Wall Street Brokering System

CS-232

Team Members

Ahmed Musharaf	2022067
Muhammad Arsal	2022350
Saaim Ali Khan	2022519

INTRODUCTION

The Wall Street Property Brokering is a web-based application designed for listing and managing properties along with their prices. It includes functionalities for users to browse and inquire about properties, as well as for administrators to manage property listings and user inquiries. This documentation provides an overview of the system's features, architecture, and usage.

BUSINESS SCENARIO ANALYSIS

The Wall Street Property Brokering caters to the needs of real estate agencies or property management firms. Key business scenarios include:

Property Management: Allows administrators to add, update, and remove property listings including details such as category, price, type, location, address, and size.

User Management: Enables users to register, browse, and inquire about properties. Users can view property details and contact administrators for further information.

Inquiry Management: Provides functionalities for administrators to manage user inquiries, respond to inquiries, and track communication regarding property listings.

SYSTEM ARCHITECTURE

The system architecture follows a client-server model. The client-side is implemented using Flask, a Python web framework, while the server-side utilizes PostgreSQL for database management.

Key components include:

- Flask: A micro web framework for Python used to handle HTTP requests and responses, along with rendering HTML templates.

- PostgreSQL: A powerful, open-source relational database management system used for storing and managing property listings and user inquiries.

- psycopg2: PostgreSQL adapter for Python used to interact with the PostgreSQL database from the Flask application.

POSTGRESQL FEATURES UTILIZED:

The Wall Street Property Brokering leverages various features provided by PostgreSQL to efficiently manage database operations:

Tables and Data Definition Language (DDL):

Creation of tables to define database structure.

Alteration of table structure.

Deletion of tables when necessary.

Data Manipulation Language (DML):

Insertion of data into tables.

Updating existing data.

Deletion of specific records.

Functions and Stored Procedures:

Creation of user-defined functions to perform specific tasks.

Implementation of stored procedures for executing predefined operations.

Triggers:

Definition of triggers to execute actions in response to database events.

Foreign Key Constraints:

Enforcement of referential integrity between related tables.

Views:

Creation of views to provide customized perspectives of data.

Calling Stored Procedures and Functions:

Invocation of stored procedures and user-defined functions to execute complex logic within SQL queries.

These PostgreSQL features facilitate robust data management and ensure the integrity and efficiency of the Property Listing System's database operations.

DATABASE SCHEMA

The database schema encompasses multiple tables designed to manage diverse entities and their interrelations concerning property listings and user inquiries.

Key tables within this schema comprise:

- adminusers: This table captures comprehensive details regarding administrative users, encompassing vital information such as login credentials, contact details, and branch affiliations.
- products: Housing essential property information, this table includes attributes such as category, price, type, location, address, and size, enabling comprehensive property management.
- customers: Dedicated to user management, this table records pertinent details of users, encompassing registration information and contact details for effective communication.
- inquiries: Serving as a repository for user inquiries, this table stores pertinent data including property IDs, user IDs, inquiry messages, and inquiry status, facilitating seamless communication and resolution of user queries.

By effectively organizing and interlinking these tables, the database schema ensures robust management of property listings and user interactions within the system.

APPLICATION COMPONENTS

The Flask application consists of multiple routes for handling different functionalities:
- User Management: Includes routes for user registration, browsing properties, and sending inquiries.
-Admin Management: Includes routes for administrator login, managing property listings, and responding to user inquiries.
-Property Management: Includes routes for adding, updating, and removing property listings.
-Inquiry Management: Includes routes for viewing, responding to, and managing user inquiries.

