DOKUZ EYLÜL UNIVERSITY
ENGINEERING FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

CME 3201

Database Management Systems

YEF BANK
(Internet Banking)

By

2019510014 Yiğit Can Akçay

2019510055 Erdoğan Kayalı

2019510068 Fatih Semirgin

A) PROBLEM DESCRIPTION

Our project is a small bank that is looking to improve their online presence by developing a new website. The website should allow customers to access their account information, make payments and transfers, and find information about the bank's products and services. Additionally, the website should provide a secure login for customers to access their accounts and perform financial transactions.

Some specific requirements for the website include:

- A responsive design that works well on desktop and mobile devices
- A clean and modern aesthetic that reflects the bank's brand
- A user-friendly interface that makes it easy for customers to navigate and find the information they need
- A secure login system that protects customer information and financial transactions
- The ability for customers to view their account balances and transaction history
- The ability for customers to make payments and transfers between accounts
- Integration with the bank's existing systems and data
- The website should be built using modern web development technologies and practices, including HTML, CSS, and JavaScript. It should be scalable and maintainable, with the ability to add new features and functionality in the future.

B) OPERATIONS LIST

- Register
- Log-in
- Reset Password
- Admin Control Panel
- Foreing Currency View
- Currency Calculator
- Nearest Branch View
- Account View
- Account Operations
    - *Transfer*
    - *Transfer Other Account*
    - *New Account*
- Card View
- Profile and Settings
    - *Profile View*
    - *Changing Password*
    - *Login Records*
    - *Changing Informations*
    - *Transaction History*

C) SYSTEM ARCHITECTURE, CONSTRAINTS AND CHALLENGES

Monolithic architecture is used our project. A monolithic architecture is a single, unified application that handles all aspects of the system. This can be simple to develop and deploy, but may not scale well as the user base grows and the system becomes more complex.

They are constraints and challenges for our project:

- **Time constraints:** The project may have tight deadlines that need to be met, which can make it difficult to adequately test and debug the website.
- **Budget constraints:** The project may have limited resources, which can impact the scope and functionality of the website.
- **Technical constraints:** The website may need to integrate with existing systems or work with specific technologies, which can introduce additional complexity.
- **User experience constraints:** The website needs to be easy to use and navigate, which can be challenging to achieve if there are conflicting design or functionality requirements.
- **Security constraints:** The website needs to be secure and protect customer information, which requires careful planning and implementation.
- **Compatibility constraints:** The website needs to work across a range of devices and browsers, which can be difficult to test and ensure consistency.
- **Scalability constraints:** The website may need to handle a large volume of traffic or data, which requires careful planning and performance optimization.
- **Maintenance constraints:** The website needs to be maintainable and easy to update, which requires a solid architecture and documentation.

D) USED TECHNOLOGY, TOOLS AND PROGRAMMING LANGUAGES

- **HTML** is a markup language used to structure and format content on the web. It is used to define the structure of web pages, including headings, paragraphs, and links.
- **CSS** is a stylesheet language used to describe the look and formatting of a document written in HTML. It is used to control the layout, color, and font of web pages.
- **JavaScript** is a programming language that is commonly used in web development to add interactivity to web pages. It can be used to manipulate the HTML and CSS of a web page, and to create dynamic and responsive user interfaces.
- **PHP** is a server-side programming language that is commonly used to build dynamic and interactive websites. It is an open-source language that is free to use and modify, and it is easy to learn. PHP code is executed on the server and generates HTML, which is then sent to the user's web browser to be displayed.
- **Databases:** Websites often need to store and retrieve data, and databases are used to manage this data. There are many different database technologies that can be used, including relational databases. In this project, we used MySql.
- **Version Control Systems:** Version control systems are used to track changes to code and collaborate with other developers. They allow developers to easily revert changes, branch code for experimentation, and merge code from different developers.
  In this project, we used Git. Also used Github.
- **Integrated development environments (IDEs):** IDEs are software applications that provide a development environment for coding, debugging, and testing. We used VsCode and PhpStorm.
- **Web servers:** Web servers are used to host websites and serve content to users. We used Apache.
- **Frameworks:** Frameworks are pre-built libraries of code that can be used to streamline the development process. We used Codeigniter3 which is a PHP framework.
- **APIs (Application Programming Interfaces):** APIs allow different systems and services to communicate with each other and exchange data. They can be used to integrate a website with other systems or platforms. We used exchangerate API.

E) SQL STATEMENTS

- *Basic Operations Examples*
    - SELECT * FROM customer ORDER BY Customer_ID ASC
        - *This statement is used to retrieve all rows from the customer table, sorted by the Customer_ID column in ascending order.*
    - SELECT * FROM customer WHERE Identity_No =? AND Password=?
        - *This statement is used to retrieve data from the customer table based on the values of the Identity_No and Password columns. It is likely being used to authenticate a user by verifying their login credentials.*
    - SELECT * FROM account WHERE account.IBAN = {}
        - *The statement retrieves all rows from the account table where the IBAN column has the specified value. If the value is not provided or is not found in the IBAN column, no rows will be returned.*
- *Complex Queries Examples*
    - SELECT * FROM account
      LEFT JOIN branch ON account.Branch_ID = branch.Branch_ID
      WHERE account.Identity_No ={}
        - *The statement uses a LEFT JOIN to combine the rows from the two tables. The LEFT JOIN returns all rows from the account table, and any matching rows from the branch table. If there are no matching rows in the branch table, the LEFT JOIN will still return a row in the result set with NULL values for the columns from the branch table. Also statement includes a WHERE clause that filters the results based on the Identity_No column of the account table. The value of this column is specified by the {} placeholder, which represents a value that is not provided in the statement. If a value is provided, the statement will only return rows where the Identity_No column has that value. If no value is provided, the statement will return all rows from the account table.*

