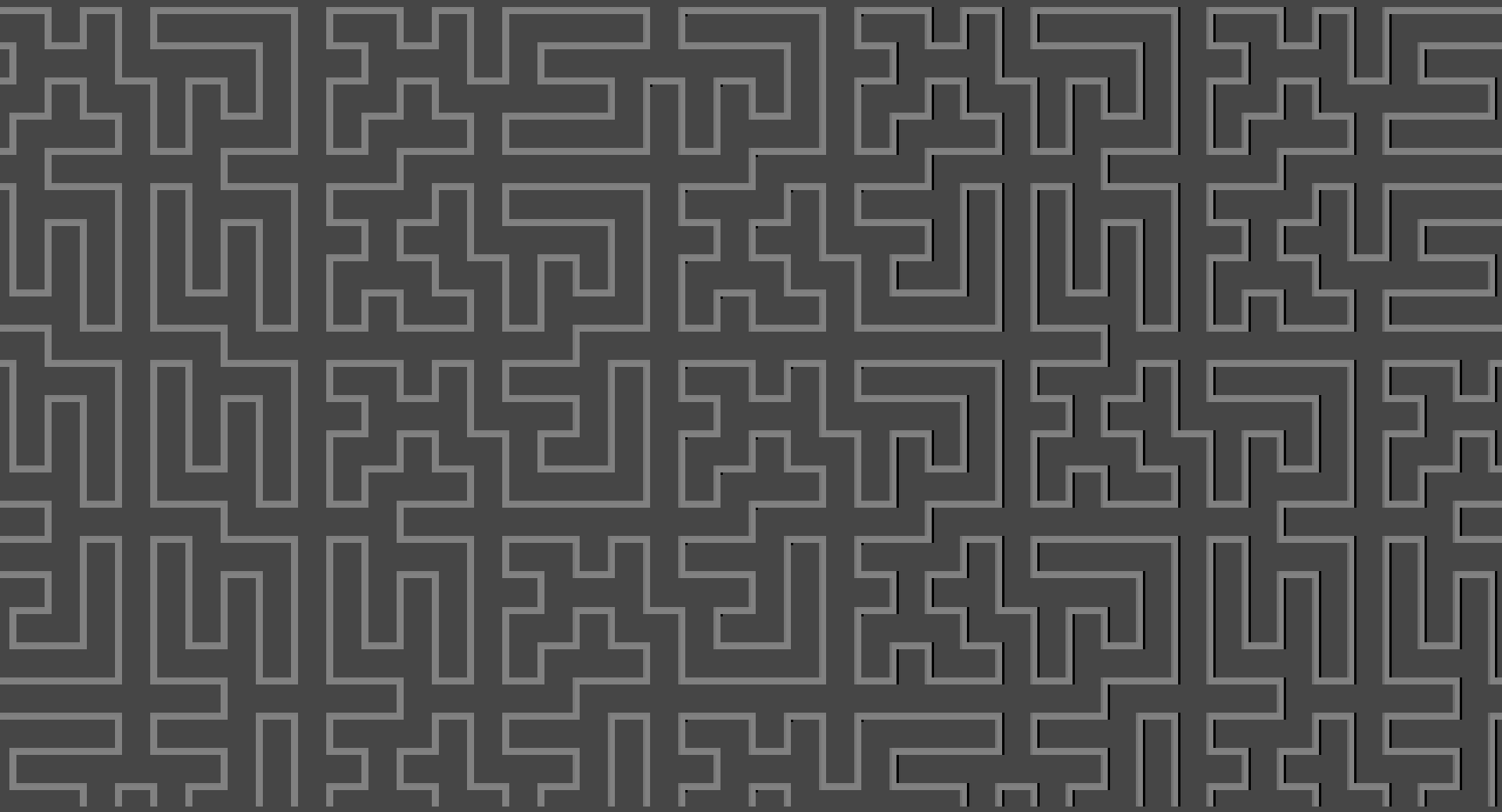


MOBILE DEVELOPMENT



Web Services and Interact with API

- 
- Web services
 - RESTful Web API
 - Retrofit
 - JSON
 - Demo
 - Exercise

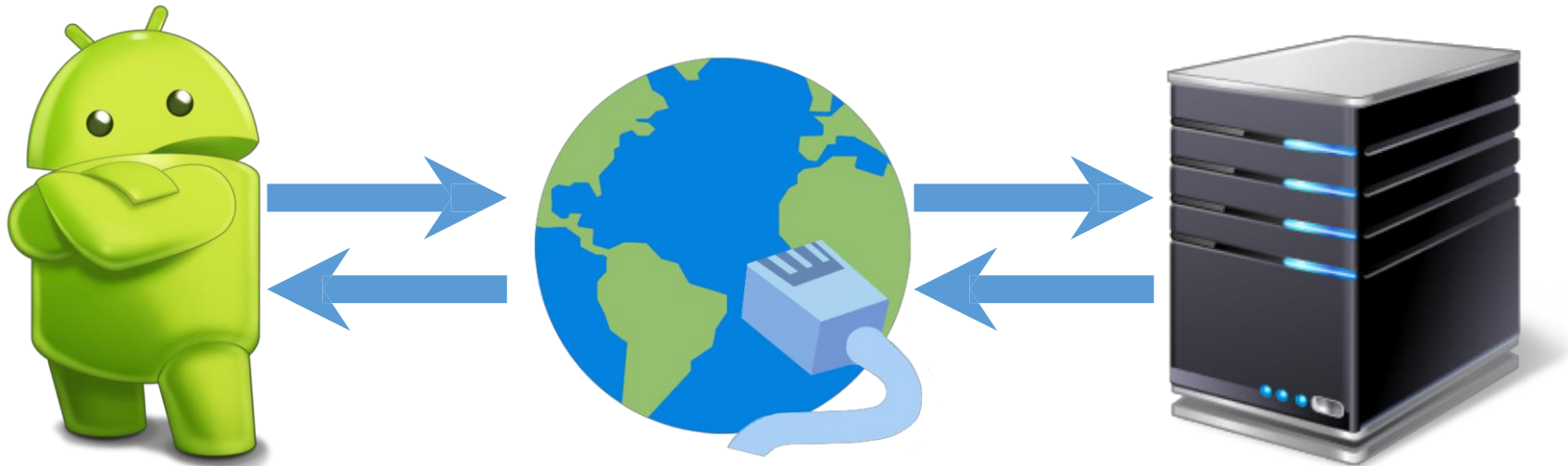
Web Services

- Expose a database / service via HTTP
- Different architectures
 - ▣ REST, RESTful
 - ▣ SOAP, Swagger etc.
- Basic idea is to manipulate a URL to allow a client application to make different requests
- Get a response in XML/JSON

