Spesifikasi Tugas Besar Milestone 1 IF3110

Milestone 1 - Monolithic PHP & Vanilla Web Application

Revisi 1 - Nama Grup Gitlab (23/09/2023)
Revisi 2 - Database, Pengumpulan (28/09/2023)

Deskripsi Persoalan

Roro yang meminta *princess treatment* ke Bondowoso

Prabu Baka, ayah dari Roro, merupakan seorang investor *startup* terkemuka. Dia ingin agar di negaranya semakin banyak *startup* sehingga dia bisa ~~menjadi semakin kaya~~ semakin banyak membantu orang lain. Dia kemudian meminta Roro mencari pasangan yang siap untuk memberikan hadiah berupa 1000 *startup* sebagai syarat untuk menikahi Roro. Roro tahu bahwa Bondowoso, orang yang sangat ingin meminangnya, memiliki banyak jin yang ahli di bidang informatika. Oleh karena itu, untuk memenuhi wasiat dari ayahnya, dia meminta Bondowoso untuk membuatkan 1000 *startup*.

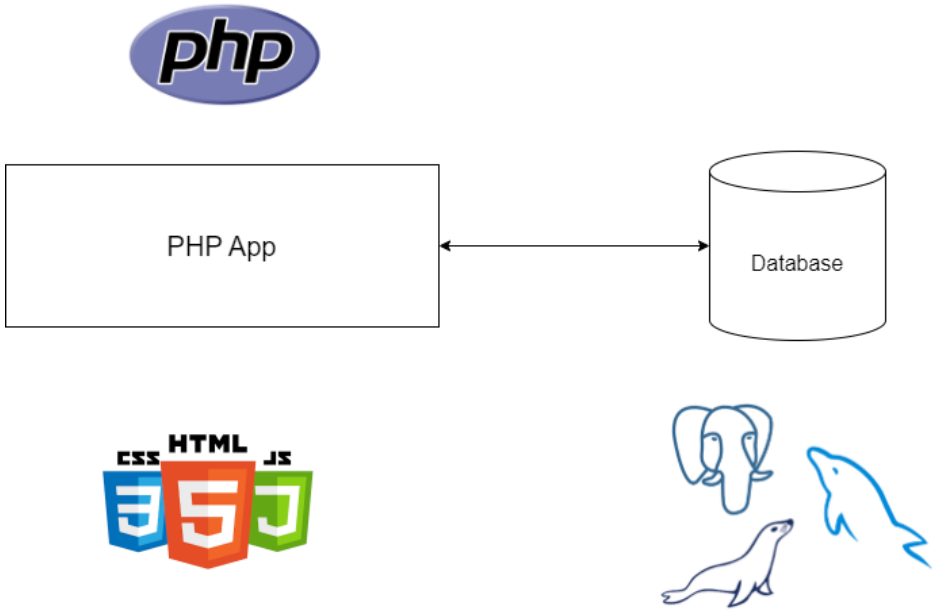
Bondowoso yang sangat ingin meminang Roro, menyanggupi keinginan Roro. Akan tetapi, Bondowoso acap kali bolos perkuliahan IF3110 Pengembangan Aplikasi Berbasis Web sehingga dia bahkan tidak mengetahui jika HTML itu bukan bahasa pemrograman. Seluruh waktunya hanya sibuk untuk melihat konten dan siniar kurang bermutu di media sosial yang beken pada zaman itu, yaitu BurBir (Burung Biru).

Oleh karena itu, kalian, para jin Bondowoso, ditugaskan untuk membantu mewujudkan 1000 *startup* menggunakan seluruh pengetahuan yang kalian miliki hingga sejauh ini. Tentunya Bondowoso akan sangat mengapresiasi kalian dengan memberikan kalian sebuah kejutan. Yuk bantu Bondowoso meminang Roro!

