**Software Design Document**

**University Project Display Platform**



**Submitted by**

Iqra Fayyyaz

**Submitted to**

Mr. Zaman Ali

**Department of Information Technology**

**Faculty of Computing**

**The Islamia University of Bahawalpur**

**Summary**

This Software Design Document (SDD) outlines the technical architecture, components, and functionalities of the University Project Repository system. The document provides a detailed explanation of the system's architecture, following the Model-View-Template (MVT) pattern implemented using the Django framework. It includes the proposed database schema, defining entities such as users, projects, and evaluations, and explains their relationships. The user interface design focuses on simplicity, accessibility, and responsiveness, catering to students, tutors, and administrators with tailored functionalities.

This repository aims to streamline project management and foster academic collaboration by creating a centralized, efficient, and transparent platform for the university. The SDD serves as a blueprint for the development process, ensuring alignment with the project goals and technical requirements.

**Table of Contents**

[**1.** **Application Architecture** 4](#_Toc188405531)

[**1.1.** **MVT Architecture Components** 4](#_Toc188405532)

[**1.2.** **Controllers** 4](#_Toc188405533)

[**1.3.** **External APIs and Libraries** 5](#_Toc188405534)

[**1.4.** **UML Diagrams** 5](#_Toc188405535)

[**2.** **Data Model Schema** 8](#_Toc188405536)

[**2.1.** **Entities and their Attributes** 8](#_Toc188405537)

[**2.2.** **ER Diagram** 9](#_Toc188405538)

[**3.** **User Interface** 11](#_Toc188405539)

[**3.1.** **Webpages** 11](#_Toc188405540)

[**4.** **Conclusion** 12](#_Toc188405541)

1. **Application Architecture**

The **University Project Display** application will be developed using the **Django** framework, which follows the **Model-View-Template (MVT)** architectural pattern. Below is a detailed explanation of the components and their roles in the system:

* 1. **MVT Architecture Components**
     1. **Models:**

For this project, the following models are proposed:

* User: To manage student and tutor accounts with roles and permissions.
* Project: To store project details, such as title, description, file uploads, linked repositories, submission dates, and plagiarism reports.
* Feedback: To handle tutor reviews, comments, and grades.
* PlagiarismReport: To store results from the Turnitin API integration.
  + 1. **Views:**  
       Django views will manage the application's workflows, such as uploading projects, grading submissions, and displaying feedback.
    2. **Templates**:  
       Templates handle the presentation layer of the application. These files will use Django's template language to dynamically render HTML content for features like project listings, submission forms, feedback views, and grade displays.
  1. **Controllers**

Below is the list of the main views (controllers) and their respective functions:

* **HomeView**:
  + index(request)
* **ProjectView**:
  + submit\_project, view\_project, list\_projects
* **FeedbackView**:
  + add\_feedback, view\_feedback
* **TurnitinView**:
  + check\_plagiarism, view\_plagiarism\_report
* **AuthenticationView**:
  + login, logout, register
  1. **External APIs and Libraries**
     1. **Turnitin API:**

The Turnitin API will be used to check the originality of student project submissions. It works by sending project files to Turnitin’s servers, where they are checked against a large database of academic resources and online content. The API will return a similarity score and a detailed report, which will be saved in the PlagiarismReport model.

* + 1. **Django Libraries:**

django.contrib.auth: Used for user authentication and authorization.

