Intermediate Java – Part 5

Yotam Avivi

Oct-Nov 2017

Rethink Possible®

REST API Methods in a Nutshell

HTTP Verb	Action	Example (URL + Body)	Response (Success)
POST	Create an entity	POST /students { “id” : “1”, “name” : “John Smit”}	201 Created
GET	Get an entity	GET /students/1	200 { “id” : “1”, “name” : “John Smit”}
PUT	Update an entity	PUT /students/1 { “id” : “1”, “name” : “John Smith”}	200
DELETE	Deletes an entity	DELETE /students/1	200

Prepare Project

1. Clone Project:

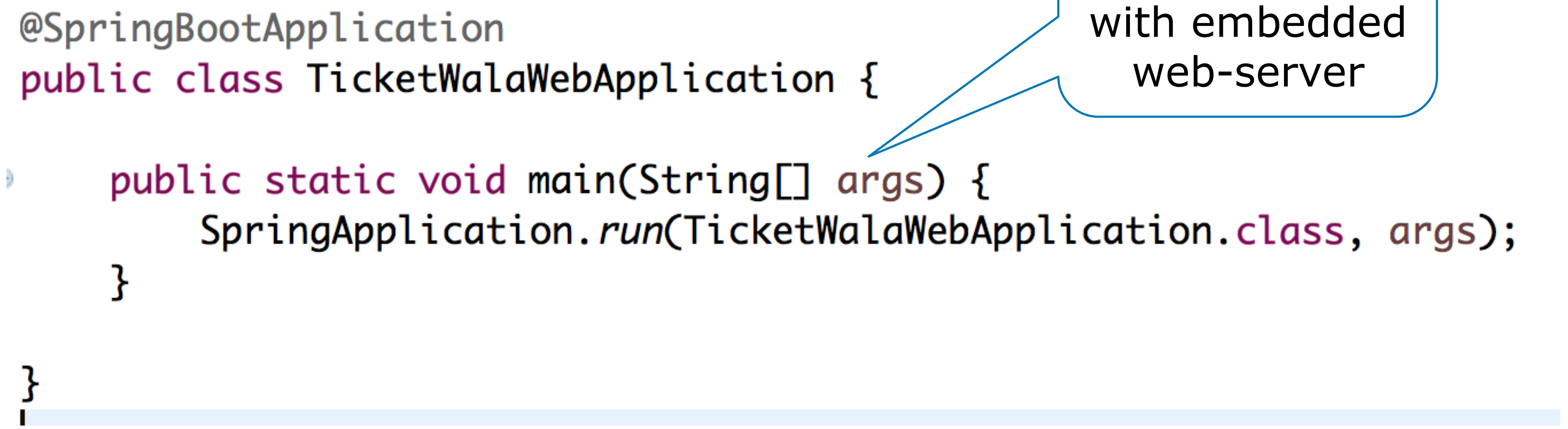
```
$ git clone https://github.com/devsketches/ticketwala.git
```

2. Set to Branch step11/cli_completed:

```
$ git checkout step11/cli_completed
```

3. Import Project to Eclipse

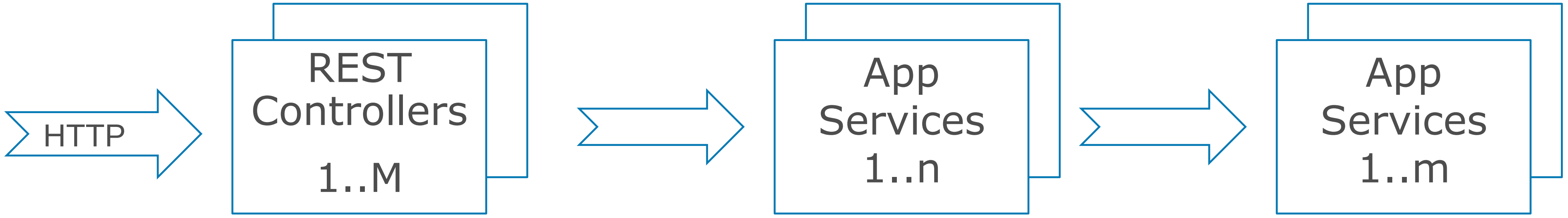
Bootstrapping a Spring Web App



```java
@SpringBootApplication
public class TicketWalaWebApplication {

    public static void main(String[] args) {
        SpringApplication.run(TicketWalaWebApplication.class, args);
    }

}
```

Will bootstrap an application with embedded web-server

Spring REST Controllers



Declaring a Spring REST Controller

Request URL

Request Method: GET

```
@RestController
public class TicketWalaController {

    @RequestMapping(value = "/", method = RequestMethod.GET)
    public String hello() {
        return "TicketWala Web Application";
    }

}
```

