OpenStreetMap

Google Summer of Code 2021

Project Proposal - Interface for reporting search bugs for Nominatim

Mentors: Sarah Hoffmann, *Marc Tobias Metten*

WHO AM I?

Name	Yash Srivastava
Gender	Male
GitHub	darkshredder
LinkedIn	Yash Srivastava
Email	srivastavayash58@gmail.com, yash_s@ph.iitr.ac.in
Personal Website	https://yashsrivastava.netlify.app/
OSM Account	Darkshredder
Phone no.	+91-8081842714
Nationality	India 🇮🇳
Timezone	IST (UTC + 5:30)
Résumé	Yash Srivastava
Education	**Degree:** Integrated Master of Technology (Int M.Tech.) **Year:** 2nd year **Majors in:** Geophysical Technology **Institute:** Indian Institute of Technology, Roorkee, Uttarakhand

ABOUT ME

I am Yash Srivastava. A sophomore currently enrolled in Geophysical Technology at the Indian Institute of Technology at Roorkee in Uttarakhand, India. My love for computers and various computer languages began from 8th standard when I was introduced to C in my curriculum. Since then, I have developed a passion for exploring more in the field of Computer Science and applying those concepts to build various daily-use applications. In my freshman year, I explored web development and worked on some projects and also did an Internship which was enlightening learning experiences for me.

In my pursuit of acquiring web development skills, I started contributing to the open-source community to give back to the organisations whose products have been valuable assets to me.

CODING SKILLS

Computer Languages	C/C++, PHP, JavaScript, HTML, CSS, SQL, BASH, Python, Ruby, Solidity
Databases	PostgreSQL, MySQL, MongoDB, SQLite
Web Frameworks	React, Redux, Django, Django REST, Django Channels, Web sockets, Flask, Node.js, Express.js
Utilities	Git, Vim, VS Code, Postman, Linux shell utilities, JetBrains IDEs

Development Environment

- Ubuntu 20.04.1 LTS
- Visual Studio Code - Insiders as a primary code editor.
- Z Shell as default command-line shell
- Vim/Nano for minor edits
- Git for Version Control

PROJECTS

Fan Clubs Portal

- It is a portal where Fans/Users can create chat rooms for any movie or series they like, which other users can follow.
- It has various functionalities like showing a leaderboard. Users can see other user's profile which includes hobbies. It also has google-oauth2 support.
- Tech stack includes Django, Django-Channels, ReactJS, Websockets, Docker.
- Visit https://github.com/darkshredder/fanclub, fanclubiitr.netlify.app/

Crypto-Estate

- Crypto-Estate is a blockchain-powered solution where users can Buy/Sell properties with the help of maps directly and see it's all details there rather than scrolling through different pages.
- I developed it's core frontend, including all maps and it's integration with the blockchain network and wrote the blockchain backend.
- Tech stack includes JavaScript, ReactJS, Google Maps API, Solidity, Truffle.
- Visit https://github.com/darkshredder/Crypto-Estate, jolly-sun-5656.on.fleek.co/

E-summit 2021

- E-Summit is the annual entrepreneurship summit of IIT Roorkee.
- I developed Backend, Created dashboards for user's and also integrated the backend with the frontend.
- Tech stack includes Python, Django, Django REST, ReactJS, Redux.
- Visit https://www.esummit.in/.

Contribute Play Earn

- Contribute Play Earn is a platform where the users can get crypto coins by playing games or contributing to the open-source world. They just need to login through GitHub, and they will be provided crypto-coins based on the pull request they have submitted.
- Tech stack includes JavaScript, ReactJS, GitHub-oauth2, Solidity, Truffle.
- Visit https://github.com/darkshredder/contribute-play-earn.

There are several personal projects I have worked on and also contributed to the development endeavours of others. All of that can be found in my GitHub profile, and the description projects in which I did internships can be found in my Resume.

PRE-GSOC INVOLVEMENTS & OPEN-SOURCE CONTRIBUTIONS

OpenStreetMap

Pull Requests

PR	Contributions	Status
#2204	**[Nominatim]** Ported Tiger-data-import to Python and Added Tarball Support	**Merged**
#2212	**[Nominatim]** Ported createCountryNames() to python and Added tests.	**Merged**
#2234	**[Nominatim]** Added Manual page for Nominatim tool.	**Merged**
#2243	**[Nominatim]** Fixed: XML format: more_url points to localhost, not base URL.	**Merged**
#2252	**[Nominatim]** Added Code coverage support using Codecov.	**Merged**
#2258	**[Nominatim]** Disabled Code coverage status checks	**Merged**

Issues opened

PR	Contributions	Status
#2209	**[Nominatim]** Adding Pull Request template	**Closed**
#2245	**[Nominatim]** Adding Code Coverage to the Project	**Closed**

Other Open-Source Contributions

I have been contributing to the open-source community for almost a year. The list of all my open-source contributions can be found here.

PROJECT SPECIFIC QUESTIONS:

Have you applied for other project ideas/organisations?

No. I am not applying for other project ideas/organisations.

Are you planning any vacations this summer?

No. I am not planning any sort of vacation this summer.

How many classes are you taking this summer?

I will have my regular classes from 17th May to 31st May. During this time, I have my semester exams from 19th May to 29th May. My summer vacations will begin from 1st June till 30th July. After this, my regular classes will begin.

Do you have any other employment this summer?

No, I don't have any other internship/project lined up this summer, So I will easily manage time for the project this summer.

How many hours per week do you expect to work on this project?

- **During semester exams:** I can easily work for **20** hours per week as I will have to give only four exams in a span of 10 days.
- **During summer vacations**: As I would not have any internships/projects lined up, I can easily devote an average of **50-55** hours per week.
- **During regular classes:** I can easily devote an average of **40-45** hours per week.

THE PROJECT

Project Idea: Interface for reporting search bugs for Nominatim

How do users report bugs now?

Currently, there is no standard platform where users can report bugs when they are working with Nominatim. Generally, if a user wants to report a bug he/she needs to go to the GitHub repository and then report the bug. But the problem here is that user's who aren't programmers and who don't have Github accounts find it tedious and challenging to report bugs.

What needs to be done to tackle this issue?

As there is no current interface for reporting bugs, A separate interface needs to be set up where users can interact and report bugs in a hassle-free way.

Implementation:

Deciding tech stack for the project

1. Backend**:**

   We need to select a backend that is lightweight and highly customisable as we just need to put the data coming from the client to the log file, Which can be later processed using a script to CSV files.

   Implementation can be done in any one of the following ways:

a. We can use **Flask** as a backend python framework. It is lightweight and fully customisable and can easily create CSV files from log files. We can also easily integrate flask with Geocode-Tester as they both are written in python.

b. We can also use **NodeJs** as a backend javascript framework. It is faster than flask, lightweight and has a non-blocking event-driven architecture that enables asynchronous input/output, which means a particular process is called as soon as the respective event occurs. That is, no process blocks the thread. We can also create CSV files from log files, which can be further processed with Geocode-Tester.

