**Software Requirements Specification**

For

Solution for mask and temperature detection for large gatherings

08-3-2022

Prepared by

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll No** | **Branch** |
| Aanchal Sharma | R110218002 | CSE-CCVT |
| Arnav Sharma | R110218032 | CSE-CCVT |
| Aman Sharma | R110218018 | CSE-CCVT |
| Gaurvendra Singh | R110218056 | CSE-CCVT |



Department of Systemics

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Topic** | | **Page No** |
| Table of Content | | 1 |
| Revision History | | 2 |
| 1 | Introduction | 2 |
|  | 1.1 Purpose of the Project | 2 |
|  | 1.2 Target Beneficiary | 2 |
|  | 1.3 Project Scope | 2 |
|  | 1.4 References | 2 |
| 2 | Project Description | 2 |
|  | 2.1 Reference Algorithm | 2 |
|  | 2.2 Data/ Data structure | 2 |
|  | 2.3 Project Features | 2-3 |
|  | 2.4 Design and Implementation Constraints | 3 |
|  | 2.5 Design diagrams | 3 |
|  | 2.6 Assumption and Dependencies | 3 |
| 3 | System Requirements | 3 |
|  | 3.1 User Interface | 3 |
|  | 3.2 Software Interface | 3 |
|  | 3.3 Database Interface | 4 |
|  | 3.4 Protocols | 4 |
| 4 | Non-functional Requirements | 4 |
|  | 4.1 Performance requirements | 4 |
|  | 4.2 Security requirements | 4 |
|  | 4.3 Software Quality Attributes | 4 |
|  | |  |

|  |  |  |
| --- | --- | --- |
| 1 | INTRODUCTION | |
|  | 1.1 Purpose of the Project | In this project, we have developed a Realtime the mask detection model can be said to be a combination of classification and face detection model. For the purpose of classification, we use transfer learning with an Xception model trained on the ImageNet dataset with a modified final fully connected layer. While using the face detection model, several different approaches were tried upon based on existing literature, and the one which worked the best was a RetinaNet Face pre-trained model which gave the highest measures of recall while experimenting on different use-cases and testing images of people in a crowded setting. The models and implementation details for them have been discussed in an objective manner as part of this section, and while providing an insight on the approach used (and why it was chosen in the first place), we delve into our final mask detection model which was built using a combination of the classification and face detection models as were briefly described above. |
|  | 1.2 Target Beneficiary | Government, Offices, Public places, Colleges, Metro, etc. |
|  | 1.3 Project Scope | Creation of Realtime face mask and temperature monitoring system using deep learning and transfer learning |
|  | 1.4 References | 1. Chollet, Francois. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. 1800-1807. 10.1109/CVPR.2017.195. 2. He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian. (2016). Deep Residual Learning for Image Recognition. 770-778. 10.1109/CVPR.2016.90. 3. Deng, Jiankang & Guo, Jia & Zhou, Yuxiang & Yu, Jinke & Kotsia, Irene & Zafeiriou, Stefanos. (2019). RetinaFace: Single-stage Dense Face Localisation in the Wild. |
| 2 | PROJECT DESCRIPTION | |
|  | 2.1 Reference Algorithm | Xception, RetinaFace, ResNet50, CNN. |
|  | 2.2 Characteristic of Data | ML Model for Realtime face mask and temperature monitoring. |
|  | 2.3 Project Features | * Data gathering and Model Training: Collecting Social Network Ads data from Kaggle and training the model using Xception for classification between mask and unmasked. * Docker file creation: Docker file creation to have an image ready for deployment. * Written in Python * Interactive and user-friendly interface. * Can be used to large gathering. * Highly precise and accurate. * Real Time Monitoring. |
|  | 2.4 Design and Implementation Constraints | 1. Docker 2. TensorFlow/Keras 3. Pytorch 4. Git/GitHub 5. Internet Connectivity. 6. Minimum 4GB RAM and 2 cores if using Virtual Machines. |
|  | 2.5 Design diagrams |  |
|  | 2.6 Assumption and Dependencies | End User must have a good quality surveillance camera and good configurated computer. |
| 3 | SYSTEM REQUIREMENTS | |
|  | 3.1 User Interface | Supported Browsers (Chrome, Firefox, Safari, etc) |
|  | 3.2 Software Interface | GIT for CD, Docker for CD, Xception, RetinaFace, ResNet50 algo |
|  | 3.3 Database Interface | No Database Requirement |
|  | 3.4 Protocols | SSL, HTTP, HTTPS |
| 4 | NON-FUNCTIONAL REQUIREMENTS | |
|  | 4.1 Performance requirements | This service is provided through a web interface. So, any device which could support a web-browser, has a camera will be able to access this service. |
|  | 4.2 Security requirements | Since this service takes a video footage of person and sends it to our backend server, privacy becomes a major issue. We will be deleting the captured source footage once it has gone through the code and an output has been received. |
|  | 4.3 Software Quality Attributes | Since this service is running its backend in the docker container and has a CLI based terminal, hence it is highly available, accessible, and scalable because of the agile model of the Docker. |