

SOFTWARE ENGINEERING PROJECT

**ON**

**EMPLOYEE LEAVE MANAGEMENT SYSTEM**

*Project 2*

*Submitted to*

*Dr. Biswajit R. Bhowmik*

***Department of Computer Science & Engineering***

***IIIT BHAGALPUR, BIHAR 813210, INDIA***

*TEAM MEMBERS*

|  |  |  |
| --- | --- | --- |
| ***Database Administrator*** | *SURAJ KUMAR* | 170101053 |
| ***UI Specialist*** | *MD. ASHIF REZA* | *170101023* |
| ***Quality Assurance Specialist*** | *AMAN MISHRA* | *170101008* |
| ***Project Leader*** | *RASHI KRISHNA* | *170102037* |

**Certificate**

This is to certify that the group of following students of 3rd year 5th semester in the Branch of Computer Science and Engineering has successfully completed a project entitled " **EMPLOYEE LEAVE MANAGEMENT SYSTEM** " under the guidance of Dr. Biswajit R. Bhowmik during the academic year 2019-2020. The project had been prescribed towards automation of the existing system at the Institute and for consideration in partial fulfilment of the Software Engineering course curriculum of the Institute leading to the award of semester examination of July – December 2019.

The project work is originally carried out by them, a bona fide work of them, and may be considered as their mark of proficiency at the beginning level in Software Development.

**Project Members**

1. RASHI KRISHNA (170102037), Team Leader
2. AMAN MISHRA (170101008)
3. Md. ASHIF REZA (170101023)
4. SURAJ KUMAR (170101053)

|  |  |  |
| --- | --- | --- |
| Dr. Biswajit R. Bhowmik  Project Supervisor |  | Dr. Pradeep Kumar Biswal  Faculty-in-Charge |

TABLE OF CONTENT

[1. INTRODUCTION 1](#_Toc25543188)

[1.1 DOCUMENT SCOPE AND PURPOSE 1](#_Toc25543189)

[1.2 TARGET AUDIENCE 1](#_Toc25543190)

[2. REQUIREMENT ANALYSIS 2](#_Toc25543191)

[2.1 SYSTEM REQUIREMENT 2](#_Toc25543192)

[2.2 SOFTWARE REQUIREMENTS 2](#_Toc25543193)

[2.2.1 PYTHON3 2](#_Toc25543194)

[2.2.2 PYTHON3 LIBRARIES 3](#_Toc25543195)

[2.2.3 HEROKU 4](#_Toc25543196)

[2.2.4 POSTGRESQL 4](#_Toc25543197)

[2.3 HOW TO INSTALL PYTHON3 AND ITS LIBRARIES 5](#_Toc25543198)

[2.3.1 Installation In UBUNTU 5](#_Toc25543199)

[2.3.2 Installation In WINDOWS 6](#_Toc25543200)

[3. DESIGN 7](#_Toc25543201)

[3.1 DESIGN APPROACH 7](#_Toc25543202)

[3.2 DESIGN PATTERNS 7](#_Toc25543203)

[3.3 FLOW CHARTS 8](#_Toc25543204)

[3.3.1 Activity Diagram for ADMINS 8](#_Toc25543205)

[3.3.2 Activity Diagram for FACULTIES 12](#_Toc25543206)

[3.3.3 Activity Diagram for STAFFS 16](#_Toc25543207)

[3.3.4 Activity Diagram for OTHERS 20](#_Toc25543208)

[4. IMPLEMENATION 24](#_Toc25543209)

[4.1 DATA DICTIONARY 24](#_Toc25543210)

[4.2 SNAPSHOTS OF WORKING APPLICATION 29](#_Toc25543211)

[4.2.1 Home Page 29](#_Toc25543212)

[4.2.2 Signup 30](#_Toc25543213)

[4.2.3 Login 31](#_Toc25543214)

[4.2.4 Reset Password 32](#_Toc25543215)

[4.2.5 Profile of HOD 33](#_Toc25543216)

[4.2.6 History of Applicants(Faculty, staff, others) 34](#_Toc25543217)

[4.2.7 Profile of Admin 35](#_Toc25543218)

[4.2.8 Leave Application Form 36](#_Toc25543219)

[4.2.9 Station Leave Form 37](#_Toc25543220)

[4.2.10 Extend Leave Application 38](#_Toc25543221)

[4.2.11 Re-joining Form 39](#_Toc25543222)

[4.2.12 The Use of Mail System 40](#_Toc25543223)

[5. TESTING 41](#_Toc25543224)

[5.1 TESTING GOAL 41](#_Toc25543225)

[5.2 FUNCTIONAL REQUIREMENTS TESTING (BLACK BOX TESTING) 41](#_Toc25543226)

[5.2.1 Registration for New User 41](#_Toc25543227)

[5.2.2 Apply for Leave (Faculty) 42](#_Toc25543228)

[5.2.3 Approving / Disapproving Leave 42](#_Toc25543229)

[5.2.4 Test Case for Login Page 43](#_Toc25543230)

[5.2.5 Test Case for Approve/Disapprove, Manage Faculty Page 43](#_Toc25543231)

[5.2.6 Test Case for Leave Page 44](#_Toc25543232)

[5.3 Non- Functional Requirements Testing 45](#_Toc25543233)

[5.3.1 Stability Testing 45](#_Toc25543234)

[5.3.2 Usability 45](#_Toc25543235)

[5.3.3 Security Testing 45](#_Toc25543236)

[6. MAINTAINENCE 46](#_Toc25543237)

[7. CONCLUSION 47](#_Toc25543238)

# 1. INTRODUCTION

*In the existing paper work related to leave management, leaves are maintained using the attendance register for staff. The staff needs to submit their leaves manually to their respective authorities. This increases the paperwork & maintaining the records becomes tedious. Maintaining notices in the records also increases the paperwork. The main objective of the proposed system is to decrease the paperwork and help in easier record maintenance by having a centralized Database System, where Leaves and Notices are maintained. The proposed system automates the existing system. It decreases the paperwork and enables easier record maintenance. It also reduces chances of Data loss.*

## 1.1 DOCUMENT SCOPE AND PURPOSE

This document provides a description of the technical design for Employee Leave Management System of Indian Institute of Information Technology. This document provides an architectural overview of the system to depict different aspects of the system. This document also functions as a foundational reference point for developers*.*

## 1.2 TARGET AUDIENCE

This document is targeted (but not limited) to technical stakeholders:

* Development Team
* Clients
* Support Staff

It is assumed that the reader has a technical background in software design and development.

# 2. REQUIREMENT ANALYSIS

## 2.1 SYSTEM REQUIREMENT

* Development: Python3 + HTML + CSS + JavaScript
* Unit Test: unittest(Python)
* Diagrams: Lucid Chart
* Database Management: Adminer
* Database: PostgreSQL
* Server: Python Server
* Discussion: WhatsApp Groups

## 2.2 SOFTWARE REQUIREMENTS

### 2.2.1 PYTHON3

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is named after a TV Show called ëMonty Pythonís Flying Circusí and not after Python-the snake.

Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. Following are important characteristics of python −

* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* It supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

### 2.2.2 PYTHON3 LIBRARIES

#### 2.2.2.1 Flask:

Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. Flask offers suggestions but doesn’t enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

#### 2.2.2.2 SMTPLib:

SMTPLib module of python defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon. An SMTP object has an instance method called “SENDMAIL”, which is used to do the work of mailing a message.

#### 2.2.2.3 SQLAlchemy:

SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL. SQLAlchemy provides a full suite of well-known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language.

#### 2.2.2.4 Psycopg2:

Psycopg is the most popular PostgreSQL database adapter for the Python programming language. Its main features are the complete implementation of the Python DB API 2.0 specification and the thread safety. It was designed for heavily multi-threaded applications that create and destroy lots of cursors and make a large number of concurrent “INSERT(s)” or “UPDATE(s)”.

### 2.2.3 HEROKU

Heroku is a container-based cloud Platform as a Service (PaaS). Developers use Heroku to deploy, manage, and scale modern apps. Its platform is elegant, flexible, and easy to use, offering developers the simplest path to getting their apps to market.

Heroku is fully managed, giving developers the freedom to focus on their core product without the distraction of maintaining servers, hardware, or infrastructure. The Heroku experience provides services, tools, workflows, and polyglot support—all designed to enhance developer productivity.

### 2.2.4 POSTGRESQL

PostgreSQL is a general purpose and object-relational database management system, the most advanced open source database system. It allows you to add custom functions developed using different programming languages such as C/C++, Java, etc.

PostgreSQL was designed to be extensible. In PostgreSQL, one can define his own data types, index types, functional languages, etc. It requires very minimum maintained efforts because of its stability.  Therefore, if a person develop applications based on PostgreSQL, the total cost of ownership is low in comparison with other database management systems.

## 2.3 HOW TO INSTALL PYTHON3 AND ITS LIBRARIES

### 2.3.1 Installation In UBUNTU

* Open terminal via Ctrl+Alt+T or searching for “Terminal” from app launcher. When it opens, run command to add the PPA:
  + sudo add-apt-repository ppa:jonathonf/python-3.6
* Then check updates and install **Python 3.6** via commands:
  + sudo apt-get update
  + sudo apt-get install python3.6
* Use the following command to install **pip** for Python 3:
  + sudo apt install python3-pip
* Once the installation is complete, verify the installation by checking the pip version:
  + pip3 --version
* Now install the **python3-venv** package that creates a virtual environment using the following command:
  + sudo apt install python3-venv
* Start by navigating to the directory where you would like to store your Python 3 virtual environments. It can be your home directory or any other directory where your user has read and write permissions. Create a new directory for your Flask application and navigate into it:
  + mkdir my\_flask\_app
  + cd my\_flask\_app
* Once inside the directory, run the following command to create your new virtual environment:
  + python3 -m venv venv
* To start using this virtual environment, you need to activate it by running the **activate** script:
  + source venv/bin/activate
* Now that the virtual environment is activated, you can use the Python package manager pip to install **Flask**:
  + pip install Flask
* Verify the installation with the following command which will print the Flask version:
  + python -m flask –version
* Similarly install **SMPTLib, SQLAlchemy, Psycopg2** and the other basic python libraries required for the development of a web application.

### 2.3.2 Installation In WINDOWS

* Go to the [https://visualstudio.microsoft.com/downloads](https://visualstudio.microsoft.com/downloads ) page to install Visual Studios for free.
* In the installer, select the Python development workload.
* Open Visual Studio 2019.
* On the start screen, select **Create a new project**.
* In the **Create a new project** dialog box, enter "Python web" in the search field at the top, choose **Web Project** in the middle list, then select **Next**.
* If you don't see the Python project templates, run the **Visual Studio Installer**, select **More** > **Modify**, select the **Python development** workload, then choose **Modify**.
* In the **Configure your new project** dialog that follows, enter "HelloPython" for **Project name**, specify a location, and select **Create**. (The **Solution name** is automatically set to match the **Project name**.)
* The new project opens in **Solution Explorer** in the right pane. The project is empty at this point because it contains no other files.
* Expand the **Python Environments** node in the project to see the default environment for the project.
* Right-click the environment and select **Manage Python Packages...**. This command opens the **Python Environments** window on the **Packages (PyPI)** tab.
* Enter "flask" in the search field. If **Flask** appears below the search box, you can skip this step. Otherwise select **Run command: pip install flask**. Accept any prompts for administrator privileges and observe the **Output** window in Visual Studio for progress. (A prompt for elevation happens when the packages folder for the global environment is located within a protected area like *C:\Program Files*.)
* Similarly install **SMPTLib, SQLAlchemy, Psycopg2** and the other basic python libraries required for the development of a web application.
* Once installed, the library appears in the environment in **Solution Explorer**, which means that you can make use of it in Python code.

# 3. DESIGN

## 3.1 DESIGN APPROACH

The design approach used here is based on the following:

1. DATA FLOW DESIGN: The data flow of the *Employee Leave Management System* is Internet-based.
2. ARCHITECHTURE DESIGN: The *Employee Leave Management System* is developed with the help of Python3 using Flask, SMPTLib, SQLAlchemy and some other basic python libraries. The database used is PostgreSQL, and the web application is deployed through Heroku.
3. UI DESIGN: The *Employee Leave Management System* uses HTML, CSS and JavaScript at the Front End. It has been made such that the web application is very user friendly and all the functions present will make the life of employees easier.

