

AMERICAN INTERNATIONAL UNIVERSITY BANGLADESH  
Faculty of Science And Technology

Midterm Report Cover Sheet

*Students must complete all details except the faculty use part.*



**Title: Paying Guest Management System**

Due Date: 28-11-2023 Semester: Fall 23-24 Degree Program: CSE

Subject Name: ADVANCE DATABASE MANAGEMENT SYSTEM Section: B

Course Instructor: Rezwan Ahmed

**Declaration and Statement of Authorship:**

1. I/we hold a copy of this report, which can be produced if the original is lost/ damaged.
2. This report is my/our original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgement is made.
3. No part of this report has been written for me/us by any other person except where such collaboration has been authorized by the lecturer/teacher concerned and is clearly acknowledged in the report.
4. I/we have not previously submitted or currently submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared and archived for the purpose of detecting plagiarism.
6. I/we give permission for a copy of my/our marked work to be retained by the School for review and comparison, including review by external examiners.

**I/we understand that**

7. Plagiarism is the presentation of the work, idea or creation of another person as though it is your own. It is a form of cheating and is a very serious academic offence that may lead to expulsion from the University. Plagiarized material can be drawn from, and presented in, written, graphic and visual form, including electronic data, and oral presentations. Plagiarism occurs when the origin of the material used is not appropriately cited.
8. Enabling plagiarism is the act of assisting or allowing another person to plagiarize or to copy your work

Group Number (if applicable):



Individual Submission



Group Submission

No.	Student Name	Student Number	Student Signature	Date
<b>Group Members:</b>				
1	Md. Shariar Hosain Sanny	21-44677-1		28-11-23
2	Shariar Rahman Apurbo	21-44738-1		28-11-23
3	Sanjida Jahan	21-44684-1		28-11-23

*For faculty use only:*

Total Marks: \_\_\_\_\_ Marks Obtained: \_\_\_\_\_

Faculty comments \_\_\_\_\_

## Table of Contents

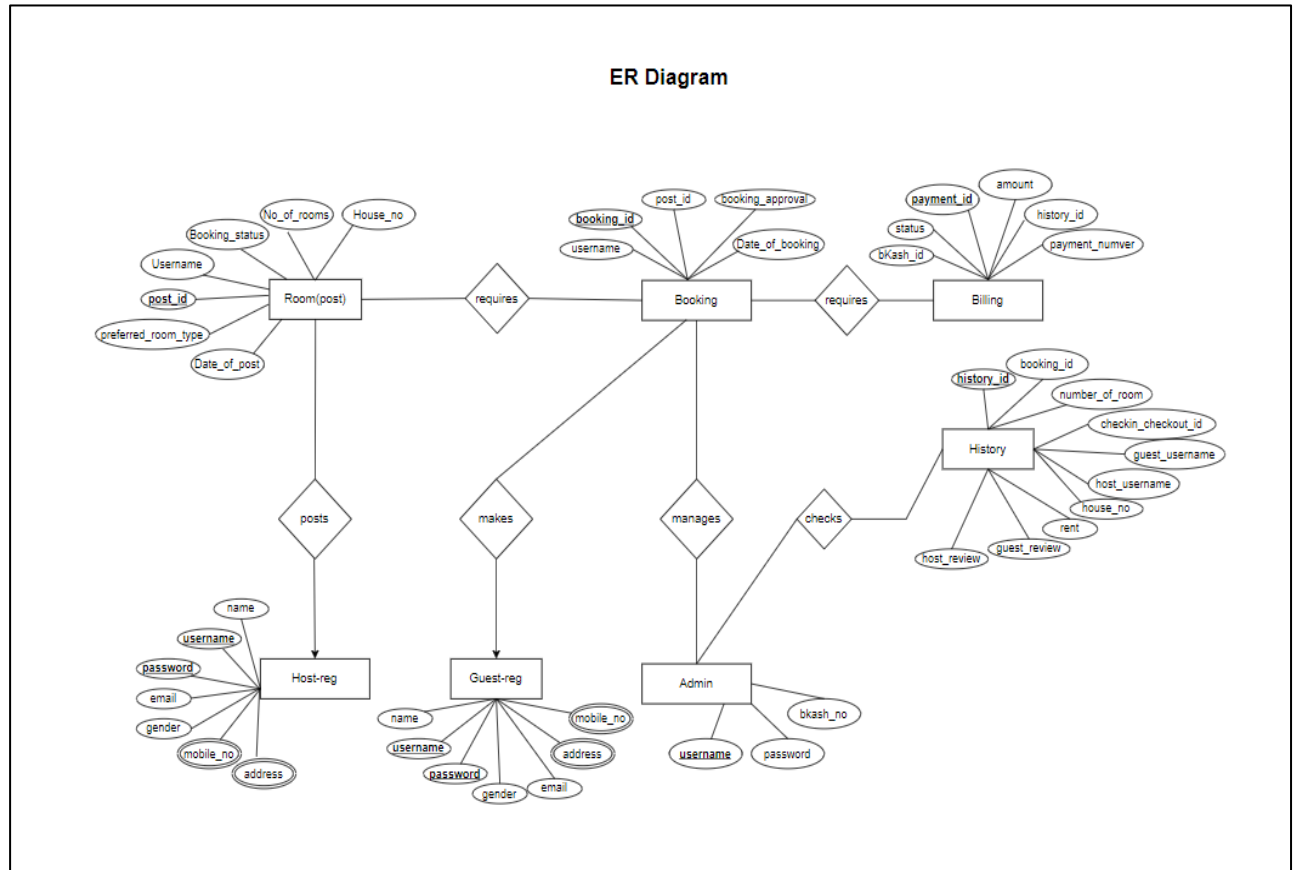
1. System summary.....	3
2. Entity Relationship Diagram .....	4
3. Use Case Diagram .....	5
4. Activity Diagram .....	6
5. Database schema diagram .....	7
6. Screen shots of sample data.....	7
7. Demonstration of database scenarios with screenshot .....	9
8. Design at least 10 questions. ....	12
9. User Interface .....	17

