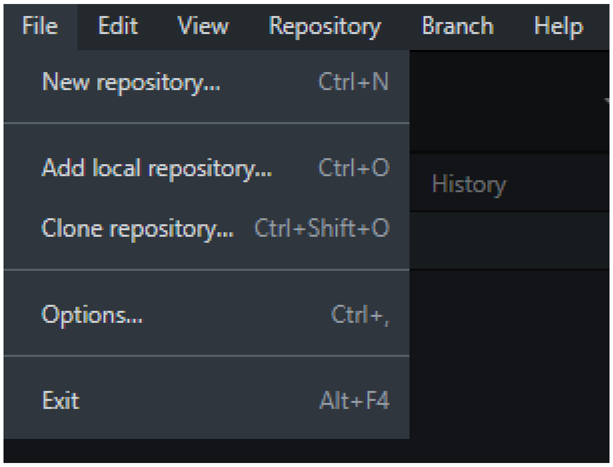
StreetCard Frontend Operations Manual

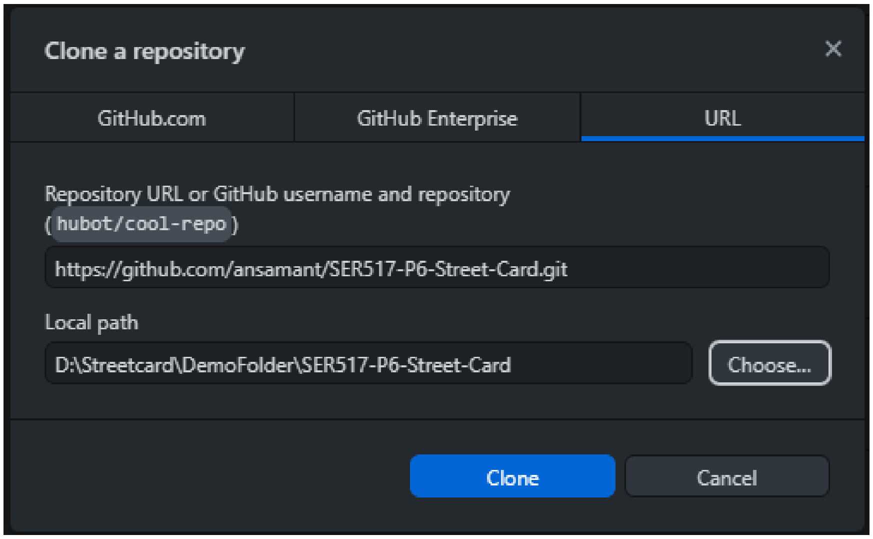
In 22 steps

1. Download Github Desktop from here: https://docs.github.com/en/desktop/installing-and-configuring-github-desktop/installing-and-authenticating-to-github-desktop/installing-github-desktop
2. Clone the Github Repository by going to file > clone repository.



Then select URL and insert this link:

   a. https://github.com/ansamant/SER517-P6-Street-Card.git



Repository URL or GitHub username and repository (`hubot/cool-repo`)

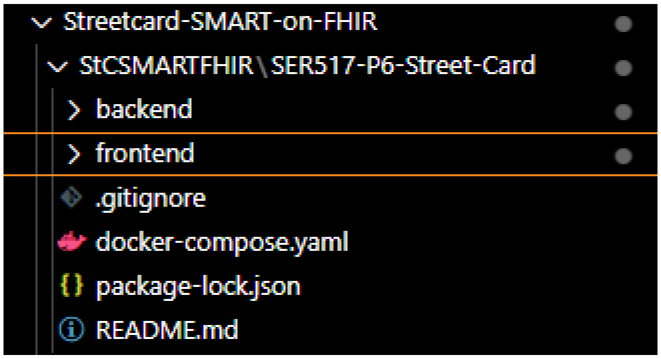
https://github.com/ansamant/SER517-P6-Street-Card.git

Local path

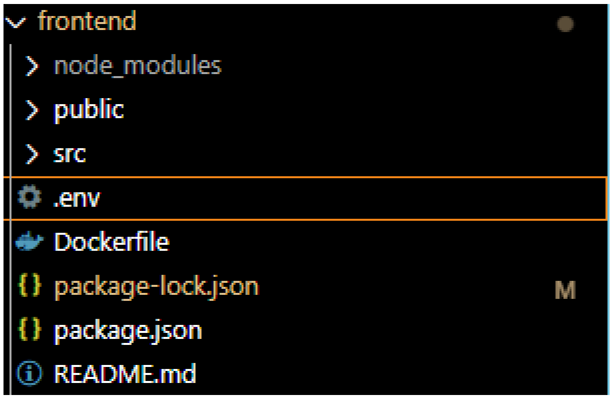
D:\Streetcard\DemoFolder\SER517-P6-Street-Card

3. Download Docker at www.docs.docker.com/get-docker/
4. Open up the Github Repository on an IDE of your choosing. From here, the instructions will be related to Visual Studio Code.

