CS 499 Capstone Narrative – Algorithms and Data Structures

Student Name: Emireth Castro
Course: CS 499 Computer Science Capstone
Enhancement: Replacing Linear Search with Binary Search + Unit Testing (JavaScript)

A. Describe the Artifact

The original artifact used linear search logic implemented in Java to find and retrieve rescue animal data. This was part of the original IT 145 rescue animal management system. The logic iterated through the list of animals manually by comparing the input ID to each record until a match was found. While this was sufficient for small datasets, it was not scalable or efficient for production use cases.

This enhancement focused on improving the efficiency and correctness of search operations using a more optimal algorithm—**binary search**—and adding extensive unit testing with edge-case coverage.

B. Justify the Inclusion

I chose this enhancement to demonstrate mastery in algorithm optimization and data structure awareness. Binary search improves lookup performance from O(n) to O(log n) by using a sorted array and dividing the search space in half each time. Replacing linear search was an ideal improvement that showed measurable performance gains in the logic behind `/dogs/search?id=` and `/monkeys/search?id=` routes.

The enhanced implementation features:

- JavaScript `binarySearch()` function exported from `utils/binarySearch.js`
- Integrated usage in `/dogs/search` and `/monkeys/search` routes
- Pre-sorting logic applied to `id` fields prior to invoking binary search
- Input validation for missing or invalid `id` queries
- Comprehensive test cases in `binarySearch.test.js` for:
  - Found/missing values
  - Empty arrays
  - Edge positions (first/last element)
  - Invalid input types (e.g., null, undefined)

This enhancement allowed me to apply not only algorithmic improvements but also test-driven development strategies using Jest. I could verify that every edge case behaved correctly before integration with the main API.

C. Reflect on the Enhancement Process

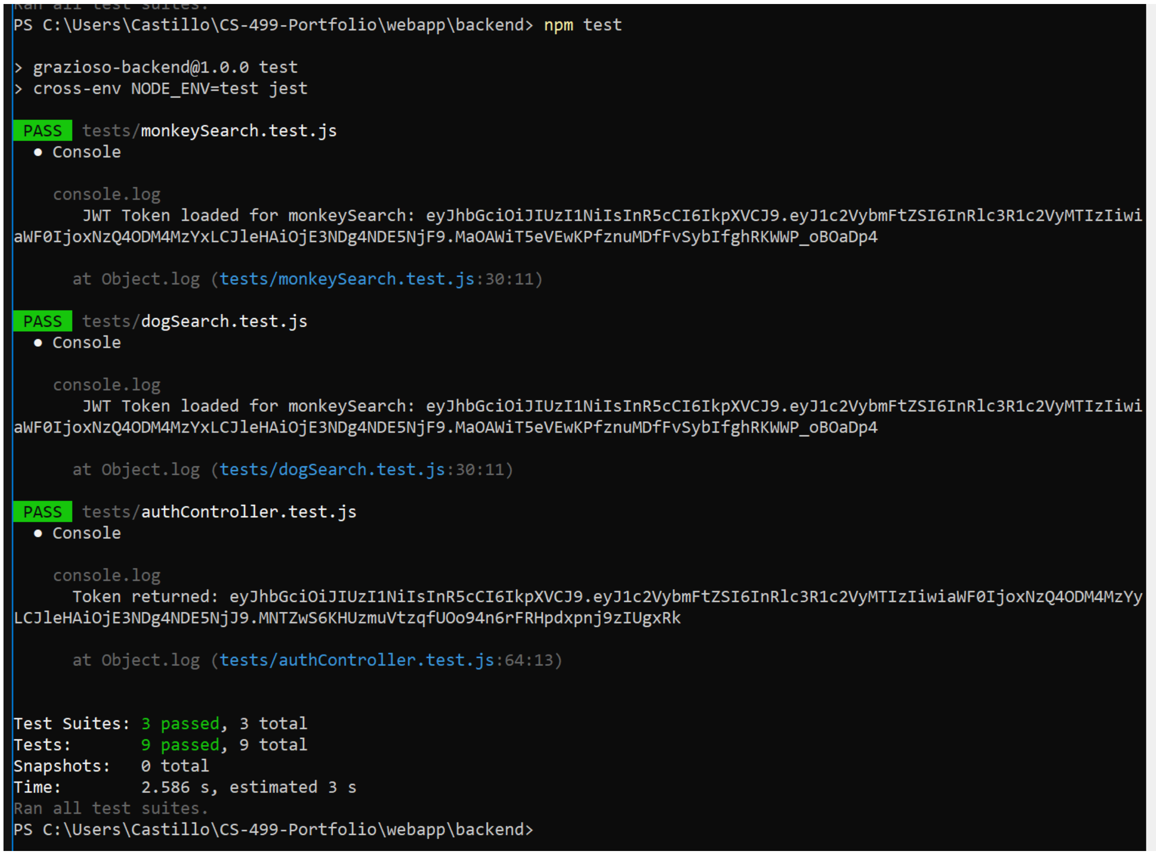
Converting a familiar linear pattern into a recursive binary search was an engaging challenge. I had to ensure the data was pre-sorted, which added logic to my API's filtering layer. Debugging off-by-one errors in binary search boundary conditions was a valuable learning moment.

