Project 4 Task 1 — iNaturalist@Nearby App

by Anran Lin (AndrewID: anranlin)

Github:

https://github.com/genie122226/iNaturalistNearby

Description:

My application enables users to search for species observation records within a certain radius and time frame from their own location. Users can specify the species name if desired. Since the IP of the virtual machine is fixed, an IP input field is provided, allowing the operation of all functions except for acquiring the IP. Ultimately, the application provides information about the species, including its name, images, observation count, ID, total number of observations, activity status, and a Wikipedia link.

Here is how my application meets the task requirements

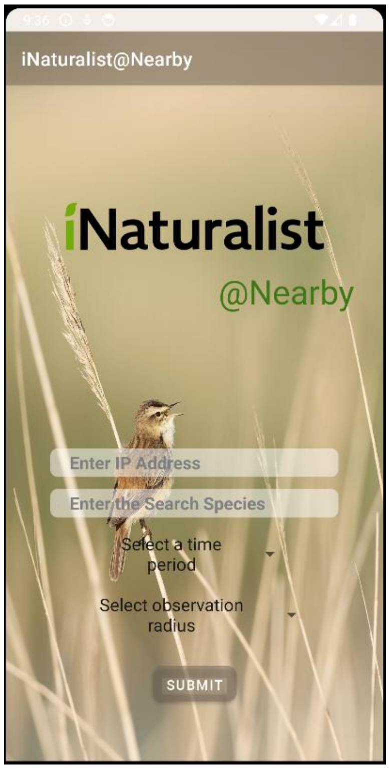
1. Implement a native Android application

The name of my native Android application project in Android Studio is: iNaturalistNear

a. Has at least three different kinds of views in your Layout (TextView, EditText, ImageView, etc.)

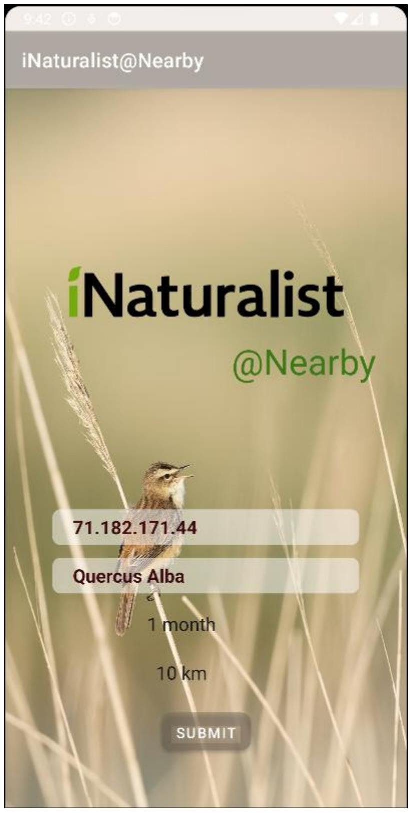
My application uses TextView, EditText, Button, Spinner and ImageView. See content_main.xml for details of how they are incorporated into the LinearLayout.

Here is a screenshot of the layout before the data has been fetched.


iNaturalist@Nearby
iNaturalist
@Nearby
Enter IP Address
Enter the Search Species
Select a time period
Select observation radius
SUBMIT

b. Requires input from the user

As mentioned earlier, the application requires users to provide a virtual IP address. Within the app, if no IP is entered, it defaults to a predefined IP value, which can be found in the main program. If no species name is inputted, the application defaults to retrieving observation records for all species. If neither a time period nor a radius is selected, the default values for these are set to 1 year and 5 km, respectively. Here is a screenshot of the user searching for a picture of a Quercus Alba in IP 71.182.171.44, with time period “1 month” and radius “10 km”.