USAGE GUIDELINES

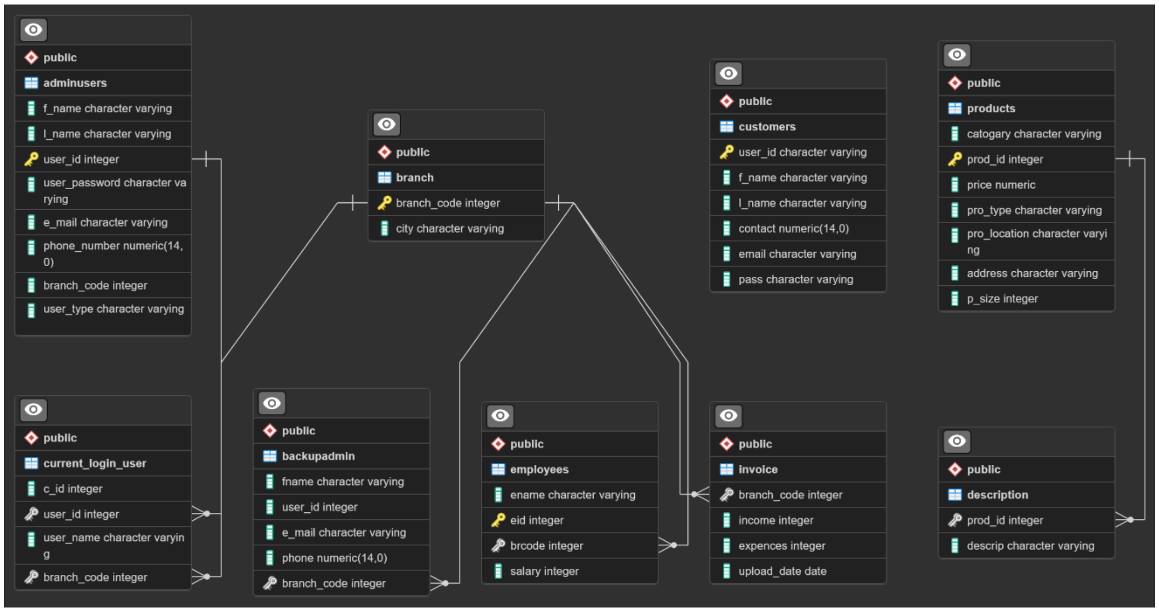
Users can access Wall Street Property Brokering through their web browser and navigate to the respective routes for browsing properties, sending inquiries, and registering. Administrators can log in to manage property listings, respond to inquiries, and track communication with users.

QUERYING AND REPORTING

Administrators have access to querying and reporting functionalities to track property listings and user inquiries. They can query the database for property details, user inquiries, and communication history. Reporting functionalities include generating reports on property availability, user engagement, and inquiry response times.

ER DIAGRAM

The Wall Street Property Brokering is a web-based application designed for listing and managing properties along with their prices. It includes functionalities for users to browse and inquire about properties, as well as for administrators to manage property listings and user inquiries. This documentation provides an overview of the system's features, architecture, and usage.



