programming Assignment

2021
Project Java Part1

Computer Engineering

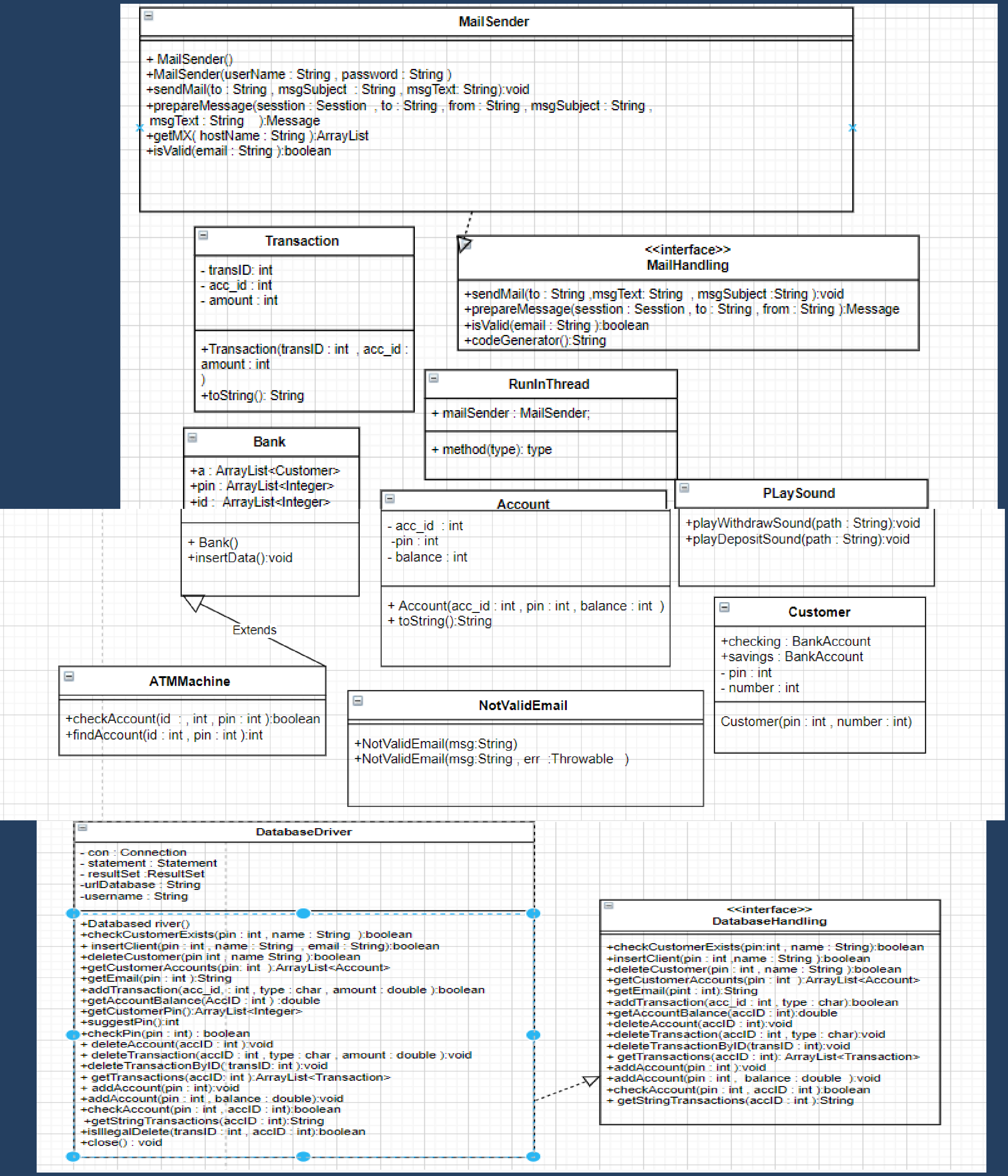
Student's :

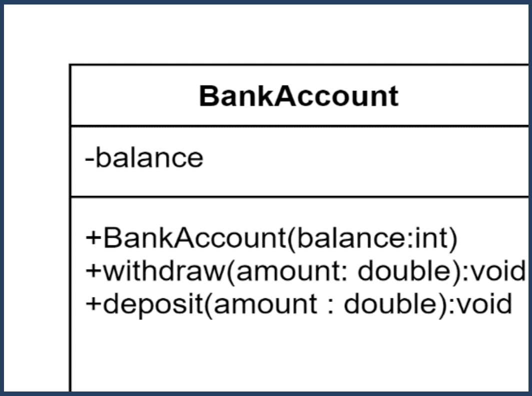
- Sadeg Ashour
- Ibrahim Isleem
- Abd Alrahman yaseen

