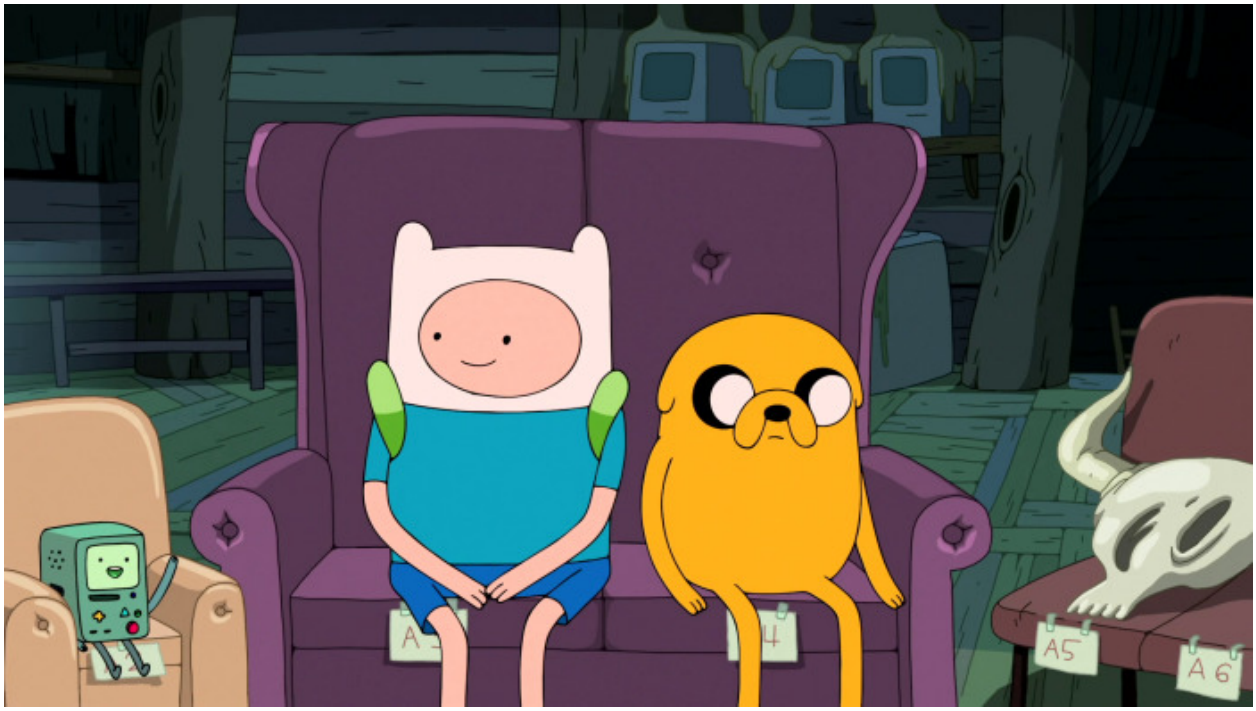
Spesifikasi Tugas Besar 2 IF3110

Milestone 2 - Web Services using SOAP and REST

Revisi 1: Penambahan teknik dan informasi mengenai teknik revalidasi *endpoint check status* Service SOAP (19 November 2022)
Revisi 2: Penambahan tombol play pada halaman list lagu premium (26 November 2022)
Revisi 3: Mengubah Penjelasan API Key Soap Service (28 November 2022)

Deskripsi Persoalan



"BNMO, Doni, dan Jak yang sedang membahas sistem bisnis yang baru"

