

# Assignment 01 – To-Do List App

Name: Shahd Ahmed Mahmoud Owais

Helwan University – Faculty of Engineering  
(FEHU)

---

## 1. Project Overview

The purpose of this assignment was to create a simple **To-Do List application** using **React Native** and **Expo**. The application allows users to add tasks, view them in a scrollable list, and provides a clear interface for goal management.

### **Key Components Implemented:**

- **Text:** For displaying the title of the app.
- **TextInput:** For user input of tasks/goals.
- **Button:** To add items to the list.
- **FlatList:** To display the list of goals and handle scrolling.

## 2. App Logic

The app uses **React Native state management** to handle user interactions.

## States:

1. `enteredGoalText`: stores the text typed in the `TextInput`.
2. `courseGoals`: stores the array of goals added by the user.

## Functions:

- `addGoalHandler()`: triggered on button click. Adds the input text to the `courseGoals` array and clears the `TextInput`.

## 3. Design

The design is clean and user-friendly. Consistent colors and custom fonts enhance usability.

### Color Palette:

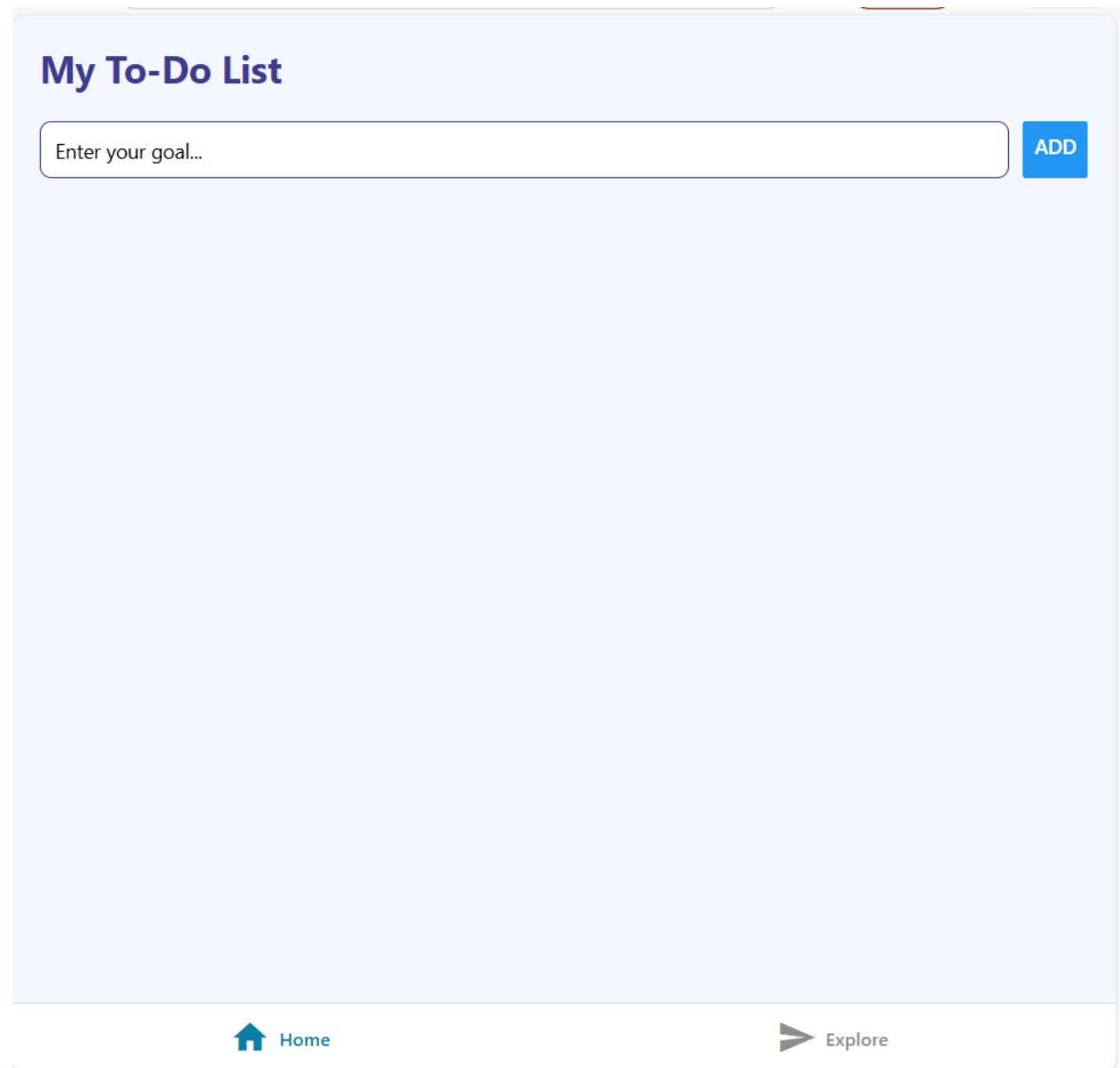
- **Primary Color:** #4a90e2 (blue)
- **Secondary Color:** #ffffff (white)
- **Accent Color:** #f5a623 (orange)

### Fonts:

- **Heading Font:** Playfair Display (Bold)
- **Body Font:** DM Sans (Regular)

Fonts added using [Expo Fonts Documentation](#).

