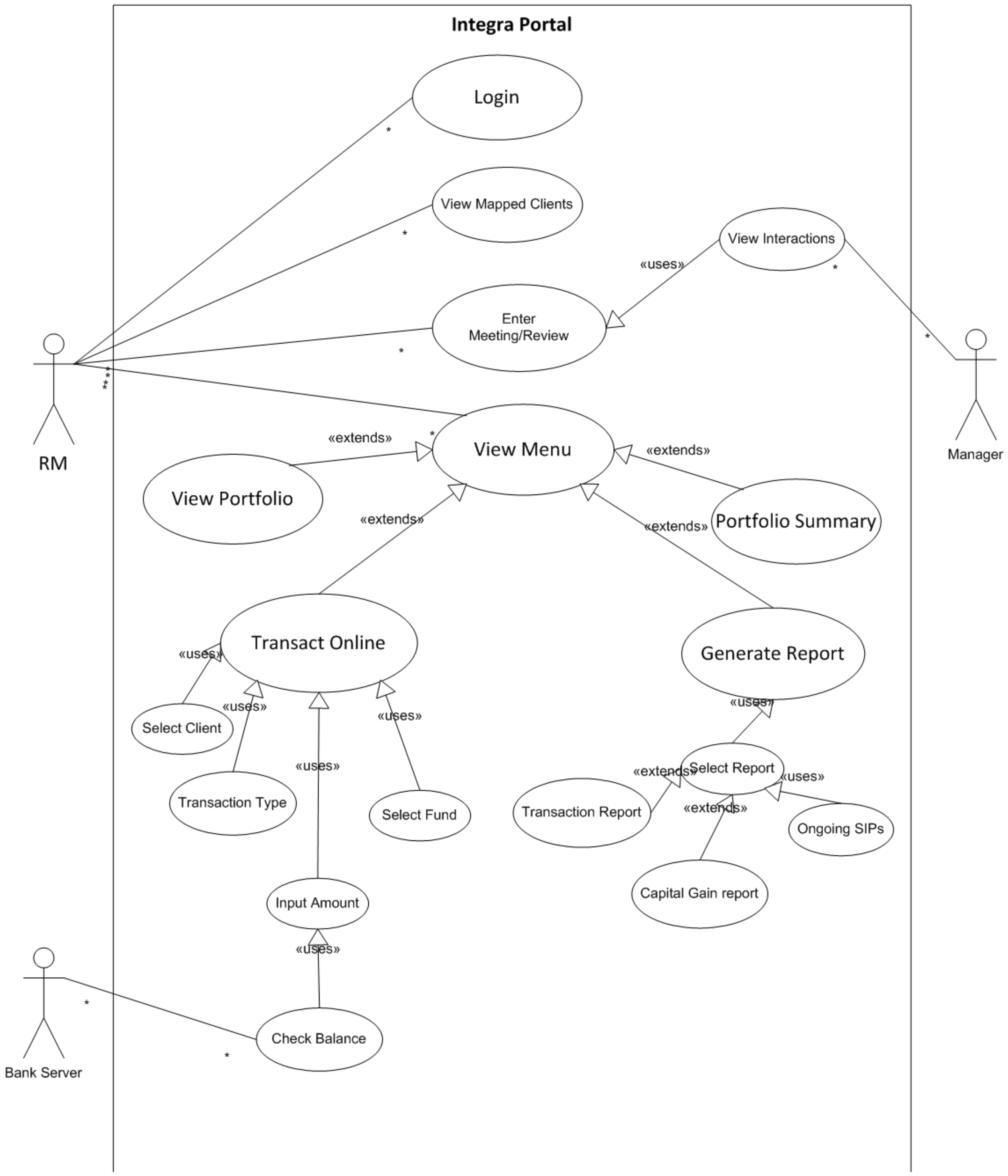
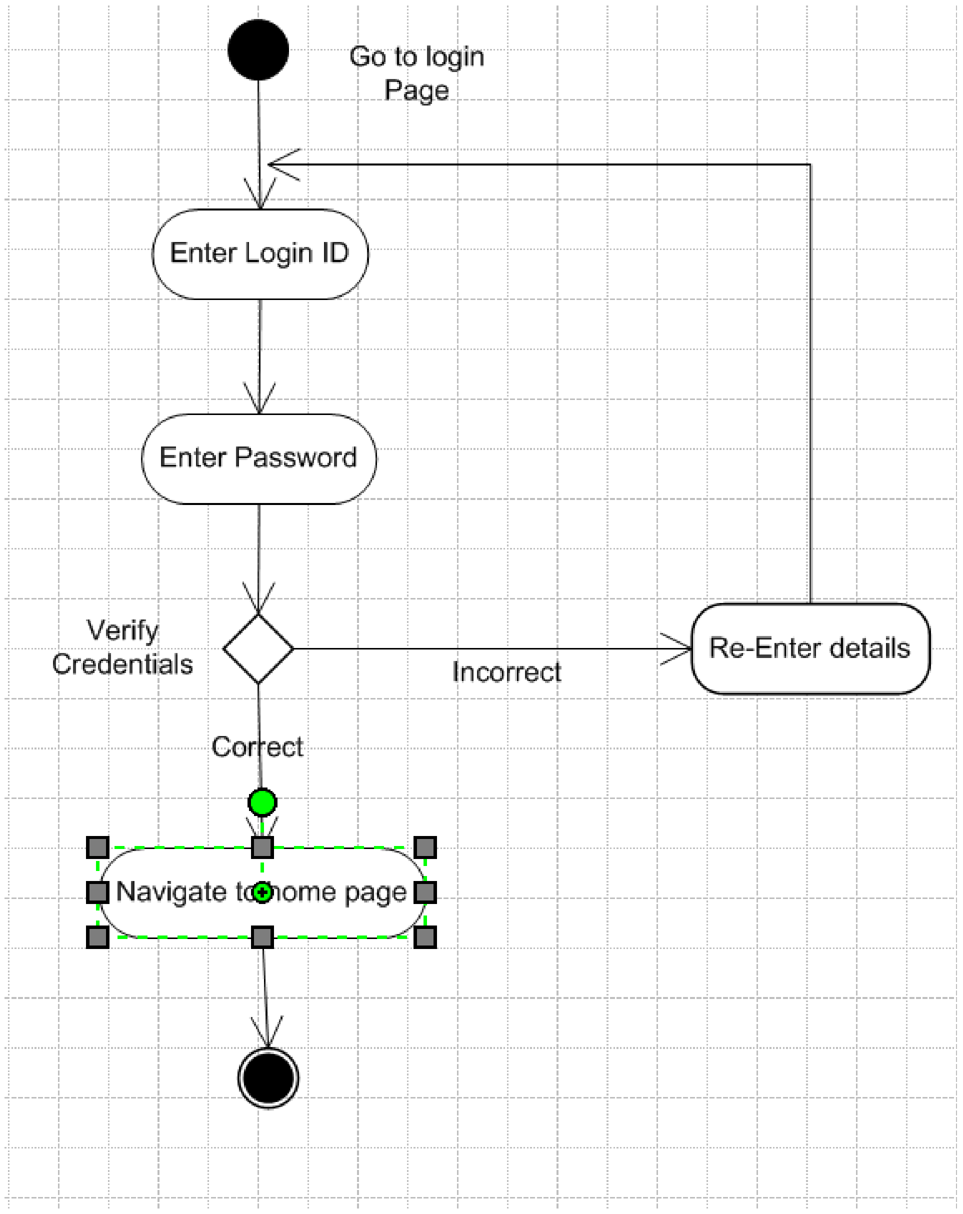
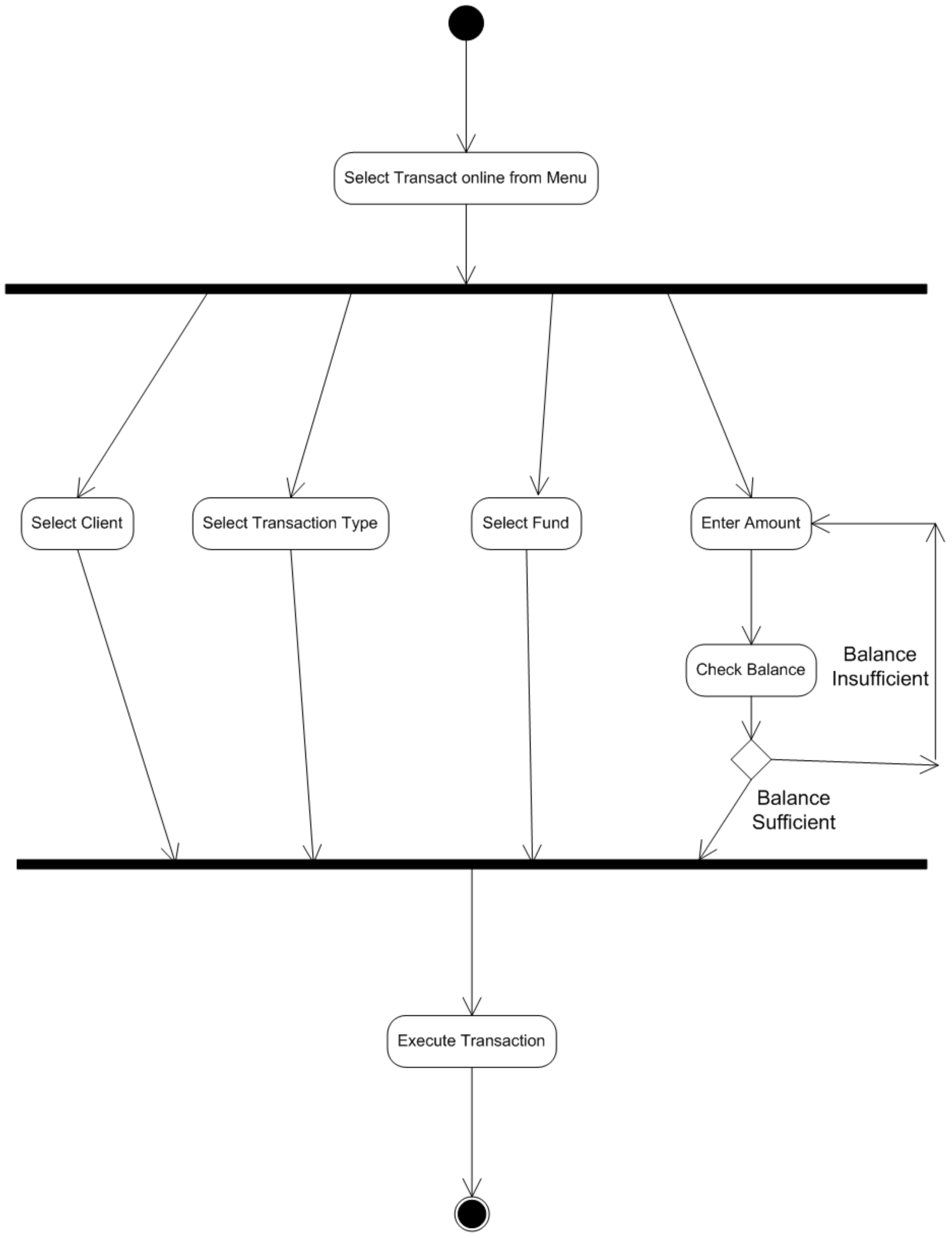
Document 6- Please prepare a use case diagram, activity diagram and a use case specification document.