## **1. System summary**

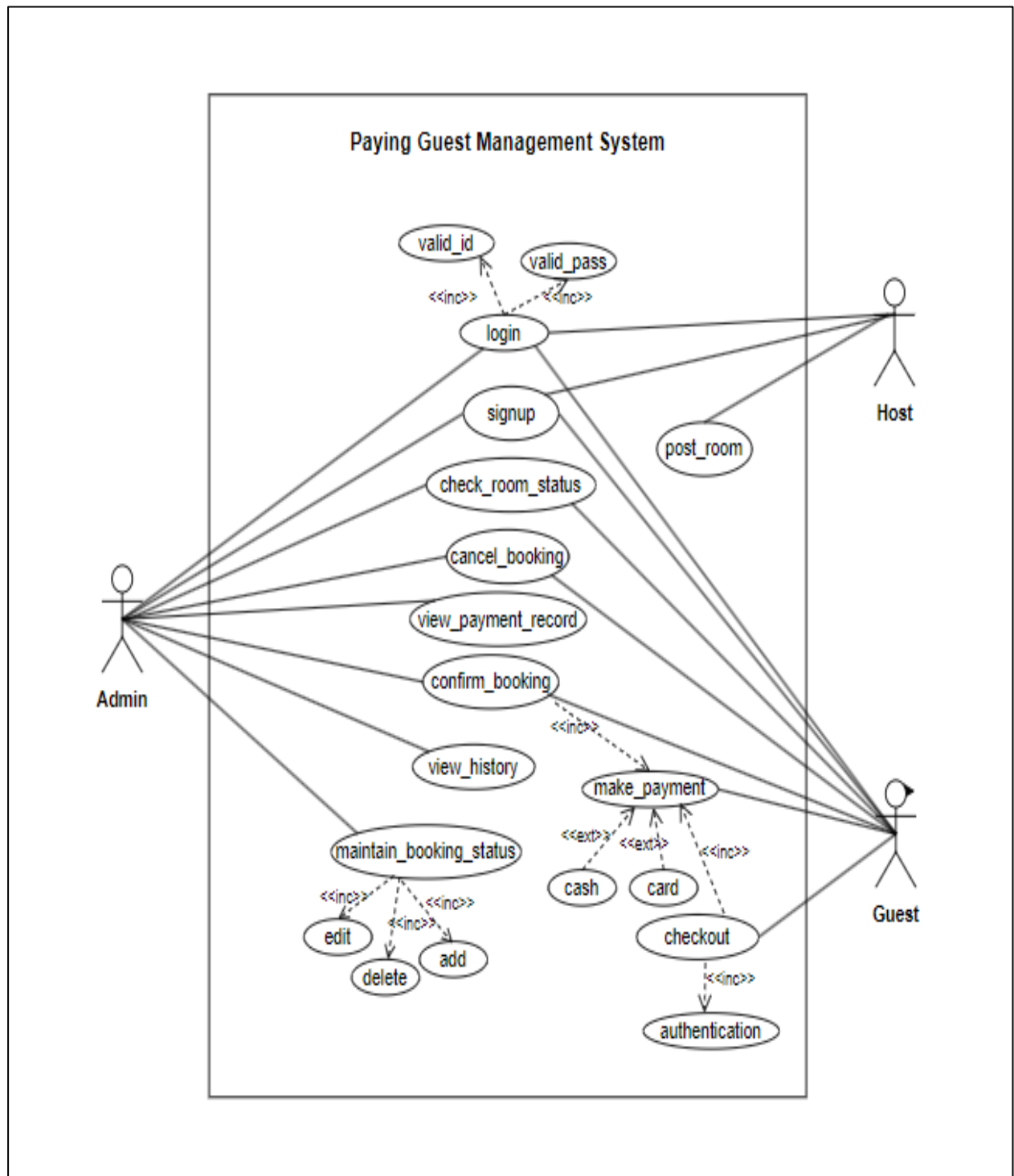
The Paying Guest Management System is a software application that allows hosts to manage their tenants and accommodation details in one place. The system has three types of users: admin, guest, and host. The host can post a room with details that can be edited, activated, or deactivated. If the posted room is empty, then a guest can book the room, and the booking status changes to booked. The booking request is sent to the admin. The admin can either reject or confirm the booking request. If the admin confirms the booking request, then the booking room details show up on the check-in page, where the guest can check in. If the admin rejects the booking request, then the booking status will be empty. If the guest checks out of the room, then the booking status will be changed to empty. After checkout, the guest and host can share their reviews with each other. The guest can pay the room rent using the bkaash number and transaction ID. The payment request is sent to the admin to verify the details of payment. If the admin confirms the payment, then all information is stored and shown in the history of the host and guest dashboard. If the admin rejects the payment, then it will again show up on the guest dashboard to submit details again. If the guest pays their rent for the desired posted room, then the host can request a payment withdrawal request to the admin. The admin will pay the desired payment for the host.

