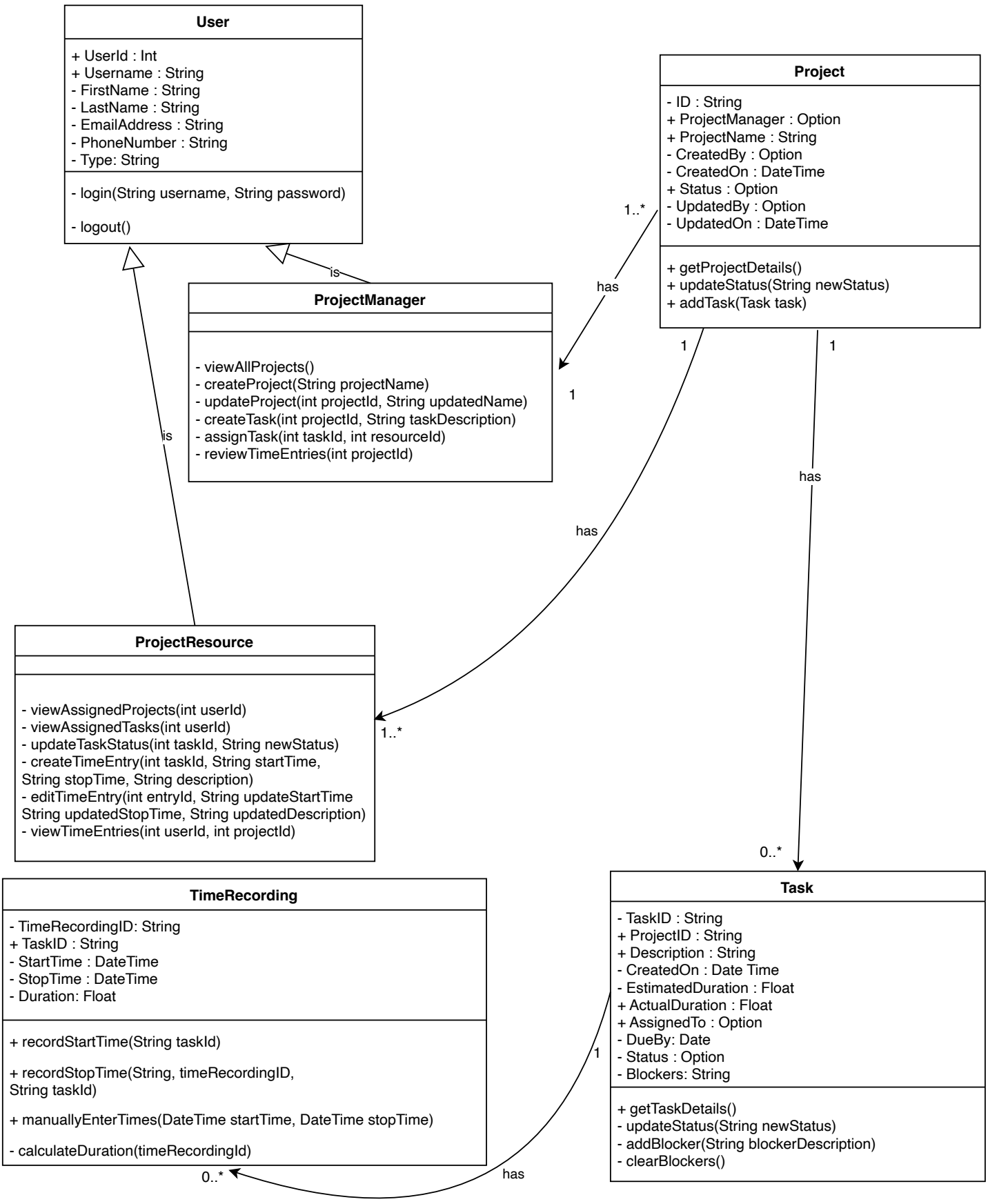
Amber Hoagland
CIS 641
Homework 5

Class Diagrams



Class Information
Note: this would not fit on the diagram above - apologies for not having it all in one place

User
+ UserId : Int + Username : String - FirstName : String - LastName : String - EmailAddress : String - PhoneNumber : String - Type: String
- Login(String username, String password) - Logout()

Class: User
Type: Abstract
Purpose: The User class is not instantiated but is used for child user classes to inherit from. A user in the system will be able to log in, log out, and have a list of common attributes as depicted in the Class diagram.

Method 1: Login
Preconditions: username and password must not be null
Method: public abstract void login(String username, String password)
Postconditions:
- The user is authenticated, and a session is started if credentials are valid.
- If credentials are invalid, an error is displayed.

Method 2: Logout
Preconditions: user must be logged in
Method: public abstract void logout()
Postconditions: user session is terminated

Object Diagram not available (abstract class - not instantiated)

ProjectManager
- viewAllProjects() - createProject(String projectName) - updateProject(int projectId, String updatedName) - createTask(int projectId, String taskDescription) - assignTask(int taskId, int resourceId) - reviewTimeEntries(int projectId)

Class: ProjectManager
Type: Child class of User
Purpose: The ProjectManager class is used for creating Project Managers who have increased access to CRUD operations for projects, users, and tasks. All attributes are inherited from the User class.

