

**STRUCTURAL HEALTH MONITORING OF PAMBAN  
BRIDGE**

**A PROJECT REPORT**

*Submitted by*

**SHIVANI CHIRANJEEVI**

**715516104044**

*in partial fulfilment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

**PSG INSTITUTE OF TECHNOLOGY AND APPLIED RESEARCH**

**COIMBATORE 641 062**

**ANNA UNIVERSITY:: CHENNAI 600 025**

APRIL 2020

## **BONAFIDE CERTIFICATE**

Certified that this design project report “**STRUCTURAL HEALTH MONITORING OF PAMBAN BRIDGE**” is the bonafide work of **SHIVANI CHIRANJEEVI (715516104044)** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

---

**SIGNATURE**

**DR. R. MANIMEGALAI**

**HEAD OF THE DEPARTMENT**

Department of Computer Science  
and Engineering

PSG Institute of Technology and  
Applied Research

Neelambur, Coimbatore - 641 062

---

**SIGNATURE**

**DR. R. MANIMEGALAI**

**SUPERVISOR**

Department of Computer Science  
and Engineering

PSG Institute of Technology and  
Applied Research

Neelambur, Coimbatore - 641 062

**Submitted for the University Viva Voice held on \_\_\_\_\_**

---

**INTERNAL EXAMINER**

---

**EXTERNAL EXAMINER**



Department of Civil Engineering  
Indian Institute of Technology Madras  
Chennai 600036

Dr. U. Saravanan  
Professor

Ph: 91-44-22574314  
Email: [saran@iitm.ac.in](mailto:saran@iitm.ac.in)

---

**TO WHOM SO EVER IT MAY CONCERN**

Shivani Chiranjeevi was an intern from January 2, 2020 to March 31, 2020 at Department of civil engineering in IIT Madras. She was under my supervision and developed python programs for downloading files from onedrive, converting the downloaded binary files to ASCII, and communicating the results of analysis of the converted files by SMS and email. Shivani was dedicated and completed the tasks assigned independently and successfully.

Sincerely,

(U. Saravanan)

# VeriGuide - Originality Report

## Individual Report

### Background Information

File Name: Thesis\_Shivani\_Chiranjeevi\_1.docx  
 Report Generated On: 02/06/2020, 10:19:53 PM

### Similarity Statistics Overview

Similar Sentence(s) Found By VeriGuide: 96 out of 856 sentences = 11.21%

Similar Sentence(s) Filtered by User: 96 out of 856 sentences = 11.21%

Sentence(s) Selected By User To Export: 0

### Similarity Statistics for Each Source

Entry	Source	From	Similarity
1	<a href="https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-auth-code-flow">https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-auth-code-flow</a>	Internet	13 / 856 = 1.52%
2	<a href="https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-permissions-and-consent">https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-permissions-and-consent</a>	Internet	13 / 856 = 1.52%
3	<a href="https://www.omega.co.uk/prodinfo/StrainGauges.html">https://www.omega.co.uk/prodinfo/StrainGauges.html</a>	Internet	12 / 856 = 1.4%
4	<a href="https://auth0.com/docs/flows/guides/auth-code/call-api-auth-code">https://auth0.com/docs/flows/guides/auth-code/call-api-auth-code</a>	Internet	11 / 856 = 1.29%
5	<a href="https://tools.ietf.org/html/rfc6749">https://tools.ietf.org/html/rfc6749</a>	Internet	10 / 856 = 1.17%
6	<a href="https://scholarworks.alaska.edu/bitstream/handle/11122/7476/Y.Dong-et-al_Bridge-SHM-Deterioration-Detection_Final-Dec2010.pdf">https://scholarworks.alaska.edu/bitstream/handle/11122/7476/Y.Dong-et-al_Bridge-SHM-Deterioration-Detection_Final-Dec2010.pdf</a>	Internet	8 / 856 = 0.93%
7	<a href="https://docs.apigee.com/api-platform/security/oauth/access-tokens">https://docs.apigee.com/api-platform/security/oauth/access-tokens</a>	Internet	7 / 856 = 0.82%
8	<a href="https://docs.microsoft.com/en-us/azure/active-directory/develop/access-tokens">https://docs.microsoft.com/en-us/azure/active-directory/develop/access-tokens</a>	Internet	7 / 856 = 0.82%
9	<a href="https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-on-behalf-of-flow">https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-on-behalf-of-flow</a>	Internet	7 / 856 = 0.82%
10	<a href="https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-protocols-oidc">https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-protocols-oidc</a>	Internet	7 / 856 = 0.82%
11	<a href="https://en.wikipedia.org/wiki/Windows_Registry">https://en.wikipedia.org/wiki/Windows_Registry</a>	Internet	7 / 856 = 0.82%
12	<a href="https://www.omega.co.uk/prodinfo/accelerometers.html">https://www.omega.co.uk/prodinfo/accelerometers.html</a>	Internet	7 / 856 = 0.82%
13	<a href="https://docs.microsoft.com/en-us/rest/api/azure/">https://docs.microsoft.com/en-us/rest/api/azure/</a>	Internet	5 / 856 = 0.58%
14	<a href="https://stackoverflow.com/questions/7030694/why-do-access-tokens-expire">https://stackoverflow.com/questions/7030694/why-do-access-tokens-expire</a>	Internet	5 / 856 = 0.58%

## **ABSTRACT**

Structural Health Monitoring (SHM) is the process of implementing a damage detection and characterization strategy for civil structures such as buildings, bridges, workspaces etc. SHM is most recently used for evaluating and monitoring structural health of civil structures mentioned above. It has been widely applied in various engineering sectors due to its ability to respond to adverse structural changes, improving structural reliability and life cycle management. SHM is a popular method of non-destructive evaluation of a structure. It involves placement of sensors along the structure, integration of sensors, data accumulation, data transmission and analysis.

SHM enables the continuous monitoring often necessary to detect the occurrence of disasters leading to collapses and also periodic monitoring to gain insights into the structural capacity over prolonged period. SHM methods have evolved greatly from the early hammer striking methods to the current wired sensors to detect the damage. Wired sensors are often deployed with minimal costs and are not affected by distance or any electronic interference. Further, high data collection rate allows the system to handle huge volume of data generated in real time.

The SHM system in this project is implemented for a truss bridge in a marine environment in India, the Pamban Bridge at Rameshwaram in Tamil Nadu. A wired sensor network is deployed that can efficiently monitor the same and collect the data to implement SHM. All sensors have been linked to form a sensor network and the readings are sent over to cloud storage. The readings are then formatted and analysed for any abnormality. A notification system is designed and implemented as a part of this project work. The proposed SHMON-PB, an IoT based Structural Health Monitoring for Pamban Bridge, intimates the concerned if any anomaly is noted. SHM often eliminates the need for manual inspection for any minor damages which saves lot of cost, effort and resources.

The project includes the necessary implementations of modules that retrieve the data from the cloud platform after authorization, convert the format for further analysis, schedule the modules to run automatically and finally send alert notifications when any mishaps occur. Analysis of sensor values continuously helps understand the dynamics of the impact various loads and factors might have on the structure over a period of time. The ultimate aim of the project is to reduce the damages that might occur in the long run if not addressed at an earlier point in time. The analysis run on the sensor data helps achieve the goal and bring down the costs required in maintenance of the structure.

## **ACKNOWLEDGEMENT**

I express our deep sense of gratitude to our Managing Trustee **Shri L Gopalakrishnan** for his invaluable guidance and blessing in the project.

I am very grateful to **Dr.P.V.Mohanram**, Principal for providing me with an environment to complete our project successfully.

I take the privilege of expressing my gratitude to **Dr. G. Chandramohan**, Vice Principal in extending his support to carry out our study.

I am greatly indebted to **Dr. R. Manimegalai**, Head of the Department, Computer Science and Engineering for her guidance which was instrumental in the completion of my project.

I express my sincere thanks to **Dr. Bhuvana**, Project Coordinator for her constant encouragement and support throughout my course.

Finally, I take this opportunity to extend my deepest appreciation to my family and friends, who supported me during the crucial times of our project.

**SHIVANI CHIRANJEEVI**

## TABLE OF CONTENTS

<b>CHAPTER NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
	<b>ABSTRACT</b>	<b>i</b>
	<b>LIST OF FIGURES</b>	<b>vii</b>
	<b>LIST OF TABLES</b>	<b>viii</b>
	<b>LIST OF ABBREVIATIONS</b>	<b>ix</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Overview of Structural Health Monitoring	1
	1.2 History and Evolution of SHM	2
	1.3 History & Overview of Pamban Railway Bridge	5
	1.3.1 Specimen Characteristics	5
	1.3.2 Truss Bridge	5
	1.3.3 Structural Properties	6
	1.4 Scope and Motivation	6
	1.5 Objectives of the Project Work	8
	1.6 Contribution of the Project Work	8
	1.7 Organization of the Project Report	9



<b>CHAPTER NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>10</b>
	2.1 Highway Bridge Assessment using an Adaptive Real-Time WSN	10
	2.2 Cyber-Physical Codesign SHM with WSN	12
	2.3 Urban Highway Bridge Structure Health Assessments using WSN	13
	2.4 A Real-time Health Monitoring and Warning System for Bridge Structures	15
	2.5 Design of WSN for Real-Time SHM	17
	2.6 Acoustic Emission and Ultrasonic sensor based SHM	19
<b>3</b>	<b>STRUCTURAL ANALYSIS</b>	<b>20</b>
	3.1 Finite Element Analysis	20
	3.2 Sensors Deployed	22
<b>4</b>	<b>SENSOR DATA RETRIEVAL FROM CLOUD</b>	<b>24</b>
	4.1 Microsoft One Drive	24
	4.2 One Drive for Python	24
	4.2.1 Microsoft Identity Platform	25
	4.2.2 Azure Active Directory	25
	4.2.3 Authentication Flows	25
	4.2.4 Scopes and Permissions	26
	4.2.5 Security Tokens	27

4.2.6	Microsoft Graph API	28
4.2.7	OAuth 2.0 Authorization Protocol	28
4.2.8	OpenID Connect	30
4.2.9	OAuth Authorization Code Flow	30
4.2.10	Refreshing the Access Token	35
4.2.11	Downloading Files	35
<b>5</b>	<b>DATA HANDLING AND STORAGE</b>	<b>37</b>
5.1	Python Libraries	37
5.1.1	struct	37
5.1.2	xml.etree.ElementTree	39
5.2	Storage of Raw and Processed Sensor Files	40
5.3	Executing the Analysis Code	40
5.4	Notification System	41
5.4.1	Mailing the Analysis Report	41
5.4.2	Alert Messages via SMS	42
5.5	Automation	45
5.5.1	Code Automation	45
5.5.2	Running on Boot-up	45
<b>6</b>	<b>CONCLUSIONS AND FUTURE WORK</b>	<b>47</b>
	<b>APPENDIX</b>	<b>48</b>

## LIST OF FIGURES

<b>FIGURE NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
1.1	Block Diagram of a Typical Structural Health Monitoring System	2
1.2	Pamban 2- Leaf Bascule Truss Bridge Making Way for Water Traffic	5
4.1	OAuth 2.0 Authorization Protocol	28
4.2	Workflow of OAuth 2.0 Authorization Protocol	29
4.3	Client App Registration on Azure AD	30
4.4	Client App Details on Azure AD Dashboard	31
4.5	Configuring Platform and Choosing Redirect URI on Azure AD	32
4.6	Adding Scopes Required to Request User's Consent on Azure AD	32
4.7	Token Returned by the Authorize Endpoint	33
4.8	Access Token and Refresh Token Returned by the Token Endpoint	34
4.9	Data retrieval Using Python	35
5.1	Header Data Conversion	38
6.1	Twilio Account Creation	41
6.2	Twilio Issue Phone Number Purchase	42
6.3	Account Details Displayed on Twilio Dashboard	42
6.4	Twilio SMS Sending Mechanism	43

## LIST OF TABLES

TABLE NO	TITLE	PAGE NO
1.1	History of Health Monitoring Techniques of Concrete Structures	4
5.1	Binary File Description	37
5.2	Format strings of <i>struct</i> with its Corresponding Size	37

## LIST OF ABBREVIATIONS

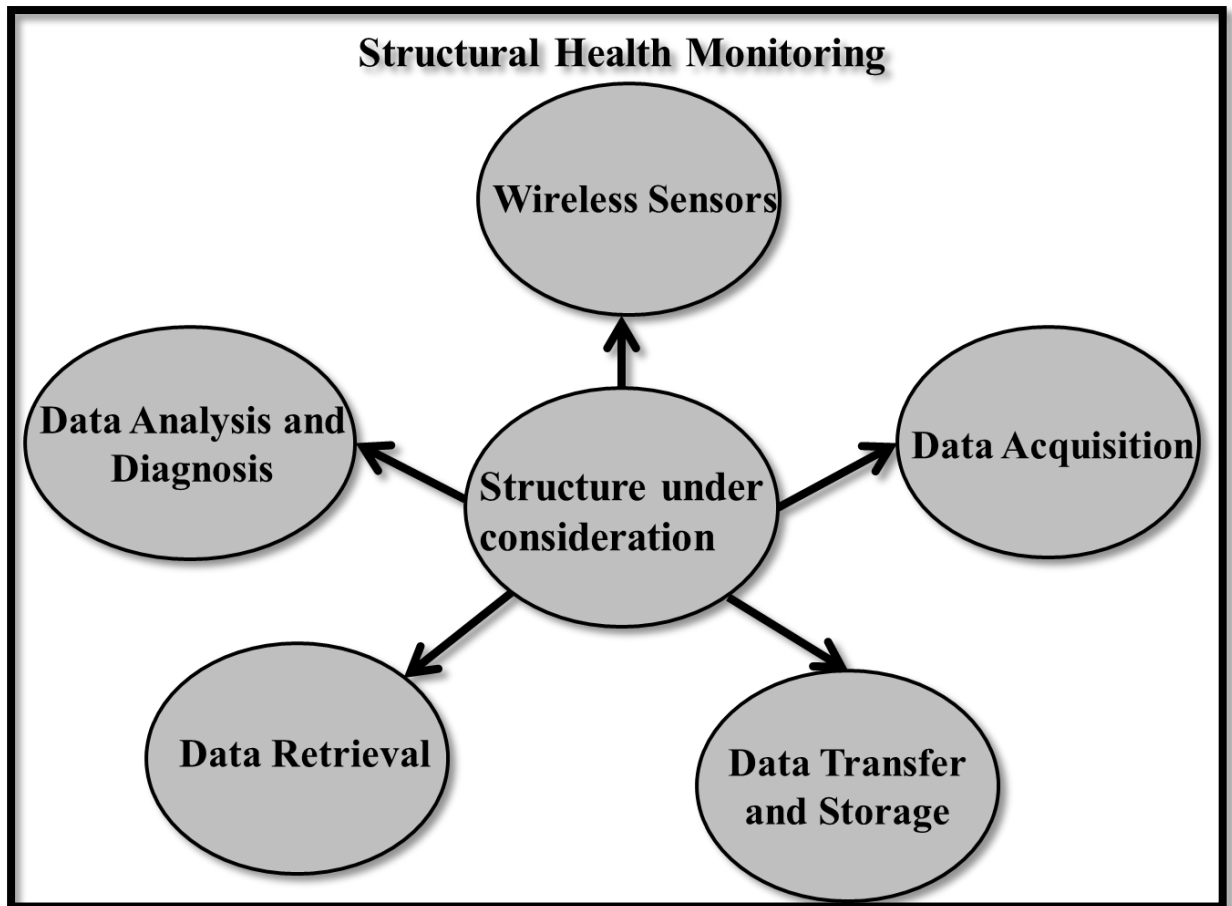
AD	Active Directory
ADAL	Azure Active Directory Authentication Library
AE	Acoustic Emission
DLAC	Damage Location Assurance Criterion
FEA	Finite Element Analysis
JWT	JSON Web Tokens
MIP	Microsoft Identity Platform
MOM	Message Oriented Middleware
MSAL	Microsoft Authentication Library
OIDC	OpenID Connect
SHM	Structural Health Monitoring
SPANNeT	Stream Processing and Artificial Neural Network
WAG	Weighted Attack Graph
WSN	Wireless Sensor Networks

# 1. INTRODUCTION

## 1.1 Overview of Structural Health Monitoring

The Pamban Railway stretch is the specimen which is being monitored with wired sensor networks . These wired sensors are placed along the entire length of the bridge at particular locations. The sensors namely the accelerometers, strain gauges and the temperature sensors measure the corresponding values and transmit the data over cloud. This data has to be formatted, analysed and checked if the values exceed the threshold or not to monitor the bridge for potential areas of damage. Structural analysis software is used to simulate the behaviour of the specimen under the anticipated loads over prolonged periods to decide the areas where retrofitting has to be done which uses information such as the dimensions of the specimen, material of construction etc. If there are discrepancies in the range of the sensor readings from the nominal values, an alert notification has to be sent to the concerned.

