PassQueen
a.k.a dumpsqueen.com

SAP Certified Associate - Backend Developer - SAP Cloud Application Programming Model

SAP C_CPE_16

Version Demo

Total Demo Questions: 14

Total Premium Questions: 142

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, SAP BTP, Cloud Foundry Runtime	36
Topic 2, SAP BTP, Kyma Runtime	4
Topic 3, SAP Business Application Studio	16
Topic 4, SAP Fiori Elements	14
Topic 5, Cloud Application Programming Model	29
Topic 6, SAPUI5	4
Topic 7, SAP Fiori	4
Topic 8, Security	21
Topic 9, Mix Questions	14
Total	142

QUESTION NO: 1

Which CAP Node.js SDK class is used to register custom event handlers?

A. cds.Service

B. cds.Event

C. cds.Request

ANSWER: A

Explanation:

`cds.Service` is the CAP Node.js SDK class used to register custom event handlers. In CAP, services are the central runtime objects that expose domain operations and events. A service instance provides the event-registration APIs, including `before`, `on`, and `after`, which let an application attach logic to standard CRUD events such as `READ`, `CREATE`, `UPDATE`, and `DELETE`, as well as to custom actions and events defined in the service model. For example, custom behavior can be registered through a handler such as `this.on('submitOrder', handler)` within a service implementation, or through the corresponding service object obtained at runtime. The framework dispatches incoming requests and emitted events to handlers registered on that service instance, preserving CAP's service-oriented programming model. This makes `cds.Service` the appropriate class when implementing or extending service behavior in a CAP Node.js application. SAP's CAP documentation describes service implementations and their event-handler methods in the Node.js runtime guides: CAP Node.js Core Services and Providing Services.

QUESTION NO: 2

Within an XSUAA security descriptor, which identifier is used to distinguish an application and its scopes?

A. tenant-mode

B. xsappname

C. xs-security

D. VCAP_SERVICES

ANSWER: B

Explanation:

`xsappname` is the application identifier defined in an XSUAA security descriptor, typically in the `xs-security.json` file. XSUAA uses this value as the application namespace when it creates and evaluates authorization artifacts. In particular, scope names are qualified with the application identifier, allowing scopes belonging to one application to remain distinct from similarly named scopes belonging to another application. For example, a scope declared as `Display` is represented in authorization contexts using the application-specific namespace derived from `xsappname`. This makes the identifier essential for reliably defining role templates, assigning scopes, and checking permissions in an SAP BTP application. The same configured application name is also reflected in the OAuth client and authorization configuration generated for the XSUAA service instance, so it must be chosen carefully and kept consistent with the application's security design. SAP documentation describes `xsappname` as the identifier used to distinguish the application and its scopes in the security descriptor. See the SAP documentation on application security descriptor configuration and the SAP Cloud Security XSUAA integration documentation.

QUESTION NO: 3

Which statements accurately describe event handling in the SAP Cloud Application Programming Model (CAP)? Choose two.

A. You can register event handlers with instances of cds.service to add custom logic.

B. You can register only one event handler for a specific event.

C. You can register multiple event handlers for each event phase.

D. You must use the handler registration API srv.emit() to de-register event handlers.

ANSWER: A C

Explanation:

“You can register event handlers with instances of cds.service to add custom logic.” is correct because CAP services expose an event-handling API that lets an application attach custom logic to service events. In CAP Node.js applications, this is commonly done on a service instance by registering handlers for the `before`, `on`, or `after` phase. These handlers can validate input before processing, implement custom business behavior while processing, or enrich result data after processing. This event-driven model is central to extending generic service providers without replacing their standard behavior unnecessarily. “You can register multiple event handlers for each event phase.” is also correct. CAP supports registering several handlers for the same event and phase, enabling a service implementation to separate responsibilities into focused handler functions. The framework executes registered handlers according to its event-processing semantics, allowing custom logic to be composed with built-in or generic service behavior. This makes it practical to add validations, authorization-related logic, calculations, and result transformations independently. SAP documents the service event phases and handler registration mechanisms in the CAP Node.js Event Handlers documentation. The broader event model, including synchronous and asynchronous event processing, is described in the CAP Events documentation.

QUESTION NO: 4

Which statements correctly describe the relationship between YAML and JSON and their design goals? Choose all that apply.