5. In the folder with all of the code from the Github, open the folder labeled “frontend”

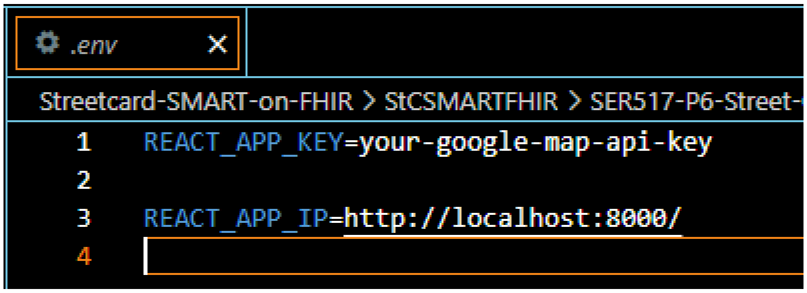


6. In the frontend folder, create a file named “.env”.



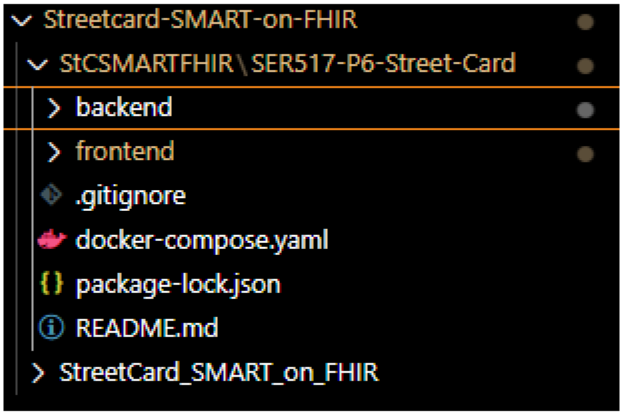
7. In the file, add the two following lines

   REACT_APP_KEY=your-google-map-api-key
   REACT_APP_IP=http://localhost:8000/

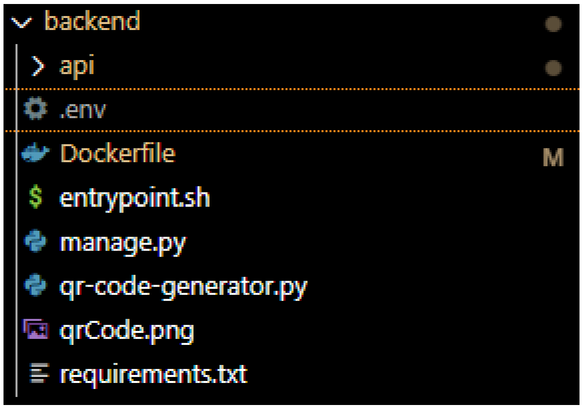


```
REACT_APP_KEY=your-google-map-api-key

REACT_APP_IP=http://localhost:8000/

```

8. Save and close the file. The text labeled “your-google-map-api-key” would be replaced with an API key linking to google maps.
9. Open a folder labeled “backend”. This should be above the frontend folder.



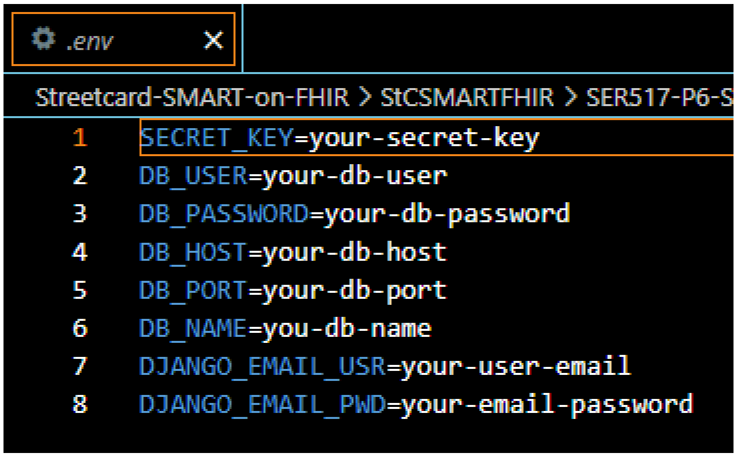
10. In the backend folder, create a file named “.env”.



```
backend
  api
  .env
  Dockerfile          M
  entrypoint.sh
  manage.py
  qr-code-generator.py
  qrCode.png
  requirements.txt
```

11. In the file, add the following lines:

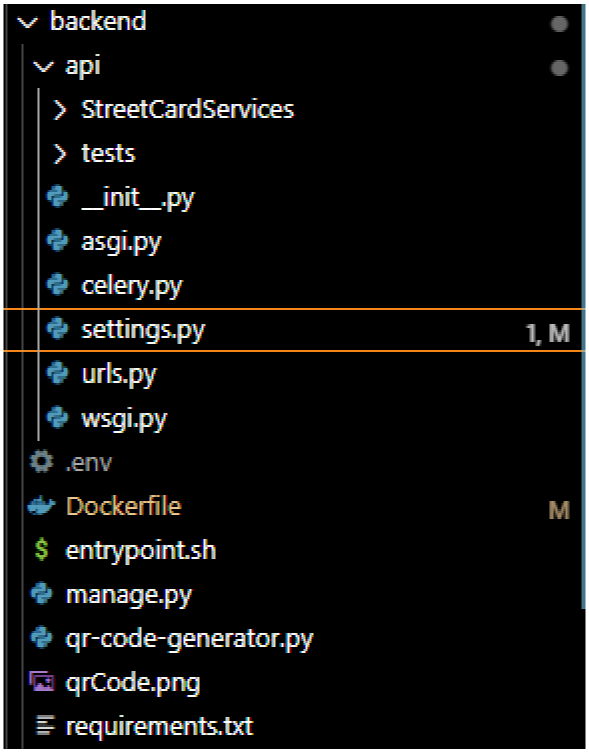
    SECRET_KEY=your-secret-key
    DB_USER=your-db-user
    DB_PASSWORD=your-db-password
    DB_HOST=your-db-host
    DB_PORT=your-db-port
    DB_NAME=you-db-name
    DJANGO_EMAIL_USR=your-user-email
    DJANGO_EMAIL_PWD=your-email-password



```
SECRET_KEY=your-secret-key
DB_USER=your-db-user
DB_PASSWORD=your-db-password
DB_HOST=your-db-host
DB_PORT=your-db-port
DB_NAME=you-db-name
DJANGO_EMAIL_USR=your-user-email
DJANGO_EMAIL_PWD=your-email-password
```

12. Save and close the file. These variables are used to hook up the database to the code.

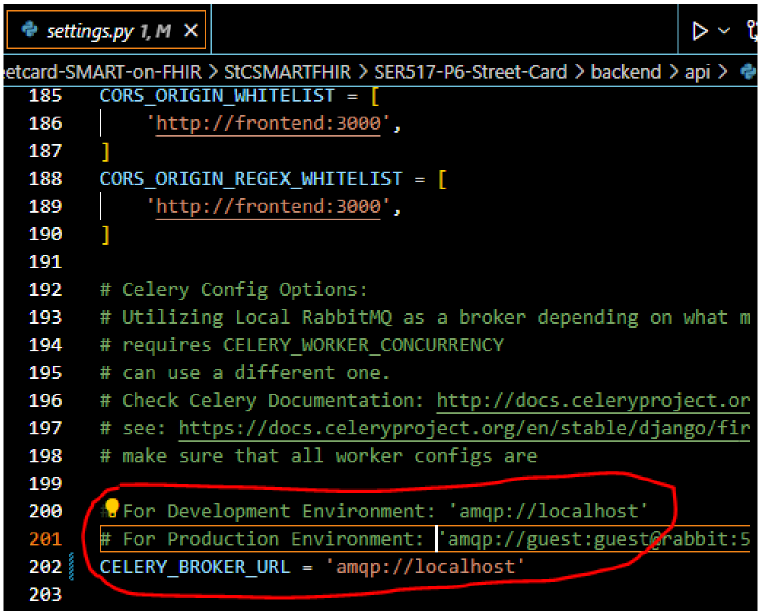
13. In the backend folder, expand the api folder and open the file named “settings.py”



```
backend
  api
    StreetCardServices
    tests
    __init__.py
    asgi.py
    celery.py
    settings.py                1, M
    urls.py
    wsgi.py
  .env
  Dockerfile                   M
  entrypoint.sh
  manage.py
  qr-code-generator.py
  qrCode.png
  requirements.txt
```

14. At around line 202, where it states “CELERY_BROKER_URL” replace the link with the following, exactly as shown below:

    'amqp://localhost'

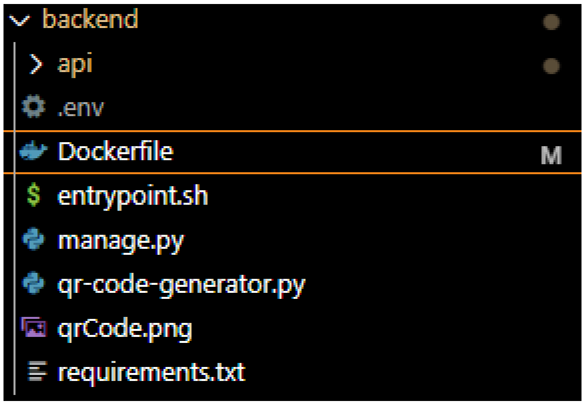


settings.py 1, M
etcard-SMART-on-FHIR > StCSMARTFHIR > SER517-P6-Street-Card > backend > api >

```
185  CORS_ORIGIN_WHITELIST = [
186      'http://frontend:3000',
187  ]
188  CORS_ORIGIN_REGEX_WHITELIST = [
189      'http://frontend:3000',
190  ]
191
192  # Celery Config Options:
193  # Utilizing Local RabbitMQ as a broker depending on what m
194  # requires CELERY_WORKER_CONCURRENCY
195  # can use a different one.
196  # Check Celery Documentation: http://docs.celeryproject.or
197  # see: https://docs.celeryproject.org/en/stable/django/fir
198  # make sure that all worker configs are
199
200  # For Development Environment: 'amqp://localhost'
201  # For Production Environment: 'amqp://guest:guest@rabbit:5
202  CELERY_BROKER_URL = 'amqp://localhost'
203
```

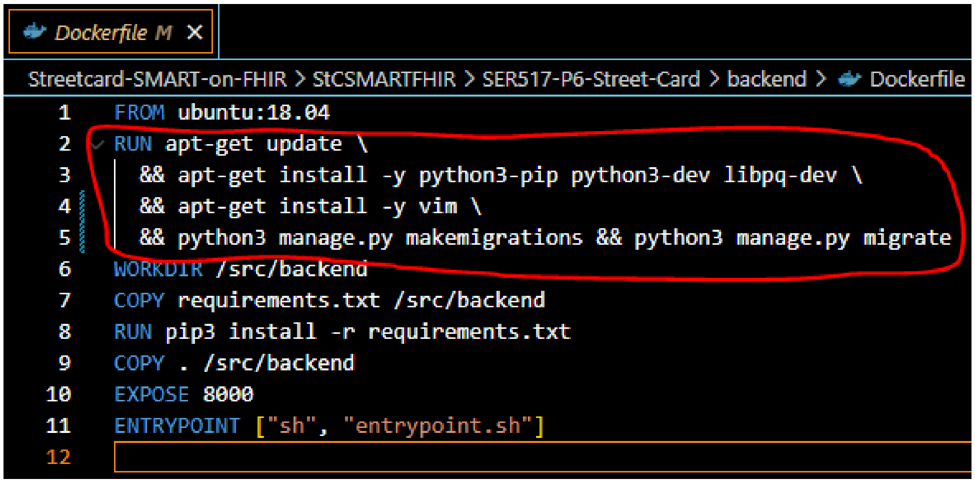
15. Save and close the file. This line is used to link Celery, which is a task queue system, to the StreetCard code. It usitlizes Local RabbitMQ. By setting it to localhost, you should be able to run it on your system. The code that was there previously was used for the Production environment.

