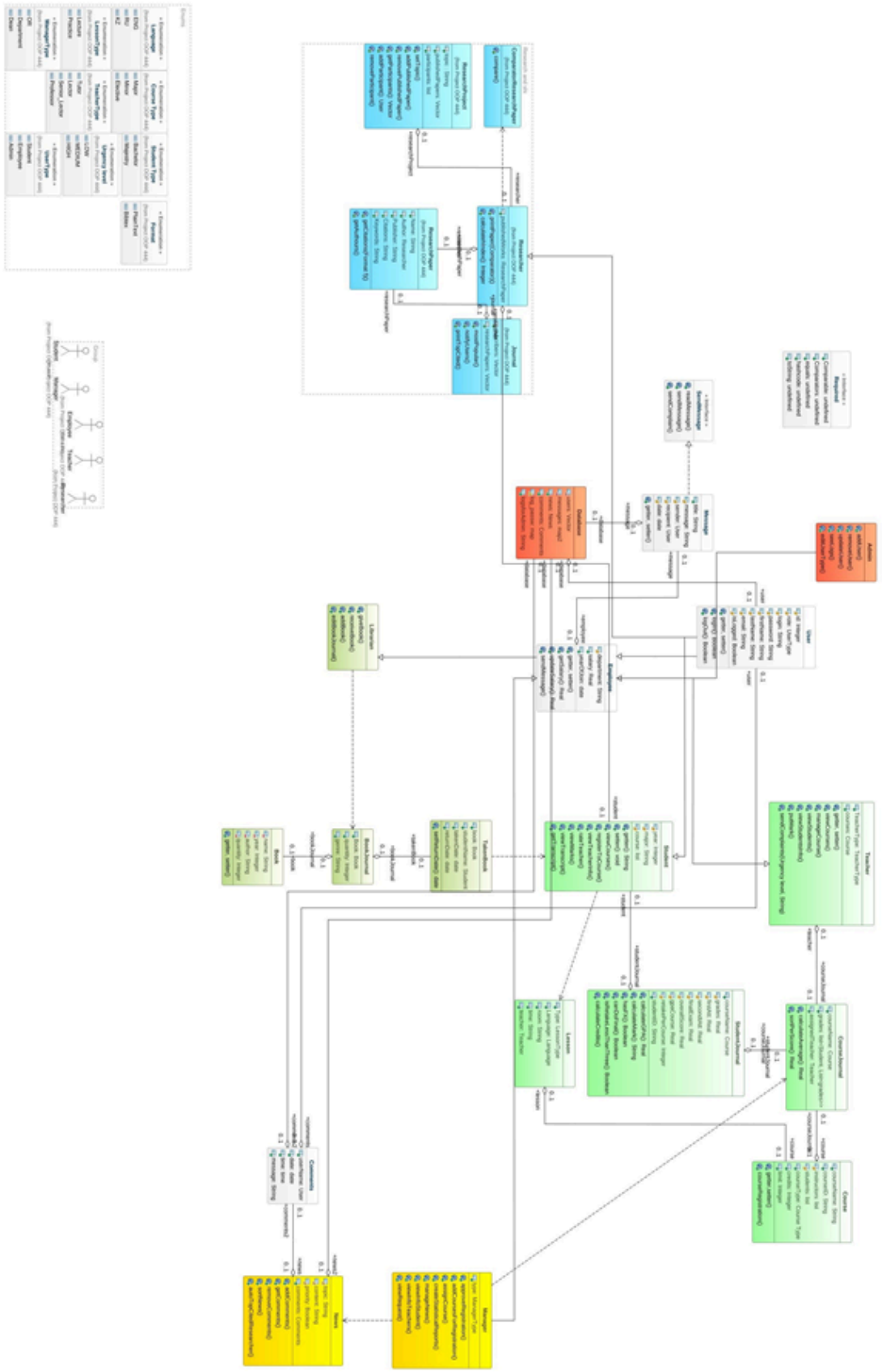
