table of contents

Introduction

The Software Requirements Specification (SRS) document outlines the functional and non-functional requirements of Tulona, a web-based comparison platform designed to help users find the best financial and telecom deals across banks, telecom operators, and other service providers. The system provides a seamless browsing experience, intelligent comparison tools, AI-driven recommendations, structured data entry management, user authentication, and personalized suggestions.

Tulona enables users to compare products such as credit cards, loans, deposits, and telecom packages through side-by-side tables, visual graphs, and personalized insights based on user profiles. This SRS describes the platform's objectives, core functionalities, user interactions, and system behavior across all modules.

Purpose

The purpose of the Toluna system is to design and implement a centralized digital comparison platform that enables users to analyze, compare, and select financial and service-based products (such as banking and telecom services) in an informed and efficient manner. The system aims to bridge the information gap between service providers and consumers by presenting accurate, structured, and visually comparable data.

Additionally, Toluna supports data entry operators with full CRUD functionality to ensure data accuracy, consistency, and up-to-date product information. By integrating personalization, filtering, and AI-driven recommendations, the system enhances decision-making and improves user engagement.

Intended Audience

The intended audience for this SRS includes all stakeholders involved in the planning, development, and usage of the Tulona. This includes:

- **Development Team**: Software engineers, system designers, and testers responsible for building and maintaining the system.

● **General Users**: Users who want to see all banking, telecom, and financial products in a platform.

●**Registered Users**: Users who require comparing product, personalized recommendations, deal notifications, filters, and AI chatbot support.

● **Data Entry Operators**: Personnel responsible for managing and maintaining product data across institutions.

● **Financial Institutions & Service Provider**: Banks, telecom companies, and service providers showcasing their products.

Conclusion

In conclusion, the SRS document for the Tulona platform provides a clear foundation for system development by defining its purpose, features, and stakeholders. The system is designed to improve transparency and decision-making through accurate comparisons, visual analytics, and personalized recommendations. With secure authentication and AI-based assistance, TULONA ensures a user-friendly and efficient experience. Overall, the platform delivers a robust solution that supports the digital transformation of service comparison systems.

Inception

Identifying the Stakeholders

Direct Stakeholders

Direct stakeholders are individuals or groups who directly interact with the Tulona system and are actively involved in its operation, management, and decision-making processes.

- **End Users (Registered Users)** – Users who sign up to receive personalized recommendations, apply filters, compare products, and access AI-based suggestions.

- **Guest Users (Unregistered Users)** – Users who browse products without authentication.

- **Data Entry Operators** – Responsible for inserting, updating, and maintaining product data across banks and service providers.

Indirect Stakeholders

Indirect stakeholders do not interact with the system directly but are affected by its performance, accuracy, and outcomes.

- **Banks and Financial Institutions** – Their products are compared, promoted, and applied for through the platform.

- **Telecom Service Providers** – Internet, call, and bundle services are listed and compared.

- **Regulatory Authorities** – Monitor compliance with financial and data protection regulations.

- **Advertisers & Partners** – Benefit from increased visibility and targeted recommendations.

- **Technology Providers** – Maintain hosting infrastructure, notification services, and AI modules.

- **General Consumers** – Benefit indirectly from increased market transparency and fair comparison.

- **Data Analytics & Research Teams** – Use anonymized insights for market trend analysis.

- **Customer Support Teams** – Handle queries, complaints, and service-related issues.

Elicitation of

The elicitation phase focuses on understanding user expectations, resolving domain complexity, and translating business needs into system requirements. A collaborative approach was adopted involving all key stakeholders to ensure clarity and completeness.The following activities were carried out during requirement elicitation:

- **Collaborative Requirements Gathering**
- **Quality Function Deployment (QFD)**
- **Usage Scenarios**

Collaborative Requirements Gathering

During the inception phase, we met with stakeholders such as users, data entry operators and financial domain experts. These initial meetings provided a general understanding but were insufficient to finalize the requirements. To clarify and elicit comprehensive requirements, multiple follow-up sessions were conducted with all stakeholders actively involved in the project lifecycle.

Quality Function Deployment (QFD)

Quality Function Deployment is a technique that translates stakeholders' needs into technical specifications for the system. It aims to convert subjective expectations into measurable requirements that can be designed, implemented, and tested. By using

QFD, we identified functional and non-functional requirements for the Tulona. Normal requirements are the basic objectives necessary to satisfy stakeholders:

- Users can browse products across multiple categories (Bank, Telecom, etc.).
- Registered users can sign up and verify accounts using OTP.
- Secure login using email/phone and password.
- Password recovery using OTP verification.
- Users can select up to four companies or products for side-by-side comparison.
- Dynamic comparison tables with key attributes (rates, fees, benefits).
- Graph generation for visual comparison and analysis.
- Data entry operators can perform full CRUD operations on product data.
- Real-time updates of product information.
- Users can submit ratings and feedback for products.

Expected Requirements

Expected requirements are implicit but critical for user satisfaction:

- User-friendly and responsive interface.
- Secure data handling.
- Cross-platform compatibility (desktop, tablet, mobile).
- Advanced filtering options across all categories.
- Seamless integration between comparison , graphs.
- Email and in-app notifications for deals and updates.

Exciting Requirements

Exciting requirements enhance user experience and provide competitive advantage:

- AI-powered chatbot supporting natural language queries.

User Story- TULONA

The data entry operator is responsible for maintaining the product database across multiple financial institutions and service providers. Upon accessing the data management interface, the operator receives a structured form containing predefined fields specific to each product category. The operator fills these fields with institution-specific information (e.g., bank name, product specifications, rates, terms, and conditions). The system provides full CRUD (Create, Read, Update, Delete) functionality, enabling the operator to save entries for immediate publishing or store them as drafts for later revision. All data modifications are logged and may be edited at any time to ensure accuracy and relevance.

Upon accessing the Tulona web application, users are presented with a clean and intuitive interface displaying the primary service categories (e.g., Banks, Telecom Companies, etc.). At the upper corner of the website, a profile icon is displayed. When a new or unregistered user clicks on this icon, a prompt appears offering the options to Sign Up or Login.

Upon selecting the Sign Up option, the user is guided through a clean and structured registration flow. First, the system prompts the user to enter their name and email address. After providing this information, the system opens an OTP verification page, and the user chooses where they would like to receive their OTP (One-Time Password)—either through email or via SMS to their phone number. Once the preferred option is selected, the system automatically sends a 6-digit OTP to the chosen destination.The user enters the received OTP, After the user enters the received OTP, the system validates it by checking whether the OTP is still within its valid time window and ensuring that the entered code matches the one generated by the system. After successful verification, the system requests additional personal details, such as the user's profession and monthly income, to complete the profile setup. The user then proceeds to create their account by setting a password and confirming it.When all required information is submitted, the registration process is finalized, and the system displays a message confirming that the user has been successfully registered.

Once registration is complete, users can log in using any of the supported authentication methods: Direct Google authentication using their Gmail account, or Email or phone number with the password they created during signup. For users who choose the Email/Phone login method, an additional Forgot Password option is available under the login form. If a user clicks on Forgot Password, the system prompts them to enter their registered email or phone number. After submission, the system sends a 6-digit OTP to the provided destination. The user enters the OTP, and once it is successfully verified, the system allows them to set a new password and confirm it. After the password is reset, the user is redirected back to the login page, where they can now sign in using their newly created password.

After the user completes the signup or login process, the system reloads the interface and redirects them to the home page. The profile icon in the menu bar is then updated to reflect the user's authenticated state. Clicking on the profile now displays additional options such as: My Profile (where the user can update their profession, monthly income, and other personal details), Change Password, Settings, Logout . This dynamic behavior ensures a personalized and seamless navigation experience for every registered user.

If a user chooses not to sign up or log in, they can still browse the website; however, they will not receive the full benefits. They will only have access to product comparison features. Advanced features such as personalized suggestions, bank references, deal notifications, and AI-based recommendations will not be available to them.

After logging in, users are redirected to the homepage. From there, they can select any category (bank or telecom).

On the homepage, all users can view multiple categories (bank, telecom). Each category contains several product groups, and each product group can have subcategories. For example, within the Bank category, product groups include Cards, Loans, and Investments. Under Cards, the subcategories are Credit Cards and Debit Cards. Under Credit Cards, users will find multiple banks offering various options. Thus, the navigation flow follows: Bank → Cards → Credit Card → Multiple Banks.

Flows vary across categories. For instance, under Telecom Operators, users may choose Internet, Call Rates, Minutes, Bundles, etc. Selecting Internet will display a list of companies providing internet services, allowing users to compare them. Here, the flow is: Telecom → Internet → Multiple Companies. In summary: Main Category → Product Category → Subcategory → Providing Companies.

When all users select a category and navigate to a product category,if the user is registered or logged in, they will see a list of companies offering that product or service. Registered users may select up to four companies for comparison. As they begin selecting companies, a “Compare” button appears at the bottom. Once they finalize their choices, they can click the button to initiate the comparison. The comparison data is sourced either through data entry operators or via web crawling. The system then generates side-by-side comparisons and relevant graphs.

On the Credit Card page, all users will find multiple banks along with their available credit card options. Registered users can select multiple cards for side-by-side comparison. The comparison table displays annual fees, interest rates/APR, rewards, cashback, special offers, credit limits, and eligibility criteria. Below the table, one or more multi-color graphs visualize comparisons—for example, BB ratings, financial growth, or stock prices of the banks. Each card also includes an “Apply” button.

In the Deposit section, all users can view a list of banks and their offered interest rates. Each bank is displayed in a block containing key details and an “Apply” button. Registered users can also select multiple banks using the Select button at the top-right of each block. When two or more banks are selected, a “Compare” button appears at the bottom. Clicking it shows a comparative table with interest rates, tenures, and minimum deposit requirements. Below the table, one or more graphs help users visualize differences quickly, displaying metrics such as financial growth or stock price. Each comparison includes an “Apply Now” button.

The Loan section functions similarly to the Deposit section. Users can view banks offering loan products along with their respective interest rates. Each loan product has an “Apply” button. Registered users can also select multiple banks for comparison. Clicking “Compare” displays a detailed comparison table and multi-color graphs, helping the user choose the best loan based on rate, return policies, etc.

When navigating from product category → company section, Registered users can view detailed product information under each company. For example, selecting Bank → Credit Card → Bank Name will show the full details of that bank’s credit card offerings.

Based on the answers submitted earlier, Registered users receive personalized offers and deal notifications in the notification section, represented by an icon in the upper right corner of the webpage.

At the final stage of navigation—Main Category → Product Category → Subcategory → Companies—Registered users can apply filters to refine results based on their criteria. For example, someone searching for a loan may filter by loan duration or loan amount. This narrows down the list (e.g., from 20 banks to 8 to 10 banks), making comparison easier. Filters are available across all categories.

When a registered user logs in, they will see a chat icon at the bottom left of the screen. Clicking it opens an AI chatbot window. By asking questions related to their interests, users receive personalized deal recommendations. The chatbot may suggest one or multiple companies.

Each company’s product page includes a feedback section where registered users can rate products on a scale of 5 and share their experiences. This helps future users make informed decisions not only through comparison data and graphs but also by considering previous user experiences.

Requirements Modeling

Scenario Based Modeling

Use Case Diagram

Level 0: Tulona - comparison for finding the best deal

Primary actor : user
Secondary actor : Data entry operator
External system : SMS service system, Email service system, google OAuth service system

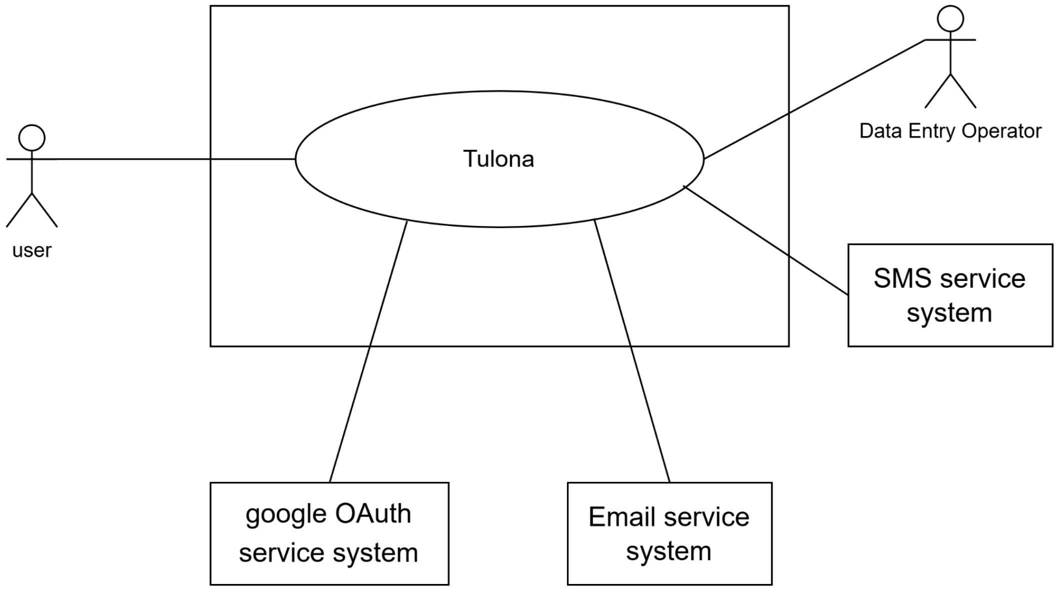


Figure 1: Use Case Diagram of Tulona

Level 1: Tulona - comparison for finding the best deal

Primary actor : user

Secondary actor : Data entry operator

External system : SMS service system, Email service system, google OAuth service system

