Backend/Architecture design role 2 - General good backend coding practices #4

1. Writing Secure Code:

- We need to protect any type of sensitive data including passwords and API keys. This can be done by using hashing and environmental variables.
- Avoid hardcoding secrets in Source code.
- Use secure authentication and authorization methods.
  Sources:
- OWASP Secure Coding Practices: https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/
- MDN Web Security: https://developer.mozilla.org/en-US/docs/Web/Security

2. Design Efficient and Scalable Databases:

- Normalize data to reduce redundancy.
- Use indexing to improve query performance.
- Optimize queries to reduce server load.
  Sources:
- PostgreSQL Documentation: https://www.postgresql.org/docs/current/indexes.html
- MongoDB Data Modeling: https://www.mongodb.com/docs/manual/core/data-model-design/

3. Write Clean and Readable Code

- Use meaningful variable and function names
- Follow consistent formatting and style
- Keep functions small and focused.
  Sources:
- Clean Code Principles (Robert C. Martin summary): https://www.freecodecamp.org/news/clean-coding-for-beginners/
- Google Style Guide: https://google.github.io/styleguide/

4. Modularize Your Code:

- Break Backend into smaller components (controllers, services, models)
- Use separation of concerns
- Makes code easier to test and maintain
  Sources:
- https://martinfowler.com/bliki/SeparationOfConcerns.html

5. Implement Proper Error handling

- Return meaningful error messages
- Handle exceptions so that it doesn't crash the servers
- Log errors for debugging purposes.
- Sources:
  MDN Error Handling:
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Control_flow_and_error_handling
- Node.js Error Handling: https://nodejs.org/api/errors.html

6. Use Version Control Properly

- Track changes using Git.
- Use branches for features and fixes.
- Write meaningful commit messages.
  Sources:
- Git Documentation: https://git-scm.com/docs
- Atlassian Git Guide: https://www.atlassian.com/git/tutorials

7. Write Documentation

- Document APIs (endpoints, parameters, responses)
- Use tools like Swagger/OpenAPI
- Helps team collaboration and future development
  Sources:

- OpenAPI Specification: https://swagger.io/specification/
- Write Docs Guide: https://www.writethedocs.org/guide/

8. Test Your Backend

- Write unit and integration tests
- Automate testing when possible
- Catch bugs early on.
  Sources:

- Testing Pyramid: https://martinfowler.com/articles/practical-test-pyramid.html
- Jest Docs: https://jestjs.io/docs/getting-started