The Online Judge System

ECODENA

SEN Team #3

Software Design Document

Ecodena – The Online Judge System

SEN TEAM #3

February 28, 2012

Authored by: The Entire Team

Table of Contents

1. Introduction

1.1 Purpose:

The purpose of this document is to describe the design descriptions for the project "Ecodena" in detail which is to be designed and implemented by SEN Team #3. This document will be very much helpful in software development as it provides the details of how the software should be implemented. This document strives to provide detailed design descriptions modularly so that the design and development of various phases of the software can go hand in hand.

This document is intended for:

- **Developers**

In order to be sure they are developing the right software that fulfils requirements provided in requirements specification of Ecodena.

- **Documentation Writers**

In order to know the design packages and in what way they have to explain. Be familiar with the steps taken in the design process.

- **Administrators**

In order to know exactly what components have been used in designing the software so that they can response in failure situations.

- **Users**

Interested users (i.e. programmers, administrators) can read this document if they want to know how the software was implemented.

1.2 Scope of the document:

The scope of this document is to depict the general architecture of the *Ecodena.* This document describes the aspects of Ecodena detailed design that are considered to be architecturally significant. This document covers the architectural design, data design, procedural design, design constraints, development schedule. This document also contains elements such as packages, classes, modules and interactions between them that are most fundamental for guiding the design of *Ecodena* and for understanding the software as a whole.

1.3 Overview

The remainder of this document is several sections.

- The next section is **System Overview** where general description of the software system including its functionality and issues related to the overall system and its design is provided.
- The next section is **Design Considerations** which addresses the special issues to be stressed or resolved before attempting to devise a complete design solution.
- The next section is **System Architecture and Design** where information domain of the project is explained, along with the various types of design including architectural design, data design and object oriented design
- The next section is **User Interface Design** where the system is described from user's perspective.
- Then there is **Time Planning**. In this section plan of project is provided for February and March.
- The last section is **Conclusion** where the conclusion of the initial design report is given

1.4 Definitions, Acronyms, and Abbreviations:

- Gedit (GNOME Editor): Gedit is the official text editor of the GNOME desktop environment.
- Git: Git is an extremely fast, efficient, distributed version control system ideal for the collaborative development of software.
- GitHub: GitHub is the best way to collaborate with others. Fork, send pull requests and manages all your **public** and **private** git repositories.
- BitBucket: *Bitbucket* is a free code DVCS hosting site for Git and Mercurial. Manage your development with a hosted wiki, issue tracker and source code.
- Python : *Python* is a general-purpose, high-level programming language whose design philosophy emphasizes code readability. Python claims to combine "remarkable power with very clear syntax".

1.5 References

[1] Roger S. Pressman, “Software Engineering-a practitioner’s approach”, 5th ed., McGraw-Hill.

[2] Prof. Asim Banerjee slides on “Software Engineering”.

[3] IEEE Recommended Practice for Software Design Descriptions – IEEE Computer Society

[4] Wikipedia

[5] [Bruade] The principal source of textbook material is “Software Engineering: An Object-Oriented Perspective” by Eric J. Bruade (Wiley 2001).

[6] Github (http://github.com)

[7] Bitbucket (http://bitbucket.org)

2. System Overview

Ecodena aims to build an environment for the people, specifically for students, to practice and improve their programming skills. It is designed to organize live online programming contests in fair and secure way. Moreover, it provides an environment for training and practicing algorithms, data- structures and programming languages.

To achieve the main goal of the project, two crucial parts should be addressed:

1. *User Friendly Interface that users can use:* The user interface will be essentially the web interface. There will be several user interface modules where each module is designed for a specific user.

2. *The back-end that will validate the submitted solutions for the users or contestants:* All the grading of problems will be done at the back-end of the system on a Linux based grading server.

The main features of this project are:

1. *Practice:* Solving the problems of the site. The coders can send their solutions to problems at any time; each problem has its own discussion forum, statistics, etc.

2. *Variety of Problems:* The problems provided on the site are from different fields (data structures, Greedy Algorithms etc.) and have varying difficulty levels (easy, medium, difficult etc). So each user has enough problems to enhance his/her skills.

3. *Ideal Environment for Education:* Students can use this environment on programming courses related to learning programming, algorithms and data structures at schools, universities, etc.

3. Design Considerations

There are few things that must be considered during design and implementation process. They are described in the following sections.

3.1 Design Assumptions, Dependencies and Constraints:

In this section we address the end user characteristics and design constraints.

3.1.1 End User Characteristics:

In this section user characteristics and concerns are provided.

1. *Users:* Normal users are primary users of the Ecodena. We expect most of the users to be university students and will have medium level knowledge on programming.

2. *Administrators:* Administrators are primarily responsible for maintenance of the system. They contribute minimally to the problem, but spend more time modifying the system configuration and making updates.

3.2.2 Design Constraints:

Many constraints will affect the design decisions of the *Ecodena.* In order to be able to finish a balanced working project in less than a semester time with all requirements met; these constraints have to be considered efficiently.

1. *Time:* The project is planned to finish at the end of the March 2012. In order to meet the deadlines and complete the project on time, team members should obey the time constraints provided at Gantt chart.

2. *Financial Costs:* All implementation software and tools that will be used are open source. Thus, project will obey the licensing of the used tools and software.

3. *Performance:* Performance is a general sense of reliability when the system is under load. It is important that a substantial number of users be able to access the system at the same time. Therefore, several numbers of tests will be applied frequently to detect lags before going through each step in the project.

4. *Versioning:* When implementation process starts, all the states of the this process should be versioned. To do so, the Git version control system will be used.

5. *Implementation Constraints:* While developing the project different programming languages will be used. Since the grading system should be real-time suggested programming language is Python (and optionally Java) and it will run on the Linux based machine, the grading system will be using Linux systems calls to achieve the given responsibilities. On the other hand, while developing user interfaces Python language will be used along with HTML, CSS, JavaScript and Ajax. Another essential practice for implementation part is programming style of the software. The suggested programming style guidelines include commonly accepted naming conventions by Python and Java developers.

3.2 Design Goals and Guidelines:

The following goals are expected to be achieved at the end of the project.

1. *Usability:* Although *Ecodena* System is composed of diverse systems, applications, and processes, the underlying architecture should be transparent to the administrators. The system should be configurable from

a single source at a central administrative position, and should provide a central, easy-to-use interface that will allow administrators to configure the user interface and features in a way that reduces page clutter. A system will be considered to meet this requirement, it has a single administrative interface, and this interface will allow administrators to easily change themes and other setting that affect page layout across the entire judge system.

2. *Reliability:* Back-up system will need to be designed, based on individual components of the system. The system should be backed up with a frequency that ensures the system and all data is protected. Since the updates and changes will be done to the database via web interface, it should be designed to allow the system to be backed up on a nightly basis, with options for weekly, as well as off-site backup when necessary.

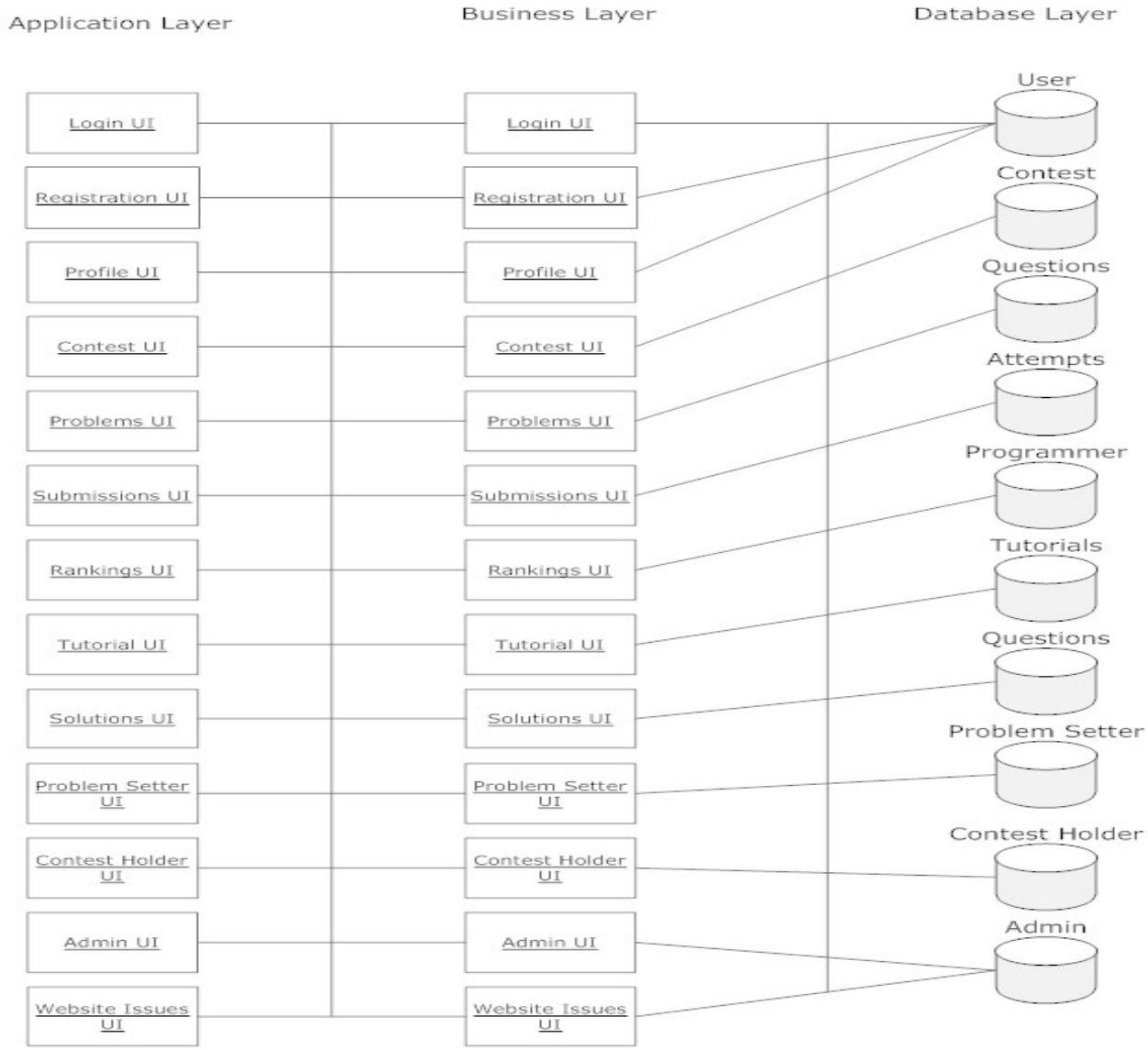
3. *Maintainability:* It is important that the system is designed to be mostly self-sustaining so that the hours spent maintaining the system is reduced. Therefore, the maintenance tasks will be simplified.

