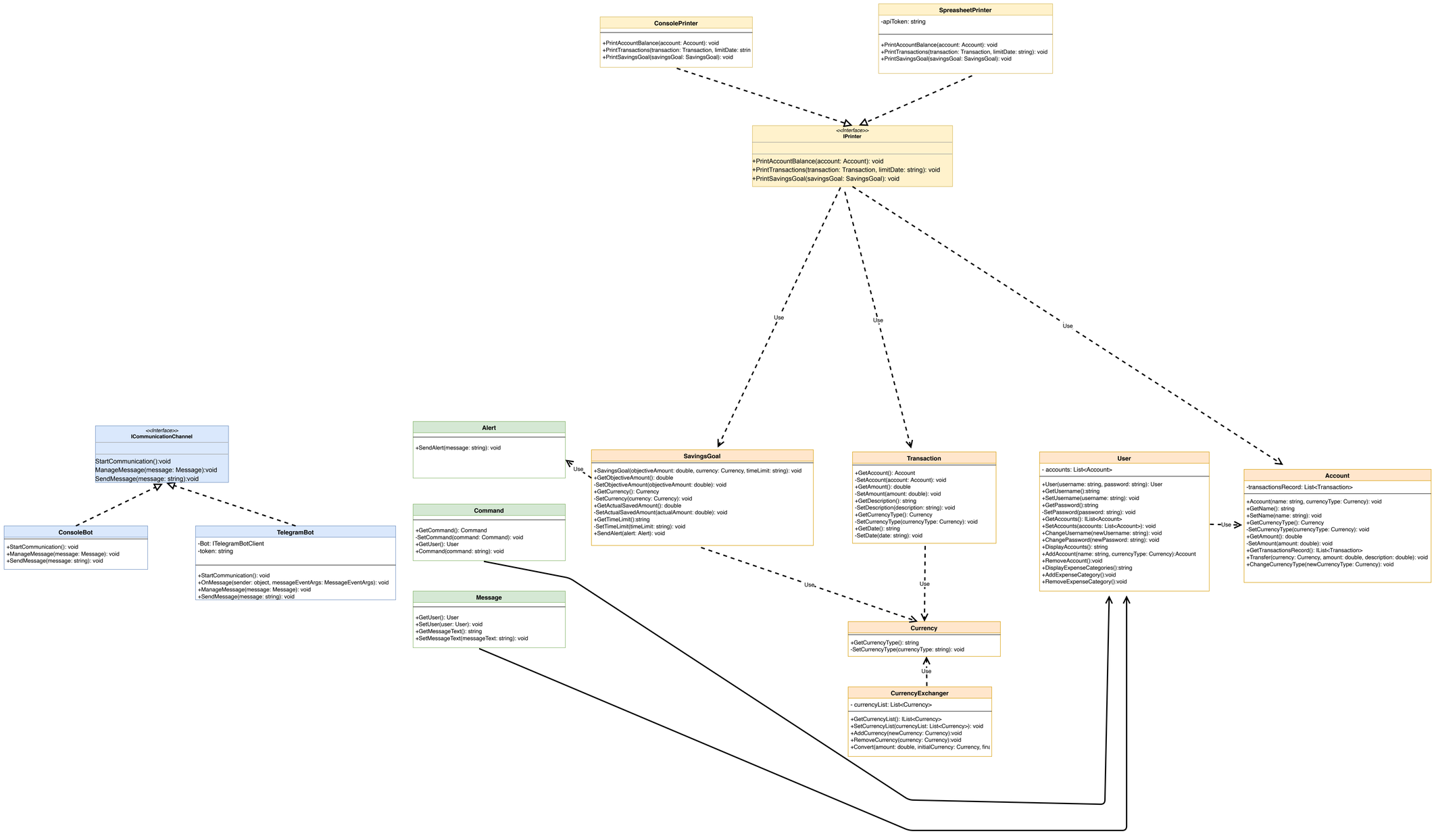
ConsolePrinter
+PrintAccountBalance(account: Account): void
+PrintTransactions(transaction: Transaction, limitDate: strin
+PrintSavingsGoal(savingsGoal: SavingsGoal): void
SpreasheetPrinter
-apiToken: string
+PrintAccountBalance(account: Account): void
+PrintTransactions(transaction: Transaction, limitDate: string): void
+PrintSavingsGoal(savingsGoal: SavingsGoal): void
<<Interface>>
IPrinter
+PrintAccountBalance(account: Account): void
+PrintTransactions(transaction: Transaction, limitDate: string): void
+PrintSavingsGoal(savingsGoal: SavingsGoal): void
Use
Use
Use
<<Interface>>
ICommunicationChannel
StartCommunication():void
ManageMessage(message: Message):void
SendMessage(message: string):void
ConsoleBot
+StartCommunication(): void
+ManageMessage(message: Message): void
+SendMessage(message: string): void
TelegramBot
-Bot: ITelegramBotClient
-token: string
+StartCommunication(): void
+OnMessage(sender: object, messageEventArgs: MessageEventArgs): void
+ManageMessage(message: Message): void
+SendMessage(message: string): void
Alert
+SendAlert(message: string): void
Use
Command
+GetCommand(): Command
-SetCommand(command: Command): void
+GetUser(): User
+Command(command: string): void
Message
+GetUser(): User
+SetUser(user: User): void
+GetMessageText(): string
+SetMessageText(messageText: string): void
SavingsGoal
+SavingsGoal(objectiveAmount: double, currency: Currency, timeLimit: string): void
+GetObjectiveAmount(): double
-SetObjectiveAmount(objectiveAmount: double): void
+GetCurrency(): Currency
-SetCurrency(currency: Currency): void
+GetActualSavedAmount(): double
-SetActualSavedAmount(actualAmount: double): void
+GetTimeLimit():string
-SetTimeLimit(timeLimit: string): void
