# BREAKDOWN AND PRIORITIZE OF REQUIREMENTS

## GOAL

A complete application in Python with:

* Database interaction using MySQL
* Object-oriented code structure
* Custom exceptions
* Modular packages (entity, dao, util, etc.)
* Unit testing to validate your system

## WHAT ARE WE BUILDING

Building a finance management system with where there is a system and a user. The System for storing expenses in database, categorize them and provide reports. The user can register, login and manage expenses by adding/editing or deleting them.

Further an aim to integrate more features to make the application user friendly and advanced.

LOGIN

EXPENSE MANAGEMENT

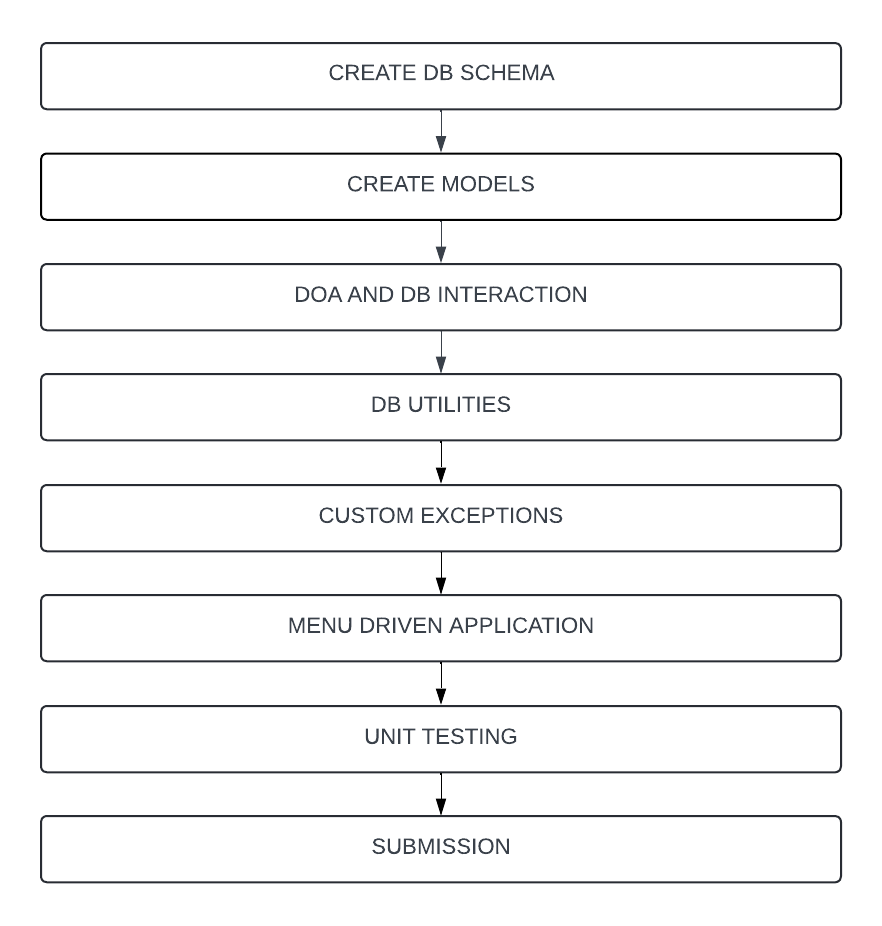
REPORT GENERATION

SYSTEM

USER

## BREAKDOWN OF REQUIREMENTS

| **Step** | **Task** | **Tech Used** |
| --- | --- | --- |
| Step 1 | Create DB Schema | MySQL |
| Step 2 | Create Models | Python Classes |
| Step 3 | DAO & DB interaction | MySQL Connector |
| Step 4 | DB Utilities & Properties | File I/O + Python Connector |
| Step 5 | Custom Exceptions | Python Exception Classes |
| Step 6 | Menu-Driven Application | Terminal + Logic |
| Step 7 | Unit Testing | unittest |
| Step 8 | GitHub Submission | Git/GitHub |



## EXPECTED RESULTS OF PRIORITIZED REQUIREMETS

### DATABASE

#### Users

|  |  |  |
| --- | --- | --- |
| Column | Type | Notes |
| user\_id | INT (PK) | Auto-incremented |
| username | VARCHAR(100) | Unique, not null |
| password | VARCHAR(100) | Hashed/plain, not null |
| email | VARCHAR(100) | Unique, not null |

#### ExpenseCategories

|  |  |  |
| --- | --- | --- |
| Column | Type | Notes |
| category\_id | INT (PK) | Auto-incremented |
| category\_name | VARCHAR(100) | Not null |

#### Expenses

|  |  |  |
| --- | --- | --- |
| Column | Type | Notes |
| expense\_id | INT (PK) | Auto-incremented |
| user\_id | INT (FK) | References Users(user\_id) |
| amount | DECIMAL(10, 2) | Not null |
| category\_id | INT (FK) | References ExpenseCategories(category\_id) |
| expense\_date | DATE | Not null |
| description | TEXT | Optional |

### MODELS

Create classes for User, Expense, ExpenseCategory. Add fields, constructors, getters/setters.

Technology used – Python

### EXCEPTION HANDLING

Create UserNotFoundException, ExpenseNotFoundException. Raise these in service methods when data isn’t found