**Documentation for Flask Application**

**Introduction**

This documentation provides an overview and explanation of the key components and functionality of the Flask application. The application includes features for user authentication, image upload to Azure Blob Storage, and protected routes accessible only with a valid token.

**Table of Contents**

1. Dependencies
2. Application Configuration
3. Azure Blob Storage Configuration
4. Database Configuration and Connection Pooling
5. Token Required Decorator
6. Endpoints
   * User Signup (/user/signup)
   * User Signin (/user/signin)
   * Image Upload (/upload\_image)

**1. Dependencies**

* **Flask:** Flask is a lightweight web framework for Python. It provides tools, libraries, and patterns to create web applications. In this code, Flask is used to define routes, handle HTTP requests, and manage the application.
* **Werkzeug:** Werkzeug is a utility library for WSGI (Web Server Gateway Interface) applications in Python. In this code, Werkzeug is used for security functions (generate\_password\_hash and check\_password\_hash) and for handling file uploads (secure\_filename).
* **Azure Storage Blob:** Azure Storage Blob is a library for interacting with Azure Blob Storage. It is used to upload images to Azure Blob Storage in the upload\_to\_azure\_blob function.
* **MySQL Connector:** MySQL Connector is a MySQL database driver for Python. It is used to connect to a MySQL database, execute queries, and manage database connections in this code.
* **JWT (JSON Web Tokens):** JSON Web Tokens (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties. It is used for user authentication and authorization in this code. The jwt library is used to encode and decode JWTs.
* **Secrets:** The secrets module is used to generate cryptographically strong random numbers suitable for managing secrets such as authentication tokens. In this code, it is used to generate a random secret key for securing JWTs.
* **OS:** The os module provides a way of using operating system-dependent functionality. In this code, it is used to access environment variables or perform OS-level operations.
* **Functools:** The functools module provides higher-order functions and operations on callable objects. In this code, the wraps decorator from functools is used to preserve the original function's metadata when creating a decorator.

**2. Application Configuration**

* A Flask application is created, and a secret key is generated for securing sessions.
* The Flask application is configured with a secret key for JWT token encoding and decoding.

**3. Azure Blob Storage Configuration**

* Azure Storage connection string and container name are defined for uploading images to Azure Blob Storage.

**4. Database Configuration and Connection Pooling**

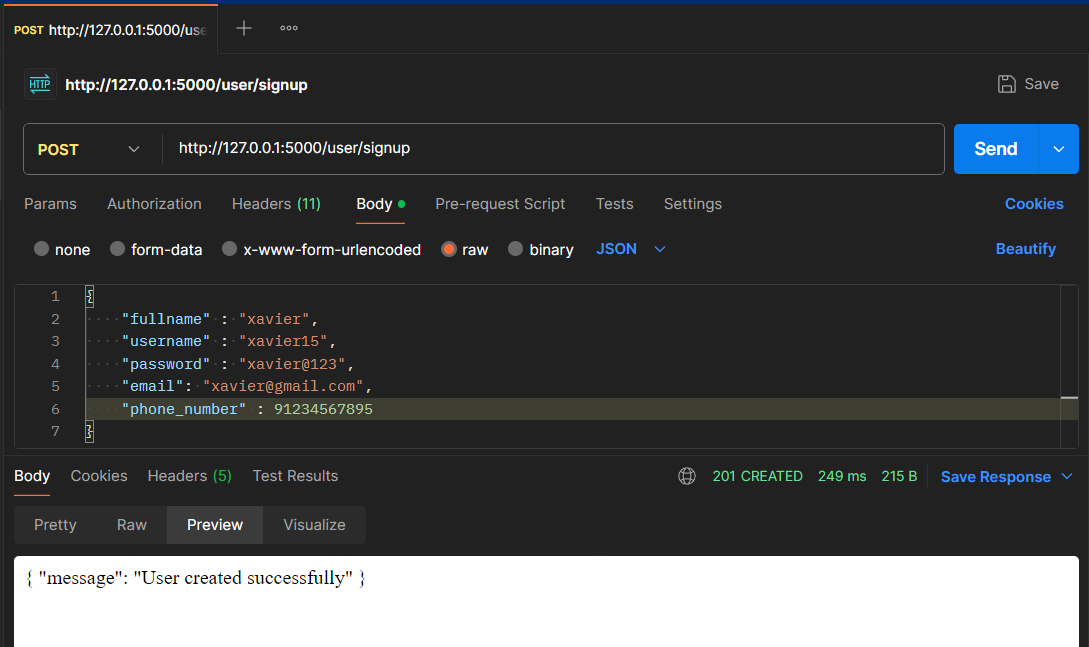
* MySQL database connection parameters (host, user, password, database) are configured.
* A connection pool is created using MySQL Connector's pooling feature.

