CODE REVIEW EVALUATION FORM

JavaScript & Express.js | Undergraduate Programming Course

1. SUBMISSION INFORMATION

Course:	ICS 385	**Section:**	
Instructor:	Dr. Debasis Bhattacharya	**Semester:**	Spring
Student Name:	Hayden Suzuki	**Student ID:**	
Project Title:	Secrets Project	**Date:**	2/14/2026
Reviewer:		**Review Type:**	Peer / Instructor

2. CODE SUBMISSION DETAILS

Repository URL:	https://github.com/debasisb/ics385spring2026/tree/main/week5/3.5%20Secrets%20Project		
Branch:	3.5 Secrets Project	**Commit Hash:**	4090094
Files Reviewed:	solution.js	**Lines of Code:**	38

3. CODE OVERVIEW & PURPOSE

Briefly describe the purpose of the submitted code, its main functionality, the Express.js routes implemented, and any middleware or external packages used.

Summary:

The purpose of the code is to create a simple web page application that uses Express.js to display a fill-in text box acting as a password checker to reveal a secret page. It uses middleware in the form of passwordCheck, and the package body-parser.

4. EVALUATION CRITERIA

Rate each criterion on the scale provided. Use the descriptors as guidance. A score of 4 = Excellent, 3 = Proficient, 2 = Developing, 1 = Beginning, 0 = Not Attempted.

Criterion	Description	Score (0–4)	Weight
Code Correctness & Functionality	Application runs without errors; all Express routes return expected responses; edge cases handled.	3	20%

Criterion	Description	Score (0–4)	Weight
Code Structure & Organization	Logical file/folder structure (e.g., routes/, controllers/, models/); separation of concerns; modular design.	3	15%
Naming Conventions & Readability	Variables, functions, and routes use clear, descriptive names following camelCase conventions; consistent formatting.	4	10%
Express.js Best Practices	Proper use of Router, middleware chaining, error-handling middleware, appropriate HTTP methods and status codes.	1	15%
Error Handling & Validation	Input validation present; try/catch or .catch() used; meaningful error messages returned to client.	1	10%
Comments & Documentation	Inline comments explain non-obvious logic; README or header comments describe setup, dependencies, and usage.	1	10%
Security Considerations	No hardcoded secrets; use of environment variables; input sanitization; helmet or CORS configured if applicable.	0	10%
Testing & Reliability	At least basic test cases provided (e.g., using Jest or Supertest); tests cover primary routes and edge cases.	0	10%

Total Weighted Score:	1.62 / 4.00	**Percentage:**	41 %

5. DETAILED FINDINGS — CODE-LEVEL OBSERVATIONS

Document specific issues, bugs, or noteworthy patterns found during the review. Reference file names and line numbers where applicable.

#	File / Line	Severity	Category	Description / Observation
1	solution.js line 16	High / Med / Low	Security	Password is visible in source code
2	solution.js line 28	High / Med / Low	Security	The user is verified until the server is reset after inputting the password a single time.
3	solution.js	High / Med / Low		Does not validate when password is input, page is simply refreshed if the password is incorrect
4		High / Med / Low		
5		High / Med / Low		
6		High / Med / Low		
7		High / Med / Low		
8		High / Med / Low		

6. EXPRESS.JS & JAVASCRIPT CHECKLIST

Check each item that applies to the submitted code. Mark Y (Yes), N (No), or N/A.

Category	Checklist Item	Y / N / N/A
Server Setup	Server listens on a configurable port (e.g., process.env.PORT)	N
Server Setup	Entry point file is clearly identified (e.g., app.js or server.js)	Y
Routing	Routes are organized using express.Router()	N
Routing	RESTful conventions followed (GET, POST, PUT/PATCH, DELETE)	N
Routing	Route parameters and query strings used correctly	N/A
Middleware	Body-parser or express.json() configured for request parsing	Y
Middleware	Custom middleware is reusable and well-documented	N
Middleware	Error-handling middleware defined with (err, req, res, next) signature	N
Async/Await	Promises and async/await used correctly (no unhandled rejections)	N
Async/Await	Callback patterns avoided in favor of modern async patterns	N
Dependencies	package.json lists all dependencies; no unused packages	Y
Dependencies	node_modules excluded via .gitignore	Y

Category	Checklist Item	Y / N / N/A
Security	Environment variables managed via .env / dotenv	Y
Security	No sensitive data committed to version control	N

7. QUALITATIVE FEEDBACK

Strengths — What does this submission do well?

:

The submission fulfills its aim in a simple and basic manner, runs correctly and uses express functions correctly

Areas for Improvement — What should the student focus on next?

:

The submission has very big security flaws, the creator should focus heavily on authentication. The password being hard-coded in the source file is a major security issue.

Suggested Learning Resources

:

8. OVERALL ASSESSMENT

Grade	Range	Description
A / Excellent	90–100%	Code is well-structured, fully functional, secure, and demonstrates mastery of Express.js concepts.
B / Proficient	80–89%	Code works correctly with minor issues; good organization and documentation; some improvements possible.
C / Developing	70–79%	Code runs but has notable gaps in structure, error handling, or best practices; needs revision.
D / Beginning	60–69%	Significant issues with functionality, structure, or documentation; substantial rework required.
F / Incomplete	Below 60%	Code does not compile/run or is largely incomplete; fundamental concepts not demonstrated.

Final Grade Assigned:	C/ Developing	**Numeric Score:**	70 / 100

9. REQUIRED REVISIONS & ACTION ITEMS

List any mandatory changes the student must complete before resubmission.

#	Action Item	Priority	Due Date
1	Line 16, remove password from code	High / Med / Low	
2	Session-based authentication	High / Med / Low	
3	Login responsiveness, if password is incorrect display a popup/text	High / Med / Low	
4		High / Med / Low	

10. ACADEMIC INTEGRITY ACKNOWLEDGMENT

By signing below, the reviewer confirms that this evaluation was conducted fairly and objectively. The student acknowledges receipt of this feedback and understands the revisions required.

Reviewer Signature:		**Date:**	
Student Signature:		**Date:**	
Instructor Signature:		**Date:**	