

LB/DON/106/2016

IT 01/125

**Utilization of Timetable Management System to IT Faculty  
at University of Moratuwa**

LIBRARY  
UNIVERSITY OF MORATUWA, SRI LANKA  
MORATUWA

Andradi D.C.S

139153U

Faculty of Information Technology

University of Moratuwa

004 "16"  
-----  
004 (043)



University of Moratuwa



TH3160

March 2016

TH 3160

-  
DVD ROM

(TH 3160 - TH 3160)

**TH3160**

**Utilization of Timetable Management System to IT Faculty  
at University of Moratuwa**

Andradi D.C.S

139153U

Dissertation submitted to the Faculty of Information Technology,  
University of Moratuwa, Sri Lanka for the partial fulfillment of the  
requirements of the Degree of Master of Science in  
Information Technology.

**March 2016**

# Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Chaya Sanjeevan: Andrad

Name of Student

Signature of Student

Date: 28.04.2016

Supervised by:

Name of Supervisor

Signature of Supervisor

Mr Saminda Premarathna

***UOM Verified Signature***

Date: 28/04/2016

## **Dedication**

This dissertation is dedicated to my beloved mother, father and husband who gave me endless courage and support to achieve my tasks whenever I was discouraged.

# Acknowledgement

My heartiest thanks should be goes to my Supervisor Mr Saminda Premarathna the guidance, assistance, encouragement and providing this opportunity of research given to me.

Also sincerely thanks to all my teachers, who taught subjects in my Msc IT degree and the things that I learnt from many subjects helped me to fulfill this hard task to be manageable one.

In addition, I would like to thank my beloved parents, my beloved husband who encouraged and helped me to success this research.

Last but not the least, my sincere thank goes to my little son and daughter for providing their valuable time.

## Abstract

University timetable construction is a laborious and complicated task when there are large number of course arrays and limited resources. Often, the timetable administrator, solve these problems of scheduling. However, the results may not always fully optimal. Every academic year, faculty of IT faces the rigorous task of preparing timetables. Although the current manually operated, timetable system is efficient enough to carry out the courses without clashes, it is very time consuming and resource optimization problems occur due to insufficient lab resources and hall facilities.

The endeavor of this work is to find out a proper near optimal solution for this highly constrained combinational timetabling problem. The efficient utilization of the resources is the main objective. After a better literature survey, it could found, by evolutionary techniques based on Darwin's theories can exploit to construct an automated timetable management system to the Information Technology faculty at University of Moratuwa. The theory named Genetic Algorithm and selected it to develop the main logic of the timetable management system. Probabilistic operators such as selection, crossover and mutation in GA, are used to plan proper timetable. Each individual is called chromosome and its validity must be evaluated using a fitness function in the implementation process. Chromosomes with higher fitness value considered as optimal solution or timetable schedules. Those optimal solutions will be further refined by manually in a lesser time.

The main stakeholders of this process will be admin of the timetable system, lecturers and the students. Interviews and observation were the main methods of data collection. PHP and Yii framework, MySQL database management system and some other software used to design and develop this timetable management system. The system was tested by using black box testing, white box testing and efficiency test. Moreover, it was evaluated using number of evaluation techniques such as interviews, observation and questionnaires. To conclude, this timetable management system automatically generates good quality timetables while optimizing the resources to the IT faculty at university of Moratuwa.

Keywords: optimize, utilize, Timetable Management System, Genetic Algorithm, fitness function

## Table of Contents

Declaration .....	I
Dedication .....	II
Acknowledgement .....	III
Abstract .....	IV
Chapter 1 .....	1
Introduction to TMSFIT .....	1
1.1 Prolegomena .....	1
1.2 Background and Motivation .....	1
1.3 Aim and Objectives .....	2
1.3.1 Aim .....	2
1.3.2 Objectives .....	2
1.4 Problem Domain .....	2
1.4.1 Research Problem .....	3
1.5 Proposed Hypothesis .....	3
1.6 The Proposed System .....	3
1.6.1 Features of the Proposed TMSFIT .....	3
1.6.2 Users, Inputs, Processes and Output of the System .....	4
1.6.2.1 Users .....	4
1.6.2.2 Input .....	4
1.6.2.3 Process .....	4
1.6.2.4 Output .....	5
1.7 Research Methodology .....	5
1.8 Research Scope .....	6
1.9 Structure of the Dissertation .....	6
1.10 Summery .....	7
Chapter 2 .....	8
Overview: Timetable Management System .....	8
2.1 Introduction .....	8
2.2 Review of The Existing Timetabling System .....	8
2.2.1 The Process of Preparing the Existing Timetables .....	8
2.2.2 Hard Constraints of the Existing Timetabling System– .....	9
2.2.3 Soft Constraints of Current Timetabling System – .....	9
2.3 Literature Review .....	10
2.4 Problem Definition .....	14
2.5 Technology Extracted from the Problem Domain .....	16

2.6	Summery .....	16
Chapter 3 .....		17
Technological Foundation of TMSFIT .....		17
3.1	Introduction.....	17
3.2	Technologies of the TMSFIT.....	17
3.3	Main Advantages of the Proposed Timetable Management System.....	17
3.4	Issues of the Proposed Timetabling System .....	18
3.5	The Genetic Algorithm Process .....	18
3.6	Application of Genetic Algorithms in This Research.....	19
3.7	Programming / Scripting Language .....	19
3.7.1	PHP Programming .....	19
3.8	Web Development Tools .....	20
3.8.1	Yii Framework.....	20
3.9	Eclipse for PHP Plugging .....	20
3.10	WAMP Server.....	20
3.11	Timetabling Engine.....	20
3.12	Database Technology .....	21
3.13	System Analysis and Design Methodology of TMSFIT.....	21
3.14	Unified Modeling Language (UML).....	21
3.15	Hardware Requirements.....	21
3.16	Creately and Diagram Designer .....	22
3.17	Summery .....	22
Chapter 4.....		23
Approach to Implement TMSFIT .....		23
4.1	Introduction.....	23
4.2	Proposed Solution .....	23
4.3	Requirements Elicitation.....	23
4.3.1	Functional Requirements of the proposed TMSFIT .....	23
4.3.2	Non Functional Requirements of the proposed TMSFIT .....	24
4.4	Current Timetable Management System.....	25
4.5	Process of the TMSFIT.....	26
4.6	Features of TMSFIT .....	26
4.7	Users of the TMSFIT .....	26
4.8	Technologies used in TMSFIT .....	26
4.8.1	Genetic Algorithm .....	26
4.8.2	How GA used in this research?.....	26



4.8.3	Hard Constraints of Proposed Solution.....	27
4.8.4	System Analysis and Design.....	27
4.8.5	Unified Modeling Language (UML).....	27
4.8.6	PHP Language .....	27
4.8.7	Eclipse for PHP Plugging .....	28
4.8.8	WAMP Server.....	28
4.8.9	MYSQL Database Management System .....	28
4.8.10	Yii framework.....	28
4.9	Software and Hardware requirements.....	28
4.10	Interface Design.....	29
4.10.1	User Interface Designing with Prototyping .....	29
4.11	Approach to Timetable Generation Process.....	29
4.12	Summery .....	30
Chapter 5.....		31
Design of the TMSFIT .....		31
5.1	Introduction.....	31
5.2	Research Planning.....	31
5.2.1	Planning the Research Project.....	31
5.2.2	System Development Methodology for TMSFIT .....	31
5.2.3	Selection of Software Process Mode for the Proposed TMSFIT .....	32
5.3	Analysis of the Existing Timetabling System.....	32
5.4	Top level Design Diagram .....	32
.....		32
5.5	MVC Architecture .....	33
5.6	First Level Module.....	33
5.6.1	Timetabling Engine.....	33
5.6.2	Database Design of Timetable Management System.....	34
5.6.2.1	ER Diagram .....	34
5.7	Second Level Module .....	35
5.7.1	How the Genetic Algorithm perform .....	35
5.8	Third Level Module .....	35
5.9	Modeling the System .....	35
5.9.1	Use Case Diagram.....	36
5.9.2	Class Diagram for the Proposed System.....	38
5.9.3	Sequence Diagram .....	39
5.10	User Interfaces Design.....	40

5.10.1	Increased Drop Down Usage:-	41
5.10.2	UI based Field Validation:-	41
5.10.3	Highlighting the Focused Field:-	42
5.11	User-friendly Error Messages	42
5.12	Summery	42
Chapter 6		43
Implementation		43
6.1	Introduction	43
6.2	Software Requirement for Implementation Process	43
6.2.1	WAMP Server	43
6.2.2	Yii Framework	44
6.2.3	Eclipse for PHP Developers	44
6.3	Hardware Requirement for Implementation Process	44
6.4	Implementation of First Level module	44
6.4.1	Interface Implementation	44
6.4.1.1	Login Interface	45
6.4.1.2	Dashboard Interface	45
6.4.1.3	Manage Degrees Interface	46
6.4.1.4	Manage Students Interface	46
6.4.1.5	Manage Lecturers Interface	46
6.4.1.6	Manage Subjects Interface	46
6.4.1.7	Manage Resources Interface	47
6.4.1.8	Manage Batches Interface	47
6.4.1.9	Generate Timetable Interface	47
6.4.2	Database Implementation	47
6.4.3	The Process of Mapping Database Tables with Model class	48
6.5	Second Level Module	49
6.5.1	Algorithm Development	49
6.5.2	Hard Constraints of TMSFIT	49
6.5.3	Initialization	50
6.5.4	Evaluation	50
6.5.5	Evaluation	51
6.5.6	Selection	52
6.5.7	Crossover	52
6.5.8	Mutation	53
6.6	Third Level Module	



6.6.1	Generate Timetable Interface.....	53
6.7	System Deployment.....	54
6.8	Summery.....	54
	Chapter 7.....	55
	How the System Works.....	55
7.1	Introduction.....	55
7.2	System Administrator's Role.....	55
7.3	Lecturer's Role of the TMSFIT.....	58
7.4	Student's Role of the TMSFIT.....	59
7.5	Summery.....	60
	Chapter 8.....	61
	Evaluation and Testing of TMSFIT.....	61
8.2	Evaluation Strategy.....	61
8.2.1	Interviews.....	61
8.2.2	Observation.....	61
8.2.3	Questionnaire.....	62
8.3	Software Testing.....	62
8.4	TMSFIT Evaluation.....	63
8.5	Summery.....	63
	Chapter 9.....	64
	Conclusion and Further Work.....	64
9.1	Introduction.....	64
9.2	Conclusion.....	64
9.3	Further Work.....	65
9.4	Summery.....	65
	References.....	66
	Appendix A.....	68
	Interfaces of the TMSFIT.....	68
	Error Messages given by the TMSFIT.....	68
	Preview of table.....	69
	Appendix B.....	70
	Code segments of the TMSFIT.....	70
	Sample code for batch form.....	70
	Calculate the fitness of the chromosome.....	71
	Cross over operation.....	75
	Mutation code segment.....	77

Dashboard code segment.....	79
Appendix C.....	84
Testing and Evaluation with Test data.....	84
Sample Test cases for Black box Testing.....	84
White Box Testing with Understand Software.....	86

# List of Figures

Figure 1-1 Milestone Approach .....	5
Figure 4-1 Flow Chart of the Current Timetabling System .....	25
Figure 5-1 Top Level Design of the Proposed System .....	32
Figure 5-2 MVC Architecture.....	33
Figure 5-3 ER Diagram.....	34
Figure 5-4 Use Case Diagram to Show the Interaction between The System and admin.....	37
Figure 5-5 Main Use Case Diagram .....	37
Figure 5-6 Class Diagram for Overall View of the System .....	39
Figure 5-7 Sequence Diagram show how the different objects interact .....	40
Figure 5-8 Drop down Usage.....	41
Figure 5-9 UI based field validation .....	42
Figure 5-10 User-friendly Error Messages .....	42
Figure 6- 1 Dashboard Interface .....	46
Figure 6-2 Tables of ttms.....	48
Figure 6- 3 Interface of _form.php .....	49
Figure 6-4 Crossover operation .....	52
Figure 6-5 Mutation operation.....	53
Figure 6-6 Timetable Generating State.....	54
Figure 6-7 Generated Timetable .....	54
Figure 7-1 Login Interface .....	55
Figure 7-2 Student Uploading Excel Sheet.....	56
Figure 7-3 Create Subject Interface .....	57
Figure 7-4 Timetable without fully optimal.....	57
Figure 7-5 Timetable Editing Window .....	58
Figure 7-6 Timetable Print Screen.....	58
Figure 7-7 Lecturer Dashboard Interface.....	59
Figure 7- 8 Student Dashboard Interface.....	60
Figure A-1 Generated Timetable Before Saved.....	68
Figure A-2 Error Message .....	68
Figure A-3 Manage Student Interface.....	69
Figure A-4 Table Preview.....	69
Figure C-1 White box Testing .....	86

## **List of Tables**

Table 2-1 limitations of timetabling problem .....	15
Table 4-2 Approach to Timetable Generation .....	29
Table 5-1 Gantt Chart .....	31
Table 6- 1 Chromosome Representation.....	51
Table C-1 Test Cases .....	86
Table C-2 Questionnaire Evaluation Table.....	89

## **Abbreviations**

TMSFIT – Timetable Management System of Faculty of IT

GA – Genetic Algorithm

## Introduction to TMSFIT

### 1.1 Prolegomena

E. Burke and coworkers say “Timetabling manually is a complex activity that is time-consuming and stressful because a large number of high and soft constraints need to be satisfied; there is therefore the need to automate the timetabling process” [1]. A well-planned and clash-free timetable is a censorious element in running a university or any other academic environment. Previous days when technology was not in wide use, the timetable administrator manually created academic timetables. Thus, the task was very time consuming, University identified necessity of an automated timetable management system. This research directed because of that real importance of the University.

### 1.2 Background and Motivation

According to Cambridge English Dictionary, timetable is “a detailed plan showing when events or activities will happen” [2]. Constructing timetables and maintaining them is often a very complex task for software as well as people[3]. However, today, timetabling process has been simplified by fully automatic or semi-automatic solutions based on timetable generation applications such as Mimosa, TimeTabler and Open Course Timetabler.

Every academic year, faculty of IT face the rigorous task of preparing timetables that satisfies the various courses and their respective examinations being offered by the three different departments which are Department of Information Technology, Department of Computational Mathematics, Department of Interdisciplinary Studies.

Current timetable management system with graph coloring heuristic technique is efficient enough to carry out the courses without clashes manually. Nevertheless, problems occur due to insufficient lab resources and hall facilities. The problem is more complex when some batches have more than three hundred students while the largest hall can be allocated only two hundred and twenty two students.

Motivation to this research is due to the necessity of an automated web based Timetabling System to faculty of Information Technology at the University of Moratuwa which is designed and developed to optimum utilization of the resources and lectures offered by the University.

### **1.3 Aim and Objectives**

#### **1.3.1 Aim**

The major purpose of this work is to develop a web based Timetable Management System to optimize the resources of IT faculty at University of Moratuwa.

#### **1.3.2 Objectives**

Following are the objectives of the research to achieve the above mention aim.

- Investigate the available lab capacity and required resources.
- To study number of scheduling algorithms
- Conduct a comparative study of Genetic Algorithm used in the timetabling problems to develop a timetable management system using the most suitable Genetic Algorithm.
- Study the development technologies for the automated timetabling.
- Build a prototype to gather and evaluate the user requirements
- Develop the automated web based timetable management system.

### **1.4 Problem Domain**

Even the currently available system can generate timetables; still have issues with constructing a clash-free, optimal and complete timetable. The tedious tasks of data introduction and revision of usually incomplete solutions are the bottleneck in these cases [4]. Normally, educational institutions such as the University of Moratuwa have utilized to manual generation of their timetables. To get completed and optimal solution it takes long time. Even at the optimal stage of the manually generated timetable, there are still a few clashes available Sometimes timetable admin has to ask students to change their preference to avoid the clash. It makes them uncomfortable.



### **1.4.1 Research Problem**

Can an Automated Timetable Management System use to resources optimization of IT faculty?

### **1.5 Proposed Hypothesis**

Formulate an automated Timetable Management System to the faculty by using Genetic Algorithm as a technique to solve the problem, as it is able to produce a feasible timetable and fulfill as many constraints imposed.

### **1.6 The Proposed System**

TMSFIT is an abbreviation for Timetable Management System in Faculty of IT. This new system will provide the facilities for the hall reservation information on the availability of the halls laboratories in the admins module. Lectures and students must register through the TMSFIT before they start using the system. Hence, the security is very high, only admin TMSFIT can update the timetable. There will be an authenticating using the users passwords. The students of the other faculties cannot allow accessing the system.

The proposed system will use to generate timetable automatically. This ensures the following features

- Easier slot assigning
- Less time consumption
- Minimum slot clashes
- User friendly

#### **1.6.1 Features of the Proposed TMSFIT**

A website built in order to improve the current system. The proposed system is suggest be called as Timetable Management System Faculty of Information Technology (TMSFIT). Features of the new system are outlined as follows. Some features of the existing system are improved and some are very significant to the proposed system.

- This proposed system provides an attractive graphical front-end and it is the main interaction point with user.

- The system also improves the flexibility of timetable construction.
- It will be able to generate printouts on timetabling.
- Upgraded versions of the timetable management system must be introduced
- To increase the optimization, generated timetables can be fine-tuned
- The system should save the time.
- Productivity is improved.

## **1.6.2 Users, Inputs, Processes and Output of the System**

### **1.6.2.1 Users**

- Timetable administrator
- Lecturer
- Student

### **1.6.2.2 Input**

- The system is able to take number of inputs from the user (Admin TMSFIT) such as Student list, Lecture list, Course list, Semester list, Hall list, Laboratory list and Timeslot as well as various and constraints such as lecturer preferred time using web based forms.
- User levels are defined and different kind of user interfaces are produced

### **1.6.2.3 Process**

- Design and develop various kind of user interfaces according to the user access level such as view admin profile, view lecturer profile, view student profile
- Develop methods to generate timetables automatically
- Design and develop a method to view timetables, view list of students, view allocated resources
- Design and develop methods to insert, update, delete data from the ttms database
- Develop methods to create student, create lecturer, create batch, create degree, upload students, student enrolment and lecturer assignment for the subjects.
- Design and develop a user Interface to accumulate lecturer preferred time

#### 1.6.2.4 Output

- Display the generated timetable for a specific semester.
- Printable timetables
- Web based system will show the availability of the resources such as labs and courses.

#### 1.6.2.5 Technology

- TMSFIT web based application (PHP 5.6)
- The language implementing the TMSFIT is PHP and MYSQL
- The framework used to implement the system is Yii with MVC architecture

### 1.7 Research Methodology

Milestone approach from Professor Karunananda, was used as the main research methodology in this work. Figure 1-1 will illustrate that Milestone Approach.

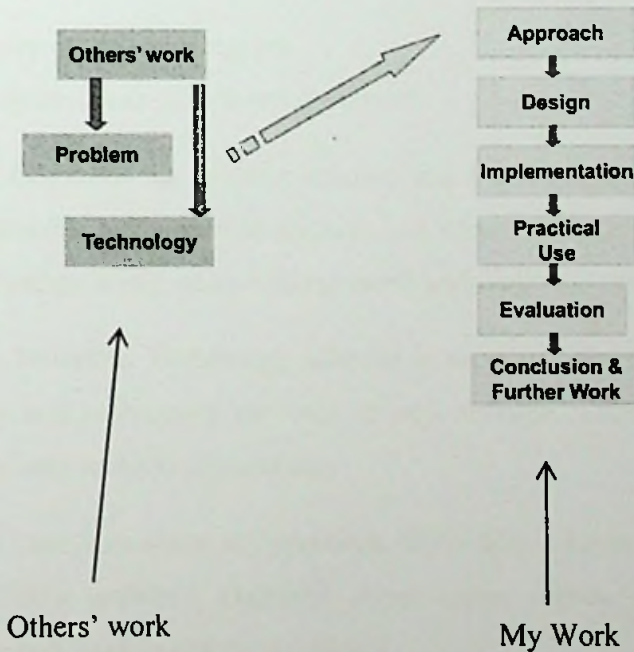


Figure 1-1 Milestone Approach

In requirement elicitation phase, other than the interviews as said some more information about subjects offering by each department, credit system and semester information were gathered from student handbook [5] and the faculty website.

