LAPORAN TUGAS PROYEK AKHIR KULIAH

PEMROGRAMAN BERBASIS OBJEK

Aplikasi Perpustakaan Bookcamps



Dosen Pengampu:

Bayu Adi Nugroho, Ph.D

Disusun Oleh:

Rizki Cahyani Fitonah 09020622042

UNIVERSITAS ISLAM NEGERI SUNAN AMPEL SURABAYA FAKULTAS SAINS DAN TEKNOLOGI SISTEM INFORMASI 2023/2024

BABI

PENDAHULUAN

1.1 Latar Belakang

Perpustakaan merupakan salah satu elemen kunci dalam proses pembelajaran dan penelitian di perguruan tinggi. Namun, pengelolaan perpustakaan yang efisien dan efektif menjadi tantangan tersendiri, terutama dengan adanya jumlah koleksi yang terus bertambah dan kebutuhan pengguna yang beragam. Dalam konteks ini, pembuatan aplikasi perpustakaan menjadi solusi yang relevan untuk meningkatkan efisiensi dan memberikan pengalaman pengguna yang lebih baik. Aplikasi perpustakaan dapat membantu dalam manajemen koleksi, peminjaman dan pengembalian buku, serta memberikan akses cepat dan mudah terhadap sumber informasi digital.

Pembuatan aplikasi perpustakaan ini tidak hanya mencerminkan upaya untuk menjawab tantangan aktual dalam pengelolaan sumber daya informasi, tetapi juga merupakan bagian integral dari tahap akhir pendidikan. Proyek ini secara khusus diarahkan sebagai tugas akhir dalam kurikulum perkuliahan, dengan penugasan langsung dari dosen pembimbing. Pembuatan aplikasi perpustakaan bukan hanya merupakan proyek teknis semata, tetapi juga merupakan demonstrasi kemampuan mahasiswa dalam menerapkan konsep dan teori yang telah mereka pelajari selama perkuliahan ke dalam suatu solusi praktis dan tangguh.

1.2 Tujuan

Adapun tujuan dari proyek ini adalah untuk memberikan kesempatan kepada kami sebagai mahasiswa untuk menerapkan pengetahuan dan keterampilan yang telah dipelajari selama masa studi.

BAB II

LANDASAN TEORI

2.1 Dasar Teori

Menurut Wikipedia pemrograman berorientasi objek atau *object-oriented programming* yang biasa disingkat OOP merupakan <u>paradigma pemrograman</u> berdasarkan konsep "objek", yang dapat beris data, dalam bentuk *field* atau dikenal juga sebagai atribut; serta kode, dalam bentuk fungsi/prosedur atau dikenal juga sebagai *method*. Semua data dan fungsi di dalam paradigma ini dibungkus dalam *kelas-kelas* atau *objek-objek*. Bandingkan dengan logika <u>pemrograman</u> terstruktur. Setiap objek dapat menerima <u>pesan</u>, memproses data, dan mengirim pesan ke objek lainnya. Berikut cakupan dasar pemrograman berbasis objek:

1. Kelas dan Objek:

- Definisi Kelas: Kelas adalah struktur dasar dalam OOP yang mendefinisikan atribut (variabel) dan metode (fungsi) yang dapat dimiliki oleh suatu objek.
- Objek: Objek adalah instansi konkret dari suatu kelas. Setiap objek memiliki atribut yang unik dan dapat memanggil metode khususnya.

2. Inheritance (Pewarisan):

- Konsep Pewarisan: Inheritance memungkinkan kelas baru ("subclass" atau "child class") mewarisi atribut dan metode dari kelas yang sudah ada ("superclass" atau "parent class").
- Keuntungan: Meningkatkan reusable code dan memfasilitasi pengelolaan hierarki kelas.

3. Polimorfisme:

- Arti Polimorfisme: Polimorfisme memungkinkan suatu metode memiliki nama yang sama tetapi dapat melakukan tindakan yang berbeda tergantung pada objek yang memanggilnya.
- Contoh Penggunaan: Metode overriding dan interface adalah implementasi polimorfisme dalam OOP.

