

Atlas

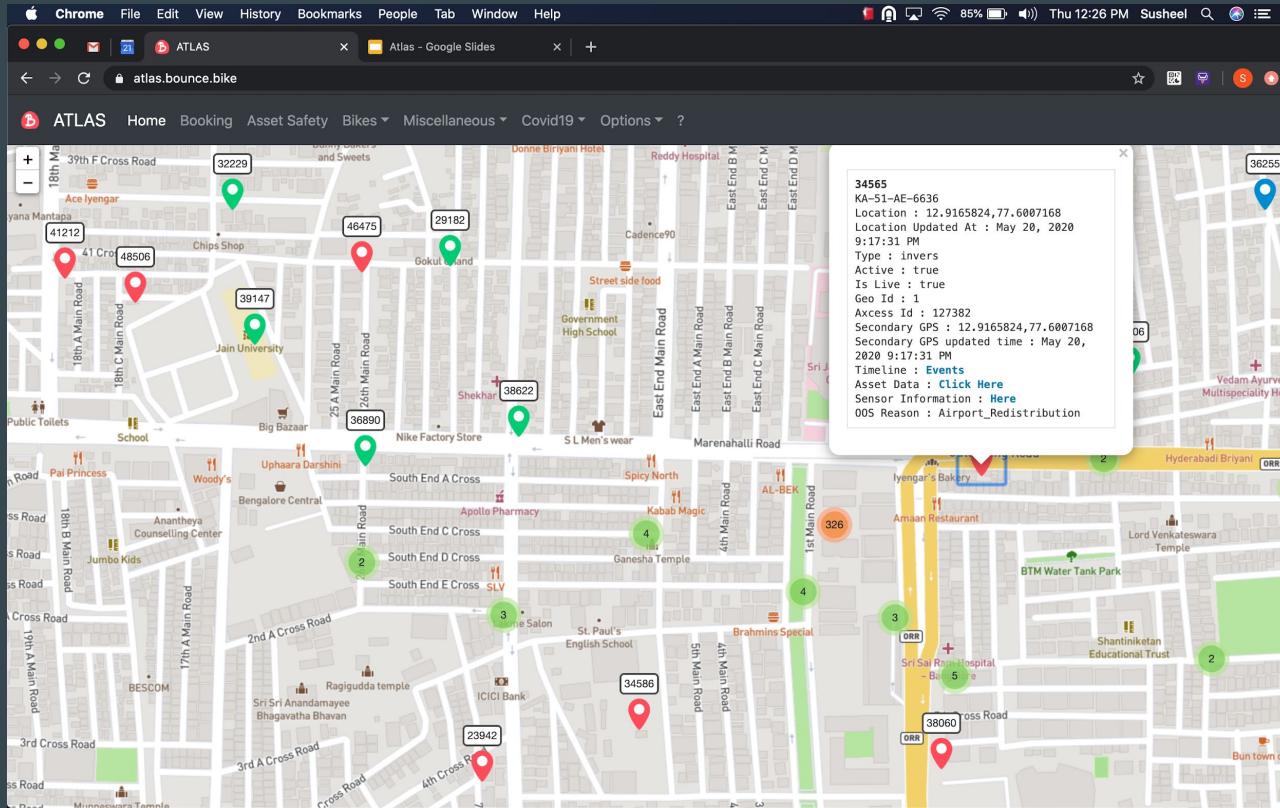
• • •

An OpenSource Internal Admin-Panel Framework
github.com/bounceshare/atlas

Features

- Map Controls - Markers, Fences, Circles & Paths
- GeoJSON Support
- Postgres CRUD UI
- Customizable Search Controls
- Config Driven
- Access Control
- Standalone Deployment Support
- Mobile Responsive
- AtlasSDK
- Backend Driven (so devs don't have to fiddle around with UI code)

