VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi-560018, Karnataka

Database Management System Lab with Mini Project (18IS5DLDBL)

REPORT

On

"ATM Machine Interface"

BACHELOR OF ENGINEERING

In

INFORMATION SCIENCE AND ENGINEERING

Submitted by:
SHASHANK (1DS18IS063)
SHETTY SHREYAS UDAYA (1DS18IS064)
SUMUKHA N SHILIGE (1DS18IS076)
MOHAMMED YASIR (1DS18IS121)

FACULTY IN-CHARGE:
Mrs. Bhavani K

DSCE

Dept. of Information Science & Engg

2020-2021

Department Of Information Science and Engineering

DAYANANDA SAGAR COLLEGE OF ENGINEERING

SHAVIGE MALLESHWARA HILLS, KUMARASWAMY LAYOUT
Bengaluru-560078

Department of Information Science & Engineering

DSCE

Dept. of Information Science & Engg

2020-21

CERTIFICATE

This is to certify that Management and Entrepreneurship, a report entitled
"ATM Machine Interface" is a bonafide work carried out by **Shashank, Shetty shreyas udaya, Sumukha N shilige, Mohammed Yasir** with **USN 1DS18IS063, 1DS18IS064, 1DS18IS076, 1DS18IS121** in the partial fulfillment for the **5th semester** of Bachelor of Engineering in **Information Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2020-21.

Signature of Faculty in-charge:

_____
[Mrs. Bhavani K]

Marks	
Maximum	Obtained

ABSTRACT

An Automated Teller Machine (ATM) allows customers to perform banking transactions anywhere and at anytime without the need of human teller. By using a debit or ATM card at an ATM, individuals can withdraw cash from checking or savings accounts, make a deposit or transfer money from one account to another or perform other functions. You can also get cash advances using a credit card at an ATM.

The ATM is online with the bank, that is, each transaction will be authorized by the bank on-demand and directly debited from the account's owner. The ATM works as follows. First, the client will give his/her client card number in the ATM and then the ATM will ask for a Personal Identification Number (PIN) if PIN is authorized then he/she can make transactions.

ATM Machine in interface is web app made with Django python web framework and user interface made with HTML and CSS

Contents

Introduction

1.1 Background

A database is an organized collection of data, generally stored and accessed electronically from a computer system. Where databases are more complex, they are often developed using formal design and modeling techniques.

The database management system (DBMS) is the software that interacts with end users, applications, the database itself to capture and analyze the data and provides facilities to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a "database system". Often the term "database" is also used to loosely refer to any of the DBMS, the database system or an application associated with the database. The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. Typical database administration tasks supported by the DBMS include change management, performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity

1.2 Introduction to ATM Banking Database

ATM Banking database is system for bank customer to withdraw or check transactions of their account. It contains customers details like name,email,phone number etc. Its very user friendly interface which help customers to use in easy way.

