Design Process And Testing

Introduction:

Tangerines is a puzzle game that involves flipping cards to reveal hidden numbers and tangerines. The objective of the game is to uncover all the number cards without revealing too many tangerines. In this document, we will explore the design choices made in the creation of Tangerines.

Gameplay:

Tangerines is a game that is easy to pick up but difficult to master. The game board consists of a grid of cards, each of which contains a number or a tangerine. The player must choose a row or column and flip all the cards in that row or column. The numbers on the cards in the selected row or column are added up, and that total is displayed on the top of the game board.

Game Difficulty:

One of the primary design choices of Tangerines is the level of difficulty. The game is designed to be challenging and engaging for players of all skill levels. The game includes multiple difficulty levels, ranging from easy to hard. This allows players to start with simpler puzzles and progress to more challenging ones as they become more skilled.

Gameplay Mechanics:

The mechanics of Tangerines are straightforward and easy to understand, but there is a lot of depth to the gameplay. The game includes a risk vs reward system, where players must balance the risk of flipping over a tangerine with the potential reward of uncovering more numbers. This creates tension and excitement as players make decisions that can either lead to success or failure.

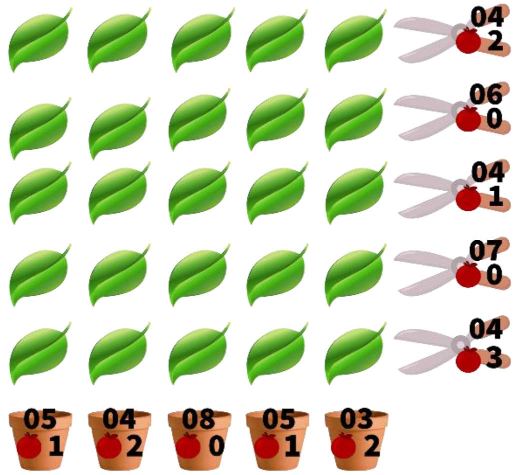
Visual Design:

The visual design of Tangerines is colorful and playful, which helps to make the game more approachable and engaging for players. The game features bright colors and cute graphics, which create a fun and inviting atmosphere.

User Interface:

The user interface of Tangerines is designed to be intuitive and user-friendly. The game board is easy to navigate, and players can quickly understand how to select rows and columns to flip cards. The game also includes a scoring system that is displayed prominently at the top of the screen, which allows players to track their progress and see how they are performing.

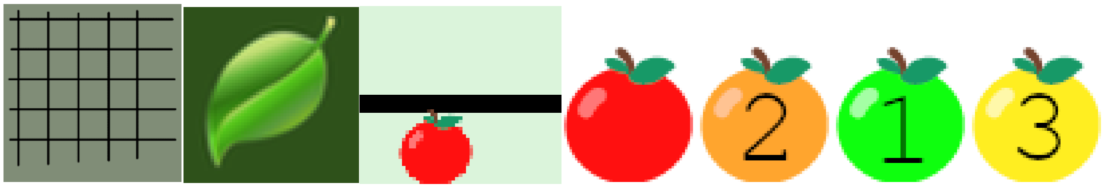
Initial concept design:



This was the initial concept design. We research few JavaScript and html resources to implement this design in our site but due to complexity and time constraints we were unable to implement this design since it would require us to build our own JavaScript library and most of the animations for this design.

Final Design process:

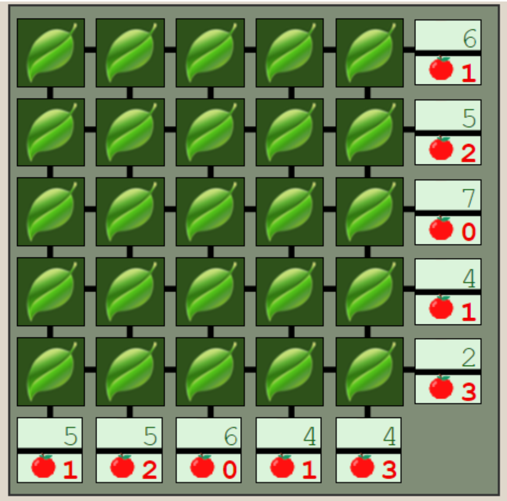
Designing our final game design was much more straightforward than our concept design. We used the HTML table to create our grid. We used the background images on the table and td section to use leaves to cover each cell of the table.



