Web Technologies

Server Side

Introduction

“Travel Blitar” adalah perusahaan rintisan lokal. Mereka ingin membuat website yang dapat membantu pengguna untuk mendapatkan itinerary transportasi umum untuk mencapai tempat tujuan mereka sesuai jadwal. Mereka akan memberikan koordinat untuk stasiun kereta api dan bus. Pengguna dapat mengatur tempat/stasiun awal dan akhir mereka, kemudian sistem akan menemukan rute tercepat antara dua stasiun yang ditentukan tergantung pada kendaraan/jalur yang berjalan. Rute dapat mencakup transfer antara beberapa kendaraan/jalur. Anda dapat menggunakan algoritme Anda sendiri untuk menyelesaikan masalah. Sistem harus membantu pengguna untuk memutuskan rute mana yang lebih cepat berdasarkan jadwal transportasi umum.

Sistem harus dipisahkan oleh arsitektur klien dan server. Pelanggan meminta arsitektur layanan web. Fase pengembangan harus dipisahkan menjadi dua fase. Tahap pertama adalah membuat layanan web backend. Anda dapat menggunakan dan meningkatkan desain yang diberikan dengan kerangka kerja sisi klien untuk berkomunikasi dengan layanan.

Glosarium:

Jadwal: adalah perpindahan dari satu stasiun/tempat ke stasiun/tempat berikutnya pada waktu tertentu.

Jalur: terdiri dari beberapa jadwal dan dijalankan oleh kendaraan (misalnya warna apa saja dari garis).

Route: adalah perjalanan penumpang dari keberangkatan/sumber ke tujuan/target.

Description of project and tasks

Deskripsi untuk fase pertama proyek tercantum di bawah ini. Tugas pertama adalah membuat a restful web service API yang dapat digunakan oleh front end untuk mengomunikasikan data

I. Web Service

“**Travel Blitar**” akan memberikan daftar web service yang perlu dibuat. Spesifikasi layanan web akan berisi URL path of web service, request method, requested parameter on URL, requested parameter on body request, response result and response status . Request and response pada layanan web hanya boleh berisi JSON

Ada tiga Roles/jenis pengguna: public, authenticated user and admin. Berikut adalah daftar web service yang diminta oleh perusahaan .

1. Authentication

a. Login (*v1/auth/login*)
Description: Untuk client mendapatkan login token via username and passwordRequest method: **POST**

Header: header authorization basic
Requested parameter:

- Body:
  - Username
  - password

Response result:

- If success,
  - header: response status: 200

- body:
  - token`: authorization token (valid sampai logout). Token akan di generate oleh system dari username yang login dengan md5 encryption method
  - Role (ADMIN / USER)
- If username/password not correct or empty,
  - header: response status: 401
  - body: message: invalid login

b. Logout (*v1/auth/logout?token={AUTHORIZATION_TOKEN}*)
Description: For server to invalid the user's token
Request method: **GET**

Header: header authorization basic
Response result:

- If success,
  - header: response status: 200
  - body:
    - message: logout success
- If unauthorized user access it, data:
  - Message: Unauthorized user
  - Response status: 401

2. Place

a. All Places (*v1/place?token={AUTHORIZATION_TOKEN}*)
Description: For client to list all places in the database

Request method: **GET**

Header: header authorization basic

Response result:

body:

- All data on array; terdiri dari id, name, latitude, longitude, image_path, description.
- Response status: 200
- If unauthorized user access it, data:
  - Message: Unauthorized user
  - Response status: 401

b. Find Place (*v1/place/{ID}?token={AUTHORIZATION_TOKEN}*))
Description: For client untuk fetch satu place object melalui place ID.
Request method: **GET**
Header: header authorization basic
Response result:

- body:
  - object; property consists of id, name, latitude, longitude, image_path, description.
  - Response status: 200

c. Create place (*v1/place?token={AUTHORIZATION_TOKEN}*), only admin can access thisAPI
Description: For client to create a new place object. Image file from client should be uploaded to server. You can use form data to upload an image.

Request method: **POST**

Header: header authorization basic
Request parameter:

- Body:

- name
- latitude
- longitude
- image
- [description]

Response result:

- If success, body:
  - Message: create success
  - Response status: 200
- If failed, body:
  - Message: Data cannot be processed
  - Response status: 422
- If unauthorized user access it, body:
  - Message: Unauthorized user
  - Response status: 401

d. Delete place (*v1/place/{ID}?token={AUTHORIZATION_TOKEN}*), only admin can accessthis API

Description: A request to delete a place object via given place ID.

