UTA Department of Computer Science and Engineering

CSE 1325 Project Documentation

AEGIS: A CAMPUS SAFETY COMPANION

Students names, surnames, IDs:

1. Kripa Neupane, 1002203947
2. Dikshya Laudari, 1002227124
3. Madhurima Shrestha, 1002209924

Mentor: Marika Apostolova

Object oriented programming CSE 1325

Student declaration:

We declare that:

- *We understand what is meant by plagiarism*
- *The implication of plagiarism has been explained to me by our professor*
- *This assignment is all team own work and we have acknowledged any use of the published and unpublished works of other people.*

1 Student name, surname, ID and **signature:** Madhurima Shrestha, 1002209924..**…….....**
2 Student name, surname, ID and **signature:** Dikshya Laudari, 1002227124....**…….....**
3 Student name, surname, ID and **signature:** Kripa Neupane, 1002203947..**…….....**

Date:…11/22/2025…

	Total number of pages including this cover page	11
Class Code / Group	CSE 1325	
Lecturer's Name	MARIKA APOSTOLOVA	

Table of Contents

1. **Project Introduction**

   A Campus Safety Companion is a console-based Java application that enhances campus safety for UTA students. It allows users to request walking companions, report non-emergency hazards, and access emergency guidance. The system simulates a real campus-safety workflow using UTA credential verification, role-based dashboards (Student, Companion, Admin), and backend logic that can later be expanded into a GUI or mobile app.

2. **Project Description**

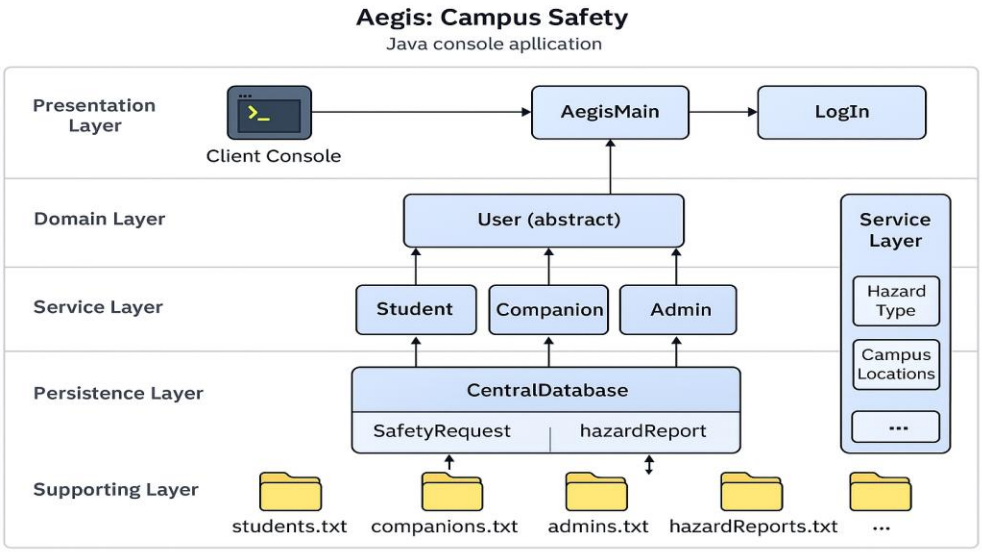
   Aegis promotes campus safety by giving students a structured way to request help and report hazards. The system features:

   a) Allows students to request a walking companion from a pool of screened volunteers.
   b) Hazard Reporting for non-emergency issues.
   c) Provides emergency guidance and escalation when keywords indicate urgent danger

   To achieve these goals, the project is divided into several logical phases:

   i. **User Account and Role Management**
      Users can login or signup as student, companion, or admin.
   ii. **Core Student Features**
      Companion requests, virtual check-ins, hazard reporting, and emergency handling are implemented in the Student class.
   iii. **Companion and Admin Workflows**
      Companions can track availability, training, approval status, and completed walks. Admins are responsible for reviewing applications, viewing hazard reports, and intervening when needed.
   iv. **Matching and Hazard Management**
      The CentralDatabase: submit and manage SafetyRequest and hazardReport objects by implementing matching algorithm (student requests with companions). Hazard reports are stored and can be reviewed, escalated, or updated in status.

3. **Project Architecture**



The Aegis system follows a modular, layered architecture:

a) Presentation / Interaction Layer
   - AegisMain: Manages startup and role selection.
   - LogIn: handles account creation, authentication, and file I/O.

b) Domain / Business Logic Layer
   - User (abstract class): Common parent for all user types, encapsulating core attributes, and abstract methods.
   - Student, Companion, and Admin extend User with role-specific behavior.

c) Data Management Layer
   - CentralDatabase: Manages in-memory maps for users, hazard reports, and safety requests, plus the companion-matching algorithm.

d) Supporting Models
   - SafetyRequest, HazardReport, and enums like HazardType and CampusLocations.

4. UML Class Diagram

User is the superclass for Student, Companion, and Admin. CentralDatabase interacts with all three roles and stores SafetyRequest and hazardReport objects. Enums help model fixed categories and campus locations. Enums (HazardType, CampusLocations) help model fixed categories.

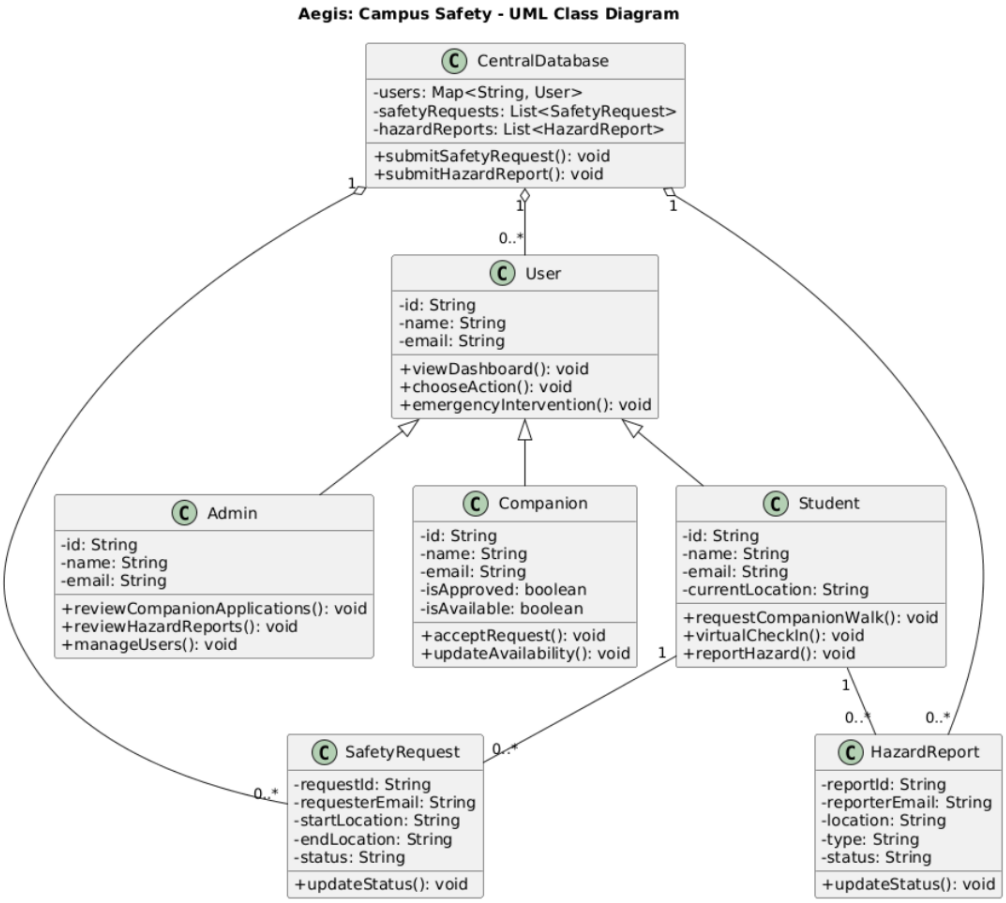


Figure 1: UML class diagram

5. OOP Concepts Used

a) **Classes and Objects**: The system is built around well-defined classes such as User, Student, Companion, Admin, CentralDatabase, SafetyRequest, and hazardReport.

b) **Encapsulation**: Each class hides its internal data using private fields and exposes only necessary getters, setters, and public methods.

c) **Inheritance**: Abstract User class is the parent of Student, Companion, and Admin. Common properties and behaviors (e.g., name, email, MAVID, gender, basic setters) are defined once in User, while each subclass extends with additional fields and role-specific methods.

d) **Abstraction**: Abstract methods viewDashboard(), chooseAction(), and EmergencyIntervention() in the User class; each subclass provides its own implementation

e) **Polymorphism**: Because Student, Companion, and Admin all extend User, the program can use polymorphism to treat them as User references while still calling overridden methods.

f) **Enums and Type Safety**: The HazardType and CampusLocations enums for type-safe input.

g) **Collections and Data Structures**: The CentralDatabase class uses Map<String, User>, Map<String, hazardReport>, and Map<String, SafetyRequest> to store and retrieve objects.

h) **File I/O and Exception Handling**: The LogIn class: BufferedReader and BufferedWriter to read and write files. I/O operations are wrapped in try–catch blocks to handle IOException.

6. Code Description

a) **Feature 1**: **Student Hazard Reporting with Emergency Keyword Detection**

Student's ReportHazard() collects hazard details and uses checkEmergencyKeywords() to detect urgent words such as fire, gun, or injured. If detected, the system triggers an emergency escalation before submitting the report.

```
public void ReportHazard() {
      System.out.println("Available Hazard Types: ");
      for(HazardType hazardType : HazardType.values()) {
        System.out.println("- " + hazardType);
       }
        System.out.println("Enter the type of hazard.");
        String hazardstr = input.nextLine().trim().toUpperCase();
        hazard = HazardType.valueOf(hazardstr);

      for(CampusLocations location : CampusLocations.values()) {
        System.out.println("- " + location);
       }
        System.out.println("Enter the location of the hazard. ");
        String locationstr = input.nextLine().trim().toUpperCase();

      System.out.println("Enter a detailed description of the hazard");
      String description = input.nextLine().trim();

      if(checkEmergencyKeywords(description)) {
        reportToUtaPD("Escalated Hazard Report");
      }
```

checkEmergencyKeywords() method to check if description has emergency words:

```
private boolean checkEmergencyKeywords(String description) {
      String lowerDesc = description.toLowerCase();
      return lowerDesc.contains("fire") ||
            lowerDesc.contains("gun") ||
            lowerDesc.contains("weapon") ||
            lowerDesc.contains("attack") ||
            lowerDesc.contains("injured");
    }
```

b) **Feature 2**: **Advanced authentication through UTA Email Domain Verification during Login**
User.verifyUTADomain() ensures only emails ending in @mavs.uta.edu or @uta.edu can register or log in.

```
public static boolean verifyUTADomain(String email) {
      return Pattern.compile(".*@(mavs\\.)?uta\\.edu$", Pattern.CASE_INSENSITIVE).matcher(email).matches();
  }
```

c) Feature 3: Companion Request and Matching Algorithm
Student.CcompanionWalk() creates a SafetyRequest, and CentralDatabase.submitSafetyRequest() processes it. The matching algorithm finds a trained, approved, and available companion (optionally same-gender) and assigns them to the student.

```
public void CompanionWalk() {
       for(CampusLocations location : CampusLocations.values()) {
             System.out.println("- " + location);
          }
       System.out.println("What is your current location? ");
       String currentLocation = input.nextLine().trim().toUpperCase();
       System.out.println("Where are you walking to?");
       String destination = input.nextLine().trim().toUpperCase();
        startLocation = CampusLocations.valueOf(currentLocation);
        endLocation = CampusLocations.valueOf(destination);

          SafetyRequest newRequest = new SafetyRequest(this.getEmail(), this.getFirstname(), requestedGender,
startLocation, endLocation);
      userCentralDatabase.submitSafetyRequest(newRequest);
      viewDashboard();
    }
```

SafetyRequest from CentralDatabase class:

```
public void submitSafetyRequest(SafetyRequest request) {
      safetyRequests.put(request.getRequestID(), request);
      System.out.println("Safety Request " + request.getRequestID() + "  submitted by " + request.getRequesterName() +
                  "(" + request.getRequesterEmail() + ")" + " at " + request.getStartLocation() + ". Searching for match.");
      matchCompanion(request);
  }
```

7. System Testing

This section includes screenshots showing successful companion account creation and login, role-based dashboards, admin approval system, companion requests from student, and final companion matching. This summarizes major functionality of our project.

Companion's dashboard: during account creation, approval status is denied by default until admin approves.

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
*                                  Welcome to Aegis!                                            *
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

What best describes your role (Student, Companion, Admin)?
companion
Do you already have an account (Y/N)
n
Please enter your email:
rachel@uta.edu
Please enter your password:
rachel123
Please enter your MAVID:
1234567890
Please enter your first name:
Rachel
Please enter your last name:
Green
Please enter your gender:
f
Companion account created successfully! You can now log in.
Please enter your email:
rachel@uta.edu
Please enter your password:
rachel123
Rachel's approval status has been set to: DENIED
Rachel's training status has been set to: INCOMPLETE
Rachel is now UNAVAILABLE
Sorry you have not been approved or trained yet.
```

Admin's dashboard: reviewing the companion's UTA PD verification and training status for approval of application

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
*                                Welcome to Aegis!                                *
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

What best describes your role (Student, Companion, Admin)?
admin
Do you already have an account (Y/N)
y
Please enter your email:
kxn@uta.edu
Please enter your password:
12345
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
*                    Welcome to Aegis, Kripa N!                    *
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

   Which action would you like to perform?
1. Review and Approve Companion Applications
2. Review Hazard Reports
3. Post Community Alerts
4. View User Statistics
5. Exit
Enter your choice (1-4)
1
Enter the email of the Companion that you would like to review.
rachel@uta.edu
Has Rachel Green
 --> passed UTA PD verification? (true/false)
true
Rachel's approval status has been set to: APPROVED
 --> completed self-defense training?  (true/false)
true
Rachel's training status has been set to: COMPLETE
```

Student's dashboard: upon requesting a companion, Aegis matches them with approved companion

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
*                              Welcome to Aegis!                                *
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

What best describes your role (Student, Companion, Admin)?
student
Do you already have an account (Y/N)
y
Please enter your email:
ava@mavs.uta.edu
Please enter your password:
ava123
ava@mavs.uta.edu
ava123
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
*                    Welcome to Aegis, Ava Morgan!                    *
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

   Which action would you like to perform?
1. Request a Companion
2. Initiate a Virtual Check-in
3. Report Hazard
4. Emergency Intervention
5. Exit
Enter your choice (1-5)
1
Companion Request has been activated for Ava Morgan
Your options are:
- LIBRARY
- UNIVERSITY_CENTER
- MAC
- BRAZOS_PARK
- ARLINGTON_HALL
- TIMBER_BROOK
- ARBOR_OAKS
- WEST_HALL
- KC_HALL
- MAVERICK_HALL
- BOOKSTORE
- WEST_CAMPUS_GARAGE
What is your current location?
library
Where are you walking to?
mac
Safety Request SR-5000  submitted by Ava(ava@mavs.uta.edu) at LIBRARY. Searching for match.
Do you want to request same-gender companion? Y/N
y
rachel@uta.edurachel@uta.edu
dxl@mavs.uta.edudxl@mavs.uta.edu
mxs@mavs.uta.edumxs@mavs.uta.edu
kxn@mavs.uta.edukxn@mavs.uta.edu

Companion Found! Your Companion is: Rachel Green
Rachel is now UNAVAILABLE
Rachel has completed one more walk! Total walks completed: 1
```

8. **Group members**
   - **Kripa Neupane**: Class-specific core code implementation, Email verification, File & Error handling, Matching algorithm
   - **Madhurima Shrestha**- User input-output and I/O files formatting, Debugging and Testing, Code refinement, Report
   - **Dikshya Laudari**- Debugging and Testing, Code refinement, Report

9. **Conclusion and future works**

   The Aegis project demonstrates how object-oriented programming in Java can be used to model a realistic campus safety system. By separating user roles into Student, Companion, and Admin, and centralizing data and operations in CentralDatabase, the application shows how role-based behavior, data structures, and control flow can be combined to solve a real-world problem. Students can request companions, report hazards, and access emergency guidance through a straightforward console interface, while the system enforces UTA email verification and uses enums and collections to maintain data consistency.

   In the future, Aegis could be extended in several important ways:

   a) Real-time notifications could be added so that companions and admins receive alerts instantly.
   b) Storage of such sensitive user data could be moved from text files to a relational database.
   c) GPS integration, push notifications, and a richer admin dashboard with analytics would make the system more powerful and closer to a production-level campus safety application.
   d) A graphical user interface or a mobile app front-end could replace the console, making the system more user-friendly.

   These improvements would build on the current object-oriented design and allow Aegis to evolve into a comprehensive safety companion for the university community.