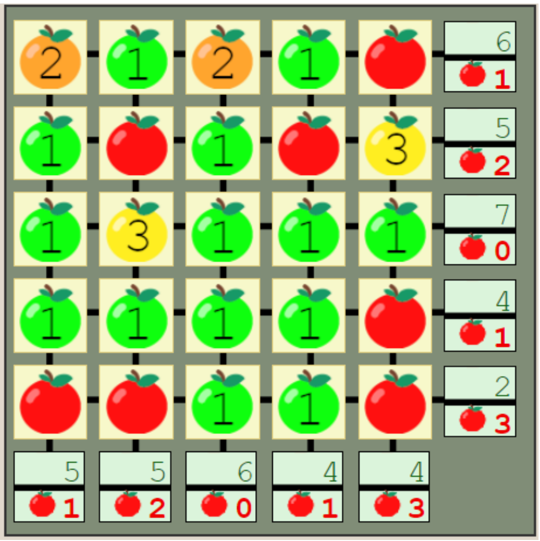
Here are all the building blocks of the game design. The first image is used as background image for the table. All these resources took some time to design since they must fit the size of the table and cells. The leaves with a green background are used to cover cells in the table. The third image used for hints cells on the right and the bottom side of the table. Red, orange, green, and yellow tangerine were used for numbering and red one to indicate user has clicked on cell with bomb. If the user revels bomb (red tangerine) by clicking on a leaf, then game is over.

Putting all these resources together we were able to design the UI of the game. Each functionality of the game is taken care of on the backend with JavaScript code. Populating the table as well as the effect to revel tangerine is taken care on the backend.

Final game user interface Design: This is what the UI looks like when the game is loaded.

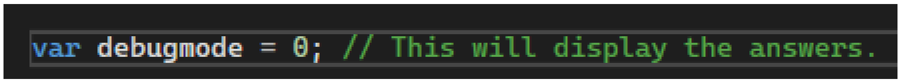


Final game user interface Design: This is the gameplay as you can see each leaf can revel either numbered tangerine or bomb (red tangerine).



Testing

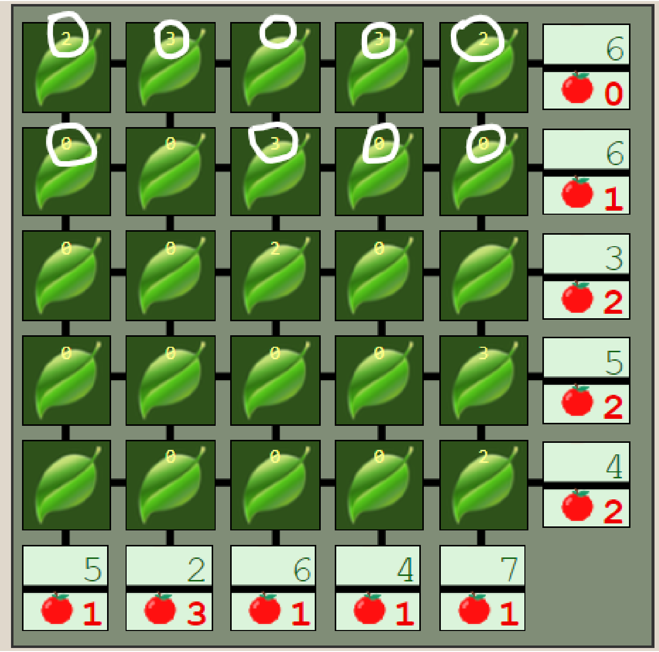
Most of the functionality required manual user testing. Even the API calls were testing using postman. To proper test our design we were able to put cheat code into the source code that would reveal what is hidden behind the leaves.



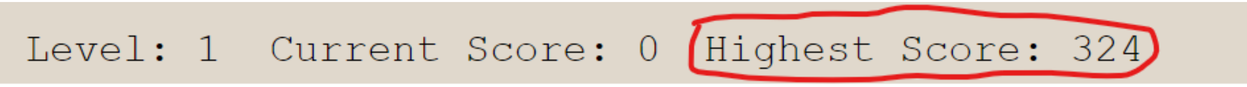
```
var debugmode = 0; // This will display the answers.
```

We called it a debug mode. The production code will have this line of code removed but in development environment debugmode can be turned into 1 and we will be able to all the answer so we can pass through the levels without struggling and test the functionality of the game.