Screenshots



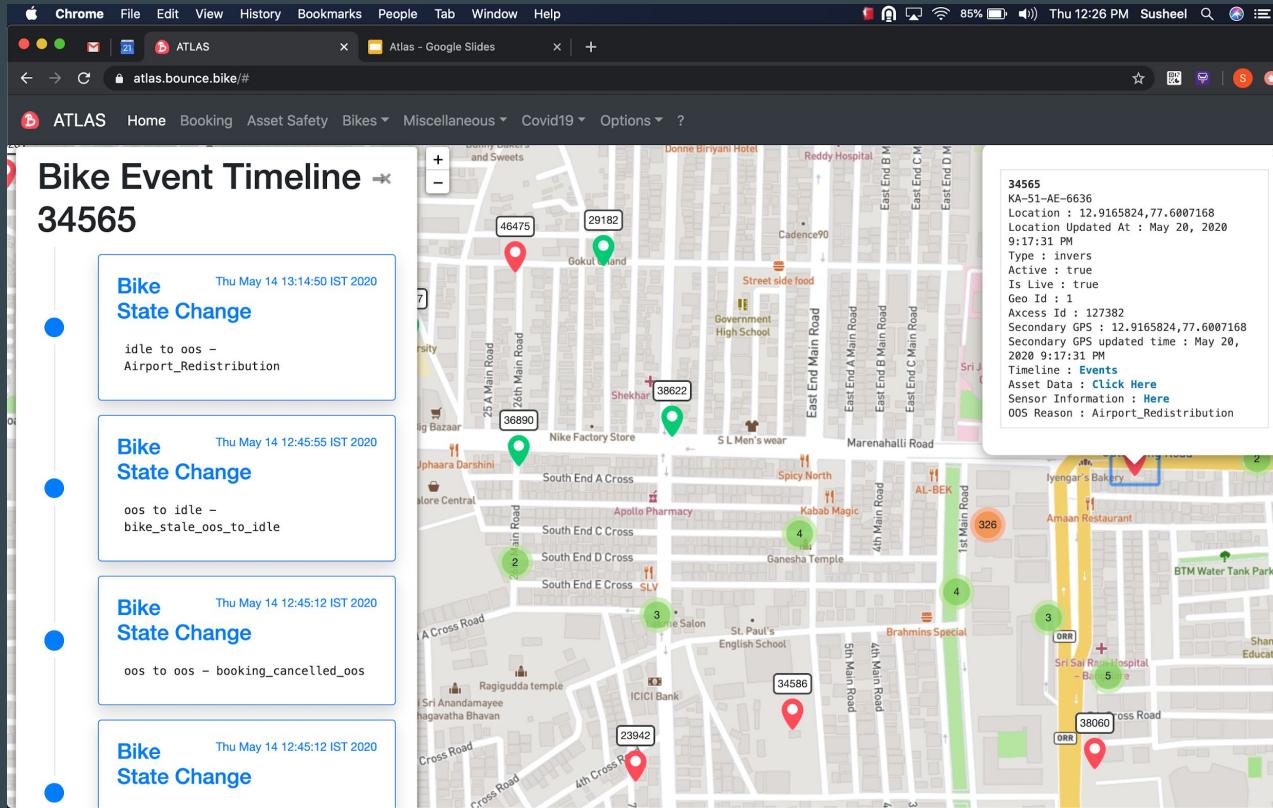
Screenshots

The screenshot shows a web browser window with the ATLAS application running on a map of a city area. A modal window titled "Bike Sensor Values - 34565" is displayed, containing "Latest Sensor Information" from "Thu May 21 12:26:27 IST 2020". The device ID listed is D240DCB818576E02. The sensor data includes:

```
ext_battery_level : 5.460000038146973
last_contact_at : 2020-05-20 10:59:29.671
battery_level_at : null
device_id : D240DCB818576E02
est_fuel_level_at : 2020-04-24 16:08:12.785
id : 1767203435
int_battery_level : null
sensor_fuel_level_at : 2020-04-24 16:08:12.927
est_fuel_level : 4.585582609304379
sensor_fuel_level : 2.0
```

At the bottom of the modal are "More" and "Close" buttons. The background map shows several red location markers and green circular icons with the number "2" on them.

