**LAPORAN PRAKTIKUM**

**PEMROGRAMAN WEB**

****

**oleh:**

**NOVANDI KEVIN PRATAMA [20081010005]**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”**

**JAWA TIMUR**

**2022**

# *Javasxript dan JQuery*

## Deskripsi

*Javascript* adalah bahasa pemrograman yang dapat terintegrasi langsung dengan halaman web bertujuan untuk memanipulasi bentuk, warna, ukuran, dan gaya lainnya. Dengan menggunakan *javascript* halaman web akan menjadi lebih interaktif dikarenakan dapat memberikan reaksi ketika suatu komponen pada halaman web berinteraksi dengan pengunjung.

*JQuery* adalah salah satu *library javascript* yang terdiri dari beberapa fungsi sederhana namun memiliki fitur yang lengkap. Dengan adanya *jquery,* dapat menyeleksi elemen halaman web lalu memanipulasi bentuk, warna, ukuran dan gaya lainnya.

## Tujuan

* + 1. Memahami dasar-dasar dari *Javascript* dan *JQuery*
    2. Memahami kegunaan dan penggunaan dari *Javascript* dan *JQuery*

## DASAR TEORI

* + 1. ***Javascript***

*Javascript* adalah bahasa pemrograman yang dipakai untuk membuat halaman web dapat berinteraksi dengan pengguna dan dapat memberikan reaksi ketika elemen berinteraksi [1]. *Javascript* terhubung dengan *HTML* sehingga *Javascript* dapat memanipulasi setiap komponen yang telah dibuat. Dengan adanya, *javacript* dapat memberikan *logic* pada komponen halaman web sehingga menjadi interaktif dengan pengguna*.*

* + 1. ***Document Object Model (DOM)***

*Document Object Model* adalah hierarki objek yang mewakili suatu halaman web browser, riwayat browser, halaman web yang sedang dimuat, dan elemen web penting lainnya [2]. *Document Object Model* merupakan kemampuan dari *javascript* untuk memanipulasi semua elemen yang terdapat pada suatu halaman web seperti bentuk, warna, posisi, dan gaya lainnya.

* + 1. ***JQuery***

*JQuery* merupakan salah satu *library javascript* yang memuat banyak fungsi yang dapat langsung dipakai untuk halaman web [3]. *JQuery* dibuat untuk menyeleksi elemen-elemen pada halaman web lalu memanipulasi bentuk, warna, posisi, dan gaya lainnya.

* + 1. ***Database***

*Database* merupakan kumpulan berkas *file* yang saling terhubung, hubungan tersebut ditandai dengan adanya kunci pada setiap *file* [4]. *Database* merupakan media penyimpanan data yang berisi informasi yang dapat ditambah, diedit, dihapus, dan ditampilkan pada halaman web.

* + 1. ***WebSQL***

*WebSQL* adalah API untuk menyimpan data lokal menggunakan turunan *SQL* (seperti *SQLite*) [5]. *WebSQL* merupakan fitur terbaru dari *HTML5*. *WebSQL* dapat digunakan sebagai *database* berbasis *client* sehingga tidak memerlukan *server.*