**5. Token Required Decorator**

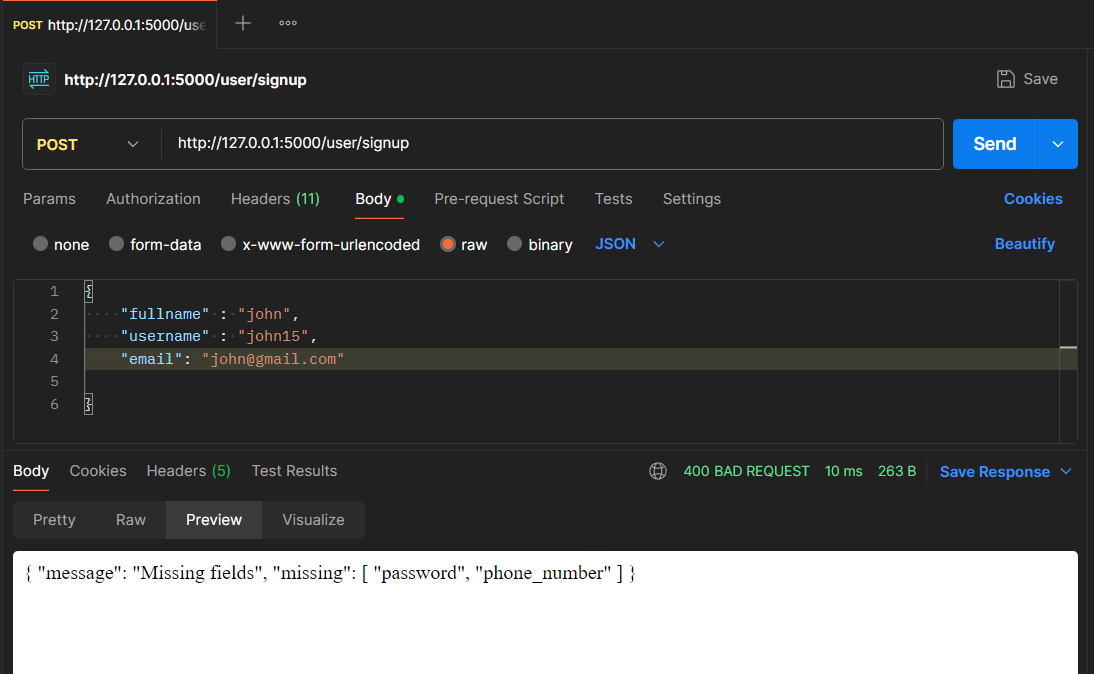
* The token\_required decorator checks for the presence of a valid JWT token in the request headers.
* If a valid token is present, the decorator decodes the token and passes the current user to the decorated function.
* Used to secure the /protected route.