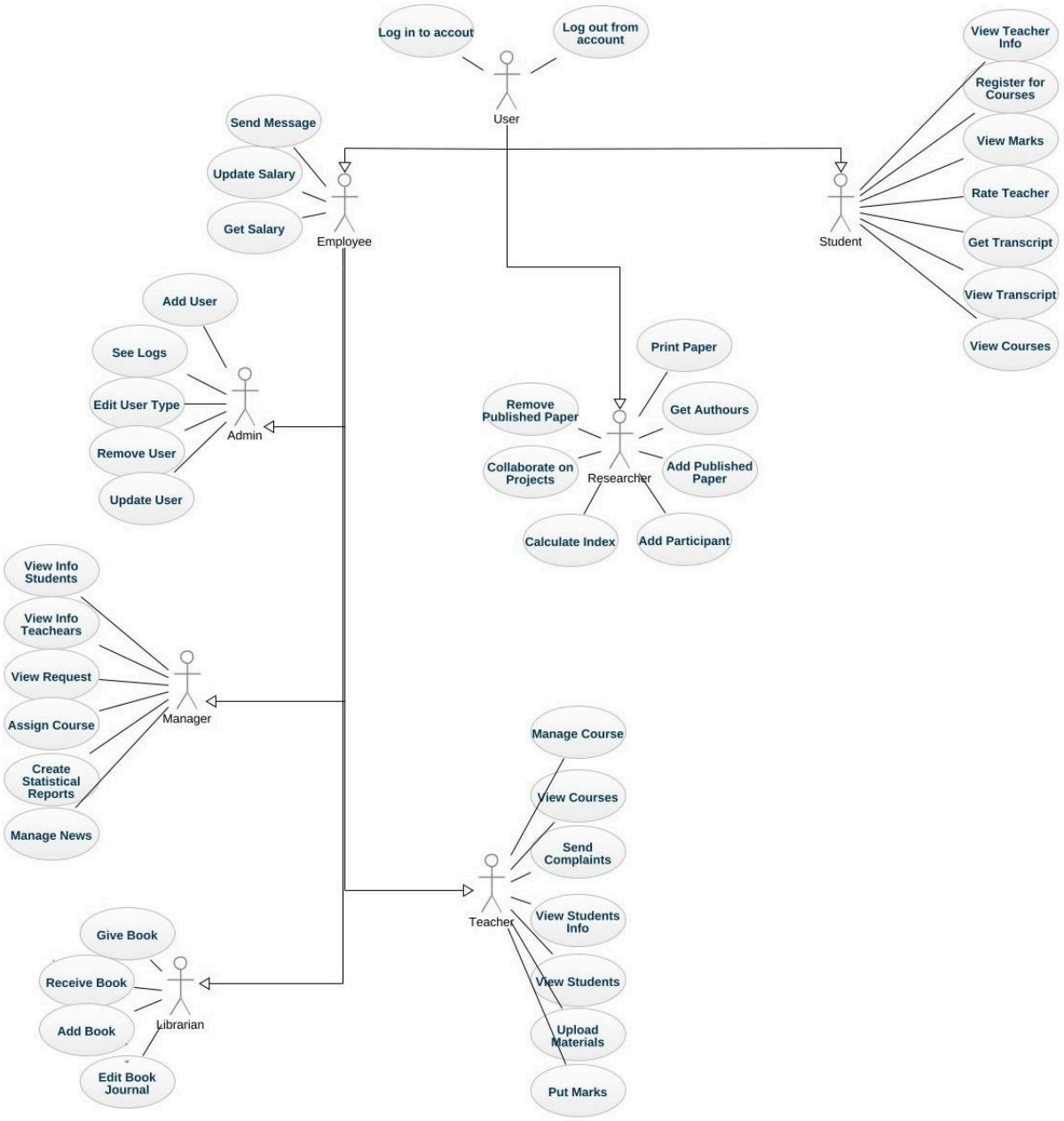
UML DIAGRAM