- SELECT * FROM customer
  JOIN account ON customer.Identity_NO = account.Identity_NO
  AND account.Account_ID != {$account_id}
  JOIN branch ON branch.Branch_ID = account.Branch_ID
  WHERE customer.Customer_ID = {$id}
    - *This statement is used to retrieve data from three tables in the database, filtered by specific conditions and joined on common columns.*
- SELECT * FROM customer
  JOIN account ON customer.Identity_NO=account.Identity_NO
  JOIN branch ON branch.Branch_ID = account.Branch_ID
  WHERE csutomer.Identity_No = {$id}
    - *This statement is used to retrieve data from three tables in the database, filtered by a specific value and joined on common columns.*
- SELECT * FROM customer
  JOIN account ON customer.Identity_NO = account.Identity_NO
  JOIN branch ON branch.Branch_ID = account.Branch_ID
  WHERE customer.Identity_No = {$id}
    - *This statement is used to retrieve data from three tables in the database, based on relationships between the tables and a specified filter condition.*

- *VIEW*
  - CREATE VIEW branches AS
    SELECT branch.Branch_ID, branch.City, branch.District, branch.Address, branch.Name FROM branch;

- *STORED PROCEDURES*
  - **GetAllCustomers**
    - BEGIN

      SELECT * FROM customer;

      END

  - **GetBranches**
    - BEGIN

      SELECT * FROM branches;

      END
  - **GetCustomerbyName(in name varchar)**
    - BEGIN

      SELECT *

      FROM customer

      WHERE First_Name = name;

      END
- *TRIGGERS*
  - **accountLog**
    - if (NEW.Balance > OLD.Balance)

      then

      insert into account_logs(Date, Last_Name, Event, Old_Balance, New_Balance, Account_ID, Sender)

      select t.Date, c.Last_Name, 'TAKING', OLD.Balance, NEW.Balance, a.Account_ID, t.Sender

      from customer c, account a, transaction t

      where a.IBAN=t.IBAN_recipient

      and a.Identity_No=c.Identity_No

      and a.Account_ID=t.Recipient

      order by t.Date DESC

      limit 1;

      elseif (NEW.Balance < OLD.Balance)

      then

insert into account_logs(Date, Last_Name, Event, Old_Balance, New_Balance, Account_ID, Sender)
select t.Date, c.Last_Name, 'SENDING', OLD.Balance, NEW.Balance, a.Account_ID, null
from customer c, account a, transaction t
where a.IBAN=t.IBAN_sender
and a.Identity_No=c.Identity_No
and a.Account_ID=t.Sender
order by t.Date DESC
limit 1;
end if

- **customerUpdateLog**
  - BEGIN
    if (OLD.Mail != NEW.Mail)
    then
    insert into customer_changes_log(Date, Identity_No, Event, Old, New)
    values(now(), OLD.Identity_No, 'CHANGE MAIL', OLD.Mail, NEW.Mail);
    elseif (OLD.Phone != NEW.Phone)
    then
    insert into customer_changes_log(Date, Identity_No, Event, Old, New)
    values(now(), OLD.Identity_No, 'CHANGE PHONE', OLD.Phone, NEW.Phone);
    elseif (OLD.Password != NEW.Password)
    then
    insert into customer_changes_log(Date, Identity_No, Event, Old, New)
    values(now(), OLD.Identity_No, 'CHANGE PASSWORD', OLD.Password, NEW.Password);
    end if;
    end

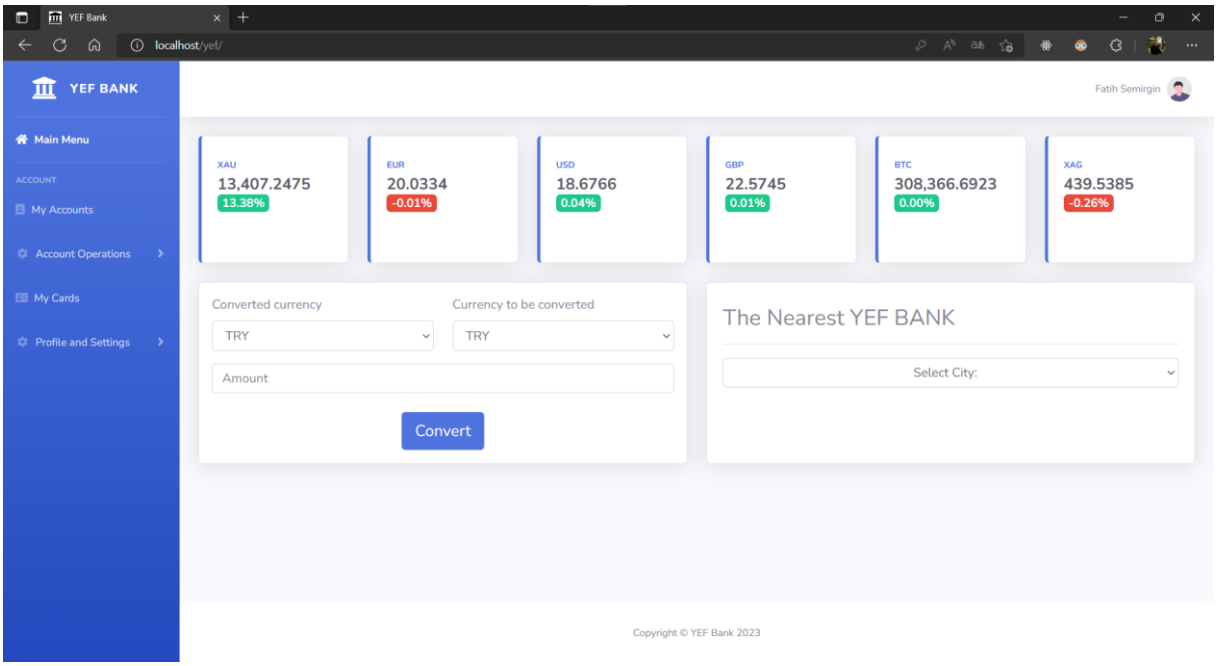
- **deleteAccountLog**
  - BEGIN
    insert into account_changes_logs(Date, Event, IBAN, Account_ID, Identity_No)
    values (NOW(), 'DELETE', OLD.IBAN, OLD.Account_ID, OLD.Identity_No);
    end
- **newAccountLog**
  - BEGIN
    insert into account_changes_logs(Date, Event, IBAN, Account_ID, Identity_No)
    values (NOW(), 'NEW', NEW.IBAN, NEW.Account_ID, NEW.Identity_No);
    end
- **newCustomerLog**
  - BEGIN
    INSERT INTO customer_changes_log(Date,Identity_No,Event,Old,New) VALUES
    (now(),NEW.Identity_No,"NEW CUSTOMER",null,null);
    end