Method 1: viewAllProjects()
Preconditions: user must be logged in/authenticated
Method: private void viewAllProjects()
Postconditions: a list containing all projects is displayed

Method 2: createProject(String projectName)
Preconditions: Project Name field must not be null or empty
Method: private void createProject(String projectName)
Postconditions: a new project is created with the given name

Method 3: updateProject(int projectId, String updatedName)
Preconditions: project must exist (Project ID is not null or empty)
Method: private void updateProject(int projectId, String updatedName)
Postconditions: the project's name is updated

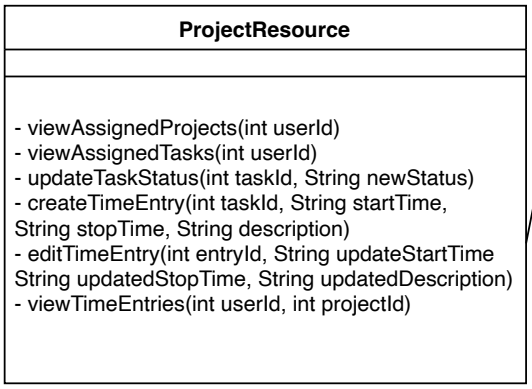
Method 4: createTask(int projectId, String taskDescription)
Preconditions: project must exist (Project ID is not null or empty)
Method: private void createTask(int projectId, String taskDescription)
Postconditions: a new task is created for the given project

Method 5: assignTask(int taskId, int resourceId)
Preconditions: both the task and the project resource must exist (IDs cannot be empty or null)
Method: private void assignTask(int taskId, int resourceId)
Postconditions: the specified task is assigned to a ProjectResource user

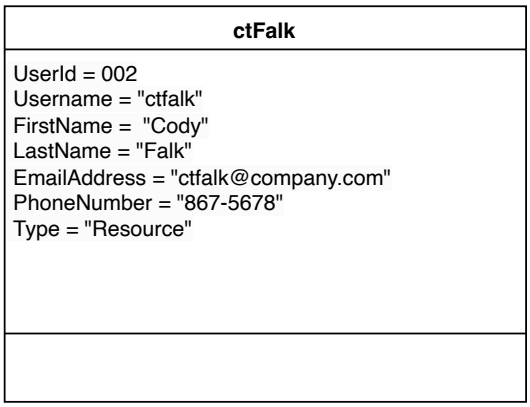
Method 6: reviewTimeEntries(int projectId)
Preconditions: the project must exist (ID cannot be null or empty)
Method: private void reviewTimeEntries(int projectId)
Postconditions: displays all time entries associated with the project

Object Diagram

amberHoagland
UserId = 001 Username = "ahoagland" FirstName = "Amber" LastName = "Hoagland" EmailAddress = "ahoagland@company.com" PhoneNumber = "867-5309" Type = "PM"



Object Diagram



Class: ProjectResource
Type: Child (inherits from User)
Purpose: The ProjectResource class is used for creating resource users to whom a Project Manager user will assign tasks. Project Resources can record time to those tasks. All attributes are inherited from the User class.

Method 1: viewAssignedProjects(int userId)
Preconditions: the user must be logged in (authenticated)
Method: private void viewAssignedProjects(int userId)
Postconditions: a list of all projects the user is assigned to is displayed

Method 2: viewAssignedTasks(int userId)
Preconditions: user must have tasks assigned
Method: private void viewAssignedTasks(int userId)
Postconditions: displays all tasks assigned to the user

Method 3: updateTaskStatus(int taskId, String newStatus)
Preconditions: the task must exist and the new status must be a valid status option
Method: private void updateTaskStatus(int taskId, String newStatus)
Postconditions: the task's status is updated

Method 4: createTimeEntry(int taskId, String startTime, String stopTime, String description)
Preconditions: the task must exist and start and stop times must be valid date times
Method: public void createTimeEntry(int taskId, String startTime, String stopTime, String description)
Postconditions: a new time entry is created for the specified task

Method 5: editTimeEntry(int entryId, String updatedStartTime, String updatedStopTime, String updatedDescription)
Preconditions: the time entry must exist (based on the entry Id), and updated values must be valid
Method: private void editTimeEntry(int entryId, String updateStartTime String updatedStopTime, String updatedDescription)
Postconditions: the specified time entry is updated

Method 6: viewTimeEntries(int userId, int projectId)
Preconditions: the project must exist, user must have recorded time entries for this project
Method: private void viewTimeEntries(int userId, int projectId)
Postconditions: all recorded time entries that this user has recorded for the given project are displayed

Project
- ID : String + ProjectManager : Option + ProjectName : String - CreatedBy : Option - CreatedOn : DateTime + Status : Option - UpdatedBy : Option - UpdatedOn : DateTime
+ getProjectDetails() + updateStatus(String newStatus) + addTask(Task task)

Class: Project
Type: Concrete
Purpose: The Project class is used to create projects. Each project can have a Project Manager, project resources, and tasks assigned to it.