Screenshots



Screenshots

The screenshot shows a Google Chrome browser window with the following details:

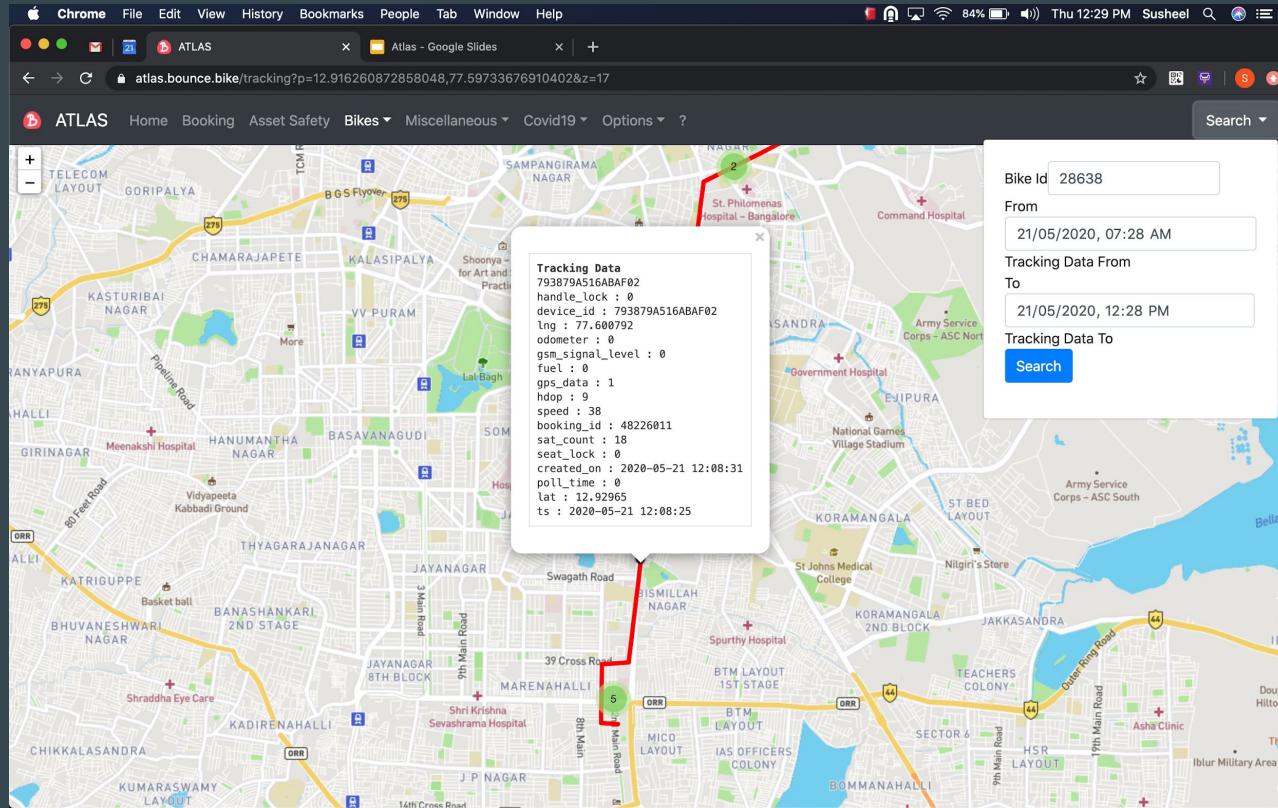
- Address Bar:** atlas.bounce.bike/config
- Tab Bar:** ATLAS (active tab), Atlas - Google Slides
- Toolbar:** Standard Chrome icons for Back, Forward, Stop, Refresh, and others.
- Header:** ATLAS Home Booking Asset Safety Bikes ▾ Miscellaneous ▾ Covid19 ▾ Options ▾
- Content Area:** A form titled "Config Json" containing a JSON configuration block. The JSON is as follows:

```
{
  "defaultLocation": "12.9160463,77.5967117",
  "title": "ATLAS",
  "favicon": "/resources/icons/favicon.ico",
  "logo": "/resources/icons/bounce.png",
  "zoom": "17",
  "tabs": [
    {
      "page": "Home",
      "pageId": "Home",
      "path": "/",
      "searchUrl": "/apis/search/",
      "searchText": "",
      "autoRefresh": "true",
      "help": "Move around the map and Atlas will load the information around those coordinates",
      "zoom": 0
    },
    {
      "tabName": "Bikes",
      "pages": [
        {
          "page": "Search For Bikes",
          "pageId": "Bikes"
        }
      ]
    }
  ]
}
```

Bottom of the Form:

- A blue "Submit" button.

