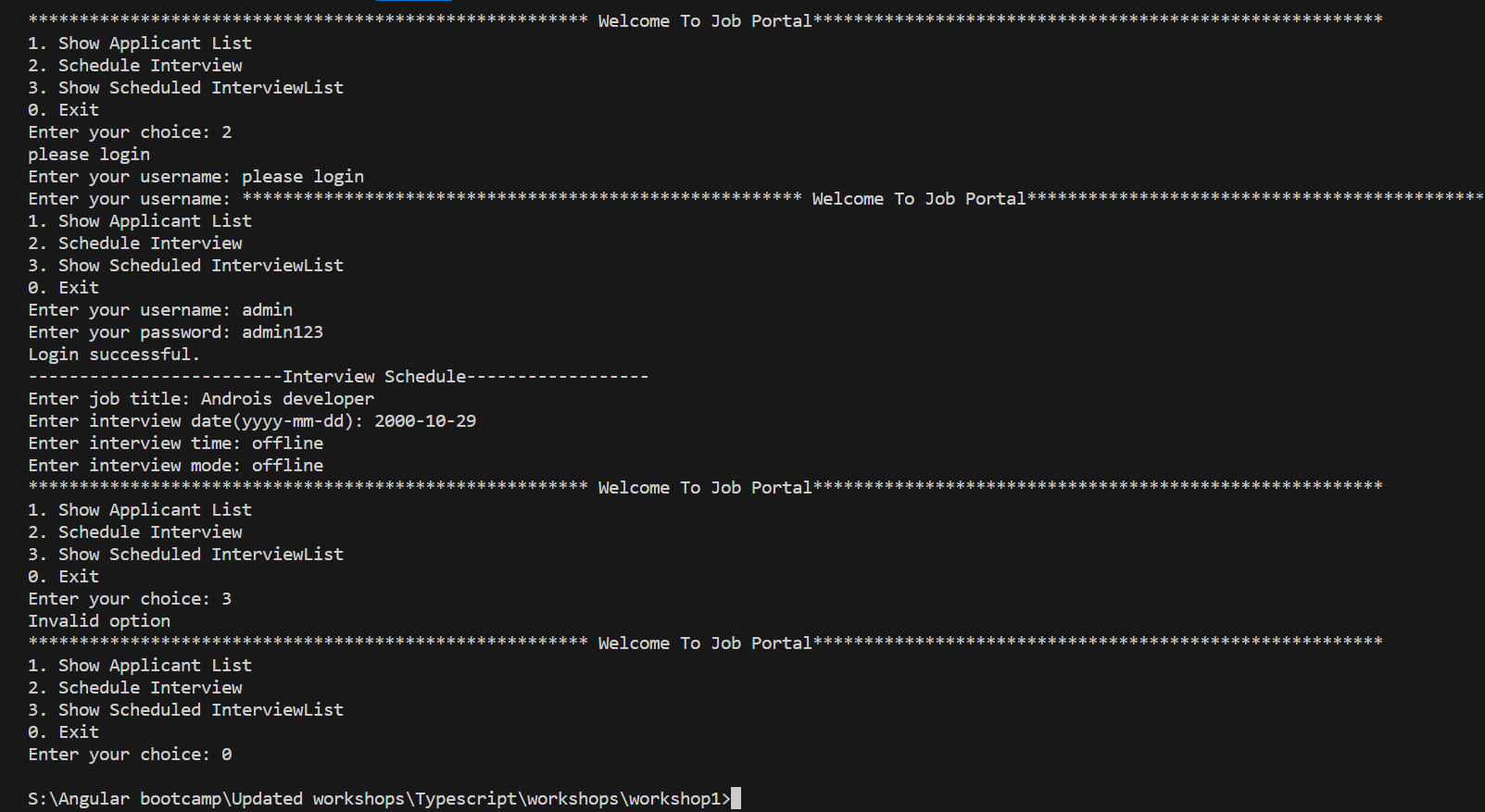
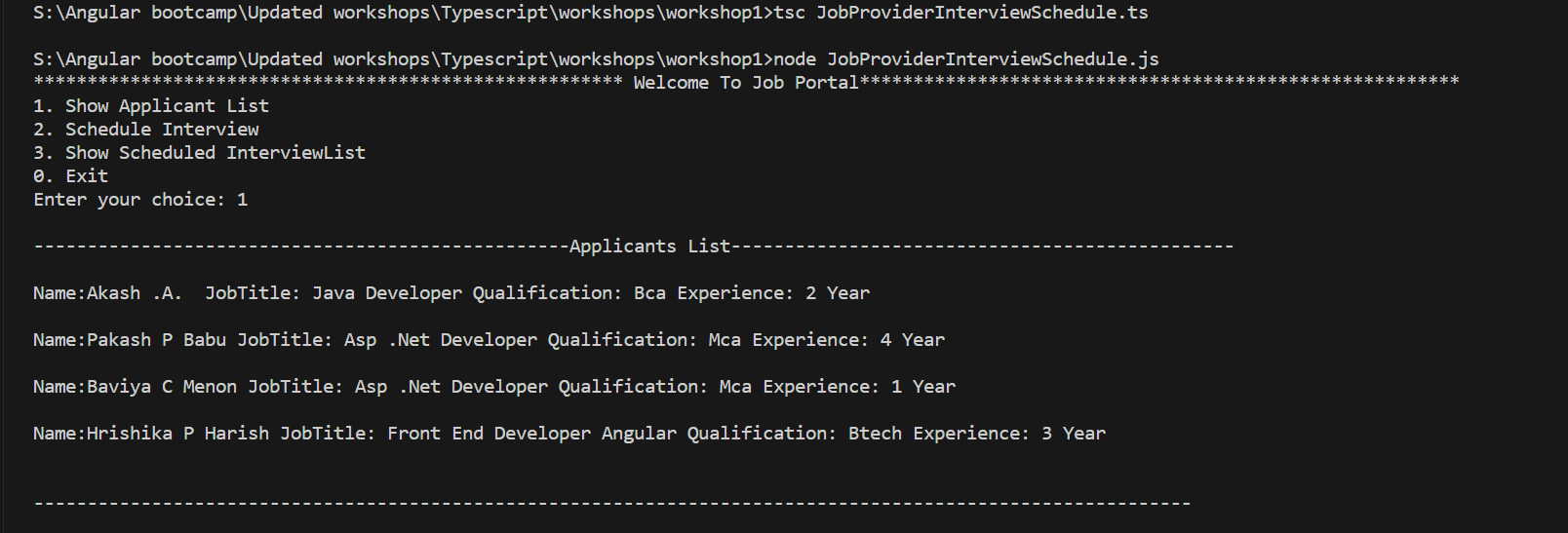
**Angular Typescript Chapter1- Students Handout**

