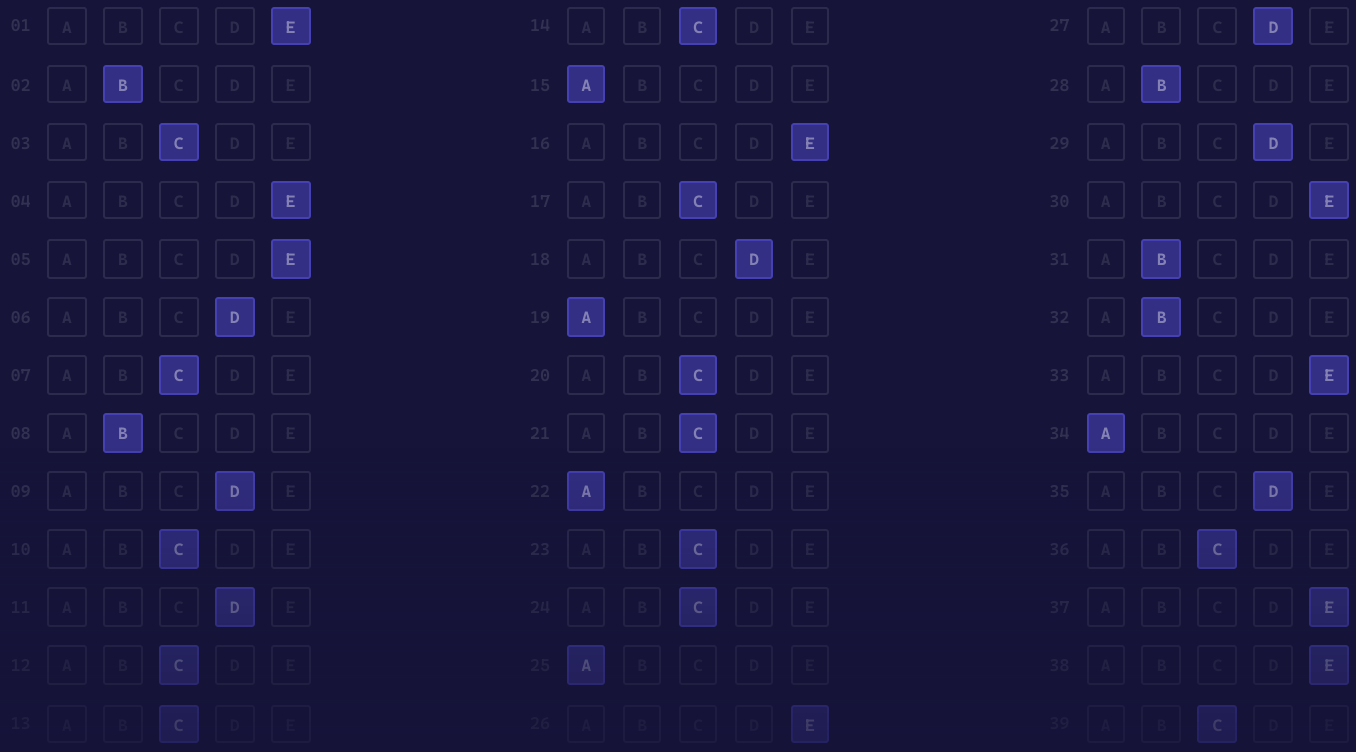
MOMETRIX
SMARTER PREP. BETTER RESULTS.




PRACTICE QUESTIONS

## OutSystems

# WS11S

Web Developer Specialist (ES)

| EDITION | QUESTIONS | FORMAT |
| --- | --- | --- |
| Demo | 9 | Limited Edition PDF |

WEB www.mometrix.net

SUPPORT support@mometrix.net

# Before you begin

## What this is

9 practice questions for OutSystems `WS11S`, each with its answer key and an explanation. Answer a question before you read its key — the explanation is worth more once you have committed.

## If something looks wrong

Send us three things and we will check it:

- the exam code, `WS11S`
- the question number
- the email address on your order

Write to **support@mometrix.net**. We reply within one business day.

mometrix.net

**QUESTION 1** SINGLE CHOICE

An OutSystems Web Developer Specialist needs to ensure a screen loads efficiently when fetching a large dataset from the database. Which approach is the recommended best practice in OutSystems to optimize data retrieval on screen preparation?

A Retrieve all records in an Aggregate and store them in a local variable on screen load

**B Use a paginated or incremental data fetch pattern (e.g., server-side pagination with a Data Action) so only the visible records are fetched**

C Disable preparation caching on the screen to always re-query the database

D Use a Client Action with an Advanced Query that selects all rows into a List on the client side

**ANSWER B**

**WHY** OutSystems best practices for screen performance recommend fetching only the data the user needs to see, using pagination or incremental loading, rather than retrieving the full dataset on preparation. Loading everything into a local variable causes unnecessary memory and network use, while client-side selection defeats server-side optimization.

**QUESTION 2** SINGLE CHOICE

Which artifact in Service Studio represents the entry point that end users navigate to in order to access a web application module's screens?