Declaring a Service

```
@Service
public class TicketWalaServiceImpl implements TicketWalaService {

...

}
```

Injecting a Service to Controller

```
@RestController
public class TicketWalaController {

    @Autowired
    TicketWalaService ticketWalaService;


    ...



}
```

Invoking the Injected Service

```
@RestController
public class TicketWalaController {

    @Autowired
    TicketWalaService ticketWalaService;

    @RequestMapping(value = "/movieshows", method = RequestMethod.GET)
    public List<MovieShow> getAllMovieShows() {
            return this.ticketWalaService.getMovieShows();
    }

}
```

Response converted to JSON Automatically

HTTP Responses & Path Parameters

```
@RestController
public class TicketWalaController {

	@Autowired
	TicketWalaService ticketWalaService;

	@RequestMapping(value = "/movieshow/{id}", method = RequestMethod.GET)
	public ResponseEntity<MovieShow> getMovieShow(@PathVariable("id") String id){

		MovieShow movieShow = this.ticketWalaService.getMovieShow(id);
		HttpStatus s = movieShow == null ? HttpStatus.NOT_FOUND : HttpStatus.OK;
		return new ResponseEntity<MovieShow>(movieShow, s);

	}
}
```

Extracting Path Patameters

POST Example: Posting a Map of Values

```
POST /postmymap
{
		"fname" : ”Avi",
		"lname" : ”Cohen”
}
```

```
@RequestMapping(value = "/postmymap", method = RequestMethod.POST)
public Map<String, String> postData(@RequestBody Map<String, String> map) {
    Map<String, String> res = new HashMap<String, String>();
    String fullName = map.get(“lname”) + “ ” + map.get(“fname”);
    res.put("status", "OK");
    return res;
}
```

Testing with REST Assured

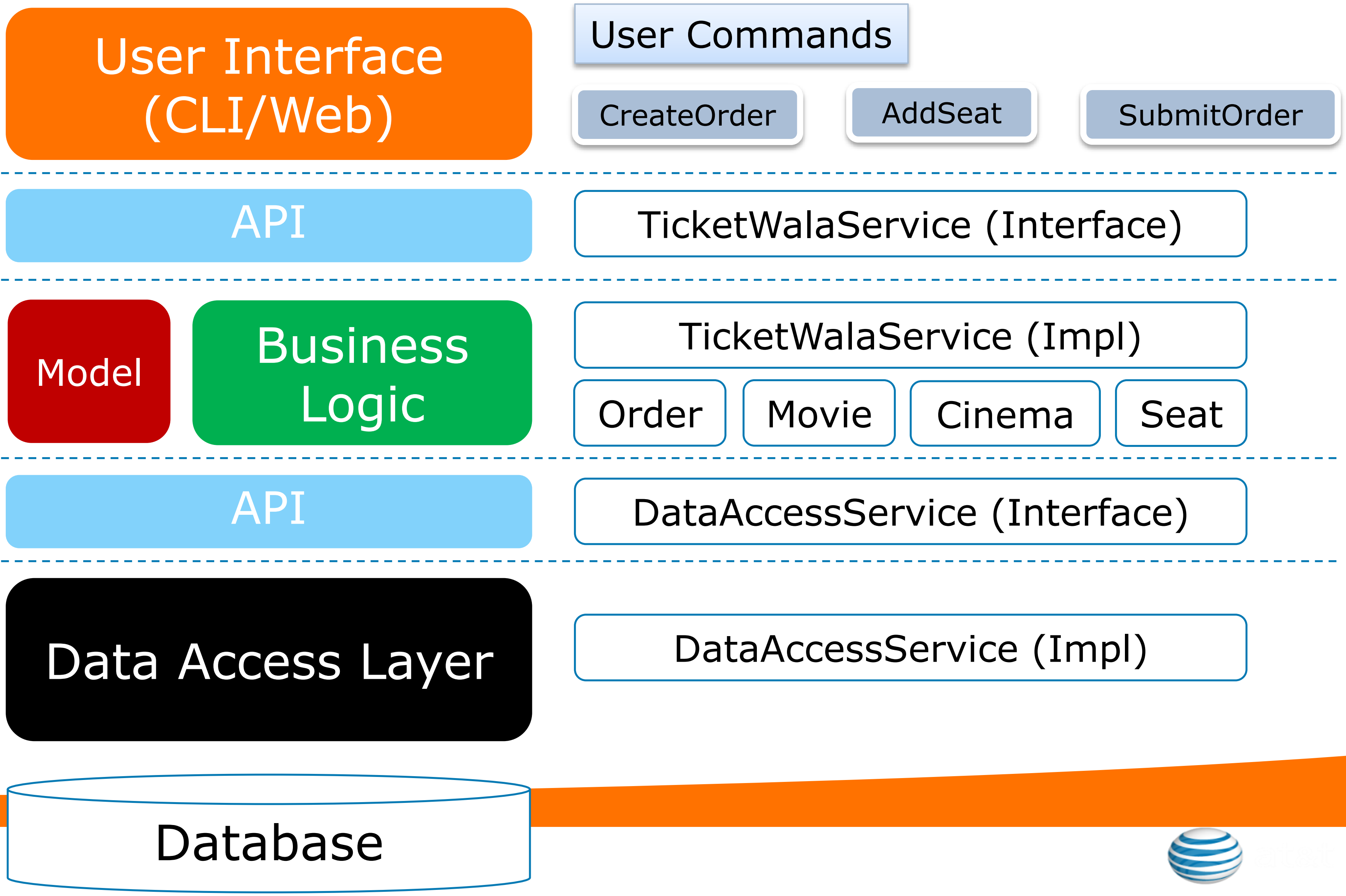
```
import static io.restassured.RestAssured.get;
import static org.hamcrest.Matchers.equalTo;


@Test
public void testGetMovieShow() {
     prepareDatabase();
     get("/movieshow/12345").
     then().body("id",equalTo("12345")).
     and().body("movieName", equalTo("Star Wars III"));
}
```

