

# **OOAD Stage 2040**

**| Payback ATM |**

Mun gi tae / Han sang min

# *Chart*

**Real Use Cases**

**System  
Architecture**

**Design Class  
Diagram**

**UI**

**Interaction  
Diagram**

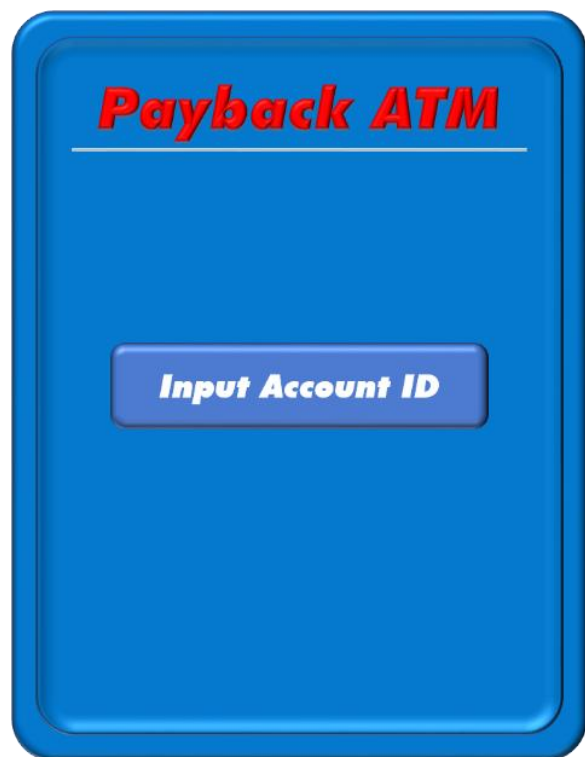
**Traceability  
Analysis**

<b>Use Case</b>	<b>1. Send</b>
<b>Actor</b>	User
<b>Purpose</b>	송금기능을 수행한다.
<b>Overview</b>	사용자가 원하는 대상의 계좌로 돈을 이체한다.
<b>Type</b>	Primary and essential
<b>Cross Reference</b>	System function: 1.2, 1.3 Use Case: Limited Amount, Check Password
<b>Pre-Requisites</b>	사용자와 송금대상의 계좌정보가 모두 txt파일에 있어야 한다.
<b>Typical Courses of Events</b>	(A): User (S): System 1. (A)가 화면에 있는 빈칸에 계좌번호를 입력한다. 2. (S)가 입력된 계좌번호에 해당되는 정보를 불러온다. 3. (A)가 송금 기능의 버튼을 누르고 비밀번호를 입력한다. 4. (A)는 상대 은행을 누르고 계좌번호를 입력한다. 5. (A)가 송금할 금액을 입력한다 6. (S)가 금액이 한도에 맞는지 확인하고 진행시킨다. 7. (S) 송금을 진행시키고 명세서를 출력할 것인지 물어본다. 8. (A) 명세서를 뽑을지 말지 Y/N을 누른다. 9. (S) 초기화면으로 되돌린다
<b>Alternative Course of Events</b>	N/A
<b>Exceptional Courses of Events</b>	(A)가 ATM기기에 걸려있는 한도금액보다 초과하여 입력하면 Limit amount에 의해 다시 인출 액 입력으로 돌아간다. (A)가 비밀번호를 잘못 입력하면 다시 비밀번호 입력으로 돌아가게 된다.