F) ADDITIONAL PROPERTIES OF OUR PROJECT

- Responsive design: Responsive design is a design approach that ensures that a website looks and functions well on a range of devices and screen sizes. This is important because more and more users are accessing the web from mobile devices.
- Performance optimization: Optimizing the performance of a website can improve the user experience and reduce bounce rates. This can involve things like optimizing images and other assets, minimizing the use of external resources, and caching content.

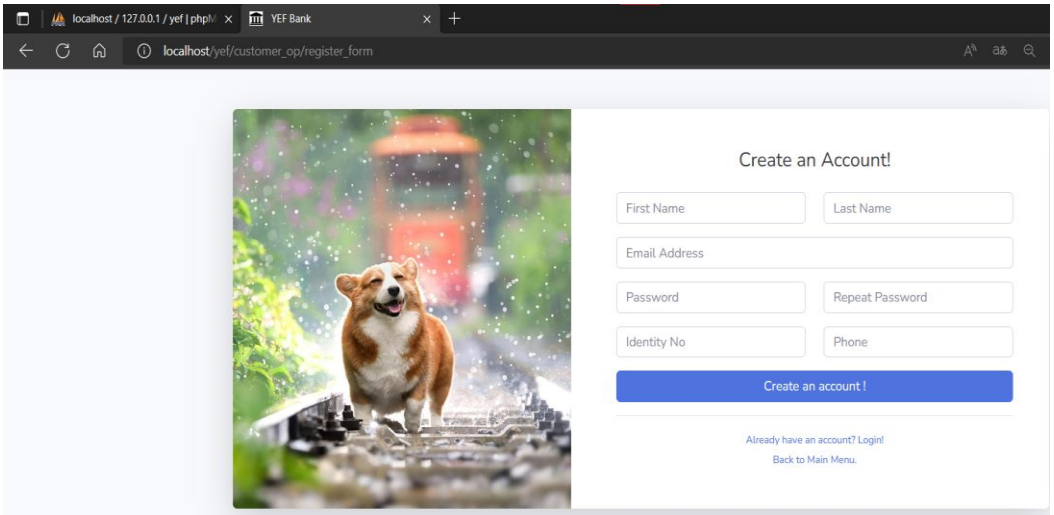
G) SCREENSHOTS OF INTERFACES

- MAIN PAGE



The homepage of a bank website is typically the first page that users see when they visit the website. It is designed to provide an overview of the bank's products and services, as well as to allow users to access their accounts and perform various tasks. Also includes ‘Currency Calculator’, ‘Currency Information’ , ‘Nearest YEF Bank’.

- REGISTER PAGE

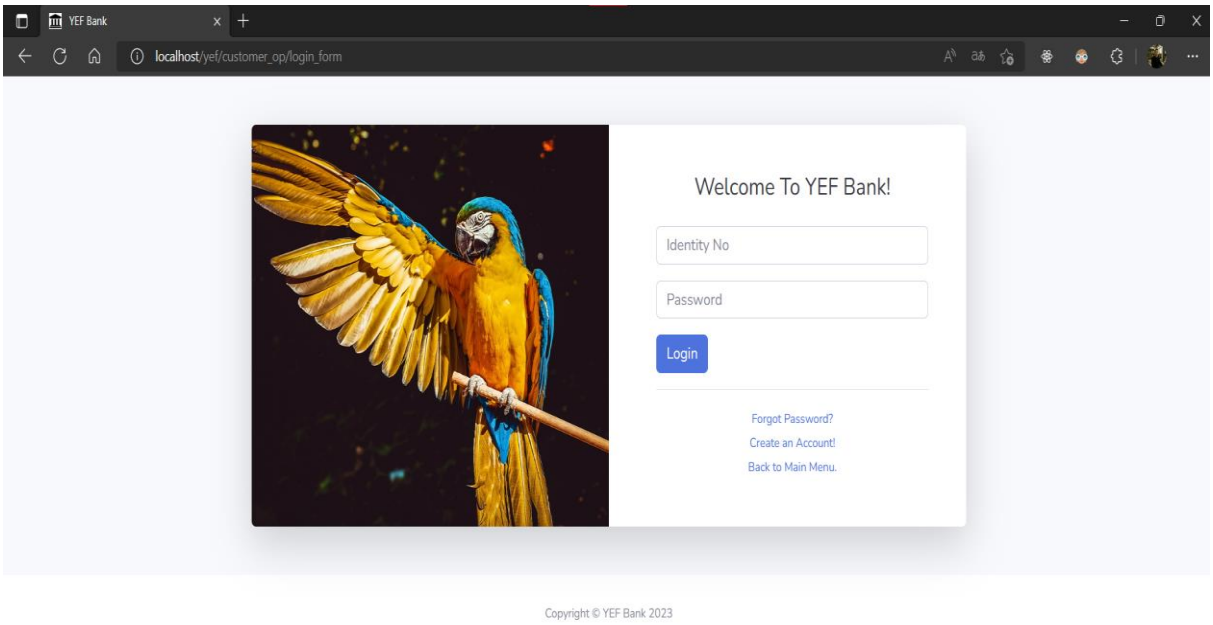


Here are the steps for registering for an internet bank account:

1. Go to the website of the bank you wish to register with.
2. Look for a link or button that says "Sign Up" or "Register" and click on it.
3. Fill out the online form with your personal information, such as your name, address, email, and phone number.
4. Create a username and password for your account. Make sure to choose a unique and secure password that you won't forget.
5. Review the terms and conditions and any other relevant information provided by the bank.
6. Submit the form to complete the registration process.
7. Once your registration has been confirmed, you will be able to access your internet bank account and start using it to manage your finances online.

Again, the specific steps and information required to register for an internet bank account may vary depending on the bank and the country you are in. It is important to carefully read and follow the instructions provided by the bank to ensure a smooth and secure registration process.

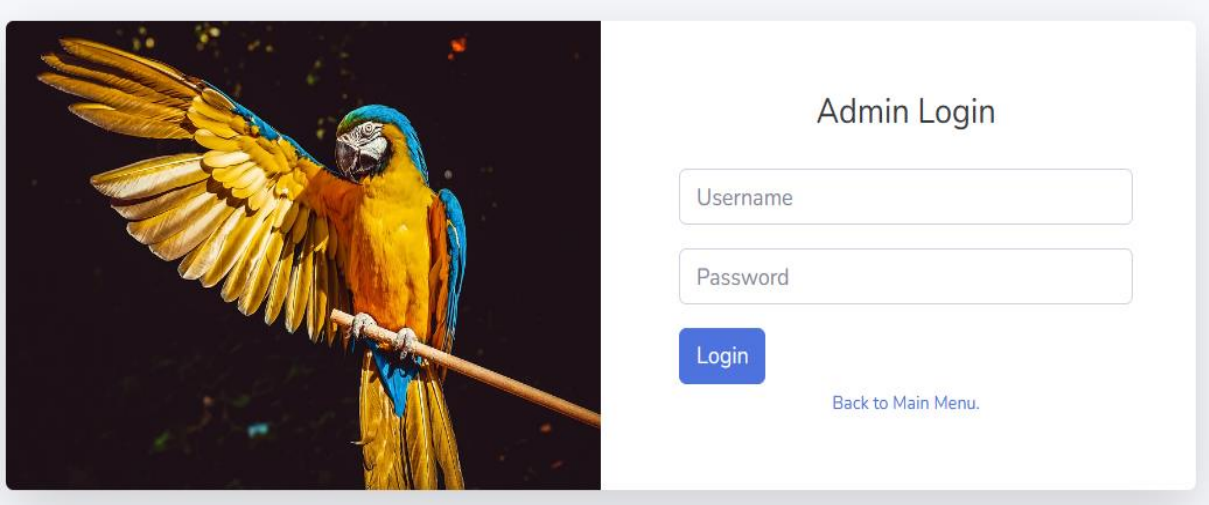
- LOGIN PAGE



When you log in to a bank website, the following process typically occurs:

You enter your identity_no and password into the login form and submit it. The website sends this information to the server, where it is checked against the bank's database of user accounts. If the identity_no and password are correct, the server allows you to log in and grants you access to your account information and other services. If the identity_no and password are incorrect, the server will not allow you to log in and will display an error message. In addition to verifying your login credentials, the server may also check for other security measures, such as verifying that you are accessing the website from a trusted device or location. This helps to protect your account from unauthorized access.

