PocketSage UML Diagrams

Team Members: Kheiven Dhaiti, Vedell Jones, Lucas Vega, Dossell Sinclair.

Classes

C Account

currency: str holdings : List['Holding'] id : Optional[int] name : str transactions : List['Transaction']

C AmortizationWriter write schedule() -> None C AppSetting

description : Optional[str] kev : str value : str

C Budget

id : Optional[int] label: str lines : List['BudgetLine'] period_end : date period start : date

c BudgetLine

budget : str budget id:int category_id : int id : Optional[int] planned_amount : float rollover enabled bool © BudgetRepository

actual spend() -> Iterable[tuple[int, float]] planned_amounts() -> Iterable[tuple[int, float]]

C DevConfig DEBUG : bool TESTING: bool

(C) Budget∨ariance

actual : float category_id: int delta : float planned : float

(C)CSVImporter

C Category

category_type: str color : Optional[str] id : Optional[int] name : str slug : str transactions : List['Transaction'] C ColumnMapping

account id: str | None account_name : str | None amount : str category : str | None currency: str | None external_id : str | None memo : str | None occurred at: str

C DebtAccount

apr:float balance : float id : int minimum_payment : float (c) Habit

cadence : str description : str entries : List['HabitEntry'] id : Optional[int] is active bool name : str

C BaseConfig

APP_NAME : str DATABASE URL: NoneType DATA_DIR : Path DB FILENAME : str SECRET_KEY: NoneType SQLCIPHER FLAG: str SQLCIPHER_KEY_ENV : str SQLITE PRAGMAS: dict USE SQLCIPHER : bool

sqlalchemy_engine_options() -> dict[str, Any]

(C) HabitEntry

habit : str habit_id : int occurred_on : date value : int

(C) HabitForm cadence : str description: str

name : str validate() -> bool C HabitsRepository

create habit() -> Habit list habits() -> Iterable[Habit] record_entry() -> HabitEntry C Holding

account : Optional['Account'] account id: Optional[int] acquired at: Optional[datetime] avg_price : float currency : str id: Optional[int] quantity : float symbol: str

C Job

created at: datetime error : Optional[str] finished_at: Optional[datetime] id : str metadata : Dict[str, Any]

name : str started_at : Optional[datetime] status : str

to dict() -> Dict[str, Any]

C LedgerEntryForm

amount : float | None category_id : Optional[int] memo: str

occurred_at : datetime | None

validate() -> bool

c LedgerRepository

create_transaction() -> Transaction list transactions() -> Iterable[Transaction] update_transaction(transaction_id: int) -> Transaction C LiabilitiesRepository

create_liability() -> Liability list_liabilities() -> Iterable[Liability] schedule_payoff() -> None

C Liability

apr:float balance : float due day: int id : Optional[int] minimum payment: float name : str opened_on : Optional[date] payoff_strategy : str

c LiabilityForm

apr : Decimal | None balance : Decimal | None minimum payment: Decimal | None name : str

CPortfolioImportForm file path: str

validate() -> bool

C PortfolioRepository allocation_summary() -> dict import positions() -> int list_holdings() -> Iterable[dict]

validate() -> bool

C ReportRenderer

render(figure: Figure) -> None

C SqlModelPortfolioRepository session

allocation_summary() -> dict import_positions() -> int list holdings()

(c) Transaction

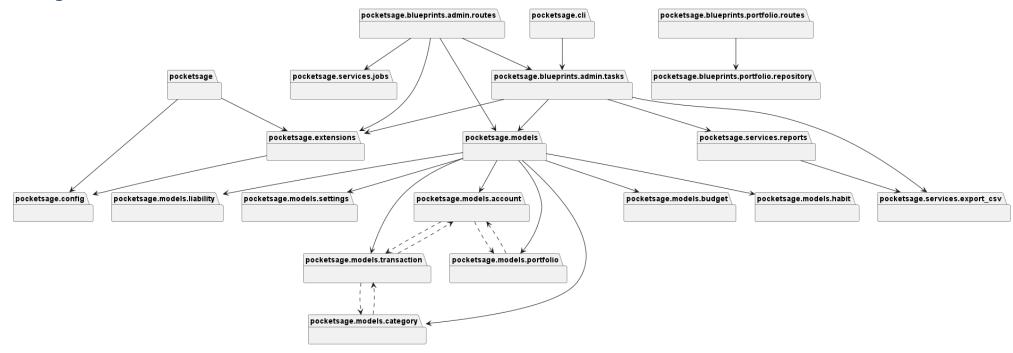
account : Optional['Account'] account_id : Optional[int] amount : float category: Optional['Category'] category_id : Optional[int] currency: str external id : Optional[str] id : Optional[int] memo: str occurred_at : datetime

(C) Transaction TagLink tag id:int

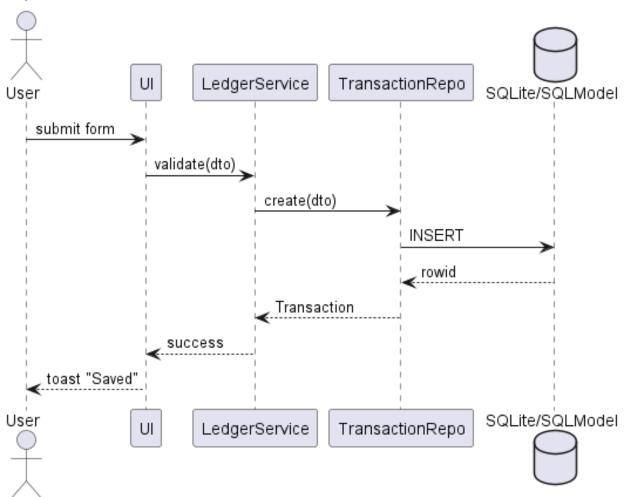
transaction_id : int

(C) Handler on_created(event) -> None

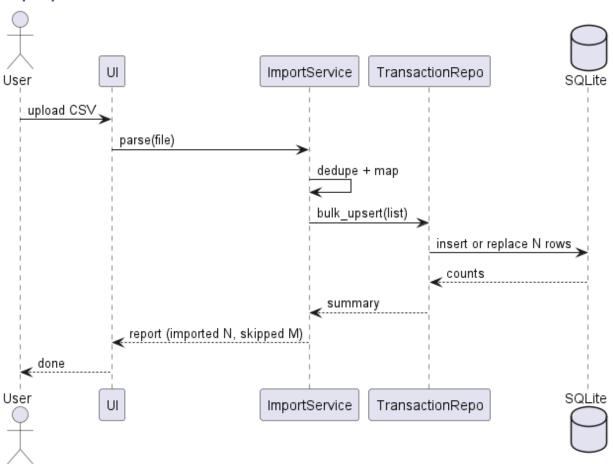
Packages



Seq Add Transaction



Seq Import Csv



Use cases