Tujuan Pembelajaran

- Mahasiswa mampu membuat sebuah aplikasi web dengan menggunakan HTML, CSS, dan JS.
- Mahasiswa mampu membuat sebuah layanan web dengan menggunakan PHP.
- Mahasiswa mampu menggunakan AJAX untuk melakukan pemanggilan yang bersifat *asynchronous*.
- Mahasiswa memahami penanganan *state* dan perbedaan HTTP *method*.

Spesifikasi Sistem



Gambaran Umum untuk Milestone 1

Spesifikasi Umum

Berikut ada beberapa ketentuan umum yang harus dipenuhi oleh sistem.

1. Untuk *client-side*, wajib menggunakan **JavaScript, HTML,** dan **CSS** secara **murni**. **Tidak diperbolehkan** menggunakan *framework* CSS atau *framework* JS (e.g. jQuery, lodash, Bootstrap, atau Tailwind). CSS harus berada di berkas yang berbeda dengan HTML (gunakan CSS *selector*). Tidak diperbolehkan **memanipulasi** CSS.
2. Untuk *server-side*, wajib menggunakan **PHP murni** tanpa kakas apapun (seperti laravel, codeigniter). Anda harus mengimplementasikan fitur menggunakan **HTTP method** yang tepat.
3. Untuk basis data, wajib menggunakan MySQL, MariaDB, atau PostgreSQL. Minimal terdapat 3 entitas berbeda dengan minimal 1 relasi ***one-to-many*** atau ***many-to-many*** antara entitas-entitas tersebut. Tidak diperbolehkan menggunakan ORM atau sejenisnya dan tidak boleh menyimpan *blob* atau *binary file* pada basis data.

Spesifikasi Teknis

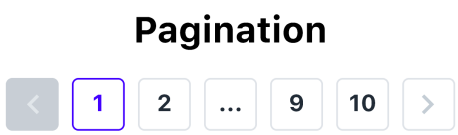
1. Autentikasi
    a. Pengguna dapat dibedakan menjadi **2 kategori**: *user* dan *admin*.
    b. Pengguna harus melakukan autentikasi untuk dapat mengakses seluruh fitur, **kecuali** disebutkan fitur dapat diakses oleh pengguna yang tidak terautentikasi.
    c. Seluruh pengguna yang terautentikasi juga harus dapat melakukan ***logout***.
2. Implementasi CRUD
    a. Setiap entitas yang ada dalam sistem (**minimal 3 entitas**) harus memiliki implementasi **CRUD** (Create, Read, Update, Delete).
    b. Pengguna dengan hak akses yang sesuai (*user* atau *admin*) harus dapat menggunakan fitur CRUD ini. Dapat disesuaikan dengan ide perangkat lunak.
    c. Untuk setiap aksi yang melakukan perubahan data, tampilkan **pesan konfirmasi aksi** (misalnya *popup window*) kepada pengguna sebelum eksekusi dilakukan.
    d. Pengguna dapat melakukan CRUD **berkas statik** (pilih **dua** dari: gambar, audio, atau video). Implementasi ketentuan ini dibebaskan dan dapat disesuaikan dengan ide perangkat lunak (contoh: tiap pengguna diasosiasikan dengan foto profil nya atau banyak audio musik seperti SoundCloud). Untuk ***method Read* berkas statik**, disesuaikan dengan jenis berkas yang diunggah. Contoh: Jika gambar, wajib bisa menampilkan gambar. Jika audio, wajib bisa dimainkan. Jika video, wajib bisa diputar videonya.
3. Implementasi Administrasi
    a. Jika pengguna adalah admin, pengguna dapat mengakses ke halaman administrasi yang menampilkan data entitas atau fitur khusus autentikasi admin (fitur dibebaskan sesuai ide produk).
    b. Pengguna sebagai admin **wajib dapat mengelola data** (menambah, memperbarui, menghapus, dan membaca).
4. Terdapat *navigation bar* di **setiap halaman** yang dibuat. Minimal terdapat 2 pranala untuk menuju halaman lain.
5. Search Bar
    a. Pengguna (*user* dan *admin*) dapat **mencari** (*substring*)**, menyortir, dan mem-*filter*** entitas berdasarkan minimal 2 atribut dari entitas tersebut untuk masing-masing fungsi. Contoh: Untuk ide P/L Spotify, mencari lagu berdasar nama artis atau judul lagu, menyortir berdasar *timestamp* dan

judul lagu, mem-*filter* berdasarkan genre dan penyanyi. Ketiganya wajib **dapat dilakukan secara bersamaan**.

