Final Report

Pasindu Jayawardene [2009084]
Nithya Ranadeerage [2009083]
Randula Wijesinghe [2009067]
Dinuk Caldera [2009062]

Acknowledgement

This final report of the project would be incomplete and wouldn't have been successful in coming to light, without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

We are grateful to our project guiders, Mrs Udayangi Perera, Mr. Pragathi Weerakon, Mr. Saman Hettiarachchi for the guidance, inspiration and constructive suggestions that helpful us in the preparation of this project.

We are grateful to Dr. Malaka Jayawardene at the National Hospital, Colombo for taking time to discuss the prevailing medical problems in the Sri Lankan Medical System, as well as his constructive suggestions that led to make this project into a better one.

We also thank our colleagues who have helped in successful completion of this project.

Contents

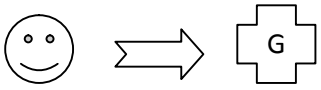
Introduction

The software in discussion is a patient management system, which is used to ***store and retrieve patient details***, that would help the doctor to ***trace back*** on the patient's past medical history, in a comprehensive manner , ***at any given time***, while reducing his tasks and helping him to come to a diagnose more quickly. The software would brings efficiency to the on-going medical procedures in Sri Lankan hospitals.

The system works to its full potential when its installed in multiple hospitals/clinics across the country where a patient in its database can enter any hospital in any part of the country and receive medical care, according to his previous medical conditions and records since he is in the patient database.

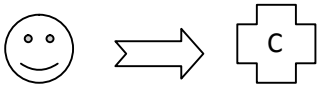
Scenario:

8 Years Ago



- Patient Receives Medical Treatment for a Heart Condition from a Hospital in Galle.
  - ✓ The System stores the Patient Details and the Treatments/ Procedures Underwent.

8 Years Later



- Patient rushed into a hospital in Colombo after a Car Accident.
  - ✓ All of patient's past medical proceedings can be obtained from the System.

The software is to be used in

- Hospitals
  - Medicine Department
  - Surgery Department
  - Obstetrics and Gynaecology Department
  - Paediatrics Department
- Clinics
- Pathology labs

Objectives

- Bring Efficiency to the Current Procedures
- Reduce the Wastage of Hospital Resources
- Organized and Comprehensive Patient Details

Main Characteristics of the System

- Patient Details Database
- Patient's Ongoing Investigations (Biopsy Reports, CT Scans, etc.)
- Blood Donor Database
- Epidemic/Infectious Disease Warning System

Patient Details Database

Patient Details will be recorded under these categories

1. Patient's Basic Details
   Name, Birthday, Age, Address, Blood Type, etc.
2. Presenting Complaint with the Date of Admission
3. History of Presenting Complaint
4. Systemic Review of the Patient
5. Past Medical History (Eg: Diabetes, Bronchial Asthma, Hyper Tension, etc.)
6. Drug History
7. Past Surgical History
8. Family History
9. Allergic History
10. Social History (Eg: Smoking, Alcohol, etc.)

Technologies Used

The Java Persistence API

The Java Persistence API is a standard framework that provides persistence and object-relational mapping as a part of the EJB 3.0 specification.

- It is a Java standard that provides many of the benefits of alternative persistence frameworks, with the added benefit of portability to any EJB 3.0 compliant container, without having to install any extra libraries.

- It allows you to persist plain-old Java objects to a relational database – a much simpler approach than using container-managed persistence (CMP) in EJB 2.1.
    - ✓ No deployment code or abstract classes.
    - ✓ No interfaces required.

- Suitable for use in different modes .
    - Standalone in Java SE environment .
    - Hosted within a Java EE Container .

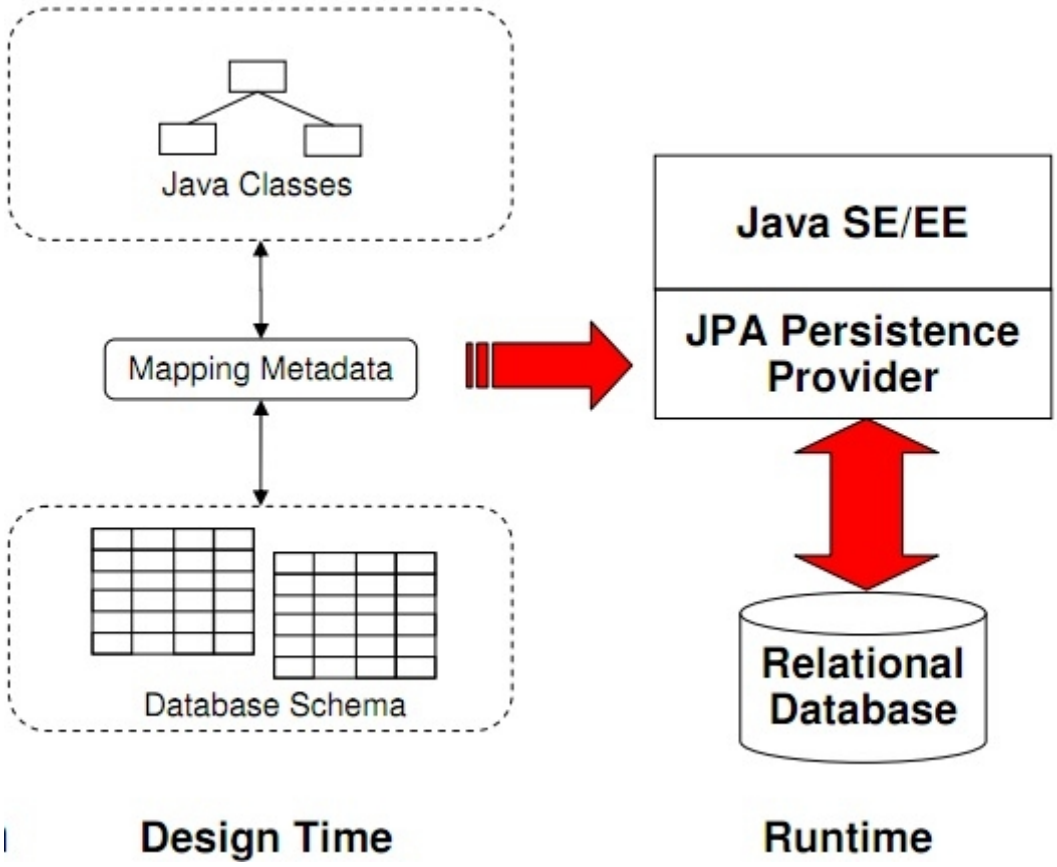
- Standardization of current persistence practices .