4. Encapsulation (Enkapsulasi):

- Prinsip Enkapsulasi: Enkapsulasi melibatkan pengemasan atribut dan metode ke dalam satu unit tunggal (kelas), dengan mengakses atribut menggunakan metode (getters dan setters).
- Keuntungan: Menyediakan kontrol akses yang lebih baik dan membatasi akses langsung ke atribut.

5. Abstraksi:

• Arti Abstraksi: Abstraksi melibatkan penyembunyian detail implementasi dan hanya mengekspos fungsionalitas penting suatu objek.

• Implementasi di OOP: Interface dan kelas abstrak digunakan untuk menciptakan tingkat abstraksi.

6. Class Diagram:

- Definisi Class Diagram: Class diagram adalah representasi visual dari struktur kelas, hubungan antar kelas, dan sifat atribut dan metode.
- Manfaat: Memberikan pandangan visual yang jelas tentang hubungan antar objek dalam suatu sistem.

7. Persistence API:

Sebuah antarmuka pemrograman aplikasi (API) yang menyediakan metode dan fungsi untuk melakukan operasi persistence, seperti menyimpan, mengambil, dan menghapus data.

2.2 Softwere yang Digunakan

Softwere yang digunakan dalam proyek aplikasi Perpustakaan Bookcamps adalah sebagai berikut:

• Bahasa Pemrograman: Java (JDK 8)

• Aplikasi IDE: Apache NetBeans IDE 15

• Database: PostgreSQL 14

• Report: Jasper Report

2.3 Hardwere yang Digunakan

Hardwere yang digunakan dalam proyek aplikasi Perpustakaan Bookcamps adalah sebagai berikut:

• RAM: 4 GB

• System Operasi: 64-bit

BAB III

DATABASE

- 1. Membuat Database dengan nama "Db perpustakaan"
- 2. Membuat Tabel:
 - a) Create table Buku
 - -- Table: public.buku
 - -- DROP TABLE IF EXISTS public.buku;

```
CREATE TABLE IF NOT EXISTS public.buku
  id_buku character(11) COLLATE pg_catalog."default" NOT NULL,
  judul character varying(50) COLLATE pg_catalog."default",
  subjudul character varying(50) COLLATE pg_catalog."default",
  isbn character(13) COLLATE pg_catalog."default",
  bahasa character varying(15) COLLATE pg_catalog."default",
  penerbit character varying(50) COLLATE pg_catalog."default",
  tahun_terbit character varying(4) COLLATE pg_catalog."default",
  jumlah_halaman integer,
  jumlah_buku integer,
  CONSTRAINT pk_buku PRIMARY KEY (id_buku)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.buku
  OWNER to postgres;
-- Index: buku_pk
-- DROP INDEX IF EXISTS public.buku_pk;
CREATE UNIQUE INDEX IF NOT EXISTS buku_pk
  ON public.buku USING btree
  (id_buku COLLATE pg_catalog."default" ASC NULLS LAST)
```

b) Create Table Peminjam

-- Table: public.peminjam

TABLESPACE pg default;

