1. Database (tables)

users (

user_id **INTEGER PRIMARY KEY**
username **CHAR(30)**,
password **CHAR(16)**

)

transactions (

transaction_id **INTEGER PRIMARY KEY**,
transaction_type **TEXT**,
transaction_category **TEXT**,
transaction_date **TEXT**,
transaction_amount **REAL**,
transaction_description **TEXT,**
user_id **INTEGER**,
FOREIGN KEY (user_id) **REFERENCES** users(user_id),

)

2. Backend

*__knows__ - *association* (line)
*__uses__ - *dependency* (line w/ single solid arrow head)
*__extends__ - *inheritance* (line /w single hollow arrow head)
*__has__ - *aggregation* (line w/ single hollow diamond head)
*__owns__ - *composition* (line w/ single solid diamond head)

```
enum TransactionTypes {
        “expense”,
        “savings”,
        “investment”,
        “income”
}

enum ExpenseCategories {
        "Bills",
        "Education",
        "Entertainment",
        "Food & Drinks",
        "Grocery",
        "Healthcare",
        "House",
        "Shopping",
        "Transportation",
        "Wellness",
        "Other"
}

enum SavingsCategories {
        "Monthly Allowance",
        "Change",
        "Miscellaneous"
}

enum InvestmentCategories {
        "Stocks",
```

```
        "Crypto",
        "Bonds",
        "Real Estate"
}

enum IncomeCategories {
        "Salary",
        "Bonus",
        "Side-hustles",
        "Tips"
}

class UserRepository {
        + addUser(username: str, password:str): int
        + deleteUser(username: str, password:str): int
}

class UserManager owns UserRepository {
        - username: str
        - password: str
        - current_user_id: int

        + signUp(): void
        + login(): bool
        + logout(): void
}

dataclass Transaction {
        - ID: int
        - date: str
        - type: str
        - category: str
        - amount: float
        - description: str
}
```

class TransactionRepository uses Transaction {

+ getAllTransactions(*current_user_id*: **int**): **List<Transaction>**
+ getTransactionsByType(*current_user_id*: **int**, *type*: **str**): **List<Transaction>**
+ getTransactionsByCategory(*current_user_id*: **int**, *category*: **str**): **List<Transaction>**
+ addTransaction(*current_user_id*: **int**, *transaction*: **Transaction**): **void**
+ modifyTransaction(*current_user_id*: **int**, *ID*: **int**, *transaction*: **Transaction**): **void**
+ deleteTransaction(*current_user_id*: **int**, *ID*: **int**): **void**

}

dataclass Finance {

- *total_income*: **float**
- *total_expenses*: **float**
- *total_savings:* **float**
- *total_investment*: **float**

}

class TransactionManager owns TransactionRepository, and uses Transaction and Finance {

- *current_user_id*: **int**
- *transaction_id*: **int**
- *transaction*: **Transaction**
- *overall_finance*: **Finance**
- *monthly_finances*: **List<Finance>**
- *quarterly_finances*: **List<Finance>**
- *overall_balance*: **float**

+ createTransaction(): **Transaction**
+ calculateOverallFinance(): **Finance**
+ calculateMonthlyFinances(): **List<Finance>**
+ calculateQuarterlyFinances(): **List<Finance>**
+ createMonthlyGraph(): **matplotlib.Figure**
+ createQuarterlyGraph(): **matplotlib.Figure**
+ calculateOverallBalance(): **float**

}