2.0 E R Diagram and Relational Schema

2.1 Description of ER Diagram

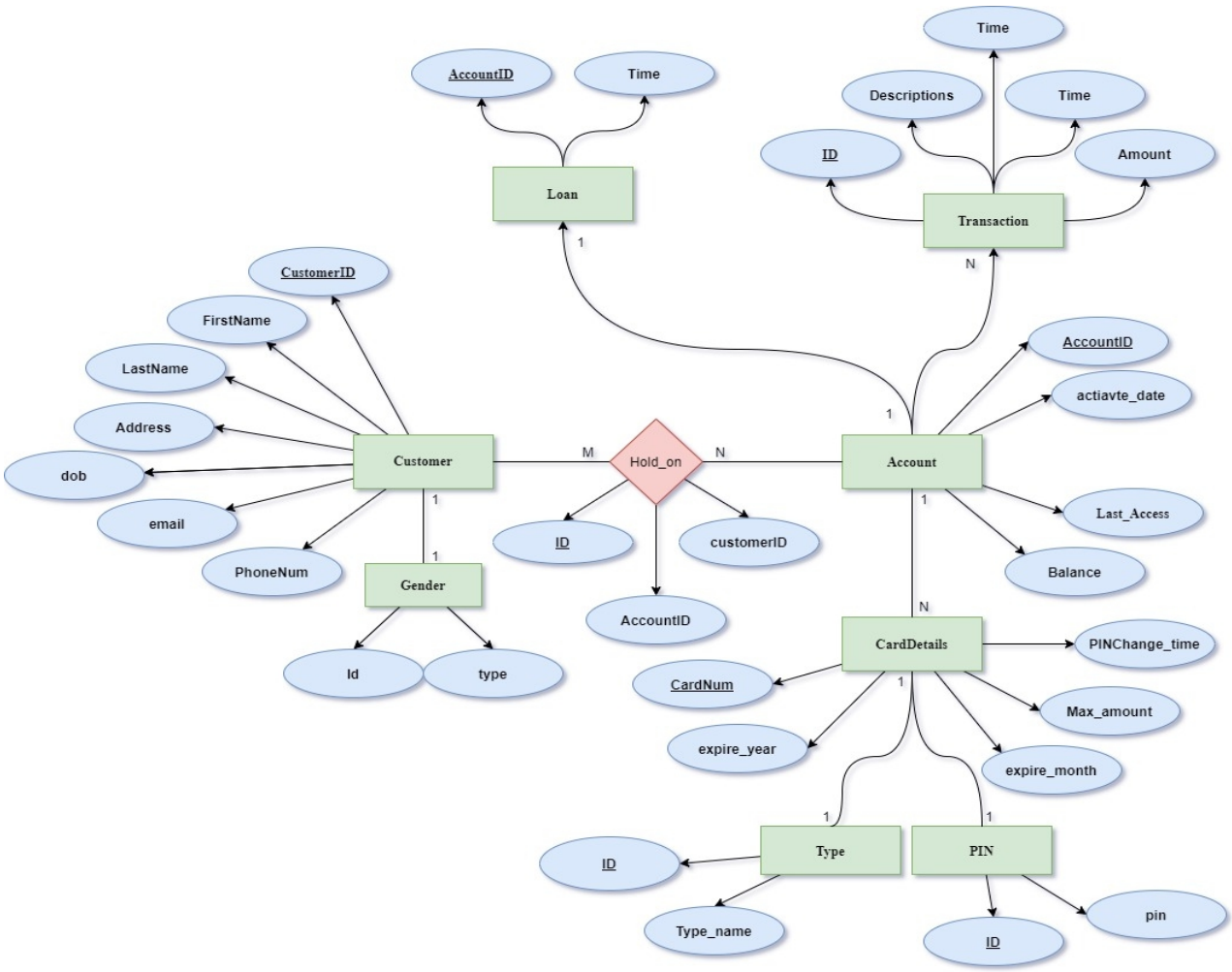

Fig: ER Diagram for the ATM Banking Management Database

The ER diagram describes the entities, attributes and relationships

- The ER diagram describes the entities, attributes and relationships. Entity types like CUSTOMER & ACCOUNT are in rectangular boxes.
- Relationships like HOLD_ON are in diamond boxes, attached to entity types with straight lines.
- Attributes are shown in ovals, each attached by a straight line to the entity or relationship types.
- Key attributes like CUSTOMERID, ACCOUNTID and ID are underlined.
- Relational attributes are ID,AccountID,CustomerID
- All the entities are in total participation in every relation.

