# PROPOSAL TUGAS BESAR

# LabLink: Sistem Manajemen Terpadu Laboratorium Riset


**Disusun Oleh:**

Rudi Firdaus 103052300064

Muh. Shafwan Faiq Rizal 103052300098

Muh. Fathier Al-Farezi 103052300029


**DS4701**

**Pemrograman Berorientasi Objek**


**PROGRAM STUDI S1 DATA SCIENCE**

**TELKOM UNIVERSITY BANDUNG**


**TAHUN 2025**

## 1. Latar Belakang

Laboratorium riset modern memiliki struktur organisasi yang kompleks. Di lab MBC, manajemen terbagi menjadi dua pilar utama: Departemen (Internal dan Eksternal) dan Divisi Keahlian (Big Data, Cyber Security, GIS, Game Tech). Asisten Riset (RA) seringkali memiliki peran ganda: mereka terikat pada satu divisi keahlian untuk pengembangan riset, sekaligus berkontribusi pada divisi operasional (misal: Divisi Internal mengurus proyek, Divisi Eksternal mengurus kegiatan seperti *company visit*).

Seorang RA dapat secara paralel mengerjakan proyek riset internal sekaligus menjadi panitia di kegiatan eksternal. Mengelola tumpang tindih tanggung jawab, progres proyek, dan partisipasi kegiatan ini secara manual sangat rumit dan rentan *human error*.

**LabLink** diusulkan sebagai solusi aplikasi 3-lapis (*3-tier*) berbasis konsol yang dirancang untuk mengatasi masalah ini. Dengan menerapkan arsitektur **Model-View-Controller (MVC)**, aplikasi ini akan memisahkan data (Model), tampilan (View), dan logika (Controller). Data akan disimpan secara permanen menggunakan **JDBC** dan alur program akan dilindungi oleh **Exception Handling** untuk menciptakan sistem manajemen yang kokoh, terstruktur, dan akurat.

## 2. Tujuan & Ruang Lingkup

**Tujuan Proyek:**

1. Menerapkan konsep inti OOP (Inheritance, Abstract Class, Interface, Polymorphism, Encapsulation) untuk memodelkan struktur lab yang kompleks.
2. Mengimplementasikan pola arsitektur MVC untuk memisahkan logika bisnis, tampilan, dan kontrol alur program.
3. Menggunakan JDBC untuk menghubungkan aplikasi ke database eksternal (misal: MySQL) agar data bersifat persisten (tidak hilang saat program ditutup).
4. Menerapkan Exception Handling untuk mengelola error saat runtime (misal: input pengguna salah, koneksi database gagal) secara profesional.

**Ruang Lingkup Proyek:**

- Aplikasi ini adalah aplikasi konsol (terminal). Fokus utama adalah pada logika *backend*.
- Data akan disimpan di database SQL (misal: MySQL/PostgreSQL).
- Sistem akan mengelola 5 entitas utama: Anggota (RA dan Dosen), Proyek (Aktivitas), Kegiatan (Eksternal), Publikasi (Hasil Riset), dan HKI (Hasil Proyek).
- Fokus *bukan* pada hierarki jabatan yang kaku, melainkan pada penugasan anggota ke berbagai aktivitas (proyek dan kegiatan).

## 3. Rancangan Fitur

- **F1: Manajemen Anggota (Research Assistant)**
  - Fitur CRUD (Create, Read, Update, Delete) untuk data RA.
  - Mencatat data penting: ID, Nama, Divisi Keahlian (String, misal: "Big Data"), Departemen (String, misal: "Internal" / "Eksternal"), dan Jabatan (String, misal: "Head of Big Data" / "Staff HRD").

- **F2: Manajemen Aktivitas Internal (Proyek & Riset)**
  - Fitur CRUD untuk Aktivitas (Proyek HKI atau Riset).
  - Mencatat data: ID Proyek, Nama Proyek, Status (misal: "Ongoing", "Completed"), Tipe Aktivitas (String, misal: 'Riset' atau 'HKI').
  - Fitur untuk menugaskan satu atau lebih RA ke dalam sebuah Proyek.
- **F3: Manajemen Kegiatan Eksternal**
  - Fitur CRUD untuk Kegiatan (misal: "Lab Exhibition", "Company Visit").
  - Mencatat data: Nama Kegiatan, Tanggal.
  - Fitur untuk menunjuk satu RA sebagai PIC (Penanggung Jawab).
  - Fitur untuk menugaskan satu atau lebih RA sebagai panitia.
- **F4: Manajemen Arsip Output (Publikasi & HKI)**
  - Fitur CRUD untuk Publikasi (hasil dari aktivitas 'Riset').
  - Fitur CRUD untuk HKI (hasil dari aktivitas 'Proyek HKI').
  - Fitur untuk menautkan Arsip (Publikasi/HKI) ke Aktivitas terkait (dari F2).
  - Fitur untuk mendaftarkan Penulis/Inventor (Anggota) ke Arsip tersebut (dari F1).
- **F5: Fitur Laporan (Cetak Info)**
  - Mencetak profil lengkap seorang RA (menampilkan Divisi, Jabatan, daftar Proyek yang diikuti, dan daftar Kegiatan yang dipanitiai).
  - Mencetak detail sebuah Proyek (termasuk daftar anggota timnya).
  - Mencetak detail sebuah Kegiatan (termasuk PIC dan daftar panitianya).
  - Mencetak detail sebuah Publikasi (termasuk penulis dan proyek terkait).
  - Mencetak detail sebuah HKI (termasuk inventor dan proyek terkait).