- ADMIN LOGIN PAGE

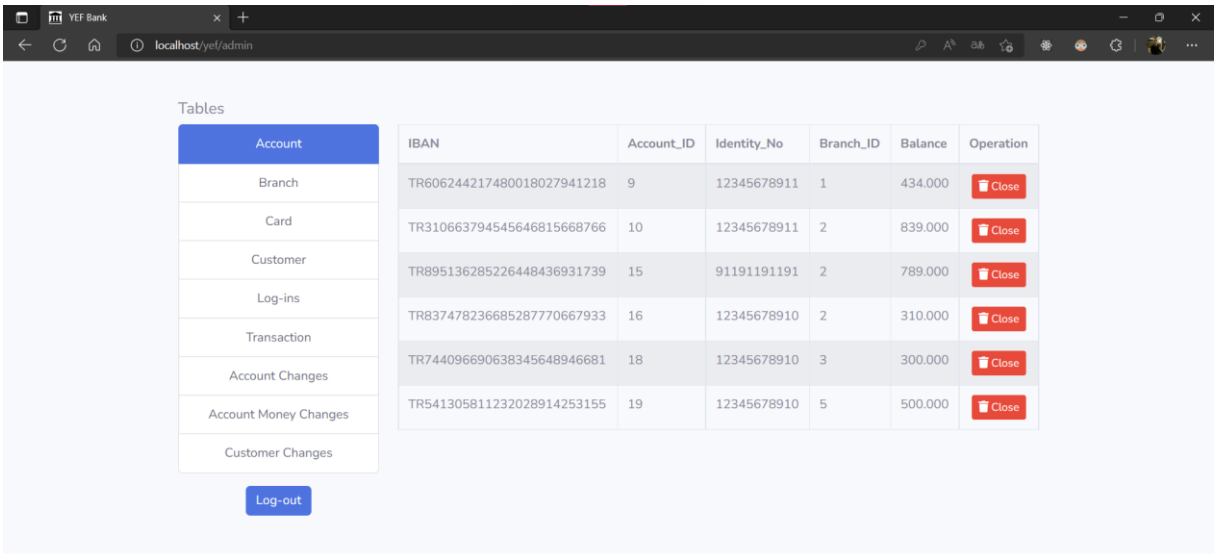


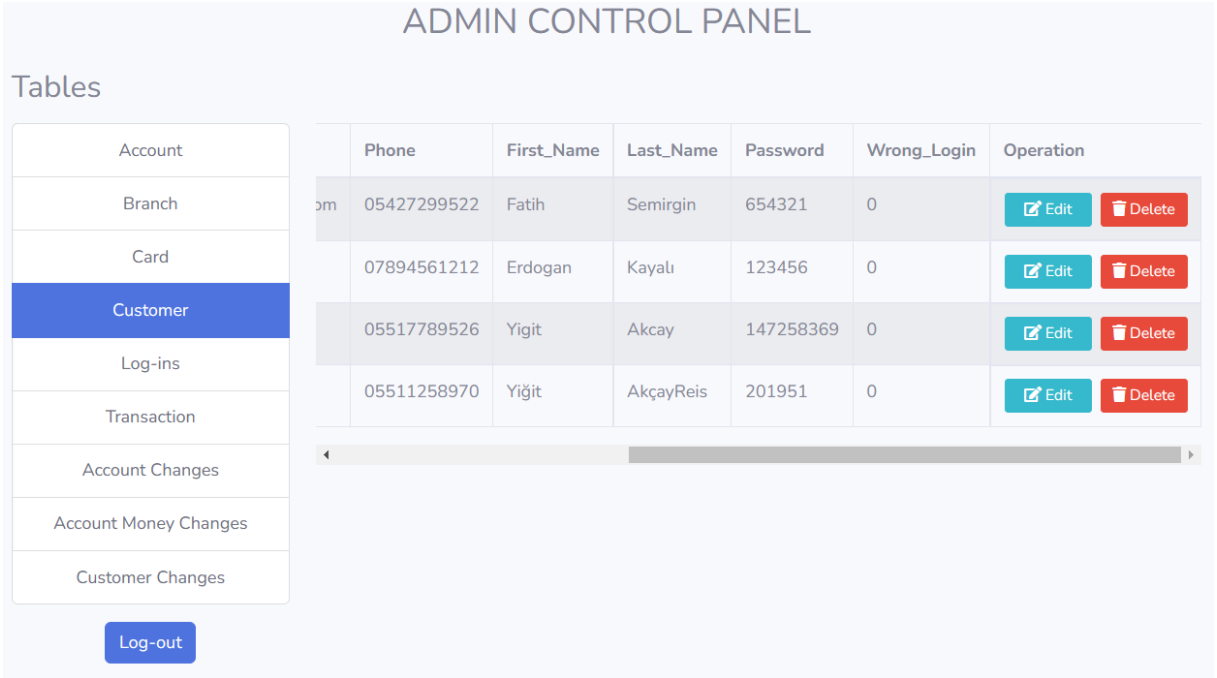
An admin page for an internet bank is typically a web page that is used by bank administrators to manage and monitor the bank's online operations. The specific features and functions available on the admin page will depend on the bank and the specific needs of its business.

Some possible features that might be included on an internet bank admin page include:

- User management: This might include the ability to create, edit, or delete user accounts, as well as view user activity and account information.
- Transaction management: This might include the ability to view, approve, or reject transactions made through the bank's online platform.

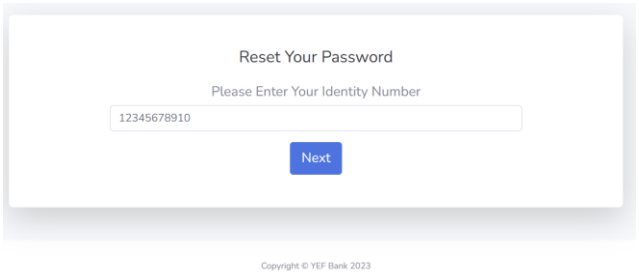
- ADMIN PAGE

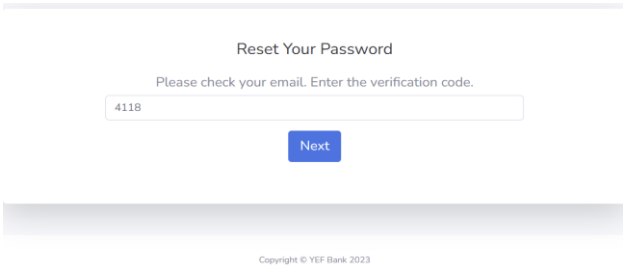


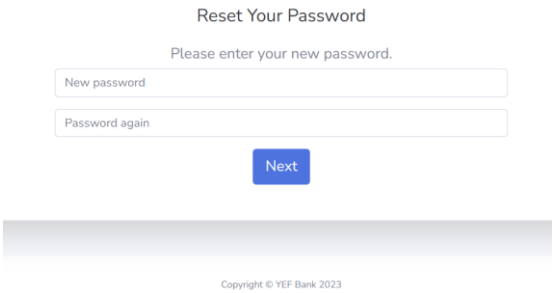


The admin page of a bank website is typically a secure section of the website that is only accessible to authorized administrators. It is used to manage various aspects of the bank's operations, such as managing user accounts, tracking financial transactions.