In conclusion, the paying guest management system provides a streamlined process for booking, payment, and checkout, making it easier for both hosts and guests to manage their stay. The system also allows for reviews to be shared between guests and hosts, providing valuable feedback for future improvements. Overall, the paying guest management system is an excellent tool for anyone looking to manage their rental properties more efficiently.

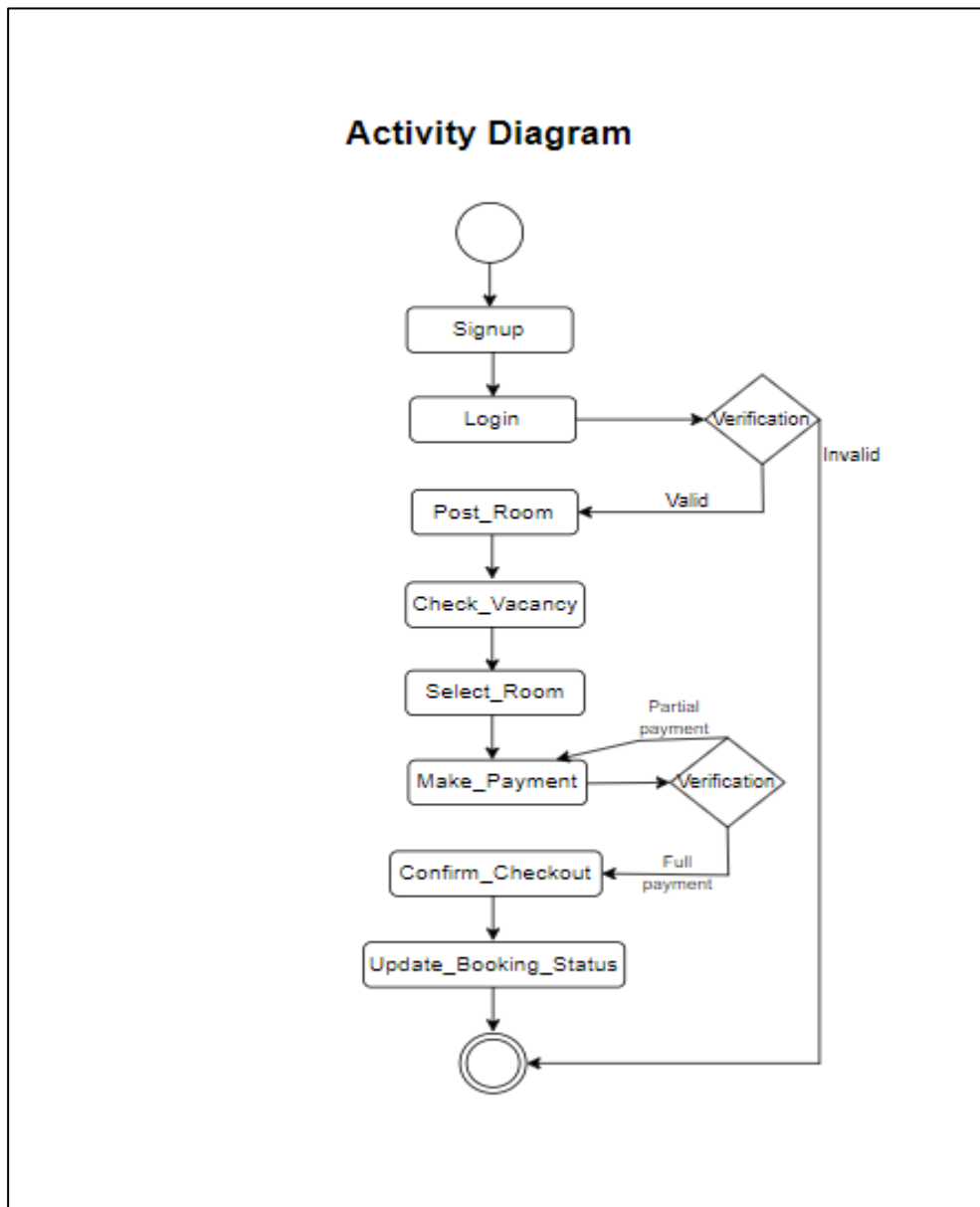
## 2. Entity Relationship Diagram



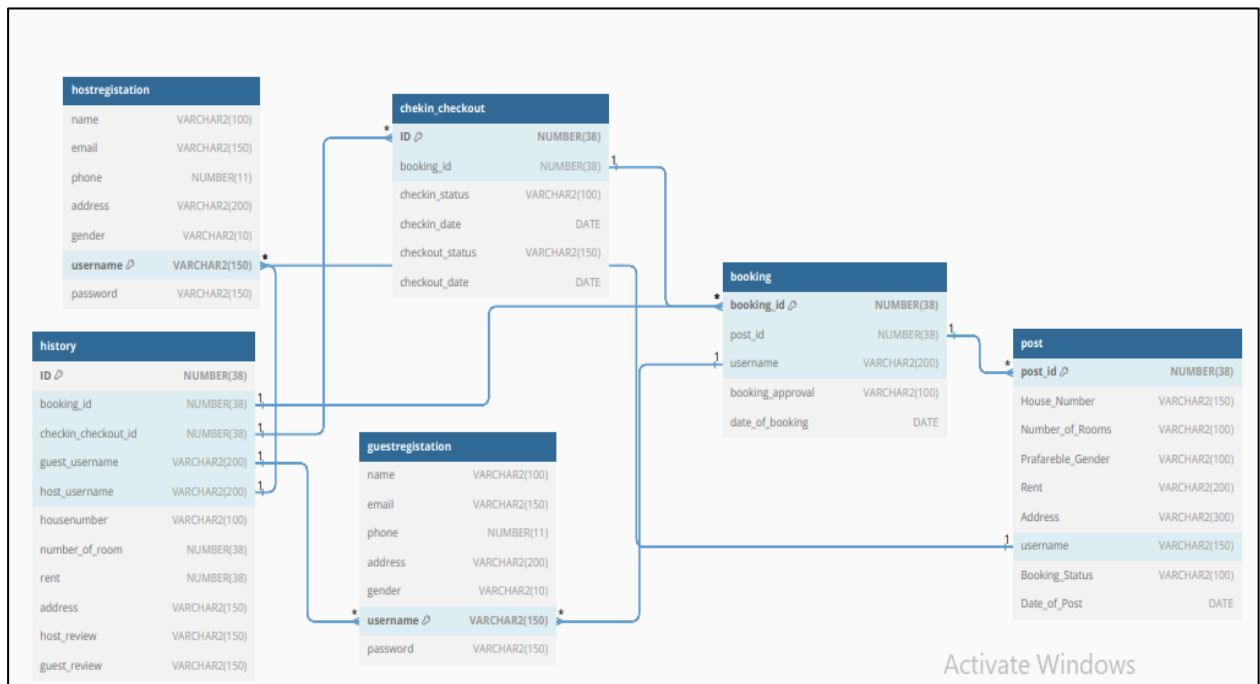
### 3. Use Case Diagram



## 4. Activity Diagram



## 5. Database schema diagram



## 6. Screen shots of sample data Create all the tables and make sure you used sequence to handle primary key values at least for two tables.

```
CREATE TABLE hostregistration (
  name VARCHAR2(100),
  email VARCHAR2(150),
  phone NUMBER(11),
  address VARCHAR2(200),
  gender VARCHAR2(10),
  username VARCHAR2(150) NOT NULL,
  password VARCHAR2(150) NOT NULL
);

CREATE TABLE guestregistration (
  name VARCHAR2(100),
  email VARCHAR2(150),
  phone NUMBER(11),
  address VARCHAR2(200),
  gender VARCHAR2(10),
  username VARCHAR2(150) NOT NULL,
  password VARCHAR2(150) NOT NULL
);

CREATE TABLE post (
  post_id NUMBER(38) NOT NULL,
  House_Number VARCHAR2(150) NOT NULL,
  Number_of_Rooms VARCHAR2(100) NOT NULL,
  Prafareble_Gender VARCHAR2(100) NOT NULL,
```

```

Rent VARCHAR2(200) NOT NULL,
Address VARCHAR2(300) NOT NULL,
username VARCHAR2(150) NOT NULL,
Booking_Status VARCHAR2(100) NOT NULL,
