A Project Report on

Creating Back-End For Online Business Site Utilizing FastAPI

Submitted by

SILASH TRIKHATRI (20IT103004)

Under the supervision and guidance of

Dr. Om Prakash Sharma

Internal Guide

Under the supervision and Internship guidance of

MR. MOHSUF PRADHAN

SAJILOKHOJ PVT LTD

In the partial fulfilment of requirements for the award of Degree in

Master of Computer Applications

Batch 2021-2022

Submitted to the

SCHOOL OF INFORMATION TECHNOLOGY

SRM UNIVERSITY SIKKIM

TADONG, GANGTOK, EAST SIKKIM - 737102

DECLARATION

I/We hereby declare that the work recorded in this project report entitled **"Creating Back-End For Online Business Site Utilizing FastAPI"**, in partial fulfillment for the requirements for the award of Degree in Master of Computer Applications from SRM University Sikkim, is a faithful and bonafide work carried out under the supervision and guidance of Dr. OM PRAKASH SHARMA from April 2022 to August 2022.

The results of this investigation reported in this project have so far not been reported for any Degree. The assistance and help received during the course of the investigation have been duly acknowledged.

STUDENT SIGNATURE

Name: Silash Trikhatri

Registration No: 20IT103004

CERTIFICATE OF ACCEPTANCE

This is to certify that Mr. SILASH TRIKHATRI bearing Registration No. 20IT103004 of School of Information Technology, SRM University Sikkim has worked on the project entitled **Creating Back-End For Online Business Site Utilizing FastAPI** under the supervision of

Dr. OM PRAKASH SHARMA, School of Information Technology, Shri Ramasamy Memorial University Sikkim. The project was carried out from April 2022 to August 2022.

The project is hereby accepted by the School of Information Technology, SRM University Sikkim, in partial fulfilment of the requirements for the award of Degree in Master of Computer Application.

Dr. Om Prakash Sharma

HOD(IT)

School of Information Technology

SRM University Sikkim

HEAD OF DEPARTMENT
SCHOOL OF INFORMATION TECHNOLOGY
SRM UNIVERSITY SIKKIM

BONAFIDE CERTIFICATE

Certified that this project report titled **Creating Back-End For Online Business Site Utilizing FastAPI** is the bonafide work of **Mr. Silash Trikhatri (20IT103004)** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein is not part of any other report or dissertation on the basis of which a degree or award was conferred on an earlier occasion to this or any other candidate.

Submitted for the viva-voce examination held on 23/8/2022

HOD- SCHOOL OF IT

HEAD OF DEPARTMENT
SCHOOL OF INFORMATION TECHNOLOGY
SRM UNIVERSITY SIKKIM

GUIDE

ASSOCIATE DEAN

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

The broad range of web-applications are moving on creating an API for an e-commerce website. APIs are increasing day-by-day in web-development and web-application projects where it has been popular for accessing back-end databases through the front-end with the use of JSON requests. Previously, connecting and manipulating data from databases were quite difficult for the developers so, most developers nowadays prefer using APIs for the web-developments. API created for this project is based on the e-commerce platform where there are customers, sellers and admin. This projects views on the large numbers of customer who can buy and sell the goods through the website.

ACKNOWLEDGMENT

I would like to express my special thanks and gratitude to **Dr. OM PRAKASH SHARMA** as well as **Mr. MOHSUF PRADHAN** for their guidance and support in completing my project. I would also extend my gratitude to the HOD "DR. OM PRAKASH SHARMA" and CEO of SAJILOKHOJ PVT LTD "Mr. AKASH SHARMA" for giving me the opportunity to work as an intern and our project coordinator "MISS SABNA SHARMA" and Associate Dean

"Dr. S. Sundar Rajan" for proving us with all the facilities that are required.

Silash Trikhatri (20IT103004)

MCA IV Semester

INTERNSHIP CERTIFICATE

Sajilo Khoj
THE QUALITY SERVICE YOU EXPECT

CERTIFICATE
OF APRICIATION

This certificate is proudly presented to

Silash Trikhatri

for successfully completing internship as Jr.Software Developer
as part of the Sajilokhoj Pvt. Ltd.
From 11 April 2022 to 11 July 2022

20-July-2022
Date

Akash Sharma
CEO

Date of Issue : 20 July, 2022

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 Overview

This project is based on back-end part of development for web applications. The main motive of this project is creating an API for the E-Commerce website. E-Commerce is most popular form of shopping in present day. E-Commerce is a website that permits individuals to trade actual merchandise, administrations, and computerized items over the web instead of at a physical area. Through an internet business site, a business can handle orders, acknowledge installments, oversee delivery and planned operations, and give client support. Much like a traditional physical retail store, e-commerce websites allow consumers and businesses to buy and sell to one another on a designated platform. The main difference between e-commerce and physical commerce, however, is that e-commerce transactions occur entirely over the internet rather than at a brick-and-mortar location. E-Commerce is the activity of electronically buying or selling of products on online services or over the Internet. E-commerce draws on technologies such as mobile commerce, electronic funds transfer, supply chain management, Internet marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems. E-commerce is in turn driven by the technological advances of the semiconductor industry, and is the largest sector of the electronics industry. In E-Commerce, customers can invest less energy looking for what they need. They can without much of a stretch peruse numerous things all at once and purchase what they like. When on the web, clients can find things that are accessible in actual stores far away from them or not tracked down in their region. One of the greatest benefits of online business to business that keep dealers intrigued by internet selling is cost decrease. Numerous dealers need to pay parts to keep up with their actual store. They might have to pay extra straightforward costs like lease, fixes, store configuration, stock and so forth. Much of the time, even subsequent to putting resources into administrations, stock, upkeep and labor force, dealers don't get wanted benefits and ROI. A significant benefit of online business to business is that dealers can give adaptability to clients. One feature is that the item and administrations are prepared 24x7. The outcome is that dealer can offer his thing

any spot, any time. Each interaction is quicker when you start selling on the web. Online business commercial centers offer you a smoothed out coordinated factors or conveyance framework. This means the purchasers request gets conveyed effectively. Item returns the executives is another in addition to point that can be dealt with rapidly - you either discount the installments or give a substitution. Rapidly activities might in fact be applied while answering business sector requests. Consider this web-based business model - when a purchaser sees that a thing is unavailable, he can tap on the 'Tell Me' Choice. This illuminates him when that thing is ready to move once more. It additionally illuminates merchants that they need to restock that thing so they can get more purchasers. Next comes the patterns - Suppose there is interest for voice enacted individual collaborators, a vender can promptly answer that interest by loading these things. He is certain that this item will sell and has seen similar occurring with different merchants to. Shippers can make bargains, advancements rapidly as well. This draws in clients and increment chances of making more deals. Online business venders might design and apply coupons when they like - even modify such proposals for their own store. An application programming point of interaction (API) is a way for at least two PC projects to speak with one another. It is a kind of programming point of interaction, offering a support of different bits of software. A record or standard that depicts how to construct or utilize such an association or point of interaction is called an API particular. A PC framework that fulfills this guideline is said to carry out or uncover an API. The term API might allude either to the determination or to the execution. Rather than a UI, which interfaces a PC to an individual, an application programming point of interaction interfaces PCs or bits of programming to one another. It isn't expected to be utilized straight by an individual (the end client) other than a software engineer who is integrating it into the product. An API is frequently comprised of various parts which go about as apparatuses or administrations that are accessible to the developer. A program or a software engineer that utilizes one of these parts is said to call that piece of the API. The hits that spread the word about up the API are additionally as subroutines, techniques, solicitations, or endpoints. An API particular characterizes these calls, implying that it clears up how for use or carry out them. One reason for APIs is to conceal the interior subtleties of how a framework functions, uncovering just those parts a software engineer will view as valuable and keeping them reliable regardless of whether the inside subtleties later change. An API might be exceptionally worked for a

specific set of frameworks, or it could be a common norm permitting interoperability among numerous frameworks. The term API is frequently used to allude to web APIs, which permit correspondence between PCs that are joined by the web. There are additionally APIs for programming dialects, programming libraries, PC working frameworks, and PC equipment. APIs started during the 1940s, however the term didn't arise until the 1960s and 1970s. Late improvements in using APIs have prompted the ascent in prevalence of microservices, which are eventually approximately coupled administrations gotten to through open APIs. An API simplifies on programming by abstracting the fundamental implementation and just uncovering articles or activities the designer needs. While a graphical point of interaction for an email client could furnish a client with a button that plays out every one of the means for bringing and featuring new messages, an API for document input/result could give the designer a capability that duplicates a record starting with one area then onto the next without expecting that the engineer comprehends the document framework tasks happening in the background. APIs carry another degree of modularity to applications. APIs allow designers to use the aptitude of different applications. At the point when an association fosters an application, they never again need to rehash an already solved problem with regards to things like confirmation, correspondence, installment handling, and guides. Rather designers can use the consistent module abilities and usefulness of APIs. APIs permit applications and framework parts to speak with one another on inward organizations as well as over the Internet. They've become basic to big business endeavors to make interior applications and administrations open over the Internet to business clients, accomplices, providers, and other outsiders. The language used for creating this API is Python 3 and the framework used is FastAPI. The reason behind creating back-end API in FastAPI is because the development of data frameworks is partitioned into two huge parts: front-end development and back-end development. For backend development, the most well-known programming languages are: C++, C#, Java, python, php. Be that as it may, nobody writes in "pure" dialects presently, for the most part utilized supporting tools like libraries and web frameworks. This project is committed to an outline of the most promising web framework for a programming language Python - FastAPI. FastAPI also assumes that for each API access point there is some limitation. In this case, a Boolean expression that defines the conditions when which the user will have access to this access point. This mechanism allows you to very flexibly configure,

or rather, restrict access to a specific functionality of the system. The web terminal system assumes the presence many roles (administrator, seller, customer, and so on), as well as a large number of rights (access to sessions, access to the user panel, etc.). Having such role separation system is easy, in terms of writing code, and in a very understandable way for a person to divide the functionality of the entire system for a user with a specific role or rights. To integrate this system directly with FastAPI itself, used a convenient and flexible tool that is part of the framework itself and called the Dependency Injection system. Its feature is ease of use, as well as ease of integration of third-party components with FastAPI.

1.2 Company Profile

Sajilokhoj is one of the finest Security and Technology Training and Consulting organization, focusing on a range of IT Security Trainings and Information Security Services. Sajilokhoj was established in the year 2019 by a team of experienced and enthusiastic professionals, who have more than 10 years of industry experience. They provide professional training, certification & consulting services related to all areas of Information Technology and Cyber Security. Sajilokhoj offers complete training and consulting solutions to its customers globally. Whether the requirements are technical services, certification or customized training, Sajilokhoj has consistently delivered the highest quality and best success rates in the industry.

1.3 Training Undergone

- Researched Django.
- Learned FastAPI.
- Researched in back-end.
- Learned using PostgresSQL.
- Learned using postman for testing FastAPI.
- Learned how to keep codes neat using routers.
- Learned installing packages in python.
- Learned how to access the database using python.

- Learned using pgAdmin4 for database.
- Learned sql-alchemy for database handling.
- Researched on encrypting password to store in database.
- Researched on access token.
- Researched on how the users are managed using access token.
- Researched on authentication and restrictions on different user groups.
- Researched on identifying the logged in user using access token provided.
- Researched on generating OTPs.
- Researched on sending OTP to email using SMTP protocol.
- Researched on uploading image in API.
- Researched on how the fetching of images is done using API.

1.4 Responsibility

- Assigned to research on different python frameworks.
- Assigned to research on e-commerce websites.
- Assigned to create users.
- Assigned to implement OTPs.
- Assigned to implement login and create access tokens.
- Assigned to beta test marketplace website of different company.
- Assigned to write contents for different company's website.
- Assigned to implement restrictions for different types of users.
- Assigned to implementations of products profile.
- Assigned to implement user dashboards view and update the dashboard.
- Assigned to implement customer's order.
- Assigned to implement multiple photos for products.
- Assigned to implement fetching photos.

1.5 Overview of Report

The project “Creating Back-End For Online Business Site Utilizing FastAPI” aims on creating an API for the e-commerce website.

The aim of this project is to achieve all the CRUD functionality through REST API for e-commerce.

The programming language used for this project is python and the application used for developing the API for this project are Pycharm, Postman and PostgresSQL. PyCharm is a Python IDE with complete set of tools for Python development. In addition, the IDE provides capabilities for professional Web development using the Django framework. Code faster and with more easily in a smart and configurable editor with code completion, snippets, code folding and split windows support. Pycharm also features in intelligent coding assistance, intelligent code editor, smart code navigation, built-in developer tools, etc. Postman is an API client that makes it easy for developers to create, share, test and document APIs. This is done by allowing users to create and save simple and complex HTTP/s requests, as well as read their responses. PostgreSQL is a powerful, open-source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. Reason for using PostgresSQL rather than MySQL is because PostgreSQL is faster when dealing with massive datasets, complicated queries, and read-write operations. On the other hand, MySQL is known to be faster for read-only commands.

1.6 Organization of the Report

This project mainly focuses on Creating back-end for online business for online business site utilizing FastAPI. There are chapters that deals with various details –

Chapter 1

This chapter briefs on the introduction and the motive of the project. It deals with overview of the project and problem statement. It outlines the entire project and provides the detail on the problem statement.

Chapter 2

This chapter includes the literature survey which involves all the problems faced during working on this project.

Chapter 3

This chapter contains the methodology of the project where detailed information about how the system was designed using UML Class diagram.

