Dialogflow Configuration

Get Starter Point From the Github

```
module.exports = {
    googleProjectID: '',
    dialogFlowSessionID: 'bot-session',
    dialogFlowSessionLanguageCode: 'en-US',
    googleClientEmail:'',
    googlePrivateKey: "",
    mongoURI:'',
}
```

Api Console

where you manage certain project tasks, such as generating API credentials, enabling APIs, and managing team and billing information associated with your project.

Google Project

a collection of settings, credentials, and metadata about the application or applications you're working on that make use of Google APIs and Google Cloud Platform resources.

You can create up to 100 service accounts per project (including the default Compute Engine service account and the App Engine service account) using the IAM API, the Cloud Console, or the gcloud command-line tool.

Service Account

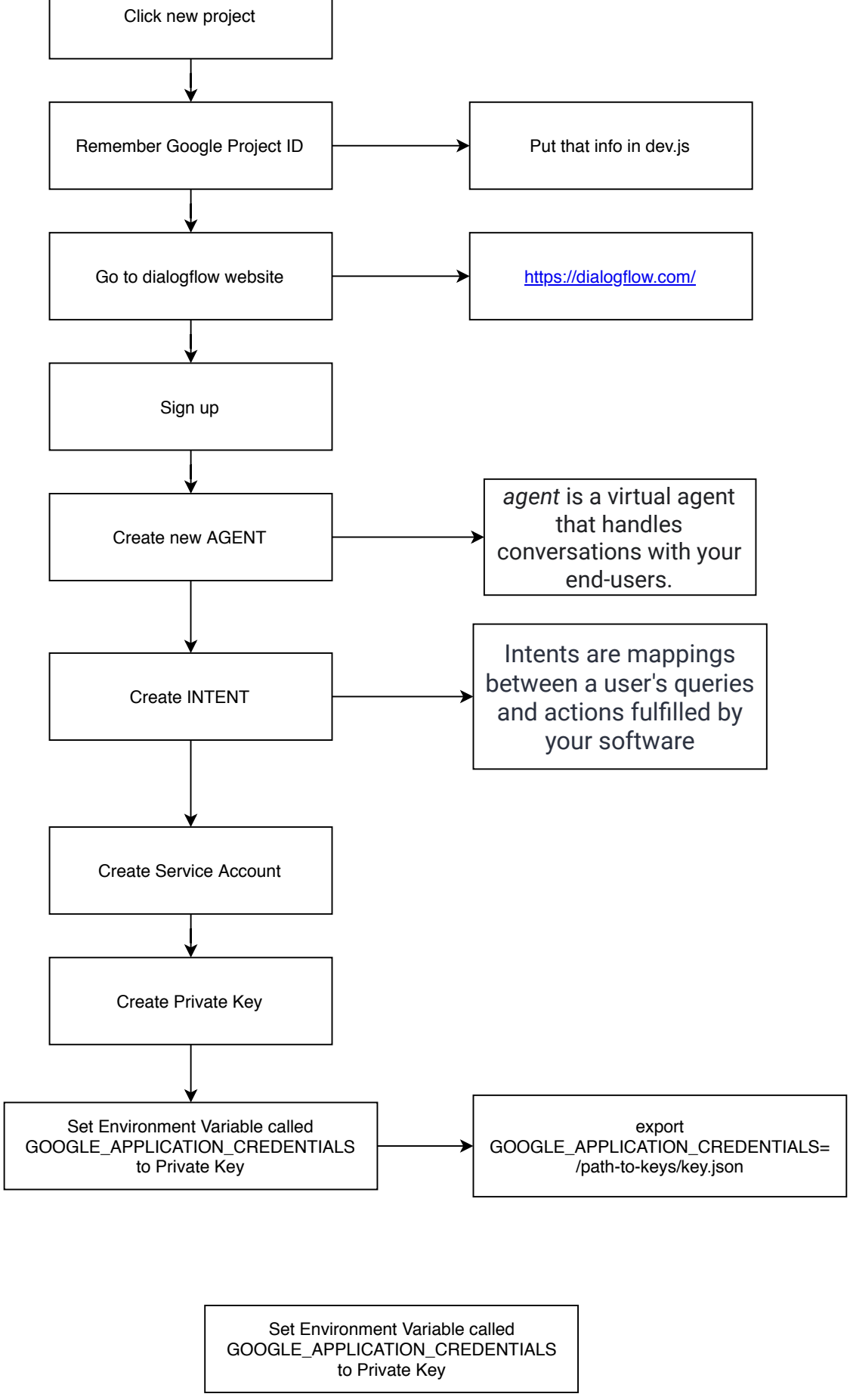
A service account is a special kind of account used by an application or a virtual machine (VM) instance, not a person. Applications use service accounts to make authorized API calls.