**Introduction**

Angular is a powerful and popular open-source framework for building dynamic web applications. Developed and maintained by Google, it leverages TypeScript, a statically-typed superset of JavaScript, to enhance code quality and maintainability. Angular provides a comprehensive set of tools and features for building single-page applications (SPAs) with ease, including a robust component-based architecture, dependency injection, and a rich ecosystem of libraries and packages. With its declarative approach to building user interfaces, Angular simplifies the development process and promotes code reusability. It also supports two-way data binding, making it efficient for handling user interactions. Whether you're a seasoned developer or just starting out, Angular and TypeScript offer a powerful combination to create modern, responsive, and scalable web applications.

**Workshop-1**



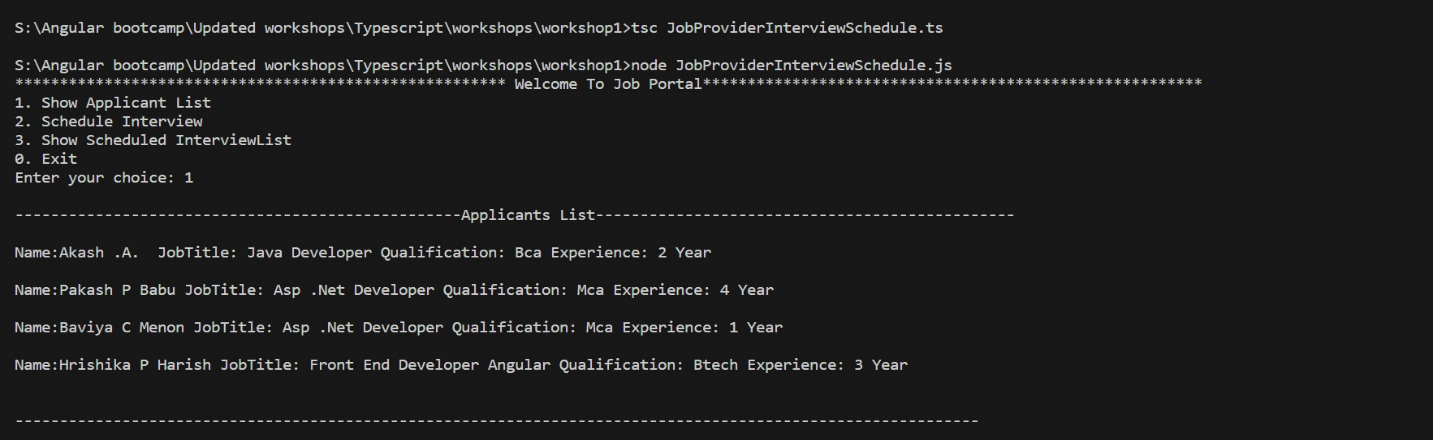
This code is a beginner-friendly command-line program designed to facilitate the management of job interviews and applicants. It utilizes the readline library to establish a user interface through the command line and imports two essential classes, "Applicants" and "Interview," from external files. The program offers a menu with various options, such as displaying the list of applicants, scheduling job interviews, viewing the scheduled interview list, and exiting the program. These functionalities are organized within the "JobProvider" class, which includes methods for menu presentation, user input handling, and action execution based on user choices.