## SOAL dan JAWABAN

### Buatlah sebuah tampilan segitiga bintang piramida yang mengarah ke empat sisi menggunakan *Javascript.*

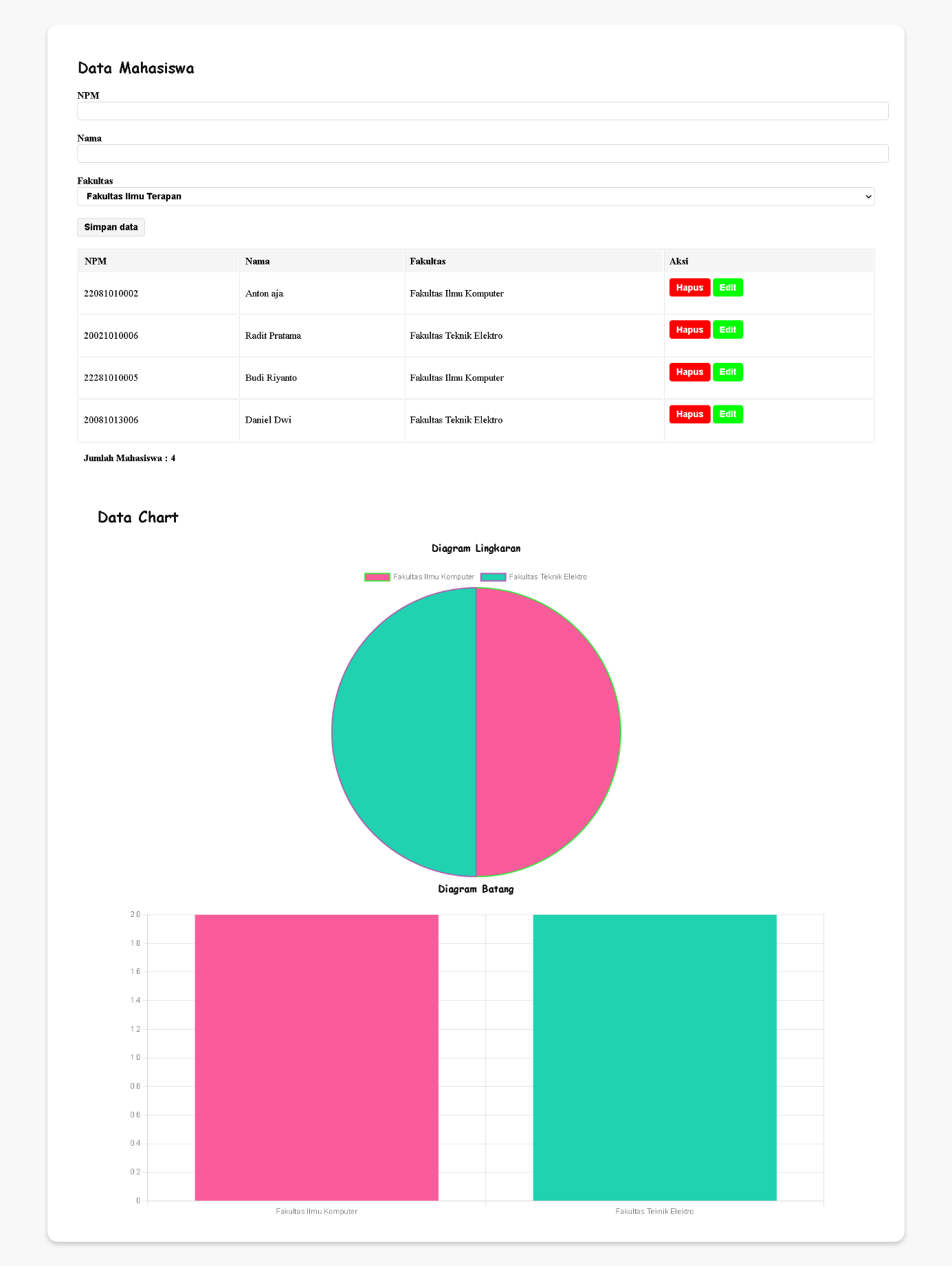
Sebuah gambar berisi teks

Deskripsi dibuat secara otomatis

Gambar 1.1. Tampilan Web Formasi Piramida Bintang

Tampilan formasi bintang piramida terdiri dari dua bagian yaitu, *form* untuk pengisian jumlah baris dan arah piramida, lalu div kosong untuk menampilkan hasil dari formasi bintang.

### Buatlah aplikasi web yang dapat melakukan proses CRUD menggunakan kombinasi *JQuery* dengan *database* yang dipilih dengan saudari adapun pilihannya *Firebase*, *MySQL*, dan *WebSQL*.



Gambar 1.2. Tampilan Web Pengolahan Data Mahasiswa

Web pengolahan data mahasiswa terdiri dari tiga bagian yaitu *form,* data tabel mahasiswa, dan grafik rekap jumlah mahasiswa setiap fakultas. *Database* yang digunakan adalah *WebSQL*.

