

Design principles

- Everything is scoped by `round_id` so Round 2 can be a new round row (or a new “game session”) without wiping tables.
- Batches are immutable after submit (`status` + `locked_at`).
Market only contains jokes that passed (rating=5).
- Buy/Return uses a transaction to keep budget + purchases + team points consistent.

Tables

Teams

Name	Type	Remarks	Description
id	Int	PK	
name	Text	Unique (not null)	
created_at	Timestamp	Not Null	

Users

Name	Type	Remarks	Description
user_id	Int	PK	
display_name	Text	Unique (not null)	Name of students
role	ENUM	Instructor/JM/QC/Customer	
team_id	Int		

Round_participants

Name	Type	Remarks	Description
round_id	Int	Composite Key	
user_id	Int	Composite Key	
status	ENUM	WAITING, ASSIGNED	

joined_at	Timestamp		
assigned_at	Timestamp		

Rounds

Name	Type	Remarks	Description
round_id	Int	PK	
round_number	Int		
status	ENUM	Configured/Active/Ended	
customer_budget	int		
batch_size	int		
started_at	Timestamp		
ended_at	Timestamp		
created_at	Timestamp		

team_rounds_state

Name	Type	Remarks	Description
round_id	Int	Composite key	
team_id	Int	Composite key	
points_earned	Int	Configured/Active/Ended	
batches_created	Int		
batches Rated	int		
accepted_jokes	Int		
created_at	Timestamp		
updated_at	Timestamp		

jokes

Name	Type	Remarks	Description
joke_id	Int	PK	
batch_id	Int		
joke_text	text		
created_at	Timestamp		

batches

Name	Type	Remarks	Description
batch_id	Int	PK	
round_id	Int		
team_id	Int		
status	ENUM	Draft / Submitted / Rated	
submitted_at	Timestamp		
avg_score	decimal		
passes_count	Int		

joke_ratings

Name	Type	Remarks	Description
joke_id	Int	PK	
qc_user_id	Int		
rating	Int		
rated_at	Timestamp		

published_jokes

Name	Type	Remarks	Description
joke_id	Int	PK	
round_id	Int		

team_id			
---------	--	--	--

customer_round_budget

Name	Type	Remarks	Description
round_id	Int	Composite key	
customer_user_id	Int	Composite key	
starting_budget			
remaining_budget			
created_at	Timestamp		
updated_at	Timestamp		

purchases

Name	Type	Remarks	Description
purchase_id	Int	PK	
round_id	Int		
customer_user_id			
joke_id			

Unique (round_id, customer_user_id, joke_id)

API Schema

Main Menu

POST /v1/session/join

User enters their role and clicks on login

Request	Response
<pre>{ "display_name": "hussain,frank" }</pre>	<pre>{ "user": { "user_id": 12, "display_name": "hussain,frank" }, "round_id": 1, }</pre>

	<pre>"participant": { "status": "WAITING", "joined_at": "2025-12-25T20:10:00Z", "assigned_at": null } }</pre> <p>409 NAME_TAKEN</p>
--	---

POST v1/instructor/login

Request	Response
<pre>{ "display_name": "professor_1", "password": "123" }</pre>	<pre>{ "user": { "user_id": 42, "display_name": "professor_1", "role": "INSTRUCTOR" }, "round_id": 1 }</pre> <p>401 if password is wrong or admin password not configured. 400 on invalid payload.</p>

GET /v1/session/me

FE waiting pop-up can poll and know when to redirect

Response (waiting)	Response (assigned)
<pre>{ "user": { "user_id": 12, "display_name": "Hussain" }, "round_id": 1, "participant": { "status": "WAITING", "joined_at": "...", "assigned_at": null }, "assignment": { "role": null, "team_id": null } }</pre>	<pre>{ "user": { "user_id": 12, "display_name": "Hussain" }, "round_id": 1, "participant": { "status": "ASSIGNED", "joined_at": "...", "assigned_at": "... }, "assignment": { "role": "JM", "team_id": 3 } }</pre>

GET /v1/rounds/active

Get current round (useful for refresh/poll)

Request	Response
	<pre>{ "round": { "id": 1, "round_number": 1, "status": "Active", "batch_size": 5, "customer_budget": 10, "started_at": "2025-12-21T23:10:00Z" } }</pre> <p>If none configured</p> <pre>{ "round": null }</pre>

