**Questions for Bro. Lyon**

How many API’s would you prefer for an “above and beyond” grade?

What does “lightly used” mean exactly?

What should we do if we only “lightly” use an API AND it doesn’t make sense to use a database for our application?

**MILESTONES**

* **~~Week 10: Milestone 1~~**
  + ~~UI Built~~
* **Week 11: Milestone 2**
  + Google Maps API integration
* **Week 12: Milestone 3**
  + Google Places API integration
* **Week 13: Project Demo, App Store Submission**
  + App Store Website

**TODO**

* UI/Prettiness Goals
  + Full screen map with information overlay
  + Different theme than default Google theme (Stretch)
  + CSS prettify the marker and path on the map
  + Allow user to select theme (Stretch)
* Geolocation
  + Prompt for permission **\*Brooks**
    - (<https://developers.google.com/web/updates/2015/04/permissions-api-for-the-web>)
    - Mobile
    - Desktop
  + Grab geolocation and forward to page as query
* Static Location Input **\*Jacob** - (Implemented, just need to pass input around to servlet once the other pieces start coming together.)
  + Autocomplete functionality
* Populate Restaurant Names and Details
  + Google Places API
  + Ratings stars - add CSS and possibly font icons? (so pretty)
  + Allow user to rate using stars (Stretch/MySQL or Google?)
* Java Servlets
  + Create servlet pages/classes
  + Call to maps
  + Call to places
  + Log queries to Google Sheets (Stretch/Version 2)
* Mobile Container (Stretch/Version 2)
  + Droid
  + iOS
* Testing
  + Autocomplete works
  + Rating Stars
    - Display
    - Update (User submitted)
  + Geolocation user prompt and redirect to manual query on User “No” answer
  + Get User feedback on UI (Usability Testing)
* Javascript **\* Joseph -** Will need to know servlet names and data requirements
  + Ajax to handle new queries calling servlets.
* Login (Stretch/Version 2) **\* Joseph -** Might do this just because it’s getting covered next week.
  + MySQL DB

**Resources**

**Google Geolocation API Info:** <https://developers.google.com/web/fundamentals/native-hardware/user-location/>

**Google Maps Styling:**

<https://mapstyle.withgoogle.com/> | <https://snazzymaps.com/>

**Google Places Autocomplete:**

<https://developers.google.com/places/web-service/autocomplete>

**Helpful JavaScript:**

Use of Local Storage to store user data

localStorage.setItem(storageKey, JSON.stringify(jsonToObj));

localStorage.getItem(localStorage.key(0));

JavaScript Object conversions to JSON

var obj = JSON.parse(jsonString);

var jsonString = JSON.stringify(obj);

**Assignments/Roles**

* Do stuff

**GIT Procedures**

1. **Fetch** - gets remote
2. **Pull** - gets remote and prompts merge
3. **Fix** - merge changes (Most painful part of development)
4. **Commit** - self explanatory (This saves the changes locally)
5. **Push** - push local to remote

Notes:

Post from Bro. Lyon about MySql and PostgreSQL - Thank you for pointing this out. It appears we missed this page. For the team project, I don't mind if you use MySQL. Make sure to make the decision as a team. MySQL is available on Heroku for free, but it does require a higer level account -- meaning it requires a credit card number. You need to be careful not to install anything that will cost money, but you can use the free tier MySQL.

I will work with the course lead to update this page, but you can search for the PostgreSQL JDBC driver in NetBeans and I will include the Maven dependency below. JDBC is like PDO, so will find that most things that are true for MySQL are also true for PostgreSQL.

|  |
| --- |
| <dependency>     <groupId>org.postgresql</groupId>     <artifactId>postgresql</artifactId>     <version>9.4.1212</version>  </dependency> |