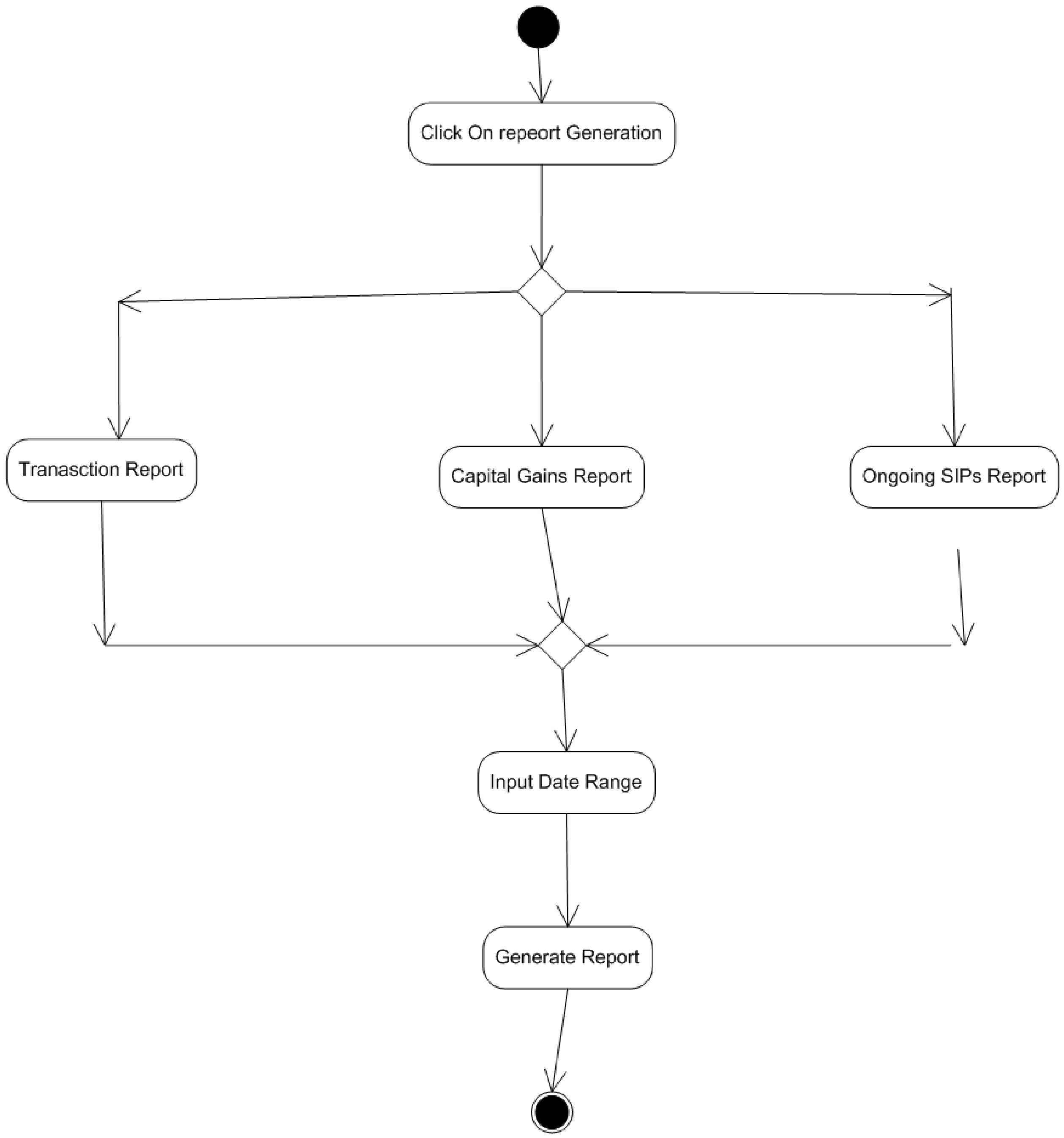
Use case diagram



Activity Diagram




Select Transact online from Menu
Select Client
Select Transaction Type
Select Fund
Enter Amount
Check Balance
Balance Insufficient
Balance Sufficient
Execute Transaction


Click On repeort Generation
Tranasction Report
Capital Gains Report
Ongoing SIPs Report
Input Date Range
Generate Report

Use case specification document

Use Case:User Login

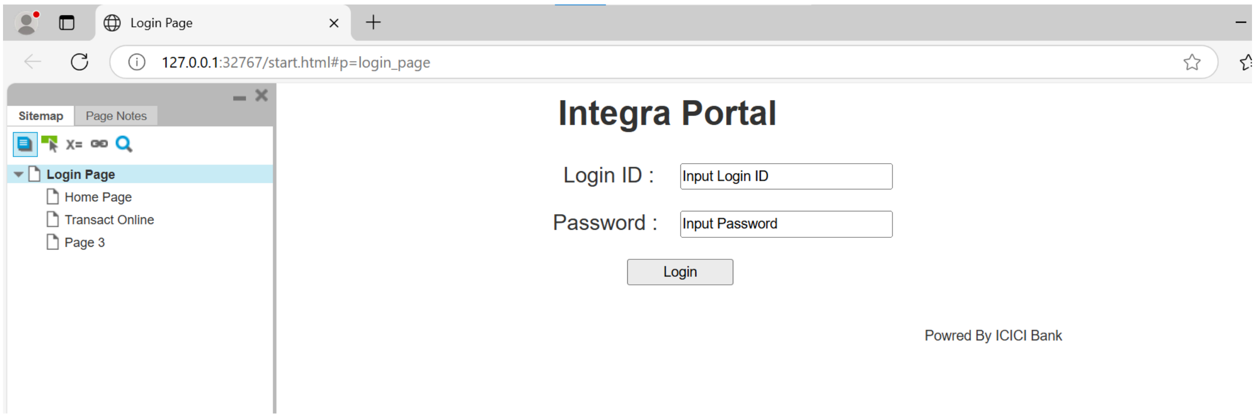
Field	Description
Use Case Name	User Login
Use Case Description	This use case describes the login process for clients and Relationship Managers (RMs) to access the platform using their User ID and password.
Actors	Primary Actors: Client, Relationship Manager (RM) Secondary Actors: Authentication System
Basic Flow	1. The user enters the platform URL. 2. The system prompts the user for their User ID and password. 3. The user enters valid credentials. 4. The system validates credentials. 5. The system logs in the user and redirects to the dashboard.
Alternate Flow	1. If the user enters incorrect credentials, the system shows an error message and allows the user to try again.
Exceptional Flows	1. If the system is unavailable, the user sees a "Service Unavailable" message. 2. If the User ID is locked, the user receives an "Account Locked" message.
Pre-Conditions	1. The user must have a valid account. 2. The system must be accessible.
Post-Conditions	1. The user is logged in successfully. 2. The user is redirected to the dashboard.
Assumptions	1. The user has an active internet connection. 2. The user knows their credentials.
Constraints	1. User ID and password must adhere to security guidelines (length, complexity, etc.).
Dependencies	1. The authentication system must be functional.
Inputs and Outputs	Inputs: User ID, password. Outputs: Login success message, user dashboard.
Business Rules	1. User passwords must be encrypted in the database. 2. Users should be locked out after 3 failed login attempts.
Miscellaneous Information	1. The login page should support two-factor authentication (if enabled for the user).