Method 1: getProjectDetails()
Preconditions: the project must exist
Method: public String getProjectDetails()
Postconditions: a string is returned containing all populated attributes for the given project

Method 2: updateStatus(String newStatus)
Preconditions: new status must be a valid status option
Method: public void updateStatus(String newStatus)
Postconditions: the project's status is updated; updatedBy is set to the userID of the user who performed the update; updatedOn attribute is set to current date and time.

Method 3: addTask(Task task)
Preconditions: the task must exist and have a valid taskId
Method: public void addTask(Task task)
Postconditions: the task is associated with the project

Object Diagram

Configurator
ID = "config2024" ProjectManager = "ahoagland" ProjectName = "Configurator 2024 Build" CreatedBy = "ahoagland" CreatedOn = 09-13-2024 00:12:01 Status = "Active" UpdatedBy = "ahoagland" UpdatedOn = 11-01-2024

Task
- TaskID : String + ProjectID : String + Description : String - CreatedOn : Date Time - EstimatedDuration : Float + ActualDuration : Float + AssignedTo : Option - DueBy: Date - Status : Option - Blockers: String
+ getTaskDetails() - updateStatus(String newStatus) - addBlocker(String blockerDescription) - clearBlockers()

Class: Task
Type: Concrete
Purpose: The Task class will create task objects so that tasks can be assigned to projects and resources; they help ensure everything necessary to the project's success is completed and house estimated and actual time information.

Method 1: getTaskDetails()
Preconditions: the task must exist and have a valid task Id
Method: public String getTaskDetails()
Postconditions: a string is returned containing all details for the task

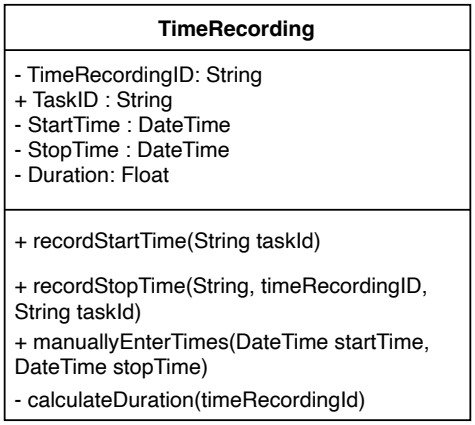
Method 2: updateStatus(String newStatus)
Preconditions: new status must be a valid status option
Method: private void updateStatus(String newStatus)
Postconditions: the status attribute of the task is updated to the specified value; if the new status is "Blocked," the Blockers field is enabled for editing

Method 3: addBlocker(String blockerDescription)
Preconditions: the task's status must be equal to "Blocked"
Method: private void addBlocker(String blockerDescription)
Postconditions: the Blockers attribute is updated with the provided description

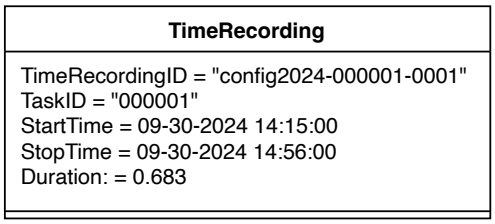
Method 4: clearBlockers()
Preconditions: the task's Blockers attribute must not be empty
Method: private void clearBlockers()
Postconditions: the Blockers attribute is set to empty; the project's status is set to "In Progress"

Object Diagram

Task1
TaskID = 000001 ProjectID = "config2024" Description = "Hold Stakeholder Kickoff Meeting" CreatedOn = 09-14-2024 00:01:00 EstimatedDuration = 1 ActualDuration = null AssignedTo = "ahoagland" DueBy = 11-30-2024 Status = "Not Started" Blockers = null



Object Diagram



Class: TimeRecording
Type: Concrete
Purpose: The TimeRecording class will create time record objects that can be associated to tasks and allow users to either record time with a start time / stop time feature or by manually entering their start and stop times for a given work unit

Method 1: recordStartTime(String taskId)
Preconditions: the startTime attribute must not have a value; the taskId must be valid
Method: public String recordStartTime(String taskId)
Postconditions: a String with the new time recording ID is returned; the startTime attribute is updated with current date and time

Method 2: recordStopTime(String, timeRecordingId, String taskId)
Preconditions: the startTime attribute must not be empty or null; the stopTime attribute must not have a value
Method: public void recordStopTime(String, timeRecordingID, String taskId)
Postconditions: the status attribute of the task is updated to the specified value; if the new status is "Blocked," the Blockers field is enabled for editing

Method 3: manuallyEnterTimes(DateTime startTime, DateTime stopTime)
Preconditions: both startTime and stopTime must not be null; stopTime must occur after startTime
Method: public void manuallyEnterTimes(DateTime, startTime, DateTime stopTime)
Postconditions: the startTime and stopTime attributes are updated; the calculateDuration() method is called to automatically update the Duration attribute

Method 4: calculateDuration(timeRecordingId)
Preconditions: both startTime and stopTime attributes must have values
Method: private void calculateDuration(timeRecordingId)
Postconditions: the duration attribute is updated with the calculated difference between startTime and stopTime