2. Frontend**:**

We need to select a frontend framework that has easily maintainable code, is flexible, provides a router, can be easily integrated with the map using Leaflet, has support for translations probably through i18n and is popular enough so that if any issues come in the way can be easily handled.

Implementation can be done in any one of the following ways:

a. We can use **Svelte.** It's a compiler that takes your declarative components and converts them into efficient JavaScript that surgically updates the DOM. It provides us with features like no virtual DOM, Truly reactive, has translations support and is used with components for easy and maintainable code.

b. We can use **React**. It provides us with a reusable coding style with the help of components. It provides a lot of creating frontend applications and also has i18n support for translations. It has one of the largest community in frontend javascript frameworks and is also extremely fast and high-performance.

c. We can use **Vue.** It is very similar to React but is less popular than React. Along with excellent documentation, it can be a perfect tool for building a frontend application and also has translation support.

d. We can use **Angular.** It provides a fast performant server, has translation support, can use reusable code using component-based architecture and has an MVC based architecture but is very complex and has a lesser community than React.

Workflow

Backend Workflow

a. **Initialise a backend server**

   I. Build a **REST API** server on the decided web framework, which shall handle the requests and query made by the users.

b. **Write endpoints to communicate with the frontend and create log files.**

c. **Integrate Geocode-Tester with backend**

   i. Add a script to export the log files to CSV files to be processed through Geocode-Tester.

d. **Setup Tests for backend**

   i. Tests need to be set up using pytest, Mocha or any other module which will be decided based on the tech stack.
   ii. Tests for the endpoints need to be written.

User Workflow

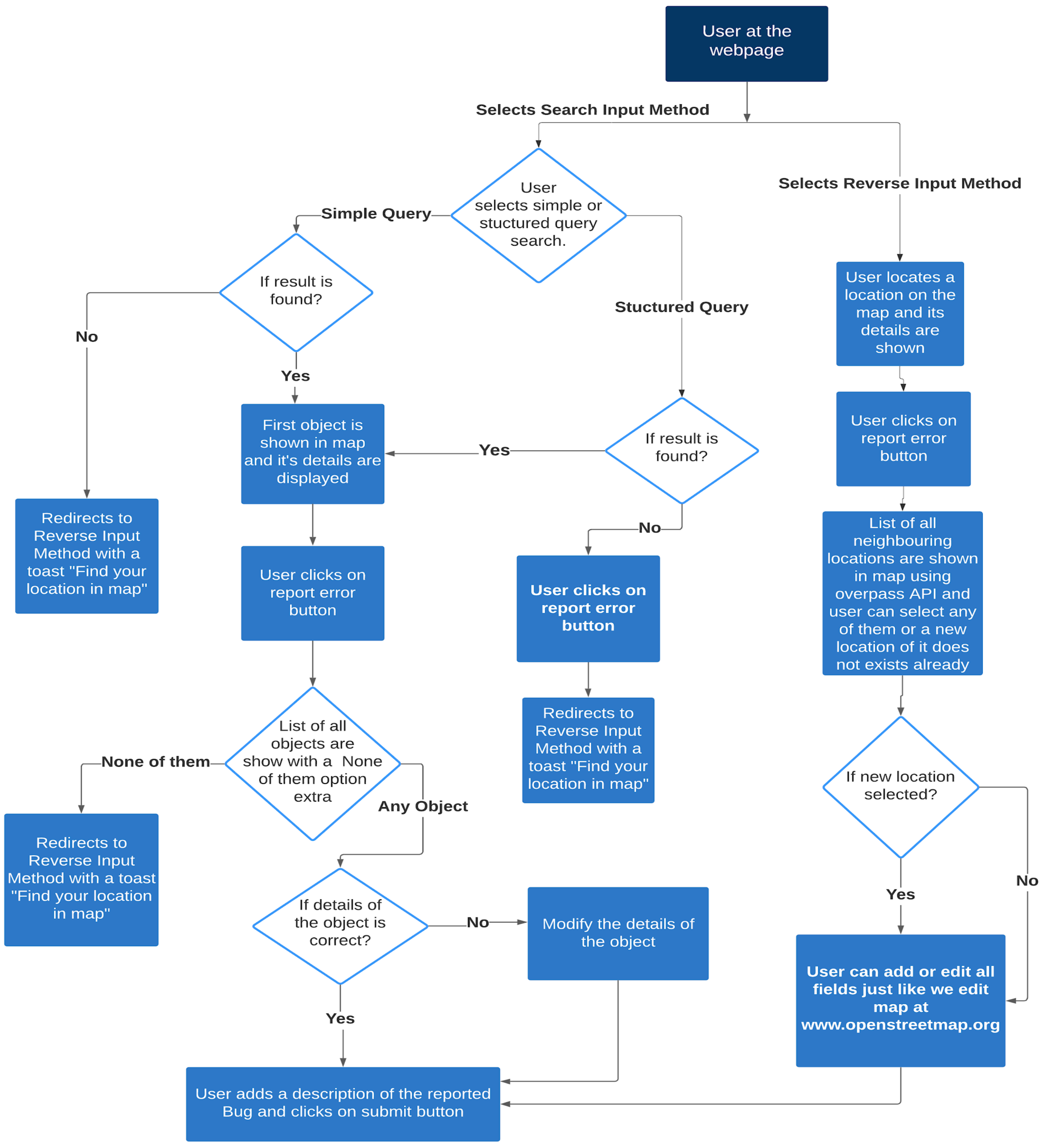

User at the webpage
Selects Search Input Method
Selects Reverse Input Method
User selects simple or stuctured query search.
Simple Query
Stuctured Query
If result is found?
No
Yes
First object is shown in map and it's details are displayed
Redirects to Reverse Input Method with a toast "Find your location in map"
User clicks on report error button
If result is found?
Yes
No
User clicks on report error button
Redirects to Reverse Input Method with a toast "Find your location in map"
List of all objects are show with a None of them option extra
None of them
Any Object
Redirects to Reverse Input Method with a toast "Find your location in map"
If details of the object is correct?
No
Modify the details of the object
Yes
User adds a description of the reported Bug and clicks on submit button
User locates a location on the map and its details are shown
User clicks on report error button
List of all neighbouring locations are shown in map using overpass API and user can select any of them or a new location of it does not exists already
If new location selected?
Yes
No
User can add or edit all fields just like we edit map at www.openstreetmap.org

Figure: The User Workflow will be similar to this.

