Indian Institute of Information Technology, Allahabad

Database Management System Project Report

On

Mess Feedback Management System

Submitted By Group 2:

Titiksha Sharma	(IIT2022117)
Sneha Jaiswal	(IIT2022120)
Prachi Kumari	(IIT2022206)
Ananya Garg	(IIT2022208)
Trisha Bharti	(IIT2022210)
Sanjana Prajapati	(IIT2022214)

Acknowledgement

We would like to thank Dr. Soumyadev Maity, our Professor-in-charge and our assigned TA, Ms. Shahnaz for their invaluable support and guidance throughout our project on Mess Feedback Management System. Their input and feedback have been instrumental in making this project a great learning experience for us.

Abstract

In today's fast-paced lifestyle, maintaining a balanced and nutritious diet can be challenging. Mess Feedback System is a web-based application designed to simplify meal planning by providing personalized menu suggestions based on user preferences and voting patterns.

The application utilizes a database of food items categorized into breakfast, lunch, snacks, and dinner options. Users can browse through these options, vote for their favorite items, and select their preferred meals for each day. Additionally, the system recommends the top-rated dishes for each meal category, taking into account user preferences and voting trends.

Mess Feedback System aims to revolutionize meal planning by offering a user-friendly and personalized solution for individuals seeking convenient and nutritious meal options tailored to their tastes and dietary requirements.

CONTENTS

Introduction

1.1 Problem Statement:

Traditional methods of collecting feedback in hostel mess environments, such as suggestion boxes or paper-based surveys, often lack efficiency and fail to provide timely insights for improvement. These methods typically involve physical forms or handwritten notes, which can be cumbersome to manage and prone to errors. Moreover, the turnaround time for processing and acting upon the feedback may be lengthy, leading to delays in addressing issues or implementing improvements.

In a hostel mess setting, where hundreds or even thousands of residents dine regularly, the sheer volume of feedback can be overwhelming to handle manually. Suggestion boxes may fill up quickly, leading to delays in emptying and processing the contents. Additionally, handwritten feedback may be difficult to decipher, resulting in misinterpretation or miscommunication of residents' concerns.

Given these challenges, there is a clear need for a systematic approach to gather, process, and act upon feedback from users in a hostel mess setting.

Project Component Description

2.1 Front-End

It provides a user-friendly interface for residents to view the daily menu, submit feedback, and access other system functionalities.

Key Front-End Features:

- Display of daily menu for breakfast, lunch, snacks, and dinner.
- Feedback submission form for rating service, food quality, and staff behaviour.
- Navigation bar for easy access to different sections of the application.
- Providing a dashboard to view top-rated food items based on user preferences.

Technologies Used: *HTML, CSS, Javascript*

2.2 Back-End

The back end of the system manages data storage, processing, and retrieval. Key components include:

- **Database:** Utilizes *MySQL* for storing user details, feedback data, and menu information.
- **Server-side Scripting:** Implemented in *PHP* to handle data retrieval, processing, and database interactions.
- **Triggers:** *MySQL* triggers are used to update percentage values based on user votes for different food items.
- **User Authentication:** Ensures secure access to the system by authenticating users.

Database Design

3.1 ER Diagram

This ER diagram represents the model of Mess Feedback System. The entityrelationship diagram of Mess Feedback system show all the visual instruments of database tables and relationship between User, User_details, Feedback, Food_list, Breakfast_votes, Lunch_votes, Dinner_votes, Snack_votes and Vote_Percentage. It used structured data and define relationship between structured data groups of online feedback system functionalities. The Relations are Adds, Manages, Views etc.

