Fashion Show Voting System

By Genevieve Awa

Project Idea

The Fashion Show Voting System allows participants of a fashion show to engage with designers by voting for their favourite designs presented during different show sessions. Each session showcases various designers, and participants can admire and vote for multiple designers. This voting system aims to enhance participant engagement and provide insights into designer popularity across sessions.

Relational Database

Participant Table

- Represents the attendees of the fashion show.
- **Relationships**:
  - Many-to-Many with **Designer**.
  - Many-to-One with **Session** table.

Designer Table

- Represents the designers presenting their work at the fashion show.
- **Relationships**:
  - Many-to-Many with **Participant**.

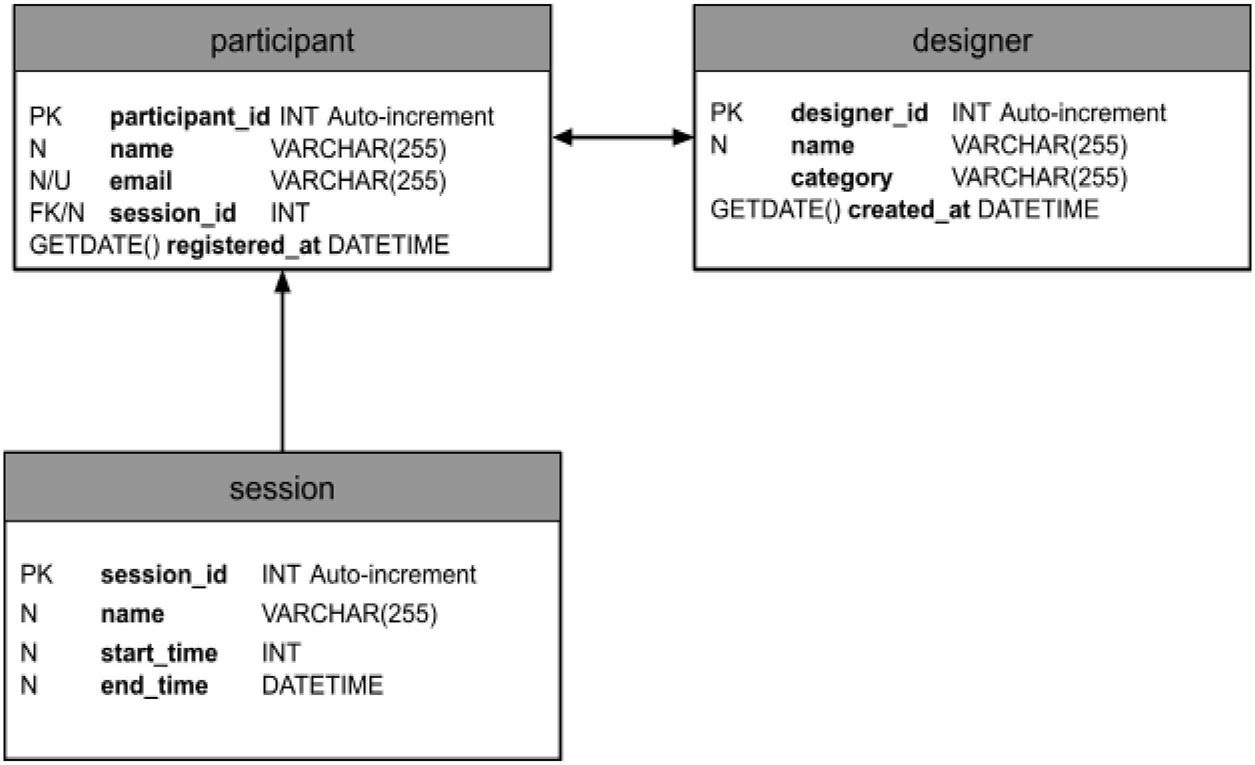
Session Table

- Represents the sessions of the fashion show. Each session corresponds to a part of the event where specific designers are showcased.
- **Relationships**:
  - One-to-Many with **Participant** (a session can have multiple participants, but each participant attends one session).