A. JSON's foremost design goal is support for serializing arbitrary native data structures.

B. Each YAML file is a valid JSON file.

C. YAML's foremost design goal is support for serializing arbitrary native data structures.

D. Each JSON file is a valid YAML file.

ANSWER: C D

Explanation:

YAML was designed to represent common native data structures in a language-independent format. Its specification emphasizes mapping naturally to familiar structures such as mappings, sequences, and scalars, while remaining suitable for serialization and configuration. Therefore, the statement that “YAML's foremost design goal is support for serializing arbitrary native data structures.” accurately reflects YAML’s purpose and design orientation.

The statement “Each JSON file is a valid YAML file.” is correct under the YAML 1.2 specification. YAML 1.2 defines JSON as an official subset of YAML, meaning a syntactically valid JSON document can be processed as YAML. This compatibility is especially useful in SAP BTP projects because configuration and payload formats can often be consumed by tooling that supports YAML while retaining JSON-compatible content. The YAML specification explicitly describes the relationship by stating that YAML 1.2 is a superset of JSON. JSON itself is primarily a lightweight data-interchange format, whereas YAML provides additional syntax and features intended to make structured data more convenient for people to author and read.

References: YAML 1.2.2 Specification; JSON.org.

QUESTION NO: 5

As a general rule, when should you use namespaces in your data models?

A. When your model names are unique

B. When your app rarely exposes services

C. When your models are reused in other projects

ANSWER: C

Explanation:

Use namespaces when your models are reused in other projects. In SAP Cloud Application Programming Model (CAP), a namespace provides a stable, globally qualified prefix for the definitions within a CDS model, such as entities, types, and aspects. This prevents naming collisions when a reusable model is consumed together with models from other projects, domains, or third-party packages that may define artifacts with identical short names. A namespace therefore makes the model safer to distribute and easier for consuming applications to identify unambiguously. CAP documentation describes namespaces as a mechanism for avoiding naming conflicts, especially for reusable models. When models are intended only for a single application and are not shared, a namespace is often unnecessary because the application's own model structure may already provide sufficient context. Choosing a namespace for shared model content is consequently a practical design rule that supports modularity, clean integration, and long-term maintainability as the number of dependent projects grows. See the CAP documentation on CDS namespaces and the guidance for domain modeling.

QUESTION NO: 6

Which standard SAP buildpacks are available for developing and deploying applications in the SAP BTP, Cloud Foundry environment? Select three options.

A. Node.js

B. Java

C. Python

D. HTML5

E. Docker

ANSWER: A B C

Explanation:

Node.js, **Java**, and **Python** are standard language buildpacks available for application development in the SAP BTP, Cloud Foundry environment. A buildpack detects the application's runtime requirements during staging, provides the required language runtime and supporting dependencies, and creates a droplet that Cloud Foundry can run. This lets developers deploy source code without manually assembling the underlying runtime container for these supported languages. For example, the Node.js buildpack identifies Node.js applications and installs the requested Node.js version and dependencies; the Java buildpack supports common Java application types and frameworks; and the Python buildpack prepares Python applications based on their declared dependencies and runtime configuration. SAP BTP provides and maintains these buildpacks for use with the Cloud Foundry deployment model, while application-specific configuration such as buildpack selection, runtime versions, memory, and routes is typically defined in deployment metadata such as a manifest. SAP documentation describes how Cloud Foundry stages applications using buildpacks and how supported buildpacks are used in the SAP BTP environment. See SAP BTP buildpacks documentation and the Cloud Foundry buildpacks documentation.

QUESTION NO: 7

Which practices are core principles of Continuous Integration? Choose all correct answers.

A. Run tests in the build.

B. Use version control.

C. Run tests only in production.

D. Fix errors only when users complain.

E. Fix errors immediately.

ANSWER: A B E

Explanation:

Continuous Integration relies on integrating changes frequently into a shared codebase and validating each integration through an automated build. **Use version control.** is fundamental because a shared repository provides a controlled, traceable source of truth for application code and related project artifacts. It enables team members to integrate work regularly rather than maintaining long-lived, isolated changes.

Run tests in the build. is also a core practice. Automated tests executed as part of the build provide rapid feedback on whether a change has introduced a regression or broken expected behavior. In SAP Continuous Integration and Delivery, pipelines automate build, test, quality, and deployment-related steps, making repeatable validation part of the delivery process.

Fix errors immediately. supports the rapid-feedback objective of Continuous Integration. A failing build should be treated as a priority so the shared mainline remains reliable and subsequent changes can be integrated safely. Prompt remediation prevents defects and integration issues from accumulating, reduces troubleshooting effort, and preserves confidence that the latest integrated version is usable. See SAP's Continuous Integration and Delivery documentation and Martin Fowler's Continuous Integration overview.