**6. Endpoints**

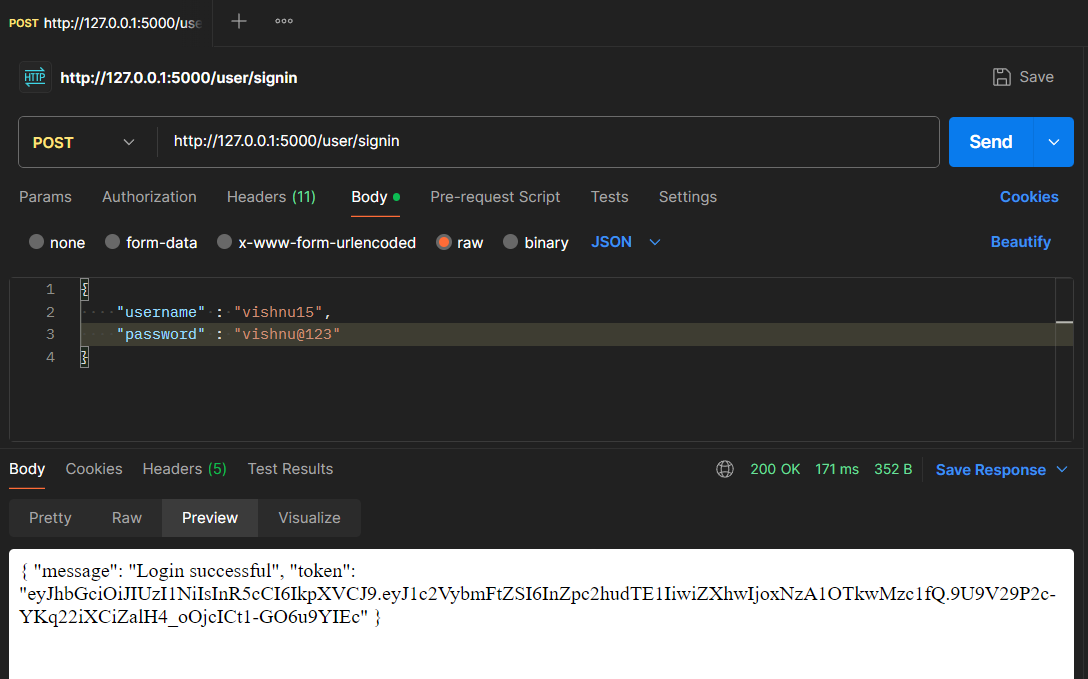
* **User Signup (/user/signup)**
  + **Method:** POST
  + Expects a JSON payload with the following fields:*fullname, username, password, email, phone\_number.*
  + Creates a new user in the database with hashed password.
  + Requires *application\_id* and *client\_id* headers for successful user creation.



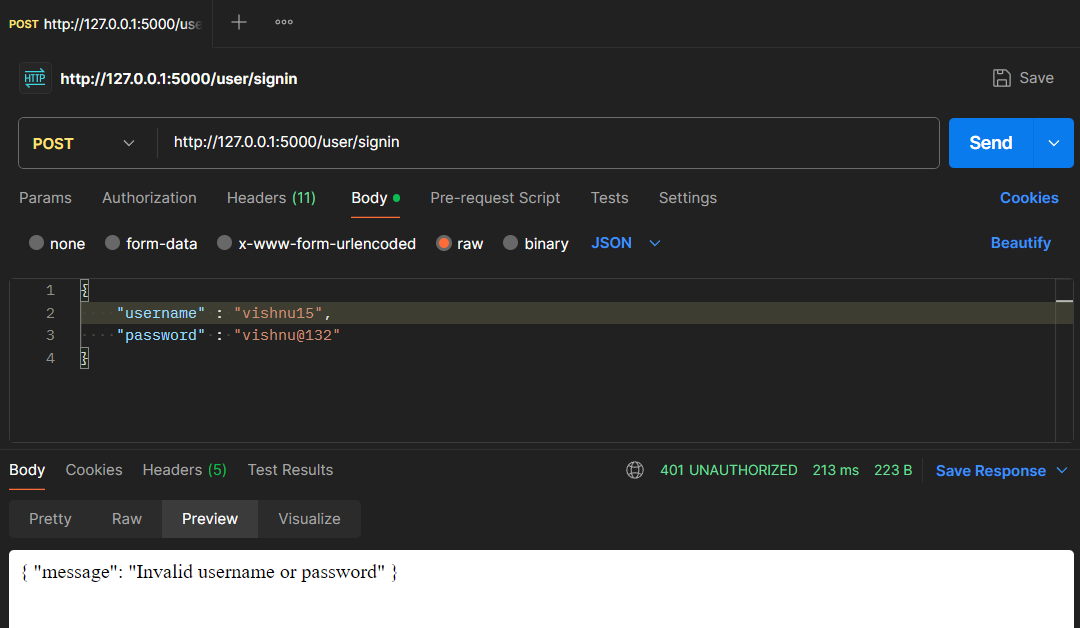
If there are any missing fields:



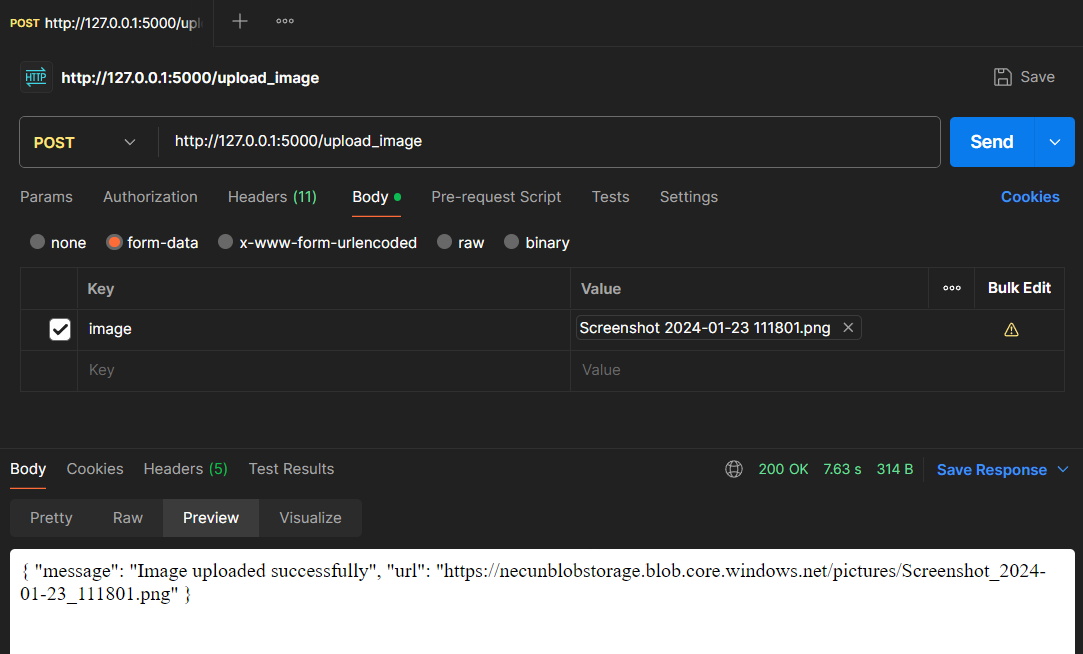
* **User Signin (/user/signin)**
  + **Method:** POST
  + Expects a JSON payload with *username* and *password.*
  + Validates user credentials against the database and returns a JWT token for successful authentication.



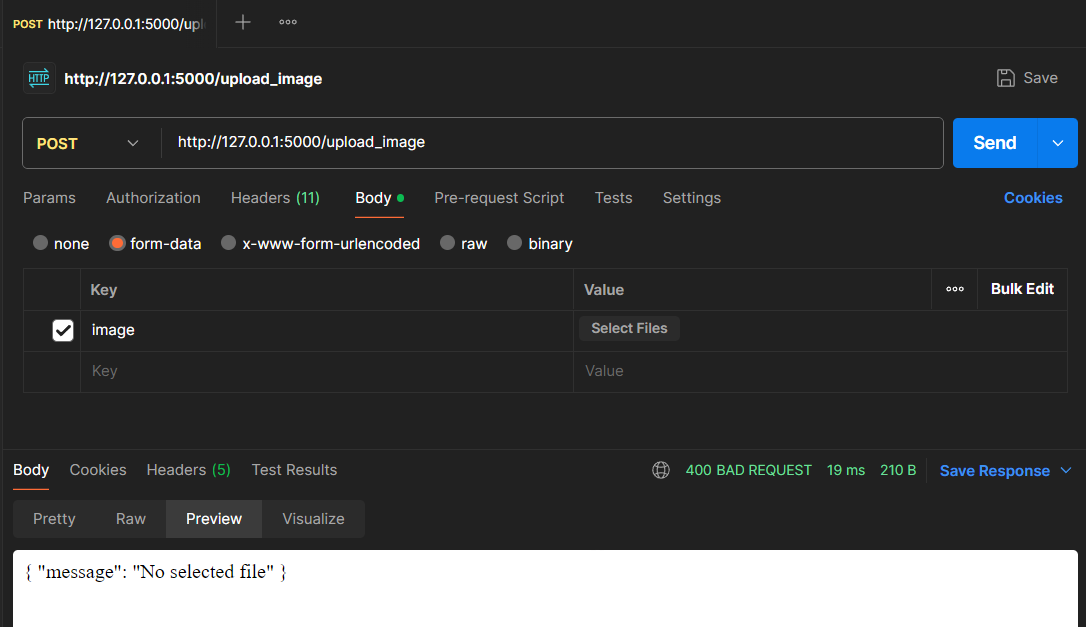
If any of the username or password is incorrect:



* **Image Upload (/upload\_image)**
  + **Method:** POST
  + Expects a file with the key image in the request.
  + Uploads the image to Azure Blob Storage and returns the image URL.



If no file is selected:



If image part is empty:

