Low Level Design

<u>Project:</u> Discord Focus Bot

<u>Team Name:</u> The Discordians

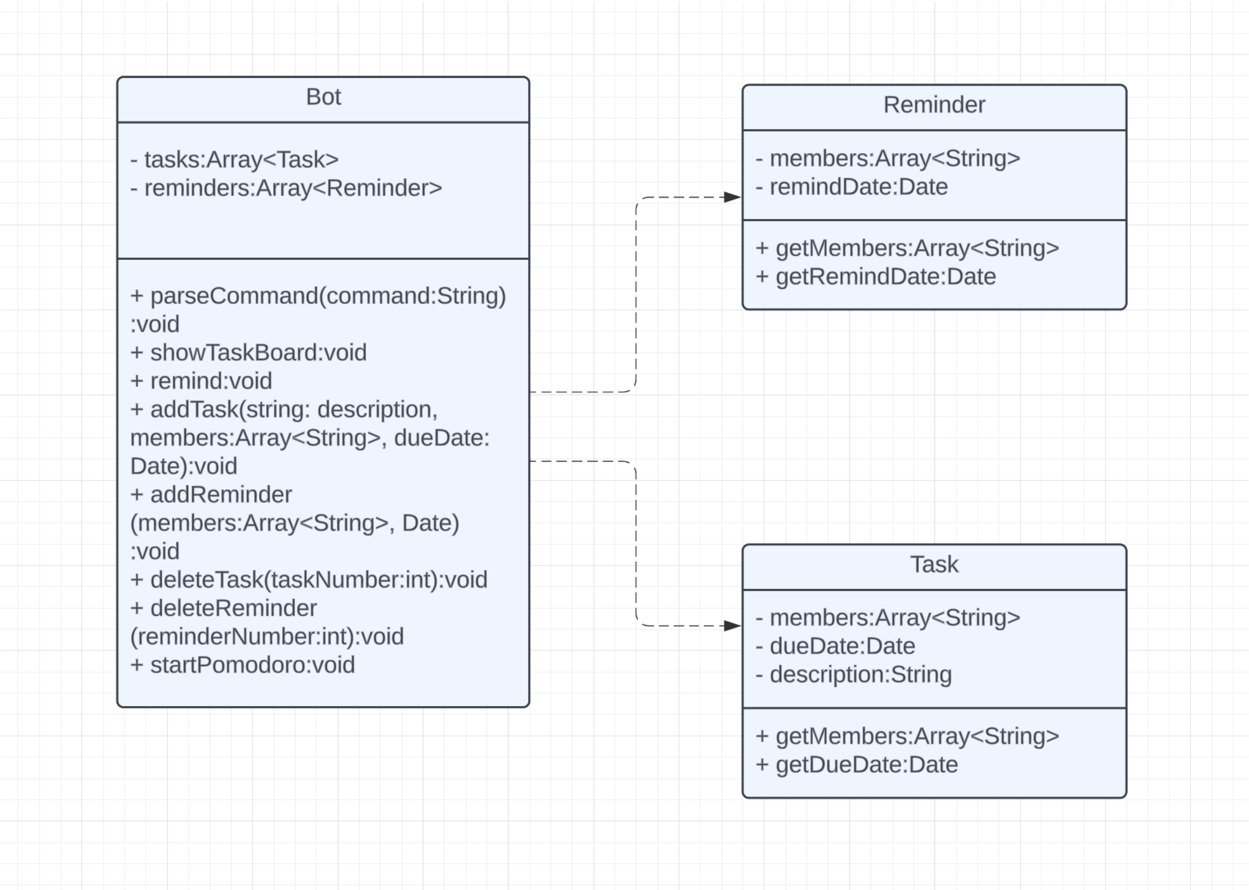
<u>Team Members:</u> Shreya Aravindan, Justin Turkiewicz, Geethika Abhilash, John Byman, Sheril George and Arav Tripathi

The bot can be represented as an object that is allowed to execute various methods. The bot can have a list of task objects and a list of reminder objects. This would be helpful to represent using a structural model of low level design because this model deals with the composition of classes and objects.

Pseudocode

```
Class Bot {
List<Task> tasks
List<Reminder> reminders
Int pomodoroCount
Int PomodoroDuration
Int break duration
parseCommand(userInputtedCommand) {
    Tokenize userInputtedCommand into command name and arguments
    Switchcase for different command names available:
    Case: addTask
    // create a new task and add it to the bot's task array
        tasks.add(new Task(argument 1, argument 2, argument 3))
    Case: addReminder
    //create a new reminder and add it to the bot's reminder array
    Reminders.add(new Reminder (argument1)
    Case: startPomodoro
    StartPomodoro()
    Case: stopPomodoro
    StopPomodoro()
    // to handle invalid command
    print(“Invalid command”)

    .......
}
Function startPomodoro(){
//if pomodoro timer is not on
If (!isPomodoroOn) {
IsPomodoroOn = true
PomodoroCount++;
// then we start the timer
StartTimer(pomodoroDuration)
} else {
//this is when the pomodoro timer is on
Print(“Pomodoro is currently in session.”)
}
}
Function stopPomodoro(){
//if pomodoro timer is on
If (isPomodoroOn) {
IsPomodoroOn = false
StopTimer()
// then we start the break session
Print (“Pomodoro session ended. Take a break!”)

StartTimer(breakduration)

} else {

//this is when the pomodoro timer is off

Print(“no pomodoro session to stop”)

}

}



Main() {

    Asynchronously accept user inputs and send reminders

    While (true) {

    ParseCommand(userInputtedCommand)

    //If reminder time is reached then

        SendReminder().

    //If git action was taken then

    If (gitActionTaken()) {

    SendGitnotification

        }

    }

}
```

UML Diagram


Bot
- tasks:Array<Task>
- reminders:Array<Reminder>
+ parseCommand(command:String) :void
+ showTaskBoard:void
+ remind:void
+ addTask(string: description, members:Array<String>, dueDate: Date):void
+ addReminder (members:Array<String>, Date) :void
+ deleteTask(taskNumber:int):void
+ deleteReminder (reminderNumber:int):void
+ startPomodoro:void
Reminder
- members:Array<String>
- remindDate:Date
+ getMembers:Array<String>
+ getRemindDate:Date
Task
- members:Array<String>
- dueDate:Date
- description:String
+ getMembers:Array<String>
+ getDueDate:Date