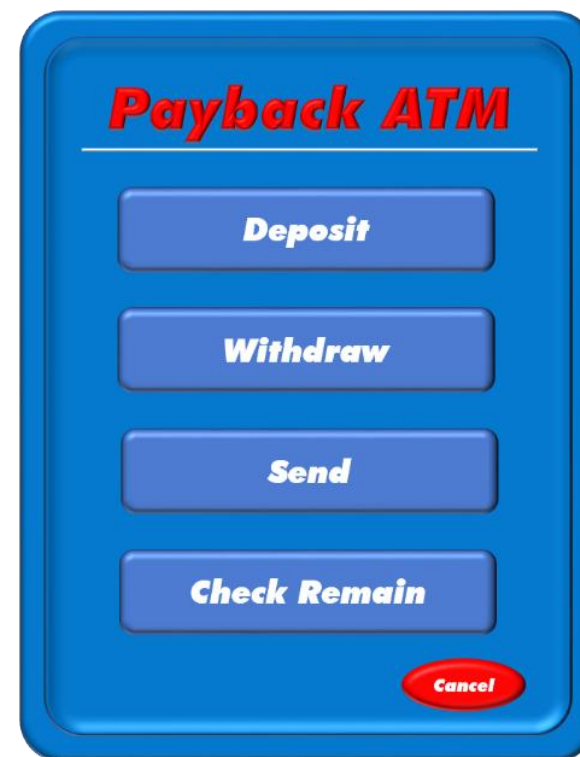
<b>Use Case</b>	<b>2. withdraw</b>
<b>Actor</b>	User
<b>Purpose</b>	인출기능을 수행한다.
<b>Overview</b>	사용자가 ATM기기에서 돈을 인출한다
<b>Type</b>	Primary and essential
<b>Cross Reference</b>	System function: 1.2, 1.3 Use Case: Limited Amount, Check Password
<b>Pre-Requisites</b>	해당 계좌의 정보가 txt파일에 있어야 한다.
<b>Typical Courses of Events</b>	(A): User (S): System 1. (A)가 화면에 있는 빈칸에 계좌번호를 입력한다. 2. (S)가 입력된 계좌번호에 해당되는 정보를 불러온다. 3. (A)가 인출 버튼을 누르고 인출할 금액을 입력한다 4. (S)가 금액이 한도에 맞는지 확인하고 비밀번호를 요구한다 5. (A)가 비밀번호를 입력한다 6. (S)가 비밀번호를 확인하고 인출을 시키며 명세서를 뽑을지 말지 물어본다 7. (A)가 Y/N을 눌러서 명세서를 뽑을지 말지 결정한다 8. (S) 초기화면으로 되돌린다
<b>Alternative Course of Events</b>	N/A
<b>Exceptional Courses of Events</b>	(A)가 ATM기기에 걸려있는 한도금액보다 초과하여 입력하면 Limit amount에 의해 다시 송금액 입력으로 돌아간다. (A)가 비밀번호를 잘못 입력하면 다시 비밀번호 입력으로 돌아가게 된다.

Use Case	3. Deposit
Actor	User
Purpose	입금기능을 수행한다.
Overview	ATM기기를 통해 계좌에 입금한다.
Type	Primary and essential
Cross Reference	N/A
Pre-Requisites	해당 계좌의 정보가 txt파일에 있어야 한다
Typical Courses of Events	(A): User (S): System 1. (A)가 화면에 있는 빈칸에 계좌번호를 입력한다. 2. (S)가 입력된 계좌번호에 해당되는 정보를 불러온다. 3. (A)가 입금 버튼을 누른 다음 입금할 금액을 입력한다. 4. (S)가 입금을 진행시키고 명세서를 뽑을지 말지 물어본다 5. (A)가 Y/N을 눌러서 명세서를 뽑을지 말지 결정한다 6. (S) 초기화면으로 되돌린다
Alternative Course of Events	N/A
Exceptional Courses of Events	N/A

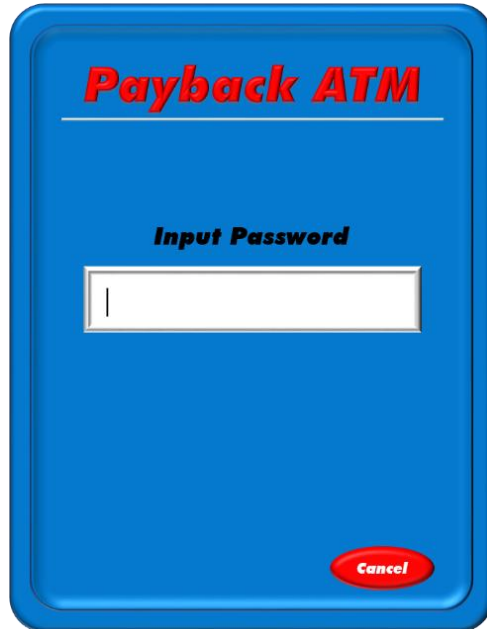
<b>Use Case</b>	<b>4. Check Remain</b>
<b>Actor</b>	User
<b>Purpose</b>	잔액조회기능을 수행한다.
<b>Overview</b>	ATM기기에서 잔액 조회를 한다
<b>Type</b>	Primary and essential
<b>Cross Reference</b>	System function: 1.2 Use Case: Check Password
<b>Pre-Requisites</b>	해당 계좌의 정보가 txt파일에 있어야 한다
<b>Typical Courses of Events</b>	(A): User (S): System 1. (A)가 화면에 있는 빈칸에 계좌번호를 입력한다. 2. (S)가 입력된 계좌번호에 해당되는 정보를 불러온다 3. (A)가 잔액조회 버튼을 누른다 4. (S)가 비밀번호를 물어본다 5. (A)가 비밀번호를 입력한다 6. (S)가 비밀번호를 확인하고 잔액을 조회한 후 명세서를 뽑을지 말지 물어본다 7. (A)가 Y/N을 눌러서 명세서를 뽑을지 말지 결정한다 8. (S) 초기화면으로 되돌린다
<b>Alternative Course of Events</b>	N/A
<b>Exceptional Courses of Events</b>	N/A



