# User Management

1. Login page
2. User management page
3. Logged in user, actions tagged with their id/name

# Users Page

1. Add User button => popup to add that user
   1. (text only) User ID (auto-generated by firebase)
   2. Name
   3. Surname
   4. Role
   5. Date Created
   6. Open Tabs
2. Table with users and columns with the data above

# Logging in – initiate active tabs which are linked to the user

# Admin area

1. Previous transactions and tabs
2. Open transactions page/ filter on list

# Main Page:

[+ Create tab] – popup with “Please enter table number”

Logged in as:   
David Jones 1775468

User Management (Admin only) | Log Out

Gin + Tonic | 1

+ Create tab

Table 45

Table 33



Lemonade | 2

Beer | 1

Close Tab

Remove Selected

Change Quantity

Table numbers:

* Add a better central way to edit Table numbers
* Add way to add general open tab No Table

# Portfolio polish

Button to launch an Example state with previous transactions

Auto Logged in as a user with open tabs.

Floating button :hover shows options:

* Switch to admin account
* Back to till user (greyed out if logged in as user already)
* Reset to Example State

Intricate features:

1. Delayed

**Project: Till Management System for Retail Business**

**Objective:** Create a user-friendly till management system for a retail business that allows users to add items to the current tab using buttons, manage multiple tabs, and easily switch between them for item addition/removal.

**Key Features:**

1. Add items to the current tab using buttons.
2. Create and manage multiple tabs.
3. Switch between tabs to add or remove items.
4. Calculate the total bill amount for each tab.
5. Provide options for clearing tabs or finalizing sales.
6. User authentication and access control for employees.

**Frontend:**

* + HTML, CSS, JavaScript
  + JavaScript Framework (Vue.js) for a dynamic user interface.
  + Firebase JavaScript SDK for real-time database integration.

**Backend:**

* + Firebase Authentication for user authentication and access control.
  + Database for storing tab and transaction data.

**Firebase Realtime Database:**

* Use Firebase Realtime Database to store tab information, item lists, and transaction data. This NoSQL database allows you to easily structure and synchronize data in real-time across clients.
* Leverage Firebase's real-time capabilities to automatically update tabs and bills on the user interface as items are added or removed.

**Firebase Authentication:**

* Utilize Firebase Authentication for user authentication, registration, and access control.
* Configure different user roles (cashier, manager, admin) with Firebase Custom Claims to control access to specific features.

**Timeline:**

* Week 1-2: Project Setup and Database Design
* Week 3-4: Frontend Development (UI design and button functionality)
* Week 5-6: Backend Development (Tab management, item addition/removal, bill calculation)
* Week 7-8: User Authentication and Access Control
* Week 9-10: Testing and Debugging
* Week 11-12: Deployment and User Training

**User Stories:**

**1. As a retail employee, I want a secure login system with access control to ensure data privacy and security.**

* Acceptance Criteria:
  + Users should be able to log in with unique credentials.
  + Only authorized employees should have access to the system's features.
  + Admins should be able to manage user roles and permissions.

**2. As a retail employee, I want to create multiple tabs for different customers so that I can manage orders simultaneously.**

* Acceptance Criteria:
  + There should be an option to create a new tab.
  + Each tab should have a unique identifier (e.g., tab name or number).

**3. As a retail employee, I want to switch between tabs to add or remove items from different orders.**

* Acceptance Criteria:
  + There should be a tab selector that allows me to switch between tabs.
  + I should be able to add or remove items from the currently selected tab.

**4. As a retail employee, I want the system to calculate the total bill amount for each tab so that I can provide accurate pricing to customers.**

* Acceptance Criteria:
  + The system should display the total bill amount for the currently selected tab.
  + The bill amount should update dynamically when items are added or removed.

**5. As a retail employee, I want the option to clear a tab or finalize a sale so that I can complete transactions with customers.**

* Acceptance Criteria:
  + There should be options to clear the currently selected tab (remove all items) or finalize the sale (generate a receipt).
  + After finalizing a sale, the tab should no longer be editable.

**6. As a retail employee, I want to add items to the current tab by clicking on buttons so that I can easily build a customer's order.**

* Acceptance Criteria:
  + On the interface, there should be a list of items with buttons to add them to the current tab.
  + Clicking a button should increment the quantity of the corresponding item in the current tab.

**7. As an admin, I want the ability to add and remove employees from the system and assign them specific roles.**

* Acceptance Criteria:
  + Admin users should have access to a user management panel.
  + Admins should be able to add, deactivate, or modify employee accounts.
  + Admins should be able to assign different roles (e.g., cashier, manager) to employees.

**8. As an admin, I want to view sales data and transaction history for auditing and reporting purposes.**

* Acceptance Criteria:
  + The system should maintain a transaction history that includes details of each sale, such as items sold, bill amount, and employee responsible.
  + Admins should be able to generate reports from this data.

These user stories, along with the project plan and technology stack, provide a solid foundation for developing a till management system for the retail business. The project should be implemented iteratively, with regular testing and feedback from users to ensure it meets their needs effectively.