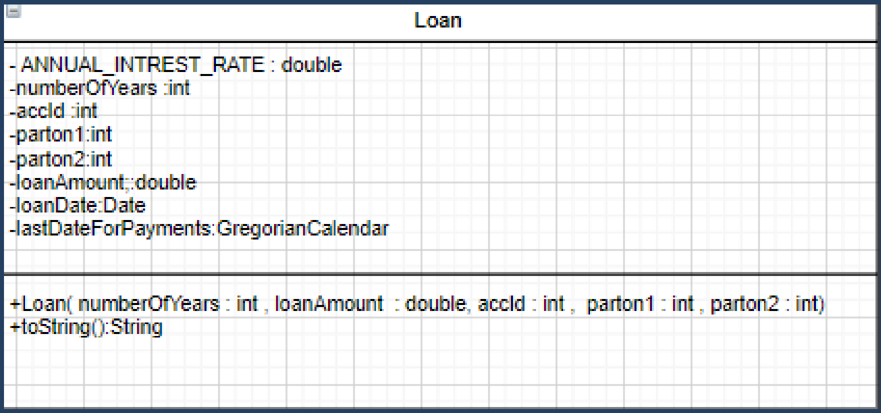
Dr : Ayman Maliha
Eng : Mohammed Al madhoon

The Overall UML diagram

- ❖The overall daiagram seprated to be readable :


MailSender
+ MailSender()
+MailSender(userName : String , password : String)
+sendMail(to : String , msgSubject : String , msgText: String):void
+prepareMessage(sesstion : Sesstion , to : String , from : String , msgSubject : String , msgText : String):Message
+getMX(hostName : String):ArrayList
+isValid(email : String):boolean
Transaction
- transID: int
- acc_id : int
- amount : int
+Transaction(transID : int , acc_id : amount : int)
+toString(): String
<<interface>>
MailHandling
+sendMail(to : String ,msgText: String , msgSubject :String):void
+prepareMessage(sesstion : Sesstion , to : String , from : String):Message
+isValid(email : String):boolean
+codeGenerator():String
RunInThread
+ mailSender : MailSender;
+ method(type): type
Bank
+a : ArrayList<Customer>
+pin : ArrayList<Integer>
+id : ArrayList<Integer>
+ Bank()
+insertData():void
Account
- acc_id : int
-pin : int
- balance : int
+ Account(acc_id : int , pin : int , balance : int)
+ toString():String
PLaySound
+playWithdrawSound(path : String):void
+playDepositSound(path : String):void
Extends
ATMMachine
+checkAccount(id : , int , pin : int):boolean
+findAccount(id : int , pin : int):int
Customer
+checking : BankAccount
+savings : BankAccount
- pin : int
- number : int
Customer(pin : int , number : int)
NotValidEmail
+NotValidEmail(msg:String)
+NotValidEmail(msg:String , err :Throwable)
DatabaseDriver
- con : Connection
- statement : Statement
- resultSet :ResultSet
-urlDatabase : String
-username : String
+Databased river()
+checkCustomerExists(pin : int , name : String):boolean
+ insertClient(pin : int , name : String , email : String):boolean
+deleteCustomer(pin int , name String):boolean
+getCustomerAccounts(pin: int):ArrayList<Account>
+getEmail(pin : int):String
+addTransaction(acc_id, : int , type : char , amount : double):boolean
+getAccountBalance(AccID : int) :double
+getCustomerPin():ArrayList<Integer>
+suggestPin():int
+checkPin(pin : int) : boolean
+ deleteAccount(accID : int):void
+ deleteTransaction(accID : int , type : char , amount : double):void
+deleteTransactionByID(transID: int):void
+ getTransactions(accID: int):ArrayList<Transaction>
+ addAccount(pin : int):void
+addAccount(pin : int , balance : double):void
+checkAccount(pin : int , accID : int):boolean
+getStringTransactions(accID : int):String
+isIllegalDelete(transID : int , accID : int):boolean
+close() : void
<<interface>>
DatabaseHandling
+checkCustomerExists(pin:int , name : String):boolean
+insertClient(pin : int ,name : String):boolean
+deleteCustomer(pin : int , name : String):boolean
+getCustomerAccounts(pin : int):ArrayList<Account>
+getEmail(pint : int):String
+addTransaction(acc_id : int , type : char):boolean
+getAccountBalance(accID : int):double
+deleteAccount(accID : int):void
+deleteTransaction(accID : int , type : char):void
+deleteTransactionByID(transID : int):void
+ getTransactions(accID : int): ArrayList<Transaction>
+addAccount(pin : int):void
+addAccount(pin : int , balance : double):void
+checkAccount(pin : int , accID : int):boolean
+ getStringTransactions(accID : int):String