b. Search Bar diimplementasikan **minimal 1** dalam sebuah halaman daftar suatu entitas bersamaan dengan pagination.

c. Wajib melakukan implementasi ***debounce*** pada pencarian untuk menghindari pemanggilan yang berlebihan.

6. *Pagination*



a. Di setiap halaman yang menampilkan daftar entitas (misalnya, daftar produk), **tampilkan sejumlah entitas** yang sesuai dengan batasan tertentu per halaman.

b. Tampilkan navigasi (halaman berikutnya dan sebelumnya) atau tautan angka halaman untuk membantu pengguna pindah ke halaman lain.

c. *Pagination* **wajib dilakukan secara *server-side***.

7. Keamanan
   a. *Client-side* wajib melakukan **validasi masukan** sederhana dari pengguna (misalnya *required field*, regex untuk nomor *handphone*, etc).
   b. *Server-side* wajib melakukan **validasi data** dari pengguna untuk memastikan integritas data (data unik, keberadaan data, logika, etc).
8. Implementasikan ***error handling*** untuk memberikan umpan balik yang informatif kepada pengguna.
9. Responsivitas
   a. Implementasikan minimal **satu** halaman yang **responsif** (minimal untuk ukuran 1280 x 768 dan 400 x 800). Artinya, tampilan mungkin berubah menyesuaikan ukuran layar.

Pesan dari Asisten: "jangan bare minimum kalo bisa karena nanti nilainya bare minimum juga hehe. udah pake sistem penilaian baru 😀"

Bonus

Catatan: Kerjakan dahulu spesifikasi wajib sebelum mengerjakan bonus.

1. ***All Responsive Web Design***

Semua tampilan dibuat **responsif** (minimal untuk ukuran 1280 x 768 dan 400 x 800). Artinya, tampilan mungkin berubah menyesuaikan ukuran layar. Hint: gunakan CSS @media rule, lebih lanjut: https://www.w3schools.com/css/css_rwd_mediaqueries.asp.

2. **Docker**
   Membuat Dockerfile dari aplikasi kalian, serta membuat file docker-compose.yml dari aplikasi kalian yang berisi aplikasi kalian serta database yang digunakan. Tujuan bonus ini adalah agar kalian dapat mendapatkan pengalaman *hands-on* menggunakan Docker. Bonus ini juga mempermudah pengerjaan untuk milestone selanjutnya.

3. **Google Lighthouse**
   Google Lighthouse adalah alat otomatis *open-source* untuk meningkatkan kualitas halaman web. Lighthouse dipakai sebagai alat pengukuran dan audit untuk performance kualitas website, aksesibilitas, aplikasi web progresif, dan banyak lagi. Tugas anda adalah melakukan pengecekan skor di Lighthouse untuk seluruh halaman dan pastikan bahwa skor untuk ***performance*, aksesibilitas, dan *best practices*** yang didapatkan memiliki **nilai di atas 95**. Lampirkan bukti tangkapan layar pada README.

Keyword & Tips

Untuk meringankan beban tugas ini, ada beberapa *keyword* yang bisa Anda cari untuk menyelesaikan tugas ini.

