Final Submission Checklist

Estimated time needed: 2 mins

Before submitting your DevOps Capstone Project, review the submission options carefully.
You will submit your work through **either Option 1: AI Graded Submission** or **Option 2: Peer Graded Submission**.

Use the checklist below to ensure you have prepared all required GitHub URLs, screenshots, and output files.

Option 1: AI Graded Submission

If you choose **Option 1: AI Graded Submission**, you must upload all required **GitHub URLs**, **screenshots**, and **output files**.
This checklist ensures that every URL points to the correct repository file, every screenshot clearly displays the required board status or story movement, and every output file contains the expected terminal or cURL results.

Use this list to verify that **all required links, screenshots, and logs generated during your labs have been saved correctly**, as the AI grader will evaluate your submission automatically based on these artifacts.

AI-Graded Final Checklist

1. **Provide the public GitHub URL of the README.md file** containing the project name and updated build status badge after a successful build. *(2 points)*
2. **Provide the public GitHub URL of the user-story.md file** containing the user story template. *(1 point)*
3. **Take and submit a screenshot** named *Planning-userstories-done.jpeg/png* showing all user stories from the “New Issues” column. *(1 point)*
4. **Take and submit a screenshot** named *planning-productbacklog-done.jpeg/png* showing the user stories listed under the Ice Box. *(1 point)*
5. **Take and submit a screenshot** named *planning-labels-done.jpeg/png* showing user stories labeled as Technical Debt or Enhancement. *(1 point)*
6. **Take and submit a screenshot** named *planning-kanban-done.jpeg/png* showing the stories in the Sprint Backlog. *(1 point)*
7. **Provide the public GitHub URL of the setup.cfg file** containing nosetests, coverage, Flake8, and Pylint configuration. *(1 point)*
8. **Take and submit a screenshot** named *rest-techdebt-done.jpeg/png* showing “Setting up the development environment” moved to Done. *(1 point)*
9. **Take and submit a screenshot** named *read-accounts.jpeg/png* showing “Read an account from the service” moved to Done. *(1 point)*
10. **Take and submit a screenshot** named *list-accounts.jpeg/png* showing “List all accounts in the service” moved to Done. *(1 point)*
11. **Take and submit a screenshot** named *update-accounts.jpeg/png* showing “Update an account in the service” moved to Done. *(1 point)*
12. **Take and submit a screenshot** named *delete-accounts.jpeg/png* showing “Delete an account from the service” moved to Done. *(1 point)*
13. **Save the cURL command and its output** demonstrating the CREATE function in a file named *rest-create-done*, and submit it. *(2 points)*
14. **Save the cURL command and its output** demonstrating the LIST function in a file named *rest-list-done*, and submit it. *(2 points)*
15. **Save the cURL command and its output** demonstrating the READ function in a file named *rest-read-done*, and submit it. *(2 points)*
16. **Save the cURL command and its output** demonstrating the UPDATE function in a file named *rest-update-done*, and submit it. *(2 points)*
17. **Save the cURL command and its output** demonstrating the DELETE function in a file named *rest-delete-done*, and submit it. *(2 points)*
18. **Take and submit a screenshot** named *sprint2-plan.jpeg/png* showing the two new Sprint 2 user stories under the Sprint Backlog. *(1 point)*
19. **Save and submit the terminal output** as *ci-workflow-done* showing all GitHub Actions workflow steps executed successfully. *(2 points)*
20. **Take and submit a screenshot** named *ci-kanban-done.jpeg/png* showing the CI automation story moved to Done. *(1 point)*
21. **Provide the public GitHub URL of the ci-build.yaml file** containing the CI pipeline configuration. *(4 points)*
22. **Provide the public GitHub URL of the init.py file** containing Talisman security headers configuration. *(1 point)*
23. **Submit the complete nosetests output** as security-headers-done, showing all tests passing after implementing the CORS policies. (1 point)
24. **Take and submit a screenshot** named *security-kanban-done.jpeg/png* showing the security story moved to Done. *(1 point)*
25. **Take and submit a screenshot** named *sprint3-plan.jpeg/png* showing the three user stories added to Sprint 3. *(1 point)*
26. **Save and submit the JSON output** as *kube-app-output* generated when the application launches on port 8080. *(1 point)*
27. **Take and submit a screenshot** named *kube-docker-done.jpeg/png* showing the Docker containerization story moved to Done. *(1 point)*
28. **Take and submit a screenshot** named *kube-kubernetes-done.jpeg/png* showing the Kubernetes deployment story moved to Done. *(1 point)*
29. **Provide the public GitHub URL of the Dockerfile** containing the complete Docker configuration. *(2 points)*
30. **Save and submit the Docker images output** as a file named *kube-images*, showing Name, Tag, Image ID, Created Time, and Size. *(2 points)*
31. **Save and submit the Kubernetes deployment details** as *kube-deploy-accounts*, including deployments, pods, replica sets, and service info. *(2 points)*
32. **Save and submit the complete Tekton pipeline logs** as *pipelinerun.txt*. *(5 points)*
33. **Take and submit a screenshot** named *cd-pipeline-done.jpeg/png* showing the CD pipeline automation story moved to Done. *(1 point)*

Option 2: Peer Graded Submission

If you choose **Option 2: Peer Review Submission**, you only need to upload **screenshots**, **files**, and **URLs**.
This checklist helps you confirm that every screenshot clearly shows the correct board status, story location, configuration file, or output.