***Initial Screen***



***Main Screen***



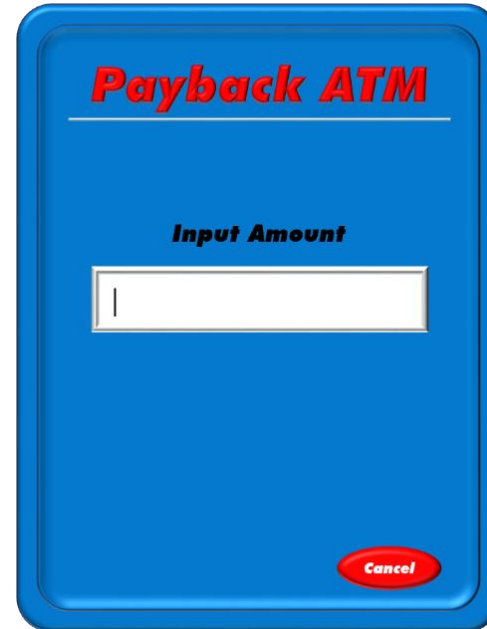
**Payback ATM**

---

**Input Password**

**Cancel**

**Input Password Screen**



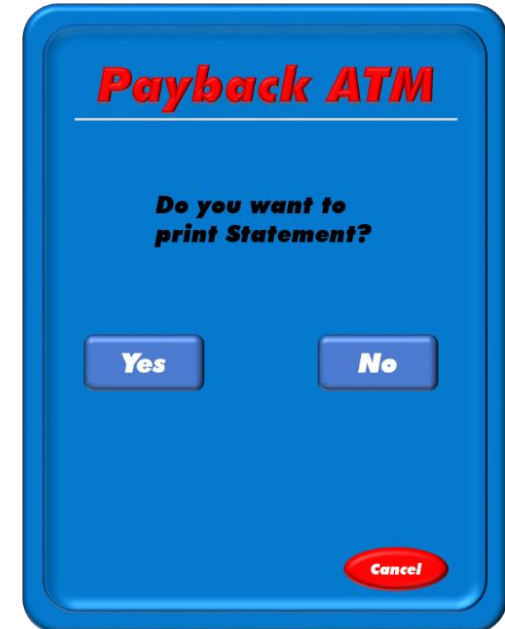
**Payback ATM**

---

**Input Amount**

**Cancel**

**Input Amount Screen**



**Payback ATM**

---

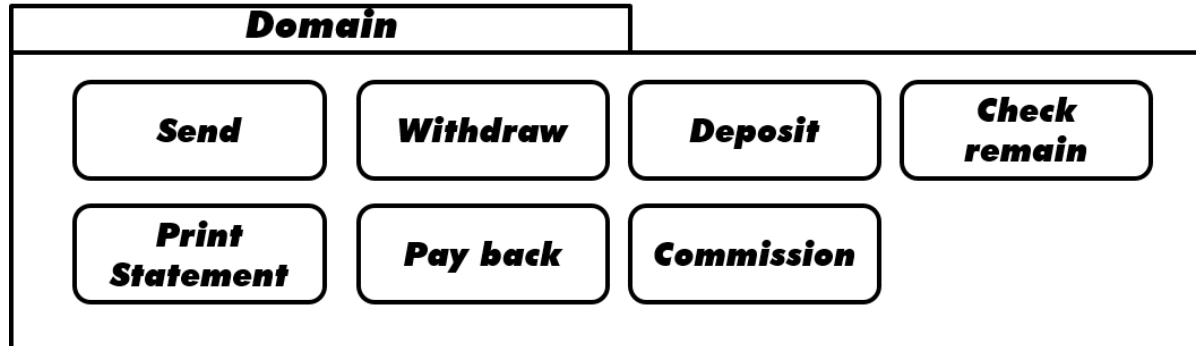
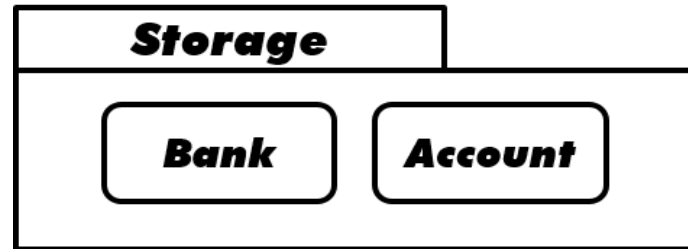
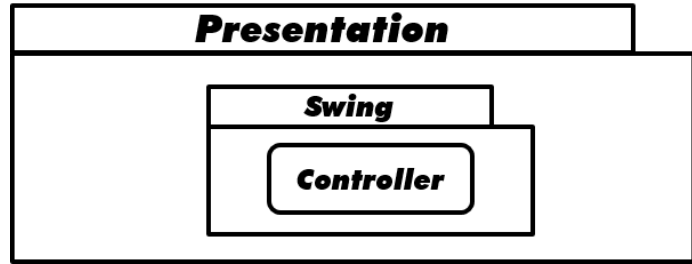
**Do you want to  
print Statement?**