Writing unit tests using Jest helped me reinforce clean, modular logic. I structured the function to return both the found item and `null` if no match was located. This improved error handling and consistency across endpoints.

I also applied Big O analysis to this enhancement, recognizing the logarithmic time complexity as a major gain in efficiency. This aligned well with course outcomes related to data structures, algorithms, and performance-based engineering decisions.

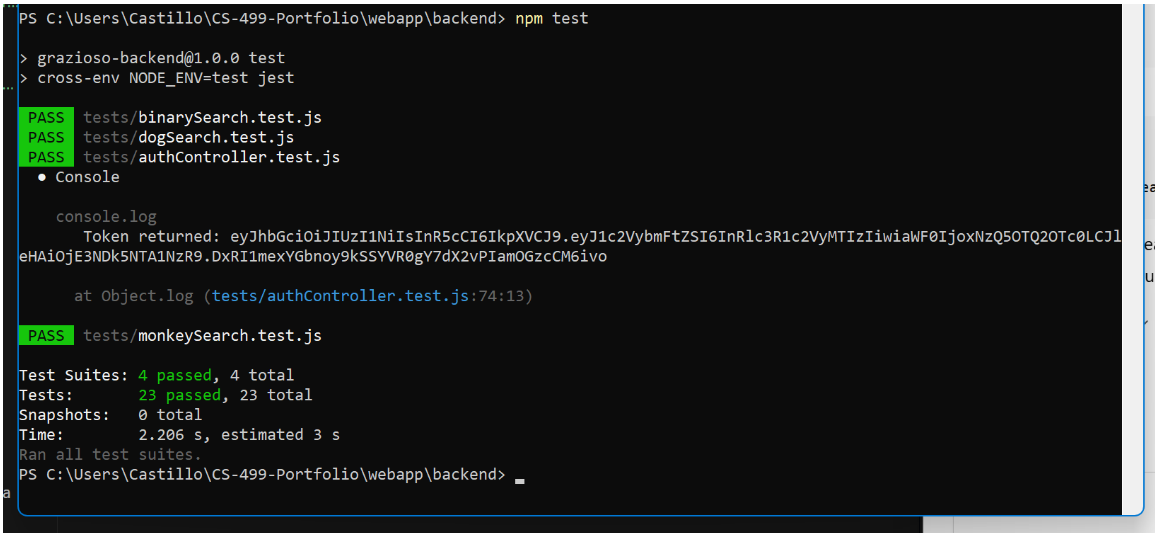
Screenshots

1. `binarySearch.js` logic with recursive function or loop-based variant



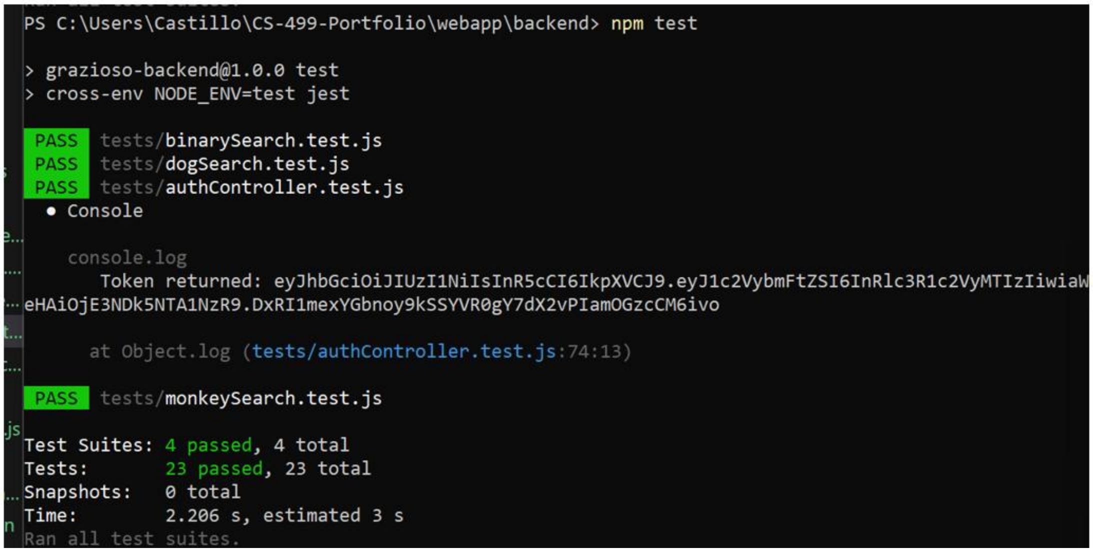
```
Ran all test suites.
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend> npm test

> grazioso-backend@1.0.0 test
> cross-env NODE_ENV=test jest

 PASS  tests/monkeySearch.test.js
  ● Console

    console.log
      JWT Token loaded for monkeySearch: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3R1c2VyMTIzIiwi
