Student Time Management Assistant

- *StudyMaster (can be changed)*

- *Name : **Vedika Jain***
- *Course : Data Structure and Object Oriented-Programming*

Scenario

- In a busy academic life, students often struggle to balance assignments, exams, personal goals, and leisure. The **Student Time Management Assistant** helps students plan, track, and prioritize their daily academic tasks effectively. This desktop-based Java application allows users to create, manage, and organize tasks, set reminders, and track progress.

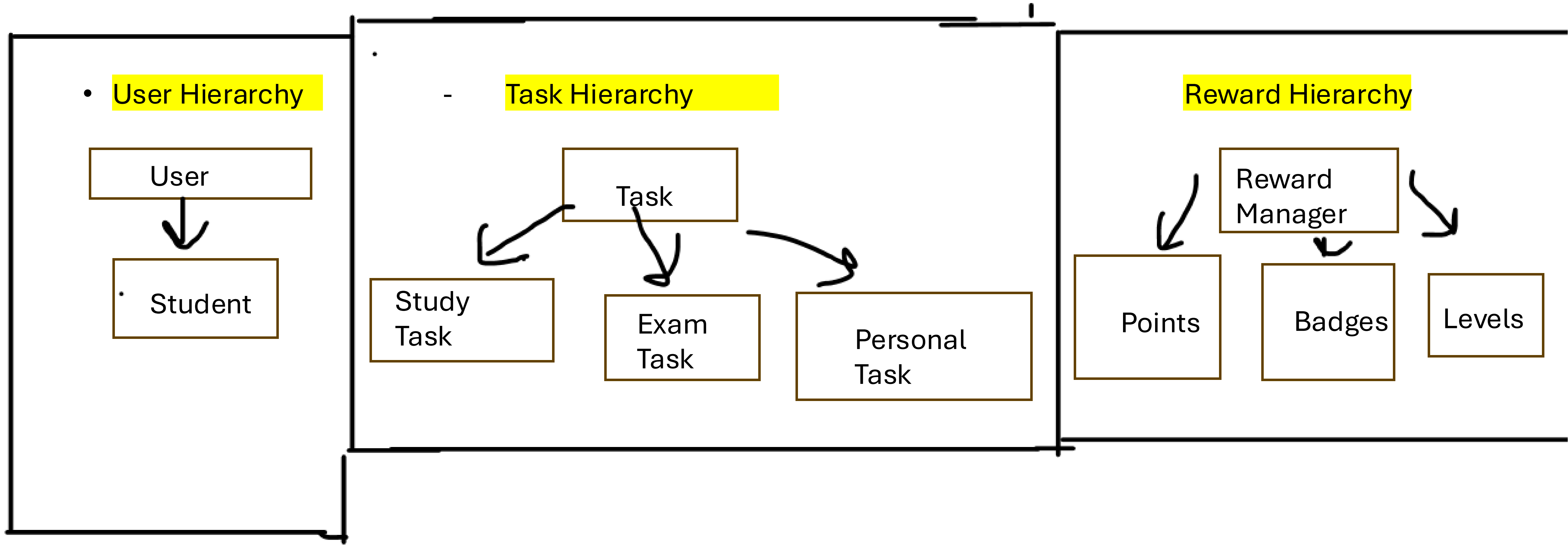
Expected Output

- Students can use the application to organize their weekly schedule, mark tasks as completed, and view their progress, receive reminders for upcoming deadlines. The app responds to task completion with motivational rewards, enhancing consistency. Students can also export their schedules.