Joke Maker Screen

Team summary (for JM/QC “my team” card)

GET /v1/rounds/{round_id}/teams/{team_id}/summary

Request	Response
	<pre>{ "team": { "id": 1, "name": "Team A" }, "round_id": 1, "rank": 1, "points": 12, "total_sales": 12, "batches_created": 4, "batches_rated": 3, "accepted_jokes": 9, "avg_score_overall": 4.27, "unrated_batches": 1 }</pre>

POST /v1/rounds/{round_id}/batches

Submit a batch of joke

Headers: X-User-Id must be a JM

Request	Response
<pre>{ "team_id": 1, "jokes": [</pre>	<p>Response 200</p> <pre>{ "batch": { "batch_id": 501,</pre>

<pre>"Why did the...", "I told my...", "Parallel lines...", "My boss said...", "I used to..."] }</pre>	<pre>"round_id": 1, "team_id": 1, "status": "SUBMITTED", "submitted_at": "2025-12-21T23:15:10Z", "jokes_count": 5 } }</pre> <p>409 ROUND_NOT_ACTIVE 400 INVALID_BATCH_SIZE (e.g., Round 1 expects exactly 5)</p>
---	--

GET /v1/rounds/{round_id}/teams/{team_id}/batches

List my submitted batches

Request	Response
	<pre>{ "batches": [{ "batch_id": 501, "status": "SUBMITTED", "submitted_at": "2025-12-21T23:15:10Z", "avg_score": null, "passes_count": null }, { "batch_id": 499, "status": "RATED", "submitted_at": "2025-12-21T23:01:10Z", "rated_at": "2025-12-21T23:05:10Z", "avg_score": 4.2, "passes_count": 2 }] }</pre>

Quality Control APIs

GET /v1/qc/queue/next?round_id=1

Fetch the earliest unrated batch assigned to me

Headers: **X-User-Id** must be QC.

Request	Response
	<pre>{ "batch": { "batch_id": 501, "round_id": 1, "team_id": 1, "submitted_at": "2025-12-21T23:15:10Z" }, "jokes": [{ "joke_id": 9101, "joke_text": "..." }, { "joke_id": 9102, "joke_text": "..." }, { "joke_id": 9103, "joke_text": "..." }, { "joke_id": 9104, "joke_text": "..." }, { "joke_id": 9105, "joke_text": "..." }], "queue_size": 3 }</pre>

POST /v1/qc/batches/{batch_id}/ratings

Submit ratings for a batch

Request	Response
<pre>{ "ratings": [{ "joke_id": 9101, "rating": 5 }, { "joke_id": 9102, "rating":</pre>	<pre>{ "batch": { "batch_id": 501, "status": "RATED", "rated_at":</pre>

<pre> 4 }, { "joke_id": 9103, "rating": 5 }, { "joke_id": 9104, "rating": 3 }, { "joke_id": 9105, "rating": 4 }] } </pre>	<pre> "2025-12-21T23:20:10Z", "avg_score": 4.2, "passes_count": 2 }, "published": { "count": 2, "joke_ids": [9101, 9103] } } </pre> <p> 409 ROUND_NOT_ACTIVE 409 BATCH_ALREADY_RATED 403 NOT_ASSIGNED_TO_THIS_QC </p>
--	---

GET /v1/qc/queue/count?round_id=1

QC queue size only

Request	Response
	{ "queue_size": 3 }

Customer APIs

GET /v1/rounds/{round_id}/market

Request	Response
	<pre> { "items": [{ "joke_id": 9103, "joke_text": "Parallel lines have so much in common...", "team": { "id": 1, "name": "Team A" }, "is_bought_by_me": true }, { "joke_id": 9101, "joke_text": "Why did the..." </pre>

	<pre> "team": { "id": 1, "name": "Team A" }, "is_bought_by_me": false }] }</pre>
--	--

GET /v1/rounds/{round_id}/customers/budget

Get customer's budget

Headers: **X-User-Id** must be customer.