## 4. Rancangan Arsitektur (OOP, MVC, JDBC)

Rancangan ini mengintegrasikan semua materi kuliah (OOP, Exception, MVC, JDBC) ke dalam satu arsitektur yang utuh.

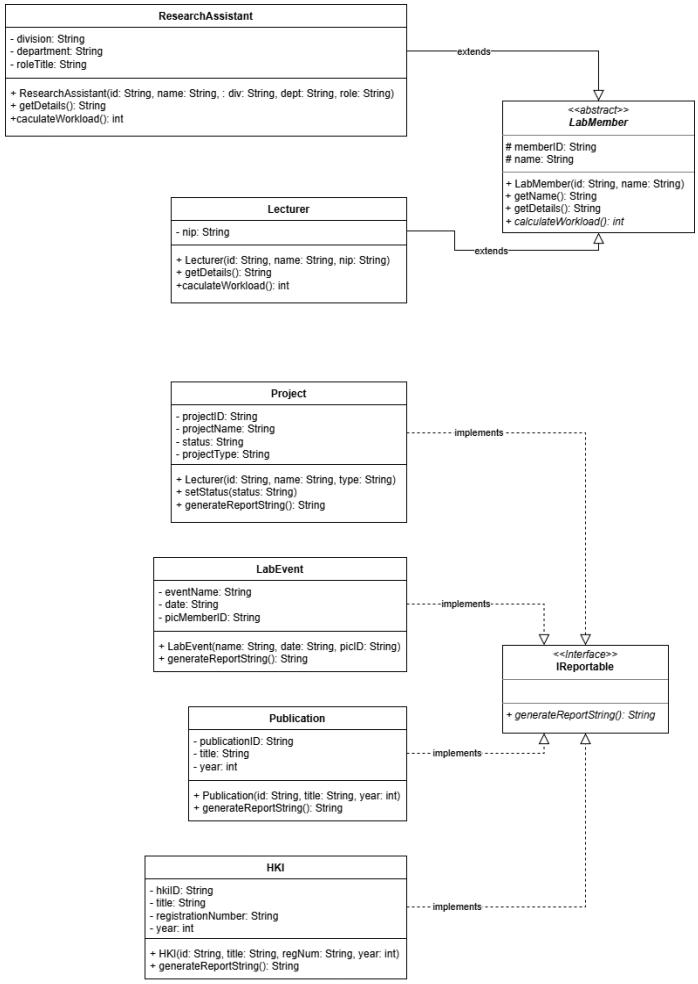
### 4.1. Pola Arsitektur: MVC + DAO

Aplikasi akan dibagi menjadi 4 "lapisan" logis dalam *package* yang berbeda:

1. **Model (Model):** Berisi *blueprint* data dan logika bisnis murni. (Misal: ResearchAssistant.java, Project.java). Tidak boleh ada Scanner atau System.out.println di sini.
2. **view (View):** Berisi kelas untuk menampilkan data ke konsol. (Misal: LabView.java). Hanya berisi System.out.println dan System.out.print.
3. **controller (Controller):** Berisi alur logika aplikasi. (Misal: LabController.java). Berisi Scanner, try-catch (Exception Handling), dan switch-case menu.
4. **DAO (Data Access Object):** Berisi logika koneksi JDBC ke database. (Misal:MemberDAO.java). Berisi semua *query* SQL (SELECT, INSERT, UPDATE, DELETE).

### 4.2. Class Diagram (Fokus pada Layer model)

Diagram ini menunjukkan *blueprint* data yang akan digunakan di seluruh aplikasi.




**Catatan Penting:**

- Jabatan (Head of Big Data, Staff HRD) disimpan sebagai String roleTitle di ResearchAssistant. Ini jauh lebih *feasible* dan fleksibel daripada membuat hierarki *class* yang kaku untuk setiap jabatan.
- Relasi *Many-to-Many* (RA ke Proyek, RA ke Kegiatan) akan dikelola oleh Database (JDBC) menggunakan tabel perantara (junction table), bukan dengan ArrayList di dalam *class* model. Ini adalah cara implementasi yang benar untuk data persisten.