Dialogflow API

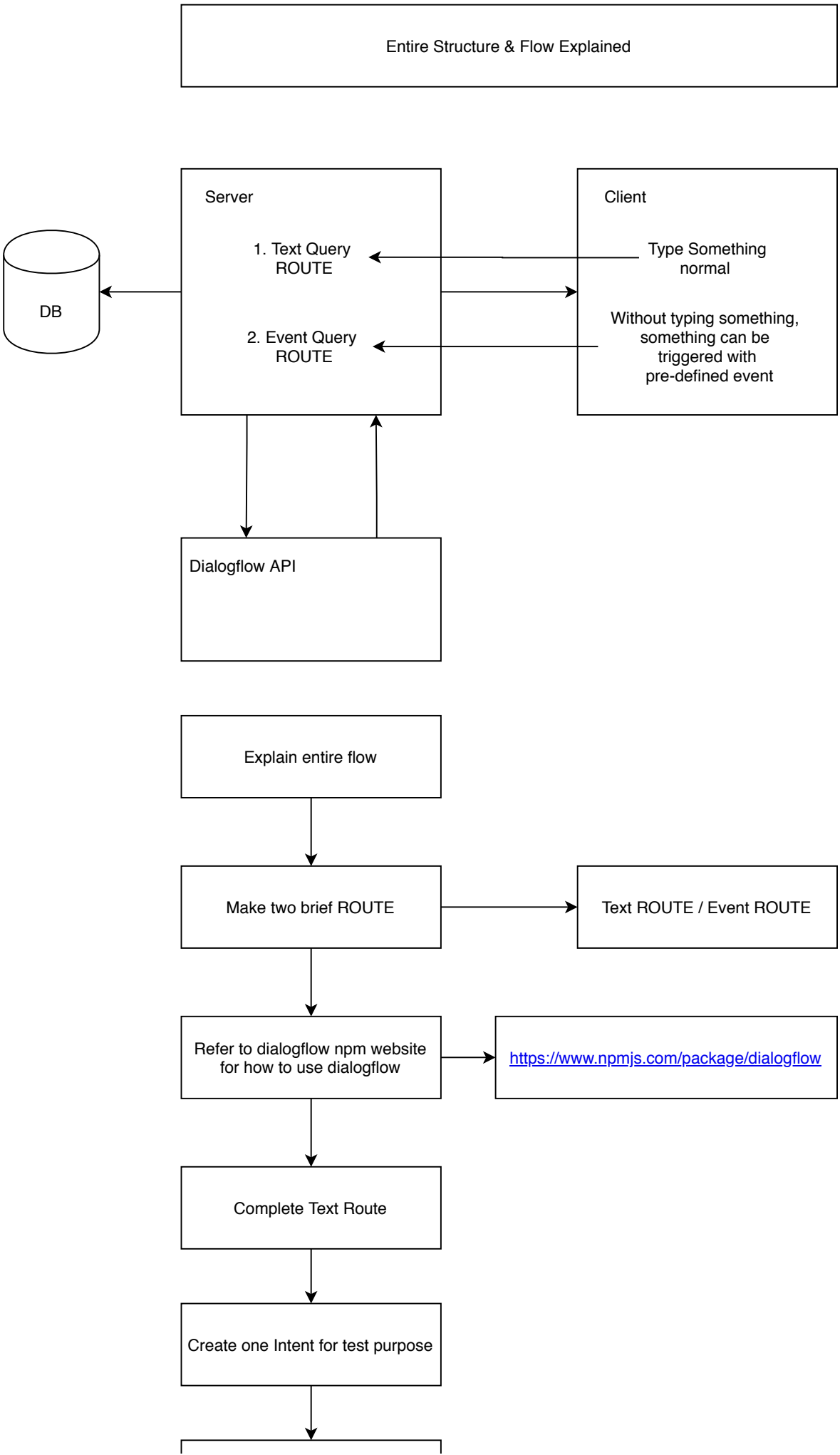
Create google project,

in Google API CONSOLE

https://console.developers.google.com/



Set Environment Variable called GOOGLE_APPLICATION_CREDENTIALS to Private Key

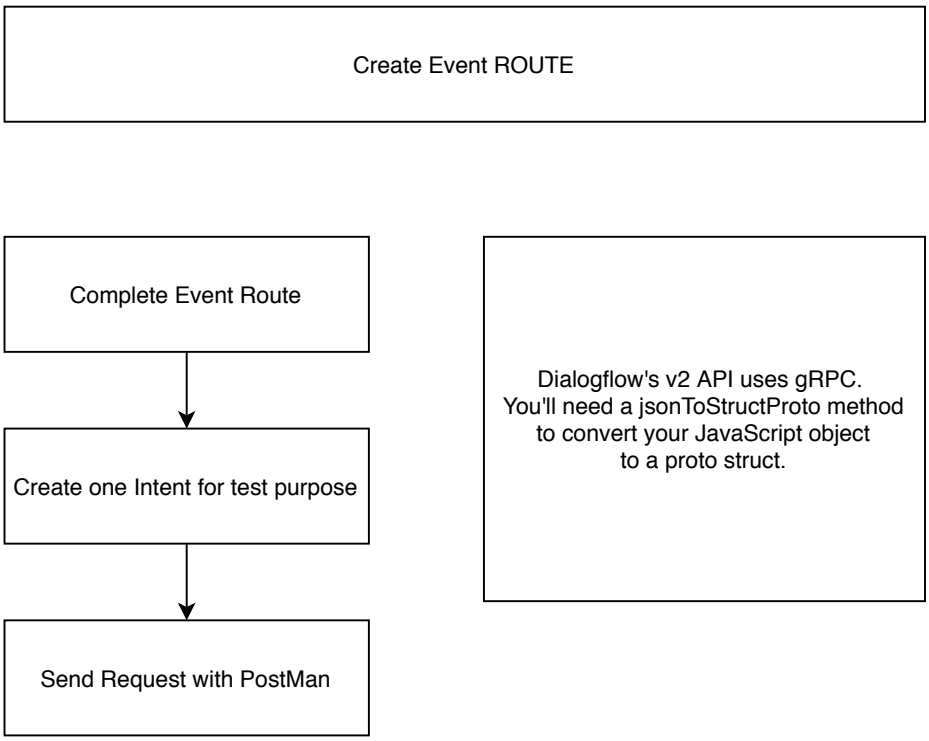

Entire Structure & Flow Explained
Server
Client
DB
1. Text Query
ROUTE
Type Something
normal
2. Event Query
ROUTE
Without typing something,
something can be
triggered with
pre-defined event
Dialogflow API
Explain entire flow
Make two brief ROUTE
Text ROUTE / Event ROUTE
Refer to dialogflow npm website
for how to use dialogflow
https://www.npmjs.com/package/dialogflow
Complete Text Route
Create one Intent for test purpose

