Bilkent University

Department of Computer Engineering

Object-Oriented Software Engineering Term Project

TrackIn: Intern Tracking System

Design Report

**Group 3G:**

Muammer TAN, Ali Semi YENİMOL, I. Kayacan KAYA, Eniselda TUSKU

Contents

[1 Introduction 1](#_Toc500188866)

[1.1 Purpose of The System 1](#_Toc500188867)

[1.2 Design Goals 1](#_Toc500188868)

[2 Software Architecture 3](#_Toc500188869)

[2.1 Subsystem decomposition 3](#_Toc500188870)

[2.2 Hardware/software mapping: 3](#_Toc500188871)

[2.3 Persistent data management 4](#_Toc500188872)

[2.4 Access control and security 5](#_Toc500188873)

[2.5 Boundary conditions 6](#_Toc500188874)

[3 Subsystem Services 7](#_Toc500188875)

[4 Low-level Design 9](#_Toc500188876)

[4.1 Object design trade-offs 9](#_Toc500188877)

[4.2 Final Object Design 10](#_Toc500188878)

[4.2.1 Internship Subsystem Classes 10](#_Toc500188879)

[4.2.2 Controller Subsystem Classes 10](#_Toc500188880)

[4.2.3 Task Management Subsystem Classes 11](#_Toc500188881)

[4.3 Packages 11](#_Toc500188882)

[4.4 Class Interfaces 11](#_Toc500188883)

[4.5 Navigational Paths and XHTML files for the User Interface Design 18](#_Toc500188884)

[4.5.1 Sign-up Navigation Path 18](#_Toc500188885)

[4.5.2 Sign-in Navigation Path 18](#_Toc500188886)

[4.5.3 Company Nagivation Path: 18](#_Toc500188887)

[4.5.4 Intern Navigation Path: 19](#_Toc500188888)

[4.5.5 Supervisor Navigation Path 20](#_Toc500188889)

[4.5.6 XHTML Files: 20](#_Toc500188890)

[5 Glossary & References 23](#_Toc500188891)

Design Report

TrackIn: Intern Tracking System

# Introduction

TrackIn is a web based application that provides companies, interns and supervisors with an easier way to manage the internship procedure and facilitate the communication between them.

## Purpose of The System

The system aims to achieve the following features:

Task Management: Managing and tracking the tasks given to interns, see the status and workload, who supervises to which intern, announcing general events to interns.

Privacy: Providing the privacy for companies as they might not want to share all details and information with the interns.

Document Conversation: Keeping the files within the system to save files, documents integrated to a internship and eliminating possible document losses.

Flexibility: Providing the flexibility to companies each has its own requirements from interns and operations to do.

## Design Goals

**Usability:**

One of the most important design goals that we aim is to develop a user friendly program that will be easily used by the customer. Weather you enter our program as an intern, supervisor or a company, all the functionalities and the relation with other users will be clearly stated in the interface so that the user will not have any difficulties in understanding and putting his functions to use. Since this system intends to facilitate the internship tracking process the program will be as user friendly as possible in order to contribute to this purpose.

**Efficiency:**

We aim to make our program as efficient as possible so the user will not have any difficulties or delays while using it. We will try to guarantee a response time that will not exceed a certain threshold value. In this way the communication between the users, regarding the tasks or other elements will be fast and will not cause the user any concerns.

**Extensibility:**

Since we are developing a platform for different companies to keep track of their internship processes, it may later be extended and further developed so it can cover more functionalities and be adopted without any problem to certain company profiles. In order for this to be achieved we indeed to build a system that can be further modified and extended without causing any problems to the current one. This will involve building a system which is clear and very well structured.

**Reliability:**

We intend to design a system that will be reliable and stable and will not crush due to irrelevant inputs or actions from the user. In order for this to be achieved frequent test will be conducted to every part so that stability can be assured.

**Well defined interface:**

In order for the user to use the program as efficiently as possible and for him not to have any inconveniences, we aim to design a well defined interface which will not only make things easier for the user but for the developers as well. A well designed interface would create facilities.

