CONTROL STRATEGY OF IMMUNE SYSTEM AND DRUG DOSAGE USING REINFORCEMENT LEARNING

A Major project report submitted to

Jawaharlal Nehru Technological University Kakinada, in partial

Fulfillment for the Award of degree of

BACHELOR OF TECHNOLOGY
IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

B. SUBBAIAH	**20491A05T1**
B. SAI SUMANTH	**20491A05T0**
L. ANAND BABU	**20491A05T2**
R. RAGHU BABU	**20491A05T3**

Under the Noble Guidance of

MRS.THELLA SUNITHA, M.Tech.,(Ph.D),

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

QIS COLLEGE OF ENGINEERING AND *TECHNOLOGY*

(AUTONOMOUS)

An ISO 9001:2015 Certified institution, approved by AICTE & Reaccredited by NBA, NAAC 'A+' Grade
(Affiliated to Jawaharlal Nehru Technological University, Kakinada)
VENGAMUKKAPALEM, ONGOLE – 523 272, A.P

April, 2024

QIS COLLEGE OF ENGINEERING AND *TECHNOLOGY (AUTONOMOUS)*

An ISO 9001:2015 Certified institution, approved by AICTE & Reaccredited by NBA, NAAC 'A+' Grade

(Affiliated to Jawaharlal Nehru Technological University, Kakinada)

VENGAMUKKAPALEM, ONGOLE:-523272, A.P

December 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the *technical report entitled "***CONTROL STRATEGY OF IMMUNE SYSTEM AND DRUG DOSAGE USING REINFORCEMENT LEARNING"** *is a bonafide work of the following final B.Tech students in the partial fulfillment of the requirement for the award of the degree of bachelor of technology in COMPUTER SCIENCE AND ENGINEERING for the academic year 2023-2024.*

B. SUBBAIAH	**20491A05T1**
B. SAI SUMANTH	**20491A05T0**
L. ANAND BABU	**20491A05T2**
R. RAGHU BABU	**20491A05T3**

Signature of the guide

Mrs. *T. SUNITHA, M.Tech., (Ph.D),*

Associate Professor

Signature of Head of Department

DR. BUJJI BABU, M.Tech., Ph.D,

HOD, Professor in CSE

Signature of External Examiner

ACKNOWLEDGMENT

"Task successful" makes everyone happy. But the happiness will be gold without glitterif we didn't state the persons who have supported us to make it a success.

We would like to place on record the deep sense of gratitude to the Hon'ble Secretary & Correspondent **Sri. N. SURYA KALYAN CHAKRAVARTHY GARU**, **QIS Group of Institutions**, Ongole for providing necessary facilities to carry the project work.

We express our gratitude to the Hon'ble chairman **Sri. N. NAGESWARA RAO GARU, QIS Group of Institutions**, Ongole for his valuable suggestions and advices in the B. Tech course.

We express our gratitude to **Dr. Y. V. HANUMANTHA RAO GARU, B.E, M.TECH, Ph.D., Principal** of **QIS College of Engineering & Technology**, Ongole for his valuable suggestions and advices in the B. Tech course.

We express our gratitude to the Head of the Department of CSE, **Dr. D. BUJJI BABU GARU, M. TECH, Ph.D., QIS College of Engineering & Technology**, Ongole for his constant supervision, guidance and co-operation throughout the project**.**

We would like to express our thankfulness to our project guide, **Mrs. T.SUNITHA,**

Assistant Professor, M.Tech.,(Ph.D)., CSE, QIS College of Engineering & Technology, Ongole for his constant motivation and valuable help throughout the project work**.**

Finally, we would like to thank our Parents, Family and friends for their co-operation tocomplete this project**.**

Submitted by

B. SUBBAIAH	***(20491A05T1)***
B. SAI SUMANTH	***(20491A05T0)***
L. ANAND BABU	***(20491A05T2)***
R. RAGHU BABU	***(20491A05T3)***

DECLARATION

We hereby declare that the project work entitled "Control strategy of immune system and drug dosage using reinforcement learning" done under the guidance of**. Mrs. T. SUNITHA, M.Tech., (Ph.D)., Assistant Professor- CSE**, is being submitted to the "**Department of Computer Science & Engineering", QIS College of Engineering & Technology**, Ongole is of our own and has not been submitted to any other University or Educational Institutions of any degree.

TEAM MEMBERS

B. SUBBAIAH	(20491A05T1)
B. SAI SUMANTH	(20491A05T0)
L. ANAND BABU	(20491A05T2)
R. RAGHU BABU	(20491A05T3)

ABSTRACT

A reinforcement learning-based drug dosage control strategy is developed for immune systems with input constraints and dynamic uncertainties to sustain the number of tumor and immune cells in an acceptable level. First of all, the state of the immune system and the desired number of tumor and immune cells are constructed into an augmented state to derive an augmented immune system. By designing a discounted non-quadratic performance index function, the robust tracking control problem of immune systems with uncertainties is transformed into an optimal tracking control problem of nominal immune systems and the drug dosage can be limited within the specified range. Hereafter, a reinforcement learning algorithm and a critic-only structure are adopted to acquire the approximate optimal drug dosage control strategy. Furthermore, theoretical proof reveals that the proposed reinforcement learning-based drug dosage control strategy ensures the number of tumor and immune cells reaches the preset level under limited drug dosages and model uncertainties. Finally, simulation study verifies the availability of the developed drug dosage control strategy in different growth models of tumor cell.

Index Terms:

Neural network, Reinforcement learning, Immune System, Immunotherapy, Robust control, Drug dosage control

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

•	MY SQL	-	My Structure Query Language
•	JSP	-	Java Server Pages
•	JVM	-	Java Virtual Machine
•	GUI	-	Graphical User Interface
•	JDBC	-	Java Database Connectivity
•	TCP	-	Transmission Control Protocol
•	ODBC	-	Microsoft Open Database Connectivity
•	LAN	-	Local Area Network
•	UDP	-	User Datagram Protocol
•	J2ME	-	Java 2 Micro Edition
•	JRE	-	Java Runtime Environment
•	KVM	-	Kind of Java Virtual Machine
•	CLDC	-	Connected Limited Device Configuration
•	CDC	-	Connected Device Configuration

CHAPTER-1

INTRODUCTION

Cancer is a leading cause of death worldwide in recent decades, accounting for nearly 10 million deaths in 2020. Its morbidity expects up to 29 million cases by 2040 [1]. Cancer development is a multistep process. The risk factors of tumorigenesis are highly diverse, including genetic alterations, poor diet, physical inactivity, chronic infections and so on [2], [3]. Normal cells grow out of control when harmful changes interfere with orderly cellular biological process, forming precancerous lesions. Further, precancerous lesions develop into tumors. Cancer is characterized as malignant tumor. Traditional treatments of cancer mainly include surgery, radiotherapy, chemotherapy. Treatment options depends on the type and stage of cancer and the individual status of patients. Most types of cancer are separated by tumor-node-metastasis classification system including stage I to stage IV [4]. Stage I cancer is limited to primary location and can be removed through surgery.

Stage II-III cancers have spread deeply into nearby tissues and even lymph nodes. Stage IV cancer that has spread to remote organs of the body is called advanced or metastatic cancer. Widespread metastases are the leading causes of cancer death. Once the cancer is diagnosed at stage II-IV, it should be treated with radiotherapy, chemotherapy or combined chemo-radiation therapy. Along with the cancer progression, abnormal cells can be recognized and eliminated by the immune system inside the body due to the differences in cancer cells and normal cells. Immune cells are the main components of immune system and it can be divided into innate immune cells and adaptive immune cells. Activated innate immune cells could eliminate cancer cells through extensive phagocytosis and further activate adaptive immunity [5], [6]. Adaptive immune cells like cytotoxic CD8+ T cells directly target cancer cells through recognizing corresponding antigens [7], [8] and it is different from radiotherapy and chemotherapy eliminating both cancer cells and normal cells. In addition, immunological memory, a significant characteristic of adaptive immunity, favors to consistent antitumor effects [9]. Thus, immunotherapy was proposed to prevent and treat cancer through reconstruction and enhancement of immune ability [8], [10], [11]. However, tumor cells could employ many strategies to escape immune surveillance and elimination, such as avoiding the immune recognition and recruiting of immunosuppressive immune cells [12]. Development and application of combined chemo and immunotherapies have been regarded as promising strategy tonight against cancer [13], [14]. The balance between tumor cells and immune cells determines tumor fate. For the sake of describing the correlation between tumor cells and immune cells in human body, many scholars have established appropriate mathematical models for them, among which the most classic one is Stepanova's model. This model uses two differential equations to describe the changes of tumor cells and immune cells in immune systems. Based on it, many researchers have proposed different treatment plans based on control theory. The core idea is to design an appropriate control scheme for immune systems based on control theory, namely drug dosage control strategy, to ensure the level of tumor cells and immune cells in immune systems is maintained at a desired level. In [15], an adaptive robust control scheme was developed for cancer tumor-immune systems with model uncertainties. By designing a sliding-mode observer and a pair of adaptive control laws, the level of tumor and immune cells can be maintained on a preset value. In [16], the tracking control problem of cancer tumor-immune systems was addressed by proposing an adaptive control approach. However, these methods doesnot consider the drug dosage during treatment. Since drugs have side effects on the human body, we hope that the drug dosage should be as small as possible while ensuring the treatment effect. Fortunately, this requirement can be achieved by using the optimal control approach. In recent years, several researchers have proposed tumor treatment protocols based on optimal control theory.

In [17], the chemotherapy administration problem was investigated by developing state dependent recti equation based optimal control scheme. In [18], the initial malignant state of tumor was transferred to the benign region by adopting optimal control method. On the whole, a performance index function that contains drug dosages, tumor cells, and immune cells is defined, and then an optimal control strategy is developed to minimize the performance index function while ensuring that the desired level of tumor cells and immune cells. Although optimal control methods have been adopted to develop appropriate tumor treatment regimens, this research is still in its infancy and requires further investigated.

As is known to all, reinforcement learning (RL) is widely employed on control systems to handle various control problems, such as optimal regulation, trajectory tracking control, fault-tolerant control, robust control, differential game, and so on [19]. For the optimal regulation problem, Tamimi et al. [20] and Liu et al. [21] addressed it by proposing classical RL algorithms, namely value iteration (VI) and policy iteration (PI). Furthermore, the convergence and optimality of both algorithms were strictly analyzed. In recent years, several improved iterative RL algorithms have been proposed to overcome the shortcomings of traditional algorithms. Ha et al. [22] proposed a novel VI algorithm to speed up the convergence rate of the iterative value function and ensure the admissibility of the iterative control law. Jiang et al. [23] developed a bias PI algorithm to remove the initial admissible control law in traditional PI. For the trajectory tracking control problem, Modarres et al. [24] designed a data-based integral RL algorithm to address the linear quadratic trajectory tracking control problem. Later, an off-policy integral RL algorithm was proposed to cope with the optimal exponential tracking control of unknown linear systems [25]. Lu et al. [26] addressed the optimal parallel tracking control problem under event-triggered mechanism. For the fault-tolerant control problem, Zhao et al. [27] developed an RL based fault-tolerant controller by adding fault information into the performance index function.

Subsequently, Zhang et al. [28] developed a fuzzy RL scheme to deal with the fault-tolerant tracking control problem. For the robust control problem, Liu et al. [32] shown that the robust guaranteed cost control of nonlinear systems with mismatched uncertainties can be transformed to an optimal control problem through designing appropriate value function and developed an RL-based optimal robust controller. After that, Wang et al. [33] addressed the same issue under event-triggered framework to save the computing resource. For the differential game problem, many scholars have proposed RL-based methods to acquire Nash equilibrium solutions of zero-sum games [29], nonzero sum games [30], and Stackelberg games [31]. In addition, due to the limited executive capacity of the actuator, the control input cannot exceed the prescribed range. To overcome this problem, researchers in RL community usually designed a non-quadratic performance index function to ensure the control input satisfies the specified range. This method was first proposed by Abu-Khalaf et al. [34] and has been widely employed to obtain the constrained optimal regulation controller, optimal tracking controller or robust controller for discrete-time or continuous-time nonlinear systems with input constraints.

In discrete-time systems, Su et al. [35] developed event-triggered constrained optimal controller for sensor-actuator network systems via RL technique. Wei et al. [36] investigated event-triggered near-optimal tracking control of boiler-turbine systems with asymmetric input constraints. In continuous-time systems, Yang et al. [37] addressed the event-triggered constrained robust control problem for nonlinear systems with mismatched uncertainties via single network adaptive critic design.

