Logo, company name

Description automatically generated

**Under Supervision of**

**Dr. Haitham Farouk**

**Dr. Hussein Sharaf**

**Team Members**

* **Baher Adel Abo El-Wafa**
* **Ali El-Sayed Ghandour**
* **Abdelrahman Mohammed Abdelfattah**
* **Mohammed Ashraf Ahmed El-Wardany**
* **Hazem Ragab**
* **Omar Diaa Mahmoud**
* **Sharaf El-Deen Ashraf Ahmed**
* **Ibraheem Hop-Allah El-Bana**
* **Eslam Mohammed Awad**
* **Ibrahim Ehab Abd-El-Megeed**
* **Omar Khalid El-Sayed**
* **Mahmoud Yousef**

|  |
| --- |
| **1. Preface** |

**Preface**

**Description**

In this Document which talks about our system **Fingerprint Identification Attendance** which makes the operation to take Attendance more Efficiently and Easy with more facilities and capabilities to reduce time and effort.

This System aim to make the process more efficiency and automated which can do by take Fingerprint and some data from user or employee to contact the Manager or Administrator directly which don’t need paper or employee to can do this job which we can this process from manual to digital process.

**Structure of this documentation**

In this system, we use many of programming languages and physical hardware Like ***Python, C, react*** in software and ***Arduino, fingerprint machine*** in hardware.

**About system**

In this case, the document doesn’t need special programmer or specialized in this field which make this document for user or customer and system requirement to achieve the main purpose with more quickly and suitable for user.

|  |
| --- |
| **2. introduction** |

**Introduction**

The proposed system will be capable to perform the following operations:

1. Manage attendance in an automated manner

2. Organize attendance in a hierarchal form to provide efficient representation and ease of use within a service orientation towards the instructors to make them do the minimal effort in order to save their time

It consists of three modules: micro control-based system, web API, and user GUI web interface to interact with the system through web GUI and micro control- system peripherals to achieve system usability and user-friendly experience.

We used the following tools and frameworks:

1. Flask

2. PostgreSQL

3. Arduino C

4. Fingerprint sensor

5. python

To provide a sufficient tool kit to the project, to ensure efficient performance and sophisticated results as well.

|  |
| --- |
| **3.**  **System Requirements Specification (SRS)** |

8.System Requirements Specification (SRS)

* Functional Requirements.

1. The system runs and collect the data via fingerprints sensor and send the payload to the API to search for the data in the database and return a token, so the employee be able to sign in.

2. A database must be made to hold all the important information about and required by the system. This database should be a relational database that holds tables about different entities of the system with all the attributes or fields necessary about that specific entity. One of these entities is the employee; the system should have the username, employee id and the fingerprint encoding.

3. The employee first must register on the system to be able to identify him before he can take attendance and that happens via a web or Mobile App for his personal data and register his fingerprint through the hardware device.

4. The database should follow the ACID (atomicity, consistency, isolation, durability) properties which is a set of properties of database transactions intended to guarantee data validity despite errors, power failures, and other mishaps. In the context of databases, a sequence of database operations that satisfies the ACID properties (which can be perceived as a single logical operation on the data) is called a transaction. For example, a transfer of funds from one bank account to another, even involving multiple changes such as debiting one account and crediting another, is a single transaction. Therefore, the database must have a backup and replications. The Quality of the database is maintained in such a way so that it can be very user friendly to all the users of the database. The database will be PostgreSQL.

5. After employee registration, the system should analyze the fingerprint he/she provided and detect his fingerprint and reject the input if it was inappropriate. After passing these conditions, the system should get the encodings of that fingerprint and store them in the database.

Another important edge case is if this specific encoding is already in the database, in this case the input is also rejected, and a message appear to inform the error.

7. For testing and usability measurements, the employee can verify attendance by try the fingerprint sensor and see if it works.

Then the manager checks from admin panel to see the active employees and their data.

8. To log in as a manager or admin the user must provide his username as well as his password. Then the system checks if the user data  
is registered in the database. If the data is valid the application will send a token and sign in if it isn’t the application will send an error message.

9.Any passwords or sensitive information cannot be stored in plain text but rather must be hashed to be secure.

10.Prevent unauthorized URL traversal like access admin page without login this done by check for user credential in user session on every page if the user was not unauthorized to view that page, he will be redirected to the login page. The session id is stored in a cookie on the user browser this allows the server to distinguish each user.

11.The system will send an error message to the employee if fingerprint wasn't clear enough to identify him/her. This will increase the system's  
accuracy. Each employee will know the problem with their registration, and this will make a better user experience.

12.The admin Panel show the data stored in the database which contains the active or waiting employees and their data like salary, shift time, start time and all the personal information.

13.The employee panel can show all the data belongs to the logged in employee and all the personal information.

14. The System is created via connecting the following components:

* *Arduino UNO R3*
* *Advanced Fingerprint module*
* *RTC module*
* *ESP8266 module*
* *Character LCD module 16x2*
* *Buzzer 5V*
* *on/off rocker switch small*
* *Tack Switch (Push Buttons)*
* *Resistors*
* *Rheostat*
* *Round LEDS*
* *9v battery& adapter*
* *Breadboard 840 pin*
* *Male-Male jumper wires*
* *Male-Female jumper wires*
* *Female-Female jumper wires*
* *Arduino USB programming cable*
* Non-Functional Requirements

1. The System Allows To check the heart rate and responds at once if the employee has a bad heart rate or so.
2. The device contains a Touch screen which allows to register easily and create more flexibility.
3. The device contains. Camera to register the with the face recognition .
4. The device contains a body temperature sensor which allows to know if the employee suffer from unsafe body temperature.

|  |
| --- |
| **4. Glossary** |

**3.Glossary**

|  |  |  |
| --- | --- | --- |
| **#** | **Terms** | **Overview** |
| **1** | **API** | An *application programming interface* (**API**) is a computing interface that defines interactions between multiple software intermediaries. It defines the kinds of calls or requests that can be made, how to make them, the data formats that should be used, the conventions to follow, etc. It can also provide extension mechanisms so that users can extend existing functionality in various ways and to varying degrees. |
| **2** | **Arduino UNO** | microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWMoutputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. |
| 3 | **Microcontroller**  **(ATmega328P)** | * **Microcontroller** is a small and low-cost microcomputer, which is designed to perform the specific tasks of embedded systems like displaying microwave’s information, receiving remote signals, etc. * The general microcontroller consists of the processor, the memory (RAM, ROM, EPROM), Serial ports, peripherals (timers, counters), etc. |
| **4** | **Web Service** | A server running on a computer device, listening for requests at a particular port over a network, serving web documents (HTML, JSON, XML, images), and creating web applications services, which serve in solving specific domain problems over the Web (WWW, Internet, HTTP). |
| 5 | **Client-server model** | The Client-server model is a distributed application structure that partitions task or workload between the providers of a resource or service, called servers, and service requesters called clients. In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and deliver the data packets requested back to the client. Clients do not share any of their resources. Examples of Client-Server Model are Email, World Wide Web, etc. |
| **6** | **REST web**  **service** | * REST stands for *representational State Transfer*. REST is web standards based architecture and uses HTTP Protocol. It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods. REST was first introduced by Roy Fielding in 2000. * In REST architecture, a REST Server simply provides access to resources and REST client accesses and modifies the resources. Here each resource is identified by URIs/ global IDs. REST uses various representation to represent a resource like (text, JSON, XML. JSON) is the most popular one. * **HTTP methods**   GET − Provides a read only access to a resource.  POST − Used to create a new resource.  DELETE − Used to remove a resource.  PUT − Used to update a existing resource or create a new resource. |
| **7** | **React** | * React is a JavaScript library for building user interfaces. * React is the most popular front-end JavaScript library in the field of web development. It is used by large, established companies and newly-minted startups alike (Netflix, Airbnb and Instagram). * React brings many advantages to the table, making it a better choice than other frameworks like Angular. |
| **8** | **Flask** | Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. |
| **9** | **Python** | * Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. * Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected.   It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. |
| **10** | **JWT** | * A *JSON WEB Token* (JWT) is JSON Object which is used to securely transfer information over the web (between two parties). * It can be used for an authentication system and can also be used for information exchange. * The token is mainly composed of header, payload, signature. These three parts are separated by dots(.).   JWT defines the structure of information we are sending from one party to the another, and it comes in two forms – Serialized, Deserialized. The Serialized approach is mainly used to transfer the data through the network with each request and response. While the deserialized approach is used to read and write data to the web token. |
| **11** | **PostgreSQL** | also known as Postgres, is a free and open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance. It was originally named POSTGRES, referring to its origins as a successor to the Ingres database developed at the University of California, Berkeley. |
| **12** | **HTTP** | * The Hypertext Transfer Protocol (HTTP) is an application layer protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web, where hypertext documents include hyperlinks to other resources that the user can easily access, for example by a mouse click or by tapping the screen in a web browser. * HTTP functions as a request–response protocol in the client–server computing model. * A web browser, for example, may be the client and an application running on a computer hosting a website may be the server. * The client submits an HTTP request message to the server. The server, which provides resources such as HTML files and other content, or performs other functions on behalf of the client, returns a response message to the client.   The response contains completion status information about the request and may also contain requested content in its message body. |
| **13** | **Algorithm of Fingerprint** | The standard way by which the sensor detect the fingerprint and compare it with stored ones. |
| **14** | **Embedded System** | An embedded system is a *microprocessor*- or *microcontroller*-based system of hardware and software designed to perform dedicated functions within a larger mechanical or electrical system.   * **Examples of embedded systems:**  1. **Mobile phones:** These consist of many embedded systems, including GUI software and hardware, operating systems (OSes), cameras, microphones, and USB (Universal Serial Bus) I/O (input/output) modules. 2. **Industrial machines:** They can contain embedded systems, like sensors, and can be embedded systems themselves. Industrial machines often have embedded automation systems that perform specific monitoring and control functions. |