- Merging of expertise from persistence vendors and communities including: TopLink, Hibernate, JDO, EJB vendors and individuals.

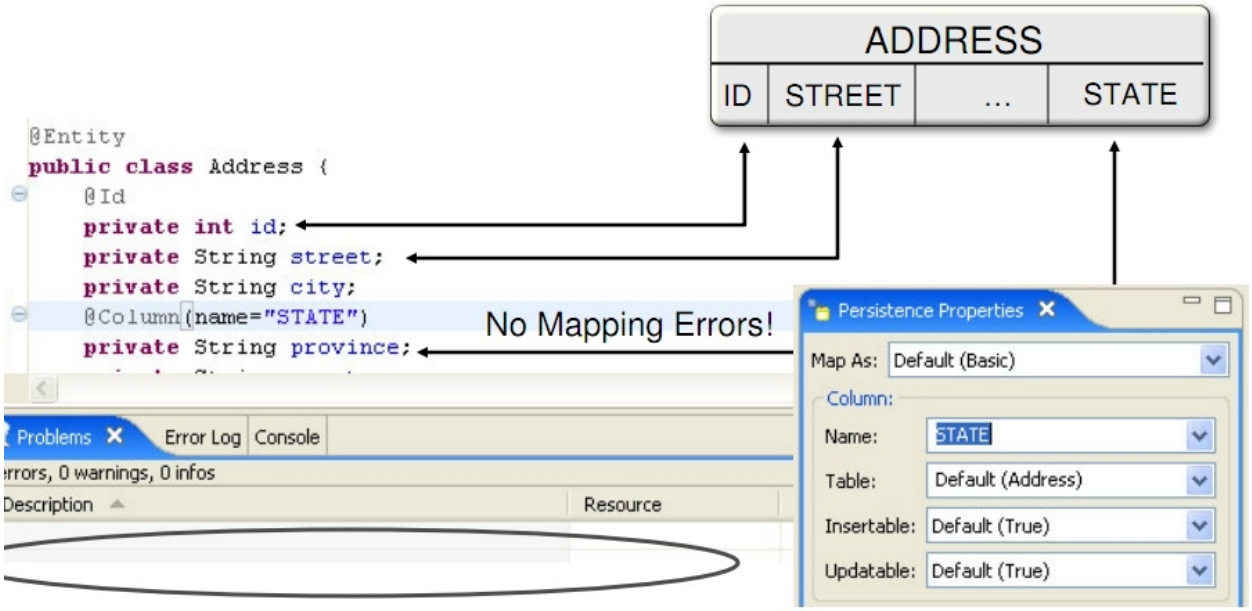
Object-Relational Mappings

Core JPA Mappings

- Id .
- Basic .
- Relationships .
  - OneToOne .
  - OneToMany/ManyToOne .
  - ManyToMan.



Basic Mapping system



Interface EntityManager

Interface used to interact with the persistence context.

An EntityManager instance is associated with a persistence context. A persistence context is a set of entity instances in which for any persistent entity identity there is a unique entity instance. Within the persistence context, the entity instances and their lifecycle are managed. The EntityManager API is used to create and remove persistent entity instances, to find entities by their primary key, and to query over entities.

MySQL

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases .

Tools :

MySQL Community Server
(Current Generally Available Release: 5.5.11)

http://www.mysql.com/downloads/mysql/

MySQL Connectors

MySQL offers standard database driver connectivity for using MySQL with applications and tools that are compatible with industry standards ODBC and JDB.

Why MySQL ?

- **Ease of use**—Go from download to complete installation in less than 15 minutes.
- **Low TCO**—Deploy MySQL for mission-critical applications with significant cost savings over Microsoft SQL Server
- **Scalability and performance**—Meet the scalability and performance requirements of the most trafficked web sites and the most demanding applications
- **Production support**—Oracle Premier Support helps lower the total cost and risk of owning your MySQL solution.