- **HTTP methods**: GET, POST, PUT, PATCH, DELETE, OPTIONS.
- **CSS**: margin, padding, font-size, text-align, flex, grid, border, color, div, span, anchor tag, box-shadow.
- **Javascript**: XMLHTTPRequest, addEventListener.
- **PHP**: mysqli_connect, mysql_query, pg_connect, pg_query, PDO, $_GET, $_POST, $_COOKIE, var_dump, print_r, echo, require.
- **SQL**: SELECT, INSERT, UPDATE, DELETE, WHERE, LIKE, ORDER BY.
- **DOCKER**: Dockerfile, Docker Compose, Docker Container, Docker Image, Docker Volume.

Berikut adalah *tips* dari asisten tercinta.

- Usahakan untuk membuat code yang maintainable (*reusable*, *loosely coupled*, dll). Sebagai contoh, dalam membuat *navbar*, jangan meng-*copy* navbar pada setiap berkas php namun manfaatkan **PHP templating**, lebih lanjut: https://css-tricks.com/php-templating-in-just-php/.
- Jika ingin eksperimen menggunakan Docker, pahami dulu istilah-istilah yang dipakai pada Docker, jangan langsung mencoba, karena akan sulit apabila tidak berjalan atau menemukan *error*. Misalnya, pahami dulu perbedaan pembuatan *docker image* dan *docker container*. Lalu, ketahui cara kerja *docker container* mulai dari *port forwarding*, *docker volume*, dan *docker network*.
- Jangan berantakan saat membuat kode PHP. Ini saran agar keberlangsungan milestone 2 kalian lancar dan tidak terhambat oleh milestone 1 ini. Apabila tugas besar ini dipakai untuk matkul lain, alur program dapat dibaca dan diingat dengan cepat.

Responsi

Diadakan responsi per kelompok bersama asisten menjelang pengumpulan tugas.

- Responsi bersifat **opsional**. Artinya **tidak wajib**.
- Kelompok yang ingin mendaftarkan diri dapat mengisi pada pranala berikut.
- Deadline pengisian jadwal responsi adalah **23 September 2023 pukul 23.59 WIB.**
- Perlu diperhatikan bahwa ketersediaan asisten **terbatas**.
  Siapkan terlebih dahulu hal-hal yang ingin ditanyakan / dikonsultasikan dan lakukan setup project di komputer anda **sebelum** sesi responsi dimulai.
- Pastikan sudah riset dan **mencoba** mempelajari sebelum bertanya ke asisten. Agar bisa bertanya lebih banyak pertanyaan dan pertanyaan lebih dekat ke pengerjaan bukan *understanding*.
- Silahkan **membuat undangan** di Google Calendar sesuai jadwal yang telah diisi menggunakan email STD asisten.

Daftar Pertanyaan

Jika masih ada pertanyaan mengenai spesifikasi tugas, dapat dilakukan melalui QnA berikut.

Pastikan tidak ada pertanyaan yang berulang.

Lain-Lain

Deliverables

Berikut adalah hal yang harus diperhatikan untuk pengumpulan tugas ini:

1. Buatlah grup pada Gitlab dengan format "IF3110-2023-01-XX", dengan XX adalah nomor kelompok (untuk K1 dan K2) atau kode kelompok (untuk K3).
2. Tambahkan anggota tim pada grup anda.
3. ***Fork*** pada repository ini dengan organisasi yang telah dibuat.
4. Ubah hak akses repository hasil *fork* anda menjadi ***private***.
5. Hal-hal yang harus diperhatikan.
    a. Silakan *commit* pada *repository* anda (hasil *fork*).
    b. Lakukan beberapa *commit* dengan pesan yang bermakna, contoh: “add register form”, “fix logout bug”, jangan seperti “final”, “benerin dikit”, “fix bug”.
    c. Disarankan untuk tidak melakukan *commit* dengan perubahan yang besar karena akan mempengaruhi penilaian (contoh: hanya melakukan satu *commit* kemudian dikumpulkan).
    d. *Commit* dari setiap anggota tim akan mempengaruhi penilaian.
    Jadi, setiap anggota tim harus melakukan commit yang berpengaruh terhadap proses pembuatan aplikasi.
    e. Sebagai panduan, anda bisa mengikuti *semantic commit*.
6. Buatlah berkas **README** yang berisi:
    a. Deskripsi aplikasi web
    b. Daftar *requirement*
    c. Cara instalasi
    d. Cara menjalankan *server*
    e. Tangkapan layar tampilan aplikasi (tidak perlu semua kasus, minimal 1 per halaman), dan
    f. Penjelasan mengenai pembagian tugas masing-masing anggota (lihat formatnya pada bagian pembagian tugas).