|  |
| --- |
| **5. user story** |

**4.User Story**

At the time of employee registration, the website gives him a unique id after this stage he should go to the fingerprint index place to put his fingerprint in the sensor. the New Employee is asked to give their fingerprint. The finger print is taken with fingerprint index, employee id, it takes three times to ensure the finger. The fingerprint scanner device scan and generate a finger print template, the process in known as fingerprint enrollment. The template is stored in the database and is used for employee verification. After the completing of enrollment, employee is verified when he/she tries to enter in office. When the employee is verified, a record is stored in database.

In this project work, the system is designed with panels; such admin, Employee Panel. The admin has the super power of all access. The admin can edit, delete, add all data for full system. He can check currently working employees and also who are absent till now. He can also check the list of all employees who takes leave. The admin can generate crystal report as Monthly Attendance Sheet.

(1) A registered employee can submit his attendance to the fingerprint sensor

(2) A registered employee can request a for an id

(3) the admin can accept the employee request

(4) the admin can add user name

(5) the admin can add new id

(6) the admin can add, delete or modify the employee status

Every employee can show his/her

personal details

Every employee can show his/her

personal details

Every employee can show his/her

personal details

Every employee can show his/her

personal details

Every employee can show his/her

personal details

International Journal of Innovative Research in Information Security (IJIRIS) ISSN: 2349-7017

Issue 09, Volume 4 (September2017) www.ijiris.com

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

IJIRIS: Impact Factor Value – SJIF: Innospace, Morocco (2016): 4.651

Indexcopernicus: (ICV 2015): 73.48

© 2014- 17, IJIRIS- All Rights Reserved Page - 8 -

Figure 3: Database Schema Diagram of the Proposed System.

V. FINGERPRINT ENROLLMENT AND VERIFICATION

At the time of employee registration, the New Employee is asked to give their fingerprint. The finger print is taken

with fingerprint index [13], employee id, it takes three times to ensure the finger. The fingerprint scanner device

scan and generate a finger print template [14], the process in known as fingerprint enrollment (see Figure-4). The

template is stored in the database and is used for employee verification (see Figure-5). After the completing of

enrollment, employee is verified when he/she tries to enter in office. When the employee is verified, a record is

stored in database

International Journal of Innovative Research in Information Security (IJIRIS) ISSN: 2349-7017

Issue 09, Volume 4 (September2017) www.ijiris.com

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

IJIRIS: Impact Factor Value – SJIF: Innospace, Morocco (2016): 4.651

Indexcopernicus: (ICV 2015): 73.48

© 2014- 17, IJIRIS- All Rights Reserved Page - 8 -

Figure 3: Database Schema Diagram of the Proposed System.

V. FINGERPRINT ENROLLMENT AND VERIFICATION

At the time of employee registration, the New Employee is asked to give their fingerprint. The finger print is taken

with fingerprint index [13], employee id, it takes three times to ensure the finger. The fingerprint scanner device

scan and generate a finger print template [14], the process in known as fingerprint enrollment (see Figure-4). The

template is stored in the database and is used for employee verification (see Figure-5). After the completing of

enrollment, employee is verified when he/she tries to enter in office. When the employee is verified, a record is

stored in database

(7) The admin can check currently working employees and also who are absent till now. He can also check the list of all employees who takes leave.

(8) the admin can request attendance report for the whole employees

(9) A registered employee can submit his attendance to fingerprint index

(10) An admin is able to log in the system using his name and a unique admin ID in order to retrieve the request and required data specifically

(11) An admin can request an attendance report for each employee separately

(12) An admin can request an attendance report for each employee separately in a specified time interval.

(13) A registered employee can submit his attendance to event through different method from such as finger print sensor or even a unique ID given to each employee,

(14) An admin can to retrieve and track the history of a specified employee in order.

(15) Any employee is able to register to our system through a dedicated tab in our website to take id to submit it in the fingerprint sensor.

|  |
| --- |
| **6.** **User Requirements Definition** |

**5.User Requirements Definition**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Requirement** | **Category** | **Type** | **Classification** | **Source** | **Priority** |
| **(1)** | **Any registered**  **employee can take**  **attendance** | employee | Functional | Usability | External | high |
| **(2)** | **Each employee will**  **have a unique ID to submit the id in the fingerprint sensor** | employee | Functional | Usability | External | high |
| **(3)** | **admin Login**  **Requiring admin full**  **name as well as**  **admin ID** | admin | Functional | Usability | External | high |
| **(4)** | **Add new employee is a**  **feature available to**  **the admin to add employees** | admin | Functional | Usability | External | high |
| **(5)** | **Add new id is a**  **feature available to**  **the admin to add**  **new id for employee in order to**  **manage and**  **organize the system.** | admin | Functional | Usability | External | high |
| **(6)** | **Add or delete is a**  **feature available to**  **the admin to add**  **employee in order to**  **manage and**  **organize the system**  **more efficiently** | admin | Functional | Usability | External | high |
| **(7)** | **Print Attendance**  **report due to**  **specified time**  **interval as well as**  **the availability to**  **show each employee**  **record separately** | admin | Functional | Usability | External | high |
| **(8)** | **Data recodes**  **update in real-time**  **manner to enable**  **the admin to check** | admin | Functional | CMS | External | high |
| **(9)** | **Give true flag to**  **indicate that the**  **system has been**  **recognized the**  **employee successfully.** | System | Functional | CMS | External | high |
| **(10)** | **Give false flag to**  **indicate the system**  **is unable to**  **recognize an employee or add him to the waiting list.** | System | Functional | CMS | External | high |
| **(11)** | **Show message if the new employee put any thing false in the fingerprint sensor.** | Admin | Functional | Usability | External | high |
| **(12)** | **Give permissions is**  **a feature which is**  **available to the**  **admin in order to know if the employee leaved at a time other than allowed to leave** | Admin | Functional | Usability | External | high |
| **(13)** | **Request for id**  **is a feature which is**  **available to the**  **employee.** | employee | Functional | Usability | External | high |
| **(14)** | **it is**  **required to take the**  **database backup** | system | Non-  Functional | CMS | External | high |
| **(15)** | **The system is**  **available 100% for**  **the user or the new employee he/she want to register** | system | Non-  Functional | CMS | External | high |
| **(16)** | **The system requires**  **a database in order**  **to store persistent data. The database**  **should have backup.** | system | Non-  Functional | CMS | External | high |
| **(17)** | **Mean Time to**  **Repair (MTTR)** | system | Non-  Functional | CMS | External | high |
| **(18)** | **The system should**  **accurately provide**  **real time**  **information taking**  **into consideration**  **various concurrency**  **issues. The system**  **shall provide 100%**  **access reliability.** | system | Non-  Functional | CMS | External | high |
| **(19)** | **Power availability as**  **there is an**  **embedded unit in**  **our system which**  **acts between the system**  **and employee** | System | Non-  Functional | Usability | External | high |
| **(20)** | **Any employee is able**  **to register to our**  **system through a**  **dedicated tab in our**  **website to take the data from him and give him a id to register in the fingerprint sensor place.** | employee | Functional | Usability | External | High |
| **(21)** | **The data of employee or admin is sent in type of encrypt token and it is saved in the device if make another login.** | employee | Non-Functional | Usability | External | High |
| **(22)** | **An error messages**  **must be shown in**  **the registration**  **form to accomplish**  **data integrity** | System | Functional | CMS | Internal | High |
| **(23)** | **A cookie software**  **must be provided to**  **ensures and provide**  **data integrity** | System | Functional | CMS | Internal | High |