Date_of_Post DATE NOT NULL
);
CREATE TABLE booking (
    booking_id NUMBER(38) NOT NULL,
    post_id NUMBER(38) NOT NULL,
    username VARCHAR2(200) NOT NULL,
    booking_approval VARCHAR2(100) NOT NULL,
    date_of_booking DATE NOT NULL);
CREATE TABLE billing (
    ID NUMBER(38) NOT NULL,
    history_id NUMBER(38) NOT NULL,
    amount NUMBER(38) DEFAULT NULL,
    payment_number VARCHAR2(100) DEFAULT NULL,
    bkash_trans_id VARCHAR2(200) DEFAULT NULL,
    status VARCHAR2(150) DEFAULT NULL
);
CREATE TABLE chekin_checkout (
    ID NUMBER(38) NOT NULL,
    booking_id NUMBER(38) NOT NULL,
    checkin_status VARCHAR2(100) NOT NULL,
    checkin_date DATE ,
    checkout_status VARCHAR2(150) NOT NULL,
    checkout_date DATE
);
CREATE TABLE history (
    ID NUMBER(38) NOT NULL,
    booking_id NUMBER(38) NOT NULL,
    chekin_checkout_id NUMBER(38) NOT NULL,
    guest_username VARCHAR2(200) NOT NULL,
    host_username VARCHAR2(200) NOT NULL,
    housenumber VARCHAR2(100) NOT NULL,
    number_of_room NUMBER(38) NOT NULL,
    rent NUMBER(38) NOT NULL,
    address VARCHAR2(150) NOT NULL,
    host_review VARCHAR2(150),
    guest_review VARCHAR2(150)
);
ALTER TABLE guestregistration
ADD CONSTRAINT pk_guestregistration PRIMARY KEY (username);
ALTER TABLE hostregistration
ADD CONSTRAINT pk_hostregistration PRIMARY KEY (username);
ALTER TABLE post
ADD CONSTRAINT pk_post PRIMARY KEY (post_id);
ALTER TABLE post
ADD CONSTRAINT fk_post_hostregistration FOREIGN KEY (username)
REFERENCES    hostregistration(username);