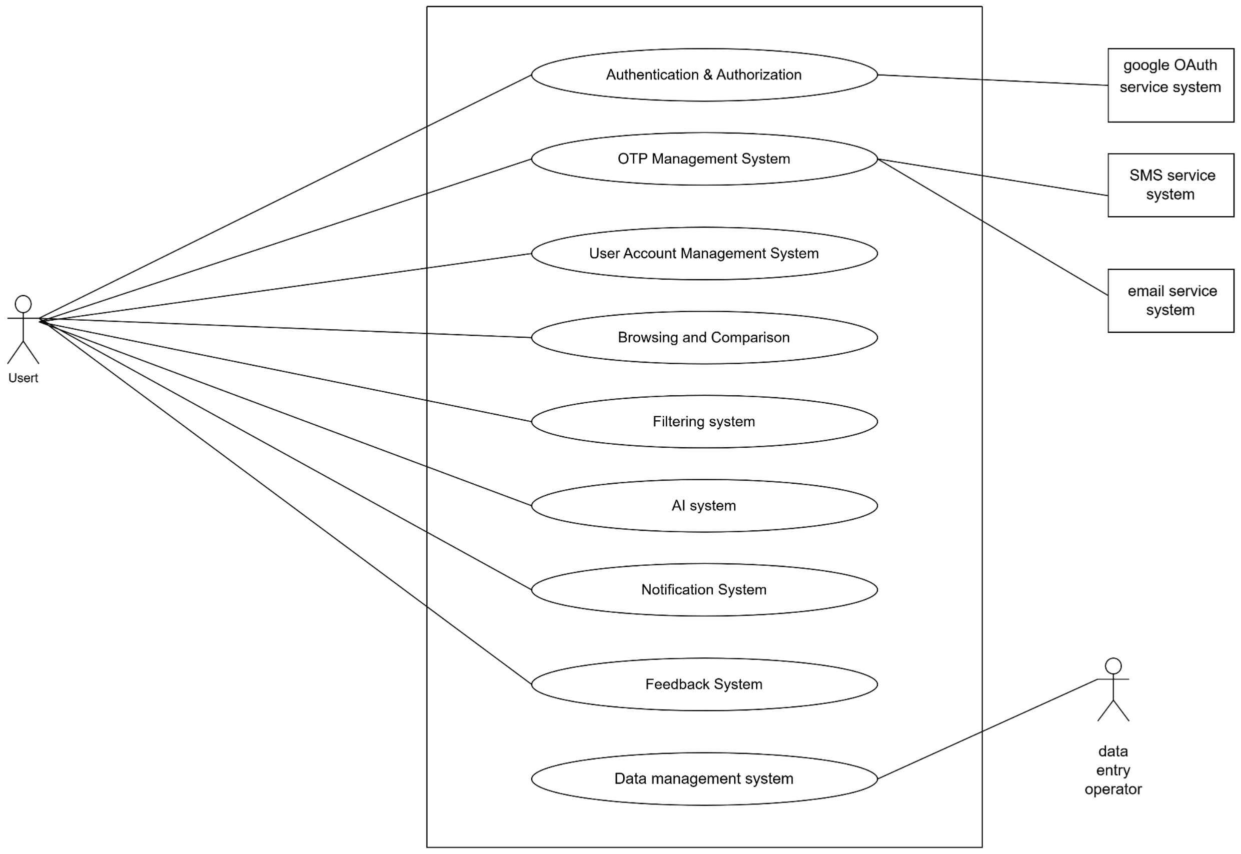


Figure 2 : Use Case Diagram of Tulona

Level 1.1: Authentication & Authorization

Primary actor : user
Secondary actor : -
External system : google OAuth service system

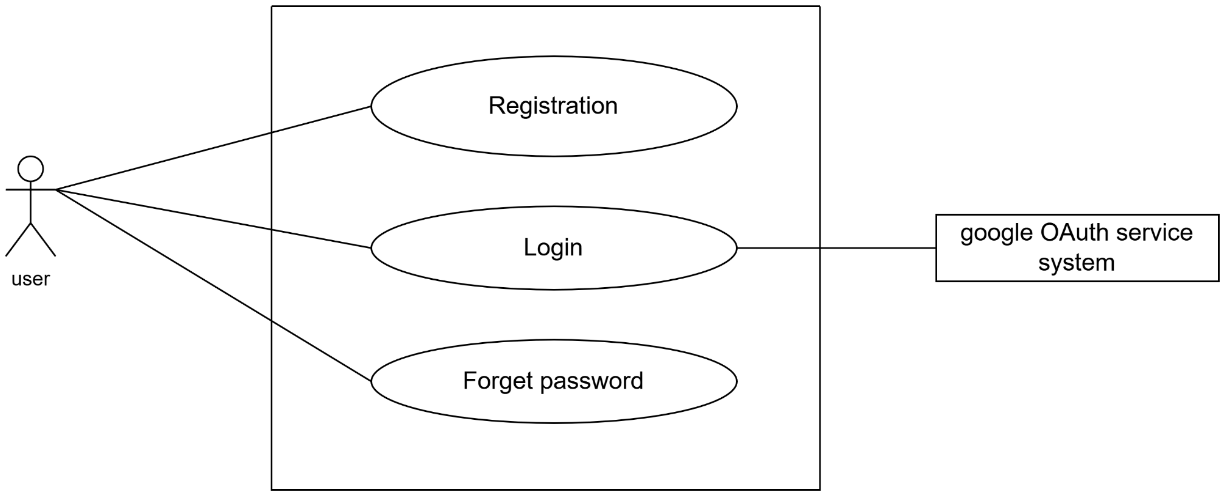


Figure 3 : Use Case Diagram of Authentication & Authorization

Action 1: User clicks Sign Up and submits name + email/phone.
Reply 1: OTP Management System sends OTP delivery options (SMS/Email).

Action 2: User submits profession, monthly income, and creates password.
Reply 2: System registers user and shows "Registration Successful" message.

Action 3: User selects Login and enters email/phone + password OR Google login.
Reply 3: System validates credentials or contacts Google OAuth service.

Action 4: User submits authentication request.
Reply 4: System creates session and redirects to homepage with updated profile icon.

Action 5: User clicks "Forgot Password" and enters registered email/phone.
Reply 5: System sends OTP using SMS/Email subsystem.

Action 6: User sets new password and submits.
Reply 6: System resets password and redirects user to login page.

Level 1.2: OTP Management system

Primary actor : user
Secondary actor : -
External system : SMS service system , Email service system

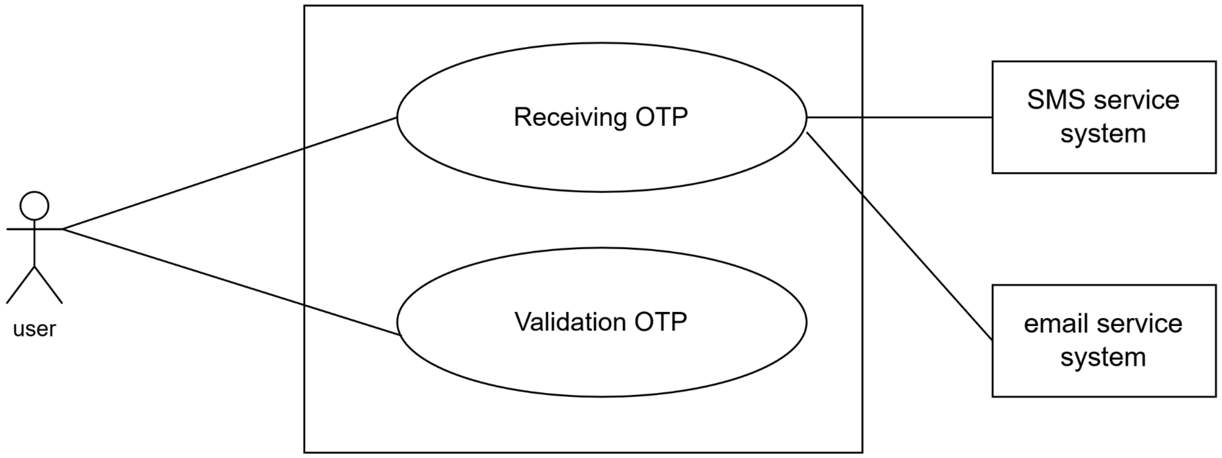


Figure 4 : Use Case Diagram of OTP Management system

Action 1: Registration/Login/Forgot Password triggers OTP request.
Reply 1: System generates OTP and sends via SMS Service system or email.

Action 2: User enters OTP into verification box.
Reply 2: System checks OTP validity, expiration, and match.

Level 1.3: User Account Management system

Primary actor : user
Secondary actor : -
External system : -

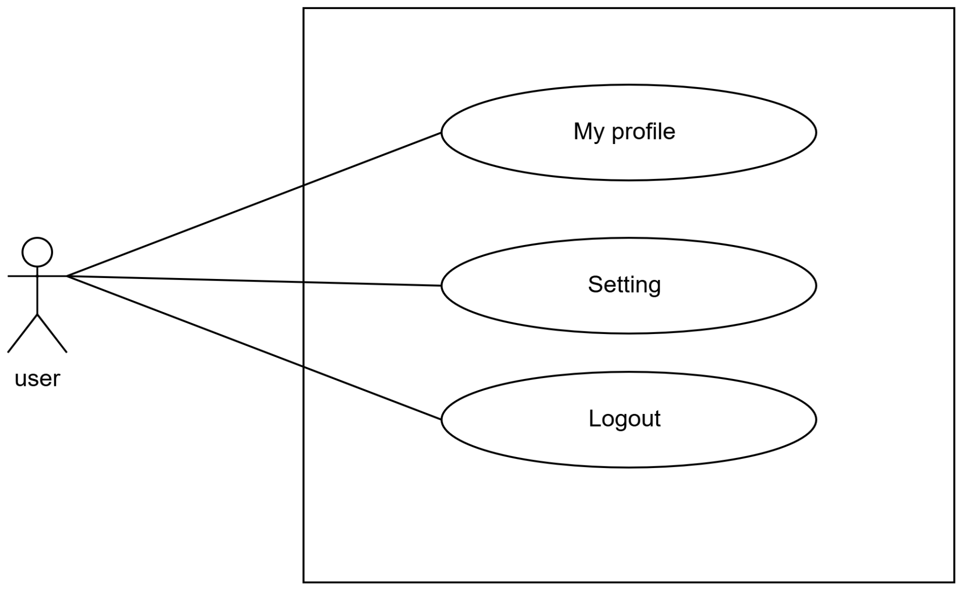


Figure 5 : Use Case Diagram of User Account Management system

Action 1: User opens “My Profile”.
Reply 1: System loads stored profile data.

Action 2: User edits profession, monthly income, or personal info.
Reply 2: System saves changes and confirms update.

Action 3: User opens “Settings”.
Reply 3: System loads all configurable preferences.

Action 4: User updates preferences.
Reply 4: System saves updated settings.

Action 5: User clicks Logout button.
Reply 5: System clears session and returns user to homepage as guest.

Level 1.4: Browsing & Comparison System

Primary actor : user
Secondary actor : -
External system : -

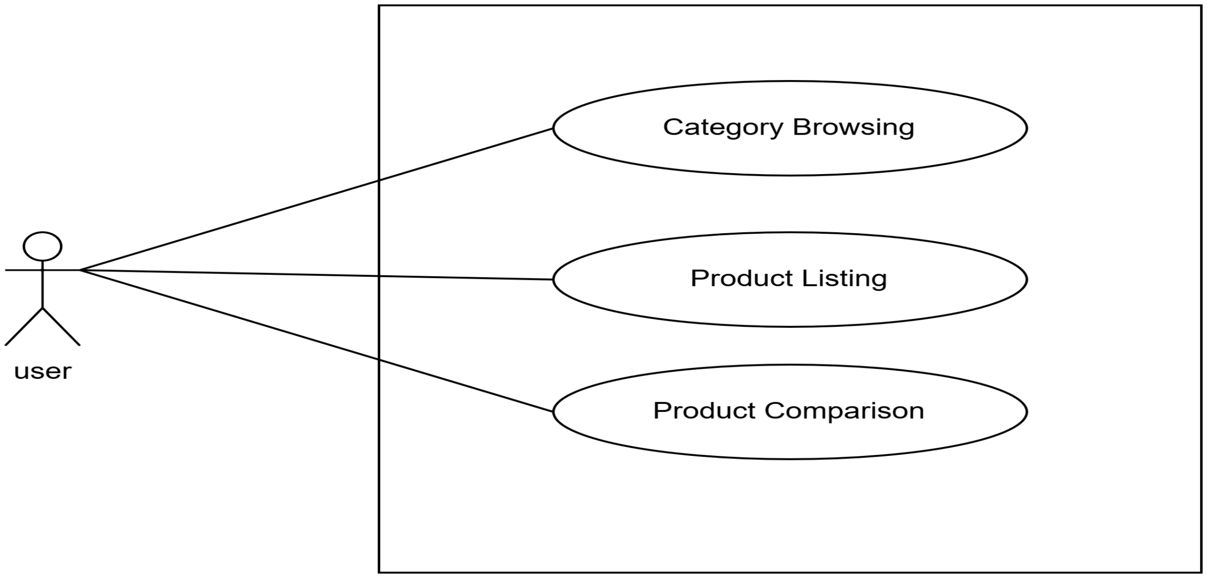


Figure 6 : Use Case Diagram of Browsing & Comparison system

Action 1: User clicks on a service category (Bank / Telecom).
Reply 1: System loads all product groups under that category.

Action 2: User selects a product group (Cards, Loans, Internet, etc.).
Reply 2: User selects a product group (Cards, Loans, Internet, etc.).

Action 3: User selects up to four companies or products.
Reply 3: System displays “Compare” button.

Action 4: User clicks “Compare”.
Reply 4: System generates comparison tables and graphs with interest rates, features, APR, ratings, etc.