Relationships Overview

1. **Participant ↔ Designer**:
   - ○ Many-to-Many relationship: A participant can like multiple designers, and a designer can be liked by multiple participants.
2. **Session → Participant**:
   - ○ One-to-Many relationship: A session can have multiple participants, but each participant belongs to only one session.

Entity Relationship Diagram (ERD)



participant

PK	**participant_id**	INT Auto-increment
N	**name**	VARCHAR(255)
N/U	**email**	VARCHAR(255)
FK/N	**session_id**	INT
GETDATE()	**registered_at**	DATETIME

designer

PK	**designer_id**	INT Auto-increment
N	**name**	VARCHAR(255)
	category	VARCHAR(255)
GETDATE()	**created_at**	DATETIME

session

PK	**session_id**	INT Auto-increment
N	**name**	VARCHAR(255)
N	**start_time**	INT
N	**end_time**	DATETIME

Table Descriptions

Participants Table

ParticipantId: A unique identifier for each participant (Primary Key).

Name: The participant’s full name.

Email: The participant’s email address, used for communication and identification.

SessionId: Links the participant to the session they are attending (Foreign Key).

RegisteredAt: The timestamp when the participant registered.

Designers Table

DesignerId: A unique identifier for each designer (Primary Key).

Name: The designer’s full name.

Category: The type of fashion the designer specializes in (e.g., Haute Couture, Casual Wear).

CreatedAt: The timestamp when the designer's profile was added to the system.

Sessions Table

SessionId: A unique identifier for each session (Primary Key).

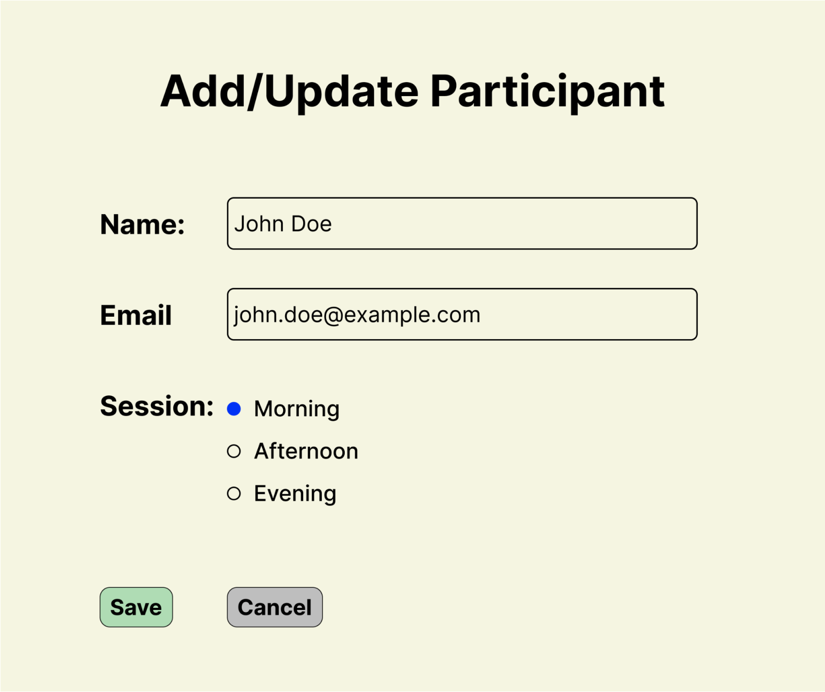
SessionName: The name or title of the session (e.g., "Morning Show").

StartTime: The starting time of the session.

EndTime: The ending time of the session.