4. *Portability:* The project should be designed to be platform independent. Therefore appropriate libraries should be chosen to achieve this goal.

4. System Architecture and Design

4.1 System Architecture

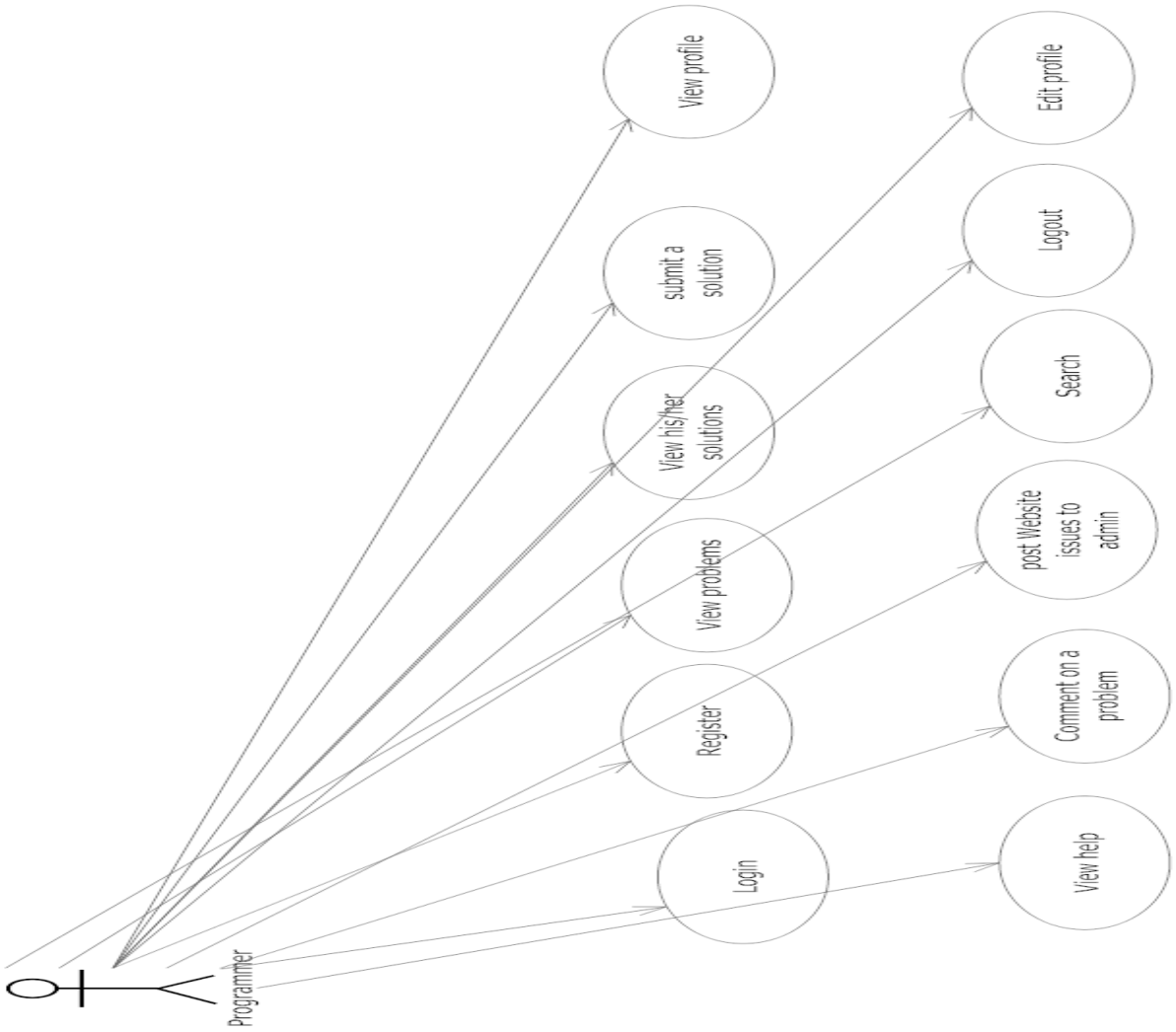
Ecodena system is composed of several modules designed according to user profiles. It consists of a database server which contains data for users and the problems. Each user's data are divided into classes and subclasses so that they are organized in the form that serves right the user. In the diagram below there are main components of the system, subsystem interconnections.



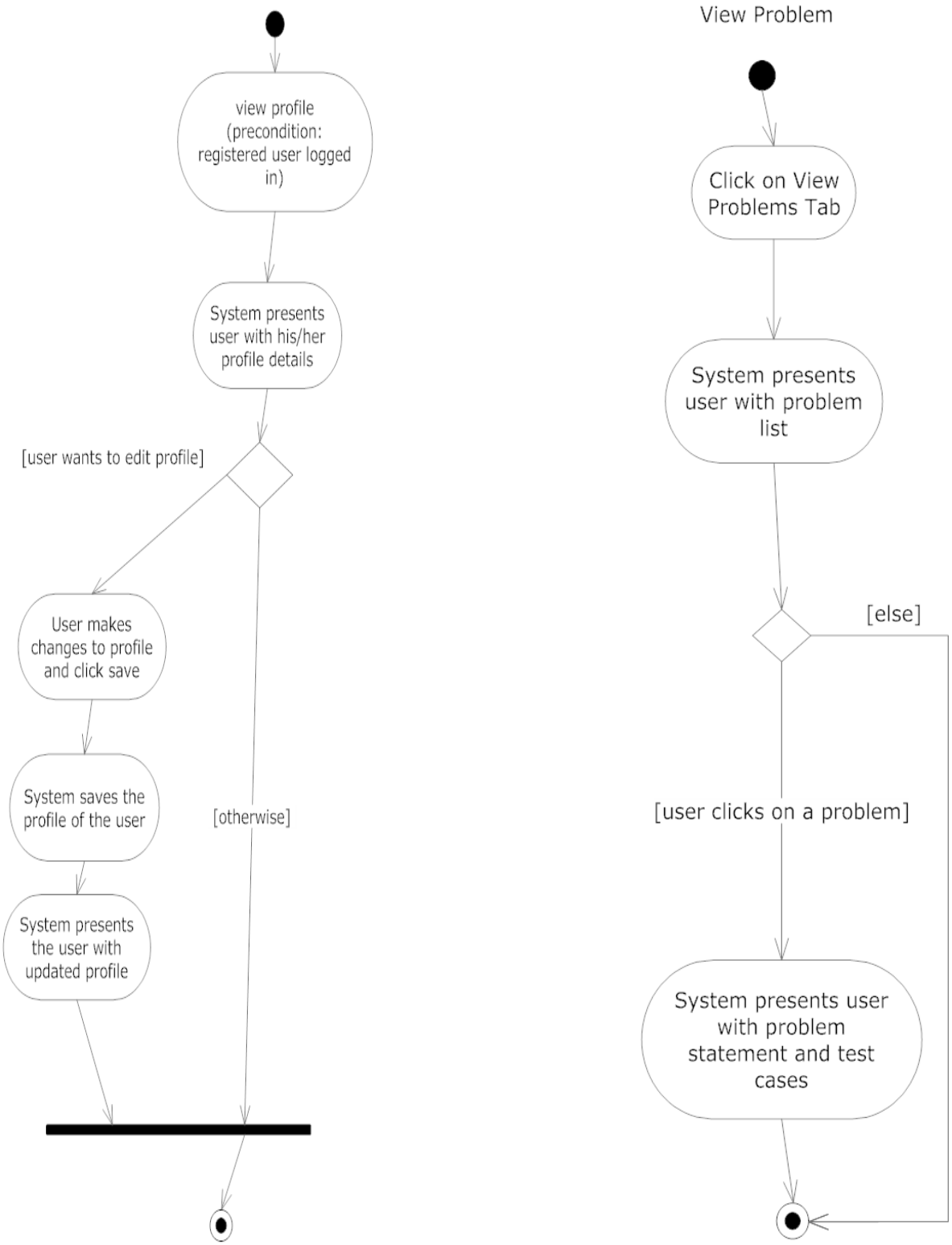
4.2 Architectural design:

Ecodena provides several functions. In this section we realize these functions in terms of activity diagram so that developers fully understand the working process of the system. The most important and most complex features of the project are explained as activity diagrams below. Detailed information and other features will be appended into document at Detailed Design Phase.

