

ACD300^{Q&As}

Appian Certified Lead Developer

Pass Appian ACD300 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.certbus.com/acd300.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Appian
Official Exam Center

-  **Instant Download** After Purchase
-  **100% Money Back** Guarantee
-  **365 Days** Free Update
-  **800,000+** Satisfied Customers



QUESTION 1

As part of an upcoming release of an application, a new nullable field is added to a table that contains customer data. The new field is used by a report in the upcoming release, and is calculated using data from another table.

Which two actions should you consider when creating the script to add the new field?

- A. Create a script that adds the field and leaves it null.
- B. Create a rollback script that removes the field.
- C. Create a script that adds the field and then populate it
- D. Create a rollback script that clears the data from the field
- E. Add a view that joins the customer data to the data used in calculation

Correct Answer: BC

When creating a script to add a new field to a table, you should consider two actions:

Create a rollback script that removes the field. A rollback script is a script that can undo the changes made by the original script, in case something goes wrong or the changes need to be reverted. A rollback script is a good practice to have,

as it can help to restore the previous state of the database and avoid any errors or inconsistencies. In this case, the rollback script should remove the new field from the table, and any other changes that were made by the original script.

Create a script that adds the field and then populate it. A script that adds the field and then populate it is a script that can create the new field in the table, and then fill it with data from another table or source. This way, you can ensure that the

new field has valid and consistent data, and that it can be used by the report in the upcoming release. In this case, the script should add the new field to the customer table, and then populate it with data from another table that contains the

data used in the calculation.

QUESTION 2

You are asked to design a case management system for a client in addition to storing some basic metadata about a case, one of the client's requirements is the ability for users to update a case. The client would like any user in their organization of 500 people to be able to make these updates. The users are all based in the company's headquarters, and there will be frequent cases where users are attempting to edit the same case. The client wants to ensure no information is lost when these edits occur and does not want the solution to burden their process administrators with any additional effort.

Which data locking approach should you recommend?

- A. Allow edits without locking the case GDI
- B. Use the database to implement two-level pessimistic locking.
- C. Add a version annotation to the case COT to manage view locking

D. Design a process report and query to determine who opened the edit form first

Correct Answer: C

The @Version annotation is a feature of Appian that allows for optimistic locking of CDTs. Optimistic locking assumes that concurrent updates to the same data are rare and does not lock the data until it is saved. If two users try to save changes to the same data, the user who saves first will succeed, while the user who saves second will get an error message and will have to resolve the conflict manually. This approach is suitable for the client's requirement, as it allows any user to update a case without locking it, ensures no information is lost when concurrent edits occur, and does not require any additional effort from the process administrators. Verified References: [Appian Documentation], section "Optimistic Locking".

QUESTION 3

You are required to configure a connection so that Jira can inform Appian when specific tickets change (using webhook).

Which three required steps will allow you to connect both systems?

- A. Create a Web API object and set up the correct security.
- B. Configure the connection in Jira specifying the URL and credentials
- C. Create a new API Key and associate a service account
- D. Give the service account system administrator privileges
- E. Create an integration object from Appian to Jira to periodically check the ticket status

Correct Answer: ABC

The three required steps that will allow you to connect both systems are:

A. Create a Web API object and set up the correct security. This will allow you to define an endpoint in Appian that can receive requests from Jira via webhook. You will also need to configure the security settings for the Web API object, such as authentication method, allowed origins, and access control. B. Configure the connection in Jira specifying the URL and credentials. This will allow you to set up a webhook in Jira that can send requests to Appian when specific tickets change. You will need to specify the URL of the Web API object in Appian, as well as any credentials required for authentication. C. Create a new API Key and associate a service account. This will allow you to generate a unique token that can be used for authentication between Jira and Appian. You will also need to create a service account in Appian that has permissions to access or update data related to Jira tickets. The other options are incorrect for the following reasons:

D. Give the service account system administrator privileges. This is not required and could pose a security risk, as giving system administrator privileges to a service account could allow it to perform actions that are not related to Jira tickets, such as modifying system settings or accessing sensitive data. E. Create an integration object from Appian to Jira to periodically check the ticket status. This is not required and could cause unnecessary overhead, as creating an integration object from Appian to Jira would involve polling Jira for ticket status changes, which could consume more resources than using webhook notifications. Verified References: Appian Documentation, section "Web API" and "API Keys".