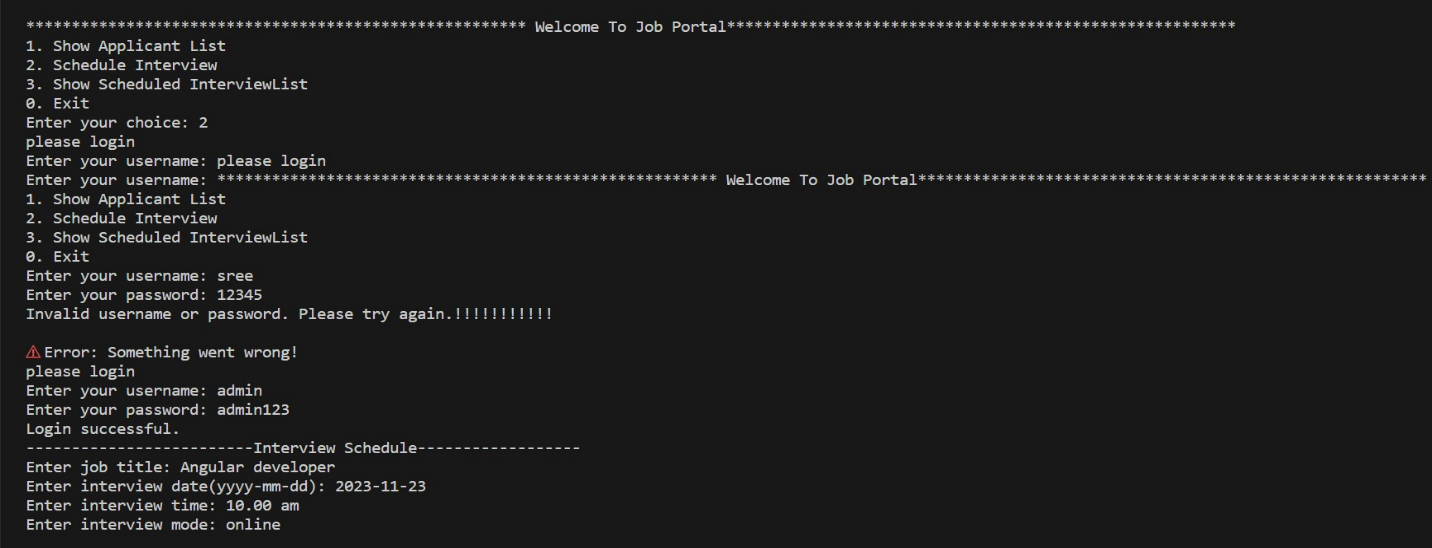
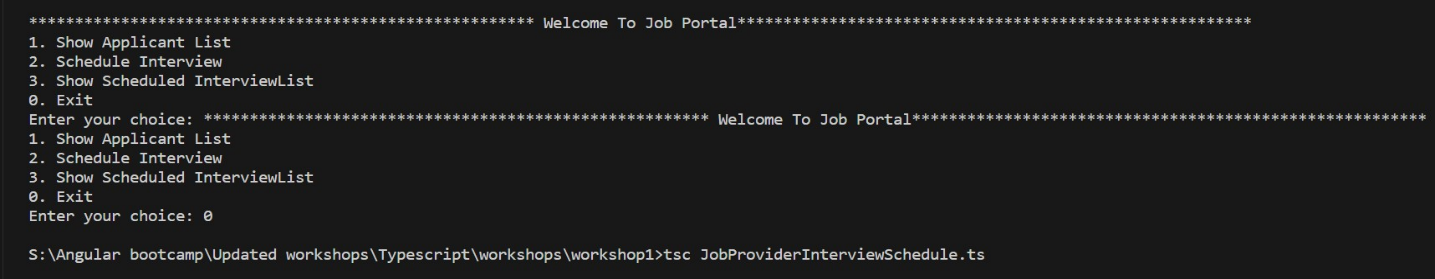
The "interviewList" serves as an array to store information about scheduled interviews, encompassing details like job titles, interview dates, times, and interview modes (online or offline). It starts with two sample interviews added during the "interviewList" method. The "showInterviewList" method is responsible for presenting the scheduled interview details to the user, and the "applicantList" method generates an array of applicant information, comprising names, job titles, qualifications, and experience, which is then displayed to the user.

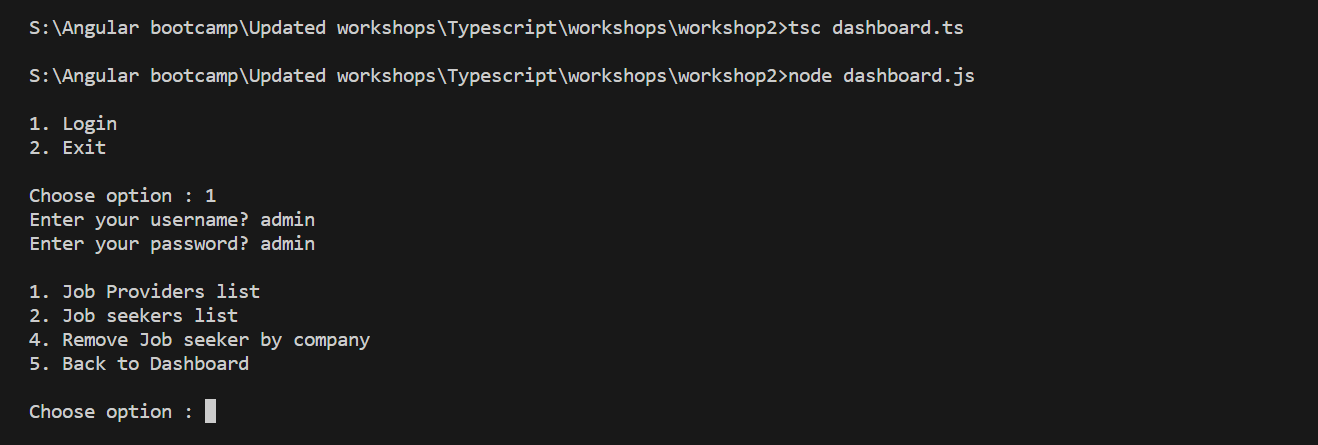
Users can schedule new interviews using the "scheduleInterview" method, entering job-specific information such as job title, date, time, and interview mode. The code includes an authentication mechanism to check if the user is logged in, executed by the "auth" method. If not, it prompts the user to log in via the "login" method, which verifies the user's identity with a predefined admin username and password. Upon successful login, the "localStorage" variable is set to "admin," granting access to interview scheduling.

The code begins by creating an instance of the "JobProvider" class and calling the "showMenu" method, initiating the primary menu for user interaction with the application. In summary, this code provides a straightforward and accessible command-line interface for managing job applicants and interview scheduling, ensuring that users must log in as an admin with the username "admin" and password "admin123" to access and utilize these features effectively.

