan efisien. Dalam konteks sistem manajemen basis data (Database Management System), database merupakan bagian dari perangkat lunak yang membantu pengguna dalam memelihara, mengontrol, dan mengakses data secara praktis dan efisien. DBMS memungkinkan data untuk disimpan, disortir, dimodifikasi, dan diambil sesuai kebutuhan pengguna, serta memberikan keamanan, konsistensi, dan kemudahan berbagi data dalam suatu organisasi (Sabbrina dkk., 2023).

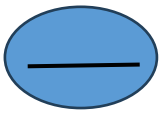
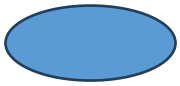
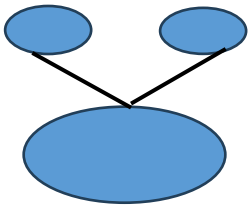
2.2.4 *Entity Relationship Diagram*

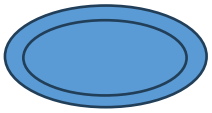
Entity-Relationship Diagram (ERD) adalah representasi grafis dari model data konseptual yang menggambarkan hubungan antara entitas dalam sebuah sistem basis data. Entitas dapat diartikan sebagai objek atau konsep yang dapat dibedakan dan memiliki data yang disimpan dalam basis data. ERD digunakan sebagai tahap awal dalam perancangan database untuk menentukan struktur data yang sesuai dengan kebutuhan pengguna (Pulungan dkk., 2022).

ERD terdiri dari beberapa elemen utama, yaitu:

1) Entitas adalah Objek dalam database yang dapat berupa orang, tempat, benda, atau konsep. Entitas direpresentasikan dengan simbol persegi panjang.
2) Atribut, adalah Karakteristik dari entitas yang menjelaskan informasi penting. Setiap entitas memiliki *primary key* sebagai identitas uniknya. Berikut adalah beberapa jenis atribut yang biasanya digunakan.

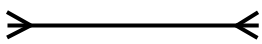
Tabel 2. 2 Jenis Atribut

Nama	Simbol	Pengertian
Atribut Kunci (*Key Attribute*)		Atribut yang digunakan untuk mengidentifikasi setiap entitas secara unik dalam suatu tabel database
Atribut Sederhana (*Simple Attribute*)		Atribut yang terdiri dari satu nilai tunggal dan tidak dapat dipecah lagi.
Atribut Gabungan (*Composite Attribute*)		Atribut yang terdiri dari beberapa bagian yang lebih kecil dan memiliki makna sendiri.

Atribut Multinilai (*Multivalued Attribute*)		Atribut yang dapat memiliki lebih dari satu nilai untuk satu entitas.
Atribut Derivatif (Derived Attribute)		Atribut yang nilainya dapat dihitung atau diturunkan dari atribut lain.

3) Relasi, adalah Hubungan antara dua atau lebih entitas yang menggambarkan keterkaitan data. Relasi digambarkan dengan simbol belah ketupat dan memiliki derajat kardinalitas seperti yang akan dijelaskan pada tabel berikut ini

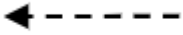
Tabel 2. 3 Relationship

Nama	simbol	pengertian
One-to-one		terjadi ketika setiap entitas cuma mempunyai satu garis relasi saja dengan entitas lainnya
One-to-many		terjadi ketika setiap entitas memiliki banyak hubungan dengan entitas lain, satu-ke-banyak
Many-to-many		terjadi ketika setiap entitas memiliki banyak hubungan dengan banyak entitas lain, banyak-ke-banyak

2.2.5 Use *Case* Diagram

Use Case Diagram adalah salah satu jenis diagram dalam Unified Modeling Language (UML) yang digunakan untuk menggambarkan interaksi antara pengguna (aktor) dan sistem dalam suatu perangkat lunak. Use Case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat (Malius dkk., 2021). Diagram ini membantu dalam memahami bagaimana suatu sistem digunakan dari perspektif pengguna, serta bagaimana sistem merespons berbagai permintaan yang diajukan oleh pengguna. Berikut adalah simbol simbol penting yang ada pada diagram use case.