Use Cases

* Fields that needs to be changed or added depending on zoom or admin_level are:

- name (name of location to be addressed as)
- Street, City, County, State, Country, Postcode
- type (type of location: administrative, unclassified, country, postcode, state, etc)
- class (place, boundary, highway etc)
- We can also modify or delete its addresses by selecting one.

a. Result order or details is incorrect in the “simple or structured query search input method”.

Flow of User

i. Users will first need to enter a query in the interface.
ii. The first object is shown.
iii. User selects the “Report bug and shows more locations” button
iv. A list of locations is shown, and users select one from them.
v. The user can modify the details shown for the location if wrong.
vi. The user adds the description of his bug and clicks on submitting a bug button.

Information collected to reproduce report:

- Simple or Structured query data
- Language
- Viewbox
- Location of the user if allowed
- osm_id, osm_type, class of the first and selected location.
- All modified fields of the location and previous field values, if any.
- Description of the bug as reported.

Causes of bug can be:

- User didn’t select the viewbox to be true.
- Tagging issue some street may not exist but is still tagged
- Used the wrong special phrase.
- Details of the data can be wrong.
- Can be due to less importance value.
- Language issue.

b. The result is empty or not in the list in the "simple or structured query search input method".

Flow of User

i. Users will first need to enter a query in the interface.
ii. The first object is shown.
iii. User selects the "Report bug and shows more locations" button, then selects none of them, then is redirected to Reverse input method or is directly redirected if no result is found.
iv. The user selects a location on the map and can use a tiny integrated scrollbar to choose different location ranks. User can also choose locations nearby using overpass API. If location is not found, the user can also choose a location from which he clicked.
v. User can add or modify details of the selected location.
vi. The user adds the description of his bug and clicks on submitting a bug button.

Information collected to reproduce report:

- Simple or Structured query data
- Language
- osm_id, osm_type, class of selected location or all the newly specified added location details if the new location is set.
- All modified fields and previous field values if modified.
- Description of the bug as reported.

Causes of bug can be:

- Wrong data present in location.
- The user used wrong special phrase.
- Wrong rank of location.
- Tagging issue some street may not exist but is still tagged or mistagging is present.
- Location might be broken.
- Location does not exist in OSM.
- Structured query parameter does not exist in OSM data.

c. Reverse input method does not contain location or has incorrect details.

Flow of User

i. The user selects a location on the map and can use a tiny integrated scrollbar to choose different location ranks.
ii. User can also choose locations nearby using overpass API. If location is not found, the user can also choose a location from which he clicked.
iii. User can add or modify details of the selected location.
iv. The user adds the description of his bug and clicks on submitting bug button.

Information collected to reproduce report:

- Zoom or location rank selected.
- Latitude and longitude of the clicked location.
- osm_id, osm_type, class of selected location or all the newly specified location details if the new location is set.
- All modified fields and previous field values if modified.
- Description of the bug as reported.

Causes of bug can be:

- Location might be broken.
- Location does not exist in OSM.
- Location details can be wrong.
- Nearby locations details were incorrect.

Testing report using Geocode Tester

a. Exporting log files to Geocode Tester

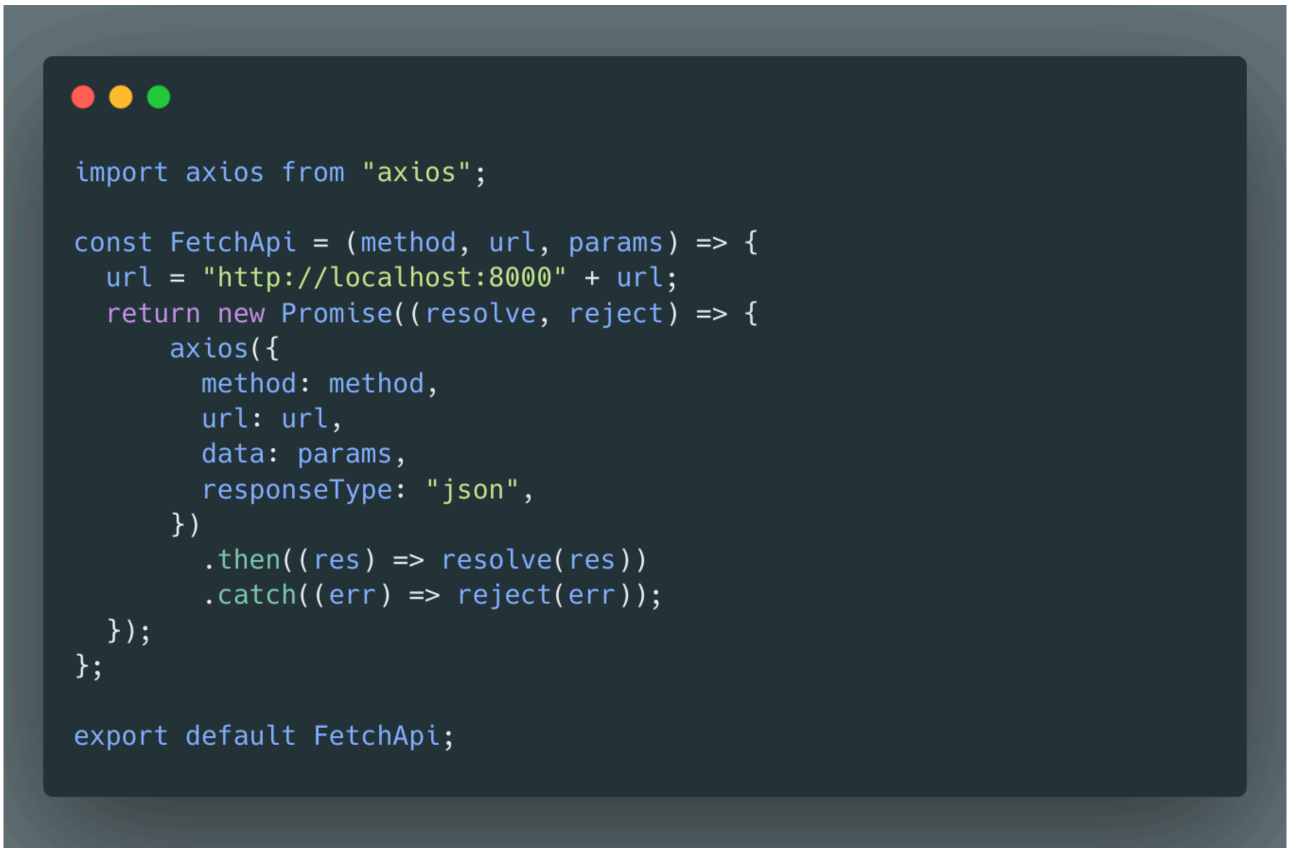
i. For Geocode-Tester to work, we need to add a query and some expected_xxx field through which Geocode-Tester will check the query and expected_xxx field value are equal.
ii. Suppose the order of fields is wrong in the bug report. Then we can easily add the query as well as the expected_xxx values from the correct location.
iii. If our result details are wrong, we can add the query as the display_name or label value and the correct values as in expected_xxx format. We can also add the latitude and longitude of the location.

iv. If the location does not exist, then we will be adding the latitude, longitude of the correct location, query as the display_name or label and expected_xxx values will be all the values added in the details section.