```



```

ALTER TABLE booking
ADD CONSTRAINT pk_booking PRIMARY KEY (booking_id);
ALTER TABLE booking
ADD CONSTRAINT fk_booking_post FOREIGN KEY (post_id) REFERENCES
post (post_id);
ALTER TABLE booking
ADD CONSTRAINT fk_booking_guestregistration FOREIGN KEY (username)
REFERENCES guestregistration (username);
ALTER TABLE chekin_checkout
ADD CONSTRAINT pk_chekin_checkout PRIMARY KEY (ID);
ALTER TABLE chekin_checkout
ADD CONSTRAINT chekin_checkout_ibfk_1
FOREIGN KEY (booking_id)
REFERENCES booking (booking_id);
ALTER TABLE billing
ADD CONSTRAINT billing_ibfk_1 FOREIGN KEY (history_id) REFERENCES
history (ID);
ALTER TABLE history
ADD CONSTRAINT history_pk PRIMARY KEY (ID);
ALTER TABLE history
ADD CONSTRAINT history_ibfk_1 FOREIGN KEY (booking_id) REFERENCES
booking (booking_id);
ALTER TABLE history
ADD CONSTRAINT history_ibfk_2 FOREIGN KEY (checkin_checkout_id)
REFERENCES chekin_checkout (ID)
ALTER TABLE history
ADD CONSTRAINT history_ibfk_3 FOREIGN KEY (guest_username)
REFERENCES guestregistration (username);
ALTER TABLE history
ADD CONSTRAINT history_ibfk_4 FOREIGN KEY (host_username)
REFERENCES hostregistration (username);

```

## 7. Demonstration of database scenarios with screenshot

```
select * from guestregistration
```

Results Explain Describe Saved SQL History

NAME	EMAIL	PHONE	ADDRESS	GENDER	USERNAME	PASSWORD
momojtaj	hogamari754@gmail.com	1792542061	mirpur	female	momo	momo@2023
shariar	Bedas77@outlook.com	1721969432	w1	male	monire	zdfgg@222
shariar	shariarhosain131529@gmail.com	1757525035	mirpur14	male	sanny	sanny@2023
shariar	dheke62772@hotmail.com	1792542060	mirpur14	male	selina	selina@123

```
select * from hostregistration
```

**Results** Explain Describe Saved SQL History

NAME	EMAIL	PHONE	ADDRESS	GENDER	USERNAME	PASSWORD
shariar	dheke62772@hotmail.com	1757525040	w1	male	aaaa	sanny@203
monari	Bedas77@outlook.com	1792542063	danmondi	male	moni	moni@2023
monari	Bedas77@outlook.com	1712578200	danmondi	male	monira	moni@203s
shariar	dheke62772@hotmail.com	1757525035	mirpur13	male	sanny	sanny@2023
selina	robin8363@outlook.com	1757525034	mirpur14	female	selina	selina@2023

```
select * from post
```

**Results** Explain Describe Saved SQL History

POST_ID	HOUSE_NUMBER	NUMBER_OF_ROOMS	PRAFAREBLE_GENDER	RENT	ADDRESS	USERNAME	BOOKING_STATUS	DATE_OF_POST
1	789	1	female	2500	Mirpur14	selina	empty	11/05/2023
3	406	1	female	2600	mirpur7	selina	empty	11/05/2023
4	752	2	both	9000	mirpur7	sanny	empty	11/06/2023
5	506	2	female	5600	mirpur14	selina	empty	11/05/2023
654	758	2	both	2600	mirpur12	sanny	empty	11/09/2023
6550	789	3	both	2000	mirpur14	selina	empty	11/12/2023

```
select * from booking
```