# Software Architecture

## Subsystem decomposition

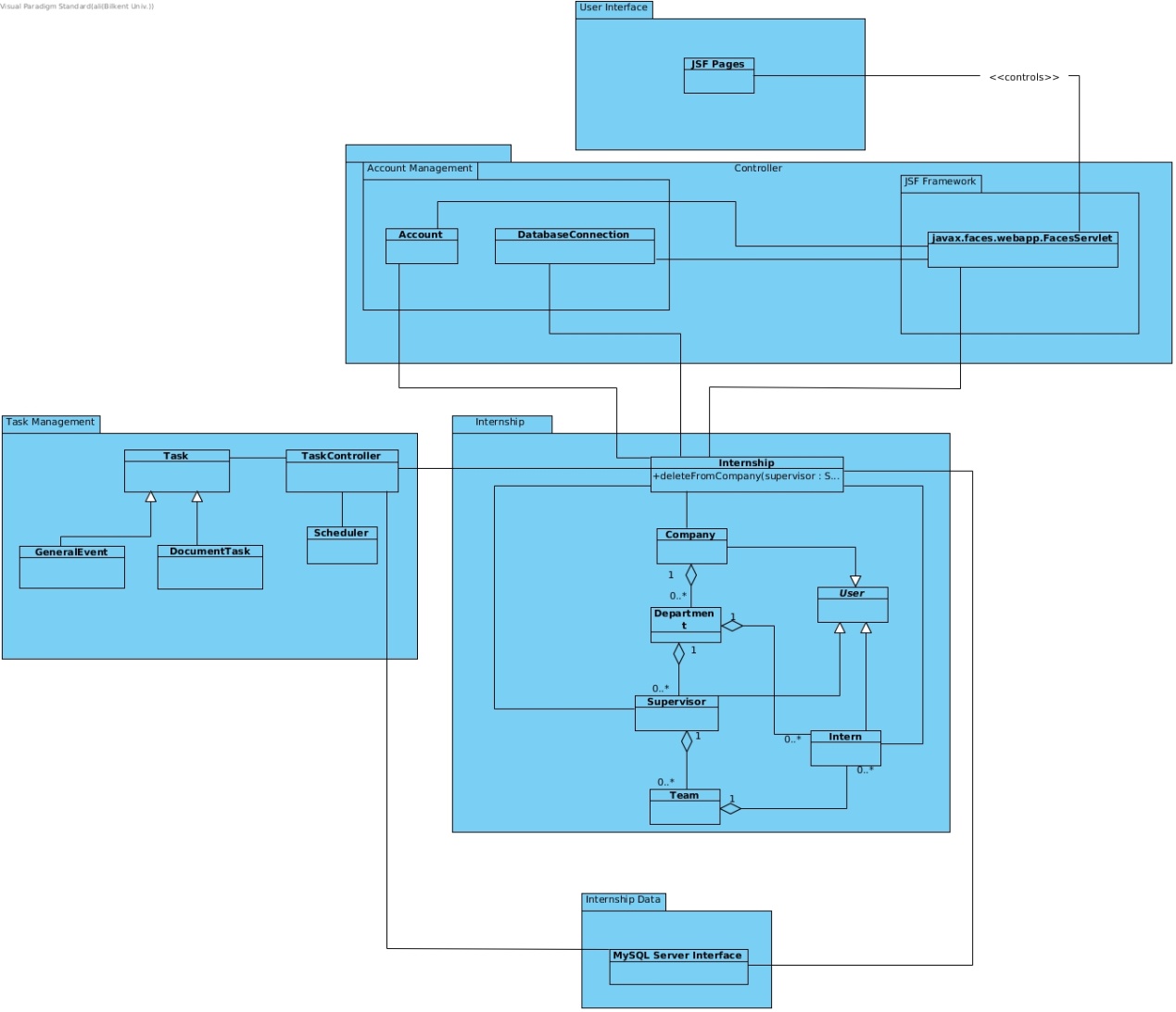


Figure TrackIn Subsystem Decomposition Diagram

## Hardware/software mapping:

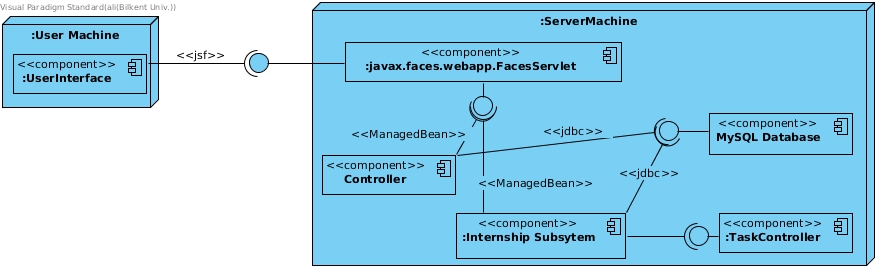


Figure TrackIn Deployment Diagram

As TrackIn is proposed to be a web application, hardware/software mapping is important for the design. For efficiency goals, the system has Client/Server architecture as it can be seen from the Deployment Diagram above. The Server Virtual Machine must support the Java environment and as we decided to use Glassfish for server-side implementation, the server also should support this library. For the storage systems, we are planning to use MySQL to construct the database. Hence the server must also have a support for MySQL. On the client side, a modern web browser is needed where PrimeFace Framework(for user-Interface) is compatible with. In addition, the user machine must obviously have a mouse and keyboard to input to the system.

## Persistent data management

Data Management is one of the most important things we will deal with since we are going to implement a user-based web application. We are going to use MySQL to store personal information, username, password and user-related data of Intern, Supervisor and Company user types. Also we are going to keep events, tasks and other Internship related data as well. So, losing or manipulating all those data would be very easy if we would keep them in .txt files. Because of the concern of security and convenience, we are going to use MySQL database.

Here is the ER Diagram for the data:

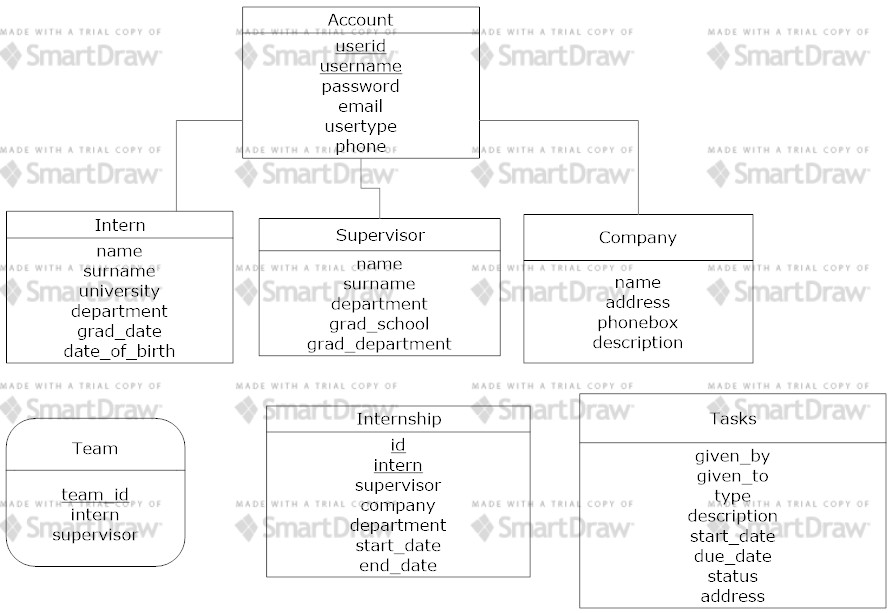


Figure TrackIn ER Diagram

Following tables are created in the database:

* **Account:** This table will keep the common data of all user types. "userid" and "username" will be unique for all users, thus they will be primary key attributes.
* **Intern:** Intern table will keep the data of user type "intern". This table will be sub-table of "account" table, so "userid" and "username" are the foreign keys in this table.
* **Supervisor:** Supervisor table is going to store the data of supervisor user type. Same as "intern"; "userid" and "username" attributes of "account" table are foreign keys in "supervisor" table.
* **Company:** This is the table which keeps data of company type of users. This table is a sub-table of "account" table as well. To maintain uniqueness, "userid" and "username" attributes will be foreign keys in this table too.
* **Team:** This table is created to keep track of teams consist of interns with one supervisor. "team\_id" attribute is the primary key of this table.
* **Tasks:** Tasks table keeps data of tasks related to job, internship applications and general events. This table is a weak entity table, since there is no unique attribute in this table.

## Access control and security

TrackIn has 3 different types of users: Intern, Company, Supervisor. Each user has different functionalities within the system and should not act as one another. The central actor of the system is company, which is able to manage the other actors and their roles in the system and acts as a host whereas Intern and Supervisor actors are users in terms of hierarchical user pattern. That’s why, Interns and Supervisors must not interfere with company specific functionalities and each others’ functionalities. Also, the companies must not access other companies’ control rights. The restrictions for such access control mechanisms is provided with Controller classes inside subsystems.

Another concern will be security for user accounts. For that reason, we need to use proper cryptographic protocols such as hashing and authentication methods to ensure the security for possible cyber-attacks.

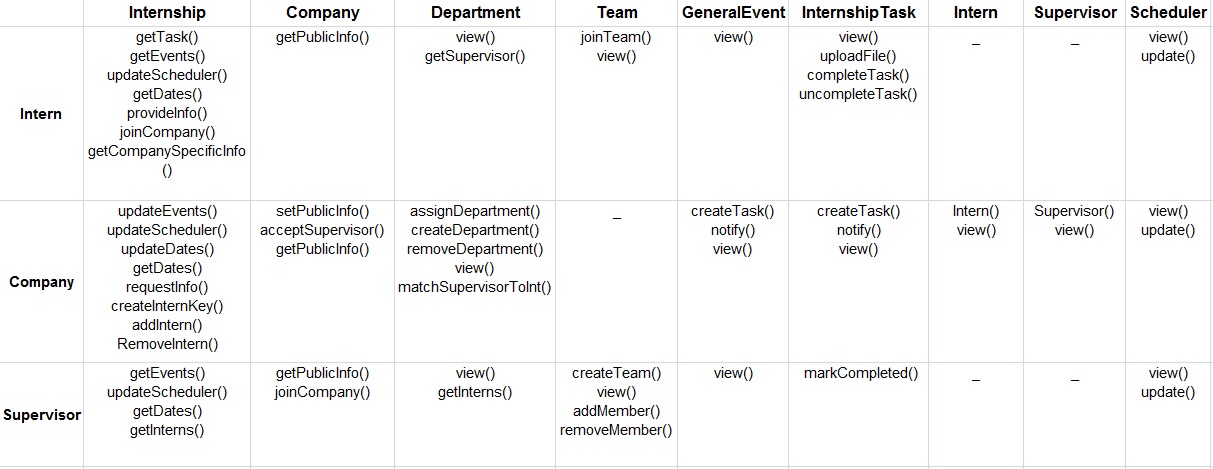
**Access Matrix:**

Figure TrackIn Acces Matrix by User Types

## Boundary conditions

* **Initialization:**

The system is initialized when the user launches the program …. When it is firstly initialized the initialization files provide the default state. Firstly the user will be faced with the logging in process.

* **Termination:**

The program terminates when the user chooses Log Out of their account.

* **Failure:**

In case of a failure, since the information is saved in a database it will not be lost. However if there is information procession which has not been saved yet during the time when the failure occurs, the system experiences loss of information.

# Subsystem Services

**TaskManagement**

This subsystem is responsible for Task related controls and acts as a inter-mediator between Database related to the storage of Tasks and Events. For easy-development purposes, Events are also designed to be a subclass of Task class within the system. When a company or supervisor adds a task to an intern, this subsystem will handle the types of the task and update the database and then notify the Internship Subsystem.

**Internship**

Internship Subsystem has the functionalities for the core mechanics of the internship procedure like relationship between interns, companies, supervisor, and teams. It is a central subsystem which communicates with TaskManagement and AccountManagement Subsystems. It is also responsible for UserInterface communication within the other subsystems.

**Controller**

The role of the AccountManagement class is to control various types of users and provide the security of accounts. It is responsible for registration and log-in mechanisms and after a user enters to the system, it will active the Internship Subsystem according to the user type. Additionally, it is responsible for controlling the storage of the accounts.