Chapter 6

This chapter contains the results of the project where all the screenshots of the output which was captures while testing API.

Chapter 7

This chapter contains the conclusions of the project.

CHAPTER 2

PROBLEM STATEMENT AND OBJECTIVES

2.1 Problem statement

The problems faced while creating an e-commerce website without API is that it will be lengthy and confusing process. With the help of API, front-end developers find it easier to fetch the required data without directly coding to access the particular data. While using API, back-end codes is not visible to the front-end developers and is not accessible to the front-end developers which helps developers to debug the codes easily. FastAPI is python-based framework to create REST APIs which uses HTTP and support Transport Layer Security (TLS) encryption, which is a standard that keeps an internet connection private and checks that the data sent between two systems (server-to-server or server-to-client) is encrypted and unmodified.

2.2 Objectives

- To understand about E-Commerce business.
- To create a FastAPI that helps in faster execution.
- To implement and develop the applications using the FastAPI.

CHAPTER 3

METHODOLOGY

3.1 Existing Methodology

The methodology used in the implementation of the software is the Agile Model of System Development Life Cycle, which allows room for scalability as time goes on. Creating back-end for online business site utilizing FastAPI would help in e-commerce to manage data. The method used in the design and collections of information from various sources.

3.2 Introduction on system

Systems are designed to make development of a project easier and the problems that are yet to occur known. All systems are designed to its respective formats and requirements. This project focuses on how the management of an ecommerce website is made easier after using API.

Using of UML Class diagram helps illustrate the data models accurately, regardless of the complexity involved with the classes and data. UML Class diagram clearly maps out the structure of a particular system by modeling its classes, attributes, operations and relationships between objects.

3.3 UML Class diagram

Given below is the UML Class diagram, which shows all the classes and tables and their assigned operations.

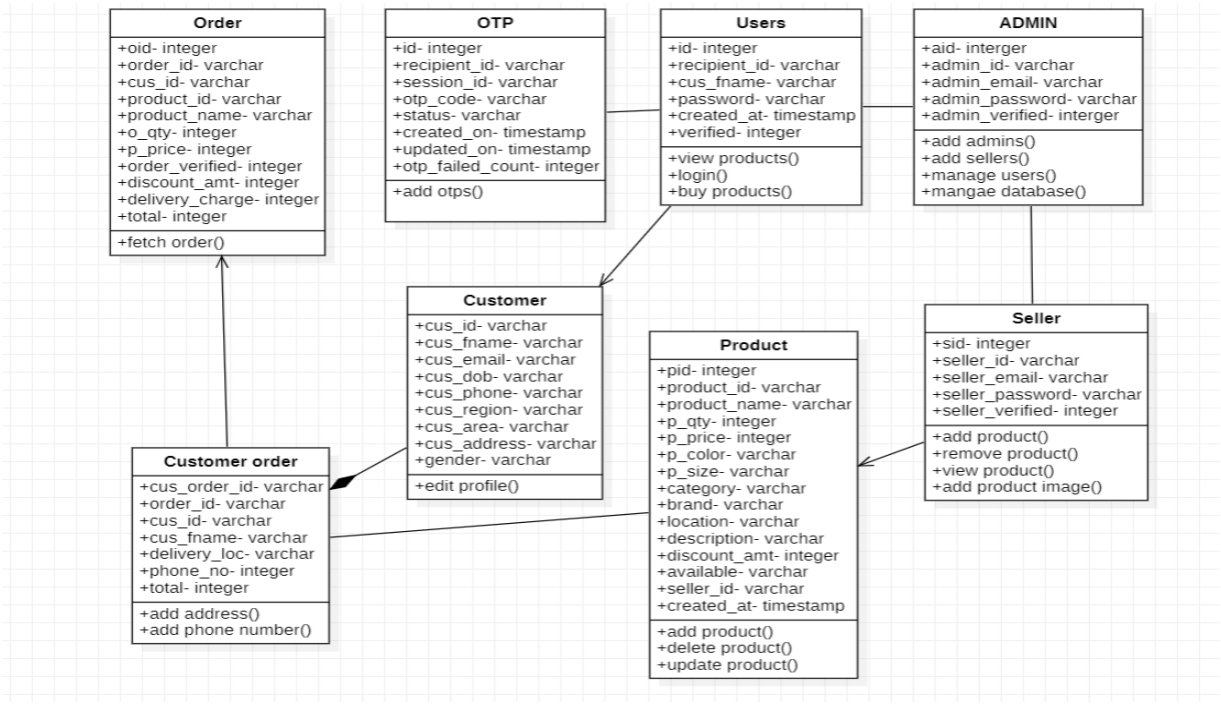


Fig 1. UML Class diagram

3.4 Packages

- **Anyio (3.5.0) -** AnyIO is an asynchronous networking and concurrency library that works on top of either asyncio or trio. It implements trio-like structured concurrency (SC) on top of asyncio, and works in harmony with the native SC of trio itself. Applications and libraries written against AnyIO's API will run unmodified on either asyncio or trio. AnyIO can also be adopted into a library or application incrementally – bit by bit, no full refactoring necessary. It will blend in with native libraries of your chosen backend.

- **Asgiref (3.5.2) -** ASGI is a standard for Python asynchronous web apps and servers to communicate with each other, and positioned as an asynchronous successor to WSGI. This package includes ASGI base libraries, such as:

- Sync-to-async and async-to-sync function wrappers, asgiref.sync
- Server base classes, asgiref.server
- A WSGI-to-ASGI adapter, in asgiref.wsgi

- **Asyncpg (0.25.0)** - asyncpg is a database interface library designed specifically for PostgreSQL and Python/asyncio. asyncpg is an efficient, clean implementation of PostgreSQL server binary protocol for use with Python's asyncpg framework. asyncpg requires Python 3.6 or later and is supported for PostgreSQL versions 9.5 to 14. Older PostgreSQL versions or other databases implementing the PostgreSQL protocol may work, but are not being actively tested.

- **Bcrypt (3.2.0)** - Good password hashing for your software and your servers
- beautifulsoup4(4.11.1) - Beautiful Soup is a library that makes it easy to scrape information from web pages. It sits atop an HTML or XML parser, providing Pythonic idioms for iterating, searching, and modifying the parse tree.
- **Cachetools (5.0.0)** - This module provides various memoizing collections and decorators, including variants of the Python Standard Library's @lru_cache function decorator.
- **Certifi (2021.10.8)** - Certifi provides Mozilla's carefully curated collection of Root Certificates for validating the trustworthiness of SSL certificates while verifying the identity of TLS hosts. It has been extracted from the Requests project.
- **Charset-normalizer (2.0.12)** - A library that helps you read text from an unknown charset encoding. Motivated by chardet, I'm trying to resolve the issue by taking a new approach. All IANA character set names for which the Python core library provides codecs are supported.

- **Click (8.1.2)** - Click is a Python package for creating beautiful command line interfaces in a composable way with as little code as necessary. It's the "Command Line Interface Creation Kit". It's highly configurable but comes with sensible defaults out of the box. It aims to make the process of writing command line tools quick and fun while also preventing any frustration caused by the inability to implement an intended CLI API.

- **Colorama (0.4.4)** - Makes ANSI escape character sequences (for producing colored terminal text and cursor positioning) work under MS Windows.

- **Cryptography (37.0.1)** – Cryptography s a package which provides cryptographic recipes and primitives to Python developers. Our goal is for it to be your "cryptographic standard library". It supports Python 3.6+ and PyPy3 7.2+. Cryptography includes both high level recipes and low-level interfaces to common cryptographic algorithms such as symmetric ciphers, message digests, and key derivation functions.
- **Cssselect (1.1.0)** - cssselect parses CSS3 Selectors and translate them to XPath 1.0 expressions. Such expressions can be used in lxml or another XPath engine to find the matching elements in an XML or HTML document.

- **Databases (0.5.5)** - Databases gives you simple asyncio support for a range of databases. It allows you to make queries using the powerful SQLAlchemy Core expression language, and provides support for PostgreSQL, MySQL, and SQLite. Databases is suitable for integrating against any async Web framework, such as Starlette, Sanic, Responder, Quart, aiohttp, Tornado, or FastAPI.

- **Dnspython (2.2.1)** - dnspython is a DNS toolkit for Python. It supports almost all record types. It can be used for queries, zone transfers, and dynamic updates. It supports TSIG authenticated messages and EDNS0. dnspython provides both high- and low-level access to DNS. The high-level classes perform queries for data of a given name, type, and class, and return an answer set. The low-level classes allow direct manipulation of DNS zones, messages, names, and records.
- **Ecdsa (0.17.0)** - This is an easy-to-use implementation of ECC (Elliptic Curve Cryptography) with support for ECDSA (Elliptic Curve Digital Signature Algorithm), EdDSA (Edwards-curve Digital Signature Algorithm) and ECDH (Elliptic Curve Diffie-Hellman), implemented purely in Python, released under the MIT license. With this library, you can quickly create key pairs (signing key and verifying key), sign messages, and verify the signatures. You can also agree on a shared secret key based on exchanged public keys. The keys and signatures are very short, making them easy to handle and incorporate into other protocols.

- **Email-validator (1.2.0)** - A robust email address syntax and deliverability validation library for Python by Joshua Tauberer. This library validates that a string is of the form name@example.com. This is the sort of validation you would want for an email-based login form on a website.
- **Fastapi (0.75.2)** - FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.6+ based on standard Python type hints.
- **Flask (2.1.1)** - Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks. Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.
- **Flask-SQLAlchemy (2.5.1)** - Flask-SQLAlchemy is an extension for Flask that adds support for SQLAlchemy to your application. It aims to simplify using SQLAlchemy with Flask by providing useful defaults and extra helpers that make it easier to accomplish common tasks.
- **Greenlet (1.1.2)** - Greenlets are lightweight coroutines for in-process concurrent programming. The "greenlet" package is a spin-off of Stackless, a version of CPython that supports micro-threads called "tasklets". Tasklets run pseudo-concurrently (typically in a single or a few OS-level threads) and are synchronized with data exchanges on "channels". A "greenlet", on the other hand, is a still more primitive notion of micro-thread with no implicit scheduling; coroutines, in other words. This is useful when you want to control exactly when your code runs. You can build custom scheduled micro-threads on top of greenlet; however, it seems that greenlets are useful on their own as a way to make advanced control flow structures. For example, we can recreate generators; the difference with Python's own generators is that our generators can call nested functions and the nested functions can yield values too.
- **Gunicorn (20.1.0)** - Gunicorn 'Green Unicorn' is a Python WSGI HTTP Server for UNIX. It's a pre-fork worker model ported from Ruby's Unicorn project. The

Gunicorn server is broadly compatible with various web frameworks, simply implemented, light on server resource usage, and fairly speedy.

- **h11 (0.13.0)** - This is a little HTTP/1.1 library written from scratch in Python, heavily inspired by hyper-h2. It's a "bring-your-own-I/O" library; h11 contains no IO code whatsoever. This means you can hook h11 up to your favourite network API, and that could be anything you want: synchronous, threaded, asynchronous, or your own implementation of RFC 6214 – h11 won't judge you. (Compare this to the current state of the art, where every time a new network API comes along then someone gets to start over reimplementing the entire HTTP protocol from scratch.) Cory Benfield made an excellent blog post describing the benefits of this approach, or if you like video then here's his PyCon 2016 talk on the same theme.
- **httplib2 (0.20.4)** - A comprehensive HTTP client library, httplib2 supports many features left out of other HTTP libraries.
    - HTTP and HTTPS

      HTTPS support is only available if the socket module was compiled with SSL support.

    - Keep-Alive

      Supports HTTP 1.1 Keep-Alive, keeping the socket open and performing multiple requests over the same connection if possible.

    - Authentication

      The following three types of HTTP Authentication are supported. These can be used over both HTTP and HTTPS.

        - Digest
        - Basic
        - WSSE

    - Caching

The module can optionally operate with a private cache that understands the Cache-Control: header and uses both the ETag and Last-Modified cache validators. Both file system and memcached based caches are supported.

- All Methods

  The module can handle any HTTP request method, not just GET and POST.

- Redirects

  Automatically follows 3XX redirects on GETs.

- Compression

  Handles both ‘deflate’ and ‘gzip’ types of compression.

- Lost update support

  Automatically adds back ETags into PUT requests to resources we have already cached. This implements Section 3.2 of Detecting the Lost Update Problem Using Unreserved Checkout

- Unit Tested

  A large and growing set of unit tests.

- **Httptools (0.4.0)** - httptools contains two classes httptools.HttpRequestParser, httptools.HttpResponseParser (fulfilled through llhttp) and a function for parsing URLs httptools.parse_url (through http-parse for now).
- **Idna (3.3)** - Support for the Internationalised Domain Names in Applications (IDNA) protocol as specified in RFC 5891. This is the latest version of the protocol and is sometimes referred to as “IDNA 2008”.
- **importlib-metadataz (4.11.3)** - This package supplies third-party access to the functionality of importlib.metadata including improvements added to subsequent Python versions
- **itsdangerous (2.1.2)** - Various helpers to pass data to untrusted environments and to get it back safe and sound. Data is cryptographically signed to ensure that a token has not been tampered with. It’s possible to customize how data is serialized. Data is

compressed as needed. A timestamp can be added and verified automatically while loading a token.