QUESTION 4

You are presented with the following application requirement:

Users must be able to navigate throughout the application while maintaining complete visibility in the application structure, and easily navigate to previous locations.

Which Appian Interface Pattern would you recommend?

- A. Use Bullous as Cards pattern on the home page lo prominently display application choices.
- B. Implement an Activity History pattern to track an organizations activity measures.
- C. implement a drilldown report pattern to show detailed information about report data.
- D. Include a breadcrumbs pattern on applicable inert aces to show the organizational hierarchy

Correct Answer: C

To meet the application requirement of allowing users to view summary and detailed information about report data, you should implement a drilldown report pattern to show detailed information about report data. A drilldown report pattern is a user interface component that displays data in a hierarchical structure, and allows users to expand or collapse different levels of data. For example, if the user is viewing a sales report by region, the drilldown report pattern could show something like "North America > USA > California > Los Angeles". The user can click on any level of data to see more or less details. This way, the user can see both summary and detailed information about report data, and explore different aspects of the data. The other options are not as effective. Option A, using Tiles as Cards pattern on the home page to prominently display application choices, would provide a way for users to access different parts of the application from the home page, but it would not show summary or detailed information about report data. Option B, implementing an Activity History pattern to track an organization's activity measures, would provide a way for users to see the recent actions performed by themselves or others in the application, but it would not show summary or detailed information about report data. Option D, including a breadcrumbs pattern on applicable interfaces to show the organizational hierarchy, would provide a way for users to see where they are in the application, and easily go back to any previous level by clicking on the corresponding link, but it would not show summary or detailed information about report data.

QUESTION 5

You have created a Web API in Appian. with the following URL to call it:

[https://exampleappiancloud.com/suite/webapi/usef_managefnt/ users ?username=\)john.smith](https://exampleappiancloud.com/suite/webapi/usef_managefnt/ users ?username=)john.smith).

Which is the connect syntax forreferring to the user name parameter\

- A. httpirequest.queryParameters users username
- B. httpirequest usees username
- C. httpirequest formData username
- D. httpirequest queryParameters.username

Correct Answer: D

The correct syntax for referring to the username parameter in the Web API URL is `httpirequest.queryParameters.username`. This syntax allows you to access the value of the username parameter that is passed in the query string of the URL after the question mark (?). For example, if the URL is https://exampleappiancloud.com/suite/webapi/user_management/users?username=john.smith, then

httprequest.queryParameters.username will return john.smith. Verified References: Appian Documentation, section "Web API".

QUESTION 6

Your Appian project just went live with the following environment setup; DEV > TEST (SIT/DAT) > PROD

Your client is considering adding a support team to manage production defects and minor enhancements, while the original development team focuses on Phase 2 Your client is asking you for a new environment strategy that will have the least impact on Phase 2 development work.

Which option involves the lowest additional server cost and the least code retrofit effort?

- A. Phase 2 development work steam: DEV > TEST (SIT) > STAGE (UAT) > PROO Production support work stream DEV > TEST2 (SIT/UAT)>PROO
- B. Phase 2 development work Stream: DEV > TEST (SIT) > STAGE (UAT) > PROO Production support work stream DEV2 > STAGE (S1T/UAT) > PROD
- C. Phase 2 development work stream: DEV > TEST (SIT/UAT) >PROD Production support work stream DEV > TEST2 (SIT/UAT) > PROO
- D. Phase 2 development work stream: OEV > TEST (Srr/DAT) > PROO Production support work stream. DEV2 > TEST (SIT/UAT) > PROD

Correct Answer: B

The option B involves the lowest additional server cost and the least code retrofit effort, as it only requires one additional environment (DEV2) for the production support work stream. The production support work stream can use the existing STAGE environment for testing and user acceptance testing, as it is shared with the phase 2 development work stream. This way, there is no need to create a separate TEST2 environment or to retrofit any code from TEST to STAGE or from STAGE to PROD. Verified References: [Appian Certified Lead Developer study guide], page 16, section "Environment Strategy".

