| Requirement List |
| --- |
|  |
| * RequirementList() * addRequirement(requirement : Requirement) : void * removeRequirement(index : int) : void * getRequirement(ID : int) : Requirement * getRequirement(index : int) : Requirement * getNumberOfRequirements() : int * toString() : String |

| Requirement |
| --- |
| * reqID : String * description : String |
| * Requirement(ID : int, estimatedTime : MyDateTime, description : String, responsibleTeamMember : TeamMember, totalHoursSpent : MyDateTime, deadline : MyDateTime, status : Status, tasks : TaskList) * updateStatus() : void * addTask(task : Task, reqID : String) : void * removeTask(task : Task) : void * removeTask(taskID : String) : void * getTask(taskID : String) : Task * getTask(index : int) : Task * updateTotalHoursSpent(totalHoursSpent : MyDateTime) * toString() : String |

| Status |
| --- |
| * ENDED\_STATUS : String = ”Ended” * STARTED\_STATUS : String = ”Started” * NOT\_STARTED\_STATUS : String = ”Not started” * IN\_PROGRESS\_STATUS : String = ”In progress” * APPROVED\_STATUS : String = ”Approved” * REJECTED\_STATUS : String = ”Rejected” * status : String * isRequirement : boolean |
| * Status(isRequirement : boolean) * updateStatus(status : String) * getStatus(status : String) * equals(obj:Object) : boolean * toString() : String |

| Priority |
| --- |
| * PRIORITIES : ArrayList<String> = {“Low Priority”,”Medium Priority”,”High Priority”,”Critical Priority”} * priority : int |
| * Priority(priority : int) * setPriority(priority : int) : void * getPriority() : String * equals(o : Object) : boolean * toString() : String |

| TeamMember |
| --- |
| * id : String |
| * TeamMember (member : Person, id : String, role : String) (Specify use of global static final variable from Role class Role.POSSIBLE\_ROLES[index]???) * setID (id : String) : void * setRole(role : String) : void * getRole () : Role * getID() : String * getName() : String * toString () : String |

| PhoneNumber |
| --- |
| * number : String * countryCode : int |
| * PhoneNumber(countryCode : int, number : String) * setNumber(countryCode : int, number : String) : void * toString() : String * equals() : boolean |

| ContactInformation |
| --- |
|  |
| * ContactInformation(phoneNumber : PhoneNumber, address : Address, email : Email) * ContactInformation(phoneNumber : PhoneNumber, address : Address) * ContactInformation(phoneNumber : PhoneNumber, email : Email) * ContactInformation(phoneNumber : PhoneNumber) * setPhoneNumber(phoneNumber : PhoneNumber) : void * setEmail(email : Email) : void * setAddress(address : Address) : void * getPhoneNumber() : PhoneNumber * getEmail() : Email * getAddress() : Address * toString() : String |

| Task |
| --- |
| * requirementID : String * taskID : String * title : String |
| * Task (requirementID : String, taskID : String, title : String, estimatedTime : MyDateTime, deadline : MyDateTime, responsibleTeamMember : TeamMember, status : Status, hoursSpent : MyDateTime)(Specify use of global static final variable from Status class Status.ENDED\_STATUS etc.???) * updateStatus(status : String) : void(Specify use of global static final variable from Status class Status.ENDED\_STATUS etc.???) * updateHoursSpent(seconds : int) : void * updateResponsibleTeamMember(responsibleTeamMember : TeamMember) : void * getTaskID() : String * getTitle() : String * getRequirementID() : String * getDeadline() : MyDateTime * getEstimatedTime() : MyDateTime * getHoursSpent() : MyDateTime * getResponsibleTeamMemer() : TeamMember * toString() : String |

| TaskList |
| --- |
|  |
| * TaskList() * addTask(task : Task) : void * getTask(index : int) : Task * removeTask(task : Task) : void * removeTask(index : int) : void * numberOffTask() : int * toString() : String |

| Role |
| --- |
| * POSSIBLE\_ROLES : Arraylist<String> = {“Product Creator”,”Product Owner”,”Team Member”,”Scrum Master”} * role : String |
| * Role(role : string) * setRole (role : String) * getRole () : String * toString() : String |

| ProjectList |
| --- |
|  |
| * ProjectList() * addProject(project : Project) : void * getProject(index : int) : Project * removeProject(project : Project) : void * removeProject(index : int) : void * numberOfProjects() : int * equals(o : Object) : boolean * toString() : String |

| Project |
| --- |
| * progress : double |
| * Project(teamMembers : TeamMemberList, requirements : RequirementList, customer : Customer, deadline : MyDateTime) * updateProgress() : void * getProgress() : double * getDeadline() : MyDateTime * getTotalHoursSpent() : MyDateTime * getTeamMembers() : TeamMemberList * getRequirements() : RequirementList * toString() : String |

| ManagementSystem |
| --- |
|  |
| * ManagementSystem () |

| ManagementSystemFile |
| --- |
|  |
| * ManagementSystemFile (fileName : String) |

| ManagementSystemFilePersistence |
| --- |
|  |
| * loadAll () : ManagementSystemModel * saveAll (model : ManagementSystemModel) : void * save (requirement : Requirement) : void * save (task : Task) : void * save (project : Project) : void * replace () |

| <<interface>>  ManagementSystemModel |
| --- |
| * addProject (project : Project) : void * addRequierment (requirement : Requirement) : void * addTask (task : Task) : void * getProject () : Project * getRequirement () : Requirement * getTask () : Task * getAllProjects () : ProjectList * getAllRequirements () : RequirementList * getAllTasks () : TaskList * removeProject (project : Project) : void * removeRequirement (requirement : Requirement, project : Project) : void * removeTask (task : Task, project : Project) : void * updateRequirementStatus (requirementStatus : Status) : void * updateTaskStatus (taskStatus : Status) : void * updateProjectStatus (projectStatus : Status) : void * changeTaskPriority(priority : Priority) : void * changeRequirementPriority(priority : Priority) : void * addTeamMember(member : TeamMember) : void * removeTeamMember(ID : String) : void * assignRole(role : Role, member : TeamMember) : void * getResponsibleTeamMember(requirement : Requirement) : Team member * getResponsibleTeamMember(task : Task) : TeamMember * setDeadline(date : MyDateTime) : void * getRequirementStatus(requirement : Requirement) : Status * getTaskStatus(task : Task) : Status * getProjectStatus(project : Project) : Status * getTotalTimeSpent(project : Project) : MyDateTime * setEstimatedTime(time : MyDateTime, requirement : Requirement) : void * setEstimatedTime(time : MyDateTime, task : Task) : void * getEstimatedTime(requirement : Requirement) : MyDateTime * getEstimatedTime(task : Task) : MyDateTime |
|  |