Comments

…

Score Legend

N/A	Not relevant to submitted code
0	Seriously needs improvement
1	Needs improvement
2	Satisfactory
3	Good
4	Excellent

Language / Library Fluency (20 / 56)

Criteria	Score	Comments
Free of syntax errors and runtime errors generated by tests	_ / 4	
Properly manages control flow (conditionals, iteration, etc.)	_ / 4	
Uses full range of API as intended rather than hacking together solutions with only a few parts	_ / 4	
Promise chains properly compose asynchronous actions	_ / 4	
Incorporates some ES6 (let, const, arrows, destructuring, etc.)	_ / 4	

Total: _ / 20

Economy and Performance (12 / 56)

Criteria	Score	Comments
Avoids repetition in favor of reusable variables and functions (DRY)	_ / 4	
Chooses the appropriate data structures and methods for the job	_ / 4	
Doesn't perform unnecessary operations or declare unused variables or functions	_ / 4	

Total: _ / 12

Readability and Maintainability (12 / 56)

Criteria	Score	Comments
Variable and function names are meaningful and self-documenting	_ / 4	
Formatting is consistent (including whitespace, indentation, semicolons, bracket placement, etc.)	_ / 4	
Comments help clarify complex or unconventional code	_ / 4	

Total: _ / 12

Cleanliness (12 / 56)

Criteria	Score	Comments
Handles errors properly and consistently	_ / 4	
Doesn't contain commented-out code (except to show incomplete thoughts) or console.log statements	_ / 4	
Conforms to best practices and avoids use of deprecated features or anything known to introduce bugs	_ / 4	

Total: _ / 12

Overall

Source Data

Rubric score: _ / 56

Base specs passing: _ / 39

Extra credit: _ / 10

Additional Variables

Spec adjustment (give / take partial credit): +/- _[1]

Percentages

(25% of final) Rubric score: (_ / 56) * 100% = _ rubric%

(75% of final) Base score: (_ / 39) * 100% = _ base%

(up to 10% added) Extra credit: (_ / 10) * 100% = _ EC%

Final Grade: (base% * 75%) + (rubric% * 25%) + (EC% * 10%) = _%

---

[1] Comments about the grader's choice(s) to give / take partial credit