\

Laporan Modul Praktik

"**Membuat REST API Pencari Kerja menggunakan Node.Js**"

(Laporan ini dibuat untuk memenuhi tugas mata kuliah Pemograman Berbasis Platform)

Dosen Pengampu :

Muhammad Ikhsan Thohir, M.Kom

Kelompok 1 :

1. **Mughis Fadhil A.Ridwan (20230040217)**
2. **Rendi Ruswandi (20230040270)**
3. **Wardatul Jannah (20230040120)**

PROGRAM STUDI TEKNIK INFORMATIKA

UNIVERSITAS NUSA PUTRA

2025

\

KATA PENGANTAR

Puji syukur kita panjatkan kehadirat Allah SWT, atas segala rahmat dan hidayah nya sehingga kami dapat menyelesaikan Tugas Laporan yang berjudul “Membuat REST API Pencari Kerja menggunakan Node.Js” pada mata kuliah Pemograman Berbasis Platform.

Shalawat serta salam tak lupa kami haturkan kepada junjungan kita Nabi Besar Muhammad SAW, keluarga, sahabat, serta para pengikut-pengikut beliau sampai akhir zaman.

Tujuan dalam pembuatan makalah ini adalah untuk memenuhi salah satu tugas mata kuliah dan juga menambah wawasan para pembaca sekalian.

Kami dapat menyadari bahwa masih banyak kekurangan dalam penyusunan makalah ini. oleh karena itu, kami menghargai akan saran dan kritik untuk membangun makalah ini lebih baik lagi. Demikian yang dapat kami sampaikan, semoga melalui makalah ini dapat memberikan manfaat bagi kita semua.

Sukabumi, 29 Januari 2025

Rendi Ruswandi

\

DAFTAR ISI

KATA PENGANTAR **2**

DAFTAR ISI **3**

BAB I PENDAHULUAN **4**

1.1 Latar Belakang 4

1.2 Perangkat Lunak yang digunakan 4

1.3 Tujuan dan Manfaat 4

1.3.1 Tujuan 4

1.3.2 Manfaat 5

BAB II ISI LAPORAN **6**

2.1 Persiapan dan Persyaratan Awal 6

2.2 Implementasi REST API 7

2.3 Membuat Rute dan Operasi CRUID 8

2.4 Pengujian API 10

BAB III PENUTUP **12**

3.1 Kesimpulan 12

\

BAB I PENDAHULUAN

1.1 Latar Belakang

Laporan ini bertujuan untuk menjelaskan langkah-langkah praktis dalam membangun REST API sebagai solusi untuk pengelolaan data konsultasi kesehatan. REST API yang dirancang berfungsi sebagai antarmuka backend yang dapat diakses oleh aplikasi atau sistem lain untuk menyediakan layanan terkait konsultasi kesehatan. Dengan menggunakan teknologi modern seperti Node.js, MySQL, dan JsonWebToken (JWT), API ini menyediakan berbagai fitur, termasuk pencari pekerjaan, pendaftaran pengguna, manajemen konsultasi karir, informasi lowongan kerja, serta pengelolaan pembayaran yang membantu pencari kerja menemukan peluang yang sesuai kriteria nya.

Perancangan API ini juga menerapkan prinsip-prinsip dasar pengembangan perangkat lunak, seperti modularitas, keamanan data, dan efisiensi. Secara khusus, validasi data melalui JWT memastikan bahwa hanya pengguna yang berwenang dapat mengakses data yang sensitif. Laporan ini diharapkan memberikan panduan teknis bagi pengembang dan akademisi yang tertarik dengan teknologi REST API.

1.2 Perangkat Lunak yang digunakan

Perangkat lunak yang digunakan mencakup :

- Node.js sebagai lingkungan runtime JavaScript
- MySQL sebagai basis data relasional
- Postman untuk pengujian API
- Visual Studio Code sebagai text editor/IDE

1.3 Tujuan dan Manfaat

1.3.1 Tujuan

Bertujuan mempermudah pencari kerja dalam menemukan lowongan yang sesuai dengan keterampilan dan pengalaman mereka dan emberikan solusi yang aman dan efisien dalam manajemen data dengan menggunakan standar keamanan seperti autentikasi JWT. Dan meningkatkan aksesibilitas layanan bimbingan karir bagi pengguna memungkinkan integrasi yang mudah dengan aplikasi atau layanan lain, lalu memfasilitasi pengujian dan pemeliharaan layanan melalui arsitektur RESTful yang modular.

