Experiment – 1 b: TypeScript

Name of Student	**Anuprita Mhapankar**
Class Roll No	**D15A_28**
D.O.P.	**30/01/2025**
D.O.S.	**06/02/2025**
Sign and Grade	

AIM : To study Basic constructs in TypeScript.

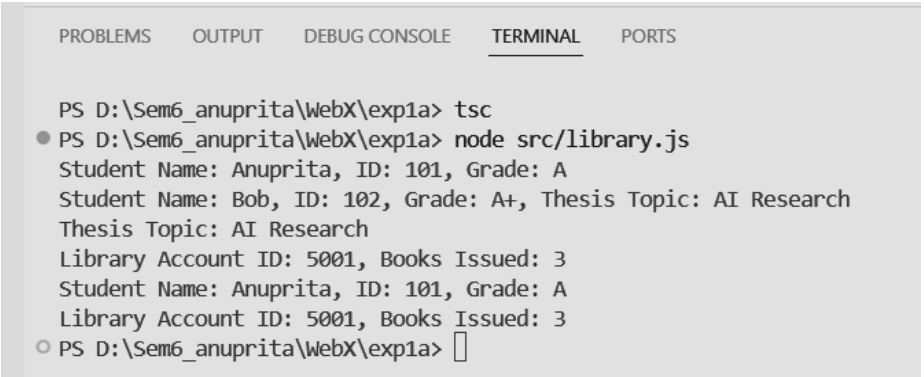
OVERVIEW OF TASKS PERFORMED:

The experiment demonstrates **inheritance, method overriding, and composition** in Java and **interface implementation** in TypeScript. A **Student** class was extended by **GraduateStudent**, overriding `getDetails()`, while **LibraryAccount** was independently associated with Student, showcasing composition. In TypeScript, an `Employee` interface was implemented by **Manager** (with `department`) and **Developer** (with `programmingLanguages`), both overriding `getDetails()`. Instances were created to observe behavior.

GITHUB LINK - https://github.com/Anuprita2022-26/WebX_Exp1b

OUTPUT

(a) Student and GraduateStudent with Composition



This screenshot displays the output of the TypeScript program implementing **inheritance**

and composition. The program first prints details of a **Student** and a **GraduateStudent**, demonstrating method overriding and inheritance. Then, it prints the **Thesis Topic** of the GraduateStudent separately. The next lines show details of a **LibraryAccount** associated with a student, demonstrating composition. Finally, it displays a combined output of both **Student and LibraryAccount**, showcasing how composition works.

(b) Employee Management System

This screenshot displays the output of the **Employee Management System** program. It shows details of an **Employee interface**, with two classes: **Manager** and **Developer**, implementing it. The output displays the details of a **Manager** named Alex, including their ID, role, and department. It also shows the details of a **Developer** named Anuprita, including their programming languages. The output is generated after running `node src/employee.js` in the terminal.

CONCLUSION

This experiment demonstrated the fundamental concepts of TypeScript, such as inheritance, method overriding, and composition through the implementation of **Student** and **GraduateStudent** classes. Instead of using inheritance, composition was demonstrated by linking **LibraryAccount** with **Student**, emphasizing flexibility in design.

Furthermore, the **Employee Management System** utilized interfaces to enforce structure and type safety, highlighting the advantages of TypeScript in maintaining scalable and well-organized code. Overall, this experiment reinforced the benefits of TypeScript's object-oriented capabilities, improving code readability, reusability, and reliability.