**Yes**      **No**

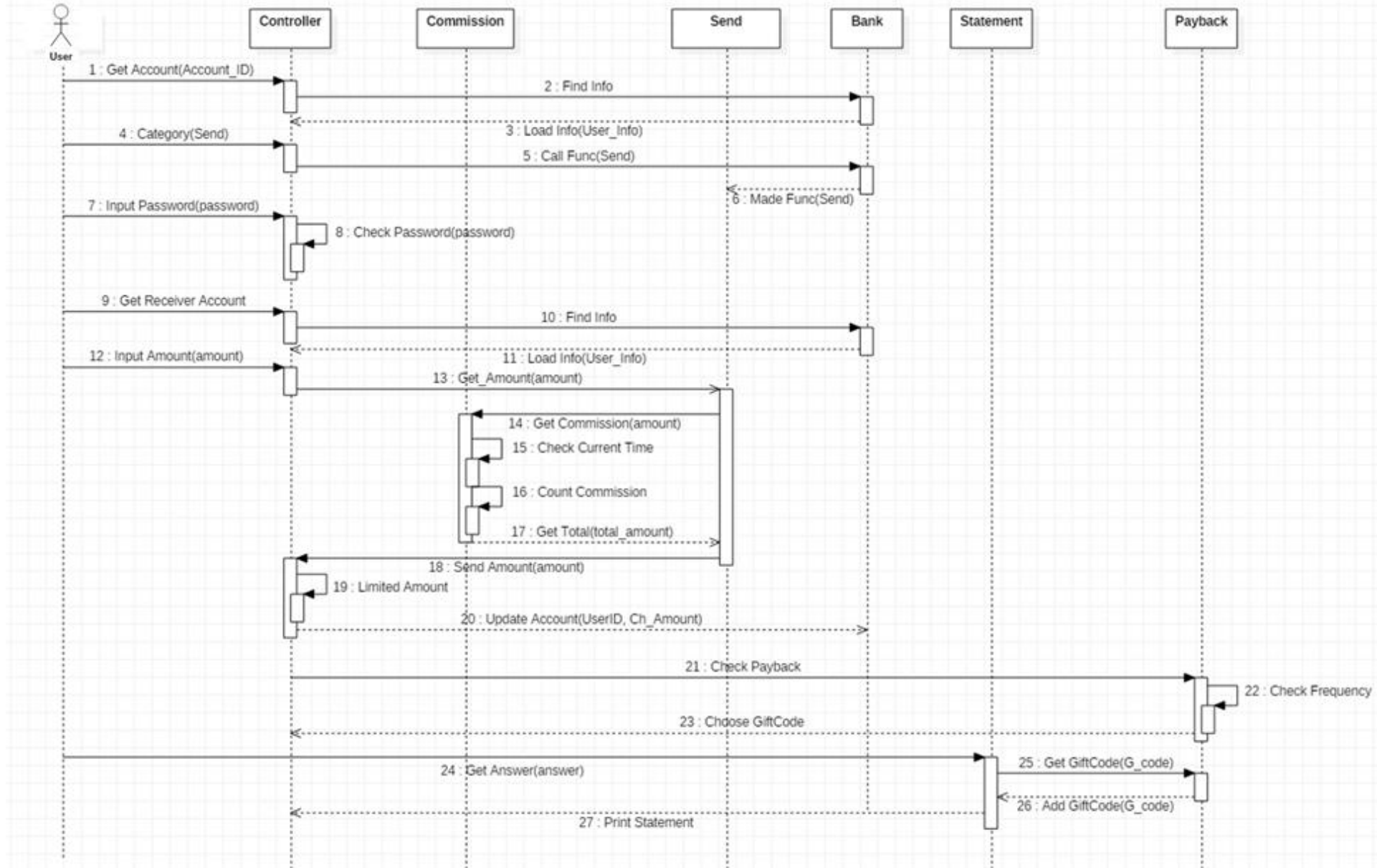
**Cancel**

**Get Answer Screen**

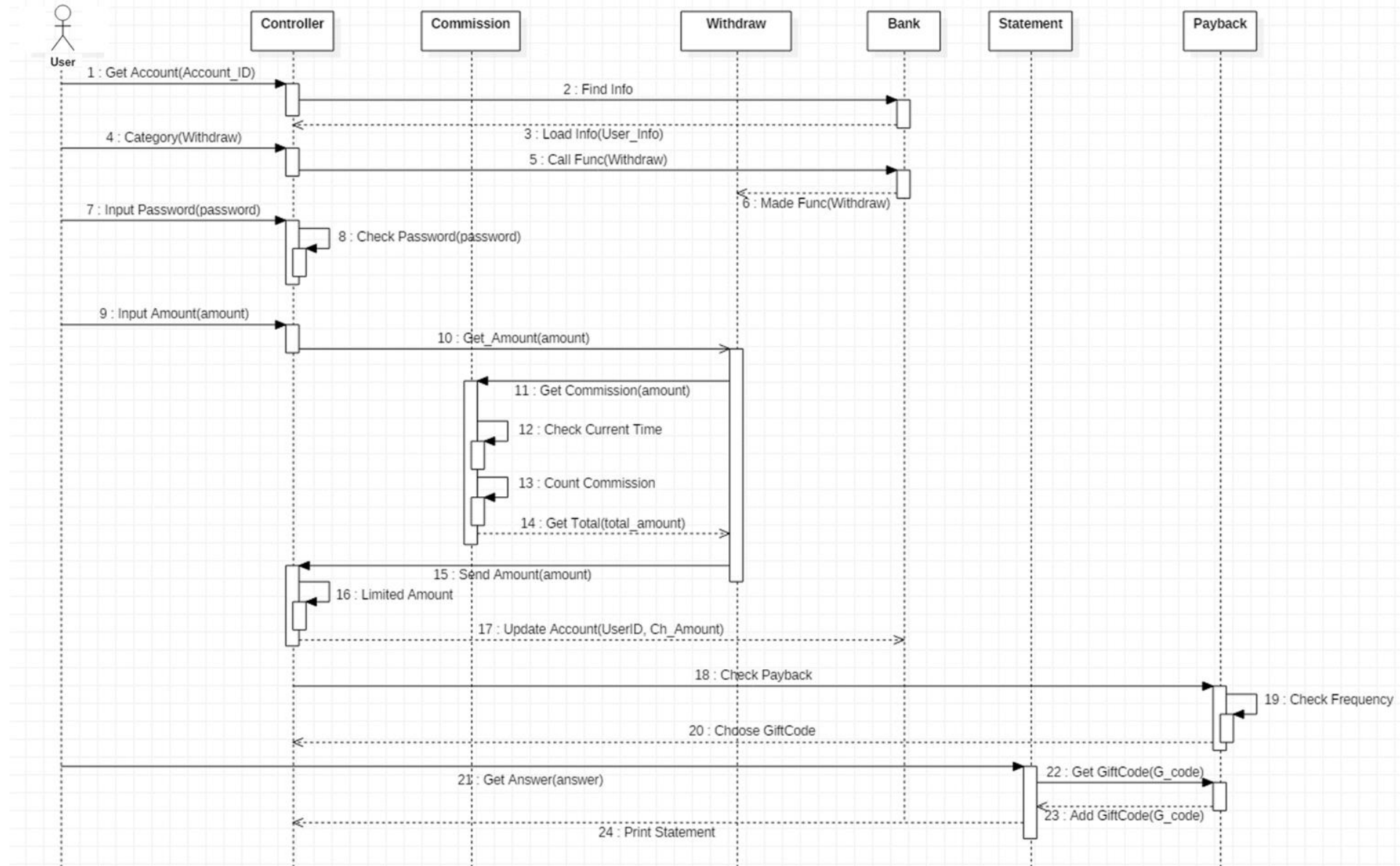




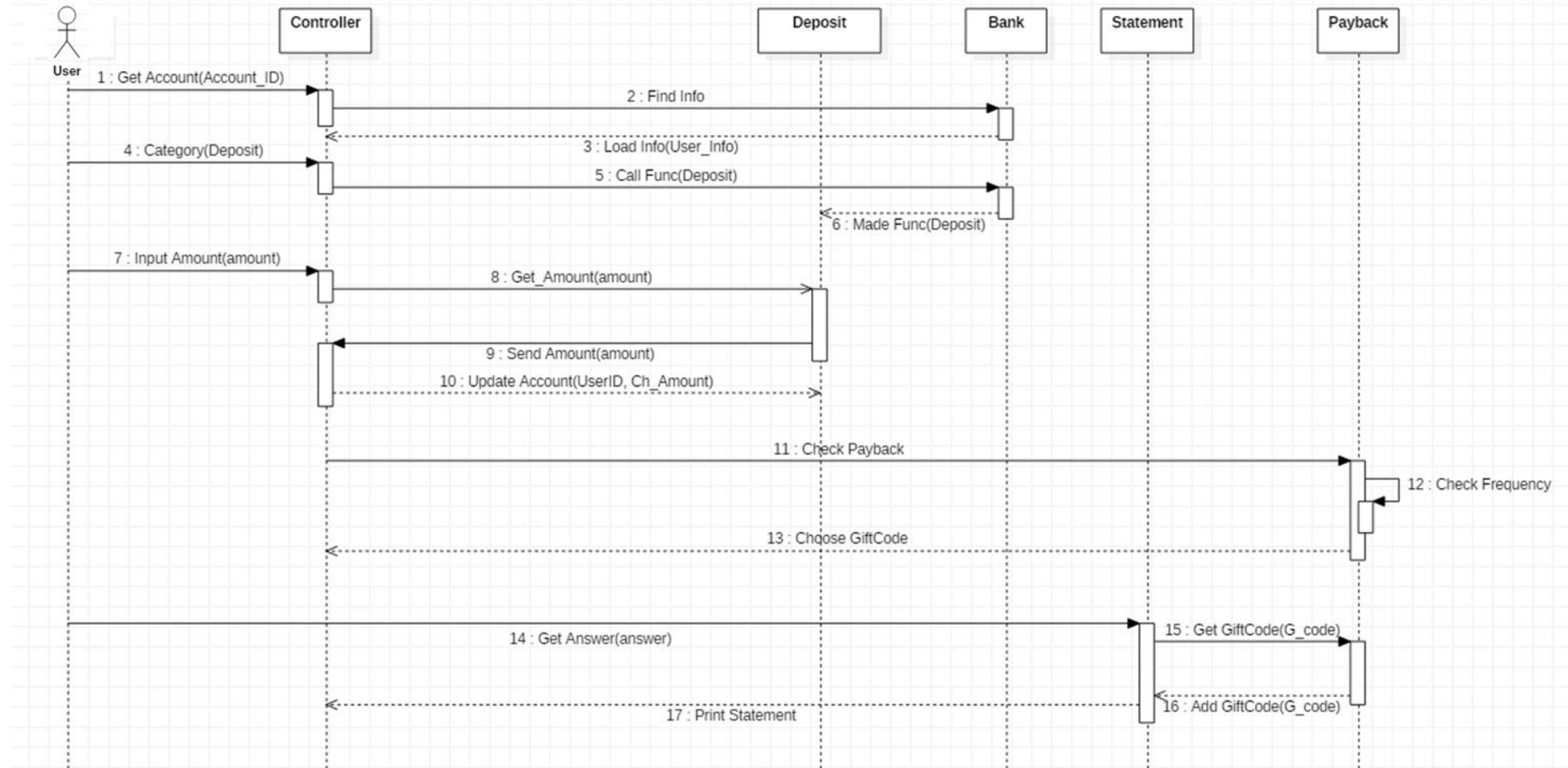
# <Send>



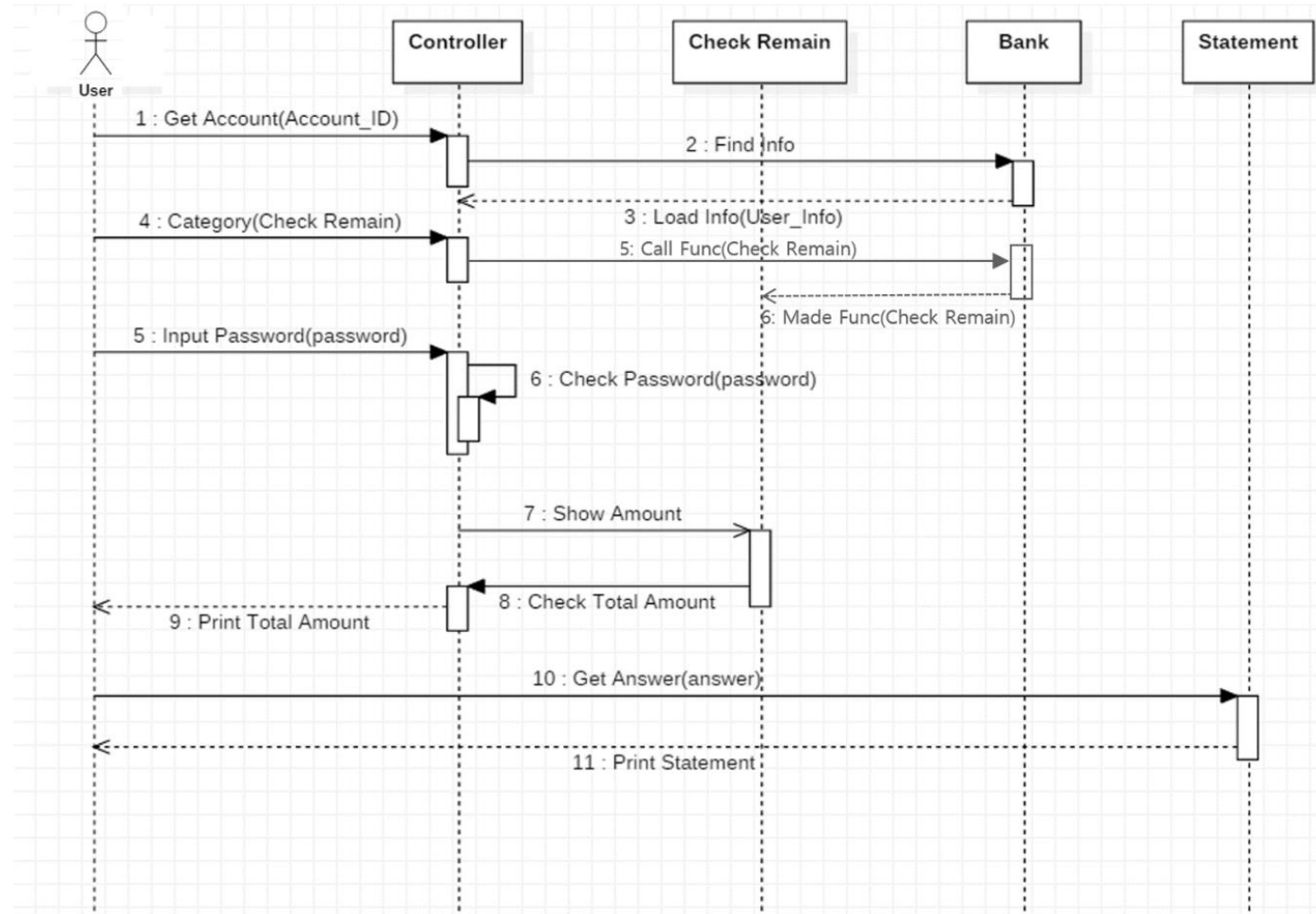
# <Withdraw>

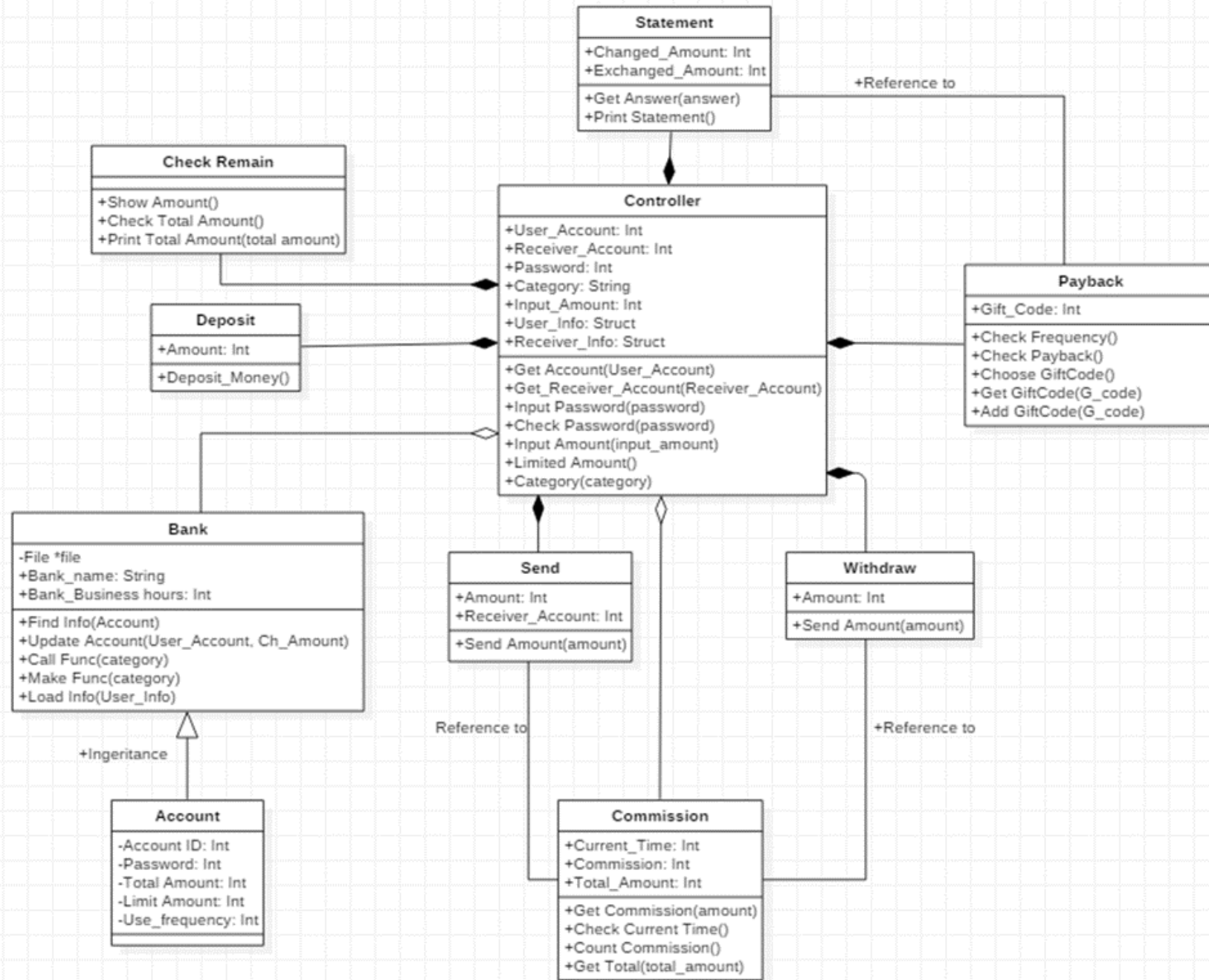


# <Deposit>



## <Check remain>





Operation in Sequence Diagrams		Operation in Interaction Diagrams		Methods	Class
