SURANA COLLEGE

Department of Computer Science

V SEMESTER PROJECT 2020-21

INDEX

ACKNOWLEDGEMENT

The development of the project has been quite challenging and is it has been sincere effort to bring this project to its completion. The successful completion of this project wouldn't have been possible without the continuous support and encouragement of many. We make use of this opportunity to thank all those who helped us in bringing out this project. We would like to express our gratitude to Mr. Srinivas.A for granting us permission to carry out this project work at college. We sincerely thank our guide Mrs. Vidya.A and all teachers for their constant help for our project during the course of this pandemic. Our project wouldn't have made it this far if not for the online courses and tutorials from which we have learnt so much. We also thank all our friends for their valuable suggestions and encouragement.

Kaushik. B

Manoj. S

Project Synopsis

1. Problem Statement

Managing sharing of study notes may seem tricky, but this is part of user service system application support direct contact with user.

2. Objective

The objective of this project entitled “Study Notes Sharing Site” is to provide a user friendly and easily understandable GUI to users to easily upload subject notes. User can also update the notes details and delete the notes uploaded by him. This project also helps in administrative work such as management of notes and users. Admin can also monitor notes uploaded by user and able to accept or reject notes file.

3. Existing System

Only registered users can access the system. The user cannot view or add any information in the system when the server is down. The students can view only the data which are uploaded by then and the administrator.

4. Proposed system

The server allows the faculty to upload data and the students can opt to download the required documents through the site. The admin can control the notes that is uploaded by the student or the administrator.

The proposed system is worked on the server and all the files and documents are uploaded in the server. The students just need to register themselves and can surf all every detail related to the document very easily. Proposed system saves a lot of time to both students and administrator.

Feasibility Report

1. Technical Feasibility Study

The system is technically feasible because here we are using Python, Django and SQLite3 for database, which is free for commercial purposes. This system is also feasible for the users as it can satisfy user requirements and maintain their records in an organized way.

2. Operational Feasibility Study

The system is operationally feasible as it covers most functions on notes sharing and is dependable, as it is easy to use.

3. Economic Feasibility Study

The system is economically feasible as Python, Django and SQLite3 is open source and free for all. For users it is feasible as it is free to use.

Modules

Admin modules

- Manage (View/Delete) Users
- Manage (View/Delete) Notes
- Accept or Reject Notes
- Change Password

Student modules

- Upload notes
- Search notes via stream, title and subject
- Download notes
- Delete and Update notes
- Change password

Software Requirements Specification

1. Purpose

This E-notes Point system project document completely describes what the system should do without describing how the software will do it. The basic goal of requirement phase is to produce the SRS, which describes the complete external behaviour of the system.

2. Scope

The project has a wide scope, as it is not intended to a particular organization. This project is going to develop generic software, which can be applied by any business' organization. Moreover, it provides facility to its users. Also, the software is going to provide a huge amount of summary data.

3. Developer's Responsibility

The developer is responsible for:

a) Developing the system.

b) Installing the software on the client's hardware.

c) Conducting user training that E-notes point report might be needed for using the system.

d) Maintaining the system for a period of two years after installation.

Product Description

In this project "**E-notes Point Project**" student and administrator can upload/download notes. Through the notes sharing process, users can upload, download and also view the study notes uploaded by others. User can also edit the notes uploaded by him. In this project two major modules that have been provided are administrator module and user module.

1. Goals

- All users can upload and share study notes with other users.

- Users can select the stream (branch) at the time of upload.
- User can upload notes in the pdf, doc, ppt, zip, txt etc. file format.
- User can search notes via branch, subject and title.
- Administrator or management can manage the users and notes.
- Admin can accept or reject the notes uploaded request. Notes can be published only after accepted by admin.

2. General Requirements

During the Requirement Analysis Phase, the development team analyzes the requirements to be fulfilled by the E-notes Point website and identifies the probable approach for meeting these requirements. To identify the requirements, we decided to study the existing process of uploading and storing the details of opening account and notes uploading.

Finally, it was identified that the notes sharing project should:

- Enable new users to fill the signup form and should be able to login.
- Enable to update the profile later if needed.
- Can upload, view and download notes that has been uploaded by them as well as others.
- Admin can also upload, view, delete the notes. And can accept and reject the notes uploaded by users and delete the user as well.
- Admin can change password if necessary.
- The webpage should be user friendly and well designed.
- It should be able to handle exceptions and run time errors.

