SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHISALAI, CHENNAI – 600119
MARCH– 2023

# CTS CAFÉ PROGRAM

## HEALTH CHECK WITH PYTHON

**BY STUDENTS:** RANJITH. G (TEAM LEADER) - 41731100
A. MOHAMED A AFTAAB - 41731075
K. MANI TEJA - 41110771
G. LAKSHMI NARASIMHA REDDY – 41613007
J N V MADHAVI - 41611078
M. VISHAL - 41614103

**MENTOR:** Mrs. PARVEEN . A

# Technical Solution Approach

CONTENTS:

# 1. INTRODUCTION

Application health has become the core of any business. An application which is not stable(unhealthy) leads to lots of revenue loss, customer trust and negative impacts on business. To avoid such cases, we should regularly monitor the status of our application. Using these health checks, we can be notified of outages before customers and reduce the impacts of outage by fixing them or informing the customer about the ongoing outage with time to up which leads to gaining the trust.

A health check is a way of monitoring an application's health, by verifying that all of its dependencies and services are running and responding as expected.

Health checks are typically used in production environments to ensure that an application is running smoothly, and can be integrated into a continuous delivery pipeline to detect issues before they become critical.

With health check, developers can create custom health checks that can be run at any time to verify the state of the application. The library provides a simple API for defining health checks, and includes a number of built-in checks forcommonservices such as databases, cache systems, and message queues.

Monitoring depends on complexity and criticality of the application. A simple application may require some monitoring its dependencies once a day, whereas a critical application may be required to be monitored as frequently as possible. We can also provide a status page to see the results and add functionality to notify developers about problems.

Health check can allow near-real-time information about the state of your application. It helps the end users to constantly monitor the health of the program, be proactive in identifying any difficulties, and make the appropriate adjustments rather than waiting on the end user or system to alert them there isa problem with the application. The health check app makes it easy to send yourtest results and other information directly to you.

## 1.1 ABOUT THIS DOCUMENT:

We are using .NET Core Web API to implement health check. Health Check is a Python library used to build and monitor the health of services. It provides a set of utilities for building simple health checks for services, containers, and infrastructure.

Health Check can be used to monitor the status of different components of a system, such as a web server, a database, a queue, or any other third-party service used by the application. The library supports different types of health checks, such as HTTP endpoint checks, TCP port checks, and custom checks.

When we create health checks, we can create very granular, specific checks for certain services, which helps us greatly when diagnosing issues with our application infrastructure, as we can easily see which service/dependency is performing poorly. Our application may still be up and running, but in a degraded state that we can't easily see by simply using the application, so having health checks in place give us a better understanding of what a healthy state of our application looks like. Instead of relying on our users reporting an issue with the application, we can monitor our application health constantly and be proactive in understanding where our application isn't functioning correctly and be proactive in needed.

A health check can specify a database query to run as a Boolean test to indicate ifthe database is responding normally.

### 1.1.1 PURPOSE AND SCOPE OF THE HEALTHCHECK

The purpose of performing a health check using Python is to ensure the proper functioning of a system or a service. This checkup validates the health of the system by verifying whether it is running properly or not, and whether it is meeting its performance requirements or not. It is essential to perform health checks regularly to prevent failures, improve the response time, and keep the system up to date. If any problem occurs while application is running then it will inform to the end users. Python programming language is utilized for automating the health check process and simplifying the detection of performance and security issues. This Python-based health check process can bescheduled to run at regular intervals and send notifications in case of any issuesfound. Overall, health checking using Python is vital to improving the reliability and availability of the system, ensuring the continuous delivery of services, and enhancing the customer experience.

## SCOPE OF THE HEALTH CHECK

The scope of healthcheck using Python can vary depending on the specific use case, but generally involves testing the integrity, availability, and performance of various components in a system or application. Some common areas that might be covered by a healthcheck include:

***1.*** **Database connectivity**: Checking that the application can successfully connectto and interact with its databases

***2.*** **Network connectivity**: Testing that the application can access external services and communicate with the network

***3.*** **Server resources**: Checking that the application has sufficient resources such as CPU, memory, and disk space

***4.*** **API endpoints**: Testing that the API endpoints are functional and respondingas expected

***5.*** **Security**: Ensuring that security measures such as password strength, encryption, and firewalls are in place and working properly.

