Amber Hoagland
IC5

Team members: Just me!

Class Diagram utilized:

```
┌─────────────────────────────────┐
│ ⊟            User                │
├─────────────────────────────────┤
│ + Name: STR                     │
│ - Type                          │
│ - ID                            │
│ - CreatedOn                     │
│ - CreatedBy                     │
│                                 │
├─────────────────────────────────┤
│                                 │
│ - updateTaskStatus()            │
│ - viewProjectList()             │
│ - viewAssignedTasks()           │
│ - selectTask()                  │
│ - startTimeRecording()          │
│ - stopTimeRecording()           │
│ - addTimeRecordingDesc()        │
│                                 │
└─────────────────────────────────┘
```
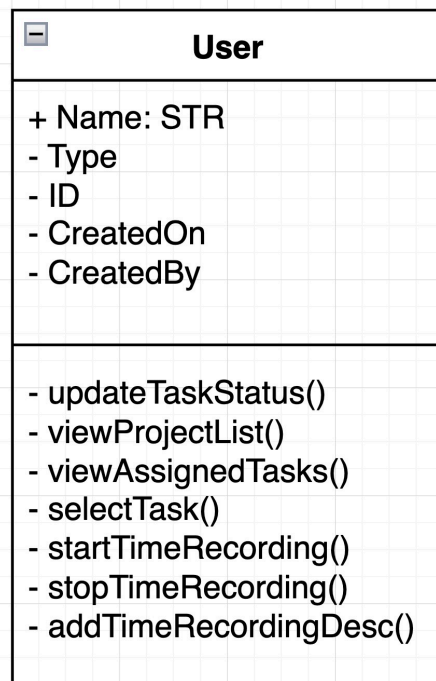
Method chosen from diagram: viewAssignedTasks()

Method Signature:
- **Name**: viewAssignedTasks
- **Params/arguments to pass**: userID - a string representing the unique user who is viewing tasks and whose tasks should be returned in a list
- **Types of values to return**: a list of task details based on the Tasks table, including values: TaskID, Description, EstimatedDuration, and ActualDuration

Example in Java:

public List<Task> viewAssignedTasks(String userID) throws UnauthorizedAccessException, IllegalStateException {
        // do all the things
                        }

Constraints that must be preserved by the objects:
- **Pre-condition**: the userID passed to the method must correspond to a valid, authenticated user
    - **Handling when violated**: system throws an UnauthorizedAccessException error, presenting an error message to the user and terminating the process

- **Post-condition**: the returned list includes only tasks assigned to the user with the specified (passed) userID; each task belongs to a project the user is assigned to
    - **Handling when violated**: the system throws an IllegalStateException error and presents an error message to the user, terminates the process
- **Invariant 1:** a task is always linked to a valid project; each task's ProjectID value must correspond to an existing Project in the Projects table
    - **Handling when violated:** an IllegalStateException error is raised, throwing an error message and terminating the process
- **Invariant 2**: the user assigned to the task must also be assigned to the corresponding project in the UserAssignments table
    - **Handling when violated**: an IllegalStateException error is raised, throwing an error message and terminating the process