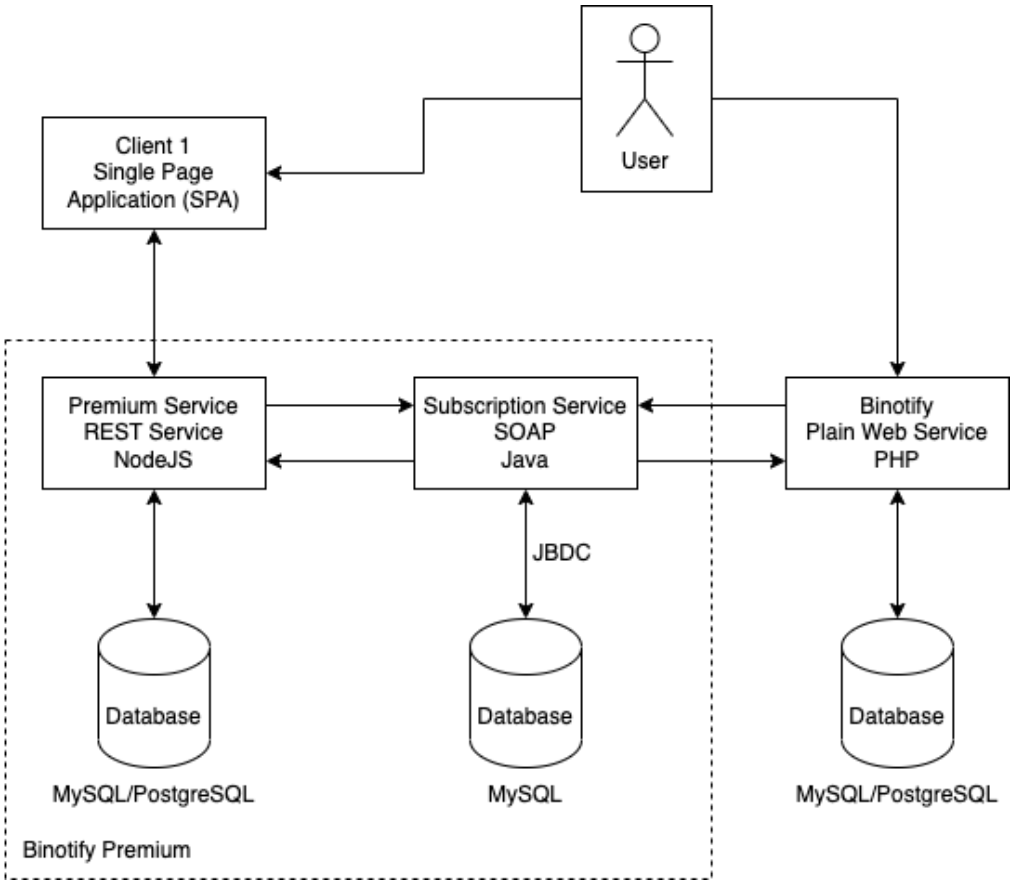
BNMO (dibaca: Binomo) dan Doni akhirnya dapat mendengar lagu "Tiba-Tiba" dengan puas dengan bantuan Indra. Jak yang sering melihat mereka menggunakan Binotify lalu bertanya kepada mereka mengenai aplikasi itu dan karena Jak merupakan seorang pebisnis, Jak mendapatkan sebuah ide, yaitu membuat sistem *subscription*. Jak mengajukan ide ke BNMO dan Doni, dan mereka menyukai ide tersebut, tetapi Jak ingin menggunakan jasa temannya untuk membuat sistem *subscription* tersebut dan teman Jak hanya ingin mengerjakannya dengan SOAP. Kesepakatan akhir yang ditentukan adalah teman Jak dan Indra akan bekerjasama untuk membuat sistem bisnis baru, yaitu dengan menggunakan aplikasi yang telah dibuat, SOAP, NodeJS, ReactJS, Karena Indra sangat terbantu oleh mahasiswa Informatika yang sudah membantu membuat Binotify, Indra meminta bantuan dari mahasiswa Informatika sekali lagi.

Tujuan Pembelajaran

- Mahasiswa mampu membuat sebuah *web service* dengan protokol SOAP.
- Mahasiswa mampu membuat sebuah *web service* yang mengimplementasikan REST.
- Mahasiswa mampu membuat sebuah aplikasi *web* yang memanggil *web service* yang mengimplementasikan REST.
- Mahasiswa mampu membuat sebuah aplikasi *web* yang memanggil *web service* dengan protokol SOAP.

Spesifikasi Sistem

Gambaran Umum Sistem



Gambaran Umum "Binotify" dan "Binotify Premium" untuk Milestone 2.

Spesifikasi Umum

Sebagai gambaran umum, sistem yang akan dikerjakan pada milestone kedua adalah sistem Binotify Premium untuk melakukan langganan lagu premium oleh seorang penyanyi. Pengguna Binotify App pada milestone sebelumnya dapat melakukan langganan pada penyanyi yang sudah terdaftar di milestone ini.

Berikut ringkasan hal-hal yang kalian kerjakan pada milestone ini:

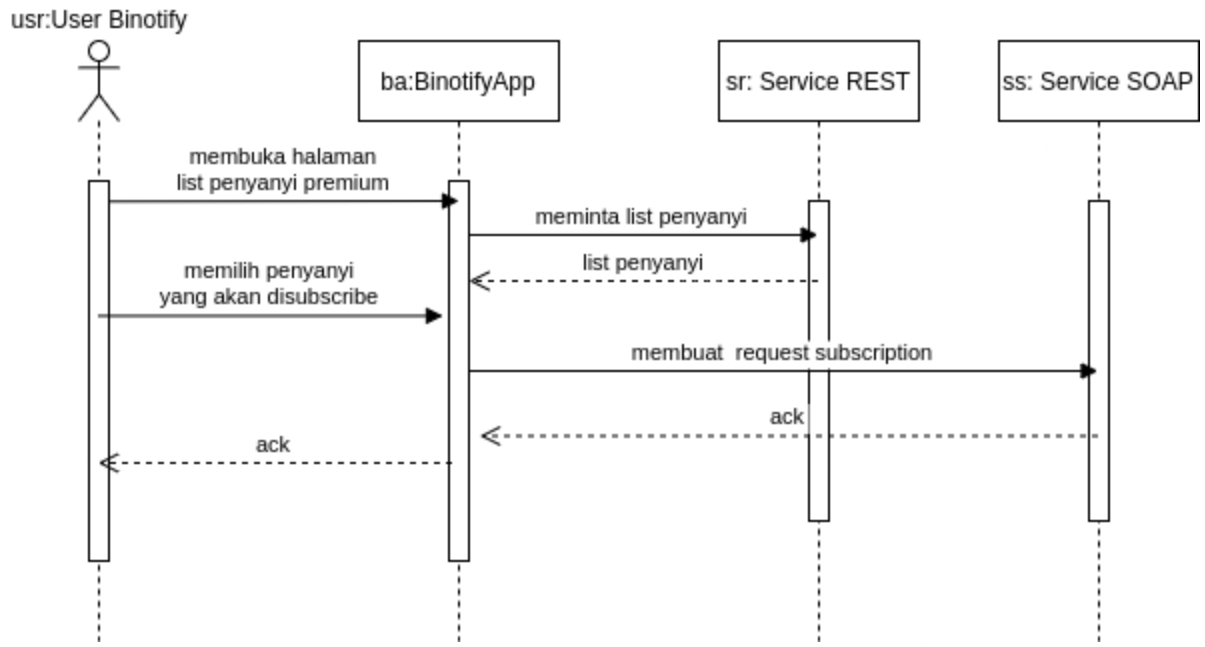
1. Membuat REST service yang bisa menangani pengelolaan lagu premium oleh seorang penyanyi. Service ini wajib dibuat menggunakan bahasa **NodeJS** (haha kalian gagal NNN: No Node November). Kalian juga akan membuat clientnya, menggunakan **ReactJS/ReactTS**, untuk admin Binotify Premium serta penyanyi dapat menggunakan service ini.