# PENULISAN KODE

## 2.1. *Javascript*

|  |
| --- |
| <div class="card">  <h2 class="title">Form Pyramid</h2>  <form id="formPiramid">  <div class="form-group">  <label>Masukkan jumlah row</label>  <input type="number" name="row" id="row" class="form-control" />  </div>  <div class="form-group">  <label>Masukkan arah Piramid</label>  <select name="arah" class="form-control">  <option value="up">Atas</option>  <option value="down">Bawah</option>  <option value="left">Kiri</option>  <option value="right">Kanan</option>  </select>  </div>  <div class="form-group">  <button type="submit" class="btn btn-primary" id="btn-submit">  Submit  </button>  </div>  </form>  </div>  <div class="card">  <h2 class="title">Hasil</h2>  <div id="result"></div>  </div> |

**Pada potongan kode di atas merupakan halaman web terdiri dari *form* dan hasil. Pada *form* berisi jumlah baris dan arah piramida. Sedangkan bawahnya terdapat *div* kosong untuk dimanipulasi dan diisi formasi piramida.**

|  |
| --- |
| constructor(row, arah) {  this.row = row;  this.enter = "<br/>";  this.space = "&nbsp;&nbsp;";  this.star = "\*";  switch (arah) {  case "right":  this.onRight();  break;  case "left":  this.onLeft();  break;  case "up":  this.onTop();  break;  case "down":  this.onBottom();  break;  }  } |

**Pada potongan kode di atas merupakan bagian *constructor* pada *class Star* yang digunakan untuk membuat formasi piramida. Pada *constructor* digunakan untuk inisialisasi variabel pada *class* dan pemanggilan *method*.**

|  |
| --- |
| onTop() {  this.result = "";  for (let i = 1; i <= this.row; i++) {  for (let j = 1; j <= this.row - i; j++) {  this.result += this.space;  }  for (let k = 0; k < 2 \* i - 1; k++) {  this.result += this.star;  }  this.result += this.enter;  }  }  onBottom() {  this.result = "";  for (let i = 0; i < this.row; i++) {  for (let j = 0; j < i; j++) {  this.result += this.space;  }  for (let k = 0; k < 2 \* (this.row - i) - 1; k++) {  this.result += this.star;  }  this.result += this.enter;  }  }  onLeft() {  this.result = this.space;  for (let i = 1; i <= this.row; i++) {  for (let j = 0; j < this.row - i; j++) {  this.result += this.space + this.space;  }  for (let k = 0; k < i; k++) {  this.result += this.star + this.space;  }  this.result += this.enter + this.space;  }  for (let i = 1; i <= this.row - 1; i++) {  for (let j = 0; j < i; j++) {  this.result += this.space + this.space;  }  for (let k = 0; k < this.row - i; k++) {  this.result += this.star + this.space;  }  this.result += this.enter + this.space;  }  }  onRight() {  this.result = "";  for (let i = 1; i <= this.row; i++) {  for (let j = 0; j < i; j++) {  this.result += this.space + this.star;  }  this.result += this.enter;  }  for (let i = 1; i <= this.row - 1; i++) {  for (let j = 0; j < this.row - i; j++) {  this.result += this.space + this.star;  }  this.result += this.enter;  }  } |

**Pada potongan kode di atas merupakan pembuatan formasi sesuai arah yang ditentukan dengan menggunakan pengulangan lalu hasilnya akan dimasukkan pada variabel *result*.**

|  |
| --- |
| const resultPanel = document.getElementById("result");  const form = document.getElementById("formPiramid");  const resultPanel1 = document.getElementById("result1");  const resultPanel2 = document.getElementById("result2");  form.addEventListener("submit", (e) => {  e.preventDefault();  const formData = new FormData(e.target);  const row = parseInt(formData.get("row"));  const arah = formData.get("arah");  if (isNaN(row)) {  resultPanel.innerHTML = "<p class='error'>Input harus angka</p>";  alert("Input harus angka");  } else if (row < 0) {  resultPanel.innerHTML = "<p class='error'>Input harus lebih dari 0</p>";  alert("Input harus lebih dari 0");  } else {  resultPanel.innerHTML = new Star(row, arah).result;  }  }); |

