Matthew Price
Selim Ishakbeyoglu
Jonathan Willits

Final Deliverable

Project Introduction:

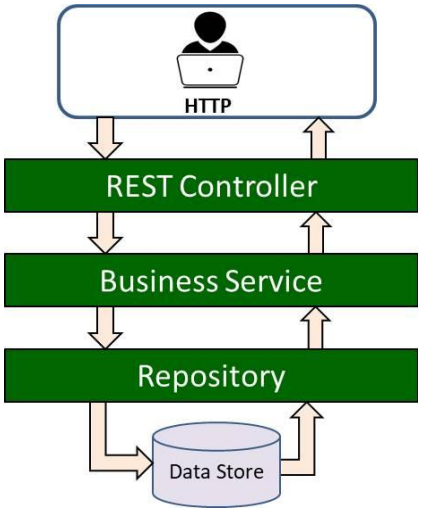
For our project, we decided to create a website that tracks the bowling scores and overall games of each person. Users are able to create accounts with their email, which will allow them to see a summary of the games they played, as well as allowing them to enter their own games which can show up on a leaderboard. The leaderboard will show the top scores recorded on the website, so everyone can see the achievements of others. In order to create this website, we utilized Spring Boot to manage the dependencies used to create a backend capable of working with H2 Database. For the frontend, we used React Bootstrap in order to construct an interactive website.

Objective:

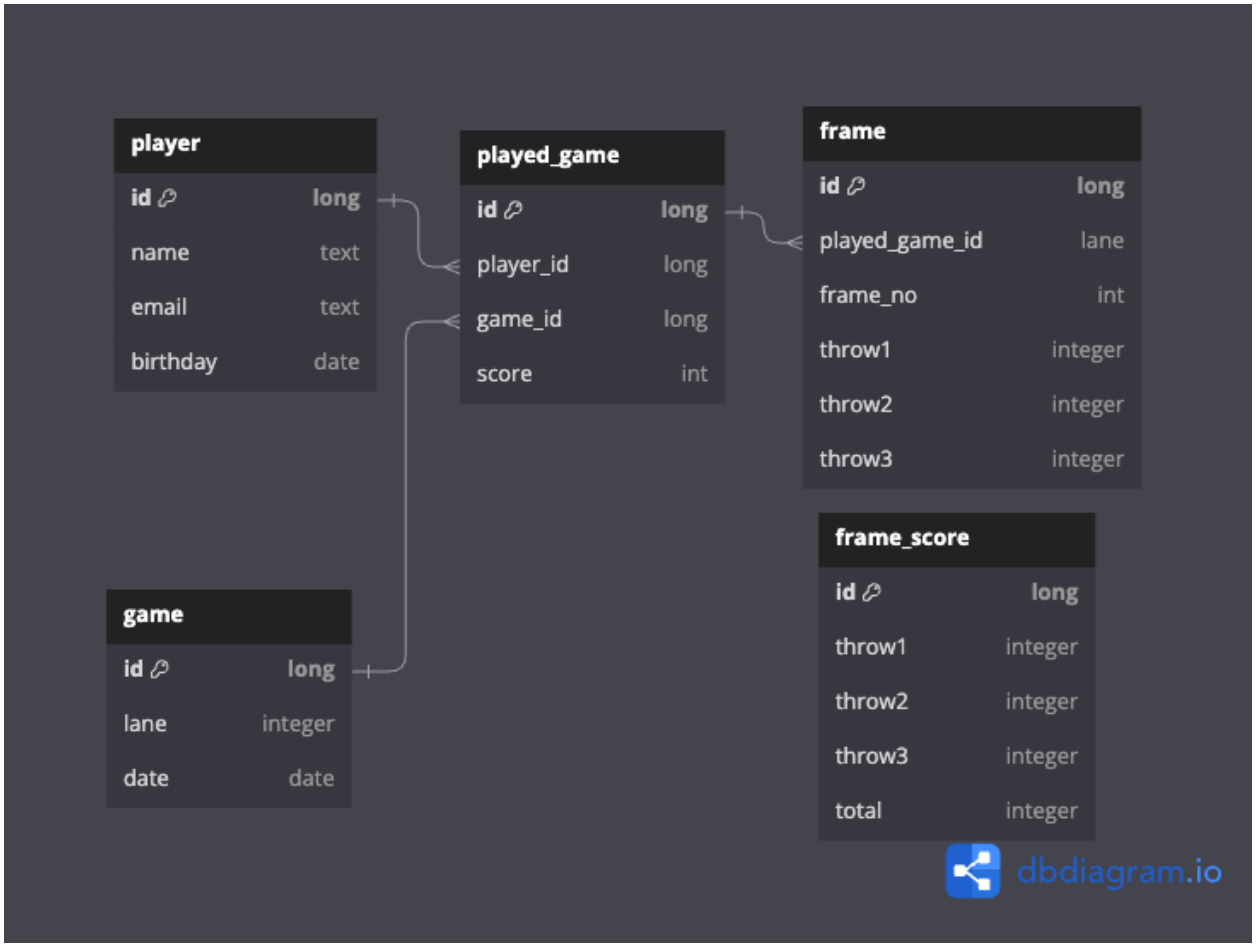
The main objective of this project was to gain a better understanding of databases and how they interact with applications such as a website. In order to achieve this objective, we constructed both the frontend and backend of a website. The backend, although complicated, taught us much about how applications connect to databases and how they manipulate them. The frontend taught us how websites connect to databases through the backend, since they aren't capable of safely accessing them alone.

