

Spring Boot

REST API

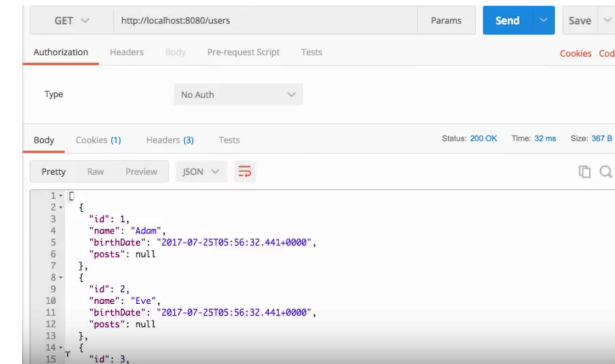
Introduction to REST

- **REST: Representational State Transfer**
 - Architectural style for the web (makes use of HTTP)
- **Key abstraction - Resource**
 - Todo Management Application
 - Examples: Users, Todos
 - Resource has **URI (Uniform Resource Identifier)**
 - `/users/Ranga` - (`/users/{id}`)
 - `/users/Ranga/todos` - (`/users/{id}/todos`)
 - `/users/Ranga/todos/1` - (`/users/{id}/todos/{id}`)
 - You can perform **ACTIONS** on resources:
 - Retrieve/Add/Update/Delete Todo
 - Retrieve/Add/Update/Delete User
 - Resource can have different **REPRESENTATIONS**
 - JSON, XML (JSON - most popular)

in28Minutes	Home	Todos	Logout
Your Todos			
Description	Target Date	Is it Done?	
Learn AWS	10/06/2032	false	<button>Update</button> <button>Delete</button>
Learn DevOps	10/06/2031	false	<button>Update</button> <button>Delete</button>
Learn React	10/06/2030	false	<button>Update</button> <button>Delete</button>
Learn Angular	10/06/2029	false	<button>Update</button> <button>Delete</button>

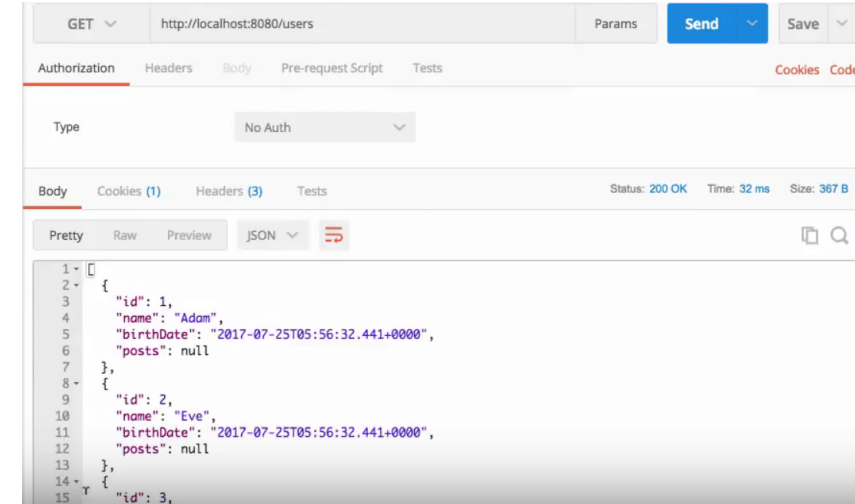
Request Methods for REST API

- **GET** - Retrieve details of a resource
- **POST** - Create a new resource
- **PUT** - Update an existing resource
- **PATCH** - Update part of a resource
- **DELETE** - Delete a resource



Response Status for REST API

- Return the **correct response status**
 - Resource is not found => 404
 - Server exception => 500
 - Validation error => 400
- **Important Response Statuses**
 - 200 — Success
 - 201 — Created
 - 204 — No Content
 - 401 — Unauthorized (when authorization fails)
 - 400 — Bad Request (such as validation error)
 - 404 — Resource Not Found
 - 500 — Server Error



Survey Questionnaire REST API

- Build a REST API for Survey Questionnaire
- **Key Resources:**
 - Surveys
 - Survey Questions
- **Key Details:**
 - Survey: id, title, description, question
 - Survey Questions: id, description, options, correctAnswer

```
localhost:8080/surveys/
[
  {
    "id": "Survey1",
    "title": "My Favorite Survey",
    "description": "Description of the Survey",
    "questions": [
      {
        "id": "Question1",
        "description": "Most Popular Cloud Platform Today",
        "correctAnswer": "AWS",
        "options": [
          "AWS",
          "Azure",
          "Google Cloud",
          "Oracle Cloud"
        ]
      },
      {
        "id": "Question2",
        "description": "Fastest Growing Cloud Platform",
        "correctAnswer": "Google Cloud",
        "options": [
          "AWS",
          "Azure",
          "Google Cloud",
          "Oracle Cloud"
        ]
      }
    ]
  },
  {
    "id": "Survey2",
    "title": "Another Survey",
    "description": "Another description",
    "questions": [
      {
        "id": "Question3",
        "description": "Another question description",
        "correctAnswer": "Google Cloud",
        "options": [
          "AWS",
          "Azure",
          "Google Cloud",
          "Oracle Cloud"
        ]
      }
    ]
  }
]
```

Survey Questionnaire REST API - Resources and Methods

- **Survey REST API:**
 - Retrieve All Surveys
 - GET /surveys
 - Retrieve Specific Survey
 - GET /surveys/{surveyId}
- **Survey Questions REST API:**
 - Retrieve Survey Questions
 - GET /surveys/{surveyId}/questions
 - Retrieve Specific Survey Question
 - GET /surveys/{surveyId}/questions/{questionId}
 - Add Survey Question
 - POST /surveys/{surveyId}/questions
 - Delete Survey Question
 - DELETE /surveys/{surveyId}/questions/{questionId}
 - Update Survey Question
 - PUT /surveys/{surveyId}/questions/{questionId}

```
localhost:8080/surveys/
[
  {
    "id": "Survey1",
    "title": "My Favorite Survey",
    "description": "Description of the Survey",
    "questions": [
      {
        "id": "Question1",
        "description": "Most Popular Cloud Platform Today",
        "correctAnswer": "AWS",
        "options": [
          "AWS",
          "Azure",
          "Google Cloud",
          "Oracle Cloud"
        ]
      },
      {
        "id": "Question2",
        "description": "Fastest Growing Cloud Platform",
        "correctAnswer": "Google Cloud",
        "options": [
          "AWS",
          "Azure",
          "Google Cloud",
          "Oracle Cloud"
        ]
      }
    ]
  }
]
```


Slides For Future

Constraints defined by REST

- **Client - Server** : Server (service provider) separated from client (service consumer)
 - **Benefits:** Loose coupling, Independent evolution of server and client (as new technologies emerge)
- Each service should be **stateless**
- Each Resource has a **resource identifier**
 - /users/Ranga - (/users/{id})
 - /users/Ranga/todos - (/users/{id}/todos)
 - /users/Ranga/todos/1 - (/users/{id}/todos/{id})
- **Caching response** should be possible
- Resource can have **multiple representations**
 - Resource can be modified through a message in any of these representations