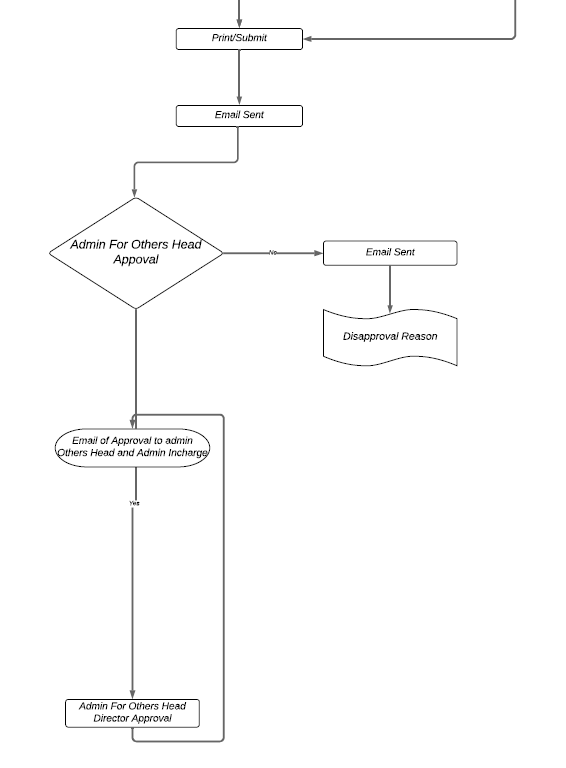
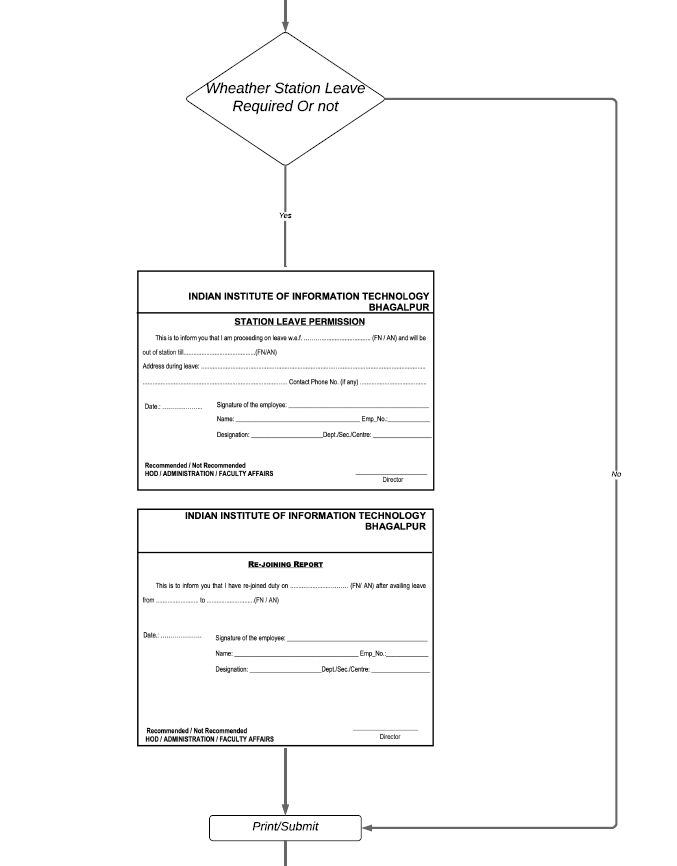
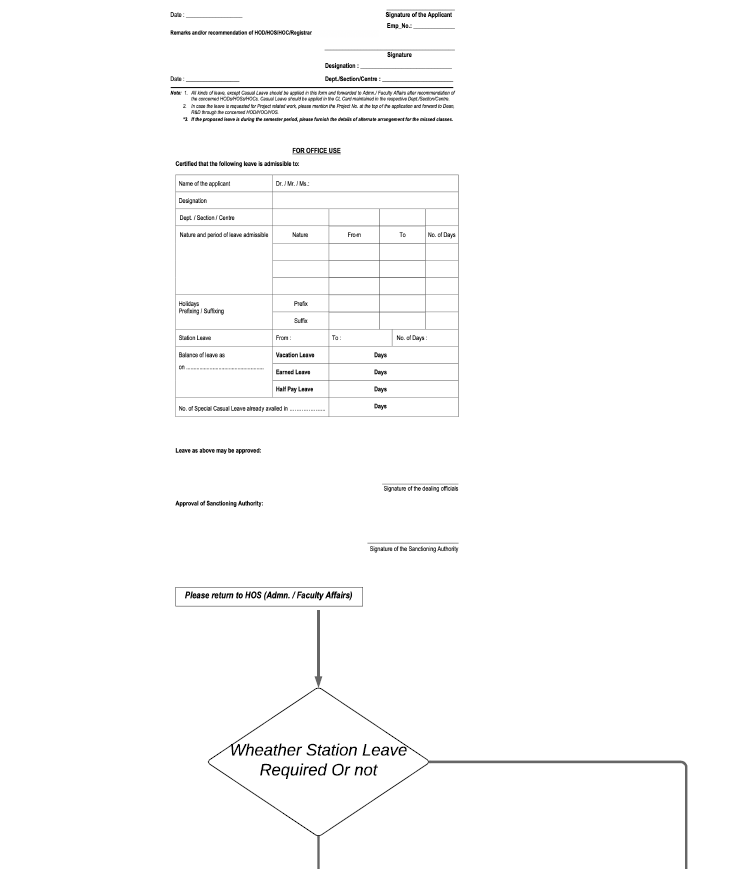
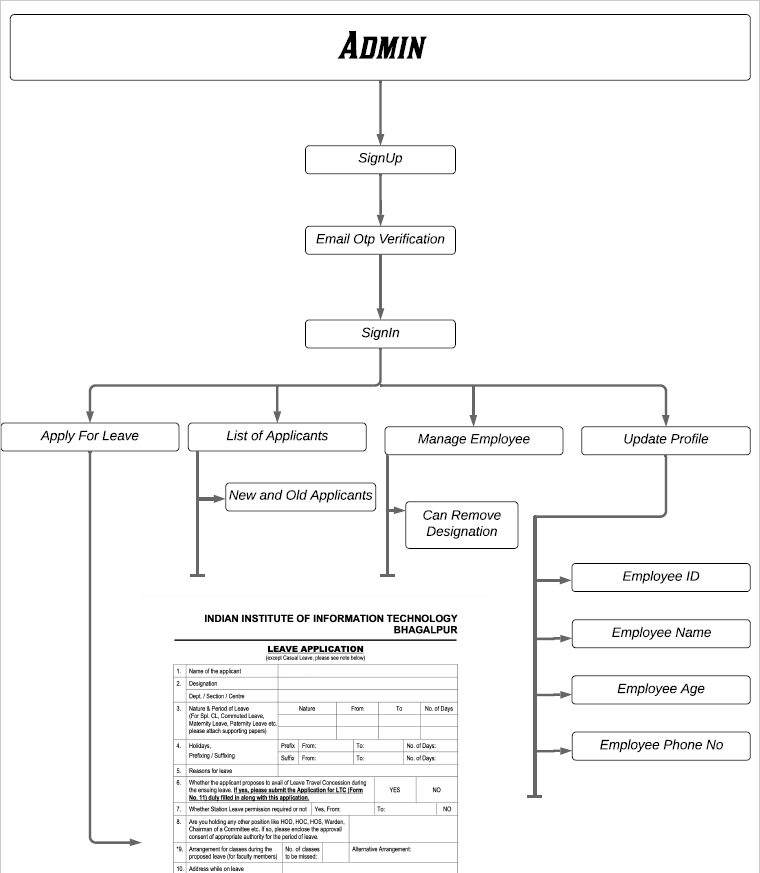
## 3.2 DESIGN PATTERNS

This application is designed as an object-oriented system for an Internet-based architecture by factoring application classes into the following layers:

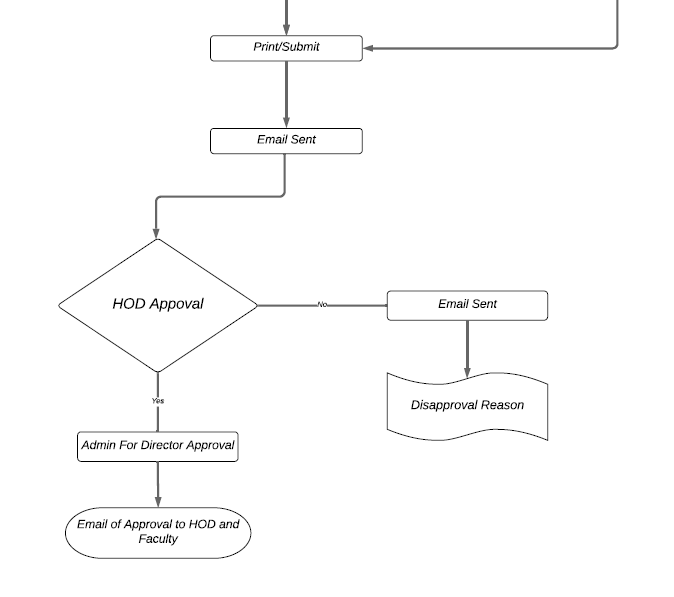
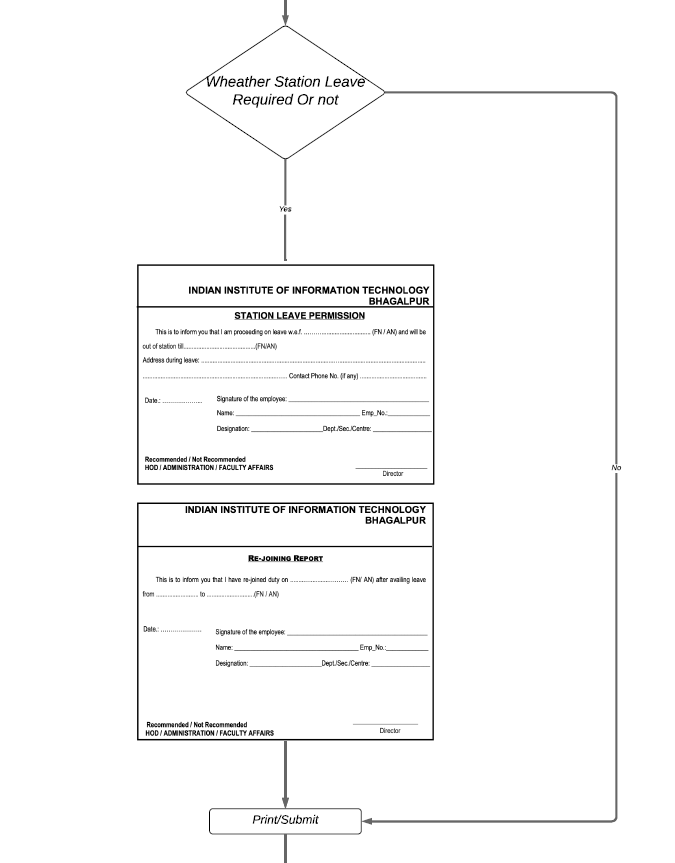
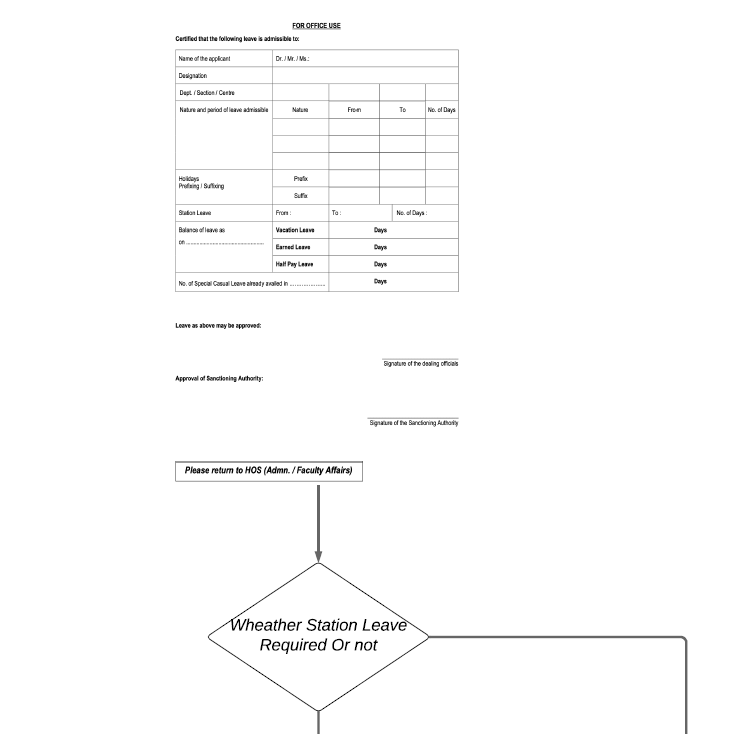
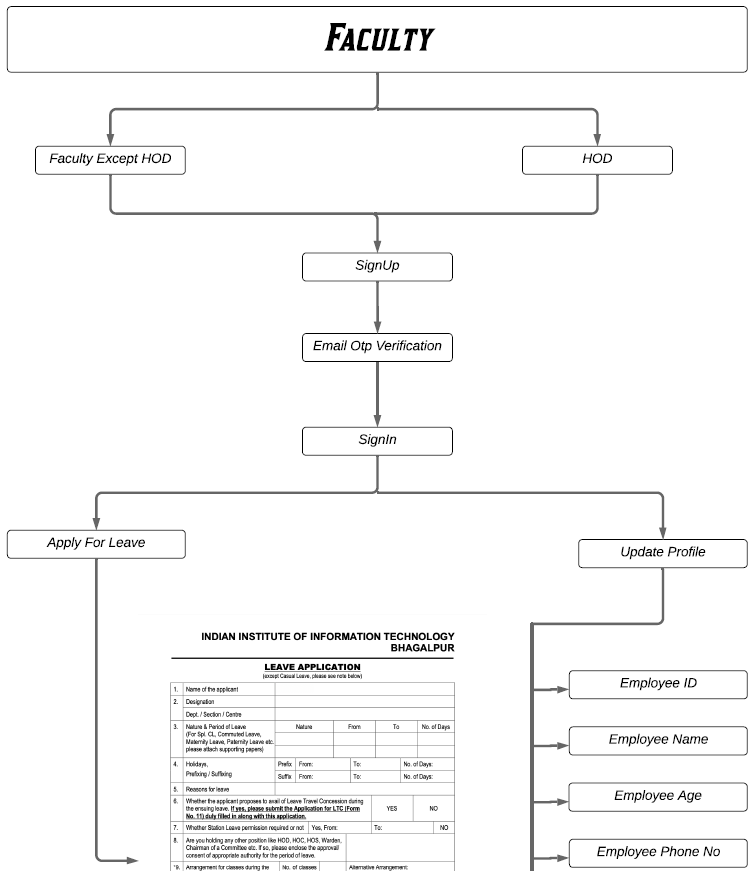
1. The Presentation layer: This is the layer where the physical window and widget objects live. Any new user interface widgets developed for this application are put in this layer.
2. THE DOMAIN MODE: Most objects identified in the OO analysis and design will reside. To a great extent, the objects in this layer can be application-independent. Generic objects may be used in this application to reap the benefits of Object Oriented programming.
3. THE DATA LAYER: The data is managed by PostgreSQL.

