A close-up of a logo

AI-generated content may be incorrect.School of Computer Science

**COMP9120 Database Management Systems   
2025 Semester 2**

# Assignment 2: “Sydney Music System” — Python DB-API Application

Notice:   
*Our goal is to create the best possible assignment for you. However, mistakes can happen. If we identify an error, we’ll announce it on Ed and record it here. We appreciate your understanding and ask that you check your email regularly for Ed announcements.*

*Update Log – (last updated: 06 Oct 2025 2:40pm)*

* *01 Oct 2025: in list\_tracks.html line 49 address the issue of updating Track.Duration only accepted the one decimal place. Users can now enter numbers with any number of decimal places. Thanks to* [Yu-Chuan Shen](https://edstem.org/au/courses/24438/discussion/2968380)
* *01 Oct 2025 :: on page 9 – Update Tracks: further clarification: “Invalid logins will be rejected. NULL is also valid, as some songs may not have a singer and/or composer.”*
* *06 Oct 2025 :: in routes.py list\_tracks() – Fixed the issue in the routing related to searches with an empty string. (this shouldn’t impact the main logic of find\_tracks()). Thanks to all posts on Ed! e.g.* [*post#198*](https://edstem.org/au/courses/24438/discussion/2971992)

## **Group Assignment (12.5 %)**

**Introduction**

In this assignment, you will be working on building a user interface for the Sydney Music database as described in Assignment 1. More specifically, your task is to implement an interface, referred to as the Sydney Music system, through which a user interacts to access and manipulate the Sydney Music database. Your main task in this assignment is to handle requests for reads and writes to the database using your User Interface (UI). We first describe the main features that the Sydney Music system must include from a UI perspective and then discuss where your database code needs to be implemented.

Important things to note:

* This document is written for Python.
* This is a group assignment for teams of 3 people per group. You must be enrolled in an assignment group on Canvas.
* Please also keep an eye on your email, Ed and the first page of this live assignment for any related announcements or posts.

**Submission Details**

The submission of your solution is due in **Week 12 on 02 November 2025 at 23:59.** You must submit the items for submission (detailed below) via Canvas.

**Items for submission**

Please submit your solution to Assignment 2 in the ’Assignment’ section of the unit’s Canvas site by the deadline, including EXACTLY THREE files:

1. An assignment coversheet as a PDF document (.pdf file suffix) which is available for download from [**this link**](https://unisyd-my.sharepoint.com/:w:/g/personal/armin_chitizadeh_sydney_edu_au/EfG8BWx1MLZEmdHqCAjC1UABacIafPRaBYHyYgUMX2g1CQ?e=evcKMn) on Canvas
2. A SQL file (**SAGschema.sql**) containing the SQL statements necessary to generate the database schema and sample data. This ***must contain the supplied original schema and insert SQL statements***, **and your additional stored procedures (functions)**.
3. A Python file (**database.py**) containing the Python code you have written to access the database

## Part 1: Application Pages & Behaviour

Build a minimal web app that connects to PostgreSQL via the Python DB-API (psycopg2). **All SQL must be parameterised** and **business rules enforced in SQL** (server-side constraints/functions), not in Python**.**

### **Login**

The user is defined as any authorised user with account at Sydney Music.

The first form presented to a user when starting the Sydney Music system is the Login, as shown in Figure 1. This feature requires the user to enter their username and password to be validated prior to being successfully logged in to the system. Security features such as password encryption/hashing are out of scope for this assignment. The username should be case-insensitive. Once logged in, the user is directed to the tracks list page, where they can view a list of available song track data for all artists.

A screenshot of a login screen

AI-generated content may be incorrect.

Figure - Login

### Viewing Information

#### Viewing the list of Tracks

Once a user successfully logs in, they are presented with a list of available tracks, as shown in Figure 2 below. Each track has a track id, title, duration, age restriction, singer, composer, and average rating. This list must be ordered by trackid in ascending order.