- **Jinja2 (3.1.1)** - Jinja is a fast, expressive, extensible templating engine. Special placeholders in the template allow writing code similar to Python syntax. Then the template is passed data to render the final document.
- **Lxml (4.8.0)** - lxml is a Pythonic, mature binding for the libxml2 and libxslt libraries. It provides safe and convenient access to these libraries using the ElementTree API.
- **MarkupSafe (2.1.1)** - MarkupSafe implements a text object that escapes characters so it is safe to use in HTML and XML. Characters that have special meanings are replaced so that they display as the actual characters. This mitigates injection attacks, meaning untrusted user input can safely be displayed on a page.
- **Numpy (1.22.3)** - NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.
- **passlib (1.7.4)** - Passlib is a password hashing library for Python 2 & 3, which provides cross-platform implementations of over 30 password hashing algorithms, as well as a framework for managing existing password hashes. It's designed to be useful for a wide range of tasks, from verifying a hash found in /etc/shadow, to providing full-strength password hashing for multi-user applications.
- **Premailer (3.10.0)** - This project is actively looking for corporate sponsorship. If you want to help making this an active project consider pinging Peter and we can talk about putting up logos and links to your company.
- **Protobuf (3.20.1)**
- **psycopg2-binary (2.9.3)** - Psycopg is the most popular PostgreSQL database adapter for the Python programming language. Its main features are the complete implementation of the Python DB API 2.0 specification and the thread safety (several threads can share the same connection). It was designed for heavily multi-threaded applications that create and destroy lots of cursors and make a large number of concurrent "INSERT"s or "UPDATE"s. Psycopg 2 is mostly implemented in C as a libpq wrapper, resulting in being both efficient and secure. It features client-side and

server-side cursors, asynchronous communication and notifications, “COPY TO/COPY FROM” support. Many Python types are supported out-of-the-box and adapted to matching PostgreSQL data types; adaptation can be extended and customized thanks to a flexible objects adaptation system.

- **Pydantic (1.9.0)** - Fast and extensible, pydantic plays nicely with your linters/IDE/brain. Define how data should be in pure, canonical Python 3.6+; validate it with pydantic.
- **Pyparsing (3.0.8)** - The pyparsing module is an alternative approach to creating and executing simple grammars, vs. the traditional lex/yacc approach, or the use of regular expressions. The pyparsing module provides a library of classes that client code uses to construct the grammar directly in Python code.
- **python-multipart (0.0.5)**
- **pytz (2022.1)** - pytz brings the Olson tz database into Python. This library allows accurate and cross platform timezone calculations using Python 2.4 or higher. It also solves the issue of ambiguous times at the end of daylight-saving time, which you can read more about in the Python Library Reference.
- **PyYAML (6.0)** - YAML is a data serialization format designed for human readability and interaction with scripting languages. PyYAML is a YAML parser and emitter for Python. PyYAML features a complete YAML 1.1 parser, Unicode support, pickle support, capable extension API, and sensible error messages. PyYAML supports standard YAML tags and provides Python-specific tags that allow to represent an arbitrary Python object. PyYAML is applicable for a broad range of tasks from complex configuration files to object serialization and persistence.
- **Requests (2.27.1)** - Requests is a simple, yet elegant, HTTP library.
- **Soupsieve (2.3.2.post1)** - Soup Sieve is a CSS selector library designed to be used with Beautiful Soup 4. It aims to provide selecting, matching, and filtering using modern CSS selectors. Soup Sieve currently provides selectors from the CSS level 1 specifications up through the latest CSS level 4 drafts and beyond (though some are

not yet implemented). Soup Sieve was written with the intent to replace Beautiful Soup's built-in select feature, and as of Beautiful Soup version 4.7.0, it now is: confetti ball. Soup Sieve can also be imported in order to use its API directly for more controlled, specialized parsing.

- **SQLAlchemy (1.4.36)** - SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL. SQLAlchemy provides a full suite of well-known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language.
- **Starlette (0.17.1)** - Starlette is a lightweight ASGI framework/toolkit, which is ideal for building async web services in Python.
- **typing_extensions (4.2.0)** - Enable experimentation with new type system PEPs before they are accepted and added to the typing module.
- **urllib3 (1.26.9)** - urllib3 is a powerful, user-friendly HTTP client for Python. Much of the Python ecosystem already uses urllib3 and you should too. urllib3 brings many critical features that are missing from the Python standard libraries:
    - Thread safety.
    - Connection pooling.
    - Client-side SSL/TLS verification.
    - File uploads with multipart encoding.
    - Helpers for retrying requests and dealing with HTTP redirects.
    - Support for gzip, deflate, and brotli encoding.
    - Proxy support for HTTP and SOCKS.
    - 100% test coverage.
- **Uvicorn (0.17.6)** - Uvicorn is an ASGI web server implementation for Python. Until recently Python has lacked a minimal low-level server/application interface for async frameworks. The ASGI specification fills this gap, and means we're now able to start building a common set of tooling usable across all async frameworks.
- **Websockets (10.3)** - websockets is a library for building WebSocket servers and clients in Python with a focus on correctness, simplicity, robustness, and performance.

- **Werkzeug (2.1.1)** - Werkzeug is a comprehensive WSGI web application library. It began as a simple collection of various utilities for WSGI applications and has become one of the most advanced WSGI utility libraries.
- **Yagmail (0.15.277)** - yagmail is a GMAIL/SMTP client that aims to make it as simple as possible to send emails.
- **Zipp (3.8.0)** - A pathlib-compatible Zipfile object wrapper. Official backport of the standard library Path object.

CHAPTER 4

RESULTS AND DISCUSSION

4. Sample of FastAPI automatic documentation

FastAPI provides automatic documentation and among them, they are – redoc and swagger documentation. Automatic documentation makes documentation of API easier to understand. The framework was originally developed with the aim of support, so the quality of documentation and the ease of its creation are high level. The server part of the terminal web system, integration with many third-party applications on other platforms such as desktop, web and mobile. The availability of API documentation greatly simplifies the integration process, since provides all the necessary data set that you may need developer working with this API. Django and Flask generate documentation, swagger using third party libraries or extensions that are being developed third-party developers, because of this, errors may appear in the documentation. With FastAPI such errors are possible only through the fault of the developer himself.

