**Online Banking Project-Dashboard Module**

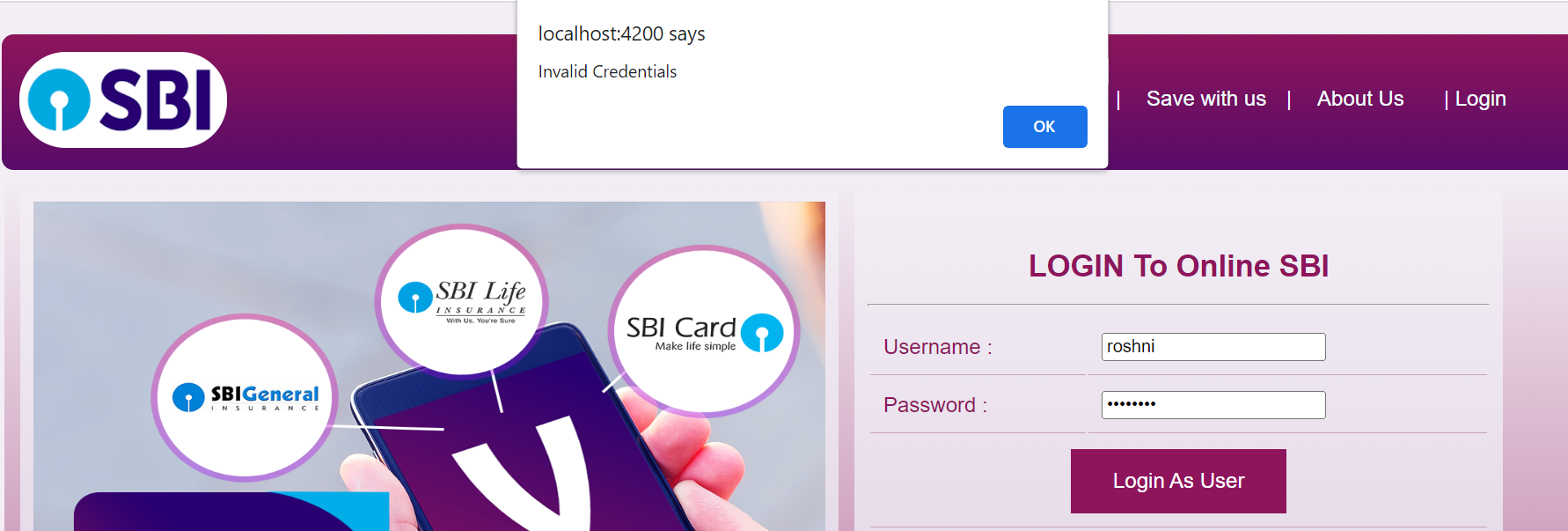
Team of Poornima,Nimya,Jayendra,Rajendra