**Note:** Add Show scheduled interview list option to this program

**Expected Output**

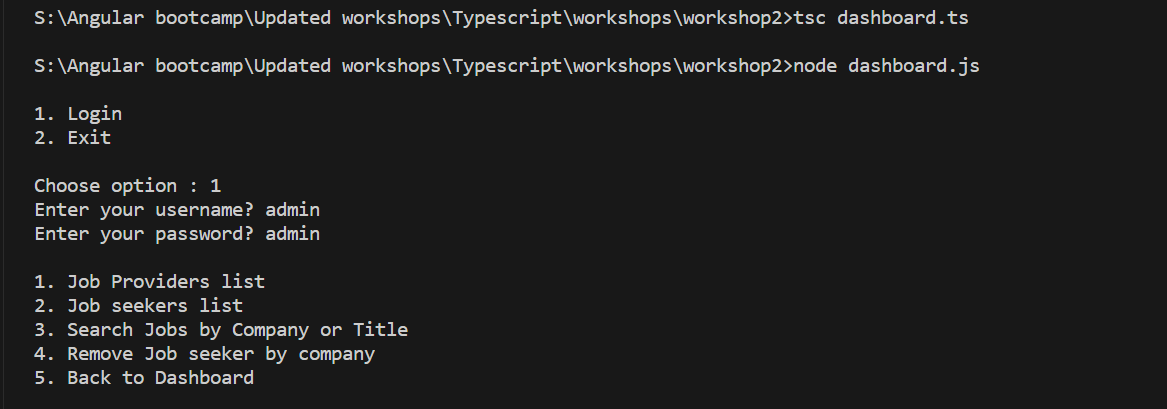
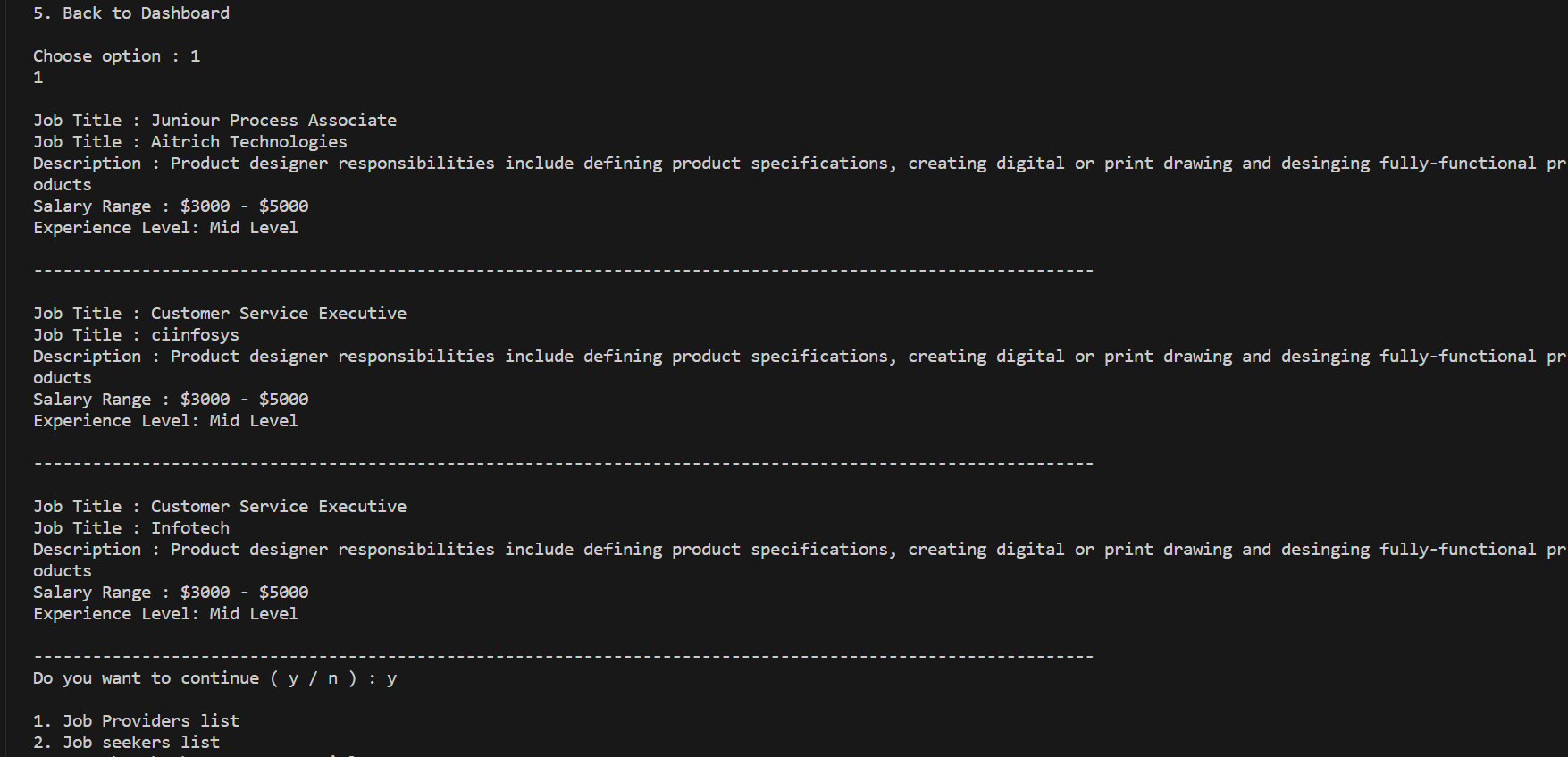
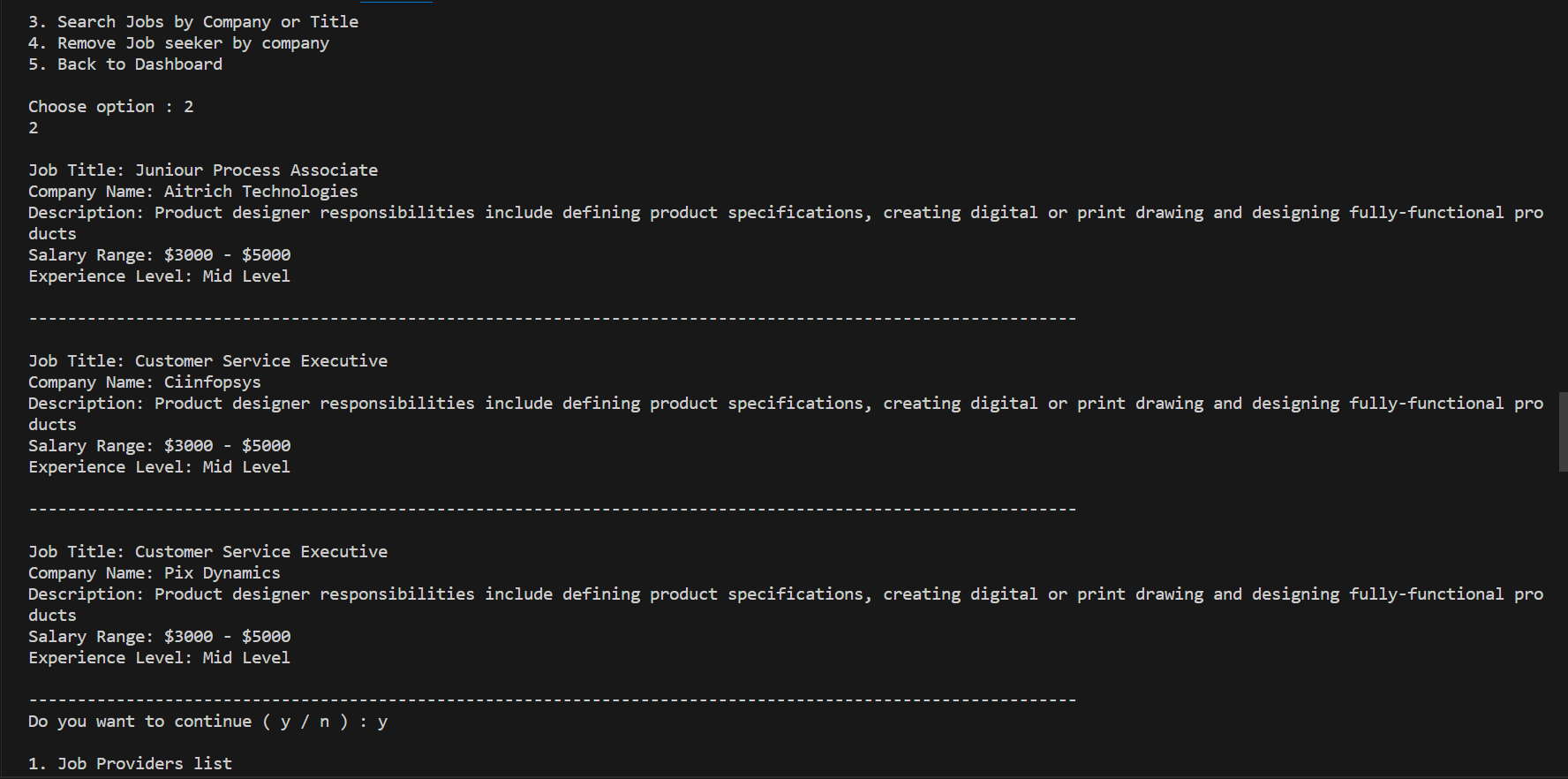
**Workshop-2**