Description of Relational Schema Diagram

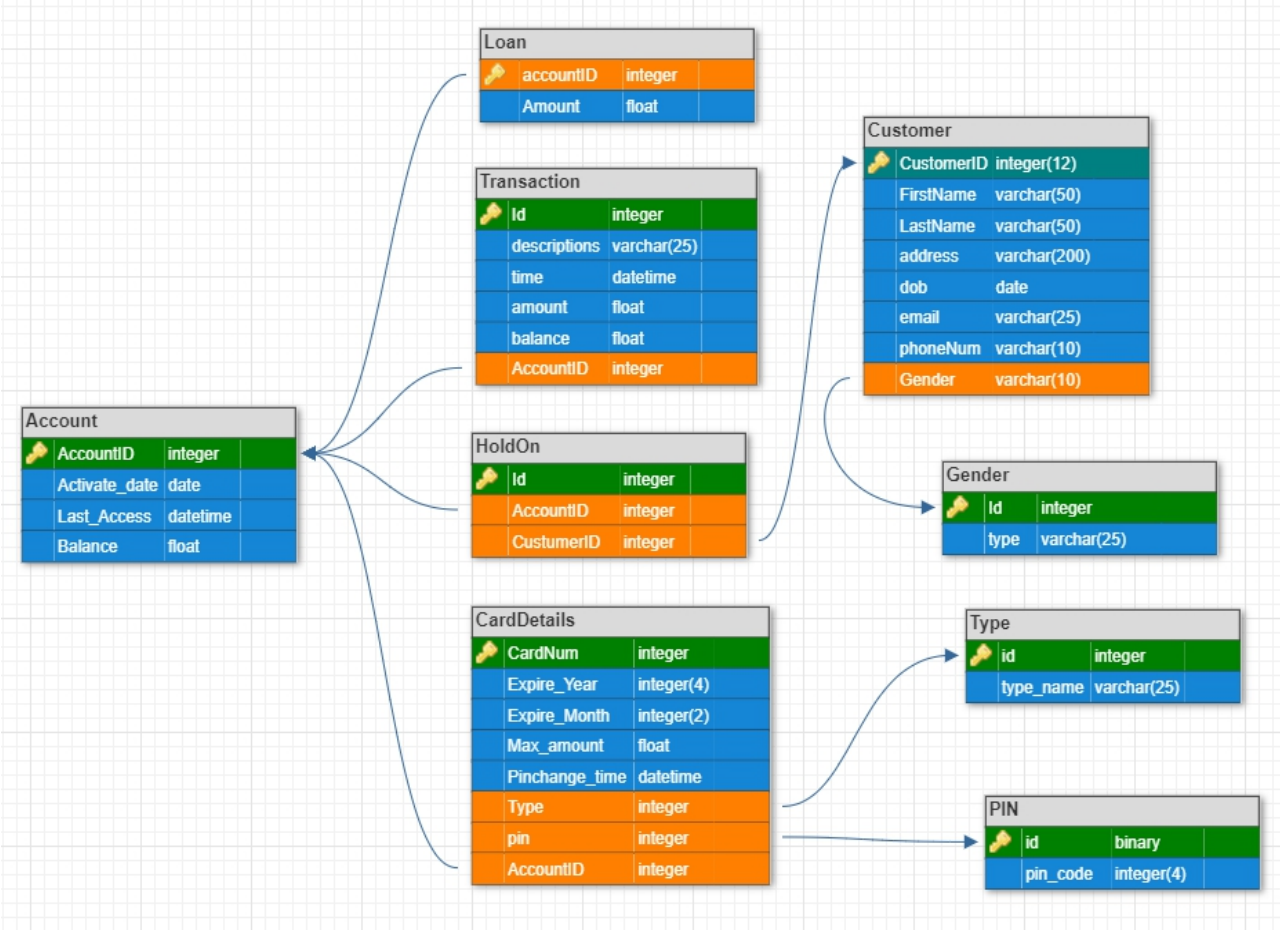

Fig: Relational schema diagram for Campus Recruitment Management Database

The term database schema refers to the description of database that includes the database structure and various constraints on the database. The schema diagram is in turn an illustrative display of the database schema. The primary keys are underlined and the referential integrity constraints are depicted by arrows pointing to the keys they reference.

General Constraints

1. **NULL Constraint**: Attributes that are under NOT NULL constraints have to be filled compulsorily. Almost all the attributes in the project are under NOT NULL constraint.
2. **Entity Integrity Constraint:** This constraint makes sure that no primary key can have a NULL value assigned to it. The primary keys involved in the project include:
   - customerID
   - AccountID
   - TransactionID
   - pinID
3. **Referential Integrity Constraints:** A table in the back end of the project may have references pointing to an attribute in another table. For example: accountID in the ACCOUNT table refers to idAccount in TRANSACTION table. The various tables are also linked with multiple foreign keys which are all set to cascade any update or delete operation on the attribute in the main table.

   The various Foreign Key attributes are:
   - idAccount
   - Idcustomer
   - pinID
   - genderID
   - statusID

SCHEMA DESCRIPTION:

Customer :- This table give us the details about all the customers registered in database. It consists of Customer id, first name, last name, address, email, phone number, image path.