**Internship Data**

Internship data subsystem will be the connection between MySQL and our Java codes. We will keep here functions that send MySQL queries to the DB Service. All the queries’ and functions’ main task is to keep the data of an internship such as lists, names, surnames, events, tasks, dates, calendar details etc.

**User Interface:**

User interface subsystem is the subsystem responsible for all the graphical interface that occur during the program. It creates relations and exchanges data with all the other subsystem and manages the user interface of the program. It makes possible the user interaction with the system by taking input, returning data, displaying all the elements that the user needs and arranging the program according to their preferences.

* Registering:

The first view that appears when the program is initialized is the register Frame. Its primary aim is to provide user login according to his profile. Different panels are displayed if the user is signing up for the first time in which case he has to provide several information in order for his account to be activated, or if he is just signing up with an already existing account. This subsystem connects with the Account Management Subsystem in order for the corresponding profile to be displayed.

* Profiles:

After signing up according to the data collected from the user, the corresponding profile is displayed. This is made possible through the Account Management Subsystem which connects with the controller of the Internship subsystem and updates the user interface. Each profile is represented from a different frame and it contains the display of every element that composes that certain profile. Each user has access to different features so they include in their frames several panels. For example the supervisor also contains the “team” panel in which he can create and manage a team of interns. The company as an additional feature has “Matching Panel” where it makes possible the matching of supervisors with interns, etc.

# Low-level Design

## Object design trade-offs

**Response Time vs. Memory usage:** Memory usage is not that important in TrackIn system since it will not use the memory of the user's device. All the data will be held in database system and functionality of the program will be held in server. So, TrackIn will use only the RAM of user's device and hopefully it will not exceed the RAM of user's device. However, response time is really important for TrackIn system because it will be a web-based application and client will not want to wait when using TrackIn.

**Usability vs. Functionality:** Usability is a vital feature for our application because it is a business app and people will not want to use it if it is hard to use. Because of that we decided to use PrimeFaces Framework. But functionality is very important as well since again, it is a business app. So, this trade-off must be very balanced because these two features are very important for clients.

**Functionality vs. Difficulty of Code:** As it is mentioned above, functionality is very important for client. But it is nearly impossible for us to use all desired functional features since we, undergrad students, are not able to write that much functional and complex code. So we will do our best to make it closer to desired functionalities.

**Minimum Number of Errors vs. Functionality:** As functionality increases, the number of errors may increase because the code becomes more complicated and naturally becomes harder to handle. Since TrackIn is a web-based data storing application, a tiny error may cause huge problems such as losing, manipulating data, unwanted changes in internship process etc. Thus, minimum number of errors are more important than functionality of application.

## Final Object Design

### Internship Subsystem Classes

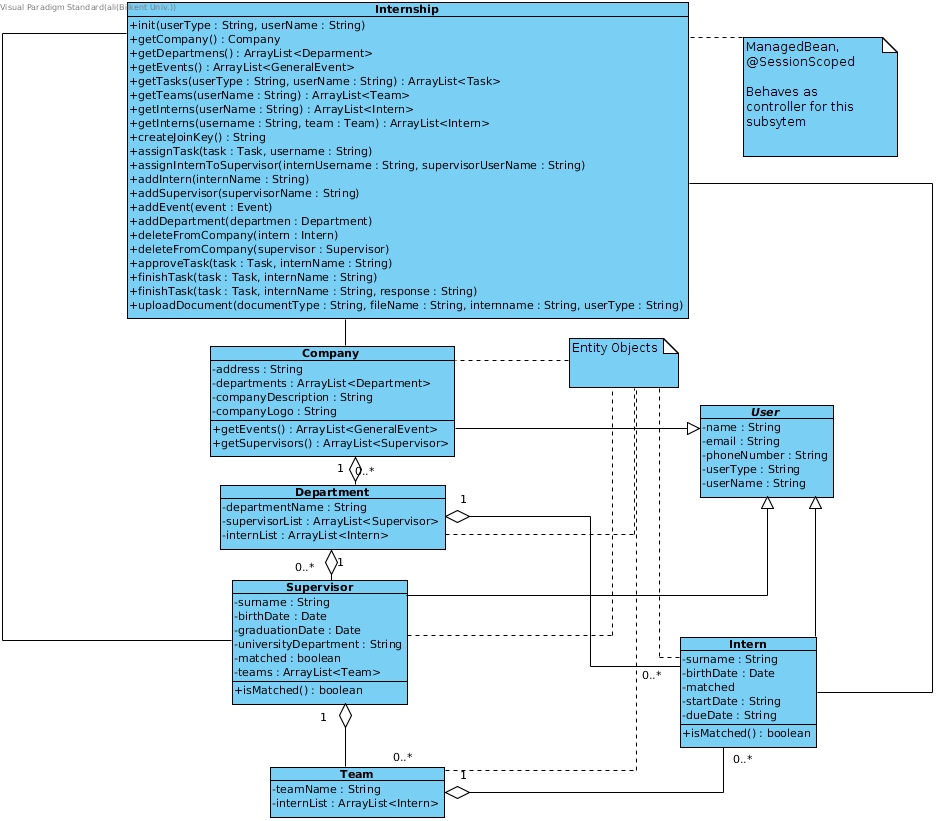
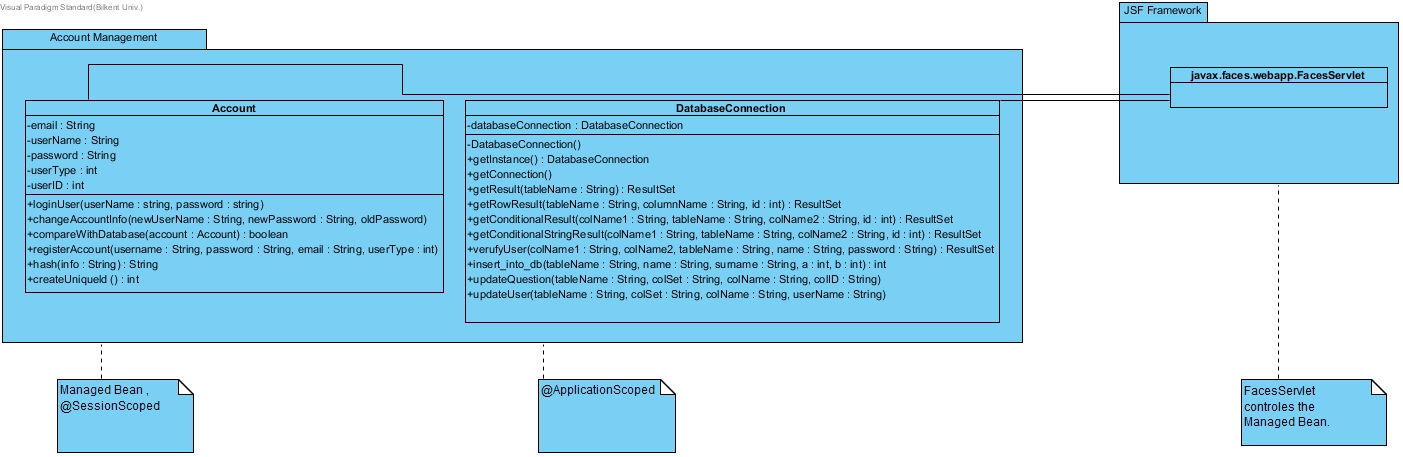