**Pages:**

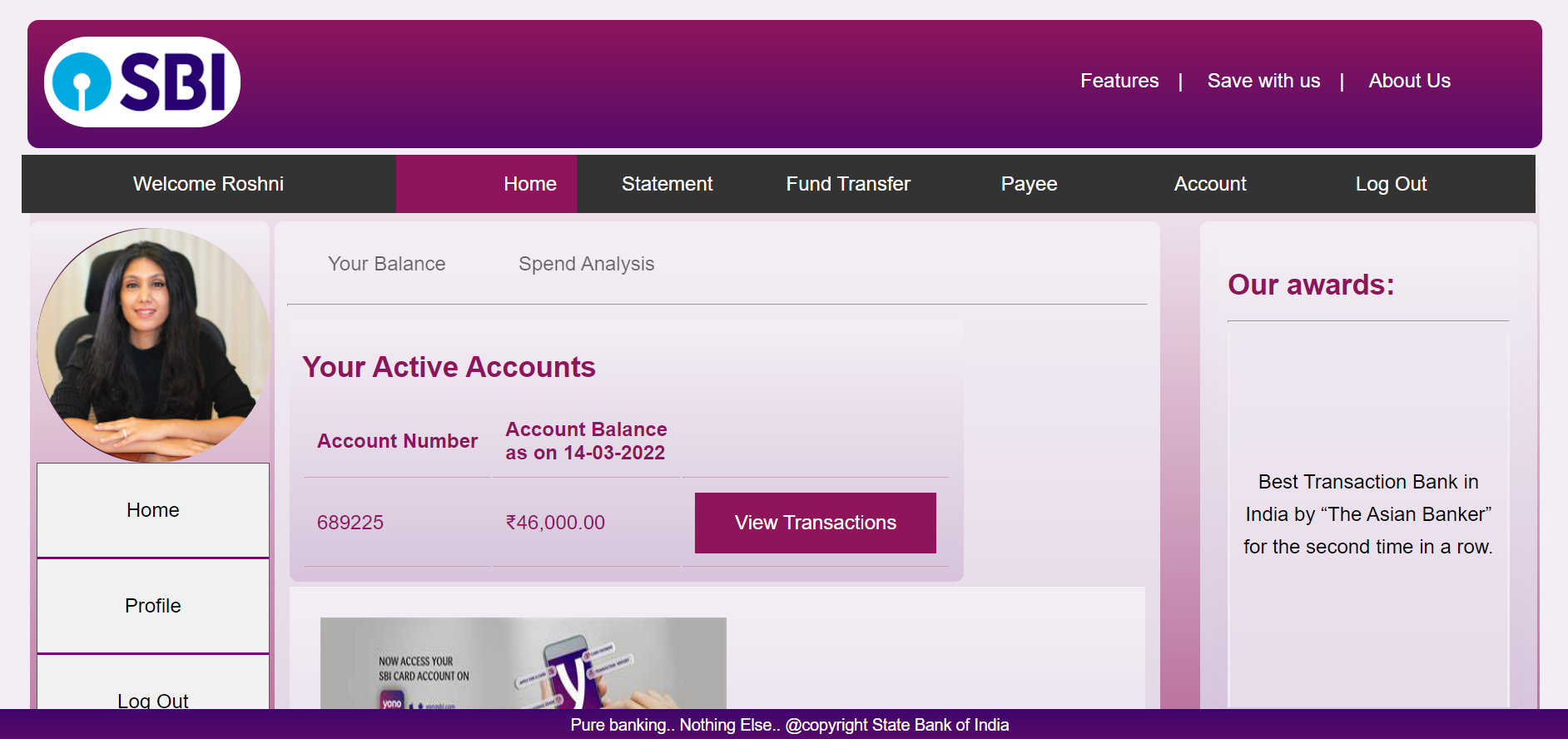
On the main page, we have the login page wherein the user keys in the login id and password, the credentials entered are checked in the login table.

On a match with entries in login table the user is allowed login, else error message of invalid credentials is displayed.