Level 1.5 : Filtering system

Primary actor : user
Secondary actor : -
External system : -

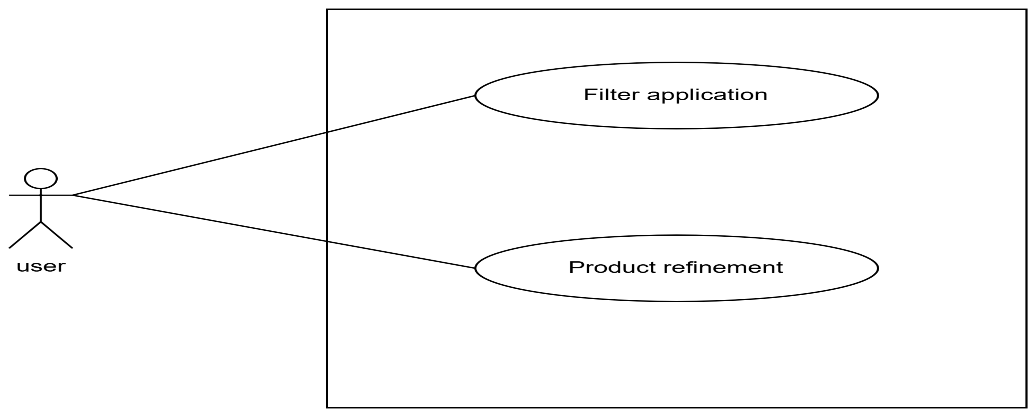


Figure 7 : Use Case Diagram of Filtering system

Action 1: User applies filters .
Reply 1: System filters products and updates the listing.

Action 2: User refines or removes filters.
Reply 2: System refreshes results, reducing results.

Level 1.6 : AI system

Primary actor : user
Secondary actor : -
External system : -

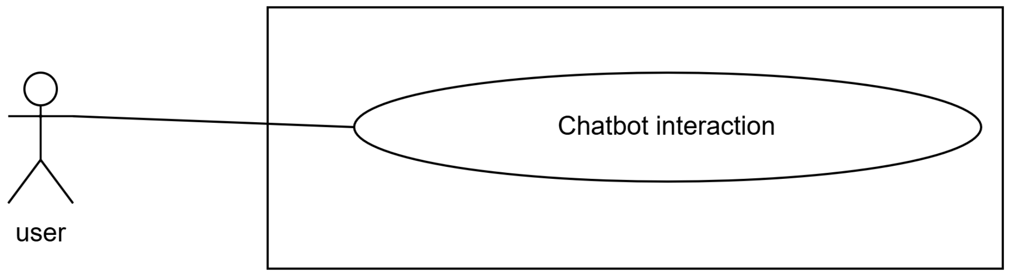


Figure 8 : Use Case Diagram of AI system

Action 1: User clicks chatbot icon and asks for product suggestions.
Reply 1: AI system analyzes profile + query and shows recommendations from multiple companies.

Level 1.7 : Notification System

Primary actor : user

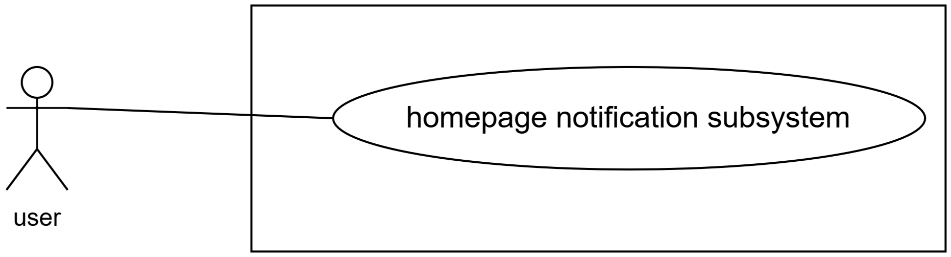


Figure 9 : Use Case Diagram of Notification system

Action 1: User checks notification icon.
Reply 1: System shows personalized deal notifications, updates, alerts.

Level 1.8 : Feedback System

Primary actor : user

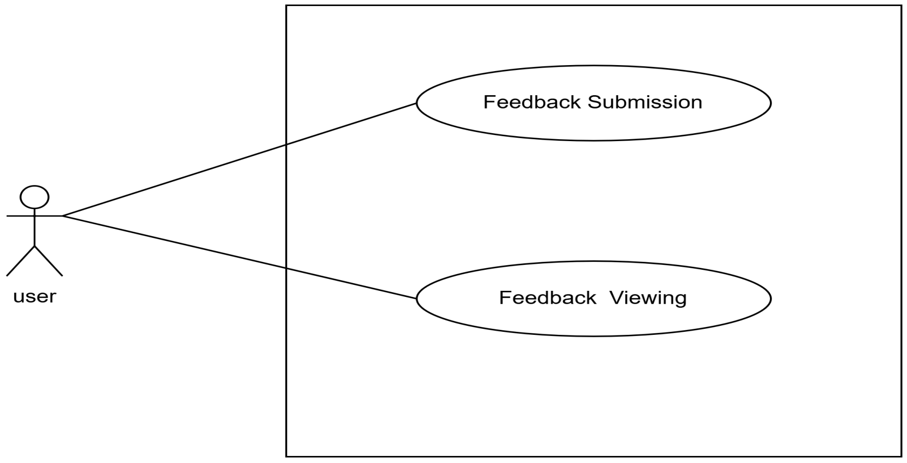


Figure 10 : Use Case Diagram of Feedback system

Action 1: User rates a product and writes a review.

Reply 1: System stores feedback and confirms successful submission.

Action 2: User opens product page and scrolls to feedback section.
Reply 2: System displays average rating and list of user reviews.

Level 1.9 : Data management system

Primary actor : -
Secondary actor : data entry operator
External system : -

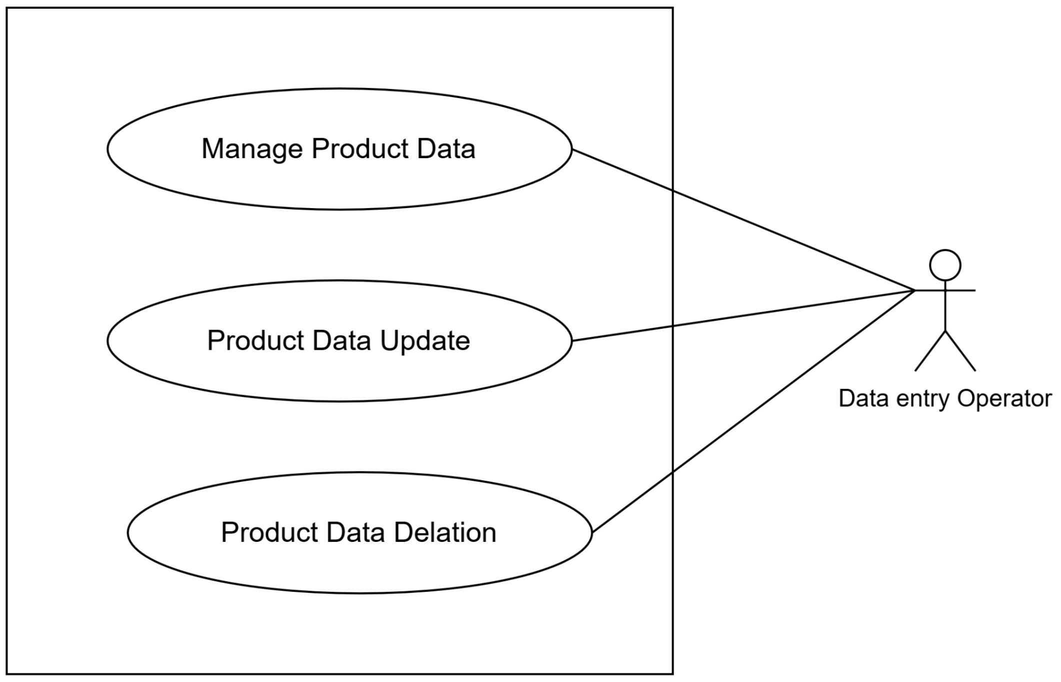


Figure 11 : Use Case Diagram of Data management system

Action 1: Data Entry Operator opens data form and enters product/company details.
Reply 1: System saves data and logs changes.

Action 2: Operator edits existing entry.
Reply 2: System updates entry and stores modification log.

Action 3: Operator deletes outdated product data.
Reply 3: System removes data and updates lists for all users.

Action 4: Operator saves as draft.
Reply 4: System stores draft for later editing.

