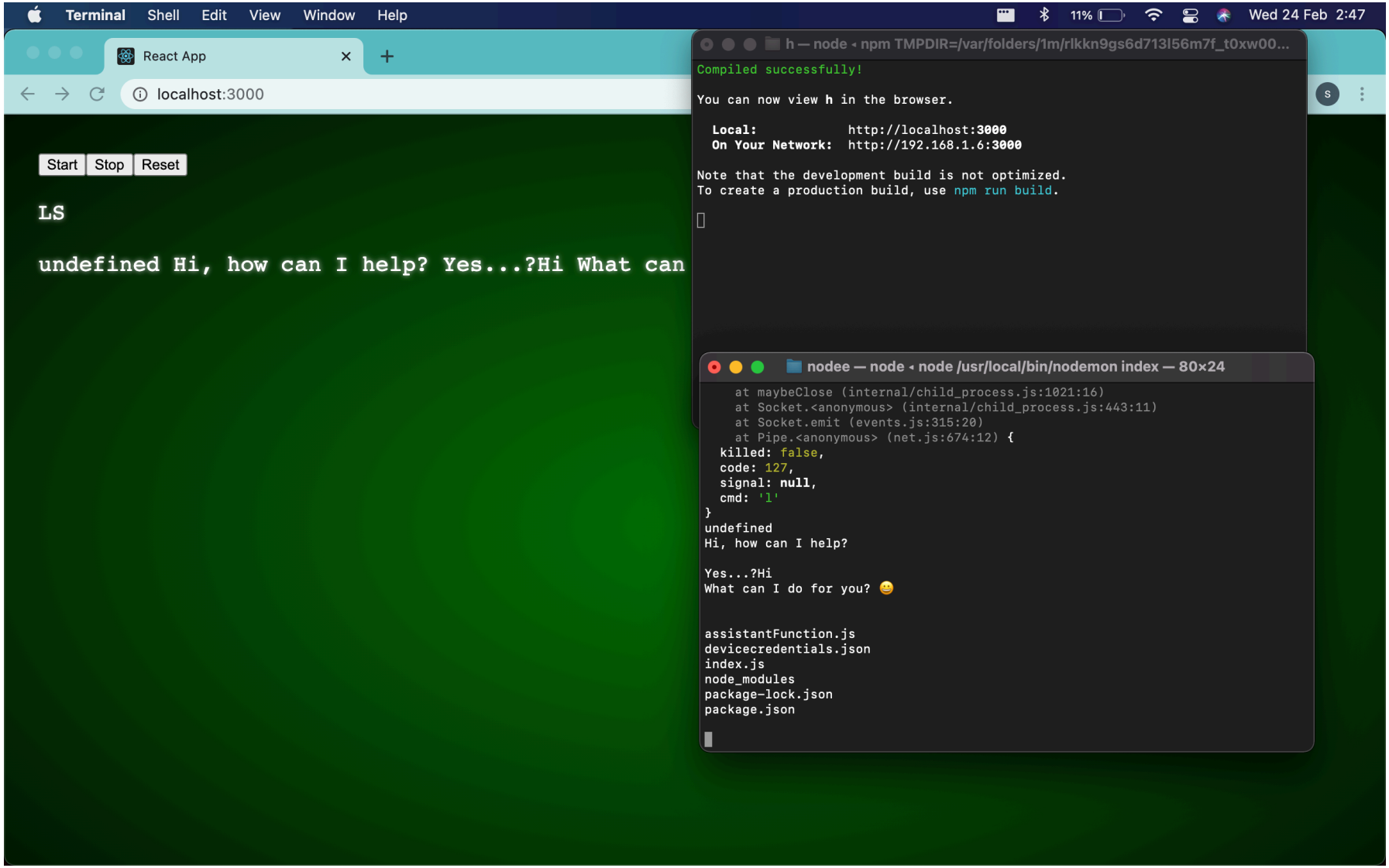
SAYASH RAAJ HIRAOU

CE19B113@SMAIL.IITM.AC.IN


Terminal Shell Edit View Window Help
Wed 24 Feb 2:47
React App
localhost:3000
Start Stop Reset
LS
undefined Hi, how can I help? Yes...?Hi What can
h — node ◂ npm TMPDIR=/var/folders/1m/rlkkn9gs6d713l56m7f_t0xw00...
Compiled successfully!
You can now view h in the browser.
Local: http://localhost:3000
On Your Network: http://192.168.1.6:3000
Note that the development build is not optimized.
To create a production build, use npm run build.
nodee — node ◂ node /usr/local/bin/nodemon index — 80×24
at maybeClose (internal/child_process.js:1021:16)
at Socket.<anonymous> (internal/child_process.js:443:11)
at Socket.emit (events.js:315:20)
at Pipe.<anonymous> (net.js:674:12) {
killed: false,
code: 127,
signal: null,
cmd: 'l'
}
undefined
Hi, how can I help?
Yes...?Hi
What can I do for you?
assistantFunction.js
devicecredentials.json
index.js
node_modules
package-lock.json
package.json

Overview:

- I am the sole member of this team for the MTX hackathon and built the project till 4 AM, sacrificing sleep for a good cause
- The problem given by the hackathon is to build a software/interface that can execute some shell commands.
- My solution does BOTH and excels in it!

