**Rincian Detail Fitur Aplikasi CBT**

Dokumen ini membedah setiap fitur utama aplikasi secara mendalam, menjelaskan komponen antarmuka, proses teknis di belakang layar, dan data yang terlibat.

**Panel Admin**

Panel admin dirancang sebagai *Single Page Application (SPA)* secara fungsional, di mana semua aksi (tambah, edit, hapus, cari) terjadi secara dinamis menggunakan AJAX tanpa perlu memuat ulang halaman.

**1. Manajemen Ujian (manage\_tests.php)**

* **Tujuan:** Memberikan admin kemampuan penuh untuk mengelola daftar ujian.
* **Komponen Antarmuka:**
  + **Form Pencarian & Filter Kategori:** Memungkinkan admin mencari ujian berdasarkan judul atau menyaringnya berdasarkan kategori.
  + **Tabel Dinamis:** Menampilkan daftar ujian dengan paginasi (navigasi halaman). Kolom "Total Poin" dihitung secara *real-time* dari soal yang sudah dirakit.
  + **Tombol "Tambah Ujian":** Membuka sebuah modal (popup) untuk membuat ujian baru.
  + **Tombol Aksi (Rakit, Edit, Hapus):** Ikon di setiap baris untuk melakukan tindakan spesifik.
* **Proses Teknis:**
  + Saat halaman dimuat atau filter diubah, JavaScript memanggil manage\_tests.php?fetch\_list=true via **AJAX**.
  + PHP mengambil data dari tabel **tests** dan melakukan LEFT JOIN ke **test\_questions** untuk menghitung SUM(points).
  + Data dikembalikan sebagai **JSON**, dan JavaScript membangun tabelnya.
  + Saat tombol "Edit" ditekan, JS memanggil get\_test\_details.php untuk mengambil data spesifik dan mengisi modal.
  + Saat form di modal disubmit, JS mengirim data via **AJAX** ke manage\_tests.php dengan action=add\_test atau action=edit\_test. PHP memproses INSERT atau UPDATE di tabel **tests**.
* **Data yang Terlibat:** tests (baca/tulis), test\_questions (baca untuk kalkulasi poin).

**2. Manajemen Bank Soal (manage\_question\_bank.php)**

* **Tujuan:** Sebagai pusat penyimpanan dan pengelolaan semua soal.
* **Komponen Antarmuka:**
  + **Filter & Pencarian:** Sama seperti Manajemen Ujian. Filter kategori di sini dibuat dinamis dari data yang ada.
  + **Tabel Dinamis:** Menampilkan daftar soal dengan paginasi.
  + **Modal Tambah/Edit Soal:** Sebuah form komprehensif yang mencakup:
    - Input teks untuk kategori dan pertanyaan.
    - Tombol *upload* untuk **gambar** dan **audio**.
    - Area **pilihan jawaban dinamis**, di mana admin bisa menambah atau mengurangi jumlah opsi.
    - Tombol *radio* di samping setiap opsi untuk menentukan **kunci jawaban**.
* **Proses Teknis:**
  + Alur pengambilan dan penampilan data sama persis dengan Manajemen Ujian, menggunakan **AJAX** dan **JSON**.
  + Saat form modal disubmit, data dikirim via **AJAX**.
  + PHP memvalidasi input. Jika ada file (gambar/audio), PHP memindahkannya ke folder uploads/ dan menyimpan path-nya (lokasi file) di database.
  + Pilihan jawaban yang dinamis dikumpulkan menjadi sebuah array, lalu diubah menjadi format **JSON** sebelum disimpan ke dalam satu kolom di tabel **questions**.
* **Data yang Terlibat:** questions (baca/tulis).

**3. Rakit Ujian (assemble\_test.php)**

* **Tujuan:** Fitur inti yang menghubungkan ujian dengan soal, serta mengatur bobot nilai.
* **Komponen Antarmuka:**
  + **Panel Kiri (Soal Ujian):** Daftar soal yang sudah terpilih. Setiap baris memiliki:
    - Ikon *grip* untuk **drag-and-drop** guna mengubah urutan.
    - Input angka untuk mengatur **poin** spesifik untuk soal tersebut.
    - Tombol hapus untuk mengeluarkan soal dari ujian.
  + **Panel Kanan (Bank Soal):** Daftar soal yang tersedia untuk ditambahkan, lengkap dengan **filter, pencarian, dan paginasi**.
  + **Tombol "Simpan Perubahan":** Untuk menyimpan urutan dan poin yang baru.
* **Proses Teknis:**
  + Saat menambahkan soal, form di panel kanan mengirim ID soal yang dipilih ke assemble\_test.php via POST biasa, yang kemudian melakukan INSERT ke tabel **test\_questions**.
  + Saat "Simpan Perubahan" ditekan, JavaScript membaca urutan dan nilai poin dari semua soal di panel kiri.
  + Data ini (sebuah array objek) dikirim via **AJAX** ke update\_test\_questions.php.
  + PHP melakukan *looping* dan UPDATE setiap baris di tabel **test\_questions** untuk memperbarui kolom question\_order dan points.
* **Data yang Terlibat:** test\_questions (baca/tulis), questions (baca), tests (baca).