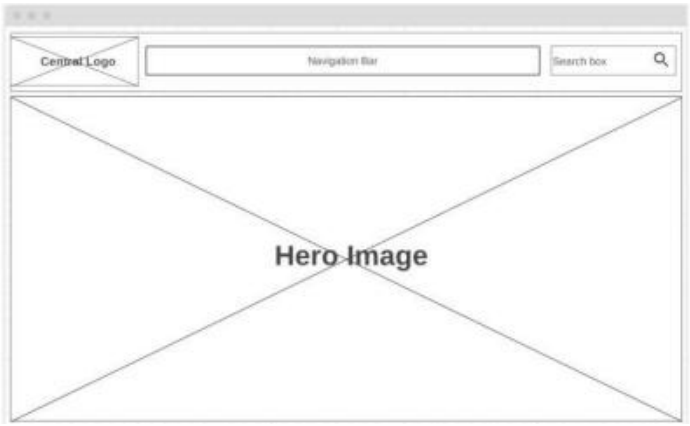
Tabel 2. 4 Simbol pada Use Case Diagram

Nama	simbol	pengertian
Actor		Mewakili pengguna atau sistem eksternal yang berinteraksi dengan sistem
Dependency		Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang tidak mandiri independent
Generalization		Hubungan dimana objek anak (*descendent*) berbagai perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*)
Include		Menspesifikasikan bahwa *use case* sumber secara eksplisit
Extend		Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan
Association		menghubungkan antara objek satu dengan objek lainnya
System		Menspesifikasikan data paket yang menampilkan system secara terbatas

2.2.6 *WireFrame*

WireFrame sendiri dapat diartikan dengan sederhana sebagai kerangka gambar (Salam dkk., 2024). Wireframe digunakan sebagai kerangka awal untuk menggambarkan bagaimana elemen-elemen pada halaman aplikasi atau situs web akan ditempatkan, tanpa mempertimbangkan desain grafis atau warna. Biasanya,

wireframe dibuat dalam bentuk sketsa atau diagram menggunakan garis, kotak, dan ikon sederhana untuk menunjukkan lokasi navigasi, tombol, gambar, serta konten utama. Dalam proses pengembangan sistem, wireframe berfungsi sebagai alat komunikasi antara desainer, pengembang, dan pemangku kepentingan untuk memastikan bahwa semua pihak memiliki pemahaman yang sama mengenai struktur halaman sebelum memasuki tahap desain yang lebih detail. Dengan adanya wireframe, kesalahan desain dapat diminimalkan sejak awal, sehingga mempercepat proses pengembangan dan memastikan pengalaman pengguna (UX) yang optimal.



Gambar 2. 2 Contoh WireFrame (Salam dkk,2004)

Proses pembuatan *wireframe* akan dilakukan dengan menggunakan figma. Figma adalah alat desain antarmuka pengguna berbasis web yang memungkinkan desainer untuk membuat, berbagi, dan berkolaborasi dalam desain aplikasi atau situs web secara langsung. Figma menggunakan pendekatan berbasis *cloud*, yang memungkinkan akses ke proyek desain dari mana saja dan kapan saja, tanpa perlu menginstal perangkat lunak tambahan.

Kelebihan Figma antara lain, pertama, Figma memiliki interface yang intuitif dan sederhana, membuatnya mudah digunakan oleh berbagai jenis pengguna, dari pemula hingga profesional. Kemampuan aplikasi Figma tersebut lah yang membuat aplikasi ini menjadi pilihan banyak UI/UX designer untuk membuat prototype website atau aplikasi dengan waktu yang cepat dan efektif (Hairah dkk., 2024). Kedua, Figma mendukung prototyping, yang memungkinkan untuk membuat