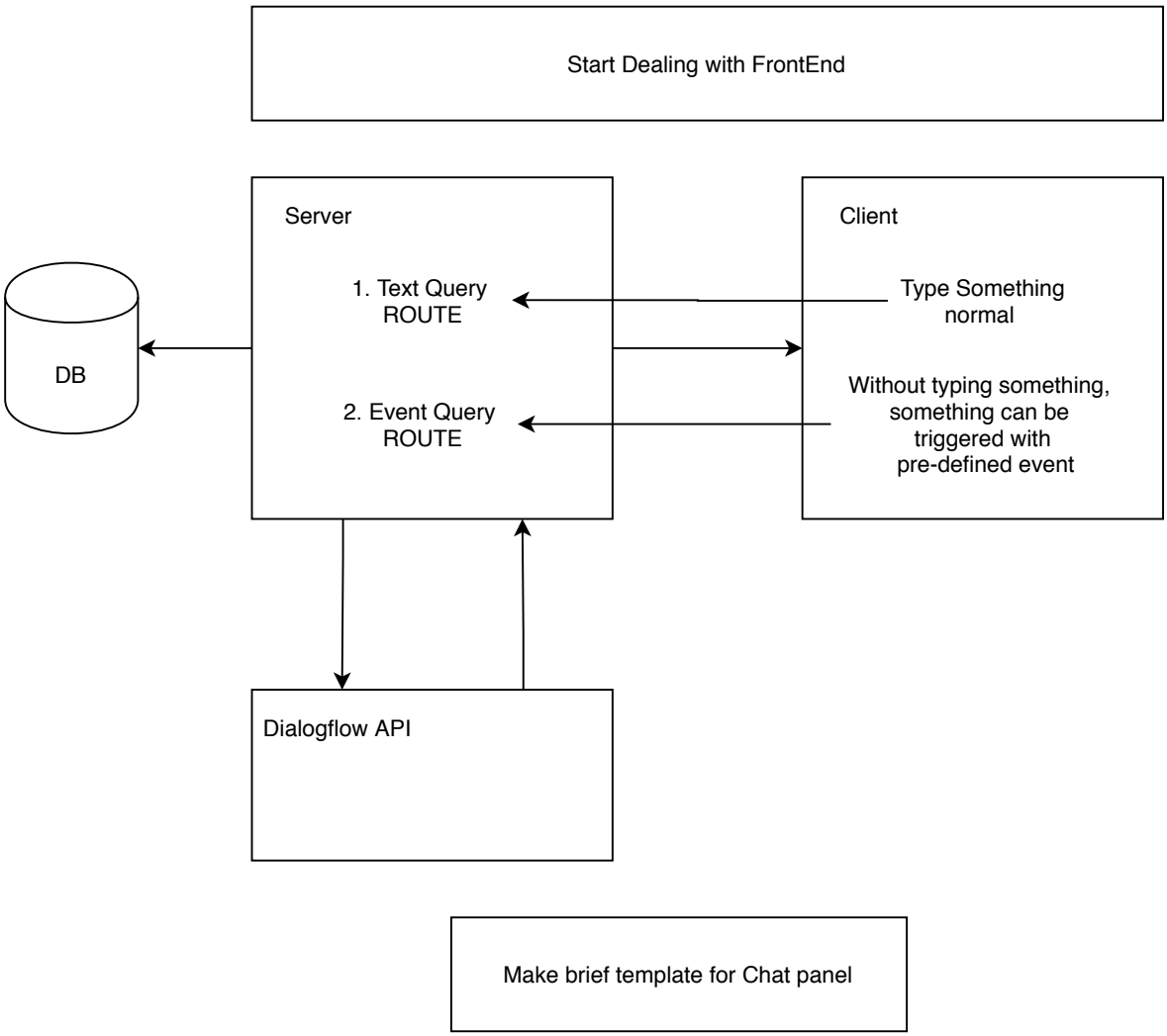
Send Request with PostMan

```
export GOOGLE_APPLICATION_CREDENTIALS=
/path-to-keys/key.json
```

For Window User

```
set GOOGLE_APPLICATION_CREDENTIALS=/path-to-keys/keys-file.json
```


Create Event ROUTE
Complete Event Route
Create one Intent for test purpose
Send Request with PostMan
Dialogflow's v2 API uses gRPC.
You'll need a jsonToStructProto method
to convert your JavaScript object
to a proto struct.

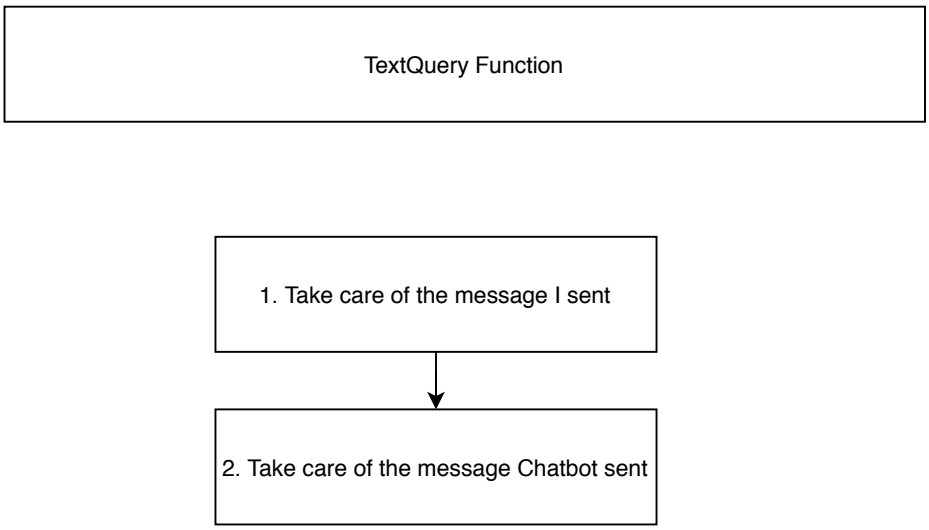


Response From Event Query Route


```
{
    "fulfillmentMessages": [
        {
            "platform": "PLATFORM_UNSPECIFIED",
            "text": {
                "text": [
                    "Greetings! How can I assist?"
                ]
            },
            "message": "text"
        }
    ],
    "outputContexts": [],
    "queryText": "welcomeToMyWebsite",
    "speechRecognitionConfidence": 0,
    "action": "",
    "parameters": {
        "fields": {}
    },
```


