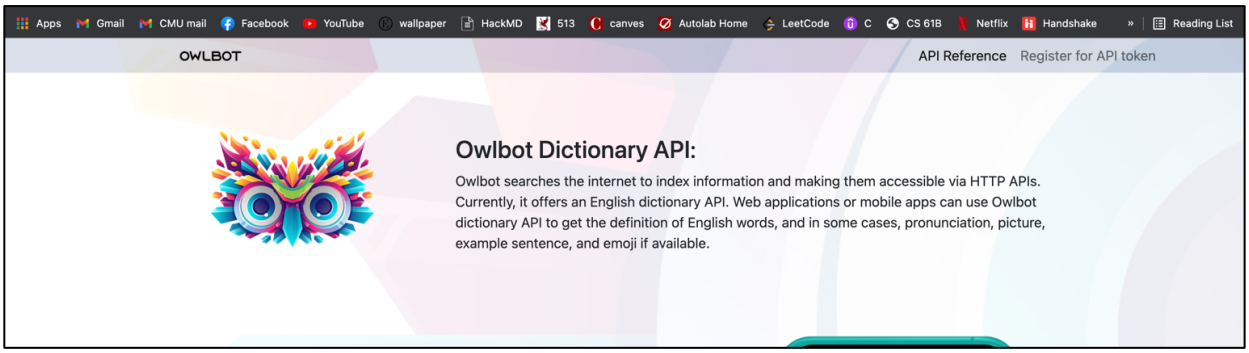
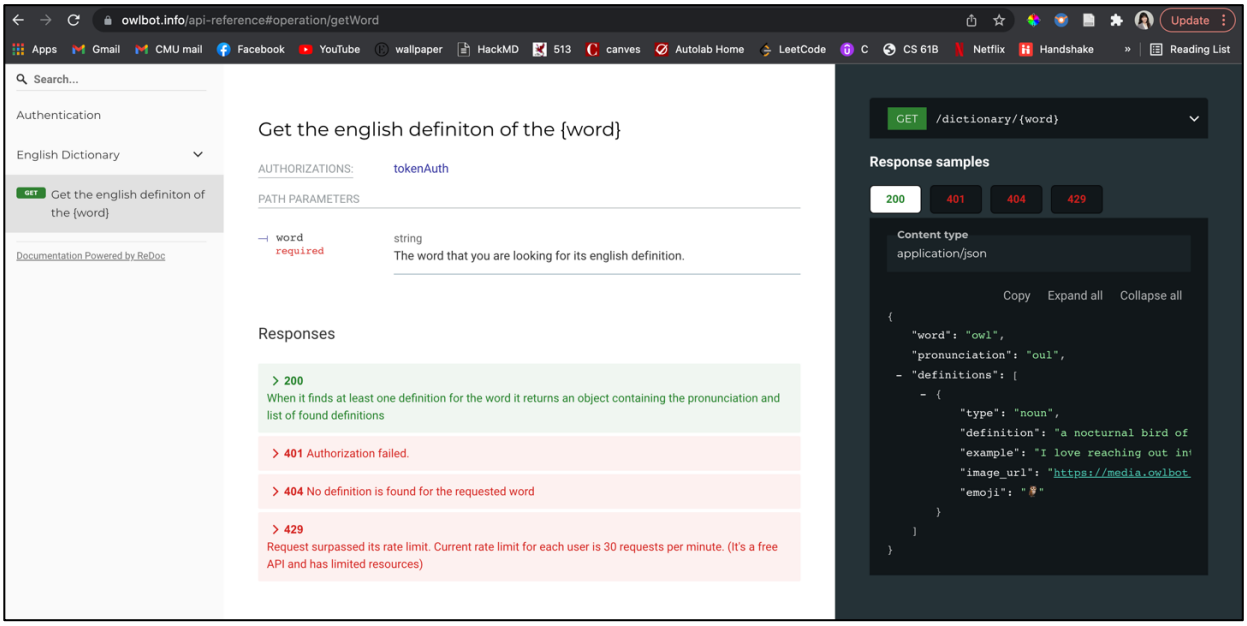
95-702 Distributed Systems
Project 4 Task 1

Author: Jennifer Chen (Yu Chen)
Andrew Id: yuc3 (yuc3@andrew.cmu.edu)
Date: 4/3/2022

