



PBP GROUP PROJECT

JOB SEEKERS AND CAREER GUIDANCE

Membangun REST API
dengan Node.js

Kelompok 1 :

Mughis Fadhil A.Ridwan (20230040217)
Rendi Ruswandi (20230040270)
Wardatul Jannah (20230040120)

REST API

Sistem Pencari Kerja dan Bimbingan Karir

Fitur Utama :

- Manajemen Pengguna
- Konsultasi Karir
- Lowongan Kerja
- Data Pembayaran

Teknologi yg dipakai :

- Node.js
- MySQL
- JSON Web Token (JWT)
- VS Code

TOOLS SOFTWARE

Perangkat Lunak yang digunakan :

- Node.js

```
{
  "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": ""
}
```

- MySQL Server (Database)

```
-- 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,
  id_konsultan INT,
  topik VARCHAR(255),
  deskripsi TEXT,
  diskusi TEXT
);

-- Tabel lowongan
▷ Run | Select | Ask Copilot
CREATE TABLE lowongan (
  id INT AUTO_INCREMENT PRIMARY KEY,
  posisi VARCHAR(50),
  perusahaan VARCHAR(100),
  lokasi VARCHAR(50),
  deskripsi TEXT,
  dibuat_pada DATE
);

-- Tabel pembayaran
▷ Run | Select | Ask Copilot
CREATE TABLE pembayaran (
  id INT AUTO_INCREMENT PRIMARY KEY,
  id_pengguna INT,
  id_konsultasi INT,
  jumlah DECIMAL(10, 2),
  metode_pembayaran VARCHAR(50),
  status VARCHAR(50),
  tanggal_pembayaran DATE
);
```

TOOLS SOFTWARE

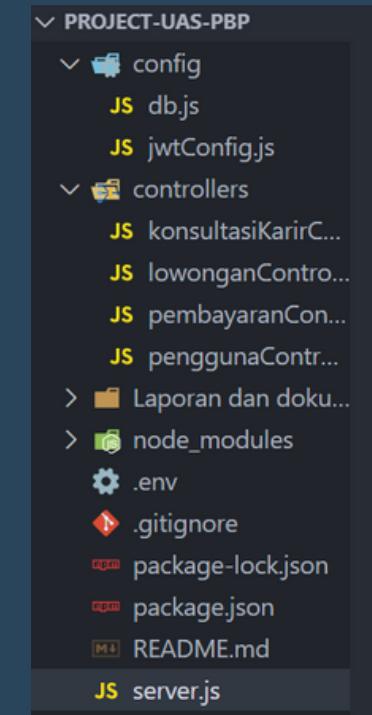
Perangkat Lunak yang digunakan :

- Postman (uji API)



POSTMAN

- VS Code (Struktur Proyek)



IMPLEMENTASI REST API

1. Konfigurasi Koneksi Database

Membuat kode program pada Config/db.js

```
config > JS db.js > ...
1  const mysql = require('mysql2'); 782.8k (gzipped: 345.6k)
2
3  const connection = mysql.createConnection({
4    host: 'localhost',
5    user: 'root',
6    password: '',
7    database: 'konsultasi_karir' // Mengubah nama database agar sesuai dengan tema baru
8  );
9
10 connection.connect((err) => {
11   if (err) {
12     console.error('Kesalahan koneksi ke database: ', err);
13     return;
14   }
15   console.log('Koneksi ke database jasa pencari lowongan pekerjaan dan konsultasi bimbingan pra kerja berhasil!');
16 });
17
18 module.exports = connection;
```

IMPLEMENTASI REST API

2. Setup Server

server.js untuk menggunakan lokal host

```
JS server.js > ...
1 const express = require('express');
2 const bodyParser = require('body-parser'); 487.5k (gzipped: 212.1k)
3 const penggunaController = require('./controllers/penggunaController');
4 const lowonganController = require('./controllers/lowonganController');
5 const konsultasiKarirController = require('./controllers/konsultasiKarirController');
6 const pembayaranController = require('./controllers/pembayaranController');
7 require("dotenv").config(); 6.3k (gzipped: 2.8k)
8
9 const app = express();
10 const PORT = 5000;
11
12 app.use(bodyParser.json());
13
14 app.use('/api', penggunaController);
15 app.use('/api', lowonganController);
16 app.use('/api', konsultasiKarirController);
17 app.use('/api', pembayaranController);
18
19 app.listen(PORT, () => {
20   console.log(`Server berjalan di http://localhost:\${PORT} untuk layanan pencarian kerja dan bimbingan karir`);
21 });
22
23 //Kelompok 1
```