|  |
| --- |
| **7.**  **System Evolution** |

**6.System Evolution**

fingerprint recognition is an important application of Image processing owing to its use in many fields. Identification of individuals in an organization for the purpose of attendance is one such application of fingerprint sensor. Maintenance and monitoring of attendance records plays a vital role in the analysis of performance of any organization. The purpose of developing attendance management system is to computerize the traditional way of taking attendance. Automated Attendance Management System performs the daily activities of attendance marking and analysis with reduced human intervention. The prevalent techniques and methodologies for detecting and recognizing the fingerprint fail to overcome issues such as scaling, pose, illumination, variations, rotation, and occlusions. The proposed system aims to overcome the pitfalls of the existing systems and provides features such as detection of fingerprint, extraction of the features, detection of extracted features, and analysis of employee attendance. The system integrates techniques such as fingerprint features and cascading classifier for feature detection. The system provides an increased accuracy due to use of a large number of features (Shape, Color, LBP, wavelet, Auto-Correlation of the fingerprint. fingerprint are recognized using Euclidean distance and k-nearest neighbor algorithms. Better accuracy is attained.

The system is tested for various use cases. We consider a specific area such as employee attendance for the purpose of testing the accuracy of the system. The metric considered is the percentage of the recognized fingerprints per total number of tested fingerprints of the same person.

The process of preventing employee time theft by monitoring employee’s work hours, login time, departures, breaks, and time off is attendance management. Organizations use a range of ideas to control employee attendance, from punching cards and databases to automated attendance management systems and biometric devices. Without an attendance. management system, it was a hard to gather the data. Common methods were to capture it on Excel sheets or records of physical attendance. These are vulnerable to environmental mistakes and are obsolete for several shifts when it comes to labeling and recording correct attendance.

To view, submit, monitor, review, accept, or deny leave applications as necessary, both staff and their managers access an employee portal. Managers will see which members of their team are punctual by viewing their members in and out over time. The attendance information is collected and shown in real-time at the end of the day, week, or month and is not stored. A great app for attendance management can give you 100percent employee working hour’s visibility, reduce missed punches, and standardize attendance monitoring.

In simplifying time and attendance monitoring, features included in the attendance management play an immense part. The attendance management system essential for, notably for hourly or non-exempt workers, employees are expected to show up for work. For non-exempt employees, who sometimes perform jobs that require a person to be there to serve clients, this is essential.

Both staff and managers will relax and not think about whether attendance was correctly recorded or not due to a software module for real-time attendance. Instead of wasting everyone’s time and money tracking and mailing attendance information, permissions, and follow-ups, they can both concentrate on completing their tasks and assignments. This has contributed to dedicated employee time-tracking applications, ranging from on premise software to cloud-based tools with advanced analytics and automated configuration of schedules.

|  |
| --- |
| **8.**  **System**  **Architecture** |

**7.System Architecture**

General View for Employee System

Fingerprint Sensor

Web API Services

End-User Application

Arduino UNO R3

RTC Module

Physical View

Advanced Fingerprint

ESP8266

Character LCD 16x2

Buzzer 5V

on/off rocker switch small

Tack Switch (Push Buttons)

9v battery &

Round LEDS

Breadboard 840 pin

Wires

Layered Architecture

Presentation Layer

Client

Client

Client

Data Access Layer

Database

to

Business Layer

API for request handling

to

DB Handler

(Read DB data)