Project High Level Design in Layers

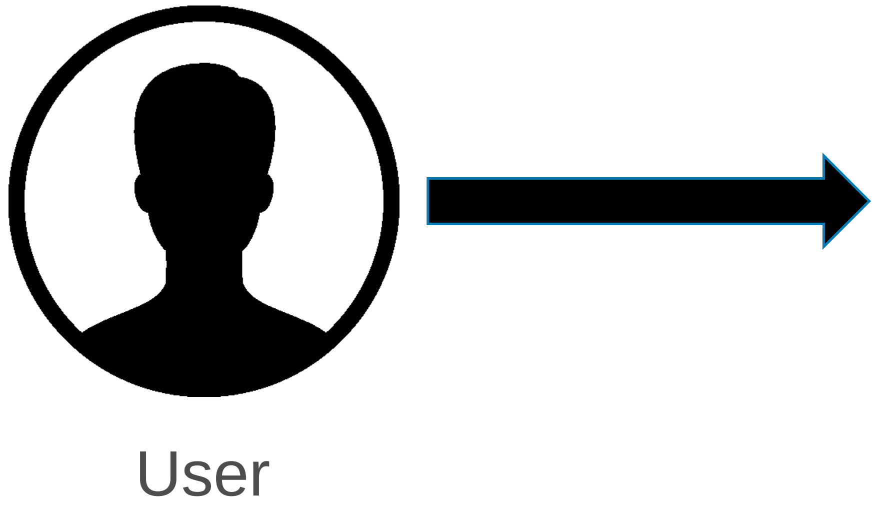
Translates user actions to Business Logic API calls

Accepts API calls and does actual business actions

Handles data persistency into database (create/get/delete/update)



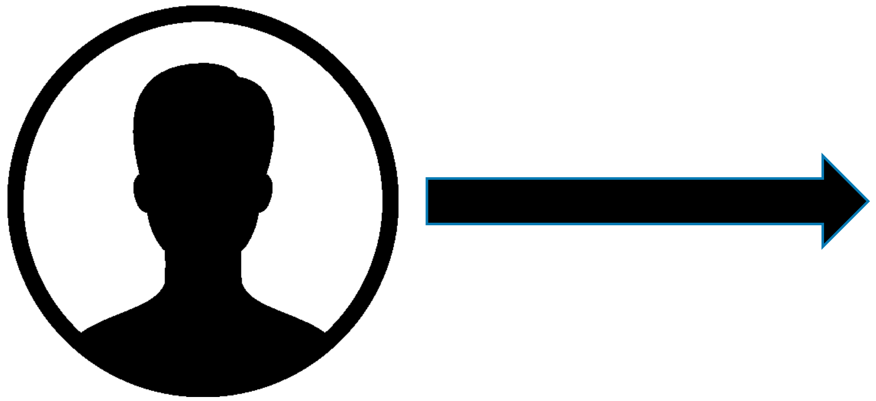
The TicketWalaService Interface



{ TicketWala API }

- Get All Movie Shows
- Get (Single) Movie Show
- Create a new Order (for a Movie Show)
- Add a Seat Ticket to an Order
- Submit Order

The TicketWalaService (Admin) Interface



Admin

{ TicketWala (Admin) API }

- Create Movie Show
- Delete Movie Show

Exercise - Implement all MovieShow Related REST API:

Implement all Movie Show Commands:

1. GET /movieshows
2. GET /movieshow/{id} ← Get Movie Show
3. POST /movieshow ← Create Movie Show
4. DELETE /movieshow/{id} ← Delete Movie Show
5. PUT /movieshow ← Update Movie Show