-- DROP TABLE IF EXISTS public.peminjam;

```
CREATE TABLE IF NOT EXISTS public.peminjam
(
id_peminjam character(5) COLLATE pg_catalog."default" NOT NULL,
nama character varying(30) COLLATE pg_catalog."default",
```

```
nim character varying(15) COLLATE pg_catalog."default",
     fakultas character varying(30) COLLATE pg_catalog."default",
     program_studi character varying(30) COLLATE pg_catalog."default",
     semester integer,
     alamat character varying(100) COLLATE pg_catalog."default",
     telephone character varying(13) COLLATE pg_catalog."default",
     email character varying(30) COLLATE pg_catalog."default",
     CONSTRAINT pk_peminjam PRIMARY KEY (id_peminjam)
   )
   TABLESPACE pg_default;
   ALTER TABLE IF EXISTS public.peminjam
     OWNER to postgres;
   -- Index: peminjam_pk
   -- DROP INDEX IF EXISTS public.peminjam_pk;
   CREATE UNIQUE INDEX IF NOT EXISTS peminjam_pk
     ON public.peminjam USING btree
     (id_peminjam COLLATE pg_catalog."default" ASC NULLS LAST)
     TABLESPACE pg_default;
c) Create Table Skripsi
   -- Table: public.skripsi
   -- DROP TABLE IF EXISTS public.skripsi;
   CREATE TABLE IF NOT EXISTS public.skripsi
     id_skripsi character(11) COLLATE pg_catalog."default" NOT NULL,
     judul character varying(50) COLLATE pg_catalog."default",
     bahasa character varying(15) COLLATE pg_catalog."default",
     fakultas character varying(30) COLLATE pg_catalog."default",
     program_studi character varying(30) COLLATE pg_catalog."default",
     tahun character varying(4) COLLATE pg_catalog."default",
     penulis character varying(30) COLLATE pg_catalog."default",
     pembimbing character varying(30) COLLATE pg_catalog."default",
     jumlah_halaman integer,
     jumlah_salinan integer,
     CONSTRAINT pk_skripsi PRIMARY KEY (id_skripsi)
   )
   TABLESPACE pg_default;
   ALTER TABLE IF EXISTS public.skripsi
     OWNER to postgres;
```

```
d) Create Table Peminjaman
   -- Table: public.peminjaman
  -- DROP TABLE IF EXISTS public.peminjaman;
   CREATE TABLE IF NOT EXISTS public.peminjaman
     id_petugas character(5) COLLATE pg_catalog."default" NOT NULL,
     id_peminjam character(5) COLLATE pg_catalog."default" NOT NULL,
     no_peminjaman character(10) COLLATE pg_catalog."default" NOT
   NULL,
     tanggal_pinjam date,
     tanggal_kembali date,
     keterangan character varying(100) COLLATE pg_catalog."default",
     status character varying(8) COLLATE pg_catalog."default",
     CONSTRAINT pk_peminjaman PRIMARY KEY (id_petugas,
  id_peminjam, no_peminjaman),
     CONSTRAINT fk_peminjam_melakukan_peminjam FOREIGN KEY
   (id_peminjam)
       REFERENCES public.peminjam (id_peminjam) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT,
     CONSTRAINT fk_peminjam_melakukan_petugas FOREIGN KEY
   (id_petugas)
       REFERENCES public.petugas (id_petugas) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT
  )
  TABLESPACE pg_default;
  ALTER TABLE IF EXISTS public.peminjaman
     OWNER to postgres;
e) Create Table Pengarang
  -- Table: public.pengarang
  -- DROP TABLE IF EXISTS public.pengarang;
   CREATE TABLE IF NOT EXISTS public.pengarang
     id_buku character(11) COLLATE pg_catalog."default",
     nama_pengarang character varying(30) COLLATE pg_catalog."default",
     CONSTRAINT fk_pengaran_memiliki_buku FOREIGN KEY (id_buku)
       REFERENCES public.buku (id_buku) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT
```

```
)
   TABLESPACE pg_default;
   ALTER TABLE IF EXISTS public.pengarang
     OWNER to postgres;
f) Create Table Petugas
   -- Table: public.petugas
   -- DROP TABLE IF EXISTS public.petugas;
   CREATE TABLE IF NOT EXISTS public.petugas
     id_petugas character(5) COLLATE pg_catalog."default" NOT NULL,
     nama character varying(30) COLLATE pg_catalog."default",
     email character varying(30) COLLATE pg_catalog."default",
     alamat character varying(100) COLLATE pg_catalog."default",
     jenis_kelamin character(1) COLLATE pg_catalog."default",
     telephone character varying(13) COLLATE pg_catalog."default",
     username character varying(30) COLLATE pg_catalog."default",
     password character varying(100) COLLATE pg_catalog."default",
     CONSTRAINT pk_petugas PRIMARY KEY (id_petugas)
   )
   TABLESPACE pg_default;
   ALTER TABLE IF EXISTS public.petugas
     OWNER to postgres;
g) Create Table Detail Buku
   -- Table: public.detail_buku
   -- DROP TABLE IF EXISTS public.detail_buku;
   CREATE TABLE IF NOT EXISTS public.detail_buku
     id_buku character(11) COLLATE pg_catalog."default" NOT NULL,
     id_petugas character(5) COLLATE pg_catalog."default" NOT NULL,
     id_peminjam character(5) COLLATE pg_catalog."default" NOT NULL,
     no_peminjaman character(10) COLLATE pg_catalog."default" NOT
   NULL,
     jumlah integer,
     CONSTRAINT pk_detail_buku PRIMARY KEY (id_buku, id_petugas,
   id_peminjam, no_peminjaman),
     CONSTRAINT fk_detail_b_memiliki7_peminjam FOREIGN KEY
   (id_peminjam, id_petugas, no_peminjaman)
```

```
REFERENCES public.peminjaman (id_peminjam, id_petugas,
  no_peminjaman) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT,
     CONSTRAINT fk_detail_b_memiliki8_buku FOREIGN KEY (id_buku)
       REFERENCES public.buku (id_buku) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT
  )
  TABLESPACE pg_default;
  ALTER TABLE IF EXISTS public.detail_buku
     OWNER to postgres;
h) Create Table Detail Skripsi
  -- DROP TABLE IF EXISTS public.detail_skripsi;
   CREATE TABLE IF NOT EXISTS public.detail_skripsi
     id_skripsi character(11) COLLATE pg_catalog."default" NOT NULL,
     id_petugas character(5) COLLATE pg_catalog."default" NOT NULL,
     id_peminjam character(5) COLLATE pg_catalog."default" NOT NULL,
     no_peminjaman character(10) COLLATE pg_catalog."default" NOT
  NULL,
     jumlah integer,
     CONSTRAINT pk_detail_skripsi PRIMARY KEY (id_skripsi,
  id_petugas, id_peminjam, no_peminjaman),
     CONSTRAINT fk_detail_s_memiliki5_peminjam FOREIGN KEY
   (id_peminjam, id_petugas, no_peminjaman)
       REFERENCES public.peminjaman (id_peminjam, id_petugas,
  no_peminjaman) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT,
     CONSTRAINT fk_detail_s_memiliki6_skripsi FOREIGN KEY
  (id_skripsi)
       REFERENCES public.skripsi (id_skripsi) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT
  )
  TABLESPACE pg_default;
  ALTER TABLE IF EXISTS public.detail_skripsi
     OWNER to postgres;
  -- Index: detail_skripsi_pk
```