My Dictionary App

Description:
My application takes a search string from the user, and uses it to fetch and display the string's definition and picture from the *Owlbot Dictionary API.* https://owlbot.info/api-reference



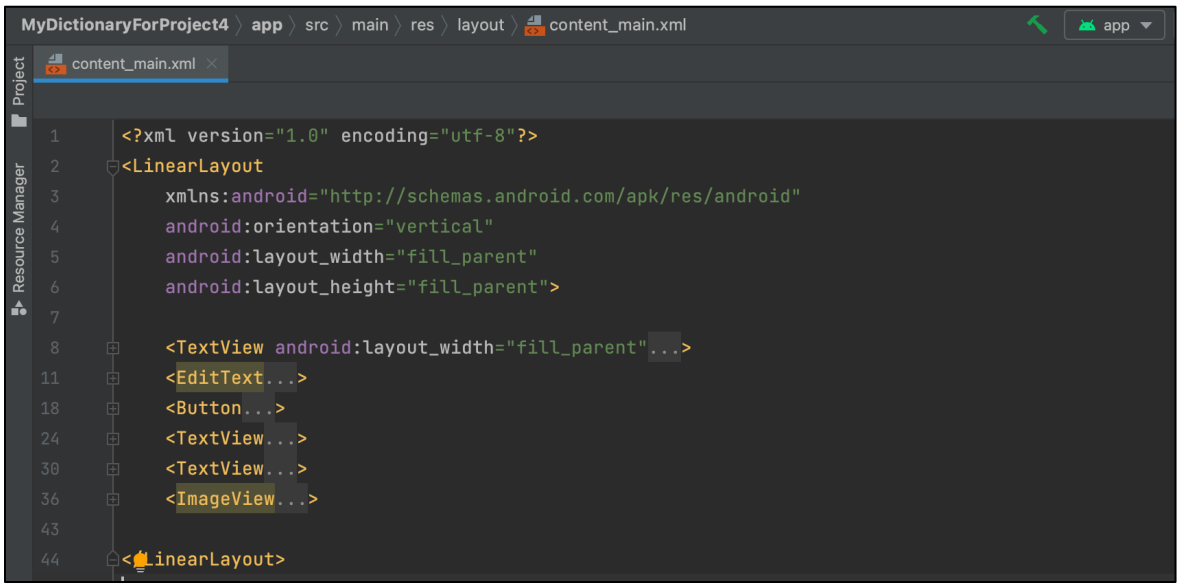


Below is how my application meets the task requirements:

1. Implement a native Android application:

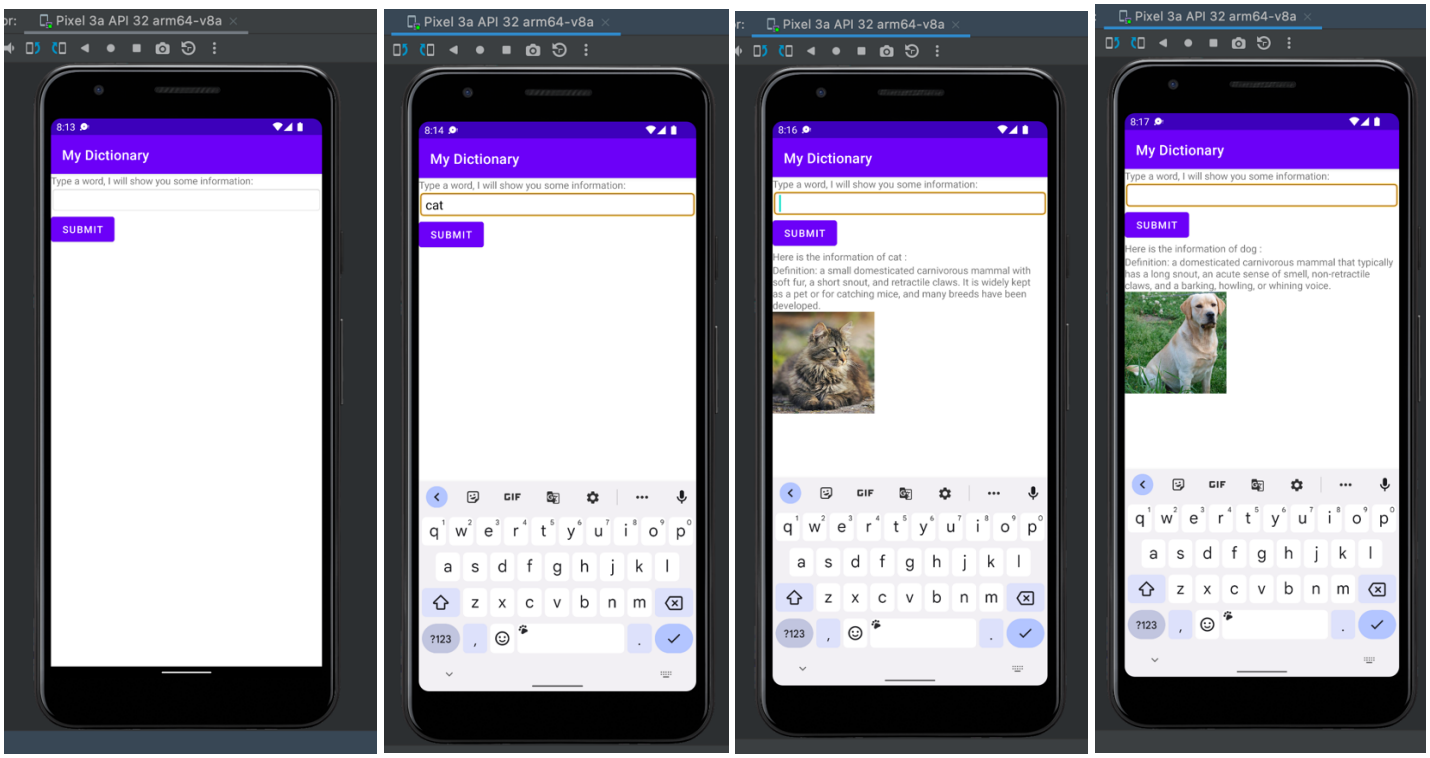
1-a. Has at least three different kinds of Views

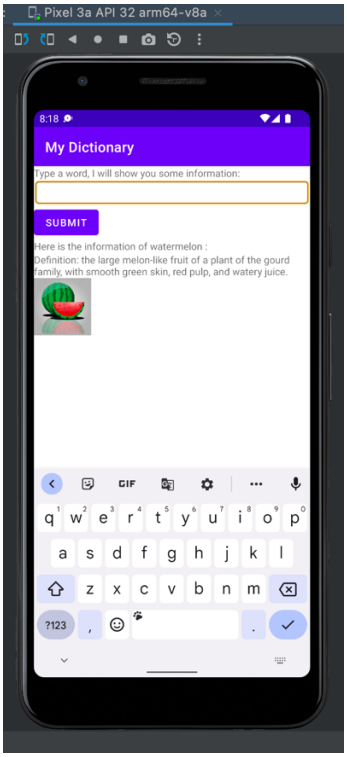
My application uses TextView, EditText, Button and ImageView. (Total: 4 Views)