Structural Health Monitoring (SHM) is vital to any concrete structure that serves a purpose to the daily functioning of humans. The abstract architecture of SHM is shown in Figure 1.1. Any disruptions to the daily routine of humans result in economic losses and inconvenience to the public. A railway bridge situated right above the sea which acts as the primary means of transportation is often very crucial but also prone to accidents which might cost many lives. Also damage to such structures often results in significant costs for replacement and construction. Hence SHM is very necessary in such an environment to prevent any inconveniences to every day's routine and to ensure the safety of the commuters.



**Figure 1.1 Block Diagram of a Typical Structural Health Monitoring System**

## **1.2 History and Evolution of SHM**

SHM has evolved greatly from the earliest methods of recurrent hammer strikes at the specimen known as the hammer test and continuous visual inspection for cracks, faults or any other visible deformations. This is primarily because of the widespread civilization resulting in rapid development of concrete structural projects and that these methods were very becoming inefficient as reliable methods of inspection. For the same reasons, the above methods don't formally count as SHM. Inefficiency of the methods due to drastic increase in the structures has led to development of more reliable methods of health monitoring in the recent past.

Few methods have been discussed in this section [1]. In 2002, Roover et al [2] have proposed a Digital Image Correlation (DIC) that uses multiple digital cameras. The pictures are captured from time to time and stored to determine strain. These patterns are compared with the currently captured pictures to identify the patterns for damage. Chain Dragging has been implemented in 2003 by Scott et al [3]. This involves dragging the chain around the structure and listening to the acoustic reflex from the structure to detect subsurface abnormalities. The accuracy and the reliability of the method depend on the inspector who listens to the sound and analyses it for any damages. An improvement to this implementation was an automated form. Fibre optics are widely used in many projects and it can be used to measure various parameters such as strain, pressure, vibration frequencies and is used for both local and global monitoring. One such work has been designed by Casas et al [4] in 2003. The change in intensity of light observed at the other end will provide information about the areas of damage in the structure. Using ultrasonic waves has been a popular method of non-destructive evaluation in many fields and has been used extensively in bridge health monitoring. When these waves hit the surface, the reflections of the wave are read by a transducer and the impact of the waves implies details about the areas of damage as suggested by Iyer et al [5] in 2003.

Another well-known technique to measure the strain measurements is by using the electrical resistance strain gauges that has been proposed by Dally et al [6] in 2005. Due to prolonged impact of loads on the material, it stretches minimally over time and it changes the electrical resistance properties of wire. This gives insight into the structural capacity of the material. A commonly used sensor for SHM is accelerometer which measures dynamic phenomena and is deployed for continuous monitoring of the structure. The vibrations caused by a moving object over the bridge are captured by the sensor. This strategy has been



put into use by Park et al [7] in 2007. Acoustic Emission (AE) generally uses piezoelectric sensors to detect minimal amount of energy released from points of damage from where the energy is emitted as stress waves as in Ji et al's work [8] in 2008. A network of sensors is used to determine the area. The release of energy is usually due to any crack, faults and corroded regions. The faults in any part of the structure can be detected with this sensor but a major disadvantage is the intervention of the background noise with the sound signal which affects the results. The chronological order of the implementation of the above methods is illustrated in Table 1.1. Many sensors ranging from the acoustic emission sensors, fibre optic sensors to the commonly used accelerometers are either used individually or in combination according to the needs. Wired sensors are still preferred due to their ease of maintenance and durability owing to their consistency of data transmission.

<b>Year of Implementation</b>	<b>Health Monitoring Strategies</b>
2002	Digital Image Correlation : Uses visual images of the structure from period to period to assess the damage
2003	Chain Dragging : Involves dragging a chain around to observe the subsurface abnormalities from the resulting sound
	Using Fibre Optic sensors : Uses fibre optic sensors to measure strain, pressure, vibration frequencies etc.
	Ultrasonic Wave evaluation : Involves making an ultrasonic wave incident on the surface and forming insights from the reflected wave
2005	Electrical Resistance Strain Gauges :Measures any change to the cross-sectional area or length and thus identifying strain at the location
2007	Health Monitoring using Accelerometers : The sensors measure dynamic phenomena and used for continuous monitoring
2008	Acoustic Emission : Use of piezoelectric sensors to capture distressed sound signals from points of damage

**Table 1.1. History of Health Monitoring Techniques of Concrete Structures**

## **1.3 History and Overview of Pamban Railway Bridge**

### **1.3.1 Specimen Characteristics**

The Pamban Bridge is located in Rameshwaram in the state of Tamil Nadu and is shown below in Figure 1.2. It connects the village of Mandapam and the Pamban Island. It has been in use ever since 1914. It was the longest sea link until the construction of the Bandra-Worli sea link in Mumbai. The structure is placed 41 feet above the sea level. It spans around 2 kilometers. Owing to its corrosive environment, the structure has been heavily corroded until it could not be used further. It became operational again in February 2019.



**Figure 1.2. Pamban 2- Leaf Bascule Truss Bridge Making Way for Water Traffic**

### **1.3.2 Truss Bridge**

A truss bridge is a bridge whose load-bearing superstructure is composed of a truss, a structure of connected elements usually forming triangular units.

The connected elements may be stressed from tension, compression, or sometimes both in response to dynamic loads. Trusses are popular for bridge building because they use a relatively small amount of material for the amount of weight they can support. They commonly are used in covered bridges, railroad bridges, and military bridges. The connected pieces forming the top and bottom of the truss are referred to respectively as the top and bottom chords. The sloping and vertical pieces connecting the chords are collectively referred to as the web of the truss. The component parts of a truss bridge are stressed primarily in axial tension. Axial forces refer to the internal resisting forces developed only along the length of the element.

### **1.3.3 Structural Properties**

The entire span is constructed using steel. It is called a bascule bridge because it is movable, i.e. it moves upwards making a passage for the boat and ship traffic. It has a double leafed mid span which carries its own weight and the external load i.e. the train when it is closed. It has a truss structure which basically has alternate triangular units fitted together and has nodes and members as its constituents. In case of a truss, the load is applied only at the nodes and only axial forces are developed along the length of its members. The structure is subjected to the load of a train passing over.

## **1.4 Scope and Motivation**

In the past, many accidents and bridge collapses have occurred due to the most common reasons such as structural and design deficiencies, corrosion, construction and supervision mistakes. Major accidents in the past century have occurred for reasons such as overloading, corrosion of girders and other minor cracks and faults which could easily been avoided if timely inspection had taken place. They are aging with years and are also subjected to harsh loading scenarios and severe environmental conditions. In most of the cases, the

accidents occur not just because of a single factor but a combination of many factors listed above. For instance, Mississippi River Bridge in Minneapolis, United States collapsed suddenly in 2007 and the reason stated was the thin gusset plates that tore a series of rivets. The same plates had withstood the loads for many years previously. The trigger here was the extra load of machinery that caused the collapse. But timely inspection could have been able to make repairs to the bridge to prevent the mishaps. Here, a combination of three factors, namely, structural deficiency, additional external load and poor supervision have been attributed to the cause.

After a handful of such major hazards costing many lives and deterioration to the structures, the need for health monitoring techniques were increasingly becoming important. In this project, the Pamban Bridge under consideration was temporarily suspended in December 2018 as a fissure was noticed and the reason was that corrosion was noticed in various parts of the steel girders. After restoration, services were resumed from February 2019. Given the location of the bridge, it is often very difficult and time consuming to visually inspect from time to time. This is the main challenge that SHM overcomes in terms of minimising the human intervention and bringing the anomalies in structure to the notice of the inspectors at an earlier stage of damage. Secondly, the use of wired sensors has countless benefits such as minimal cost, easy installation, replacement, power efficiency and damage localization. SHM using sensors is by far the most economically feasible and practical solution presented in recent times. Ultimately, the aim is to localize the damage at preliminary phases to increase the lifetime of the structures, enhance public safety and reduce the operational costs. The project addresses many of the points discussed above with an efficient and affordable implementation.

## **1.5 Objectives of the Project Work**

The structure is subjected to loads, external and unexpected environmental conditions all the time and both continuous and periodic monitoring are required to know if any sudden mishaps might occur earlier to take any measures required and also know how various external factors impact the structural capacity over time. The sensors deployed are capable of both the types of monitoring. The objective of any monitoring system is to be implemented with ease and minimal cost. It is possible to even bring down the constructional costs if any damage is identified earlier by carrying out retrofitting to strengthen the existing structure. The sensors are entities that could be easily installed, maintained, replaced and very inexpensive. They could be placed along the length of the bridge wherever desired and if any sensor is found to be faulty, it could be easily replaced. The data acquired every time is transmitted over the network to a cloud platform. The wired sensors are not impacted by electronic interference or distance . The monitoring of the sensor values over a prolonged period helps improve the future design based on experience in the future. It also helps in assessing the integrity of the structure after any accident or major damage to it. Consequently, it leads to the decline in additional construction and any growth in maintenance needs. This potentially makes way for an efficient designing mechanism that is heavily based on the structural performance.

## **1.6 Contribution of the Project Work**

SHMON-PB is a solution containing modules which perform the functions of acquiring data, transmitting it, storing it, formatting it and finally transmitting alert messages. The sensor readings are stored into binary files which are very minimal in size and hence the transmission is fast and the power consumption is reduced. The One Drive where the files are stored is a very secure platform

which provides only authorized access to the protected resources. The files are downloaded at the side where analysis is done. The data is initially converted to desired format. Then analysis of the values is performed and based on the results, notifications are sent out if required. This project aims to bring down the transmission time of the data and intimate at a premature stage of damage to have preventive measures taken from time to time.

## **1.7 Organization of the Project Report**

The project report comprises of six chapters. Chapter 2 discusses about the implementations done in the past with its benefits and drawbacks. Chapter 3 briefly describes about the structural analysis using analysis software which builds a three-dimensional model and simulates real-time conditions such as anticipated loads and other external factors. Chapter 4 discusses in detail about how the cloud is set up, the authorization and authentication protocols followed, and finally how the files are downloaded to the system. Chapter 5 explains how the data is formatted, the Python packages used, how the files are stored to the system and briefs about the execution of the analysis script. It also deals with how the results from the data analysis are used to send out alert messages to the concerned and discusses briefly about the module that automates the running of the programs to minimize the user dependency.

## 2. LITERATURE REVIEW

Chapter 1 discussed about the overview of the implementation of the project, characteristics of the specimen, approaches in the past and the need for SHM. Chapter 2 would be explaining in detail about the different implementations of health monitoring of bridges and about its benefits, drawbacks and how it differs from their previous works.

### 2.1 Highway Bridge Assessment Using an Adaptive Real-Time WSN

Matthew et al [9] have proposed a continuous vibration and periodic strain-based monitoring to track the health of a highway bridge. The authors have proposed a system that uses three sensors namely, the strain transducers to derive the structural characteristics, the temperature sensors to monitor the temperature and the accelerometers to measure the vibration values. It emphasizes on the transmission of data at a high rate with minimal to no loss of packets with dedicated hardware. Other related works use either a strain based or a vibration-based monitoring for diagnosing the structures. The strain transducers reveal information about the characteristics of the structure under an applied load such as its structural capacity whereas continuous vibration system would be able to imply the anomalies occurring due to external factors such as collisions.

But their sole usage doesn't serve its purpose to the fullest. Using only strain-based monitoring requires some properties to be known in advance and can be used only at scheduled intervals and can't account for collisions and other mishaps that is achieved using continuous monitoring of the structure. Due to lack of well-defined database of real time sensor readings, building the diagnostics and prognostics for a structure has been very difficult. The monitoring system consists of three sensors: Strain transducers, dual axis accelerometers, temperature sensors. The length of the bridge is instrumented at

thirty locations and there are total of sixty vibration sensors and thirty strain transducers. The system uses both analogue and digital sensors.

For data transmission, it uses an ultralow power microcontroller with a multiplier and a transceiver chip. Uses an adaptive sensing approach which utilises both experimental load rating and vibration sensing methods where the former which is scheduled periodically, provides information about the structural characteristics such as structural capacity and any existing damages to the structure whereas the latter that is continuously monitored, measures the impact of external factors on the structure. When a load such as a truck passes over the length of bridge at particular intervals, the local strain transducers measure the strain experienced at the location and derive the structural properties and in between the intervals, it switches to vibration mode where accelerometers measure the values. Matthew et al have overcome the power and bandwidth limitations by using an optimal cycle of sensing and transmission where the former is done with ultra-low power sensors and the latter is carried out at a low duty cycle. The low power consumption is accounted to the reduced usage of the radio transceiver. The transceiver in the system is solely responsible for, (i) Superior data rates, (ii) selectivity and (iii) sensitivity. The above mentioned three factors consequently lead to greater real time data transfers. The radio transceiver also has a very high bandwidth that ensures high rate lossless real time transmission. The load and vibration data sent over is then compiled to form a historic database for future analysis. The data success rate over a 11-sample duration was 99.999%. The software on the host computer, due to the high data transmission rate causes the external buffer to overrun when any interrupts irrelevant to the current task arise. Consequently, this corrupts the data packets and contaminates the data processing.