Response From Text Query Route


```
{
    "fulfillmentMessages": [
        {
            "platform": "PLATFORM_UNSPECIFIED",
            "text": {
                "text": [
                    "Good day! What can I do for you today?"
                ]
            },
            "message": "text"
        }
    ],
    "outputContexts": [],
    "queryText": "hello!",
    "speechRecognitionConfidence": 0,
    "action": "input.welcome",
    "parameters": {
        "fields": {}
    },
    "allRequiredParamsPresent": true,
    "fulfillmentText": "Good day! What can I do for you today?",
```

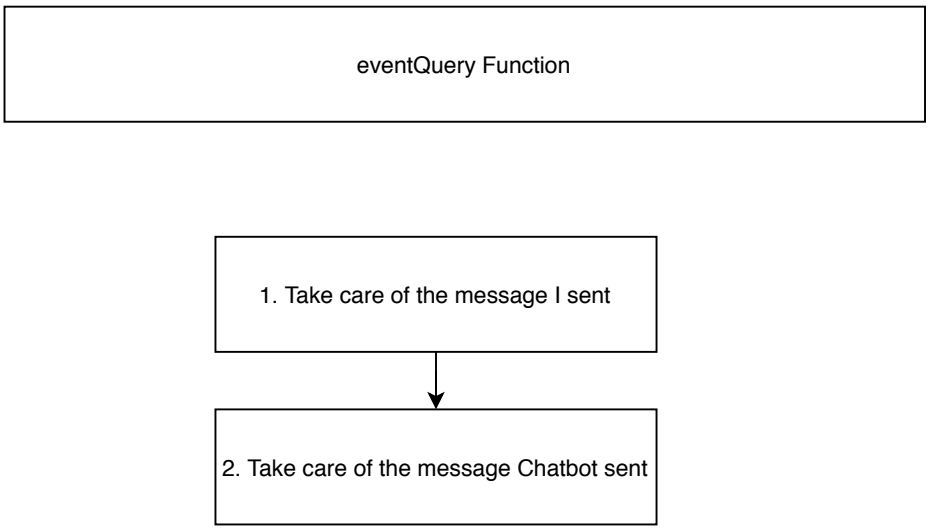


Response From Event Query Route


```
{
    "fulfillmentMessages": [
        {
            "platform": "PLATFORM_UNSPECIFIED",
            "text": {
                "text": [
                    "Greetings! How can I assist?"
                ]
            },
            "message": "text"
        }
    ],
    "outputContexts": [],
    "queryText": "welcomeToMyWebsite",
    "speechRecognitionConfidence": 0,
    "action": "",
    "parameters": {
        "fields": {}
    },
```


Response From Text Query Route


```
{
    "fulfillmentMessages": [
        {
            "platform": "PLATFORM_UNSPECIFIED",
            "text": {
                "text": [
                    "Good day! What can I do for you today?"
                ]
            },
            "message": "text"
        }
    ],
    "outputContexts": [],
    "queryText": "hello!",
    "speechRecognitionConfidence": 0,
    "action": "input.welcome",
    "parameters": {
        "fields": {}
    },
    "allRequiredParamsPresent": true,
    "fulfillmentText": "Good day! What can I do for you today?",
```