Screenshots



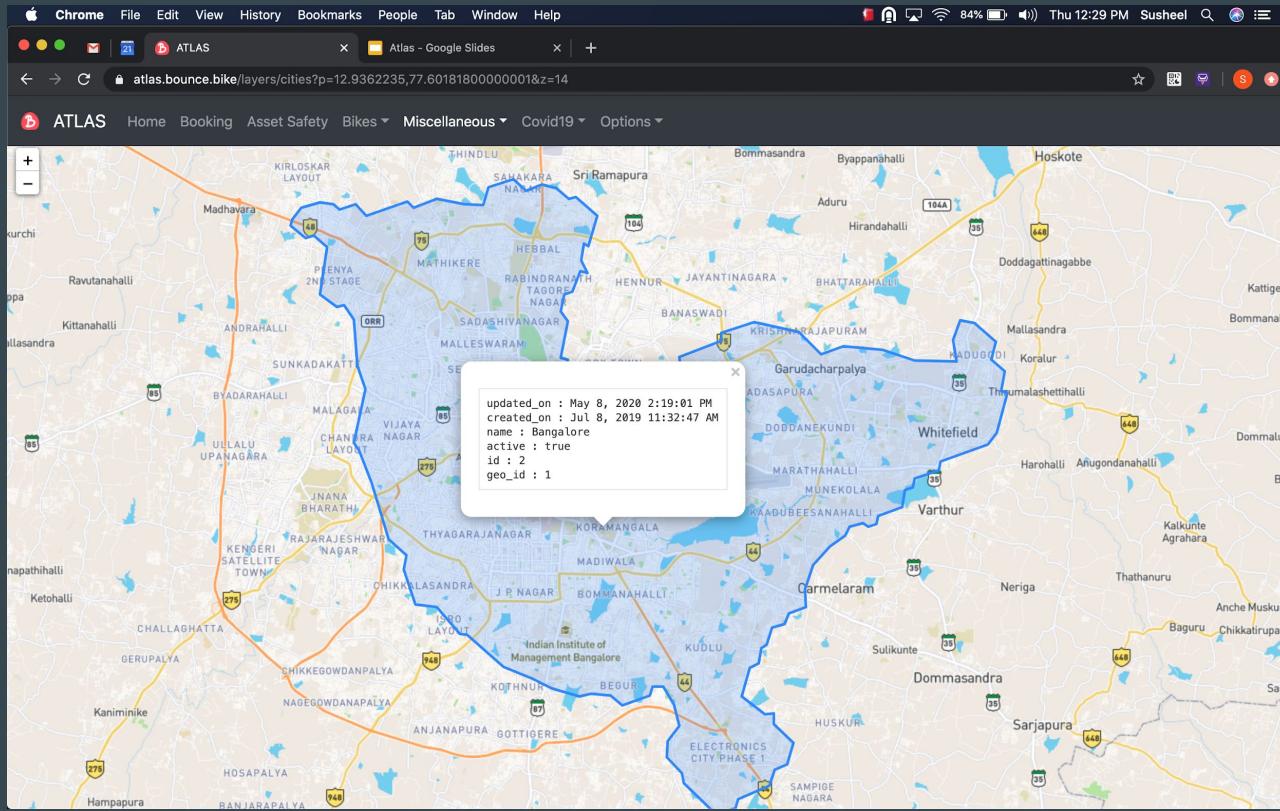
Screenshots

The screenshot shows a web browser window with the following details:

