**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 15 May 2023 |
| Team ID | NM2023TMID13277 |
| Project Name | Intelligent Garbage Classification using Deep learning |

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

**Example: Intelligent Garbage Classification using Deep learning**



Guidelines:

Include all the processes (As an application logic / Technology Block)

Provide infrastructural demarcation (Local / Cloud)

Indicate external interfaces (third party API’s etc.)

Indicate Data Storage components / services

Indicate interface to machine learning models (if applicable)

**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | Dataset | Collection of garbage images for training | Image Dataset |
|  | Preprocessing | Resize, normalize, and split dataset | Data preprocessing techniques |
|  | Deep Learning Model | Architecture for garbage classification | Convolutional Neural Network (CNN) |
|  | Training | Train the model using the dataset | Deep learning frameworks (e.g., TensorFlow, PyTorch) |
|  | Hyperparameter Optimization | Optimize model hyperparameters | Hyperparameter optimization algorithms |
|  | Validation | Evaluate model performance on validation set | Metrics (e.g., accuracy, loss) |
|  | Testing | Test model on separate test dataset | Test dataset |
|  | Performance Evaluation | Measure model accuracy, precision, recall | Performance metrics (e.g., F1 score) |
|  | Transfer Learning | Leverage pre-trained models for classification | Pre-trained CNN models (e.g., VGG16, ResNet) |
|  | User Interface | Develop an interface for user interaction | Web application or mobile app |
|  | Real-time Classification | Classify garbage images in real-time | Real-time image processing |
|  | Optimization | Improve system performance and efficiency | Performance optimization techniques |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Utilizing open-source frameworks for implementing the intelligent garbage classification system allows for flexibility, community support, and access to pre-trained models. | TensorFlow, PyTorch, or Keras. |
|  | Security Implementations | Implementing security measures ensures the protection of data and prevents unauthorized access to the system. | encryption, secure authentication, and secure communication protocols (e.g., HTTPS) |
|  | Scalable Architecture | A scalable architecture allows the system to handle increasing amounts of data and user requests efficiently. | leveraging cloud-based platforms, containerization technologies (e.g., Docker), and scalable deep learning frameworks (e.g., TensorFlow Serving). |
|  | Availability | The system should be available and accessible to users at all times. | reliable hosting solutions, load balancing techniques, and effective monitoring to identify and resolve any downtime or performance issues. |
|  | Performance | Optimizing the performance of the system ensures fast and accurate garbage classification. | deep learning models, utilizing hardware accelerators (e.g., GPUs), implementing caching mechanisms, and leveraging distributed computing frameworks (e.g., Apache Spark) for large-scale data processing. |