## 3.3 FLOW CHARTS

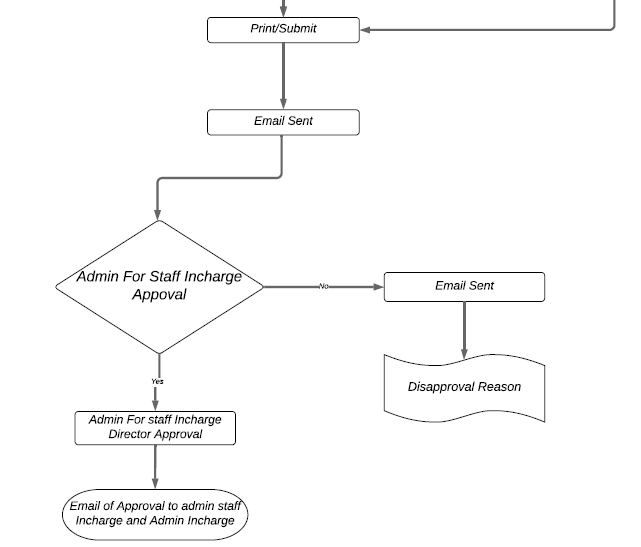
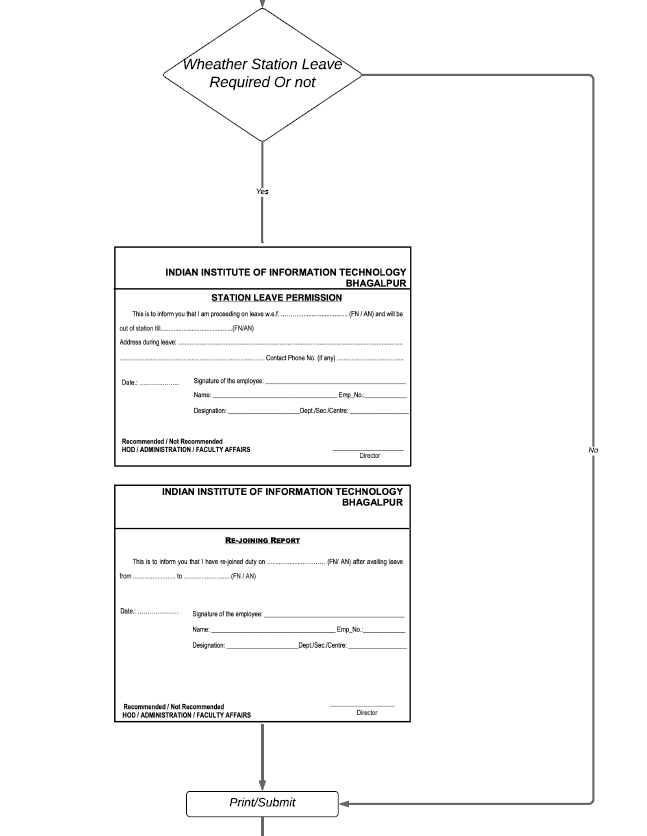
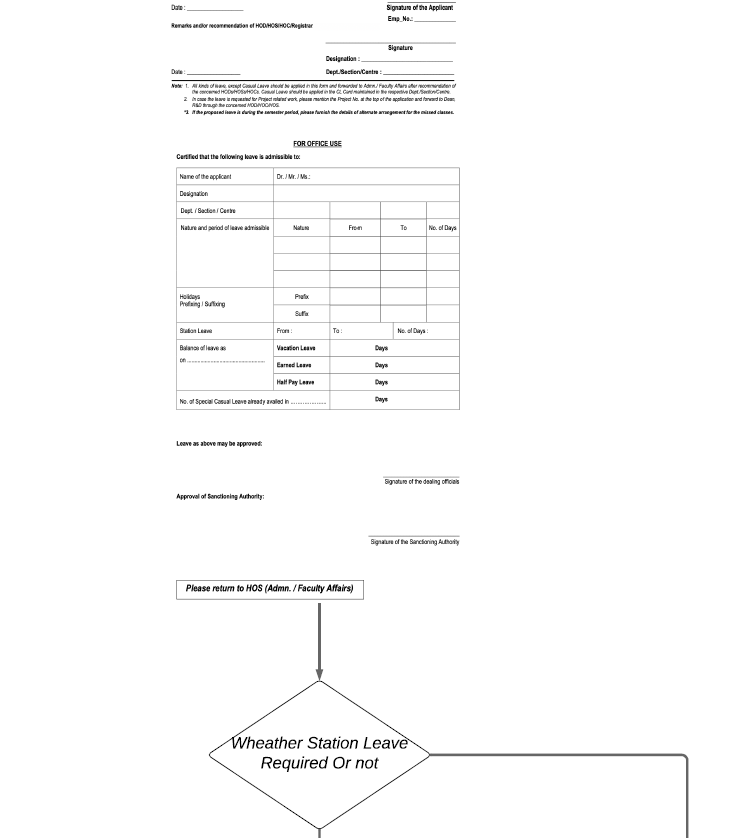
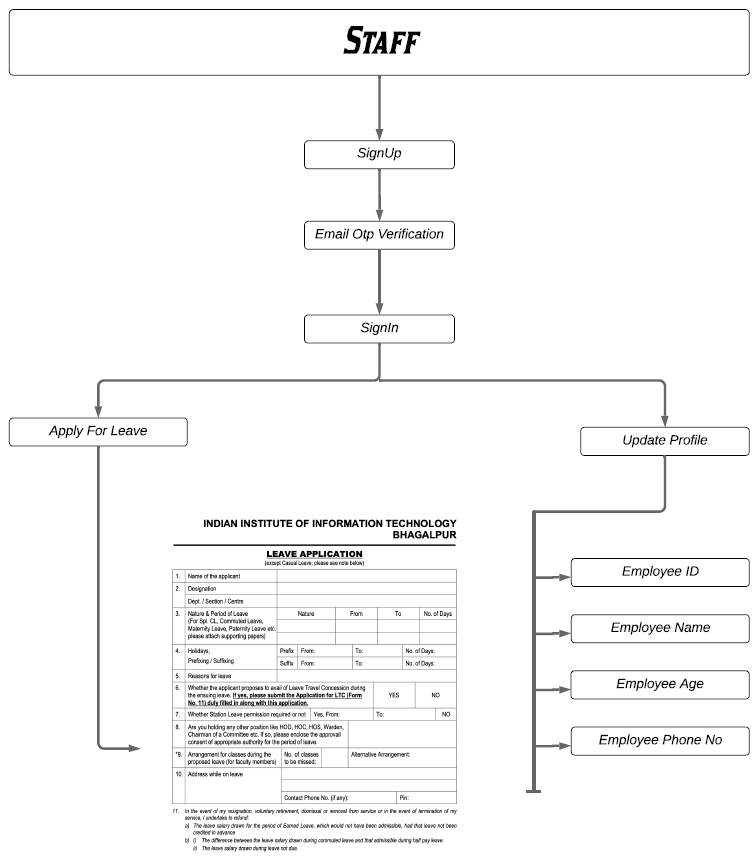
### 3.3.1 Activity Diagram for ADMINS



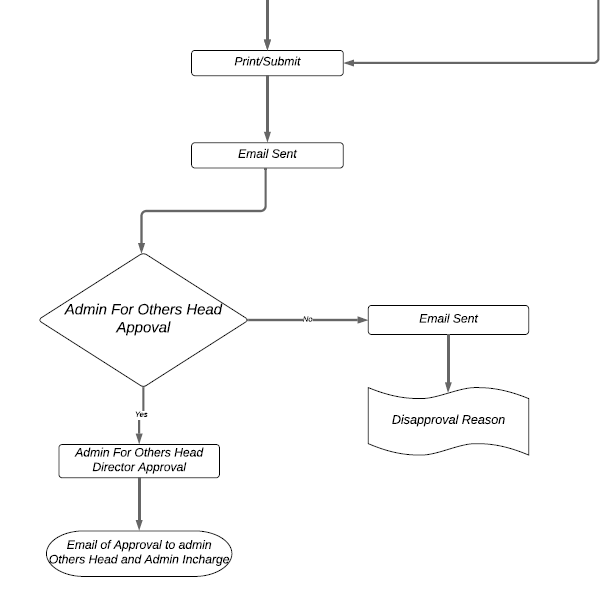
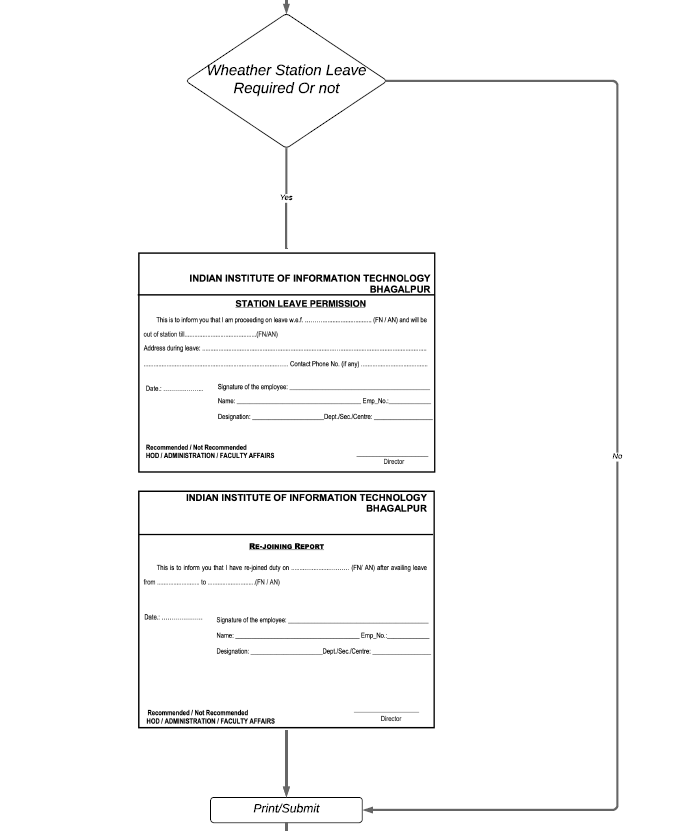
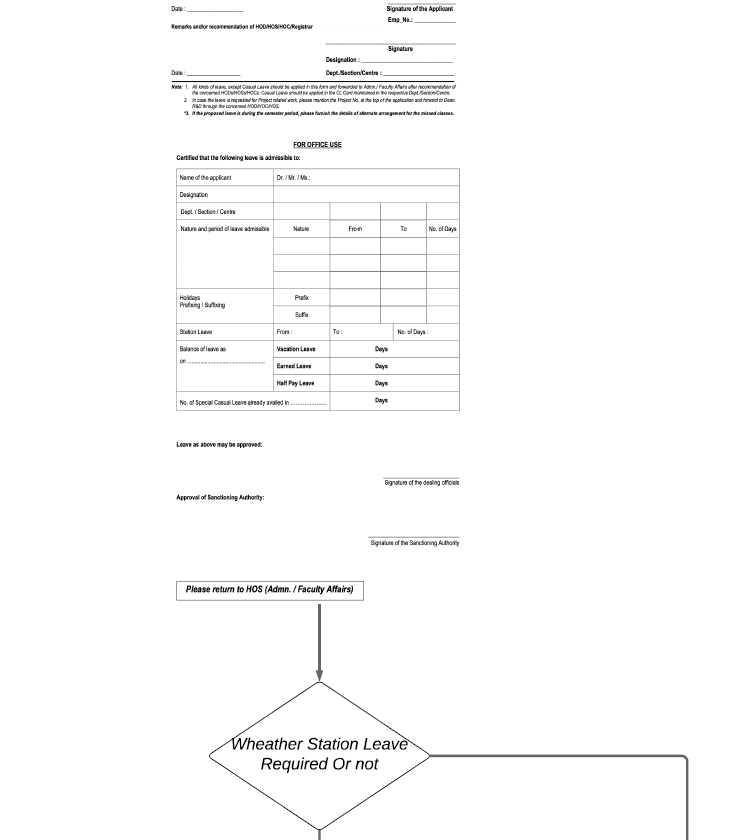
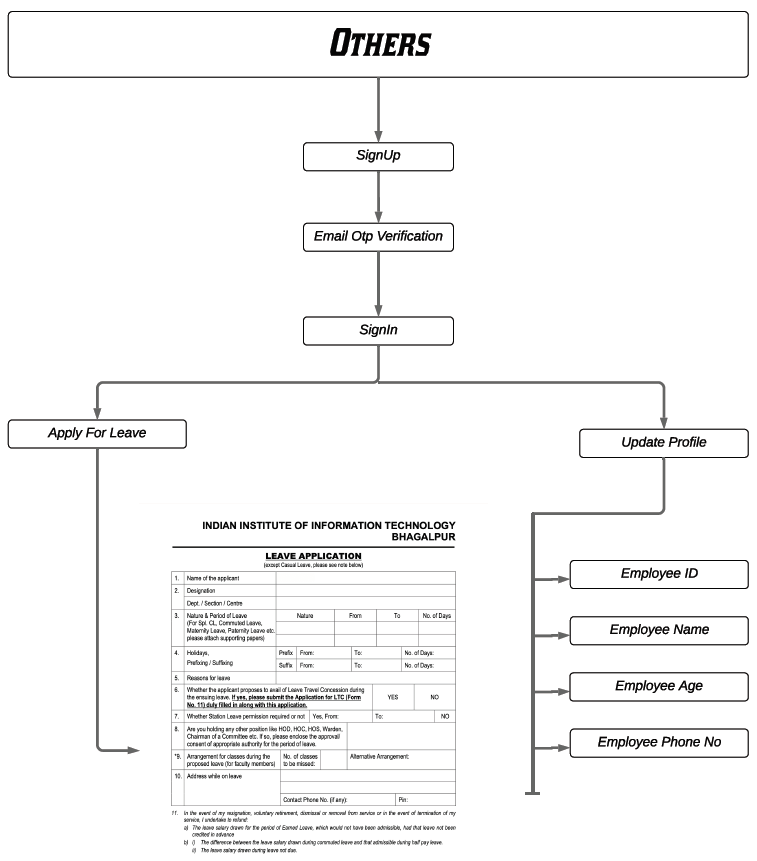
### 3.3.2 Activity Diagram for FACULTIES



