# Acknowledgement

Firstly, I want to express my sincere gratitude to my tutor Achyut Timsina Sir and our foremost Pramod Poudel Sir who gave me this golden opportunity to do this challenging yet educational project on the topic Patient Management System, which helped me to improve my researching skills and my technical skills. I could learn many new things during my research and was able to apply it in this project.

Secondly, I want to thank my guardians and friends who were always there for me supporting on every step I take during finalizing this project.

# Abstract

Patient Management System is a web application. This application is an information system that is used by hospitals, clinics for handling and storing the data of the organization simply and smoothly. It gives different functions and features. The fundamental advantage is to store record that are acquired in the hospital. It gives distinctive facilities, for example, it helps storing record and enrolling patient and inpatient. Inpatient are those patients who are conceded inside the hospital. In addition, it gives the facility of managing appointments in a hospital. The detail documentation for our application is below.

# Chapter 1

## 1.1 Introduction:

In this project, I am going to develop Patient Management System. It is an information system which is recently getting popularity due to its massive advantages. It has been playing important role in our day to day life. Due to its automation facility, more peoples are attracted to it. Information system is helping many organizations, companies, etc. for achieving their goals easier and faster.

Patient Management System is developed by hospitals, clinics, etc. for managing their huge data of people who are involved in the organization. This system helps the organization store their data easily and quicker without any data redundancy.

This system provides lots of methods and characteristics to its end-users. It helps users to store and register data of patients, doctors, staffs and inpatients. It also makes easier for patients to take appointment for checkup with available doctors.

## 1.2 Main Features:

Followings are the main features of Patient management system:

* Create, Read, update and delete the data for patients that come in hospital.
* Create, read, update and delete the data for doctors in hospital.
* Create, read, update and delete the data for staff in hospital.
* Create, read, update and delete the data for inpatients in hospital.
* Provide search features for many records.
* Manage and Create Appointment records.

The main tools and technologies that I will be using to create my project are as follows:

* Docker with Laradock package as environment for Laravel project.
* Laravel 5.5 with Laravel collective HTML and Yajra (Laravel datatables oracle) as additional packages.
* Heidi SQL for managing MySQL database.
* PHP Storm for integrated development environment.

## 1.3 Aims:

Followings are the aims for developing the Patient Management System:

* To develop an information system that stores records in computerized way.
* To reduce the human effort in hospital.
* To reduce paper works.
* To give the end-users with the facility for managing and searching records in seconds.
* To avoid data redundancy and loss of data.
* To provide an app which allow for automation.

## 1.4 Objectives:

Following are the main objectives for developing the Patient Management System:

* To gather project related data and information.
* To gather the end-user’s requirements for the system what makes them easy.
* To design the entire system and code.
* To provide user friendly system.
* To provide Documentation with all kinds of helpful diagrams for the system.
* To make long-term use of the system.

## 1.5 Development Method:

For the development of this patient management system I have used Modified Waterfall methodology as this methodology is similar as basic waterfall model but its phases are overlapped. This difference allows the project to be more flexible. Hence, this methodology helped me to fill the missing requirements by moving to earlier phase.

By using this methodology, I could split into subproject at a required phase i.e. database design and backend design.

Advantages of using Modified Waterfall Methodology in this project are as follows:

* Made it easier to manage every phase.
* In case of problems, overlapping phases allowed the project to be more flexible.
* I could complete this task efficiently within the deadline.
* I could complete phases one after another and was also able to move back to the previous phase.

# Chapter 2: Analysis Specification

## 2.1 Introduction:

The main analysis phase of our project starts here. The main moto of this phase is to provide document which helps in description of design and is to find out the requirements which is generally associated with our project. With the help of use case diagram, functional and non-functional requirement, architecture, prioritization, etc. we are evaluating the necessity of the project. Firstly, all the necessities are taken from different foundations then it is studied and documented which helped us for the development of the project. Finally, we could know the real functional and non-functional requirement of the system and much more.

## 2.2 Use Case:

Use Case is a vital aspect for finding out the necessity of the system. Use Case diagrams helps us to detect and simplify the suitable methods that needs to be carried out by the system. This diagram also helps us to know about the way that end user relates with the system and what methods they will be using in the system. With the help of this diagram we got the blue print of the system from which we could find the complete view of the system. Communication among unlike persons such as developers, clients etc. is also been possible due to use case diagram. With the help of use cases, high level view of the system is also possible.

Use Case diagram contains unlike features for example use case which displays different methods, actor which will complete the methods and association among those methods.

Following is the use case diagram for our system:

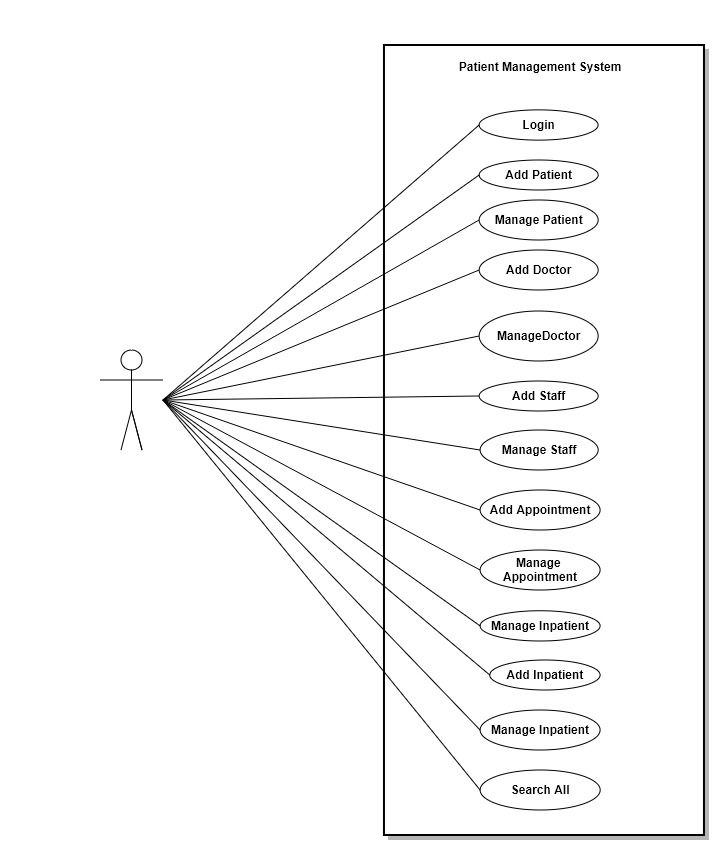


Fig: Use Case Diagram for the system.

1. Explanation of login in use case:

* Firstly, Administrator of the system is redirected to the login page of the system.
* Then, Admin inputs their email and password.
* If the system detects the details to be correct then the admin can go inside the system else the admin cannot go inside the system and should try again.
* Finally, after the login is successful the admin enters to the system.

1. Explanation of adding patient in use case:

* When admin enters to the system, he or she can easily add a new record of patient.
* All the information of the new patient is entered in the system.
* Lastly, all the information is kept in the system.

1. Explanation of managing Patient in use case:

* When the information of new patient is stored in the system we can also manage them.
* We can edit and update certain information of the patient and we can also easily destroy that record.

1. Explanation of adding doctor in use case:

* When admin enters to the system, he or she can easily add a new record of doctor.
* All the information of the new doctor is entered in the system.
* Lastly, all the information is kept in the system.

1. Explanation of managing doctor in use case:

* When the information of new doctor is stored in the system we can also manage them.
* We can edit and update certain information of the doctor and we can also easily destroy that record.

1. Explanation of adding staff in use case:

* When admin enters to the system, he or she can easily add a new record of staff.
* All the information of the new staff is entered in the system.
* Lastly, all the information is kept in the system.

1. Explanation of managing staff in use case:

* When the information of new staff is stored in the system we can also manage them.
* We can edit and update certain information of the staff and we can also easily destroy that record.

1. Explanation of adding inpatient in use case:

* When admin enters to the system, he or she can easily add a new record of inpatient.
* All the information of the new inpatient is entered in the system.
* Lastly, all the information is kept in the system.

1. Explanation of managing inpatient in use case:

* When the information of new inpatient is stored in the system we can also manage them.
* We can edit and update certain information of the inpatient and we can also easily destroy that record.

1. Explanation of adding appointment in use case:

* When admin enters to the system, he or she can easily add a new record of appointment.
* All the information of the new appointment is entered in the system.
* Lastly, all the information is kept in the system.

1. Explanation of managing appointment in use case:

* When the information of new appointment is stored in the system we can also manage them.
* We can edit and update certain information of the appointment and we can also easily destroy that record.

1. Explanation of Searching all records in use case:

* All information that have been saved in the system can be searched manually.
* User should enter proper search key words and the result will be displayed.

## 2.3 Requirements:

This is the analysis segment where we should find and organize unlike necessity of the system which are functional and non-functional. With the help of user analysis and problem report, we should differentiate whether they are non-functional or functional. Functional requirements contain the basic and vital features of the system i.e. login system. Non-functional requirement contains the features which are important for the system but lack of their presence the system won’t get much damaged i.e. performance etc.

Followings are the functional and non-functional requirement for our system:

### 2.3.1 Functional Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Requirement Description** | **Input** | **Output** | **Dependencies** |
| Y1  functional | Administrator should login | Correct E-mail and Password | If the data provided by admin matches the data then admin is redirected to the system. If not, then admin should try again. | Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Y10, Y11, Y12 |
| Y2  functional | Admin should be able to add patient. | correct information for new patient | if the data provided are correct then is stored in database | Y3 |
| Y3  functional | Admin should be able to manage patient. | new information for patient | if the data provided are correct then is updated in database | Y2 |
| Y4  functional | Admin should be able to add doctor. | correct information for new doctor | if the data provided are correct then is stored in database | Y5 |
| Y5  functional | Admin should be able to manage doctor. | new information for doctor | if the data provided are correct then is updated in database | Y4 |
| Y6  functional | Admin should be able to add staff. | correct information for new staff | if the data provided are correct then is stored in database | Y7 |
| Y7  functional | Admin should be able to manage staff. | new information for staff | if the data provided are correct then is updated in database | Y6 |
| Y8  functional | Admin should be able to add inpatient. | correct information for new inpatient | if the data provided are correct then is stored in database | Y9 |
| Y9  functional | Admin should be able to manage inpatient. | new information for inpatient | if the data provided are correct then is updated in database | Y8 |
| Y10  functional | Admin should be able to add appointment. | correct information for new appointment | if the data provided are correct then is stored in database | Y11 |
| Y11  functional | Admin should be able to manage appointment. | new information for appointment | if the data provided are correct then is updated in database | Y10 |
| Y12  functional | Admin should be able to search all records. | correct search keywords | if the data provided matches with the data within the system then the result is displayed by the system immediately | Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Y10, Y11 |

### 2.3.2 Non-Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirement** | **Description** | **Rational** |
| Y1  non-functional | Security | System must be secure so no one should enter the system without correct email and password. | So, security |
| Y2  non-functional | Available | Should provide 24-hour service | system should be available |
| Y3  non-functional | Reliable | System should give precise output of the inputs provided by the users. | for increasing the users trust |

|  |  |  |  |
| --- | --- | --- | --- |
| Y4  non-functional | Efficient | System should give its best also at its worst condition in a limited time and effort. | for increasing the users efficient. |
| Y5  non-functional | Fast browsing experience | The system should work faster. Slow internet bandwidth should not degrade the overall performance of the system | to satisfy users who have low bandwidth internet |

## 2.4 Prioritization

Now after the identification of the functional and non-functional requirement is done, we must prioritize them correctly. Prioritization helps us to confirm that at certain time if we have problem while finishing the project we can develop the methods that are very necessary for the system.

For the system prioritization, I have used Moscow technics which supports in categorizing the must have functions, could have functions, should have functions and won’t have functions.

Following is the requirement prioritization for our system:

|  |  |  |
| --- | --- | --- |
| **S.N.** | **Functions** | **Prioritization** |
| 1. | Login | Must Have |
| 2. | Add Patient | Must Have |
| 3. | Manage Patient | Should Have |
| 4. | Add Doctor | Must Have |
| 5. | Manage Doctor | Should Have |
| 6. | Add Staff | Must Have |
| 7. | Manage Staff | Should Have |
| 8. | Add Inpatient | Must Have |
| 9. | Manage Inpatient | Should Have |
| 10. | Add Appointment | Must Have |
| 11. | Manage Appointment | Should Have |
| 12. | Search All | Could Have |
| 13. | Security | Must Have |
| 14. | Available | Could Have |
| 15. | Reliable | Should Have |
| 16. | Efficient | Should Have |
| 17. | Fast browsing experience | Could Have |

## 2.5 Architecture

After the completion of the prioritization, we should be developing the structure of the system. With the help of architecture of the system we came to know the overall view of the system. It also helps us to define several mechanisms of the system and how it would be intended and estimated.

### 2.5.1 System Architecture

It is used to outline the standard key which helps in getting the system’s necessity. It helps in advancing the features of the system and helps to deal with core application. System Architecture is very important because like any other composite structure, system should be developed on solid foundation.

Followings are the advantages of using system architecture:

* Helps to know that all the functionality is completed as the client’s requirement.
* Isolating the systems in fragments so that composite methods and designs are simply understood and developed.
* Helps in finding out the bugs and helps in decreasing the errors.

#### 2.5.1.1 Framework

Among the huge section of frameworks, we have chosen Laravel version 5.5 PHP framework for developing our project. Laravel is leading free and open-source PHP web framework with lots of features and functions. It makes the developer more easier and faster to complete the task with more secure, reliable and effective way.

Followings are features of Laravel Framework:

* Full login system with a simple command line.
* Additional packages for the system could be added easily.
* Has Best routing system with middleware.
* Sensitive data are protected in a single file called “.env” which is ignored even while pushing in git server.

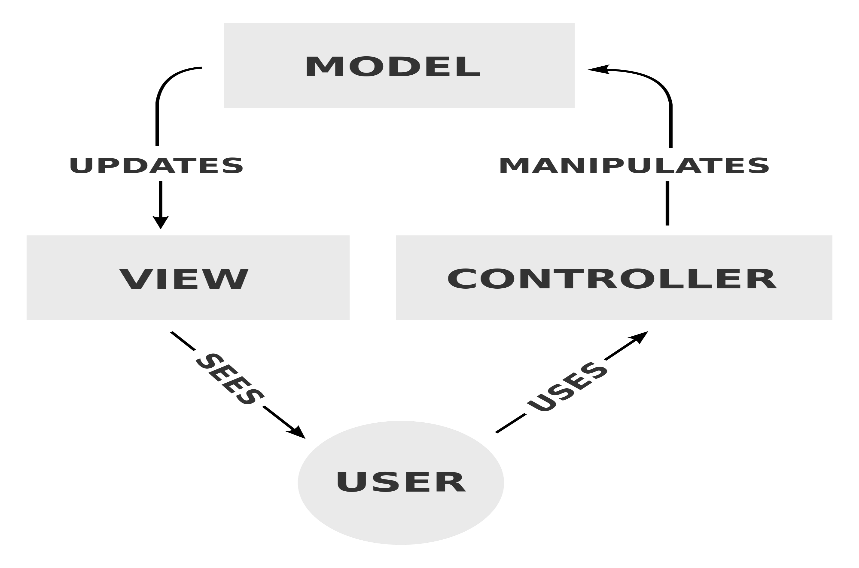
Followings are the benefits of using Laravel framework:

* Simple and easier to use.
* Makes the coding more fun and effortless.
* Security is maintained to its best to protect form different kinds of attacks.
* Database tables creation and migrations are easier in Laravel.