Web Services to Access Data

Web Service - HTTP

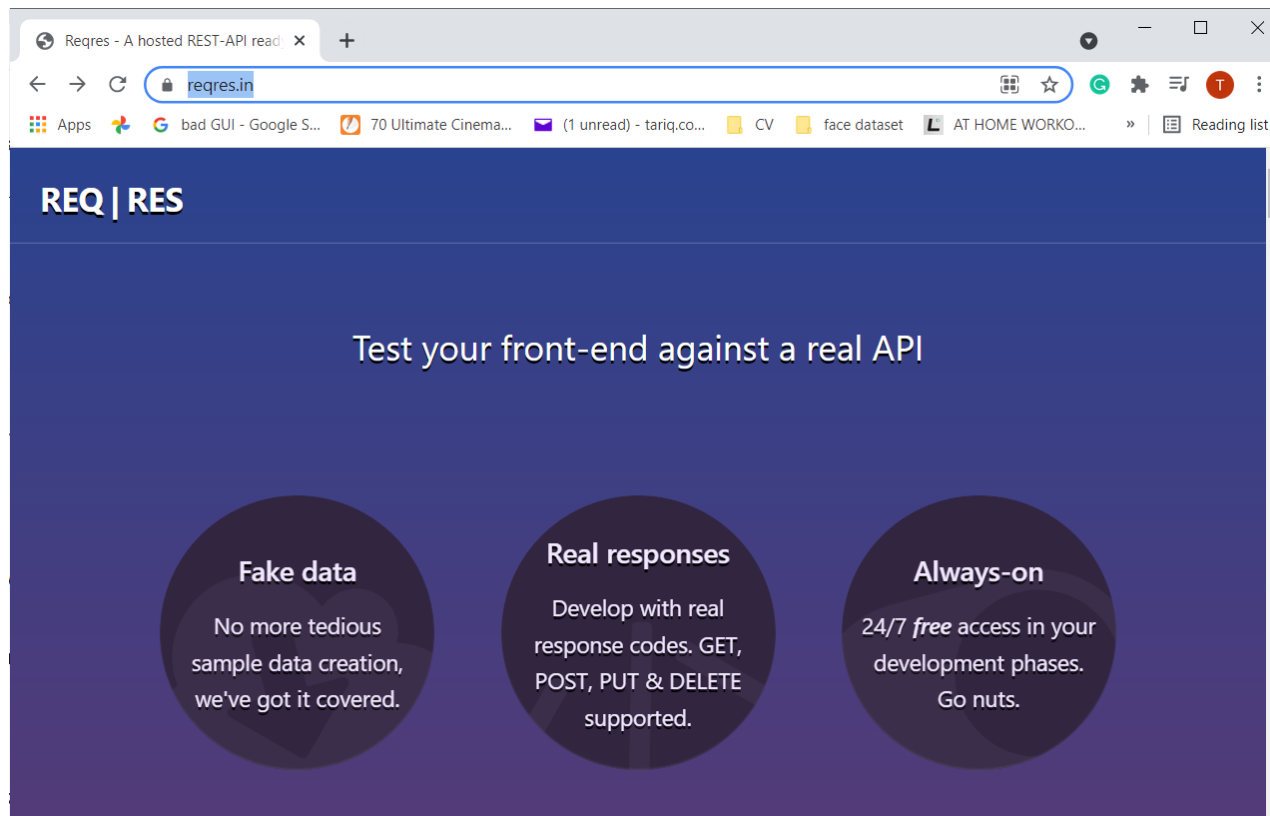
GET <http://example.co.nz/students/007>



Data as XML, JSON, ...

Test API

□ <https://reqres.in/>



HTTP GET

□ reqres.in/api/users?page=2

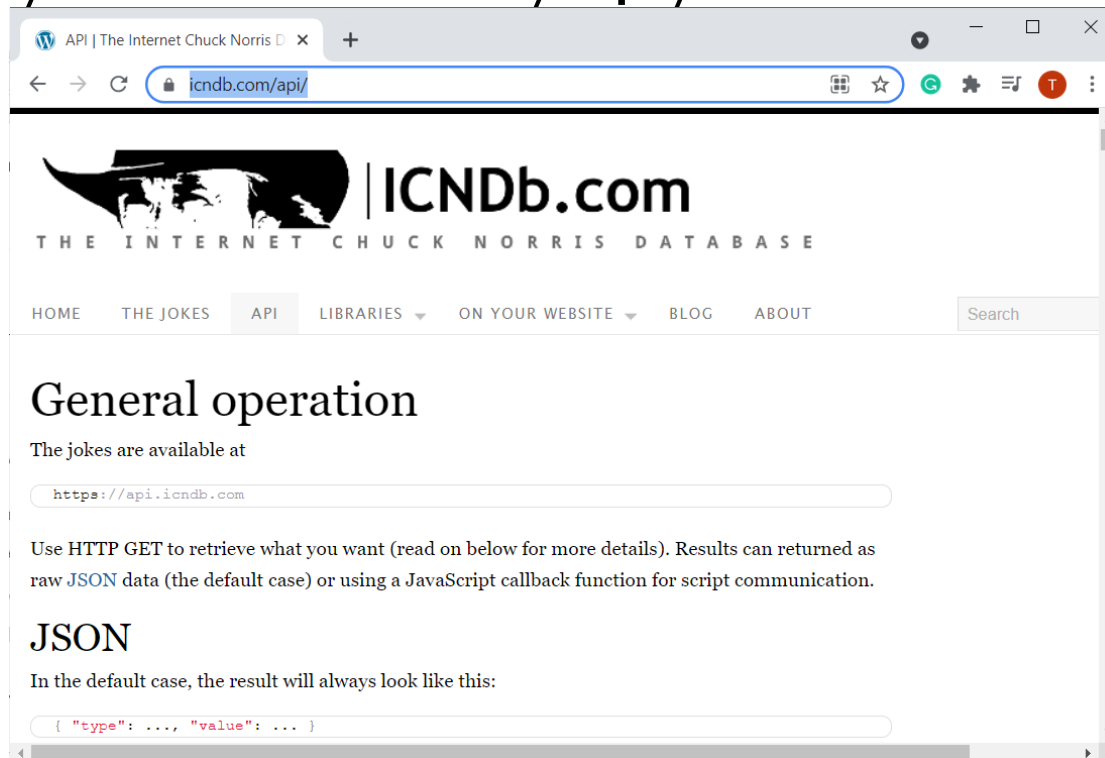
The screenshot shows the Postman application interface. The top bar includes the Postman logo, menu items (File, Edit, View, Help), and buttons for '+ New', 'Import', 'Runner', 'My Workspace', 'Invite', and 'Upgrade'. The left sidebar shows a 'History' tab with a list of requests, including the current one: 'GET reqres.in/api/users?page=2'. The main panel displays the 'Untitled Request' for the GET method to 'reqres.in/api/users?page=2'. The 'Params' tab is active, showing a 'Query Params' table with a single entry: 'page' with value '2'. The 'Body' tab is also visible, showing the response in 'Pretty' format. The response is a JSON object with the following structure:

```
1 {
2   "page": 2,
3   "per_page": 6,
4   "total": 12,
5   "total_pages": 2,
6   "data": [
7     {
8       "id": 7,
9       "email": "michael.lawson@reqres.in",
10      "first_name": "Michael",
11      "last_name": "Lawson",
12      "avatar": "https://reqres.in/img/faces/7-image.jpg"
13    }
14  ]
15 }
```