**Pada potongan kode di atas menyeleksi setiap elemen yang terkait dan dimasukkan dalam variabel. Pada *form* diberikan *event* ketika telah *submit* lalu diberikan ”*e.preventDefault()*” untuk mencegah reaksi *default* dari *submit* *form* dan melakukan pemanggilan *class Star*. Lalu mengatur *“innerHTML”* pada elemen diisi dengan variabel *result* pada *class Star*.**

## *2.2. JQuery*

|  |
| --- |
| <div class="container">  <div class="card">  <div class="data-mahasiswa">  <h2 class="title">Data Mahasiswa</h2>  <form id="formMahasiswa">  <input type="hidden" name="id" />  <div class="form-group">  <label for="nim">NPM</label>  <input required type="text" name="npm" id="nim" class="form-control" />  </div>  <div class="form-group">  <label for="nama">Nama</label>  <input required type="text" name="nama" id="nama" class="form-control" />  </div>  <div class="form-group">  <label for="fakultas">Fakultas</label>  <select name="fakultas" id="fakultas" class="form-control">  <option value="Fakultas Ilmu Terapan">  Fakultas Ilmu Terapan  </option>  <option value="Fakultas Teknik Elektro">  Fakultas Teknik Elektro  </option>  <option value="Fakultas Ilmu Komputer">  Fakultas Ilmu Komputer  </option>  <option value="Fakultas Teknik Industri">  Fakultas Teknik Industri  </option>  <option value="Fakultas Teknik Sipil">  Fakultas Teknik Sipil  </option>  <option value="Fakultas Ekonomi">Fakultas Ekonomi</option>  <option value="Fakultas Hukum">Fakultas Hukum</option>  <option value="Fakultas Pertanian">Fakultas Pertanian</option>  <option value="Fakultas Kedokteran">Fakultas Kedokteran</option>  <option value="Fakultas Ilmu Budaya">  Fakultas Ilmu Budaya  </option>  <option value="Fakultas Teknologi Pertanian">  Fakultas Teknologi Pertanian  </option>  <option value="Fakultas Pertanian">Fakultas Pertanian</option>  <option value="Fakultas Peternakan">Fakultas Peternakan</option>  <option value="Fakultas Keperawatan">  Fakultas Keperawatan  </option>  <option value="Fakultas Kesehatan Masyarakat">  Fakultas Kesehatan Masyarakat  </option>  <option value="Fakultas Ilmu Sosial dan Ilmu Politik">  Fakultas Ilmu Sosial dan Ilmu Politik  </option>  </select>  </div>  <button class="btn btn-submit">Simpan data</button>  </form>  <table class="table table-mahasiswa">  <thead>  <tr>  <th>NPM</th>  <th>Nama</th>  <th>Fakultas</th>  <th>Aksi</th>  </tr>  </thead>  <tbody></tbody>  </table>  <p class="label">  Jumlah Mahasiswa : <span id="jumlahMahasiswa"></span>  </p>  </div>  <div class="data-chart">  <h2 class="title">Data Chart</h2>  <div class="chart-container">  <h4 class="title text-center">Diagram Lingkaran</h4>  <canvas id="chartPie"></canvas>  </div>  <div class="chart-container">  <h4 class="title text-center">Diagram Batang</h4>  <canvas id="chartBar"></canvas>  </div>  </div>  </div>  </div>} |

**Pada potongan kode di atas bagian HTML yang terbagi tiga bagian yaitu *form*, data tabel, dan grafik. *Form* terdiri dari NPM, nama, dan fakultas. Fakultas akan berupa *select* sehingga pengguna tinggal memilih saja tanpa perlu mengetik fakultas. Untuk grafik terdiri dari grafik batang dan grafik lingkaran.**

|  |
| --- |
| class Database {  constructor() {  this.db = openDatabase("DataMahasiswa", "1.0", "DataMahasiswa", 2000);  this.createTable();  }  createTable() {  this.exc(`create table if not exists Mahasiswa  (  NPM TEXT PRIMARY KEY,  NamaMahasiswa TEXT,  Fakultas TEXT  );`);  }  query(sql, callback = () => {}) {  this.db.transaction(function (tx) {  tx.executeSql(sql, [], function (tx, results) {  callback(results.rows, results.rows.length);  });  });  }  exc(sql, data = [], callback = () => {}) {  this.db.transaction(function (tx) {  tx.executeSql(sql, data, function (tx, results) {  callback(results);  });  });  }  } |