UI when debugging mode is on:



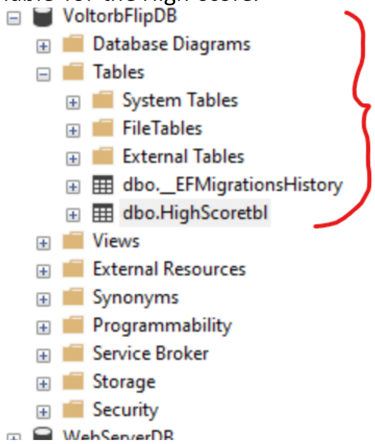
As you can see on this image, I have debugging mode on the backend code. I have circled the part that would let me know what is hidden and would show if I click that cell. 0 refers to red tangerine which is the bomb and ends the game. 1 show as blank which represents the green tangerine with number 1 on it. 2 represents the orange tangerine with number 2 on it. At last, 3 represents the yellow tangerine with number 3 on it.



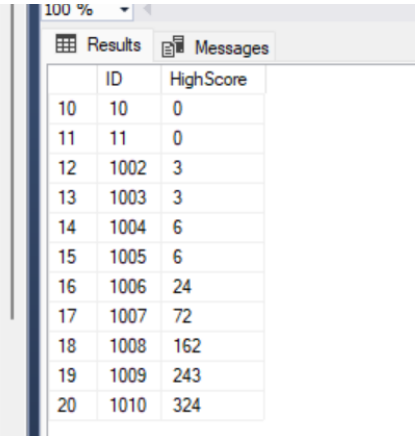
Level: 1 Current Score: 0 Highest Score: 324

This is another functionality that requires testing. This is GET API call that gets the data from DB.

Table for the High score:



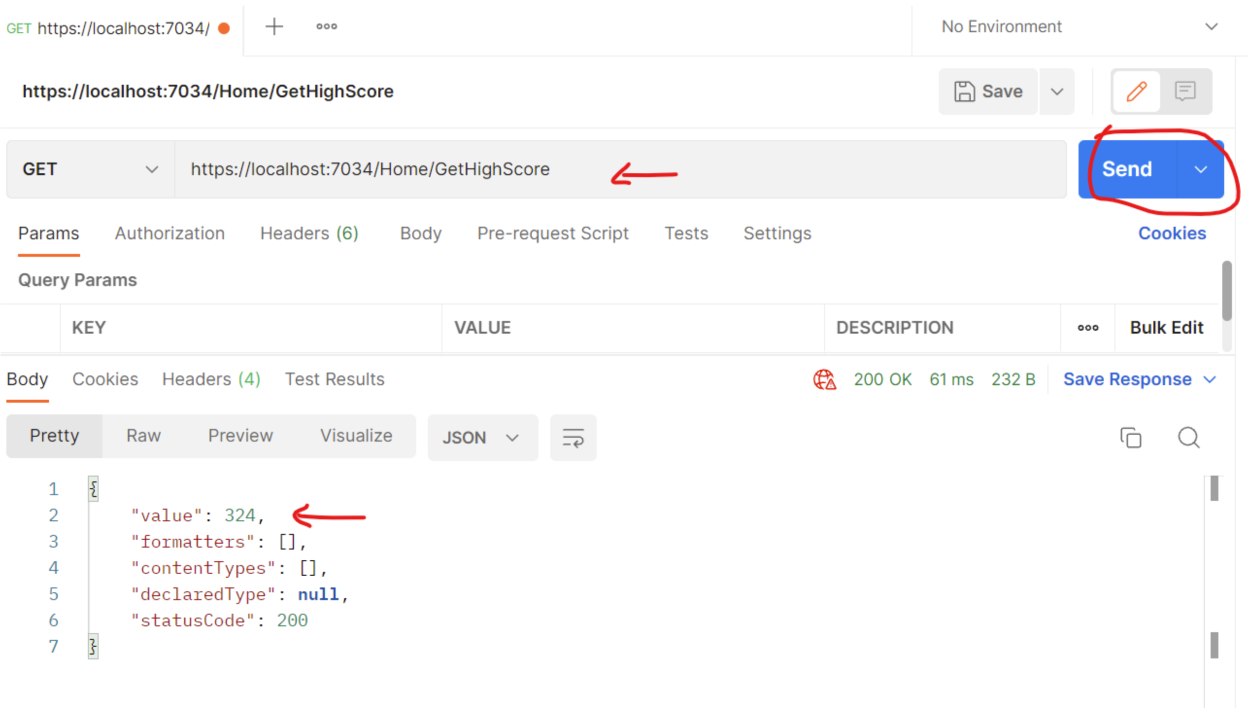
- VoltorbFlipDB
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - dbo.__EFMigrationsHistory
    - dbo.HighScoretbl
  - Views
  - External Resources
  - Synonyms
  - Programmability
  - Service Broker
  - Storage
  - Security
- WebServerDB