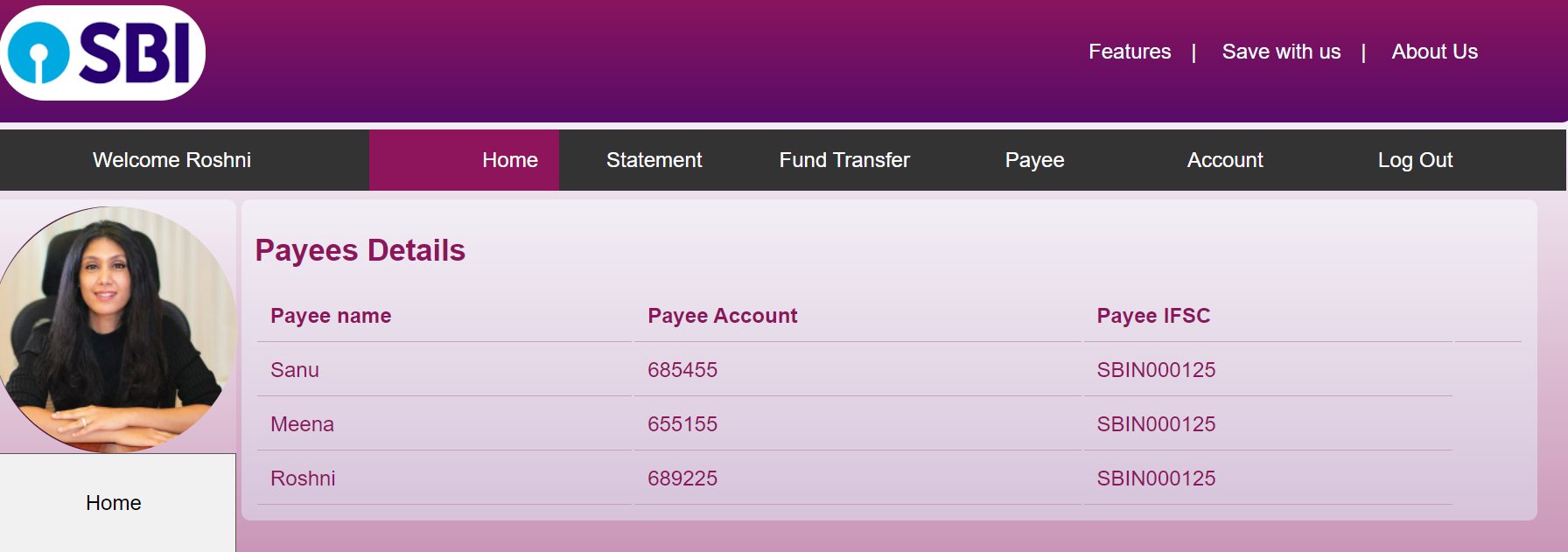
**Invalid Credential Scenario:**



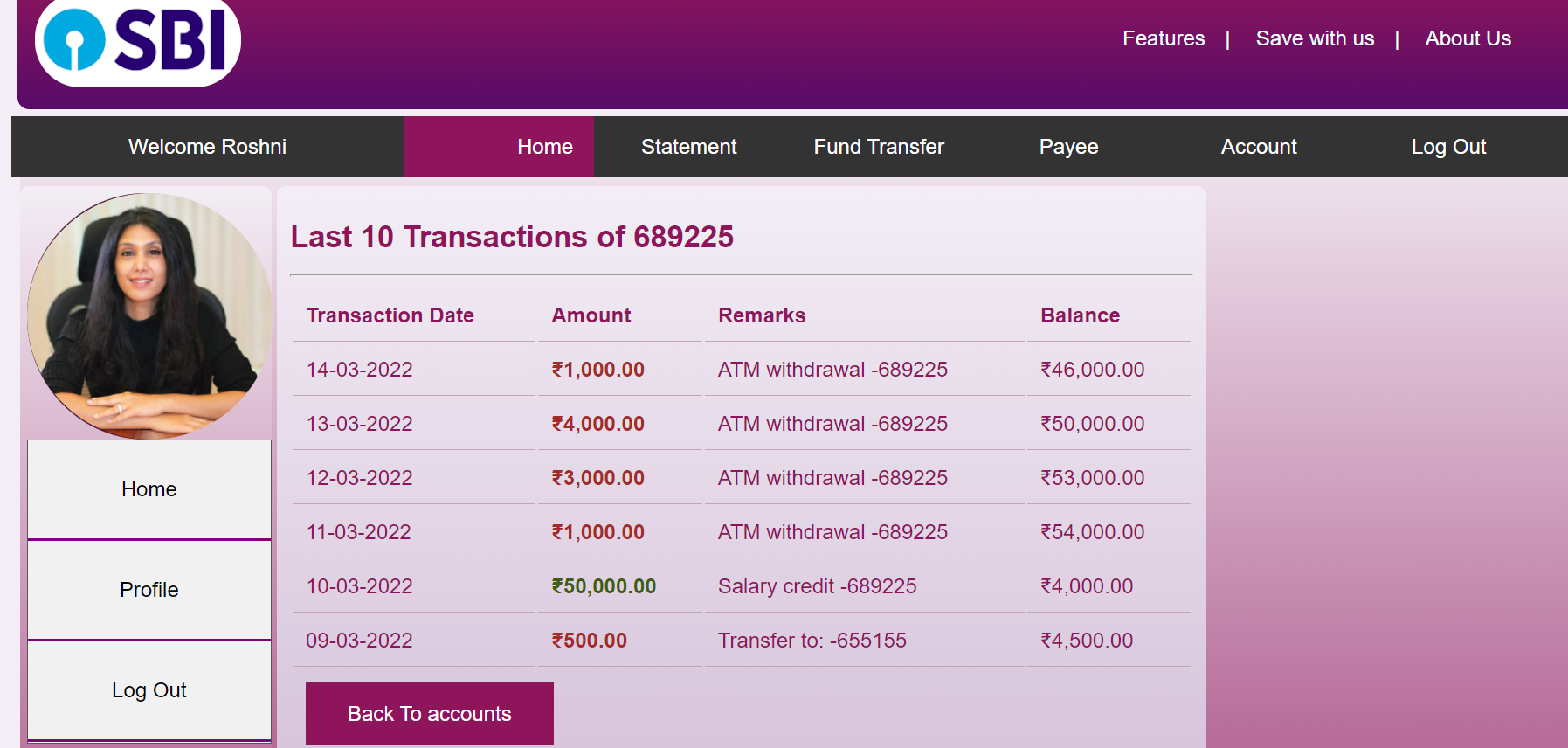
**Home Page-Valid Credential:-Post Login**