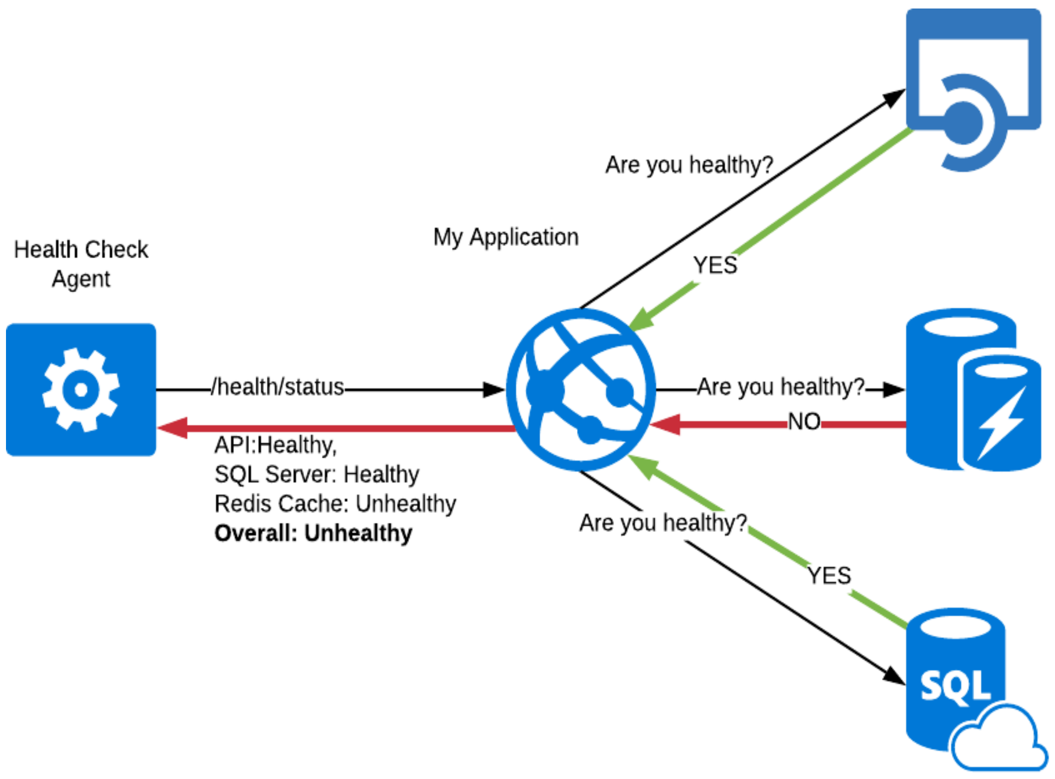
***6.*** **Error logs**: Checking application error logs and identifying the potential issuesthat require debugging.

By implementing a comprehensive healthcheck strategy that includes multiple areas like these, developers can ensure that their applications are reliable, secure, and performant, reducing the occurrence of production failures and improving overall user experience. Python, with its extensive range of libraries, can help in automating the healthchecks and provide a more efficient diagnosisof issues.

In this topic we will look into how we can design a proper health check component using python programming, along with the flowchart to illustrate the process.

- First, let's start with the python code to get a better understanding.