## 4.3. Penanganan Error (Exception Handling)

Aplikasi akan mengelola error runtime secara terstruktur untuk mencegah *crash*:

- **InputMismatchException:** Akan ditangani di *layer Controller* menggunakan try-catch untuk memastikan input numerik dari pengguna valid.
- **SQLException:** Akan ditangani di *layer DAO* untuk menangkap error koneksi atau *query* ke database (misal: database mati, *query* gagal).
- **(Opsional) DataNotFoundException:** *Exception* kustom yang bisa di-*throw* oleh DAO jika data yang dicari (misal: findMemberById) tidak ditemukan.

### 4.4. Penjelasan Pemenuhan Syarat OOP

- **Encapsulation:** Semua atribut *class* model (expertDivision, projectName, dll.) akan bersifat private atau protected.
- **Inheritance (Wajib):** ResearchAssistant dan Lecturer akan extends *abstract superclass* _LabMember_.
- **Abstract Class (Wajib):** _LabMember_ adalah abstract class yang menyediakan atribut umum (memberID, name) dan *abstract method* (_calculateWorkload()_).
- **Interface (Wajib):** IReportable adalah interface yang mewajibkan Project, LabEvent, **Publication**, dan **HKI** untuk memiliki method generateReportString().
- **Polymorphism:** _calculateWorkload() akan di-override secara berbeda oleh ResearchAssistant dan Lecturer. Kita juga bisa membuat List<IReportable> untuk menyimpan Project, LabEvent, Publication, dan HKI secara bersamaan.

## 5. Rancangan Fitur dan Pembagian Tugas (Berbasis MVC)

Pembagian tugas disesuaikan dengan arsitektur MVC & DAO.

**Anggota 1: Rudi Firdaus (PIC: Layer Model & DAO Anggota)**

- **Tugas Utama:** Mengimplementasikan fondasi data Anggota (Model), logika database-nya (DAO), dan logika *controller* untuk fitur F1.
- **Class yang Dibuat & Diprogram:**
  - **Model:** _LabMember_.java (Abstract), ResearchAssistant.java, Lecturer.java.
  - **DAO:** MemberDAO.java (Logika JDBC untuk tb_member dan relasi terkait Dosen).
  - **Controller (Logika):** Mengimplementasikan method-method di LabController yang khusus menangani alur Fitur F1 (misal: private void handleAddMember(), private void handleUpdateMember(), dll.).
- **Fitur:** Bertanggung jawab penuh atas **(F1) Manajemen Anggota**.

**Anggota 2: Muh. Shafwan Faiq Rizal (PIC: Layer Controller & Logika Proyek)**

- **Tugas Utama:** Menjadi "otak" aplikasi (Controller Utama) dan mengelola logika Aktivitas Internal (Proyek & HKI).
- **Class yang Dibuat & Diprogram:**
  - **Controller:** LabController.java (Berisi Scanner, switch-case menu, *main loop* run(), dan **Exception Handling** try-catch untuk input menu).
  - **Model:** Project.java, HKI.java, dan IReportable.java (Interface).
  - **DAO:** ProjectDAO.java (Logika JDBC tb_project), HKIDAO.java (Logika JDBC tb_hki), dan DAO untuk tabel relasi terkait (misal: RelasiProjectMemberDAO.java).
- **Fitur:** Mengimplementasikan alur untuk (F2) Manajemen Aktivitas Internal dan (F4) Manajemen Arsip Output.

**Anggota 3: Muh. Fathier Al-Farezi (PIC: Layer View & Logika Kegiatan)**

- **Tugas Utama:** Bertanggung jawab atas semua *output* ke pengguna (View), logika Kegiatan Eksternal, dan logika untuk menghasilkan Laporan.
- **Class yang Dibuat & Diprogram:**

    - **View:** LabView.java (Berisi *semua* method System.out.println untuk menu, laporan, pesan error, *prompt* input, dll.).
    - **Model:** LabEvent.java dan Publication.java.
    - **DAO:** EventDAO.java (Logika JDBC tb_event), PublicationDAO.java (Logika JDBC tb_publication), dan DAO relasi terkait (misal: RelasiEventMemberDAO.java, RelasiPublicationAuthorDAO.java).
- **Fitur:** Mengimplementasikan **(F3) Manajemen Kegiatan Eksternal** dan **(F5) Fitur Laporan** (dengan memanggil method-method DAO yang relevan dan mencetaknya menggunakan LabView).

## 6. Penutup

Proposal ini secara akurat memodelkan struktur organisasi lab yang kompleks dalam desain OOP yang *feasible*. Dengan mengadopsi arsitektur MVC dan DAO sejak awal, proyek ini tidak hanya memenuhi semua persyaratan mata kuliah, tetapi juga membangun fondasi yang kokoh, terstruktur, dan siap untuk pengembangan di masa depan.