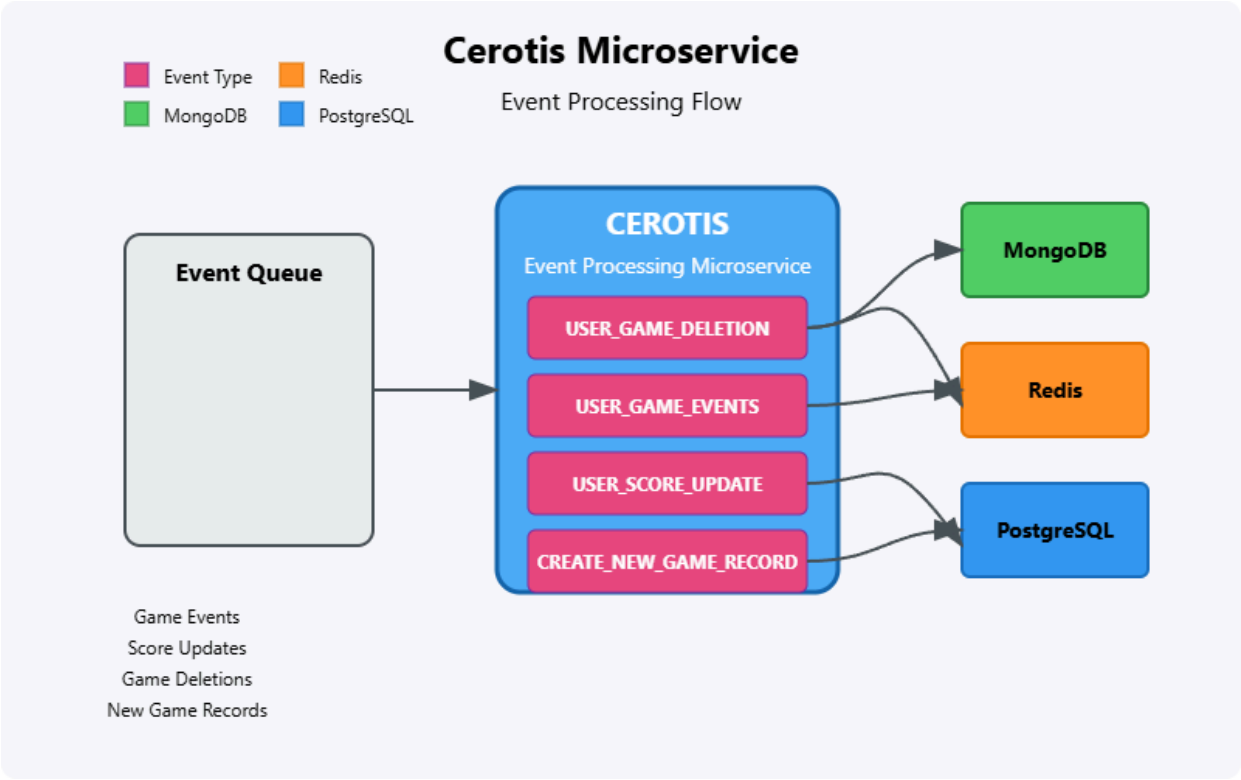
Brief:

[(Github Repo)]

Cerotis is the microservice responsible for consuming and do the necessary processing for 4 events:



Events Description:

USER_GAME_EVENTS:

This event is generated by **vortex-pub-sub** and represents a move made by a user. The event is parsed on **cerotis**, and the state of the chessboard is updated accordingly.

When a game initializes, the default state of the chessboard is stored in **Redis** under the key **"ChessState_${game_id}"** in **FEN string format**. Each time a `USER_GAME_EVENTS` event is

consumed, the game state corresponding to the provided `game_id` is updated based on the event details.

The event typically consists of:

- `game_id`
- `player_id`
- `player_move`

Once updated, the new game state is saved back in **Redis cache**, allowing spectators to view the current state when they join later. This eliminates the need to maintain a full history of moves, as spectators can simply receive the latest board state instead of replaying all previous moves.

USER_SCORE_UPDATE:

This event contains:

- `user_id`
- `score_value` (which is added to the user's current score)

If the `score_value` is negative, the score decreases, but a user's score never falls below **zero**. Additionally, users who disconnect mid-game are penalized with a **15-point score reduction**.

USER_GAME_DELETION:

Once a game ends or becomes invalid (e.g., if a player disconnects abruptly), temporary game-related records are deleted. These records are stored in **MongoDB** and **Redis** and serve other APIs by providing game details. Deleting them ensures system consistency.

CREATE_NEW_GAME_RECORD:

For **staked games**, a record is created in **PostgreSQL** to track game-related details for other services. This record includes:

- Whether the game was **valid** or **invalid**
- The **winner** (if the game was valid)

Based on these details, bets are generated accordingly.