DESIGN AND IMPLEMENTATION OF ANDROID BASED TIMETABLE GENERATION SYSTEM USING PARTICLE SWARM OPTIMIZATION (PSO) METHOD

BY

DANJUMA, Hussaini Joshua

2015/1/55157CT

DEPARTMENT OF COMPUTER SCIENCE
FEDERAL UNIVERSITY OF TECHNOLOGY
MINNA

AUGUST, 2021

DESIGN AND IMPLEMENTATION OF ANDROID BASED TIMETABLE GENERATION SYSTEM USING PARTICLE SWARM OPTIMIZATION (PSO) METHOD

BY

DANJUMA, Hussaini Joshua

2015/1/55157CT

PROJECT SUMBITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY, MINNA, NIGEREIA IN PARITIAL FULFILLMRNT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE

AUGUST, 2021

DECLARATION

I hereby declare that this project titled "Design and Implementation of Android Based Timetable Generation System Using Particle Swarm Optimization (PSO) Method" is a collection of my original work and it has not been presented for any other qualification anywhere. Information from other source (published or unpublished) has been duly acknowledged.

DANJUMA, Hussaini Joshua

2015/1/55157CT

Signature and Date

Federal University of Technology,

Minna, Nigeria

CERTIFICATION

This project titled "Design and Implementation of Android Based Timetable Generation System Using Particle Swarm Optimization (PSO) Method" by DANJUMA, Hussaini Joshua (2015/1/55157CT) meets the regulations governing the award of Bachelor of Technology (B. Tech) in Computer Science of the Federal University of Technology Minna, Nigeria.

Dr. J. K. ALHASSAN
______________________ ______________________
Project Supervisor Signature and Date

Dr. (Mrs.) Opeyemi Aderike Abisoye
______________________ ______________________
Head of Department Signature and Date

ACKNOWLEDGEMENT

I would like to take this opportunity to express my heartfelt gratitude to God almighty for His love towards me, as well as special thanks to my parents Mr. & Mrs Aboki Danjuma and siblings, for your endless love, prayers, and support,

I would like to thank my supervisor DR. J. K. Alhassan for his encouragement and his selfless nature despite other commitments at his disposal, as well as all past Heads of Department, and the present Head of Department, Dr. (Mrs) Opeyemi Aderike Abisoye, for her valuable and constructive suggestions during the course of my stay in school, planning and development of this project.

I wish to thank various lecturers for their contributions to this project; Dr. Muhammad Bashir ABDULLAHI, Dr. Solomon Adelowo ADEPOJU, Mr. Adamu Muhammad SALIU, Dr. Mohammed Danlami ABDULMALIK, Dr. Sulaimon Adebayo BASHIR, Dr. Oluwaseun Adeniyi OJERINDE, Mr. Cosmas Uchenna UGWUOKE, Mr. Ibrahim Shehi SHEHU, Mr. Enesi Femi AMINU, Mr. Idris Kolo MOHAMMED, Mr. Adesuyi Adeyinka FALAYE, Mr. Kudu Muhammad MUHAMMAD, Mr. Victor Ndako ADAMA, Mr. Kehinde Husseini LAWAL, Mrs. Ayobami EKUNDAYO, Mr. Yakubu Boyi-Musa LASOTTE, Mr. Olamilekan Lawal LAWAL and Mr. Alkali Umar SANI for their contributions and immeasurable impact, my supportive. May the Lord bless you all.

Finally, I would like to express my heartfelt gratitude to all my course mates and friends, Job, Priscilla, Sunday, Joshua, Tobi, Maryam, Matthias and many others, who have contributed in some way to my success in the department, you are one in a million. I love you all.

DEDICATION

I dedicate this project first and foremost to God almighty the author finisher of everything who has been there for me from the beginning down to this very point. Special dedication goes to my humble and very supportive parents (Mr and Mr Aboki Danjuma) for their relentless effort and compassion towards me during the course of my five years in school.

ABSTRACT

Timetabling is a vital process in every area. In each educational institution, it is an open-finished program wherein courses should be masterminded around a set of time-slot and remains with the goal that a few constraints are fulfilled. The aim of this work is to utilize Particle Swarm Optimization (PSO) to take care of the timetable scheduling issue which happens from a few issues that have been recognized from the traditional system where timetable scheduling is been handled manually. The methodology utilized is an altered Rapid Application Development (RAD) method. From results the system gives average response time of 1.07 seconds; effectiveness of 100% and efficiency of 0.82 seconds. The new versatile algorithm dependent on PSO and applied to make exceptionally proficient timetable. Because of the intricacy of PSO Algorithm, it ought to be instructed as a course for Computer Science students to empower students have better comprehension of PSO Algorithm.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

Students, faculties, and administrators can get information about their schedule using the University Time Table system also it can assign different jobs to different faculties, Using the Federal University of Technology Minna (FUT), School of Information and Communication Technology (SICT), Department of Computer Science as a case study, this research looks at the school's timetabling problem and provides a solution.

Timetabling is the process of planning an action or series of actions that must be independent of time constraints. The optimization of an operation or actions in an expected time to satisfy a set of acceptable constraints is described as timetabling. Short-term planning, such as daily and weekly schedules, as well as long-term planning, such as monthly and yearly schedules, are covered by timetables (Raju et Mallesh, 2016). For students' convenience, the timetable lists the number of courses and venues that have been assigned to each course. The data in the timetable can only be viewed, added to, and deleted by the administrator. In recent years, timetabling issues have been uncovered, and there has certainly been an increase in study work in this field. Timetabling is necessary to ensure that the flow of the schedule is smooth and successful and that there are no time conflicts between events (Chuen, 2015).

The problem of course scheduling studied in this work deals with resource allocation restrictions; under restricted resources, settings regarding lecturers, students, courses, and instructional facilities must be met. The limits differ depending on the college's atmosphere, system, and other variables. Constraints are usually classified as either hard or soft and soft constraints. Hard constraints are the ones that have to be met and must not be violated while the soft ones should be met but can be violated. The following are some of the hard restrictions considered in this study:

(1) In the same time slot, each lecturer can only teach one course (teaching two courses at the same time is not permitted).

(2) In the same time frame, each venue can only be utilized for one course (teaching two courses in the same venue at the same time is not allowed).

(3) No lectures in the fourth period each day because it’s the time for the school lunch.

(4) Each lecture should terminate at the fourth hour every Wednesday.

In addition, the following are the soft restrictions that have been taken into account:

(1) The available or desired day and periods for the courses delivered by the provided lecturer(s) should be assigned.

(2) Course schedule should be done following professors' and students' preferred timeslots, as well as their levels of preference.

(3) No lecturer may teach more than the department's maximum number of hours.

(4) No more than two events in a row per day (Chen and Shih, 2013).

J. Kennedy and R. Eberhart introduced the Particle Swarm Optimization (PSO) technique, which is a self-adaptive global search optimization technique. PSO is an Artificial intelligence (AI) procedure to find an approximate solution in a numeric maximization and minimization issues. PSO is undoubtedly displayed after the idea of bird running and fish schooling. The idea of PSO is only that it is joining to worldwide with a practical answer for a specific issue, and it is executed using the stochastic nature of the particle. Attributable to its viability in a wide scope of utilizations, ease, and minimal expense, PSO is notable and inescapable AI field

(Chuen, 2015). Various methodologies from tasks research, for example, graph colouring, numerical programming, local search measures like tabu search and simulated annealing, genetic algorithm, and particle swarm optimization, can be utilized to handle timetabling challenges.

Albeit most of the managerial work at FUT Minna has been computerized, because of specialized issues, the timetable is still chiefly done physically. Physically organizing of lecture timetable takes a ton of time and work. The timetabling issue is a limitation satisfaction issue in which we should find the best arrangement that meets the entirety of the necessities. We should choose some schedule openings and talk settings that satisfy the restrictions set on the courses, educators, lecture venues accessible to take care of the talk timetabling issue. Accordingly, the factors to be launched are the given time slots and lecture venues of offered courses. The significant objective of making the Timetable application is to make an excellent and reasonable timetable that will work on the school's current construction. Another objective is for the framework to be easy to understand and clear to keep up with. The framework made ought to be easy to alter to oblige any limitations that might emerge in a genuine circumstance. The framework ought to likewise be dependable and fit for producing a feasible answer for any significant issue.

1.2 Statement of the Problem

The desire to develop an automated timetable scheduling system stems from several issues that have been identified with the traditional system, which involves manually scheduling timetables. Every session or semester, the school system must design and manage a timetable, but there are sometimes issues with venue collision when a venue is assigned to many classes at the same time. This type of situation destabilizes both students and lecturers since it either causes students, particularly the class head, to seek another location for the lecture, which is time-consuming, in the event of venue collision, or the lecturer(s) cancels the speaker until a new lecture venue is arranged. It's a challenging process that takes a lot of time and effort.

This project work is intended to ease the challenges identified in the conventional technique, as well as to instil a degree of confidence in the feasibility and usefulness of the created table for the pupils.

1.3 Aim and Objectives

The aim of the project is the Design and Implementation of Android Base-Timetable Generation System using Particle Swarm Optimization Method.

The objectives of this project are to:

i. To generate the PSO algorithm for the system.

ii. To design the system using appropriate design methodology.

iii. To implement the system using a suitable programming language.

iv. Evaluate the system

1.4 Scope and Limitation of the Study

There are several timetabling issues from diverse domains, such as course, test, and exam. This research will focus on the teaching schedule at the Department of Computer Science, SICT, FUT Minna. The data for the study will be gathered from the Computer Science department, and the FUT Minna schedule administrator. The scheduling administrator for the Computer Science department will give information such as room availability, room capacities, lecture duration, and courses available for the lecture.

The timetable is created depending on the venues and courses that are available in a given semester. The procedure is created using a technology known as PSO, which searches for available locations and courses for the timetable.

Because the restrictions between lecture and exam timetables sometimes change, this study is confined to a departmental lecture timetable only. In the school, pay attention to lecture halls and course scheduling. The college exam officer will make the changes.

1.5 Significance of the Study

To minimize mistakes or redundancy, a strong focus, discipline, and logistics are necessary while scheduling timetables, this system replaces the human method with an automated approach that creates appropriate timetable allocations with zero percent collision tolerance. Staff will not have to worry about class redundancy because of this system. Additionally, the time spent allocating and re-allocating venues or courses, which typically causes delays, will be decreased. There will also be less stress for students and lecturers alike while attempting to choose a lecture location because there are no conflicts.

1.6 Project Organization

Project organization clarifies the content that is remembered for each section so the user can get the oversight thought for every part.

In Chapter one: Introduction, an explanation about the venture is formed so client can by and large get some answers concerning the proposed project.

In Chapter two: Literature Review, references are found for the venture and past work done in such fields.

In Chapter three: Methodology, used for the endeavour is discussed to get the construction of the undertaking.

In Chapter four: Design and Implementation, this time of the proposition is to encourage the construction or plan of the undertaking.

In Chapter five: Recommendation and Conclusion gives the general finish of the work and further suggestions for future work.

CHAPTER TWO

LITERATURE REVIEW

2.1 The Emergence of Artificial Intelligence (AI)

Despite the fact that it is hard to distinguish, the fundamental bases of AI may most perhaps be followed back to the 1940s, to be specific 1942, after American Science Fiction creator Isaac Asimov distributed his short story Run-around. The storyline of Run-around rotates around the three laws of mechanical technology, a tale about a robot worked by Gregory Powell and Mike Donavan. Initial, a robot may not harm an individual or, by inertia, permit an individual to be hurt; also, a robot will submit to orders trained by individuals aside from where such orders would conflict with the main law: finally, a robot ought to guarantee its existence given that doing as such doesn't conflict with first or second laws. Asimov's work empowered ages of scientists in mechanical innovation, AI, and computer programming, including the American Intellectual specialist Marvin Minsky.

At about a similar period, yet in excess of 3,000 miles away, the English Mathematician Alan Turing dealt with a code-breaking PC named "The Bombe" for the British government, totally keen on figuring out the Enigma code utilized by the German military during World War II. The Bombe is by and large viewed as the first practical electro-mechanical PC framework. Turing was astonished by the capacity of The Bombe to break the Enigma code, a task that even the best human arithmeticians discovered outlandish, and he pondered about the knowledge of such PCs. "Handling Machinery and Intelligence" was distributed in 1950, and it disclosed how to fabricate scholarly PCs and how to survey their mind. Indeed, even today, the Turing test is utilized to decide whether an individual is talking with a bogus structure when defied with someone else and a machine that can't separate the machine from the human, the machine is supposed to be sharp. At the point when Marvin Minsky and John McCarthy (a Stanford PC researcher) arranged the eight-week Dartmouth summer research project on manmade reasoning (DSRPAI) at Dartmouth College in New Hampshire in 1956, the expression "AI" was authoritatively instituted six years after the fact. This work area, which denoted the beginning of the AI and was supported by the Rockefeller Foundation, re-joined individuals who might thusly be viewed as the primary AI modelers. Individuals included PC specialist Nathaniel Rochester, who later planned the IBM 701, the primary business intelligent PC, and number-cruncher Claude Shannon, who created information speculation. The objective of DSRPAI was to re-join experts from different callings so to make one more assessment zone pointed toward building machines fit for replicating human information. The earlier explanation demonstrates that AI will become as much a part of daily life as the internet or social media did previously. As a result, AI will have a profound influence on not just our personal lives, but also on how businesses make choices and engage with their external stakeholders (e.g., workers, clients) (Haenlein and Kaplan, 2019). AI sensing devices will boost human abilities in a variety of areas.

AI it is said, is used extensively in computer science research and operational research. Even though artificial intelligence has its roots in Computer Science, and that its early commercial use was in a relatively narrow domain such as robotics, the learning algorithms that are presently being developed propose that artificial intelligence may eventually have applications across a very wide range (Kamble, Shah, 2018).

More important than whether AI will play a part in these components is what role it will play, and how AI systems can reside next to each other (peacefully) in a society with humans. It's a question that all organizations must address in today's environment (Haenlein *et al.*,2019).

2.1.1 Artificial Intelligence

AI is the practice of making computers intelligent, and intelligence is the characteristic that allows an object to operate correctly and predictably in its environment (Grosz, Altman, Horvitz, Mackworth, Mulligan, Mitchell and Shoham, 2016). Artificial intelligence refers to computer capabilities that we anticipate will necessitate human intellect. Because this is a subjective definition, if an assured machine skill is deemed AI may vary as the hopes of computers grow.

Human intellect isn't required for activities that robots have been able to achieve better than humans for generations (for example, a simple automated calculator can execute computations faster than the human brain and nearly never gives an error). As a result, we may or may not consider language translation algorithms as artificial intelligence today, as they can now readily perform jobs that previously needed human intellect (for example, converting text into a new language). We still believe human intelligence to be required for correct visual perception and voice recognition, consequently, machine learning algorithms that effectively do these jobs will most likely be termed AI (for the time being).

Human intelligence, on the other hand, differs from machine learning algorithms in that the former can manage a wide range of complicated jobs. So-called "weak" AI today includes the most powerful machine learning application, such as DeepMind’s AlphaGo programs, which is better than any human player at the complicated board game Go. We have yet to develop a strong AI (also called Artificial General Intelligence) with the ability to do all of the cognitive activities that a person can. However, substantial development in artificial intelligence over the previous several years suggests that an influential AI may be within reach soon (Ceo and Pichai, 2018).

2.1.2 The Future: Need for Regulation

AI systems will become more prevalent in our daily lives shortly, which the demand of whether or not guideline is required and in what form. Though AI is inherently neutral and without prejudice, this does not rule out the possibility of bias in AI-based systems. In reality, any partiality existing in the raw data used to train an AI system endures and may even be magnified by its very nature. For example, research has validated that the sensors used in self-driven cars identify lighter skin tones compared to the darker ones or decision support systems used by judges may be drastically biased since they are centred on the evaluation of historical verdicts). Rather than attempting to control AI, the pre-eminent method to evade such mistakes is likely to set widely agreed rules for the teaching and analysis of AI algorithms, maybe in combination with some type of guarantee, like users and safety testing rules used for physical items. So long as the technical components of AI systems don't change, there will be a stable regulatory environment (Haenlein *et al.*,2019).

2.2 Artificial Intelligence Techniques

AI structure is a module that must be perused by the PC and the code that sends by the developer into the PC will be performed. The program will take the contribution from the client and show the outcome as a number after the limitations have been incorporated. The variation kind of AI procedures will deliver various sorts of results as the choice made is contorted with the requirements included and in light of the client interest (Chuen, 2015). AI is assuming a significant part in comprehension and performing astute undertakings expressed as judicious, mastering new abilities, and adjusting to new circumstances and issues. Here, we manage fundamental AI methods: Heuristics, Support Vector Machines, Neural Networks, the Markov Decision Process, and Natural Language Processing, with some distinctive methods like Machine Learning, Machine Vision, Automation and Robotic

2.2.1 Support Vector Machines

A classification difficulty is determining whether an email is a spam or not. The goal of these tasks is to assess whether a particular data item fits a specific class or not. After teaching a classifier model on data points for which the class is known (for example, a collection of emails labelled as either spam or not), you may use the prototype to identify the class of new, previously unknown data points. Support Vector Machines are a strong approach for dealing with these sorts of difficulties (SVM).

The main idea behinds SVM is that you try to get the bound differentiation that splits the two classes but in specified a way that the bound series generates a peak alteration between the classes.

2.2.2 Artificial Neural Networks

Creatures are fit to activity (visual or inverse) data from their current circumstance and adjust to change over. They utilize their uncomfortable framework to perform a lot of animation. Their nervous system can be demonstrated and reproduced and it ought to be plausible to show connatural movement in fake frameworks. Artificial Neural Networks (ANN) are characterized as handling gadgets that are inexactly demonstrated after the neural design of a mind. The critical contrast between the two is that the ANN may have hundreds or thousands of neurons, while the neural construction of a creature or human mind has billions.

A neural construction's centre idea is that every neuron is connected to different neurons with changing levels of solidarity. Yield is made dependent on the sources of info taken from the yield of different neurons (close by the association strength), which may then be used as contribution by different neurons. By adding weight to address the power of the association between neurons, this key idea has been transformed into an ANN. Moreover, every neuron will utilize an arithmetical capacity to control its yield by taking the yield from the associated neurons as information. This yield is then utilized by different neurons once more.

2.2.3 Markov Decision Process

A Markov Decision Process (MDP) is a decision-making modelling framework by which the outcome is partly random and partially depending on the decision maker's input in some scenarios. Optimal planning is another instance where MDP is employed. MDP's main objective is to find a policy for the decision-maker that specifies what specific action should be executed at when the point in time. A policy can be taught using "value iteration" or "policy iteration" after the MDP has been defined. The anticipated rewards for each state are calculated using these approaches. The policy then determines the appropriate course of action for each state. (Diepen, Hicham and Bouazzaoui, 2018).

2.2.4 Machine Learning

It's one of the AI applications in which robots aren't expressly designed to execute certain tasks but instead lean and improve via experience. Deep learning is a type of machine learning that uses ANN to make predictions. Unsupervised learning. Supervised learning and reinforcement learning are examples of machine learning algorithms. The algorithm in unsupervised learning does not use categorized data to operate on it without any supervision. It realizes a function from the training data, which comprises a set of an input object and the intended output in Supervised learning. Reinforcement learning is used by machines to take fitting actions to increase the reward to find the best option which should be taken into account.

2.2.5 Automation and Robotics

Robot means to get the monotonous and redundant assignments done by machines which additionally further develops effectiveness and in getting a practical and more powerful outcome. Numerous associations use machine learning, neural network exchange online by utilizing CAPTCHA innovation. Mechanical interaction computerization is customized to play out a high-volume cyclic assignment that can adjust to the adjustment of various conditions. (Pedamkar, n.d., 2020).

2.3 Particle Swarm Optimization (PSO)

Particle swarm optimization is a transformative advancement approach roused commonly that is utilized to address computationally testing improvement issues. It is a strong stochastic improvement approach that depends on swarm development and insight. It was made in 1997 by James Kennedy and Russ Eberhart established on the social conduct of organic animals that movement in bunches swarms. By abstracting the functioning system of normal wonders, it has been viably used in a wide scope of search and enhancement issues. Since PSO is a populace based (swarm) transformative calculation, it has a few attributes with GA. The transformative calculations, then again, are established on regular advancement standards, for example on a cutthroat way of thinking, which suggests that unquestionably the fittest people will in general endure.

PSO, then again, coordinates an agreeable way to deal with critical thinking, since all particles that might live change themselves after some time, and one particle's effective variation is shared and reflected in the exhibition of its neighbours. The major unit of PSO is a particle, which might fly across search space towards an ideal using the two its data and the data given by different particles in its area. In PSO, a multitude of *N* particles (individuals) speaks with each other either straightforwardly or in a roundabout way through search headings (gradients). The procedure utilized was a set of particles flying across a search space to track down the global optimum.

During a PSO cycle, each particle update depends on its related knowledge and the experience of its neighbours. PSO has a few applications in a wide scope of fields. Arriving at the scholarly circle PSO can be taken advantage of in every one of the undertakings where ideal and efficient usage of assets are required (Rathod, Saha, and Sinha, 2020). Its entirety with a populace of particles and every particle shows a candidate solution of the enhancement work in multidimensional search space (Chen and Shil, 2013). The fitness of the particle is to compute the fitness function of every particle. PSO starts with a populace of particles that are relegated to the inquiry search space. Toward the finish of every cycle, each PSO particle changes its position dependent on the speed determined considering;

(i) Its present position

(ii) Personal best position and

(iii) Global best position of the population (Imran, Akhand, Shuvo, Siddique, and Adeli, 2019).

Before working with the PSO, we have to know about the elements used in the PSO. First of all, we shall look talk about a brief overview of the concept of the PSO elements.

i. **Particle**: Particles are characterized as P_i for real numbers.
ii. **Fitness Function**: This capacity is utilized to track down the ideal arrangement. Ordinarily, it is viewed as the goal work.
iii. **Local Best**: It is the best position of the particle among everything positions visited up until this point.
iv. **Global Best**: The position where the best fitness is accomplished among every one of the particles visited up until this point.
v. **Velocity Update**: This is a vector used to decide the speed and heading of the particle speed is refreshed.
vi. **Position Update**: All the particles attempt to push toward the best situation for ideal wellness. Every particle in PSO refreshes its situation to track down the global optima. The position is refreshed (Muhammad Imran, Hashima, and Khalid, 2013).

Therefore, it has also been applied in this study to create the optimal timetabling for university courses.

2.3.1 The Flow of PSO

The test is dealt with in PSO by indicating the quantity of particles in an search space to handle the objective function at its current position. The particle's development will be chosen by arbitrarily blending it in with the best fitness position (Chuen, 2015), which is enlivened by friendly conduct, for example, bird rushing and fish tutoring. Anyway, how would you approach discovering the food? Accept a group of birds is scouring a region for food. There is just a single food thing advertised. Birds are totally uninformed of the area of the food. They, nonetheless, realize how far the food is from where they are at present. The ideal method is to pursue the bird that is nearest to the food. A flying bird has a position and a velocity whenever. Looking for food, the bird changes its position by changing the velocity. The velocity changes dependent on his past experience and furthermore the inputs got from his neighbour. This looking through interaction can be misleadingly reproduced for tackling non-straight optimization issues (Bhattacharjya, 2012).

The part of PSO can be masterminded into three segments which are current position x_i, previous best position p_i, and the velocity v_{id}. For each individual in atom large number will have the three D-estimation of vectors, the D suggests the component of the search space.

As the cycle starts, it will study the current position x_{id} as the issue game plan. The current position x_i will contain the match of a particle in the search space. At the point when the accentuation closes, it will choose if the current position of the particle is at this point the best position. In the event that it is, the position will be set something to the side for the accompanying vector p_{id}.

After the position has been saved, in the accompanying cycle, it will be named as the previous best position $pbest_{id}$ and taken for the accompanying accentuations to further develop result. As the cycle progresses forward, it will keep on invigorating the value of the p_{id} and $pbest_{id}$.

As of now, the vector vi will be changed and added to the current position x_{id}. Therefore, another point will be picked. At the point when the local accentuation closes, expecting the not really set in stone is the amazing the close by, it will be investigated among the around the world. Expecting the vector is the amazing the around the world, it will normally be picked as the best among overall p_{gd} (Chuen, 2015).

2.3.2 Particle Swarm Optimization Process

Beginning a PSO calculation, the underlying velocity and position of each particle in a gathering of still up in the air haphazardly, developing the accompanying cycles:

Let, $A \subset R^n$ be search-space and the swarm are characterized as a set S = {X1, X2..., Xm} or M particles (up-and-comer arrangement), where M is the boundary characterized by clients or the calculation. The ith particle measurement or *d* is expressed as:

$X_i = (X_{i1}, ..., X_{id})^T$, I = 1,2, …, M. every particle is a likely answer for an issue arranged by three amounts: Velocity (V) = $(V_{i1}, ..., V_{id})^T$, current position X = $(X_{i1}, ..., X_{id})^T$ and individual best position $pbest_1 = (pbest_{i1}, ..., pbest_{id})^T$. Let *t* be current cycle and *gbest* be it global best position gotten so far be any of its particles. At first, swarms are spread arbitrarily inside search space and arbitrary velocity is doled out to each particle. Particles share data and associate with each other to track down the ideal arrangements. Every particle moves toward its own best (pbest) and its global best position (qbest). To get ideal arrangement, each particle changes its velocity as per the sane and social parts addressed as:

a) Velocity Update Equation:

To look for optimal arrangement, every particle changes its velocity as per the intellectual and social parts given by:

V_{ij} (t + 1) = w(t)V_{ij}(t) + c_1R_1[$pbest_{ij}$(t) – X_{ij}(t)] + c_2R_2[$gbest_j$(t) – X_{ij}(t)] Eqn 2.1

where, I = 1,2, ...M and j = 1,2. ..., d. however, in the event of swarm upheaval impact, relating velocity part is confined to the storage room velocity bound as expressed by:

V_{ij} (t + 1) = - V_{max}if V_{ij} (t + 1) < - V_{max} Eqn 2.2

=V_{max}if V_{ij} (t + 1) >V_{max}

The position is updated utilizing the position update condition in (2).

b) Position Update Equation:

At the point when velocity is updated, each particle moves to another likely arrangement by refreshing its position as expressed by:

X_{ij} (t + 1) = X_{min}if X_{ij} (t + 1) <X_{min}

=X_{ij}(t) + β V_{ij} (t + 1), if X_{min}

X_{ij}(t + 1) ≤ X_{max}

=X_{max}if X_{ij} (t + 1) >X_{max} Eqn 2.3

Where I = 1, 2..., M: j = 1, 2..., *d*. from the situation above, V_{ij}, X_{ij}, and $pbest_{ij}$ are separately velocity, current position, and personal best position of particle *i* on the jth measurement. what's more, *gbest* is the jth measurement global best position accomplished so far among all particles at emphasis t. Arbitrary upsides of R_1 and R_2, which are similarly not reliant and appropriated equally over I 0,1j, fl is an imperative factor used to control velocity weight, with values normally set to 1. c_1 and c_2 are positive constants normally called "acceleration factors". Factor c_1 is in some cases called the "cognitive" boundary, and c_1 is known as the "social" boundary.

c) **Inertia weight:**

Inertia weight at iteration *t* is *w(t)* which is used to balance global and local discovery which is defined as:

$w(t) = w_{up} – (w_{up} – w_{tow})t/T_{max}$ Eqn 2.4

where *t* is the current number of iterations, w_{up} and w_{tow} denote the lower and upper limits of *w and* T_{max} is the maximum number of iterations (Rathod, *et al.*2020). The PSO steps is defined as:

1) Initialization
   i. Create *n* population of the particle by positioning them randomly in the search space.
   ii. Calculate the fitness of each particle then assign its current position (*Xp*) as personal best position (*B*p).
   iii. Get the global best position (*G*) amidst all the particles.
2) Update position of each particle:
   i. Calculate the velocity using Eq. (2.1).
   ii. Move to the new position using Eq. (2.3).
   iii. Calculate fitness.
   iv. Personal best (*Bp*) and global best position (*G*) update, considering the fitness of new position.
3) Terminate and result:

   If termination criteria are not met then go to step 2: else, stop and consider global best position (*G*) as the final result (Imran *et al.,*2019).

2.3.3 PSO for Timetable Scheduling

The suggested technique tries to modify two slots of particles for each particle, giving the possibility of getting the best global and local solutions. The chances of particles finding a better solution and leaping over local optimal solutions are high. First and foremost, similar to

PSO operations, the suggested self-mutated PSO technique likewise uses random creation of an initial population to solve the issue.

Given M set of particles to be utilized, the particles are defined as:

$$X = \{x1, x2, \dots, xM\} \qquad \text{Eqn 2.5}$$

Initially, a group of *n* candidate solutions as particles are randomly produced. Each particle is equal to a candidate solution of a problem. The n candidate solution performance is first evaluated and ordered. The best previous position of the kth particle is put in P_k^i at the ith iteration.

From the first to the ith iteration, the best position amongst all the particles should be in G_i. the new timetable of next-generation is produced using defined steps as:

Each particle moves by the following procedures:

i. Each particle X_k^i must be changed two slots at random by itself.

$$S_k^{i+1} = rand - mutate\ (X_k^i) \qquad \text{Eqn 2.6}$$

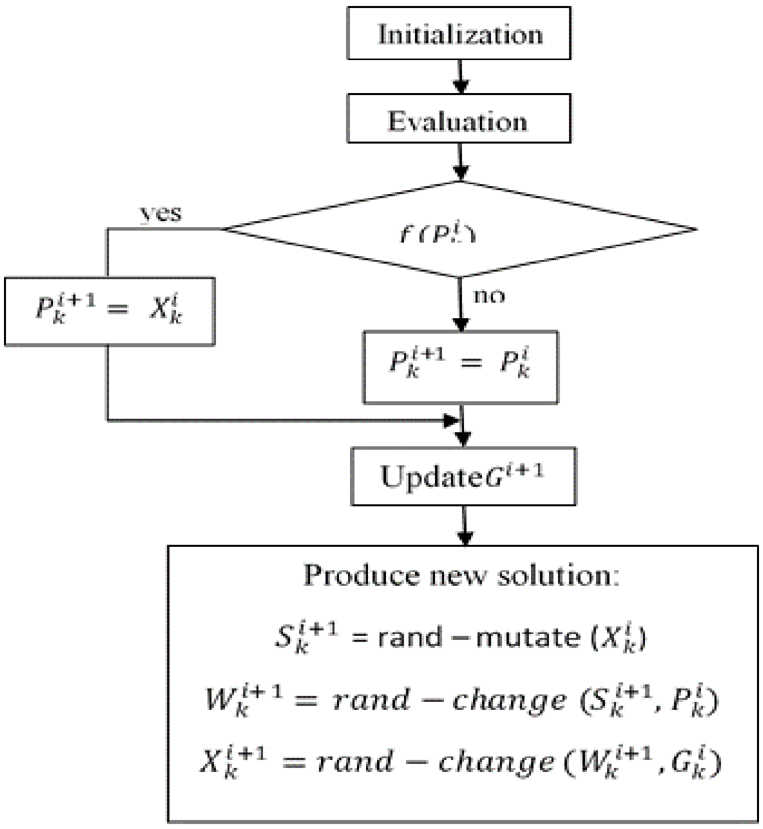
ii. Copy a random slot from the local best (P_k^i) to particle (S_k^i) for a subject.

$$W_k^{i+1} = rand - change\ (S_k^{i+1}, P_k^i) \qquad \text{Eqn 2.7}$$

iii. Copy a random slot from the global best (G^i) to W_k^{i+1} at random for a subject.

$$X_k^{i+1} = rand - change\ (W_k^{i+1}, G_k^i) \qquad \text{Eqn 2.8}$$

The development cycle will repeat continuously until an optimal timetable is found or a maximum number of iterations is reached.



Initialization
Evaluation
yes
$f(P_k^i)$
no
$P_k^{i+1} = X_k^i$
$P_k^{i+1} = P_k^i$
UpdateG^{i+1}
Produce new solution:
S_k^{i+1} = rand − mutate (X_k^i)
$W_k^{i+1} = rand - change\ (S_k^{i+1}, P_k^i)$
$X_k^{i+1} = rand - change\ (W_k^{i+1}, G_k^i)$

Figure 2.1: Framework of PSO for timetabling

Hence, to escape from the local optimum and move towards the global optimum in a short time is done with ease (Chu and Chen, 2006).

2.3.4 Course Timetabling Problem

The literature has a wide range of variances based on the structure of organizations and the types of limits or constraints that are inherent in timetabling issues. Timetabling issues are divided into three categories. A school schedule provides a weekly calendar for all of a building's pupils, barring teachers from attending two courses at the same time, and vice versa. The else collaborator's course timetabling is a weekly calendar including all lectures for a range of university classes, reducing the gap between general students' lectures.

Exam timetabling is the programming of a program of university courses for examinations, preventing the copy of examinations of courses with tourist students, and extending the examinations as far as practicable to the (Techie-menson and Nyagorme, 2021).

2.3.5 Constriction Factor Version of PSO

Every particle has it cut off points for position and velocity. The lower and upper limits are from the restrictions of factors addressed by the particle's position, for as far as possible. However, clients can draw the velocity lines for every particle. Ordinarily, the nature of the answer for the PSO optimization strategy is delicate to the psychological and social boundaries and velocity cut-off of particles. Along these lines, a few endeavours to control the investigation and double-dealing capacities of the PSO calculation are directed by changing the psychological and social factors or restricting the scope of velocity in the reach [-v_{id}, max, v_{id}, max]. Clerc (1999) in his past completed work, shows that the utilization of constriction factor might be needful to guarantee intermingling of the PSO calculation. The changed velocity for the particles with constriction factor is characterized as:

$$v_{id}^{(k+1)} = C \times [\, v_{id}^{(k)} + c_1 \times rand_1 \times (\, pbest_{id}^{(k)} - x_{id}^{(k)} + c_2 \times rand_2 \times \left(gbest_i^{(k)} - x_{id}^{(k)}\right)] \qquad \text{Eqn 2.10}$$

$$C = 2 \,/\, |2-\varphi+\sqrt{\ }\, \varphi 2 - 4\varphi\,| \qquad \text{Eqn 2.11}$$

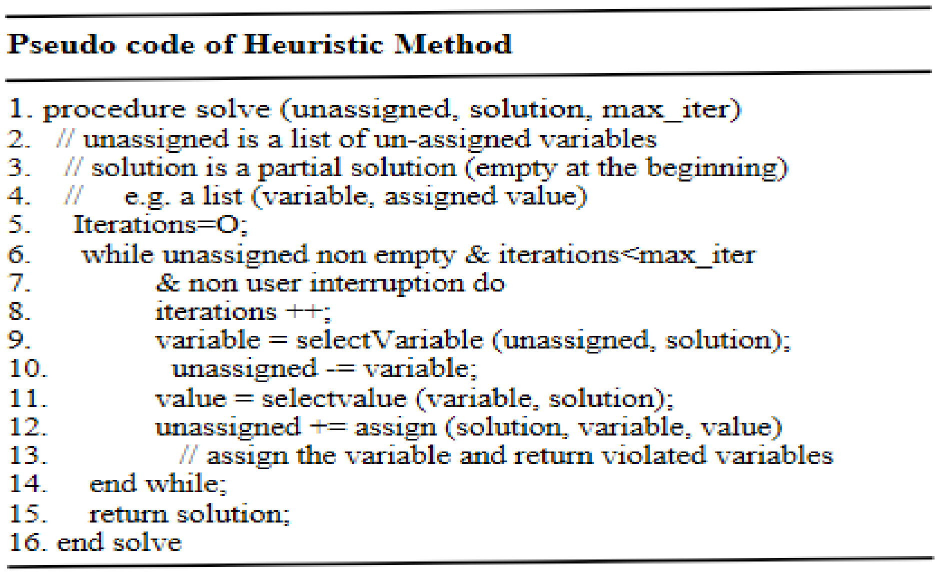
The factor (ϕ) influences the consolidating attribute of the framework and the worth should be set to be more prominent than 4.0 to guarantee security. Be that as it may, the constriction (C), diminishes delivering uniqueness which prompts a slower reaction. The standard worth of ϕ is given as 4.1 (for example $c_1 = c_2 = 2.05$). Having executed the constriction factor (CF), and in light of the numerical hypothesis the search interaction guarantees the union for the technique. Hence, the PSO-CF can get a superior quality arrangement contrasted with the fundamental PSO strategy. Be that as it may, the particles not just move rapidly and accomplish union when their worth is bigger than 4, yet the intermingling can likewise occur at a proper velocity. The settings are ordinarily relegated as K = 0.72984, and $c_1 = c_2 = 2.05$.

Clerc and Kennedy (2002) in their work later called attention to that confining the particle velocity was excessive: CF introduced to update equation with speed could likewise understand the reason for restricting particle velocity. Anyway, analyses show that even the CF is utilized, a superior outcome will be acquired if the particle velocity was restricted simultaneously (Eberhart and Shi, 2000), accordingly the speed constraint of the calculation will in any case be held.

2.4 Heuristic Method

A heuristic or heuristic methodology is said to be any method for solving the problem that employs a practical approach or a different of shortcuts to generate answers which are not ideal but are adequate given a restricted period or deadline. Heuristic approaches are meant to be flexible and used for fast decisions, especially when seeking an optimal solution is either impossible or impractical and when working with difficult data (Heuristics & Heuristics, 2021). In contrast to correct methodologies, which guarantee ideal arrangement, the heuristics approach just endeavours to yield a decent yet not really ideal arrangement. In any case, the time needed by a precise strategy to find an ideal answer for a troublesome issue, if such a technique exists, its significant degrees is more than the time by a heuristic methodology (some of the time it takes such a long time which much of the time it is unimportant). Therefore, the heuristic methodology is utilized oftentimes to address true streamlining issues (Mart and Reinelt, 2011). Heuristics help to make fast judgments, answer and issue, analysts in every business utilize rules of thumb such as informed guessing, trial and error, and historical data analysis. Heuristic methods use shortcuts and good-enough calculation to enable decision-making easier and faster. With heuristics applications, some trade-offs render the method prone to bias and mistakes in judgment. The final option of the user may not be the best or most optimum answer, the decision may be incorrect, and the data selected may be inadequate (lead to an inaccurate solution to an issue) (James Chen, 2021).

In the heuristic methodology, two kinds of conditions are expressed as: given arrangement of factors that are not being appended and the midway arrangement, which attempts to change over the midway arrangement into a total answer for every cycle of the calculation which can tackle the given issue. After beginning with the existed arrangement in the calculation, it will circle until it arrives at the greatest emphases or it has allotted every one of the factors.



Pseudo code of Heuristic Method

```
procedure solve (unassigned, solution, max_iter)
  // unassigned is a list of un-assigned variables
  // solution is a partial solution (empty at the beginning)
  //     e.g. a list (variable, assigned value)
  Iterations=O;
  while unassigned non empty & iterations<max_iter
          & non user interruption do
          iterations ++;
          variable = selectVariable (unassigned, solution);
            unassigned -= variable;
          value = selectvalue (variable, solution);
          unassigned += assign (solution, variable, value)
            // assign the variable and return violated variables
  end while;
  return solution;
end solve
```

By going through the variables, the algorithms decide the value. The value of the variable will be selected by passing over every domain using the function given in the algorithms. The circle will continue till the best fitness value is found (Chuen, 2015). Other motivations for utilizing heuristic approaches, in addition to the requirement to discover good answers to complex issues in a fair amount of time, include the following:

a) The technique for taking care of the issue to optimality isn't known.

b) Even however there be a definite strategy to take care of the issue, it can't be utilized on the accessible equipment.

c) The heuristics approach is more adaptable than the specific technique.

d) The heuristic methodology is utilized as a component of a worldwide technique that guarantees to track down the ideal answer for an issue.

Properties of a decent heuristic calculation:

a) An arrangement can be acquired with sensible computational exertion.
b) The arrangement ought to be close ideal (with high likelihood).
c) The likelihood of getting a terrible arrangement (a long way from ideal) ought to be low.

There exist several heuristic techniques, each with its own set of characteristics. As a result, providing a complete categorization is challenging. Furthermore, many of them were created to tackle a specific problem and have no way of being generalized or applied to other comparable situations. The following overview tries to provide broad, non-exclusive categories under which the more well-known heuristics can be classified:

i. **Decomposition methods**

   The original problem is broken into sub-problems easier to solve, with the understanding that sub-problems all belong to the same problem class.

ii. **Inductive approach**

   The aim is to apply the smaller versions to the entire situation. properties or approaches which have been discovered as being simpler to analyse in certain situations can be applied to the entire problem (Mart *et al.*,2011).