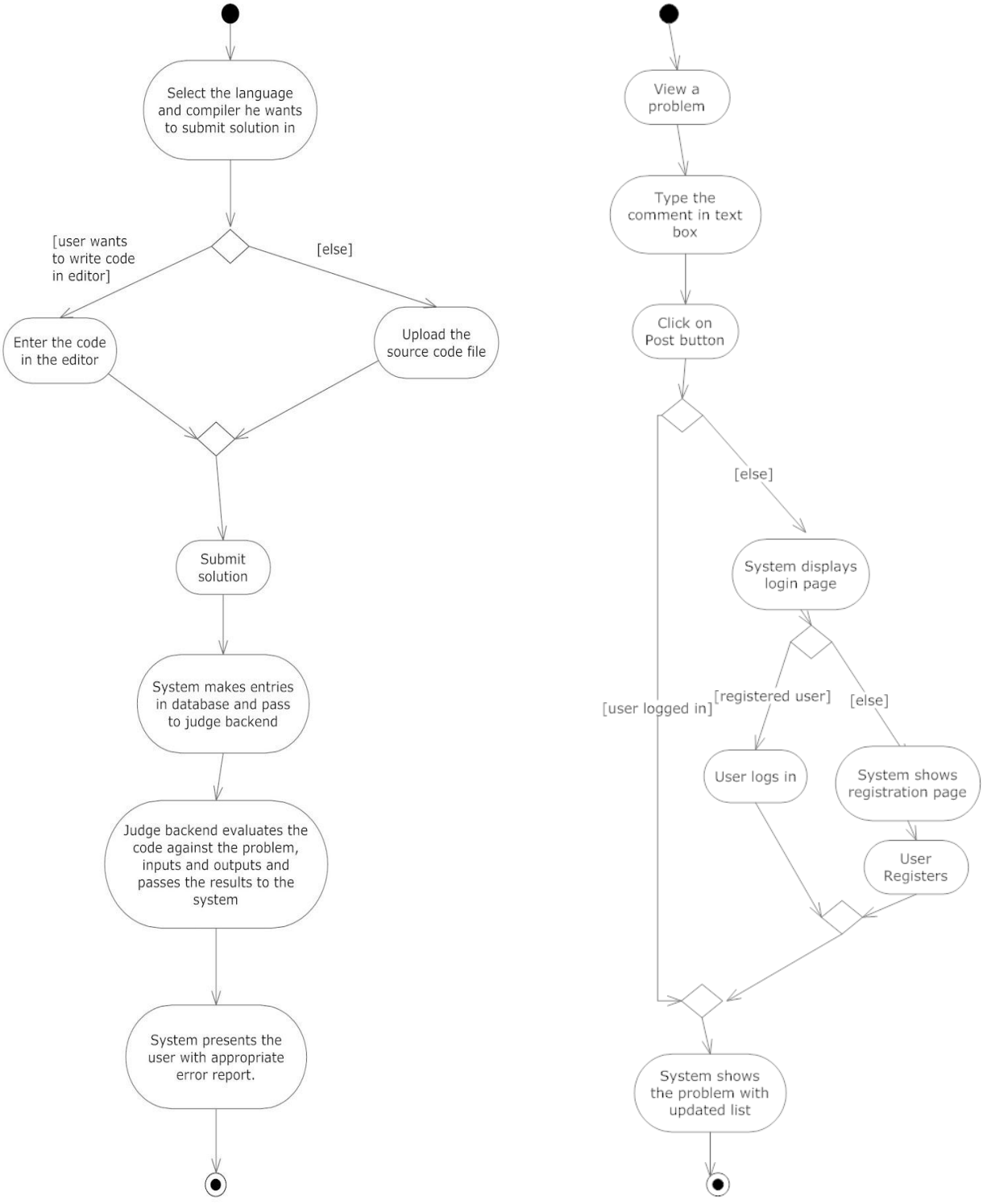
4.2.1 Use Case Diagram:



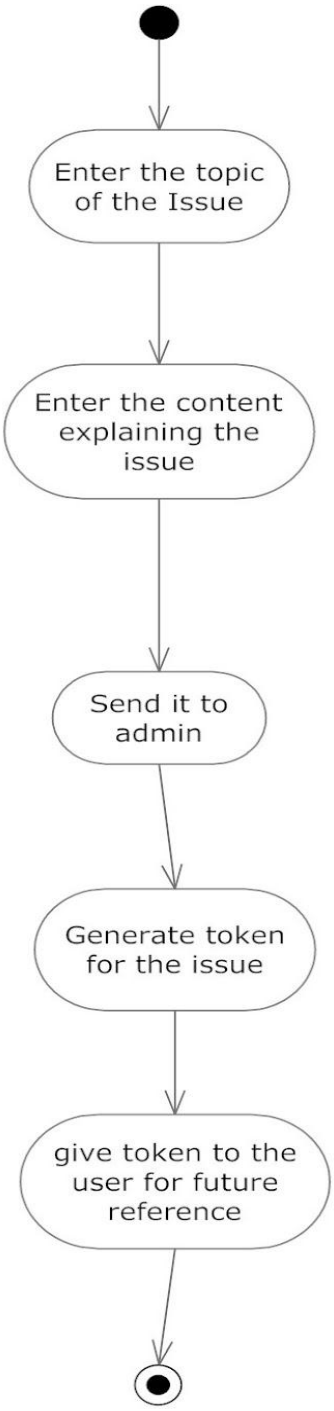
4.2.2 Activity diagrams:

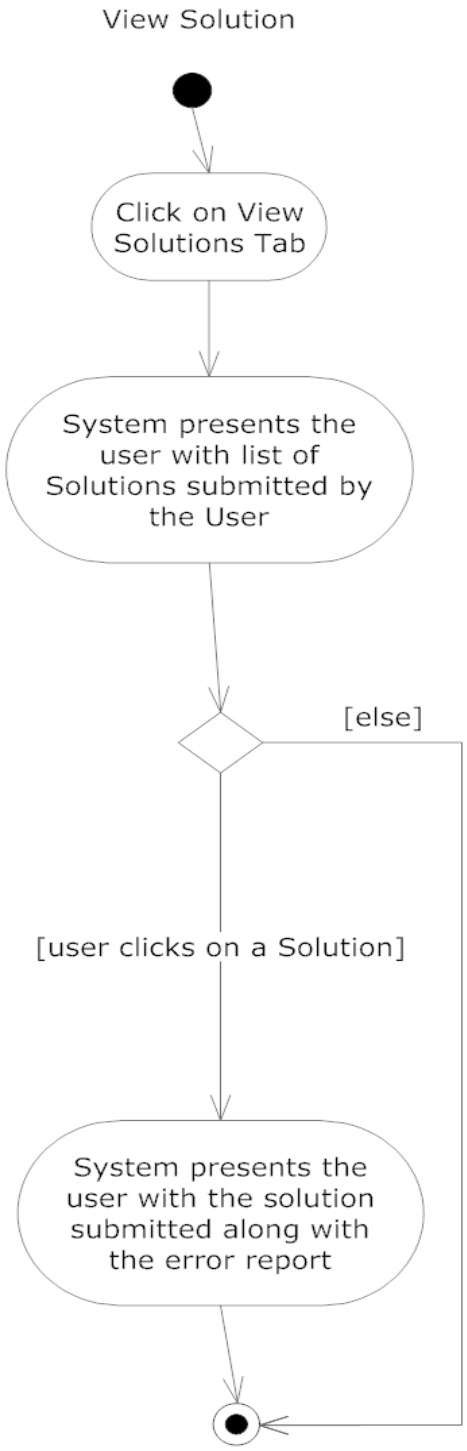


Activity Diagrams for "view/edit profile" (left) and "view problems" (right)



Activity Diagrams for "submit solution" (left) and "post comment" (right)