#### 2.5.1.2 MVC Pattern

To build our system architecture we will be using MVC pattern. MVC consists of Model, View and Controller.

* Model: This component deals with all the data-related logic that the end-user works with.
* View: This component deals with the UI logic of the system.
* Controller: This component acts as a middle man which communicates between model and view by processing all the logics, requests and data.



#### 2.5.1.3 Initial Class Diagram

It is the procedure of searching out the basic view of the system by the expansion of the class diagram using natural language analysis. It is the process where a text is studied and then adjective, noun and verb are recognized which are class, function and attribute respectively.

Following is the initial class diagram for our system:

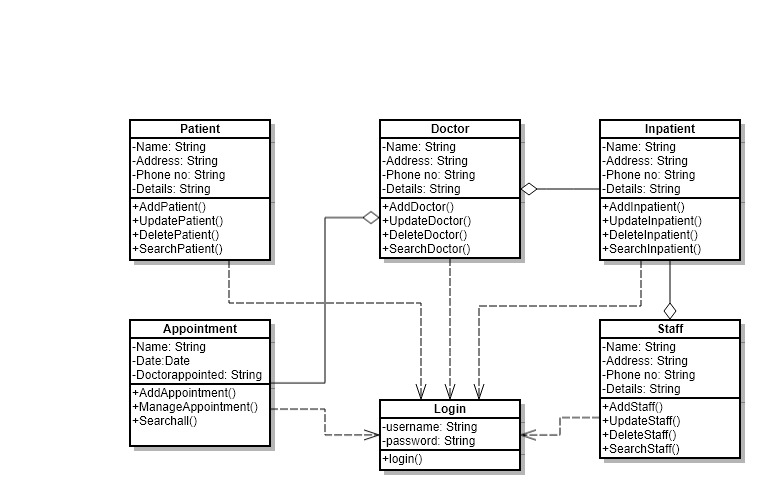


Fig: Initial Class Diagram

## 2.6 Conclusion

Hence, our analysis phase has been completed successfully. In this part, we have drawn use case diagrams with functional and non-functional requirements. We have also covered system architecture which altogether helped to analyze our system. Lastly, this will help us in design specification phase and Implementation phase too.

# Chapter 3: Design Specification

With the help of Analysis Specification, designing outline of the system is done in Design Specification. This phase shows the actual requirement and features of the system. In this phase, several design documents are made which helps us for viewing the actual structure and actions of an application. It is also known as vital facet for developing an application. We are demonstrating several models of the designing phase in this design specification i.e. behavior model and structural model. One of the good example of structural model is “Class Diagram”, where many classes, attributes, relations and their functions are shown and defined. Likewise, Sequence Diagram and Activity Diagram are the behavior model where interactions between different methods takes place.

We are using UML (Unified Modeling Language) for creating design specification. For graphical illustration of several design specification, Unified modeling language is used. The main features of using this tool is that it is less expensive and designer friendly. The Tool that we have used for creating design specifications are Gliffy Diagrams and Visual Paradigm. I have chosen these software because of their numerous advantages, features and many more.

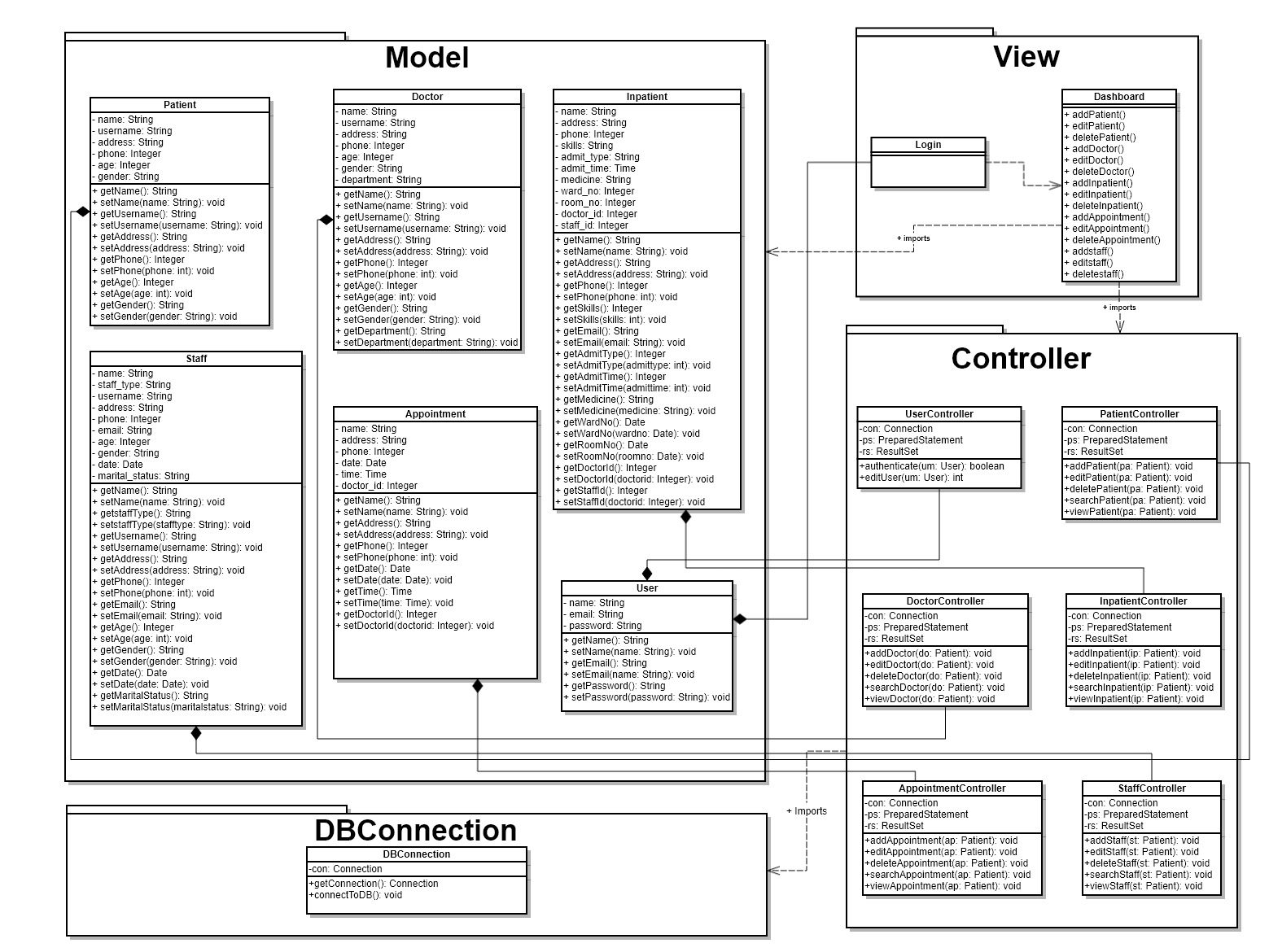
I have shown the diagrams for the class diagram and entity relationship diagram in the structural model as 3.1.1 and 3.1.2 respectively. And for the sequence diagram and activity diagram I have shown the diagram in 3.2.1 and 3.2.2 below respectively.

## 3.1 Structural Model

Structural model is the model which supports us to discover the complete structure of an application. This model could be used for demonstrating various forms of necessities which includes both functional and non-functional requirements. It helps in developing clear architecture of an application.

#### 3.1.1 Class Diagram

Class Diagram plays vital role in the development of an object-oriented programming language for development of an application. Mainly class diagram helps us for demonstrating design modeling and conceptual modeling more professionally. With the help of class diagram, we could demonstrate various packages, classes with their attributes and relationships between them. In the Analysis Specification phase, we have completed an initial class diagram which demonstrates slight representation of the system. Now with its help we are developing a full class diagram for this application.



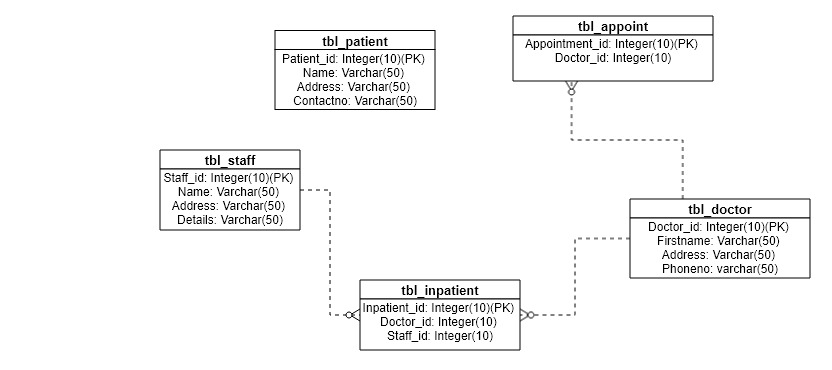
Follows are the description for the class diagram:

* Here we have four different packages.
* Package Model has six different classes i.e. Patient, Doctor, Inpatient, Staff, Appointment and User with many attributes and methods in them.
* Package View has two different classes i.e. login and dashboard.
* Package DB Connection has one class i.e. DBConnection.
* Package Controller has six different classes i.e. UserController, PatientController, DoctorController, InpatientController, AppointmentController and StaffController.
* They all have related association with each other.

#### 3.1.2 Entity Relationship Diagram

The procedure of defining entity and their association from the database is known as entity relationship model. Entity relationship model helps us to express the entire structure of an application and helps us to express the entity which are set in the DB.

Following is the entity relationship for our application which we are developing.



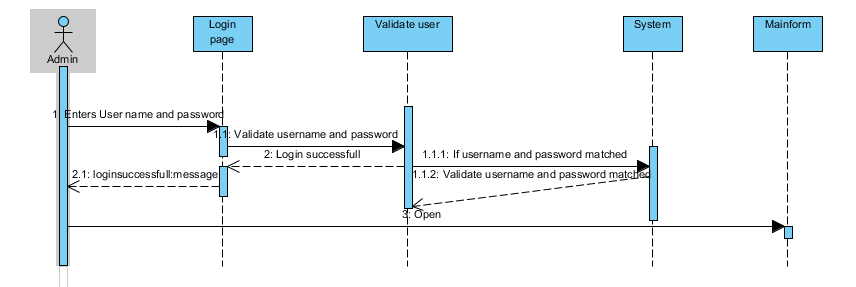
## 3.2 Behavior Model

The procedure of defining the interactions and progressions among two different classes is known as behavior model. It helps in defining how actual communications takes place in an application. The major aspect of the behavior model is to display the management of data in an application. Sequence diagram and Activity diagram are two different behavior model which is described below.

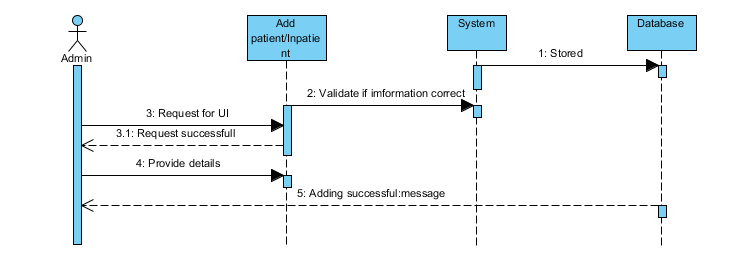
#### 3.2.1 Sequence Diagram

Sequence diagrams deals with the all the logics that are passed out inside an application. It is the drawing which is used for the design purpose. It helps us to define the behavior model of an application. The reason among two various methods and the order they run is the main feature of sequence diagram. As events are taken in the diagram it is also known as Event Traced Diagram.

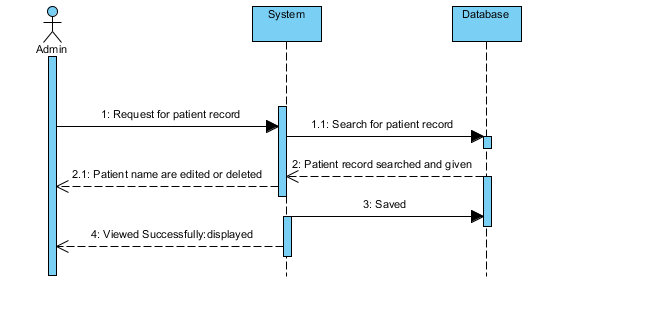
Following are the various sequence diagram for our application.



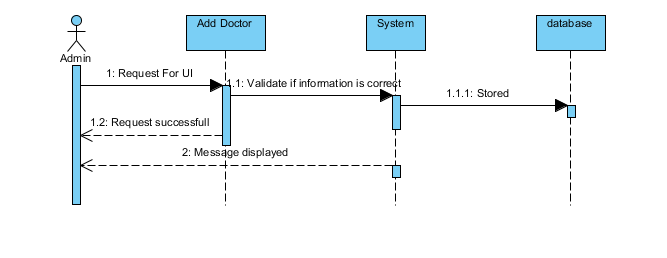
* The above diagram is the sequence diagram for our login system.
* Actor named Admin delivers the login information to the system’s login page then our system authenticates that information whether it is right or wrong.
* If the provided information is right then only Admin is able to enter in an application.



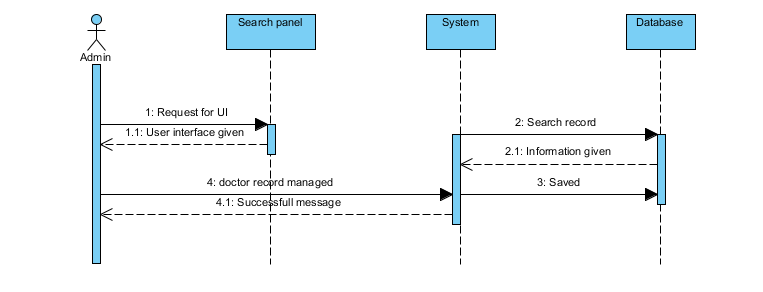
* The above sequence diagram shows the creating part of Inpatient and patient.
* Actor Admin adds new patient or inpatient by filling their details in the form of system. Then clicks submit.
* If the provided system is valid then only system accepts that information and stores it in database and throws successfully added message to admin.



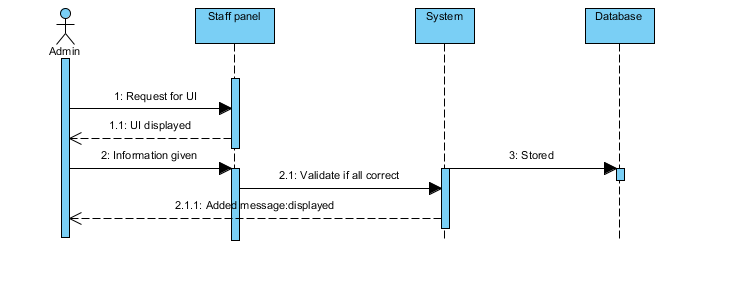
* The above diagram represents the editing part of patients and inpatients.
* If the system has an information stored of a patient or inpatient then admin can manage that information.
* So, the admin search for the information of patient or inpatient and edits or deletes the information.
* After that the information is pushed in the database and successfully edited or deleted message id displayed in the screen.