QUESTION NO: 8

According to SAP CAP error-handling best practices, which types of errors should application code **not** catch? Choose two.

A. Unexpected errors

B. Programming errors

C. Rejections of promises

D. Runtime errors

ANSWER: A C

Explanation:

Unexpected errors and **Rejections of promises** should generally not be caught in CAP application handlers when the application cannot meaningfully recover from them. CAP provides centralized error processing for requests, including converting errors into appropriate protocol responses and logging them consistently. Allowing unexpected failures to propagate preserves the original error context and enables the framework and platform-level observability tools to record the failure correctly. Catching such failures merely to log and rethrow them can produce duplicate log entries, conceal useful stack-trace information, or inadvertently change the response behavior.

Similarly, an unhandled failure from an asynchronous operation should normally propagate through the promise chain to CAP's error handling. A promise rejection represents the asynchronous transport of an error; it should only be intercepted when the handler can actually resolve the condition or deliberately translate it into a domain-specific, expected request error. In CAP, expected business or validation conditions are best expressed through request error APIs such as `req.reject(...)`, rather than by broadly catching asynchronous failures. This keeps technical failures distinguishable from controlled business responses and supports reliable error reporting. See the SAP CAP Node.js Error Handling documentation and the Node.js Errors documentation.

QUESTION NO: 9

Which statements accurately describe Continuous Delivery? Select all correct answers.

A. Code changes are pushed to a remote source code management system.

B. The trigger for deployment to a productive system is a human decision.

C. Software is ready for deployment to a productive system all the time.

D. Feedback from a productive system gets quickly integrated into teams' backlog.

E. Deployment to a productive system is triggered automatically.

ANSWER: B C D

Explanation:

Continuous Delivery is a software-delivery practice in which every change that passes the automated build, test, and validation pipeline results in a deployable release candidate. Therefore, **"Software is ready for deployment to a productive system all the time."** describes its central outcome: the team maintains the application in a production-ready state rather than accumulating unreleased changes. Continuous Delivery deliberately retains a business or operational approval before the final productive deployment, so **"The trigger for deployment to a productive system is a human decision."** is also applicable. This approval can account for release timing, business readiness, governance, and risk controls while preserving automation in the preceding delivery stages. Fast learning is another essential aspect of an effective delivery process. **"Feedback from a productive system gets quickly integrated into teams' backlog."** reflects the feedback loop from production monitoring, user behavior, and operational experience into prioritization and future development. This supports continuous improvement and helps teams deliver valuable changes safely and frequently. SAP's CI/CD capabilities support automated quality gates and transport processes that help establish this repeatable, production-ready delivery flow. See SAP Continuous Integration and Delivery documentation and Martin Fowler's overview of Continuous Delivery.

QUESTION NO: 10

How do you make your company's existing user base available in an SAP BTP subaccount?

A. Establish a trust relationship with the default identity provider.

B. Import the users via .csv file upload in the SAP BTP cockpit.

C. Establish a trust relationship with your corporate identity provider.

D. Create the users manually in the security section of your subaccount.

ANSWER: C

Explanation:

Establish a trust relationship with your corporate identity provider. SAP BTP subaccounts use trust configuration to delegate user authentication to an identity provider (IdP). By configuring a trust relationship with the organization's corporate IdP, such as SAP Cloud Identity Services – Identity Authentication, Microsoft Entra ID, or another SAML 2.0-compliant provider, employees can sign in to SAP BTP using their established corporate identities. The IdP authenticates the user and sends identity assertions, including relevant attributes and group information where configured, to SAP BTP. Administrators can then map these identity attributes or groups to role collections in the subaccount, which grants the appropriate authorizations for applications and services. This approach centralizes identity lifecycle management in the corporate identity system: when users join, leave, or change roles, their identities remain governed by the organization's existing processes rather than being maintained separately in SAP BTP. SAP BTP supports configuring custom identity providers at subaccount level through the Trust Configuration area. For detailed configuration guidance, see the SAP BTP documentation on trust and federation with identity providers and the SAP Cloud Identity Services documentation for Identity Authentication.

QUESTION NO: 11

Are role collections defined for a global account automatically available in its subaccounts? Determine whether the statement is true or false.