The Entities involved in the ER diagram are –

(1) User

(2) User_Details

(3) Feedback

(4) Food_list

(5) Breakfast_votes

(6) Lunch_votes

(7) Snacks_votes

(8) Dinner_votes

(9) Vote_Percentage

TABLE DESCRIPTION

(1) User

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Id	Admin's id	INT
(2)	usn	Roll. No.	Varchar(10)
(3)	Password	User's Password	Varchar(50)
(4)	User_name	Name	Varchar(50)

(2) User_detail

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Id	ID	INT
(2)	Student_usn	Roll. No.	Varchar(10)
(3)	name	Name of student	Varchar(50)
(4)	branch	Branch	Varchar(4)
(5)	Sem	Current Semester	INT

(3) Feedback

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Id	ID	INT
(2)	service	Service quality	Varchar(255)
(3)	Staff_feedback	About staff	Varchar(255)

(4) Food_list

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Food_Id	Food's id	Varchar(10)
(2)	Food_name	Food_name	Varchar(255)
(3)	Food_type	Food_type	Varchar(1)

(5) Breakfast_vote

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Food_Id	Food's id	int
(2)	votes	Counting of votes	int

(6) Lunch_vote

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Food_Id	Food’s id	int
(2)	votes	Counting of votes	int

(7) Snacks_vote

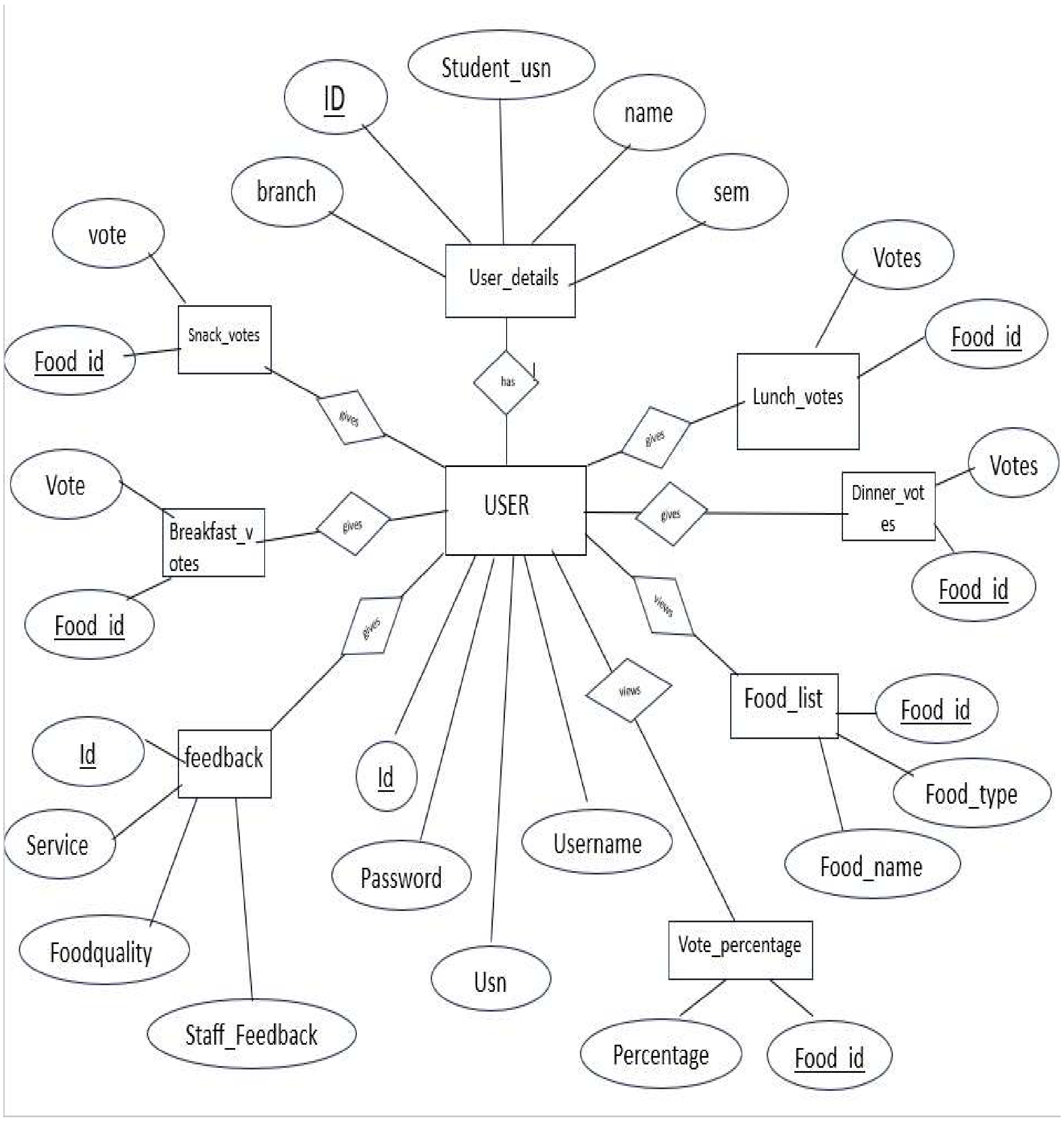
SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Food_Id	Food’s id	int
(2)	votes	Counting of votes	int