Figure Internship Subsystem Class Design Diagram

### Controller Subsystem Classes

Figure Controller Subsystem Class Design Diagram

### Task Management Subsystem Classes

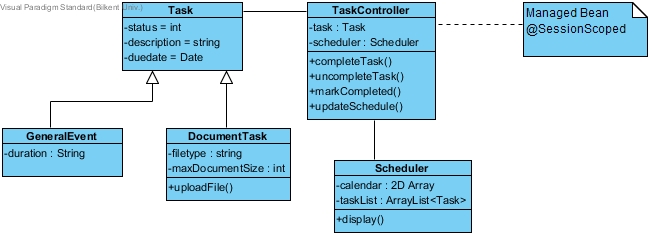


Figure Task Management Subsystem Class Design Diagram

## Packages

**4.3.1 View Package**

In this project, we are going to use JSF (Java Server Faces) Framework. So we are not going to use Java tools or packages for GUI (Graphical User Interface). Thus we are not going to have a view package.

**4.3.2 Controller Package**

Controller package's main role is to manage and the control the application. To manage and control the application, controller package uses five subsystems to manage internships, tasks, accounts, and database.

## Class Interfaces

* **Controller Subsystem Classes**

**Account**

Controller class for the communication between UserInterface Subsytem, Internship Subsystem, and Database related to account information.

***Attributes:***

--email : String – Stores the e-mail adress of the account

-userName : String – Stores the user name as hashed

-password : String – Stores the password as hashed

-userType : String – Stores the userType, can be INTERN, SUPERVISOR, or COMPANY

-userId : int – Stores the unique user id

***Methods:***

* +sendToInternController( Account)

Sends the required account information to Internship Subsystem that needs especially for the user type and user id for the operations

* +registerAccount(username : String, password : String, eMail : String, userType : String)

Creates a new account according to the given parameters, initializes the account member of the class, and activates the database(AccountData Subsystem) for storing it.

* +logInUser(userName : String, password : String)

Verifies the user information and initializes the account.

* +changeAccountInfo(newUsername : String, newPassword : String, oldPassword : String)

Make alterations on the account and activates the database(AccountData Subsystem) for storing it. newUsername is null if only password changes, newPassword is null if only username changes.

* -hash(info : String) : String

Used to hash private info like password and user name before initializing the account and storing it into the database

* -compareWithDatabase(account : Account) : boolean

Do the comparison if the user has the access to continue

* -notifyAccountStorage(account : Account) : Boolean

notifying the database for storage, called by registerAccount, logInUser, and changeAccountInfo methods

* -createUniqueId() : int

Creates a unique id for new accounts, called by registerAccount

* **Task Management Subsystem** **Classes**
* Task – The Task class is a general class that holds shared attributes for tasks. It contains all the tasks that are related with the internship such as the intern’s assigned and completed work. It has properties like:

<checked> - Will denote if the supervisor has checked the intern’s work or not.

<startDate> - To denote the starting date of the task.

* DocumentTask – Contains all the tasks that are related with different documents needed during the internship procedure. It has properties like:

<File Type> - Which specifies the type of file that the user uploads.

<Max Document Size> - Specifies the max size that the uploaded document must have.

* **Task Management Subsystem** **Methods**
* uploadFile()- Method used by intern class in order to upload different files such as internship task files, documents etc.
* completeTask() – Method use by the intern class in order to mark the task as completed. This method, after the task is marked as completed, notifies the supervisor that the intern has completed his assigned task.
* uncompleteTask()- Method use by the intern class in order to mark a certain task as uncompleted. This method is needed in case the intern marks a task as completed by mistake
* markCompleted () – Method used by the supervisor in order to mark the intern’s task as completed.
* createTask() – Method used by the supervisor in order to create a certain task for the intern.
* notify() – Method used from the company to notify information such as events to the intern and the supervisor.