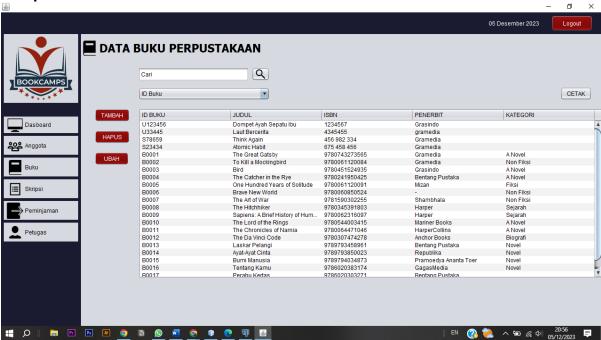
```
-- DROP INDEX IF EXISTS public.detail_skripsi_pk;
   CREATE UNIQUE INDEX IF NOT EXISTS detail_skripsi_pk
     ON public.detail_skripsi USING btree
     (id_skripsi COLLATE pg_catalog."default" ASC NULLS LAST,
   id_petugas COLLATE pg_catalog."default" ASC NULLS LAST,
   id_peminjam COLLATE pg_catalog."default" ASC NULLS LAST,
   no_peminjaman COLLATE pg_catalog."default" ASC NULLS LAST)
     TABLESPACE pg_default;
   -- Index: memiliki5_fk
   -- DROP INDEX IF EXISTS public.memiliki5_fk;
   CREATE INDEX IF NOT EXISTS memiliki5_fk
     ON public.detail_skripsi USING btree
     (id_petugas COLLATE pg_catalog."default" ASC NULLS LAST,
   id_peminjam COLLATE pg_catalog."default" ASC NULLS LAST,
   no_peminjaman COLLATE pg_catalog."default" ASC NULLS LAST)
     TABLESPACE pg_default;
   -- Index: memiliki6_fk
   -- DROP INDEX IF EXISTS public.memiliki6_fk;
   CREATE INDEX IF NOT EXISTS memiliki6_fk
     ON public.detail_skripsi USING btree
     (id_skripsi COLLATE pg_catalog."default" ASC NULLS LAST)
     TABLESPACE pg_default;
i) Create Table Kategori Buku
   -- Table: public.kategori_buku
   -- DROP TABLE IF EXISTS public.kategori_buku;
   CREATE TABLE IF NOT EXISTS public.kategori_buku
     id_buku character(11) COLLATE pg_catalog."default",
     kategori character varying(20) COLLATE pg_catalog."default",
     CONSTRAINT fk_kategori_memiliki1_buku FOREIGN KEY (id_buku)
       REFERENCES public.buku (id_buku) MATCH SIMPLE
       ON UPDATE RESTRICT
       ON DELETE RESTRICT
   )
   TABLESPACE pg_default;
   ALTER TABLE IF EXISTS public.kategori_buku
```