(8) Dinner_vote

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Food_Id	Food’s id	int
(2)	votes	Counting of votes	int

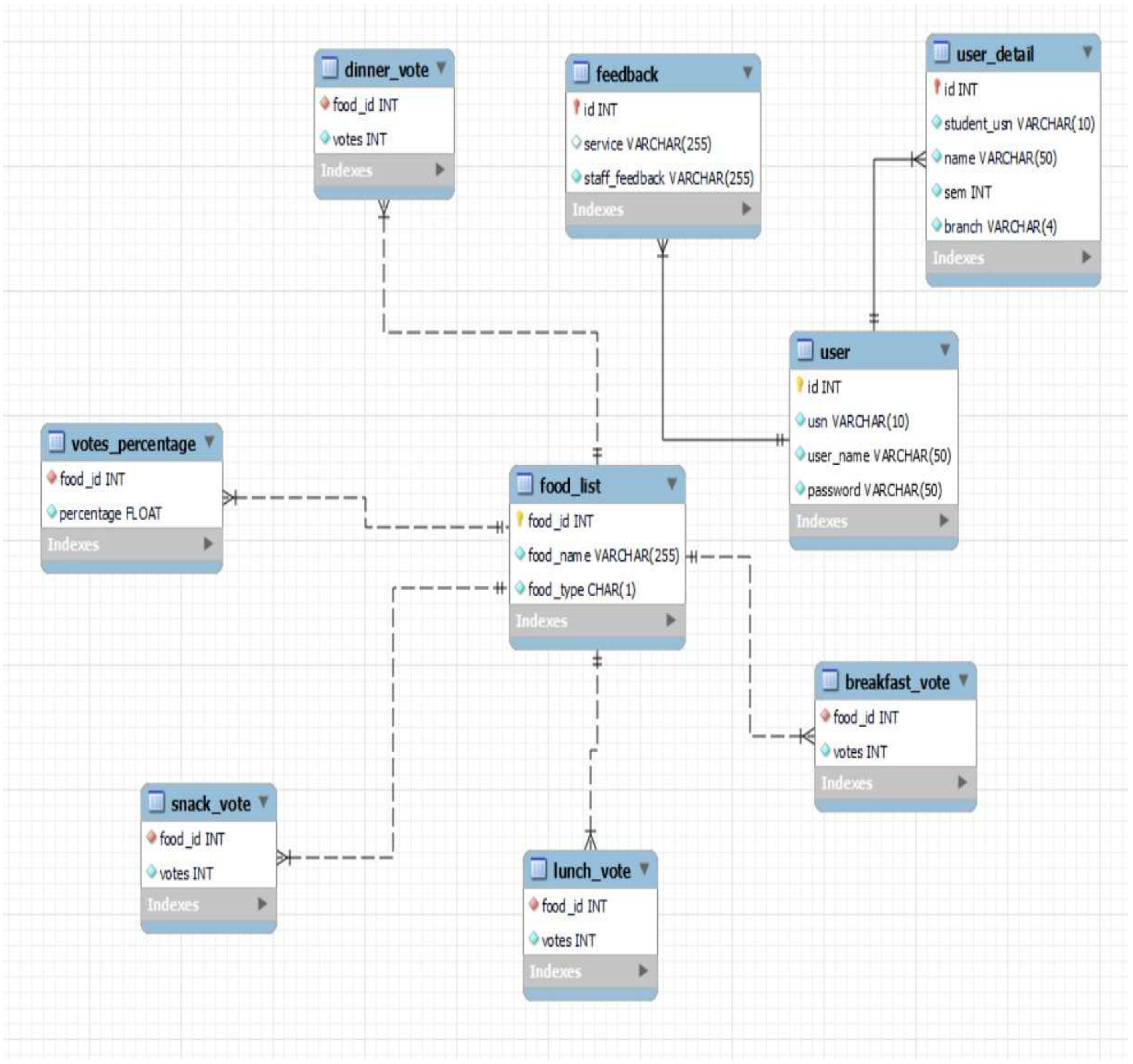
(9) vote_percentage

SR. NO.	ATTRIBUTE NAME	ATTRIBUTE MEANING	ATTRIBUTE VALUE
(1)	Food_Id	Food’s id	int
(2)	percentage	percentage	float



Entity-Relationship diagram for Mess Feedback Management System

3.2 Relational Schema



Relational schematic diagram for Mess Feedback Management System

3.3 Constraints in relation Schema

- Key Constraints:

Relation	**Primary Key**	**Foreign Key**
User	Id	
User_Details	Id	Id
Feedback	Id	Id
Food_list	Food_Id	
Breakfast_votes	Food_Id	Food_Id
Lunch_votes	Food_Id	Food_Id
Snack_votes	Food_Id	Food_Id
Dinner_votes	Food_Id	Food_Id
Votes_percentage	Food_Id	Food_Id

EXPLAINATION:

1. *breakfast_votes:* This table is used to store the number of votes received for each breakfast food item.
2. *dinner_votes:* Similar to breakfast_votes, this table stores the number of votes received for each dinner food item.
3. *feedback*: This table stores feedback from users regarding service, food quality, and staff. It's linked to the user table, indicating which user provided the feedback.
4. *food_list:* This table contains a list of all food items available in the mess, along with their IDs and types (Breakfast, Lunch, Snack, or Dinner).
5. *lunch_votes*: Similar to breakfast_votes, this table stores the number of votes received for each lunch food item.
6. *snacks_votes:* Similar to breakfast_votes, this table stores the number of votes received for each snack food item.
7. *user:* This table stores user details such as ID, username, password, and usn.
8. *user_details*: This table contains additional details about users, such as their name, room number, semester, and branch. It's linked to the user table.