## 2.2 Cyber-Physical Codesign of Distributed SHM with WSN

Gregory et al [10] have proposed a simple system of Wireless Sensor Networks (WSN) for its low expenses in terms of simple installation and maintenance for SHM of a Truss structure. A truss is a structure that consists of members organised into connected triangles so that the overall assembly behaves as a single object. Trusses consist of triangular units constructed with straight members. The ends of these members are connected at joints, known as nodes. These sensors could be deployed at numerous locations making it easier for localised and accurate damage identification. The previous iterations of similar works view the sensor nodes as mere data collecting units which ultimately to serve its purpose cause high energy and bandwidth consumption and long detection latencies. The sensors are organized to form a centralized network architecture. The sensors are required to transmit the entire raw sensor readings. Consequently, this causes increased network overhead. Decentralized algorithms such as Damage Location Assurance Criterion (DLAC) were used previously for damage localization but are at disadvantage due to the following limitations, (i) Damage patterns should be known in advance, (ii) Insensitive to small damages and (iii) Not applicable to asymmetric structures. The system considers both the cyber components as well as the physical components where the former implies the WSN system and the latter implies the SHM requirements. The on-board accelerometers measure the vibration values. It uses flexibility based methods to localize the damage identification. These algorithms work on the basic principle that the structure flexes when a force is applied. As this happens, the stiffness of the material decreases with time and this ultimately localizes the damage. Initially, the base modal parameters are identified. The sensors measure the vibration values simultaneously. Data from multiple sensors are fused to identify the modal parameters. The parameters are

processed to generate the flexibility matrix. The new matrix is subtracted from the old one to identify any damage to the structure

The sensor nodes are organized into clusters with each cluster having elected a cluster head and thus form a cluster hierarchical architecture. Imote2 WSN platform is used which has a CPU that has an embedded SRAM that is capable of satisfying the sophisticated memory requirements for the flexibility computations. The CPU is also dynamically clocked which means the CPU speed could be varied on the go. The platform is designed specifically to meet the limited energy and bandwidth requirements. The limited consumption of energy and bandwidth is largely due to the system's ability to leverage the powerful processing capability of wireless sensor nodes to not fully transmit or process the raw sensor data but only partially process and extract the relevant features. The flexibility-based algorithms are used instead of DLAC which outperforms the latter in various aspects. The sensors are organized into a hierarchical architecture where the sensors at the damaged region are only activated whereas the rest are in sleep mode. This greatly reduces the power consumption. The nodes only transmit the intermediate results that are relevant to the flexibility calculations which in turn reduce both the energy and bandwidth usage. The sensor nodes are organized into a hierarchical decentralized architecture which uses clusters. A clustering algorithm is often complex to be implemented and often involves extra computations to re-elect the cluster head if the existing fails.

### **2.3 Urban Highway Bridge Structure Health Assessments Using WSN**

A WSN consisting of accelerometers which measures the vibration values of different regions of highway bridge giving information about the structural health of the bridge has been proposed by Li et al [11]. The sensor measures the

vibration values at different times under different loads. Also, Finite Element Analysis (FEA) is carried out and the vibration values are simulated for given load conditions. The simulation results are compared with the actual values to analyse the condition of the bridge. Some SHM systems use strain gauges to measure the strain values which are used as indicators of the structure's health. But using the strain measurements for the same has its disadvantages. Strain gauges are instruments that are bulky and consume a lot of power. More over their accuracy is also not as good as that of accelerometers. Also, their installation and maintenance cost are high. The bridge has two pre-stressed box beam bridges which consists of eight accelerometers nodes. The sensor nodes produce approximately 100 samples per second. The sensor nodes are connected to each other in a peer to peer fashion which exchange information with each other and pass on. Each node broadcasts its own data in a single or multiple hops. They are all connected to a base station. The base station is in turn connected to a server computer which collects the sensor data.

The sensor nodes use JAVA based software which are connected with a mesh topology. They send out radio packets which contain the sensor readings of that particular sensor node. The accelerometers provide the vibration values of the bridge under different loads and speeds. These values are mapped to a frequency domain. The vibration values recorded are different for the healthy regions and damaged regions due to its degradation. This is potentially a health indicator of the bridge. A finite element model is built manually where loads and speeds of the vehicle are simulated to provide conditions similar to real time conditions. The vibration values recorded under the simulated conditions are recorded. For this model, the original dimensions of the bridge are provided. The actual results are compared with the simulated results to prove that the reduction of the bridge vibration frequency is a vital indicator of the structure's

health. An enhancement is proposed where a database is to be created which will observe the frequency reductions throughout the lifetime of the bridge.

## **2.4 A Real-time Health Monitoring and Warning System for Bridge Structures**

Considering the huge economic losses, disruption of daily routine of people and their safety, the SHM of the civil structures is becoming more important and the need of the hour. Also, the structures might be subject to some external forces which may cause unexpected accidents. An enhanced SHM System using Stream processing and Artificial Neural Network technique (SPANNeT) has been proposed by Khemapech et al [12]. It provides real time monitoring of the bridges and also a warning system which alerts the concerned. The wireless sensor network collects the sensor data and on-the-go processing of the data happens and artificial neural network is built to handle the strain values of different regions of the bridge which is subject to loads only in the vertical direction. The previous SHM systems used a centralized server to which all the sensor nodes send the accelerometer readings to. The sensor nodes could also be sent into a sleep and wake mode by detecting the near arrival of a train on the bridge to reduce the power consumption significantly. But a centralized unit for processing and data collection poses several disadvantages.

This has urged the development of a decentralized approach for such purposes. Many solutions for the same have been proposed. One among them suggests the usage of the real time sensor measurements to be compared with the pre-determined optimal values to assess the damage. Other suggests the usage of modal frequency shift along with the localization algorithm which can accurately give the area of damage. One solution was to use DLAC whose disadvantages had been mentioned earlier and the main being its inefficiency in identifying the small damages to the specimen. The structure under study is pre-

stressed bridge girders in Bangkok spanning around 20 meters. The structure is simply supported. According to the nature of the structure, the sensor platforms have been chosen. The sensor platform consists of strain gauges with a gauge length of 5 millimetre and accelerometers. The girders are subject to vertical loads both due to the structure itself and vehicles. Due to the loads, internal forces are developed resulting in bending strain which is consequently measured by the strain gauges. The data collected is sent over to the base station. The real-time incoming sensor data is essentially continuously streaming data which is handled by the stream processing sub-system. The input of the stream processing system comes from the Message Oriented Middleware (MOM). The readings are compared to pre-defined patterns to assess the condition of the specimen. The output labelled as *normal*, *warning* or *critical* is intimated accordingly using a web-services based alert system. JAVA has been selected to implement the system keeping in mind the heterogeneity of the sensor platform due to its decentralized nature, data exchange and communication. MOM is used for message exchange and JAVA messaging server is used for real time data exchange.

The algorithm used for damage detection is the SPANNeT Weighted Attack Graph (WAG) which is designed especially for multiple comparisons at a time and takes into account other characteristics that are often important while monitoring the condition. The success of the message transfers is 90% which makes the SPANNeT a very reliable implementation. The SPANNeT WAG unlike the normal WAG is capable of concurrent verifications. Damage localization is achieved here unlike the conventional methods that use DLAC. The usage of decentralized systems for SHM poses a variety of problems. The significant being, it requires the real time data to be persisted and also in order to perform any computations on it. Since huge volume of data is being generated every day, storing such volume causes serious storage issues. T

Harms et al [13] focus on a more power efficient alternative to the above proposal using the Smart Brick technology that uses both on-board and external sensors for measuring both external and structural phenomena and a quad-band modem for long range communication. One major drawback is the cost of the modem and the major advantage being the increased power and data transmission efficiency.

## **2.5 Design of WSN for Real-Time SHM**

SHM has been proposed for industrial and residential buildings by Giammarini et al [14]. The system uses WSN which is also energy and cost efficient. A real time data acquisition of data has to be done to prevent or monitor in case of any unpredictable events such as earthquakes. The system proposes four features which are characteristic of SHM in such setup. They are (i) Synchronized Sampling, (ii) Indoor Installation of Sensor Nodes, (iii) Real Time Monitoring and (iv) Wireless Communication.

The microcontroller used is an ARM Cortex which has sufficient storage, 8 KB of internal RAM, consumes little power and is highly energy efficient. For the first among the characteristics mentioned above, Global Positioning System (GPS) is used. For the second, the sensors used are small enough to be installed indoors without any disruption to normal functioning. The sensor nodes continuously monitor and transmit the data and thus satisfy the third requirement. Wireless communication is made possible using Zigbee-IEEE 802.15.4 and the system uses a specific low power module. Transparent mode of communication is performed where all the nodes communicate with each other. The accelerometer forms the sensor module and a single axis Micro-Electro-Mechanical Systems (MEMS) capacitive accelerometer is used here. This works in a low power mode and is fully calibrated, thus making it a highly stable device. The microcontroller is programmed using an open source library

which is written in C. The data acquisition by the sensor nodes is performed in two ways. In event-triggered, the data acquired is analysed to find if the structure was deformed or not. Based on the result of the analysis, it is decided if the data before and after the event has to be collected or not. In case of time-triggered, the nodes collect the data only during the scheduled time. The sensors are placed at various locations in the building and all communicate with a central repository which manages the slave nodes. This data centre performs the analysis of the collected data. The data synchronization which is very important considering the huge volume of data and for any possible analysis to be performed on it is nearly perfectly implemented for the system. The modules used are low power thus making it highly energy efficient. The cost of the microcontroller, communication modules and sensor nodes are very feasible cost wise. The system is centralized meaning all the sensor nodes are connected to it and exchange data with it. This could possibly cause a communication overhead and also storage burden due to the large volumes of data generated. The analysis of the entire data collected is the sole responsibility of the data centre which might cause computation overhead and also total failure of the system if the central server fails.

Lei et al [15] have implemented a SHM system for curved girder skew bridge. It uses a data acquisition and processing software called Lab View. The types of monitoring done are environment, static and dynamic monitoring using the respective sensors. The built-in packages in Lab View are integrated with the communication modules to collect data, pre-process it and apply algorithms for data analysis and it makes the entire process more reliable and precise.

## 2.6 Acoustic Emission and Ultrasonic sensor based SHM

Swit et al [16] have proposed an AE based SHM for a cable-stayed bridge in Vietnam. When any micro fracture occurs, energy is released in the form of an AE wave. The sensor captures this wave and converts it to an electrical wave for further analysis. The AE method is capable of identifying earlier stages of damage and enables retrofitting to the existing structures by identifying the matured damage prone regions. There is also a very high possibility of external noises interfering with the sensing system resulting in false alarms. On the other hand, earlier identification helps timely restoration of the existing structure before any serious damage is done.

Patil et al [17] have used a combination of ultrasonic sensors and accelerometers. One sensor node contains both ultrasonic and accelerometer sensors and the other node contains only the accelerometers. The accelerometers measure the tilting angle of the bridge pillar whereas the ultrasonic sensors measure the level of water. The corrosion damages to the structure cause a change to the tilting angle which is measured by the accelerometers. An Android application interface is used to intimate the user of any significant deviation in the sensor readings. The centralized server has many disadvantages; (i) delay in the relay of data packets, (ii) server failure brings the entire system down. But the data transmission packet loss rate is significantly improved in this particular implementation.

Chapter 2 has discussed in detail about the various approaches to SHM using different hardware and software components. Chapter 3 would be reviewing how the analysis of the structure under the impact of several factors is carried out by the simulation in structural analysis software and also sketches the details about the sensors deployed at the site.



### **3. STRUCTURAL ANALYSIS**

Chapter 2 has discussed about the various methods that were used to deploy SHM of bridges. Chapter 3 attempts to outline how structural analysis is performed using Finite Element Analysis (FEA) in a software and briefly describes about the sensors deployed in the project.

Structural analysis is a broad term that often studies the behaviour of any structure under a particular load. The internal forces, deformations, stresses, support reactions, etc. developed as a result would give us an approximation of how good the structure would be under the anticipated loads over a prolonged time. In this case, the effects of load of a moving train are used to study the deformations caused in the bridge and predict durability of its members.

#### **3.1 Finite Element Analysis**

When considering a simple structure, the deformations caused as a result of the applied force can be easily studied. However, for complex structures such as bridges, studying the effects at all regions is often tedious and time-consuming. The sensors which measure deformations, accelerations of nodes of the truss when subjected to a moving trainload, are placed at seven nodes, which are yet to be decided by performing structural analysis of the truss. The entire cross-section of the two leafed truss bridge can be divided into four identical parts, and hence any single part is considered for a rough estimation of deformations performed manually, the results of which can be used as a guide while modelling the whole structure in finite element analysis software.

#### **FEA for Truss Structures**

Finite element analysis is performed using ABAQUS software. The whole model of the bridge in STAAD pro software was given by the SERC authorities

with all the essential material properties and dimensions of all the members. The STAAD pro model has the cross-section properties when the bridge was constructed. With time, as the bridge members get corroded, the effective cross-section area of the members to be considered for analysis reduces, and hence they provide with the data of current cross-section properties. The analysis was done using both the models, and the results were compared. Finite element analysis involves meshing of all members into smaller parts, and all the mesh elements are analysed separately with the appropriate loads and boundary conditions for stresses, strains, and deformations. The elements are then added at the appropriate nodes for it to represent the whole model to get net deformations at the respective nodes.

While performing the analysis manually to get a rough estimate on the deformation values, the symmetry of the bridge can be taken to our advantage, and only one-quarter of the truss system can be analysed, and the combined action of the whole system can be gained by linear superposition as the deformations are very small when compared to the length of the bridge. The difficulty in this analysis is arriving at a suitable boundary condition for the separated part because the actual boundary conditions in the real field are not applicable after separation from the whole model. Dynamic analysis is performed by simulating the motion of a train along the length of the bridge. This simulation was done using Fortran code. After performing analysis, if any of the elements are found to be prone to failure soon, which may so happen by exceeding the permissible stress levels, retrofitting measures are suggested to the Railway authorities, i.e., total replacement of the member or enhancing its strength with an additional support.

## **3.2 Sensors Deployed**

### **Accelerometers**

An accelerometer is an electro-mechanical device that will measure acceleration forces. These forces are caused by moving or vibrating the accelerometer, in this case the moving train. The force caused by vibration or a change in motion i.e., acceleration causes the mass to press the piezoelectric material which produces an electrical charge. The mass of the accelerometers should be significantly smaller than the mass of the system to be monitored. The accelerometer dynamic range should be broader than the expected vibration amplitude range of the sample. The sensitivity of the accelerometer should produce an electrical output compatible with existing instrumentation. Using a low sensitivity accelerometer to measure high amplitude vibrations and conversely using a high sensitivity accelerometer to measure low amplitude vibrations is recommended.