* The Singer and Composer are artists who contributed to the track. Each track can have exactly one singer and one composer. However, in some cases, this information may be missing.
* The Average rating is calculated as the mean of all ratings provided in the reviews for that track.

A screenshot of a music track

AI-generated content may be incorrect.

Figure - Viewing Tracks List

While every user can view this information, only staff members have permission to update certain details of a track, as explained in the Updating Information in the Database section.

#### Viewing the list of Reviews

All users can also view reviews that have been submitted for tracks. Each review shows the review ID, the title of the track being reviewed, the review content, the full name of the customer who wrote it, and the review date (formatted as dd-mm-yyyy). Reviews are displayed in order of the most recent firstand then by review ID, as shown in Figure 3. Customers may update their own reviews if needed, but they cannot edit reviews written by others.

A screenshot of a music review

AI-generated content may be incorrect.

Figure - Viewing Reviews

#### Viewing the list of users

In addition, staff members have special permission to view the list of all users in the database. For each user, staff can see the login, first name, last name, email, and role. The list of users is displayed ordered first by role and then by login in ascending order, as shown in Figure 4. Staff can also update certain details of a user account, as described in the Updating Information in the Database section.

A screenshot of a user account

AI-generated content may be incorrect.

Figure - Viewing Users

### Finding Tracks

A user can search through all menu items by entering a word or phrase (a ‘keyword’) in the field next to the Find button, as shown in Figure 5, and then clicking on Find, When such a keyword is specified, then the retrieved tracks and shown on the list are those that include this word or phrase in the track title. The search is case insensitive. Searching with a blank/empty keyword field will show all tracks. Any search results returned must be ordered by the average rating in descending order.

**A screenshot of a music score

AI-generated content may be incorrect.**

Figure - Finding tracks with search keyword "OF"

### **Adding New Information to the Database**

**Sydney Music allows different types of users to add new information to the database depending on their role.**

#### Staff: Creating a New User

**A user with the role Staff is able to create new user by adding a record to the Account.** **To do this, go to the User section** **and select Add User. When creating a new user, the staff member must provide a unique login username,** **the** **new user’s first name, the new user’s last name, password, and role, which must be either Customer, Staff, or Artist. An email address may also be provided at the time of account creation; however, this is optional and may be supplied later if not initially available. No matter how the login username is typed (uppercase, lowercase, or a mix), the system will automatically save and store it in lowercase. Once all required fields are entered, click Add to create the new account as shown in Figure 6. The system will confirm that the account has been successfully created.**

A screenshot of a login form

AI-generated content may be incorrect.

Figure - Adding a new user

#### **Customer: Creating a New Review**

**A user with the role Customer can share their opinions on tracks by adding reviews. To add a review, go to the Review section, and select Add Review. Each review must be associated with a valid track and must include a rating between 1 and 5. Customers are also required to provide a review date, which cannot be set in the future (from today’s date). In addition to the rating and date, customers may optionally include written content as part of their review. If customer only want to leave a rating without text, the system will still save their review. Once you complete the form,** **click Add button,** **as shown in Figure 7. Your review will then appear alongside other reviews**

A screenshot of a computer

AI-generated content may be incorrect.

Figure - Adding a review

### Updating **Information in the Database**

**Sydney Music allows different types of users to updating information to the database depending on their role.**

#### **Staff: Updating User**

**A user with the role Staff is able to update certain details of user accounts. When editing an account, staff may change the user’s first name, last name, or email. However, they cannot change the login username, role, or** **password once the account has been created. This ensures that the identity and access level of each account remain consistent. After making the changes, click Update to confirm, as shown in Figure 8. The system will update the account and display a confirmation message.**