BAB IV PERANCANGAN APLIKASI

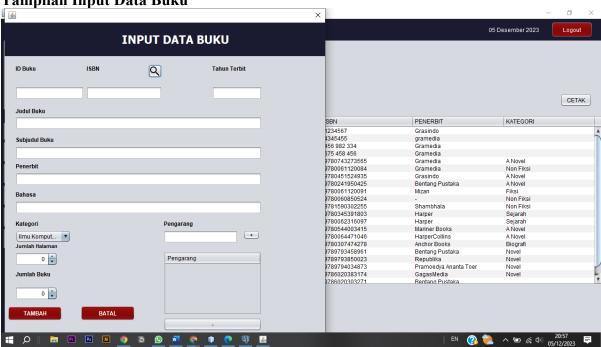
1. Tampilan Dashboard



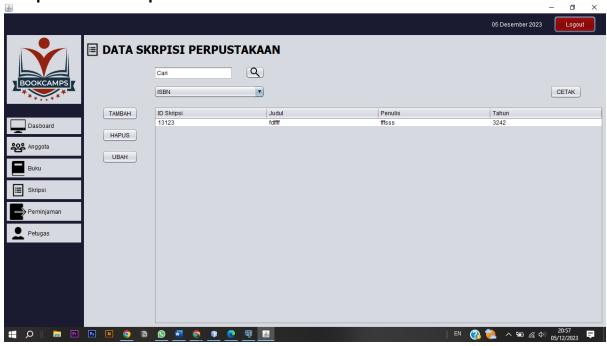
2. Tampilan Menu Buku



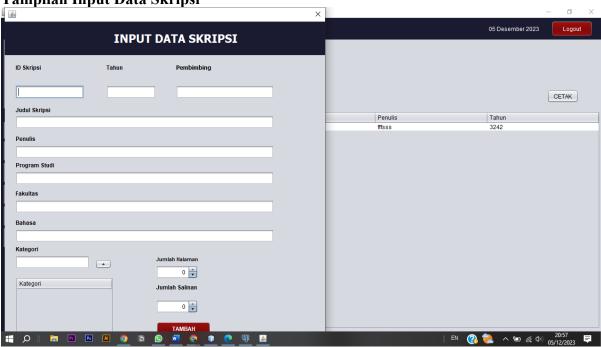
3. Tampilan Input Data Buku



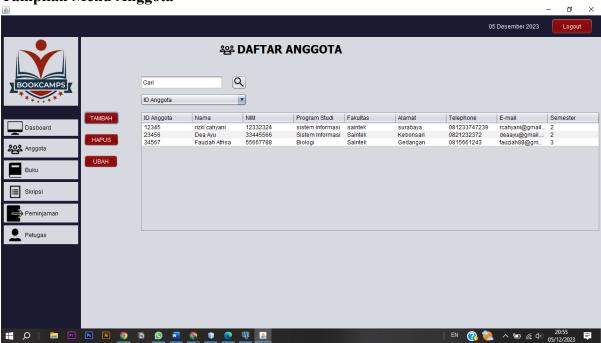
4. Tampilan Menu Skripsi



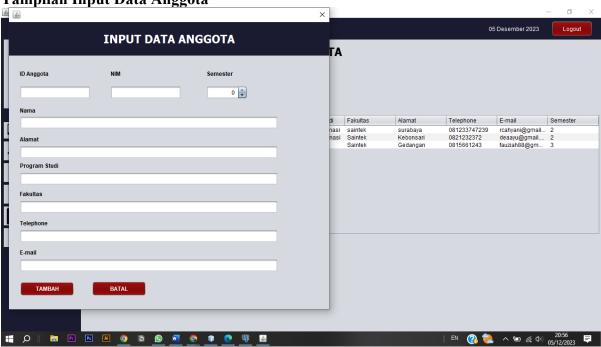
5. Tampilan Input Data Skripsi



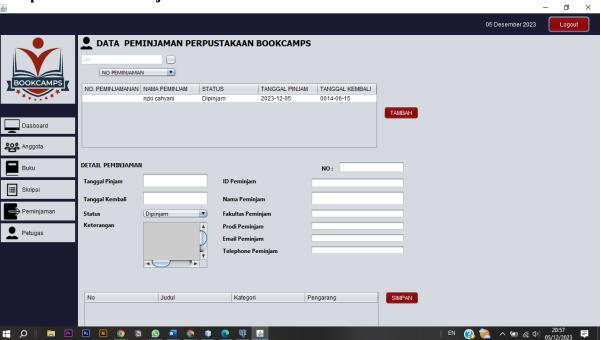
6. Tampilan Menu Anggota



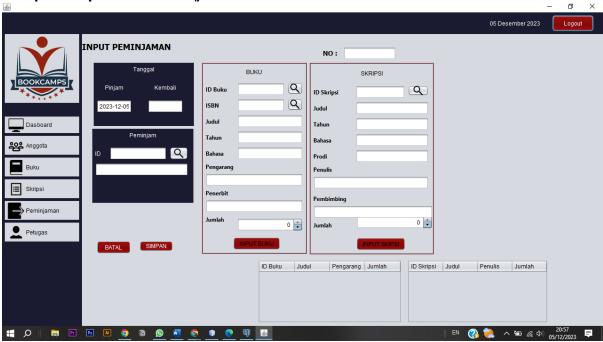
7. Tampilan Input Data Anggota



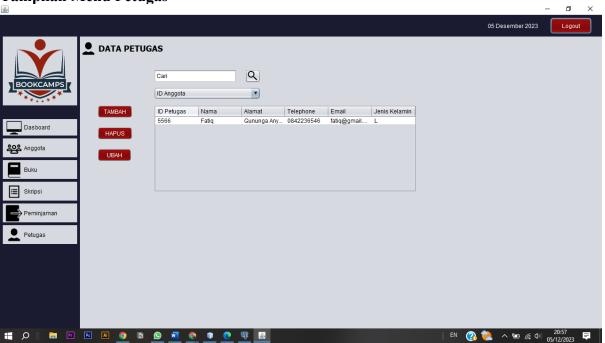
8. Tampilan Menu Peminjaman