**Panel Siswa**

Panel siswa dirancang untuk alur kerja yang linear dan jelas, memandu siswa dari pemilihan hingga penyelesaian ujian.

**1. Pengerjaan Ujian (test\_page.php)**

* **Tujuan:** Menyediakan antarmuka yang fokus dan bebas gangguan bagi siswa untuk mengerjakan soal.
* **Komponen Antarmuka:**
  + **Area Soal:** Menampilkan satu soal per waktu, lengkap dengan **gambar atau audio** jika ada.
  + **Timer:** Penghitung waktu mundur yang selalu terlihat.
  + **Tombol Navigasi (Next/Back):** Untuk berpindah antar soal secara berurutan.
  + **Panel Navigasi Soal:** Kotak-kotak bernomor yang memungkinkan siswa melompat ke soal mana pun. Warna kotak berubah menjadi hijau jika soal sudah dijawab.
  + **Modal Konfirmasi Selesai:** Muncul saat siswa menekan "Selesaikan Ujian" untuk mencegah penyelesaian yang tidak disengaja.
* **Proses Teknis:**
  + PHP memuat semua data soal dan jawaban siswa (jika ada) ke dalam halaman HTML.
  + JavaScript menangani semua logika tampilan: menyembunyikan/menampilkan soal, memperbarui timer setiap detik, dan mengubah warna panel navigasi.
  + Tidak ada komunikasi dengan server selama pengerjaan, semua jawaban disimpan sementara di dalam form HTML.
  + Jika timer habis, JavaScript secara otomatis memicu pengiriman form.
* **Data yang Terlibat:** questions (baca), test\_questions (baca), student\_answers (baca untuk memuat jawaban lama jika ada).

**2. Proses Penilaian (submit\_test.php)**

* **Tujuan:** Mesin penilai otomatis yang bekerja di belakang layar.
* **Proses Teknis:**
  1. Menerima semua jawaban siswa dari form.
  2. Melakukan satu query untuk mengambil semua **kunci jawaban** dan **poin** dari soal-soal yang ada di ujian tersebut.
  3. Melakukan *looping* untuk setiap soal:
     + Membandingkan jawaban siswa dengan kunci jawaban.
     + Jika benar, menambahkan nilai points dari soal itu ke variabel total\_score.
     + Melakukan INSERT ke tabel **student\_answers** untuk mencatat jawaban siswa dan status kebenarannya (is\_correct).
  4. Setelah loop selesai, PHP **tidak** menghitung skor berdasarkan persentase (jawaban benar / total soal), melainkan langsung menggunakan total\_score yang sudah diakumulasi.
  5. Melakukan UPDATE pada tabel **test\_results**, mengisi kolom score dengan total\_score dan mengubah status menjadi 'completed'.
* **Data yang Terlibat:** student\_answers (tulis), test\_results (tulis/update), questions & test\_questions (baca untuk kunci jawaban dan poin).

**3. Halaman Hasil (result\_page.php)**

* **Tujuan:** Memberikan umpan balik yang jelas dan detail kepada siswa.
* **Komponen Antarmuka:**
  + **Kartu Skor Utama:** Menampilkan skor akhir yang didapat.
  + **Ringkasan Jawaban:** Menampilkan total soal, total jawaban benar, dan total jawaban salah.
  + **Dropdown Review:** Sebuah tombol yang bisa diklik untuk menampilkan atau menyembunyikan detail pembahasan semua soal.
  + **Detail Pembahasan:** Saat dibuka, menampilkan setiap soal, semua opsi, menandai mana jawaban yang benar (hijau), mana jawaban siswa yang salah (merah), dan mana jawaban siswa yang benar.
* **Proses Teknis:**
  + PHP mengambil data skor dari **test\_results**.
  + PHP mengambil semua detail soal dan jawaban dari **student\_answers** dan **questions**.
  + Semua data ditampilkan di HTML.
  + JavaScript hanya digunakan untuk fungsi sederhana *toggle* (buka/tutup) pada dropdown review.
* **Data yang Terlibat:** test\_results (baca), student\_answers (baca), questions (baca).