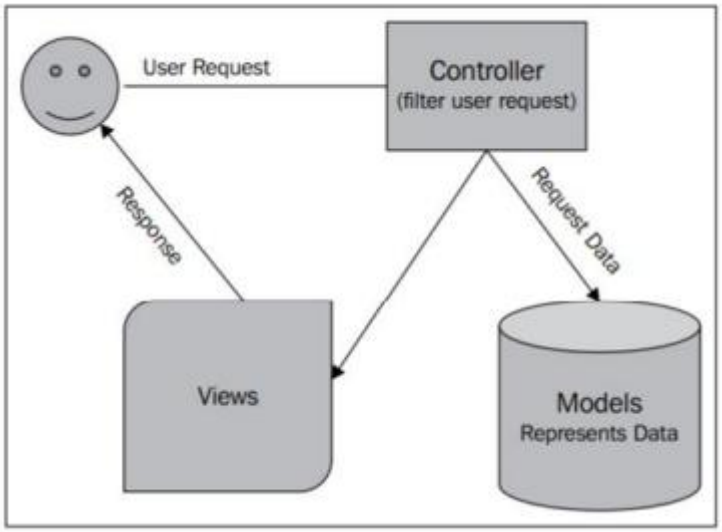
interaksi dan alur aplikasi secara langsung, serta dapat diuji tanpa perlu menulis kode terlebih dahulu.

Dalam proyek rancang bangun sistem ujian online berbasis web dan mobile ini, Figma sangat tepat digunakan karena klien yang berada jauh. Figma akan memungkinkan pengembang untuk berkolaborasi dengan klien secara langsung tanpa harus bertemu secara fisik, sehingga mempermudah komunikasi dan memastikan bahwa desain yang dibuat sesuai dengan harapan klien. Selain itu, dengan kemampuannya untuk berbagi dan berkolaborasi secara online, Figma memungkinkan pengenbang untuk mendapatkan umpan balik instan dari klien, mempercepat proses desain, dan mengurangi kemungkinan miskomunikasi yang biasanya terjadi dalam pengembangan proyek jarak jauh

2.2.7 *Laravel*

Laravel adalah *framework* berbasis bahasa PHP yang bisa digunakan membantu proses pengembangan sebuah *website* agar lebih maksimal. Dengan menggunakan *Laravel, website* yang dihasilkan akan lebih dinamis (Amarulloh, 2023). Dengan sintaks yang efektif dan fitur-fitur canggih, *Laravel* dengan cepat mendapatkan perhatian dan dukungan dari komunitas pengembang yang aktif dalam pengembangan, dokumentasi, serta ekosistemnya.

Laravel didasarkan pada arsitektur MVC yang merupakan sebuah pola desain yang terdiri dari tiga bagian utama, yaitu Model, View, dan Controller. Komponen-komponen ini dirancang untuk menangani aspek-aspek khusus dalam pengembangan aplikasi. Komponen View bertugas untuk menangani logika antarmuka pengguna (user interface), sementara Komponen Controller bertugas menerima data input dan memprosesnya, serta berfungsi sebagai penghubung antara komponen Model dan View. Komponen model merupakan bagian yang berisi logika terkait data pengguna dan menjadi komponen utama dalam pola ini, karena merepresentasikan data yang ditransfer antara View dan Controller (Subecz, 2021).



Gambar 2. 3 Kerangka MVC (Subecz,2021)

Dalam sistem ujian digital ini, *Laravel* digunakan untuk membangun website yang akan digunakan oleh guru dalam mengelola soal ujian. *Laravel* sangat cocok untuk peran ini karena memiliki fitur *Eloquent ORM (Object-Relational-Mapping*) yang disediakan oleh *Laravel* untuk mengakses dan memanipulasi data dalam database menggunakan *PHP Objects* dan model-model terkait. Fitur ini akan memudahkan pengelolaan bank soal, kategori ujian, dan jawaban siswa dalam database secara efisien. Dengan sistem manajemen pengguna dan autentikasi bawaan, *Laravel* memungkinkan setiap guru memiliki akses yang terstruktur sesuai dengan haknya, seperti membuat, mengedit, dan menghapus soal ujian. *Middleware* dalam Laravel juga dapat digunakan untuk mengontrol akses dan validasi data, memastikan bahwa hanya guru yang berwenang yang dapat mengelola soal. *Laravel* menjadi solusi ideal untuk mengembangkan sistem pengelolaan ujian digital yang aman, terstruktur, dan efisien.