Usage Scenario

The E-notes Point project provides usage scenario for the software. It has organized information collected during requirements elicitation into use-cases.

1. User Profiles

   There will be two levels of users:

   - Administrator level
   - User level

2. Use Cases

   - Administrator level

     The admin can upload, view, delete the notes and accept or reject notes.

   - User level

     The user can upload, delete notes and can also view the notes uploaded by others.

Data objects and description:

During the requirement analysis phase, the development team examines the system. After examining al the process and feasibility, the following points were considered while designing the database:

- It should store the registered information of both users and administrator.
- It should store the uploaded notes and details of the file.

Functional Requirements

- Edit profile
- Upload notes
- View notes
- Change password
- Manage users
- Manage notes
- Download notes

- Administrator

Non-Functional Requirements

Usability

The user interface of the system will be user friendly so it will be easy for the user to use the system.

Reliability

The system would be easy to maintain and able to carry further extensions occurred.

Performance

The system should provide the services in considerable time interval.

Security

The information provided by the user should authenticated which protect the system from external attack and spamming.

Data Dictionary

User Register Table

COLUMN NAME	**DATATYPE**	**PRIMARY KEY**	**NOT NULL**
User id	integer	Yes	Yes
Contact	Varchar(10)	No	No
Branch	Varchar(30)	No	Yes
Role	Varchar(15)	No	Yes

Query

CREATE TABLE "user_signup" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "contact" varchar(10) NULL, "branch" varchar(30) NOT NULL, "role" varchar(15) NOT NULL, "user_id" integer NOT NULL REFERENCES "auth_user" ("id") DEFERRABLE INITIALLY DEFERRED)

Notes Table

COLUMN NAME	DATATYPE	PRIMARY KEY	NOT NULL
User id	Integer	Yes	Yes
Uploadingdate	Varchar(30)	No	Yes
Branch	Varchar(30)	No	Yes
Subject	Varchar(30)	No	Yes
Notesfile	Varchar(100)	No	Yes
Filetype	Varchar(15)	No	Yes
Status	Varchar(15)	No	Yes
description	Varchar(200)	No	No

Query

CREATE TABLE "notes_notes" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "uploadingdate" varchar(30) NOT NULL, "branch" varchar(30) NOT NULL, "subject" varchar(30) NOT NULL, "notesfile" varchar(100) NULL, "filetype" varchar(30) NOT NULL, "status" varchar(15) NOT NULL, "user_id" integer NOT NULL REFERENCES "auth_user" ("id") DEFERRABLE INITIALLY DEFERRED, "description" varchar(200) NULL)

Admin table

COLUMN NAME	DATATYPE	PRIMARY KEY	NOT NULL
adminname	Varchar(30)	yes	yes
password	Varchar(30)	no	yes

Query

CREATE TABLE "admin" ("adminname" varchar(30) NOT NULL PRIMARY KEY AUTOINCREMENT, "password" varchar(30) NOT NULL)

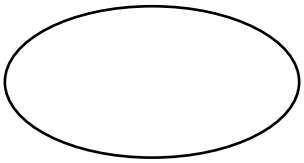
Data Flow Diagram (DFD)

A data flow diagram (DFD) is a graphical representation of the “flow” of data through and information system. A data flow diagram can be used for visualisation of data processing. It is a common practice for designer to draw context level DFD first which was the interaction between the system and outside entities. This context level DFD yes then exploded to show more details of the system being modelled.

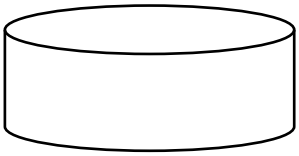
A DFD represents flow of data through a system. Data flow diagrams are commonly used in urine problem analysis. It views a system as a function that transforms the input into desired output. A DFD shows moment of data through the different transformations or processes in the system.

Data flow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately has an effect upon the structure of the whole system from order to dispatch to restock any system is developed can be determined through a data flow diagram. The appropriate register saved in database and maintained by appropriate authorities.