1. Get Account		Get Account(account_id)		Get Account(account_id: int): void	Controller
2. Category		Get Receiver_Account(account_id)		Get Receiver_Account(account_id: int): void	
2. Input Password		Input Password(password)		Input Password(password: int): void	
3. Get Receiver Account		Check Password(password)		Check Password(password: int): boolean	
4. Input Amount		Input Amount(amount)		Input Amount(amount: int): void	
5. Print Remain Amount		Check Limit(amount)		Check Limit(amount: int): Boolean	
6. Get Answer		Category(category)		Category(category: String): void	Bank
		Find Info(account_id)		Find Info(account_id : int): void	
		Load Info()		Load Info(): void	
		Call Func(category)		Call Func(category: String): void	
		Make Func(category)		Make Func(category: String): void	Commission
		Update Account(User_id,Ch_amount)		Update Account(User_id: int,Ch_amount: int):void	
		Count Commission()		Count Commission(amount: int): int	
		Get Commission(amount)		Get Commission(amount: int): void	
		Check Current Time()		Check Current Time(): int	Payback
		Get Total(total_amount)		Get Total(total_amount)	
		Check Payback(frequency)		Check Payback(frequency: int): void	
		Check Frequency(frequency)		Check Frequency(frequency: int): Boolean	
		Get Gift Code(G_code)		Get Gift Code(G_code: int): void	Statement
		Add Gift Code(G_code: int)		Add Gift Code(G_code: int): void	
		Choose Gift Code()		Choose Gift Code(): void	Send/Withdraw
		Get Answer(answer)		Get Answer(answer: String): Boolean	
		Print Statement()		Print Statement(): void	Check Remain
		Send Amount(amount)		Send Amount(amount: int): int	
		Show Amount()		Show Amount(): void	
		Check Total Amount()		Check Total Amount(): int	Check Remain
		Print Total Amount(total_amount)		Print Total Amount(total_amount: int): void	

**Q & A**