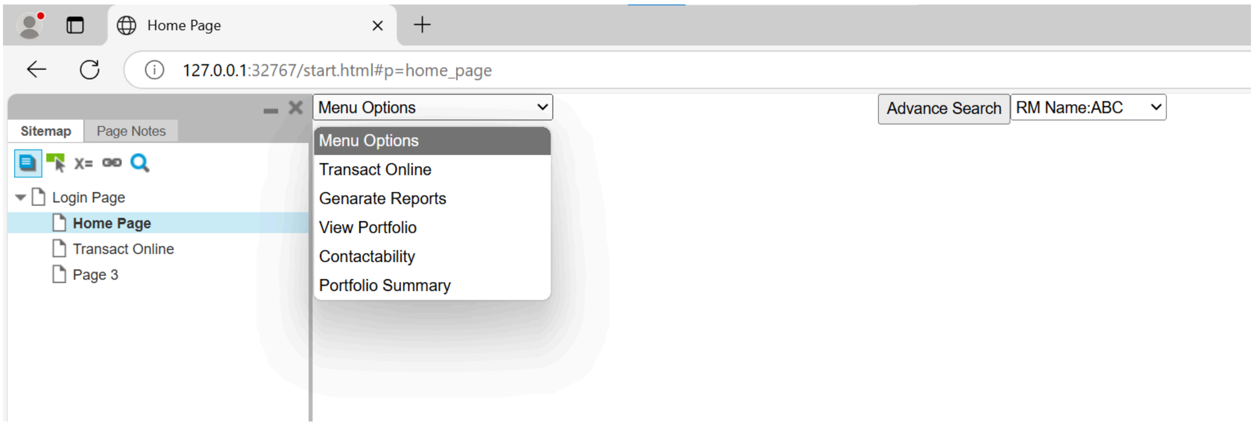
Use Case:Capital Gains Report Generation

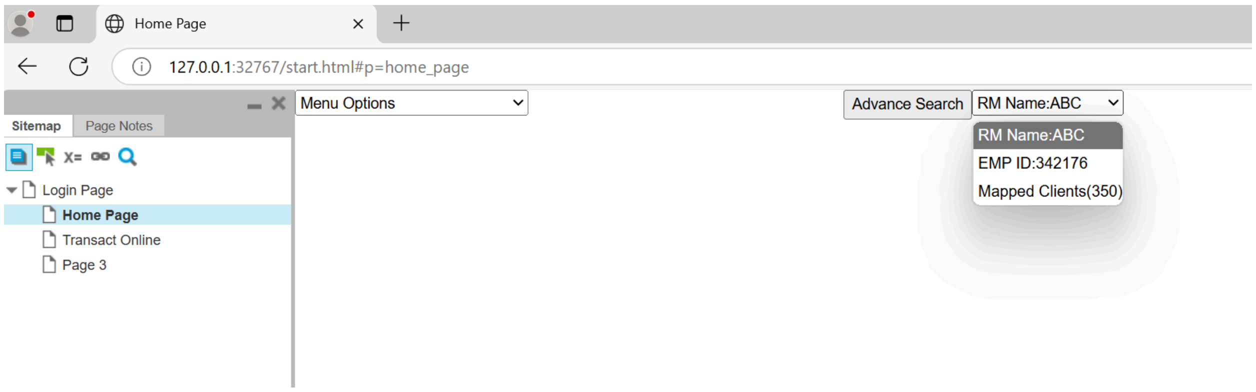
Field	Description
Use Case Name	Generate Capital Gain Reports
Use Case Description	This use case allows the client or Relationship Manager (RM) to generate detailed capital gain reports, which include short-term and long-term gains for tax filing purposes.