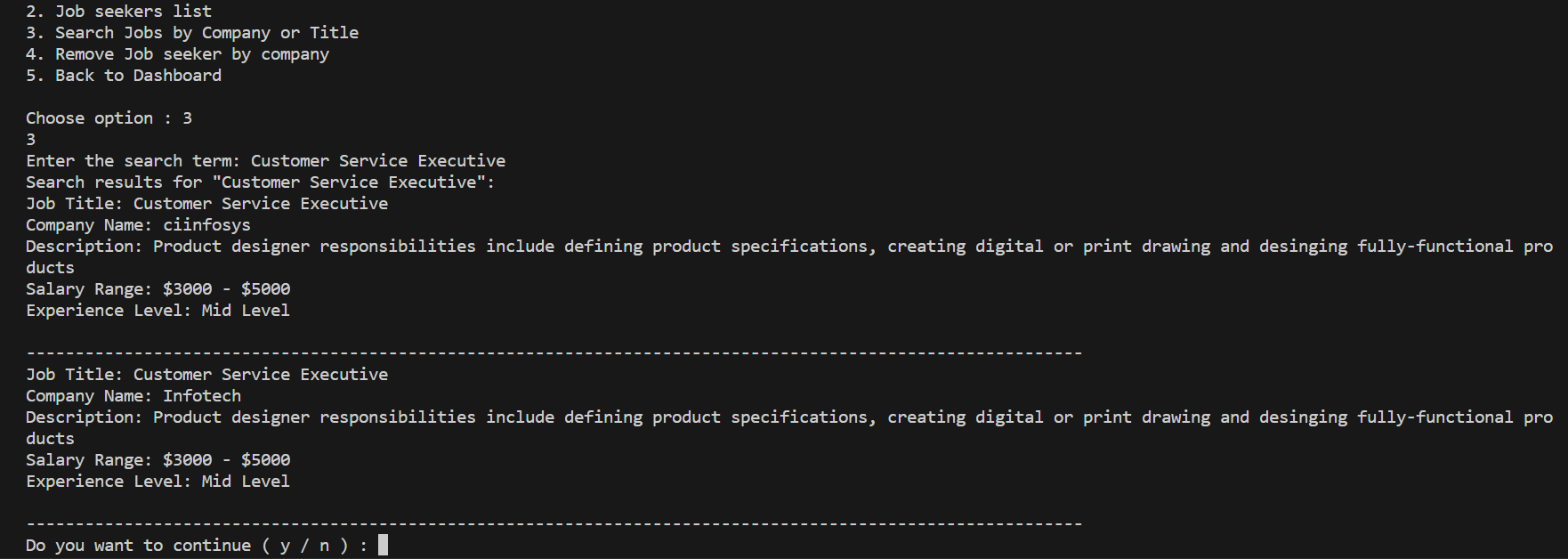
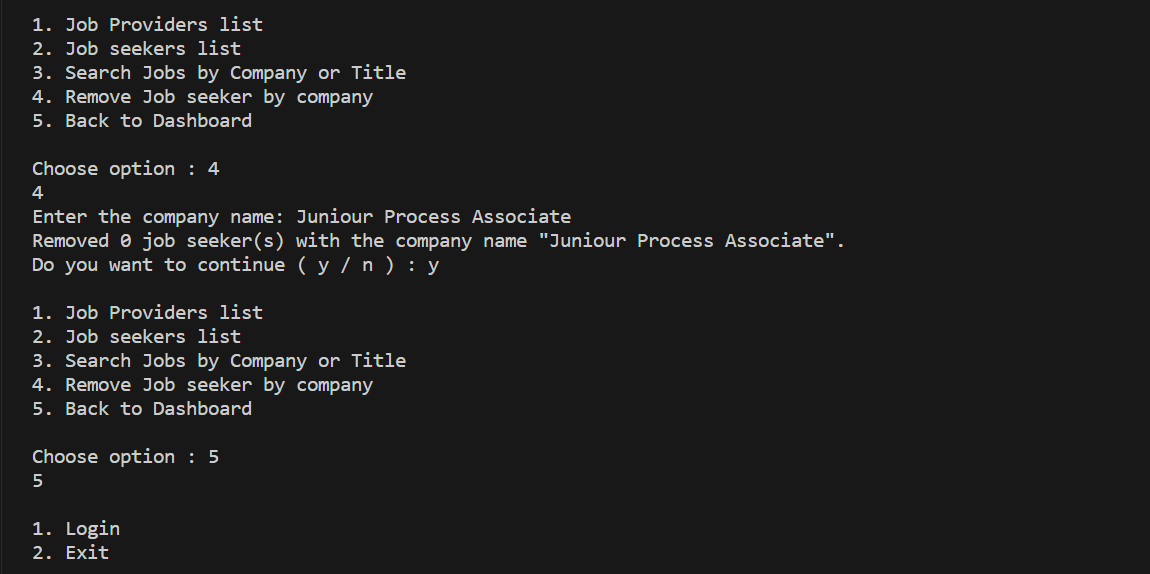
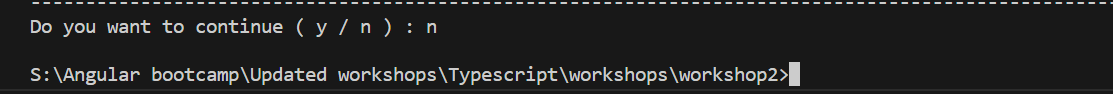