Wireframes

Add/Update Participant



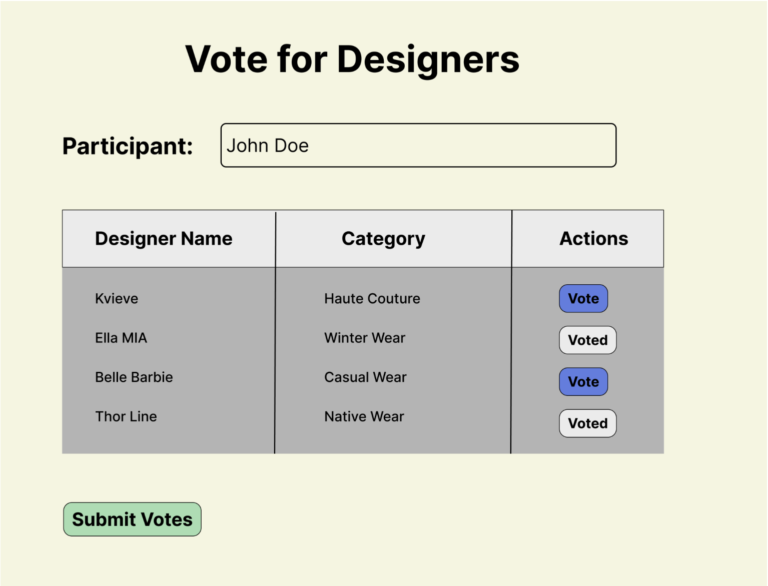
Add/Update Participant

Name: John Doe

Email john.doe@example.com

Session:
● Morning
○ Afternoon
○ Evening

Vote for a Designer

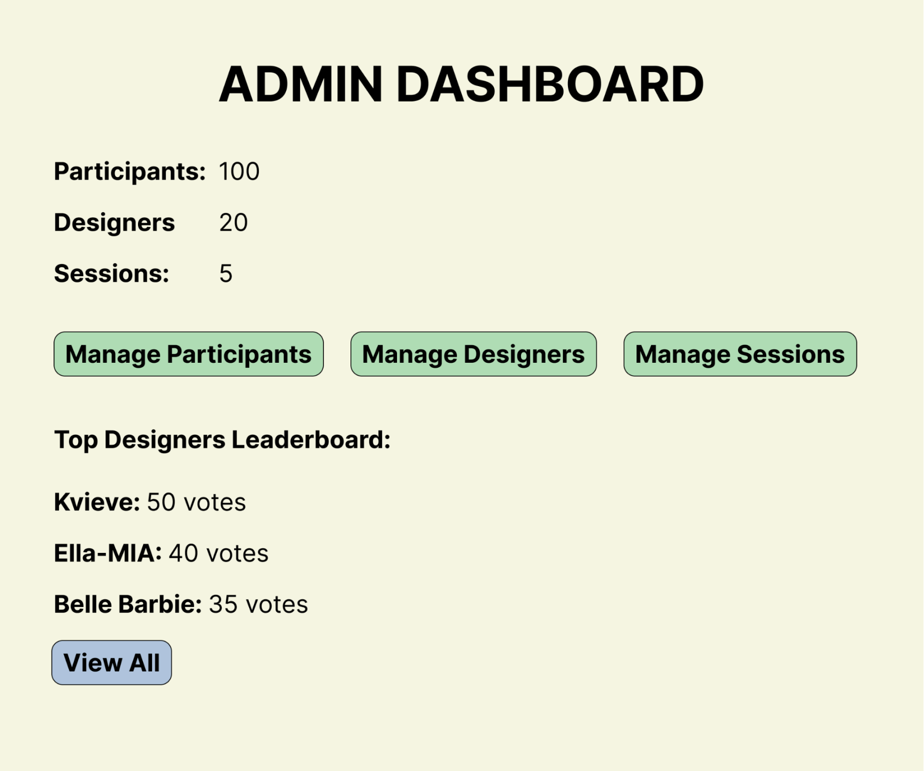


Vote for Designers

Participant: John Doe

Designer Name	Category	Actions
Kvieve	Haute Couture	Vote
Ella MIA	Winter Wear	Voted
Belle Barbie	Casual Wear	Vote
Thor Line	Native Wear	Voted