Sample Output :

```
 +-----+------+-----------+---------+---------+-------+-------+-------------+
    | Id  | User | Host      | db      | Command | Time  | State | Info
|
   +-----+------+-----------+---------+---------+-------+-------+-----------+
    | 657 | prog | localhost | weather | Sleep   | 28619 |       | NULL

    | 782 | prog | localhost | weather | Sleep   |   853 |       | NULL
|
    | 785 | prog | localhost | NULL    | Query   |     0 | NULL  | show  |
    +-----+------+-----------+---------+---------+-------+-------+-----------+
    3 rows in set (0.00 sec)
```

NetBeans IDE

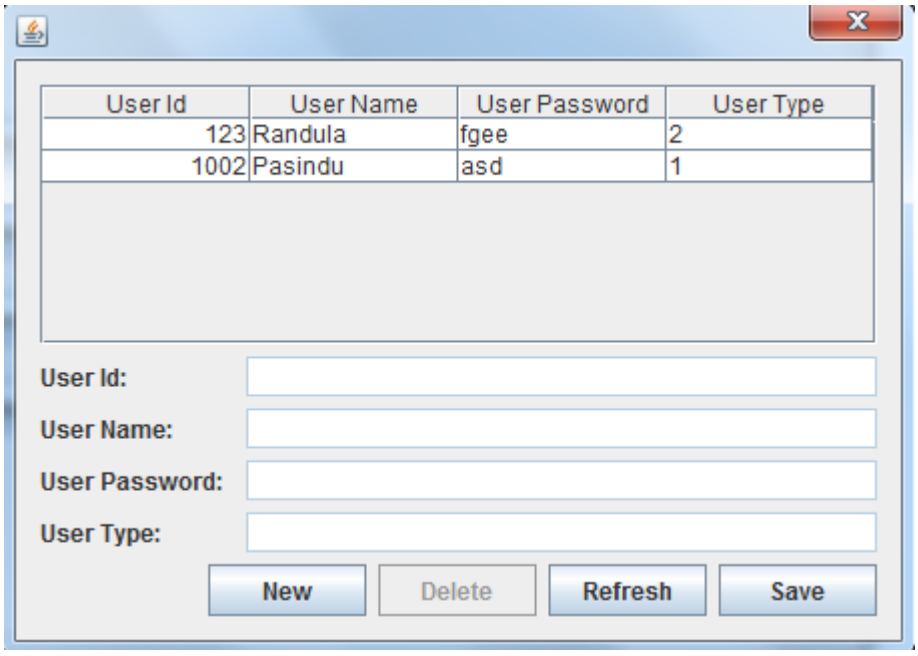
Master Detail Form

In computer user interface design, a **master-detail** page is one where a master area and its related detail area are represented on the same page. The content of the detail area is displayed based on the current record selection in the master area.

A master area can be a form, list or tree of items, and a detail area can be a form, list or tree of items typically placed either below or next to the master area.

- Using Master Detail Form to register and edit details of the patient and other user components.

The following is an example of a Master/Detail Sample Form, that was created using the NetBeans IDE for this Patient Management System. It enables the administrator to add Users(Doctors, Lab Staff) to the database as possible users of this system.