### **Temperature Sensors**

Temperature Sensors measure the amount of heat energy or coldness that is generated by an object or system and produce either an analog or digital output. There are many different types of temperature sensors available and all have different characteristics depending upon their actual application. Contact temperature sensors are most frequently used and are a type of temperature sensors that are required to be in physical contact with the object being sensed and use conduction to monitor the changes in temperature.

### **Strain Gauges**

A strain gauge is a sensor whose resistance varies with applied force. It converts force, pressure, tension, weight, etc., into a change in electrical resistance which can then be measured. When external forces are applied to a static object, stress and strain are the result. Stress is defined as the object's

internal resisting forces, and strain is defined as the displacement and deformation that occur. The strain gauge is one of the most important sensors of the electrical measurement technique applied to the measurement of mechanical quantities. As their name indicates, they are used for the measurement of strain. Strain gauges can be used to measure expansion as well as contraction. The strain of a body is always caused by an external influence or an internal effect. It is also possible to derive the amount or the value of the influencing quantity from the measured strain value. Experimental stress analysis uses the strain values measured on the surface of a specimen, or structural part, to state the stress in the material and also to predict its safety and endurance.

The mechanism behind the structural analysis has been briefly discussed and the characteristic of the wired sensors deployed have been described in Chapter 3. Chapter 4 discusses how the cloud platform is set up to perform authorization and authentication to allow protected access to the data files on One Drive.

## 4. SENSOR DATA RETRIEVAL FROM CLOUD

Chapter 3 outlined how structural analysis is performed using a software and briefly described about the sensors deployed. Chapter 4 discusses about the cloud platform being used, the authorization and authentication protocols followed, role of Azure portal and how the files are downloaded.

The wired sensors set up across multiple locations along the bridge collect the measurements when the train passes over the bridge. The files are transmitted over to a cloud server in binary format as the size of the file is minimal and transmission rates are fast.

### 4.1 Microsoft's One Drive

The cloud platform used is the Microsoft One Drive. It first was launched by Microsoft in August 2007. Users are allowed to store their files and the data can be synced across multiple devices. All the user needs to do is to create a Microsoft account with which he/she can log in to upload, store or download the stored files. One Drive for Business is used for commercial purposes where the organization has to pay a subscription fee according to the plan chosen.

### 4.2 One Drive for Python

A python script is used to download the files from OneDrive. In order to access the resources from One Drive, authorization and authentication have to be performed.

**Authorization:** The authenticated user is often required to perform certain operations such as accessing, reading or writing a file. The user must be *authorized* or in other words delegated the required permissions.

**Authentication:** This is where the users prove their identity or what they claim to be.

### **4.2.1 Microsoft Identity Platform**

Microsoft Identity Platform (MIP) is a platform that allows applications developed to sign into the Microsoft accounts. It obtains tokens from the end point to call APIs that provide the desired permissions to the application it could perform over the resources belonging to that account. Before MIP, Azure Active Directory (Azure AD) Developer Platform v1.0 was deployed for this purpose. MIP evolved from Azure AD. Azure AD accepted work, school azure accounts only whereas MIP was designed to accept the work, school or other organizational accounts as well as personal Microsoft accounts. Azure AD used Azure AD Authentication Library (ADAL) and MIP uses Microsoft Authentication Library (MSAL). OAuth 2.0 and Open ID Connect are the security standards used to achieve the authorization and authentication.

### **4.2.2 Azure Active Directory**

Azure AD is a central identity management platform. MIP delegates the authorization and authentication to Azure AD. The Client app has to be registered with Azure AD so that it can perform authorization and authentication on its behalf.

### **4.2.3 Authentication Flows**

The authentication flow that has to be followed depends on the type of application architecture. There are different types of applications namely web apps, mobile apps, desktop apps, web APIs etc. Here for a desktop app that calls web APIs, authorization code flow is used.

#### 4.2.4 Scopes and Permissions

The app requires permissions for performing operations such as reading a file, writing to a file. The Azure AD requests these permissions to the users or administrators and they can either approve or deny the requests on user's behalf. It is a safe practice for the app developers to request only for the permissions they need for the app to function. The permissions are referred to as scopes and they are represented by a string. The requests for the scopes can be made in the authorization request using the scope parameter.

The Microsoft Identity Platform provides provisioning for the following types of permissions [18]:

***Delegated Permissions:*** The app requires certain permissions to be provided if it has to perform the files uploaded to the cloud. These permissions require a user signed in to the Microsoft One Drive account to consent to the permissions asked for. Some high privileged permissions require the administrator of the account to consent to the permissions. The app which requests the consent may not have more privileges than the user who consents to these permissions

***Application Permissions:*** These permissions don't require a signed in user to consent to the permissions. They are consented by the administrator account

The scopes used for the application are

***Files.Read*** : A delegated permission which doesn't require administrator consent and is required if the app has to read the resources uploaded to the cloud

***Offline\_access*** : It is an OpenID Connect scope which gives the app access to the files on behalf of the user for a prolonged time. This scope enables the

app to retrieve *refresh\_token* from the token endpoint. The refresh tokens don't expire unlike the *access\_token*. When the access token expires, a new one can be obtained using the refresh token

***User.Read*** : A delegated permission that allows the app to read the profile details of the signed in user

#### **4.2.5 Security Tokens**

##### ***Access Tokens***

These security tokens are JSON Web Tokens (JWTs), Base 64 encoded JSON objects signed by Azure. Client receives the access token from the v2.0 endpoint. This is the given in response to the GET request to the endpoint where the Client includes the permissions it requires along with other details. The response also contains information such as the expiry period of the token, the scopes/permissions it has granted the access to and other metadata about the token. The contents of the tokens are called *claims*. The token is split into three parts namely the header, payload and the signature, each separated by a period and Base64 encoded. The Client treats the token as an opaque string.

##### ***ID Tokens***

This security token is used to verify the identity of the user by the Client as a part of the OIDC protocol to enable a single sign-on that can be used across multiple applications. This is returned to the Client along with access token. ID tokens are JWTs such as the access tokens and the claims are the same as that of access token. Here the payload contains the information that the Client wants to know about the user. For the same purpose the scope *User.Read* is requested by the Client and only if the user consents to this scope in the request, the ID token would be issued successfully.



## 4.2.6 Microsoft Graph API

Microsoft Graph API is a RESTful web API that allows users to access the Microsoft Cloud resources. It exposes REST APIs and client libraries to access the data on One Drive. The Client App has to be registered with the Azure AD and Microsoft Graph can be used to access the files stored across in One Drive. The Files API in Microsoft Graph is used for the same purpose. Some of the permissions are *Files.Read*, *Files.ReadWrite*, *Files.Read.All* etc.

## 4.2.7 OAuth 2.0 Authorization Protocol

OAuth 2.0 [19] is a security protocol used solely for the purpose of authorization. It gives an application i.e., the python script that downloads the files, permission to access the resources stored in One Drive on behalf of the user without the user having to give his or her credentials. The app has to post a HTTP request to the authorization endpoint of MIP to receive the authorization code and exchange this code with the token endpoint of MIP to receive access token which provides access to the Microsoft Graph API protected Azure AD. The above mentioned steps are illustrated in detail in Figure 4.1.

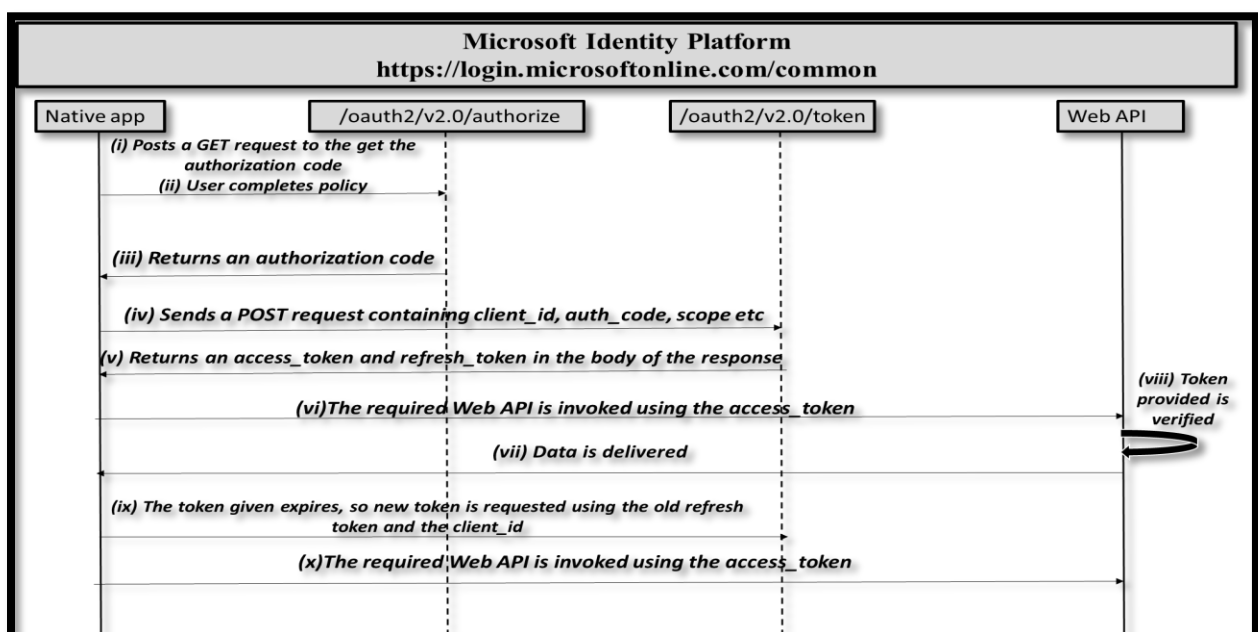


Fig 4.1. OAuth 2.0 Authorization Protocol

The Resource Owner in this protocol is the user who owns the data One Drive. Client is the app created with Azure AD which requires APIs to access the data. Authorization Server here is the MIP which is capable of providing and revoking access to resources and also issue tokens. Finally, the Resource Server is the platform where the data resources are hosted which is the One Drive. The interactions between the above entities of OAuth 2.0 protocol and how tokens are exchanged from endpoints are illustrated in Figure 4.2

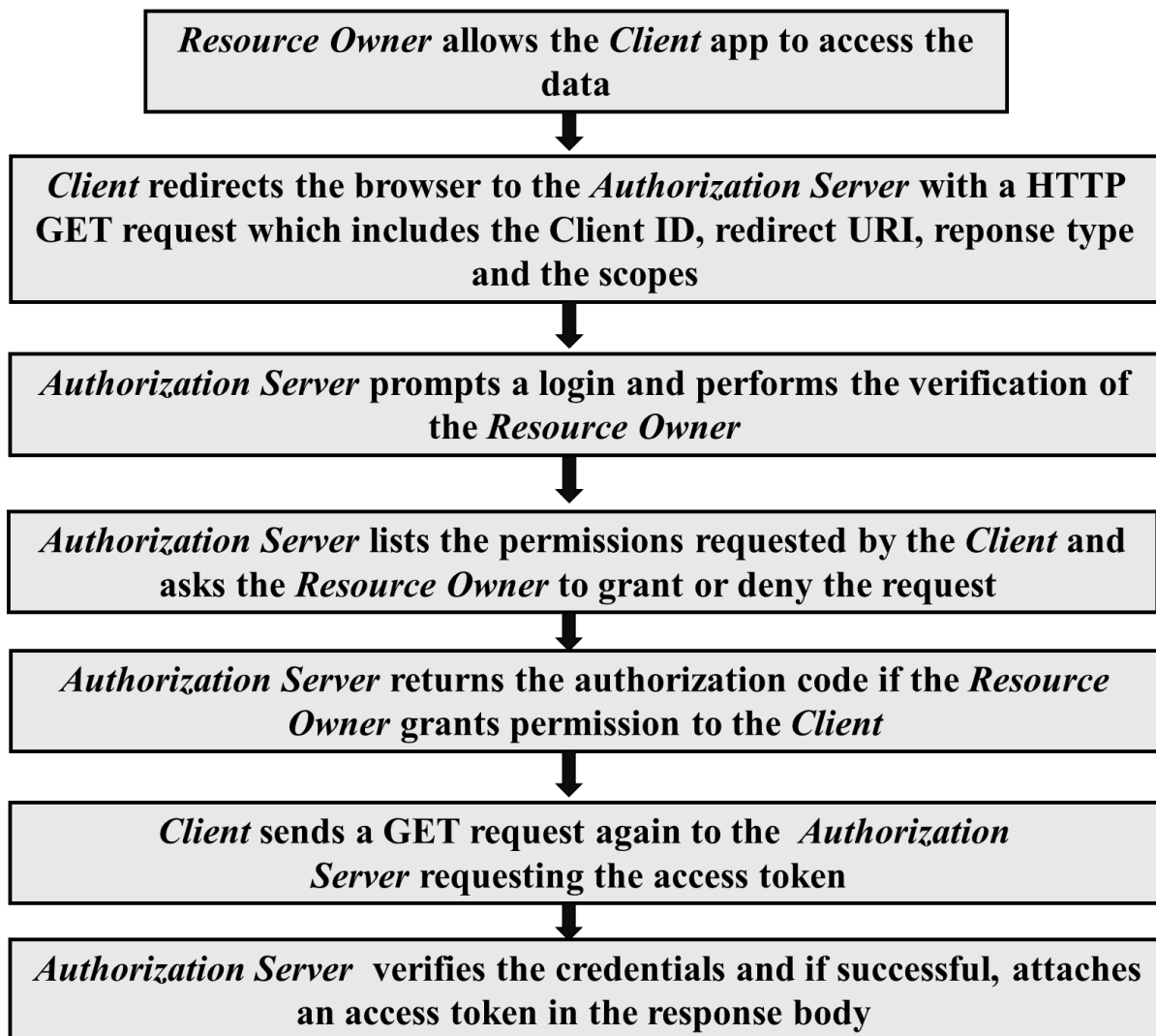


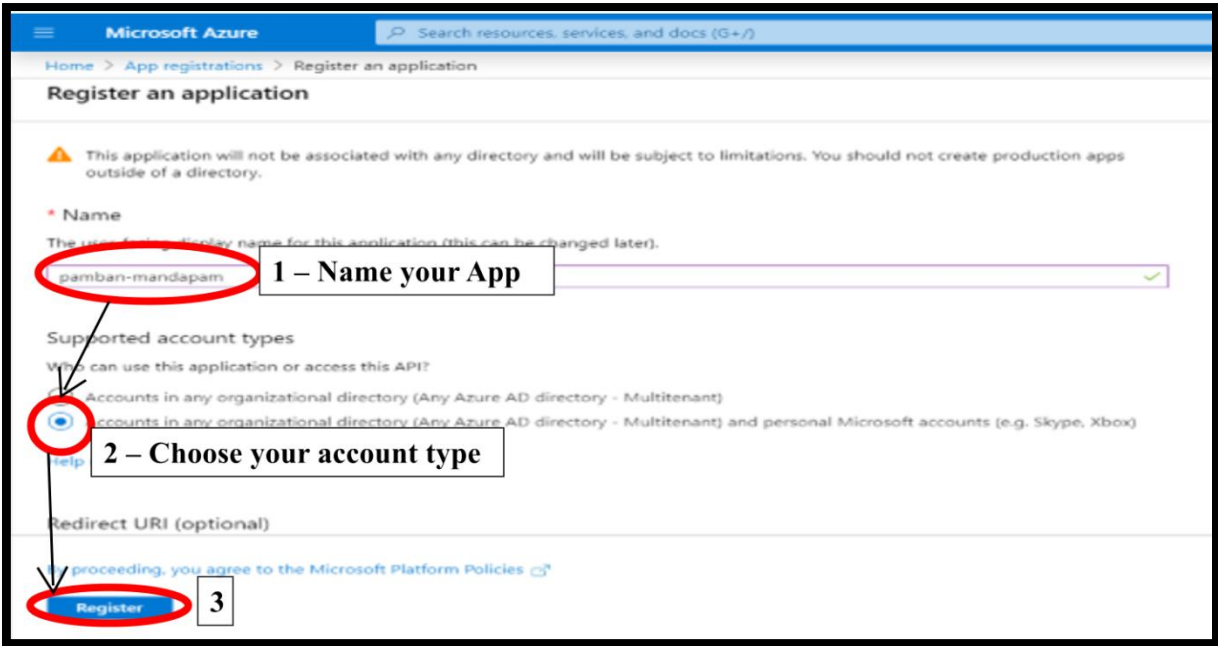
Figure 4.2. Workflow of OAuth 2.0 Authorization protocol