Project High-Level Design:

The backend of the project is a standard Java Spring Boot REST application. There are 5 entities that are the database tables; this is in the "Model" package. Each entity has a repository, service, and controller. When the frontend needs anything from the database, it sends an HTTP request via axios and the controller layer picks it up. The controller layer calls the corresponding service layer method which calls the required repository. The repository layer is the connection to the database; calls to it are converted to SQL statements. The service layer can then also do any calculations or logic it requires before returning the statement results to the controller layer. The controller layer returns an HTTP response.



Database Design:



Normalization of Tables:

Frame_score is 3NF because throw1, throw2, throw3 → total. We were planning to make the primary key {throw1, throw2, throw3}, but having a multiple column primary key seemed too hard to do in Spring Data JPA.

The rest of the tables are BCNF because they don't have any functional dependencies other than the primary key.

The following are alternate keys not shown on the diagram:

- {player_id, game_id} in played_game
- {played_game_id, frame_no} in frame (frame_no should only be 1-10)
- {throw1, throw2, throw3} in frame_score

Results:

Complete:

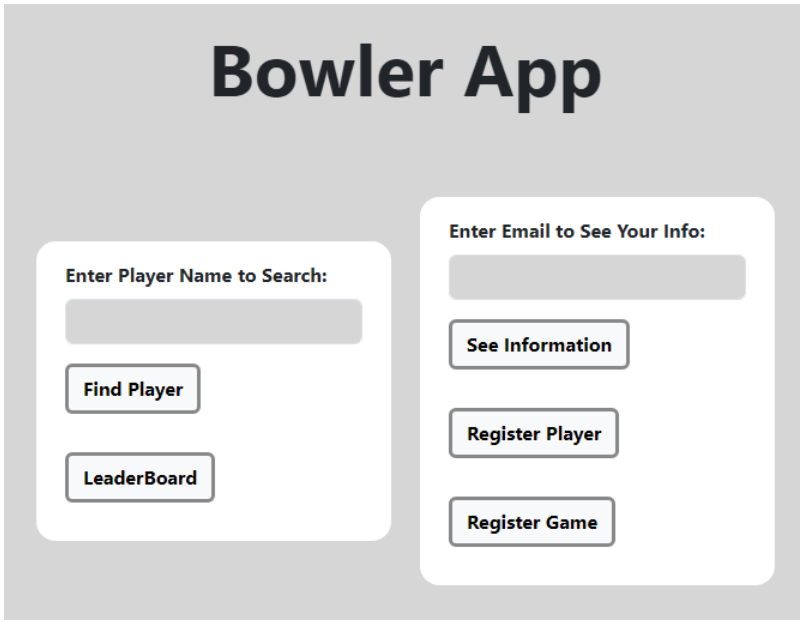
- Register player, look up player
- Register game with one player
- Leaderboard
- Create and read operations in the backend