Request	Response
	<pre>{ "round_id": 1, "starting_budget": 10, "remaining_budget": 7 }</pre>

POST /v1/rounds/{round_id}/market/{joke_id}/buy

Buy a joke

Request	Response
	<pre>{ "purchase": { "purchase_id": 8001, "joke_id": 9101 }, "budget": { "starting_budget": 10, "remaining_budget": 6 }, "team_points_awarded": { "team_id": 1, "points_delta": 1 } }</pre> <p>409 ROUND_NOT_ACTIVE 409 INSUFFICIENT_BUDGET 409 ALREADY_BOUGHT</p>

POST /v1/rounds/{round_id}/market/{joke_id}/return

Return a joke

Request	Response
	<pre>{</pre>

	<pre> "purchase": { "purchase_id": 8001, "joke_id": 30001 }, "budget": { "starting_budget": 10, "remaining_budget": 7 }, "team_points_awarded": { "team_id": 1, "points_delta": -1 } } </pre> <p>409 ROUND_NOT_ACTIVE 409 NOT_BOUGHT_YET 409 ALREADY_RETURNED</p>
--	---

Instructor APIs

GET /v1/instructor/rounds/{round_id}/lobby

Request	Response
	<pre> { "round_id": 1, "summary": { "waiting": 2, "assigned": 10, "dropped": 0, "team_count": 3, "customer_count": 2 }, "teams": [</pre>

	<pre> { "team": { "id": 1, "name": "Team 1" }, "members": [{ "user_id": 101, "display_name": "casd", "role": "JM" }, { "user_id": 102, "display_name": "bar", "role": "QC" }] }], "customers": [{ "user_id": 201, "display_name": "vs", "role": "CUSTOMER" }], "unassigned": [{ "user_id": 301, "display_name": "late", "status": "WAITING" }] } </pre>
--	---

POST /v1/instructor/rounds/{round_id}/assign

Request	Response
<pre> { "customer_count": 6, "team_count": 3 } </pre>	<pre> { </pre>

```
}
```

```
"round_id": 1,  
  
"summary": {  
  "waiting": 2,  
  "assigned": 10,  
  "team_count": 3,  
  "customer_count": 2  
},  
  
"teams": [  
  {  
    "team": { "id": 1, "name":  
"Team 1" },  
    "members": [  
      { "user_id": 101,  
"display_name": "casd", "role":  
"JM" },  
      { "user_id": 102,  
"display_name": "bar", "role":  
"QC" }  
    ]  
  }  
],  
  
"customers": [  
  { "user_id": 201,  
"display_name": "vs", "role":  
"CUSTOMER" }  
],  
  
"unassigned": [  
  { "user_id": 301,  
"display_name": "late", "status":
```

	<pre>"WAITING" }] }</pre>
--	--

PATCH /v1/instructor/rounds/{round_id}/users/{user_id}

Request	Response
<pre>{"status": "ASSIGNED", "role": "JM", "team_id": 2}</pre> <p>If moving to unassigned. Just send status as "WAITING"</p>	<pre>{ "round_id": 1, "summary": { "waiting": 2, "assigned": 10, "dropped": 0, "team_count": 3, "customer_count": 2 }, "teams": [{ "team": { "id": 1, "name": "Team 1" }, "members": [{ "user_id": 101, "display_name": "casd", "role": "JM" }, { "user_id": 102, "display_name": "bar", "role":</pre>

	<pre> "QC" }] }], "customers": [{ "user_id": 201, "display_name": "vs", "role": "CUSTOMER" }], "unassigned": [{ "user_id": 301, "display_name": "late", "status": "WAITING" }] } </pre>
--	---

POST /v1/instructor/rounds/{round_id}/start

Start Round

Request	Response
<pre> { "customer_budget": 10, "batch_size": 5 } </pre>	<pre> { "round": { "id": 1, "round_number": 1, "status": "Active", "started_at": </pre>

	<pre>"2025-12-21T23:10:00Z" } }</pre>
--	---

POST /v1/instructor/rounds/{round_id}/end

End Round

Request	Response
	<pre>{ "round": { "id": 1, "round_number": 1, "status": "Ended", "ended_at": "2025-12-21T23:59:59Z" } }</pre>

GET /v1/instructor/rounds/{round_id}/stats

Instructor live stats

Request	Response
	<pre>{ "round_id": 1, "teams": [</pre>

	<pre>{ "rank": 1, "team": { "id": 1, "name": "Team A" }, "points": 12, "total_sales": 12, "batches Rated": 3, "avg_score_overall": 4.27, "accepted_jokes": 9 }]</pre>
--	--