Activity Diagram

Level 1 : Tulona -comparison for finding the best deal

Ref : Use Case Diagram Level 1

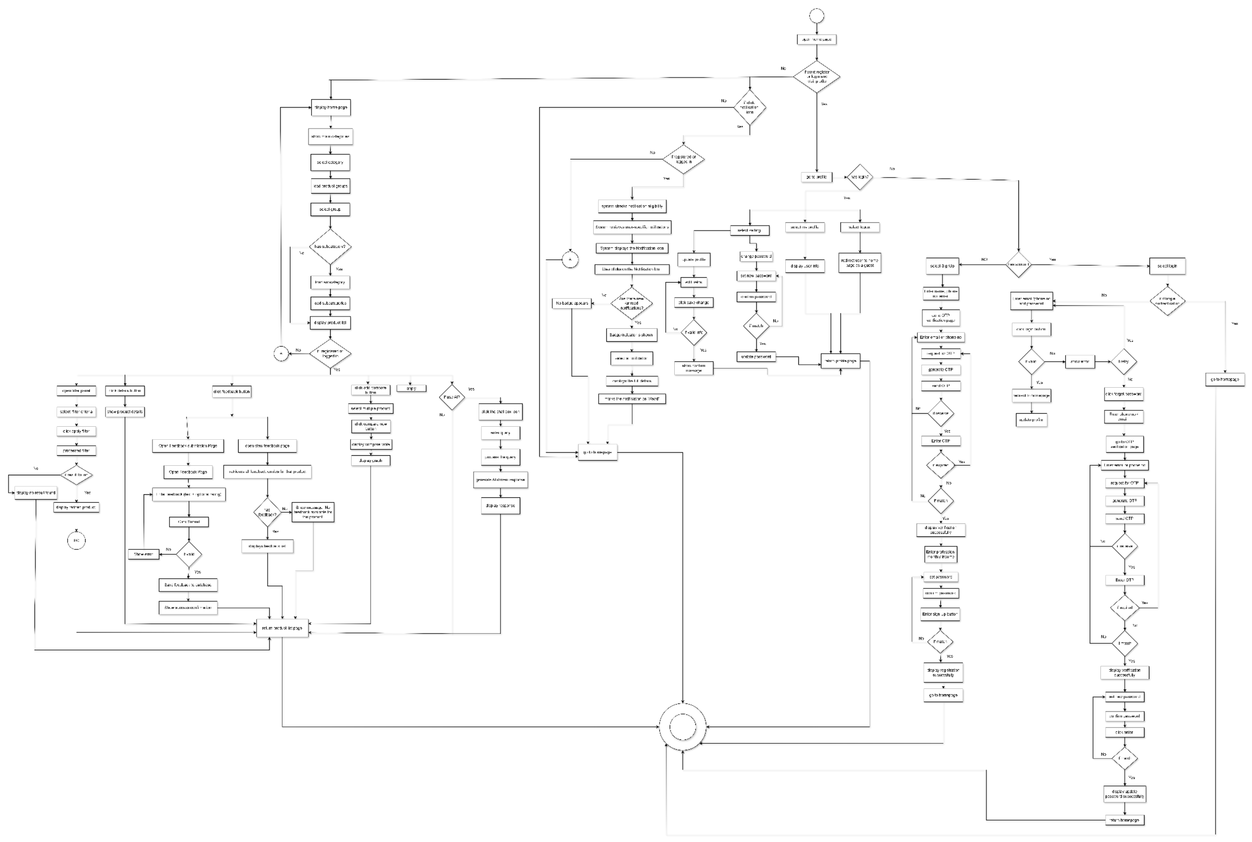


Figure 12 : Activity Diagram of Tulona

Level 1.1: Authentication & authorization

Ref : Use Case Diagram Level 1.1

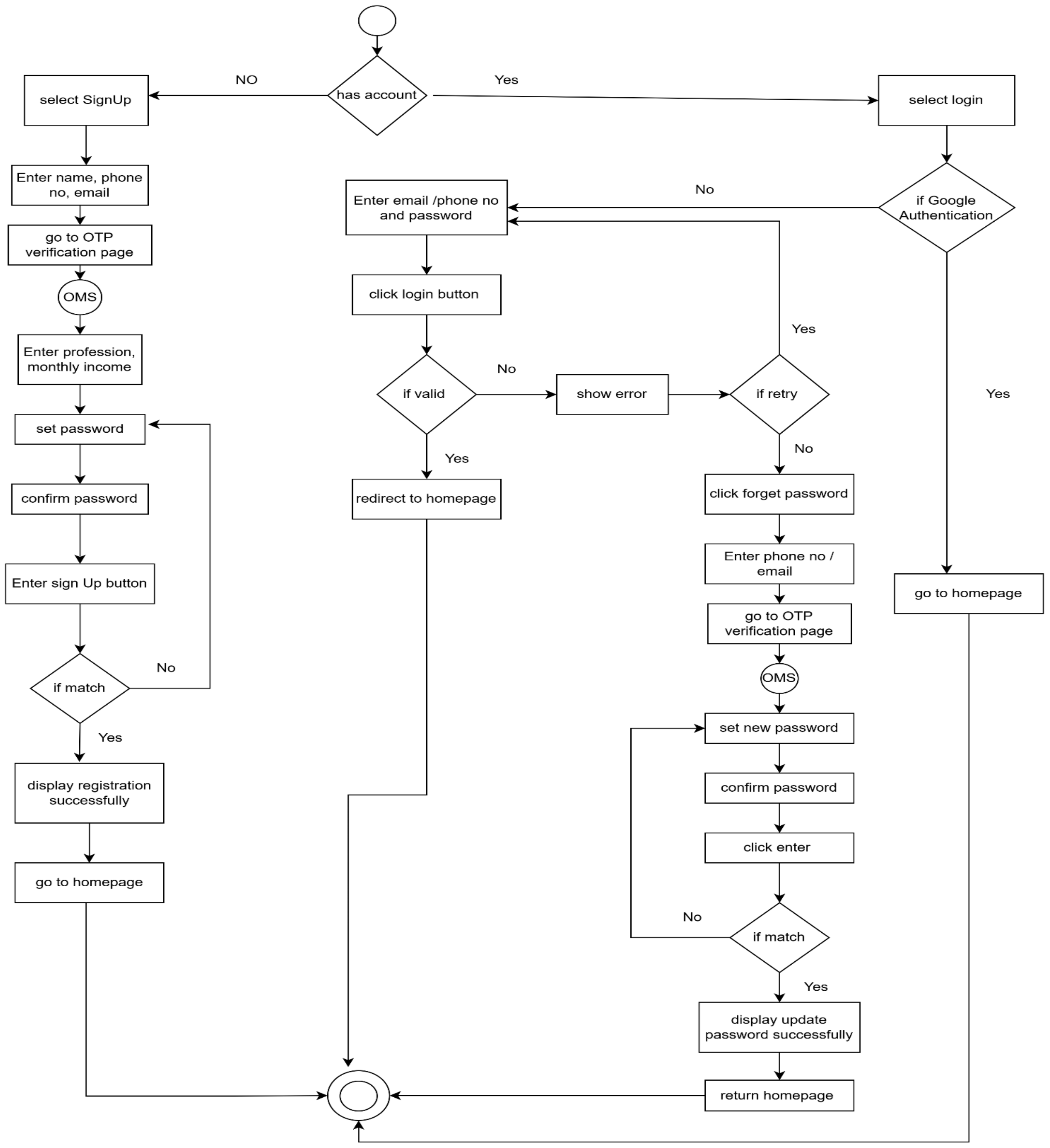


Figure 13 : Activity Diagram of Authentication & authorization

Level 1.2 : OTP Management system

Ref : Use Case Diagram Level 1.2

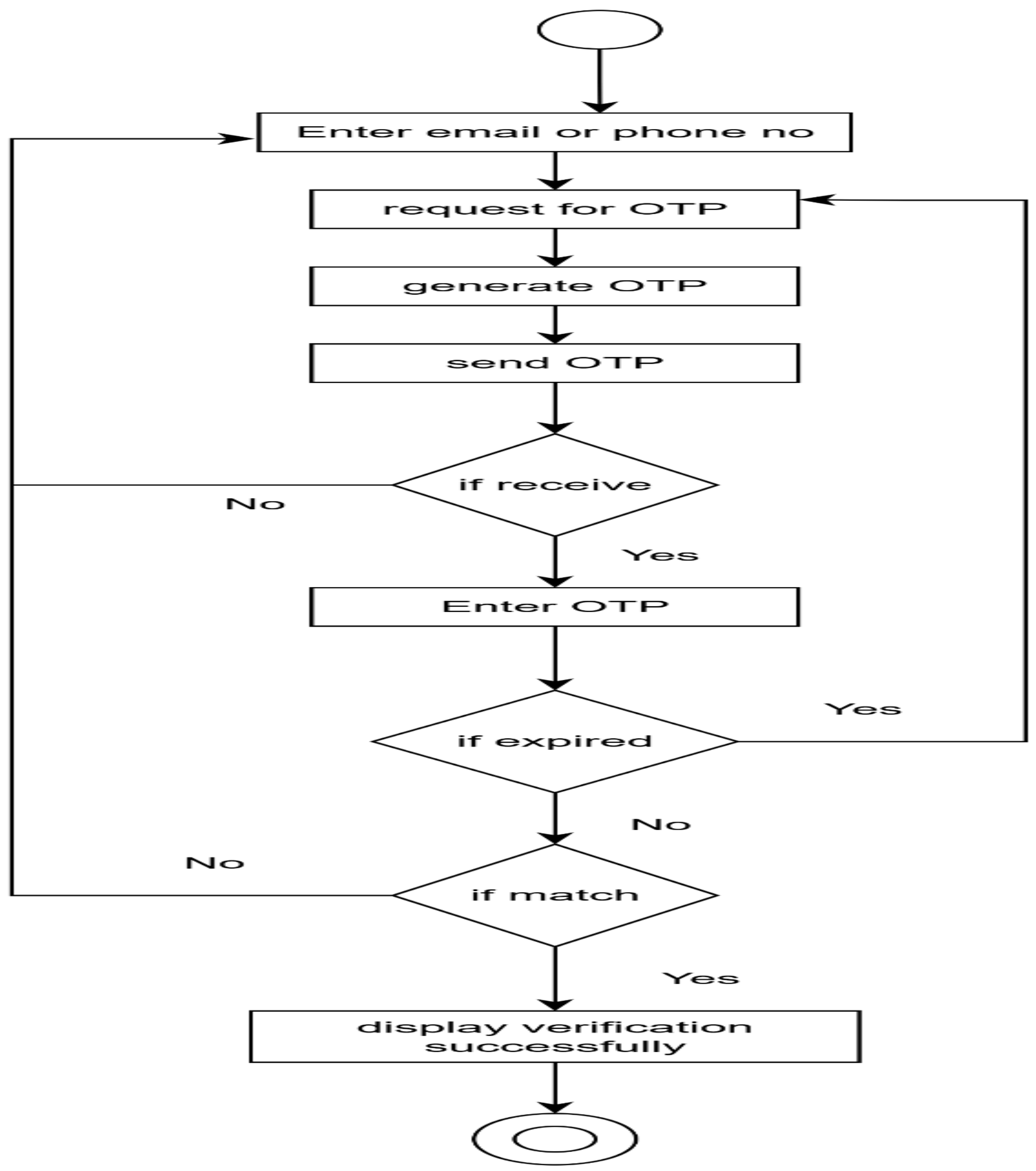


Figure 14 : Activity Diagram of OTP Management system

Level 1.3 : User Account Management system

Ref : Use Case Diagram Level 1.3

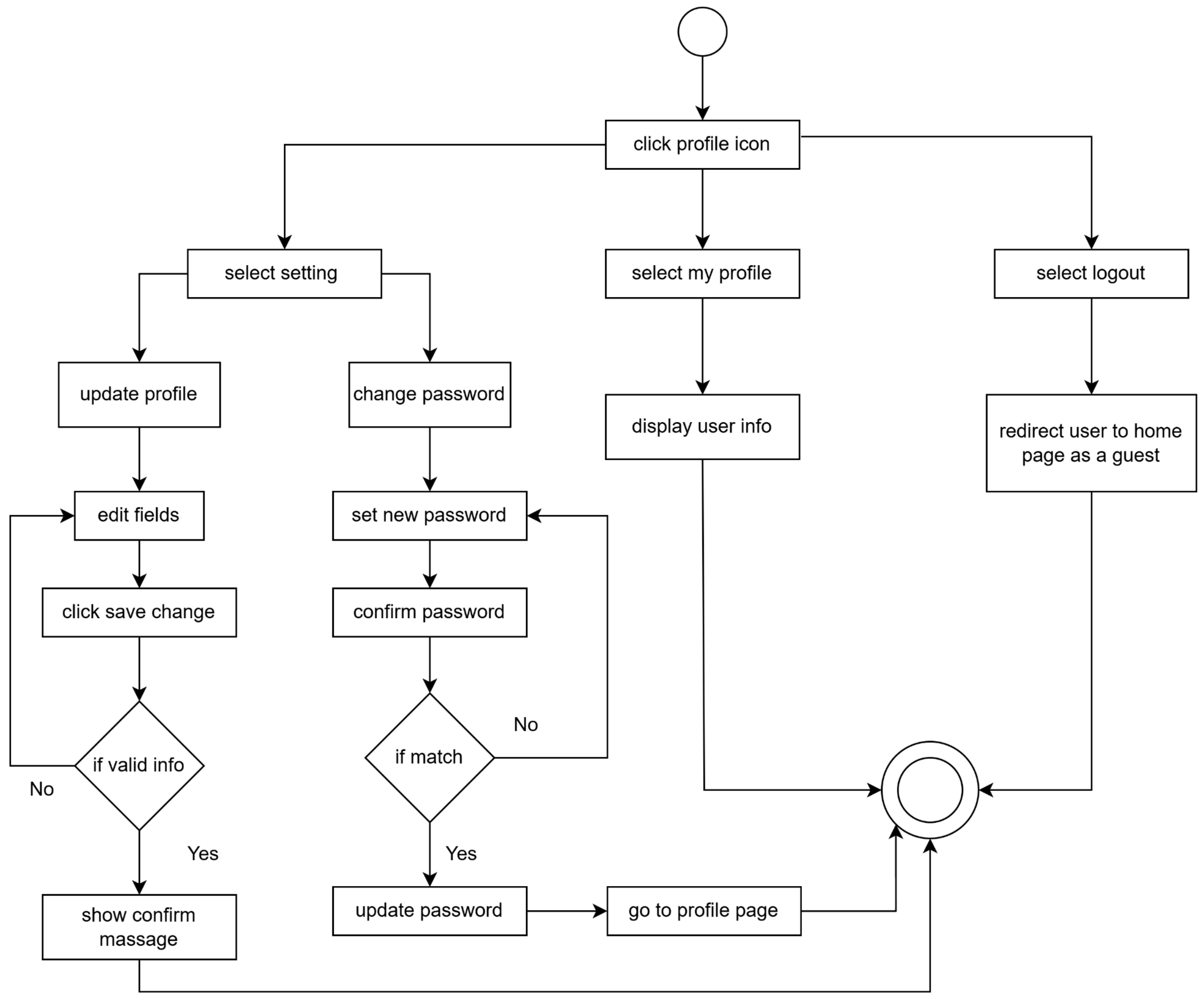


Figure 15 : Activity Diagram of User Account Management system

Level 1.4 : Browsing & Comparison

Ref : Use Case Diagram Level 1.4

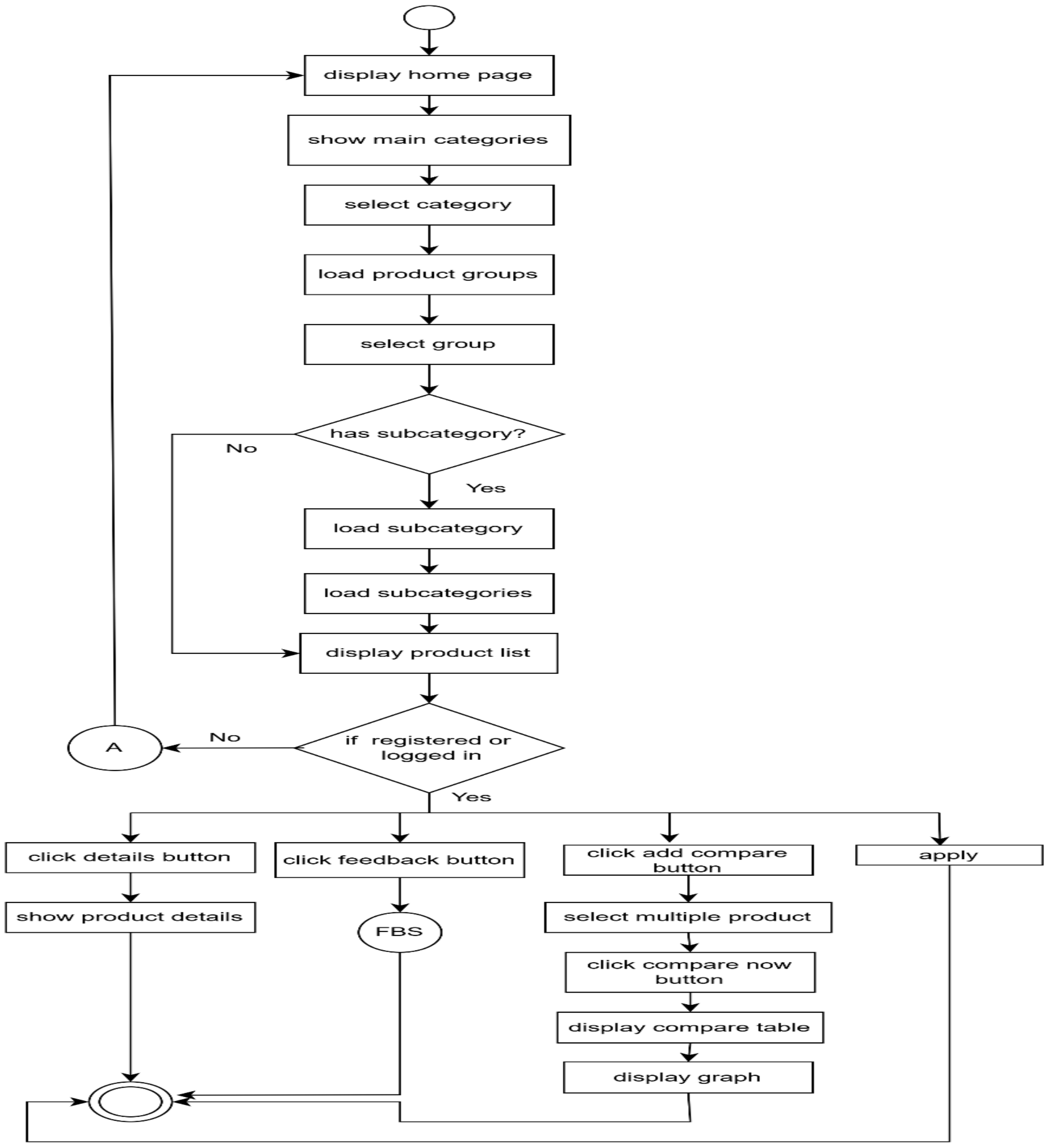


Figure 16 : Activity Diagram of Browsing & Comparison

Level 1.5 : Filtering system

Ref : Use Case Diagram Level 1.5

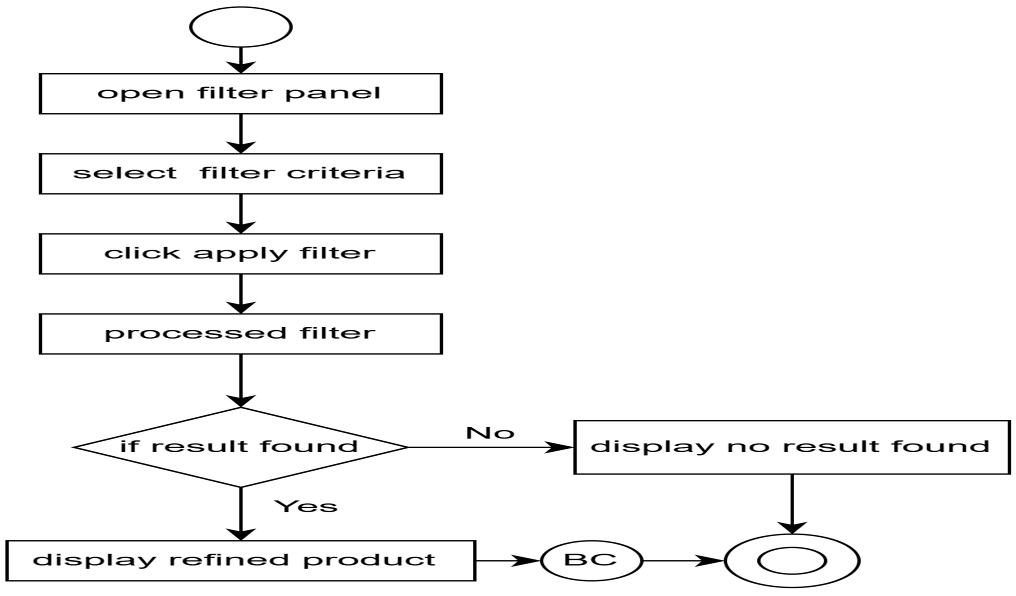


Figure 17 : Activity Diagram of Filtering system

Level 1.6 : AI system

Ref : Use Case Diagram Level 1.6

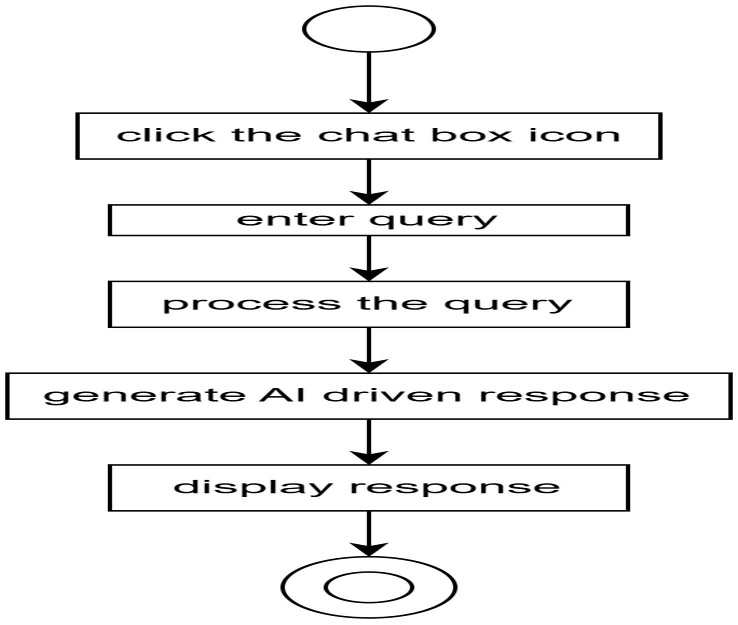


Figure 18 : Activity Diagram of AI system

Level 1.7 : Notification System

Ref : Use Case Diagram Level 1.7

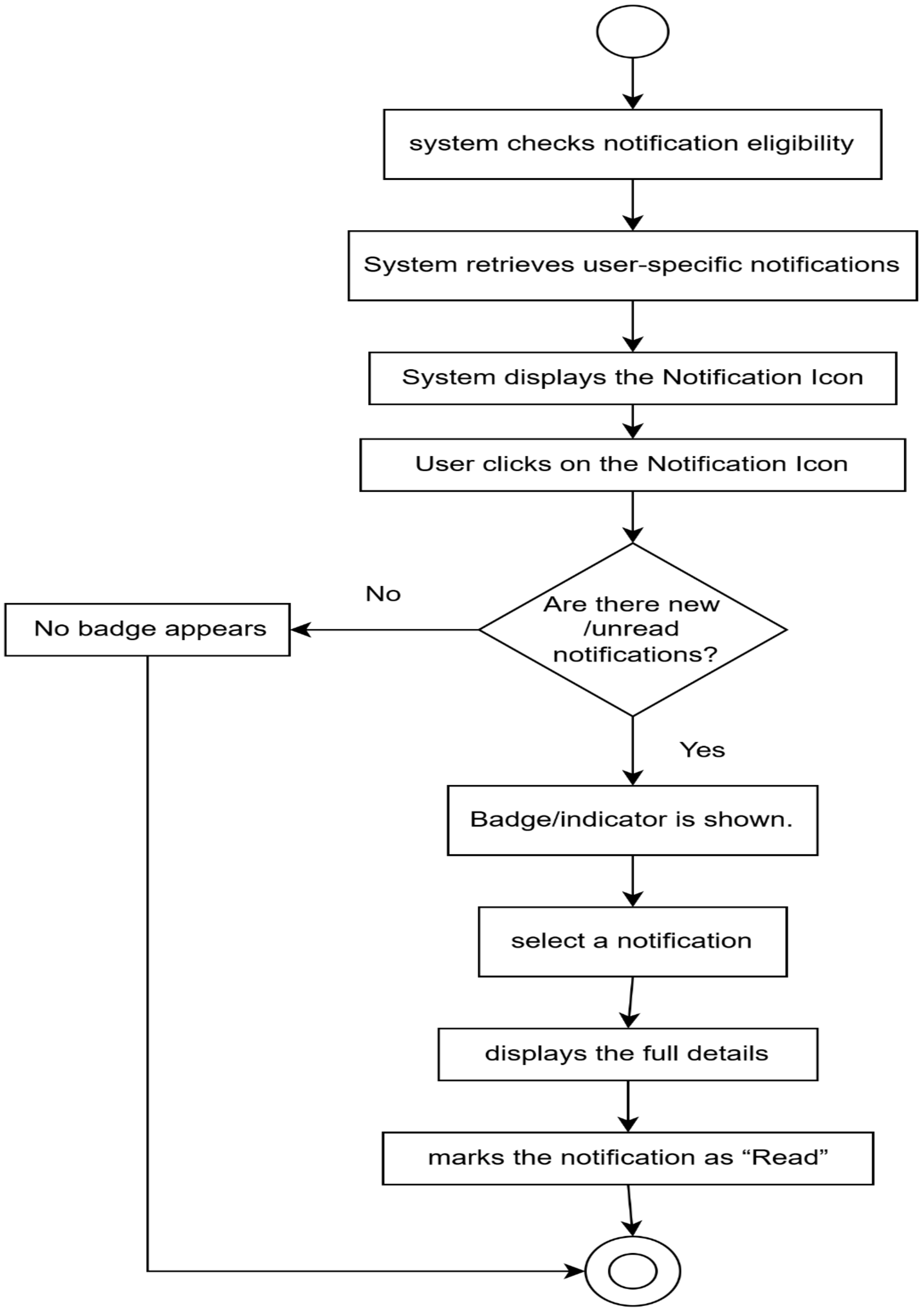

system checks notification eligibility
System retrieves user-specific notifications
System displays the Notification Icon