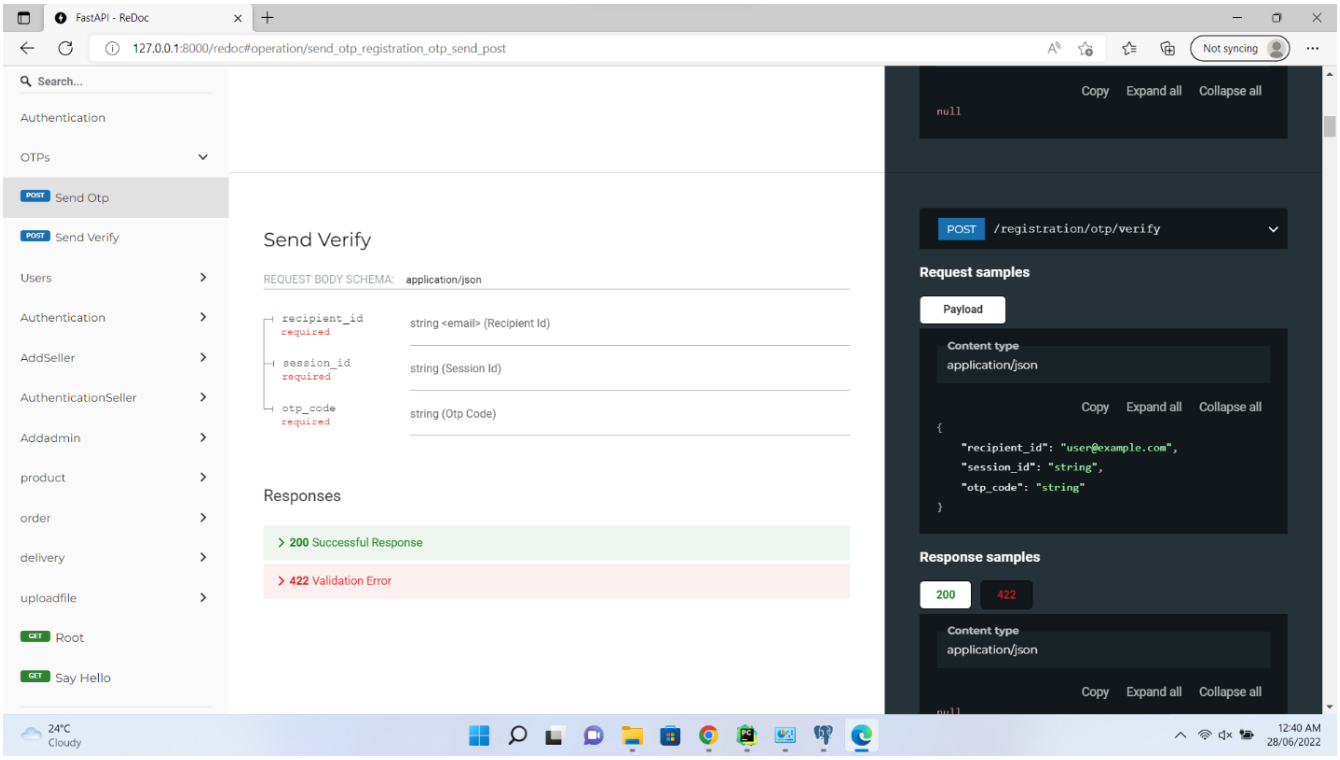


Fig 2.1. Sample on FastAPI Redoc automatic documentation

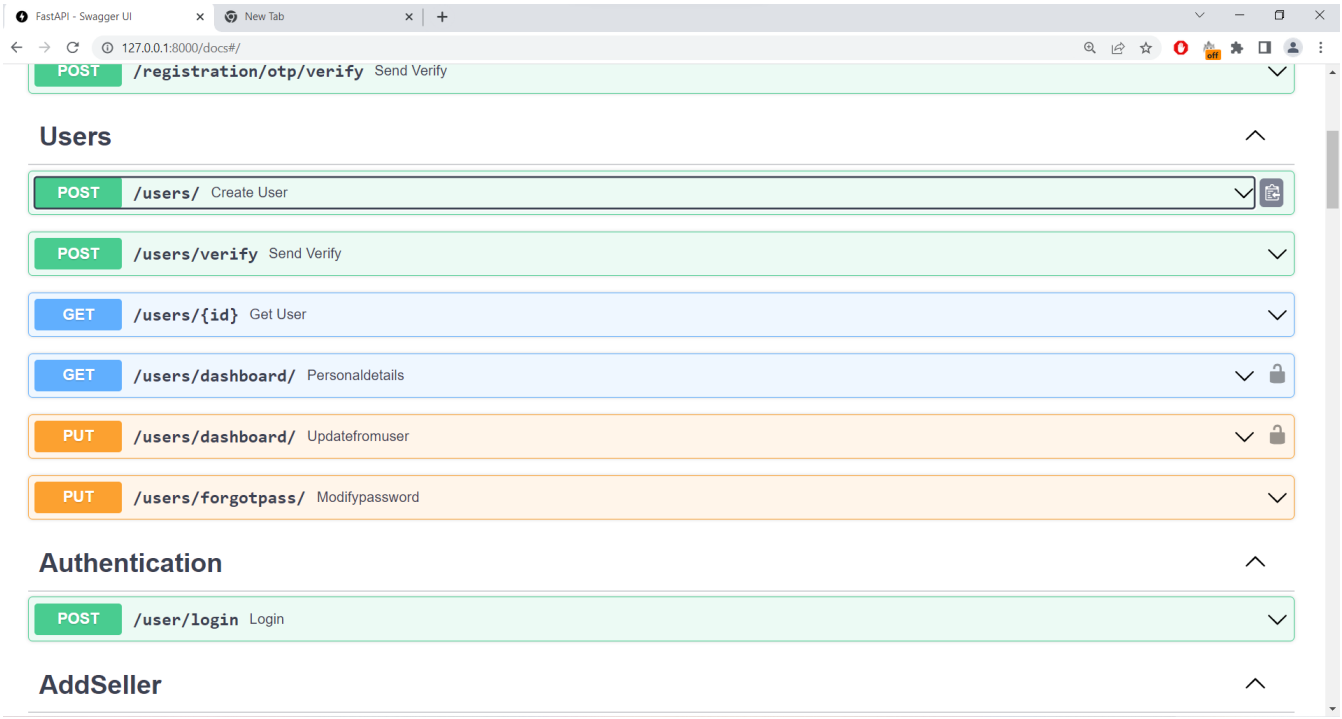


Fig 2.2. Sample on FastAPI swagger automatic documentation

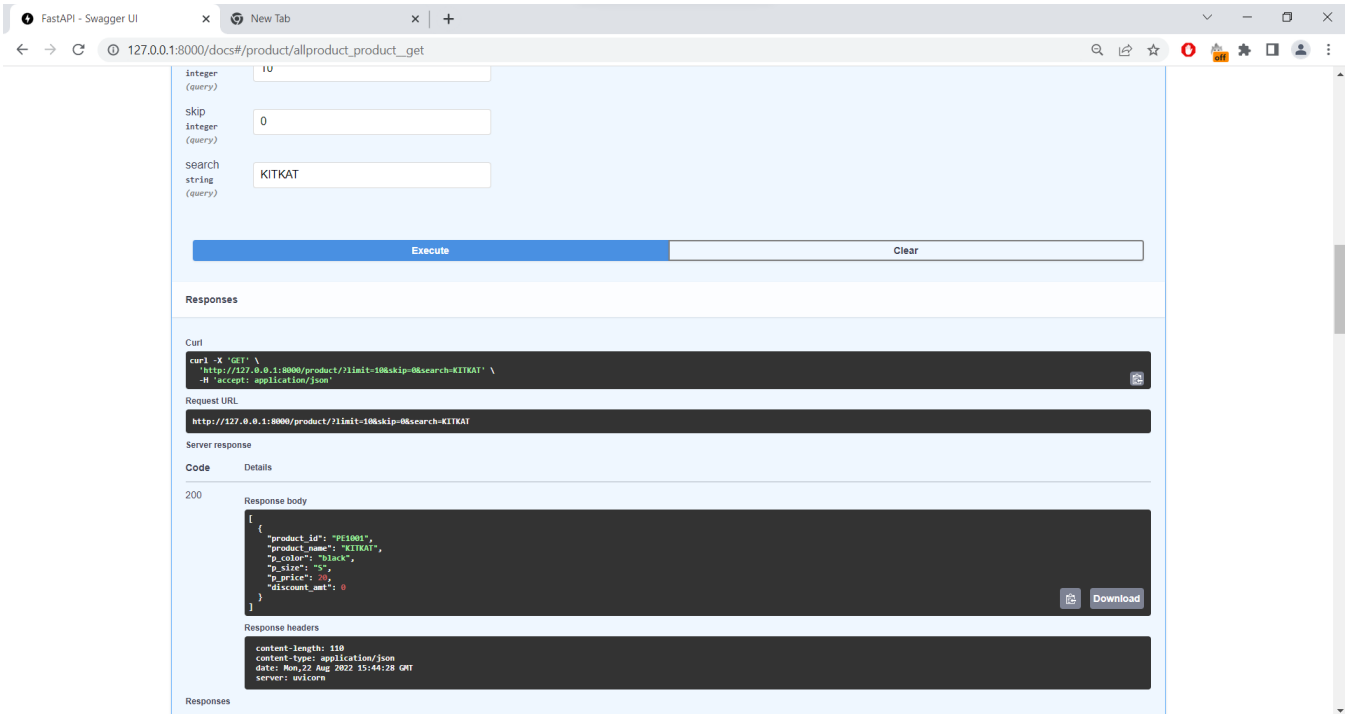


Fig 2.3. Sample on testing API using automatic documentation

Screenshots of Postman

4.1 Creating users

Given below figure shows the creation of users' login using the user's email, their full name and user's password. The input is given in JSON script in postman software to test the API and the OTP code and session-id.

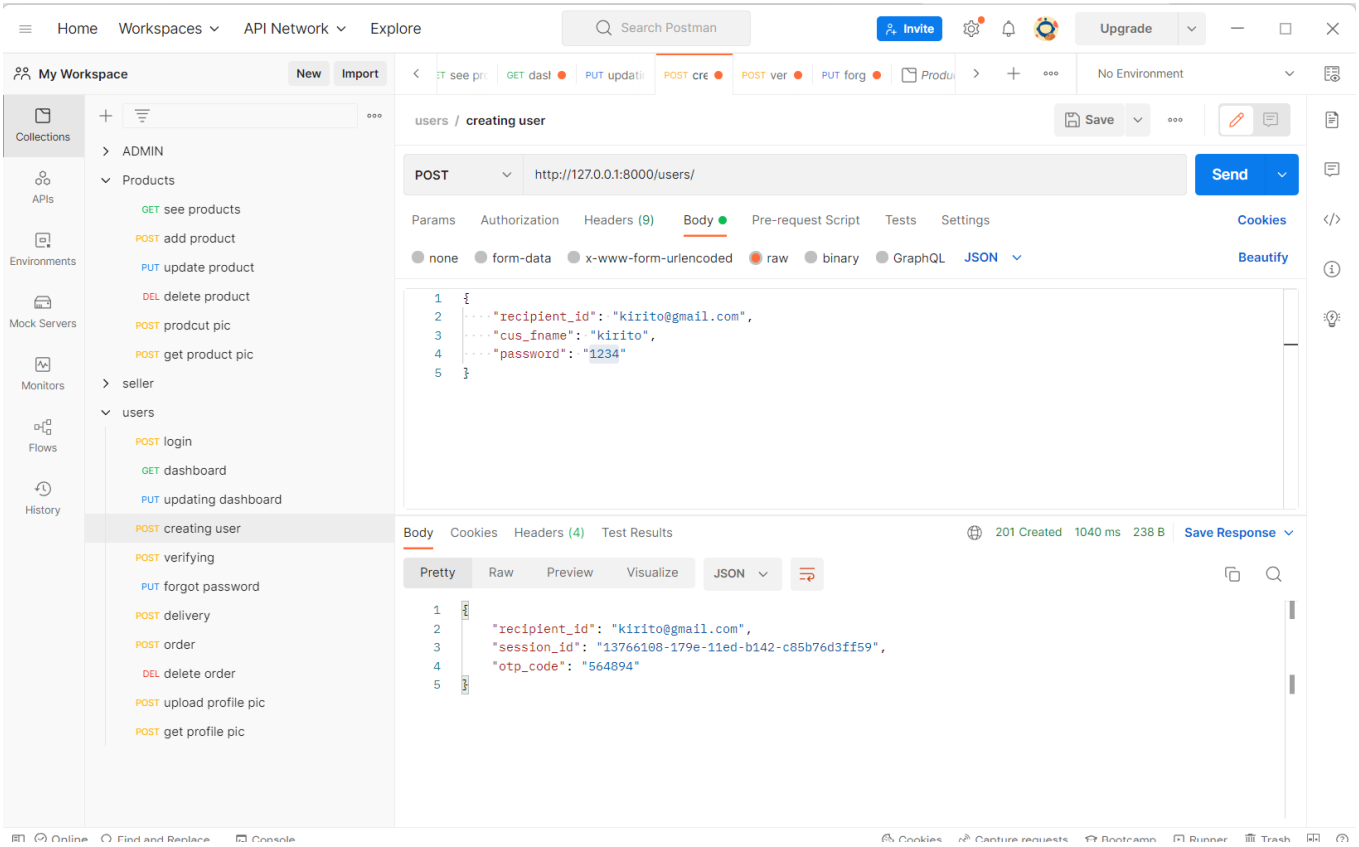


Fig 3.1. Creating users

4.2 Verifying the user

Given below figure shows the verification process of the user. After the OTP is received by the user through email and the session-id and OTP. The verification status and success message are sent through JSON.

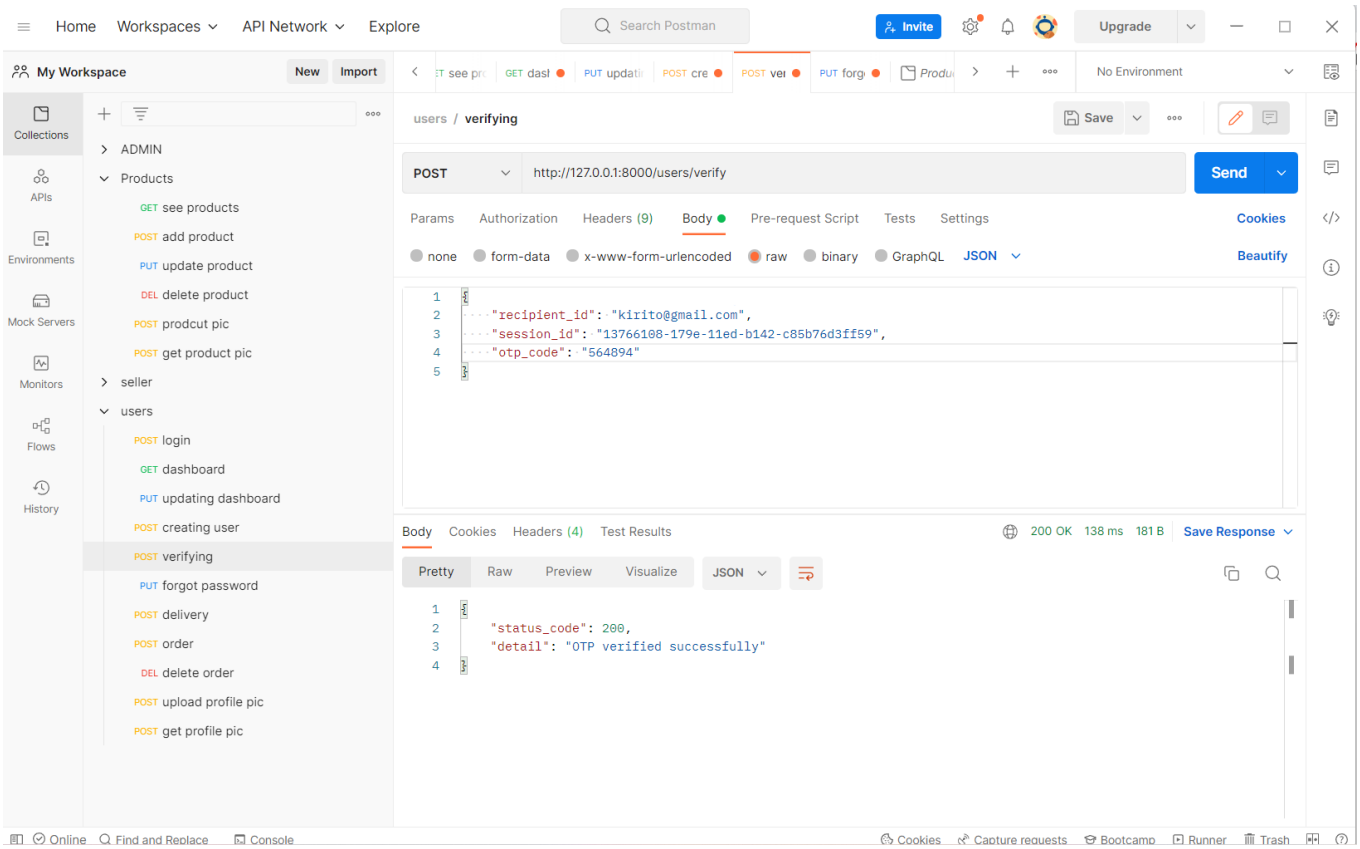


Fig 3.2. Verifying users

4.3 User login

Given below figures shows user login. The user email and password can be placed in form data to login. After the login is successful the access token and user id are sent from API to the front-end.

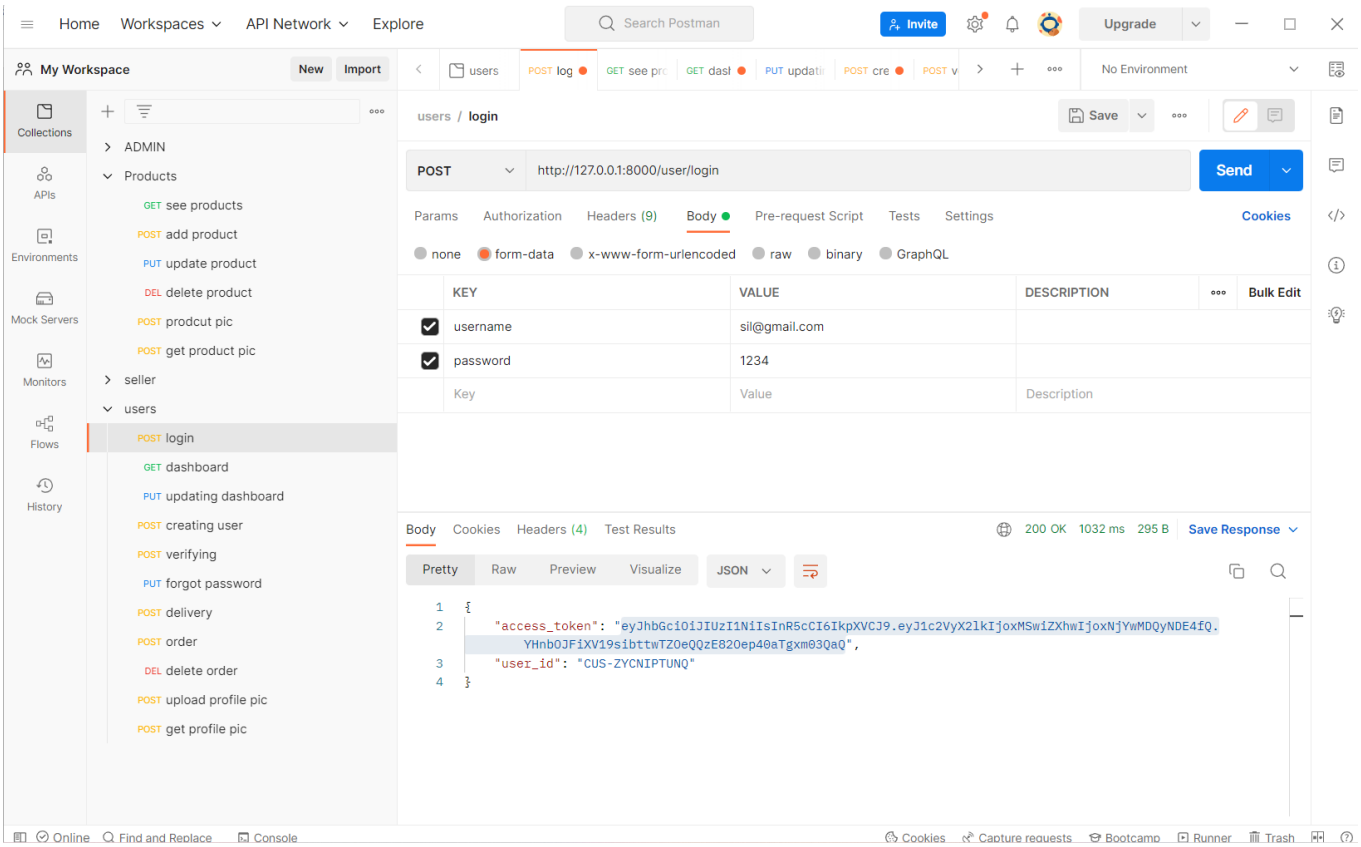


Fig 3.3. User Login

4.4 Updating Profile

Given below figure shows updating of user's profile/dashboard. The front-end provides JSON script to be executed by API and after the update in database the new updated details are then sent to the front-end with success response.