**Pada potongan kode di atas merupakan pembuatan *database WebSQL*. Fungsi *query* digunakan untuk mengambil data, sedangkan fungsi *exc* digunakan untuk mengeksekusi *sql*. Lalu pembuatan *table* pada fungsi *createTable* yang nanti dipanggil pada *constructor*.**

|  |
| --- |
| const getData = () => {  database.query("SELECT \* FROM Mahasiswa", (data) => {  renderData(data);  });  };  const renderData = (data) => {  const dataRekap = {  fakultas: [],  jumlah: [],  };  // Kosongkan body table  $(".table-mahasiswa tbody").empty();  // Lakukan pengisian  for (let i = 0; i < data.length; i++) {  // Tambah data setiap row  $(".table-mahasiswa tbody").append(`<tr>  <td>${data[i].NPM}</td>  <td>${data[i].NamaMahasiswa}</td>  <td>${data[i].Fakultas}</td>  <td>  <button class="btn btn-danger btn-delete" data-id="${data[i].NPM}">Hapus</button>  <button class="btn btn-primary btn-edit" data-id="${data[i].NPM}">Edit</button>  </td>  </tr>`);  // Mencari index dari fakultas  const indexFakultas = dataRekap.fakultas.indexOf(data[i].Fakultas);  // Jika index < 0 maka fakultas tidak ada di array  if (indexFakultas === -1) {  // Tambahkan data fakultas  dataRekap.fakultas.push(data[i].Fakultas);  dataRekap.jumlah.push(1);  } else {  // Tambahkan jumlah mahasiswa  dataRekap.jumlah[indexFakultas]++;  }  }  // add event delete  $(".btn-delete").on("click", (e) => {  const npm = $(e.target).data("id");  if (confirm(`Apakah anda yakin ingin menghapus data dengan npm ${npm}?`)) {  database.exc(  "DELETE FROM Mahasiswa WHERE NPM = ?", [npm.toString()],  () => {  alert("Berhasil");  getData();  }  );  }  });  // add event edit  $(".btn-edit").on("click", (e) => {  const npm = $(e.target).data("id");  database.query(  `SELECT \* FROM Mahasiswa WHERE NPM = '${npm.toString()}'`,  (data) => {  if (data.length > 0) {  $("input[name='npm']").val(data[0].NPM);  $("input[name='id']").val(data[0].NPM);  $("input[name='nama']").val(data[0].NamaMahasiswa);  $("select[name='fakultas'] option").removeAttr("selected");  $(`select[name='fakultas'] option[value='${data[0].Fakultas}']`).attr(  "selected",  "true"  );  }  }  );  });  // Ubah jumlah mahasiswa  $("#jumlahMahasiswa").text(data.length);  setDataChart(dataRekap);  };  // Mendapatkan config chart sesuai jenis  const getConfig = (data, type = "pie") => {  const config = {  type: type,  data: data,  options: {  responsive: true,  maintainAspectRatio: false,  plugins: {  legend: {  display: type === "pie" ? true : false,  },  },  },  };  return config;  };  const setDataChart = (value) => {  const data = {  labels: value.fakultas,  datasets: [{  label: value.fakultas,  backgroundColor: randomColor(value.fakultas.length),  borderColor: randomColor(value.fakultas.length),  data: value.jumlah,  }, ],  };  if (chartBar) chartBar.destroy();  if (chartPie) chartPie.destroy();  // Atur data chart  chartPie = new Chart($("#chartPie"), getConfig(data, "pie"));  // Ubah legend menjadi jumlah mahasiswa  data.datasets[0].label = "Jumlah Mahasiswa";  chartBar = new Chart($("#chartBar"), getConfig(data, "bar"));  };  // Untuk mendapatkan array warna random  const randomColor = (length) => {  let color = [];  for (let i = 0; i < length; i++) {  const colorTemp = `rgba(${Math.floor(Math.random() \* 255)}, ${Math.floor(  Math.random() \* 255  )}, ${Math.floor(Math.random() \* 255)})`;  if (color.indexOf(colorTemp) === -1) {  color.push(colorTemp);  } else {  i--;  }  }  return color;  }; |

