Chrono Quest

By:
Anika, Ghazanfar, Lakindu, Nirmani

Agenda

01 Vision and purpose of the game

02 Why this idea

03 Database solutions

04 Coding solutions

05 Demo version of the game

06 Further development ideas

07 Conclusion and Q & A

Goal of the game

- Collect 5 Chrono Shards from different time eras
- Travel between European airports as time portals.
- Choose destinations wisely to avoid getting stranded.
- Return to your starting airport with all shards.
- Fix the time machine and restore the world.

Why this idea?

- Epic adventure designed for fans of sci-fi and time travel.
- Combines mystery, exploration, and survival in one story.
- To give players an experience they could never have in real life.
- To give players a hero’s journey across time

Database Solutions

- Biggest problem was to add timelines to airports
- Created a new column named era in airport table
- Used case statement to assign random eras to airports
- So, timelines randomly distributed among airports

```
MariaDB [test1]> ALTER TABLE airport
    -> ADD COLUMN era VARCHAR(20)
    -> DEFAULT 'MODERN';
```

A MariaDB SQL command to add column in a table

```
MariaDB [test1]> UPDATE airport
    -> SET era = CASE
    ->     WHEN RAND() < 0.25 THEN 'ANCIENT'
    ->     WHEN RAND() < 0.33 THEN 'MEDIEVAL'
    ->     WHEN RAND() < 0.5 THEN 'FUTURE'
    ->     ELSE 'MODERN'
    -> END
    -> WHERE continent = 'EU' AND type = 'large_airport';
```

A MariaDB SQL command to update

Coding Solutions

- We wanted aircrafts to spend different energy levels in different timelines
- Created a dictionary named eras
- Assigned each era with different values

```
# Era modifiers
ERAS = {
    'ANCIENT': 1.2,
    'MEDIEVAL': 1.1,
    'MODERN': 1.0,
    'FUTURE': 0.9
}
```

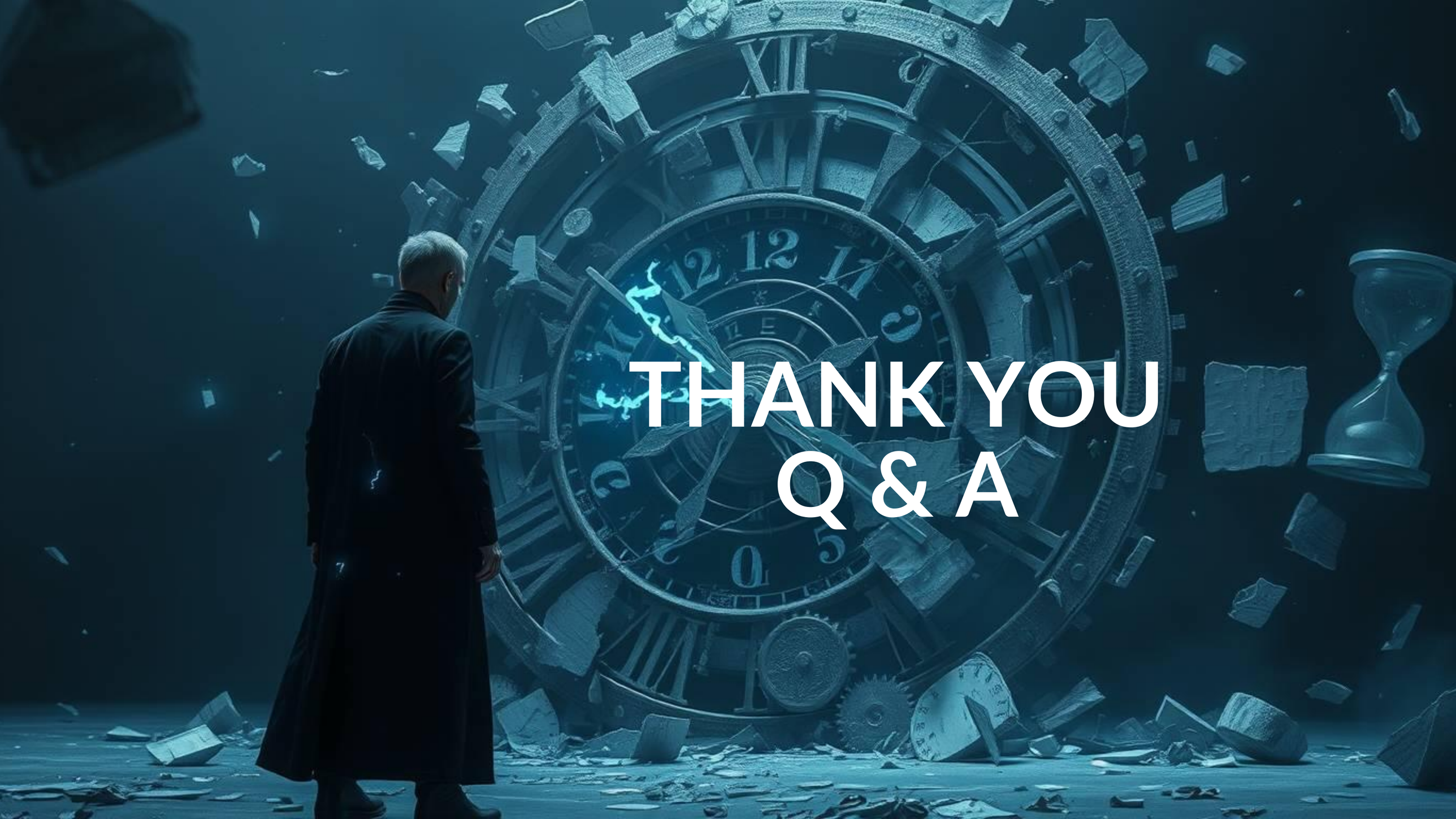
- So, calculating distance is multiplied by key values of eras
- Used pyfiglet for bigger fonts

Future development ideas

- Add a **“missions” table** to store time-based quests and objectives linked to each era.
- Introduce **player progress tracking** using tables for inventory, energy levels, and collected shards.
- Implement **data relationships** (foreign keys) between airports, eras, and player data for consistency.
- Add **event logging tables** to record travel history and in-game decisions.

Conclusion

- Chrono Quest demonstrates how database logic and coding can power creative gameplay.
- Using **MariaDB**, we successfully connected time-era data with in-game travel mechanics.
- Our design highlights teamwork, problem-solving, and real database integration.
- The system can now expand easily with new eras, missions, and features.


THANK YOU
Q & A