Account :- This table contains account details which can have one or more than customers. It consists of Account Number, Account activation date, Account last accessed time

Transaction :- This table contains details of transactions made with all accounts are inserted to this table. It consists of Transaction ID , Time, Descriptions, Amount, Balance

CardDetails :- It contains details of the card in which many cards can have associated with one account. It consists of Card Number, Card type, expire year and month ,max amount, pin ID

Loan :- It Contains the total amount of loan withdraw with credit card.

3.0 SYSTEM DESIGN
3.1 Table Description

CUSTOMER :
This table contains the details of the customers. It consists of CustomerID, FirstName, LastName, Address, Email, PhoneNum, Image, gender. Here CustomerID is the primary key and foreign key for the GENDER

FIELD	TYPE	NULL?	KEY	DEFAULT
CustomerID	VarChar(12)	NOT NULL	Primary Key	None
FirstName	VarChar(50)	NOT NULL		None
LastName	VarChar(25)	NOT NULL		None
Address	LONGTEXT	NOT NULL		None
Email	VarChar(25)	NOT NULL		None
PhoneNum	VarChar(10)	NOT NULL		None
Image	VarChar(25)	NOT NULL		None
gender	INT	NOT NULL	Foreign Key	None

Account :
This table contains the Account details. It consists of AccountID, Last_Access, Activate_date, Balance, status. Here AccountID is the primary key and foreign key for the STATUS

FIELD	TYPE	NULL?	KEY	DEFAULT
AccountID	INT	NOT NULL	Primary Key	None
Last_Access	DateTime	NOT NULL		Auto_now
Activate_date	Date	NOT NULL		Auto_now
Balance	Double	NOT NULL		None
status	INT	NOT NULL	Foreign Key	None

CardDetails :

This table contains the Card details. It consists of CardNum, Expire_Year, Expire_Month, PINChange_time, AccountID, Type, PIN. Here AccountID is the primary key and foreign key for the STATUS

FIELD	TYPE	NULL?	KEY	DEFAULT
CardNum	VarChar(12)	NOT NULL	Primary Key	None
Expire_Year	INT	NOT NULL		None
Expire_Month	INT	NOT NULL		None
PINChange_time	DateTime	NOT NULL		Auto_now
AccountID	INT	NOT NULL	Foreign Key	None
Type	INT	NOT NULL	Foreign Key	None
PIN	INT	NOT NULL	Foreign Key	None

Transaction :

This table contains the Card details. It consists of ID, Descriptions, time, AccountID, Amount, Balance. Here ID is the primary key and foreign key for the Account

FIELD	TYPE	NULL?	KEY	DEFAULT
ID	INT	NOT NULL	Primary Key	None
Descriptions	VarChar(50)	NOT NULL		None
time	DateTime	NOT NULL		Auto_now
AccountID	INT	NOT NULL	Foreign Key	None
Amount	Double	NOT NULL		None
Balance	Double	NOT NULL		None

Loan :

This table contains the Card details. It consists of IAccountID, Amount. Here AccountID is the primary key and foreign key for the Account

FIELD	TYPE	NULL?	KEY	DEFAULT
AccountID	INT	NOT NULL	Primary Key, Foreign Key	None
Amount	Double	NOT NULL		None

4.0 IMPLEMENTATION

4.1 Front-end Development

The front-end is built using a combination of technologies such as Hypertext Markup Language (HTML), JavaScript and Cascading Style Sheets (CSS). Front-end developers design and construct the user experience elements on the web page or app including buttons, menus, pages, links, graphics and more.

Hypertext Markup Language

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create.
HTML is the standard markup language for creating Web pages. It stands for Hyper Text Markup Language. It describes the structure of a Web page. It

consists of a series of elements. Its elements tell the browser how to display the content. Its elements are represented by tags. HTML tags label pieces of content such as "heading", "paragraph", "table", and so on. Browsers do not display the HTML tags, but use them to render the content of the page.