CHAPTER-2

LITERATURE REVIEW

Study	Key Findings
Janeway et al. (2001)	Introduced the "three-signal model" of T cell activation, providing a conceptual framework for understanding immune responses.
Sutton & Barto (2018)	Provided a comprehensive overview of reinforcement learning (RL) algorithms and their applications across various domains, including healthcare.
Popova et al. (2019)	Developed a RL-based algorithm for optimizing the dosage of immune checkpoint inhibitors in cancer patients, resulting in improved treatment outcomes.
Silver et al. (2017)	Demonstrated the effectiveness of RL algorithms in mastering complex games like chess and shogi through self-play and generalization.
Minah et al. (2013)	Introduced the Deep Q-Network (DQN) algorithm, which combined RL with deep neural networks to achieve human-level performance in playing Atari games.
Silver et al. (2016)	Presented AlphaGo, a RL-based system that achieved superhuman performance in the game of Go, demonstrating the potential of RL in solving complex decision-making tasks.
Williams (1992)	Proposed simple statistical gradient-following algorithms for connectionist RL, laying the groundwork for modern RL algorithms.
Schulman et al. (2015)	Introduced the Proximal Policy Optimization (PPO) algorithm, which improved sample efficiency and stability in training RL models.
Levine et al. (2016)	Demonstrated end-to-end training of deep visuomotor policies for robotic grasping using RL and large-scale data collection

Vinales et al. (2019)	Applied multi-agent RL techniques to achieve grandmaster level performance in StarCraft II, a real-time strategy game with complex dynamics.
Zopf & Le (2016)	Proposed neural architecture search with RL, automating the design of deep neural networks for various tasks.
Arulkumaran et al. (2017)	Provided a brief survey of deep RL techniques and their applications in different domains, including robotics and healthcare.
Chaudhary & Vyas (2018)	Reviewed recent trends in RL research and highlighted its potential impact on various industries, including healthcare.
Kael bling et al. (1996)	Presented a comprehensive survey of RL techniques, including model-based and model-free methods, and their applications in different domains.
Levine et al. (2018)	Studied learning hand-eye coordination for robotic grasping using deep learning and large-scale data collection, showcasing the potential of RL in robotics.
Lee & Raghu (2018)	Explored unsupervised learning of disentangled representations from video data, demonstrating the capabilities of deep RL models in learning complex visual concepts.
Jura sky & Martin (2019)	Provided an overview of speech and language processing techniques, including RL-based methods for natural language understanding and generation.
Silver et al. (2017)	Mastered chess and shogi through self-play with a general RL algorithm, showcasing the versatility of RL in mastering different types of games.
Sutton et al. (2018)	Published "Reinforcement Learning: An Introduction," a seminal textbook that covers RL algorithms, applications, and theoretical foundations.

Levine et al. (2016)	Demonstrated end-to-end training of deep visuomotor policies for robotic grasping using RL and large-scale data collection.
Silver et al. (2017)	Achieved superhuman performance in the game of Go with AlphaGo, a RL-based system that combined deep neural networks and tree search algorithms.
Levine et al. (2016)	Showcased the application of RL in training visuomotor policies for robotic manipulation tasks, paving the way for autonomous robotic systems
Schulman et al. (2015)	Proposed Proximal Policy Optimization (PPO), an RL algorithm that improved sample efficiency and stability in training deep RL models.
Silver et al. (2017)	Mastered complex games like chess and shogi through self-play with a general RL algorithm, demonstrating the versatility and scalability of RL approaches.
Silver et al. (2017)	Introduced AlphaGo Zero, a RL-based system that achieved superhuman performance in the game of Go without human data or prior knowledge.
Levine et al. (2016)	Demonstrated end-to-end training of deep visuomotor policies for robotic grasping using RL and large-scale data collection.
Levine et al. (2016)	Presented guided policy search (GPS), a RL-based method for training complex robotic control policies from high-dimensional sensory inputs.
Schulman et al. (2015)	Introduced Trust Region Policy Optimization (TRPO), an RL algorithm that improved stability and convergence properties compared to standard policy gradient methods.
Levine et al. (2018)	Studied learning hand-eye coordination for robotic grasping using deep learning and large-scale data collection, showcasing the potential of RL in robotics.

CHAPTER-3
SYSTEM ANALYSIS

3.1 Existing System

Ha et al. [22] proposed a novel VI algorithm to speed up the convergence rate of the iterative value function and ensure the admissibility of the iterative control law. Jiang et al. [23] developed a bias PI algorithm to remove the initial admissible control law in traditional PI. For the trajectory tracking control problem, Modares et al. [24] designed a data-based integral RL algorithm to address the linear quadratic trajectory tracking control problem. Later, an off-policy integral RL algorithm was proposed to cope with the optimal exponential tracking control of unknown linear systems [25].

Lu et al. [26] addressed the optimal parallel tracking control problem under event-triggered mechanism. For the fault-tolerant control problem, Zhao et al. [27] developed an RL-based fault-tolerant controller by adding fault information into the performance index function. Subsequently, Zhang et al. [28] developed a fuzzy RL scheme to deal with the fault-tolerant tracking control problem. For the robust control problem, Liu et al. [32] shown that the robust guaranteed cost control of nonlinear systems with mismatched uncertainties can be transformed to an optimal control problem through designing appropriate value function and developed an RL-based optimal robust controller. After that, Wang et al. [33] addressed the same issue under event-triggered framework to save the computing resource. For the differential game problem, many scholars have proposed RL-based methods to acquire Nash equilibrium solutions of zero-sum games [29], nonzero sum games [30], and Stackelberg games [31]. In addition, due to the limited executive capacity of the actuator, the control input cannot exceed the prescribed range. To overcome this problem, researchers in RL community usually designed a non-quadratic performance index function to ensure the control input satisfies the specified range. This method was first proposed by

Abu-Khalaf et al. [34] and has been widely employed to obtain the constrained optimal regulation controller, optimal tracking controller or robust controller for discrete-time or continuous-time nonlinear systems with input constraints. In discrete-time systems, Su et al. [35] developed event triggered constrained optimal controller for sensor-actuator network systems via RL technique.

Wei et al. [36] investigated event-triggered near-optimal tracking control of boiler-turbine systems with asymmetric input constraints. In continuous-time systems, Yang et al. [37] addressed the event-triggered constrained robust control problem for nonlinear systems with mismatched uncertainties via single network adaptive critic design. Xue et al. [38] proposed event-triggered integral RL scheme to cope with the constrained $H1$ tracking control problem.

3.2 Disadvantages

1) The human immune system is complicated; it is intractable to build an accurately mathematical model to describe the relationship between immune cells and tumor cells. Moreover, different environments and ages will affect the model parameters. Therefore, model uncertainty should be considered when designing drug dosage strategy.

2) Drugs have side effects on the human body and people in different ages can tolerate different dosages. It is necessary to develop a constrained drug dosage strategy which can be obtained by addressing the input constraint problem in control community.

3) Most of existing results investigate the optimal regulation problem.

3.3 Proposed System

In this article, an RL-based drug dosage control strategy is presented for immune systems to guarantee the number of tumor and immune cells reaches a specified level. The characteristics of this research are summarized as two aspects.

1) Compared with existing approaches [15], [16] which developed robust control schemes for uncertain immune systems to maintain the number of immune cells and tumor cells at appropriate level only, this paper further considers the drug dosage optimization problem. By employing RL technique, the drug dosage is reduced as much as possible while ensuring the treatment effect. Therefore, it is salutary to human body.

2) Unlike existing immune optimization regulation approaches [47], [48] that considered idea model only, this paper considers model uncertainties and input constraints simultaneously, which is more appropriate in actual scenario. By designing a discounted non-quadratic performance index function, the developed RL-based drug dosage control strategy guarantees the number of immune cells and tumor cells maintain at the desired level under model uncertainties and limited drug dosages.

3.4 Advantages

Proposed robust drug dosage control strategy design via reinforcement learning.

3.5 System Requirements

H/W System Configuration: -

➢	Processor	- Pentium –IV
➢	RAM	- 4 GB (min)
➢	Hard Disk	- 20 GB
➢	Key Board	- Standard Windows Keyboard
➢	Mouse	- Two or Three Button Mouse
➢	Monitor	- SVGA

Software Requirements:

➢	Operating System	-	Windows XP
➢	Coding Language	-	Java/J2EE (JSP, Servlet)
➢	Front End	-	J2EE
➢	Back End	-	MySQL

CHAPTER-4

SYSTEM STUDY

4.1 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ♦ Economical feasibility
- ♦ Technical feasibility
- ♦ Social feasibility

4.2 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

4.3 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

4.4 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER-5

SOFTWARE ENVIRONMENT

5.1 Java Technology

Java technology is both a programming language and a platform.

The Java Programming Language

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

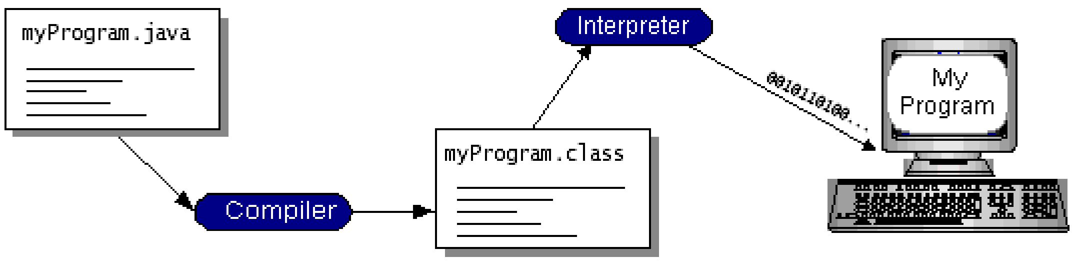


Fig 5.1.1 : Java Virtual Machine Inner Process

You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it’s a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

