Course Code : CSE 404

Course Title : Software Engineering and ISD Laboratory

Project name : Bus ticket booking management system

Experiment no: 09

Experiment name: Applying Coding Standard principles to the Java/C# Code Generated from UML Class Diagram

Submitted To

Dr. Mohammad Zahidur Rahman
Professor
Dr. Md. Humayun Kabir
Professor
Department of Computer Science & Engineering
Jahangirnagar University, Savar.

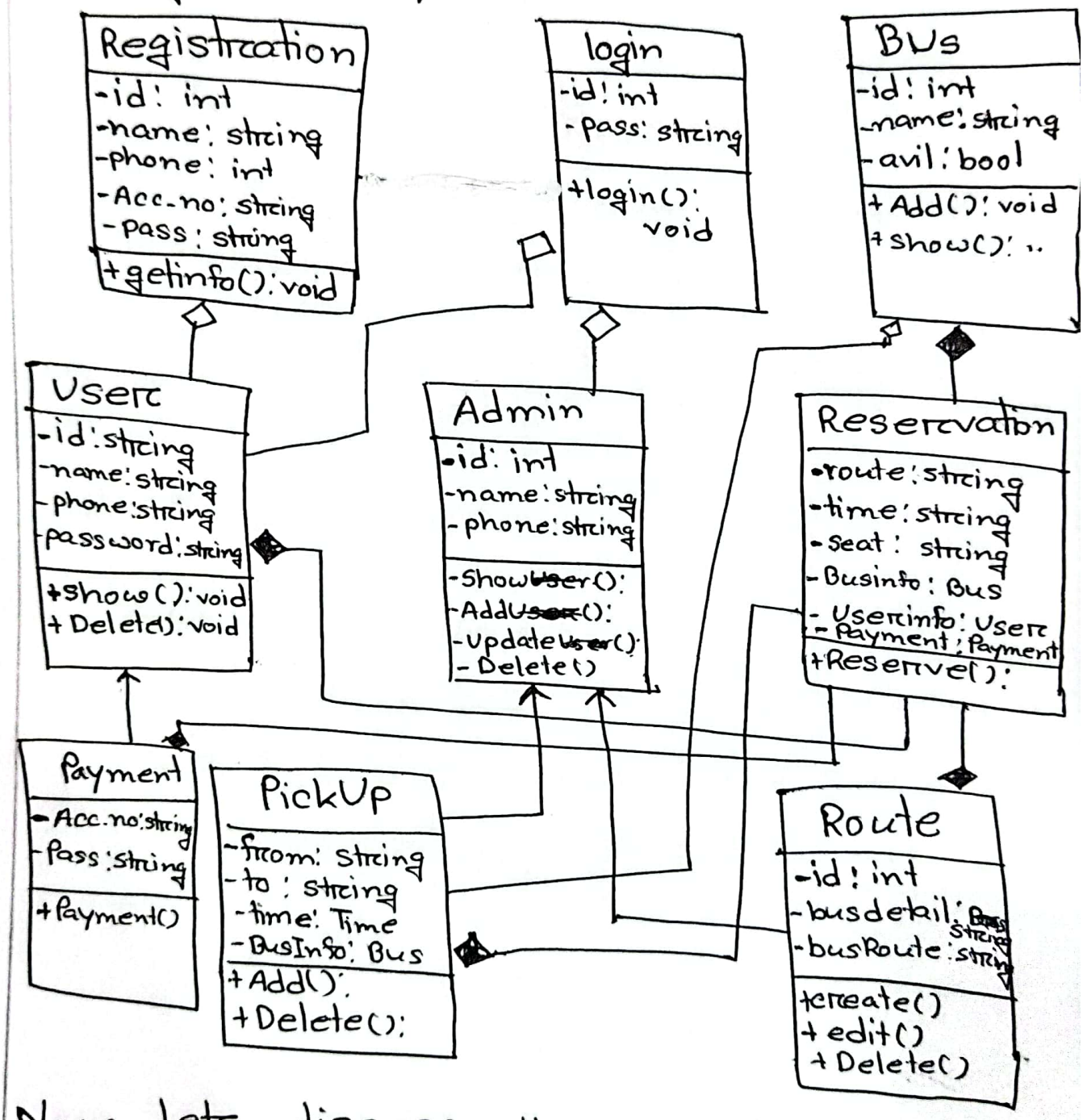
Group No : 02

Group members :

Sl	Class Roll	Name
01	342	Tama Shil
02	370	Prokash Maitra
03	374	Mubasher adnan jihad
04	375	Pritam Saha

Tama Shil
ID - 342
ExamRoll - 191324

The UML class diagram of bus ticket management system-



Now lets discuss the coading standard forc bus ticket management system.

Name: Pritam Saha
ID: 375

Title: Applying coding standards to Java principles to the Java/c# code Generated from UML Class Diagram

Appying coding standards to Java or C# code generated from a UML class diagram for a bus ticket management system is crucial for ensuring code consistency, readability, maintainability and collaboration among developers.

Lets provide more detaild guidelines for appying coding standared principles to the Java and C# code generated from a UML class diagram for a bus ticket management system.

Naming Conventions:

1. Classes:
- Use clear, descriptive names for classes, reflecting their purpose
- Follow Pascal case for class names.

Example (Java):

```
class BusTicketManager {
  //
}
```

2. Variables and Fields
- Use meaningful, camelcase names for variables and fields
- Prefix member variables with 'this' (Java) for distinguishing them from local variables.

Example (Java):

```
privale String passengerName;
```

3. **Methods**:
- Use descriptive names for methods that reflect their actions.
- Follow camelcase for method names

Example (Java):

```
Public void bookTicket (String passengerName)
//..
}
```

Name : Mubasher Adnan Jihad
Roll : 374

Experiment No : 9

4. ** Constants **
   - Use uppercase letters for constants.
   - Separate words with underscores (Java) or use Pascal case (C#) for constant names.

Example (Java):
```
public static int MAX_TICKETS = 10;
```

Code formatting and style

5. Indentation and Bracing
   - Use consistent indentation (typically 4 spaces)
   - Place opening braces on the same line (Java) or a new line (C#)
   - Maintain consistent brace style throughout the codebase.

Example (Java):
```
if (condition) {
  // codeblock
} else {
    // codeblock
}
```

6. ** Comments and documentation **

- Add comments to clarify complex logic, especially if it's not immediately obvious.
- Use JavaDoc (Java) or XML comments (C#) to document methods and classes for auto generating documentation.

Example (Java):

```
/*
 * This method books a ticket for the given
   passenger.
 * @ passengerName The name of the passenger.
 */
public void bookTicket (String passengerName)
{
  //....
}
```

Name: Prokash Maitra

Roll : 370

Exam Roll: 191372

Experiment: Applying Code standard principles to the Java/C# code generated from UML Class diagram of Bus Ticket Management System

---

Code structure and organization:

7. **Modularity and Single Responsibility**:
   - Follow the single Responsibility Principle (SRP). Keep classes focused on one task.
   - organize code into logical packages or namespace
   - Use appropriate access modifiers (public, private) to control visibility and enforce encapsulation.

8. ** Code Reusability **
   - Encapsulate reusable code into functions, methods, or libraries to promote code reuse and maintainability
   - Avoid duplicating code.

Error Handling and Exception Handling:

9. ** Error Handling ** :

- Implement proper error handling using try-catch blocks.

- Provide meaningful error messages or log exceptions for debugging.

Example (Java):

```
try {
  // Code that may throw an exception
}
catch (SomeException e) {
    // Handle the exception or log it
}
```

Tama Shil

ID - 342

ExamRoll - 191324

10. Version Control and Collaboration:

- Use a version control system (Git) for code management
- Collaborate with team members through version control repositories.
- Follow branching and merging strategies as needed.

Code Analysis Reviews:

11. Static Code Analysis -

- Utilize static code analysis tools (Checkstyle for Java, StyleCop for C#) to enforce coding standard automatically.

12. Code Reviews

- Conduct code reviews with team members to ensure adherence to coding standards and identify potential improvements.

By adhering to these coding standard principles and guidelines, you can produce well-structured, readable and maintainable code for bus ticket management system, regardless of whether it is generated from a UML class diagram or written manually. Consistency in coding standards helps improving collaboration among developers and reduces the chances of introducing bugs and reports.

—o—