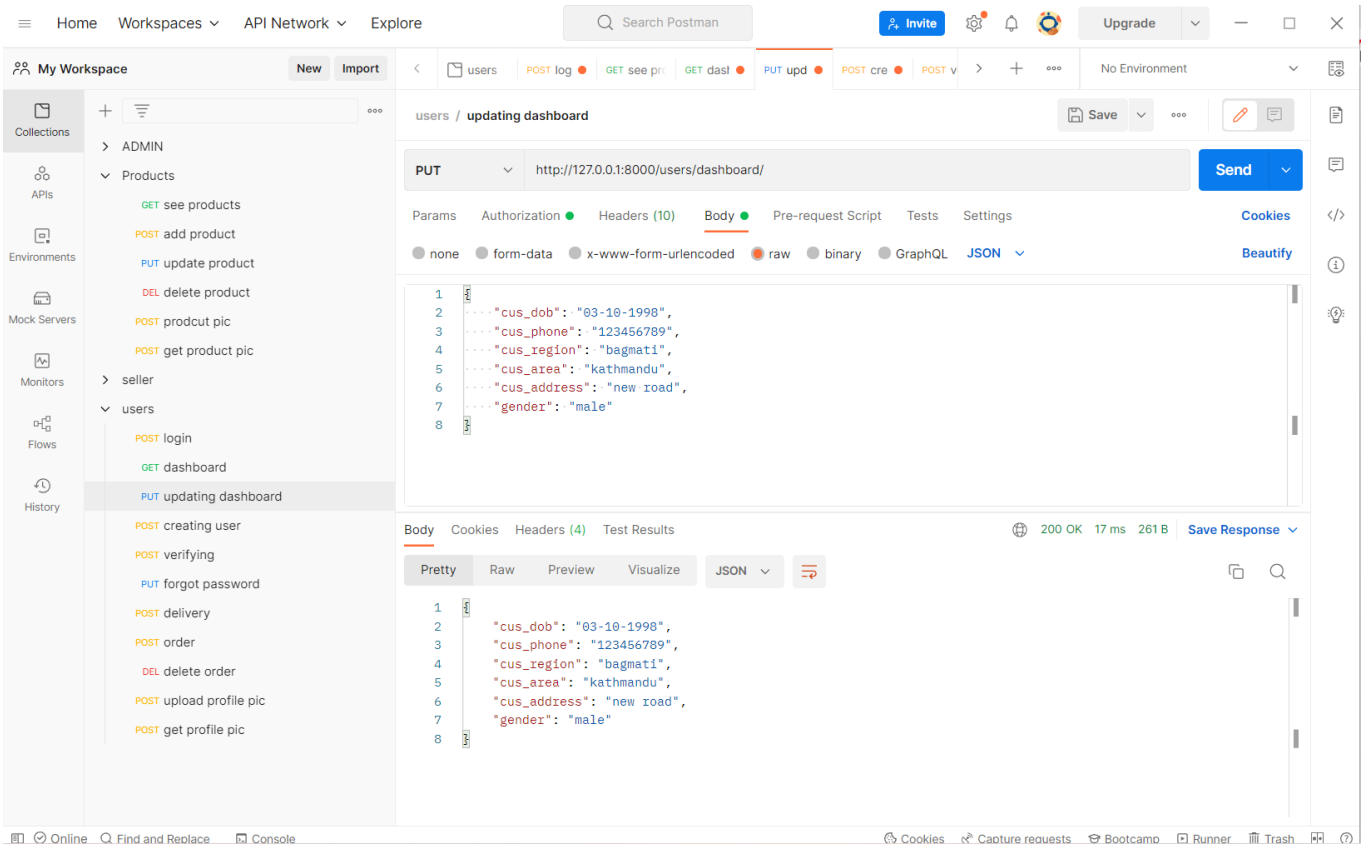


Fig 3.4. Updating Profile

4.5 Dashboard of user

Given below figure shows the dashboard of the user with the access token.

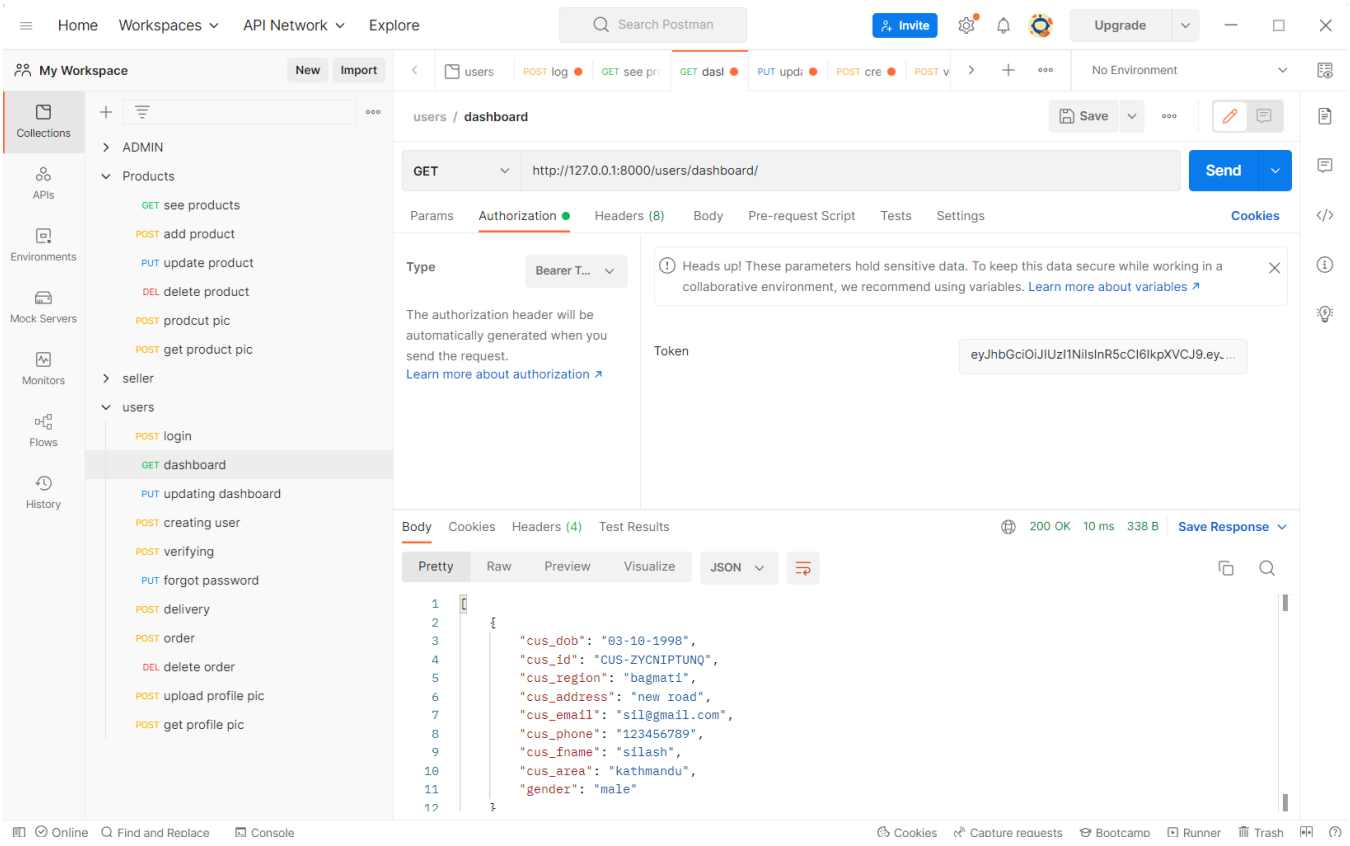


Fig 3.5. Dashboard of user

4.6 Uploading user's profile picture

Given below figure shows uploading of user's profile using user's access token and how the picture is uploaded through postman.

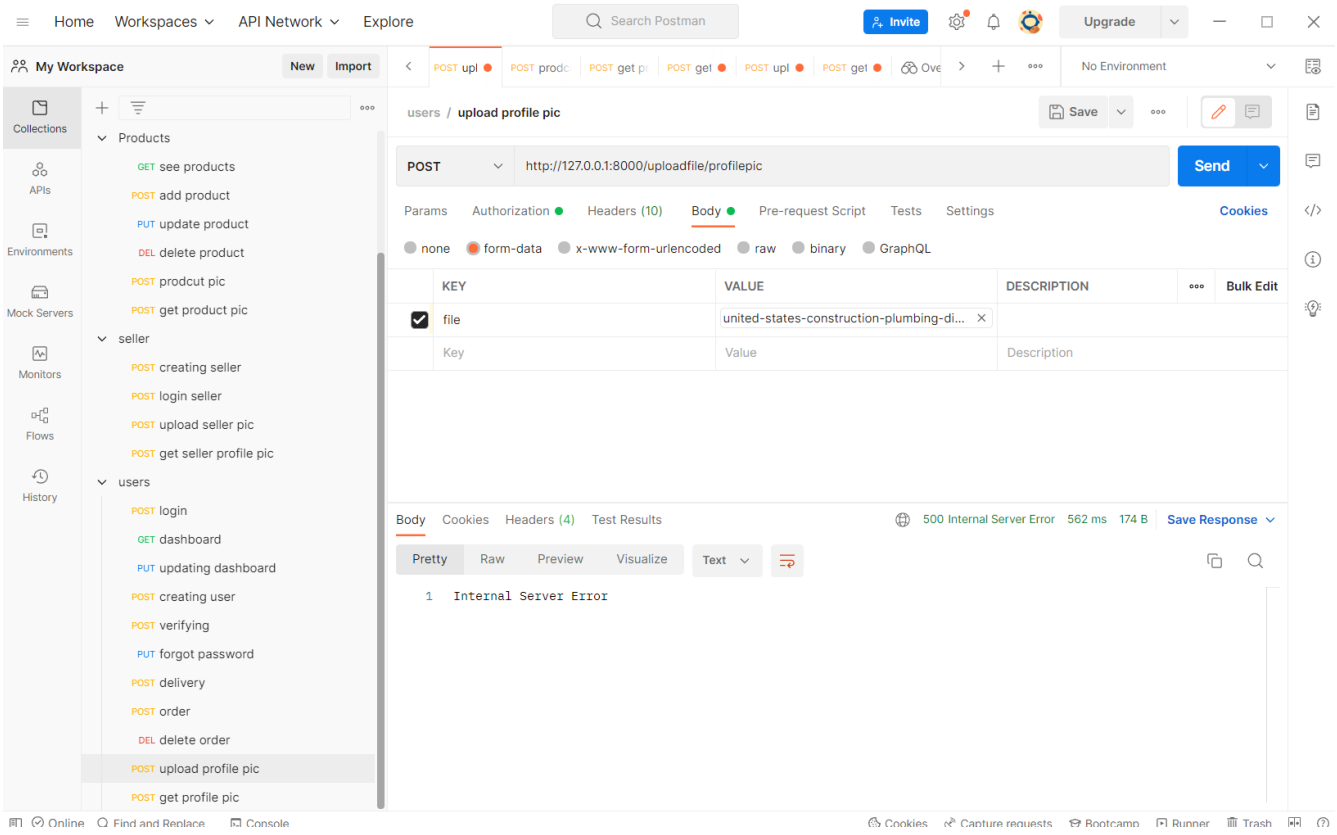


Fig 3.6. Uploading user profile picture

4.7 Retrieving user's profile picture

Given below figure shows retrieving of user's profile picture after the access token is given in postman. The directory of the image is given as JSON response to the front-end.