2.5 Constraints

Constraints are defined as the requirements that must be satisfied to create a realistic timetable in the University Timetabling issue. There are two types of constraints: hard and soft constraints. Hash constraints should not be broken under any circumstances. Soft restrictions, on the other hand, are the appealing type of requirements that may be controlled as minor yet allow for optional conditions.

A good schedule produces a viable result, but one of higher quality has the fewest overall violations of the soft restrictions. For specific cases, the hard restriction can also be considered as the soft constraint to find a feasible solution. There are varieties of other restrictions that exist in a timetabling problem. Two basic constraints in any timetable issue are conflict and availability of the resource recorded by Yeasin and Khader. Lecturer’s essential task constraints (accessibility, clashes,) school prerequisites (timeslots), institutes constraints (precedence), and constraints related to (Imran *et al.*,2019) lecturers' preferences were classified by Schimmelpfeng and Helber into four categories (work load, day-off and meeting pattern). The improvement of limitations is ongoing all through time. In this field of research, even the smallest modifications to a restriction have demonstrated significant progress. The institution course timetabling characteristics may be divided into the following categories based on the literature:

i. Completeness: each class meeting must be booked into an opening. All course exercises remembered for the prospectus should be relegated in the plan.
ii. Conflict of assets: assets talk about instructors, understudies, homerooms and timing. They ought to be booked once at a time and a spot.
iii. Work load: they have a predetermined number of educating and learning hours either for a day or week that ought to be thought of.
iv. Availability of assets: this is identified with the accessibility of instructors, rooms, students, and time allotments.
v. Meeting designs: this discussions regarding how the class meeting is to be relegated (Aziz and Aizam, 2018).

2.6 Comparison of Heuristic Method and PSO Method

The Heuristic approach and PSO are nearly identical in that both evaluate the function to find the optimum solution to the issue. In PSO, however, the problem is simulated as particle particles with their current best location in a search space.

Table 2.1: PSO comparison with Heuristics method

	Particle Swarm Optimization (PSO)	**Heuristics**
Simulation	Swarm flocking in a population	Arithmetical algorithm
Process	1. Generating particle's position and velocity. 2. Update the velocity and the position. 3. Retrieve the best global position.	By evaluating the function, the search for the best solution is made in the algorithm. The evaluation function will update the existed solution to a better solution through the iteration in the algorithms.

By adding velocity to the particle with the current best position and getting another position for the particles, the arrangement will then, at that point be delivered. The particles' fitness worth will be surveyed together close by their velocity until they arrive at a condition that makes the cycle end, so, all things considered the local best area will be made which will then, at that point be presented and contrasted with global best positions to track down the best global position, which is the best arrangement among the populace. There are search spaces in the heuristic methodology likewise, yet the cycle will go through the entirety of the accessible answers for track down a superior one. By applying the calculations into the cycle, the capacity set will update the arrangement into a superior arrangement and it will require some investment with the assessing demonstrates (Chuen, 2015).

2.7 Related Work

Adrianto (2014) introduced research work on examination using PSO and GA for timetable scheduling which expected to talk about the utilization of (PSO) that can be utilized to consequently create ideal instructor plan planning. The computation was performed by looking at the measure of punishment procured each time a hard limitation or delicate requirement is disregarded by the execution of PSO or GA to the absolute punishment got when all imperatives are abused. The aftereffects of the execution of PSO and GA showed that the plan planning by utilizing PSO is far superior to GA.

Naderi (2017) proposed three algorithms for solving the problem of university course timetabling in form of a linear integer programming model. The imperialist competitive algorithm, variable neighbourhood search, and simulated annealing are the three algorithms suggested. The result reveals that in terms of performance, the imperialist competitive algorithm outperformed the other algorithm.

Ekanayake, Subasinghe, Ragel, Gamage, Attanayaka (2019) presented a paper on Intelligent Timetable Scheduler: This paper has inspected the Exam Timetabling issue with Genetic and Graph Coloring estimations and the Semester Timetabling issue with Heuristic and Iterated Local Search computations taking a gander at four computations. The paper had the option to track down the best calculation dependent on the running time of the calculation as a rate given as Genetic Algorithm (GA) take 25% of the time, Graph Coloring Algorithm (GCA) takes 11% of Time, Iterated Local Search Algorithm (ILSA) takes 44% of the time, and Heuristic Algorithm (HA) takes 20% of the time.

Marada, Kaki, and Pilaka (2019) worked on Automatic Timetable Generation Using Genetic Algorithm. In this paper, a Genetic Algorithm was used in other to find a solution to the timetable problem. The major challenge was that using a Genetic Algorithm is a little slower due to the steps it has to undergo. The result showed that the Genetic Algorithm is one of the most effective ways of generating timetables although it does not give a 100% optimal timetable. The recommendation due to the result was that for the Genetic Algorithm to be implemented, the constraints to be used should be chosen carefully to get an optimal solution.

Mallick, Majumdar, Mukherjee, Bhat (2020) proposed a paper on Hybrid Cuckoo Search Approach for Course Time-Table Generation Problem. This article proposes a clever mixture Cuckoo scan approach for addressing the Course Time-Table Generation (CTTG) issue for secondary schools associated toward the West Bengal Board of Secondary Education (WBBSE), India. The creators research the exhibition of local search Hill moving against the populace put together fundamental Cuckoo search calculation with respect to the issue. The outcome shows that albeit the quantity of emphases for the cuckoo search approach is more, still it delivers a superior arrangement in contrast with the enrolled Hard Constraints before the expense decrease becoming immersed. The scientists are occupied with their undertaking towards examination towards the adjustment of the Hybrid Cuckoo Search calculation to diminish the tally of cycles.

Hambali, Olasupo, and Dalhatu (2020) presented a paper on an Automated university lecture timetable using a heuristic approach. This paper presents a combination of genetic algorithm (GA) and simulated annealing (SA) to have a heuristic approach (HA) for solving course timetabling problems in Federal University Wukari (FUW). To achieve an optimum output design, the researchers adopt Charles's Darwin theory on survival of the fittest (genetic algorithm), simulated annealing along with graph colouring heuristic to generate a multidimensional array as space for referencing the entire courses in any given semester. The result shows that the implemented system was more flexible to work perfectly in other institutions that have similar constraints with FUW. In the area of further works, the researcher suggests the implementation of the online or web-based application that is accessible to all students and lecturers within the institution and the world at large.

Arratia-Martinez, Maya-Padron, and Avila-Torres (2021) presented a paper on University Course Timetabling Problem with Professor Assignment. In this paper, the analysts introduced the clever whole number direct programming model proposed to settle the case considered. The test was that the division will require an individual with information about numerical displaying to alter the model and adjust it to new prerequisites. The outcome shows that the number programming model was carried out with ILOG CPLEX Optimization Studio adaptation 12.8. It was noticed that the ideal arrangement in 3.16 seconds was acquired. As future work, the analysts are taking a gander at the chance of expanding the schedule for various investigation programs freely and afterward concentrate all ramifications for an incorporation of all courses in the establishment.

Budi, Wen Chen, Larasati, Prastyo, and Dwiastuti (2020) presented a paper on Multi-Objective Modelling for a Course Timetabling Problem. This paper presents a multi-objective modelling approach for the Curriculum-Based Course Timetabling (CBCTT) problem. The problem comprises optimizing weekly scheduling by assigning offered courses to classrooms and periods. The result shows that the developed model considers the issue to satisfy the stakeholders by minimizing the cost objective to accommodate the university policy with favourable results within a second compared to the manual process. For future work, since the finding is simple, basic, and practical, this very beginning conceptual simplified cost-minimizing could be a basis for further model modification

2.8 Summary of Review

As a result, schedule production should be efficient in terms of time and usefulness in a timetabling scheduling system. The timetable can be created using the PSO approach by selecting particles (courses or locations) at random, rather than searching one by one. In comparison to the Heuristic and Manual methods, the timetable being created will have to search for every venue before selecting one for a course, which is inefficient. As a result, this Timetabling Scheduling System's timetable is generated using the Particle Swarm Optimization technique.

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.1 Tools and Materials used

The tools used for the design and methodology include Algorithm, Modified RAD, and Flow chart. The materials and tools used to implement this new system are the Kotlin programming language for the design interface and MySQL for the database of the system.

3.1.1 Requirement analysis of the proposed agent

The functional requirement, non-functional requirement, and system justification are the three subheadings that the proposed agent's needs are broken down into.

3.1.1.1 Functional Requirements

Functional requirements explain what the proposed schedule system should do or how it should behave, and the requirements it must meet are listed below.

i. Administrators should be able to add, change, and delete lecturers, rooms, departments, times, and dates in the system.
ii. It does not require entering a password to access the system.
iii. All smart gadgets will be supported by the system.

3.1.1.2 Non-functional requirements

A non-functional requirement restricts how the proposed schedule system should work or the formulation of the system's performance characteristics. The conditions it must meet are as follows:

i. The suggested system should feature a user-friendly, easy-to-follow interface.
ii. The system's execution time should be reduced.

iii. The system should take up less phone space and should only work on devices running Android ranging from version 4.0.
iv. For the system's database, the suggested system should use Kotlin and MySQL.

3.1.2 Justification for the New System

The new system is intended to address issues with the current manual system. It is intended to be utilized on phones, easing the exam officer of most of the burden associated with the manual approach. The suggested system automatically analyses data.

3.1.3 Advantages of the Proposed System

i. Except for the manual timetabling mechanism, the system ensures consistency.
ii. The method is more flexible than the manual timetabling system.
iii. It has an intuitive interface that makes data entering and updating simple.
iv. Create schedule picture snapshots that are ready to print.

3.1.4 limitation of the proposed system

The following are the difficulties in the framework because of time limitations:

i. Only a few hard course restrictions can be used to create schedules in the proposed approach.
ii. This system's timetable is still subject to change by academic centre bodies.
iii. The system is aimed towards a certain department.

3.2 Feasibility Study

Planning is a theme that is oftentimes examined in the computer area since making a schedule sets aside time. It should consider a few variables to assemble a schedule that might be used by the client without time conflicts. A great deal of timetabling studies has been completed by utilizing various types of AI procedures like the Graph Heuristic Method, GA. By applying the methods, the framework will assist the client with making a schedule naturally in the wake of satisfying every one of the requirements that have been characterized in the framework.

3.3 Methodology

By following the progression of the methodology, the framework is created from one stage to another, guaranteeing that the framework is grown properly before sending. The technique might be utilized as a rule in a framework to make the framework with the method. This undertaking will be carried out the Rapid Application Development (RAD) strategy. With the RAD strategy, the framework will be grown productively as this technique give a more sensible and pragmatic technique during improvement.

3.3.1 Architectural Design

A suggested system's architectural design is a graphical depiction. Understanding how a system should be structured and developing the overall structure of that system is what architectural design is all about. This is the first process. It highlights the major structural components of a system as well as their interrelationships. The application architecture, shown in Figure 3.1, is a graphical depiction of the application.

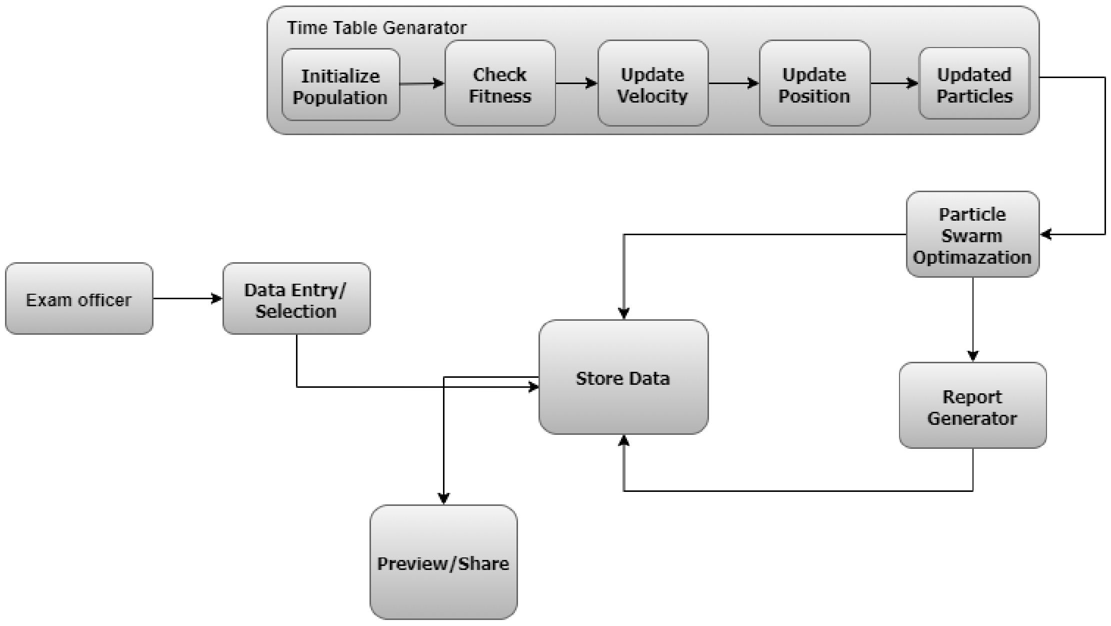


Figure 3.1: The architectural design of the system

It also depicts the possible linkages and interactions inside the system. The process function, system database, and external entities that will interact with the system are all depicted in Figure 3.1. Figure 3.1 shows the timetable system architecture. One user, the exam officer, is responsible for maintaining the timetable; he manages and modifies the timetable then prints or shares it with the lecturers.