**Pada potongan kode di atas untuk pengambilan data dan memasukkannya pada tabel dan grafik. Lalu dilakukan pengulangan dan pembentukan baris dan kolum lalu dimasukkan pada tabel. Setelah itu, dilakukan pengelompokan data berdasarkan fakultas dan memasukkan datanya pada grafik. Untuk pembuatan grafik menggunakan *library Chart.js***

|  |
| --- |
| const saveData = (data, id = null) => {  if (id) {  // edit  // cek npm telah ada pada database  database.query(  `select \* from Mahasiswa where NPM = '${data.npm}' and NPM != '${id}'`,  (exists) => {  if (exists.length > 0) {  alert("NPM sudah ada");  } else {  database.exc(  `update Mahasiswa set NPM = '${data.npm}', NamaMahasiswa = '${data.nama}', Fakultas = '${data.fakultas}' where NPM = '${id}'`  );  alert("Data berhasil diubah");  $("#formMahasiswa").trigger("reset");  getData();  }  }  );  } else {  // add  database.query(  `select \* from Mahasiswa where NPM = '${data.npm}'`,  (exists) => {  if (exists.length > 0) {  alert("NPM sudah ada");  } else {  database.exc(  `insert into Mahasiswa (NPM,NamaMahasiswa,Fakultas) values ('${data.npm}','${data.nama}','${data.fakultas}')`  );  console.log(  `insert into Mahasiswa (NPM,NamaMahasiswa,Fakultas) values ('${data.npm}','${data.nama}','${data.fakultas}')`  );  alert("Data berhasil ditambahkan");  $("#formMahasiswa").trigger("reset");  getData();  }  }  );  }  }; |

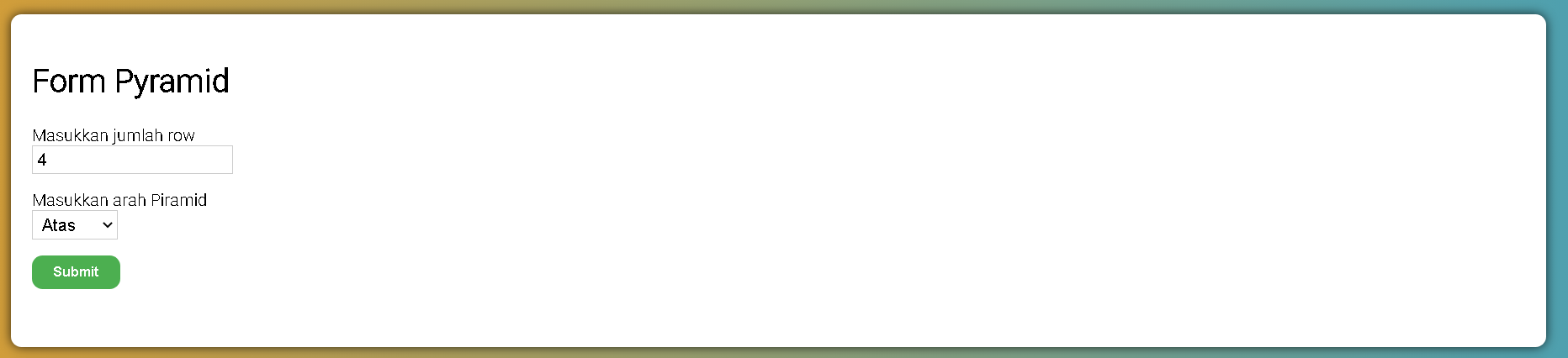
**Pada potongan kode di atas merupakan fungsi untuk menyimpan data. Fungsi ini membutuhkan satu parameter wajib dan satu parameter opsional. Ketika parameter opsional diisi maka fungsi akan mengubah data jika tidak diisi maka akan menambah data. Sebelum menyimpan data dilakukan pengecekan NPM, jika sudah ada maka akan diberikan peringatan.**

|  |
| --- |
| var database;  var chartBar, chartPie;  $(document).ready(() => {  database = new Database();  getData();  $(".btn-refresh").on("click", getData);  $("#formMahasiswa").on("submit", (e) => {  e.preventDefault();  const data = {  npm: $("[name='npm']").val(),  nama: $("[name='nama']").val(),  fakultas: $("[name='fakultas']").val(),  };  let id = $("[name='id']").val();  if (id == "") {  id = null;  }  saveData(data, id);  });  }); |