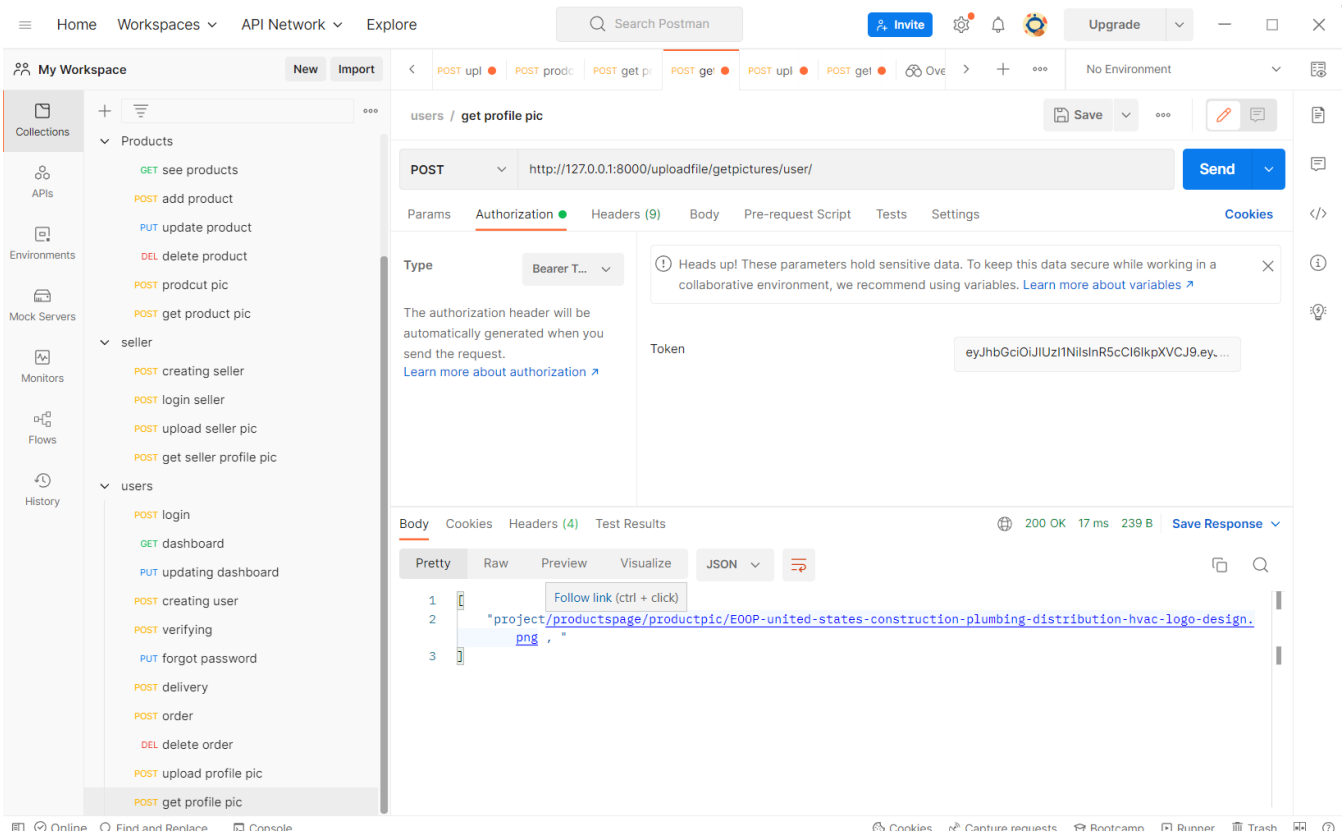


Fig 3.7. Retrieving user's profile pic

4.8 Adding product picture

Given below figure shows adding of multiple product picture using seller's credentials.

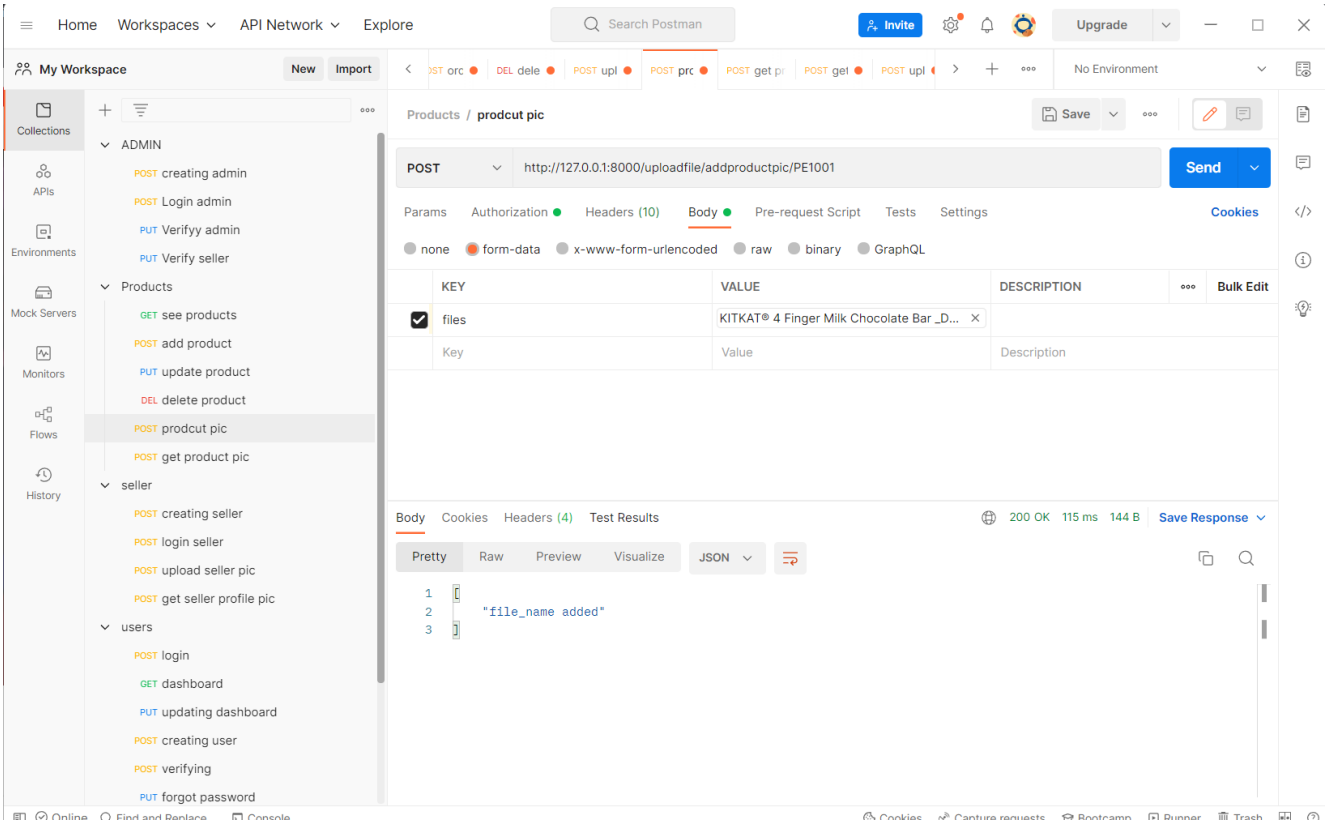


Fig 3.8. Adding product picture using seller

4.9 Retrieving product picture

Given below figure shows the retrieval of multiple product picture from the front-end through product-id. JSON response is sent to front-end from API.

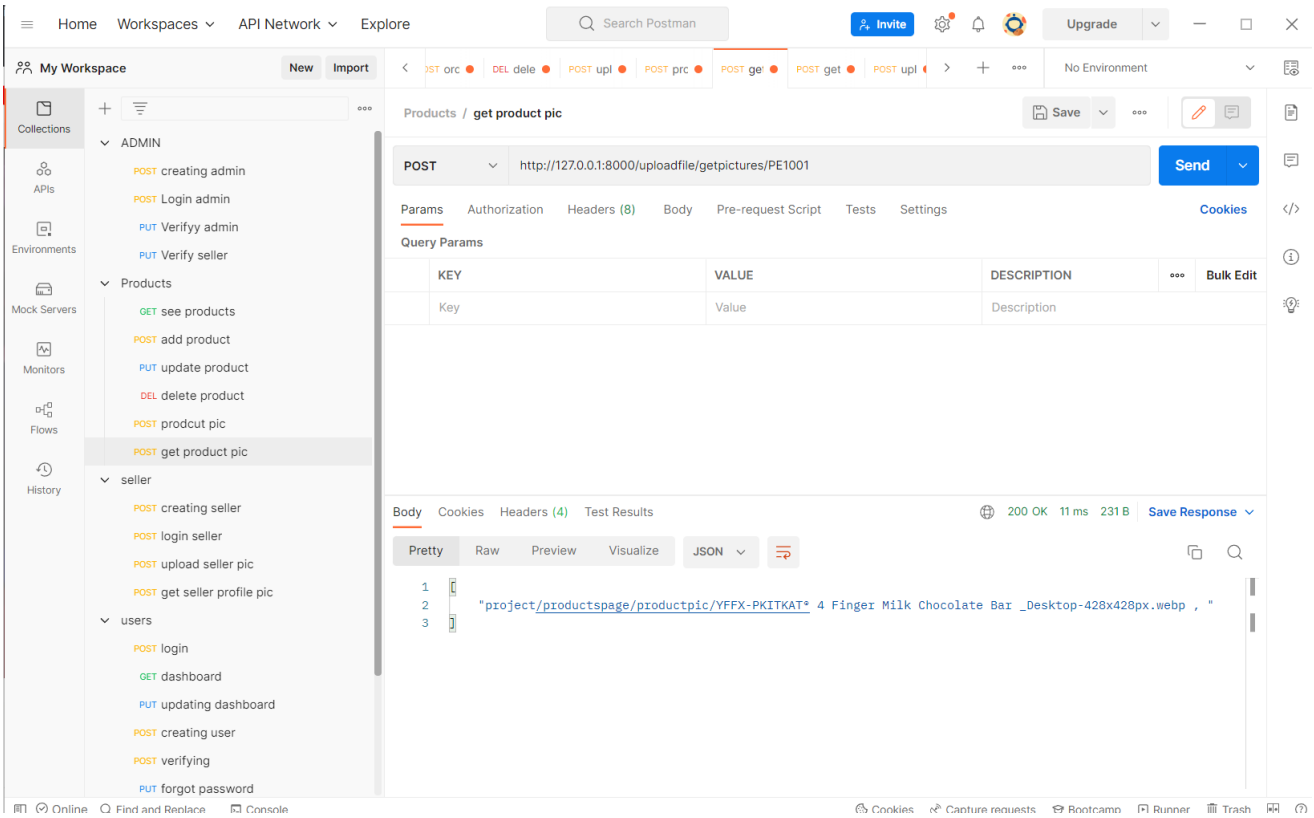


Fig 3.9. Retrieving product picture

4.10 Showing list of products

Given below figure shows the list of available products through the link http://127.0.0.1:8000/product and available products are sent through API in JSON format.

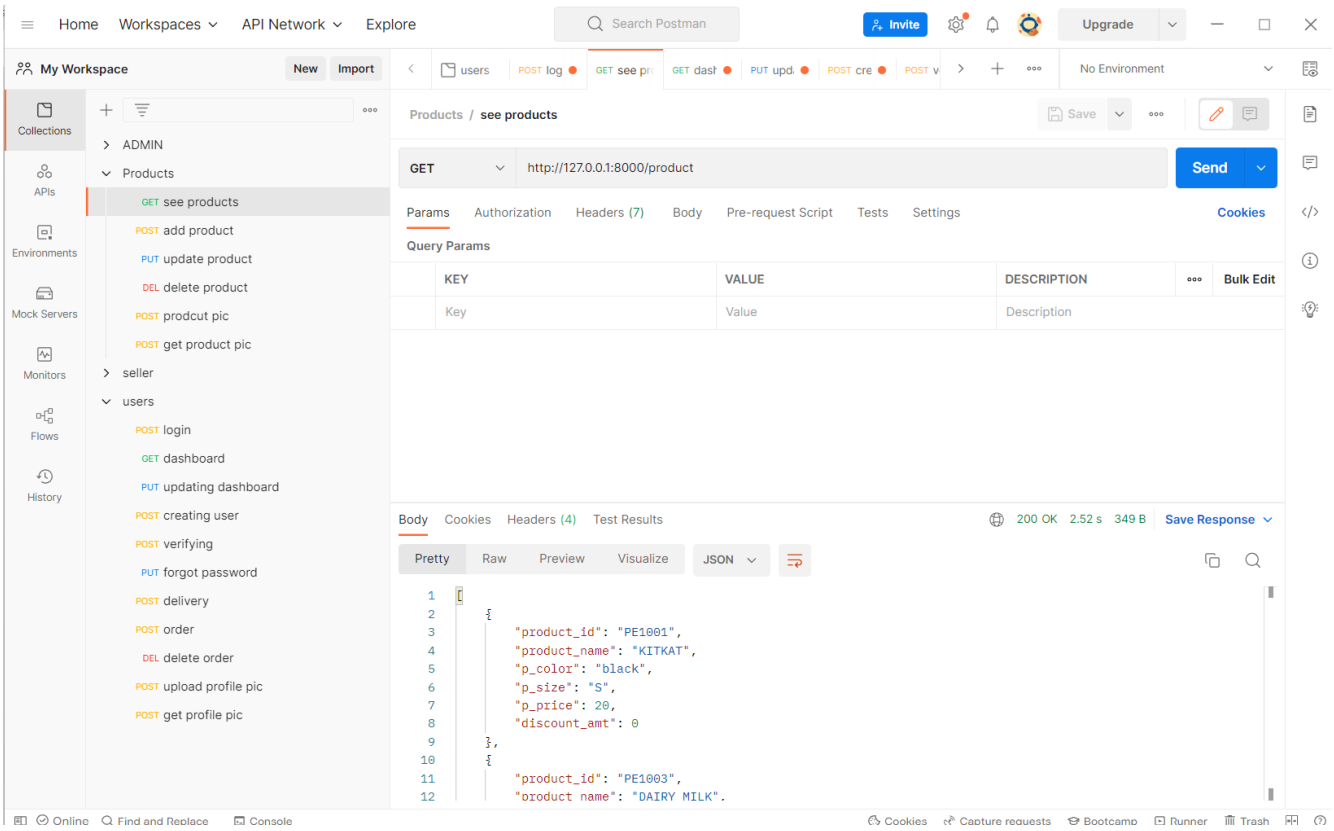


Fig 3.10. All products

4.11 Ordering through user's credential

Given below figure shows how the multiple order is sent from front-end after the customer orders. After the order is confirmed, the order id and the total price is sent through JSON response.