9. *votes_percentage*: This table stores the percentage of votes received for each food item. It's used to calculate and display the top items in each category.

   These tables collectively manage user data, food items, feedback, and voting statistics for the mess system.

Functional Components:

1. *User Registration and Authentication:*
    - Users can register with their username and passwords.
    - Authentication ensures secure access to the system, allowing only registered users to log in.

2. *Feedback Submission:*
    - Users can submit feedback on various aspects such as service quality, food quality, and staff behavior.
    - The feedback form captures user inputs and stores them in the database and shows on the page itself.

3. *Menu Display:*
    - The system displays the daily menu for breakfast, lunch, snacks, and dinner.
    - Users can view the available food items and make informed choices based on their preferences.

4. *Voting System:*
    - Users can vote for their favourite food items in each meal category.
    - Users can select upto seven food items in each menu category for voting.
    - Votes are stored in the database to determine the popularity of different dishes.

5. *Top-Rated Recommendations:*
    - The system generates recommendations based on user votes and preferences.
    - Top-rated three food items for breakfast, lunch, snacks, and dinner respectively along with their percentage of votes are displayed on the dashboard, helping users discover popular choices.

6. *User Profile Management:*
    - Users can manage their profiles, update personal information, and change passwords as needed.
    - Profile management features enhance user experience and ensure data accuracy.