- Browser Header:** Chrome, File, Edit, View, History, Bookmarks, People, Tab, Window, Help.
- Address Bar:** atlas.bounce.bike/config
- Page Title:** ATLAS
- Page Content:** A form titled "Config Json" containing JSON code. The code defines a configuration object with properties like "page", "pagelid", "path", "searchUrl", "searchPage", "searchText", "autoRefresh", "help", "zoom", and "searchDataSchema". The "searchDataSchema" object contains fields for "bikeld", "from", and "to".

```
{
  "page": "Tracking",
  "pagelid": "Bikes",
  "path": "/tracking",
  "searchUrl": "/apis/tracking/search",
  "searchPage": "true",
  "searchText": "Bikeld <from> <to>",
  "autoRefresh": "false",
  "help": "You can search for tracking data for a bike and filter on time. The format to search for it is - <Bikeld> 2020-02-13T23:59:59 2020-02-13T20:59:59",
  "zoom": 0,
  "searchDataSchema": {
    "bikeld": {
      "title": "Bike Id",
      "description": "",
      "type": "number"
    },
    "from": {
      "title": "From",
      "description": "Tracking Data From",
      "type": "datetime-local"
    },
    "to": {
      "title": "To"
    }
  }
}
```
- Buttons:** A blue "Submit" button at the bottom left of the form.

Screenshots



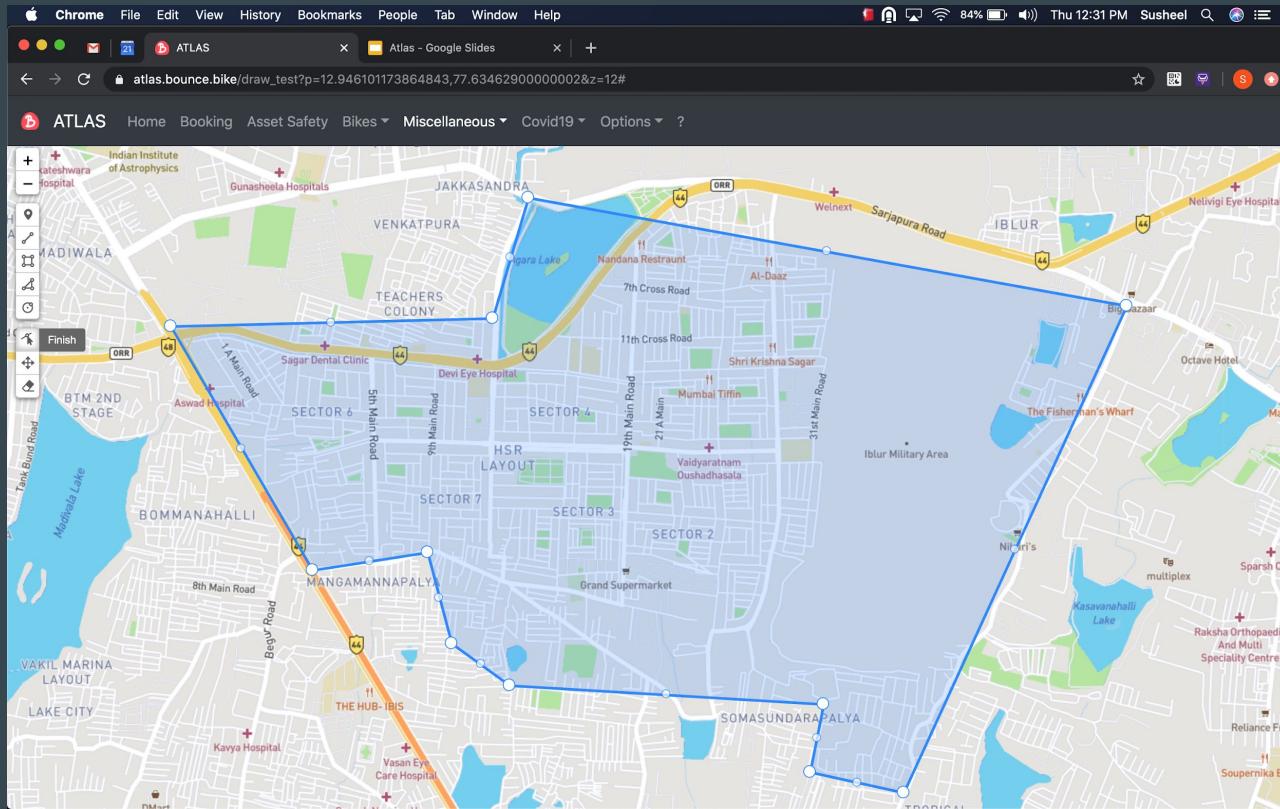
Screenshots

The screenshot shows a Google Chrome browser window with the following details:

- Address Bar:** atlas.bounce.bike/config
- Title Bar:** ATLAS - Google Slides
- Page Content:** The main content area displays a JSON configuration object. The JSON starts with a closing brace `}` followed by a new opening brace `{` and then continues with the following code:

```
        },
{
  "page": "City Geofences",
  "pageId": "Miscellaneous",
  "path": "/layers/cities",
  "autoRefresh": "true",
  "zoom": 0,
  "editControl": {
    "editFenceUrl": "/apis/test/drawnObjs",
    "isEditControlSupported": false
  },
  "geoJsonRecordConfig": {
    "jdbcUrl": "jdbc:postgresql://[REDACTED]",
    "[REDACTED]",
    "dbUsername": [REDACTED],
    "dbPassword": [REDACTED],
    "schema": "public",
    "table": "bounce_service_area",
    "geoJsonSqlQuery": "select st_asgeojson(fence) as geojson, * from bounce_service_area
where ST_DWithin(fence::geography, ST_MakePoint($lon,$lat)::geography, $radius)"
  }
}
```
- Buttons:** A blue "Submit" button is located at the bottom left of the form.
- Bottom Right:** A small inset window shows a terminal or command-line interface with some text visible.

Screenshots



Screenshots

The screenshot shows a Google Chrome browser window with the following details:

- Address Bar:** atlas.bounce.bike/config
- Page Title:** ATLAS - Google Slides
- Page Content:** A JSON configuration editor titled "Config Json". The JSON code is as follows:

```
        "searchUrl": "/apis/corona/testcentres/",
        "autoRefresh": "true",
        "help": "Move around the map and Atlas will load the information around those coordinates",
        "auth": "open",
        "defaultLocation": "17.62308179131177,77.95898437500001",
        "zoom": 6
    }
],
"zoom": 0
},
{
"tabName": "Miscellaneous",
"pages": [
{
"page": "ParkingV2",
"pageId": "Miscellaneous",
"path": "/layers/parkingv2",
"searchUrl": "https://appapi.bounce.bike/zone_service/search",
"searchText": "",
"autoRefresh": "true",
"help": "",
"zoom": 0
},
{

```

Bottom of the page:

Submit

Screenshots

The screenshot shows a web browser window with the title "ATLAS". The main content area displays a table of users with columns "id" and "name". Each row has "Edit" and "Delete" links. A modal window titled "Form Data" is open, containing fields for "id", "name", "mobile_num", "email", and "pagePath". The "pagePath" field contains the value "/test_users". At the bottom of the modal are "Submit" and "Close" buttons. Below the table is a navigation bar with "Previous", page numbers (1, 2, 3, 4, 5, ..., 10), and "Next" buttons, along with an "Add" button. The status bar at the bottom right shows a small screenshot of the application interface.

	id	name
Edit	1	SUJITH REDDY
Delete		
Edit	721	Sameer
Delete		
Edit	3	Ananth
Delete		
Edit	4	Narender
Delete		
Edit	5	Gnaneshwara
Delete		
Edit	6	Thaha pc
Delete		
Edit	7	Kiran Kumar Gov
Delete		
Edit	8	venu gopal
Delete		
Edit	9	Naveen kumar
Delete		
Edit	10	Yogesh Jain
Delete		