BankAccount
-balance
+BankAccount(balance:int)
+withdraw(amount: double):void
+deposit(amount : double):void


Loan
- ANNUAL_INTREST_RATE : double
-numberOfYears :int
-accId :int
-parton1:int
-parton2:int
-loanAmount::double
-loanDate:Date
-lastDateForPayments:GregorianCalendar
+Loan(numberOfYears : int , loanAmount : double, accId : int , parton1 : int , parton2 : int)
+toString():String

- To see it in aa good way follow this link and then open it on darw.io :

the UML diagram

Breif description about the classes and its methods:

This description without the constructors simply because they don't need any description

and also the description of the method is related to the corresponding method

The Diagram

The Descrition

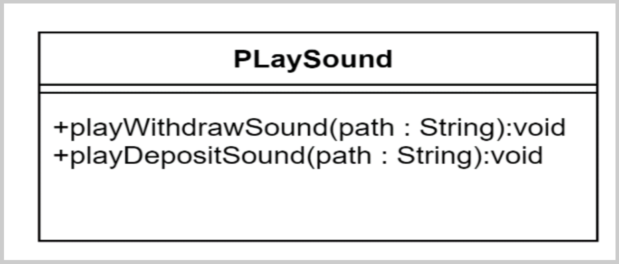
```
MailSender
----------------------------------------
----------------------------------------
+ MailSender()
+MailSender(userName : String , password : String )
+sendMail(to : String , msgSubject  : String , msgText: String):void
+prepareMessage(sesstion : Sesstion  , to : String , from : String , msgSubject : String ,
 msgText : String    ):Message
+getMX( hostName : String ):ArrayList
+isValid(email : String ):boolean
```

- This class for the email sending-related operations.
- sendMail : to send an email
- prepareMessage: to prroduce message can be sent
- getMX: check server email Is be or not
- isValid : check email is valid or not

```
DatabaseDriver
----------------------------------------
- con : Connection
- statement : Statement
- resultSet :ResultSet
-urlDatabase : String
-username : String
----------------------------------------
+Databased river()
+checkCustomerExists(pin : int , name : String  ):boolean
+ insertClient(pin : int , name : String  , email : String):boolean
+deleteCustomer(pin int , name String ):boolean
+getCustomerAccounts(pin: int  ):ArrayList<Account>
+getEmail(pin : int ):String
+addTransaction(acc_id, : int , type : char , amount : double ):boolean
+getAccountBalance(AccID : int ) :double
+getCustomerPin():ArrayList<Integer>
+suggestPin():int
+checkPin(pin : int) : boolean
+ deleteAccount(accID : int ):void
+ deleteTransaction(accID : int , type : char , amount : double ):void
+deleteTransactionByID( transID: int ):void
+ getTransactions(accID: int ):ArrayList<Transaction>
+ addAccount(pin : int):void
+addAccount(pin : int , balance : double):void
+checkAccount(pin : int , accID : int):boolean
 +getStringTransactions(accID : int):String
+isIllegalDelete(transID : int , accID : int):boolean
+close() : void
```