Service

Handler

Client

Request

Get, Post, Put

Response

JSON

Database

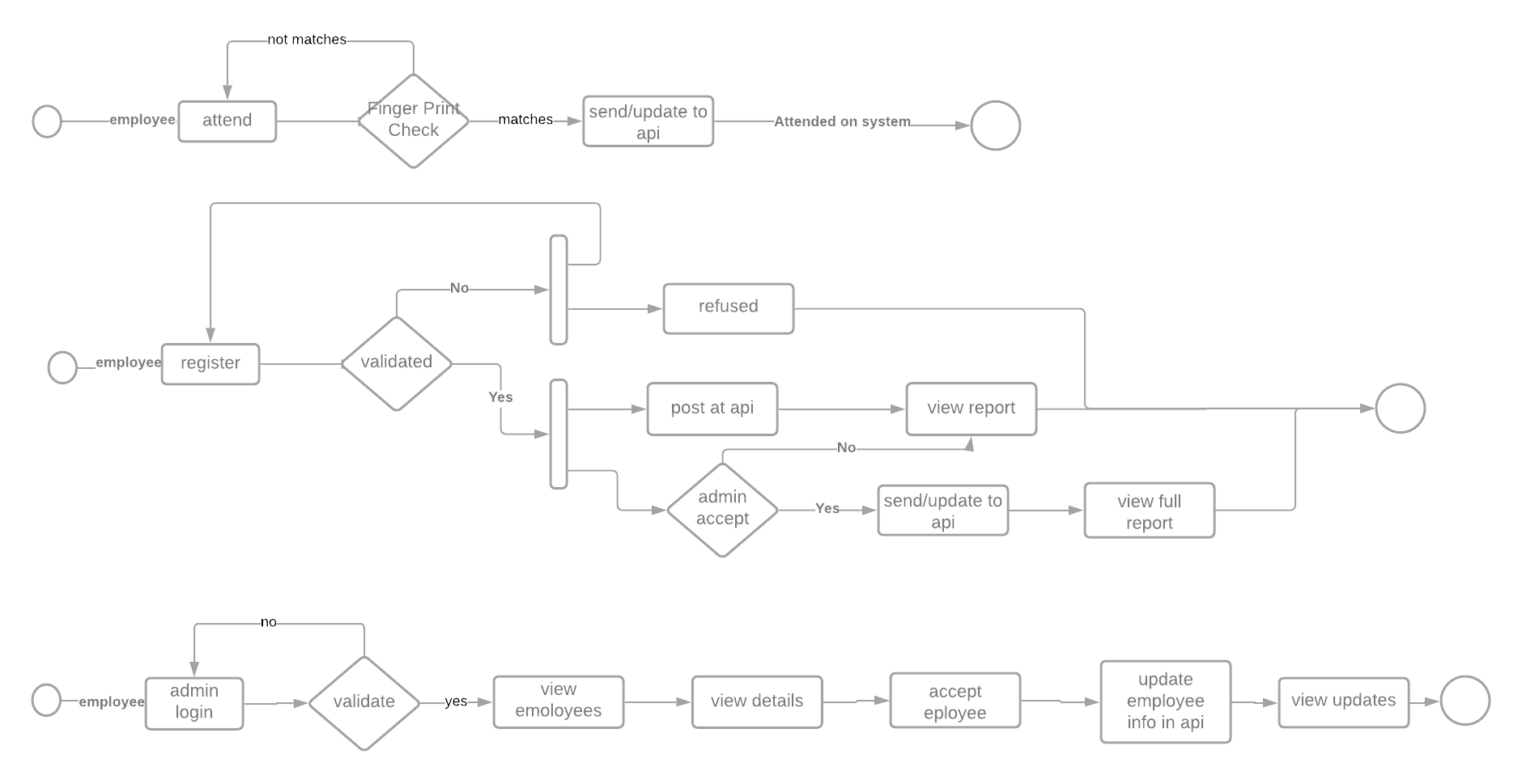
REST Request Handler

High Level View

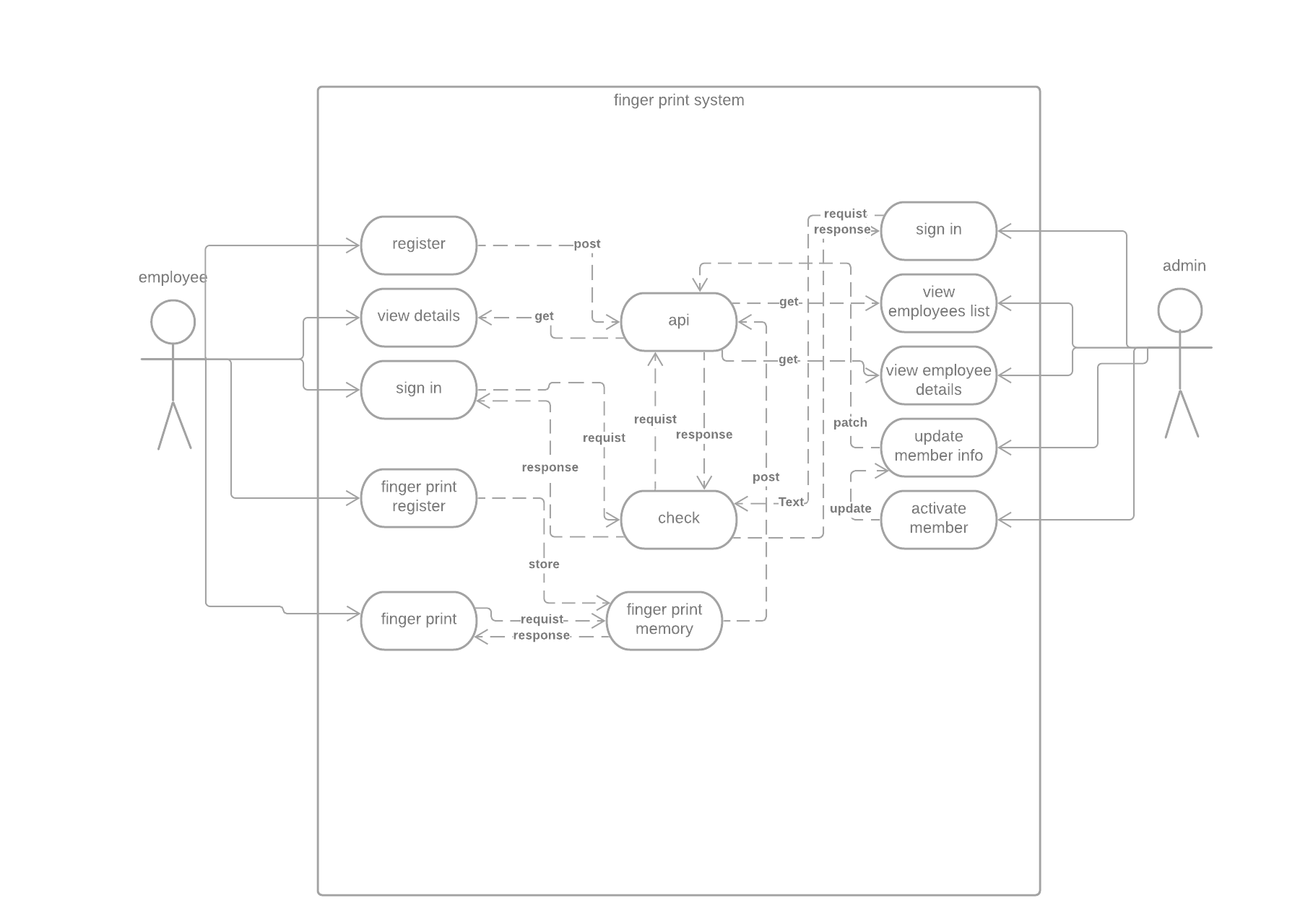
|  |
| --- |
| **9.**  **System models** |

**9.System models**

**Activity diagram**

****

**Use case diagram**

****

**10.**  **Wireframes**

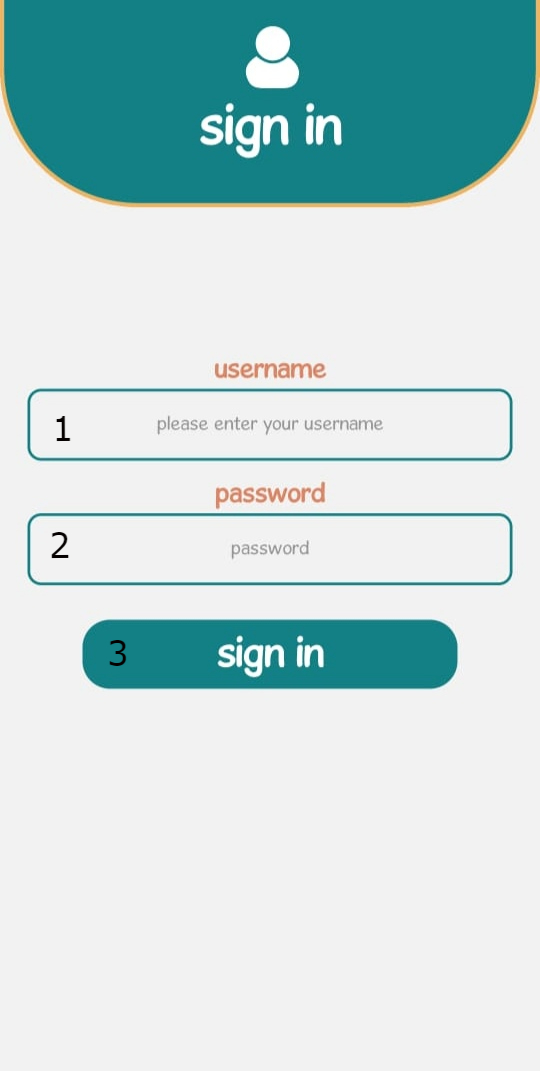
**10.1**

**Mobile App**

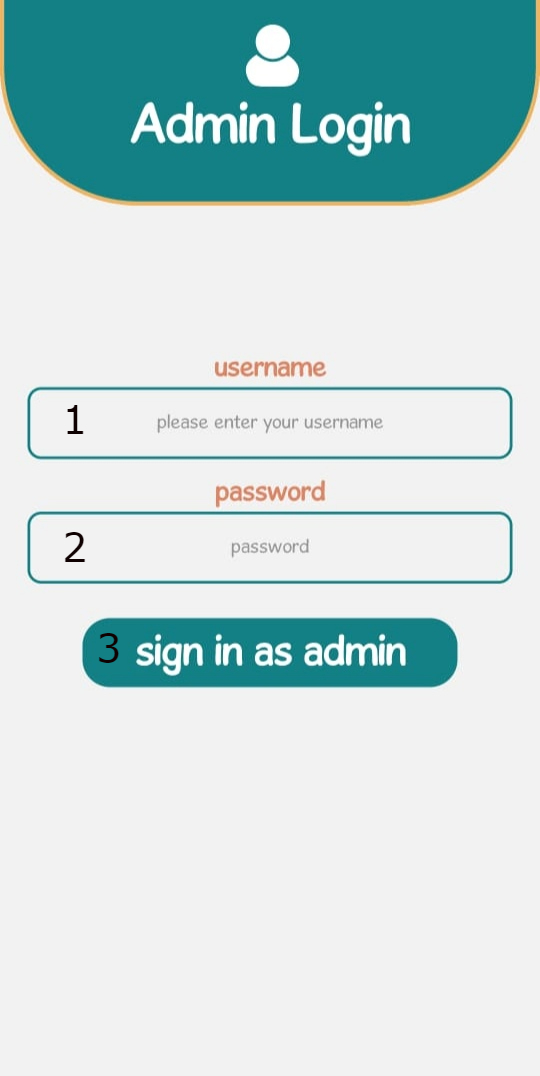
**Logo, company name

Description automatically generated**

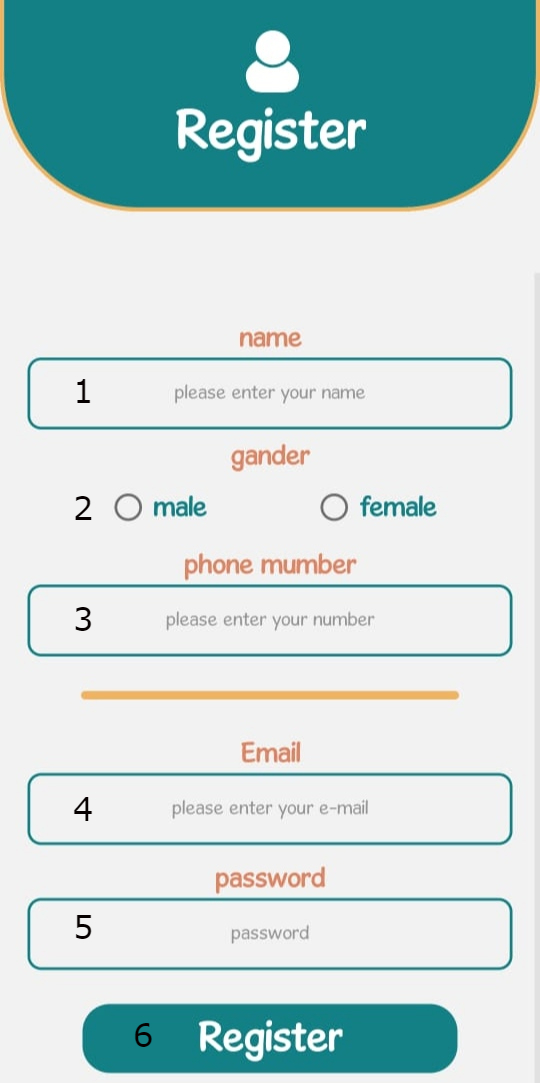
1. **Register as a new member.**
2. **Sign in as Admin / Employee.**