The solution I have designed uses Node.js in the backend and to execute the shell commands, React.js for frontend and taking voice input. The project uses Google Assistant SDK, making this the most advanced, robust, and unique solution spanning a wide range of technologies

The project I have designed can execute ANY and ALL shell commands used. The project has been tested in a BASH workspace in Mac Big Sur

Reason for choosing the problem:

I have always wanted to venture into the automation side of software. It has always appealed to me, but I have not yet gone out of my comfort zone because as appealing as the idea and technology is, it is ten times more intimidating as well!

I want to thank you for providing me with this opportunity to venture out in the unknown, out of my comfort zone and learn something new and exciting

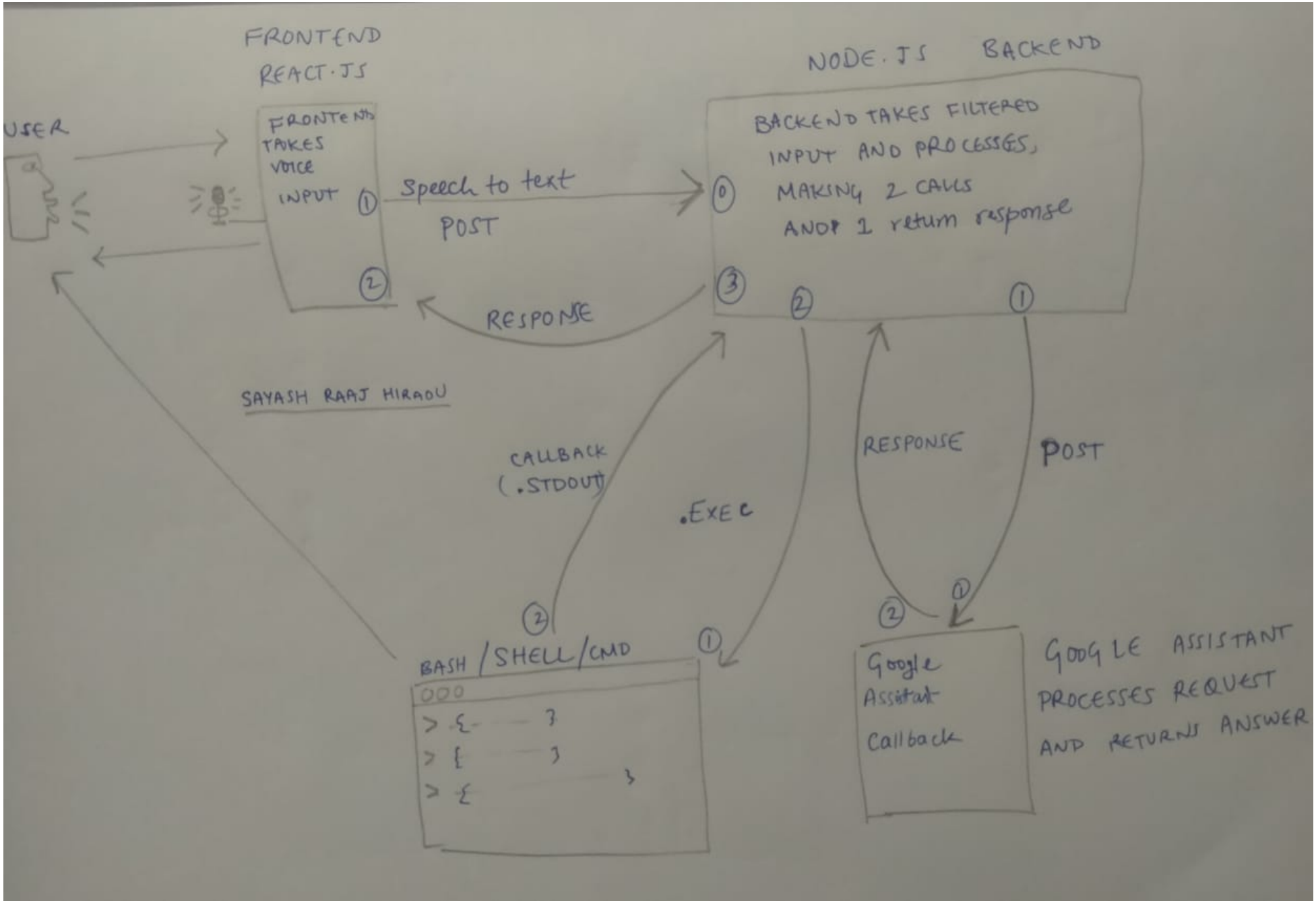
Summary:

All the requirements for the project have been met, all guidelines have been followed, and the best solution has been put forward by this team of 1.
The technological stack used to design the solution uses Node.js for the backend and shell commands and uses React.js for the user interface and input

Solution

The following flowchart explains it all:

Next page


FRONTEND
REACT.JS
NODE.JS BACKEND
USER
FRONTEND TAKES VOICE INPUT
Speech to text
POST
BACKEND TAKES FILTERED INPUT AND PROCESSES, MAKING 2 CALLS AND 1 return response
RESPONSE
SAYASH RAAJ HIRAOU
CALLBACK (.STDOUT)
.EXEC
RESPONSE
POST
BASH / SHELL / CMD
Google Assistant Callback
GOOGLE ASSISTANT PROCESSES REQUEST AND RETURNS ANSWER