**A screenshot of a login form

AI-generated content may be incorrect.**

Figure - Updating User details

#### **Staff: Updating Track**

**A user with the role Staff is able to update certain details of tracks stored in the database. When editing a track, they can change the title, duration, age restriction, singer, or composer, as shown in Figure 9. When updating the singer or composer, staff must enter a valid artist login that already exists in the system.** *Invalid logins will be rejected. NULL is also valid, as some songs may not have a singer and/or composer.* **Logins for singers and composers are case-insensitive—for example, entering ‘MUSICSTAR’, ‘****musicstar’, or ‘MusicStar’ will all be recognised as the same artist if that login exists. If the login provided does not match a valid artist record, the update will not be accepted.**

**A screenshot of a computer

AI-generated content may be incorrect.**

Figure - Updating Track details

#### **Customer: Updating Review**

A user with the role Customer is able to update reviews they have written. A customer may edit the rating and the content of their review, but they cannot change the review date or edit reviews created by other users, **as shown in Figure 10**. Once a review is updated, the system will automatically recalculate the track’s average rating to reflect the changes.

**A screenshot of a review

AI-generated content may be incorrect.**

Figure - Updating the rating and content of a review

**Database Interaction Code**

**The files that are needed for the Python version of the assignment are as follows:**

1. [schema.sql](https://canvas.sydney.edu.au/courses/66447/files/45792739?wrap=1)**: a file which contains SQL statements you need to run to create and initialise the Sydney Music database, before starting the application.**
2. [Assignment2\_PythonSkeleton\_UPDATED\_06\_OCT.zip](https://canvas.sydney.edu.au/courses/66447/files/45923098?wrap=1): **a zip file encapsulating the Python project for the Sydney Music System**

**To inspect the Sydney Music system code, you need to unzip the ZIP archive first, which will create a folder that includes the name Assignment2\_PythonSkeleton. If you experience any difficulties installing and exploring the project, ask your tutor or lecturer for assistance.**

**The skeleton code uses a number of Python modules to implement a simple browser-based GUI for the Sydney Music system. The main modules are the Flask framework for the GUI and the psycopg2 module for the PostgreSQL database access.**

**Similar to tutorial 7, you will need to install the Psycopg2 module and the Flask module.**

***pip install psycopg2-binary***

***pip install flask***

**The skeleton code follows the structure described below:**

* **The main program starts in the main.py file. You need to use the correct username/password details as specified in tutorial 7 and then implement the missing database access functions – including any necessary SQL code statements required – in the data layer database.py.**
* **The presentation layer is done via a simple HTML interface that can be accessed from a web browser. The corresponding page templates are located in the templates/ subdirectory, and their CSS style file is in static/css.**
* **The transition between the different GUI pages and the initialisation of the Flask framework is done in the routes.py file. It currently just invokes the pages, but there is no further business logic implemented yet.**

**You can run the code by running “python main.py”. This starts a local web server and prints out some debug messages in the terminal; the GUI can then be accessed with any web browser on the same computer via the local URL http://127.0.0.1:5001/ (If that doesn’t work, you can also try http://0.0.0.0:5002/). Please note that, to terminate the application, you will need to stop the local web server, which is running in the background.**

## Part 2: Functions Implementation

**In this assignment, you are provided with a Python skeleton project that must serve as the starting point for your assignment. Your task is to provide a complete implementation for the file database.py, as well as make any modifications necessary to the database schema (i.e., schema.sql).**

**Specifically, you need to complete the following functions:**

1. **checkLogin (for login)**
2. **list\_tracks (for viewing the list of tracks)**
3. **list\_users (for viewing the list of users)**
4. **list\_reviews (for viewing the list of reviews)**
5. **find\_tracks (for finding tracks)**
6. **add\_user** **(for creating a new user)**
7. **add\_review (for creating a new review)**
8. **update\_track (for updating track)**
9. **update\_review (for updating review)**
10. **update\_user (for updating user)**

**Note that, for each function****, the corresponding action and outcome should be implemented by issuing SQL queries to the database management system. If you directly output the result set, pre-process, manipulate and/or make changes to the input or output datasets using Python code or additional modules (libraries), i.e. without issuing SQL queries, you are considered to be cheating, and you will get penalised heavily and most likely get zero points for the assignment.**

