Janani S K

[jananishivk@gmail.com](mailto:jananishivk@gmail.com)

**Language used:**

* NodeJs
* PostgreSQL

Tools Used:

* Visual Studio Code
* Postman

GITHUB LINK: <https://github.com/janani-shivkumar/BankAPI/>

TIME CONSUMED: 2 HRS (WITH BREAKS IN B/W) – FOR CODING, 1 HR TESTING

1. use PostgreSQL as a backend database

Used.

1. GET API to fetch a bank details, given branch IFSC code

**URL: localhost:3000/branches/ALLA0210002**

Code:

router.get('/:ifsc', checkauth, (req, res, next) => {

const ifsc = req.params.ifsc;

pool.query(' SELECT ID, NAME FROM BANKS INNER JOIN BRANCHES ON BANKS.ID = BRANCHES.BANK\_ID AND BRANCHES.IFSC = $1', [ifsc], (err, results) => {

if(results)

{

res.status(200).json({

bankDetails: results["rows"]

})

}

else{

res.status(404).json({

message: "No details for bank with ID - " + ifsc

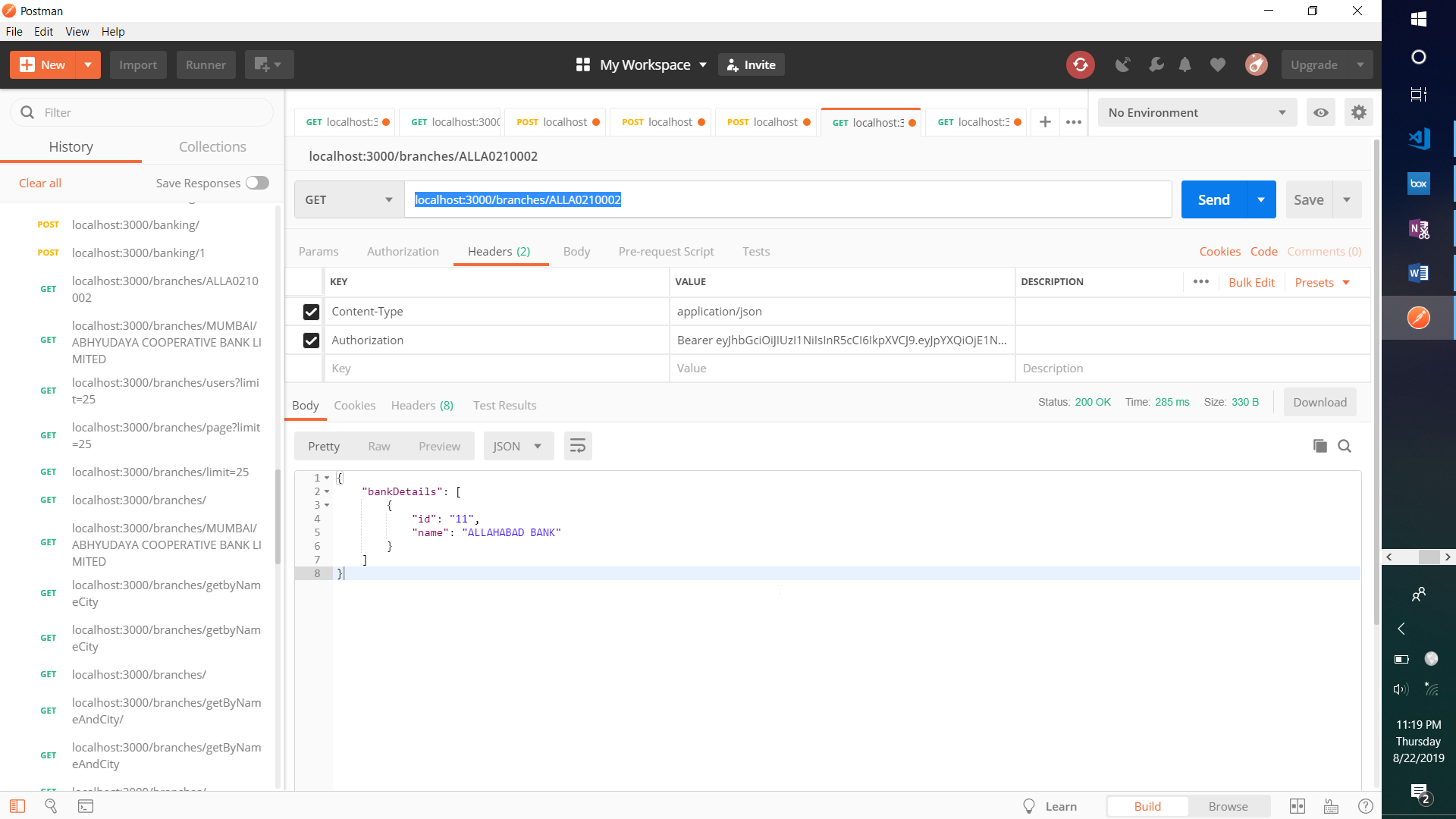
})

}

})

});

Output:



1. GET API to fetch all details of branches, given bank name and a city

URL: localhost:3000/branches/MUMBAI/ABHYUDAYA COOPERATIVE BANK LIMITED

CODE:

router.get('/:city/:name', checkauth, (req, res, next) => {

const name = req.body.name;

const city = req.body.city;