NORMALIZATION

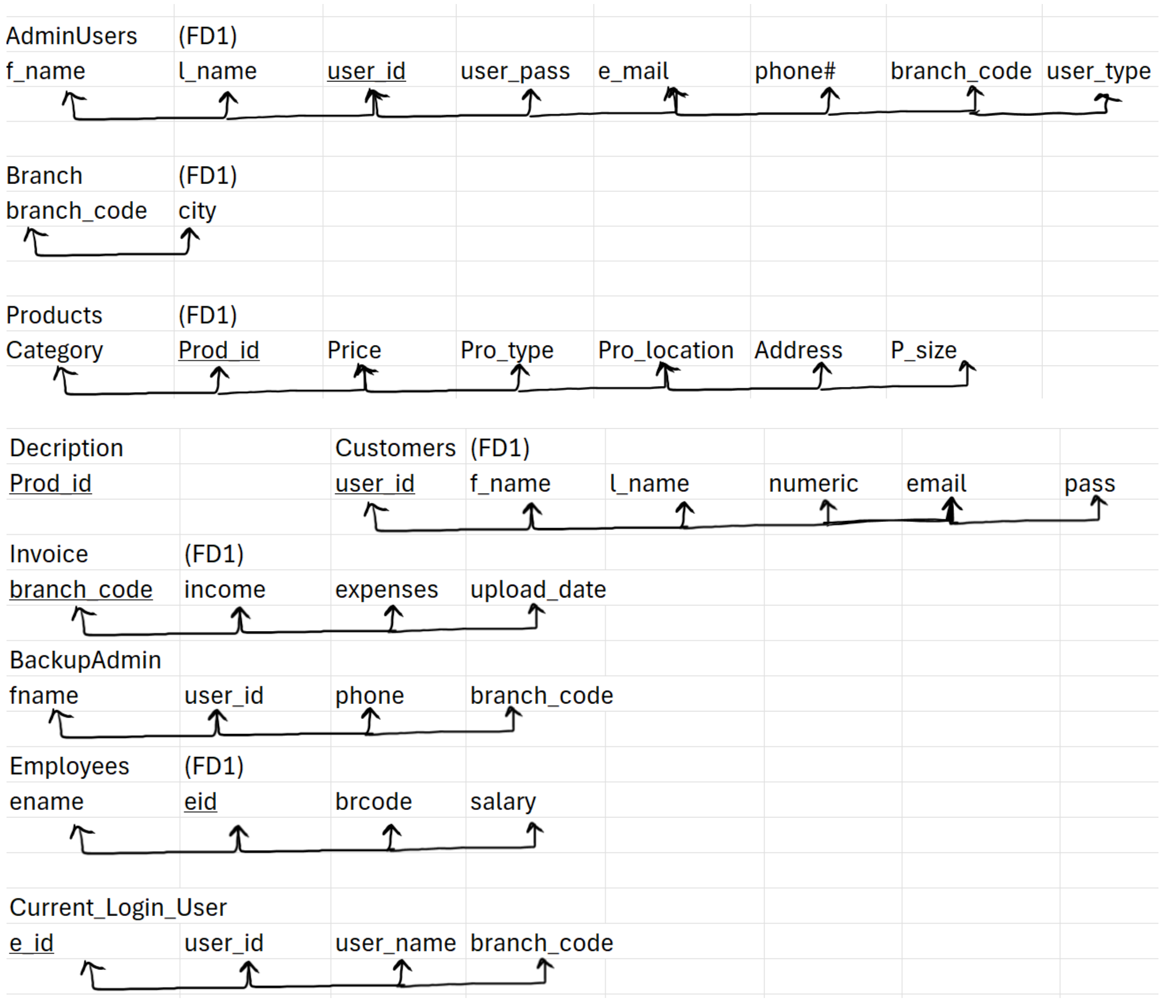
The Wall Street Property Brokering provides a comprehensive solution for managing property listings and user inquiries in the real estate industry. With its user-friendly interface and robust functionality, it streamlines property management processes and enhances user engagement. Future enhancements may include additional features for analytics and reporting.

TABLES:

1) Customers
2) AdminUsers

3) Branch
4) Products
5) Current_Invoice (contains data related to invoices) (trigger function)
6) Current_Login_User(contains data related to logged-in user) (trigger function)
7) Description
8) Employees

1st Normal Form (No multi valued attributes)



AdminUsers	(FD1)						
f_name	l_name	user_id	user_pass	e_mail	phone#	branch_code	user_type

Branch	(FD1)
branch_code	city

Products	(FD1)					
Category	Prod_id	Price	Pro_type	Pro_location	Address	P_size

Decription
Prod_id

Customers	(FD1)				
user_id	f_name	l_name	numeric	email	pass

Invoice	(FD1)		
branch_code	income	expenses	upload_date

BackupAdmin			
fname	user_id	phone	branch_code

Employees	(FD1)		
ename	eid	brcode	salary

Current_Login_User			
e_id	user_id	user_name	branch_code

2nd Normal Form

1) Table is in 1NF
2) No partial dependency

Admin Users							
f_name	l_name	user_id	user_pass	e_mail	phone#	branch_code	user_type
↑	↑	↑	↑	↑	↑	↑	↑