**Pada potongan kode di atas merupakan bagian pemanggilan fungsi pengambilan data dan pendeklarasian *event* pada *form*.**

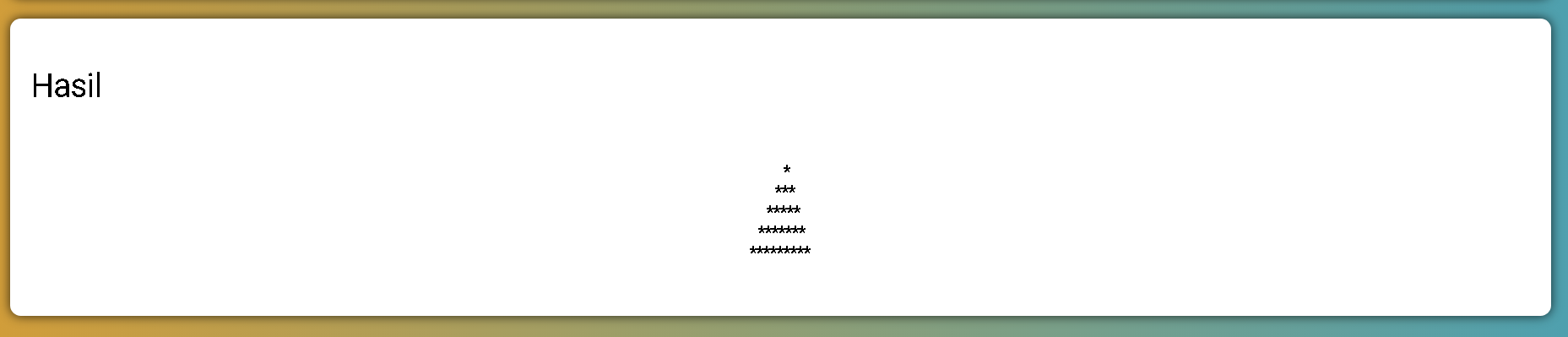
# SCREENSHOT HASIL PERCOBAAN

## 3.1. *Javascript*



Gambar 3.1. Form Inputan Piramida

Pada gambar 3.1. menunjukkan bagian *form* piramida yang berisi meminta inputan jumlah baris dan arah piramida.



Gambar 3.2. Hasil Piramida

Pada gambar 3.2. menunjukkan bagian yang berisi hasil piramida sesuai inputan dari pengguna arah maupun jumlah baris.