**Results** Explain Describe Saved SQL History

BOOKING_ID	POST_ID	USERNAME	BOOKING_APPROVAL	DATE_OF_BOOKING
1699443444	4	sanny	checkout	11/08/2023
1699443445	5	momo	checkout	11/08/2023
1699443447	3	momo	checkout	11/09/2023
1699443448	1	momo	checkout	11/09/2023
1699443449	654	momo	checkout	11/09/2023
1699443451	3	momo	checkout	11/12/2023
1699443453	6550	sanny	checkout	11/13/2023

```
select * from chekin_checkout
```

**Results** Explain Describe Saved SQL History

ID	BOOKING_ID	CHECKIN_STATUS	CHECKIN_DATE	CHECKOUT_STATUS	CHECKOUT_DATE
6	1699443445	yes	11/12/2023	yes	11/12/2023
7	1699443447	yes	11/12/2023	yes	11/12/2023
8	1699443449	yes	11/12/2023	yes	11/12/2023
9	1699443448	yes	11/12/2023	yes	11/12/2023
10	1699443451	yes	11/12/2023	yes	11/12/2023
11	1699443444	yes	11/12/2023	yes	11/12/2023
13	1699443453	yes	11/13/2023	yes	11/13/2023
14	1699443447	yes	11/14/2023	no	-

8 rows returned in 0.04 seconds

[Download](#)

```
select * from billing
```

**Results** Explain Describe Saved SQL History

ID	HISTORY_ID	AMOUNT	PAYMENT_NUMBER	BKASH_TRANS_ID	STATUS
1	6	2600	017587898	xfgxf655xh	paid
2	2	5600	0251518181	xfgxf655xh	reject
3	8	9000	0179578298	xfsrsttwtwt4	paid
4	8	9000	01789555299	xfgxf6agaeg	paid

4 rows returned in 0.01 seconds

[Download](#)

```
select * from history
```

**Results** Explain Describe Saved SQL History

ID	BOOKING_ID	CHECKIN_CHECKOUT_ID	GUEST_USERNAME	HOST_USERNAME	HOUSENUMBER	NUMBER_OF_ROOM	RENT	ADDRESS	HOST_REVIEW	GUEST_REVIEW
2	1699443445	6	momo	selina	506	2	5600	mirpur14	-	good
4	1699443447	7	momo	selina	406	1	2600	mirpur7	poor	bad
5	1699443451	10	momo	selina	406	1	2600	mirpur7	-	-
6	1699443449	8	momo	sanny	758	2	2600	mirpur12	guest also valo	-
7	1699443448	9	momo	selina	789	1	2500	Mirpur14	poor	-
8	1699443444	11	sanny	sanny	752	2	9000	mirpur7	-	good
9	1699443453	13	sanny	selina	789	3	2000	mirpur14	-	good

## 8. To demonstrate some database use scenarios, design at least 10 questions

- 1) Show all the bookings made by a specific guest(like: guest username=>sanny)

```
SELECT * FROM booking WHERE username = 'sanny';
```

Results Explain Describe Saved SQL History

BOOKING_ID	POST_ID	USERNAME	BOOKING_APPROVAL	DATE_OF_BOOKING
1699443444	4	sanny	checkout	11/08/2023
1699443453	6550	sanny	checkout	11/13/2023

- 2) Show all the posts made by a specific host(like host username=>selina) that are currently empty

```
SELECT * FROM post WHERE username = 'selina' AND Booking_Status = 'empty';
```

Results Explain Describe Saved SQL History

POST_ID	HOUSE_NUMBER	NUMBER_OF_ROOMS	PRAFAREBLE_GENDER	RENT	ADDRESS	USERNAME	BOOKING_STATUS	DATE_OF_POST
1	789	1	female	2500	Mirpur14	selina	empty	11/05/2023
3	406	1	female	2600	mirpur7	selina	empty	11/05/2023
5	506	2	female	5600	mirpur14	selina	empty	11/05/2023
6550	789	3	both	2000	mirpur14	selina	empty	11/12/2023

- 3) show all the guests details who have made a booking for a specific post(like post\_id=3)

```
select guestregistration.* from guestregistration,booking
where booking.username=guestregistration.username
and booking.post_id='3'
```

**Results** Explain Describe Saved SQL History

NAME	EMAIL	PHONE	ADDRESS	GENDER	USERNAME	PASSWORD
momojtaj	hogamari754@gmail.com	1792542061	mirpur	female	momo	momo@2023
momojtaj	hogamari754@gmail.com	1792542061	mirpur	female	momo	momo@2023

2 rows returned in 0.01 seconds [Download](#)

