Adaptavist Assessment Task
Your task is to develop a working program that accomplishes the following objectives:
1. Input Handling: The program should accept a Git repository URL as input.
2. Repository Cloning: Upon receiving the URL, the program should clone the specified Git repository to a local directory.
3. Modification: After cloning, make a demonstrable change to the repository. This could be as simple as modifying a README file or any other file within the repo.
4. Push Changes: Finally, push the changes back to the original repository.
Technical Requirements:
● Framework and Language: The project must be developed using React and TypeScript, specifically, with the NextJS framework.
● Design System: Incorporate an existing Design System of your choice to ensure consistency and efficiency in design implementation.

Clone and Modify Git Repository Application

Program Implementation Workflow

This application provides a simple and intuitive UI to clone, modify, and push changes to a GitHub repository. The workflow is as follows:

npm run dev command to run the application

1. **Initial State**:
    - The UI displays a text field for the GitHub URL.
    - The "Clone Repository" button is enabled.
    - The "Modify and Push" button is disabled.
2. **Validation**:
    - If the user enters an invalid URL or leaves the field empty, an error message is displayed.
3. **Cloning**:
    - If the specified cloning path already contains the project, the program replaces it with the new clone folder.
    - The local cloning path is `/tmpGitClonePath`.
4. **Post-Cloning**:
    - Upon successful cloning, a text field for entering a modification message is displayed.
    - The "Modify and Push" button is enabled.
5. **Modification and Push**:

- The modification text field is optional. If the user provides a modification message, the program modifies a file with the provided content.
- If no message is entered, a default modification message is added to the file.
- The current implementation modifies the README file.
- The program then commits and pushes the changes back to the original repository.

Assumptions

- GitHub credentials are managed and stored in the GitHub credential manager for cloning a GitHub URL.
- For SSH URLs, the passkey is already set.

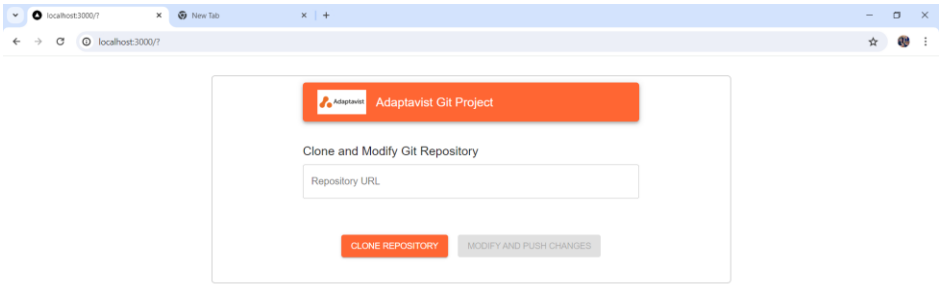
Program Installation Steps

6. **Prerequisites**:
    - Node.js installed on machine.
    - Git installed and configured on machine.
    - GitHub credentials are set to allow the program to push changes to the repository.
7. **Setup**:
    - Next.js application setup.
    - Install the necessary packages.
8. **Components**:
    - Created `CloneForm.tsx` component that handles the form components, including the text fields for the GitHub URL and modification message, along with their validations on the UI.
9. **API Routes**:
    - Created reusable API routes `Clone.ts` and `Push.ts` to handle Git functions.
    - `Clone.ts` handles the cloning functionality after providing the GitHub URL.
    - `Push.ts` handles modification, commit, and push changes to the original repository.
    - Used `NextApiRequest` and `NextApiResponse` to handle Git functionalities.

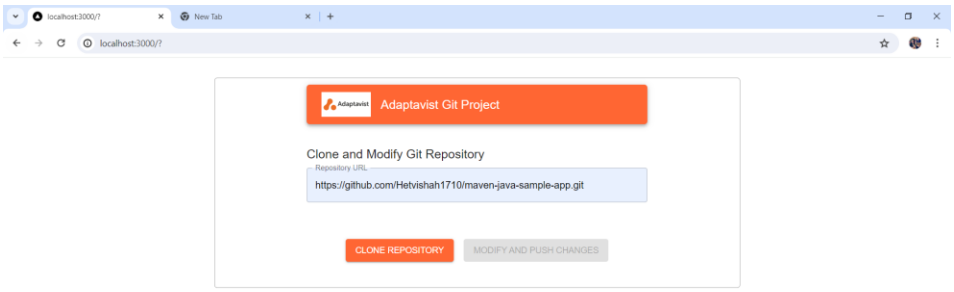
Design System

The application utilizes Material-UI for consistent and efficient UI styling. Key components and custom styling include:

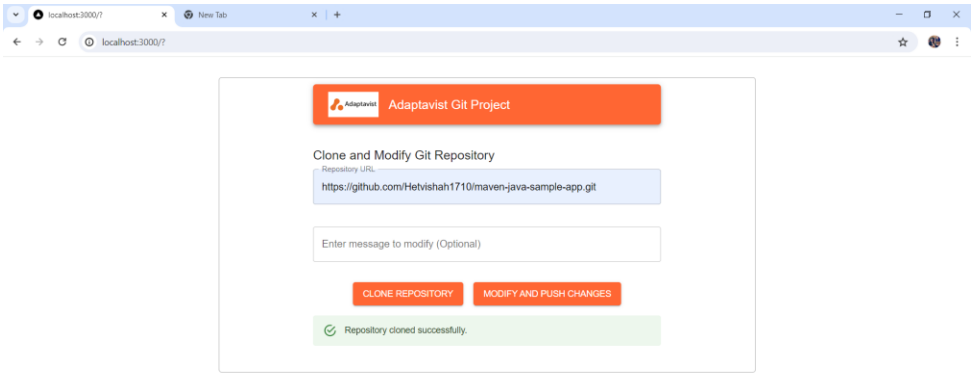
- **Custom Styling**:
  - `Theme.ts` handles custom styling.
- **UI Components**:
  - Added a logo in the navigation bar.
  - Used Material-UI components:
    - `TextField`
    - `Button`
    - `Typography`
    - `Container`
    - `Box`
    - `Alert`
    - `FormControl`
    - `FormHelperText`
    - `useTheme`
    - `AppBar`
    - `Toolbar`
- **Results**:



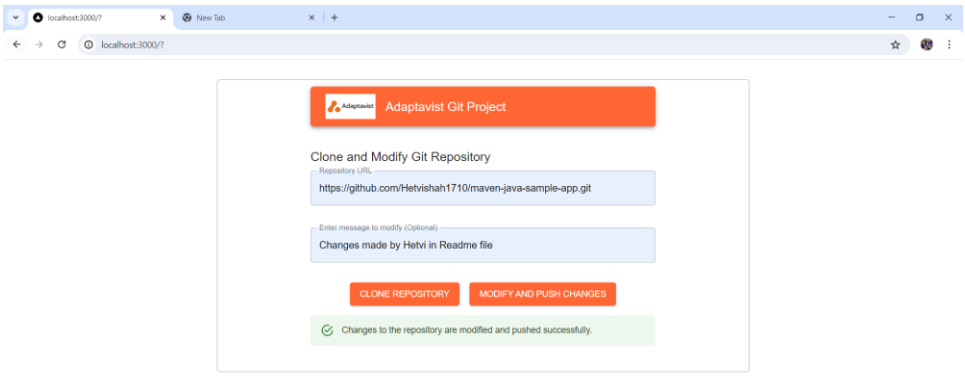
On Load – Initial page UI



Enter GitHub Url



Message textfield is displayed



Push the changes