Frontend Workflow

a. Initialise the frontend framework

i. Setup the decided frontend framework.
ii. Create folder structures for maintainable code.
iii. Create Utilities for fetching data from Nominatim and sending data to the backend service.



```
import axios from "axios";

const FetchApi = (method, url, params) => {
  url = "http://localhost:8000" + url;
  return new Promise((resolve, reject) => {
      axios({
        method: method,
        url: url,
        data: params,
        responseType: "json",
      })
        .then((res) => resolve(res))
        .catch((err) => reject(err));
  });
};

export default FetchApi;
```

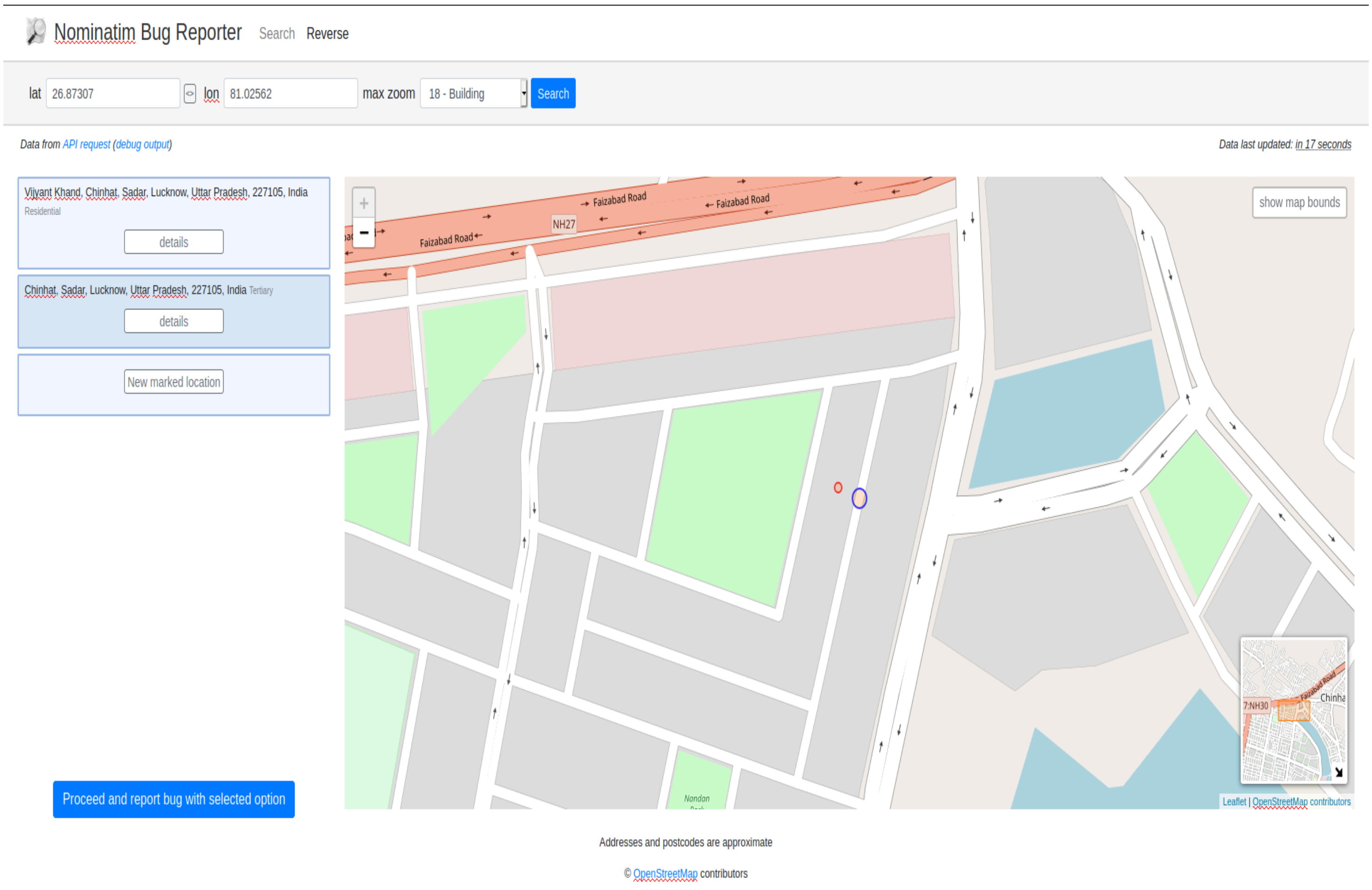
Figure: Illustrates how data can be sent to the backend using the FetchApi utility

b. **Create pages for the Interface as per the design and user workflow**

Reverse Search Method:

i. As per the timeline given below and the user workflow, we will first create the "Reverse Search Input Method" page, which will be similar to Nominatim's "Reverse Search" Method. But here, on clicking on the location in the map, we will be using the overpass API, which will show more places nearby, not just one location.

ii. We can also select different zoom levels, which will show other locations on the map depending on its admin_level. If we cannot find our choice location, we can also select "New marked location", which is the location where we clicked first.


Nominatim Bug Reporter Search Reverse

lat 26.87307 lon 81.02562 max zoom 18 - Building Search

Data from API request (debug output)

Data last updated: in 17 seconds

Vijyant Khand, Chinhat, Sadar, Lucknow, Uttar Pradesh, 227105, India
Residential
details