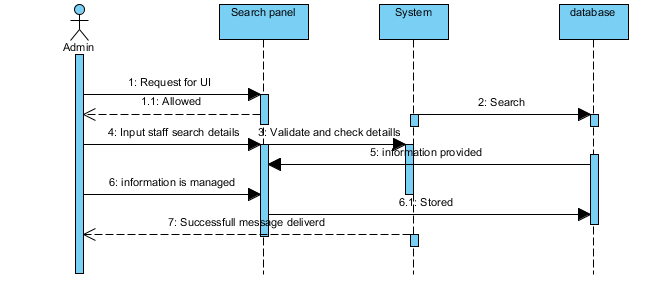
* The above sequence diagram shows the creating part of Doctor.
* Actor Admin adds new doctor by filling their details in the form of system. Then clicks submit.
* If the provided system is valid then only system accepts that information and stores it in database and throws successfully added message to admin.



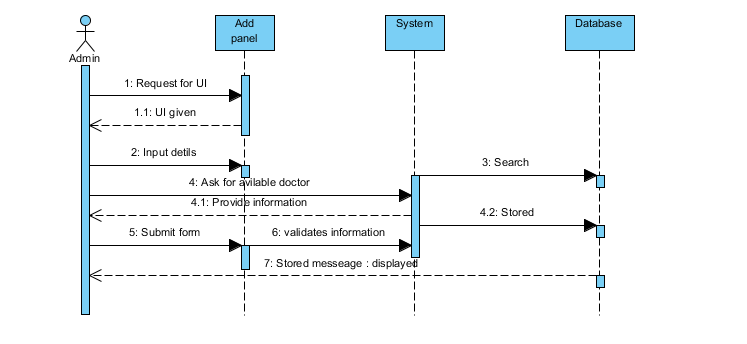
* The above diagram represents the editing part of doctor.
* If the system has an information stored of a doctor then admin can manage that information.
* So, the admin search for the information of doctor and edits or deletes the information.
* After that the information is pushed in the database and successfully edited or deleted message id displayed in the screen.



* The above sequence diagram shows the creating part of Staff.
* Actor Admin adds new staff by filling their details in the form of system. Then clicks submit.
* If the provided system is valid then only system accepts that information and stores it in database and throws successfully added message to admin.



* The above diagram represents the editing part of staff.
* If the system has an information stored of a staff then admin can manage that information.
* So, the admin search for the information of staff and edits or deletes the information.
* After that the information is pushed in the database and successfully edited or deleted message id displayed in the screen.

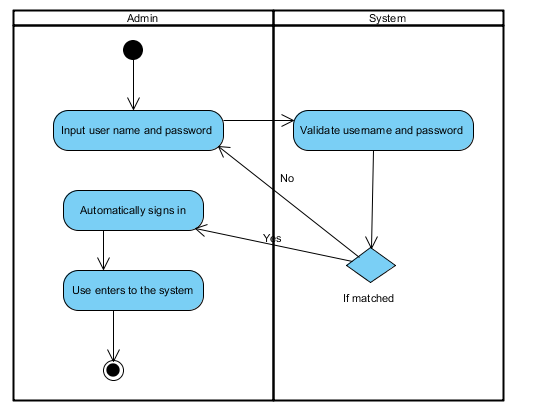


* The above diagram represents the editing part of appointment.
* If the system has an information stored of an appointment then admin can manage that information.
* So, the admin search for the information of appointment and edits or deletes the information.
* After that the information is pushed in the database and successfully edited or deleted message id displayed in the screen.

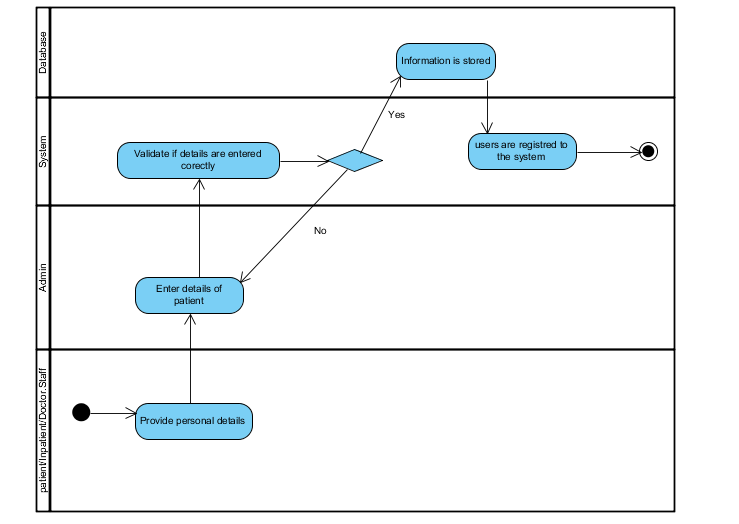
#### 3.2.2 Activity Diagram

Activity diagram helps us to find out the overall workflow of the system. Activity diagram is more structural and innovative then the flow chart. Generally, Activity diagrams are created on the basic of Use Case Diagram. Activity diagram helps in describe simple use cases to composite element structure.

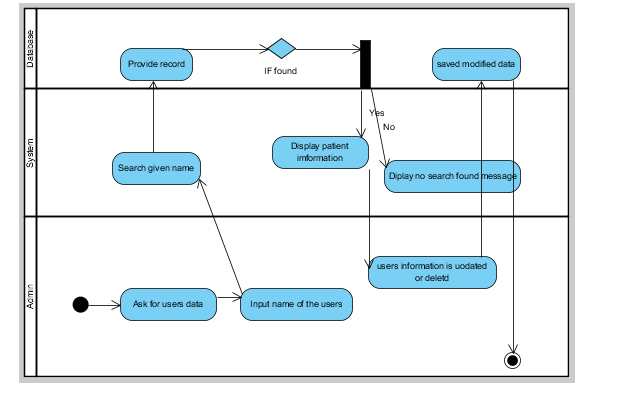
Following is the various activity diagram for the application.



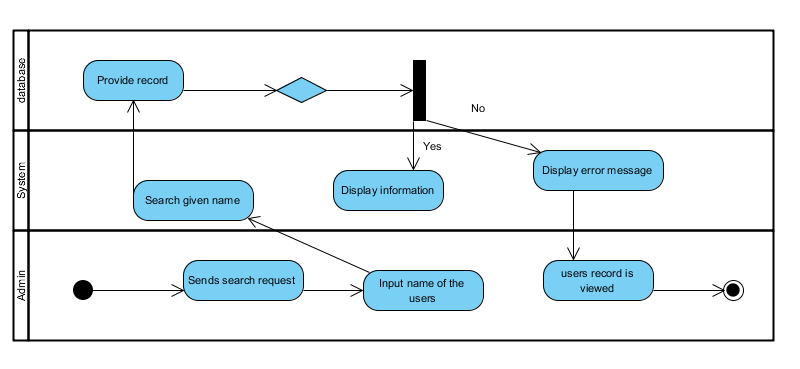
* The above diagram is the activity diagram for our login system.
* Here, admin enters their login details in the system. Then the system validates the information provided by admin.
* Admin enters into the system.



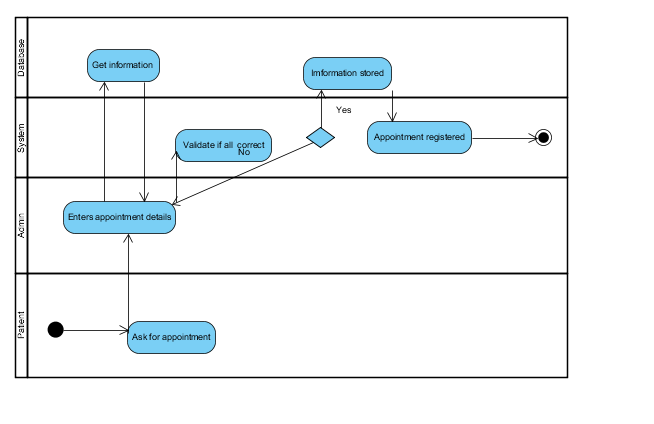
* In the above diagram, new information is added in the system.
* Information of patient, inpatient, doctor and staff are added.
* Firstly, user provides their information to admin, then admin inserts their valid information to the system.
* If the provided information is right then only that information is stored.



* The above diagram shows the management of the information.
* Firstly, admin collects the information from user and enters in the system.
* Then, system finds the record of the searched information.
* Finally, Admin can edit or delete the information and stores into the system.



* The above diagram is to find users.
* The admin delivers searched report to the user.
* If the system has the searched information then the application displays information in the screen.



* The above diagram carries activity diagram for booking appointment.
* Firstly, a patient request for an appointment to the admin. Then, admin provides the required information in the system.
* The information is stored and then the doctor gets informed.

## 3.3 Conclusion

Finally, we have accomplished our design specification correctly. Here, we have defined and wrote about use cases. Then we have defined the class diagram and the static model of our application. After that we have developed an ER-Diagram. We have also successfully described about the behavior model with the help of both activity and sequence diagram.

# Chapter 4: Implementation

The procedure of evolving a solution from analysis specification and design specification to a concluding outcome is known as implementation. As an application is formed as a final product, this phase is also known as deployment phase. For developing this application, we have used many kinds of software and methods. This is the phase where we will be showing a summary concerning the software development language we are using and the software we are using.

## 4.1 Programming Language

Among various programing languages we have chosen PHP as our programming language. And for database we have used MySQL.

### 4.1.1 PHP

* Helps us in developing dynamic websites because it is mostly used server-side scripting language.
* This programming language is widely used and is very faster and smoother compared to other.
* This programming language follows OOP i.e. Object-Oriented Programming.
* It delivers different frameworks for building dynamic websites. We are using Laravel as our PHP framework.
* PHP is also more flexible programming language as it is platform independent.

### 4.1.2 MySQL

* MySQL is widely used DBMS i.e. Database Management System that helps us in executing database of SQL.
* It can be installed easily both in local and production environment.
* It is more useful for smaller web applications.
* As it is one of the widely used DBMS, many tutorials are provided in the internet.

## 4.2 Development Environment

For developing our application, we have used various tools which are described below:

### 4.2.1 Platform

For the platform, we have chosen Docker as our development environment with Laradock. This includes pre-packaged docker images for providing perfect PHP development environment. Laradock is a Docker PHP development environment that facilitates running PHP Apps on Docker.

### 4.2.2 Framework

For our project, we have used Laravel as our framework for PHP. This framework is the leading PHP open source framework. This framework provides us lots of functionality that helps us in building dynamic application easily and quickly. This framework helps us to make the application more secure. It is also developer friendly and anyone can get huge support from the internet. Many Packages can be installed easily in this framework that makes easier to us.

### 4.2.3 IDE (Integrated Development Environment)

For our project, we have used PHP Storm as our IDE. This tool is very intelligent and supports us while building web applications. This IDE is finest for code navigation and permits for harmless and quick refactoring. It gives us the finest interface and user experience with wonderful support for PHP. This IDE also has built in terminal which makes us easy writing commands.

### 4.2.4 Administration tool for MySQL

For our project, we have used Heidi SQL as our administration tool for MySQL. This tool gives GUI feature which helps us to get server connection, understand the database, table, views, procedures, triggers and events.

## 4.3 Development Strategy

For Development Strategy, we have discussed two different points below.

### 4.3.1 System migration

When the system is completed, we have migrated it from local server to production server. We have transferred application files with the help of FTP and SSH.

### 4.3.2 Data migration

As we have all the data of application in Heidi SQL so all the data of database including table were exported to SQL file.

## 4.4 User Training

User training is the important part of implementation phase where users can get all the knowledge regarding the newly built application. With the good training user can use the full out of the system. A good trainer helps user to know all the features of the system. For the better understanding of the application, User Manual was also provided.

## 4.5 Conclusion

Finally, the implementation phase is finished. Here, we have shown the programming language, development environment and additional aspects like framework, platform and IDE. We have described the importance of user training and we have similarly provided a assistance guide along with it.

# Chapter 5: Other project issues:

## 5.1 Project Management

Aside from building up an application project management is additionally one of the critical aspect in programming. Project management is the way toward dealing with every one of the perspectives accessible, for example, asset, time, information and so forth so project can be finished as expected. There is different project that task supervisor ought to do and they include distinguishing and stakeholder and conveying between them, keeping up project quality and so on.

Numerous issues may raise while we are dealing with the project, for example, absence of labor, project deviation and some more. Among them project deviation is the one of the issue that will emerge while building up the project. It is the condition in which expected task is huge more unique in relation to the undertaking that have been produced. Such issue may make extend come up short and a few negative circumstances will emerge. The different sorts of project deviation that can happen in our task is as follows:

• Time management is one of the task deviation. Project ought to be finished in given time and if not finished project may have certain negative effect, for example, punishment, application without customer necessity and so forth. So, for this deviation we should perform prioritization systems, time estimation methods and some more.

• Cost management is likewise one of the project deviation. The evaluated cost for the project in the first place may not be adequate and we might be in huge loss so appropriate spending estimation ought to be done as such that this issue may not emerge.

• Risk management is additionally project deviation. Hazard ought to be constantly distinguished and appropriate reinforcement design ought to be created as an answer for this issue generally may totally hurt the project. So legitimate risk management ought to be finished.

• Managing HR is additionally project deviation. Human asset ought to oversee in a legitimately with the goal that we could have no issue building up the undertaking. So, for this we ought to comprehend the necessity of human asset.

## 5.2 Risk Management

Risk management is the way toward distinguishing, breaking down, contrasting and furnishing arrangement with the hazard that has been raised while building up any project. Many individuals commit error of not considering risk important and later they need to experience the ill effects of gigantic loss as an absence of risk management. Risk ought to be overseen and keep up for project success. In our proposition area, we have characterized certain risk for our undertaking. We have denied diverse risk, for example, change in necessity, hard disk crashed and so forth.

We had confronted many dangers in our venture, for example, change in prerequisite, hard disk crash and so on bit no issue happened because we had just worked for limiting those hazards. More dangers have happened while building up our task and some of them are as per the following.

• Lack of suitable time was one of the hazard. Because of various festival our project had issue in playing out some assignment and it was done of late so I consider it as hazard.

• Using new framework was likewise a hazard as we had confronted numerous issue and blunder while finishing assignment.

## 5.3 Configuration Management

Configuration management is the way toward refreshing and keeping record of the project.it had given different favorable position while finishing our task and they are as per the following.

• Due to this part and sub segment of the framework were sorted out in legitimate way.

• Proper recording and refreshing consider overseeing project proficiently.

• Helps staying up with the latest and so on.

## 5.4 Testing

After the product have been created the product must be tested. It is the way toward guaranteeing that the product we have created fill in as we anticipated that or not. It helps to explore whether the program and code we create is free from error and robust. Moreover, it additionally helps in checking whether the product is executing according to arrange and satisfying the client prerequisite. Numerous issue will emerge if program isn't tested properly. In this task, we have performed two diverse sorts of testing.one is the joining testing and other is the unit testing. Both testing is as follows:

## 5.5 Unit Testing

It is testing of individual program modules or segments. It falls under the class of white-box testing and frequently done by the software engineer to test the single capacity of the framework.