CODE

Database_Project.sql:

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";

-- --------------------------------------------------------

--
-- Table structure for table `breakfast_votes`
--

CREATE TABLE `breakfast_votes` (
  `food_id` int(11) NOT NULL,
  `votes` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `breakfast_votes`
--

INSERT INTO `breakfast_votes` (`food_id`, `votes`) VALUES
(1, 7),
(2, 8),
(3, 4),
(4, 3),
(5, 3),
(6, 2),
(7, 0),
(8, 0),
(9, 1),
(10, 3);

--
-- Triggers `breakfast_votes`
--

DELIMITER $$
CREATE TRIGGER `b_percentage` AFTER UPDATE ON `breakfast_votes` FOR EACH ROW
update votes_percentage set percentage = (SELECT b.votes from food_list a,
breakfast_votes b where b.food_id=a.food_id and a.food_id=NEW.food_id)/(SELECT
sum(votes) from breakfast_votes)*100 where food_id = NEW.food_id
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `dinner_votes`
--

CREATE TABLE `dinner_votes` (
  `food_id` int(11) NOT NULL,
  `votes` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `dinner_votes`
--

INSERT INTO `dinner_votes` (`food_id`, `votes`) VALUES
(31, 5),
(32, 4),
(33, 2),
(34, 1),
(35, 1),
(36, 1),
(37, 0),
(38, 2),
(39, 0),
(40, 0);

--
-- Triggers `dinner_votes`
--

DELIMITER $$
CREATE TRIGGER `D_percentage` AFTER UPDATE ON `dinner_votes` FOR EACH ROW
update votes_percentage set percentage = (SELECT b.votes from food_list a,
dinner_votes b where b.food_id=a.food_id and a.food_id=NEW.food_id)/(SELECT
sum(votes) from dinner_votes)*100 where food_id = NEW.food_id
$$
DELIMITER ;

-- --------------------------------------------------------
```

```
--
-- Table structure for table `feedback`
--

CREATE TABLE `feedback` (
  `id` int(11) NOT NULL,
  `service` varchar(255) DEFAULT NULL,
  `food_quality` varchar(255) DEFAULT NULL,
  `staff_feedback` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `feedback`
--

INSERT INTO `feedback` (`id`, `service`, `food_quality`, `staff_feedback`)
VALUES
(1, 'Good tasty food', 'Service is bad, can be improved', 'Unfriendly staff'),
(2, 'Bad..Dissapointed', 'Hygiene can be improved', 'Unfriendly staff'),
(3, 'Good! But can be improved', 'Bad', 'Unfriendly.');

-- --------------------------------------------------------

--
-- Table structure for table `food_list`
--

CREATE TABLE `food_list` (
  `food_id` int(11) NOT NULL,
  `food_name` varchar(255) NOT NULL,
  `food_type` char(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `food_list`
--

INSERT INTO `food_list` (`food_id`, `food_name`, `food_type`) VALUES
(1, 'Dosa', 'B'),
(2, 'Idli', 'B'),
(3, 'Chole Bhature', 'B'),
(4, 'Poha', 'B'),
(5, 'Aloo Paratha', 'B'),
(6, 'Pav Bhaji', 'B'),
(7, 'Upma', 'B'),
(8, 'Bread Jam', 'B'),
(9, 'Lemon Rice', 'B'),
(10, 'Noodles', 'B'),
(11, 'Soyabean', 'L'),
(12, 'Daal Chawal', 'L'),
(13, 'Rice Sambhar', 'L'),
(14, 'Fried Rice', 'L'),
(15, 'Mixed Veg', 'L'),
(16, 'Kadhi Rice', 'L'),
(17, 'Baingan Bharta', 'L'),
(18, 'Lemon Rice', 'L'),
(19, 'Paneer', 'L'),
(20, 'Chole', 'L'),
(21, 'Oreo', 'S'),
(22, 'Bonda', 'S'),
(23, 'Egg Puff', 'S'),
(24, 'Bhel', 'S'),
(25, 'Cake', 'S'),
(26, 'Sandwich', 'S'),
(27, 'Potato Chips', 'S'),
(28, 'Chai', 'S'),
(29, 'Samosa', 'S'),
(30, 'Spring Roll', 'S'),
(31, 'Biriyani', 'D'),
(32, 'Roti Sabzi', 'D'),
(33, 'Rajma Chawal', 'D'),
(34, 'Chicken Gravy', 'D'),
(35, 'Gobi Munchurian', 'D'),
(36, 'Paneer Bhurji', 'D'),
(37, 'Fried Rice', 'D'),
(38, 'Egg Bhurji', 'D'),
(39, 'Chicken Kebab', 'D'),
(40, 'Mixed Veg', 'D');