Chinhat, Sadar, Lucknow, Uttar Pradesh, 227105, India Tertiary
details

New marked location

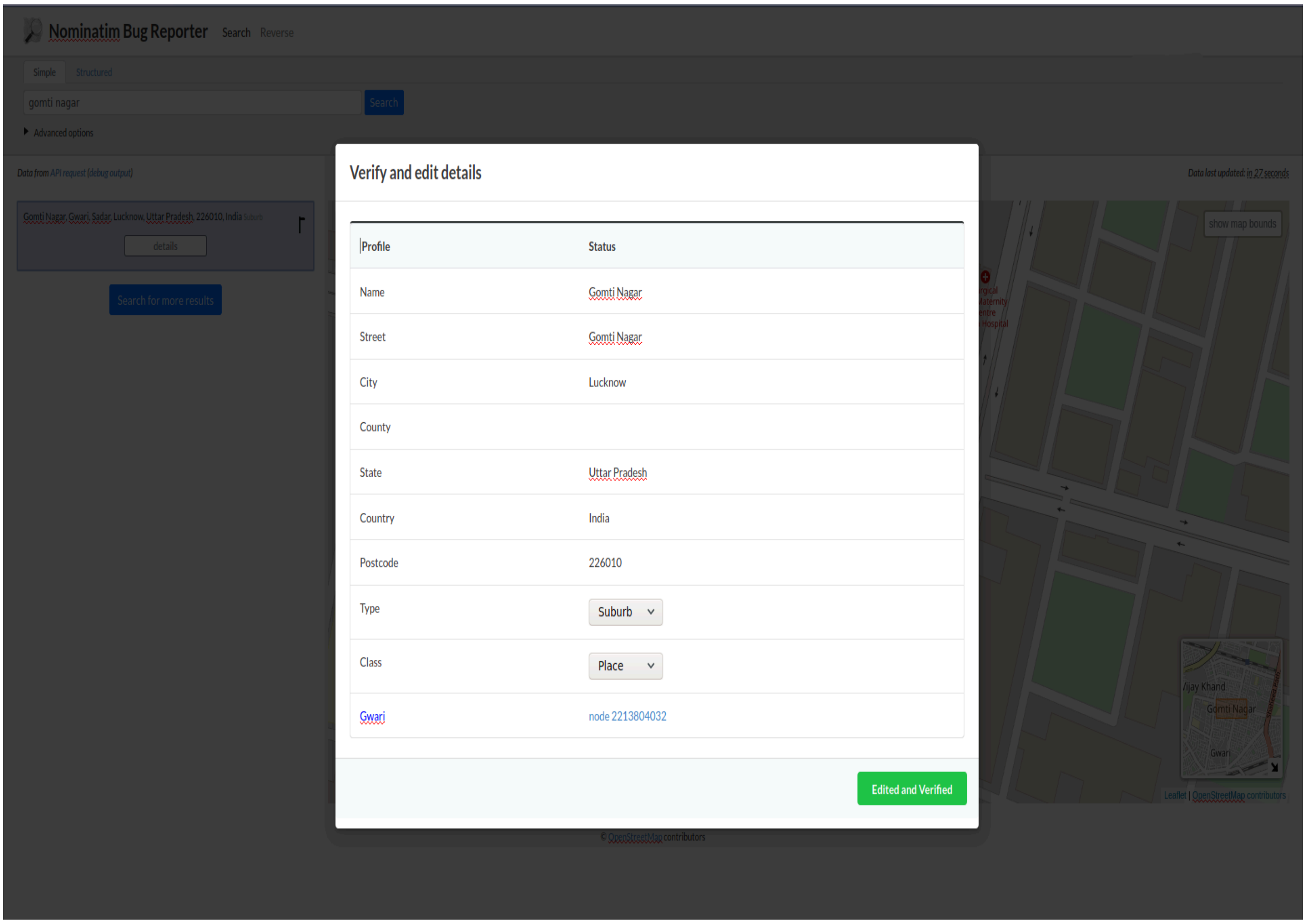
Proceed and report bug with selected option

Addresses and postcodes are approximate

© OpenStreetMap contributors

Figure: Illustrates how the Reverse search method will look on clicking a location.

iii. When we click to “proceed and report bug” button, a Modal will be seen, which will be identical in Simple Search Input Method as well where we can edit or verify field and then a bug description Modal will be seen which will send the bug with comments. The data will be fetched from lookup and details endpoint of Nominatim.


Nominatim Bug Reporter
Search
Reverse
Simple
Structured
gomti nagar
Search
Advanced options
Data from API request (debug output)
Gomti Nagar, Gwari, Sadar, Lucknow, Uttar Pradesh, 226010, India Suburb
details
Search for more results
Data last updated: in 27 seconds
show map bounds
Leaflet | OpenStreetMap contributors
© OpenStreetMap contributors

Verify and edit details

Profile	Status
Name	Gomti Nagar
Street	Gomti Nagar
City	Lucknow
County	
State	Uttar Pradesh
Country	India
Postcode	226010
Type	Suburb
Class	Place
Gwari	node 2213804032

Edited and Verified

Figure: Illustrates Modal which can add, edit or verify details of the location.

Search Input Method:

I. In this search method, we will be just writing either a simple or structured query that can produce zero or one results. If no result is found, we will be redirected to the "Reverse Search Input Method", where we will need to select the correct location and follow the user workflow.

II. If a result is found, the first result will be shown, and as per user workflow, we will be selecting the "Report button and show more locations". Which will show all locations, and we have to choose one, then we will be shown modal as was demonstrated in "Reverse Search Method." After which we have to write the bug description.

