**Dokumen Desain Perangkat Lunak (Software Design Document - SDD)**

**Penilian Dan Hasil Kartu Hasil Studi**

**1.Pendahuluan** Dokumen ini menjelaskan desain perangkat lunak untuk sistem penilaian dan hasil kartu hasil studi. Sistem ini dirancang untuk membantu institusi pendidikan dalam mengelola penilaian siswa dan menghasilkan laporan hasil studi.

**1.1 Tujuan** 1. **Meningkatkan Efisiensi Pengelolaan Data  
 2. Menyediakan Akses yang Mudah dan Cepat  
3. Meningkatkan Transparansi  
4. Mendukung Pengambilan Keputusan  
5. Meningkatkan Akurasi Penilaian  
6. Mendukung Proses Pembelajaran yang Berkelanjutan  
7. Memfasilitasi Pelaporan dan Analisis**

#### 1.2 Ruang Lingkup 1. Deskripsi Umum Sistem Sistem Penilaian dan Hasil Kartu Hasil Studi adalah aplikasi berbasis web yang dirancang untuk memfasilitasi pengelolaan penilaian siswa dan menghasilkan laporan hasil studi. Sistem ini akan digunakan oleh berbagai pemangku kepentingan di institusi pendidikan, termasuk admin, guru, siswa, dan orang tua. 2.  ****Fungsi Utama yang Termasuk dalam Ruang Lingkup****

* **Manajemen Pengguna**:
  + Pendaftaran dan pengelolaan akun pengguna (admin, guru, siswa, orang tua).
  + Autentikasi dan otorisasi pengguna berdasarkan peran.
* **Input dan Pengelolaan Nilai**:
  + Fitur bagi guru untuk memasukkan, mengedit, dan menghapus nilai siswa.
  + Pengelolaan mata pelajaran dan pengaturan kriteria penilaian.
* **Pembuatan Kartu Hasil Studi**:
  + Menghasilkan kartu hasil studi secara otomatis berdasarkan nilai yang dimasukkan.
  + Menyediakan format laporan yang dapat dicetak dan diunduh.
* **Akses Hasil Studi**:
  + Menyediakan antarmuka bagi siswa dan orang tua untuk melihat hasil studi.
  + Menyediakan fitur notifikasi untuk memberi tahu siswa dan orang tua tentang hasil studi terbaru.
* **Analisis dan Laporan**:
  + Menyediakan laporan analisis kinerja siswa berdasarkan data penilaian.
  + Fitur untuk mengunduh laporan dalam berbagai format (PDF, Excel).

#### 3. Batasan Sistem

* Sistem ini hanya akan mencakup pengelolaan penilaian dan hasil studi. Fitur lain seperti manajemen kurikulum, kehadiran, dan kegiatan ekstrakurikuler tidak termasuk dalam ruang lingkup ini.
* Sistem akan dirancang untuk digunakan di lingkungan pendidikan formal, seperti sekolah dasar, menengah, dan perguruan tinggi.

**4. Pengguna yang Terlibat**

* **Admin**: Bertanggung jawab untuk mengelola pengguna dan pengaturan sistem.
* **Guru**: Memasukkan dan mengelola nilai siswa.
* **Siswa**: Mengakses dan melihat hasil studi mereka.
* **Orang Tua**: Mengakses hasil studi anak mereka.

**5. Keterbatasan Teknologi**

* Sistem ini akan dibangun menggunakan teknologi web modern (HTML, CSS, JavaScript, dan framework backend seperti Node.js atau PHP).
* Keterbatasan dalam hal perangkat keras dan perangkat lunak yang digunakan oleh pengguna akhir (misalnya, akses internet yang stabil).

**6. Kriteria Keberhasilan**

* Sistem dapat diakses oleh semua pengguna yang berwenang dengan antarmuka yang ramah pengguna.
* Nilai yang dimasukkan oleh guru dapat disimpan dan ditampilkan dengan akurat.
* Kartu hasil studi dapat dihasilkan dan diunduh tanpa kesalahan.
* Pengguna dapat dengan mudah mengakses informasi yang relevan sesuai dengan peran mereka.

**1.3 Referensi**

* + Sommerville, I. (2011). *Software Engineering* (9th ed.). Addison-Wesley.  
    Buku ini memberikan panduan komprehensif tentang prinsip-prinsip rekayasa perangkat lunak, termasuk dokumentasi desain perangkat lunak.
  + Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw-Hill.  
    Buku ini membahas berbagai aspek pengembangan perangkat lunak, termasuk analisis kebutuhan, desain, dan pengujian.

**2. Desain Arsitektur  
 2.1 Diagram arsitektuk**

**1. Komponen Utama**

* **Pengguna**:
  + Admin
  + Guru
  + Siswa
  + Orang Tua
* **Antarmuka Pengguna (UI)**:
  + Halaman Login
  + Halaman Input Nilai (Guru)
  + Halaman Kartu Hasil Studi (Siswa/Orang Tua)
  + Halaman Laporan (Admin/Guru)
* **Server Aplikasi**:
  + Logika Bisnis
  + Pengelolaan Data
  + Autentikasi dan Otorisasi
* **Database**:
  + Tabel Pengguna
  + Tabel Nilai
  + Tabel Kartu Hasil Studi

#### 2. ****Diagram Arsitektur****

Berikut adalah representasi diagram arsitektur dalam bentuk teks. Anda dapat menggambarkannya menggunakan alat diagram seperti Lucidchart, Draw.io, atau Microsoft Visio.

1. **2.2 Komponen utama sistem  
     
    Pengguna**
   * **Admin**:
     + Bertanggung jawab untuk mengelola sistem, termasuk pendaftaran pengguna, pengaturan hak akses, dan pemeliharaan data.
     + Memiliki akses penuh untuk melihat dan mengelola semua data dalam sistem.
   * **Guru**:
     + Memasukkan, mengedit, dan menghapus nilai siswa.
     + Mengelola mata pelajaran dan kriteria penilaian.
     + Menghasilkan laporan hasil studi untuk siswa.
   * **Siswa**:
     + Mengakses dan melihat hasil studi mereka.
     + Melihat nilai dan umpan balik dari guru.
   * **Orang Tua**:
     + Mengakses hasil studi anak mereka.
     + Melihat laporan dan perkembangan akademis anak.
2. **Antarmuka Pengguna (UI)**
   * **Halaman Login**:
     + Formulir untuk autentikasi pengguna (username dan password).
     + Menyediakan akses ke fitur sesuai dengan peran pengguna.
   * **Halaman Input Nilai (Guru)**:
     + Antarmuka untuk guru memasukkan dan mengelola nilai siswa.
     + Menyediakan tabel untuk memasukkan nilai berdasarkan mata pelajaran.
   * **Halaman Kartu Hasil Studi (Siswa/Orang Tua)**:
     + Menampilkan hasil studi siswa dalam format yang mudah dibaca.
     + Menyediakan opsi untuk mengunduh atau mencetak kartu hasil studi.
   * **Halaman Laporan (Admin/Guru)**:
     + Menyediakan analisis kinerja siswa dan laporan yang dapat disesuaikan.
     + Menampilkan statistik dan grafik terkait hasil studi.
3. **Server Aplikasi**
   * **Logika Bisnis**:
     + Mengelola alur kerja aplikasi, termasuk pengolahan data penilaian dan pembuatan laporan.
     + Menangani permintaan dari antarmuka pengguna dan mengembalikan respons yang sesuai.
   * **Pengelolaan Data**:
     + Mengelola interaksi dengan database, termasuk penyimpanan, pengambilan, dan pembaruan data.
   * **Autentikasi dan Otorisasi**:
     + Memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses fitur tertentu berdasarkan peran mereka.
4. **Database**
   * **Tabel Pengguna**:
     + Menyimpan informasi tentang semua pengguna sistem, termasuk username, password (ter-enkripsi), dan peran.
   * **Tabel Nilai**:
     + Menyimpan data penilaian siswa, termasuk mata pelajaran, nilai, dan ID siswa yang terkait.
   * **Tabel Kartu Hasil Studi**:
     + Menyimpan informasi tentang kartu hasil studi, termasuk semester, tahun ajaran, dan rata-rata nilai siswa.
5. **Fitur Tambahan**
   * **Notifikasi**:
     + Mengirimkan pemberitahuan kepada siswa dan orang tua tentang hasil studi terbaru atau pembaruan penting.
   * **Analisis dan Laporan**:
     + Menyediakan fitur untuk menghasilkan laporan analisis kinerja siswa berdasarkan data penilaian.
     + Menyediakan grafik dan statistik untuk membantu guru dan admin dalam evaluasi.