Software Overview

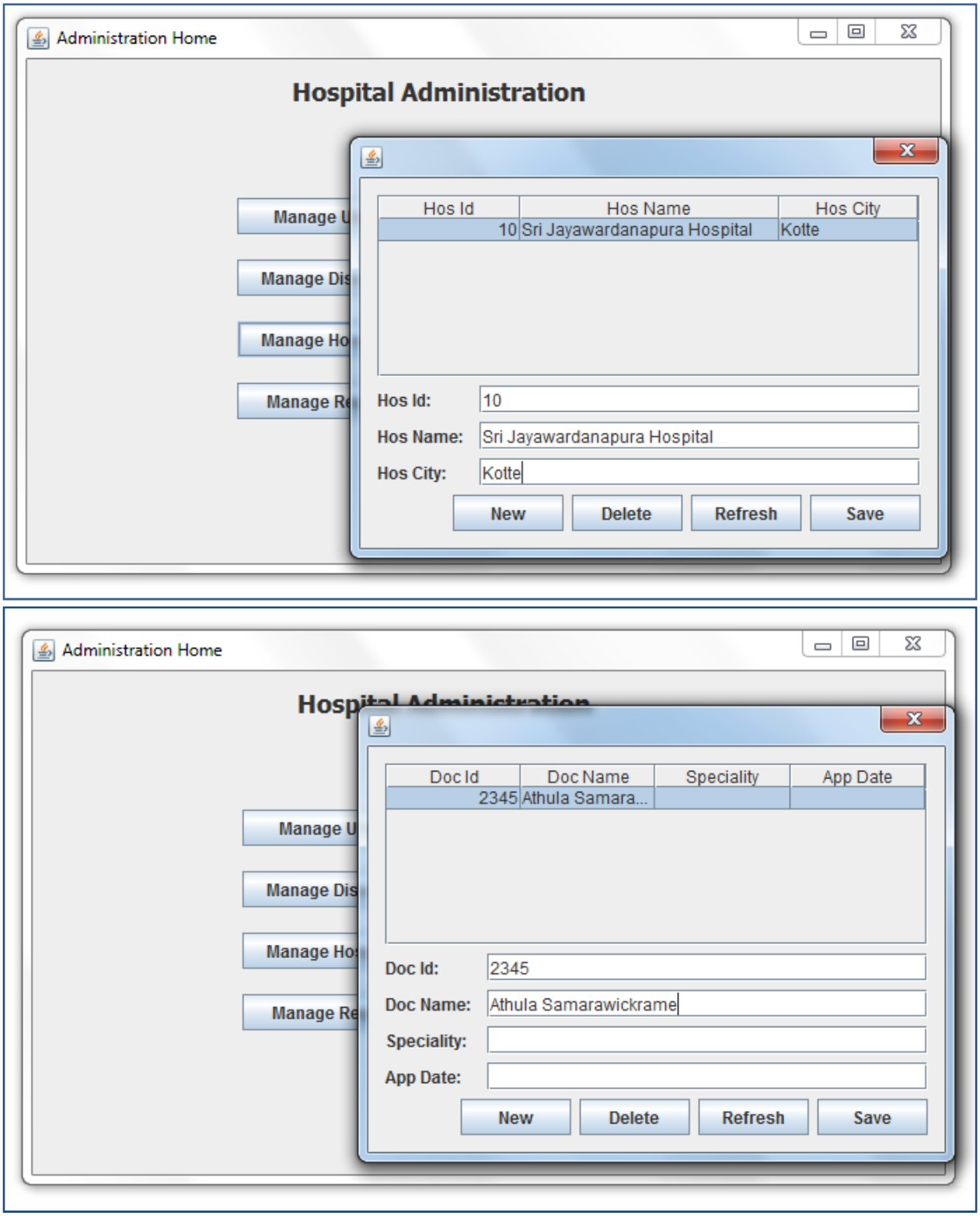
Administrator Menu

The administrator of this application has the capability to manipulate all the main data tables of the database system.

- Add/Remove/Edit Users
- Add/Remove/Edit Doctors
- Add/Remove/Edit Hospitals
- Add/Remove/Edit Patients
- Add/Remove/Edit Diseases, Drugs, Symptoms, Reports

The above mentioned Data Tables were created using the Master/Details Sample form in NetBeans IDE.

Below is a screenshot of the Administrator Menu.



The above two screenshots are a clear depiction of how the Hospital Administrator Menu works.

Doctor(User)'s Interface

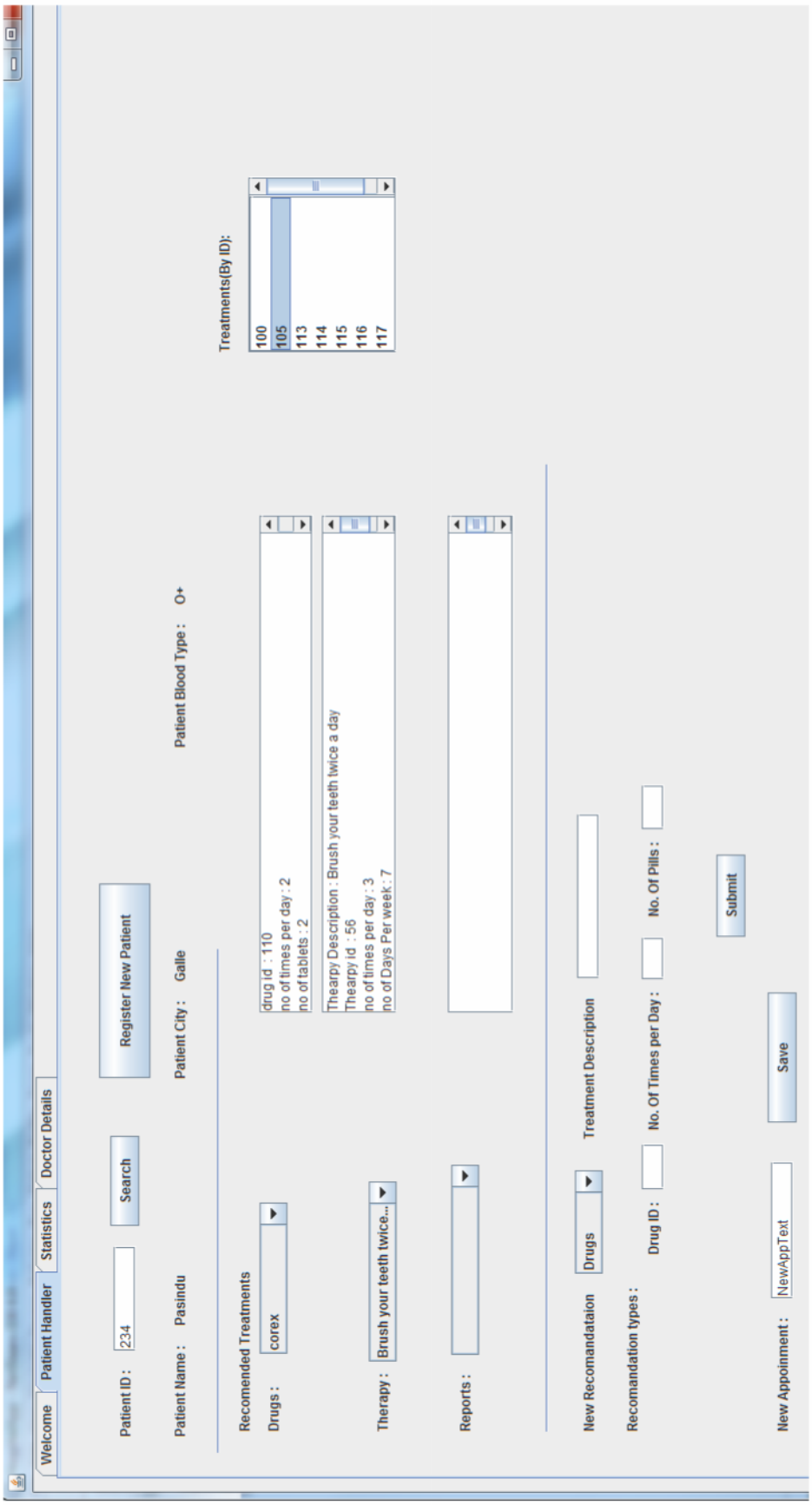
The main user interface of this is application is a tabbed pane with the following tabs.