Design Paradigm – Functionalities

- **Create and manage tasks** (title, description, due date).
- **Categorize tasks** (study, personal, leisure, etc.).
- **View tasks by day/week/month.**
- **Add, delete, edit, or mark tasks as completed**
- **Filter tasks by category** (Study, Personal, Exam, etc.)
- **Organize tasks by date**
- **Receive reminders for upcoming deadlines.**
- **Search Task by keyword or date**
- **Export the schedule to a file**
- **Prioritize tasks using different criteria.**
- **Mark tasks as completed.**
- **View productivity stats** (e.g., tasks done this week).
- **Save/load tasks from local files.**
- **User profile with preferences** (notification settings, theme).
- **Earn rewards (badges, points, levels) to stay motivated**

Hierarchies



Interfaces

1. Notifiable -> standardize how notifications will be sent -> implemented by notification class
2. Rewardable -> unified logic (criteria) for assigning rewards -> implemented by task and its subclasses
3. Schedulable -> Help Schedule Manager organize all tasks uniformly -> implemented by task and its subclasses
4. Storable -> simplifies saving / loading data --> implemented by task, reward classes
5. 5. Trackable -> support progress tracking features -> implemented by task, reward classes

Methods that apply Run-time Polymorphism

Methods	Class / Interface
SendNotification()	Notification interface

ToString()
GetType()
	Task and its subclasses

FilterTaskByCategory()	TaskManager and its subclassses
GetProgress()	Task and its subclasses
GetPriorityLevel()	Task and its subclasses

Text I/o

- Used in the Main class that starts the program or any other that interacts with the user
- **Purpose**: To read input from the user and show messages in the console.
  Example: asking the user to enter a new task or showing their weekly schedule.

• **For saving/loading data** Reading tasks from a saved file.

• Writing the task list to a file when exporting a schedule.

Comparator and Comparable

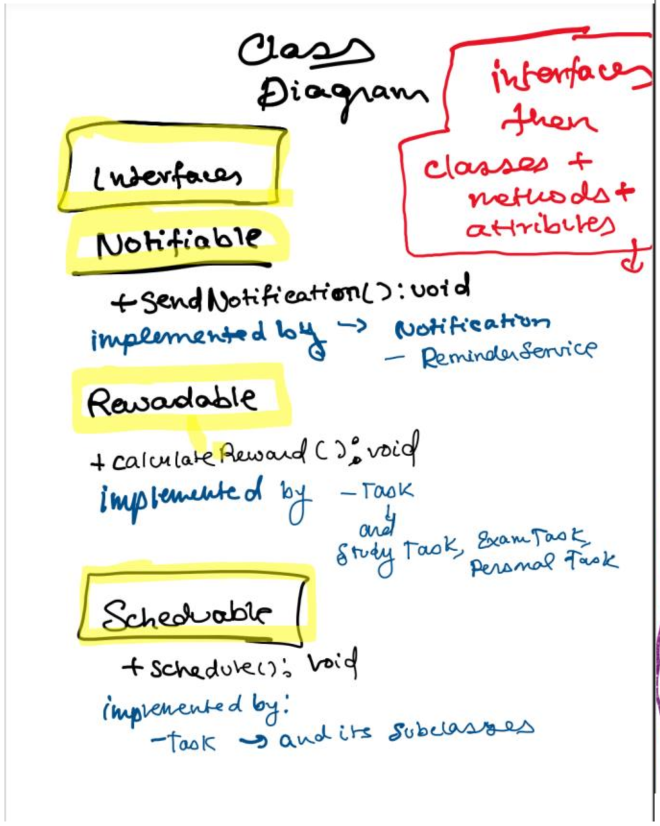
- **Comparable**
- In the Task Class , -> To sort tasks by due date by default (earlier tasks come first).
- in subclasses like `ExamTask` to sort by exam date or importance.
- **Comparator**
- **Used when**: the user want to sort tasks in **other ways**, like:
  - By **category** (Study, Personal, etc.)
  - By **title** or **priority**
- Used in -> In the class that manages tasks (like `TaskManager` or `ScheduleManager`), when sorting tasks in different ways.