User clicks on the Notification Icon
Are there new /unread notifications?
No
No badge appears
Yes
Badge/indicator is shown.
select a notification
displays the full details
marks the notification as “Read”

Figure 19 : Activity Diagram of Notification System

Level 1.8 : Feedback System

Ref : Use Case Diagram Level 1.8

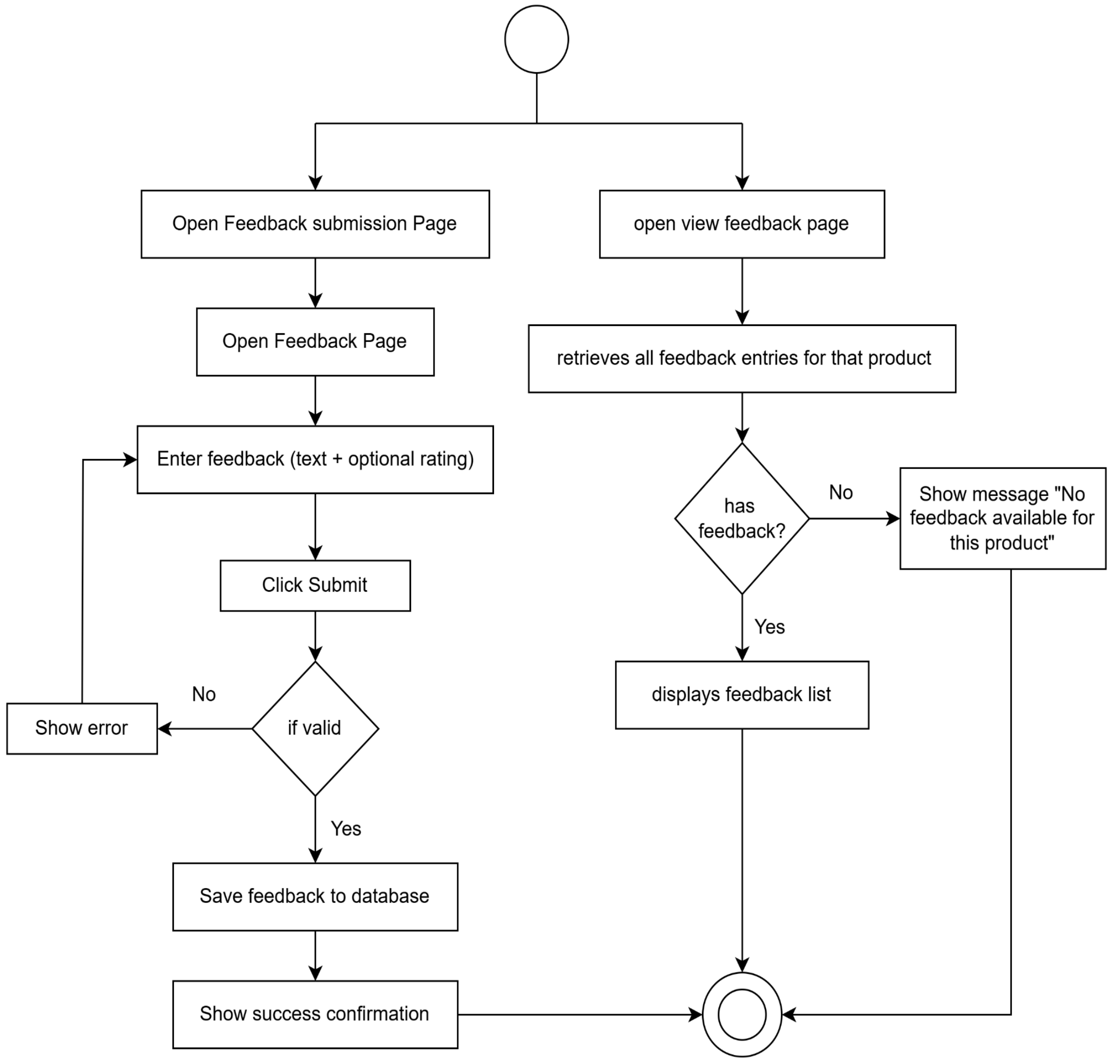

Open Feedback submission Page
open view feedback page
Open Feedback Page
retrieves all feedback entries for that product
Enter feedback (text + optional rating)
has feedback?
No
Show message "No feedback available for this product"
Click Submit
Yes
No
displays feedback list
Show error
if valid
Yes
Save feedback to database
Show success confirmation

Figure 20 : Activity Diagram of Feedback System

Level 1.9 : Data management system

Ref : Use Case Diagram Level 1.9

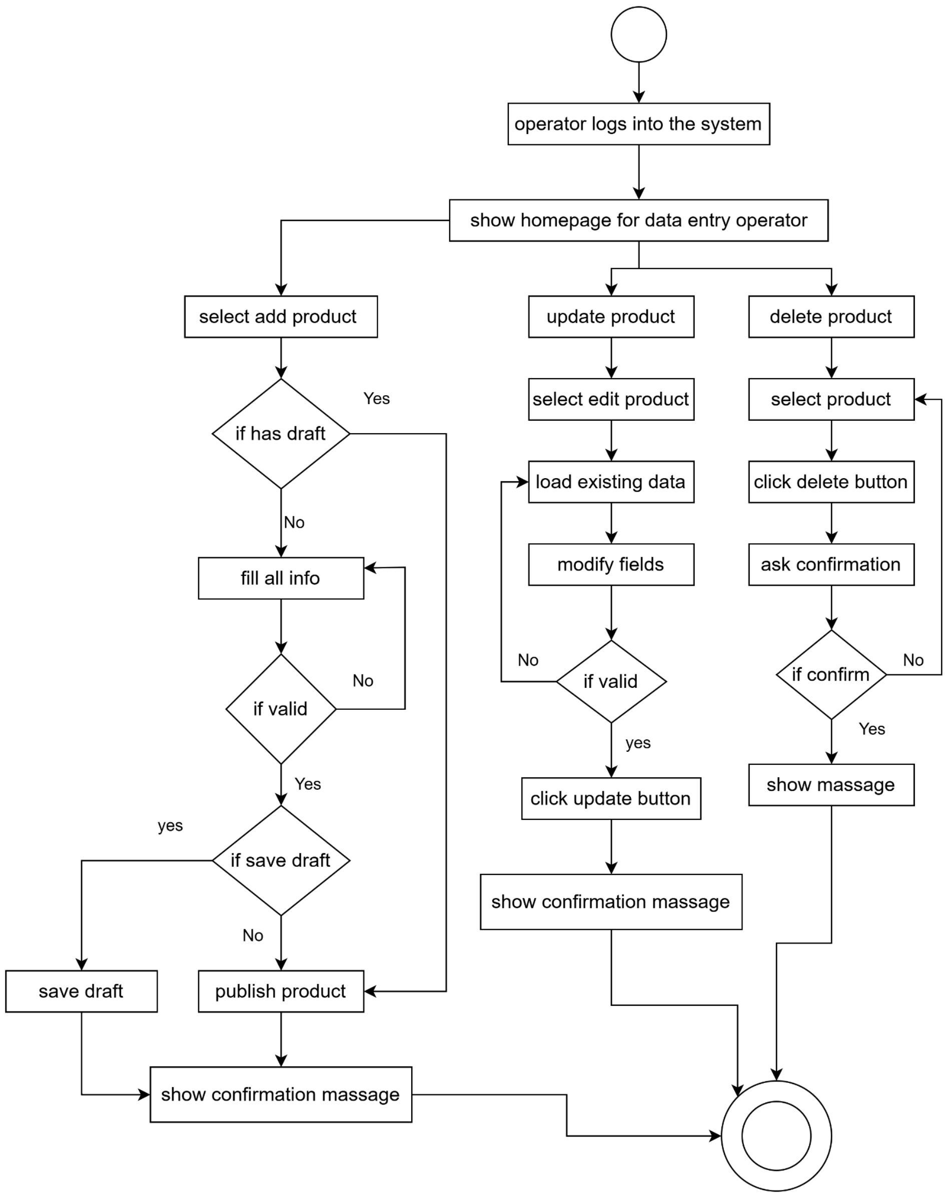


Figure 21 : Activity Diagram of Data management system

Data Based Modeling

Noun Identification

Serial no	Noun	P/S	Attribute
1	Tulona	p	
2	web application	p	
3	user	s	16, 17, 19, 23, 24, 26
4	Interface	s	
5	category	s	16,15,21
6	Bank	p	
7	Telecom Companies	p	
8	Website	p	
9	registered user	s	16, 17, 19, 23, 24
10	experience	s	
11	sign-up	s	
12	login	s	
13	profile	s	16, 17, 19, 23, 24
14	registration	s	
15	information	p	
16	name	p	
17	email	p	
18	OTP	s	20,17,19,21
19	Phone number	p	
20	code	s	
21	time	p	
22	verification	s	

