Micro Front-end Architecture

List of frameworks used

- **Nginx** - for reverse proxy
- **NodeJS** - for path configuration
- **Docker** - for Nginx container

Requirements list

- Should able to direct various subpaths of a domain to specific different microservices on the go
- Should able to use iframe inside to show component of different microservice inside each other

Nginx findings

We can use "**nginx -s realod**" for soft reload of Nginx on config change without affecting server

```
# for adding a new microservice to a subpath
location /website {
    proxy_pass https://stag.bombayshirts.com;
}
-------------------------------------------------------------------------
--
# when reverse proxy transfer request to microservice
# a new subpath "/website" will effect the microservice resposce
# we can solve using two option

1. Add "baseurl" in the microservice itself to handle it
2. we can add rewrites in our niginx config like below

      rewrite /website$ / break;
      rewrite /website(.*) $1 break;
 -------------------------------------------------------------------------
---
# to make sure that ssl handshake works properly we can add

    proxy_ssl_server_name on;
-------------------------------------------------------------------------
--
# for redirectly a subpath to an another subpath inside nginx

      return 307 $scheme://$host/{new sub path}$request_uri;
-------------------------------------------------------------------------
--
# microservice may use "a" href like "/store" for redirect
# this can cause the url to lose the subpath "/website" in the browser
# to prevent that we can write our own redirect rules in "/" home route

if ($http_referer ~* /(\w+)/(\w+)(.*)$ ) {
```

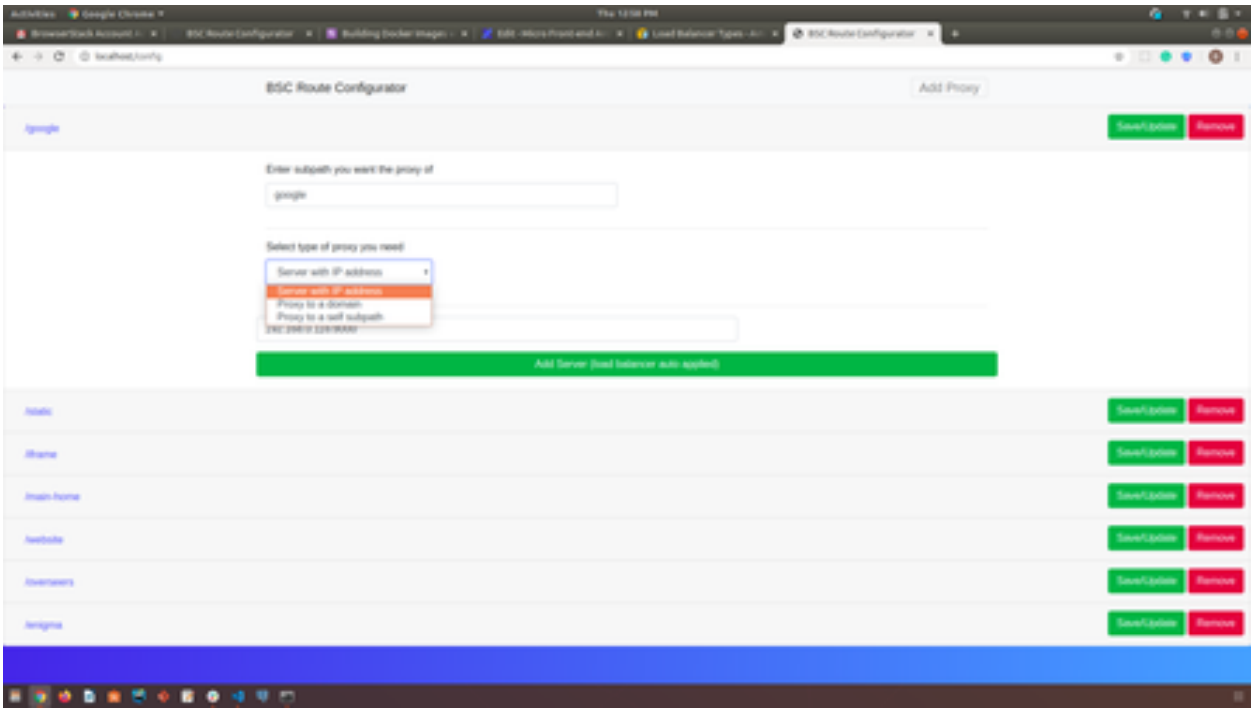
```
    # save sub path if any
    set $subpath $2;
}

# if referer subpath is one of the config subRoutes
if ($subpath ~ (website|shot|enigma|screenShot) ){
    return 307 $scheme://$host/$subpath$request_uri;
}
-------------------------------------------------------------------------
--
# for reverse proxy to a microservice with ip address we can use upstream

upstream microserver {
        server 192.168.0.116:3000;
        server 192.168.0.117:3000;
}

# this will create a reverse proxy to microserver
# as well as it will create a Network Load Balancer for all server listed
inside
-------------------------------------------------------------------------
--
# 301 redirect convert all request to GET type
# 307 redirect will preserve the request type as well like "POST"
-------------------------------------------------------------------------
--
```

Nginx Auto Configuration



- Above is the UI for auto-configuration of Nginx
- We can add new subpath using “Add Proxy” button

- We can direct a subpath to a server, domain or to an another subpath
- When save is clicked the config data is saved in Firebase DB
- While saving a new config file is generated and saved in a shared volume
- Nginx auto-reload itself when the config file in shared volume is changed

Running the application

```
> sudo docker-compose up --build
# to start the nginx docker on localhost:80
> npm start
# to start the node app  for configuration
```

NodeJS working

On the first start, “server/updateConfig.js” is called this will bring data from firebase DB and will generate a new config file based on DB data.

DB schema

{

subpath: "New Subpath",

proxyType: "ip|domain|subpath",

schema: "https",

host: [],

domain: "",

proxySubpath: ""

}

Final config file is saved in “nginx-config/sites-default.conf”

Nginx dockerfile is present in “nginx-docer/Dockerfile”

All api routes are in “routes/index.js” file