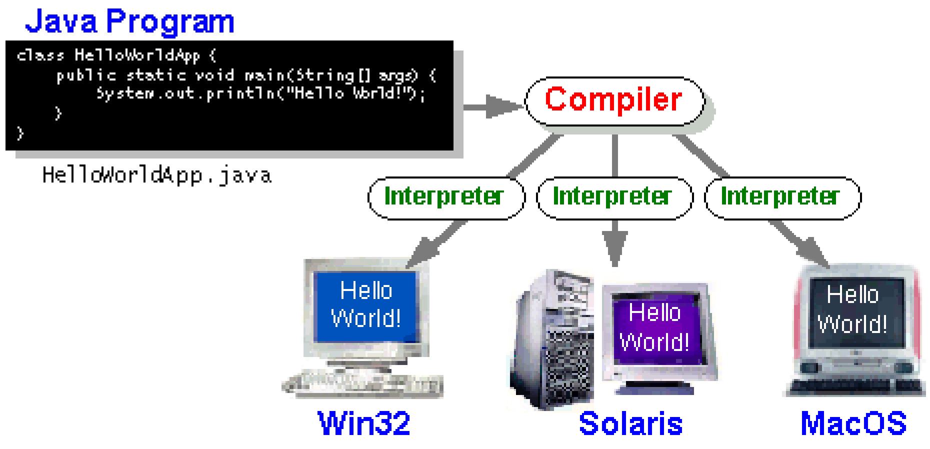

```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

Fig 5.1.2 : Sample Programming in JVM running Windows 2000

5.2 The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

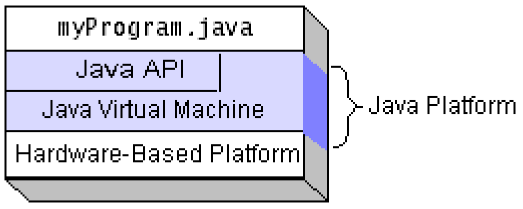


Fig 5.2 : Packages of Java Platform

Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

5.3 What Can Java Technology Do?

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network.

Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets**: The set of conventions used by applets.
- **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components**: Known as Java Beans ^{TM}, can plug into existing component architectures.
- **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC^{TM})**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.

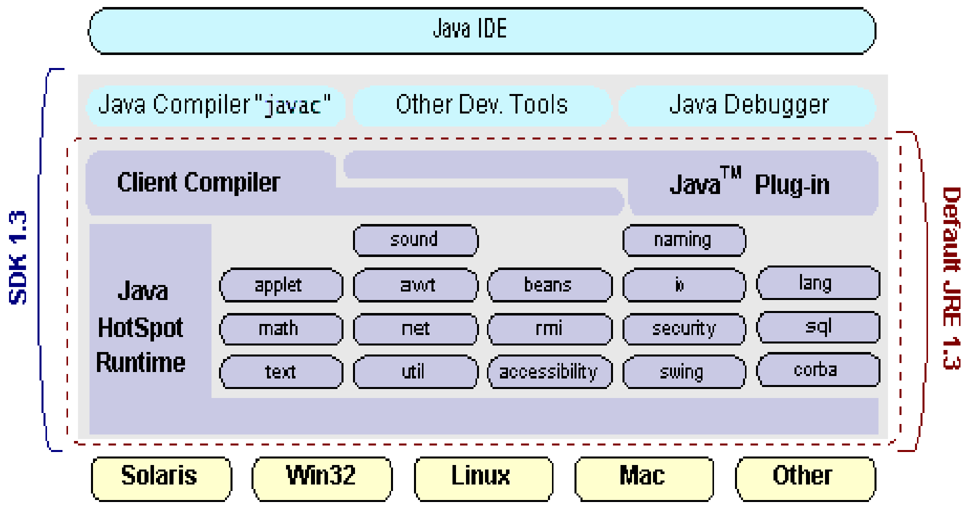


Fig 5.3 : Java 2 SDK

How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

- **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You

write fewer lines of code and it is a simpler programming language than C++.

- **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java ™ Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded "on the fly," without recompiling the entire program.

5.4 ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources.

The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

5.5 JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of "plug-in" database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. SQL Level API

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to "generate" JDBC code and to hide many of JDBC's complexities from the end user.

2. ***SQL Conformance***

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

3. JDBC must be implemental on top of common database interfaces

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4. Provide a Java interface that is consistent with the rest of the Java system

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. Keep it simple

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. Use strong, static typing wherever possible

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

7. ***Keep the common cases simple***

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally we decided to proceed the implementation using Java Networking.

And for dynamically updating the cache table we go for MS Access database.

Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple	Architecture-neutral
Object-oriented	Portable
Distributed	High-performance
Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.

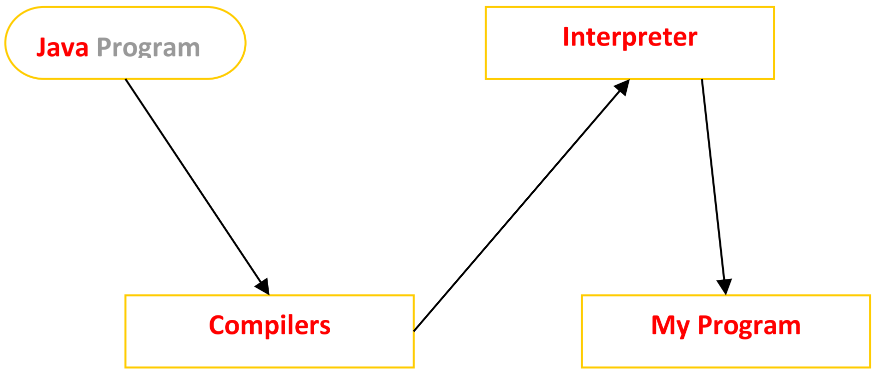


Fig 5.5 : Java Program working Illustration

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

5.6 Networking

TCP/IP stack

The TCP/IP stack is shorter than the OSI one:

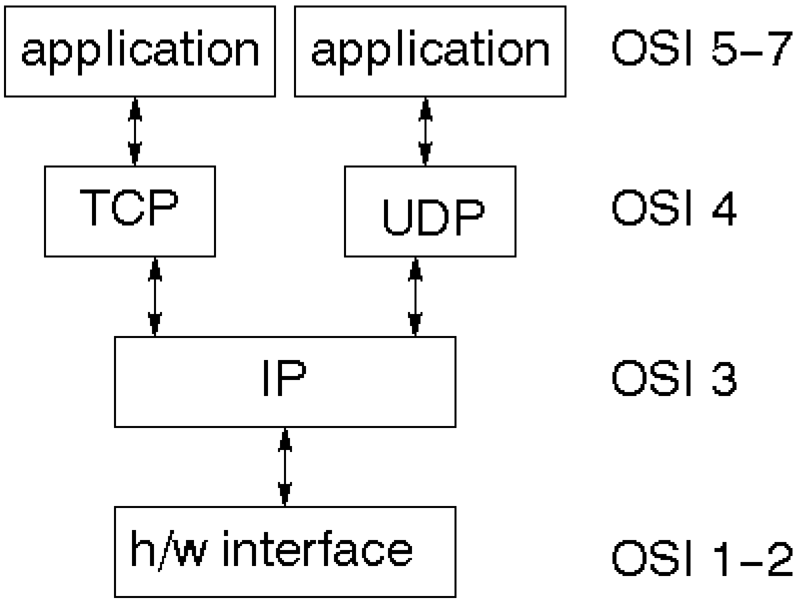


Fig 5.6.1 : TCP/IP Stack in OSI

TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

IP datagram’s

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16bit network addressing. Class C uses 24bit network addressing and class D uses all 32.

Subnet address

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

Total address

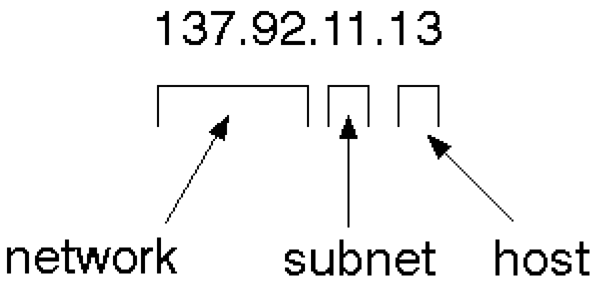


Fig 5.6.2 : 32 bit Address Operator

The 32 bit address is usually written as 4 integers separated by dots.

Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