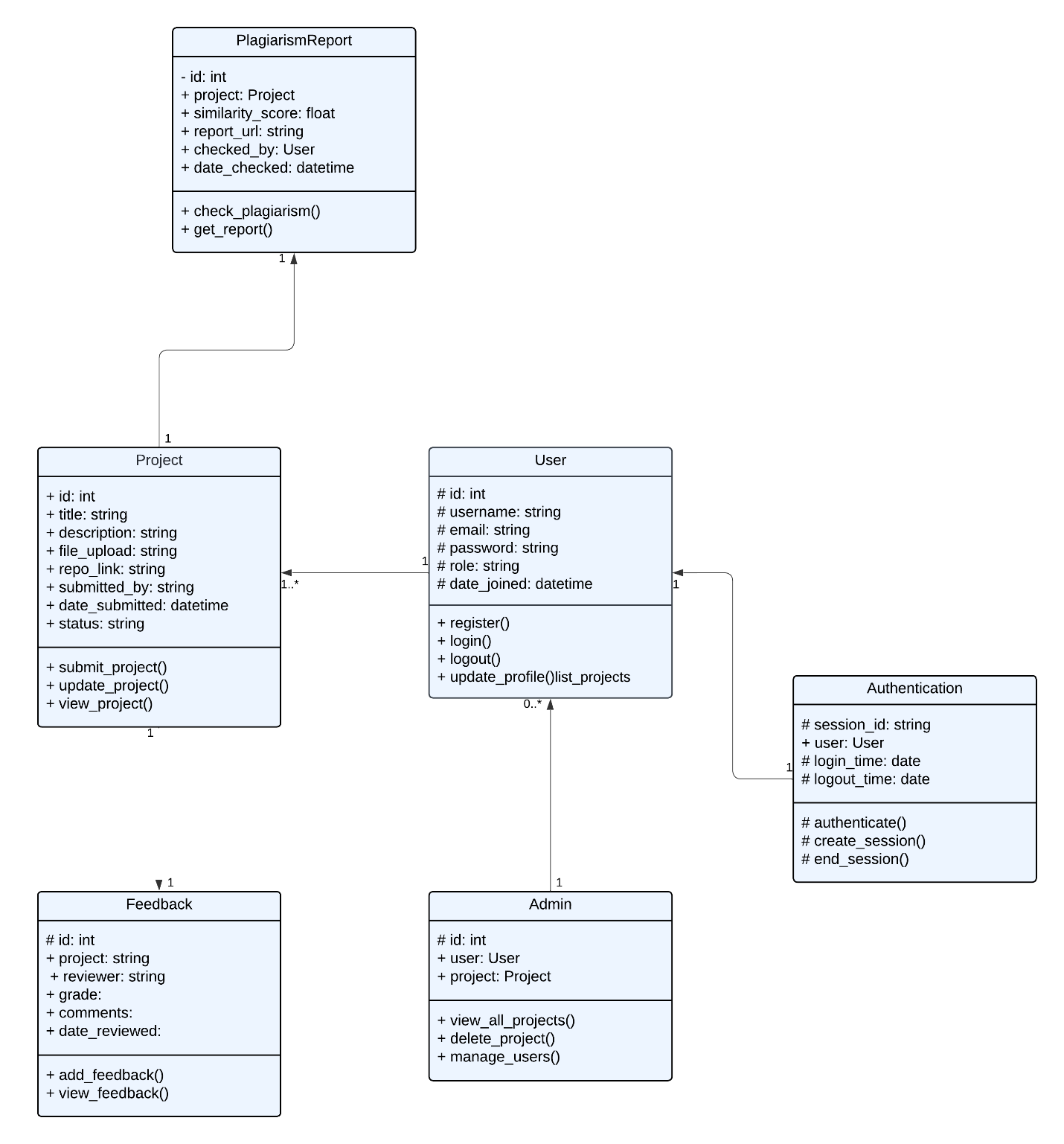
django.db.models: Used for defining and interacting with database models.

django.forms: Used to handle form submissions for project uploads and feedback.