4) Show all the guests who have checked out and left a review for the host

```
SELECT guestregistration.name,history.host_username,history.host_review
from guestregistration,history
WHERE history.guest_username=guestregistration.username
and history.host_review IS NOT null
```

**Results** Explain Describe Saved SQL History

NAME	HOST_USERNAME	HOST_REVIEW
momojtaj	selina	poor
momojtaj	selina	poor
momojtaj	sanny	guest also valo

5) Show all the billing details for a specific history record

```
SELECT * FROM billing WHERE history_id = '6';
```

**Results** Explain Describe Saved SQL History

ID	HISTORY_ID	AMOUNT	PAYMENT_NUMBER	BKASH_TRANS_ID	STATUS
1	6	2600	017587898	xfgxf655xh	paid

6) Find the post with the highest rent that is currently available

```
SELECT *  
FROM post  
WHERE Booking_Status = 'empty' AND Rent = (  
    SELECT MAX(Rent)  
    FROM post)
```

**Results** Explain Describe Saved SQL History

POST_ID	HOUSE_NUMBER	NUMBER_OF_ROOMS	PRAFAREBLE_GENDER	RENT	ADDRESS	USERNAME	BOOKING_STATUS	DATE_OF_POST
4	752	2	both	9000	mirpur7	sanny	empty	11/06/2023

1 rows returned in 0.01 seconds Download

7) Find the total number of posts made by each host, ordered by the host's name

```
SELECT username, COUNT(*) as post_count FROM post GROUP BY username ORDER BY username;
```

**Results** Explain Describe Saved SQL History

USERNAME	POST_COUNT
sanny	2
selina	4

- 8) Find the post with the highest rent that is currently available, along with the host's details

```
SELECT post.*,hostregistration.*
FROM hostregistration,post
WHERE hostregistration.username=post.username
and post.Booking_Status='empty' ORDER BY post.Rent
```

POST_ID	HOUSE_NUMBER	NUMBER_OF_ROOMS	PRAFAREBLE_GENDER	RENT	ADDRESS	USERNAME	BOOKING_STATUS	DATE_OF_POST	NAME	EMAIL	PHONE	ADDRESS
6550	789	3	both	2000	mirpur14	selina	empty	11/12/2023	selina	robin8363@outlook.com	1757525034	mirpur14
1	789	1	female	2500	Mirpur14	selina	empty	11/05/2023	selina	robin8363@outlook.com	1757525034	mirpur14
654	758	2	both	2600	mirpur12	sanny	empty	11/09/2023	shariar	dheike62772@hotmail.com	1757525035	mirpur13
3	406	1	female	2600	mirpur7	selina	empty	11/05/2023	selina	robin8363@outlook.com	1757525034	mirpur14
5	506	2	female	5600	mirpur14	selina	empty	11/05/2023	selina	robin8363@outlook.com	1757525034	mirpur14
4	752	2	both	9000	mirpur7	sanny	empty	11/06/2023	shariar	dheike62772@hotmail.com	1757525035	mirpur13

- 9) Find the number of bookings made each day

```
SELECT date_of_booking, COUNT(*) as booking_count FROM booking GROUP BY date_of_booking;
```

Results Explain Describe Saved SQL History

DATE_OF_BOOKING	BOOKING_COUNT
11/12/2023	1
11/13/2023	1
11/08/2023	2
11/09/2023	3

4 rows returned in 0.00 seconds Download

- 10) Find the total amount paid by each guest, ordered by the total amount in descending order

```
select history.guest_username,SUM(billing.amount) as total_amount
from billing,history
where billing.history_id=history.ID
and billing.status='paid'
GROUP by history.guest_username
ORDER by total_amount DESC;
```

