# Node JS/ Ajax

In this lab, we will implement a basic authentication and user profile view, with some API functionalities.

Modules needed: ejs (you can use handlebars if you want), express, fs, openweather-apis

## Step1

Build a node server that responds to requests on port 8081, and render a basic homepage as a respond to the “/” get request.

We will learn how to authenticate users with google and take their google profile credentials to use them on our website. Use this link: https://developers.google.com/identity/sign-in/web/sign-in to help you add a basic sign in with google button. Be careful, you will need to add http://localhost:8081 as a domain that is accepted by your oAuthClient.

After signing in with google and taking the user’s name, email and image, send this information via a get request to the server’s “/user” path, that renders a user interface with the user’s name and image.

Now usually, this is used on websites to help users login/signup quickly. The frontend sends the user’s email and an ID token (as commented on the google link attached above) to the database to be saved, but we will skip this. Keep this in mind, if you are ever required to add google signup/login options.

## Step2

When logged in, the user will have access to weather query functionalities. Using the weather api https://openweathermap.org/api In order to use the Open Weather API you will need to signup and get an API Key, to use in your queries. E.g. api.openweathermap.org/data/2.5/weather?q={city name}&appid={API key} You may use a wrapper module to request API functionalities. Check out the openweather-apis module in npmjs.com. You will need to implement 2 functionalities. Add a select dropdown textbox to toggle between the options.

Option 1: by City Name

If you select city name, render under the select menu a textbox and a button. When the user clicks on the button, send a weather ajax request to the API using the key that you have, and render the result in a table containing cityname, temperature and pressure fields. Render this table in a certain element, to be emptied out if we ask for another city.

Option 2: by Current Location

If you select by current location. Send an ajax request to the api containing your current Lat Long coordinates and get back weather data. Render a similar table (cityname, temperature and pressure) in the element after emptying it out. To get current lat long, use the HTML5 geolocation feature https://www.w3schools.com/html/html5\_geolocation.asp