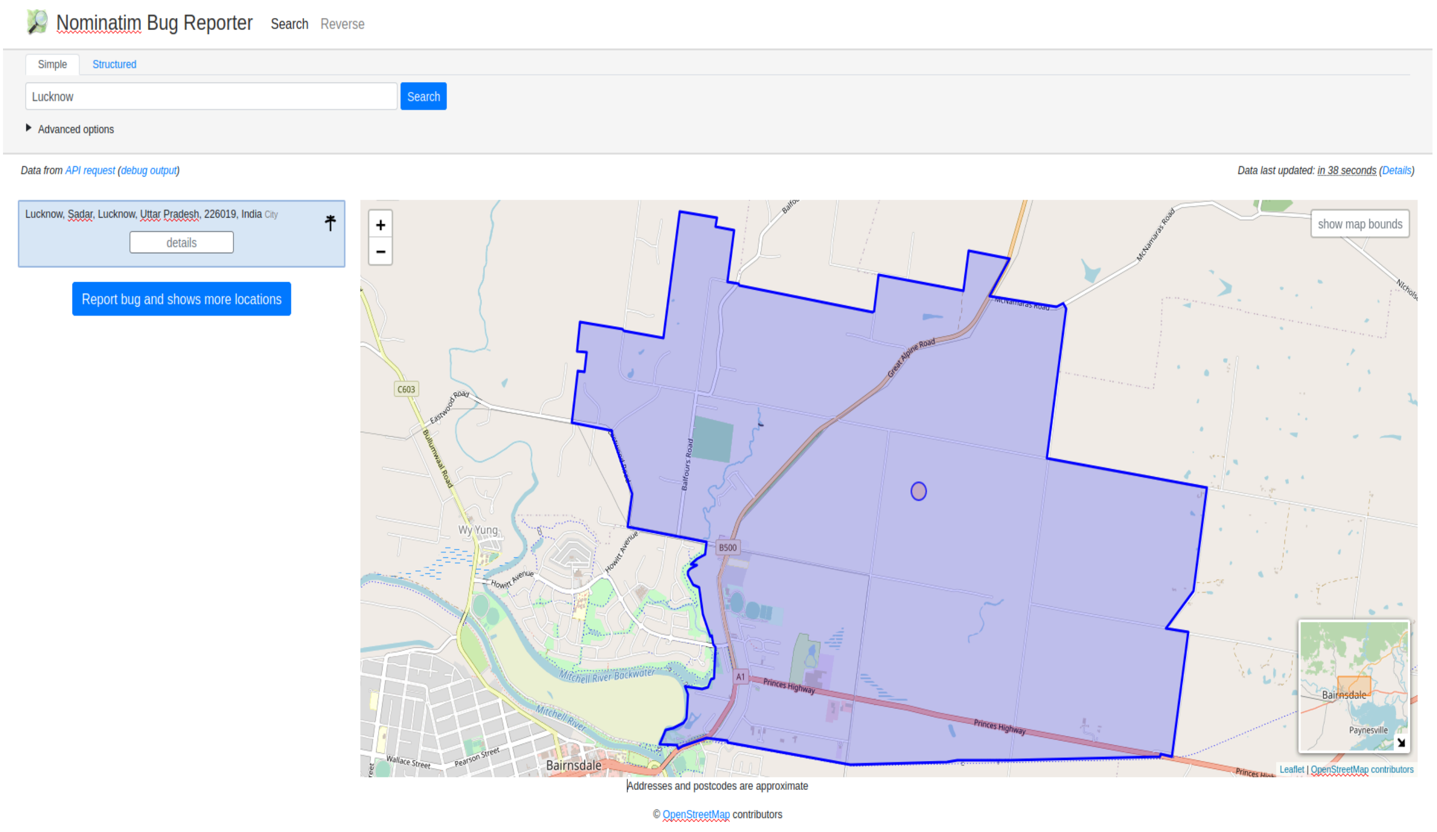

Nominatim Bug Reporter
Search
Reverse
Simple
Structured
Lucknow
Search
Advanced options
Data from API request (debug output)
Data last updated: in 38 seconds (Details)
Lucknow, Sadar, Lucknow, Uttar Pradesh, 226019, India City
details
Report bug and shows more locations
show map bounds
Bairnsdale
Paynesville
Leaflet | OpenStreetMap contributors
Addresses and postcodes are approximate
© OpenStreetMap contributors

Figure: Illustrates the page where we see only one result after Simple or Structured Search.

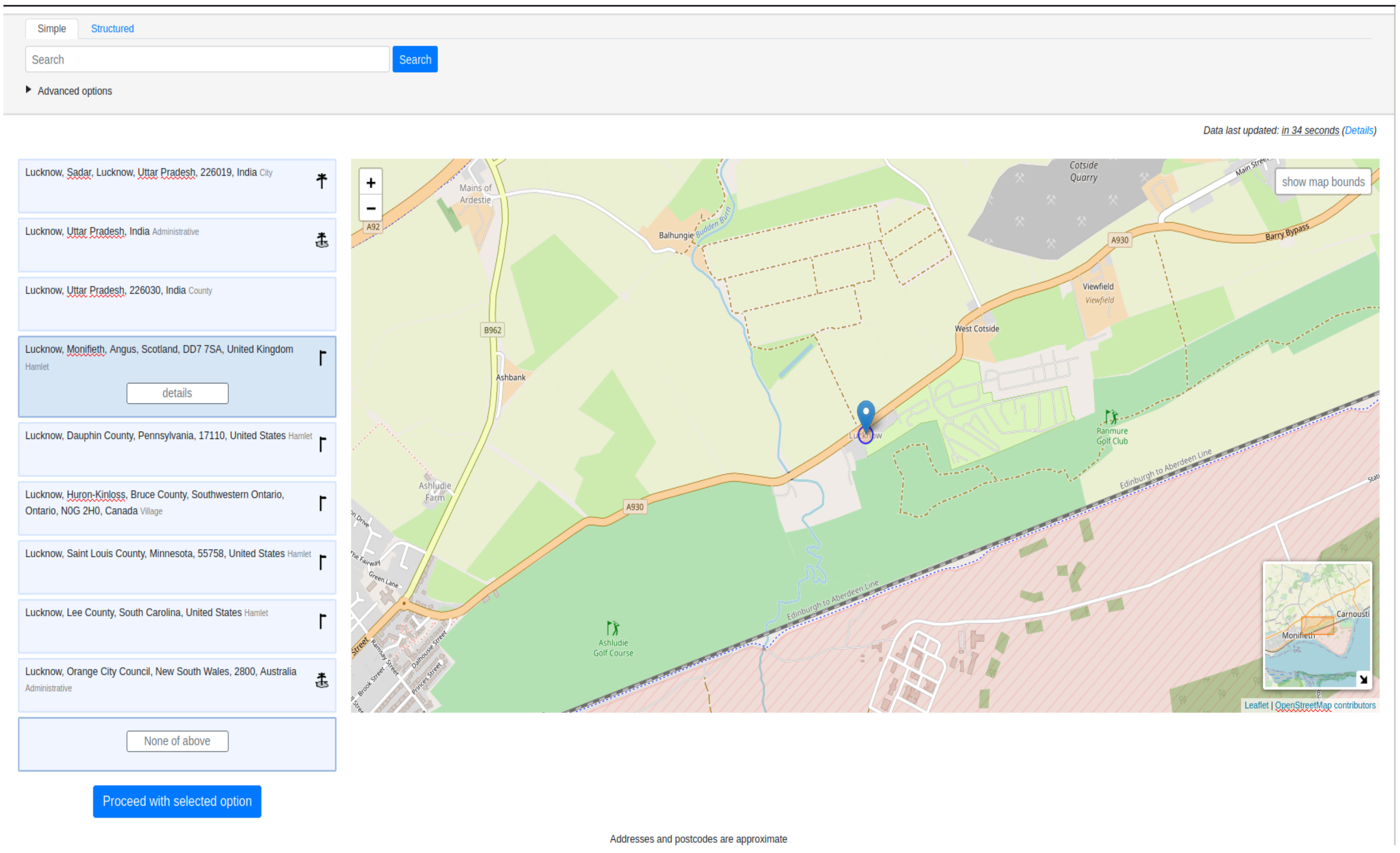

Simple
Structured
Search
Search
Advanced options
Data last updated: in 34 seconds (Details)
Lucknow, Sadar, Lucknow, Uttar Pradesh, 226019, India City
Lucknow, Uttar Pradesh, India Administrative
Lucknow, Uttar Pradesh, 226030, India County
Lucknow, Monifieth, Angus, Scotland, DD7 7SA, United Kingdom Hamlet
details
Lucknow, Dauphin County, Pennsylvania, 17110, United States Hamlet
Lucknow, Huron-Kinloss, Bruce County, Southwestern Ontario, Ontario, N0G 2H0, Canada Village
Lucknow, Saint Louis County, Minnesota, 55758, United States Hamlet
Lucknow, Lee County, South Carolina, United States Hamlet
Lucknow, Orange City Council, New South Wales, 2800, Australia Administrative
None of above
Proceed with selected option
show map bounds
Leaflet | OpenStreetMap contributors
Addresses and postcodes are approximate

Figure: Illustrates the page where we see all results clicking the report button on the above image.