Products						
Category	Prod_id	Price	Pro_type	Pro_location	Address	P_size
↑	↑	↑	↑	↑	↑	↑

Customers	(FD1)				
user_id	f_name	l_name	numeric	email	pass
↑	↑	↑	↑	↑	↑

Invoice	(FD1)		
branch_code	income	expenses	upload_date
↑	↑	↑	↑

Employees	(FD1)		
ename	eid	brcode	salary
↑	↑	↑	↑

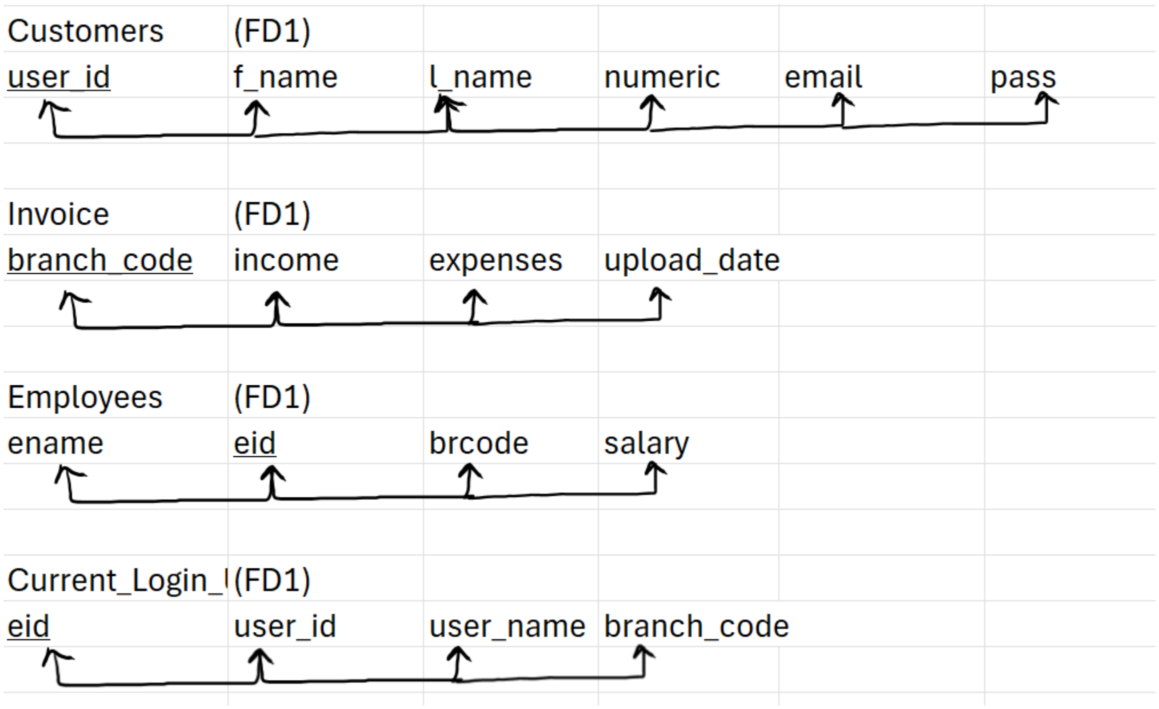
Current_Login_l	(FD1)		
eid	user_id	user_name	branch_code
↑	↑	↑	↑

3rd Normal Form

1) Table should be in 2NF
2) No transition dependencies

Admin Users							
f_name	l_name	user_id	user_pass	e_mail	phone#	branch_code	user_type
↑	↑	↑	↑	↑	↑	↑	↑