3.3.2 Modified Rapid Application Development (RAD)

The RAD model depends on prototyping and iterative improvement with no particular arranging included. Quick application improvement is a product advancement procedure that utilizes insignificant arranging for fast prototyping. RAD is an advancement worldview that emerged from the acknowledgment that the waterfall style of improvement was insufficient. The waterfall strategy has an essential issue in that it's hard to alter the product's centre activities and provisions once it's in the testing stage. Subsequently, you are left with programming that might meet your evolving needs.

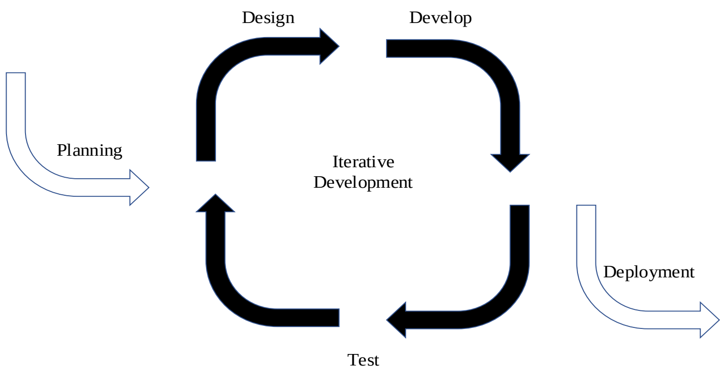


Figure 3.2: Rapid Application Development Model (Modified)

The useful parts are made at the same time as models under the RAD approach and are incorporated to make the total item for quicker item conveyance. Since there is no point by point preplanning, it makes it simpler to join the progressions inside the advancement interaction. The main perspective for this model to be fruitful is to ensure that the models created are reusable.

3.3.3 Planning Phase

This phase represents the system's first phase, which necessitates knowing and obtaining the system's requirements. A variety of artificial intelligence approaches have been investigated for implementation into the system, as well as the system's needs. In addition, during this phase, investigations of lecture schedules, including the selection of appropriate timeslots and working hours, were conducted. After that, after grasping the notion of running a timetable, the appropriate artificial intelligence approaches will be chosen.

3.3.4 Iterative Development Phase

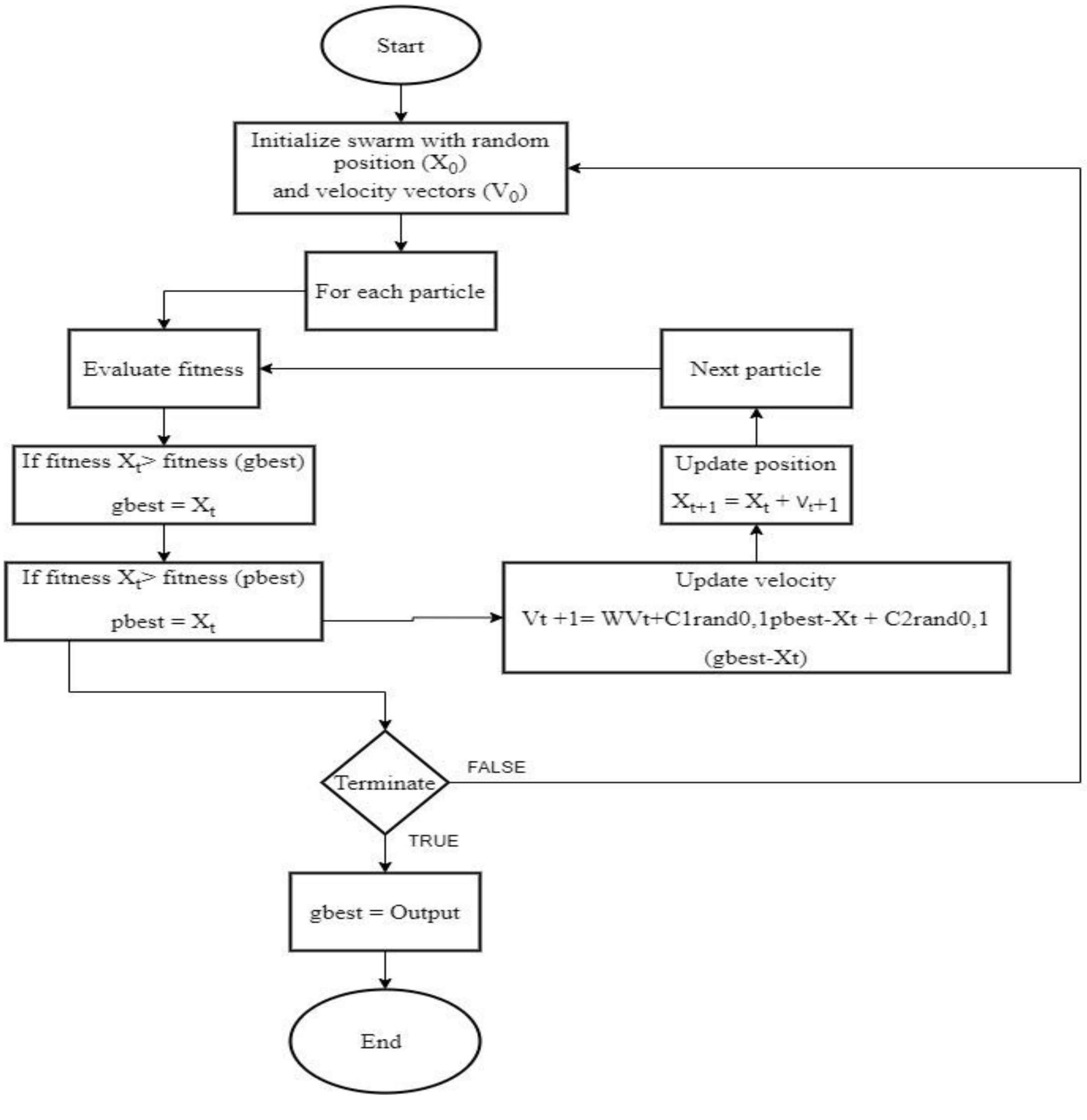
The essential objective of this stage is to deal with any progressions in the framework's requirements. This stage is planned as a cycle that incorporates plan, advancement, and testing. Toward the beginning of the advancement cycle, a model is made dependent on the client's prerequisites, and the framework's stream is archived with the goal that the framework's improvement can satisfy the client. Following the advancement of the framework, the testing stage will evaluate the framework's convenience and usefulness. At this stage, the client can check the framework whether the framework satisfy their necessities.

3.3.5 Deployment Phase

This phase will only be completed when the system has been fully designed and tested, ensuring that the system generated meets user needs and does not include serious flaws. Here, the software's features, functionalities, aesthetics, and interface are completed before providing to the customer, stability, usability, and maintainability must be prioritized.

3.4 Development of PSO

We based our algorithm implementation on Shu-Chuan Chu et al fundamental method (see Figure 3.3) and enlarged it to better represent the scheduling.



Start
Initialize swarm with random position (X_0) and velocity vectors (V_0)
For each particle
Evaluate fitness
Next particle
If fitness X_t> fitness (gbest) gbest = X_t
Update position $X_{t+1} = X_t + V_t+1$
If fitness X_t> fitness (pbest) pbest = X_t
Update velocity Vt +1= WVt+C1rand0,1pbest-Xt + C2rand0,1 (gbest-Xt)
Terminate
FALSE
TRUE
gbest = Output
End

Figure 3.3: Graphical framework of PSO

Real-world timings are compared to the suggested method. Particle encoding, initialization method, fitness function, hard and soft restrictions, cost assignment, and weight modification are all discussed in the proposed PSO approach.

Pseudo Code For a Standard PSO Algorithm:

```
1.  //Function initialize:
2.  Foreach particle do
3.      Initialize particle
4. End
5. While termination criteria not met do
6.      Foreach particle do
7.          Calculate fitness value
8.              If fitness value better than current best fitness value then
9.                  Set current fitness value as the best fitness value
10.             End
11.     End
12.     Choose the particle with the best fitness value of all particles as the group best
13.     Foreach particle do
14.         Calculate particle velocity see equation 2.1
15.             Update particle position see equation 2.3
16.     End
17. End
```

3.4.1 Particle encoding

To begin, we need 150 particles x[p], p = 0...149, and timeslot denote *ts*, each of which is made up of a two-dimensional matrix. Each particle (matrix) contains class no rows, where class no is the number of distinct courses offered by each school, and 35 columns since a week's timeslots are limited to 35, i.e. seven timeslots each day of the week's five working days. The goal is to find the best particle location possible (the optimal course timetabling solution).

Table 3.1 Particle encoding

	ts 1	ts 2	…	ts 35
Venue 1	**Lecturer**	**Lecture**	**…**	**Lecture**
Venue 2	Lecturer	Lecture	…	Lecture
Venue 3	Lecturer	Lecture	…	Lecture
…	…	Lecture	…	Lecture
Venue m	Lecture	Lecture	…	Lecture

To solve the university, course scheduling problem (UCSP), the suggested PSO employs z number of particles. Each particle object includes a viable UCSP solution, i.e., the whole schedule for teachers and rooms.

3.4.2 Population initialization

The suggested PSO technique, like other population-based algorithms, begins with an initial population. The goal of this technique is to generate as much variety in the initial random structure of each of the 150 particles as possible; the number of variables is 5, the inertia weight Wmax is 0.9, and Wmin is 0.4, and both c_1 and c_2 are 2. As a result, the algorithm can subsequently explore the whole search space rather than converge prematurely, i.e. avoid being caught in local optima. A random solution is assigned to each particle in the initial population. Time slots are assigned while adhering to all strict limitations, and courses needing many consecutive slots are double-checked to ensure that there are no break times.

3.4.3 Fitness calculation

The fitness function is critical since the quality of the produced schedules is entirely dependent on it. It returns a value representing each particle's fitness, which is the criteria of possible updating of the local and global best by the particle with the best fitness at each generation. The fitness function inspects each particle (timetable) for breaches of hard and soft constraints, and a sub-cost is applied based on these violations. The total of these sub-costs is the fitness function's return value for each particle.

3.4.4 Hard constraints

The first hard limitation is that no lecture venue be assigned more than once at the same time. The second hard limitation is that no lecture should be taken more than once each day. The final hard limitation is that no two venues shall overlap. The fourth hard limitation applies to classes that have an empty timeslot during the day that is not the day's last timeslot. The cost of this restriction is affected by the quantity of these vacant timeslots.

3.4.5 Soft constraints

The first soft limitation is that instructors have vacant moments in their schedules. The cost of this restriction is affected by the number of teachers who have empty times, the number of days they occur, and the number of these vacant periods. The second soft limitation is that no instructor may teach more hours than the undergraduate department or graduate institute allows. The third limitation pertains to the maximum of three consecutive events per day of instruction in the class schedule. The ideal situation is one in which each class has as many as five distinct lessons per day.

3.4.6 Speed limits Vmax

The particles' speed was restricted by a maximum speed Vmax, which may be utilized as a restriction to govern the particle swarm's global search capabilities. In the original PSO method, = 1, $c_1 = c_2 = 2$, particle velocity soon grows to a very high value, affecting the efficiency of the PSO algorithm, therefore particle velocity must be restricted. Later, Clerc and Kennedy (2002) pointed out that it was not required to limit particle velocity; instead, inserting a constriction factor into the speed update calculation may achieve the same result. However, even when the constriction factor was utilized, studies revealed that limiting particle velocity simultaneously produced superior results, therefore the notion of speed limitation was kept in the PSO algorithm. In general, Vmax was set to the dynamic range of each variable and was generally a constant number, but it may also linearly or dynamically decrease depending on the success of search history.

3.4.7 Position limits Xmax

Particle positions can be limited by a maximum position Xmax to prevent particles from flying out of the physical solution space. Robinson and Rahmat-Samii (2004) proposed three particular control procedures: retaining dividers, reflecting dividers, and undetectable dividers.

At the point when one of a particle's measurements crossed the limit of the arrangement space, the engrossing divider set the speed in that measurement to nothing, while the reflecting divider adjusted the course of particle speed, and the particle was at last pulled back to the passable arrangement space by the two dividers. To diminish the computation time and stay away from influence the movements of different particles, the undetectable dividers didn't ascertain the fitness values of the particles flying out of the limit.

3.4.8 Particle Swarm Optimization Parameters

The choice of parameters substantially influences the convergence speed and ability of any population-based algorithm to discover an optimal solution. In general, 20–50 particles are advised. Scaling factors c1 and c2 are also parameters. These parameters determine the particle step size for the following iteration. In other words, c1 and c2 determine particle speed. With this option, the particle's speed rises without control, which is excellent for faster convergence but bad for better search space utilization. Particles will gravitate towards the average of pbest and gbest if $c1 = c2 > 0$. During the search method, small values of c1 and c2 will produce smooth particle trajectories, but bigger values of c1 and c2 will cause abrupt motions with increased acceleration.

3.4.9 Optimizing with PSO

The search space for the PSO method is vectors with many dimensions. Our approach defines venues, time-slots, and events as the dimensions of the search space to solve the University Course Timetabling Problem. The objective of utilizing high particle sizes toward the beginning of a development technique and afterward diminishing them to 30 or less toward the end is to have the option to investigate a bigger looking through locale while downplaying the execution time by holding few strong particles. Another essential thought of the fundamental calculation is to pick a timeslot having delicate conflicts from particle "delivered W", which is created in the middle stage, and substitute it with the relating timeslot of the worldwide best.

3.4.10 Flowchart

A flowchart is a graphical portrayal of a progression of steps. It is usually used to introduce the progression of calculations, work processes, or cycles since it shows steps in consecutive request. Figure 3.4 depicts the suggested system flowchart. The actions taken by the system exam officer are depicted in the flow chart as follows: The application is launched by the system exam officer. The administrator then chooses the courses based on their codes. At this point, the exam officer will continue to add courses until the required number of courses is reached. In the event of a mistake, the exam officer can add or remove a course that has been entered.