2. Membuat SOAP service yang bisa menangani pengajuan request subscription, serta menerima approval/rejection dari admin Binotify Premium. Service dibuat menggunakan **JAX-WS** (Java Servlet).
3. Mengubah beberapa skema basis data dari Binotify App, supaya pengguna dapat melakukan pencarian penyanyi yang menyediakan lagu premium dan bisa meminta untuk berlangganan ke penyanyi.

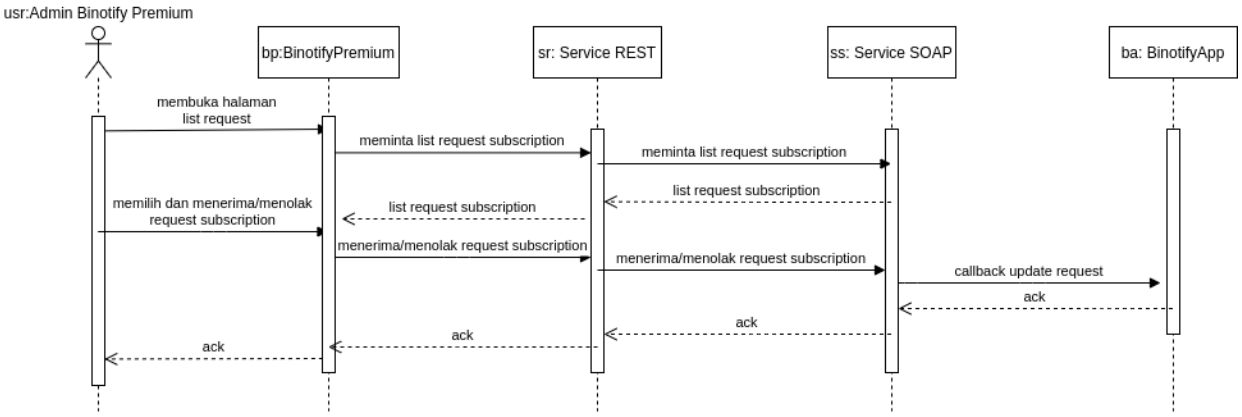
User Story

Membangun sistem Binotify Premium yang dapat melakukan CRUD lagu premium dan *approve/reject* permintaan *subscription*.

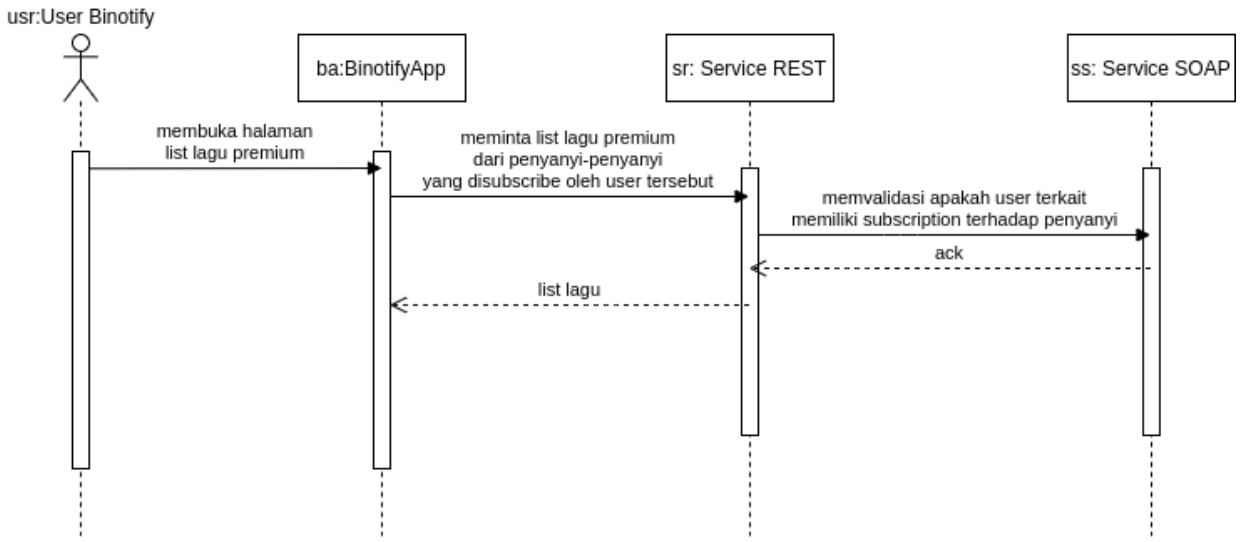
1. Autentikasi
    a. Pengguna harus melakukan autentikasi untuk dapat mengakses seluruh fitur.
    b. Pengguna dibedakan menjadi 2 kategori: *penyanyi* dan *admin*.
    c. Seluruh pengguna yang terautentikasi juga harus dapat melakukan *logout*.
2. Penyanyi dapat melakukan pengelolaan lagu premium miliknya.
3. Admin Binotify Premium dapat melihat list permintaan *subscription*.
4. Admin Binotify Premium dapat menerima atau menolak permintaan *subscription*.
5. Pengguna Binotify (aplikasi PHP pada milestone sebelumnya) dapat membuat permintaan *subscription*.
6. Pengguna Binotify dapat melakukan eksplorasi nama penyanyi yang menyediakan lagu premium.