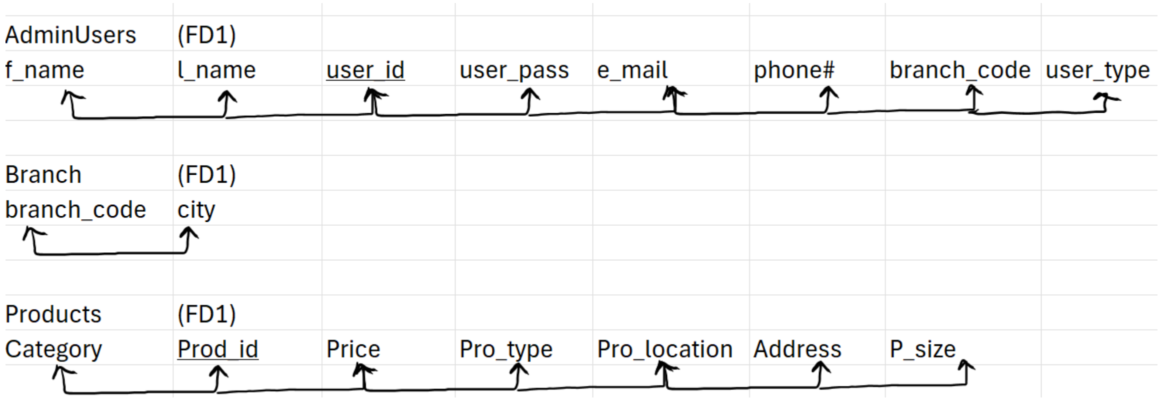
Products						
Category	Prod_id	Price	Pro_type	Pro_location	Address	P_size
↑	↑	↑	↑	↑	↑	↑



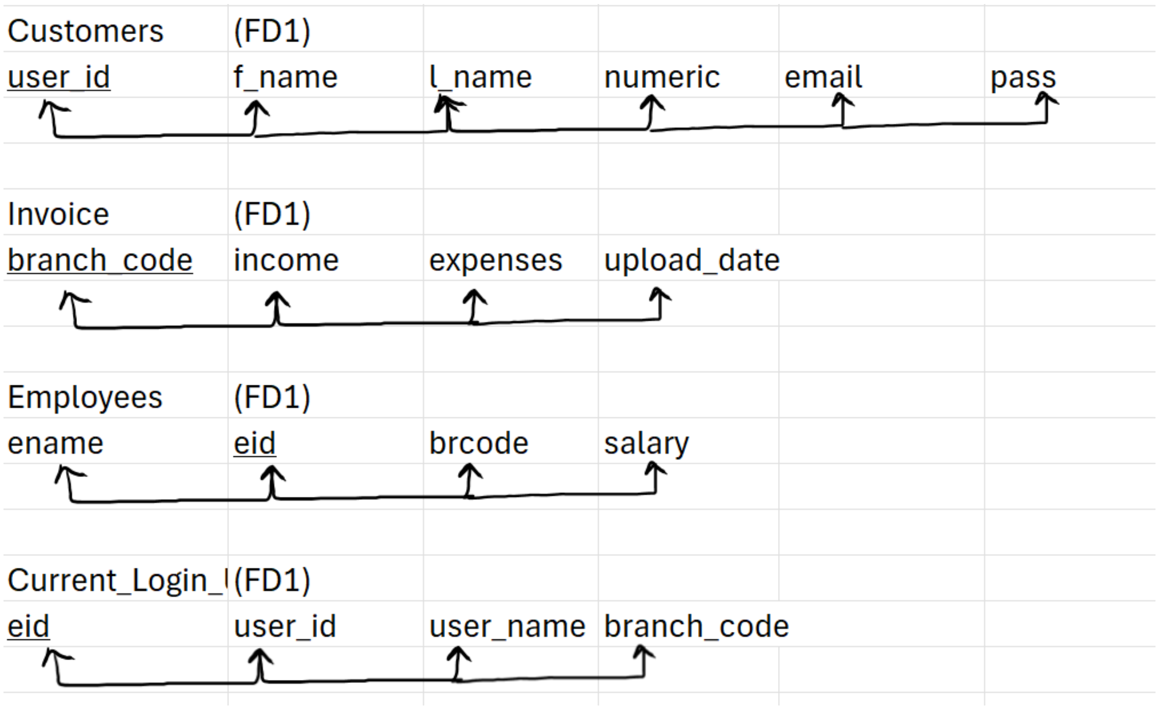
Customers	(FD1)				
user_id	f_name	l_name	numeric	email	pass
Invoice	(FD1)				
branch_code	income	expenses	upload_date		
Employees	(FD1)				
ename	eid	brcode	salary		
Current_Login_I	(FD1)				
eid	user_id	user_name	branch_code		