#### 3. Desain Modul Dan Komponen 3.1 Modul Auntentifikasi Pengguna 1. Deskripsi Umum

Modul autentikasi pengguna bertanggung jawab untuk mengelola proses masuk (login) dan keluar (logout) pengguna ke dalam sistem. Modul ini memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses fitur dan data yang sesuai berdasarkan peran mereka (admin, guru, siswa, orang tua).

**2. Fungsi Utama**

* **Pendaftaran Pengguna**:
  + Memungkinkan pengguna baru untuk mendaftar dan membuat akun.
  + Mengumpulkan informasi yang diperlukan seperti username, password, dan peran.
* **Login Pengguna**:
  + Memverifikasi kredensial pengguna (username dan password) saat pengguna mencoba untuk masuk ke sistem.
  + Menggunakan metode enkripsi untuk menyimpan dan memverifikasi password.
* **Logout Pengguna**:
  + Menghapus sesi pengguna dan mengarahkan pengguna kembali ke halaman login.
* **Pengelolaan Sesi**:
  + Mengelola sesi pengguna untuk memastikan bahwa pengguna tetap terautentikasi selama mereka menggunakan sistem.
  + Mengatur waktu kedaluwarsa sesi untuk meningkatkan keamanan.
* **Pemulihan Password**:
  + Menyediakan opsi bagi pengguna untuk mereset password jika mereka lupa.
  + Mengirimkan email pemulihan dengan tautan untuk mengatur ulang password.

**3. Arsitektur Modul**

Modul autentikasi pengguna akan terdiri dari beberapa komponen utama:

* **Antarmuka Pengguna**:
  + Formulir pendaftaran dan login.
  + Halaman untuk pemulihan password.
* **Logika Bisnis**:
  + Proses untuk memverifikasi kredensial pengguna.
  + Proses untuk menyimpan dan mengambil data pengguna dari database.
* **Database**:
  + Tabel pengguna yang menyimpan informasi pengguna, termasuk username, password (ter-enkripsi), dan peran.

**4. Desain Database**

Tabel pengguna dalam database akan memiliki struktur sebagai berikut:

| **Kolom** | **Tipe Data** | **Deskripsi** |
| --- | --- | --- |
| user\_id | INT (PK) | ID unik untuk setiap pengguna |
| username | VARCHAR(50) | Nama pengguna yang digunakan untuk login |
| password | VARCHAR(255) | Password yang ter-enkripsi |
| role | ENUM | Peran pengguna (admin, guru, siswa, orang tua) |
| email | VARCHAR(100) | Alamat email untuk pemulihan password |
| created\_at | DATETIME | Tanggal dan waktu pendaftaran pengguna |
| updated\_at | DATETIME | Tanggal dan waktu terakhir diperbarui |

**5. Proses Autentikasi**

1. **Pendaftaran**:
   * Pengguna mengisi formulir pendaftaran.
   * Sistem memverifikasi bahwa username dan email belum terdaftar.
   * Password dienkripsi sebelum disimpan di database.
   * Pengguna menerima konfirmasi pendaftaran.
2. **Login**:
   * Pengguna memasukkan username dan password.
   * Sistem memverifikasi kredensial dengan membandingkan password yang dimasukkan dengan password yang ter-enkripsi di database.
   * Jika berhasil, sesi pengguna dibuat dan pengguna diarahkan ke halaman utama sesuai peran mereka.
3. **Logout**:
   * Pengguna mengklik tombol logout.
   * Sistem menghapus sesi pengguna dan mengarahkan kembali ke halaman login.
4. **Pemulihan Password**:
   * Pengguna mengisi alamat email untuk pemulihan.
   * Sistem mengirimkan email dengan tautan untuk mengatur ulang password.
   * Pengguna mengklik tautan dan memasukkan password baru yang akan dienkripsi dan disimpan.

**6. Keamanan**

* **Enkripsi Password**: Menggunakan algoritma hashing yang kuat (seperti bcrypt) untuk menyimpan password dengan aman.
* **Validasi Input**: Memastikan bahwa semua input pengguna divalidasi untuk mencegah serangan seperti SQL Injection dan Cross-Site Scripting (XSS).
* **Pengaturan Sesi**: Menggunakan token sesi yang aman dan mengatur waktu kedaluwarsa untuk sesi pengguna.

**7. Pengujian Modul**

* **Pengujian Unit**: Menguji setiap fungsi dalam modul secara terpisah untuk memastikan bahwa mereka berfungsi dengan benar.
* **Pengujian Integrasi**: Menguji interaksi antara modul autentikasi dan modul lain dalam sistem.
* **Pengujian Keamanan**: Melakukan pengujian penetrasi untuk mengidentifikasi dan memperbaiki kerentanan

#### 3.2 Modul menejemen data  Deskripsi Umum

Modul manajemen data bertanggung jawab untuk mengelola semua data yang terkait dengan penilaian siswa, termasuk input nilai, pengelolaan mata pelajaran, dan pembuatan laporan hasil studi. Modul ini memastikan bahwa data disimpan, diambil, dan dikelola dengan cara yang efisien dan aman.