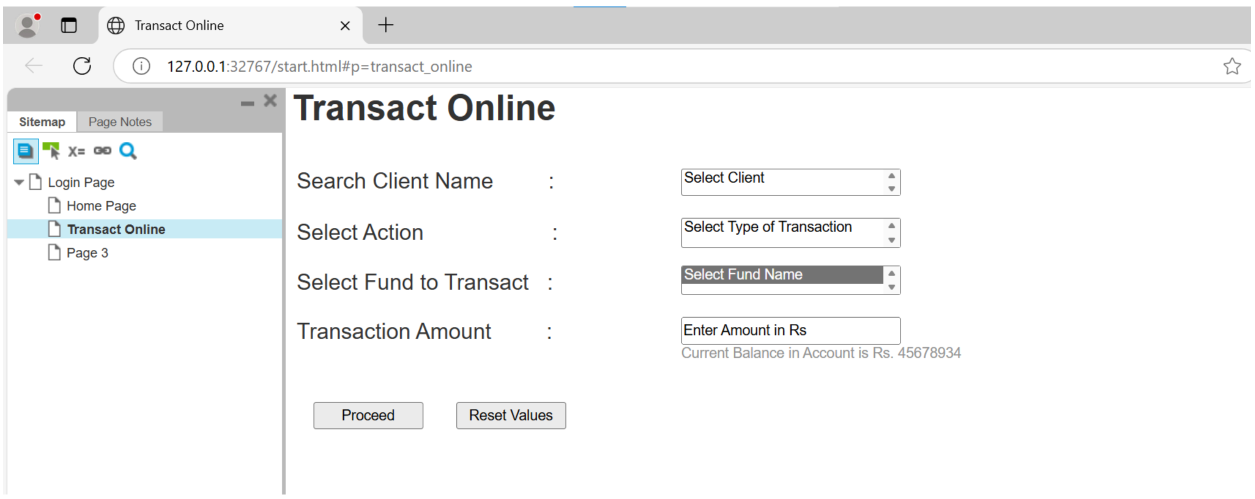
Actors	Primary Actors: Client, Relationship Manager (RM) Secondary Actors: Reporting System, Taxation System, Database
Basic Flow	1. The user logs into the system. 2. The user navigates to the "Capital Gain Reports" section. 3. The user selects the report type (e.g., Short-term gains, Long-term gains). 4. The system prompts the user to specify a time period for the report (e.g., last tax year). 5. The user selects the time period and submits the request. 6. The system generates the capital gain report. 7. The system displays the report preview on the screen. 8. The user can download or email the report.
Alternate Flow	1. If the user provides an invalid time period, the system will ask the user to select a valid period. 2. If there are no capital gains during the selected period, the system will notify the user with a message such as "No capital gains to report for the selected period."
Exceptional Flows	1. If the system encounters an error while generating the capital gain report, it will display a failure message like "Error in report generation, please try again later." 2. If the user session expires, they will be logged out, and they will need to log in again to generate the report.
Pre-Conditions	1. The user must be logged in. 2. The user must have made at least one transaction that generated capital gains.
Post-Conditions	1. The capital gain report is generated and available to the user. 2. The user can download or email the report.
Assumptions	1. The system has access to up-to-date transaction data. 2. The user has appropriate permissions to generate capital gain reports.
Constraints	1. The system can only generate reports based on available capital gain data. 2. The report generation may take time if the dataset is large.
Dependencies	1. The taxation system must be integrated with the reporting system to ensure accurate capital gain calculations.
Inputs and Outputs	Inputs: Time period selection, report type (Short-term or Long-term gains). Outputs: Generated capital gain report (PDF/Excel).
Business Rules	1. The system must distinguish between short-term and long-term gains based on the time of investment and the holding period. 2. Reports should include all relevant transactions that contribute to capital gains.
Miscellaneous Information	1. Capital gain reports must adhere to taxation standards and be downloadable in multiple formats (PDF, Excel). 2. Reports should ensure data confidentiality and meet regulatory standards.