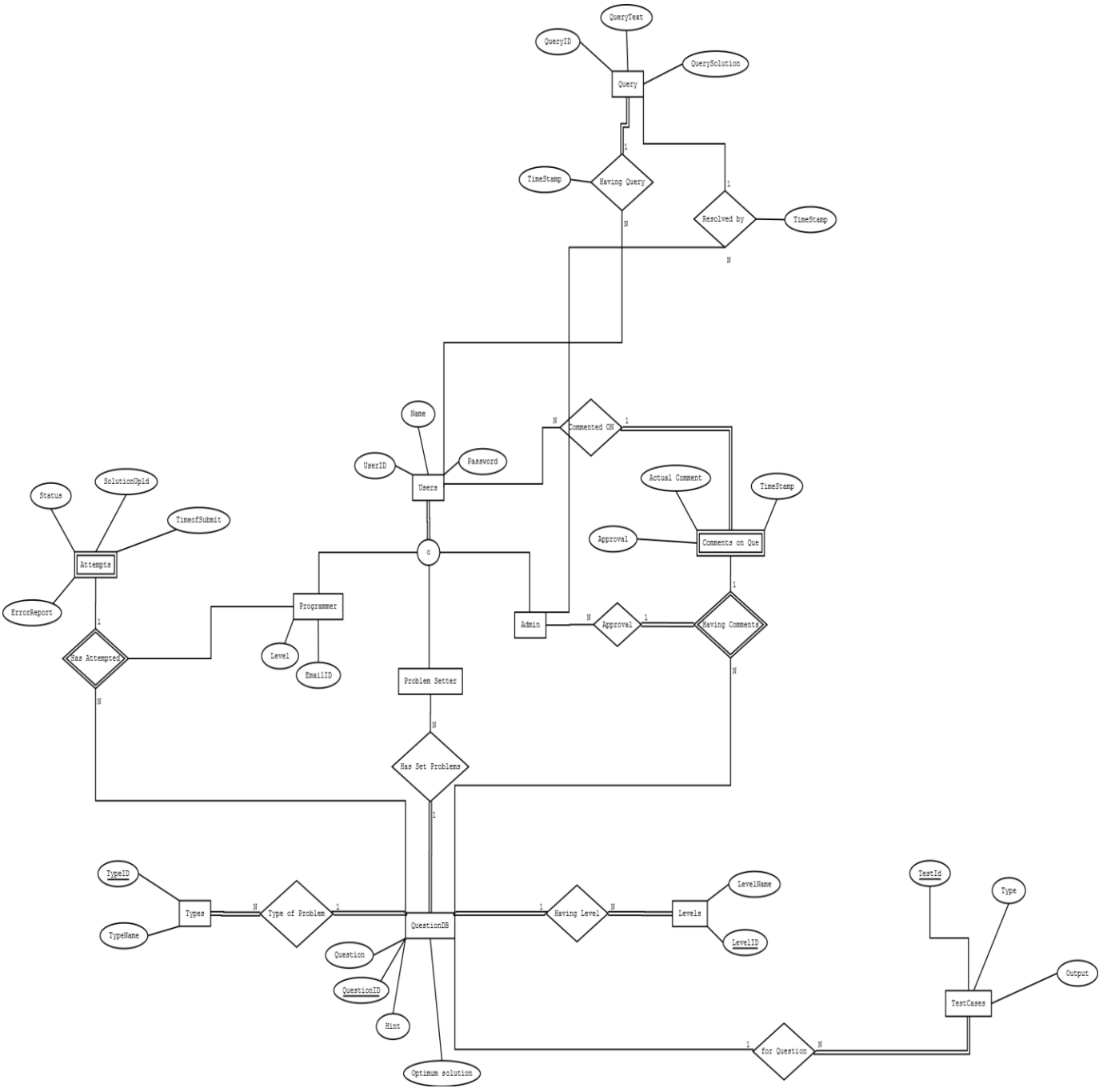


Activity Diagrams for "post website issue" (left) and "view solutions" (right)

4.3 Data Design:

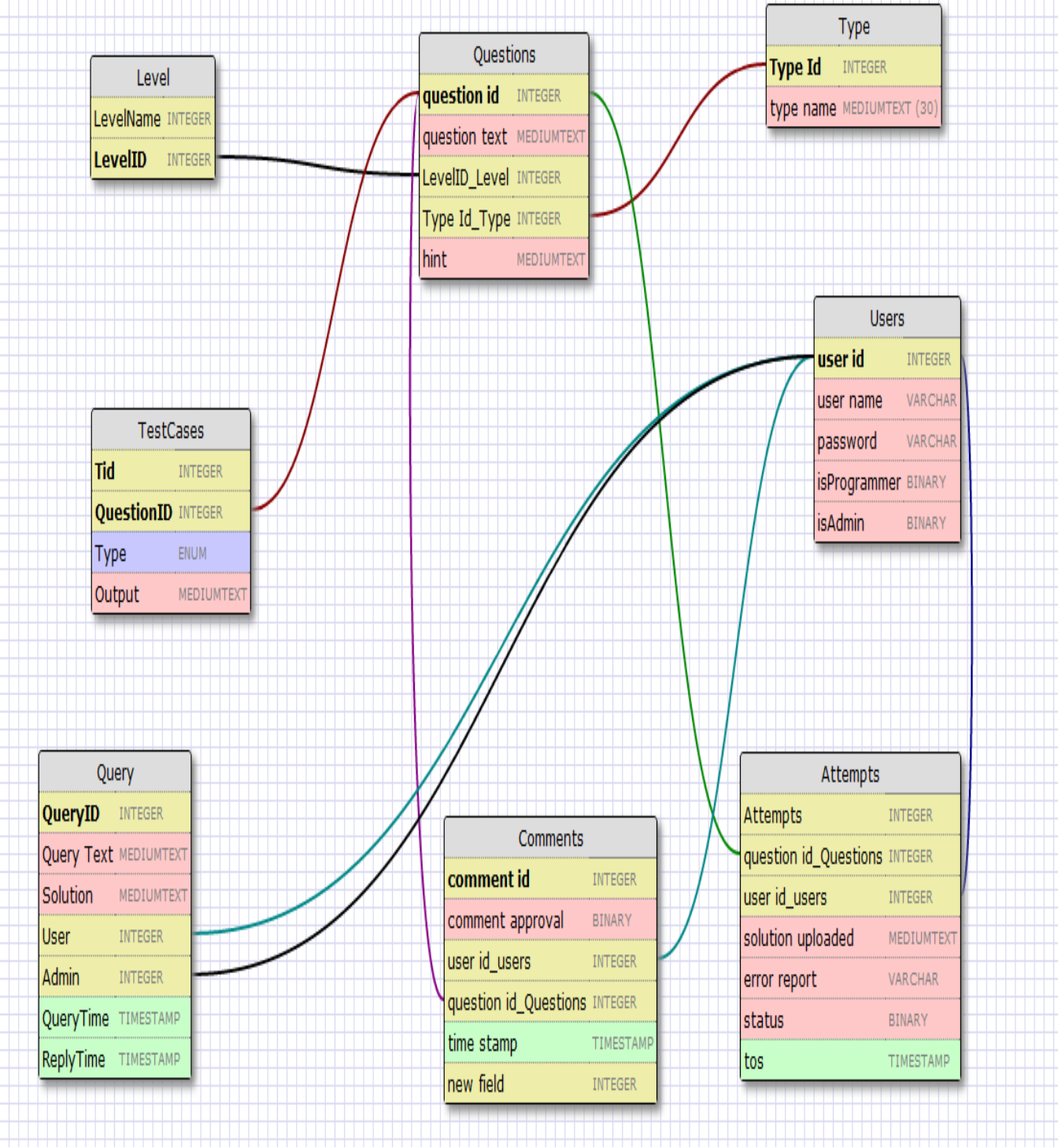
4.3.1 Entity Relationship Diagram

Relations among data objects are described in this section using ER Diagrams



4.3.2 Relational Schema:

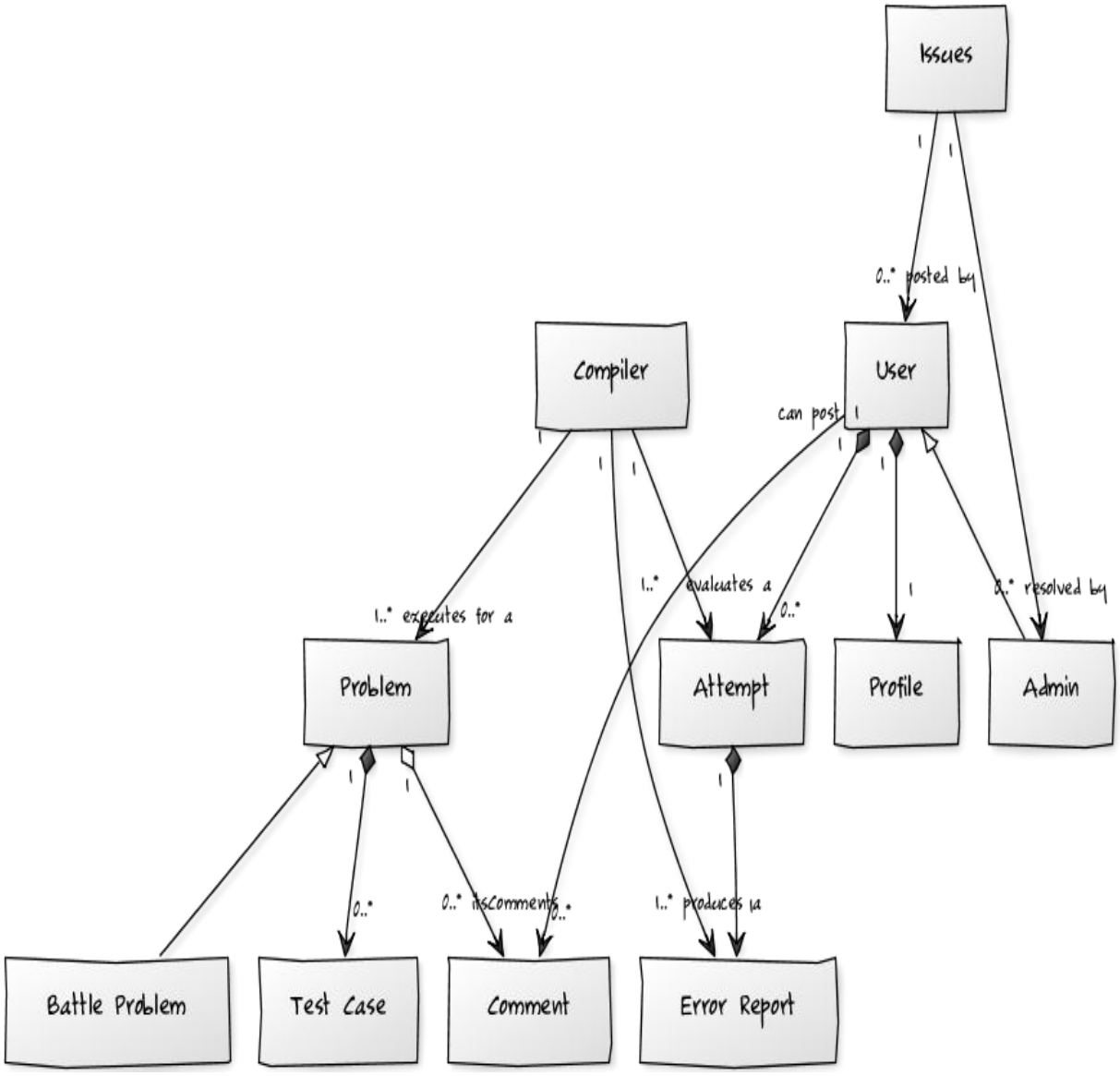
For the efficiency of the project the data query and update should be effective. The schema below shows the data storage of our project's databases.