\

1.3.2 Manfaat

- Kemudahan Akses: Data lowongan kerja dapat diakses dari berbagai platform yang mendukung HTTP requests.
- Efisiensi Operasional: Memungkinkan pengolahan data yang lebih cepat dan otomatisasi dalam pencarian kerja.
- Keamanan Data: Dengan implementasi JWT, hanya pengguna yang memiliki otorisasi yang dapat mengakses data sensitif.
- Kemudahan Pengembangan: Struktur API yang modular dan terorganisir memudahkan pengembang dalam memperbarui dan menambahkan fitur baru.
- Skalabilitas: Sistem dapat dikembangkan lebih lanjut untuk mendukung fitur tambahan seperti pencocokan otomatis antara kandidat dan lowongan.

\

BAB II ISI LAPORAN

2.1 Persiapan dan Persyaratan Awal

Untuk memulai pengembangan REST API ini, berikut adalah persyaratan perangkat lunak dan perangkat keras yang harus dipenuhi:

1. Software dan Tools yang Digunakan :

   a. Node.js : Instalasi ntuk membangun dan menjalankan server backend.

```
{
  "name": "api-konsultasi-kesehatan-",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

   b. MySQL Server: Untuk mengelola database yang berisi data lowongan kerja dan pencari kerja.

```
-- Tabel pengguna
▷ Run | Select | Ask Copilot
CREATE TABLE pengguna (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nama VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL,
    kata_sandi VARCHAR(255) NOT NULL,
    peran VARCHAR(50)
);

-- Tabel konsultasi
▷ Run | Select | Ask Copilot
CREATE TABLE konsultasi (
    id INT AUTO_INCREMENT PRIMARY KEY,
    id_pengguna INT NOT NULL,
    id_konsultan INT NOT NULL,
    topik VARCHAR(255) NOT NULL,
```

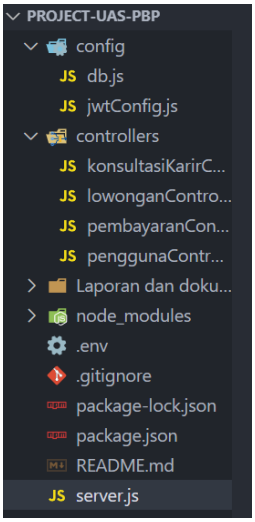
```
MariaDB [konsultasi_karir]> SHOW TABLES
    -> ;
+---------------------------+
| Tables_in_konsultasi_karir |
+---------------------------+
| konsultasi                |
| lowongan                  |
| pembayaran                |
| pengguna                  |
+---------------------------+
4 rows in set (0.026 sec)

MariaDB [konsultasi_karir]> SELECT * FROM pengguna;
+----+-------+---------------------+--------------------------------------------------------------+-------+
| id | nama  | email               | kata_sandi                                                   | peran |
+----+-------+---------------------+--------------------------------------------------------------+-------+
|  1 | Ucup  | ucupkece@gmail.com  | $2b$10$irpcKJZDxX14KeJmgbfAO.vJxzA1dboo..xfF1MWBwfLQBklnC0ae |       |
|  2 | Rendi | rendi@gmail.com     | $2b$10$QLSTTsVFcZ1BkseF8y3AAeXcIjP0LBpgZWmWah3XBGq3UkBUbTLDC |       |
|  4 | Emul  | mulgans@gmail.com   | $2b$10$3FocvH2fRt7nk/bH7o//ReJdEPA.7riTo8jmJzLHibddxqPhC70AO |       |
|  5 | Ken   | kentang@gmail.com   | $2b$10$YKHto8OGXFt.haL7r4LLn.MKVD9nOfVRjyiELZtqK1su473r/hFGm |       |
|  6 | Bayu  | bayu@gmail.com      | $2b$10$k.jmaSY5vFLixjwxNNVqlunvSsFffVBglg8bNLpsWiKD/YYzSiSfS |       |
+----+-------+---------------------+--------------------------------------------------------------+-------+
5 rows in set (0.017 sec)