**Note:** Add Search jobs by company or title option to this program

This TypeScript-based command-line application consists of several classes and interfaces that work together to create a basic job management system. The central component is the `Dashboard` class, which serves as the application's main menu. Users can choose to log in or exit. Upon login, the `Authentication` class collects and validates user credentials. If the login is successful, the main menu displays options for managing job listings and job seekers. The program leverages the `Job` class to list job listings and perform searches, while the `JobseekerManager` class handles job seeker information and removal by company name. The code illustrates the use of classes, interfaces, and user input handling, making it a valuable example for students learning TypeScript and command-line application development.

**Expected Output**





**Workshop-3**

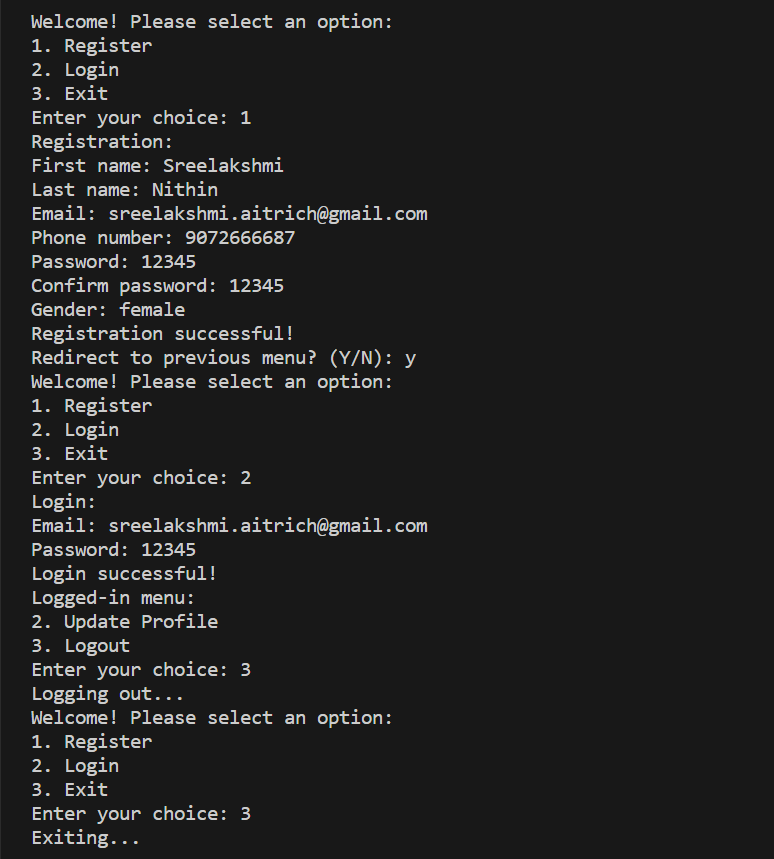
This TypeScript code creates a simple command-line interface for user registration and login. It uses the 'readline' library to interact with the user. The program offers three main functionalities: registration, login, and logged-in actions.

1. Registration: Users can provide their personal information like first name, last name, email, phone number, password, and gender. This data is stored in an array of user objects.