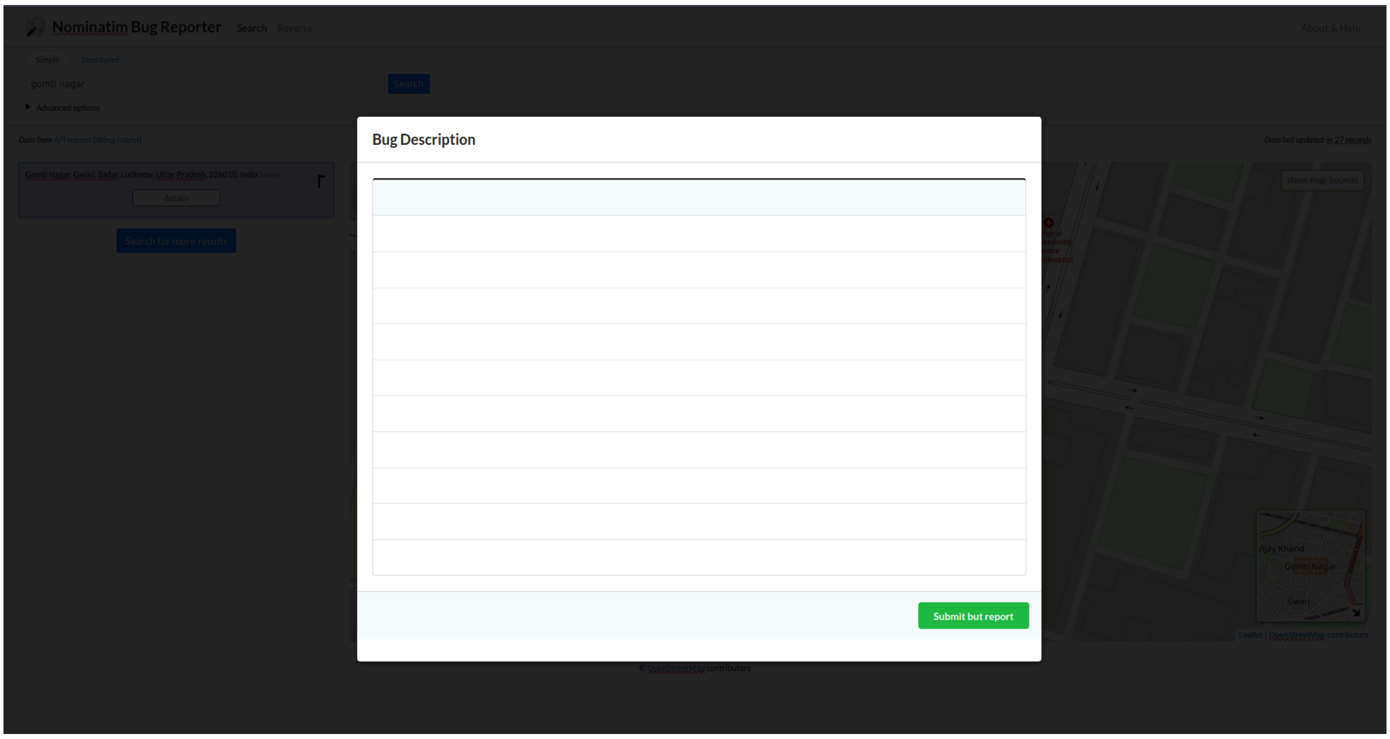
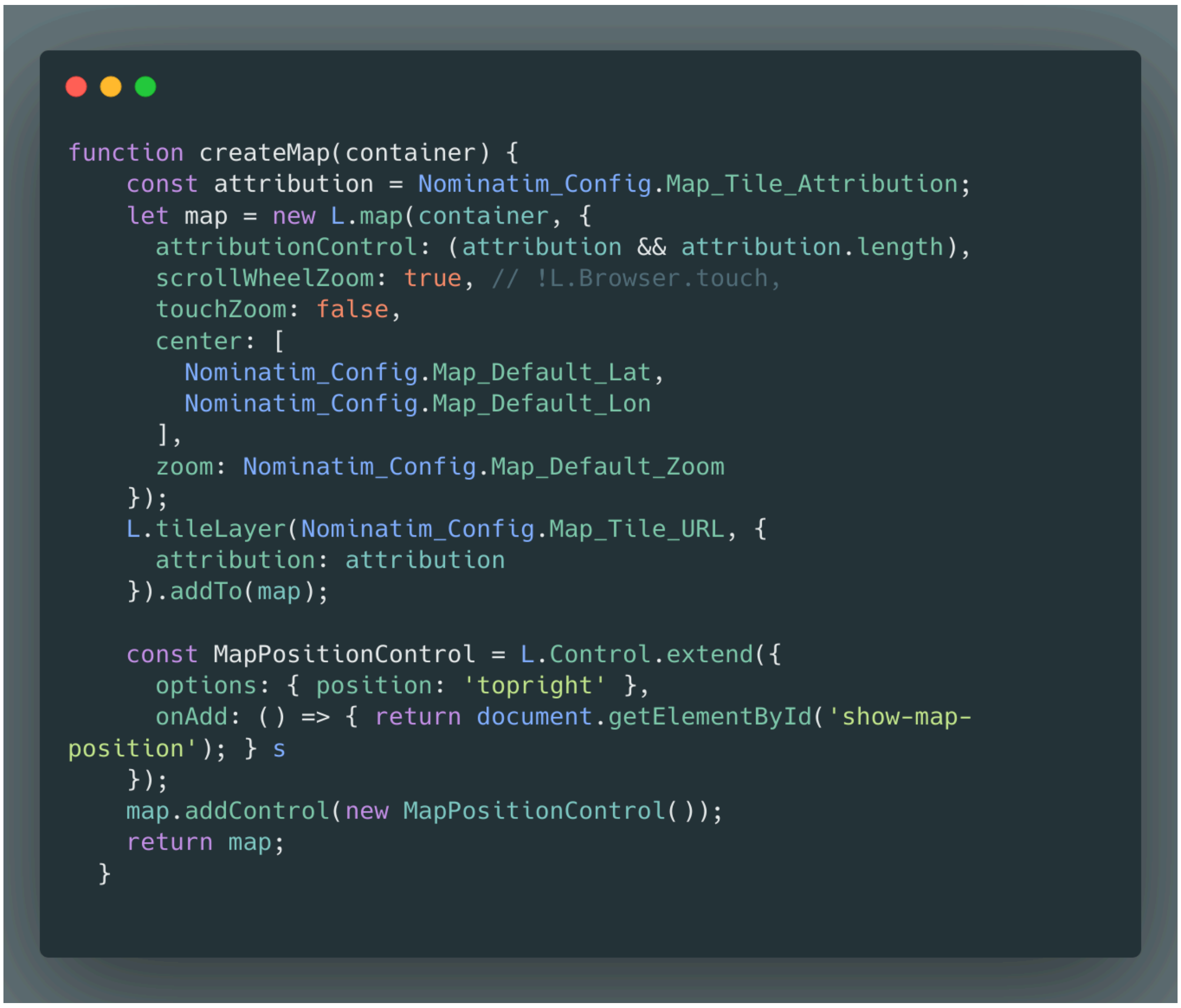