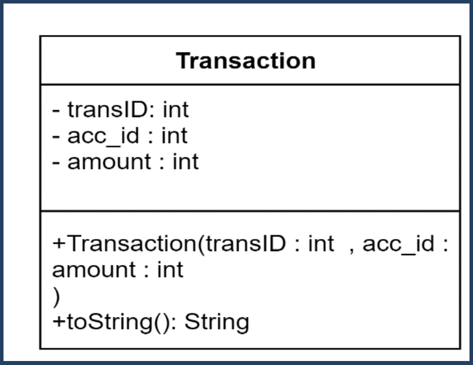
- this class for database management operations .
- checkCustomerExists : check this customer is exist or not
- insertClient : insert a new customer into Customer table.
- deleteCustomer : delete a customer and automatically delete all his accounts and transactions and loans.
- getCustomerAccounts : return all Accounts this customer have as an ArrayList<Accounts>.
- getEmail : return the cusotmer email.
- addTransaction : Not in use anymore
- addTransaction : add a transaction to Transactions table
- getTransactionsNumber : return how much transactions this account have.
- getAccountBalance : calculate the account balance from transaction table.
- getCustomerPin : return all PIN's from customer table.
- suggestPin : generate a random pin as a suggestion.
- checkPin : check if this pin is in use or not.
- Delete Account: remove an account from Accounts table
- deleteTransaction : delete a general transaction which match a conditions for specific account
- deleteTransactionByID: delete a specific transaction by id.
- deleteAllTransactions: clean transactions for specific account.
- getEmailFromAccid : get the customer email, the owner of this account
- getPin : return the pin for customer, the owner of this account
- get Transaction's: add an account with initial balance is zero.
- addAccount :add an account with initial balance as an argument into Accounts table.
- checkAccount: check if an account with specific id exist or not
- get String Transactions: return transaction for specific account as a String

- isIllegalDelete: check if this deletion method is legal or not
- getLastAccount: return id for last account this custmer created
- isThisForThat: return if this account to this customer or not
- showPages: return the group of transaction as a page
- getLastNRecords : return the last n for the transactions
- insertLoan : insert row in the loan table
- get Start DateLoan: return the string of the loan withdrawal date from database
- getEndDateLoan: return the last payment date as a string from database
- getLoanAmount: Returns the amount to be repaid from the loan
- setAccountOpen : set the account status
- checkAccountStatus : return The status of the account if it is closed or not
- getLoanPaid : Returns the value the account paid
- getPatrons : return the patrons to this account
- deleteLoan : delete the loan in database
- getWithdrawn: return the balance that the customer withdraw it in the loan
- getLoan: return the loan object to account
- close : to close the connection with the database which we created it in the Constructor
-

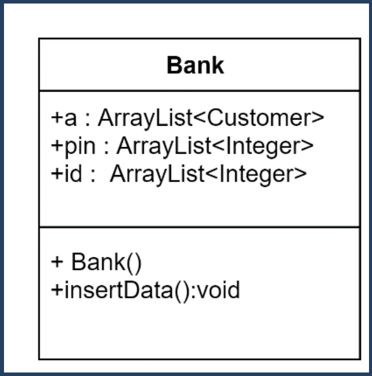


```
PLaySound
---------------------------------------
+playWithdrawSound(path : String):void
+playDepositSound(path : String):void
```

- To open a play a music or sounds while other code excuted and the extention of it .wav .

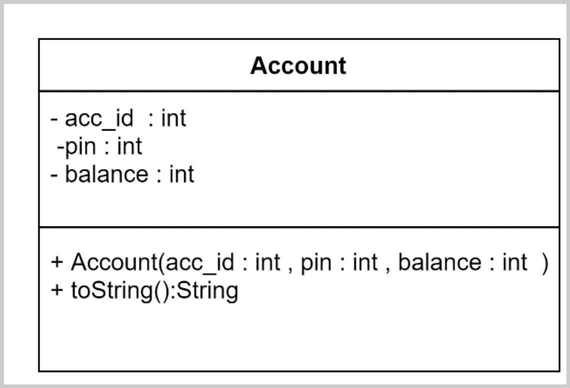


```
Transaction
---------------------------------------
- transID: int
- acc_id : int
- amount : int
---------------------------------------
+Transaction(transID : int , acc_id :
amount : int
)
+toString(): String
```

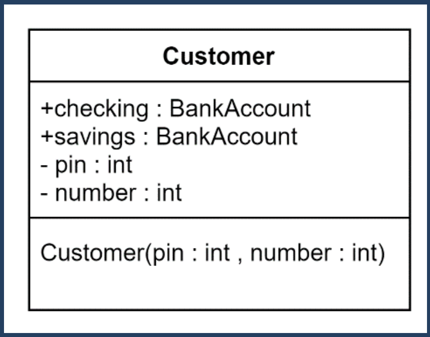


```
Bank
---------------------------------------
+a : ArrayList<Customer>
+pin : ArrayList<Integer>
+id : ArrayList<Integer>
---------------------------------------
+ Bank()
+insertData():void
```