****

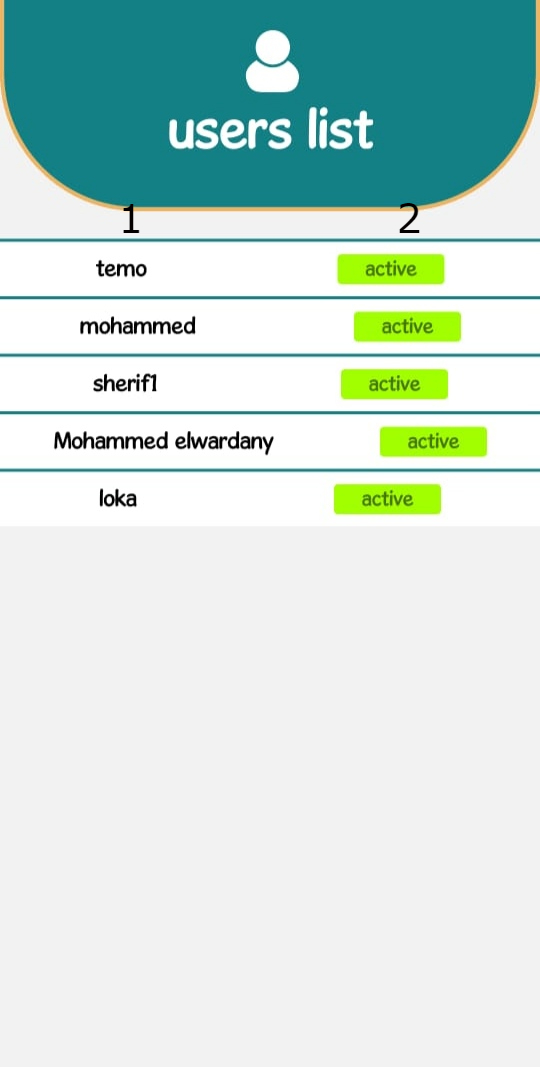
1. **Employee Username.**
2. **Employee password.**
3. **Sign in Button.**

****

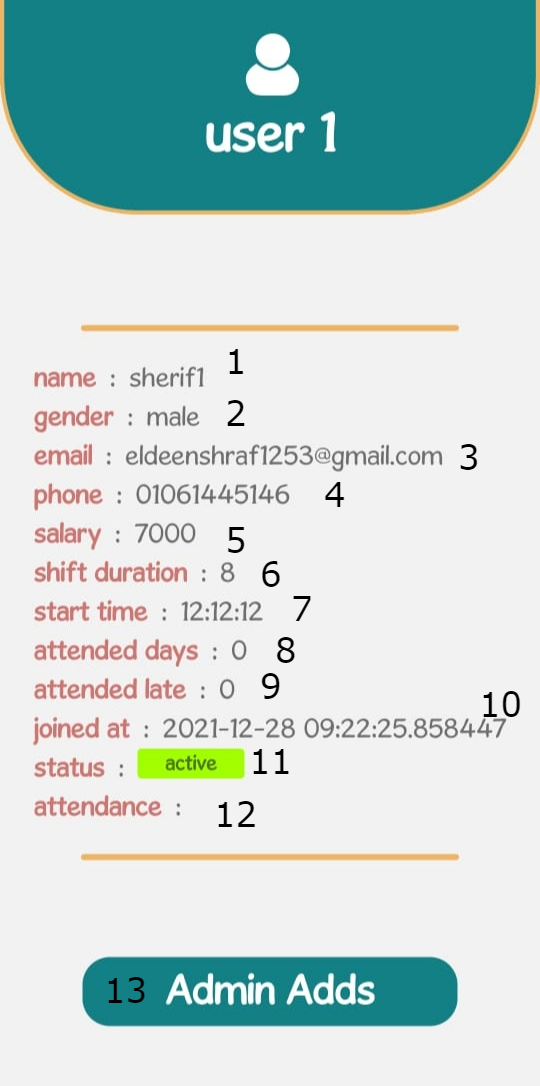
1. **Admin Username.**
2. **Admin Password.**
3. **Sign in button.**

****

1. **Username. 5. Password.**
2. **Gender. 6.Register Button.**
3. **Phone Number.**
4. **E-Mail.**

****

1. **Name.**
2. **Status (Active / Waiting).**

****

1. **Name. 7.Start Time.**
2. **Gender. 8.Attendace Days.**
3. **E-mail. 9.Late Attendance.**
4. **Phone Number. 10.Start day.**
5. **Salary. 11.Status (Active/Waiting).**
6. **Shift Duration. 12. Attendance. 13.Admin Panel**