**Internship Subsystem Classes**

**Internship**

***Methods:***

void init(String userType, String userName ): Initializes the internship controller according to the user type. It is needed for internal implementation of the Internship class.

Company getCompany(): Company: This method will be accessed by the company for internal alterations.

ArrayList<Department> getDepartments() : Returns the departments that belongs to the company initialized by the init method. This method is used by company user type.

ArrayList<GeneralEvent> getEvents():This method will show the general events to the intern and the supervisor so that the intern and the supervisor user will be able to see the general events given by the company. Company might also call this method for the events page.

ArrayList<Task> getTasks( String usertype, String username): This method returns the tasks related to the user. If user is an intern, only the tasks of theirs. Supervisor will see the tasks that they give the interns. Company will get the tasks that it sent directly to the interns.

ArrayList<Team> getTeams(String username): Supervisor calls this method to get their teams.

ArrayList<Intern> getInterns( String username): Company calls this method to get its interns.

ArrayList<Intern> getInterns( String username, Team team): Supervisor calls this method to get interns belonging to a team.

String createJoinKey(): This method is used by company to create unique join keys for interns and supervisors.

void assignTask(Task task, String username): This method might be called by either company or supervisor to give a task to intern. username is the name of the intern.

void assignInternToSupervisor(String internUsername, String supervisorUsername): This method is called by company for matching interns to supervisors.

void addIntern( String internName): This method adds and intern to the intern list of company, called by company.

void addSupervisor( String supervisorName): This method adds and supervisor to the supervisor list of company, called by company.

void addEvent( Event event): This method is called by company and adds an event to the eventlist.

void addDepartment( Department department): This method is called by company and adds an department to the department list.

void deleteFromCompany( Intern intern): This method should be called after the internship time has expired in order to delete it from the intern list. Deleted intern becomes a free intern and remains registered to the server.

void deleteFromCompanySupervisor( Supervisor supervisor): This method might be called by company to delete the supervisors that no longer belong to the company. Deleted supervisor becomes a free supervisor and remains registered to the server.

void approveTask( Task task, String internName): Approves the task that has been given. Called by supervisor or company.

void finishTask( Task task, String internName): Finished the checkbox task or document task that is given to the intern. Called by intern.

void finishTask( Task task, String internName, String response): Finish the task that involves a text-box response from the intern. Called by intern.

void uploadDocument( String documentType, String filename, String internName, String userType): Called by intern to upload documents that might be text files or image files. Also can be called by company to provide documents.

**Company**

***Attributes:***

companyDescription : String: Description of the company.

address : String: Address of the company

companyLogo : String: The address of the logo file of the company in the server that is kept by database.

departments : ArratList<Department>: The departments of the company which is specified by the company user.

***Methods:***

ArrayLisy<Supervisor> getSupervisors(): Returns the supervisors under the company.

getDepartments(): Returns the existing departments of the company.

**Department**

***Attributes:***

departmentName : String: The name of the department

supervisorList : ArrayList<Supervisor>: The list of all the supervisors belong to this department.

internList : ArrayList<Intern>: The list of all the inters belong to this department.

**Supervisor**

***Attributes:***

surname : String: Surname of the supervisor

birthDate : Date: Birthddate of the supervisor

graduation : String: The university of the supervisor that he/she had graduated

uniDepartmentName : String: The university department of the supervisor

teams : ArrayList <Team>: The existing teams of the supervisor

matched : boolean: The status of the supervisor whether he is matched or not

***Methods:***

boolean isMatched(): Checks if the supervisor is matched to at least one intern.

**Team**

***Attributes:***

name : String: Name of the team

internList : ArrayList <Intern>: The interns that are belong to this particular team.

**Intern**

***Attributes:***

surname : String: Surname of the intern

birthDate : Date: Surname of the intern

universityName : String: The university of the intern

startDate: Date: Start date of the internship for this intern

endDate: Date: End date of the internship for this intern

matched : boolean: The status of the intern whether he is matched or not

***Methods:***

boolean isMatched(): Checks if the intern is matched to one supervisor

**User**

Abstract class for the parent of Company, Supervisor, and Intern.

***Attributes:***

name : String: Name of the user.

email : String: E-mail of the user.