Alur pengguna Binotify melakukan permintaan subscription terhadap lagu premium



Alur admin Binotify Premium melakukan approval permintaan subscription



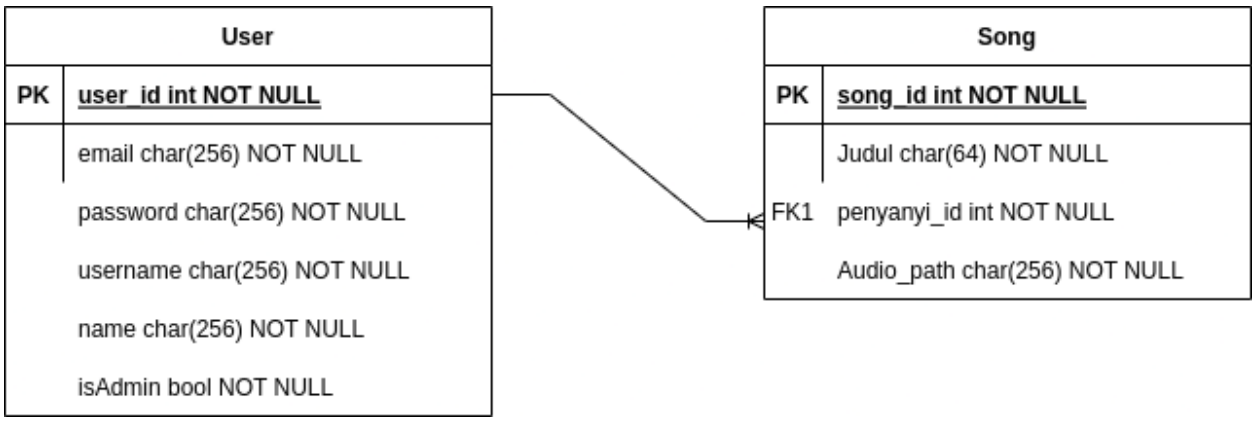
Alur pengguna Binotify melakukan permintaan lagu premium

Spesifikasi Fitur: *Service* REST

Berikut ada beberapa ketentuan terkait fitur-fitur apa saja yang ada di dalam sistem.

1. ***Database***

   Berikut adalah data yang harus disimpan oleh *service* REST. Seorang penyanyi dapat membuat lagu premium. Lagu tersebut hanya menyimpan judul dan file audio saja supaya sederhana.



2. **Autentikasi Pengguna**

   Ketentuan autentikasi sama seperti yang telah dijelaskan di *user story*. Standar yang dipakai pada service ini adalah JWT atau Oauth 2.0 (diperbolehkan menggunakan library).

3. **Pengelolaan Lagu Premium**

   Service perlu menyediakan endpoint untuk melakukan *create, read, update,* dan *delete* dari lagu premium yang dimiliki oleh penyanyi. *Field* yang bisa diupdate oleh penyanyi adalah judul dan file audionya.

4. ***Endpoint* List Penyanyi**

   Service perlu menyediakan *endpoint* untuk melakukan *read/fetch* list penyanyi. Perhatikan bahwa *response* dari *endpoint* ini jarang berubah, kecuali ada penyanyi baru yang register.

5. **Endpoint List Lagu dari Penyanyi**

   Service juga perlu menyediakan *endpoint* untuk melakukan *read/fetch* list lagu dari penyanyi. Perhatikan bahwa *endpoint* ini harus menerima minimal 2 data, yaitu pengguna Binotify yang melakukan *request* serta penyanyi yang diminta. Lalu *service* REST akan meminta validasi kepada *service* SOAP jika pengguna Binotify terkait sudah subscribe terhadap penyanyi yang diminta.