MariaDB [konsultasi_karir]> SELECT * FROM konsultasi;
+----+-------------+--------------+-------------------------------+------------------------------------------
| id | id_pengguna | id_konsultan | topik                         | deskripsi
+----+-------------+--------------+-------------------------------+------------------------------------------
|  1 |           1 |            2 | Karir di Teknologi            | Diskusi tentang peluang karir di bidang te
|  2 |           1 |            2 | Karir di Design dan Properti  | Diskusi tentang peluang karir di bidang De
+----+-------------+--------------+-------------------------------+------------------------------------------
2 rows in set (0.015 sec)
```

   c. Postman: Untuk pengujian endpoint API.

   d. Text Editor/IDE: Visual Studio Code disarankan karena mendukung pengembangan berbasis JavaScript dengan berbagai ekstensi yang membantu.

\

2. Struktur Direktori Proyek :



a. Config/db.js: File konfigurasi untuk koneksi database.

b. server.js: File utama untuk menjalankan server.

c. controllers/: Direktori untuk menyimpan logika bisnis CRUD.

d. routes/: Direktori untuk mendefinisikan endpoint API.

e. models/: Berisi definisi model database.

2.2 Implementasi REST API

REST API ini dirancang untuk menyediakan akses ke data lowongan kerja dan bimbingan karir. langkah-langkah implementasi yang telah dilakukan :

Konfigurasi Koneksi Database

```
config > JS db.js > ...
const mysql = require('mysql2');  782.8k (gzipped: 345.6k)

const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'konsultasi_karir' // Mengubah nama database agar sesuai dengan tema baru
});

connection.connect((err) => {
  if (err) {
    console.error('Kesalahan koneksi ke database: ', err);
    return;
  }
  console.log('Koneksi ke database jasa pencari lowongan pekerjaan dan konsultasi bimbingan pra kerja berhasil!');
});

module.exports = connection;
```

- File Config/db.js digunakan untuk mengatur koneksi ke database MySQL.
- Modul mysql2 digunakan untuk membuat koneksi dan mengelola query.

\

- Contoh kode konfigurasi :

Setup Server

```
JS server.js > ...
const express = require('express');
const bodyParser = require('body-parser');  487.5k (gzipped: 212.1k)
const penggunaController = require('./controllers/penggunaController');
const lowonganController = require('./controllers/lowonganController');
const konsultasiKarirController = require('./controllers/konsultasiKarirController');
const pembayaranController = require('./controllers/pembayaranController');
require("dotenv").config();  6.3k (gzipped: 2.8k)

const app = express();
const PORT = 5000;

app.use(bodyParser.json());

app.use('/api', penggunaController);
app.use('/api', lowonganController);
app.use('/api', konsultasiKarirController);
app.use('/api', pembayaranController);

app.listen(PORT, () => {
  console.log(`Server berjalan di http://localhost:${PORT} untuk layanan pencarian kerja dan bimbingan karir`);
});

//Kelompok 1
```

- File server.js digunakan untuk memulai server HTTP pada localhost.
- Server dibuat menggunakan modul express.
- Contoh kode :

2.3 Membuat Rute dan Operasi CRUID

Setiap entitas dalam database memiliki kontroler dan rute tersendiri mendukung operasi CRUD :

1. penggunaController.js :

```
controllers > JS penggunaController.js > ⓕ router.get('/pengguna') callback
const express = require('express');
const router = express.Router();
const db = require('../config/db'); // Gunakan db dari config.js
const bcrypt = require('bcrypt');  Calculating...
const jwt = require('jsonwebtoken');  53.2k (gzipped: 15.9k)
require('dotenv').config(); // Pastikan menggunakan environment variable  6.3k (gzipped: 2.8k)

const SECRET_KEY = process.env.SECRET_KEY; // Gunakan SECRET_KEY dari environment variable

// Registrasi pengguna untuk platform pencari lowongan kerja dan konsultasi pra-kerja
router.post('/register', async (req, res) => {
    const { nama, email, kata_sandi, peran } = req.body;

    if (!nama || !email || !kata_sandi) {
        return res.status(400).json({ message: 'Semua field harus diisi!' });
    }

    try {
        const [existingUser] = await db.promise().query('SELECT * FROM pengguna WHERE email = ?', [email]);
        if (existingUser.length > 0) {
            return res.status(400).json({ message: 'Email sudah digunakan!' });
        }

        const hashedPassword = await bcrypt.hash(kata_sandi, 10);