Figure: Illustrates the Modal where we have to write our bug description and submit a report.

c. **Interacting with maps:**

i. We will be using Leaflet for creating and interacting with the map. There are also various third-party libraries available for different frameworks like React Leaflet for **React** and Vue Leaflet for Vue.js.



```
function createMap(container) {
    const attribution = Nominatim_Config.Map_Tile_Attribution;
    let map = new L.map(container, {
      attributionControl: (attribution && attribution.length),
      scrollWheelZoom: true, // !L.Browser.touch,
      touchZoom: false,
      center: [
        Nominatim_Config.Map_Default_Lat,
        Nominatim_Config.Map_Default_Lon
      ],
      zoom: Nominatim_Config.Map_Default_Zoom
    });
    L.tileLayer(Nominatim_Config.Map_Tile_URL, {
      attribution: attribution
    }).addTo(map);

    const MapPositionControl = L.Control.extend({
      options: { position: 'topright' },
      onAdd: () => { return document.getElementById('show-map-
position'); } s
    });
    map.addControl(new MapPositionControl());
    return map;
  }
```

Figure: Illustrates how we can create a Map with Position control with Leaflet

d. **Integrate frontend with backend**

i. Integrating the REST API to send the data to the server.

****The images regarding frontend can be found here**

Project Deliverables

- A responsive web-based Interface for reporting bugs.
- The backend can export data to CSV files so that Geocoder-Tester can process it.
- Document README.md file for all repositories.
- Write blogs on the work done weekly or fortnightly.
- Write the GSoC end project report.
- Any other feature mentioned in further details that are different from what mentioned here.

Bucketlist

If all of the above-mentioned tasks are complete before the GSoC timeline ends and the time permits, I'd like to improve this project in the following way:-

- Dockerizing the entire codebase.
- Adding Continuous-Integration pipeline through Github-Actions or Travis CI.
- Add i18n support to the frontend for Internationalization.
- Make the frontend WCAG 2.0 accessible.

THE TIMELINE

I will be posting weekly or fortnightly blogs of my work. I am not adding that in the timeline to prevent redundancy.

First Phase	
Period	**Tasks**
May 17 - June 7 (Community Bonding)	❏ Collect relevant data from the mentors. ❏ Work on mapping practices for the OSM project. ❏ Dive deeper into OSM and Nominatim data model. ❏ Work on the flow of the app. ❏ Workout on wireframes of Interface. ❏ Discuss and finalise the flow and tech stack of the app. ❏ Learn about any new tech stack (language, library, package, etc.) if required.

June 7 - June 14 (Week 1)	❏ Set Up a backend server as discussed. ❏ Initialise the frontend framework ❏ Setup Tests for both frontend and backend. ❏ Create various utilities for communicating with API's ❏ Add tests for those utilities. ❏ Create a navbar and set up a router for navigating through pages
June 15 - June 22 (Week 2)	❏ Create core components as per the mockup. ❏ Create REST API endpoints for reporting bugs through the reverse input method ❏ Add Tests of the frontend and backend
June 23 - June 30 (Week 3)	❏ Create views as per design and user workflow to report bugs through the reverse input method. ❏ Integrate the reverse input method frontend and backend structure. ❏ Create REST API endpoints for reporting bugs through the search input method. ❏ Add Tests for backend and some core frontend functionality.
July 1 - July 8 (Week 4)	❏ Create views for a search input method for a simple query. ❏ Integrate frontend and backend for a search input method for a simple query.
July 9 - July 16 (including Phase 1 evaluation) (Week 5)	❏ Fix bugs or UI issues in the frontend ❏ Phase 1 evaluation, finalize the completed task.

Expected Deliverables after First Phase

❏ A responsive application that can report bugs from the reverse input method and a simple query search input method.
❏ Working backend with endpoints to report bugs and bugs are logged into a file.
❏ Added Tests for backend and core frontend functionalities.
❏ Written blogs about my work.

Final Phase	
July 16 - July 23 (Week 6)	❏ Create views for a search input method for the structured query. ❏ Integrate frontend and backend for a search input method for the structured query. ❏ Add Tests for backend and some core frontend functionality.
July 24 - July 31 (Week 7)	❏ Fix bugs or UI issues in the frontend ❏ Create a script to convert the logs to CSV files so to be processed through Geocoder-Tester. ❏ Add Tests for the script to export logs to CSV files that can be processed through Geocoder-Tester.
August 1 - August 8 (Week 8)	❏ Workaround any other feature introduced prior to the project start ❏ Document the service neatly describing how-to-use
August 9 - August 16 (Week 9)	❏ Final, touch all the things and fill the gaps if any are left from the previous week's tasks. ❏ Write the GSoC end project report.
August 17 - August 23 (Week 10)	❏ The final evaluation, finalise all small to-dos ❏ Submit the report.

MOTIVATION

Google Summer of Code is an excellent platform to get acquainted with the open-source community and their skilful mentors. It gives one a professional work experience in their college years where they collaboratively build a product for the welfare of the society. In this process, both the individual and the community improve and grow.

This project has the potential to let me explore more in the field of web development. It will let me involve myself in every aspect of a product cycle from designing wireframes, writing backend and frontend, to writing tests and documentation for it collaboratively.

EXPECTATIONS

It is a great opportunity for me to apply my knowledge on a practical scale. I wish I can be a part of this program for gaining experience in software development and give back to the open-source community.

This project can be best beneficial for a full stack developer like me for this summer. This project will offer me to explore many fields that are untouched by me. I would become much more aware of the open-source community and will gain experience in collaborative working.

POST GSoC

I am already learning a lot by contributing to various open-source projects. I came across the many things that were new and enlightening to me like Tests, CLI's, Documentations, and conventions and conduct while contributing to open-source. It really feels good to give back to the community, helping other people. After completion of this programme, I would continue to contribute to this project including items included in my Bucket List. I would be happy to help and mentor new open-source enthusiasts interested in this project.

*****