Spesifikasi Fitur: *Service* SOAP

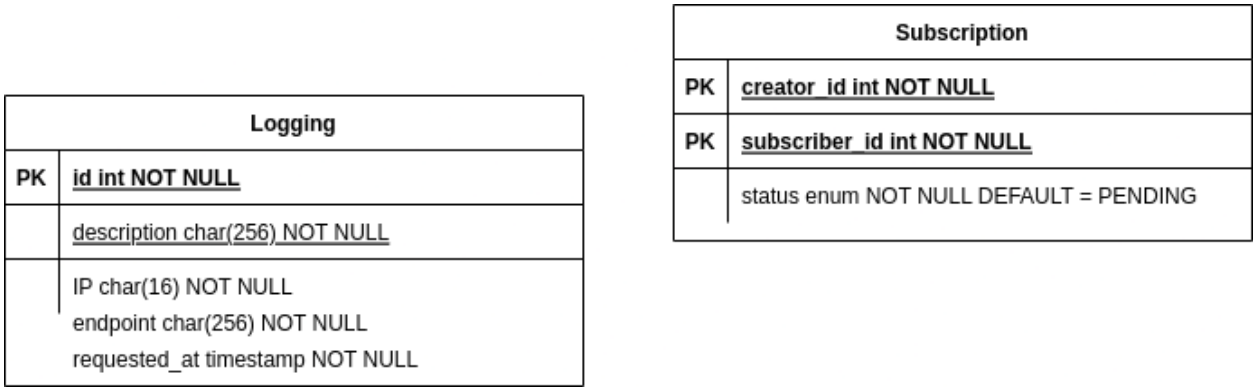
Berikut ada beberapa ketentuan

1. ***Security***

   Security merupakan fitur yang paling penting pada web aplikasi apapun. Karena hampir semua aplikasi web akan di-ekspos pada internet, akan ada kesempatan terhadap ancaman keamanan terhadap aplikasi web. Pada tugas besar kali ini, terdapat beberapa *best practice* yang harus diimplementasikan untuk meningkatkan keamanan aplikasi kalian:

   - Membuat *logging* untuk semua *request* yang masuk ke *service* SOAP. Data yang tersimpan pada *database* SOAP minimal memiliki *field* **deskripsi request**, **ip**, **endpoint** yang dituju, dan **timestamp** saat itu.
   - ~~Membuat *API key* untuk semua hal yang berhubungan dengan *subscription* (termasuk dengan *callback request* ke Binotify App). Service SOAP perlu men-***generate*** untuk setiap *client* yang melakukan *request* yang berhubungan dengan *subscription*, lalu *API key* tersebut perlu dimasukkan ke dalam setiap *request* yang dikirim oleh *client*.~~
   - Membuat *API Key* untuk kunci memanggil *SOAP Service/Subscription Service* (termasuk *callback request* ke Binotify App). Silakan **generate** kunci (secara manual saja) dan simpan di *environment file* setiap service. Jangan lupa untuk validasi *API Key* yang telah kalian buat di *SOAP Service*. Fitur ini berfungsi seperti *password* yang dimiliki *SOAP Service*. *API Key* yang ada di Binotify App dan REST Service **harus berbeda** agar *SOAP Service* dapat membedakan service mana yang memanggilnya.

2. ***Database***



Untuk *field description* pada tabel *logging* di atas berisi deskripsi atas *request* yang dilakukan. Untuk tabel *subscription*, status yang dimiliki berupa *enum*: PENDING, ACCEPTED, dan REJECTED. Khusus *service* SOAP, database yang dipakai MySQL, agar memudahkan koneksi dengan JDBC.

3. **Menerima Permintaan *Subscription* dari Binotify App**

   Saat data request subscription, admin web app dapat melihat dan melakukan approval terhadap request subscription dengan memanggil service REST, dan dibelakang service REST terdapat service SOAP yang sebenarnya melakukan approval.

4. **Menerima Penerimaan / Penolakan Permintaan *Subscription***

   Saat admin Binotify Premium telah melakukan *approval/rejection*, maka *service* SOAP akan mengirim *callback* ke Binotify App sehingga data *subscription status*-nya dapat diperbaharui.

5. **Endpoint *Check Status* Permintaan**

   Endpoint ini berguna untuk menghindari adanya potensi kegagalan *callback*, sehingga Binotify App dapat melakukan revalidasi data secara *polling*. Selain itu *endpoint* ini bisa dipakai dengan revalidasi seperti biasa (melakukan *refresh page* list penyanyi premium di Binotify App, misalnya). Disarankan menggunakan strategi *stale-while-revalidate*. *Endpoint* ini juga dipakai untuk memvalidasi *request* lagu premium yang diminta oleh *service* REST.

Spesifikasi Halaman Binotify Premium

1. **Halaman *Login***

   Halaman yang ditampilkan jika pengguna belum *login* atau sesudah *logout*. Pengguna dapat melakukan *login* sebagai *penyanyi* atau *admin*. Gunakan *service* REST untuk login sesuai dengan spesifikasi yang sudah dijelaskan. Setelah login, pengguna akan diarahkan ke halaman yang sesuai dengan peran mereka, admin

akan diarahkan ke halaman **List Permintaan *Subscription*** sedangkan penyanyi akan diarahkan ke halaman **Pengelolaan Lagu**.

2. **Halaman *Register***

   Pengguna dapat mendaftarkan akun baru jika belum *login* atau sudah *logout*. Pada halaman ini, pengguna mendaftarkan diri pada sebuah form dengan *field* nama, *username* (unik), *email* (unik), *password*, dan *confirm password*. Pengguna hanya dapat mendaftar sebagai penyanyi karena admin ditambahkan secara manual pada basis data.

   Validasi lain yang dilakukan pada sisi klien pada halaman ini adalah: *Email* memiliki format email standar seperti "example@example.com". *Username* hanya menerima kombinasi alphabet, angka, dan underscore.