Boyle Codd Normal Form:

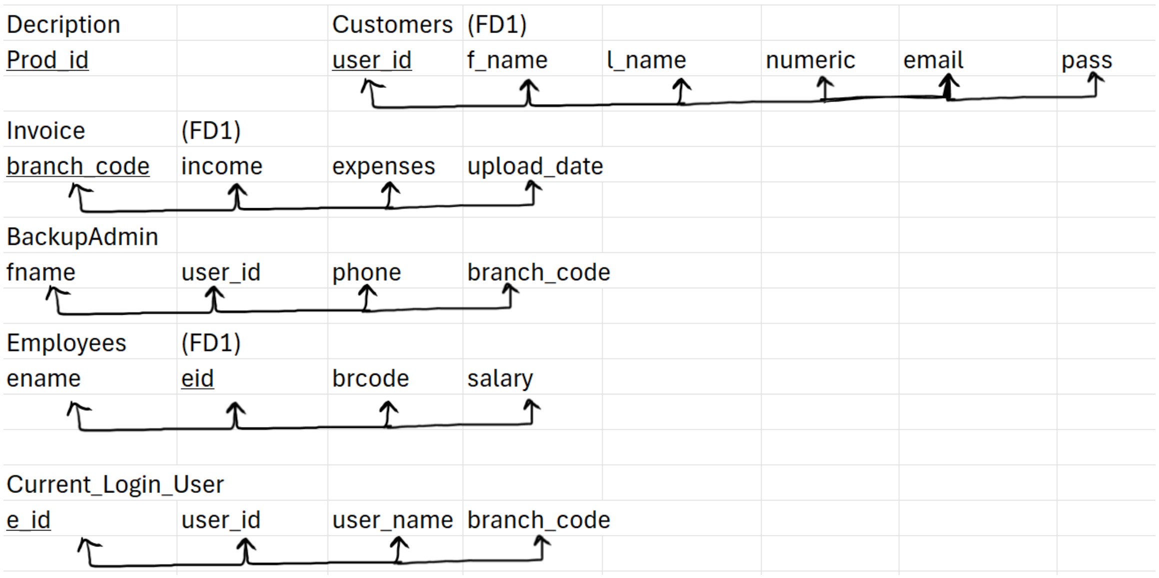
All the tables all the tables in given schema already satisfy BCNF because they do not have any nontrivial functional dependencies where the determinant is not a separate key.



AdminUsers	(FD1)						
f_name	l_name	user_id	user_pass	e_mail	phone#	branch_code	user_type
Branch	(FD1)						
branch_code	city						
Products	(FD1)						
Category	Prod_id	Price	Pro_type	Pro_location	Address	P_size	



Customers	(FD1)				
user_id	f_name	l_name	numeric	email	pass
Invoice	(FD1)				
branch_code	income	expenses	upload_date		
Employees	(FD1)				
ename	eid	brcode	salary		
Current_Login_l	(FD1)				
eid	user_id	user_name	branch_code		



Decription		Customers	(FD1)				
Prod_id		user_id	f_name	l_name	numeric	email	pass
Invoice	(FD1)						
branch_code	income	expenses	upload_date				
BackupAdmin							
fname	user_id	phone	branch_code				
Employees	(FD1)						
ename	eid	brcode	salary				
Current_Login_User							
e_id	user_id	user_name	branch_code				

CONCLUSION

The Wall Street Property Brokering provides a comprehensive solution for managing property listings and user inquiries in the real estate industry. With its user-friendly interface and robust functionality, it streamlines property management processes and enhances user engagement. Future enhancements may include additional features for analytics and reporting.