**Author:**

Paul M Kallarackal

Roll No: 21f1003005

Email\_id: 21f1003005@student.onlinedegree.iitm.ac.in

Description:

2nd Year B.E C.S.E Student with a basic understanding of Python and Java. Is the Google Cloud Computing Foundations Program Mentor on Campus. Is an avid reader and chess enthusiast.

**Description of Problem Statement**:

Implementation of a FlashCard application with login facility, with permissions to update,view,delete and make cards as well as decks. The application should also allow a review feature to measure progress of learning. The styling, features, validation and API creation are optional.

**Technologies Used:**

Flask : For the Application in general, the routes and the server.

Flask-SQLAlchemy: For the Database Models

Jinja2: for rendering the HTML dynamically

HTML,CSS,Javascript: For basic frontend functionality styling and structure.

**Database Schema**:

User table: Used to Store username and password for verification

"User" ("userid" INTEGER,

"username"-TEXT NOT NULL UNIQUE,

"password"-INTEGER NOT NULL,

PRIMARY KEY("userid" AUTOINCREMENT))

HTML table:

Master HTML Database for quick importing of HTML attributes, which is what my app is for primarily. It is preloaded with 131 HTML Attributes. Cannot be modified by any user.

"HTML" ("s\_no" INTEGER, PRIMARY KEY("s\_no" AUTOINCREMENT)

"name" TEXT,

"description"TEXT,

)

Cards table:

Table to store all users cards. Connected to User table through Foreign Key username. Users can only modify their cards.

"Cards" ("c\_no" INTEGER, PRIMARY KEY("c\_no" AUTOINCREMENT))

"c\_name" TEXT NOT NULL,

"c\_description" INTEGER NOT NULL,

"username" TEXT NOT NULL, FOREIGN KEY("username") REFERENCES "User"("username"),

Decks table:

Table to store all user decks. Connected to User table through Foreign Key username. Users can only modify their Decks.

"Decks" ("deck\_id" INTEGER, PRIMARY KEY("deck\_id" AUTOINCREMENT),

"username" TEXT NOT NULL, FOREIGN KEY("username") REFERENCES "User"("username"))

"deck\_name" TEXT NOT NULL,

"cardlist" TEXT NOT NULL,

"time" TEXT,

"scorelist" TEXT,)

**API Design**

Not Created, CRUD operations are done solely through controllers.

**Architecture and Features**:

Architecture: static folder with 2 css files. 18 html templates with self-explanatory names. Controllers with self explanatory routes like /username/action/extra\_info\_for\_action. One main.py file where the Database Models and the Controllers are clearly distinct. Comment lines are present for every block of code.

Default features: CRUD operations on Deck/Cards, Scoring operations on Review – done through controllers. Login operations through forms.

Extra features:

Validation done either through custom HTML pages or form requirements, most error Cases covered.

Importing cards from master deck for ease of access.(since my app is designed as a HTML Attribute Flashcard App, but has the feature to have other types of cards as well.

**Video Link:**