## 5.6 Integration Testing

It is testing of gathering of segments which are consolidated to deliver result. It falls under both black-box testing and white-box testing. It is testing where all the framework is tried.

# Chapter 6: Conclusion

At last we have touched base to the conclusion area. In this section, we give the outline of the aggregate application documentation. The application we were creating has been finished. The project we have created is the patient management system. This is the system that enable us to keep record of the general population that are associated with the organization. They incorporate patient, doctor, staff, Inpatient and appointment also. The framework gives diverse facilities, for example, managing record, searching them and some more. In addition, we likewise have done documentation with respect to the task.

I think the software I have created will have numerous business scope. This is imperative for hospital to look after record. More over its features helps for work automation.

# Chapter 7: Future Works

The application we are creating is very small. Many future work ought to be completed in this project. Some of the future work that should be performed are as per the following.

• I would utilize more secure strategy for login that consider assurance of information.

• I would include extra highlights that aides in making my undertaking more capacity capable.

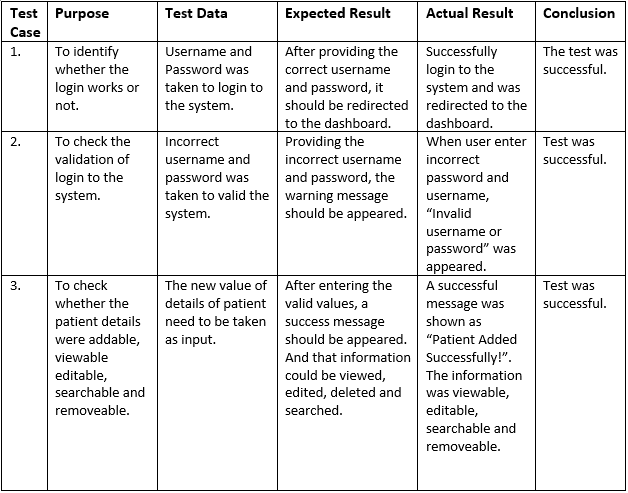
• I would utilize more secure database with the goal that application can be utilized for enterprises also.

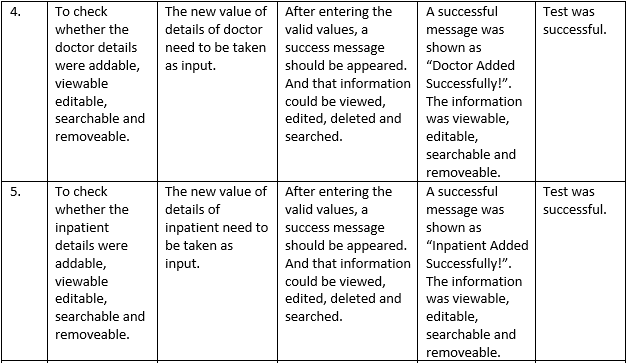
• I would expand the functionalities and features with the goal that system can be used by everyone.

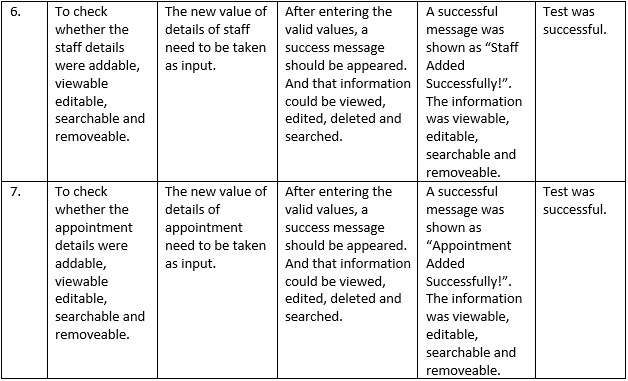
• I would utilize advanced java scripts i.e. Angular2, Vue2 and numerous other languages with the goal that it can be quick and more powerful.

# Chapter 8: Appendices

## 8.1 Test Script







## 8.2 Integration testing

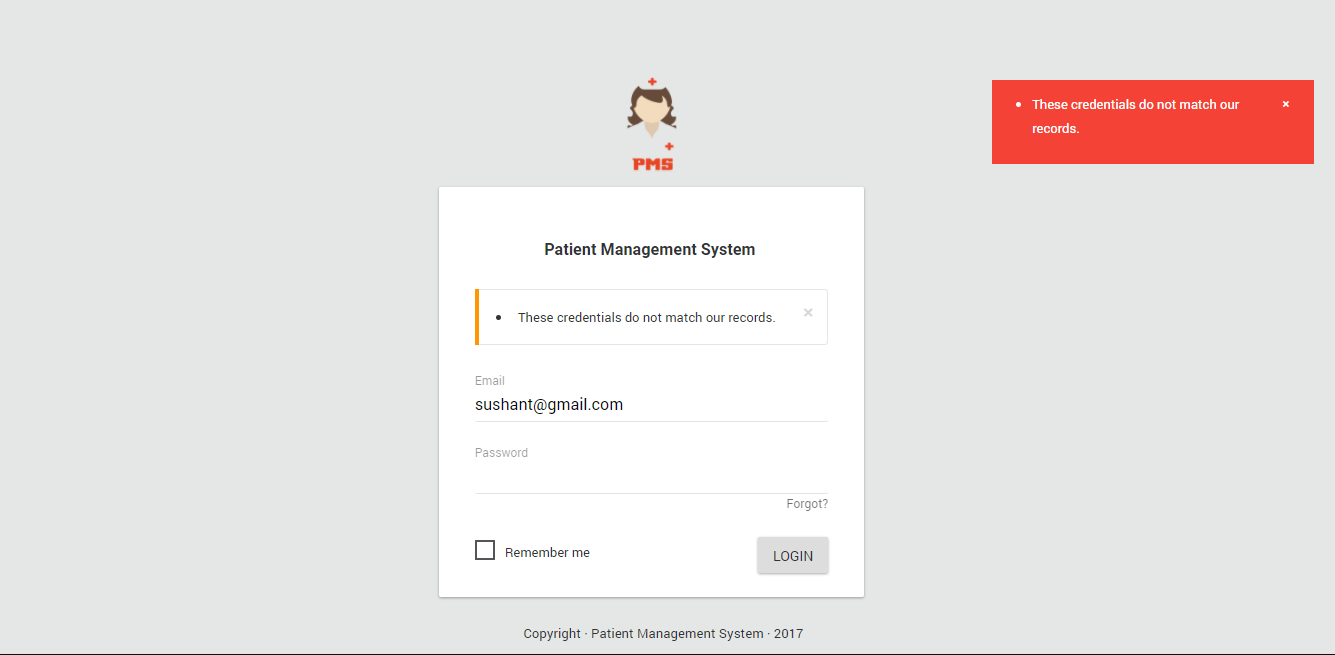


Fig: Login Failed for bad email or password.

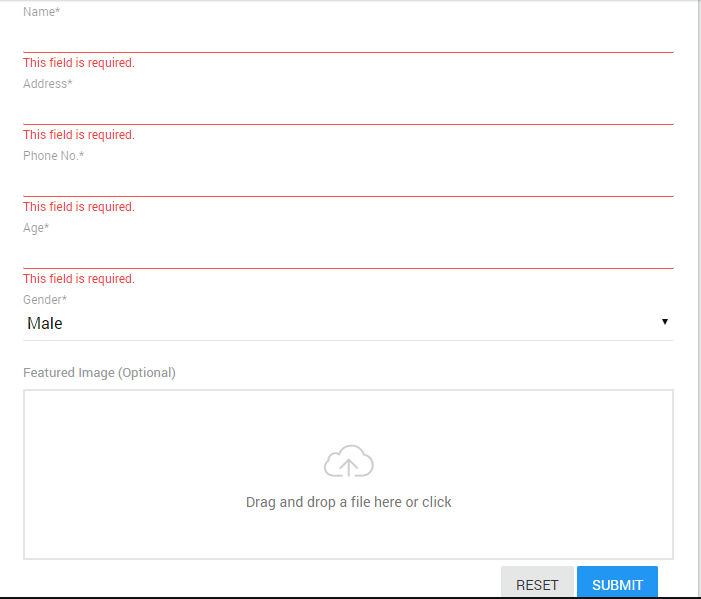


Fig: Frontend Validation for empty data submission.

## 8.3 User Guide

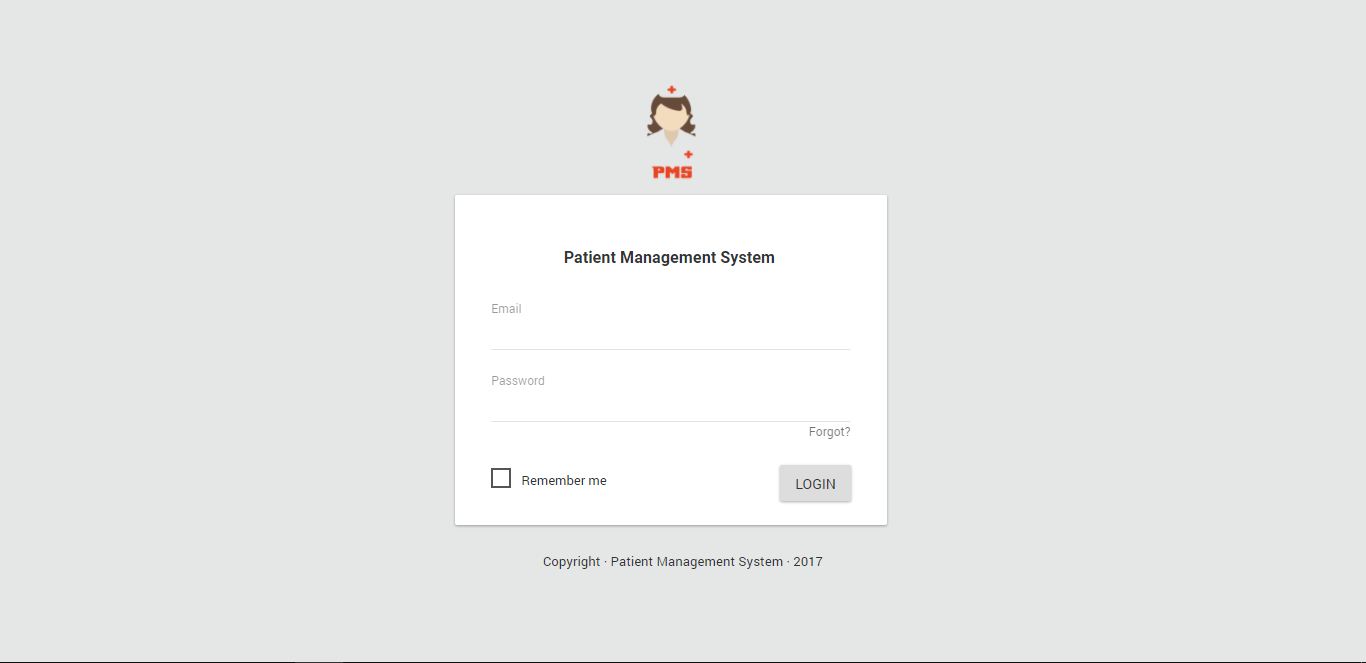


Fig: Login Panel

* Above picture shows our login form where credentials must be entered to get inside the application.

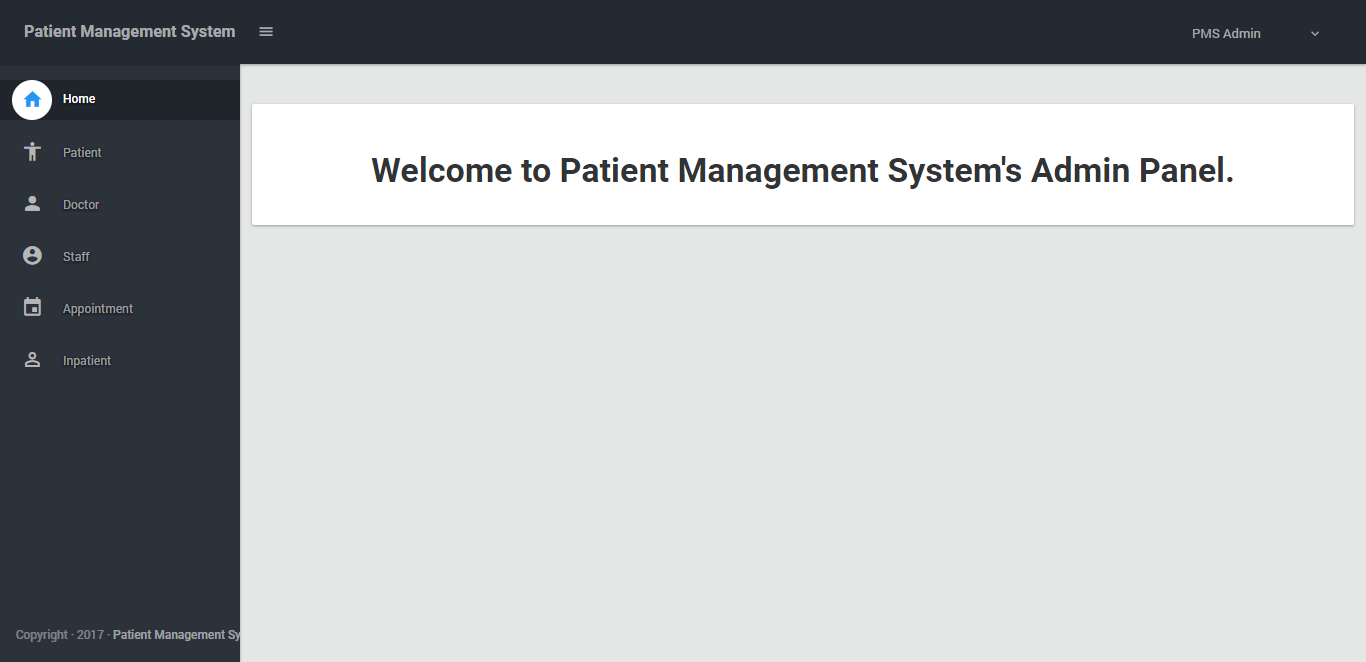


Fig: Dashboard

* After login this is the page where user is redirected. On the header contains the name of application and name of the user. Menu is in left hand side.