- FORGOT PASSWORD



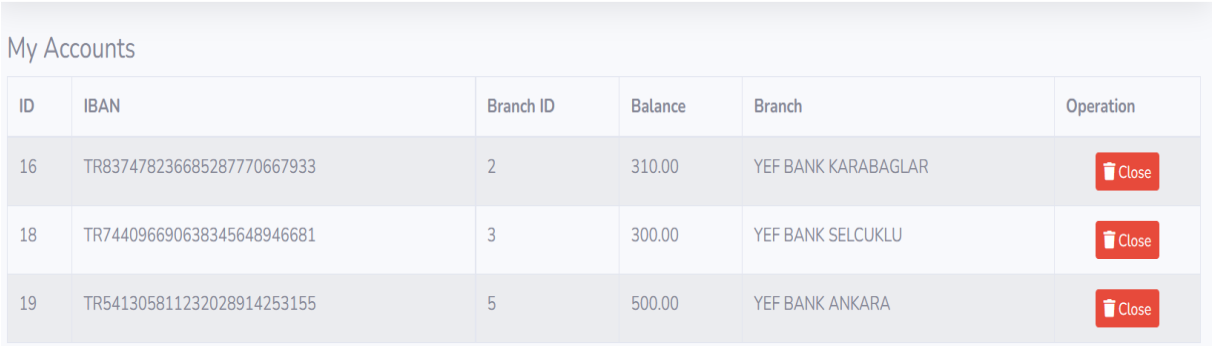




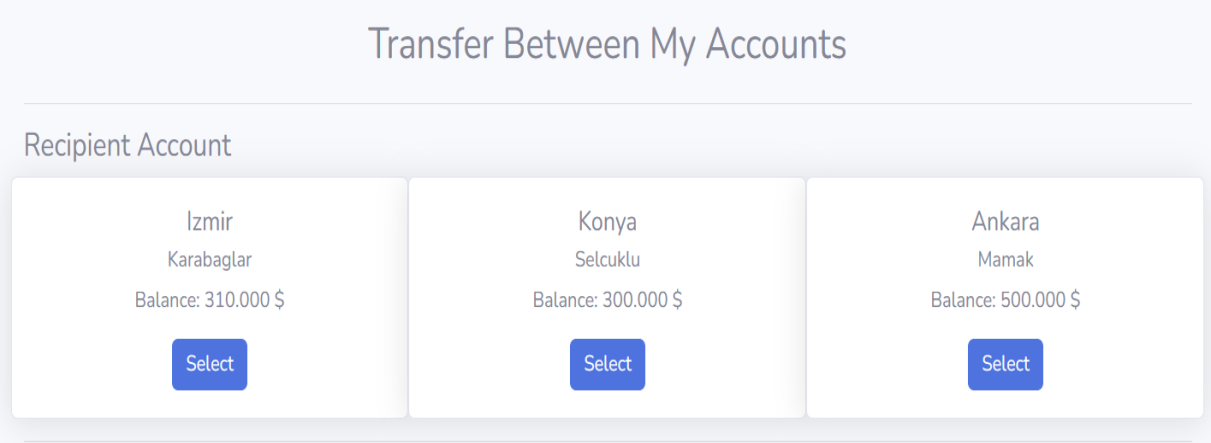
The forgot password page of a bank website is a feature that allows users to reset their password if they have forgotten it or are unable to log in to their account.

Typically, the forgot password page will contain a form where the user can enter their email address. The user may also provide other identifying information. Once the user has submitted the form, the bank will send a password reset code to the user's email address. The user can then follow the instructions to reset their password and regain access to their account.