+SendAlert(alert: Alert): void
Transaction
+GetAccount(): Account
-SetAccount(account: Account): void
+GetAmount(): double
-SetAmount(amount: double): void
+GetDescription(): string
-SetDescription(description: string): void
+GetCurrencyType(): Currency
-SetCurrencyType(currencyType: Currency): void
+GetDate(): string
-SetDate(date: string): void
User
- accounts: List<Account>
+User(username: string, password: string): User
+GetUsername():string
+SetUsername(username: string): void
+GetPassword():string
-SetPassword(password: string): void
+GetAccounts(): IList<Account>
+SetAccounts(accounts: List<Account>): void
+ChangeUsername(newUsernamne: string): void
+ChangePassword(newPassword: string): void
+DisplayAccounts(): string
+AddAccount(name: string, currencyType: Currency):Account
+RemoveAccount():void
+DisplayExpenseCategories():string
+AddExpenseCategory():void
+RemoveExpenseCategory():void
Use
Account
-transactionsRecord: List<Transaction>
+Account(name: string, currencyType: Currency): void
+GetName(): string
+SetName(name: string): void
+GetCurrencyType(): Currency
-SetCurrencyType(currencyType: Currency): void
+GetAmount(): double
-SetAmount(amount: double): void
+GetTransactionsRecord(): IList<Transaction>
+Transfer(currency: Currency, amount: double, description: double): void
+ChangeCurrencyType(newCurrencyType: Currency): void
Use
Use
Currency
+GetCurrencyType(): string
-SetCurrencyType(currencyType: string): void
Use
CurrencyExchanger
- currencyList: List<Currency>
+GetCurrencyList(): IList<Currency>
+SetCurrencyList(currencyList: List<Currency>): void
+AddCurrency(newCurrency: Currency):void
+RemoveCurrency(currency: Currency):void
+Convert(amount: double, initialCurrency: Currency, fina