- Welcome
- Patient Handler
- Statistics
- Doctor Details

Welcome Tab

The user is greeted with the Welcome tab when the application starts, and to move on with the rest of the procedures, he can select the above mentioned tabs. The following is a screenshot of the Welcome Screen of the Application.

Patient Handler Tab

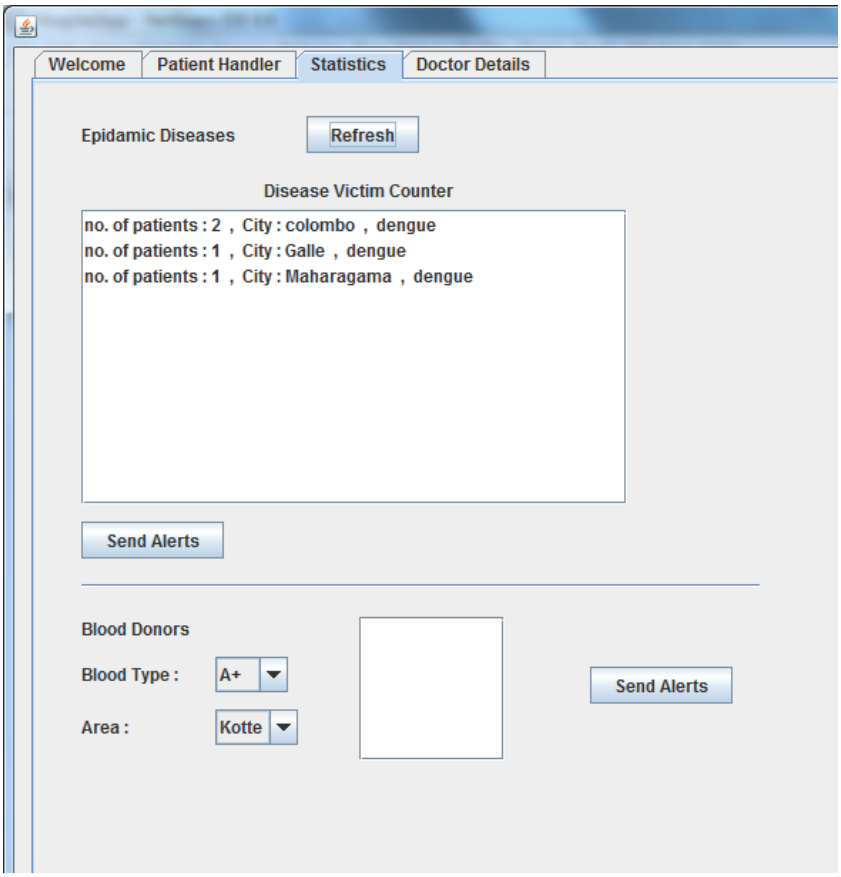


The above screen Illustrates the process of handling patients in the system. In this tab a doctor can add new Patients to the system, view the history of the patient through the Patient Search process and recommend new treatments to the patient. When doctor is in need of registering a new patient to the system the "Register New Patient " button is clicked and a new window consisting fields that are require to register a new patient to the system is appeared. Once all the detail fields are filled patient will be registered within the database.

The unique patient ID that was given when registering to the system, is used to identify the patient and for the search function. Once a patient is found(matched with the Patient ID on the Search Patient Text Field), all the treatments done are viewed in the list box under the name of "Treatments(By ID)". When a certain treatment is selected relevant details about the treatments are viewed under the categories of Drugs, Therapy and Reports. When a particular drug is selected all the details about the dosage and the drug is viewed in the relevant text area. The selection of a particular Therapy or Report combo box would result in the same way.

When a new treatment is taking place the type of the treatment(drugs/therapy/Reports) is selected and the necessary fields about the treatment will be displayed. Once the details are filled the submit button will save the information about the treatment in the database. And when again that patient is searched and once the treatments are displayed that new treatment will also be displayed under the treatments took place.

