HACKDUKE

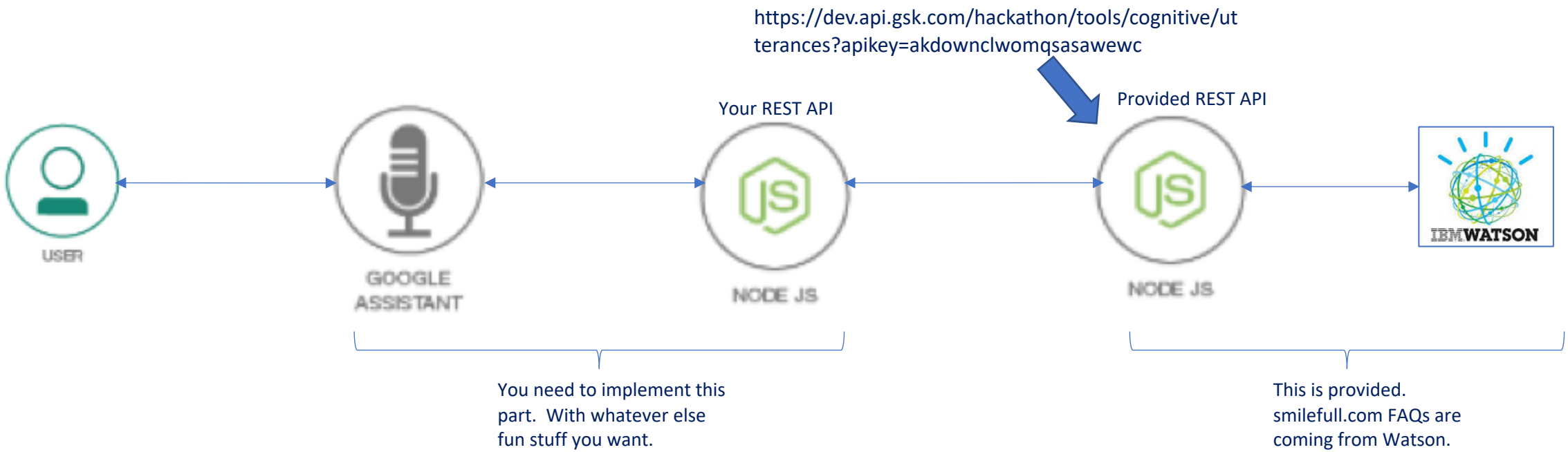
Endpoint: https://dev.api.gsk.com/hackathon/tools/cognitive/utterances

ApiKey: akdownclwomqsasawewc

Business case:

Analysis of usage data from GSK's *smilefull.com* blog has surprisingly revealed significant interest in denture information among millennials. Reliance on dentures to a millennial can impact self-confidence, strain social relationships and ultimately lead to depression. To better serve denture information to millennials we would like to create a Google Home enabled voice experience leveraging GSK's existing natural language conversation technology.

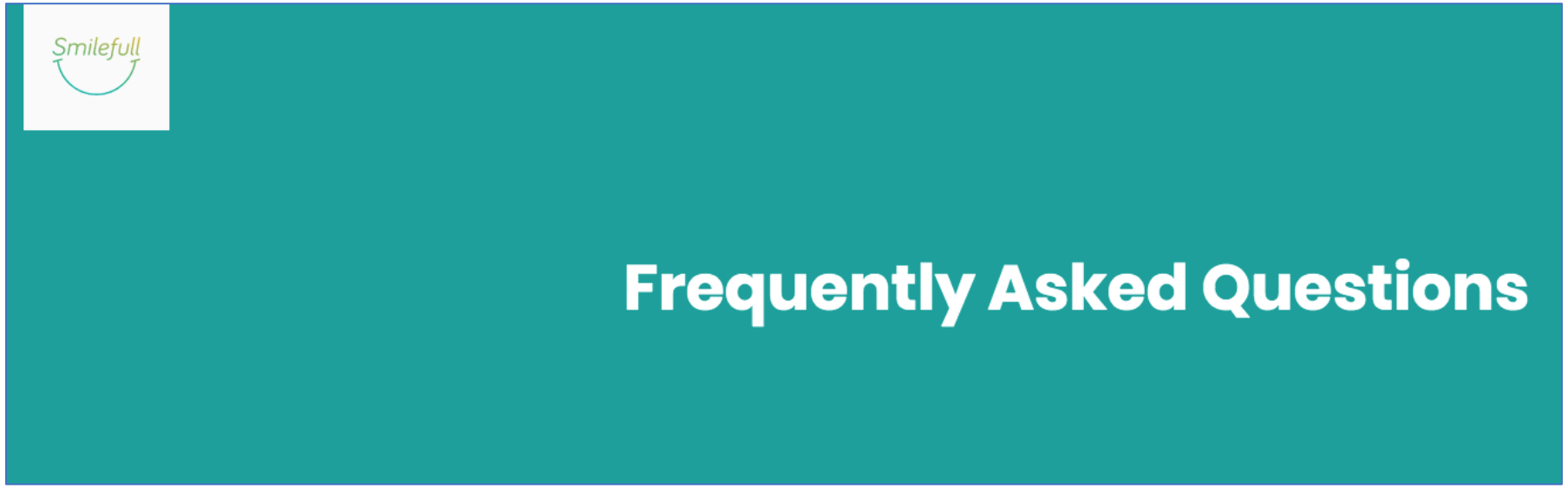
High Level Architecture



- Your REST API does not need to be written in Node.js
- Your REST API needs to be available on the internet (i.e., not running locally on your laptop). One suggestion is to use Heroku. Details later on that.