### 3.3.3 Activity Diagram for STAFFS



### 3.3.4 Activity Diagram for OTHERS



# 4. IMPLEMENATION

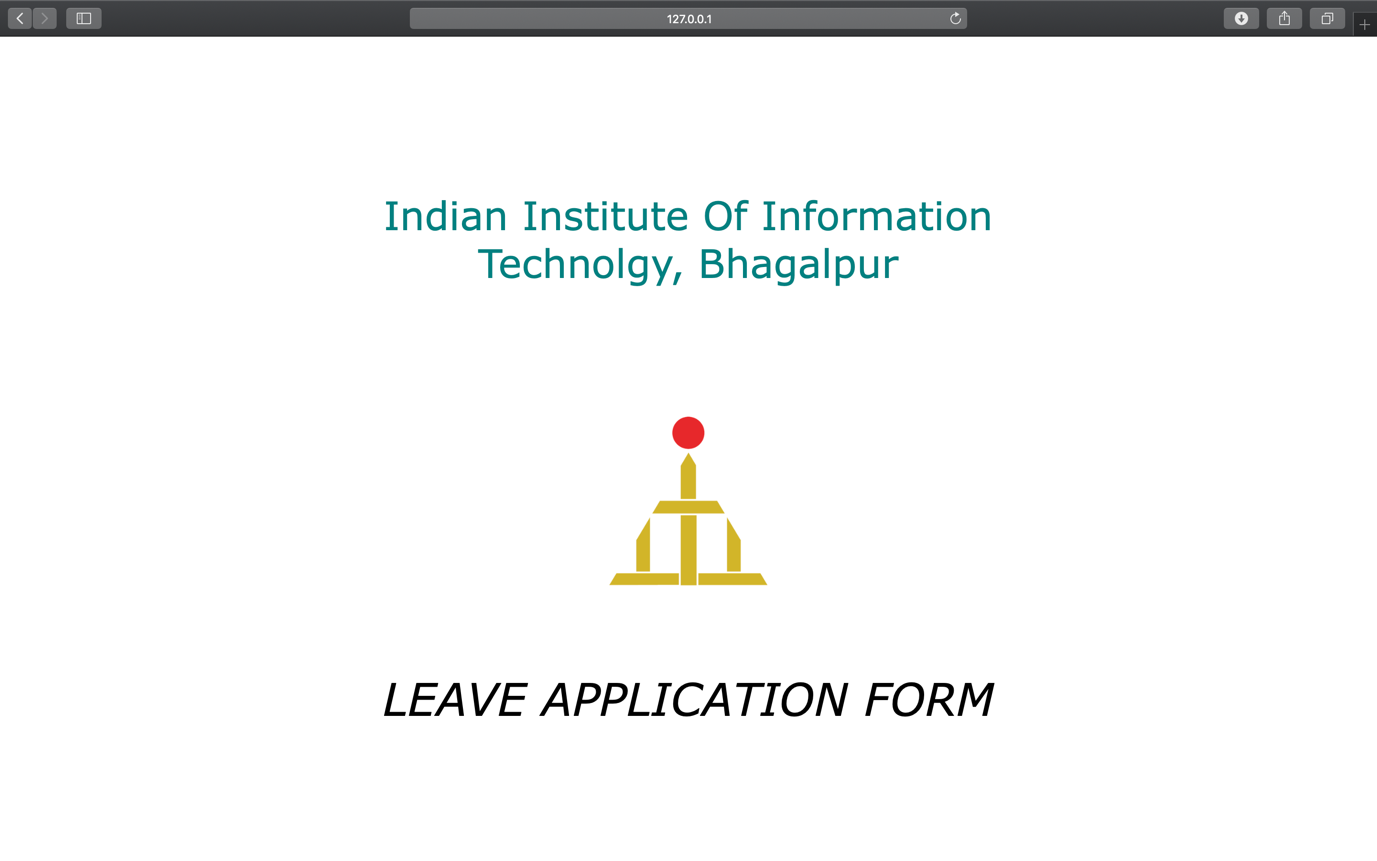
## 4.1 DATA DICTIONARY

|  |  |  |
| --- | --- | --- |
| TABLE | COLUMN | TYPE |
| **Admin** | id | integer *Auto Increment* [nextval('admin\_id\_seq')] |
| name | text |
| userid | character varying |
| password | character varying |
| verify | integer *NULL* [0] |
|  |  |  |
| **Admin\_Info** | id | integer |
| name | text |
| age | integer |
| gender | character(20) |
| position | text *NULL* |
|  |  |  |
| **Admin\_Leave** | id | integer |
| leave\_from | date |
| leave\_upto | date |
| approved | integer |
| nature | text |
| no\_of\_days | integer |
| reason | text |
| application\_no | integer NULL |
| prefix\_from | date NULL |
| prefix\_upto | date NULL |
| suffix\_from | date NULL |
| suffix\_upto | date NULL |
| sufix\_days | integer NULL |
| prefix\_days | integer NULL |
| travel\_concession | text NULL |
| station\_leave\_permission | text NULL |
| position | text NULL |
| missed\_classes | integer NULL |
| address | text NULL |
| district | text NULL |
| phone\_no | integer NULL |
| pin | integer NULL |
| name | smallint |
| status | integer NULL [0] |
| cur\_date | timestamp NULL |
| comment | text NULL |
|  |  |  |
| **Admin\_Rejoin** | rejoin\_date | date |
| id | integer |
|  |  |  |
| **Faculty** | id | integer Auto Increment [nextval('employee\_id\_seq')] |
| name | text |
| userid | character varying |
| password | character varying |
| verify | integer NULL [0] |
|  |  |  |
| **Faculty\_Info** | id | integer |
| name | text |
| age | integer |
| department | character varying |
| gender | character(20) |
| position | text NULL |
| available\_cl | integer NULL [12] |
| available\_pl | smallint NULL [18] |
| available\_el | smallint NULL [15] |
| available\_vl | smallint [20] |
|  |  |  |
| **Faculty\_Leave** | id | integer |
| leave\_from | date |
| leave\_upto | date |
| approved | smallint [0] |
| no\_of\_days | integer NULL |
| reason | text NULL |
| nature | text NULL |
| application\_no | integer NULL |
| prefix\_from | date NULL |
| prefix\_upto | date NULL |
| suffix\_from | date NULL |
| suffix\_upto | date NULL |
| prefix\_days | integer NULL |
| suffix\_days | integer NULL |
| travel\_conseesion | text NULL |
| station\_leave\_permission | text NULL |
| position | text NULL |
| missed\_classes | integer NULL |
|  |  |  |
| **Faculty\_Rejoin** | rejoin\_date | date |
| id | smallint |
|  |  |  |
| **Others** | id | integer *Auto Increment* [nextval('others\_id\_seq')] |
| name | text |
| userid | character varying |
| password | character varying |
| verify | integer *NULL* [0] |
|  |  |  |
| **Others\_Info** | id | integer |
| name | text |
| age | integer |
| gender | character varying |
| available\_cl | smallint NULL [12] |
| available\_el | smallint NULL [18] |
| available\_pl | smallint NULL [15] |
| available\_vl | smallint NULL [20] |
| department | character varying |
| position | character varying |
|  |  |  |
| **Others\_Leave** | id | integer |
| leave\_from | date |
| leave\_upto | date |
| approved | smallint |
| no\_of\_days | integer |
| reason | text |
| nature | text |
| prefix\_upto | date NULL |
| prefix\_from | date NULL |
| suffix\_upto | date NULL |
| suffix\_from | date NULL |
| prefix\_days | integer NULL |
| suffix\_days | integer NULL |
| travel\_conseesion | text NULL |
| station\_leave\_permission | text NULL |
| missed\_classes | integer NULL |
| position | text NULL |
| address | text NULL |
| district | text NULL |
| phone\_no | integer NULL |
| pin | integer NULL |
| name | text |
| application\_no | bigint |
| department | character varying |
| status | integer NULL [0] |
| cur\_date | timestamp NULL |
| comment | text NULL |
|  |  |  |
| **Others\_Rejoin** | rejoin\_date | date |
| id | smallint |
|  |  |  |
| **Staff** | id | integer Auto Increment [nextval('staff\_id\_seq')] |
| userid | text |
| password | text |
| name | text |
| verify | integer [0] |
|  |  |  |
| **Staff\_Info** | id | integer |
| name | text |
| age | integer |
| gender | text |
|  |  |  |
| **Staff\_Leave** | id | integer |
| leave\_from | date |
| leave\_upto | date |
| nature | text |
| no\_of\_days | integer |
| reason | text |
| application\_no | integer NULL |
| prefix\_from | date NULL |
| prefix\_upto | date NULL |
| suffix\_from | date NULL |
| suffix\_upto | date NULL |
| suffix\_days | integer NULL |
| prefix\_days | integer NULL |
| travel\_conseesion | text NULL |
| station\_leave\_permission | text NULL |
| position | text NULL |
| missed\_classes | integer NULL |
| address | text NULL |
| district | text NULL |
| phone\_no | integer NULL |
| pin | integer NULL |
| name | text NULL |
| approved | integer |
| status | integer NULL [0] |
| cur\_date | timestamp NULL |
| comment | text NULL |
|  |  |  |
| **Staff\_Rejoin** | id | integer |
| rejoin\_date | date |
|  |  |  |
| **Staff\_Leave** | id | integer |
| from\_date | date |
| to\_date | date |
| from\_time | time without time zone |
| to\_time | time without time zone |
| from\_type | character varying |
| to\_type | character varying |
| category | character varying |
|  |  |  |
| **Responsibility** | id | integer |
| assign\_duty\_to\_name | character varying |
| assign\_duty\_to\_email | character varying |
| assign\_duty\_to\_contact\_no | bigint NULL |
| category | character varying |