// SELECT \* FROM BRANCHES INNER JOIN BANKS ON BANKS.ID = BRANCHES.BANK\_ID WHERE BANKS.NAME = 'ABHYUDAYA COOPERATIVE BANK LIMITED' AND BRANCHES.CITY = 'MUMBAI';

pool.query(' SELECT \* FROM BRANCHES INNER JOIN BANKS ON BANKS.ID = BRANCHES.BANK\_ID WHERE BANKS.NAME = $1 AND BRANCHES.CITY = $2', [name, city], (err, results) => {

res.status(200).json({

err: err,

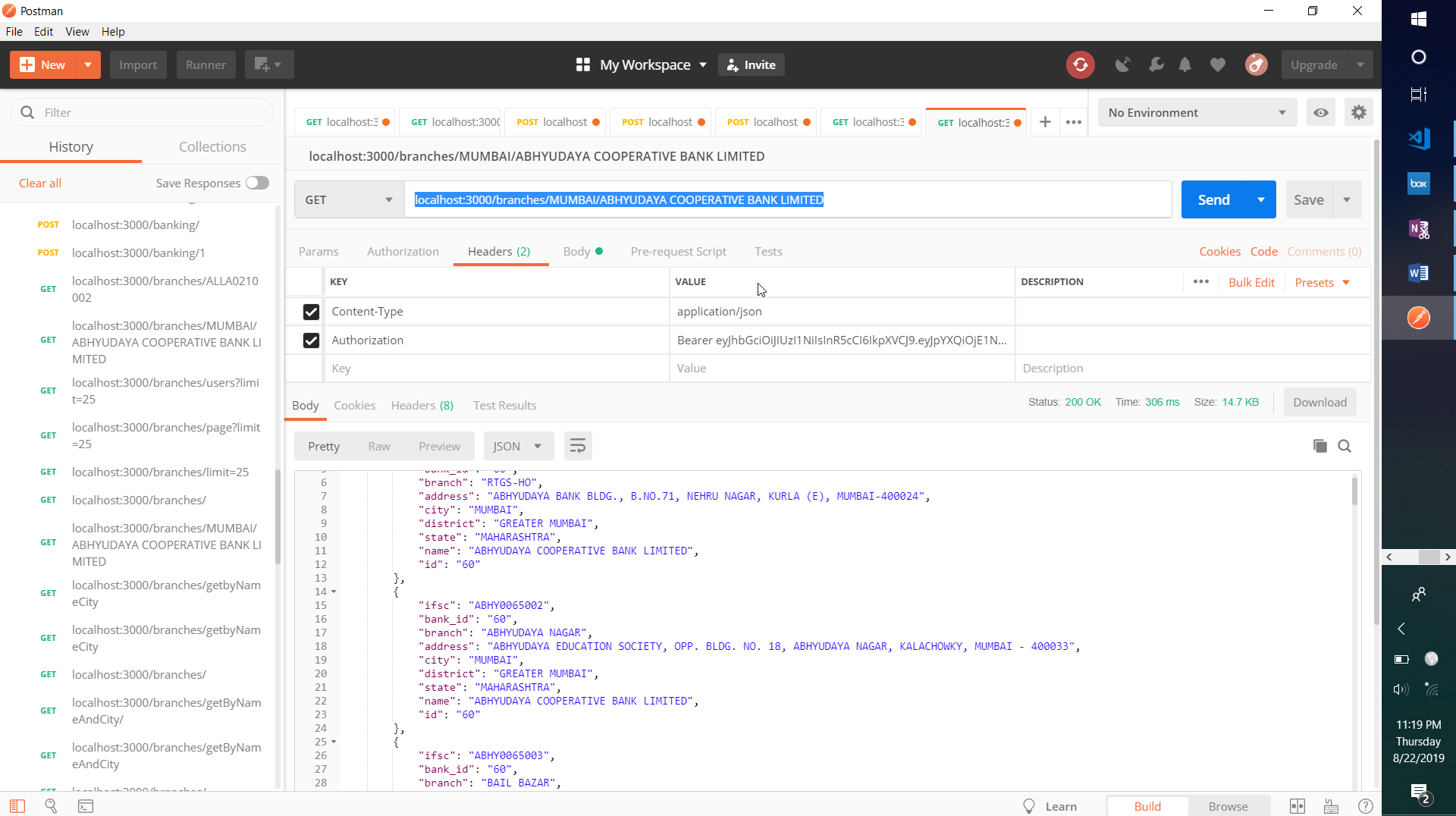
res: results["rows"]

})

})

})

OUTPUT:



1. APIs should be authenticated using a JWT key, with validity = 5 days

CODE:

router.post('/login', (req, res, err) => {

email = req.body.emailid

pwd = req.body.password

pool.query('SELECT \* FROM users\_tbl WHERE emailid = $1',[email], (err, results) => {

if(results["rowCount"])

{

// console.log(results)

if(results["rows"][0]["password"] == pwd)

{

const token = jwt.sign({

email: results["rows"]["emailid"],

id: results["rows"]["id"]

}, process.env.JWT\_KEY,

{

**expiresIn: "120h"**

})

res.status(200).json({

message: "User logged in!",

UserDetails: results["rows"],

token: token

})

}

else{

res.status(404).json({

message: "Entered wrong password!"

})

}

}

else{

res.status(404).json({

message: "User not found!"

})

}

})

})