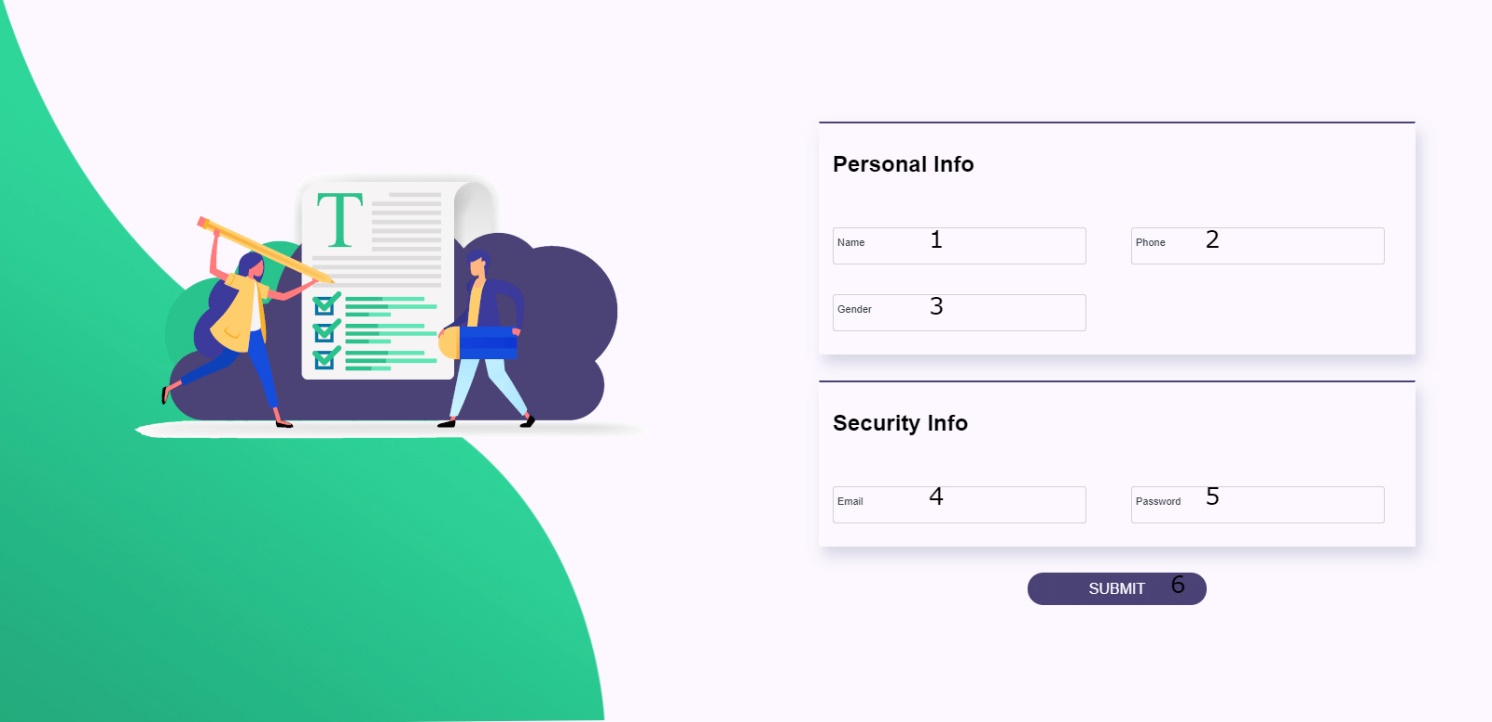
**10.2**

**Web App**

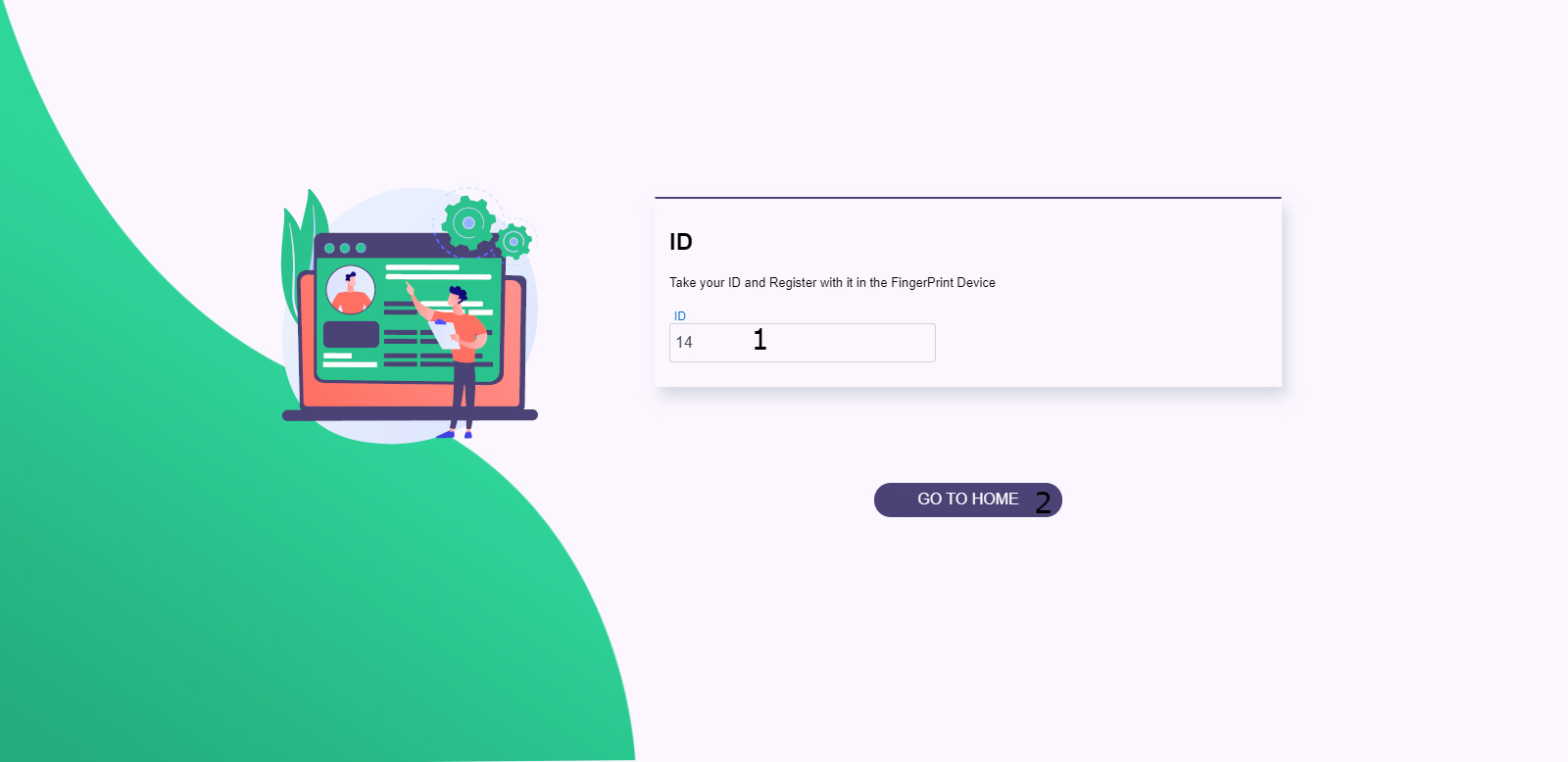
A picture containing diagram

Description automatically generated

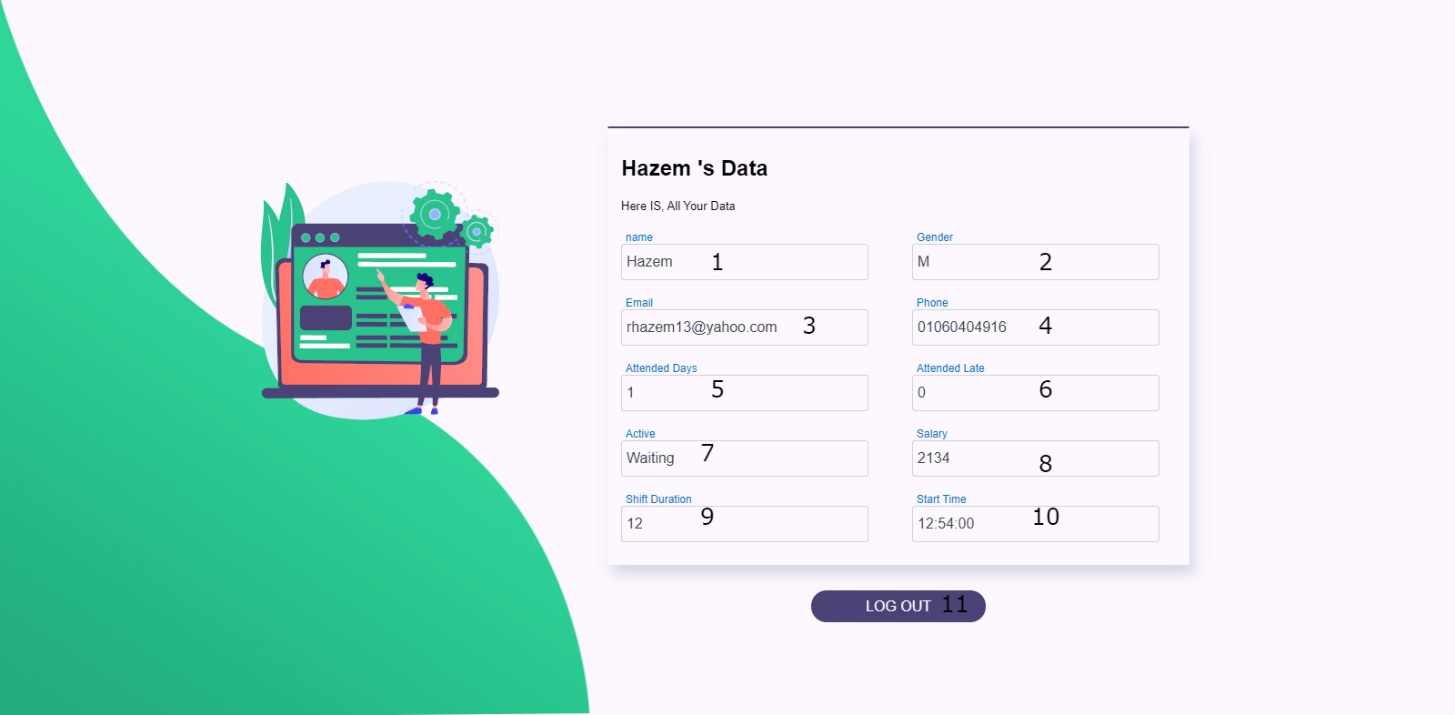
1. **Username.**
2. **Password.**
3. **Login in as (Admin/Employee).**
4. **Login Button.**



1. **Employee Name.**
2. **Employee Phone.**
3. **Employee Gender.**
4. **Employee E-mail.**
5. **Employee Password.**