## 4.2 SNAPSHOTS OF WORKING APPLICATION

### 4.2.1 Home Page

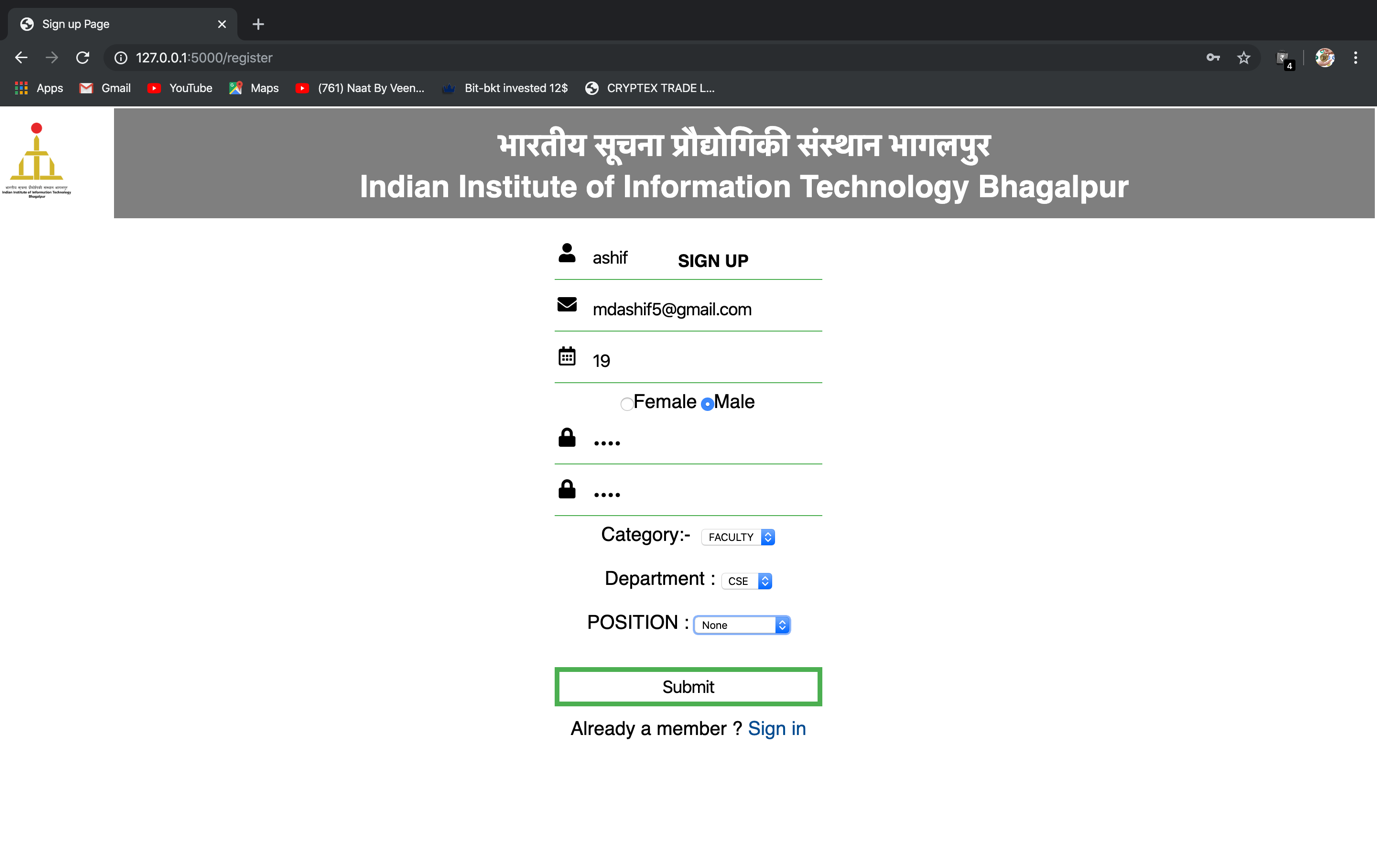
Any user inside the college campus can use the intranet connection and can access this web interface application using the address: 172.16.0.195/elms



### 4.2.2 Signup

The access to this application is restricted to authorized users only. Each user should register using an OTP verification via an email.

The OTP is sent to the provided Email Address after clicking the Submit Button.



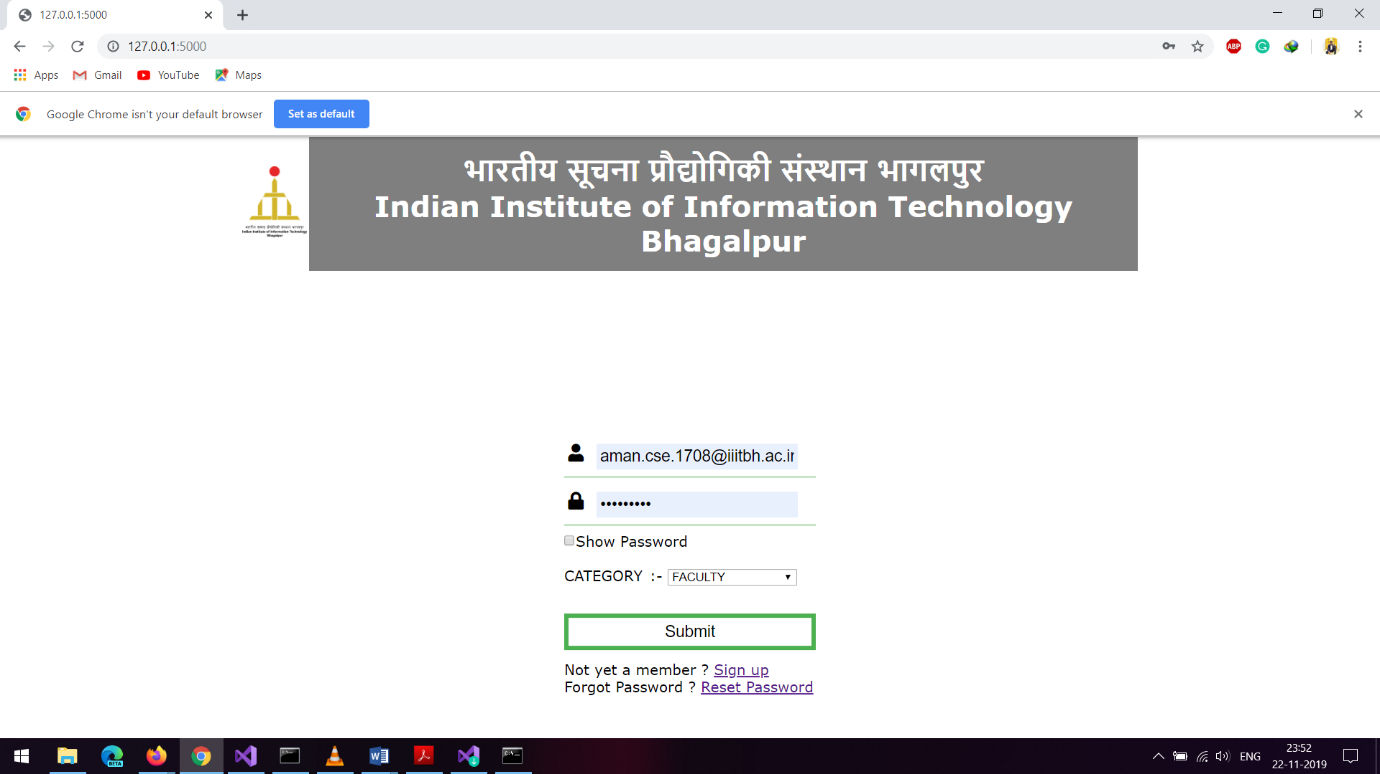
After OTP Verification, the registration will be successful.



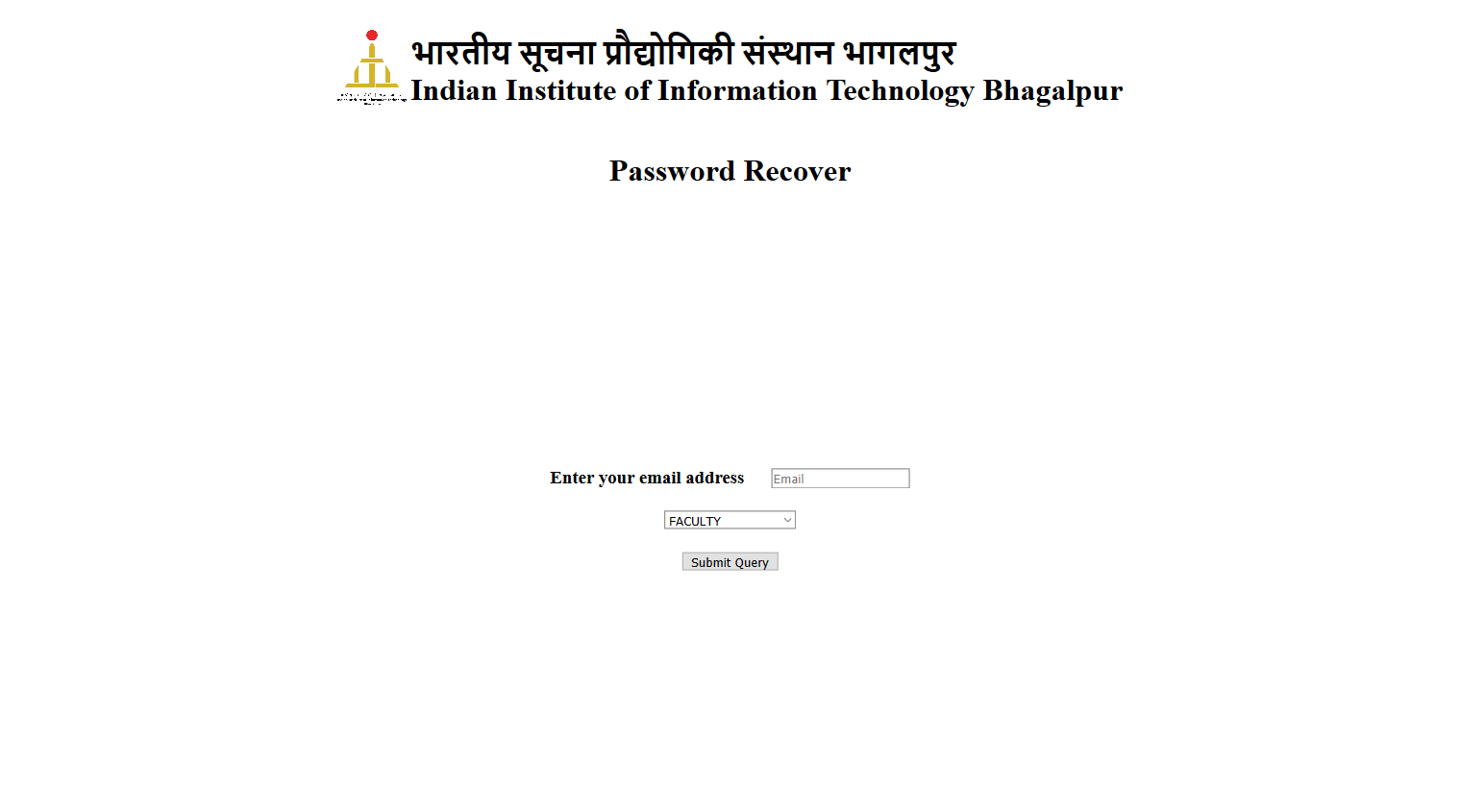
4.2.3 Login

Each user should know his user name and password to access this Web Application and he must be authenticated to be able to use the application.

When you visit this application, Login Page is the first page displayed.



### 4.2.4 Reset Password



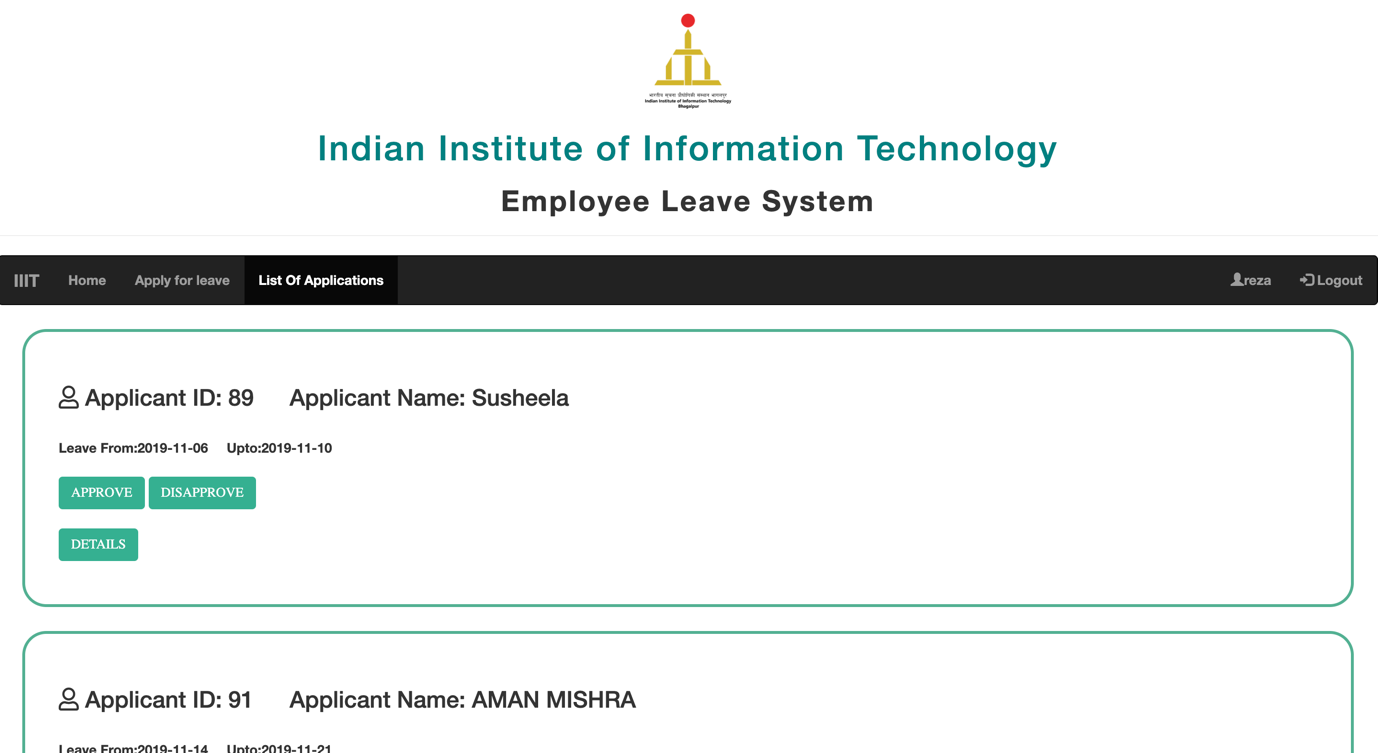
**After login page**

Faculty, HOD, Staff and Others will get their profile page where applicants (faculty, staff) can apply for leave and can see their list of applied application till now and HOD can see list of applicants and approve or disapprove according to applicant left leave.