23	profession	p	
24	Monthly income	p	
25	account	s	16, 17, 19, 23, 24
26	password	s	
27	message	s	
28	authentication	s	
29	Google	p	
30	Gmail	p	
31	Method	s	
32	Forget Password option	s	
33	Login page	s	
34	Home page	s	
35	Menu bar	s	
36	My profile	s	16, 17, 19, 23, 24
37	Changed password	s	
38	Setting	s	
39	Logout	s	
40	product	s	15,60,79,53,40, 21
41	comparison	s	3,40,21,53,80
42	feature	s	
43	Advanced	s	
44	Personalized suggestion	s	
45	references	s	
46	deal	p	
47	notification	s	3, 27, 21,40
48	AI	s	3, 21, 76, 77,
49	recommendation	s	

50	Credit card	p	
51	loan	p	
52	Product group	s	16,15,21
53	sub categories	s	16,15,21
54	investment	p	
55	Debit card	p	
56	Internet	p	
57	Call rate	p	
58	minute	p	
59	Bundle	p	
60	Comparison data	s	
61	graph	s	
62	APR	s	
63	rewards	s	
64	cashback	s	
65	credit limit	s	
66	eligibility criteria	s	
67	comparison table	s	
68	BB ratings	s	
69	financial growth	s	
70	stock prices	s	
71	Main Category	s	53
72	filter	s	
73	criteria	s	
74	visualization	s	
75	chatbot	s	3, 21, 76, 77,49
76	chat	s	

77	question	s	
78	feedback	s	3, 40, 79, 27,, 21
79	scale(rating)	s	
80	Companies	s	16,15, 68, 69, 70
81	data	s	
82	Data operator	s	16,17, 26, 21
83	Data modification	s	81,40, 81,21

Final Data Object:

Data object name	Attribute
User	User_id, name, email,phone_number, password_hash, profession, monthly_income, is_verified, created_at, updated_at
otp_verifications	Otp_id, user_id , otp_code, otp_type, delivery_method, created_at
companies	company_id ,name, description, website_url,metrics,updated_at
products	product_id, company_id, subcategory_id, name description, created_at, updated_at
product_attributes	attribute_id,product_id,attribute_name, attribute_value,attribute_type
product_subcategories	subcategory_id, category_id, name, description
c	category_id, name, description
notifications	notification_id, user_id, title, message, notification_type, product_id, is_read , created_at
feedback	feedback_id, user_id, product_id, rating, review_text, created_at
chatbot_conversations	conversation_id, user_id, started_at, ended_at ,status,message
data_operators	operator_id , name, email, password_hash, role, created_at
data_change_logs	log_id,operator_id,product_id ,action,old_data,new_data, timestamp

Analysis of Data Object

1. User

The User entity stores information about all registered individuals in the Tulona Comparison System. Each user is uniquely identified by user_id. Attributes such as name, email, phone_number, profession, and monthly_income capture the user's personal and demographic details, which the system later uses for personalized recommendations.The email and phone_number fields also serve as unique identifiers for account creation and secure login.The password_hash attribute stores the securely encrypted password to protect user access. The is_verified attribute indicates whether the user has successfully completed OTP-based identity verification. Attributes like created_at and updated_at help track account activity and support account management.This entity ensures role-limited access to features—registered users gain access to advanced functionalities such as personalized suggestions, deal notifications, and AI-based recommendations, whereas unregistered users have limited access.

2. otp_verifications

The OTP Verification entity manages the secure authentication process of the Tulona system. Attributes such as otp_code, user_id, otp_type, and delivery_method (Email/SMS) record the details of the OTP sent to users. Each OTP entry is timestamped using created_at, which allows the system to enforce validity checks. The otp_type specifies why the OTP was generated—registration, login validation, or password reset—ensuring purpose-specific verification. This entity ensures that each authentication step is time-bound, secure, and aligned with the system's verification requirements.

3. Companies

The Companies entity represents all service providers available in the Tulona platform, such as Banks, Telecom Operators, and other institutions. Each company is uniquely identified by company_id. Attributes like name, description, website_url, and is_active provide essential company information and indicate whether the company is currently available for comparison. The entity serves as the parent unit for product listings, financial metrics, and comparison functionality.

4. company_metrics

The Company Metrics entity stores performance indicators for each company. Examples include BB ratings, stock prices, financial growth, market stability, etc. Each metric is represented by metric_name and metric_value, while updated_at records when the metric was last refreshed. These metrics directly support visual insights through comparison graphs, enabling users to evaluate companies based on real-time or periodically updated analytics.

5. categories

The Categories entity defines top-level service categories in the system, such as Banks, Telecom Operators, and future segments like Insurance or Investments. Each category is uniquely identified using category_id. Attributes like name, description help classify and control visibility of these categories. This generic design supports expansion, allowing new categories to be added without modifying the database structure.

6. product_subcategories

The Product Subcategories entity defines category-specific groupings such as Cards, Loans, Deposits under Banks and Internet Packages, Call Rates, Bundles under Telecom Operators. Each subcategory is linked to a specific category through category_id. This hierarchical structure enables a flexible navigation flow (e.g., Bank → Cards → Credit Cards). The is_active attribute helps control visibility and manage deprecated subcategories without system-level changes.

7. products

The Products entity stores individual products offered by companies—such as Credit Cards, Personal Loans, Internet Packages, and Deposit Schemes. Each product is uniquely identified using product_id. Attributes like company_id, subcategory_id, name, description, and timestamps ensure proper categorization and traceability. This entity acts as the central reference for product attributes, user feedback, comparison tables, and application links.

8. product_attributes

The Product Attributes entity is designed using an EAV (Entity–Attribute–Value) model, allowing unlimited and dynamic product specifications. Each attribute is represented by attribute_name, attribute_value, and attribute_type (text, number, percentage, etc.). For example, a credit card may have attributes like annual_fee, cashback_rate, APR, credit_limit, while an internet package may include data_volume, validity, speed. This flexible structure allows the system to introduce new product types or features without schema modification.

9. notifications

The Notifications entity manages personalized alerts sent to users. Each notification includes a title, message, and notification_type, and is linked to a specific user_id.

Notifications include:

- Personalized deal suggestions
- New offers from banks or telecom companies
- Product updates based on user preferences

The is_read attribute ensures accurate tracking of viewed and pending notifications.

10. feedback

The Feedback entity collects user ratings and reviews for specific products. Each feedback entry includes rating (out of 5) and review_text, representing real user experiences. This data helps future users make educated choices based on both technical specifications and user opinions. It also enhances the credibility of comparison results.

11. chatbot_conversations

This entity manages AI chatbot session tracking. Attributes like conversation_id, user_id, started_at, ended_at, and status record each interaction session's lifecycle. Tracking conversations enables continuity, reporting, and personalized assistance throughout the system.

12. chatbot_messages

The Chatbot Messages entity stores individual messages exchanged between the user and the AI chatbot. Attributes include message_text, sender_type, and recommended_products, enabling the system to deliver smart comparison suggestions. This entity enhances personalized navigation and ensures context-aware recommendations.

13. data_operators

The Data Operators entity represents backend administrative users responsible for adding, updating, or deleting product data. Attributes such as name, email, role, and password_hash help identify operators and secure their access. This entity ensures controlled management of sensitive product data across all categories.

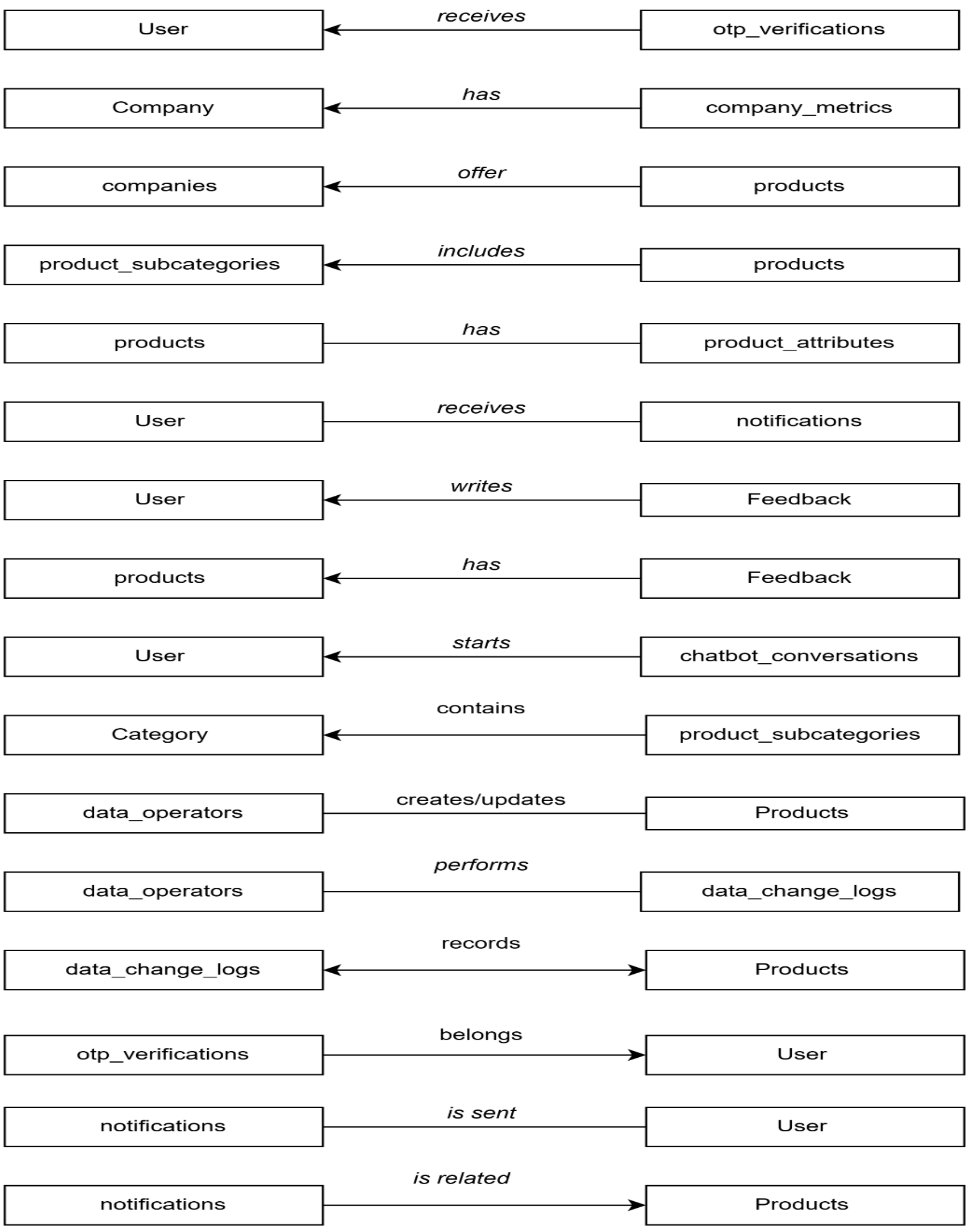
14. data_change_logs

This entity implements auditing for all product changes made by data operators.
Each log entry includes:

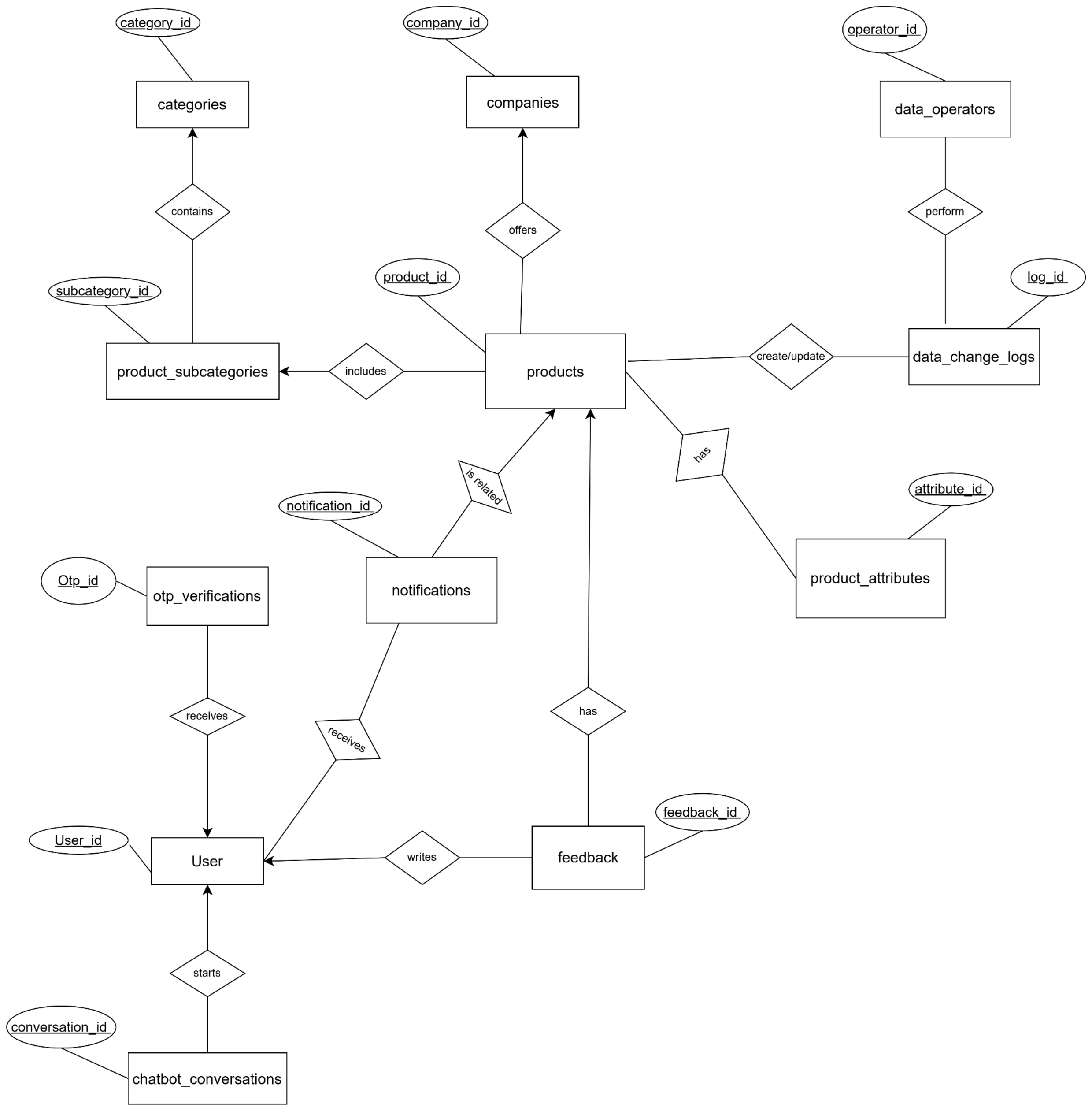
- operator_id – who made the change
- product_id – which product was affected
- action – create/update/delete
- old_data / new_data – before-after comparisons
- timestamp – when the change occurred

This ensures transparency, accountability, and traceability within the system.

Relational Model



ER Diagram


category_id
categories
contains
subcategory_id
product_subcategories
company_id
companies
offers
product_id
products
includes
operator_id
data_operators
perform
log_id
data_change_logs
create/update
has
attribute_id
product_attributes
is related
notification_id
notifications
Otp_id
otp_verifications
receives
receives
has
feedback_id
feedback
writes
User_id
User
starts
conversation_id
chatbot_conversations

Schema Table

1. users

Attribute	Type	Key
user_id	NUMBER	PK
name	VARCHAR(100)	-
phone_number	VARCHAR(100)	-
password_hash	VARCHAR(100)	-
profession	VARCHAR(100)	-
monthly_income	VARCHAR(100)	-
is_verified	BOOLEAN	-
created_at	DATETIME	-
updated_at	DATETIME	-

2. otp_verifications

Attribute	Type	Key
otp_id	NUMBER	PK
user_id	NUMBER	FK
otp_code	VARCHAR(100)	-
otp_type	VARCHAR(100)	-
delivery_method	VARCHAR(100)	-
created_at	DATETIME	-

3. companies

Attribute	Type	Key
company_id	NUMBER	PK
website_url	VARCHAR(100)	-
name	VARCHAR(100)	-
description	VARCHAR(100)	-
metric	JSON	-
updated_at	DATETIME	-

4. products

Attribute	Type	Key
product_id	NUMBER	PK
company_id	NUMBER	FK
subcategory_id	NUMBER	FK
name	VARCHAR(100)	-
description	VARCHAR(100)	-
created_at	DATETIME	-
updated_at	DATETIME	-

5. product_attributes

Attribute	Type	Key
attribute_id	NUMBER	PK
product_id	NUMBER	FK
attribute_name	VARCHAR(100)	-
attribute_name	VARCHAR(100)	-

attribute_type	VARCHAR(100)	-

6. product_subcategories

Attribute	Type	Key
subcategory_id	NUMBER	PK
category_id	NUMBER	FK
name	VARCHAR(100)	-
description	VARCHAR(100)	-

7. categories

Attribute	Type	Key
category_id	NUMBER	PK
name	VARCHAR(100)	-
description	VARCHAR(100)	-

8. notifications

Attribute	Type	Key
notification_id	NUMBER	PK
user_id	NUMBER	FK
title	VARCHAR(100)	-
message	VARCHAR(100)	-
product_id	NUMBER	FK
is_read	BOOLEAN	-
created_at	DATETIME	-

9. feedback

Attribute	Type	Key
feedback_id	NUMBER	PK
user_id	NUMBER	FK
product_id	NUMBER	FK
rating	NUMBER	-
review_text	VARCHAR(100)	-
created_at	DATETIME	-

10. chatbot_conversations

Attribute	Type	Key
conversation_id	NUMBER	PK
user_id	NUMBER	FK
started_at	DATETIME	-
ended_at	DATETIME	-
status	VARCHAR(100)	-
message	VARCHAR(100)	-

11. data_operators

Attribute	Type	Key
operator_id	NUMBER	PK
name	VARCHAR(100)	-
email	VARCHAR(100)	-

password_hash	VARCHAR(100)	-
role	VARCHAR(100)	-
created_at	DATETIME	-

12. data_change_logs

Attribute	Type	Key
log_id	NUMBER	PK
operator_id	NUMBER	FK
product_id	NUMBER	FK
action	VARCHAR(100)	-
old_data	VARCHAR(100)	-
new_data	VARCHAR(100)	-
timestamp	DATETIME	-

Class Based Modeling

General Classification:

1) External Entities
2) Things
3) Occurrences or Events
4) Roles
5) Organizational Unit
6) Places
7) Structures

Noun List

Noun	GC
web application	7,2
user	4,2
Interface	7,2
category	2
Bank	5,1
Telecom Companies	5,1
Website	7,2
registered user	4
experience	2
sign-up	3
login	3
profile	2,7
registration	3
information	2

name	2
email	2
OTP	2,3
Phone number	2
code	2
time	2
verification	3
profession	2
Monthly income	2
account	2
password	2
message	2
authentication	3,2
Google	1
Gmail	1
Method	2
Forget Password option	3
Login page	7
Home page	7
Menu bar	7
My profile	7,2
Changed password	3
Setting	7
Logout	3
product	2
comparison	3
feature	2

Advanced Features	2
Personalized suggestion	2
references	2
deal	2
notification	3
AI	7
recommendation	2
Credit card	2
loan	2
Product group	2
sub categories	2
investment	2
Debit card	2
Internet	2
Call rate	2
minute	2
Bundle	2
Comparison data	2
graph	7
APR	2
rewards	2
cashback	2
credit limit	2
eligibility criteria	2
comparison table	7
BB ratings	2
financial growth	2

stock prices	2
Main Category	2
filter	2
criteria	2
visualization	2,7
chatbot	7
chat	2,3
question	2
feedback	2,3
scale(rating)	2
Companies	5,1
data	2
Data operator	4
Data modification	3

Potential Class List:

1. User
2. Registration
3. Login
4. Authentication
5. Profile
6. ProductCategory
7. ProductGroup
8. SubCategory
9. Product
10. Bank
11. CreditCard
12. DebitCard
13. LoanProduct
14. DepositProduct
15. Telecom.
16. Comparison
17. GraphVisualization

18. Filter
19. UserFilterSelection
20. AI_Chatbot
21. Feedback
22. Rating
23. OTP
24. Notification
25. PersonalizedOffer
26. DealNotification
27. DataOperator
28. DataModificationLog
29. RecommendationEngine
30. DraftProduct
31. Company
32. ForgetPassword

Selection Criteria:

1) Retained Information

2) Needed Services

3) Multiple Attributes

4) Common Attributes

5) Common Operations

6) Essential Requirements

class	**SC**
User	1,2,3,5,6
Registration	2,3,5
Login	2,3,5
Authentication	2,3,5,6
Profile	1,2,3

ProductCategory	1,3,5,6
ProductGroup	1,3,5
SubCategory	1,3,5
Product	1,2,3,4,5,6
Bank	1,2,3,5
CreditCard	1,2,3,5
DebitCard	1,2,3,5
LoanProduct	1,2,3,5,6
DepositProduct	1,2,3,5,6
Telecom	1,2,3,5
Comparison	2,3,5,6
GraphVisualization	1,3,4,5
Filter	2,3,6
UserFilterSelection	2,3,6
AI_Chatbot	1,2,3,4,5
Feedback	1,2,3
Rating	1,2,3
OTP	1,2,3,6
Notification	1,2,3,6
PersonalizedOffer	1,2,3,5
DealNotification	1,2,3,6
RecommendationEngine	1,2,3,4,5
DataOperator	2,3,4,6
DataModificationLog	1,2,3,6

DraftProduct	1,2,3,6
Company	1,2,3,5,6
ForgetPassword	2,3,5,6

List of Verbs:

Serial no	Verb
1	maintain
2	access
3	receive
4	fill
5	save
6	publish
7	store
8	log
9	edit
10	ensure
11	present
12	click
13	display
14	select
15	enter
16	open
17	choose

18	send
19	validate
20	match
21	verify
22	complete
23	create
24	confirm
25	register
26	login
27	reset
28	submit
29	check
30	finalize
31	redirect
32	update
33	browse
34	view
35	navigate
36	compare
37	apply
38	filter
39	refine
40	visualize
41	ask
42	rate

43	read
44	load
45	show
46	generate
47	fetch
48	restrict
49	provide
50	notify
51	appear
52	initiate
53	suggest
54	include
55	prompt
56	authenticate
57	revise

Final classes

1. User
2. OTP
3. AuthService
4. Profile
5. Company
6. ProductCategory
7. ProductGroup
8. Product
9. Comparison
10. Filter
11. AI_Chatbot

12. Feedback
13. Notification
14. DataOperator
15. DataModificationLog

Selected Classes

1) User

- **Attributes:**userId,name,email,phoneNumber,passwordHash,profession,monthlyIncome,isVerified, registrationDate, lastLogin
- **Methods:**register(),login(),updateProfile(profileData),requestPasswordReset(),logout(),verifyAccount(), deleteAccount()

2)OTP

- **Attributes:**otpId,userId,email,phoneNumber,otpCode,deliveryMethod,purpose,generatedAt,expiresAt,isUsed
- **Methods:**generateOTP(),sendOTP(deliveryMethod),verifyOTP(inputCode),isExpired(),invalidateOTP(),resendOTP()

3) AuthService

- **Attributes:**authId,userId,authMethod,resetToken,otpId,status,createdAt
- **Methods:**registerUser(userData),login(credentials),authenticate(credentials),initiatePasswordReset(emailOrPhone),resetPassword(newPassword)

4) Profile

- **Attributes:** profileId,userId,profession,monthlyIncome,profilePicture,updatedAt
- **Methods:**updateProfession(profession),updateIncome(income),uploadProfilePicture(image),getCompleteProfile()

5) Company

- **Attributes:**companyId,companyName,companyType,description,logoUrl,websiteUrl,contactEmail,contactPhone,headquarters,establishedYear,isActive,createdAt
- **Methods:**getProducts(categoryId),getProductDetails(productId),updateCompanyInfo(data),uploadLogo(image)activateCompany(),deactivateCompany(),getCompanyStats()

6) ProductCategory

- **Attributes:**categoryId,categoryName,description,iconUrl,displayOrder,isActive,createdAt
- **Methods:**getProductGroups(),getCompanies(),activateCategory(),deactivateCategory(),updateCategoryInfo(data),reorderDisplay(newOrder)

7) ProductGroup

- **Attributes:**groupId,categoryId,groupName,description,displayOrder,isActive
- **Methods:**getSubCategories(),addSubCategory(),removeSubCategory(),updateGroupInfo(data),reorderDisplay(newOrder)

8) Product

- **Attributes:**productId,companyId,categoryId,productName,description,isActive,applyUrl,createdAt,updatedAt
- **Methods:**getDetails(),getComparisonData(),applyForProduct(),activateProduct(),deactivateProduct(),updateProductInfo(data)

9) Comparison

- **Attributes:**comparisonId,userId,categoryId,productIds,createdAt,status,comparisonType
- **Methods:**addProduct(productId),removeProduct(productId),generateComparison(),generateGraph()

10) Filter

- **Attributes:**filterId,categoryId,filterName,filterType,minValue,maxValue,options,displayOrder,isActive
- **Methods:**applyFilter(products,criteria),validateFilterValue(value),getFilterOptions(),updateFilterConfig(data),activateFilter(),deactivateFilter()

11) AI_Chatbot

- **Attributes:**chatbotId,userId,sessionId,conversationHistory,userIntent,contextData,isActive,startedAt,lastInteractionAt
- **Methods:**processQuery(query),generateRecommendation(userProfile),analyzeUserIntent(query),getConversationContext(),suggestProducts(criteria),endSession(),exportConversation()

12) Feedback

- **Attributes:**feedbackId,userId,productId,rating,reviewTitle,reviewText,isVerified,helpfulCount,reportCount,submittedAt,updatedAt
- **Methods:**submitFeedback(),updateFeedback(data),deleteFeedback(),markAsHelpful(),reportAbuse(),verifyFeedback(),getFeedbackStats()