## 2.1 Component Design Diagram:



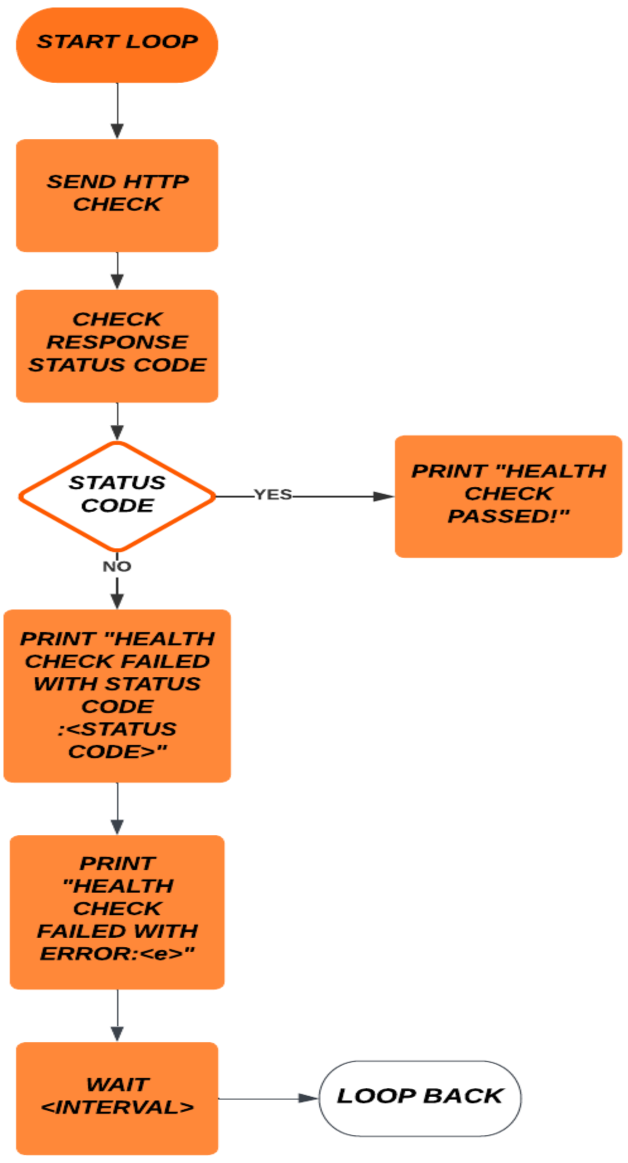
## 2.1.1 Overall Workflow

**Python code (SKETCH):**

```
Import
requestsImport
time
#Define the URL to check and the expected response status code
CHECK_URL = "www.example.com"
EXPECTED_STATUS_CODE = 200
#Define the time interval between checks (in seconds)
CHECK_INTERVAL = 60
          def
  check_health():
     while True:
          try:
            response = requests.get(CHECK_URL)
              if response.status_code ==
                  EXPECTED_STATUS_CODE:print("Health check
                  passed!")
              else:
                  print("Health check failed with status code:" ,
                           response.status_code)
          except Exception as e:
              print("Health check failed with error:",
             str(e))time.sleep(CHECK_INTERVAL)
check_health()
```

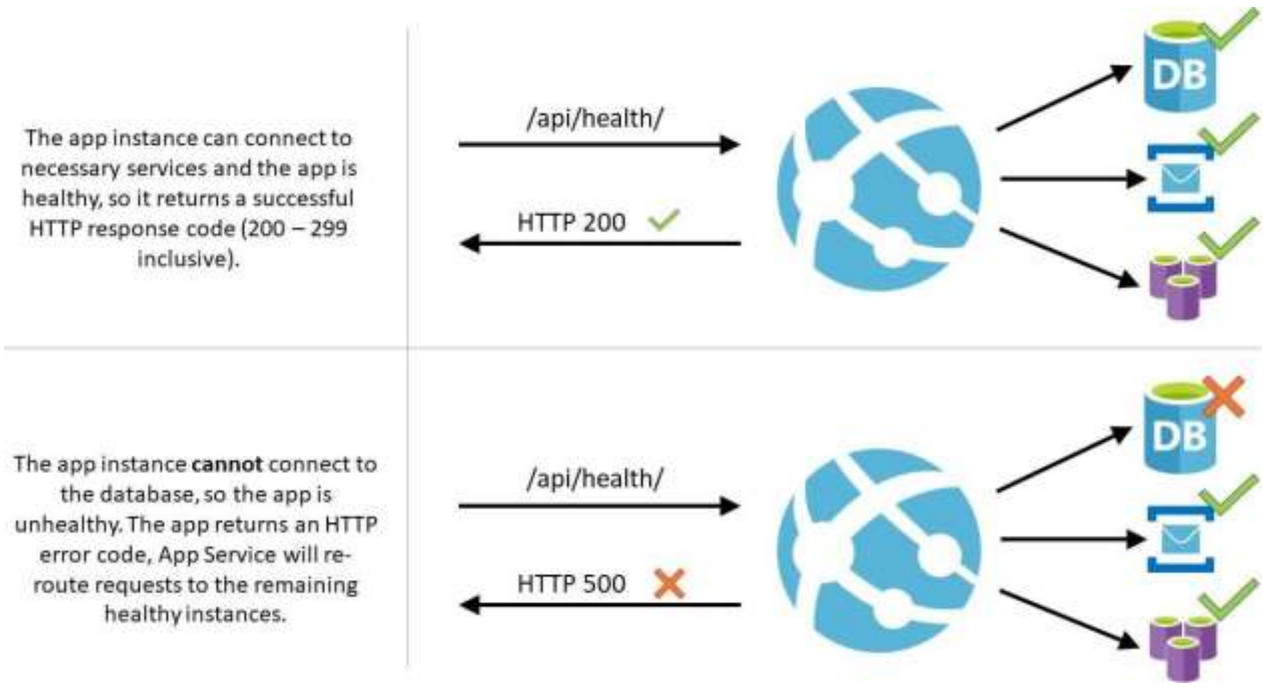
The above defines a '**check_health**' function that continuously sends requests to a specified URL and checks the response status code. If the status code matches the expected value (in this case, 200), the health check passes. If not, the health check fails and an appropriate message is printed to the console.