**2. Fungsi Utama**

* **Input Nilai**:
  + Memungkinkan guru untuk memasukkan, mengedit, dan menghapus nilai siswa untuk berbagai mata pelajaran.
  + Menyediakan antarmuka yang mudah digunakan untuk memasukkan nilai secara batch atau individu.
* **Pengelolaan Mata Pelajaran**:
  + Memungkinkan admin dan guru untuk menambah, mengedit, dan menghapus mata pelajaran yang tersedia dalam sistem.
  + Mengelola kurikulum dan kriteria penilaian untuk setiap mata pelajaran.
* **Pembuatan Kartu Hasil Studi**:
  + Menghasilkan kartu hasil studi secara otomatis berdasarkan nilai yang dimasukkan.
  + Menyediakan format laporan yang dapat dicetak dan diunduh oleh siswa dan orang tua.
* **Analisis Data**:
  + Menyediakan fitur untuk menganalisis kinerja siswa berdasarkan data penilaian.
  + Menghasilkan laporan statistik dan grafik untuk membantu guru dan admin dalam evaluasi.

**3. Arsitektur Modul**

Modul manajemen data akan terdiri dari beberapa komponen utama:

* **Antarmuka Pengguna**:
  + Halaman untuk input nilai.
  + Halaman untuk pengelolaan mata pelajaran.
  + Halaman untuk melihat dan mengunduh kartu hasil studi.
* **Logika Bisnis**:
  + Proses untuk menyimpan, mengambil, dan memperbarui data penilaian dan mata pelajaran.
  + Proses untuk menghasilkan laporan hasil studi.
* **Database**:
  + Tabel untuk menyimpan data nilai, mata pelajaran, dan kartu hasil studi.

#### 3.3 Modul laporan dan analitik 1. Deskripsi Umum

Modul laporan dan analitik bertanggung jawab untuk menghasilkan laporan yang berkaitan dengan kinerja siswa, analisis data penilaian, dan pembuatan kartu hasil studi. Modul ini memberikan wawasan yang berguna bagi guru, admin, siswa, dan orang tua untuk memahami perkembangan akademis dan membuat keputusan yang lebih baik.

**2. Fungsi Utama**

* **Pembuatan Laporan Kartu Hasil Studi**:
  + Menghasilkan kartu hasil studi untuk siswa berdasarkan nilai yang telah dimasukkan.
  + Menyediakan format yang dapat dicetak dan diunduh dalam berbagai format (PDF, Excel).
* **Analisis Kinerja Siswa**:
  + Menyediakan analisis mendalam tentang kinerja siswa berdasarkan data penilaian.
  + Menghasilkan grafik dan statistik yang menunjukkan tren kinerja siswa dari waktu ke waktu.
* **Laporan Statistik**:
  + Menghasilkan laporan statistik yang mencakup rata-rata nilai, distribusi nilai, dan perbandingan antar siswa atau kelas.
  + Menyediakan laporan yang dapat disesuaikan berdasarkan kriteria tertentu (misalnya, mata pelajaran, semester, tahun ajaran).
* **Notifikasi dan Umpan Balik**:
  + Mengirimkan notifikasi kepada siswa dan orang tua tentang hasil studi dan laporan yang tersedia.
  + Menyediakan umpan balik kepada siswa berdasarkan analisis kinerja mereka.

**3. Arsitektur Modul**

Modul laporan dan analitik akan terdiri dari beberapa komponen utama:

* **Antarmuka Pengguna**:
  + Halaman untuk menghasilkan dan mengunduh laporan kartu hasil studi.
  + Halaman untuk melihat analisis kinerja siswa dan laporan statistik.
* **Logika Bisnis**:
  + Proses untuk mengambil data dari database dan menghasilkan laporan berdasarkan permintaan pengguna.
  + Proses untuk menganalisis data penilaian dan menghasilkan grafik serta statistik.
* **Database**:
  + Menggunakan tabel yang ada (seperti tabel nilai dan tabel kartu hasil studi) untuk mengambil data yang diperlukan untuk laporan.

**4. Desain Database**

Modul ini akan menggunakan tabel yang sudah ada dalam sistem, seperti:

* **Tabel Nilai**: Menyimpan data penilaian siswa.
* **Tabel Kartu Hasil Studi**: Menyimpan informasi tentang kartu hasil studi yang telah dihasilkan.

**5. Proses Pembuatan Laporan**

1. **Pembuatan Kartu Hasil Studi**:
   * Pengguna (guru atau admin) memilih siswa dan semester untuk menghasilkan kartu hasil studi.
   * Sistem mengambil data nilai dari tabel nilai dan menghitung rata-rata nilai.
   * Kartu hasil studi dihasilkan dalam format yang dapat dicetak atau diunduh.
2. **Analisis Kinerja Siswa**:
   * Pengguna memilih kriteria analisis (misalnya, mata pelajaran, semester).
   * Sistem mengambil data dari tabel nilai dan menghasilkan grafik yang menunjukkan tren kinerja siswa.
   * Menyediakan ringkasan analisis yang mencakup rata-rata nilai, nilai tertinggi, dan nilai terendah.
3. **Laporan Statistik**:
   * Pengguna memilih jenis laporan yang diinginkan (misalnya, distribusi nilai, perbandingan antar siswa).
   * Sistem menghasilkan laporan berdasarkan data yang diambil dari tabel nilai.
   * Laporan dapat diunduh dalam format yang diinginkan (PDF, Excel).

**6. Keamanan dan Akses**

* **Otorisasi Pengguna**: Hanya pengguna yang berwenang (guru dan admin) yang dapat mengakses modul laporan dan analitik.
* **Validasi Input**: Memastikan bahwa semua input pengguna divalidasi untuk mencegah kesalahan dan potensi serangan.

**7. Pengujian Modul**

* **Pengujian Unit**: Menguji setiap fungsi dalam modul secara terpisah untuk memastikan bahwa mereka berfungsi dengan benar.
* **Pengujian Integrasi**: Menguji interaksi antara modul laporan dan analitik dengan modul lain dalam sistem.
* **Pengujian Kinerja**: Menguji kecepatan dan efisiensi dalam menghasilkan laporan, terutama saat menangani data dalam jumlah besar.

#### 3.4 Modul integrasi API 1. Deskripsi Umum

Modul integrasi API bertanggung jawab untuk menyediakan antarmuka pemrograman aplikasi (API) yang memungkinkan komunikasi antara sistem Penilaian dan Hasil Kartu Hasil Studi dengan aplikasi lain, seperti sistem manajemen pembelajaran (LMS), aplikasi mobile, atau sistem eksternal lainnya. Modul ini memastikan bahwa data dapat diakses dan dikelola dengan cara yang aman dan efisien.

**2. Fungsi Utama**

* **Autentikasi API**:
  + Menyediakan mekanisme autentikasi untuk memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses API.
  + Menggunakan token berbasis OAuth 2.0 atau JWT (JSON Web Token) untuk autentikasi.
* **Endpoint untuk Manajemen Data**:
  + **Input Nilai**: Endpoint untuk memasukkan nilai siswa.
  + **Pengelolaan Mata Pelajaran**: Endpoint untuk menambah, mengedit, dan menghapus mata pelajaran.
  + **Pengambilan Kartu Hasil Studi**: Endpoint untuk mengambil kartu hasil studi siswa.
  + **Analisis Kinerja**: Endpoint untuk mengambil data analisis kinerja siswa.
* **Format Data**:
  + Menggunakan format JSON untuk pertukaran data antara sistem dan aplikasi lain.
  + Menyediakan dokumentasi API yang jelas untuk memudahkan pengembang dalam menggunakan API.