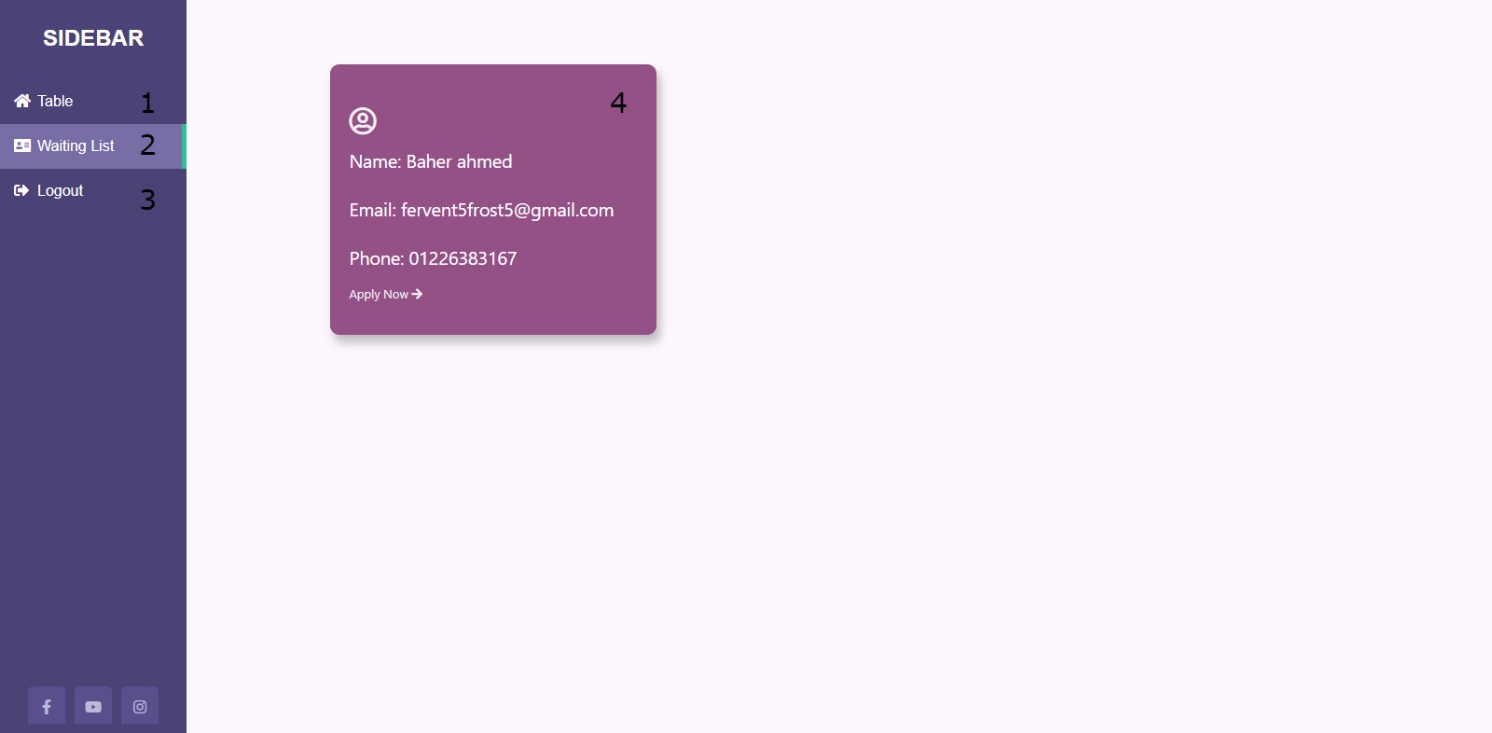


1. **Employee ID (Used to register the Fingerprint).**
2. **Go To Home page button.**

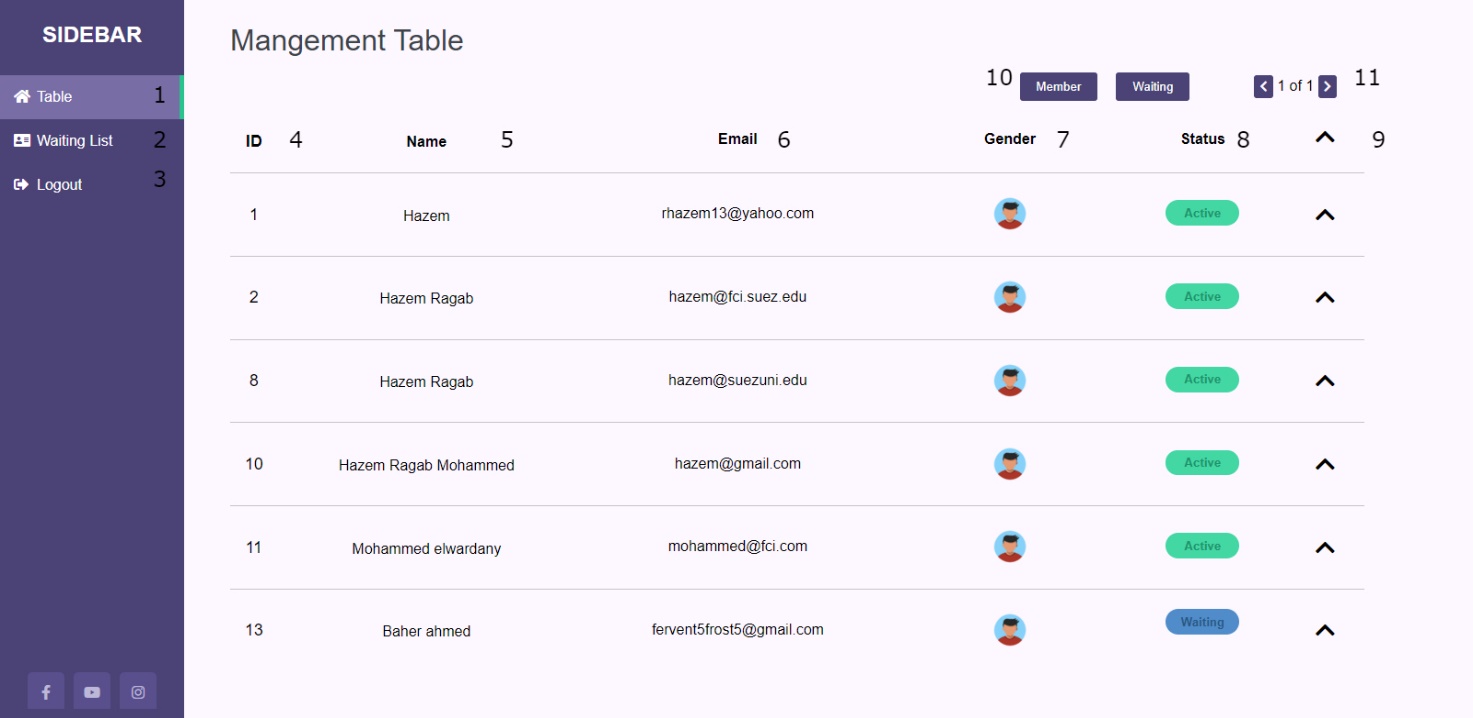


1. **Employee Name.**
2. **Employee Gender.**
3. **Employee E-Mail.**
4. **Employee Phone.**
5. **Attendance Days.**
6. **Late Attendance Days.**
7. **Status (Active / Waiting).**
8. **Days Salary.**
9. **Shift Duration.**

**10.Shift Time.**

****

1. **Table Button.**
2. **Waiting List Button.**
3. **Log out Button.**
4. **Waiting Employee Card.**

****

1. **Table Button.**
2. **Waiting List Button.**
3. **Log out Button.**
4. **IDs.**
5. **Names.**
6. **E-Mails.**
7. **Gender.**
8. **Status.**
9. **Expand & Collapse (To show or hide more data).**

**10. Searching By (Active / Waiting).**

**11. Pages Navigation.**

|  |
| --- |
| **11.**  **Appendices** |

**10.Appendices**

* **C**

**Definition:**

is a procedural programming language.

**Description:**

It was initially developed by Dennis Ritchie in the year 1972. It was mainly developed as a system programming language to write an operating system. The main features of the C language include low-level memory access, a simple set of keywords, and a clean style, these features make C language suitable for system programming like an operating system or compiler development.

Many later languages have borrowed syntax/features directly or indirectly from the C language. Like syntax of Java, PHP, JavaScript, and many other languages are mainly based on the C language. C++ is nearly a superset of C language (Few programs may compile in C, but not in C++).

**Structure of a C program:**

1. **Header Files Inclusion:**

The first and foremost component is the inclusion of the Header files in a C program.

A **header file** is a file with extension **.h** which contains C function declarations and macro definitions to be shared between several source files.

**Some of C Header files:**

- stdio.h – Defines core input and output functions.

-string.h – Defines string handling functions.

- math.h – Defines common mathematical functions.

1. **Main Method Declaration:**

The next part of a C program is to declare the main() function.

Like:

*int main()*

*{}*

1. **Variable Declaration:**

It refers to the variables that are to be used in the function. Please note that in the C program, no variable can be used without being declared. Also in a C program, the variables are to be declared before any operation in the function.

Example:

*Int a;*

1. **Body:**

refers to the operations that are performed in the functions. It can be anything like manipulations, searching, sorting, printing.

1. **Return Statement:**

refers to the returning of the values from a function. This return statement and return value depend upon the return type of the function. For example, if the return type is void, then there will be no return statement. In any other case, there will be a return statement and the return value will be of the type of the specified return type.

**Used for:**

used to develop system applications that forms major portion of operating systems such as Windows, UNIX and Linux. Operating systems, C compiler and all UNIX application programs are written in C language. Below are some examples of uses of C language.

