

Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;

import org.springframework.stereotype.Service;

import com.ashokit.bindings.Book;

@Service
public class BookService {

    public boolean saveBook(Book book) {
        return true;
    }

}

package com.ashokit.rest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;

import org.springframework.stereotype.Service;

import com.ashokit.bindings.Book;

@Service
public class BookService {

    public boolean saveBook(Book book) {
        return true;
    }

}

package com.ashokit.rest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;

import org.springframework.stereotype.Service;

import com.ashokit.bindings.Book;

@Service
public class BookService {

    public boolean saveBook(Book book) {
        return true;
    }

}

package com.ashokit.rest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");

MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =  
    MockMvcRequestBuilders.get("");  
  
MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;  
  
import org.springframework.stereotype.Service;  
  
import com.ashokit.bindings.Book;  
  
@Service  
public class BookService {  
  
    public boolean saveBook(Book book) {  
        return true;  
    }  
  
}
```

```
package com.ashokit.rest;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.bind.annotation.RequestBody;  
import org.springframework.web.bind.annotation.RestController;
```



Add

Yesterday's session : REST Controller Unit Testing



Ashok

Dec 2,

02-Dec-2020 c



02-Dec

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");
```

```
MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```



Ashok

Dec 1,

01-Dec-2020 C



01-Dec

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



12-JRTP @





Add class comment



Gang

Nov 2

From 23rd onv

1 class comm



Bijend

Sir, ple



Add



Asho

Nov 2

28-Nov-2020 C



Add



Kund

Nov 2

Sir,
please upload

Add

Yesterday's session : REST Controller Unit Testing

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");
```

```
MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

What is Mocking?

-> It is the process of creating substitute/alternate object for the

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```



12-JRTP @



3 class comments



Ganga

No att

Yesterday's session : REST Controller Unit Testing



Add

-> We can perform Unit testing for rest controller using MockMvc.

```
MockMvcRequestBuilders reqBuilder =
    MockMvcRequestBuilders.get("");
```

```
MvcResult result = mockMvc.perform(reqBuilder).andReturn();
```

-> We can validate rest api method results using response body conte

```
int status = result.getResponse().getStatus();
```

4 class comm



siri 20

Sir, Pl

What is Mocking?

-> It is the process of creating substitute/alternate object for the



Add

```
package com.ashokit.service;
```

```
import org.springframework.stereotype.Service;
```

```
import com.ashokit.bindings.Book;
```

```
@Service
```

```
public class BookService {
```

```
    public boolean saveBook(Book book) {
        return true;
    }
```

```
}
```

```
package com.ashokit.rest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.PostMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
```

```
import org.springframework.web.bind.annotation.RestController;
```