Admin Dashboard



ADMIN DASHBOARD

Participants: 100

Designers 20

Sessions: 5

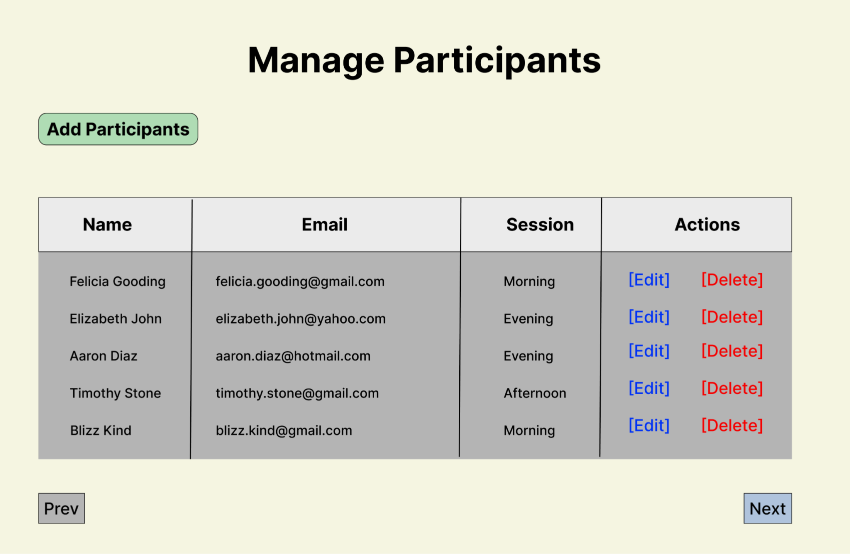
Top Designers Leaderboard:

Kvieve: 50 votes

Ella-MIA: 40 votes

Belle Barbie: 35 votes

Manage Participants



Manage Participants

Name	Email	Session	Actions
Felicia Gooding	felicia.gooding@gmail.com	Morning	[Edit] [Delete]
Elizabeth John	elizabeth.john@yahoo.com	Evening	[Edit] [Delete]
Aaron Diaz	aaron.diaz@hotmail.com	Evening	[Edit] [Delete]
Timothy Stone	timothy.stone@gmail.com	Afternoon	[Edit] [Delete]
Blizz Kind	blizz.kind@gmail.com	Morning	[Edit] [Delete]

API Endpoints (MVP)

Participants API

GET /api/participants: Retrieve a list of all participants.

GET /api/participants/{id}: Retrieve details of a specific participant by their ID.

POST /api/participants: Add a new participant.

PUT /api/participants/{id}: Update an existing participant.

DELETE /api/participants/{id}: Delete a participant by their ID.

Designers API

GET /api/designers: Retrieve a list of all designers.

GET /api/designers/{id}: Retrieve details of a specific designer by their ID.

POST /api/designers: Add a new designer.

PUT /api/designers/{id}: Update an existing designer.

DELETE /api/designers/{id}: Delete a designer by their ID.

Votes API

GET /api/votes: Retrieve all votes cast.

- **Query Parameters** (Optional):
    - ?participantId: Filter votes by a specific participant.
    - ?designerId: Filter votes by a specific designer.

GET /api/votes/{id}: Retrieve a specific vote by its ID.

POST /api/votes: Cast a vote (link a participant and a designer).

DELETE /api/votes/{id}: Remove a specific vote by its ID.

Extra Features

1. **Live Voting Results**

   Display real-time updates of voting results, such as:

   - Top 3 designers in a leaderboard format.
   - Total votes received by each designer.

   Benefit: Creates excitement and encourages participation.