****

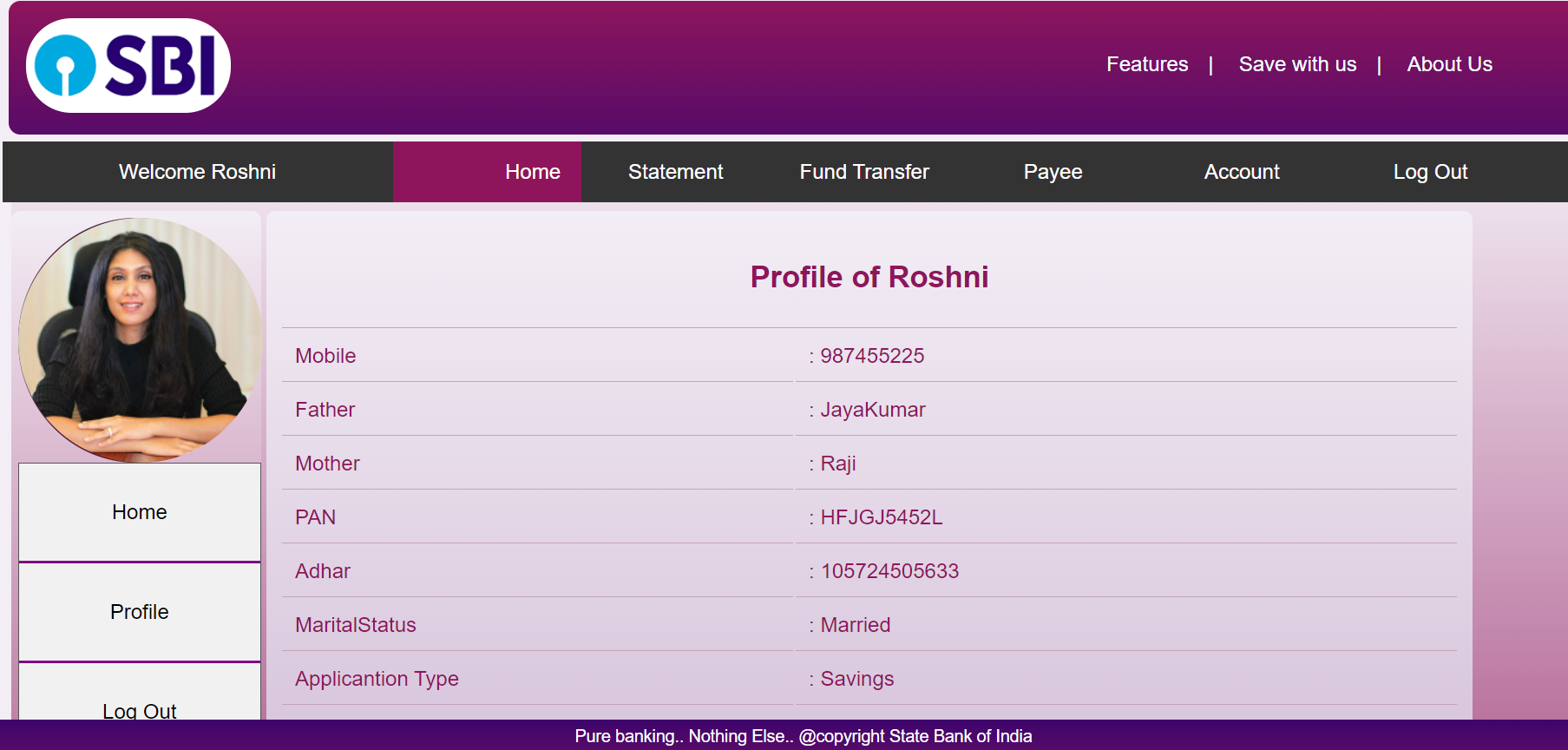
**Payee Tab-** Retrieves data from Payee Table and displays for the particular Account