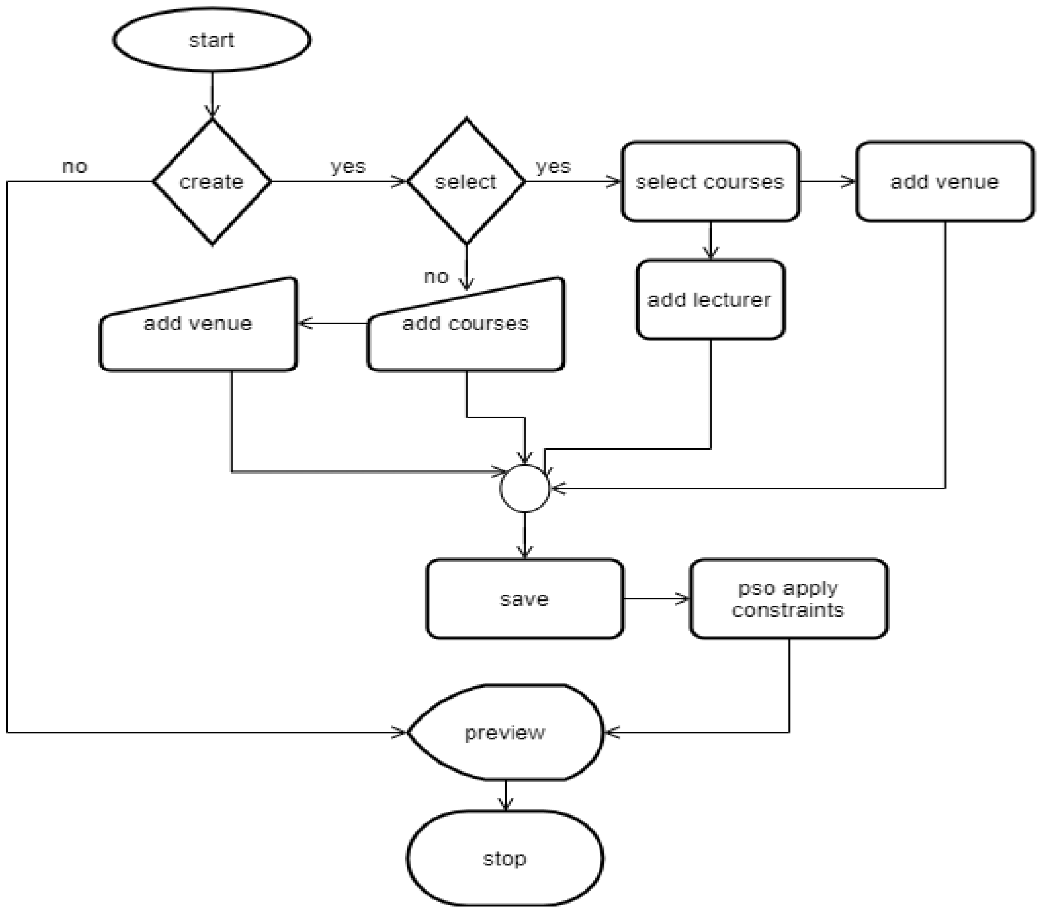


Figure 3.4: System Flowchart

After you've entered the courses, you will be able to add all of the venues that will be used. After you have chosen your venues. After these are entered, the system creates the timetable system.

3.5 Modelling the System

Modelling is the process of creating software applications before they are coded. This system was modelled using tools from the Unified Modelling Language (UML).

3.5.1 UML (Unified Modelling Language) Modelling

The Unified Modelling Language (UML) has become an object modelling standard and brings a range of approaches to the field of systems analysis and development, making it the preferred choice for this project. The Use case diagram and Class diagram will be utilized for system modelling in this project.

3.5.1.1 Use Case Diagrams

The proposed Timetable System's key operations and functions are described using use case diagrams.

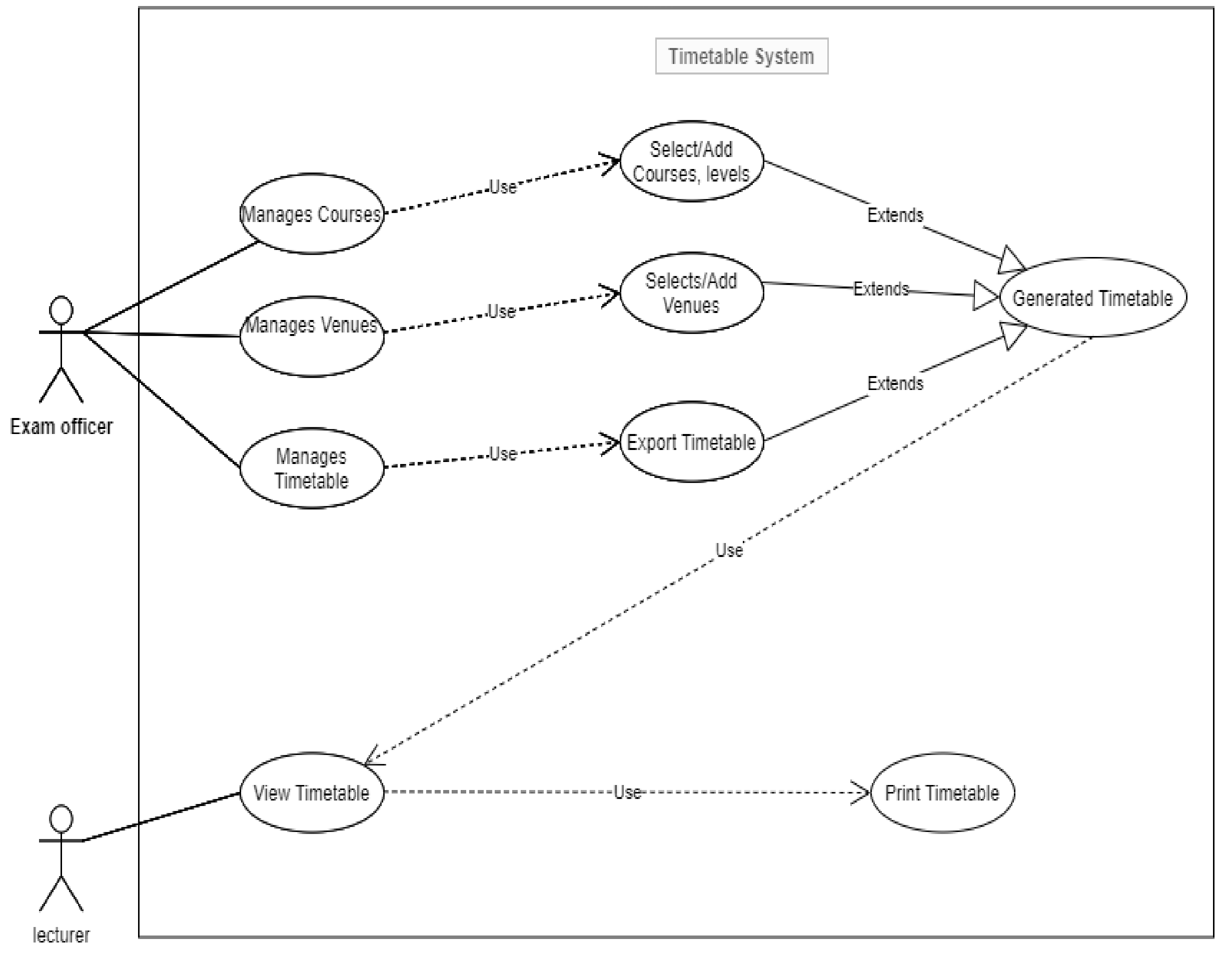


Figure 3.5: Use Case Diagram of the System

3.5.1.2 Class Diagram

A class diagram in objects-oriented modelling is a type of static structure diagram that depicts the component or structure of a system, including characteristics, actions, and connections between the system's classes. The purpose of a Class Diagram in a System is to graphically depict what the various objects will do by illustrating the relationships between the various classes on the system. The scheduling system is organized into courses that are well-coordinated.

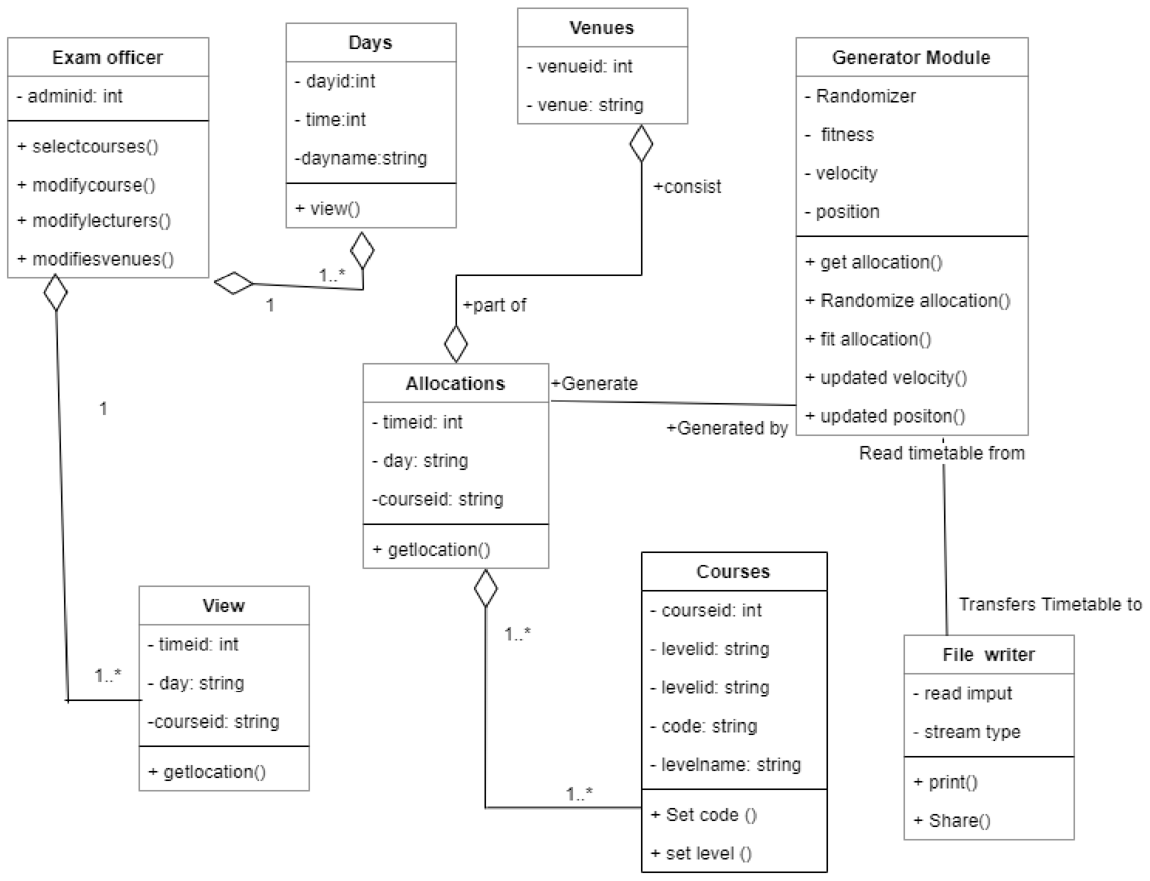


Figure 3.6: Class Diagram of the System

3.6 Performance Metrics

The quantity of meaningful work completed by a computer system is measured by performance metrics. It is calculated based on the correctness, efficiency, and speed with which computer program instructions are executed. The various assessment metrics and testing performed on the prototype are detailed in this section. The metrics to be used to assess the system's performance:

3.6.1 Efficiency

Task time is used to calculate efficiency. That is, the amount of time (in seconds and/or minutes) it takes the participant to perform a job effectively. The time needed to achieve an undertaking is controlled by deducting the beginning time from the completion time, as stated in the equation: $Task\ Time = End\ Time - Start\ Time$ Eqn 3.1

3.6.2 Effectiveness

The precision and thoroughness with which users attain their objectives. The completion rate can be used to calculate effectiveness. The completion rate, also known as the fundamental usability measure, is determined by giving a binary value of ‘1' if the test participant completes a job and ‘0' if he or she does not.

Effectiveness can thus be represented as a percentage by using this simple equation:

$$Effectiveness = (Number\ of\ tasks\ completed\ successfully \div Total\ number\ of\ tasks\ undertaken) * \ 100\% \qquad \text{Eqn 3.2}$$

3.6.3 Response time: is defined to mean the time between when the user initiates an action, and when the computer starts to display the result. It can be expressed with the mathematical formula:

$$Average\ Response\ Time := \sum time\ of\ transaction\ duration\ /\ the\ number\ of\ transactions\ started \qquad \text{Eqn 3.4}$$

CHAPTER FOUR

IMPLEMENTATION, RESULTS, AND DISCUSSION

4.1 Introduction

The phase of system implementation is when the designs created in Chapter 3 are developed and coded into a computer language. Both the front-end and back-end of the proposed system are created during this phase. As a result, this chapter describes the schedule generating software's implementation and test output.

4.2 System Requirement for Development

System requirement is a fundamental requirement for any software system that was developed. The prototype of the timetable generating software was developed and implemented using object-oriented programming language-Kotlin and MySQL database was the database used.

4.2.1 Hardware Requirements

The timetable generation system was designed and developed on Personal Computer (laptop) with intel core i5 Central Processing Unit with the configuration shown in table 4.1.