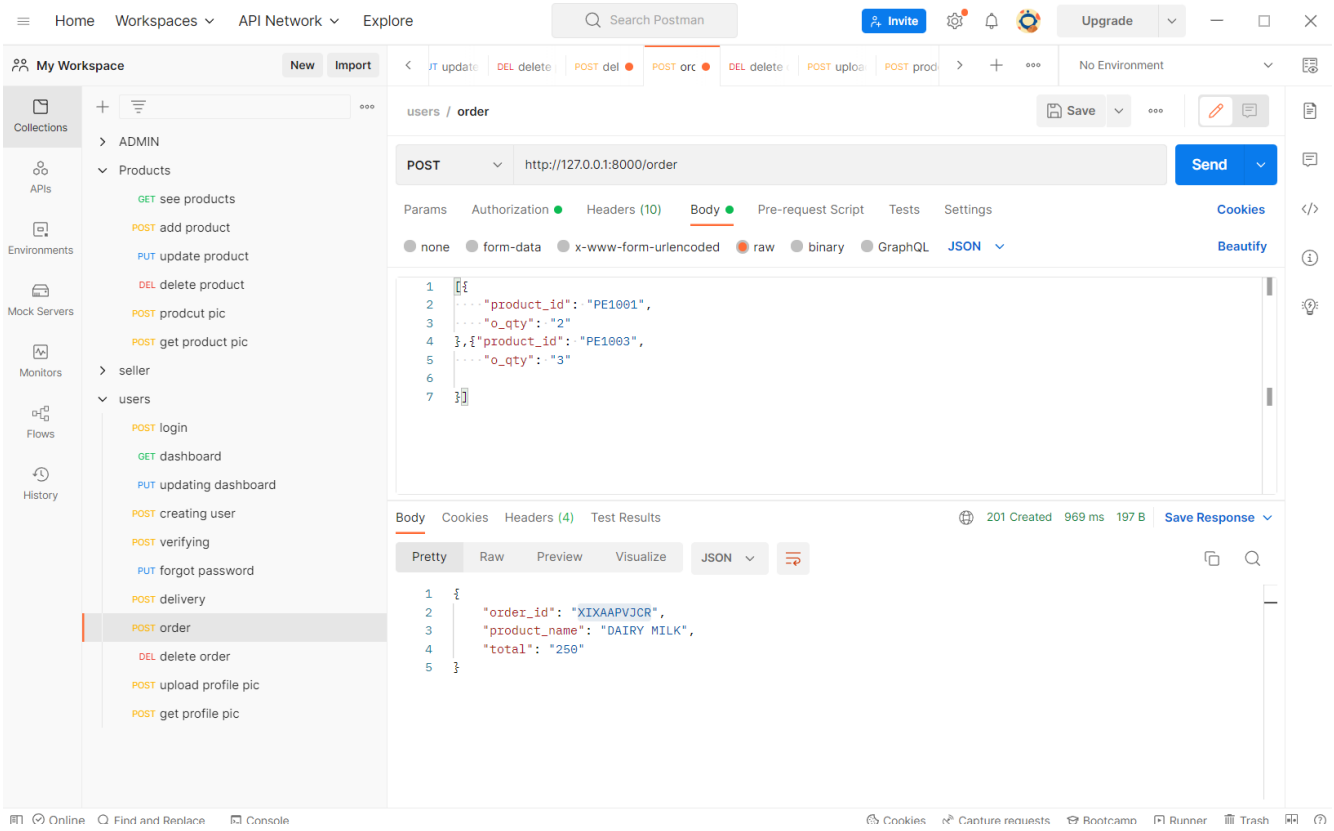


Fig 3.11. Order view of the customer

4.12 Order confirmation

Given below figure shows after the delivery location of the customer is given sends the data to API using JSON script and the API will respond with JSON response with the customer details, total amount and order id.

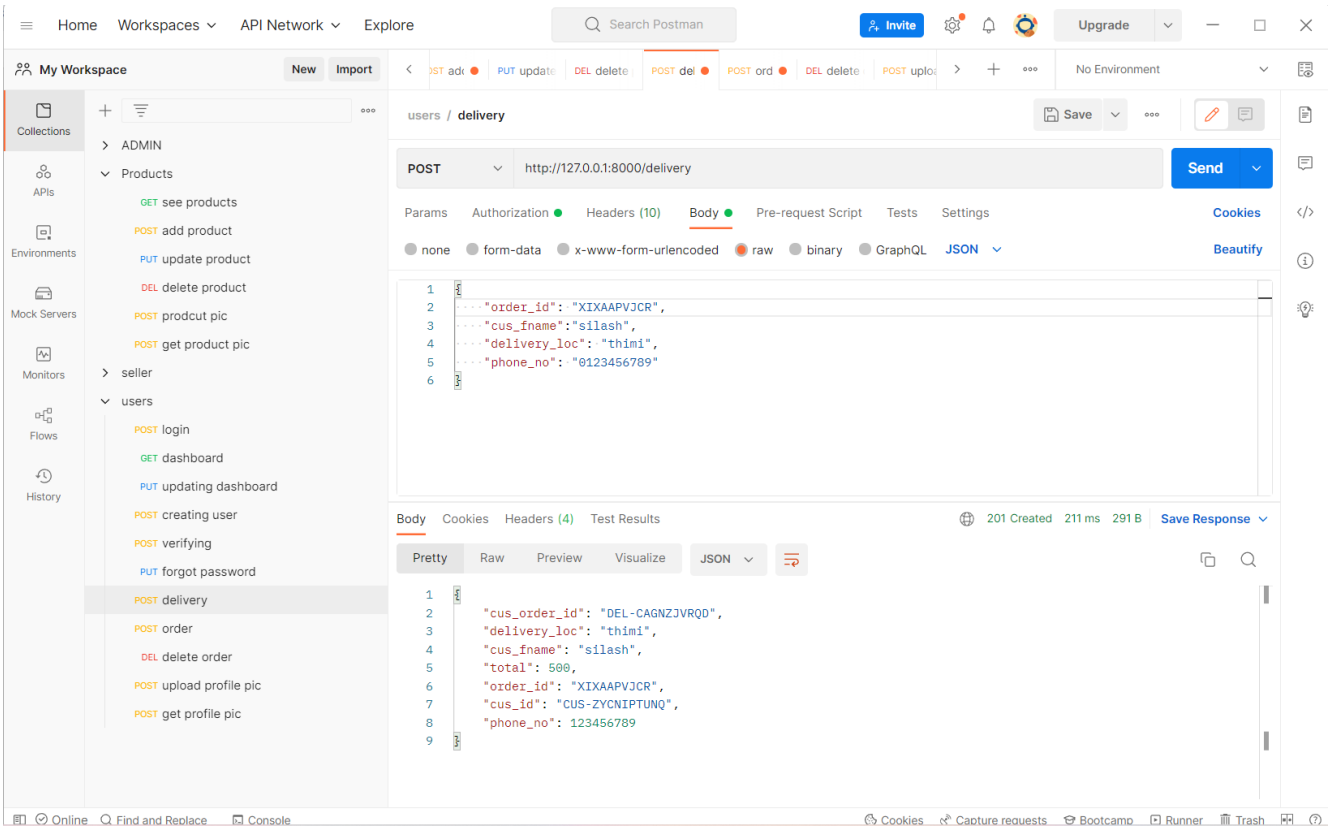


Fig 3.12. After order confirmation, ready for delivery

4.13 Deleting order

Given below figure shows the deletion of order through order-id.

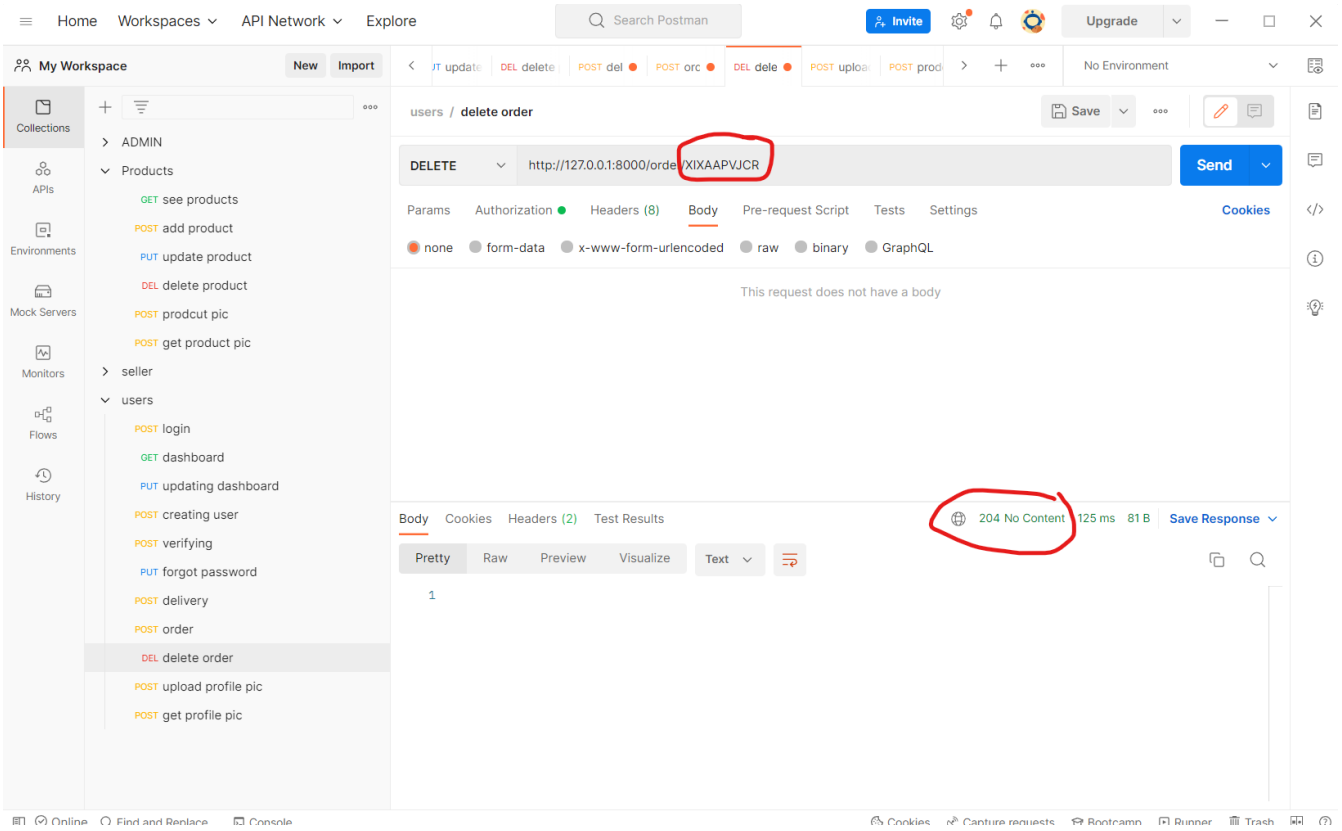


Fig 3.13. Deleting Order

4.14 Creating seller

Given below figure shows creation of user type seller who can add, update and delete product. While creating seller user, they have to request to admin user for permission for them to be added as seller.

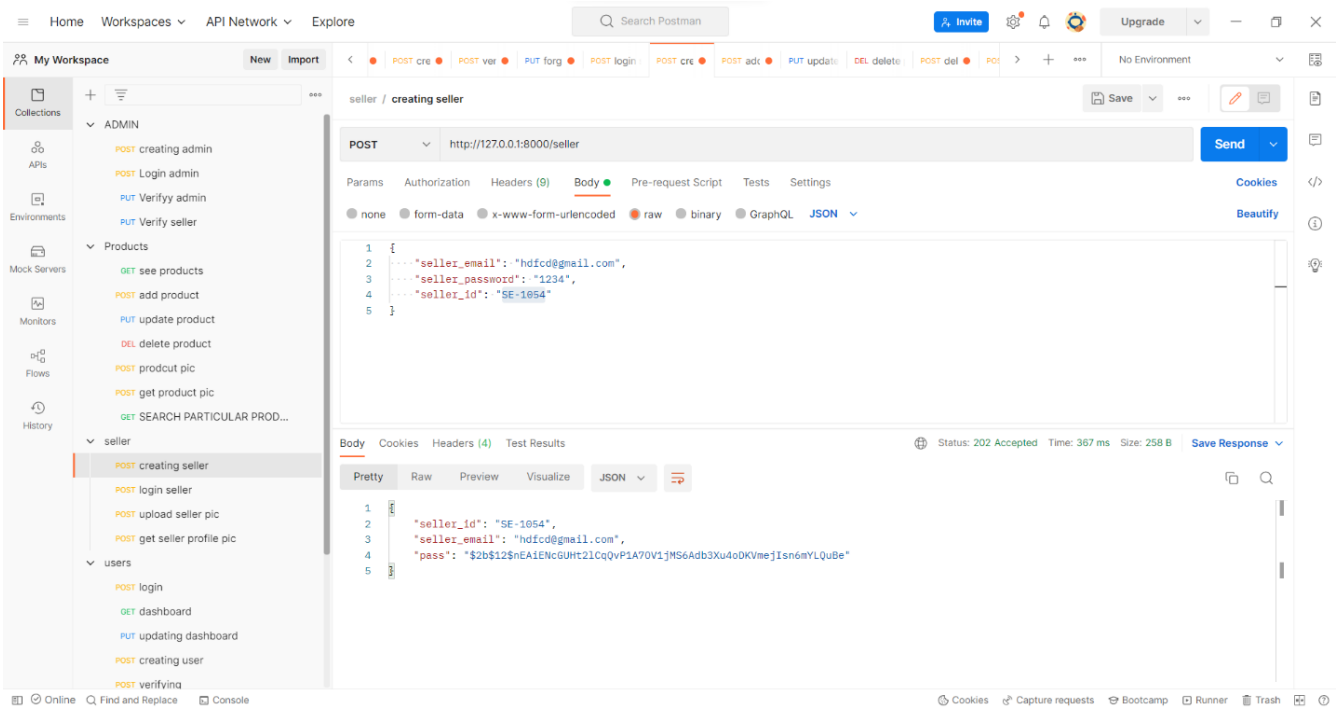


Fig 3.14. Creating Seller

4.15 Verifying seller

Given below figure shows the verification done through admin user credential. The seller id is inserted and the value of verification is changed to ‘0’, by default while creating seller the verification number is ‘1’.