Pengumpulan Tugas

- Tenggat waktu tugas adalah pada hari **6 Oktober 2023 pukul 15.00 WIB**.
- Tahap pengumpulan adalah sebagai berikut.

a. Buat *release* dengan format **vX** pada Gitlab Informatika dan judul bebas tetapi bermakna.
   X adalah adalah nomor yang dimulai dari 1.
b. Apabila anda ingin merevisi tugas, buat *release* dengan penambahan angka sebelumnya.
   Contoh, kelompok anda sudah merilis v1, tetapi ternyata ada revisi sedikit, silakan buat rilis v2.
   Tugas yang dinilai adalah *release* versi terakhir.
c. Waktu pengumpulan tugas yang dilihat adalah **waktu *release version* terakhir ke *server* Gitlab** terakhir.
   Akan ada pengurangan terhadap *release version* yang melebihi waktu tenggat waktu tugas.

- Alasan tenggat waktu jam 3 sore adalah pencegahan apabila terjadi *error* pada *server* Gitlab Informatika. Karena pada jam tersebut masih ada staf yang dapat menangani hal tersebut.
- Bagi **kelompok dengan mahasiswa yang tidak bisa melakukan akses ke Gitlab Informatika**, dapat menggunakan *platform repository management* lain seperti Github atau Bitbucket sebagai wadah pengembangan. Untuk repository tetap dalam keadaan **private** sebelum tenggat waktu. Lakukan release dengan ketentuan yang sama seperti release version Gitlab. Pengumpulan akan dilakukan ke **form berikut**.
- (Opsional, hanya saran) Silakan buat git cadangan ke platform lain seperti Github, Gitlab (bukan yang gitlab informatika) atau Bitbucket, sebagai git cadangan apabila Gitlab Informatika ada masalah. Sehingga, apabila gitlab informatika sedang mati sementara atau *down* masih bisa mengerjakan.
  a. Caranya, tinggal tambah *remote repository* menggunakan *command* git.
  b. Saat git push, targetkan ke *remote repository* cadangan.
  c. Kalau git pull, targetkan ke *remote repository* yang memiliki *commit* paling baru.

Pembagian Kelompok (K1 dan K2)

Pembagian kelompok dilakukan untuk **K1 dan K2**. Untuk K3, dapat menggunakan pembagian dan kode kelompok yang telah dibagikan dosen sebelumnya.

Pembagian kelompok ada di pranala berikut.

Bagi yang belum mendapatkan kelompok dapat mengontak asisten melalui Teams.

Pembagian Tugas

Setiap anggota kelompok **diwajibkan** untuk mengerjakan bagian *server-side* (PHP) dan *client-side* (HTML, CSS, JS), dengan harapan kalian bisa mencoba mempelajari semua bagian dan cara kerjanya.

Server-side

Login : 13520xxx, 13520xxx

Register : 13520xxx

(Lanjutkan ...)

Client-side

Login : 13520xxx, 13520xxx

Register : 13520xxx

(Lanjutkan...)

Catatan Tambahan

Milestone ini akan berhubungan dengan *milestone* selanjutnya; artinya tugas yang kalian buat di sini **akan digunakan kembali pada *milestone* berikutnya**. Usahakan untuk menyelesaikan spesifikasi utama agar tidak terhambat pada milestone berikutnya.

Sangat disarankan untuk mengerjakan **bonus 2 (Docker)** karena akan membantu **memudahkan** kalian untuk *milestone* berikutnya.