Results Explain Describe Saved SQL History

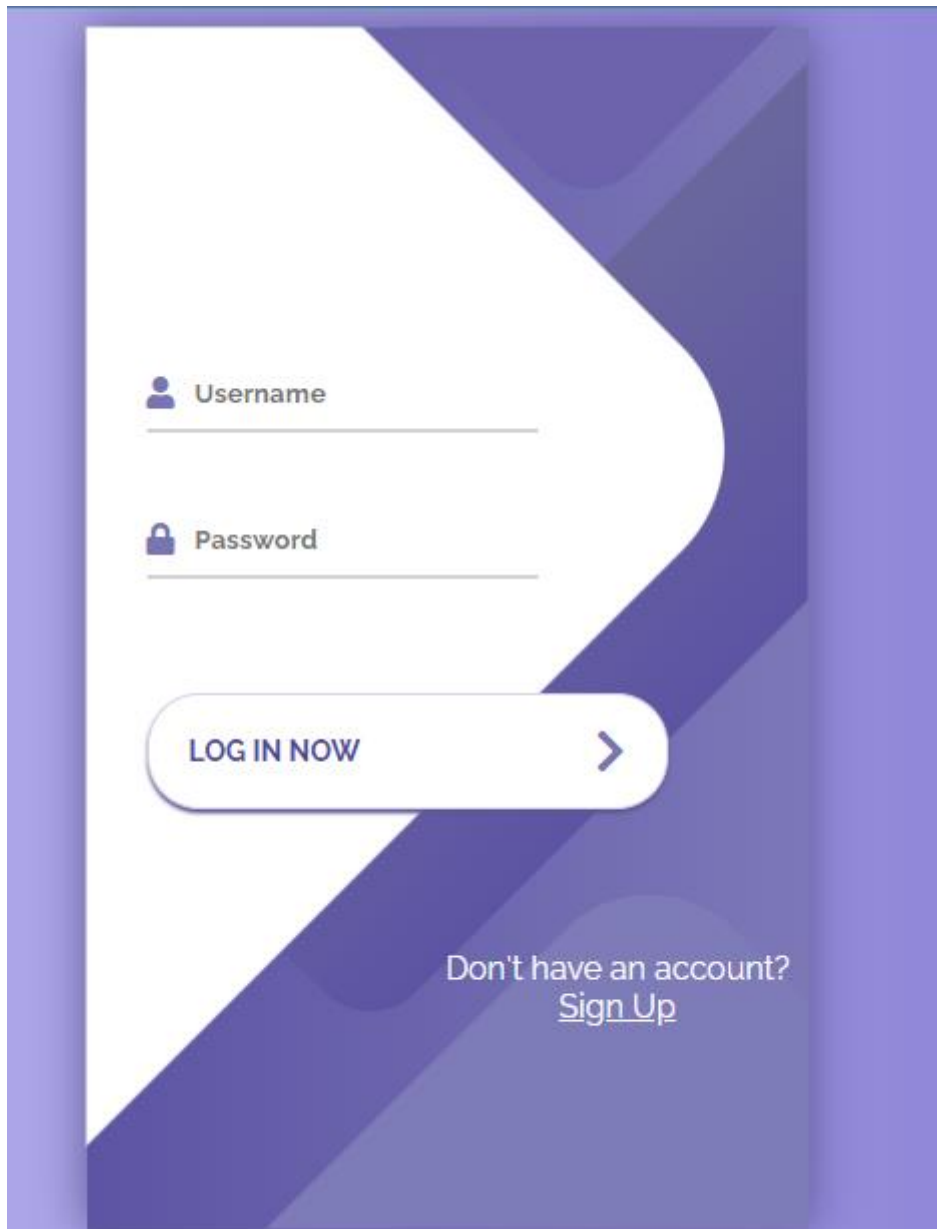
GUEST_USERNAME	TOTAL_AMOUNT
sanny	18000
momo	2600

### Queries:

1. select \* from booking WHERE username = 'sanny';
2. select \* from post WHERE username='selina' and Booking\_Status='empty';
3. select guestregistration.\* from guestregistration, booking where  
booking.username=guestregistration.username and booking.post\_id=3
4. SELECT guestregistration.name, history.host\_username, history.host\_review From  
guestregistration, history WHERE history.guest\_username=guestregistration.username  
and history.host\_review IS NOT null
5. SELECT \* FROM billing WHERE history\_id=6;
6. SELECT \* FROM post WHERE Booking\_Status = 'empty' AND Rent = (SELECT  
MAX(RENT) FROM post)
7. SELECT username, COUNT(\*) as post\_count FROM post GROUP BY username  
ORDER BY username;
8. SELECT post. \*,hostregistration.\* from hostregistration, post where  
hostregistration.username=post.username and post. Booking\_Status = 'empty' ORDER  
BY post.Rent
9. SELECT date\_of\_booking, COUNT(\*) as booking\_count FROM booking GROUP  
BY date\_of\_booking;
10. select history.guest\_username, SUM(billing.amount) as total\_amount from billing,  
history where billing.history\_id=history.ID and billing.status='paid' GROUP by  
history.guest\_username ORDER by total\_amount DESC;



## 9. User Interface



The image shows a user login interface. It features a white background with a large, stylized purple arrow pointing to the right. The arrow is composed of several overlapping, semi-transparent shapes. On the left side of the arrow, there are two input fields. The first field is labeled 'Username' and has a small purple user icon to its left. The second field is labeled 'Password' and has a small purple lock icon to its left. Below these fields is a large, rounded rectangular button with a white background and a purple border. The button contains the text 'LOG IN NOW' in a bold, sans-serif font, followed by a purple right-pointing chevron. Below the button, there is a link that says 'Don't have an account? [Sign Up](#)'. The entire interface is set against a solid purple background.

Username

Password

LOG IN NOW >

Don't have an account?  
[Sign Up](#)

# Choose user

Host Guest

Name:  Please input name

Email:  Please input email

Phone:  Please input phone number

Address:  Please input address

Gender: ☐ Male ☐ Female

Username:  Gu-

Password:  Please input password

|| [Change User](#)

Already have an account? [login](#)

Name:

Email:

Phone:

Address:

Gender: ☐ Male ☐ Female

Username: **Ho-**

Password:

[Change User](#)

Already have an account? [login](#)