```
#include <sys/types. h>
#include <sys/socket. h>
int socket (int family, int type, int protocol);
```

Here "family" will be AF_INET for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

5.7 J Free Chart

J Free Chart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. J Free Chart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

J Free Chart is “open source” or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XY Plot class in J Free Chart;

Testing, documenting, testing some more, documenting some more.

2. Time Series Chart Interactivity

Implement a new (to J Free Chart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of J Free Chart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

4. Property Editors

The property editor mechanism in J Free Chart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

J2ME (Java 2 Micro edition):-

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the Java One Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

1. General J2ME architecture

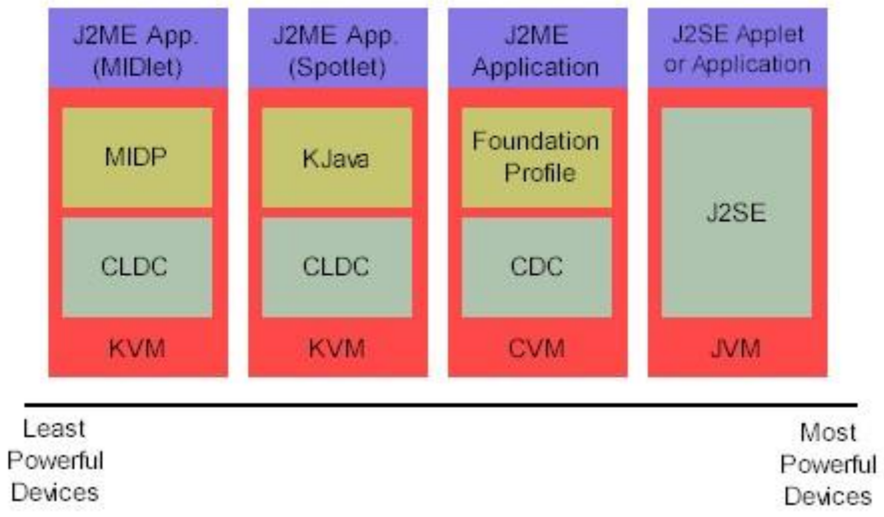


Fig 5.7 : General J2ME Architecture

J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

2. Developing J2ME applications

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role pre verification plays in this process.

3. Design considerations for small devices

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

* Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.

* Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.

* Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

4. Configurations overview

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

* **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

5.8 J2ME profiles

What is a J2ME profile?

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: K Java and MIDP. Both K Java and MIDP are associated with CLDC and smaller devices. Profiles are built on top of

configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

Profile 1: K Java

K Java is Sun's proprietary profile and contains the K Java API. The K Java profile is built on top of the CLDC configuration. The K Java virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. K Java contains a Sun-specific API that runs on the Palm OS. The K Java API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is com. Sun. k java. We'll learn more about the K Java API later in this tutorial when we develop some sample applications.

Profile 2: MIDP

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like K Java, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

* java.lang
* java.io
* java.util
* javax.microedition.io
* javax.microedition.lcdui
* javax.microedition.midlet
* javax. Micro edition. rms

CHAPTER-6

SYSTEM DESIGNS

6.1 Architecture Diagram:

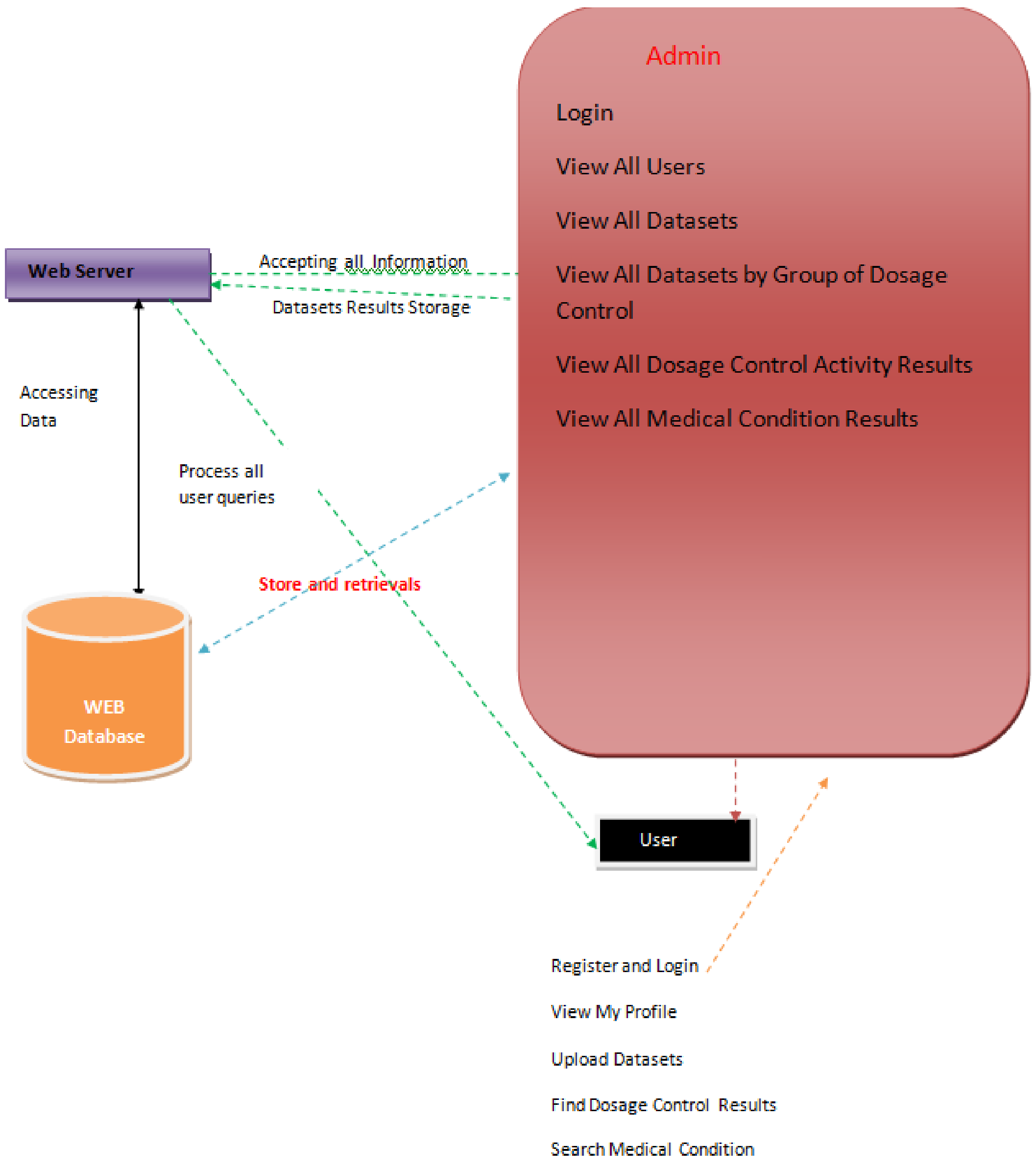
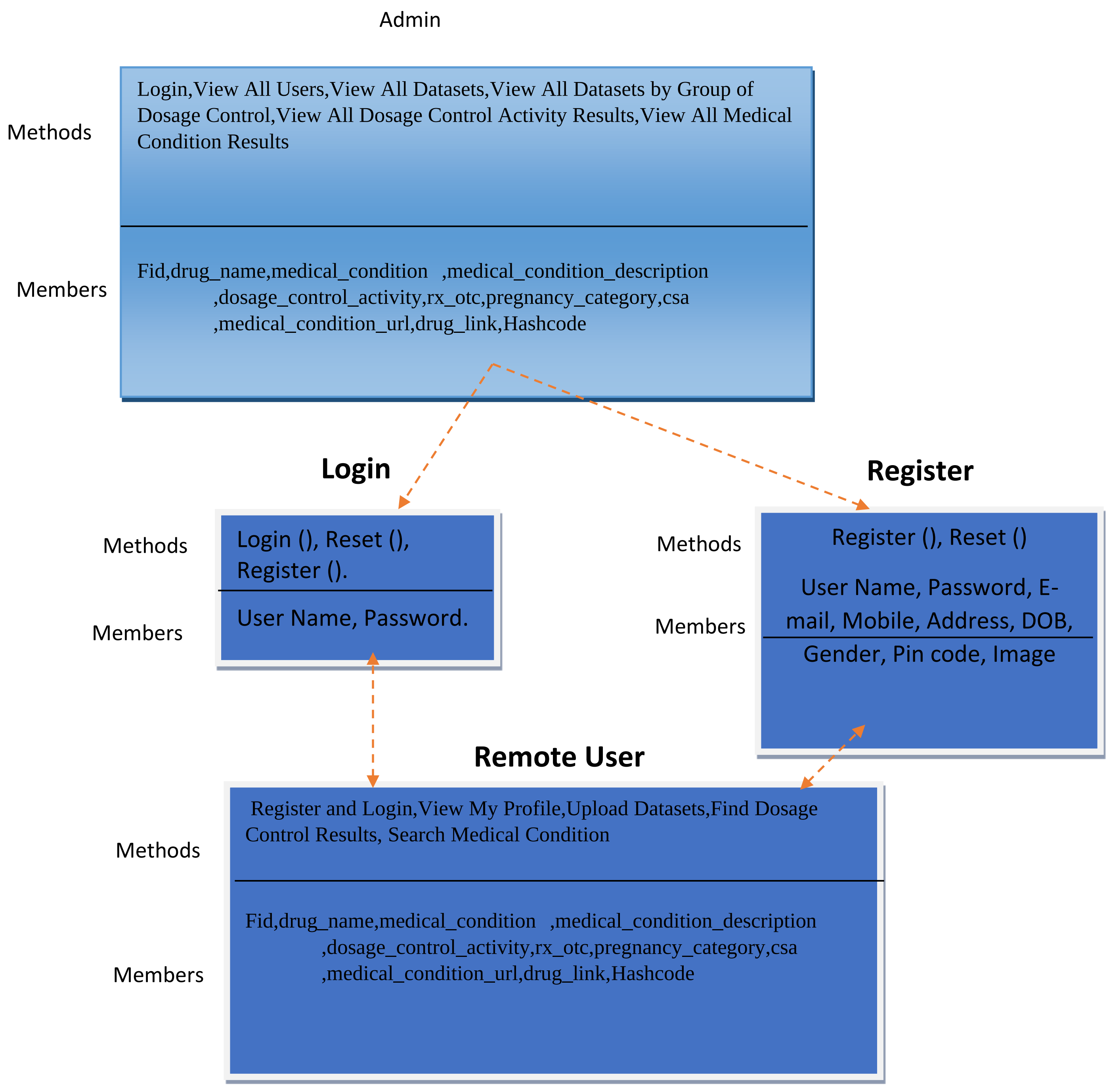


Fig 6.1 : Architecture Diagram

6.2 Class Diagram:



Admin
Methods
Login,View All Users,View All Datasets,View All Datasets by Group of Dosage Control,View All Dosage Control Activity Results,View All Medical Condition Results
Members
Fid,drug_name,medical_condition ,medical_condition_description ,dosage_control_activity,rx_otc,pregnancy_category,csa ,medical_condition_url,drug_link,Hashcode
Login
Methods
Login (), Reset (), Register ().
Members
User Name, Password.
Register
Methods
Register (), Reset ()
Members
User Name, Password, E-mail, Mobile, Address, DOB, Gender, Pin code, Image
Remote User
Methods
Register and Login,View My Profile,Upload Datasets,Find Dosage Control Results, Search Medical Condition
Members
Fid,drug_name,medical_condition ,medical_condition_description ,dosage_control_activity,rx_otc,pregnancy_category,csa ,medical_condition_url,drug_link,Hashcode

Fig 6.2 : Class Diagram

6.3 Data Flow Diagram:

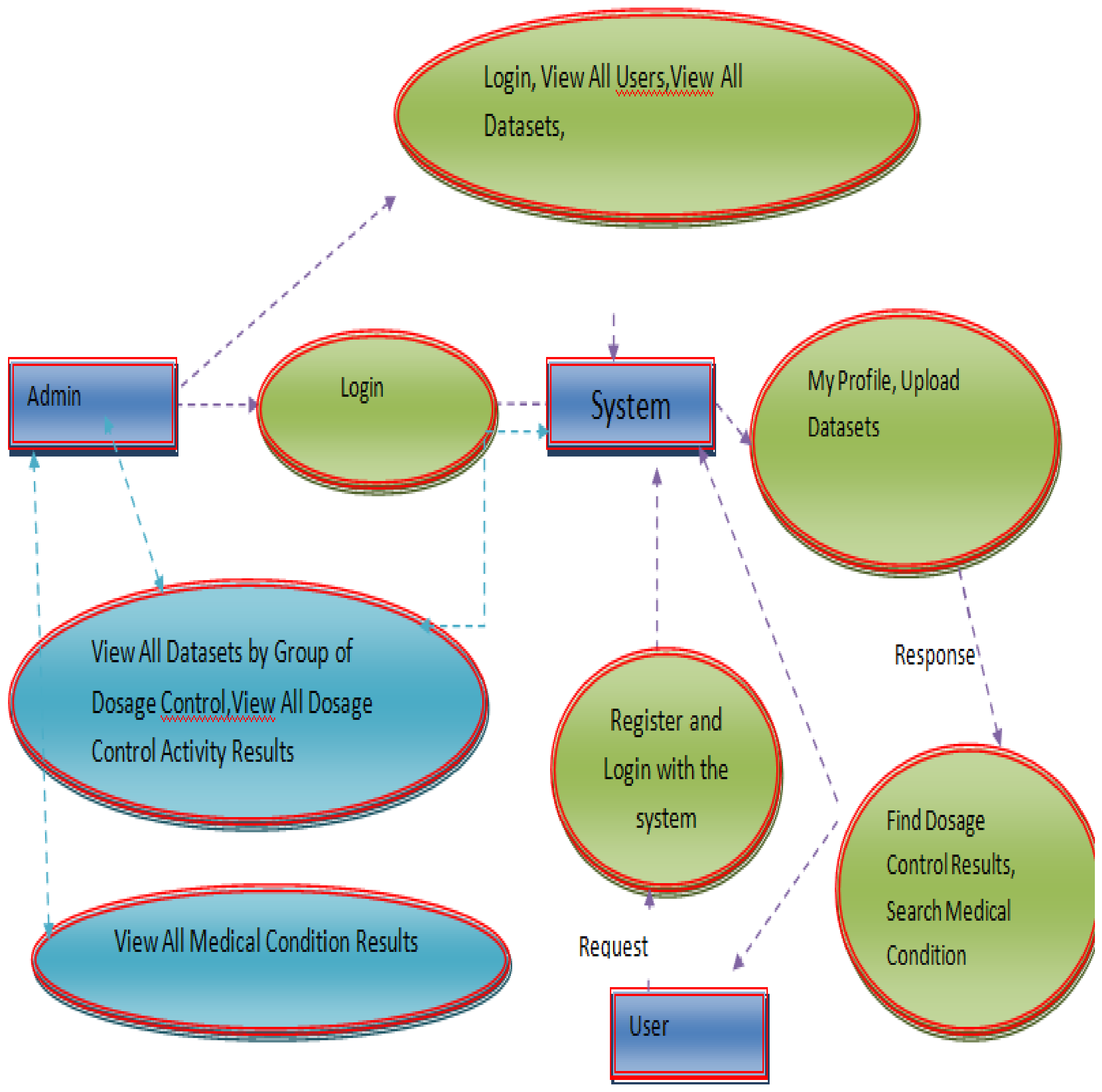
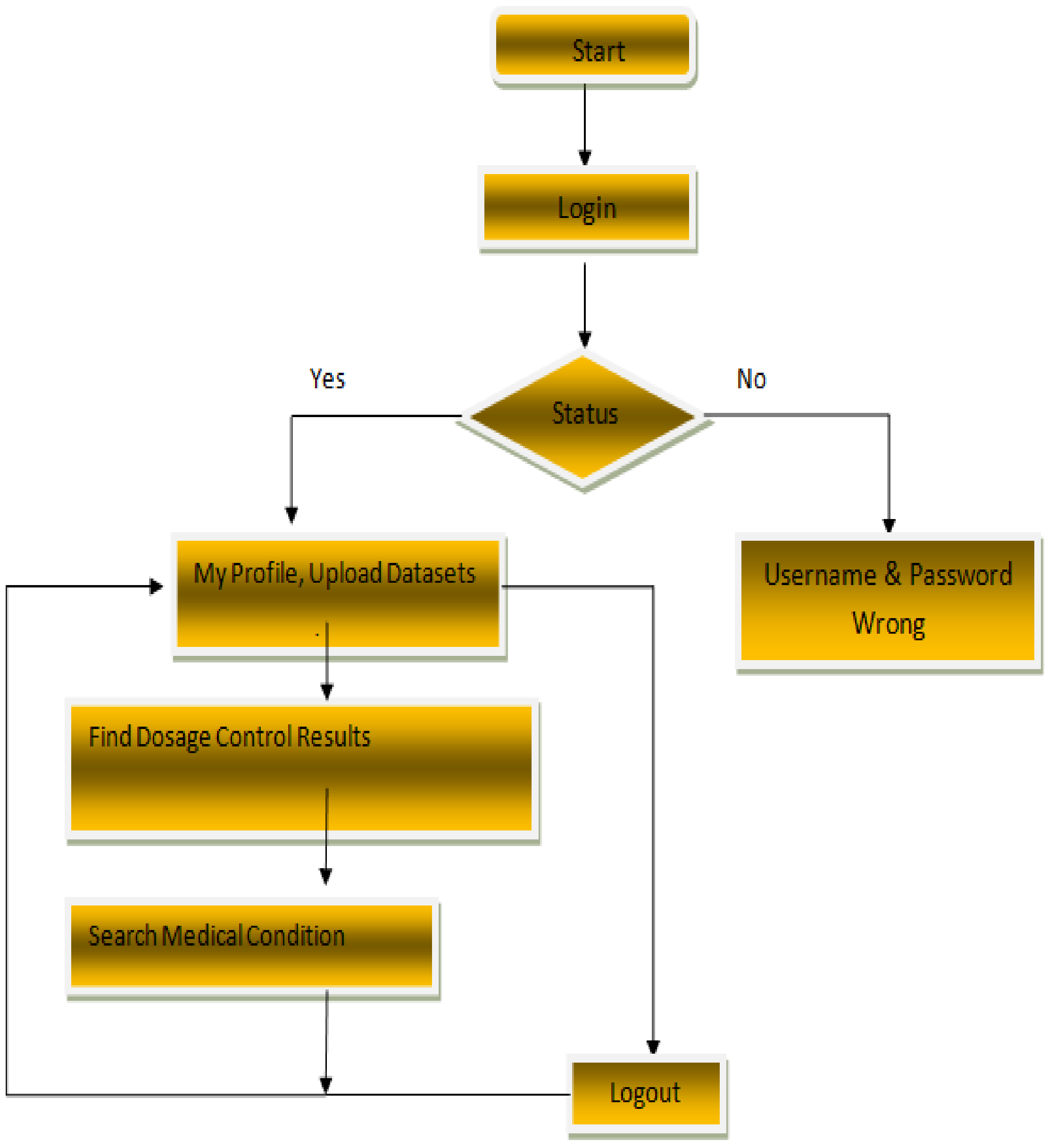

Login, View All Users,View All Datasets,
Admin
Login
System
My Profile, Upload Datasets
View All Datasets by Group of Dosage Control,View All Dosage Control Activity Results
Register and Login with the system
Response
Find Dosage Control Results, Search Medical Condition
View All Medical Condition Results
Request
User

Fig 6.3 : Data Flow Diagram

6.4 Flow Chart:



Start
Login
Yes
Status
No
My Profile, Upload Datasets
Username & Password Wrong
Find Dosage Control Results
Search Medical Condition
Logout

Fig 6.4 : Flow Chart

6.5 Sequence Diagram

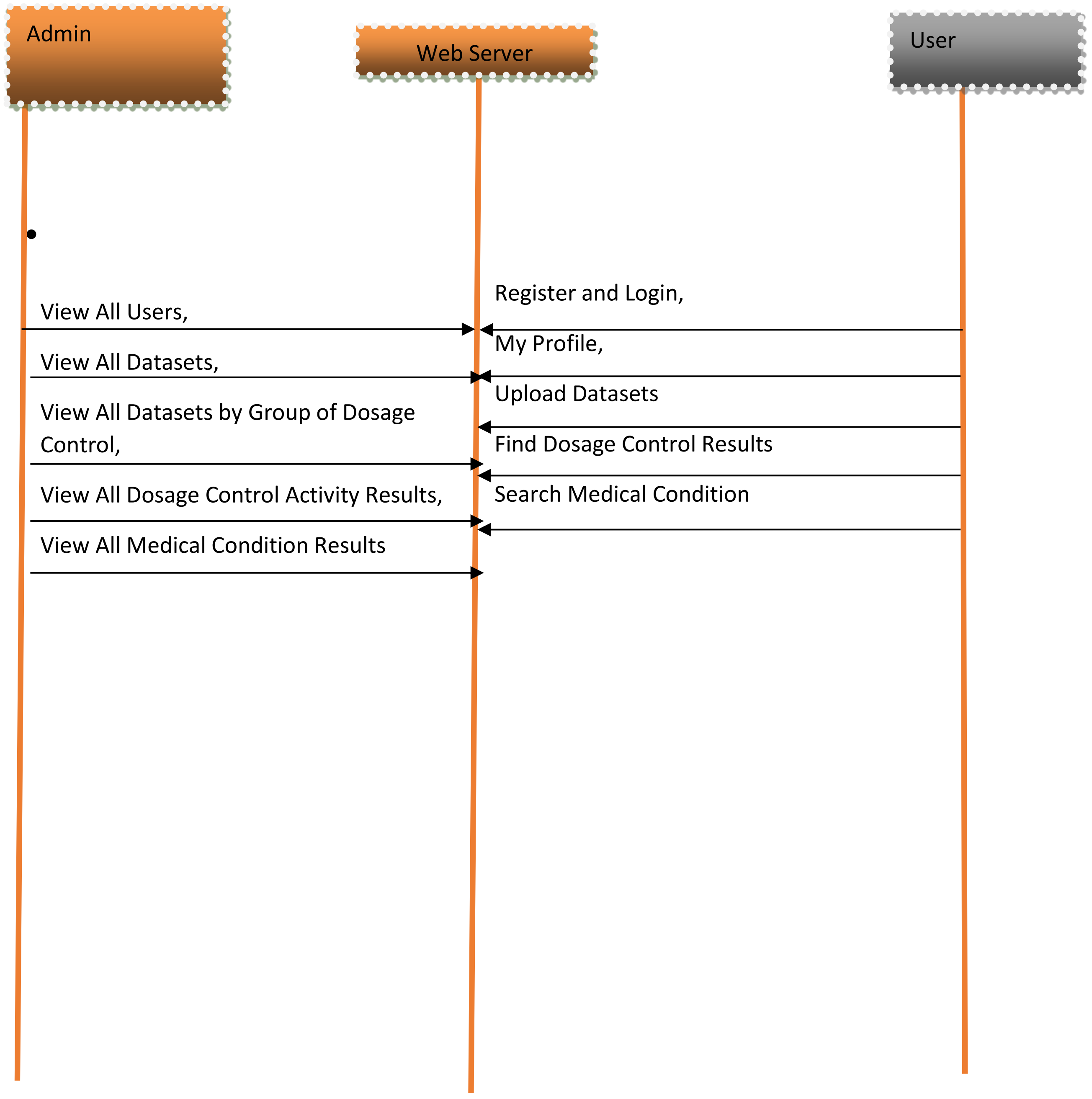
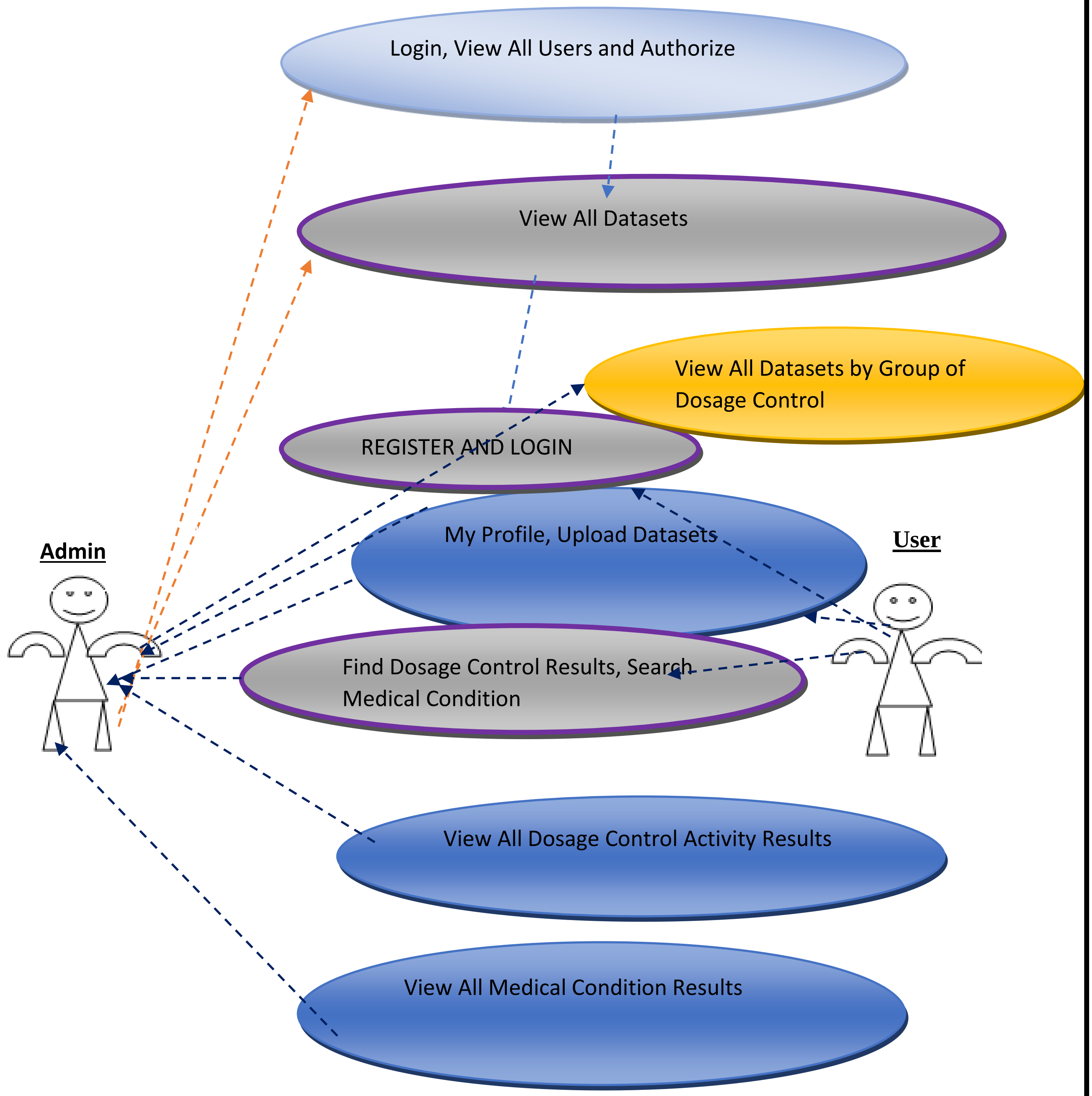

Admin
Web Server
User
View All Users,
View All Datasets,
View All Datasets by Group of Dosage Control,
View All Dosage Control Activity Results,
View All Medical Condition Results
Register and Login,
My Profile,
Upload Datasets
Find Dosage Control Results
Search Medical Condition

Fig 6.5 : Sequence Diagram

6.6 Use Case Diagram



Login, View All Users and Authorize
View All Datasets
View All Datasets by Group of Dosage Control
REGISTER AND LOGIN
My Profile, Upload Datasets
Admin
User
Find Dosage Control Results, Search Medical Condition
View All Dosage Control Activity Results
View All Medical Condition Results

Fig 6.6 : Use Case Diagram

CHAPTER-7

IMPLEMENTATION

7.1 Sample Code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>All End Users </title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style9 {font-size: 24px}
.style12 {
	font-size: 30px;
	color: #CC33FF;
}
.style22 {font-size: 16px; color: #FFFFFF; }
.style5 {	color: #66CCFF;
	font-size: 21px;
	font-weight: bold;
}
.style33 {color: #FF9900}
.style35 {font-size: 16px}
.style37 {font-size: 14px}
.style38 {color: #6699FF}
.style39 {
	color: #9900FF;
	font-weight: bold;
}
.style40 {color: #FF0000}
-->
</style>
</head>
<body>
<div class="main">
  <div class="header">
    <div class="header_resize">
      <div class="logo style9">
        <h1><strong><a  href="index.html"  class="style12">Control  Strategy  of
Immune Systems and Drug Dosage <br />
        Using Reinforcement Learning </a></strong> </h1>
      </div>
      <div class="menu_nav">
        <ul>
          <li><a href="index.html"><span>Home Page</span></a></li>
          <li><a href="UserLogin.jsp">User</a></li>
```

```
<li><class="active"><ahref="AdminLogin.jsp"><span>Admin</span></a></li
>
   </ul>
   </div>
   <div class="clr"></div>
   <div class="slider">