## 3.2. *JQuery*

Sebuah gambar berisi meja

Deskripsi dibuat secara otomatis

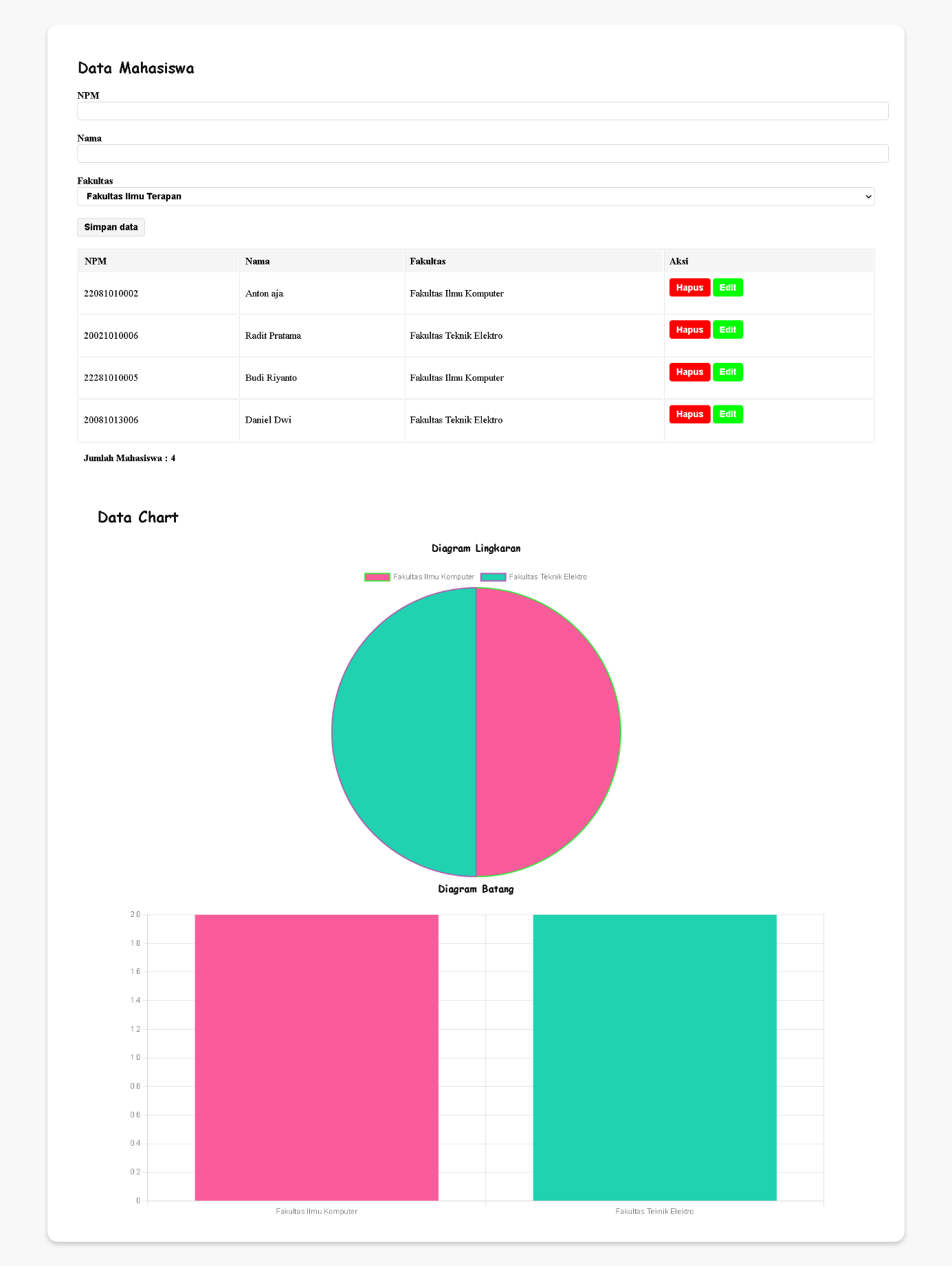
Gambar 3.3. *Form Mahasiswa*

Pada gambar 3.3. menunjukkan bagian *form* yang digunakan untuk mengubah dan menambah data mahasiswa.



Gambar 3.4. Tabel Mahasiswa

Pada gambar 3.4. menunjukkan data mahasiswa yang ditampilkan menggunakan tabel.



Gambar 3.5. Grafik Rekap Mahasiswa Setiap Fakultas

Pada gambar 3.5. menunjukkan grafik yang menunjukkan jumlah mahasiswa pada setiap fakultas. Grafik yang ditampikan berupa grafik batang dan grafik lingkaran.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | R. H. Sianipar, Pemrograman Javascript: Teori Dan Implementasi, Bandung: Penerbit INFORMATIKA, 2015. |
| [2] | C. Easttom, Advanced Javascript, Burlington: Jones & Bartlett Learning, 2007. |
| [3] | R. H. Sianipar, JQuery & Ajax Untuk Web Designer, Surabaya: Penerbit ANDI, 2016. |
| [4] | I. H. Kristanto, Konsep & Perancangan Database, Yogyakarta: Penerbit Andi, 1994. |
| [5] | S. Sarris, HTML5 Unleashed, London: Pearson Education, 2013. |