#### **4.2.8 OpenID Connect (OIDC)**

OAuth 2.0 only provides authorization not authentication. OIDC is used for this purpose. It is an authentication protocol heavily based on and is an extension of OAuth 2.0 protocol [20]. It ensures that a user signs into a web application securely. It differs from OAuth 2.0 mainly in the use of an ID token. This ID token helps the Client app to verify the identity of the user by reading the profile details of the user. This in turn enables the user to experience a Single sign-on (SSO) at the Client. The ID Token is typically a JWT. The Client app is able to acquire an access token as it is an extension of OAuth 2.0 protocol. The OIDC varies from the OAuth in the scopes mentioned in the initial GET request. The *offline\_access* scope of OIDC is used here. In the HTTP response, an access token and an ID token is returned.

#### **4.2.9 OAuth Authorization Code Flow**

Before the authorization code flow starts, the Client app has to be registered with the Azure AD and the necessary configurations for the app. The app name is given, supported account types are selected and the point from where tokens are to be received and requests are to be submitted i.e. redirect URI is selected as shown in Figure 4.3.

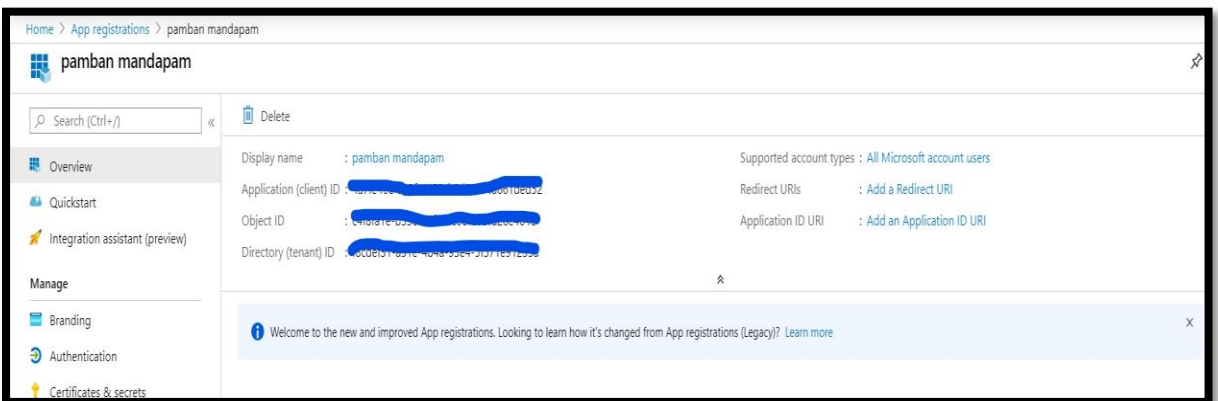


**Figure 4.3. Client App Registration on Azure AD**

Client ID, an object ID and a directory ID are created which is noted down for obtaining the tokens at a later point. The details of the app created are displayed on the Azure Dashboard which is shown in Figure 4.4.

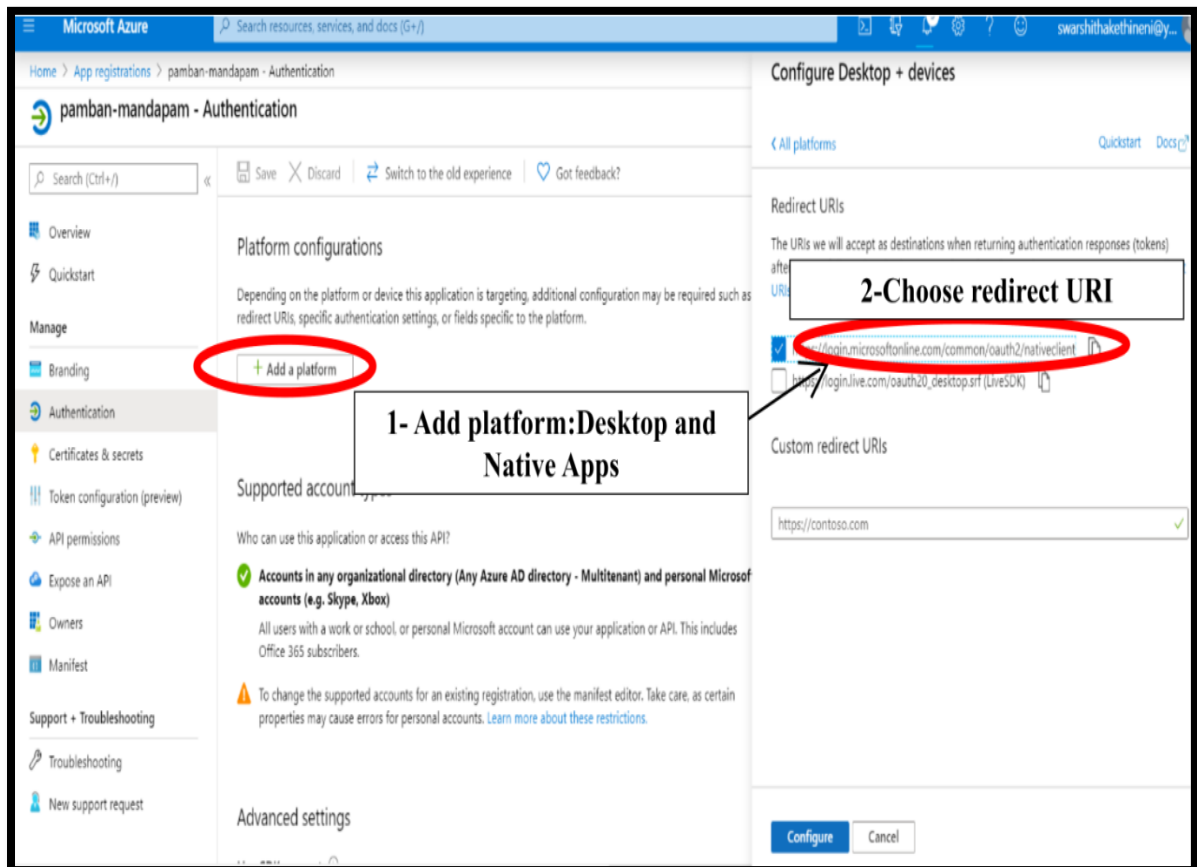
*Client ID:* The Application ID assigned to the app by the Microsoft Azure portal

*Directory ID:* The tenant ID assigned to the user who is the owner of the app which is useful when there are multiple tenants



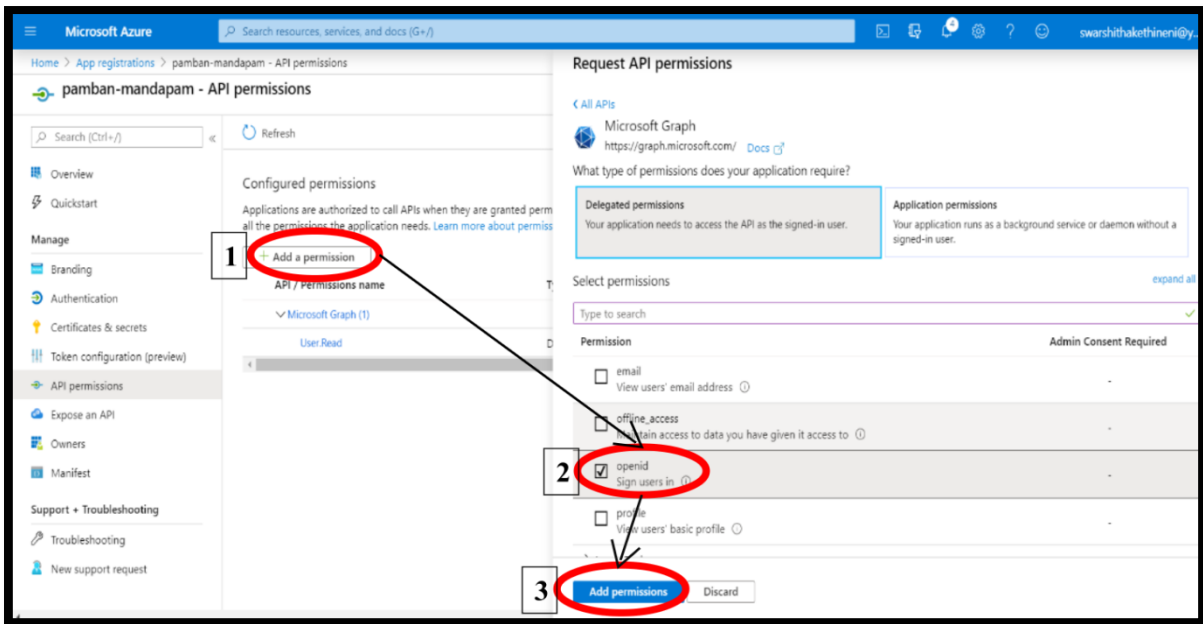
**Figure 4.4. Client App Details on Azure AD Dashboard**

Under the authentication tab, the devices on which the app would be used are selected and any default redirect URI provided is selected for use. Redirect URI is typically where the app can send and receive authentication requests and responses. The *redirect\_uri* is assigned by default according to the platform the app is used on. The process of configuring the *redirect\_uri* is illustrated in Figure 4.5



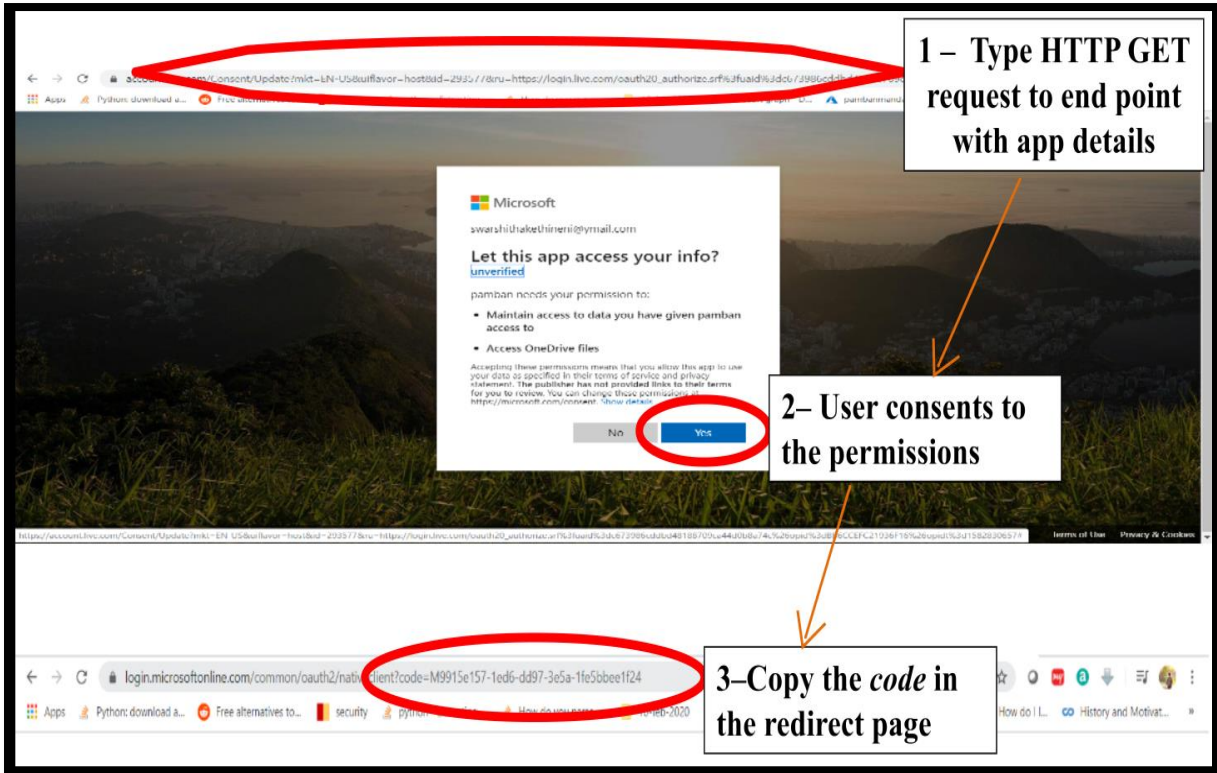
**Figure 4.5. Configuring Platform and Choosing Redirect URI on Azure AD**

Under the API permission tab, according to the operation that would be performed over the files, the required APIs in Microsoft Graph are chosen as shown in Figure 4.6.



**Figure 4.6. Adding Scopes Required to Request User’s Consent on Azure AD**

If the permissions require admin consent, it has to be approved by the admin. To obtain the authorization code or the *auth\_code*, a GET request is made with the redirect URI with parameters such as the *client\_id*, scopes. In the response, the *auth\_code* is returned which is saved to obtain the access token in the next step. The process of submitting the request and obtaining the token from the endpoint is illustrated in Figure 4.7.