Cascading style sheets

CSS stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces.
CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications. Before CSS, tags like font, color, background style, element alignments, border and size had to be repeated on every web page. This was a very long process. CSS solved that issue.
SS style definitions are saved in external CSS files so it is possible to change the entire website by changing just one file. CSS provides more detailed attributes than plain HTML to define the
look and feel of the website

Back-end Development

Backend is server side of the website. It stores and arranges data, and also makes sure everything on the client-side of the website works fine. It is the part of the website that you cannot see and interact with. It is the portion of software that does not come in direct contact with the users. The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application. Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend

Backend scripting language - Python Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

Database – MySQL MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. It is developed, marketed and supported by MySQL AB, which is a Swedish company. It is released under an open-source license. So, you have nothing to pay to use it. It is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc. It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customization. The open source GPL license allows programmers to modify the MySQL software to fit their own specific environments

4.3 Insertion of few records in the database

Customer :

CustomerID	First Name	Last Name	Address	dob	email	image	Phone Num	gender
11111	Sharukh	khan	Mannat, Land's End, Bandstand, Bandra West, Mumbai 400050, Maharashtra, India	1965-11-02	srk@khan.com	8474154555	profile/srk-1.jpeg	1
22222	Deepika	Padukone	B Wing, Cozihom, Pali Hill, Bandra West, Mumbai, Maharashtra.	1986-01-05	deepika@manager.com	8794654854	profile/unnamed.jpg	2
33333	Hrithk	Roshan	Chire Bandee, 10th N. S. Road, JVPD Scheme, Mumbai – 400049	1974-01-10	hrith@roshan.com	9878455554	profile/hrithik-6.jpg	1
44444	Joseph Vijay	Chandrasekhar	64, Kaveri Street, Saligramam, Chennai – 600093, Tamil Nadu, India	1974-06-22	vijay@thalapthy.com	8798849484	profile/Vijay-Tamil-Actor.jpg	1

Account:

AccountID	Activate date	Last access	balance	status
1	1994-05-25	2021-01-12 12:59:29.000000	404	1
2	1996-01-10	2021-01-12 12:56:30.000000	4011	1
3	2001-01-25	2021-01-12 12:47:50.000000	31547	1
4	2018-01-19	2021-01-10 13:49:19.000000	59494	1

Transaction:

ID	description	time	amount	balance	Account ID
1	debit	2021-01-05 18:31:55.469584	1500	63045	4
2	debit	2021-01-05 18:33:03.583609	551	62494	4
3	debit	2021-01-06 11:51:09.642645	2000	49464	3
4	debit	2021-01-06 11:51:31.194933	4000	45464	3

CardDetails :

Card Num	Expire Year	Expire Month	Max Amount	PinChange Time	Account ID	type	PIN
142879328274	2022	9	75000	2021-01-07 15:19:42.000000	10	2	9
145441451455	2019	4	25000	2021-01-05 17:46:50.928183	1	2	1
147855487632	2023	5	50000	2021-01-06 19:42:30.000000	2	1	2
231212121512	2024	1	50000	2021-01-07 15:19:42.000000	8	2	7

Loan :

Account ID	Amount
1	2000
2	2000
7	3000

4.4 NORMALIZATION

- **Third Normal Form(3NF)**

Third normal form (**3NF**) is a normal form that is used in normalizing a database design to reduce the duplication of data and ensure referential integrity by ensuring that:

1. The entity is in second normal form.
2. No non-prime (non-key) attribute is transitively dependent on any key i.e. no non-prime
attribute depends on other non-prime attributes. All the non-prime attributes must depend
only on the candidate keys.

5.0 Stored Procedures

A stored procedure is a set of Structured Query Language (SQL) statements with an assigned name, which are stored in a relational database management system as a group, so it can be reused and shared by multiple programs.

```
DELIMITER $$
CREATE PROCEDURE SelectTransaction (IN id INT)
BEGIN
SELECT * FROM atm_db.atm_app_transcations
   WHERE    account_id_id = id;
END$$
DELIMITER ;
```

This stored procedure will help us to determine transactions of particular account

6.0 Triggers

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database.

```
DELIMITER $$
CREATE TRIGGER change_lastAccess AFTER INSERT ON
atm_db.atm_app_transcations
FOR EACH ROW
BEGIN
        UPDATE atm_db.atm_app_accounts SET
last_access=CURRENT_TIMESTAMP
      WHERE idAccount=NEW.account_id_id;
END$$
DELIMITER ;
```

When TRANSACTION table inserted with new value this trigger change the last access of account with account id from newly inserted row

```
DELIMITER $$
CREATE TRIGGER last_passChange AFTER UPDATE ON
atm_db.atm_app_pin
FOR EACH ROW
BEGIN
        UPDATE atm_db.atm_app_carddetails SET
last_pinChange=CURRENT_TIMESTAMP
      WHERE pin_id_id=NEW.id;
END$$
DELIMITER ;
```

When PIN table updated with new value then triggers change the pinChange_time to current time in Carddetails tabl

7.0 Code Review

This section talks about the backed developed in Django which queries the database. This is the part which connects the frontend HTML code and the backend database.