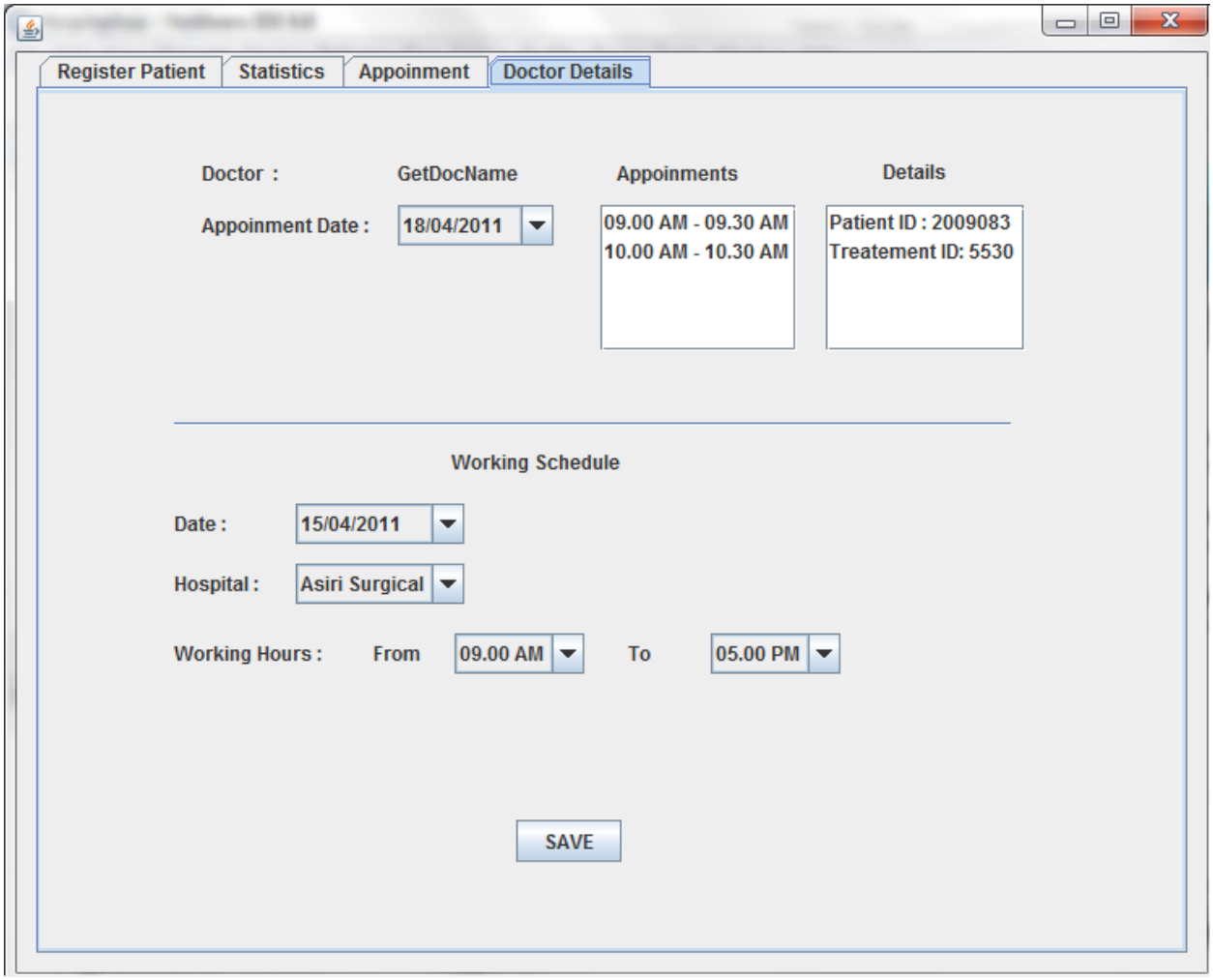
Statistics Tab



The statistics tab holds the Epidemic Disease Warning Sector and The Blood Donor Search Sector. The Epidemic Disease Warning Sector will only bring in the Patient Count for a specific Disease in a certain Area. As the above screenshot displays the, number of Dengue Patients in Colombo, Galle and Maharagama are as 2, 1 and 1. This sector's full potential is explained in the Future Enhancements Part of this report.

The blood Donor Search sector is in full under construction mode at the moment, this report was written and is also described under the Future Enhancements Part of this report.

