1. Your code will be evaluated for both functional and nonfunctional requirements such as  code quality, ease of setting up dev and production environment, load testing scripts etc.
2. Automated tests are mandatory, so please include tests/specs. Additionally, it’s a huge  plus if you test drive your code.
3. We are really, really interested in clean and maintainable codebase, so please solve the  problem keeping this in mind.
4. Please ensure that the coding conventions, directory structure and build approach of  your project follow the conventions set by popular open source projects in the language  that you’re using.

**Readme**

Readme for your project should include:

1. Overview of tech stack — language, web framework, database etc.
2. Brief description of why this tech stack was chosen
3. Infrastructure requirements for running your solution
4. Setup instructions, automated deployment of the program and dependencies to  development and test environment is a plus

**Where** **is** **My** **Driver**

We have 50,000 drivers who go around the city looking for rides. Typically, drivers are evenly distributed across the city. There are customers all over the place trying to find the driver. To facilitate this, we need to keep track of driver’s current location and provide an ability to search drivers in a given area. You need to build 2 APIs to achieve this.

Both APIs should respond within 100ms.

**1. Driver** **Location**

Drivers should be able to send their current location every 60 seconds. They’ll call following API to update their location

**Request:**PUT /drivers/{id}/location

{ "latitude": 12.97161923, "longitude": 77.59463452, "accuracy": 0.7

}

**Response:**

* -  200 OK — on successful update, Body: {}
* -  404 Not Found — if the driver ID is invalid (valid driver ids - 1 to 50000) Body: {}
* -  422 Unprocessable Entity — with appropriate message. For example:  {"errors":["Latitude should be between +/- 90"]}
* **Expected** **Load:**50,000 drivers sending location every 60 seconds. You can use this API to generate seed data to search drivers in a given area.

**2. Find** **Drivers**

Customer applications will use following API to find drivers around a given location

**Request:**GET /drivers

Parameters:

"latitude" — mandatory "longitude" — mandatory "radius" — optional defaults to 500 meters "limit" — optional defaults to 10

**Response:**

* -  200 OK — on successful get, Body: [  {id: 42, latitude: 12.97161923, longitude: 77.59463452, distance: 123}, {id: 84, latitude: 12.97161923, longitude: 77.59463452, distance: 123}  ]
* -  422 Unprocessable Entity — with appropriate message. For example:  {"errors": ["Latitude should be between +/- 90"]} Distance in the response is a straight line distance between driver's location and location in the  query
* **Expected** **Load:**20 concurrent requests.