To illustrate the flow of this program:Flowchart for health check:


START LOOP
SEND HTTP CHECK
CHECK RESPONSE STATUS CODE
STATUS CODE
YES
NO
PRINT "HEALTH CHECK PASSED!"
PRINT "HEALTH CHECK FAILED WITH STATUS CODE :<STATUS CODE>"
PRINT "HEALTH CHECK FAILED WITH ERROR:<e>"
WAIT <INTERVAL>
LOOP BACK

The above flowchart shows the basic process of the health check component. The loop continually sends HTTP requests to the specified URL and checks the response status code. If the status code matches the expected value, the health check passes and a message is printed to the console. If not, a different messageis printed depending on the type of error encountered. The loop then waits for a specified interval before starting the process again.

## 2.1.1 LOW LEVEL DESIGN


The app instance can connect to necessary services and the app is healthy, so it returns a successful HTTP response code (200 – 299 inclusive).
/api/health/
HTTP 200
DB
The app instance **cannot** connect to the database, so the app is unhealthy. The app returns an HTTP error code, App Service will re-route requests to the remaining healthy instances.
/api/health/
HTTP 500
DB

# 3. TECHNOLOGY AND FRAMEWORK

Building a healthcheck system requires a combination of technologies and skills, including backend and frontend development, data processing, visualization, and deployment. In our project we are using the following technologies.

- **Visual Studio 2019:** Visual Studio is an integrated development environment (IDE) that will be used to develop the backend of a healthcheck software using .NET Framework or .NET Core. Visual Studio provides a rich set of tools and features that make it easier to write, test, and deploy code.

- **Docker :** Docker is a containerization platform that allows applications to be packaged and deployed in a consistent and portable way. With the help of Docker we will isolate our application and its dependencies in a container, which can then be easily deployed to different environments without any configuration changes

- **.NET Core 6.0:** .NET Core is a cross-platform, open-source framework which will be used to build the backend of a Healthcheck system. With its extensive support for building APIs and web applications, .NET Core will be used to create RESTful APIs that can be used to expose the health status of various components in the system. .NET Core will also be used to create scripts that can be run in a Docker container, which can be deployed to various environments to run the Healthcheck system.

- **MongoDB :** MongoDB is a NoSQL database which is used to store data for a healthcheck software. MongoDB is a document-oriented database, which means that it stores data in JSON-like documents instead of tables. This makes it well- suited for storing and querying data in a flexible and scalable way.

- **Django :** Django is a popular web framework for Python.we are going to use Django framework to develop the frontend of a healthcheck software. Django provides a set of tools and libraries for building web applications quickly and efficiently. It includes built-in support for features such as authentication, routing,templating, and database integration.

.

- **Python :** Python is a popular programming language that will be used to build the backend of a Healthcheck system. With its extensive libraries and frameworks such as Requests and Selenium, We will use Python to automate the process of checking the health of various components in the system, such as servers, databases, and applications. With help of Python we will create scripts that can be run in a Docker container, which will be deployed to various environments to run the Healthcheck system.

## 4. SOLUTION APPROACH

To implement a health check for an HTTP service using Python, you can follow these steps:

1. **Choose a framework**: For creating HTTP services, Python provides a number of frameworks, including Flask, Django, and Pyramid. Pick the one that best fits your requirements and area of expertise.

2. **Create a health check endpoint**: Create an endpoint that provides a straightforward response confirming that the service is active. This could be a JSON answer with more details about the service's status or a plain text response.

3. **Add more health checks**: You might need to add more health checks to make sure the service is operating properly depending on how sophisticated your application is. You might wish to examine the connectivity of your application's dependencies, such as the database and cache, and other requirements.

4. **Schedule health checks**: To make sure your service is always operating properly, you can schedule the health checks to run on a regular basis. You can use a librarylike APScheduler to schedule the health checks.

5. **Stakeholder notification**: Using email, SMS, or other communication channels, you can alert the necessary stakeholders if a health check is unsuccessful. To send notifications, you can make use of a package like smtplib or Twilio.

6. **Log health check results**: For the purpose of troubleshooting and auditing, it is crucial to record the results of the health check. The results of the health check can be recorded using a logging library like logging.