Use case


Log in to accout
Log out from account
User
Send Message
Update Salary
Get Salary
Employee
View Teacher Info
Register for Courses
View Marks
Rate Teacher
Get Transcript
View Transcript
View Courses
Student
Add User
See Logs
Edit User Type
Remove User
Update User
Admin
Print Paper
Remove Published Paper
Get Authours
Collaborate on Projects
Researcher
Add Published Paper
Calculate Index
Add Participant
View Info Students
View Info Teachears
View Request
Assign Course
Create Statistical Reports
Manage News
Manager
Manage Course
View Courses
Send Complaints
View Students Info
View Students
Upload Materials
Put Marks
Teacher
Give Book
Receive Book
Add Book
Edit Book Journal
Librarian

Classes

1. Academic Part

Book

Description: Represents a book with attributes like ID, name, author, and availability.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int bookID`: Unique identifier for the book.
- `private String bookName`: Name of the book.
- `private String author`: Author of the book.
- `private boolean available`: Indicates whether the book is available.

Methods:

- `Book(int bookID, String bookName, String author)`: Constructor to initialize a book.
- `boolean isAvailable()`: Returns the availability status of the book.
- `void setAvailable(boolean available)`: Updates the availability status.
- `int getBookID()`: Returns the book's ID.
- `String getBookName()`: Returns the book's name.
- `String getAuthor()`: Returns the book's author.

Complaint

Description: Represents a user complaint with attributes like text, urgency, and involved users.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int complaintId`: Unique identifier for the complaint.
- `private String complaintText`: Description of the complaint.
- `private UrgencyLevel urgencyLevel`: The urgency level of the complaint.
- `private int complaintSender`: ID of the user who filed the complaint.
- `private int complaintGuilty`: ID of the user responsible for the complaint.

Methods:

- `Complaint()`: Default constructor.

- `Complaint(String complaintText, UrgencyLevel urgencyLevel, int complaintSender, int complaintGuilty)`: Constructor to initialize a complaint.
- `int getComplaintId()`: Returns the complaint's ID.
- `String getComplaintText()`: Returns the complaint's description.
- `UrgencyLevel getUrgencyLevel()`: Returns the urgency level.
- `User getComplaintSender()`: Returns the sender of the complaint.
- `User getComplaintGuilty()`: Returns the user at fault.
- `boolean equals(Object obj)`: Checks for equality with another object.
- `String toString()`: Converts the complaint to a string representation.

Course

Description: Represents a course with attributes such as name, type, and associated instructors and students.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int courseID`: Unique identifier for the course.
- `private String courseName`: Name of the course.
- `private Vector<Integer> instructors`: List of instructors for the course.
- `private Vector<Integer> students`: List of students in the course.
- `private CourseType type`: Type of the course.
- `private int requiredYearOfStudy`: Required year of study for enrollment.
- `private int credits`: Number of credits for the course.

Methods:

- `Course()`: Default constructor.
- `Course(String courseName, CourseType type, int requiredYearOfStudy, int credits)`: Constructor to initialize a course.
- `int getCourseID()`: Returns the course's ID.
- `String getCourseName()`: Returns the course's name.
- `Vector<Teacher> getInstructors()`: Returns the list of instructors.
- `void addInstructor(Teacher instructor)`: Adds an instructor to the course.
- `Vector<Student> getStudents()`: Returns the list of students.
- `void addStudent(int student)`: Adds a student to the course.
- `CourseType getType()`: Returns the course type.
- `int getRequiredYearOfStudy()`: Returns the required year of study.
- `int getCredits()`: Returns the number of credits.

Journal

Description: Represents a journal for storing news and managing subscribers.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int journalId`: Unique identifier for the journal.
- `private Vector<Integer> news`: List of news items in the journal.
- `private Vector<Integer> subscribers`: List of subscribers to the journal.

