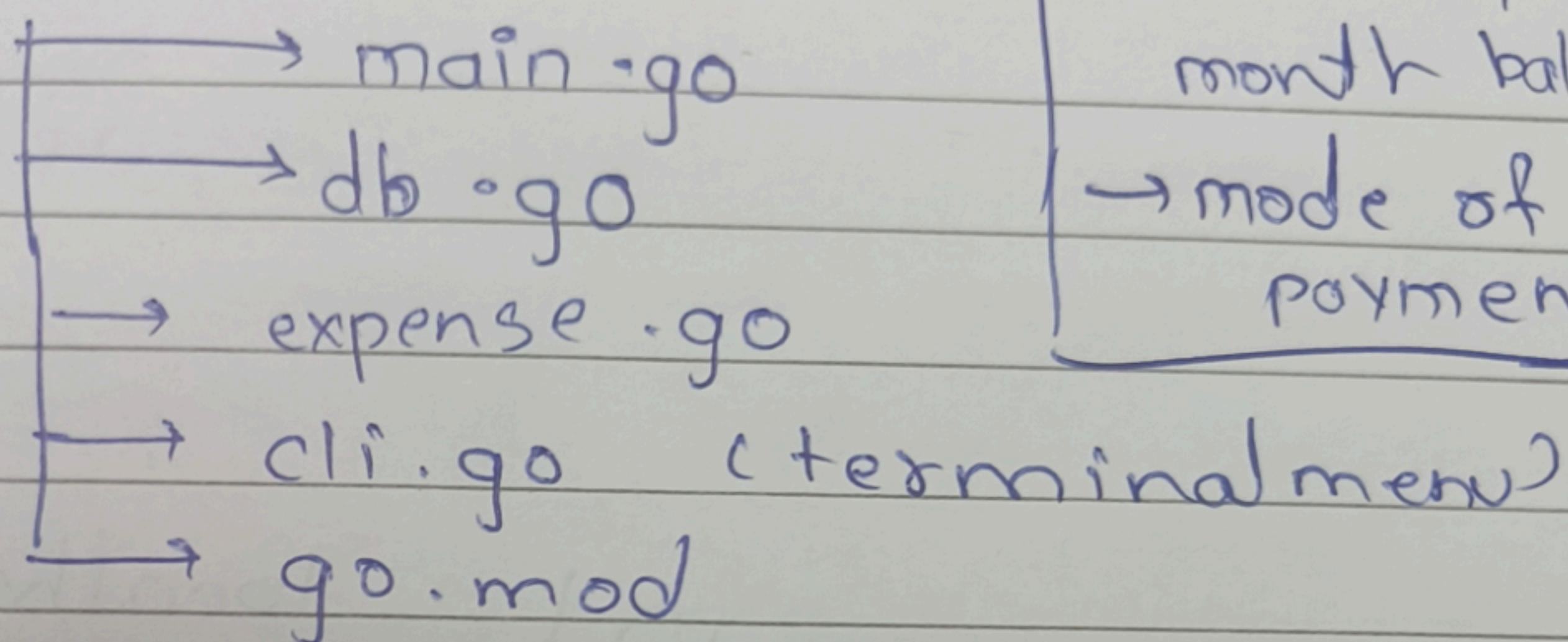


Terminal Based Personal Expense

Database : SQLite

Structure



- Changes
 - ↳ Added per month balance
 - ↳ mode of payment

DB structure

ID	amt	desc	category	date	Notes
int	float64	string	String	Time	string

Features

① add / view all expensence

② filters

→ category

→ data range

③ search

④ delete

⑤ stats

→ total expense

→ category

→ monthly

Dependency

1) dustin/go-humanize

↳ converts no. into human readable string

Ex : 1234567 → 1.2 MB

86400 → 1 day

2) uuid : unique id that will never collide

3) mattn/go-isatty :

↳ adds colours

↳ displaying at terminal or not

4) nreduces/go strftime

↳ date formatting

5) bigfft

↳ very large integers / floats efficient arithmetic

6) x/exp

↳ performance optimization

(go features that are not in STL)

7) x/sys

↳ directly talks to system / os

↳ Application → Model ← GoLang (4)
 Model → Application → User

87 /libc

↳ Go implementation of CSL

9) | memory

↳ ensures no memory leakage / crash

10) | mathutil

↳ mathematical helper func

① expense.go : struct / data model

② database.go : Handles all database operations

① connection ② table_creation ③ CRUD

↳ pure data access - executes SQL query & return data, no business logic

③ cli.go : User Interface Layer

Role 8 User prompts, validates input, formats & output displays
→ No db calls

- ④ main.go → Application entry
↳ sets scanned
↳ menu for CLI
↳ connects cli.go with database.go