Document 7- Screens and pages









Transact Online

127.0.0.1:32767/start.html#p=transact_online

Sitemap | Page Notes

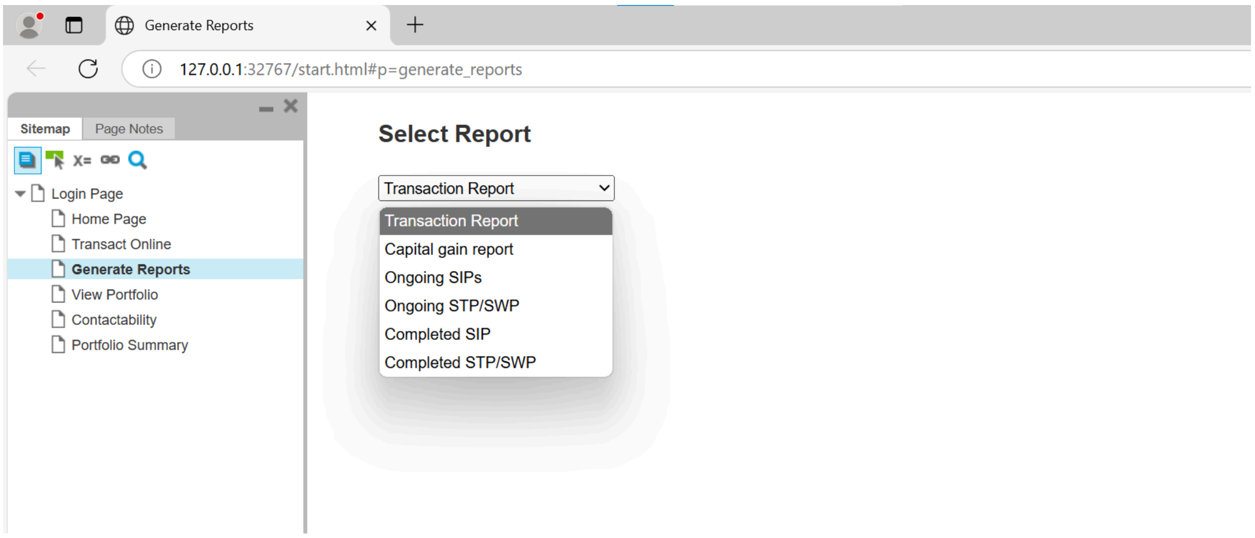
- Login Page
  - Home Page
  - Transact Online
  - Page 3

Transact Online

Search Client Name	:	Select Client
Select Action	:	Select Type of Transaction
Select Fund to Transact	:	Select Fund Name
Transaction Amount	:	Enter Amount in Rs

Current Balance in Account is Rs. 45678934

Proceed | Reset Values



Generate Reports

127.0.0.1:32767/start.html#p=generate_reports

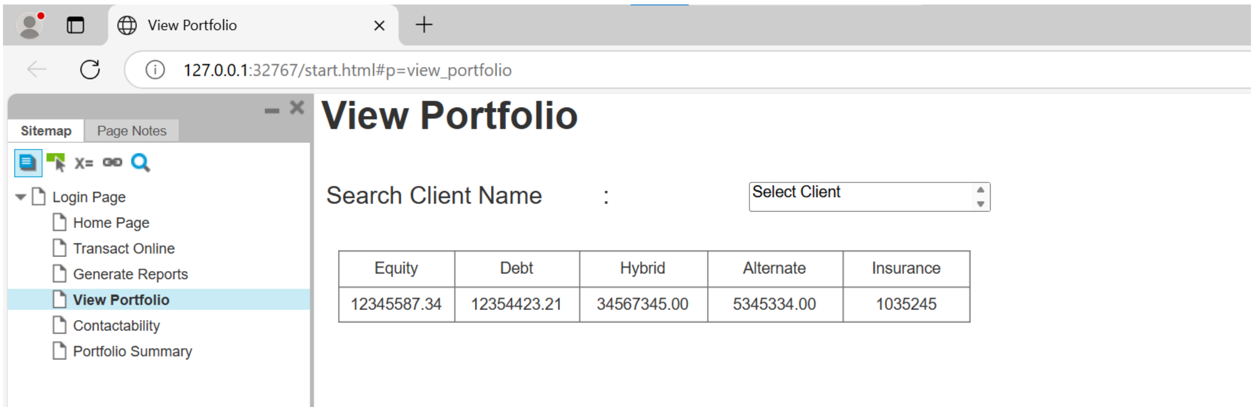
Sitemap | Page Notes

- Login Page
  - Home Page
  - Transact Online
  - Generate Reports
  - View Portfolio
  - Contactability
  - Portfolio Summary

Select Report

Transaction Report

- Transaction Report
- Capital gain report
- Ongoing SIPs
- Ongoing STP/SWP
- Completed SIP
- Completed STP/SWP



View Portfolio

127.0.0.1:32767/start.html#p=view_portfolio

Sitemap | Page Notes

- Login Page
  - Home Page
  - Transact Online
  - Generate Reports
  - View Portfolio
  - Contactability
  - Portfolio Summary

View Portfolio

Search Client Name : Select Client

Equity	Debt	Hybrid	Alternate	Insurance
12345587.34	12354423.21	34567345.00	5345334.00	1035245

Document 8- Tools-Visio and Axure
Write a paragraph on your experience using Visio and Axure for the project.

Visio:

In this project, **Visio** helps me create **visual diagrams** that outline how the system works. It's great for showing **high-level designs** and the **overall flow** of processes.