**3. Arsitektur Modul**

Modul integrasi API akan terdiri dari beberapa komponen utama:

* **Antarmuka API**:
  + Endpoint yang dapat diakses oleh aplikasi eksternal.
  + Menggunakan RESTful API untuk komunikasi.
* **Logika Bisnis**:
  + Proses untuk memproses permintaan API dan mengembalikan respons yang sesuai.
  + Mengelola validasi dan sanitasi data yang diterima dari permintaan API.
* **Database**:
  + Menggunakan tabel yang ada dalam sistem untuk mengambil dan menyimpan data yang diperlukan.

**4. Desain Endpoint API**

Berikut adalah beberapa endpoint yang akan disediakan dalam modul integrasi API:

* **Autentikasi**
  + **POST /api/auth/login**: Mengautentikasi pengguna dan mengembalikan token.
  + **POST /api/auth/logout**: Mengakhiri sesi pengguna.
* **Manajemen Nilai**
  + **POST /api/nilai**: Menambahkan nilai siswa.
  + **PUT /api/nilai/{id}**: Memperbarui nilai siswa berdasarkan ID.
  + **DELETE /api/nilai/{id}**: Menghapus nilai siswa berdasarkan ID.
  + **GET /api/nilai/{user\_id}**: Mengambil semua nilai siswa berdasarkan ID pengguna.
* **Manajemen Mata Pelajaran**
  + **POST /api/mata-pelajaran**: Menambahkan mata pelajaran baru.
  + **PUT /api/mata-pelajaran/{id}**: Memperbarui mata pelajaran berdasarkan ID.
  + **DELETE /api/mata-pelajaran/{id}**: Menghapus mata pelajaran berdasarkan ID.
  + **GET /api/mata-pelajaran**: Mengambil semua mata pelajaran.
* **Kartu Hasil Studi**
  + **GET /api/kartu-hasil-studi/{user\_id}**: Mengambil kartu hasil studi siswa berdasarkan ID pengguna.
* **Analisis Kinerja**
  + **GET /api/analisis/{user\_id}**: Mengambil analisis kinerja siswa berdasarkan ID pengguna.

**5. Keamanan**

* **Autentikasi dan Otorisasi**: Menggunakan token untuk memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses endpoint tertentu.
* **Validasi Input**: Memastikan bahwa semua data yang diterima melalui API divalidasi untuk mencegah serangan seperti SQL Injection dan Cross-Site Scripting (XSS).
* **Penggunaan HTTPS**: Menggunakan protokol HTTPS untuk mengenkripsi data yang ditransfer antara klien dan server.

**6. Dokumentasi API**

* Menyediakan dokumentasi yang jelas dan terperinci untuk setiap endpoint, termasuk:
  + Deskripsi fungsi endpoint.
  + Metode HTTP yang digunakan (GET, POST, PUT, DELETE).
  + Format data yang diterima dan dikembalikan.
  + Contoh permintaan dan respons.

**7. Pengujian Modul**

* **Pengujian Unit**: Menguji setiap endpoint secara terpisah untuk memastikan bahwa mereka berfungsi dengan benar.
* **Pengujian Integrasi**: Menguji interaksi antara modul API dan modul lain dalam sistem.
* **Pengujian Keamanan**: Melakukan pengujian penetrasi untuk mengidentifikasi

**4.Desain Basis Data  
 4.1 Model Data** Model data menggambarkan struktur data yang akan digunakan dalam sistem Penilaian dan Hasil Kartu Hasil Studi. Model ini mencakup tabel-tabel yang menyimpan informasi tentang pengguna, nilai, mata pelajaran, dan kartu hasil studi. Setiap tabel akan memiliki atribut yang relevan dan hubungan antar tabel.

**1. Tabel Pengguna**

Tabel ini menyimpan informasi tentang semua pengguna sistem, termasuk admin, guru, siswa, dan orang tua.

| **Kolom** | **Tipe Data** | **Deskripsi** |
| --- | --- | --- |
| user\_id | INT (PK) | ID unik untuk setiap pengguna |
| username | VARCHAR(50) | Nama pengguna yang digunakan untuk login |
| password | VARCHAR(255) | Password yang ter-enkripsi |
| role | ENUM | Peran pengguna (admin, guru, siswa, orang tua) |
| email | VARCHAR(100) | Alamat email untuk pemulihan password |
| created\_at | DATETIME | Tanggal dan waktu pendaftaran pengguna |
| updated\_at | DATETIME | Tanggal dan waktu terakhir diperbarui |

**2. Tabel Mata Pelajaran**

Tabel ini menyimpan informasi tentang mata pelajaran yang diajarkan di sekolah.

| **Kolom** | **Tipe Data** | **Deskripsi** |
| --- | --- | --- |
| mata\_pelajaran\_id | INT (PK) | ID unik untuk setiap mata pelajaran |
| nama\_mata\_pelajaran | VARCHAR(100) | Nama mata pelajaran |
| deskripsi | TEXT | Deskripsi tentang mata pelajaran |
| created\_at | DATETIME | Tanggal dan waktu mata pelajaran ditambahkan |
| updated\_at | DATETIME | Tanggal dan waktu terakhir diperbarui |

**3. Tabel Nilai**

Tabel ini menyimpan data penilaian siswa untuk setiap mata pelajaran.

| **Kolom** | **Tipe Data** | **Deskripsi** |
| --- | --- | --- |
| nilai\_id | INT (PK) | ID unik untuk setiap nilai |
| user\_id | INT (FK) | ID siswa yang terkait (referensi ke tabel Pengguna) |
| mata\_pelajaran\_id | INT (FK) | ID mata pelajaran yang terkait (referensi ke tabel Mata Pelajaran) |
| nilai | FLOAT | Nilai yang diberikan kepada siswa |
| semester | ENUM | Semester (Ganjil/Genap) |
| tahun\_ajaran | VARCHAR(10) | Tahun ajaran (misalnya, 2023/2024) |
| created\_at | DATETIME | Tanggal dan waktu nilai dimasukkan |
| updated\_at | DATETIME | Tanggal dan waktu terakhir diperbarui |

**4. Tabel Kartu Hasil Studi**

Tabel ini menyimpan informasi tentang kartu hasil studi siswa.

| **Kolom** | **Tipe Data** | **Deskripsi** |
| --- | --- | --- |
| kartu\_id | INT (PK) | ID unik untuk setiap kartu hasil studi |
| user\_id | INT (FK) | ID siswa yang terkait (referensi ke tabel Pengguna) |
| semester | ENUM | Semester (Ganjil/Genap) |
| tahun\_ajaran | VARCHAR(10) | Tahun ajaran (misalnya, 2023/2024) |
| rata\_rata | FLOAT | Rata-rata nilai siswa |
| created\_at | DATETIME | Tanggal dan waktu kartu hasil studi dibuat |
| updated\_at | DATETIME | Tanggal dan waktu terakhir diperbarui |

**5. Relasi Antar Tabel**

* **Pengguna dan Nilai**:
  + Tabel **Nilai** memiliki relasi many-to-one dengan tabel **Pengguna**, di mana satu siswa dapat memiliki banyak nilai.
* **Mata Pelajaran dan Nilai**:
  + Tabel **Nilai** memiliki relasi many-to-one dengan tabel **Mata Pelajaran**, di mana satu mata pelajaran dapat memiliki banyak nilai dari siswa yang berbeda.
* **Pengguna dan Kartu Hasil Studi**:
  + Tabel **Kartu Hasil Studi** memiliki relasi one-to-one dengan tabel **Pengguna**, di mana setiap siswa memiliki satu kartu hasil studi untuk setiap semester.