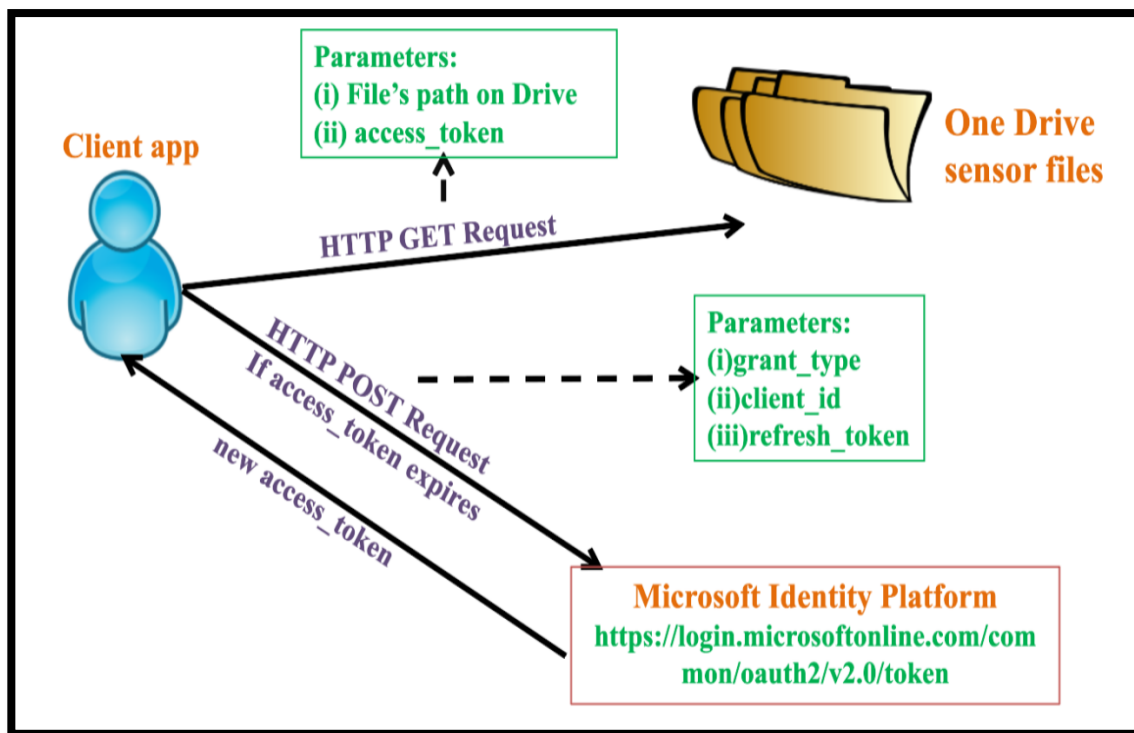
**Figure 4.7. Token Returned by the Authorize Endpoint**

The user is requested to sign in at this step to verify his/her credentials to proceed to get the *auth\_code*. A successful response returns two parameters, the code and state. The former contains the authorization code and the latter is a verification parameter for the app to check if the values in it are identical during both the request and response. If the response is unsuccessful, an error code and the error description are returned. Now a POST request of type *FORM URL Encoded* is made where the body contains the following parameters (i) *grant\_type*, (ii) *client\_id* and (iii) *code*. The body of the response contains an access token and a refresh token which expires in the time specified in the *expiry\_period* in the response body which is shown in Figure 4.8.





downloaded is achieved by posting a GET request containing the path of the file and the access token. A folder is generated each day which contains the date it was generated. The system's date is computed and only the folder containing the files of that specific date are downloaded into the local computer into folders organized by date and type. Initially, a GET request to search the files in the drive containing today's files is made. The files are stored in a list and these files alone are downloaded everyday by making a GET request again instead of downloading all the files in the drive folder for efficient storage. The entire process required to download the files is summarized in Figure 4.9



**Figure 4.9. Data Retrieval using Python**

Chapter 4 had discussed how to download the data files from One Drive and the Chapter 5 discusses how data is formatted and stored everyday automatically for the files to be analysed and finally how the results of the data analysis are used to send out alert notifications.

## 5. DATA HANDLING AND STORAGE

The Chapter 4 discussed in details about the authorization and authentication protocols required to download the files from One Drive securely. Chapter 5 aims to describe how the data downloaded is formatted and stored to the system for consequent analysis.

Binary files containing the sensor measurements are downloaded, each for accelerometer, strain values and temperature values. The binary file contains header information followed by the readings. A part of the header information is embedded within Extensible Mark-up Language (XML) tags that are extracted and then the binary encoded sensor readings are obtained.

### 5.1 Python Libraries

#### 5.1.1 struct

This python package is used for handling the binary data stored in files. It uses format strings to specify how the binary data has to be structured.

```
struct.unpack(format_string, binary_data_buffer)
```

The above function is used to unpack the bytes into the desired format specified by the format string which is a string representing the way the binary data was packed into. The size of the format string in bytes is found out using the *struct.calcsize(format\_string)* which should be equal to the buffer size which contains the binary data. The result is stored in a tuple. The structure of the binary file i.e., the way the bytes are organized into data with its corresponding encoding format is exported into a separate file which is shown in Table 5.1. This file is used as reference to decode the binary data.

Short	Channel location in catman database (0,1,2...)
Long	Channel length (Number of samples) = NVals[i]
Short	Length L of channel name
Byte	Channel name (L Bytes)
Short	Length L of unit
Byte	Unit (L Bytes)
Short	Length L of channel comment
Byte	Channel comment (L Bytes)
Short	Format: 0=numeric (double), 1=string, 2=binary object, 4=numeric (float)
Short	Data width in Bytes (for numeric format always=8, for string>=8)
Double	Date and time of measurement (NOW format)
Long	Size of the extended channel header in Bytes
VB_DB_CHANHEADER (148 Byte)	Extended channel header (Traceability data) - description see below
Byte	Linearisation mode (0=none, 1=Extern Hardware, 2=user scale, 3=Thermistor)
Byte	User scale type (0=Linearisation table, 1=Polynom, 2=f(x), 3=DMS scale)
Byte	Number of points for user scale linearisation table (nScalData)

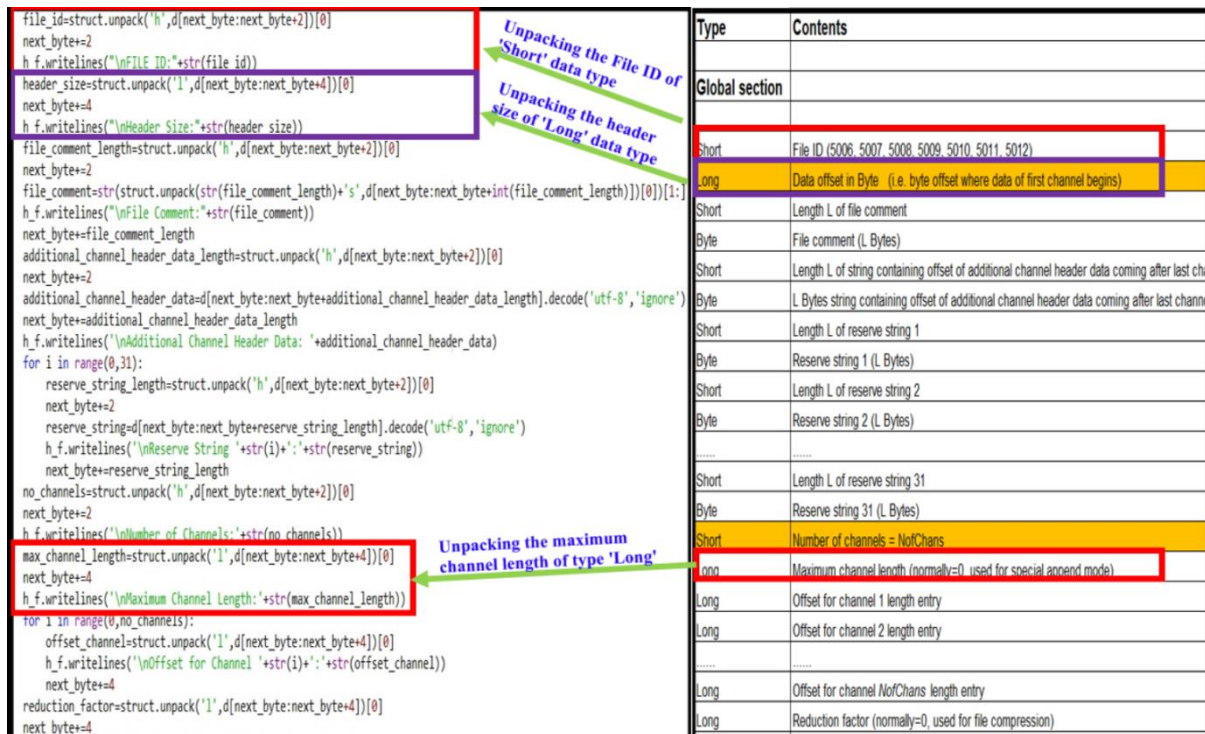
**Table 5.1 Binary File Structure Description**

The encodings such as short, long, double etc. should be specified in the format string. The format strings used for the code are mentioned below in Table 5.2 with its size calculated using *struct.calcsize()*. The size of the binary data buffer should match the size of the format string for the unpacking to be successful.

Format Strings	Data Type	Size in bytes
'h'	Short Integer	2
'l'	Long Integer	4
'f'	Single-precision Float	4
'd'	Double-precision Float	8
'c'	Character	1
's'	Character array/String	Length of the string

**Table 5.2. Format Strings of struct with its Corresponding Size**

The conversion performed for the corresponding encoded information given in the description file is shown in Figure 5.1.



**Figure 5.1 Header Data Conversion**

### 5.1.2 xml.etree.ElementTree

The header information is present partly as XML data which is parsed using this package. The entire XML data is structured into a tree where each element represents a node in the tree. The XML data is stored into a string and then iteratively processed for each node. The tag, attribute and text value are printed as it traverses hierarchically. Importing the XML data from a string and creating a tree object is illustrated below.

```
xmlTree=ET.ElementTree(ET.fromstring(xmlstring))
```

Hierarchically traversing through the nodes is illustrated in the below code.

```
for elem in xmlTree.iter():
    if (elem.tag):
        if(elem.attrib):
            xml_write.write(elem.tag+"="+str(elem.attrib))
        else:
            xml_write.write(elem.tag )
    if (elem.text):
        xml_write.write(":"+elem.text)
```

## 5.2 Storage of Raw and Processed Sensor Files

The files are downloaded into time stamped folders. The files generated only on that day have to be retrieved for further analysis. The files generated one a single day are taken as input for formatting and stored into the desired file hierarchy for ease of access in the future. The binary files' name contains the date it was generated on and type of reading. Essentially the file hierarchy is created from the file's name.

We use *os package* in python for creating the desired directories.

```
if not os.path.exists(directory_path):  
    os.makedirs(directory_path)
```

## 5.3 Executing the Analysis code

MATLAB script is written which analyses the sensor readings and checks if the values lie within the optimal range. The values are compared with predefined threshold values. Running this MATLAB program produces two files. One file is the analysis report which is to be mailed and the other is the file which has a flag variable that would be set if the values are critical and also contains the message indicating the region where the values exceeded the normal range which is to be sent to the concerned via Short Message Service (SMS). The script has to be run from Python. MATLAB provides packages for Python to call the software as a computational engine.

### **System Requirements:**

Python versions 2.7, 3.6 and 3.7 are supported. The architecture of the MATLAB software must match with that of Python.

### **Installing the MATLAB Engine API:**

In the MATLAB command prompt, the following commands have to be typed,

```
cd (fullfile(matlabroot, 'extern', 'engines', 'python'))
system('python setup.py install')
```

### **Starting and stopping the MATLAB Engine:**

For this the matlab.engine package has to be imported, then the engine can be started with the following command,

```
eng=matlab.engine.start_matlab()
```

The engine can be stopped using,

```
eng.quit()
```

The MATLAB script can be called using,

```
eng.matlab_script_name(nargout=0)
```

## **5.4 Notification System**

### **5.4.1 Mailing the Analysis Report**

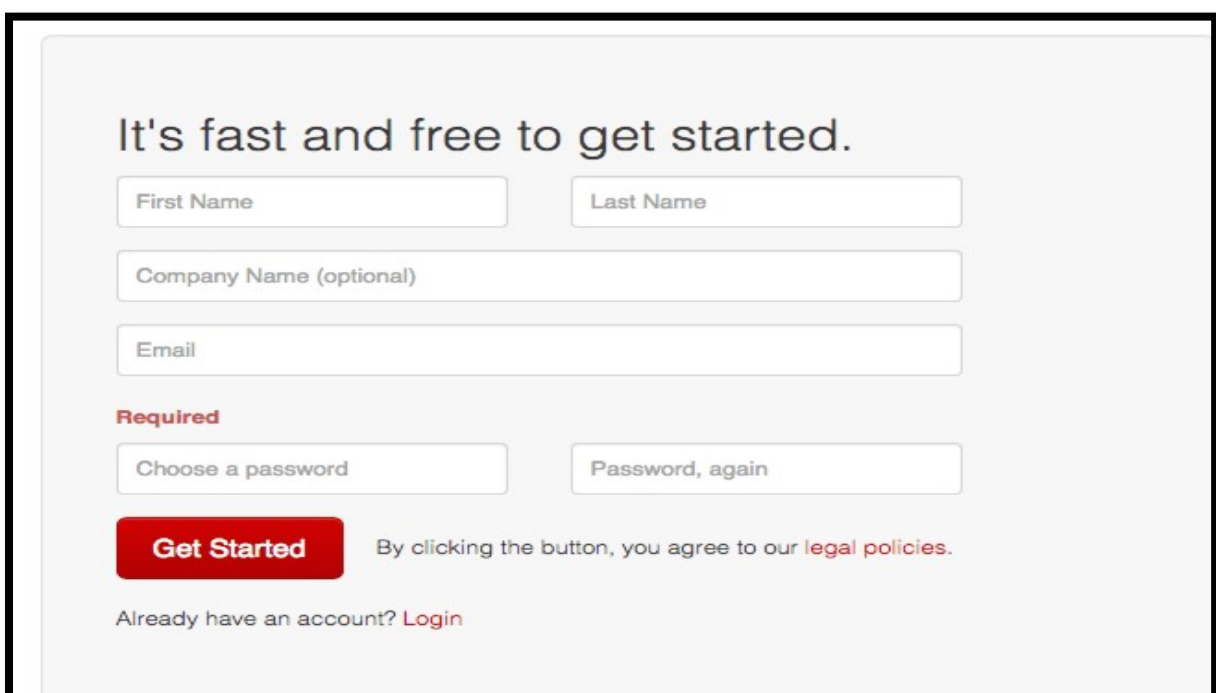
The analysis file which is generated after reading the sensor measurements is mailed via a python script to the concerned who could take necessary measures to inspect the site of potential damage. *smtplib* module from python is used which can send email to any computer device with an internet connection.