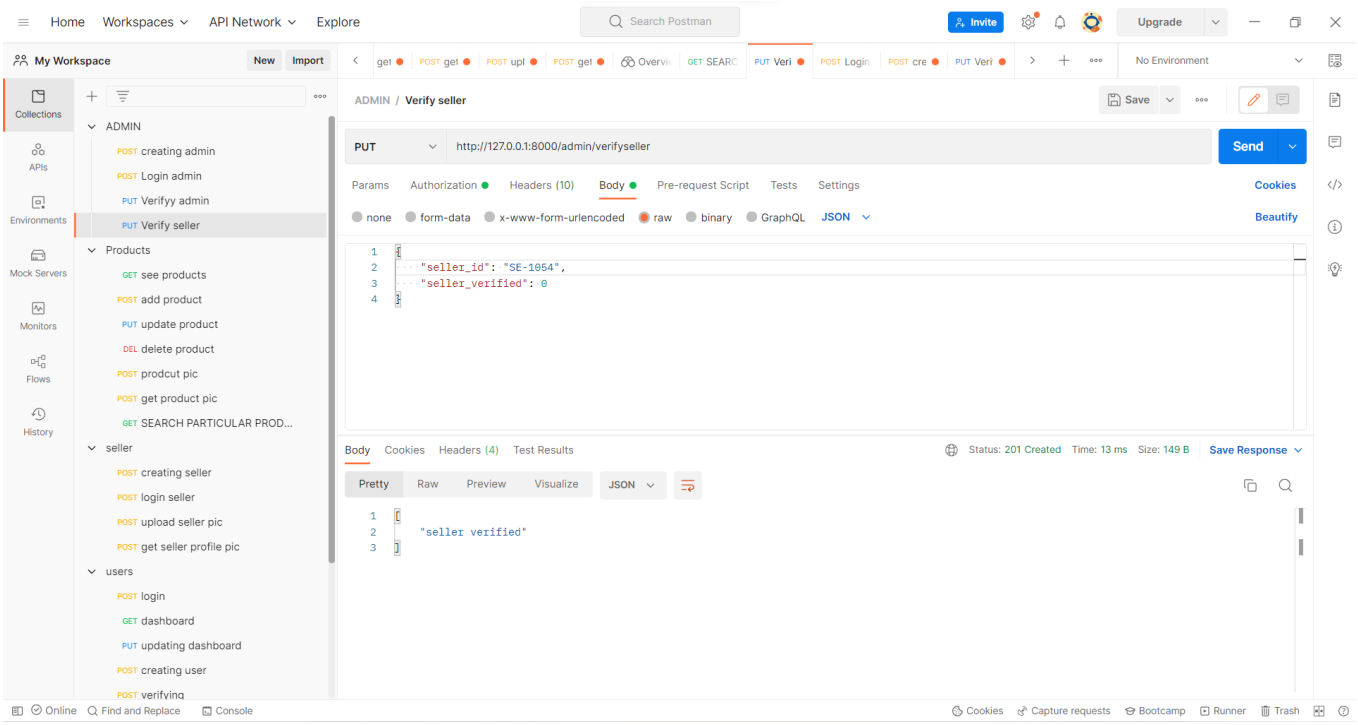


Fig 3.15. Verifying Seller

4.16 Creating admin

Given below figure shows the creation of user type admin who has permission and rights of all the modifications. While creating admin user, they have to request to admin user for permission for them to be added as admin.

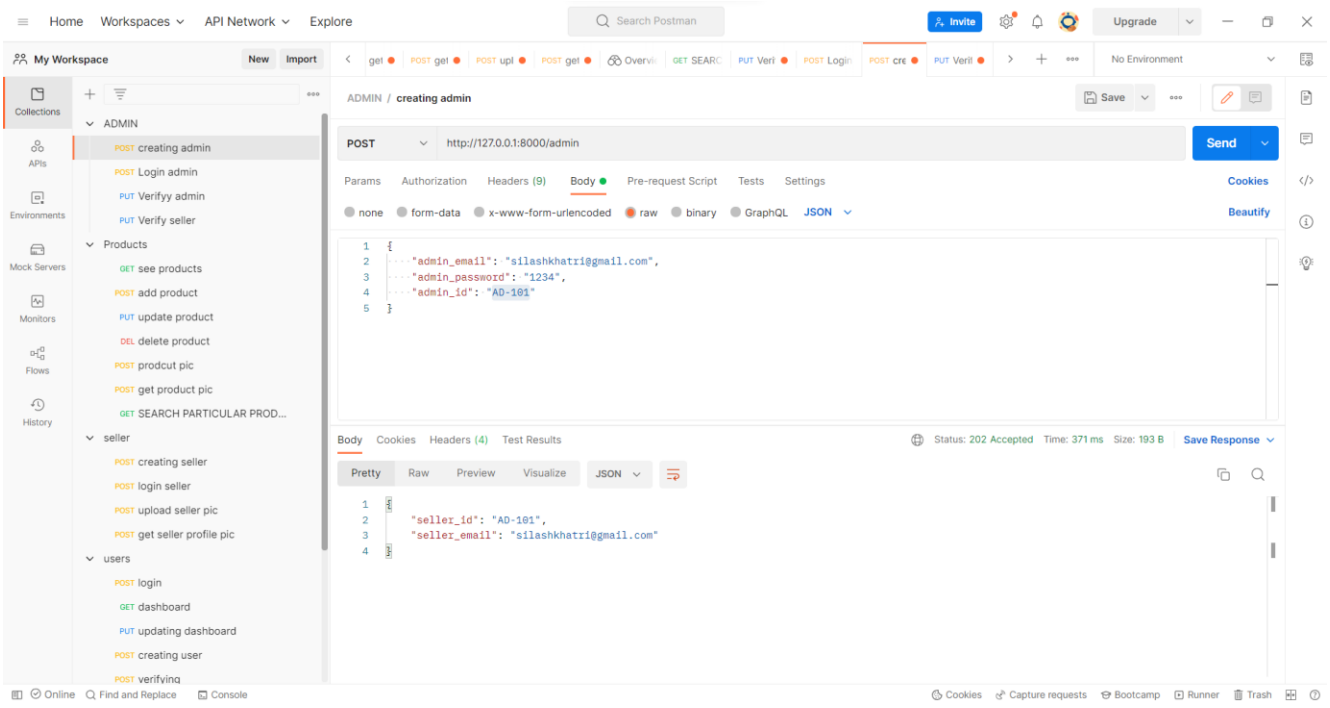
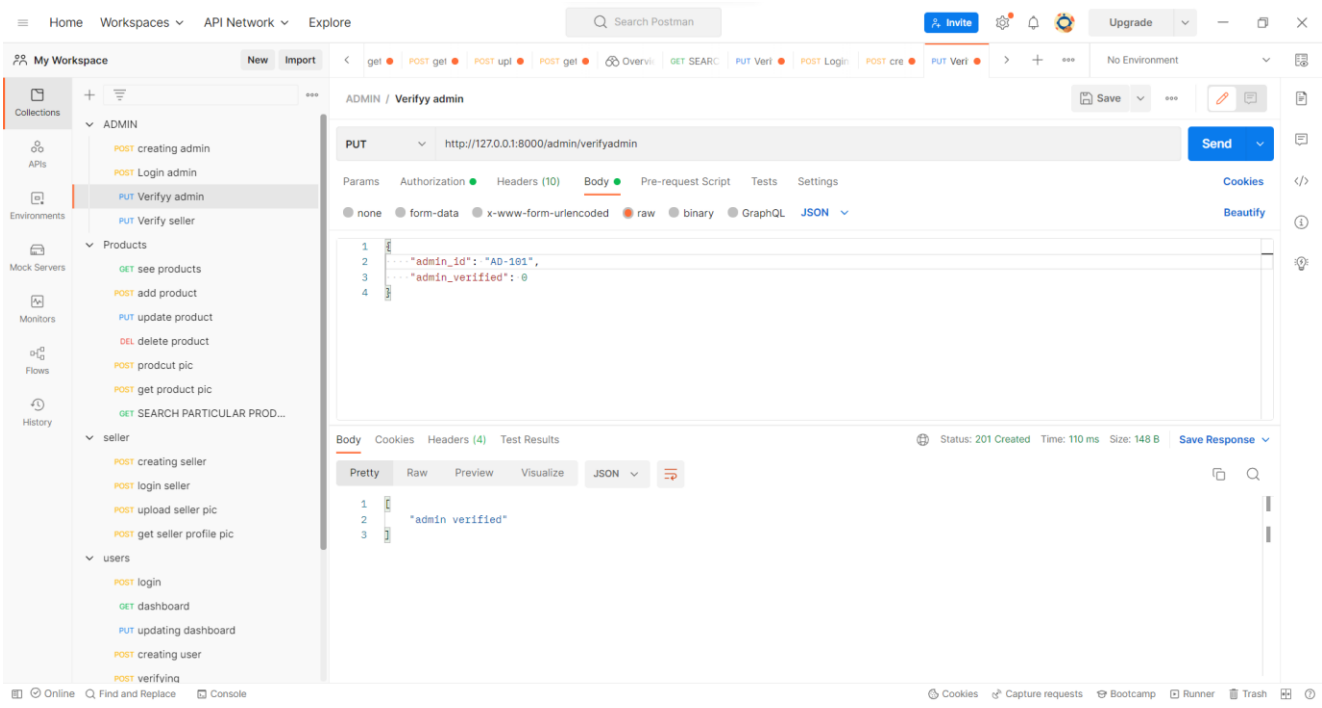


Fig 3.16. Creating Admin

4.17 Verifying admin

Given below figure shows the verification done through admin user credential. The admin id is inserted and the value of verification is changed to ‘0’, by default while creating seller the verification number is ‘1’.



CHAPTER 5
CONCLUSION

API for ecommerce help developers transfer information from one software to another and then use this data in a single user interface. APIs work as an extensible platform that integrates with various solutions so companies can take advantage of their existing functionality without building features from scratch. FastAPI has proven to be an excellent framework containing many modern technologies and development trends. Supplemented with own unique mechanisms, as well as showing excellent performance systems written on it, it is a very powerful and promising tool development, which, quite possibly, will soon become more popular than Django and Flask. A single API can be accessed from various front-ends, whether that be by mobile applications or the website.

REFERENCES

1 https://www.slant.co/versus/125/34241/~php_vs_fastapi

2 https://fastapi.tiangolo.com/

3 https://quintagroup.com/services/python/fastapi

4 https://pypi.org/project/anyio/

5 https://pypi.org/project/asgiref/

6 https://magicstack.github.io/asyncpg/current/

7 https://pypi.org/project/bcrypt/

8 https://pypi.org/project/cachetools/

9 https://pypi.org/project/certifi/

10 https://pypi.org/project/charset-normalizer/

11 https://pypi.org/project/click/

12 https://pypi.org/project/colorama/

13 https://pypi.org/project/cryptography/

14 https://pypi.org/project/cssselect/

15 https://pypi.org/project/databases/

16 https://pypi.org/project/dnspython/

17 https://pypi.org/project/ecdsa/

18 https://pypi.org/project/email-validator/

19 https://pypi.org/project/fastapi/

20 https://pypi.org/project/Flask/

21 https://pypi.org/project/greenlet/

22 https://pypi.org/project/Flask-SQLAlchemy/

23 https://pypi.org/project/gunicorn/

24 https://pypi.org/project/httptools/

25 https://pypi.org/project/idna/

26 https://pypi.org/project/importlib-metadata/

27 https://pypi.org/project/Jinja2/

28 https://pypi.org/project/lxml/

29 https://pypi.org/project/MarkupSafe/

30 https://pypi.org/project/numpy/
31 https://pypi.org/project/passlib/
32 https://pypi.org/project/premailer/
33 https://pypi.org/project/psycopg2-binary/
34 https://pypi.org/project/pydantic/
35 https://pypi.org/project/pytz/
36 https://pypi.org/project/pyparsing/
37 https://pypi.org/project/PyYAML/
38 https://pypi.org/project/requests/
39 https://pypi.org/project/sniffio/
40 https://pypi.org/project/soupsieve/
41 https://pypi.org/project/SQLAlchemy/
42 https://pypi.org/project/starlette/
43 https://pypi.org/project/typing-extensions/
44 https://pypi.org/project/urllib3/
45 https://pypi.org/project/typing-extensions/
46 https://pypi.org/project/websockets/
47 https://pypi.org/project/uvicorn/
48 https://pypi.org/project/Werkzeug/
49 https://pypi.org/project/zipp/
50 https://pypi.org/project/Werkzeug/
51 https://pypi.org/project/yagmail/