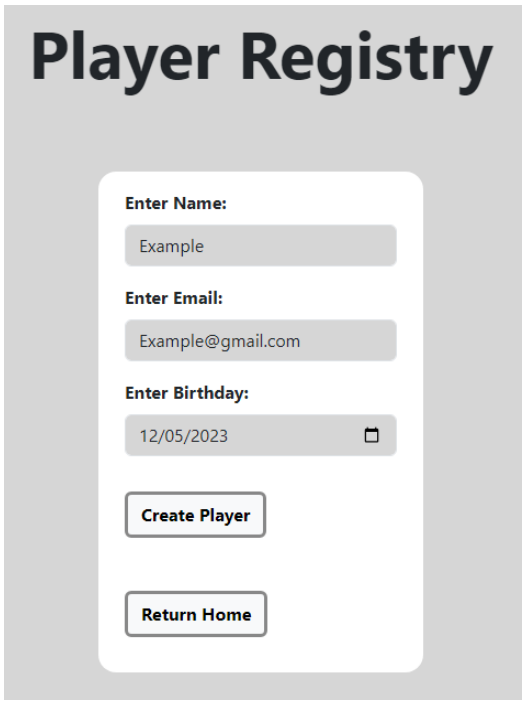
To do:

- Register a game with multiple players
- Validate inputs
- Calculate score from frames instead of having user input it
- Minor change to database design

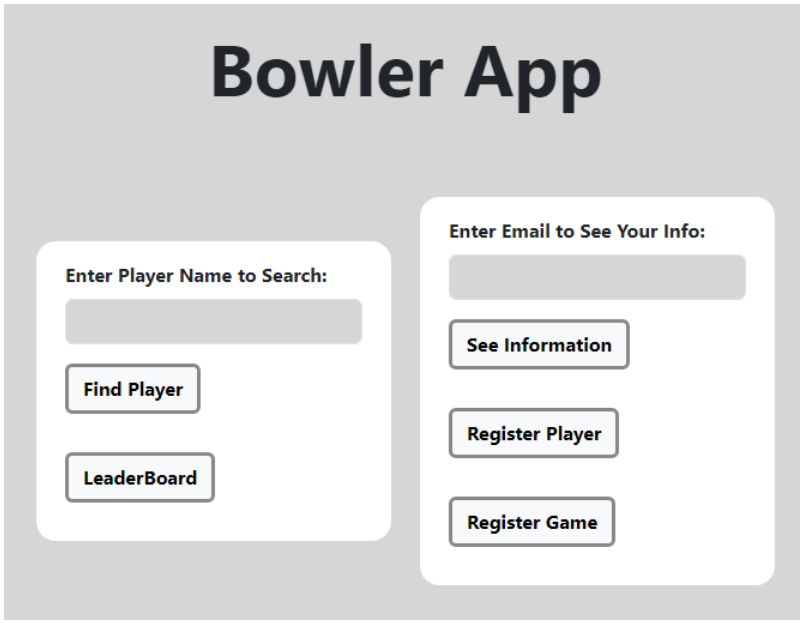
Screenshots:

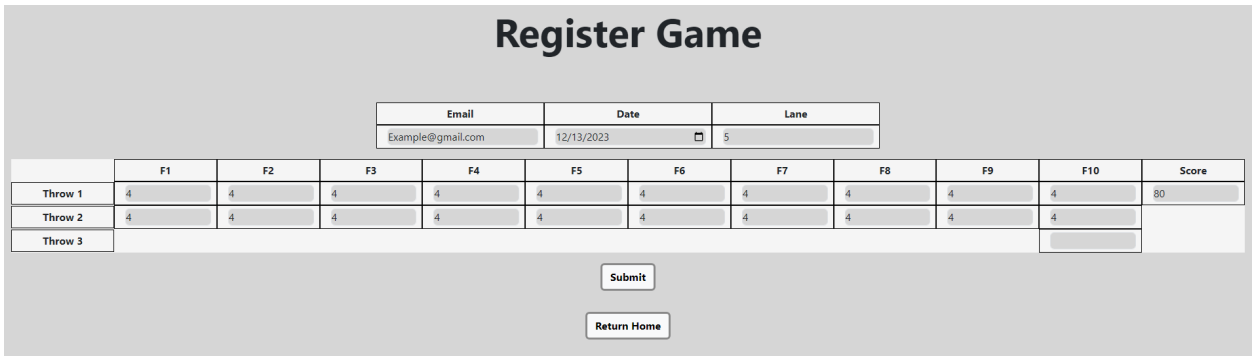
Register Player:



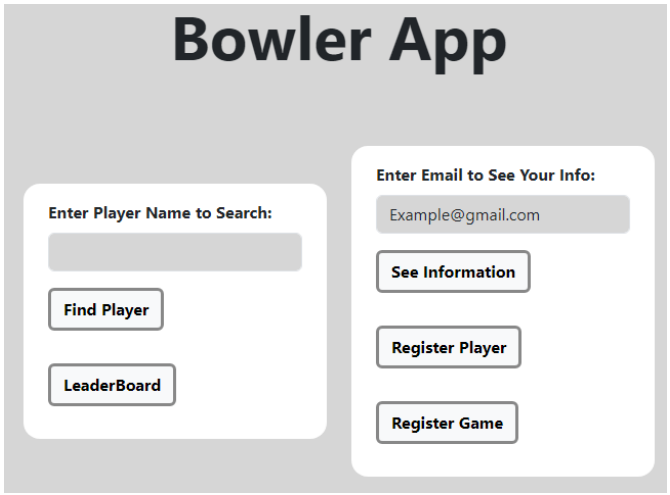


Register Game:

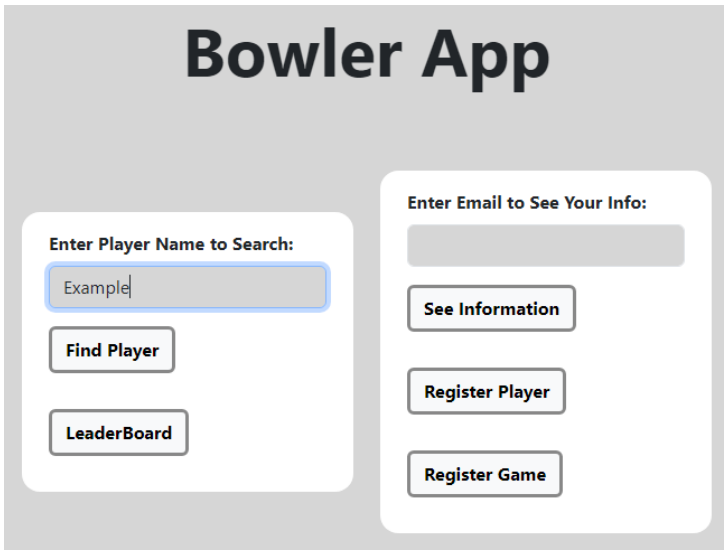


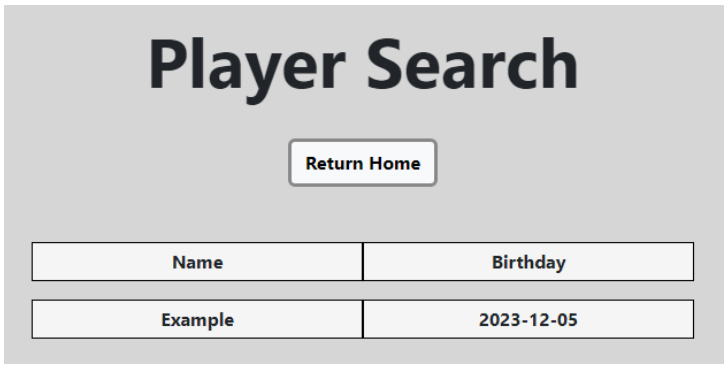


See Information:

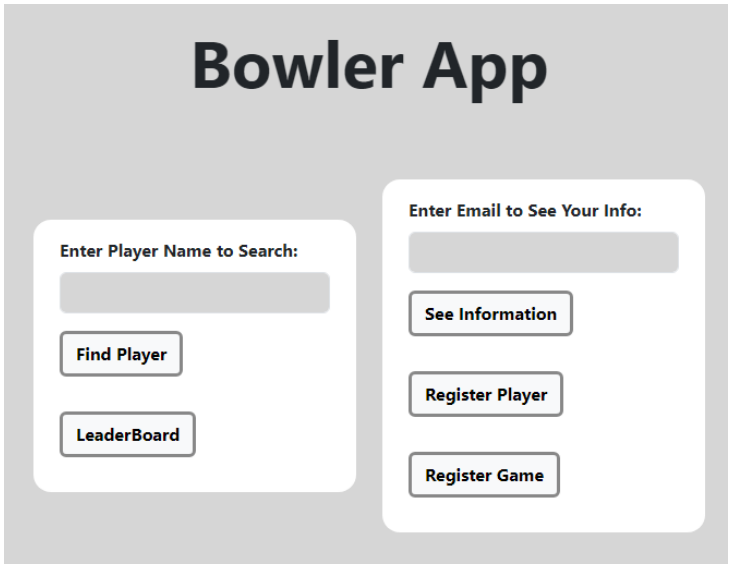


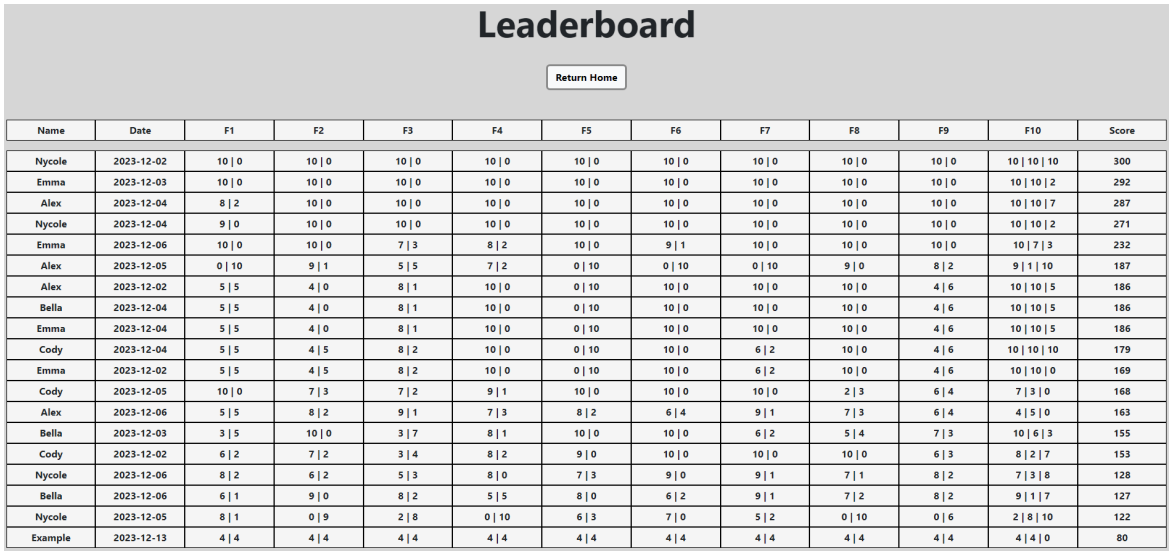
Find Player:





Leaderboard:





Leaderboard

Return Home

Name	Date	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	Score
Nycole	2023-12-02	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 10 | 10	300
Emma	2023-12-03	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 10 | 2	292
Alex	2023-12-04	8 | 2	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 10 | 7	287
Nycole	2023-12-04	9 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 0	10 | 10 | 2	271
Emma	2023-12-06	10 | 0	10 | 0	7 | 3	8 | 2	10 | 0	9 | 1	10 | 0	10 | 0	10 | 0	10 | 7 | 3	232
Alex	2023-12-05	0 | 10	9 | 1	5 | 5	7 | 2	0 | 10	0 | 10	0 | 10	9 | 0	8 | 2	9 | 1 | 10	187
Alex	2023-12-02	5 | 5	4 | 0	8 | 1	10 | 0	0 | 10	10 | 0	10 | 0	10 | 0	4 | 6	10 | 10 | 5	186
Bella	2023-12-04	5 | 5	4 | 0	8 | 1	10 | 0	0 | 10	10 | 0	10 | 0	10 | 0	4 | 6	10 | 10 | 5	186
Emma	2023-12-04	5 | 5	4 | 0	8 | 1	10 | 0	0 | 10	10 | 0	10 | 0	10 | 0	4 | 6	10 | 10 | 5	186
Cody	2023-12-04	5 | 5	4 | 5	8 | 2	10 | 0	0 | 10	10 | 0	6 | 2	10 | 0	4 | 6	10 | 10 | 10	179
Emma	2023-12-02	5 | 5	4 | 5	8 | 2	10 | 0	0 | 10	10 | 0	6 | 2	10 | 0	4 | 6	10 | 10 | 0	169
Cody	2023-12-05	10 | 0	7 | 3	7 | 2	9 | 1	10 | 0	10 | 0	10 | 0	2 | 3	6 | 4	7 | 3 | 0	168
Alex	2023-12-06	5 | 5	8 | 2	9 | 1	7 | 3	8 | 2	6 | 4	9 | 1	7 | 3	6 | 4	4 | 5 | 0	163
Bella	2023-12-03	3 | 5	10 | 0	3 | 7	8 | 1	10 | 0	10 | 0	6 | 2	5 | 4	7 | 3	10 | 6 | 3	155
Cody	2023-12-02	6 | 2	7 | 2	3 | 4	8 | 2	9 | 0	10 | 0	10 | 0	10 | 0	6 | 3	8 | 2 | 7	153
Nycole	2023-12-06	8 | 2	6 | 2	5 | 3	8 | 0	7 | 3	9 | 0	9 | 1	7 | 1	8 | 2	7 | 3 | 8	128
Bella	2023-12-06	6 | 1	9 | 0	8 | 2	5 | 5	8 | 0	6 | 2	9 | 1	7 | 2	8 | 2	9 | 1 | 7	127
Nycole	2023-12-05	8 | 1	0 | 9	2 | 8	0 | 10	6 | 3	7 | 0	5 | 2	0 | 10	0 | 6	2 | 8 | 10	122
Example	2023-12-13	4 | 4	4 | 4	4 | 4	4 | 4	4 | 4	4 | 4	4 | 4	4 | 4	4 | 4	4 | 4 | 0	80

Contributions:

Matthew Price:

- Worked on frontend
    - Created every single frontend page
- Connected the frontend to the backend by using axios calls
- Initialized the project using Spring Boot and Node
- Worked on the Project Final Submission
    - Project introduction
    - Objective
    - Screenshots
- Populated the database with example data

Selim Ishakbeyoglu:

- Worked on backend
    - Created the database and connected it to our application
    - Wrote all the models, controllers, services, repositories.
- Worked on final submission
    - Wrote project high-level design, database design, normalizations, result

References:

N/A