Then, a literature survey was conducted to seek the best approach to develop the new system.

PHP language was used to develop the system. Commonly, it is combined with databases written in the SQL language.

The output was a prototype of the TMSFIT.

## **1.8 Research Scope**

- Used only requirements of IT faculty
- Used two main degree programs offered by the IT faculty
- Used only the information of under graduates
- Used only the lecturers information of the IT faculty
- Took the details of the IT faculty resources

## **1.9 Structure of the Dissertation**

This dissertation contains following chapters.

Chapter 2: Describes the problem domain and it will discuss about the issues of the current timetable management system and some other approaches for solve the problem. Further, it will address the strength and weakness of those approaches.

Chapter 3: Describes Technology adapted in proposed TMSFIT. Provide how these technology and techniques are used in my research. Literature review of others approaches with refers to my problem.

Chapter 4: Describes about my approach. Show how I adopt the technology to solve the timetabling problem. Describe about users, inputs, outputs, processes and technology that implements the solution.

Chapter 5: Describes system analysis and design of the proposed system with diagrams. Describes about each module

Chapter 6: Describes the implementation of the TMSFIT. Implementation details of each module. It state about, software, hardware, flowcharts, algorithms, pseudo codes, code segments as per each module in the design

Chapter 7: Describes how the system works. It will act as a user guide for this system.

Chapter 8: Describes about the discussion of testing and evaluation of the system.

Chapter 9: Describes conclusion and further work of the system. Further, it will be discussed if this research achieved its goal and objectives.

## **1.10 Summery**

This chapter describes the brief and overall description of the project. It contains sub sections as background and motivation, aim and objectives, problem domain and some more.

Next chapter will discuss about review of the others work. It gives a full description about background information of the project. Based on a literature survey, statements about others' approaches to solve similar problems and highlight my problem. It has provided a table for comparison of different approaches.

# Overview: Timetable Management System

## 2.1 Introduction

In the previous chapter, an introduction to the proposed system, motivation, aim and objectives are described precisely. Further, there was brief introduction of problem definition and a brief introduction of users, inputs, outputs, and software hardware requirements. This chapter will further describe the problem domain and regarding the issues of the current timetable management system and some other approaches for solve the problem and the strength and weakness of those approaches. This will also describe benefits of implementing automated TMSFIT. Further, a comparison of the propose system with other systems available and justifications for why the propose system is selected to be developed.

## 2.2 Review of The Existing Timetabling System

University timetabling in this context refers to the rigorous task admin of the timetable in a Moratuwa university undergo to draw up timetables that satisfies various courses that should compulsorily be inherent in the final timetable solution.

The lecturers in each department wish to specify preferred time on their courses. All the courses and course details must be given to admin of the timetable of university who is having the responsibility of creating a near optimal timetables, which would serve as a guide for academic activities in the university.

The traditional manual timetabling system is very time-consuming and resource-intensive. Existing timetabling process contains many steps and requires re-processing the same data sin number of times.

### 2.2.1 The Process of Preparing the Existing Timetables

- The timetable preparation is usually done by the admin of the timetable management system.
- To prepare timetable, the senate must approve the courses offered by the departments.

- Then the admin of the timetable demands for meeting to preparing timetable with the staff. All the lecturers who are going to teach next batch must be participated that meeting.
- The admin of the timetable brings timetable sheet without filling as Appendix C. The timetable part as will be filled according to the lecturers' requirements of the time slots.
- Lecture halls are allocated according to the number of students in a batch.
- There are mainly seven lecturer halls and six labs with the hardware lab.
- Resources or labs allocated according to the needed credits of a subject and number of students doing that subject in a particular batch
- A color-coded timetable system uses
- All the halls and labs assign a particular color.
- Admin manually assigns those halls and labs to desired time slots.
- Finally, sends the prepared timetables to the lecturers and students as their requirement.

### **2.2.2 Hard Constraints of the Existing Timetabling System–**

These are the constraints, which cannot be violated

- ✓ No lecturer can be assigned to more than one class at a same time.
- ✓ No student can be assigned to more than one class at a same time.
- ✓ For each course, there should be required number of periods.
- ✓ Each subject must have some required consecutive periods.
- ✓ A lesson cannot take place in a smaller room than required.
- ✓ One lecture hall or lab cannot be used for two lessons for the same time.
- ✓ Specific range of days by week should be in the limit. Eg: Can't teach in weekends.
- ✓ A schedule cannot exceed a specific range of hours by day.

### **2.2.3 Soft Constraints of Current Timetabling System –**

These are the constraints, which increase the cost. These can be violated.

- ✓ Same lecturers must not have consecutive periods unless specified.
- ✓ Before a certain hour and after an hour, a lesson should not take place.
- ✓ A class can be having a number of free days during the week.
- ✓ A class cannot exceed maximum number of hours.
- ✓ While lecturing a class should have some breaks.

## 2.3 Literature Review

Nelishia Pillay says even though there are number of researches found in timetabling few of them only developed as software [6]. Further, this domain has been discussed isolatable and problem is varying from one school to another. Comparative studies of several methodologies, which can be applied in this area, also have not conducted as researches yet. The paper provides an overview of methodologies such as Bee algorithm, Constraint programming, Cyclic transfers, Evolutionary algorithms, Integer programming, Neural networks, Simulated annealing and so on employed to solve the school timetabling problem and details of publicly available school timetabling data sets. Further, some details of timetabling and further research areas in future also discussed. Although it does not highlight the most suitable techniques for solving the timetable generation issue.

In a similar study, Abdelaziz and his coworkers conducted a research on Course Timetabling Problem at an institution in a Tunisian University[7]. They introduce a heuristic procedure called Timetabling Heuristic Approach to construct a feasible timetable for all lectures and tutorials taken by different groups of each subsection of any section. Then, they described the timetabling problem using a list of all specific hard and soft constraints and formulated the problem as a set of linear constraints using two sets of binary variables corresponding to lectures and tutorials, which illustrated with real data. Finally, they could compare the results with manually generated timetables and it was comparatively fair. As further work, they mention tabu search is better to minimize the number of soft constraints violations such as reduce the number of holes and isolated activities.

In addition, Hadrien and coworkers, said many approaches of timetabling based on local search or constraints programming[8]. They have introduced a practical application of an explanation based system to solve a school timetabling problem with the technology of constraints programming. For that, they presented a different local search technique that tries to take advantages both of constraint programming for satisfying hard constraints and local search for its performance in an optimization context. However, the problem must be understand further before it applying to real world situations and further experiments have to be done due to optimization of the resources [8].



Moreover, Salman and his team have addressed another same type of question. They have presented a new tabu search heuristic for high school timetabling problem [9]. To consider characteristics of different schools, the algorithm has been changed several times. The experiments shown that, this algorithm can build near optimal timetables acceptable by school's staff but, the proposed algorithm does not independent on data structures commonly used in timetabling research like matrix of teachers or classes over time period. Their solution is to optimized using tabu search algorithm based on frequency-based diversification. The algorithm could not use in other timetabling problems like universities and it should use real data.

In a different study, E.K Bruke and colleagues could discuss the scheduling problem under recent research directions in automated timetabling and introduced some resent approaches [10]. According to that as the first step, two evolutionary algorithms were discussed. Secondly, multi criteria decision problem was presented for the time tabling problem. Thirdly, case base reasoning approach used the foundation of employee experience. From the research, they investigated some issues, as methods for the problem solution must be identified before in hand and some "Knowledge poor" algorithms for scheduling problem.

Yet another research, E.K.Burke and colleagues presented a solution for constructing an automated system using graph coloring and room allocation algorithms and show how the two can be combined together to provide the basis of a flexible and widely applicable timetabling system [11]. The main issue of that approach is it cannot guarantee an optimal solution but it can guarantee a solution the user is happy with it.

Further, Burke and coworkers could indicate, researchers have addressed the timetabling problem using a universal algorithm called Synthesis-Algo for solving complex and highly constrained timetabling problems by hybridizing concepts from evolutionary algorithms and tabu searching technique [12]. Unlike manual timetable generation that usually takes months to produce, this automating the timetabling generation process has been reduced to as little as 4 to 6 hours and the approach is able to generate suitable solutions for problems from dissimilar domains. This can be further improved by applying the enhanced genetic and mimetic algorithm and improved tabu searching technique[1].

Yet another heuristic approach for the timetabling problem introduced by Alberto and coworkers, as genetic algorithm[13]. In their paper, they have presented a model, a class of algorithms and a computing program for the timetable problem, with special reference to a real world application (the timetable of an Italian high school). Further, they have compared that GA-based approach with various versions of simulated annealing and tabu search. Finally, they conclude their experiments as GAs produced better timetables than simulated annealing, but slightly worse timetables than tabu search. An advantage of GAs over both SA and TS is that GAs gives the user the flexibility of choosing within a set of different timetables. Finally, they were identified their approach is a useful generalization of the GA and can be applied to other highly constrained combinatorial optimization problems.

In a different case, according to Chiu-Hung Chen and team workers, GA can be used in several other applications also like Manufacturing Robots such as niche genetic algorithm (GA) based on a novel Twin space Crowding (TC) approach is proposed for solving multimodal manufacturing optimization problems[14].

However, Moreira could find a reasonable answer to the difficulty due to be great complexity of the construction of timetables for exams [15]. They present a method of solution to the problem of automatic construction timetables for the exams. Among several mathematical models of representation, the final option was for a model matrix, which is justified by the benefits that this model presents when used in the algorithm of solution. The method of solution is a meta-heuristics that includes a genetic algorithm. The main issue is the results achieved in all tests performed with real scenarios, in general, are satisfactory only.

In a similar scenario, Omar and coworkers could introduce same solution. He said the increasing number of courses and students in universities has led to the need for effective automatic scheduling of examination timetables. They presented a timetable with binary representation is applied with several operators with the aim of preventing the violation of the fundamental constraints [16]. The algorithm is guaranteed to always produce a feasible solution by satisfying the hard constraints. It utilizes one-point and two-point crossover operators and propagates distinctive timetable features to generate better solutions even for complex cases. They have used open source software. Finally, they understood, for future work, other GA's

techniques and operators are used as multi-point crossover, variable and direct mutation, selection schemes like Random walk, Neighborhood, or Rank will increase the performance and the efficiency of the algorithm.

Yet another different problem, Cantú-Paz and the colleagues say Genetic Algorithms are easy to apply to a wide range of problems and GA tends to be more accurate but inefficient with respect to computation time [17]. In their research, as Travelling Salesman they found the usability of GA when require more accurate and optimize solution. MoHPBGA presented in this paper is a genetic algorithm designed for hierarchical classification for multi objective problems. Further, proposed method executes the instances of GA on the clusters and uses the complete population for optimization. Although they have not discussed how Multi objective Hierarchical Population Balanced Genetic Algorithm (MoHPBGA) apply for the scheduling problem.

In a similar problem with different approach, Joe Henry and colleagues wanted to tackle the course timetabling problem with multi agent approach [19]. The proposed design seeks to deal with the problem using a distributed solution environment in which a mediator agent coordinates various timetabling agents that cooperate to improve a common global solution.

Furthermore, Branimir Sigl and the coworkers also described in their paper use of genetic algorithm for solving timetable scheduling problem [18]. The algorithm was tested on small and large instances of the problem. Algorithm performance was significantly enhanced with modification of basic genetic operators, which restrain the creation of new conflicts in the individual. They described how set of active rules can be used to express the knowledge of intelligent and how a genetic algorithm can be used to dynamically prioritize rules in the face of dynamically evolving environments. In extreme cases where there is only one good solution, the genetic algorithm may fail. From their point of view, to completely solve the full-scale problem, further algorithms improvements must be done.

In view of Professor Ashoka Karunanada, applications of GA are miracles in new technology[19]. In book of Artificial Intelligence, he has mentioned, when there is a necessity of some optimal solution such as timetabling, GA is applicable. Further, some data mining issues without having any solution and lottery games with

probabilistic theory also use this algorithm. The major disadvantage of the GA is when the population is large the algorithm execution time also increasing.

Creating and maintaining timetables is often a complex task for both people and software. When consider a Mimosa like commercial application, the technical side of Mimosa is kept as simple and as self-contained as possible. The technology is based on a collection of efficient optimization algorithms[3]. Moreover, some other semi-automatic timetabling software such as TimeTabler[20] and Open Course Timetabler[21] are also stand-alone applications. While Timetabler is commercial software, Open Course TimeTabler is a free software.

## 2.4 Problem Definition

The above study shows the numerous limitations of the timetabling systems and summaries in the table as Table 1 -1 Limitations of Timetabling.

Research	Limitations
A survey of school timetabling research by Nelishia Pillay	This is only a survey and it doesn't mention best solution for timetabling issue
A research on Course Timetabling Problem at an institution in a Tunisian University by Abdelaziz and his coworkers	Used heuristic procedure to construct a feasible timetable not give optimal solution
Interactively solving school timetabling problems using extensions of constraint programming by Hadrien and coworkers	The problem must be understand well before it applying to real world situations
Recent research directions in automated timetabling by E.K Burke	Methods for the problem must be identified before its uses.
Graph coloring techniques in timetabling by E.K Bruke	The main issue of that approach is it cannot guarantee an optimal solution
A Tabu Search Algorithm With Efficient Diversification Strategy for High School	The algorithm could not use in other timetabling problems like universities

Timetabling Problem by Salman and his colleagues.	and it should use real data.
Manufacturing Robots by Chiu-Hung	GA used with Robotics
A System of Automatic Construction of Exam Timetable Using Genetic Algorithms by Moreira	The main issue is the results achieved in all tests performed with real scenarios, in general, are satisfactory only.
Comparing Performance of Genetic Algorithm with Varying Crossover in Solving Examination Timetabling Problem	For performance efficiency, GA's techniques and operators should use as multi-point crossover, variable and direct mutation and selection schemes like Random-walk
A genetic algorithm to solve the timetable problem by Alberto and coworkers	They identified their approach is a useful generalization of the GA and can be applied to other highly constrained combinatorial optimization problems.
MoHPBGA: Multi-objective Hierarchical Population Balanced Genetic Algorithm using MapReduce by Cantú-Paz and coworkers	MoHPBGA did not use in scheduling problems.
Artificial Intelligence by professor Ashoka. S. Karunananda	When the initial population is large, the algorithm execution time also increasing.
Mimosa Timetabling Software	It does not have an import mechanism to import details from other sources and it is a commercial, stand-alone application.

Table 2-1 limitations of timetabling problem

Although there has been fair amount of researches about timetabling, few of them only considered the issues of school timetabling and university timetabling. Furthermore, there have been no comparative studies on the success of different methodologies on timetabling problems. The complexity of the timetabling is another issue. Manual scheduling is a tedious work and it generally takes number of weeks to generate timetables is also a problem. Even today, there are many semi-automatic applications developed such as Mimoso, Time Tabler, still do not solve the whole problem. The increasing number of students and the courses of universities also should take as an issue of timetabling. Another problem occurs due to variation of constraints from one institution to another.

## **2.5 Technology Extracted from the Problem Domain**

Concerning the Literature Survey, Genetic Algorithm is selected to implement the timetabling problem of Faculty of IT at the University of Moratuwa.

Some open source software will be used in TMSFIT while most of the products used by Moratuwa University are free software.

Other technologies: Apache web server, MySql Database Management System with PHP

## **2.6 Summery**

During this chapter described the problem domain and it discussed about the issues of the current timetable management system and some other approaches for solve the problem. Further, it addressed the strength and weakness of those approaches. Next chapter will be discussed the technology to be use in the proposed system. In conclusion, through the literature review of this chapter, problems of the timetable management systems could be properly identified.

# Technological Foundation of TMSFIT

### 3.1 Introduction

A descriptive literature review has conducted through the previous chapter. During that study, strengths and weaknesses of the technologies used by the other researches properly found. This chapter describes the technology and theories in order to solve the problem. Further, it describes reviews of technologies and justification of using them in this research.

### 3.2 Technologies of the TMSFIT

Free Open source software products and technologies will be mainly used in proposed TMSFIT.

- Genetic Algorithm
- Programming Language – PHP
- Timetabling Engine
- Server - WAMP
- Database Management System – MYSQL
- Yii Framework
- Designing Tools – Rational Rose and MS Diagram Designer
- Testing tool – Understand tool

Below sections gives the small description about the technologies used to implements the system and reasons to select those technologies.

### 3.3 Main Advantages of the Proposed Timetable Management System

- The system is more flexible than the previous manual
- Minimal processing/computing power is enough to utilize it.
- It's efficient than the manual one.
- Its interfaces are attractive and user friendly.
- The scheduling process is very simple and increases the productivity.
- Timetables generated are between to 80%-90% optimum.
- It almost eliminates paperwork.

### 3.4 Issues of the Proposed Timetabling System

The following are the major challenges

- The timetables are generated based on some hard constraints
- Because of the probabilistic theory of GA, it always gives near optimal solution rather than a fully optimal solution.
- To get a fully success solution, some manual work also have to be done.
- The execution time of GA itself is high.

### 3.5 The Genetic Algorithm Process

- The Initialization Process - An initial population must be created. It should be randomly generated and can be any desired size, from only a few individuals to thousands. In this work, it is hundred chromosomes.
- The Evaluation Process - Then, each member of the population is evaluated and 'fitness' for that individual must be calculated. The fitness value is calculated by how well it fits with our desired requirements. In this solution main hard constraints are the anticipated requirements.
- The Selection Process - We want to be constantly improving our populations overall fitness. Selection helps us to do this by discarding the bad designs and only keeping the best individuals in the population. There are a few different selection methods but the basic idea is the same, make it more likely that fitter individuals will be selected for our next generation. Therefore, a selection process includes in this TMSFIT also. It always keeps the best five chromosomes in the flag and if better chromosome found through the crossover and mutation process, it will be override to one of the previous solutions.
- The Crossover Process - During this crossover, we create new individuals by combining aspects of our selected individuals. It just likes imitating how sex works in nature. The hope is that by combining certain traits from two or more individuals we will create an even 'fitter' offspring which will inherit the best traits from each of its parents. In this solution, by using crossover (\$parent2) function, make new offspring by combining its parent codes. There crossover probability is 80% and there are two crossover points.
- The Mutation Process - We need to add a little bit randomness into our populations' genetics otherwise, every combination of solutions we can create



would be in our initial population. Mutation naturally works by making very small changes at random to an individual's genome. The new child may have completely different features from its parents. If the new chromosome is better than previous ones, that new one will be replaced and go to the selected best five flags.

- Repeat the above Processes - Now we have our next generation we can start again from step two until we reach a termination condition. These processes are repeating with in 500 while loops. As a result, more than 80% optimal solution will be produced.

### **3.6 Application of Genetic Algorithms in This Research**

Having considered the basis for a genetic algorithm, the outline below highlights the applications of the proposed system in generating timetables.

The basic technology of the timetable problem is the attempt of the genetic algorithm to optimize a function over a discrete structure with many independent variables. Even though the timetabling problem is treated as an optimization problem, there is actually no fixed objective function. Therefore, GA can be used construction of course timetables developed for the University of Moratuwa. The concept though developed for course timetabling, can be adapted to fit the construction of examination timetables.

The genetic algorithm employed combines two heuristic algorithms, the first finding a non-conflicting set of courses and the second assigning the selected course to halls and labs. The process is repeated until 500 loops and all courses have been scheduled with minimum conflicts. Like every other genetic algorithms, this algorithm can quickly produce large populations of random feasible course timetables. Uniquely, the process takes each subject of the batch population and assigns it to the hall or a lab. The mutation and crossover procedures will then be applied to the population so that constraints associated with each course in the assignment are satisfied.

### **3.7 Programming / Scripting Language**

#### **3.7.1 PHP Programming**

PHP stands for Hypertext Preprocessor. PHP can be interspersed within Hypertext Markup Language (HTML) that makes developing dynamic websites more reachable.

It uses a server side, cross-platform technology. Server-side actually refers to the fact that everything PHP does occurs on the server instead of the client's site. PHP runs on any operating system such as Windows, UNIX or Macintosh. Further, PHP is better, faster and easier to learn than the alternatives when developing dynamic websites. As required by the user the system has to be set up on a server. Since it was open source, PHP was selected to develop this web-based system at University of Moratuwa.

