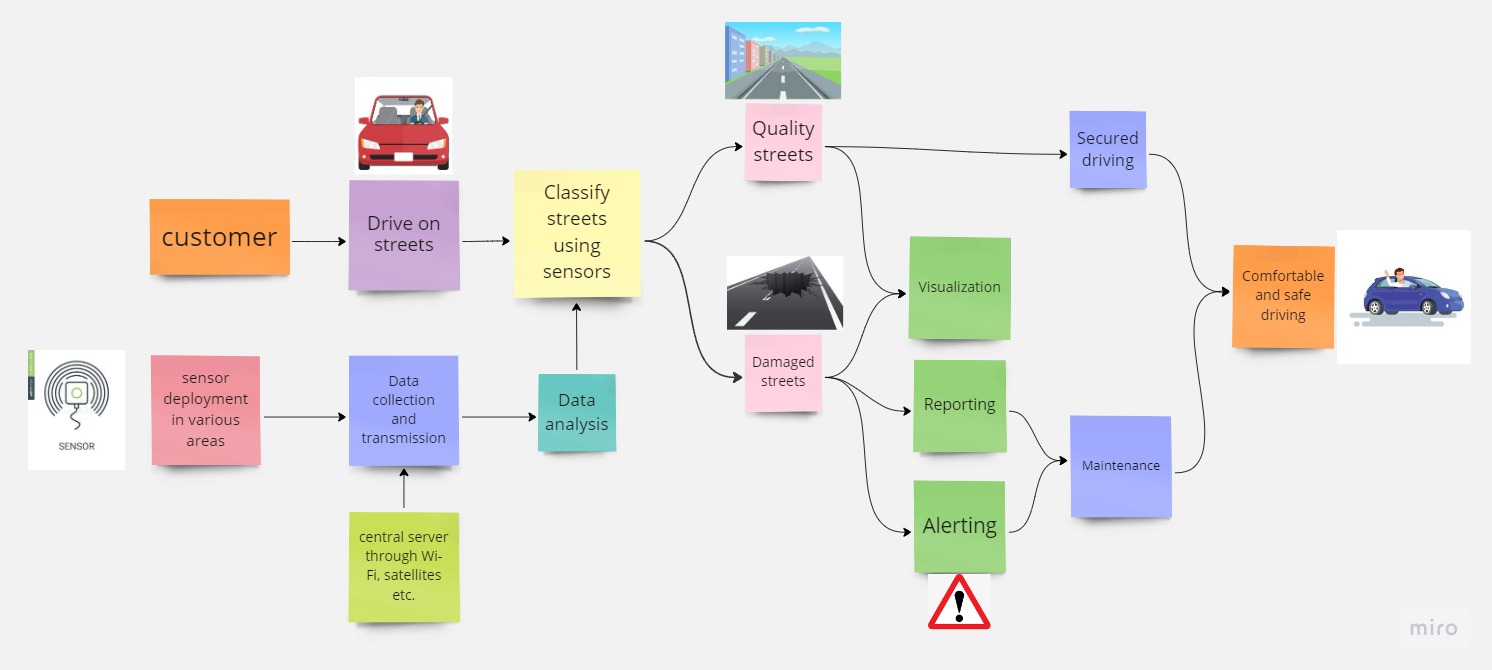
**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 12 May 2023 |
| Team ID | NM2023TMID22230 |
| Project Name | Project - SQUID: Street Quality Identification |

**Technical Architecture:**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | IoT Sensors | Sensors installed on street pavements to collect data on various parameters such as vibration, temperature, humidity, and pressure. | Accelerometers, temperature sensors, humidity sensors, pressure sensors. |
|  | IoT Gateway | A device that preprocesses the data from the sensors and transmits it to the cloud-based server. It can also perform data aggregation and filtering. | Raspberry Pi, Arduino, or other microcontroller boards with built-in wireless communication modules such as Wi-Fi, LoRa, or Bluetooth. |
|  | Cloud-based Server | A cloud-based server to store and process the preprocessed data. The server can use a variety of technologies for data storage and processing such as databases and big data processing tools. | IBM Cloud, Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, Apache Cassandra, Apache Hadoop, Apache Spark. |
|  | Data Analytics | Machine learning algorithms to analyze the collected data and identify street quality parameters such as road roughness, potholes, and cracks. | Python libraries for machine learning such as scikit-learn, TensorFlow, Keras, and PyTorch. |
|  | Data Visualization | Techniques to present the identified quality parameters in a visual format such as graphs, charts, or maps. | Web-based visualization tools such as D3.js, Plotly, or Tableau. |
|  | Alert Generation | A system to generate alerts when identified quality parameters exceed a certain threshold. The alert can be sent via SMS, email, or push notification to the concerned authorities. | Python, AWS Simple Notification Service (SNS), AWS Simple Queue Service (SQS). |
|  | Mobile Application | A mobile application for users to view street quality parameters and provide feedback on maintenance needs. | Android or iOS development frameworks such as Flutter, React Native, or Xamarin. |
|  | System Integration | Integration with existing transportation management systems to streamline maintenance workflows and improve decision-making. | APIs, microservices architecture, event-driven architecture, message queuing. |
|  | Security | Purpose of Machine Learning Model | Object Recognition Model, etc. |
|  | Infrastructure | Infrastructure components such as load balancers, failover mechanisms, and cloud-based services to ensure high availability and minimize downtime. | Load balancers such as AWS Elastic Load Balancer, failover mechanisms such as AWS Route 53, and cloud-based services such as AWS Lambda. |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Accelerometers, gyroscopes, sensors, cameras and other IoT devices are used. | TensorFlow, Keras, OpenCV, Scikit-learn, PyTorch, Apache Spark etc. |
|  | Security Implementations | Authentication, Authorization, Data encryption, Use of firewalls, Intrusion prevention, Regular security audits, Secure data storage, Secure communication protocols and User activity logging | TLS/SSL, role-based access control (RBAC), Linux based IoT device, Snort,VLANs etc. |
|  | Scalable Architecture | Both 3-tier and microservices architectures are designed to be highly scalable. In a 3-tier architecture, each tier can be scaled independently to handle the load. | A 3-tier architecture can be implemented using technologies such as Java, .NET, and Node.js, while a microservices architecture can be implemented using technologies such as Docker, Kubernetes, and Spring Boot. |
|  | Availability | Load Balancers, Distributed Servers, Cloud Services, Replication, Monitoring and Alerting | HAProxy, Nginx, Apache Cassandra, Apache Kafka, IBM Cloud, Apache ZooKeeper, Apache Hadoop, PagerDuty, Slack etc. |
|  | Performance | Number of requests per second, Use of Cache, Use of Content Delivery Networks (CDNs), Use of Asynchronous Processing, Use of Load Balancers, Use of In-Memory Databases | Docker, Kubernetes, Redis, Celery, HAProxy or Nginx, Memcached etc. |