Data flow diagram notation



Function

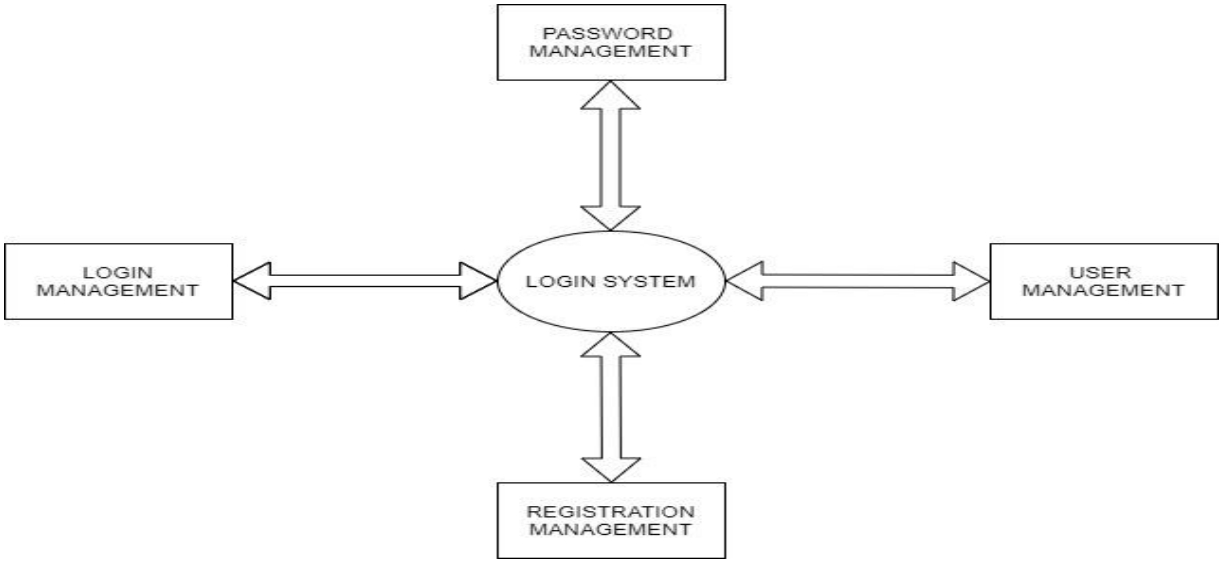


database

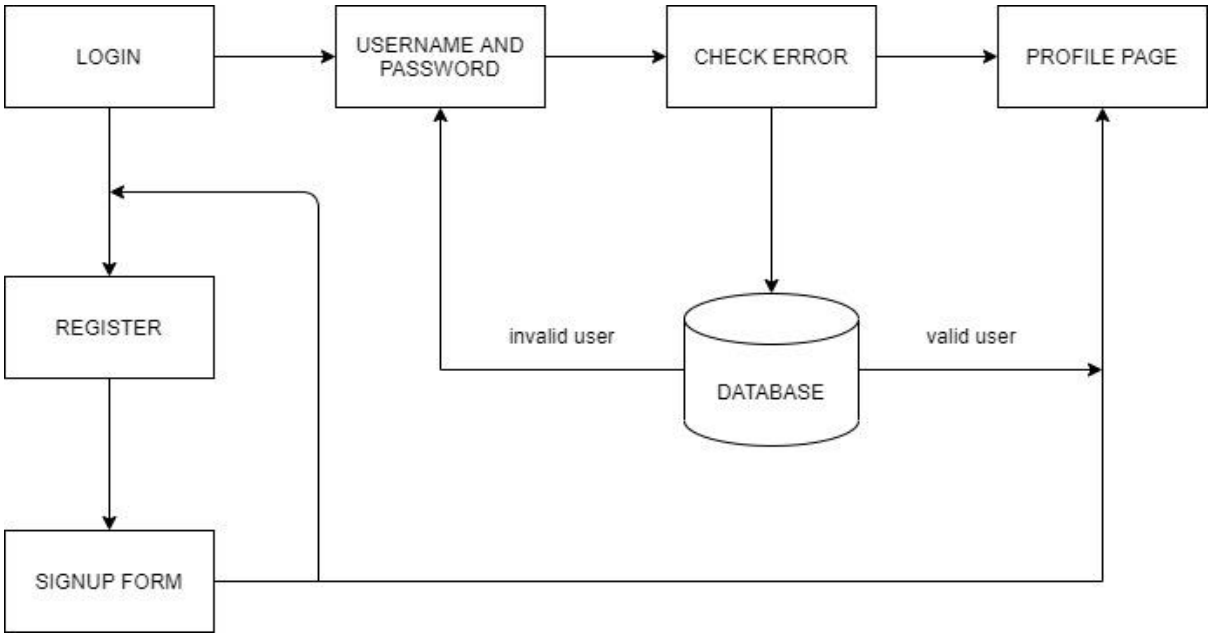
input/output

flow

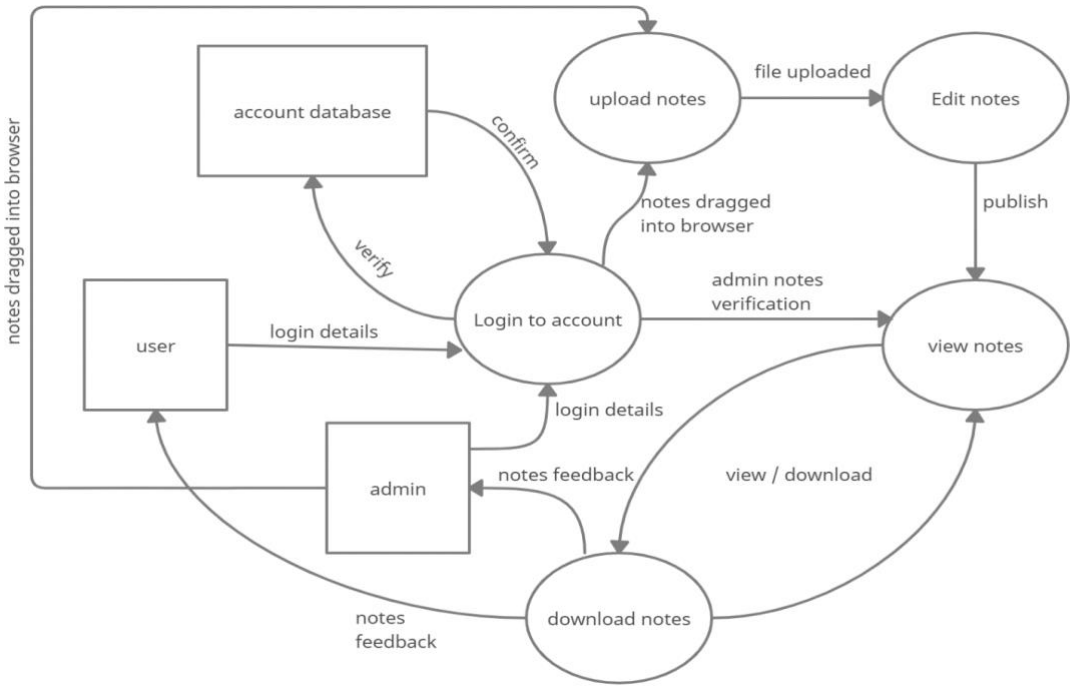
Level 0 DFD



LEVEL 1 DFD



LEVEL 2 DFD



Entity relationship diagram (ER-diagram)

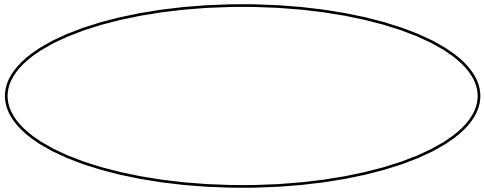
Entity relationship diagram is a specialised graphic that illustrates the interrelationship between entities in a database. ER diagrams often use symbols to represent 3 different types of information. Boxes are commonly used to represent entities. Diamonds are normally used to represent relationships and ovals are used to represent attributes.

An entity relationship model in software engineering is an abstract and conceptual representation of data. Entity relationship modelling is a relation schema database modelling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion.

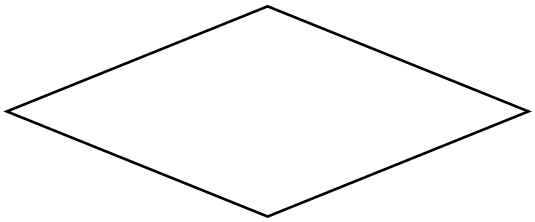
Symbols used in this E-R diagram:

Entity: Entity is a “thing” in the real world with an independent existence. An entity may be an object with a physical existence such as person, car or employee. Entity symbol is as follows