* Database systems
* Graphics packages
* Word processors
* Spread sheets
* Operating system development
* Compilers and Assemblers
* Network drivers
* Interpreters
* Main languages use in Arduino

**Note:**

(**Procedural language:** is a computer programming language that follows, in order, a set of commands.

Examples of computer procedural languages are BASIC, C, FORTRAN, Java, and Pascal.)

* **Web Service:**

**Definition:**

open standard (XML, SOAP, HTTP.) based web applications that interact with other web applications for the purpose of exchanging data. Web services can convert your existing applications into web applications. In this tutorial, you will learn what exactly web services are and why and how to use them.

**How Used:**

Enables communication among various applications by using open standards such as HTML (***HyperText Markup Language***), XML (***Extensible Markup Language***), WSDL (***Web Services Description Language***), and SOAP/Restful. A web service takes the help of:

-XML / JSON to tag the data

-SOAP / Restful to transfer a message.

-WSDL to describe the availability of service.

You can build a Java-based web service on Solaris that is accessible from your Visual Basic program that runs on Windows.

* **Restful (*Representational State Transfer*):**

is an architectural style not a protocol used to transfer a message or data from, to hardware and software.

* **SOAP (*Simple Object Access Protocol*):**

-it is a XML-based protocol for accessing web services.

-SOAP is a W3C recommendation for communication between two applications.

-SOAP is XML based protocol. It is platform independent and language independent. By using SOAP, you will be able to interact with other programming language applications.

* **This Table will show the differences between SOAP and REST web services.**

|  |  |
| --- | --- |
| **SOAP** | **REST** |
| **can't use REST** because it is a protocol. | **can use SOAP** web services because it is a concept and can use any protocol like HTTP, SOAP. |
| **JAX-WS** is the java API for SOAP web services. | **JAX-RS** is the java API for RESTful web services. |
| **uses services interfaces to expose the business logic**. | **uses URI to expose business logic**. |
| **permits XML** data format only. | REST **permits different** data format such as Plain text, HTML, XML, JSON. |
| is **less preferred** than REST. | **more preferred** than SOAP. |
| **requires more bandwidth** and resource than REST. | **requires less bandwidth** and resource than SOAP. |
| **defines its own security**. | RESTful web services **inherits security measures** from the underlying transport. |

* **Flask:**

Flask is a web application framework written in Python. It is developed by

Armin Ronacher, who leads an international group of Python enthusiasts named Pocco.

Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine.

Both are Pocco projects. Flask is often referred to as a micro framework.

It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have form a validation support. Instead, Flask supports the extensions to add such functionality to the application.

* **WSGI:**

Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

* **Werkzeug:**

It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

* **Jinja2:**

Jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

* **Python:**

**Definition:**

Python is a high-level, general-purpose and a very popular programming language. Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry. Python Programming Language is very well suited for Beginners, also for experienced programmers with other programming languages like C++ and Java.

**Some Fact of Python:**

1. Python is currently the most widely used multi-purpose, high-level programming language.

2. Python allows programming in Object-Oriented and Procedural paradigms.

3. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

1. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber….
2. The biggest strength of Python is huge collection of standard library which can be used for the following:

* Machine Learning.
* GUI Applications (like Kivy, Tkinter, PyQt).
* Web frameworks like Django (used by YouTube, Instagram, Dropbox).
* Image processing (like OpenCV, Pillow).
* Web scraping (like Scrapy, BeautifulSoup, Selenium).
* Test frameworks.
* **REST API using Flask in Python**

Flask is a popular micro framework for building web applications. Since it is a micro-framework, it is very easy to use and lacks most of the advanced functionality which is found in a full-fledged framework. Therefore, building a REST API in Flask is very simple.

**There are two ways of creating a REST API in Flask:**

1-Using Flask without any external libraries.

2-Using flask\_restful library.

**using only Flask:**

there are two functions: One function to just return or print the data sent through GET or POST and another function to calculate the square of a number sent through GET request and print it.

**Using flask-restful:**

Flask Restful is an extension for Flask that adds support for building REST APIs in Python using Flask as the back-end. It encourages best practices and is very easy to set up. Flask restful is very easy to pick up if you’re already familiar with flask.

In flask-restful, the main building block is a resource. Each resource can have several methods associated with it such as GET, POST, PUT, DELETE.

for example, there could be a resource that calculates the square of a number whenever a get request is sent to it. Each resource is a class that inherits from the Resource class of flask-restful.

Once the resource is created and defined, we can add our custom resource to the API and specify a URL path for that corresponding resource.

**Cookie:**

A cookie is stored on a client’s computer in the form of a text file.

Its purpose is to remember and track data pertaining to a client’s usage for better visitor experience and site statistics.

A Request object contains a cookie’s attribute. It is a dictionary object of all

the cookie variables and their corresponding values, a client has transmitted.

In addition to it, a cookie also stores its expiry time, path and domain name of the site.

In Flask, cookies are set on response object.

Use make\_response () function to get response object from return value of a view function. After that, use the set\_cookie () function of response object to store a cookie.

Reading back a cookie is easy.

The get () method of request.cookies attribute is used to read a cookie.

is used to read a cookie.

**Session:**

Like Cookie, Session data is stored on client. Session is the time interval when a client logs into a server and logs out of it. The data, which is needed to be held across this session, is stored in the client browser.

A session with each client is assigned a Session ID. The Session data is stored on top of cookies and the server signs them cryptographically.

For this encryption, a Flask application needs a defined SECRET\_KEY.

Session object is also a dictionary object containing key-value pairs of session variables and associated values.

* **JWT:**

A ***JSON web token*** (JWT) is JSON Object which is used to securely transfer information over the web (between two parties). It can be used for an authentication system and can also be used for information exchange.

The token is mainly composed of header, payload, signature. These three parts are separated by dots(.).

JWT defines the structure of information we are sending from one party to the another, and it comes in two forms – Serialized, Deserialized.

The Serialized approach is mainly used to transfer the data through the network with each request and response. While the deserialized approach is used to read and write data to the web token.

**Deserialized:**

JWT in the deserialized form contains only the header and the payload.Both of them are plain JSON objects.

**Header:**

A header in a JWT is mostly used to describe the cryptographic operations applied to the JWT like signing/decryption technique used on it. It can also contain the data about the media/content type of the information we are sending.

-This information is present as a JSON object then this JSON object is encoded to BASE64URL. The cryptographic operations in the header define whether the JWT is signed/unsigned or encrypted and are so then what algorithm techniques to use.

**Like:**

*{*

*"typ":"JWT",*

*"alg":"HS256"*

*}*

The ‘alg’ and ‘typ’ are object key’s having different values and different functions like the ‘typ’ gives us the type of the header this information packet is, whereas the ‘all’ tells us about the encryption algorithm used.

-Some JWT’s can also be created without a signature or encryption. Such a token is referred to as unsecured and its header should have the value of the ‘alg’ object key assigned to as ‘none’.

**Like:**

*{*

*"alg":"none"*

*}*

**Payload:**

The payload is the part of the JWT where all the user data is actually added. This data is also referred to as the ‘claims’ of the JWT.This information is readable by anyone so it is always advised to not put any confidential information in here. This part generally contains user information. This information is present as a JSON object then this JSON object is encoded to BASE64URL. We can put as many claims as we want inside a payload, though unlike header, no claims are mandatory in a payload.

**Serialized:**

JWT in the serialized form represents a string of **the following format:**

*[header].[payload].[signature]*

**Signature:**

This is the third part of JWT and used to verify the authenticity of token. BASE64URL encoded header and payload are joined together with dot(.) and it is then hashed using the hashing algorithm defined in a header with a secret key. This signature is then appended to header and payload using dot(.) which forms our actual token header. payload. Signature.

**Syntax:**

*{HASHINGALGO( base64UrlEncode(header) + “.” + base64UrlEncode(payload),secret)}*

|  |
| --- |
| **12.**  **index** |

|  |  |  |
| --- | --- | --- |
| **No.** | **index** | **page** |
| **1** | **Preface** | 1 |
| **2** | **Introduction** | 5 |
| **3** | **(SRS) system requirements** | 7 |
| **4** | **Glossary** | 13 |
| **5** | **user story** | 19 |
| **6** | **User requirements definition** | 22 |
| **7** | **System Evolution** | 30 |
| **8** | **System architecture** | 34 |
| **9**  **10** | **System Models**  **Wireframes** | 38  41 |
| **11** | **Appendices** | 56 |
| **12** | **index** | 70 |