   Setelah semua nilai *field* sudah diisi dan *valid*, pengguna dapat mendaftarkan akun barunya. Jika akun berhasil didaftarkan, pengguna langsung diarahkan ke halaman yang sesuai dengan perannya, dalam kasus ini ke halaman **Pengelolaan Lagu**, karena akun selalu memiliki peran sebagai penyanyi jika baru didaftarkan. Mekanisme autentikasi seperti yang dijelaskan sebelumnya.

3. **Halaman List Permintaan *Subscription***

   Halaman yang dapat diakses admin untuk menerima atau menolak permintaan *subscription* yang datang dari pengguna Binotify. Perhatikan bahwa pada halaman ini web app Binotify Premium **tidak** langsung berkomunikasi dengan *service* SOAP, melainkan berkomunikasi hanya dengan *service* REST yang akan kemudian dilanjutkan ke *service* SOAP sesuai dengan *sequence diagram* yang sudah diberikan.

   Gunakanlah ***pagination*** pada halaman ini dengan jumlah permintaan per halaman yang kalian tentukan sendiri. *Pagination* boleh diimplementasikan secara *server-side* maupun *client-side*.

4. **Halaman Pengelolaan Lagu**

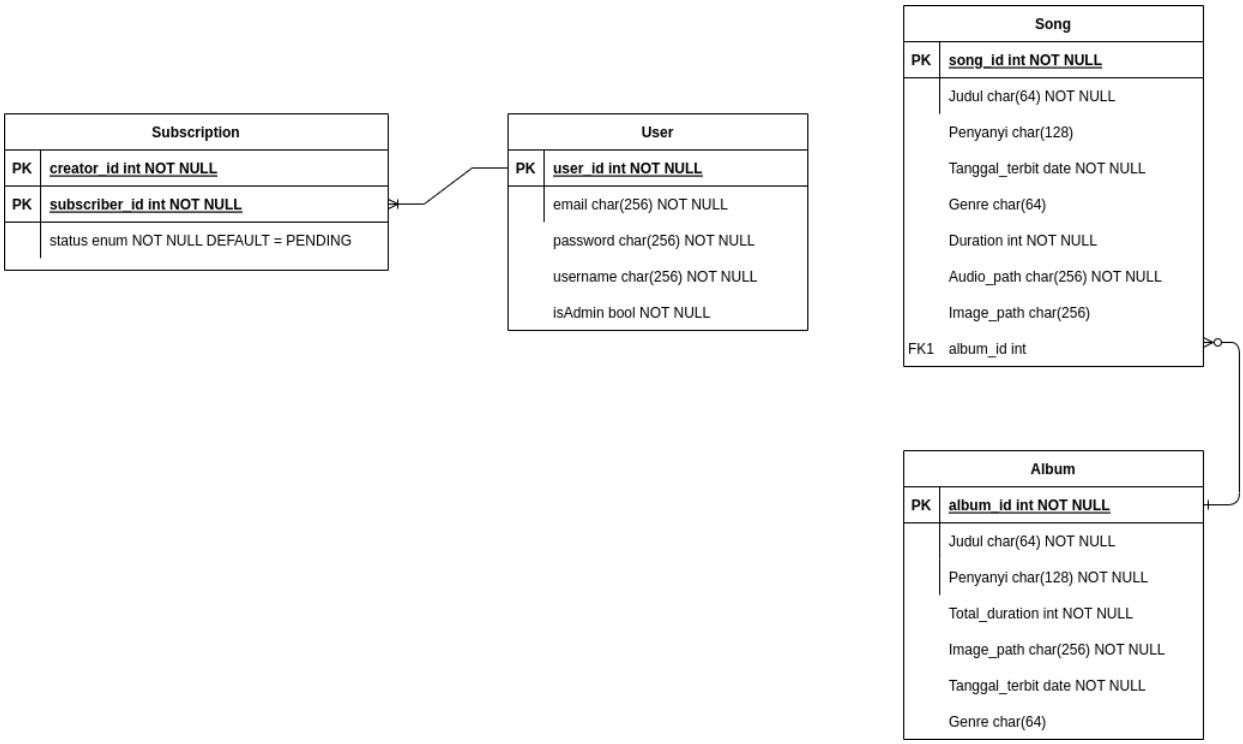
   Halaman yang dapat diakses penyanyi untuk mengelola lagu-lagu mereka. Pada halaman ini penyanyi dapat **menambah**, **menghapus**, **mengubah**, dan **melihat** lagu-lagu premium yang ditawarkan. Perhatikan bahwa lagu-lagu yang dapat dikelola seorang penyanyi adalah lagu-lagu mereka sendiri, penyanyi tidak dapat mengelola lagu penyanyi lain. Field yang dapat diedit oleh penyanyi adalah judul dan juga file audio lagu tersebut.

   Gunakanlah ***pagination*** pada halaman ini dengan jumlah lagu per halaman yang kalian tentukan sendiri. *Pagination* boleh diimplementasikan secara *server-side* maupun *client-side*.