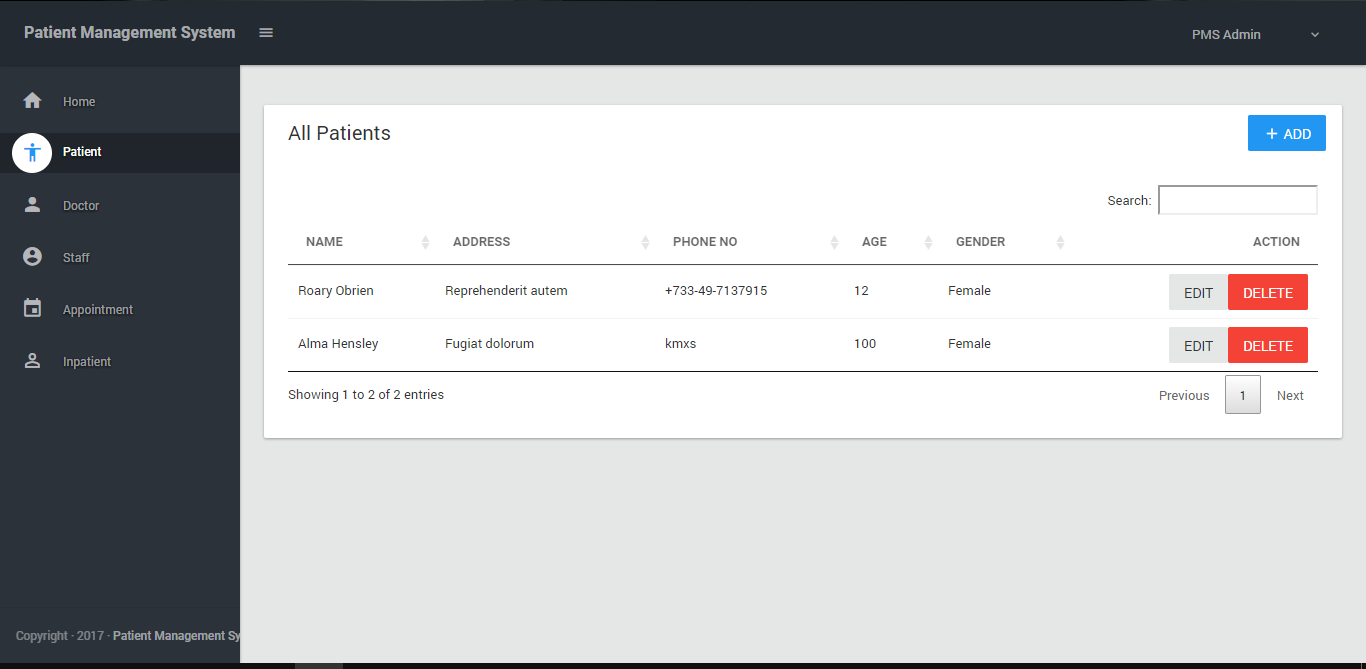


Fig: Patients list

* When user clicks the “patient” from menu then user is redirected in this page which shows the list of the available patients and offers user with the functionality to create new or edit and delete a particular patient.

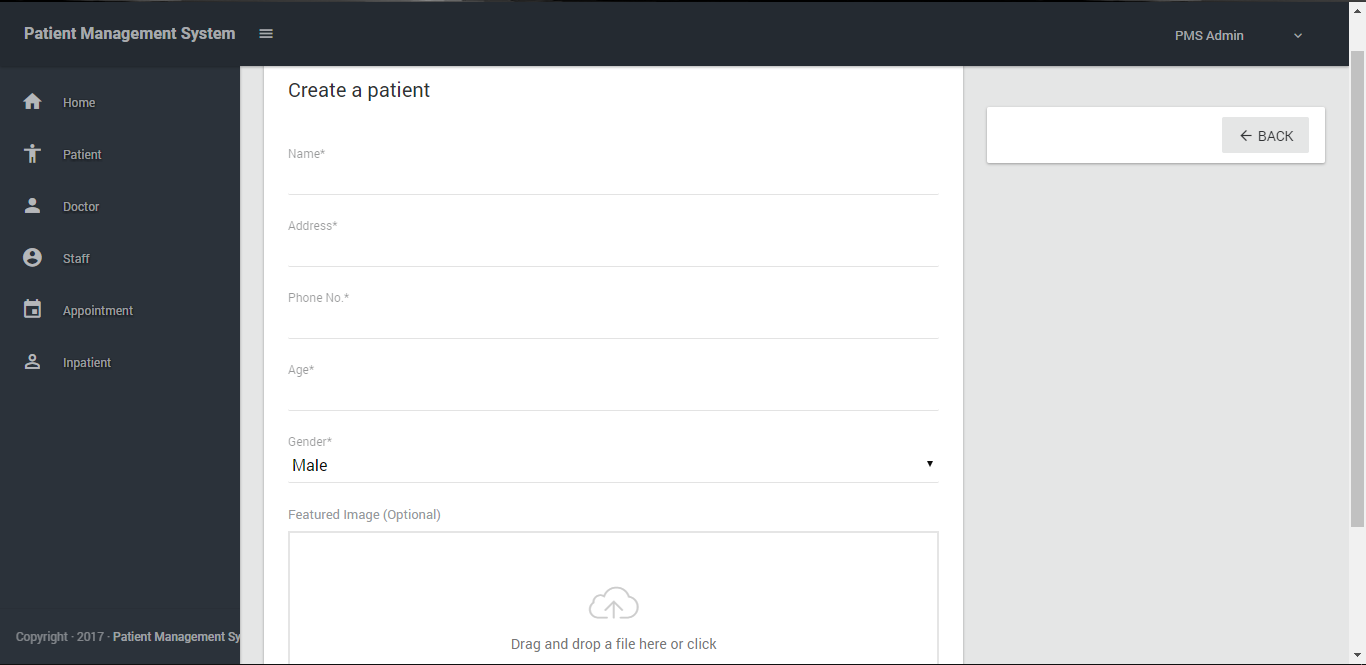


Fig: Create a patient

* When user clicks in Add button then this form appears where information of users is entered and are submitted. After user clicks submit then if all data are correct then “Patient Created Successfully” is shown. Then the data is visible in previous page.

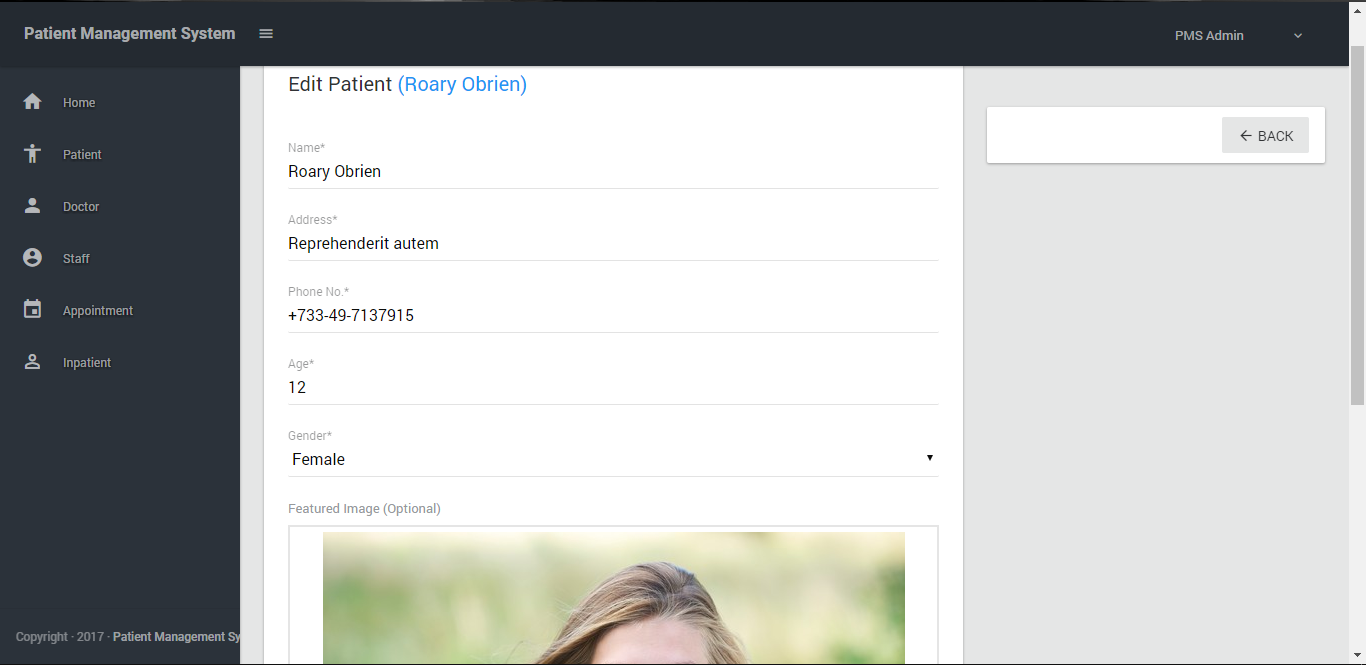


Fig: Edit Patient

* When user clicks in Edit button then this form appears where information of users is modified and are submitted. After user clicks submit then if all data are correct then “Patient Updated Successfully” is shown. Then the data is visible in its first page.

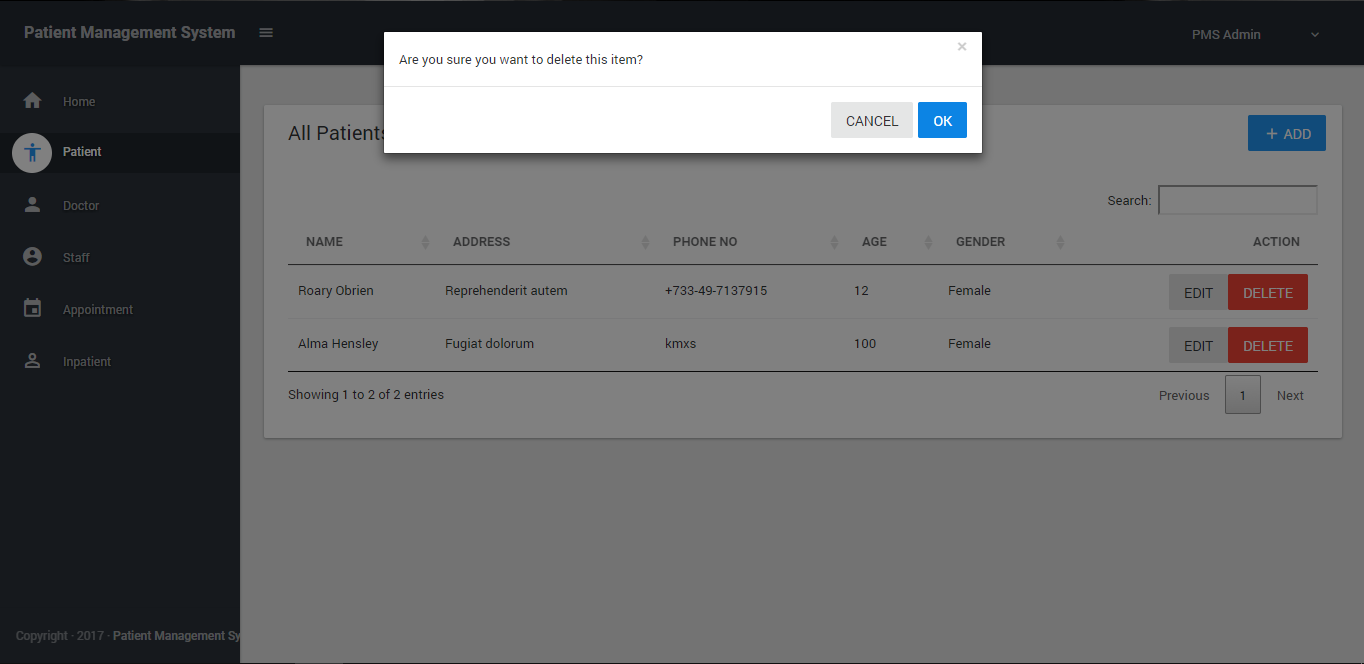


Fig: Delete Patient

* If the user wants to remove an info of a patient then he should click in “Delete” button. Then, the model appears with the above question. User are supposed to click “OK” to delete. All goes same for other menus too.