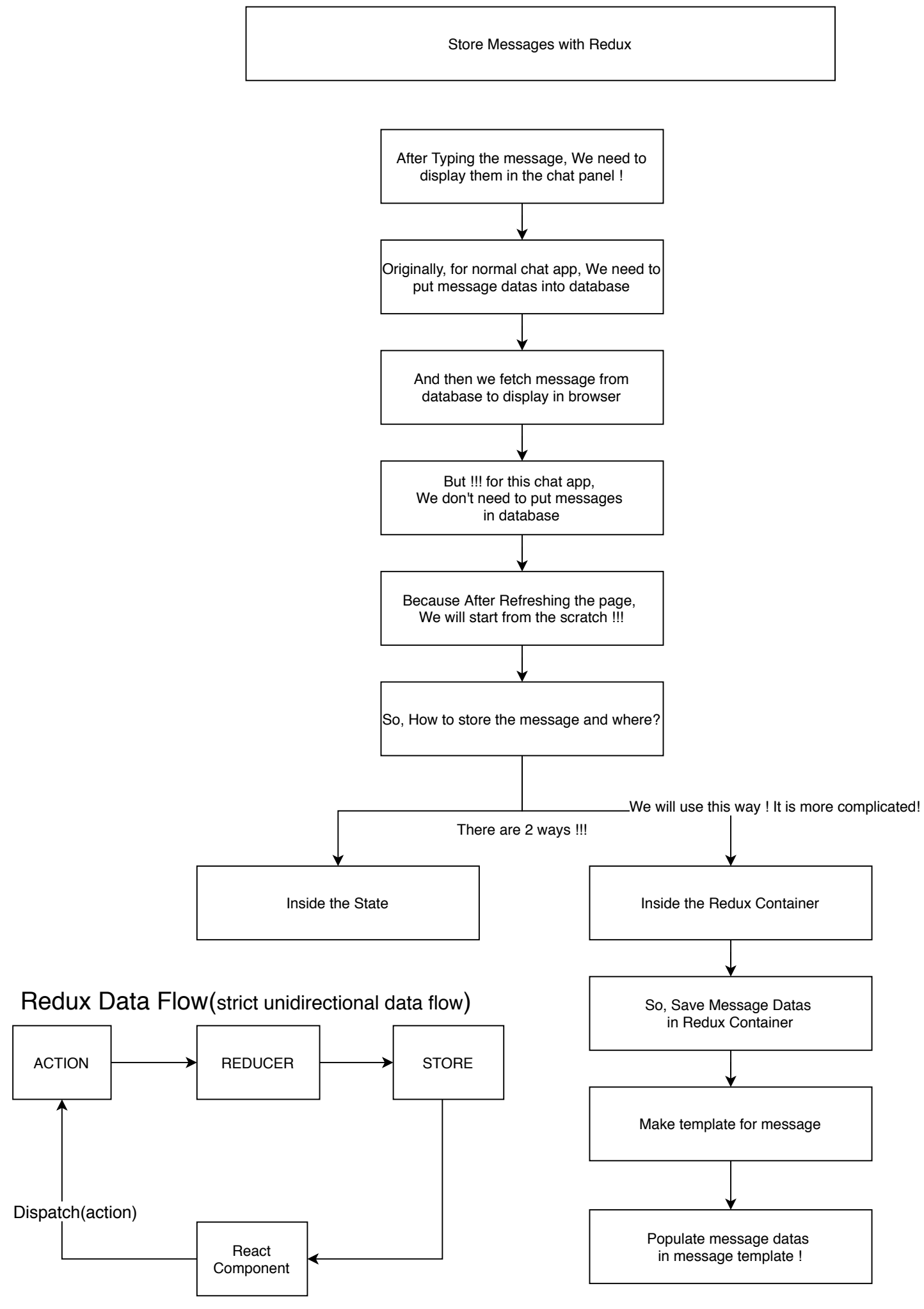


Response From Event Query Route

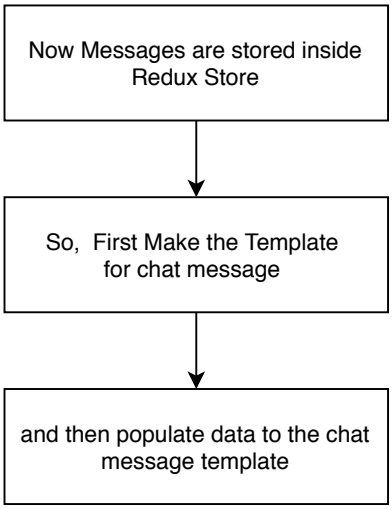

```
{
    "fulfillmentMessages": [
        {
            "platform": "PLATFORM_UNSPECIFIED",
            "text": {
                "text": [
                    "Greetings! How can I assist?"
                ]
            },
            "message": "text"
        }
    ],
    "outputContexts": [],
    "queryText": "welcomeToMyWebsite",
    "speechRecognitionConfidence": 0,
    "action": "",
    "parameters": {
        "fields": {}
    },
```

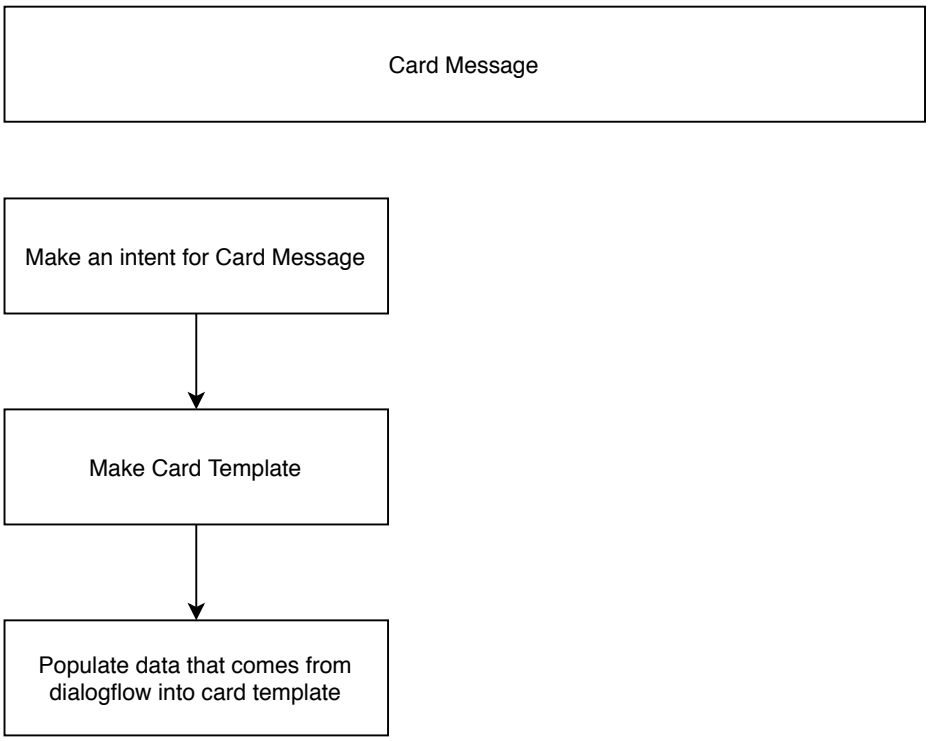

Response From Text Query Route

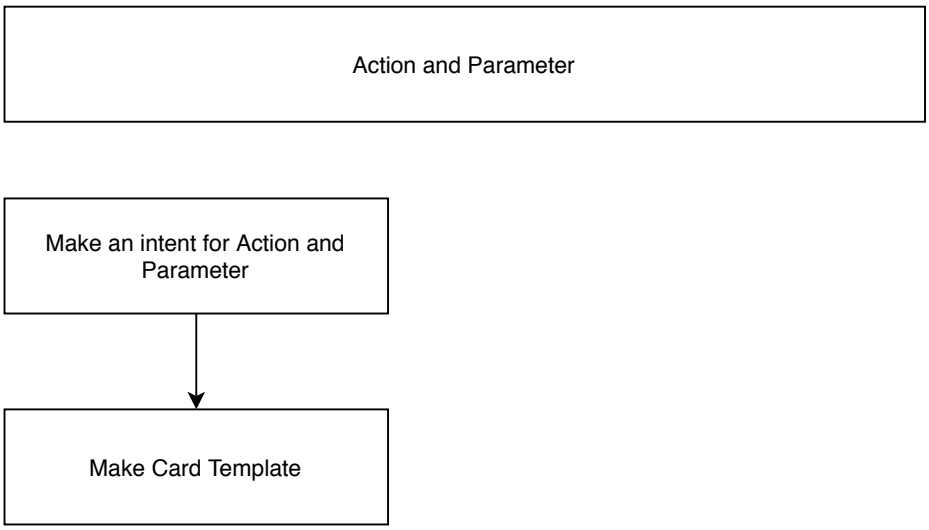

```
{
    "fulfillmentMessages": [
        {
            "platform": "PLATFORM_UNSPECIFIED",
            "text": {
                "text": [
                    "Good day! What can I do for you today?"
                ]
            },
            "message": "text"
        }
    ],
    "outputContexts": [],
    "queryText": "hello!",
    "speechRecognitionConfidence": 0,
    "action": "input.welcome",
    "parameters": {
        "fields": {}
    },
    "allRequiredParamsPresent": true,
    "fulfillmentText": "Good day! What can I do for you today?",
```