    <div id="coin-slider"> <a href="#"><img src="images/slide1.jpg"
width="935" height="272" alt="" /> </a> <a href="#"><img
src="images/slide2.jpg" width="935" height="272" alt="" /> </a> <a
href="#"><img src="images/slide3.jpg" width="935" height="272" alt="" />
</a> </div>
    <div class="clr"></div>
   </div>
   <div class="clr"></div>
  </div>
 </div>
 <div class="content">
  <div class="content_resize">
   <div class="mainbar">
    <div class="article">
     <h2 class="style40"><strong>View All Users !!! </strong></h2>
      <p class="infopost">&nbsp;</p>
     <div class="clr"></div>
     <div class="img">
      <table width="636" border="1" align="center" cellpadding="0"
cellspacing="0"  ">
       <tr>
```

```
            <td   width="35"  height="34"   valign="middle"  bgcolor="#00FFFF" style="color: #2c83b0;"><div align="center" class="style5 style33 style35">ID</div></td>

            <td   width="127"  height="34"  valign="middle"  bgcolor="#00FFFF" style="color: #2c83b0;"><div align="center" class="style5 style33 style35"><strong>User</strong> Image</div></td>

            <td   width="136"  height="34"  valign="middle"  bgcolor="#00FFFF" style="color: #2c83b0;"><div align="center" class="style5 style33 style35"><strong>User</strong> Name</div></td>

            <td   width="122"  height="34"  valign="middle"  bgcolor="#00FFFF" style="color: #2c83b0;"><div align="center" class="style5 style33 style35"><strong>User</strong> Email</div></td>

            <td   width="114"  height="34"  valign="middle"  bgcolor="#00FFFF" style="color: #2c83b0;"><div align="center" class="style5 style33 style35"><strong>User</strong> Address</div></td>

            <td   width="67"  height="34"   valign="middle"  bgcolor="#00FFFF" style="color: #2c83b0;"><div align="center" class="style5 style33 style35">Authorize Membership </div></td>

           </tr>

           <%@ include file="connect.jsp" %>

           <%


String s1,s2,s3,s4,s5,s6;

int i=0;

try

{

String query="select * from user";

Statement st=connection.createStatement();

ResultSet rs=st.executeQuery(query);

while ( rs.next() )

{
```

```
i=rs.getInt(1);
s1=rs.getString(2);
s2=rs.getString(4);
s3=rs.getString(7);
s4=rs.getString(5);
s5=rs.getString(6);
s6=rs.getString(9);

                                        %>
            <tr>
              <td height="0" align="center"  valign="middle"><p class="style22 style5 style29 style37 style38">&nbsp;</p>
                <div align="center" class="style22 style5 style29 style37 style38">
                  <%out.println(i);%>
                 <p>&nbsp; </p>
                </div></td>
              <td width="127" rowspan="1" align="center" valign="middle" ><div class="style22 style5 style29 style37 style38" style="margin:10px 13px 10px 13px;" > <a class="#" id="img1" href="#" >
                <input name="image" type="image" src="user_Pic.jsp?id=<%=i%>" style="width:90px; height:90px;" />
              </a> </div></td>
              <td height="0" align="center"  valign="middle"><p class="style22 style5 style20 style29 style37 style40">&nbsp;</p>
                <div align="center" class="style22 style5 style20 style29 style37 style40">
                  <%out.println(s1);%>
                  <p>&nbsp; </p>
                </div></td>
                <td height="0" align="center"  valign="middle"><p class="style22
style5 style20 style29 style37 style40">&nbsp;</p>
                    <div align="center" class="style22 style5 style20 style29 style37
style40">
                      <%out.println(s2);%>
                      <p>&nbsp; </p>
                    </div></td>
                <td height="0" align="center"  valign="middle"><p class="style22
style5 style20 style29 style37 style40">&nbsp;</p>
                    <div align="center" class="style22 style5 style20 style29 style37
style40">
                      <%out.println(s5);%>
                      <p>&nbsp; </p>
                    </div></td>
                <%
if(s6.equalsIgnoreCase("waiting"))
{
                                        %>
                <td valign="middle" height="0"
style="color:#000000;"align="center"><div align="center" class="style22 style5
style20 style29 style37 style40">
                    <div align="center" class="style29 style20"><strong><a
href="Admin_Status.jsp?id=<%=i%>">waiting</a></strong></div>
                </div></td>
                <%
}
else
{
                                        %>
```

```
        <td  width="19"  height="0"  align="center"   valign="middle"><div
align="center" class="style22 style5 style20 style29 style37 style40">
          <%out.println(s6);%>
        </div></td>
        <%
}
                                %>
       </tr>
       <%
}

connection.close();
}
catch(Exception e)
{
out.println(e.getMessage());
}
                           %>
       <tr>
        <td  valign="baseline" height="0">&nbsp;</td>
        <td  valign="baseline" height="0">&nbsp;</td>
        <td  valign="baseline" height="0">&nbsp;</td>
        <td  valign="baseline" height="0">&nbsp;</td>
        <td  valign="baseline" height="0">&nbsp;</td>
        <td  valign="baseline" height="0">&nbsp;</td>
       </tr>
      </table>
      <p>&nbsp;</p>
```

```
          <p><a href="AdminMain.jsp" class="style39">Back</a></p>
        </div>
        <div class="clr"></div>
      </div>
    </div>
    <div class="sidebar">
      <div class="searchform">
        <form id="formsearch" name="formsearch" method="post" action="#">
          <span>
          <input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80" value="Search our ste:" type="text" />
          </span>
          <input name="button_search" src="images/search.gif" class="button_search" type="image" />
        </form>
      </div>
      <div class="clr"></div>
      <div class="gadget">
        <h2 class="star"><span>Sidebar</span> Menu</h2>
        <div class="clr"></div>
        <ul class="sb_menu">
          <li><a href="a_AllUsers.jsp">Home</a></li>
          <li><a href="index.html">Log Out</a>  </li>
        </ul>
      </div>
    </div>
    <div class="clr"></div>
  </div>
  </div>
  <div class="fbg"></div>
  <div class="footer">
    <div class="footer_resize">
      <div style="clear:both;"></div>
    </div>
  </div>
</div>
<div align=center></div>
</body>
</html
```