Form Data

id

name

mobile_num

email

pagePath

Submit Close

Previous 1 2 3 4 5 ... 10 Next

Add

Screenshots

The screenshot shows a web browser window with the title bar "Chrome" and various system icons. The address bar displays "atlas.bounce.bike/config". The main content area is titled "Config Json" and contains a JSON configuration object. The JSON structure includes fields for page settings, database configuration, and crud operations. Several fields in the JSON object have been redacted with black boxes.

```
Config Json
{
  "isEditControlSupported": true
}
},
{
  "page": "Test Users",
  "pageId": "Miscellaneous",
  "path": "/test_users",
  "searchPage": "true",
  "autoRefresh": "false",
  "help": "Mb Test Users",
  "zoom": 0,
  "crudConfig": {
    "isCreateAllowed": true,
    "isEditAllowed": true,
    "isDeleteAllowed": true,
    "dbcUrl": "jdbc:postgresql://[REDACTED]", [REDACTED],
    "dbUsername": [REDACTED],
    "dbPassword": [REDACTED],
    "schema": "public",
    "table": "mb_test_users"
  }
},
{
```

Submit

Screenshots

The screenshot shows a web browser window with the title "ATLAS" open at the URL atlas.bounce.bike/bike_inventory?p=12.945341132980488,77.65771865844728&z=13#. The browser's top bar includes the standard Apple logo, menu items like Chrome, File, Edit, View, History, Bookmarks, People, Tab, Window, Help, and system status icons.

The main content area displays a table of bike inventory data and a "Query Builder" modal window.

Table Data:

created_on	updated_on	access_code	axcess_id	created_								
2019-11-06 17:03:44.872796	2020-05-21 12:37:44.736	null	140586	null								
2019-08-28 19:17:37.627	2020-04-02 18:43:34.25775	6242	112716	530441								
2019-11-21 19:15:46.895	2020-04-02 18:43:35.06626	1234	99598	530441								
2019-12-31 19:47:45.857	2020-05-07 11:56:30.716	1234	140146	3233111								
2019-06-20 16:12:56.007546	2020-05-21 10:43:49.004	7254	136084	1045495								
2019-06-18 17:19:00.372583	2020-05-21 12:33:41.855	true	28395	41	KA-51-AD- 8845	12.896406	77.628387	idle	oos	1486	140738	530435
2019-10-10 18:53:08.768	2020-05-21 08:26:51.38	true	40369	41	KA-51-AF- 0693	12.982575	77.71212	oos	idle	1234	118151	49331
2019-06-20 07:45:03.563688	2020-05-21 12:37:44.738	true	28509	41	KA-51-AD- 8972	12.993501	77.747948	oos	idle	1612	139573	1045495
2019-10-14 19:59:25.062	2020-05-21 09:34:14.796247	false	40986	41	KA-51-AF- 0856	12.850304	77.6885248	oos	idle	1234	118948	1918832
2019-04-24 15:43:22.09962	2020-05-21 12:37:54.83	false	26318	41	KA-03-AG- 8931	13.047337	77.594658	oos	idle	3781	144630	530441

Query Builder:

The "Query Builder" modal contains a search interface with the following filters:

- license_plate: contains "1234"
- type: equal "invers"

Buttons in the modal include "Add rule", "Add group", "Search", and "Close".

Pagination at the bottom of the table indicates pages 1 through 10.

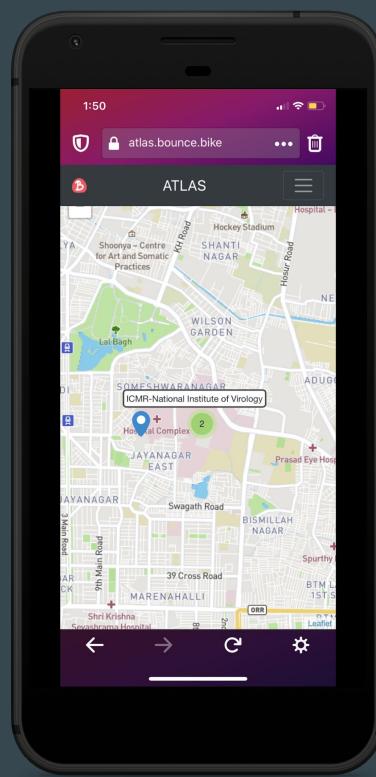
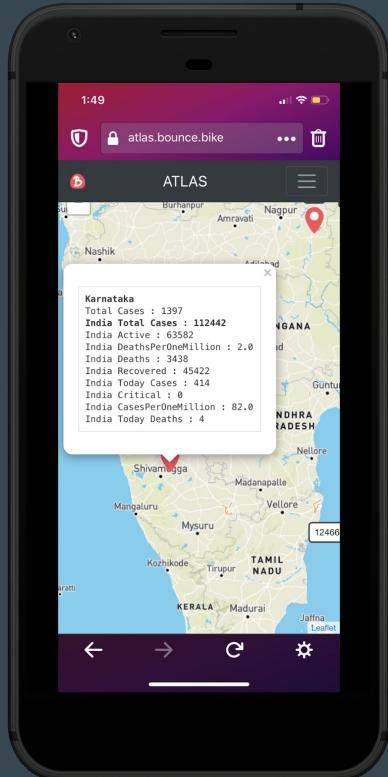
Screenshots