-- --------------------------------------------------------

--
-- Table structure for table `lunch_votes`
--

CREATE TABLE `lunch_votes` (
  `food_id` int(11) NOT NULL,
  `votes` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `lunch_votes`
```

```
--

INSERT INTO `lunch_votes` (`food_id`, `votes`) VALUES
(11, 3),
(12, 3),
(13, 0),
(14, 0),
(15, 0),
(16, 0),
(17, 0),
(18, 0),
(19, 1),
(20, 1);

--
-- Triggers `lunch_votes`
--

DELIMITER $$
CREATE TRIGGER `L_percentage` AFTER UPDATE ON `lunch_votes` FOR EACH ROW
update votes_percentage set percentage = (SELECT b.votes from food_list a,
lunch_votes b where b.food_id=a.food_id and a.food_id=NEW.food_id)/(SELECT
sum(votes) from lunch_votes)*100 where food_id = NEW.food_id
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `snacks_votes`
--

CREATE TABLE `snacks_votes` (
  `food_id` int(11) NOT NULL,
  `votes` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `snacks_votes`
--

INSERT INTO `snacks_votes` (`food_id`, `votes`) VALUES
(21, 2),
(22, 2),
(23, 1),
(24, 0),
(25, 0),
(26, 0),
(27, 1),
(28, 0),
(29, 0),
(30, 0);

--
-- Triggers `snacks_votes`
--

DELIMITER $$
CREATE TRIGGER `S_percentage` AFTER UPDATE ON `snacks_votes` FOR EACH ROW
update votes_percentage set percentage = (SELECT b.votes from food_list a,
snacks_votes b where b.food_id=a.food_id and a.food_id=NEW.food_id)/(SELECT
sum(votes) from snacks_votes)*100 where food_id = NEW.food_id
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `user`
--

CREATE TABLE `user` (
  `id` int(11) NOT NULL,
  `usn` varchar(10) NOT NULL,
  `username` varchar(255) NOT NULL,
  `password` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `user`
--

INSERT INTO `user` (`id`, `usn`, `username`, `password`) VALUES
(1, 'IIT2022206', 'Prachi', 'aaaaa'),
(2, 'IIT2022208', 'Ananya', 'aaaaa'),
(3, 'IIT2022117', 'Titiksha', 'aaaaa'),
(4, 'IIT2022214', 'Sanjana', 'aaaaa'),
(6, 'IIT2022120', 'Sneha', 'aaaaa'),
(7, 'IIT2022210', 'Trisha', 'aaaaa');

-- --------------------------------------------------------
```

```
--
-- Table structure for table `user_details`
--

CREATE TABLE `user_details` (
  `id` int(11) NOT NULL,
  `student_usn` varchar(10) NOT NULL,
  `name` varchar(255) NOT NULL,
  `room_no` varchar(4) NOT NULL,
  `sem` int(1) NOT NULL,
  `branch` varchar(4) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `user_details`
--

INSERT INTO `user_details` (`id`, `student_usn`, `name`, `sem`, `branch`)
VALUES
(1, 'IIT2022206', 'Prachi Kumari', 4, 'IT'),
(2, 'IIT2022208', 'Ananya Garg', 4, 'IT'),
(3, 'IIT2022117', 'Titiksha Sharma', 4, 'IT'),
(4, 'IIT2022214', 'Sanjana Prajapati', 4, 'IT'),
(6, 'IIT2022120', 'Sneha Jaiswal', 4, 'IT'),
(7, 'IIT2022210', 'Trisha Bharti', 4, 'IT');