Spesifikasi Tambahan Binotify App

1. Perubahan *Database*

Terdapat perubahan skema *database* pada Binotify App digambarkan sebagai berikut:



Tabel *subscription* akan berisi status *subscription* dari pengguna. *Enum* dari statusnya adalah PENDING, ACCEPTED, dan REJECTED. Tabel ini merupakan “duplikasi” dari tabel di *service* SOAP, supaya untuk mengetahui *status subscription* pengguna tanpa perlu melakukan *request* apakah pengguna sudah *subscribe* atau belum. Data dari tabel ini akan berisi data yang dikirim via *callback endpoint*.

2. Halaman List Penyanyi Premium

Tambahkan sebuah halaman dengan spesifikasi sebagai berikut:

- Halaman ini akan memanfaatkan *endpoint* dari *service* REST.
- Menampilkan daftar nama penyanyi yang menyediakan lagu premium.
- Untuk setiap penyanyi pada daftar, tambahkan sebuah tombol untuk melakukan *subscribe* kepada penyanyi tersebut apabila pengguna belum *subscribe*. Jika sudah *subscribe*, maka akan ada tombol yang jika diklik akan diteruskan ke halaman list lagu premium dari penyanyi tersebut.
- Ubah sidebar/navbar sehingga ada menu untuk menuju hal ini.

3. Halaman List Lagu Premium

Tambahkan sebuah halaman dengan spesifikasi sebagai berikut:

- Halaman ini akan memanfaatkan *endpoint* dari *service* REST.
- Menampilkan daftar lagu premium dari penyanyi yang telah di-*subscribe* oleh pengguna.
- Tiap lagu cukup ditampilkan judul dan tombol play

4. ***Endpoint Callback***

   Tambahkan *endpoint callback* dari *service* SOAP yang menerima *field* yang sesuai dengan tabel *subscription* sebagai berikut:

   - Subscriber_id
   - Creator_id
   - Status

   Endpoint dapat bersifat REST (jadi nanti *service* SOAP akan memanggil menggunakan *client* REST). Perhatikan sifat-sifat *endpoint callback*, seperti *HTTP method* yang dipakai, *idempotency*, *security*, dll.

Bonus

Catatan: Kerjakan dahulu spesifikasi wajib sebelum mengerjakan bonus.

1. ***Responsive Web Design***

   Tampilan dibuat **responsif** (minimal untuk ukuran 1280 x 768 dan 800 x 600). 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. Tidak perlu membuat dockerfile yang kompleks, asalkan bisa dijalankan (kalian bisa saja membuat Dockerfile yang hanya berisi 3 baris). Tujuan bonus ini adalah agar kalian dapat mendapatkan pengalaman *hands-on* menggunakan docker.

Sebagai catatan, untuk men-*dockerize* React usahakan untuk melakukan serving hasil static build aplikasi kalian.

3. Notifikasi Email

Ketika ada permintaan *subscription* baru, **admin Binotify Premium** menerima email notifikasi. Admin yang mendapatkan email bisa salah satu admin saja atau seluruhnya. Notifikasi dikirim dari ***service* SOAP** (perhatikan kalian perlu melakukan pemanggilan ke *service* REST untuk mendapatkan email admin) menggunakan pustaka JavaMail atau pustaka lainnya. Pakai mailtrap untuk testing agar inbox Anda tidak penuh dengan email testing.

4. ***Caching***

   Melakukan *query* ke *database* merupakan operasi yang mahal. *Caching* akan membantu meningkatkan performa aplikasi kalian secara signifikan pada traffic yang tinggi. *Caching* bisa dilakukan pada *endpoint list* penyanyi di ***service* REST,** karena pada *endpoint* tersebut jarang dilakukan perubahan, kecuali ada user baru yang mendaftar. *Caching* dapat dilakukan pada level *query* maupun pada level *response* sebuah *request*. *Service / stack* yang umum digunakan untuk *caching* adalah Redis.

Bantuan

Referensi

Berikut adalah referensi dari asisten tercinta.

- https://www.baeldung.com/jax-ws
- https://betterprogramming.pub/how-to-perform-soap-requests-with-node-js-4a9627070eb6
- https://roytuts.com/consume-soap-web-service-in-php/
- https://www.baeldung.com/java-email

Daftar Library

Berikut ada beberapa library yang disarankan oleh asisten. Kalian tidak harus mengikuti apa yang disarankan di sini, tapi bisa menjadikan ini referensi.

Untuk Java, diwajibkan menggunakan Jax-WS, sehingga kami tidak akan menyarankan library yang lain.

Untuk NodeJS, ada beberapa library yang disarankan:

1. Express (https://expressjs.com/), web framework NodeJS yang paling terkenal.
2. Fastify (https://www.fastify.io/), web framework yang low-overhead dan secara benchmark mengalahkan Express.
3. Hapi (https://hapi.dev/), web framework dengan tujuan untuk menjaga keamanan.

Tetapi, **routing, model, dan handler function** wajib dibuat sendiri. Tidak diperkenankan library/framework yang melakukan generate/menghindari *coding* seperti Content Management System (Library **strapi**).

Untuk React, diperbolehkan untuk menggunakan library styling apapun (seperti TailwindCSS, WindiCSS, Bootstrap, Bulma, dll.). Untuk development server dan hal terkait ada beberapa opsi yang disarankan:

1. Vite (https://vitejs.dev/), menyediakan development server yang menggunakan Rollup dan lebih cepat dibandingkan CRA. (**RECOMMENDED, RIL NO FEK)**
2. create-react-app/CRA (https://create-react-app.dev/), cara standar yang disarankan dokumentasi React untuk membuat aplikasi React yang menggunakan Webpack.
3. Melakukan setup lewat Webpack sendiri.
4. Melakukan serving React melalui server framework seperti Express.