phoneNumber : String: Phone number of the user

userName : String: Username of the user

userType : String: Usertype of the user

## Navigational Paths and XHTML files for the User Interface Design

### Sign-up Navigation Path

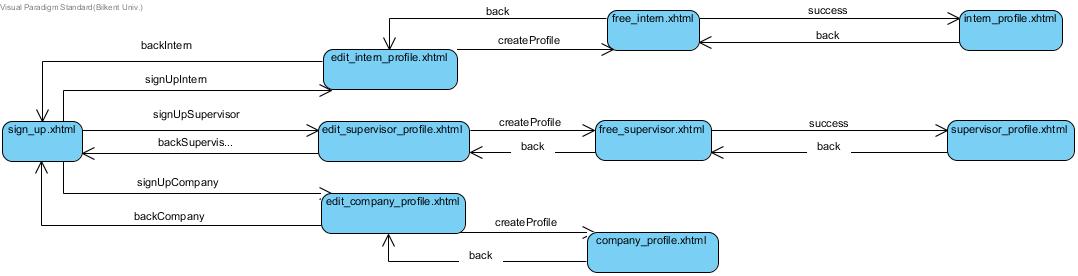


Figure Sign-up Navigation Path

### Sign-in Navigation Path

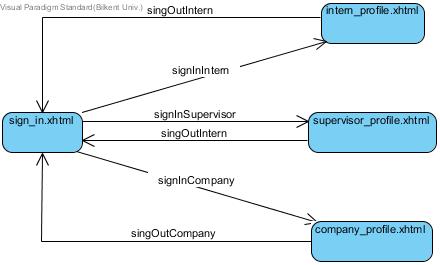


Figure Sign-in Navigation Path

### Company Nagivation Path:

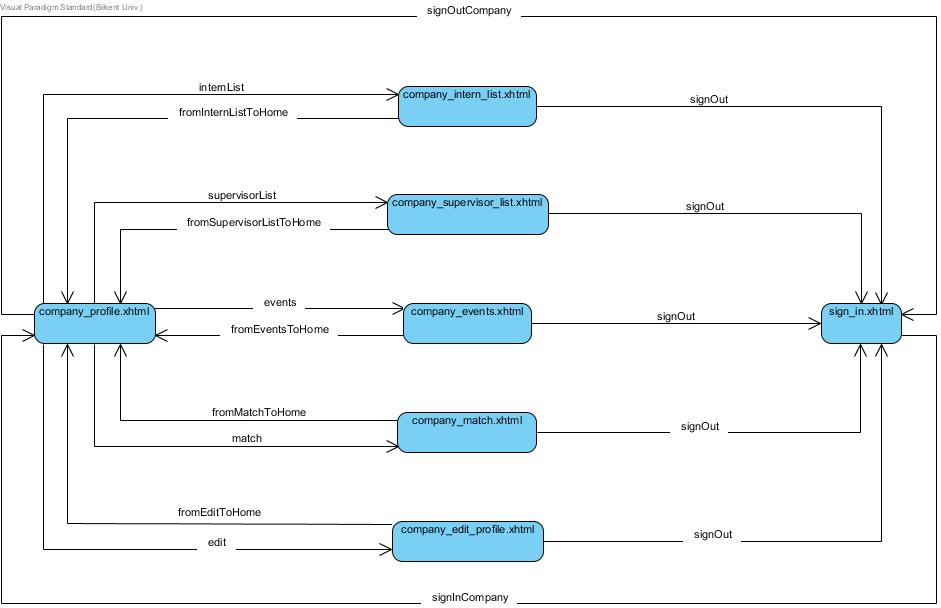


Figure Company Navigation Path

### Intern Navigation Path:

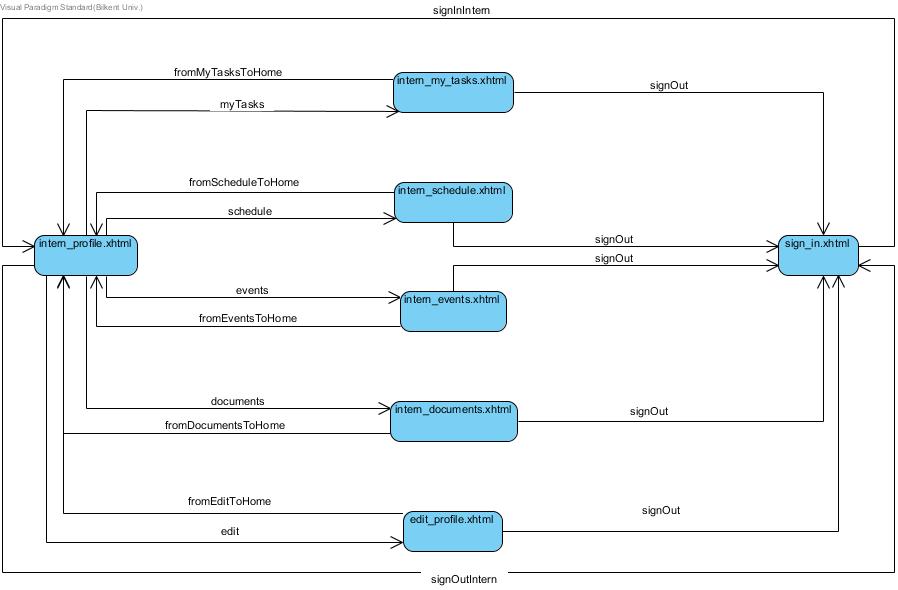


Figure Intern Navigation Path

### Supervisor Navigation Path

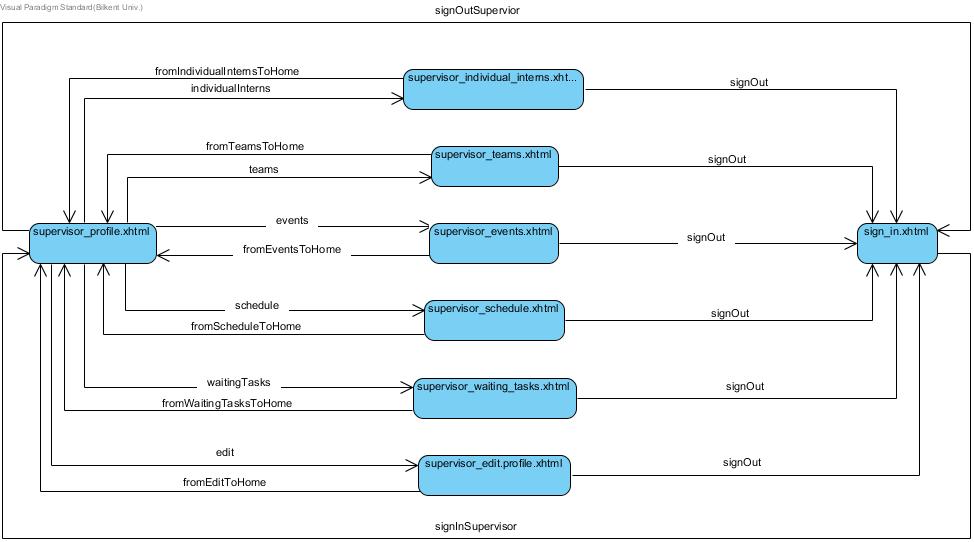


Figure Supervisor Navigation Path

### XHTML Files:

* **sign\_up.xhtml (SignUp – LogIn Page) :** this page is the initial page of the TrackIn. All users should sign up or log in when they want to use system.
* **sign\_in.xhtml:** This page is when the user enters the program after he created his profile. In this case he does not need to complete any additional information but is directed directly to his profile.
* **index.xhtml (Home Page):** this page is the main page of each user types. Same as "edit.xhtml", it shows information according to the user type.
* **free\_intern.xhtml (Free Intern Home Page):** When the user type is intern and the user is not registered to a company for internship, user will encounter with this page. User will have limited features until registered to a company for internship. The only features a free intern can use are the edit profile and join a company features. When clicked to join a company, a pop-up will show up and asks user to enter verification code sent by company to registration.
* **edit\_intern\_profile.xhtml (Initial User Information Page):** if an intern entered the system for the first time, i.e. registered to the system for the first time, or clicked edit profile button, he will encounter with this page. The intern will be required to enter additional information, specific only to him .
* **intern\_edit\_profile.xhtml**: When an intern has already created his profile and he enters the system, this page allows him to make changes to the existing profile.
* **intern\_profile.xhtml (Profile Page):** This page will show user the information about account.
* **intern\_docs.xhtml (Documents Page):** In this page, intern will be able to see the documents uploaded, delete them and upload new documents.
* **intern\_events.xhtml (Events Page):** Using this page intern will be able to see the events arranged by company.
* **intern\_tasks.xhtml (Tasks Page):** In this page, intern will be able to see tasks given by his/her supervisor and by clicking checkbox next to a task, s/he can notify the supervisor – task is done.
* **intern\_scheduler.xhtml (Schedule Page):** In this page, interns will be able to add events to their schedule, also events created by company will be added on the schedule of all users by the system.
* **supervisor\_profile.xhtml (Profile Page):** This page will show user information about account.
* **supervisor\_scheduler.xhtml (Schedule Page):** In this page, supervisors will be able to add events to their schedule, also events created by company will be added on the schedule of all users by the system.
* **supervisor\_individual\_interns.xhtml (Individuals Page):** In this page, supervisors will be able to see the information of their individual interns. In this page they can create teams. Also they can assign tasks to interns in this page.