**6. Diagram ER (Entity-Relationship)**

Untuk menggambarkan model data secara visual, Anda dapat membuat diagram ER yang menunjukkan entitas (tabel) dan

Bookmark messageCopy messageExport

1. **4.2 diagram entity relationship (ERD)  
    Pengguna**
   * **Atribut**:
     + user\_id (PK)
     + username
     + password
     + role
     + email
     + created\_at
     + updated\_at
2. **Mata Pelajaran**
   * **Atribut**:
     + mata\_pelajaran\_id (PK)
     + nama\_mata\_pelajaran
     + deskripsi
     + created\_at
     + updated\_at
3. **Nilai**
   * **Atribut**:
     + nilai\_id (PK)
     + user\_id (FK)
     + mata\_pelajaran\_id (FK)
     + nilai
     + semester
     + tahun\_ajaran
     + created\_at
     + updated\_at
4. **Kartu Hasil Studi**
   * **Atribut**:
     + kartu\_id (PK)
     + user\_id (FK)
     + semester
     + tahun\_ajaran
     + rata\_rata
     + created\_at
     + updated\_at

### Hubungan Antar Entitas

1. **Pengguna - Nilai**:
   * Hubungan: Satu pengguna (siswa) dapat memiliki banyak nilai.
   * Tipe: One-to-Many (1:N)
2. **Mata Pelajaran - Nilai**:
   * Hubungan: Satu mata pelajaran dapat memiliki banyak nilai dari siswa yang berbeda.
   * Tipe: One-to-Many (1:N)
3. **Pengguna - Kartu Hasil Studi**:
   * Hubungan: Satu pengguna (siswa) memiliki satu kartu hasil studi untuk setiap semester.
   * Tipe: One-to-One (1:1)

**4.3 skema indexing dan optimasi  
 1. Deskripsi Umum**

Indexing adalah teknik yang digunakan untuk meningkatkan kecepatan pengambilan data dari database. Dengan menggunakan indeks yang tepat, sistem dapat mengurangi waktu yang dibutuhkan untuk melakukan query, terutama pada tabel yang besar. Optimasi database juga mencakup pengaturan dan pengelolaan data untuk memastikan kinerja yang optimal.

**2. Skema Indexing**

Berikut adalah skema indexing yang diusulkan untuk tabel-tabel dalam sistem:

* **Tabel Pengguna**
  + **Indeks Utama**:
    - **PRIMARY KEY (user\_id)**: Indeks utama untuk kolom **user\_id** untuk memastikan keunikan dan mempercepat pencarian berdasarkan ID pengguna.
  + **Indeks Tambahan**:
    - **INDEX (username)**: Indeks pada kolom **username** untuk mempercepat pencarian pengguna berdasarkan nama pengguna.
    - **INDEX (email)**: Indeks pada kolom **email** untuk mempercepat pencarian pengguna berdasarkan alamat email.
* **Tabel Mata Pelajaran**
  + **Indeks Utama**:
    - **PRIMARY KEY (mata\_pelajaran\_id)**: Indeks utama untuk kolom **mata\_pelajaran\_id**.
  + **Indeks Tambahan**:
    - **INDEX (nama\_mata\_pelajaran)**: Indeks pada kolom **nama\_mata\_pelajaran** untuk mempercepat pencarian mata pelajaran berdasarkan nama.
* **Tabel Nilai**
  + **Indeks Utama**:
    - **PRIMARY KEY (nilai\_id)**: Indeks utama untuk kolom **nilai\_id**.
  + **Indeks Tambahan**:
    - **INDEX (user\_id)**: Indeks pada kolom **user\_id** untuk mempercepat pencarian nilai berdasarkan ID siswa.
    - **INDEX (mata\_pelajaran\_id)**: Indeks pada kolom **mata\_pelajaran\_id** untuk mempercepat pencarian nilai berdasarkan mata pelajaran.
    - **INDEX (semester, tahun\_ajaran)**: Indeks komposit pada kolom **semester** dan **tahun\_ajaran** untuk mempercepat pencarian nilai berdasarkan semester dan tahun ajaran.
* **Tabel Kartu Hasil Studi**
  + **Indeks Utama**:
    - **PRIMARY KEY (kartu\_id)**: Indeks utama untuk kolom **kartu\_id**.
  + **Indeks Tambahan**:
    - **INDEX (user\_id)**: Indeks pada kolom **user\_id** untuk mempercepat pencarian kartu hasil studi berdasarkan ID siswa.
    - **INDEX (semester, tahun\_ajaran)**: Indeks komposit pada kolom **semester** dan **tahun\_ajaran** untuk mempercepat pencarian kartu hasil studi berdasarkan semester dan tahun ajaran.

**3. Optimasi Database**

Untuk memastikan kinerja database yang optimal, beberapa langkah optimasi dapat diambil:

* **Normalisasi**:
  + Pastikan bahwa database dinormalisasi untuk mengurangi redundansi data dan meningkatkan integritas data. Namun, perlu diingat bahwa normalisasi yang berlebihan dapat mempengaruhi kinerja, sehingga perlu menemukan keseimbangan yang tepat.
* **Penggunaan Query yang Efisien**:
  + Tulis query SQL yang efisien dengan menggunakan JOIN yang tepat dan menghindari subquery yang tidak perlu. Gunakan LIMIT untuk membatasi jumlah hasil yang dikembalikan jika tidak semua data diperlukan.
* **Pengaturan Indeks**:
  + Secara berkala tinjau dan sesuaikan indeks berdasarkan pola akses data. Hapus indeks yang tidak digunakan dan tambahkan indeks baru jika diperlukan.
* **Penggunaan Caching**:
  + Implementasikan caching untuk menyimpan hasil query yang sering diakses. Ini dapat mengurangi beban pada database dan meningkatkan kecepatan akses data.
* **Monitoring Kinerja**:
  + Gunakan alat monitoring untuk memantau kinerja database dan mengidentifikasi query yang lambat. Analisis dan optimalkan query tersebut untuk meningkatkan kinerja.
* **Pengaturan Partisi**:
  + Jika tabel menjadi sangat besar, pertimbangkan untuk menggunakan partisi untuk membagi tabel menjadi bagian yang lebih kecil. Ini dapat meningkatkan kinerja query dan manajemen data.

**4.**Pengujian Kinerja

Setelah menerapkan skema indexing dan optimasi, lakukan pengujian kinerja untuk memastikan bahwa perubahan yang dilakukan memberikan peningkatan yang signifikan. Uji waktu respons query sebelum dan sesudah optimasi untuk mengevaluasi efektivitasnya.