4.2.5 Profile of HOD

Here HOD can see details of applicants through the provided “DETAILS” tab and can approve or disapprove an applicant’s leave form.

HOD can also apply for leave and see his own history.



### 4.2.6 History of Applicants(Faculty, staff, others)

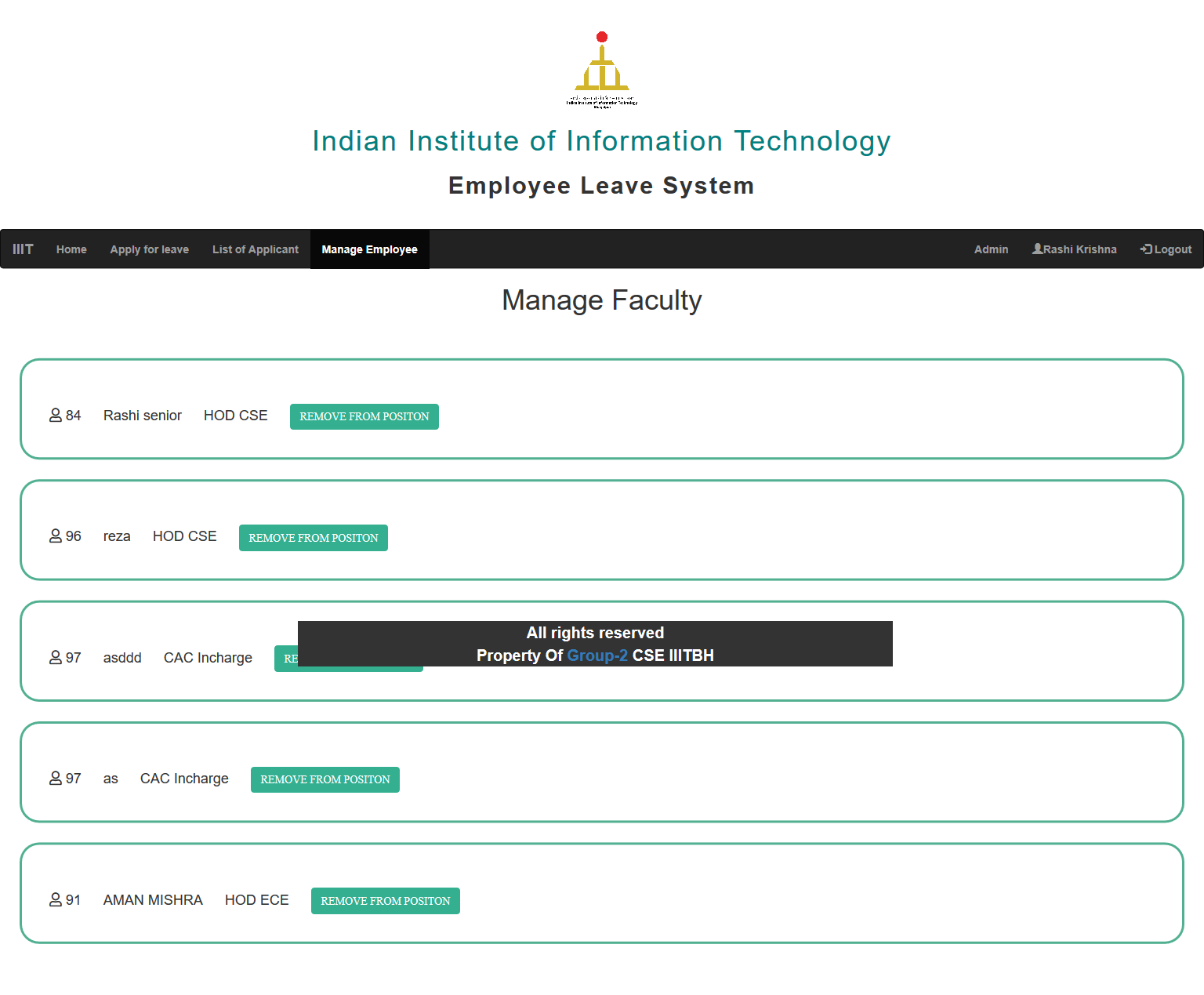
Applicants can see their history of leaves by clicking their respective “name” tab on the top right.



### 4.2.7 Profile of Admin

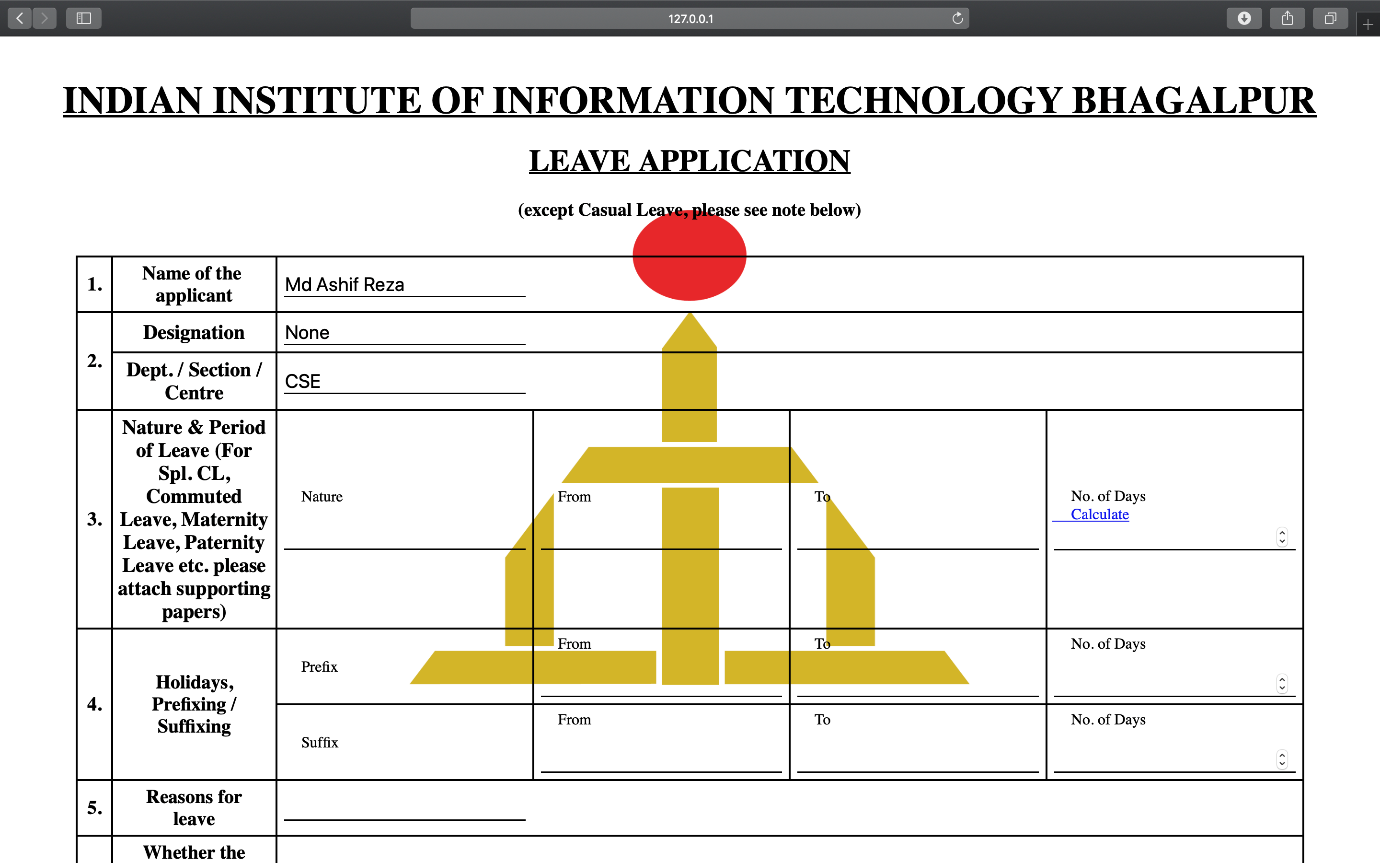
After HOD’s approval, the leave application is forwarded to the admin where he is requested to fill the office side form and forward it to the director for the final approval or disapproval.

Admin also has the privilege to remove Faculties from their respective positions in the “Manage Employee” tab.



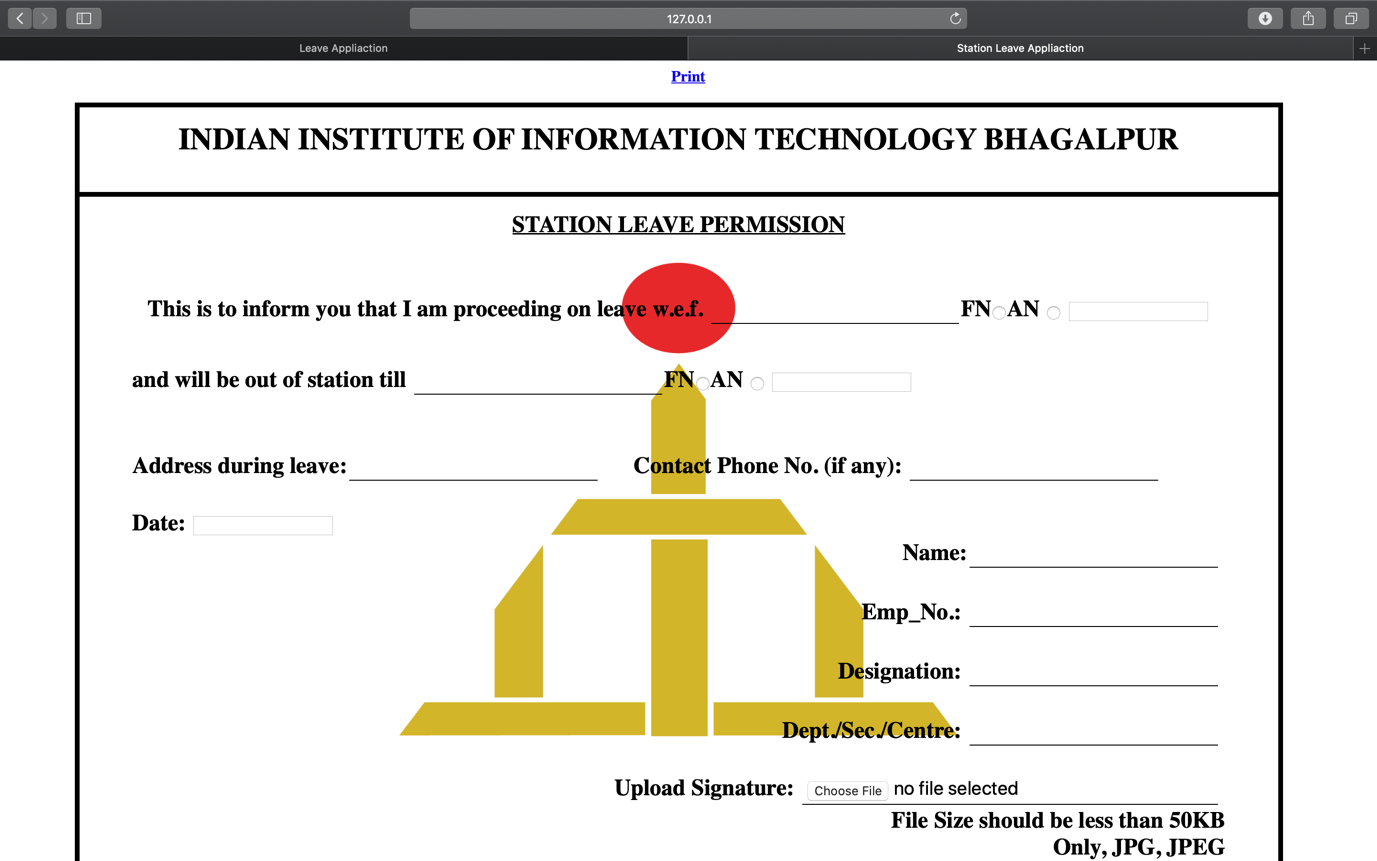
### 4.2.8 Leave Application Form

Applicant can fill the Leave Application Form by clicking on the “Apply for Leave” tab on the top left corner if their profile