* **edit\_supervisor\_profile.xhtml (Initial User Information Page):** if a supervisor entered the system for the first time, i.e. registered to the system for the first time, or clicked edit profile button, he will encounter with this page. The supervisor will be required to enter additional information, specific only to him .
* **supervisor\_edit\_profile.xhtml**: When a supervisor has already created his profile and he enters the system, this page allows him to make changes to the existing profile.
* **supervisor\_teams.xhtml (Teams Page):** This page is to control and assign tasks to teams by supervisors.
* **supervisor\_events.xhtml (Events Page):** Using this page the supervisor will be able to see the events arranged by company. He can see the events shared with everyone or the ones shared only with supervisors.
* **supervisor\_waiting\_tasks.xhtml (Waiting Tasks Page):** In this .xhtml page supervisor will see the notifications sent by interns about finishing their tasks. When it is approved by the supervisor, task will be done.
* **company\_profile.xhtml** : In this page, company will be able to see the information of their supervisors and interns and arrange the matching between the available supervisors and interns. It can also create events which will be visible in the profile of all the users.
* **company\_intern\_list.**xhtml: In this page the company will be able to see the interns list together with their status, if they are matched or unmatched.
* **company\_supervisors\_list.xhtml** In this page the company will be able to see the list of all the supervisors, the number of their current interns and if they can accept more interns or not.
* **company\_match.xhtml** In this page company will be able to match the interns with a suitable supervisor. It will see the unmatched interns and the supervisors who have available quota to accept more interns and will arrange a match between them, assigning the intern to a specific supervisor.
* **company\_events.xhtml** In this page company will create events which will be visible in the schedules of all the other users. The company may also decide to show certain events to only a specific category of users (example: only supervisors and not interns). In this case the event would only be visible in the schedule of the chosen user.
* **edit\_company\_profile.xhtml (Initial User Information Page):** if a company entered the system for the first time, i.e. registered to the system for the first time, or clicked edit profile button, he will encounter with this page. The company will be required to enter additional information, specific only to him.
* **company\_edit\_profile.xhtml**: When a company has already created his profile and he enters the system, this page allows him to make changes to the existing profile.

**Template XHTML Files**

* **header.xhtml:** All the view pages in the system will have this file. This file contains logo and name of the project on top of each page.
* **menubar.xhtml:** All the view pages except "sign\_up.xhtml", "sign\_in.xhtml" and "free\_intern.xhtml" will have this file. This XHTML file contains two XHTML files inside it.
* **navigation.xhtml:** This file is the main part of menubar, it shows user to navigation options in systems according to user type.
* **loginControl.xhtml:** This file contains logout button in necessary pages.

# Glossary & References

1. Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.