Store Messages with Redux
After Typing the message, We need to display them in the chat panel !
Originally, for normal chat app, We need to put message datas into database
And then we fetch message from database to display in browser
But !!! for this chat app, We don't need to put messages in database
Because After Refreshing the page, We will start from the scratch !!!
So, How to store the message and where?
There are 2 ways !!!
We will use this way ! It is more complicated!
Inside the State
Inside the Redux Container
So, Save Message Datas in Redux Container
Make template for message
Populate message datas in message template !
Redux Data Flow(strict unidirectional data flow)
ACTION
REDUCER
STORE
Dispatch(action)
React Component

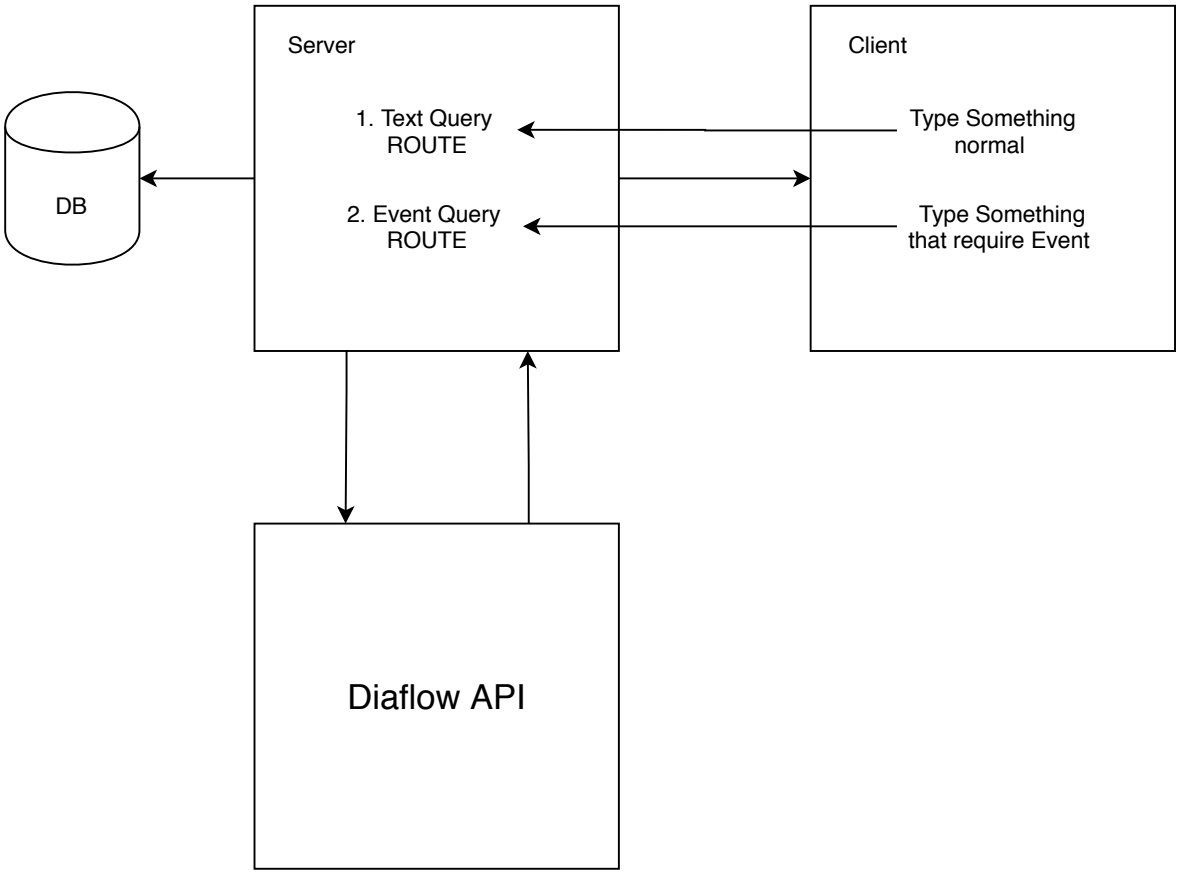
Populate Message Datas to Chat panel



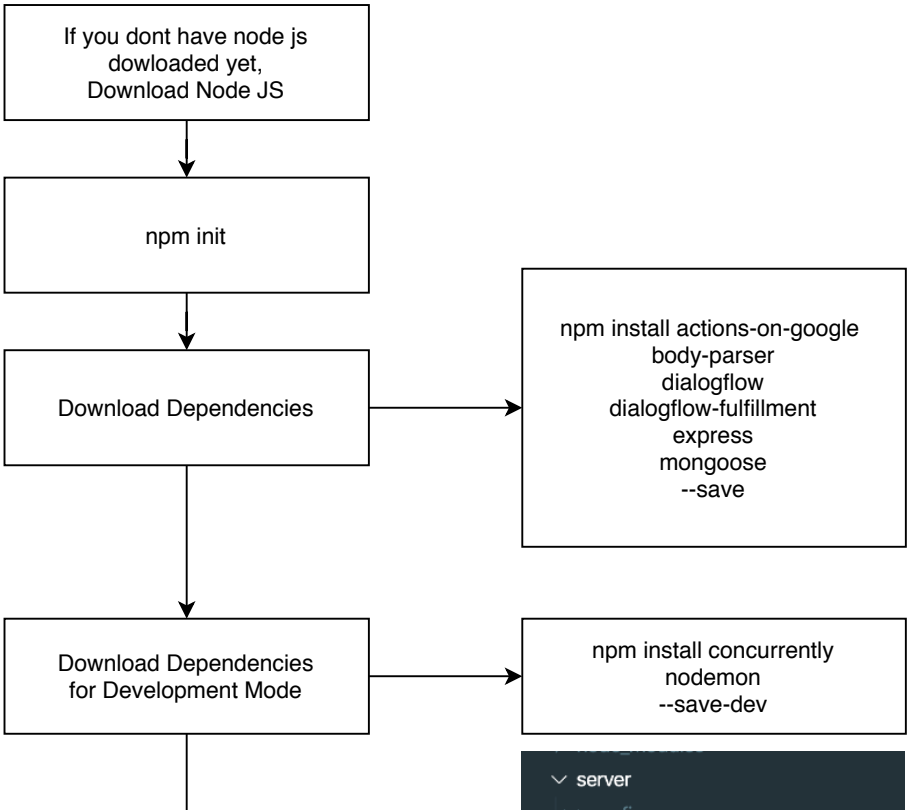

Card Message
Make an intent for Card Message
Make Card Template
Populate data that comes from dialogflow into card template