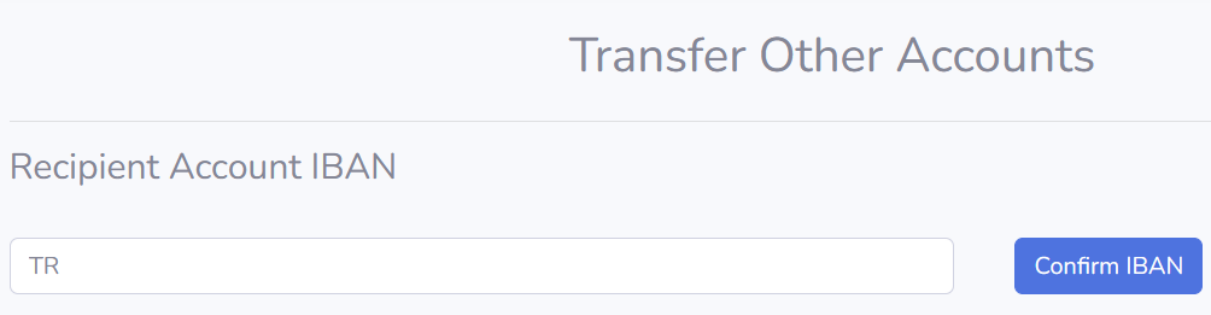
- ACCOUNT OPERATIONS
    - MY ACCOUNTS



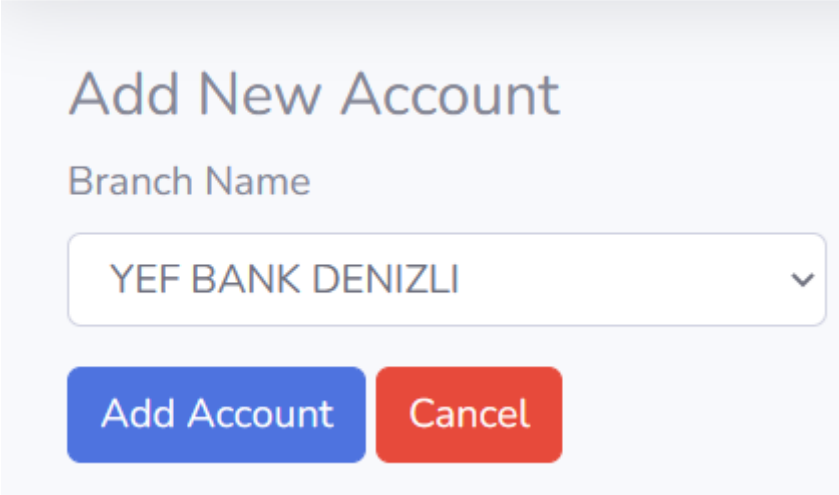
- TRANSFER



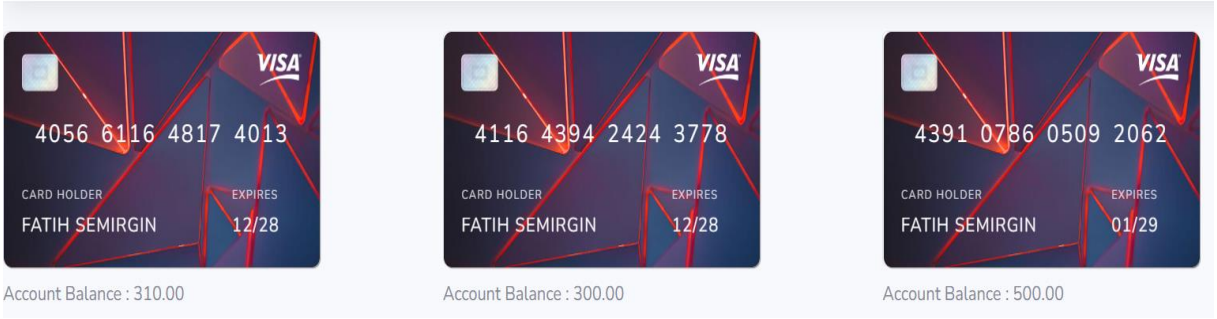
- TRANSFER OTHER ACCOUNT



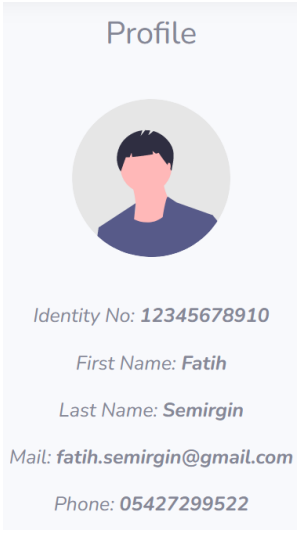
- ADD NEW ACCOUNT



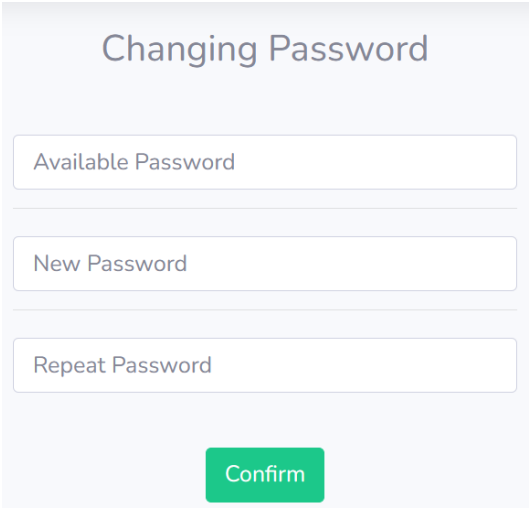
- MY CARDS



- PROFILE AND SETTINGS



  - CHANGING PASSWORD



- LOGIN RECORDS

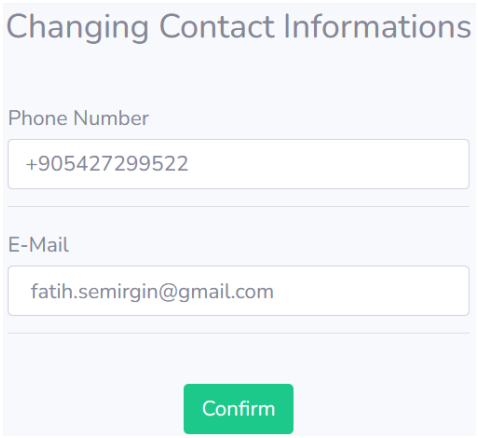
Last 5 Successful Log-ins

#	Date
1	2023-01-01 16:57:08
2	2023-01-01 16:19:50
3	2023-01-01 16:14:23
4	2023-01-01 16:03:29
5	2023-01-01 15:59:59

Last 5 Wrong Log-ins

#	Date
1	2022-12-12 01:44:48
2	2022-12-12 01:14:26
3	2022-12-11 21:20:40
4	2022-12-11 21:11:46

- CHANGING INFORMATIONS

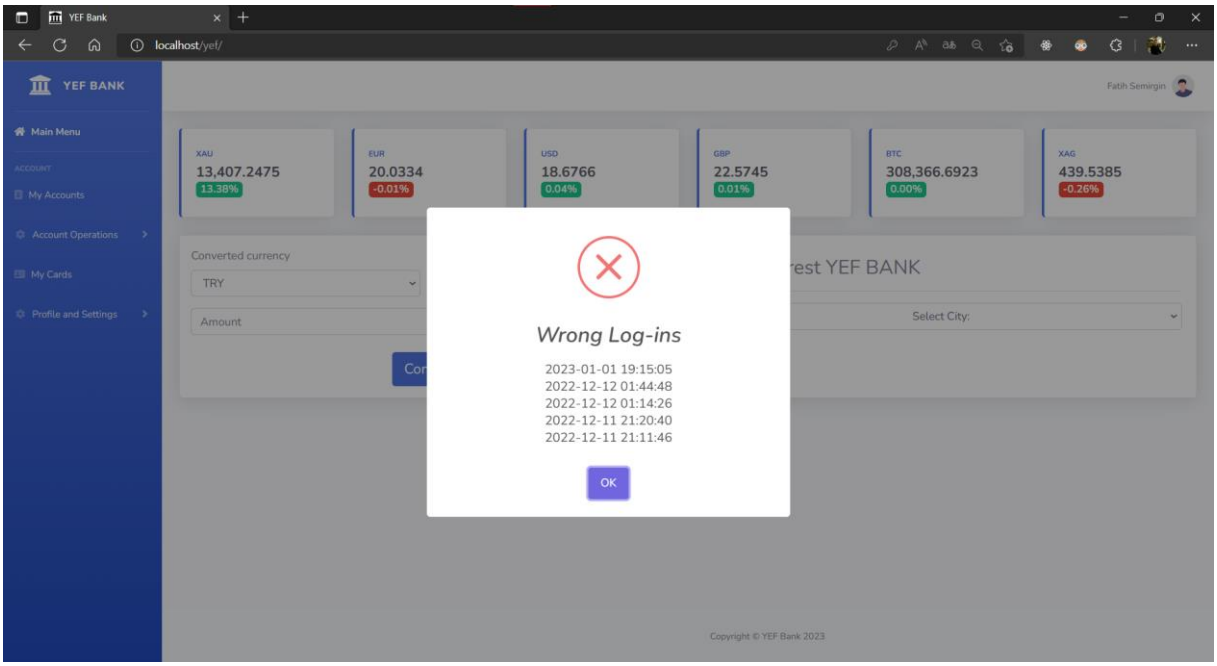


- TRANSACTION HISTORY

Transaction History

#	Date	IBAN Sender	IBAN Recipient	Amount
1	2022-12-31 19:29:54	TR8374782366852877706679 33	TR7440966906383456489466 81	22.000$
2	2022-12-31 19:31:26	TR8374782366852877706679 33	TR7440966906383456489466 81	13.000$
3	2022-12-31 19:37:07	TR8374782366852877706679 33	TR8951362852264484369317 39	240.000$
4	2022-12-31 19:37:59	TR8374782366852877706679 33	TR8951362852264484369317 39	150.000$