13) Notification

- **Attributes:**notificationId,userId,notificationType,title,message,relatedProductId,relatedCompanyId,priority,isRead,sentAt,readAt,expiresAt
- **Methods:**sendNotification(),markAsRead(),deleteNotification(),getUnreadCount(),scheduleNotification(sendTime),filterByType(type)

14) DataOperator

- **Attributes:**operatorId,name,email,passwordHash,role,permissions,isActive,createdAt,lastLogin
- **Methods:**login(credentials),addProduct(productData),updateProduct(productId,data),deleteProduct(productId),saveDraft(productData),publishDraft(draftId),viewLogs()moderateFeedback(feedbackId)

15) DataModificationLog

- **Attributes:**logId,operatorId,entityType,entityId,action,previousData,newData,changeDescription,timestamp,ipAddress,userAgent
- **Methods:**logAction(),getChangeHistory(entityId),revertChange(logId),exportLogs(dateRange),searchLogs(criteria),auditOperator(operatorId)

CRC Card

1. User

Attributes	Methods

userId
Name
Email
phoneNumber
passwordHash
Profession
	resister()
	login()
	updateProfile(profileData)
	requestPasswordReset()
	logout()
	verifyAccount()
	deleteAccount()

monthlyIncome isVerified registrationDate lastLogin	
Responsibilities	**Collaborations**
1. Store and manage user account information 2. Maintain verification status 3. Hold user personal and financial data	1. AuthService 2. OTP 3. Profile 4. Comparison 5. Feedback 6. Notification 7. AI_Chatbot

2. OTP

Attributes	**Methods**

otpId
userId
email
phoneNumber
otpCode
deliveryMethod
purpose
generatedAt
expiresAt
isUsed
	generateOTP()
	sendOTP(deliveryMethod)
	verifyOTP(inputCode)
	isExpired()
	invalidateOTP()
	resendOTP()

Responsibilities	**Collaborations**
1. Generate secure 6-digit OTP codes 2. Send OTP via email or SMS 3. Verify user-entered OTP 4. Manage OTP expiration 5. Prevent brute force attacks 6. Handle OTP resend requests	1. AuthService 2. User

3. AuthService

Attributes	**Methods**

authId
userId
authMethod
resetToken
otpId
status
createdAt
	registerUser(userData)
	login(credentials)
	authenticate(credentials)
	initiatePasswordReset(emailOrPhone)
	resetPassword(newPassword)

Responsibilities	**Collaborations**
1. Handle user registration workflow 2. Authenticate users securely 3. Manage password reset process 4. Coordinate OTP-based verification 5. Control login/logout lifecycle	1. User 2. OTO

4. Profile

Attributes	**Methods**

profileId
userId
profession
monthlyIncome
profilePicture
updatedAt
	updateProfession(profession)
	updateIncome(income)
	uploadProfilePicture(image)
	getCompleteProfile()

Responsibilities	**Collaborations**
1. Store detailed user profile information 2. Manage user preferences and settings 3. Handle profile updates 4. Provide personalization data 5. Store notification preferences	1. User 2. Notification

5. Company

Attributes	Methods

companyId
companyName
companyType
description
logoUrl
websiteUrl
contactEmail
contactPhone
headquarters
establishedYear
isActive
createdAt
	getProducts(categoryId)
	getProductDetails(productId)
	updateCompanyInfo(data)
	uploadLogo(image)
	activateCompany()
	deactivateCompany()
	getCompanyStats()

Responsibilities	**Collaborations**
1. Represent service providers 2. Store organization-level information 3. Manage company availability 4. Act as product owner	1. Product 2. Comparison 3. DataModificationLog

6. ProductCategory

Attributes	Methods

categoryId
categoryName
description
iconUrl
displayOrder
isActive
createdAt
	getProductGroups()
	getCompanies()
	activateCategory()
	deactivateCategory()
	updateCategoryInfo(data)
	reorderDisplay(newOrder)

Responsibilities	**Collaborations**

1. Organize products into main categories 2. Manage category hierarchy 3. Provide navigation structure 4. Control category visibility 5. Store category metadata	1. ProductGroup 2. Company 3. Filter

7. ProductGroup

Attributes	**Methods**

groupId
categoryId
groupName
description
displayOrder
isActive
	getSubCategories()
	addSubCategory()
	removeSubCategory()
	updateGroupInfo(data)
	reorderDisplay(newOrder)

Responsibilities	**Collaborations**
1. Group related products together 2. Organize subcategories 3. Manage product group hierarchy 4. Control group visibility	1. ProductCategory 2. Product

8. Product

Attributes	**Methods**

productId
companyId
categoryId
productName
description
isActive
applyUrl
createdAt
updatedAt
	getDetails()
	getComparisonData()
	applyForProduct()
	activateProduct()
	deactivateProduct()
	updateProductInfo(data)

Responsibilities	**Collaborations**
1. Serve as base class for all product types 2. Store common product attributes	1. Company 2. ProductCategory

3. Provide common product operations 4. Manage product lifecycle 5. Handle product eligibility checks	3. Comparison 4. Feedback 5. Filter

9. Comparison

Attributes	**Methods**

comparisonId
userId
categoryId
productIds
createdAt
status
comparisonType
	addProduct(productId)
	removeProduct(productId)
	generateComparison()
	generateGraph()

Responsibilities	**Collaborations**
1. Manage product selection 2. Generate comparison results 3. Produce visual comparison graphs	1. User 2. Product

10. Filter

Attributes	**Methods**

filterId
categoryId
filterName
filterType
minValue
maxValue
options
displayOrder
isActive
	applyFilter(products,criteria)
	validateFilterValue(value)
	getFilterOptions()
	updateFilterConfig(data)
	activateFilter()
	deactivateFilter()

Responsibilities	**Collaborations**
1. Define available filter criteria 2. Filter products based on user selection	1. Product 2. ProductCategory

3. Validate filter inputs 4. Manage filter configurations 5. Support multiple filter types	

11. AI_Chatbot

Attributes	**Methods**

chatbotId
userId
sessionId
conversationHistory
userIntent
contextData
isActive
startedAt
lastInteractionAt
	processQuery(query)
	generateRecommendation(userProfile)
	analyzeUserIntent(query)
	getConversationContext()
	suggestProducts(criteria)
	endSession()
	exportConversation()

Responsibilities	**Collaborations**
1. Handle user queries via chat 2. Analyze user intent and context 3. Generate personalized recommendations 4. Maintain conversation history 5. Provide intelligent product suggestions	1. User 2. Product 3. Comparison

12. Feedback

Attributes	**Methods**

feedbackId
userId
productId
rating
reviewTitle
reviewText
isVerified
	submitFeedback()
	updateFeedback(data)
	deleteFeedback()
	markAsHelpful()
	reportAbuse()
	verifyFeedback()
	getFeedbackStats()

helpfulCount reportCount submittedAt updatedAt	
Responsibilities	**Collaborations**
1. Store user product reviews 2. Manage rating system (1-5 scale) 3. Track helpful votes 4. Handle feedback moderation 5. Provide feedback statistics	1. User 2. Product

13. Notification

Attributes	**Methods**

notificationId
userId
notificationType
title
message
relatedProductId
relatedCompanyId
priority
isRead
sentAt
readAt
expiresAt
	sendNotification()
	markAsRead()
	deleteNotification()
	getUnreadCount()
	scheduleNotification(sendTime)
	filterByType(type)

Responsibilities	**Collaborations**
1. Send personalized notifications to users 2. Store notification content and metadata 3. Track read/unread status 4. Manage notification priority 5. Handle notification expiration 6. Provide notification filtering	1. User 2. Profile 3. product 4. Company

14. DataOperator

Attributes	Methods

operatorId
name
email
passwordHash
role
permissions
isActive
createdAt
lastLogin
	login(credentials)
	addProduct(productData)
	updateProduct(productId,data)
	deleteProduct(productId)
	saveDraft(productData)
	publishDraft(draftId)
	viewLogs()
	moderateFeedback(feedbackId)

Responsibilities	**Collaborations**
1. Maintain system data quality 2. Manage product information 3. Moderate user content	1. Product 2. DataModificationLog 3. Company

15. DataModificationLog

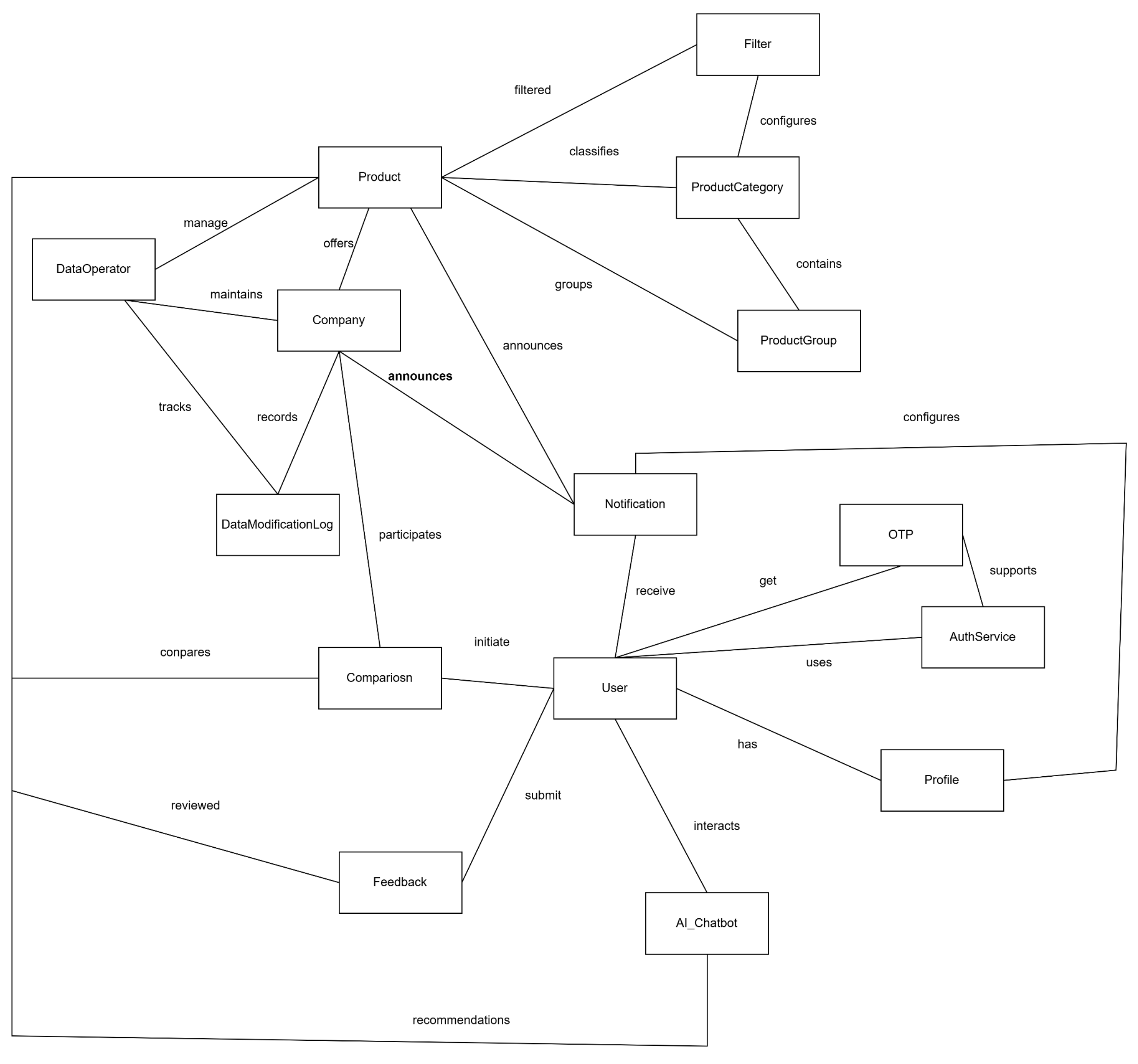
Attributes	Methods

logId
operatorId
entityType
entityId
action
previousData
newData
changeDescription
timestamp
ipAddress
userAgent
	logAction()
	getChangeHistory(entityId)
	revertChange(logId)
	exportLogs(dateRange)
	searchLogs(criteria)
	auditOperator(operatorId)

Responsibilities	**Collaborations**
1. Track all data modifications 2. Store before/after snapshots 3. Maintain audit trail	1. DataOperator 2. Product

4. Enable change reversion 5. Support compliance reporting 6. Monitor operator activities	3. Company

CRC Diagram



Behavioural Modeling

Event Identification

Event	Event name	Initiator Class	Collaborator Class
Open Sign Up		User	AuthService
Submit Basic Info		User	AuthService
Select OTP Delivery Method		User	
Send OTP		OTP	User
Verify OTP		OTP	AuthService
complete Profile Info		User	
set Account Password		User	
Finalize Registration		AuthService	User
Login User		User	AuthService
Login via Google		AuthService	User
Initiate Forgot Password		User	AuthService
Reset Password		AuthService	User
Logout User		User	AuthService
Load Homepage		User	ProductCategory
View Categories		ProductCategory	
Select Category		User	ProductCategory
View Product Groups		ProductGroup	user

View Product Details		Product, Company	
Add Product to Comparison		Comparison	Product
Generate Comparison Table		Comparison	Product
Generate Comparison Graph		Comparison	Product
Apply Filter Criteria		Filter	Product
Generate Personalized Offer		Filter	Product
Send Deal Notification		Notification	User, Product
Open Chatbot		User	AI_Chatbot
Submit User Query		User	AI_Chatbot
Suggest Products		AI_Chatbot	Product
Submit Product Feedback		User	Feedback
Store Rating		Feedback	Product
View Feedback		User	Feedback
Receive Notification		User	Notification
Mark Notification as Read		User	Notification
Access Data Management Interface		DataOperator	
Create Product Entry		DataOperator	Product, Company
Update Product Entry		DataOperator	Product

Delete Product Entry		DataOperator	Product
Publish Product		DataOperator	Product
Log Data Modification		DataOperator	DataModificationLog
Edit Logged Data		DataOperator	DataModificationLog