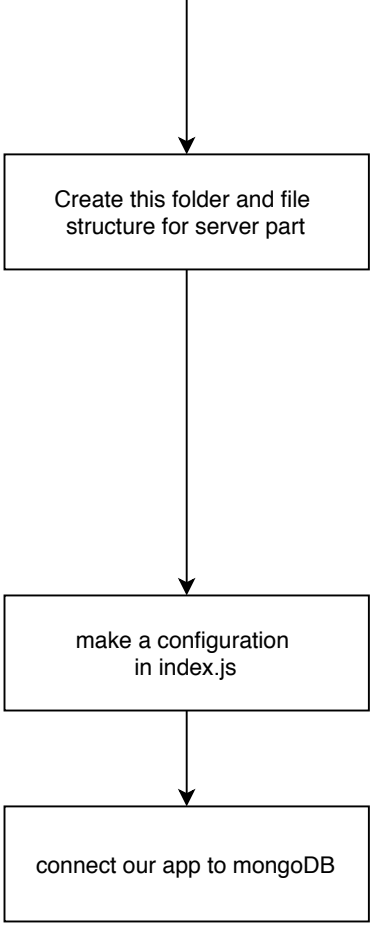

Action and Parameter
Make an intent for Action and Parameter
Make Card Template

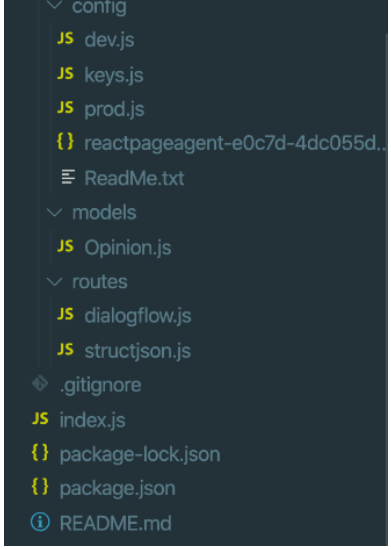
Build up Structure for Backend

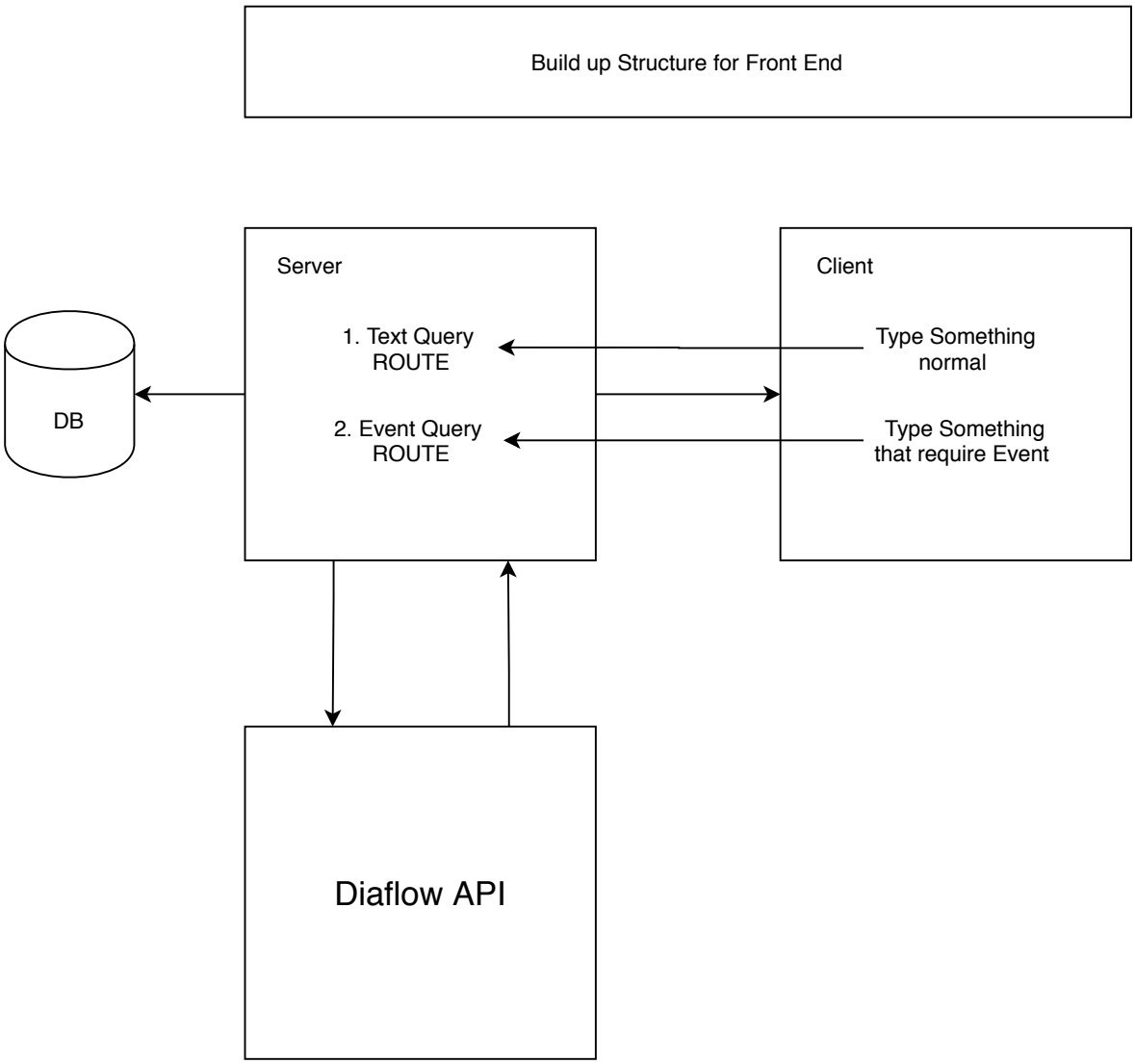


Build Server First !!!

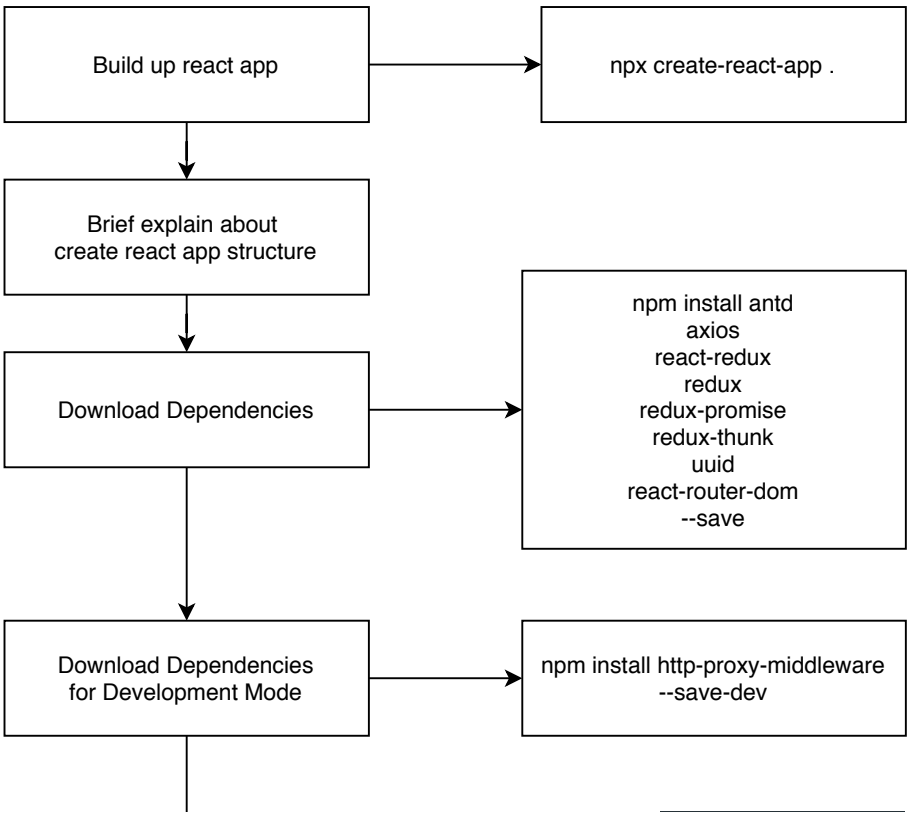


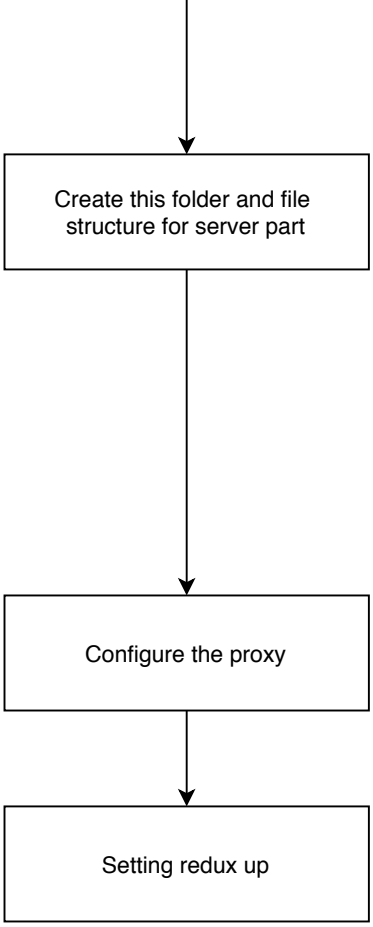

Create this folder and file
structure for server part
make a configuration
in index.js
connect our app to mongoDB

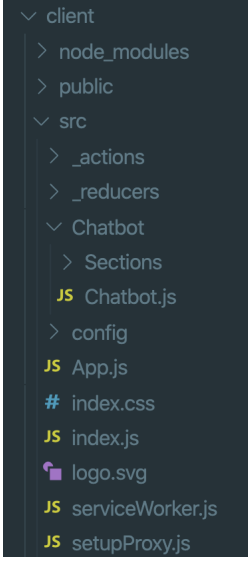

config
dev.js
keys.js
prod.js
reactpageagent-e0c7d-4dc055d..
ReadMe.txt
models
Opinion.js
routes
dialogflow.js
structjson.js
.gitignore
index.js
package-lock.json
package.json
README.md



Build Front End




Create this folder and file structure for server part
Configure the proxy
Setting redux up


client
node_modules
public
src
_actions
_reducers
Chatbot
Sections
Chatbot.js
config
App.js
index.css
index.js
logo.svg
serviceWorker.js
setupProxy.js