### **3.8 Web Development Tools**

#### **3.8.1 Yii Framework**

- Yii finest for developing both web applications and APIs[22] and it is high performance new PHP framework.
- Yii works to streamline your application development and helps to ensure an extremely efficient, extensible, and maintainable product [22].
- Most of the products come under open source category at the University of Moratuwa. As a result, Yii also selected to develop this Timetable Management System. Further, it needs Apache web server that supports PHP 5.1.0

#### **3.9 Eclipse for PHP Plugging**

In order to eclipse is written in java its need to make sure java installer has been installed to the machine. It is a professional Integrated Development Environment (IDE) for developing PHP programmers and apps. PHP User Model that provides an API for navigating and PHP Source Code Formatter this eclipse for PHP plugging was successfully used in this research to modify the code.

#### **3.10 WAMP Server**

WAMP stands for Windows, Apache, MYSQL and PHP. WAMP is an open source software and no cost obtaining it. Therefore, it was selected as the local host for this web based system.

#### **3.11 Timetabling Engine**

The timetabling engine is primarily a web server which connects the database. Apache web server is considered as the Timetabling Engine in this system.

### **3.12 Database Technology**

MySQL is a database management system (DBMS) for relational databases. MySQL was selected to develop the database for this web-based system because, like PHP, MySQL offers excellent performance, portability and reliability, with moderate learning curve at little to no cost because MySQL is the world's most popular open source database. Further, PHP always gives good support for MySQL.

### **3.13 System Analysis and Design Methodology of TMSFIT**

Object Oriented Analysis and Design (OOAD) approach selected to use in the project because of the following benefits.

- It increases robustness of the system.
- It is capable for tackling number of challenging problem domains.
- It provides better quality communication among users, analysts, designers and programmers.
- It has Increased consistency among analysis, design and programming activities
- It increases reusability of analysis and designing

### **3.14 Unified Modeling Language (UML)**

Because of the following benefits, UML was selected to graphically represent the proposed TMSFIT.

- Usually, it offers regulations for software development.
- Reduce cost for diagram development.
- Lesser time for development.

### **3.15 Hardware Requirements**

Following is the list of minimum hardware requirements for a personal computer to develop the TMSFIT:-

- Computer with 2GB RAM
- 2GHz or more processing power
- 500GB Hard-disk
- 15" Monitor
- Modem
- Keyboard and Mouse

### **3.16 Creately and Diagram Designer**

Used creately online diagram designing tool Diagram Designer tool to draw the designs.

### **3.17 Summery**

This described about the technology this research adapt to solve the problem and how these techniques are appropriate to solve this timetabling problem. These techniques were highlighted with regards to the issues of the timetabling system. Next chapter will discuss the approach to the timetabling solution.



# Approach to Implement TMSFIT

### 4.1 Introduction

Using the technology mentioned in the previous chapter, we provide an approach to solve the problem as follows. This chapter will discuss the proposed solution (hypothesis), functional requirements (Inputs), nonfunctional requirements (outputs), process, features, the users, and the technology that implements the solution.

### 4.2 Proposed Solution

Formulate an automated Timetable Management System to the faculty by using Genetic Algorithm as a technique to solve the problem, as it is able to produce a feasible timetable and fulfill as many constraints imposed.

### 4.3 Requirements Elicitation

Interviews and observation are the main methods of data collection through this research. Requirement elicitation was done with the users of the current system. The main user of the timetabling system is the Timetable administrator. A proper literature survey of previous timetable systems and their algorithms were required to define problem domain.

Then, tried out Number of trial versions of generic products such as TimeTabler, Mimosa scheduling software, OCTT, EduSwift, eTimetable were installed to a computer to analyze their development and the implementation. After that, let the admins to check the features of those generic products and according to their feedback prepared a model how it would be in future.

#### 4.3.1 Functional Requirements of the proposed TMSFIT

- Register for TMSFIT
- Admin login
- Admin shall add new subject, Batch, Degree (course), student, resource and lecturer
- Admin shall update the details of subject, Batch, Degree (course), student, resource and lecturer

- Admin shall delete the details of subject, Batch, Degree (course), student, resource and lecturer
- Admin shall do the student enrolment via excel sheet
- Admin shall upload the students via excel sheet
- Admin input year
- Admin input semester
- Admin shall generate timetables
- Admin shall view timetable
- Admin shall change allocated time slots manually
- Admin shall allocate more than two labs as required manually
- Admin shall save the timetable.
- Admin shall view the resources according to the time allocation
- Admin shall view the lecturer preferred time
- Lecturer login
- Lecturer view the lecturer profile
- lecturers can view the timetables
- Lecturer shall submit time preferences
- Student Login
- Student shall view the student profile
- Student shall view timetable
- Change password
- Print Timetable

#### **4.3.2 Non Functional Requirements of the proposed TMSFIT**

- Load and accuracy
- The system should be available 24 hours
- The system should at the time
- System must be provide specific information to specific user
- The system should not be failed
- Web accessibility

All these functional and nonfunctional requirements were achieved through the proposed Timetable Management System

#### 4.4 Current Timetable Management System

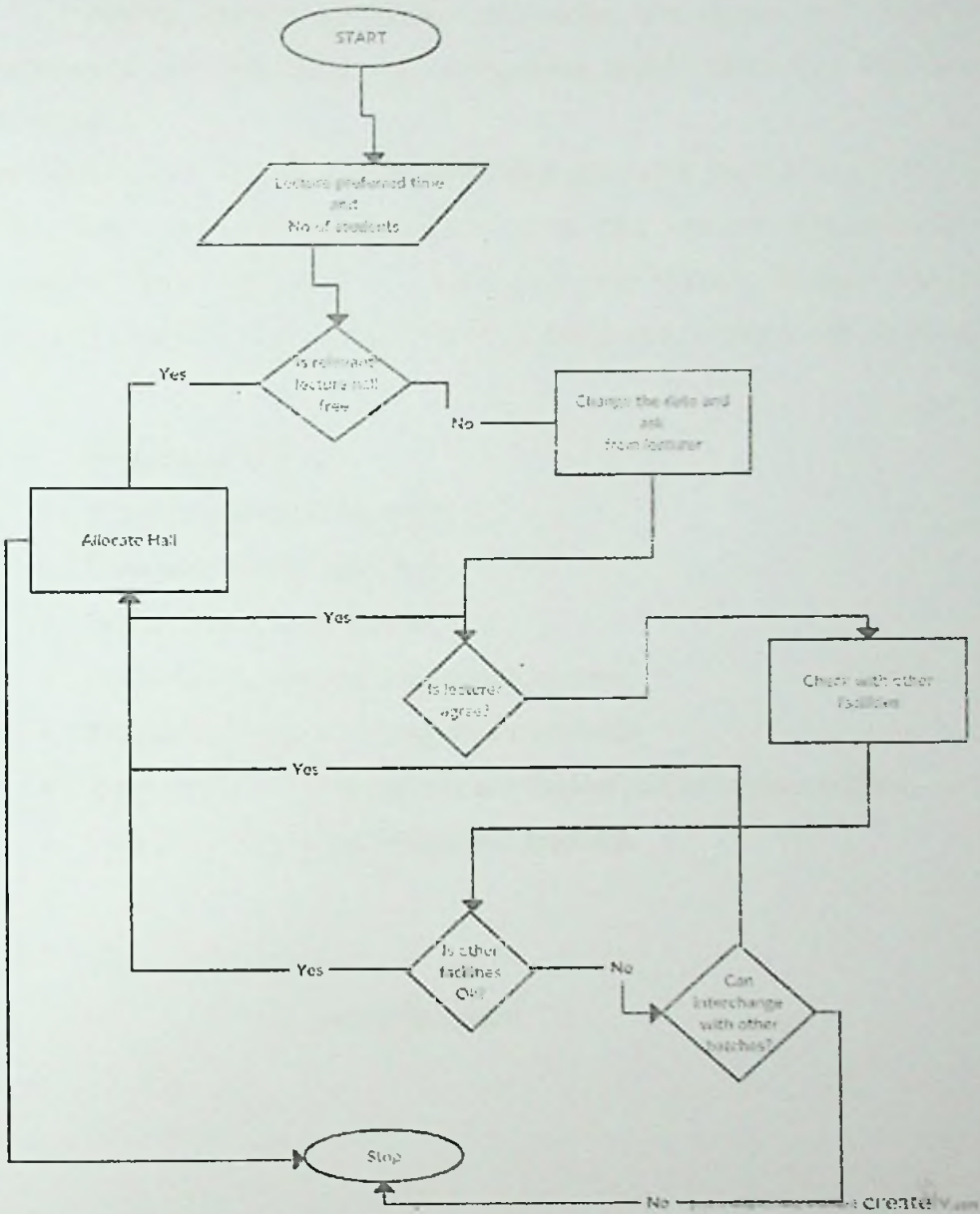


Figure 4-1 Flow Chart of the Current Timetabling System

The processes of current timetable management system also used to observe the system well and it is given by a flow chart as above Figure 4 -1

#### **4.5 Process of the TMSFIT**

Having received the admin information, the system should verify that information provided. With that, authenticate details admin can log on to the dashboard.

Information must be kept in a database called ttms, with Student data, Lecturer data, Class data, Timeslot data, batch data, semester data, resources data and more. This database must be connected to a web server. Use MYSQL database management system. Using PHP language Yii framework design and develop a web application.

#### **4.6 Features of TMSFIT**

- It is a web based online system
- It has user friendly interfaces
- Easy to access to the system
- Accessible through any kind of web browser
- Printable versions of the outputs are available
- Admin profile, Lecturer profile and Student profile can be modified.
- System can be evolved as customer feedback

#### **4.7 Users of the TMSFIT**

- Admin of the Timetable System
- Lecturers
- Students

#### **4.8 Technologies used in TMSFIT**

##### **4.8.1 Genetic Algorithm**

GA is the main technology behind this timetable management system. Using GA, system will generate near optimal timetable solutions not the best timetables.

##### **4.8.2 How GA used in this research?**

- Hard constraints (which can't be violated) were used to calculate the fitness value.



- Soft constraints cannot be evaluated
- If breaks one of the hard constraints the schedule is infeasible

#### 4.8.3 Hard Constraints of Proposed Solution

- Room Overlap – Check if there are two lectures in one room
- Room not enough – No of students of a class is > seats of room
- Required resource not available - Does the lab has required no of Computers?
- Lecturer Overlap – One lecturer can't be in two rooms at the same time
- Student Overlap - One student can't be in two rooms at the same time

#### 4.8.4 System Analysis and Design

Next, system analysis and design sections were done.

According to that, database and the top-level architecture of the system could be figure out.

Because of the fellow facilities, OOAD method was used in the research

- It is better for recognizing objects
- OOAD method organizes the objects by creating object model diagram
- It defines object attributes
- It expressing the behavior of the objects
- It describes how the objects interact

#### 4.8.5 Unified Modeling Language (UML)

Rational Rose and UML diagrams were used in design phase of this TMSFIT to create use case diagrams, class diagrams, sequence diagrams and state chart diagrams

#### 4.8.6 PHP Language

Because of the PHP is a server support, server side language, it was the developing tool used in the TMSFIT. With PHP Object Oriented Programming is accepted.

#### **4.8.7 Eclipse for PHP Plugging**

This was used as the code editor of the proposed timetable management system.

#### **4.8.8 WAMP Server**

WAMP server with PHP 5.6 and MYSQL installed as the local host. Therefore, TMSFIT could easily install in the www. Root directory.

#### **4.8.9 MYSQL Database Management System**

MYSQL was selected to define and store the timetable management (ttms) database.

Database – ttms and the tables were created successfully.

#### **4.8.10 Yii framework**

- Yii framework configured to the computer.
- This uses the object oriented concepts.
- Appropriate code construction was done using Yii code generator.
- Yii code generator consist of Controller generator, Curd generator, Form generator, Model generator and the Module generator [23].
- Model generator construct a model class for specified database table.
- ttmsfit PHP project was generated successfully with use of Yii.
- AlgorithmController class is in the controller class and the main algorithm is inside the model class.

#### **4.9 Software and Hardware requirements**

The software and hardware mentioned in the previous chapter, were needed to success the proposed TMSFIT

## 4.10 Interface Design

### 4.10.1 User Interface Designing with Prototyping

User Interface (UI) prototyping is used as an iterative development techniques in which users are actively involved in the mocking-up of the UI for this system. UI prototypes have several purposes. See the Appendix A

- An item enables to explore the problem space with the users.
- As a design item that enables to explore the solution space of the system.
- It is a way to communicate the possible UI design(s) of the system.
- It is a good foundation to continue developing the system.

### 4.11 Approach to Timetable Generation Process

Following table will illustrate the input, output, process, features and users of the timetable generation process.

Inputs	Outputs	Process	Features	Users
Semester	Generated Timetable	Generate Timetable	Web based system	Admin of the Timetable
Year	Printed Timetable		User friendly interfaces	Lecturers
	View available resources		Accurate	Students

Table 4-1 Approach to Timetable Generation

#### **4.12 Summery**

This chapter focuses on the approaches to the design, appropriate development environment and implementation. Design of the approach will be discuss in detail through the next chapter.

## Design of the TMSFIT

### 5.1 Introduction

Previous chapter included the hypothesis, inputs, outputs, users, features and the technologies of the proposed timetabling management system. This chapter contains details of design (or analysis and design) of the TMSFIT. The top-level design of the proposed system also included. This chapter will describe the Analysis and designing part of the system such as research planning and system designing.

### 5.2 Research Planning

Following major tasks conducted through the analysis and design phases of the research.

- Project Planning was done using an online project-designing tool.
- System Requirement Specification (SRS) completed.
- System Design with necessary design diagrams could introduce.

#### 5.2.1 Planning the Research Project

To planning and scheduling the research project, rough Gantt chart was drawn as the following table 5-1.

Task	Q2		Q3				Q4			Q1		Q2			
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar
1 Identify the Problem		Identify the Problem													
2 Gather Requirements			Gather Requirements												
3 System Design						System Design				System Design					
4 Implementation											Implementation			Implementation	
5 Testing														Testing	
6 Deploy the system															

Table 5-1 Gantt Chart

#### 5.2.2 System Development Methodology for TMSFIT

Waterfall model was used as the system development methodology of this system. Because, it is having precise requirements and well understood milestones.

### 5.2.3 Selection of Software Process Mode for the Proposed TMSFIT

Detail requirement analysis was conducted at each different user category getting help of admin of the current timetable management system.

After the system study, the Software Requirement Specification (SRS) for the proposed system was prepared.

### 5.3 Analysis of the Existing Timetabling System

In order to identify the requirements of the existing system a detailed study at each user category was conducted. User interviews, observation and study of relevant documents were the techniques used to gather requirements of the current system.

### 5.4 Top level Design Diagram

The system's top level diagram presented as below Figure 5-1

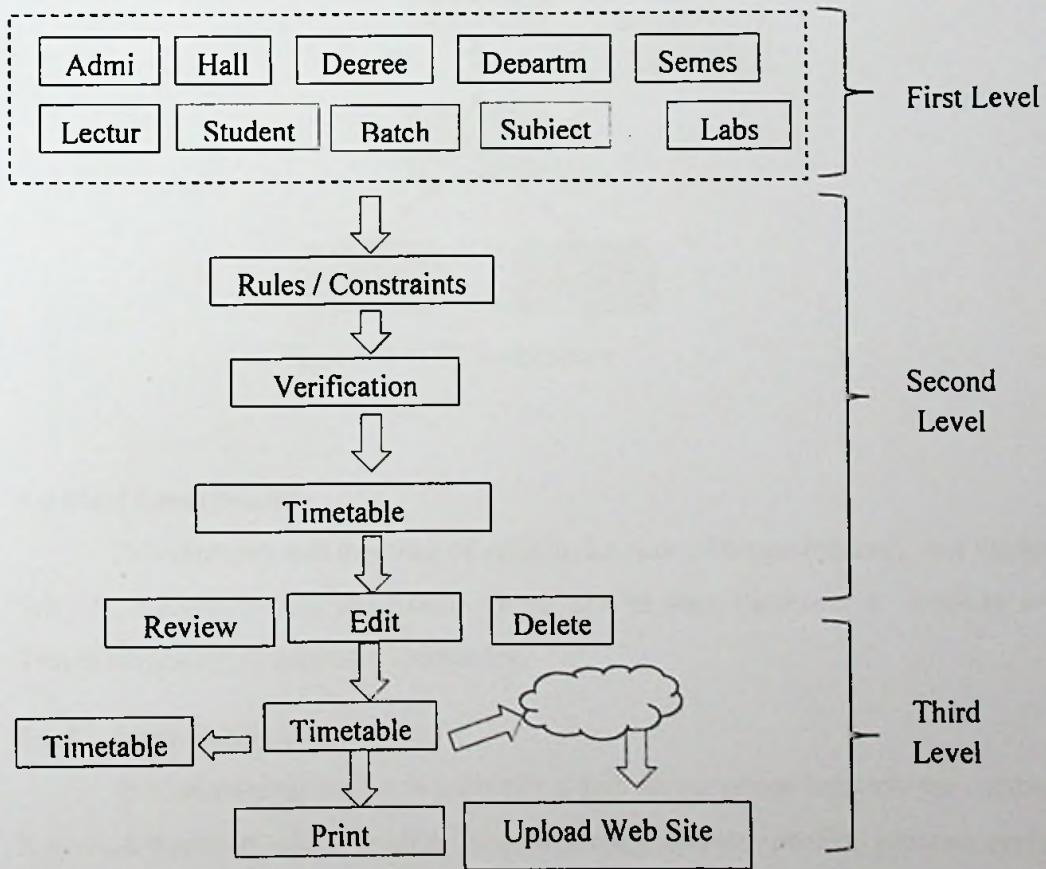


Figure 5-2 Top Level Design of the Proposed System

## 5.5 MVC Architecture

Yii framework employs the model-view-controller (MVC) design pattern and it is broadly accepted in Web programming [24]. Below Figure 5-2 will illustrate the MVC architecture and it shows how user request is handled by the Yii application as a standard workflow.

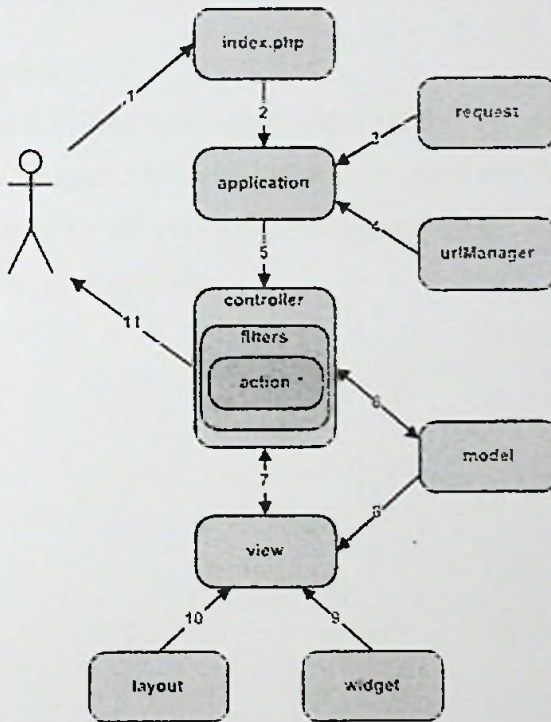


Figure 5- 2 MVC Architecture

## 5.6 First Level Module

This contains sub modules of Admin, Lecturer, Degree (course), and Student, Subject, Resources and the batch. Those are in ttms database. It interacts with Timetabling Engine generates timetables.

### 5.6.1 Timetabling Engine

The timetabling engine is primarily a web server which connects the database. It should maintain admin profile, student profile, lecturer profile, process queries; prepare outputs in various formats and so on. This engine also responsible for accuracy and up-to-date information in the database. It is basically designed for maintain the system integrity, security and the privacy.

## 5.6.2 Database Design of Timetable Management System

The Timetable Management System Database abbreviated as ttms. It stores data of students, lecturers, users, degree programs, subjects, timetables and some more. Student data, resources data, lecturer data, batch data, subject data and timetable data can retrieve from the database. Admin has the authority of modifying and deleting data. Details were taken from the faculty of Information Technology at University of Moratuwa.