CHAPTER-8

SYSTEM TESTING

TESTING METHODOLOGIES

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

8.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

8.2 Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing

1. Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2. Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

8.3 User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

8.4 Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

8.5 Validation Checking

Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error message. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

8.6 Test Case

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true system test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

User Training

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose, the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

Maintenance

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

Testing Strategy:

A strategy for system testing integrates system test cases and design techniques into a well-planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

8.7 System Testing:

Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

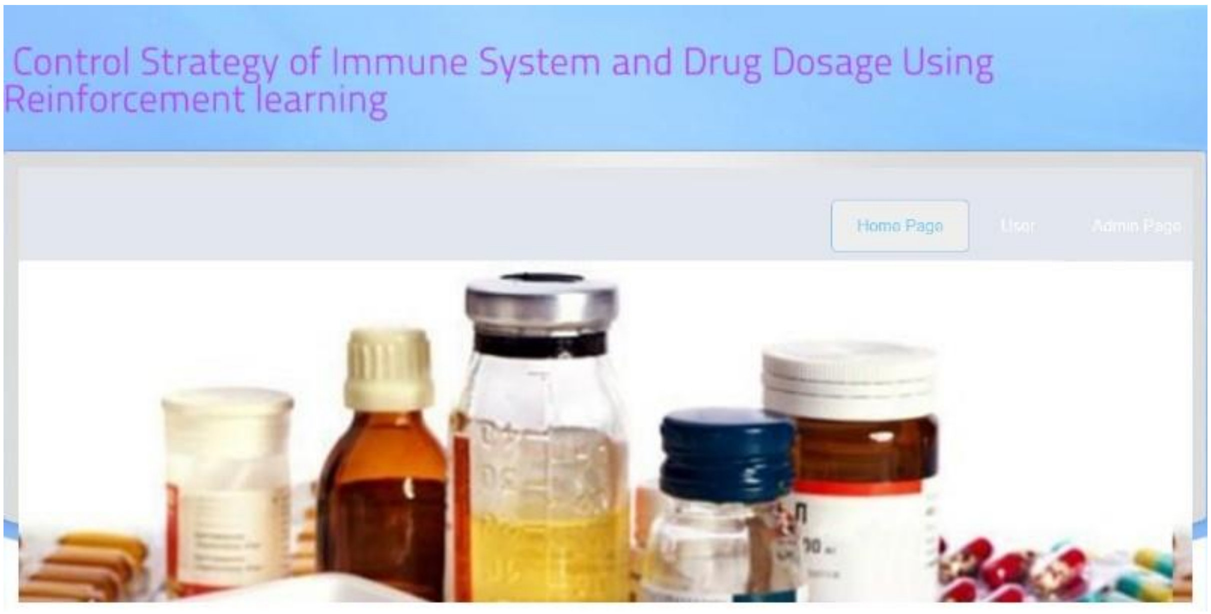
Unit Testing:

In unit testing different are modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each module was found to be working satisfactorily as regards to the expected output from the module.

In Due Course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

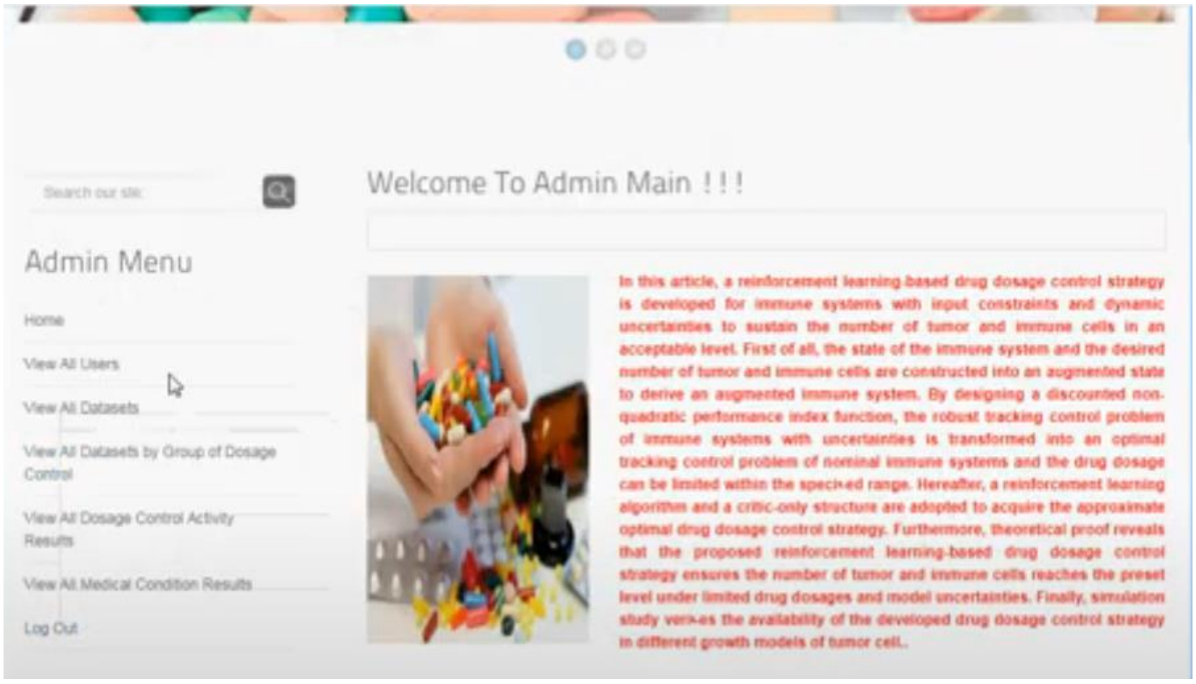
CHAPTER-9

SAMPLE SCREENS



Control Strategy of Immune System and Drug Dosage Using Reinforcement learning
Home Page
User
Admin Page

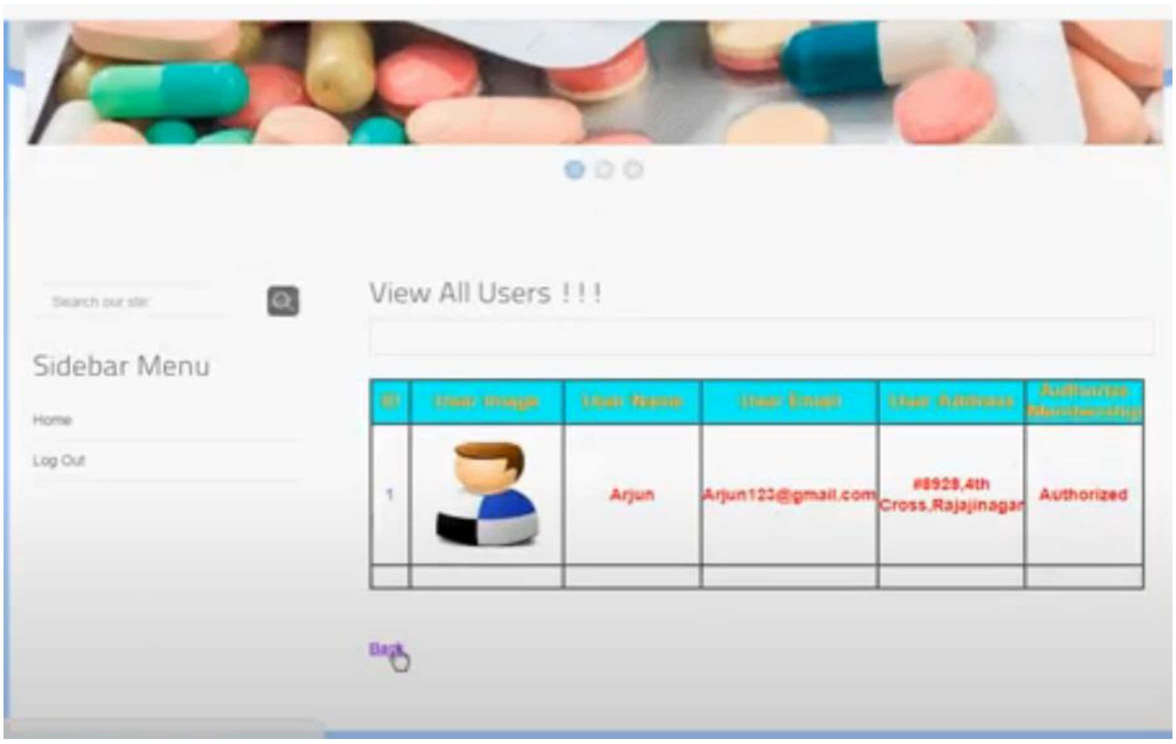
Fig 9.1 : Home In Webpage



Search our site
Admin Menu
Home
View All Users
View All Datasets
View All Datasets by Group of Dosage Control
View All Dosage Control Activity Results
View All Medical Condition Results
Log Out
Welcome To Admin Main ! ! !

In this article, a reinforcement learning-based drug dosage control strategy is developed for immune systems with input constraints and dynamic uncertainties to sustain the number of tumor and immune cells in an acceptable level. First of all, the state of the immune system and the desired number of tumor and immune cells are constructed into an augmented state to derive an augmented immune system. By designing a discounted non-quadratic performance index function, the robust tracking control problem of immune systems with uncertainties is transformed into an optimal tracking control problem of nominal immune systems and the drug dosage can be limited within the speci•ed range. Hereafter, a reinforcement learning algorithm and a critic-only structure are adopted to acquire the approximate optimal drug dosage control strategy. Furthermore, theoretical proof reveals that the proposed reinforcement learning-based drug dosage control strategy ensures the number of tumor and immune cells reaches the preset level under limited drug dosages and model uncertainties. Finally, simulation study veri•es the availability of the developed drug dosage control strategy in different growth models of tumor cell..