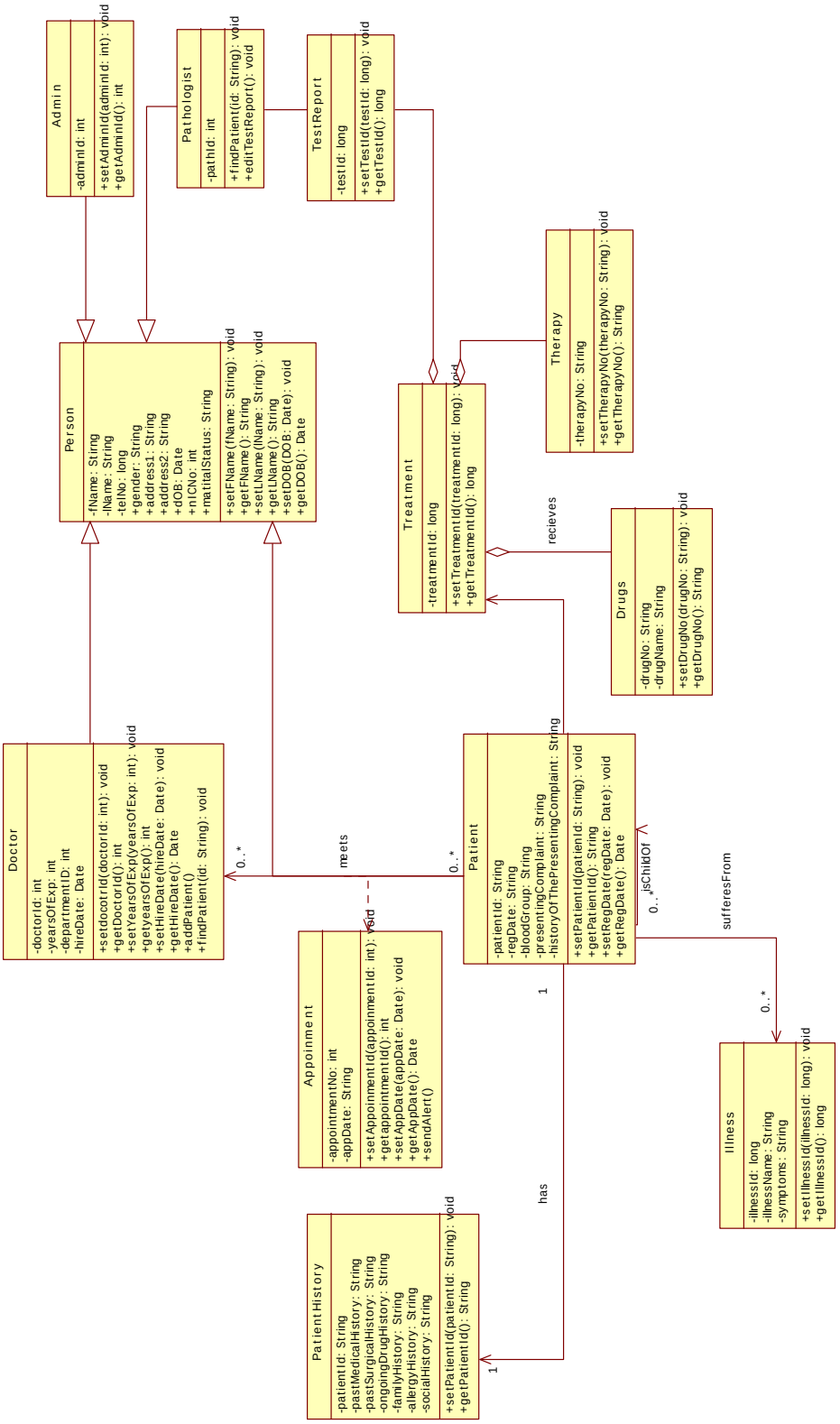
Doctor Details Tab



The full functionality of this Tab is explained in the Future Enhancements Part of the Report.

Software Designer's Note

Class Diagram



SQL Query Entries

The following query entry was used to create all the tables in the database.

```
CREATE TABLE Hospital (
hosId INT PRIMARY KEY,
hosName VARCHAR(200),
hosCity VARCHAR(200));

CREATE TABLE Doctor (
docId INT PRIMARY KEY,
docName VARCHAR(300),
speciality VARCHAR(200),
appDate DATE);

CREATE TABLE Patient (
patId INT PRIMARY KEY,
patName VARCHAR(300),
patCity VARCHAR(200),
PatBloodType VARCHAR(3));

CREATE TABLE Disease (
disId INT PRIMARY KEY,
disName VARCHAR(400),
disLevel VARCHAR(40));

CREATE TABLE Symptom (
symId INT PRIMARY KEY,
symDescription VARCHAR(600));

CREATE TABLE Therapy(
theId INT PRIMARY KEY,
timePerDay INT,
daysPerWeek INT,
theDescription VARCHAR(700));

CREATE TABLE Drug(
drugId INT PRIMARY KEY,
drugName VARCHAR(300),
drugWeight DEC(5,2));

CREATE TABLE ResultItem(
varId INT PRIMARY KEY,
normalValue VARCHAR(50),
varDescription VARCHAR(500));

CREATE TABLE Report(
repId INT PRIMARY KEY,
repName VARCHAR(400),
repStatus VARCHAR(50));

CREATE TABLE Results(
resId INT PRIMARY KEY,
reportItem INT,
repId INT,
itemResult VARCHAR(200),
```

```
FOREIGN KEY(reportItem) REFERENCES ResultItem(varId),
FOREIGN KEY(repId) REFERENCES Report(repId));

CREATE TABLE User (
userId INT PRIMARY KEY,
userName VARCHAR(300),
userPassword VARCHAR(30),
userType INT);

CREATE TABLE Treatment(
treId INT PRIMARY KEY,
treDescription VARCHAR(700));

CREATE TABLE Appoinment(
appDate DATE ,
apptime DEC(4,2),
patId INT,
docId INT,
treId INT,
PRIMARY KEY (docId,patId),
FOREIGN KEY(treId) REFERENCES Treatment(treId),
FOREIGN KEY(patId) REFERENCES Patient(patId),
FOREIGN KEY(docId) REFERENCES Doctor(docId));

CREATE TABLE PatientHasParent (
patId INT,
parId INT,
city VARCHAR(200),
PRIMARY KEY (patID,parId),
FOREIGN KEY(patId) REFERENCES Patient(patId),
FOREIGN KEY(parId) REFERENCES Patient(patId));

CREATE TABLE TreatementRecommendedForPatient(
patId INT,
treId INT,
dateRecommended DATE,
dueDate DATE,
dateDone DATE,
PRIMARY KEY (patId,treId),
FOREIGN KEY(patId) REFERENCES Patient(patId),
FOREIGN KEY(treId) REFERENCES Treatment(treId));

CREATE TABLE TherapyRecByTreatment(
treId INT,
theId INT,
PRIMARY KEY (treId,theId),
FOREIGN KEY(treId) REFERENCES Treatment(treId),
FOREIGN KEY(theId) REFERENCES Therapy(theId));

CREATE TABLE DrugRecByTreatment(
treId INT,
drugId INT,
numberOfTimesPerDay INT,
numberOfTablets INT,
PRIMARY KEY (treId,drugId),
FOREIGN KEY(treId) REFERENCES Treatment(treId),
```

```
FOREIGN KEY(drugId) REFERENCES Drug(drugId));

CREATE TABLE PatientHasDisease(
patId INT,
disId INT,
PRIMARY KEY (patId,disId),
FOREIGN KEY(patId) REFERENCES Patient(patId),
FOREIGN KEY(disId) REFERENCES Disease(disId));

CREATE TABLE DiseaseHasSymptom(
disId INT,
symId INT,
PRIMARY KEY (disId,symId),
FOREIGN KEY (disId) REFERENCES Disease(disId),
FOREIGN KEY (symId) REFERENCES Symptom(symID));


CREATE TABLE ReportForTreatment(
repId INT,
treId INT,
PRIMARY KEY (repId,treId),
FOREIGN KEY(repId) REFERENCES Report(repId),
FOREIGN KEY(treId) REFERENCES Treatment(treId));

CREATE TABLE DoctorAtHospital (
docId INT,
hosId INT,
DocCity VARCHAR(200),
PRIMARY KEY (docId,hosId),
FOREIGN KEY(docId) REFERENCES Doctor(docId),
FOREIGN KEY(hosId) REFERENCES Hospital(hosId));
```