        await db.promise().query('INSERT INTO pengguna (nama, email, kata_sandi, peran) VALUES (?, ?, ?, ?)', [
            nama,
            email,
            hashedPassword,
            peran || 'pelamar'
        ]);
```

Full code (https://github.com/RendiRuswandi/Project-UAS-PBP)

- Menangani pendaftaran pengguna baru, autentikasi login, dan validasi JWT.
- Endpoint utama:
  - POST /api/register : Untuk mendaftarkan pengguna baru.

\

- POST /api/login : Untuk login dan menghasilkan token JWT.
- GET /api/pengguna : Untuk menampilkan data semua pengguna.

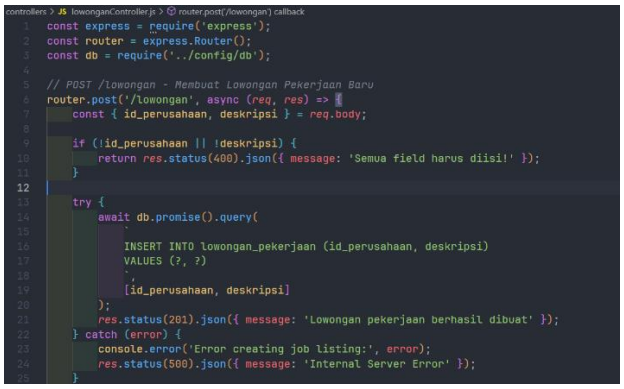
2. konsultasiKarirController.js:

```
const express = require('express');
const router = express.Router();
const db = require('../config/db');

// POST /konsultasiKarir - Membuat janji konsultasi bimbingan pra kerja baru
router.post('/karir', async (req, res) => {
    const { id_pelamar, id_mentor, jadwal_konsultasi } = req.body;

    if (!id_pelamar || !id_mentor || !jadwal_konsultasi) {
        return res.status(400).json({ message: 'Semua field harus diisi!' });
    }

    try {
        await db.promise().query(
            `
            INSERT INTO janji_konsultasi (id_pelamar, id_mentor, jadwal_konsultasi)
            VALUES (?, ?, ?)
            `,
            [id_pelamar, id_mentor, jadwal_konsultasi]
        );
        res.status(201).json({ message: 'Janji konsultasi bimbingan pra kerja berhasil dibuat' });
    } catch (error) {
        console.error('Error creating appointment:', error);
        res.status(500).json({ message: 'Internal Server Error' });
    }
});

// GET /konsultasiKarir - Mengambil daftar janji konsultasi untuk pelamar tertentu
router.get('/karir', async (req, res) => {
    const { id_pelamar } = req.query;
```

Full code (https://github.com/RendiRuswandi/Project-UAS-PBP)

- Mengelola sesi bimbingan karir, termasuk pendaftaran dan penjadwalan.
- Endpoint utama:
  - POST /api/konsultasi : Untuk mendaftarkan jadwal baru.
  - PUT /api/konsultasi/:id : Untuk memperbarui jadwal berdasarkan ID

3. lowonganController.js:



```
const express = require('express');
const router = express.Router();
const db = require('../config/db');

// POST /lowongan - Membuat Lowongan Pekerjaan Baru
router.post('/lowongan', async (req, res) => {
    const { id_perusahaan, deskripsi } = req.body;

    if (!id_perusahaan || !deskripsi) {
        return res.status(400).json({ message: 'Semua field harus diisi!' });
    }

    try {
        await db.promise().query(
            `
            INSERT INTO lowongan_pekerjaan (id_perusahaan, deskripsi)
            VALUES (?, ?)
            `,
            [id_perusahaan, deskripsi]
        );
        res.status(201).json({ message: 'Lowongan pekerjaan berhasil dibuat' });
    } catch (error) {
        console.error('Error creating job listing:', error);
        res.status(500).json({ message: 'Internal Server Error' });
    }
```

Full code (https://github.com/RendiRuswandi/Project-UAS-PBP)

- Mengelola data lowongan kerja, termasuk pendaftaran, pembaruan, dan penghapusan lowongan.

\

- Endpoint utama :
  - GET /api/rekam-medis: Untuk menampilkan semua rekam medis.
  - DELETE /api/rekam-medis/:id: Untuk menghapus data berdasarkan ID.

4. pembayaranController.js :

```
controllers > pembayaranController.js > router.post('/pembayaran') callback
const express = require('express');
const router = express.Router();
const db = require('../config/db');

