**RV College of Engineering®, Bengaluru – 59**

**Department of Computer Science and Engineering**

**Database Design Laboratory (18CS53)**

**Requirement specification**

|  |  |  |
| --- | --- | --- |
| **TITLE: Face Recognition(Attendance) System** | | |
| **TEAM** | USN :- 1RV18CS124 | Name :- PULAK PATHAK |
| USN :- 1RV18CS139 | Name :- ROUNAK JAIN |

1. **Hardware Specification**

* x86 64-bit CPU (Intel / AMD architecture)
* 4 GB RAM
* 5 GB free disk space or more

1. **Software Specification**

* Flask - These are API’s of Python to build up web-applications.
* Sublime Text - Used as open source text and source code editor.
* React JS- npm is a package manager for the JavaScript programming language.
* MySQL/SQLite - SQLite is a software library that provides a relational database management system.
* MongoDB - To map NoSQL database

1. **Functional Requirements**

Every college has multiple branches. Each branch contains a unique branch id, branch name, and seats. Every branch has multiple students and faculties. Each faculty contains a unique name, id, and address. Each student contains a unique name, id, and address. Every faculty give one attendance to each student on every working day based on if they are present in the respective classes. Every attendance has a unique id. Every faculty and student takes multiple leaves on working days. Each leave contains leave id and number of days the student/faculty is absent.

**Roles of Facial Recognition, API and the Front-End:**

Facial Recognition:

As mentioned above, for [Facial recognition](https://github.com/ageitgey/face_recognition) we will use the python face recognition library. We **give a picture** of a user to record his "facial identity". A first model will dig up whether there is **a face**or not and **determine its location** on the photo. A second model will **calculate the facial parameters**. (distance between eyes, shape of the chin,…)We save this so-called "encoded" data by **linking them to a name** so that they can be **compared** with a future picture. Then we give a **new nameless photo** and the same process will be repeated except that this time, a **third model will compare the parameters of the face** with those it **already knows**.

If we want to add more users, we have to repeat those steps for each one.

API:

As mentioned above, for the API we use [Flask](https://github.com/pallets/flask). The purpose here is to receive the data from our face recognition model and redistribute it to the front when requested. But also to have the possibility to add a new employee with his name and photo and to be able to delete some only by recovering his name.

Database Use:

The arrival or departure time in the database MongoDB. At the end you get for every day and every employee a *record* with:

* Name
* Date
* Arrival time
* Arrival picture
* Departure time
* Departure picture
* Is he late?
* Has he left early?

Front End:

The sending receiving and displaying of data is managed by ReactJS front end components.