

# ***KillTest***

高品質 より良いサービス



## 練習問題

<https://www.killtest.jp>

1年間無料アップデート

**Exam : Certified Backstage Associate**

**Title : Certified Backstage  
Associate (CBA)**

**Version : DEMO**

1.You have successfully run 'yarn docker:build' and 'yarn docker:up' to start Backstage services. However, when accessing <http://localhost:3000>, the Backstage interface does not load. What is the most likely cause?

- A. The app-config.yaml is missing.
- B. Yarn dependencies are not installed.
- C. Node.js version is incompatible.
- D. Docker containers failed to start correctly.

**Answer: D**

**Explanation:**

When running Backstage using Docker, it's crucial that all Docker containers start correctly to provide the necessary services. If the Backstage interface is not accessible, the most probable issue is that the Docker containers did not initialize as expected. This could be due to configuration errors, port conflicts, or resource limitations. Checking the Docker logs can help identify the exact cause. For troubleshooting steps, refer to the <https://medium.com/@yashpal.bhardwaj/setting-up-backstage-ac2cc96b24de>.

2.Which of the following configurations prevents the removal of statically configured locations through the catalog locations API?

- A. Adding locations under the catalog.locations key in the configuration
- B. Using 'readonly: true' in the catalog configuration
- C. Setting orphanStrategy to 'retain' in the catalog configuration
- D. Not providing delete permissions to the catalog API

**Answer: A**

**Explanation:**

In Backstage, statically configured locations added under the catalog.locations key cannot be removed through the catalog locations API. To remove these locations, they must be deleted directly from the configuration files. This design ensures that static locations remain persistent and are not inadvertently altered through API interactions. This behavior is crucial for maintaining the integrity of predefined catalog sources. For more information, refer to the Backstage Catalog documentation: <https://backstage.io/docs/features/software-catalog/configuration/>

3.What is the primary difference between 'full' and 'delta' mutations in Backstage's Entity Providers?

- A. 'Full' mutations allow entity processing loops to skip processing, while 'delta' mutations trigger processing loops.
- B. 'Full' mutations replace the entire set of entities in a provider's bucket, while 'delta' mutations only add, update, or delete specific entities.
- C. 'Full' mutations are used only for static locations, whereas 'delta' mutations are used for dynamic providers.
- D. 'Full' mutations require manual triggering, while 'delta' mutations are automatically triggered.

**Answer: B**

**Explanation:**

In Backstage's catalog ingestion process, 'full' mutations involve replacing the entire set of entities associated with a provider, which is useful when batch-fetching the entire dataset. On the other hand, 'delta' mutations focus on making specific changes by adding, updating, or deleting individual entities, which is more efficient for event-driven or incremental updates. Understanding the distinction helps in

choosing the appropriate mutation type based on the data source and update requirements. More information can be found in Backstage's catalog documentation:

<https://backstage.io/docs/features/software-catalog/external-integrations/>

4.You are new to Backstage and want to quickly explore its features without setting up a local environment.

What is the simplest way to achieve this?

- A. Refer to the Backstage API documentation
- B. Clone the Backstage repository and run it locally
- C. Deploy Backstage using Docker on your local machine
- D. Visit the live demo site of Backstage

**Answer: D**

**Explanation:**

The easiest way to explore Backstage is by visiting the live demo site. This allows users to interact with Backstage's features and functionalities without the need for any local setup or deployment. It's an ideal starting point for new users to familiarize themselves with the platform. For access, refer to the Backstage development workflow documentation: <https://backstage.io/docs/deployment/>

5.Your organization wants to integrate a new Single Sign-On (SSO) provider with Backstage to manage user access.

Which steps should you follow to properly set up this authentication method?

- A. Modify the 'app-config.yaml' to include the new SSO provider's client ID and secret, and implement the corresponding authentication plugin.
- B. Directly edit the Backstage source code to add the SSO provider's SDK.
- C. Set up the SSO provider in your organization's identity management system and Backstage will automatically detect it.
- D. Use the Backstage UI to select the new SSO provider without any configuration changes.

**Answer: A**

**Explanation:**

To integrate a new Single Sign-On (SSO) provider with Backstage, you need to update the 'app-config.yaml' file with the provider's client ID and secret, and then implement or configure the corresponding authentication plugin to handle the SSO process. This ensures that Backstage correctly authenticates users via the new SSO provider, maintaining secure access management. For detailed steps, refer to Backstage's authentication integration documentation.

<https://backstage.io/docs/overview/what-is-backstage/>