* + 1. **SQLite:**

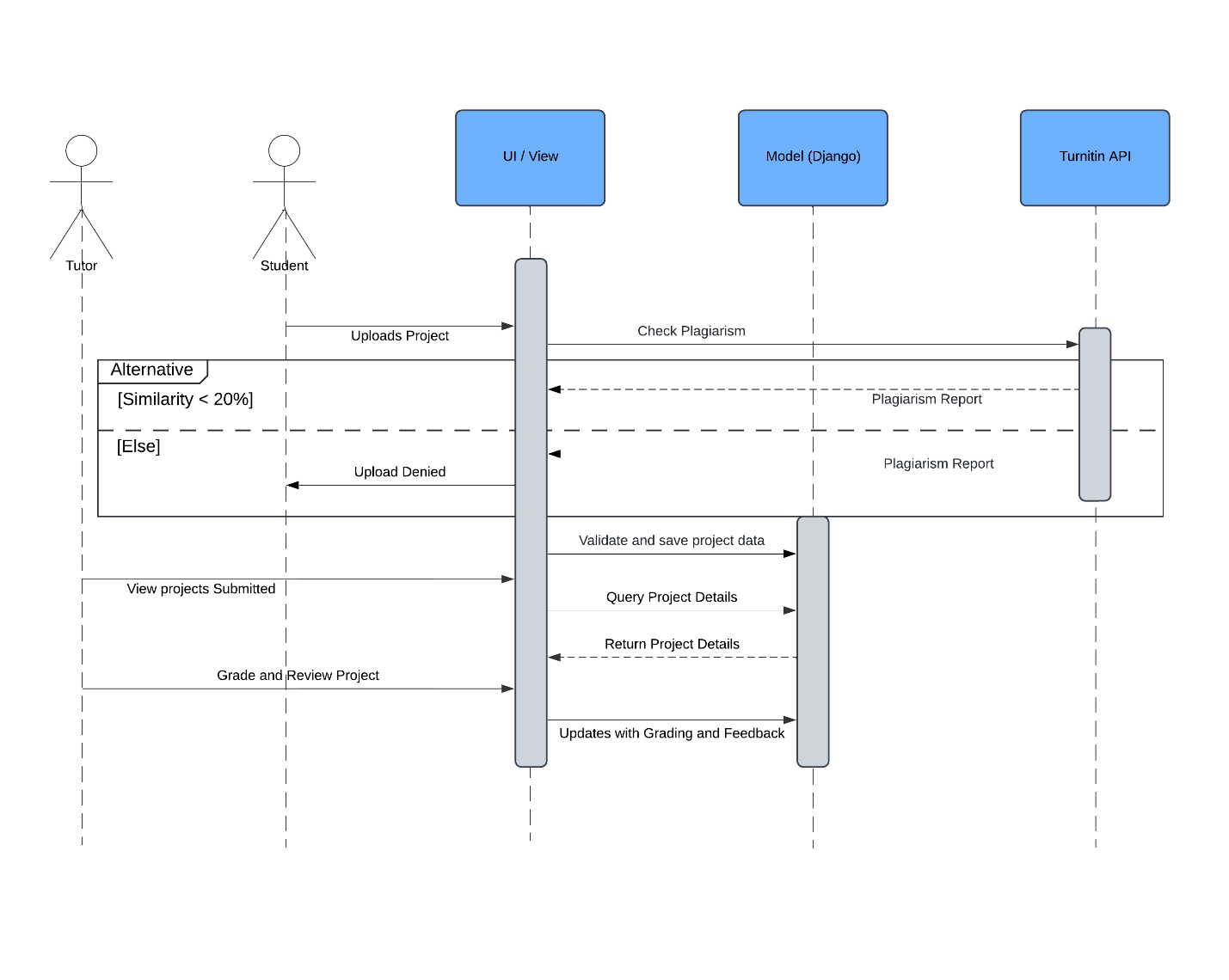
SQLite will serve as the relational database, providing robust support for storing user data, project files, and feedback.

* 1. **UML Diagrams**
     1. **Class Diagram**:



**Diagram Link:** <https://lucid.app/lucidchart/b14b867c-1d87-41d9-9b72-d7c1a16242c0/edit?viewport_loc=1539%2C-239%2C1764%2C723%2CHWEp-vi-RSFO&invitationId=inv_dbba3faf-7d04-4f94-aa66-b3dfa5b2c0e6>

* + 1. **Sequence Diagram**:



**Diagram Link:** https://lucid.app/lucidchart/b14b867c-1d87-41d9-9b72-d7c1a16242c0/edit?viewport\_loc=1539%2C-239%2C1764%2C723%2CHWEp-vi-RSFO&invitationId=inv\_dbba3faf-7d04-4f94-aa66-b3dfa5b2c0e6

1. **Data Model Schema**

This schema includes explanations of entities, their attributes, relationships, and an **ER diagram structure** that maps out the system.