A The Module's properties page

**B The Screen flow entry point defined in the Flow of the main screen**

C The Application's consumer key

D The Module's dependencies list

**ANSWER B**

**WHY** In an OutSystems web module, the entry point for end users is the Screen flow's start node (Entry point) defined in the main screen's flow. Publishing the module makes this entry point URL available to consumers.

**QUESTION 3** SINGLE CHOICE

A developer is building a form in OutSystems and needs to provide real-time inline feedback to the user when an input value is invalid, without waiting for a server roundtrip. Which OutSystems UI capability supports client-side input feedback out of the box?

A A Server Action triggered by the OnChange event of the input widget

**B The built-in validation and feedback provided by the Form pattern combined with input widgets' Mandatory and Validation properties**

C A custom JavaScript library loaded from a third-party CDN

D A Database Trigger that re-evaluates the screen on save

**ANSWER B**

**WHY** OutForms in the OutSystems UI framework provide built-in client-side validation and inline feedback through the Mandatory and Validation properties on input widgets, so invalid values are surfaced immediately without a server roundtrip. A server action adds latency, and custom JavaScript is not the recommended default approach.

**QUESTION 4** SINGLE CHOICE

When configuring a web screen in OutSystems that should only be accessible by authenticated users with a specific role, where is this access restriction configured?

**A In the screen's Permission property under the Properties pane, selecting the required role**

B In a web flow's Exception Handler

C In the application's Service Center deployment configuration

D In the database's user roles table

**ANSWER A**

**WHY** OutSystems screens expose a Permission property where the developer assigns one or more roles. Only users who are members of at least one granted role can access the screen; otherwise they are redirected to the login flow.

**QUESTION 5** SINGLE CHOICE

Which of the following is a recommended OutSystems security best practice when exposing a web service or REST endpoint to external consumers?

A Disable HTTPS so internal services can call the endpoint faster

B Use Anonymous authentication for every endpoint to simplify integration

**C Apply the principle of least privilege by granting only the specific roles that need each endpoint and validating inputs in Server Actions**

D Publish sensitive service endpoints on the same anonymous path as the public site

**ANSWER C**

**WHY** OutSystems security guidance recommends granting only the roles that strictly need access to each endpoint (least privilege) and validating all inputs in Server Actions. Anonymous and broad access, or disabling HTTPS, weaken the application's security posture and contradict OutSystems best practices.

**QUESTION 6** SINGLE CHOICE

An Entity has a Boolean attribute named IsActive used to mark soft-deleted rows. A developer wants the application to filter out inactive records by default in every Aggregate that uses this entity. What is the OutSystems-recommended way to enforce this default filter across the module?

A Rename the entity to remove inactive rows at the database level

B Manually add a 'IsActive = True' filter to every Aggregate on the entity

C Define a Calculated Attribute on the entity that always returns True, and use it as a default filter

**D Set the entity's Default Fetch Filter in Service Studio so that all Aggregates inherit it**

**ANSWER D**

**WHY** OutSystems lets developers define a Default Fetch Filter on an Entity that is automatically applied to every Aggregate that uses it. This is the recommended way to enforce a default condition such as 'IsActive = True' across the module without manually editing every Aggregate. Calculated attributes cannot be used as default filters, and renaming the entity is unrelated.

mometrix.net

**QUESTION 7** SINGLE CHOICE

A developer needs to validate two dependent fields on a screen — an end date must be greater than a start date — and show feedback without a server roundtrip. Which is the correct OutSystems approach?

**A** **Use a Client Action invoked from the End Date widget's OnChange event to compare the dates and assign the validation message**

B Validate the dates only inside a Database Trigger

C Run a scheduled timer that polls the database every second

D Add a custom CSS class that hides the End Date field

**ANSWER** **A**

**WHY** OutSystems Client Actions execute in the browser and can be wired to widget events such as OnChange. Comparing the two date values and assigning a validation message inside a Client Action provides immediate feedback without a server roundtrip. Server triggers, timers, and CSS hacks are not appropriate for this kind of validation.

**QUESTION 8** SINGLE CHOICE

In Service Studio, which action must a developer take to expose a previously reusable screen so that it can be navigated to directly by an end user through a URL?

**A** **Mark the screen as Expose as Web Screen and publish the module**

B Rename the screen to start with an uppercase letter

C Add a new database entity to the module

D Move the screen into a Library module

**ANSWER** **A**

**WHY** A screen that is only reused by other screens is not directly navigable by end users. To expose it as a web screen accessed via URL, the developer toggles 'Expose as Web Screen' on the screen properties and publishes the module so the entry point URL is generated.

**QUESTION 9** SINGLE CHOICE

A developer wants to prevent a user from submitting a form when required fields are empty. Which OutSystems approach is the most appropriate for enforcing required-field rules before the form's submit action runs?

A Add a mandatory widget extension to each required field

**B Set the Required property to Yes on the input widgets and rely on built-in validation**

C Use a web block to hide the Save button when fields are empty

D Add a server action that checks for null values and returns an error string

**ANSWER B**

**WHY** In OutSystems, setting the Required property to Yes on an input widget is the platform's standard mechanism for enforcing mandatory input. The framework prevents the form from being submitted while a required field is empty and surfaces the built-in validation feedback. The other options describe patterns that are either not first-class platform features or do not reliably enforce the rule.

mometrix.net