### 5.6.2.1 ER Diagram

An Entity Relationship Diagram can show the database design of the TMSFIT. Each box of the diagram represents a table and ovals are the attributes of that table. Underlined attribute are the primary keys of those tables. Following Figure5-3 is the ER diagram of this database ttms.

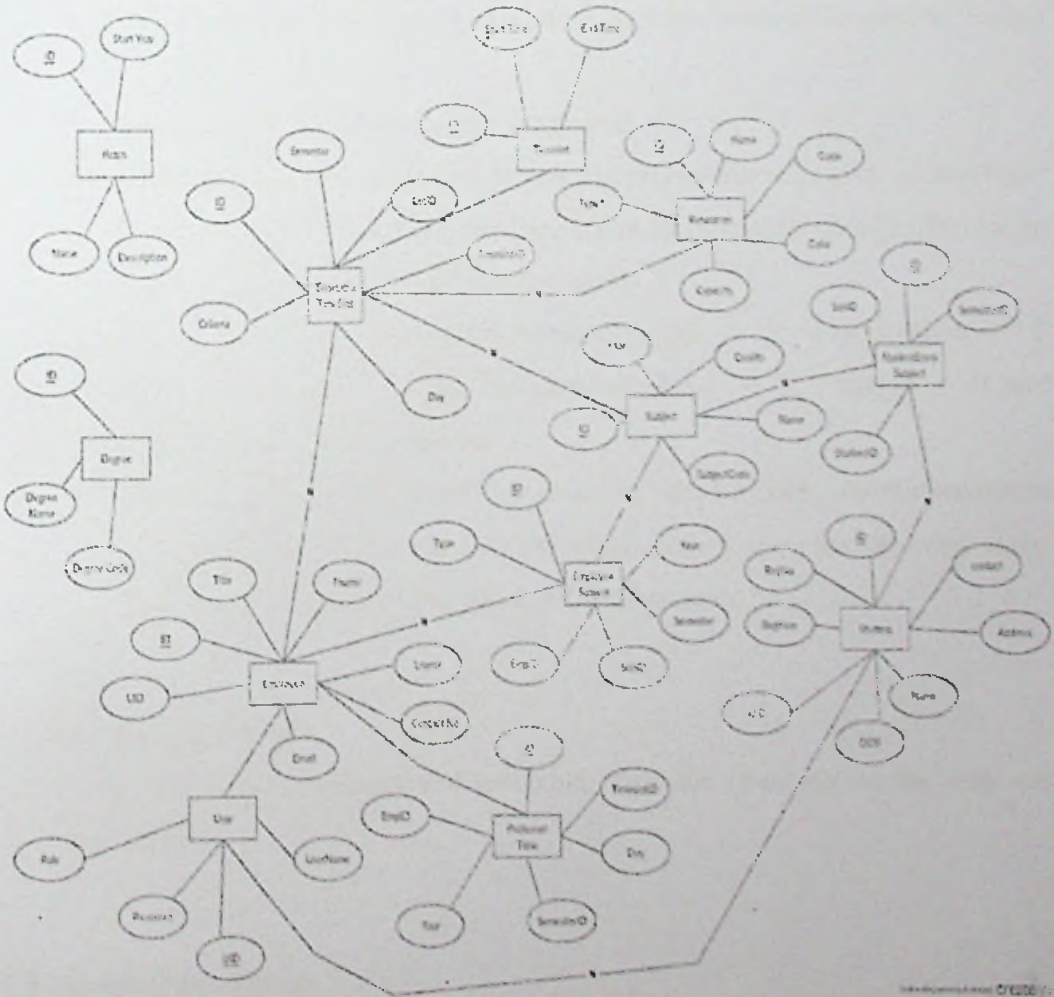


Figure 5-3 ER Diagram



## 5.7 Second Level Module

This includes the logic of the timetable, constraints or rules, verification, timetable generation, view, delete and edit. Algorithms usually kept in this level. These sub modules act as methods and product of this is the generated timetable.

### 5.7.1 How the Genetic Algorithm perform

Following is a brief description of algorithms performance.

- AlgorithmController is reside in the controllers module
- actionLoad() ---> basic action method calling from url
- SubjectClass reside in the model class
- loadAllClassByYearSem() --->Load all the subject registered by student for given year and semester
- Algorithm
- getInstance(\$subjectClass) --->Create Algorithm instance by passing loaded subjectclass
- We are using 100 chromosomes to generate this.
- Inside this method, creates prototype of chromosomes by calling initObject (\$numberOfCrossoverPoints,\$mutationSize,\$crossoverProbability,\$mutationProbability,\$fitness,\$subjectClass) method.
- We need to pass numberOfCrossoverPoints = 2, mutationSize = 2 ,crossoverProbability = 80, mutationProbability = 3 , fitness = 0 and subjectClasses for that method.
- After calling start () in Algorithm class. In here for each 100 chromosomes initialized new Chromosome by assigning subject class for time slot array. Then calculate fitness value for each Chromosome.

## 5.8 Third Level Module

This level includes generated timetable, view the timetable on the web and print the timetable.

## 5.9 Modeling the System

Modeling a system is the process of abstracting and organizing significant features of how the system would look like. Modeling is the designing of the software

applications before coding. Unified Modeling Language (UML) tools are used in modeling this system. UML offers different diagrams to model a system.

Diagrams of this research are listed below:

- Use case diagram
- Class diagram
- Object diagram
- Sequence diagram

In this project, the Use case diagram, Class diagram, Sequence diagram were used for system modeling.

### 5.9.1 Use Case Diagram

Use case diagrams describe what a system does from the standpoint of an external observer. They are used to show the interactions between users of the system and the system. A use case represents the several users called actors and the different ways in which they interact with the system. The main Use Case Diagram is given as below Figure 5- 4

#### Actors

- User (Admin/ Lecturer/ Student)

#### Use Cases

- Login
- Create/Modify and Delete Timetable
- Add Lecturer/Course or degree/batch/subject/resource and Student
- View Timetable
- Change Password
- Add Lecture Resources (Hall / Lab)
- Modify and Delete Lecturer/Course/Student/Resources (Hall/Lab)

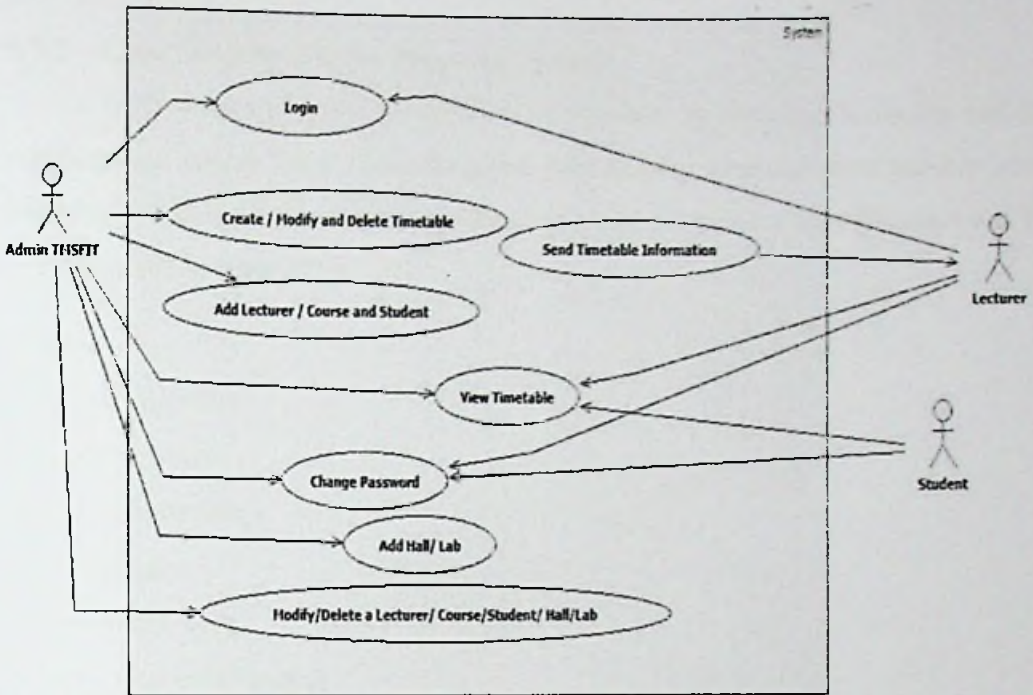


Figure 5-4 Main Use Case Diagram

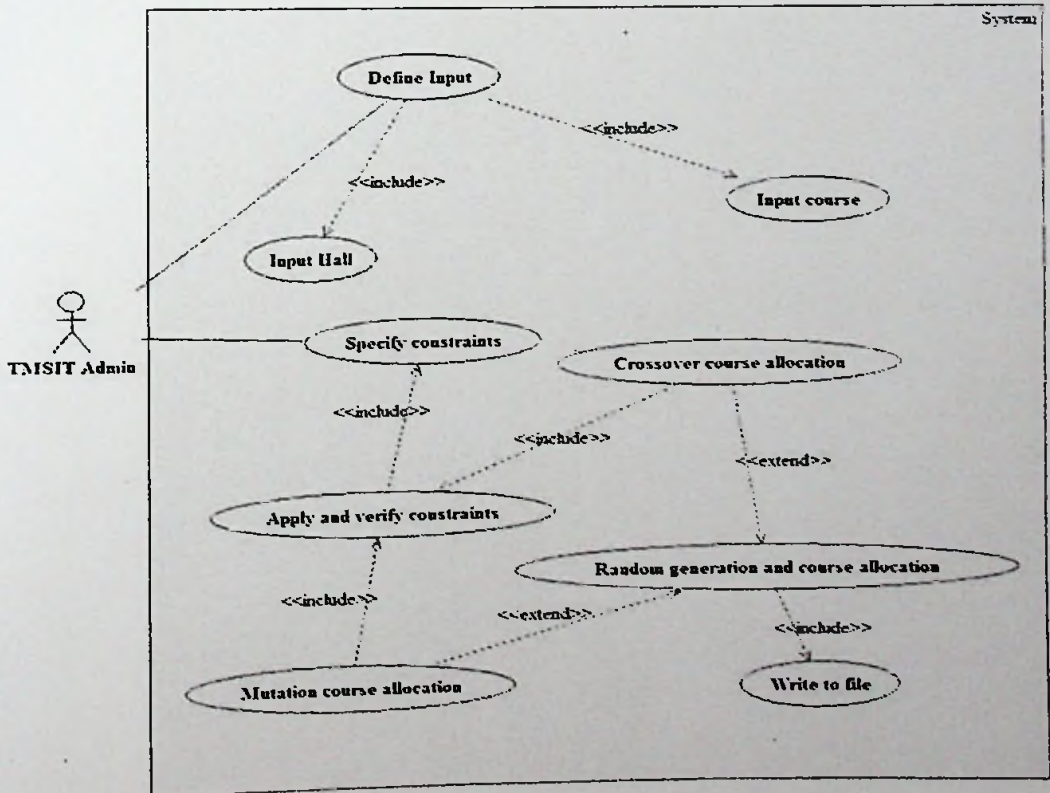


Figure 5-5 Use Case Diagram to Show the Interaction between The System and admin

### 5.9.2 Class Diagram for the Proposed System

A class diagram gives an overview of a system by showing its classes and the relationships among them. Class diagrams only display what interacts but not what happens during the interaction hence they are static diagrams. Class Diagram can be present as below Figure 5-6

#### Classes

- IT Faculty
- Lecturers
- Department
- Rooms
- Halls, Labs
- Degrees (Courses)
- Undergraduates
- Postgraduates

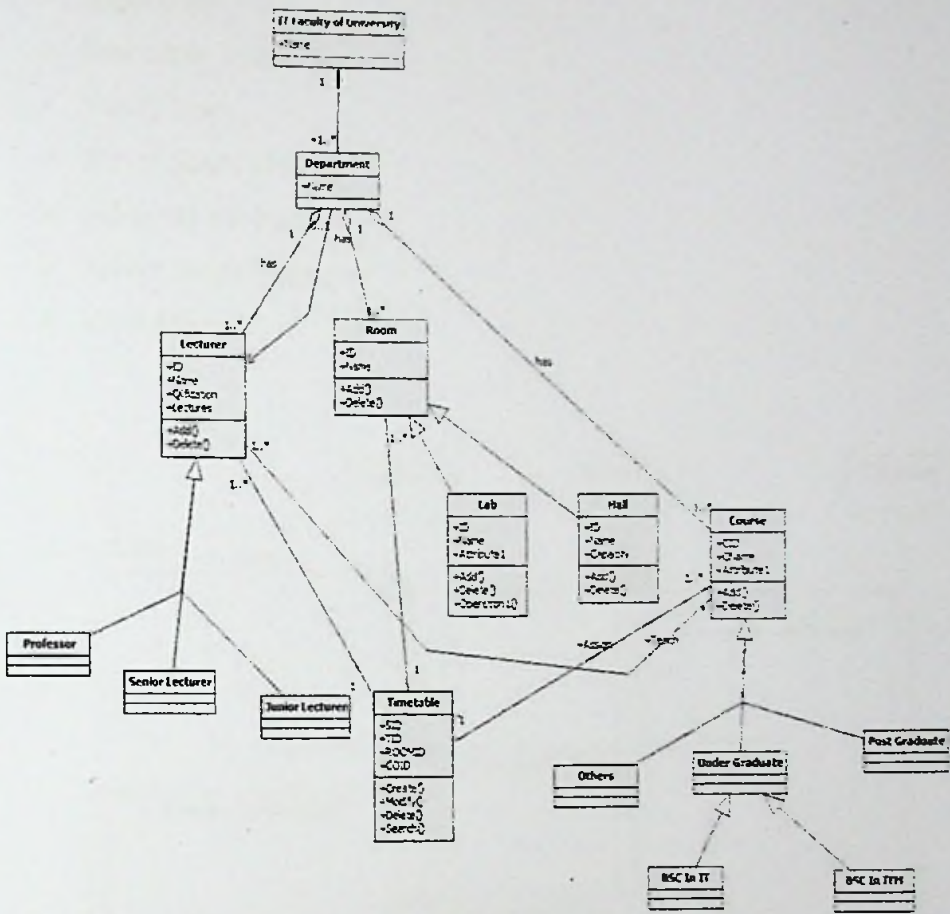


Figure 5- 6 Class Diagram for Overall View of the System

### 5.9.3 Sequence Diagram

A sequence graphically depicts how objects interact with each other via messages in the execution of a use case or operation. They illustrate how messages are sent and received between objects and the sequence of message transfer. It also details in Figure 5-7 how operations are carried out according to the time of operation.

#### Classes

- Admin
- Browser
- Web Server
- Database

#### Messages

- Login

- Valid User
- Successful Validation
- Verify User
- Server Sends Message
- Admin Home Page
- Server Sends Message
- Error Message

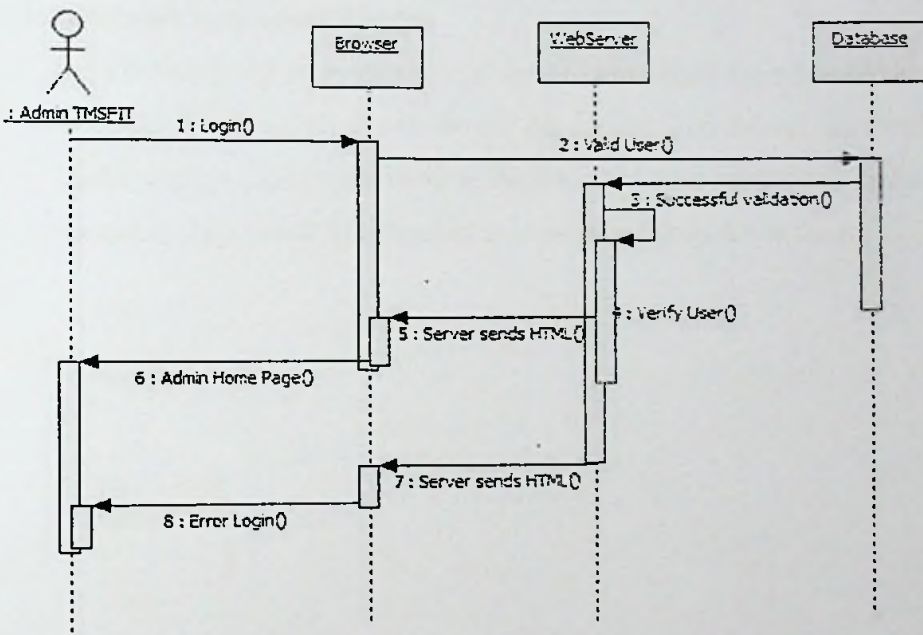


Figure 5-7 Sequence Diagram show how the different objects interact

### 5.10 User Interfaces Design

This is a description of logical characteristics of each interface between the software product and the users interact with the system. User interfaces are consist of any GUI standards, sample screen images, standard buttons (add, delete) and so on. See Appendix A

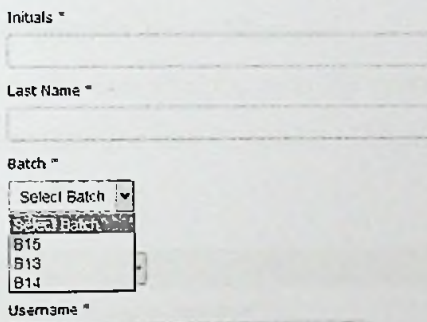
However, UI design must be considered whenever users interact with controls or displays. Throughout the project when designing the HCI, I have followed several UI designing techniques.

Basically, targeted users knowledge type is inexperience, so that I have to consider the following techniques when designing the UI.

- Increased Drop Down usage
- UI based field validation
- Highlighting the focused field
- User friendly Error Messages

### 5.10.1 Increased Drop Down Usage:-

By increasing the drop down field usage rather than typing a value to the system, user can simply select an item from a pre-loaded drop down. This technique is very useful when working with users to decrease the input errors and increase the user performance. Figure 5-8 will illustrate that increased drop down usage.



The image shows a form with four input fields. The first field is labeled 'Initials' and is empty. The second field is labeled 'Last Name' and is empty. The third field is labeled 'Batch' and is a dropdown menu. The dropdown menu is open, showing a list of options: 'Select Batch', 'B15', 'B13', and 'B14'. The fourth field is labeled 'Username' and is empty.

Figure 5- 8 Drop down Usage

### 5.10.2 UI based Field Validation:-

User is allowed only to enter character value to character type fields, which means rather than validating the user input by the system, numeric fields validations has been implemented at the user interface level.

Initials \*

Last Name \*

Batch \*

Username \*

Figure 9 UI based field validation

### 5.10.3 Highlighting the Focused Field:-

When the user working with the system as an extra facility, currently prompted (focused) field would be colored by the system. Because of this technique, user can easily identify the current working field.

### 5.11 User-friendly Error Messages

When the user working with the system, some validation rules with the error messages navigate the user to correct operation. Below Figure 5-10 will show a user friendly error message.

CONTROL PANEL    LOGOUT (ADMIN)

Create Lecturers  
 Manage Lecturers  
 Manage Assign Subjects  
 Logout of This Site

### Create Lecturer

Fields with \* are required.

Please fix the following input errors:

- Email cannot be blank
- Title cannot be blank

Title \*

Title cannot be blank.

First Name \*

Last Name \*

Phone \*

Username \*

Figure 5- 10 User-friendly Error Messages

### 5.12 Summery

This chapter briefly described the analysis and design methodologies of the system. It described what each module does. Further, it has given design diagrams examples like Use case diagrams, Class diagrams, Sequence diagram. Next chapter will discuss the how solution is devised as implementation.



# Implementation

## 6.1 Introduction

Previous chapter provided the design of the proposed solution. This chapter is about how the solution will be implemented in the context of proposed timetable management system. This chapter will further describe about equipment, tools, software and hardware required for implementing processes of the proposed system. Moreover, Algorithm implementation, Database implementation and Interface implementation are been thoroughly discussed in this chapter. Some of the user interfaces are defined as Appendix A

## 6.2 Software Requirement for Implementation Process

Since the proposed system is web-based system, some of the following open source software were installed and configured before implement the proposed system. This system front end has been developed using HTML, CSS, JavaScript and Bootstrap. Further, for making server application and communicate with database it used server side language PHP. MYSQL serves the front end.