16. In the backend folder, open a file named Dockerfile.



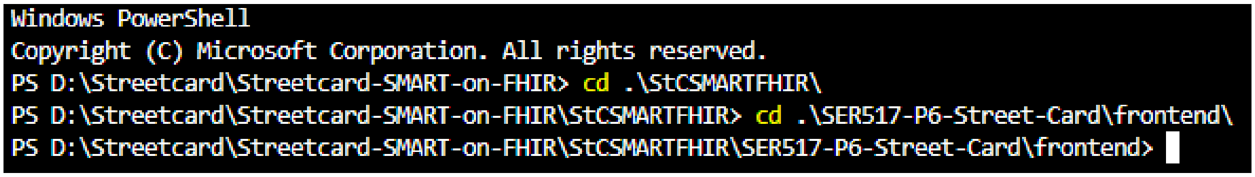
17. Replace the RUN command with the following:

```
RUN apt-get update \
  && apt-get install -y python3-pip python3-dev libpq-dev \
  && apt-get install -y vim \
  && python3 manage.py makemigrations && python3 manage.py migrate
```


Dockerfile M ×
Streetcard-SMART-on-FHIR > StCSMARTFHIR > SER517-P6-Street-Card > backend > Dockerfile

```
FROM ubuntu:18.04
RUN apt-get update \
  && apt-get install -y python3-pip python3-dev libpq-dev \
  && apt-get install -y vim \
  && python3 manage.py makemigrations && python3 manage.py migrate
WORKDIR /src/backend
COPY requirements.txt /src/backend
RUN pip3 install -r requirements.txt
COPY . /src/backend
EXPOSE 8000
ENTRYPOINT ["sh", "entrypoint.sh"]

```

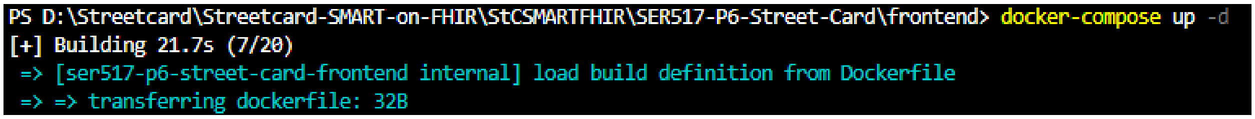
18. Save the file and close. This was to fix some issues within the docker file. Docker was used to create an environment for the code to run in.
19. Using terminal, navigate to the front end folder (cd frontend)



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
PS D:\Streetcard\Streetcard-SMART-on-FHIR> cd .\StCSMARTFHIR\
PS D:\Streetcard\Streetcard-SMART-on-FHIR\StCSMARTFHIR> cd .\SER517-P6-Street-Card\frontend\
PS D:\Streetcard\Streetcard-SMART-on-FHIR\StCSMARTFHIR\SER517-P6-Street-Card\frontend>
```

20. Type in the following:

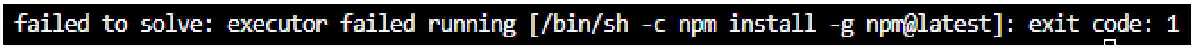
```
docker-compose up -d
```