Table 4.1: Hardware Requirement

Hardware	Specification
Elite book Laptop	8 GB RAM
Hard Disk space	320GB
Processor speed	2.5GHz
Keyboard and mouse, if the system is to run on desktop	QWERTY Laptop keys

4.2.2 Software Requirements

The timetable generation system was executed or test-run on a system with a higher configuration. Nevertheless, for the system to perform optimally, the minimum requirements are a laptop with a minimum of duo core and above, that has a RAM of at least 3GB. The system must have a hard drive with at least 30GB, with a speed of not less than 1.2 GHz. The system must have a Java Development Kit (JDK) installed. The software used for the development of the timetable generation system and its specifications are shown in Table 4.2.

Table 4.2: Software Requirement

Hardware	Specification
Operating System	Windows 10
Java Development Kit	JDK 8
Relational Database Management system	MySQL database
Front end Language	Kotlin
Visual Paradigm for UML	Draw.io

4.2.3 Development Tools

Various tools were used throughout the development of the prototype of the schedule generation system to efficiently develop and improve the system's quality viewpoint and output. The tools are included in Table 4.3, along with an explanation and description.

Table 4.3: Descriptive Table on Tools used for the Development

Tools Used	Description
Java development Tool (Android studio and Kotlin)	Creates an enabling environment that makes application development easy. The tool suites provide built-in combinations of language support, framework support, and runtime support.
Draw.io	Designed diagrams described at the design development phase were designed with the aid of these tools.

4.2.4 Deployment Platform

When it is time to deploy and evaluate the prototype of the timetable generation system, the platform or the technologies that supported the deployment and their description are shown in Table 4.4.

Table 4.4: Deployment Platform

Technology	Specification
Database	MySQL database
Number of servers	1 (one)
Security	SSL and CAPTCHA to be implemented
Operating system	Window 10

4.3 System Menus Implementation

The implementation of system menus in this project includes creating a graphical user interface for simple access by the proposed system's intended users. The menus implementation is the use of programming code to convey the system design. The primary system menus of the timetable system are described and explained in this section. Figure 4.1 depicts the system's initial interface, which serves as a user interface for the program.

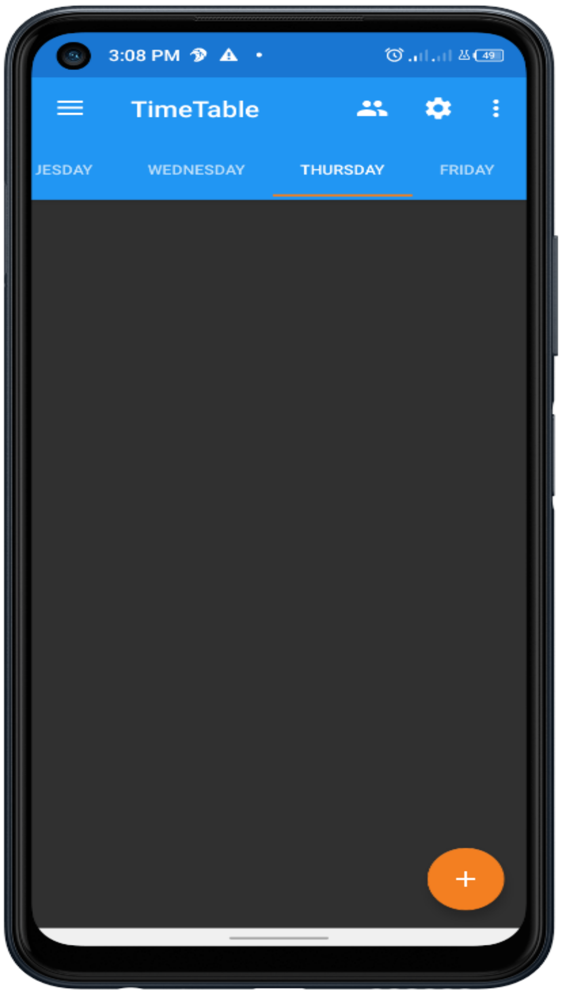
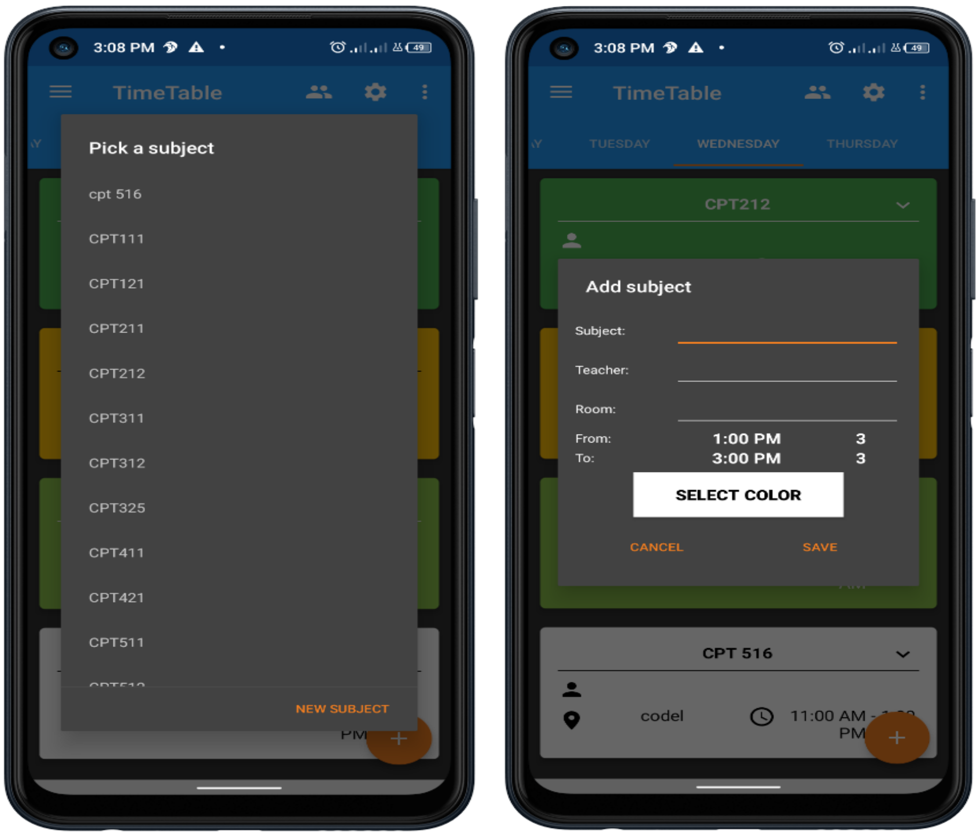


Figure 4.1: Home Page

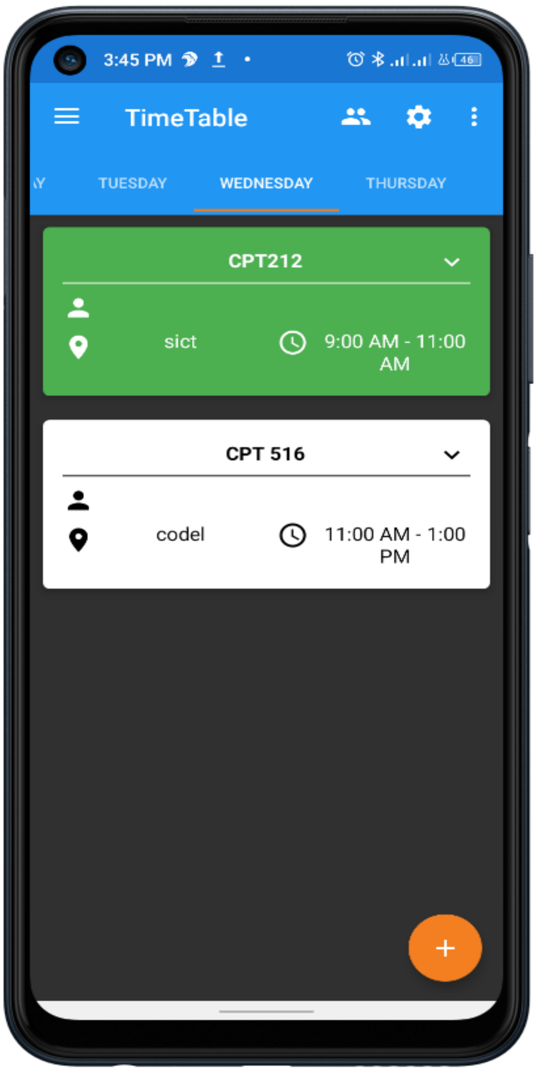
As shown in figure 4.1, A plus button could be seen on the interface. The button is for adding and selecting predefined subjects to the timetable.



From: 1:00 PM 3
To: 3:00 PM 3

Figure 4.2: a) The select subject screen b) The Add subject screen

The select subject screen is depicted in Figure 4.2 on clicking the plus button, a choice list view appears showing which course to add. On selection of a course, or clicking on 'new course', the add subject screen gets displayed, allowing the user to enter a subject, teacher, room, and time. The add screen is depicted in Figure. 4.2b). From this page, users can add new courses, venues, and lecturers.



CPT212
sict
9:00 AM - 11:00 AM

CPT 516
codel
11:00 AM - 1:00 PM

Figure 4.3: The display screen

The display page is depicted in Figure 4.3. This page displays the input processed by the Dean/HOD, it has a drop-down menu at the top right which enables the user to edit or delete a particular input.

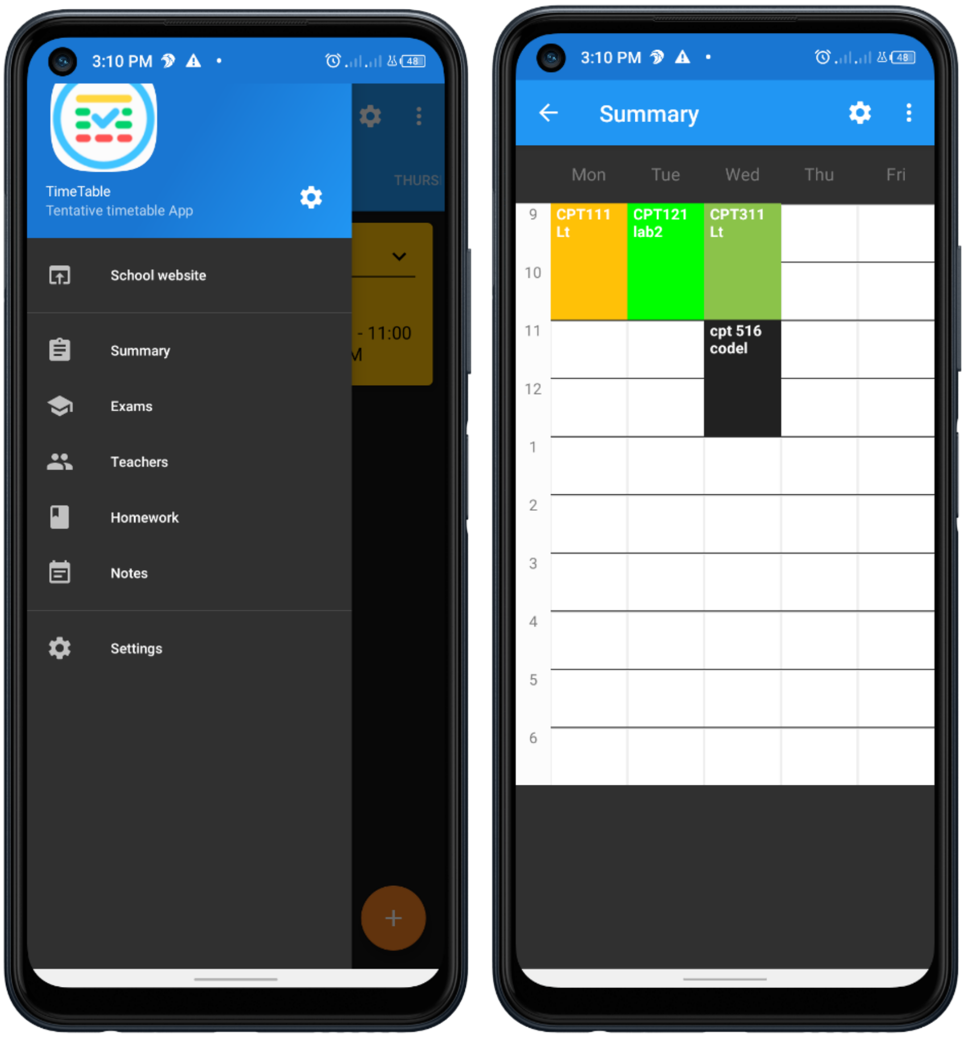


Figure 4.4: Summary Screen (a) and preview screen (b)

The summary page is depicted in Figure 4.5. From this page, user can preview the final work using the summary page which is located on the menu page at the left top of the system which leads to the preview of the final work

4.4 System Testing

The system testing phase in software design is exactly that phase when the user interfaces are connected to produce seamless navigation and flow of the system as a unified unit. During system testing, all of the system's functionalities or modules are integrated and linked to the database. System testing is typically performed in software systems to ensure that each function works as expected and is error-free. Testing procedures used in this project are listed below:

4.4.1 Unit Testing

Unit testing is the examination of several software components or modules that comprise the time-tabling system. The objective is to certify that each entity of the software code works as it should. Unit testing is often conducted by the software developers of each module in a large software business as part of a test-driven development. In this project, the researcher performed unit testing during the system's coding (development) phase. We isolated a piece of code and verified its validity.

4.4.2 Whole System Testing

System testing entails testing the entire system by interfacing the system's hardware and software components to find errors with the system. We used real data to test the system output during system testing. We examined the overall outcome using several data sets. Section 4.3 covered system testing of the proposed timetabling system and its output.