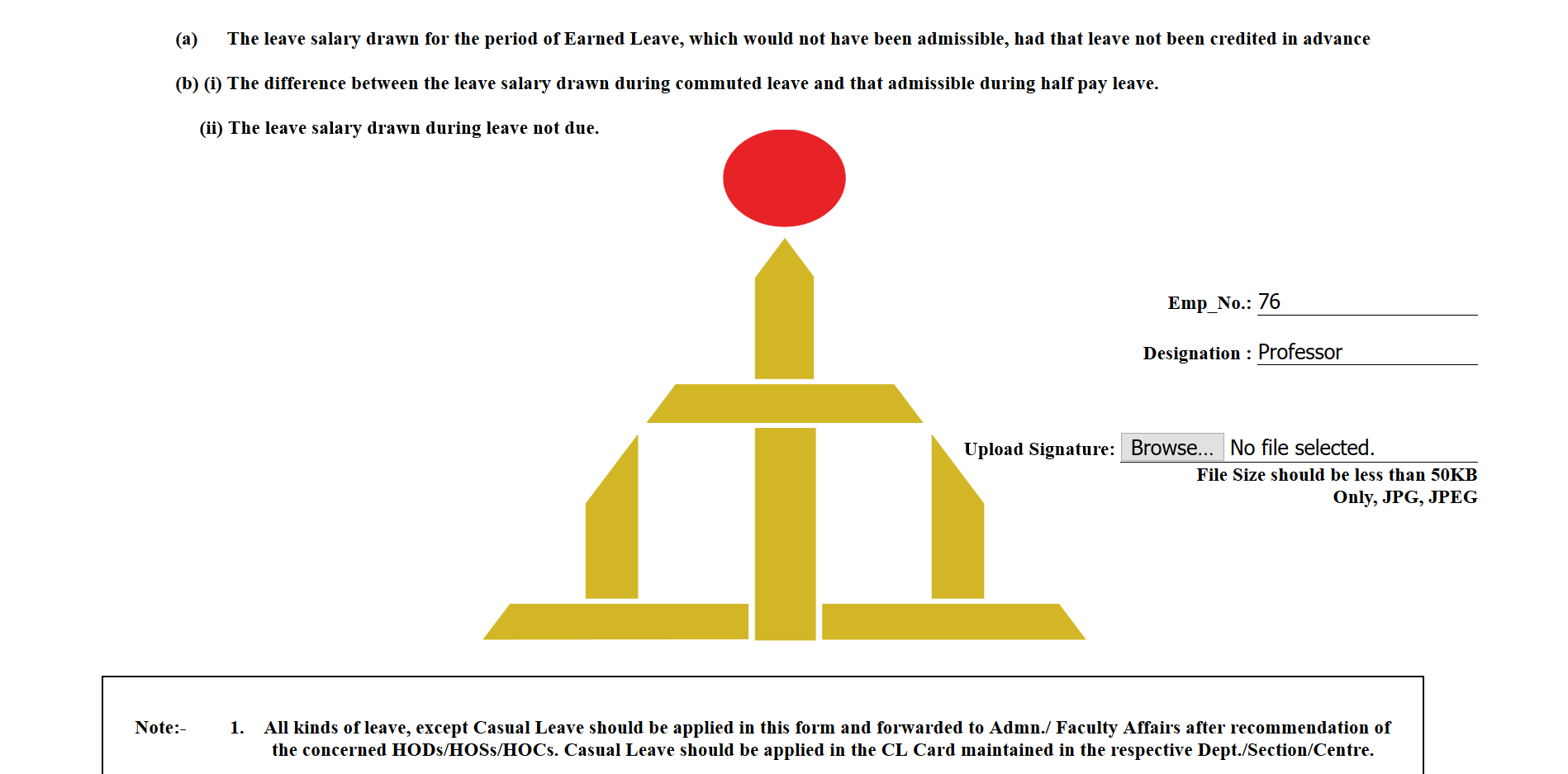


### 4.2.9 Station Leave Form

If the applicant requires a station leave, they need to choose yes in the radio button which would pop the Station Leave Form.

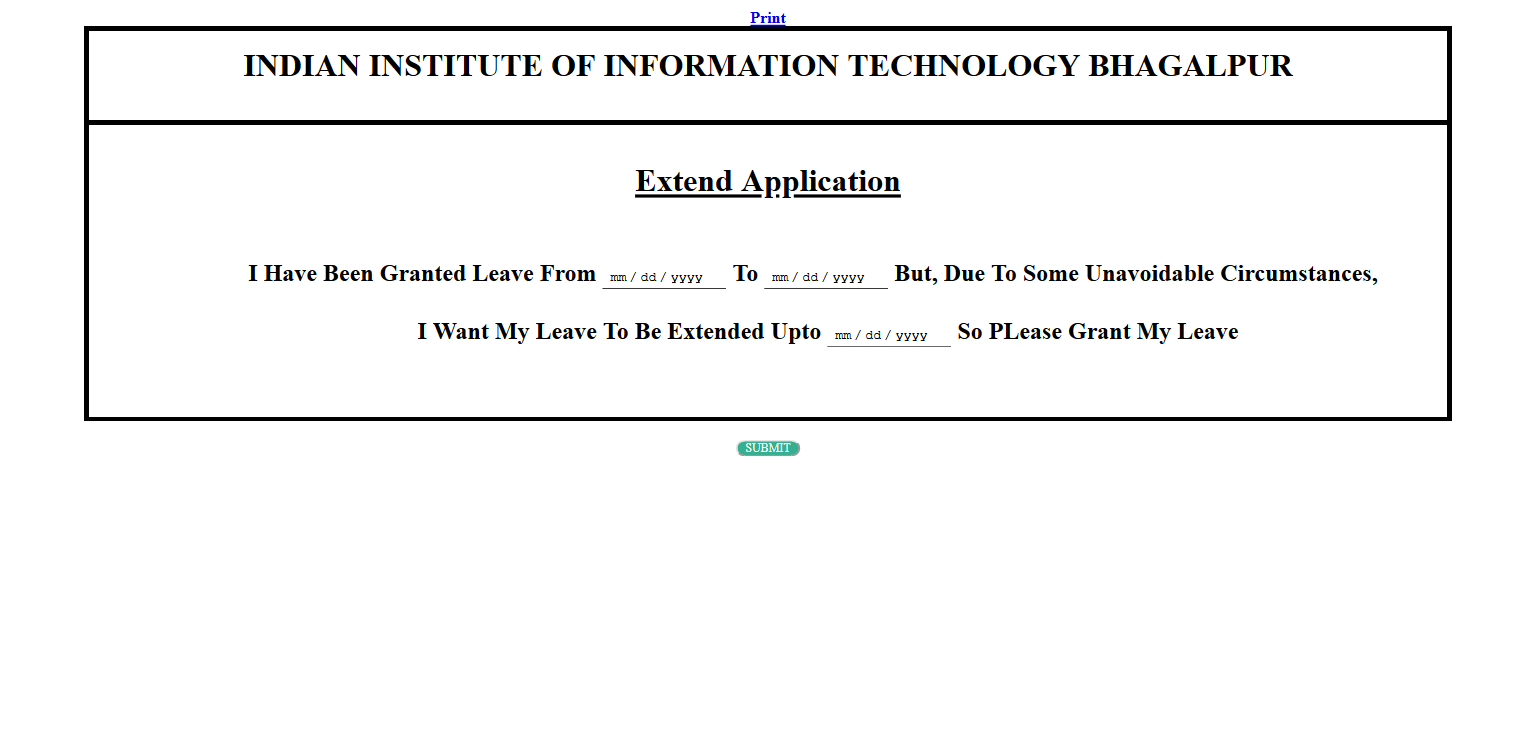
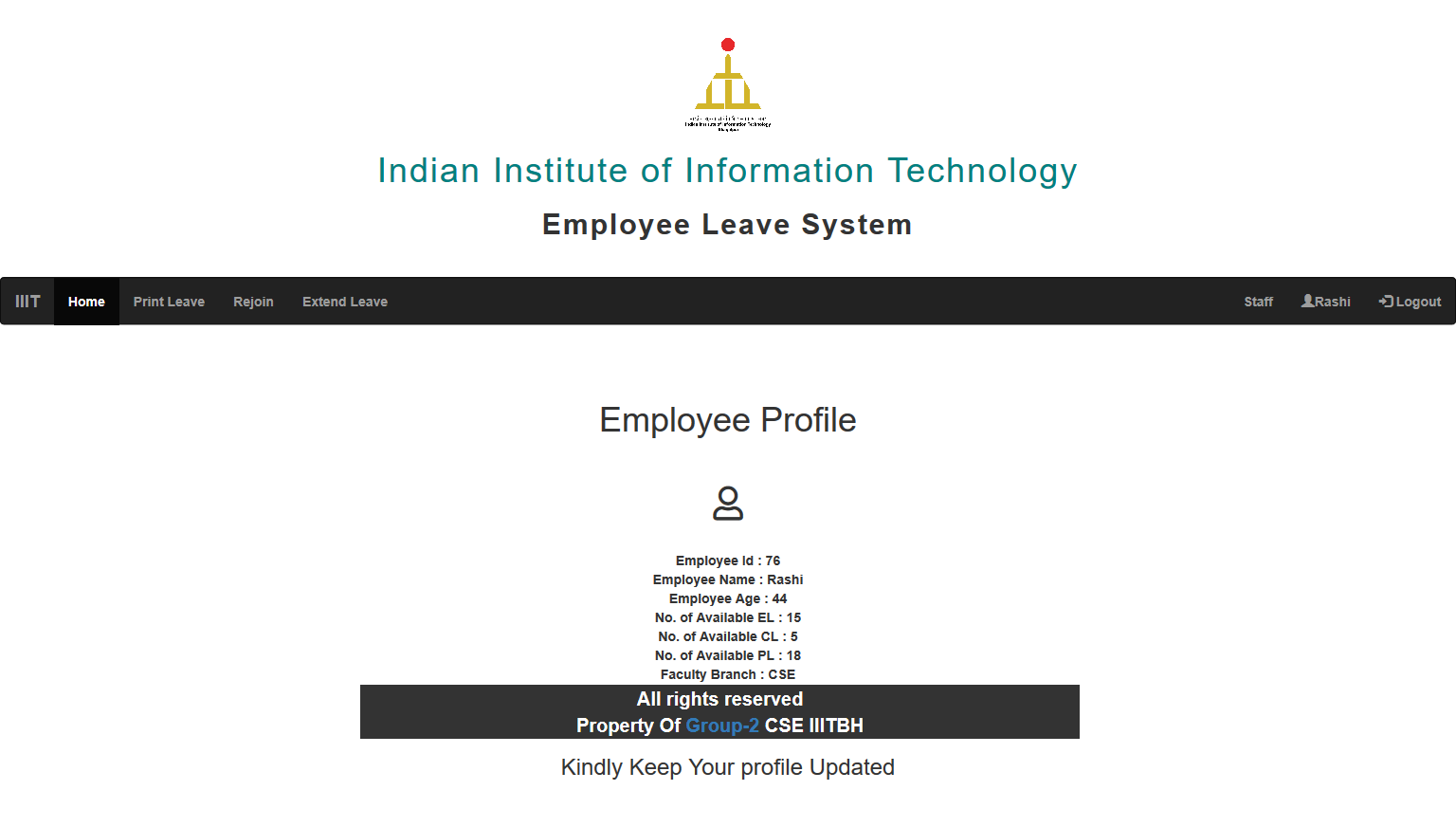


Applicant also need to upload their digit signature while submitting any form.