```
PS D:\Streetcard\Streetcard-SMART-on-FHIR\StCSMARTFHIR\SER517-P6-Street-Card\frontend> docker-compose up -d
[+] Building 21.7s (7/20)
 => [ser517-p6-street-card-frontend internal] load build definition from Dockerfile
 => => transferring dockerfile: 32B
```

21. Press enter. If you get an error, look for an exit code. Below are some examples of error codes that we faced when troubleshooting the code.

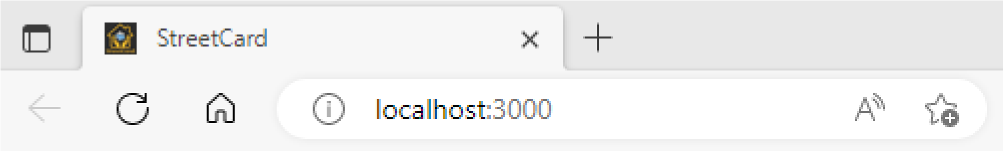
a. Exit Code 1: If this error is received, try running “npm install”. Once this finishes, run “npm start”.



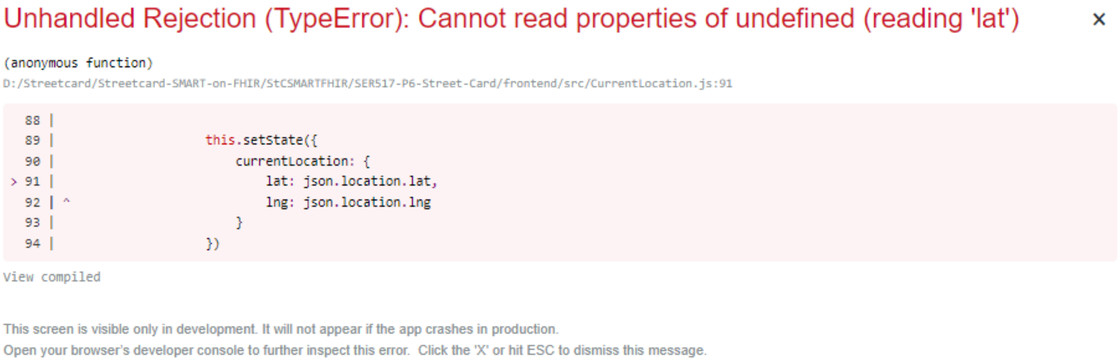
```
failed to solve: executor failed running [/bin/sh -c npm install -g npm@latest]: exit code: 1
```

b. Exit Code 2: If this error is received, Check your dockerfile. Make sure it is saved, and rerun the command. After this is complete, run “npm start”

22. If npm start launches localhost:3000. Then you are good to go! It may take some time to load



a. If you get “Unhandled Rejection (TypeError): Cannot read properties of undefined (reading 'lat')”, make sure you have a react app key placed in the ENV files. Ensure it is a google map API key. This may happen with other API keys that were not included/linked.

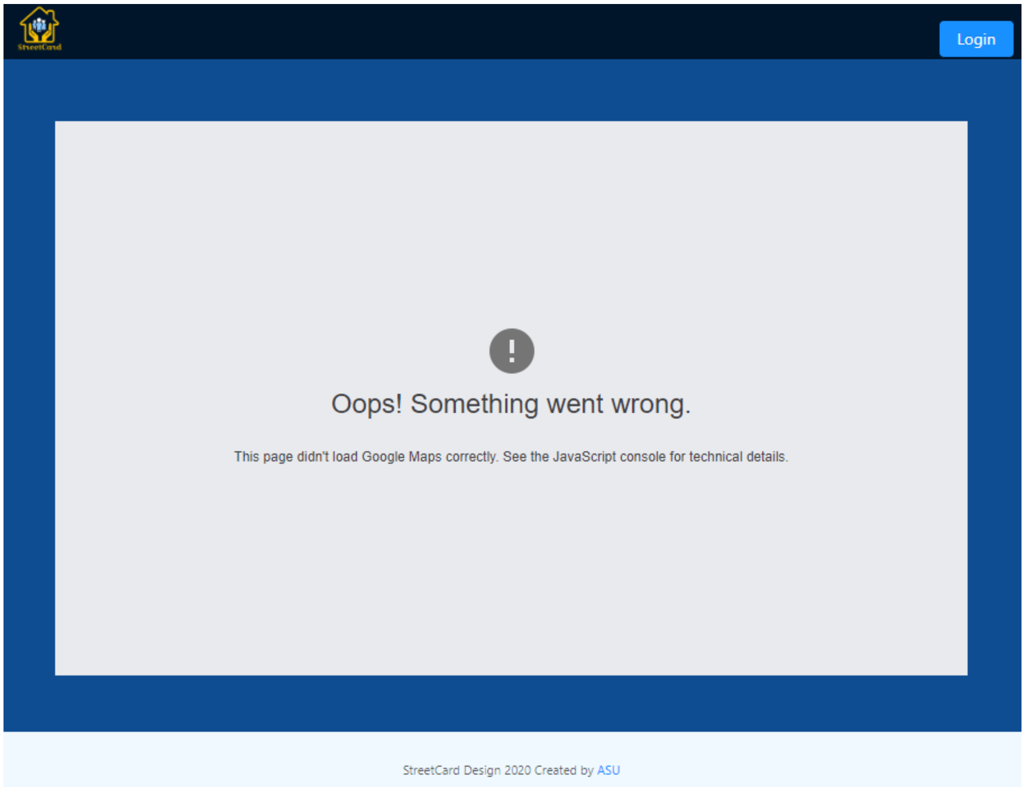


Unhandled Rejection (TypeError): Cannot read properties of undefined (reading 'lat')

```
(anonymous function)
D:/Streetcard/Streetcard-SMART-on-FHIR/StCSMARTFHIR/SER517-P6-Street-Card/frontend/src/CurrentLocation.js:91

                this.setState({
                    currentLocation: {
                        lat: json.location.lat,
                        lng: json.location.lng
                    }
                })

View compiled
```

This screen is visible only in development. It will not appear if the app crashes in production.
Open your browser's developer console to further inspect this error. Click the 'X' or hit ESC to dismiss this message.

b. Click the x in the top right corner to close this error and you should see this:



Oops! Something went wrong.

This page didn't load Google Maps correctly. See the JavaScript console for technical details.

StreetCard Design 2020 Created by ASU