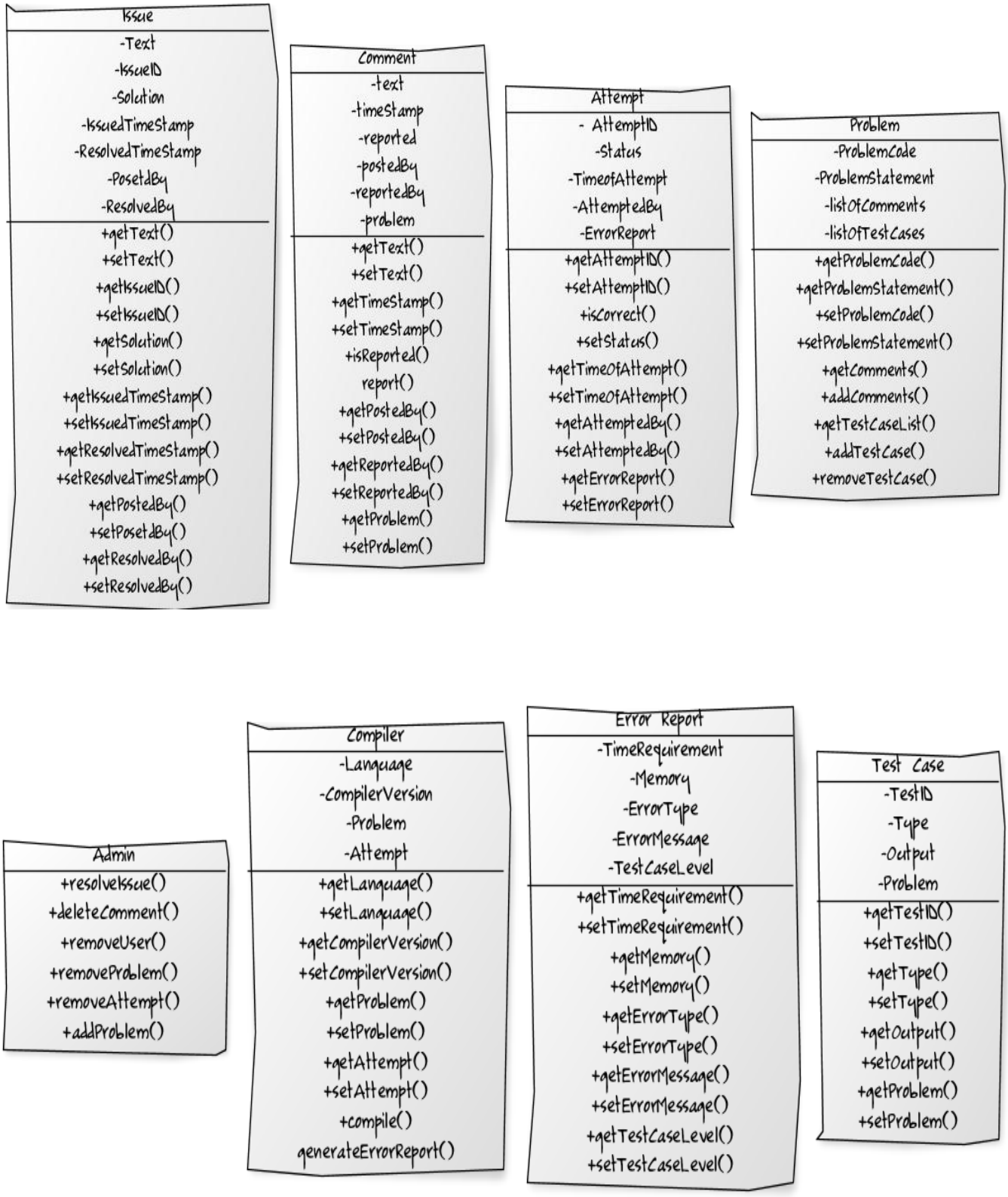
4.4 Object Oriented Design:

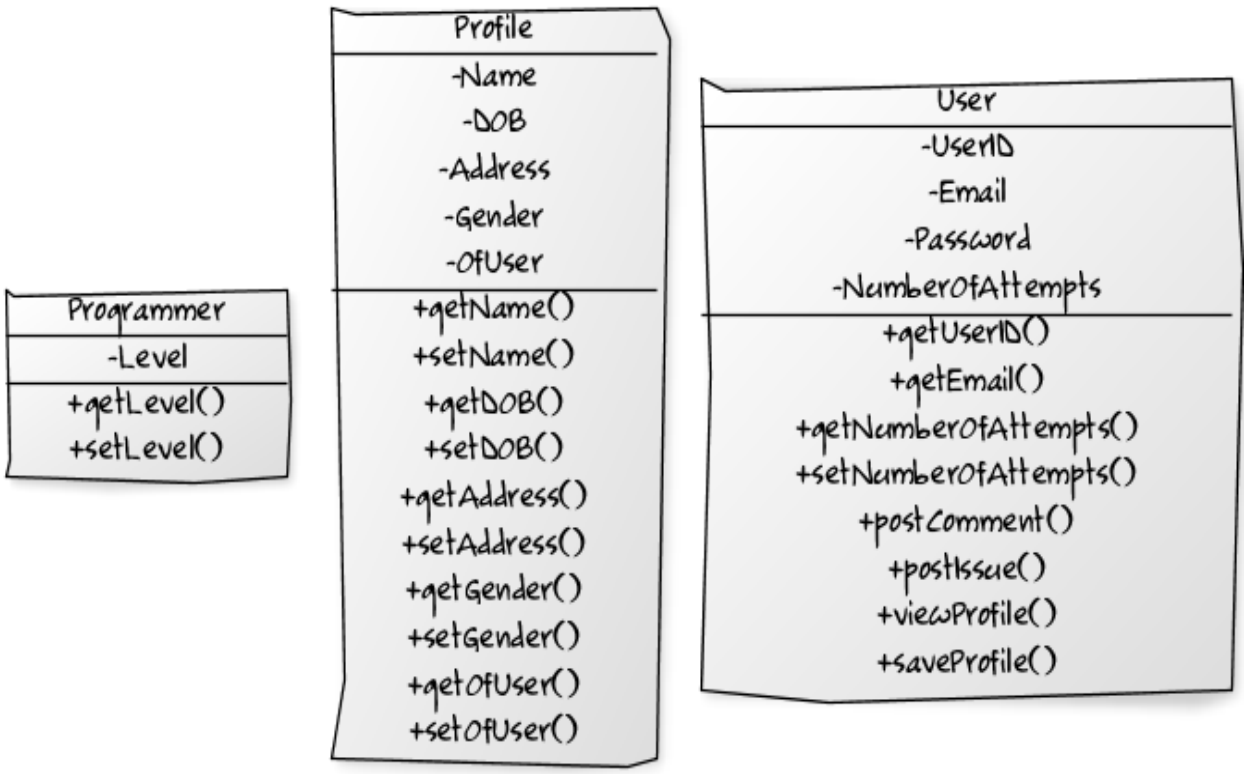
4.4.1 Class diagram:

In this section we describe interface for each package by providing a class diagram corresponding to that package and describe each methods.