- WAMP Server
- Yii Framework
- Eclipse for PHP Developers

### 6.2.1 WAMP Server

While WAMP Server is a Windows web development environment, it permits us to create web applications with Apache2, PHP and a MySQL database. Further, it is combined with PHP MyAdmin for the purpose of database management.

Download WAMP from <http://www.wampserver.com/> website. Upon our computer configuration, I have chosen 64bit version WAMP.

WAMP Server installs automatically (with an installer). Therefore, we can fine tune our server.

WAMP Server also has a tray icon to manage our server and its settings. If the icon blinks green, server is running well and can log in to the local host. If the icon

does not turn to green color and still in red or orange color, check whether the port 80 sharing with Skype. If so, uncheck the Skype port 80 and restart the all the services of the server.

### 6.2.2 Yii Framework

Yii framework needs PHP 5.1. Therefore, the server must have PHP 5.1 or above versions and available to the web server. Moreover, Yii has been tested with Apache HTTP server on Windows and Linux [25]

1. Downloaded the latest version of Yii
2. Extracted at most contented route
3. Registered the path at Window Environment Variables as  
C:\Wamp2.4\bin\php\php5.4.16
4. Logged off windows to take effect
5. Open the Windows command prompt and then at the console environment typed "c:\framework\yiiic webapp wamp2.4\www\name\_of\_the\_application".

### 6.2.3 Eclipse for PHP Developers

Eclipse is tools useful for PHP developers creating Web applications.

## 6.3 Hardware Requirement for Implementation Process

- Computer with 2GB RAM
- 2GHz or more processing power
- Internet connection
- 15" Monitor
- Router
- Keyboard and Mouse

## 6.4 Implementation of First Level module

### 6.4.1 Interface Implementation

There are fifty interfaces include in this TMSFIT system. Following classes are associated with some of those interfaces. Those classes are residing in model and controller of the MVC architecture.



- LoginForm, SiteController and User classes used for login interface.
- DashboardController class used for dashboard interface.
- UserController class used all the interfaces with CRUD operators.
- StudentController, Student and Studentenrolsubject classes used for Create Student, Update Student, Delete Student, Manage Student, View Student Timetable,
- BatchController and Batch classes used for Create Batch, Update Batch, Delete Batch, View Timetable and Manage Batch interfaces.
- SubjectController class Subject classes used for View Subject, Create Subject, Delete Subject, Enrole Students, Manage Subject and Assign Lecturer to Subject interfaces.
- ResourceController class used for Create Resource, Update Resource, Delete Resource and View Resource Allocation interfaces.
- EmployeeController, EmployeeSubjectController, Employee and EmployeeSubject classes use Create Lecturer, Update Lecturer, Delete lecturer, View Lecturer Timetable, Lecturer Preferred Time and Manage Lecturers interfaces.
- EmployeeSubject class use for Assign Lecturer to Subject interface
- Algorithm class use for Generate timetable and View timetable interfaces

#### 6.4.1.1 Login Interface

- By entering Admin login name and password, admin dashboard interface can be accessed.
- By entering Lecturer login name and password, admin dashboard interface can be accessed.
- By entering Student login name and password, admin dashboard interface can be accessed.

#### 6.4.1.2 Dashboard Interface

Dashboard interface consist of a menu and icons. Further, Dashboard menu contains control panel and logout. Degree, Students, Lecturer, Subjects, Resources, Batch and Timetable icons are seen on the dashboard. See Appendix A

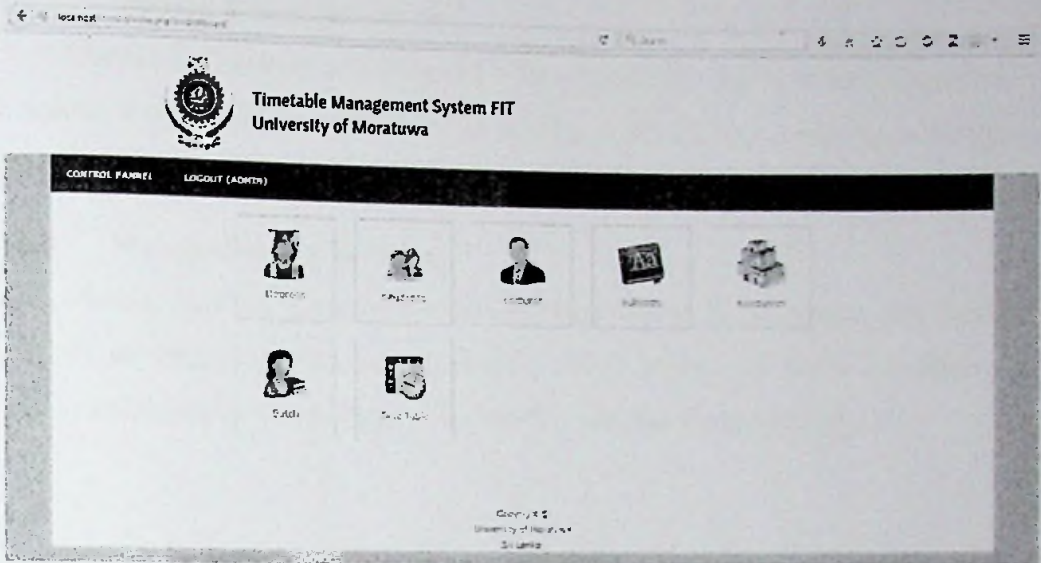


Figure 6-1 Dashboard Interface

#### 6.4.1.3 Manage Degrees Interface

This Manage Degrees interface contains Create Degree sub menu and list of degrees. Each degree can be searched, edited and deleted.

#### 6.4.1.4 Manage Students Interface

Manage Students interface includes create and upload sub menus. Moreover, it has a list of students' details and options of searching, updating and deleting.

#### 6.4.1.5 Manage Lecturers Interface

Manage Lecturers interface consist of create lecturer, manage assign subjects and lecturer timetable sub menus. Further, it has a list of lecturers' details and options of searching, updating and deleting.

#### 6.4.1.6 Manage Subjects Interface

Manage Lecturers interface encloses create subject, enroll students and enrolled student sub menus. It also contains a list of subjects' details and options of Assign lecturer to subject, searching, updating and deleting.

#### **6.4.1.7 Manage Resources Interface**

Manage Lecturers interface consist of create resource and view time allocation sub menus. It also contains a list of resources details and options searching, updating and deleting.

#### **6.4.1.8 Manage Batches Interface**

Manage Batches interface contains Manage Batch, Create Batch and Batch Timetable sub menus. It also contains a list of batch details and options searching, updating and deleting. Below figure will show the Manage Batches Interface.

#### **6.4.1.9 Generate Timetable Interface**

Generate Timetable interface holds create View Timetable and Generate Timetable sub menus. Further, it contains run algorithm option.

### **6.4.2 Database Implementation**

- MySQL is used as the back end of this system. Because it includes number of engines and delivers SQL commands to operate database.
- As the first step of developing this system, a proper database was constructed in phpMyAdmin. In this system, it is called ttms as mentioned in the ER diagram.
- InnoDB engine use for foreign keys, support transactions and row level locking.
- According to the ER diagram in Design chapter with main entities such as student, subject, timeslot, lecturer, degree and department main tables were generated. Moreover, batch, degree, department, employee, employeesubject, preferredtime, resource, student, studentenrolsubject, subject, timeslot, timetableslot and user tables also implemented.
- Primary keys are assigned properly to avoid duplicate fields.
- In addition, user also can back up database, import database and export database any time.

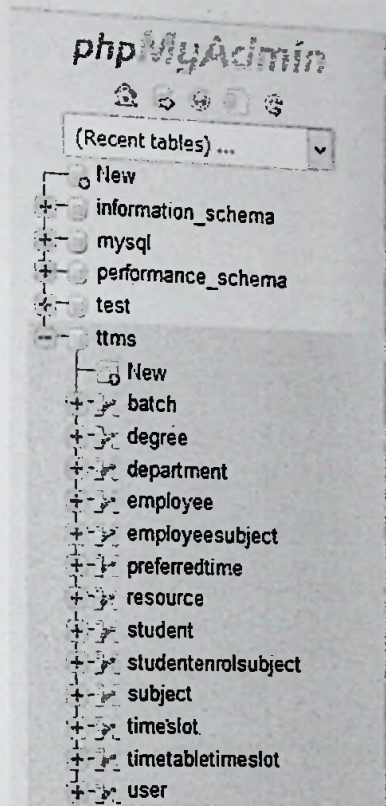


Figure 6-2 Tables of ttms

### 6.4.3 The Process of Mapping Database Tables with Model class

- By login in to <http://localhost/ttmsfit/index.php?r=gii>, we come to the Yii code generator.
- Then, we can use the model generator, which generates a model class for the specified database table Eg batch.
- It generates all the appropriate user interfaces mapping the tables of the database.
- Using Curd generator, generate a view script file that displays a form to collect input for the specified model class. Eg batch is under view module
- After creating the appropriate model classes for the tables in the database CURD generator generates a controller and views that implement CRUD operations for the specified data model.

Code regarding `_form.php` can be referred in Appendix B and Interface according to the `_form.php` is as below in Figure 6 -3.

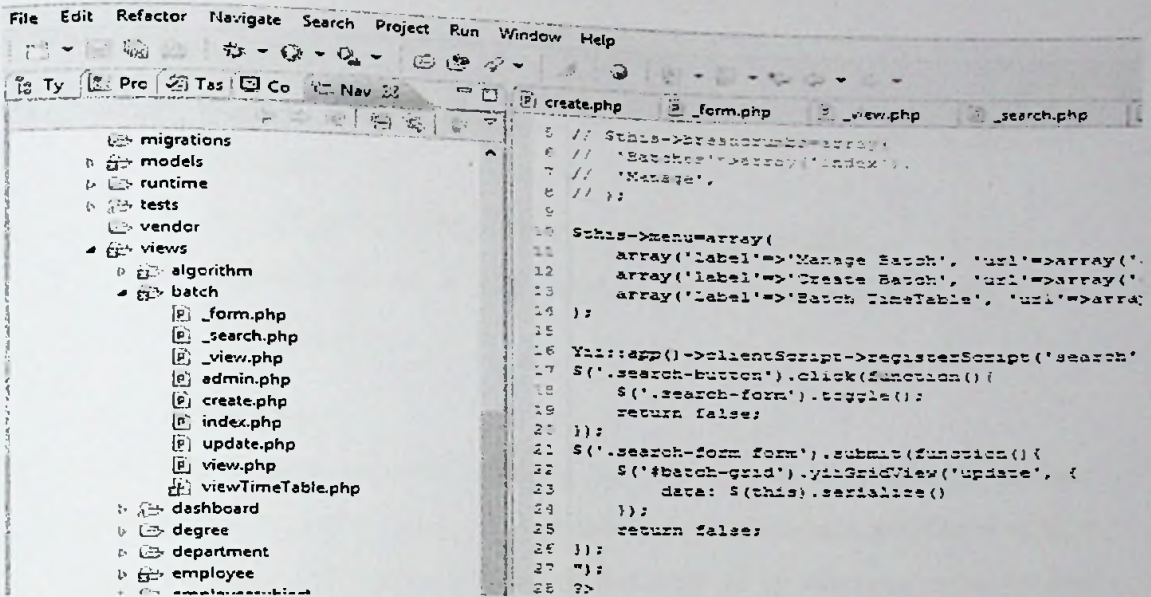


Figure 6-3 Interface of `_form.php`

## 6.5 Second Level Module

Implementation of the logic of the timetable (Algorithm), constraints or rules, verification, timetable generation, view, delete and edit is describe as below.

### 6.5.1 Algorithm Development

- AlgorithmController class extends the controller class.
- Algorithm class developed under model class by extending CFormModel.
- Code regarding the algorithm implementation can be found in Appendix B

### 6.5.2 Hard Constraints of TMSFIT

Hard constraints cannot be violated when the generation of the timetable. These hard constraints are reside in the `calFitness()` and always use to evaluate the fitness value of the timetable schedule. See the Appendix B for fitness function

- Room Overlap – Check if there are two lectures in one room
- Room not enough – No of students of a class is > seats of room
- Required resource not available - Does the lab available the required no of Computers?

- Lecturer Overlap – One lecturer can't be in two rooms at the same time
- Student Overlap - One student can't be in two rooms at the same time

### 6.5.3 Initialization

- Create an initial population and this population is usually randomly generated.
- Regarding this solution 100 chromosomes must be taken.
- Chromosome or a class schedule must be defined first. Eg public `$_chromosomes`

### 6.5.4 Evaluation

The fitness value is calculated by how well it fits with our desired requirements. Evolution's goal is to find better individuals in each generation using a fitness function. The process of evolution is maintained by selection, crossover and mutation. Below code segment will illustrate the chromosome construction.

```
FunctioninitObject($numberOfCrossoverPoints,$mutationSize,$crossoverProbability,
$mutationProbability,$fitness,$subjectClass)
```

```
{
    // reserve space for time-space slots in chromosomes code
    $this->_slots = new SplFixedArray(Schedule::DAYS_NUM *
Schedule::DAY_HOURS * count(Resource::model()->findAll()));
    $this->_criteria = new SplFixedArray(5 * count($subjectClass));
    $this->_mutationSize = $mutationSize;
    $this->_numberOfCrossoverPoints = $numberOfCrossoverPoints;
    $this->_crossoverProbability = $crossoverProbability;
    $this->_mutationProbability = $mutationProbability;
    $this->_fitness = $fitness;
    $this->_subjectClass = $subjectClass;
    $labs = Resource::model()->findAll("type = 'lab'");
    $IHalls = Resource::model()->findAll("type != 'lab'");
    $this->_allClassRoom = array_merge($IHalls,$labs);
    $this->_noOfClassRoom = count($this->_allClassRoom);
    $this->_noOfLabs = count($labs);
}
```



```

$this->_noOfHalls = count($IHalls);
}

```

Class table for chromosome and used to determine first time-space slot used by class

```

public $_classes, public $_subjectClass, public $_noOfClassRoom, public
$_noOfLabs, public $_noOfHalls, public $_allClassRoom;

```

Example:

Each hour of a class has a separate entry in the array, but there is only one entry per class in the class index map. For instance, if a class starts at 1pm and lasts for three hours, it has entries in the 8.15am, 9.15am, and 10.30 slots. Below Table 6-1 will illustrate the Chromosome

$$\text{Array size} = \text{No of time slots (10)} * \text{No of Halls(5)} * \text{No of Days(5)}$$

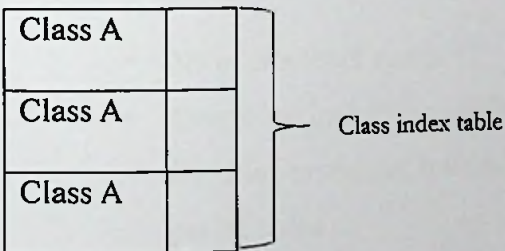
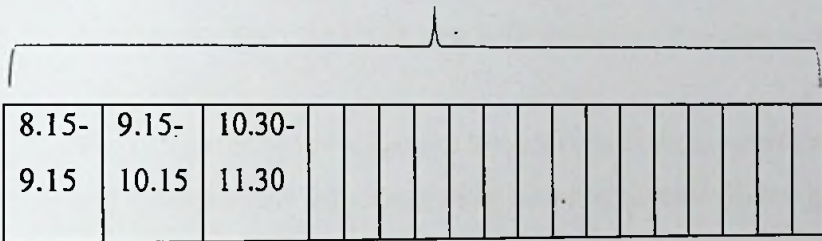


Table 6-1 Chromosome Representation

### 6.5.5 Evaluation

The fitness value is calculated by how well it fits with our desired requirements. Evolution's goal is to find better individuals in each generation using a fitness function. The process of evolution is maintained by selection, crossover and mutation. Below diagram, illustrates

### 6.5.6 Selection

- Selection chooses superior individuals in every generation
- We want to be constantly improving our populations overall fitness.
- Selection helps us to do this by discarding the bad designs and only keeping the best individuals in the population

In this research, find the Appendix B for calFitness function code segment

- Use 100 Chromosomes as this pool(gene)
- Calculate fitness value for chromosome by using calFitness()
- calFitness() uses the hard constraints and increase the score one by one 0 to 5
- Then select the most suitable 5 chromosomes as timetable schedules.

### 6.5.7 Crossover

- During crossover we create new individuals by combining aspects of our selected individuals.
- The crossover operator chooses two individuals from current population (parents) and creates a new individual (child) based on parents' genetic material.

In this research it describes as Figure -14

- No of crossover points =2
- Crossover probability =80%
- By using crossover(\$parent2) function, make new offspring by combining parent codes
- Then check the fitness again using the calFitness()
- If found a fitter chromosome than a previous selected one change it to new one

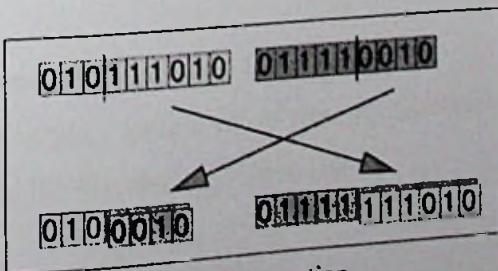


Figure 6-4 Crossover operation

### 6.5.8 Mutation

- Mutation typically works by making very small changes at random to an individual's genome.
- The mutation operator changes the value of some genes in an individual and helps to search other parts of problem space

With regards to this solution, see the Appendix B for mutation code segment.

- Mutation probability = 3
- By using mutation() generate new chromosomes
- Then again check their fitness with previous 5 best chromosomes using calFitness()
- If found a fitter chromosome than a previous selected one change it to new one

After these three steps selection, crossover, mutation, 5 best chromosomes (timetable schedules) have been generated.

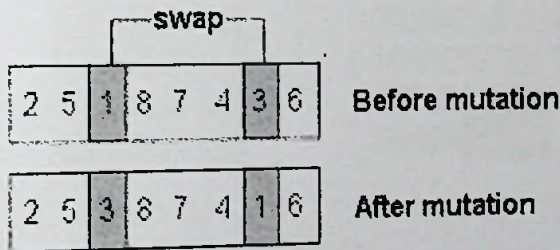


Figure 6-5 Mutation operation

## 6.6 Third Level Module

This includes generated timetable, view the timetable on the web and print the timetable.

### 6.6.1 Generate Timetable Interface

While it takes quite a lot of time to generate a timetable, use a progress bar as waiting facility. See the Figure 6-6



Figure 6-6 Timetable Generating State

Time Slot	Mon	Tue	Wed	Thu	Fri
08:15 - 09:15				14 2320 1281	14 2320 1281
09:15 - 10:15				14 2320 1281	14 2320 1281
10:30 - 11:30	14 2320 1281				
11:30 - 12:30	14 2320 1281				
12:30 - 13:30					
01:30 - 02:30	14 2320 1281	14 2320 1281			14 2320 1281
02:30 - 03:30	14 2320 1281	14 2320 1281			14 2320 1281
03:30 - 04:30	14 2320 1281				14 2320 1281
04:30 - 05:30	14 2320 1281				14 2320 1281
05:30 - 06:30					
06:30 - 07:30					

Figure 6-7 Generated Timetable

### 6.7 System Deployment

As agreed earlier with timetable administrator, parallel deployment was conducted. As a result, both previous system and new system work together. If the new system is better than the previous one and success, then, they can smoothly shift to the new system.

### 6.8 Summary

This chapter was about implementation details such as implementation of software, hardware, and algorithm, modules of the top-level design architecture, main system user interfaces, sample codes and some more.

## How the System Works

### 7.1 Introduction

Previous chapter explain the implementation phase of the TMSFIT. That included Interface implementation, Database implementation and the Algorithm implementation in the various levels. Further, it included software and hardware requirements for the sample code segments with main system interfaces. This chapter will discuss how the system works with regards Admin, Lecturer and Students roles. The purpose of this chapter is to provide a user guide for this TMSFIT.

### 7.2 System Administrator's Role

System Administrator has all the responsibilities of the Timetable Management System. Further, he or she has authority to register lecturers and students to the system.

- Admin shall log on to the TMSFIT using a valid user name and a password. Login interface can be finding as following figure.

