```
Problem Statement: Design a Job Searching Platform

-    Internet/Website:
    * Behavior: establishWebService, monsterResponse
    * Data: Internet Connection

-    Website User/Job Seeker: searchJob, applytoJob, selectSalaryRange, selectLocation, uploadResume, reachouttoJobPoster
    * Data: User Password, User Email Address, User Phone, User Address, User Resume, Employment History

-    Website Login:
    * Behavior: logintoWebsite, createnewAccount, forgotPassword, validate(username, password)
    * Data: Username, Email Address, Password, Phone Number

-    Job:
    * Behavior: isAvailable, hasapplicationDeadline, jobSummary, saveJobforLater
    * Data: Job Availability, Job Summary, Application Deadline, Save Job

* Sequence of Flow - Invoke Objects with Behaviors:

User User
Platform Monster

//Validate if the entered URL is valid
IF url.isValid = true
    boolean connection = monster.establishWebService()
        IF connection.isValid = true
            //Authenticate the user if credentials are valid
            boolean authenticated = user.logintoWebsite(userUsername, userPassword)
            monsterResponse response = monster.validate(username, password)
            //Monster validates the right user and allows the user to proceed further
            IF authenticated.isValid = true && monster.validate = true
                userResponse response = user.searchJob(job)
                userResponse response = user.selectSalaryRange()
                //Monster website requests user location
                monsterResponse response = allowLocationDetection()
                userResponse response = user.selectLocationManually() || user.allowLocationDetection()
                //Monster will list the Job Summary and Salary Range based on closest search results
                string jobArray[] = new job[jobType, jobSummary]
                //Job Summary would enlist the salary range and the minimum experience required for the Job
                monsterResponse response = monster.showListofJobTypes(jobArray)
                //User gets to select the job and enter the job page or save the job for later
                userResponse response = user.selectJob(job) || user.saveJobforLater(job)
                IF user.selectJob(job) = true
                    userResponse response = user.applytoJob(job)
                    //Monster validates the Job availablity and verifies the job application deadline with system date and time
                    monsterResponse response = monster.validate(job.isAvailable) && monster.validate(job.hasapplicationDeadline>system.date)
                    IF response.isValid = true
                        monsterResponse response = monster.requestResumeFromUser()
                        userResponse response = user.uploadResume(resume)
                        userResponse response = user.reachouttoJobPoster(companyWebsite)
                    ELSE
                        monsterResponse response = print("Job Application is no longer valid")
                    END
                ELSE
                    monsterResponse response = print("Kindly select a Job from List")
                END
            ELSE
                //User has the ability to create new account, recover forgotten password
                monsterResponse response = user.createnewAccount || monsterResponse response = user.forgotPassword
            END
        ELSE

        END
ELSE

END
```