(a)


Issue
-Text
-IssueID
-Solution
-IssuedTimeStamp
-ResolvedTimeStamp
-PosetdBy
-ResolvedBy
+getText()
+setText()
+getIssueID()
+setIssueID()
+getSolution()
+setSolution()
+getIssuedTimeStamp()
+setIssuedTimeStamp()
+getResolvedTimeStamp()
+setResolvedTimeStamp()
+getPostedBy()
+setPosetdBy()
+getResolvedBy()
+setResolvedBy()
Comment
-text
-timeStamp
-reported
-postedBy
-reportedBy
-problem
+getText()
+setText()
+getTimeStamp()
+setTimeStamp()
+isReported()
report()
+getPostedBy()
+setPostedBy()
+getReportedBy()
+setReportedBy()
+getProblem()
+setProblem()
Attempt
- AttemptID
-Status
-TimeofAttempt
-AttemptedBy
-ErrorReport
+getAttemptID()
+setAttemptID()
+isCorrect()
+setStatus()
+getTimeOfAttempt()
+setTimeOfAttempt()
+getAttemptedBy()
+setAttemptedBy()
+getErrorReport()
+setErrorReport()
Problem
-ProblemCode
-ProblemStatement
-listOfComments
-listOfTestCases
+getProblemCode()
+getProblemStatement()
+setProblemCode()
+setProblemStatement()
+getComments()
+addComments()
+getTestCaseList()
+addTestCase()
+removeTestCase()
Admin
+resolveIssue()
+deleteComment()
+removeUser()
+removeProblem()
+removeAttempt()
+addProblem()
Compiler
-Language
-CompilerVersion
-Problem
-Attempt
+getLanguage()
+setLanguage()
+getCompilerVersion()
+setCompilerVersion()
+getProblem()
+setProblem()
+getAttempt()
+setAttempt()
+compile()
generateErrorReport()
Error Report
-TimeRequirement
-Memory
-ErrorType
-ErrorMessage
-TestCaseLevel
+getTimeRequirement()
+setTimeRequirement()
+getMemory()
+setMemory()
+getErrorType()
+setErrorType()
+getErrorMessage()
+setErrorMessage()
+getTestCaseLevel()
+setTestCaseLevel()
Test Case
-TestID
-Type
-Output
-Problem
+getTestID()
+setTestID()
+getType()
+setType()
+getOutput()
+setOutput()
+getProblem()
+setProblem()



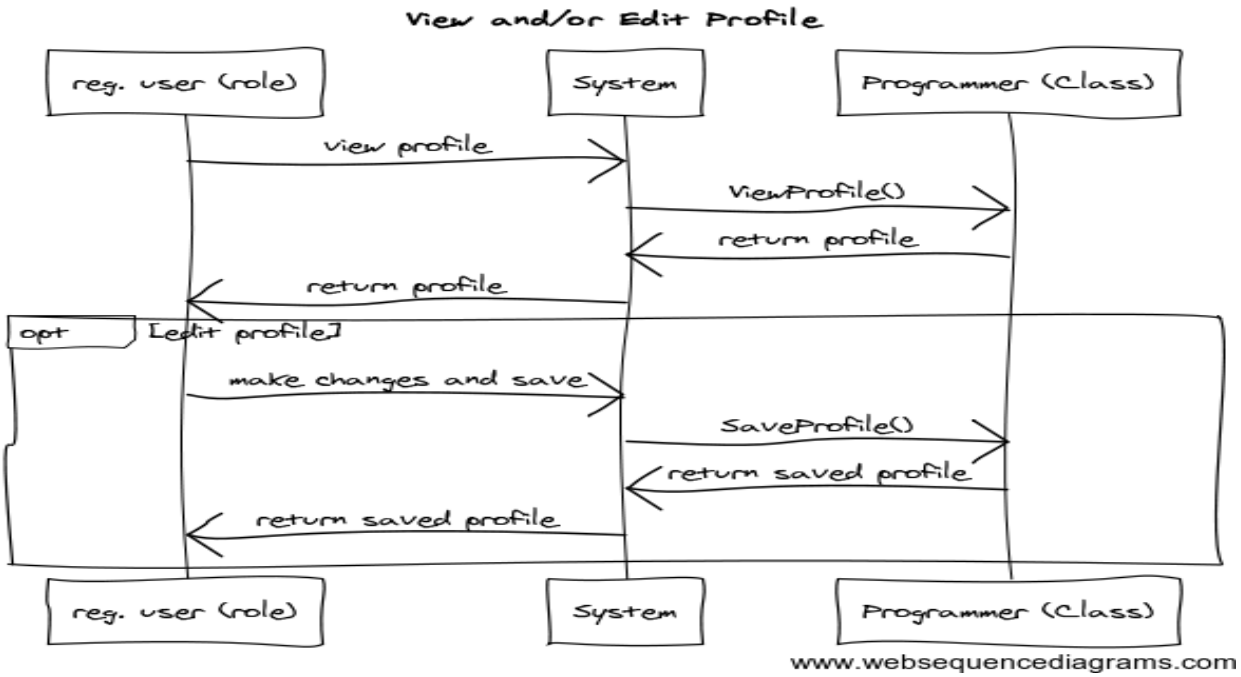
(b)

Diagram

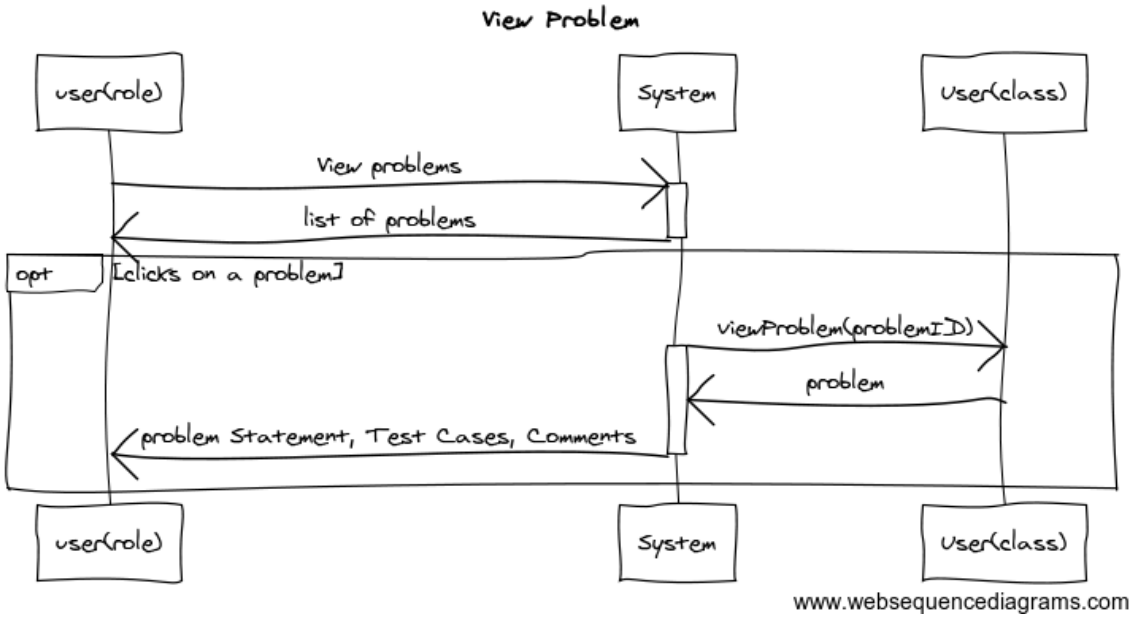
(a) Class Diagram showing interconnection between various classes

(b) Class Diagram showing public and private members of all classes

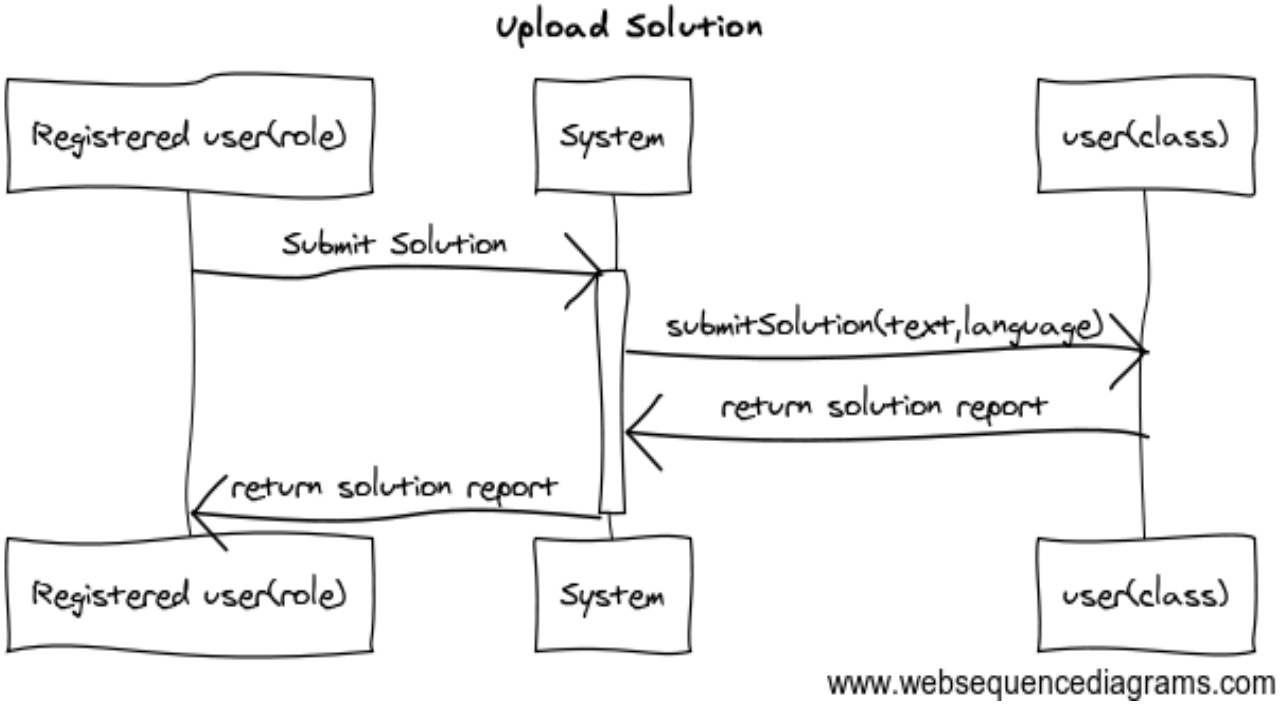
4.4.2 Sequence diagram:



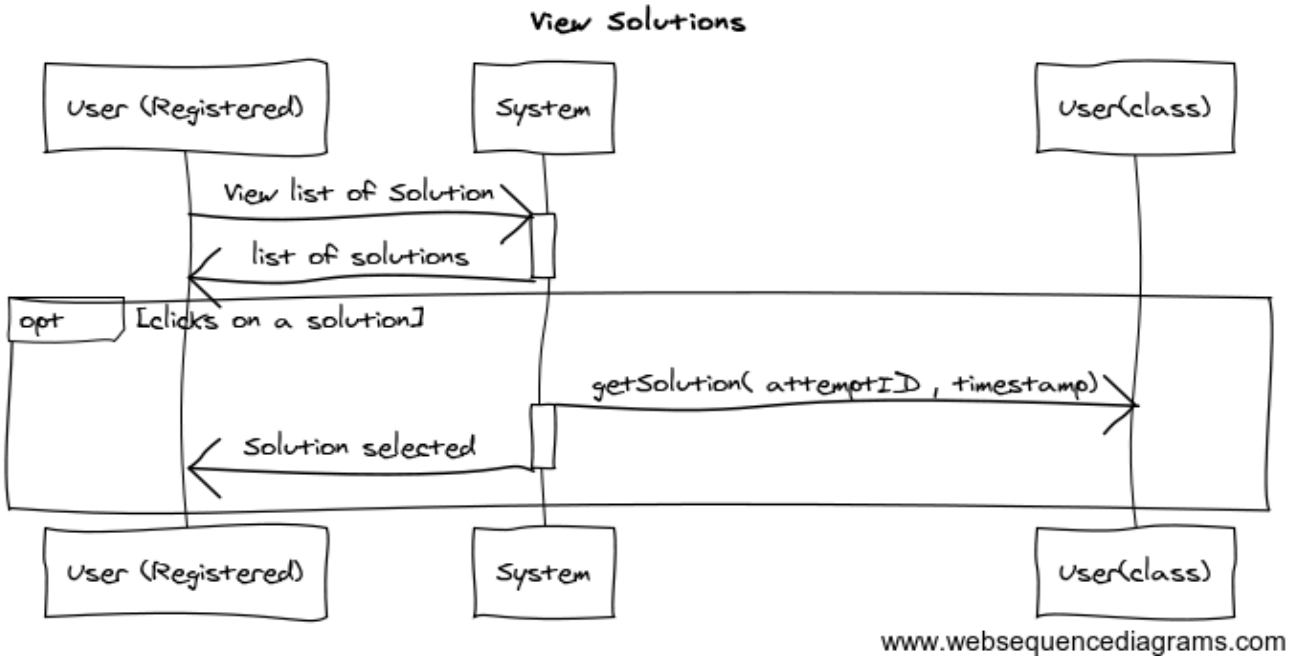
Sequence Diagram for use case "View and/or Edit Profile"



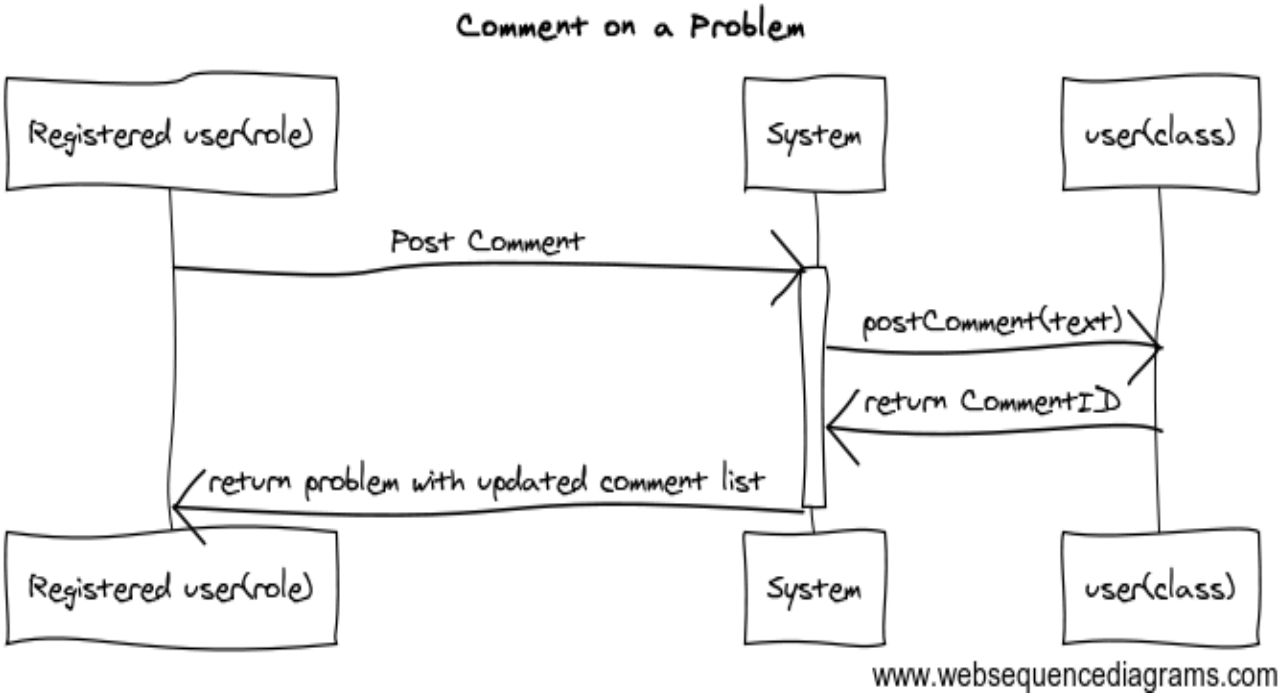
Sequence Diagram for use case "View and/or Edit Profile"



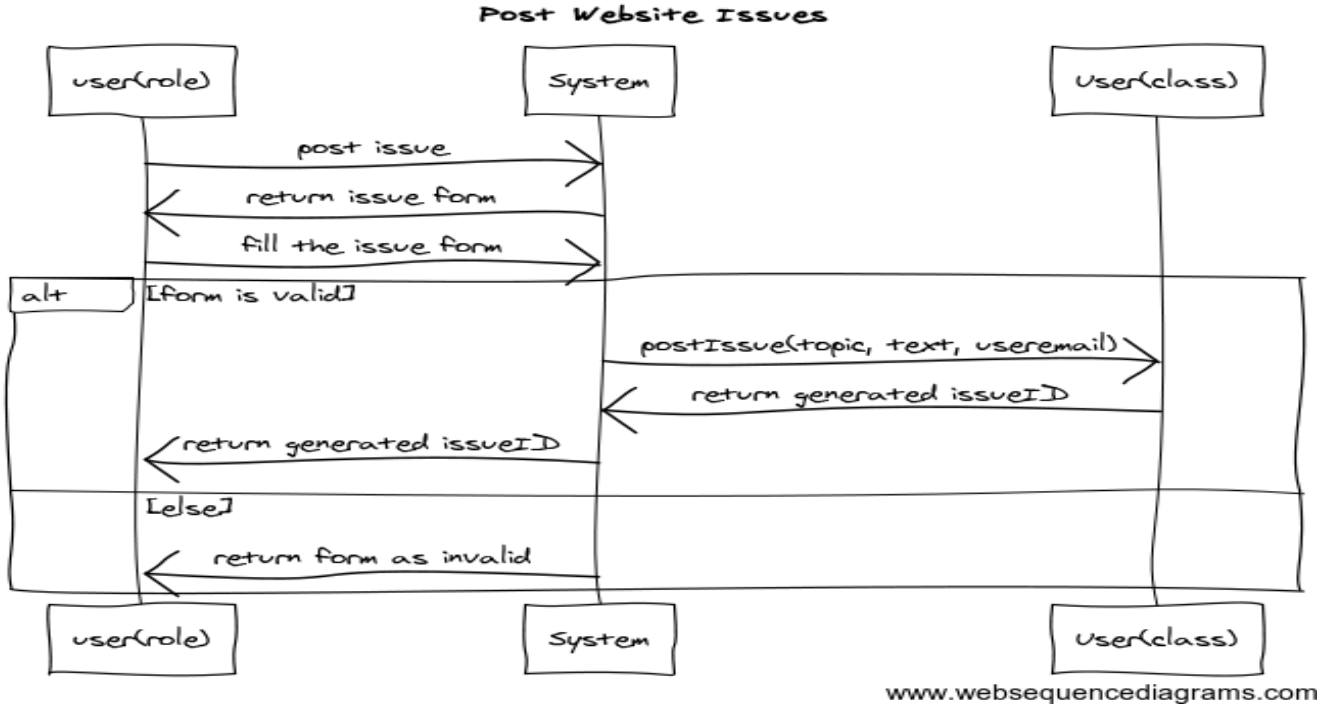
Sequence Diagram for use case "View and/or Edit Profile"



Sequence Diagram for use case "View and/or Edit Profile"



Sequence Diagram for use case "View and/or Edit Profile"



Sequence Diagram for use case "View and/or Edit Profile"