This code shows the working of check balance

```
def check_balance(request):
    num=request.session.get('num')
    account_num=cardDetails.objects.all().filter(numCard=num).values_list('idAccount',flat=True)[0]
    account_balance=Accounts.objects.all().filter(pk=account_num).values_list('balance',flat=True)[0]
    last_access=Accounts.objects.all().filter(pk=account_num).values_list('last_access',flat=True)[0]
    return(render(request,'checkbl.html',{'account_balance':account_balance,'last_access':last_access}))
```

This code shows how customer withdraw money from their accounts

```
def amount_with(request):
    if request.method=='POST':
        num=request.session.get('num')
        amount=float(request.POST['amount'])
        max_amount=cardDetails.objects.all().filter(numCard=num).values_list('max_amount',flat=True)[0]
        if amount<=max_amount:
            account_num=cardDetails.objects.all().filter(numCard=num).values_list('idAccount',flat=True)[0]
            account_balance=Accounts.objects.all().filter(pk=account_num).values_list('balance',flat=True)[0]
            account=Accounts.objects.get(pk=account_num)
            if request.session.get('submit_type')=='debit':
                if amount<=account_balance-250:
                    new_balance=account_balance-amount
                    Accounts.objects.all().filter(pk=account_num).update(balance=new_balance)
                    trans=transcations(account_id=account,description='debit',balance=new_balance,amount=amount)
                    trans.save()
                    return(render(request,'collectYourMoney.html'))
                else:
                    messages.warning(request,'Account balance is not sufficient')
                    return(render(request,'withamount.html'))
            else:
                try:
                    obj=loan(pk=account_num,ammount=amount)
                    obj.save()
                except:
                    obj=loan.objects.get(pk=account_num)
                    current_loan=obj.ammount
                    loan.objects.all().filter(pk=account_num).update(ammount=current_loan+float(amount))
                trans=transcations(account_id=account,description='credit',balance=account_balance,amount=amount)
                trans.save()
                return(render(request,'collectYourMoney.html'))
        else:
            messages.error(request,'limit exceded')
            return(render(request,'withamount.html'))
    else:
        return(render(request,'withamount.html'))
```

This code shows how pin changing feature works

```
def pinchange(request):
    if request.method=='POST':
        new_pass=request.POST['new_pass']
        re_pass=request.POST['re_pass']
        if new_pass==re_pass:
            num=request.session.get('num')
            old=cardDetails.objects.get(pk=num)
            old_pass=old.pin_id.code
            if old_pass!=new_pass:
                pin.objects.all().filter(pk=old.pin_id.id).update(code=new_pass)
                return(render(request,'successful.html'))
            else:
                messages.error(request,'New PIN same as old PIN')
                return(render(request,'pchange.html'))
        else:
            messages.error(request,'password not matched')
            return(render(request,'pchange.html'))
    else:
        return(render(request,'pchange.html'))
```

This code shows how to get transaction details of the account using stored procedures

```
def my_custom_sql(num):
    with connection.cursor() as cursor:
        cursor.execute("CALL  SelectTransaction(%s)", [num])
        row = cursor.fetchall()
    return row

def history(request):
    num=request.session.get('num')
    account_num=cardDetails.objects.all().filter(numCard=num).values_list('idAccount',flat=True)[0]
    hist=tuple(reversed(my_custom_sql(account_num)))
    return(render(request,'history.html',{'hist':hist}))
```

Report Generating :

ATM system generate the below two reports from TRANSACTION table in which sum of amount is group by accountID which result in below Bar Plot showing the transaction amount of 24 hours

Below Pie chart shows the how much amount of money credit and debit in a day