IMPLEMENTASI REST API

3. Membuat Route dan Operasi CRUD

penggunaController.js dan validasi data JWT

```
controllers > js penggunaController.js > router.get('/pengguna') callback
1 const express = require('express');
2 const router = express.Router();
3 const db = require('../config/db'); // Gunakan db dari config.js
4 const bcrypt = require('bcrypt'); Calculating...
5 const jwt = require('jsonwebtoken'); 53.2k (gzipped: 15.9k)
6 require('dotenv').config(); // Pastikan menggunakan environment variable 6.3k (gzipped: 2.8k)
7
8 const SECRET_KEY = process.env.SECRET_KEY; // Gunakan SECRET_KEY dari environment variable
9
10 // Registrasi pengguna untuk platform pencari lowongan kerja dan konsultasi pra-kerja
11 router.post('/register', async (req, res) => {
12   const { nama, email, kata_sandi, peran } = req.body;
13
14   if (!nama || !email || !kata_sandi) {
15     return res.status(400).json({ message: 'Semua field harus diisi!' });
16   }
17
18   try {
19     const [existingUser] = await db.promise().query('SELECT * FROM pengguna WHERE email = ?', [email]);
20     if (existingUser.length > 0) {
21       return res.status(400).json({ message: 'Email sudah digunakan!' });
22     }
23
24     const hashedPassword = await bcrypt.hash(kata_sandi, 10);
25
26     await db.promise().query('INSERT INTO pengguna (nama, email, kata_sandi, peran) VALUES (?, ?, ?, ?)', [
27       nama,
28       email,
29       hashedPassword,
30       peran
31     ]);
32
33     const token = jwt.sign({ id: existingUser[0].id }, SECRET_KEY, { expiresIn: '1h' });
34
35     res.json({ message: 'Pengguna berhasil ditambahkan!', token });
36   } catch (error) {
37     console.error(error);
38     res.status(500).json({ message: 'Terjadi kesalahan pada server!' });
39   }
40 }
41
42 module.exports = router;
```

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

IMPLEMENTASI REST API

3. Membuat Route dan Operasi CRUD

konsultasiKarirController.js

```
controllers > JS konsultasiKarirController.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const db = require('../config/db');

4
5  // POST /konsultasiKarir - Membuat janji konsultasi bimbingan pra kerja baru
6  router.post('/karir', async (req, res) => {
7    const { id_pelamar, id_mentor, jadwal_konsultasi } = req.body;

8
9    if (!id_pelamar || !id_mentor || !jadwal_konsultasi) {
10      return res.status(400).json({ message: 'Semua field harus diisi!' });
11    }

12    try {
13      await db.promise().query(
14        `
15          INSERT INTO janji_konsultasi (id_pelamar, id_mentor, jadwal_konsultasi)
16          VALUES (?, ?, ?)
17        `,
18        [id_pelamar, id_mentor, jadwal_konsultasi]
19      );
20      res.status(201).json({ message: 'Janji konsultasi bimbingan pra kerja berhasil dibuat' });
21    } catch (error) {
22      console.error('Error creating appointment:', error);
23    }
  
```

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

IMPLEMENTASI REST API

3. Membuat Route dan Operasi CRUD

lowonganController.js

```
controllers > JS lowonganController.js > router.post('/lowongan') callback
1  const express = require('express');
2  const router = express.Router();
3  const db = require('../config/db');

4
5 // POST /lowongan - Membuat Lowongan Pekerjaan Baru
6 router.post('/lowongan', async (req, res) => {
7   const { id_perusahaan, deskripsi } = req.body;

8
9   if (!id_perusahaan || !deskripsi) {
10     return res.status(400).json({ message: 'Semua field harus diisi!' });
11   }

12
13   try {
14     await db.promise().query(
15       `INSERT INTO lowongan_pekerjaan (id_perusahaan, deskripsi)
16       VALUES (?, ?)
17       `,
18       [id_perusahaan, deskripsi]
19     );
20     res.status(201).json({ message: 'Lowongan pekerjaan berhasil dibuat' });
21   } catch (error) {
22     console.error('Error creating job listing:', error);
23   }
24 });

25 module.exports = router;
```