****

**Accounts Tab-**Retrieves the last 10 transactions from Transaction table based on the Account Number

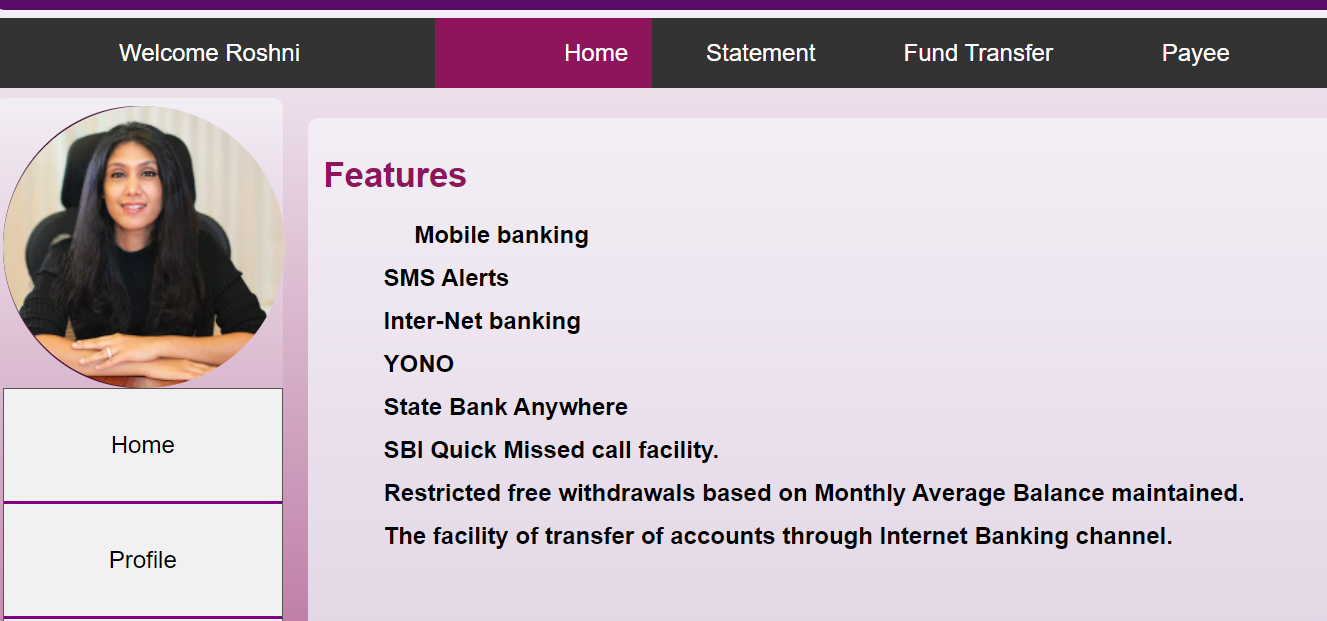
****

**Profile Tab-** Retrieves the profile of the logged in user from the Applicant Table

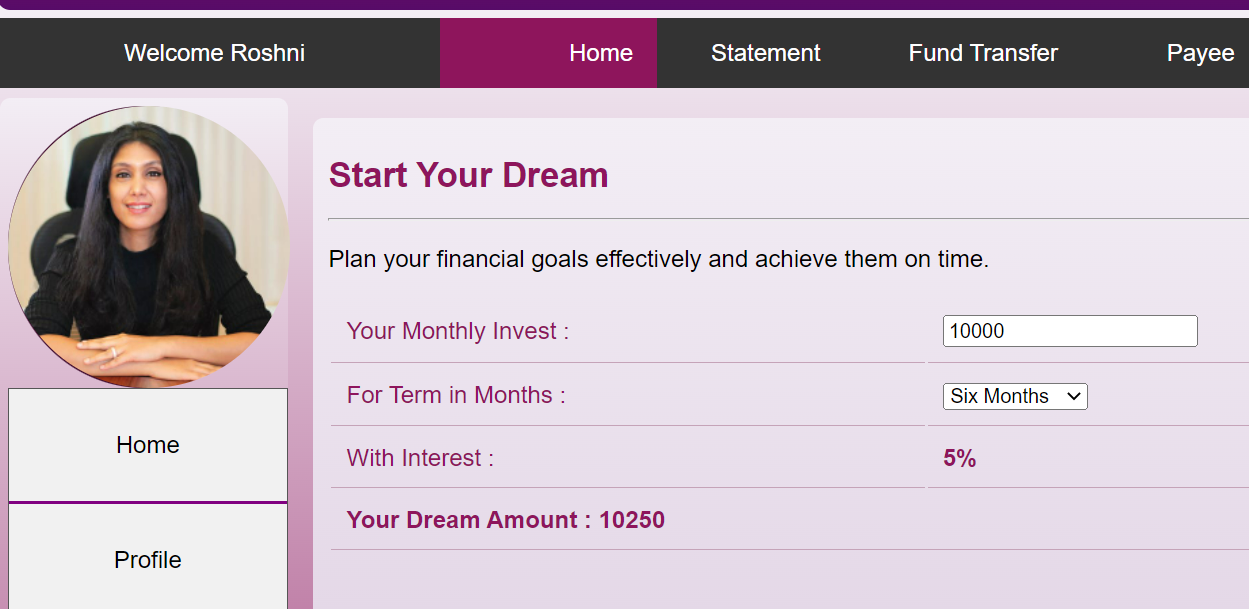