**5.Antar Muka Pengguna  
5.1 Wireframe desain** Halaman ini digunakan oleh pengguna untuk masuk ke sistem.

| **Username: [\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_]** |
| --- |
| Password: [\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_] |
|  |
| [Login] [Forgot Password?] |
|  |
| [Register] |
|  |

+--------------------------------------------------+

### 5.2 Desain Responsif 1. ****Prinsip Desain Responsif****

Desain responsif mengutamakan fleksibilitas dan adaptabilitas. Beberapa prinsip dasar yang harus diperhatikan dalam desain responsif adalah:

* **Grid Layout**: Menggunakan sistem grid yang fleksibel untuk mengatur elemen-elemen di halaman. Grid ini akan menyesuaikan ukuran kolom dan baris berdasarkan lebar layar.
* **Media Queries**: Menggunakan media queries dalam CSS untuk menerapkan gaya yang berbeda berdasarkan ukuran layar. Ini memungkinkan elemen untuk beradaptasi dengan baik pada perangkat yang berbeda.
* **Ukuran Fleksibel**: Menggunakan unit ukuran yang fleksibel seperti persentase, em, atau rem untuk elemen, sehingga mereka dapat menyesuaikan diri dengan ukuran layar.
* **Gambar Responsif**: Menggunakan gambar yang dapat menyesuaikan ukuran dengan baik, seperti menggunakan CSS **max-width: 100%** untuk memastikan gambar tidak melampaui lebar kontainer.

### 2. ****Desain Responsif Halaman Utama****

Berikut adalah contoh bagaimana halaman utama (Dashboard) dapat dirancang responsif:

#### a. ****Desktop View****

* **Header**: Menampilkan logo dan menu navigasi horizontal.
* **Konten**: Menampilkan ringkasan informasi dalam kolom.
* **Sidebar**: Menampilkan menu navigasi tambahan di sisi kiri.

#### b. ****Tablet View****

* **Header**: Menu navigasi tetap horizontal, tetapi mungkin menyusut.
* **Konten**: Ringkasan informasi ditampilkan dalam dua kolom.
* **Sidebar**: Menu navigasi dapat disembunyikan di dalam dropdown.

#### c. ****Mobile View****

* **Header**: Menu navigasi menjadi dropdown atau hamburger menu.
* **Konten**: Ringkasan informasi ditampilkan dalam satu kolom.
* **Sidebar**: Menu navigasi tidak ditampilkan, tetapi dapat diakses melalui hamburger menu.

### 6.Pertimbangan Keamanan 1. ****Autentikasi dan Otorisasi****

* **Autentikasi Pengguna**:
  + Implementasikan mekanisme autentikasi yang kuat, seperti penggunaan username dan password yang kompleks. Pertimbangkan untuk menggunakan autentikasi dua faktor (2FA) untuk meningkatkan keamanan.
* **Otorisasi**:
  + Pastikan bahwa hanya pengguna yang berwenang (guru, admin, siswa, orang tua) yang dapat mengakses data tertentu. Gunakan kontrol akses berbasis peran (RBAC) untuk mengelola izin akses.

### 2. ****Enkripsi Data****

* **Enkripsi Data Sensitif**:
  + Enkripsi data sensitif seperti password pengguna menggunakan algoritma hashing yang kuat (misalnya, bcrypt, Argon2) sebelum menyimpannya di database.
* **Enkripsi Data dalam Transit**:
  + Gunakan protokol HTTPS untuk mengenkripsi data yang ditransfer antara klien dan server, sehingga mencegah penyadapan data oleh pihak ketiga.

### 3. ****Validasi dan Sanitasi Input****

* **Validasi Input**:
  + Lakukan validasi pada semua input pengguna untuk memastikan bahwa data yang diterima sesuai dengan format yang diharapkan. Ini dapat mencegah serangan seperti SQL Injection dan Cross-Site Scripting (XSS).
* **Sanitasi Input**:
  + Sanitasi input untuk menghapus karakter berbahaya dan mencegah eksekusi kode berbahaya. Gunakan library yang sudah teruji untuk sanitasi input.

### 4. ****Pengelolaan Sesi****

* **Manajemen Sesi yang Aman**:
  + Gunakan token sesi yang aman dan pastikan token tersebut memiliki masa berlaku. Implementasikan pengaturan timeout untuk sesi yang tidak aktif.
* **Penggunaan Cookie yang Aman**:
  + Setel atribut **HttpOnly** dan **Secure** pada cookie untuk mencegah akses JavaScript dan memastikan cookie hanya dikirim melalui HTTPS.

### 5. ****Audit dan Logging****

* **Audit Keamanan**:
  + Lakukan audit keamanan secara berkala untuk mengidentifikasi dan mengatasi kerentanan dalam sistem.
* **Logging Aktivitas**:
  + Implementasikan logging untuk mencatat aktivitas pengguna, termasuk login, pengubahan data, dan akses ke data sensitif. Pastikan log disimpan dengan aman dan tidak dapat diubah oleh pengguna.

### 6. ****Pencegahan Serangan Umum****

* **Cross-Site Scripting (XSS)**:
  + Gunakan teknik seperti Content Security Policy (CSP) untuk mencegah serangan XSS. Sanitasi semua output yang ditampilkan kepada pengguna.
* **Cross-Site Request Forgery (CSRF)**:
  + Implementasikan token CSRF untuk melindungi formulir dari serangan CSRF. Pastikan setiap permintaan yang mengubah data dilindungi oleh token yang valid.
* **SQL Injection**:
  + Gunakan parameterized queries atau prepared statements untuk mencegah serangan SQL Injection. Hindari penggunaan query dinamis yang menggabungkan input pengguna secara langsung.

### 7. ****Pendidikan Pengguna****

* **Pelatihan Keamanan**:
  + Berikan pelatihan kepada pengguna (guru, admin, siswa) tentang praktik keamanan yang baik, termasuk cara membuat password yang kuat dan mengenali phishing.

### 8. ****Backup dan Pemulihan****

* **Backup Data**:
  + Lakukan backup data secara berkala untuk memastikan bahwa data dapat dipulihkan jika terjadi kehilangan data atau serangan.
* **Rencana Pemulihan Bencana**:
  + Siapkan rencana pemulihan bencana untuk mengatasi situasi darurat, termasuk serangan siber atau kegagalan sistem.

### 7.Pengujian Dan Validasi 1. ****Tujuan Pengujian****

* Memastikan bahwa sistem berfungsi dengan baik dan memenuhi persyaratan fungsional dan non-fungsional.
* Mengidentifikasi dan memperbaiki bug atau masalah sebelum sistem diluncurkan.
* Memastikan bahwa sistem aman dan dapat diandalkan.

### 2. ****Jenis Pengujian****

Berikut adalah beberapa jenis pengujian yang akan dilakukan:

#### a. ****Pengujian Unit****

* **Deskripsi**: Menguji setiap unit atau komponen terkecil dari sistem secara terpisah untuk memastikan bahwa mereka berfungsi dengan benar.
* **Tujuan**: Memastikan bahwa setiap fungsi atau metode dalam kode bekerja sesuai harapan.
* **Alat**: JUnit (untuk Java), NUnit (untuk .NET), pytest (untuk Python).