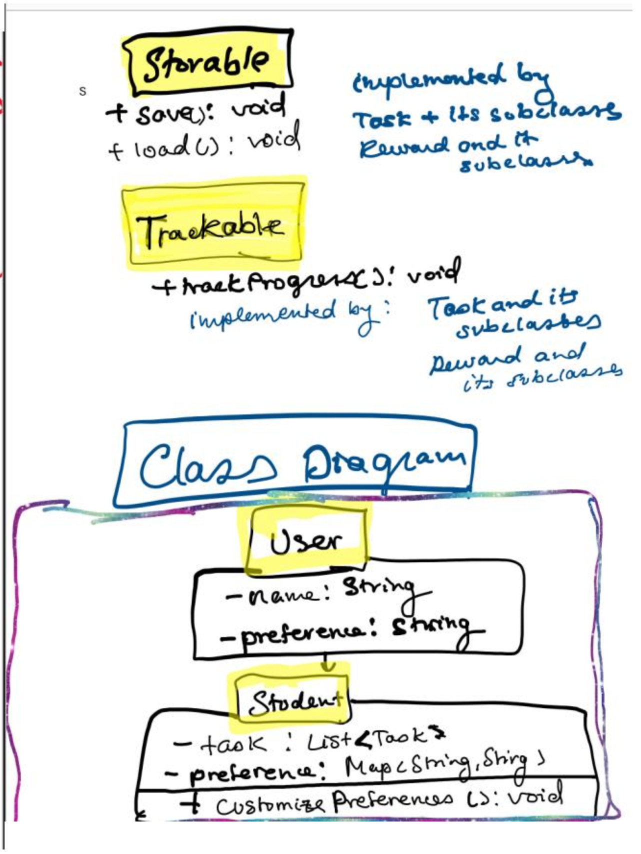
More use of Comparator

- Sort tasks by completion status (ex. incomplete first). -> in task class
- Sort rewards by type (Points before Badges, etc.). -> in reward class

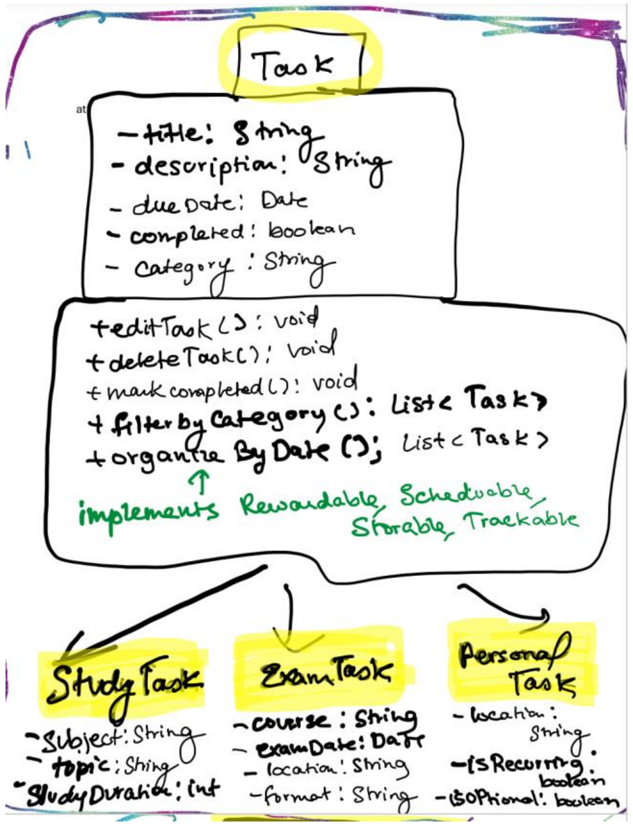
Class Diagram

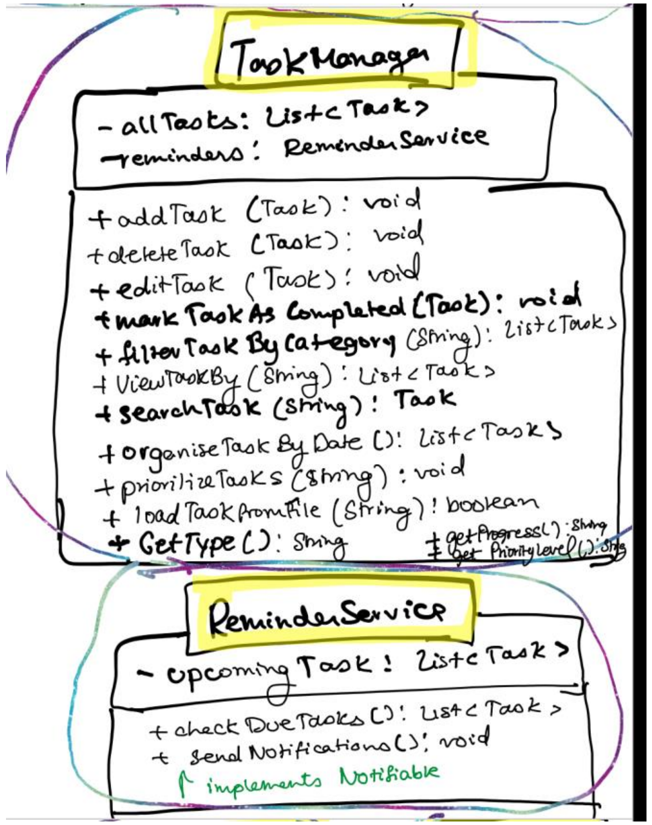
- Class Diagram changed copy

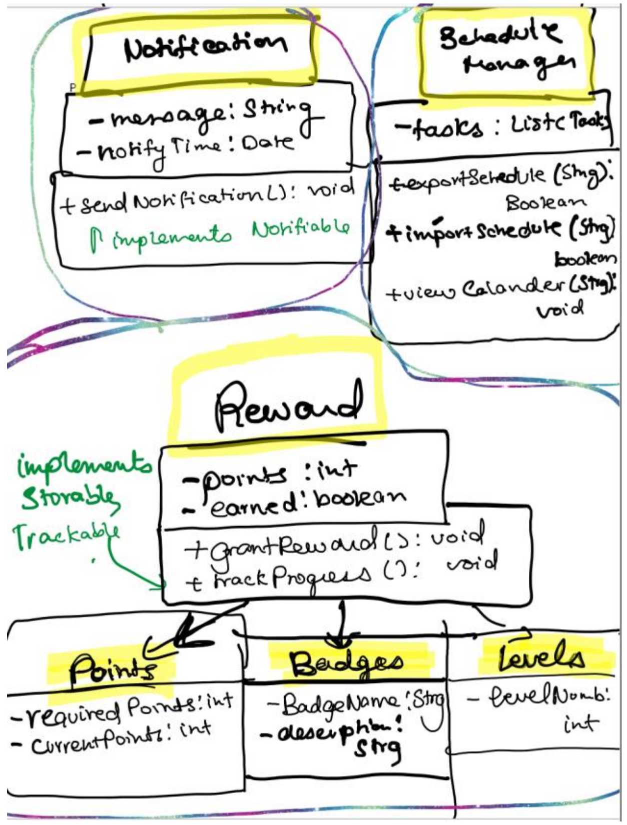

Class Diagram
interfaces then classes + methods + attributes
Interfaces
Notifiable
+sendNotification(): void
implemented by → Notification
– ReminderService
Rewadable
+ calculateReward (): void
implemented by – Task
and
Study Task, ExamTask, Personal Task
Schedvable
+ schedule(): void
implemented by:
–Task → and its Subclasses



Class Diagram




TaskManager
- allTasks: List<Task>
- reminders: ReminderService
+ addTask(Task): void
+ deleteTask(Task): void
+ editTask(Task): void
+ markTaskAsCompleted(Task): void
+ filterTaskByCategory(String): List<Task>
+ ViewTaskBy(String): List<Task>
+ searchTask(String): Task
+ organiseTaskByDate(): List<Tasks>
+ prioritizeTasks(String): void
+ loadTaskFromFile(String): boolean
+ GetType(): String
+ getProgress(): String
+ Get PriorityLevel(): String
ReminderService
- upcomingTask: List<Task>
+ checkDueTasks(): List<Task>
+ sendNotifications(): void
implements Notifiable



Part that will be implemented for Deliverable 2

- All interfaces : Notifiable, Rewardable , Schedulable, Storable, Trackable
- The User Hierarchy following the Class Diagram completed
- All other Hierarchies classes with all attributes, methods with empty bodies, javadoc, and Unit testing