Methods:

- `Journal()`: Default constructor.
- `int getJournalId()`: Returns the journal's ID.
- `Vector<News> getNews()`: Returns the list of news items.
- `Vector<Student> getSubscribers()`: Returns the list of subscribers.
- `void addSubscriber(int subscriber)`: Adds a subscriber to the journal.
- `void addNews(int newsItem)`: Adds a news item to the journal.

Lesson

Description: Represents a lesson within a course.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int lessonID`: Unique identifier for the lesson.
- `private int course`: The course to which the lesson belongs.
- `private String topic`: The topic of the lesson.
- `private LessonType type`: The type of lesson.

Methods:

- `Lesson()`: Default constructor.
- `Lesson(int course, String topic, LessonType type)`: Constructor to initialize a lesson.
- `int getLessonID()`: Returns the lesson's ID.
- `Course getCourse()`: Returns the associated course.
- `String getTopic()`: Returns the topic of the lesson.
- `LessonType getType()`: Returns the type of the lesson.

StudentOrganization

Description: Represents a student organization with members and an identifier.

Attributes:

- `private int organizationId`: Unique identifier for the organization.
- `private Vector<Integer> members`: List of member IDs in the organization.

Methods:

- `StudentOrganization()`: Default constructor.
- `int getOrganizationId()`: Returns the organization's ID.
- `Vector<Student> getMembers()`: Returns the list of members.
- `void addMember(int student)`: Adds a student to the organization.
- `int getNumberOfMembers()`: Returns the number of members.
- `boolean equals(Object obj)`: Checks if two organizations are equal.
- `String toString()`: Returns a string representation of the organization.

StudentRegistration

Description: Represents a student's registration in a course.

Attributes:

- `private int registartionId`: Unique identifier for the registration.
- `private Integer student`: ID of the student.
- `private Integer course`: ID of the course.

Methods:

- `StudentRegistration()`: Default constructor.
- `StudentRegistration(Integer student, Integer course)`: Constructor to initialize a registration.
- `int getRegistrationId()`: Returns the registration ID.
- `Student getStudent()`: Returns the student associated with the registration.
- `Course getCourse()`: Returns the course associated with the registration.

---

2. Data Part

`DataRepository`

Description: Manages and stores data for the system.

Attributes:

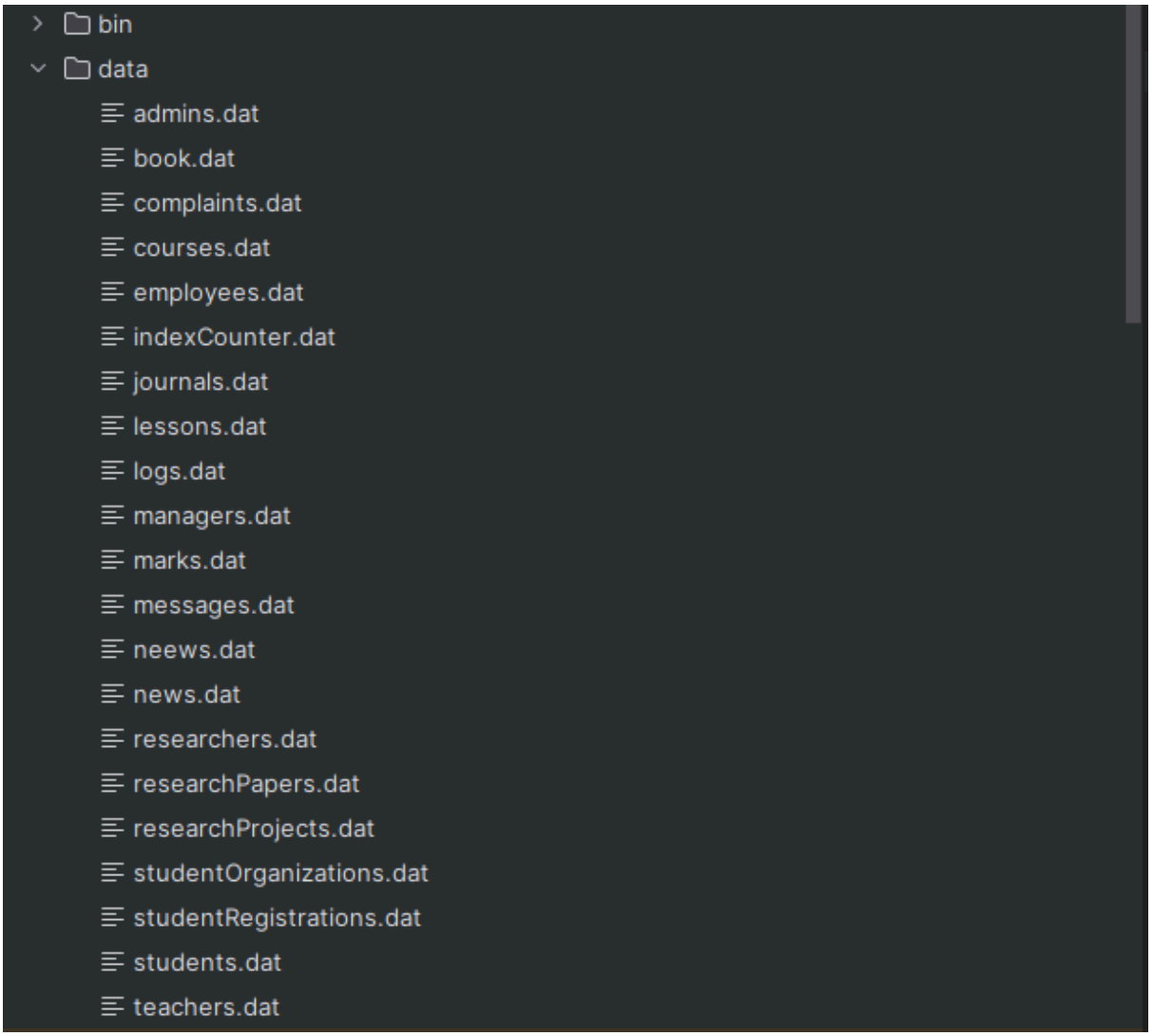
- Various vectors for storing instances, including:
  - `Vector<Employee> employees`
  - `Vector<Student> students`
  - `Vector<Teacher> teachers`
  - `Vector<Admin> admins`
  - `Vector<Manager> managers`
  - `Vector<Course> courses`
  - `Vector<Lesson> lessons`
  - and etc…

Methods:

- `DataRepository()`: Default constructor.
- `int getNextId()`: Returns the next available ID.
- `void pullDataFromDatabase()`: Pulls data from the database.
- `void saveTransactionDataToDB()`: Saves data to the database.
- `User login(String name, String password)`: Logs in a user.
- `void logout()`: Logs out the current user.
- A series of `get`, `add`, and `remove` methods for various entities like `Employee`, `Student`, `Course`, etc.

Realization:

The data storage in the project is implemented using serialization and deserialization. The DataRepository class maintains a dedicated Vector for each class in the project, allowing structured in-memory data management. The serialized data for each class is persistently stored in corresponding **.dat** files within the **data/** directory.


bin
data
admins.dat
book.dat
complaints.dat
courses.dat
employees.dat
indexCounter.dat
journals.dat
lessons.dat
logs.dat
managers.dat
marks.dat
messages.dat
neews.dat
news.dat
researchers.dat
researchPapers.dat
researchProjects.dat
studentOrganizations.dat
studentRegistrations.dat
students.dat
teachers.dat

3. Human Part

`Admin`

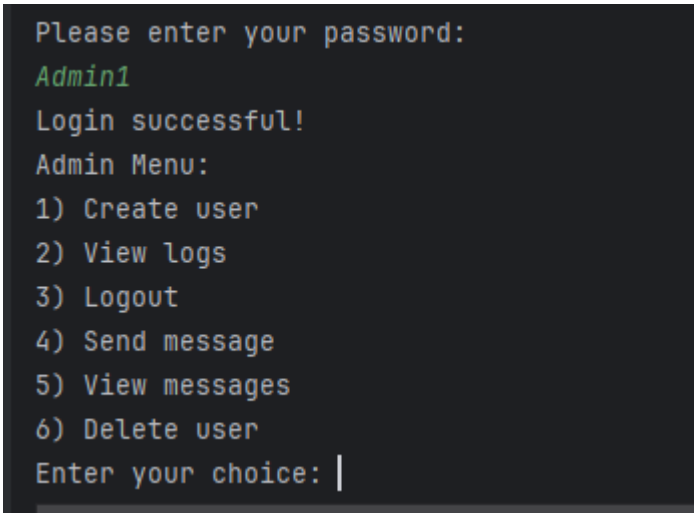
Description: Represents an administrative user with management privileges.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.

Methods:

- `Admin()`: Default constructor.
- `Admin(String name, String email, String password, int salary)`: Constructor to initialize an admin.
- Methods to add and remove users of various roles:
    - `void addEmployee(String name, String email, String password, Boolean isResearcher, int salary)`
    - `void addTeacher(String name, String email, String password, Boolean isResearcher, TeacherTitle teacherTitle, int salary)`
    - `void addStudent(String name, String email, String password, Boolean isResearcher, DegreeType degreeType)`
    - `void addManager(String name, String email, String password, ManagerType managerType, int salary)`
    - `void removeEmployee(Employee employee)`
    - `void removeTeacher(Teacher teacher)`
    - `void removeStudent(Student student)`
    - `void removeManager(Manager manager)`
- `void viewLogs()`: Views system logs.
- `void deleteUser(int userId)`: Deletes a user by ID.



ComparatorByArticlesLength

Description: Comparator class for sorting research papers by article length.

Methods:

- `int compare(ResearchPaper paper1, ResearchPaper paper2)`: Compares two research papers by their length.

ComparatorByCitations

Description: Comparator class for sorting research papers by citation count.

Methods:

- `int compare(ResearchPaper paper1, ResearchPaper paper2)`: Compares two research papers by their citation count.

ComparatorByDate

Description: Comparator class for sorting research papers by date.

Methods:

- `int compare(ResearchPaper paper1, ResearchPaper paper2)`: Compares two research papers by their date.

Employee

Description: Represents an employee with basic attributes such as salary.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int salary`: The salary of the employee.

Methods:

- `Employee()`: Default constructor.

- `Employee(String name, String email, String password, Boolean isResearcher, int salary)`: Constructor to initialize an employee.
- `boolean equals(Object obj)`: Checks for equality with another object.
- `int getSalary()`: Returns the employee's salary.

Librarian

Description: Represents a librarian managing library activities.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.

Methods:

- `Librarian(String name, String email, String password, Boolean isResearcher, int salary)`: Constructor to initialize a librarian.
- `String borrowBook(int studentId, int bookId)`: Borrows a book for a student.
- `String returnBook(int studentId, int bookId)`: Processes the return of a book.
- `String addNewBook(String name, String author, String publicationDate)`: Adds a new book to the library.
- `Vector<Book> viewBorrowedBooks(int studentId)`: Views books borrowed by a student.
- `String removeBook(int bookId)`: Removes a book from the library.

Manager

Description: Represents a manager responsible for administrative tasks.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private ManagerType managerType`: Specifies the type of manager.

Methods:

- `Manager()`: Default constructor.
- `Manager(String name, String email, String password, ManagerType managerType, int salary)`: Constructor to initialize a manager.
- `ManagerType getManagerType()`: Returns the type of the manager.
- `void viewStudentRegistrations()`: Views all student registrations.
- `void approveStudentRegistration(int registrationId)`: Approves a student registration.

- `void createCourse(String courseName, CourseType type, int requiredYearOfStudy, int credits)`: Creates a new course.
- `void assignCourseToTeacher(Course course, Teacher teacher)`: Assigns a course to a teacher.
- `void viewInfoAboutStudents()`: Displays information about students.
- `void viewInfoAboutTeachers()`: Displays information about teachers.
- `void publishNews(String title, String content)`: Publishes news.
- `void createStudentOrganization()`: Creates a student organization.
- `void viewComplaints()`: Views complaints filed in the system.
- `void addJournal()`: Adds a new journal.
- `void publishNewsToJournal(Journal journal, String title, String content)`: Publishes news to a journal.

ResearcherDecorator

Description: Enhances functionality for users involved in research activities.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private Integer hIndex`: The researcher's h-index.
- `private Vector<Integer> researchPapers`: List of IDs of research papers.
- `private Vector<Integer> researchProjects`: List of IDs of research projects.
- `private Integer researchSupervisor`: ID of the research supervisor.
- `private Vector<Integer> borrowedBooks`: IDs of borrowed books.

Methods:

- `ResearcherDecorator()`: Default constructor.
- `ResearcherDecorator(Integer decoratedUser)`: Constructor to wrap a user as a researcher.
- `ResearcherDecorator getResearchSupervisor()`: Returns the research supervisor.
- `void calculateHIndex()`: Calculates the researcher's h-index.
- `Integer getHIndex()`: Returns the h-index.
- `void submitResearchPaper(String title, String authors, String journal, int pagesNumber, String publicationDate, String doi)`: Submits a new research paper.
- `void joinResearchProject(ResearchProject project)`: Joins a research project.
- `void assignSupervisor(ResearcherDecorator supervisor)`: Assigns a supervisor.
- `void createResearchProject(String topic)`: Creates a new research project.
- `void addPaperToProject(ResearchProject project, ResearchPaper paper)`: Adds a paper to a project.

- `void printPapers(ResearchPaperComporator comparator)`: Prints research papers using a comparator.
- `Vector<ResearchPaper> getPapers()`: Returns the list of research papers.
- `Vector<ResearchProject> getProjects()`: Returns the list of research projects.
- `void createLogRecord(String text)`: Creates a log record.
- `void borrowBook(Book book)`: Borrows a book.
- `Vector<Book> getBorrowedBooks()`: Returns the list of borrowed books.

Student

Description: Represents a student with attributes such as enrolled courses, marks, and organizations.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private Vector<Integer> enrolledCourses`: List of IDs of enrolled courses.
- `private int credits`: The number of credits earned.
- `private Vector<Integer> marks`: List of IDs of marks.
- `private Vector<Integer> studentOrganizations`: List of IDs of student organizations.
- `private DegreeType degreeType`: The degree type of the student.
- `private Vector<Book> borrowedBooks`: List of borrowed books.

Methods:

- `Student()`: Default constructor.
- `Student(String name, String email, String password, Boolean isResearcher, DegreeType degreeType)`: Constructor to initialize a student.
- `Vector<Book> getBorrowedBooks()`: Returns the list of borrowed books.
- `Vector<Course> getEnrolledCourses()`: Returns the list of enrolled courses.
- `int getCredits()`: Returns the number of credits.
- `void viewTranscript()`: Views the student's transcript.
- `void viewCourses()`: Views available courses.
- `void viewMarks()`: Views the student's marks.
- `void registerForCourse(Course course)`: Registers for a course.
- `void rateTeacher(Teacher teacher, double mark)`: Rates a teacher.
- `Vector<StudentOrganization> getStudentOrganizations()`: Returns the student's organizations.
- `DegreeType getDegreeType()`: Returns the student's degree type.
- `void subscribeToJournal(Journal journal)`: Subscribes to a journal.
- `void applyForStudentOrganization(StudentOrganization studentOrganization)`: Applies for a student organization.

User

Description: Represents a general user in the system with attributes like name, email, and borrowed books. **Abstract class.**

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int userId`: Unique identifier for the user.
- `private String name`: Name of the user.
- `private String email`: Email of the user.
- `private String password`: Password of the user.
- `private Boolean isResearcher`: Indicates if the user is a researcher.
- `private Vector<Book> borrowedBooks`: List of books borrowed by the user.

Methods:

- `User()`: Default constructor.
- `User(String name, String email, String password, Boolean isResearcher)`: Constructor to initialize a user.
- `int getUserId()`: Returns the user ID.
- `String getName()`: Returns the user's name.
- `String getEmail()`: Returns the user's email.
- `Boolean getIsResearcher()`: Checks if the user is a researcher.
- `void addBorrowedBook(Book book)`: Adds a borrowed book to the list.
- `void removeBorrowedBook(Book book)`: Removes a borrowed book from the list.
- `Vector<Book> getBorrowedBooks()`: Returns the list of borrowed books.

UserDecorator

Description: Extends the functionality of a `User`.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.

Methods:

- `UserDecorator()`: Default constructor.
- `UserDecorator(Integer decoratedUser)`: Constructor to wrap a user for decoration.
- `User getDecoratedUser()`: Returns the decorated user.

---

4. Language Part

`LanguageSettings`

Description: Manages language-related settings for the system.

Attributes:

- `private static LanguageSettings instance`: Singleton instance for the class.
- `private Language currentLanguage`: The current language setting.
- `private static ResourceBundle resourceBundle`: Resource bundle for localization.

Methods:

- `static LanguageSettings getInstance()`: Returns the singleton instance.
- `static void setCurrentLanguage(String languageCode)`: Sets the current language.
- `String getString(String key)`: Gets a localized string for a given key.

---

5. Logs

`LogRecord`

Description: Represents a single log entry in the system.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int logId`: Unique identifier for the log.
- `private int userId`: ID of the user who created the log.
- `private String text`: Text of the log entry.
- `private String time`: Timestamp for the log.

Methods:

- `LogRecord(int userId, String text, String time)`: Constructor to initialize a log entry.
- `int getLogId()`: Returns the log ID.
- `String getText()`: Returns the log text.
- `String getTime()`: Returns the log timestamp.

`LogsSettings`

Description: Handles log-related settings and operations.

Attributes:

- `private static Vector<LogRecord> logs`: List of all log records.

Methods:

- `LogsSettings()`: Default constructor.
- `static void addLogRecord(LogRecord logRecord)`: Adds a new log record.
- `static Vector<LogRecord> getLogs()`: Returns all logs.

---

6. Research Part

ResearchPaper

Description: Represents a research paper.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int paperID`: Unique identifier for the paper.
- `private String title`: Title of the paper.
- `private String authors`: Authors of the paper.
- `private String journal`: Journal where the paper was published.
- `private int pagesNumber`: Number of pages in the paper.
- `private String publicationDate`: Publication date of the paper.
- `private String doi`: DOI of the paper.
- `private int citationsNumber`: Number of citations for the paper.

Methods:

- `ResearchPaper(String title, String authors, String journal, int pagesNumber, String publicationDate, String doi)`: Constructor to initialize a research paper.
- `int getPaperID()`: Returns the paper ID.
- `String getTitle()`: Returns the title.
- `String getAuthors()`: Returns the authors.
- `String getJournal()`: Returns the journal.
- `String getPublicationDate()`: Returns the publication date.

ResearchProject

Description: Represents a research project.

Attributes:

- `private static final long serialVersionUID`: Serial version for the class.
- `private int projectID`: Unique identifier for the project.
- `private String topic`: Topic of the project.
- `private Vector<Integer> publishedPapers`: List of published paper IDs.
- `private Vector<Integer> participants`: List of participant IDs.

Methods:

- `ResearchProject()`: Default constructor.

- `ResearchProject(String topic)`: Constructor to initialize a research project.
- `int getProjectID()`: Returns the project ID.
- `String getTopic()`: Returns the project topic.

Project Management