#### b. ****Pengujian Integrasi****

* **Deskripsi**: Menguji interaksi antara berbagai modul atau komponen sistem untuk memastikan bahwa mereka bekerja sama dengan baik.
* **Tujuan**: Memastikan bahwa data dapat mengalir dengan benar antara modul yang berbeda dan tidak ada masalah dalam integrasi.
* **Alat**: Postman (untuk API), Selenium (untuk pengujian web).

#### c. ****Pengujian Fungsional****

* **Deskripsi**: Menguji fungsionalitas sistem berdasarkan spesifikasi yang telah ditentukan.
* **Tujuan**: Memastikan bahwa semua fitur berfungsi sesuai dengan yang diharapkan oleh pengguna.
* **Metode**: Pengujian manual dan otomatisasi.

#### d. ****Pengujian Sistem****

* **Deskripsi**: Menguji sistem secara keseluruhan untuk memastikan bahwa semua komponen berfungsi dengan baik dalam lingkungan yang terintegrasi.
* **Tujuan**: Memastikan bahwa sistem memenuhi semua persyaratan fungsional dan non-fungsional.
* **Alat**: Selenium, JMeter (untuk pengujian kinerja).

#### e. ****Pengujian Keamanan****

* **Deskripsi**: Menguji sistem untuk mengidentifikasi kerentanan dan memastikan bahwa data pengguna dilindungi.
* **Tujuan**: Memastikan bahwa sistem aman dari serangan dan kebocoran data.
* **Alat**: OWASP ZAP, Burp Suite.

#### f. ****Pengujian Kinerja****

* **Deskripsi**: Menguji kinerja sistem di bawah beban yang berbeda untuk memastikan bahwa sistem dapat menangani jumlah pengguna yang diharapkan.
* **Tujuan**: Memastikan bahwa sistem responsif dan dapat diandalkan di bawah beban tinggi.
* **Alat**: JMeter, LoadRunner.

#### g. ****Pengujian Pengguna (User Acceptance Testing - UAT)****

* **Deskripsi**: Melibatkan pengguna akhir untuk menguji sistem dan memberikan umpan balik.
* **Tujuan**: Memastikan bahwa sistem memenuhi kebutuhan dan harapan pengguna.
* **Metode**: Sesi pengujian dengan pengguna yang sebenarnya.

### 3. ****Proses Pengujian****

* **Perencanaan Pengujian**: Menyusun rencana pengujian yang mencakup jenis pengujian, alat yang digunakan, dan jadwal pengujian.
* **Pelaksanaan Pengujian**: Melakukan pengujian sesuai dengan rencana yang telah disusun.
* **Pelaporan Hasil**: Mencatat dan melaporkan hasil pengujian, termasuk bug yang ditemukan dan status perbaikan.
* **Perbaikan dan Retesting**: Memperbaiki bug yang ditemukan dan melakukan pengujian ulang untuk memastikan bahwa perbaikan berhasil.

### 4. ****Validasi****

* **Deskripsi**: Validasi adalah proses untuk memastikan bahwa sistem memenuhi kebutuhan dan harapan pengguna.
* **Metode**:
  + **Review Spesifikasi**: Memastikan bahwa semua persyaratan telah diidentifikasi dan didokumentasikan dengan baik.
  + **Prototyping**: Menggunakan prototipe untuk mendapatkan umpan balik awal dari pengguna.
  + **UAT**: Melibatkan pengguna akhir dalam pengujian untuk memastikan bahwa sistem memenuhi kebutuhan mereka.

### 5. ****Dokumentasi Pengujian****

* Semua hasil pengujian, termasuk rencana pengujian, kasus uji, dan laporan bug, harus didokumentasikan dengan baik untuk referensi di masa mendatang dan untuk audit.

### 7.1 Pengujian Unit 1. ****Tujuan Pengujian Unit****

* Memastikan bahwa setiap unit kode berfungsi sesuai dengan spesifikasi yang telah ditentukan.
* Mengidentifikasi dan memperbaiki bug pada tahap awal pengembangan.
* Memfasilitasi perubahan kode di masa depan dengan memberikan jaminan bahwa perubahan tidak merusak fungsionalitas yang ada.

### 2. ****Pendekatan Pengujian Unit****

* **Pengujian Otomatis**: Menggunakan framework pengujian otomatis untuk menjalankan pengujian unit secara efisien dan konsisten.
* **Pengujian Manual**: Dalam beberapa kasus, pengujian manual dapat dilakukan untuk fungsi yang lebih kompleks atau ketika pengujian otomatis tidak praktis.

### 3. ****Framework Pengujian Unit****

Beberapa framework yang dapat digunakan untuk pengujian unit tergantung pada bahasa pemrograman yang digunakan:

* **Java**: JUnit
* **C#**: NUnit
* **Python**: unittest atau pytest
* **JavaScript**: Jest atau Mocha

### 4. ****Contoh Kasus Pengujian Unit****

Berikut adalah beberapa contoh kasus pengujian unit yang dapat diterapkan dalam sistem Penilaian dan Hasil Kartu Hasil Studi:

#### a. ****Pengujian Fungsi Validasi Input****

* **Deskripsi**: Menguji fungsi yang memvalidasi input nilai siswa.
* **Input**: Nilai yang valid dan tidak valid.
* **Output yang Diharapkan**: Fungsi harus mengembalikan true untuk nilai valid dan false untuk nilai tidak valid.

### 7.2 Pengujian Keamanan 1. ****Tujuan Pengujian Keamanan****

* **Identifikasi Kerentanan**: Mengidentifikasi kerentanan yang dapat dieksploitasi oleh penyerang.
* **Perlindungan Data Sensitif**: Memastikan bahwa data pengguna, termasuk informasi pribadi dan hasil penilaian, dilindungi dari akses yang tidak sah.
* **Kepatuhan terhadap Standar**: Memastikan bahwa sistem mematuhi standar keamanan dan regulasi yang berlaku, seperti GDPR atau HIPAA.

### 2. ****Lingkup Pengujian Keamanan****

Pengujian keamanan akan mencakup:

* Aplikasi web (frontend dan backend)
* API yang digunakan untuk komunikasi antara klien dan server
* Database yang menyimpan data pengguna dan hasil penilaian

### 3. ****Jenis Pengujian Keamanan****

Berikut adalah beberapa jenis pengujian keamanan yang akan dilakukan:

#### a. ****Pengujian Penetrasi (Penetration Testing)****

* **Deskripsi**: Menguji sistem dengan cara mensimulasikan serangan dari penyerang untuk mengidentifikasi kerentanan.
* **Metode**: Menggunakan alat otomatis dan teknik manual untuk menguji berbagai vektor serangan.
* **Alat**: OWASP ZAP, Burp Suite, Metasploit.