5. Human Interface Design

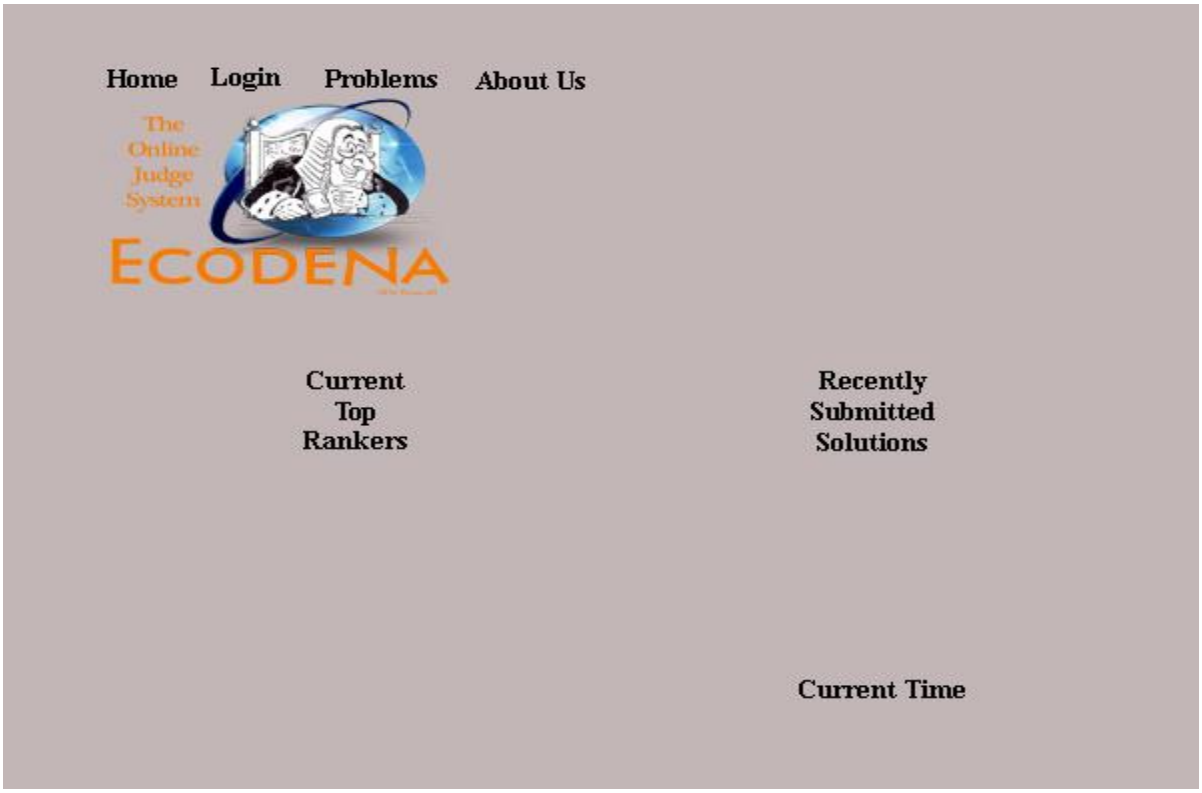
5.1 Overview of user interface:

The website will have an easy to use and intuitive user interface. The user who is not logged in would be able to view problems, list of top users and contact the system admins. In addition to functionalities provided to unregistered user the registered/ logged in users can submit solution of the problem (which would be stored in his/her account), view his/her rank.

Home Page:

The Home page will display dynamically the top users and recent problems for which the solutions have been submitted. It will also have links to different pages for different functionalities like

- Login
- Register
- Problems
- Contact Us



6. Libraries and Tools

Choosing suitable tools for a software projects is a challenging process. Therefore a wide research was made on the tools that can be used in this project.

6.1 Database

In this project, database management system and database itself have a very important role. There will be important data stored in database. After some research on database technologies and detailed results are explained in the following sections.

6.1.1 Postgre:

It implements the majority of the SQL:2008 standard, is ACID-compliant, is fully transactional (including all DDL statements), has extensible data types, operators, and indexes, and has a large number of extensions written by third parties.

6.2 Programming Languages

Python is a general-purpose, high-level programming language whose design philosophy emphasizes code readability. Python claims to combine "remarkable power with very clear syntax" and its standard library is large and comprehensive.

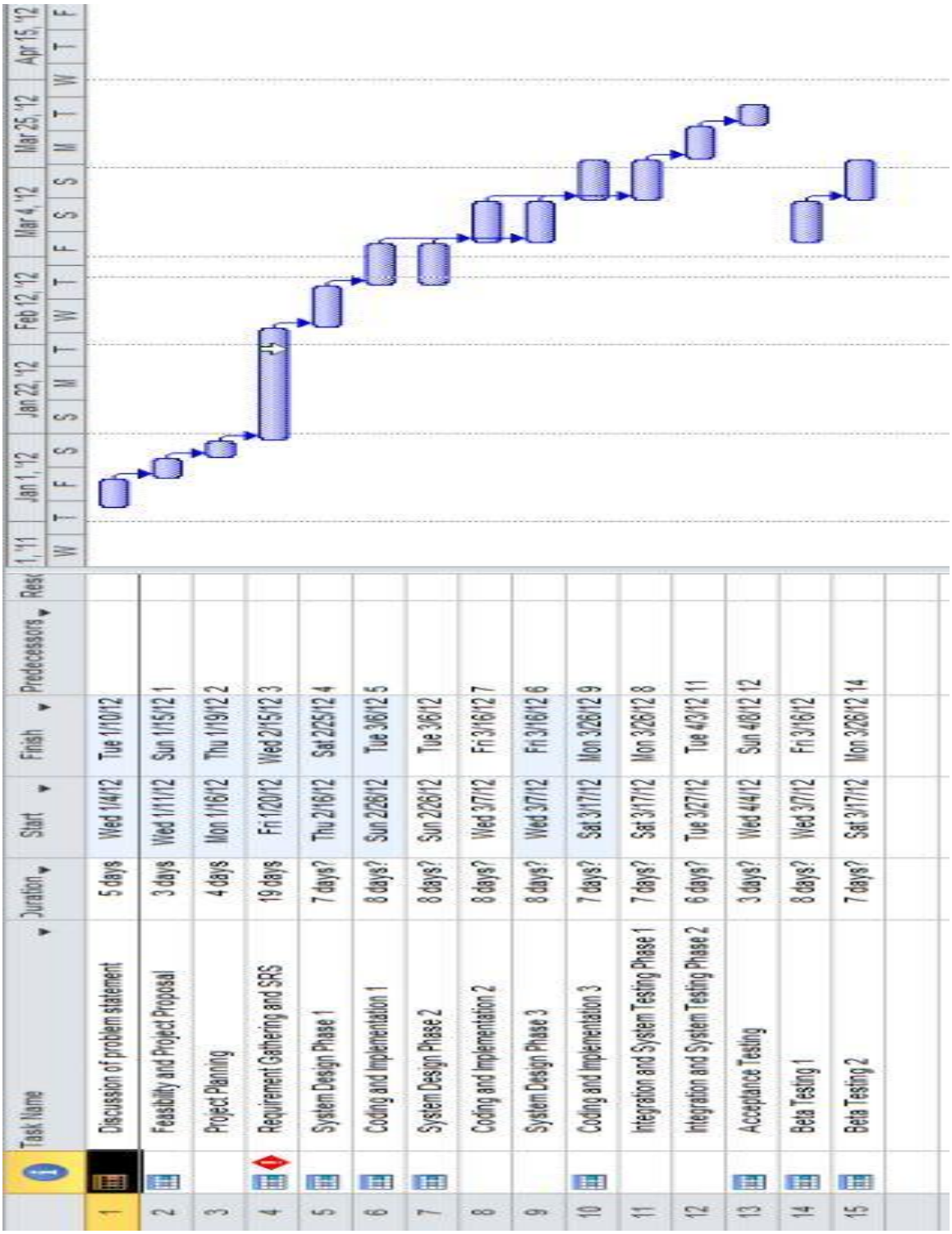
Python supports multiple programming paradigms, primarily but not limited to object-oriented, imperative and, to a lesser extent, functional programming styles. It features a fully dynamic type system and automatic memory management, similar to that of Scheme, Ruby, Perl, and Tcl. Like other dynamic languages, Python is often used as a scripting language, but is also used in a wide range of non-scripting contexts. Using third-party tools, Python code can be packaged into standalone executable programs. Python interpreters are available for many operating systems.

Java is a programming language originally developed by James Gosling at Sun Microsystems (which has since merged into Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is a general-purpose, concurrent, class-based, object-oriented language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java is currently one of the most popular programming languages in use, particularly for client-server web applications, with a reported 10 million users.

6.3 Working Environment

As a working environment mostly we will use Linux operating systems. Moreover, IDEs such as Eclipse and editor Vim will be used by the SEN Team #3 members.

7.Time Planning(Gantt chart)



8. Conclusion

During the design process, we over viewed all parts of the project and we gained almost all the information that we need. In this report, we gave information about our design process. In the document, determined design constraints and the design requirements of our project. How the design of user interface will be was figured in user interface part and some examples for submenus and screenshots were given. Architectural design of the system and its components were dealt with in next part. The general structure of the *Ecodena* also exists in this part. Activity diagrams were given in this section, also. Finally the team planning is given.

This report has assisted us to perceive the difficulties faced when working on a team based software product. Moreover, it has helped us to gain a deep understanding of what we will code and what we can achieve and this brings us one step closer to the completion of our project.

9. The Nine Unknown Men

Team Leader - Avadhesh Gadia (200901028)

Pankaj Bhambhani (200901047)

Arunanghshu Bhakta (200901026)

Mukesh Makwana (200901032)

Harsh Chawada (200901052)

Unique Jain(200901036)

Anand Mudgerikar (200901056)

Jay Oza (200901014)

Chinmay Modi (200901039)

Index