-- --------------------------------------------------------

--
-- Table structure for table `votes_percentage`
--

CREATE TABLE `votes_percentage` (
  `food_id` int(11) NOT NULL,
  `percentage` float NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `votes_percentage`
--

INSERT INTO `votes_percentage` (`food_id`, `percentage`) VALUES
(1, 22.5806),
(2, 25.8065),
(3, 12.9032),
(4, 9.67742),
(5, 9.67742),
(6, 6.45161),
(7, 0),
(8, 0),
(9, 3.22581),
(10, 9.67742),
(11, 37.5),
(12, 37.5),
(13, 0),
(14, 0),
(15, 0),
(16, 0),
(17, 0),
(18, 0),
(19, 12.5),
(20, 12.5),
(21, 33.3333),
(22, 33.3333),
(23, 16.6667),
(24, 0),
(25, 0),
(26, 0),
(27, 16.6667),
(28, 0),
(29, 0),
(30, 0),
(31, 31.25),
(32, 25),
(33, 12.5),
(34, 6.25),
(35, 6.25),
(36, 6.25),
(37, 0),
(38, 12.5),
(39, 0),
(40, 0);


--
-- Indexes for dumped tables
--

--
-- Indexes for table `breakfast_votes`
--
ALTER TABLE `breakfast_votes`
```

```
  ADD PRIMARY KEY (`food_id`);

--
-- Indexes for table `dinner_votes`
--
ALTER TABLE `dinner_votes`
  ADD PRIMARY KEY (`food_id`);

--
-- Indexes for table `feedback`
--
ALTER TABLE `feedback`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `food_list`
--
ALTER TABLE `food_list`
  ADD PRIMARY KEY (`food_id`);

--
-- Indexes for table `lunch_votes`
--
ALTER TABLE `lunch_votes`
  ADD PRIMARY KEY (`food_id`);

--
-- Indexes for table `snacks_votes`
--
ALTER TABLE `snacks_votes`
  ADD PRIMARY KEY (`food_id`);

--
-- Indexes for table `user`
--
ALTER TABLE `user`
  ADD PRIMARY KEY (`id`,`usn`);

--
-- Indexes for table `user_details`
--
ALTER TABLE `user_details`
  ADD PRIMARY KEY (`id`,`student_usn`);

--
-- Indexes for table `votes_percentage`
```

```
--
ALTER TABLE `votes_percentage`
  ADD PRIMARY KEY (`food_id`);


--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `food_list`
--
ALTER TABLE `food_list`
  MODIFY `food_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=41;

--
-- AUTO_INCREMENT for table `user`
--
ALTER TABLE `user`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=8;


--
-- Constraints for dumped tables
--

--
-- Constraints for table `feedback`
--
ALTER TABLE `feedback`
  ADD CONSTRAINT `feedback_ibfk_1` FOREIGN KEY (`id`) REFERENCES `user` (`id`)
ON DELETE CASCADE ON UPDATE CASCADE;

--
-- Constraints for table `user_details`
--
ALTER TABLE `user_details`
  ADD CONSTRAINT `user_details_ibfk_1` FOREIGN KEY (`id`) REFERENCES `user`
(`id`) ON DELETE CASCADE ON UPDATE CASCADE;

COMMIT;
```

Conclusion

The Mess Feedback Management System is a comprehensive solution designed to streamline the feedback collection process and improve the dining experience for hostel residents. By leveraging technology, the system eliminates the shortcomings of traditional feedback methods and provides a user-friendly platform for submitting feedback, viewing menus, and accessing personalized recommendations. With its intuitive interface and robust functionality, the system aims to enhance satisfaction levels among users and facilitate continuous improvement in mess services. Overall, the Mess Feedback Management System represents a significant step towards achieving efficiency, transparency, and accountability in hostel mess operations.