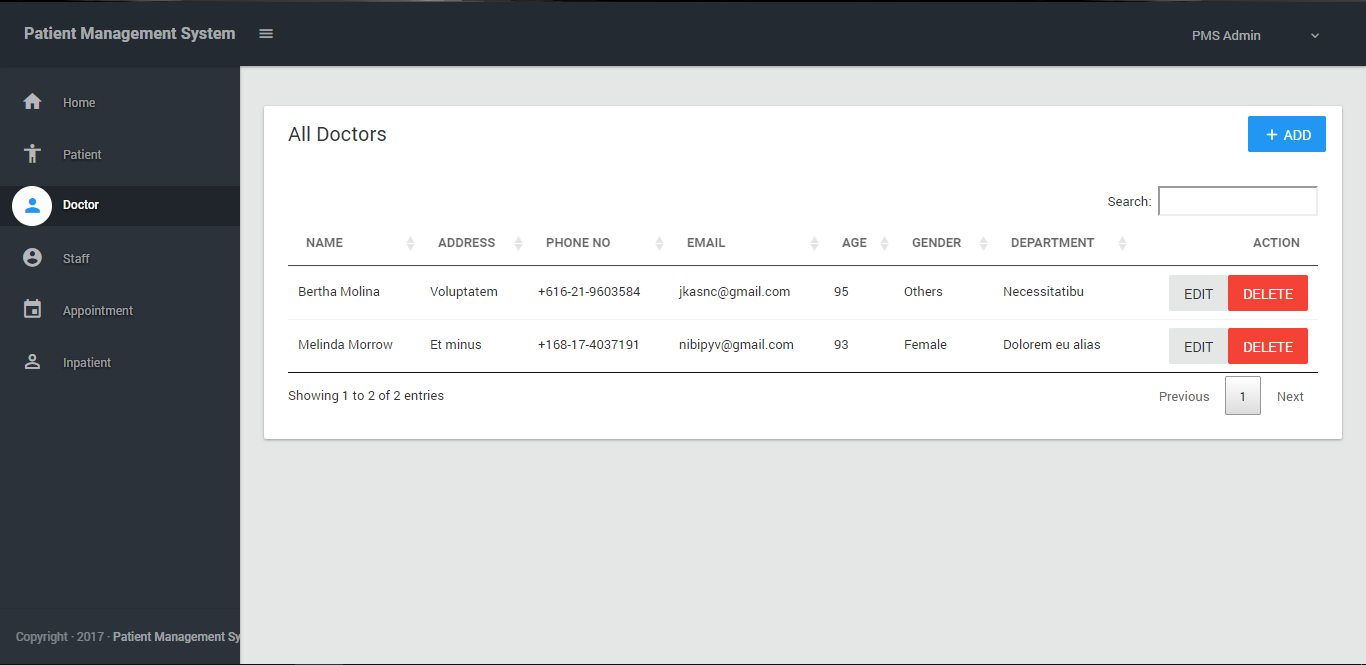


Fig: Doctors list

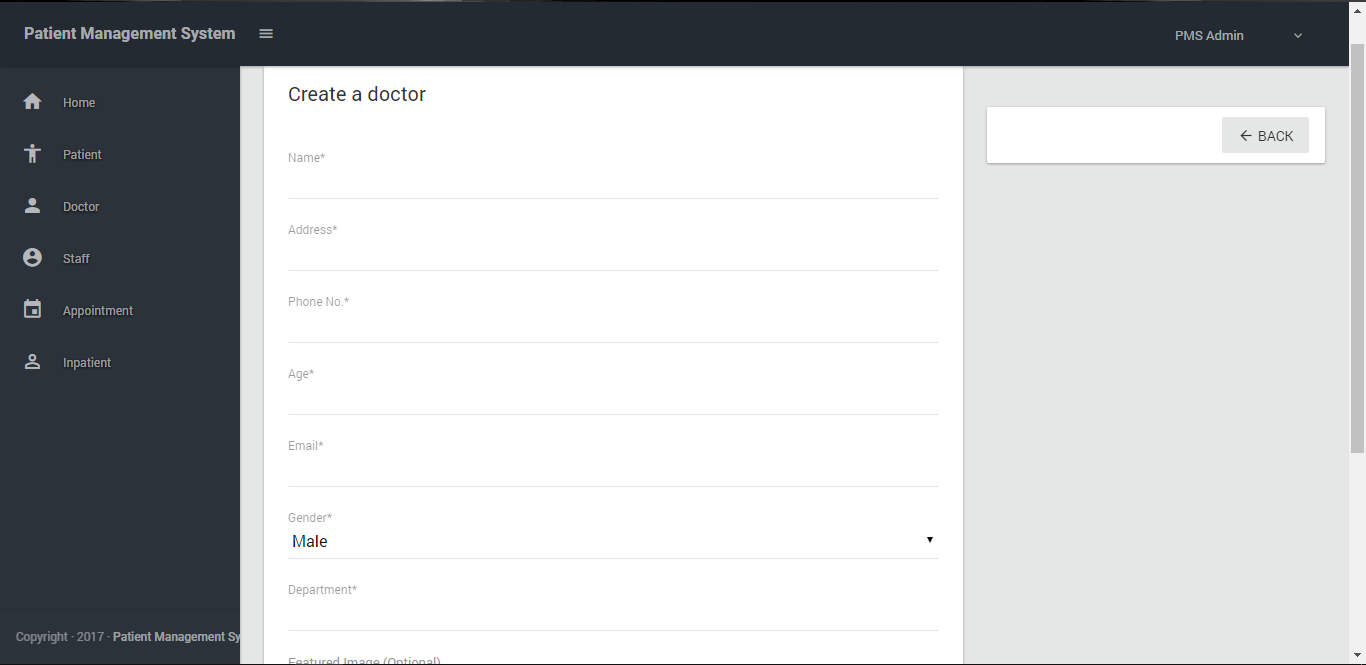


Fig: Create a doctor

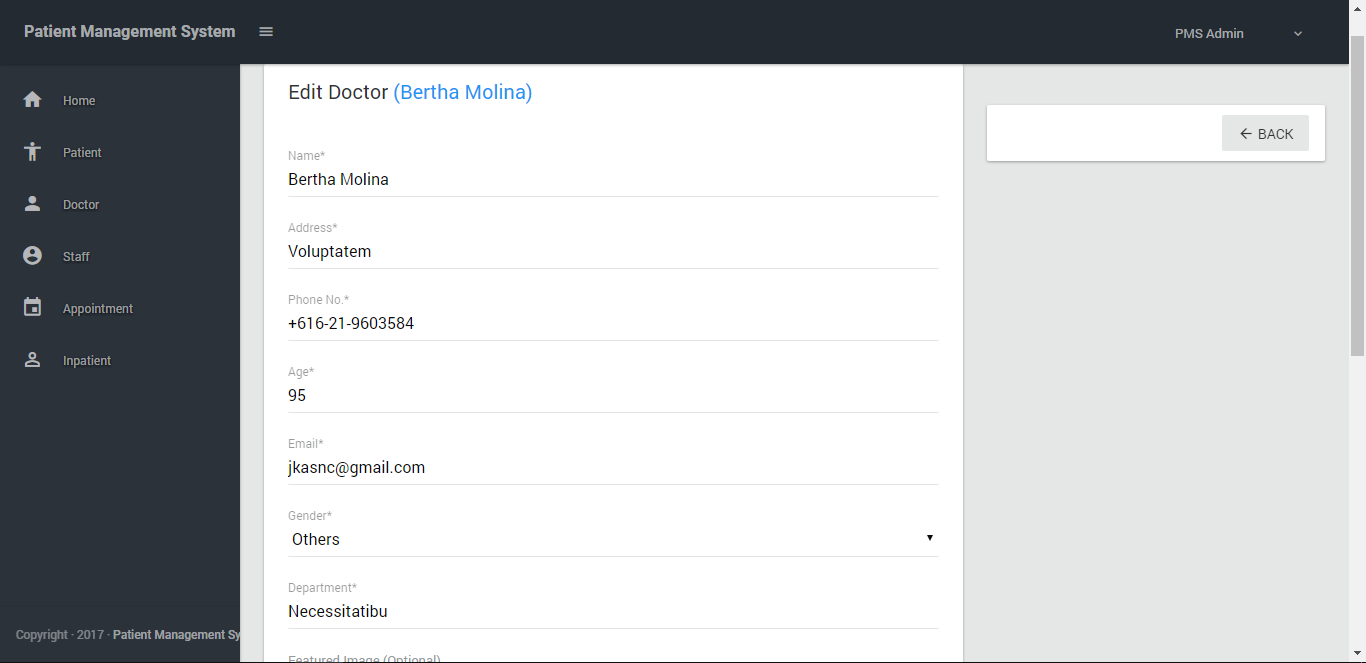


Fig: Edit Doctor

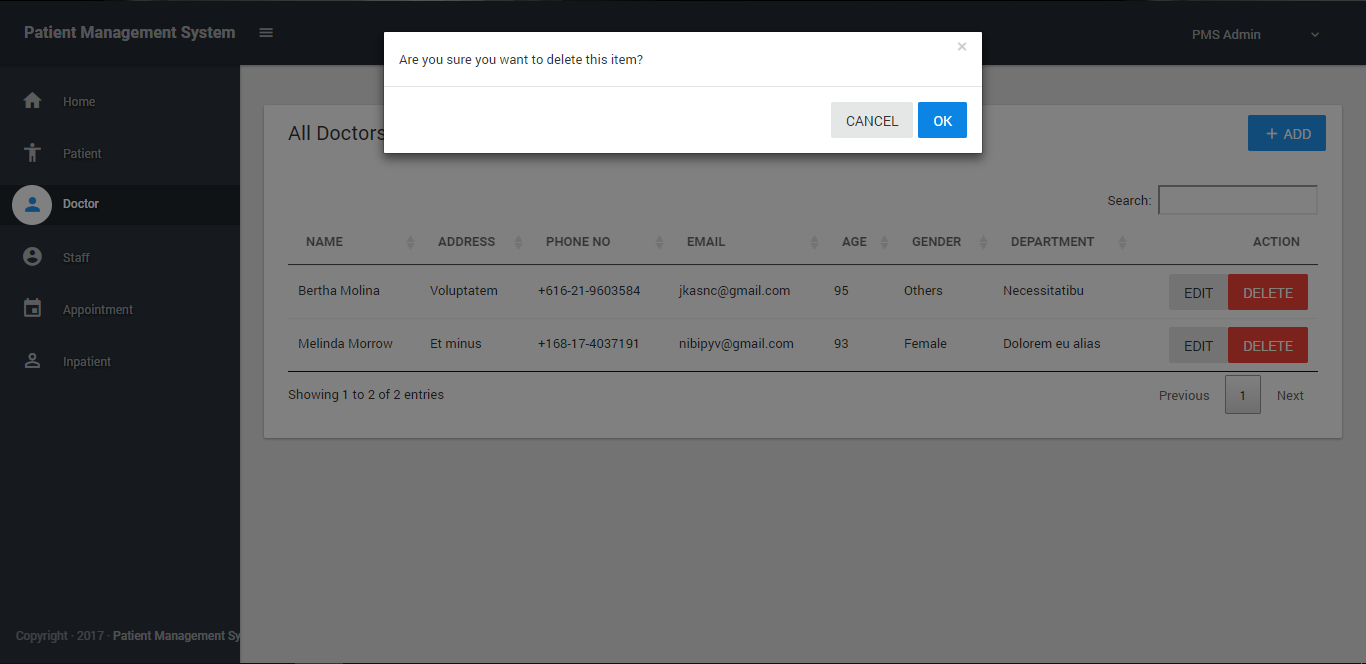


Fig: Delete Doctor

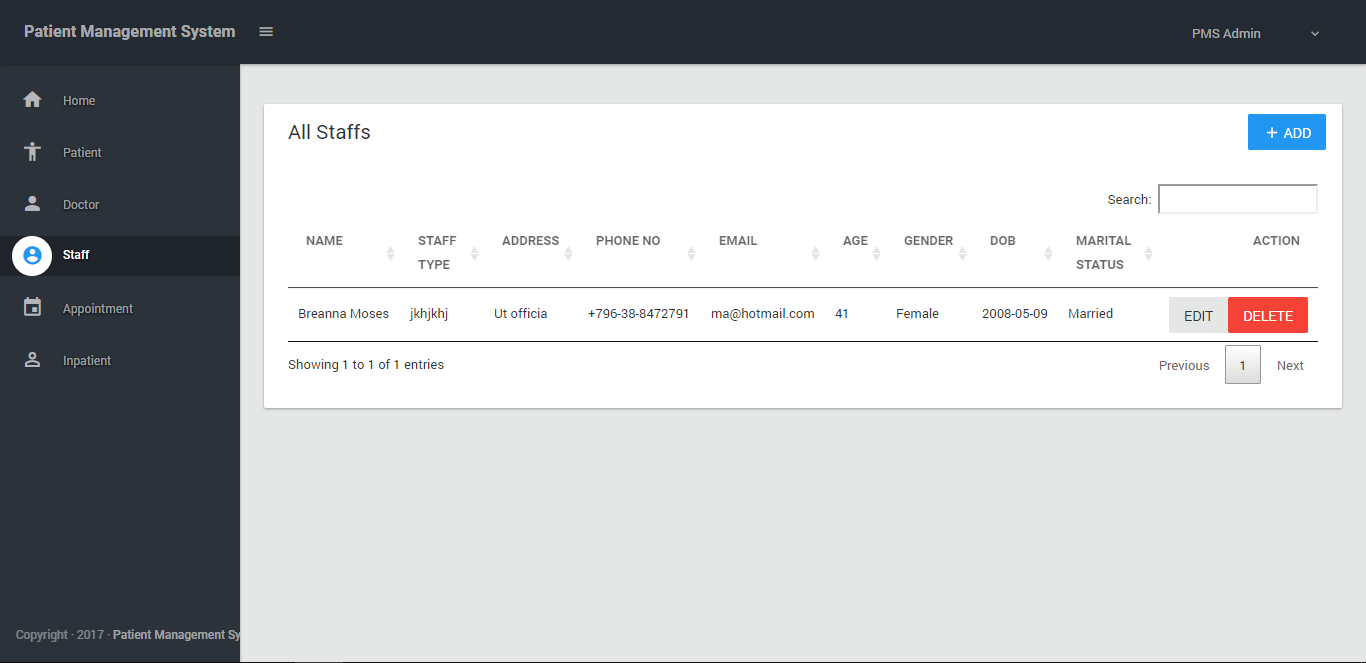


Fig: Staffs list

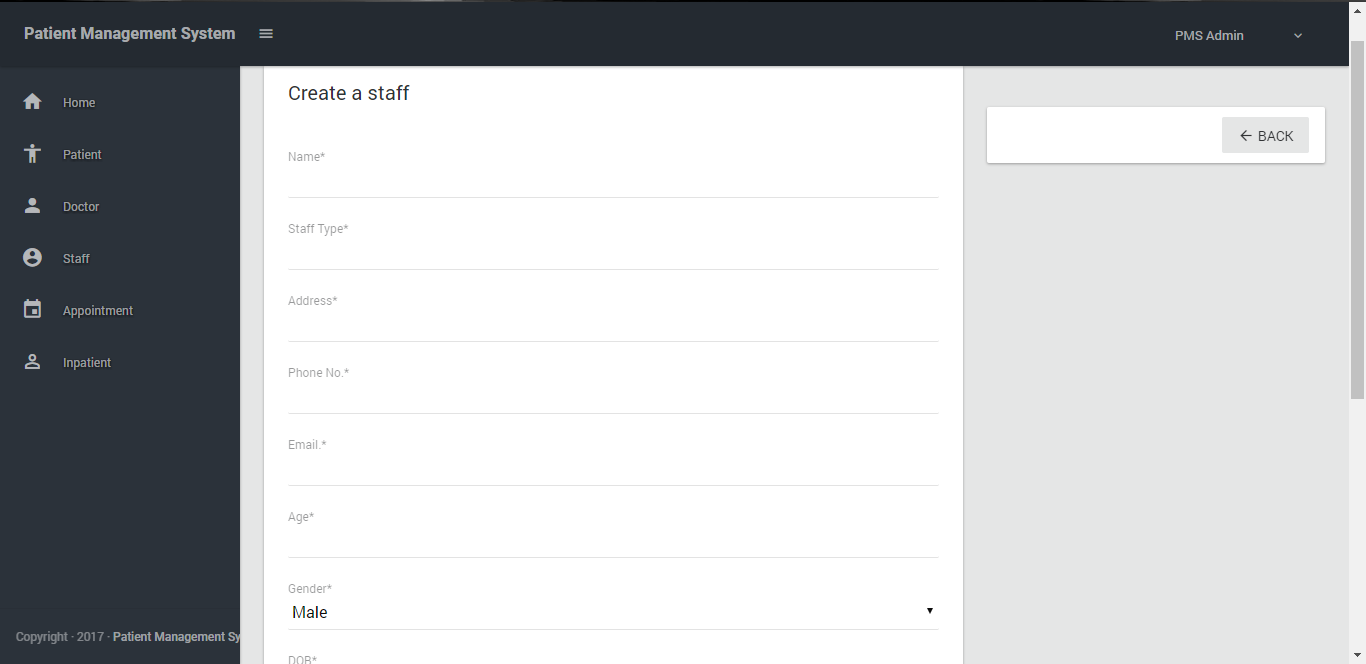


Fig: Create a staff

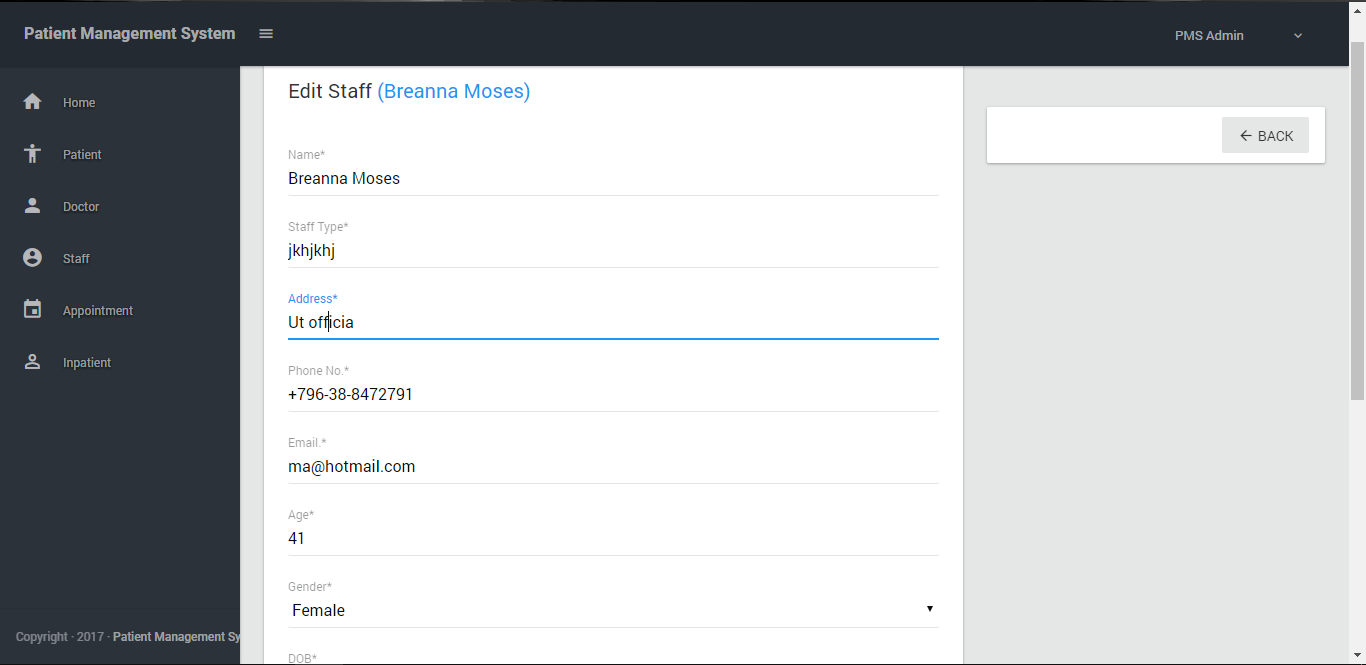