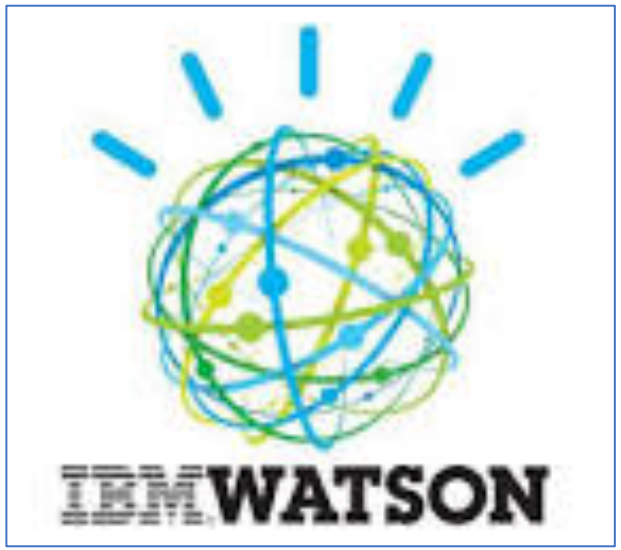
The original data is from the smilefull FAQ page

https://www.smilefull.com/frequently-asked-questions/



Full site = https://www.smilefull.com

That data has been incorporated into IBM Watson Machine Learning Platform



This is what is meant when the Business Case refers to GSK's existing natural language conversation technology.

Sample payload for calling Provided REST API

```
{
   "workspace_id":"24245ae4-a6e3-4236-9557-96ee33616aef",
   "context":{},
   "input":{"text":"Will dentures change the way I look"}
}
```

- You were given the swagger for this API. This is a concrete example of how to build the payload with a real question
- Always use that workspace_id
- context can be empty
- Try this out in Postman (or your favorite REST tool) to get a feel for how it works and the response that you get back

Using Heroku for your Your REST API

- Sign up for a free account – heroku.com
- Download the CLI
- Typical flow (Assuming you are inside a Node/Express project)
  - heroku login
  - heroku create - note the application name
  - git init
  - heroku git:remote -a appName
  - git push heroku master (after git add/commit)
  - heroku logs –tail

You do not have to use Heroku, but it is one way to deploy a REST API.

Deploy Your REST API

- Step 1 could be to deploy Your REST API and have it call the Provided REST API

Node/Express example: It needs to handle a POST from Google Assistant, call the Provided REST API and then return a response. Not showing doing anything with the input or calling the Provided REST API here.

```
app.post('/', (req, res) => {
  res.setHeader('Content-Type', 'application/json');
  res.append('Google-Assistant-API-Version', 'v2');
  res.json(getResponse());
});
```

See index.js for more details.

Google Assistant

- Read the basics - https://developers.google.com/assistant

- Sign up if you do not have a Google/gmail account, otherwise use your current credentials

Google Assistant – “Passthrough Approach”

- Passing through Google Assistant
- Using it only for the speech to text/text to speech conversion
- Not using DialogFlow
- Not creating Intents or Entities in the GUI
- Not creating Actions in the GUI. Creating them from the command line tool
- Not using Firebase

There are many ways to use Google Assistant. This is one. Feel free to use other approaches if you are familiar with them.

Google Assistant – Creating Project

- Go to Actions Console - https://console.actions.google.com
- Click on New Project
  - Name it and click on Create Project
- Click on Actions SDK at bottom (even though we will not use that SDK)
  - Then click OK
- Click on Decide how your Action is invoked
  - Enter name and then click Save
- Click on the vertical 3 dots next to your account avatar and go to Project settings.
  - Save the Project name and Project ID to use later.

Google Assistant – Running gactions commands to send your action(s) to Google Assistant

- Download the gactions CLI - https://developers.google.com/actions/tools/gactions-cli
  - Make the command executable on your laptop
- Edit the action.json file from the git repo
  - Put your project name where it says projectName
  - Put your endpoint URL where it says thisIsTheURLForYourRESTAPI
- Run the following commands
  - gactions update --action_package action.json --project ”Project ID”
  - gactions test --action_package action.json --project “Project ID”

Google Assistant - Testing

- Go back to the Actions Console and into your project if not there already
  - Click on Develop and then Actions. You should see the action that you posted with the gactions command – Default Welcome Intent
- Click on Test
  - Bottom left you should see “Talk to displayNameYouGave”
  - Click enter or use the microphone icon to speak
  - You should see the response coming back from Your REST API

Setting up your device

https://www.tomsguide.com/us/google-home-mini-setup,review-4963.html