	ID	HighScore
10	10	0
11	11	0
12	1002	3
13	1003	3
14	1004	6
15	1005	6
16	1006	24
17	1007	72
18	1008	162
19	1009	243
20	1010	324

Since we are using LINQ query to store our data it has the EF builtin table which keep tracks of the all the migrations which are our table changes. The dbo.HighScoretbl is the actual table that communicates with frontend to get and store the highest score.

GET high score API call test with Postman:



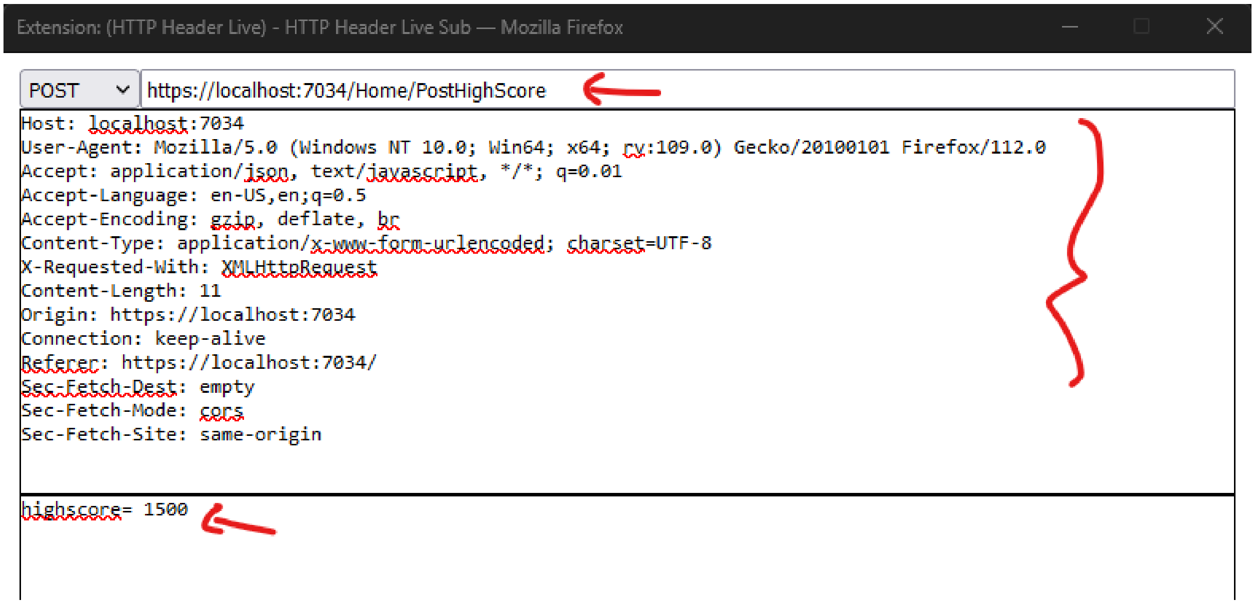
https://localhost:7034/Home/GetHighScore

GET https://localhost:7034/Home/GetHighScore

```
{
    "value": 324,
    "formatters": [],
    "contentTypes": [],
    "declaredType": null,
    "statusCode": 200
}
```

Here you can see to test my API endpoint, I have entered the API endpoint for getting the highest score from DB. When I send the get request it sends me back the body which has the value which is the highest score from my DB.