Fig: Edit Staff

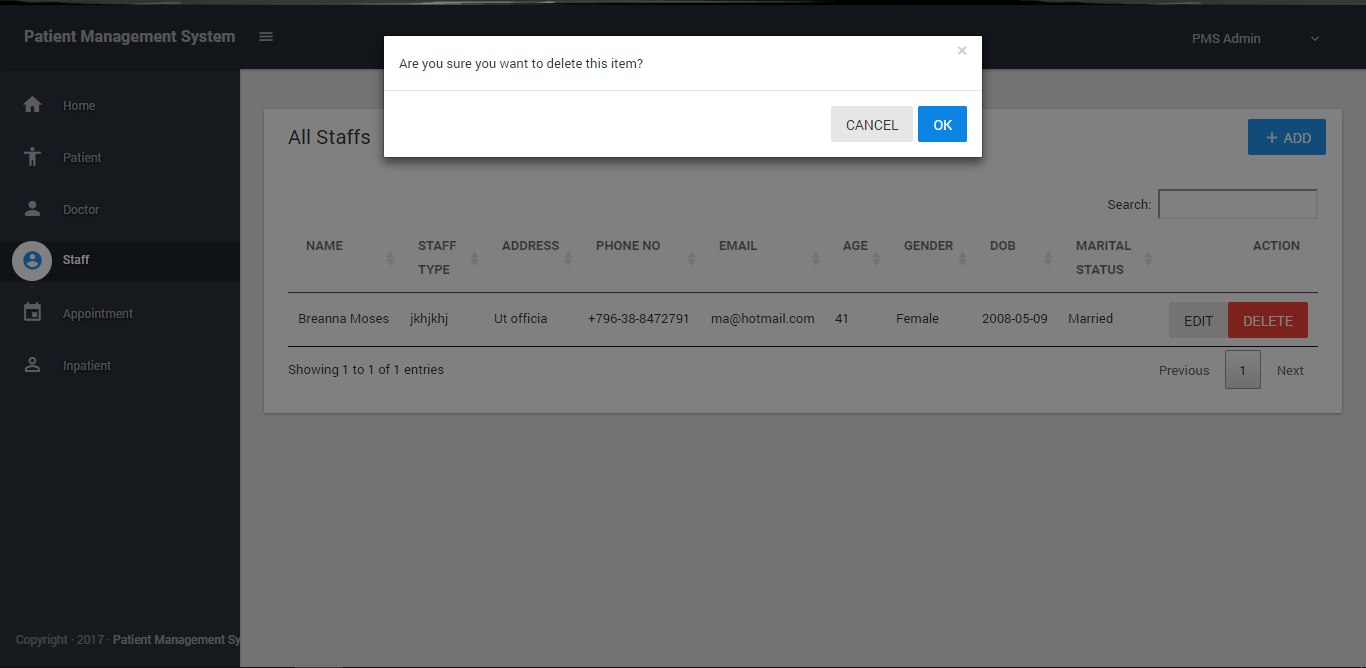


Fig: Delete Staff

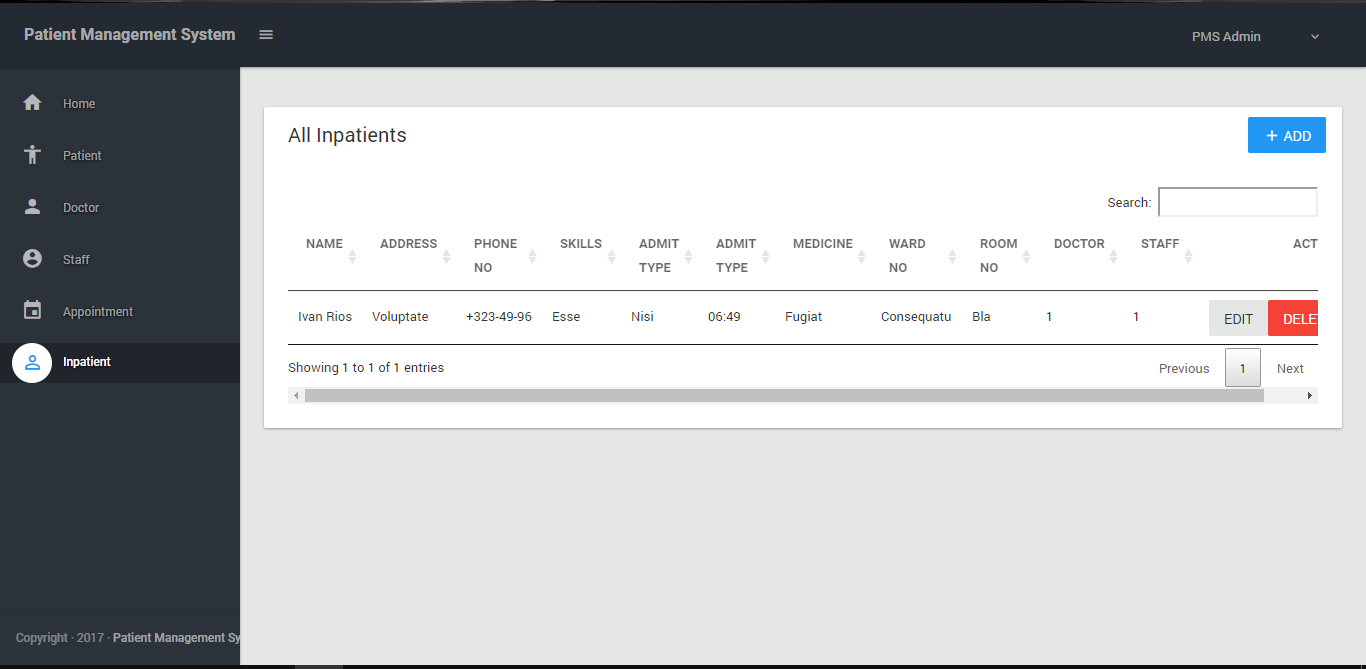


Fig: Inpatients list

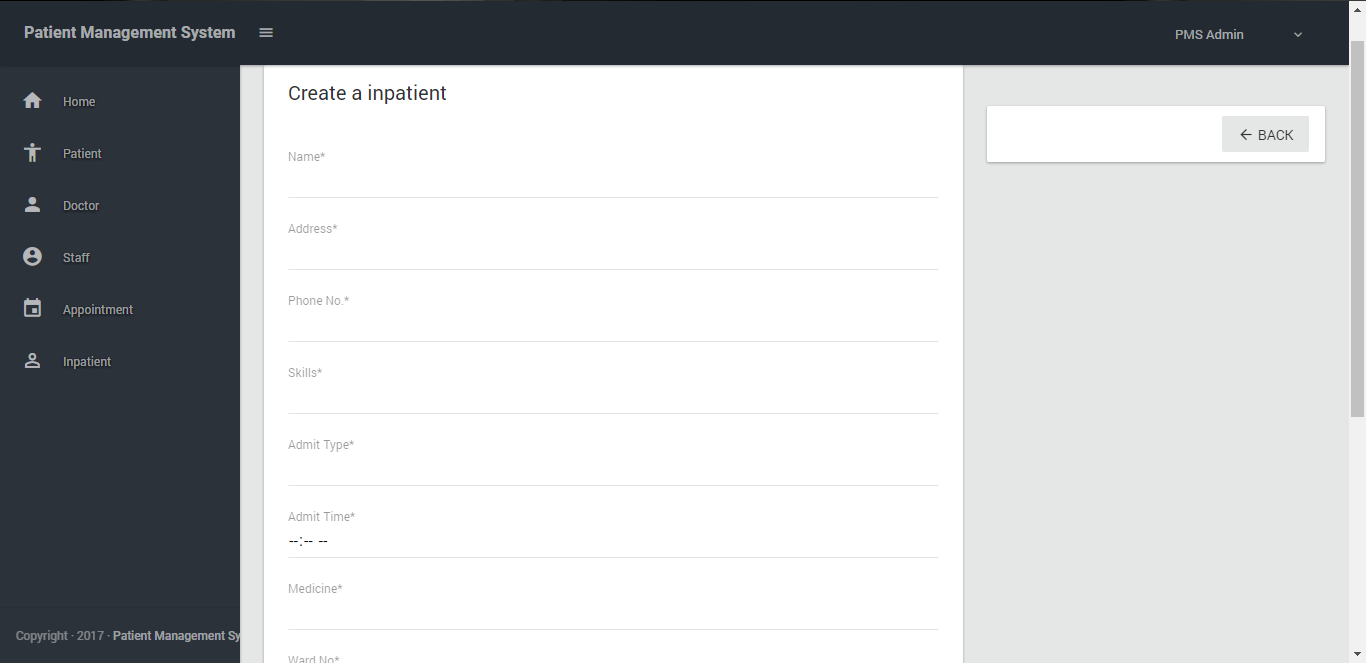


Fig: Create an inpatient

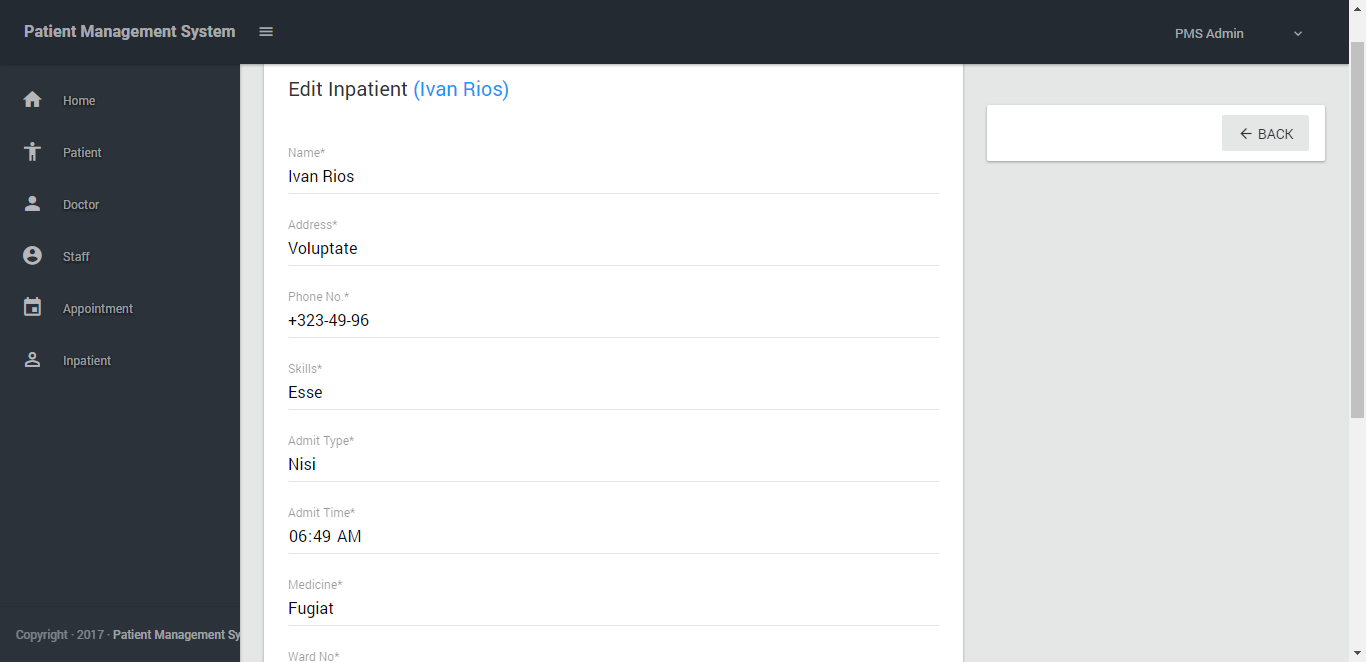


Fig: Edit an inpatient

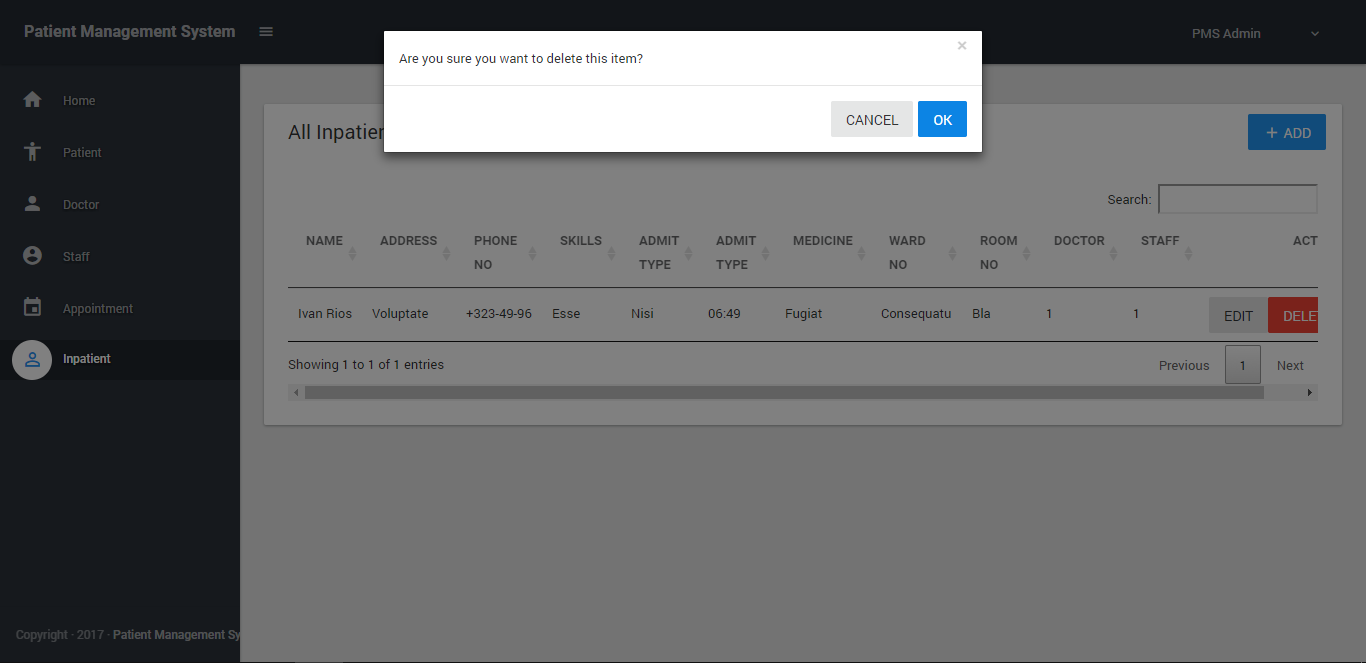


Fig: Delete an inpatient

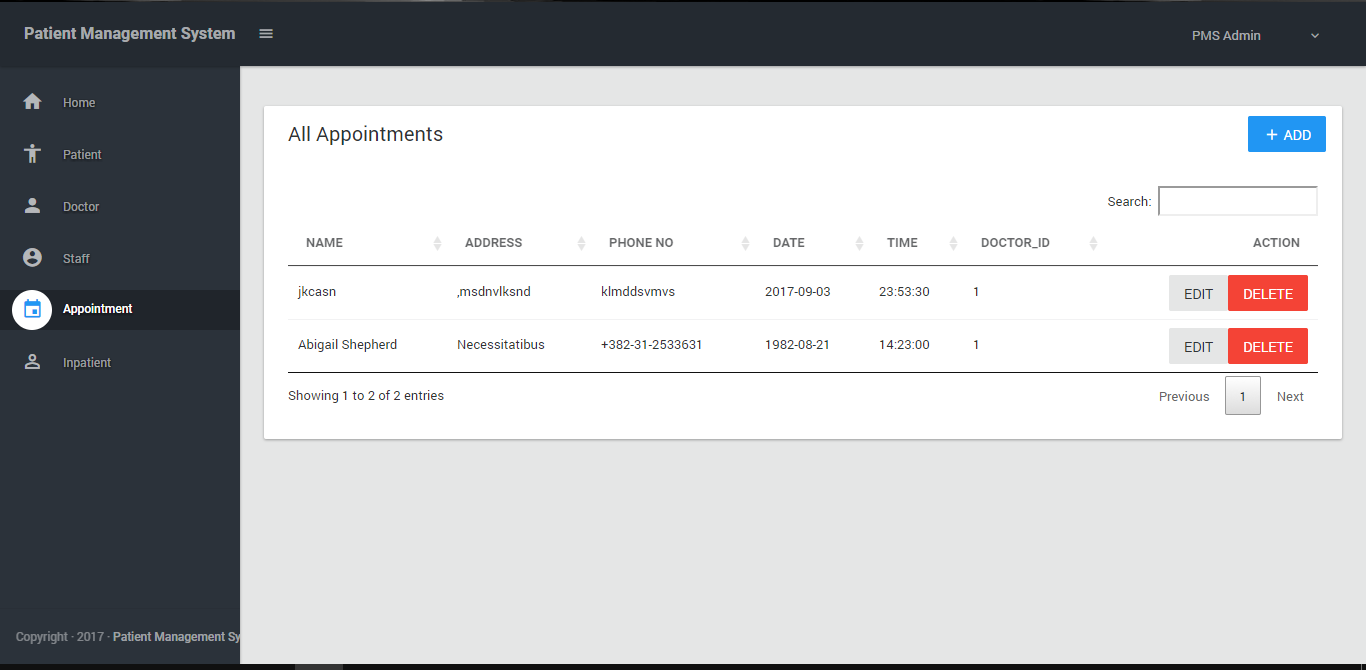


Fig: Appointments list