**Note: No additional Python/Java modules or libraries should be imported.**

## Part 3: Marking Rubrics

This assignment is worth **12.5%**.

|  |  |  |
| --- | --- | --- |
|  | Novice (0.5 pt) | Proficient (1 pt) |
| checkLogin | Can correctly login the user ‘jdoe’ and validate the username and password. | All valid users can be logged in successfully, and unsuccessful user logins should be rejected. Implementation fully working with the UI. |
| list\_tracks | Correctly display all tracks. | Correctly list all tracks in the correct order. Implementation fully working with the UI. |
| list\_users | Correctly display all users. | Correctly list all users in the correct order. Implementation fully working with the UI. |
| list\_reviews | Correctly display all reviews. | Correctly list all reviews in the correct order. Implementation fully working with the UI. |
| find\_tracks | Correctly display tracks for the keyword “lo” | Correctly list tracks for all possible keywords. Implementation fully working with the UI. |
| add\_user | Can add a user to the database. | Can correctly add all valid users to the database. User entered with invalid details should be rejected. Implementation fully working with the UI. |
| add\_review | Can add a review to the database. | Can correctly add all valid reviews to the database. Review entered with invalid details should be rejected. Implementation fully working with the UI. |
| update\_track | Can update a track record. | Can correctly update details of all tracks, ensuring the updated details for a track are valid. Implementation fully working with the UI. |
| update\_review | Can update a review record. | Can correctly update details of all reviews, ensuring the updated details for a review are valid. Implementation fully working with the UI. |
| update\_user | Can update a user record. | Can correctly update details of all users, ensuring the updated details for a user are valid. Implementation fully working with the UI. |
| Stored Procedures (Function) | One or more stored procedures (functions) are correctly created in the submitted SQL file. | Two or more stored procedures (functions) are correctly created in the submitted SQL file and correctly called in two of the ten specified functions. |
| Database Practices | Minor flaws in transaction management or data processing logic. | Correctly uses database connections, transactions, and parameterized queries for all database interactions. |

**Your code will be assessed on its adherence to good database practices. Marks will be deducted for violations such as the unnecessary explicit use of transactions, performing data processing in the application code instead of within SQL queries, or failing to use parameterized queries, which exposes the application to SQL injection vulnerabilities.**

## Group member participation

If you are having difficulties with a teammate and have already tried to resolve it without success, please complete the form below as soon as possible to let me, Armin, know so I can help you address the issue. Keep in mind that unequal contribution percentages will only be accepted if you have previously reported the problem to both your tutor and the unit coordinator using this form: <https://forms.office.com/r/1ZvePBafFj>

|  |  |
| --- | --- |
| Percentage of contribution | Proportion of final grade received |
| < 5% | 0% |
| 5–10% | 20% |
| 11–15% | 40% |
| 16–20% | 50% |
| 21–24% | 60% |
| 25–28% | 80% |
| 29–30% | 90% |
| > 30% | 100% |

**Note: The above table assumes that each group will have 3 members, so, on average, around 33% contribution is expected from each member of the group. In a special case, if the group has fewer than 3 members, then the contribution percentage will be adjusted accordingly. You must justify your contribution percentage by providing a detailed explanation of your individual contribution on the assignment cover sheet mentioned before. You should also maintain a diary of your group meetings and discussions on Canvas. Furthermore, we may run random face-to-face interviews to understand and justify your contribution, if needed.**

**Team work:**

|  |  |  |
| --- | --- | --- |
|  | No Marks (0 pt) | Full Marks (1 pt) |
| Record Keeping of  Group Discussion | One or more issues reported or found with group member contribution, or with maintaining records of group meetings and discussions regularly on Canvas. | No issue reported or found with group member participation. All group members participate and regularly maintain a diary of group meetings and discussions on Canvas. |