4.4.3 Usability Testing

Usability testing is a method of determining a system's user-friendliness in the context of software development. Usability testing was carried out in this study utilizing a set of smartphones in a classroom setting. The basis for choosing these individuals was that many of them have smartphones and can readily comprehend and operate the suggested system. Their response to the usability test shows that the system is extremely adaptable and capable of achieving the aim and goal.

Table 4.5 gives a summary of the test result for all the tested features. From table 4.5 it can be seen that all the features passed the unit, integration, and system testing successfully.

Table 4.5: Test Case Summary

Description	Test case	Test Result
Ensuring user access the system interface	1	Pass
Ensuring data are selected	2	Pass
Ensuring data are entered	3	Pass
Ensuring data are deleted	4	Pass
Ensuring data are modified	5	Pass
Ensuring data are stored	6	Pass
Ensuring data are previewed and available	7	Pass

A summary of the total number of tests conducted, the number of tests failed and the ones passed, the percentage of the tests passed and failed are shown in figure 4.6.

Table 4.6: Test Report Summary

Summary of test conducted	Results
Counts of Tested functions	7
Counts of Functions not tested	0
Counts of Tests passed	7
Counts of Tests that failed	0
Passed test percentage	100%
Failed test percentage	0%

4.5 Performance Evaluation

Before assessing the system, evaluation cases were developed for each module to be assessed. Various test cases were used to assess the system. Following the test run, there was a variety of data indicating that the new system was great and successful. To evaluate the suggested system's performance, 15 users were given access to execute timetabling activities on several cell phones.

Table 4.7: Performance Metrics

Performance metrics	Results
Average response time	1.07 seconds
Effectiveness	100%
Efficiency	0.82 seconds

Table 4.7 displays a feedback summary of system performance based on the performance measures mentioned in sections 3.6.1, 3.6.2, and 3.6.3.

CHAPTER FIVE

SUMMARY, CONCLUSION, AND RECOMMENDATION

5.1 Summary

The experimental study shows that PSO is capable of producing the university course timetable based on the outcomes. The created approach takes into account the issue of satisfying stakeholders by lowering the cost objective to meet the university policy with favourable outcomes in a fraction of a second compared to the manual procedure.

Part one covers presentation, the foundation of the investigation, explanation of issue, point, and goals, the meaning of the examination, and the degree and impediment of the examination. Section two covers an audit of related works and the outline of survey to this examination work. Section three covers the investigation of the proposed framework. It shows the framework design of the framework, utilize a case graph, grouping outline, and flowchart chart of significant segments of the framework Section four covers the executed framework and conversation of the outcomes gotten from the framework. Section five shows the rundown, end, proposed suggestion, and future work for the exploration work.

5.2 Conclusion

The Timetable Scheduler as an application for generating lecture timetables has been effectively and successfully deployed. The Timetable system is capable of producing near-optimal schedules depending on courses with reduced course constraints. The system timetable system allows users access irrespective of their location, provided the device is within their reach.

The PSO algorithm, which includes characteristics such as particles, particle placement, fitness functions, local best, global best, and velocity for timetabling, was implemented to fulfil the first goal, which was to develop the PSO algorithm for the system. This method was used to build the timetabling systems as described on page 36, and it addressed both hard and soft restrictions to arrive at an ideal solution.

Objective 2 was met by using various structural representations of the system design approaches such as the flow chart, architectural design, modified RAD methodology, and UML diagram used as a lower prototype of the system that showed the system flow and admin interaction with the system as specified on pages 41, 33, and 42.

As stated in objective 3, the developed system demonstrates the accomplishment of the project's goal, which is to implement timetabling systems using a suitable programming language, such as Kotlin for the front end and MySQL for the database, as stated in objective iii and as specified on page 48

Objective 4: A thorough evaluation was carried out by ensuring that thorough testing was conducted on different smartphones with different models, as well as testing on systems with different screen resolutions to ensure it met the requirements, to ensure the system was user friendly, ensuring that the application did not consume much phone memory space, and ensure the system is bug-free and deliver the expected outcome, as mentioned in objective 4 and on page 51.

5.3 Recommendation

With this motion, timetable management has been made simpler as it is a sound utilization to create and goods timetable in appearance information that can be easily mutual on enabled devices. For recommendation, after having understudied the difficulties recovered in the manual timetable system, PSO Algorithm should be taught as a course for Computer Science students due to its complexity. Students will have a better grasp of the PSO Algorithm as a result of this.

5.4 Suggestion for Future Works

As future work, the author intends to look at the option of the timetable system improved to be large to handle the timetable management in the higher infirmary of the whole division (academic) and prefab easy to all students.

REFERENCE

Imran, M., Hashim, R., & Khalid, N. E. A. (2013). An overview of particle swarm optimization variants. *Procedia Engineering*, *53*(1), 491–496. https://doi.org/10.1016/j.proeng.2013.02.063

Imran, S., Akhand, M. A. H., Shuvo, M. I. R., Siddique, N., & Adeli, H. (2019). Optimization of University Course Scheduling Problem using Particle Swarm Optimization with Selective Search. *Expert Systems With Applications*, *127*, 9–24. https://doi.org/10.1016/j.eswa.2019.02.026

Chuen, teh yung. (2015). *Dynamic Timetable Generator Using Particle Swarm Optimization (Pso) Method*. 1–24.

Bhattacharjya, R. K. (2012). Introduction to Particle Swarm Optimization. *Indian Institute of Tech*, 1–10.

Rathod, S., Saha, A., & Sinha, K. (2020). *Particle Swarm Optimization and its applications in agricultural research*. *November*.

Chen, R., & Shih, H. (2013). *Solving University Course Timetabling Problems Using Constriction Particle Swarm Optimization with Local Search*. 227–244. https://doi.org/10.3390/a6020227

Chu, S., & Chen, Y. (2006). *Timetable Scheduling Using Particle Swarm Optimization*. 9–12.

Diepen, G., Everlo, T. S., & Bouazzaoui, H. (2018). Part 2 : Artificial Intelligence Techniques Explained | Zooming in on fundamental AI techniques. *Deloitte* \, 1–12. https://www2.deloitte.com/se/sv/pages/technology/articles/part2-artificial-intelligence-techniques-explained.html

Pedamkar, B. P. (n.d.). *Arti cial Intelligence Techniques Introduction to What is Arti cial Intelligence ? Top 4 Techniques of Arti cial Intelligence*.

Haenlein, M., & Kaplan, A. (2019). A brief history of artificial intelligence: On the past, present, and future of artificial intelligence. *California Management Review*, *61*(4), 5–14. https://doi.org/10.1177/0008125619864925

Ceo, G., & Pichai, S. (2018). *The Emergence of Artificial Intelligence*.

Heuristics, W. A., & Heuristics, U. (2021). *What Are Heuristics?*

Ekanayake, T. W., Subasinghe, P., Ragel, S., Gamage, A., & Attanayaka, S. (2019). Intelligent Timetable Scheduler: A Comparison of Genetic, Graph Coloring, Heuristic and Iterated Local Search Algorithms. *2019 International Conference on Advancements in Computing, ICAC 2019*, 85–90. https://doi.org/10.1109/ICAC49085.2019.9103403

Mallick, S., Majumdar, D., Mukherjee, S., & Bhattacharjee, A. K. (2020). Hybrid cuckoo search approach for course time-table generation problem. *International Journal of Applied Metaheuristic Computing, 11*(4), 214–230. https://doi.org/10.4018/IJAMC.2020100110

Hambali, A. M., Olasupo, Y. A., & Dalhatu, M. (2020). Automated university lecture timetable using Heuristic Approach. *Nigerian Journal of Technology, 39*(1), 1–14. https://doi.org/10.4314/njt.v39i1.1

Arratia-martinez, N. M., Maya-padron, C., & Avila-torres, P. A. (2021). *University Course Timetabling Problem with Professor Assignment*. *2021*.

Thepphakorn, T., & Pongcharoen, P. (2019). *Variants and Parameters Investigations of Particle Swarm Optimisation for Solving Course Timetabling Problems* (Vol. 2). Springer International Publishing. https://doi.org/10.1007/978-3-030-26369-0

Grosz, B. J., Altman, R., Horvitz, E., Mackworth, A., Mitchell, T., Mulligan, D., & Shoham, Y. (2016). *Artificial intelligence and life in 2030: One hundred year study on artificial intelligence*.

Kamble, R., & Shah, D. (2018). Applications of Artificial Intelligence in Human Life. *International Journal of Research -GRANTHAALAYAH*, *6*(6), 178–188. https://doi.org/10.29121/granthaalayah.v6.i6.2018.1363

Techie-menson, H., & Nyagorme, P. (2021). *Design and Implementation of a Web-Based Timetable System for Higher Design and Implementation of a Web-Based Timetable System for Higher Education Institutions*. *7*(March), 1–13.

Eberhart, R. C., & Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. *Proceedings of the 2000 Congress on Evolutionary Computation, CEC 2000*, *1*(February 2015), 84–88. https://doi.org/10.1109/CEC.2000.870279

R. Mart´ı and G. Reinelt, *The Linear Ordering Problem*, Exact and Heuristic Methods in Combinatorial Optimization 175, DOI: 10.1007/978-3-642-16729-4 2, Springer-Verlag Berlin Heidelberg 2011

Aziz, N. L. A., & Aizam, N. A. H. (2018). A brief review on the features of university course timetabling problem. *AIP Conference Proceedings*, *2016*(September). https://doi.org/10.1063/1.5055403

Bansal, J. C. (n.d.). *Particle Swarm Optimization*. Springer International Publishing. https://doi.org/10.1007/978-3-319-91341-4

Adrianto, D. (2014). *COMPARISON USING PARTICLE SWARM OPTIMIZATION AND GENETIC*. *10*(2), 341–346. https://doi.org/10.3844/jcssp.2014.341.346

Budi Darmawan, V. E., Chen, Y. W., Larasati, A., Prastyo, D., & Dwiastuti, A. (2020). *Multi-objective Modeling for a Course Timetabling Problem*. *January*, 10–14. https://doi.org/10.5220/0009857300100014

Naderi, B. (2017). *Modeling and Scheduling University Course Timetabling Problems*. *5*(1), 1–15. https://doi.org/10.22105/riej.2017.49167

Raju, D., & Mallesh, V. (2016). *An Experimental On Time Table Generator*. *5*(11), 18755–18770. https://doi.org/10.18535/ijecs/v5i11.05

Marada, S. R., Kaki, L. P., & Pilaka, A. (2019). Automatic Timetable Generation Using Pbil Algorithm. *I-Manager's Journal on Information Technology*, *8*(2), 31. https://doi.org/10.26634/jit.8.2.15514

Appendix

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="15dp">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"
        android:orientation="horizontal">
        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".3"
            android:text="@string/subject_dialog"
            android:textAlignment="textStart"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="14sp" />
        <EditText
            android:id="@+id/subject_dialog"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".7"
            android:imeOptions="actionDone"
            android:singleLine="true"
            android:textColor="?android:attr/textColorPrimary" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"
        android:orientation="horizontal">
        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".3"
            android:text="@string/teacher_dialog"
            android:textAlignment="textStart"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="14sp" />
        <EditText
            android:id="@+id/teacher_dialog"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".7"
            android:imeOptions="actionDone"
            android:singleLine="true"
            android:textColor="?android:attr/textColorPrimary" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"
        android:orientation="horizontal">
        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".3"
            android:text="@string/room_dialog"
            android:textAlignment="textStart"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="14sp" />
        <EditText
            android:id="@+id/room_dialog"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".7"
            android:imeOptions="actionDone"
            android:inputType="textVisiblePassword"
            android:singleLine="true"
            android:textColor="?android:attr/textColorPrimary" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"
        android:orientation="horizontal">
        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".3"
            android:text="@string/time_from_dialog"
            android:textAlignment="textStart"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="14sp" />
        <TextView
            android:id="@+id/from_time"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".45"
            android:background="?attr/selectableItemBackground"
            android:gravity="center"
            android:text="@string/select_start_time"
            android:textAllCaps="true"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="18sp"
            android:textStyle="bold" />
        <TextView
            android:id="@+id/from_hour"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".25"
            android:background="?attr/selectableItemBackground"
            android:gravity="center"
            android:text="@string/lesson"
            android:textAllCaps="true"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="18sp"
            android:textStyle="bold" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"
        android:orientation="horizontal">
        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".3"
            android:text="@string/time_to_dialog"
            android:textAlignment="textStart"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="14sp" />
        <TextView
            android:id="@+id/to_time"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".45"
            android:background="?attr/selectableItemBackground"
            android:gravity="center"
            android:text="@string/select_end_time"
            android:textAllCaps="true"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="18sp"
            android:textStyle="bold" />
        <TextView
            android:id="@+id/to_hour"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight=".25"
            android:background="?attr/selectableItemBackground"
            android:gravity="center"
            android:text="@string/lesson"
            android:textAllCaps="true"
            android:textColor="?android:attr/textColorPrimary"
            android:textSize="18sp"
            android:textStyle="bold" />
    </LinearLayout>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center_vertical"
    android:orientation="horizontal">
    <Button
        android:id="@+id/select_color"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="50dp"
        android:layout_marginTop="5dp"
        android:layout_marginEnd="50dp"
        android:layout_marginBottom="5dp"
        android:background="@color/white"
        android:text="@string/select_color"
        android:textAlignment="center"
        android:textAllCaps="true"
        android:textSize="18sp"
        android:textStyle="bold" />
</LinearLayout>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <Button
        android:id="@+id/cancel"
        style="@style/Widget.AppCompat.Button.ButtonBar.AlertDialog"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
```

```
            android:text="@string/cancel" />
        <Button
            android:id="@+id/save"
            style="@style/Widget.AppCompat.Button.ButtonBar.AlertDialog"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="@string/save" />
    </LinearLayout>
</LinearLayout>
```