Use this list to verify that **all required screenshots and URLs from previous labs have been captured correctly**.

Peer Review Final Checklist

1. **Task 1 a)** Upload the screenshot **planning-repository-done.jpg**, showing the repository successfully created. *(1 point)*
2. **Task 1 b)** Upload the screenshot **planning-storytemplate-done.jpg**, showing the user-story template file in the repository. *(1 point)*
3. **Task 1 c)** Upload the screenshot **planning-userstories-done.jpg**, showing all user stories listed in the *New Issues* column. *(1 point)*
4. **Task 1 d)** Upload the screenshot **planning-productbacklog-done.jpg**, showing all user stories under the *Ice Box* column. *(1 point)*
5. **Task 1 e)** Upload the screenshot **planning-labels-done.jpg**, showing user stories labeled as *Technical Debt* or *Enhancement*. *(1 point)*
6. **Task 1 f)** Upload the screenshot **planning-kanban-done.jpg**, showing stories in the *Sprint Backlog* column. *(1 point)*
7. **Task 2 a)** Upload the screenshot **rest-setupcfg-done.jpg**, showing the `setup.cfg` file containing linting/testing configurations. *(1 point)*
8. **Task 2 b)** Upload the screenshot **rest-techdebt-done.jpg**, showing “Setting up the development environment” story moved to *Done*. *(1 point)*
9. **Task 2 c)** Upload **read-accounts.jpg**, showing “Read an account” story moved to *Done*. *(1 point)*
10. **Task 2 d)** Upload **list-accounts.jpg**, showing the “List accounts” story moved to *Done*. *(1 point)*
11. **Task 2 e)** Upload **update-accounts.jpg**, showing the “Update account” story moved to *Done*. *(1 point)*
12. **Task 2 f)** Upload **delete-accounts.jpg**, showing the “Delete account” story moved to *Done*. *(1 point)*
13. **Task 2 g)** Upload **rest-create-done.jpg**, showing the cURL request and response for the CREATE API. *(2 points)*
14. **Task 2 h)** Upload **rest-read-done.jpg**, showing the cURL request and response for the READ API. *(2 points)*
15. **Task 2 i)** Upload **rest-list-done.jpg**, showing the cURL request and response for the LIST API. *(2 points)*
16. **Task 2 j)** Upload **rest-update-done.jpg**, showing the cURL request and response for the UPDATE API. *(2 points)*
17. **Task 2 k)** Upload **rest-delete-done.jpg**, showing the cURL request and response for the DELETE API. *(2 points)*

18. **Task 3 a)** Upload **sprint2-plan.jpg**, showing two new Sprint 2 stories added to the *Sprint Backlog*. *(1 point)*
19. **Task 3 b)** Upload **ci-workflow-done.jpg**, showing the successful GitHub Actions workflow run. *(2 points)*
20. **Task 3 c)** Upload **ci-badge-done.jpg**, showing the CI status badge visible in the README. *(1 point)*
21. **Task 3 d)** Upload **ci-kanban-done.jpg**, showing the "Automate CI checks" story in the *Done* column. *(1 point)*
22. **Task 3 e)** Provide the GitHub URL containing **ci-build.yaml**, showing the full workflow configuration. *(4 points)*
23. **Task 3 i)** Upload **security-code-done.jpg**, showing the implemented security header and CORS code. *(1 point)*
24. **Task 3 j)** Upload **security-headers-done.jpg**, showing Talisman and CORS headers in the running application. *(1 point)*
25. **Task 3 k)** Upload **security-kanban-done.jpg**, showing the "Add security headers and CORS" story moved to Done. *(1 point)*
26. **Task 4 a)** Upload **sprint3-plan.jpg**, showing all three Sprint 3 stories added under the Sprint Backlog. *(1 point)*
27. **Task 4 b)** Upload **kube-app-output.jpg**, showing the JSON response from the app running on **port 8080**. *(1 point)*
28. **Task 4 c)** Upload **kube-docker-done.jpg**, showing the Docker containerization story moved to *Done*. *(1 point)*
29. **Task 4 d)** Provide the GitHub URL of the **Dockerfile**, showing the complete Docker build instructions. *(2 points)*
30. **Task 4 e)** Upload **kube-images.jpg**, showing Docker image details (name, tag, ID, created time, size). *(2 points)*
31. **Task 4 f)** Upload **kube-deploy-accounts.jpg**, showing Kubernetes deployment, pods, replicasets, and services. *(2 points)*
32. **Task 4 g)** Upload **kube-kubernetes-done.jpg**, showing the "Deploy to Kubernetes" story moved to Done. *(1 point)*
33. **Task 5 a)** Upload **pipelinerun.txt**, showing complete logs of the Tekton pipeline run. *(5 points)*
34. **Task 5 b)** Upload **cd-pipeline-done.jpg**, showing the CD pipeline story moved to Done and automated deployment verified. *(1 point)*

Author(s)

Nikesh Kumar