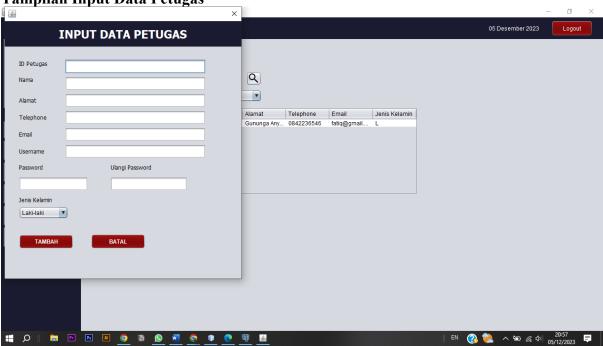
9. Tampilan Input Data Peminjaman



10. Tampilan Menu Petugas



11. Tampilan Input Data Petugas



BAB V KESIMPULAN

Proyek pembuatan aplikasi perpustakaan "BookCamps" bukan hanya menghasilkan produk yang memenuhi tujuan awal, tetapi juga memberikan pengalaman yang berharga dalam pengembangan perangkat lunak dan manajemen proyek. Harapan saya aplikasi ini akan memberikan dampak positif dalam mendukung kegiatan belajarmengajar dan penelitian di lingkungan perpustakaan. Terima kasih kepada semua pihak yang terlibat dalam kesuksesan proyek ini.

DAFTAR PUSTAKA

Link GitHub: