Cyberbullies
Coding Standards

1 Introduction

This document outlines the standards, rules, and conventions that each member of the software development team will adhere to during the Cyberbullies project's software-building phase. These standards include organizing file structure and naming conventions for classes, methods, and variables, etc.

2 Description

Implementing a coding standard is essential in collaborative software development environments. It ensures a standardized approach to naming, formatting, and organizing code, facilitating quick comprehension and easy collaboration among team members. By involving the team in establishing these standards, allows anyone to understand and work with code written by others. This document represents the team's consensus on rules and conventions, governing code style, structure, and comments. Adhering to these guidelines minimizes the time required to understand and modify code and reduces software defects.

3 Coding Standards Specifications

3.1 Naming standards

3.1.1 Classes: Each class name should be in ***PascalCase***. If the class is a part of the service code, it should end with the noun “Service”, for example “class UserService”.

3.1.2 Methods: Each method name should be in ***camelCase***.

3.1.3 Variables: Each local or class variable name should be in camelCase. Parameter names should also be in ***camelCase***.

3.1.4 Constants: Each constant field should be named in ***CONSTANT_CASE.***

3.1.5 Packages: Each package in the folder structure should be named in ***lowercase*** letters. Consecutive words should be concatenated together without any uppercase letters or underscores.

Example: **com.cyberbullies.iceshu4.repository**

3.2 File organization

3.2.1 File Names: There should only be one class in each source file in the file system, and it should be named in ***PascalCase.***

3.2.2 Folder Structure: The source file tree's folders should follow the Java language definition for package names. Layer-by-layer organization of the packages means that logically, source code from the same layer should be placed inside the same package.

3.3 Comment standards

3.3.1 Line Comment: The first letter of each line comments in the source code should be capitalized, and there should be 1 whitespace after "//". If the line comment comes after a line of source code, there should be two whitespaces before "//" .

3.3.2 Inline Comment: The first character of inline comments in source code should be lowercase, and there should be one space between "/*" and "*/" before and after them.

3.4 Coding conventions

3.4.1 If Statements: When there are more than two nested if clauses, the design should be questioned. The cognitive complexity of the source code and the work required of the reader are both significantly increased by nested if clauses.

3.4.2 Loops: Whenever there are more than three nested loops, the design should be questioned.

3.5 White space

3.5.1 Indentation: Spaces should be used to create the indentation, which should be 4 whitespaces long.

4 Traceability Table

Traceability Table	**Umut Güngör**	**Numan Kafadar**	**Mustafa Çağrı Korkmaz**	**Yunus Emre Terzi**	**Osman Faruk Derdiyok**
Introduction			X		
Description			X		
Coding Standards Specifications			X		