QUESTION 7

Your application contains a process model that is scheduled to run daily at a certain time, which kicks off a user input task to a specified user on the 1ST time zone for morning data collection The time zone is set to the (default) pm!timezone.

In this situation, what does the pm!tinezone reflect?

- A. The time zone of the server where Applan is intuited
- B. The line zone of the user who most recently published the process model
- C. The default time zone for the environment as specified in the Administration Console
- D. The time zone of the user who is completing the input task.

Correct Answer: C

In this situation, pm!timezone reflects the default time zone for the environment as specified in the Administration

Console. `pm!timezone` is a process variable that returns the time zone of the process. If the time zone is not explicitly set in the process model, then `pm!timezone` returns the default time zone for the environment, which can be configured in the Administration Console. In this case, the time zone is set to the (default) `pm!timezone`, which means that the process model does not have a specific time zone, and therefore uses the default time zone for the environment. The other options are not correct. Option A, the time zone of the server where Appian is installed, is not what `pm!timezone` reflects, as the server time zone may not be the same as the default time zone for the environment. Option B, the time zone of the user who most recently published the process model, is not what `pm!timezone` reflects, as the user's time zone may not be the same as the default time zone for the environment. Option D, the time zone of the user who is completing the input task, is not what `pm!timezone` reflects, as the user's time zone may not be the same as the default time zone for the environment.

QUESTION 8

You are running an inspection as a part of the first deployment process from TEST to PROD. You receive a notice that one of your objects will not deploy because it is dependent on an object from an application owned by a separate team.

What should be your next step?

- A. Create your own object with the same code base, replace the dependent object in the application, and deploy to PROD.
- B. Halt the production deployment and contact the other team for guidance on promoting the object to PROD
- C. Check the dependencies of the necessary object. Deploy to PROD if there are few dependencies and it is low risk
- D. Push a functionally viable package to PROD without the dependencies, and plan the rest of the deployment accordingly with the other team's constraints

Correct Answer: B

Deploying an object that is dependent on another object from a different application can cause errors and inconsistencies in the production environment. The best practice is to halt the production deployment and contact the other team for guidance on how to promote the object to PROD. The other team may have a different deployment schedule, or they may have some dependencies or customizations that need to be considered. By communicating with the other team, you can ensure that the object is deployed in a safe and coordinated manner, and avoid any potential conflicts or issues. Verified References: [Appian Deployment Guide], [Appian Best Practices]

QUESTION 9

You are tasked to build a large scale acquisition application for a prominent customer. The acquisition process tracks the time it takes to fulfill a purchase request with an award.

The customer has structured the contract so that there are multiple application dev teams.

How should you design for multiple processes and forms, while minimizing repeated code?

- A. Create a Center of Excellence (CoE)
- B. Create a common objects application.
- C. Create a Scrum of Scrums sprint meeting for the team leads
- D. Create duplicate processes and forms as needed

Correct Answer: B

To build a large scale acquisition application for a prominent customer, you should design for multiple processes and forms, while minimizing repeated code. One way to do this is to create a common objects application, which is a shared application that contains reusable components, such as rules, constants, interfaces, integrations, or data types, that can be used by multiple applications. This way, you can avoid duplication and inconsistency of code, and make it easier to maintain and update your applications. You can also use the common objects application to define common standards and best practices for your application development teams, such as naming conventions, coding styles, or documentation guidelines. Verified References: [Appian Best Practices], [Appian Design Guidance]

QUESTION 10

HOTSPOT

For each requirement, match the most appropriate approach to creating or utilizing plug-ins Each approach will be used once.

Note: To change your responses, you may deselect your response by clicking the blank space at the top of the selection list.