A. true

B. false

ANSWER: B

Explanation:

The statement is false. In SAP BTP, authorization assignments are scoped to the account level where they are maintained. A global account and each of its subaccounts have separate authorization contexts, because they serve different administrative and operational purposes. Role collections used to grant access within a subaccount are managed in that subaccount and are assigned to users, user groups, or identity-provider groups there. Consequently, a role collection associated with global-account administration does not automatically become a role collection that can be used in a subaccount. Administrators must maintain the appropriate subaccount role collections and assignments in each relevant subaccount to grant access to subaccount resources, services, and applications. This separation supports least-privilege access: global-account administration can be delegated independently from permissions needed to administer or use individual subaccounts. SAP's documentation describes role collections as authorization artifacts that bundle roles for assignment in the applicable account context, while account administration is organized across the global account and its subaccounts. See the SAP BTP authorization and trust management documentation and the SAP guidance for managing role collections.

QUESTION NO: 12

After how long does GitHub automatically remove a personal access token that has not been used? Choose the correct answer.

A. 28 days

B. 1 month

C. 1 year

D. 100 days

ANSWER: C

Explanation:

1 year is correct because GitHub automatically removes personal access tokens that remain unused for one year. This inactivity-based removal helps reduce the security risk of credentials that may no longer be needed but could still provide access if exposed. Token use means the token is used to authenticate a request to GitHub; maintaining a token securely and using it only when required are important practices for controlling access to repositories and other GitHub resources. This automatic removal applies independently of a token's configured expiration date: a token can also become invalid earlier if it reaches an expiration date selected when it was created, or if an organization's security policy requires a shorter lifetime. Developers should therefore plan for token rotation and avoid relying on long-lived credentials in automation. For automated workloads, GitHub recommends using the most appropriate authentication mechanism and limiting token permissions to the minimum required scope. GitHub documents the one-year inactivity rule in its guidance on managing personal access tokens and also provides details on token expiration and secure usage. See Managing your personal access tokens and Creating a personal access token.

QUESTION NO: 13

When publishing an API, which practices does SAP recommend? Choose two answers.

A. Use meaningful, clear, and self-explanatory API names.

B. Remove obsolete APIs without notice.

C. Use version numbers in the API names.

D. Provide good API documentation.

ANSWER: A D

Explanation:

SAP recommends using meaningful, clear, and self-explanatory API names because an API name is a key part of its discoverability and usability. Consumers should be able to understand the business purpose and scope of an API from its name without needing to inspect its implementation. Clear, consistent naming also supports easier searching, governance, and reuse in an API catalog or developer portal. SAP's API design guidance emphasizes understandable, business-oriented APIs that can be readily consumed by application developers and integration teams.

Providing good API documentation is equally essential when an API is published. Documentation should give consumers the information needed to evaluate and successfully use the API, including its purpose, available operations, request and response structures, authentication requirements, error handling, and practical usage examples. Complete documentation reduces onboarding time, promotes correct consumption, and makes an API more reliable as a reusable product. These practices align with SAP API design and API management guidance, which treats clarity, discoverability, and consumer-focused documentation as core elements of a well-managed API. See the SAP API Style Guide and SAP API Management documentation.

QUESTION NO: 14

In an SAP Business Application Studio project, what is the purpose of running the `cf push` command?

A. It updates the service instances of the services defined in the service-manifest.yml file.

B. It creates the service instances of the services defined in the service-manifest.yml file.

C. It deploys the application modules defined in the manifest.yaml file into the Cloud Foundry account.

ANSWER: C

Explanation:

The `cf push` command deploys an application to the Cloud Foundry environment using the application deployment settings in a manifest file, such as `manifest.yaml`. When run from the appropriate project or module directory, the Cloud Foundry CLI packages the application source, uploads it to the targeted Cloud Foundry account and space, creates or updates the application, and applies manifest-defined settings such as the application name, memory allocation, disk quota, buildpack, routes, environment variables, and service bindings. It then stages and starts the application unless configuration instructs otherwise. In a Business Application Studio development workflow, this is a standard way to deploy a Cloud Foundry application module directly from the workspace after logging in and targeting the intended org and space. The manifest acts as deployment metadata, allowing the deployment configuration to be kept with the application source and used consistently across deployments. Cloud Foundry documents `cf push` as the command for pushing an app, with manifest properties supplying its deployment configuration. SAP Business Application Studio supports Cloud Foundry development and deployment through the Cloud Foundry CLI. See the Cloud Foundry CLI cf push reference and SAP's Business Application Studio Cloud Foundry development documentation.