****

**Other Pages:**

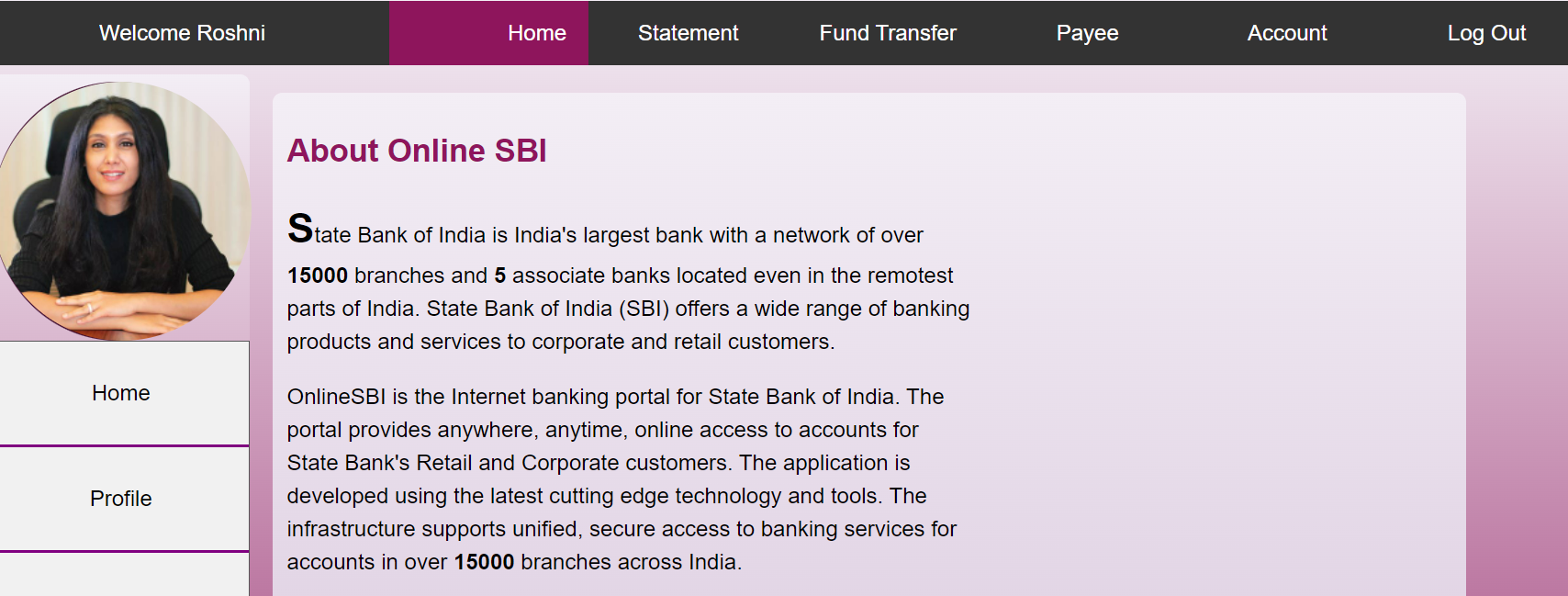
**Features:**

****

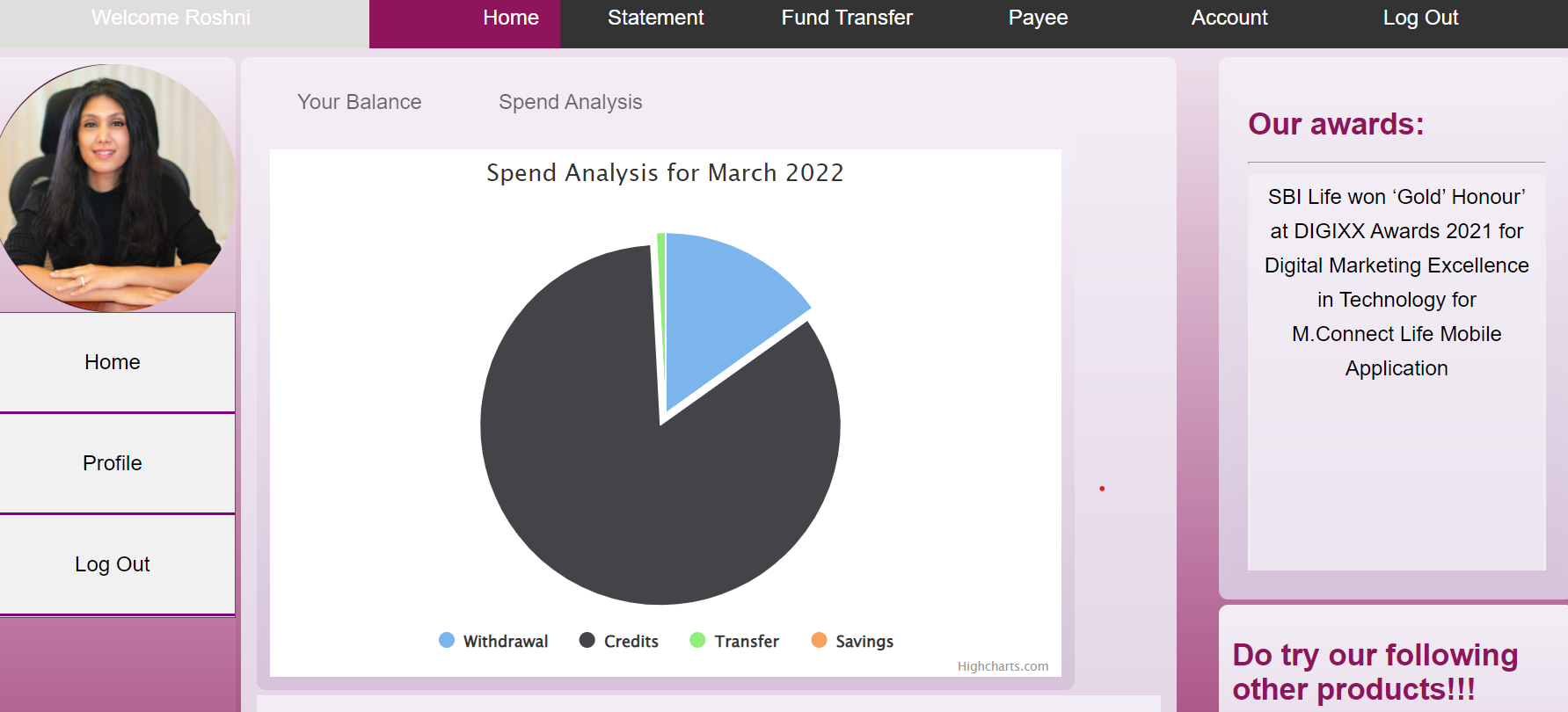
**RD Calculator**

****

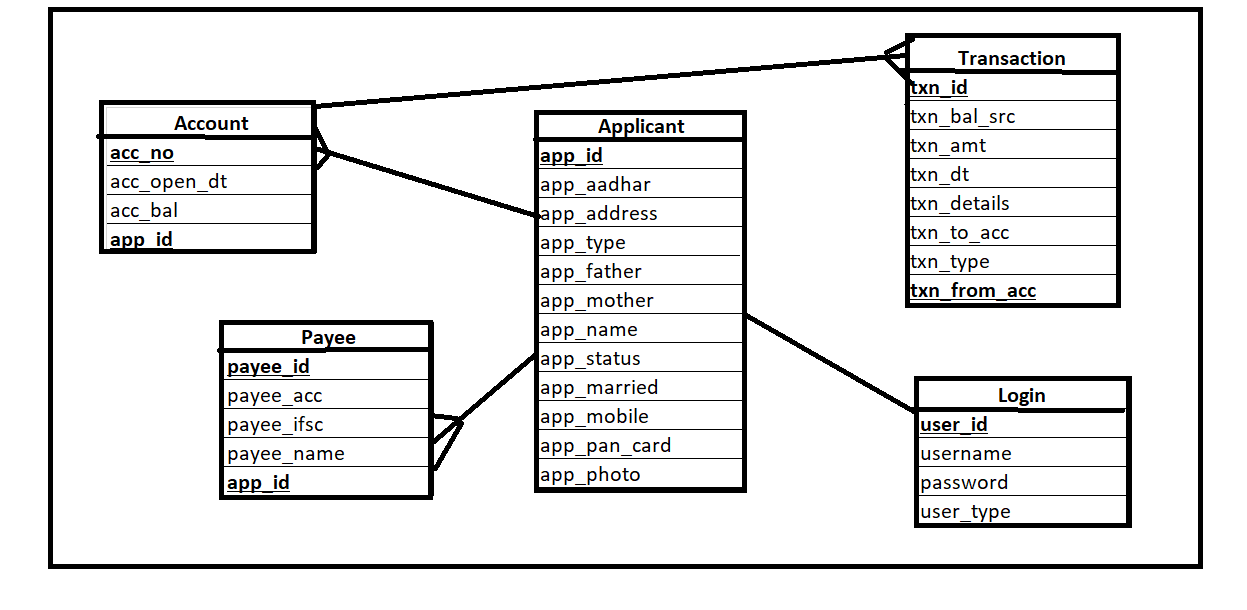
**About Us**

****

**Spend Analysis:**

****

**LAYER1:TABLES USED AND THEIR RELATION:**



**LAYER2: POJOS/ENTITIES**

**Account.java-**This class has setter getter methods for account entity,many to one relation with Applicant object,one to many relation with transaction entity(fetches transactions in an array).