2. Login: Users can enter their email and password, and the program checks if the provided credentials match any registered user. If a match is found, the user is considered logged in, and they can view or update their profile.

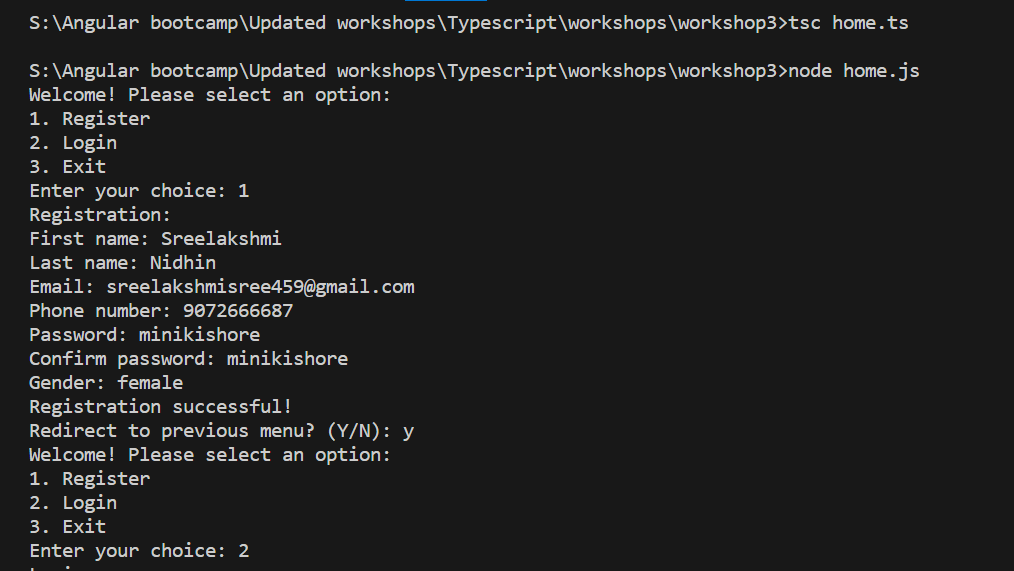
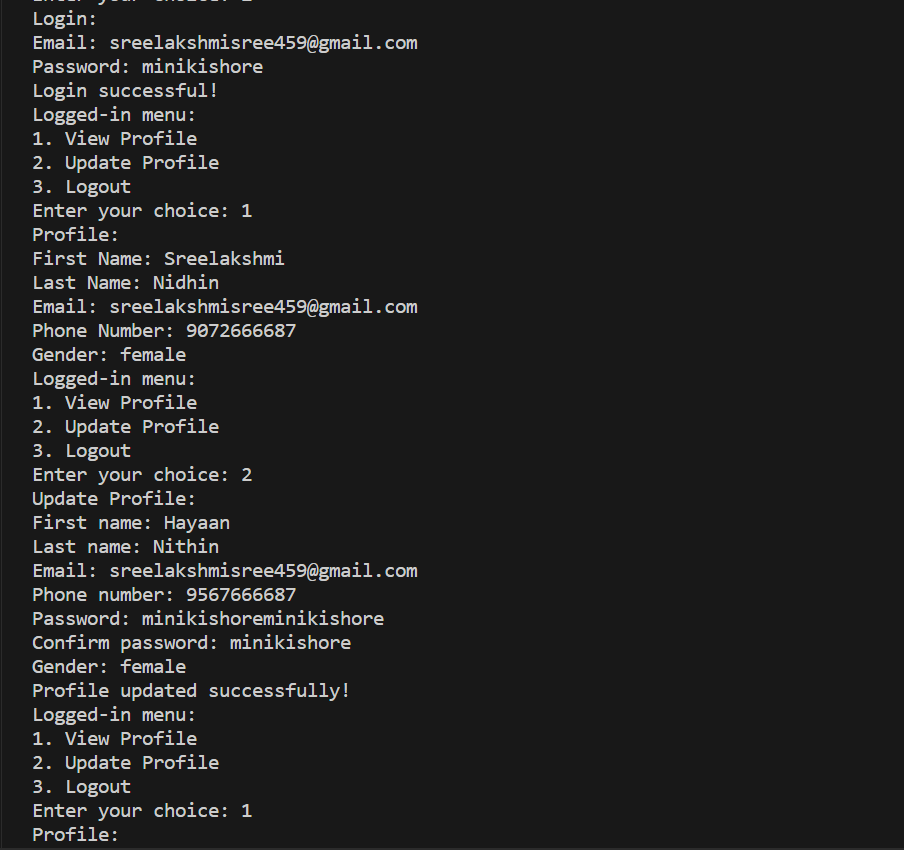
3. Logged-In Menu: When logged in, users have options to view their profile, update it, or log out. The profile information is displayed based on the user object fetched during login.

