

Template Design Pattern

→ Behavioral,
why required & when to use?

→ When we want all classes to follow the specific steps to process the task but also

Need to provide the flexibility that Each class can have their own logic in that specific step.

⇒ UML

Payment flow << Interface >>

- ① Validate Request() = 0
- ② Calculate fees() = 0
- ③ debit Amount() = 0
- ④ Credit Amount() = 0

find Send Money()

- ① Validate Request();
- ② debit Amount();
- ③ Calculate fees();
- ④ Credit Amount();

Unique Impl. with
Each children class but
order of Impl. remain
Same.

Pay To Friend

- 1 Validate Request()
- 2 debit Amount()
- 3 Calculate fees()
- 4 Credit Amount()

Pay To Merchant

- 1 Validate Request()
- 2 debit Amount()
- 3 Calculate fees()
- 4 Credit Amount()