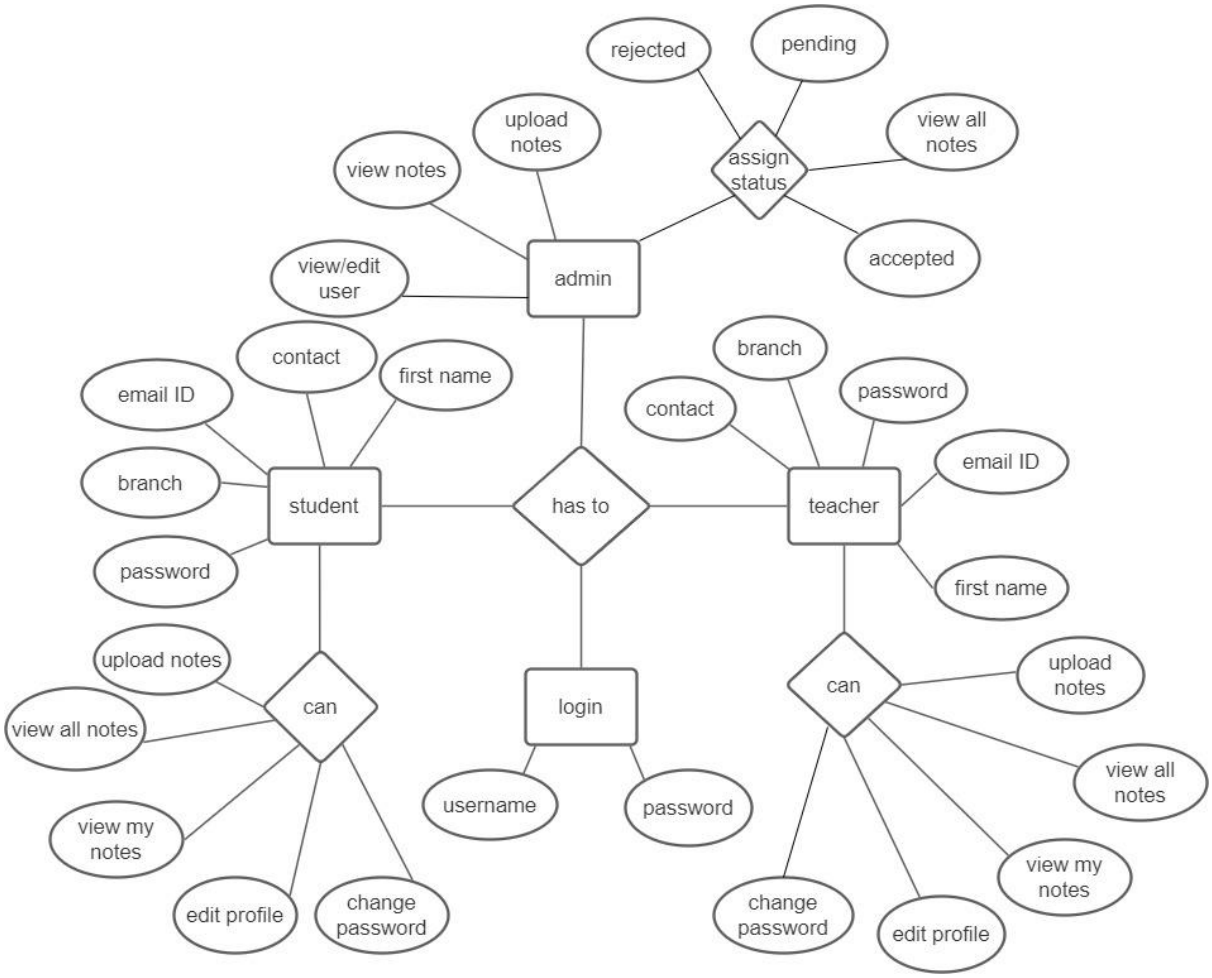
Attribute: Attribute is a particular property that describes the entity. Attribute symbol is



Relationship: Relationship will be several implicit relationships among various entity types whenever an attribute of one entity refers to another entity type some relationship exits. Relationship symbol is



ER diagram:


rejected
pending
upload notes
view all notes
view notes
assign status
accepted
view/edit user
admin
contact
first name
email ID
branch
password
student
has to
branch
password
contact
email ID
teacher
first name
upload notes
can
view all notes
view my notes
edit profile
change password
login
username
password
can
upload notes
view all notes
view my notes
change password
edit profile