Typescript **diperbolehkan** dan **sangat disarankan**. (https://www.typescriptlang.org/)

Tips

Berikut adalah *tips* dari asisten tercinta.

- Usahakan untuk membuat code yang maintainable (reusable, loosely coupled, dll).
- Kalau mau eksperimen 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*.
- Usahakan untuk mencicil tubes, supaya kalian bisa tidur di tengah badai tubes ini 😀.
- Jangan **deadliner**. Pasti *gak* selesai.

Responsi

Apabila tutorial diatas masih sulit untuk dicerna. Akan diadakan responsi per kelompok bersama asisten menjelang pengumpulan tugas (tanggalnya menyusul).

- Responsi bersifat **opsional**. Artinya **tidak wajib** dan jika mau saja.
- Kelompok yang ingin mendaftarkan diri dapat mengisi pada teams.
- 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*.

Daftar Pertanyaan

Jika masih ada pertanyaan mengenai spesifikasi tugas, dapat dilakukan melalui sheet teams.

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 "IF3110-2022-KXX-02-YY", dengan XX adalah nomor kelas dan YY adalah nomor kelompok.
2. Tambahkan anggota tim pada grup anda.
3. Buatlah **empat** repository private berikut pada grup tersebut:
    a. Binotify App (PHP + HTML-CSS-JS, gunakan aplikasi Binotify salah satu anggota anda sekarang)
    b. Binotify REST Service (Rest, NodeJS)
    c. Binotify SOAP Service (SOAP, JAX-WS)
    d. Binotify Premium App (ReactJS)
    e. (opsional) Binotify Config (berisi config untuk melakukan *compose* pada Docker dari repo yang lain, silakan buat untuk mempermudah bonus docker)
4. Hal-hal yang harus diperhatikan.
    a. Silakan commit pada repository anda.
    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. Sebaiknya *commit* dilakukan setiap ada penambahan fitur.
    e. *Commit* dari setiap anggota tim akan mempengaruhi penilaian.
    Jadi, setiap anggota tim harus melakukan commit yang berpengaruh terhadap proses pembuatan aplikasi.
    f. Sebagai panduan bisa mengikuti semantic commit.
5. Perbaruilah file README di repository Binotify App dengan menyebutkan perubahan yang kalian implementasikan.

6. Tambah file README di repositori Binotify SOAP Service dan Binotify REST Service, minimal berisi:
    a. Deskripsi singkat web service
    b. Skema basis data yang digunakan
    c. (opsional) Endpoint, payload, dan response API
    d. Penjelasan mengenai pembagian tugas masing-masing anggota (lihat formatnya pada bagian pembagian tugas).
7. Tambah file README di repositori Binotify Premium App (ReactJS), minimal berisi:
    a. Deskripsi singkat aplikasi
    b. Screenshot tampilan aplikasi (tidak perlu semua kasus, minimal 1 per halaman), dan
    c. Penjelasan mengenai pembagian tugas masing-masing anggota (lihat formatnya pada bagian pembagian tugas).
8. Buatlah ***script seeding*** untuk mengisi data awal agar memudahkan demonstrasi.

Pengumpulan Tugas

- Deadline tugas adalah pada hari **Jumat, 2 Desember 2022 pukul 15.00 WIB**.
- Step pengumpulan adalah sebagai berikut.
    a. Buat release dengan format **vX** pada gitlab informatika dan judul bebas namun 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, namun 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 deadline tugas.
- Alasan deadline jam 3 sore adalah pencegahan apabila terjadi error pada server Gitlab Informatika. Karena pada jam tersebut masih ada staff yang dapat menangani hal tersebut.
- (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

Silakan isi daftar kelompok kalian pada sheet yang terdapat di teams.

Kelompok maksimal terdiri dari 3 orang, diperbolehkan lintas kelas, diperbolehkan berbeda dengan tugas 1. Gunakan salah satu sistem binotify anggota kelompok kalian (disarankan yang memiliki *code* yang *clean* dan UI UX yang bagus agar lebih mudah melakukan pengubahan dan penggunaan).

Pembagian Tugas

Setiap anggota kelompok **diwajibkan** untuk mengerjakan bagian REST, SOAP, dan React, dengan harapan kalian bisa mencoba mempelajari semua bagian dan cara kerjanya

REST

Backend Binotify Premium Fungsi X : 13520xxx, 13520xxx

Backend Binotify Premium Fungsi Y : 13520xxx

(Lanjutkan ...)

SOAP

Request Subscription : 13520xxx, 13520xxx

Check Request : 13520xxx

(Lanjutkan...)

React

Frontend Binotify Premium Halaman X : 13520xxx, 13520xxx

Frontend Binotify Premium Halaman Y : 13520xxx

(Lanjutkan...)

PHP App

Frontend Binotify Halaman X : 13520xxx, 13520xxx

Backend Binotify Fungsi Y : 13520xxx

(Lanjutkan...)

Lain-lain

Files penting diletakkan di teams untuk kepentingan pengarsipan dokumen / artefak tugas.