The screenshot shows a web browser window with the title bar "Chrome" and various system icons. The address bar displays "atlas.bounce.bike/config". The main content area is titled "Config Json" and contains a JSON object:

```
    {
      "page": "Bike Inventory",
      "pageld": "Miscellaneous",
      "path": "bike_inventory",
      "searchPage": "true",
      "autoRefresh": "false",
      "help": "Inventory of bikes",
      "zoom": 0,
      "crudConfig": {
        "isCreateAllowed": false,
        "isEditAllowed": false,
        "isDeleteAllowed": false,
        "jdbcUrl": "jdbc:postgresql://[REDACTED]_[REDACTED]",
        "dbUsername": "[REDACTED]",
        "dbPassword": "[REDACTED]",
        "schema": "public",
        "table": "bike"
      }
    },
    "zoom": 0
  },
```

At the bottom of the form is a blue "Submit" button.

Works on Mobile (Responsive)



Tech Stack

- JAX-RS using Jersey deployed on Jetty
- Leaflet.js + jQuery + Bootstrap 4
- Freemarker for templating

Example - Listing Bikes on Map

```
@POST
@Path("/search/{path:.*}")
@Produces(MediaType.APPLICATION_JSON)
@Consumes({MediaType.APPLICATION_JSON})
@GoogleAuth
public void bikeListing(String inputString, @Suspended final AsyncResponse asyncResponse, @PathParam("path") String path) {
    logger.info("/apis/search");
    LayersApi layersApi = new LayersApi(inputString, asyncResponse, httpRequest, httpResponse, path);
    layersApi.onRequest();
}
```

Example - Listing Bikes on Map

```
public class LayersApi extends BaseApiHandler {

    private String path;

    public LayersApi(String inputString, AsyncResponse asyncResponse, HttpServletRequest httpRequest,
                     HttpServletResponse httpResponse, String path) {
        super(inputString, asyncResponse, httpRequest, httpResponse);
        this.path = path;
    }

    @Override
    public void onRequest(){
        try {
            super.onRequest();
        } catch (Exception e) {
            e.printStackTrace();
        }

        List<MarkerPojo> markers = Lists.newArrayList();
        List<FencePojo> fences = Lists.newArrayList();
        List<CirclePojo> circles = Lists.newArrayList();
        List<PathPojo> paths = Lists.newArrayList();

        double lat = input.optDouble( key: "lat", defaultValue: 12.9160463);
        double lon = input.optDouble( key: "lon", defaultValue: 77.5967117);
        int radius = input.optInt( key: "radius", defaultValue: 5000);
        String searchQuery = input.optString( key: "searchQuery", defaultValue: null);

        List<BikeRecord> bikes = QueryUtils.getBikes(lat, lon, limit: 20000, radius, status: null, searchQuery);
        List<MarkerPojo> bikeMarkers = RenderUtils.getBikesAsMarkers(bikes);
        markers.addAll(bikeMarkers);

        Map<Object, Object> response = Maps.newHashMap();
        response.put("markers", markers);
        response.put("fences", fences);
        response.put("circles", circles);
        response.put("paths", paths);
        if(TextUtils.isEmpty(searchQuery)) {
            response.put("autoRefresh", true);
        } else {
            response.put("autoRefresh", false);
        }

        sendSuccessResponse(asyncResponse, response);
    }
}
```