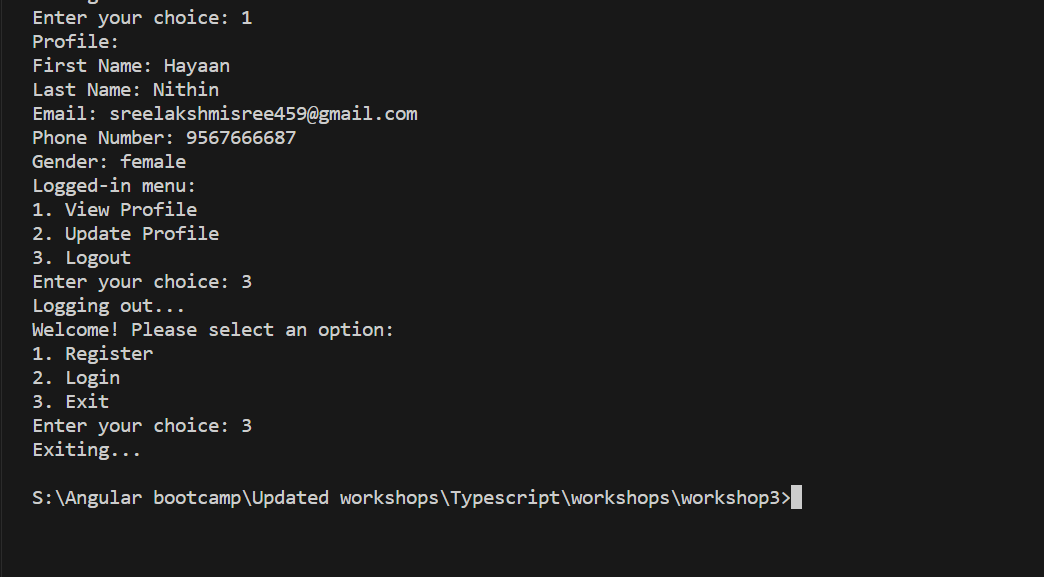
The code maintains a 'loggedInUser' variable to track the currently logged-in user, and it provides simple navigation and error handling. When the user is done, they can exit the program. This code serves as a basic example of user authentication and interaction with a command-line interface.



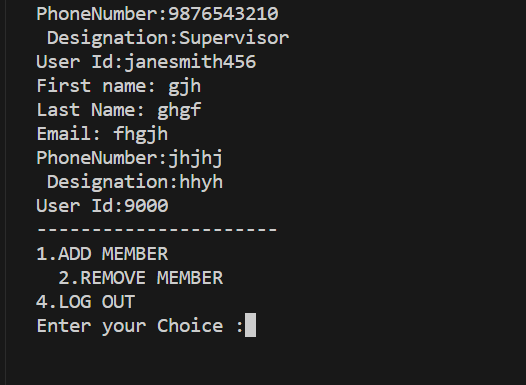
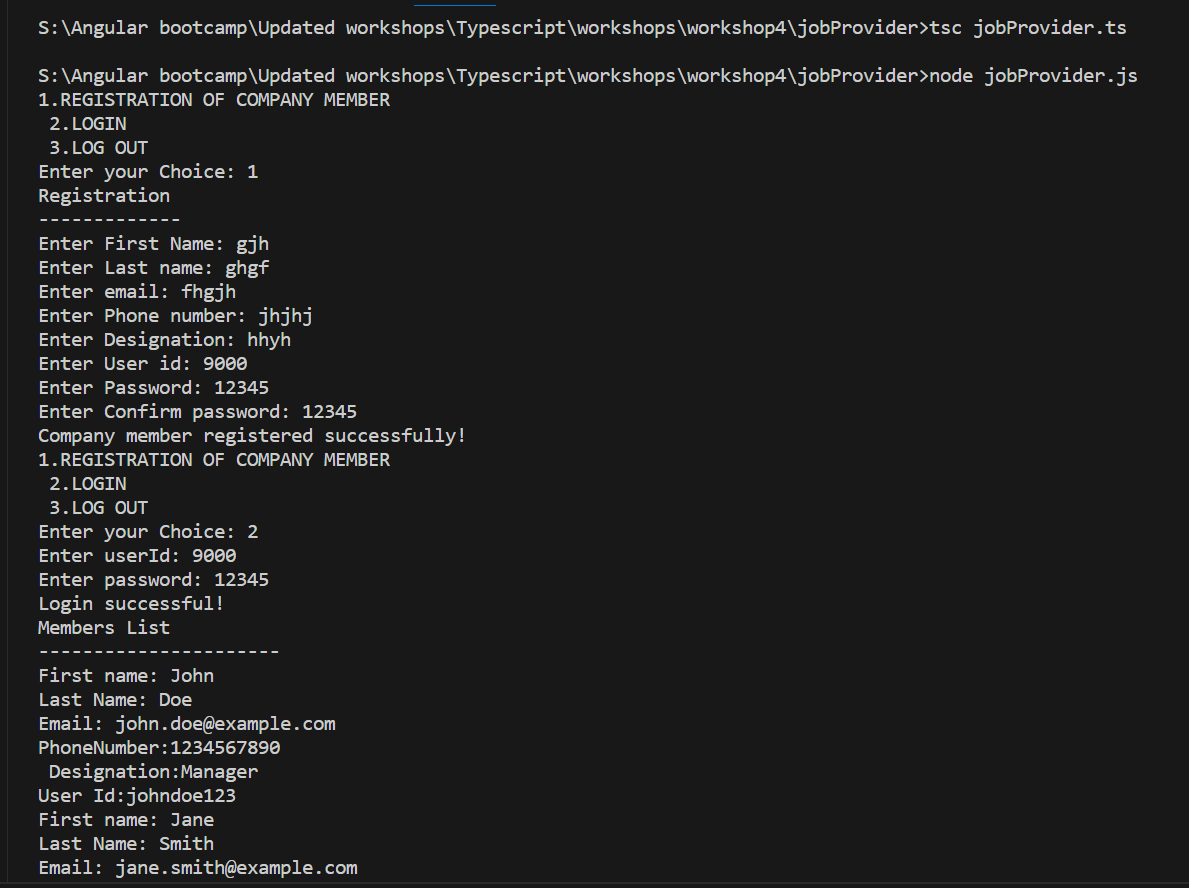
**Note:** Add View profile option to this program.

**Expected Output**





**Workshop 4**

****

This TypeScript program creates a command-line interface for managing company members, allowing users to register new members, log in, and perform various member-related actions. Users can register new company members by providing personal information, including their first name, last name, email, phone number, designation, user ID, and password. The program checks for existing user IDs and password confirmation. Registered company members can log in, view a list of members, and access a submenu to add, remove, find, or log out members. The code uses an array to store company member data and provides error handling for various scenarios, making it a practical example for students learning about command-line interfaces and user management systems.

**Note:** Please add find member case in this program.

**Expected Output**