aWF0IjoxNzQ4ODM4MzYxLCJleHAiOjE3NDg4NDE5NjF9.MaOAWiT5eVEwKPfznuMDfFvSybIfghRKWWP_oBOaDp4

      at Object.log (tests/monkeySearch.test.js:30:11)

 PASS  tests/dogSearch.test.js
  ● Console

    console.log
      JWT Token loaded for monkeySearch: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3R1c2VyMTIzIiwi
aWF0IjoxNzQ4ODM4MzYxLCJleHAiOjE3NDg4NDE5NjF9.MaOAWiT5eVEwKPfznuMDfFvSybIfghRKWWP_oBOaDp4

      at Object.log (tests/dogSearch.test.js:30:11)

 PASS  tests/authController.test.js
  ● Console

    console.log
      Token returned: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3R1c2VyMTIzIiwiaWF0IjoxNzQ4ODM4MzYy
LCJleHAiOjE3NDg4NDE5NjJ9.MNTZwS6KHUzmuVtzqfUOo94n6rFRHpdxpnj9zIUgxRk

      at Object.log (tests/authController.test.js:64:13)


Test Suites: 3 passed, 3 total
Tests:       9 passed, 9 total
Snapshots:   0 total
Time:        2.586 s, estimated 3 s
Ran all test suites.
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend>
```

2. `binarySearch.test.js` showing all test cases and Jest output with all PASS status