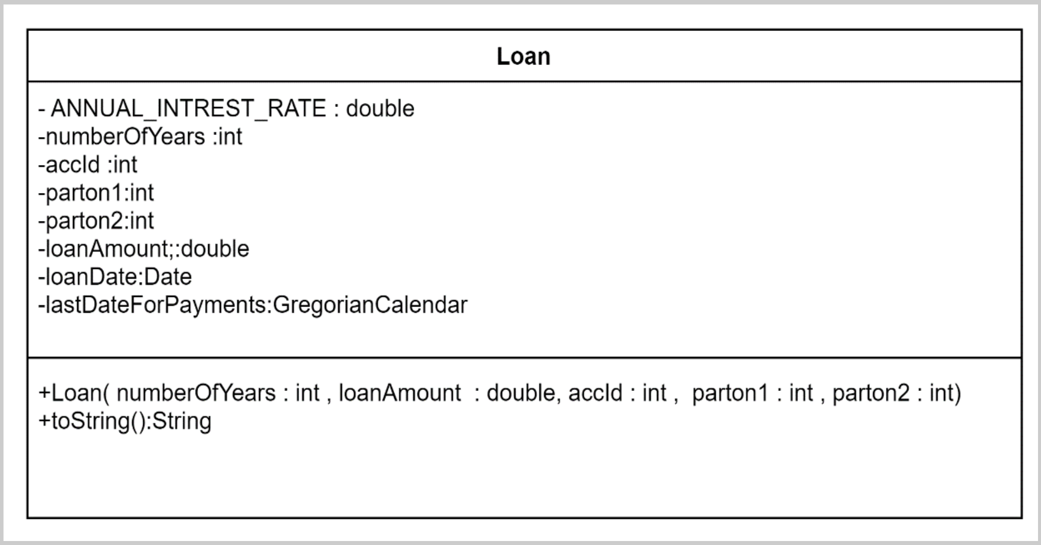
- Represnts an operation .



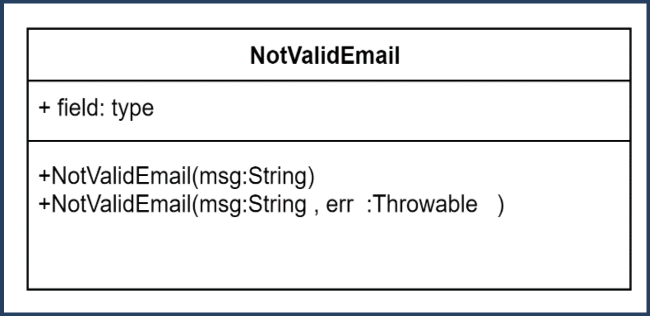
- This represents the information of an account .



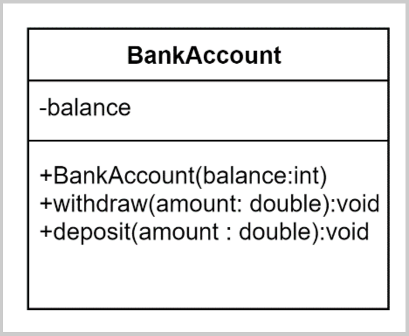
- Represents the information of a customer .


Loan
- ANNUAL_INTREST_RATE : double
-numberOfYears :int
-accId :int
-parton1:int
-parton2:int
-loanAmount;:double
-loanDate:Date
-lastDateForPayments:GregorianCalendar
+Loan(numberOfYears : int , loanAmount : double, accId : int , parton1 : int , parton2 : int)
+toString():String

- Represents the information of the loaan


NotValidEmail
+ field: type
+NotValidEmail(msg:String)
+NotValidEmail(msg:String , err :Throwable)

- This custom exception to be thown when there a problem in the email

	• This to store the value of balance of a customer and making withdrawing and despositing on it

BankAccount
-balance
+BankAccount(balance:int)
+withdraw(amount: double):void
+deposit(amount : double):void

The point of each relationship:

1. **ATMachine extends Bank:**
2.