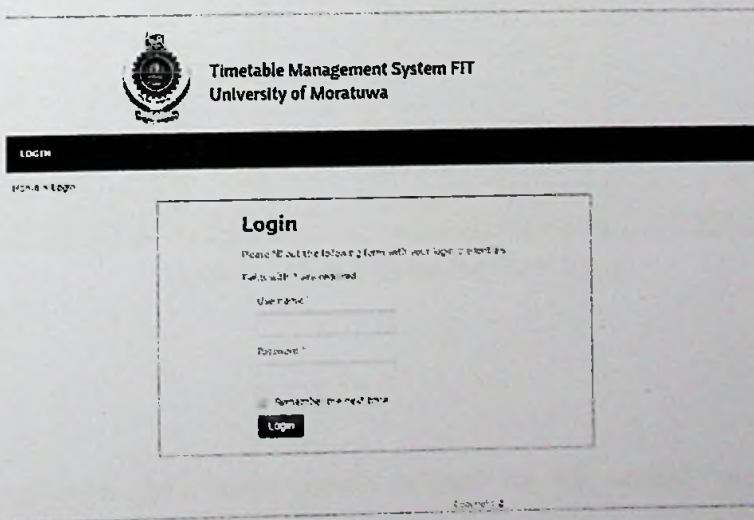


Figure 7-1 Login Interface

- Admin has the power of create, update and delete Degree, Student, Lecturer, Resource and Batch.

- Then, he or she will be able to see the dashboard or control panel of the system.
- Admin has the authority of creating resources such as labs, halls and assign specific colors to identify those in the timeslots. See Appendix
- Most probably, faculty previously defined the offering degree programs. Therefore, not always necessary to create or change the existing degrees.
- Student registration for the system must be essential when anew academic year starts. It can be done as Create Student or Upload Students options. When using create students, only one student can be created at a time. On the other hand, Using upload option, admin can upload as below mentioned Figure 7-2 an excel sheet of all the students in a particular batch.

Student reg	Fname	Lname	Gender(Male/Female)	Date of Birth (YYYY-MM-DD)	Address	Contact	Email	Degree
134002T	M.M.S.U.	SEERATHNE	Female	2/4/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134004C	M.M.A.	M.M.A.	Female	2/3/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134005F	S.L.S.	A.W.M.D.M	Female	2/5/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134006J	E.V.K	ALDIS	Female	2/7/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134007M	M.Z.F.	ANALA	Female	2/9/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134008A	G.K.S.M.	SHARASINDHE	Female	2/9/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134009V	S.M.A.	SUNANARODANA	Male	2/15/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134012A	M.W.L.	ASAUSA	Female	2/13/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134014D	A.S.W.E.A.	BANDARA	Female	2/22/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134016N	M.M.C.M.	BANDARA	Male	2/13/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134017T	P.P.B.M.	BANDARA	Male	2/14/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134018V	R.M.U.S.	BANDARA	Male	2/15/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134021B	V.D.	CHAMARA	Male	2/16/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134022E	U.L.R.	CHANCHAGINTHA	Male	2/27/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134026D	D.	DANDENYA	Female	2/13/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134029G	W.M.A	DARSHANI	Female	2/19/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134030C	D.V.A.U.	DARSHANI	Female	2/20/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134031F	W.H.M.T.	DARSHANI	Female	2/21/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134034R	K.T.M.B.	DE SILVA	Female	2/23/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134038V	M.S.D.	DE SILVA	Female	2/24/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134037E	M.G.R.	DIDAKA	Female	2/24/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134032H	V.K.D.	DILMI	Female	2/25/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134039L	Q.M.R.	DISSANAYAKA	Female	2/26/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT
134040S	I.G.A.N.	DISSANAYAKA	Female	2/27/1993	Gampaha	33224969	msu12@yahoo.com	BSc.IT

Figure 7-2 Student Uploading Excel Sheet

- Admin shall create certain subjects for the desired batch for that academic year using below create subject interface.
- Admin has to create lecturers before assign them to specific subjects

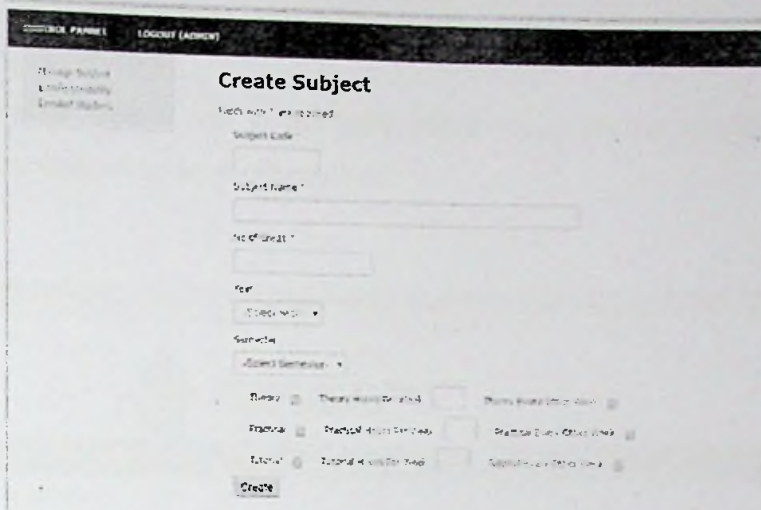


Figure 7-3 Create Subject Interface

- System also provides the facility of advanced searching of students, lecturers and subjects.
- Then, admin shall assign lecturers for particular subjects.
- After that, admin has to enroll students to subjects by using an excel sheet or add students one by one through the system.
- When the above activities are success, admin can generate a timetable for a particular semester of the desired academic year via system.
- This timetable may not fully optimal as Figure 7-4. Hard constraints are shown in the top left hand side of the timetable.
- Therefore, by considering hard constraints and previously defined soft

CONTROL PANNEL      LOGOUT (ADMIN)

### Timetable

- Room Overlap.
- Room not enough.
- Required resource not available.
- Lecturer Overlap.
- Student Overlap.

	Mon	Tue	Wed	Thu	Fri
08.15 - 09.15					IN 2110 PLAC LAB 08
09.15 - 10.15	IN 2110 PLAC LAB 08				IN 2110 PLAC LAB 08
10.30 - 11.30	IN 2110 PLAC LAB 08				
11.30 - 12.30					
12.30 - 13.30				IN 2110	IN 2110
01.30 - 02.30		IS 2010 4L-02-TU		1L-02-TU	4L-02-TU
02.30 - 03.30	IS 2010 4L-02-TU	IS 2010 4L-02-TU		IN 2110 1L-02-TU	IN 2110 4L-02-TU
03.30 - 04.30	IS 2010			IN 2110 1L-02-TU	

Figure 7-4 Timetable without fully optimal

constraints, admin has to manually adjust the generated timetable using another window as below Figure 7-5.

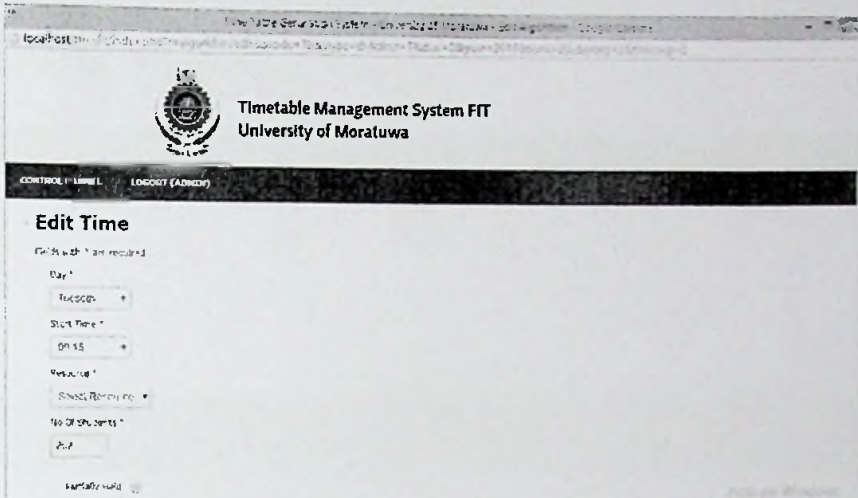


Figure 7-5 Timetable Editing Window

- Finally, the fully optimal timetable can be saved and it can be viewed any time. Further, when a new timetable saved, the previous timetable for that semester will be automatically discarded by the system.
- Moreover, Admin shall view generated timetable for desired batch and view allocated resources through the TMSFIT.
- System has the facility of getting printouts as Figure 7-6 of the generated timetable too.

### 7.3 Lecturer's Role of the TMSFIT

- Admin of the TMSFIT creates lecturer profile.

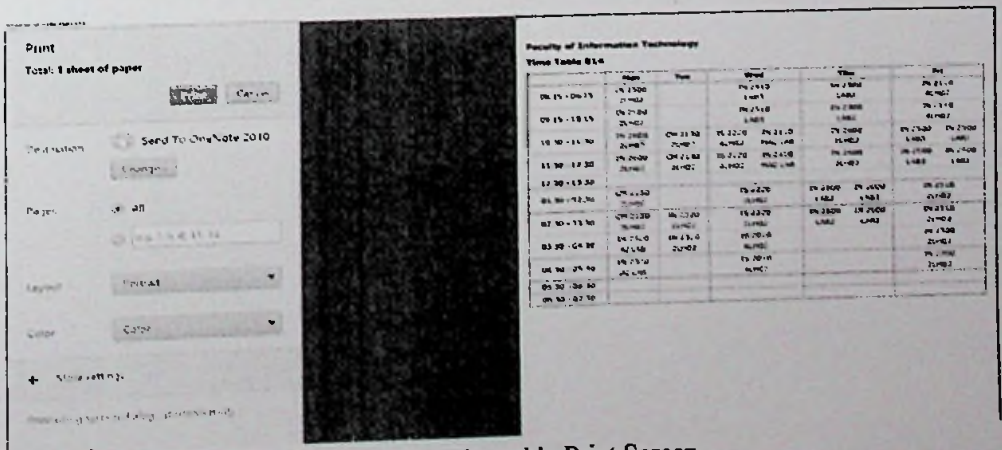


Figure 7- 6Timetable Print Screen



- By using valid username and a password, he or she can log on to the system. Lecture dashboard can be seen as below figure 7- 7.

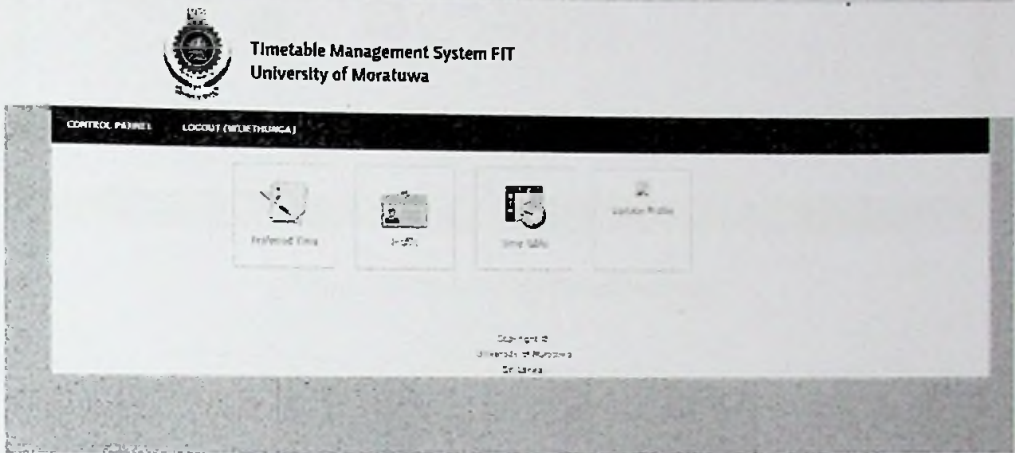


Figure 7-7 Lecturer Dashboard Interface

- Further, Lecturer shall enter preferred time through the system.
- Lecturer can view the timetable with accordance to their subject areas.
- Lecturer shall change the password as necessary.

#### 7.4 Student's Role of the TMSFIT

- Admin of the TMSFIT creates student profile.
- By using valid user name and password student can log on to the system.
- Further, student can view the particular timetable.
- He or she can change the password from update profile option.

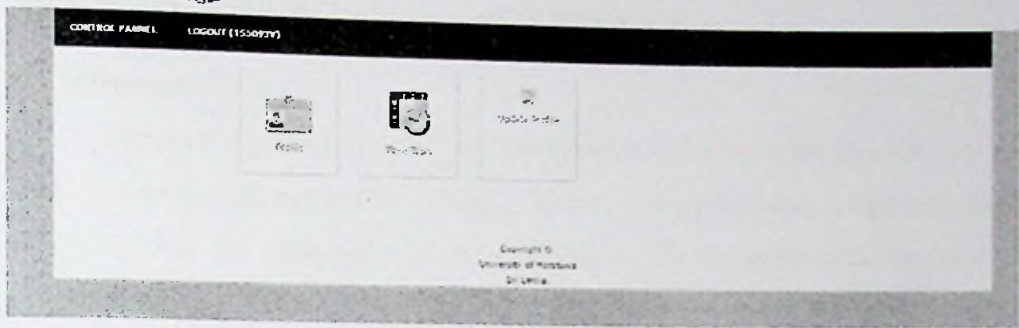


Figure 7-8 Student Dashboard Interface

## 7.5 Summery

This chapter described how the system works. Next chapter will define about how the system evaluated. It will further explain whether the objectives have been achieved using black box testing and white box testing, drawbacks, and limitations of the proposed TMSFIT.

# Evaluation and Testing of TMSFIT

## 8.1 Introduction

Previous chapter described how the system works from the view of user role such as Admin, Lecturer and the Student. Testing is important due to maintaining the reliability and the robustness of an application. This chapter will describe the evaluation of the system with sample test cases. Further, it will discuss whether the system meets objectives defined earlier. In addition, it will discuss about performance and robustness of the system.

## 8.2 Evaluation Strategy

A proper system evaluation strategy formulated for the purpose of evaluation process. In that sense, the answers for the following questions evaluated through a suitable mechanism.

- Can this system save time and optimize resources such as labs and halls?
- Can the users of the system, use it effectively and easily?
- Is this system meets its goal and objectives?

This TMSFIT was evaluated using number of evaluation techniques such as interviews, observation and questionnaires.

### 8.2.1 Interviews

Several interviews were conducted with Admin of the timetable management system, some of the lecturers and some of the undergraduate students. System deployed in parallel way and Admin gave her direct feedback about the system functionalities.

There are more than three face-to-face interviews could be conducted with admin staff and some of the lecturers. During the interview their feedback about the system were noted down.

### 8.2.2 Observation

System observed through sample input data for each interface, their anticipated outputs and their definite output results in the evaluation phase of the TMSFIT system. Several browser capabilities such as Google Chrome, Firefox, and

Opera also successfully tested. If the system takes too much loading time, users may not satisfy about it. Therefore, loading time for all the interfaces has to be considered. Until it is hosted on a server of the campus, timetable component has to be given to admin staff. It was installed their personal computers with WAMP server. Finally, their feedback evaluation could be gathered by a questionnaire.

### 8.2.3 Questionnaire

TMSFIT was further evaluated through a questionnaire by supplying that to timetable admin staff, some of the lecturers and some undergraduate students. From that, their satisfaction level of the system could be discovered. See Appendix C.

## 8.3 Software Testing

Software Testing involves executing an implementation of the software with test data and examining the outputs of the software and its operational behavior to check that it is performing as required. Testing is a dynamic technique of validation and verification because it works with an executable representation of the system.

Validation is set of activities that ensure that the software that has been built is the one that user needs which matches with their requirements.

Verification is set of activities that ensure that software correctly implements the specified functions. Following testing methods were adopted during the testing process.

- Unit Testing
- Integration Testing
- Validation Testing
- System Testing
- Acceptance Testing

Detailed descriptions about 14 test cases are included in Appendix C.

By using, understand tool as a software white box testing could be done. See the Appendix C



## 8.4 TMSFIT Evaluation

As manual timetabling, is tedious task and time consuming, automated timetabling is becoming more and more useful. TMSFIT makes that easier than ever, because of its flexibility, and simplicity. Timetable admin staff provided a very successful feedback through above mentioned evaluation processes.

From the admin's point of view, overall system is success. Further, this TMSFIT is time effective and user friendly. She has a small confusion of the quantity of data handle by the system. There are four batches running through the year, and we have to input all the details of them. When the quantity system data increasing, timetable generation process also gradually increasing and fitness value of the generated timetable is decreasing. As a result, additional manual work has to be handled. However, resource optimization phase is satisfied.

Even, it is set up under probability theory; sometimes it supplies not the optimal but the best timetable schedule. It was a great achievement. See appendix A for the best schedule which reached the fitness value as 1

The significance on evaluating the system is described through this system. By examine the expected output and the actual output we can expect that the system is behaving well. The main objectives such as resource optimization issues and the time wasting problem were successfully achieved through this automated timetable management system to the IT faculty at University of Moratuwa.

## 8.5 Summery

This chapter discussed about the evaluation and the testing the system according to the aim and objectives defined, results of the test cases and the questionnaires given to the admin of the timetable staff and the course coordinator. Next chapter will describe the conclusion of the research project and further work can be implemented.

# Conclusion and Further Work

### 9.1 Introduction

Utilization of Timetable Management System to Faculty of IT is successfully achieved throughout this research and accomplished tasks by the TMSFIT as follows.

- Generation of the timetable management system
- Resource optimization
- View the resources available

### 9.2 Conclusion

Utilization of Timetable Management System to Faculty of IT at University of Moratuwa is successfully achieved through this research process. From a thorough study of literature survey a proper algorithm could find as Genetic Algorithm. According to Prof Kaunananda, Genetic Algorithm can be applied to problems which are unsolvable [19]. The proposed timetabling system for this project seeks to generate near optimal timetables using the principles of genetic algorithm (selection, mutation and crossover). It is easily understandable, efficient and automated system, which is helpful for authorities of the IT faculty, lecturers and students. In addition, it is less paper work.

I have pleased to state; that I could get experience in Yii framework and the MVC design patterns. As a result, now I can develop any Yii component by my own. By following, this research my patience quality increased. For example, I spent number of days to find a searching algorithm. Writing about literature review and find the theories of that. Find information about research methodology and writing of interim document also interesting. There, I could learn about number of algorithms, about knowledgebase systems, searching methods, technologies used by others and Genetic algorithm and Programming languages such as PHP and DBMS such as MYSQL.

As conclusion, the generating timetables and viewing available resources tasks are successfully accomplished by the system within five minutes.

### **9.3 Further Work**

Even this solution is ideal for a middle-sized campus as Moratuwa, large universities, cannot get the best results due to complexity of their problem in case of fitness value is getting decrease. Therefore, I propose those universities can use a combination of ruled base and Genetic Algorithm solution to develop such kind of system.

### **9.4 Summery**

This chapter provided a conclusion of overall achievement met through the research project called utilization of timetable management system to faculty of IT at University of Moratuwa and further work in future as an enhancement of the current solution.