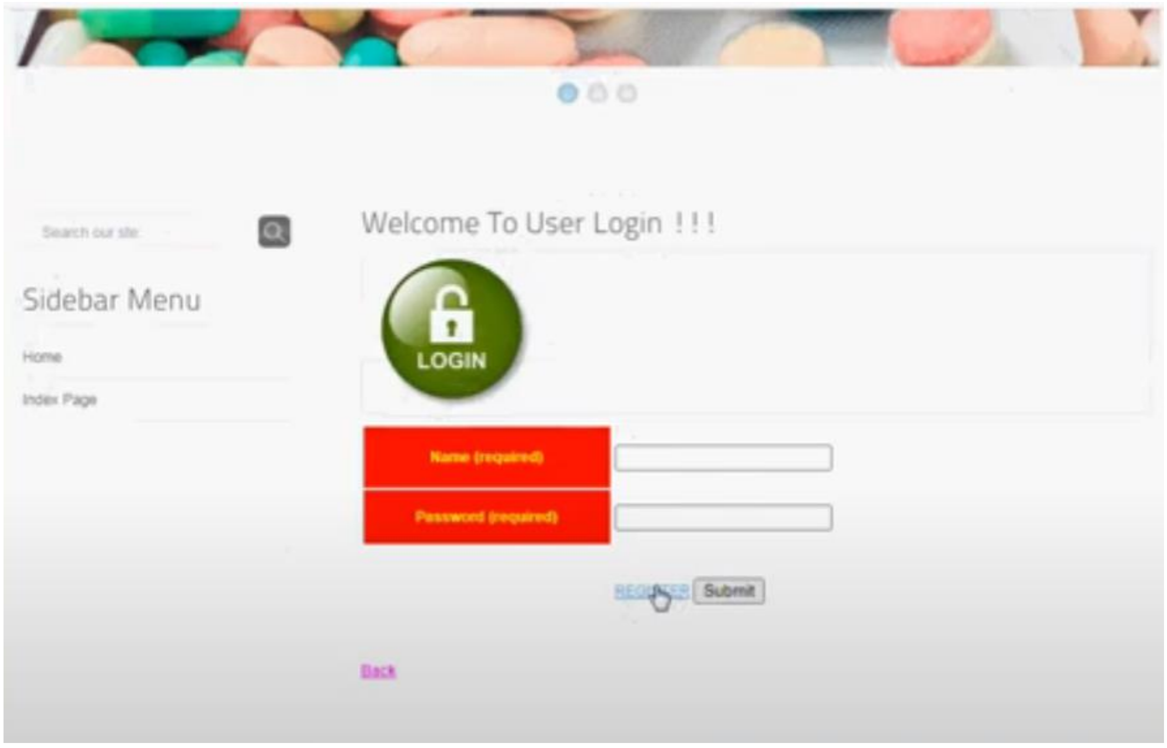
Fig 9.2 : Admin Menu In Webpage



View All Users !!!

ID	User Image	User Name	User Email	User Address	Authorize Membership
1		Arjun	Arjun123@gmail.com	#8928,4th Cross,Rajajinagar	Authorized

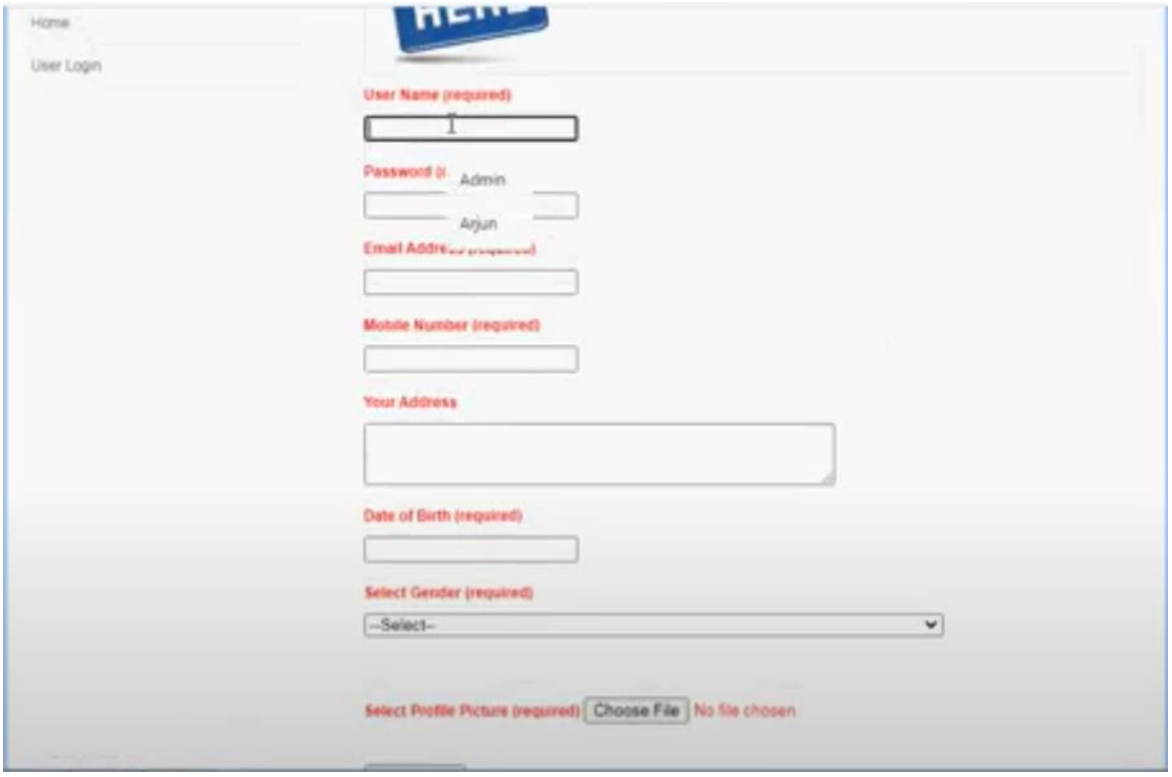
Fig 9.3 : Sidebar Menu In Webpage



Welcome To User Login !!!

Name (required)	
Password (required)	

Fig 9.4 : User Login In Webpage



User Name (required)

Password (required)

Email Address (required)

Mobile Number (required)

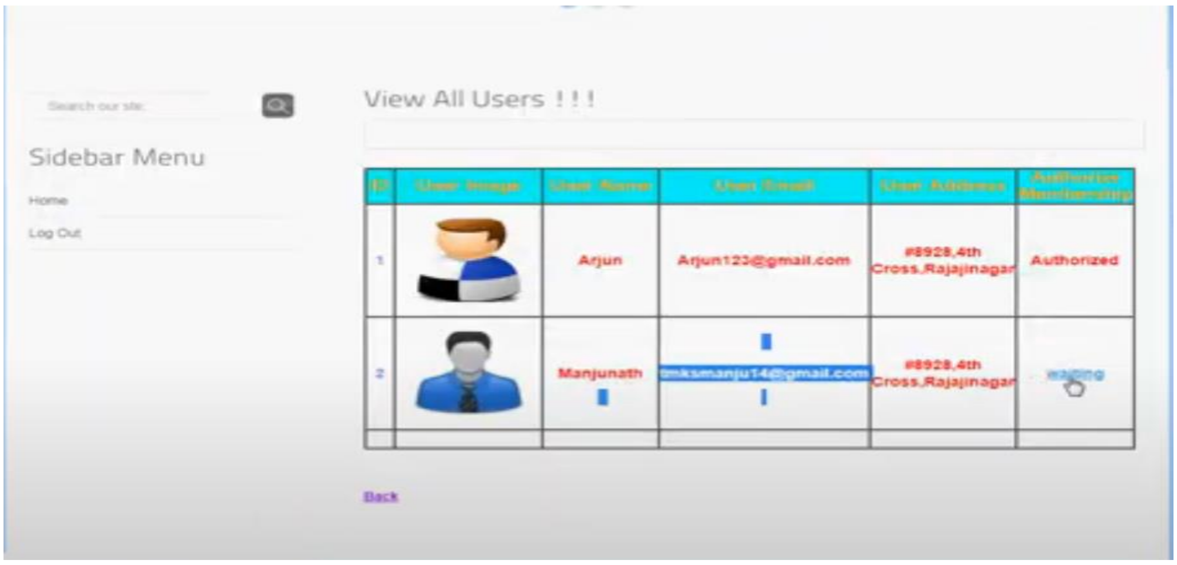
Your Address

Date of Birth (required)

Select Gender (required) --Select--

Select Profile Picture (required) No file chosen

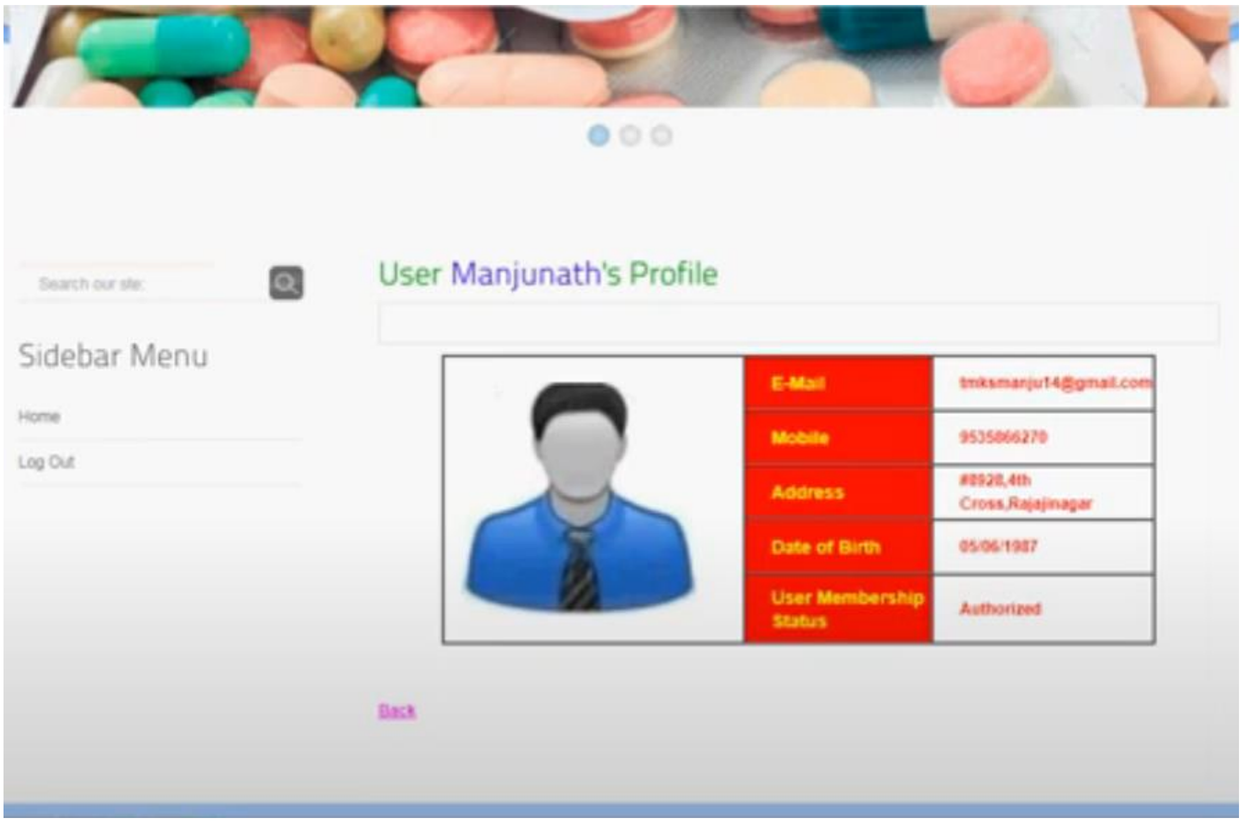
Fig 9.5 : User Registration In Webpage



View All Users !!!

ID	User Image	User Name	User Email	User Address	Authorize Membership
1		Arjun	Arjun123@gmail.com	#8928,4th Cross,Rajajinagar	Authorized
2		Manjunath	mksmanju14@gmail.com	#8928,4th Cross,Rajajinagar	waiting

Fig 9.6 : Updated User Sidebar Menu In Webpage



User Manjunath's Profile

E-Mail	tmksmanju14@gmail.com
Mobile	9535866270
Address	#8928,4th Cross,Rajajinagar
Date of Birth	05/06/1987
User Membership Status	Authorized