The status bar at the bottom shows '200 OK', '38 ms', '1.93 KB', and a 'Save Response' button. The bottom of the window has a 'Find and Replace' bar, a 'Console' button, and a 'Bootcamp' button.

Jokes

□ <https://www.icndb.com/api/>



JSON Properties

- ❑ Define objects and arrays with key value pairs
- ❑ Square brackets define arrays
- ❑ Curly Brackets define objects
- ❑ Key-values define object properties

```
[  
  {  
    "color": "red",  
    "value": "#f00"  
  },  
  {  
    "color": "green",  
    "value": "#0f0"  
  },  
  {  
    "color": "blue",  
    "value": "#00f"  
  },  
  {  
    "color": "cyan",  
    "value": "#0ff"  
  }  
]
```


- Dependencies build.gradle (Module: App) file

```
def retrofit_version = "2.8.1"  
// RETROFIT  
implementation "com.squareup.retrofit2:retrofit:$retrofit_version"  
implementation "com.squareup.retrofit2:converter-gson:$retrofit_version"
```

Add Internet Access To Manifest file

```
<uses-permission android:name="android.permission.INTERNET" />  
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Use following line to application tag in Manifest file for api 28 and above

```
android:usesCleartextTraffic="true"
```

<https://developer.android.com/guide/topics/manifest/application-element>

REST Api

```
public class User {  
    /*{  
        "id": 1,  
        "email": "george.bluth@reqres.in",  
        "first_name": "George",  
        "last_name": "Bluth",  
        "avatar": ""  
    }*/  
    int id;  
    String email;  
    String first_name;  
    String last_name;  
    String avatar;  
}
```

```
public class Users {  
    private List<User> data;  
  
    public List<User> getValue()  
  
}
```

User REST Api

- Create **Data Model Class(s)**
- Create **RESTApi Interface**

```
public interface UsersRESTAPI {  
  
    @GET("users/")  
    Call<Users> getUsers();  
  
}
```

User App

- Create **Controller**

```
final String BASE_URL = "http://regres.in/api/";
```

```
public void start(){  
    Retrofit retrofit = new Retrofit.Builder()  
        .baseUrl(BASE_URL)  
        .addConverterFactory(GsonConverterFactory.create())  
        .build();  
    UsersRESTAPI usersRESTAPI =  
        retrofit.create(UsersRESTAPI.class);  
    Call<Users> call = usersRESTAPI.getUsers();  
    call.enqueue(this);  
}
```

```
@Override  
public void onResponse(  
    Call<Users> call, Response<Users> response)
```

```
@Override  
public void onFailure(Call<Users> call, Throwable t) {  
    t.printStackTrace();  
}
```

Exercise

