70-486: Developing ASP.NET MVC Web Applications

The following tables show where changes to exam 70-486 have been made to include updates that relate to MVC 5, Visual Studio 2013, and Windows Azure tasks. These changes are effective as of April 30, 2014.

1. Design the application architecture

Tasks currently measured	Tasks Added/Changed post April 2014
Plan the application layers	No Change
Plan data access; plan for separation of	
concerns; appropriate use of models, views, and	
controllers; choose between client-side and	
server side processing; design for scalability	
Design a distributed application	No Change
Design a hybrid application (on premise vs. off	
premise, including Windows Azure); plan for	
session management in a distributed	
environment; plan web farms	
Design and implement the Windows	No Change
Azure role life cycle	
Identify and implement Start, Run, and Stop	
events; identify startup tasks (IIS configuration	
[app pool], registry configuration, third-party	
tools)	
Configure state management	Modified subtask:
Choose a state management mechanism (in-	Choose a state management mechanism (in-process and
process and out of process state management);	out of process state management, ViewState)
plan for scalability; use cookies or local storage	
to maintain state; apply configuration settings in	
web.config file; implement sessionless state (for	
example, QueryString)	Add ad authoriu
Design a caching strategy	Added subtask:
Implement page output caching (performance	Implement Azure caching
oriented); implement data caching; implement	
HTTP caching	Added subtack
Design and implement a Web Socket	Added subtask:
strategy	Implement SignalR

Read and write string and binary data	
asynchronously (long-running data transfers);	
choose a connection loss strategy; decide a	
strategy for when to use Web Sockets	
Design HTTP modules and handlers	No Change
Design HTTP modules and handlers Implement synchronous and asynchronous	No Change
	No Change

2. Design the user experience

Tasks currently measured	Tasks Added/Changed post April 2014
Apply the user interface design for a	No Change
web application	
Create and apply styles by using CSS; structure	
and lay out the user interface by using HTML;	
implement dynamic page content based on a	
design	
Design and implement UI behavior	No Change
Implement client validation; use JavaScript and	
the DOM to control application behavior; extend	
objects by using prototypal inheritance; use AJAX to make partial page updates; implement	
the UI by using JQuery	
Compose the UI layout of an application	No Change
Implement partials for reuse in different areas of	The change
the application; design and implement pages by	
using Razor templates (Razor view engine);	
design layouts to provide visual structure;	
implement master/application pages	
Enhance application behavior and style	No Change
based on browser feature detection	
Detect browser features and capabilities; create	
a web application that runs across multiple	
browsers and mobile devices; enhance	
application behavior and style by using vendor-	
specific extensions, for example, CSS	
Plan an adaptive UI layout	No Change
Plan for running applications in browsers on	
multiple devices (screen resolution, CSS, HTML);	
plan for mobile web applications	

3. Develop the user experience

Tasks currently measured	Tasks Added/Changed post April 2014
Plan for search engine optimization and	No Change
accessibility	_
Use analytical tools to parse HTML; view and	
evaluate conceptual structure by using plugs-in	
for browsers; write semantic markup (HTML5	
and ARIA) for accessibility, for example, screen	
readers	
Plan and implement globalization and	No Change
localization	
Plan a localization strategy; create and apply	
resources to UI including JavaScript resources;	
set cultures; create satellite resource assemblies	Mandification to the sale.
Design and implement MVC controllers	Modified subtask:
and actions	Apply authorization attributes, global filters, and authentication filters
Apply authorization attributes and global filters;	authentication liners
implement action behaviors; implement action results; implement model binding	Added subtask:
results, implement model binding	Specify an override filter
Design and implement routes	No Change
Define a route to handle a URL pattern; apply	
route constraints; ignore URL patterns; add	
custom route parameters; define areas	
Control application behavior by using	No Change
MVC extensibility points	
Implement MVC filters and controller factories;	
control application behavior by using action	
results, viewengines, model binders, and route	
handlers	
Reduce network bandwidth	No Change
Bundle and minify scripts (CSS and JavaScript);	
compress and decompress data (using	
gzip/deflate; storage); plan a content delivery network (CDN) strategy, for example, Windows Azure CDN	

4. Troubleshoot and debug web applications

Tasks currently measured	Tasks Added/Changed post April 2014
Prevent and troubleshoot runtime issues	No Change
Troubleshoot performance, security, and errors; implement tracing, logging (including using attributes for logging), and debugging (including IntelliTrace); enforce conditions by using code contracts; enable and configure health monitoring (including Performance Monitor)	
Design an exception handling strategy	No Change
Handle exceptions across multiple layers; display custom error pages using global.asax or creating your own HTTPHandler or set web.config attributes; handle first chance exceptions	
Test a web application	Modified subtask:
Create and run unit tests, for example, use the Assert class, create mocks; create and run web	Create and run web tests (including using Browser Link)
tests	Added subtask:
	Debug a web application in multiple browsers and mobile emulators
Debug a Windows Azure application	Modified subtask:
Collect diagnostic information by using Windows Azure Diagnostics API Implement on demand vs. scheduled; choose log types, for	Debug a Windows Azure application by using IntelliTrace, Remote Desktop Protocol (RDP), and remote debugging
example, event logs, performance counters, and crash dumps; debug a Windows Azure application by using IntelliTrace and Remote Desktop Protocol (RDP)	Added subtask: Interact directly with remote Windows Azure websites using Server Explorer

5. Design and implement security

Tasks currently measured	Tasks Added/Changed post April 2014
Configure authentication	Added subtask:
Authenticate users; enforce authentication settings; choose between Windows, Forms, and custom authentication; manage user session by using cookies; configure membership providers;	Configure ASP.NET Identity
create custom membership providers	
Configure and apply authorization Create roles; authorize roles by using configuration; authorize roles programmatically; create custom role providers; implement WCF service authorization	No Change
Design and implement claims-based	Modified subtask:
authentication across federated identity stores Implement federated authentication by using Windows Azure Access Control Service; create a custom security token by using Windows Identity Foundation; handle token formats (for example, oAuth, OpenID, LiveID, and Facebook) for SAML and SWT tokens	Handle token formats (for example, oAuth, OpenID, Microsoft Account, Google, Twitter, and Facebook) for SAML and SWT tokens
Manage data integrity Apply encryption to application data; apply encryption to the configuration sections of an application; sign application data to prevent	No Change
Implement a secure site with ASP.NET Secure communication by applying SSL certificates; salt and hash passwords for storage; use HTML encoding to prevent cross-site scripting attacks (ANTI-XSS Library); implement deferred validation and handle unvalidated requests, for example, form, querystring, and URL; prevent SQL injection attacks by parameterizing queries; prevent cross-site request forgeries (XSRF)	No Change