# References

- [1] E. Burke and P. De Causmaecker, Eds., *Practice and theory of automated timetabling IV: 4th international conference, PATAT 2002, Gent, Belgium, August 21-23, 2002: selected revised papers*. Berlin ; New York: Springer, 2003.
- [2] "timetable Meaning in the Cambridge English Dictionary." [Online]. Available: <http://dictionary.cambridge.org/dictionary/english/timetable>. [Accessed: 10-Mar-2016].
- [3] "Mimosa - Scheduling Software for School and University Timetables." [Online]. Available: <http://www.mimosasoftware.com/>. [Accessed: 08-Mar-2016].
- [4] L. Carpenente, A. Cerdeira-Pena, G. de Bernardo, and D. Seco, "An Integrated System for School Timetabling.," in *ICAART (1)*, 2011, pp. 599–603.
- [5] University of Moratuwa, Faculty of Information Technology - Student Handbook, 2013th ed.
- [6] N. Pillay, "A survey of school timetabling research," *Ann. Oper. Res.*, vol. 218, no. 1, pp. 261–293, Jul. 2014.
- [7] A. Dammak, A. Elloumi, H. Kamoun, and J. A. Ferland, "Course timetabling at a Tunisian University: A case study," *J. Syst. Sci. Syst. Eng.*, vol. 17, no. 3, pp. 334–352, Sep. 2008.
- [8] H. Cambazard, F. Demazeau, N. Jussien, and P. David, "Interactively solving school timetabling problems using extensions of constraint programming." in *Practice and Theory of Automated Timetabling V*, Springer, 2004, pp. 190–207.
- [9] S. Hooshmand, M. Behshameh, and O. Hamidi, "A Tabu Search Algorithm With Efficient Diversification Strategy for High School Timetabling Problem," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 4, pp. 21–34, Aug. 2013.
- [10] E. K. Burke and S. Petrovic, "Recent research directions in automated timetabling," *Eur. J. Oper. Res.*, vol. 140, no. 2, pp. 266–280, 2002.
- [11] E. K. Burke, D. G. Elliman, and R. Weare, "A university timetabling system based on graph colouring and constraint manipulation," *J. Res. Comput. Educ.*, vol. 27, no. 1, pp. 1–18, 1994.
- [12] R. Qu, E. K. Burke, B. McCollum, L. T. G. Merlot, and S. Y. Lee, "A survey of search methodologies and automated system development for examination timetabling," *J. Sched.*, vol. 12, no. 1, pp. 55–89, Oct. 2008.
- [13] A. Colomi, M. Dorigo, and V. Maniezzo, "A genetic algorithm to solve the timetable problem," *Politec. Milano Milan Italy TR*, pp. 90–060, 1992.
- [14] C.-H. Chen, T.-K. Liu, and J.-H. Chou, "A Novel Crowding Genetic Algorithm and Its Applications to Manufacturing Robots," *IEEE Trans. Ind. Inform.*, vol. 10, no. 3, pp. 1705–1716, Aug. 2014.
- [15] J. J. Moreira, "A system for automatic construction of Exam Timetable using Genetic Algorithms," *Rev. Estud. Politécnicos Polytech. Stud. Rev.*, vol. 6, no. 9, 2008.



- [16] O. I. Obaid, M. Ahmad, S. A. Mostafa, and M. A. Mohammed. "Comparing performance of genetic algorithm with varying crossover in solving examination timetabling problem," *J Emerg Trends Comput Inf Sci*, vol. 3, pp. 1427–1434, 2012.
- [17] E. Cantú-Paz, *Efficient and accurate parallel genetic algorithms*. Boston, Mass: Kluwer Academic Publishers, 2000.
- [18] B. Sigl, M. Golub, and V. Mornar, "Solving timetable scheduling problem using genetic algorithms," in *Proc. of the 25th int. conf. on information technology interfaces*, 2003, pp. 519–524.
- [19] Professor Ashoka Karunananda Bsc., MPhil, PhD, *Artificial Intelligence*, 2004.05 ed. Tharanji Prints, Highlevel Road, Nawinna, Maharagama, 2004.
- [20] "Free trial : timetable software ... for timetabling school timetables ... for easier scheduling." [Online]. Available: <http://timetabler.com/>. [Accessed: 19-Apr-2016].
- [21] "Open Course Timetabler 0.8.1 - Free download." [Online]. Available: <http://open-course-timetabler.soft112.com/>. [Accessed: 19-Apr-2016].
- [22] "About Yii | Yii PHP Framework." [Online]. Available: <http://www.yiiframework.com/about/>. [Accessed: 12-Mar-2016].
- [23] "Take The Tour | Yii PHP Framework." [Online]. Available: <http://www.yiiframework.com/tour/#>. [Accessed: 12-Mar-2016].
- [24] "Fundamentals: Model-View-Controller (MVC) | The Definitive Guide to Yii | Yii PHP Framework." [Online]. Available: <http://www.yiiframework.com/doc/guide/1.1/en/basics.mvc>. [Accessed: 12-Mar-2016].
- [25] "Getting Started: Installation | The Definitive Guide to Yii | Yii PHP Framework." [Online]. Available: <http://www.yiiframework.com/doc/guide/1.1/en/quickstart.installation#requirements>. [Accessed: 13-Mar-2016].

## Interfaces of the TMSFIT

Before saving the Timetable which the fitness value = 1

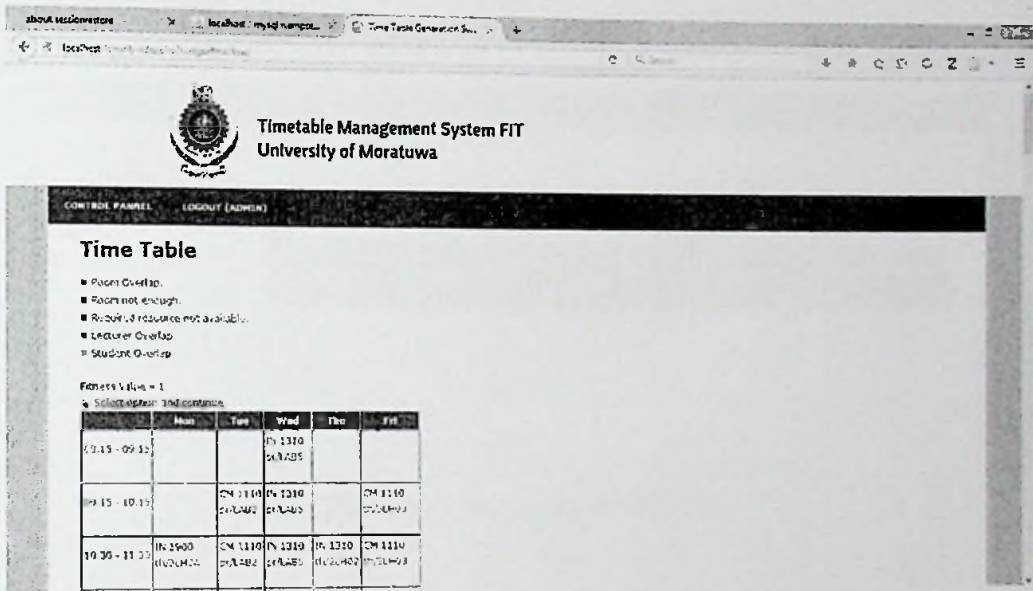


Figure A-1 Generated Timetable Before Saved

## Error Messages given by the TMSFIT

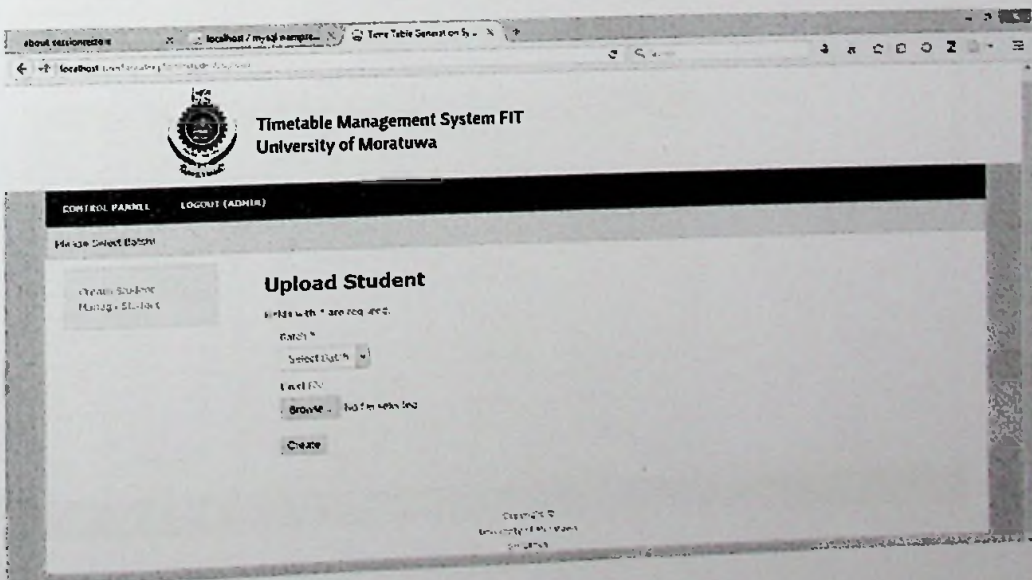


Figure A-2 Error Message

# Manage Students Interface

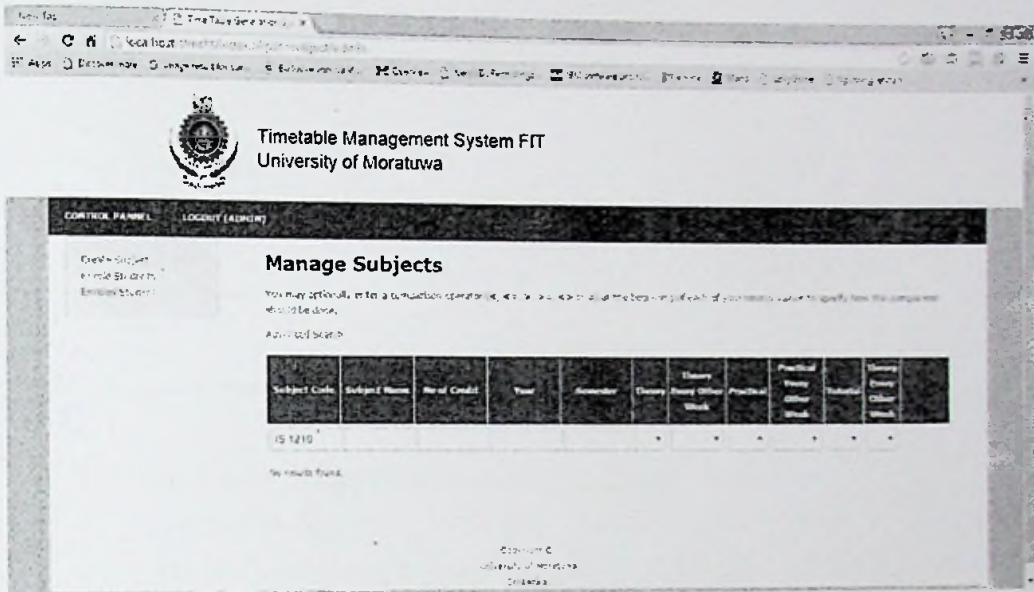


Figure A- 3 Manage Student Interface

## Preview of tables

Host: mysql.wampserver  
 Database: uni  
 Generation Time: Mar 13, 2016 at 03:45 AM  
 Generated by: phpMyAdmin: 4.1.14 - MySQL: 5.6.17  
 SQL query: SELECT \* FROM `student` LIMIT 0, 25.  
 Rows: 25

id	fname	lname	gender	dateofbirth	address	contact	email	userid	regno	regyear	course
41	H M S U	ABEYRATHNE	F	2034-00-04	Gampaha	332224569	test1@yahoo.com	50	1340217	2	1
42	H M A	ABEYWARDANA H M A	F	2034-00-09	Gampaha	332224569	test2@yahoo.com	51	1340218	2	1
43	S L S	AHAMADH	F	2034-00-06	Gampaha	332224569	test3@yahoo.com	52	1340219	2	1
44	E V K	ALWIS	F	2034-00-09	Gampaha	332224569	test4@yahoo.com	53	1340220	2	1
45	M Z F	AMARA	F	2034-00-08	Gampaha	332224569	test5@yahoo.com	54	1340221	2	1
46	G K S M	AMARASINGHE	F	2034-00-09	Gampaha	332224569	test6@yahoo.com	55	1340222	2	1
47	S M A	GUNAWARDANA	M	2034-01-00	Gampaha	332224569	test7@yahoo.com	56	1340223	2	1
48	M W L	ASANGA	F	2034-01-01	Gampaha	332224569	test8@yahoo.com	57	1340224	2	1
49	A S W E A	BANDARA	F	2034-01-02	Gampaha	332224569	test9@yahoo.com	58	1340225	2	1
50	M M C M	BANDARA	M	2034-01-03	Gampaha	332224569	test10@yahoo.com	59	1340226	2	1
51	P P D M	BANDARA	M	2034-01-04	Gampaha	332224569	test11@yahoo.com	60	1340227	2	1
52	R M U S	BANDARA	M	2034-01-05	Gampaha	332224569	test12@yahoo.com	61	1340228	2	1
53	V D	CHANARA	M	2034-01-06	Gampaha	332224569	test13@yahoo.com	62	1340229	2	1
54	C L R	CHANDRANATHA	M	2034-01-07	Gampaha	332224569	test14@yahoo.com	63	1340230	2	1
55	D	DANDENTIA	F	2034-01-08	Gampaha	332224569	test15@yahoo.com	64	1340231	2	1

Figure A-4 Table Preview

## Code segments of the TMSFIT

## Sample code for batch form

```

<?php
/* @var $this BatchController */
/* @var $model Batch */
/* @var $form CActiveForm */
?>

<div class="form">

<?php $form=$this->beginWidget('CActiveForm', array(
    'id'=>'batch-form',
    // Please note: When you enable ajax validation, make sure the
corresponding
    // controller action is handling ajax validation correctly.
    // There is a call to performAjaxValidation() commented in
generated controller code.
    // See class documentation of CActiveForm for details on this.
    'enableAjaxValidation'=>false,
)); ?>

    <p class="note">Fields with <span class="required">*</span> are
required.</p>

    <?php echo $form->errorSummary($model); ?>

    <div class="row">
        <?php echo $form->labelEx($model, 'name'); ?>
        <?php echo $form-
>textField($model, 'name', array('size'=>50, 'maxlength'=>50)); ?>
        <?php echo $form->error($model, 'name'); ?>
    </div>

    <div class="row">
        <?php echo $form->labelEx($model, 'description'); ?>

```

```

        <?php echo $form-
>textField($model, 'description', array('size'=>60, 'maxlength'=>200));
?>

        <?php echo $form->error($model, 'description'); ?>
    </div>

    <div class="row">
        <?php echo $form->labelEx($model, 'startyear'); ?>
        <?php echo $form-
>textField($model, 'startyear', array('size'=>4, 'maxlength'=>4)); ?>
        <?php echo $form->error($model, 'startyear'); ?>
    </div>

    <div class="row buttons">
        <?php echo CHtml::submitButton($model->isNewRecord ?
'Create' : 'Save', array('class'=>'btn btn-primary')); ?>
    </div>

<?php $this->endWidget(); ?>

</div><!-- form -->

```

## Calculate the fitness of the chromosome

```

    //Calculate fitness value for chromosome
    public function calFitness(){
        // chromosome's score
        $score = 0;

        $numberOfRooms = $this->_noOfClassRoom;
        $roomDaySize = Schedule::DAY_HOURS * $numberOfRooms;

        $sci = 0;
        // check criterias and calculate scores for each class in
        schedule
        foreach($this->_classes as $value)
        {
            // coordinate of time-space slot
            $p = $value[0]; //slot number
            $day = (int)($p / $roomDaySize);

```

```

$time = $p % $roomDaySize;
$room = (int)($time / Schedule::DAY_HOURS);
$time = $time % Schedule::DAY_HOURS;
// echo "<br>rooms <del></del>". $room." // slot = ". $p ;
$dur = round($value[1]->duration, 0, PHP_ROUND_HALF_UP);
// check for room overlapping of classes
$ro = false;
for( $i = $dur - 1; $i >= 0; $i-- )
{
    if( array_key_exists ((int)($p + $i), $this->_slots)
    && sizeof($this->_slots[ (int)($p + $i) ]) > 1 )
    {
        $ro = true;
        break;
    }
}
// on room overlapping
if( !$ro ){
    $score++;
    // echo "<br>no room overlapping";
}
$this->_criteria[ (int)($ci + 0) ] = !$ro;
$cc = $value[1];
$r = $this->_allClassRoom[$room];
// does current room have enough seats
$roomEnough = false;
if($r->capacity >= count($cc->students)){
    $roomEnough = true;
}
$this->_criteria[ (int)($ci + 1) ] = $roomEnough;
if($roomEnough ){
    $score++;
    // echo "<br>Room enough";
}

// does current room have computers if they are required
$labRequired = false;
if($cc->reqLab){
    $labRequired = true;
}
}
$resourceType = false;

```

```

//      echo "<br>labRequired ".$labRequired ;
//      echo "<br>type>>> ".$r->type ;
      if( $labRequired ){

          if($r->type == 'lab'){
              $score++;
              $resourceType = true;
//          }
//          }

          }else{
              if($r->type != 'lab'){
                  $score++;
                  $resourceType = true;
//              }
//              }

          }else{
//              }
//              }

          $this->_criteria[(int) ($ci + 2) ] = $resourceType;
          set_time_limit(20);
          $po = false;
          $go = false;
//      check overlapping of classes for professors and student groups
          for( $ii = $numberOfRooms, $t = $day * $roomDaySize +
$time; $ii > 0; $ii--, $t += Schedule::DAY_HOURS )
          {
//      for each hour of class
              for( $i = $dur - 1; $i >= 0; $i-- )
                  {
//      check for overlapping with other classes at same time
                      if (array_key_exists((int)($t + $i), $this->_slots) && isset($this->_slots[($t + $i)]) {
//echo "slot"-----
>". ($t + $i);
                          $classes = $this->_slots[ $t + $i ];

foreach ($classes as $cls){

```

```

                                if( $cc != $cls )
                                {
// professor overlaps?
if( !$po && $cc->lecturerOverlaps( $cls ) )
                                $po = true;

// student group overlaps?
if( !$go && $cc->studentOverlap( $cls ) )
                                $go = true;

// both type of overlapping? no need to check more
                                if( $po && $go )
                                goto total overlap;
                                }
                                }
                                }
}

```

total\_overlap:

```

// professors have no overlapping classes?
if( !$po ){
$lectpref = $cc->checkLectPrefTime($cc->lecturer-
>id,$day,$time,$dur,$cc->semester,$cc->year);
if($lectpref){
    $score++;
}else {
    $po = true;
}
}

```

\$this->\_criteria[ \$ci + 3 ] = !\$po;

```

// student groups has no overlapping classes?
if( !$go ){
    $score++;
    echo "<br>Student overlap";
}

```



```

        $this->_criteria[ $ci + 4 ] = !$go;

        $ci += 5;

    //

    }

    // calculate fitness value based on score
    $this->_fitness = $score / ( count($this->_subjectClass) *
Schedule::DAYS_NUM );

}

```

### Cross over operation

// Performes crossover operation using 2 chromosomes and returns 2 offspring

```

public function crossover($parent2){
    // check probability of crossover operation
    if( rand() % 100 > $this->_crossoverProbability ){
        // no crossover, just copy first parent
        return $this->makeCopy(false );
    }

```

//echo

```

"<br><br>crossover=====
===<br>";

```

```

// new chromosome object, copy chromosome setup
$n = $this->makeCopy(true );

```

```

// number of classes
$size = (int)count($this->_classes);

```

```

$cp = array();

```

```

// determine crossover point (randomly)
for( $i = $this->_numberOfCrossoverPoints; $i > 0; $i-- )
    {
        while( true )
            {
                $p = rand() % $size;

```

```

        if(!array_key_exists($p, $cp))
        {
            $cp[ $p ] = true;
            break;
        }
    }
}

$it1 = $this->_classes;
$it2 = $parent2->_classes;
$pkindex1 = 0;
$pkindex2 = 0;
// make new code by combining parent codes
$first = rand() % 2 == 0;
for( $i = 0; $i < $size; $i++ )
{
    if( $first )
    {
// insert class from first parent into new chromosome's class table
        $n->_classes[$pkindex1] =
array($it1[$pkindex1][0],$it1[$pkindex1][1]);

// all time-space slots of class are copied
for( $ii = round($it1[$pkindex1][1]->duration, 0, PHP_ROUND_HALF_UP)
- 1; $ii >= 0; $ii-- ){

if($n->_slots[(int)($it1[$pkindex1][0] + $ii )] == null){
//if array is null then initialize new array and add class
$n->_slots[(int)($it1[$pkindex1][0] + $ii )] =
array($it1[$pkindex1][1]);
}else{
$array = $n->_slots[(int)($it1[$pkindex1][0] + $ii )];
$array[] = $it1[$pkindex1][1];
$n->_slots[(int)($it1[$pkindex1][0] + $ii )] = $array;
        }
    }
}
    }
else
    {
// insert class from second parent into new chromosome's class table

