Backend Assessment III

PT. Kledo Berhati Nyaman

Syarat Teknis

Gunakan Laravel 8 (PHP 7.4) sebagai framework, dengan syarat sebagai berikut:

- Webserver menggunakan `artisan serve`.
- Validasi menggunakan Form Request Validation.
- Tidak ada algoritma alur proses di Controller.
- Transaksi basis data menggunakan Eloquent (MySQL/MariaDB).
- Dokumentasi REST-API menggunakan Swagger (l5-swagger).
- Testing menggunakan PHPUnit.
- Sertakan README sebagai petunjuk penggunaan.

Kriteria Penilaian

- Aplikasi berjalan sesuai spesifikasi, seluruh syarat di atas wajib dipenuhi. README akan sangat berpengaruh dalam proses uji coba yang kami lakukan.
- Kualitas kode.
- Kecepatan. Maksimal waktu pengerjaan adalah 24 jam, terhitung mulai dokumen ini dikirim.

Berikut adalah penjelasan mengenai apa saja yang wajib Anda selesaikan dalam tugas ini.

Kebutuhan

Buat aplikasi dengan konsep REST-API pada sisi backend untuk memenuhi kebutuhan sistem perhitungan lembur (atau overtime)

Ada 2 metode perhitungan lembur yang digunakan.

1. Salary / 173 (uang lembur per jam nilainya tergantung dari gaji pegawai)
2. Fixed (uang lembur per jam nilainya selalu sama)

Jika yang digunakan adalah metode pertama, maka semua upah lembur dari pegawai yang ada akan dihitung menggunakan metode tersebut. Begitu juga jika yang dipilih adalah metode kedua.

Tabel

references

- id
- code
- name

- expression

settings

- key
- value
- expression

employees

- id
- name
- salary

overtimes

- id
- employee_id
- date
- time_started
- time_ended

Seeder

references

id	code	name	expression
1	overtime_method	Salary / 173	(salary / 173) * overtime_duration_total
2	overtime_method	Fixed	10000 * overtime_duration_total

settings

key	value	expression
overtime_method	1	(salary / 173) * overtime_duration_total

Endpoint

Setting

PATCH /settings

Mengubah data `settings`.

Request
- key
- value

Rule
- key
  - Hanya bisa diisi `overtime_method`.
- value
  - Hanya bisa diisi oleh nilai dari `references`.`id` dengan kriteria `code` = `overtime_method`.
- expression
  - Otomatis didapat berdasarkan `value` yang diterima, diambil dari `references`.`expression`.

Employee

POST /employees

Membuat data `employees`.

Request
- name
- salary

Rule
- name
  - String
  - Minimal 2 karakter
  - Harus unik
- salary
  - Integer
  - Minimal 2 juta
  - Maksimal 10 juta

Overtime

POST /overtimes

Membuat data `overtimes`.

Request
- employee_id
- date
- time_started
- time_ended

Rule
- employee_id
  - Integer
  - Sesuai dengan yang ada di `employees`.`id`
- date
  - Date
  - Tidak boleh ada `date` yang sama pada `employee_id` tersebut
- time_started
  - Format HH:mm
  - Tidak boleh lebih dari `time_ended`
- time_ended
  - Format HH:mm
  - Tidak boleh kurang dari `time_started`

Overtime Pay

GET /overtime-pays/calculate

Menampilkan hasil perhitungan dari `overtimes` yang ada pada setiap `employees`, berdasarkan bulan yang ditentukan, tanpa format pagination.

Request
- month

Rule
- month
  - Format YYYY-MM

Response
- id
- name
- salary
- overtimes[]
  - id
  - date

    - time_started
    - time_ended
    - overtime_duration *perhitungan durasi sesuai penjelasan di awal
- overtime_duration_total *akumulasi `overtime_duration`
- amount *perhitungan upah yang diterima sesuai pengaturan `overtime_method` di `settings`

Sesuai namanya, gunakan `settings`.`expression` sebagai rumus untuk menghasilkan upah lembur, bukan manual di kode.

Testing

Buat testing yang bersifat fungsional menggunakan PHPUnit untuk memastikan apa yang telah Anda kerjakan sesuai dengan aturan yang diberlakukan. Testing mencakup semua skenario di atas tanpa terkecuali. Pastikan testing dapat berjalan dengan satu perintah.

Checklist

Pastikan kriteria-kriteria di bawah terpenuhi

Topik	**Requirement**	**Nilai Tambah**
Validasi endpoint	Gunakan Form Request Validation	Menggunakan file class request terpisah
Controller	Alur logika kode diletakkan diluar file class Controller. Dari file class Controller hanya memanggil method dari class lain.	Menggunakan Repository Pattern
Dokumentasi	Gunakan dokumentasi swagger, di controller dan di model. Pastikan Swagger UI bisa diakses, dan bisa digunakan untuk mencoba endpoint.	
Testing	Setiap endpoint di test untuk memastikan bisa berjalan dengan baik.	Tidak hanya test response code saja, tapi berbagai kondisi di test. Mulai dari test validasi-nya, serta test alur didalamnya. Semakin lengkap semakin bagus.
Readme	Pastikan tim penilai bisa	Semakin jelas readme nya

	menginstall script Anda dengan mudah. Bisa copy paste perintah yang Anda cantumkan di readme, dan bisa berjalan normal.	semakin bagus.
Upload Hasil	Pastikan tidak mempublikasikan hasil kerja di Github atau GIT repository sejenisnya. Upload hasil kerja Anda hanya pada Google Form yang disediakan. Pastikan juga tidak menyertakan file yang ada pada daftar .gitignore untuk mereduksi ukuran file yang akan diupload.	