#### b. ****Pengujian Kerentanan (Vulnerability Scanning)****

* **Deskripsi**: Menggunakan alat untuk secara otomatis memindai sistem dan aplikasi untuk menemukan kerentanan yang dikenal.
* **Tujuan**: Mengidentifikasi kerentanan yang dapat dieksploitasi sebelum penyerang melakukannya.
* **Alat**: Nessus, Qualys, OpenVAS.

#### c. ****Pengujian Keamanan Aplikasi Web****

* **Deskripsi**: Menguji aplikasi web untuk kerentanan umum seperti SQL Injection, Cross-Site Scripting (XSS), dan Cross-Site Request Forgery (CSRF).
* **Metode**: Menggunakan teknik manual dan otomatis untuk menguji input pengguna dan interaksi dengan aplikasi.
* **Alat**: OWASP ZAP, Burp Suite.

#### d. ****Pengujian Keamanan API****

* **Deskripsi**: Menguji API untuk memastikan bahwa mereka aman dari serangan dan tidak mengungkapkan data sensitif.
* **Metode**: Menguji endpoint API untuk kerentanan seperti otorisasi yang tidak tepat dan eksposur data.
* **Alat**: Postman, OWASP ZAP.

### 4. ****Proses Pengujian Keamanan****

1. **Perencanaan Pengujian**: Menyusun rencana pengujian keamanan yang mencakup ruang lingkup, jenis pengujian yang akan dilakukan, dan alat yang akan digunakan.
2. **Pengumpulan Informasi**: Mengumpulkan informasi tentang sistem, termasuk arsitektur, teknologi yang digunakan, dan potensi vektor serangan.
3. **Pengujian**: Melakukan pengujian sesuai dengan rencana yang telah disusun, menggunakan alat dan teknik yang sesuai.
4. **Analisis Hasil**: Menganalisis hasil pengujian untuk mengidentifikasi kerentanan dan risiko yang ditemukan.
5. **Pelaporan**: Menyusun laporan pengujian keamanan yang mencakup temuan, rekomendasi perbaikan, dan langkah-langkah mitigasi.
6. **Perbaikan dan Retesting**: Memperbaiki kerentanan yang ditemukan dan melakukan pengujian ulang untuk memastikan bahwa perbaikan berhasil.

### 5. ****Kerentanan Umum yang Diuji****

* **SQL Injection**: Menguji apakah aplikasi rentan terhadap serangan SQL Injection dengan mencoba menyisipkan perintah SQL berbahaya melalui input pengguna.
* **Cross-Site Scripting (XSS)**: Menguji apakah aplikasi rentan terhadap XSS dengan mencoba menyisipkan skrip berbahaya ke dalam input yang ditampilkan kepada pengguna.
* **Cross-Site Request Forgery (CSRF)**: Menguji apakah aplikasi memiliki perlindungan terhadap CSRF dengan mencoba mengirim permintaan yang tidak sah atas nama pengguna.
* **Pengelolaan Sesi**: Memastikan bahwa sesi pengguna dikelola dengan aman, termasuk penggunaan cookie yang aman dan pengaturan timeout sesi.

### 6. ****Dokumentasi Pengujian Keamanan****

* Semua hasil pengujian, termasuk kerentanan yang ditemukan, langkah-langkah perbaikan yang diambil, dan rekomendasi untuk meningkatkan keamanan, harus

### 7.3 Uji Fungsional 1. ****Tujuan Uji Fungsional****

* Memastikan bahwa semua fitur dan fungsi dalam sistem beroperasi sesuai dengan yang diharapkan.
* Mengidentifikasi dan memperbaiki bug atau masalah sebelum sistem diluncurkan.
* Memastikan bahwa sistem memenuhi persyaratan fungsional yang telah ditetapkan.

### 2. ****Lingkup Uji Fungsional****

Uji fungsional akan mencakup semua fitur utama dari sistem, termasuk:

* Autentikasi pengguna (login dan registrasi)
* Input nilai siswa
* Penghitungan dan penyimpanan nilai
* Pembuatan dan tampilan kartu hasil studi
* Analisis kinerja siswa
* Pengelolaan data pengguna (guru, siswa, orang tua)

### 3. ****Metodologi Uji Fungsional****

* **Pengujian Manual**: Melibatkan pengujian manual oleh penguji untuk memastikan bahwa semua fungsi beroperasi dengan baik.
* **Pengujian Otomatis**: Menggunakan alat otomatis untuk menjalankan skrip pengujian yang telah ditentukan untuk menguji fungsionalitas sistem.

### 4. ****Kasus Uji Fungsional****

Berikut adalah beberapa contoh kasus uji fungsional yang dapat diterapkan dalam sistem:

#### a. ****Uji Fungsional Autentikasi Pengguna****

* **Deskripsi**: Menguji proses login dan registrasi pengguna.
* **Langkah Uji**:
  1. Buka halaman login.
  2. Masukkan username dan password yang valid.
  3. Klik tombol "Login".
* **Hasil yang Diharapkan**: Pengguna berhasil masuk ke sistem dan diarahkan ke dashboard.
* **Langkah Uji untuk Registrasi**:
  1. Buka halaman registrasi.
  2. Masukkan data yang valid (username, password, email).
  3. Klik tombol "Daftar".
* **Hasil yang Diharapkan**: Pengguna berhasil terdaftar dan menerima konfirmasi.

#### b. ****Uji Fungsional Input Nilai****

* **Deskripsi**: Menguji proses input nilai siswa oleh guru.
* **Langkah Uji**:
  1. Masuk sebagai pengguna dengan peran guru.
  2. Akses halaman input nilai.
  3. Masukkan data nilai siswa (nama, mata pelajaran, nilai).
  4. Klik tombol "Simpan".
* **Hasil yang Diharapkan**: Nilai siswa berhasil disimpan dan ditampilkan dalam daftar nilai.

#### c. ****Uji Fungsional Kartu Hasil Studi****

* **Deskripsi**: Menguji tampilan kartu hasil studi siswa.
* **Langkah Uji**:
  1. Masuk sebagai pengguna dengan peran siswa atau orang tua.
  2. Akses halaman kartu hasil studi.
  3. Pilih siswa yang ingin dilihat kartu hasil studinya.
* **Hasil yang Diharapkan**: Kartu hasil studi ditampilkan dengan informasi yang benar (mata pelajaran, nilai, rata-rata).

#### d. ****Uji Fungsional Analisis Kinerja****

* **Deskripsi**: Menguji fitur analisis kinerja siswa.
* **Langkah Uji**:
  1. Masuk sebagai pengguna dengan peran guru.
  2. Akses halaman analisis kinerja.
  3. Pilih siswa dan mata pelajaran untuk dianalisis.
* **Hasil yang Diharapkan**: Grafik dan statistik kinerja siswa ditampilkan dengan benar.

### 5. ****Dokumentasi Uji Fungsional****

* Semua kasus uji, hasil pengujian, dan bug yang ditemukan harus didokumentasikan dengan baik untuk referensi di masa mendatang.
* Buat laporan uji fungsional yang mencakup ringkasan hasil, jumlah kasus uji yang berhasil dan gagal, serta langkah-langkah perbaikan yang diambil.

#### 