Video 3.4

Implementing PUT

|  |
| --- |
| **Metadata**: Spot the problem, highlight it, and design the solution in 3 core steps  (To be covered in the video) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Problem / Solution (Not more than 50 words)** | **Step 1 (Not more than 10 words)** | **Step 2 (Not more than 10 words)** | **Step 3 (Not more than 10 words)** |
| In this video, we will implement Put verb and see how the data is passed at server to update an item and response comes back to client | Create a Put method in controller | Write logic and required code in put method | Explain it with the help of running and debugging |

|  |
| --- |
| **Script** the Video – Plan your narration (viewers will see and hear this) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Introduction** | | | |
| **No.** | **Action on Screen  (Code / Screenshots / One line about the action occurring on screen)** | **Narration**  **(The corresponding explanation to the Action on Screen)** | | |
| 1 | Replace the slide with a screenshot of the actual slide  C:\Data\Work\ForPackt\Deliverables\Draft-Scripts\Section 3\Video 3.4\images\341.png | | **Video Introduction**  In previous video, we implemented Post verb. Now it is time implement HTTP Put. |
| 2 |  | |  |

|  |  |  |
| --- | --- | --- |
| **Steps or Tasks**  (Refer to the Writing Guidelines- Script Best Practices) | | |
| **No.** | **Action on Screen  (Code / Screenshots / One line about the action occurring on screen)** | **Narration**  **(The corresponding explanation to the Action on Screen)** |
| 3 |  | Till now we have seen Get and Post implementation. In Get we passed data via URL and in Post the data was passed via request body |
| 4 |  | In Put we use a combination of both. The Identifier should be passed via URL and the object should be passed in request body. While using PUT for updating a resource, the complete object should be passed in formbody even if we are just updating a specific value in in the resource at server. There is another Verb for that purpose that we will see in coming video. |
| 5 | [HttpPut("{id:int}")]  public IActionResult Put(int id, [FromBody] Book book)  {  if (book == null)  {  return HttpBadRequest("Data is not received in proper format");  }  bool bookUpdated = this.repository.UpdateBook(book);  if (!bookUpdated)  {  return HttpBadRequest("Error while updating the book");  }    return new HttpOkResult();  } | I have already implemented Put method. Lets have look on it. Here we have two parameters one that we are reading from the url and another that is from request body similar to post.  Inside the method, first we are checking the input and if it is null returning bad request and once the item is updated, returning Ok. In case of an error, returning 400 again. |
| 6 |  | Let’s run the application and test the Put method  For simplicity we will fetch one item with author, update it manually then make a put request with that  So I fetched the book with id .. , now appending the title with *New* and executing the Put method.  We also need to send the content type in header as we sending the data in request body so lets add the header : content-type : application/json |
| 7 |  | [Debug]Now we can see that here we got the Id initialized with 4 and we have our book is initialized with updated value. Perfect!!  Lets continue and get it updated. So here we can see that we got 200 response it means the item got updated successfully |
| 8 |  | Let’s quickly verify it via a Get on that resource  And yes we have updated book at server. |

|  |  |
| --- | --- |
| **Summary** | |
|  | In this video, we have implemented Put key word which is used to update an resource |
| [Mandatory slide] – Next Video  C:\Data\Work\ForPackt\Deliverables\Draft-Scripts\Section 3\Video 3.4\images\343.png | In the next video, we will cover delete verb |