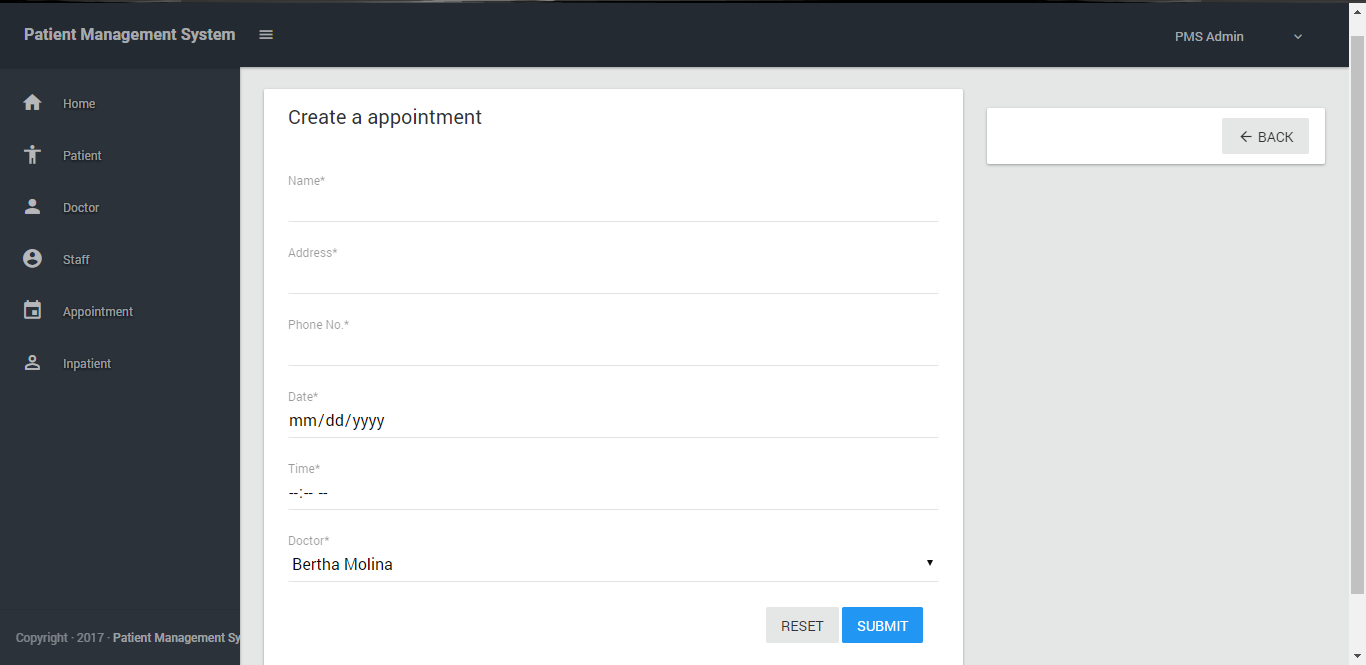


Fig: Create an Appointment

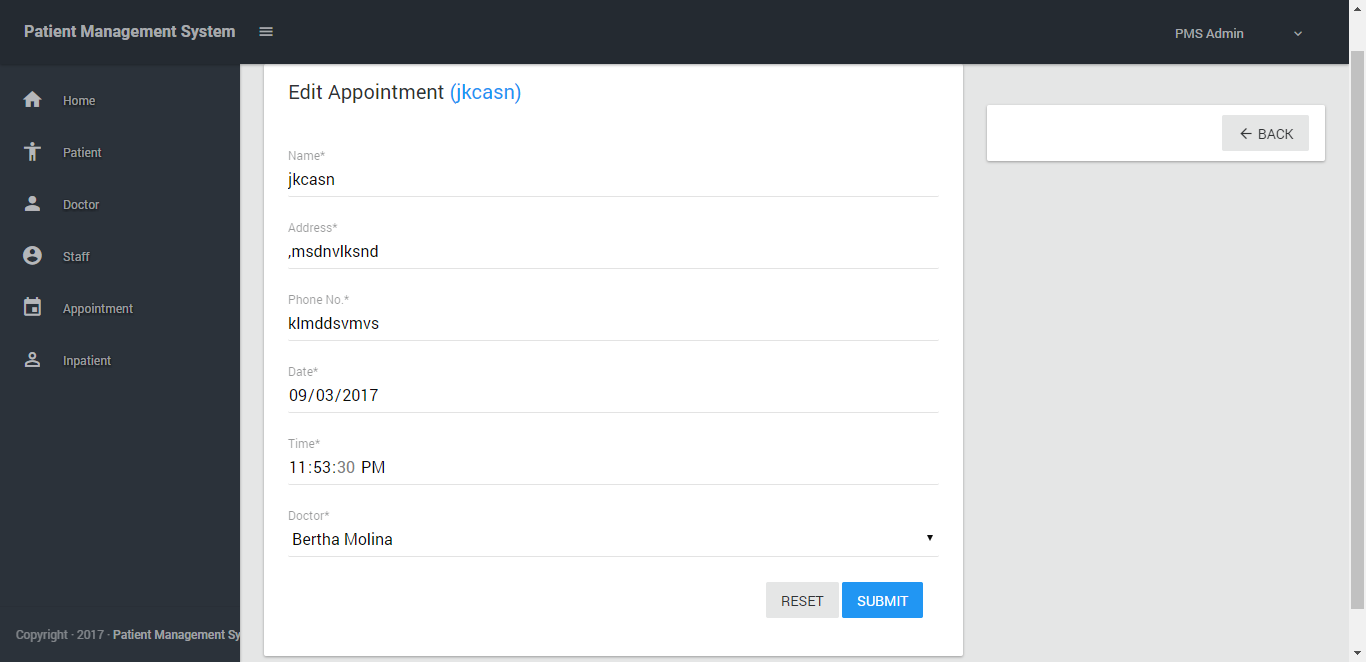


Fig: Edit an Appointment

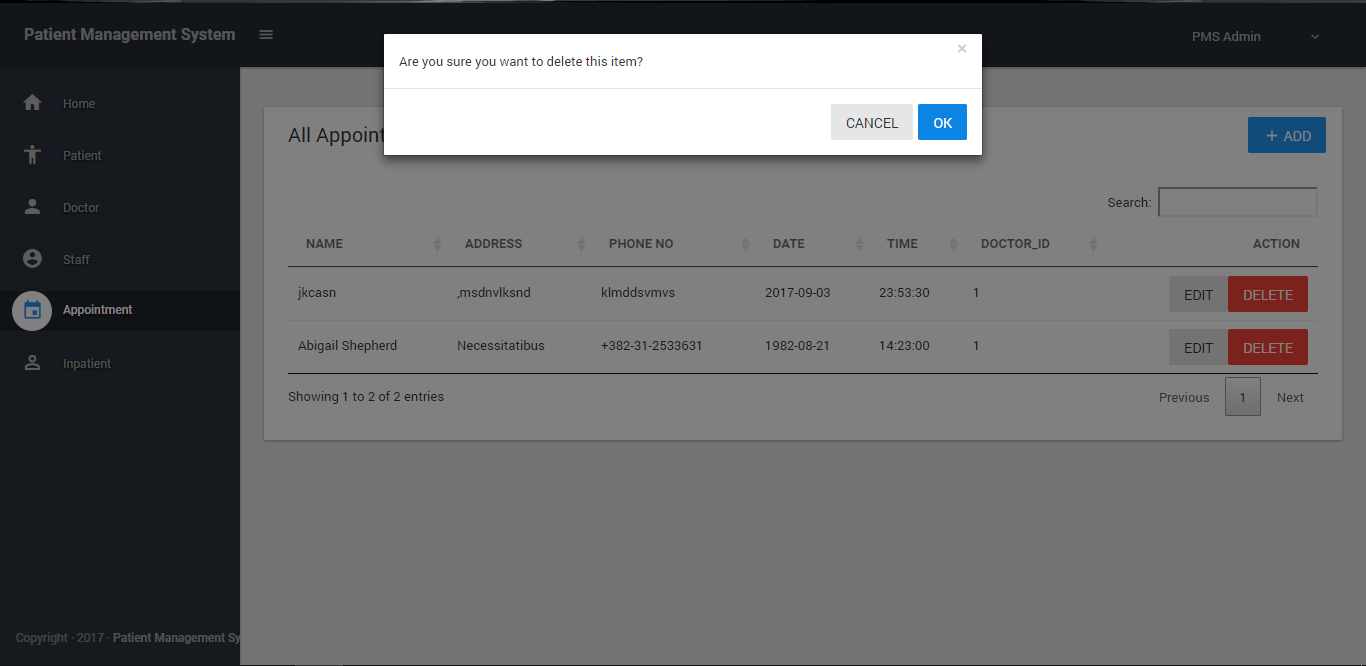


Fig: Delete an Appointment

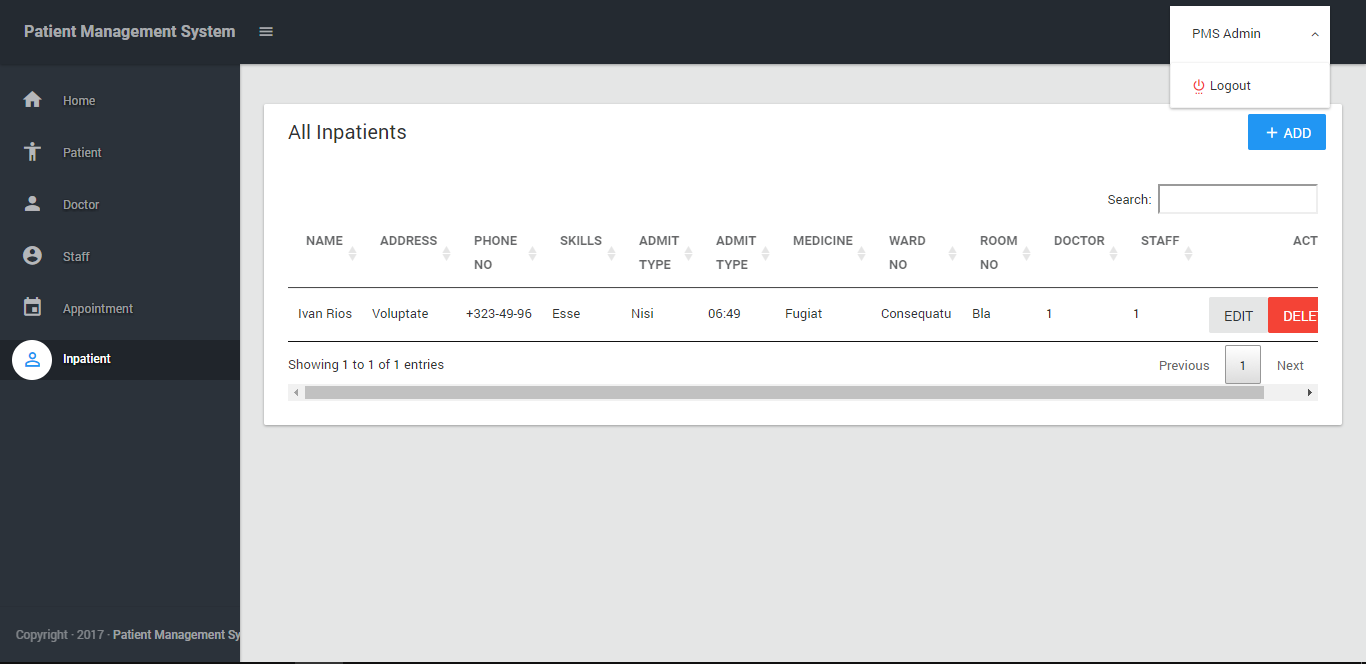


Fig: Logout from Application.

* Finally, when every this is completed users can exit from the application using logout button on the right top of the page.

# Bibliography