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

IMPLEMENTASI REST API

3. Membuat Route dan Operasi CRUD

pembayaranController.js

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

4
5  // POST /pembayaran - Membuat pembayaran untuk layanan bimbingan pra kerja
6  router.post('/pembayaran', async (req, res) => {
7    const { id_konsultasi_karir, jumlah, status } = req.body;

8
9    if (!id_konsultasi_karir || !jumlah) {
10      return res.status(400).json({ message: 'Field id_konsultasi_karir dan jumlah harus diisi!' });
11    }

12    try {
13      await db.promise().query(
14        `
15          INSERT INTO pembayaran (id_konsultasi_karir, jumlah, status)
16          VALUES (?, ?, ?)
17        `,
18        [id_konsultasi_karir, jumlah, status || 'tertunda']
19      );
20      res.status(201).json({ message: 'Pembayaran untuk konsultasi bimbingan pra kerja berhasil dibuat' });
21    } catch (error) {
22      console.error('Error creating pembayaran:', error);
23      res.status(500).json({ message: 'Internal Server Error' });
24    }
25  }
```

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

TUJUAN PENELITIAN

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec ullamcorper nisl et justo dignissim consequat. Integer tincidunt placerat libero. Ut non tincidunt nisl.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec ullamcorper nisl et justo dignissim consequat. Integer tincidunt placerat libero. Ut non tincidunt nisl. Donec et nulla ut nisl aliquam maximus sit amet eu orci.

PENGUJIAN API DENGAN “POSTMAN”

penggunaController.js

Pengujian dengan endpoint :

1. GET /api/pengguna : Menampilkan semua pengguna yang telah login
2. POST /api/register : Mendaftarkan pengguna baru
3. POST/api/login : Menambahkan pengguna baru dan menampilkan kode JWT

4. GET/api/pengguna/id : Menampilkan pengguna berdasarkan id dan memasukan JWT
5. PUT/api/pengguna/id : Merubah data pengguna
6. DELETE/api/pengguna/id: Menghapus pengguna

PENGUJIAN API DENGAN “POSTMAN”

lowonganController.js

Pengujian dengan endpoint :

1. GET /api/konsultasi : Menampilkan semua lowongan

2. GET/api/konsultasi/id : Menampilkan hasil lowangn berdasarkan id

3. POST /api/konsultasi : Membuat data lowongan

4. PUT/api/konsultasi/id : Merubah data hasil lowongan

5. DELETE/api/konsultasi/id : Menghapus data hasil lowongan

PENGUJIAN API DENGAN “POSTMAN”

konsultasiKarirController.js

Pengujian dengan endpoint :

1. GET /api/konsultasi : Menampilkan semua hasil konsultasi
2. GET/api/konsultasi/id : Menampilkan hasil konsultasi berdasarkan id
3. POST /api/konsultasi : Membuat data untuk hasil konsultasi

4. PUT/api/konsultasi/id : Merubah data hasil konsultasi
5. DELETE/api/konsultasi/id : Menghapus data hasil konsultasi

PENGUJIAN API DENGAN “POSTMAN”

pembayaranController.js

Pengujian dengan endpoint :

1. GET /api/konsultasi : Menampilkan semua hasil konsultasi
2. GET/api/konsultasi/id : Menampilkan hasil konsultasi berdasarkan id
3. POST /api/konsultasi : Membuat data untuk hasil konsultasi

4. PUT/api/konsultasi/id : Merubah data hasil konsultasi
5. DELETE/api/konsultasi/id : Menghapus data hasil konsultasi

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 yang berpengaruh besar dalam industri dengan teknologi.

TERIMAKASIH

By Kelompok 1 (Pemograman Berbasis Patform)

<https://github.com/RendiRuswandi/Project-UAS-PBP>