Fig 9.7 : Example Users Profile

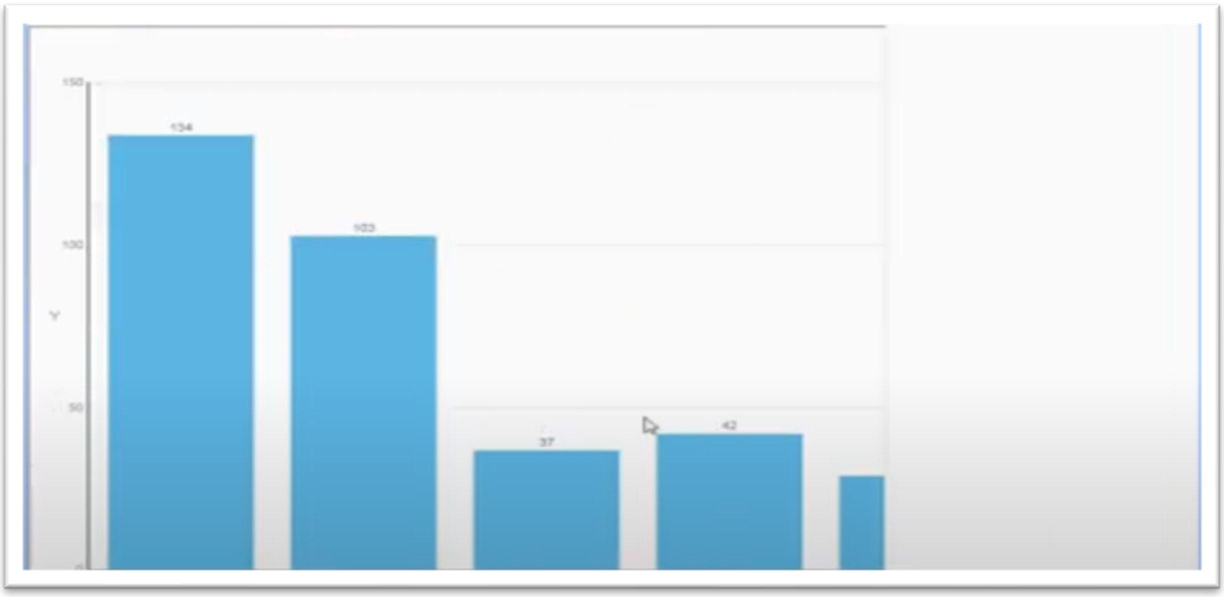
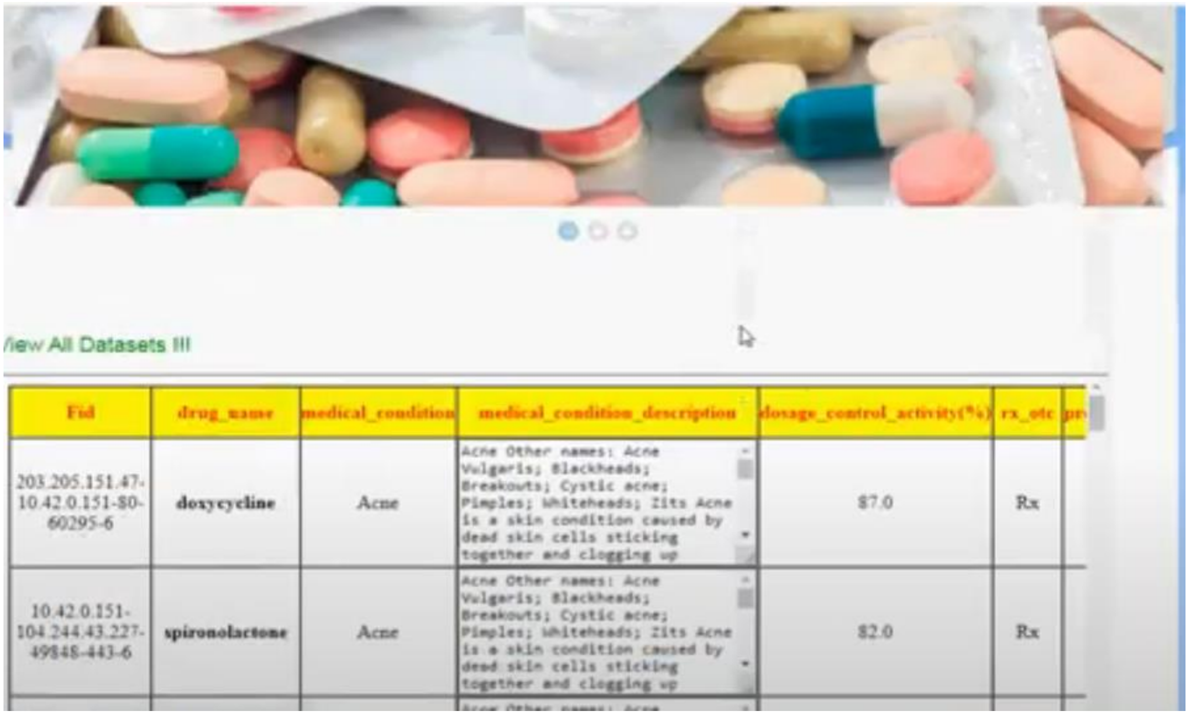


Fig 9.8 : Data Graph



iew All Datasets !!!

Fid	drug_name	medical_condition	medical_condition_description	dosage_control_activity(%)	rx_otc	pr
203.205.151.47-10.42.0.151-80-60295-6	**doxycycline**	Acne	Acne Other names: Acne Vulgaris; Blackheads; Breakouts; Cystic acne; Pimples; Whiteheads; Zits Acne is a skin condition caused by dead skin cells sticking together and clogging up	87.0	Rx	
10.42.0.151-104.244.43.227-49848-443-6	**spironolactone**	Acne	Acne Other names: Acne Vulgaris; Blackheads; Breakouts; Cystic acne; Pimples; Whiteheads; Zits Acne is a skin condition caused by dead skin cells sticking together and clogging up	82.0	Rx	

Fig 9.9 : View All Data in Webpage

CHAPTER-10

CONCLUSION

This paper provides an immunotherapy regimen for cancer via RL technique. We show that it can be obtained by addressing the robust tracking control problem of immune systems subject to input constraints and dynamic uncertainties in control community. To accomplish this goal, an augmented immune system and a discounted non-quadratic performance index function are established such that the robust tracking control problem of uncertain immune systems is converted to an optimal tracking control problem of its nominal plant. Subsequently, we develop constrained drug dosage control strategy by using RL algorithm and critic only structure. According to the Lyapunov theory, we proof that the developed RL-based drug dosage control strategy ensures the number of tumor and immune cells reaches to the preset level with limited drug dosages. At last, simulation results display that the developed immunotherapy regimen is feasible.

CHAPTER-11

REFERENCES

[1] H. Sung, J. Fer lay, R. L. Siegel, M. Lavers Anne, I. Seromata, A. Jemal, and F. Bray, ``Global cancer statistics 2020: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries," *CA, Cancer J. Clinicians*, vol. 71, no. 3, pp. 209_249, May 2021.

[2] D. Hanahan, ``Hallmarks of cancer: New dimensions," *Cancer Discovery*, vol. 12, no. 1, pp. 31_46, Jan. 2022.

[3] D. Hanahan and Weinberg, ``Hallmarks of cancer: The next generation," *Cell*, vol. 144, no. 5, pp. 646_774, Mar. 2011.

[4] F. Greene and L. Sobin, ``The staging of cancer: A retrospective and prospective appraisal," *CA, Cancer J. Clinicians*, vol. 58, no. 3, pp. 180_190, May 2008.

[5] R.Nowarski, N.Gagliani, S. Huber, and R. A. Flavell, ``Innate immune cells in in ammation and cancer," *Cancer Immunol. Res.*, vol. 1, no. 2, pp. 77_84, Aug. 2013.

[6] S. Woo, L. Corrales, and T. Gajewski, ``Innate immune recognition of cancer," *Annu. Rev. Immunol.*, vol. 33, pp. 445_474, Jan. 2015.

[7] M. St. Paul and P. S. Ohashi, ``The roles of CD8C T cell subsets in antitumor immunity," *Trends Cell Biol.*, vol. 30, no. 9, pp. 695_704, Sep. 2020.

[8] T. K. Kim, E. N. Vandsemb, R. S. Herbst, and L. Chen, ``Adaptive immune resistance at the tumour site: Mechanisms and therapeutic opportunities," *Nature Rev. Drug Discovery*, vol. 21, no. 7, pp. 529_540, Jul. 2022.

[9] T. Wang, Y. Shen, S. Luyten, Y. Yang, and X. Jiang, ``Tissue-resident memory CD8C T cells in cancer immunology and immunotherapy," *Phar-macolog. Res.*, vol. 159, Sep. 2020, Art. no. 104876.

[10] D. S. Chen and I. Mellman, ``Elements of cancer immunity and the cancer_immune set point,'' *Nature*, vol. 541, no. 7637, pp. 321_330, Jan. 2017.
[11] M. Nishino, N. H. Ramaiya, H. Hatabu, and F. S. Hodi, ``Monitoring immune-checkpoint blockade: Response evaluation and biomarker development,'' *Nature Rev. Clin. Oncol.*, vol. 14, no. 11, pp. 655_668, Nov. 2017.
[12] J. Cao and Q.Yan, ``Cancer epigenetics, tumor immunity, and immunotherapy,'' *Trends Cancer*, vol. 6, no. 7, pp. 580_592, Jul. 2020.
[13] L. Zitvogel, L. Apetoh, F. Ghiringhelli, and G. Kroemer, ``Immunological aspects of cancer chemotherapy,'' *Nature Rev. Immunol.*, vol. 8, no. 1, pp. 59_73, Jan. 2008.
[14] P. Gotwals, S. Cameron, D. Cipolletta, V. Cremasco, A. Crystal, B. Hewes, B. Müeller, S. Quaratino, C. Sabatos-Peyton, L. Petruzzelli, J. A. Engelman, and G. Dranoff, ``Prospects for combining targeted and conventional cancer therapy with immunotherapy,'' *Nature Rev. Cancer*, vol. 17, no. 5, pp. 286_301, May 2017.
[15] M. Shari_, A. A. Jamshidi, and N. N. Sarvestani, ``An adaptive robust control strategy in a cancer tumor-immune system under uncertainties,'' *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 16, no. 3, pp. 865_873, May 2019.
VOLUME 11, 2023
[16] H. Jiao, Q. Shen, Y. Shi, and P. Shi, ``Adaptive tracking control for uncertain cancer-tumor-immune systems,'' *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 18, no. 6, pp. 2753_2758, Nov. 2021.
[17] M. Itik, M. U. Salamci, and S. P. Banks, ``SDRE optimal control of drug administration in cancer treatment,'' *Turkish J. Electr. Eng. Comput. Sci.*, vol. 18, pp. 715_729, Jan. 2010.
[18] U. Ledzewicz, M. Naghnaeian, and H. Schattler, ``Bifurcation of singular arcs in an optimal control problem for cancer immune system interactions under treatment,'' in *Proc. 49th IEEE Conf. Decis. Control (CDC)*, Dec. 2010, pp. 7039_7044.

[19] D. Liu, S. Xue, B. Zhao, B. Luo, and Q. Wei, ``Adaptive dynamic programming for control: A survey and recent advances," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 51, no. 1, pp. 142_160, Jan. 2021.
[20] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, ``Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 943_949, Jun. 2008.
[21] D. Liu and Q. Wei, ``Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 621_634, Mar. 2014.
[22] M. Ha, D. Wang, and D. Liu, ``A novel value iteration scheme with adjustable convergence rate," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 28, 2022, doi: 10.1109/TNNLS.2022.3143527.
[23] H. Jiang and B. Zhou, ``Bias-policy iteration based adaptive dynamic programming for unknown continuous-time linear systems," *Automatica*, vol. 136, Feb. 2022, Art. no. 110058.
[24] H. Modares and F. L. Lewis, ``Linear quadratic tracking control of partially-unknown continuous-time systems using reinforcement learning," *IEEE Trans. Autom. Control*, vol. 59, no. 11, pp. 3051_3056, Nov. 2014.
[25] C. Chen, H. Modares, K. Xie, F. L. Lewis, Y. Wan, and S. Xie, ``Reinforcement learning-based adaptive optimal exponential tracking control of linear systems with unknown dynamics," *IEEE Trans. Autom. Control*, vol. 64, no. 11, pp. 4423_4438, Nov. 2019.
[26] J. Lu, Q. Wei, Y. Liu, T. Zhou, and F.-Y. Wang, ``Event-triggered optimal parallel tracking control for discrete-time nonlinear systems," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 52, no. 6, pp. 3772_3784, Jun. 2022.
[27] B. Zhao, D. Liu, and Y. Li, ``Observer based adaptive dynamic programming for fault tolerant control of a class of nonlinear systems," *Inf. Sci.*, vol. 384, pp. 21_33, Apr. 2017.

[28] H. G. Zhang, K. Zhang, Y. Cai, and H. Jian, ``Adaptive fuzzy fault-tolerant tracking control for partially unknown systems with actuator faults via integral reinforcement learning method,'' *IEEE Trans. Fuzzy Syst.*, vol. 27, no. 10, pp. 1986_1998, Oct. 2019.

[29] X. Shan, L. Biao, and L. Derong, ``Event-triggered adaptive dynamic programming for zero-sum game of partially unknown continuous-time nonlinear systems,'' *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 50, no. 9, pp. 3189_3199, Sep. 2020.

[30] K. G. Vamvoudakis and F. L. Lewis, ``Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton Jacobi equations,'' *Automatic a*, vol. 47, no. 8, pp. 1556_1569, Aug. 2011.

[31] M. Li, J. Qin, N. M. Freris, and D. W. C. Ho, ``Multiplayer Stackelberg_ nash game for nonlinear system via value iteration-based integral reinforcement learning,'' *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 4, pp. 1429_1440, Apr. 2022.

[32] D. Liu, D. Wang, F.-Y. Wang, H. Li, and X. Yang, ``Neural-network based online HJB solution for optimal robust guaranteed cost control of continuous-time uncertain nonlinear systems,'' *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2834_2847, Dec. 2014.

[33] D. Wang and D. Liu, ``Learning and guaranteed cost control with eventbased adaptive critic implementation,'' *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6004_6014, Dec. 2018.

[34] M. Abu-Khalaf and F. L. Lewis, ``Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach,'' *Automatic a*, vol. 41, no. 5, pp. 779_791, May 2005.