// POST /pembayaran - Membuat pembayaran untuk layanan bimbingan pra kerja
router.post('/pembayaran', async (req, res) => {
    const { id_konsultasi_karir, jumlah, status } = req.body;

    if (!id_konsultasi_karir || !jumlah) {
        return res.status(400).json({ message: 'Field id_konsultasi_karir dan jumlah harus diisi!' });
    }

    try {
        await db.promise().query(
            `
            INSERT INTO pembayaran (id_konsultasi_karir, jumlah, status)
            VALUES (?, ?, ?)
            `,
            [id_konsultasi_karir, jumlah, status || 'tertunda']
        );
        res.status(201).json({ message: 'Pembayaran untuk konsultasi bimbingan pra kerja berhasil dibuat' });
    } catch (error) {
        console.error('Error creating pembayaran:', error);
        res.status(500).json({ message: 'Internal Server Error' });
    }
});

// GET /pembayaran - Mengambil semua pembayaran layanan bimbingan pra kerja
router.get('/pembayaran', async (req, res) => {
    try {
        const [results] = await db.promise().query(
```

Full code (https://github.com/RendiRuswandi/Project-UAS-PBP)

- Mengelola data pembayaran, termasuk pembuatan catatan pembayaran dan pembaruan status pembayaran.
- Endpoint utama:
  - POST /api/pembayaran: Untuk mencatat pembayaran baru.
  - GET /api/pembayaran/:id: Untukrincian pembayaran berdasarkan ID.

2.4 Pengujian API

Pengujian digunakan menggunakan Postman

1. Pengujian pada penggunaController.js dengan endpoint :
   - GET /api/pengguna : Menampilkan semua pengguna yang telah login
   - POST /api/register : Mendaftarkan pengguna baru
   - POST/api/login : Menambahkan pengguna baru dan menampilkan kode JWT
   - GET/api/pengguna/id : Menampilkan pengguna berdasarkan id dan memasukan JWT
   - PUT/api/pengguna/id : Merubah data pengguna
   - DELETE/api/pengguna/id: Menghapus pengguna

\

2. Pengujian pada lowonganController.js dengan endpoint :
    - GET /api/konsultasi : Menampilkan semua lowongan
    - GET/api/konsultasi/id : Menampilkan hasil lowangn berdasarkan id
    - POST /api/konsultasi : Membuat data lowngan
    - PUT/api/konsultasi/id : Merubah data hasil lowongan
    - DELETE/api/konsultasi/id : Menghapus data hasil lowongan

3. Pengujian dengan konsultasiKarirController.js dengan endpoint :
    - GET /api/konsultasi : Menampilkan semua hasil konsultasi
    - GET/api/konsultasi/id : Menampilkan hasil konsultasi berdasarkan id
    - POST /api/konsultasi : Membuat data untuk hasil konsultasi
    - PUT/api/konsultasi/id : Merubah data hasil konsultasi
    - DELETE/api/konsultasi/id : Menghapus data hasil konsultasi

4. Pengujian pada pembayaranController.js dengan endpoint :
    - GET /api/konsultasi : Menampilkan semua hasil konsultasi
    - GET/api/konsultasi/id : Menampilkan hasil konsultasi berdasarkan id
    - POST /api/konsultasi : Membuat data untuk hasil konsultasi
    - PUT/api/konsultasi/id : Merubah data hasil konsultasi
    - DELETE/api/konsultasi/id : Menghapus data hasil konsultasi

\

BAB III PENUTUP

3.1 Kesimpulan

REST API yang dirancang memberikan solusi menyeluruh untuk pengelolaan data lowongan kerja dan bimbingan karir. Dengan menggunakan teknologi modern seperti Node.js dan MySQL, API ini mendukung integrasi dengan aplikasi lain serta menyediakan fitur utama dalam pengelolaan pencari kerja dan bimbingan karir.

Proyek ini berhasil menunjukkan bagaimana REST API dapat digunakan untuk mengelola data dengan efisien dan aman. Dengan implementasi CRUD dan validasi JWT, API ini mendukung fitur seperti pendaftaran pengguna, manajemen jadwal konsultasi, data lowongan kerja, dan pengelolaan pembayaran. API ini dapat dikembangkan lebih lanjut untuk memenuhi kebutuhan skala besar, seperti integrasi yang lebih kompleks dan analitik data.

\