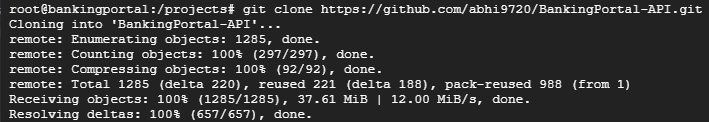
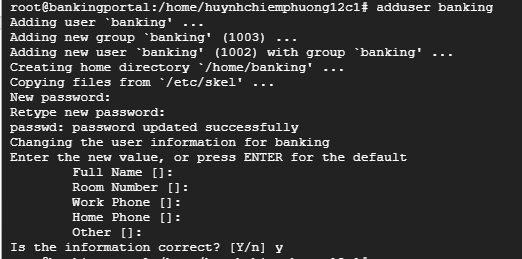
**I. BankingPortal-API using Java Spring Boot + MySQL**

Chuẩn bị: tạo user, folder( phân quyền), get source code cho dự án

1. Cài đặt công cụ để chạy dự án: java, mvn

-apt install openjdk-17-jdk openjdk-17-jre

-apt install maven

1. Cài đặt và cấu hình database: MySQL

-sudo apt-get install mysql-server

-mysql –u root

-create database banking;

-create user 'banking'@'%' identified by 'banking';

-grant all privileges on banking.\* to 'banking'@'%';

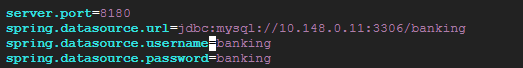
-flush privileges;

\_Sửa file cấu hình:

-vi /etc/mysql/mysql.conf.d/mysqld.cnf

bind-address: 127.0.0.1 -> 0.0.0.0: các sv bên ngoài có thể truy cập DB

-vi src/main/resources/application.properties



\_edit file pom.xml

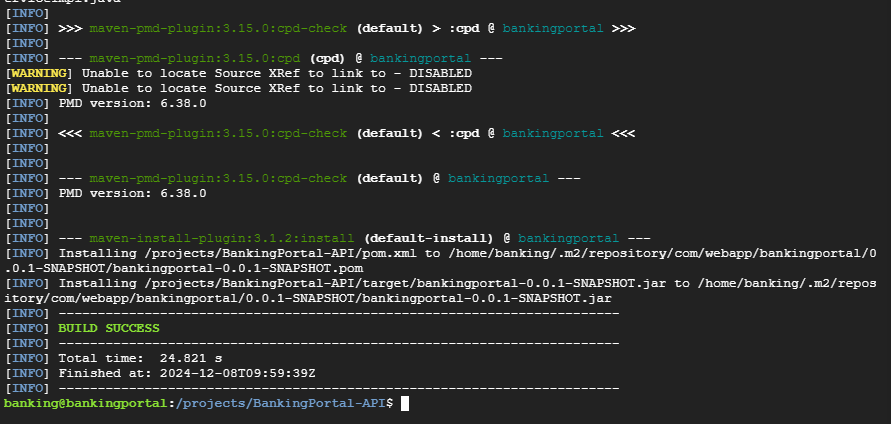
-Đổi <ruleset>/usr/pmd/rulesets/strings.xml</ruleset>

-> <ruleset>/rulesets/java/strings.xml</ruleset>

-Xoá: <ruleset>http://localhost/design.xml</ruleset>

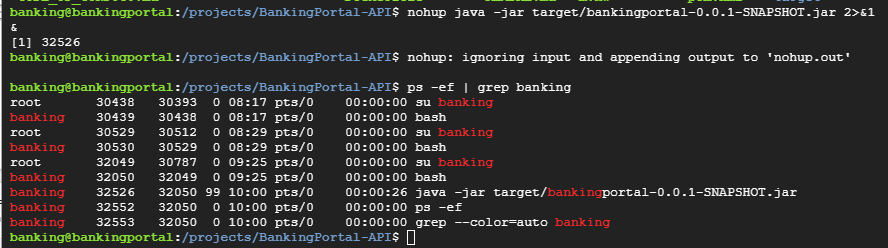
1. Build dự án

-mvn install -DskipTests=true

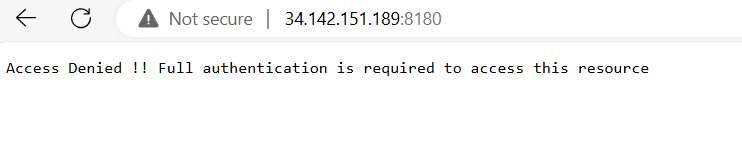


1. Run dự án

-nohup java –jar target/bankingportal-0.0.1-SNAPSHOT.jar 2>&1 &

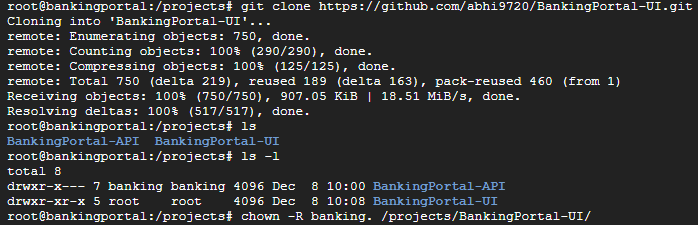


Truy cập Banking Portal API thành công



**II. Banking Portal UI**

1.Phân quyền, get source code



2.Cài đặt dependencies

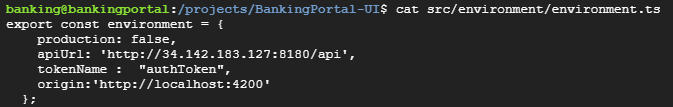
-apt npm install

- curl -fsSL https://deb.nodesource.com/setup\_18.x | sudo -E bash -ng

- npm install - g @angular/cli

\_Sửa file cấu hình:

- /src/environment/environment.prod.ts và src/environment/environment.ts



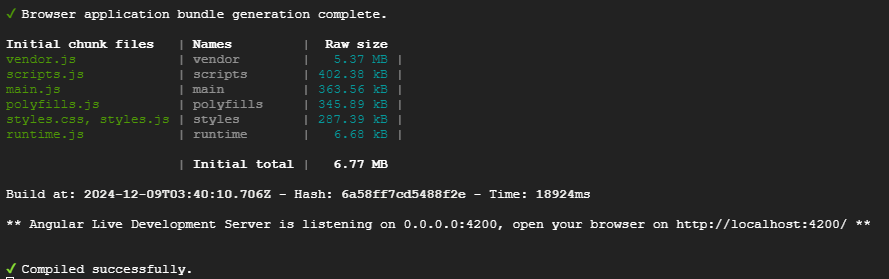
3.Build

-npm run build( ng build)

4.Chạy ứng dụng

C1: Chạy trực tiếp từ file angular.json

-npm start( ng serve –host 0.0.0.0): có thể truy cập bằng IP public



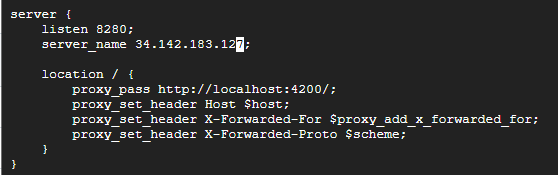
C2: Chạy bằng web server(Nginx)

Cài đặt nginx:

-apt install nginx –y

Cấu hình file dự án chạy trên ngnix:

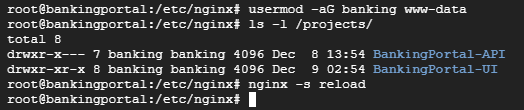
-vi /etc/nginx/conf.d/bankingapp.conf

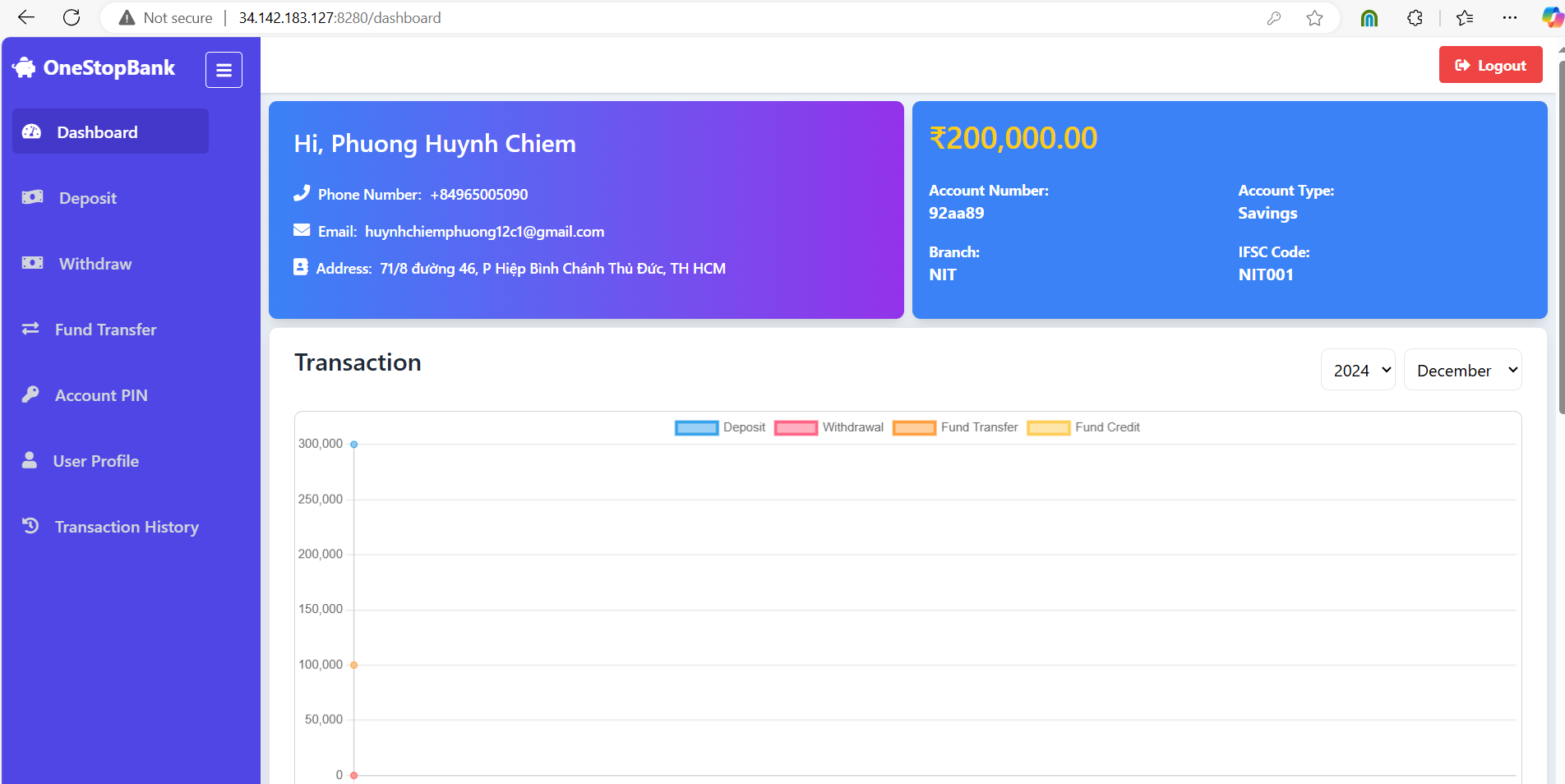


Phân quyền user nginx( www-data) có quyền chạy ứng dụng và reload service nginx:

-usermod -aG banking www-data

-nginx -s reload





-Create Account Pin

curl -X POST http://34.142.183.127:8180/api/account/pin/create -H "Authorization: Bearer eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiI5MmFhODkiLCJpYXQiOjE3MzM3NTMxNDIsImV4cCI6MTczMzgzOTU0Mn0.p3K3GTLV3hKWhlqHjLDLXeqtRV8u8yQhOaYVtedQbj74uo41h-IYM2blwrhe9gomqL8q4FPXy4P6gErw29qjCg" -H "Content-Type: application/json" -d '{ "pin": "1234", "password": "Acb123!@#" }'