The steps involved are as follows. The smtplib module from Python has to be downloaded. This has to be followed by the creation of a smtp session by passing the mail server and the port number. The *starttls()* function is then called to ensure the information being exchanged is encrypted for security purposes. For the authentication, the email-id and the password have to be provided. After authentication, the message to be sent is now attached and sent.

The mail settings have to be modified to provide access to the Python script to access the mail account.

### 5.4.2 Alert Messages via SMS

For sending SMS via python to a registered mobile number, SMS APIs have to be used. Twilio is one such SMS API which provides python helper library developed by them. The steps involved are illustrated below. A Twilio account has to be created by typing in the details as shown in Figure 5.2.



The image shows a registration form for Twilio. At the top, it says "It's fast and free to get started." Below this are several input fields: "First Name", "Last Name", "Company Name (optional)", and "Email". Underneath these is a "Required" section with two password fields: "Choose a password" and "Password, again". A prominent red "Get Started" button is located below the password fields. To the right of the button, there is a line of text: "By clicking the button, you agree to our legal policies." At the bottom left of the form, there is a link that says "Already have an account? Login".

**Figure 5.2. Twilio Account Creation**

Twilio issue phone number with SMS capabilities has to be purchased with plans as depicted in Figure 5.3.

Search Term Match: First part of number		Type: All	Requirement: Any				<a href="#">Show Advanced Search</a>
NUMBER	TYPE	CAPABILITIES				PRICE	
		VOICE	SMS	MMS	FAX		
<b>+1 (312) 553-6965</b> CHICAGO, IL	Local					<b>\$1.00</b> monthly	<input type="button" value="Buy"/>
<b>+1 (312) 667-3017</b> CHICAGO, IL	Local					<b>\$1.00</b> monthly	<input type="button" value="Buy"/>
<b>+1 (312) 313-7274</b> CHICAGO, IL	Local					<b>\$1.00</b> monthly	<input type="button" value="Buy"/>

**Figure 5.3. Twilio Issue Phone Number Purchase**

On purchase, the Account SID and the Auth Token would appear on the Twilio dashboard which has to be noted for later use in the Python script. This is illustrated in Figure 5.4.

The screenshot shows the Twilio Console Dashboard. On the left is a navigation menu with 'Home' selected. The main content area is titled 'Console Dashboard' and contains an 'Account Summary' section. In this section, the 'Account SID' and 'Auth Token' are displayed. The 'Account SID' is a long alphanumeric string, and the 'Auth Token' is a long alphanumeric string with a lock icon next to it. Both are circled in red. An arrow points from the text 'Copy this token into your code:' to the Account SID. Another arrow points from the text 'Click the lock and then copy this token into your code:' to the Auth Token. Below the Auth Token, there is a 'Balance' section showing 'Auto Recharge is OFF'. At the bottom, there is a 'Recently Used Products' section with three cards: 'Phone Numbers', 'Programmable SMS', and 'Programmable Voice'.

**Figure 5.4 Account Details Displayed on the Twilio Dashboard**

Initially, the Python helper library for Twilio has to be installed.

```
(base) C:\Users\Chiranjeevi> pip install twilio
```



In the python script, client session is established, passing the *account\_sid* issued and the *auth\_token* as its arguments.

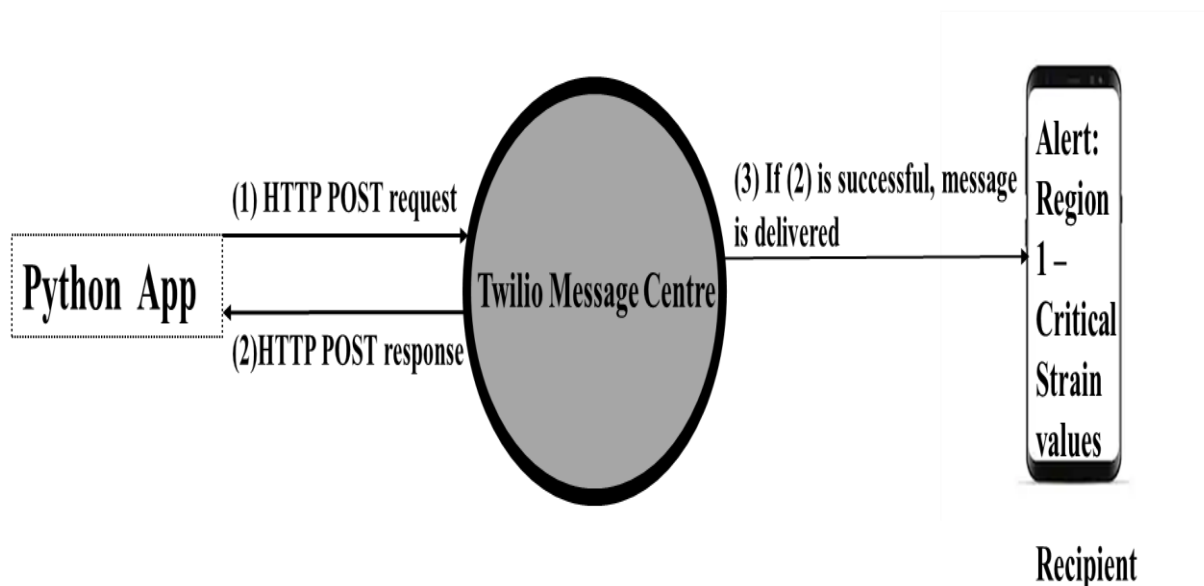
```
client = Client ( account_sid , auth_token )
```

The message is sent by inserting the message text and from and to phone number.

```
message=client.Messages.create(body=message_body_from_file,from_='+12564488661' , to= '+919962602318')
```

### Working

The python script makes a HTTP POST request to Twilio's message centre. Twilio authenticates the credentials provided in the script. If the authentication is successful, the SMS is sent to the recipient and a HTTP response containing the message delivery details is returned. If the authentication was not successful a HTTP response is sent which contains the error message. The above mechanism is depicted in Figure 5.5.



**Figure 5.5 Twilio SMS Sending Mechanism**

## 5.5 Automation

The sensors record data multiple times a day whenever the train runs over the bridge. It is unreliable to have a person download the files, run the code and retrieve the results. So, the code has to be run at the end of every day automatically. Also, the code has to be run on boot up of the system eliminating the need for any user intervention.

### 5.5.1 Code Automation

The system's date is retrieved using the *datetime* package and using that any folder with the date is matched and the files inside those folders are used for processing. This process is done at 23:00 hours every day. The module from python library that is used is *schedule*.

```
schedule.every().day.at("23:30").do(script_to_be_run,function_parameters)
```

### 5.5.2 Running on Boot-up

The code is required to run on windows start up. This enables the program to be run every time the system starts up. The steps are as follows. In the Users folder, *Hidden Apps* on top of the menu bar of the file explorer is clicked. The *App Data* is selected and later navigated to the following directory and the Python files are stored here.

*/AppData/Roaming/Microsoft/Windows/Start Menu/Programs/Startup/*

### Windows Registry

The Windows Registry is a collection of databases of configuration settings for Microsoft Windows operating systems. The Windows Registry stores much of the information and settings for software programs, hardware devices, user preferences, and operating system configurations. The Windows Registry is accessed and configured using the Registry Editor program, a free registry

editing utility included by default with every version of Microsoft Windows going back to Windows 95. Registry Editor can be accessed by executing *regedit* from the Command Prompt. Registry Editor allows viewing and making changes to the registry. The registry is the collective name for various database files located in the Windows installation directory.

The registry contains registry values, located within registry keys, all within one of several registry hives. Making changes to these values and keys using Registry Editor change the configuration that a particular value controls. The registry contains two basic elements: keys and values. Registry keys are container objects similar to folders. Registry values are non-container objects similar to files. Keys may contain values and sub keys. Keys are referenced with a syntax similar to Windows' path names, using backslashes to indicate levels of hierarchy. Keys must have a case insensitive name without backslashes.

The script has to be added to the Windows Registry and this is also done using a Python script where the windows registry key has to be edited. Registry has a list of programs that must be executed when the particular user logs onto the system. The Python script that has to be automated along with its path must be added to the registry. The Python code which adds the script to the windows registry should be run as the administrator when it is done for the first time.

## **6. Conclusions and Future Work**

The system proposed was successfully able to retrieve the binary sensor measurement readings using the Microsoft's cloud set up. The binary files are much compressed than the other default encoded files. This saves the uploading and downloading time of the files. The data conversion then takes place at the local computer where the header information is also extracted and processed separately. The converted data values are analysed further to check if they lie in the optimal range. If not, an alert system was developed to intimate the railway officials and the concerned for further measures to be taken. This greatly avoids the need for manual inspection at sites where such tasks are difficult to be carried out. Further the location that has to be assessed is also intimated such enabling localized damage detection.

An efficient mechanism that downloads the files automatically as soon as it becomes available to process and analyse the conditions on the go could be integrated with the existing modules. This effectively would make the analysis and notification modules more dynamic and relevant to the needs. Modules that enable a more compressed data transmission can bring down the overall processing time. These changes could be added to the existing modules with minor modifications and they help achieve the same objectives as the original system but with an increased efficiency and decreased processing time and usage of resources.

## Appendix

### Appendix A

#### Module 1: *access\_token.py*

This script gets the access token which is required to access the files on One Drive any time. The requests package has to be imported from Python which is used to send an HTTP POST request of type *Form URL Encoded* to the end point to get the access\_token.

```
import requests
```

requests.post accepts 2 parameters (i) the token end point to which the request has to be made, (ii) *data* parameter which accepts a dictionary where the app details are passed, grant type and chosen redirect URI. The *data* parameter is defined this way

```
params = {  
    'grant_type': 'authorization_code',  
    'client_id': 'client_id_assigned' ,  
    'redirect_uri': 'https://login.microsoftonline.com/common/oauth2/nativeclient',  
    'code': 'auth_code_in_GET_response'  
}
```

The HTTP POST request is made and response is stored in 'response' variable.

```
response=requests.post('https://login.microsoftonline.com/common/oauth2/v2.0/ token' , data=params)
```

The refresh\_token returned is stored as illustrated below.

```
refresh_token = response . json () [ ' refresh_token ' ]
```

access\_token() is defined which refreshes the access token using the refresh token. Again, a HTTP POST request is made to the end point to get the new access token. Initially the *data* parameter is defined, the request is made and the new access token and refresh token is stored. The new access token is returned to download.py which calls it.



## Appendix B

### Module 2: *download.py*

Module which uses the valid access token to have continued access to the resources uploaded to the cloud and downloads the sensor files to the local computer. The present day's files are searched in the One Drive's directory by making a get request to the end point and passing the access token received from *access\_token()*. For this, the 'header' parameter is defined, as follows.

```
header = { 'Authorization' : 'Bearer' + access_token () }
```

Then, the system's date is obtained using the datetime object. The *datetime* package is imported and system's date is computed.

```
import datetime
date=datetime . datetime .now() . strftime ("%Y-%m-%d")
```

The One Drive directory is searched by making a GET request to the end point for the present day's files.

```
response=requests.get ('https://graph.microsoft.com/v1.0/me/drive/root /
search (q='\ ' +date+'\ ')?select=name,id,webUrl',headers=header)
```

Retrieving the names of those files from the response object and appending them to a list *files\_today*,

```
result=response.json()value=result [ 'value ' ]
#append the list of today 's files to a list , file_today
files_today =[]
for i in value :
    files_today .append( i [ 'name' ])
```

The file names are passed to the end point and are downloaded and are stored to the respective directories.

```
for i in files_today :
#HTTP request to get the content of files whose name if present in
'files_ today'
```

```

response = requests.get ('https ://graph.microsoft.com/v1.0/me/drive/
root :/ ' +i+ ':/content ',headers=header)
#creating the file hierarchy to store the copied file content (
hierarchy is derived from the file name)
# splitting the file name to create the hierarchy
base= 'E:/ ' +i.replace( '_' '/' ) . split ('.')[0]+ '/'
#creating three folders inside the hierarchy to store the original file
, the converted header and the converted data file respectively
raw=base+ 'raw' header=base+ 'header' data=base+ 'data'
if not (os.path.exists(raw)):
    os.makedirs(raw)
if not (os.path.exists(header)):
    os . makedirs (header)
if not (os . path . exists(data)) :
    os . makedirs ( data )
#creating file object for storing the original file 's contents in
'Write Binary' mode
file=open(os.path.join(raw,i ) , 'wb')
#writing the contents to the file
file.write (response.content )
#closing the file object
file.close ()

```

Running the Python script, creates folders and places the downloaded files into them which is shown in Figure B.1

The screenshot shows a Windows File Explorer window with the address bar set to 's PC > New Volume (E:) > Mandapam > Tmp > 2020-04-14 >'. The main area displays a table of files and folders:

Name	Date modified	Type
data	4/14/2020 12:10 AM	File folder
header	4/14/2020 12:10 AM	File folder
raw	4/14/2020 12:10 AM	File folder

**Figure B.1. Files Created Automatically After the Files are Dumped to One Drive**



## Appendix C

### Module 3: *header\_processing.py*

This python script defines a function *header\_convert()* which converts the header information. The structure of the binary file is provided in binary format file which contains the byte by byte description of the binary file. *struct* package from Python is used extensively for unpacking the bytes into the respective format specified.

```
def header_convert( file_path , file_name , header_path , data_path ) :
file=open( file_path , 'rb ' )
d=file . read ( )
h_f=open(os . path . join ( header_path , file_name+ '_header . asc ' ) ,
'w')
#variable that points to the next byte to be converted
next_byte=0
#file_id attribute decoding
file_id=struct . unpack( 'h' ,d[ next_byte : next_byte +2][0]
#incrementing next_byte
#writing the converted data to the file
h_f . writelines ( 'File ID: '+str ( file_id ) )
```

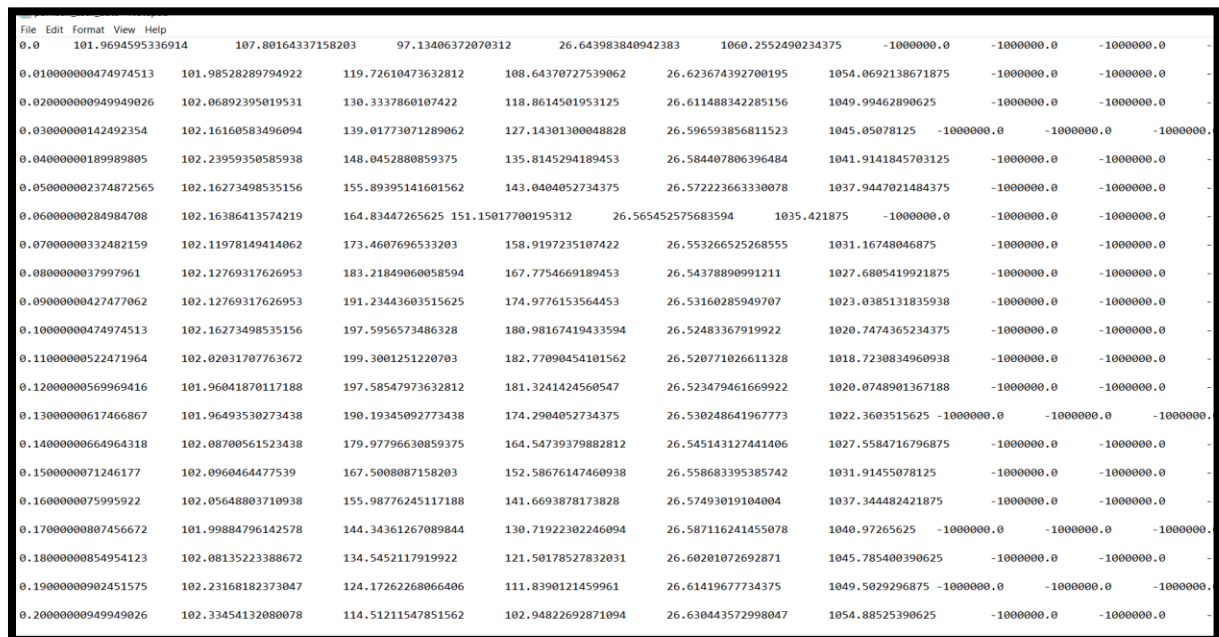
## Appendix D

### Module 4: format.py

Module which keeps converts the binary data values to ASCII format. It stores the converted files into folders sorted by date. The segment that performs the data conversion is shown below.

```
for i in range(header_size, total_size, struct_size):
    #unpacks the bytes from the binary data to double precision float
    values
    converted_column= struct.unpack(format_string,
    binary_data[i:i+struct_size])
    #appending the converted data by column to the list
    converted.append(converted_column)
```

The converted output file is displayed in Figure D.1.