**Applicant.java-**This class has setter getter methods for applicant entity,shares one to many relation with Account and Payee.

**Login.java-**This class has setter getter methods for user related details,userid is the applicantid,used for validating the user during login,one to one relation with Applicant.

**Payee.java-**This class has setter getter methods for payee related details and shares a many to one relation with Applicant.

**SpendAnalysis.java-**This has setter getter methods for storing values related to spend analysis graph to be displayed like withdrawal,deposit,savings totals.

**Transaction.java-** This class has setter getter methods for transaction related details and shares many to one relation with Account entity.

**LAYER3: REPOSITORY-Interfaces and Implementations are used.**

**BaseRepository-**This is the template for implementation of the CRUD operations,other repositories can extend from this repository.

**Account/Applicant/Login/Payee/Transaction Repositories**-They perform CRUD Operations

**Specifically the following methods are of importance:**

1.AccountRepo has a method that finds account based on applicant id.

2.LoginRepo has a method that finds the Applicant based on user id,here the authentication is performed that logged in user id and applicant id is same.

3.PayeeRepo has a method that gets all the Payees for the particular applicant

4.TransactionRepo has a method that gets transaction based on account number

5.TranscationRepo has also a methos that updates the spend analysis graph based on transaction types made

**LAYER4: SERVICE-Interfaces and Implementations are used.**

**Account/Applicant/Payee/Transaction Service**-Call to respective Repositories is done.

**LAYER5: CONTROLLER**

**Account Controller:**Retrieval of accounts based on applicantID

**Payee Controller:**Retrieval of Payees based on applicantID

**ApplicantController:** Authentication of Applicant,populating the spend analysis tab for the particular month

**Transaction Controller:**Retrieval of Transactions based on the particular ApplicantID

**TEST CASES:**

Test cases for the working of the specific methods are in place:

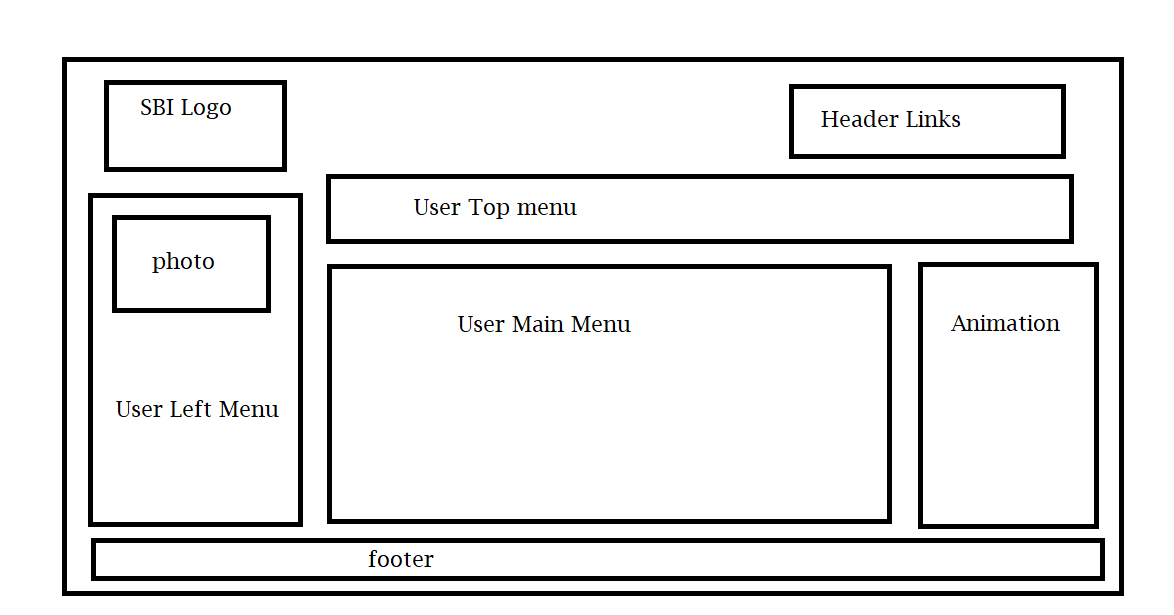
Retrieval of Payees

Retrieval of Transactions

**LAYER6:ANGULAR-UI and Related Routing**

**Specific Components of Importance:**

Home Page:



**Services:**

**Account, Applicant , Payee, SpendAnalysis, Transaction-**retrieves data based on logged in applicant.

**Authenticate-**stores session related details and authenticates the User.