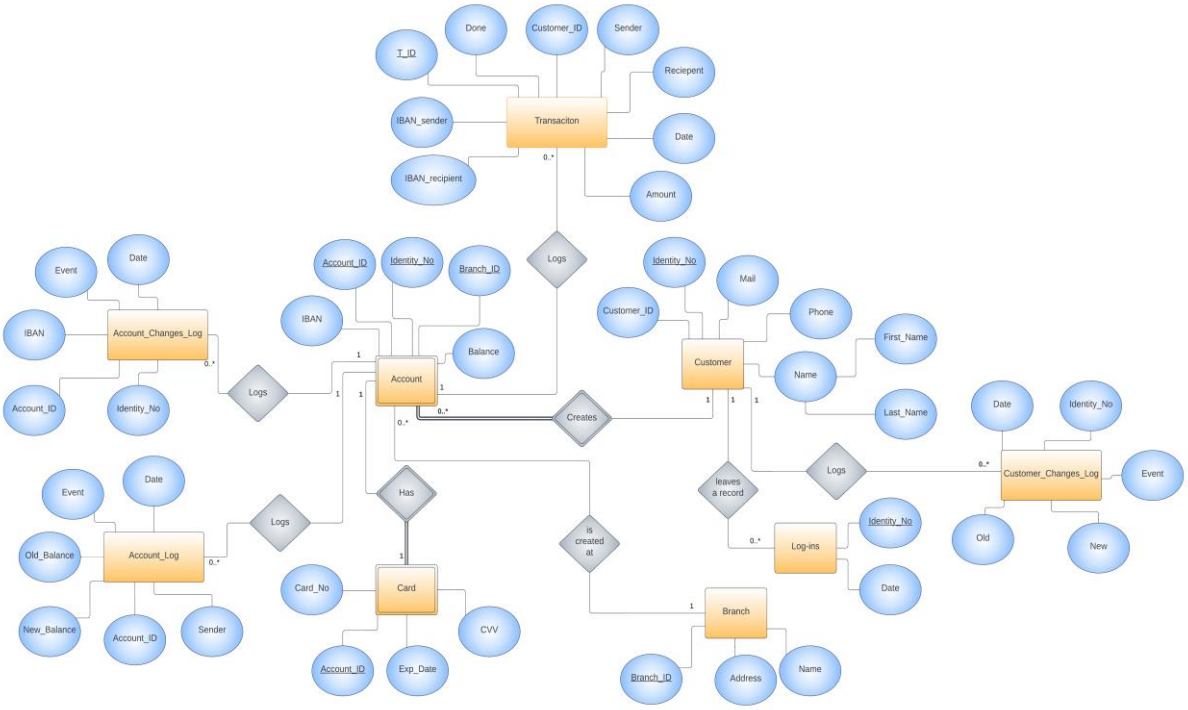
- WRONG LOGIN



The account operations section of a bank website is typically a secure area of the website that allows users to view and manage their accounts. This section may include various features and functions that allow users to perform tasks such as:

1) View account balances and transaction history
2) Transfer money between accounts
3) Transaction history
4) View and update account information, such as contact details or security settings
5) Wrong Login (First correct login after incorrect password)

H) ER DIAGRAM


Done
Customer_ID
Sender
T_ID
Reciepent
IBAN_sender
Transaciton
Date
IBAN_recipient
Amount
0..*
Logs
Event
Date
Account_ID
Identity_No
Branch_ID
Identity_No
Mail
IBAN
Account_Changes_Log
IBAN
Customer_ID
Phone
First_Name
Balance
Account
Customer
Name
Account_ID
Identity_No
Logs
1
1
1
1
1
1
0..*
Creates
Last_Name
Date
Identity_No
Event
Date
Has
Logs
leaves
a record
Logs
Customer_Changes_Log
Event
Old_Balance
Account_Log
is
created
at
Identity_No
Old
New
Log-ins
Card_No
Card
Date
New_Balance
Account_ID
Sender
CVV
Branch
Account_ID
Exp_Date
Branch_ID
Address
Name