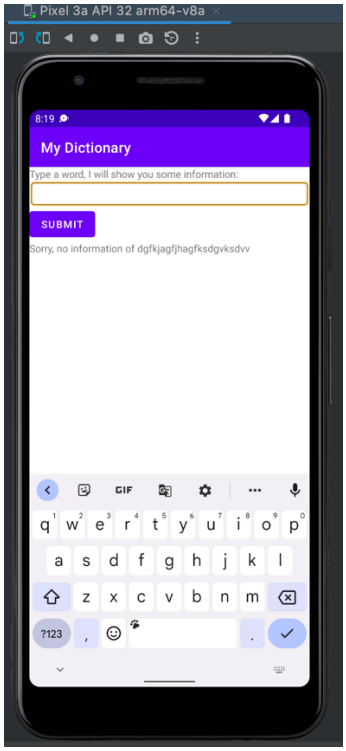


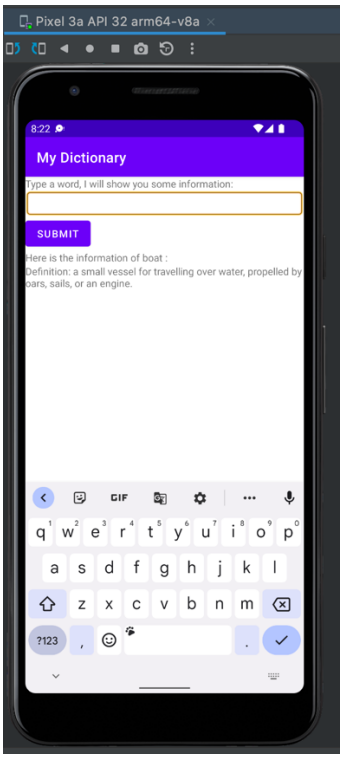
screenshot of *content_main.xml*

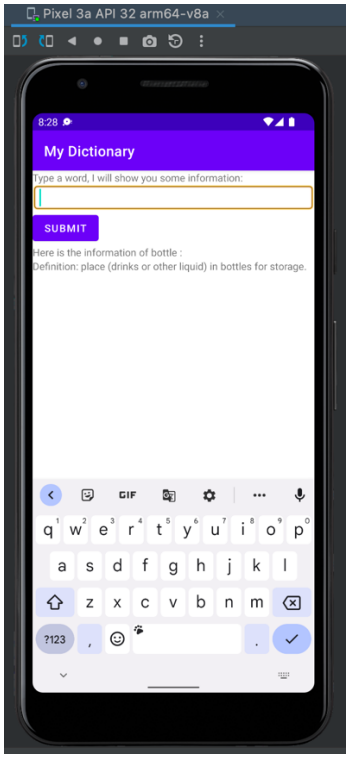
1-b. Requires input from the user
1-e. Displays new information to the user
1-f. Is repeatable (I.e. the user can repeatedly reuse the application without restarting it.)

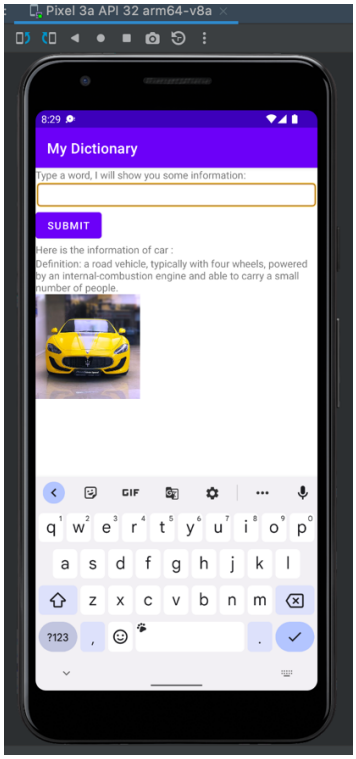


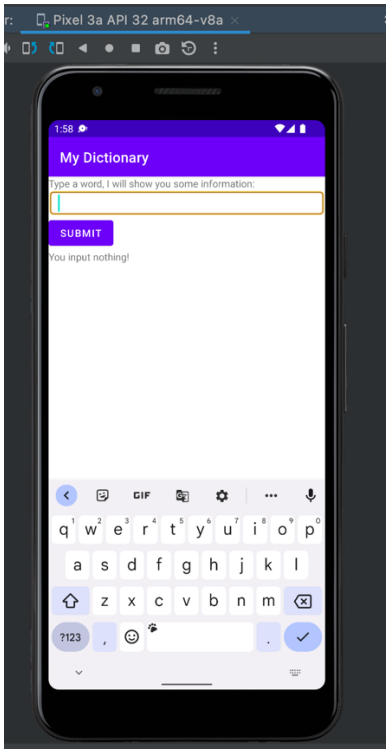












These screenshots show that my application could display the information from the api.
Will give “no information of XXX” if the user input is invalid. (definition is null)
Some word only have definition but no picture. (In this case, only show the definition to user)
If the user input an empty string, it will show “You input nothing!”

1-c. Makes an HTTP request (using an appropriate HTTP method) to your web service

My application does an HTTP GET request in *GetInfo.java*. The HTTP request is:

```
"https://still-refuge-70928.herokuapp.com/getWordInfo?word=" + searchTerm
```

where searchTerm is the user's search string.