- **Use Case Diagrams**: For example, I'd use Visio to map out **who does what** in the system, like how **clients** can log in, check their portfolio, or make an investment. It shows all the actions each user (like a **client** or **Relationship Manager**) can take.
- **Workflow Diagrams**: I can also create **step-by-step diagrams** of processes, like how a **client** would go about buying mutual funds or redeeming them. These diagrams help everyone understand how the system should behave at each stage.

Axure:

Axure is used for building **interactive prototypes** that look and feel like the actual system. It lets stakeholders click around and get a sense of how things will work.

- **Wireframing**: After gathering all the system's requirements (like showing the **portfolio summary** or **searching for funds**), I use Axure to create **wireframes**, which are like the "blueprints" of the system. For example, I might create a simple prototype of a **client dashboard** where users can view their investments.
- **Prototypes**: I can also create working models of important features, like how a **client** would **select a fund** and **enter an investment amount**. With Axure, I can simulate these actions, making the prototype feel interactive and realistic.
- **Conditional Logic**: One of Axure's cool features is adding **smart actions**. For example, if a user tries to redeem more funds than they have, the system will show a warning message. This makes the prototype feel even more like a real system.

Combining Both Tools:

In the project, **Visio** and **Axure** work together at different stages:

- **Visio** is used early on to **map out the system's structure** and **show how things flow**. It's great for high-level planning, like defining how users interact with the system. For example, Visio would help map out the steps for things like **fund redemption**.
- Once we have the basic design down, **Axure** comes in to create **interactive prototypes**. These let stakeholders click through things like the **dashboard** or the process of making an investment, which helps us get feedback on the user experience.

In short, **Visio** helps us plan the system and show how it works, while **Axure** helps us build interactive mockups so people can try it out and give feedback before development starts.

Conclusion:

Using **Visio** and **Axure** together helps ensure that the **Integra Portal** is well-planned and easy to use. **Visio** helps us lay out the system's structure, and **Axure** lets stakeholders interact with a working prototype. This combination ensures the system works as expected and provides a great user experience.

Document 9- BA experience
My experience as BA in following phases:

1. Requirement Gathering:

- **MOSCOW Technique**: I used the **MOSCOW** method to prioritize the system's features, ensuring key functionalities like **"User Login"** were prioritized over others like **"Client Meeting Log"**.
- **Client Communication**: As the client wasn't always available, I quickly sourced **point of contacts** from their team to keep the information flowing smoothly.
- **Validation**: I validated requirements using **FURPS** (Functionality, Usability, Reliability, etc.) to ensure the system met all necessary standards.
- **Removing Duplicates**: I identified and removed **duplicate requirements** to keep things efficient.
- **Prototyping**: For clarity, I created **prototypes** to refine requirements, such as the **Client Dashboard**, to align with client needs.

2. Requirement Analysis:

- **UML Diagrams**: I created **UML** diagrams to visually map out system interactions, like how **clients** would interact with features such as **viewing portfolio summaries**.
- **Activity Diagrams**: I used **activity diagrams** to define the process flows for features like **fund redemption** and **transaction approvals**.
- **Team Feedback**: I communicated these diagrams to the team and handled differing opinions, adjusting the designs where necessary.
- **BRS/SRS**: I wrote the **BRS** (Business Requirements Specification) and **SRS** (System Requirements Specification) to outline the functional and technical details of the system.

3. Design:

- **Test Case Preparation**: I worked with the testing team to write **test cases** based on the use cases, ensuring full coverage for both positive and negative scenarios.
- **Client Communication**: I shared the **design documents** with the client to confirm the solution met their expectations.
- **RTM Updates**: I made sure the **Requirements Traceability Matrix (RTM)** was regularly updated to track requirements against the design.

4. Development:

- **JAD Sessions**: I organized **JAD** sessions to clarify any requirements and kept the development process on track.
- **Conflict Resolution**: If there was disagreement during sessions, I facilitated one-on-one discussions to address concerns and keep the team motivated.

- **Collaboration**: I worked closely with both the **client** and **technical team**, ensuring alignment on the project, especially when key members couldn't attend meetings.

5. Testing:

- **Test Case Execution**: I assisted the testing team by preparing and executing **test cases**, focusing on key functionalities like **investment transactions**.
- **RTM Updates**: I made sure the **RTM** was always updated with the testing progress.
- **Client UAT**: After ensuring everything was tested, I coordinated **User Acceptance Testing (UAT)**, getting final client sign-off.

6. Deployment:

- **Final RTM and Documentation**: I shared the completed **RTM** with the client and finalized the **project closure** document.
- **Training & Support**: I organized **training sessions** for the client and ensured they received all the necessary **end-user manuals** to effectively use the system.

Throughout the project, I acted as the bridge between the client and the development team, ensuring smooth communication and alignment across all phases, from gathering requirements to deployment.