## Screenshots:



## My To-Do List

[ADD](#)

study lecture 1

study lecture 2

study lecture 3

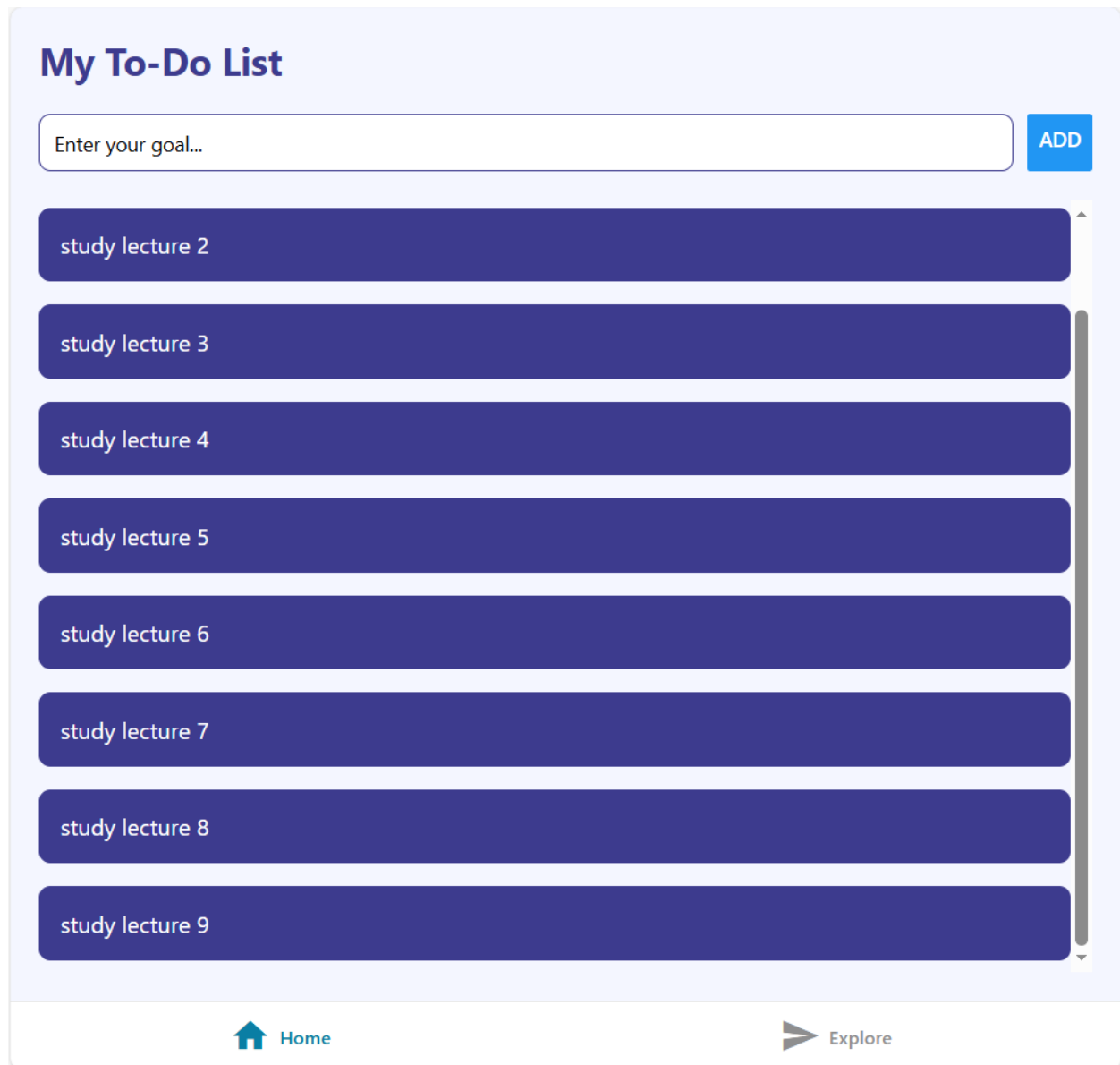
study lecture 4

study lecture 5

study lecture 6

 Home

 Explore



## 4. Code & Deployment

- **Expo Snack Link:** [Talking Blue Fudge](#)
- **GitHub Repository Link:** [To-Do List Mobile App](#)

## Note for Instructor

Please note that the **node\_modules** folder has been **removed** from this repository to reduce its size.

Before running the project, make sure to install the dependencies by running:

```
npm install
```

## 5. Conclusion

The app successfully demonstrates:

- State management in React Native
- Event handling via buttons
- Dynamic rendering using FlatList
- Custom fonts and color consistency

It fulfills all the requirements of **Assignment 01** and provides a simple yet elegant user interface.