POST high score API Testing with HTTP Live Header on Firefox:

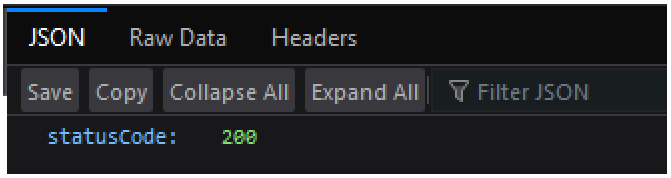

Extension: (HTTP Header Live) - HTTP Header Live Sub — Mozilla Firefox

POST https://localhost:7034/Home/PostHighScore

```
Host: localhost:7034
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/112.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 11
Origin: https://localhost:7034
Connection: keep-alive
Referer: https://localhost:7034/
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
```

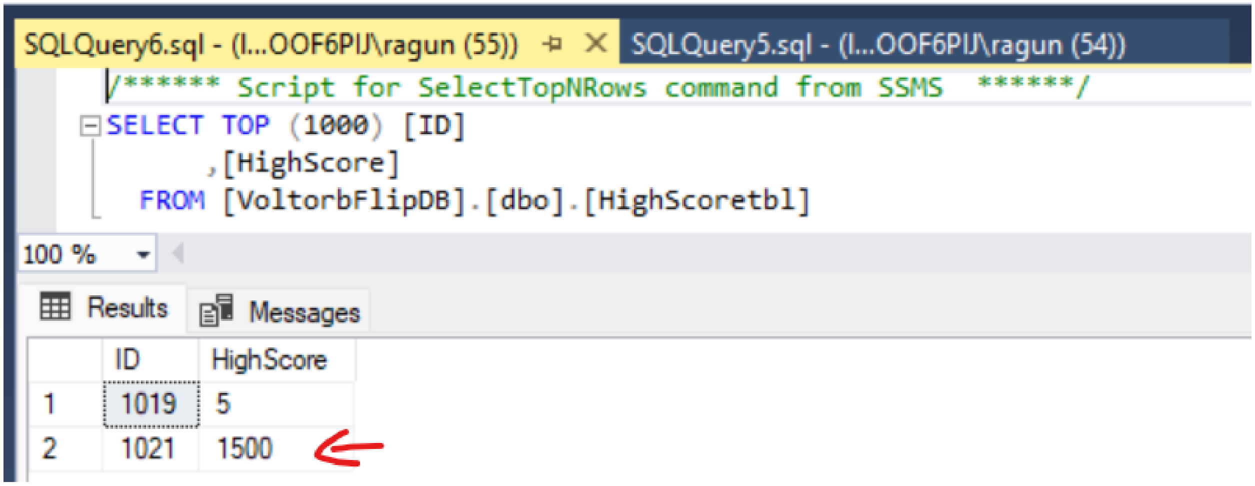
```
highscore= 1500
```

Two different methods were used to test the API calls. The GET call was tested with Postman and the POST high score was tested with HTTP Live Header extension on Firefox. The reason our team used that to test the POST is because it will auto populate the all the headers required to send the API Post request.



```
statusCode:    200
```

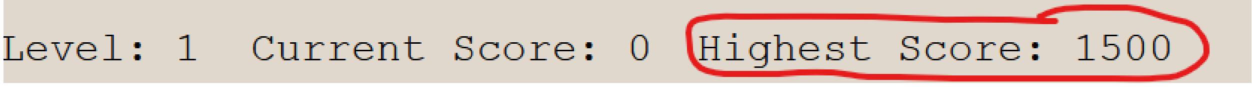
When I send the above Post request it gave us statusCode: 200 which is HTTP ok code.


SQLQuery6.sql - (l...OOF6PIJ\ragun (55))
SQLQuery5.sql - (l...OOF6PIJ\ragun (54))
100 %
Results
Messages

```sql
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [ID]
      ,[HighScore]
  FROM [VoltorbFlipDB].[dbo].[HighScoretbl]
```

	ID	HighScore
1	1019	5
2	1021	1500

Before the request I had cleared the database old data's and had only one entry with 5 as high score. After sending that POST call as you can see new entry was added to our database with value 1500.



```
Level: 1  Current Score: 0  Highest Score: 1500
```

Here you can see the result reflected on the website when we refresh the site after the updating the value with HTTP POST.

Conclusion:

Tangerines is a fun and engaging puzzle game that was designed with the player in mind. The game features challenging gameplay mechanics, colorful visuals, and an intuitive user interface. These design choices help to make Tangerines a game that is enjoyable and accessible for players of all skill levels.