71.182.171.44

Quercus Alba

1 month

10 km

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

After receiving user input, my application first parses the information and uses either the entered IP or the actual user's IP address to obtain the corresponding latitude and longitude. The API for acquiring latitude and longitude is as follows: 'https://ipapi.co/' + [IP] + '/json/'. Upon receiving a JSON response, the app parses the information to extract latitude and longitude. These, along with other user-inputted data, are then used in a second API to obtain species observation records. The second API is formatted as follows: https://api.inaturalist.org/v1/observations/species_counts?' +'taxon_name=' + [search term] +'&year=' + [current year (if a 1-year time range is selected)] +'&month=' + [current month (if a 1-month time range is selected)] +'&lat=' + [latitude] +'&lng=' + [longitude] +'&radius=' + [radius].

The search method makes this request of my web application. Then, my program parses the returned JSON into information corresponding to a class I created called

ResponseSpecies. After extracting the necessary log information from this, the JSON is submitted to the main program for further processing in the next steps.

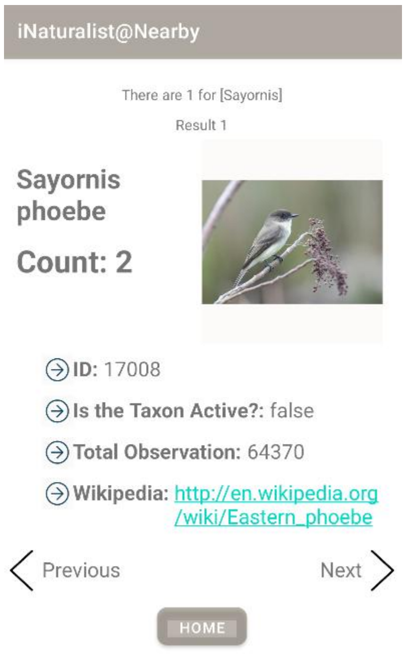
d. Receives and parses an XML or JSON formatted reply from the web service

An example of the JSON reply is:

```
{
 "total_results": 4,
 "page": 1,
 "per_page": 500,
 "results": [
   {
    "count": 12041,
    "taxon": {
      "id": 17008,
      "rank": "species",
      "rank_level": 10,
      "iconic_taxon_id": 3,
      "ancestor_ids": [
        48460,
        1,
        2,
        355675,
        3,
        7251,
        980017,
        17007,
        17008
      ],
      "is_active": true,
      "name": "Sayornis phoebe",
      "parent_id": 17007,
      "ancestry": "48460/1/2/355675/3/7251/980017/17007",
      "extinct": false,
      "default_photo": {
        "id": 171693341,
        "license_code": null,
        "attribution": "(c) j_albright, all rights reserved",
        "url": "https://static.inaturalist.org/photos/171693341/square.jpg",
        "original_dimensions": {
          "height": 1365,
          "width": 2048
        },
        "flags": [],
        "square_url": "https://static.inaturalist.org/photos/171693341/square.jpg",
        "medium_url": "https://static.inaturalist.org/photos/171693341/medium.jpg"
      },
      "taxon_changes_count": 0,
      "taxon_schemes_count": 9,
      "observations_count": 64370,
      "flag_counts": {
        "resolved": 1,
        "unresolved": 0
      },
      "current_synonymous_taxon_ids": null,
      "atlas_id": null,
    "complete_species_count": null,
    "wikipedia_url": "http://en.wikipedia.org/wiki/Eastern_phoebe",
    "complete_rank": "subspecies",
    "iconic_taxon_name": "Aves",
    "preferred_common_name": "Eastern Phoebe"
  }
},
{
  "count": 8765,
  "taxon": {
    "id": 17013,
    "rank": "species",
    "rank_level": 10,
    "iconic_taxon_id": 3,
    "ancestor_ids": [
      48460,
      1,
      2,
      355675,
      3,
      7251,
      980017,
      17007,
      17013
    ],
    "is_active": true,
    "name": "Sayornis nigricans",
    "parent_id": 17007,
    "ancestry": "48460/1/2/355675/3/7251/980017/17007",
    "extinct": false,
    "default_photo": {
      "id": 105821053,
      "license_code": null,
      "attribution": "(c) Robyn Waayers, all rights reserved, uploaded by Robyn Waayers",
      "url": "https://static.inaturalist.org/photos/105821053/square.jpg",
      "original_dimensions": {
        "height": 1399,
        "width": 2048
      },
      "flags": [],
      "square_url": "https://static.inaturalist.org/photos/105821053/square.jpg",
      "medium_url": "https://static.inaturalist.org/photos/105821053/medium.jpg"
    },
    "taxon_changes_count": 0,
    "taxon_schemes_count": 9,
    "observations_count": 57839,
    "flag_counts": {
      "resolved": 0,
      "unresolved": 0
    },
    "current_synonymous_taxon_ids": null,
    "atlas_id": null,
    "complete_species_count": null,
    "wikipedia_url": "https://en.wikipedia.org/wiki/Black_phoebe",
    "complete_rank": "subspecies",
    "iconic_taxon_name": "Aves",
    "preferred_common_name": "Black Phoebe"
  }
},
{
  "count": 4649,
  "taxon": {
    "id": 17009,
    "rank": "species",
    "rank_level": 10,
    "iconic_taxon_id": 3,
    "ancestor_ids": [
      48460,
      1,
      2,
      355675,
      3,
      7251,
      980017,
      17007,
      17009
    ],
    "is_active": true,
    "name": "Sayornis saya",
    "parent_id": 17007,
    "ancestry": "48460/1/2/355675/3/7251/980017/17007",
    "extinct": false,
    "default_photo": {
      "id": 477283,
      "license_code": null,
      "attribution": "(c) BJ Stacey, all rights reserved",
      "url": "https://static.inaturalist.org/photos/477283/square.jpg",
      "original_dimensions": {
        "height": 1365,
        "width": 2048
      },
      "flags": [],
      "square_url": "https://static.inaturalist.org/photos/477283/square.jpg",
      "medium_url": "https://static.inaturalist.org/photos/477283/medium.jpg"
    },
    "taxon_changes_count": 0,
    "taxon_schemes_count": 9,
    "observations_count": 27338,
    "flag_counts": {
      "resolved": 0,
      "unresolved": 0
    },
    "current_synonymous_taxon_ids": null,
    "atlas_id": null,
    "complete_species_count": null,
    "wikipedia_url": "http://en.wikipedia.org/wiki/Say's_phoebe",
    "complete_rank": "subspecies",
    "iconic_taxon_name": "Aves",
    "preferred_common_name": "Say's Phoebe"
  }
},
{
    "count": 4,
    "taxon": {
      "id": 1465306,
      "rank": "hybrid",
      "rank_level": 10,
      "iconic_taxon_id": 3,
      "ancestor_ids": [
        48460,
        1,
        2,
        355675,
        3,
        7251,
        980017,
        17007,
        1465306
      ],
      "is_active": true,
      "name": "Sayornis nigricans × phoebe",
      "parent_id": 17007,
      "ancestry": "48460/1/2/355675/3/7251/980017/17007",
      "extinct": false,
      "default_photo": {
        "id": 268876269,
        "license_code": "cc-by",
        "attribution": "(c) ecovore, some rights reserved (CC BY), uploaded by ecovore",
        "url": "https://inaturalist-open-data.s3.amazonaws.com/photos/268876269/square.jpeg",
        "original_dimensions": {
          "height": 1536,
          "width": 2048
        },
        "flags": [],
        "square_url": "https://inaturalist-open-data.s3.amazonaws.com/photos/268876269/square.jpeg",
        "medium_url": "https://inaturalist-open-data.s3.amazonaws.com/photos/268876269/medium.jpeg"
      },
      "taxon_changes_count": 0,
      "taxon_schemes_count": 0,
      "observations_count": 6,
      "flag_counts": {
        "resolved": 0,
        "unresolved": 0
      },
      "current_synonymous_taxon_ids": null,
      "atlas_id": null,
      "complete_species_count": null,
      "wikipedia_url": null,
      "complete_rank": "subspecies",
      "iconic_taxon_name": "Aves",
      "preferred_common_name": "Black × Eastern Phoebe"
    }
  }
 ]
}
```

e. Displays new information to the user

After the user receives the data, since there are multiple groups of results, these are processed in batches in 'GetOneOfResult.java' to be displayed on 'activity_main2.xml'. Logic for selecting the previous and next items has been implemented in the 'activity_main2.xml' interface, which can be activated by clicking on arrows. Here is the screen shot after the result has been dealt with.


iNaturalist@Nearby

There are 1 for [Sayornis]

Result 1

Sayornis phoebe

Count: 2

ID: 17008

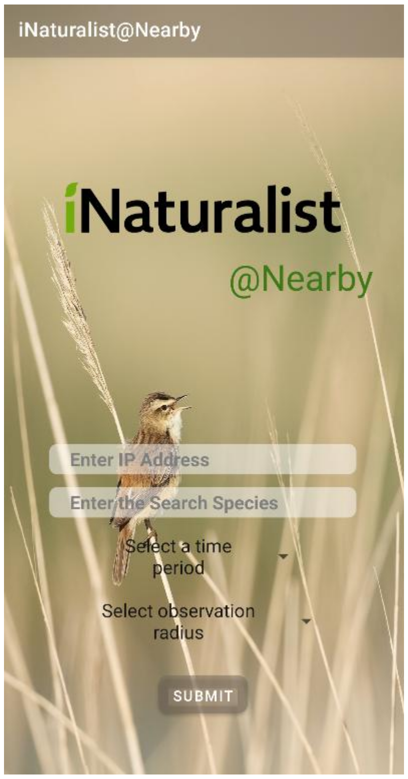
Is the Taxon Active?: false

Total Observation: 64370

Wikipedia: http://en.wikipedia.org/wiki/Eastern_phoebe

f. Is repeatable (I.e. the user can repeatedly reuse the application without restarting it.)

Users have the option to click the home button to return to the main page ('activity_main.xml'), as illustrated in the interface.


iNaturalist@Nearby
iNaturalist
@Nearby
Enter IP Address
Enter the Search Species
Select a time period
Select observation radius
SUBMIT

2. Implement a web application, deployed to Heroku

The URL of my web service deployed to Github is:

https://congenial-space-telegram-wr7x4qg5vrwp29wv5-8080.app.github.dev/inaturalist

a. Using an HttpServlet to implement a simple (can be a single path) API

In my web app project:

Server.java: receive the data and send response

MongoDBUtil.java: used for connecting MongoDB, logging and get data

LogDashboardServlet: used for constructing the dashboard

dashboard.jsp: to show the stored data (/dashboard)

b. Receives an HTTP request from the native Android application

Server.java receives the HTTP POST request with the argument "search".

c. Executes business logic appropriate to your application

.
After receiving user input, Server.java first parses the information and uses either the entered IP or the actual user's IP address to obtain the corresponding latitude and longitude. The API for acquiring latitude and longitude is as follows: 'https://ipapi.co/' + [IP] + '/json/'. Upon receiving a JSON response, the app parses the information to extract latitude and longitude. These, along with other user-inputted data, are then used in a second API to obtain species observation records. The second API is formatted as follows: https://api.inaturalist.org/v1/observations/species_counts?' +'taxon_name=' + [search term] +'&year=' + [current year (if a 1-year time range is selected)] +'&month=' + [current month (if a 1-month time range is selected)] +'&lat=' + [latitude] +'&lng=' + [longitude] +'&radius=' + [radius]. It then parses the JSON response and extracts the parts it needs to store, then send the JSON response to the Android application.

Then the data will be stored in a Logentry object, and it will be passed to MongoDBUtil for store. If the user want to see the stored data, they can see "/dashboard", and LogDashboardServlet.java will take data from database and show them in dashboard.jsp.

d. Replies to the Android application with an XML or JSON formatted response.

See as 1.d.

To document the rest of the requirements:

3. Handle error conditions - Does not need to be documented.

4. Log useful information - Itemize what information you log and why you chose it.

Most Common IP: Identifies which IP address has searched for species observation records the most frequently.

Most Common Search Term: Determines which species is the most popular or receives the most attention.

Average Operation Time: Measures the response speed of the application.

Request Message: Logs the user's request.

Request Timestamp: The timestamp of when the user's request is made.

API1url: The query URL used for the first API.

API1 Response: The address information returned by the API.

API2 URL: The query URL used for the second API.

API2 Response: Total Number of Results: Due to an abundance of species observation records, not all are displayed on the dashboard. Instead, it shows the total number of species found.

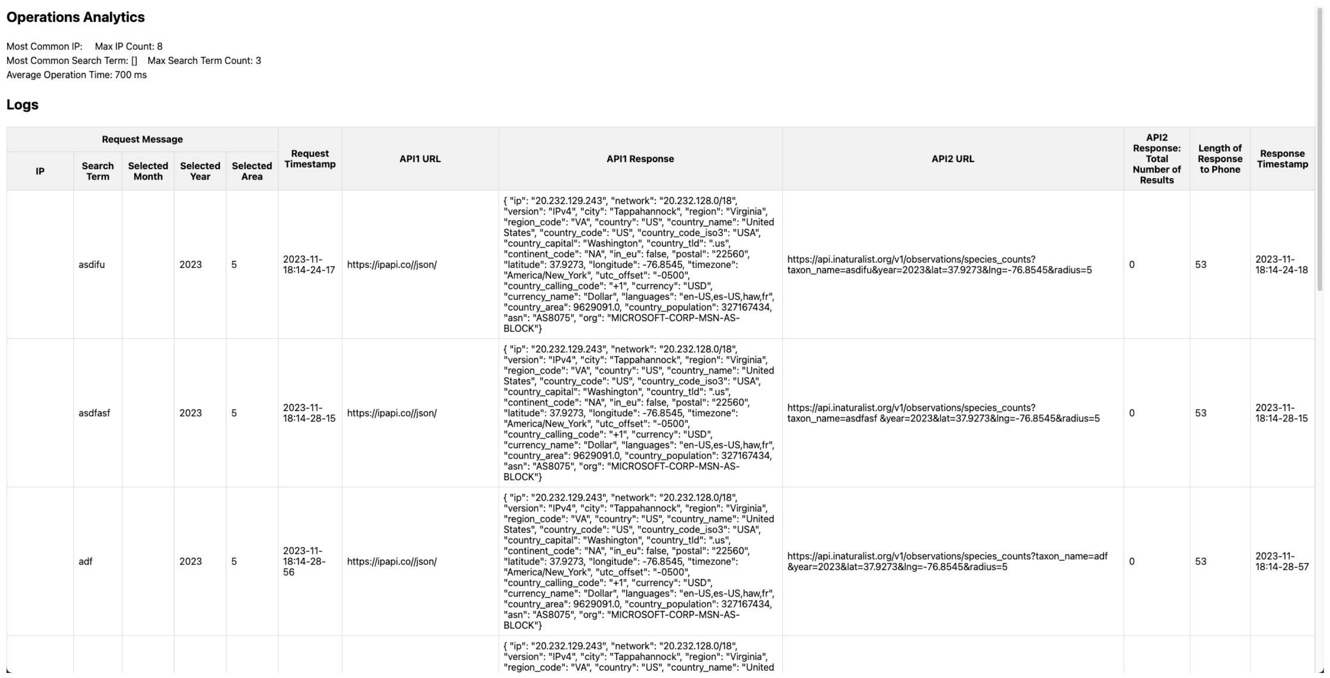
Length of Response to Phone: The length of the response sent to the Android application.

Response Timestamp: The timestamp of when the user's request is responded to.

5. Store the log information in a database - Give your Atlas connection string with the three shards

"mongodb://xilouhexi:d4daBSH2R2ruyX21@ac-mvrozk9-shard-00-00.99f3ywm.mongodb.net:27017,ac-mvrozk9-shard-00-01.99f3ywm.mongodb.net:27017,ac-mvrozk9-shard-00-02.99f3ywm.mongodb.net:27017/test?w=majority&retryWrites=true&tls=true&authMechanism=SCRAM-SHA-1"

6. Display operations analytics and full logs on a web-based dashboard - Provide a screen shot.



Operations Analytics

Most Common IP: Max IP Count: 8
Most Common Search Term: []   Max Search Term Count: 3
Average Operation Time: 700 ms

Logs

Request Message					Request Timestamp	API1 URL	API1 Response	API2 URL	API2 Response: Total Number of Results	Length of Response to Phone	Response Timestamp
IP	Search Term	Selected Month	Selected Year	Selected Area							
	asdifu		2023	5	2023-11-18:14-24-17	https://ipapi.co//json/	{ "ip": "20.232.129.243", "network": "20.232.128.0/18", "version": "IPv4", "city": "Tappahannock", "region": "Virginia", "region_code": "VA", "country": "US", "country_name": "United States", "country_code": "US", "country_code_iso3": "USA", "country_capital": "Washington", "country_tld": ".us", "continent_code": "NA", "in_eu": false, "postal": "22560", "latitude": 37.9273, "longitude": -76.8545, "timezone": "America/New_York", "utc_offset": "-0500", "country_calling_code": "+1", "currency": "USD", "currency_name": "Dollar", "languages": "en-US,es-US,haw,fr", "country_area": 9629091.0, "country_population": 327167434, "asn": "AS8075", "org": "MICROSOFT-CORP-MSN-AS-BLOCK"}	https://api.inaturalist.org/v1/observations/species_counts?taxon_name=asdifu&year=2023&lat=37.9273&lng=-76.8545&radius=5	0	53	2023-11-18:14-24-18
	asdfasf		2023	5	2023-11-18:14-28-15	https://ipapi.co//json/	{ "ip": "20.232.129.243", "network": "20.232.128.0/18", "version": "IPv4", "city": "Tappahannock", "region": "Virginia", "region_code": "VA", "country": "US", "country_name": "United States", "country_code": "US", "country_code_iso3": "USA", "country_capital": "Washington", "country_tld": ".us", "continent_code": "NA", "in_eu": false, "postal": "22560", "latitude": 37.9273, "longitude": -76.8545, "timezone": "America/New_York", "utc_offset": "-0500", "country_calling_code": "+1", "currency": "USD", "currency_name": "Dollar", "languages": "en-US,es-US,haw,fr", "country_area": 9629091.0, "country_population": 327167434, "asn": "AS8075", "org": "MICROSOFT-CORP-MSN-AS-BLOCK"}	https://api.inaturalist.org/v1/observations/species_counts?taxon_name=asdfasf &year=2023&lat=37.9273&lng=-76.8545&radius=5	0	53	2023-11-18:14-28-15
	adf		2023	5	2023-11-18:14-28-56	https://ipapi.co//json/	{ "ip": "20.232.129.243", "network": "20.232.128.0/18", "version": "IPv4", "city": "Tappahannock", "region": "Virginia", "region_code": "VA", "country": "US", "country_name": "United States", "country_code": "US", "country_code_iso3": "USA", "country_capital": "Washington", "country_tld": ".us", "continent_code": "NA", "in_eu": false, "postal": "22560", "latitude": 37.9273, "longitude": -76.8545, "timezone": "America/New_York", "utc_offset": "-0500", "country_calling_code": "+1", "currency": "USD", "currency_name": "Dollar", "languages": "en-US,es-US,haw,fr", "country_area": 9629091.0, "country_population": 327167434, "asn": "AS8075", "org": "MICROSOFT-CORP-MSN-AS-BLOCK"}	https://api.inaturalist.org/v1/observations/species_counts?taxon_name=adf &year=2023&lat=37.9273&lng=-76.8545&radius=5	0	53	2023-11-18:14-28-57
							{ "ip": "20.232.129.243", "network": "20.232.128.0/18", "version": "IPv4", "city": "Tappahannock", "region": "Virginia", "region_code": "VA", "country": "US", "country_name": "United				