```

```

$n->_classes[$pkindex1] =
array($it2[$pkindex2][0],$it2[$pkindex2][1]);

// all time-space slots of class are copied
for( $ii = round($it2[$pkindex2][1]->duration, 0, PHP_ROUND_HALF_UP)
- 1; $ii >= 0; $ii-- ){
if($n->_slots[(($it2[$pkindex2][0] + $ii )] == null){
//if array is null then initialize new array and add class
$n->_slots[(($it2[$pkindex2][0] + $ii )] = array($it2[$pkindex2][1]);
}else{
$array = $n->_slots[(($it2[$pkindex2][0] + $ii )];
$array[] = $it2[$pkindex2][1];
$n->_slots[(($it2[$pkindex2][0] + $ii )] = $array;
}
}
}

// crossover point
if( array_key_exists($i, $cp)){
// change source chromosome
    $first = !$first;
}

    $pkindex1++;
    $pkindex2++;

}
    $n->calFitness();
//
    echo
"<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<br>";
    return $n;

```

### Mutation code segment

```

// Performs mutation on chromosome
    public function mutation()
    {
//echo "<br>.....mutation.....";
// check probability of mutation operation
if( rand() % 100 > $this->_mutationProbability )
    return;

```

```

// number of classes
$numberOfClasses = (int)count($this->_classes);
// number of time-space slots
$size = (int)count($this->_slots);
// move selected number of classes at random position
for( $ii = $this->_mutationSize; $ii > 0; $ii-- )
    {
// select random chromosome for movement

$mpos = rand() % $numberOfClasses;
$pos1 = 0;
$classeIndex = array_keys($this->_classes);

// current time-space slot used by class
$pos1 = $this->_classes[$mpos][0];

$scl = $this->_classes[$mpos][1];

// determine position of class randomly
$nr = $this->_noOfClassRoom;
$dur = round($scl->duration, 0, PHP_ROUND_HALF_UP);
        $day = rand() % Schedule:::DAYS_NUM;
        $room = rand() % $nr;
        $time = rand() % ( Schedule:::DAY_HOURS + 1 - $dur );
$pos2 = $day * $nr * Schedule:::DAY_HOURS + $room *
Schedule:::DAY_HOURS + $time;
// move all time-space slots
    for( $i = $dur - 1; $i >= 0; $i-- )
        {
// remove class hour from current time-space slot
            $cl = $this->_slots[ $pos1 + $i ];
            foreach ( $cl as $key=>$cls ){
                if( $cls == $scl )
                    {
                        unset($cl[$key]);
                        break;
                    }
            }
        }

// move class hour to new time-space slot
if($this->_slots[(int)($pos2 + $i)] == null){
//if array is null then initialize new array and add class

```

```

$this->_slots[(int)($pos2 + $i) ] = array($scl);

        }else{
$array = $this->_slots[(int)($pos2 + $i) ];
        $array[] = $scl;
        $this->_slots[(int)($pos2 + $i) ] = $array;

        }

    }

// change entry of class table to point to new time-space slots
        unset($this->_classes[$mpos]);
        $this->_classes[ $mpos ] = array($pos2,$scl);
    }

    $this->calFitness();
}
}

```

## Dashboard code segment

```

<?php
/* @var $this DashboardController */

// $this->breadcrumbs=array(
//     'Dashboard',
// );
?>
<div class="dashboard">
    <div class="dashboard-wrapper">

        <!-- Dashboard for Administrators -->
        <?php if (Yii::app()->user->getState('roles') ==
"admin"): ?>
        <!--
            <div class="dash-icon">
                <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('candidate/admin'); ?>">
                    

```

```
        <div class="link-text">New Applicants</div>
    </a>
</div> -->
```

```
    <div class="dash-icon">
        <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('degree/admin'); ?>">
            
            <div class="link-text">Degrees</div>
        </a>
    </div>
```

```
    <div class="dash-icon">
        <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('student/admin'); ?>">
            
            <div class="link-text">Students</div>
        </a>
    </div>
```

```
    <div class="dash-icon">
        <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('employee/admin'); ?>">
            
            <div class="link-text">Lecturer</div>
        </a>
    </div>
```

```
    <div class="dash-icon">
        <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('subject/admin'); ?>">
            
            <div class="link-text">Subjects</div>
        </a>
    </div>
```

```

        <div class="dash-icon">
            <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('resource/admin'); ?>">
                
                <div class="link-text">Resources</div>
            </a>
        </div>

        <div class="dash-icon">
            <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('batch/admin'); ?>">
                
                <div class="link-text">Batch</div>
            </a>
        </div>

        <div class="dash-icon">
            <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('algorithm/load'); ?>">
                
                <div class="link-text">Time Table</div>
            </a>
        </div>

    <?php endif; ?>
    <!-- Lecturer Dashboard -->
    <?php if (Yii::app()->user->getState('roles') ==
"lecturer"): ?>
        <div class="dash-icon">
            <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('employee/time', array('id'=>Yii::app()->user-
>getState('loggeduserid'))); ?>">
                
                <div class="link-text">Preferred Time</div>
            </a>
        </div>

```

```

<!-- <div class="dash-icon"> -->
<!-- <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('employee/timetable'); ?>">
 -->
<!-- <div class="link-text">Time Table</div> -->
<!-- </a> -->
<!-- </div> -->

<div class="dash-icon">
    <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('employee/view', array('id'=> Yii::app()->user-
>getState('loggeduserid'))); ?>">
        
        <div class="link-text">Profile</div>
    </a>
</div>

<div class="dash-icon">
    <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('employee/timetable'); ?>">
        
        <div class="link-text">Time Table</div>
    </a>
</div>

<?php endif; ?>

<!-- Dashboard for Students -->
<?php if (Yii::app()->user->getState('roles') ==
"student"): ?>
<div class="dash-icon"><a class="dash-link" href="<?php echo
Yii::app()->createUrl('student/view', array('id'=> Yii::app()->user-
>getState('loggeduserid'))); ?>"> 
        <div class="link-text">Profile</div>

```



```
        </a>
    </div>

    <div class="dash-icon">
        <a class="dash-link" href="<?php echo Yii::app()-
>createUrl('student/timetable'); ?>">
            
            <div class="link-text">Time Table</div>
        </a>
    </div>

    <?php endif; ?>
</div>
</div>
```

# Testing and Evaluation with Test data

## Sample Test cases for Black box Testing

Test No	Test Data	Expected Results	Actual Results	Conclusion
1	Logging with incorrect user name	Operation must be rejected	Message - Incorrect Username or password	Achived
2	Logging with incorrect password	Operation must be rejected	Message - Incorrect Username or password	Achived
3	Logging with incorrect user name and password	Operation must be rejected	Message - Incorrect Username or password	Achived
4	Logging with empty user name	Operation must be rejected	Message - Incorrect Username or password	Achived
5	Logging with empty password	Operation must be rejected	Message - Incorrect Username or password	Achived
6	Logging with empty Username and password	Operation must be rejected	Message - Incorrect Username or password	Achived
7	Search with incorrect name	Empty results	Message - No results found	Achived

8	Input Empty Field Name to a field with astric	Operation must be rejected	Message - Name cannot be blank.	Achived
9	Input value to a field with astric	Operation must be success	No error Message	Achived
10	Input submit number of Empty Fields with astric	Operation must be rejected	Message - Name cannot be blank.	Achived
11	Search with correct No	Operation success	Show the searched data	Achived
12	Search with correct batch No	Operation success	Show the searched data	Achived
13	Advanced Search with one the data Name Perera	Operation success	Show all the searched name with Perea	Achived
14	Update student without registration No	Operation success	Message- Refistration No cannot be blank	Achived

Table C -1 Test Cases

# White Box Testing with Understand Software

## Analysis Log

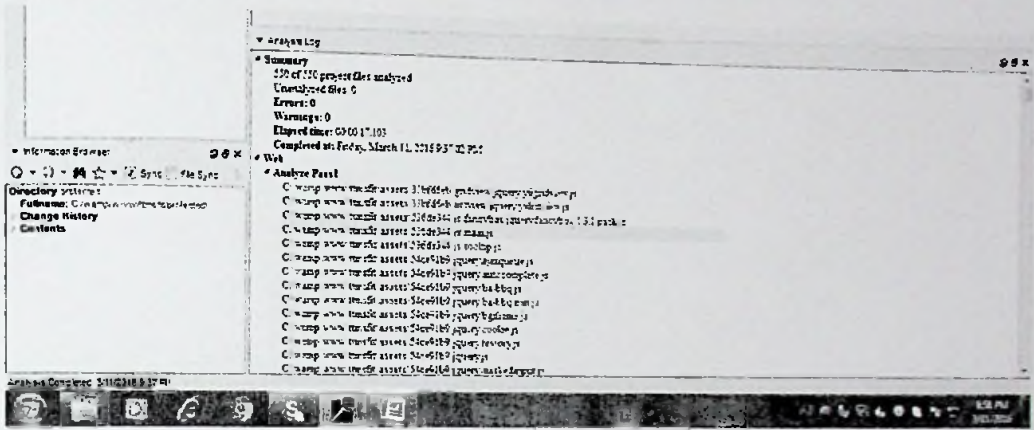
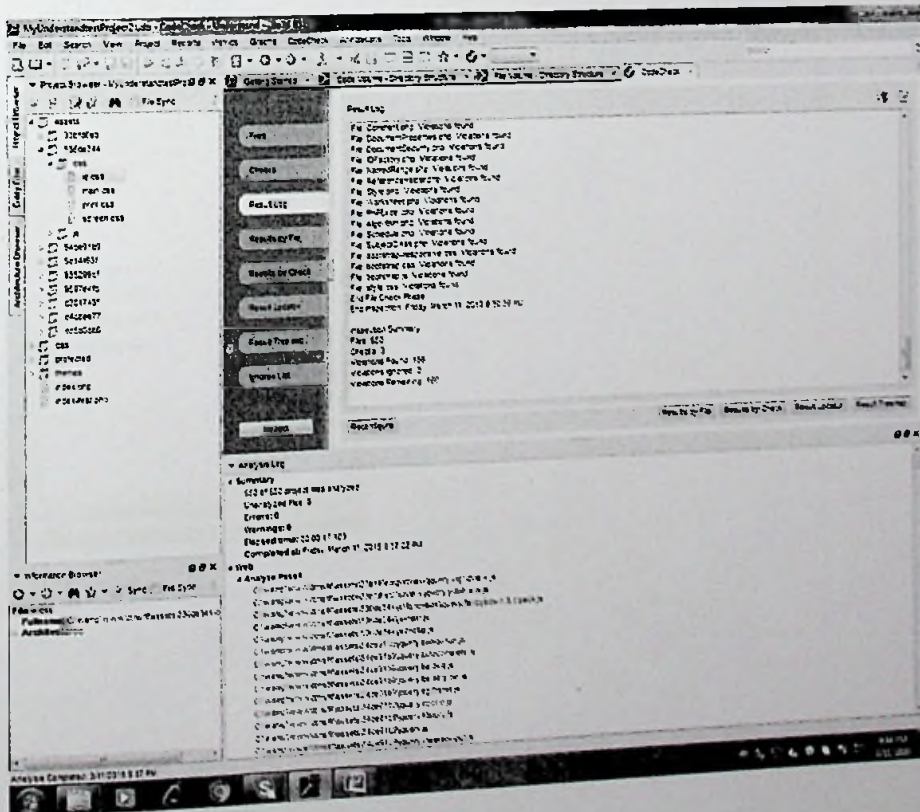
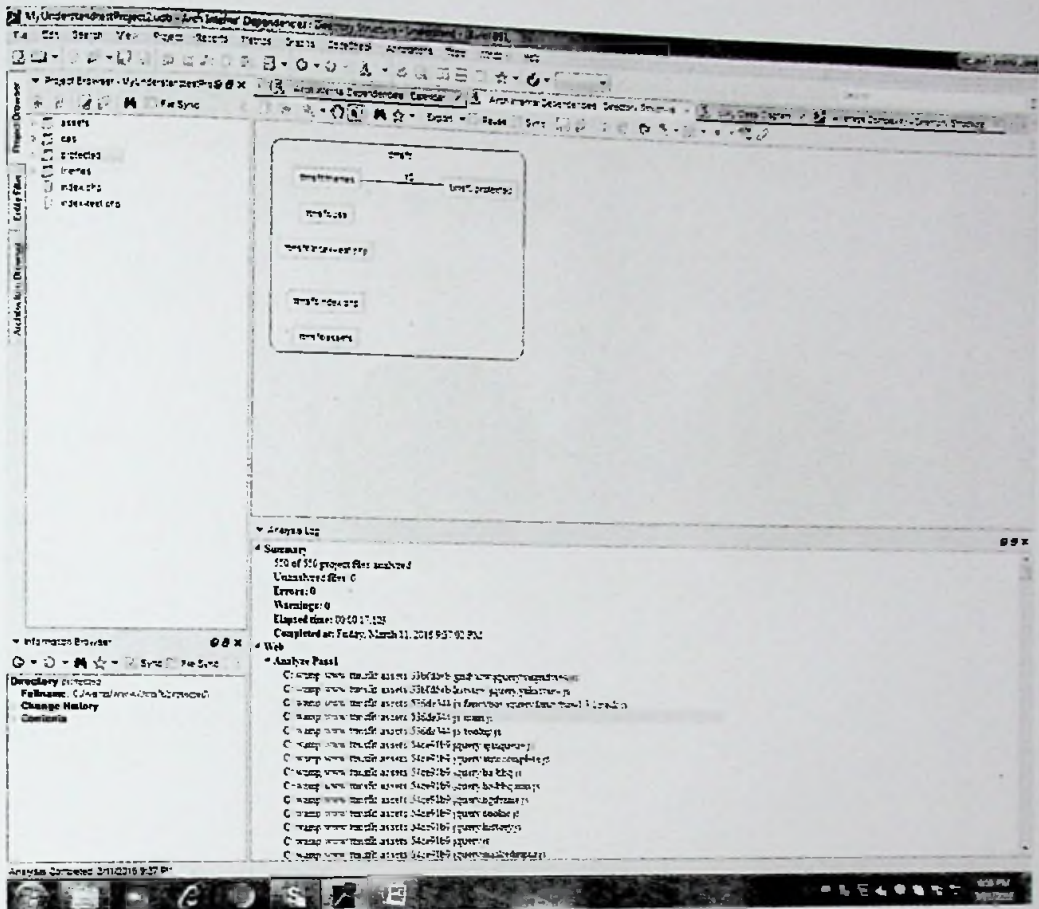


Figure C-1 White box Testing

## Results log



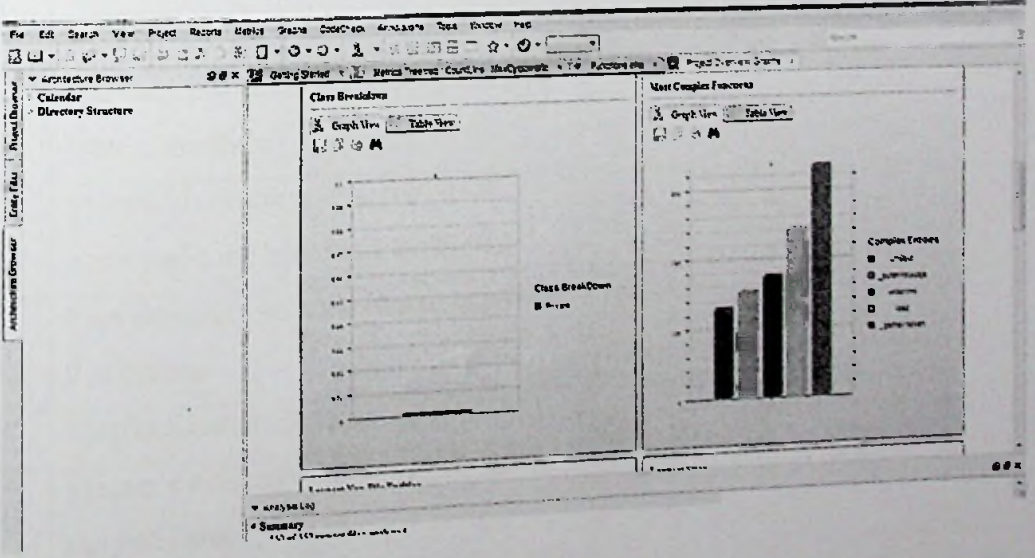
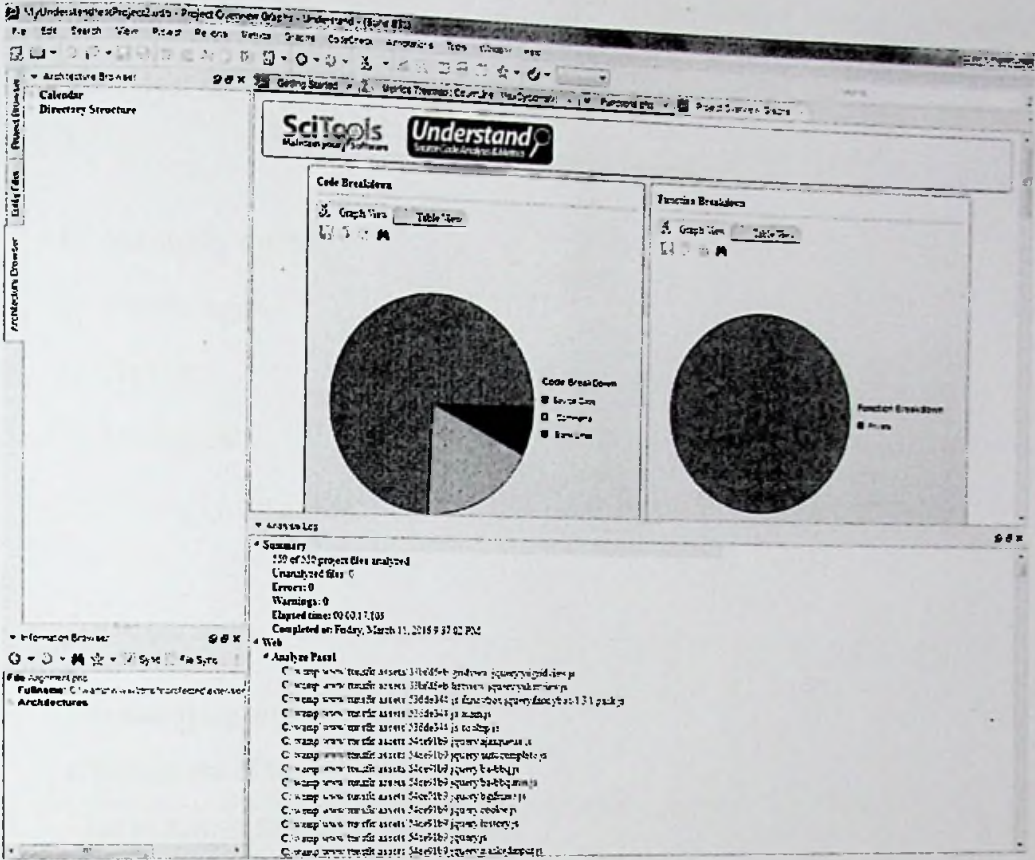


## Metric Summary

The 'Metric Summary' dialog box displays the following data:

Metric	Value
Blank Lines	21,266
Classes	230
Code Lines	213,852
Comment Lines	27,284
Comment to Code R.	0.24
Declarative Statements	17,873
Executable Statements	66,201
Files	370
Functions	12,516
Lines	302,271

Buttons: Copy All, What do the metric values mean?, Close



# Evaluation Questionnaire

## Evaluation of the Timetable Management System of Faculty of IT at University of Moratuwa

1. Strongly agree
2. Fairly agree
3. Agree
4. Disagree
5. Strongly disagree

		1	2	3	4	5	NA
1	I'm satisfying with the feature of effective use of time						
2	This system is simple and easy to use						
3	Interfaces of the system are good and attractive						
4	It is easy to search information from this system						
5	It is easy to recover and correct from the system mistakes						
6	System gives error messages which clearly tells me how to fix problems						
7	I am satisfying with the functionalities of this system						
8	System functionalities are user friendly						
9	System's manually changing part also, can be tolerate						
10	Overall, I am satisfied with TMSFIT						

Table C-2 Questionnaire Evaluation Table