Gage White

Lobster Notes Advanced Feature Implementation

Transaction Management/ACID compliance

Within the project I made a few meaningful changes that ensures ACID compliance within the database and in the interactions between the web application and the database.

Atomicity:

Within **app.py,** atomicity is enforced throughout with a few operations. *Try...catch/except* blocks are implemented throughout to handle transactions. I commented out redundant *conn.commit()* operations to ensure that if a transaction is a success, it is only committed once and to prevent partial writes in case of failure. I also implemented *conn.rollback()* for each modifying POST, PUT, DELETE method, so that in case of failure, the transaction is undone. All three enforce atomicity by ensuring all changes succeed or none of the changes succeed.

Consistency:

Consistency is mostly enforced through **Lobster Notes Tables.sql** and the constraints defined for each table and attribute. Within child tables referencing a parent table through a foreign key, I implemented *on delete cascade* commands. Initially, it was only for Student and Professor tables because attempting to delete users caused the web application to crash but I decided to add to each one in case additional functionality was added in the future. These prevent orphaned records in case a parent is deleted from the database.

Isolation:

Isolation is already handled by MySQL which enforces an isolation level that prevents transactions occurring at the same time to interfere with each other.

Durability:

Conn.*commit() handles durability by ensuring that transactions are permanently written to disk*.