Hot Area:

Read barcode values from images containing barcodes and QR codes.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Display an externally hosted geolocation/mapping application's interface within Appian to allow users of Appian to see where a customer (stored within Appian) is located.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Display an externally hosted geolocation/mapping application's interface within Appian to allow users of Appian to select where a customer is located and store the selected address in Appian.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Generate a barcode image file based on values entered by users.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Correct Answer:

Read barcode values from images containing barcodes and QR codes.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Display an externally hosted geolocation/mapping application's interface within Appian to allow users of Appian to see where a customer (stored within Appian) is located.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Display an externally hosted geolocation/mapping application's interface within Appian to allow users of Appian to select where a customer is located and store the selected address in Appian.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Generate a barcode image file based on values entered by users.

Select a match:

Web-content field
Component plug-in
Smart Service plug-in
Function plug-in

Requirement: Read barcode values from images containing barcodes and QR codes. Correct approach: C. Smart Service plug-in
 Exact explanation of correct approach taken from Appian Documentation: A smart service plug-in is a type of plug-in that allows you to create custom smart services that can be used in process models. A smart service can perform complex logic, interact with external systems, or manipulate data in Appian. A smart service plug-in can also leverage Java code to implement the functionality of the smart service. A smart service plug-in would be suitable for reading barcode values from images, as it can use Java libraries or APIs that can scan and decode barcodes and QR codes from image files. A smart service plug-in can also return the barcode values as outputs that can be used by other nodes or processes in Appian. A smart service plug-in can also be configured with input parameters, such as the image file, the barcode type, or the output format, that can customize the behavior of the smart service. A smart service plug-in can also have error handling and logging features that can handle any exceptions or failures that might occur during the barcode reading process.

Requirement: Display an externally hosted geolocation mapping applications interface within Appian to allow users of Appian to see where a customer (stored within Appian) is located. Correct approach: A. Web-content field
 Exact explanation of correct approach taken from Appian Documentation: A web-content field is a type of user interface component that allows you to display web content from an external source in a SAIL interface. A web-content field would be suitable for displaying an externally hosted geolocation mapping applications interface, as it can embed the web content in an iframe and render it within the Appian interface. You can also pass parameters to the web content, such as the customer's location, using the url parameter of the web-content field. A web-content field can also interact with other components in the Appian interface, such as buttons, grids, or forms, using the postMessage API. This way, you can create a seamless user experience that integrates the external geolocation mapping applications interface with the Appian functionality.

Requirement: Display an externally hosted geolocation mapping applications

interface within Appian to allow users of Appian to select where a customer is located and store the selected address in Appian. Correct approach: A. Web-content field and C. Smart Service plug-in Exact explanation of correct approach taken from Appian Documentation: A web- content field and a smart service plug-in would be suitable for displaying an externally hosted geolocation mapping applications interface within Appian to allow users of Appian to select where a customer is located and store the selected address in Appian. A web- content field would be suitable for displaying the external geolocation mapping applications interface, as explained above. A smart service plug-in would be suitable for storing the selected address in Appian, as it can use Java code to receive the address data from the web content, validate it, and write it to a data store entity or a process variable. Requirement: Generate a barcode image file based on values entered by users. Correct approach: B. Component plug-in Exact explanation of correct approach taken from Appian Documentation: A component plug-in is a type of plug-in that allows you to create custom user interface components that can be used in SAIL interfaces. A component plug-in can also leverage Java code to implement the functionality of the component. A component plug-in would be suitable for generating a barcode image file, as it can use Java libraries or APIs that can encode values into barcode formats and generate image files. A component plug-in can also display the barcode image file in the Appian interface and allow users to download or print it. A component plug-in can also interact with other components in the Appian interface, such as text fields, buttons, or forms, using the `!refreshVariable()` function. This way, you can create a dynamic user experience that updates the barcode image file based on the values entered by users.

[Latest ACD300 Dumps](#)

[ACD300 Study Guide](#)

[ACD300 Exam Questions](#)