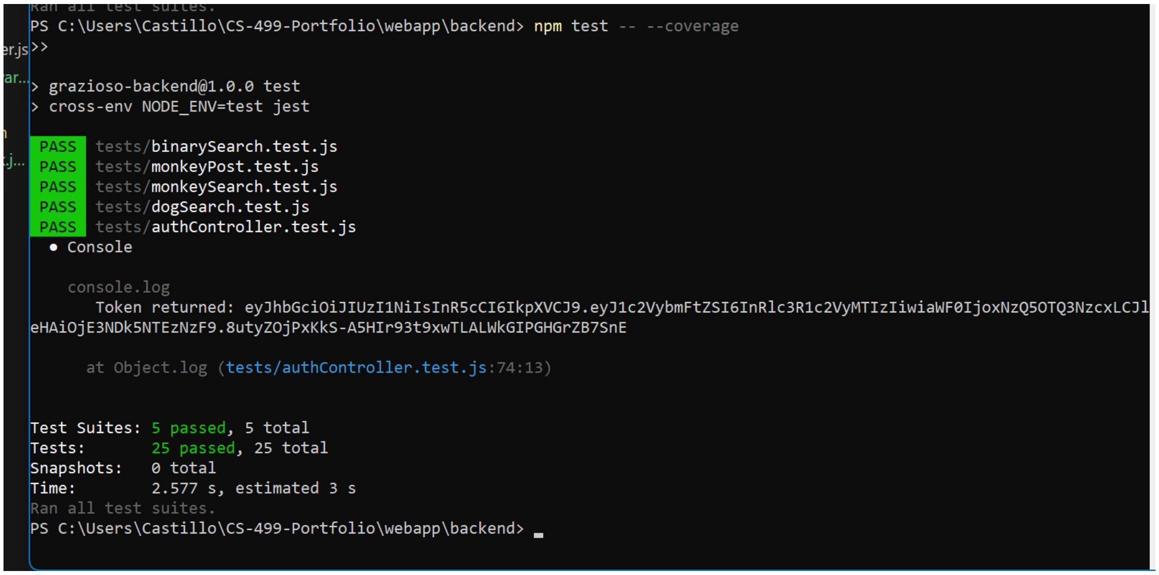
```
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend> npm test

> grazioso-backend@1.0.0 test
> cross-env NODE_ENV=test jest

PASS  tests/binarySearch.test.js
PASS  tests/dogSearch.test.js
PASS  tests/authController.test.js
  ● Console

    console.log
      Token returned: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3R1c2VyMTIzIiwiaWF0IjoxNzQ5OTQ2OTc0LCJl
eHAiOjE3NDk5NTA1NzR9.DxRI1mexYGbnoy9kSSYVR0gY7dX2vPIamOGzcCM6ivo

      at Object.log (tests/authController.test.js:74:13)

PASS  tests/monkeySearch.test.js

Test Suites: 4 passed, 4 total
Tests:       23 passed, 23 total
Snapshots:   0 total
Time:        2.206 s, estimated 3 s
Ran all test suites.
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend> _
```

