Home Page

* This is a functional component in React called "Home" that renders a homepage for a task manager app. It contains two sections, one with descriptions and images about what a task manager is and what it's used for, and another with information about the company's top five employees. It also includes links to other pages within the app using React Router's Link component.

Task Manager

* component that manages tasks in columns with statuses "To Do," "In Progress," "In Review," and "Done." It uses Firebase Firestore for data storage and retrieval.
* The component has two useEffect hooks that run once on mount. The first useEffect retrieves a list of employees from the "employees" collection and sets it to state using the setEmployees method. The second useEffect retrieves a list of tasks from the "tasksManager" collection, sets the list of tasks to state using the setTasks method, and then sets four filtered lists of tasks to state based on their status ("To do," "In Progress," "In Review," and "Done").
* The component also has functions that update an employee's completed tasks count when a task is marked as complete (updateEmployee), add a new task to the "tasksManager" collection (addTask), update a task's status (updateTask), update a task's status only (updateStatus), and delete a task from the "tasksManager" collection and state (deleteTask).
* Finally, the component renders an input field to search for tasks by title, a button to open a modal to add a new task, and three columns of tasks in different statuses. The columns of tasks use the filtered lists of tasks from state to display the appropriate tasks based on their status. Each task is rendered as a Tasks component, which is passed several props including the updateTask, updateStatus, and deleteTask functions, as well as the task data itself

**Employees**

* The code is a React component that manages a list of employees using Firebase Firestore. This code is rendering a table of employees using React. It first checks if the filteredEmployees is true. If it is, it maps through the filteredEmployees array and returns a table row for each employee. Each row displays the employee's fullName, email, dateOfBirth, phoneNumber, monthlySalary, and completedTasks.
* If the filteredEmployees array has not been passed in, the code maps through the employees array instead and returns a table row for each employee.
* The table also includes buttons to edit or delete an employee's information. The EditModal component is rendered when the user clicks the edit button for a particular employee. The deleteEmployee function is called when the user clicks the delete button, passing in the employee's id as an argument.
* Overall, this code is rendering a dynamic table of employee data that allows for filtering, editing, and deleting employee information.
* Used Swiper.js that displays the top five employees with the most completed tasks. The component also allows adding, editing, and deleting employees.

**Task Component**

* This code is rendering a single to-do task, including its title, description, assignee, due date, status, and buttons for editing, deleting, and updating the status of the task.
* find() method is used to look up the full name of the employee who is assigned to the task, based on their id.
* The buttons for editing and deleting the task are represented by icons, which are wrapped in button elements with onClick event handlers. When the user clicks the edit button, an EditTaskModal component is rendered, allowing the user to edit the task details. The deleteTask() function is called when the user clicks the delete button, passing in the task's id as an argument.
* The Next Stage button is used to update the status of the task. When the user clicks this button, the updateTask() function is called, passing in the task's id and an object representing the updated status. The code uses conditional statements to check the current status of the task and update it accordingly. If the task is in the done status, the updateEmployee() function is called, passing in the id of the employee who completed the task, and an alert is displayed indicating that the employee completed a new task.
* Overall, this code is rendering a single to-do task with options for editing, deleting, and updating its status, as well as displaying the details of the task, such as the assignee and due date.

Unresolved bugs or unimplemented logic due to time frame

- Refreshing is necessary to view changes between the to-do stage and the in-progress stage.

- react-beautiful-dnd library to implement drag and drop functionality.