**Understanding of React Native Navigation I:** In this hos07\_01, it was demonstrated how to use React Native and React Navigation to build a cross-platform mobile application called CitiesApp, developed using Expo. The app utilizes both stack-based navigation and bottom tab navigation to organize screen flow effectively. Bottom tab navigation is used for top-level views, which is ideal for apps with clearly defined sections, while stack navigation is employed to simulate native mobile transitions by pushing screens on top of each other. The app includes key components such as `AddCity` and `Cities`, where users can fill out a form to add new cities and view a scrollable list of saved cities. Tapping on a city navigates the user to a detail screen. The user interface is supported by reusable components like `CenterMessage` to gracefully handle empty state scenarios. Additionally, essential packages like UUID generators and navigation dependencies were installed to enable full functionality. The basic UI and navigation setup were completed, with further enhancements planned for the next development module.

**Understanding of React Native Navigation II:** This module builds upon the CitiesApp by introducing a new City component that enables location management within each city. To support hierarchical navigation, a Native Stack Navigator is integrated alongside the existing bottom tab navigation. The `City.js` screen allows users to add and view specific locations tied to a selected city. Data such as cities and their locations are managed using shared state and props, ensuring consistency across screens. The Cities screen is updated to pass selected city data for detailed navigation, and the AddCity component is refactored to improve UUID handling. The `App.js` file is restructured to combine both navigators within a single `NavigationContainer`, enabling seamless screen transitions and centralized data management. With React Navigation managing context and navigation flow, users can now move smoothly from the Cities tab into detailed City screens that include form inputs for location entries. Overall, the enhanced navigation structure supports a more dynamic, multi-view experience while preserving data integrity across the app.