Future Enhancements

Mobile Alert System

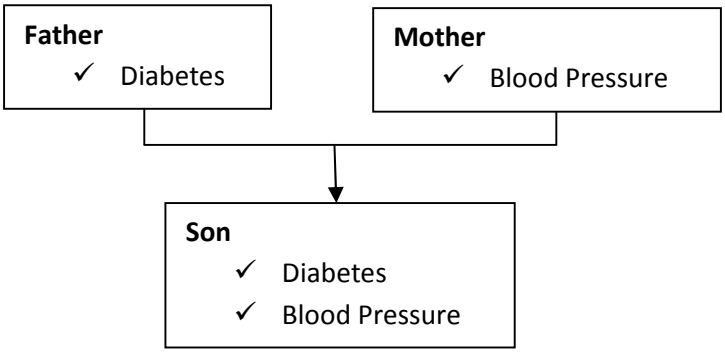
Send alerts to mobile with relevant details about Epidemic Diseases

When registering a new patient to the system, the mobile phone number can be saved as one of the personal details about the patient and with that information a whole new level of patient-hospital relationship can be developed. This system has the functionality of viewing information about diseases and the patient count of a particular disease with details about the city that this disease is spreading. So once that disease has risen up to the epidemic level with the information that has been saved in the database this system will be able to notify the people in the relevant area about the disease and save a lot of lives. This functionality can be easily developed with the help of relevant mobile phone networks and once it has been done, with such a system it is easy to recognize mass epidemic out breaks such as swine flu, dengue, malaria etc.

Hereditary Disease Detection

The patient's immediate family members (Father, Mother, Siblings) profile's will be connected with the patient's profile so that hereditary disease complications that are shown in their details will be automatically detected and is shown in patient's profile indicating that patient's presenting complaint **might have** a connection to his/her father or mother's hereditary disease.

*Important: This doesn't indicate that the patient was actually diagnosed with the disease; this functionality will be developed merely to point out the possibility of this patient having that particular hereditary disease.



Doctor Appointment Viewer

This is pane is a prototype view of what the future version of this system will look like. This tabbed pane is for storing, editing details of the doctor. Currently the functions in this pane will not execute any command but in the future more advanced functions will take place in this pane like appointment scheduler, change appointments etc.

In this pane the top part is about the appointment schedule for the coming days and the lower part consists of editing working hours of the doctor. When a particular date is selected all the appointment scheduled to that date will be displayed in the appointment box and when a necessary appointment is clicked the details like the date, time, place , with whom is the appointment etc will be displayed in the details box.

In the lower part doctor can set his/her working hours at a particular hospital. When the needed date and the hospital is set the working hours can be selected by using two dropdown combo boxes. And once all the necessary changes has been done save button will save these data inside the database.

Reference

List of References

- Biswas, R . (2006). A Simpler Programming Model for Entity Persistence.*The Java Persistence API*. 20 (2), p2-p12.

- -. (2008). *MySQL Tutorial.* Available: http://www.tizag.com/mysqlTutorial/. Last accessed 25th March 2011.

- O'Conner, J. (2007). Using the Java Persistence API in Desktop Applications. *Technical Articles and Tips* . (1), .

- Das, A. (2008). The object-oriented paradigm of data persistence.*Understanding JPA*. (1),

- -. (2010). *Java Persistence API.* Available: http://en.wikipedia.org/wiki/Java_Persistence_API. Last accessed 22 March 2011.