The *getRemoteJson* method makes this request of my web application, parses the returned JSON to get the definition and picture URL. For the picture, it fetches the picture url, and returns the bit image of the picture.

```
80  @       private String getRemoteJson(String searchTerm){
81              StringBuilder res = new StringBuilder();
82              try {
83                  URL url = new URL( spec: "https://still-refuge-70928.herokuapp.com/getWordInfo?word=" + searchTerm);
84                  HttpURLConnection conn = (HttpURLConnection) url.openConnection();
85                  conn.setRequestMethod("GET");
86                  conn.setRequestProperty("Accept", "application/json");
87                  if (conn.getResponseCode() != 200) {
88                      throw new RuntimeException("Failed : HTTP Error code : "
```

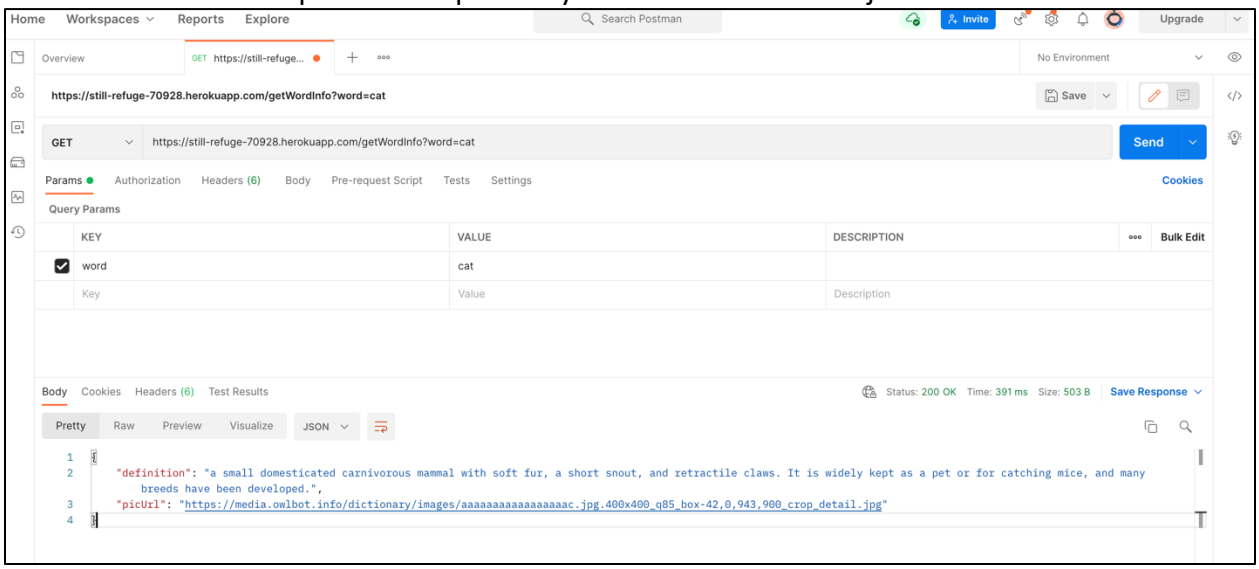
1-d. Receives and parses an XML or JSON formatted reply from your web service

An example of the JSON reply from my web service is as below:


```
{
    "definition": "a small domesticated carnivorous mammal with soft fur, a short snout, and retractile claws. It is widely kept as a pet or for catching mice, and many breeds have been developed.",
    "picUrl": "https://media.owlbot.info/dictionary/images/aaaaaaaaaaaaaaaac.jpg.400x400_q85_box-42,0,943,900_crop_detail.jpg"
}
```


Get the *definition* and *picUrl* to show to user.

Use postman to proof my web service returns a json format

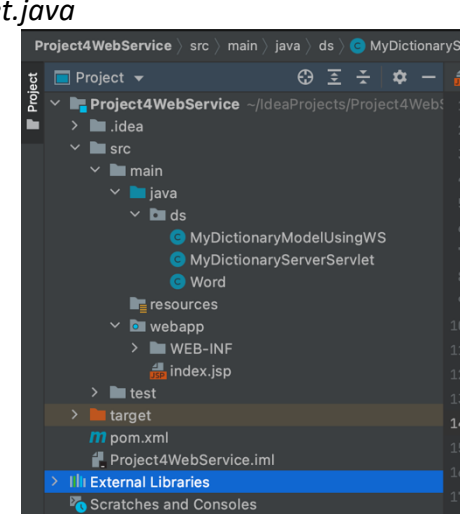


2. Implement a web service, deployed to Heroku

The URL of my web service deployed to Heroku is: still-refuge-70928
https://still-refuge-70928.herokuapp.com
The project directory name is Project4WebService.

2-a. Implement a simple (can be a single path) API

In my web app project:
Model: *MyDictionaryModelUsingWS.java*
View: *index.jsp*
Controller: *MyDictionaryServerServlet.java*



2-b. Receives an HTTP request from the native Android application

MyDictionaryServerServlet.java receives the HTTP GET request with the argument "search". It passes this search string on to the model.

```
25      @Override
26  @   protected void doGet(HttpServletRequest request,
27                           HttpServletResponse response)
28              throws IOException {
29
30          // get the search parameter if it exists
31          String search = request.getParameter( s: "word");
32
33          // use model to do the search
34          Word word = dm.fetchFromOwlApi(search);
35
36          //send response as json format
37          String res = word.toString();
```

2-c. Executes business logic appropriate to your application. This includes fetching XML or JSON information from some 3rd party API and processing the response.

MyDictionaryModelUsingWS.java makes an HTTP request to the Owlbot Dictionary API:

https://owlbot.info/api/v4/dictionary/{word}

It then parses the JSON response and extracts the parts it needs to respond to the Android application.

```java
public Word fetchFromOwlApi(String searchTag) throws IOException {
    searchTag = URLEncoder.encode(searchTag, enc: "UTF-8");
    StringBuilder res = new StringBuilder();
    try {
        URL url = new URL( spec: "https://owlbot.info/api/v4/dictionary/" + searchTag);
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        String API_KEY = "[redacted]";
        String basicAuth = "Token " + API_KEY;
        conn.setRequestProperty ("Authorization", basicAuth);
        conn.setRequestMethod("GET");
        conn.setRequestProperty("Accept", "application/json");

        if (conn.getResponseCode() != 200) {
            throw new RuntimeException("Failed : HTTP Error code : "
```

2-d. Replies to the Android application with an XML or JSON formatted response. The schema of the response can be of your own design.

Word.java's toString() method formats the response to the mobile application in a simple JSON format of my own design:

```java
@Override
public String toString() {
    String res = null;
    ObjectMapper Obj = new ObjectMapper();
    try {
        res = Obj.writeValueAsString(this);
    } catch (JsonProcessingException e) {
        e.printStackTrace();
    }
    return res;
}
```

I use Jackson library to map java objects to JSON.
The *Word.java* is an Object class that simply have two instances: definition and picUrl
The response example:

```
{
    "definition": "a small domesticated carnivorous mammal with soft fur, a short snout, and retractile claws. It is widely kept as a pet or for catching mice, and many breeds have been developed.",
    "picUrl": "https://media.owlbot.info/dictionary/images/aaaaaaaaaaaaaaaac.jpg.400x400_q85_box-42,0,943,900_crop_detail.jpg"
}
```

Send it from: *MyDictionaryServerServlet.java*

```java
PrintWriter out = response.getWriter();
response.setContentType("application/json");
response.setCharacterEncoding("UTF-8");
out.print(res);
out.flush();
```

For you reference:
The json get from Owlbot Dictionary API looks like this:


```
{
  "definitions":[
    {
      "type":"noun",
      "definition":"a domesticated carnivorous mammal that typically has a long snout, an acute sense of smell, non-retractile claws, and a barking, howling, or w
      "example":"she went for long walks with her dog",
      "image_url":"https://media.owlbot.info/dictionary/images/aaaaaaaaaaaaaaaaab.jpg.400x400_q85_box-0,0,2039,2039_crop_detail.jpg",
      "emoji":"🐕"
    }
  ],
  "word":"dog",
  "pronunciation":null
}
```


I only get the *definition* and *image_url* for my Android application.
Below is part of my *MyDictionaryModelUsingWS.java*

```
47      public Word parseInfoFromJson(String jsonString) throws IOException {
48          ObjectMapper mapper = new ObjectMapper();
49          JsonNode root = mapper.readTree(jsonString);
50          String def = root.get("definitions").get(0).get("definition").asText();
51          String picUrl = root.get("definitions").get(0).get("image_url").asText();
52          return new Word(def, picUrl);
53      }
```