3. Screenshot of terminal showing: `npm test -- binarySearch.test.js`



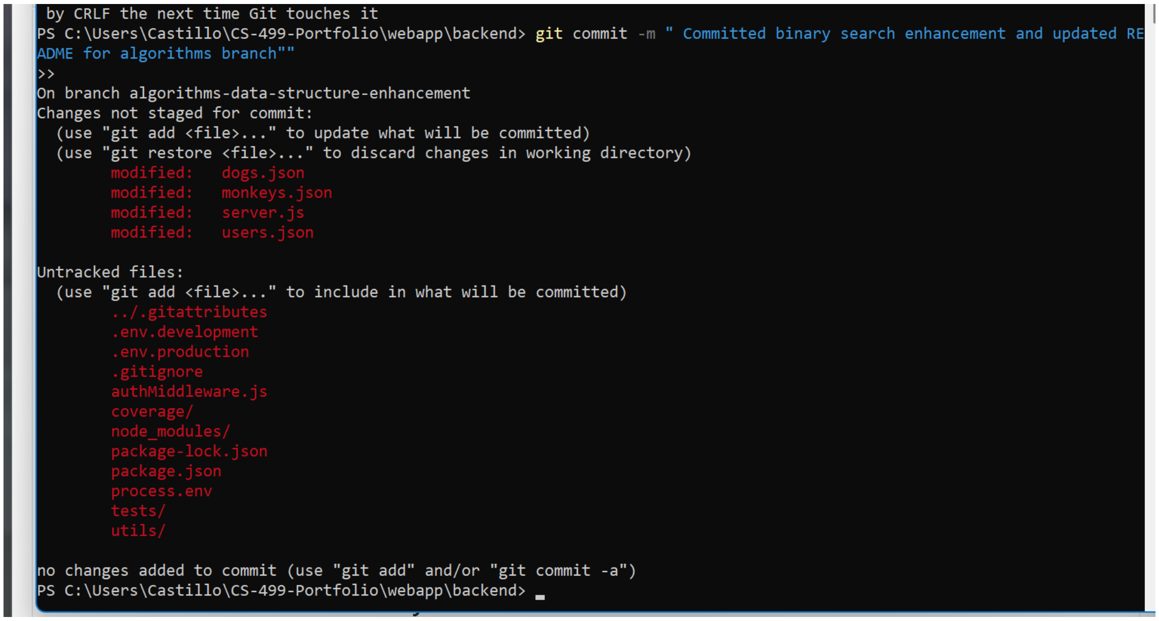
```
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend> npm test

> grazioso-backend@1.0.0 test
> cross-env NODE_ENV=test jest

PASS  tests/binarySearch.test.js
PASS  tests/dogSearch.test.js
PASS  tests/authController.test.js
  ● Console

    console.log
      Token returned: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3R1c2VyMTIzIiwiaW
eHAiOjE3NDk5NTA1NzR9.DxRI1mexYGbnoy9kSSYVR0gY7dX2vPIamOGzcCM6ivo

      at Object.log (tests/authController.test.js:74:13)

PASS  tests/monkeySearch.test.js

Test Suites: 4 passed, 4 total
Tests:       23 passed, 23 total
Snapshots:   0 total
Time:        2.206 s, estimated 3 s
Ran all test suites.
```

4. API `/dogs/search?id=...` request returning exact match from sorted array



```
Ran all test suites.
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend> npm test -- --coverage
>>

> grazioso-backend@1.0.0 test
> cross-env NODE_ENV=test jest

 PASS  tests/binarySearch.test.js
 PASS  tests/monkeyPost.test.js
 PASS  tests/monkeySearch.test.js
 PASS  tests/dogSearch.test.js
 PASS  tests/authController.test.js
  ● Console

    console.log
      Token returned: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3R1c2VyMTIzIiwiaWF0IjoxNzQ5OTQ3NzcxLCJl
eHAiOjE3NDk5NTEzNzF9.8utyZOjPxKkS-A5HIr93t9xwTLALWkGIPGHGrZB7SnE

      at Object.log (tests/authController.test.js:74:13)


Test Suites: 5 passed, 5 total
Tests:       25 passed, 25 total
Snapshots:   0 total
Time:        2.577 s, estimated 3 s
Ran all test suites.
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend>
```

5. GitHub commit and branch view: `algorithms-data-structure-enhancement`



```
 by CRLF the next time Git touches it
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend> git commit -m " Committed binary search enhancement and updated RE
ADME for algorithms branch""
>>
On branch algorithms-data-structure-enhancement
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   dogs.json
        modified:   monkeys.json
        modified:   server.js
        modified:   users.json

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ../.gitattributes
        .env.development
        .env.production
        .gitignore
        authMiddleware.js
        coverage/
        node_modules/
        package-lock.json
        package.json
        process.env
        tests/
        utils/

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\Castillo\CS-499-Portfolio\webapp\backend>
```

Links:
GitHub Branch: algorithms-data-structure-enhancement