**III. Build Docker**  
**Doker file Banking Portal API**

**##Build Stage##**

**From maven:3.5.4-jdk-8-alpine as build**

**Workdir /app**

**Copy . .**

**Run mvn install -DskipTests=true**

**##Run Stage##**

**From alpine:3.19**

**Run adduser –D banking**

**Run apk add openjdk8**

**Workdir /run**

**Copy –from=build /app/target/bankingportal-0.0.1-SNAPSHOT.jar /run/bankingportal-0.0.1-SNAPSHOT.jar**

**Run chown –R banking:banking /run**

**User banking**

**Expose 8180**

**Entrypoint java –jar /run/bankingportal-0.0.1-SNAPSHOT.jar**

Docker build –t bankingapi:v1 –f dockerfile .

Docker run –name bankingapi –dp 8180:8180 bankingapi:v1

**Doker file Banking Portal UI**

**## build stage ##**

**FROM node:18.18-alpine as build**

**WORKDIR /app**

**COPY . .**

**RUN npm install**

**RUN npm run build**

**## run stage ##**

**FROM nginx:alpine**

**RUN mkdir /run**

**COPY --from=build /app/dist/banking-portal /run**

**COPY nginx.conf /etc/nginx/nginx.conf**

**##### nginx.conf ####**

**user nginx;**

**worker\_processes 1;**

**error\_log /var/log/nginx/error.log warn;**

**pid /var/run/nginx.pid;**

**events {**

**worker\_connections 1024;**

**}**

**http {**

**include /etc/nginx/mime.types;**

**default\_type application/octet-stream;**

**log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '**

**'$status $body\_bytes\_sent "$http\_referer" '**

**'"$http\_user\_agent" "$http\_x\_forwarded\_for"';**

**access\_log /var/log/nginx/access.log main;**

**sendfile on;**

**keepalive\_timeout 65;**

**server {**

**listen 8280;**

**server\_name 34.142.183.127;**

**location / {**

**root /run;**

**index index.html;**

**try\_files $uri $uri/ /index.html;**

**}**

**error\_page 500 502 503 504 /50x.html;**

**location = /50x.html {**

**root /usr/share/nginx/html;**

**}**

**}**

**}**

-----------------------

**IV. Spring Boot Logs with Elasticsearch, Logstash and Kibana(ELK)**

1. Configure Spring Boot’s log file

- Trong thư mục: src/main/resources/application.properties

Thêm **logging.file=logs/app.log**

C1: 2 . Configure Logstash to Understand Spring Boot’s Log File

-Cấu hình file: logstash.conf

Example:

input {

file {

path => "/path/to/your/logfile.log" # Đường dẫn đến file log

start\_position => "beginning" # Đọc từ đầu file

sincedb\_path => "/dev/null" # Không lưu trạng thái

}

}

filter {

# Bộ lọc để phân tích log (nếu cần)

grok {

match => { "message" => "%{COMBINEDAPACHELOG}" } # Ví dụ phân tích log Apache

}

# Có thể thêm các bộ lọc khác như mutate, date, etc.

}

output {

elasticsearch {

hosts => ["http://localhost:9200"] # Địa chỉ Elasticsearch

index => "your-log-index-%{+YYYY.MM.dd}" # Tên chỉ mục

}

stdout { # In ra console để kiểm tra

codec => rubydebug

}

}

3. run logstash

bin/logstash -f path/to/logstash.conf

### **Giải thích Các Phần**

* **Input**:
  + file: Sử dụng plugin file để theo dõi log từ file.
  + path: Đường dẫn đến file log bạn muốn theo dõi.
  + start\_position: Xác định vị trí bắt đầu đọc log; "beginning" sẽ đọc từ đầu file.
  + sincedb\_path: Đường dẫn đến file lưu trạng thái; /dev/null không lưu trạng thái.
* **Filter**:
  + grok: Sử dụng để phân tích và cấu trúc log. Bạn có thể định nghĩa các mẫu để phân tích log theo định dạng cụ thể.
  + Các bộ lọc khác như mutate, date, và geoip có thể được thêm vào để tùy chỉnh thêm.
* **Output**:
  + elasticsearch: Chỉ định địa chỉ của Elasticsearch nơi log sẽ được gửi.
  + index: Tên chỉ mục trong Elasticsearch, có thể sử dụng biến thời gian để tạo chỉ mục hàng ngày.
  + stdout: In log ra console, hữu ích trong quá trình phát triển và kiểm tra.

C2: 2. log with logback

- add logback-elasticsearch-appender dependency to file pom.xml:

<dependency>  
 <groupId>com.internetitem</groupId>  
 <artifactId>logback-elasticsearch-appender</artifactId>  
 <version>1.6</version>  
 </dependency>

-Config file: src/main/resource/logback-spring.xml

<?xml version="1.0" encoding="UTF-8"?>

<configuration debug="true" scan="true">

<springProfile name="!local "> <!-- when the profile is not local, below appender will be applied -->

<appender name="CONSOLE" class="ch.qos.logback.core.ConsoleAppender">

<!-- encoders are assigned the type

ch.qos.logback.classic.encoder.PatternLayoutEncoder by default -->

<encoder>

<pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{35} - %msg %n</pattern>

</encoder>

</appender>

<appender name="ELASTIC" class="com.internetitem.logback.elasticsearch.ElasticsearchAppender">

<url>http://localhost:9200/\_bulk</url>

<index>demo-%date{yyyy-MM-dd}</index>

<type>tester</type>

<loggerName>es-logger</loggerName> <!-- optional -->

<errorLoggerName>es-error-logger</errorLoggerName> <!-- optional -->

<connectTimeout>30000</connectTimeout> <!-- optional (in ms, default 30000) -->

<errorsToStderr>false</errorsToStderr> <!-- optional (default false) -->

<includeCallerData>false</includeCallerData> <!-- optional (default false) -->

<logsToStderr>false</logsToStderr> <!-- optional (default false) -->

<maxQueueSize>104857600</maxQueueSize> <!-- optional (default 104857600) -->

<maxRetries>3</maxRetries> <!-- optional (default 3) -->

<readTimeout>30000</readTimeout> <!-- optional (in ms, default 30000) -->

<sleepTime>250</sleepTime> <!-- optional (in ms, default 250) -->

<rawJsonMessage>false</rawJsonMessage> <!-- optional (default false) -->

<includeMdc>false</includeMdc> <!-- optional (default false) -->

<maxMessageSize>100</maxMessageSize> <!-- optional (default -1 -->

<authentication class="com.internetitem.logback.elasticsearch.config.BasicAuthentication" /> <!-- optional -->

<properties>

<property>

<name>host</name>

<value>${HOSTNAME}</value>

<allowEmpty>false</allowEmpty>

</property>

<property>

<name>severity</name>

<value>%level</value>

</property>

<property>

<name>thread</name>

<value>%thread</value>

</property>

<property>

<name>stacktrace</name>

<value>%ex</value>

</property>

<property>

<name>logger</name>

<value>%logger</value>

</property>

</properties>

<headers>

<header>

<name>Content-Type</name>

<value>application/json</value>

</header>

</headers>

</appender>

<!-- We don't want to see a never ending cascade of connection failed ERRORS -->

<!-- <logger name="my-error-logger" level="OFF" /> -->

<logger name="es-logger" level="INFO" additivity="false">

<appender name="ES\_FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">

<encoder>

<pattern>%msg</pattern> <!-- This pattern is important, otherwise it won't be the raw Elasticsearch format anymore -->

</encoder>

</appender>

</logger>

<root level="info">

<appender-ref ref="CONSOLE" />

<appender-ref ref="ELASTIC" />

</root>

</springProfile>

</configuration>

----------------

**Cài đặt ELK**