2.2.8 *Kotlin*

Kotlin adalah bahasa pemrograman berbasis Java Virtual Machine (JVM). Kotlin merupakan bahasa pemrograman yang pragmatis untuk android yang mengkombinasikan object oriented programming (OOP) dan bahasa fungsional (Febriandirza, 2020).

Kotlin telah menjadi bahasa pemrograman utama bagi lebih dari 50% pengembang Android profesional, menggantikan Java yang hanya digunakan oleh

30% pengembang. Sebanyak 70% pengembang yang menggunakan Kotlin mengaku bahwa bahasa ini meningkatkan produktivitas mereka. Keunggulan Kotlin meliputi kode yang lebih ringkas dan mudah dibaca, mengurangi waktu penulisan dan pemahaman kode. Berdasarkan data internal Google, aplikasi yang dibangun dengan Kotlin 20% lebih kecil kemungkinannya untuk mengalami crash. Kotlin juga didukung penuh oleh library Jetpack, termasuk Jetpack Compose untuk membangun UI modern, serta KTX yang menambahkan fitur Kotlin seperti coroutines dan lambda ke library Android yang sudah ada. Selain itu, Kotlin mendukung pengembangan multiplatform melalui Kotlin Multiplatform, memungkinkan pembuatan aplikasi untuk Android, iOS, backend, dan web. Kotlin telah berkembang sejak 2011 menjadi ekosistem yang matang dengan alat yang kuat, terintegrasi sempurna di Android Studio, dan digunakan oleh banyak perusahaan besar serta startup (Kotlin for Android, 2024). Dukungan komunitas yang kuat dan pembaruan rutin dari JetBrains menjadikan Kotlin pilihan ideal untuk pengembangan aplikasi modern, terutama di lingkungan Android dan multi-platform (Guśpiel, 2024).

Kotlin sangat cocok digunakan untuk membangun aplikasi mobile bagi siswa dalam sistem ujian digital karena memiliki performa yang optimal dan efisien dalam menangani pemrosesan real-time, seperti pengerjaan soal berbasis waktu dan penguncian aplikasi saat ujian berlangsung. Dengan dukungan coroutines, Kotlin dapat mengelola proses asinkron seperti penyimpanan jawaban otomatis, sehingga jawaban siswa tidak hilang meskipun terjadi gangguan jaringan atau aplikasi tertutup secara tidak sengaja. Selain itu, Kotlin memiliki integrasi yang baik dengan Firebase, memungkinkan penggunaan autentikasi yang aman, penyimpanan cloud. Kotlin juga mendukung desain UI yang responsif dan interaktif, memastikan siswa dapat mengerjakan ujian dengan pengalaman yang nyaman dan lancar. Dengan berbagai fitur ini, Kotlin menjadi pilihan ideal untuk mengembangkan aplikasi ujian berbasis mobile yang stabil, aman, dan mudah digunakan oleh siswa.

2.2.9 *Androis Studio*

Android Studio merupakan Lingkungan Pengembangan Perangkat Lunak Tèrpadu - Integrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA. Selain merupakan editor kode IntelliJ

dan alat pengembang yang berdaya guna, Android Studio juga menawarkan banyak fitur untuk meningkatkan produktivitas saat membuat aplikasi android (Saroji dkk., 2023).

Android Studio, alat pengembangan resmi untuk Android, menyediakan lingkungan yang terintegrasi dan lengkap. Ini mencakup fitur desain antarmuka yang memungkinkan pengembang melihat hasilnya secara langsung, alat debugging, dan proses deployment yang mudah. Selain itu, android studio mendukung bahasa pemrograman populer seperti Java dan Kotlin, yang memudahkan pengembang yang sudah familiar dengan bahasa tersebut. Hal ini memungkinkan pengoptimalan aplikasi sesuai kebutuhan. Dukungan dari komunitas pengembang yang besar dan dukungan resmi dari Google membuat android studio menjadi platform yang stabil dan terus berkembang, ideal untuk menciptakan aplikasi yang andal dan berkinerja tinggi (You & Hu, 2021).