* 1. **Entities and their Attributes**
     1. **User**

The user model will store information about all users (students, tutors and admin). Following are its attributes:

* User\_id (Primary Key)
* Name
* Email
* Password
* Created\_at
* Updated\_at
  + 1. **Project**

Project entitywill store information about student projects. Following are its attributes:

* Project\_id (Primary Key)
* Title
* Description
* Repo\_link
* File\_path
* Similarity\_score
* Status
* Student\_id (Foreign Key)
* Created\_at
* Updated\_at
  + 1. **Grade**

Grade entity will store grading and review information provided by tutors. Following are its attributes:

* grade\_id (Primary Key)
* grade
* review
* project\_id (Foreign Key)
* tutor\_id (Foreign Key)
* created\_at
  + 1. **Comment**

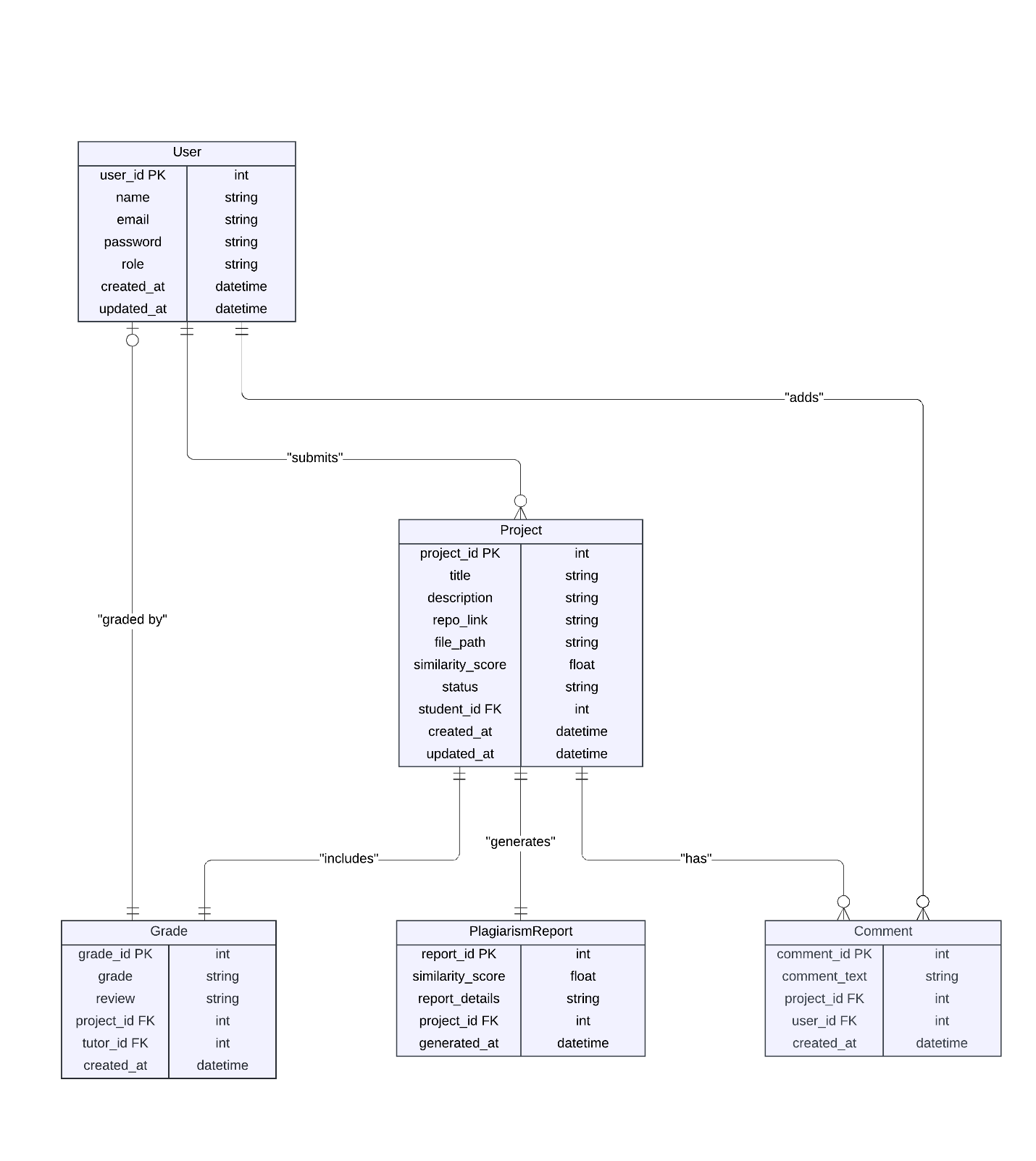
Comment entity will store comments made by tutors or admins on a project. Following are its attributes:

* comment\_id (Primary Key)
* comment\_text
* project\_id (Foreign Key)
* user\_id (Foreign Key)
* created\_at
  + 1. **PlagiarismReport**

PlagiarismReport entity will store detailed plagiarism results for projects. Following are its attributes:

* report\_id (Primary Key)
* similarity\_score
* report\_details
* project\_id (Foreign Key)
* generated\_at

* 1. **ER Diagram**



**Diagram Link:** <https://lucid.app/lucidchart/b14b867c-1d87-41d9-9b72-d7c1a16242c0/edit?viewport_loc=1539%2C-239%2C1764%2C723%2CHWEp-vi-RSFO&invitationId=inv_dbba3faf-7d04-4f94-aa66-b3dfa5b2c0e6>

1. **User Interface**

This section will provide descriptions about our interfaces that we plan to use when creating our University Project Display web application. Each description will explain the responsibilities of the components used and how it gels with other parts of our web application. Since we are using Django to develop our web application, it will automatically keep the DOM in sync with our data which makes things beneficial towards us.

* 1. **Webpages**

Once a user logs into the web application, they will be able to access a Home Tab, My Project Tab, Project Submission Tab and a Logout Tab.

* + 1. **Home Tab:**

The Home Tab will allow the user to navigate list of projects uploaded by other students, use search bar to search the projects using keywords and apply filters.

* + 1. **My Project Tab:**

My Project Tab will navigate the user to the status of his project. Here, the user can view his/her Grades, feedback from the supervisor and plagiarism score. Our goal is to help students to remain up to date about their projects and a place where they can interact with their supervisor through the Comment section.

* + 1. **Project Submission Tab:**

This tab will basically display the Form the student will use to submit the project. There will be input fields for uploading SRS and SDD documents and uploading the link to the repo where the user has stored his/her code file. There will be a submit at the end of the form. After the form submission, the user will view the similarity score retrieved by application from Turnitin API. If the score is less than 20%, the data will be successfully updated in the database else, the student will be informed with an alert message indicating that the project submission was not successful and will prompt the user to submit the project again.

* + 1. **Logout tab:**

This tab will show the user details and a logout button. Once user clicks on it, the user will be automatically logout from the application.

1. **Conclusion**

To conclude, many students are struggling with the submission of their projects to their respective tutors on time as sometimes the tutors are not available and sometimes students are unable to come to university to submit their projects on time. That is why it is our goal to provide students and supervisors a central hub where they can interact with each other and can check project’s status.

This document highlights the core design philosophies, database schema, UML diagrams, and user interface specifications to deliver a seamless project management and showcase platform. This system will address the inefficiencies of traditional project handling methods, setting a new standard for academic excellence and project transparency within the university.