0.0	101.9694595336914	107.80164337158203	97.13406372070312	26.643983840942383	1060.2552490234375	-1000000.0	-1000000.0	-1000000.0
0.010000000474974513	101.98528289794922	119.72610473632812	108.64370727539062	26.623674392700195	1054.0692138671875	-1000000.0	-1000000.0	-1000000.0
0.020000000949949026	102.06892395019531	130.3337860107422	118.8614501953125	26.611488342285156	1049.99462890625	-1000000.0	-1000000.0	-1000000.0
0.03000000142492354	102.16160583496094	139.01773071289062	127.14301300048828	26.596593856811523	1045.05078125	-1000000.0	-1000000.0	-1000000.0
0.04000000189989805	102.23959350585938	148.0452880859375	135.8145294189453	26.584407806396484	1041.9141845703125	-1000000.0	-1000000.0	-1000000.0
0.050000002374872565	102.16273498535156	155.89395141601562	143.0404052734375	26.572223663330078	1037.9447021484375	-1000000.0	-1000000.0	-1000000.0
0.06000000284984708	102.16386413574219	164.83447265625	151.15017700195312	26.565452575683594	1035.421875	-1000000.0	-1000000.0	-1000000.0
0.07000000332482159	102.11978149414062	173.4607696533203	158.9197235107422	26.553266525268555	1031.16748046875	-1000000.0	-1000000.0	-1000000.0
0.0800000037997961	102.12769317626953	183.21849060058594	167.7754669189453	26.54378890991211	1027.6805419921875	-1000000.0	-1000000.0	-1000000.0
0.09000000427477062	102.12769317626953	191.23443603515625	174.9776153564453	26.53160285949707	1023.0385131835938	-1000000.0	-1000000.0	-1000000.0
0.10000000474974513	102.16273498535156	197.5956573486328	180.98167419433594	26.52483367919922	1020.7474365234375	-1000000.0	-1000000.0	-1000000.0
0.11000000522471964	102.02031707763672	199.3001251220703	182.77090454101562	26.520771026611328	1018.7238834960938	-1000000.0	-1000000.0	-1000000.0
0.12000000569969416	101.96041870117188	197.58547973632812	181.3241424560547	26.523479461669922	1020.0748901367188	-1000000.0	-1000000.0	-1000000.0
0.13000000617466867	101.96493530273438	190.19345092773438	174.2904052734375	26.530248641967773	1022.3603515625	-1000000.0	-1000000.0	-1000000.0
0.14000000664964318	102.08700561523438	179.97796630859375	164.54739379882812	26.545143127441406	1027.5584716796875	-1000000.0	-1000000.0	-1000000.0
0.1500000071246177	102.0960464477539	167.5008087158203	152.58676147460938	26.558683395385742	1031.91455078125	-1000000.0	-1000000.0	-1000000.0
0.1600000075995922	102.05648803710938	155.98776245117188	141.6693878173828	26.57493019104004	1037.344482421875	-1000000.0	-1000000.0	-1000000.0
0.17000000807456672	101.99884796142578	144.34361267089844	130.71922302246094	26.587116241455078	1040.97265625	-1000000.0	-1000000.0	-1000000.0
0.18000000854954123	102.08135223388672	134.5452117919922	121.50178527832031	26.60201072692871	1045.785400390625	-1000000.0	-1000000.0	-1000000.0
0.19000000902451575	102.23168182373047	124.17262268066406	111.8390121459961	26.61419677734375	1049.5029296875	-1000000.0	-1000000.0	-1000000.0
0.20000000949949026	102.33454132080078	114.51211547851562	102.94822692871094	26.630443572998047	1054.88525390625	-1000000.0	-1000000.0	-1000000.0

Figure D.1. ASCII Converted Data File

## Appendix E

### Module 5: mail.py

Module that mails the analysis report which is a PDF file obtained after running the MATLAB analysis code. The code segment that attaches the PDF file to the mail and sends it to the intended destination is shown below.

```
s=smtplib.SMTP('smtp.gmail.com',587)
s.starttls()
with open(password_path) as f:
    config = json.load(f)
    s.login('me@gmail.com', config['password'])
    msg = MIMEMultipart()
    message = '{message}\nSend from Hostname: {gethostname()}'
    msg['Subject'] = subject
    msg['From'] = sender_email
    msg['To'] = destination_email
    msg.attach(MIMEText(message, "plain"))
    with open(path_to_pdf, "rb") as f:
        attach = MIMEApplication(f.read(),_subtype="pdf")
        attach.add_header('Content-
Disposition','attachment',filename=str(path_to_pdf))
    msg.attach(attach)
    s.send_message(msg)
```

The snapshot of the attachment received by the receiver is displayed below in Figure E.1.



**Figure E.1. Mail Attached with the Analysis Report Received by the Intended**

## Appendix F

### Module 4: *send\_sms.py*

Module which sends an SMS to the concerned if the sensor readings exceed the optimal range. The code segment where the session is created and the message is sent is illustrated below.

```
account_sid= 'sid_given_after_account_registration'  
auth_token= 'user's_auth_token'  
client= Client(account_sid, auth_token)  
message= client.messages.create (  
    body=message_to_be_sent,  
    from=twilio_purchased_number  
    to=receiver's_mobile_number  
)
```

The snapshot of the SMS received on the receiver's mobile is displayed in Figure F.1.

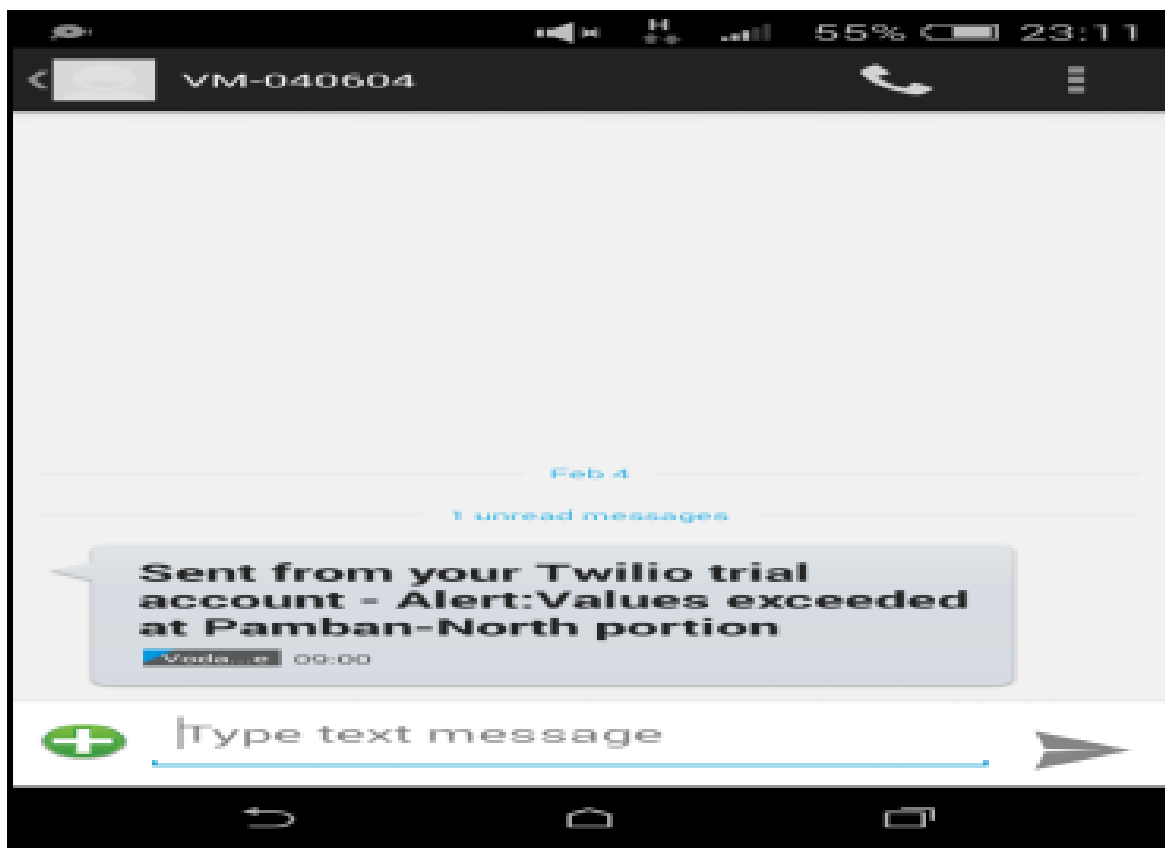


Figure F.1. The SMS Received by the Intended Displaying the Alert Message

## References

- [1] Gastineau, Andrew & Johnson, T & Schultz, Arturo, “Bridge Health Monitoring and Inspection – A Survey of Methods,” University of Minnesota, 2009.
- [2] De Roover, C., Vantomme, J., Wastiels, J. and Taerwe, L. “DIC for Deformation Assessment: A Case Study.” *European Journal of Mechanical and Environmental Engineering*. March 2003. pp. 13-19.
- [3] Scott, M., Rezaizadeh, A., Delahaza, A., Santos, C., Moore, M., Graybeal, B. and Washer, G. “A Comparison of Nondestructive Evaluation Methods for Bridge Deck Assessment.” *NDT&E International*. Vol 36. 2003. pp. 245-255.
- [4] Casas, J. and Cruz, P. “Fiber Optic Sensors for Bridge Monitoring.” *Journal of Bridge Engineering*. Vol. 8, No. 6, 2003. pp. 362-373.
- [5] Iyer, S., Schokker, A. and Sinha, S. “Ultrasonic C-Scan Imaging: Preliminary Evaluation for Corrosion and Void Detection in Posttensioned Tendons.” *Transportation Research Record* 1827. 2003. pp. 44-52.
- [6] Dally, J. and Riley, W. *Experimental Stress Analysis*. 4th ed. College House Enterprises, LLC, Knoxville, TN, 2005.
- [7] Park, H.S., Lee, H.M., Adeli, H. and Lee, I. “A New Approach for Health Monitoring of Structures: Terrestrial Laser Scanning.” *Computer-Aided Civil and Infrastructure Engineering*. Vol. 22, No. 1, 2007. pp. 19-30.
- [8] Ji, Baifeng and Weilian Qu. “The Research of Acoustic Emission Techniques for Non Destructive Testing and Health Monitoring on Civil Engineering Structures.” *International Conference on Condition Monitoring and Diagnosis*. Beijing, China, April 21-24, 2008.

- [9] M. J. Whelan, M. V. Gangone and K. D. Janoyan, "Highway Bridge Assessment Using an Adaptive Real-Time Wireless Sensor Network," in *IEEE Sensors Journal*, vol. 9, no. 11, pp. 1405-1413, Nov. 2009.
- [10] G. Hackmann, W. Guo, G. Yan, Z. Sun, C. Lu and S. Dyke, "Cyber-Physical Codesign of Distributed Structural Health Monitoring with Wireless Sensor Networks," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 63-72, Jan. 2014.
- [11] F. X. Li, A. A. Islam, A. S. Jaroo, H. Hamid, J. Jalali and M. Sammartino, "Urban highway bridge structure health assessments using wireless sensor network," 2015 *IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, San Diego, CA, 2015, pp. 75-77.
- [12] I. Khemapech, W. Sansrimahachai and M. Toahchoodee, "A real-time Health Monitoring and warning system for bridge structures," 2016 *IEEE Region 10 Conference (TENCON)*, Singapore, 2016, pp. 3010-3013.
- [13] T. Harms, S. Sedigh and F. Bastianini, "Structural Health Monitoring of Bridges Using Wireless Sensor Networks," in *IEEE Instrumentation & Measurement Magazine*, vol. 13, no. 6, pp. 14-18, December 2010, doi: 10.1109/MIM.2010.5669608.
- [14] M. Giammarini, D. Isidori, E. Concettoni, C. Cristalli, M. Fioravanti and M. Perialisi, "Design of Wireless Sensor Network for Real-Time Structural Health Monitoring," 2015 *IEEE 18th International Symposium on Design and Diagnostics of Electronic Circuits & Systems*, Belgrade, 2015, pp. 107-110.
- [15] Su-su Lei, Yong-tao Gao and Dan-guang Pan, "Comprehensive Real-Time Bridge Health Monitoring System of Tongtai Bridge" in *MATEC Web of Conferences* 31 11003 (2015), DOI: 10.1051/matecconf/20153111003

- [16] Swit, Grzegorz & Krampikowska, Aleksandra & Lương Minh, Chinh & Chinh,. (2016), "A Prototype System for Acoustic Emission-Based Structural Health Monitoring of Mỹ Thuận Bridge," 10.1109/PHM.2016.7819877.
- [17] P. K. Patil and S. R. Patil, "Structural health monitoring system using WSN for bridges," 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, 2017, pp. 371-375, doi: 10.1109/ICCONS.2017.8250746.
- [18] Ryan Wike, "Permissions and consent in the Microsoft identity platform endpoint," Microsoft identity platform- Azure,01-2020
- [19] Ryan Wike, "OAuth authorization code flow," Microsoft identity platform- Azure, 01-2020
- [20] Celeste de Guzman, "Microsoft identity platform and OpenID Connect protocol", 06-2020