Request method: **DELETE**

Header: header authorization basic

Response result:

- If success, body:
  - Message: delete success
  - Response status: 200
- If failed, body:
  - Message: Data cannot be deleted
  - Response status: 400
- If unauthorized user access it, data:
  - Message: Unauthorized user
  - Response status: 401

e. Update place (*v1/place/{ID}?token={AUTHORIZATION_TOKEN}*), only admin canaccess this API

Description: For client to update an existing place object via given place ID. If an imagefile is provided, it should be uploaded to server.

Request method: **POST**

Header: header authorization basic

Request parameter:

- Body:
  - [name]
  - [latitude]
  - [longitude]
  - [image]
  - [description]

Response result:

- If success, body:
  - Message: update success
  - Response status: 200

- If failed, body:
  - Message: Data cannot be updated
  - Response status: 400
- If unauthorized user access it, body:
  - Message: Unauthorized user
  - Response status: 401

3. Schedule

a. Create schedule (*v1/schedule?token={AUTHORIZATION_TOKEN}*), only admin canaccess this API

Description: For client to create a schedule in database. Schedule menjelaskan **when and where** a bus/train departs (berangkat) dari stasiun/terminal dan arrive (datang) di stasiun/terminal berikutnya.

Request method: **POST**

Header: header authorization basic

Request parameter:

- Body:
  - Object: consisting of type (bus or train), line, from_place_id, to_place_id, departure_time, arrival_time, distance (km), speed (hour)

Response result:

- If success,
  - header: response status: 200
  - body: message: create success
- If failed,
  - header: response status: 422
  - body: message: Data cannot be processed
- If unauthorized user access it,
  - header: response status: 401
  - body: message: Unauthorized user

b. All Schedule (*v1/schedule?token={AUTHORIZATION_TOKEN}*)

Description: For client to show schedule in database. Schedule menjelaskan **when and where** a bus/train departs (berangkat) dari stasiun/terminal dan arrive (datang) di stasiun/terminal berikutnya.

Request method: **GET**

Header: header authorization basic

Response result:

body:

- All data on array; consisting of type (bus or train), line , from_place_id, to_place_id,departure_time, arrival_time, distance, speed.
- Response status: 200
- If unauthorized user access it, data:
  - Message: Unauthorized user
  - Response status: 401

c. Delete schedule (*v1/schedule/{ID}?token={AUTHORIZATION_TOKEN}*), only admin canaccess this API
Description: A request to delete an existing schedule via given schedule ID.
Request method: **DELETE**

Header: header authorization basic
Response result:

- If success,
  - header: response status: 200
  - body: message: delete success
- If unauthorized user access it,
  - header: response status: 401
  - body: message: Unauthorized user

4. Route

a. Route Search (*v1/route/search/{FROM_PLACE_ID}/{TO_PLACE_ID}?token={AUTHORIZATION_TOKEN}*)
Description: A request to fetch multiple route suggestions to depart from a given stop (departure/source) and arrive at another stop (destination/target). By default, the search uses current server time. It also allows an optional departure time to override the default server time.

The search should search the routes that depart from the given place at specific time and arrive the destination place, sorting by the earliest arrival time and **limited to 5 routes result**.

The route allows transfer to different bus/train at the same station. All transfer happens at the same stop. There is no walk and no minimum transfer time required.

Request method: **GET**

Header: header authorization basic
Response result:

- If success, data:
  - Array of routes. Each route contains :
    - Number of history selection of this route.
    - Array of schedules:
      - id
      - type
      - line
      - departure_time
      - arrival_time
      - travel_time
      - from_place; consist of id, name, longitude, latitude, x, y, description, image_path
      - to_place; consist of id, name, longitude, latitude, x, y, description, image_path
  - Response status: 200

- If failed, data:
    - Message: Unauthorized user
    - Response status: 401

b. Store Route Selection History (*v1/route/selection?token=[AUTHORIZATION_TOKEN]*)
Description: For client to save a user selected route into the system.
Request method: **POST**

Header: header authorization basic
Request parameter:

- Body:
    - from_place_id
    - to_place_id
    - schedule_id, array of schedule_id for the route

Response result:
- If success, body:
    - Message: create success
    - Response status: 200
- If failed, body:
    - Message: Data cannot be processed
    - Response status: 422

Komponen website yang telah disediakan adalah:

1. Pencarian Route

Functionalities:

a. Memilih dari suatu tempat ke tempat lain
- Ambil daftar tempat dari layanan. Diurutkan berdasarkan abjad secara default. Tempat yang digunakan oleh pengguna ini (jika login) akan berada di urutan teratas (berdasarkan frekuensi)

b. Searching routes
- Cari rute dengan menggunakan input dari (sumber) dan ke (target)

2. Daftar Routes

Setelah pengguna mencari rute, maka komponen ini akan menampilkan:

Hasil pencarian di sebelah kiri, jadwal rute ini akan muncul

Sama seperti yang dijelaskan oleh spesifikasi API: Pencarian harus mencari rute yang berangkat dari tempat tertentu pada waktu tertentu dan tiba di tempat tujuan, diurutkan berdasarkan waktu kedatangan paling awal dan hasil terbatas pada 5 rute.

Rute memungkinkan transfer ke bus/kereta yang berbeda di stasiun yang sama. Transfer terjadi di halte yang sama. Tidak ada jalan kaki dan tidak ada waktu transfer minimum yang diperlukan

Daftar jadwal kereta api atau bus dari hasil, terdiri dari:

- Penomoran.
- Jadwal Waktu (Waktu keberangkatan di tempat, waktu kedatangan di tempat).

- Total waktu perjalanan
- Jumlah transfer/perubahan jalur
- Jumlah pilihan pada rute ini oleh semua pengguna (lebih lanjut tentang itu nanti di 4. Riwayat pencarian).
- Ini adalah contoh skenario untuk hasil pencarian, dengan asumsi dari A ke B yang berangkat s pada pukul 13:00:
    1. A ke B => waktu keberangkatan -> 13:00:00 waktu kedatangan -> 13:14:00, Bus Line 3, 14 menit, 0 transfer.
    2. A ke B => waktu keberangkatan -> 13:02:00 waktu kedatangan -> 13:22:00, Bus Jalur 2, Kereta Jalur 4, 22 menit, 1 transfer
    3. A ke B => waktu keberangkatan -> 13:01:00 waktu kedatangan -> 13:35:00, Bus Line 2, Bus Line 1, Bus Line 5, 36 menit, 2 transfer

3. Route(s) selection history

Sistem menyimpan semua rute yang dipilih semua pengguna

a. Sistem menyimpan semua pilihan pengguna

4. User Authentication

Functionalities:

a. Login dan logout harus terjadi pada halaman yang sama tanpa redirect
b. Login
    - Tampilkan modal login, setelah pengguna mengklik link login.
    - Pada dialog login akan ada input username dan password.
    - Setelah pengguna login, link login akan diubah menjadi link logout dan username saat ini akan ditampilkan di samping link logout.
    - Akan ada peran untuk dua jenis pengguna yang diautentikasi: jika pengguna adalah admin, menu admin akan ditampilkan.
    - Nama pengguna dimasukkan dan token diterima, token akan disimpan di klien untuk permintaan lebih lanjut, juga setelah refresh halaman
c. Logout
    - Tampilan di reset: login link is terlihat, nama pengguna dan fungsi terkait menghilang

Notes

- Peserta harus mengimplementasikan server side yang disediakan.
- Tampilkan pesan kesalahan/umpan balik berdasarkan respons dari API.
- Tabel database yang ditentukan perlu diimplementasikan. Lebih banyak tabel dapat ditambahkan jika diperlukan. Berikan layar SQL-dump dan ERD akhir seperti yang ditentukan di bawah ini.
  Semua API harus memenuhi semua persyaratan seperti yang tercantum dalam deskripsi. Semua awalan, RESTful-URL dan Metode HTTP dari tautan API yang diberikan harus diterapkan dengan benar dan tidak diubah. Jika diperlukan, Anda dapat menambahkan API lain, selain semua API yang telah disebutkan dalam dokumen ini.

Buat pengguna berikut untuk login ke sistem:

- Admin dengan username: admin dan password: adminpass,

- User1 dengan username: user1 dan password: user1pass,
- User2 dengan username: user2 dan password: user2pass

Instruksi

Kerjakanlah seluruh tugas yang diberikan sesuai ketentuan masing-masing.

Simpanlah semua hasil pekerjaan anda dalam sebuah folder sesuai dengan nomor meja anda (XX_ServerSide) dimana XX adalah nomor meja anda. Untuk setiap part soal yang anda kerjakan harus disimpan di dalam sub folder pekerjaan anda dengan diberi nama sesuai part maisng-masing.