2.2.10 Usability Testing

Usability Testing adalah salah satu metode yang berfungsi untuk mengevaluasi tingkat kegunaan suatu *software*, *website*, atau produk yang dibuat (Kurniawan & Yuamita, 2023). Tujuan utama dari usability testing adalah untuk mengidentifikasi masalah yang dihadapi pengguna dalam berinteraksi dengan sistem, serta untuk memahami pengalaman pengguna (user experience) secara keseluruhan. Dalam usability testing, pengguna yang menjadi subjek uji diminta untuk menyelesaikan serangkaian tugas menggunakan produk tersebut, sementara pengujian dilakukan untuk mengamati dan menganalisis interaksi mereka, mengidentifikasi kendala, dan mengumpulkan masukan mengenai antarmuka dan fungsi produk.

2.2.11 *Blackbox Testing*

Blackbox Testing adalah metode pengujian perangkat lunak yang meneliti fungsi (Functional Testing) dari aplikasi tanpa melihat ke dalam struktur aplikasi (Fahrezi dkk., 2022). Pengujian ini berfokus pada masukan yang diberikan kepada sistem dan keluaran yang dihasilkan untuk memastikan bahwa aplikasi bekerja sesuai dengan ekspektasi dan memenuhi persyaratan fungsional.

2.2.12 System Usability Scale (SUS)

System Usability Scale (SUS) merupakan metode pengujian yang diperkenalkan pada tahun 1989 untuk mengukur tingkat kebergunaan (usability) sebuah sistem komputer atau produk berdasarkan sudut pandang subjektif pengguna. Metode yang sering dikenal sebagai pendekatan "quick and dirty" ini berfokus pada pengukuran tiga aspek utama, yaitu efektivitas, efisiensi, dan kepuasan pengguna yang mencakup tingkat kenyamanan serta perilaku positif pengguna terhadap suatu produk.(Ginting dkk., 2025).

Metode SUS memiliki tiga karakteristik utama yang membuatnya sangat efektif untuk pengujian sistem:

1) Instrumen Ringkas: SUS hanya terdiri dari 10 instrumen pertanyaan baku yang dirancang agar relatif mudah dan cepat dipahami serta diselesaikan oleh responden.
2) Technology Agnostic: SUS bersifat fleksibel dan dapat digunakan secara luas untuk mengevaluasi berbagai jenis antarmuka (interface), baik berupa website, aplikasi smartphone, maupun platform teknologi lainnya.
3) Skala Penilaian Universal: Metode ini menghasilkan nilai akhir dengan rentang skor antara 0 hingga 100, sehingga hasilnya relatif mudah diinterpretasikan.

Keunggulan lain dari metode SUS adalah tingkat reliabilitasnya yang tinggi. SUS terbukti sebagai instrumen yang dapat dipercaya dan tetap memberikan hasil yang diandalkan meskipun diterapkan pada ukuran sampel pengujian yang sangat kecil (minimal 2 responden), karena tidak adanya hubungan langsung antara ukuran sampel dengan tingkat reliabilitasnya.

Aturan Penilaian dan Perhitungan Skor SUS Dalam pengaplikasiannya, kuesioner SUS menggunakan skala Likert yang terdiri dari 5 pilihan jawaban untuk mengukur respons pengguna, dengan rincian bobot nilai sebagai berikut:

- Tidak Setuju (TS) = 1 poin
- Kurang Setuju (KS) = 2 poin
- Netral / Ragu-ragu (N) = 3 poin
- Setuju (S) = 4 poin
- Sangat Setuju (SS) = 5 poin

Untuk mendapatkan skor akhir, metode SUS menerapkan aturan perhitungan khusus berdasarkan nomor urut pertanyaan:

- Pada butir pertanyaan bernomor ganjil: Skor dihitung dengan rumus (Skor Jawaban Responden - 1).
- Pada butir pertanyaan bernomor genap: Skor dihitung dengan rumus (5 - Skor Jawaban Responden).

Seluruh skor dari masing-masing pertanyaan kemudian dijumlahkan dan dikalikan dengan 2,5 untuk mendapatkan skor akhir SUS pada skala 0 hingga 100.