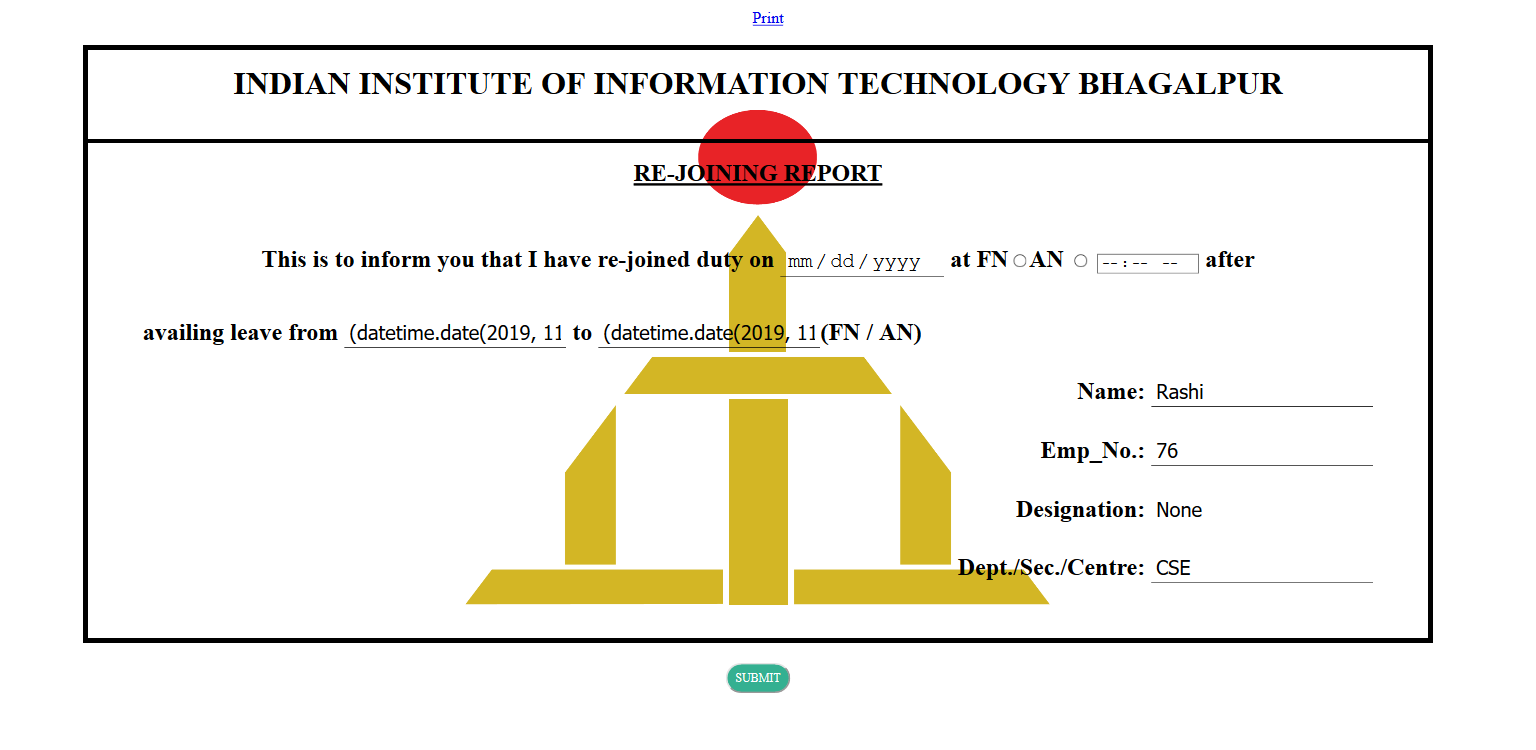
### 4.2.10 Extend Leave Application

While applicants are on leave he/she can also request for extending their leave.



### 4.2.11 Re-joining Form

Once the applicants leave period is over they must fill the re-joining application form to notify the institute that he/she will be joining shortly.

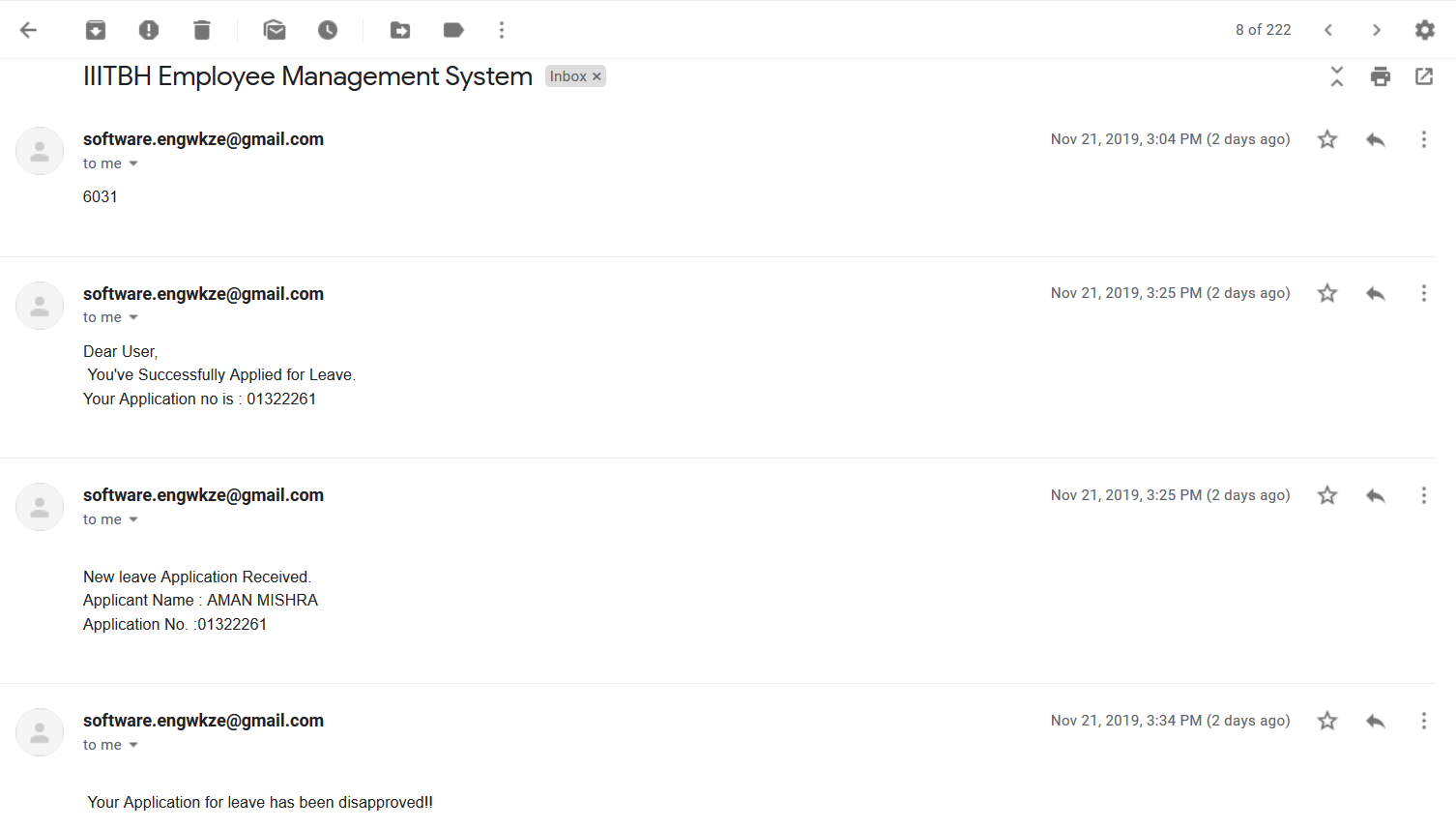


### 4.2.12 The Use of Mail System

After submitting any form, the applicants get notified via an email.

Adding to that, the HOD or the person In-charge also receive an email regarding your request.

And finally, the applicant also gets notified whether his leave application got approved or rejected.



# 5. TESTING

## 5.1 TESTING GOAL

The goal of Employee Leave Management System Testing is to ensure that the system performs as per the functional requirements specified by client. Most cases tested here are done manually per module.

## 5.2 FUNCTIONAL REQUIREMENTS TESTING (BLACK BOX TESTING)

### 5.2.1 Registration for New User

|  |  |  |  |
| --- | --- | --- | --- |
| **Description : This test will ensure registration of new user** | | | |
| **Data Requirements : Connectivity to database** | | | |
| **Steps #** | **Step Description** | **Expected Results** | **Remarks** |
| **1** | Open sign Up page | Sign Up page opens | Pass |
| **2** | Fill Registration Form | All required information  entered properly, no error encountered | Pass |
| **3** | Generate OTP | OTP sent through mail | Pass |
| **4** | Enter OTP and click on register | Successful registration | Pass |
| **5** | Login using credential entered on Sign Up page | Successfully Logged in | Pass |

### 5.2.2 Apply for Leave (Faculty)

|  |  |  |  |
| --- | --- | --- | --- |
| **Description : This test will ensure successful application of leave** | | | |
| **Data Requirements : Connectivity to employee leave DB** | | | |
| **Steps #** | **Step Description** | **Expected Results** | **Remarks** |
| **1** | Sign In using username and password | Home page for faculty opens | Pass |
| **2** | If no leave applied earlier, click on Apply Leave | Faculty Leave form opens | Pass |
| **3** | Upload Signature | Signature applied successfully | Pass |
| **4** | Submit form | PDF generated, email send to the respected faculty, HOD of the faculty and Admin | Pass |

### 5.2.3 Approving / Disapproving Leave

|  |  |  |  |
| --- | --- | --- | --- |
| **Description : This test will check if leave can be successfully approved/disapproved** | | | |
| **Data Requirements : Connectivity to leave DB** | | | |
| **Steps #** | **Step Description** | **Expected Results** | **Remarks** |
| **1** | Sign In using HOD/Admin username and password | Home page for HOD/Admin opens | Pass |
| **2** | Click on List of Applicant | List of application appears | Pass |
| **3** | Click on approve/Disapprove | If disapprove clicked, textbox appears asking for reason for disapproval, email sent to faculty | Pass |

### 5.2.4 Test Case for Login Page

|  |  |  |  |
| --- | --- | --- | --- |
| **Description : This test will ensure secure login of user** | | | |
| **Sr No.** | **Input/Action** | **Expected Results** | **Remarks** |
| **1** | Leave text field empty | Will show error message “Required Field” | Pass |
| **2** | Entered invalid username and password | Will show error message “Invalid username or password” | Pass |
| **3** | Entered valid username | Opens faculty home page | Pass |

### 5.2.5 Test Case for Approve/Disapprove, Manage Faculty Page

|  |  |  |  |
| --- | --- | --- | --- |
| **Description : This test will ensure registration of new user** | | | |
| **Sr No.** | **Input/Action** | **Expected Results** | **Remarks** |
| **1** | Click on View Details | Details of specific applicant opens | Pass |
| **2** | Click on approve | Leave application disappears from pending application | Pass |
| **3** | Click on disapprove | Text box appears for entering reason for disapproval | Pass |
| **4** | Click on Delete faculty | Remove a specific faculty from HOD list | Pass |

### 5.2.6 Test Case for Leave Page

|  |  |  |  |
| --- | --- | --- | --- |
| **Description : This test will ensure meaningful leave form filling by user** | | | |
| **Sr No.** | **Input/Action** | **Expected Results** | **Remarks** |
| **1** | Leave text field empty | Will show error message “Required Field” | Pass |
| **2** | Enter leave from and leave up to field | Automatically calculate duration of leave | Pass |
| **3** | Click on Yes radio button of station leave permission required or not | Station leave form opens | Pass |
| **4** | Upload Signature | Signature Uploaded | Pass |
| **5** | Enter nature of leave (e.g. Casual Leave, Half Pay Leave, Paternity Leave, Maternity Leave etc) | Check if given type of leave applicable for given applicant | Pass |
| **6** | Click on Yes radio button of responsibility undertaken | Responsibility handover form opens | Pass |

## 5.3 Non- Functional Requirements Testing

### 5.3.1 Stability Testing

Stability testing checks to see if the software can continuously function well in or above an acceptable period. This activity of non-functional software testing is oftentimes referred to as load (or endurance) testing.

### 5.3.2 Usability

Usability testing is needed to check if the user interface is easy to use and understand. This test is used to verify if a user that never use the application can search and read result list within a reasonable time.

### 5.3.3 Security Testing

Security testing is essential for software which processes confidential data and to prevent system intrusion by hackers.

# 6. MAINTAINENCE

The Employee Leave Management System is developed using Flask, a lightweight python based WSGI web application framework and hence, it provides an easy Maintainability and Optimization.

Software maintenance in software engineering is the modification of a software product after delivery to correct faults or to improve performance and other attributes. Maintenance covers about 60% of the phase in Software Development Life Cycle.

The ELMS is the soul property of Group CS311-P2 and is developed for the Employees of Indian Institute of Information Technology, Bhagalpur. The group shall provide regular updates for the maintenance of the Employee Leave Management System.

The Maintenance of ELMS shall include:

* Error Correction
* Bug Fixes
* Enhancement of Capabilities
* Deletion of Obsolete Capabilities
* Optimization

The ELMS has been developed for simplifying the process of taking leave from the Institute and thus it is not affordable to have bugs in the Application but nonetheless some sneaky bugs may find its way to the users. The users are requested to report the bugs as soon as they encounter it so that it could be fixed in the next update.

# 7. CONCLUSION

The proposed Employee Leave Management System will make the whole leave management process efficient. Users will be able to know their leave status, ask for leave extension and re-join, all via the use of Intranet.

This will also help the admins and HODs as they will be able to approve or disapprove the leave application forms on their PCs or laptops with college intranet connection.

Ultimately, this project aims at reducing all the paperwork which must be done while applying for a leave. Moreover, this project will simplify the tedious process of applying for a leave and make it swift and reliable.